

Computational Topology (II): Persistence Theory

- 1. Algorithmic Foundation**
- 2. Algebraic Foundation**
- 3. Stability Theorem**

Computational Topology (II): Persistence Theory

1. Algorithmic Foundation

2. Algebraic Foundation

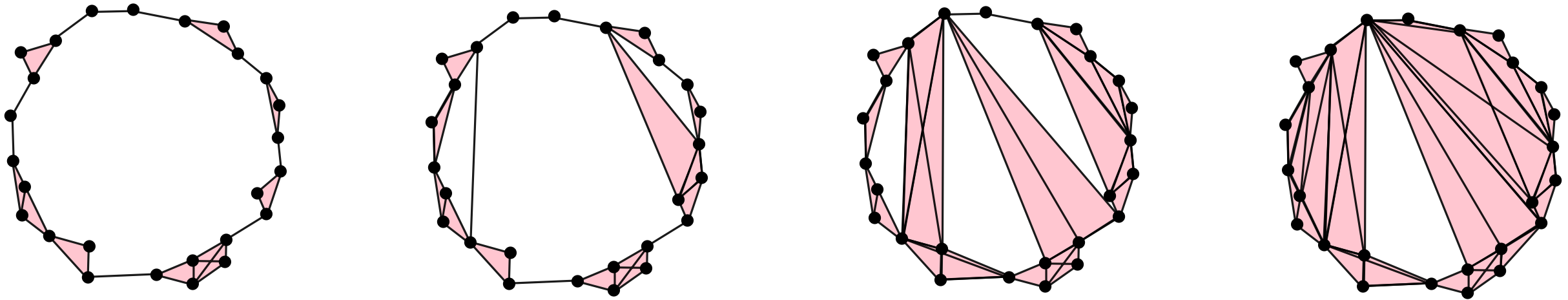
3. Stability Theorem

Computation with filtrations and matrix reduction

Algorithms for computing the homology groups of a simplicial complex work by *decomposing* it with a so-called *filtration*.

Computation with filtrations and matrix reduction

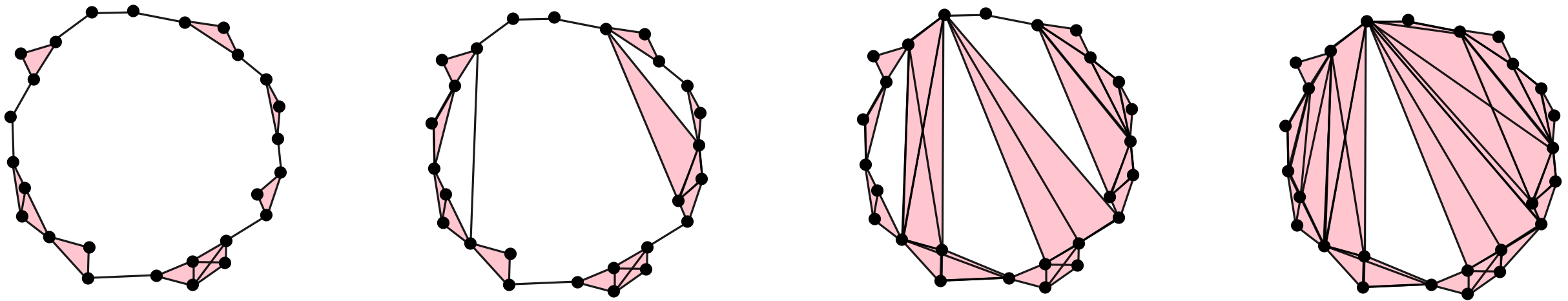
Algorithms for computing the homology groups of a simplicial complex work by *decomposing* it with a so-called *filtration*.



Def: A **filtered simplicial complex** \mathcal{S} is a family $\{S_u\}_{u \in \mathbb{R}}$ of subcomplexes of some fixed simplicial complex S s.t. $S_a \subseteq S_b$ for any $a \leq b$.

Computation with filtrations and matrix reduction

Algorithms for computing the homology groups of a simplicial complex work by *decomposing* it with a so-called *filtration*.



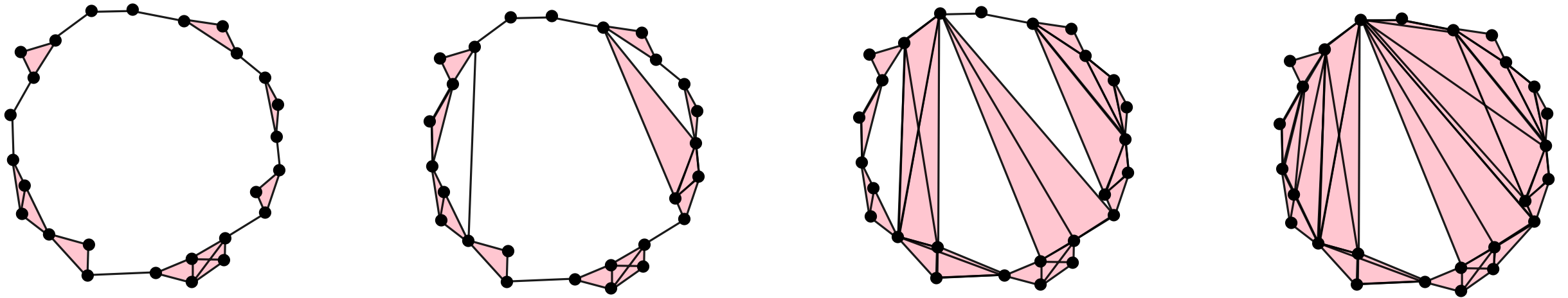
Def: A **filtered simplicial complex** \mathcal{S} is a family $\{S_u\}_{u \in \mathbb{R}}$ of subcomplexes of some fixed simplicial complex S s.t. $S_a \subseteq S_b$ for any $a \leq b$.

Def: Let f be a real valued function defined on the vertices of K . For $\sigma = [v_0, \dots, v_k] \in K$, let $f(\sigma) = \max_{i=0, \dots, k} f(v_i)$, and order the simplices of K in increasing order w.r.t. the function f values (and break ties with dimension in case some simplices have the same function value).

Q: Show that this is a filtration.

Computation with filtrations and matrix reduction

Algorithms for computing the homology groups of a simplicial complex work by *decomposing* it with a so-called *filtration*.



Def: A **filtered simplicial complex** \mathcal{S} is a family $\{S_u\}_{u \in \mathbb{R}}$ of subcomplexes of some fixed simplicial complex S s.t. $S_a \subseteq S_b$ for any $a \leq b$.

For a given simplicial complex, one can study filtrations $\{S_i\}_{i \in \mathbb{R}}$ such that $S_{i+1} = S_i \cup \{\sigma\}$, i.e., simplices are added one at a time. This allows for an efficient practical method.

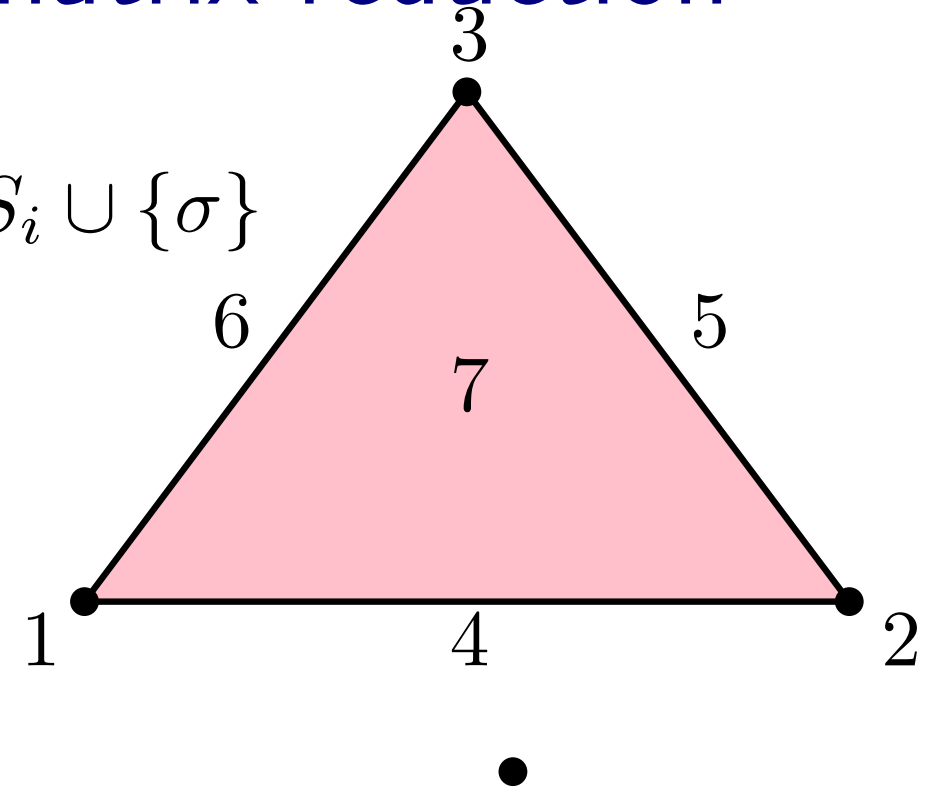
Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$

Indeed, homology can be computed by using the fact that each simplex is either:

positive, i.e., it creates a new homology class

negative, i.e., it destroys an homology class



•
1

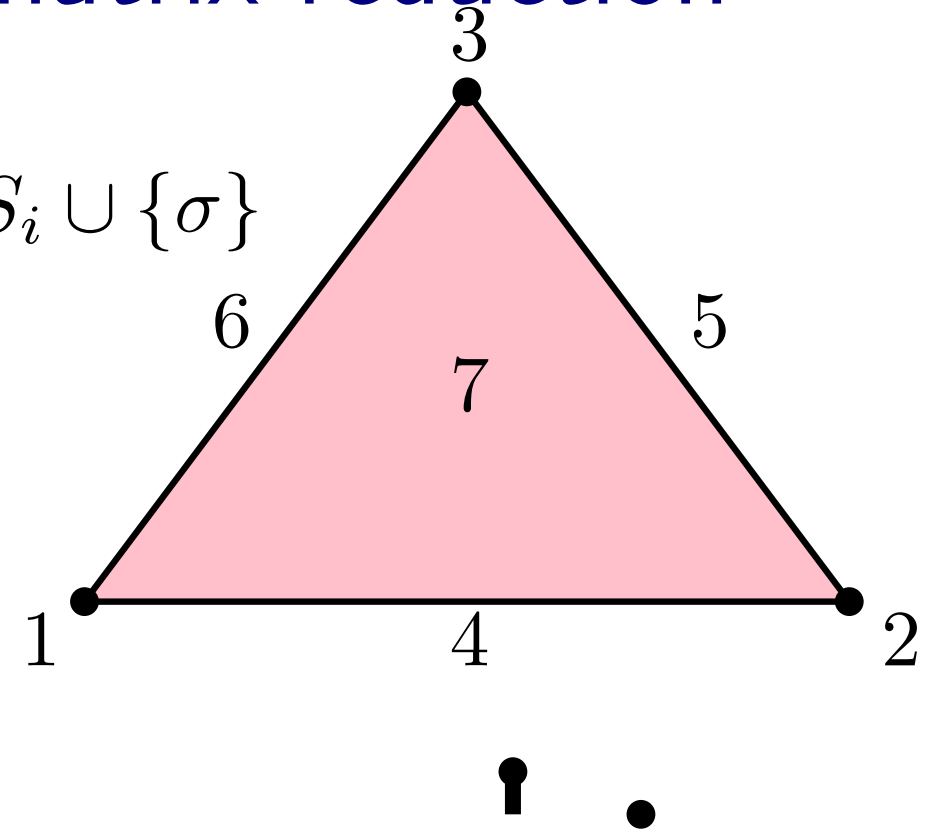
Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$

Indeed, homology can be computed by using the fact that each simplex is either:

positive, i.e., it creates a new homology class

negative, i.e., it destroys an homology class



•
1

•
1

•
2

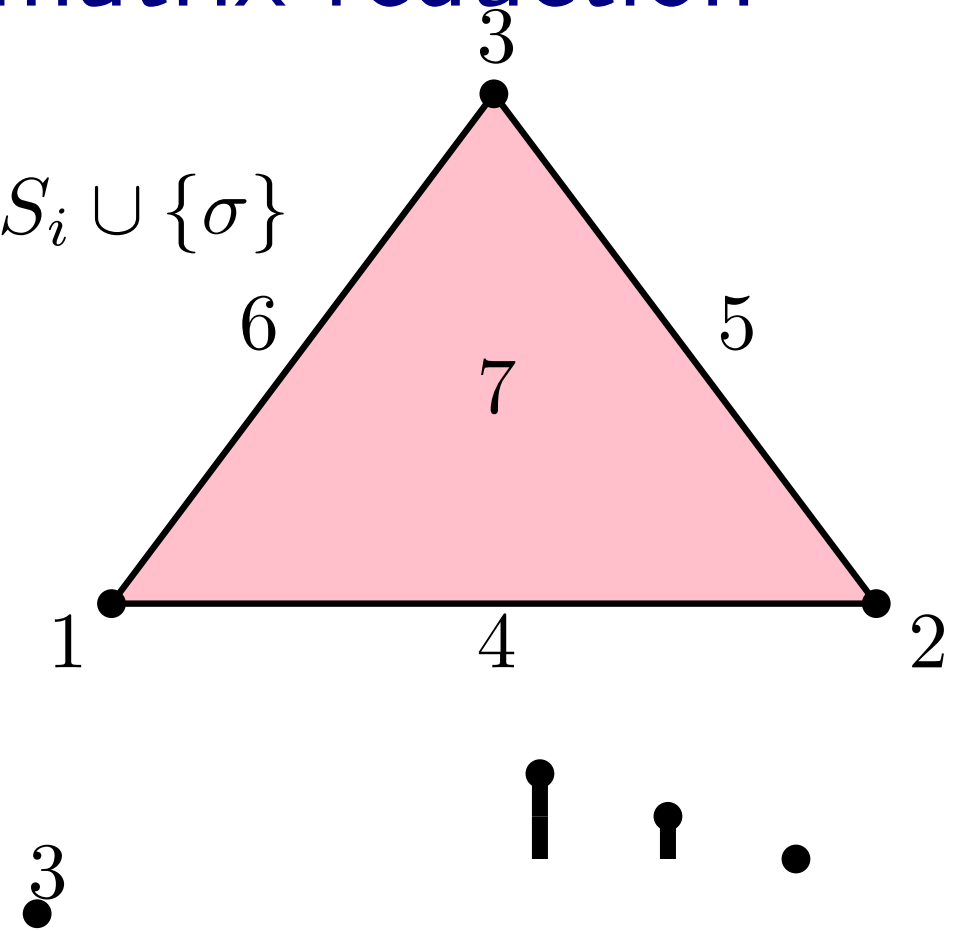
Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$

Indeed, homology can be computed by using the fact that each simplex is either:

positive, i.e., it creates a new homology class

negative, i.e., it destroys an homology class



1

1

2

1

2

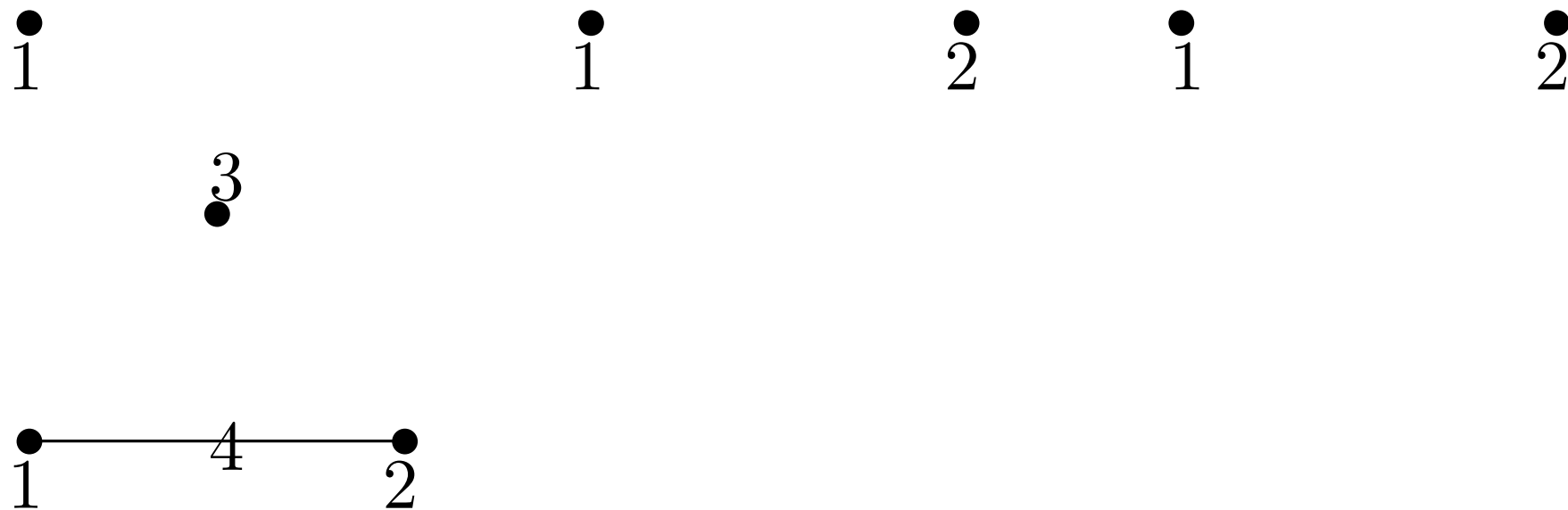
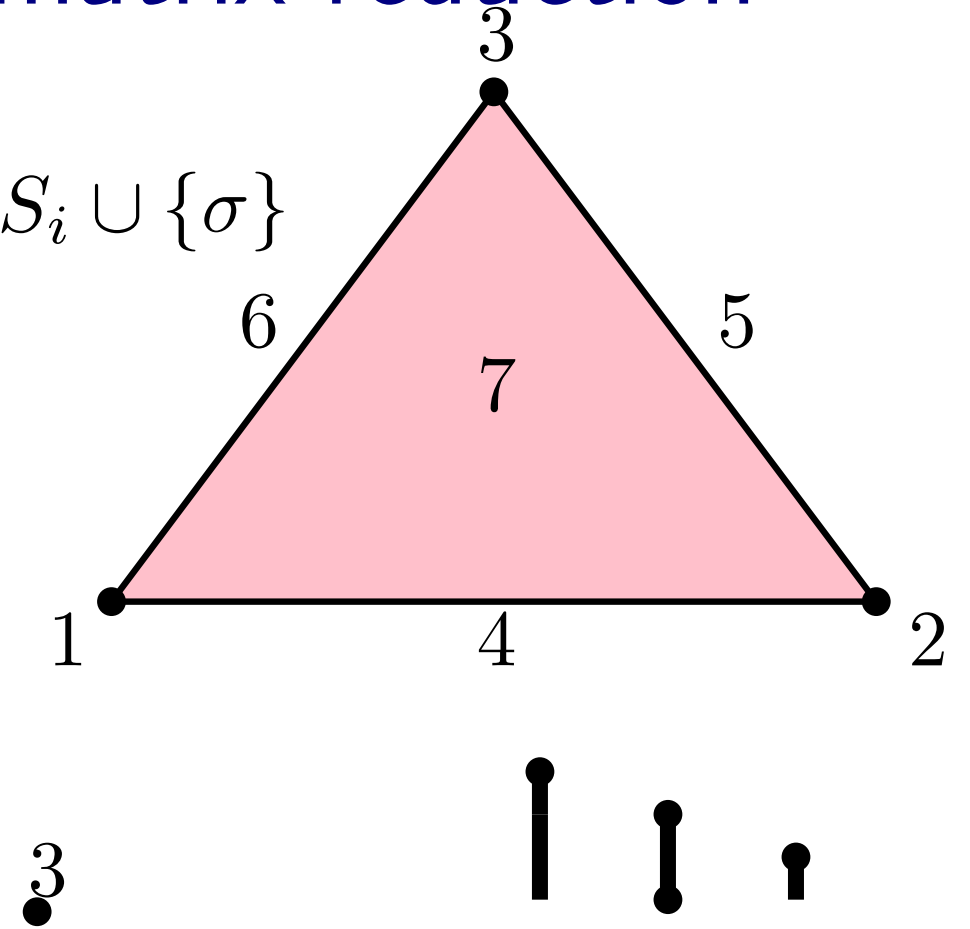
Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$

Indeed, homology can be computed by using the fact that each simplex is either:

positive, i.e., it creates a new homology class

negative, i.e., it destroys an homology class



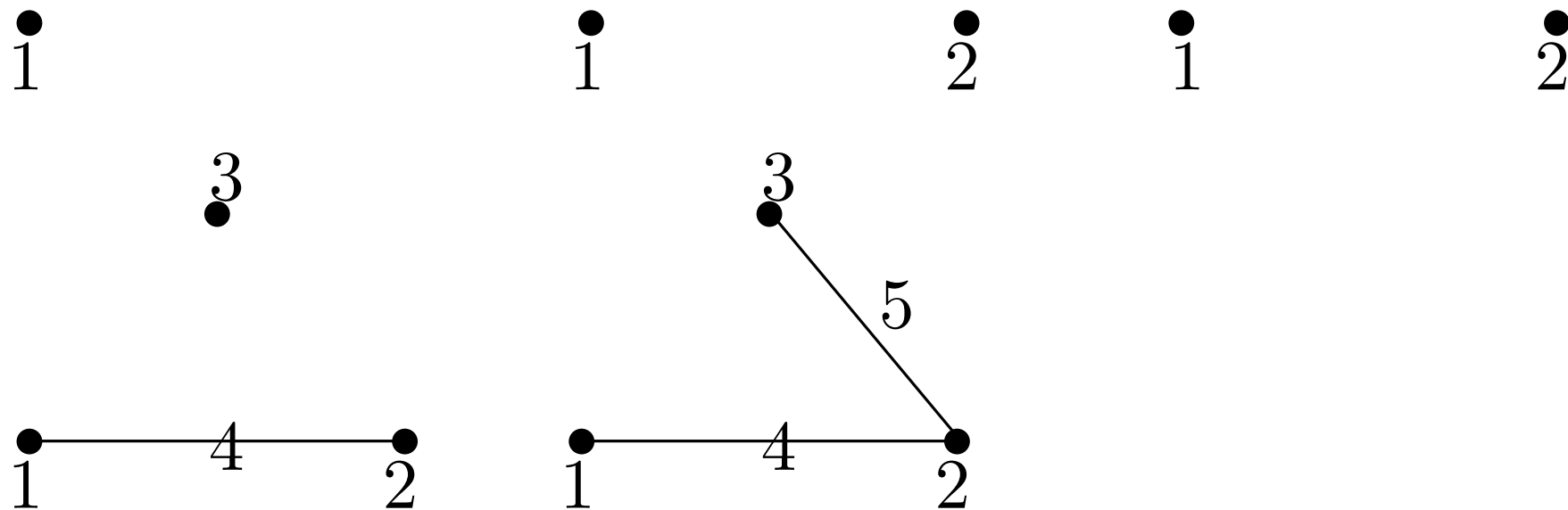
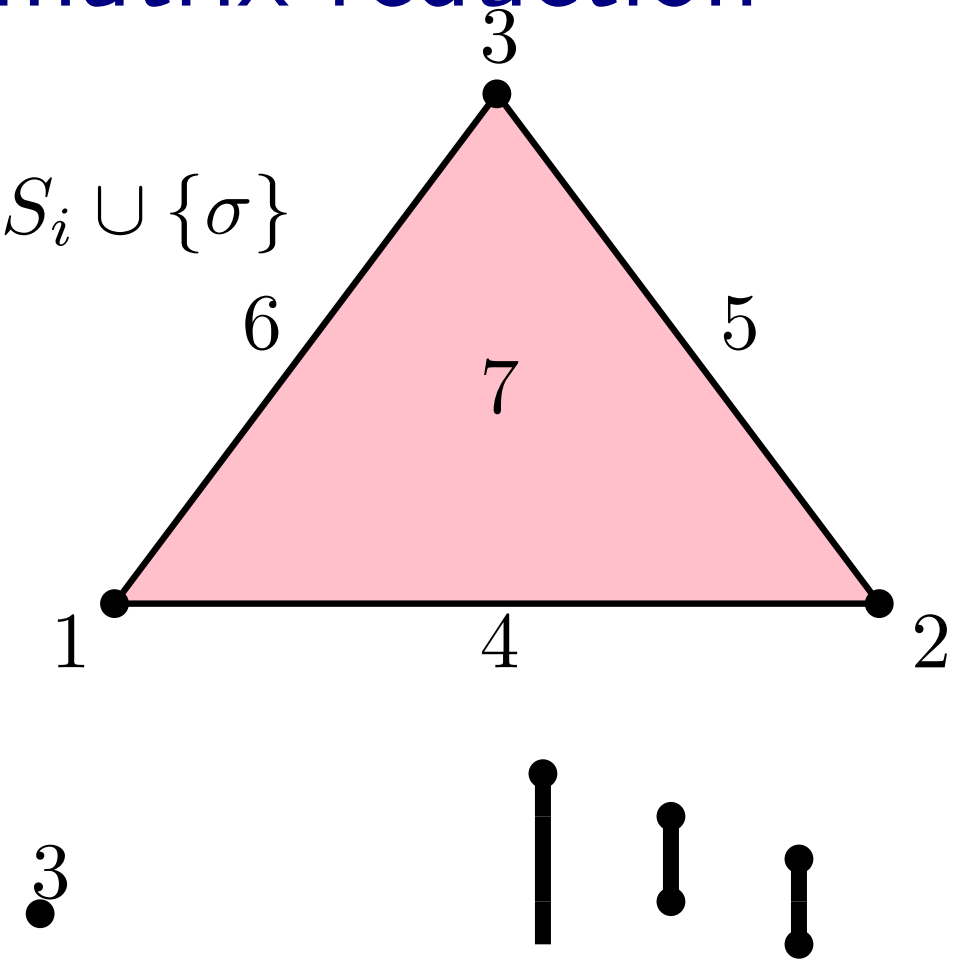
Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$

Indeed, homology can be computed by using the fact that each simplex is either:

positive, i.e., it creates a new homology class

negative, i.e., it destroys an homology class



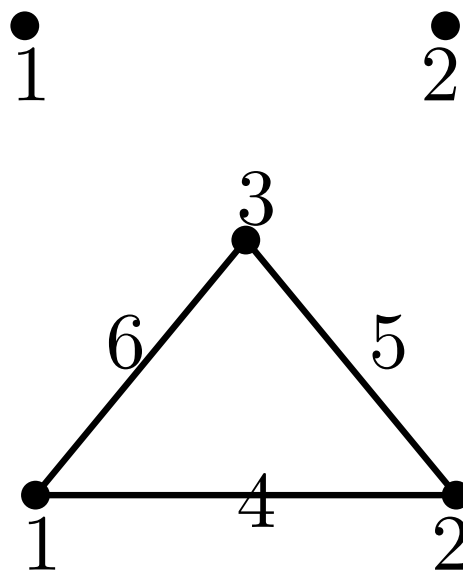
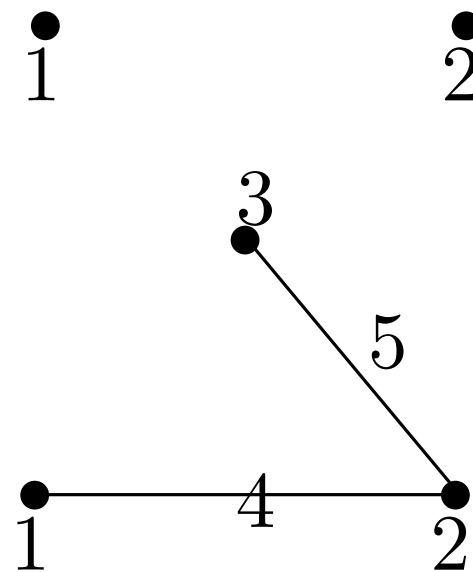
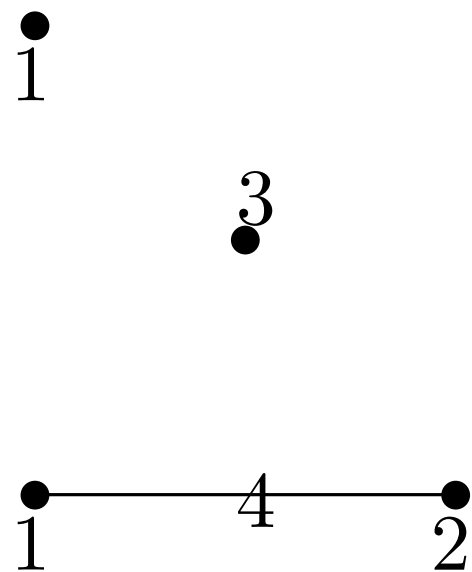
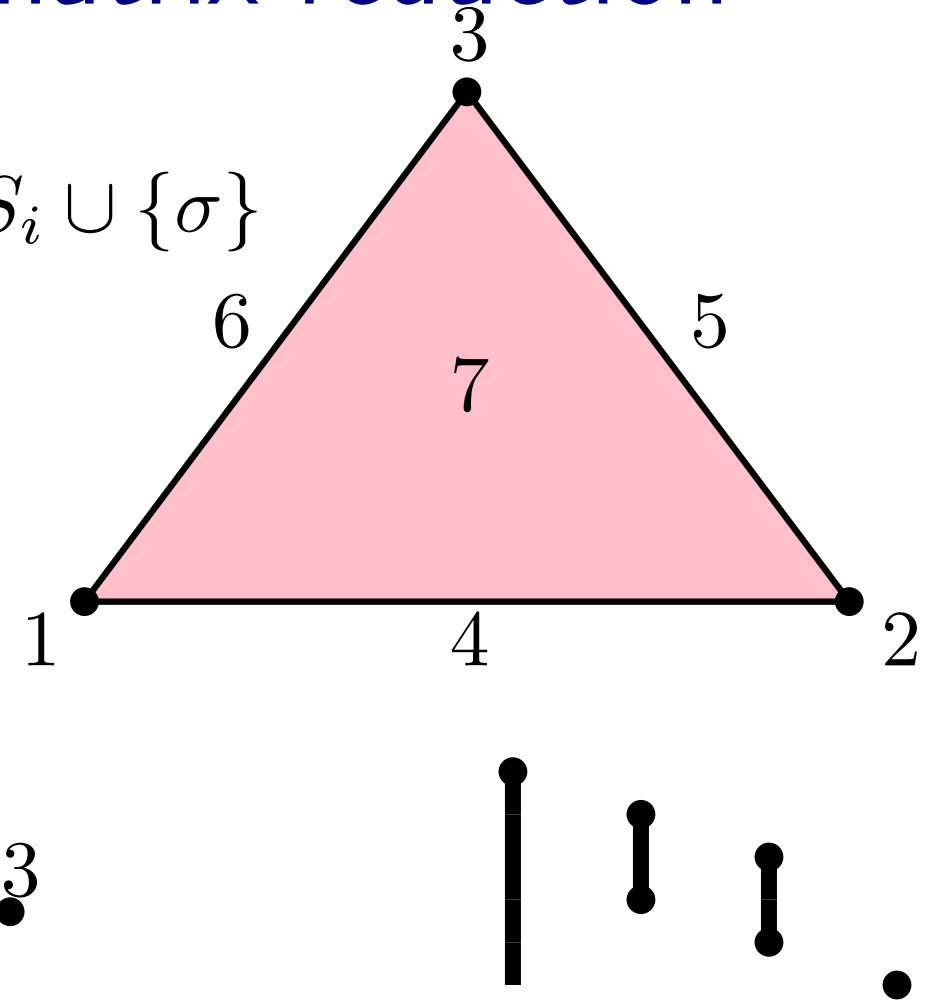
Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$

Indeed, homology can be computed by using the fact that each simplex is either:

positive, i.e., it creates a new homology class

negative, i.e., it destroys an homology class



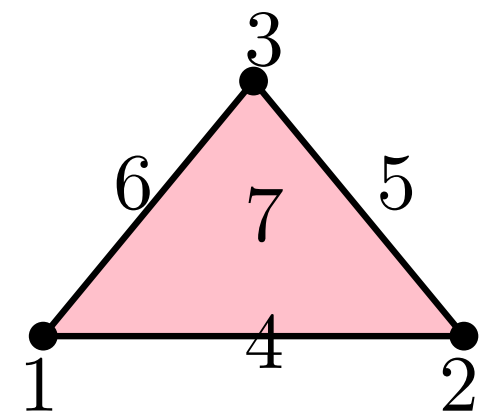
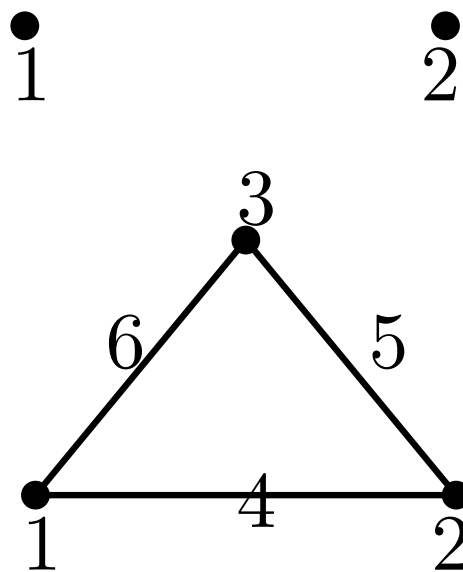
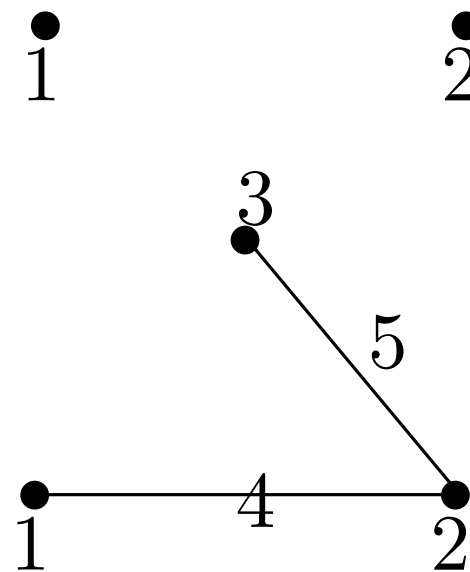
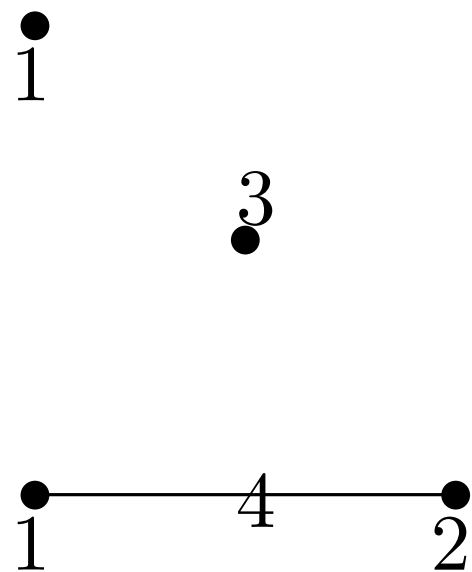
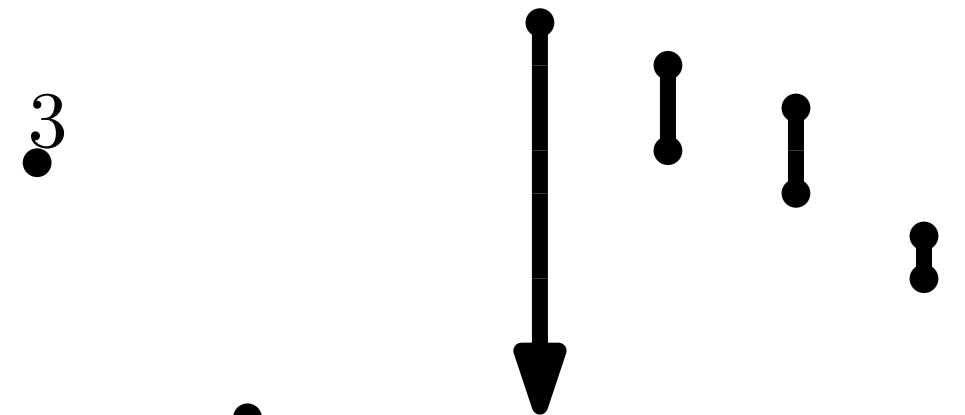
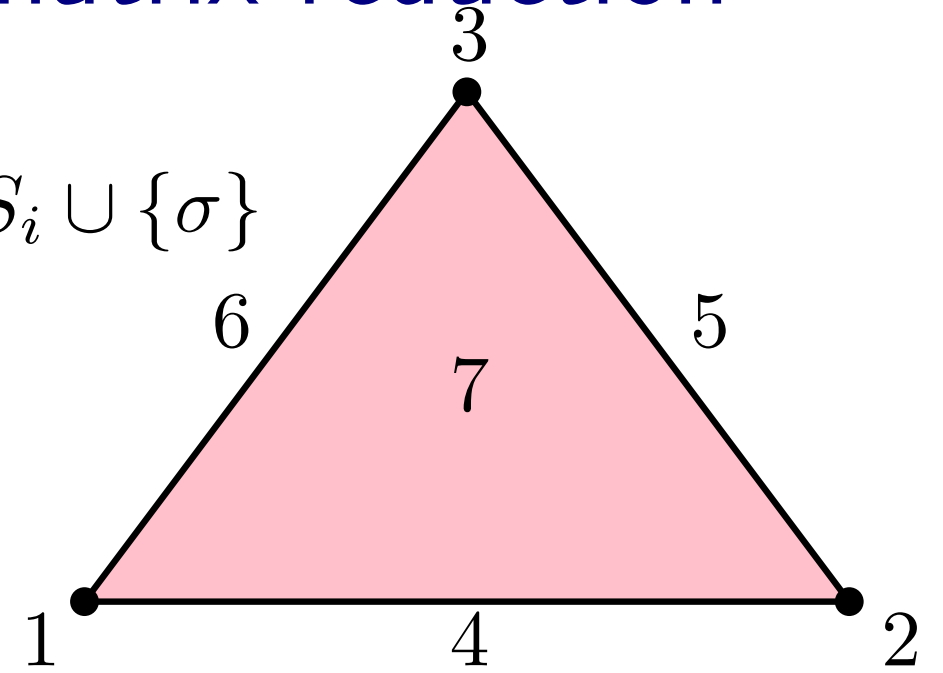
Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$

Indeed, homology can be computed by using the fact that each simplex is either:

positive, i.e., it creates a new homology class

negative, i.e., it destroys an homology class



Computation with filtrations and matrix reduction

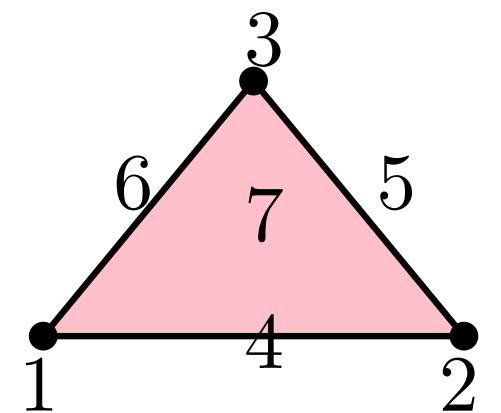
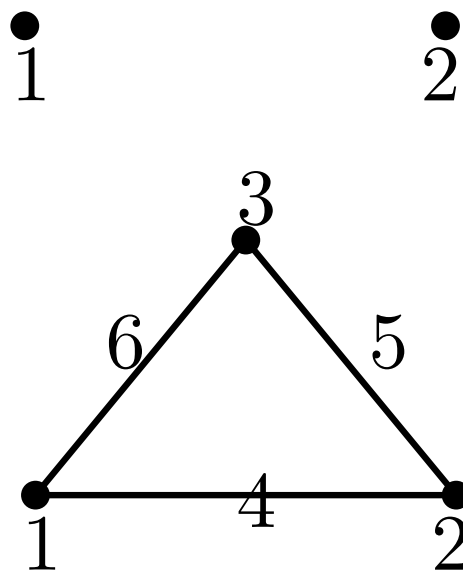
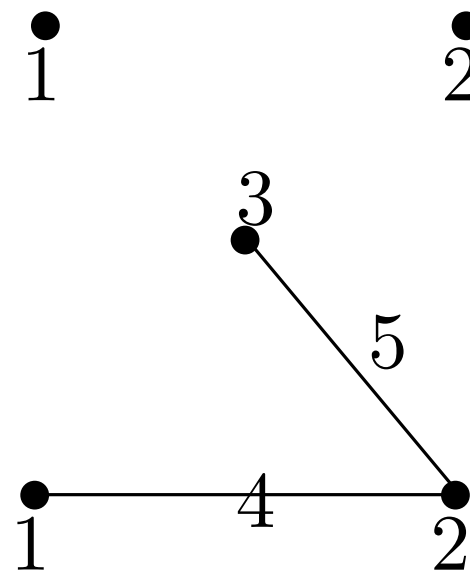
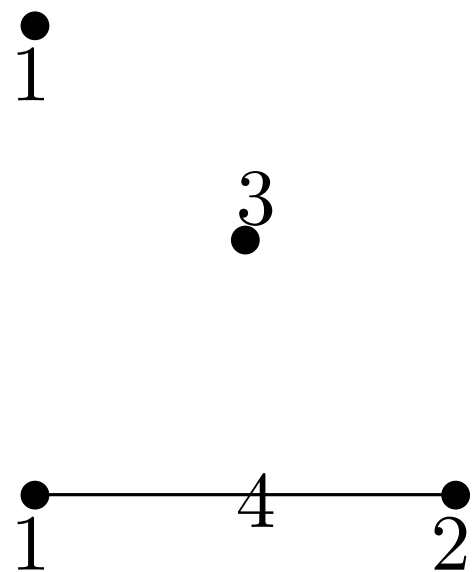
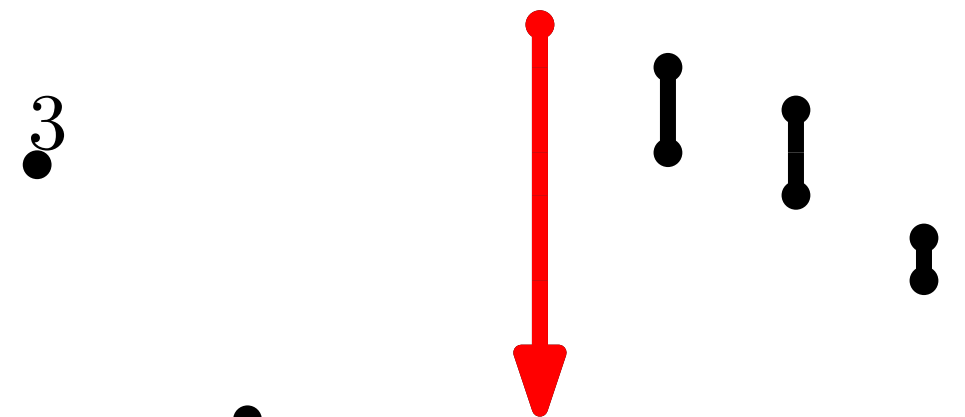
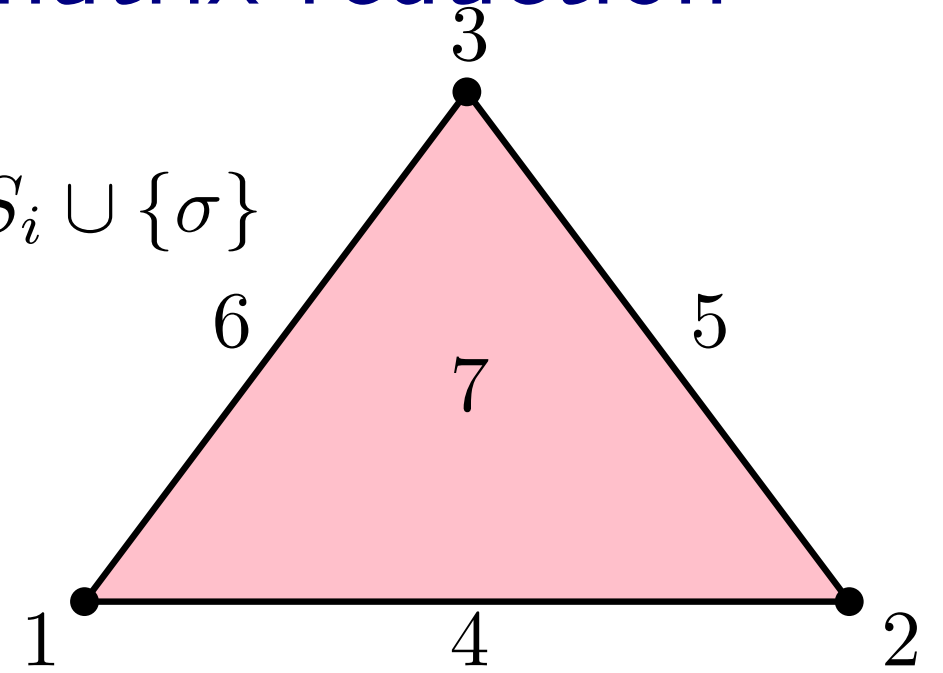
Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$

