IFPU focus week
Intepretable and higher-order statistics
for late-time cosmology

Introduction to Topological Machine Learning

Mathieu Carrière
INRIA Sophia-Antipolis
mathieu.carriere@inria.fr

June 29th, 2022

https://github.com/GUDHI/TDA-tutorial

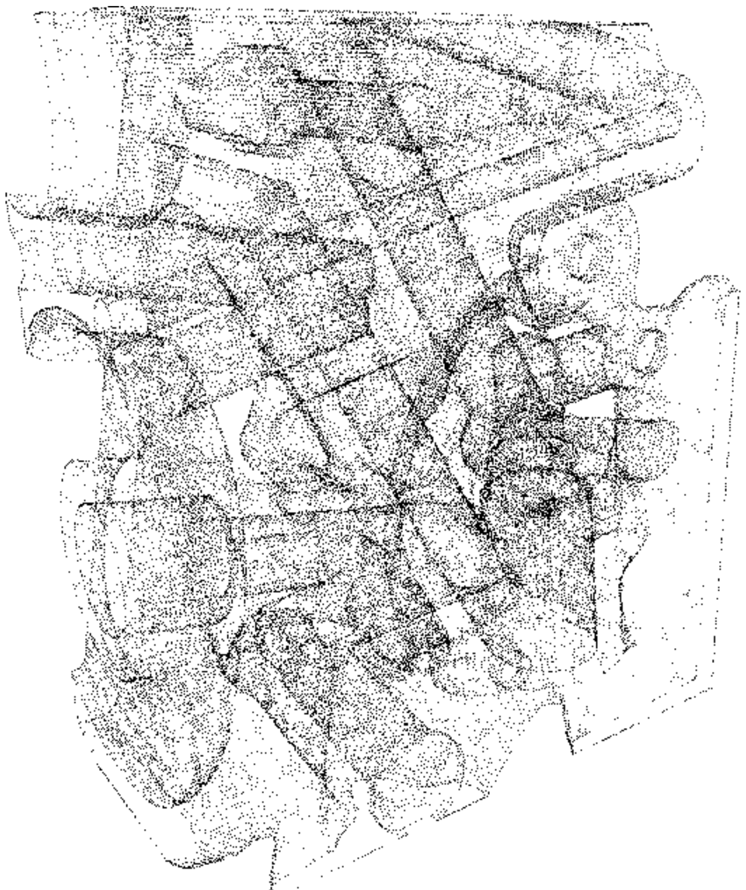
Topological Data Analysis

The purpose of Topological Data Analysis is to build topological features from data sets...

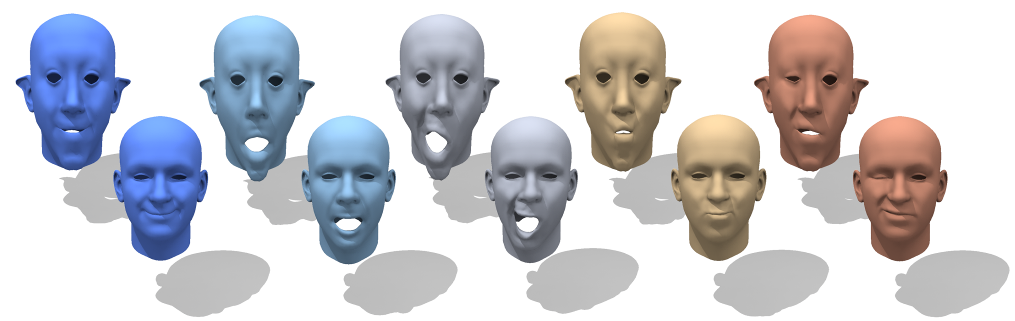
Topological Data Analysis

The purpose of Topological Data Analysis is to build topological features from data sets... but why is that interesting?

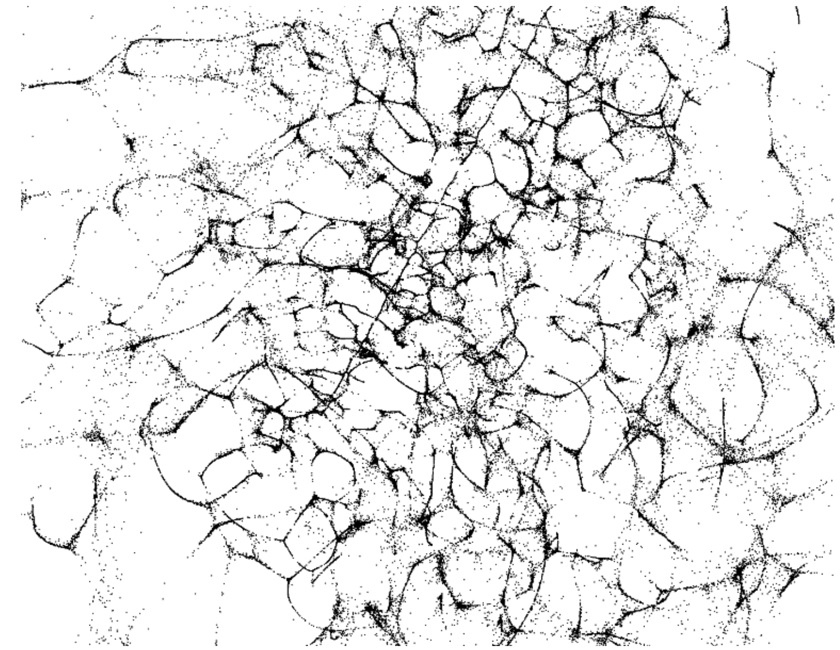
Topological Data Analysis



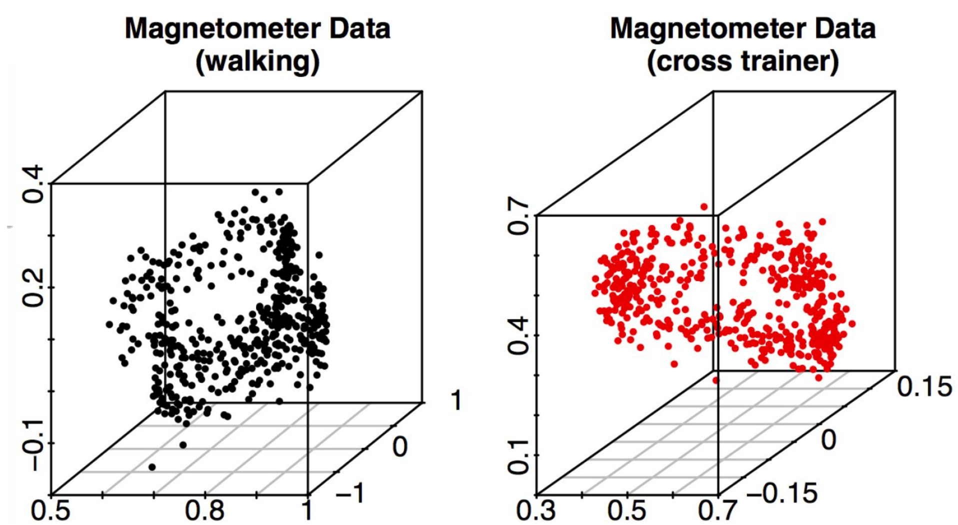
Scans



3D shapes

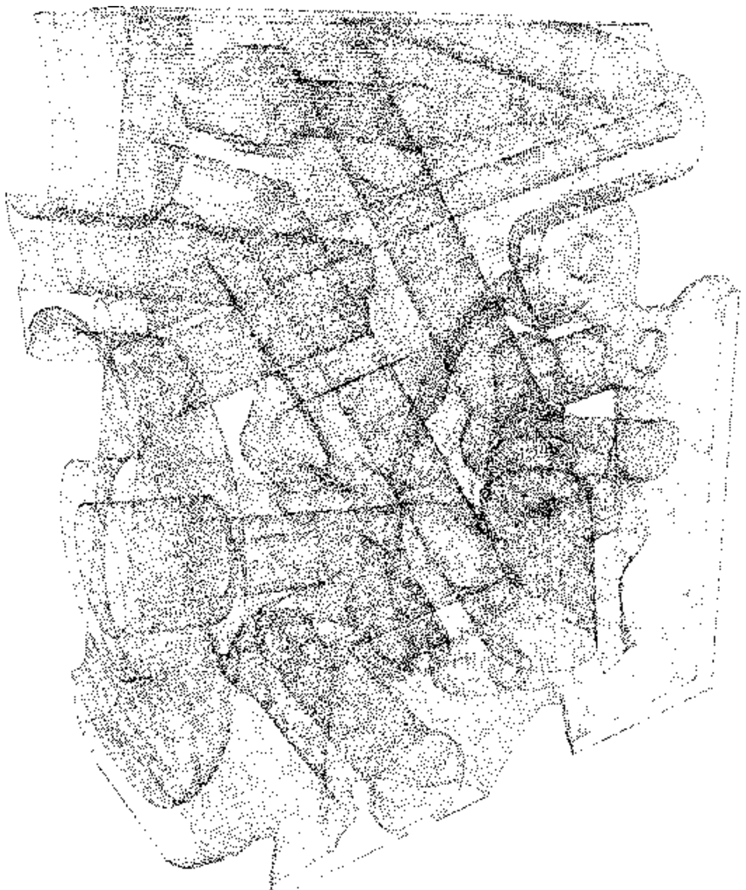


Galaxies

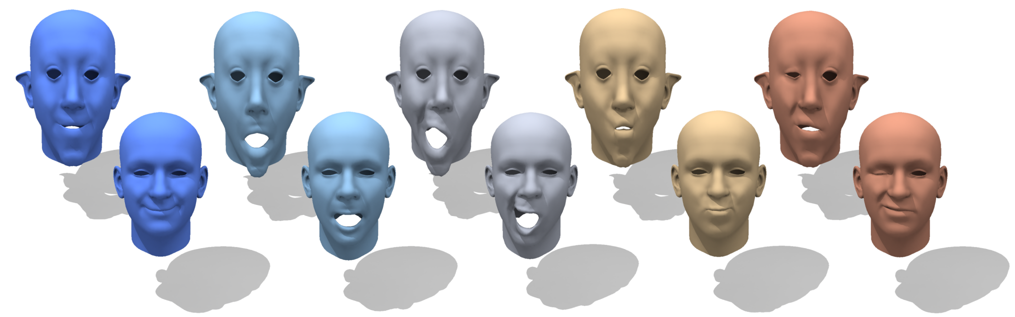


Magnetometer

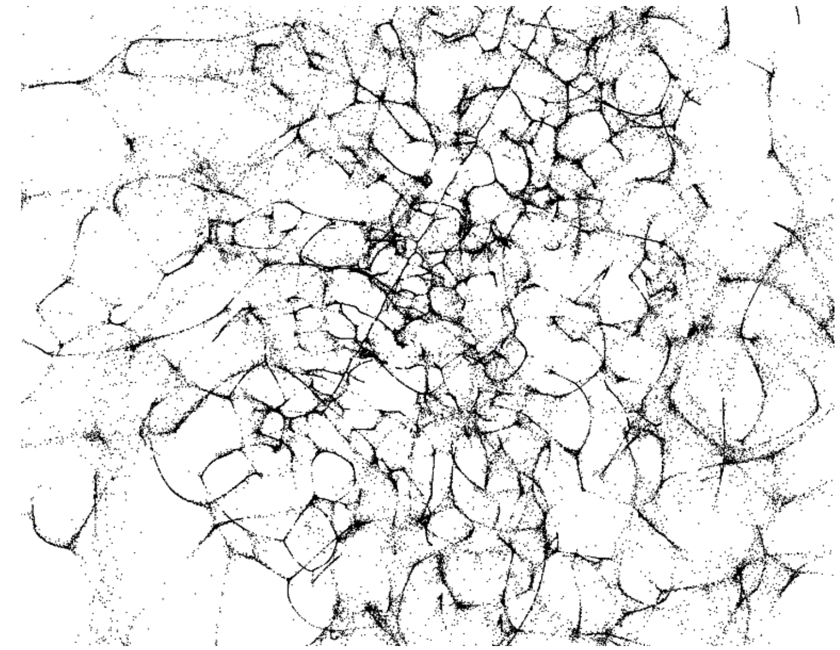
Topological Data Analysis



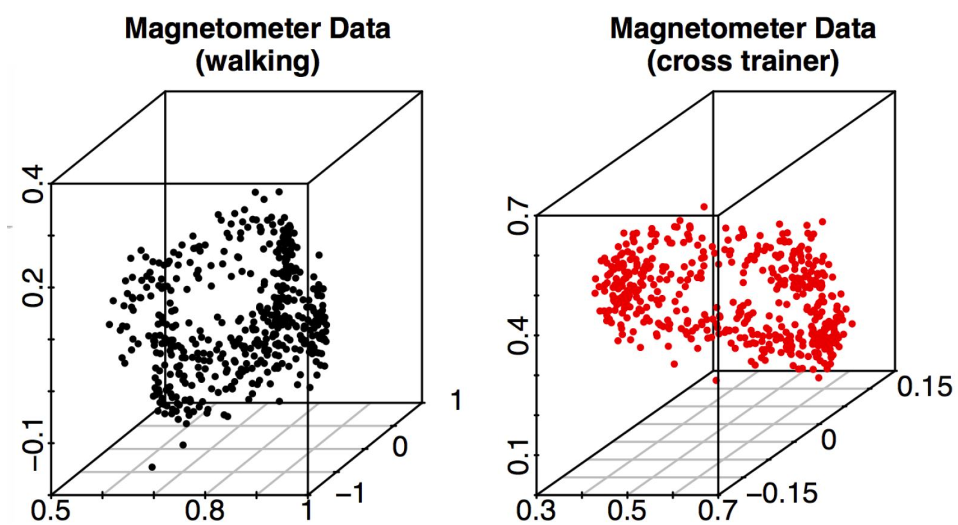
Scans



3D shapes



Galaxies



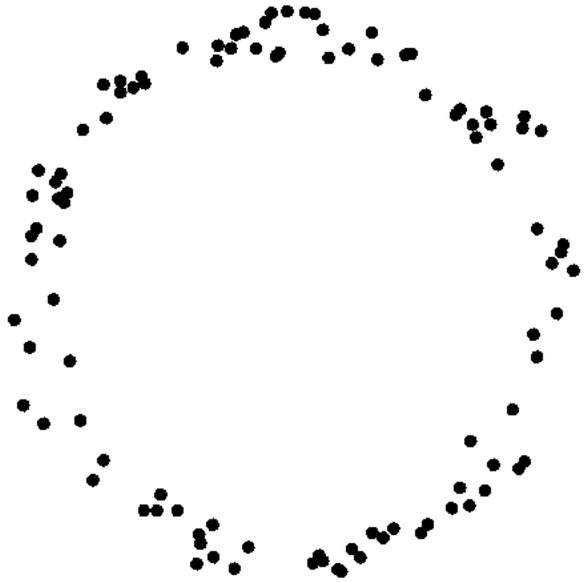
Magnetometer

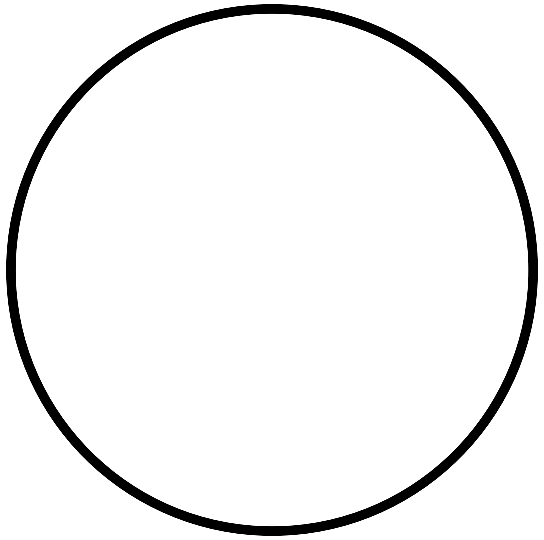
Data often come as (sampling of) metric spaces or sets/spaces endowed with a similarity measure with, possibly complex, topological/geometric structure.

Data carrying geometric information is usually high dimensional.

Topological Data Analysis

Problem: how to actually compute the topology, or *homology groups*, of a data set given as a finite point cloud?

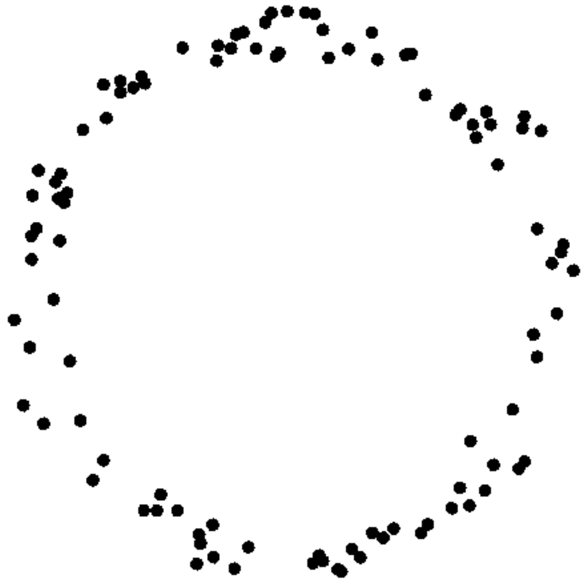


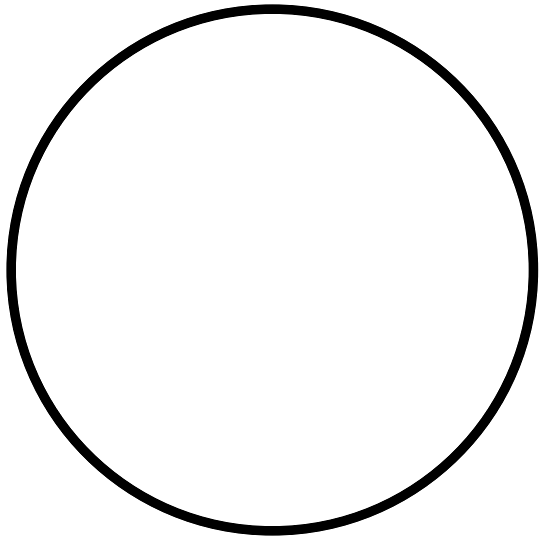


Topological Data Analysis

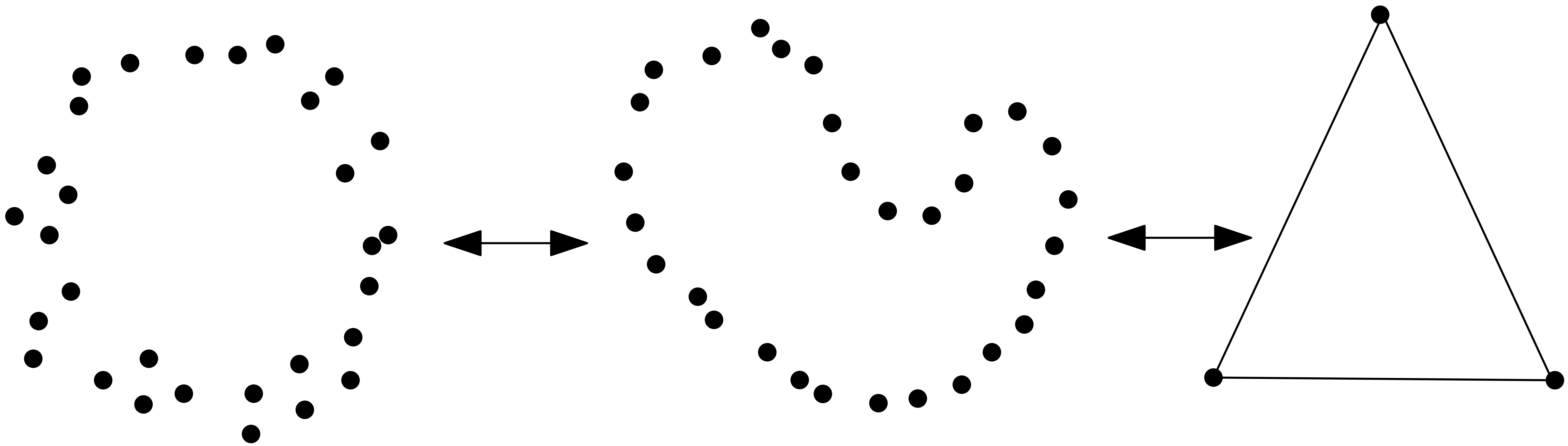
Problem: how to actually compute the topology, or *homology groups*, of a data set given as a finite point cloud?

A: See Magnus talk!





Topological Data Analysis



Advantages:

→ **coordinate invariance:** topological features/invariants do not rely on any coordinate system.

→ **deformation invariance:** topological features are invariant under homeomorphism and reparameterization.

→ **compressed representation:** topology offers a set of tools to summarize the data in compact ways while preserving its topological structure.

Topological Clustering with 0-dimensional persistence

Persistence diagrams, Kernels and Deep Learning

Persistence Diagrams and Statistics

Persistence Diagrams and Optimization

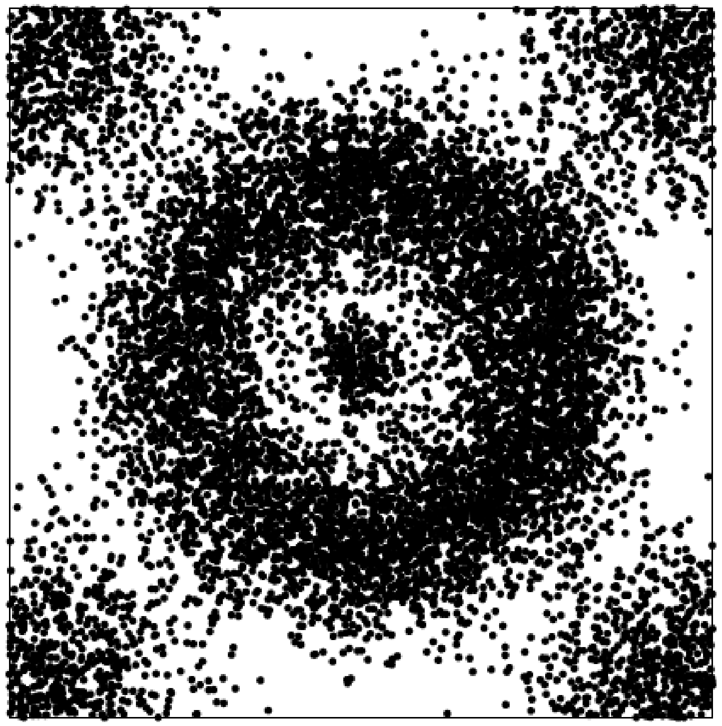
Persistence Approximation and Robustness

Persistence Diagram Embeddings into Hilbert Spaces

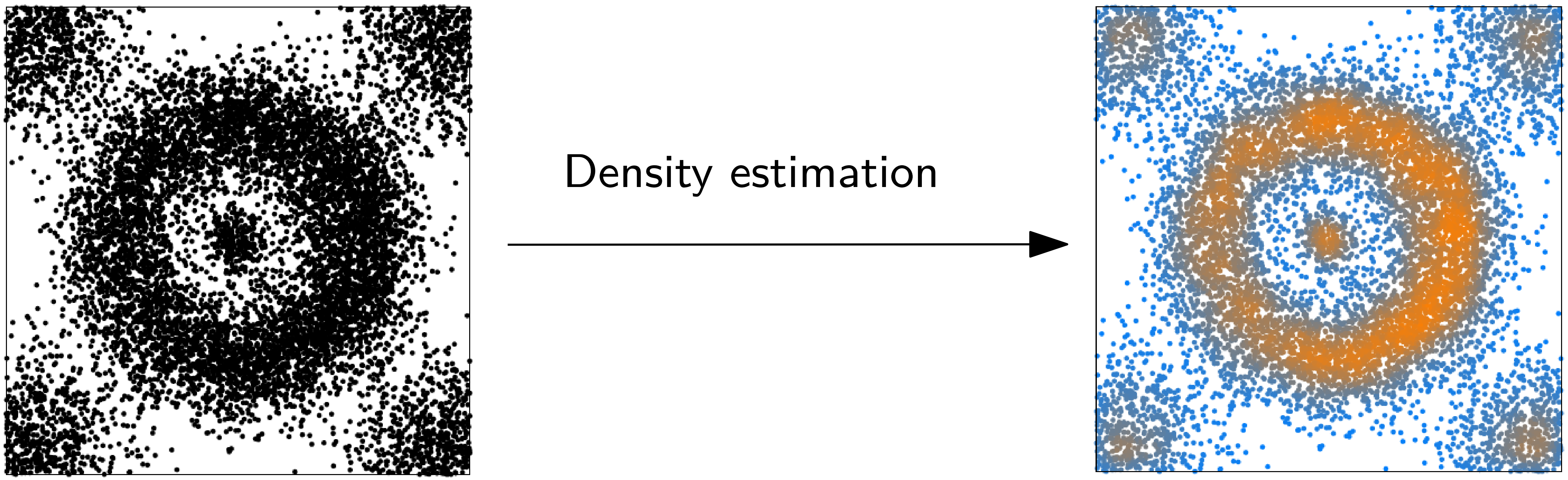
Topological Clustering with 0-dimensional persistence

[*Persistence-Based Clustering in Riemannian Manifolds*, Chazal, Oudot, Skraba, Guibas, J. ACM, 2013]

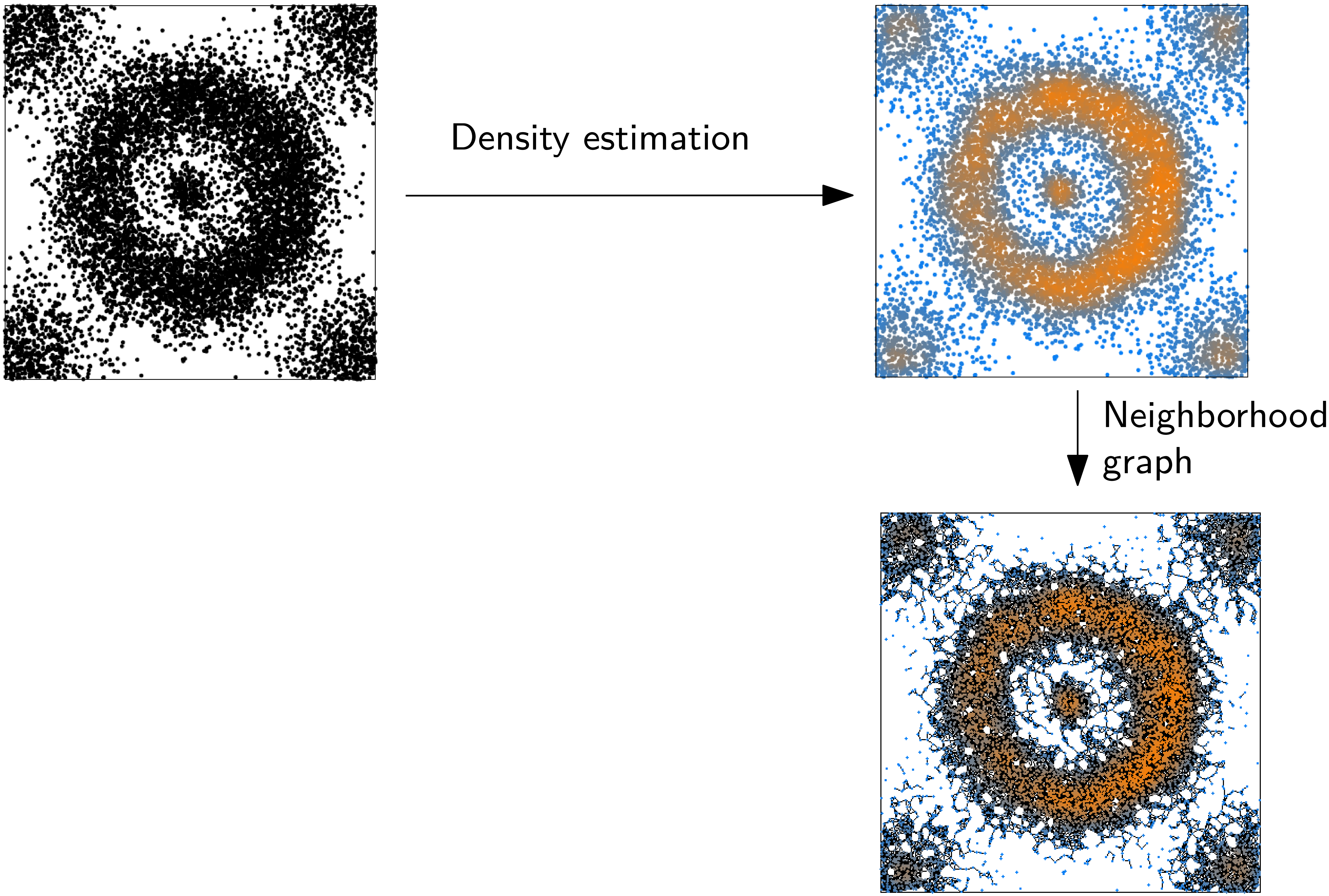
The Koonz, Narendra and Fukunaga algorithm (1976)



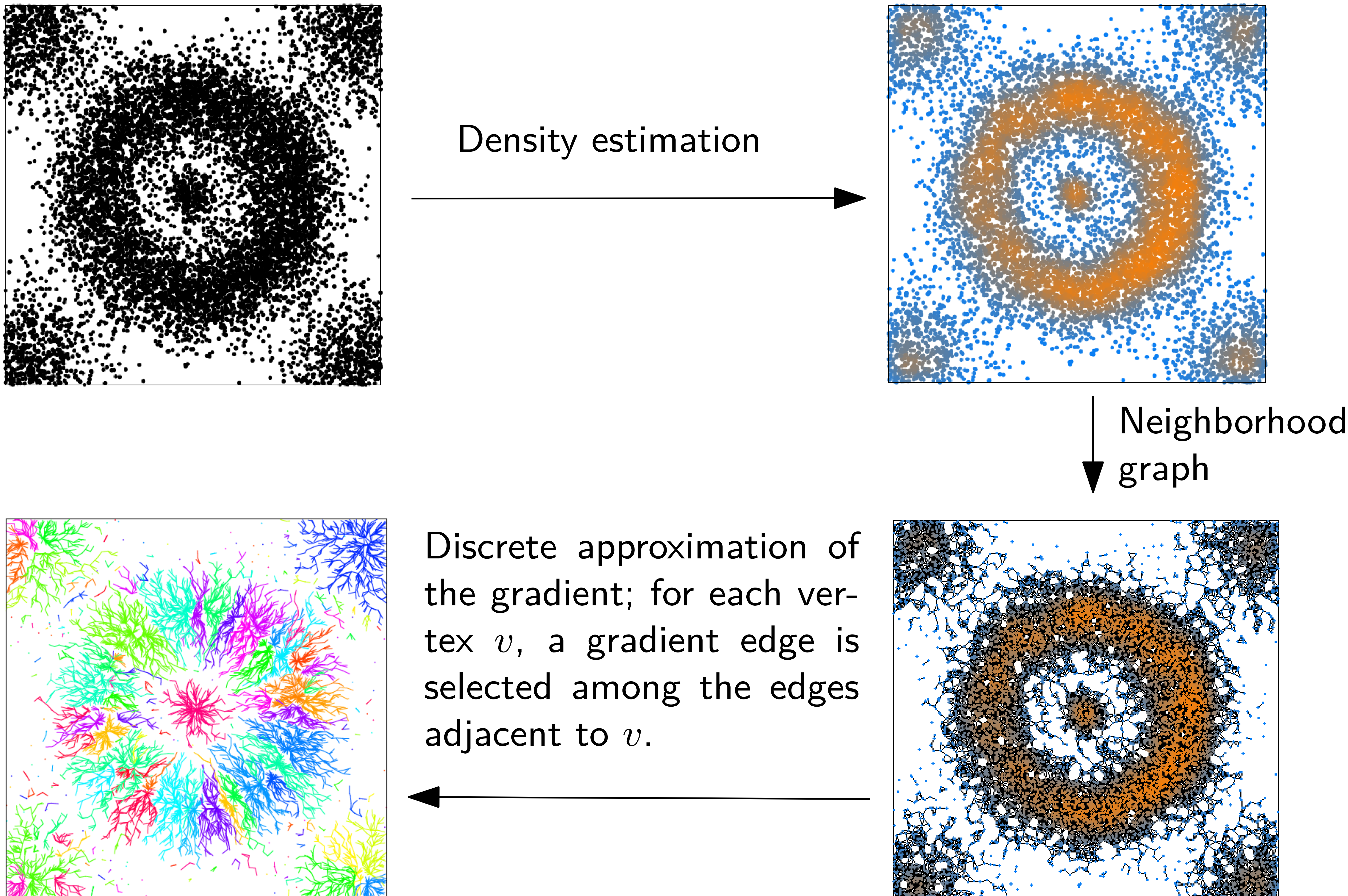
The Koonz, Narendra and Fukunaga algorithm (1976)



The Koonz, Narendra and Fukunaga algorithm (1976)

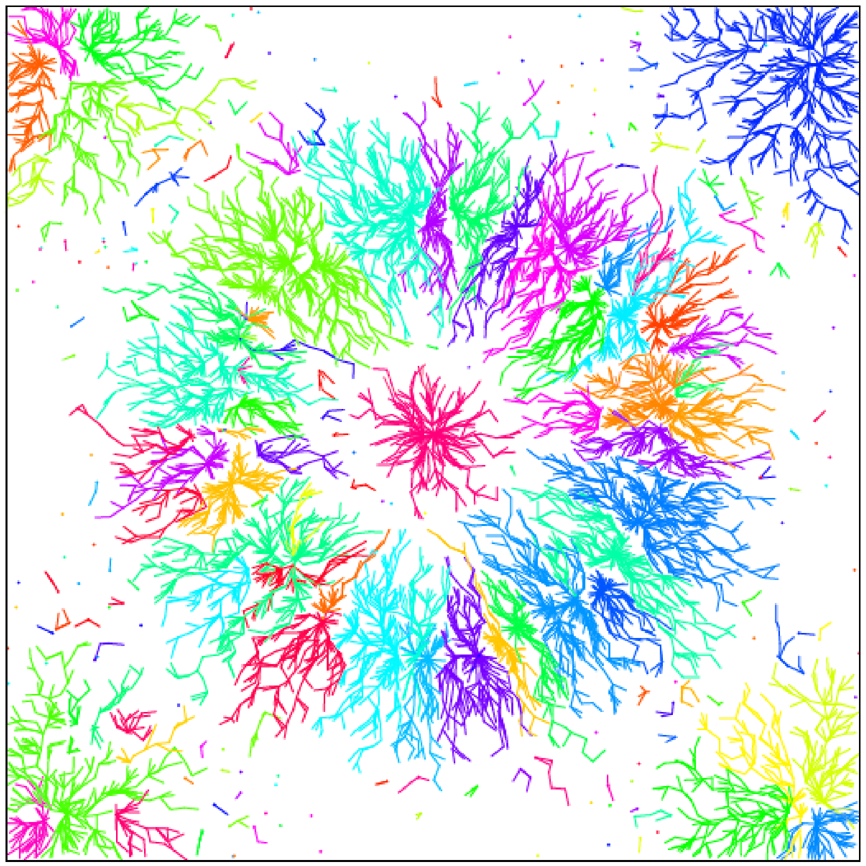


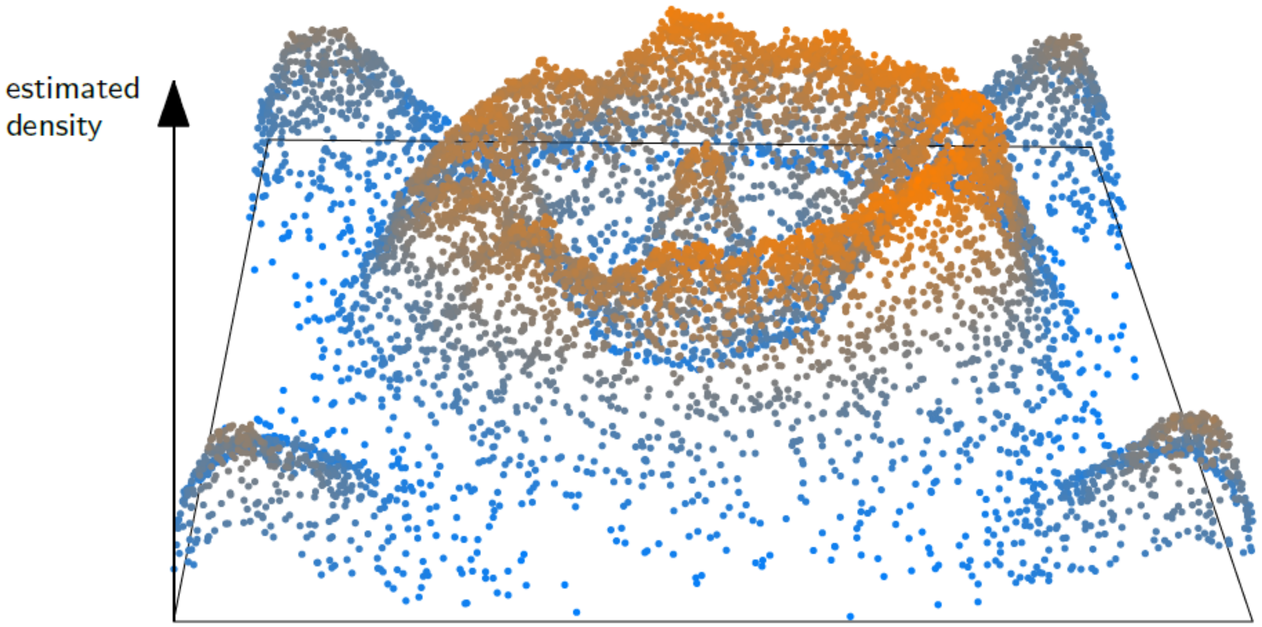
The Koonz, Narendra and Fukunaga algorithm (1976)



The Koonz, Narendra and Fukunaga algorithm (1976)

Drawbacks:

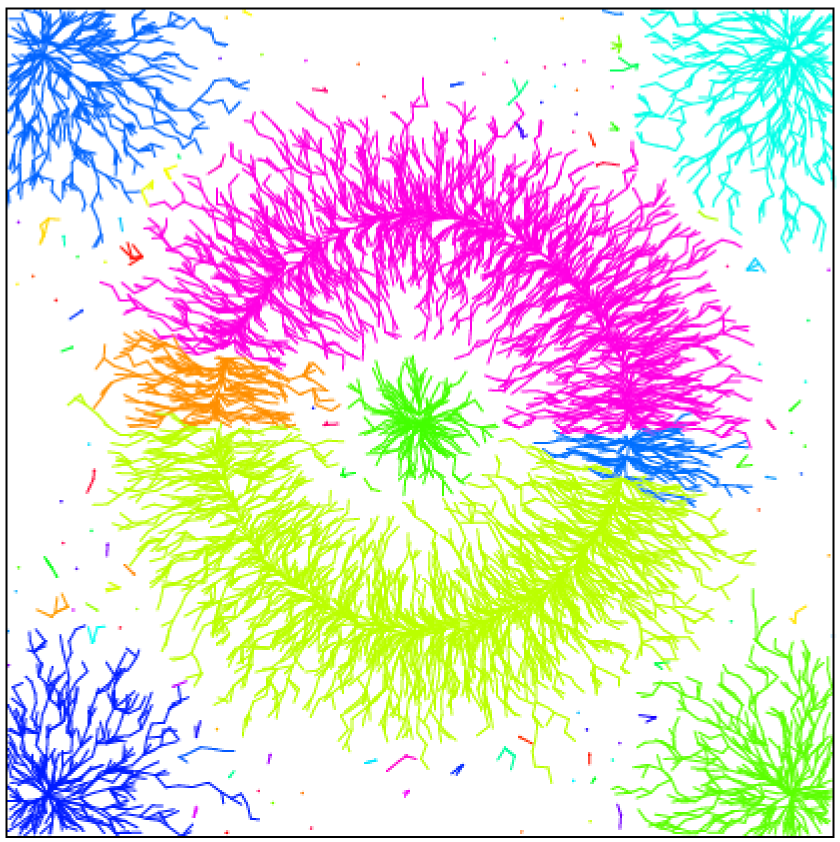


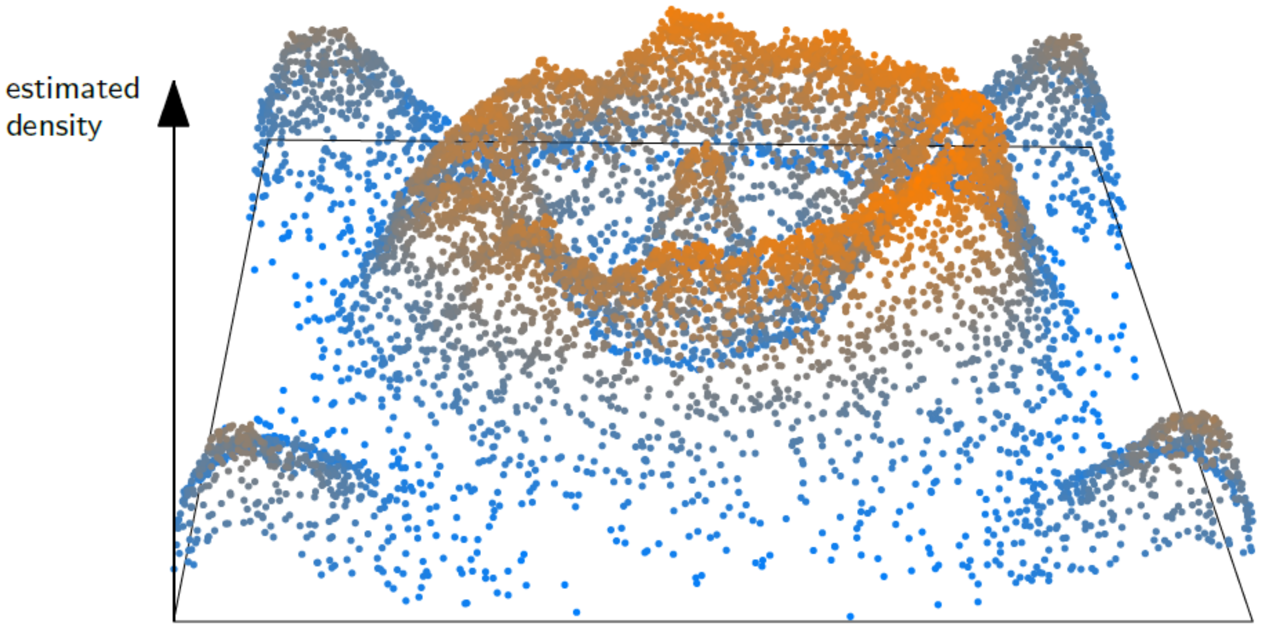


Pb 1: As many clusters as local maxima of the density $\rightarrow$ sensitivity to noise!

The Koonz, Narendra and Fukunaga algorithm (1976)

Drawbacks:



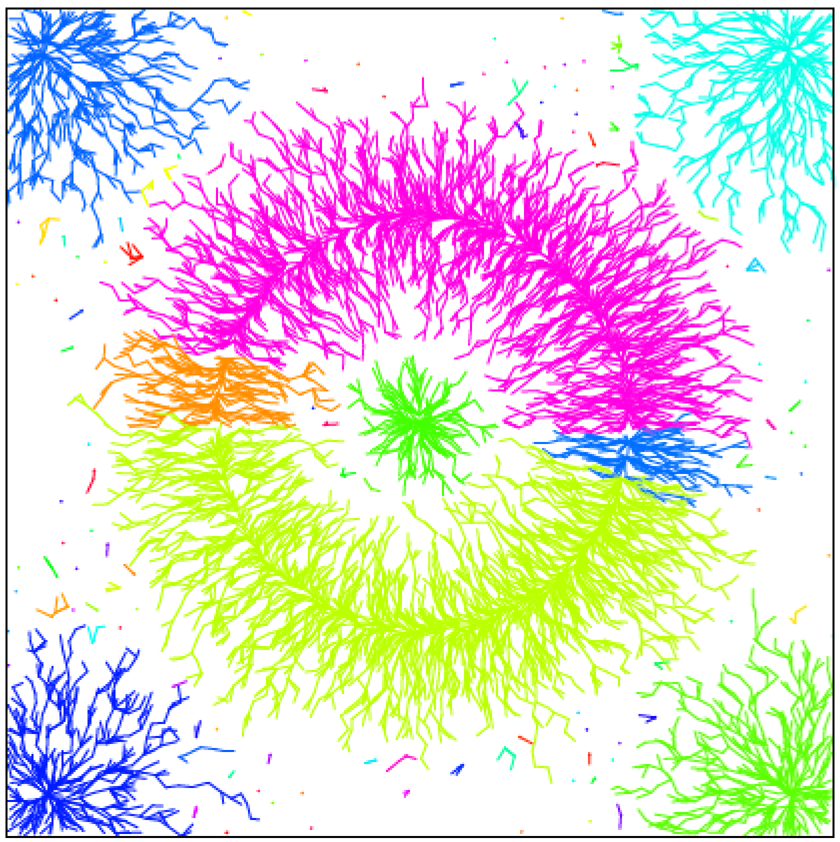


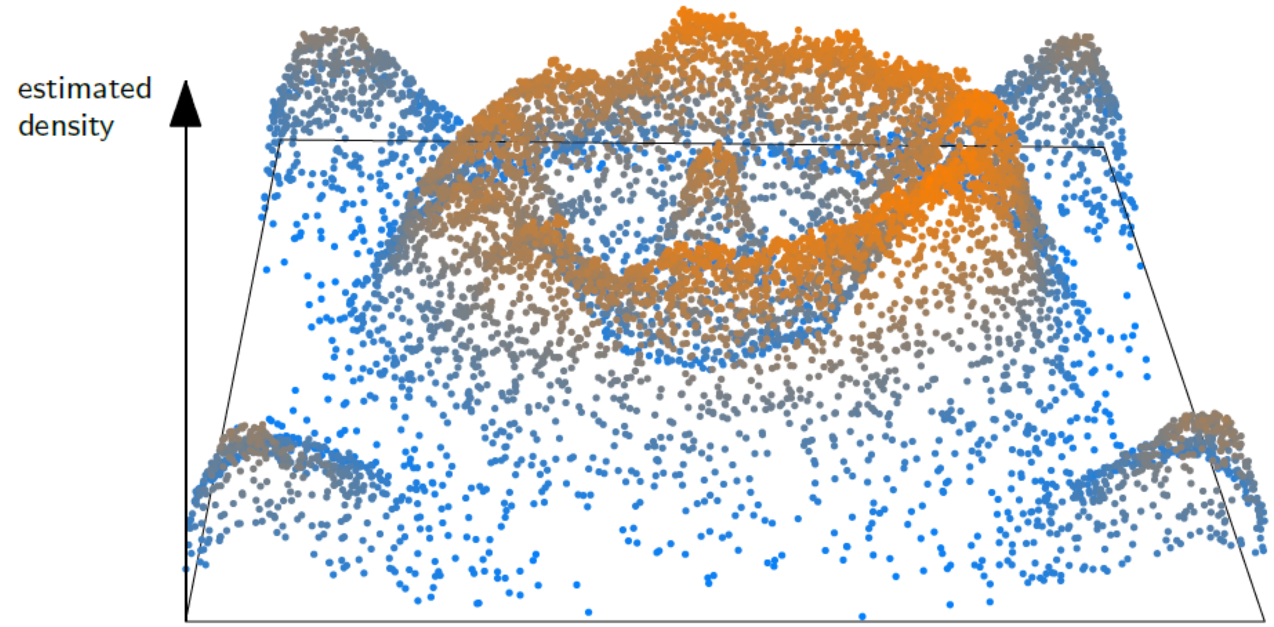
Pb 1: As many clusters as local maxima of the density $\rightarrow$ sensitivity to noise!

Pb 2: Choice of neighborhood graph may result in wide changes in the output.

The Koonz, Narendra and Fukunaga algorithm (1976)

Drawbacks:





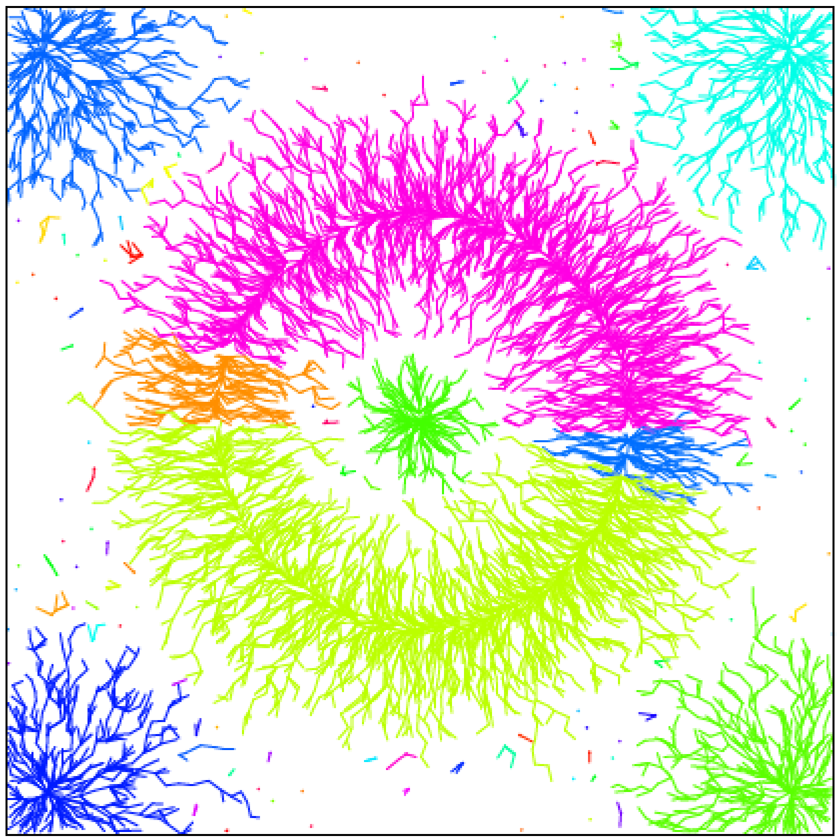
Pb 1: As many clusters as local maxima of the density $\rightarrow$ sensitivity to noise!

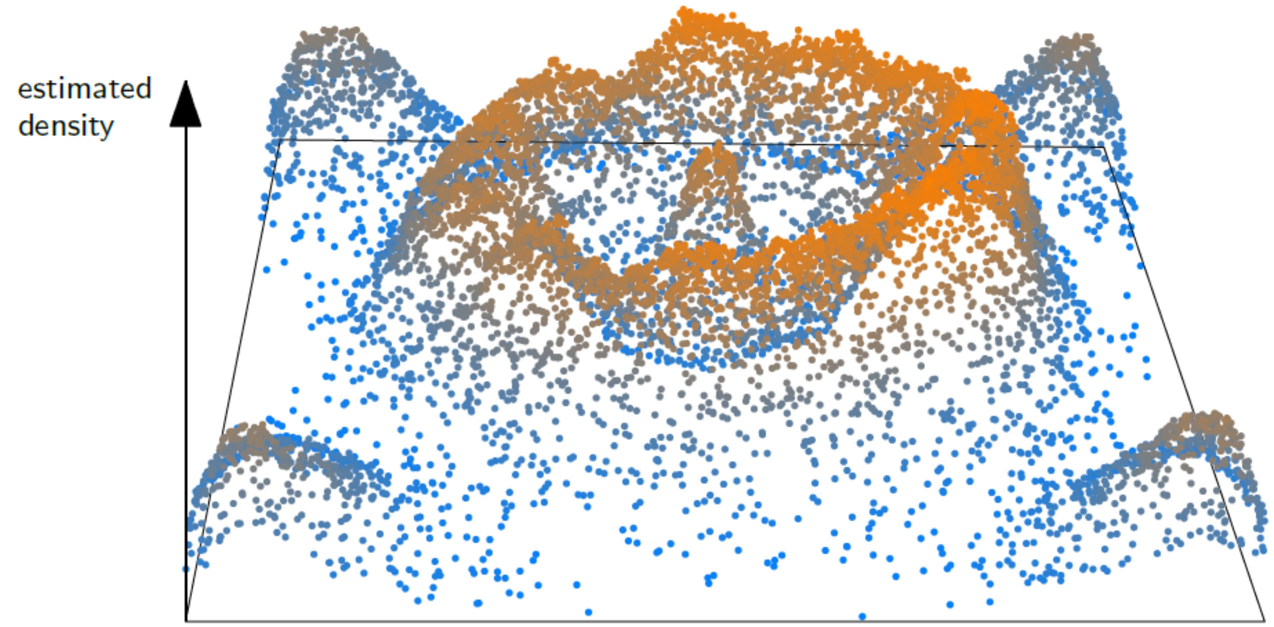
Pb 2: Choice of neighborhood graph may result in wide changes in the output.

$\rightarrow$ Smooth out the density estimate (e.g. mean-shift)... But how to choose the smoothing parameter?

The Koonz, Narendra and Fukunaga algorithm (1976)

Drawbacks:





Pb 1: As many clusters as local maxima of the density $\rightarrow$ sensitivity to noise!

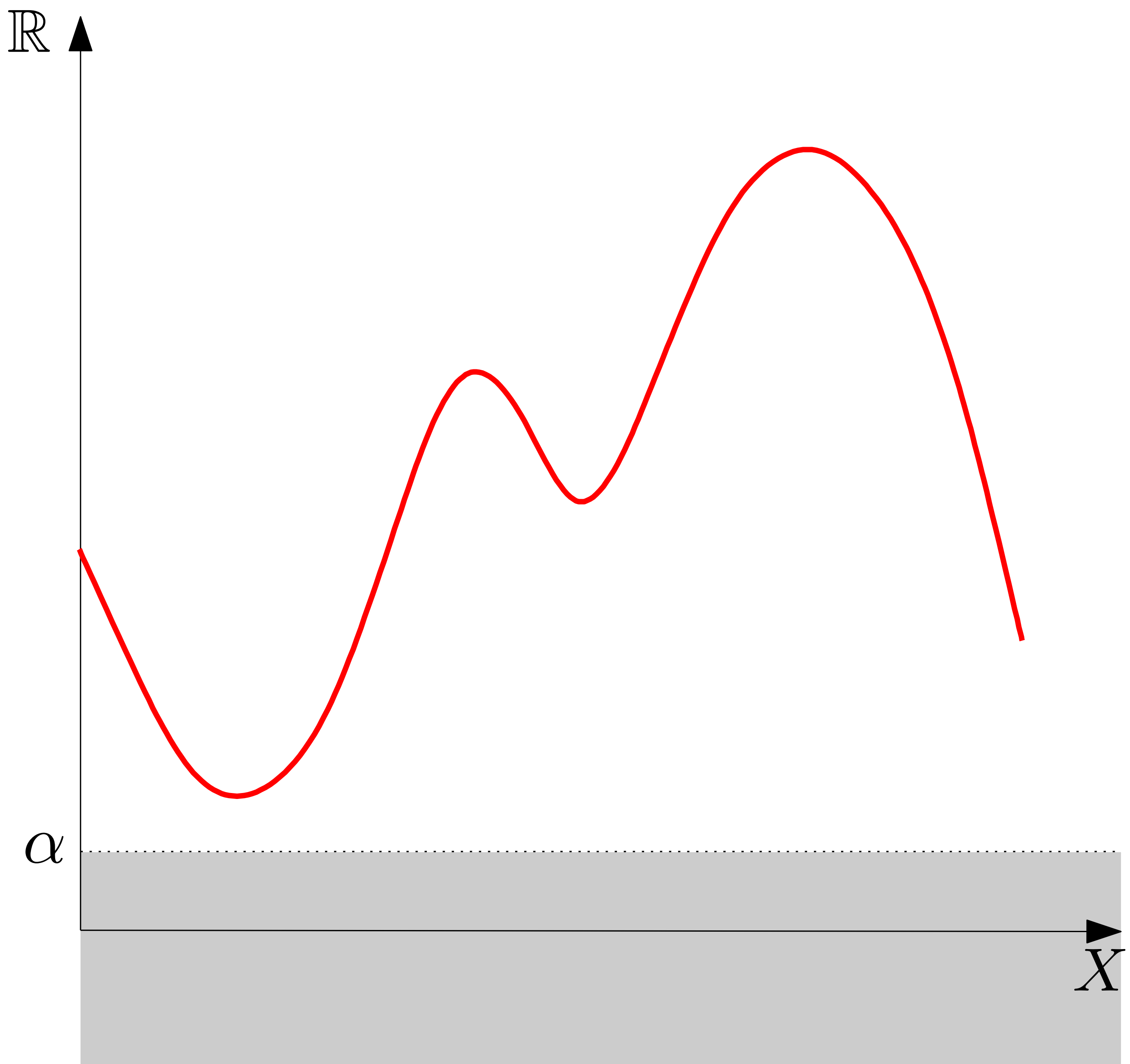
Pb 2: Choice of neighborhood graph may result in wide changes in the output.

$\rightarrow$ Smooth out the density estimate (e.g. mean-shift)... But how to choose the smoothing parameter?

$\rightarrow$ Merge clusters with persistent homology!

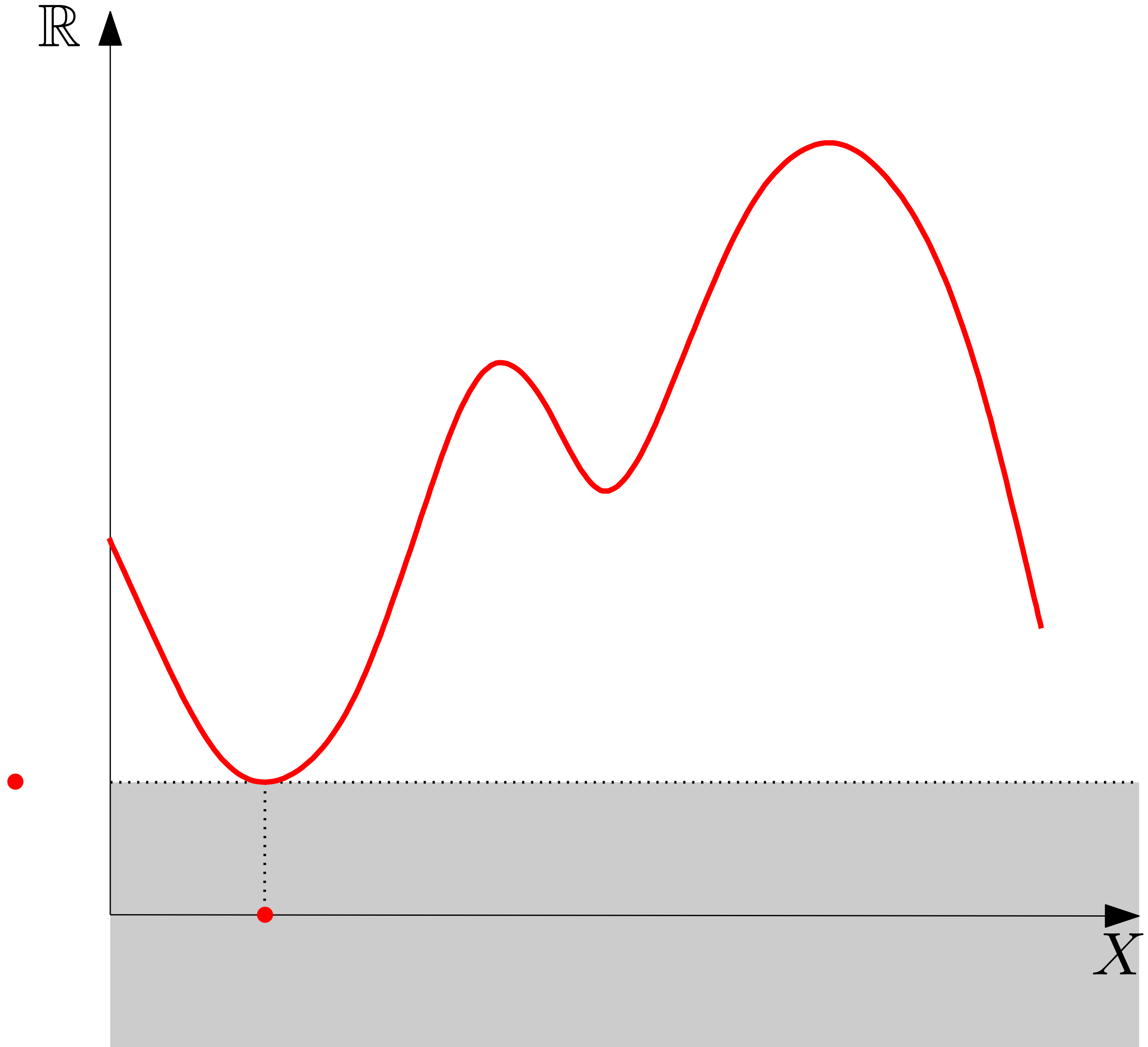
(0-dimensional) persistent homology for functions

- Nested family (filtration) of sublevel-sets $f^{-1}((-\infty, \alpha])$ for $\alpha = -\infty$ to $+\infty$.
- Track evolution of connectedness throughout the family.



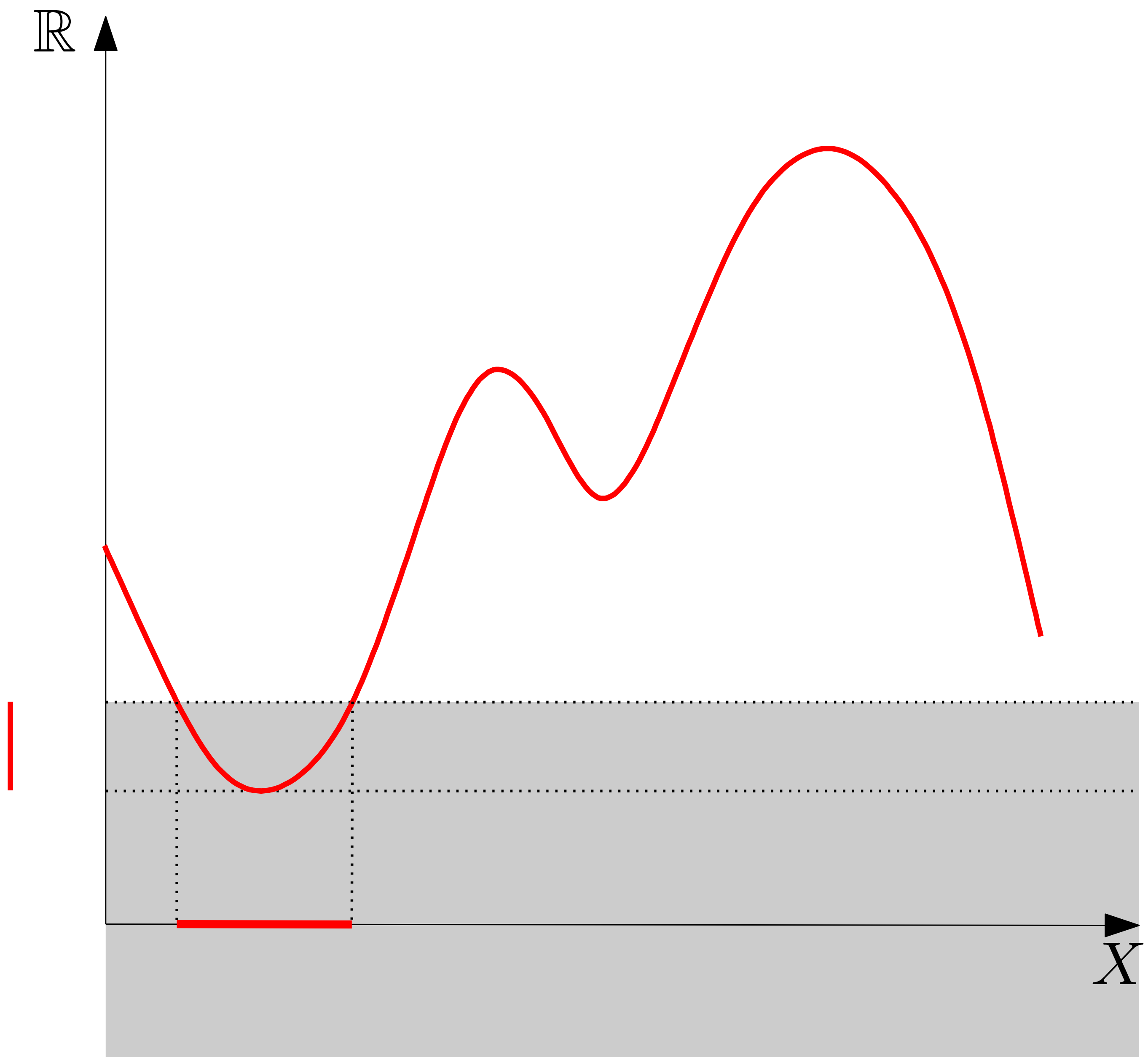
(0-dimensional) persistent homology for functions

- Nested family (filtration) of sublevel-sets $f^{-1}((-\infty, \alpha])$ for $\alpha = -\infty$ to $+\infty$.
- Track evolution of connectedness throughout the family.



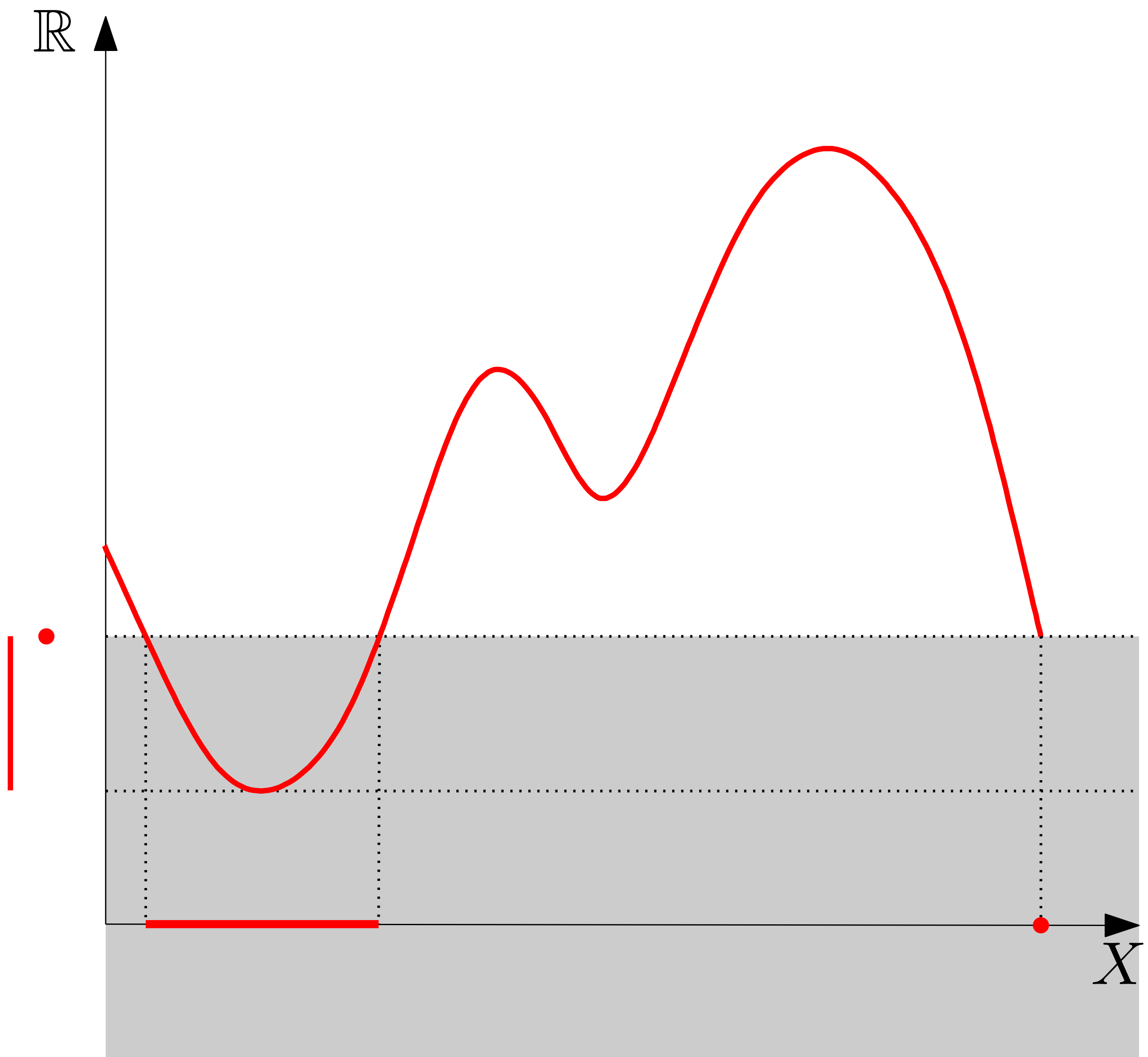
(0-dimensional) persistent homology for functions

- Nested family (filtration) of sublevel-sets $f^{-1}((-\infty, \alpha])$ for $\alpha = -\infty$ to $+\infty$.
- Track evolution of connectedness throughout the family.



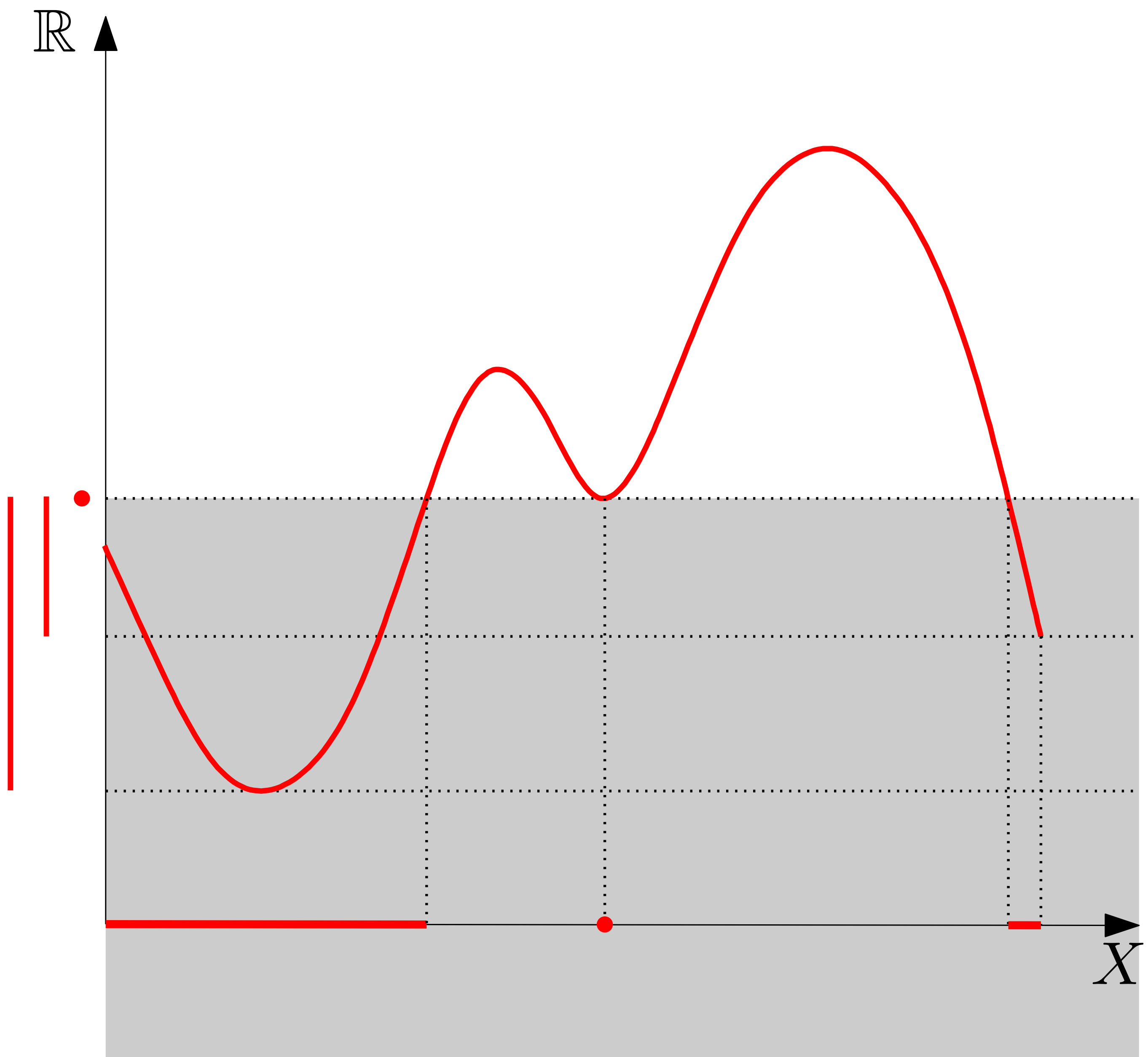
(0-dimensional) persistent homology for functions

- Nested family (filtration) of sublevel-sets $f^{-1}((-\infty, \alpha])$ for $\alpha = -\infty$ to $+\infty$.
- Track evolution of connectedness throughout the family.



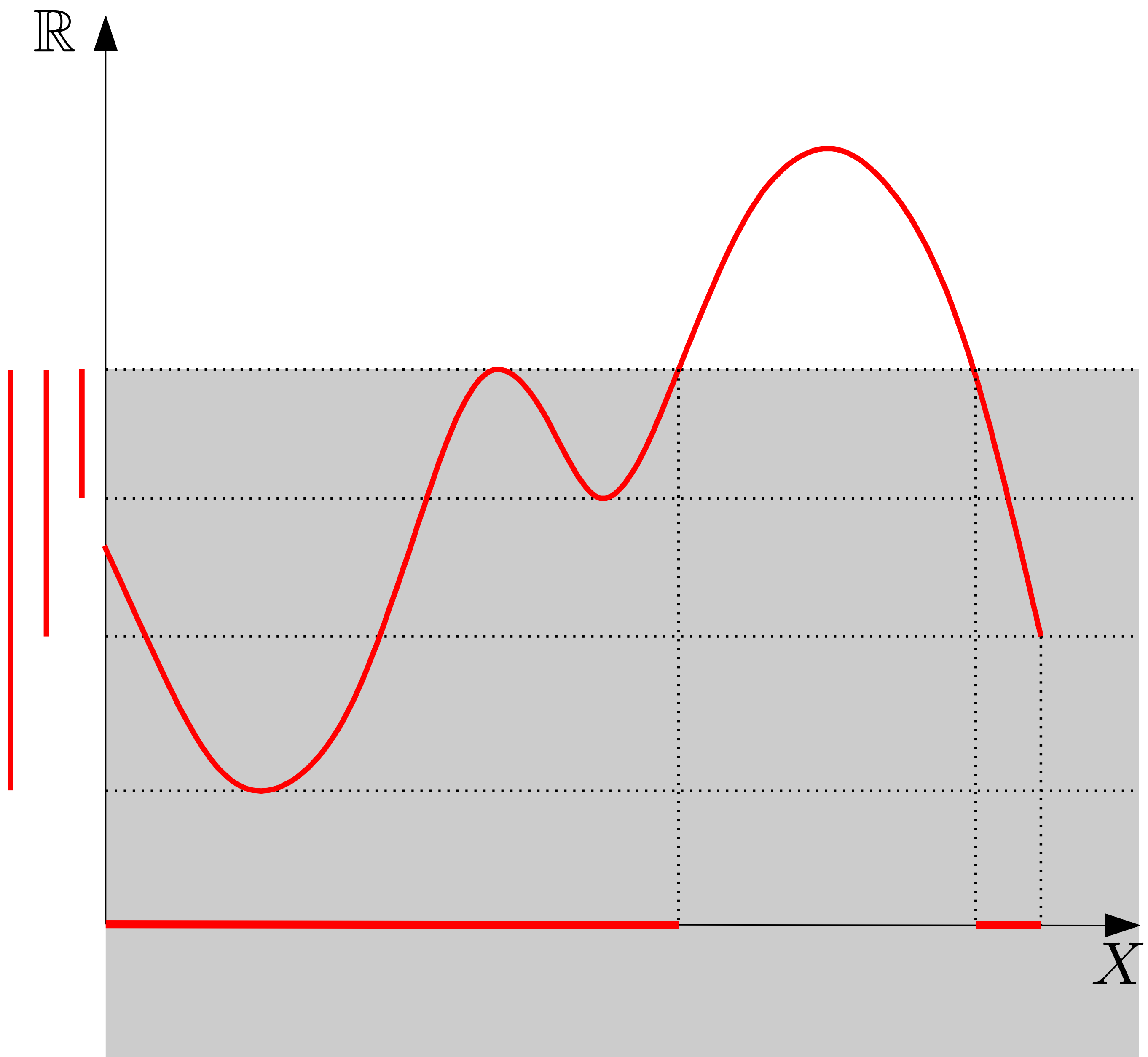
(0-dimensional) persistent homology for functions

- Nested family (filtration) of sublevel-sets $f^{-1}((-\infty, \alpha])$ for $\alpha = -\infty$ to $+\infty$.
- Track evolution of connectedness throughout the family.



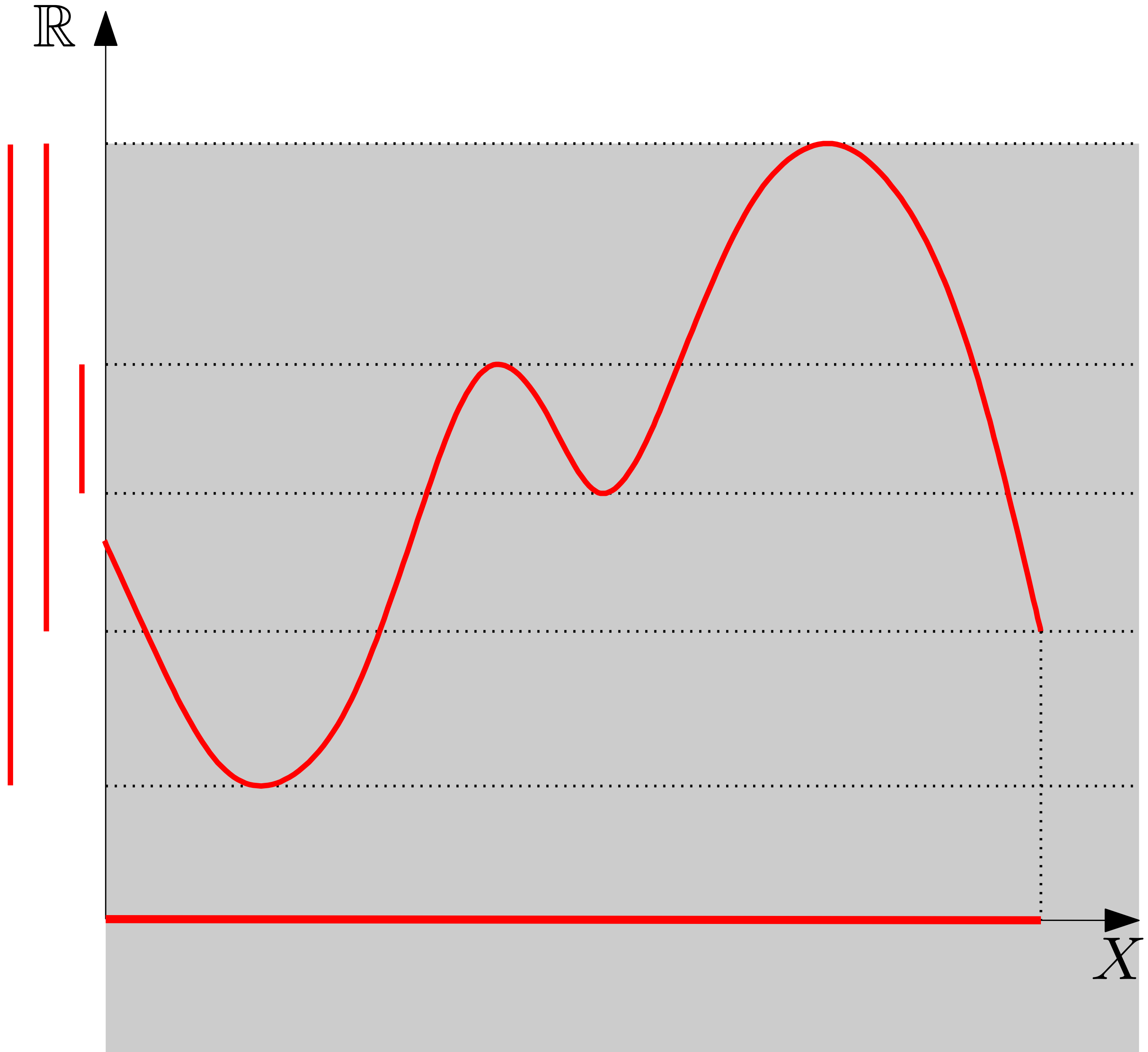
(0-dimensional) persistent homology for functions

- Nested family (filtration) of sublevel-sets $f^{-1}((-\infty, \alpha])$ for $\alpha = -\infty$ to $+\infty$.
- Track evolution of connectedness throughout the family.