Indeed, homology can be computed by using the fact that each simplex is either:

positive, i.e., it creates a new homology class

negative, i.e., it destroys an homology class

The Betti number is equal to the number of bars that are still alive when the full complex is reached in the filtration



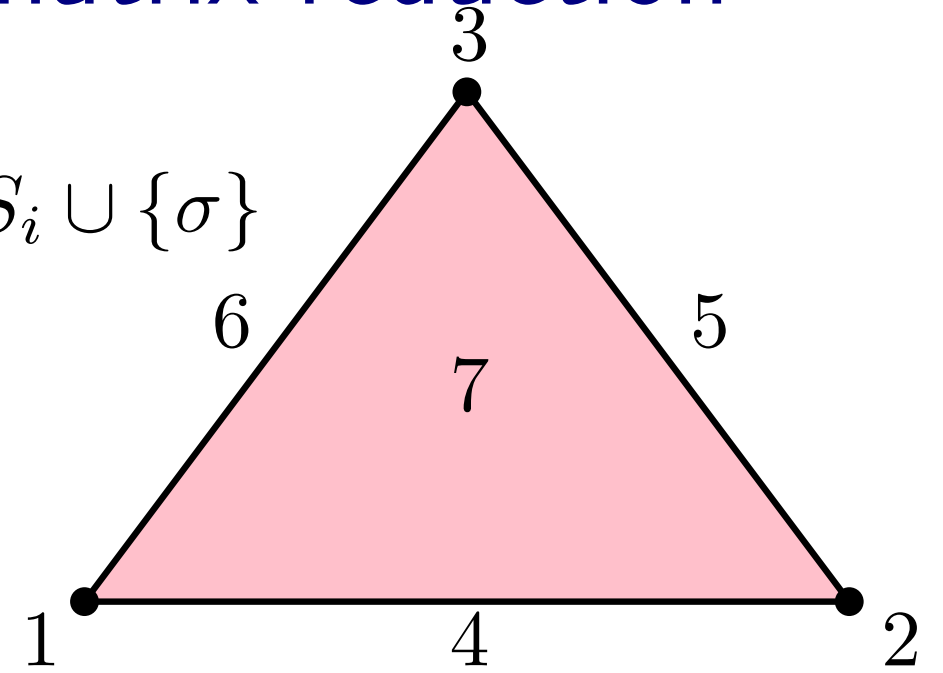
Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$

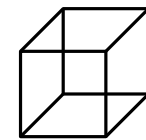
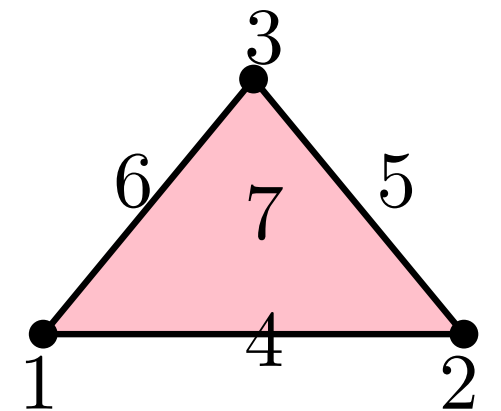
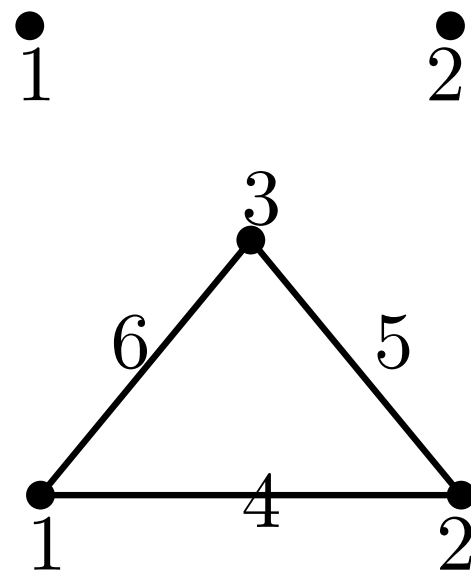
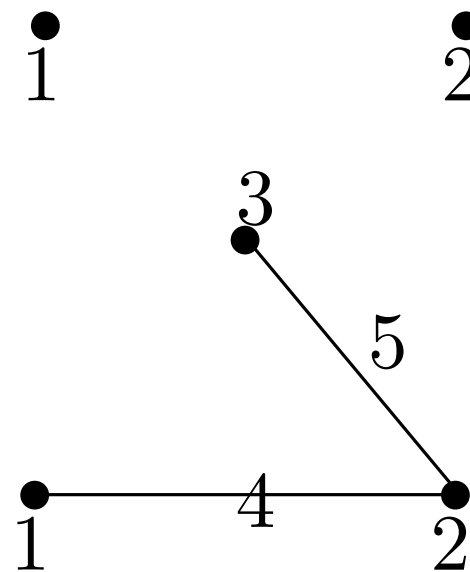
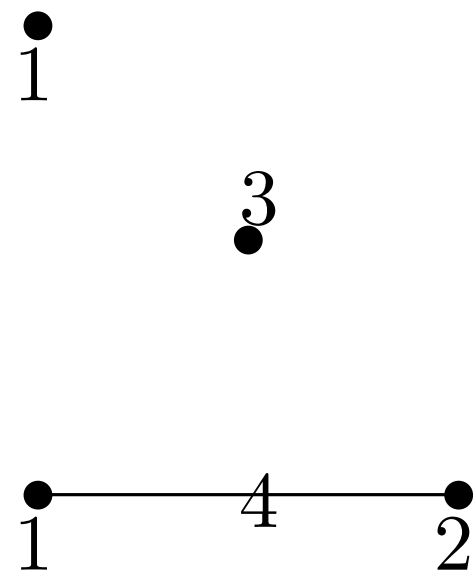
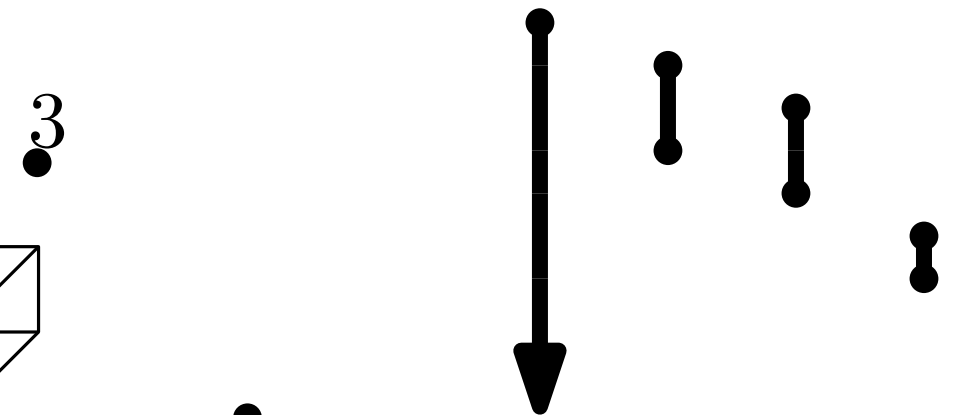
Indeed, homology can be computed by using the fact that each simplex is either:

positive, i.e., it creates a new homology class

negative, i.e., it destroys an homology class



Q: Do the same for the homology of the cube.



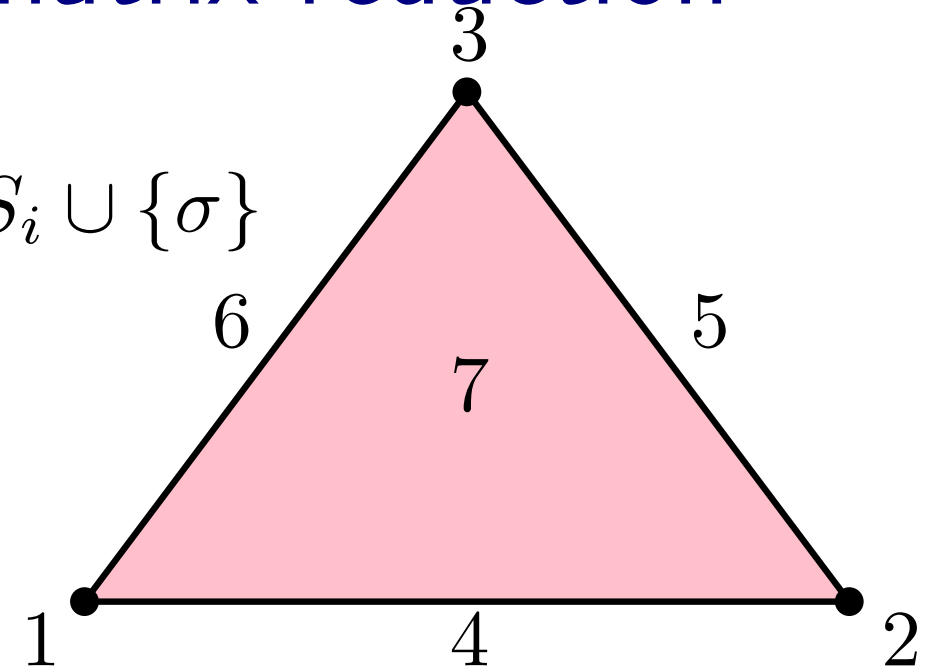
Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$

Indeed, homology can be computed by using the fact that each simplex is either:

positive, i.e., it creates a new homology class

negative, i.e., it destroys an homology class



In order to decide whether an inserted simplex is positive or negative with an algorithm, one can reduce its boundary by adding the boundaries of previous simplices, until the simplex boundary cannot be reduced anymore.

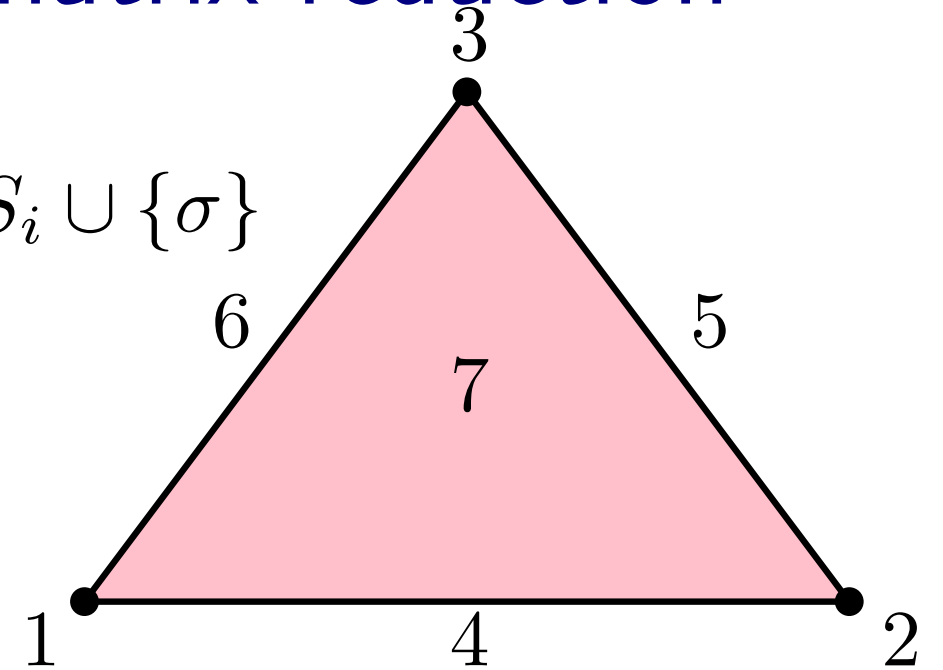
Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$

Indeed, homology can be computed by using the fact that each simplex is either:

positive, i.e., it creates a new homology class

negative, i.e., it destroys an homology class



In order to decide whether an inserted simplex is positive or negative with an algorithm, one can reduce its boundary by adding the boundaries of previous simplices, until the simplex boundary cannot be reduced anymore.

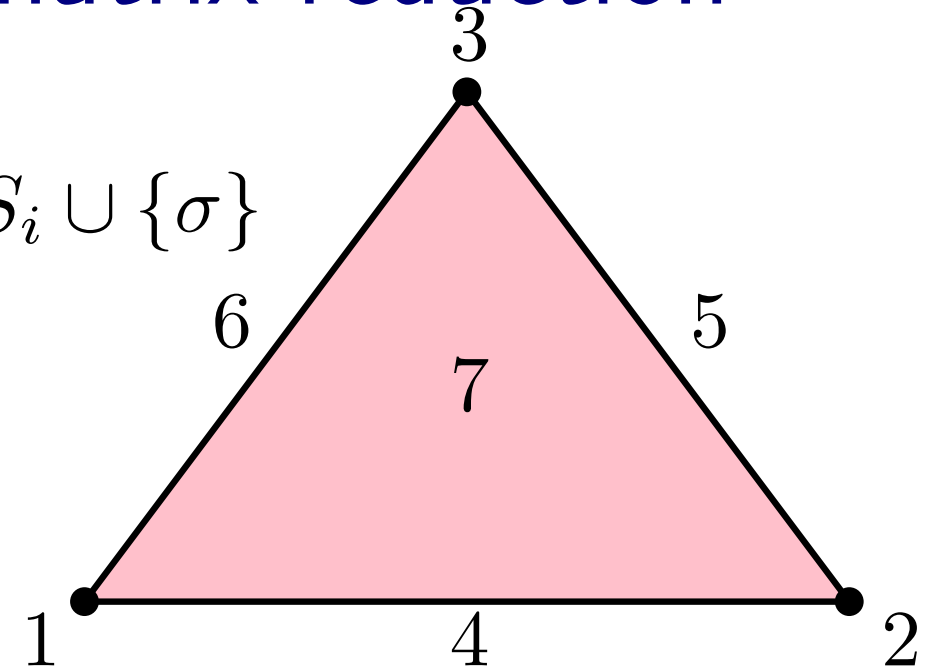
- If the reduced boundary is zero, the inserted simplex is positive.
- If the reduced boundary is not zero, the inserted simplex is negative.

Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$

Indeed, homology can be computed by using the fact that each simplex is either:

- positive*, i.e., it creates a new homology class
- negative*, i.e., it destroys an homology class



In order to decide whether an inserted simplex is positive or negative with an algorithm, one can reduce its boundary by adding the boundaries of previous simplices, until the simplex boundary cannot be reduced anymore.

- If the reduced boundary is zero, the inserted simplex is positive.
- If the reduced boundary is not zero, the inserted simplex is negative.

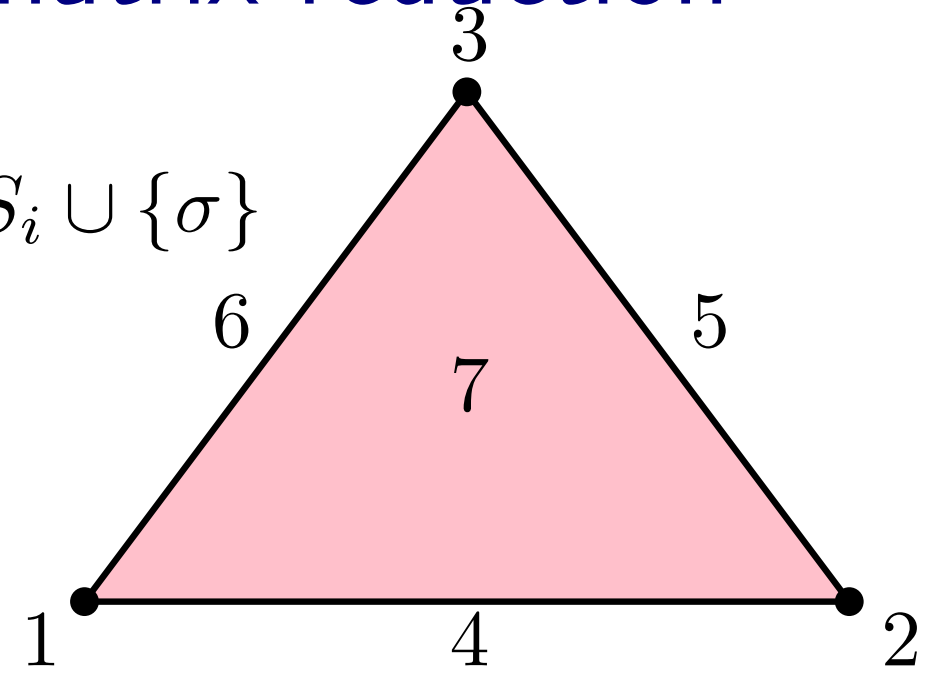
In practice, this amounts to applying a change of basis to the canonical chain basis, so that each element of the new chain basis is either an element of the cycle basis or an element of the boundary basis.

$$Z_d \simeq \ker(\partial_d) \oplus \text{coim}(\partial_d)$$

Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$
given as *boundary matrix*

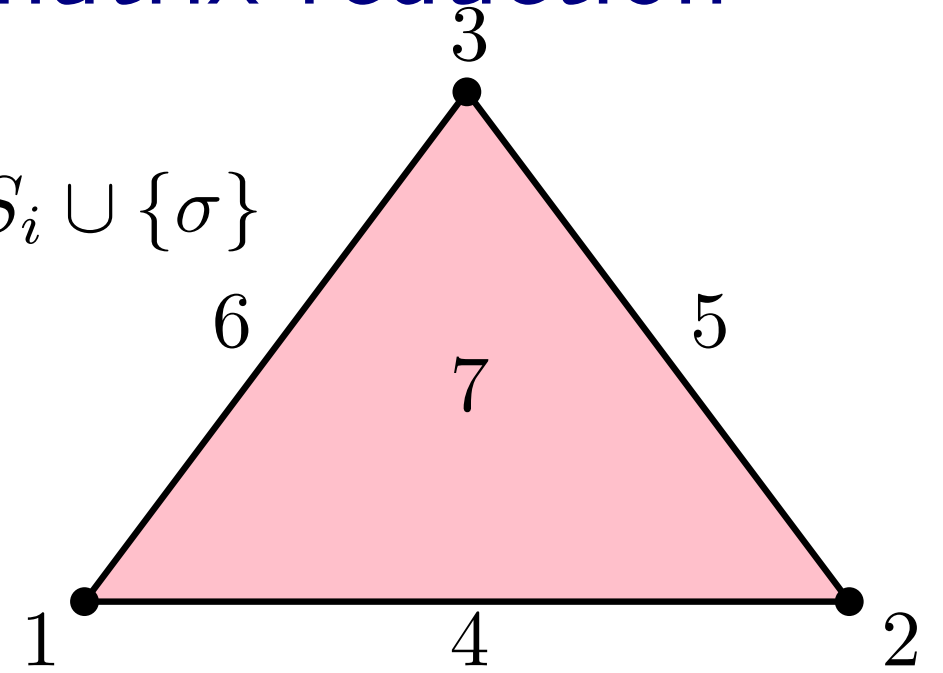
	1	2	3	4	5	6	7
1							
2							
3							
4							
5							
6							
7							



Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$
given as *boundary matrix*

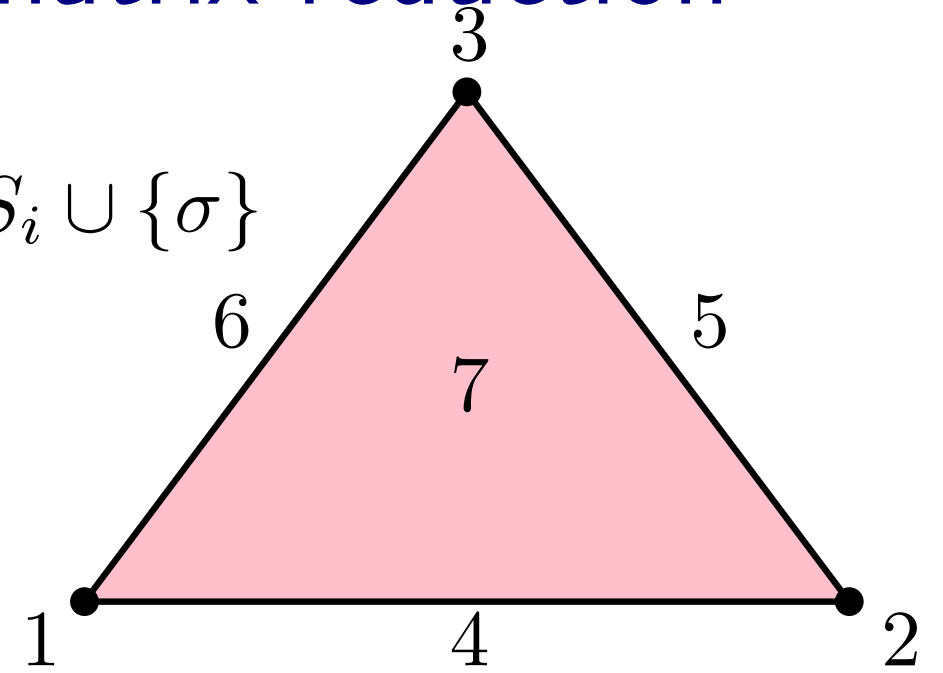
	1	2	3	4	5	6	7
1				•			
2				•			
3							
4							
5							
6							
7							



Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$
 given as *boundary matrix*

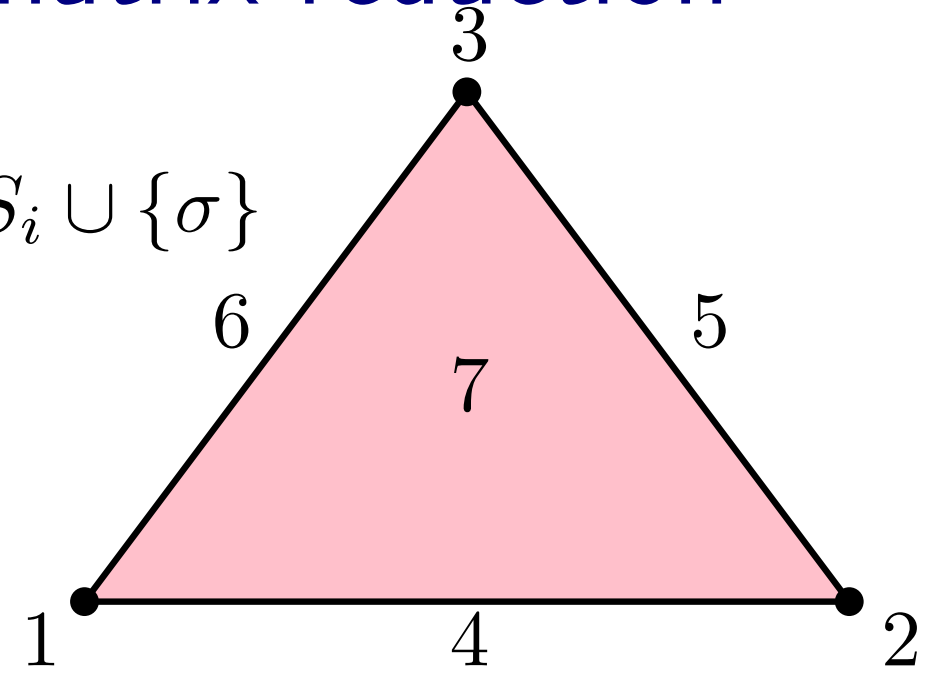
	1	2	3	4	5	6	7
1				•			
2				•	•		
3					•		
4							
5							
6							
7							



Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$
 given as *boundary matrix*

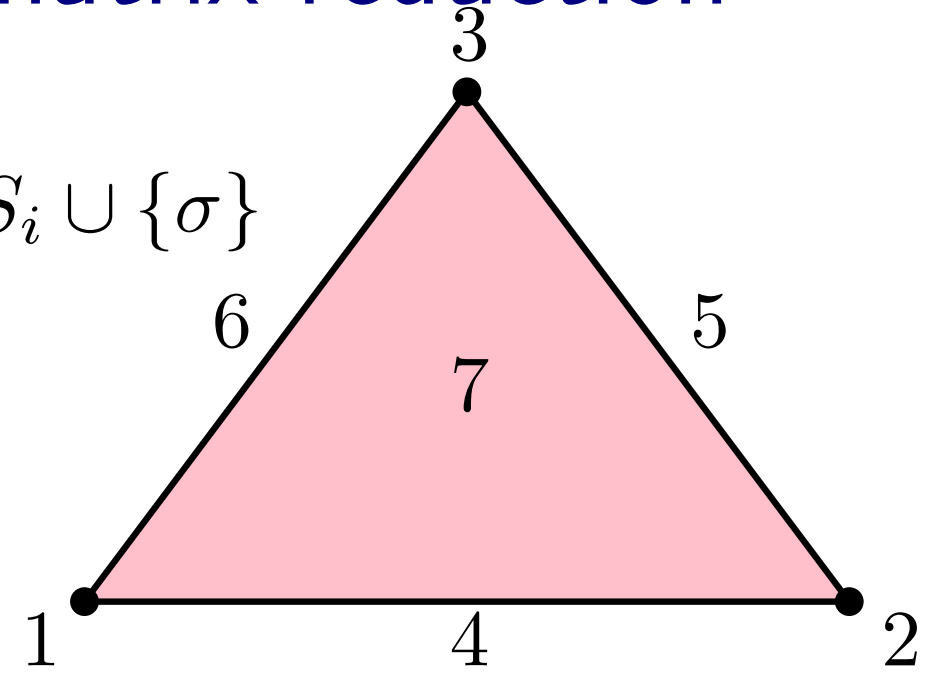
	1	2	3	4	5	6	7
1				•		•	
2				•	•		
3					•	•	
4							
5							
6							
7							



Computation with filtrations and matrix reduction

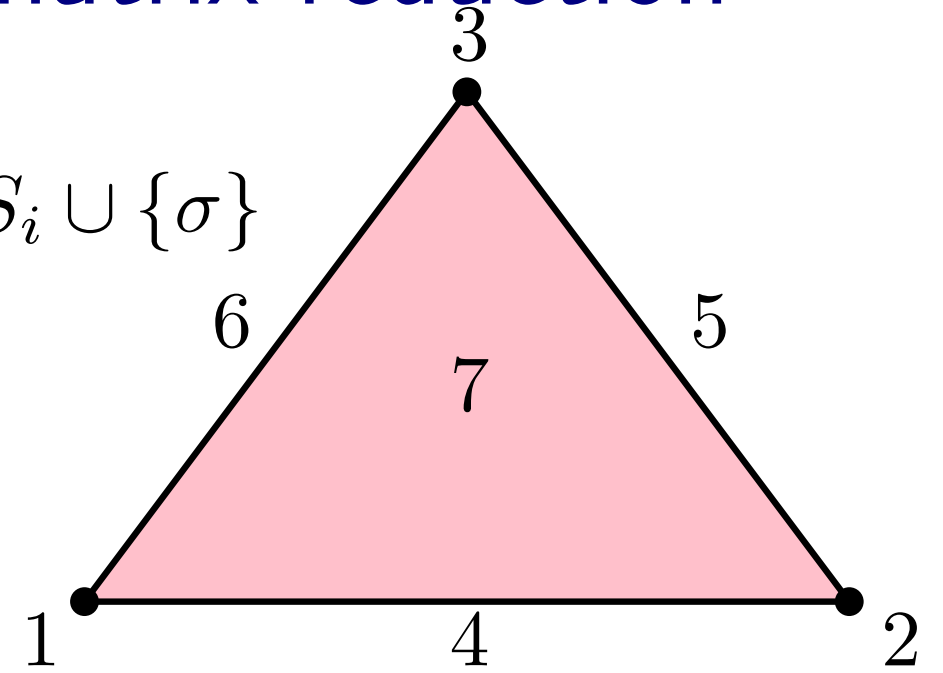
Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$
 given as *boundary matrix*

	1	2	3	4	5	6	7
1				•		•	
2				•	•		
3					•	•	
4							•
5							•
6							•
7							



Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$
 given as *boundary matrix*



	1	2	3	4	5	6	7
1				•		•	
2				•	•		
3					•	•	
4							•
5							•
6							•
7							

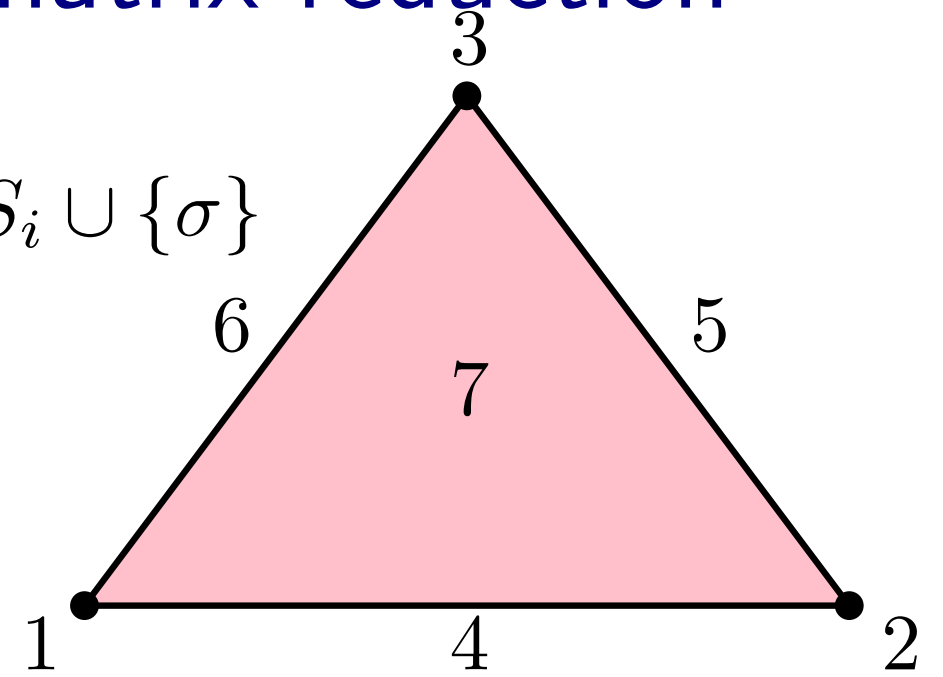
for $j=1$ to m do:

 while $\exists k < j$ s.t. $\text{low}(k) == \text{low}(j)$ do:

$\text{col}(j) = \text{col}(j) + \text{col}(k)$

Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$
 given as *boundary matrix*



\dashrightarrow \setminus $/$

	1	2	3	4	5	6	7
1				•		•	
2				•	•		
3					•	•	
4							•
5							•
6							•
7							

For every inserted simplex

for $j=1$ to m **do:**

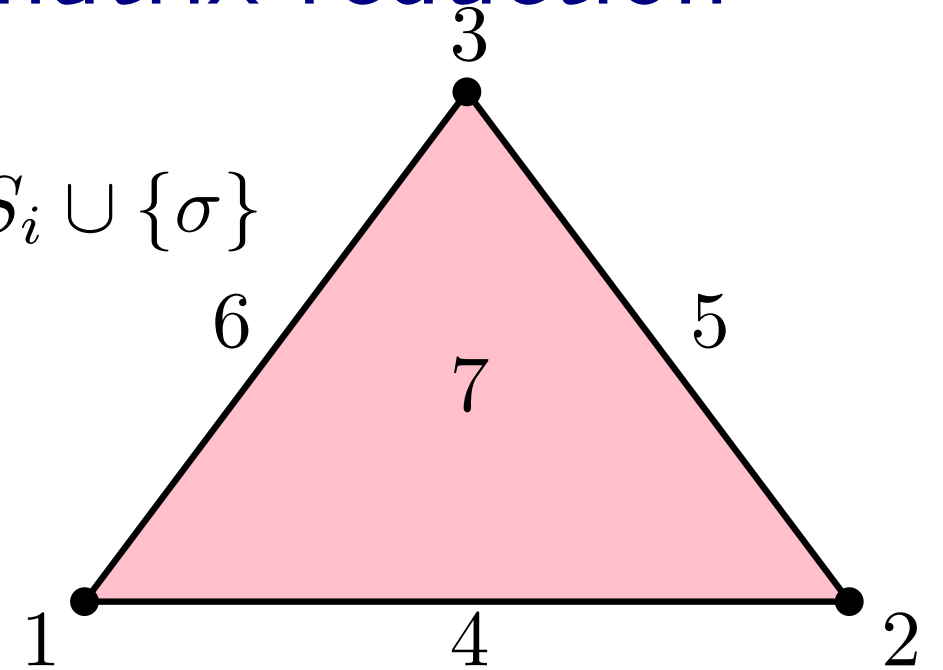
While the boundary of the inserted simplex
 can be reduced with a change of basis...

while $\exists k < j$ s.t. $\text{low}(k) == \text{low}(j)$ **do:**

$\text{col}(j) = \text{col}(j) + \text{col}(k)$...reduce the boundary

Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$
 given as *boundary matrix*



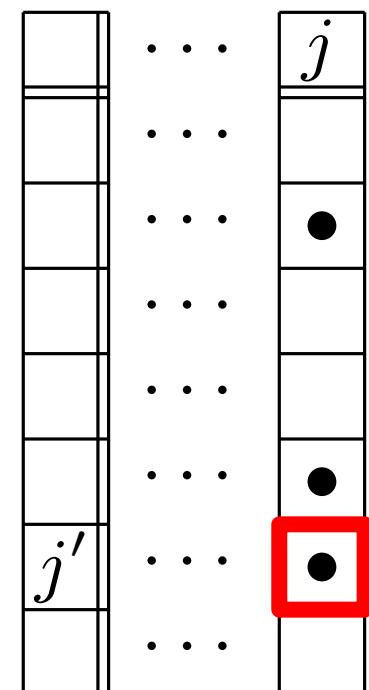
	1	2	3	4	5	6	7
1				•		•	
2				•	•		
3					•	•	
4							•
5							•
6							•
7							

for $j=1$ to m do:

while $\exists k < j$ s.t. $\text{low}(k) == \text{low}(j)$ do:

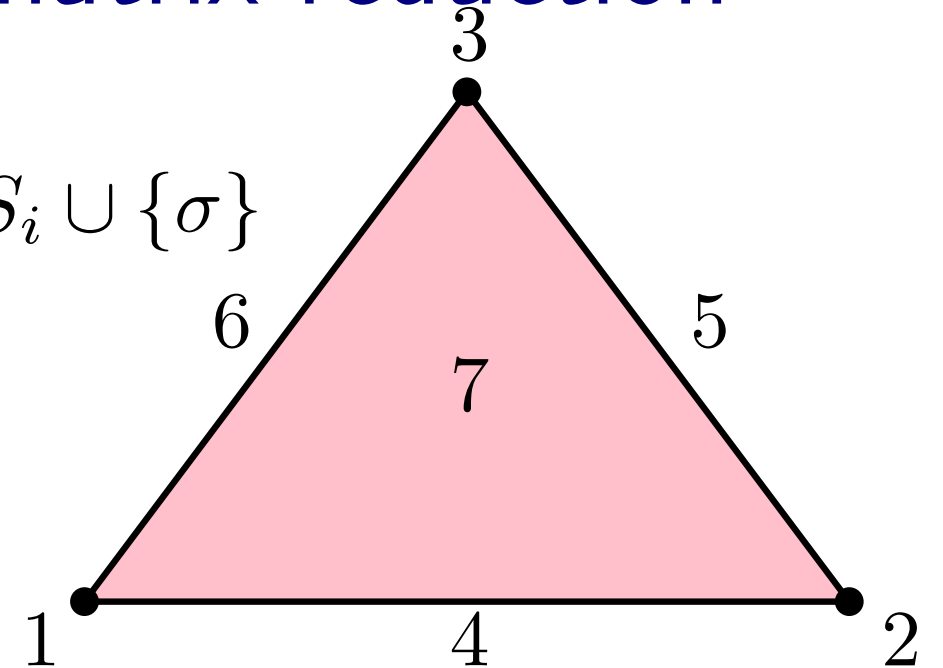
$$\text{col}(j) = \text{col}(j) + \text{col}(k)$$

$$\text{low}(j) = j'$$



Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$
 given as *boundary matrix*



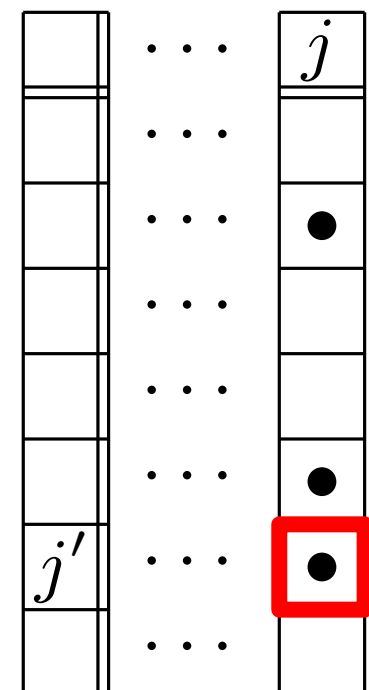
← \ /

	1	2	3	4	5	6	7
1				•		•	
2				•	•		
3				•	•		
4							•
5							•
6							•
7							

for $j=1$ to m do:

while $\exists k < j$ s.t. $\text{low}(k) == \text{low}(j)$ do:

$$\text{col}(j) = \text{col}(j) + \text{col}(k)$$



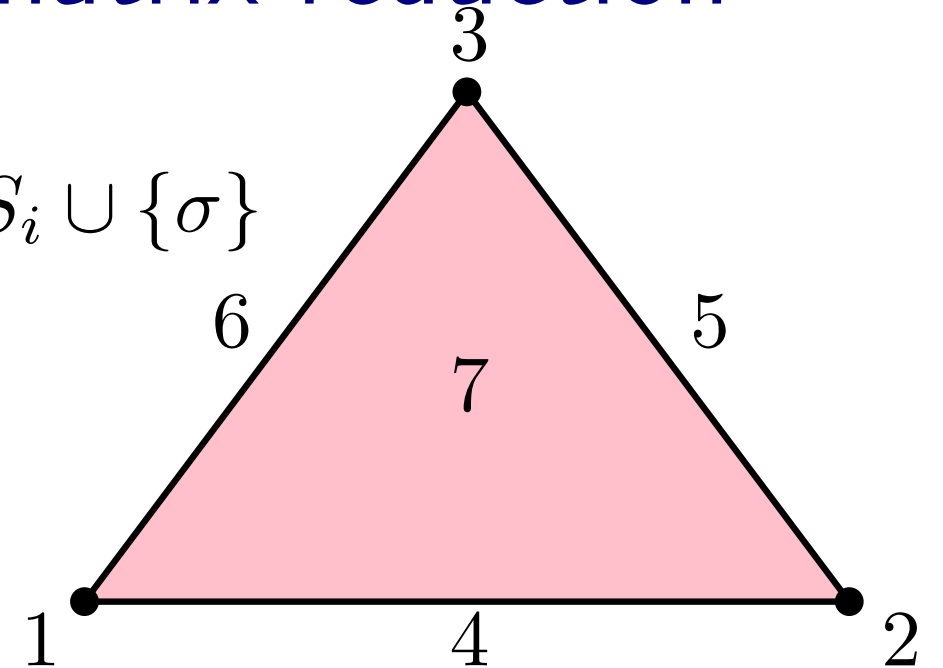
	5	6
1		•
2	•	
3	•	•

$$6 = 6 + 5$$

$$\text{low}(j) = j'$$

Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$
 given as *boundary matrix*