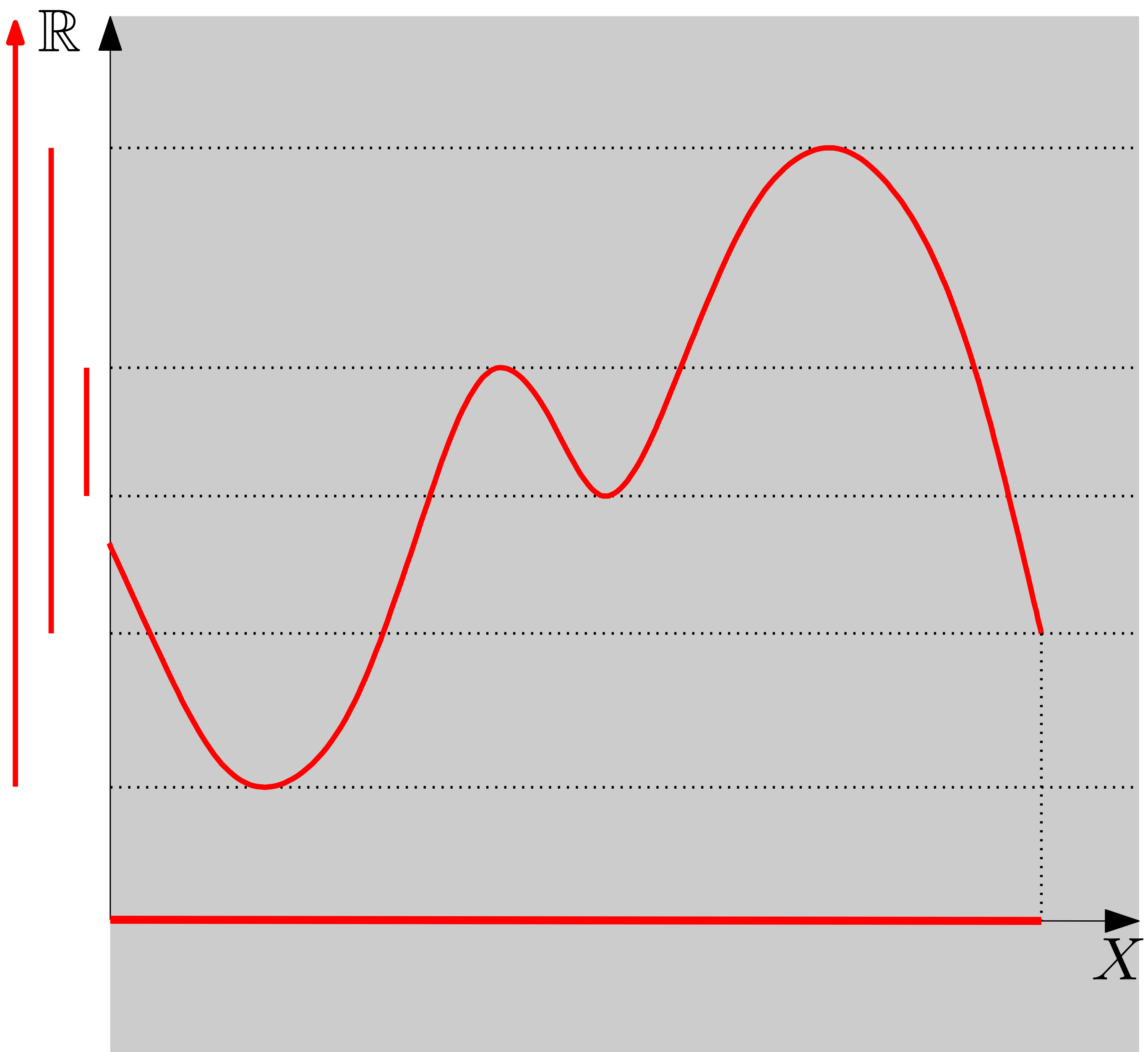
(0-dimensional) persistent homology for functions

- Nested family (filtration) of sublevel-sets $f^{-1}((-\infty, \alpha])$ for $\alpha = -\infty$ to $+\infty$.
- Track evolution of connectedness throughout the family.



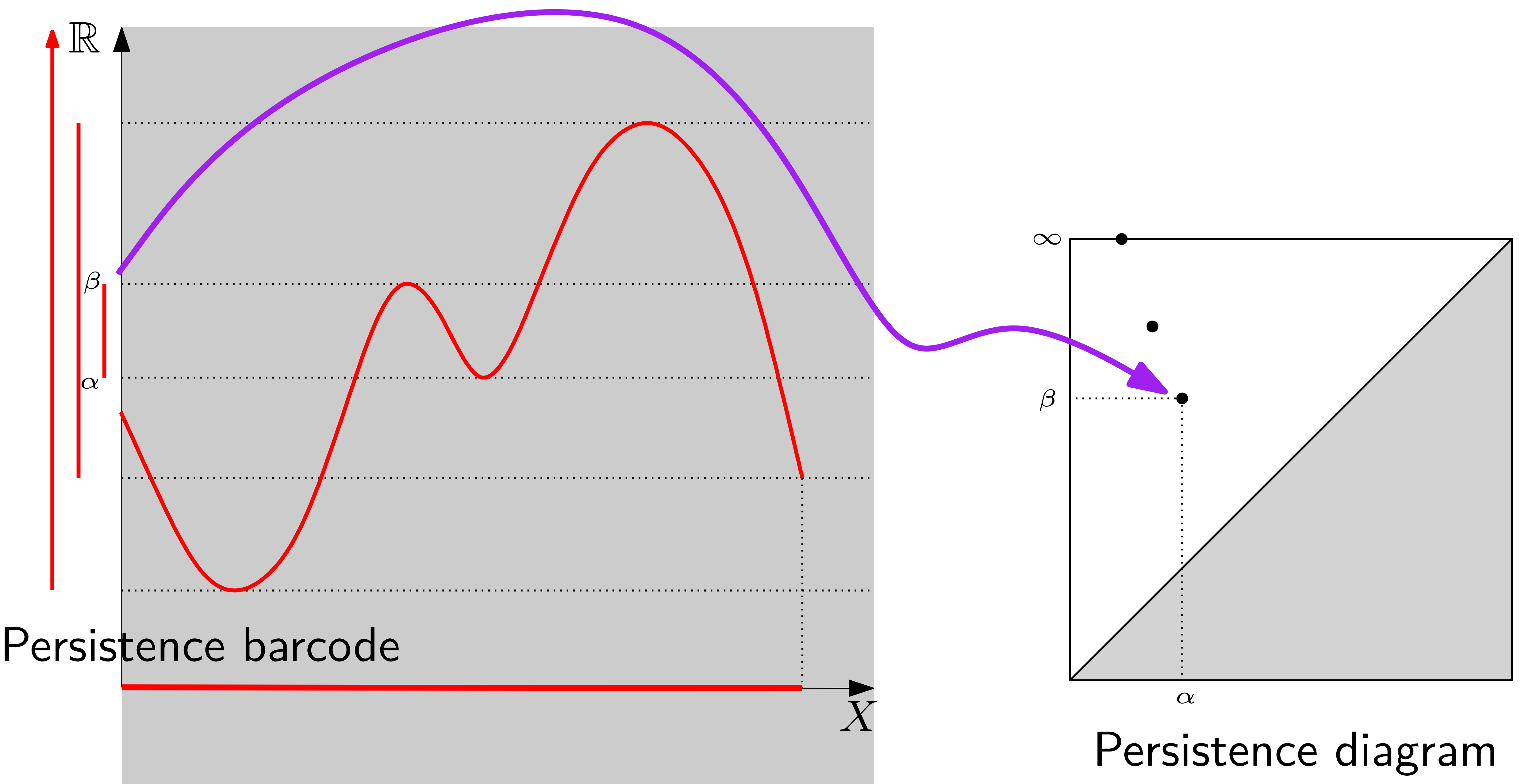
(0-dimensional) persistent homology for functions

- Nested family (filtration) of sublevel-sets $f^{-1}((-\infty, \alpha])$ for $\alpha = -\infty$ to $+\infty$.
- Track evolution of connectedness throughout the family.
- Finite set of intervals (barcode) encodes births/deaths of topological features.



(0-dimensional) persistent homology for functions

- Nested family (filtration) of sublevel-sets $f^{-1}((-\infty, \alpha])$ for $\alpha = -\infty$ to $+\infty$.
- Track evolution of connectedness throughout the family.
- Finite set of intervals (barcode) encodes births/deaths of topological features.

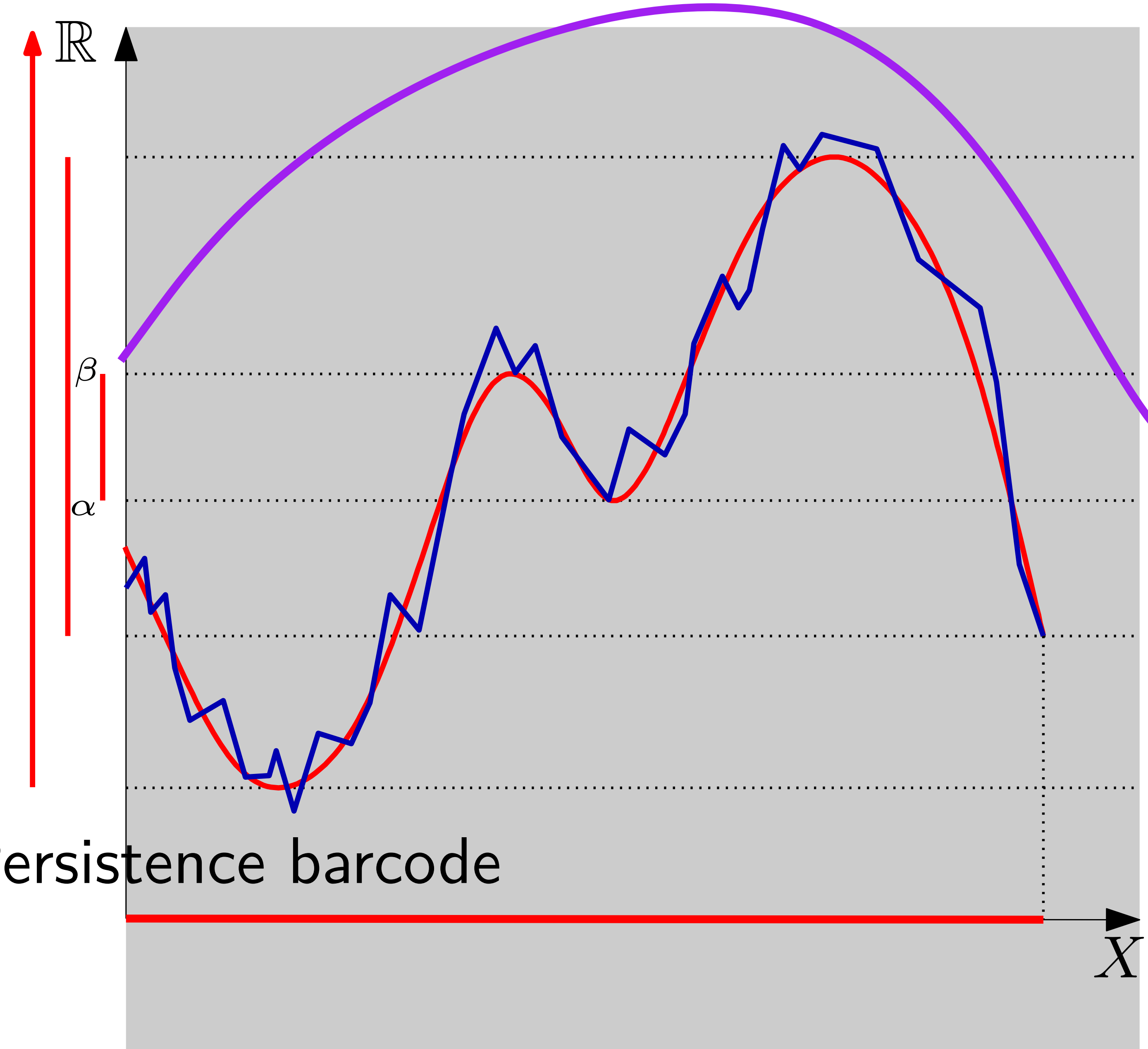


(0-dimensional) persistent homology for functions

Thm: (Stability)
For any *tame* functions $f, g : \mathbb{X} \to \mathbb{R}$, $d_B(D_f, D_g) \leq \|f - g\|_\infty$.

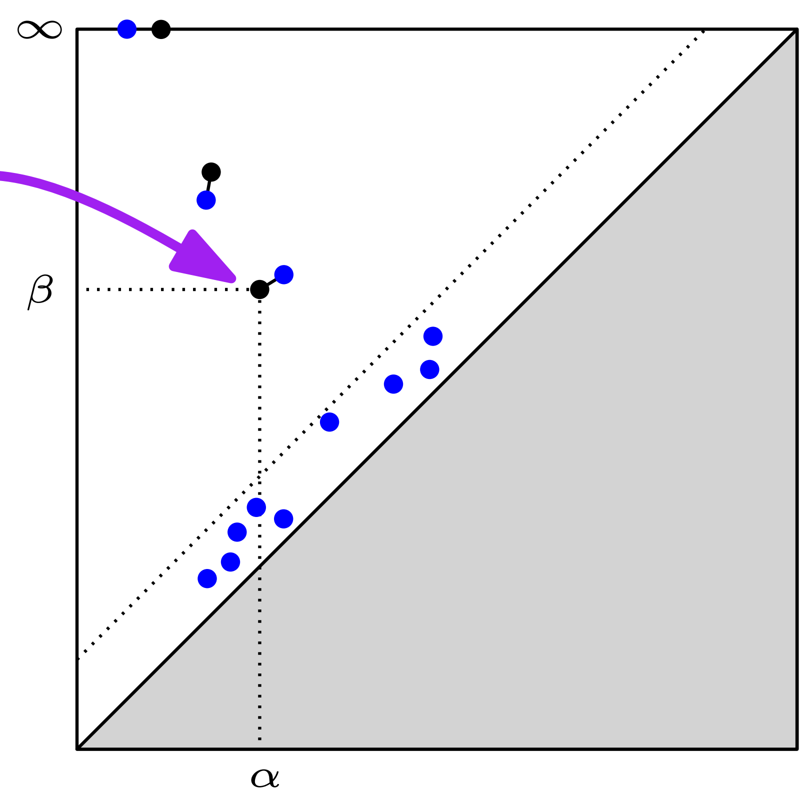
[*Stability of Persistence Diagrams*, Cohen-Steiner, Edelsbrunner, Harer, Symp. Comput. Geom., 2005]

[*The structure and stability of persistence modules*, Chazal, de Silva, Glisse, Oudot, AMS, 2012]



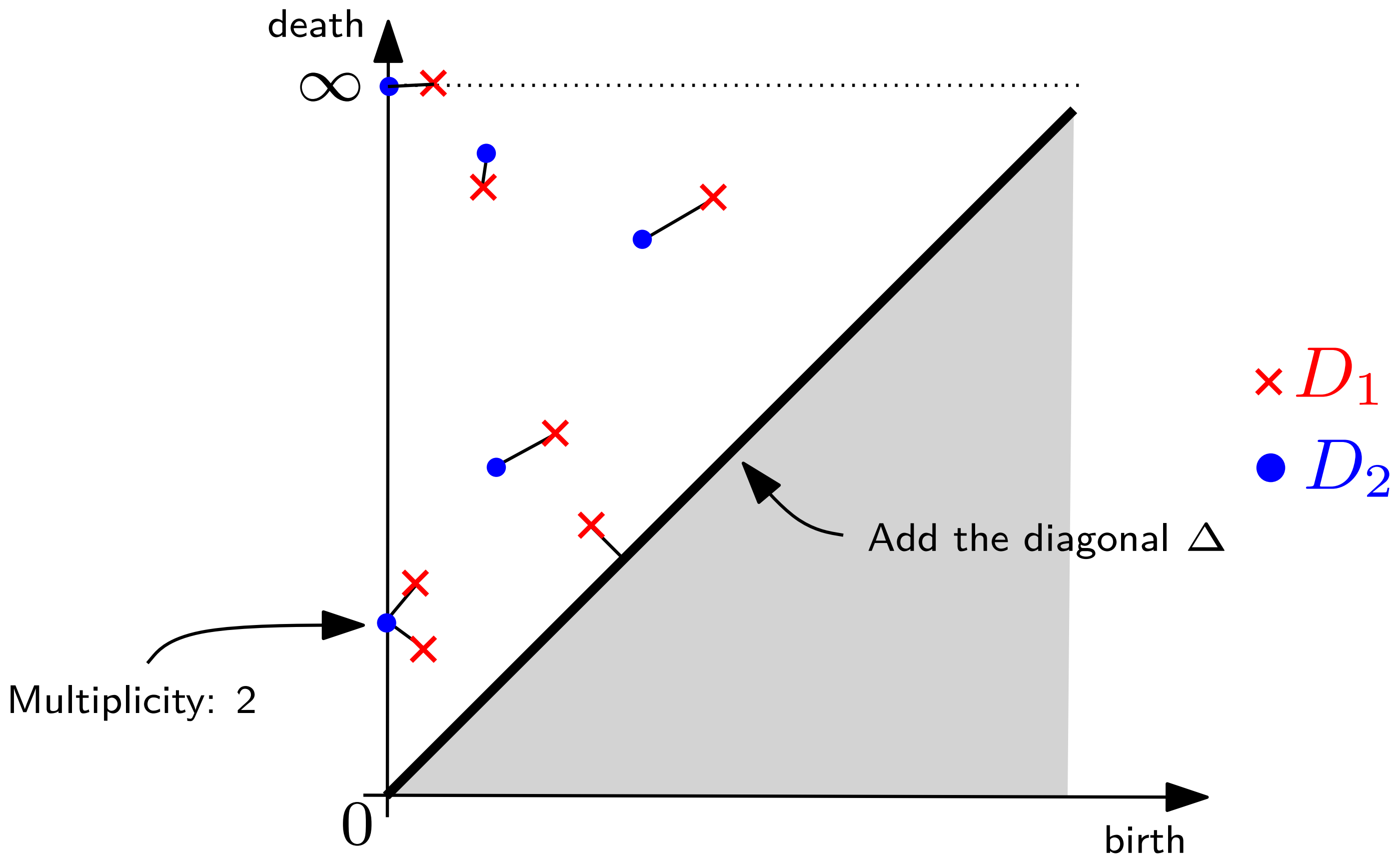
Persistence barcode

What if f is slightly perturbed?



Persistence diagram

Distance between persistence diagrams

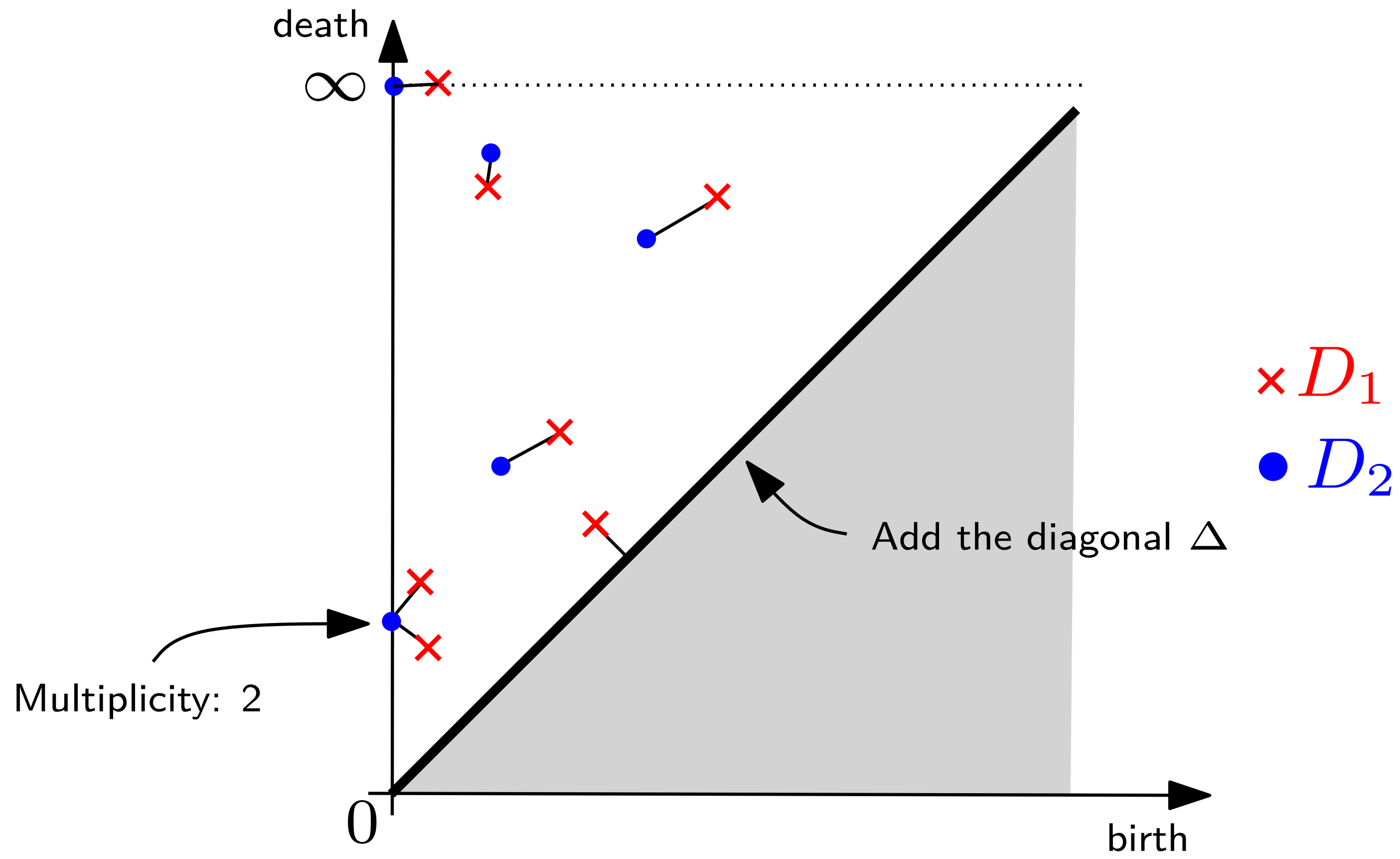


Def: The bottleneck distance between two diagrams D_1 and D_2 is

$$d_B(D_1, D_2) = \inf_{\gamma \in \Gamma} \sup_{p \in D_1} \|p - \gamma(p)\|_\infty$$

where Γ is the set of all the bijections between $D_1 \cup \Delta$ and $D_2 \cup \Delta$ and $\|p - q\|_\infty = \max(|x_p - x_q|, |y_p - y_q|)$.

Distance between persistence diagrams



Def: The Wasserstein distance between two diagrams D_1 and D_2 is

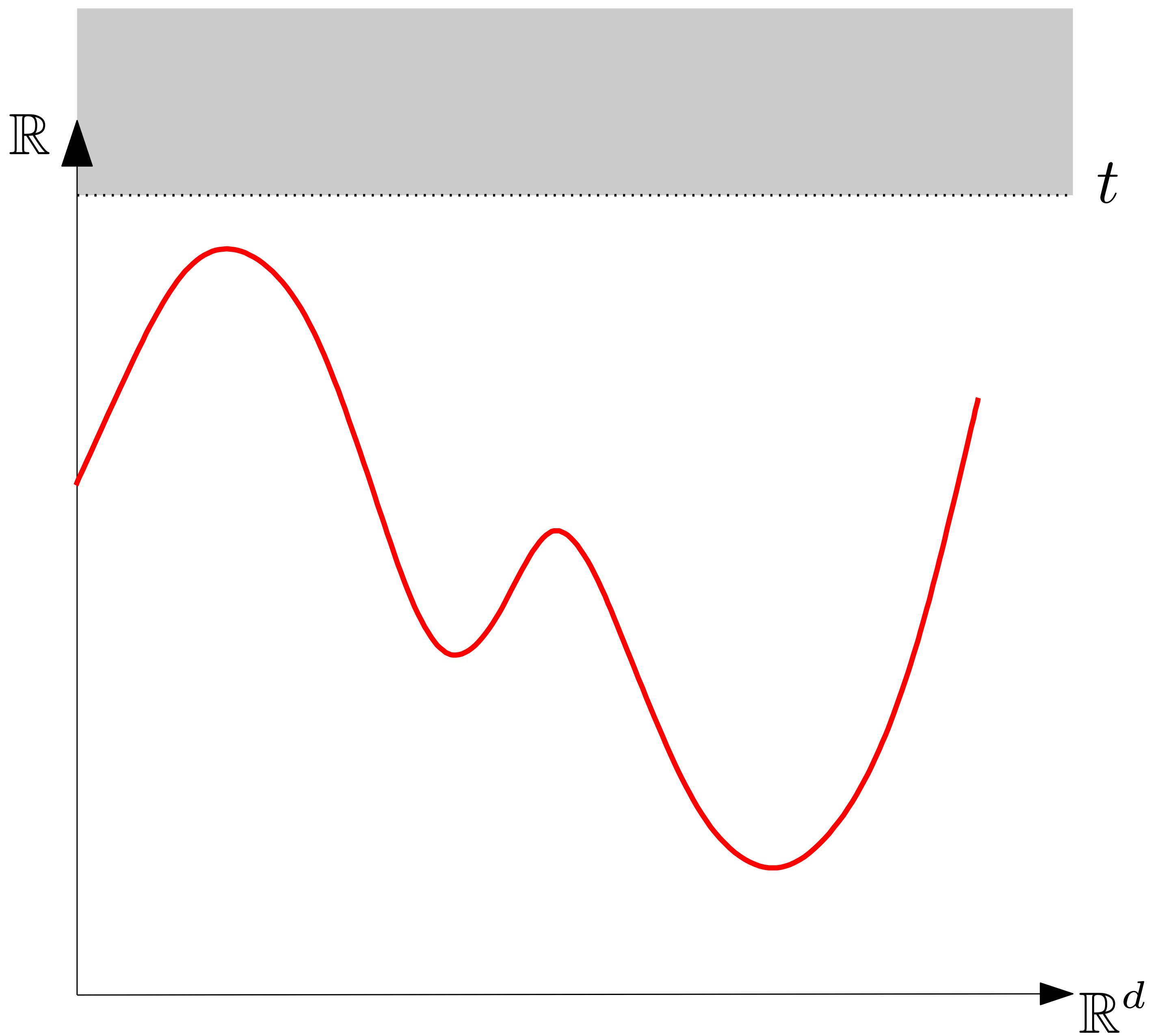
$$d_q(D_1, D_2)^q = \inf_{\gamma \in \Gamma} \sum_{p \in D_1} \|p - \gamma(p)\|_\infty^q$$

where Γ is the set of all the bijections between $D_1 \cup \Delta$ and $D_2 \cup \Delta$ and $\|p - q\|_\infty = \max(|x_p - x_q|, |y_p - y_q|)$.

The case of density and back to mode seeking

Given a probability density f:

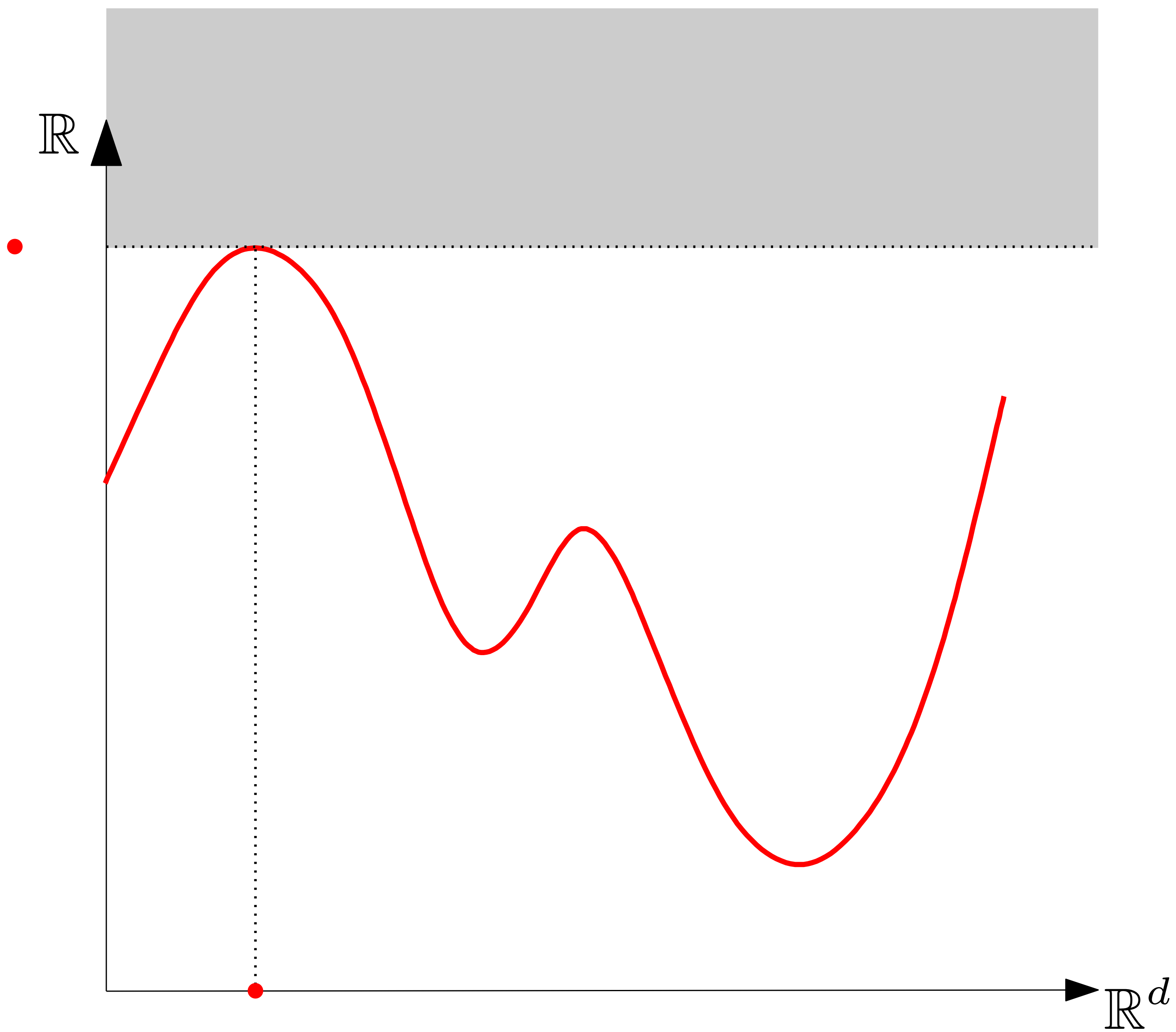
- Consider the superlevel-sets filtration $f^{-1}([t, +\infty))$ for t from $+\infty$ to $-\infty$, instead of the sublevel-sets filtration.
- Persistence is defined in the same way



The case of density and back to mode seeking

Given a probability density f:

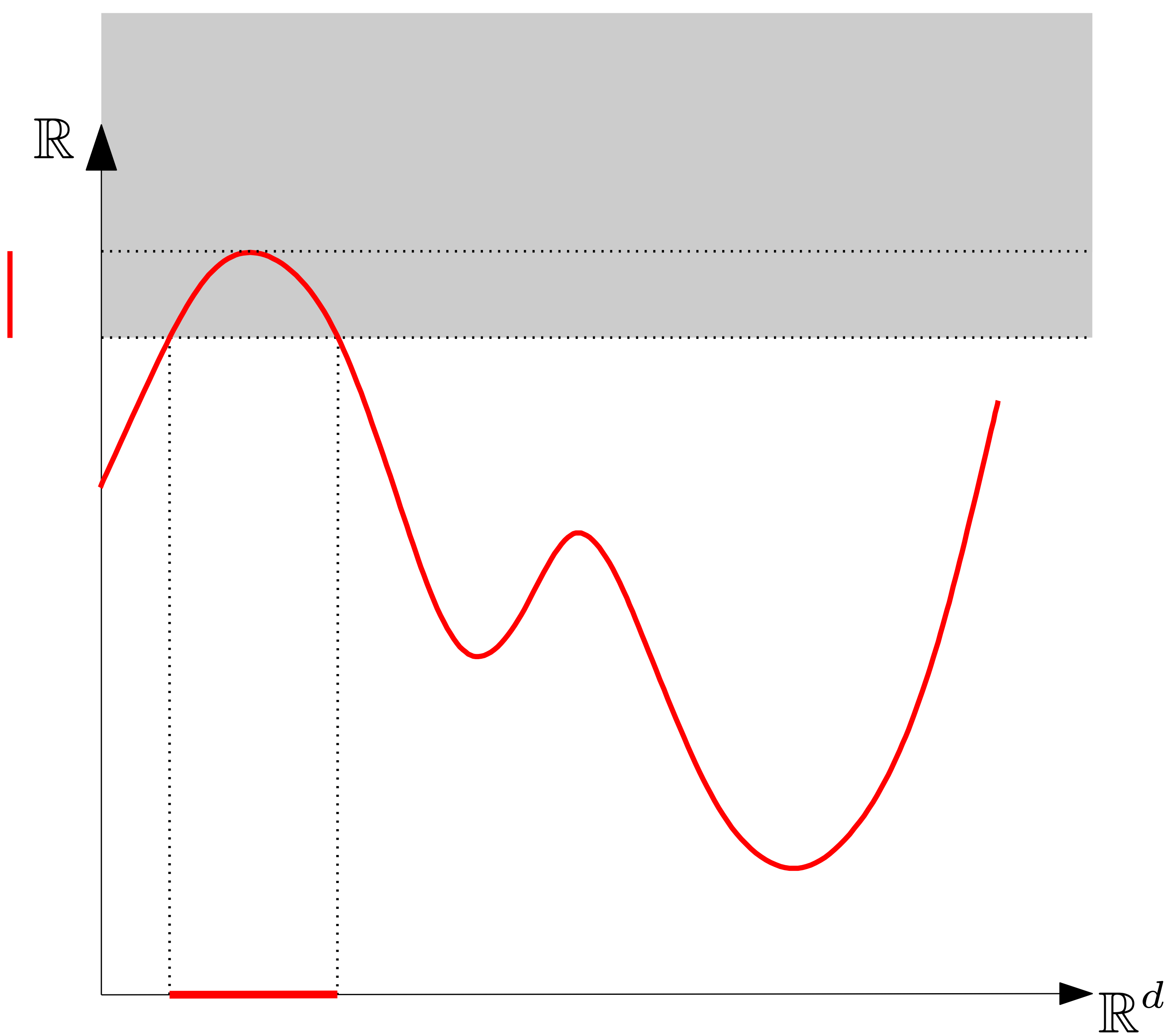
- Consider the superlevel-sets filtration $f^{-1}([t, +\infty))$ for t from $+\infty$ to $-\infty$, instead of the sublevel-sets filtration.
- Persistence is defined in the same way



The case of density and back to mode seeking

Given a probability density f:

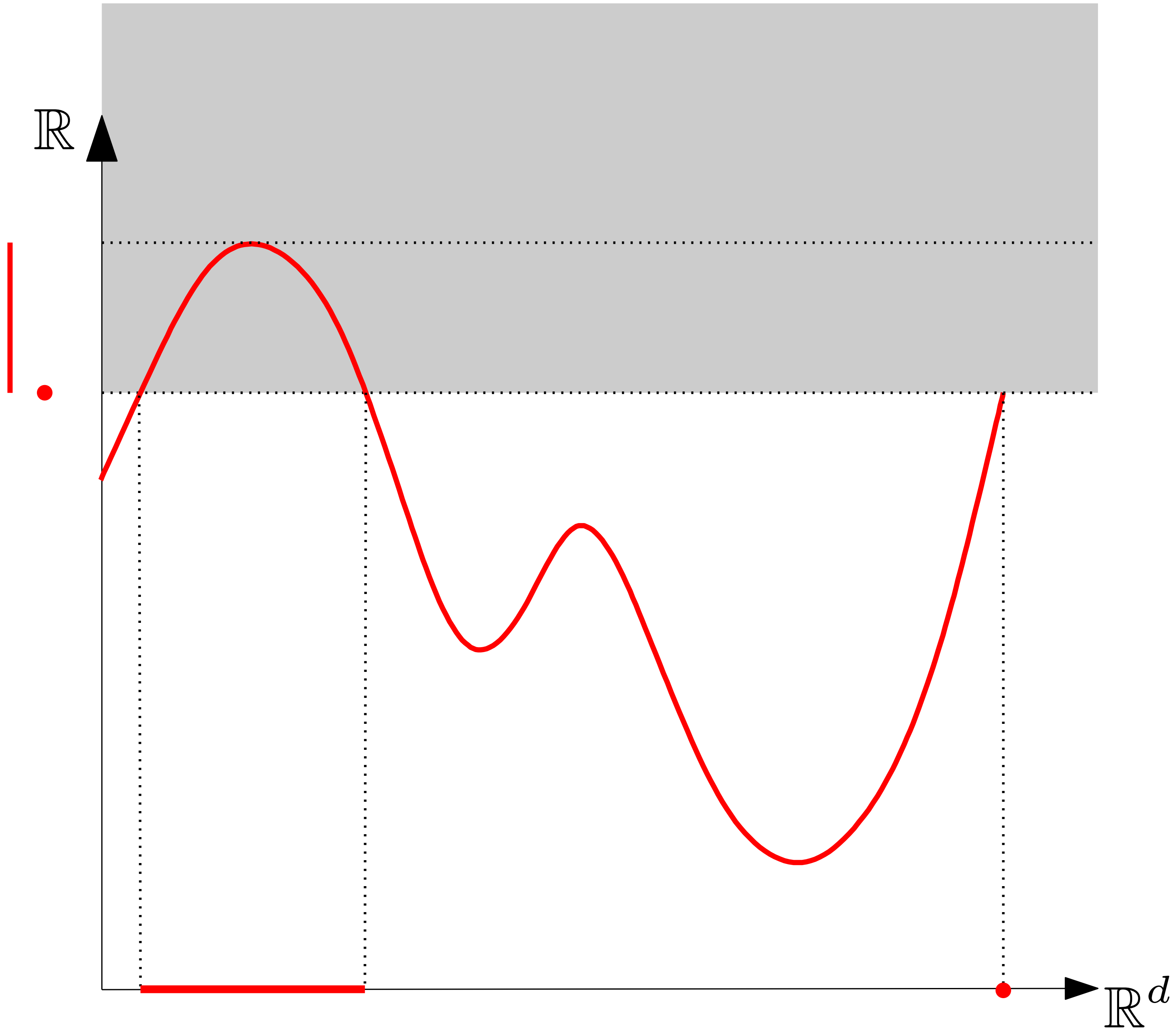
- Consider the superlevel-sets filtration $f^{-1}([t, +\infty))$ for t from $+\infty$ to $-\infty$, instead of the sublevel-sets filtration.
- Persistence is defined in the same way



The case of density and back to mode seeking

Given a probability density f:

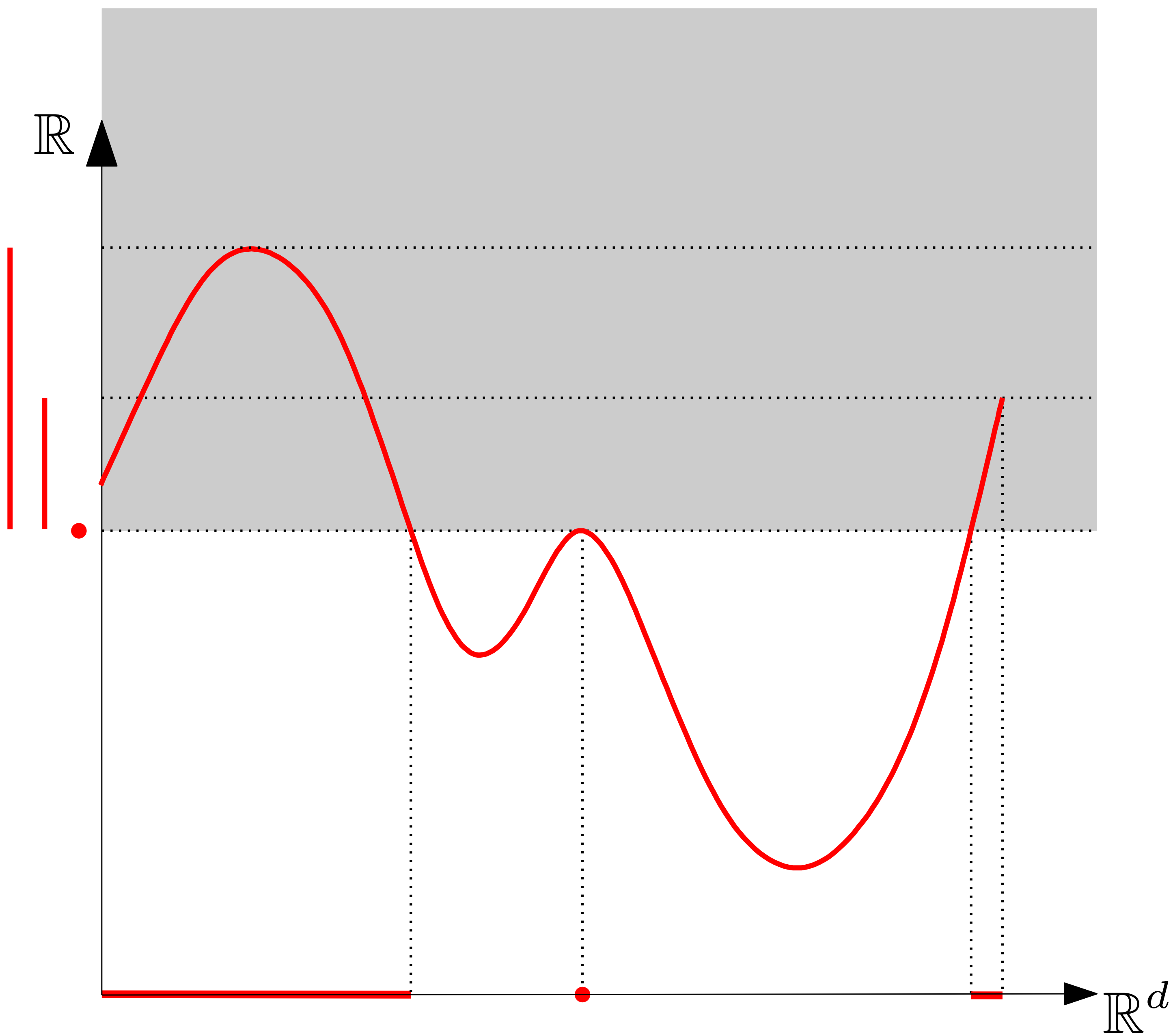
- Consider the superlevel-sets filtration $f^{-1}([t, +\infty))$ for t from $+\infty$ to $-\infty$, instead of the sublevel-sets filtration.
- Persistence is defined in the same way



The case of density and back to mode seeking

Given a probability density f:

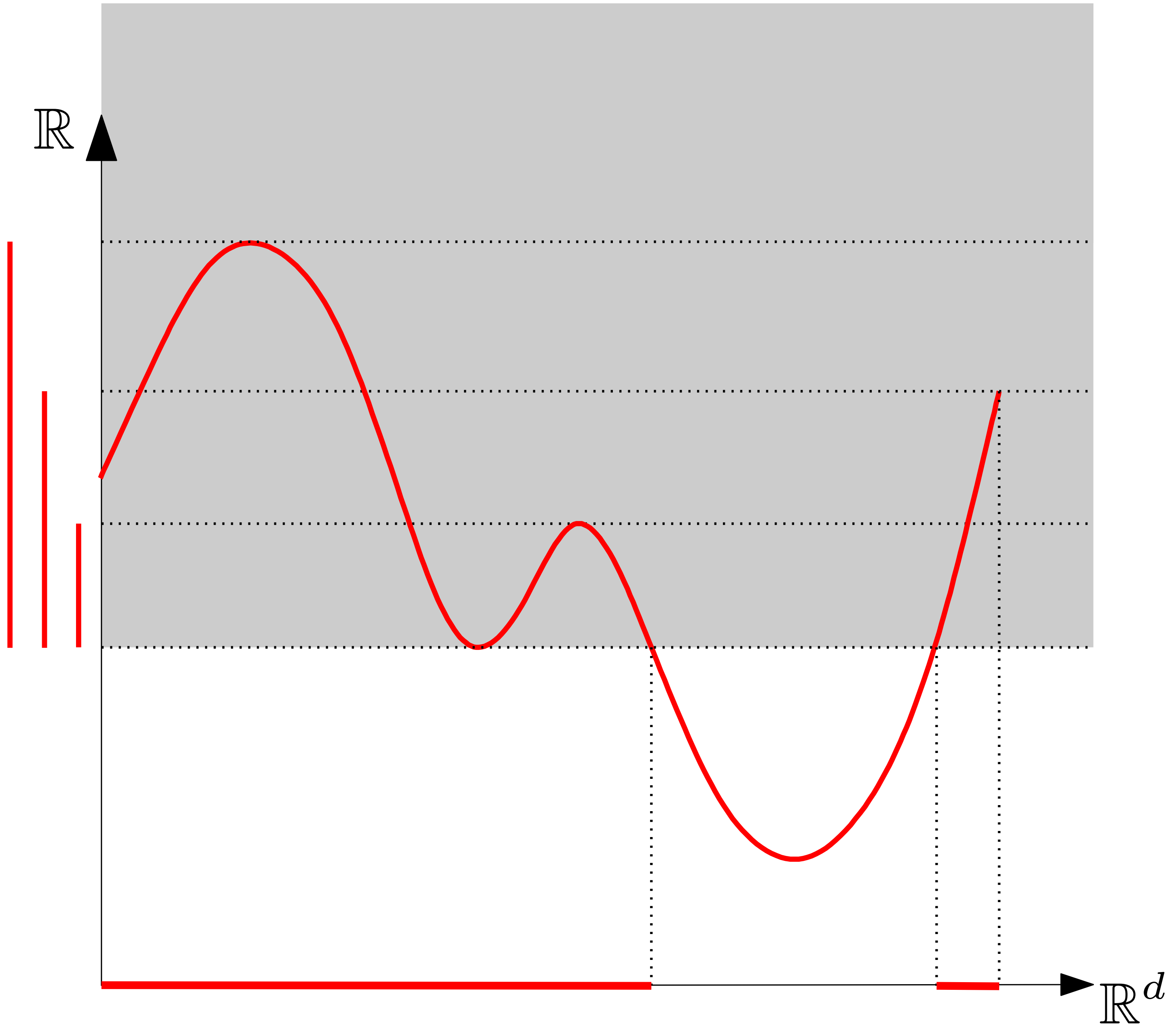
- Consider the superlevel-sets filtration $f^{-1}([t, +\infty))$ for t from $+\infty$ to $-\infty$, instead of the sublevel-sets filtration.
- Persistence is defined in the same way



The case of density and back to mode seeking

Given a probability density f:

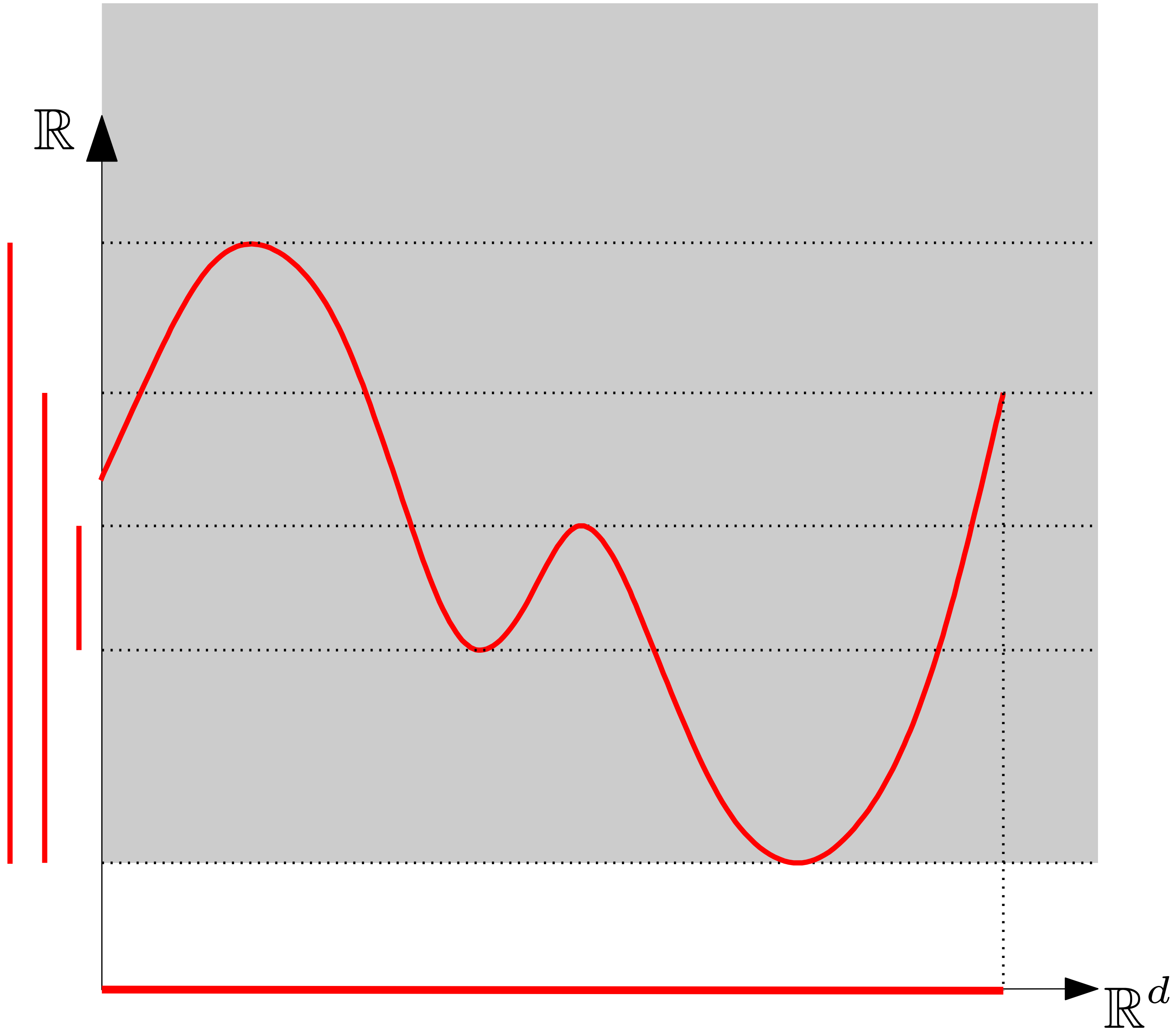
- Consider the superlevel-sets filtration $f^{-1}([t, +\infty))$ for t from $+\infty$ to $-\infty$, instead of the sublevel-sets filtration.
- Persistence is defined in the same way



The case of density and back to mode seeking

Given a probability density f:

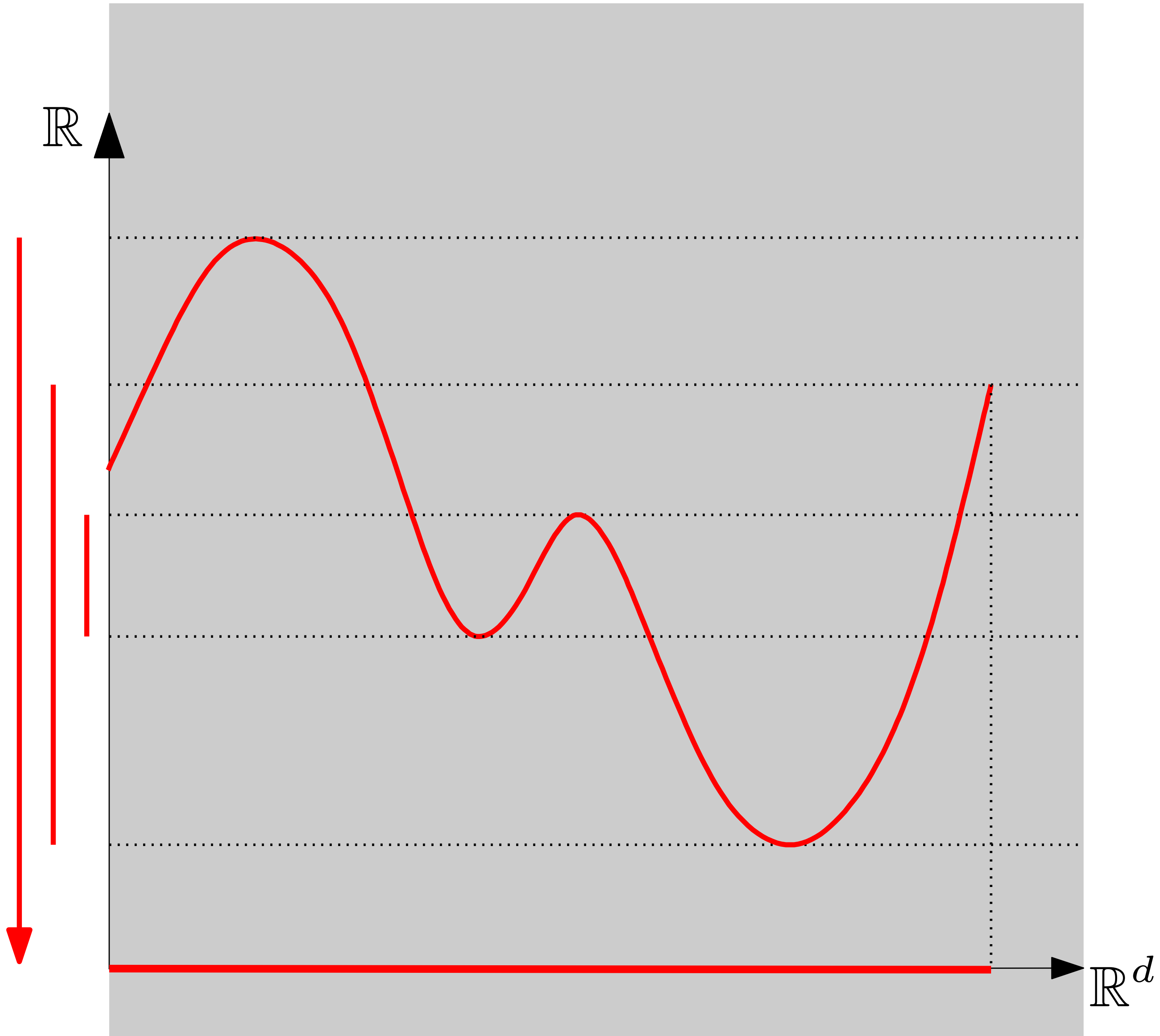
- Consider the superlevel-sets filtration $f^{-1}([t, +\infty))$ for t from $+\infty$ to $-\infty$, instead of the sublevel-sets filtration.
- Persistence is defined in the same way



The case of density and back to mode seeking

Given a probability density f:

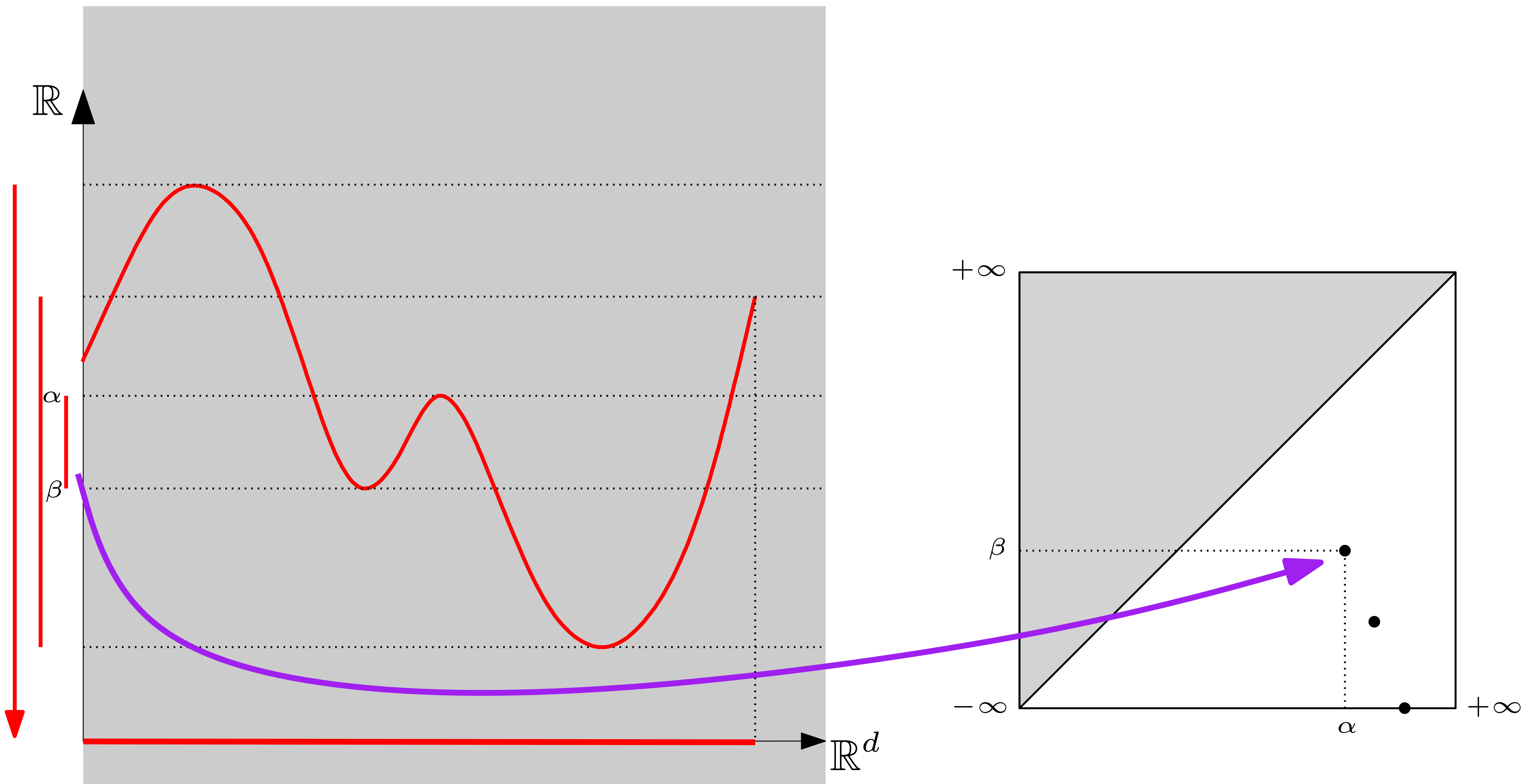
- Consider the superlevel-sets filtration $f^{-1}([t, +\infty))$ for t from $+\infty$ to $-\infty$, instead of the sublevel-sets filtration.
- Persistence is defined in the same way



The case of density and back to mode seeking

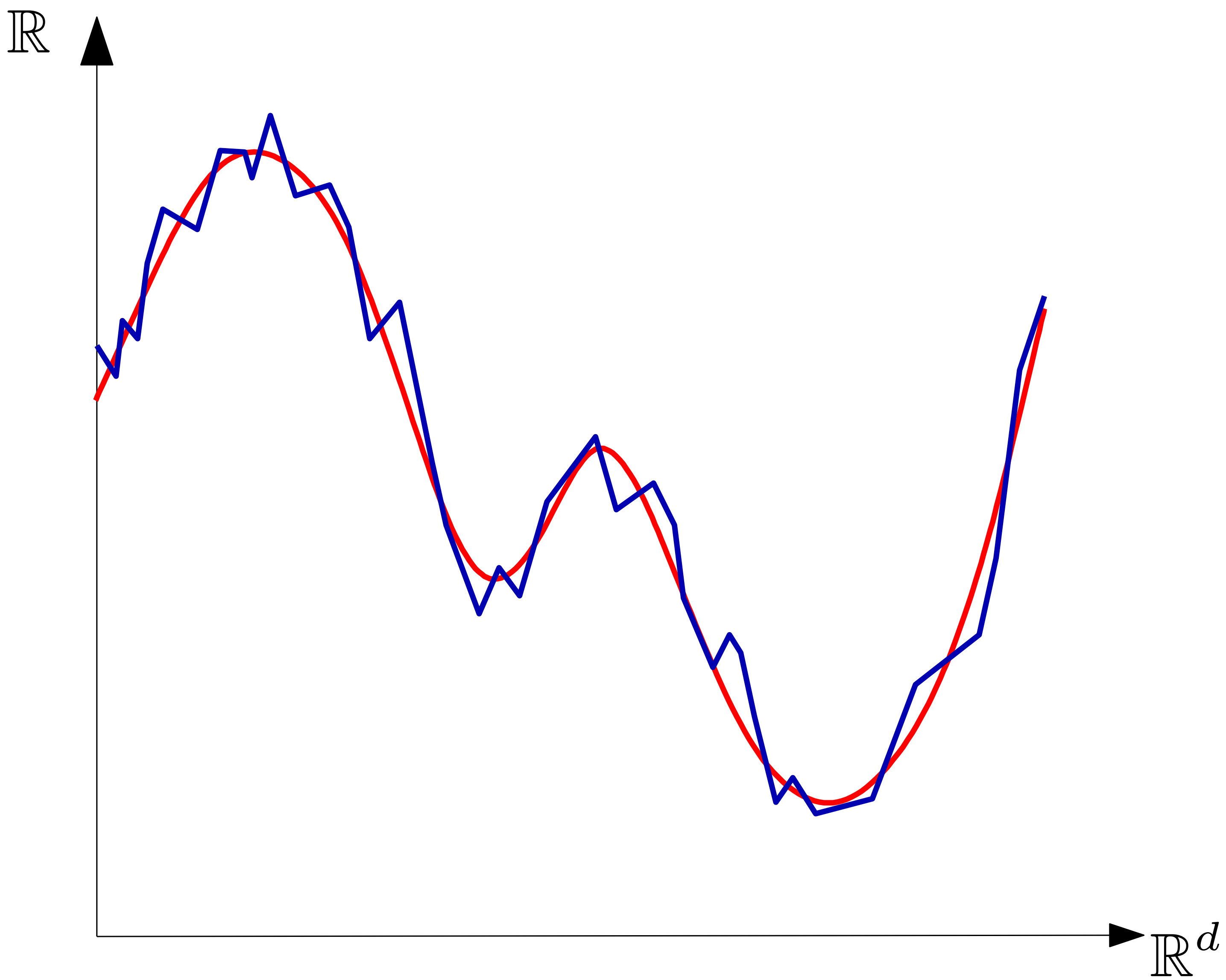
Given a probability density f:

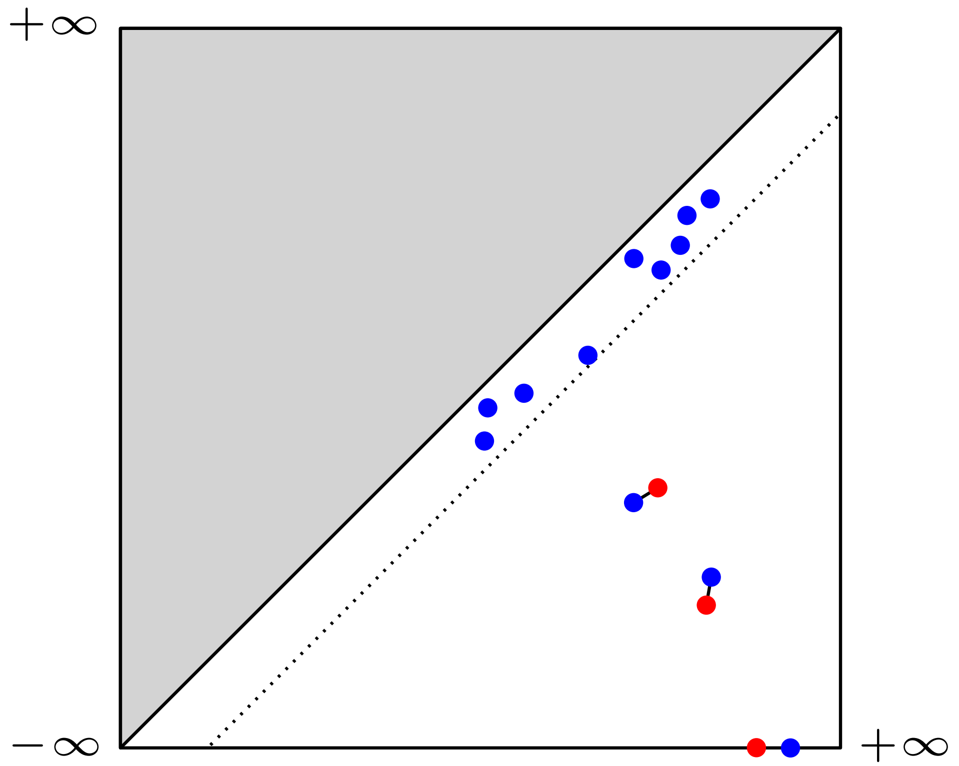
- Consider the superlevel-sets filtration $f^{-1}([t, +\infty))$ for t from $+\infty$ to $-\infty$, instead of the sublevel-sets filtration.
- Persistence is defined in the same way



The case of density and back to mode seeking

Given an estimator $\hat{f}$: Stability theorem $\Rightarrow d_B(D_f, D_{\hat{f}}) \leq \|f - \hat{f}\|_\infty$.

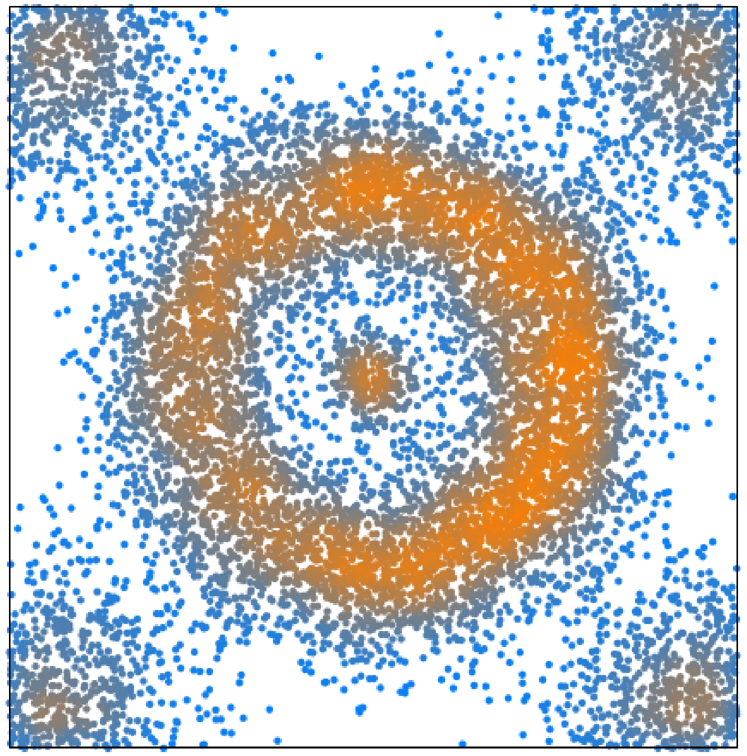




Persistence-based clustering

- Density estimator $\hat{f}$ defines an order on the point cloud

  (sort data points by **decreasing** estimated density values)



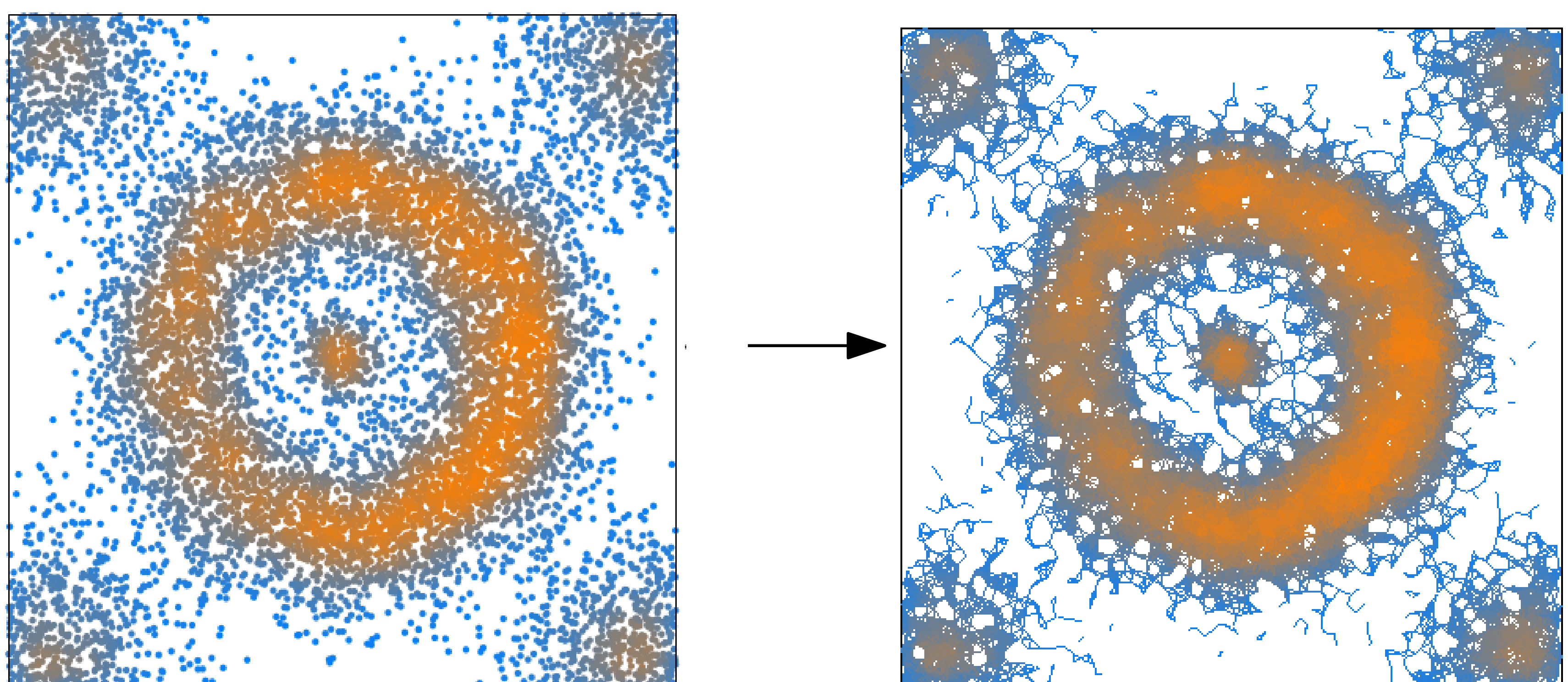
Persistence-based clustering

- Density estimator $\hat{f}$ defines an order on the point cloud

  (sort data points by **decreasing** estimated density values)

- Extend order to the graph edges $\rightarrow$ *upper-star filtration*

  $(\hat{f}([u,v]) = \min\{\hat{f}(u),\ \hat{f}(v)\})$



Persistence-based clustering

- Density estimator $\hat{f}$ defines an order on the point cloud

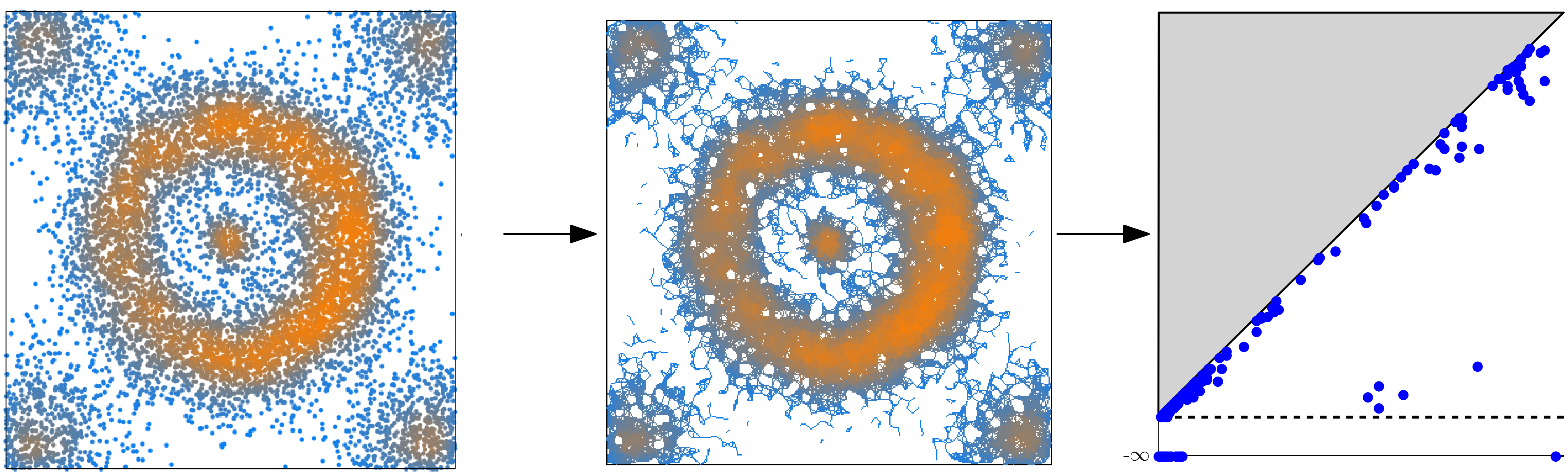
  (sort data points by **decreasing** estimated density values)

- Extend order to the graph edges $\rightarrow$ *upper-star filtration*

  $(\hat{f}([u,v]) = \min\{\hat{f}(u),\ \hat{f}(v)\})$

- Compute the 0-dimensional persistence diagram of this filtration

  (apply 0-dimensional persistence algorithm $\rightarrow$ union-find data structure)



Estimating the correct number of clusters

- Density estimator $\hat{f}$ defines an order on the point cloud

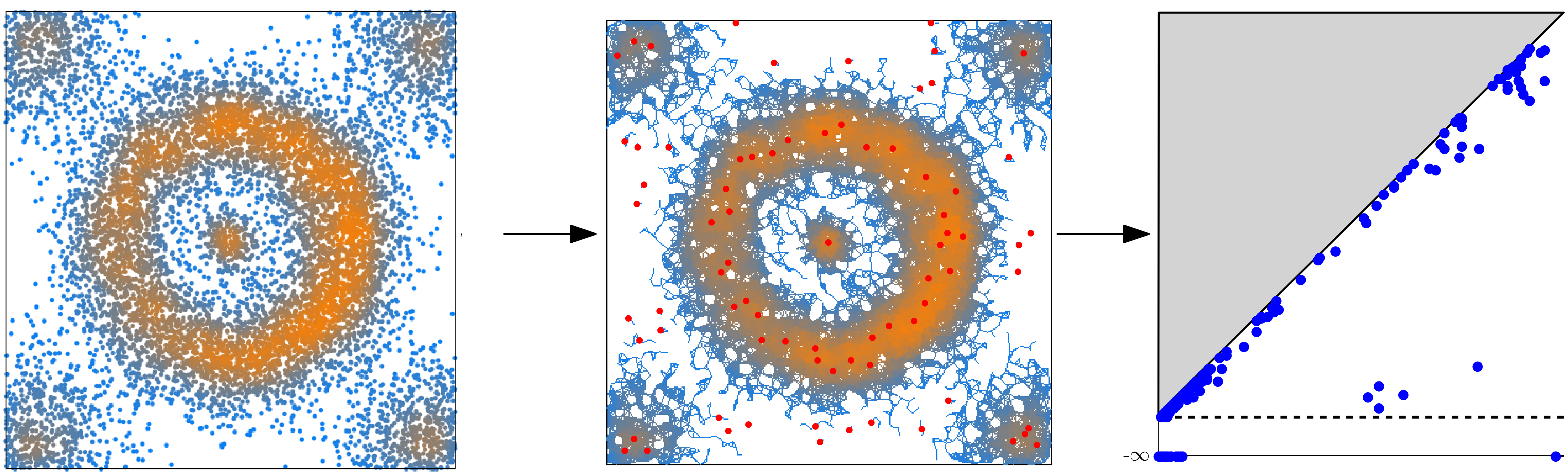
  (sort data points by **decreasing** estimated density values)

- Extend order to the graph edges $\rightarrow$ *upper-star filtration*

  $(\hat{f}([u,v]) = \min\{\hat{f}(u),\ \hat{f}(v)\})$

- Compute the 0-dimensional persistence diagram of this filtration

  (apply 0-dimensional persistence algorithm $\rightarrow$ union-find data structure)



Estimating the correct number of clusters

- Density estimator $\hat{f}$ defines an order on the point cloud

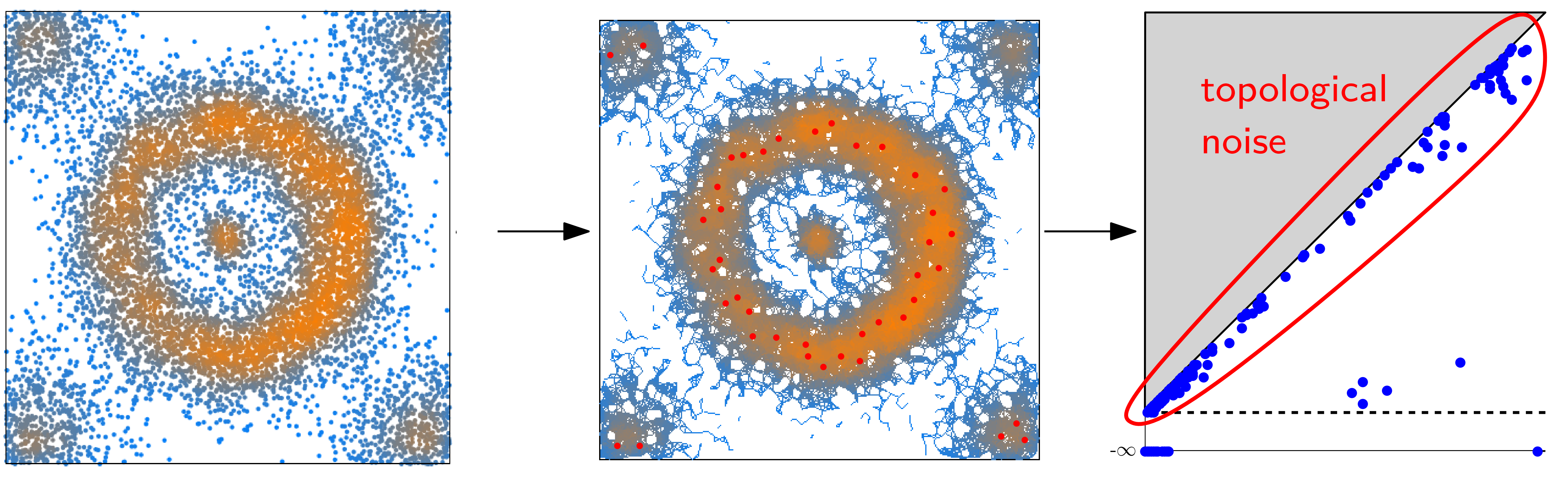
  (sort data points by **decreasing** estimated density values)

- Extend order to the graph edges $\rightarrow$ *upper-star filtration*

  $(\hat{f}([u,v]) = \min\{\hat{f}(u),\ \hat{f}(v)\})$

- Compute the 0-dimensional persistence diagram of this filtration

  (apply 0-dimensional persistence algorithm $\rightarrow$ union-find data structure)



Estimating the correct number of clusters

- Density estimator $\hat{f}$ defines an order on the point cloud

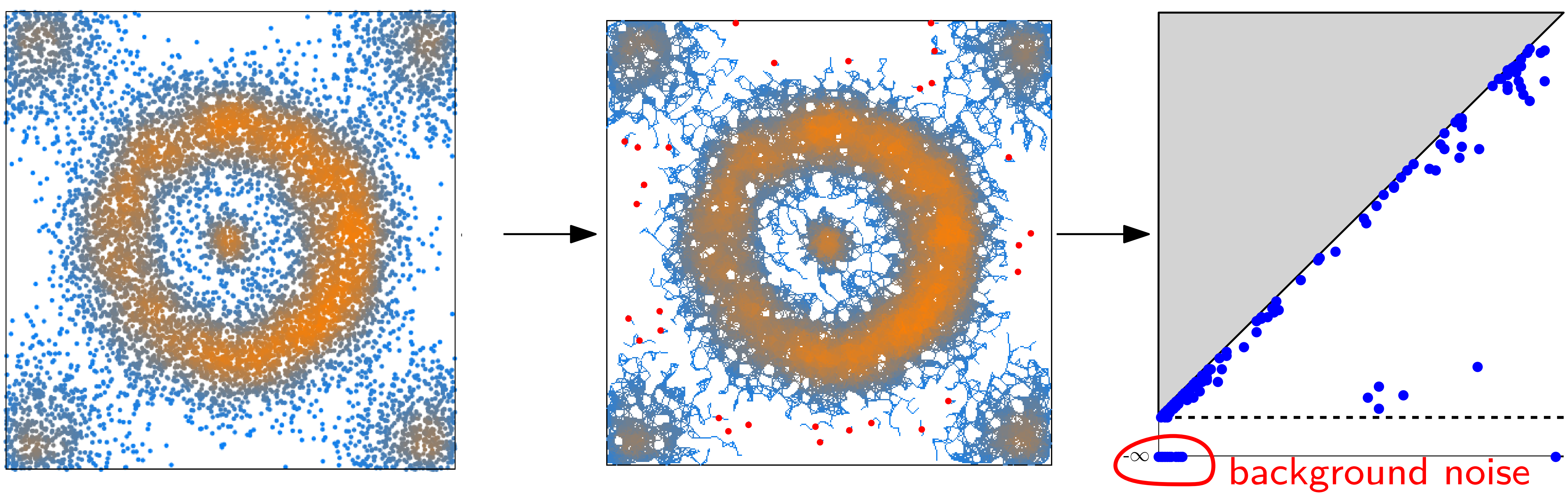
  (sort data points by **decreasing** estimated density values)

- Extend order to the graph edges $\rightarrow$ *upper-star filtration*

  $(\hat{f}([u,v]) = \min\{\hat{f}(u),\ \hat{f}(v)\})$

- Compute the 0-dimensional persistence diagram of this filtration

  (apply 0-dimensional persistence algorithm $\rightarrow$ union-find data structure)



Estimating the correct number of clusters

- Density estimator $\hat{f}$ defines an order on the point cloud

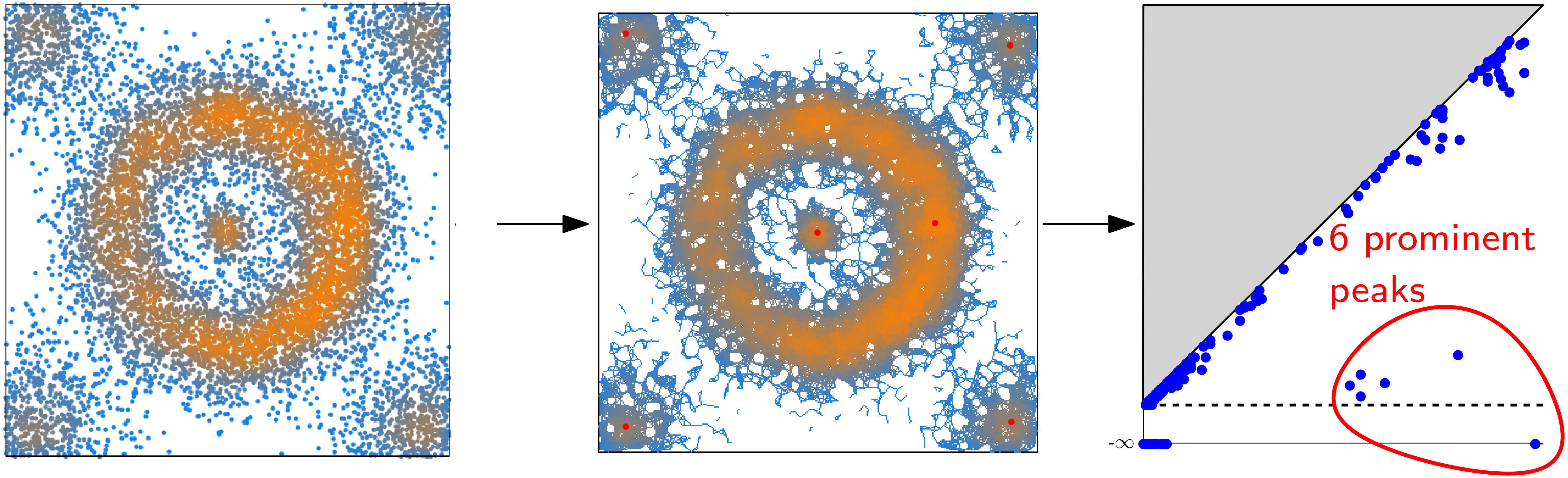
  (sort data points by **decreasing** estimated density values)

- Extend order to the graph edges $\rightarrow$ *upper-star filtration*

  $(\hat{f}([u,v]) = \min\{\hat{f}(u),\ \hat{f}(v)\})$

- Compute the 0-dimensional persistence diagram of this filtration

  (apply 0-dimensional persistence algorithm $\rightarrow$ union-find data structure)



Estimating the correct number of clusters

- Density estimator $\hat{f}$ defines an order on the point cloud

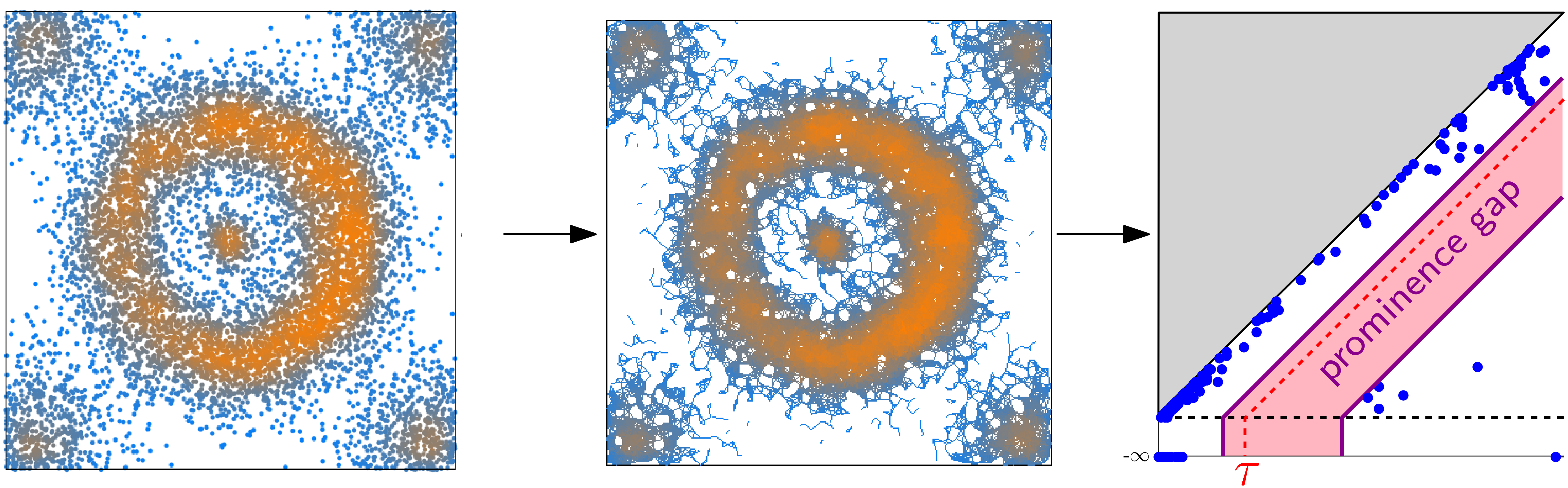
  (sort data points by **decreasing** estimated density values)

- Extend order to the graph edges $\rightarrow$ *upper-star filtration*

  $(\hat{f}([u,v]) = \min\{\hat{f}(u),\ \hat{f}(v)\})$

- Compute the 0-dimensional persistence diagram of this filtration

  (apply 0-dimensional persistence algorithm $\rightarrow$ union-find data structure)



Estimating the correct number of clusters

Hypotheses:

- $f : \mathbb{R}^d \to \mathbb{R}$ a c-Lipschitz probability density function,
- $P \subset \mathbb{R}^d$ a finite set of n points sampled i.i.d. according to f,
- $\hat{f} : P \to \mathbb{R}$ a density estimator such that $\eta := \max_{p \in P} |\hat{f}(p) - f(p)| < \Pi/5$,
- $G = (P, E)$ the δ-neighborhood graph for some positive $\delta < \frac{\Pi - 5\eta}{5c}$.