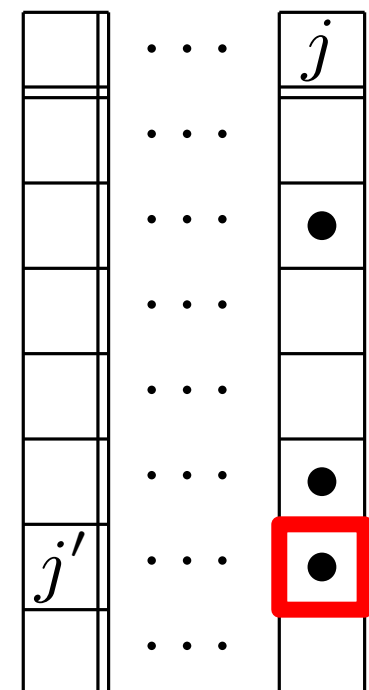
$\dashv \setminus \wedge$

	1	2	3	4	5	6	7
1				•		•	
2				•	•	•	
3					•		
4							•
5							•
6							•
7							

for $j=1$ to m do:

while $\exists k < j$ s.t. $\text{low}(k) == \text{low}(j)$ do:

$$\text{col}(j) = \text{col}(j) + \text{col}(k)$$



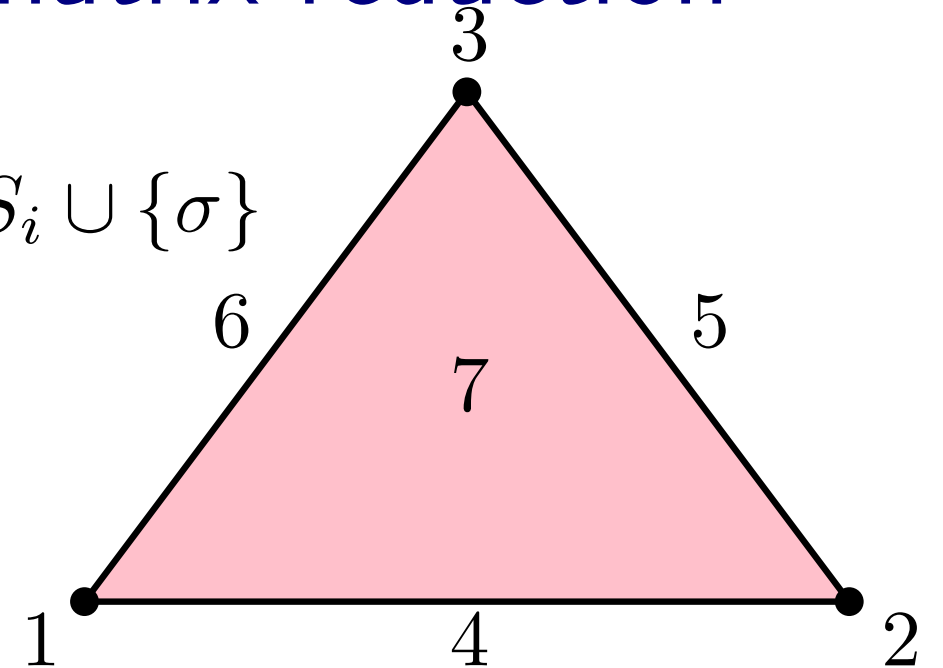
	5	6
1		•
2	•	•
3	•	

$$6 = 6 + 5$$

$$\text{low}(j) = j'$$

Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$
 given as *boundary matrix*



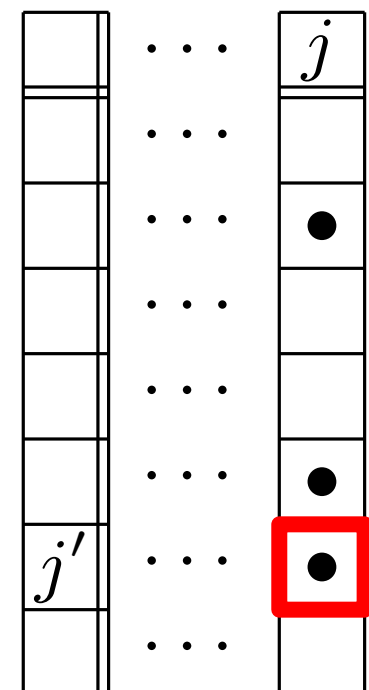
\dashv \setminus \wedge

	1	2	3	4	5	6	7
1				•		•	
2				•	•	•	
3					•		
4							•
5							•
6							•
7							

for $j=1$ to m do:

 while $\exists k < j$ s.t. $\text{low}(k) == \text{low}(j)$ do:

$$\text{col}(j) = \text{col}(j) + \text{col}(k)$$



	5	6
1		•
2	•	•
3	•	

	4	6
1	•	•
2	•	•
3		

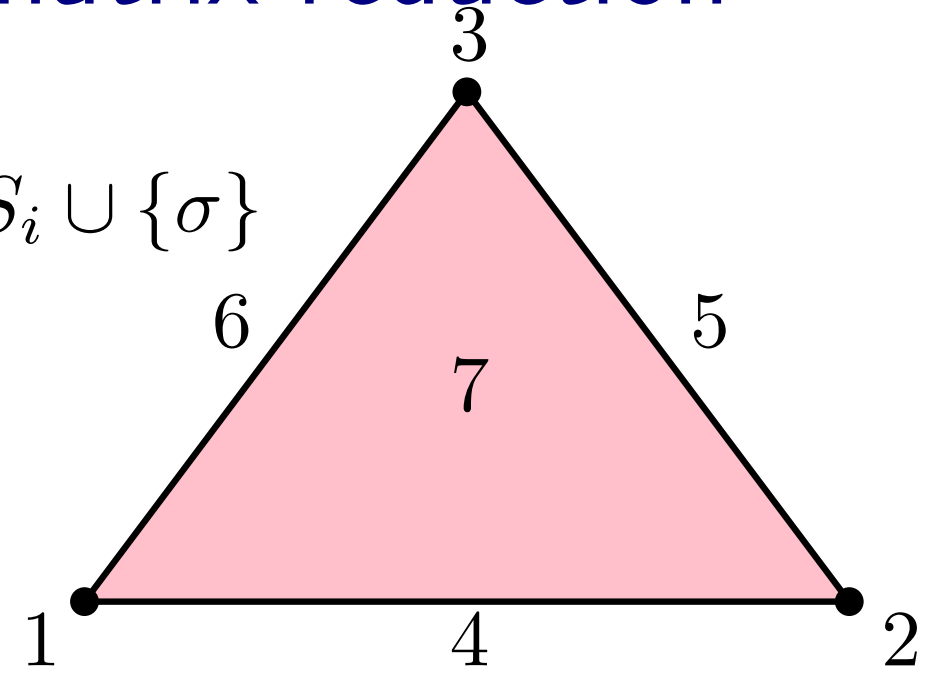
$$6 = 6 + 5$$

$$6 = 6 + 4$$

$$\text{low}(j) = j'$$

Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$
 given as *boundary matrix*



$\dashv \setminus \Delta$

	1	2	3	4	5	6	7
1				•			
2				•	•		
3					•		
4							•
5							•
6							•
7							

for $j=1$ to m do:

while $\exists k < j$ s.t. $\text{low}(k) == \text{low}(j)$ do:

$\text{col}(j) = \text{col}(j) + \text{col}(k)$

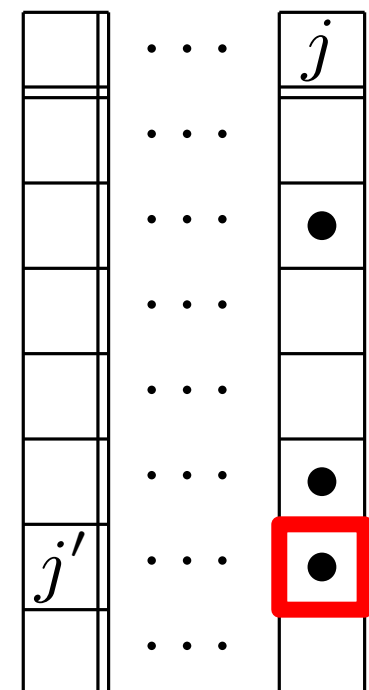
	5	6
1		•
2	•	•
3	•	

	4	6
1	•	
2	•	
3		

$$6 = 6 + 5$$

$$6 = 6 + 4$$

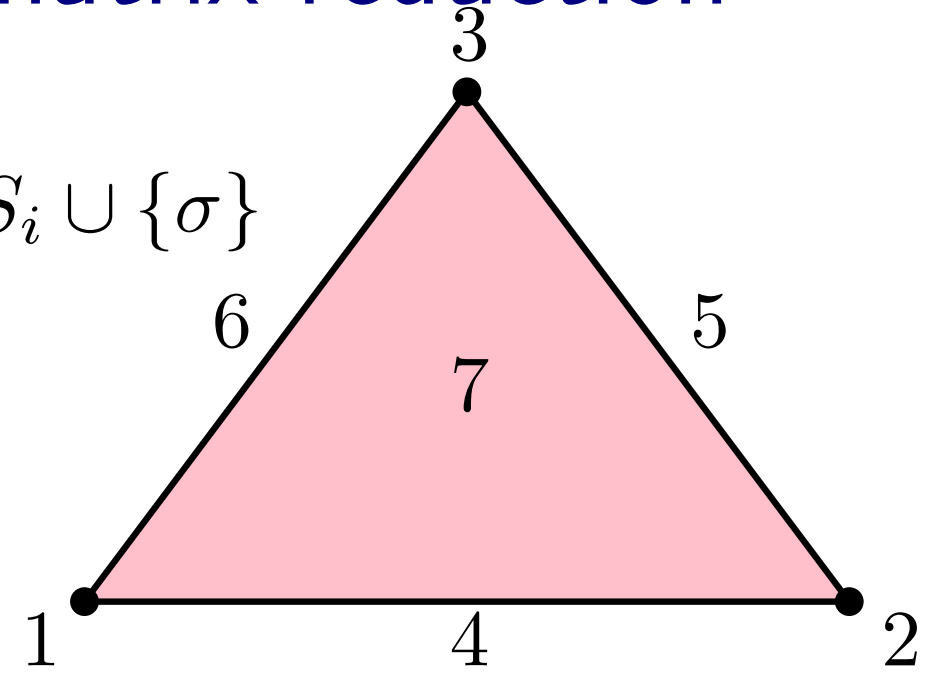
$$\text{low}(j) = j'$$



Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$

Output: boundary matrix

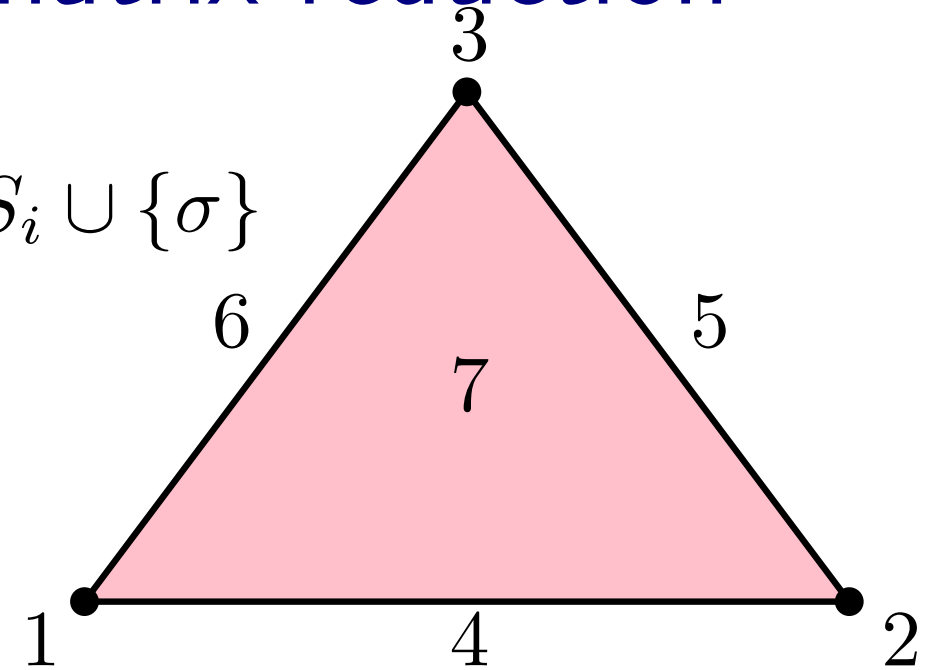


	1	2	3	4	5	6	7
1				*		*	
2				*	*		
3					*	*	
4							*
5							*
6							*
7							

Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$

Output: boundary matrix
reduced to column-echelon form



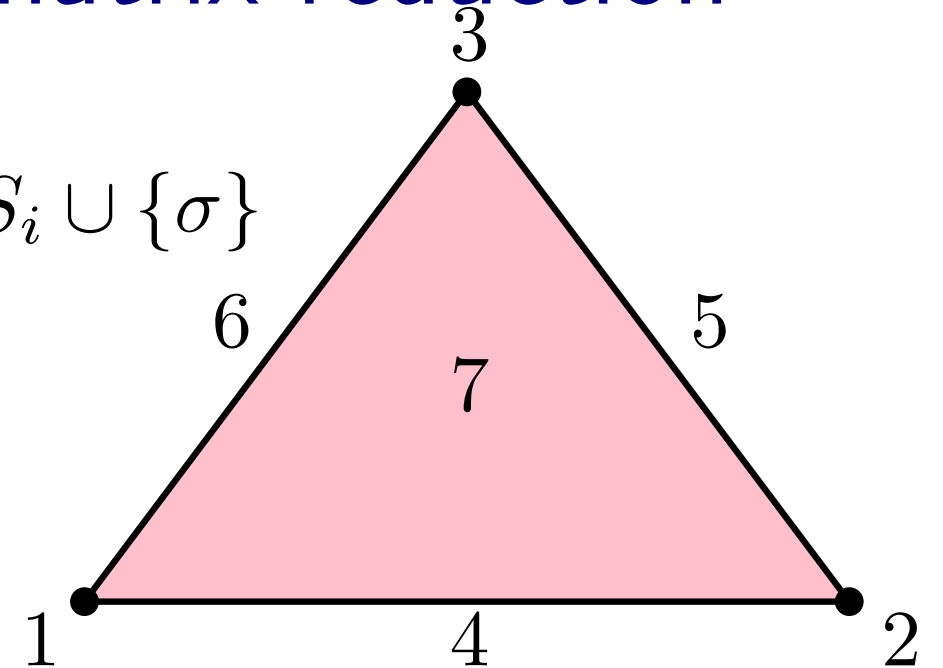
	1	2	3	4	5	6	7
1				*		*	
2				*	*		
3					*	*	
4							*
5							*
6							*
7							

	1	2	3	4	5	6	7
1				*			
2				1	*		
3					1		
4							*
5							*
6							1
7							

Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$

Output: boundary matrix
reduced to column-echelon form



○ some positive-negative simplices are paired
[2, 4), [3, 5), [6, 7)

□ unpaired simplices provide homology basis: $[1, +\infty)$

	1	2	3	4	5	6	7
1				*		*	
2				*	*		
3					*	*	
4							*
5							*
6							*
7							

	1	2	3	4	5	6	7
1				*			
2				①	*		
3					①		
4							*
5							*
6							①
7							

Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$

Output: boundary matrix
reduced to column-echelon form

Q: Complexity?

Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$

Output: boundary matrix
reduced to column-echelon form

Q: Complexity?

PLU factorization:

- Gaussian elimination
- fast matrix multiplication (divide-and-conquer)
- random projections?

Computation with filtrations and matrix reduction

Input: simplicial filtration $\{S_i\}_{i \in \mathbb{R}}$ s.t. $S_{i+1} = S_i \cup \{\sigma\}$

Output: boundary matrix
reduced to column-echelon form

Q: Complexity?

PLU factorization:

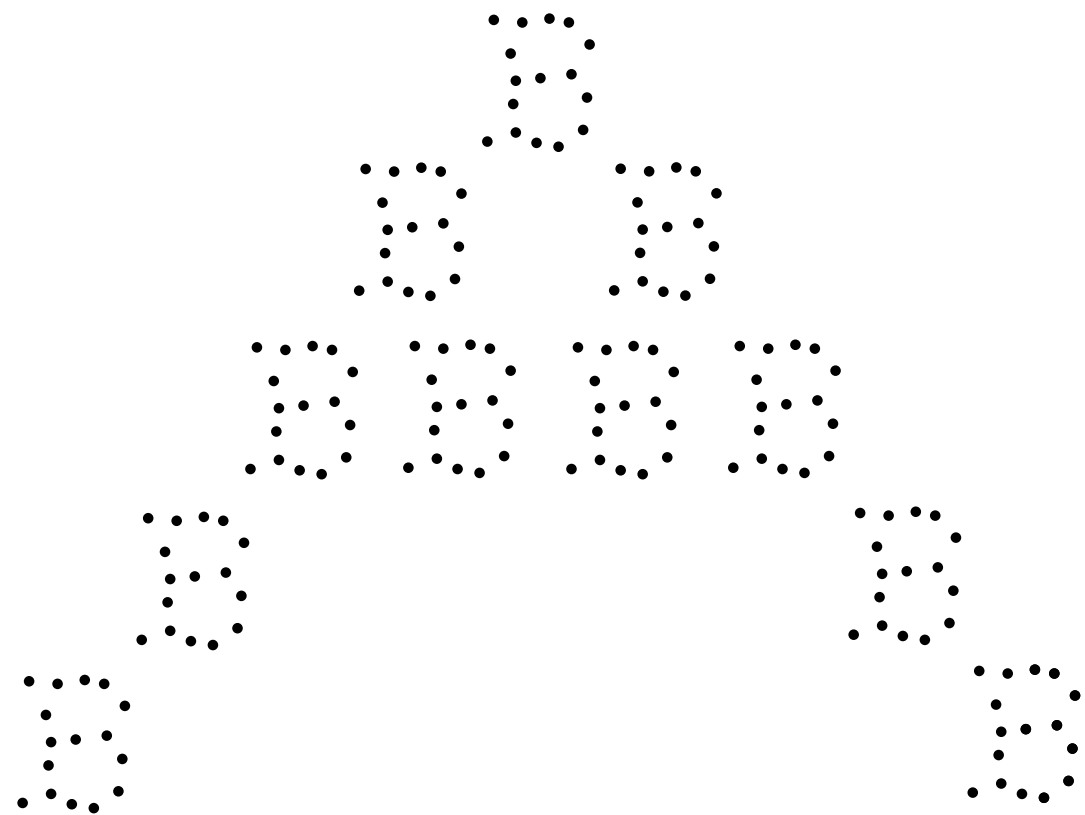
- Gaussian elimination

- PLEX / JavaPLEX (<http://appliedtopology.github.io/javaplex/>)
- Dionysus (<http://www.mrzv.org/software/dionysus/>)
- Perseus (<http://www.sas.upenn.edu/~vnanda/perseus/>)
- Gudhi (<http://gudhi.gforge.inria.fr/>)
- PHAT (<https://bitbucket.org/phat-code/phat>)
- DIPHA (<https://github.com/DIPHA/dipha/>)
- CTL (<https://github.com/appliedtopology/ctl>)

Computational Topology (II): Persistence Theory

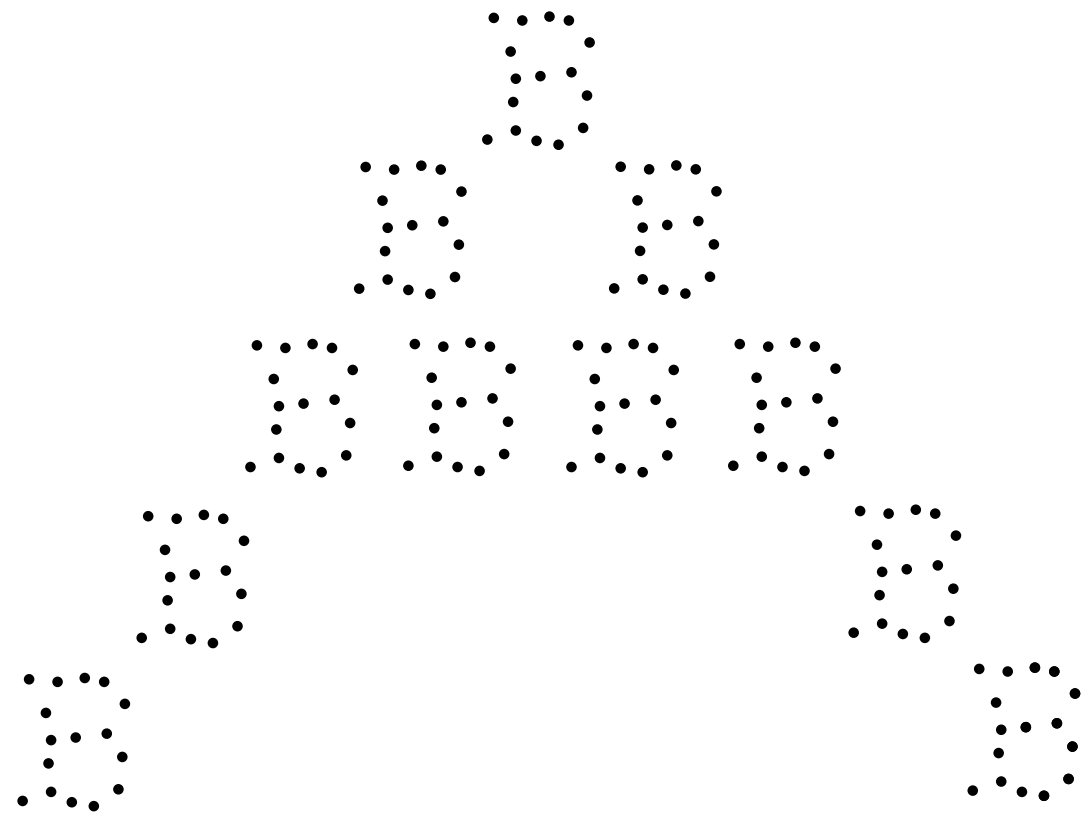
1. Algorithmic Foundation
- 2. Algebraic Foundation**
3. Stability Theorem

Introduction: problems with homology



First, the algorithm for computing homology contains much more information than the mere homology of the last complex in the filtration.

Introduction: problems with homology

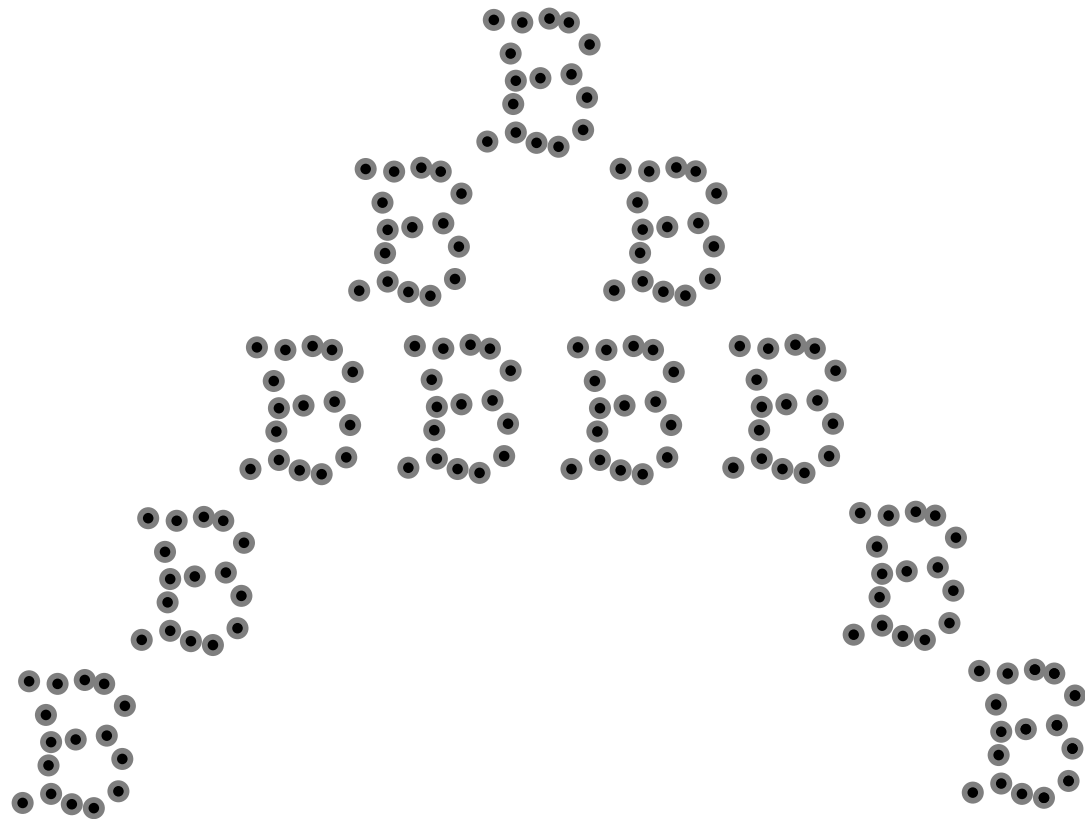


First, the algorithm for computing homology contains much more information than the mere homology of the last complex in the filtration.

Indeed, it contains the homology of *all* the subcomplexes in the filtration.

This is very interesting in the sense that data can be analyzed at *multiple* scales.

Introduction: problems with homology

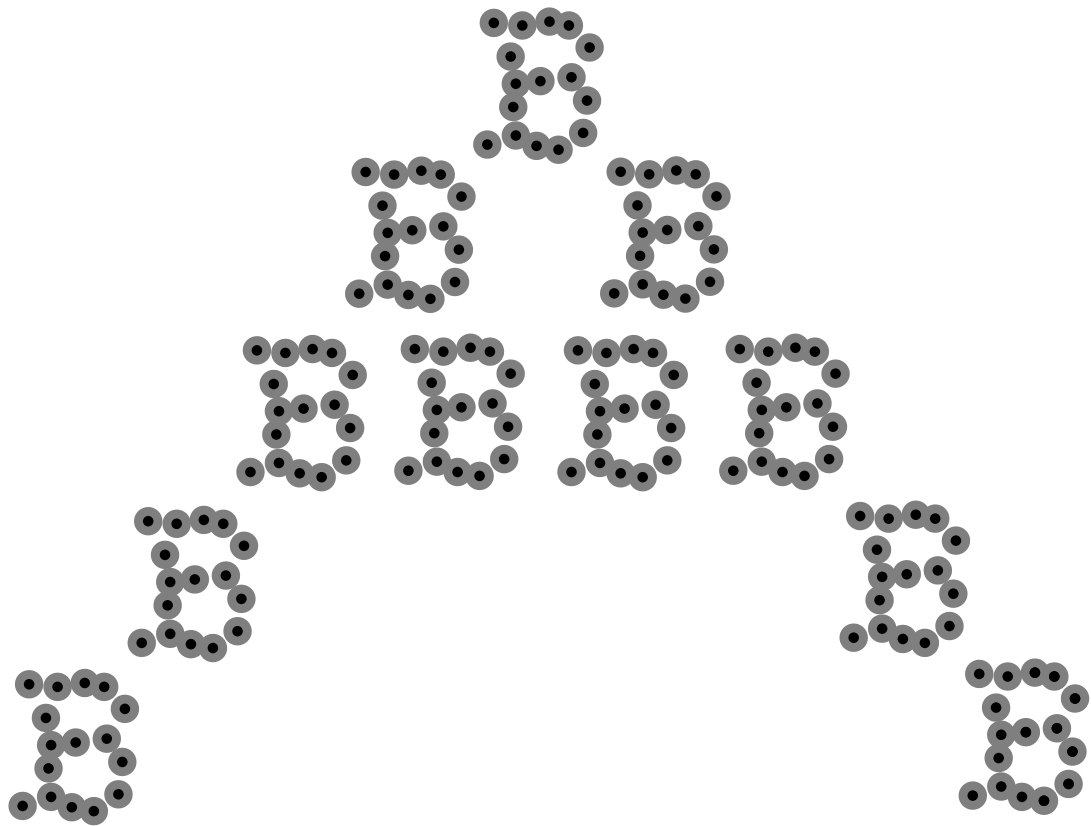


First, the algorithm for computing homology contains much more information than the mere homology of the last complex in the filtration.

Indeed, it contains the homology of *all* the subcomplexes in the filtration.

This is very interesting in the sense that data can be analyzed at *multiple* scales.

Introduction: problems with homology

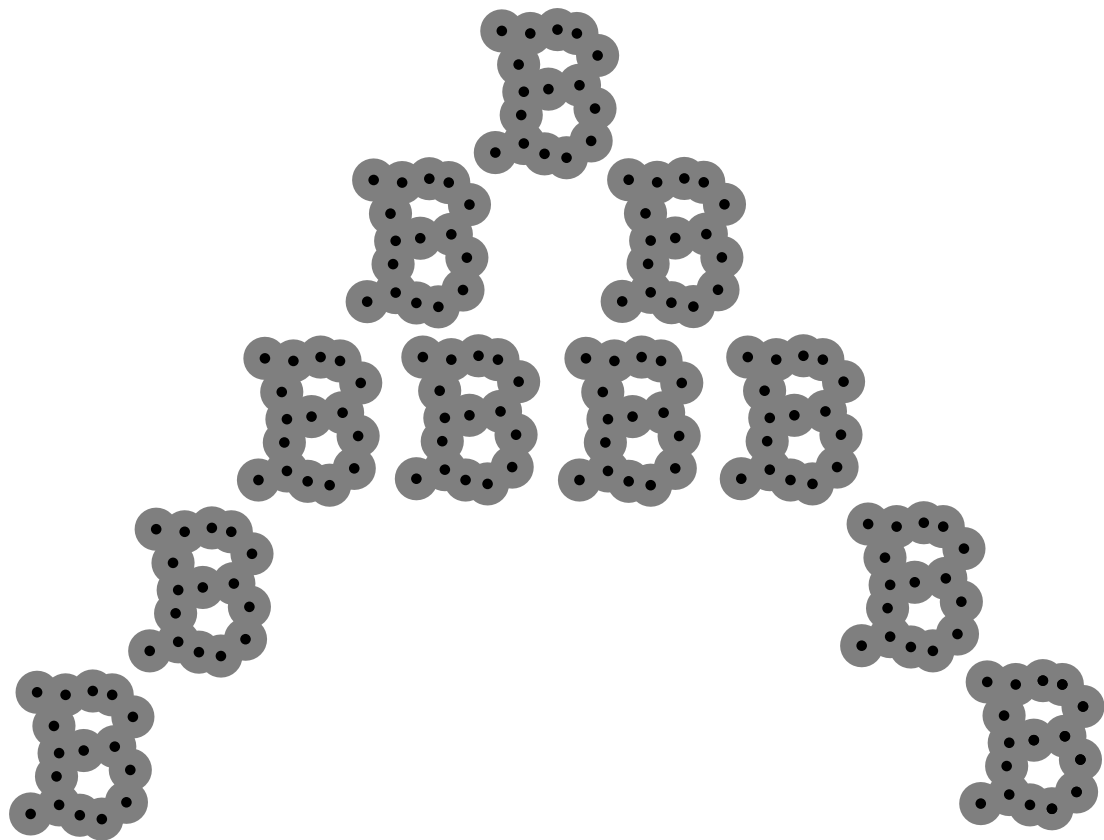


First, the algorithm for computing homology contains much more information than the mere homology of the last complex in the filtration.

Indeed, it contains the homology of *all* the subcomplexes in the filtration.

This is very interesting in the sense that data can be analyzed at *multiple* scales.

Introduction: problems with homology

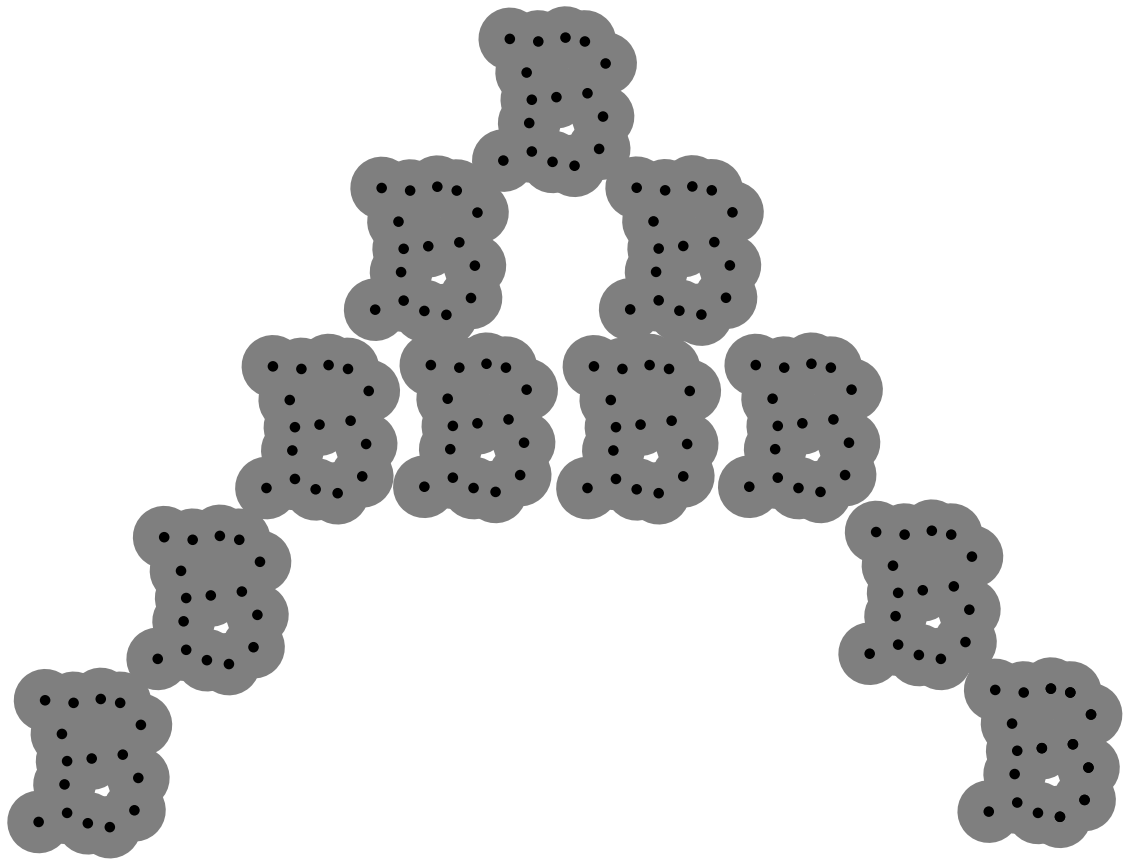


First, the algorithm for computing homology contains much more information than the mere homology of the last complex in the filtration.

Indeed, it contains the homology of *all* the subcomplexes in the filtration.

This is very interesting in the sense that data can be analyzed at *multiple* scales.

Introduction: problems with homology

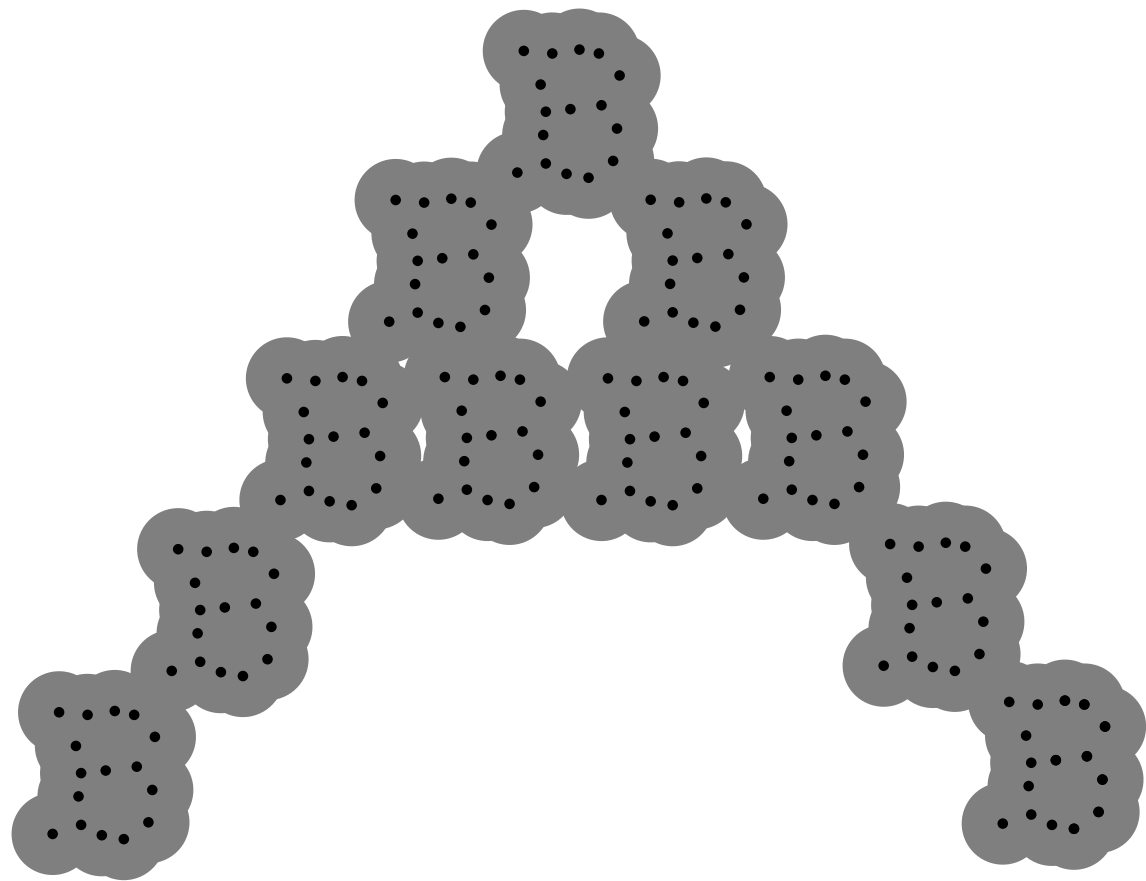


First, the algorithm for computing homology contains much more information than the mere homology of the last complex in the filtration.

Indeed, it contains the homology of *all* the subcomplexes in the filtration.

This is very interesting in the sense that data can be analyzed at *multiple* scales.

Introduction: problems with homology

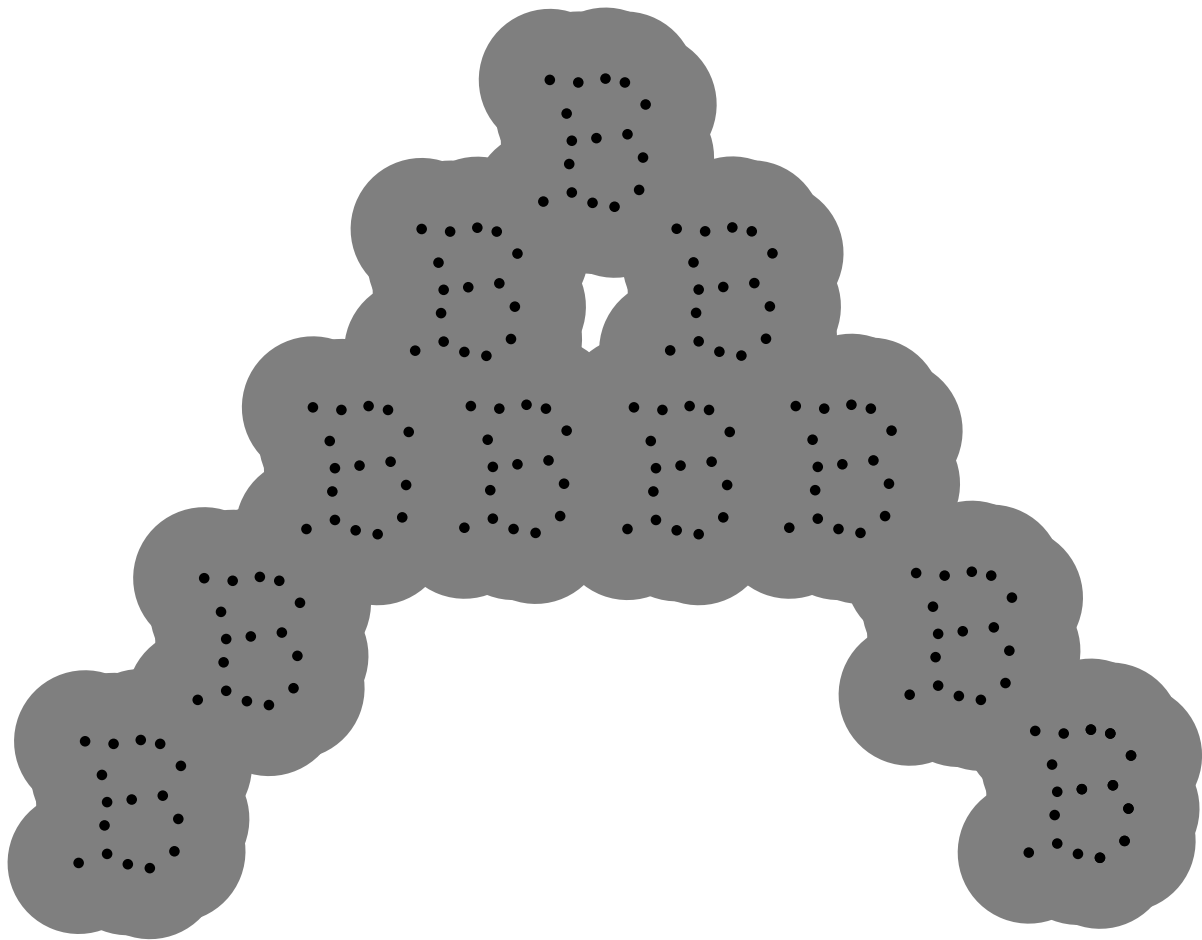


First, the algorithm for computing homology contains much more information than the mere homology of the last complex in the filtration.

Indeed, it contains the homology of *all* the subcomplexes in the filtration.

This is very interesting in the sense that data can be analyzed at *multiple* scales.

Introduction: problems with homology

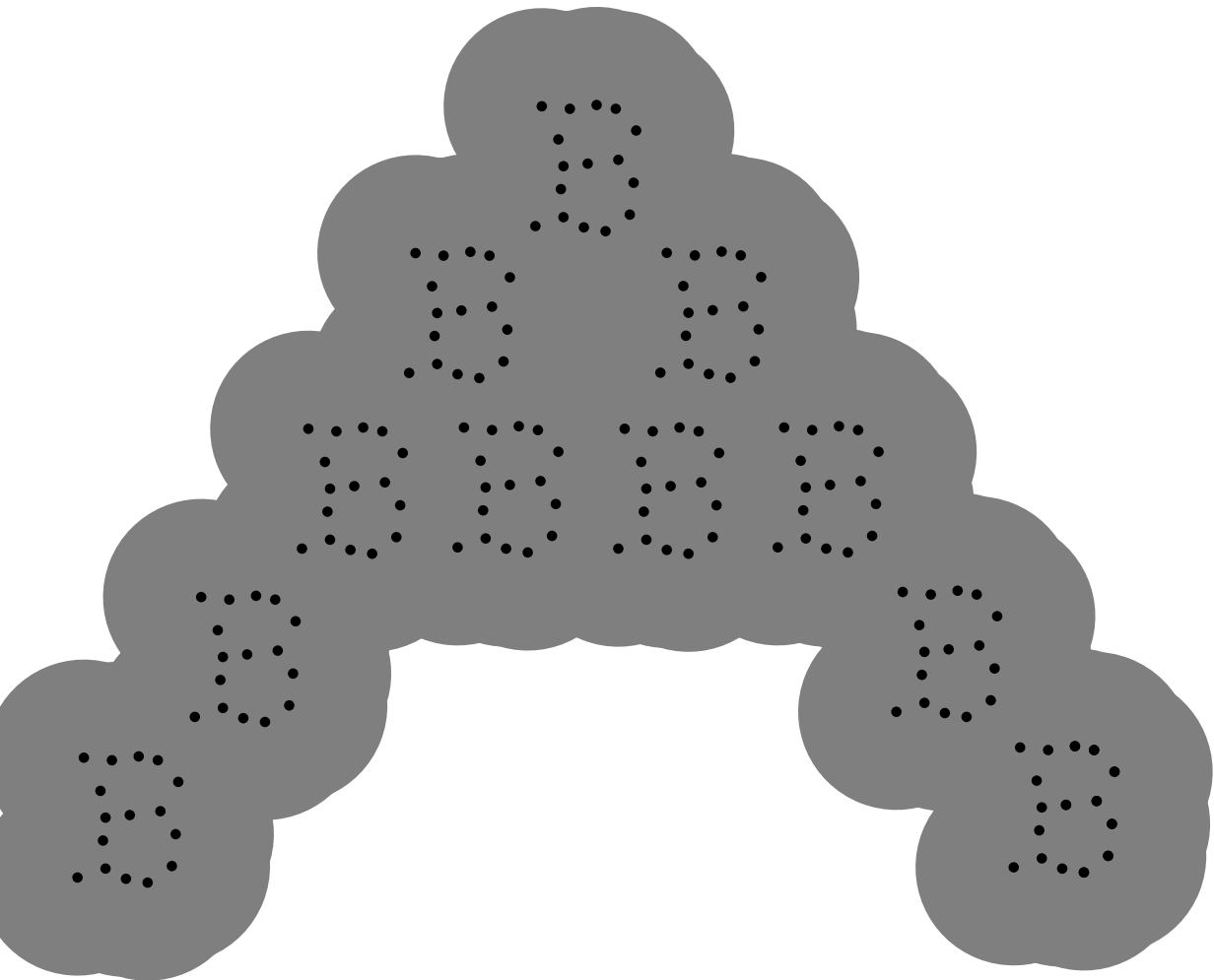


First, the algorithm for computing homology contains much more information than the mere homology of the last complex in the filtration.

Indeed, it contains the homology of *all* the subcomplexes in the filtration.

This is very interesting in the sense that data can be analyzed at *multiple* scales.

Introduction: problems with homology

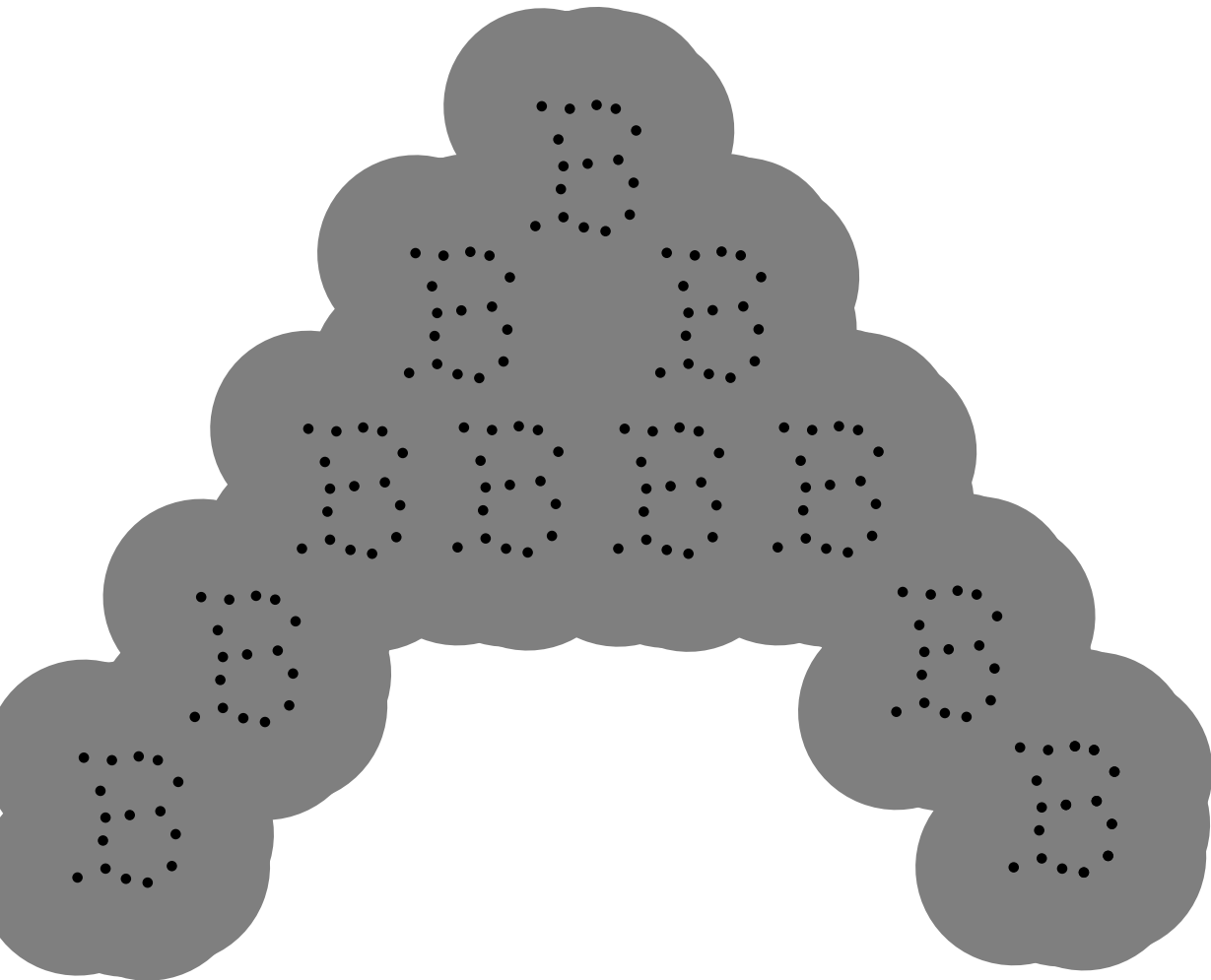


First, the algorithm for computing homology contains much more information than the mere homology of the last complex in the filtration.

Indeed, it contains the homology of *all* the subcomplexes in the filtration.

This is very interesting in the sense that data can be analyzed at *multiple* scales.

Introduction: problems with homology



First, the algorithm for computing homology contains much more information than the mere homology of the last complex in the filtration.

Indeed, it contains the homology of *all* the subcomplexes in the filtration.

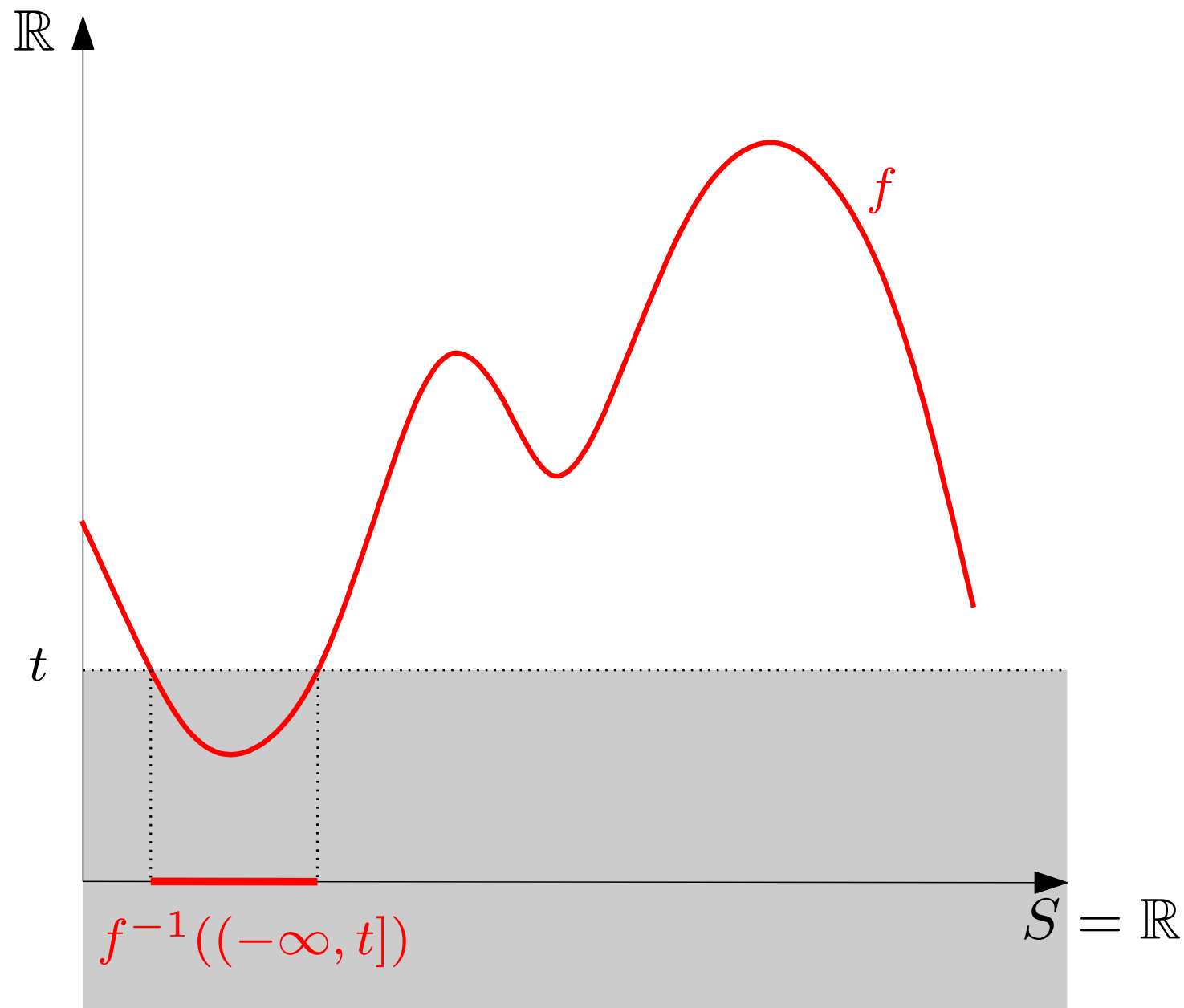
This is very interesting in the sense that data can be analyzed at *multiple* scales.

Persistent homology aims at encoding the homology of the complex at *all possible scales* into a compact descriptor.

Persistence of sublevel sets of function

- input: *filtration* = nested family of sublevel-sets $f^{-1}((-\infty, t])$ for t ranging over \mathbb{R}
- track the evolution of the topology (homology) throughout the family

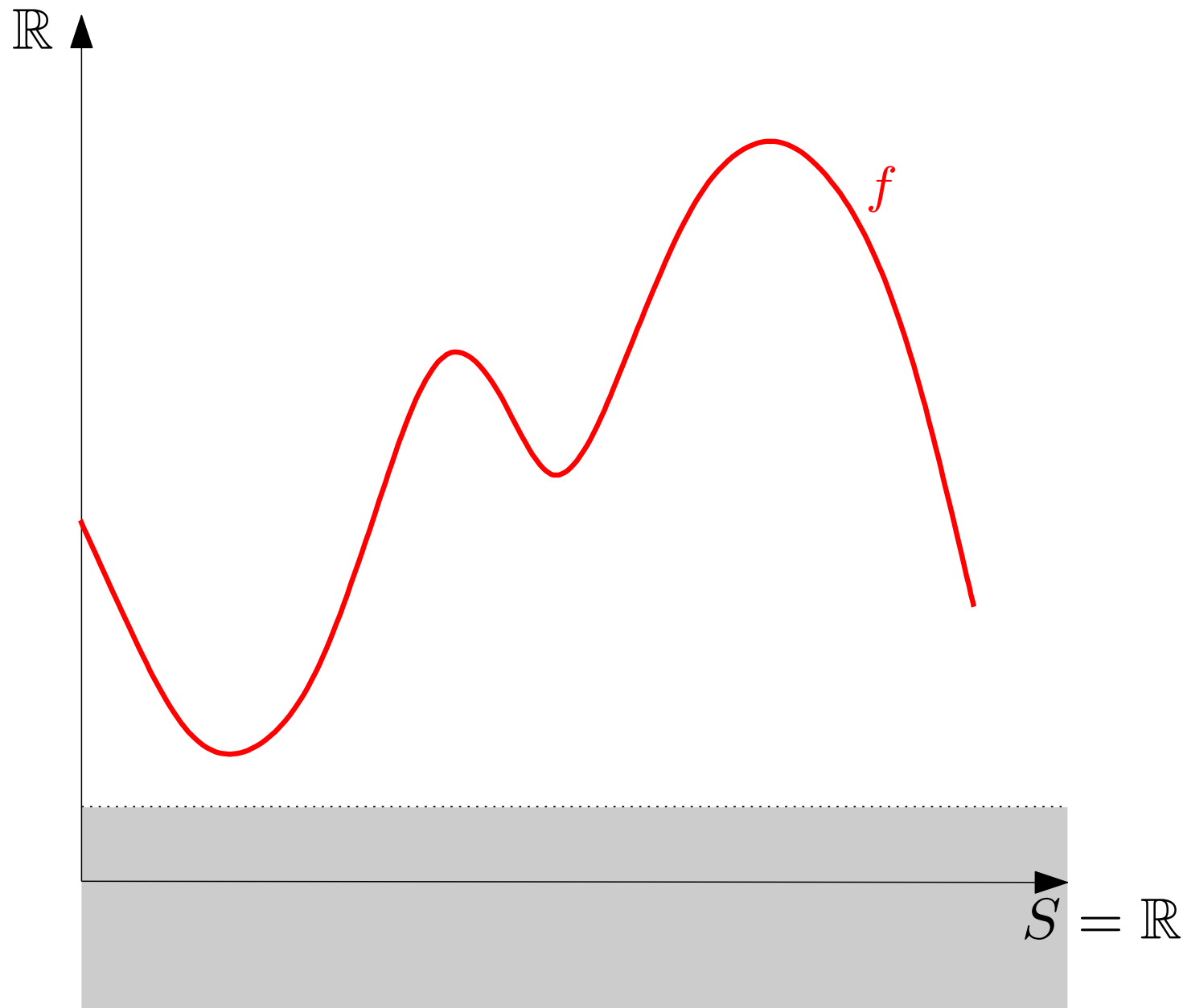
Ex: H_0 (connected components)



Persistence of sublevel sets of function

- input: *filtration* = nested family of sublevel-sets $f^{-1}((-\infty, t])$ for t ranging over \mathbb{R}
- track the evolution of the topology (homology) throughout the family

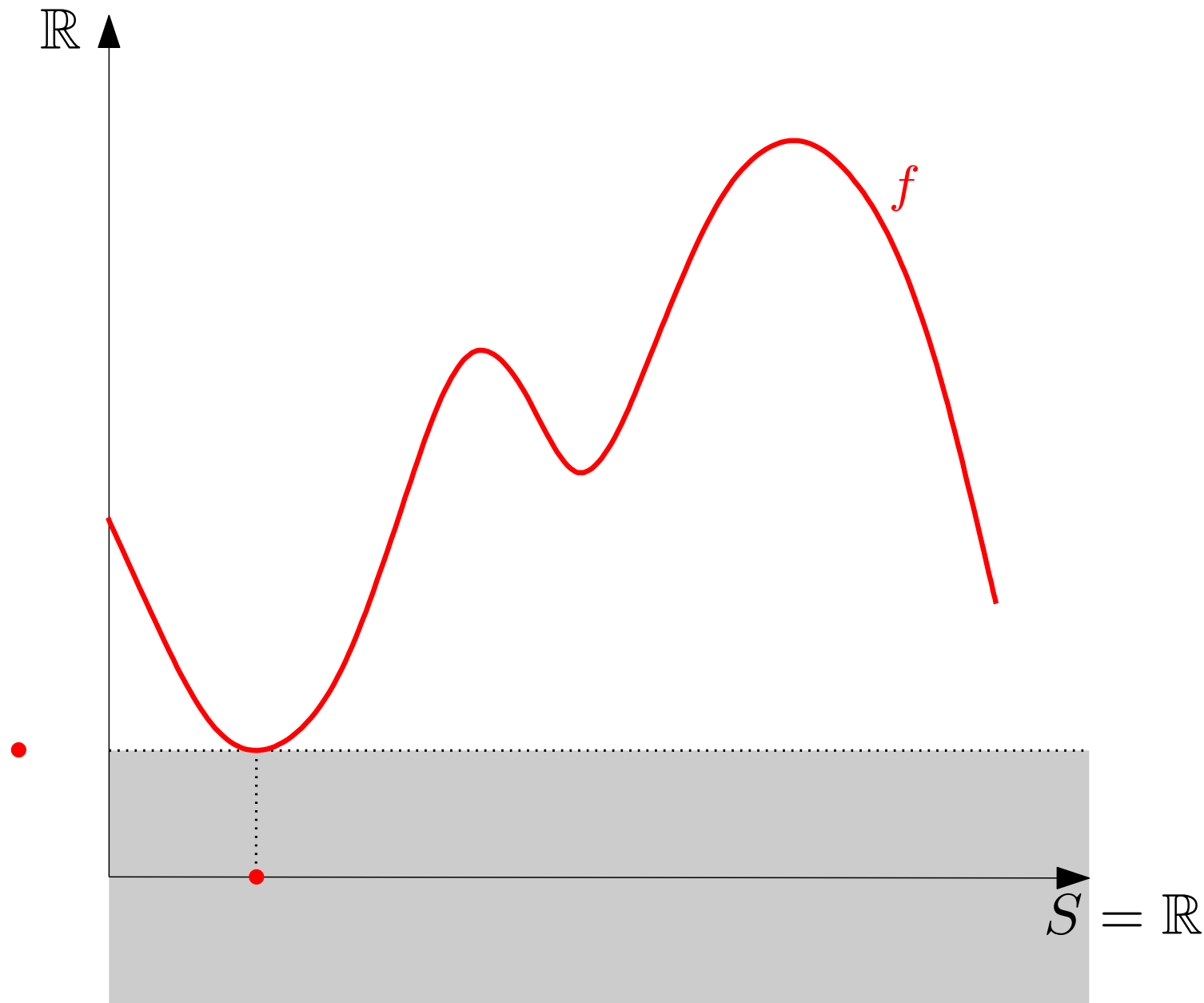
Ex: H_0 (connected components)



Persistence of sublevel sets of function

- input: *filtration* = nested family of sublevel-sets $f^{-1}((-\infty, t])$ for t ranging over \mathbb{R}
- track the evolution of the topology (homology) throughout the family

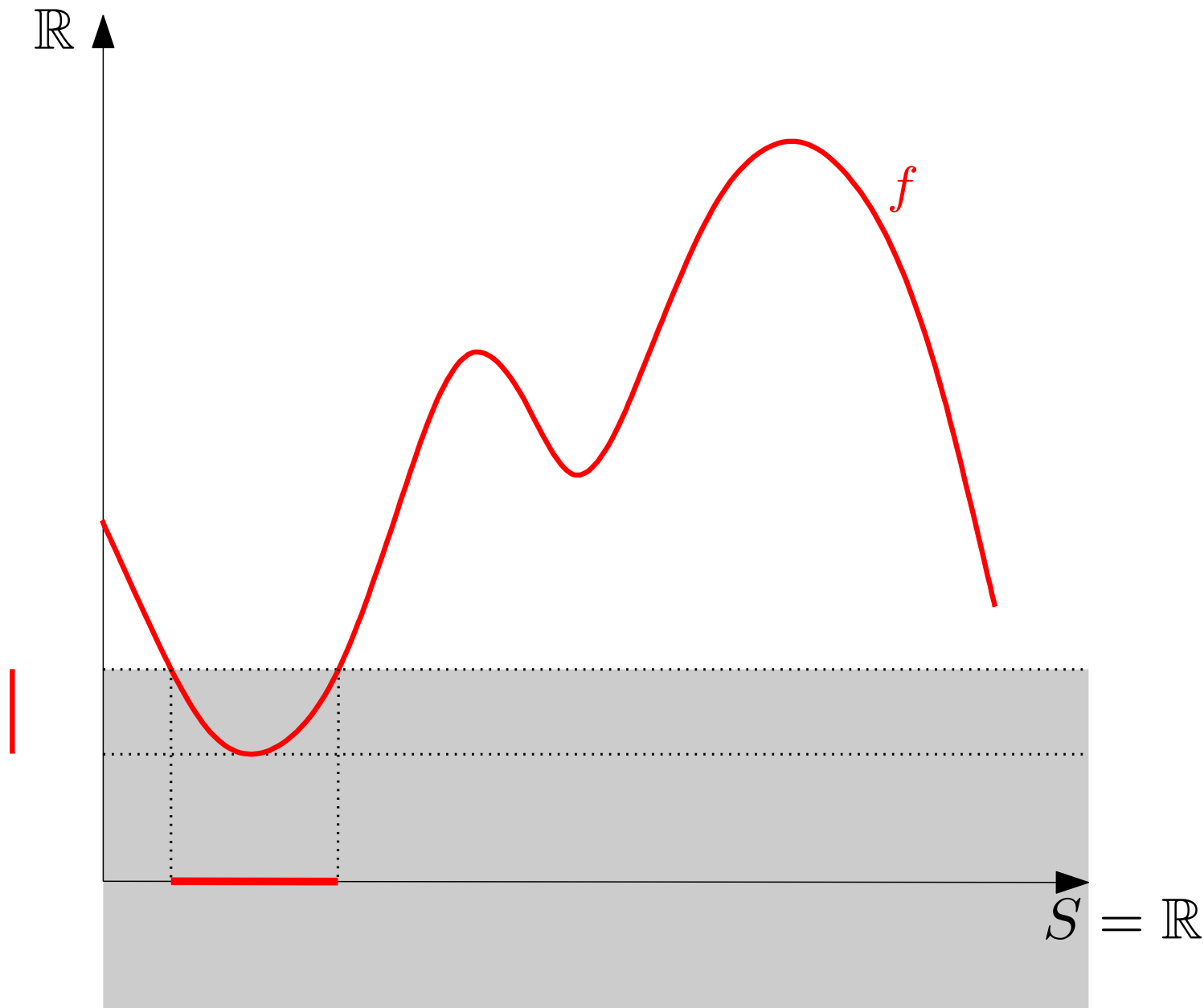
Ex: H_0 (connected components)



Persistence of sublevel sets of function

- input: *filtration* = nested family of sublevel-sets $f^{-1}((-\infty, t])$ for t ranging over \mathbb{R}
- track the evolution of the topology (homology) throughout the family

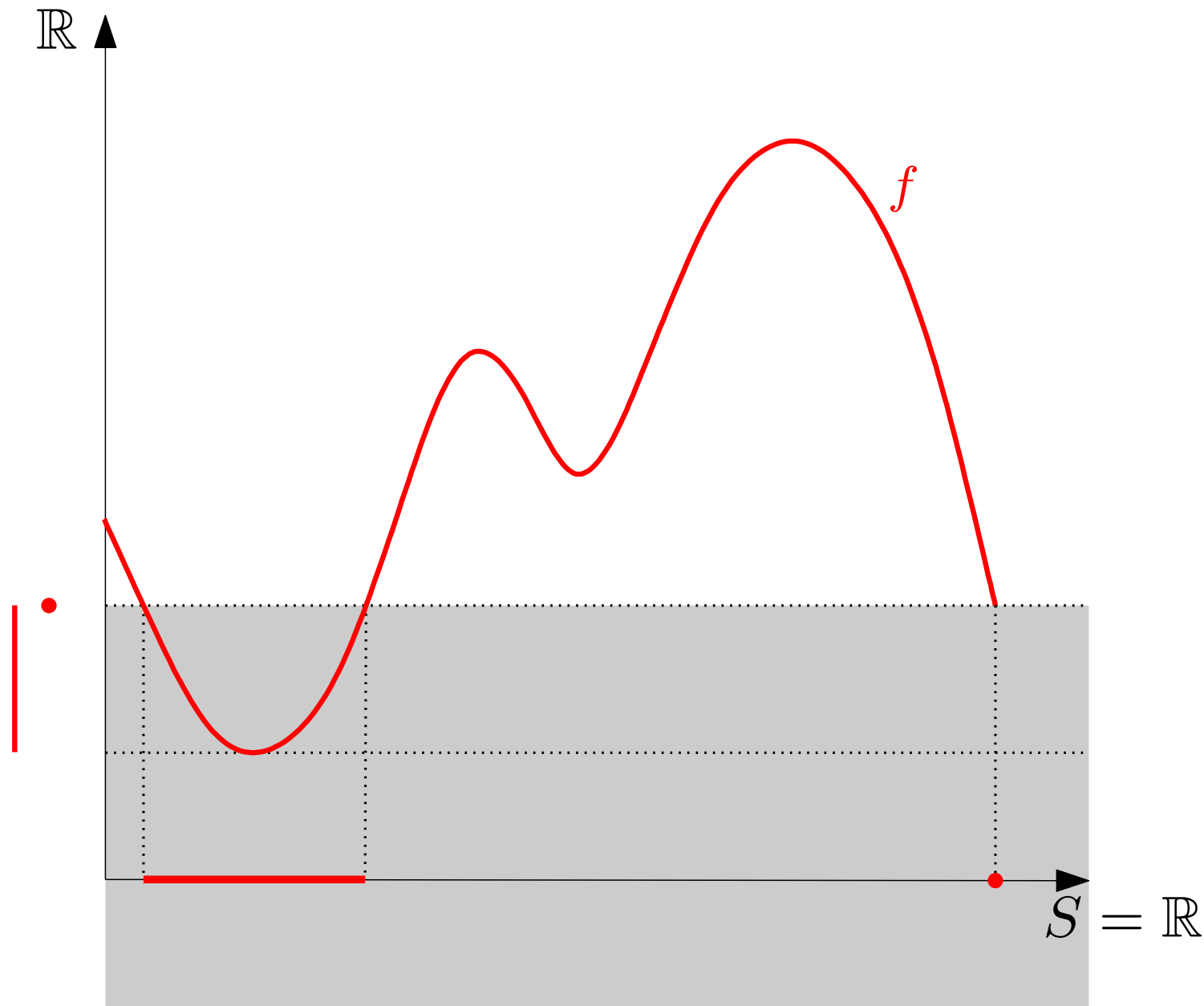
Ex: H_0 (connected components)



Persistence of sublevel sets of function

- input: *filtration* = nested family of sublevel-sets $f^{-1}((-\infty, t])$ for t ranging over \mathbb{R}
- track the evolution of the topology (homology) throughout the family

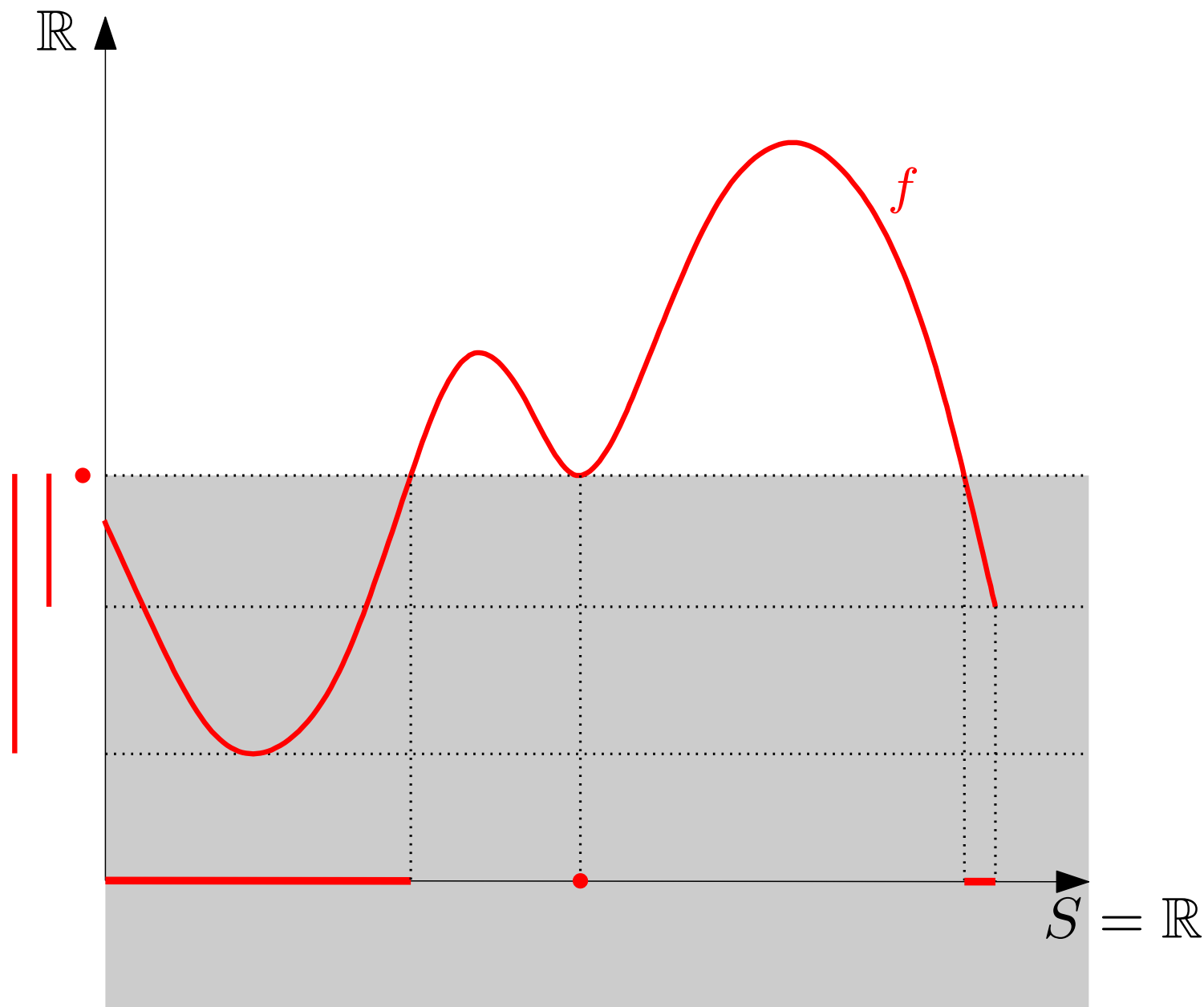
Ex: H_0 (connected components)



Persistence of sublevel sets of function

- input: *filtration* = nested family of sublevel-sets $f^{-1}((-\infty, t])$ for t ranging over \mathbb{R}
- track the evolution of the topology (homology) throughout the family

Ex: H_0 (connected components)

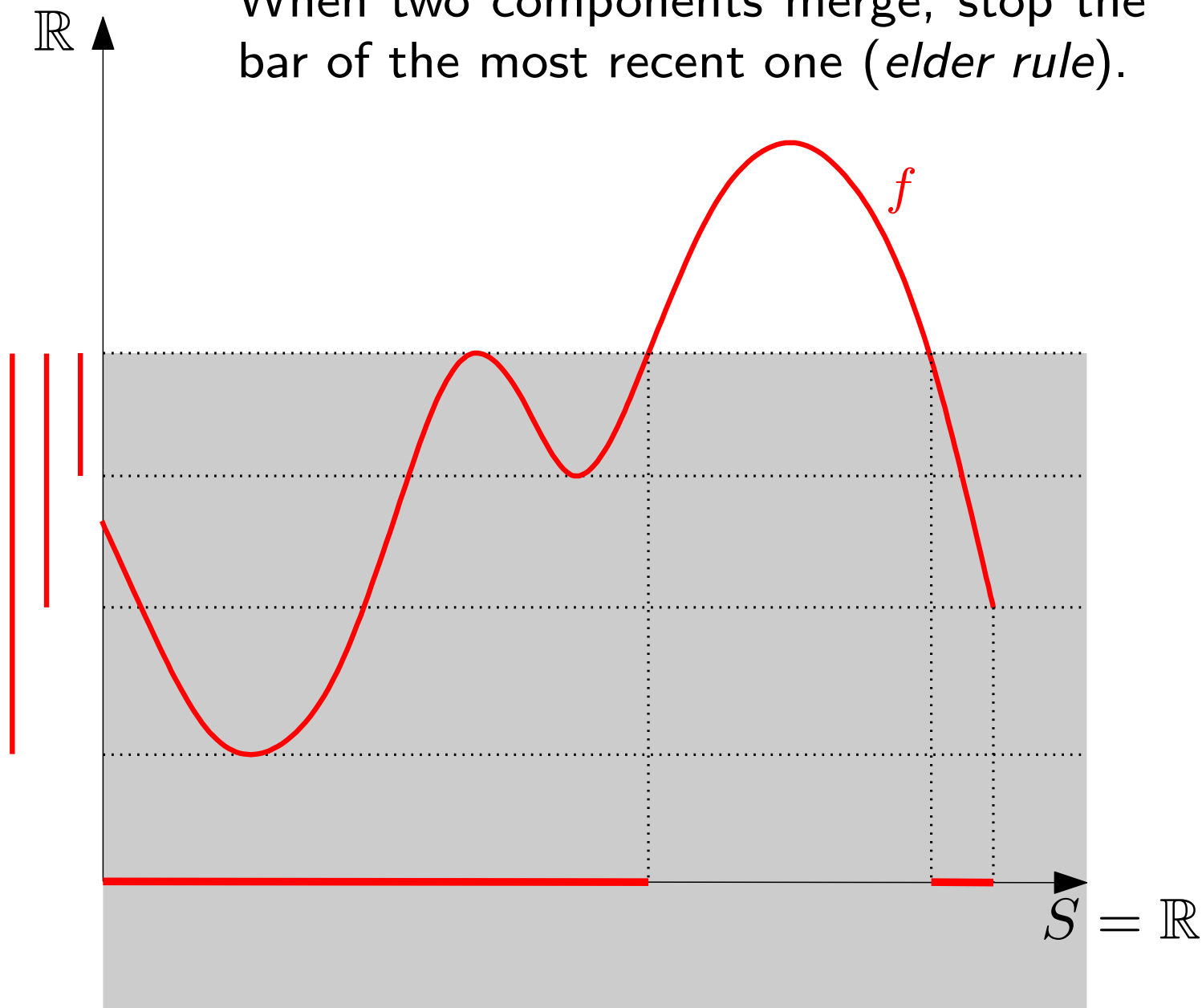


Persistence of sublevel sets of function

- input: *filtration* = nested family of sublevel-sets $f^{-1}((-\infty, t])$ for t ranging over \mathbb{R}
- track the evolution of the topology (homology) throughout the family

Ex: H_0 (connected components)

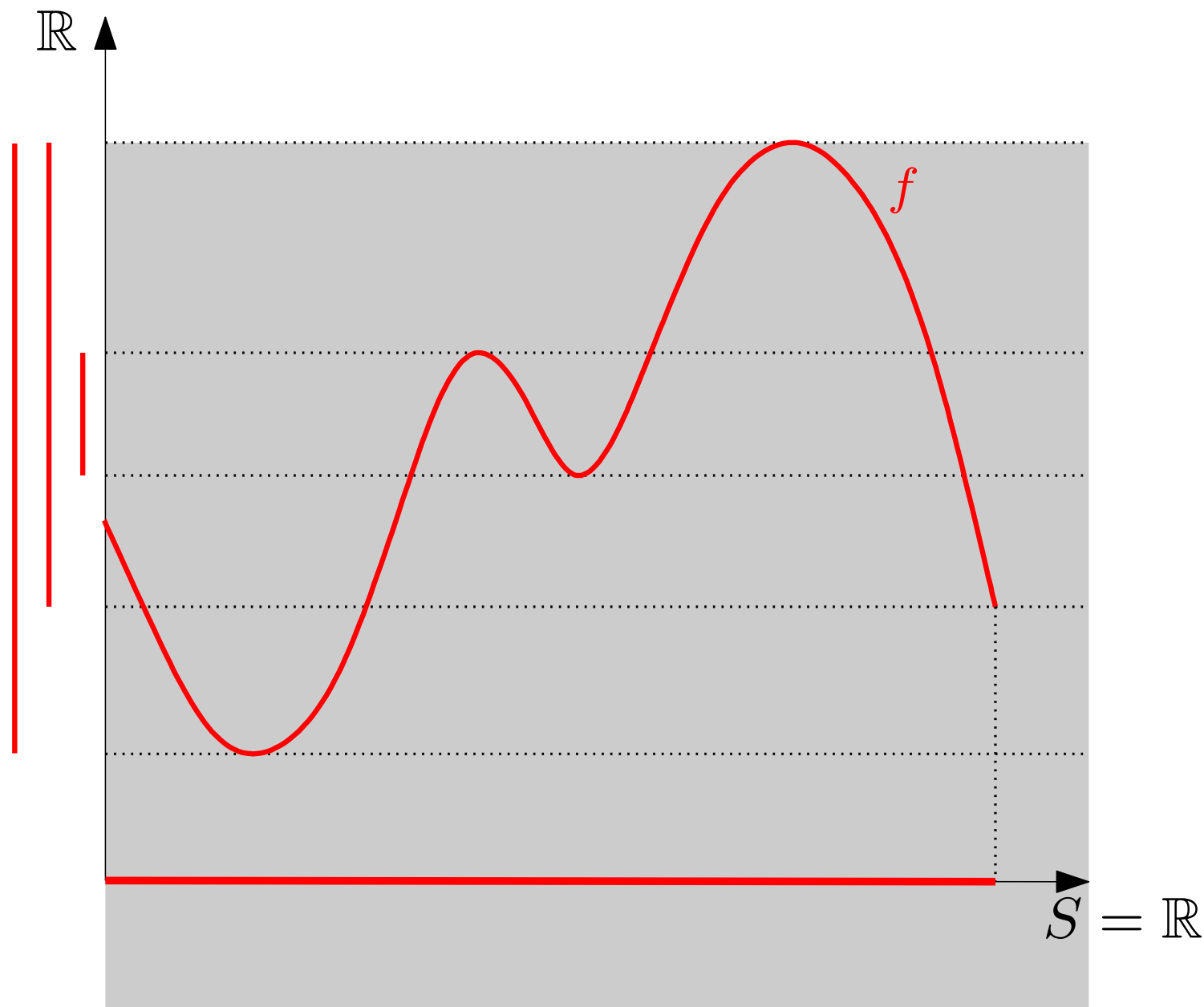
When two components merge, stop the bar of the most recent one (*elder rule*).