Note: Π is the prominence of the least prominent peak of f

Estimating the correct number of clusters

Hypotheses:

- $f : \mathbb{R}^d \to \mathbb{R}$ a c-Lipschitz probability density function,
- $P \subset \mathbb{R}^d$ a finite set of n points sampled i.i.d. according to f,
- $\hat{f} : P \to \mathbb{R}$ a density estimator such that $\eta := \max_{p \in P} |\hat{f}(p) - f(p)| < \Pi/5$,
- $G = (P, E)$ the δ-neighborhood graph for some positive $\delta < \frac{\Pi - 5\eta}{5c}$.

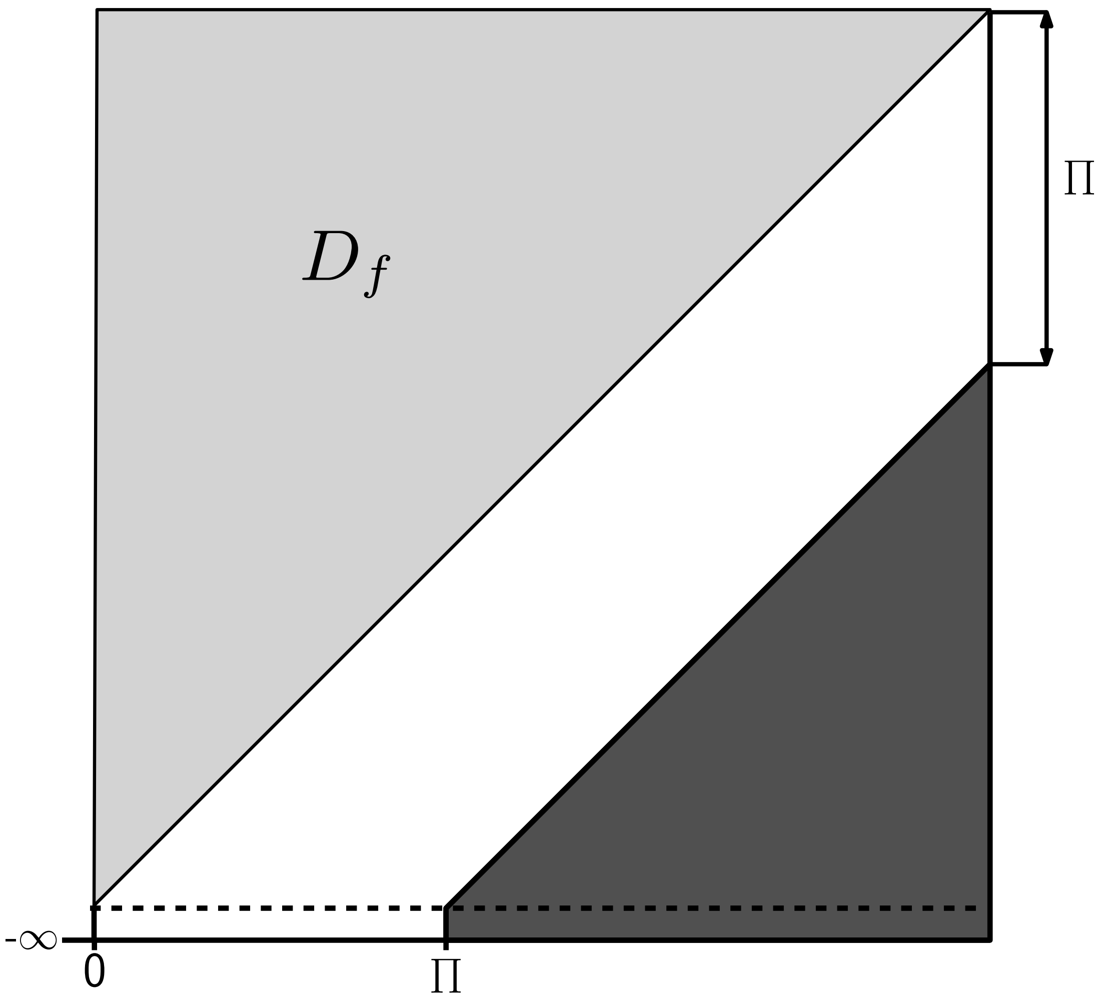
Note: Π is the prominence of the least prominent peak of f

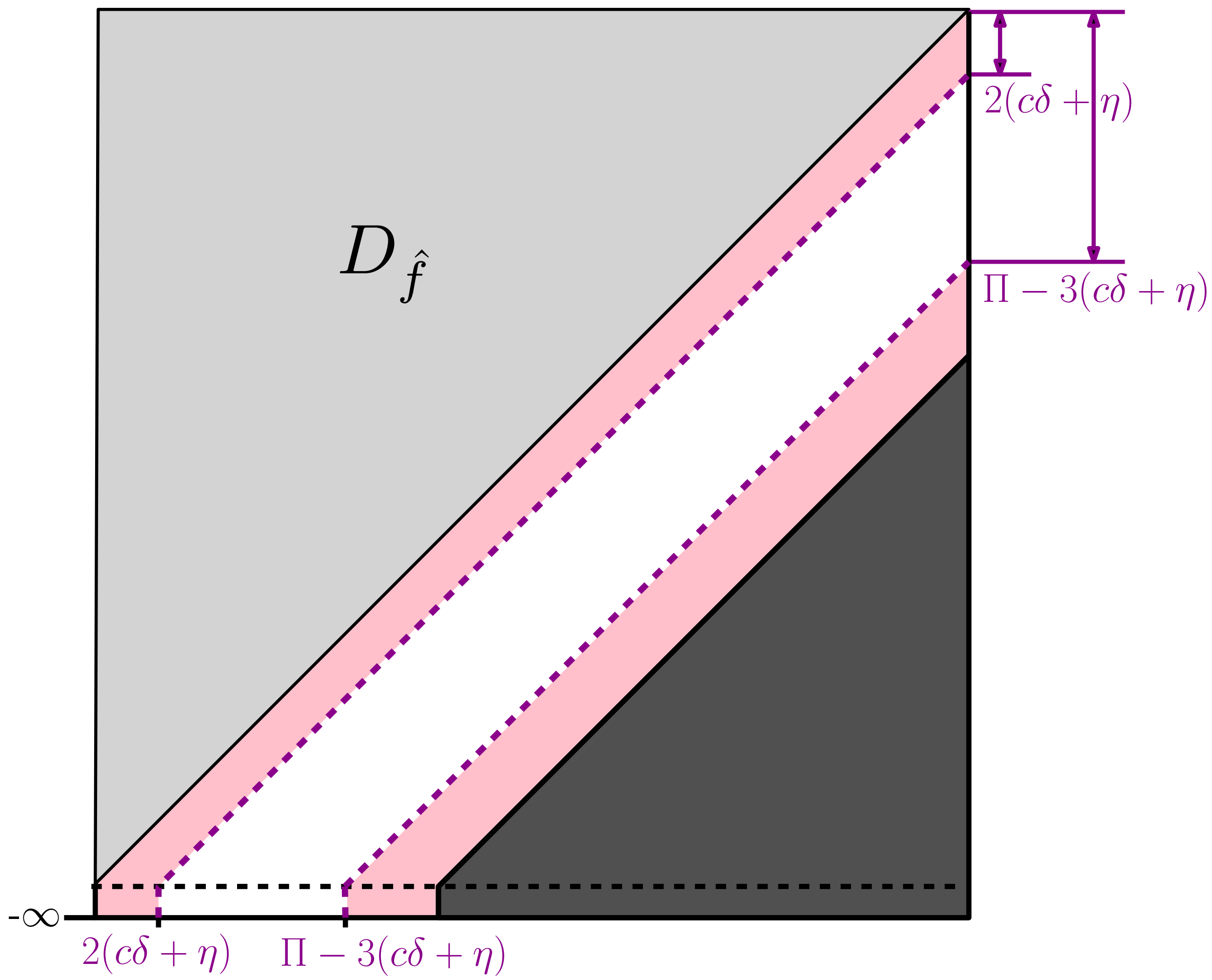
Conclusion:

For any choice of τ such that $2(c\delta + \eta) < \tau < \Pi - 3(c\delta + \eta)$, the number of clusters computed by the algorithm is equal to the number of peaks of f with probability at least $1 - e^{-\Omega(n)}$.

(the Ω notation hides factors depending on c, δ)

Estimating the correct number of clusters



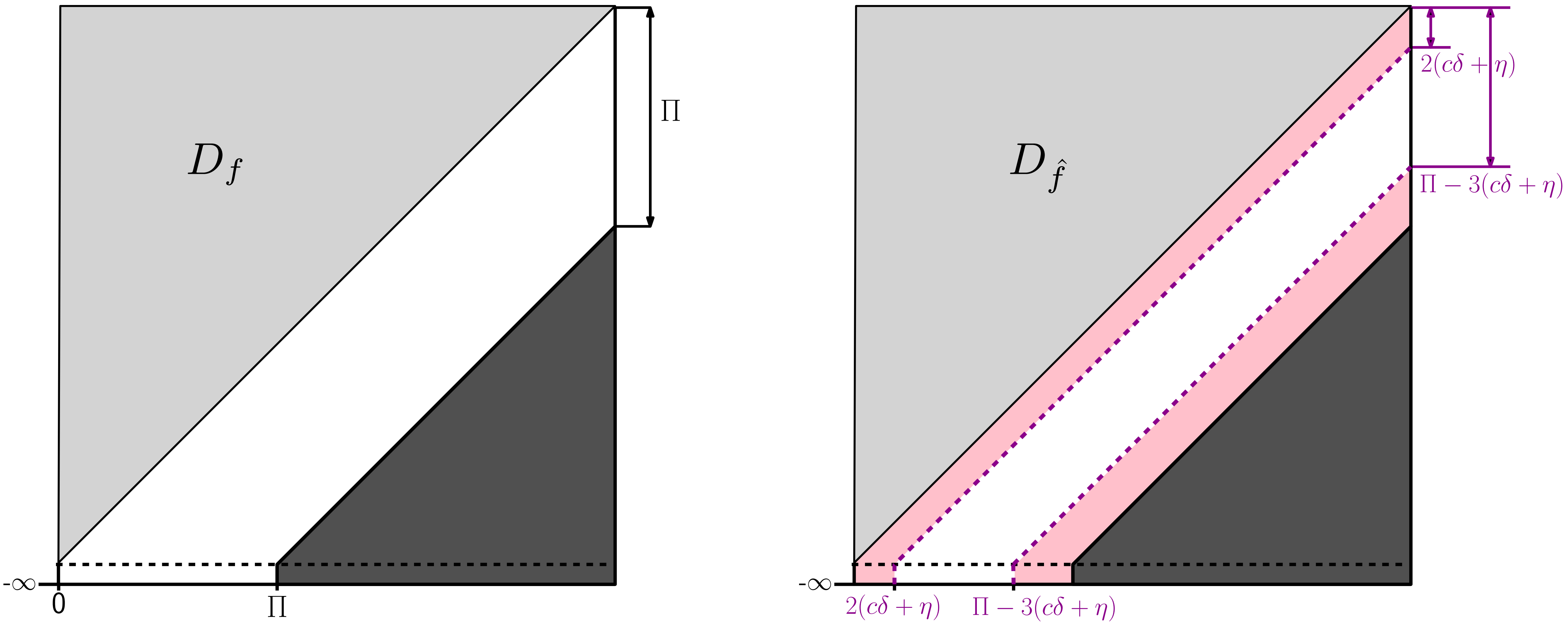


Conclusion:

For any choice of τ such that $2(c\delta + \eta) < \tau < \Pi - 3(c\delta + \eta)$, the number of clusters computed by the algorithm is equal to the number of peaks of f with probability at least $1 - e^{-\Omega(n)}$.

(the Ω notation hides factors depending on c, δ)

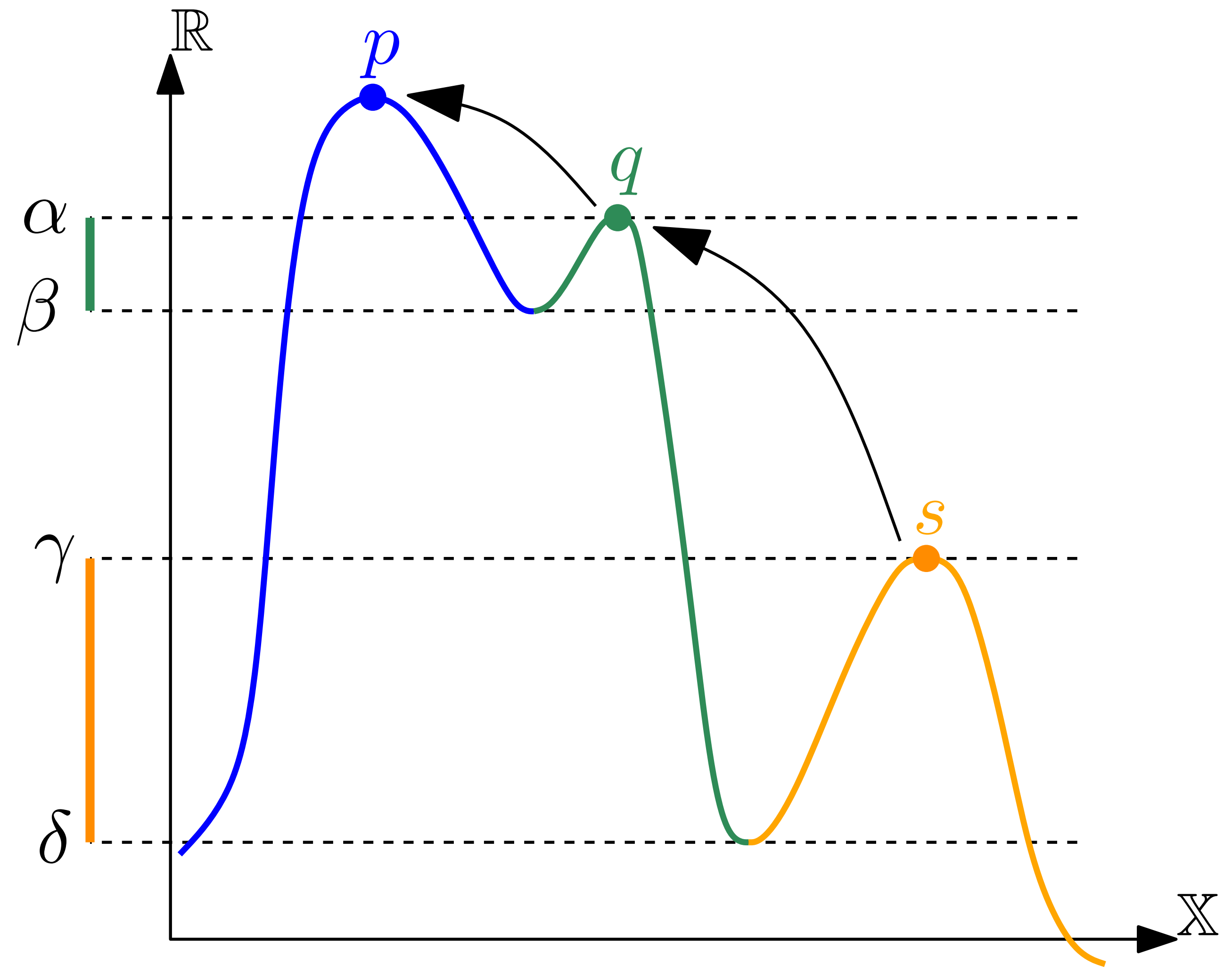
Estimating the correct number of clusters



Proof's main ingredient: stability theorem for persistence diagrams

Merging clusters

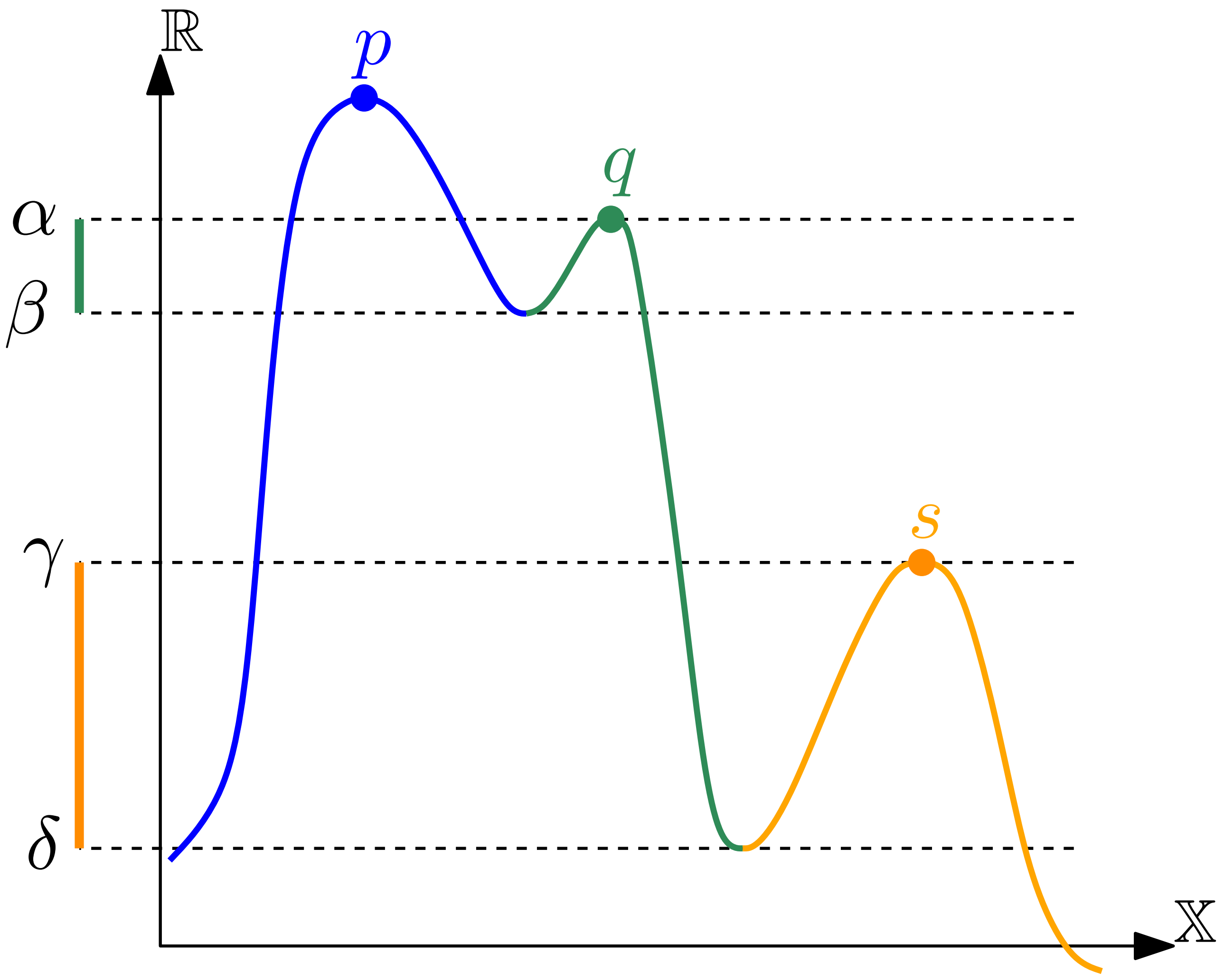
- degree-0 persistence algo. builds a hierarchy of the peaks of $\hat{f}$ (merge tree)
- merge clusters according to the hierarchy (merge each cluster into its parent)



Merging clusters

- degree-0 persistence algo. builds a hierarchy of the peaks of $\hat{f}$ (merge tree)
- merge clusters according to the hierarchy (merge each cluster into its parent)
- given a fixed threshold $\tau \geq 0$, only merge those clusters of prominence $< \tau$

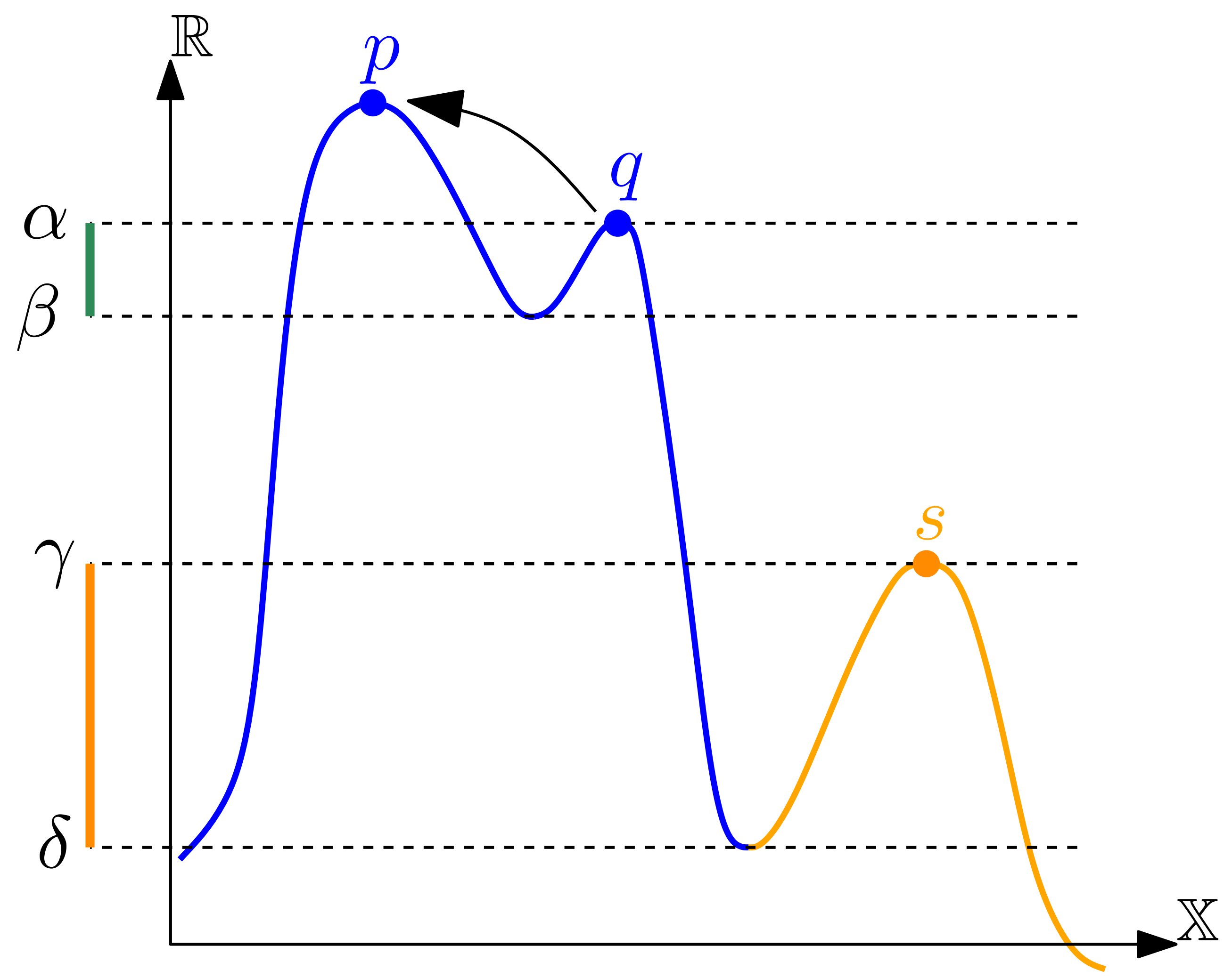
$$0 \leq \tau \leq \alpha - \beta$$



Merging clusters

- degree-0 persistence algo. builds a hierarchy of the peaks of $\hat{f}$ (merge tree)
- merge clusters according to the hierarchy (merge each cluster into its parent)
- given a fixed threshold $\tau \geq 0$, only merge those clusters of prominence $< \tau$

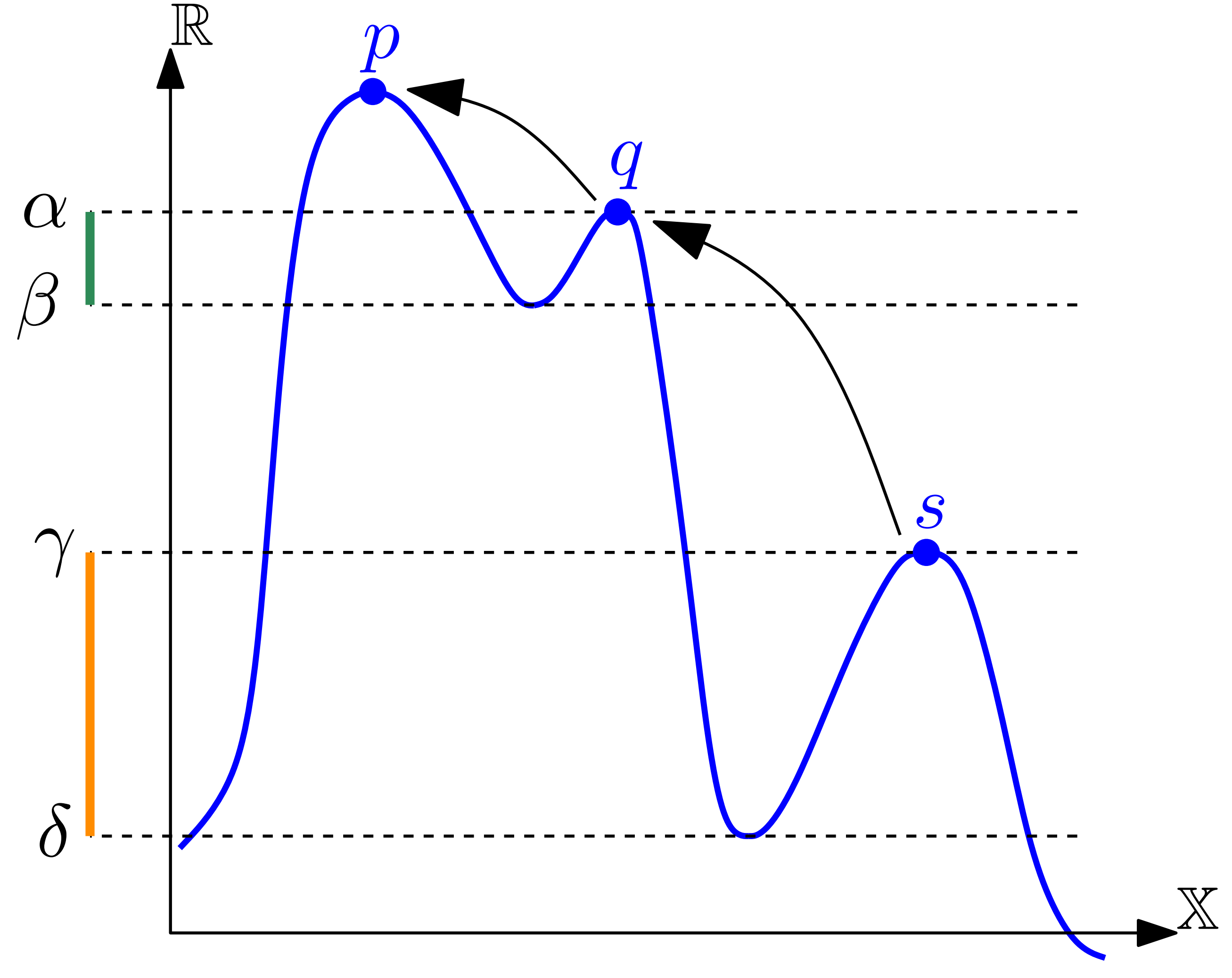
$$\alpha - \beta < \tau \leq \gamma - \delta$$



Merging clusters

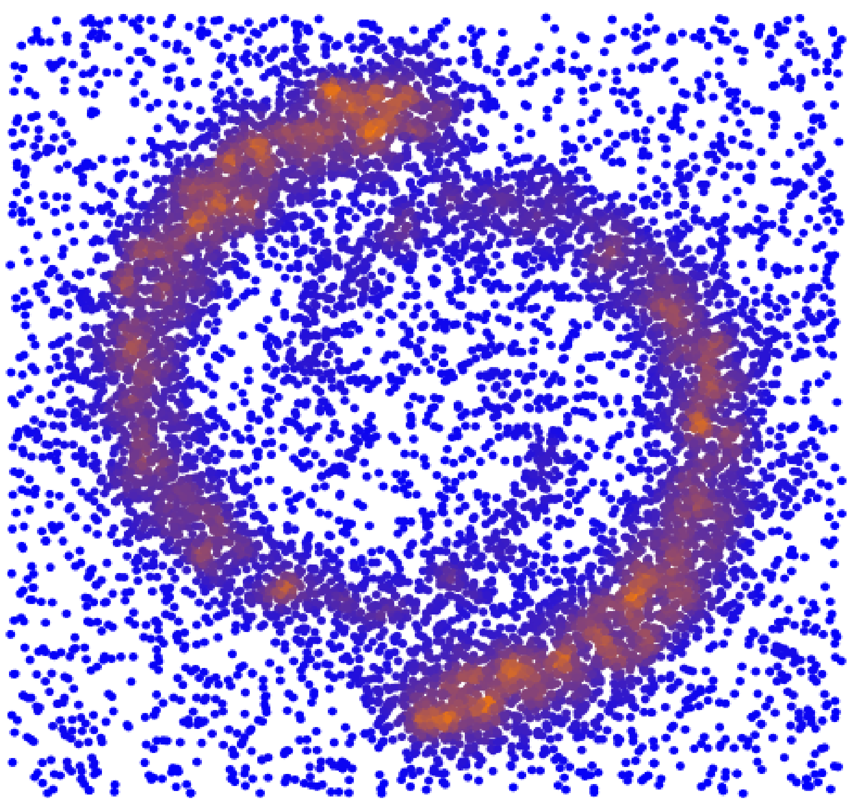
- degree-0 persistence algo. builds a hierarchy of the peaks of $\hat{f}$ (merge tree)
- merge clusters according to the hierarchy (merge each cluster into its parent)
- given a fixed threshold $\tau \geq 0$, only merge those clusters of prominence $< \tau$

$$\gamma - \delta < \tau \leq +\infty$$

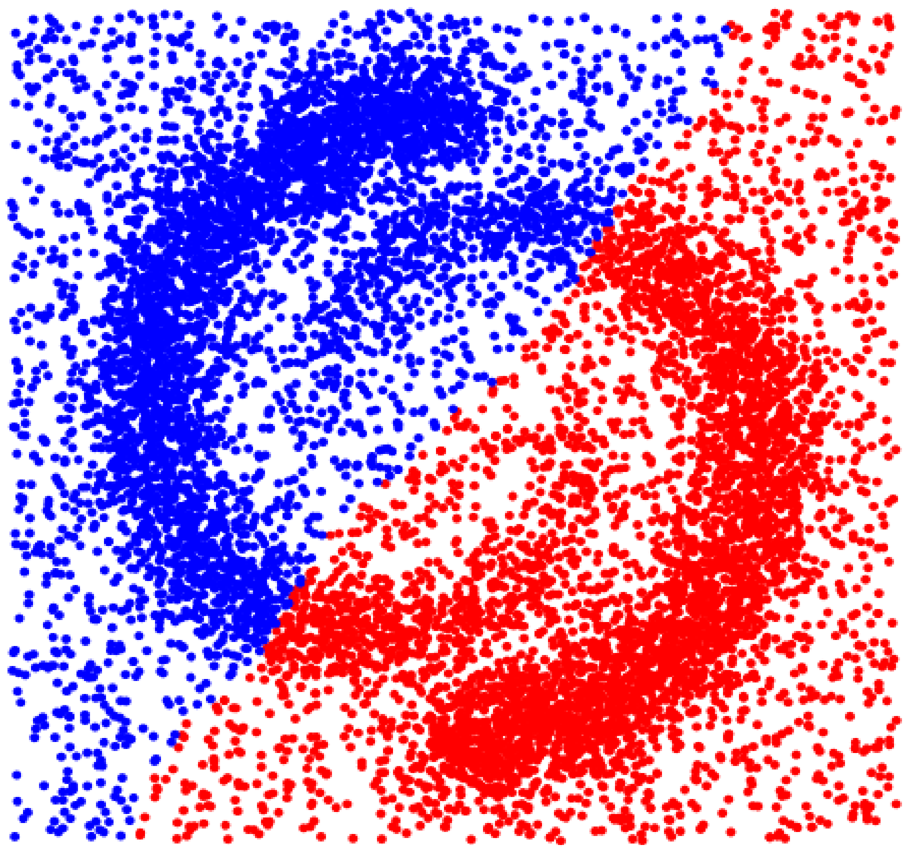


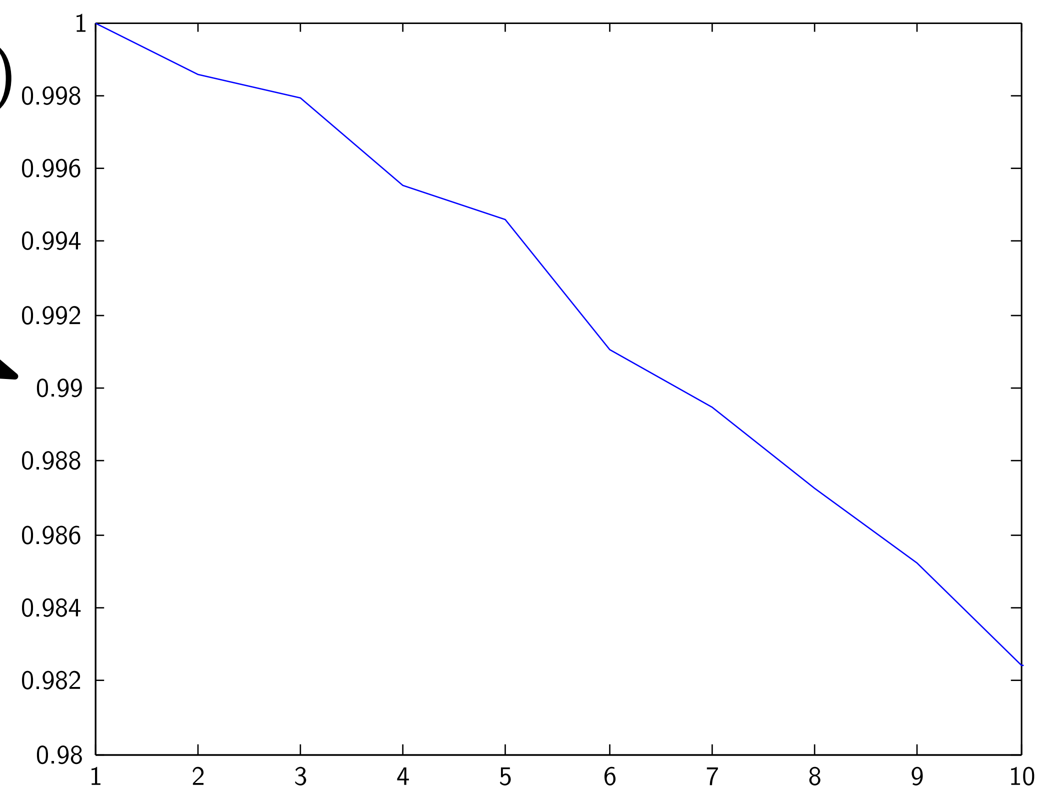
Experimental results

Synthetic Data



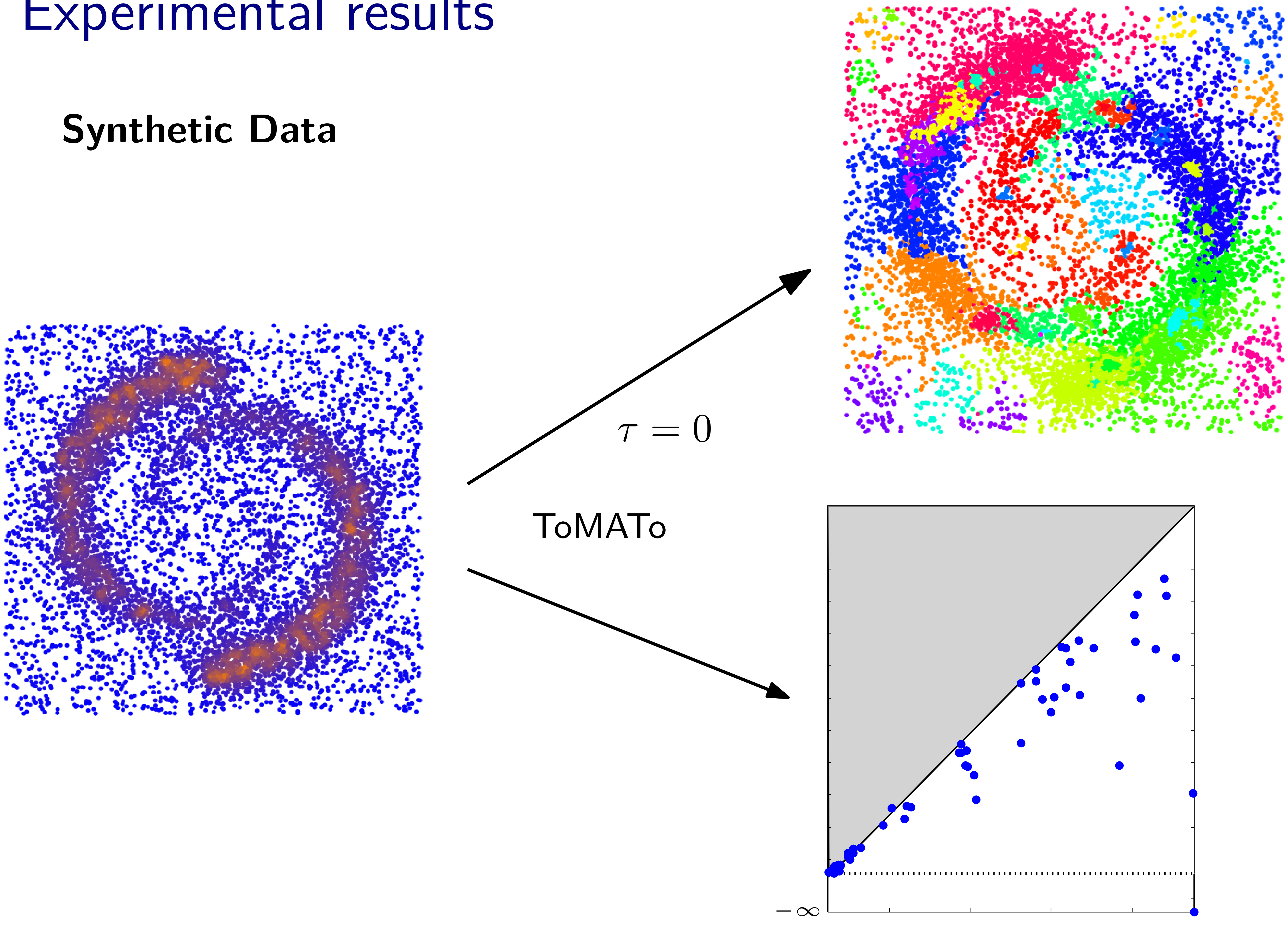
Spectral clustering
(k-means in eigenspace)





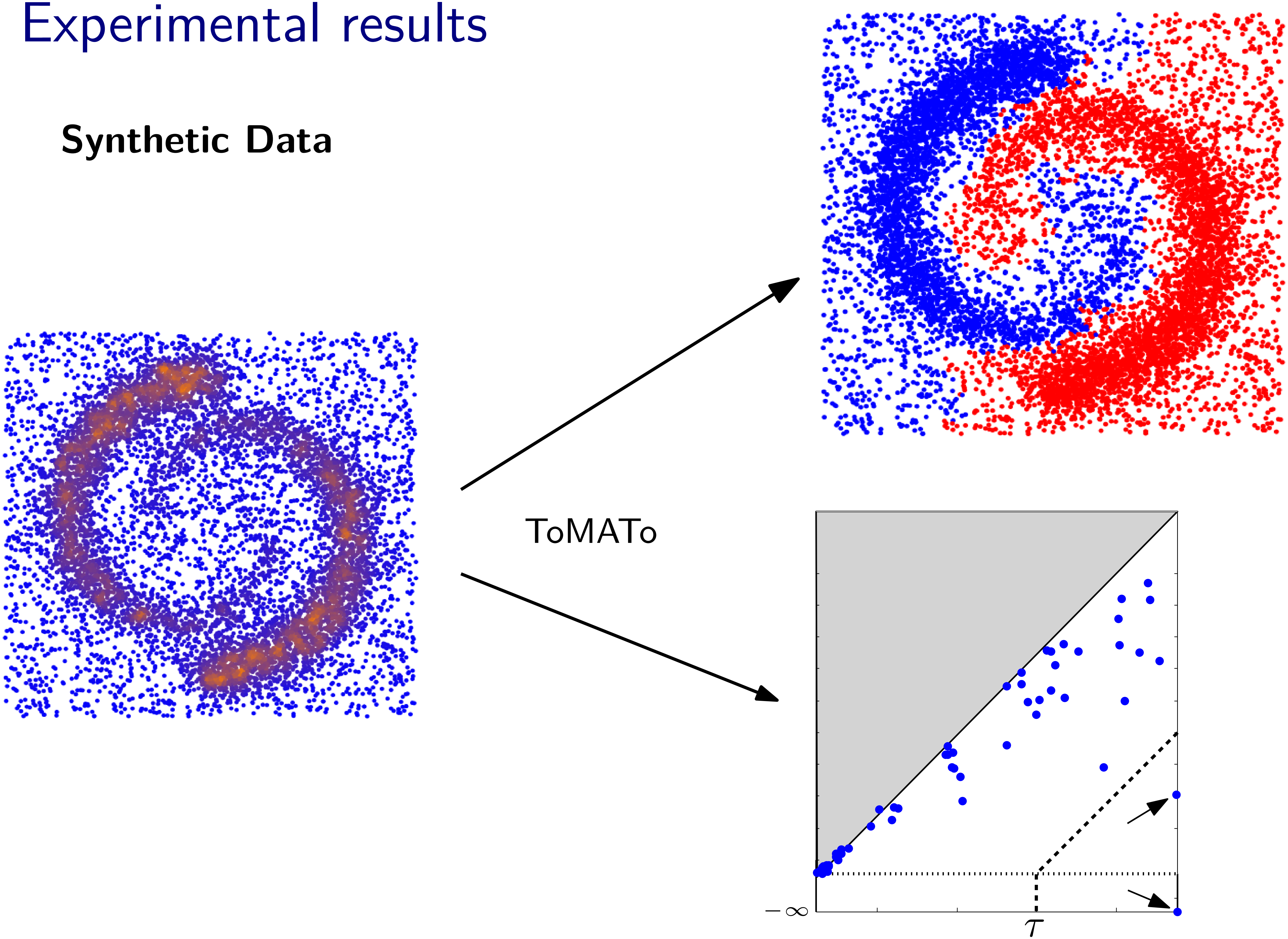
Experimental results

Synthetic Data



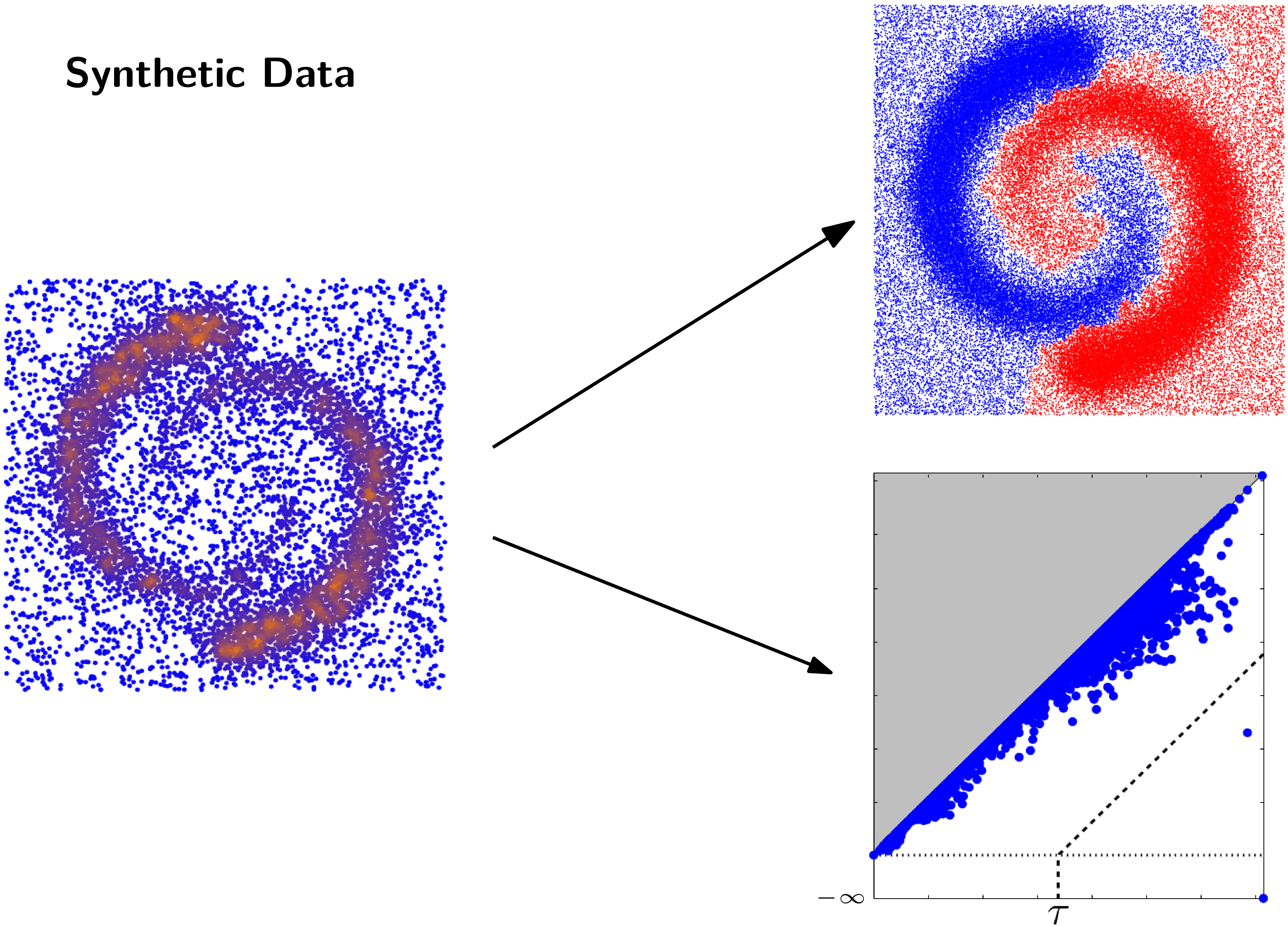
Experimental results

Synthetic Data



Experimental results

Synthetic Data

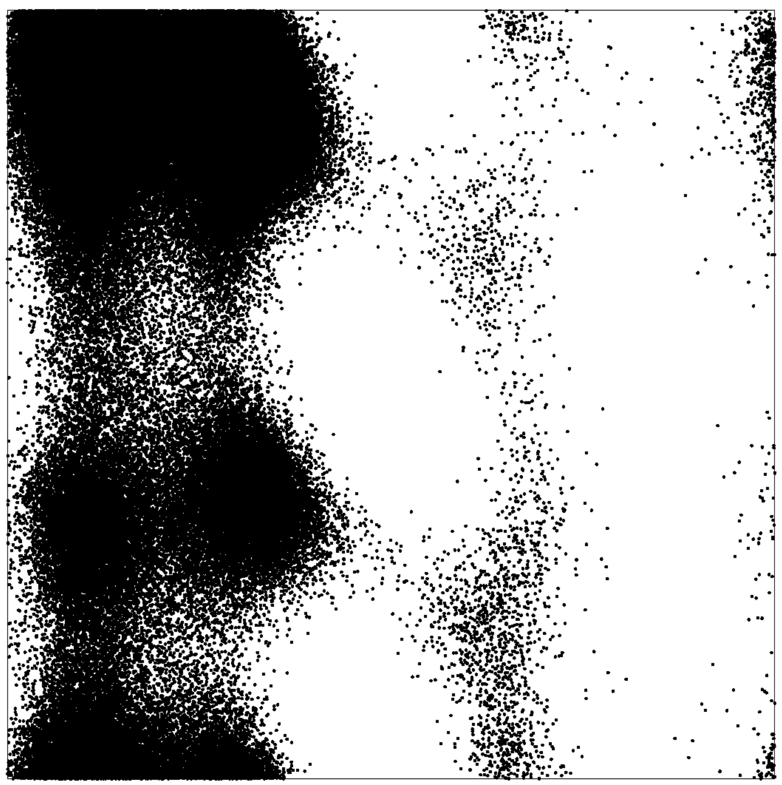


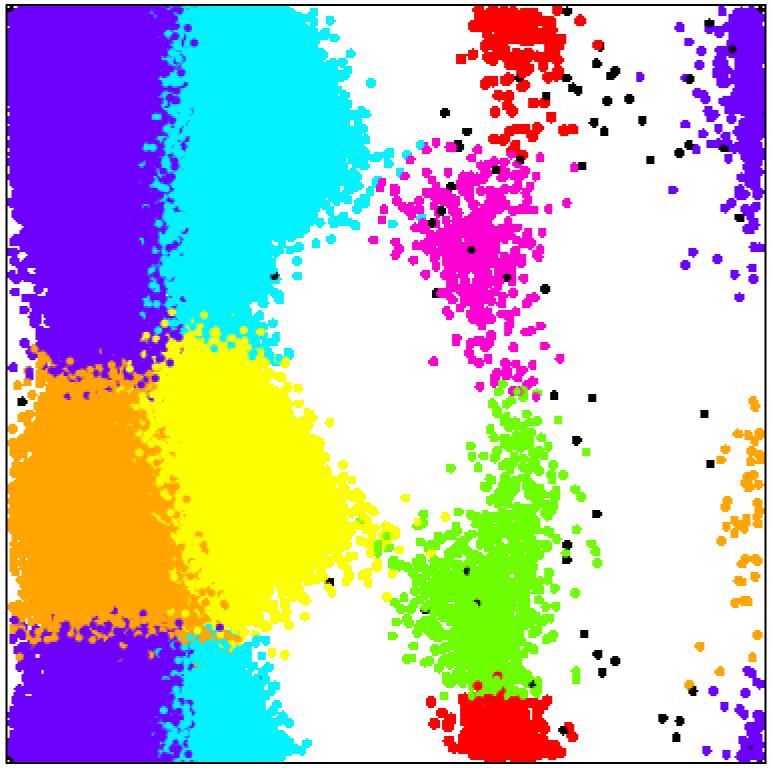
Experimental results

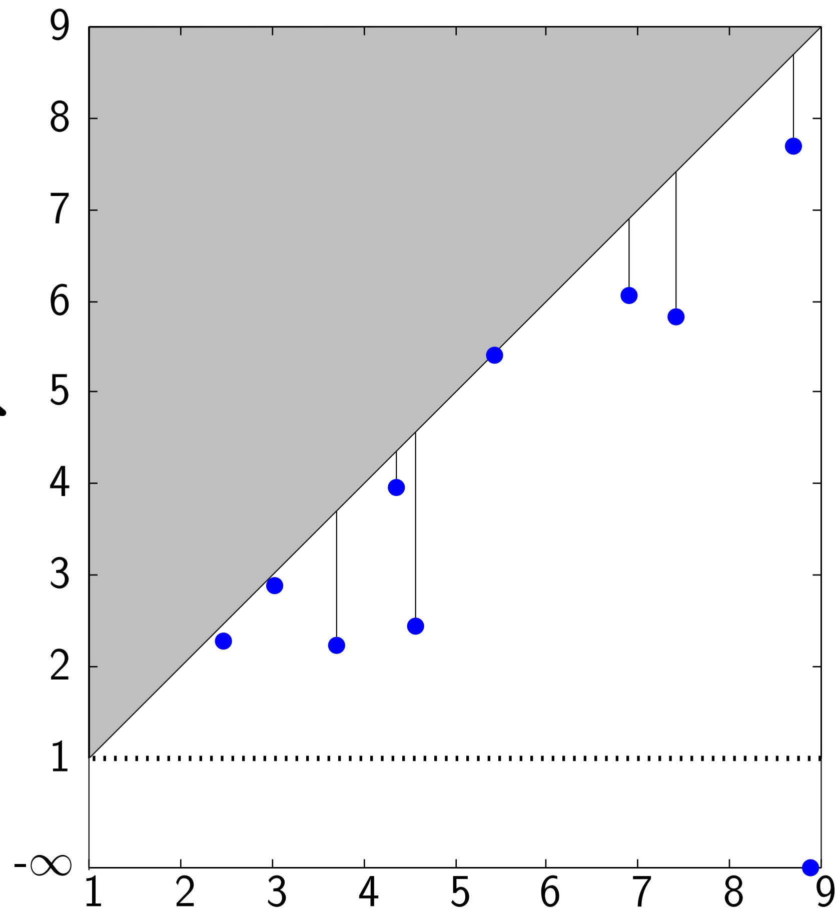
Biological Data

Alanine-Dipeptide conformations ($\mathbb{R}^{21}$)

RMSD distance (non-Euclidean)







Common belief: 6 metastable states

PD shows anywhere between 4 and 7 clusters

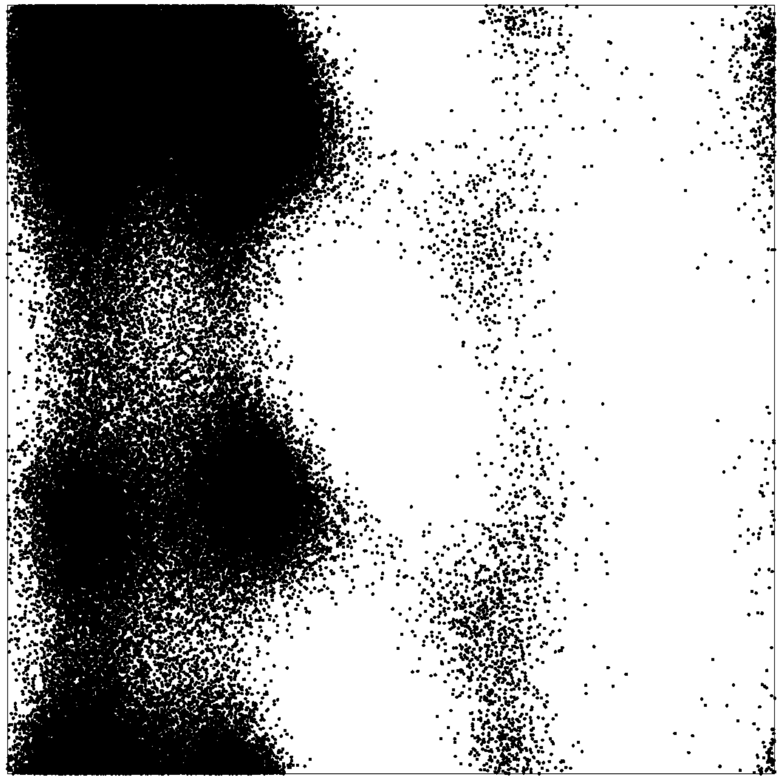
Experimental results

[*Topological methods for exploring low-density states in biomolecular folding pathways*, Yao, Sun, Huang, Bowman, Singh, Lesnick, Guibas, Pande, Carlsson, J. Chem. Phys., 2009]

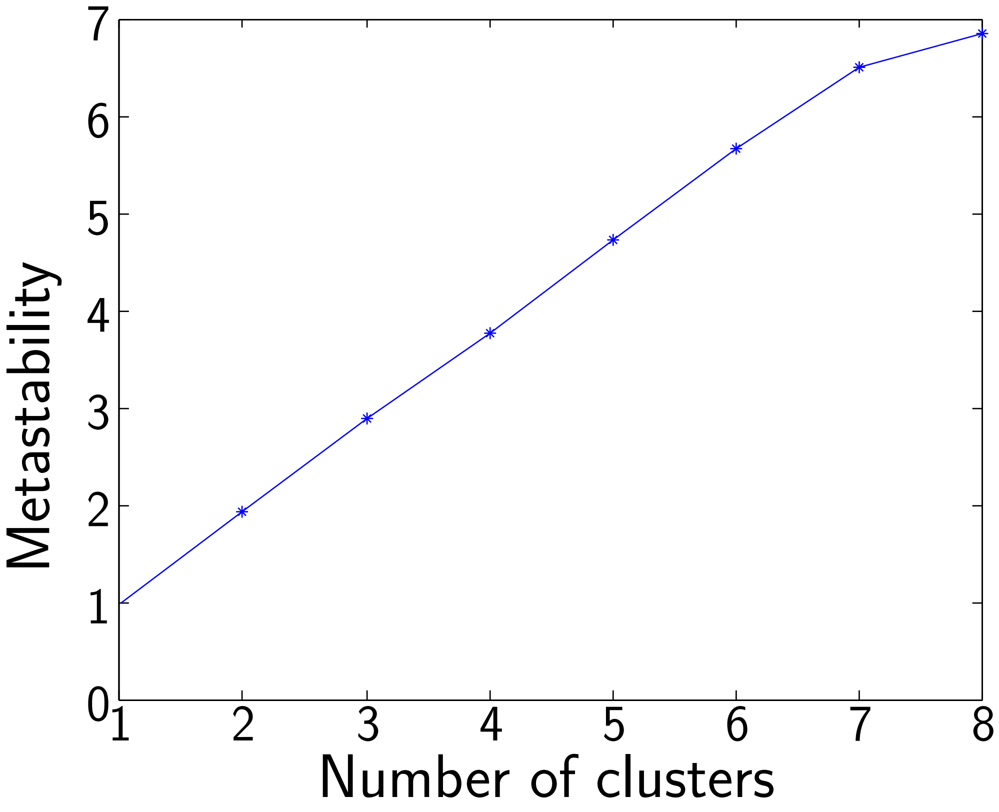
Biological Data

Alanine-Dipeptide conformations ($\mathbb{R}^{21}$)

RMSD distance (non-Euclidean)



Rank	Prominence	Metastability
1	$+\infty$	0.99982
2	3827	1.91865
3	1334	2.8813
4	557	3.76217
5	85	4.73838
6	32	5.65553
7	26	6.50757
8	7.2	6.8193
9	3.0	-
10	2.2	-



Common belief: 6 metastable states

PD shows anywhere between 4 and 7 clusters

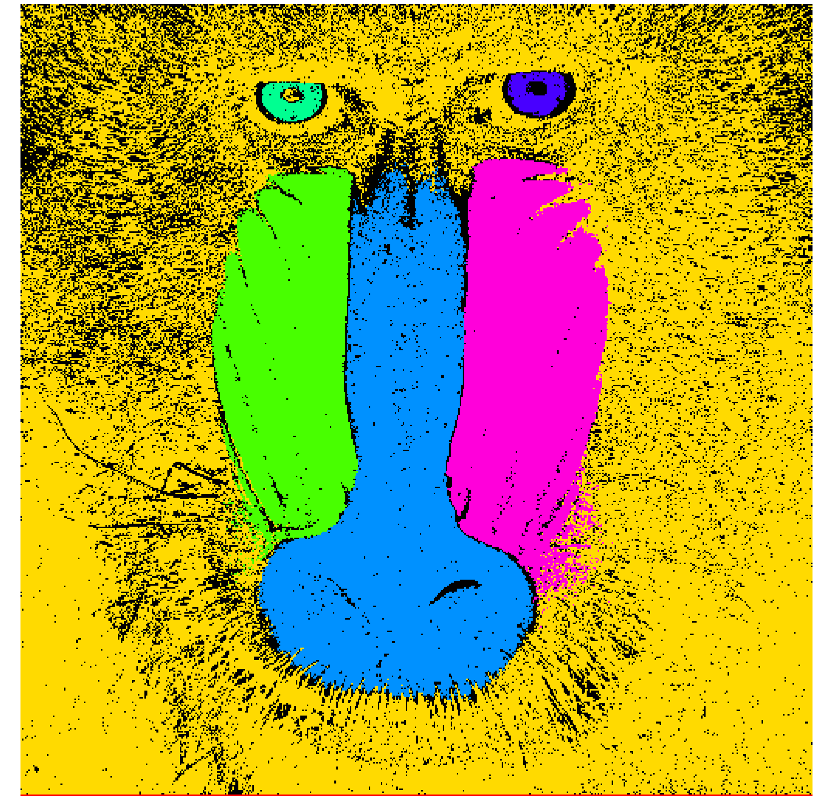
Measures of metastability confirm this insight

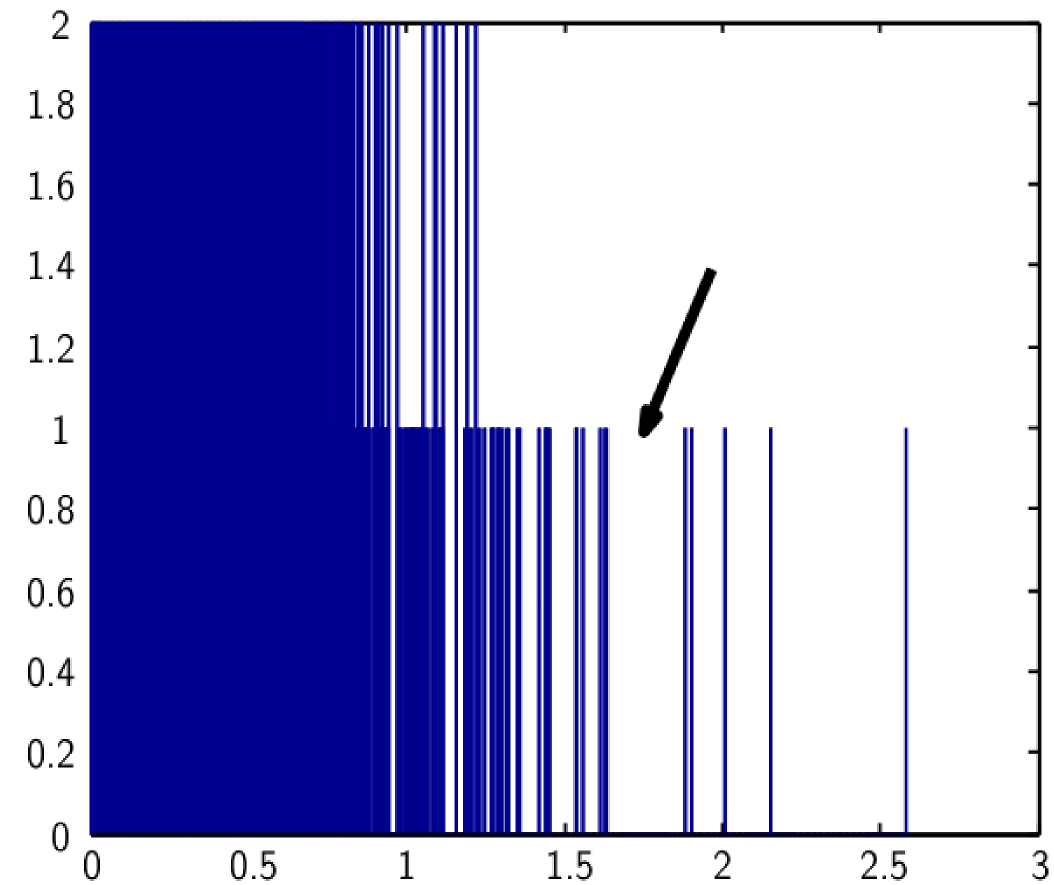
Experimental results

Image Segmentation

Density is estimated in 3D color space (Luv)

Neighborhood graph is built in image domain





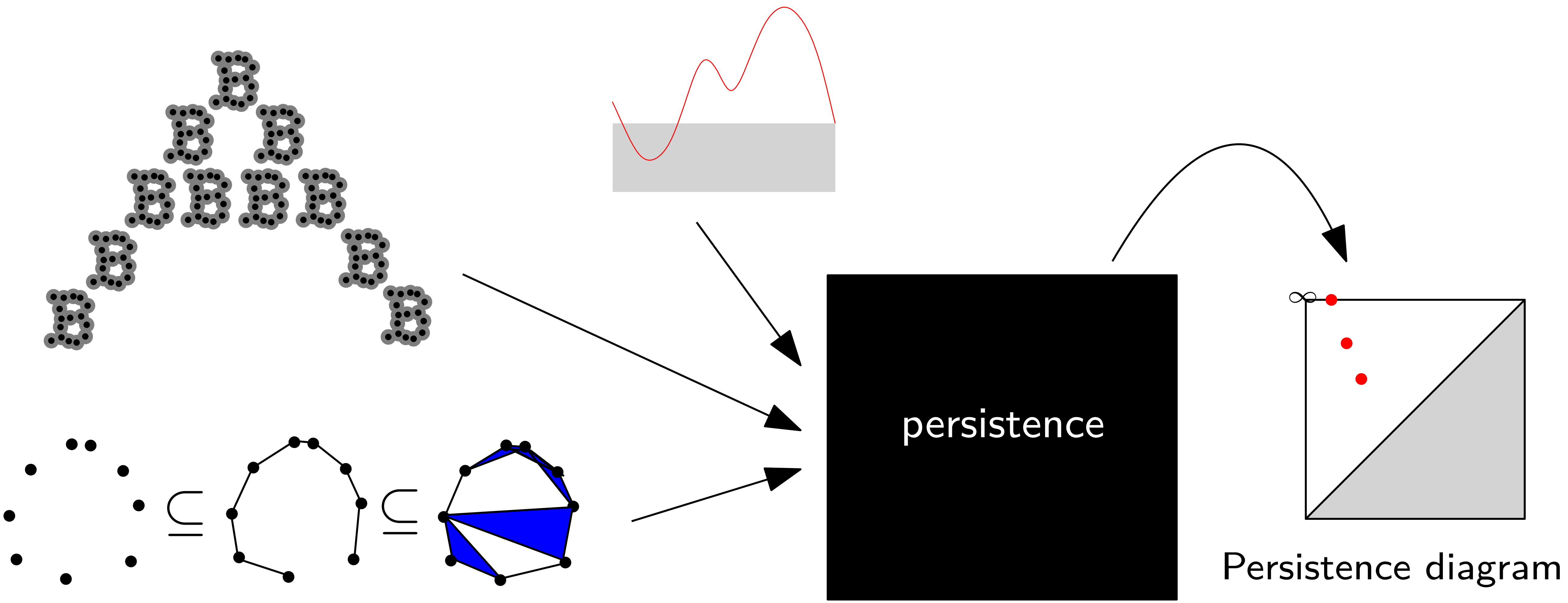
Distribution of prominences does not usually show a clear unique gap

Still, relationship between choice of τ and number of obtained clusters remains explicit

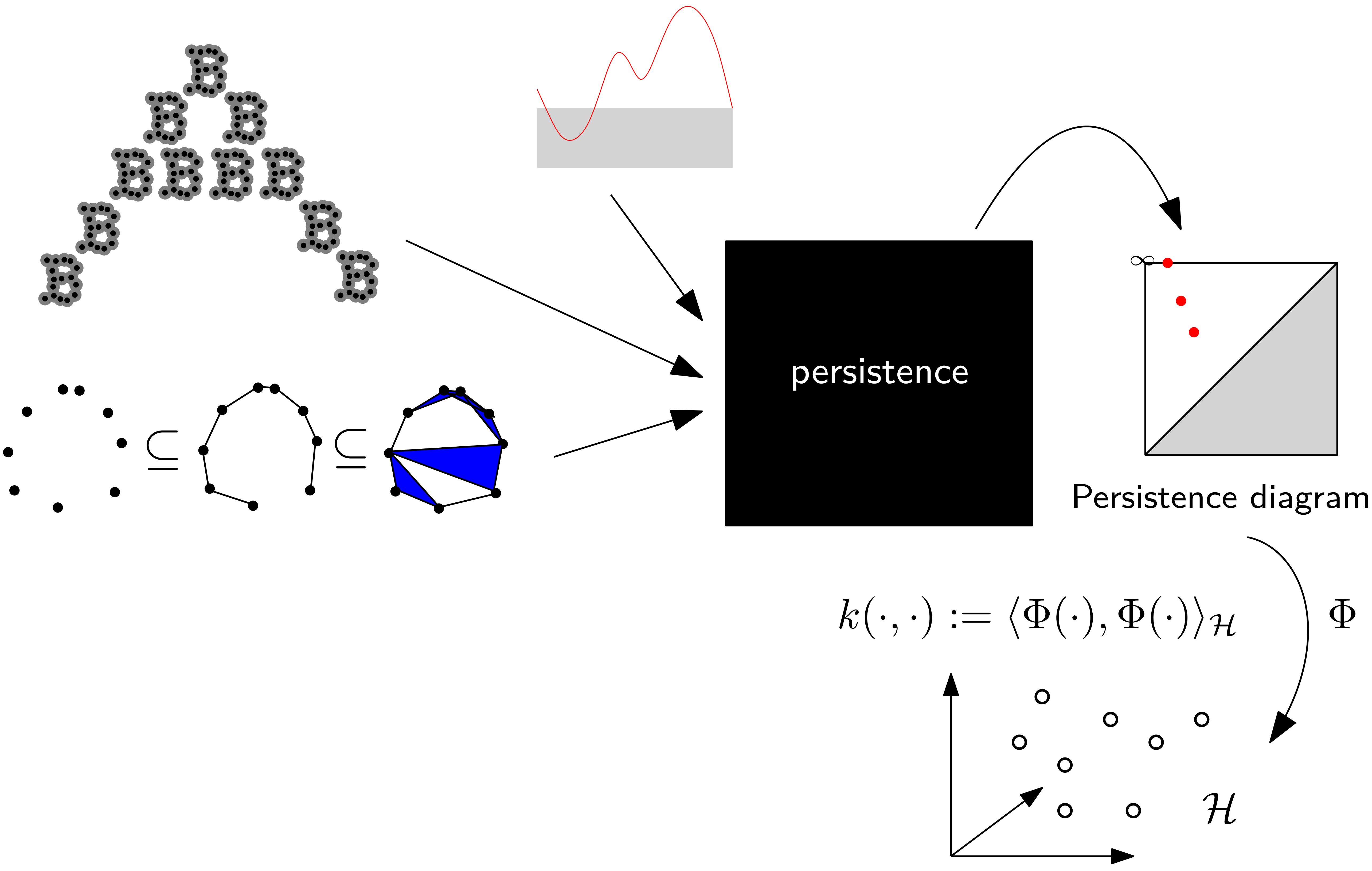
Persistence diagrams, Kernel Methods and Deep Learning

Persistence diagrams and machine learning

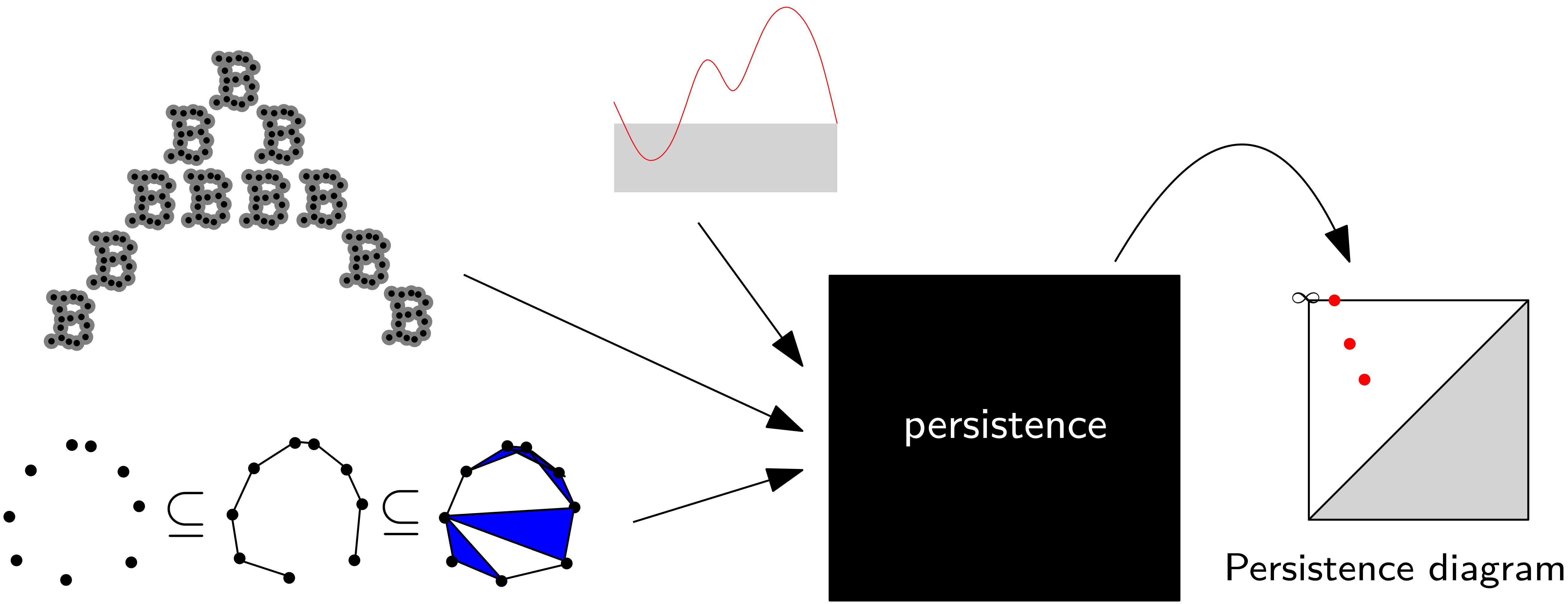
Persistence diagrams and machine learning

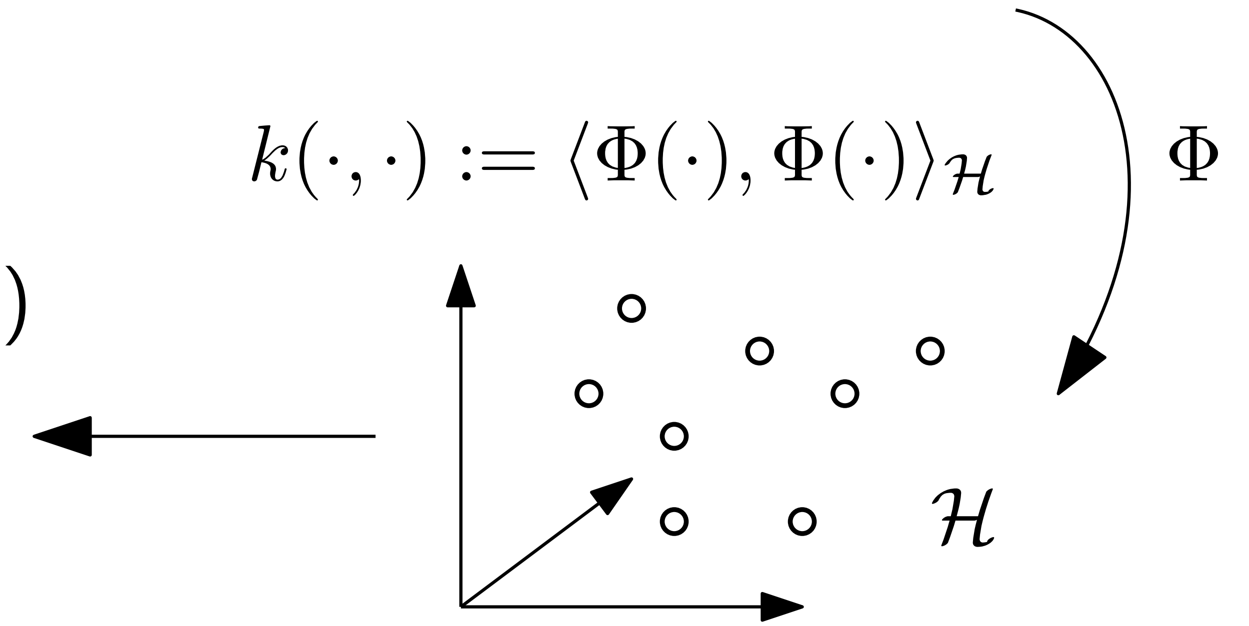


Persistence diagrams and machine learning



Persistence diagrams and machine learning

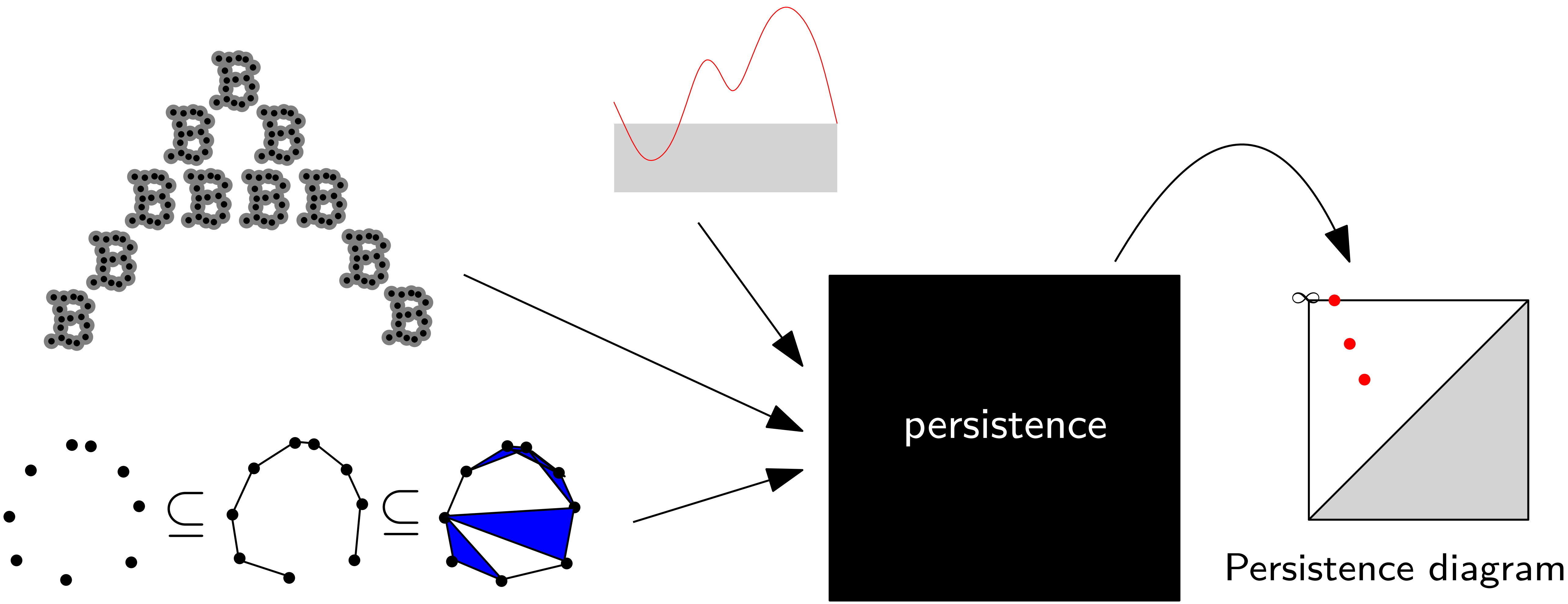




- Classifier (RF, SVM, NN etc.)
- Dim. red. (PCA, MDS, UMAP, t-SNE)
- Clustering (DBSCAN, K-means, etc.)