Persistence of sublevel sets of function

- input: *filtration* = nested family of sublevel-sets $f^{-1}((-\infty, t])$ for t ranging over \mathbb{R}
- track the evolution of the topology (homology) throughout the family

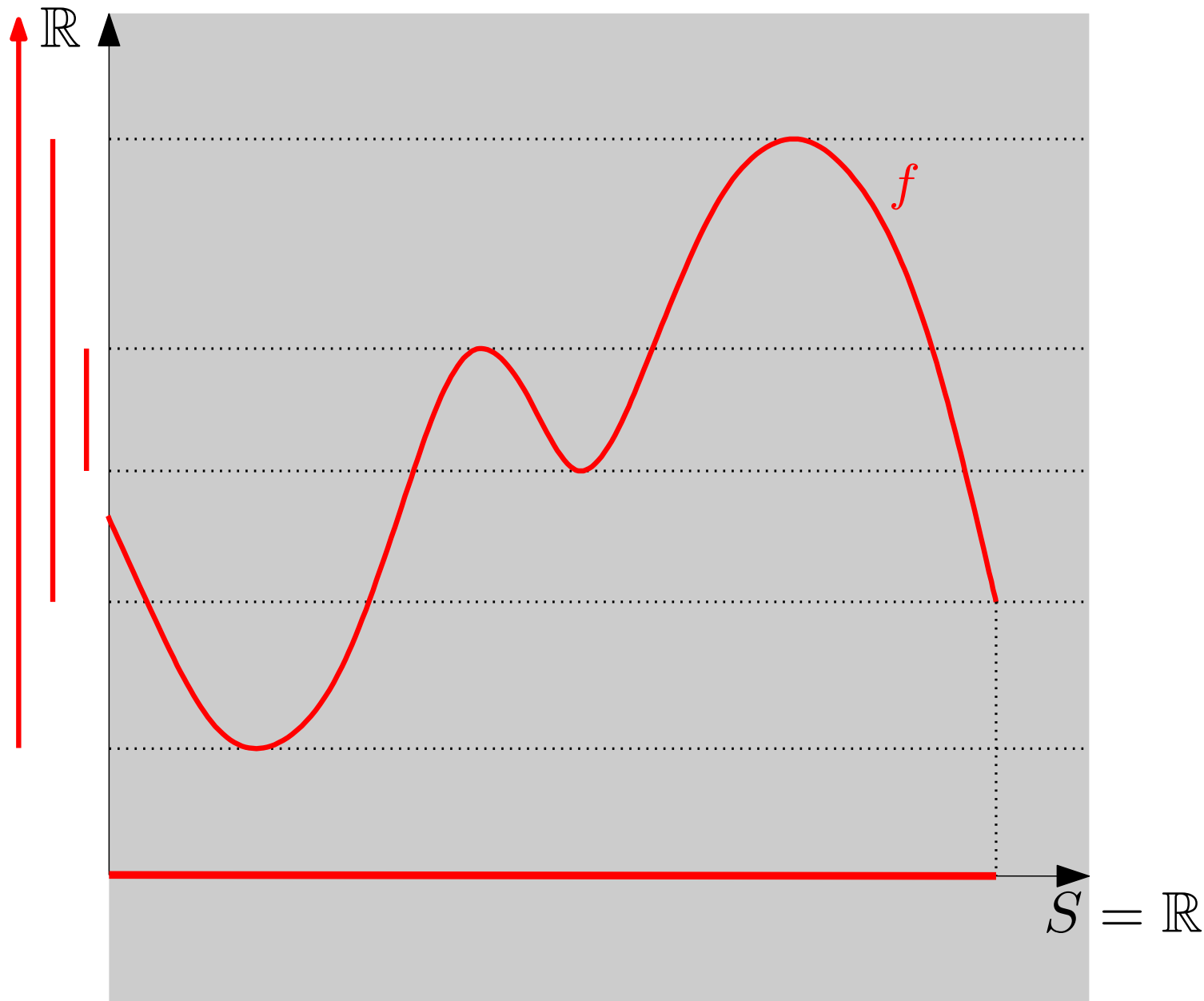
Ex: H_0 (connected components)



Persistence of sublevel sets of function

- input: *filtration* = nested family of sublevel-sets $f^{-1}((-\infty, t])$ for t ranging over \mathbb{R}
- track the evolution of the topology (homology) throughout the family
- finite set of intervals (barcode) encodes births/deaths of homology classes

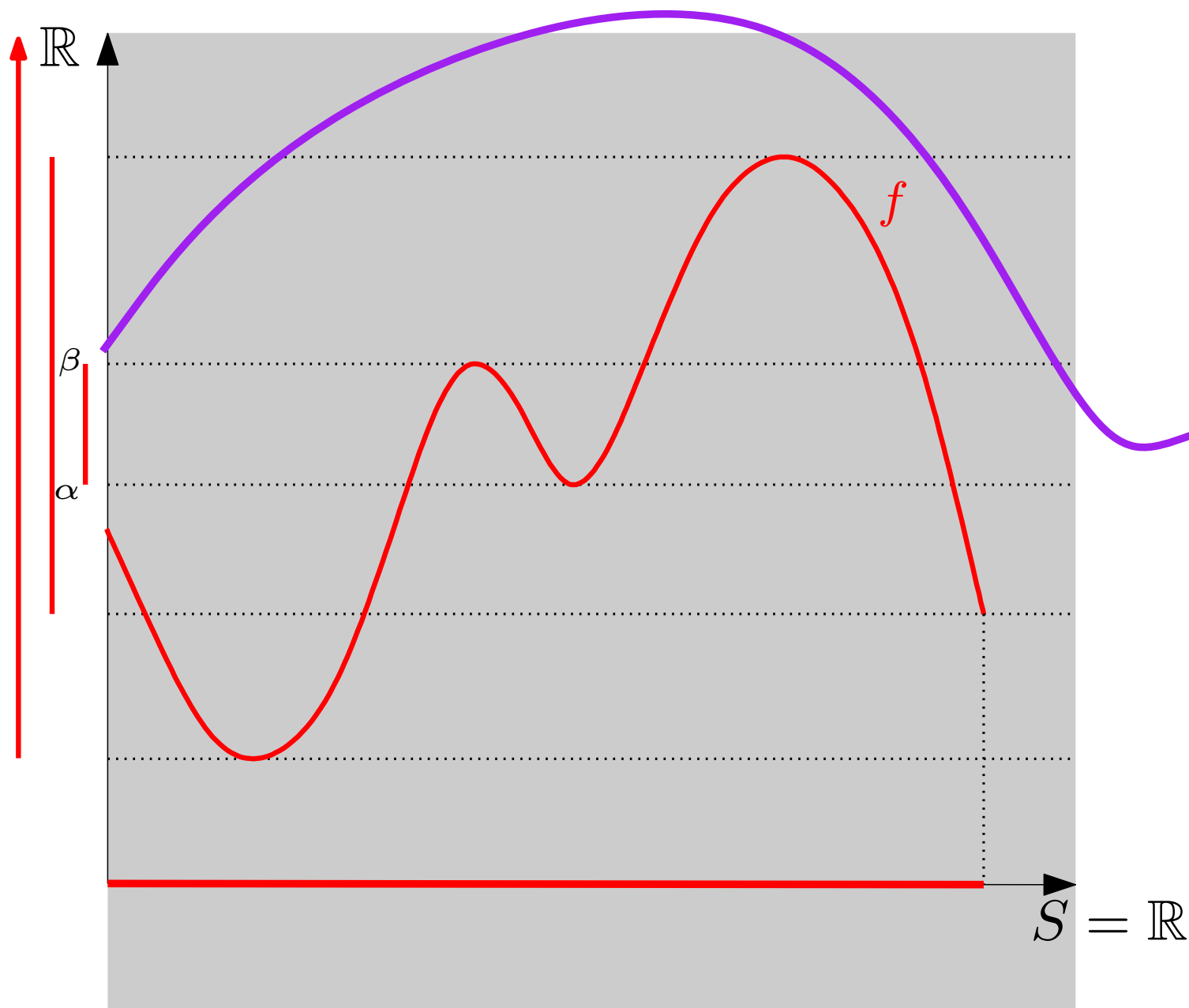
Ex: H_0 (connected components)



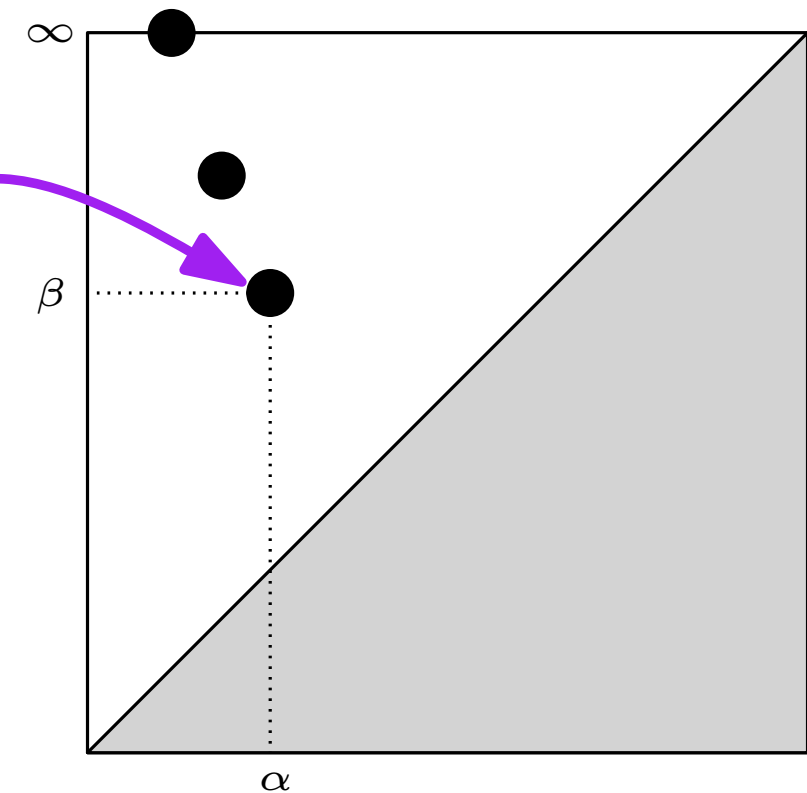
Persistence of sublevel sets of function

- input: *filtration* = nested family of sublevel-sets $f^{-1}((-\infty, t])$ for t ranging over \mathbb{R}
- track the evolution of the topology (homology) throughout the family
- finite set of intervals (barcode) encodes births/deaths of homology classes

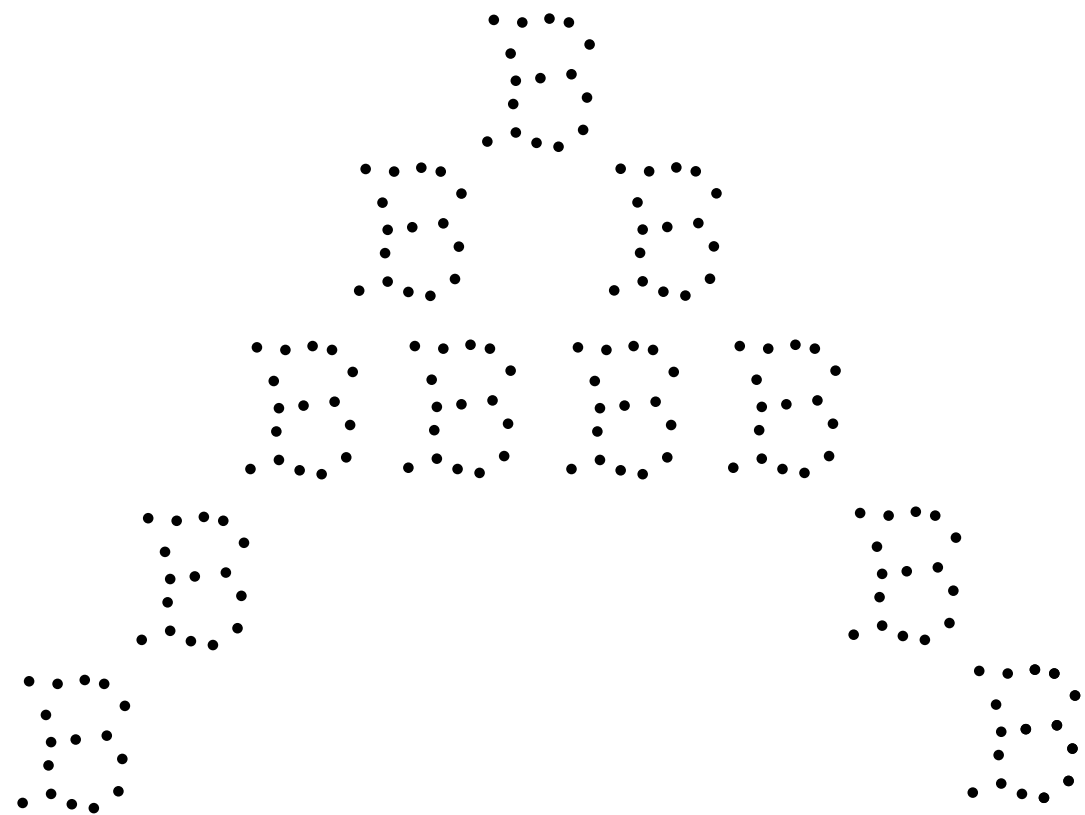
Ex: H_0 (connected components)



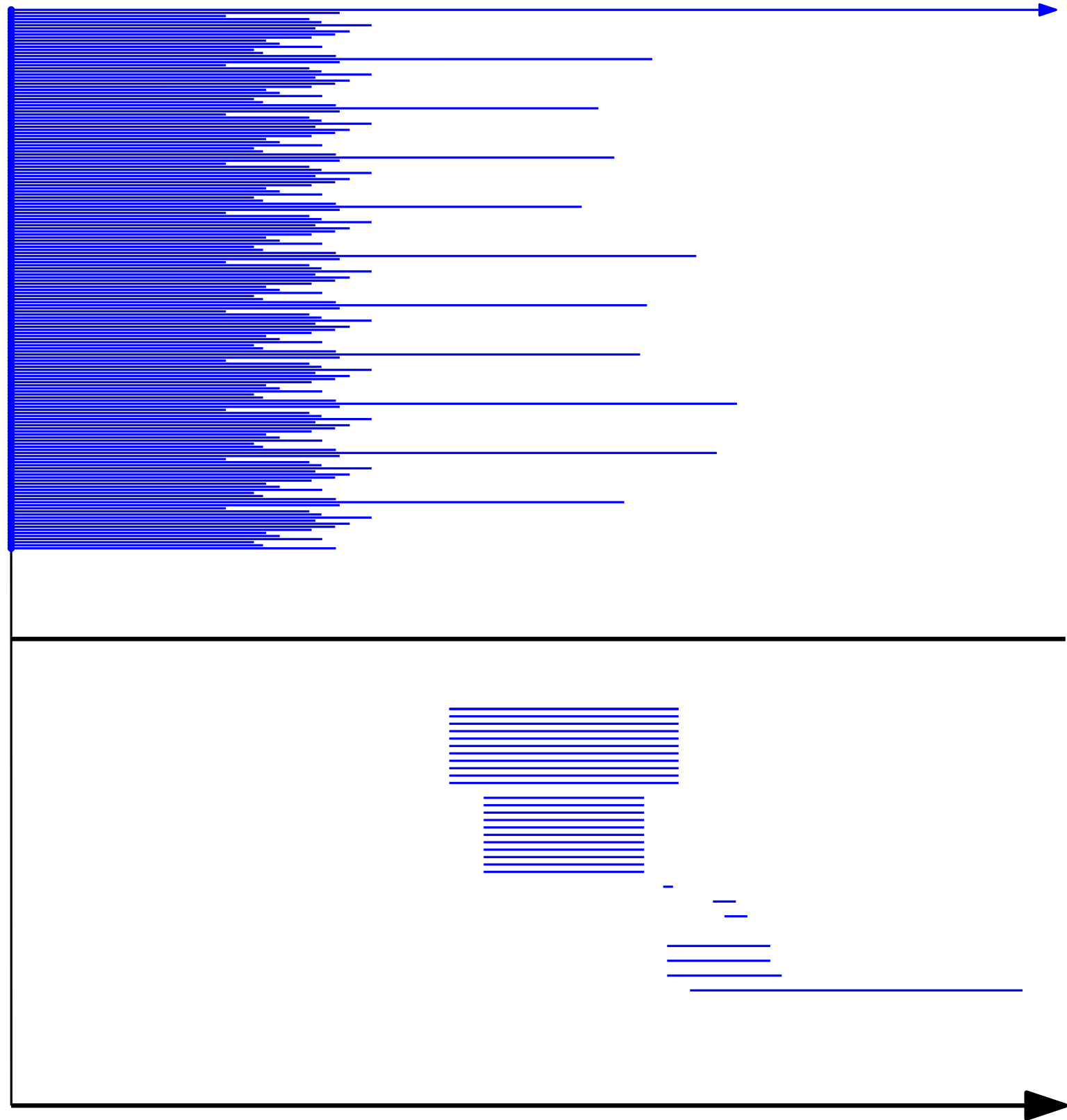
- alternate representation as a (multi-) set of points in the plane (*persistence diagram*).