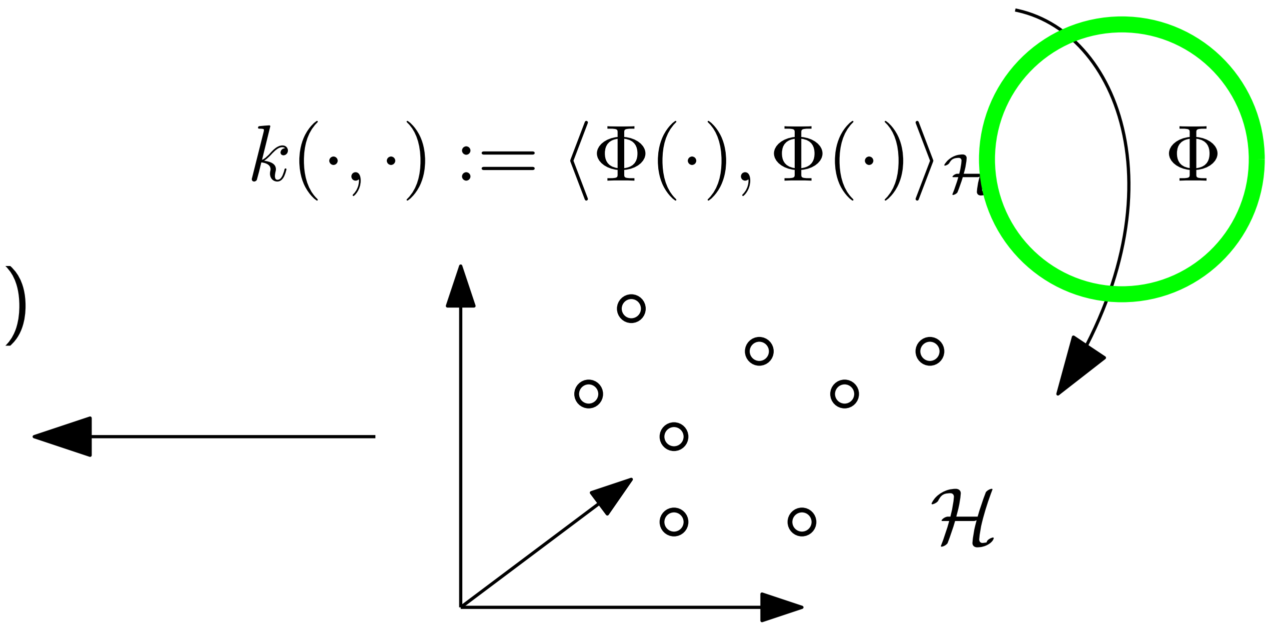
Etc.

Persistence diagrams and machine learning



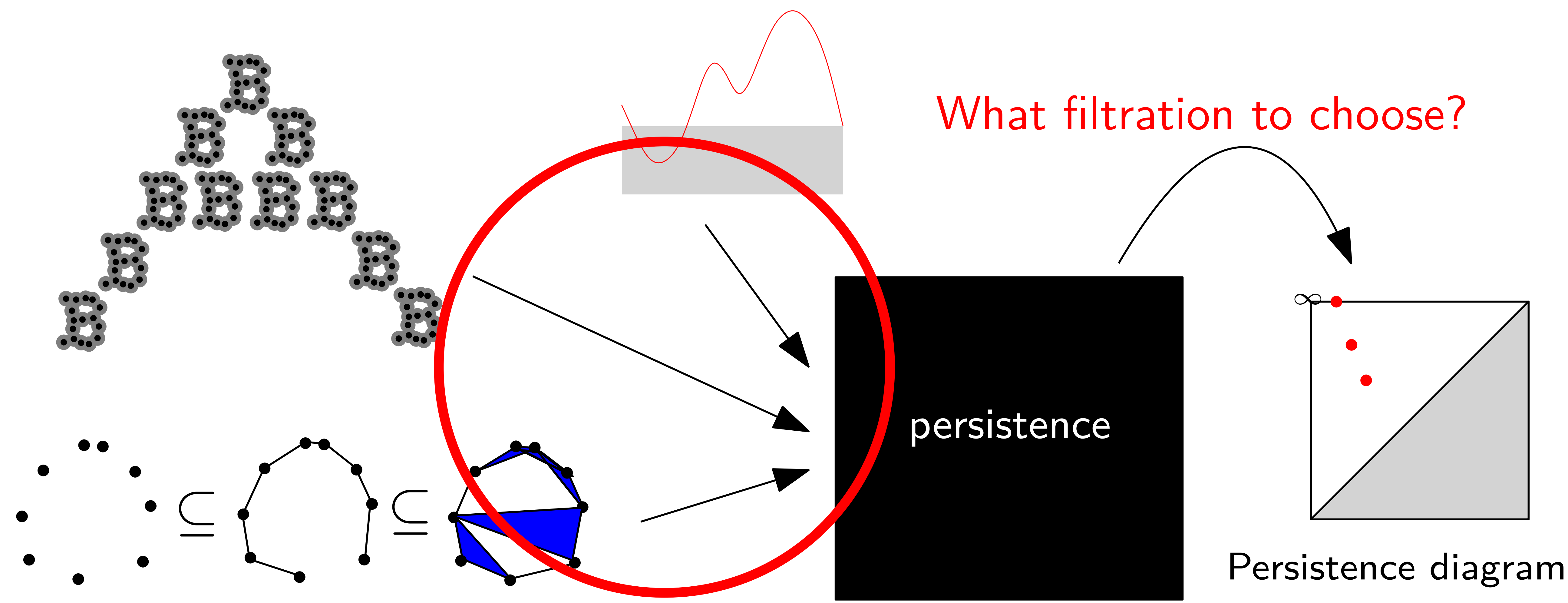
- Classifier (RF, SVM, NN etc.)
- Dim. red. (PCA, MDS, UMAP, t-SNE)
- Clustering (DBSCAN, K-means, etc.)

Etc.

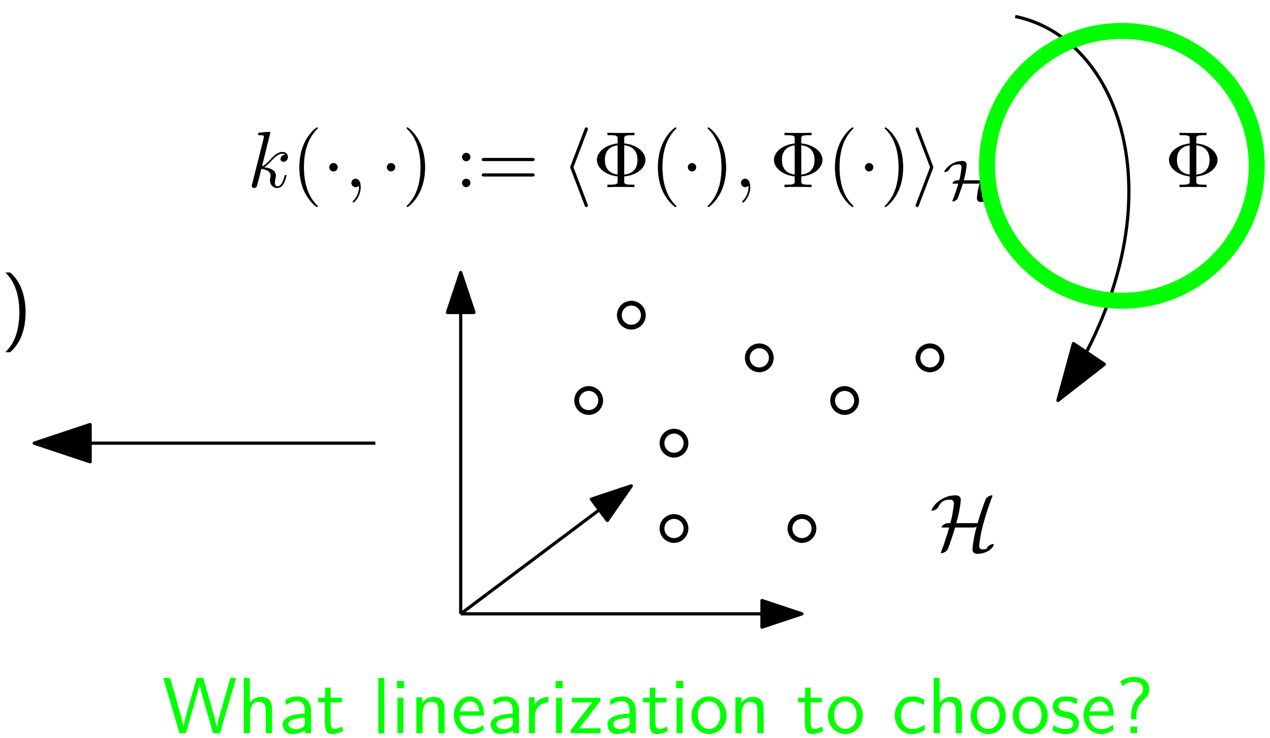


What linearization to choose?

Persistence diagrams and machine learning



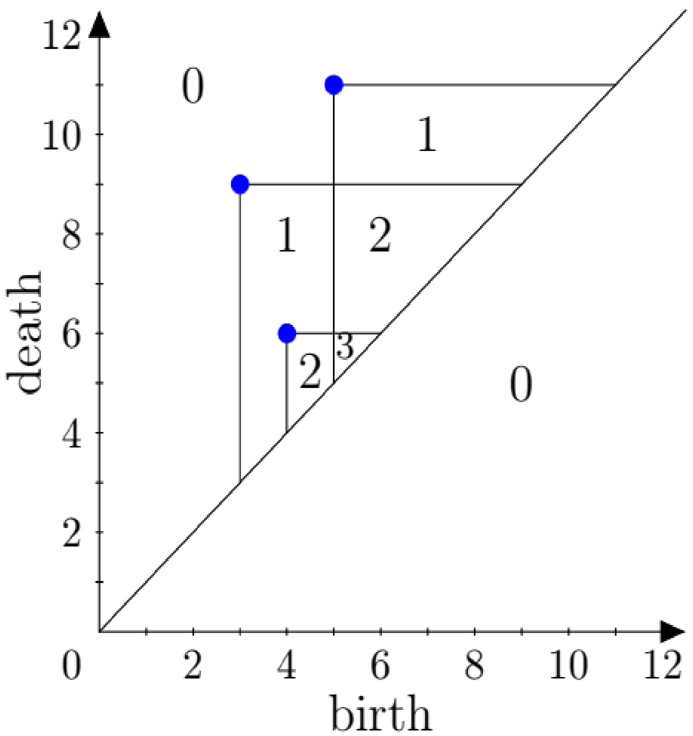
- Classifier (RF, SVM, NN etc.)
- Dim. red. (PCA, MDS, UMAP, t-SNE)
- Clustering (DBSCAN, K-means, etc.)

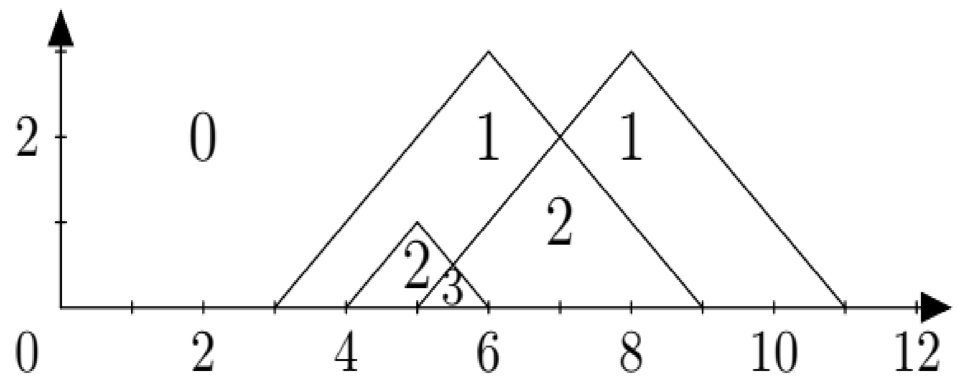


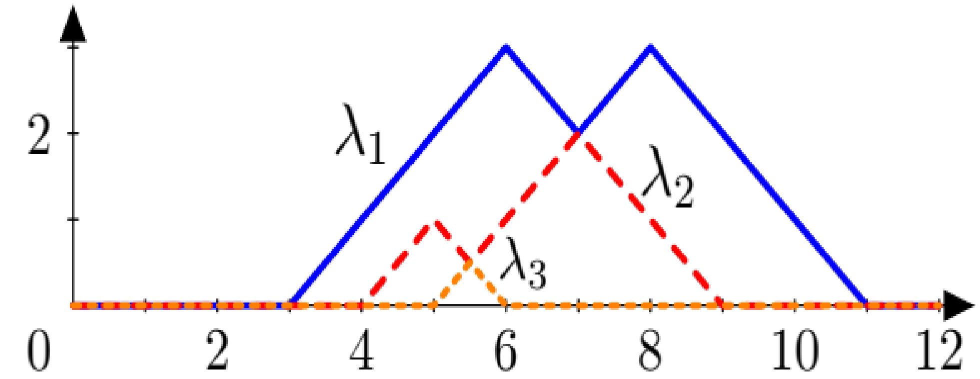
Etc.

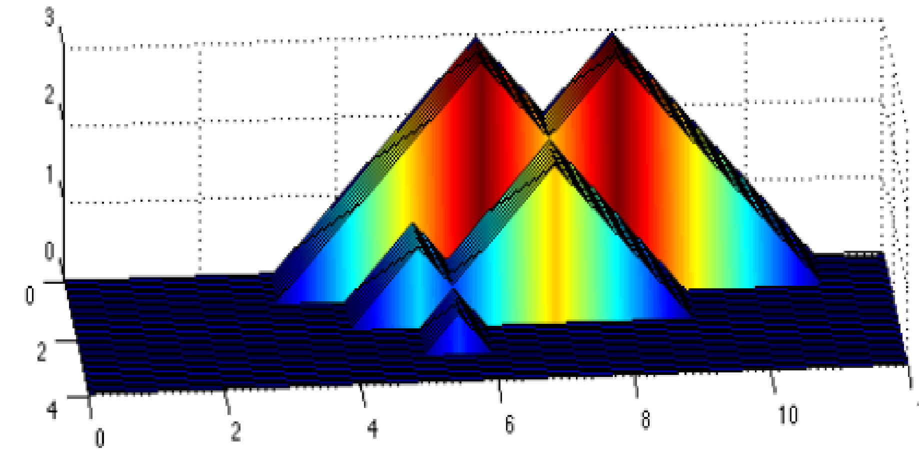
Landscapes

[*Statistical Topological Data Analysis using Persistence Landscapes*, Bubenik, JMLR, 2015]







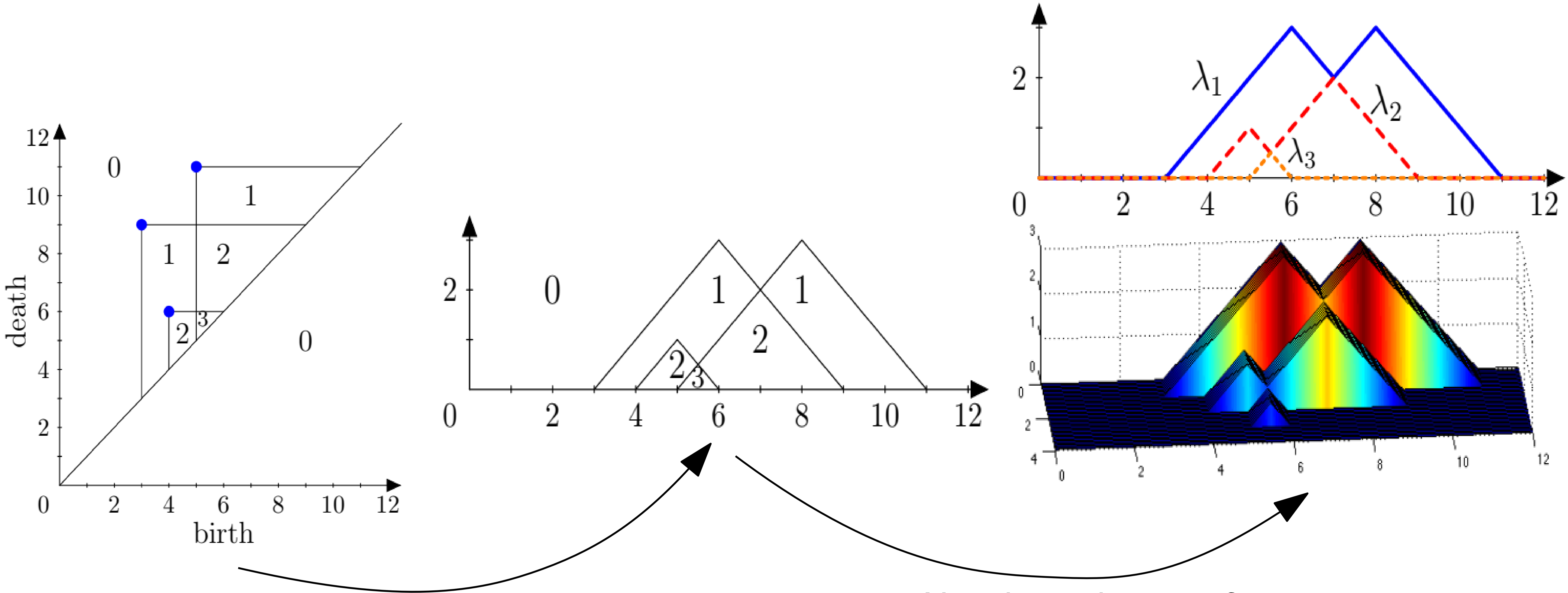


Rotate PD
Compute rank function

Use boundaries of rank function

Landscapes

[*Statistical Topological Data Analysis using Persistence Landscapes*, Bubenik, JMLR, 2015]



Rotate PD
Compute rank function

Use boundaries of rank function

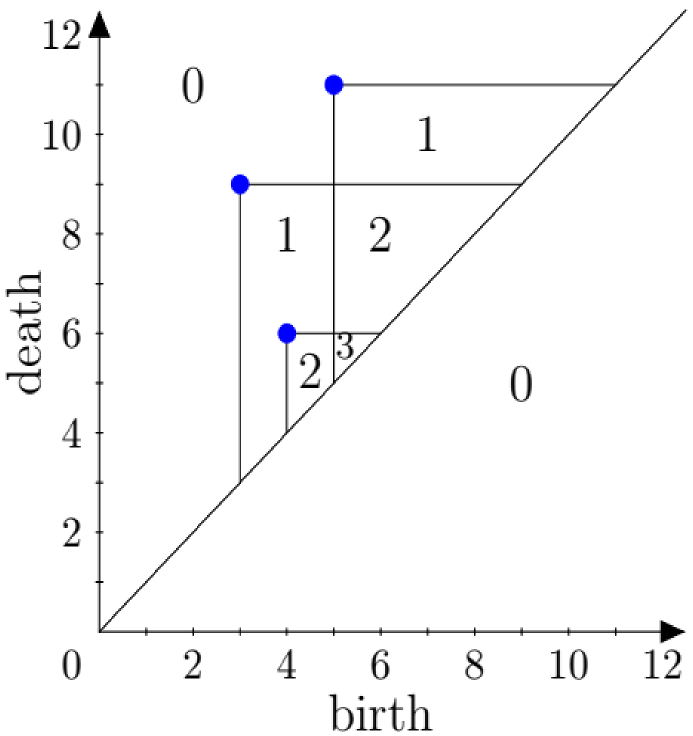
$$x \leq y \implies f^{-1}(-\infty, x) \subseteq f^{-1}(-\infty, y)$$

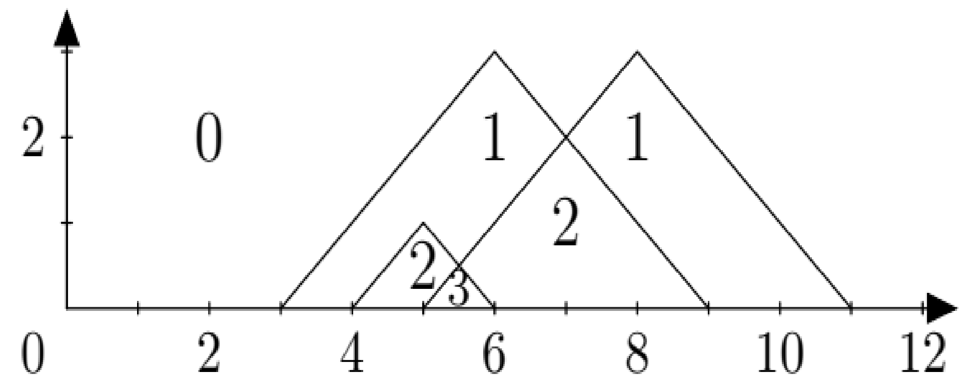
$\iota_x^y : H(f^{-1}(-\infty, x)) \to H(f^{-1}(-\infty, y))$ induced linear map

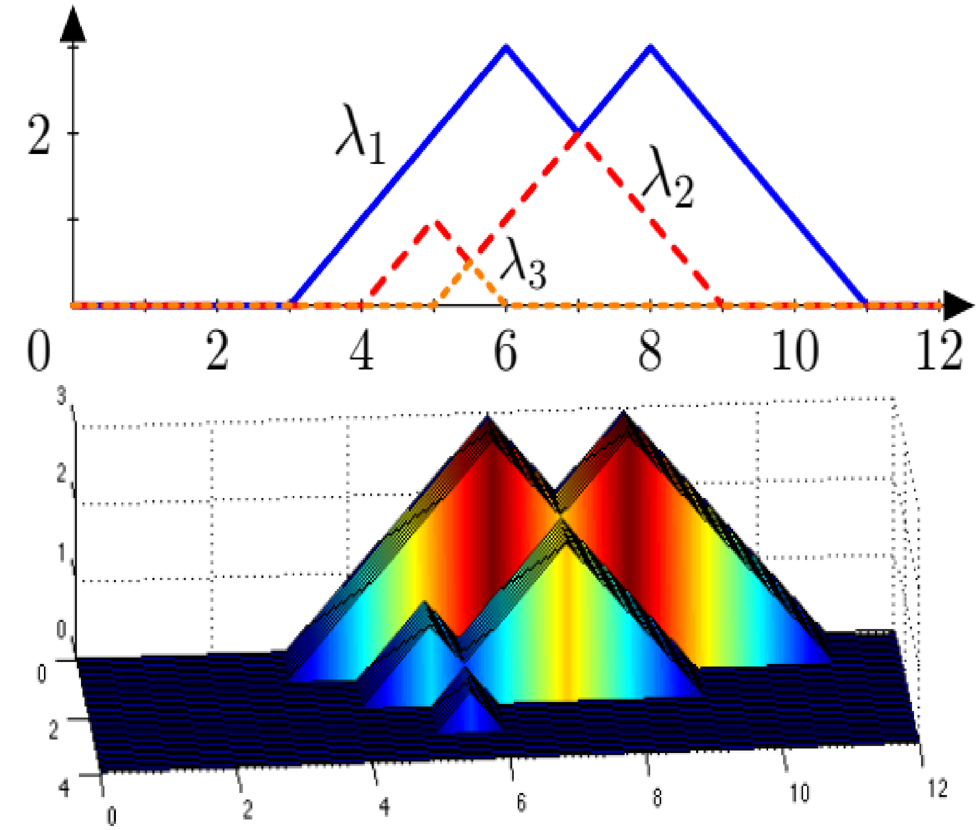
Rank function is defined as $\lambda(x, y) = \operatorname{rank} \iota_x^y$

Landscapes

[*Statistical Topological Data Analysis using Persistence Landscapes*, Bubenik, JMLR, 2015]







Rotate PD
Compute rank function

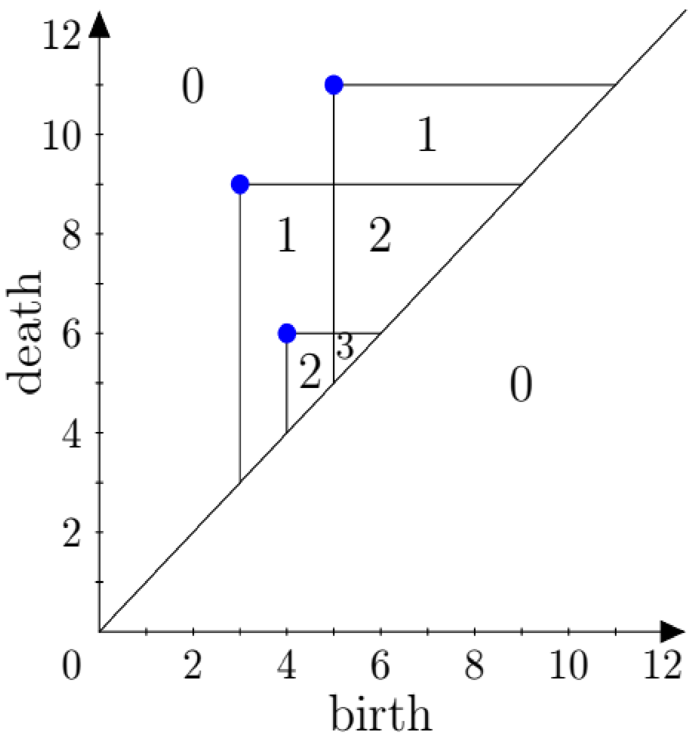
Use boundaries of rank function

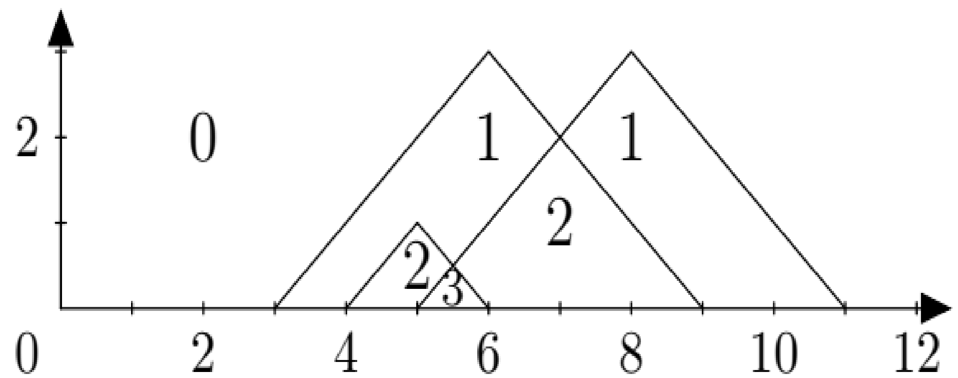
Boundaries of rank function: $\lambda_i(t) = \sup\{s \geq 0 : \lambda(t-s, t+s) \geq i\}$

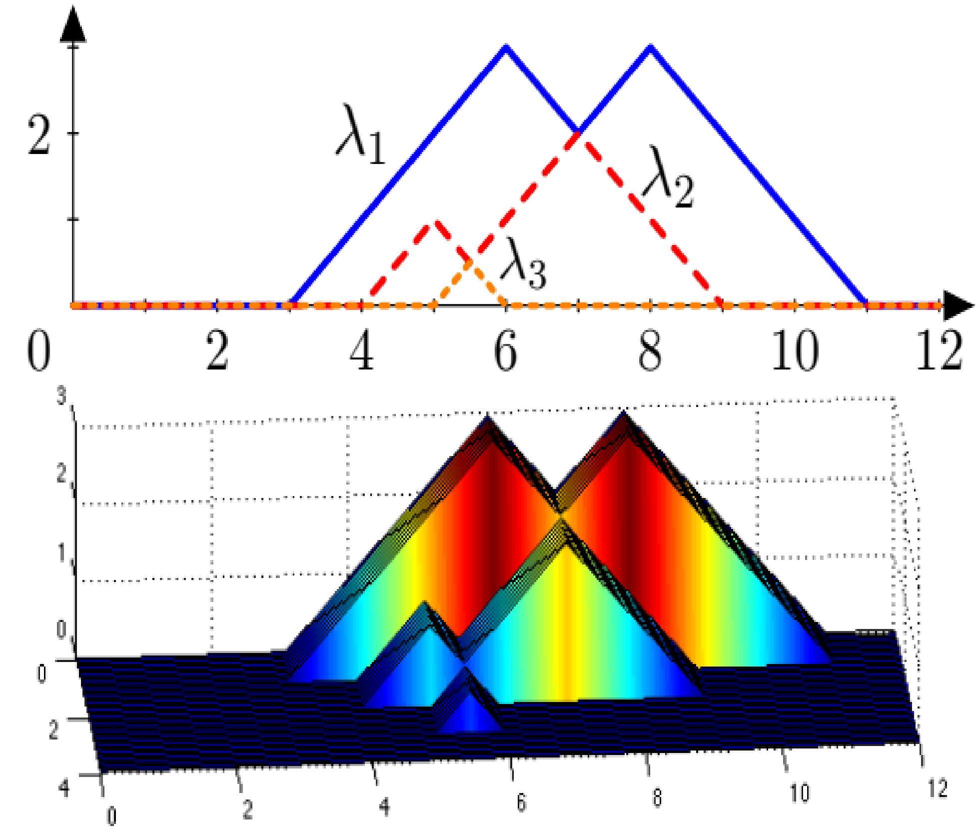
Landscape $\Lambda : \mathbb{R}^2 \to \mathbb{R}$ is defined as: $\Lambda(i,t) = \lambda_{\lfloor i \rfloor}(t)$

Landscapes

[*Statistical Topological Data Analysis using Persistence Landscapes*, Bubenik, JMLR, 2015]







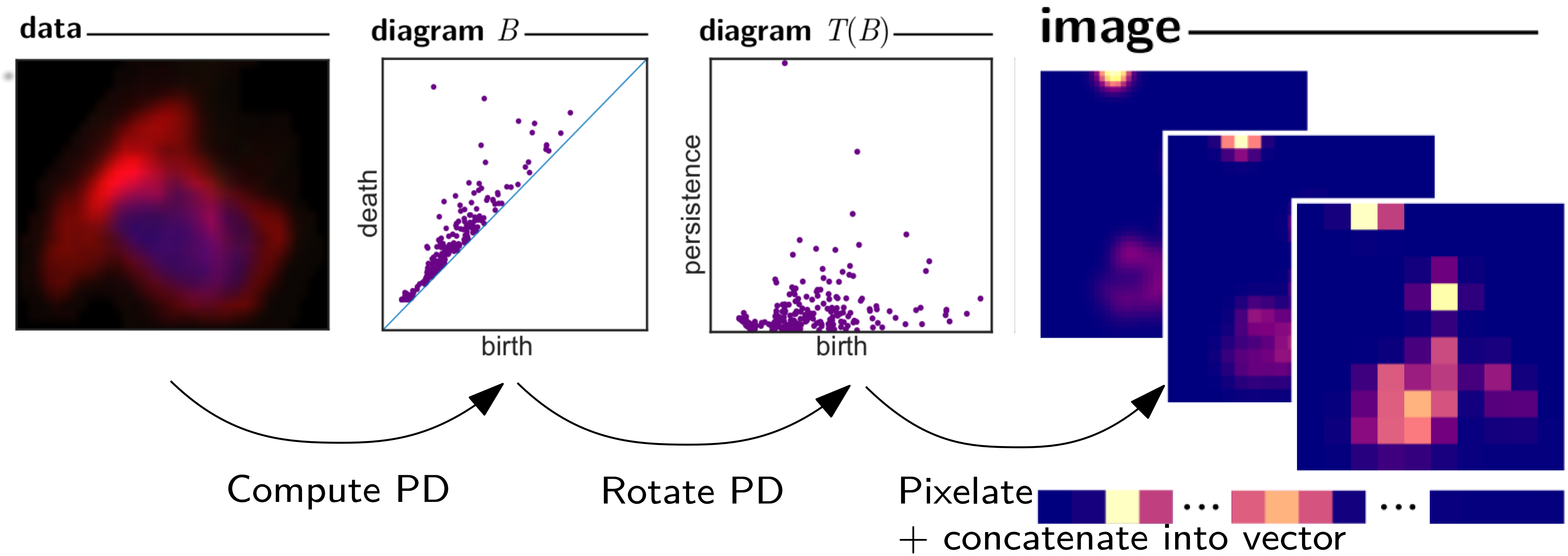
Rotate PD
Compute rank function

Use boundaries of rank function

Prop: $\|\Lambda(D) - \Lambda(D')\|_\infty \leq d_B(D, D')$

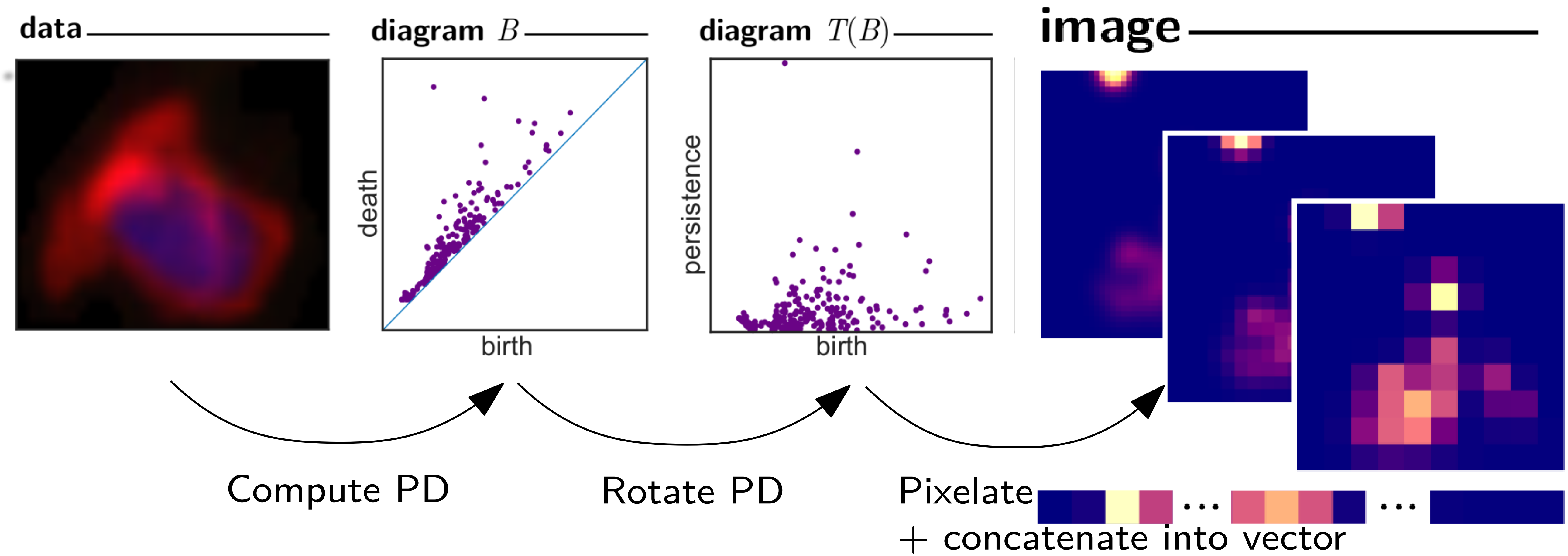
Persistence Images

[*Persistence Images: A Stable Vector Representation of Persistent Homology*, Adams et al., JMLR, 2017]

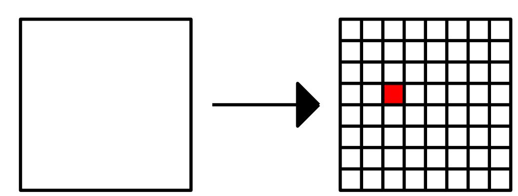


Persistence Images

[*Persistence Images: A Stable Vector Representation of Persistent Homology*, Adams et al., JMLR, 2017]



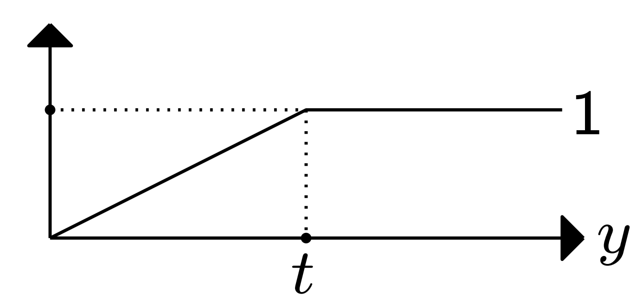
Discretize plane into one or several grid(s):



For each pixel P, compute $I(P) = \int\int_P \sum_{p\in D} w(p)\cdot\mathcal{N}(p,\sigma)$

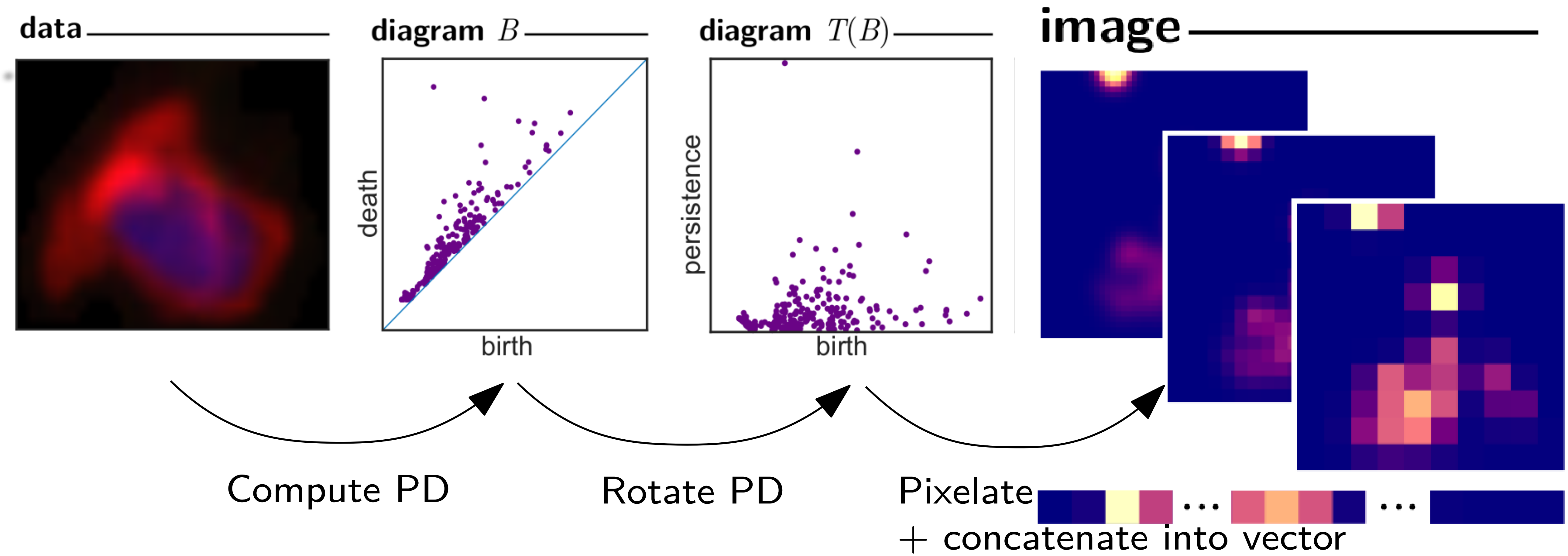
Concatenate all $I(P)$ into a single vector $\mathrm{PI}(D)$

Example: $w_t(x,y) =$



Persistence Images

[*Persistence Images: A Stable Vector Representation of Persistent Homology*, Adams et al., JMLR, 2017]



Prop:

- $\|\mathrm{PI}(D) - \mathrm{PI}(D')\|_\infty \leq C(w)\, d_1(D, D')$
- $\|\mathrm{PI}(D) - \mathrm{PI}(D')\|_2 \leq \sqrt{d}\, C(w)\, d_1(D, D')$

Sliced Wasserstein distance

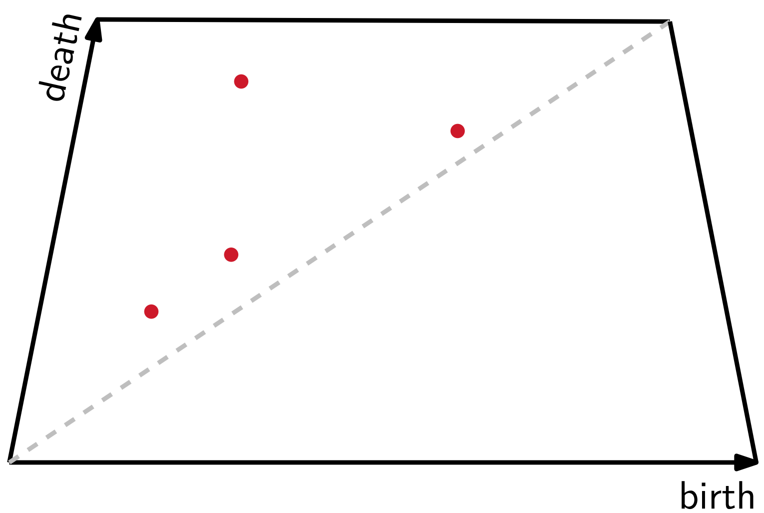
[*Sliced Wasserstein Kernel for Persistence Diagrams*, C., Cuturi, Oudot, ICML, 2017]

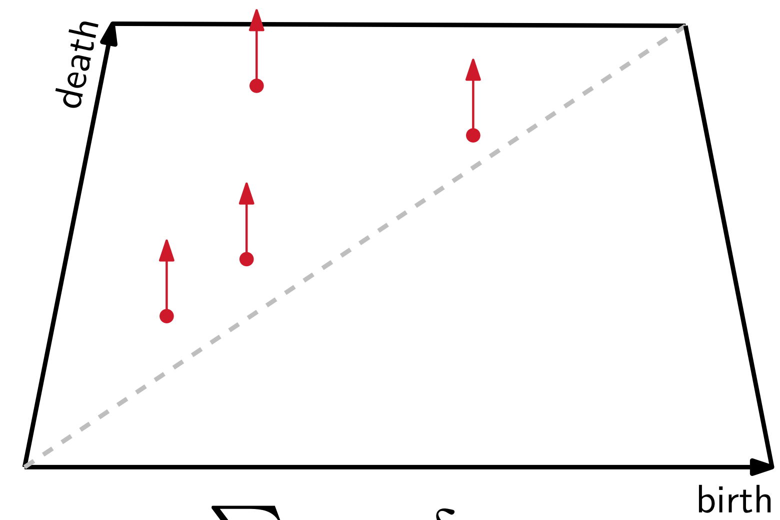
In practice, we are also interested in finding *kernels*, i.e., functions $k(\cdot,\cdot)$ such that $\exists \Phi : \mathcal{D} \to \mathcal{H}$ s.t. $k(\cdot,\cdot) = \langle \Phi(\cdot), \Phi(\cdot) \rangle_{\mathcal{H}}$

It is well known that $\exp\left(-\frac{d(\cdot,\cdot)}{\sigma}\right)$ is a kernel iif d is conditionally negative semidefinite (cnsd).

Sliced Wasserstein distance

[*Sliced Wasserstein Kernel for Persistence Diagrams*, C., Cuturi, Oudot, ICML, 2017]





$\mu_D = \sum_{p \in D} \delta_p$

Discrete measures accurately represent persistence diagrams

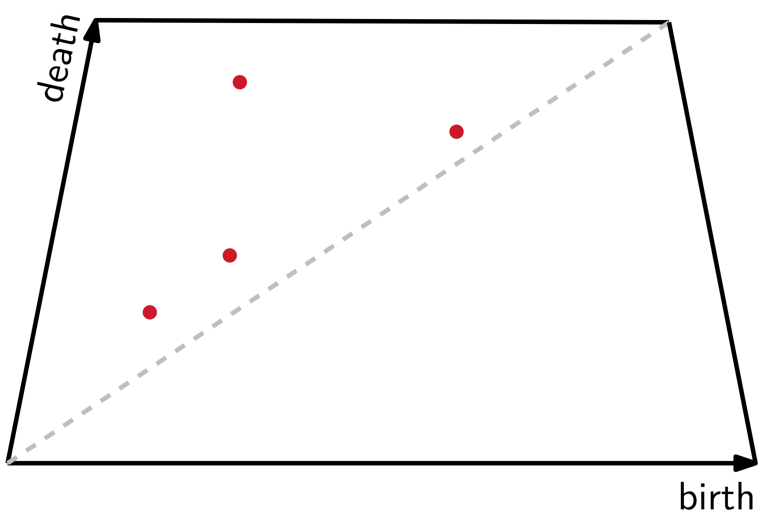
Discrete measures can be compared with 1-Wasserstein distance which is (almost) cnsd and looks like d_1

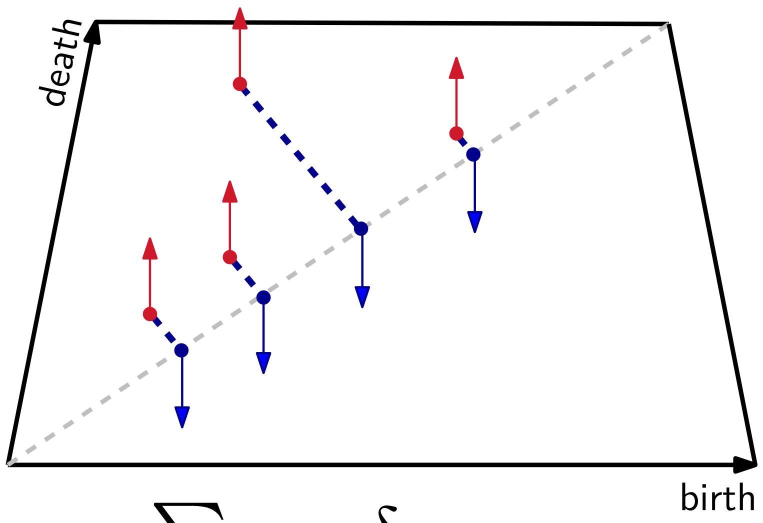
Def: Let $\mu = \sum_{i=1}^{n} \delta_{x_i}$ and $\nu = \sum_{i=1}^{n} \delta_{y_i}$

$$W_1(\mu, \nu) = \inf_\pi \ \sum_{i=1}^{n} \|x_i - y_{\pi(i)}\|$$

Sliced Wasserstein distance

[*Sliced Wasserstein Kernel for Persistence Diagrams*, C., Cuturi, Oudot, ICML, 2017]





$\mu_D = \sum_{p \in D} \delta_p$

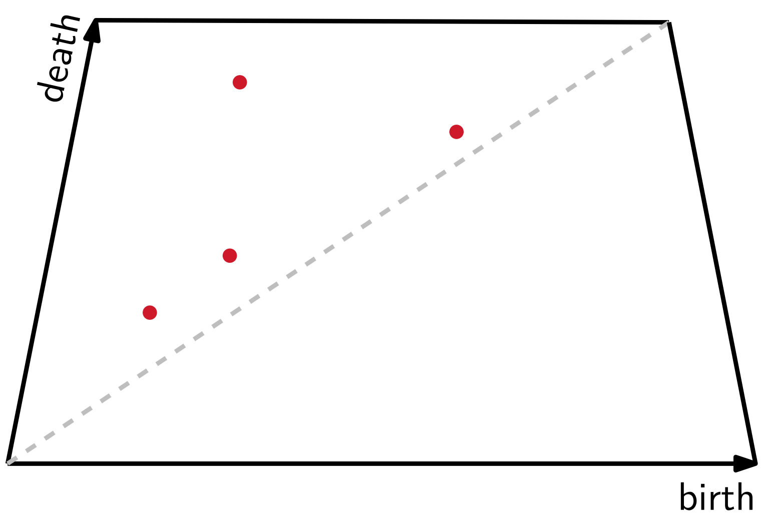
Pb: $d_1(D, D') \neq W_1(\mu_D, \mu_{D'})$ (W_1 does not even make sense)

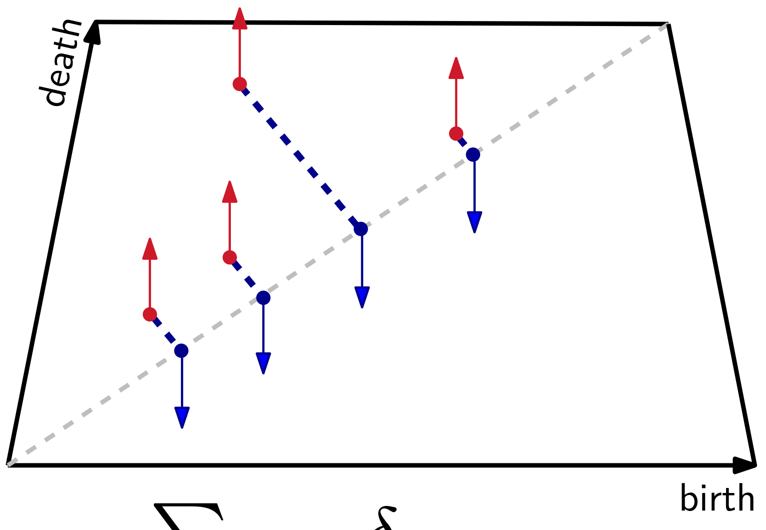
Solution: use projections onto the diagonal

$$\mu_D^+ = \sum_{p \in D} \delta_p \qquad \mu_D^- = \sum_{p \in D} \delta_{\pi_\Delta(p)}$$

Sliced Wasserstein distance

[*Sliced Wasserstein Kernel for Persistence Diagrams*, C., Cuturi, Oudot, ICML, 2017]





$\mu_D = \sum_{p \in D} \delta_p$

Pb: $d_1(D, D') \neq W_1(\mu_D, \mu_{D'})$ (W_1 does not even make sense)

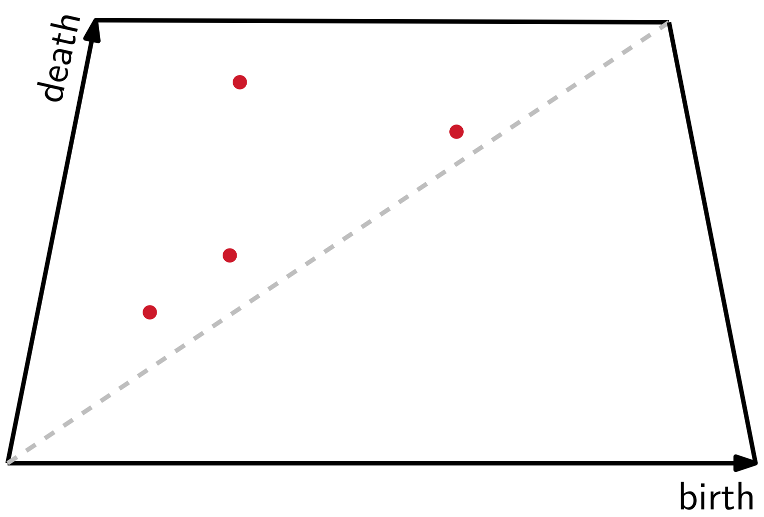
Solution: use projections onto the diagonal

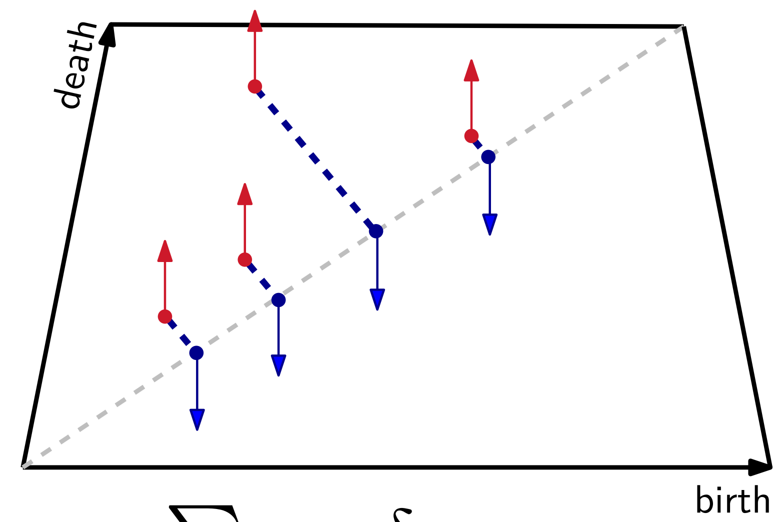
$$\mu_D^+ = \sum_{p \in D} \delta_p \qquad \mu_D^- = \sum_{p \in D} \delta_{\pi_\Delta(p)}$$

Prop: $d_1(D, D') \leq W_1(\mu_D^+ + \mu_{D'}^-, \, \mu_{D'}^+ + \mu_D^-) \leq 2\, d_1(D, D')$

Sliced Wasserstein distance

[*Sliced Wasserstein Kernel for Persistence Diagrams*, C., Cuturi, Oudot, ICML, 2017]





$\mu_D = \sum_{p \in D} \delta_p$

Pb: $d_1(D, D') \neq W_1(\mu_D, \mu_{D'})$ (W_1 does not even make sense)

Solution: use projections onto the diagonal

$$\mu_D^+ = \sum_{p \in D} \delta_p \qquad \mu_D^- = \sum_{p \in D} \delta_{\pi_\Delta(p)}$$

Prop: $d_1(D, D') \leq W_1(\mu_D^+ + \mu_{D'}^-, \mu_{D'}^+ + \mu_D^-) \leq 2\, d_1(D, D')$

Pb: W_1 is not cnsd, neither is d_1

Solution: relax the metric with slicing

Sliced Wasserstein distance

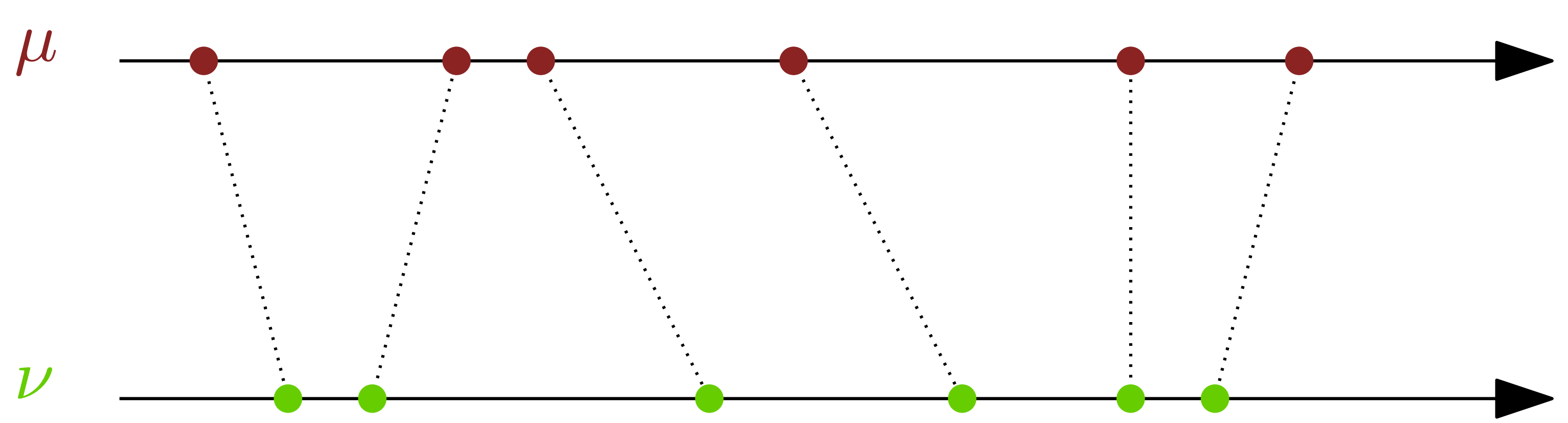
[*Sliced Wasserstein Kernel for Persistence Diagrams*, C., Cuturi, Oudot, ICML, 2017]

Special case: $X = \mathbb{R}$, μ, ν discrete measures of mass n

$\mu = \sum_{i=1}^n \delta_{x_i}, \;\; \nu = \sum_{i=1}^n \delta_{y_i}$

Sort the atoms of μ, ν along the real line: $x_i \leq x_{i+1}$ and $y_i \leq y_{i+1}$ for all i

Then: $W_1(\mu, \nu) = \sum_{i=1}^n |x_i - y_i| = \|(x_1, \ldots, x_n) - (y_1, \ldots, y_n)\|_1$



$\rightarrow W_1$ is cnsd and easy to compute

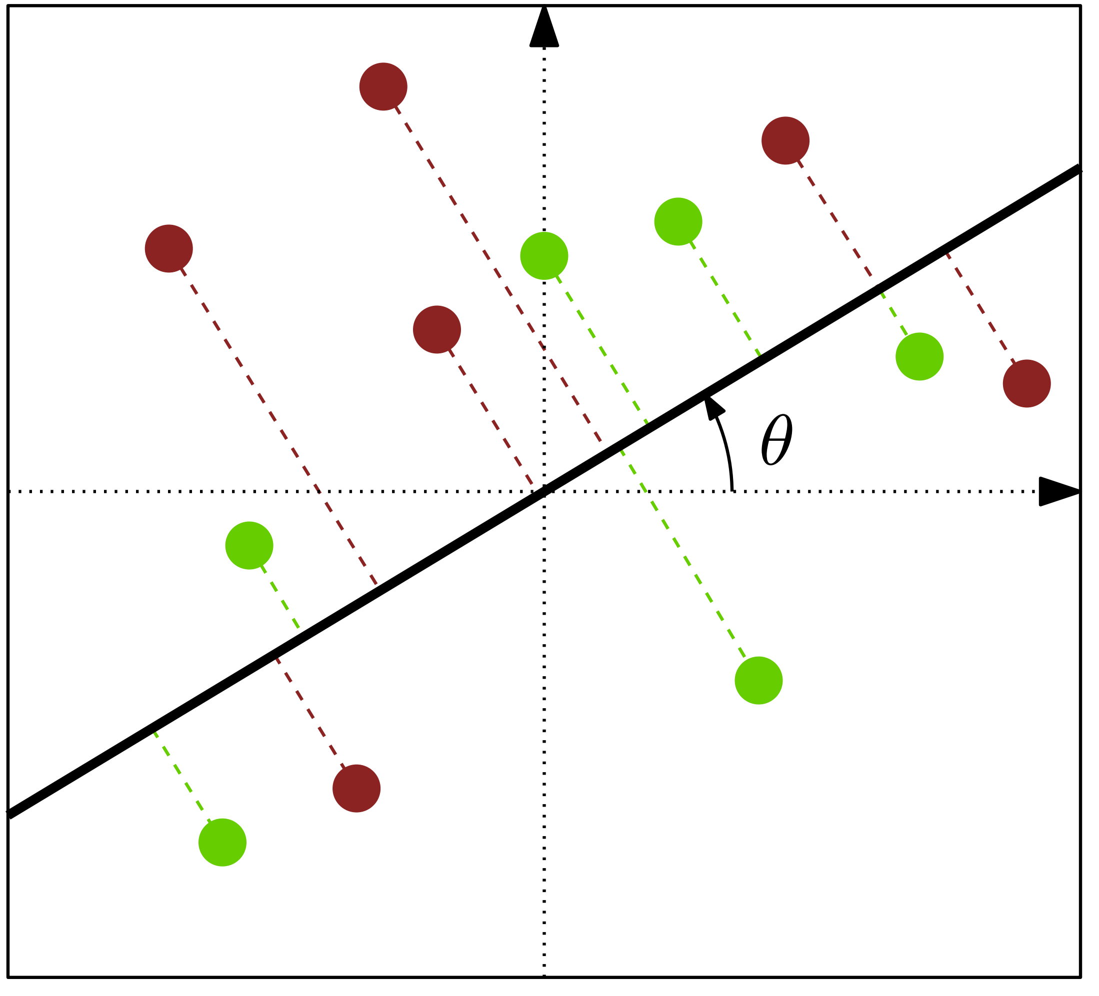
Sliced Wasserstein distance

[*Sliced Wasserstein Kernel for Persistence Diagrams*, C., Cuturi, Oudot, ICML, 2017]

Def (Sliced Wasserstein distance): for D, D',

$$SW(D, D') = \frac{1}{2\pi} \int_{\mathbb{S}^1} W_1(\pi_\theta \#(\mu_D^+ + \mu_{D'}^-), \pi_\theta \#(\mu_{D'}^+ + \mu_D^-))\, \mathrm{d}\theta$$

where π_θ = orthogonal projection onto line passing through origin with angle θ



Sliced Wasserstein distance

[*Sliced Wasserstein Kernel for Persistence Diagrams*, C., Cuturi, Oudot, ICML, 2017]

Def (Sliced Wasserstein distance): for D, D',

$$SW(D, D') = \frac{1}{2\pi} \int_{\mathbb{S}^1} W_1(\pi_\theta \#(\mu_D^+ + \mu_{D'}^-), \pi_\theta \#(\mu_{D'}^+ + \mu_D^-)) \, \mathrm{d}\theta$$

where $\pi_\theta =$ orthogonal projection onto line passing through origin with angle θ

Prop: (inherited from W_1 over $\mathbb{R}$)

- satisfies the axioms of a metric

- well-defined barycenters, fast to compute via SGD, etc.

- conditionally negative semidefinite

Sliced Wasserstein distance

[*Sliced Wasserstein Kernel for Persistence Diagrams*, C., Cuturi, Oudot, ICML, 2017]

Def (Sliced Wasserstein distance): for D, D',

$$SW(D, D') = \frac{1}{2\pi} \int_{\mathbb{S}^1} W_1(\pi_\theta \#(\mu_D^+ + \mu_{D'}^-), \pi_\theta \#(\mu_{D'}^+ + \mu_D^-)) \, \mathrm{d}\theta$$

where $\pi_\theta =$ orthogonal projection onto line passing through origin with angle θ

Th: d_1 and SW are strongly equivalent, namely: for D, D',

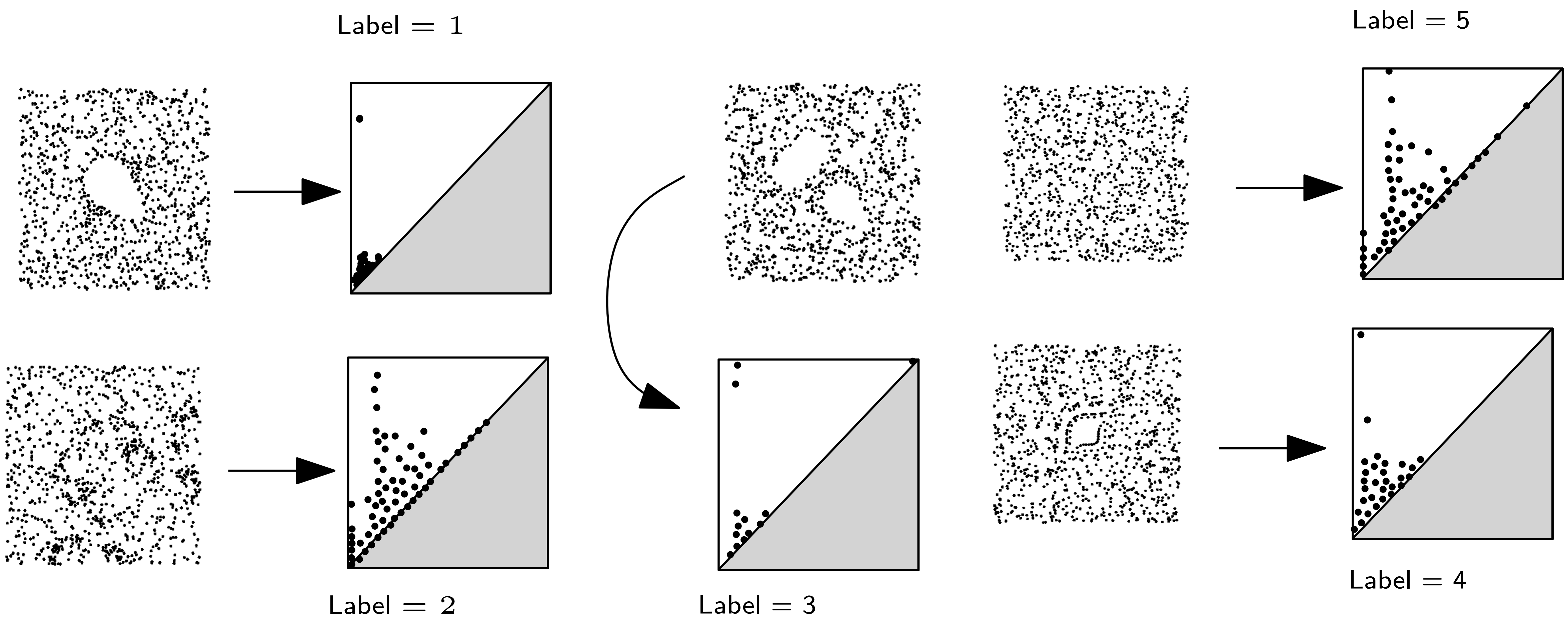
$$\frac{1}{2+4N(2N-1)} d_1(D, D') \leq SW(D, D') \leq 2\sqrt{2}\, d_1(D, D')$$

Orbit classification

Goal: classify orbits of *linked twisted map*, modelling fluid flow dynamics

Orbits described by (depending on parameter r):

$$\begin{cases} x_{n+1} = x_n + r\, y_n(1 - y_n) \mod 1 \\ y_{n+1} = y_n + r\, x_{n+1}(1 - x_{n+1}) \mod 1 \end{cases}$$



Orbit classification

Goal: classify orbits of *linked twisted map*, modelling fluid flow dynamics

Orbits described by (depending on parameter r):

$$\begin{cases} x_{n+1} = x_n + r\, y_n(1 - y_n) \mod 1 \\ y_{n+1} = y_n + r\, x_{n+1}(1 - x_{n+1}) \mod 1 \end{cases}$$

Accuracies (%) using only TDA descriptors (kernels on barcodes):

	k_{PSS}	k_{PWG}	k_{SW}
Orbit	64.0 ± 0.0	78.7 ± 0.0	$\mathbf{83.7} \pm 1.1$

(PDs as discrete measures)

Orbit classification

Goal: classify orbits of *linked twisted map*, modelling fluid flow dynamics

Orbits described by (depending on parameter r):

$$\begin{cases} x_{n+1} = x_n + r\, y_n(1-y_n) \mod 1 \\ y_{n+1} = y_n + r\, x_{n+1}(1-x_{n+1}) \mod 1 \end{cases}$$

Accuracies ($\%$) using only TDA descriptors (kernels on barcodes):

	k_{PSS}	k_{PWG}	k_{SW}
Orbit	64.0 ± 0.0	78.7 ± 0.0	$\mathbf{83.7} \pm 1.1$

(PDs as discrete measures)

Running times (in seconds on N-sized parameter space from 100 orbits):

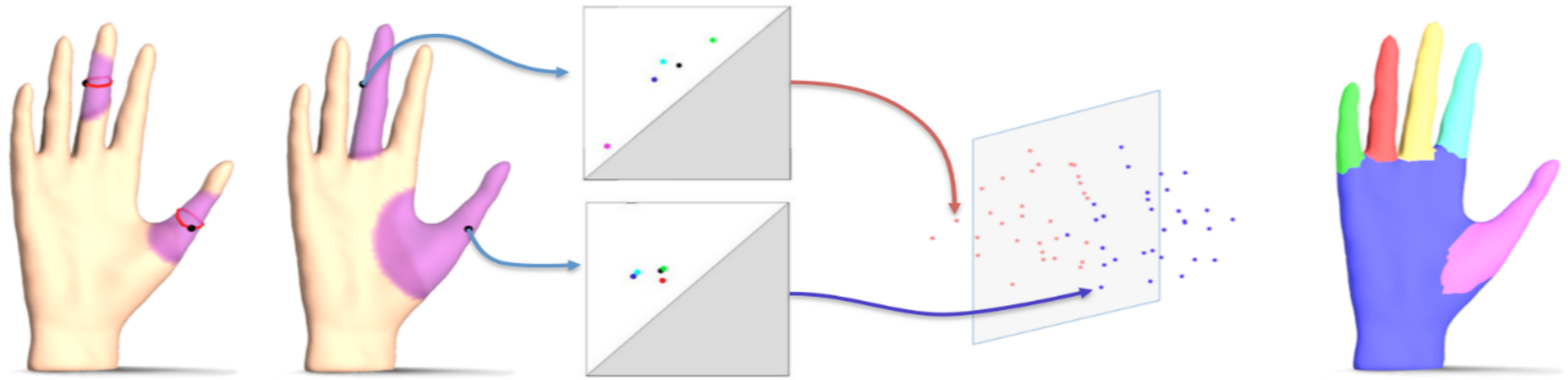
	k_{PSS}	k_{PWG}	k_{SW}
Orbit	$N \times 9183.4 \pm 65.6$	$N \times 69.2 \pm 0.9$	$385.8 \pm 0.2 + NC$

($\phi(\cdot)$ recomputed for each σ) (SW computed only once)

Shape segmentation

[*Stable Topological Signatures for points on 3D Shapes*, C., Oudot, Ovsjanikov, SGP, 2015]

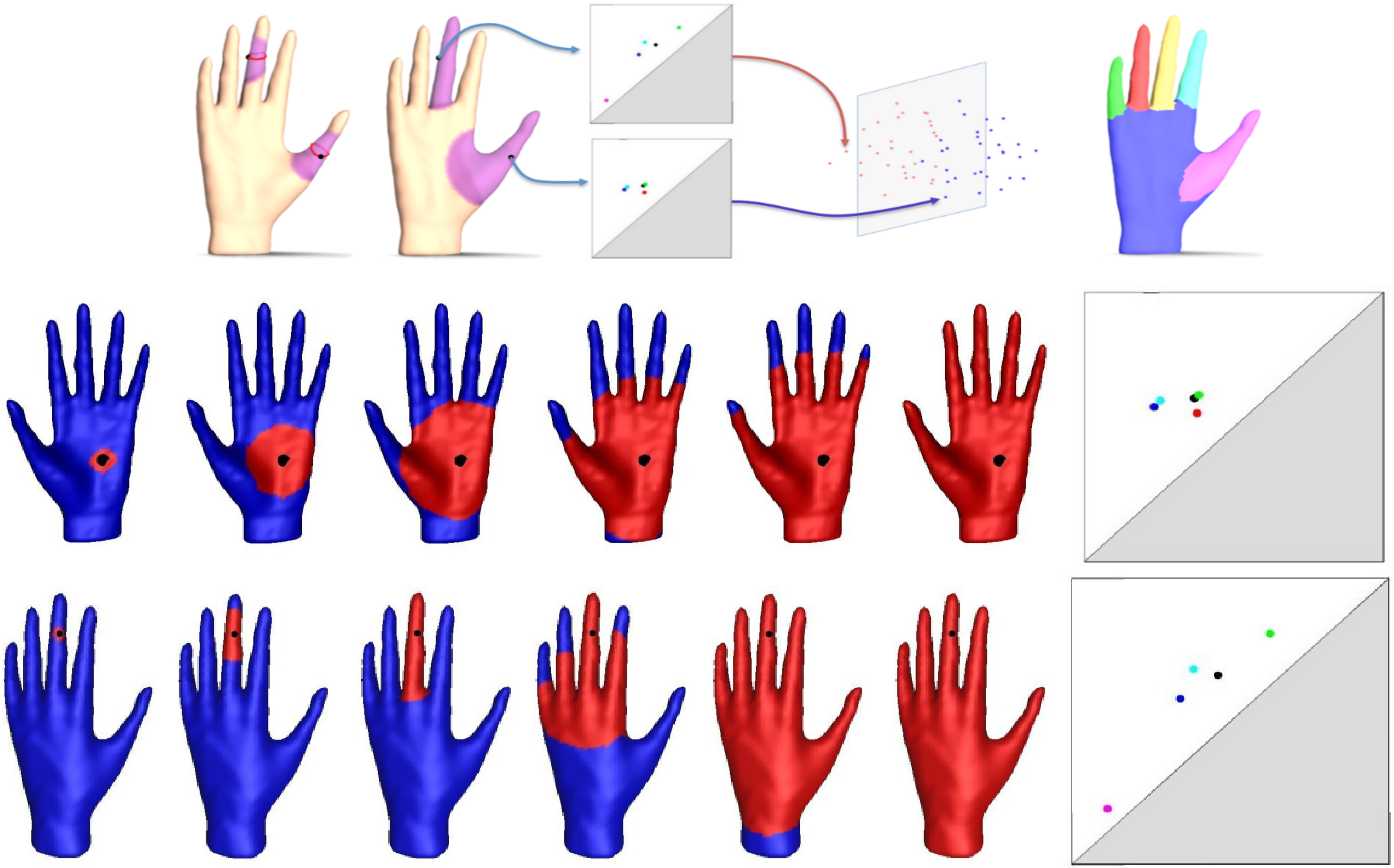
Goal: segment 3d shapes



Shape segmentation

[*Stable Topological Signatures for points on 3D Shapes*, C., Oudot, Ovsjanikov, SGP, 2015]

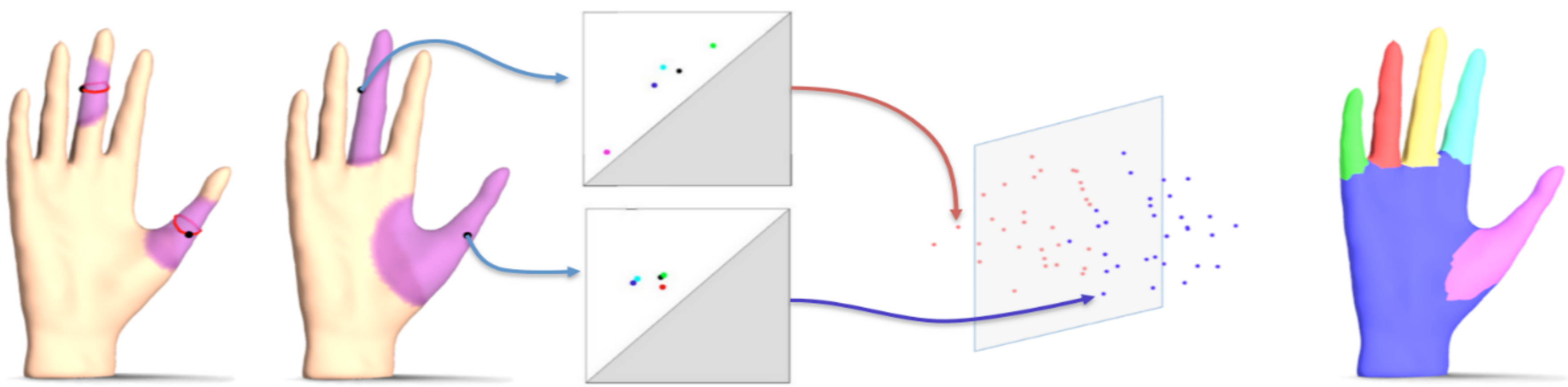
Goal: segment 3d shapes

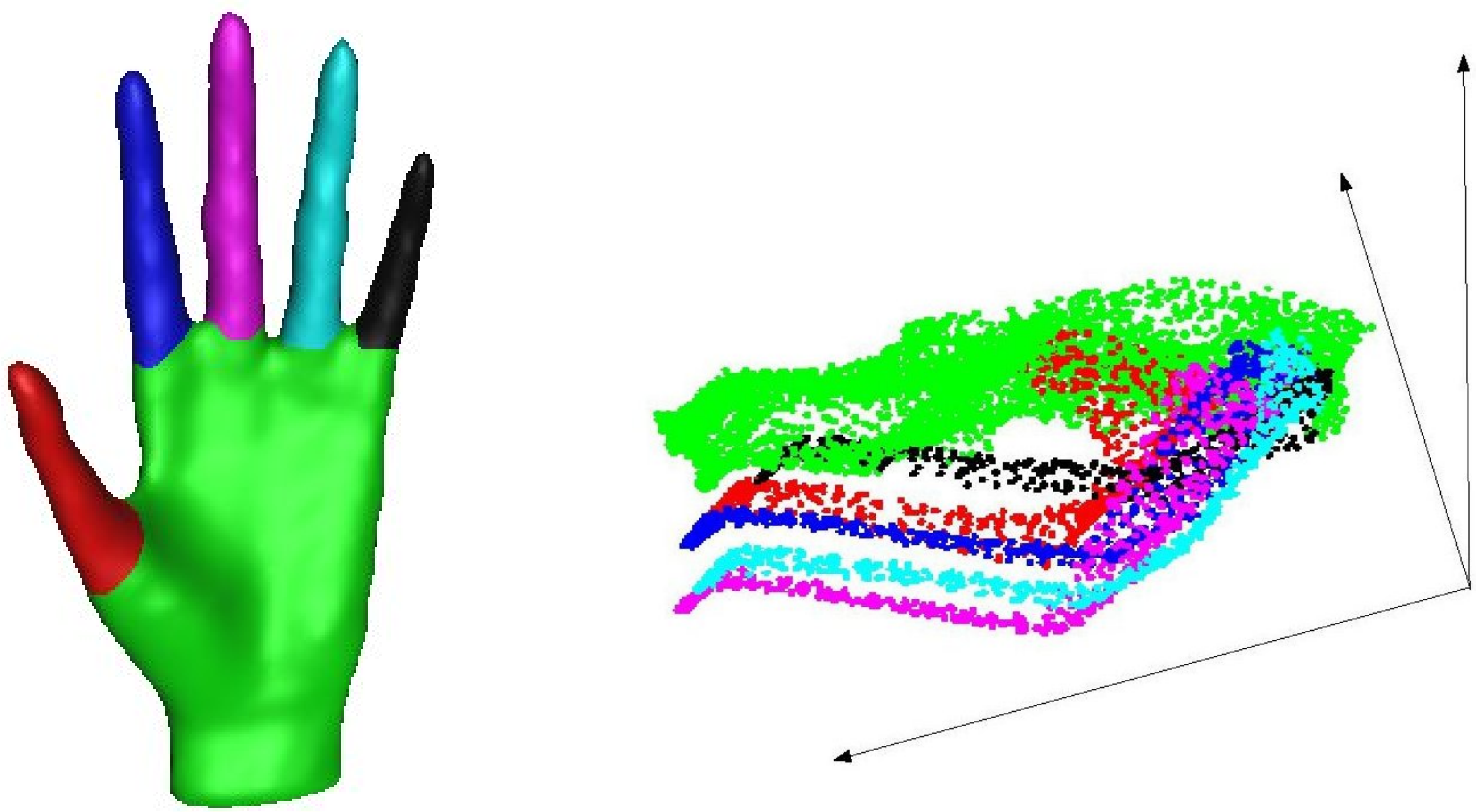


Shape segmentation

[*Stable Topological Signatures for points on 3D Shapes*, C., Oudot, Ovsjanikov, SGP, 2015]

Goal: segment 3d shapes





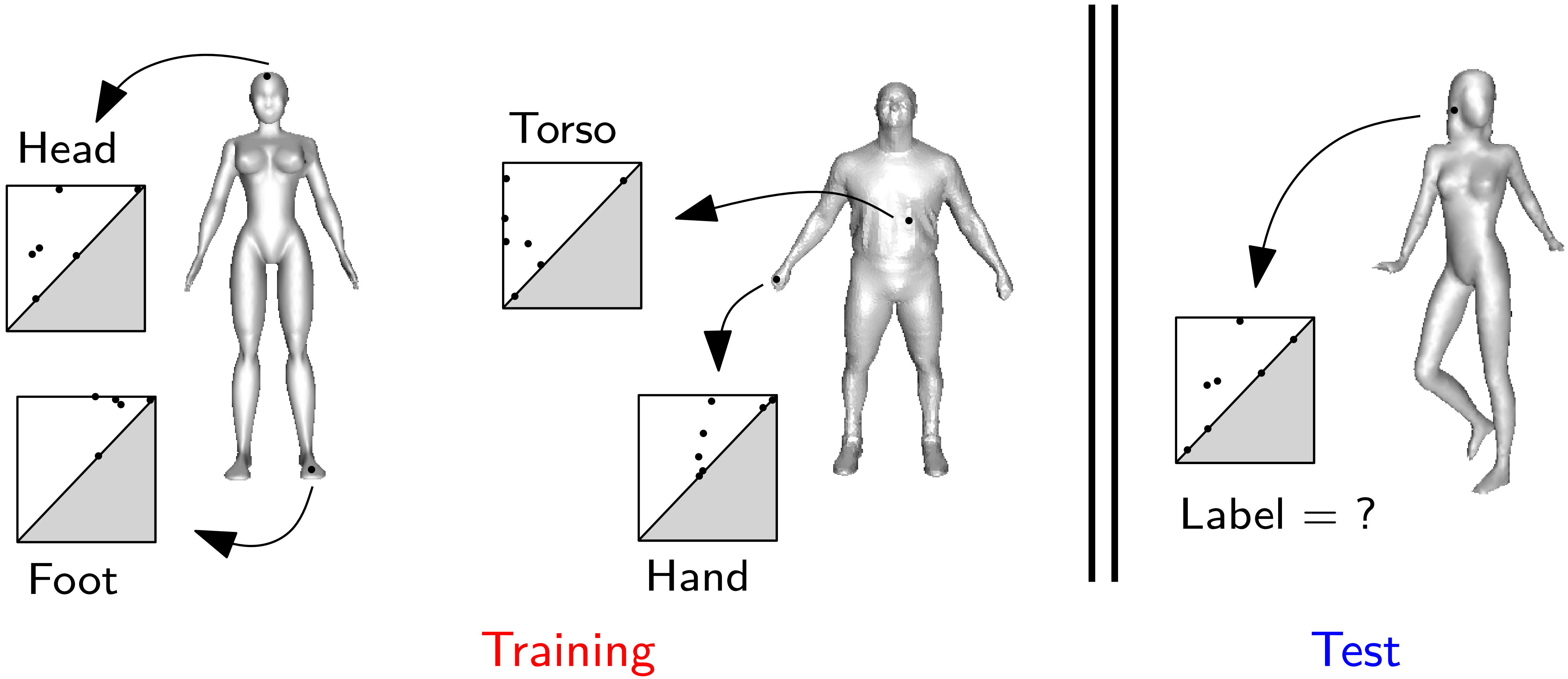
Shape segmentation

[*Stable Topological Signatures for points on 3D Shapes*, C., Oudot, Ovsjanikov, SGP, 2015]

Goal: segment 3d shapes based on examples

Approach:

- train a (multiclass) classifier on PDs extracted from the training shapes
- apply classifier to PDs extracted from query shape



Shape segmentation

[*Stable Topological Signatures for points on 3D Shapes*, C., Oudot, Ovsjanikov, SGP, 2015]

Goal: segment 3d shapes based on examples

Approach:

- train a (multiclass) classifier on PDs extracted from the training shapes
- apply classifier to PDs extracted from query shape

Accuracies (%) using only TDA descriptors (kernels on barcodes):

	k_{PSS}	k_{PWG}	k_{SW}
Human	68.5 ± 2.0	64.2 ± 1.2	$\mathbf{74.0} \pm 0.2$
Airplane	55.4 ± 2.4	61.3 ± 2.9	$\mathbf{72.6} \pm 0.2$
Ant	86.3 ± 1.0	87.4 ± 0.5	$\mathbf{92.3} \pm 0.2$
FourLeg	67.0 ± 2.5	64.0 ± 0.6	$\mathbf{73.0} \pm 0.4$
Octopus	77.6 ± 1.0	78.6 ± 1.3	$\mathbf{85.2} \pm 0.5$
Bird	67.6 ± 1.8	$\mathbf{72.0} \pm 1.2$	67.0 ± 0.5
Fish	76.1 ± 1.6	$\mathbf{79.6} \pm 0.5$	75.0 ± 0.4

Shape segmentation

[*Stable Topological Signatures for points on 3D Shapes*, C., Oudot, Ovsjanikov, SGP, 2015]

Goal: segment 3d shapes based on examples

Approach:

- train a (multiclass) classifier on PDs extracted from the training shapes
- apply classifier to PDs extracted from query shape

Accuracies (%) using only TDA descriptors (kernels on barcodes):

	k_{PSS}	k_{PWG}	k_{SW}
Human	68.5 ± 2.0	64.2 ± 1.2	$\mathbf{74.0} \pm 0.2$
Airplane	55.4 ± 2.4	61.3 ± 2.9	$\mathbf{72.6} \pm 0.2$
Ant	86.3 ± 1.0	87.4 ± 0.5	$\mathbf{92.3} \pm 0.2$
FourLeg	67.0 ± 2.5	64.0 ± 0.6	$\mathbf{73.0} \pm 0.4$
Octopus	77.6 ± 1.0	78.6 ± 1.3	$\mathbf{85.2} \pm 0.5$
Bird	67.6 ± 1.8	$\mathbf{72.0} \pm 1.2$	67.0 ± 0.5
Fish	76.1 ± 1.6	$\mathbf{79.6} \pm 0.5$	75.0 ± 0.4

The Deep Set architecture

[*Deep Sets*, Zaheer, Kottur, Ravanbakhsh, Poczos, Salakhutdinov, Smola, NeurIPS, 2017]

Deep Set is a neural net architecture that is able to handle sets of points instead of single finite-dimensional vectors.

Input: $\{x_1, ..., x_n\} \subset \mathbb{R}^d$ instead of $x \in \mathbb{R}^d$

The Deep Set architecture

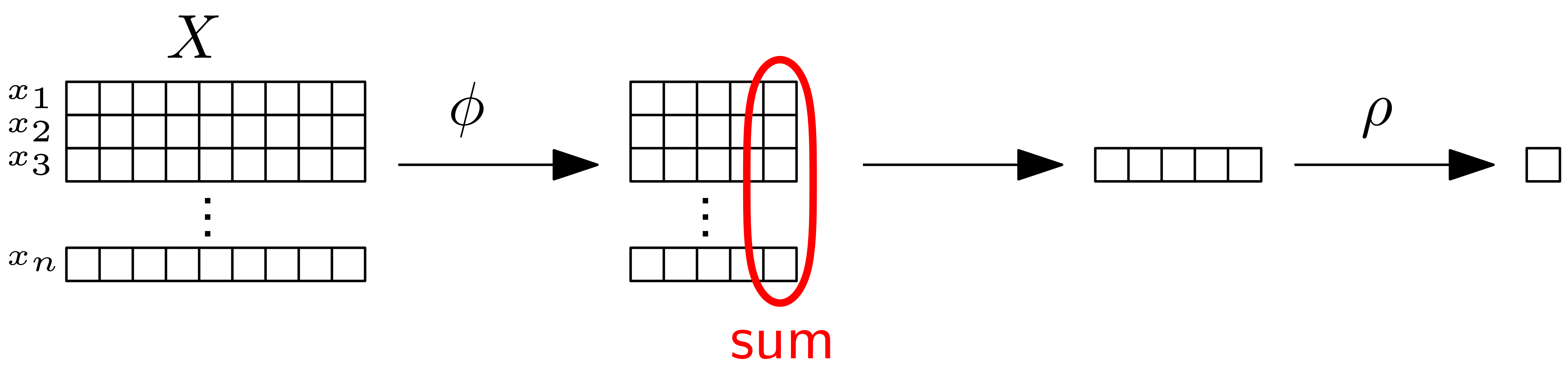
[*Deep Sets*, Zaheer, Kottur, Ravanbakhsh, Poczos, Salakhutdinov, Smola, NeurIPS, 2017]

Deep Set is a neural net architecture that is able to handle sets of points instead of single finite-dimensional vectors.

Input: $\{x_1, ..., x_n\} \subset \mathbb{R}^d$ instead of $x \in \mathbb{R}^d$

Network is *permutation invariant*: $\mathrm{DS}(X) = \rho\left(\sum_i \phi(x_i)\right)$.

$$\Rightarrow \mathrm{DS}(\{x_1, ..., x_n\}) = \mathrm{DS}(\{x_{\sigma(1)}, ..., x_{\sigma(n)}\}),\ \forall \sigma$$



In practice: $\phi(x_i) = \sigma(W \cdot x_i + b)$

The Deep Set architecture

[*Deep Sets*, Zaheer, Kottur, Ravanbakhsh, Poczos, Salakhutdinov, Smola, NeurIPS, 2017]

Deep Set is a neural net architecture that is able to handle sets of points instead of single finite-dimensional vectors.

Input: $\{x_1, ..., x_n\} \subset \mathbb{R}^d$ instead of $x \in \mathbb{R}^d$

Network is *permutation invariant*: $\mathrm{DS}(X) = \rho\left(\sum_i \phi(x_i)\right)$.

DeepSet networks also enjoy a universality theorem:

Thm: A function f is permutation invariant iif $f(X) = \rho\left(\sum_i \phi(x_i)\right)$ for some ρ and ϕ, whenever X is included in a *countable* space.

The Deep Set architecture

[*Deep Sets*, Zaheer, Kottur, Ravanbakhsh, Poczos, Salakhutdinov, Smola, NeurIPS, 2017]

Deep Set is a neural net architecture that is able to handle sets of points instead of single finite-dimensional vectors.

Input: $\{x_1, ..., x_n\} \subset \mathbb{R}^d$ instead of $x \in \mathbb{R}^d$

Network is *permutation invariant*: $\mathrm{DS}(X) = \rho\left(\sum_i \phi(x_i)\right)$.

DeepSet networks also enjoy a universality theorem:

Thm: A function f is permutation invariant iif $f(X) = \rho\left(\sum_i \phi(x_i)\right)$ for some ρ and ϕ, whenever X is included in a *countable* space.

It turns out that DeepSet networks are also *stable* in the Wasserstein distance.

Application to PDs

Application to PDs

Permutation invariant layers generalize several TDA approaches

Application to PDs

Permutation invariant layers generalize several TDA approaches

$\longrightarrow$ persistence images

Application to PDs

Permutation invariant layers generalize several TDA approaches

$\longrightarrow$ persistence images $\longrightarrow$ landscapes

Application to PDs

Permutation invariant layers generalize several TDA approaches

$\longrightarrow$ persistence images $\longrightarrow$ landscapes $\longrightarrow$ Betti curves [*Time Series Classification via Topological Data Analysis*, Umeda, Trans. Jap. Soc. for AI, 2017]

Application to PDs

Permutation invariant layers generalize several TDA approaches

$\rightarrow$ persistence images $\rightarrow$ landscapes $\rightarrow$ Betti curves [*Time Series Classification via Topological Data Analysis*, Umeda, Trans. Jap. Soc. for AI, 2017]

But not all of them since $\mathbb{R}^2$ is not countable

Application to PDs

Permutation invariant layers generalize several TDA approaches

$\rightarrow$ persistence images $\rightarrow$ landscapes $\rightarrow$ Betti curves [*Time Series Classification via Topological Data Analysis*, Umeda, Trans. Jap. Soc. for AI, 2017]

But not all of them since $\mathbb{R}^2$ is not countable

Using any permutation invariant operation (such as max, min, kth largest value) allows to generalize other TDA approaches

Application to PDs

[*PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures*, C., Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]

Permutation invariant layers generalize several TDA approaches

$\rightarrow$ persistence images $\rightarrow$ landscapes $\rightarrow$ Betti curves

[*Time Series Classification via Topological Data Analysis*, Umeda, Trans. Jap. Soc. for AI, 2017]

But not all of them since $\mathbb{R}^2$ is not countable

Using any permutation invariant operation (such as max, min, kth largest value) allows to generalize other TDA approaches

$$\mathrm{PersLay}(\mathrm{D}) = \rho\left(\mathsf{op}\{w(p)\cdot\phi(p)\}_{p\in\mathrm{D}}\right)$$

Application to PDs

[*PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures*, C., Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]

Permutation invariant layers generalize several TDA approaches

$\rightarrow$ persistence images $\rightarrow$ landscapes $\rightarrow$ Betti curves

[*Time Series Classification via Topological Data Analysis*, Umeda, Trans. Jap. Soc. for AI, 2017]

But not all of them since $\mathbb{R}^2$ is not countable

Using any permutation invariant operation (such as max, min, kth largest value) allows to generalize other TDA approaches

$$\mathrm{PersLay}(\mathrm{D}) = \rho\left(\mathsf{op}\{w(p)\cdot\phi(p)\}_{p\in\mathrm{D}}\right)$$

Permutation-invariant operation

Application to PDs

[*PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures*, C., Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]

Permutation invariant layers generalize several TDA approaches

$\rightarrow$ persistence images $\rightarrow$ landscapes $\rightarrow$ Betti curves

[*Time Series Classification via Topological Data Analysis*, Umeda, Trans. Jap. Soc. for AI, 2017]

But not all of them since $\mathbb{R}^2$ is not countable

Using any permutation invariant operation (such as max, min, kth largest value) allows to generalize other TDA approaches

$$\mathrm{PersLay}(\mathrm{D}) = \rho\left(\mathsf{op}\{w(p) \cdot \phi(p)\}_{p \in \mathrm{D}}\right)$$

Permutation-invariant operation

Weight function

Application to PDs

[*PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures*, C., Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]

Permutation invariant layers generalize several TDA approaches

$\rightarrow$ persistence images $\rightarrow$ landscapes $\rightarrow$ Betti curves

[*Time Series Classification via Topological Data Analysis*, Umeda, Trans. Jap. Soc. for AI, 2017]

But not all of them since $\mathbb{R}^2$ is not countable

Using any permutation invariant operation (such as max, min, kth largest value) allows to generalize other TDA approaches

$$\mathrm{PersLay}(\mathrm{D}) = \rho\left(\mathsf{op}\{w(p)\cdot\phi(p)\}_{p\in\mathrm{D}}\right)$$

Permutation-invariant operation

Weight function

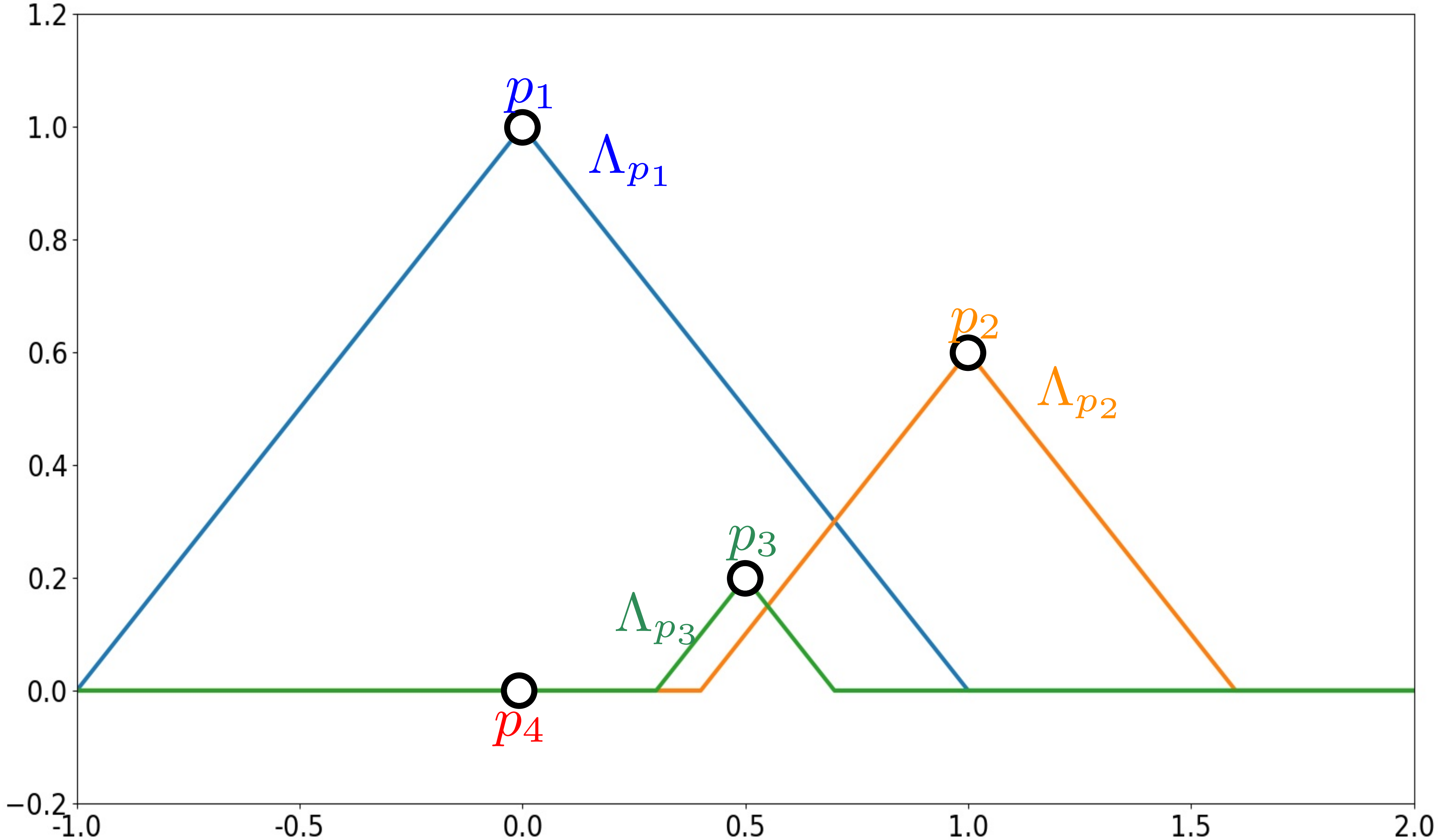
Point transformation

Application to PDs

[*PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures*, C., Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]

Parameters $t_1, \cdots, t_q \in \mathbb{R}$

$w(p) = 1$ $\quad\quad \phi_\Lambda : p \mapsto \begin{bmatrix} \Lambda_p(t_1) \\ \Lambda_p(t_2) \\ \vdots \\ \Lambda_p(t_q) \end{bmatrix}$ $\quad\quad$ op $=$ top-k

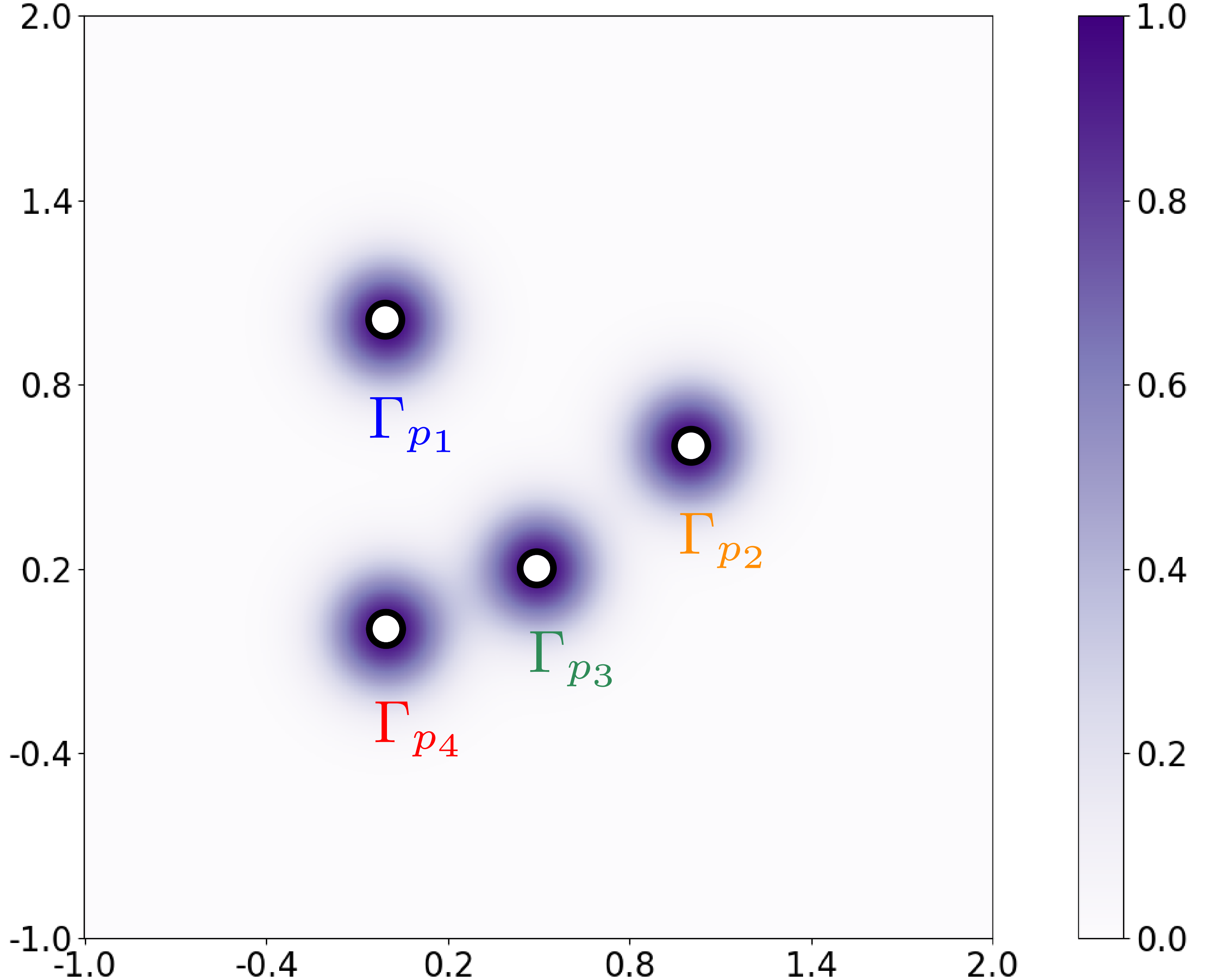


Application to PDs

[*PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures*, C., Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]

Parameters $t_1, \cdots, t_q \in \mathbb{R}^2$

$$w(p) = w_t((x,y)) \qquad \phi_\Gamma : p \mapsto \begin{bmatrix} \Gamma_p(t_1) \\ \Gamma_p(t_2) \\ \vdots \\ \Gamma_p(t_q) \end{bmatrix} \qquad \text{op} = \text{sum}$$



Application to PDs

[*PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures*, C., Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]

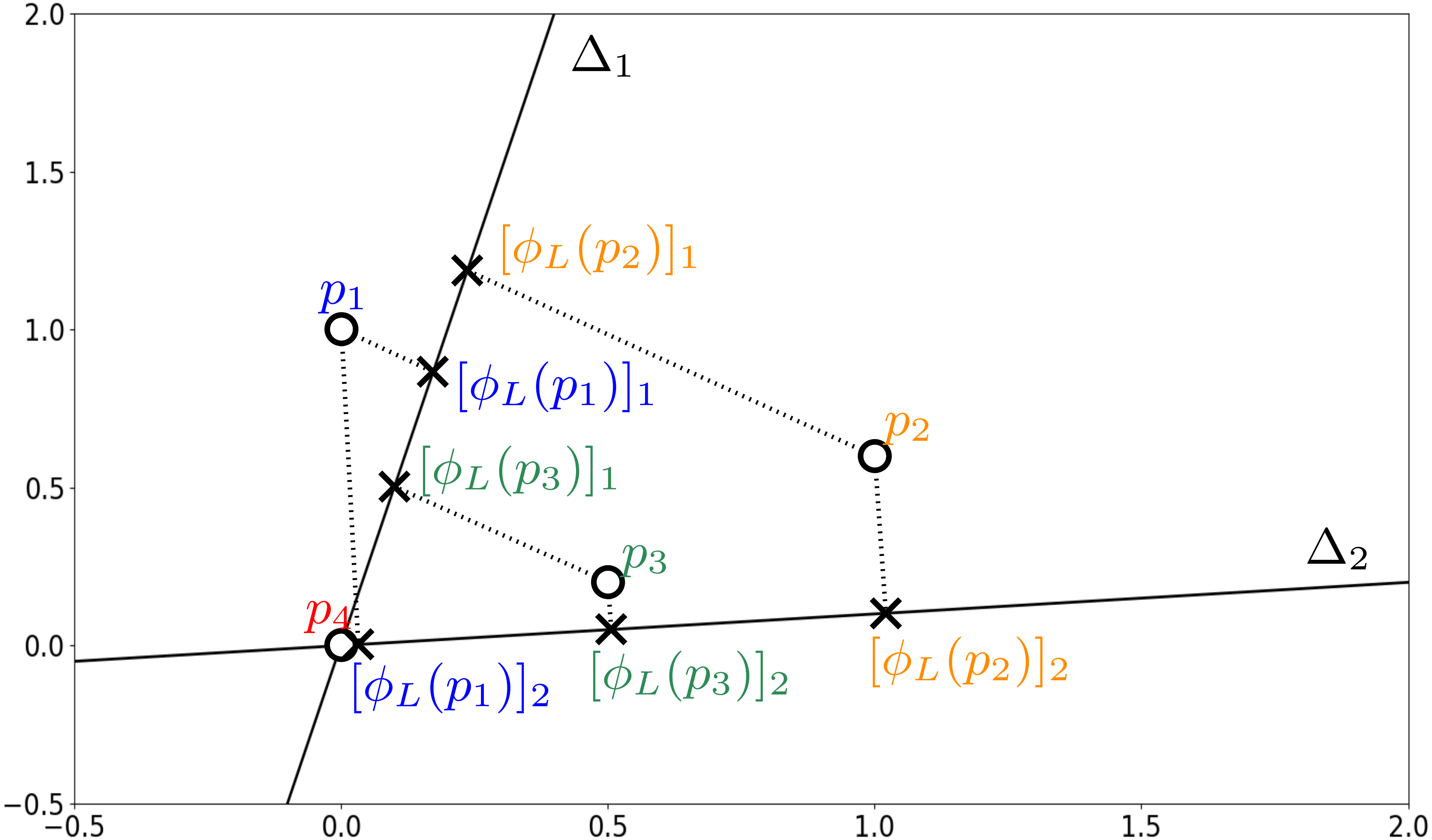
Parameters $\Delta_1, \cdots, \Delta_q \in [-\frac{\pi}{2}, \frac{\pi}{2}]$

$b_{\Delta_1}, \cdots, b_{\Delta_q} \in \mathbb{R}$

$$\phi_L : p \mapsto \begin{bmatrix} \langle p, e_{\Delta_1} \rangle + b_{\Delta_1} \\ \langle p, e_{\Delta_2} \rangle + b_{\Delta_2} \\ \vdots \\ \langle p, e_{\Delta_q} \rangle + b_{\Delta_q} \end{bmatrix}$$

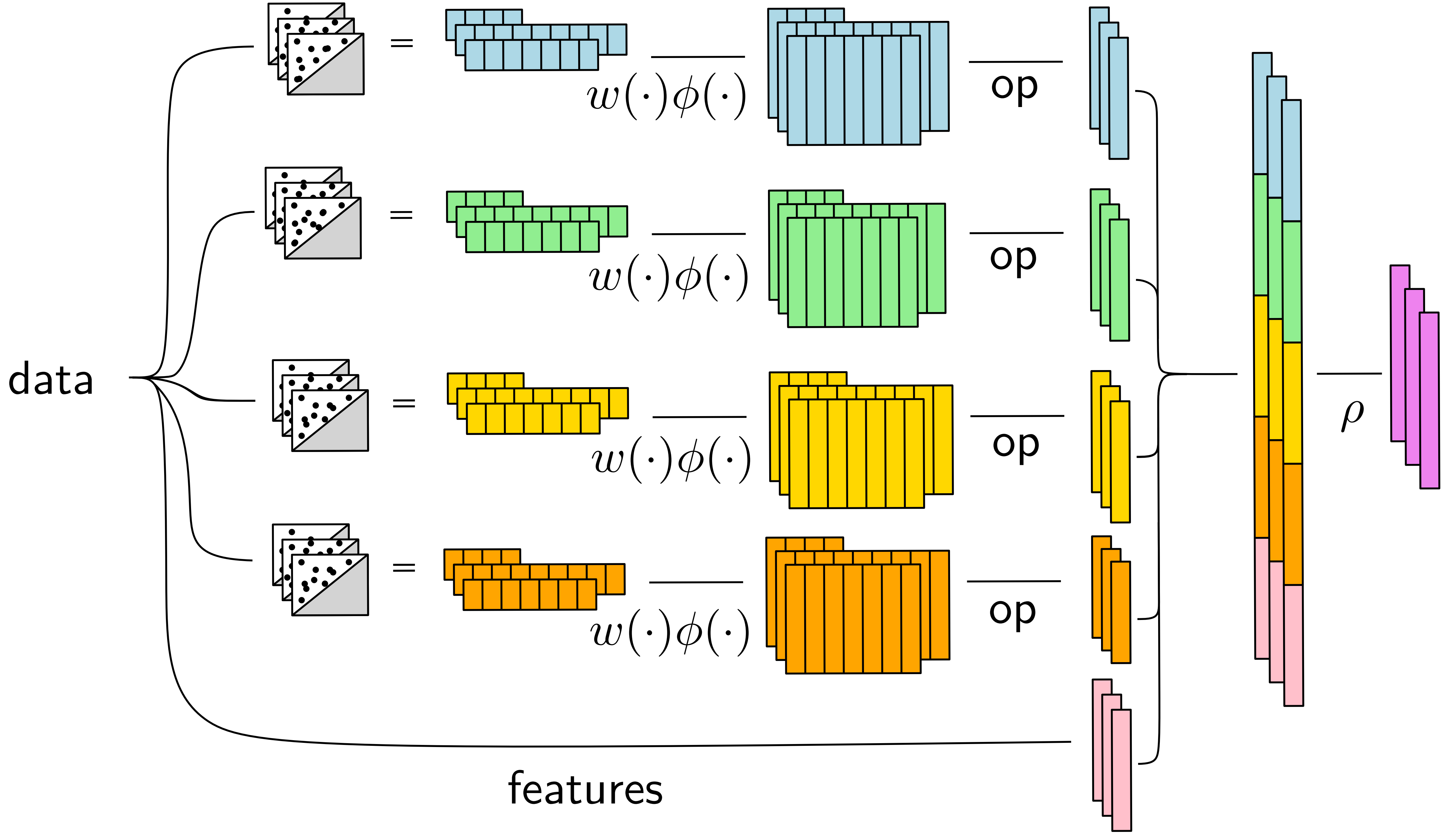
$w(p) = 1$

op $=$ top-k



Application to PDs

[*PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures*, C., Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]



Application to graph classification

[*PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures*, C., Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]

Let $G = (V, E)$ be a graph, A its adjacency matrix
D its degree matrix
and $L_w(G) = I - D^{-1/2}AD^{-1/2}$ its normalized Laplacian.

Application to graph classification

[*PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures*, C., Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]

Let $G = (V, E)$ be a graph, A its adjacency matrix

D its degree matrix

and $L_w(G) = I - D^{-1/2} A D^{-1/2}$ its normalized Laplacian.

$L_w(G)$ decomposes on a orthonormal basis $\phi_1, \dots, \phi_n$

with eigenvalues $0 \leq \lambda_1 \leq \cdots \leq \lambda_n \leq 2$

Application to graph classification

[*PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures*, C., Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]

Let $G = (V, E)$ be a graph, A its adjacency matrix

D its degree matrix

and $L_w(G) = I - D^{-1/2} A D^{-1/2}$ its normalized Laplacian.

$L_w(G)$ decomposes on a orthonormal basis $\phi_1, \ldots, \phi_n$

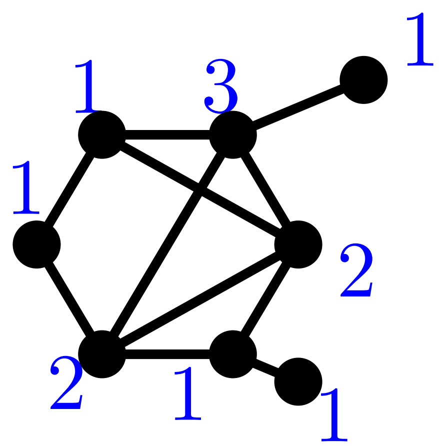
with eigenvalues $0 \leq \lambda_1 \leq \cdots \leq \lambda_n \leq 2$

Def: Let $t \geq 0$, and define the *Heat Kernel Signature* of param t:

$$\mathrm{hks}_{G,t} : v \mapsto \sum_{k=1}^{n} \exp(-\lambda_k t) \phi_k(v)^2$$

Application to graph classification

[*PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures*, C., Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]

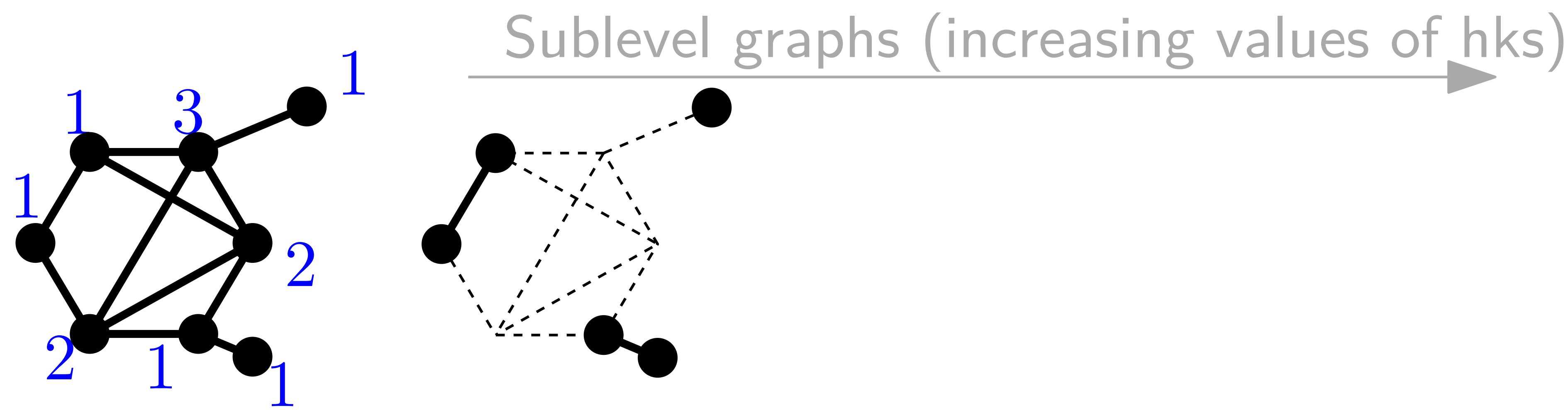


Def: Let $t \geq 0$, and define the *Heat Kernel Signature* of param t:

$$\mathrm{hks}_{G,t} : v \mapsto \sum_{k=1}^{n} \exp(-\lambda_k t)\phi_k(v)^2$$

Application to graph classification

[*PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures*, C., Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]

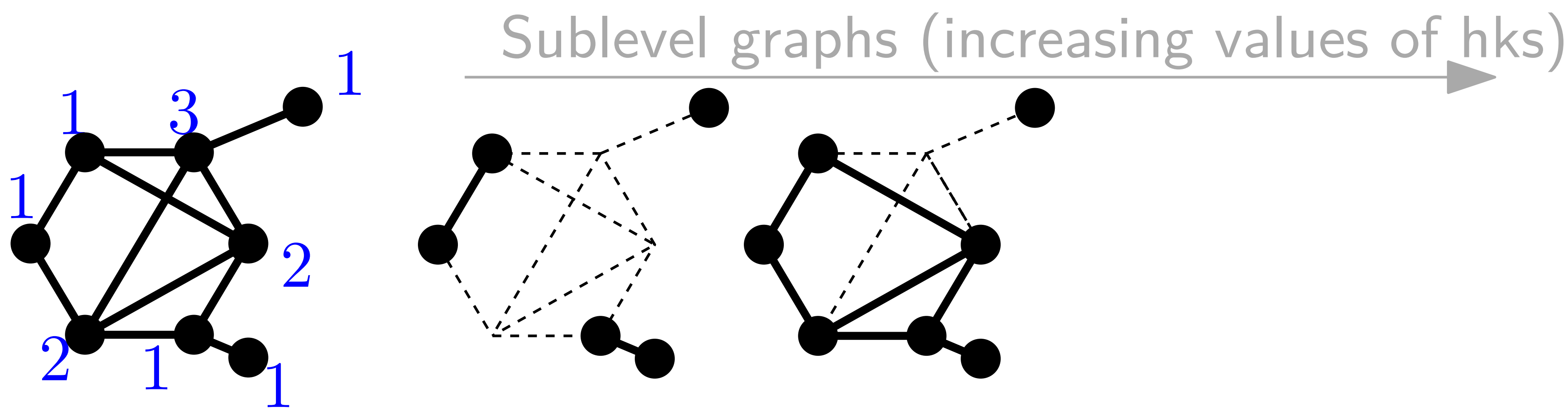


Def: Let $t \geq 0$, and define the *Heat Kernel Signature* of param t:

$$\mathrm{hks}_{G,t} : v \mapsto \sum_{k=1}^{n} \exp(-\lambda_k t)\phi_k(v)^2$$

Application to graph classification

[*PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures*, C., Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]

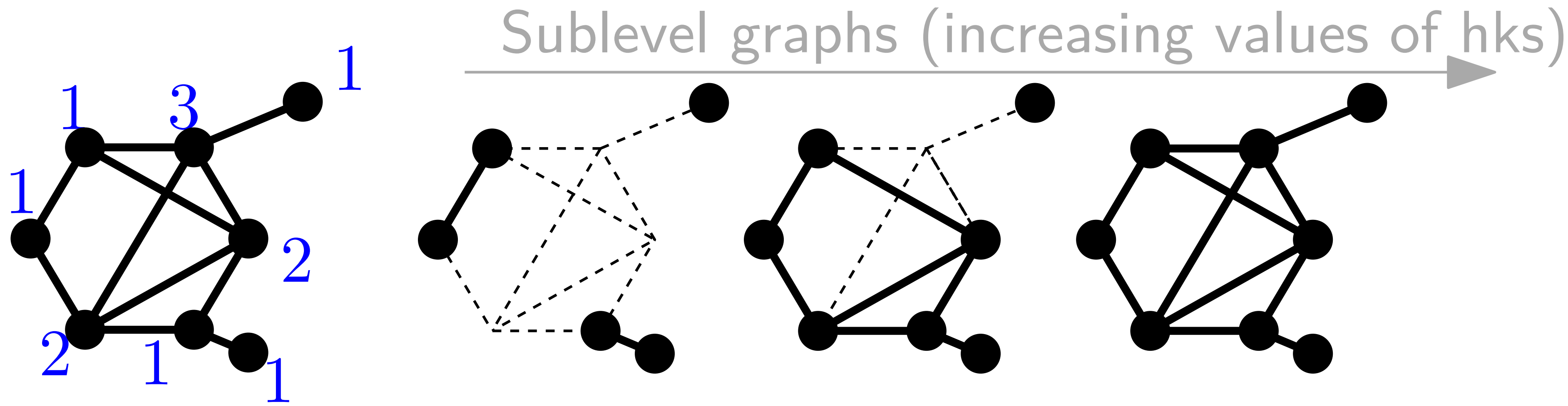


Def: Let $t \geq 0$, and define the *Heat Kernel Signature* of param t:

$$\mathrm{hks}_{G,t} : v \mapsto \sum_{k=1}^{n} \exp(-\lambda_k t)\phi_k(v)^2$$

Application to graph classification

[*PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures*, C., Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]

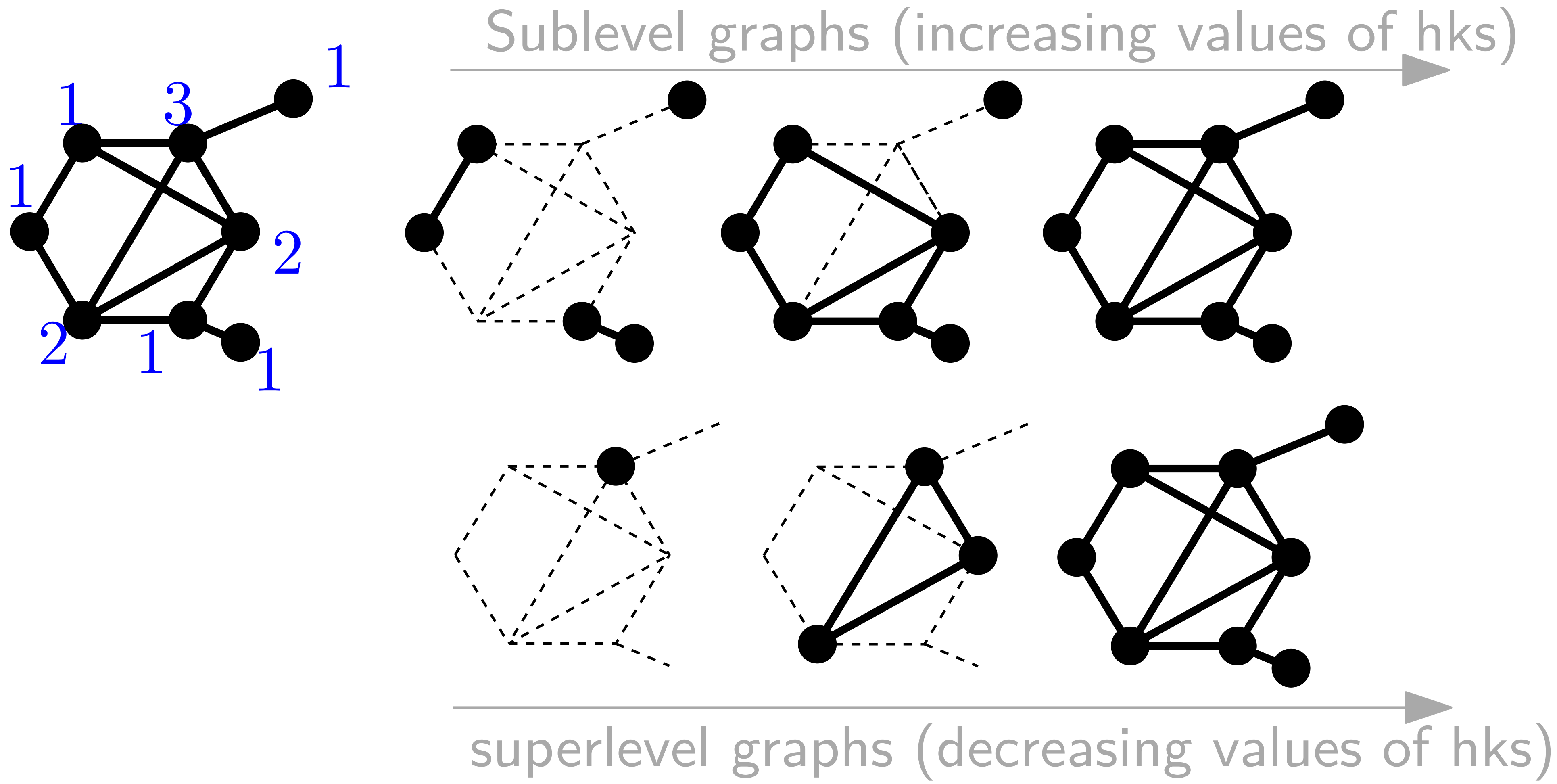


Def: Let $t \geq 0$, and define the *Heat Kernel Signature* of param t:

$$\mathrm{hks}_{G,t} : v \mapsto \sum_{k=1}^{n} \exp(-\lambda_k t)\phi_k(v)^2$$

Application to graph classification

[*PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures*, C., Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]



Def: Let $t \geq 0$, and define the *Heat Kernel Signature* of param t:

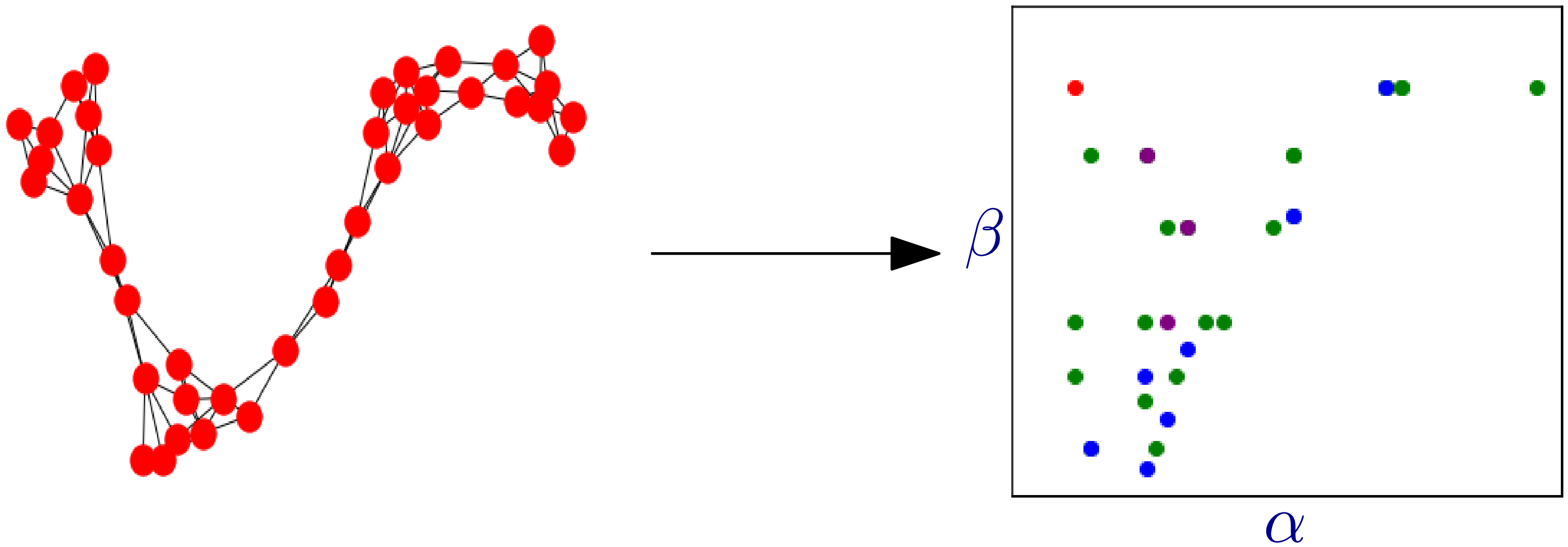
$$\mathrm{hks}_{G,t} : v \mapsto \sum_{k=1}^{n} \exp(-\lambda_k t)\phi_k(v)^2$$

Application to graph classification

[*PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures*, C., Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]

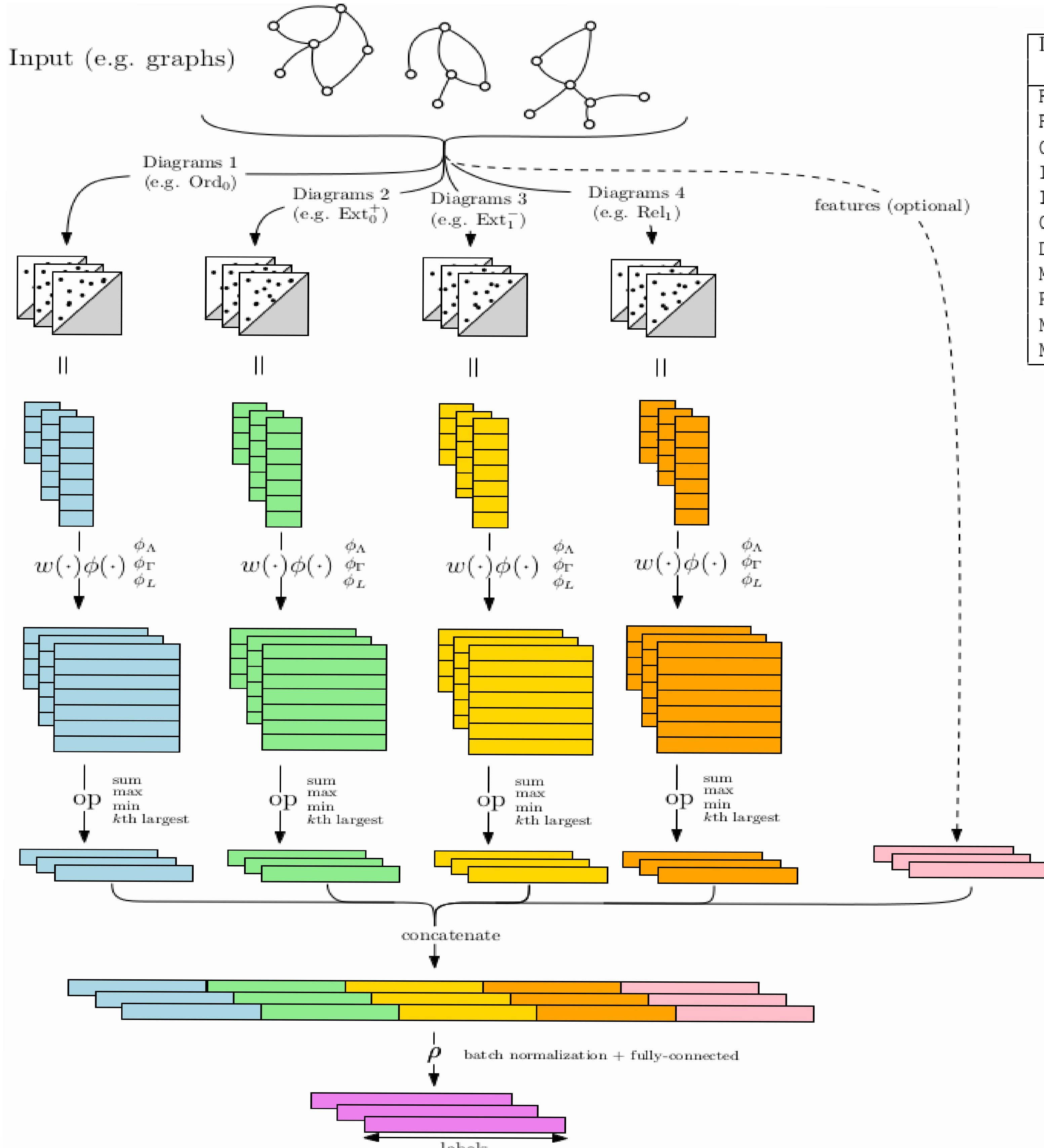
Graph from the `PROTEINS` dataset

Corresponding extended persistence diagram



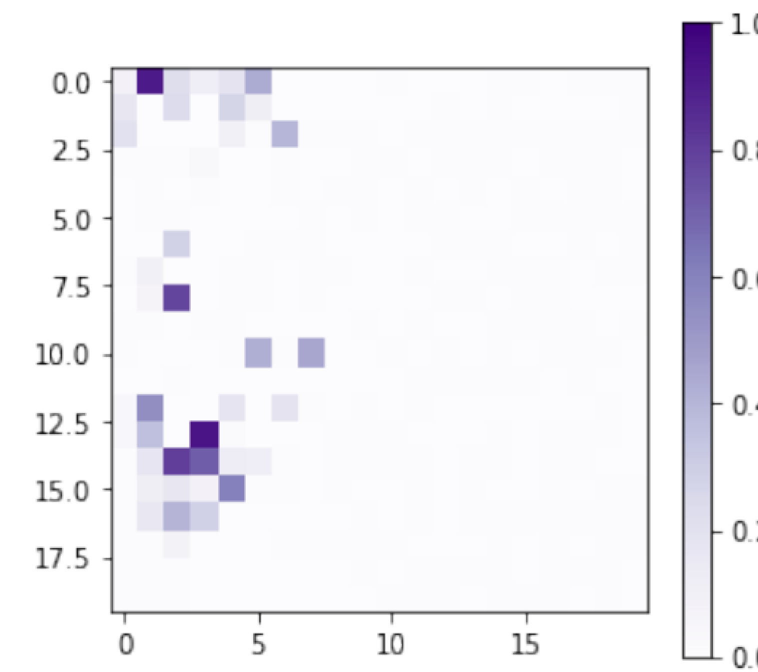
Application to graph classification

[*PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures*, C., Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]



Dataset	SV[1]	RetGK* [2]	FGSD [3]	GCNN [4]	GIN [5]	PersLay Mean	PersLay Max
REDDIT5K	—	56.1	47.8	52.9	57.0	55.6	56.5
REDDIT12K	—	48.7	—	46.6	—	47.7	49.1
COLLAB	—	81.0	80.0	79.6	80.1	76.4	78.0
IMDB-B	72.9	71.9	73.6	73.1	74.3	71.2	72.6
IMDB-M	50.3	47.7	52.4	50.3	52.1	48.8	52.2
COX2*	78.4	80.1	—	—	—	80.9	81.6
DHFR*	78.4	81.5	—	—	—	80.3	80.9
MUTAG*	88.3	90.3	92.1	86.7	89.0	89.8	91.5
PROTEINS*	72.6	75.8	73.4	76.3	75.9	74.8	75.9
NCI1*	71.6	84.5	79.8	78.4	82.7	73.5	74.0
NCI109*	70.5	—	78.8	—	—	69.5	70.1

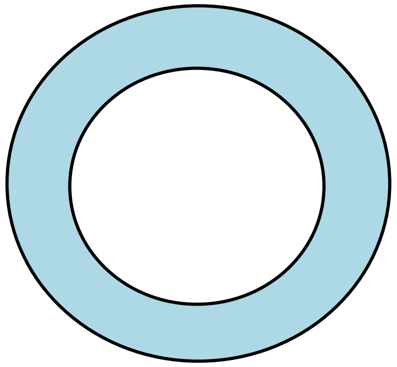
Weight function learnt



(after training on the `MUTAG` dataset)

Persistence Diagrams and Statistics

Statistics on Persistence Diagrams

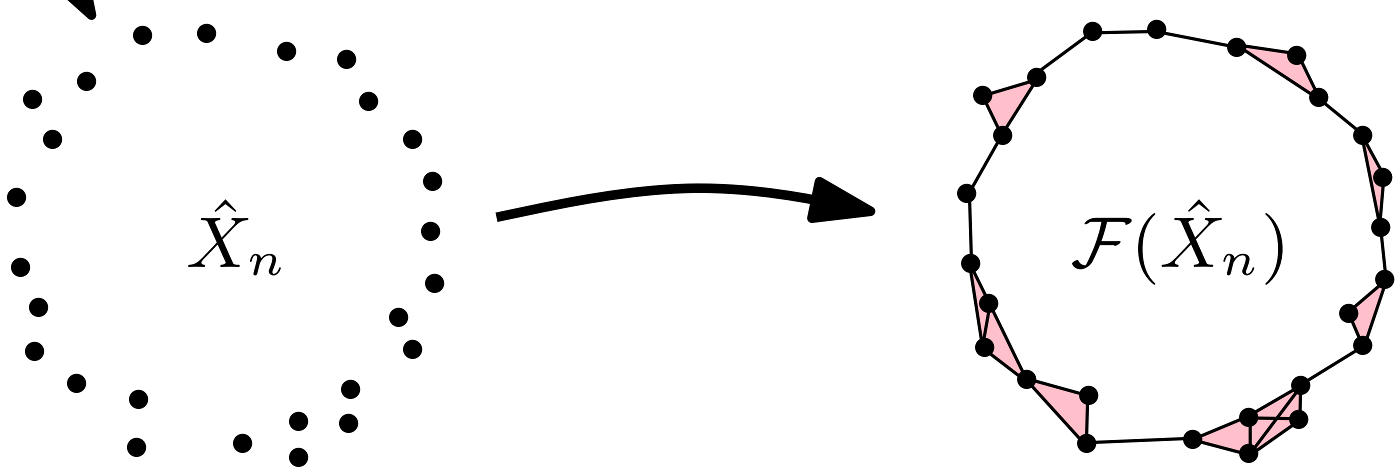


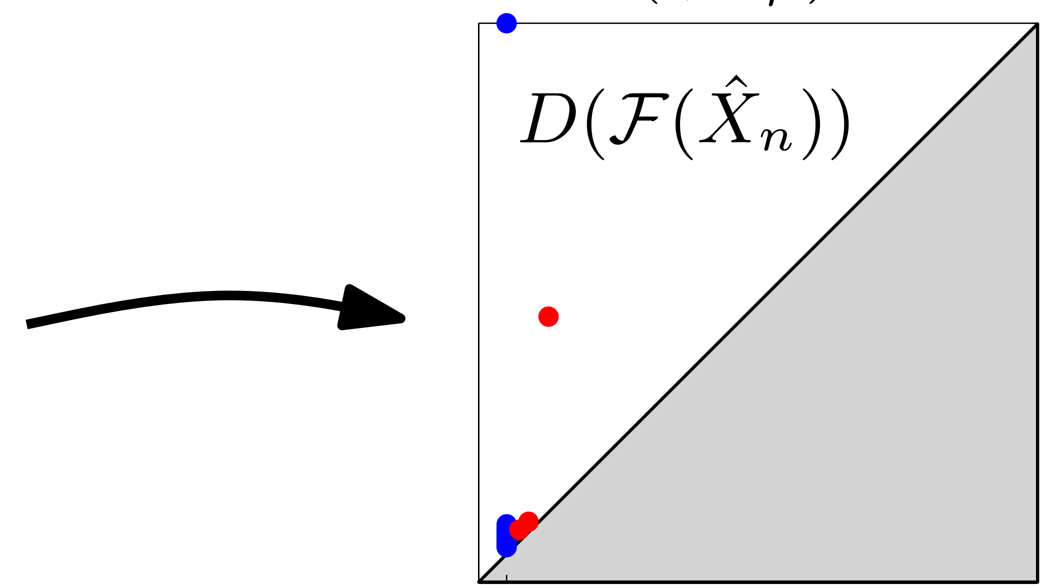
(X, d) metric space

μ probability measure with compact support X_μ

Examples:
- $\mathcal{F}(\hat{X}_n) = \mathrm{Rips}(\hat{X}_n)$
- $\mathcal{F}(\hat{X}_n) = \check{\mathrm{C}}\mathrm{ech}(\hat{X}_n)$
- $\mathcal{F}(\hat{X}_n) =$ sublevelset filtration of $d(., X_\mu)$.

Sample n points according to μ.

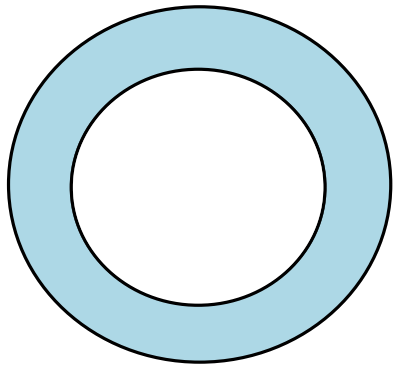




Questions:

- Statistical properties of $D(\mathcal{F}(\hat{X}_n))$? $D(\mathcal{F}(\hat{X}_n)) \to ?$ as $n \to +\infty$?

Statistics on Persistence Diagrams



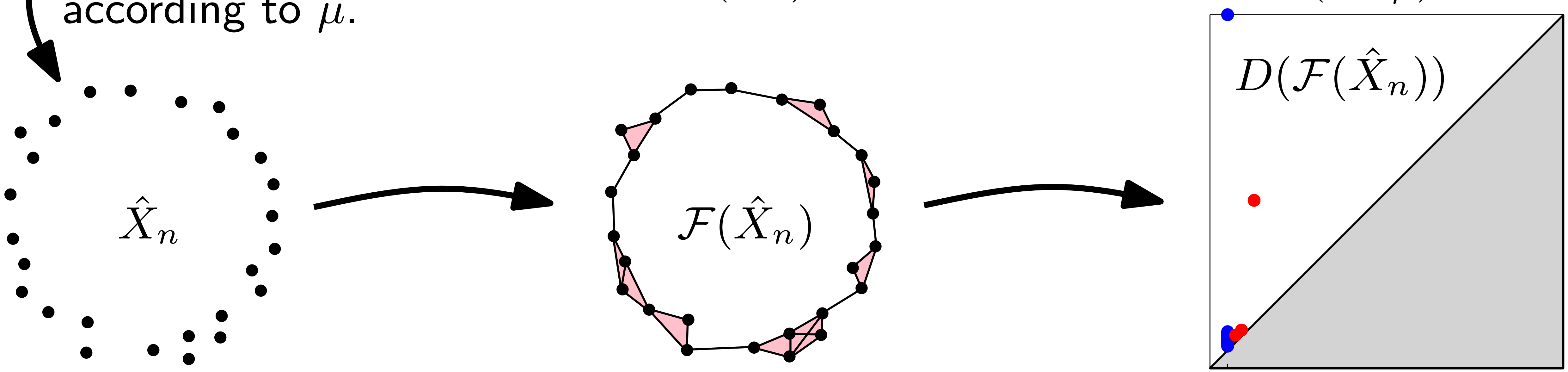
(X, d) metric space

μ probability measure with compact support X_μ

Examples:

- $\mathcal{F}(\hat{X}_n) = \mathrm{Rips}(\hat{X}_n)$
- $\mathcal{F}(\hat{X}_n) = \check{\mathrm{C}}\mathrm{ech}(\hat{X}_n)$
- $\mathcal{F}(\hat{X}_n) =$ sublevelset filtration of $d(., X_\mu)$.

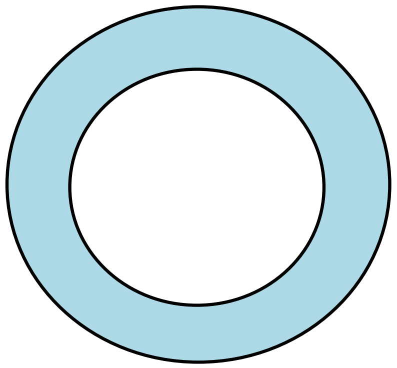
Sample n points according to μ.



Questions:

- Statistical properties of $D(\mathcal{F}(\hat{X}_n))$? $D(\mathcal{F}(\hat{X}_n)) \to ?$ as $n \to +\infty$?
- Can we do more statistics with persistence diagrams?

Statistics on Persistence Diagrams

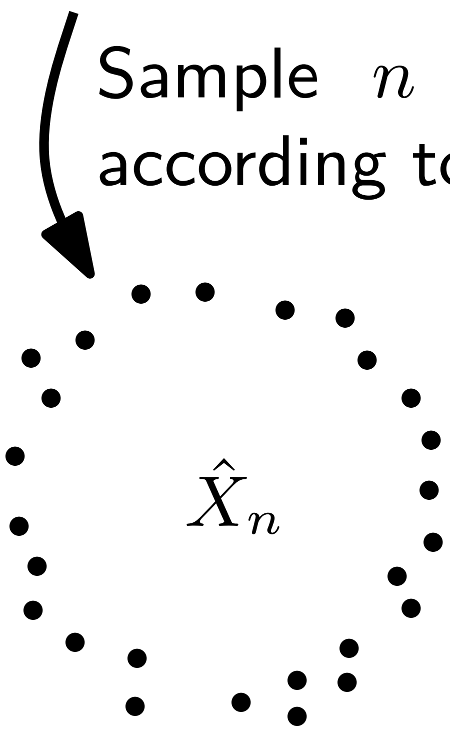


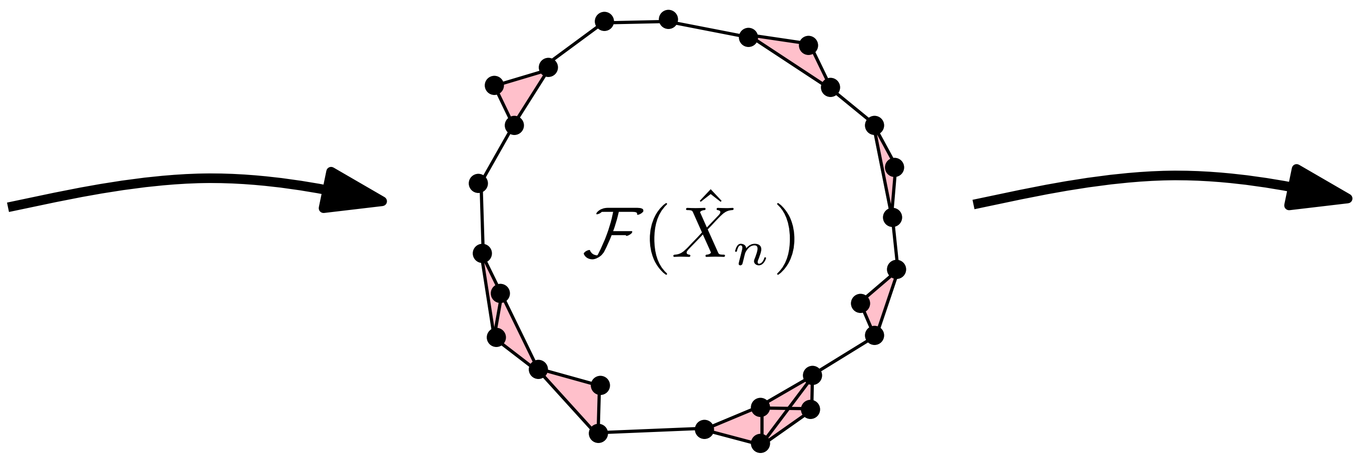
(X, d) metric space

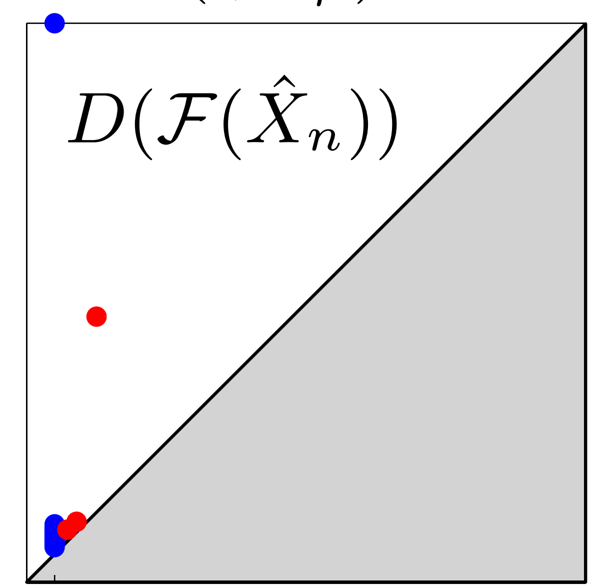
μ probability measure with compact support X_μ

Examples:
- $\mathcal{F}(\hat{X}_n) = \mathrm{Rips}(\hat{X}_n)$
- $\mathcal{F}(\hat{X}_n) = \check{\mathrm{C}}\mathrm{ech}(\hat{X}_n)$
- $\mathcal{F}(\hat{X}_n) =$ sublevelset filtration of $d(., X_\mu)$.

Sample n points according to μ.







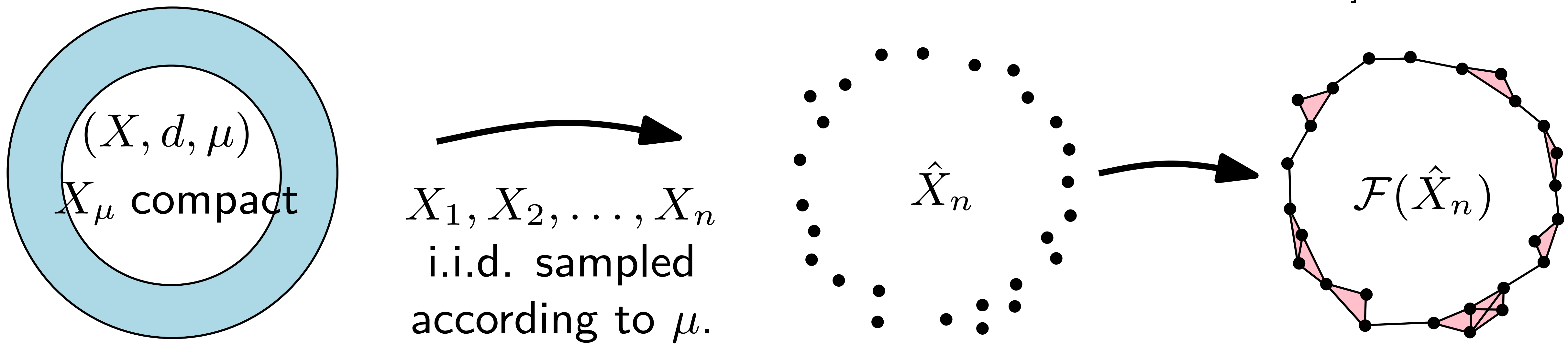
Stability thm: $d_B(D(\mathcal{F}(X_\mu)), D(\mathcal{F}(\hat{X}_n))) \leq 2 d_{GH}(X_\mu, \hat{X}_n)$

So, for any $\varepsilon > 0$,

$$\mathbb{P}\left(d_B\left(D(\mathcal{F}(X_\mu)), D(\mathcal{F}(\hat{X}_n))\right) > \varepsilon\right) \leq \mathbb{P}\left(d_{GH}(X_\mu, \hat{X}_n) > \frac{\varepsilon}{2}\right)$$

Deviation inequality

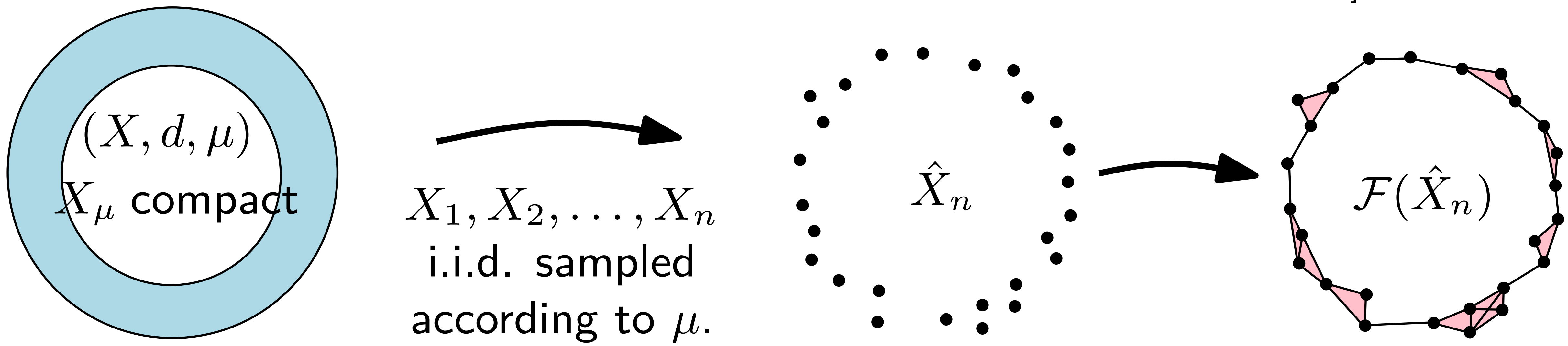
[*Convergence rates for persistence diagram estimation in Topological Data Analysis*, Chazal, Glisse, Labruère, Michel ICML, 2014]



For $a, b > 0$, μ satisfies the (a, b)-standard assumption if for any $x \in X_\mu$ and any $r > 0$, we have $\mu(B(x, r)) \geq \min(ar^b, 1)$.

Deviation inequality

[*Convergence rates for persistence diagram estimation in Topological Data Analysis*, Chazal, Glisse, Labruère, Michel ICML, 2014]



For $a, b > 0$, μ satisfies the (a, b)-standard assumption if for any $x \in X_\mu$ and any $r > 0$, we have $\mu(B(x, r)) \geq \min(ar^b, 1)$.

Thm: If μ satisfies the (a, b)-standard assumption, then for any $\varepsilon > 0$:

$$\mathbb{P}\left(d_B\left(D(\mathcal{F}(X_\mu)), D(\mathcal{F}(\hat{X}_n))\right) > \varepsilon\right) \leq \min\left\{\frac{8^b}{a\varepsilon^b}\exp\left(-na\varepsilon^b\right), 1\right\}.$$

Minimax rate of convergence

[*Convergence rates for persistence diagram estimation in Topological Data Analysis*, Chazal, Glisse, Labruère, Michel ICML, 2014]

Let $\mathcal{P}(a, b, X)$ be the set of all the probability measures on the metric space (X, d) satisfying the (a, b)-standard assumption on X:

Minimax rate of convergence

[*Convergence rates for persistence diagram estimation in Topological Data Analysis*, Chazal, Glisse, Labruère, Michel ICML, 2014]

Let $\mathcal{P}(a, b, X)$ be the set of all the probability measures on the metric space (X, d) satisfying the (a, b)-standard assumption on X:

Thm: Let $\mathcal{P}(a, b, X)$ be the set of (a, b)-standard proba measures on X. Then:

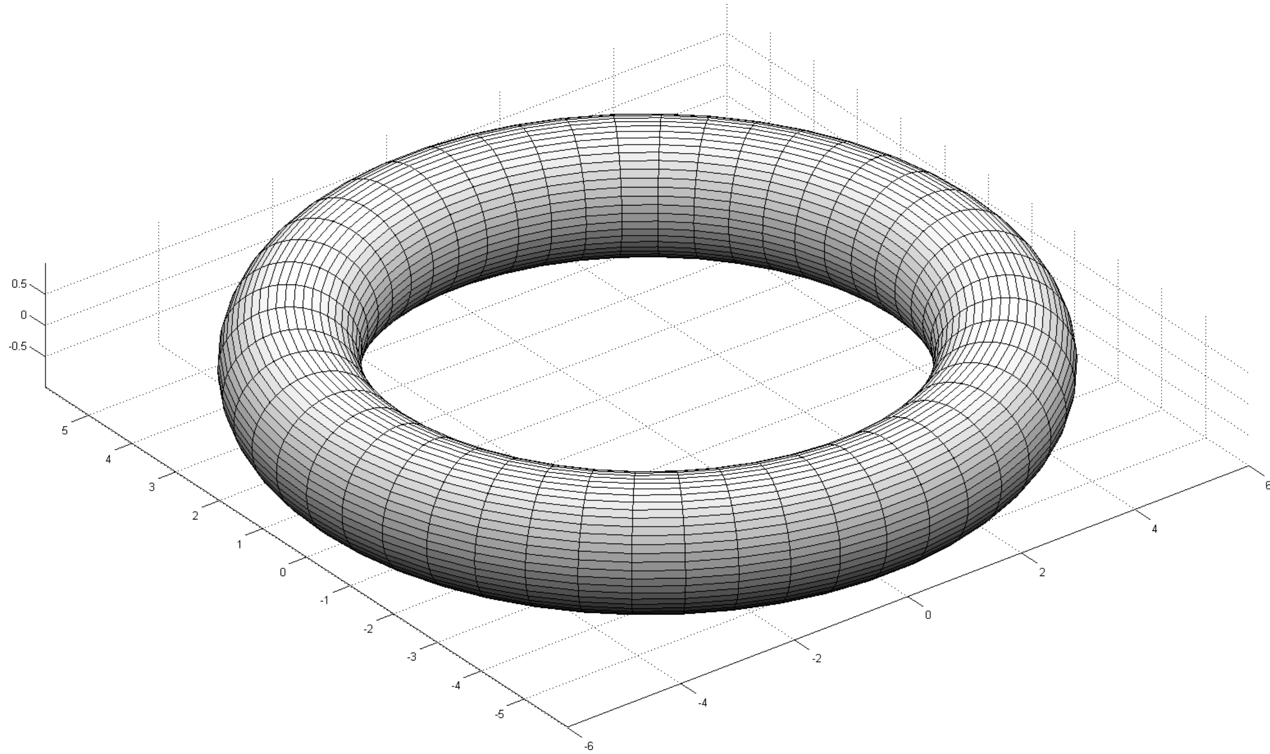
$$\sup_{\mu \in \mathcal{P}(a,b,X)} \mathbb{E}\left[d_B(D(\mathcal{F}(X_\mu)), D(\mathcal{F}(\hat{X}_n)))\right] \leq C \left(\frac{\log n}{n}\right)^{1/b}$$

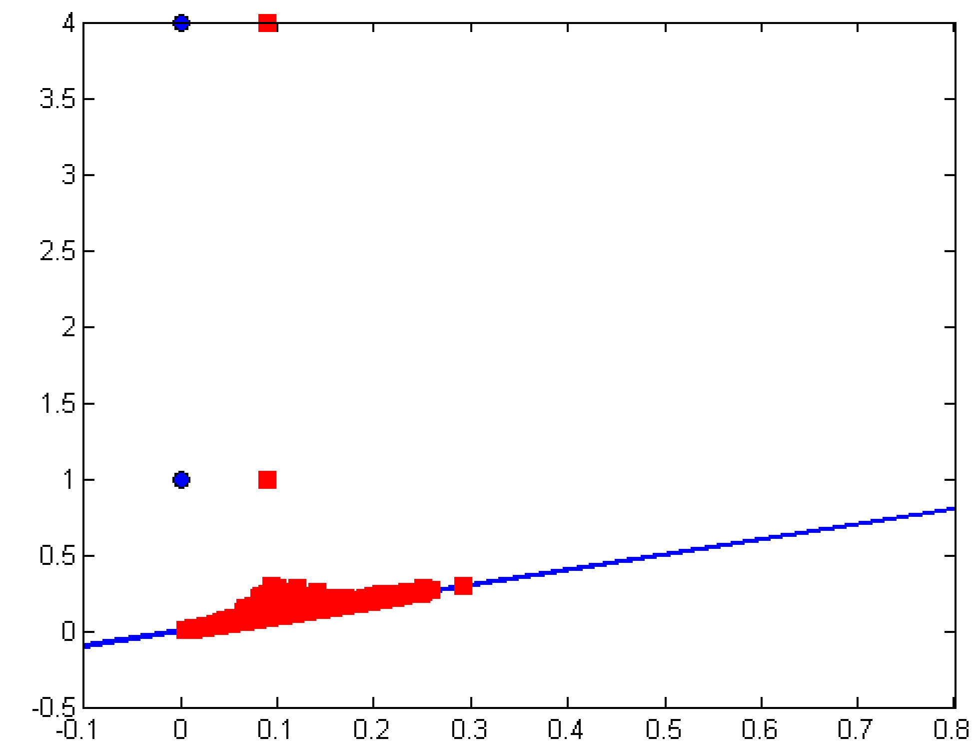
where the constant C only depends on a and b (not on X!).