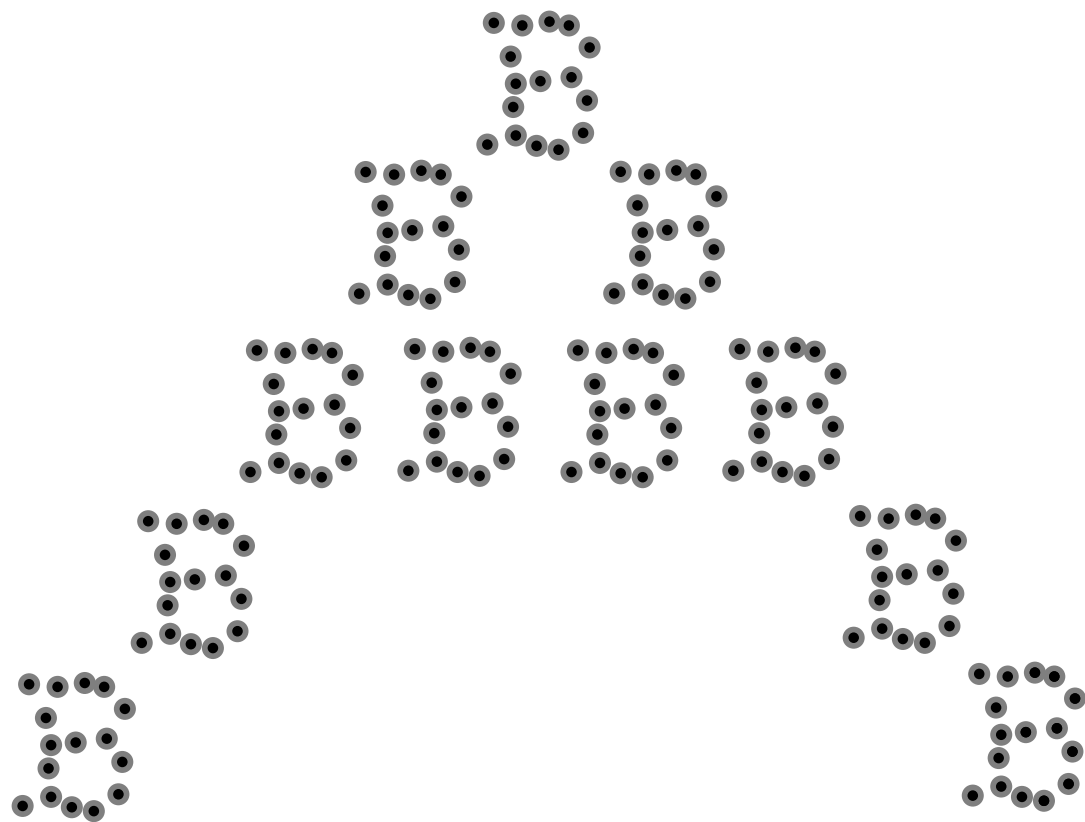
Persistence of Čech complexes



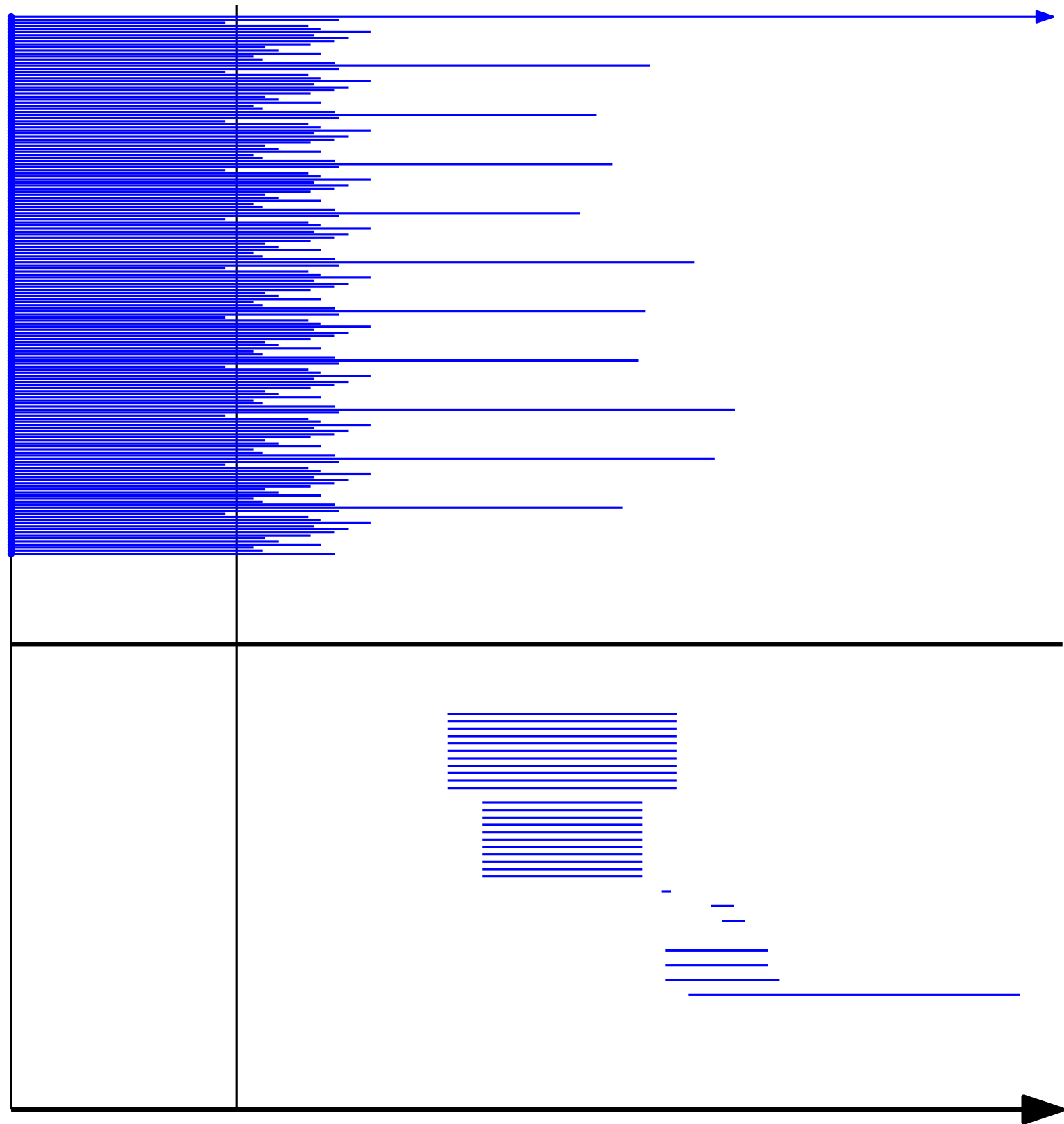
$$S = \mathbb{R}^2$$



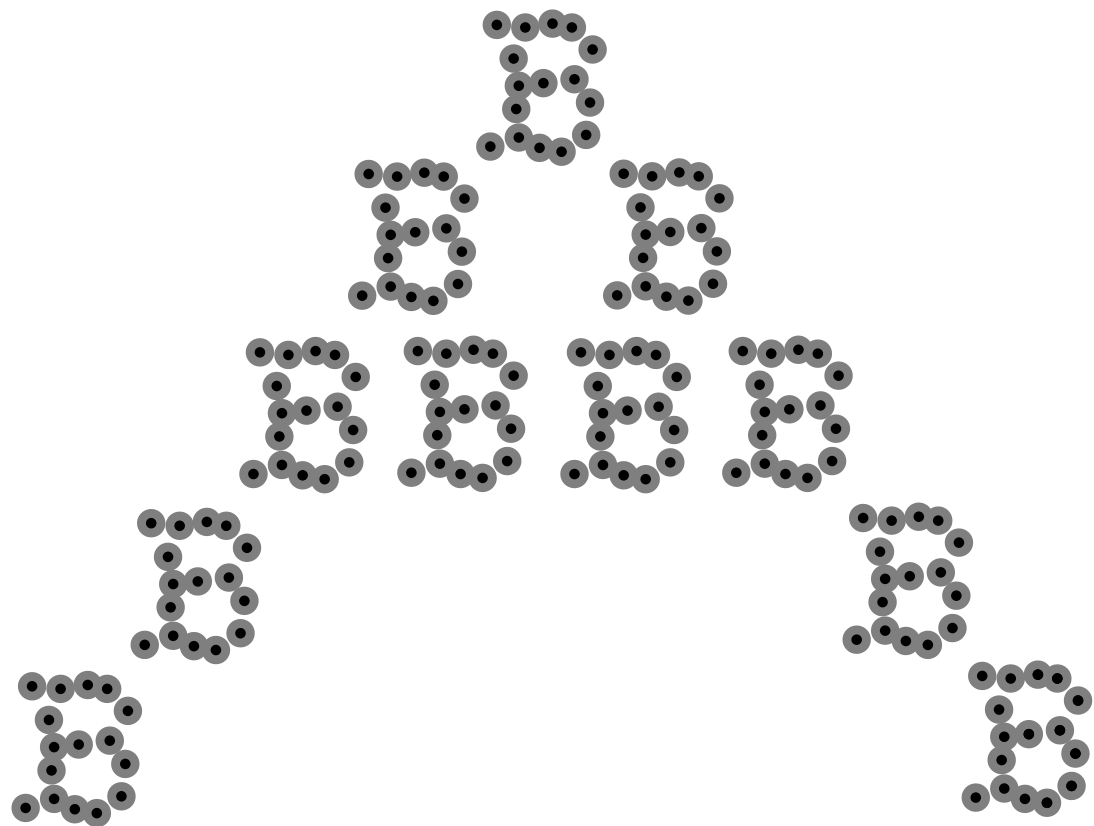
Persistence of Čech complexes



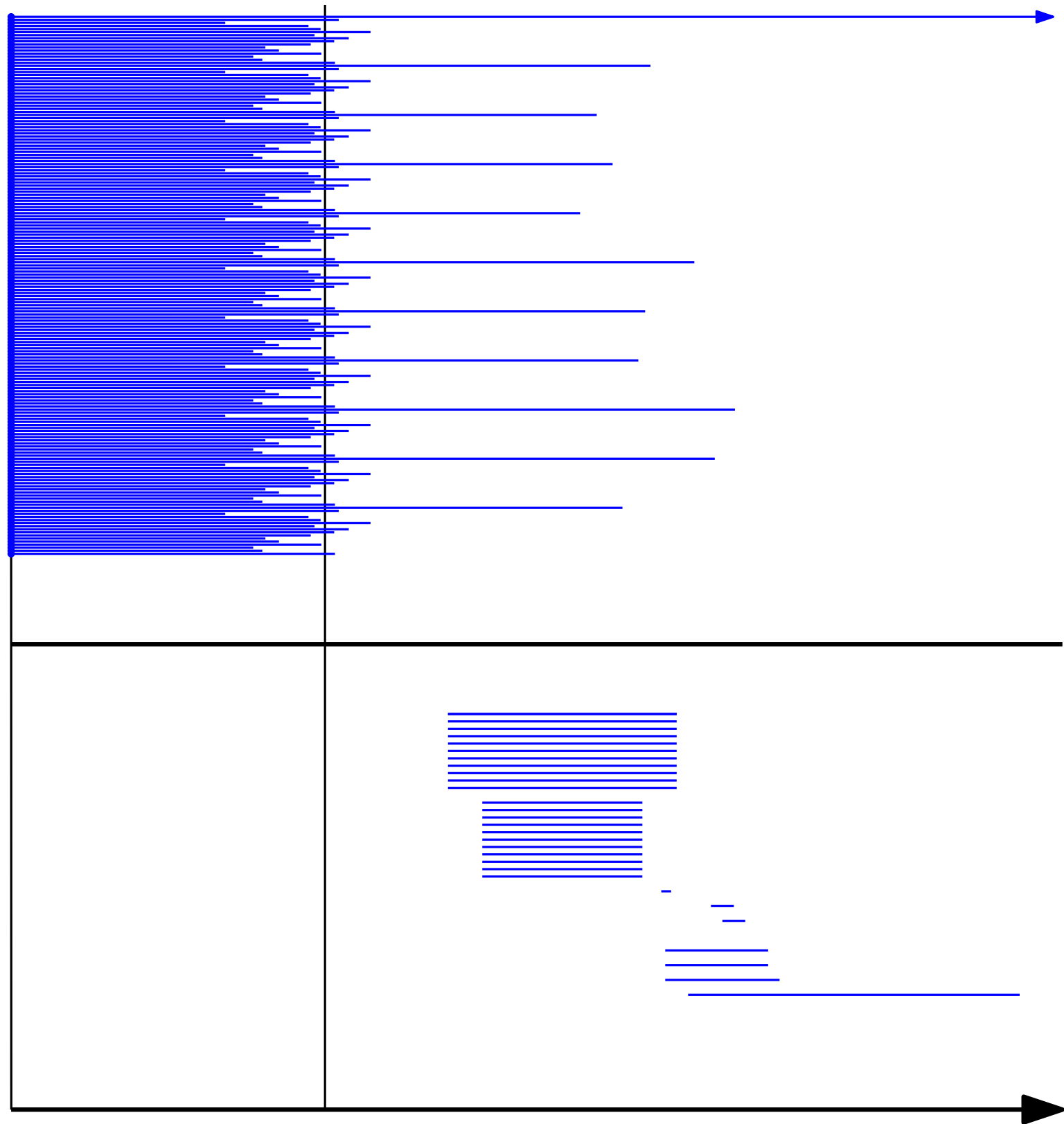
$$S = \mathbb{R}^2$$



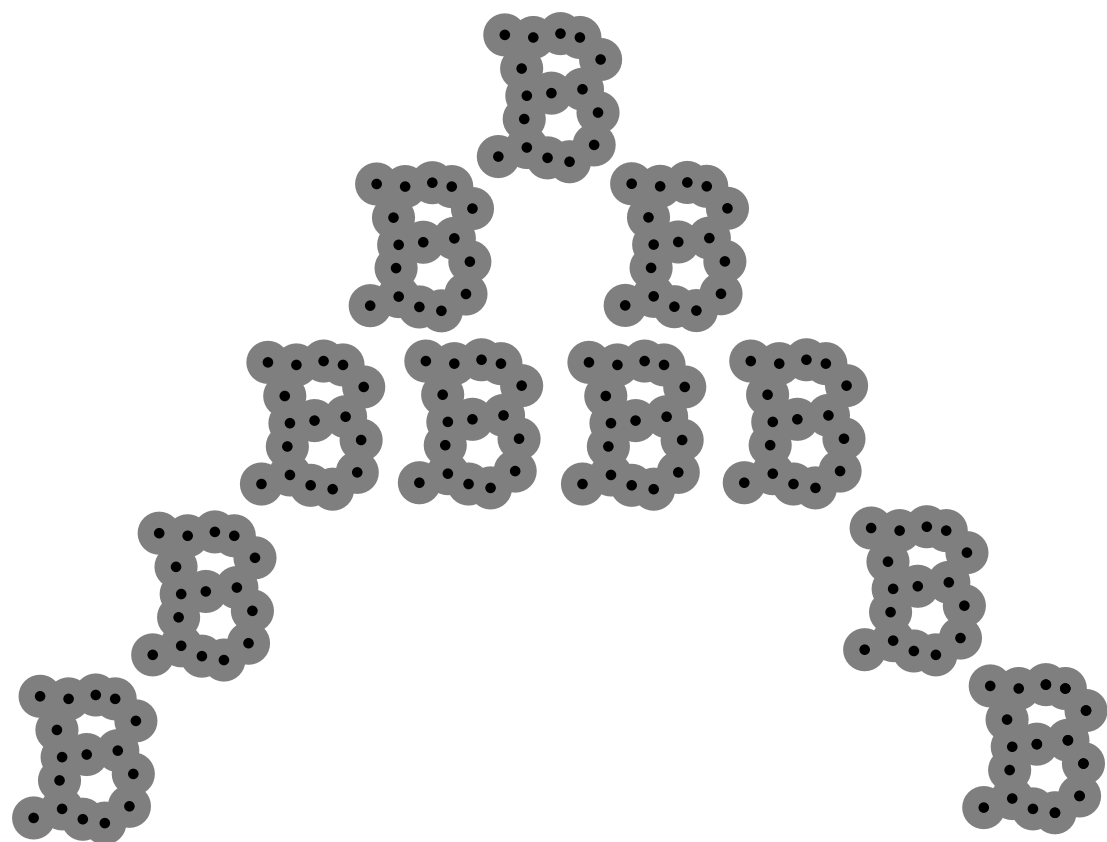
Persistence of Čech complexes



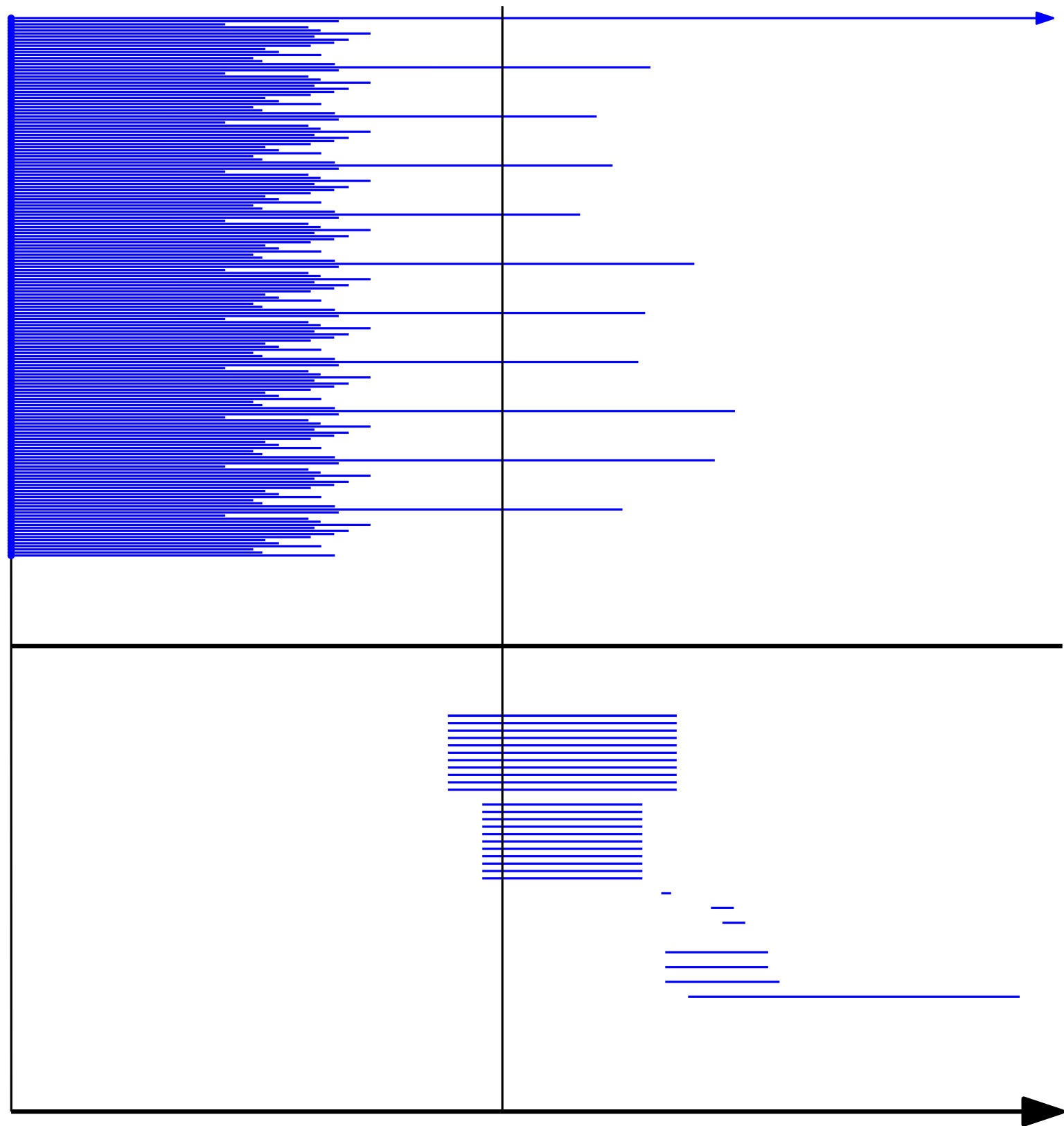
$$S = \mathbb{R}^2$$



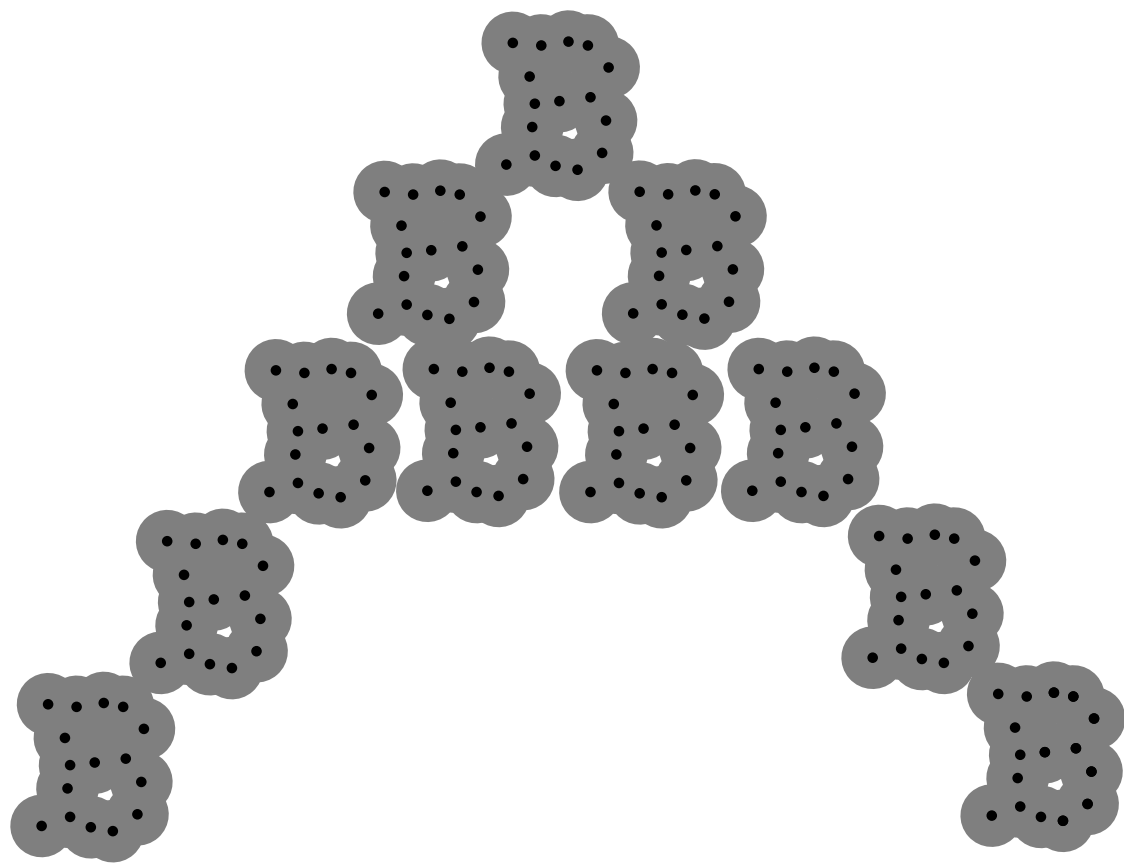
Persistence of Čech complexes



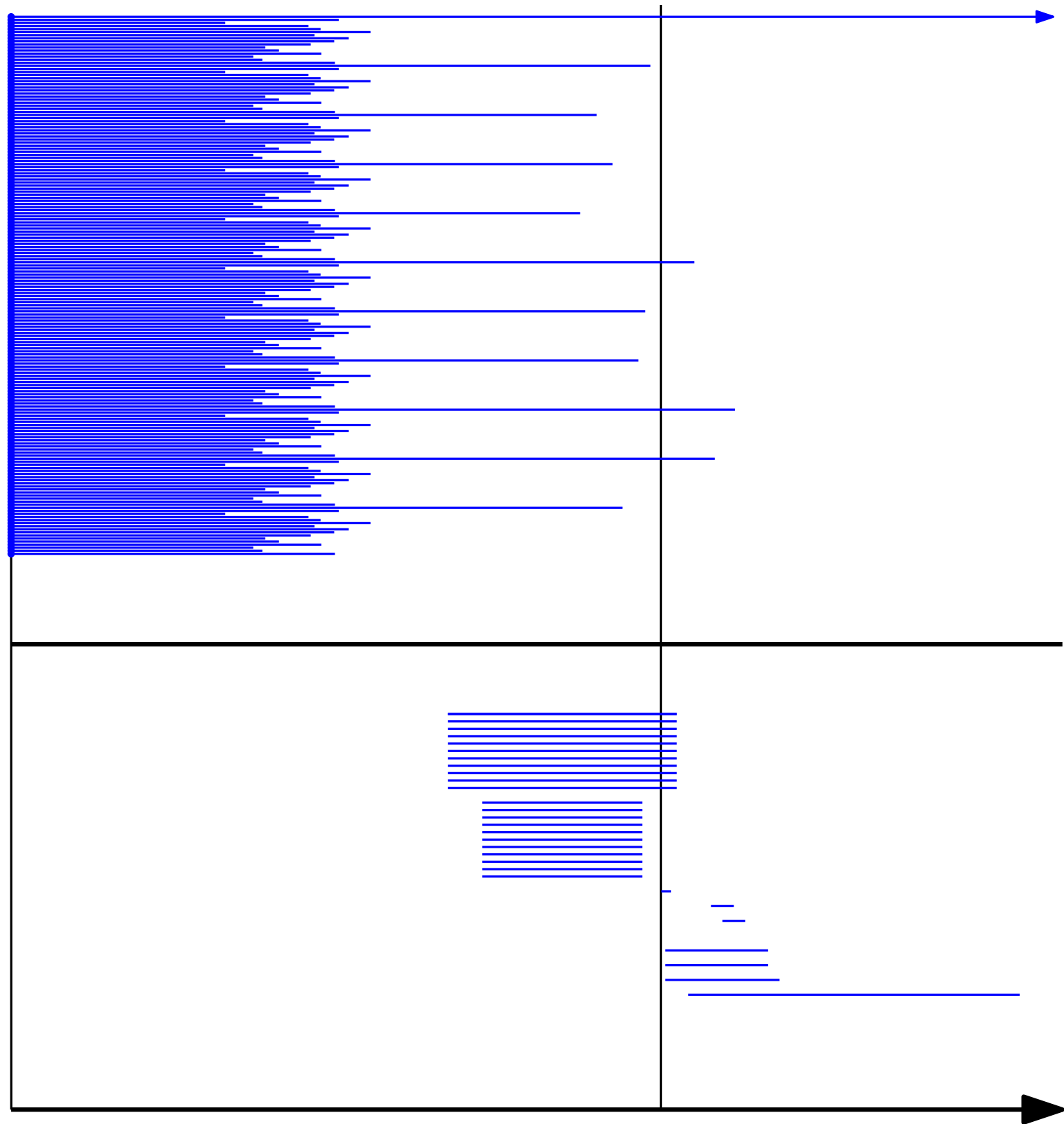
$$S = \mathbb{R}^2$$



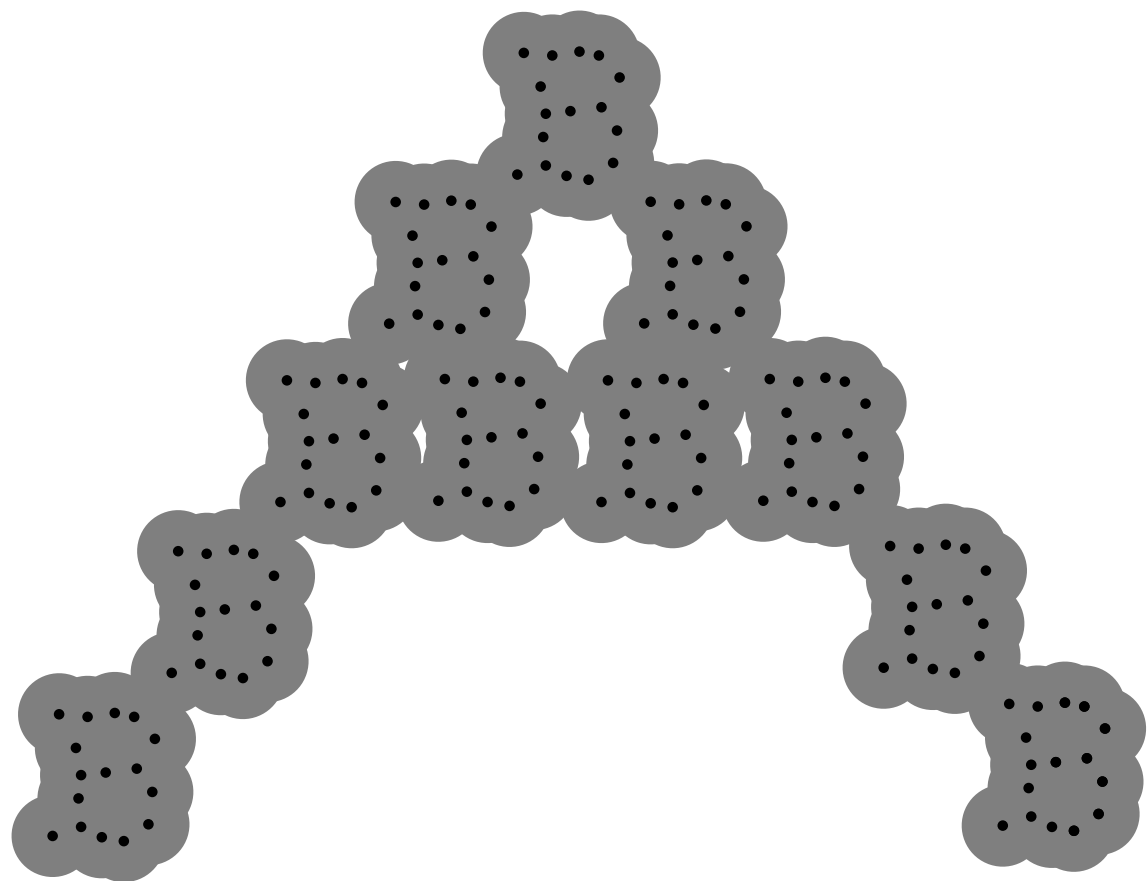
Persistence of Čech complexes



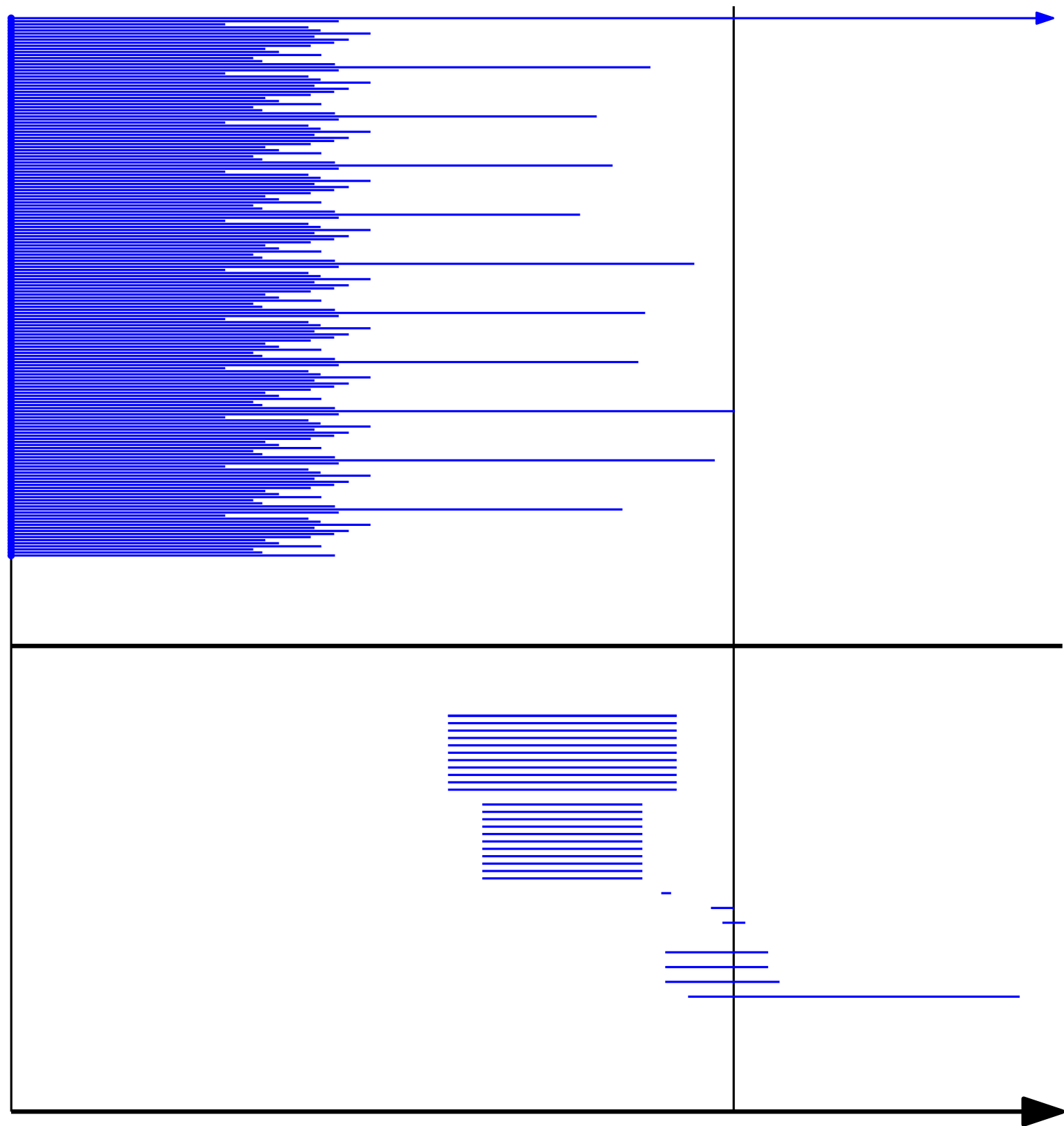
$$S = \mathbb{R}^2$$



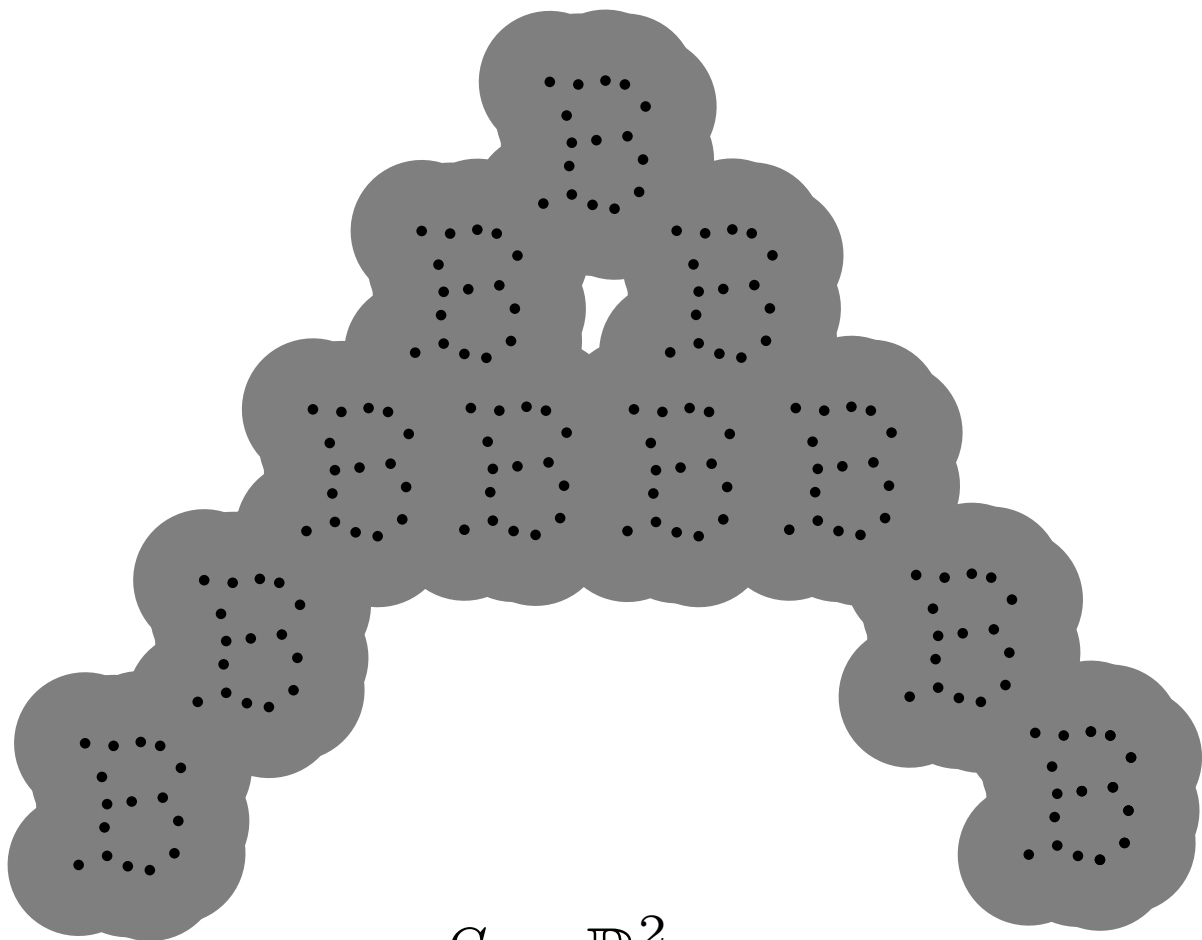
Persistence of Čech complexes



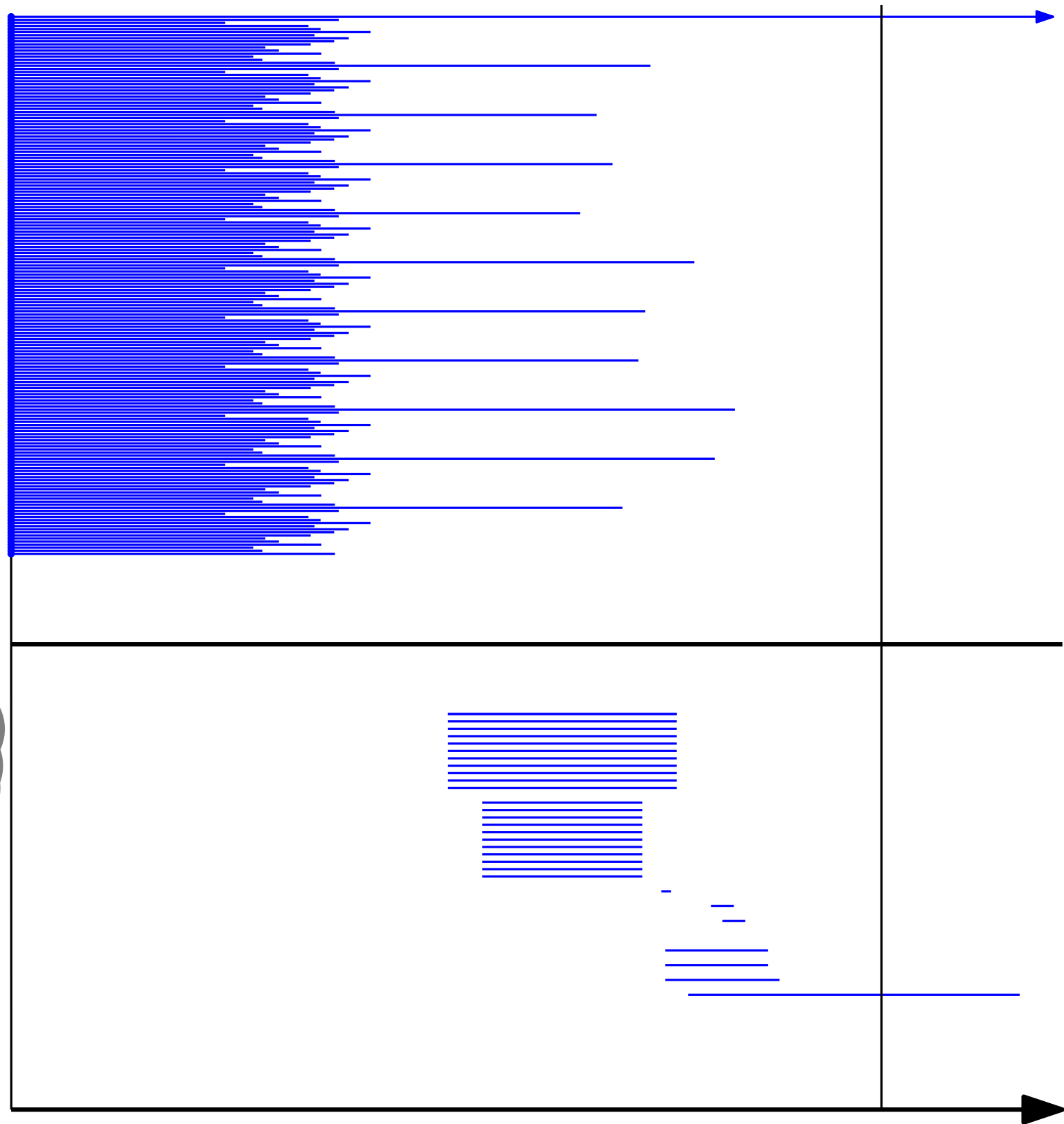
$$S = \mathbb{R}^2$$



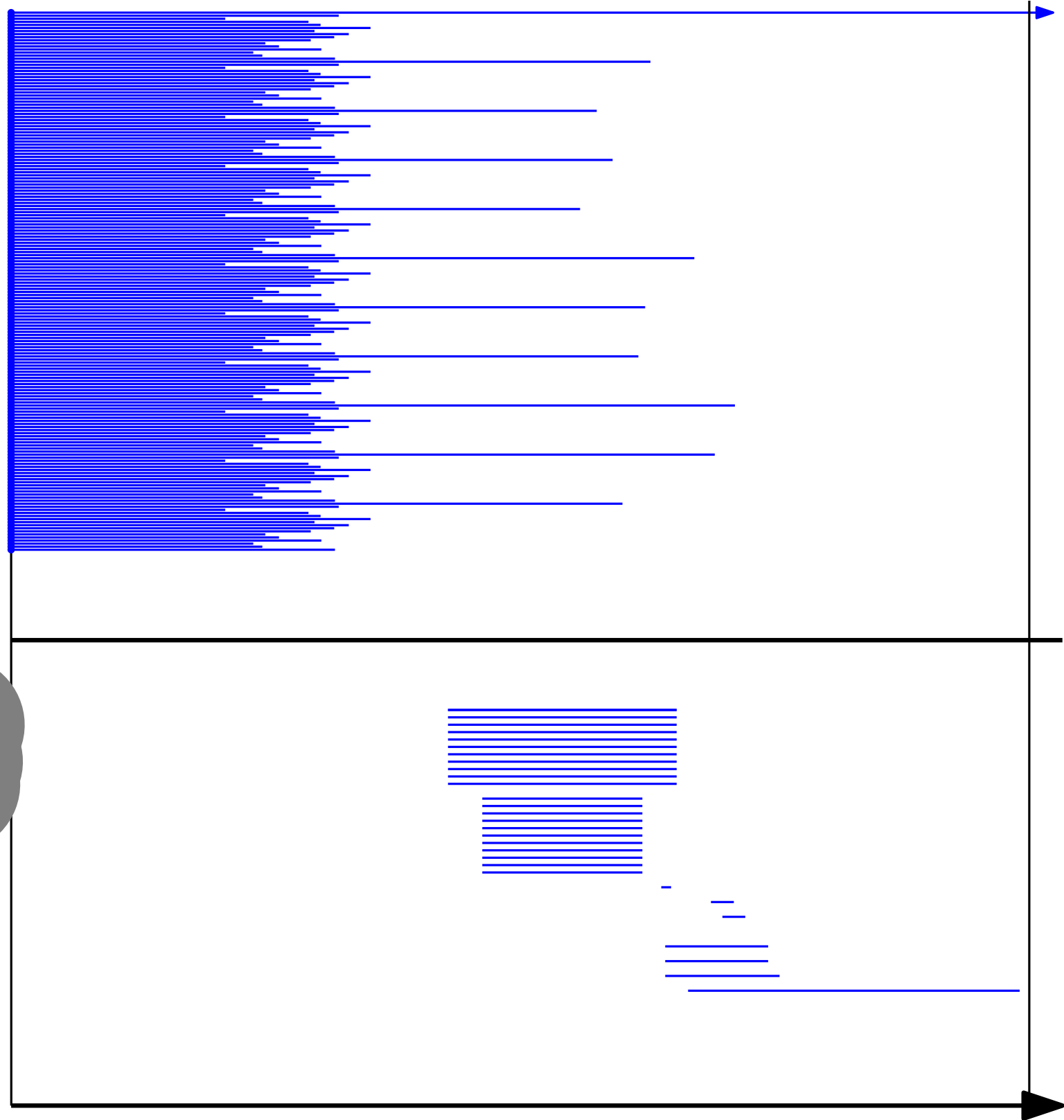
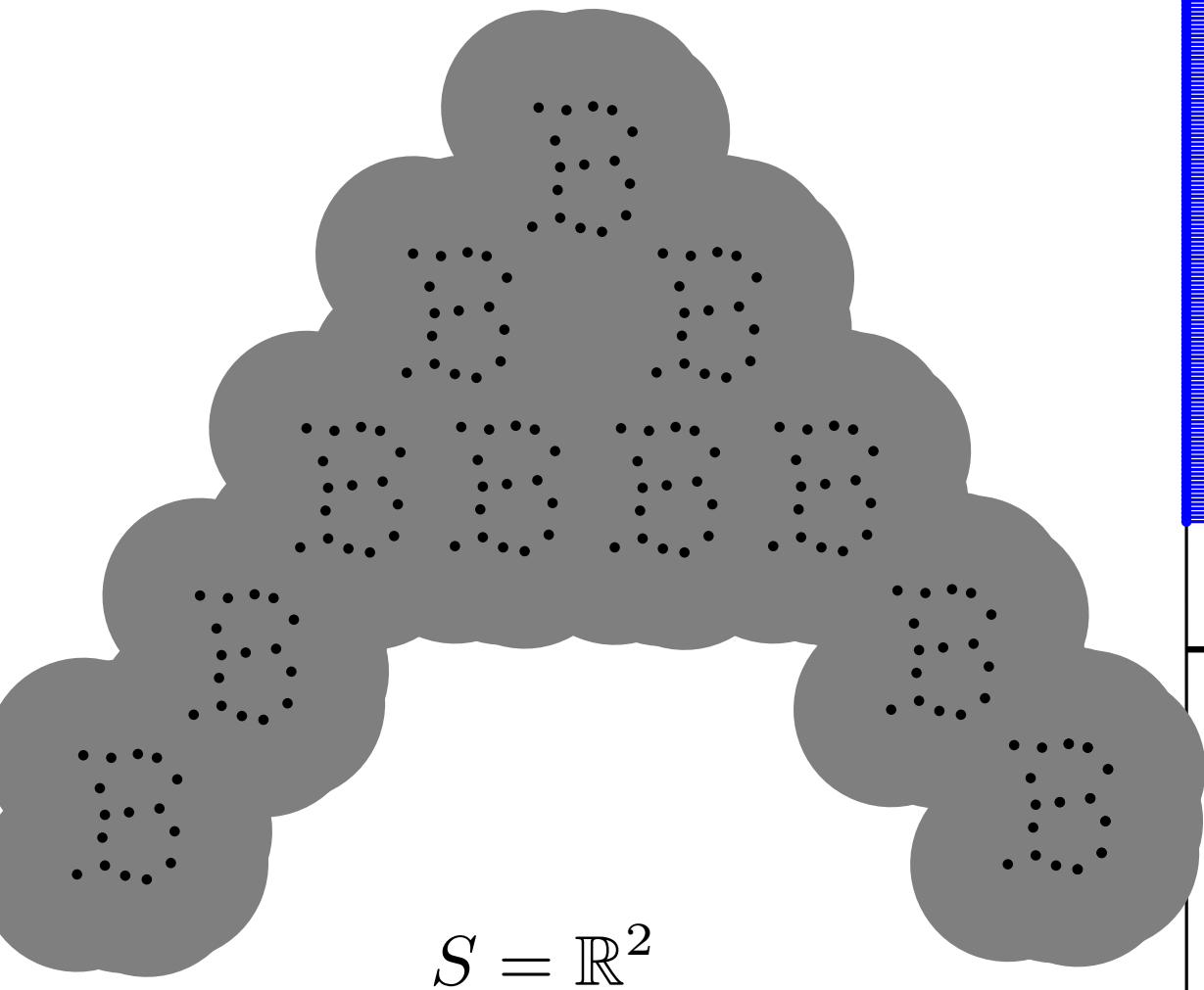
Persistence of Čech complexes



$$S = \mathbb{R}^2$$

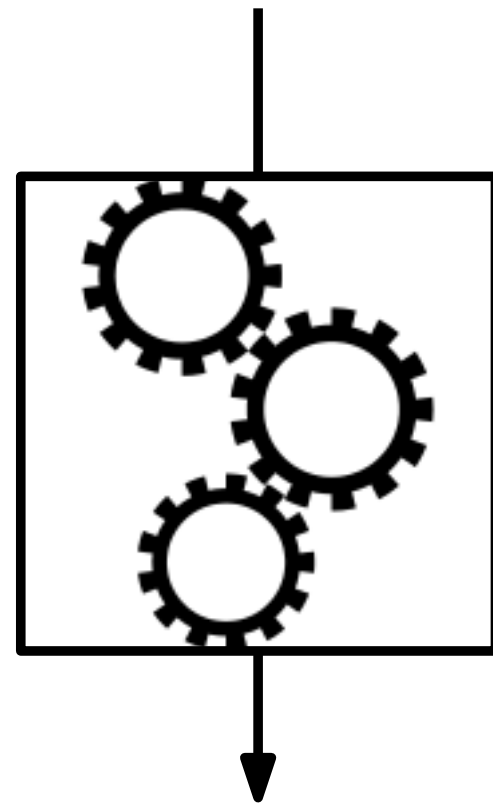


Persistence of Čech complexes



Algebraic foundations

Filtration: $F_1 \subseteq F_2 \subseteq F_3 \subseteq F_4 \subseteq F_5 \cdots$



topological level

(homology functor)

algebraic level

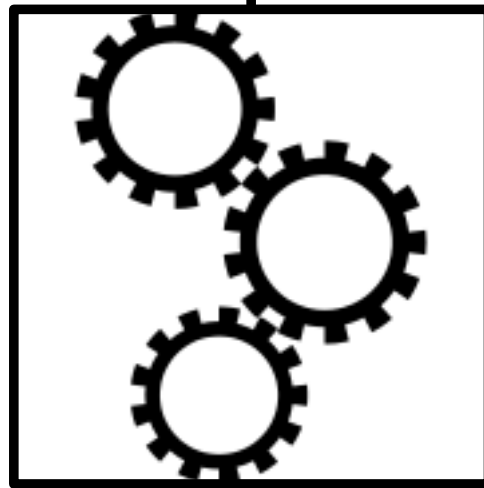
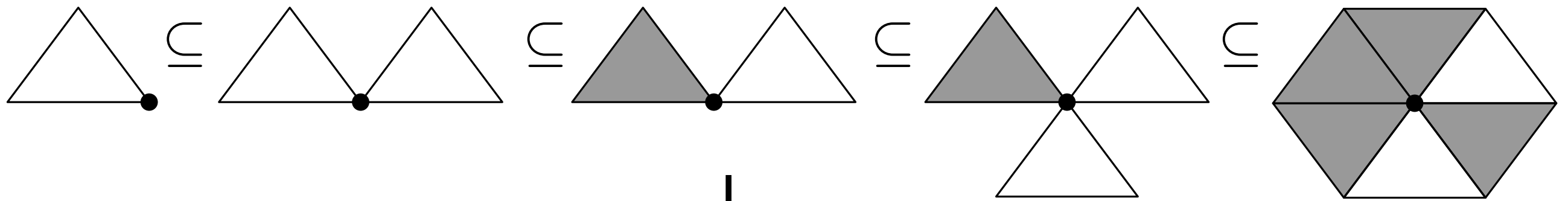
$$H_*(F_1) \rightarrow H_*(F_2) \rightarrow H_*(F_3) \rightarrow H_*(F_4) \rightarrow H_*(F_5) \rightarrow \cdots$$

Def: A *persistence module* is a sequence of vector spaces connected with linear maps:

$$H_*(F_1) \rightarrow H_*(F_2) \rightarrow H_*(F_3) \rightarrow H_*(F_4) \rightarrow \cdots$$

Algebraic foundations

Ex:

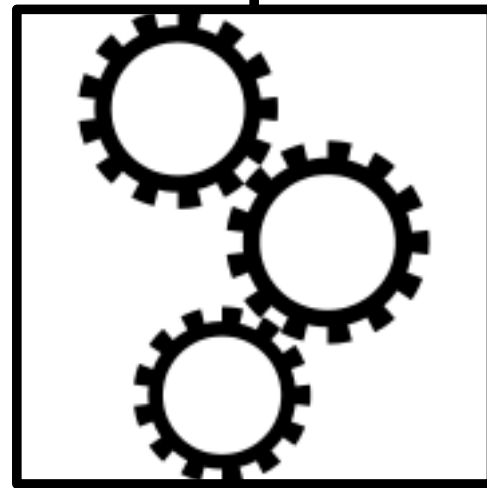
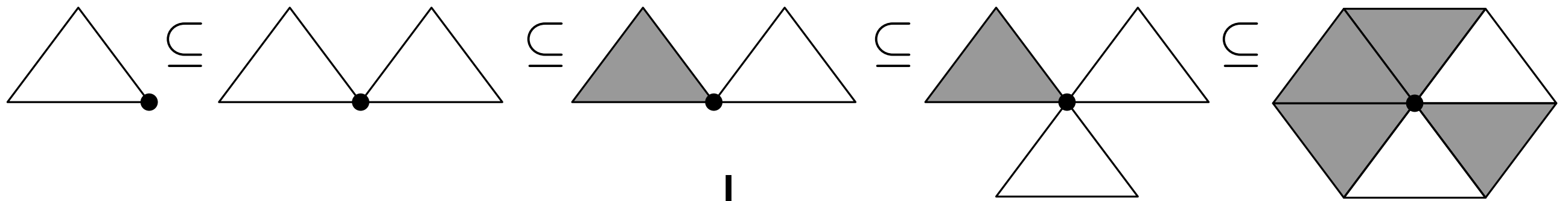


(degree-1 homology)

$$\mathbf{k} \xrightarrow{\begin{pmatrix} 1 \\ 0 \end{pmatrix}} \mathbf{k}^2 \xrightarrow{\begin{pmatrix} 0 & 1 \end{pmatrix}} \mathbf{k}$$

Algebraic foundations

Ex:

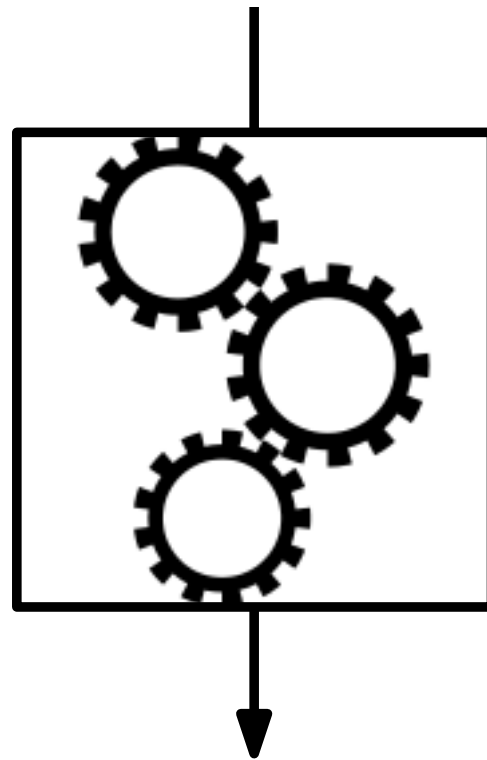
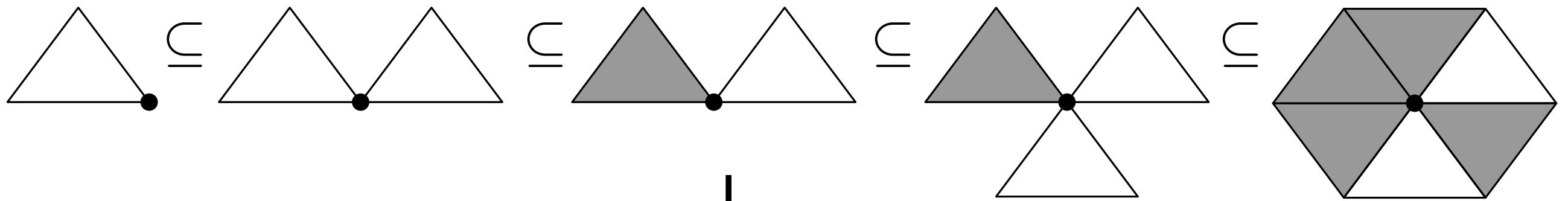


(degree-1 homology)

$$\mathbf{k} \xrightarrow{\begin{pmatrix} 1 \\ 0 \end{pmatrix}} \mathbf{k}^2 \xrightarrow{\begin{pmatrix} 0 & 1 \end{pmatrix}} \mathbf{k} \xrightarrow{\begin{pmatrix} 0 \\ 1 \end{pmatrix}} \mathbf{k}^2$$

Algebraic foundations

Ex:



(degree-1 homology)

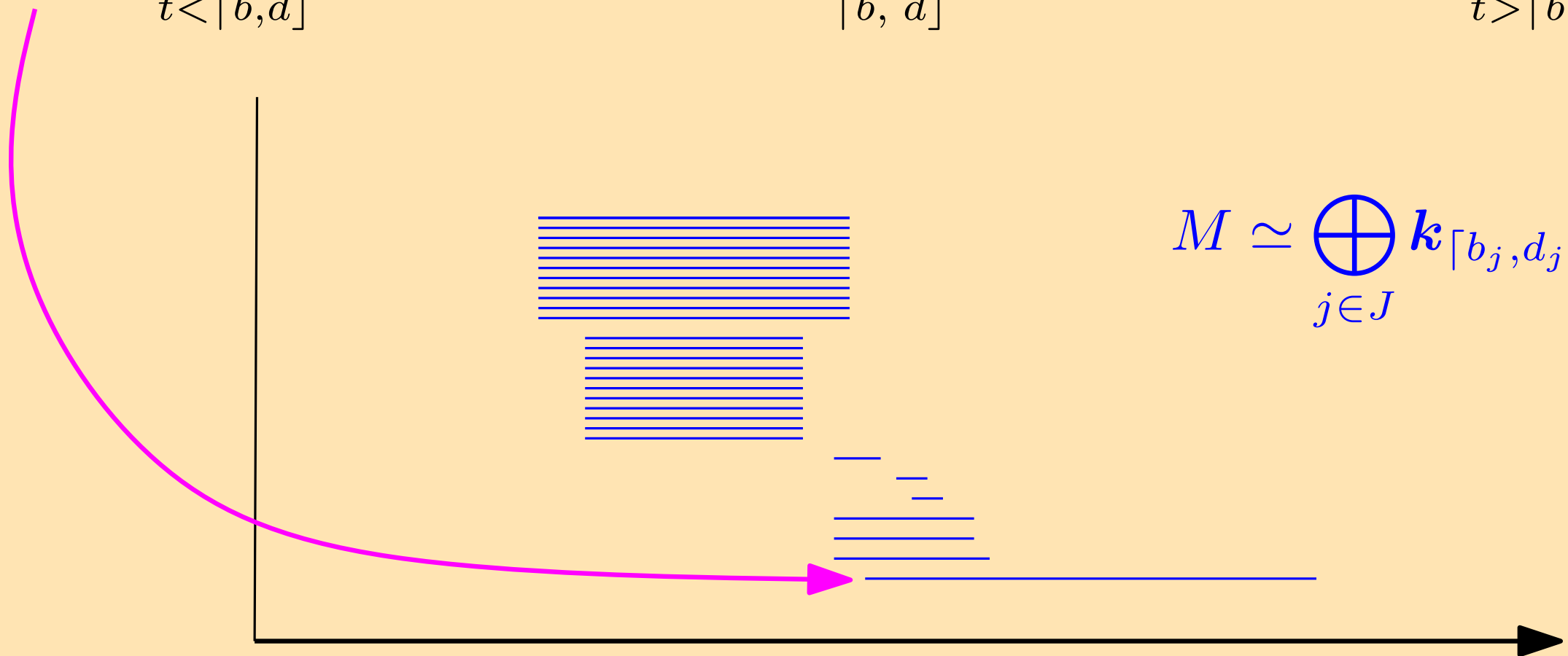
$$\mathbf{k} \xrightarrow{\begin{pmatrix} 1 \\ 0 \end{pmatrix}} \mathbf{k}^2 \xrightarrow{\begin{pmatrix} 0 & 1 \end{pmatrix}} \mathbf{k} \xrightarrow{\begin{pmatrix} 0 \\ 1 \end{pmatrix}} \mathbf{k}^2 \xrightarrow{\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}} \mathbf{k}^2 \dots$$

Algebraic foundations

[The structure and stability of persistence modules, Chazal, de Silva, Glisse, Oudot, Springer, 2016].

Thm: Let M be a persistence module over an index set $T \subseteq \mathbb{R}$. Then, M decomposes as a direct sum of *interval modules* $k_{[b,d]}$:

$$\underbrace{0 \xrightarrow{0} \dots \xrightarrow{0} 0}_{t < [b,d]} \xrightarrow{0} \underbrace{k \xrightarrow{1} \dots \xrightarrow{1} k}_{[b,d]} \xrightarrow{0} \underbrace{0 \xrightarrow{0} \dots \xrightarrow{0} 0}_{t > [b,d]}$$



(the barcode is a complete descriptor of the algebraic structure of M)

Algebraic foundations

[*The structure and stability of persistence modules*, Chazal, de Silva, Glisse, Oudot, Springer, 2016].

Thm: Let M be a persistence module over an index set $T \subseteq \mathbb{R}$. Then, M decomposes as a direct sum of *interval modules* $k_{[b,d]}$:

$$\underbrace{0 \xrightarrow{0} \dots \xrightarrow{0} 0}_{t < [b,d]} \xrightarrow{0} \underbrace{k \xrightarrow{1} \dots \xrightarrow{1} k}_{[b,d]} \xrightarrow{0} \underbrace{0 \xrightarrow{0} \dots \xrightarrow{0} 0}_{t > [b,d]}$$

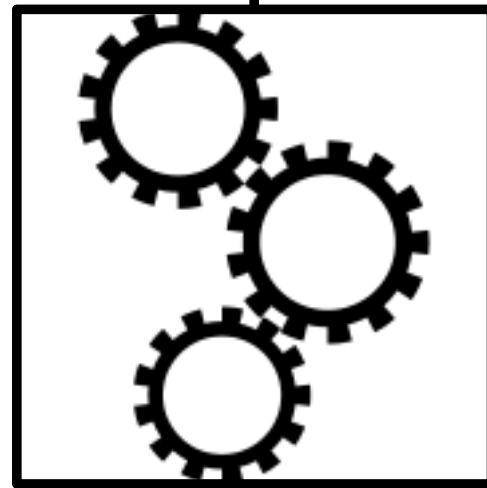
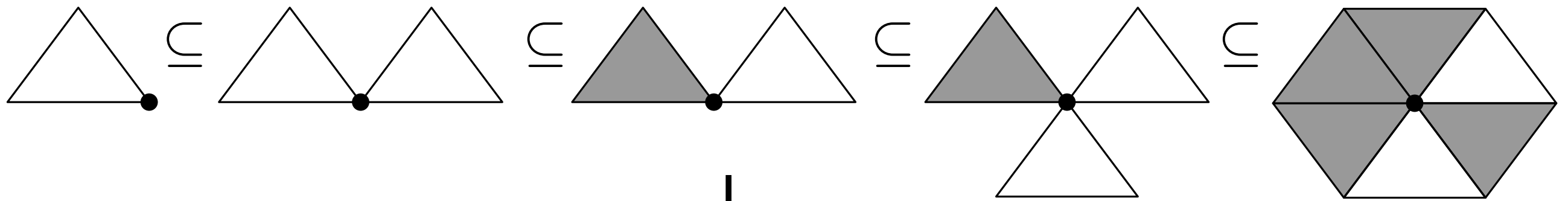
in the following cases:

- T is finite,
- M is *pointwise finite-dimensional* (pfd), i.e., every space M_t has finite dimension.

Moreover, when it exists, the decomposition is **unique** up to isomorphism and permutation of the terms.

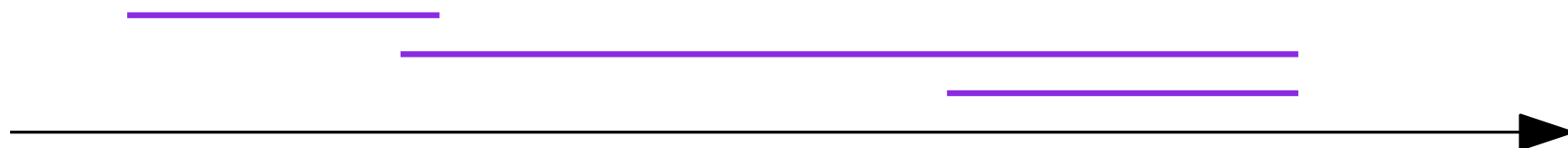
Algebraic foundations

Ex:



(degree-1 homology)

$$\mathbf{k} \xrightarrow{\begin{pmatrix} 1 \\ 0 \end{pmatrix}} \mathbf{k}^2 \xrightarrow{\begin{pmatrix} 0 & 1 \end{pmatrix}} \mathbf{k} \xrightarrow{\begin{pmatrix} 0 \\ 1 \end{pmatrix}} \mathbf{k}^2 \xrightarrow{\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}} \mathbf{k}^2 \dots$$



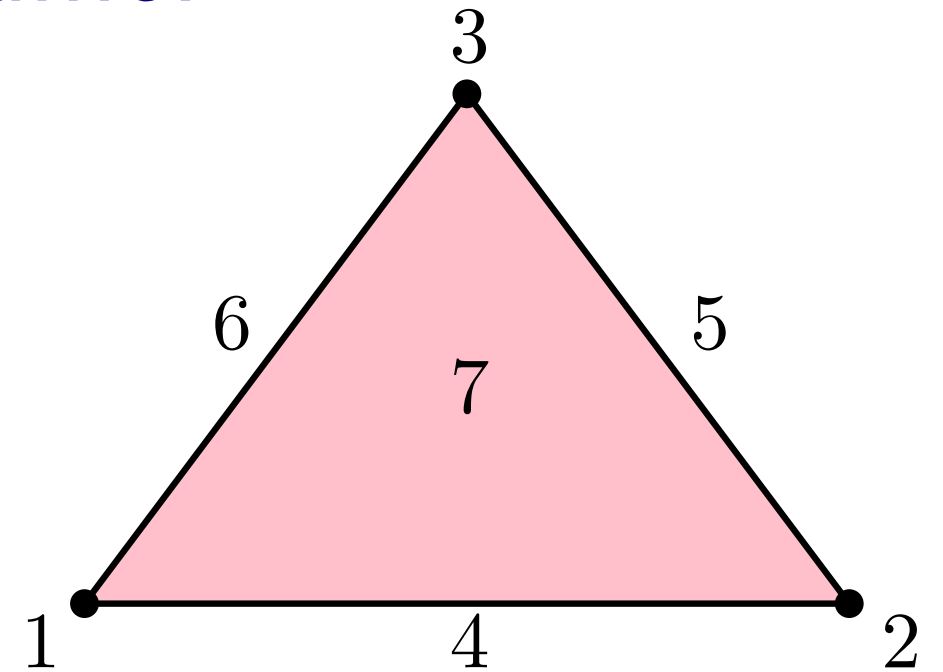
Good news: the algorithm is the same!

Input: simplicial filtration

Output: boundary matrix
reduced to column-echelon form

○ simplex pairs give finite intervals:
[2, 4), [3, 5), [6, 7)

□ unpaired simplices give infinite intervals: $[1, +\infty)$



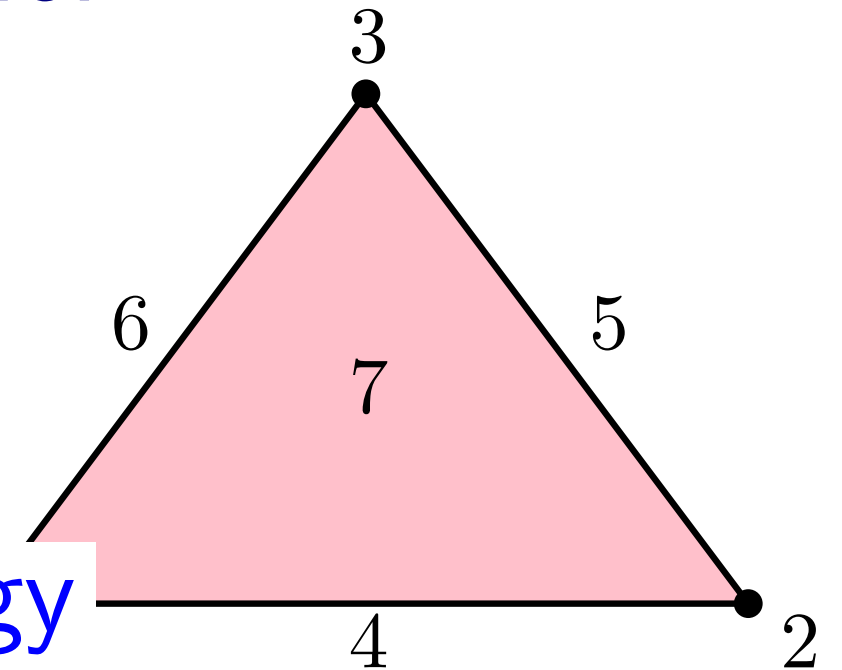
	1	2	3	4	5	6	7
1				*		*	
2				*	*		
3					*	*	
4							*
5							*
6							*
7							

	1	2	3	4	5	6	7
1				*			
2				①	*		
3					①		
4							*
5							*
6							①
7							

Good news: the algorithm is the same!