Rem: we can obtain slightly better bounds if X_μ is a submanifold of $\mathbb{R}^D$.

Numerical illustrations

[*Convergence rates for persistence diagram estimation in Topological Data Analysis*, Chazal, Glisse, Labruère, Michel ICML, 2014]

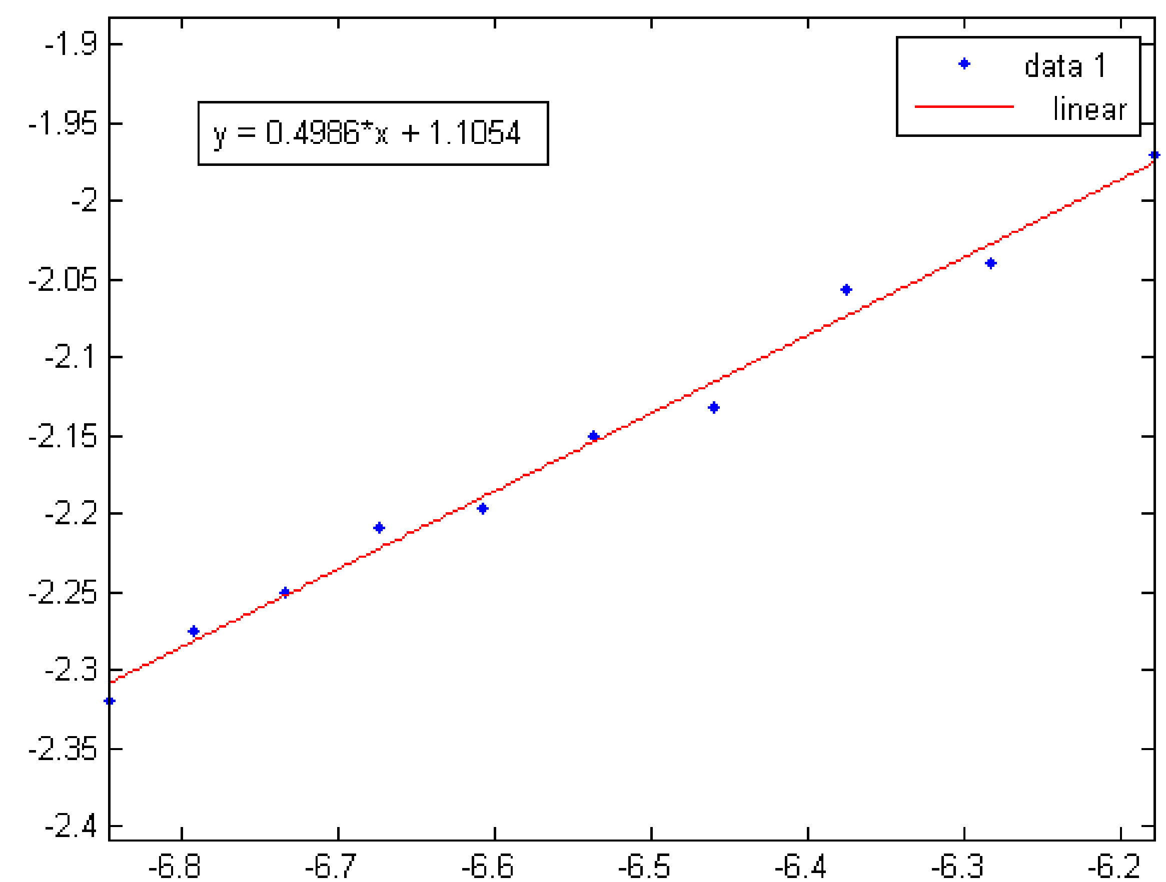




- μ: unif. measure on a torus X_μ.
- $\mathcal{F}$: distance to X_μ in $\mathbb{R}^3$.
- sample $k = 300$ sets of n points for $n = [12000 : 1000 : 21000]$.
- compute

$$\hat{\mathbb{E}}_n = \hat{\mathbb{E}}[d_B(D(\mathcal{F}(X_\mu)), D(\mathcal{F}(\hat{X}_n)))].$$

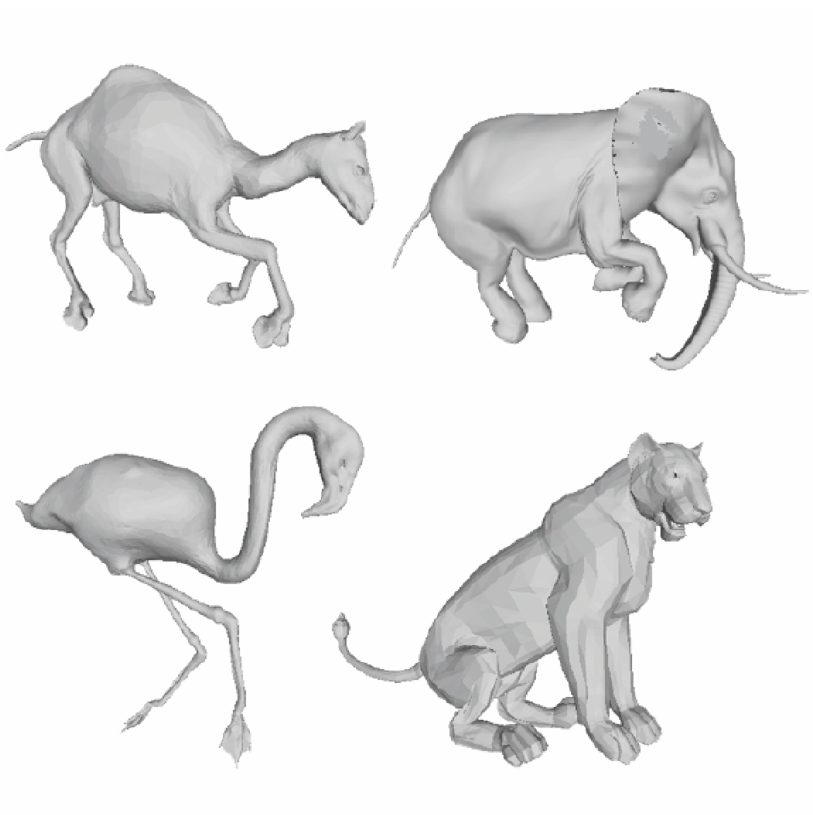
- plot $\log(\hat{\mathbb{E}}_n)$ as a function of $\log(\log(n)/n)$.

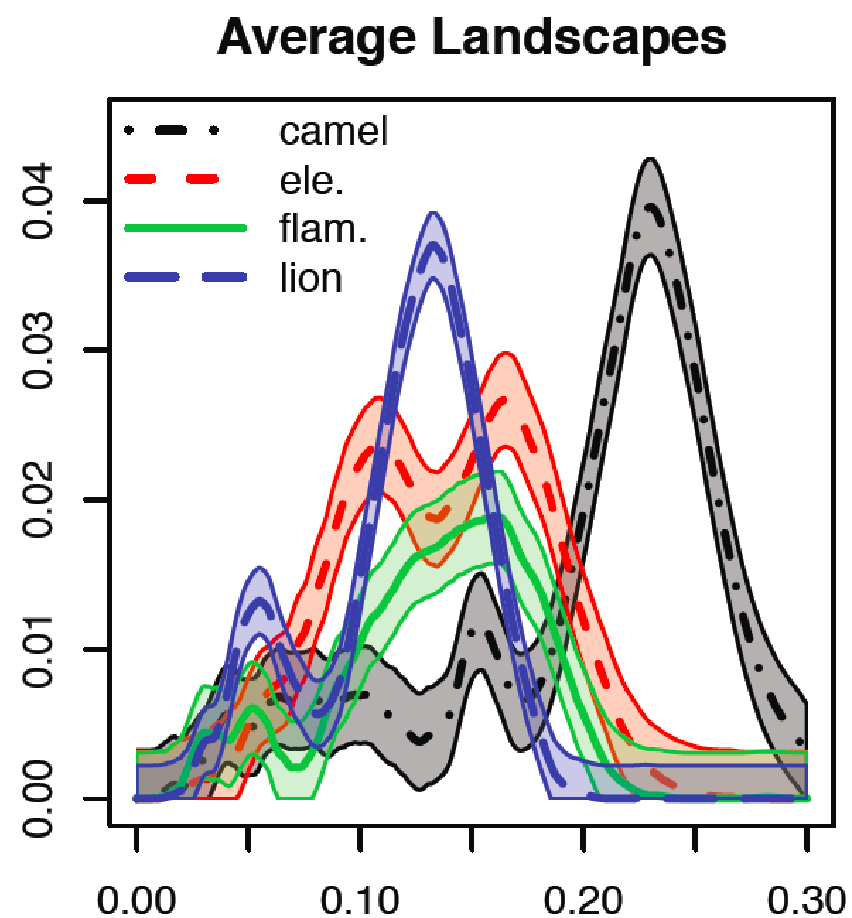


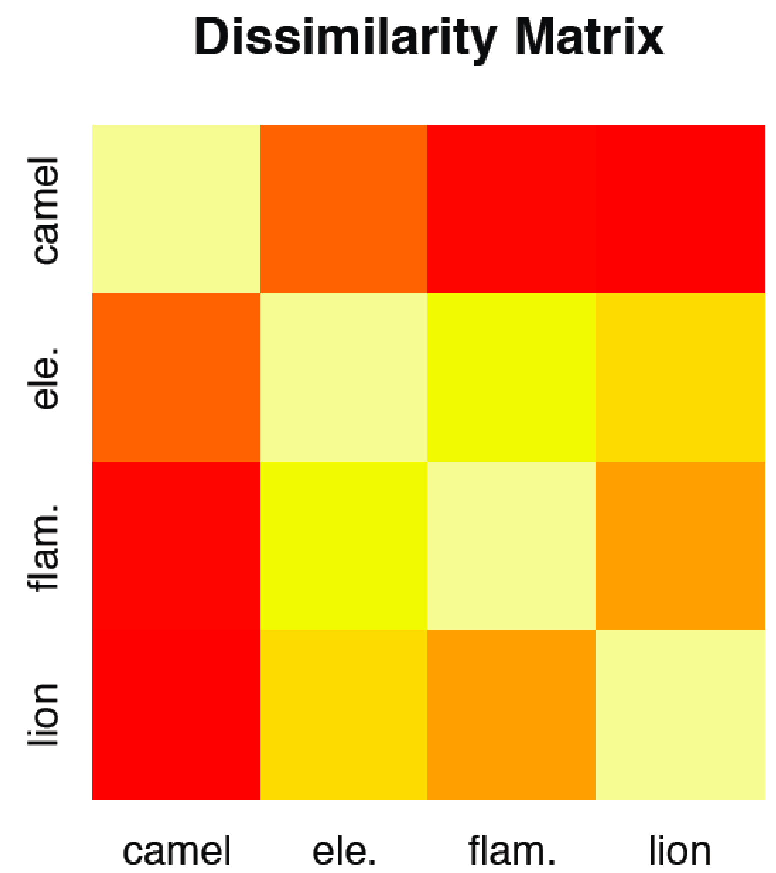
Numerical illustrations: confidence for landscapes

[*On the Bootstrap for Persistence Diagrams and Landscapes*, Chazalet al., Model. Anal. Inform. Sist., 2013]

Example: 3D shapes





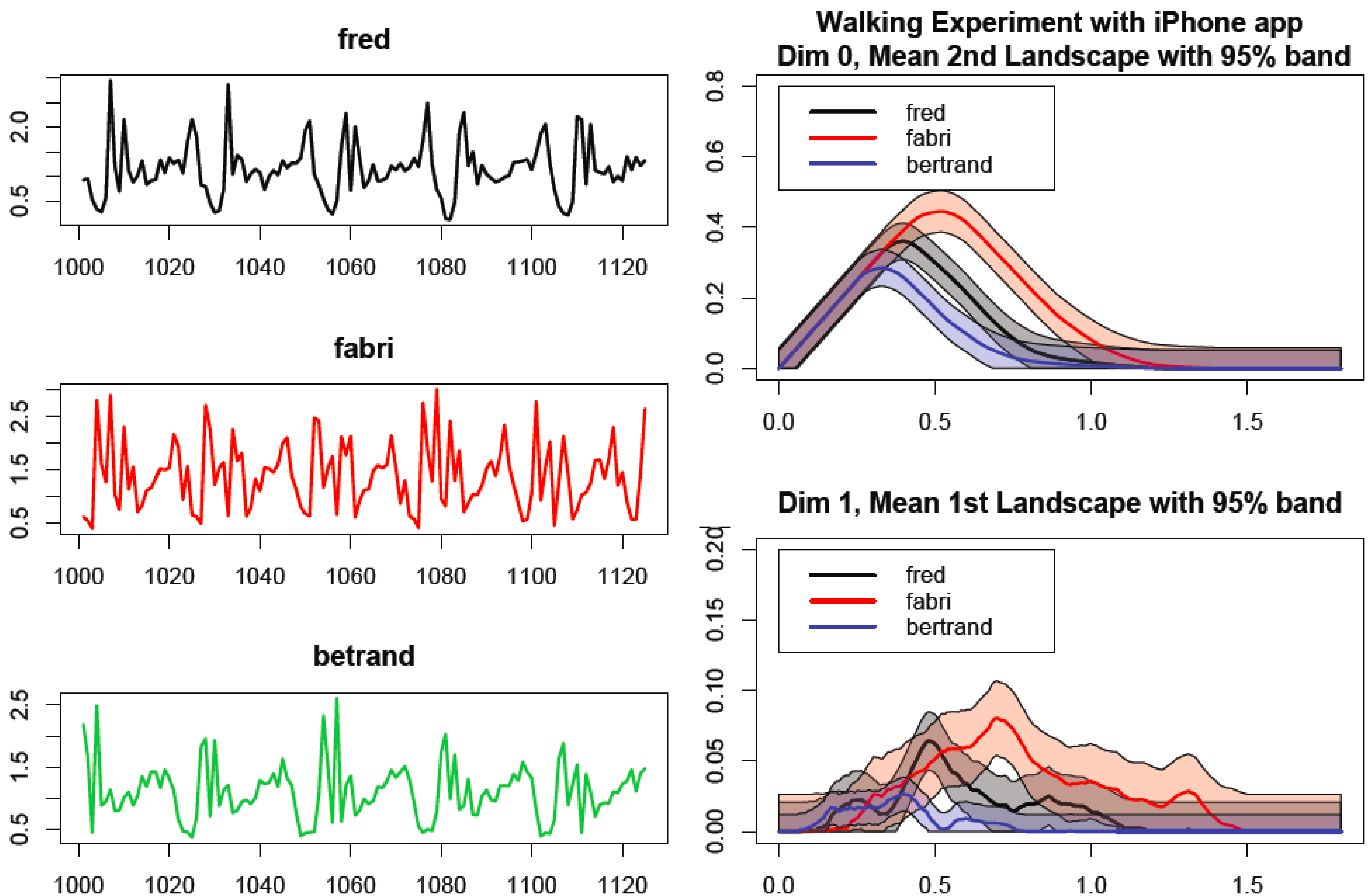


From $k = 100$ subsamples of size $n = 300$

Numerical illustrations: confidence for landscapes

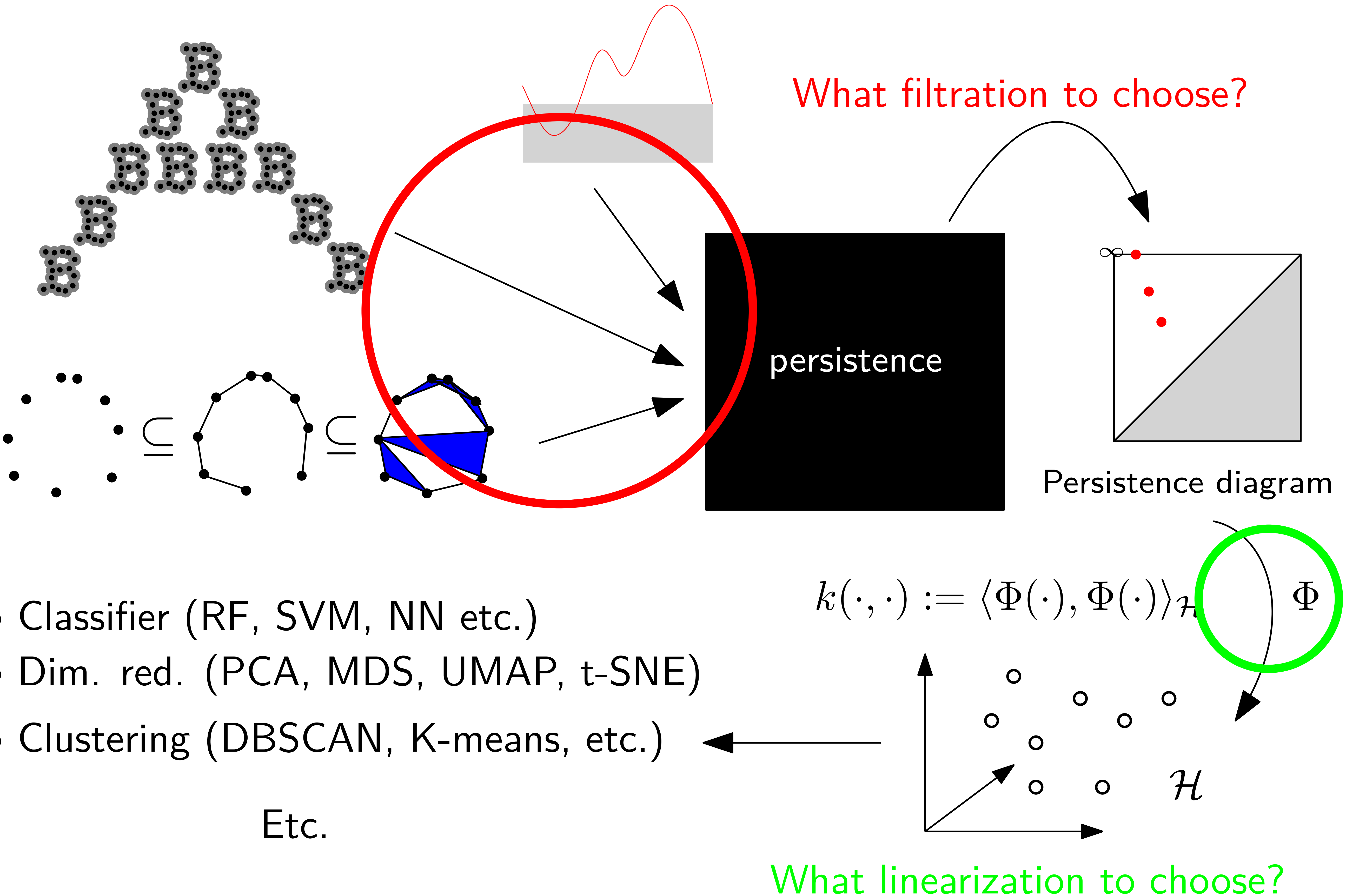
(Toy) Example: Accelerometer data from smartphone.

[*On the Bootstrap for Persistence Diagrams and Landscapes*, Chazalet al., Model. Anal. Inform. Sist., 2013]



Persistence Diagrams and Optimization

Persistence diagrams and machine learning



- Classifier (RF, SVM, NN etc.)
- Dim. red. (PCA, MDS, UMAP, t-SNE)
- Clustering (DBSCAN, K-means, etc.)

Etc.

Problem setting

Q: How to define ∇D?

Problem setting

Q: How to define ∇D?

Q: Given a parameterized family of functions $\mathcal{F} = \{f_\theta \ : \ \theta \in \Theta\}$, how to define $\nabla_\theta D_{f_\theta}$?

Q: Given a point cloud $X \subseteq \mathbb{R}^d$, how to define $\nabla_X D_{\mathrm{Rips}}(X)$?

Problem setting

Q: How to define ∇D?

Q: Given a parameterized family of functions $\mathcal{F} = \{f_\theta : \theta \in \Theta\}$, how to define $\nabla_\theta D_{f_\theta}$?

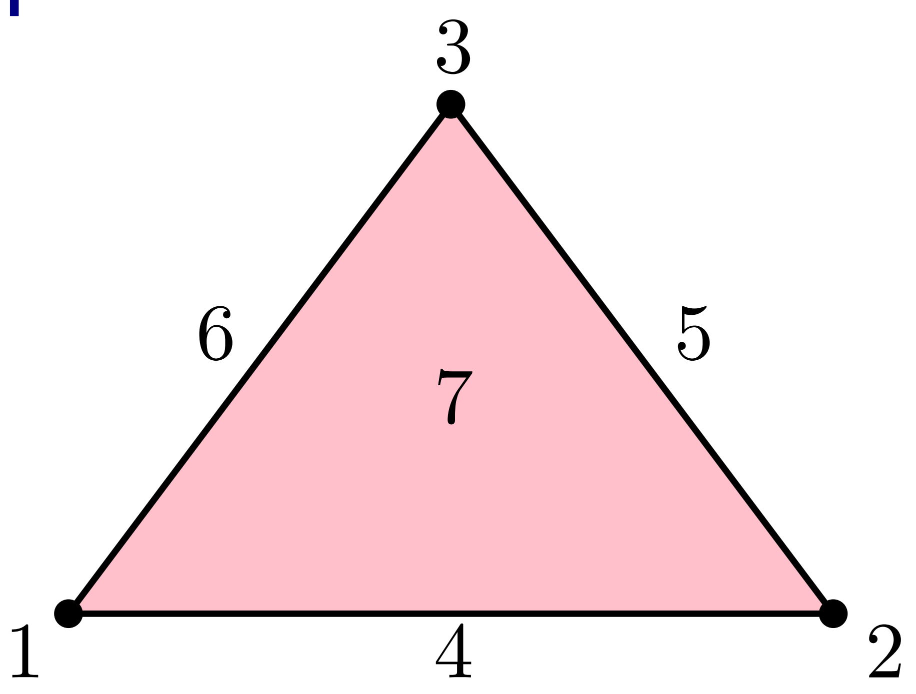
Q: Given a point cloud $X \subseteq \mathbb{R}^d$, how to define $\nabla_X D_{\mathrm{Rips}}(X)$?

Idea: Let's go back to the PD construction...

Computation with matrix reduction

Input: simplicial filtration

Output: boundary matrix

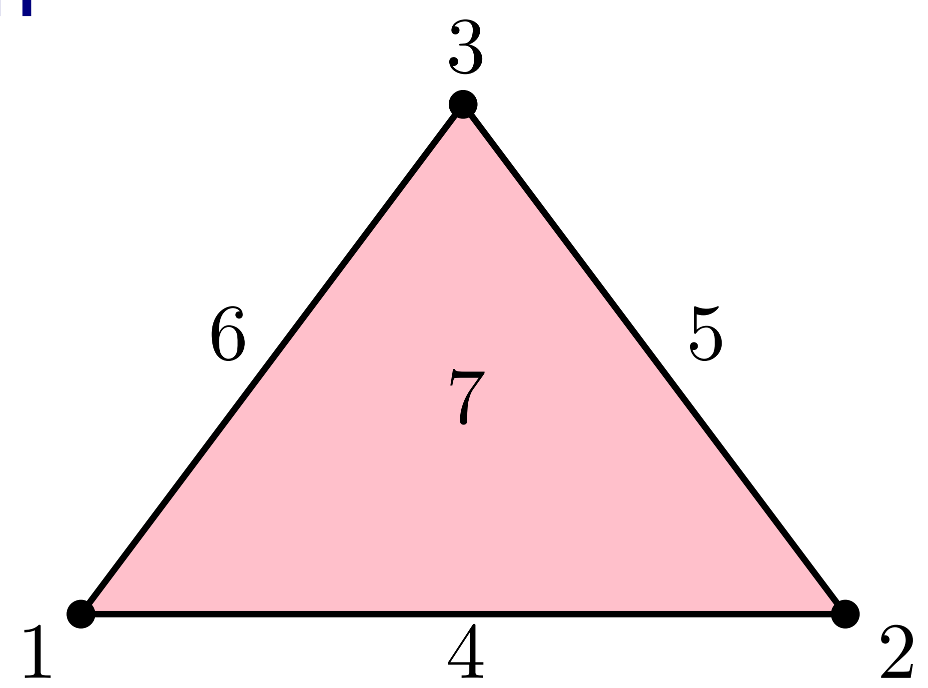


	1	2	3	4	5	6	7
1				*		*	
2				*	*		
3					*	*	
4							*
5							*
6							*
7							

Computation with matrix reduction

Input: simplicial filtration

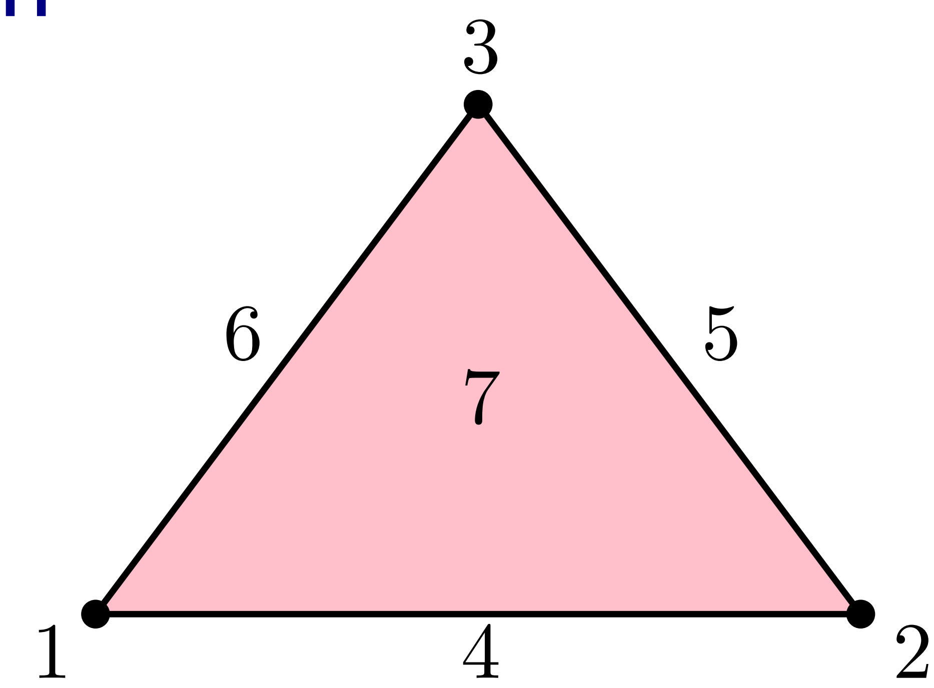
Output: boundary matrix
reduced to column-echelon form



	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 matrix reduction



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: $[1,+\infty)$

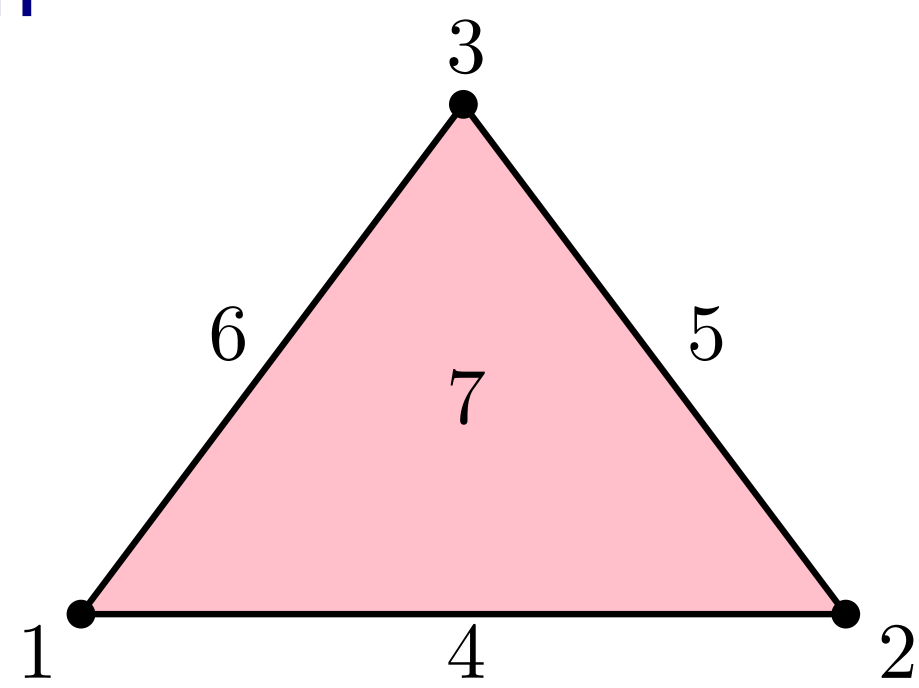
	1	2	3	4	5	6	7
1				*		*	
2				*	*		
3					*	*	
4							*
5							*
6							*
7							

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

Computation with matrix reduction

Input: simplicial filtration

Output: boundary matrix
reduced to column-echelon form



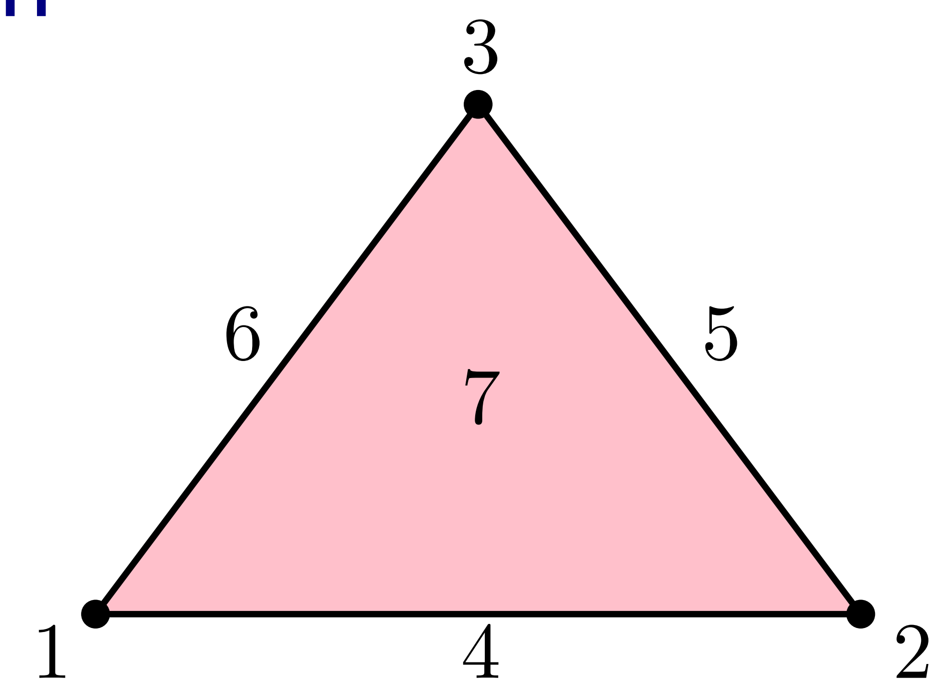
○ simplex pairs give finite intervals:
$[2,4)$, $[3,5)$, $[6,7)$

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

A persistence diagram D is made of all $(\mathcal{F}(\sigma_+), \mathcal{F}(\sigma_-)) \in \mathbb{R}^2$ where σ_+ (resp. σ_-) is positive (resp. negative), and $\mathcal{F}$ is the filtration function.

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

Computation with matrix reduction



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: $[1,+\infty)$

A persistence diagram D is made of all $(\mathcal{F}(\sigma_+),\mathcal{F}(\sigma_-)) \in \mathbb{R}^2$ where σ_+ (resp. σ_-) is positive (resp. negative), and $\mathcal{F}$ is the filtration function.

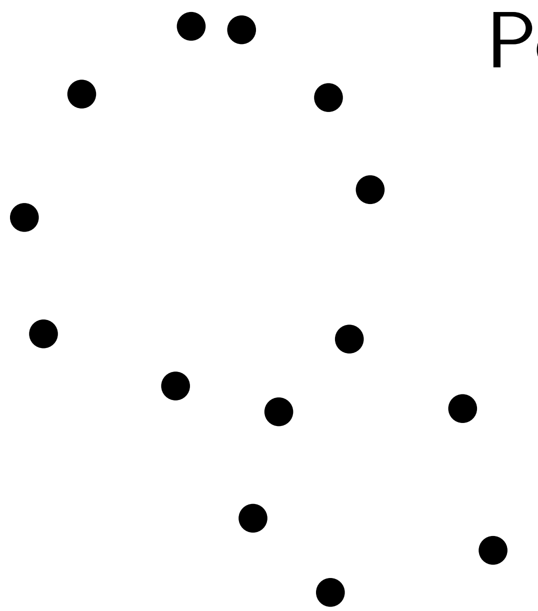
Thus we can define the gradient of a point $p = (\mathcal{F}(\sigma_+),\mathcal{F}(\sigma_-)) \in D$ as

$$\nabla p = [\nabla\mathcal{F}(\sigma_+), \nabla\mathcal{F}(\sigma_-)]$$

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

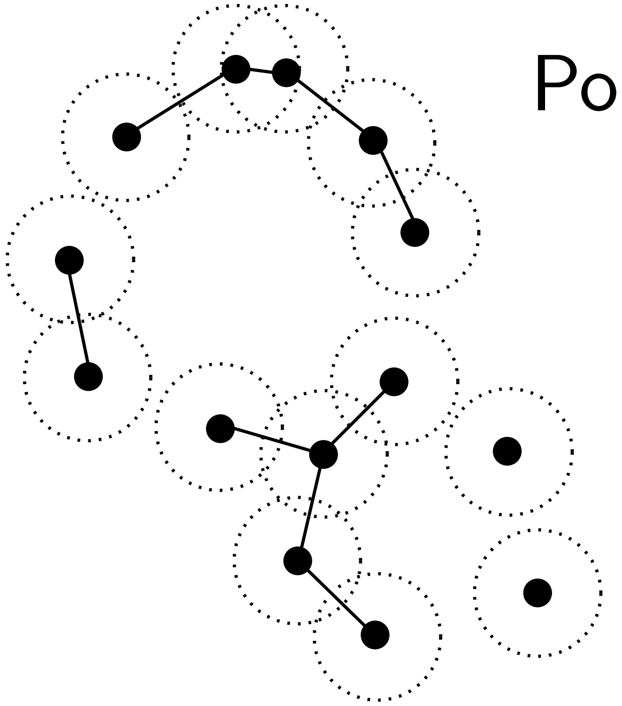
Example: Vietoris-Rips gradient

Example: Vietoris-Rips gradient



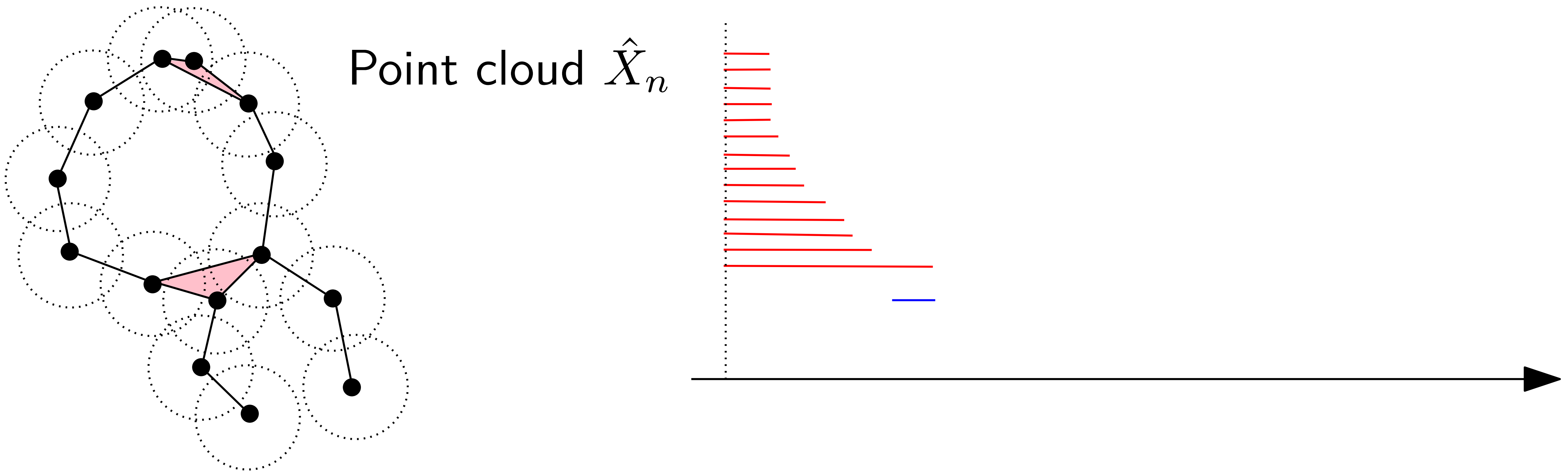
Point cloud $\hat{X}_n$

Example: Vietoris-Rips gradient

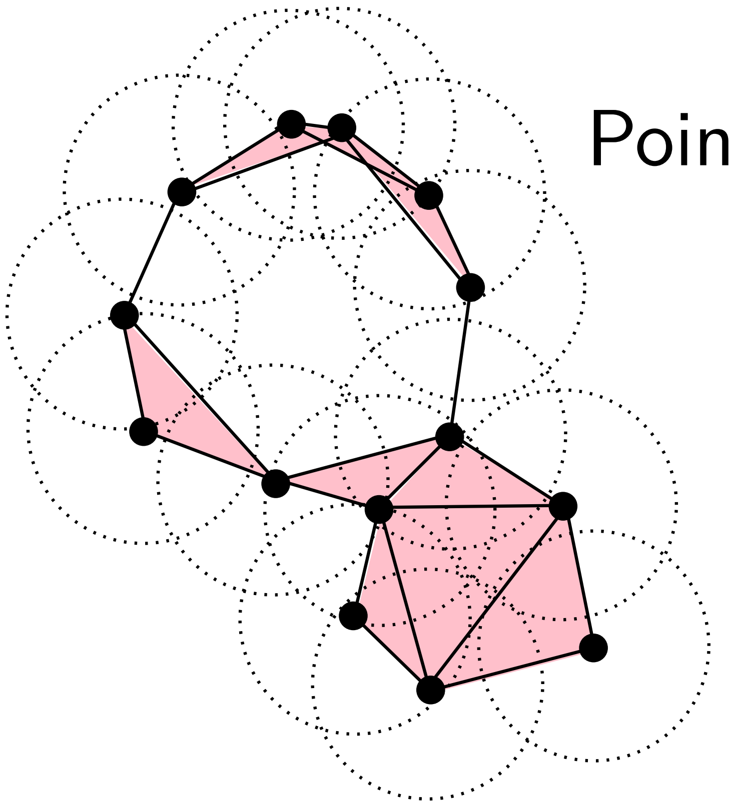


Point cloud $\hat{X}_n$

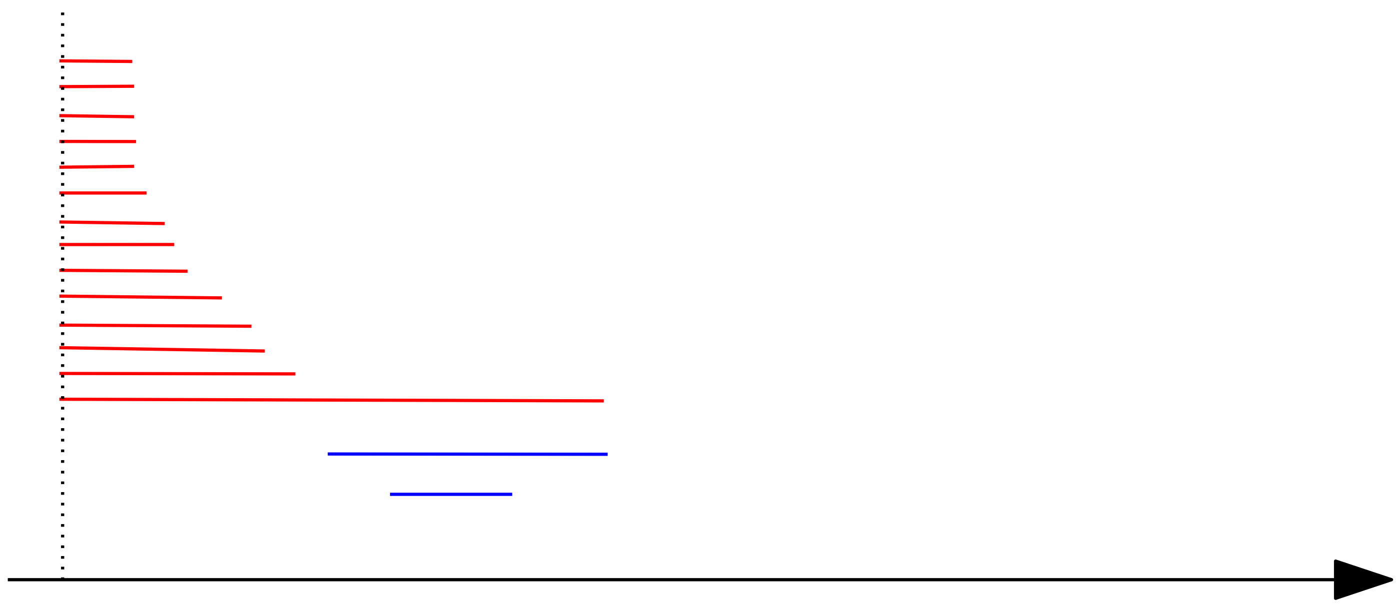
Example: Vietoris-Rips gradient



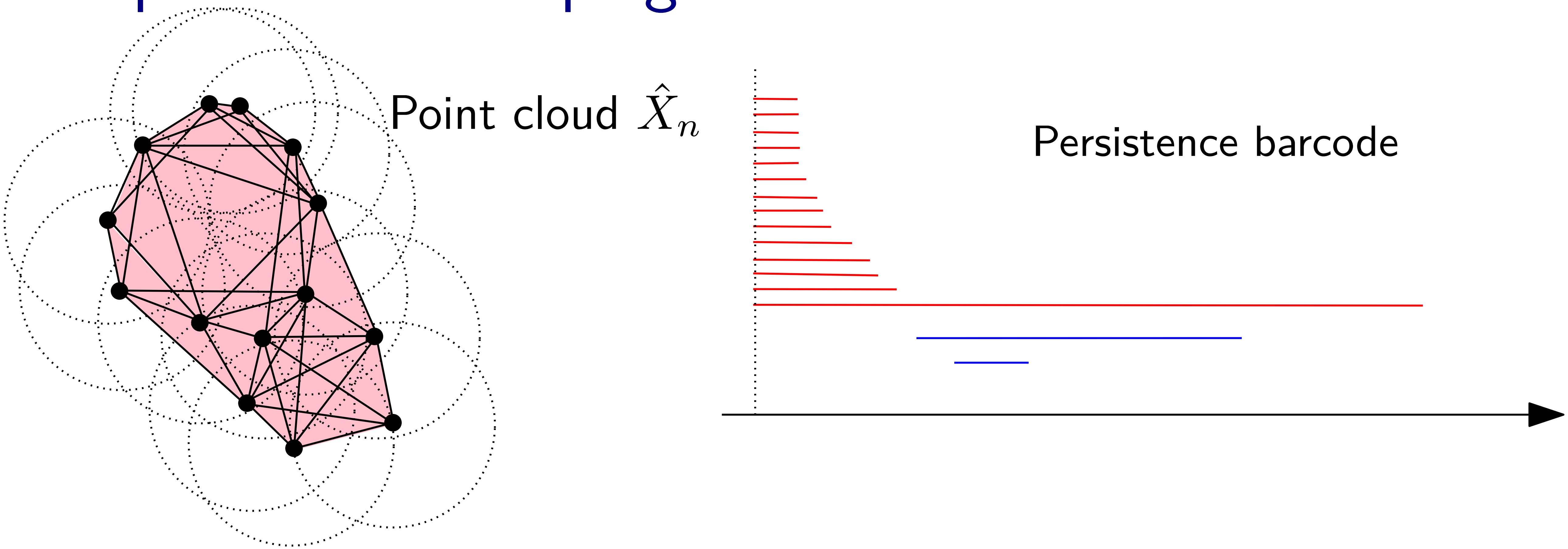
Example: Vietoris-Rips gradient



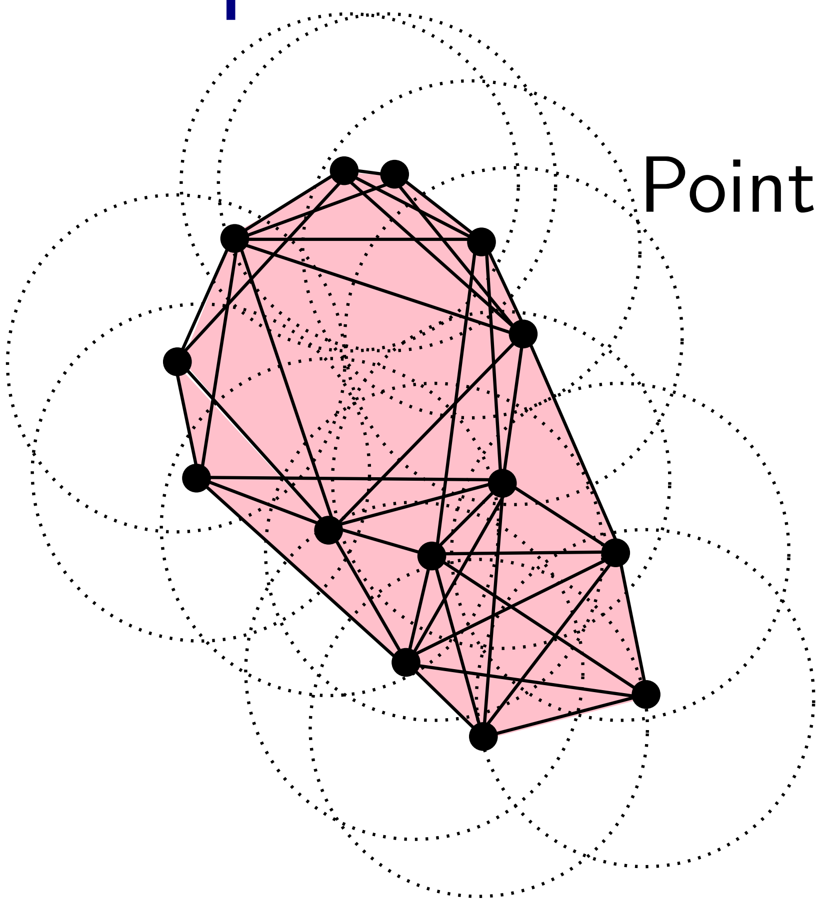
Point cloud $\hat{X}_n$



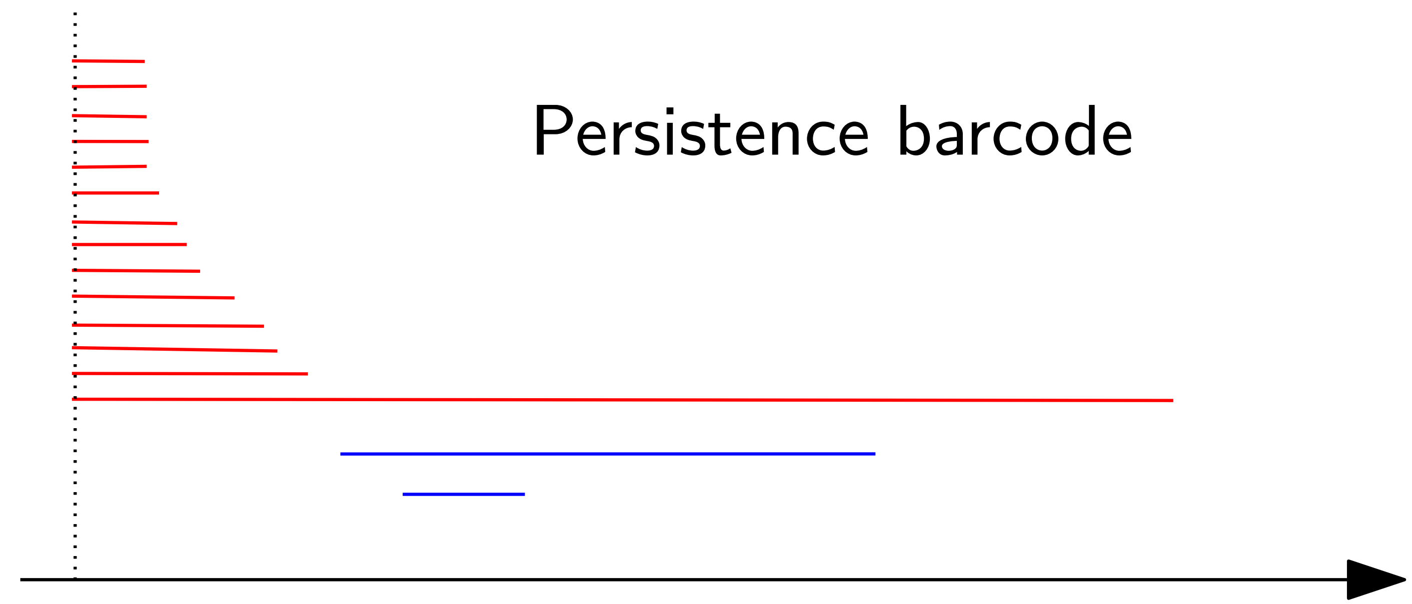
Example: Vietoris-Rips gradient



Example: Vietoris-Rips gradient



Point cloud $\hat{X}_n$

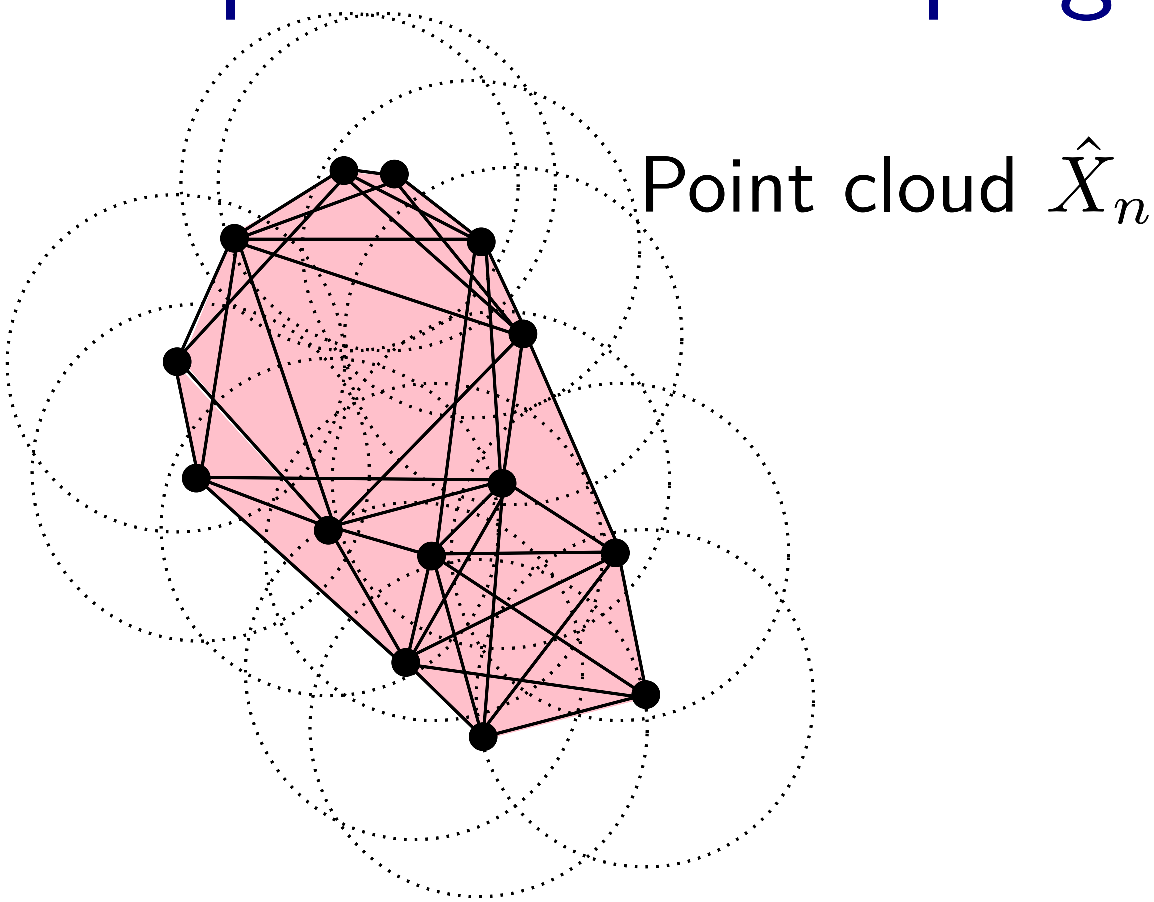


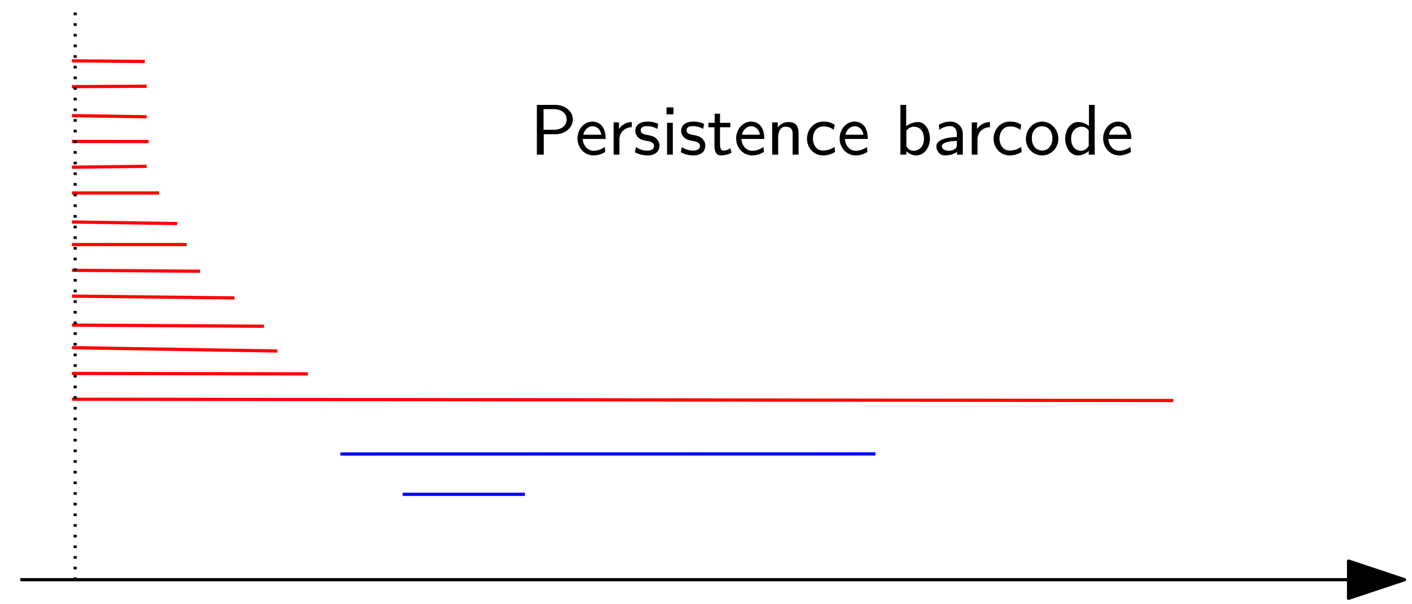
Persistence barcode

Given k-dim. simplex $\sigma = [v_0, \ldots, v_k]$, one has

$$\mathcal{F}(\sigma) = \max_{i,j} \|v_i - v_j\|$$

Example: Vietoris-Rips gradient



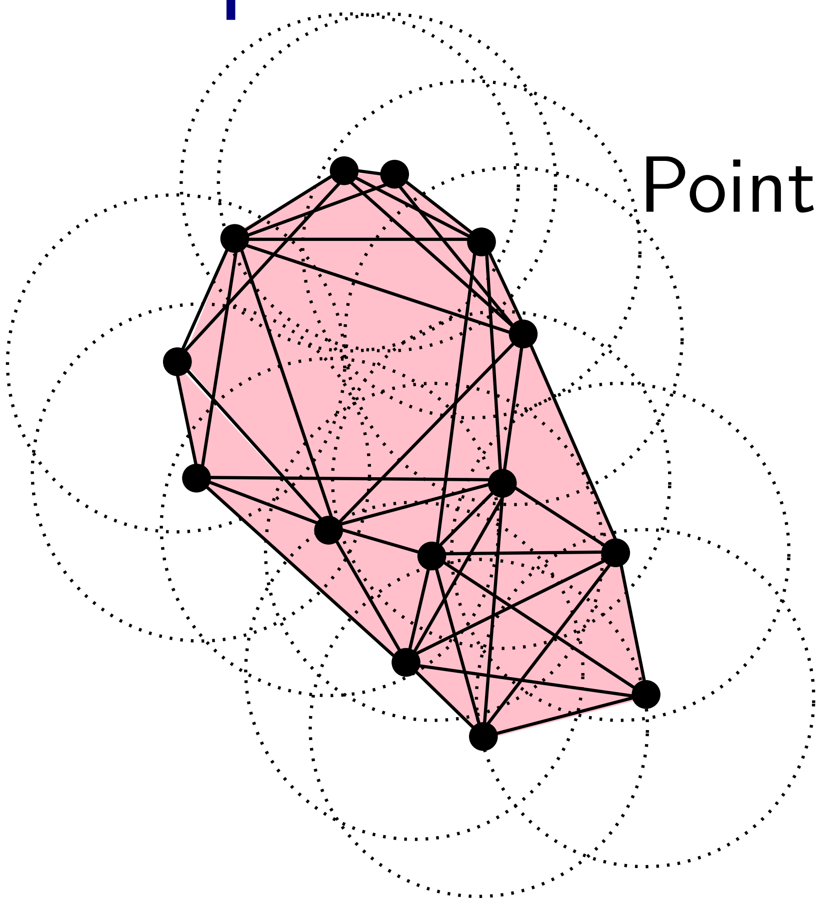


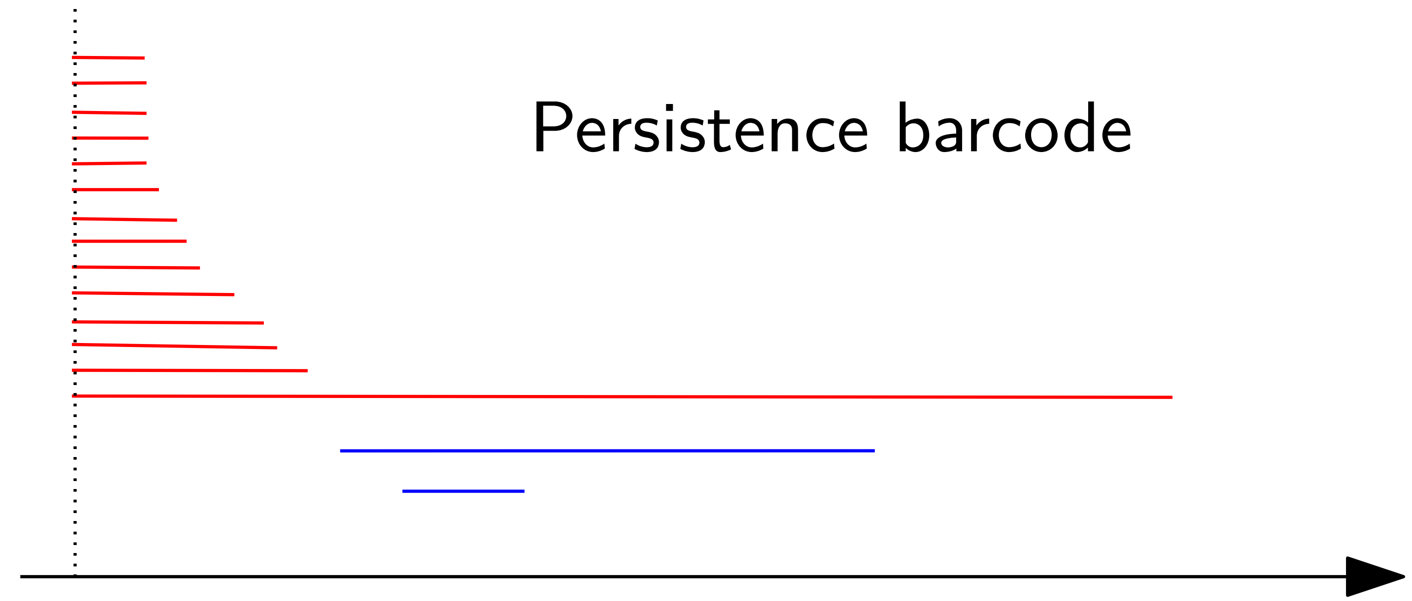
Given k-dim. simplex $\sigma = [v_0, \ldots, v_k]$, one has

$$\mathcal{F}(\sigma) = \max_{i,j} \|v_i - v_j\|$$

Let $p = (\mathcal{F}(\sigma_+), \mathcal{F}(\sigma_-)) \in D_{\mathrm{Rips}}(X)$

Example: Vietoris-Rips gradient





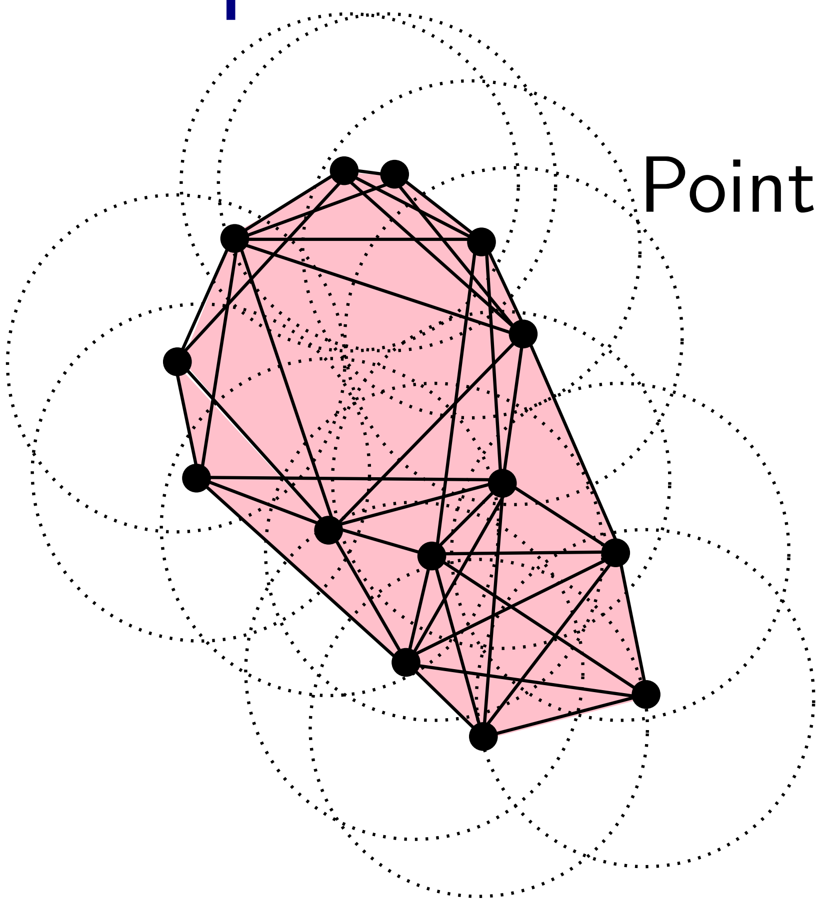
Given k-dim. simplex $\sigma = [v_0, \ldots, v_k]$, one has

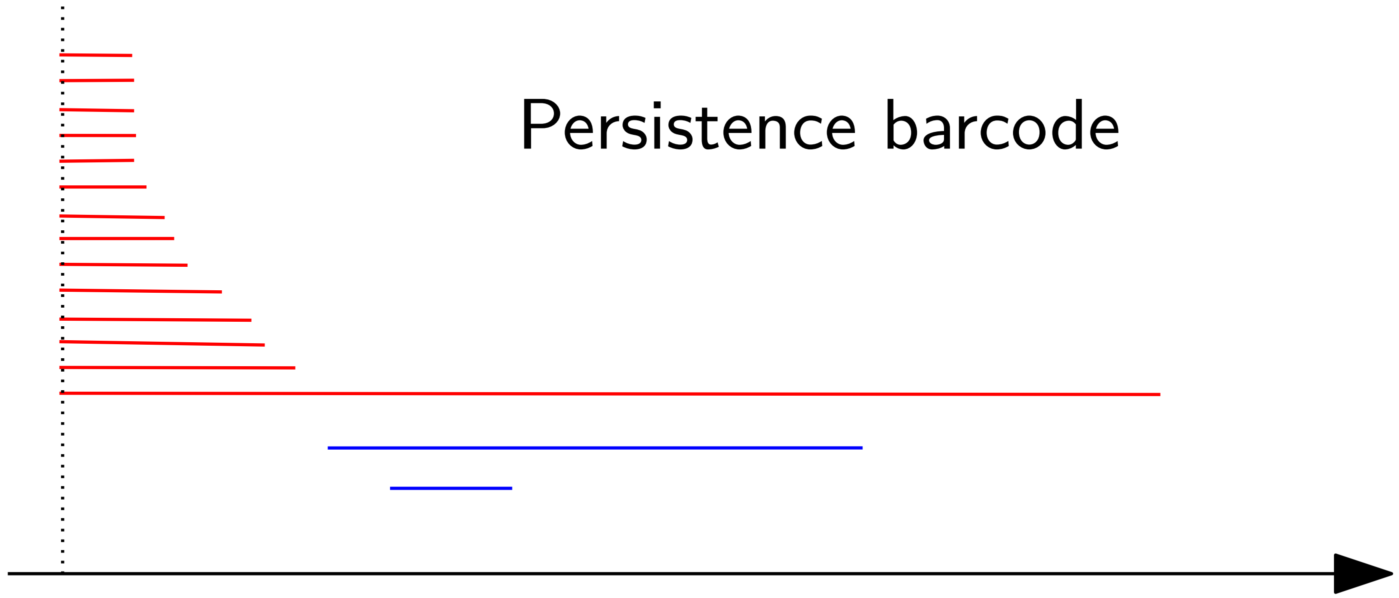
$$\mathcal{F}(\sigma) = \max_{i,j} \|v_i - v_j\|$$

Let $p = (\mathcal{F}(\sigma_+), \mathcal{F}(\sigma_-)) \in D_{\mathrm{Rips}}(X)$

with $\sigma_+ = \{v_0, \ldots, v_k\}$ and $\sigma_- = \{w_0, \ldots, w_{k+1}\}$

Example: Vietoris-Rips gradient



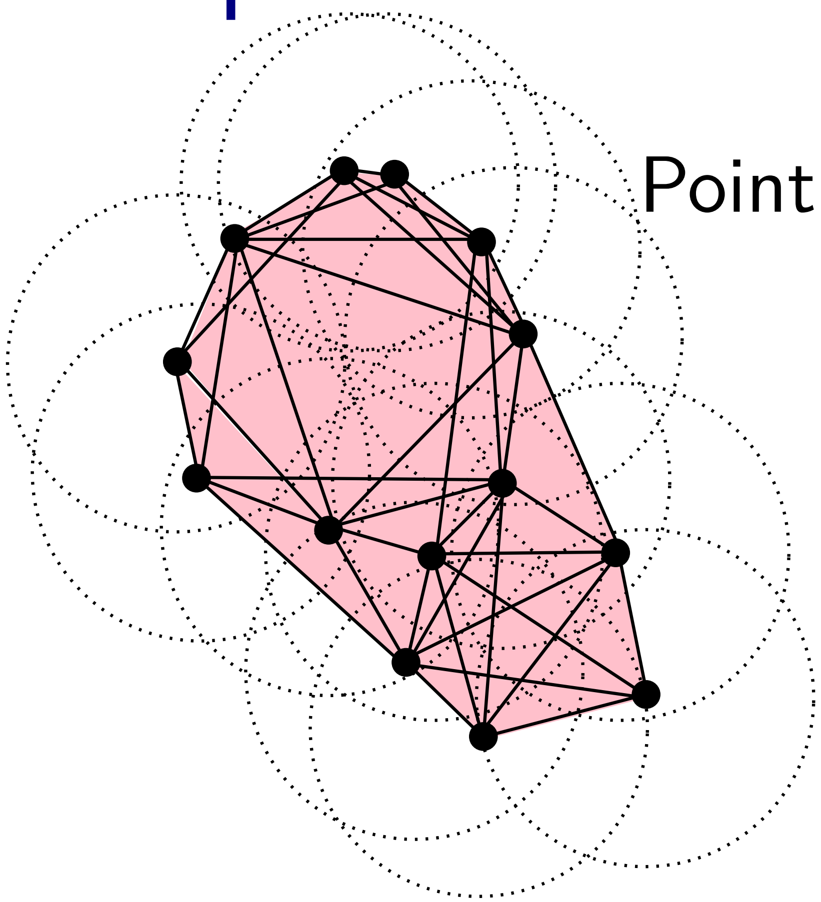


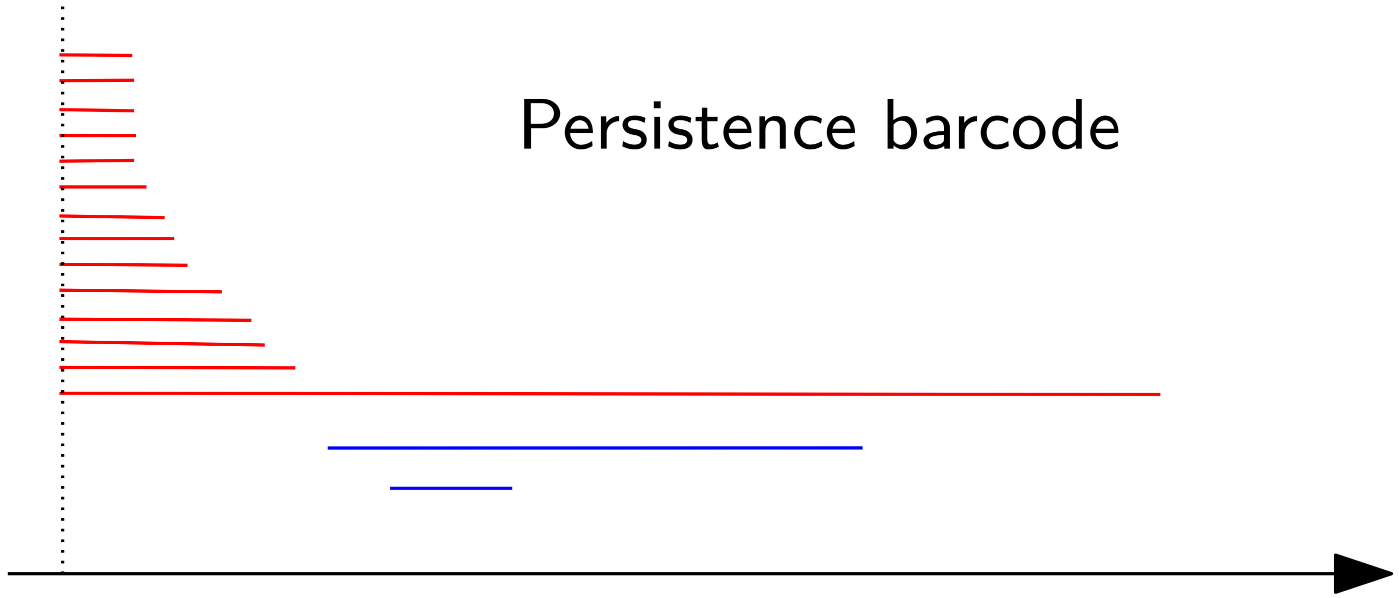
$$\nabla_X p = \left[\frac{\partial}{\partial X} \|v_{i^*} - v_{j^*}\|, \frac{\partial}{\partial X} \|w_{a^*} - w_{b^*}\| \right]$$

Let $p = (\mathcal{F}(\sigma_+), \mathcal{F}(\sigma_-)) \in D_{\mathrm{Rips}}(X)$

with $\sigma_+ = \{v_0, \ldots, v_k\}$ and $\sigma_- = \{w_0, \ldots, w_{k+1}\}$

Example: Vietoris-Rips gradient





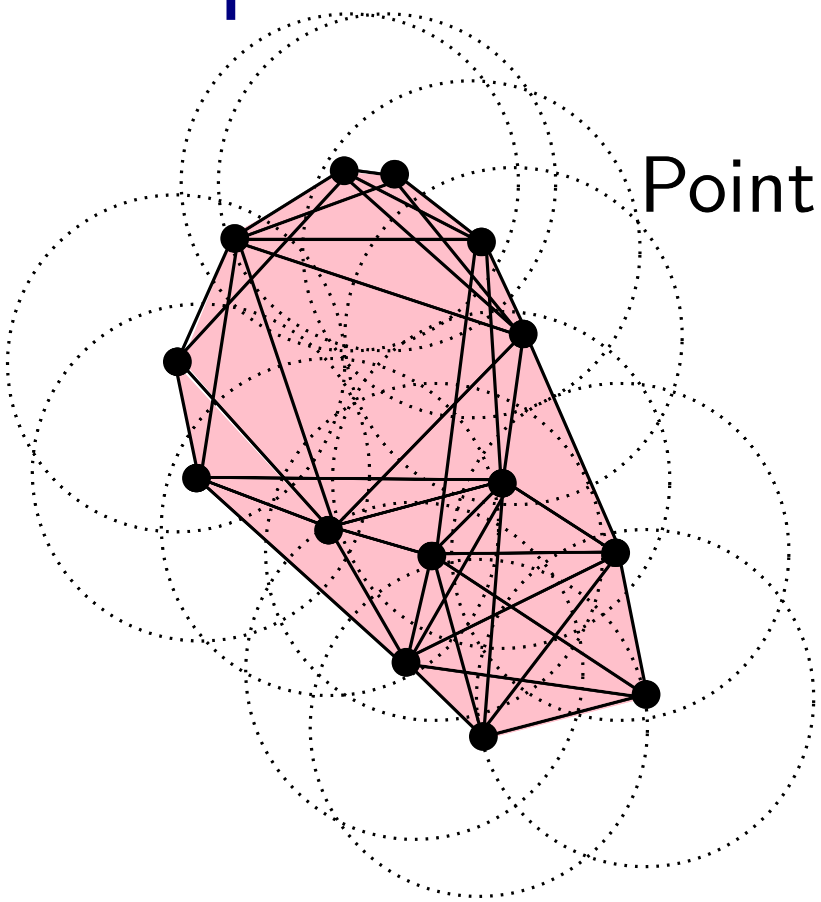
$$\nabla_X p = \left[\frac{\partial}{\partial X}\|v_{i^*} - v_{j^*}\|, \frac{\partial}{\partial X}\|w_{a^*} - w_{b^*}\|\right]$$

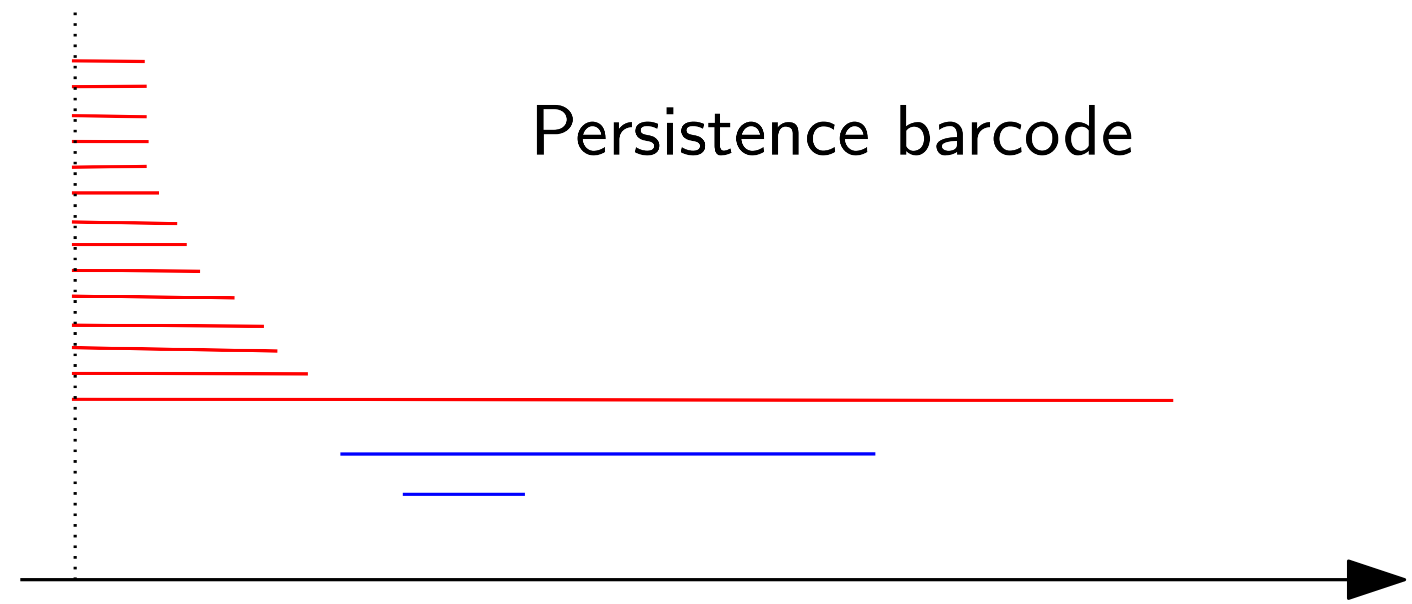
$\frac{\partial}{\partial v_i^{(d)}}\|v_{i^*} - v_{j^*}\| = (-)\frac{1}{\|v_{i^*} - v_{j^*}\|}(v_{i^*}^{(d)} - v_{j^*}^{(d)})$ if $i = i^*$ (j^*) and 0 otherwise

Let $p = (\mathcal{F}(\sigma_+), \mathcal{F}(\sigma_-)) \in D_{\mathrm{Rips}}(X)$

with $\sigma_+ = \{v_0, \ldots, v_k\}$ and $\sigma_- = \{w_0, \ldots, w_{k+1}\}$

Example: Vietoris-Rips gradient



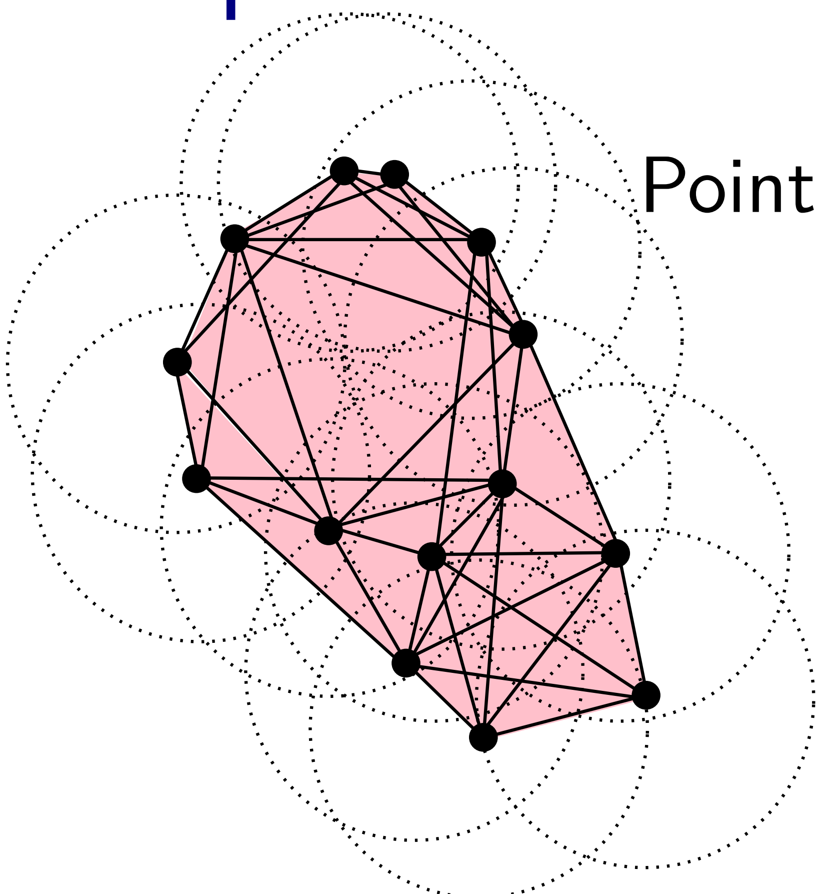


$$\nabla_X p = \left[\frac{\partial}{\partial X}\|v_{i^*} - v_{j^*}\|, \frac{\partial}{\partial X}\|w_{a^*} - w_{b^*}\|\right]$$

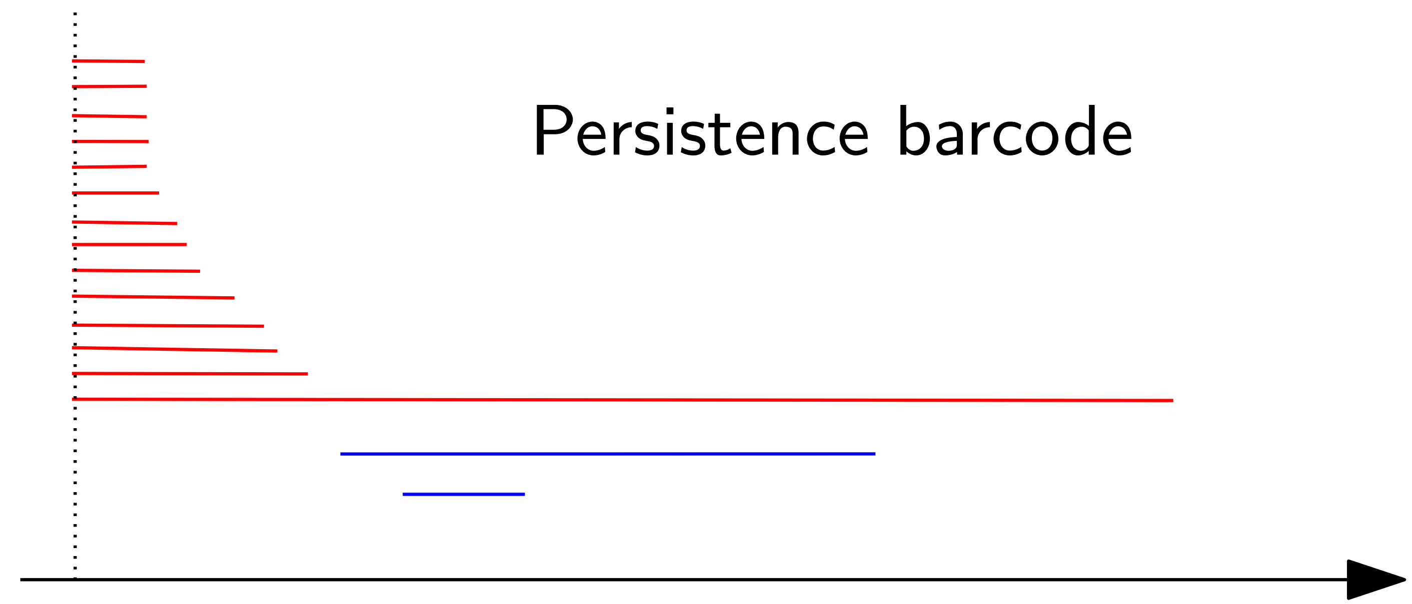
$$\frac{\partial}{\partial v_i^{(d)}}\|v_{i^*} - v_{j^*}\| = (-)\frac{1}{\|v_{i^*} - v_{j^*}\|}(v_{i^*}^{(d)} - v_{j^*}^{(d)}) \text{ if } i = i^* \ (j^*) \text{ and } 0 \text{ otherwise}$$

With this gradient rule, one can do gradient descent with any function of persistence!

Example: Vietoris-Rips gradient

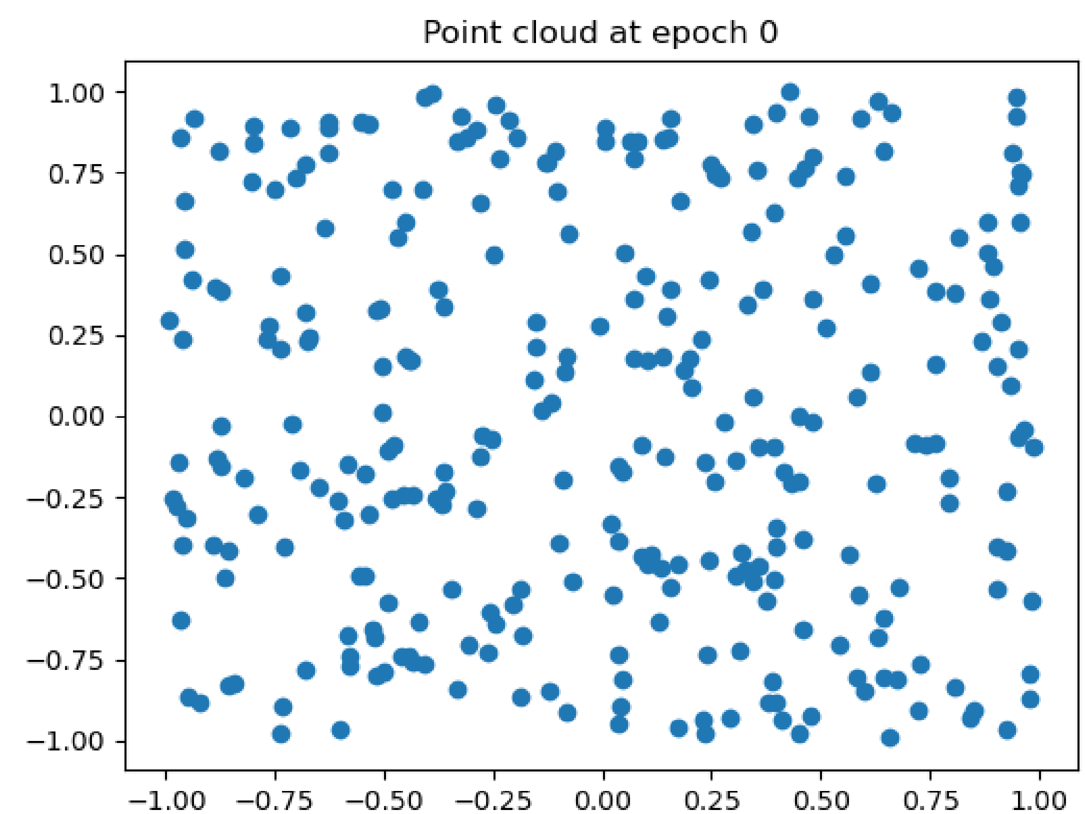


Point cloud $\hat{X}_n$

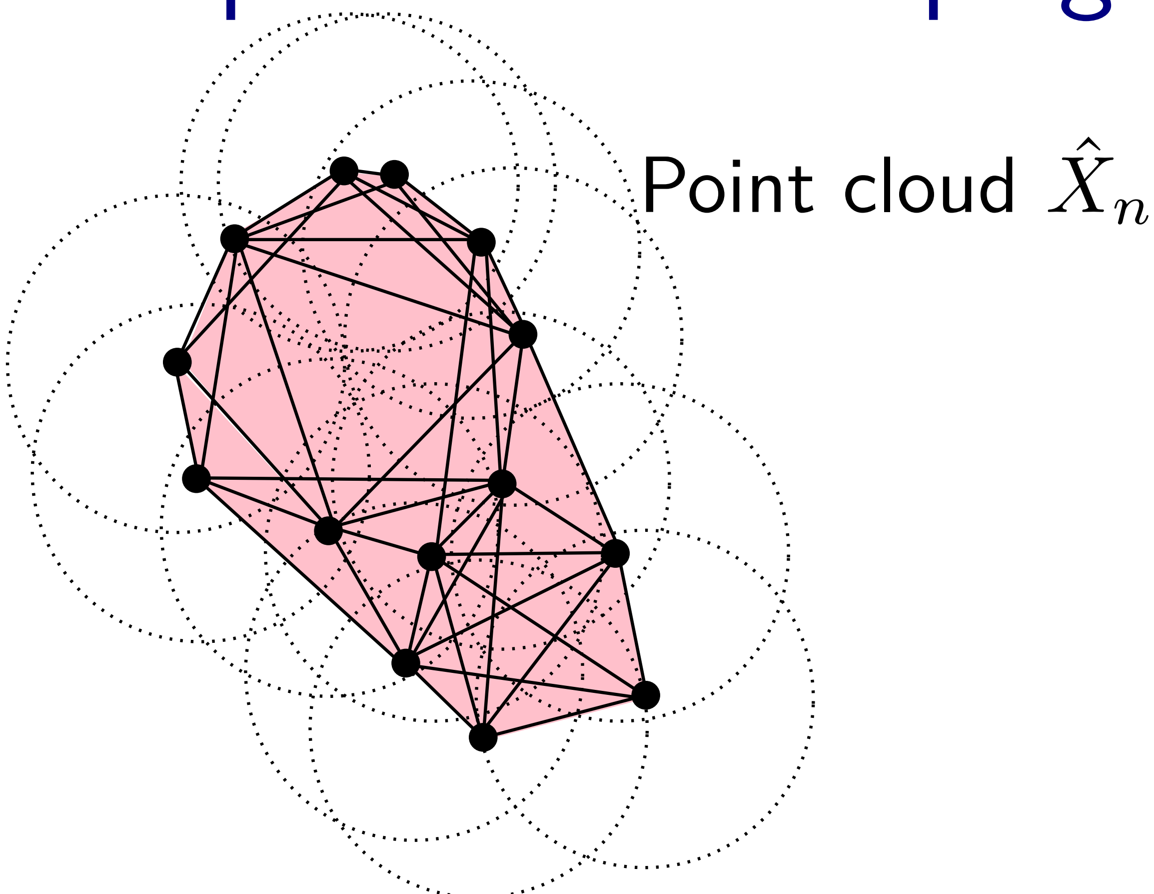


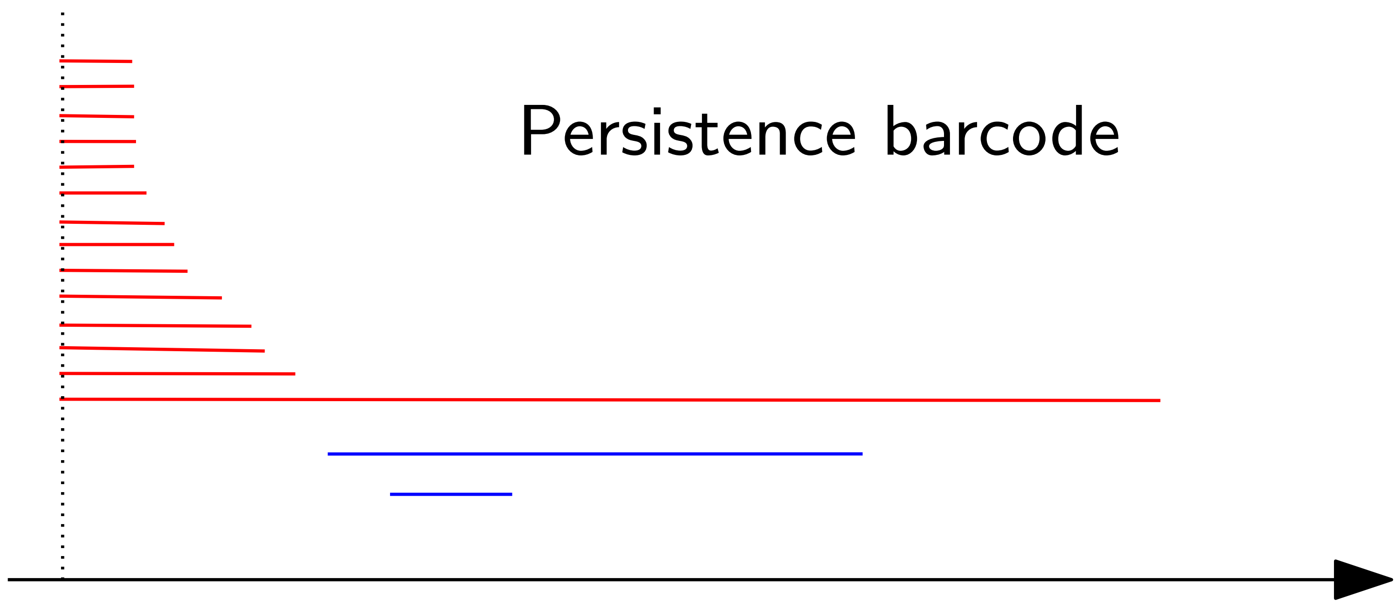
Persistence barcode

Let's say we want to maximize the number of holes in that point cloud.



Example: Vietoris-Rips gradient



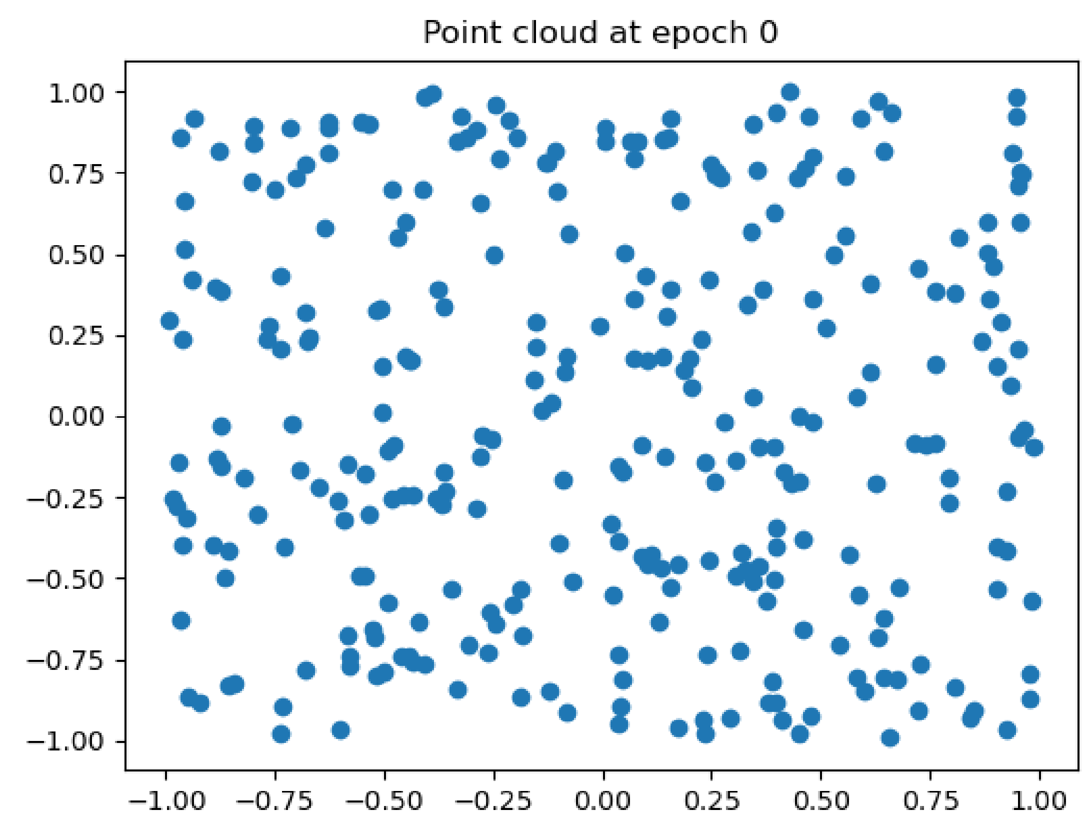


Let's say we want to maximize the number of holes in that point cloud.

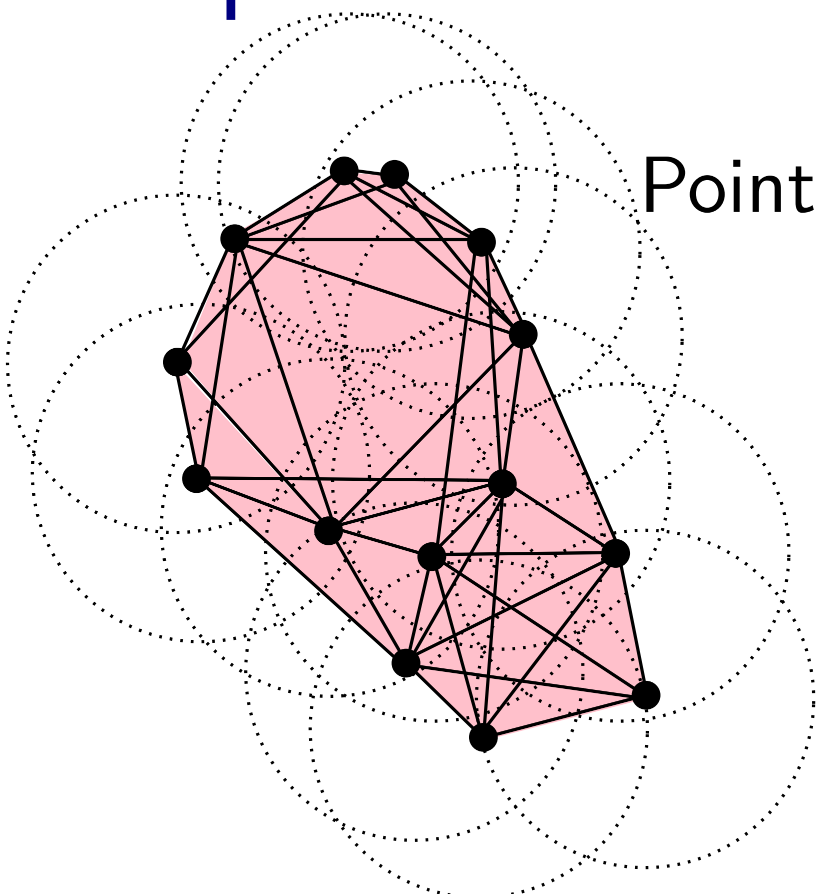
We can use gradient descent to minimize loss

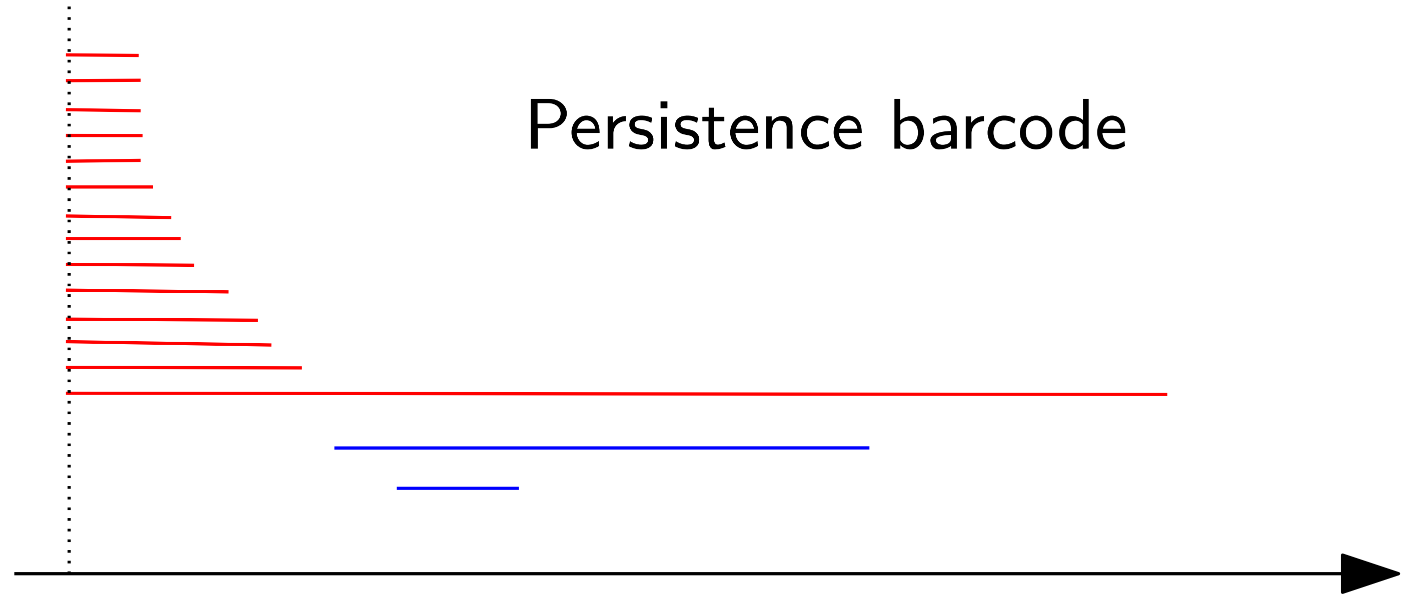
$$\mathcal{L}(X) = -\sum_p \|p\|_2^2,$$

with $p \in D_{\text{Rips}}(X)$ (in dim. 1)

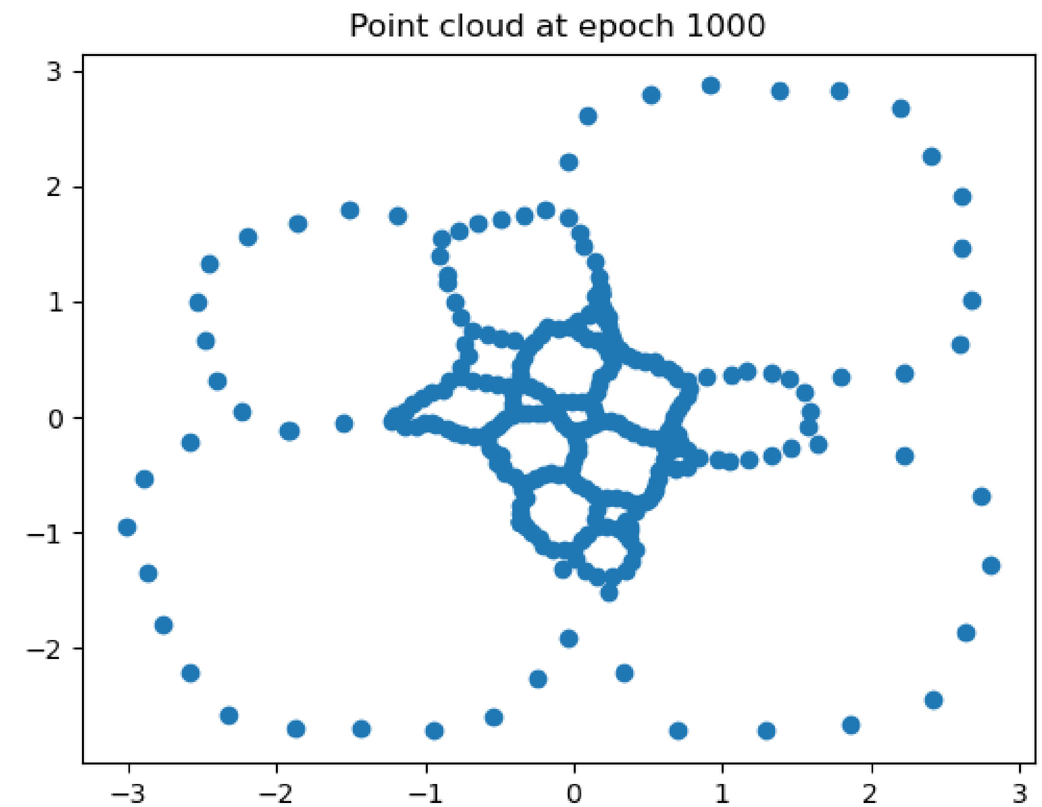


Example: Vietoris-Rips gradient





Let's say we want to maximize the number of holes in that point cloud.

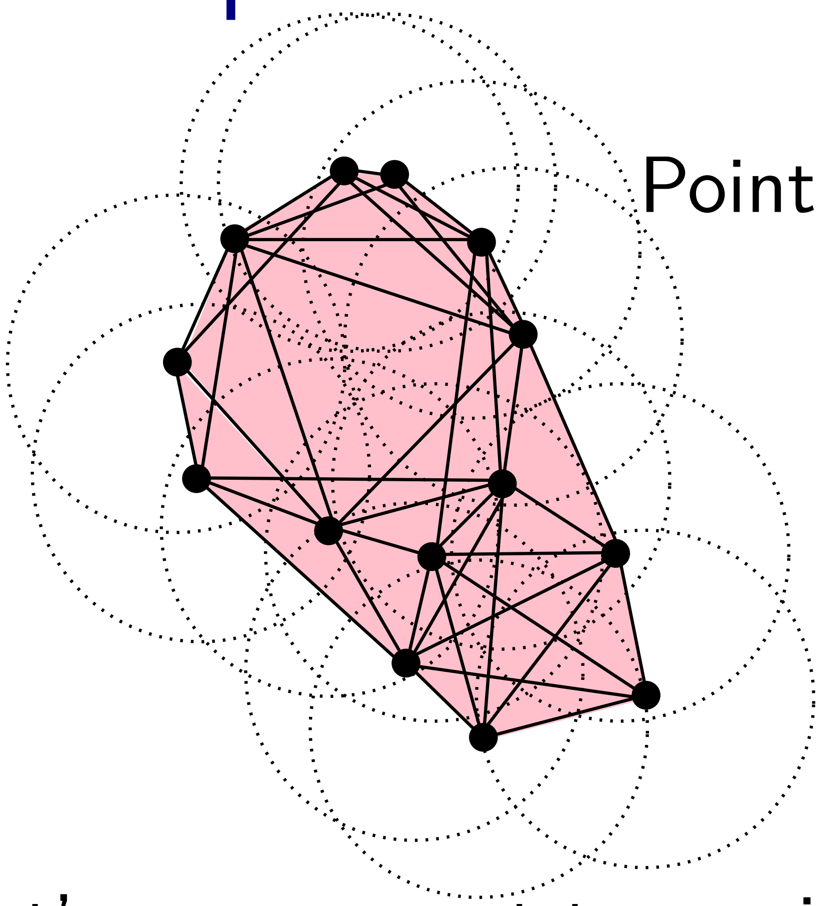


We can use gradient descent to minimize loss

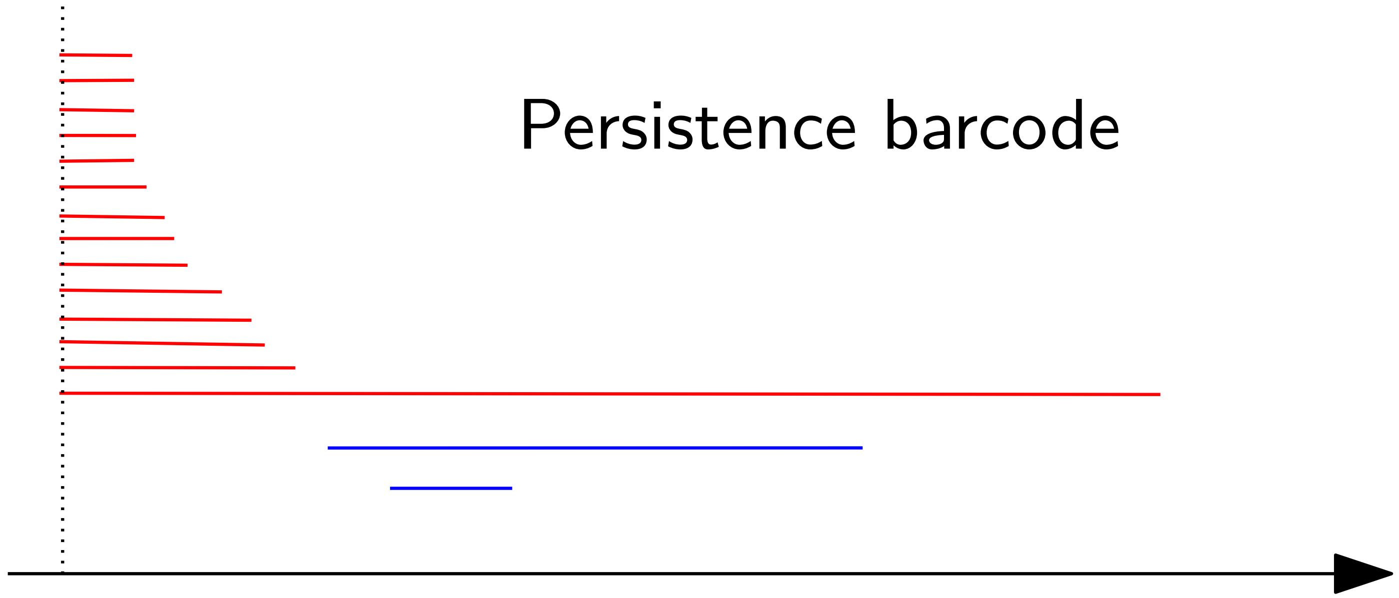
$$\mathcal{L}(X) = -\sum_{p} \|p\|_2^2,$$

with $p \in D_{\mathrm{Rips}}(X)$ (in dim. 1)

Example: Vietoris-Rips gradient

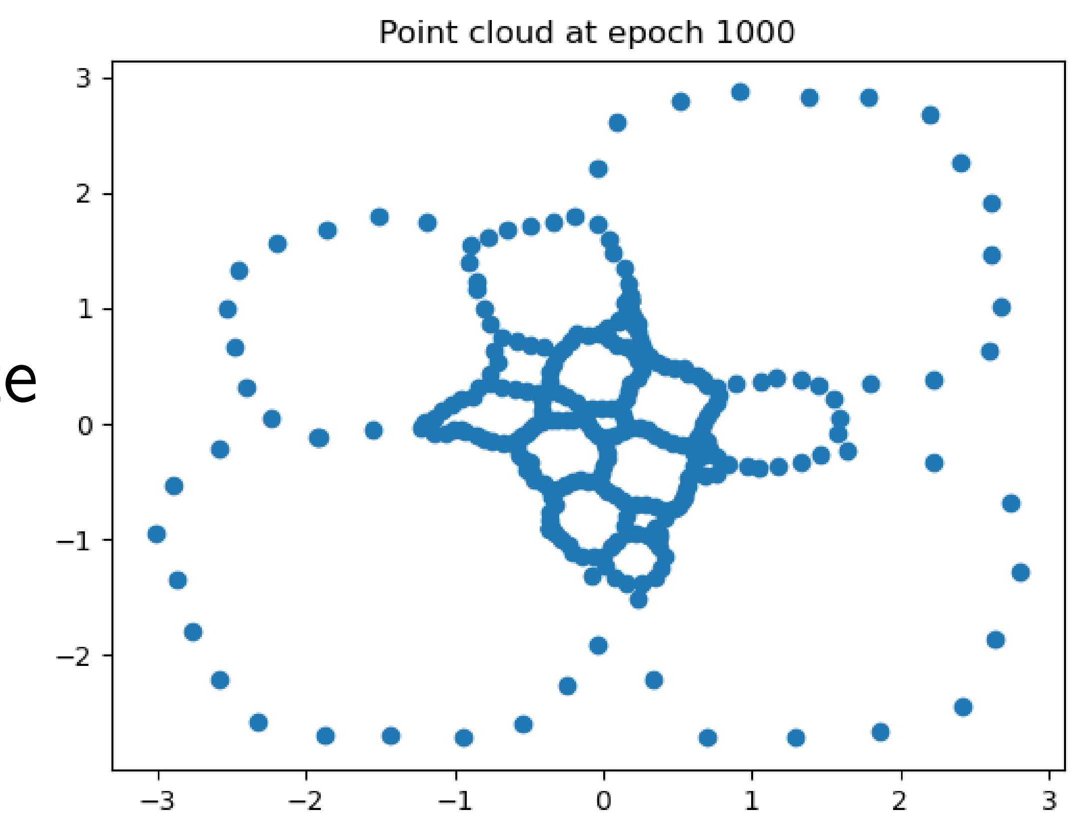


Point cloud $\hat{X}_n$



Persistence barcode

Let's say we want to maximize the number of holes in that point cloud.

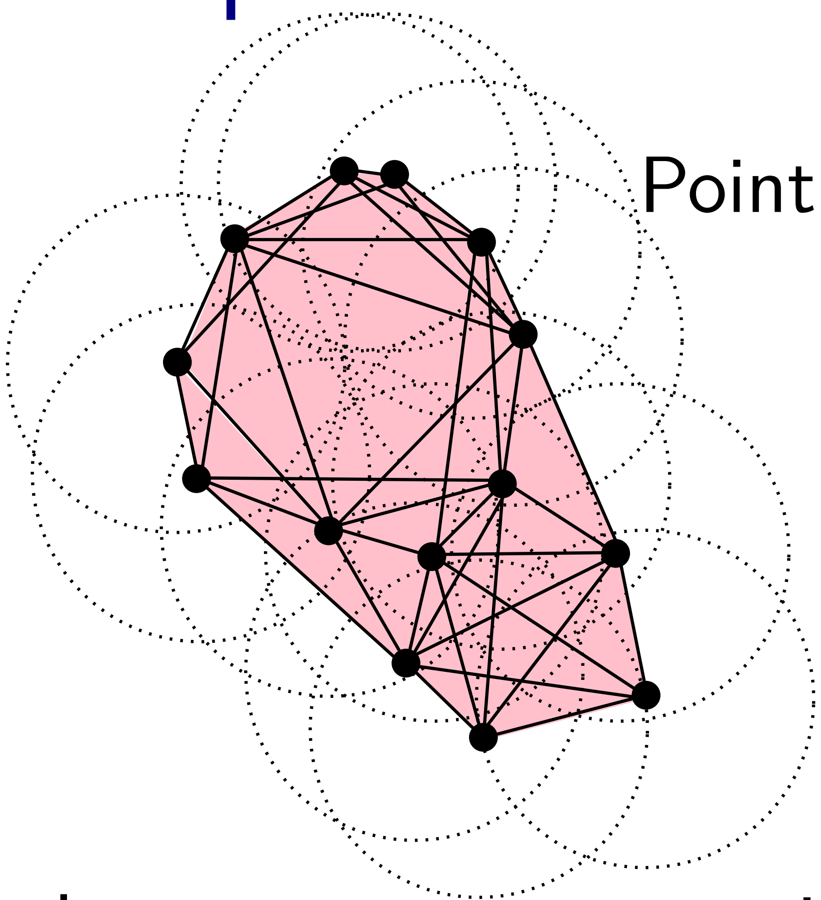


We can use gradient descent to minimize loss

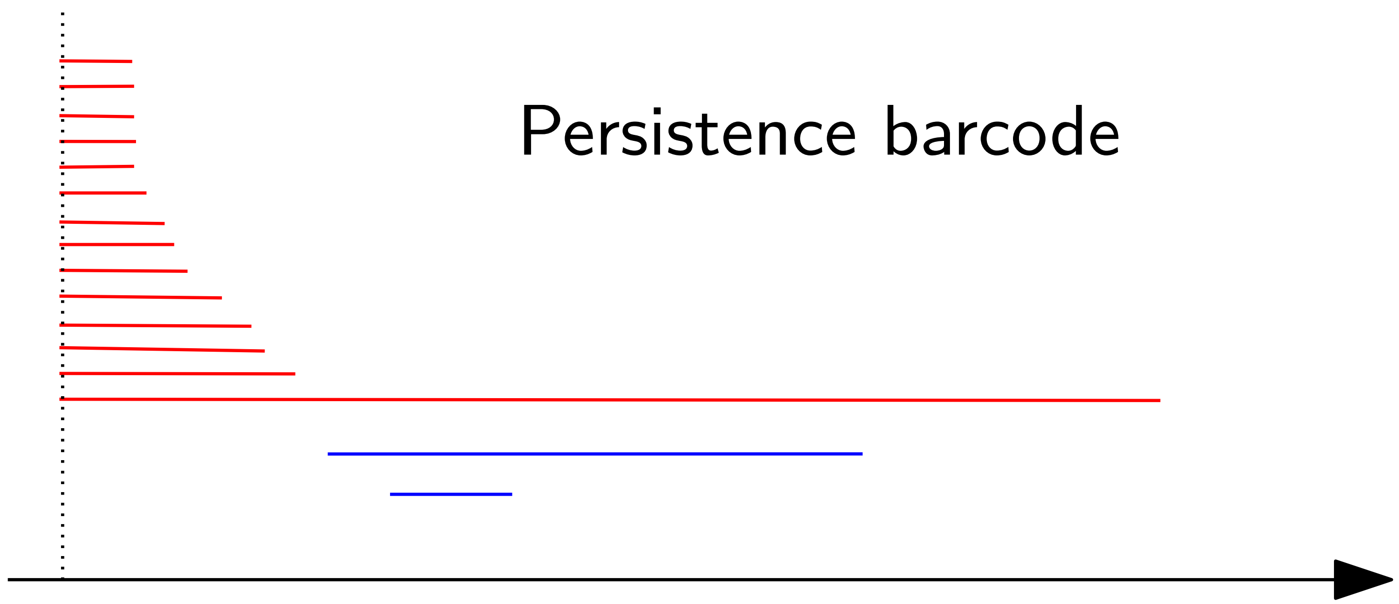
$$\mathcal{L}(X) = -\sum_p \|p\|_2^2 + d(X, C),$$

with $p \in D_{\mathrm{Rips}}(X)$ and C unit square

Example: Vietoris-Rips gradient



Point cloud $\hat{X}_n$

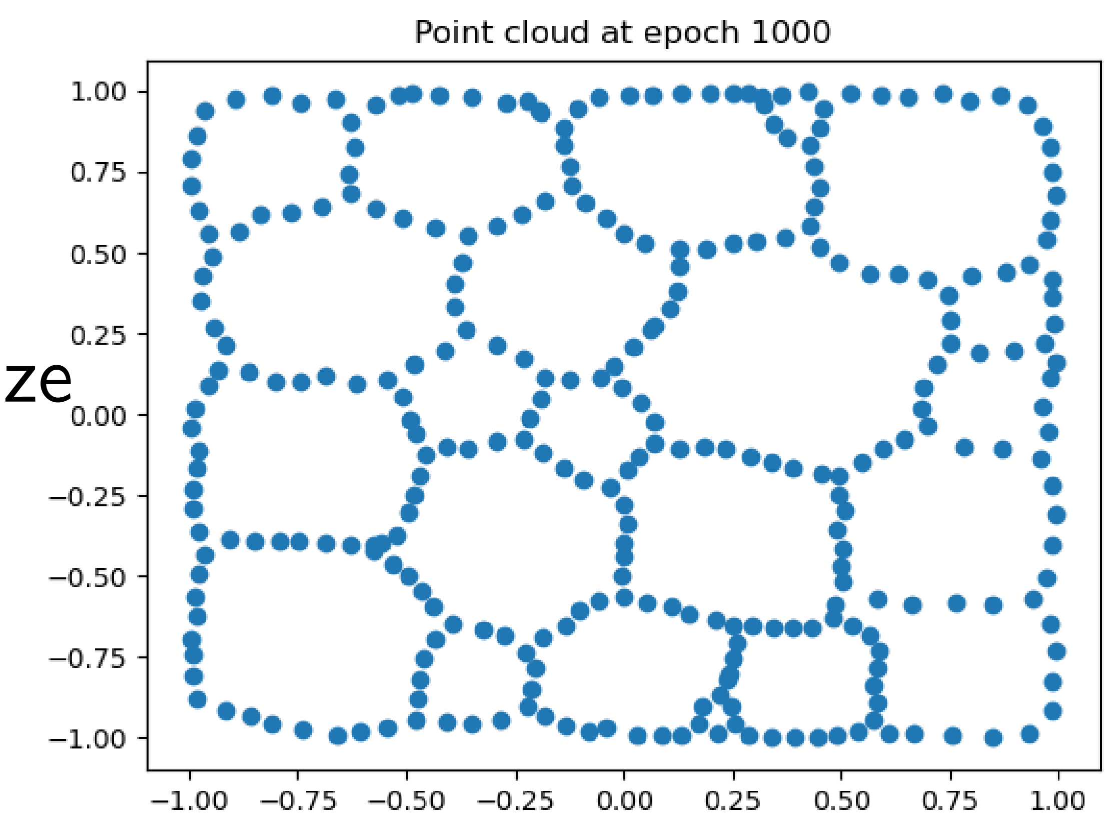


Let's say we want to maximize the number of holes in that point cloud.

We can use gradient descent to minimize loss

$$\mathcal{L}(X) = -\sum_p \|p\|_2^2 + d(X, C),$$

with $p \in D_{\text{Rips}}(X)$ and C unit square



Example: Sublevel sets

Given k-dim. simplex $\sigma = [v_0, \ldots, v_k]$, one has

$$\mathcal{F}(\sigma) = \max_i \; f_\theta(v_i)$$

Example: Sublevel sets

Given k-dim. simplex $\sigma = [v_0, \ldots, v_k]$, one has

$$\mathcal{F}(\sigma) = \max_i \; f_\theta(v_i)$$

$$\nabla_\theta p = \left[\tfrac{\partial}{\partial\theta} f_\theta(v_{i^*}), \tfrac{\partial}{\partial\theta} f_\theta(w_{a^*}) \right]$$

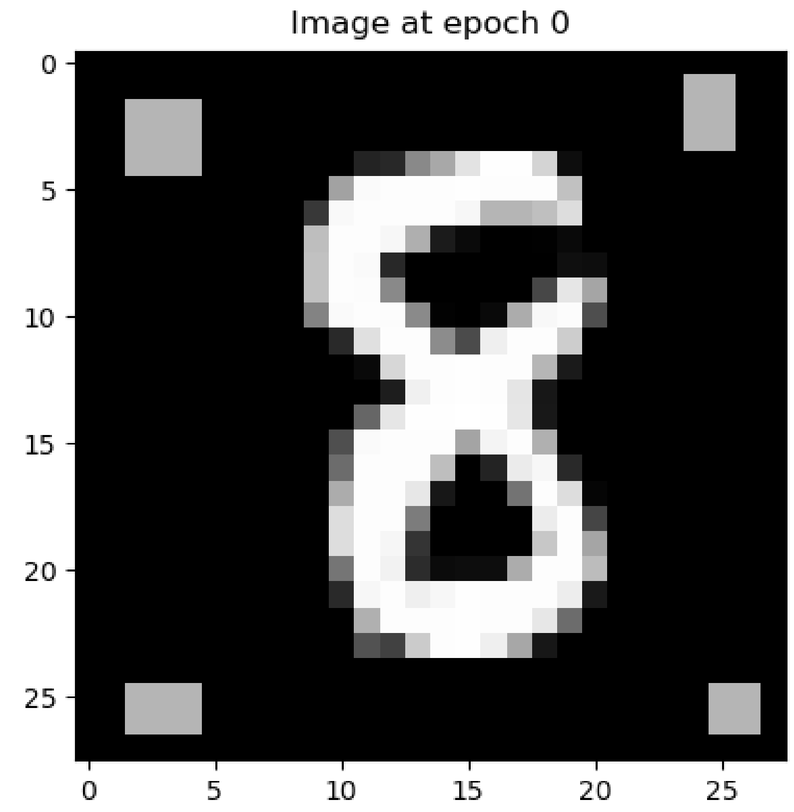
Example: Sublevel sets

Given k-dim. simplex $\sigma = [v_0, \ldots, v_k]$, one has

$$\mathcal{F}(\sigma) = \mathrm{max}_i \; f_\theta(v_i)$$

$$\nabla_\theta p = \left[\frac{\partial}{\partial\theta} f_\theta(v_{i^*}), \frac{\partial}{\partial\theta} f_\theta(w_{a^*})\right]$$

Let's say we want to remove the stains in that image.



Example: Sublevel sets

Given k-dim. simplex $\sigma = [v_0, \ldots, v_k]$, one has

$$\mathcal{F}(\sigma) = \max_i\ f_\theta(v_i)$$

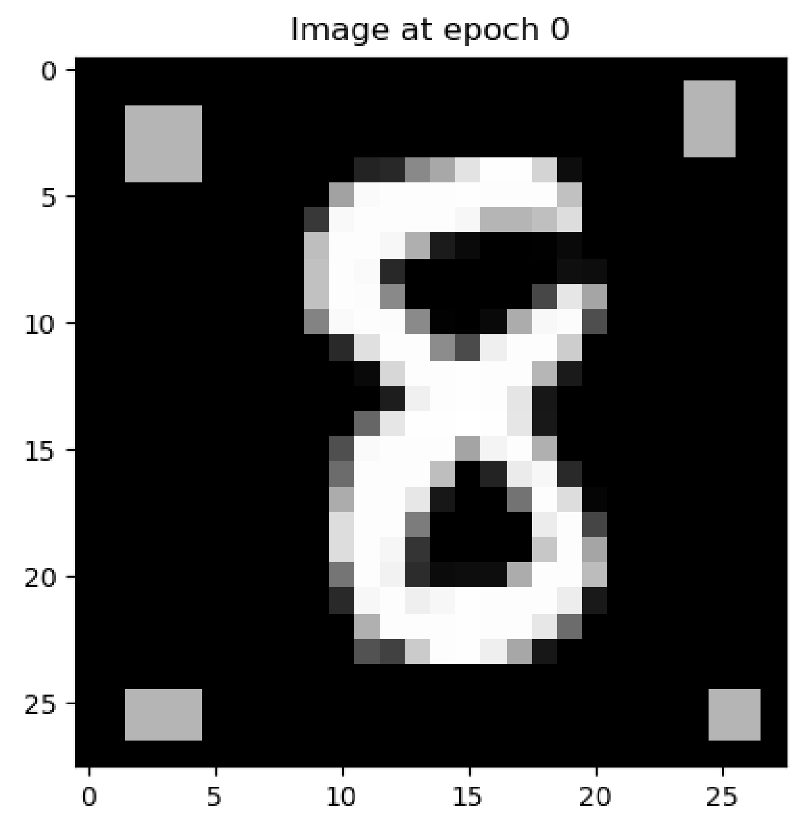
$$\nabla_\theta p = \left[\tfrac{\partial}{\partial\theta} f_\theta(v_{i^*}), \tfrac{\partial}{\partial\theta} f_\theta(w_{a^*})\right]$$

Let's say we want to remove the stains in that image.

We can use gradient descent to minimize loss

$$\mathcal{L}(X) = \sum_p \|p\|_2^2,$$

with $p \in D_{\mathrm{grey}}$ (in dim. 0)



Example: Sublevel sets

Given k-dim. simplex $\sigma = [v_0, \ldots, v_k]$, one has

$$\mathcal{F}(\sigma) = \mathrm{max}_i \ f_\theta(v_i)$$

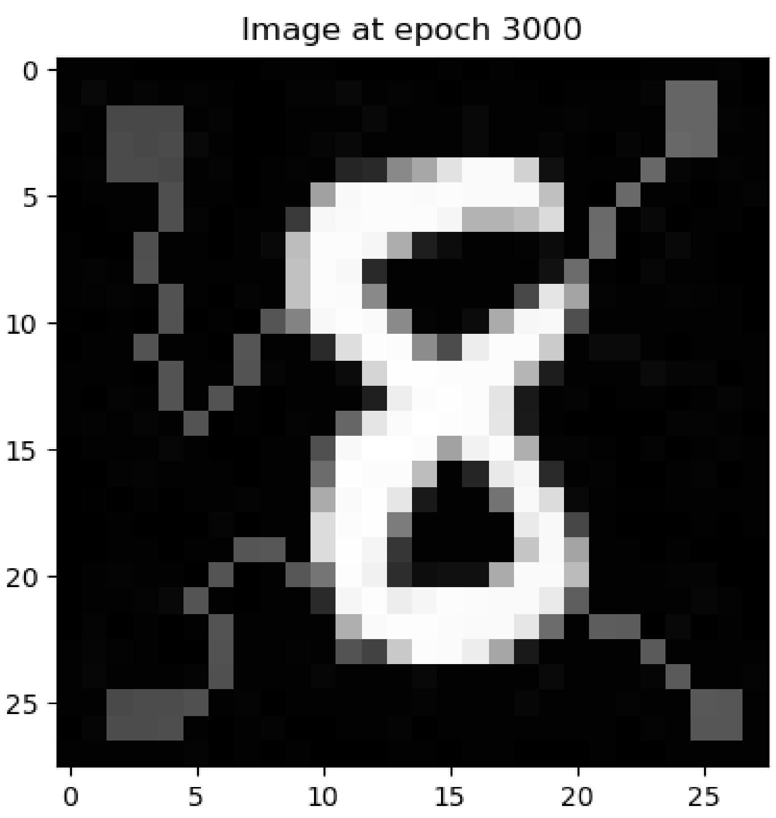
$$\nabla_\theta p = \left[\tfrac{\partial}{\partial \theta} f_\theta(v_{i^*}), \tfrac{\partial}{\partial \theta} f_\theta(w_{a^*})\right]$$

Let's say we want to remove the stains in that image.

We can use gradient descent to minimize loss

$$\mathcal{L}(X) = \sum_p \|p\|_2^2,$$

with $p \in D_{\mathrm{grey}}$ (in dim. 0)



Example: Sublevel sets

Given k-dim. simplex $\sigma = [v_0, \ldots, v_k]$, one has

$$\mathcal{F}(\sigma) = \mathrm{max}_i \ f_\theta(v_i)$$

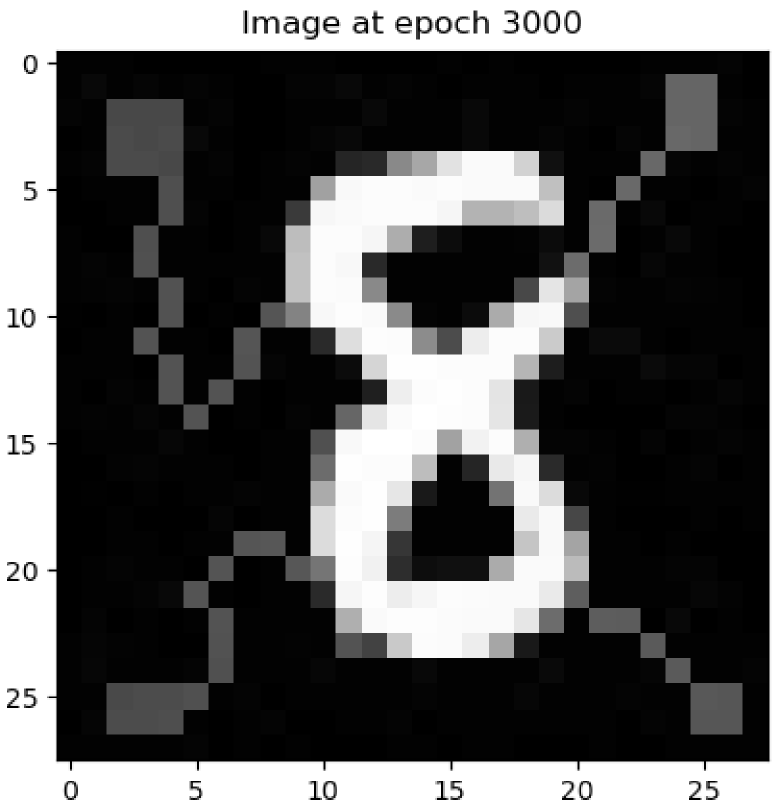
$$\nabla_\theta p = \left[\tfrac{\partial}{\partial\theta} f_\theta(v_{i^*}), \tfrac{\partial}{\partial\theta} f_\theta(w_{a^*})\right]$$

Let's say we want to remove the stains in that image.

We can use gradient descent to minimize loss

$$\mathcal{L}(X) = \sum_p \|p\|_2^2 + \sum_{P \in I} \max\{|P|, |1 - P|\},$$

with $p \in D_{\mathrm{grey}}$ (in dim. 0)



Example: Sublevel sets

Given k-dim. simplex $\sigma = [v_0, \ldots, v_k]$, one has

$$\mathcal{F}(\sigma) = \max_i \ f_\theta(v_i)$$

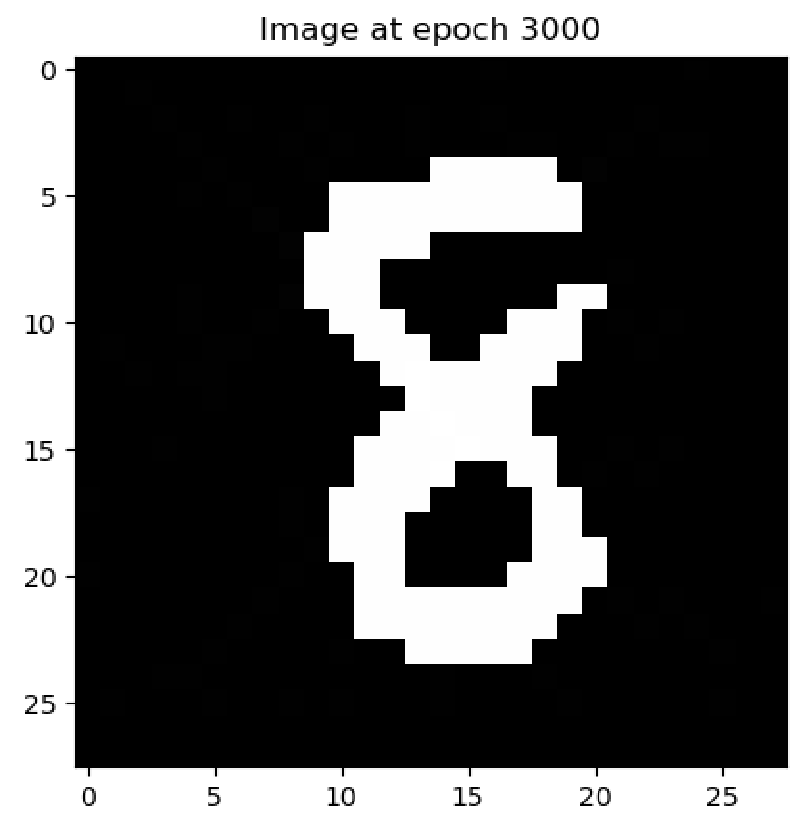
$$\nabla_\theta p = \left[\tfrac{\partial}{\partial\theta} f_\theta(v_{i^*}), \tfrac{\partial}{\partial\theta} f_\theta(w_{a^*})\right]$$

Let's say we want to remove the stains in that image.

We can use gradient descent to minimize loss

$$\mathcal{L}(X) = \sum_p \|p\|_2^2 + \sum_{P \in I} \max\{|P|, |1-P|\},$$

with $p \in D_{\text{grey}}$ (in dim. 0)



Topological gradient descent

[*Optimizing persistent homology based functions*, C., Chazal, Glisse, Ike, Kanna, Umeda, ICML, 2021]

Topological gradient descent

[*Optimizing persistent homology based functions*, C., Chazal, Glisse, Ike, Kanna, Umeda, ICML, 2021]

For a fixed ordering of the simplices in a simplicial complex K, the corresponding persistence diagram always has the same number of points: its gradient is well-defined!

Topological gradient descent

[*Optimizing persistent homology based functions*, C., Chazal, Glisse, Ike, Kanna, Umeda, ICML, 2021]

For a fixed ordering of the simplices in a simplicial complex K, the corresponding persistence diagram always has the same number of points: its gradient is well-defined!

If the ordering changes, the boundary matrix can have a new reduced form and the persistence diagram can have a new, different number of points.

Topological gradient descent

[*Optimizing persistent homology based functions*, C., Chazal, Glisse, Ike, Kanna, Umeda, ICML, 2021]

For a fixed ordering of the simplices in a simplicial complex K, the corresponding persistence diagram always has the same number of points: its gradient is well-defined!

If the ordering changes, the boundary matrix can have a new reduced form and the persistence diagram can have a new, different number of points.

Prop: Let K be a simplicial complex and let $\Phi : A \to \mathbb{R}^{|K|}$ a (parameterized) filtration of K. There exists a partition $A = S \sqcup O_1 \sqcup \cdots \sqcup O_k$ s.t. all the restrictions $\Phi : O_i \to \mathbb{R}^{|K|}$ are differentiable.

The O_i's are the parts of A where the ordering of the simplices of K is preserved, and S is the boundaries of all O_i's.

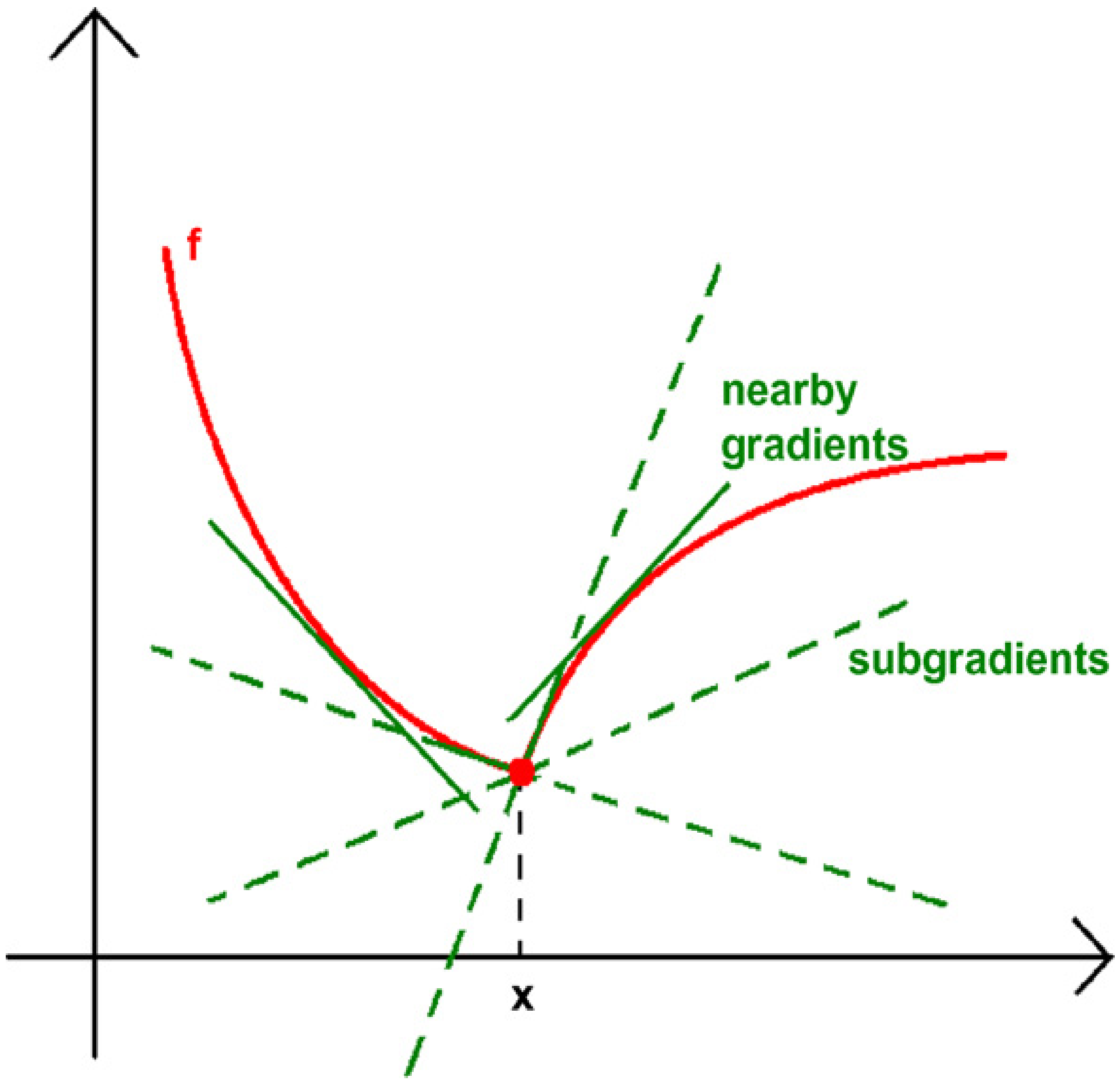
Topological gradient descent

[*Optimizing persistent homology based functions*, C., Chazal, Glisse, Ike, Kanna, Umeda, ICML, 2021]

Def: The *Clarke subdifferential* $\partial\mathcal{L}$ of $\mathcal{L}$ is the set:

$$\partial_x\mathcal{L} = \mathrm{conv}\{\lim_{x_i\to x}\nabla\mathcal{L}(x_i) : \mathcal{L} \text{ is diff. at } x_i\},$$

where conv denotes the convex hull.



Topological gradient descent

[*Optimizing persistent homology based functions*, C., Chazal, Glisse, Ike, Kanna, Umeda, ICML, 2021]

Let $\{\alpha_k\}_k$, $\{\zeta_k\}_k$ s.t.

$\alpha_k \geq 0$, $\sum_k \alpha_k = +\infty$ and $\sum_k \alpha_k^2 < +\infty$

ζ_k random variables s.t. $E[\zeta_k] = 0$ and $E[\|\zeta_k\|^2] < C$ for some $C > 0$

Thm: As long as $\mathcal{L} \circ \mathrm{Pers} \circ \Phi$ is locally Lipschitz, the sequence

$$x_{k+1} = x_k - \alpha_k(g_k + \zeta_k),$$

where $g_k \in \partial_{x_k}(\mathcal{L} \circ \mathrm{Pers} \circ \Phi)$, converges to a critical point of $\mathcal{L} \circ \mathrm{Pers} \circ \Phi$.

Example: filter selection

[*Optimizing persistent homology based functions*, C., Chazal, Glisse, Ike, Kanna, Umeda, ICML, 2021]

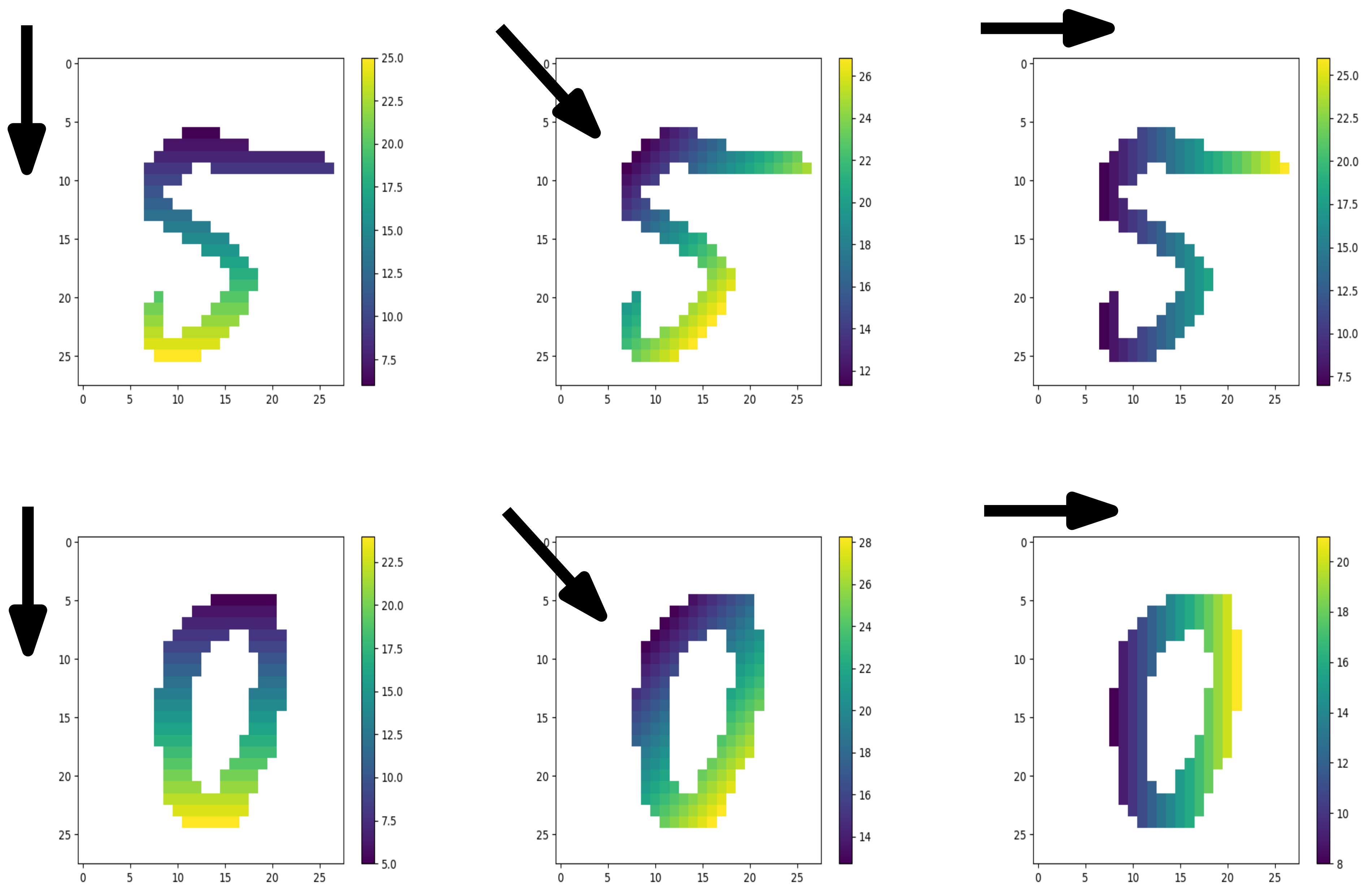
Assume we have a supervised classification task. The goal is to find a filtration from a family $\mathcal{F}$ such that the corresponding persistence diagrams give the best classification score.

Example: filter selection

[*Optimizing persistent homology based functions*, C., Chazal, Glisse, Ike, Kanna, Umeda, ICML, 2021]

Assume we have a supervised classification task. The goal is to find a filtration from a family $\mathcal{F}$ such that the corresponding persistence diagrams give the best classification score.

Ex: images filtered by a direction parameterized by angle.



Example: filter selection

[*Optimizing persistent homology based functions*, C., Chazal, Glisse, Ike, Kanna, Umeda, ICML, 2021]

Assume we have a supervised classification task. The goal is to find a filtration from a family $\mathcal{F}$ such that the corresponding persistence diagrams give the best classification score.

Idea: minimize:

$$\mathcal{L}(f) = \sum_{l} \frac{\sum_{y_i = y_j = l} d_q(D_f(x_i), D_f(x_j))}{\sum_{y_i = l} d_q(D_f(x_i), D_f(x_j))},$$

one can also use Sliced Wasserstein for speedup.

Example: filter selection

[*Optimizing persistent homology based functions*, C., Chazal, Glisse, Ike, Kanna, Umeda, ICML, 2021]

Assume we have a supervised classification task. The goal is to find a filtration from a family $\mathcal{F}$ such that the corresponding persistence diagrams give the best classification score.

Idea: minimize:

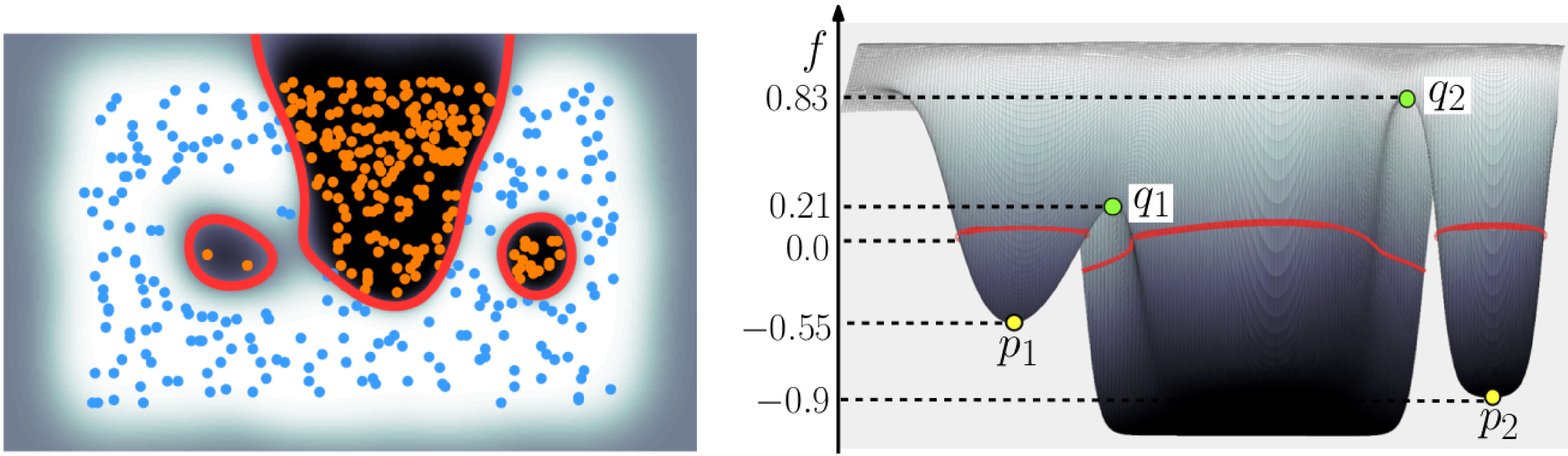
$$\mathcal{L}(f) = \sum_{l} \frac{\sum_{y_i = y_j = l} d_q(D_f(x_i), D_f(x_j))}{\sum_{y_i = l} d_q(D_f(x_i), D_f(x_j))},$$

one can also use Sliced Wasserstein for speedup.

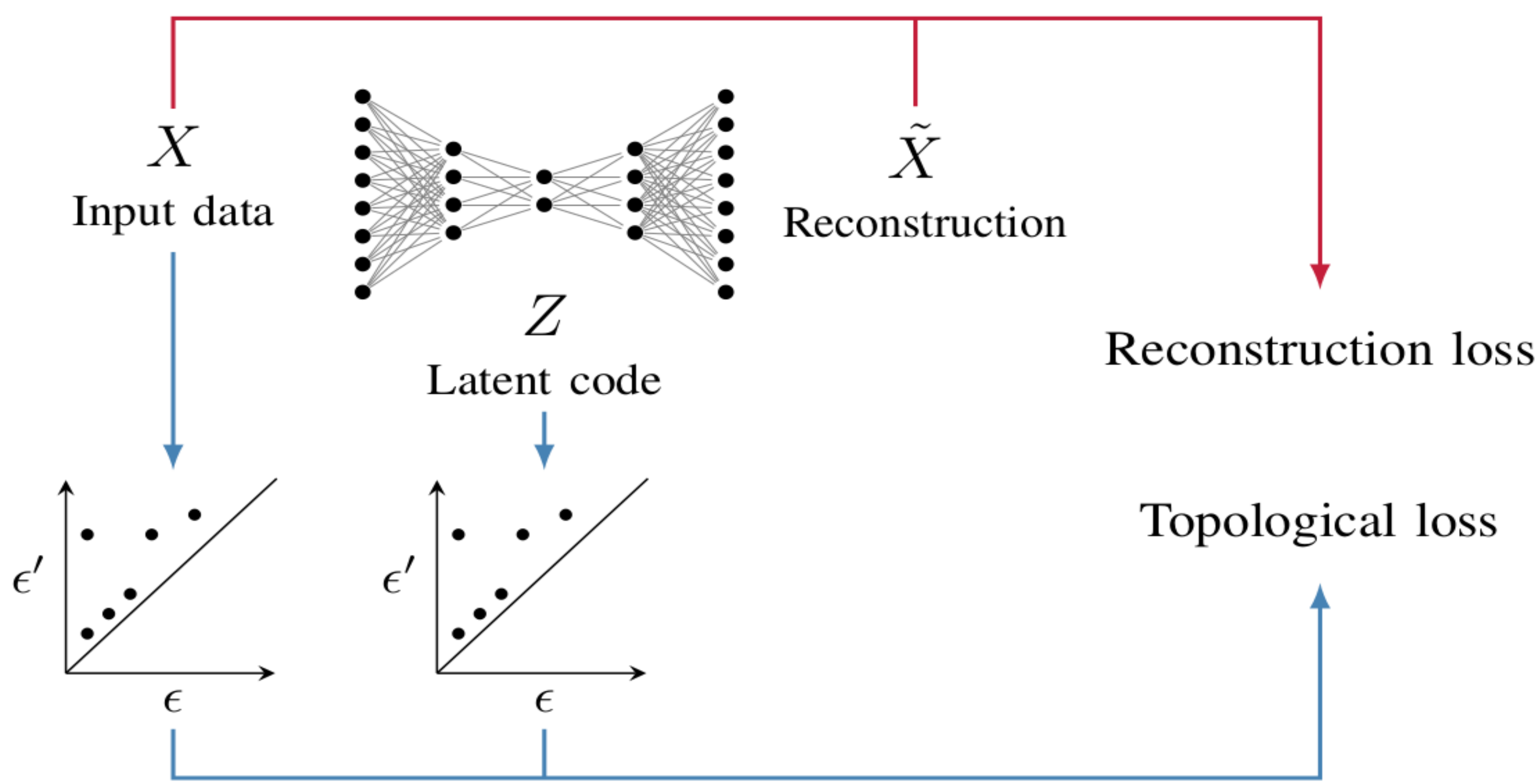
Dataset	Baseline	Before	After	Difference	Dataset	Baseline	Before	After	Difference
vs01	100.0	61.3	99.0	**+37.6**	vs26	99.7	98.8	98.2	-0.6
vs02	99.4	98.8	97.2	-1.6	vs28	99.1	96.8	96.8	0.0
vs06	99.4	87.3	98.2	**+10.9**	vs29	99.1	91.6	98.6	**+7.0**
vs09	99.4	86.8	98.3	**+11.5**	vs34	99.8	99.4	99.1	-0.3
vs16	99.7	89.0	97.3	**+8.3**	vs36	99.7	99.3	99.3	-0.1
vs19	99.6	84.8	98.0	**+13.2**	vs37	98.9	94.9	97.5	**+2.6**
vs24	99.4	98.7	98.7	0.0	vs57	99.7	90.5	97.2	**+6.7**
vs25	99.4	80.6	97.2	**+16.6**	vs79	99.1	85.3	96.9	**+11.5**

More examples

[*A Topological Regularizer for Classifiers via Persistent Homology*, Chen, Ni, Bai, Wang, AISTATS, 2019]



[*Topological autoencoders*, Moor, Horn, Rieck, Borgwardt, ICML, 2020]



Some limitations

Still, this gradient definition has some weaknesses:

- at most two simplices are updated for each PD point
$\longrightarrow$ the gradient is very sparse
- nothing has been said about the smoothness/speed of convergence
- no stopping criterion

Some limitations

[*A gradient sampling algorithm for stratified maps with applications to topological data analysis*, Leygonie, C., Lacombe, Oudot, 2021]

Still, this gradient definition has some weaknesses:

- at most two simplices are updated for each PD point
  $\longrightarrow$ the gradient is very sparse
- nothing has been said about the smoothness/speed of convergence
- no stopping criterion

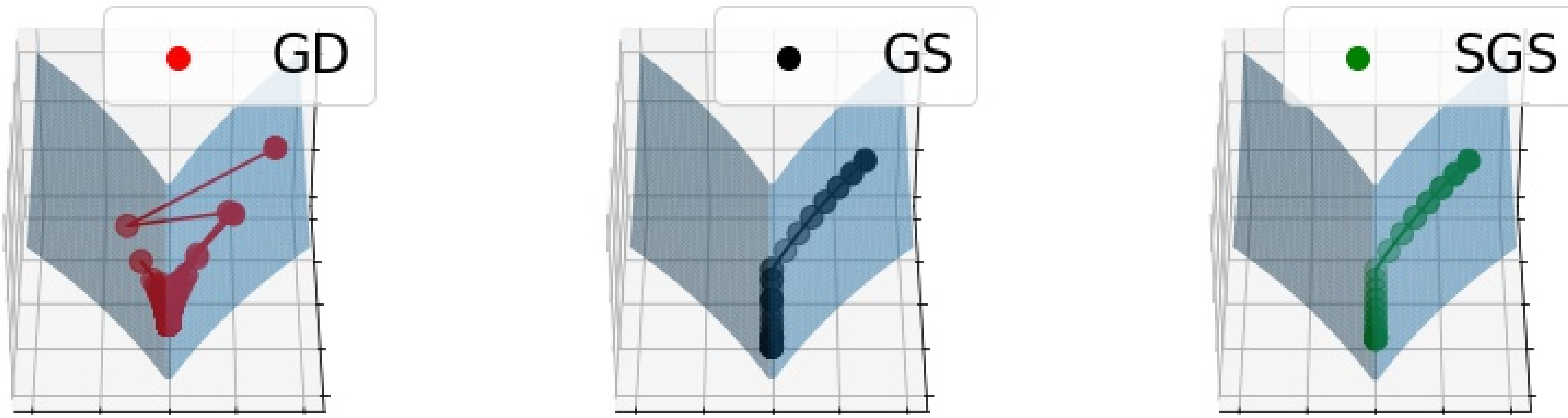
These problems can be tackled using the persistence map stratification!

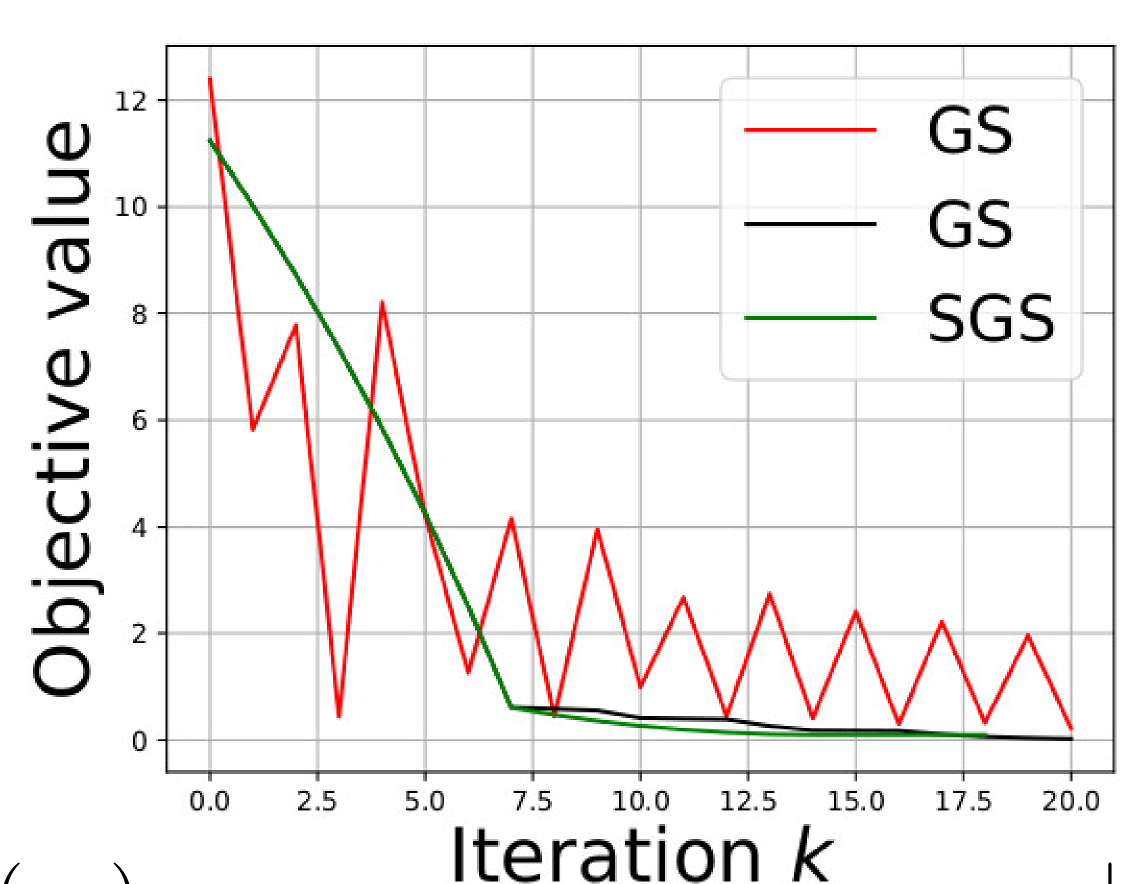
Gradient Sampling (GS) method computes current gradient by collecting the gradients at randomly sampled point around the current estimate.

We define *Stratified Gradient Sampling* (SGS) in a similar way, except we sample points in neighboring strata of the current estimate.