Input: simplicial filtration

Output: boundary matrix
reduced to column-echelon form



○ simplex pairs give **Persistent homology**
[2, 4), [3, 5), [6, 7)

□ unpaired simplices give **Regular homology**

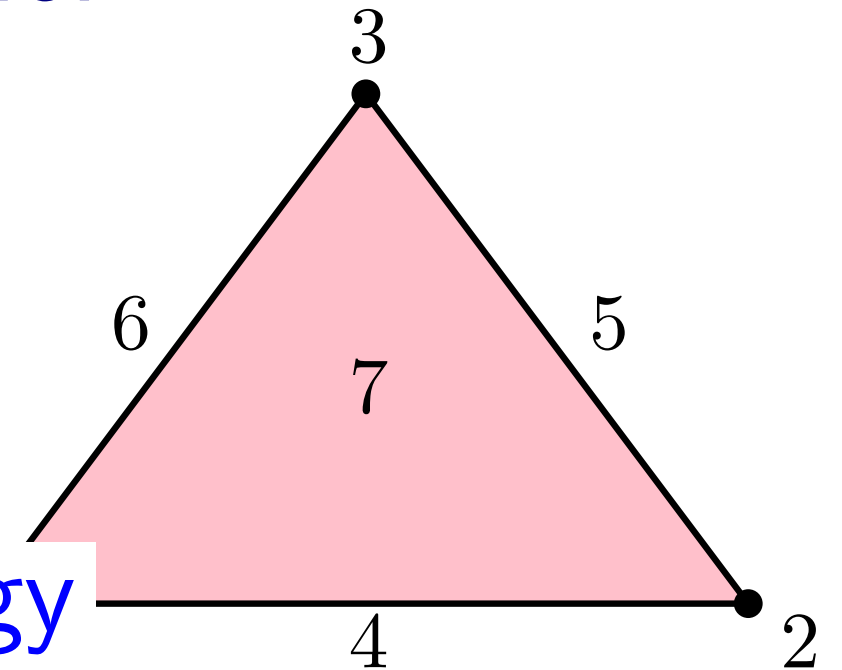
	1	2	3	4	5	6	7
1				*		*	
2				*	*		
3					*	*	
4							*
5							*
6							*
7							

	1	2	3	4	5	6	7
1				*			
2				①	*		
3					①		
4							*
5							*
6							①
7							

Good news: the algorithm is the same!

Input: simplicial filtration

Output: boundary matrix
reduced to column-echelon form



○ simplex pairs give **Persistent homology**
[2, 4), [3, 5), [6, 7)

□ unpaired simplices give **Regular homology**

	1	2	3	4	5	6	7
1				*		*	
2				*	*		
3					*	*	
4							*
5							*
6							*
7							

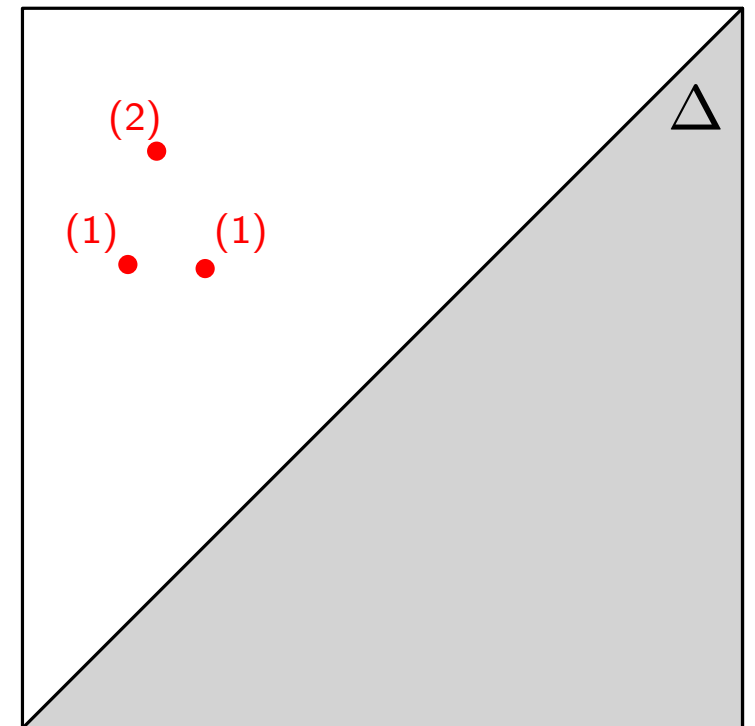
	1	2	3	4	5	6	7
1				*			
2				①	*		
3					①		
4							*
5							*
6							①
7							

Computational Topology (II): Persistence Theory

1. Algorithmic Foundation
2. Algebraic Foundation
3. **Stability Theorem**

Distance between persistence diagrams

Persistence diagram \equiv **finite** multiset in the open half-plane $\Delta \times \mathbb{R}_{>0}$.



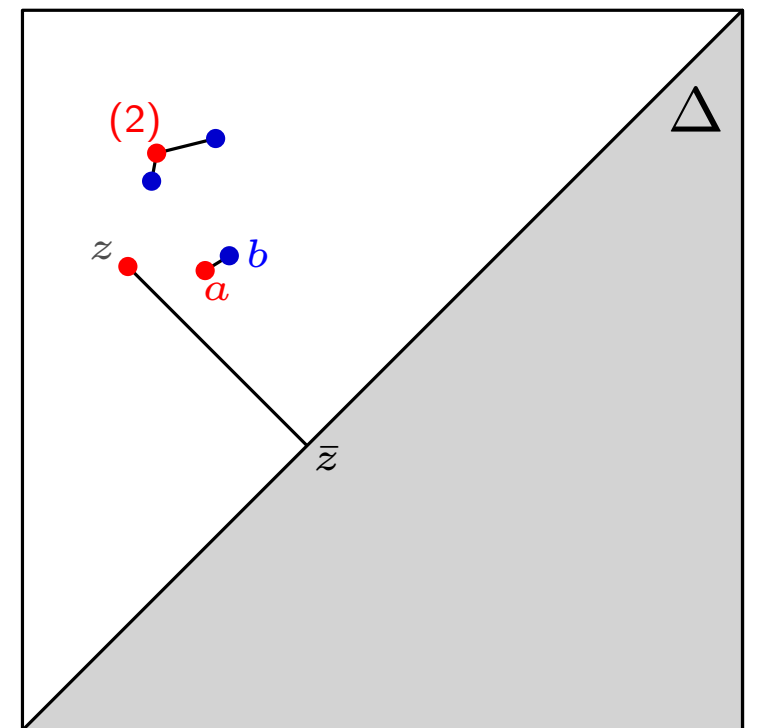
Distance between persistence diagrams

Persistence diagram \equiv **finite** multiset in the open half-plane $\Delta \times \mathbb{R}_{>0}$.

Given a **partial matching** $M : D \leftrightarrow D'$:

- cost of a matched pair $(a, b) \in M$: $c_p(a, b) := \|a - b\|_\infty^p$,
- cost of an unmatched point $c \in D \sqcup D'$: $c_p(c) := \|c - \bar{c}\|_\infty^p$,
- **cost of M** :

$$c_p(M) := \left(\sum_{(a, b) \text{ matched}} c_p(a, b) + \sum_{c \text{ unmatched}} c_p(c) \right)^{1/p}$$



Distance between persistence diagrams

Persistence diagram \equiv **finite** multiset in the open half-plane $\Delta \times \mathbb{R}_{>0}$.

Given a **partial matching** $M : D \leftrightarrow D'$:

- cost of a matched pair $(a, b) \in M$: $c_p(a, b) := \|a - b\|_\infty^p$,
- cost of an unmatched point $c \in D \sqcup D'$: $c_p(c) := \|c - \bar{c}\|_\infty^p$,
- **cost of M** :

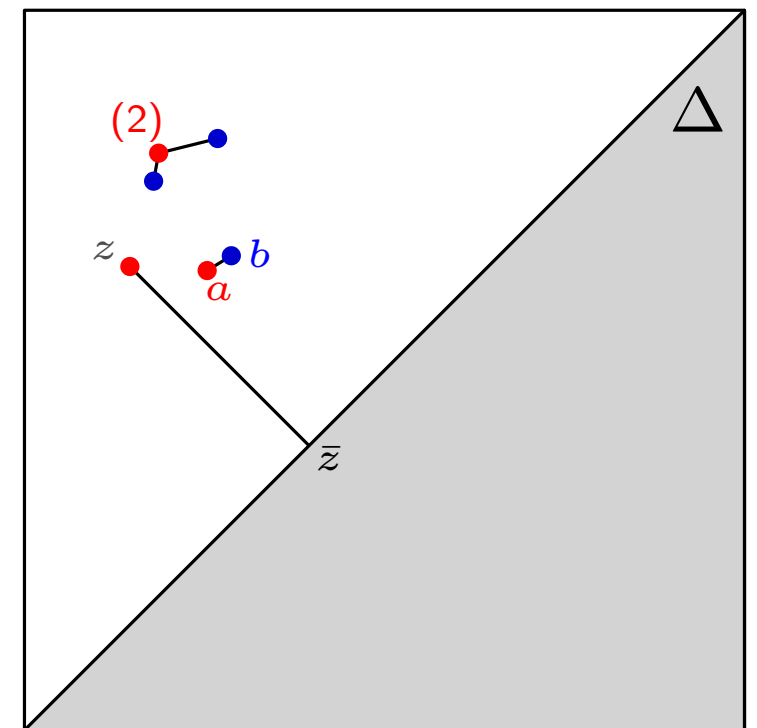
$$c_p(M) := \left(\sum_{(a, b) \text{ matched}} c_p(a, b) + \sum_{c \text{ unmatched}} c_p(c) \right)^{1/p}$$

Def: p -th diagram distance (extended metric):

$$d_p(D, D') := \inf_{M: D \leftrightarrow D'} c_p(M)$$

Def: bottleneck distance:

$$d_b(D, D') = d_\infty(D, D') := \lim_{p \rightarrow \infty} d_p(D, D')$$

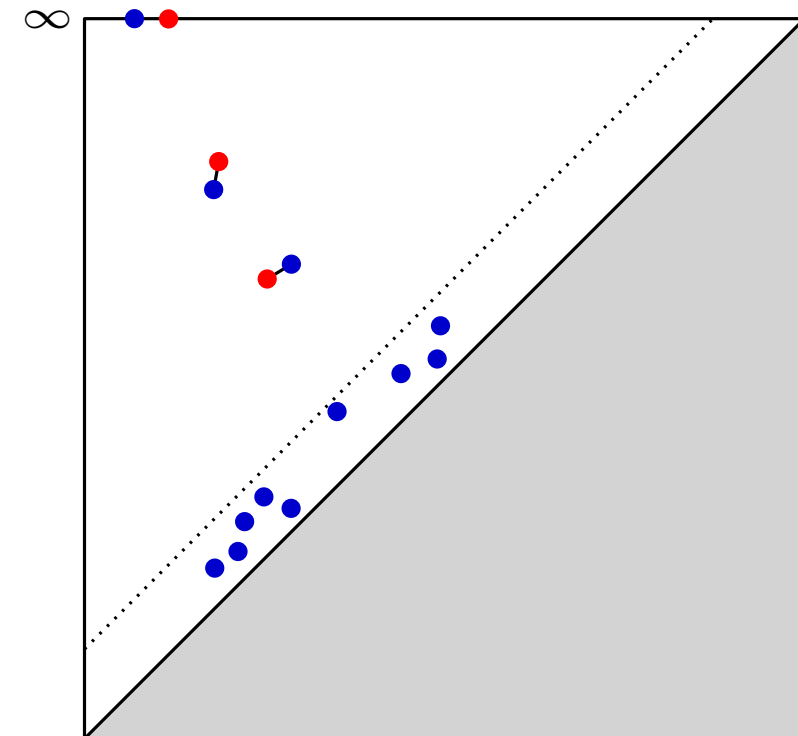
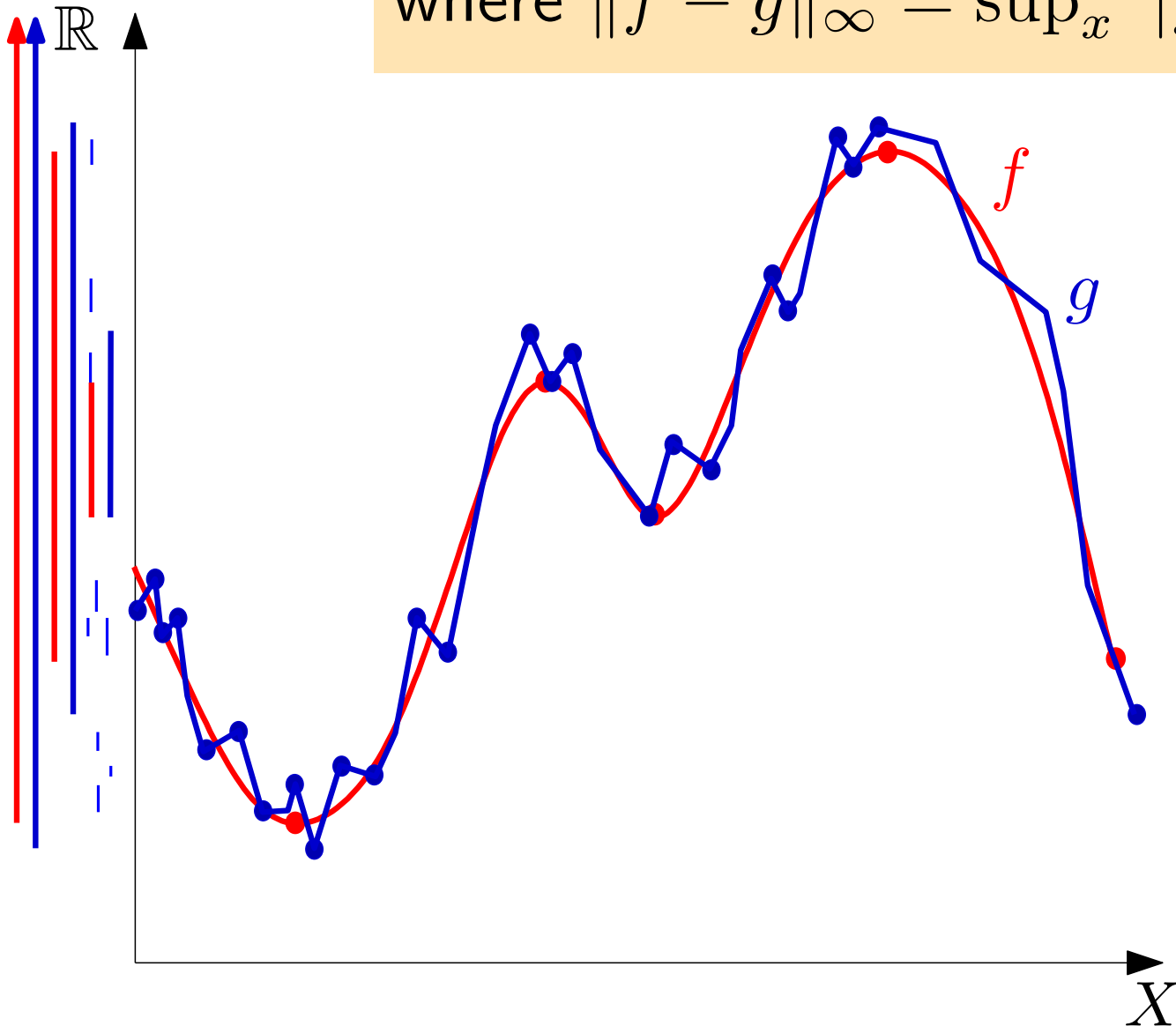


Stability Theorem

Thm: For any pfd functions $f, g : X \rightarrow \mathbb{R}$ and homology dimension d ,

$$d_b(D_f, D_g) \leq \|f - g\|_\infty,$$

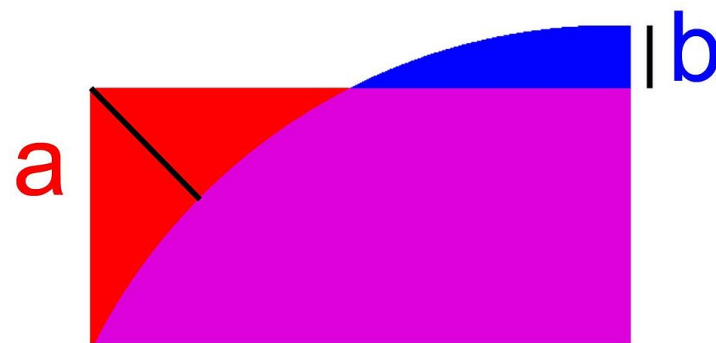
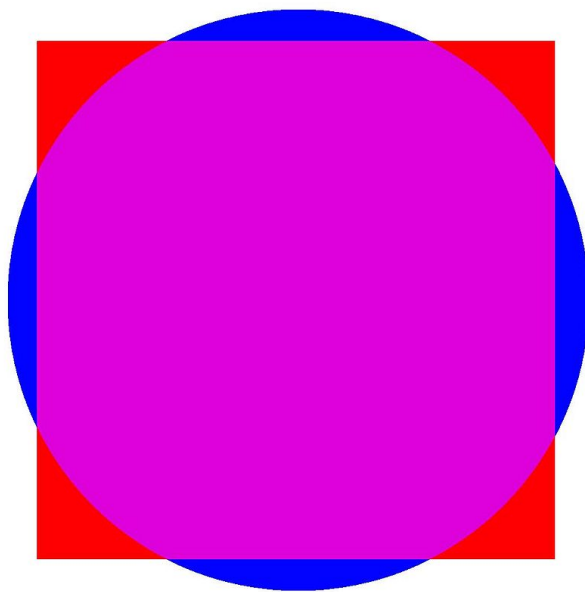
where $\|f - g\|_\infty = \sup_x |f(x) - g(x)|$.



Stability properties for point clouds

Def: The **Hausdorff distance** between two subspaces X, Y of a common metric space (Z, d) is:

$$\begin{aligned}d_H(X, Y) &= \max\{\sup_{y \in Y} d(y, X), \sup_{x \in X} d(x, Y)\} \\ &= \max\{\sup_{y \in Y} \inf_{x \in X} d(y, x), \sup_{x \in X} \inf_{y \in Y} d(x, y)\}\end{aligned}$$



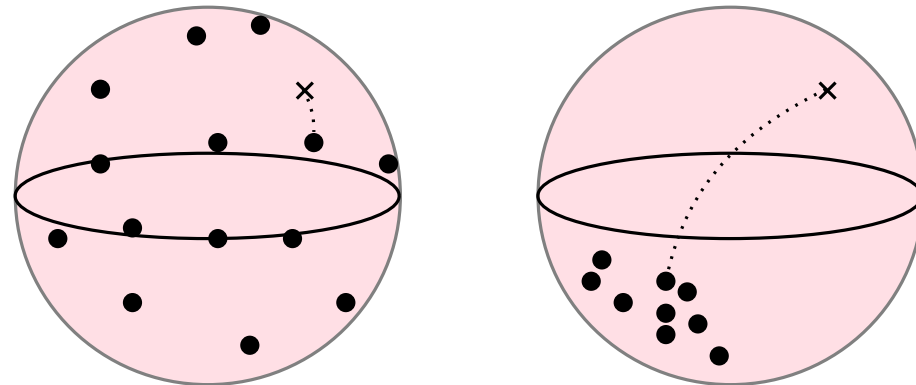
$$d_H(X, Y) = \max\{a, b\}$$

Stability properties for point clouds

Def: The **Hausdorff distance** between two subspaces X, Y of a common metric space (Z, d) is:

$$\begin{aligned}d_H(X, Y) &= \max\{\sup_{y \in Y} d(y, X), \sup_{x \in X} d(x, Y)\} \\ &= \max\{\sup_{y \in Y} \inf_{x \in X} d(y, x), \sup_{x \in X} \inf_{y \in Y} d(x, y)\}\end{aligned}$$

Ex: Given a sampling $\hat{X}_n \subseteq X$, $d_H(\hat{X}_n, X)$ is a measure of sampling quality.



Q: Show that $d_H(X, Y) = \inf\{\epsilon > 0 : X^\epsilon \subseteq Y \text{ and } Y^\epsilon \subseteq X\}$, where $X^\epsilon = \{z : \exists x \in X \text{ s.t. } d(x, z) \leq \epsilon\}$.

Stability properties for point clouds

Def: The **Hausdorff distance** between two subspaces X, Y of a common metric space (Z, d) is:

$$\begin{aligned}d_H(X, Y) &= \max\{\sup_{y \in Y} d(y, X), \sup_{x \in X} d(x, Y)\} \\ &= \max\{\sup_{y \in Y} \inf_{x \in X} d(y, x), \sup_{x \in X} \inf_{y \in Y} d(x, y)\}\end{aligned}$$

Def: The **Gromov-Hausdorff distance** between metric spaces $(X, d_X), (Y, d_Y)$ is the Hausdorff distance of the best common isometric embedding:

$$d_{GH}((X, d_X), (Y, d_Y)) = \inf_{\gamma} d_H(\gamma(X), \gamma(Y)),$$

where $d(\gamma(x), \gamma(x')) = d_X(x, x')$ and $d(\gamma(y), \gamma(y')) = d_Y(y, y')$.

Stability properties for point clouds

Def: The **Hausdorff distance** between two subspaces X, Y of a common metric space (Z, d) is:

$$\begin{aligned}d_H(X, Y) &= \max\{\sup_{y \in Y} d(y, X), \sup_{x \in X} d(x, Y)\} \\ &= \max\{\sup_{y \in Y} \inf_{x \in X} d(y, x), \sup_{x \in X} \inf_{y \in Y} d(x, y)\}\end{aligned}$$

Def: The **Gromov-Hausdorff distance** between metric spaces $(X, d_X), (Y, d_Y)$ is metric distortion of the best correspondence:

$$d_{GH}((X, d_X), (Y, d_Y)) = \inf_{\mathcal{C}} \sup_{(x, y), (x', y') \in \mathcal{C}} |d_X(x, x') - d_Y(y, y')|,$$

where $\mathcal{C} \subseteq X \times Y$ s.t. $\forall x, \exists y_x \in Y$ s.t. $(x, y_x) \in \mathcal{C}$ (and vice-versa).

Stability properties for point clouds

Def: The **Hausdorff distance** between two subspaces X, Y of a common metric space (Z, d) is:

$$\begin{aligned}d_H(X, Y) &= \max\{\sup_{y \in Y} d(y, X), \sup_{x \in X} d(x, Y)\} \\ &= \max\{\sup_{y \in Y} \inf_{x \in X} d(y, x), \sup_{x \in X} \inf_{y \in Y} d(x, y)\}\end{aligned}$$

Def: The **Gromov-Hausdorff distance** between metric spaces $(X, d_X), (Y, d_Y)$ is metric distortion of the best correspondence:

$$d_{GH}((X, d_X), (Y, d_Y)) = \inf_{\mathcal{C}} \sup_{(x,y), (x',y') \in \mathcal{C}} |d_X(x, x') - d_Y(y, y')|,$$

where $\mathcal{C} \subseteq X \times Y$ s.t. $\forall x, \exists y_x \in Y$ s.t. $(x, y_x) \in \mathcal{C}$ (and vice-versa).

Thm: If X and Y are common subspaces of a common metric space (Z, d) , then

$$d_b(D_{\text{Cech}}(X), D_{\text{Cech}}(Y)) \leq d_H(X, Y).$$

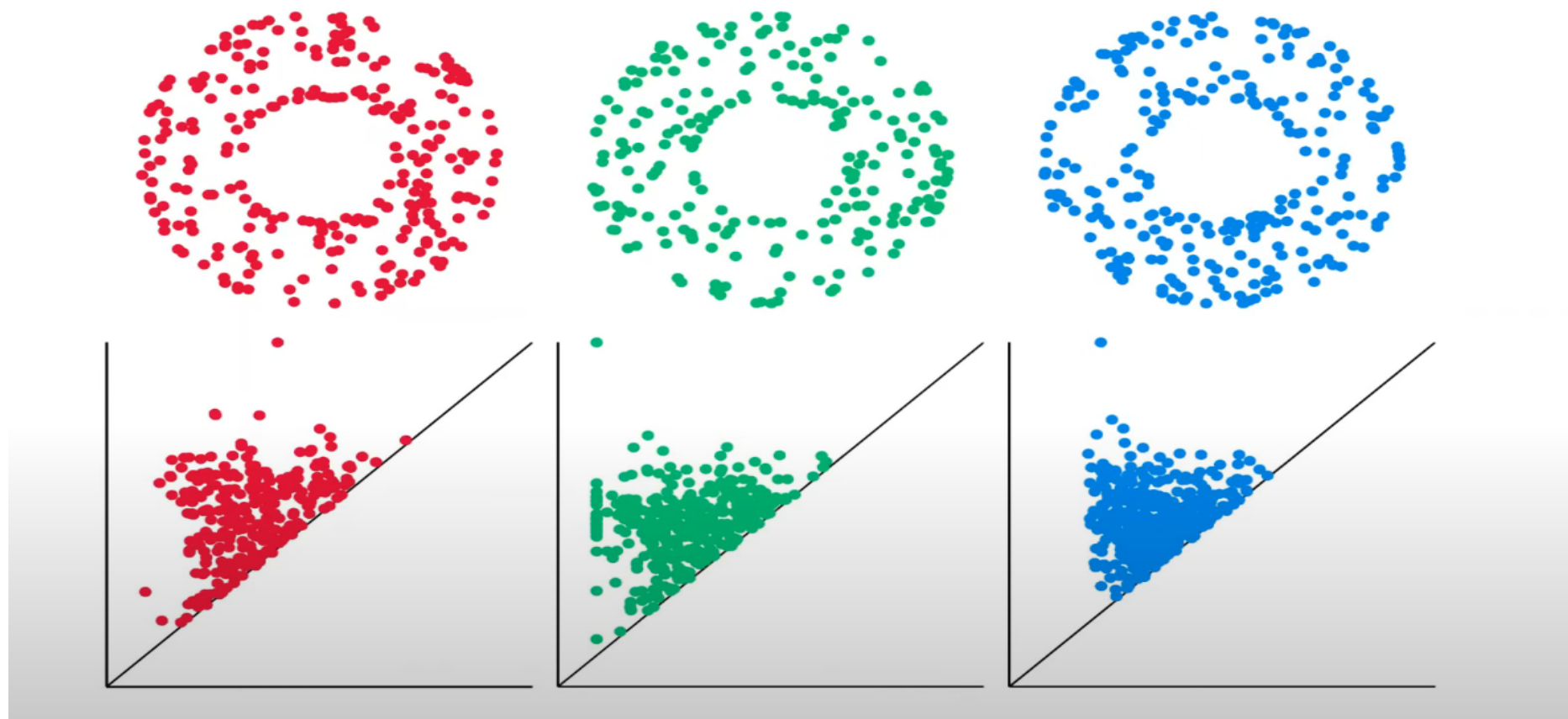
Q: Prove it.

Stability properties for point clouds

[Persistence stability for geometric complexes, Chazal, de Silva, Oudot, Geom. Dedicata, 2013].

Thm: If X and Y are pre-compact metric spaces, then

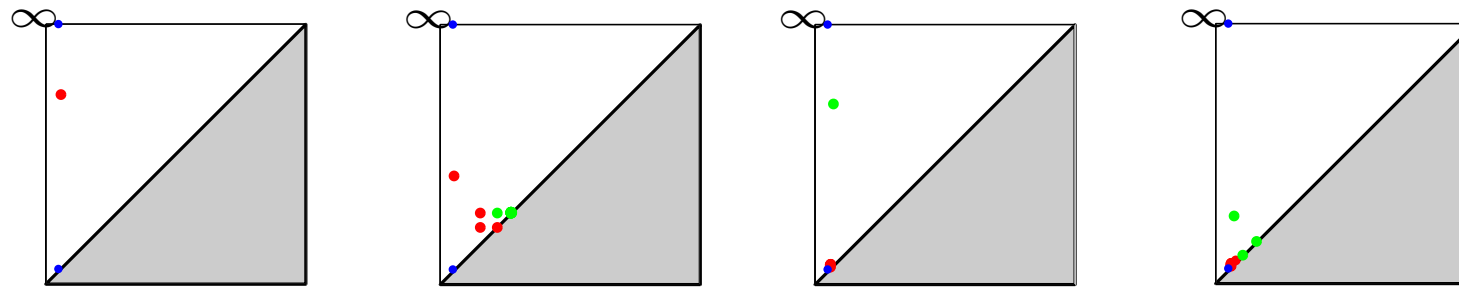
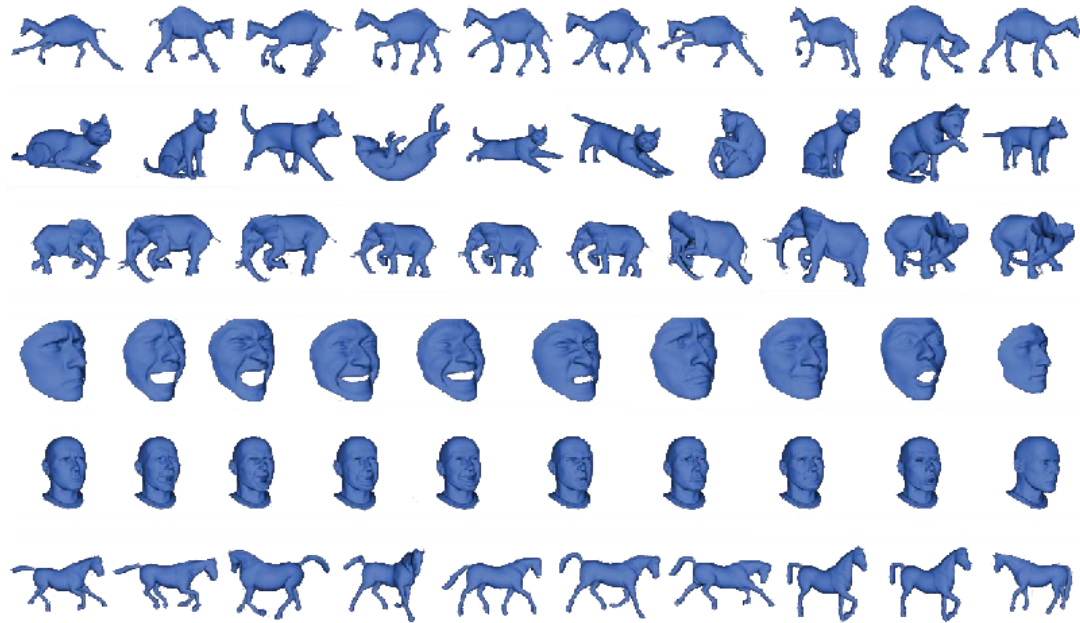
$$d_b(D_{\text{Rips}}(X), D_{\text{Rips}}(Y)) \leq d_{GH}(X, Y).$$



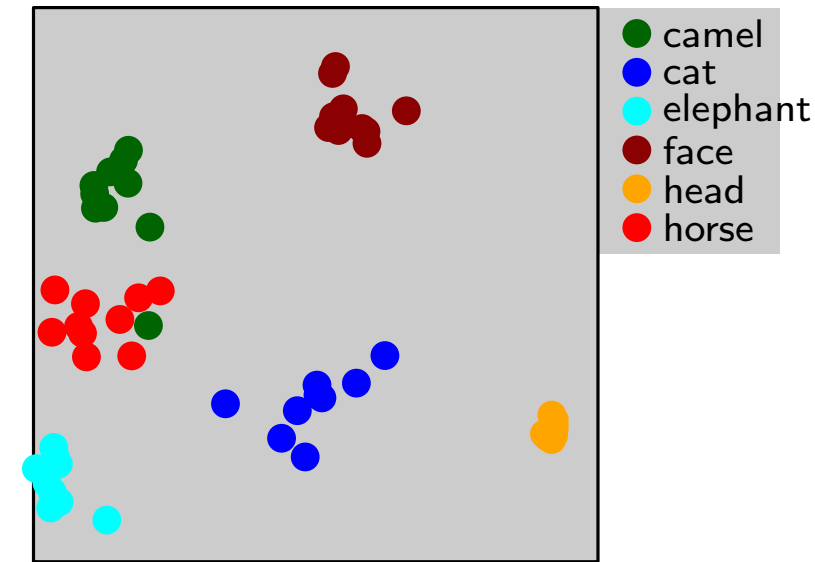
Rem: This result also holds for Čech and other families of filtrations (particular case of a more general theorem).

Application: non rigid shape classification

[*Gromov-Hausdorff Stable Signatures for Shapes using Persistence*, Chazal et al., Symp. Geom. Process., 2009]



MDS using bottleneck distance.



Non rigid shapes in a same class are almost isometric, but computing Gromov-Hausdorff distance between shapes is extremely expensive, so one can compare persistence diagrams of sampled shapes instead of shapes themselves.

Limitations

Thm: If X and Y are pre-compact metric spaces, then

$$d_b(D_{\text{Rips}}(X), D_{\text{Rips}}(Y)) \leq d_{GH}(X, Y).$$

→ Vietoris-Rips (or Čech, witness) filtrations become quickly prohibitively large as the size of the data increases: $O(2^{|X|})$, making the practical computation of persistence almost impossible.

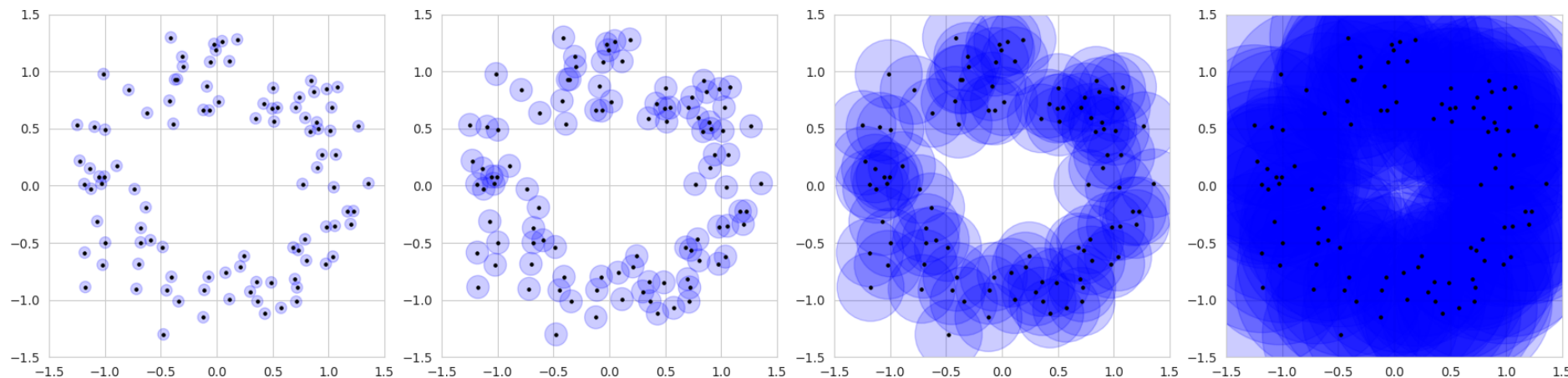
Limitations

Thm: If X and Y are pre-compact metric spaces, then

$$d_b(D_{\text{Rips}}(X), D_{\text{Rips}}(Y)) \leq d_{GH}(X, Y).$$

→ Vietoris-Rips (or Čech, witness) filtrations become quickly prohibitively large as the size of the data increases: $O(2^{|X|})$, making the practical computation of persistence almost impossible.

→ Persistence diagrams of Vietoris-Rips (as well as Čech, witness,..) filtrations and Gromov-Hausdorff distance are very sensitive to noise and outliers.



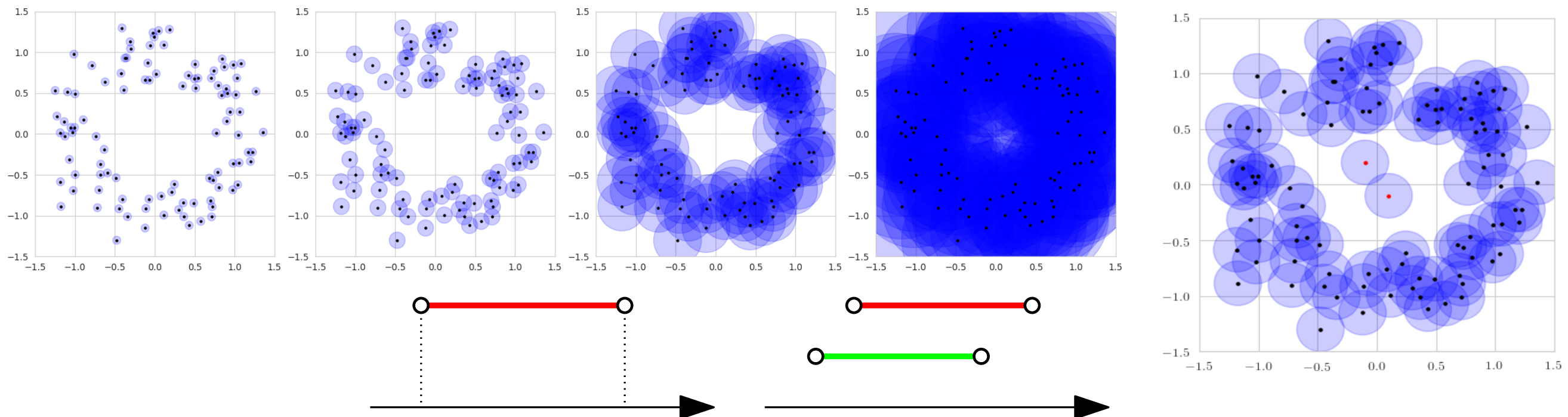
Limitations

Thm: If X and Y are pre-compact metric spaces, then

$$d_b(D_{\text{Rips}}(X), D_{\text{Rips}}(Y)) \leq d_{GH}(X, Y).$$

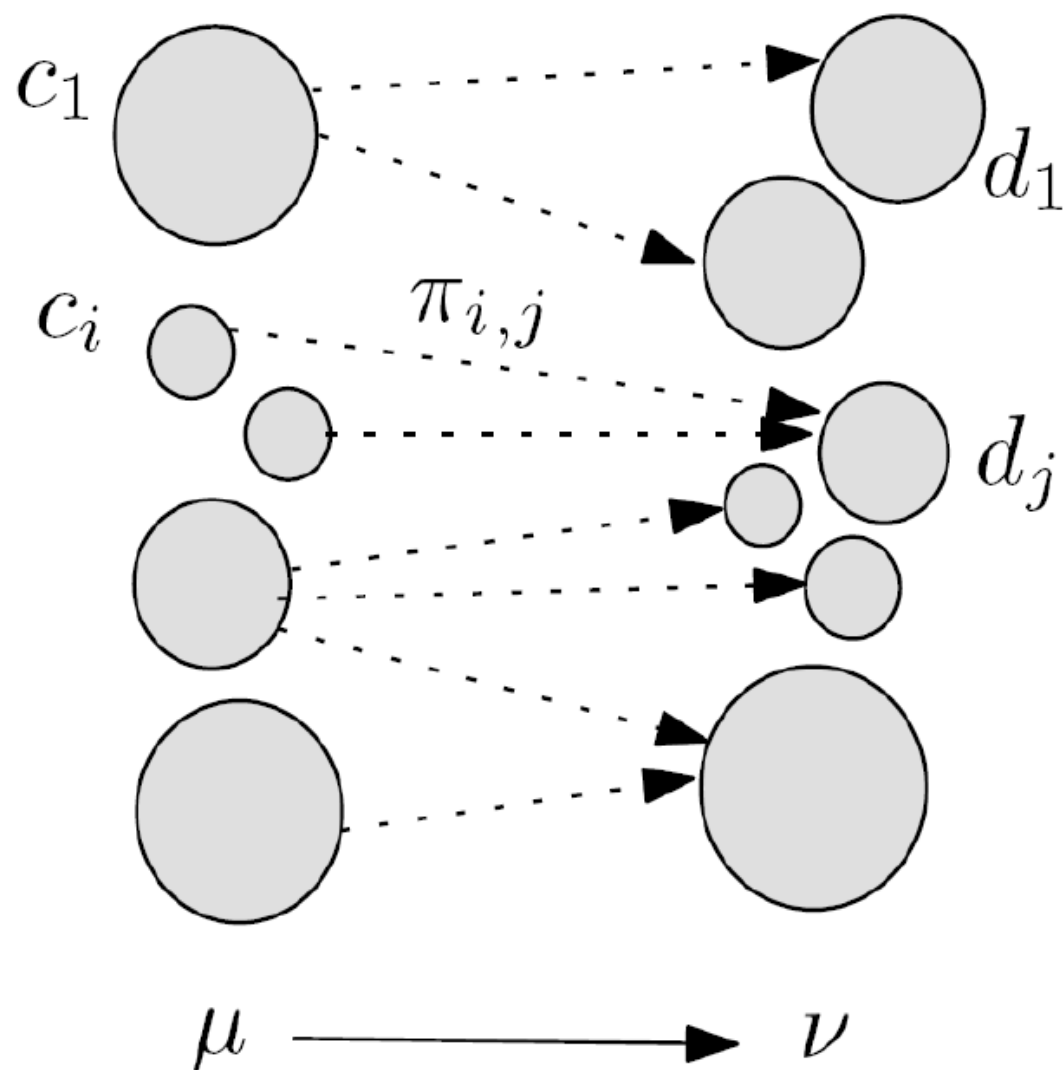
→ Vietoris-Rips (or Čech, witness) filtrations become quickly prohibitively large as the size of the data increases: $O(2^{|X|})$, making the practical computation of persistence almost impossible.

→ Persistence diagrams of Vietoris-Rips (as well as Čech, witness,..) filtrations and Gromov-Hausdorff distance are very sensitive to noise and outliers.



The Wasserstein distance

Let (X, d) be a metric space and let μ, ν be probability measures on X with finite p -moments ($p \geq 1$). The Wasserstein distance $W_p(\mu, \nu)$ quantifies the optimal cost of pushing μ onto ν , the cost of moving a small mass dx from x to y being $d(x, y)^p dx$.



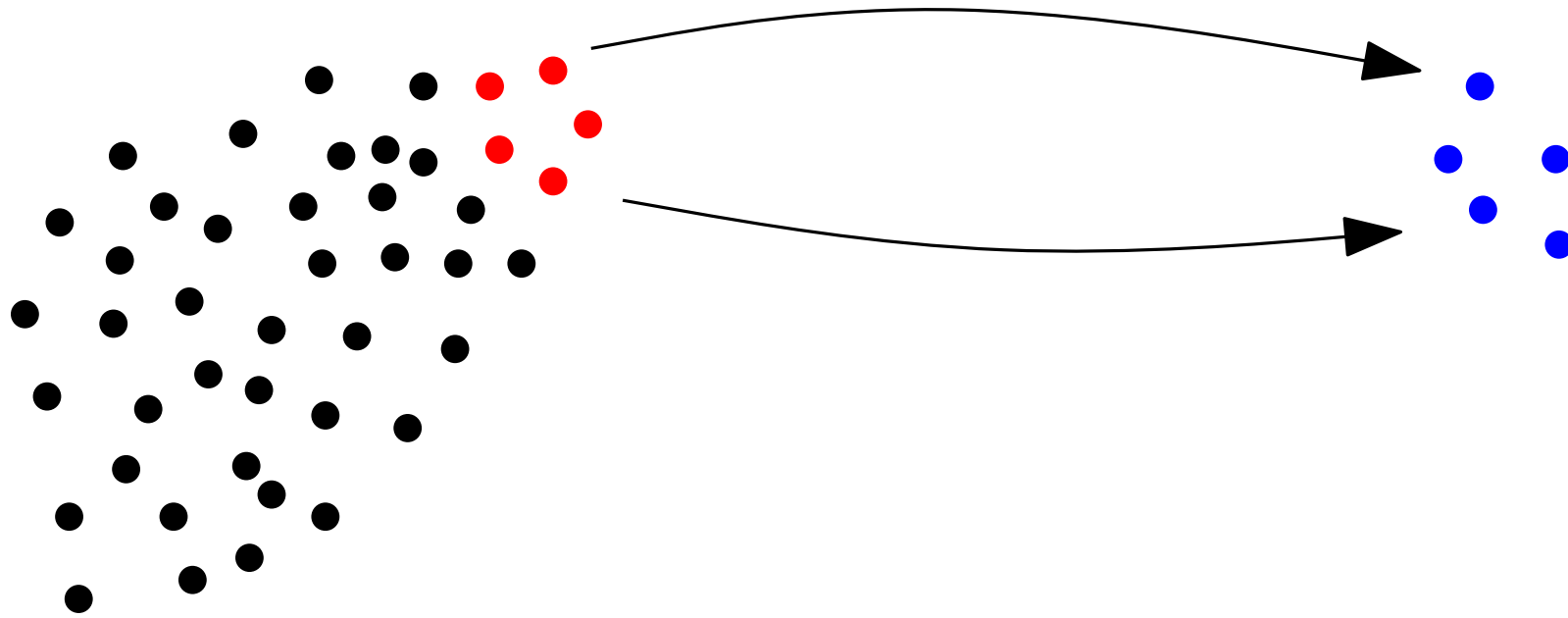
- Transport plan: Π a probability measure on $X \times X$ s.t. $\Pi(A \times \mathbb{R}^d) = \mu(A)$ and $\Pi(\mathbb{R}^d \times B) = \nu(B)$ for any borelian sets $A, B \subseteq X$.

- Cost of a transport plan:

$$C(\Pi) = \left(\int_{X \times X} d(x, y)^p d\Pi(x, y) \right)^{\frac{1}{p}}$$

- $W_p(\mu, \nu) = \inf_{\Pi} C(\Pi)$.

The Wasserstein distance



Ex: If $P = \{p_1, \dots, p_n\}$ is a point cloud, and $P' = \{p_1, \dots, p_{n-k-1}, o_1, \dots, o_k\}$ with $d(o_i, P) = R$, then

$$d_H(P, P') \geq R \quad \text{but} \quad W_2(\mu_P, \mu_{P'}) \leq \sqrt{\frac{k}{n}}(R + \text{diam}(P))$$

The Distance To Measure (DTM)

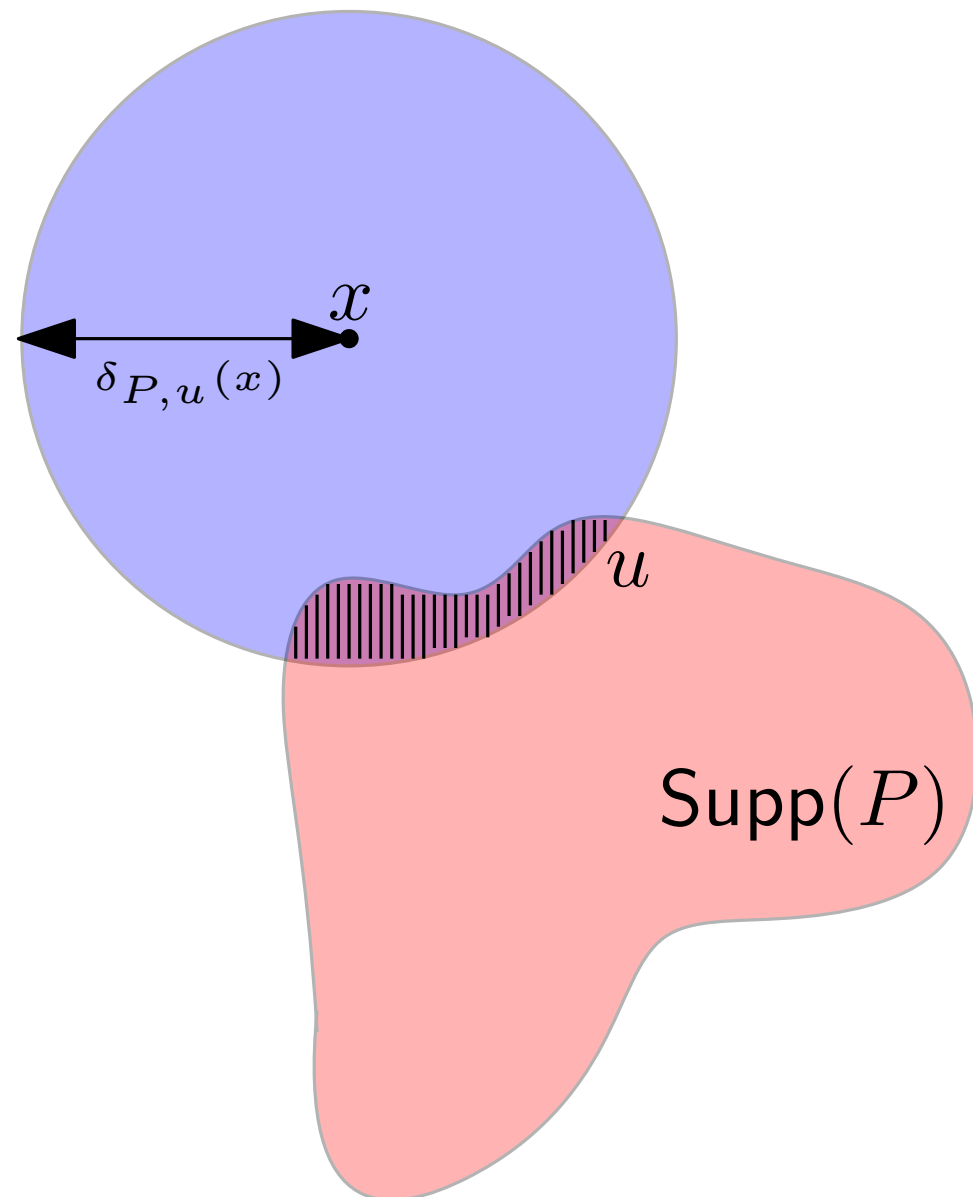
[*Geometric inference for probability measures*, Chazal, Cohen-Steiner, Mérigot, Found. Comput. Math., 2011]

The Distance To Measure (DTM)

[*Geometric inference for probability measures*, Chazal, Cohen-Steiner, Mérigot, Found. Comput. Math., 2011]

Preliminary distance function to a measure P : let $u \in]0, 1[$ be a positive mass, and P a probability measure on \mathbb{R}^d :

$$\delta_{P,u}(x) = \inf\{r > 0 : P(B(x, r)) \geq u\}$$

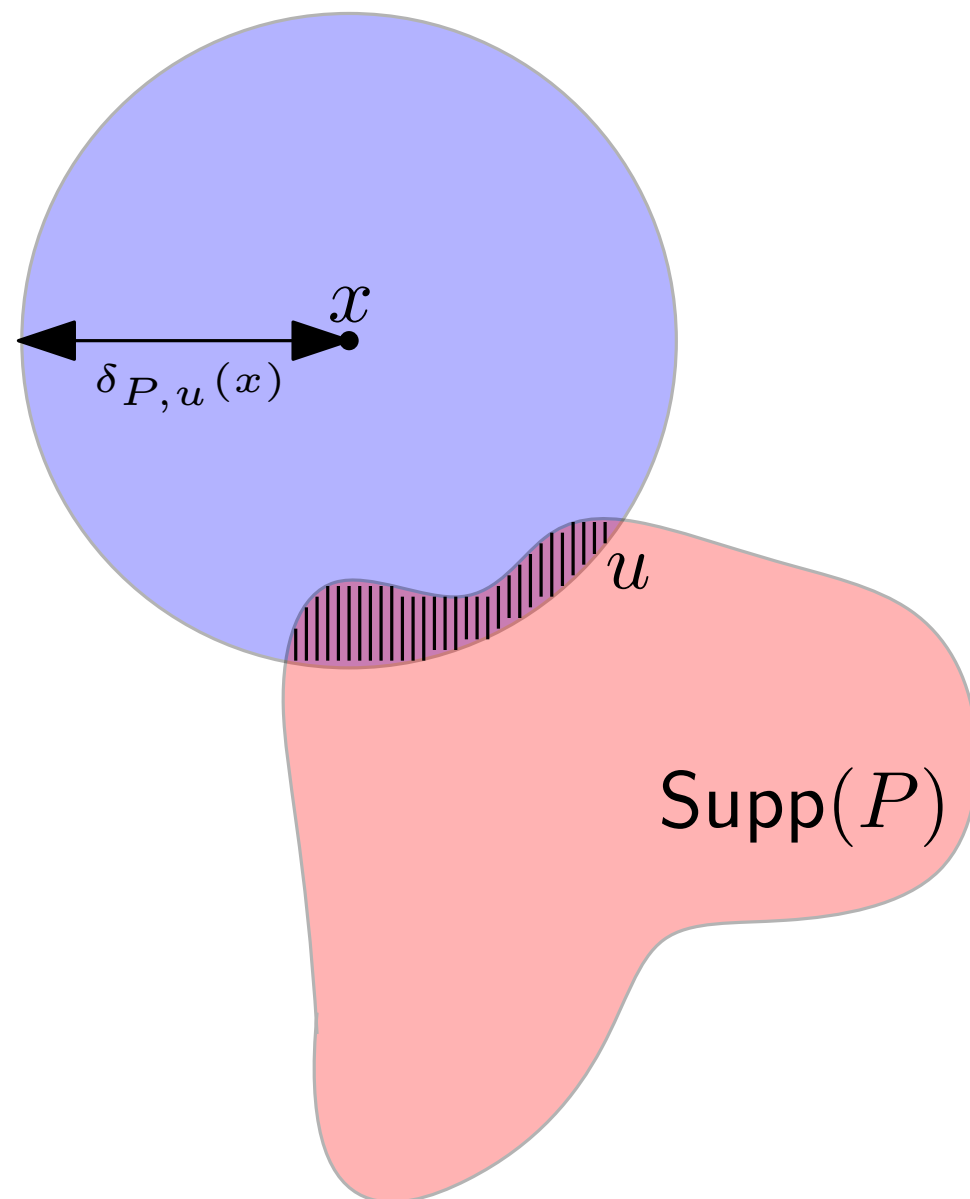


The Distance To Measure (DTM)

[*Geometric inference for probability measures*, Chazal, Cohen-Steiner, Mérigot, Found. Comput. Math., 2011]

Preliminary distance function to a measure P : let $u \in]0, 1[$ be a positive mass, and P a probability measure on \mathbb{R}^d :

$$\delta_{P,u}(x) = \inf\{r > 0 : P(B(x, r)) \geq u\}$$



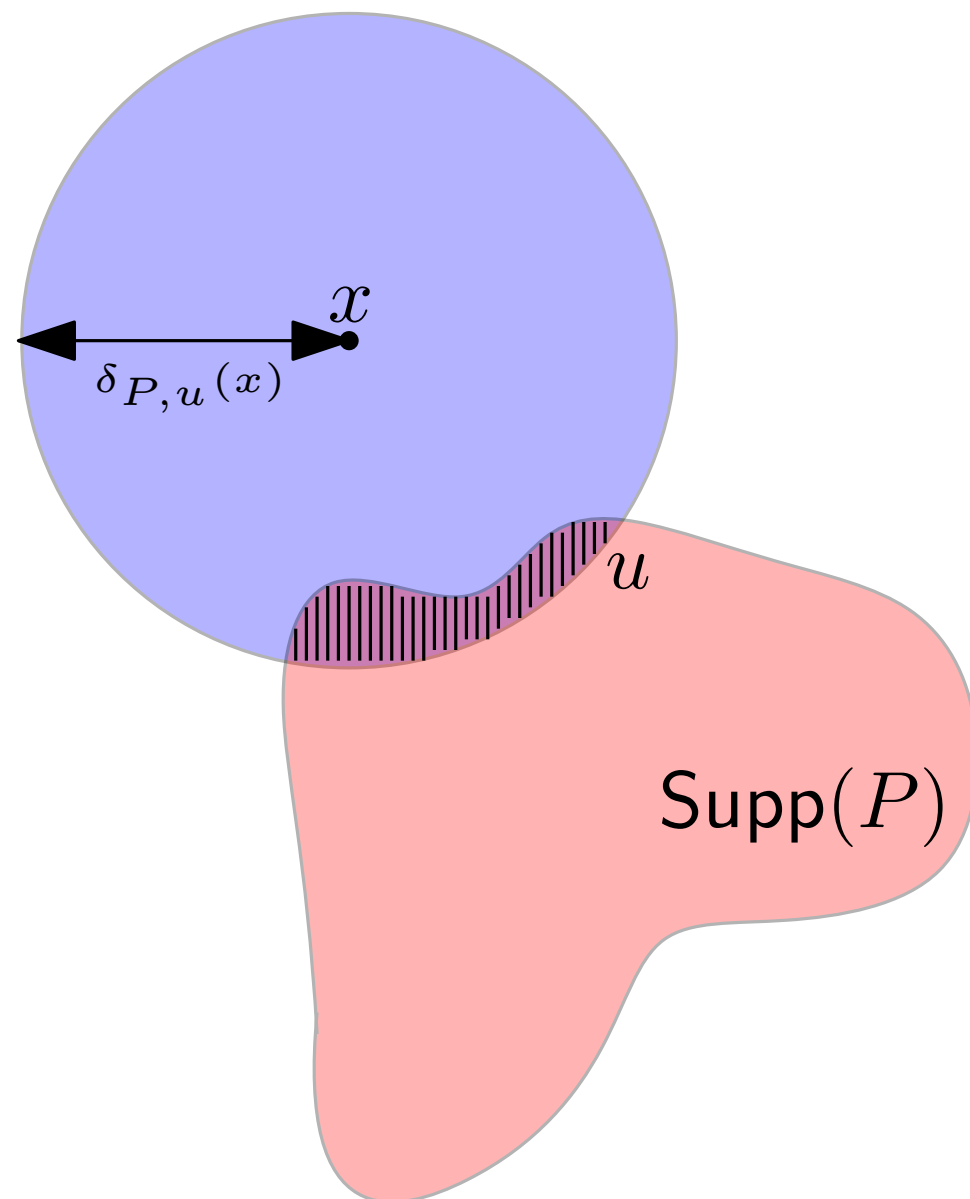
$\delta_{P,u}$ is the smallest distance needed to capture a mass of at least u .

The Distance To Measure (DTM)

[Geometric inference for probability measures, Chazal, Cohen-Steiner, Mérigot, Found. Comput. Math., 2011]

Preliminary distance function to a measure P : let $u \in]0, 1[$ be a positive mass, and P a probability measure on \mathbb{R}^d :

$$\delta_{P,u}(x) = \inf\{r > 0 : P(B(x, r)) \geq u\}$$



$\delta_{P,u}$ is the smallest distance needed to capture a mass of at least u .

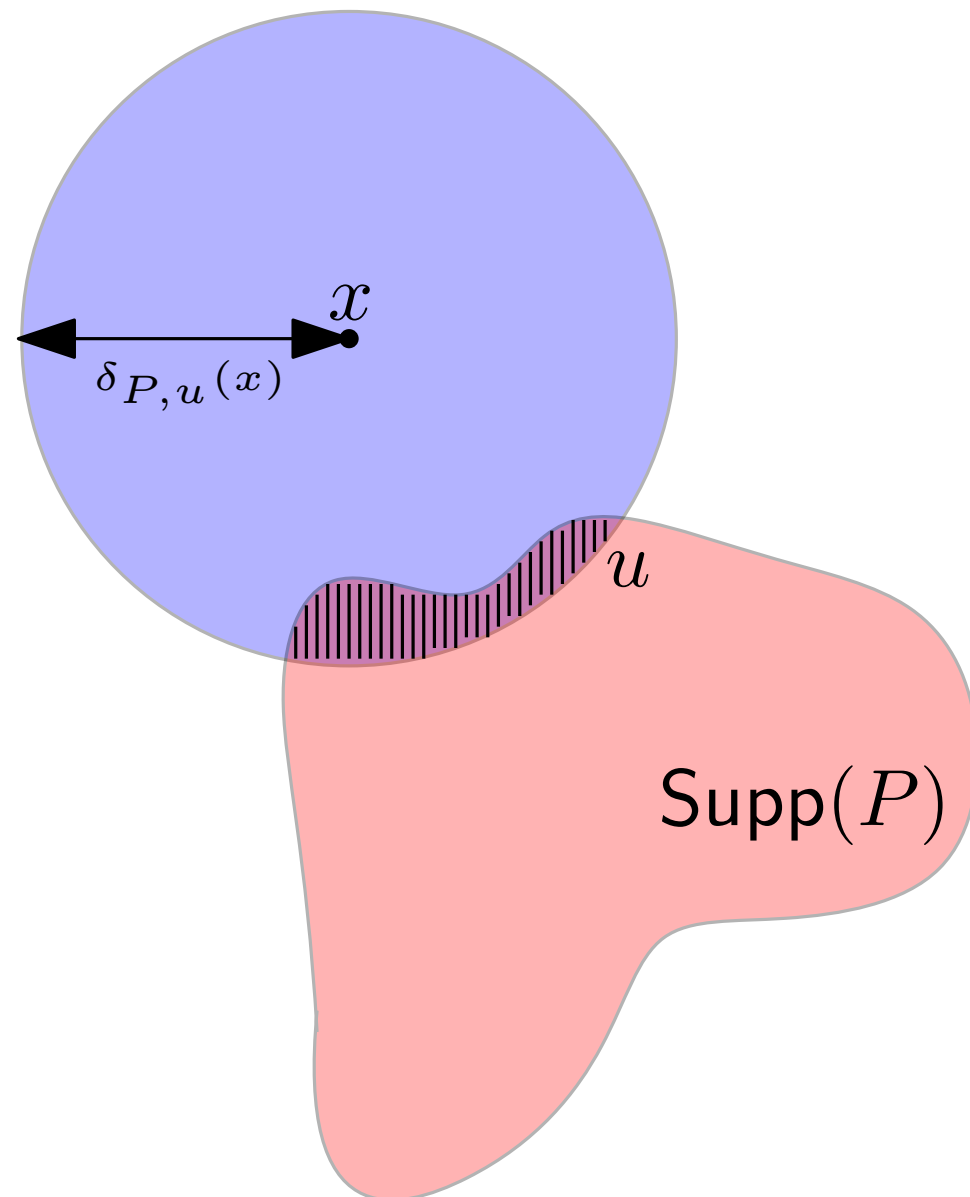
$\delta_{P,u}$ is the quantile function at u of the r.v. $\|x - X\|$ where $X \sim P$.

The Distance To Measure (DTM)

[Geometric inference for probability measures, Chazal, Cohen-Steiner, Mérigot, Found. Comput. Math., 2011]

Preliminary distance function to a measure P : let $u \in]0, 1[$ be a positive mass, and P a probability measure on \mathbb{R}^d :

$$\delta_{P,u}(x) = \inf\{r > 0 : P(B(x, r)) \geq u\}$$



Def: Given a probability measure P on \mathbb{R}^d and $m > 0$, the distance function to the measure P (DTM) is defined by

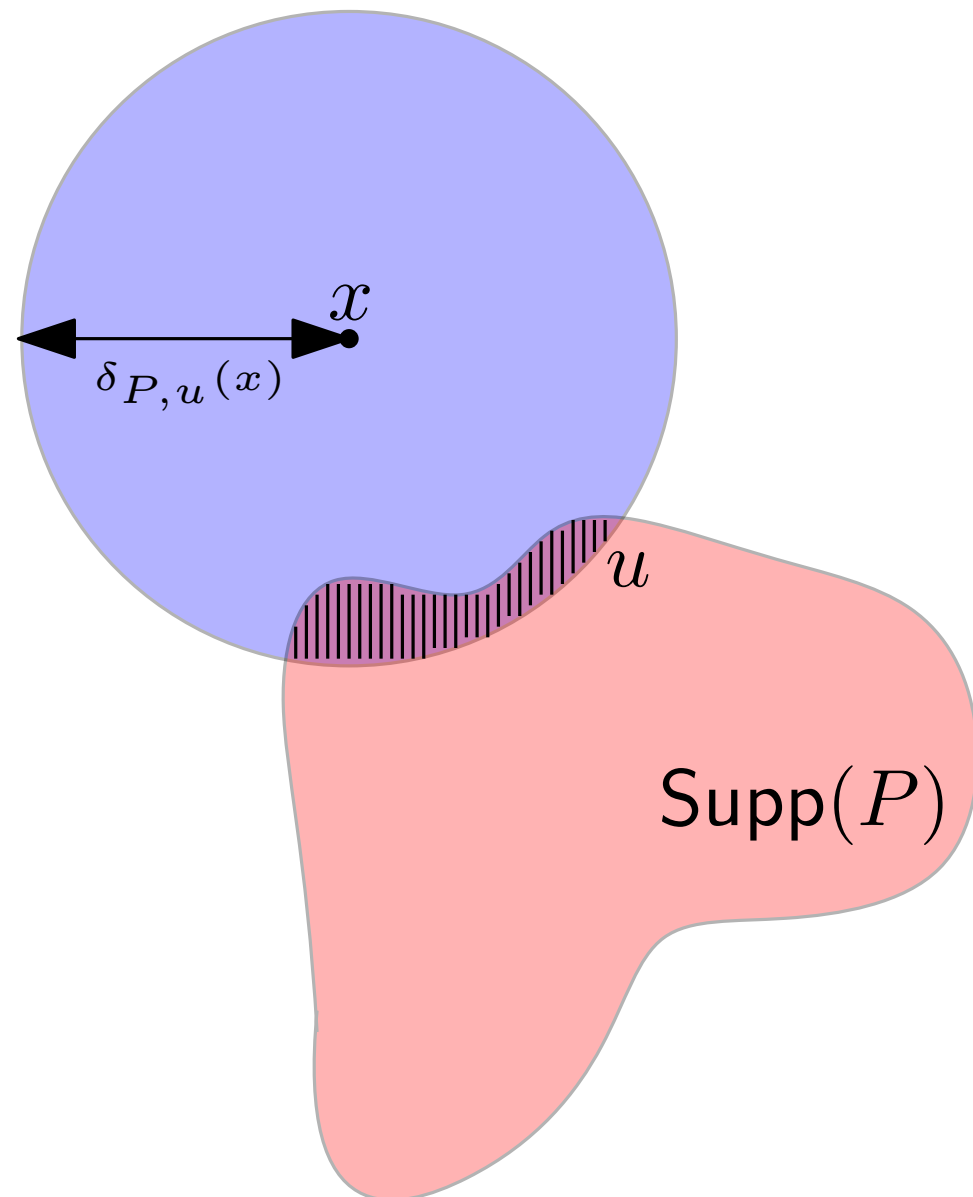
$$d_{P,m} : x \in \mathbb{R} \mapsto \left(\frac{1}{m} \int_0^m \delta_{P,u}^2(x) du \right)^{1/2}$$

The Distance To Measure (DTM)

[Geometric inference for probability measures, Chazal, Cohen-Steiner, Mérigot, Found. Comput. Math., 2011]

Preliminary distance function to a measure P : let $u \in]0, 1[$ be a positive mass, and P a probability measure on \mathbb{R}^d :

$$\delta_{P,u}(x) = \inf\{r > 0 : P(B(x, r)) \geq u\}$$



Def: Given a probability measure P on \mathbb{R}^d and $m > 0$, the distance function to the measure P (DTM) is defined by

$$d_{P,m} : x \in \mathbb{R} \mapsto \left(\frac{1}{m} \int_0^m \delta_{P,u}^2(x) du \right)^{1/2}$$

The DTM is robust, i.e., stable under Wasserstein perturbations:

$$\|d_{P,m} - d_{Q,m}\|_{\infty} \leq \frac{1}{\sqrt{m}} W_2(P, Q)$$

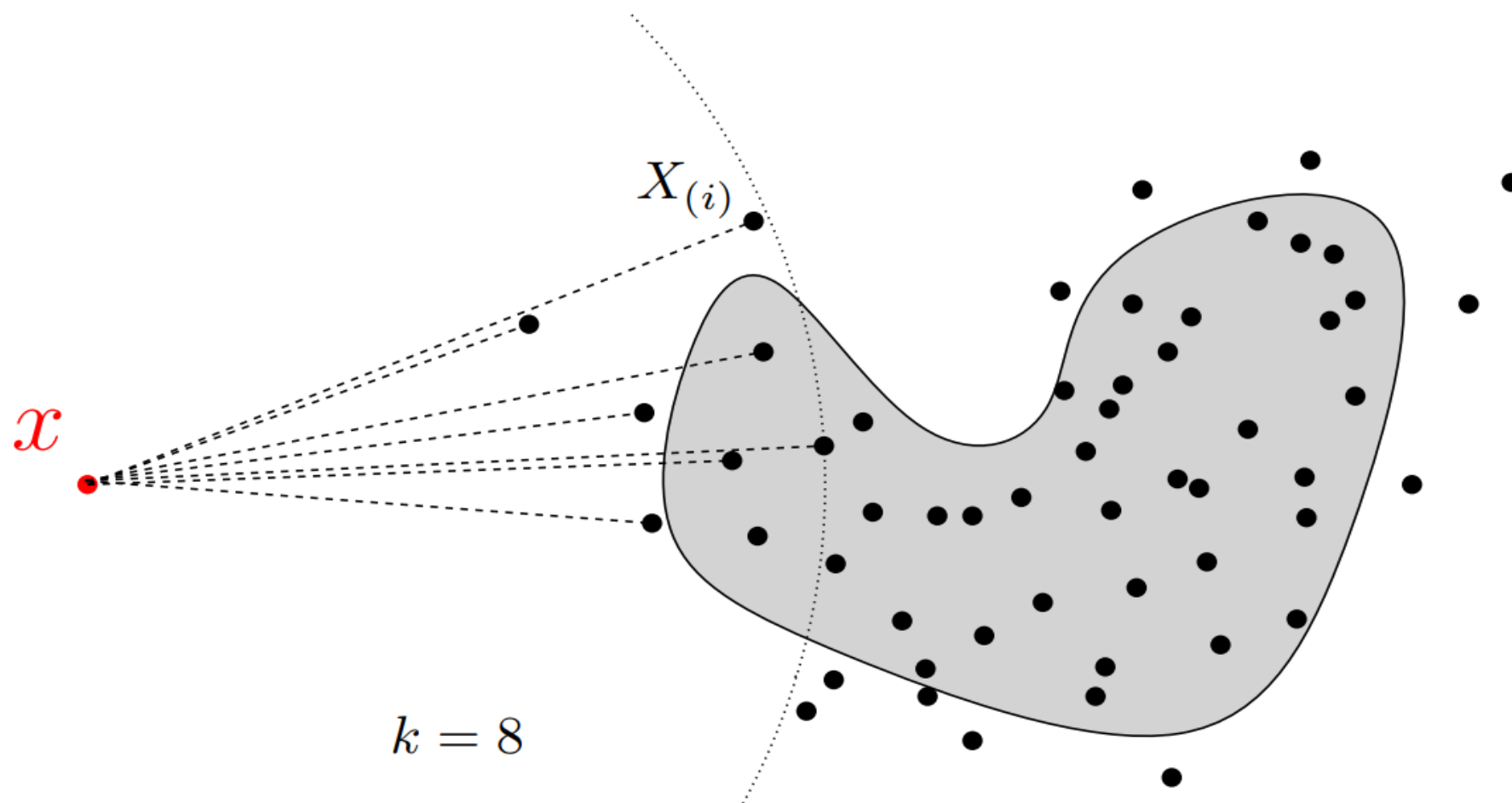
The Distance To Measure (DTM)

[Geometric inference for probability measures, Chazal, Cohen-Steiner, Mérigot, Found. Comput. Math., 2011]

Def: Let X_1, \dots, X_n sampled according to P and let P_n be the empirical measure. Then

$$d_{P_n, k/n}(x) = \frac{1}{k} \sum_{i=1}^k \|x - X_{(i)}\|^2,$$

where $\|X_{(1)} - x\| \leq \|X_{(2)} - x\| \leq \dots \leq \|X_{(k)} - x\| \leq \dots \leq \|X_{(n)} - x\|$.



DTM-based filtrations

[DTM-based filtrations, Anai et al.,
Symp. Comp. Geom., 2019]

Def: Let V be a point cloud (in a metric space). The **DTM-based complex** $W(V)$ is the filtered simplicial complex indexed by \mathbb{R} whose vertex set is V and whose other simplices are defined with

$$\sigma = [p_0, p_1, \dots, p_k] \in W(V, \alpha) \iff \bigcap_{i=0}^k B(p_i, r_{p_i}(\alpha)) \neq \emptyset$$

where $r_p(\alpha) = 0$ if $\alpha \leq d_{P_{n,k}/n}(p)$ and $|\alpha^q - d_{P_{n,k}/n}(p)^q|^{1/q}$ otherwise.

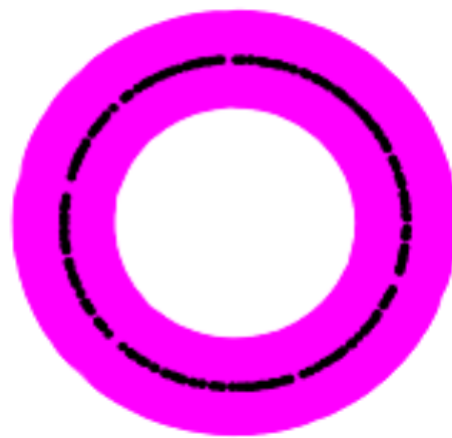
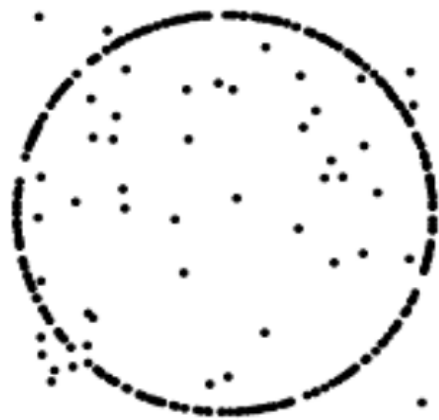
DTM-based filtrations

[DTM-based filtrations, Anai et al.,
Symp. Comp. Geom., 2019]

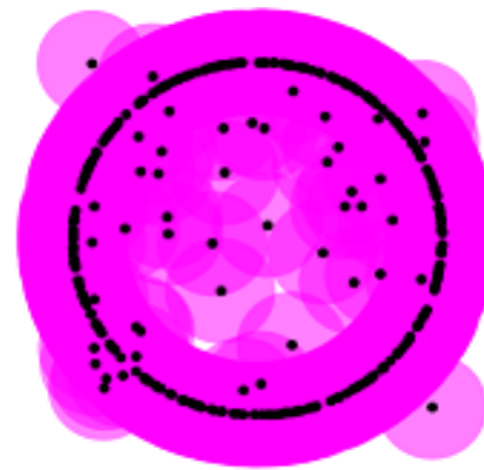
Def: Let V be a point cloud (in a metric space). The **DTM-based complex** $W(V)$ is the filtered simplicial complex indexed by \mathbb{R} whose vertex set is V and whose other simplices are defined with

$$\sigma = [p_0, p_1, \dots, p_k] \in W(V, \alpha) \iff \bigcap_{i=0}^k B(p_i, r_{p_i}(\alpha)) \neq \emptyset$$

where $r_p(\alpha) = 0$ if $\alpha \leq d_{P_{n,k}/n}(p)$ and $|\alpha^q - d_{P_{n,k}/n}(p)^q|^{1/q}$ otherwise.



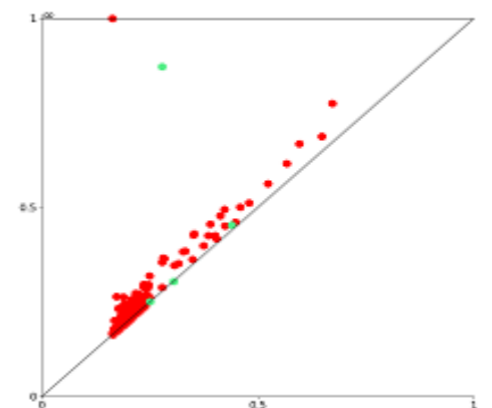
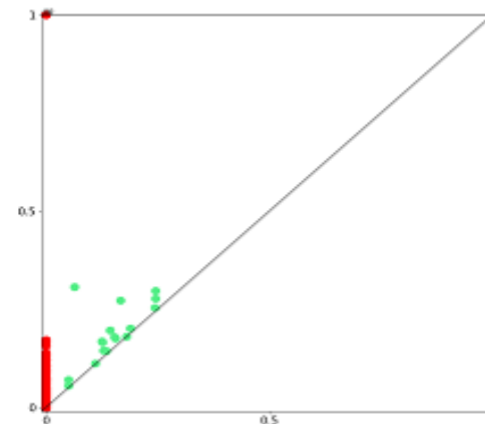
Rips



Rips



DTM-based



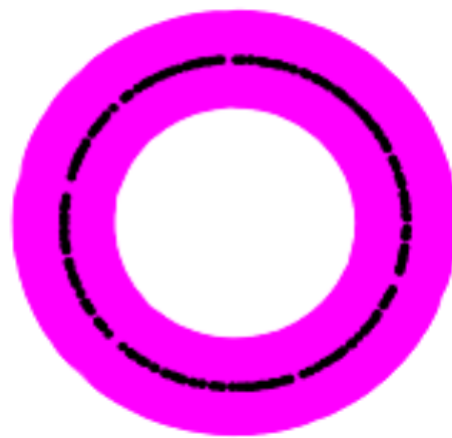
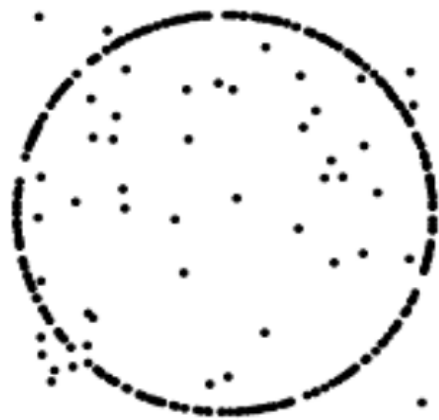
DTM-based filtrations

[DTM-based filtrations, Anai et al.,
Symp. Comp. Geom., 2019]

Def: Let V be a point cloud (in a metric space). The **DTM-based complex** $W(V)$ is the filtered simplicial complex indexed by \mathbb{R} whose vertex set is V and whose other simplices are defined with

$$\sigma = [p_0, p_1, \dots, p_k] \in W(V, \alpha) \iff \bigcap_{i=0}^k B(p_i, r_{p_i}(\alpha)) \neq \emptyset$$

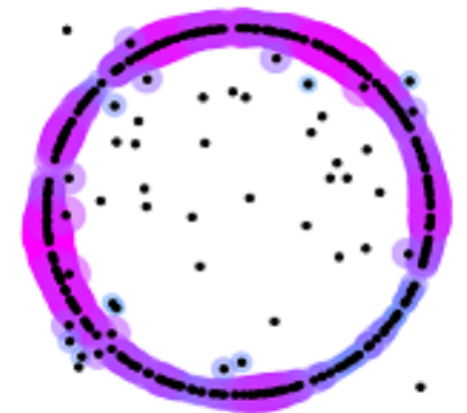
where $r_p(\alpha) = 0$ if $\alpha \leq d_{P_{n,k}/n}(p)$ and $|\alpha^q - d_{P_{n,k}/n}(p)^q|^{1/q}$ otherwise.



Rips

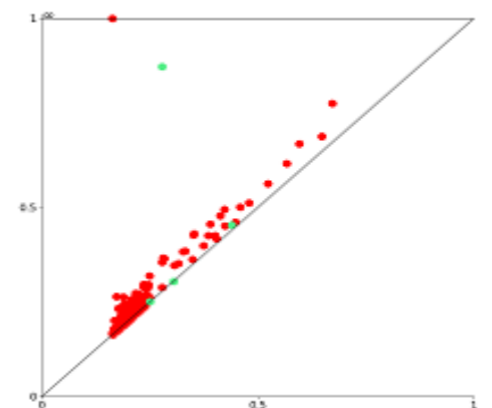
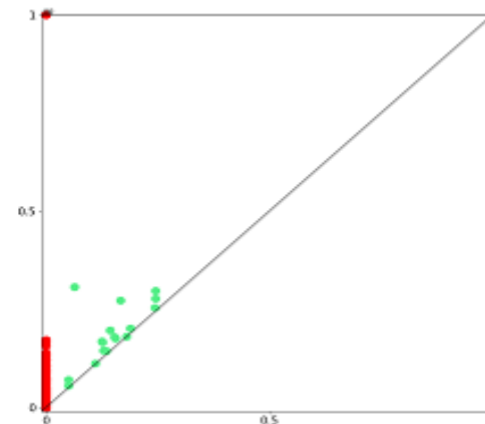


Rips



DTM-based

Thm: One has: $d_b(D_W(X), D_W(Y)) \leq \sqrt{\frac{n}{k}}(W_2(\Omega, X) + W_2(\Omega, Y)) + 2C_{\Omega, n, k}$.



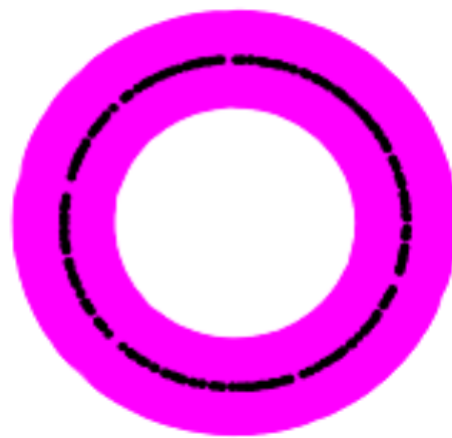
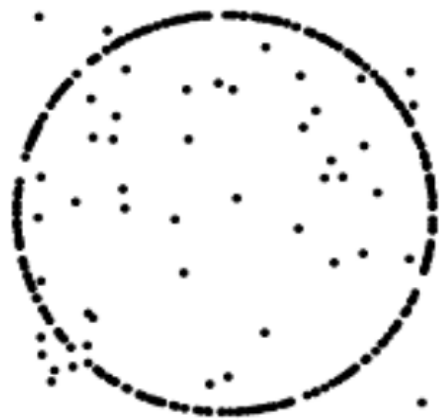
DTM-based filtrations

[DTM-based filtrations, Anai et al.,
Symp. Comp. Geom., 2019]

Def: Let V be a point cloud (in a metric space). The **DTM-based complex** $W(V)$ is the filtered simplicial complex indexed by \mathbb{R} whose vertex set is V and whose other simplices are defined with

$$\sigma = [p_0, p_1, \dots, p_k] \in W(V, \alpha) \iff \bigcap_{i=0}^k B(p_i, r_{p_i}(\alpha)) \neq \emptyset$$

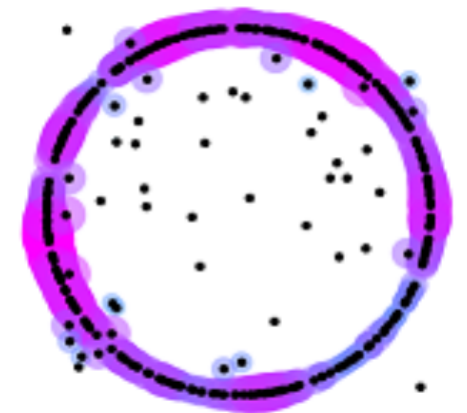
where $r_p(\alpha) = 0$ if $\alpha \leq d_{P_{n,k}/n}(p)$ and $|\alpha^q - d_{P_{n,k}/n}(p)^q|^{1/q}$ otherwise.



Rips



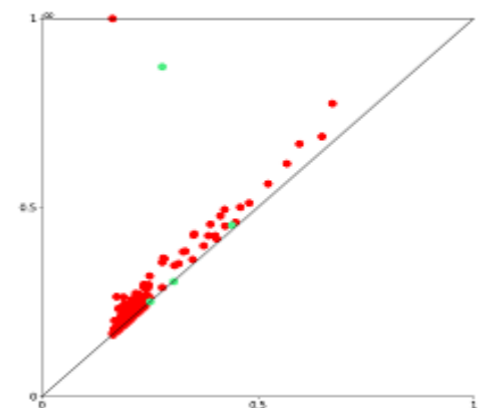
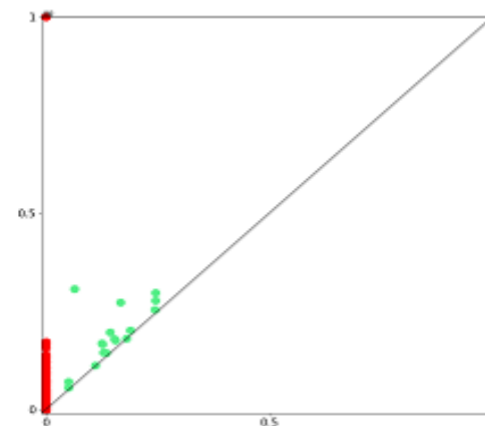
Rips



DTM-based

Thm: One has: $d_b(D_W(X), D_W(Y)) \leq \sqrt{\frac{n}{k}} (W_2(\Omega, X) + W_2(\Omega, Y)) + 2C_{\Omega, n, k}$.

”Clean subset” without outliers



Summary

In this class, I introduced the basic bricks of **persistent homology**.

We have seen how to compute the homology groups of simplicial complexes with **filtrations** and their **positive and negative simplices**.

We have seen that positive and negative simplices can be paired together to form **persistence barcodes/diagrams**.

We have seen that persistence barcodes/diagrams are **stable** with respect to the **bottleneck distance**.

Next time, we will study the **representations** and **statistical properties** of persistence diagrams, that allow to combine them with standard machine learning models in a robust way.