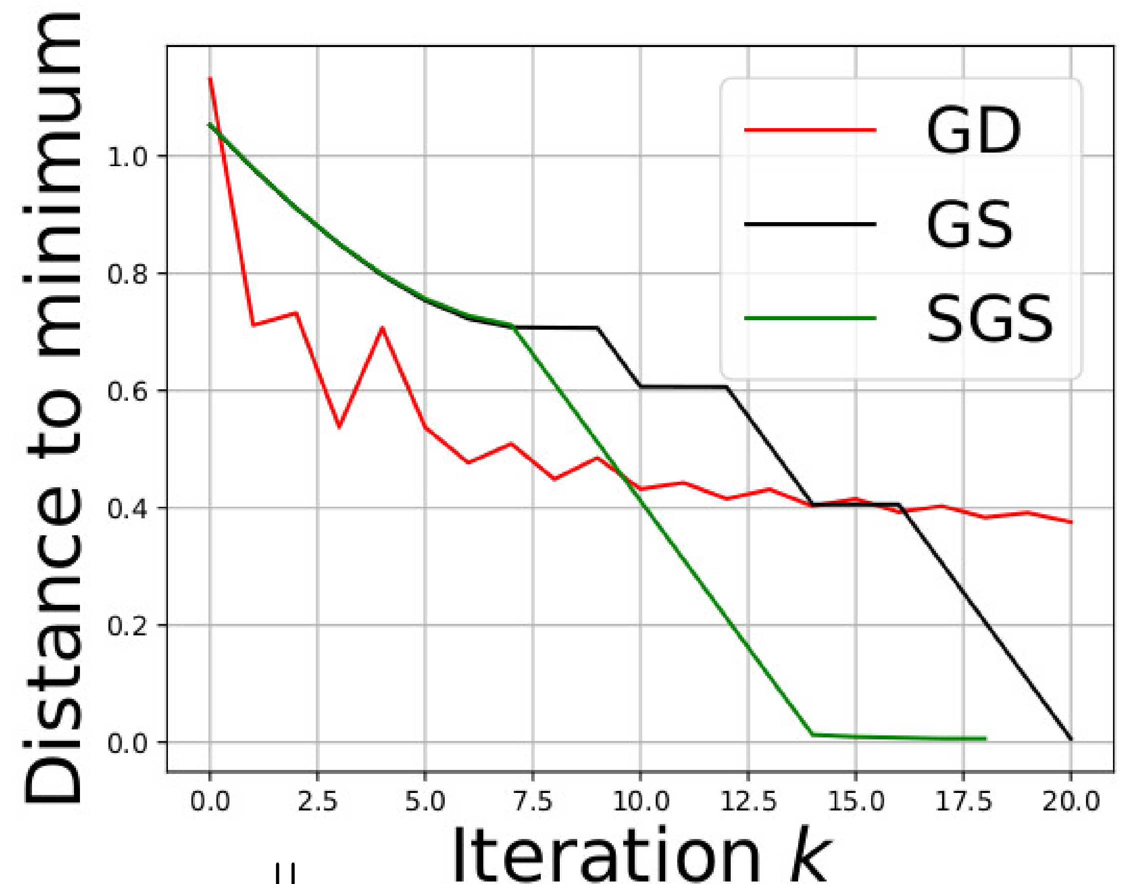
Some limitations

[*A gradient sampling algorithm for stratified maps with applications to topological data analysis*, Leygonie, C., Lacombe, Oudot, 2021]





$f(x_k)$



$\|x_k - x_*\|$

Approximate stationary points

[*A gradient sampling algorithm for stratified maps with applications to topological data analysis*, Leygonie, C., Lacombe, Oudot, 2021]

Def: The *Goldstein subgradient* is a relaxation of the Clarke subdifferential defined, for $\epsilon > 0$, as:

$$\partial_\epsilon \mathcal{L}(x) := \mathrm{Conv}\{\lim_{x_i \to x'} \nabla \mathcal{L}(x_i) \ : \ \mathcal{L} \text{ is differentiable at } x_i \text{ and } \|x - x'\| \leq \epsilon\}.$$

x is ϵ-stationary if $0 \in \partial_\epsilon \mathcal{L}(x)$

x is (ϵ, η)-stationary if $\|0 - \partial_\epsilon \mathcal{L}(x)\| := d(0, \partial_\epsilon \mathcal{L}(x)) \leq \eta$

Approximate stationary points

[*A gradient sampling algorithm for stratified maps with applications to topological data analysis*, Leygonie, C., Lacombe, Oudot, 2021]

Def: The *Goldstein subgradient* is a relaxation of the Clarke subdifferential defined, for $\epsilon > 0$, as:

$$\partial_\epsilon \mathcal{L}(x) := \mathrm{Conv}\{\lim_{x_i \to x'} \nabla \mathcal{L}(x_i) \ : \ \mathcal{L} \text{ is differentiable at } x_i \text{ and } \|x - x'\| \leq \epsilon\}.$$

x is ϵ-stationary if $0 \in \partial_\epsilon \mathcal{L}(x)$

x is (ϵ, η)-stationary if $\|0 - \partial_\epsilon \mathcal{L}(x)\| := d(0, \partial_\epsilon \mathcal{L}(x)) \leq \eta$

Def: Given a current estimate x_k, let $\mathcal{A}_{x_k,\epsilon} = \{A^1, \ldots, A^m\}$ denote the set of strata whose closure intersects $B(x_k, \epsilon)$ and define g_k as:

$$g_k^\epsilon = \mathrm{argmin}_{G_{x_k,\epsilon}} \|g\|^2, \text{ where } G_{x_k,\epsilon} = \mathrm{Conv}\{\nabla \mathcal{L}(x^i) : x^i \in A^i \cap B(x_k, \epsilon)\}.$$

Stratified Gradient Descent

[*A gradient sampling algorithm for stratified maps with applications to topological data analysis*, Leygonie, C., Lacombe, Oudot, 2021]

Let $C_{\mathcal{L}}$ be a Lipschitz constant for $\mathcal{L}$ and $\beta > 0$ a decrease rate.

Stratified gradient descent algorithm.

$$x_{k+1} = x_k - \epsilon_k \cdot g_k^{\epsilon_k} / \|g_k^{\epsilon_k}\|,$$

such that $\epsilon_k \leq ((1-\beta)/2C_{\mathcal{L}}) \cdot \|g_k^{\epsilon_k}\|$

Stratified Gradient Descent

[*A gradient sampling algorithm for stratified maps with applications to topological data analysis*, Leygonie, C., Lacombe, Oudot, 2021]

Let $C_{\mathcal{L}}$ be a Lipschitz constant for $\mathcal{L}$ and $\beta > 0$ a decrease rate.

Stratified gradient descent algorithm.

$$x_{k+1} = x_k - \epsilon_k \cdot g_k^{\epsilon_k} / \|g_k^{\epsilon_k}\|,$$

such that $\epsilon_k \leq ((1-\beta)/2C_{\mathcal{L}}) \cdot \|g_k^{\epsilon_k}\|$

always possible to find by progressively decreasing ϵ and recomputing g_k^{ϵ} (if condition is not satisfied at first)

Stratified Gradient Descent

[*A gradient sampling algorithm for stratified maps with applications to topological data analysis*, Leygonie, C., Lacombe, Oudot, 2021]

Let $C_{\mathcal{L}}$ be a Lipschitz constant for $\mathcal{L}$ and $\beta > 0$ a decrease rate.

Stratified gradient descent algorithm.

$$x_{k+1} = x_k - \epsilon_k \cdot g_k^{\epsilon_k} / \|g_k^{\epsilon_k}\|,$$

such that $\epsilon_k \leq ((1-\beta)/2C_{\mathcal{L}}) \cdot \|g_k^{\epsilon_k}\|$

Prop: One has $\mathcal{L}(x_{k+1}) \leq \mathcal{L}(x_k) - \beta\epsilon_k \|g_k^{\epsilon_k}\|$

Stratified Gradient Descent

[*A gradient sampling algorithm for stratified maps with applications to topological data analysis*, Leygonie, C., Lacombe, Oudot, 2021]

Let $C_{\mathcal{L}}$ be a Lipschitz constant for $\mathcal{L}$ and $\beta > 0$ a decrease rate.

Stratified gradient descent algorithm.

$$x_{k+1} = x_k - \epsilon_k \cdot g_k^{\epsilon_k} / \|g_k^{\epsilon_k}\|,$$

such that $\epsilon_k \leq ((1-\beta)/2C_{\mathcal{L}}) \cdot \|g_k^{\epsilon_k}\|$

Prop: One has $\mathcal{L}(x_{k+1}) \leq \mathcal{L}(x_k) - \beta \epsilon_k \|g_k^{\epsilon_k}\|$

Thm: Using stopping criterion $\|g_k^{\epsilon_k}\| \leq \eta$, one has that SGS converges to (ϵ, η)-stationary point in $O(1/(\eta \cdot \min\{\eta, \epsilon\}))$ iterations.

Stratified Gradient Descent

[*A gradient sampling algorithm for stratified maps with applications to topological data analysis*, Leygonie, C., Lacombe, Oudot, 2021]

Let $C_{\mathcal{L}}$ be a Lipschitz constant for $\mathcal{L}$ and $\beta > 0$ a decrease rate.

Stratified gradient descent algorithm.

$$x_{k+1} = x_k - \epsilon_k \cdot g_k^{\epsilon_k} / \|g_k^{\epsilon_k}\|,$$

such that $\epsilon_k \leq ((1-\beta)/2C_{\mathcal{L}}) \cdot \|g_k^{\epsilon_k}\|$

Prop: One has $\mathcal{L}(x_{k+1}) \leq \mathcal{L}(x_k) - \beta\epsilon_k \|g_k^{\epsilon_k}\|$

Thm: Using stopping criterion $\|g_k^{\epsilon_k}\| \leq \eta$, one has that SGS converges to (ϵ, η)-stationary point in $O(1/(\eta \cdot \min\{\eta, \epsilon\}))$ iterations.

Thm: Using stopping criterion $\|g_k^{\epsilon_k}\| = 0$, one has that SGS converges to ϵ-stationary point in finitely many iterations.

Application to Persistence Diagrams

[*A gradient sampling algorithm for stratified maps with applications to topological data analysis*, Leygonie, C., Lacombe, Oudot, 2021]

Recall that strata correspond to orderings of the simplices. Hence, we can use graph traversal of the Cayley graph of permutations to explore neighboring strata.

Application to Persistence Diagrams

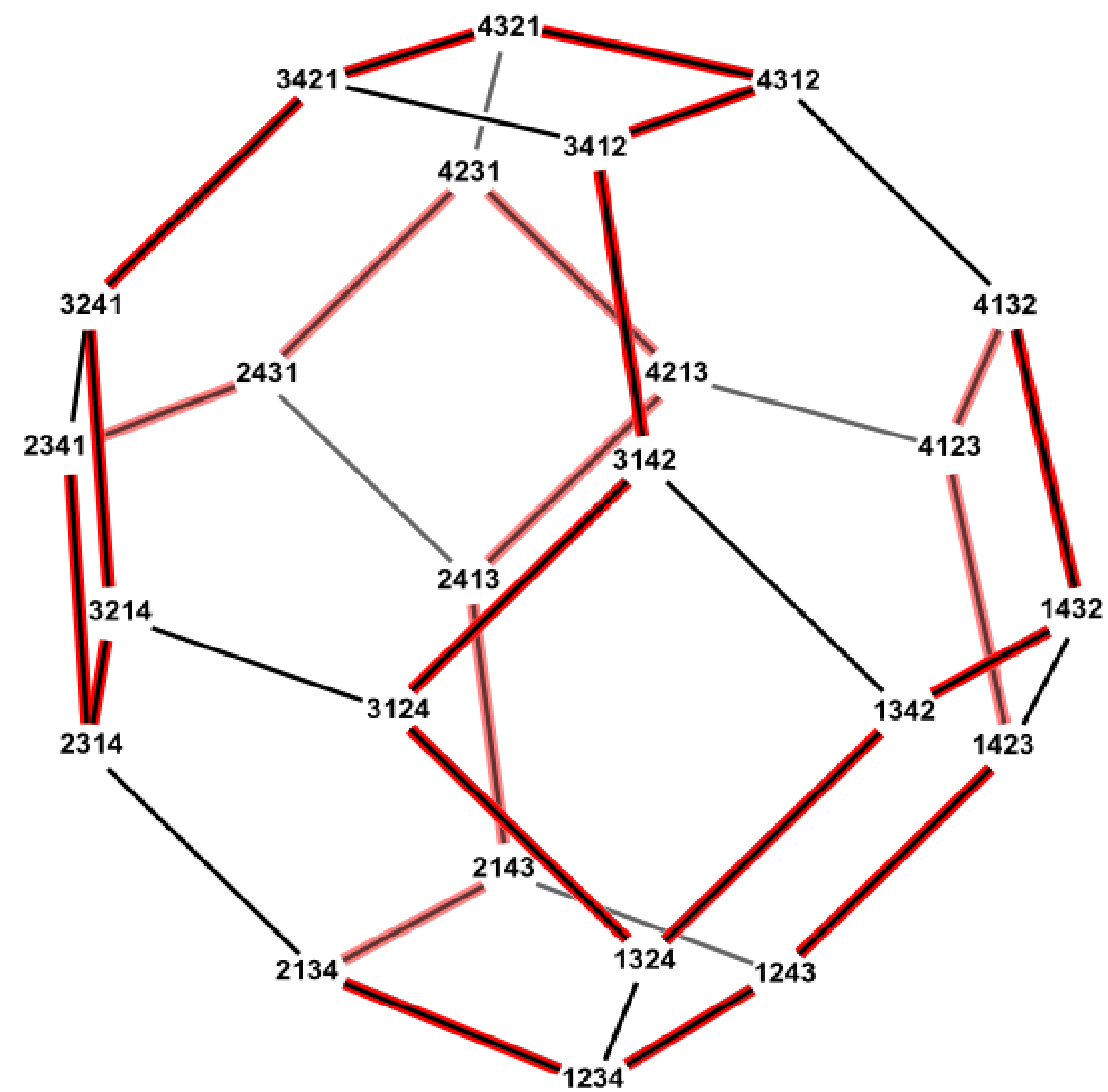
[*A gradient sampling algorithm for stratified maps with applications to topological data analysis*, Leygonie, C., Lacombe, Oudot, 2021]

Recall that strata correspond to orderings of the simplices. Hence, we can use graph traversal of the Cayley graph of permutations to explore neighboring strata.

Each node corresponds to an ordering of the simplices.

We sample the strata by simply swapping the filtration values of the current estimate.

Distance increases along paths so we can collect strata by increasing distance using Dijkstra.



Application to Persistence Diagrams

[*A gradient sampling algorithm for stratified maps with applications to topological data analysis*, Leygonie, C., Lacombe, Oudot, 2021]

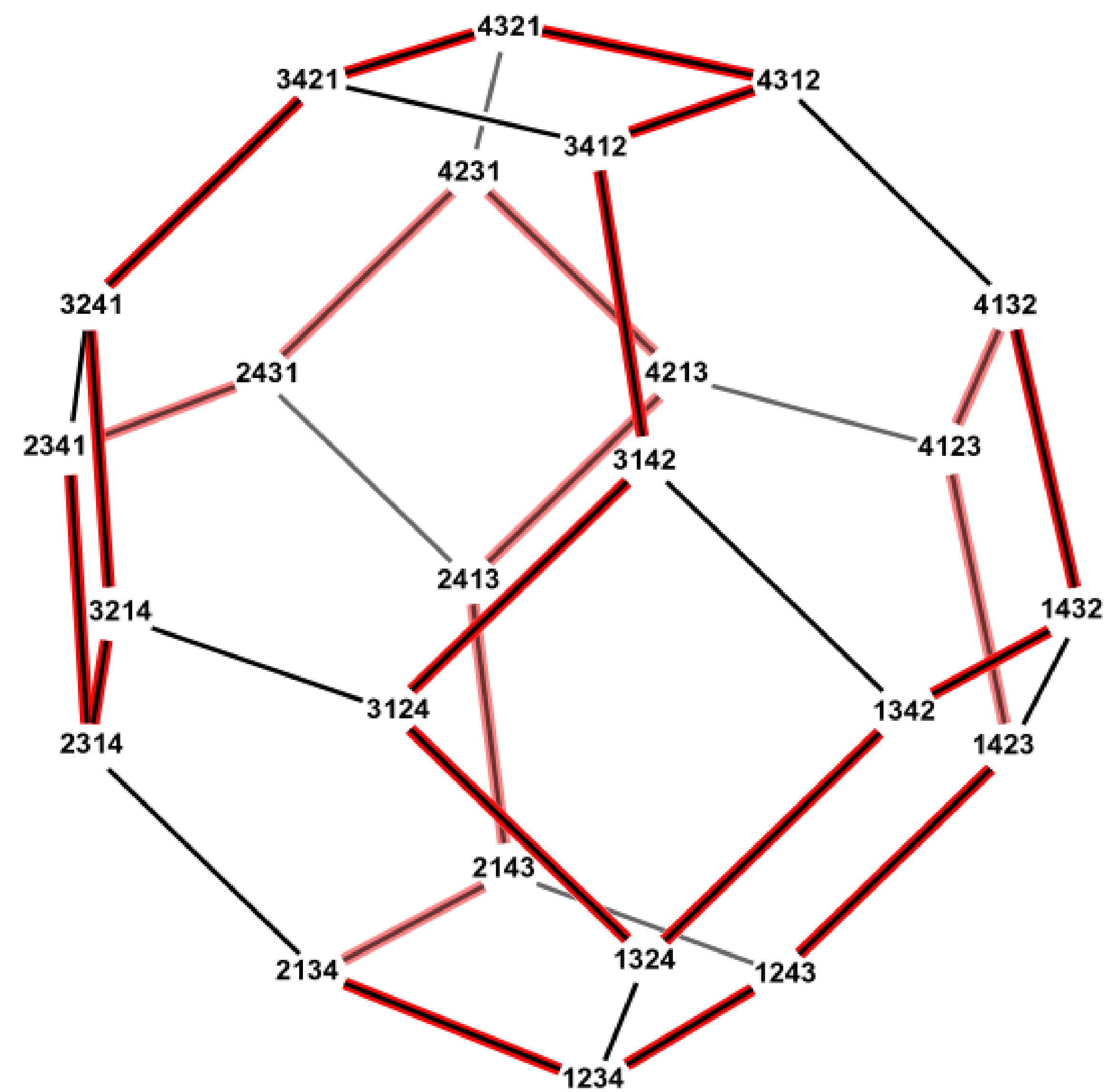
Recall that strata correspond to orderings of the simplices. Hence, we can use graph traversal of the Cayley graph of permutations to explore neighboring strata.

Each node corresponds to an ordering of the simplices.

We sample the strata by simply swapping the filtration values of the current estimate.

Distance increases along paths so we can collect strata by increasing distance using Dijkstra.

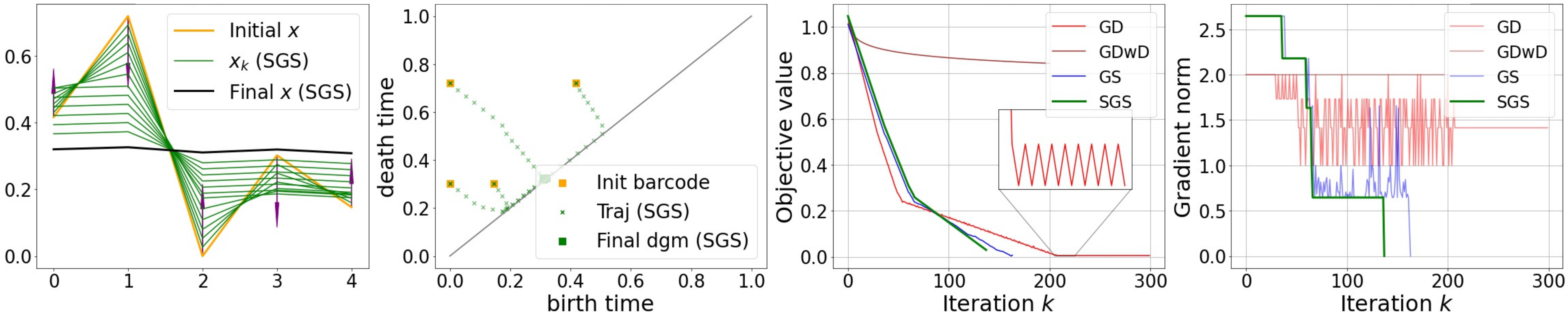
Other options include random walks, memoization...



Application to Persistence Diagrams

[*A gradient sampling algorithm for stratified maps with applications to topological data analysis*, Leygonie, C., Lacombe, Oudot, 2021]

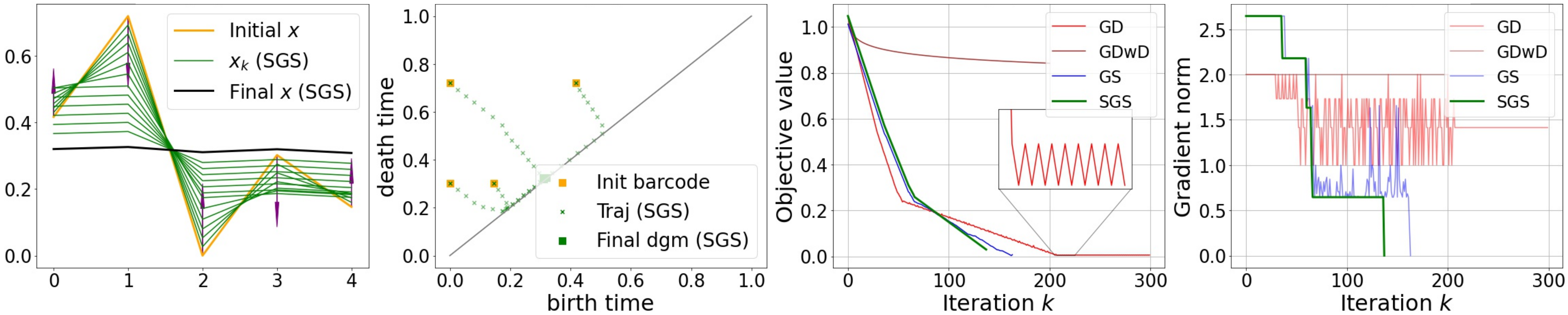
Minimization of total persistence of function.



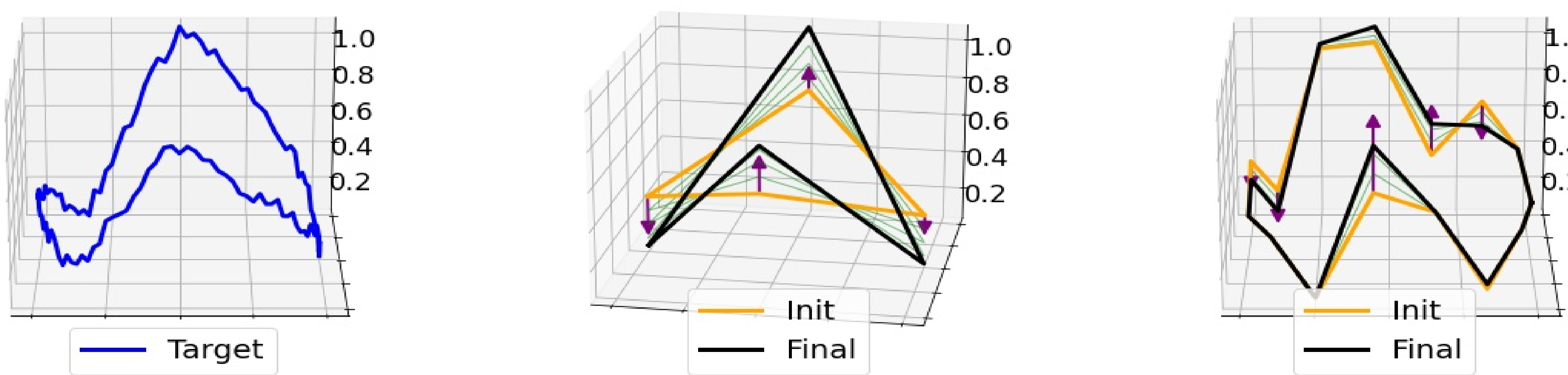
Application to Persistence Diagrams

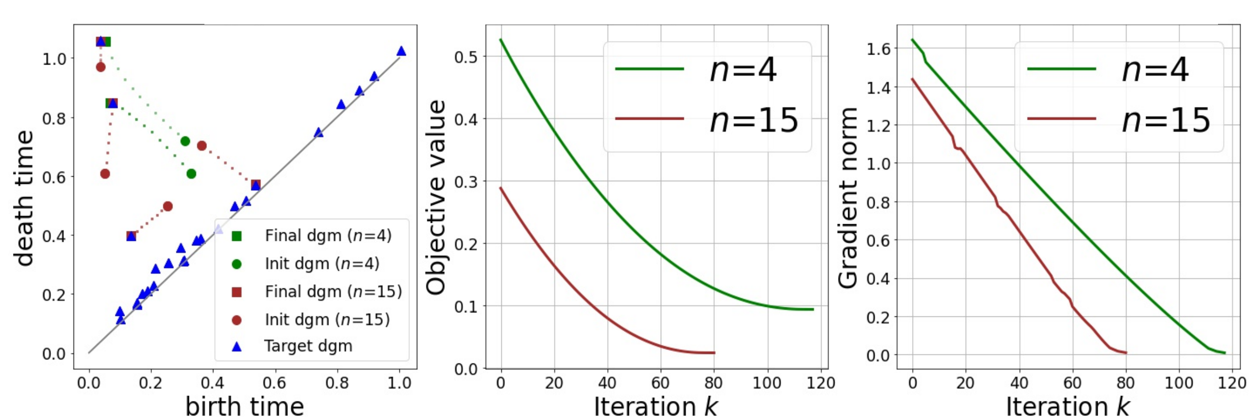
[*A gradient sampling algorithm for stratified maps with applications to topological data analysis*, Leygonie, C., Lacombe, Oudot, 2021]

Minimization of total persistence of function.



Registration: replicate a complex topology in smaller complexes.



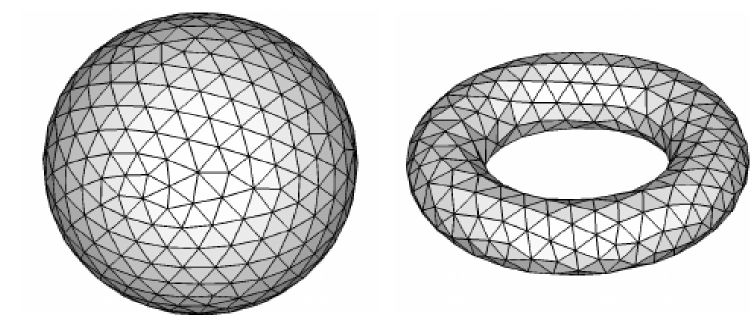


Take home message

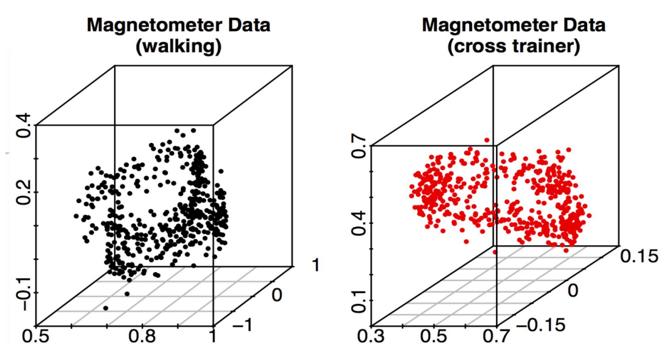
Topological Data Analysis is:

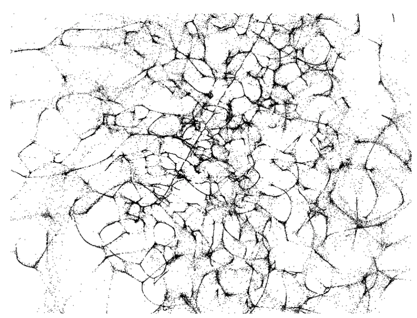
a mathematically grounded framework...

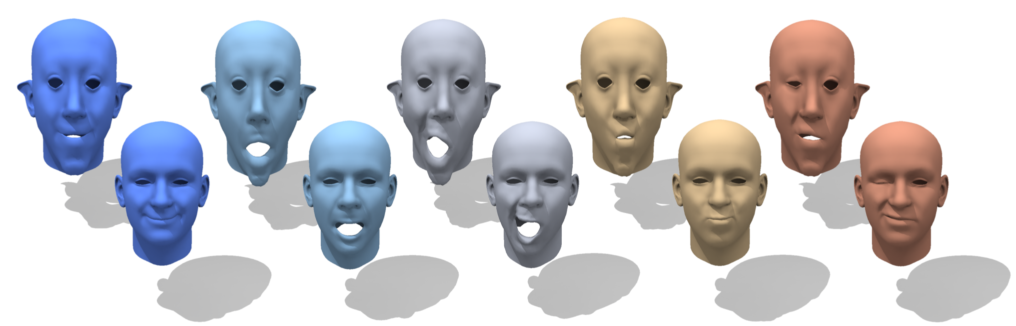
$$H_k = Z_k / B_k$$



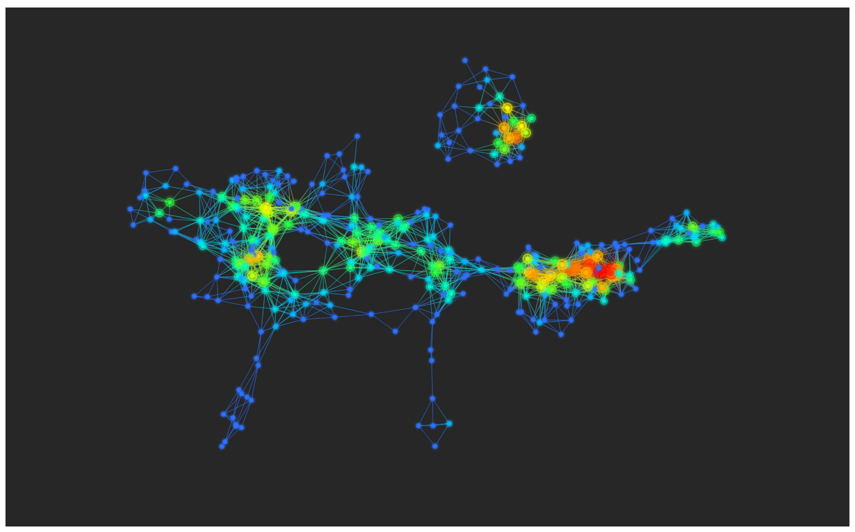
...that applies to a wide variety of data sets...



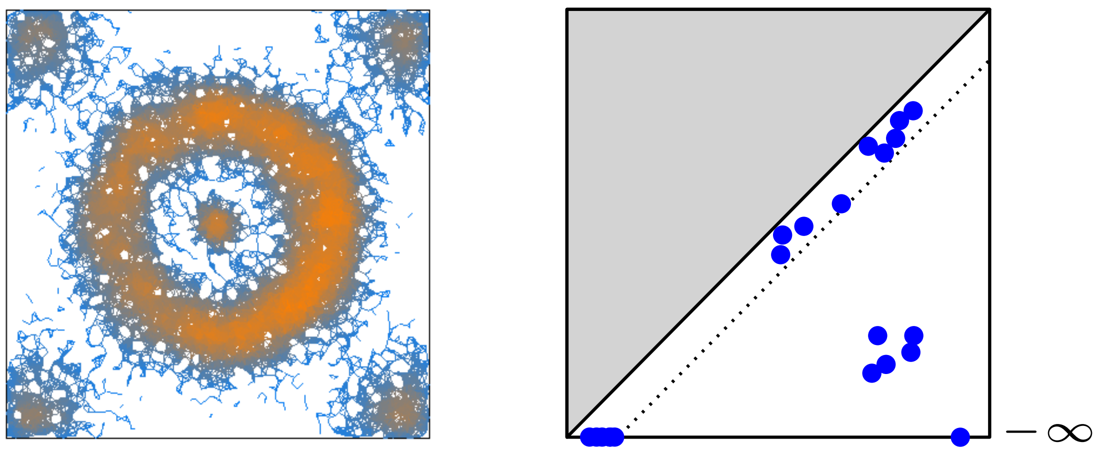




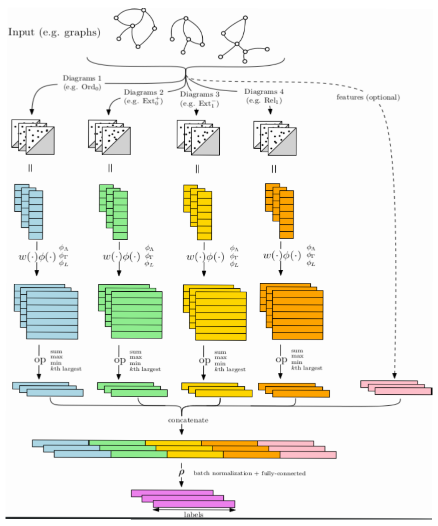
...for a wide variety of tasks.



Mapper: exploratory data analysis



ToMATo: clustering



Persistence diagrams: machine learning

The Koonz, Narendra and Fukunaga algorithm (1976)

The algorithm:

Input: neighborhood graph G with n vertices (the data points) and a n-dimensional vector $\hat{f}$ (density estimate)

Sort the vertex indices $\{1, 2, \ldots, n\}$ in decreasing order: $\hat{f}(1) \geq \hat{f}(2) \geq \cdots \geq \hat{f}(n)$;
Initialize a union-find data structure (disjoint-set forest) $\mathcal{U}$ and two vectors g, r of size n;

for $i = 1$ to n **do**
  Let N be the set of neighbors of i in G that have indices higher than i;
  if $N = \emptyset$
    Create a new entry e in $\mathcal{U}$ and attach vertex i to it: $\mathcal{U}$.`MakeSet`(i);
    $r(e) \leftarrow i$ *// $r(e)$ stores the root vertex associated with the entry e*
  else
    $g(i) \leftarrow \text{argmax}_{j \in \mathcal{N}} \hat{f}(j)$ *// $g(i)$ stores the approximate gradient at vertex i*
    $e_i \leftarrow \mathcal{U}.\texttt{Find}(g(i))$;
    Attach vertex i to the entry e_i: $\mathcal{U}.\texttt{Union}(i, e_i)$;

Output: the collection of entries e in $\mathcal{U}$

ToMATo Pseudo-code

Input: simple graph G with n vertices, n-dimensional vector $\hat{f}$, real parameter $\tau \geq 0$.

Sort the vertex indices $\{1, 2, \cdots, n\}$ so that $\hat{f}(1) \geq \hat{f}(2) \geq \cdots \geq \hat{f}(n)$;
Initialize a union-find data structure $\mathcal{U}$ and two vectors g, r of size n;

for $i = 1$ to n **do**
    Let $\mathcal{N}$ be the set of neighbors of i in G that have indices lower than i;
    if $\mathcal{N} = \emptyset$ *// vertex i is a peak of $\hat{f}$ within G*
        Create a new entry e in $\mathcal{U}$ and attach vertex i to it: $\mathcal{U}.\texttt{MakeSet}(i)$;
        $r(e) \leftarrow i$ *// $r(e)$ stores the root vertex associated with the entry e*
    else *// vertex i is not a peak of $\hat{f}$ within G*
        $g(i) \leftarrow \mathrm{argmax}_{j \in \mathcal{N}} \hat{f}(j)$ *// $g(i)$ stores the approximate gradient at vertex i*
        $e_i \leftarrow \mathcal{U}.\texttt{Find}(g(i))$;
        Attach vertex i to the entry e_i: $\mathcal{U}.\texttt{Union}(i, e_i)$;

graph-based hill-climbing (1976)

        for $j \in \mathcal{N}$ **do**
            $e \leftarrow \mathcal{U}.\texttt{Find}(j)$;
            if $e \neq e_i$ and $\min\{\hat{f}(r(e)),\ \hat{f}(r(e_i))\} < \hat{f}(i) + \tau$
                $\mathcal{U}.\texttt{Union}(e,\ e_i)$;
                $r(e \cup e_i) \leftarrow \mathrm{argmax}_{\{r(e),\ r(e_i)\}} \hat{f}$;
                $e_i \leftarrow e \cup e_i$;

cluster merges with persistence (2013)

Output: the collection of entries e of $\mathcal{U}$ such that $\hat{f}(r(e)) \geq \tau$.

Complexity

Given a neighborhood graph with n vertices (with density values) and m edges:

1. the algorithm sorts the vertices by decreasing density values,

2. the algorithm makes a single pass through the vertex set, creating the spanning forest and merging clusters on the fly using a union-find data structure.

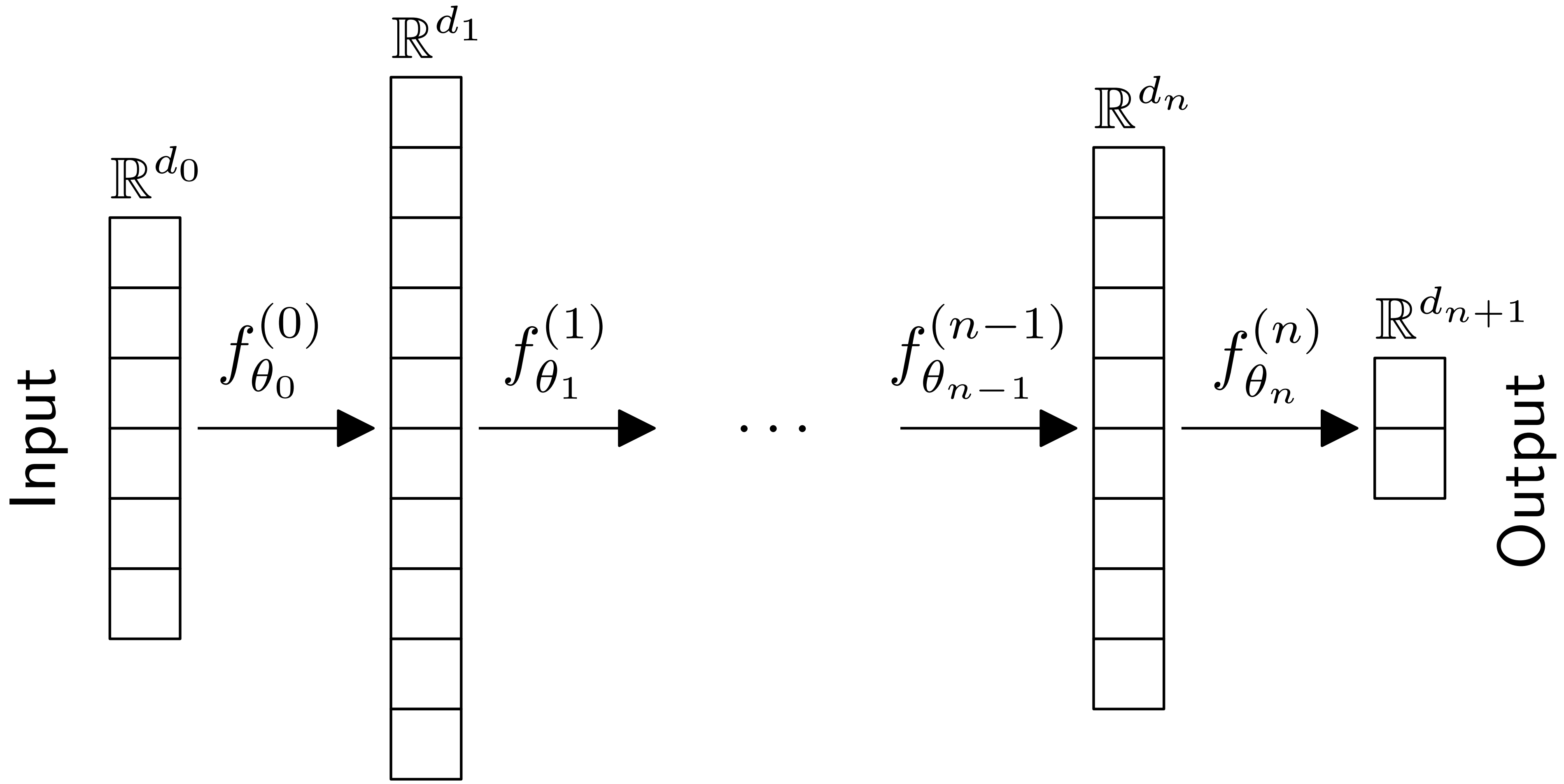
$\rightarrow$ Running time: $O(n \log n + (n+m)\alpha(n))$

$\rightarrow$ Space complexity: $O(n+m)$

$\rightarrow$ Main memory usage: $O(n)$

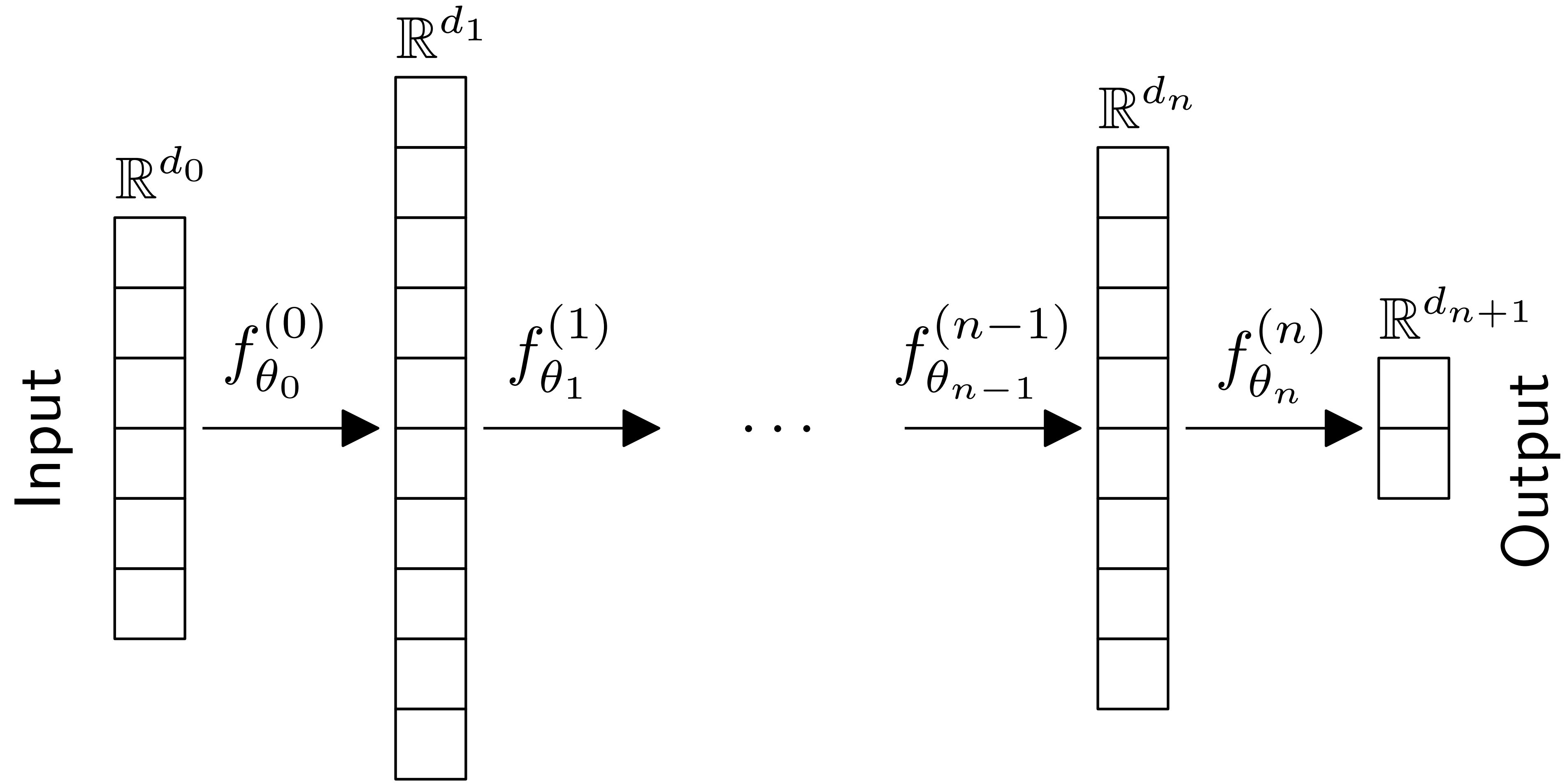
Reminder

Neural network with depth $n \in \mathbb{N}^*$



Reminder

Neural network with depth $n \in \mathbb{N}^*$



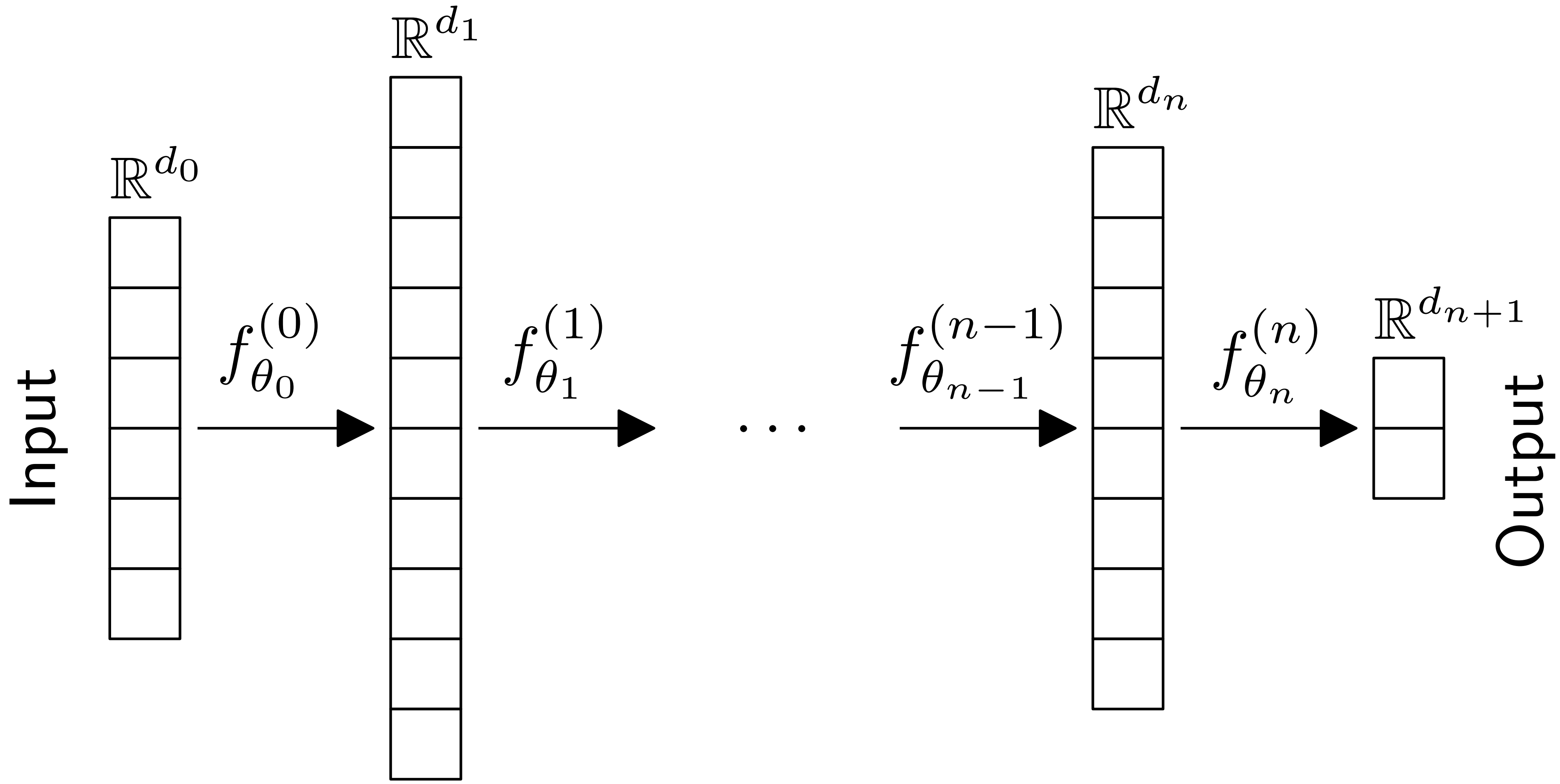
$$\theta_k = (W_k \in \mathbb{R}^{d_{k+1} \times d_k},\ b_k \in \mathbb{R}^{d_{k+1}}), \quad \sigma : x \mapsto \max(0, x) \text{ or } (1 + \mathrm{e}^{-x})^{-1}$$

$$f^{(k)}_{\theta_k} : x \in \mathbb{R}^{d_k} \mapsto \sigma(W_k \cdot x + b_k) \in \mathbb{R}^{d_{k+1}}$$

Final classifier: $f_\theta = f^{(n)}_{\theta_n} \circ \cdots \circ f^{(0)}_{\theta_0}$

Reminder

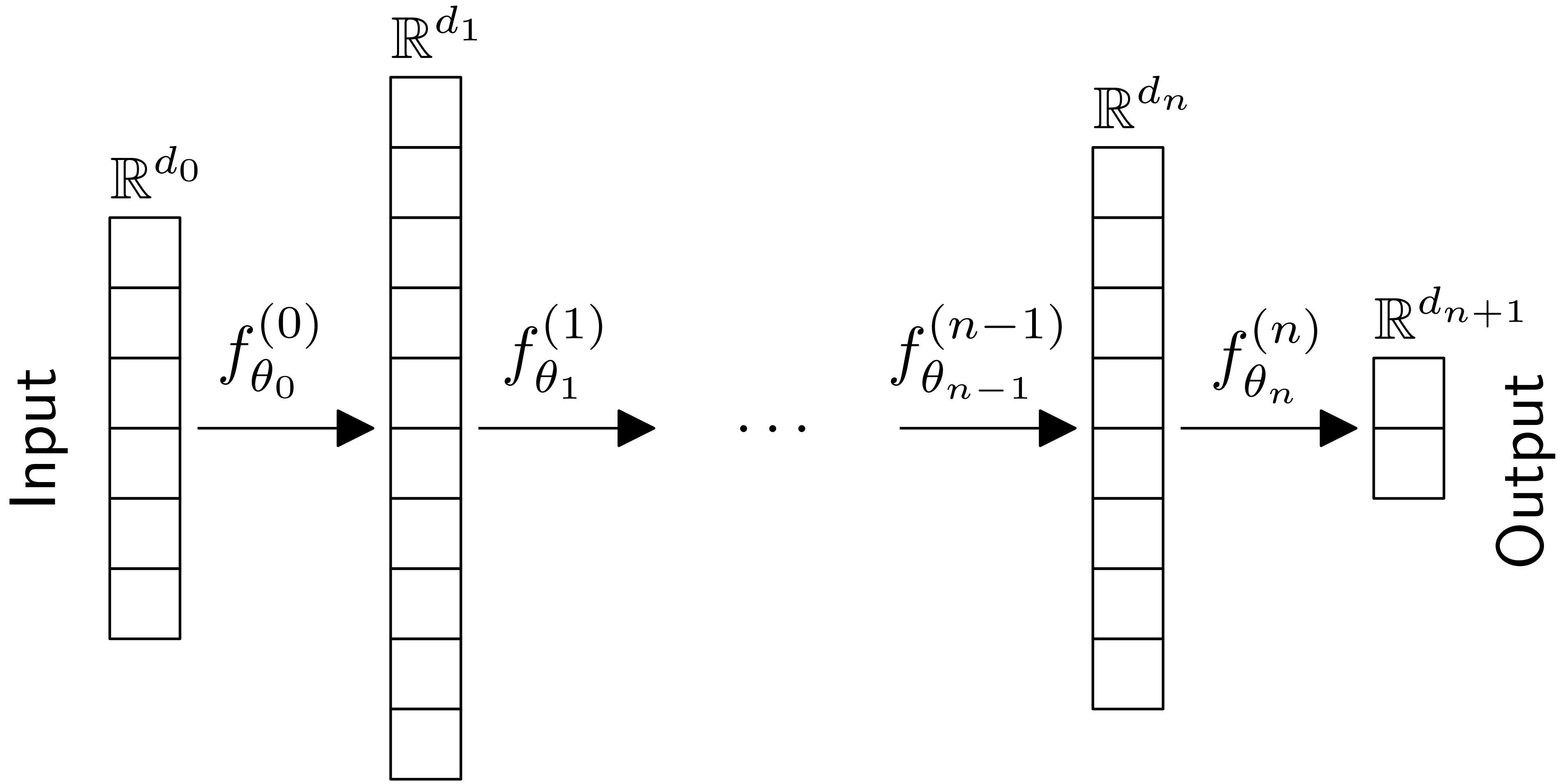
Neural network with depth $n \in \mathbb{N}^*$



Goal: Minimize $\ell(\theta) = \sum_i \|f_\theta(x_i) - y_i\|_2^2$ w.r.t. θ

Reminder

Neural network with depth $n \in \mathbb{N}^*$



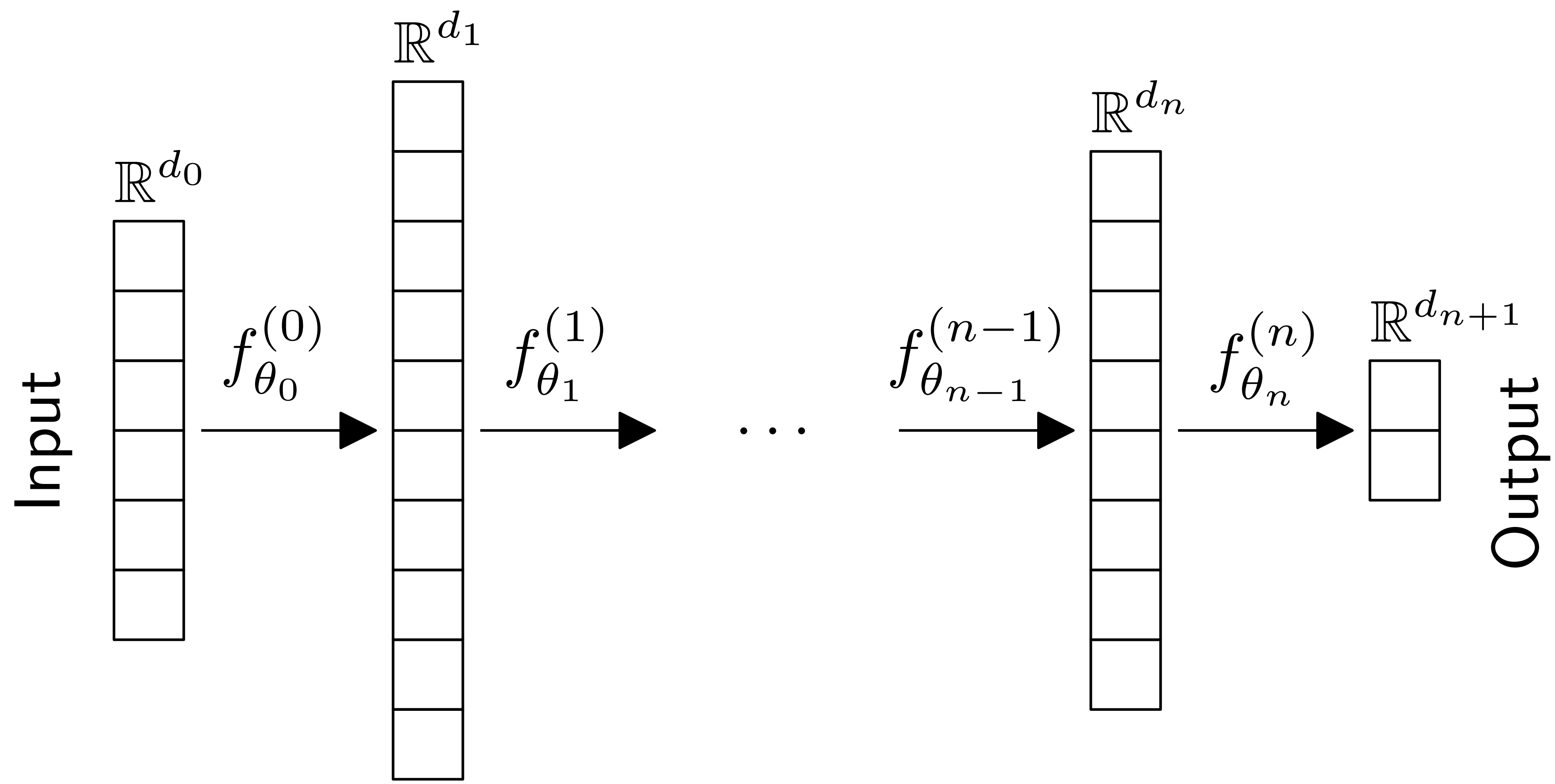
Goal: Minimize $\ell(\theta) = \sum_i \|f_\theta(x_i) - y_i\|_2^2$ w.r.t. θ

Backpropagation: for each k:

1. compute $\nabla \ell(\theta_k)$ with chain rule
2. update $\theta_k := \theta_k - \eta \nabla \ell(\theta_k)$

Reminder

Neural network with depth $n \in \mathbb{N}^*$



Goal: Minimize $\ell(\theta) = \sum_i \|f_\theta(x_i) - y_i\|_2^2$ w.r.t. θ

Backpropagation: for each k:

1. compute $\nabla \ell(\theta_k)$ with chain rule
2. update $\theta_k := \theta_k - \eta \nabla \ell(\theta_k)$

Requirement: $f^{(k)}_{\theta_k}$ needs to be differentiable w.r.t. θ_k and x

Persistence Approximation and Robustness

[*RipsNet: a general architecture for fast and robust estimation of the persistent homology of point clouds*, de Surrel, Hensel, C., Lacombe, Ike, Kurihara, Glisse, Chazal, 2022]

Motivation

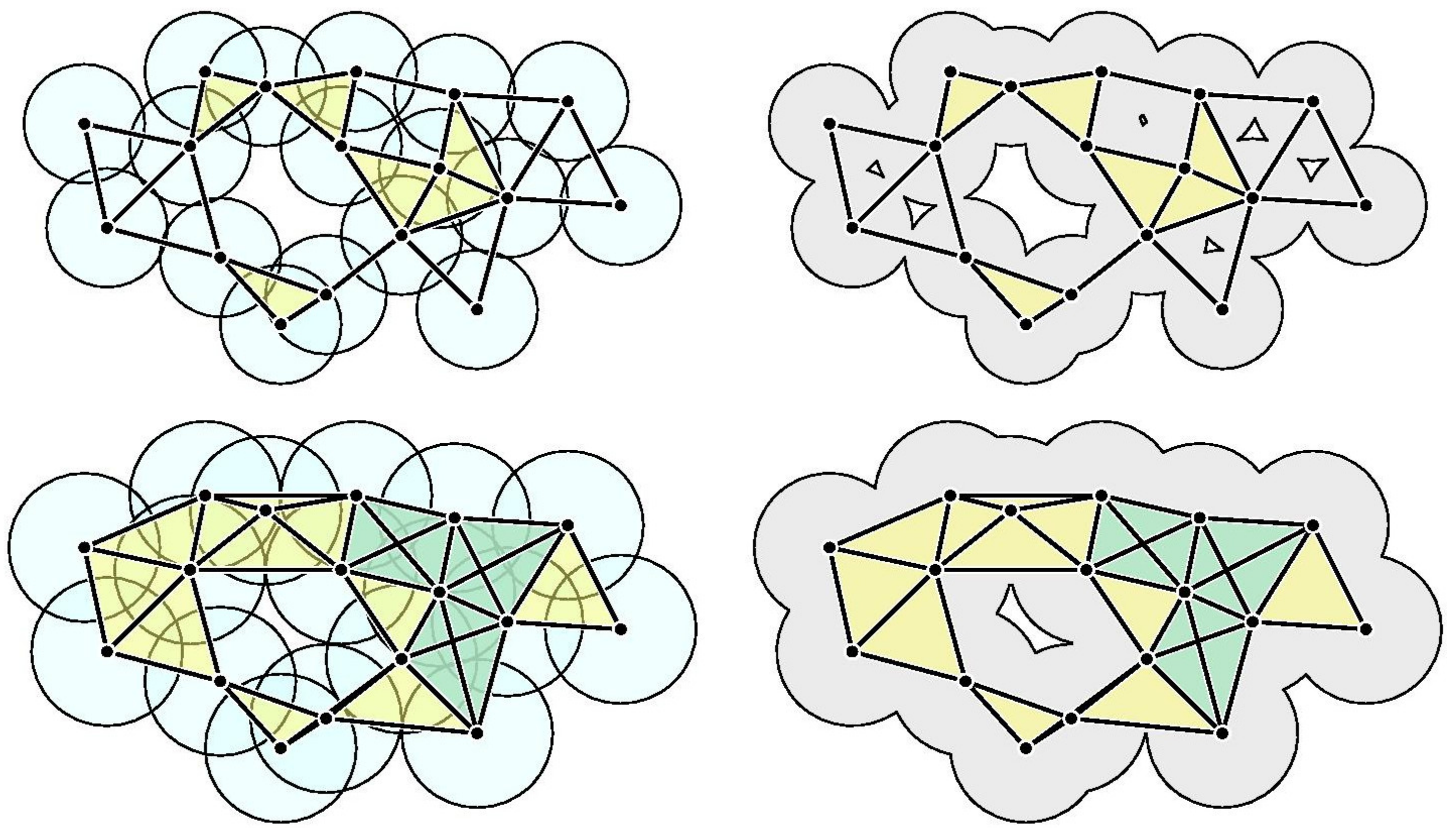
We now focus on the persistent homology of filtrations well suited for point clouds: (Vietoris-)Rips, Čech, Alpha.

Motivation

We now focus on the persistent homology of filtrations well suited for point clouds: (Vietoris-)Rips, Čech, Alpha.

Def: Given a point cloud $P = \{P_1, \ldots, P_n\} \subset \mathbb{R}^d$, its Čech complex of radius $r > 0$ is the abstract simplicial complex $C(P, r)$ s.t. $\mathrm{vert}(C(P, r)) = P$ and

$$\sigma = [P_{i_0}, P_{i_1}, \ldots, P_{i_k}] \in C(P, r) \quad \text{iif} \quad \cap_{j=0}^{k} B(P_{i_j}, r) \neq \emptyset.$$



Motivation

We now focus on the persistent homology of filtrations well suited for point clouds: (Vietoris-)Rips, Čech, Alpha.

Def: Given a point cloud $P = \{P_1, \ldots, P_n\} \subset \mathbb{R}^d$, its Čech complex of radius $r > 0$ is the abstract simplicial complex $C(P, r)$ s.t. $\mathrm{vert}(C(P, r)) = P$ and

$$\sigma = [P_{i_0}, P_{i_1}, \ldots, P_{i_k}] \in C(P, r) \quad \text{iif} \quad \cap_{j=0}^{k} B(P_{i_j}, r) \neq \emptyset.$$

Def: Given a point cloud $P = \{P_1, \ldots, P_n\} \subset \mathbb{R}^d$, its Rips complex of radius $r > 0$ is the abstract simplicial complex $R(P, r)$ s.t. $\mathrm{vert}(R(P, r)) = P$ and

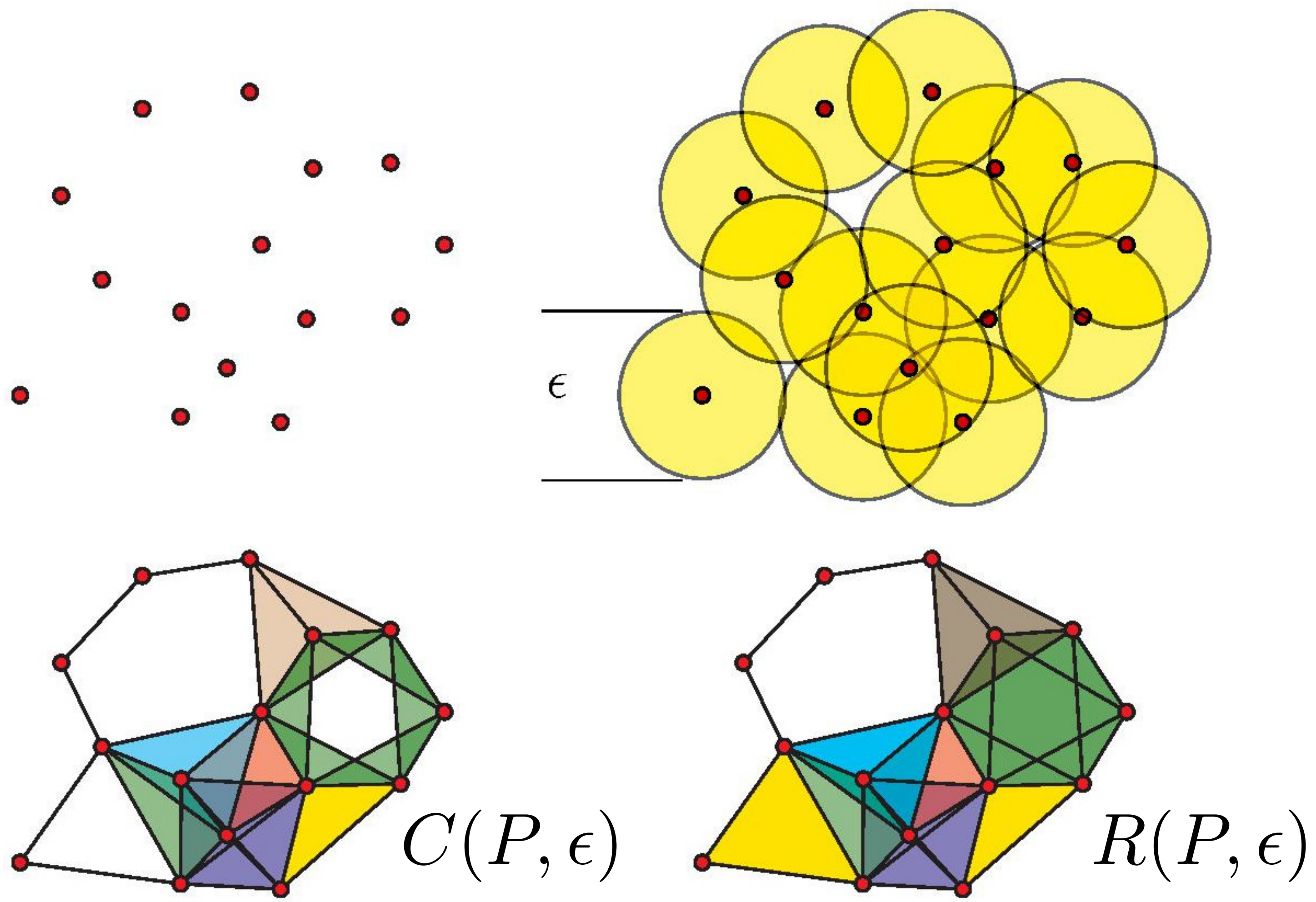
$$\sigma = [P_{i_0}, P_{i_1}, \ldots, P_{i_k}] \in R(P, r) \quad \text{iif} \quad \|P_{i_j} - P_{i_{j'}}\| \leq 2r, \forall 1 \leq j, j' \leq k.$$

Motivation

We now focus on the persistent homology of filtrations well suited for point clouds: (Vietoris-)Rips, Čech, Alpha.

Def: Given a point cloud $P = \{P_1, \ldots, P_n\} \subset \mathbb{R}^d$, its Čech complex of radius $r > 0$ is the abstract simplicial complex $C(P, r)$ s.t. $\mathrm{vert}(C(P, r)) = P$ and

$$\sigma = [P_{i_0}, P_{i_1}, \ldots, P_{i_k}] \in C(P, r) \quad \text{iif} \quad \cap_{j=0}^{k} B(P_{i_j}, r) \neq \emptyset.$$



Motivation

We now focus on the persistent homology of filtrations well suited for point clouds: (Vietoris-)Rips, Čech, Alpha.

Def: Given a point cloud $P = \{P_1, \ldots, P_n\} \subset \mathbb{R}^d$, its Čech complex of radius $r > 0$ is the abstract simplicial complex $C(P, r)$ s.t. $\mathrm{vert}(C(P, r)) = P$ and

$$\sigma = [P_{i_0}, P_{i_1}, \ldots, P_{i_k}] \in C(P, r) \ \text{ iif } \ \cap_{j=0}^{k} B(P_{i_j}, r) \neq \emptyset.$$

Def: Given a point cloud $P = \{P_1, \ldots, P_n\} \subset \mathbb{R}^d$, its Rips complex of radius $r > 0$ is the abstract simplicial complex $R(P, r)$ s.t. $\mathrm{vert}(R(P, r)) = P$ and

$$\sigma = [P_{i_0}, P_{i_1}, \ldots, P_{i_k}] \in R(P, r) \ \text{ iif } \ \|P_{i_j} - P_{i_{j'}}\| \leq 2r, \forall 1 \leq j, j' \leq k.$$

Def: Given a point cloud $P = \{P_1, \ldots, P_n\} \subset \mathbb{R}^d$, its Alpha complex of radius $r > 0$ is the abstract simplicial complex $A(P, r)$ s.t. $\mathrm{vert}(A(P, r)) = P$ and

$$\sigma = [P_{i_0}, P_{i_1}, \ldots, P_{i_k}] \in A(P, r) \ \text{ iif } \ \begin{array}{c} \sigma \text{ is a cell in the Delaunay triangulation of } P \\ \mathrm{ccsph}(\sigma) = \{P_{i_0}, \ldots, P_{i_k}\} \text{ and } \sqrt{\mathrm{ccrad}(\sigma)} \leq r \\ \mathrm{ccsph}(\sigma) \supset \{P_{i_0}, \ldots, P_{i_k}\} \text{ and } \sqrt{\mathrm{ccrad}(\tau)} \leq r \end{array}$$

Motivation

We now focus on the persistent homology of filtrations well suited for point clouds: (Vietoris-)Rips, Čech, Alpha.

These complexes are all related:

Prop: $R(P, r/2) \subseteq C(P, r) \subseteq R(P, r)$.

Prop: $A(P, r)$ and $C(P, r)$ are homotopy equivalent.

Motivation

We now focus on the persistent homology of filtrations well suited for point clouds: (Vietoris-)Rips, Čech, Alpha.

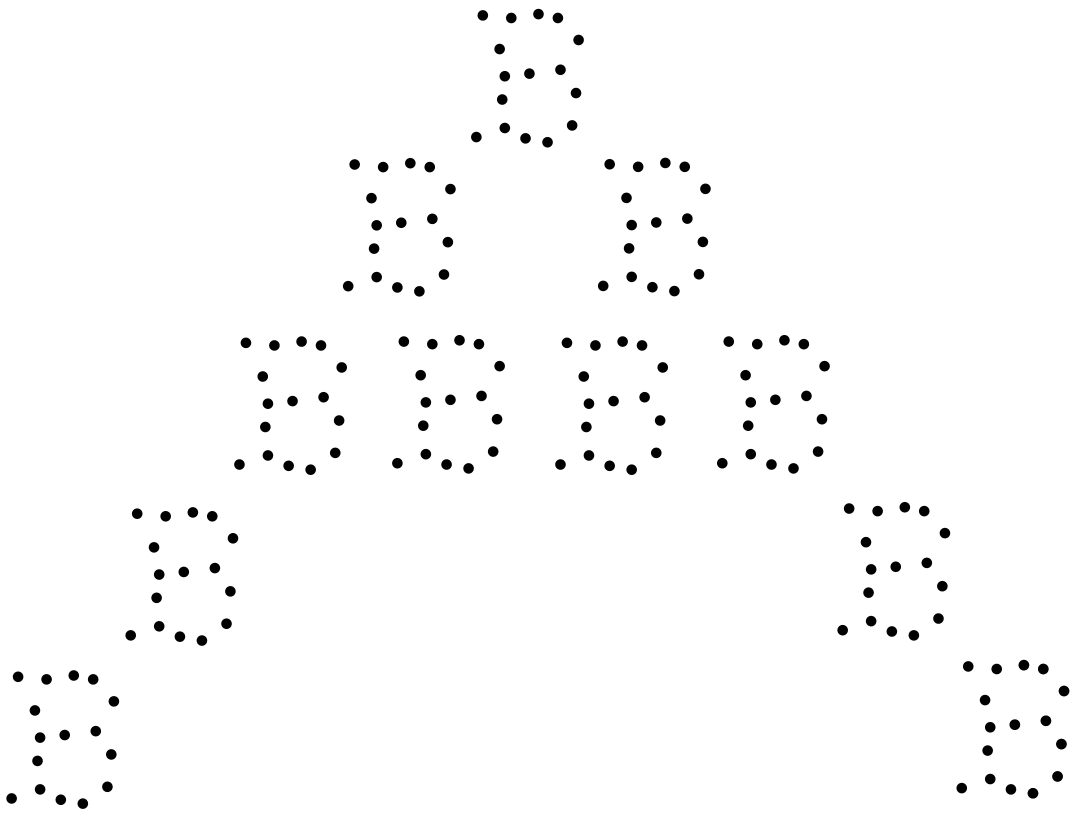
These complexes are all related:

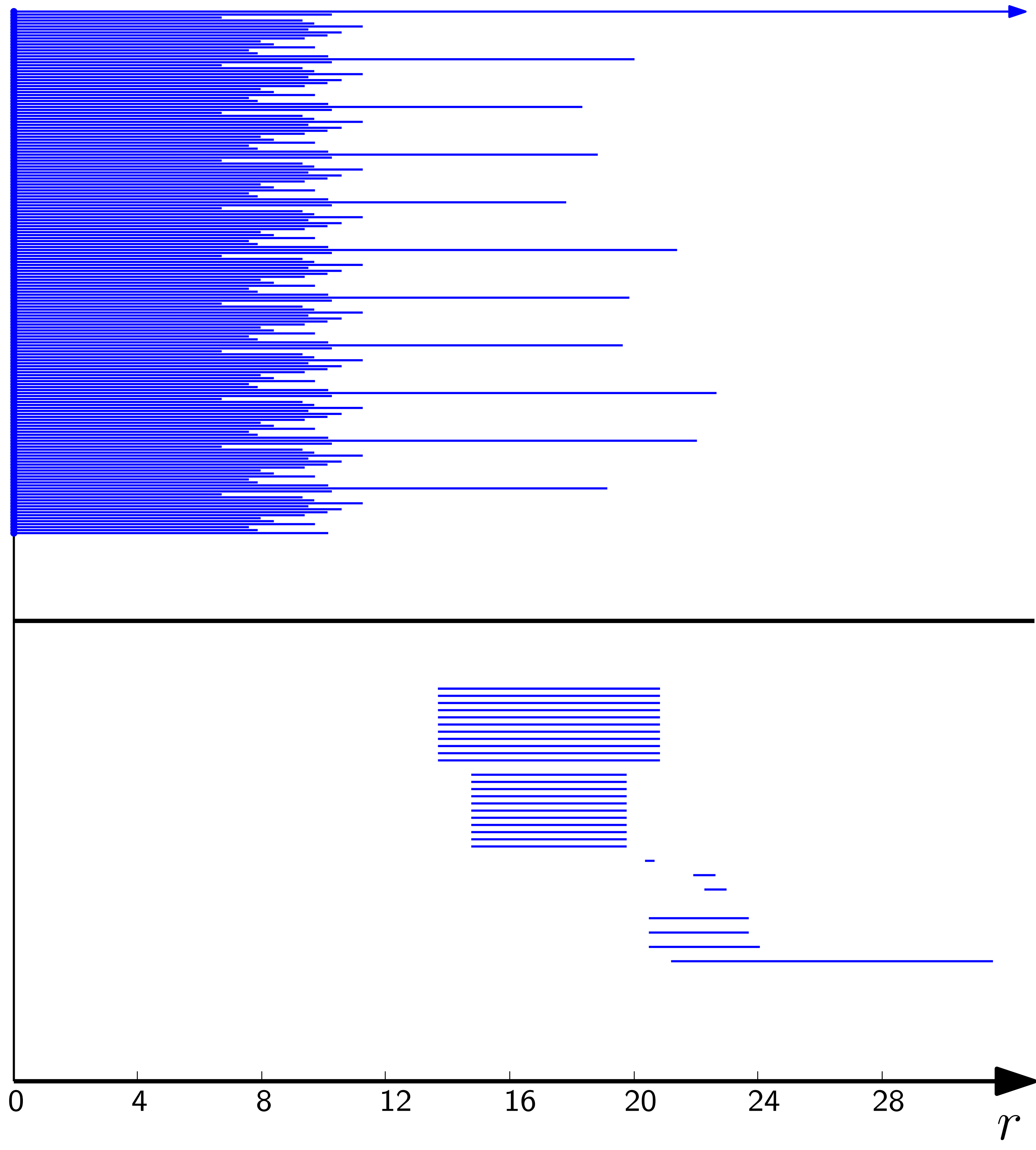
Prop: $R(P, r/2) \subseteq C(P, r) \subseteq R(P, r)$.

Prop: $A(P, r)$ and $C(P, r)$ are homotopy equivalent.

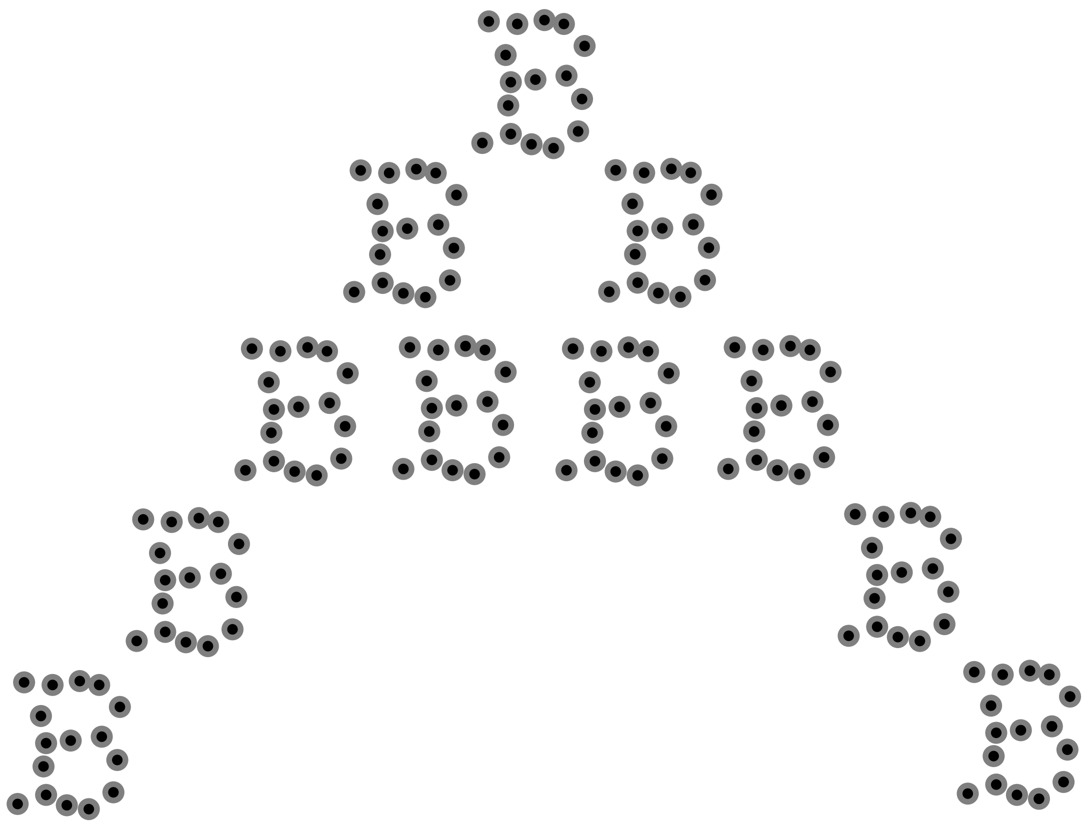
Morevover, their persistent homology are known to encode the geometric and topological features of the point cloud, which is quite useful for generating descriptors.

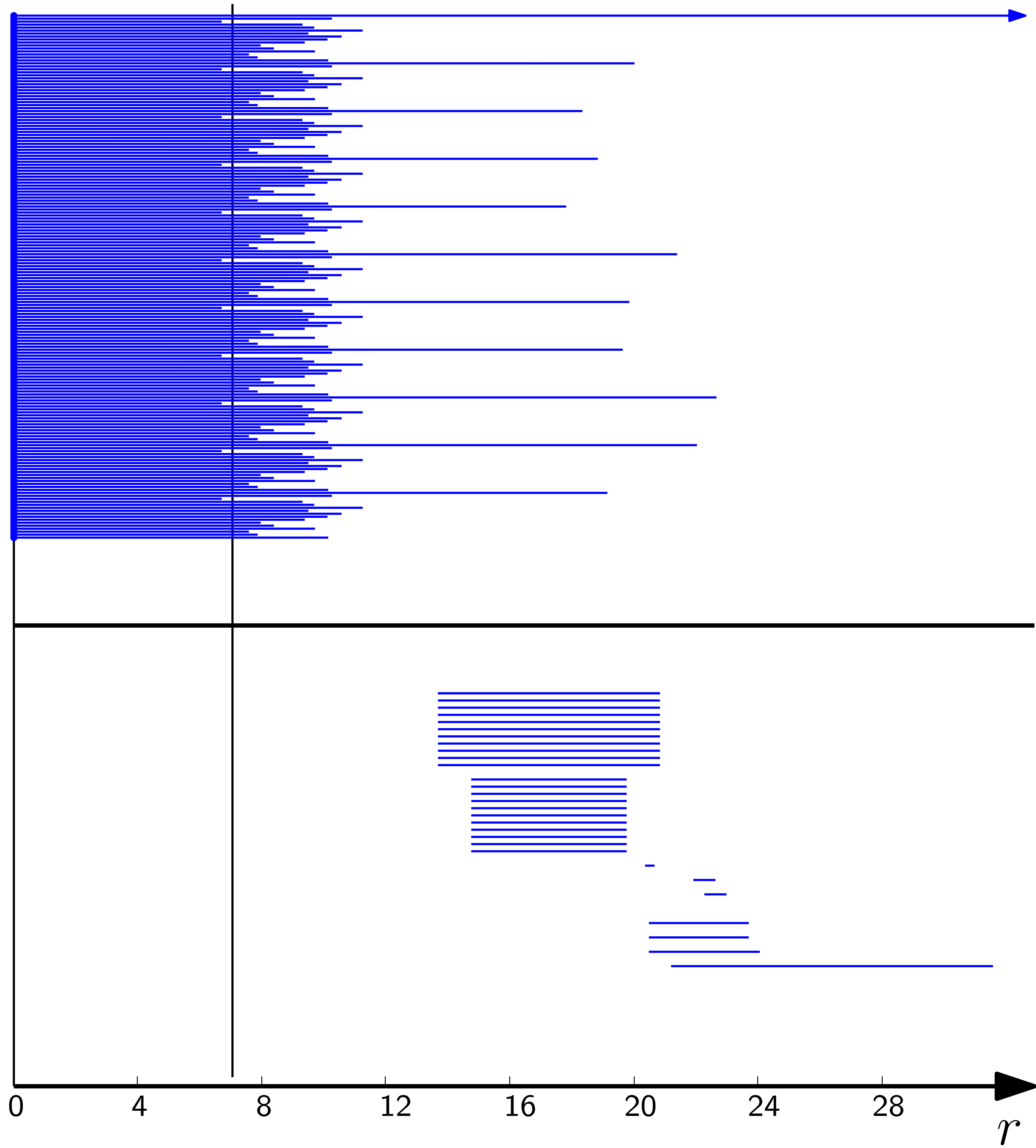
Motivation



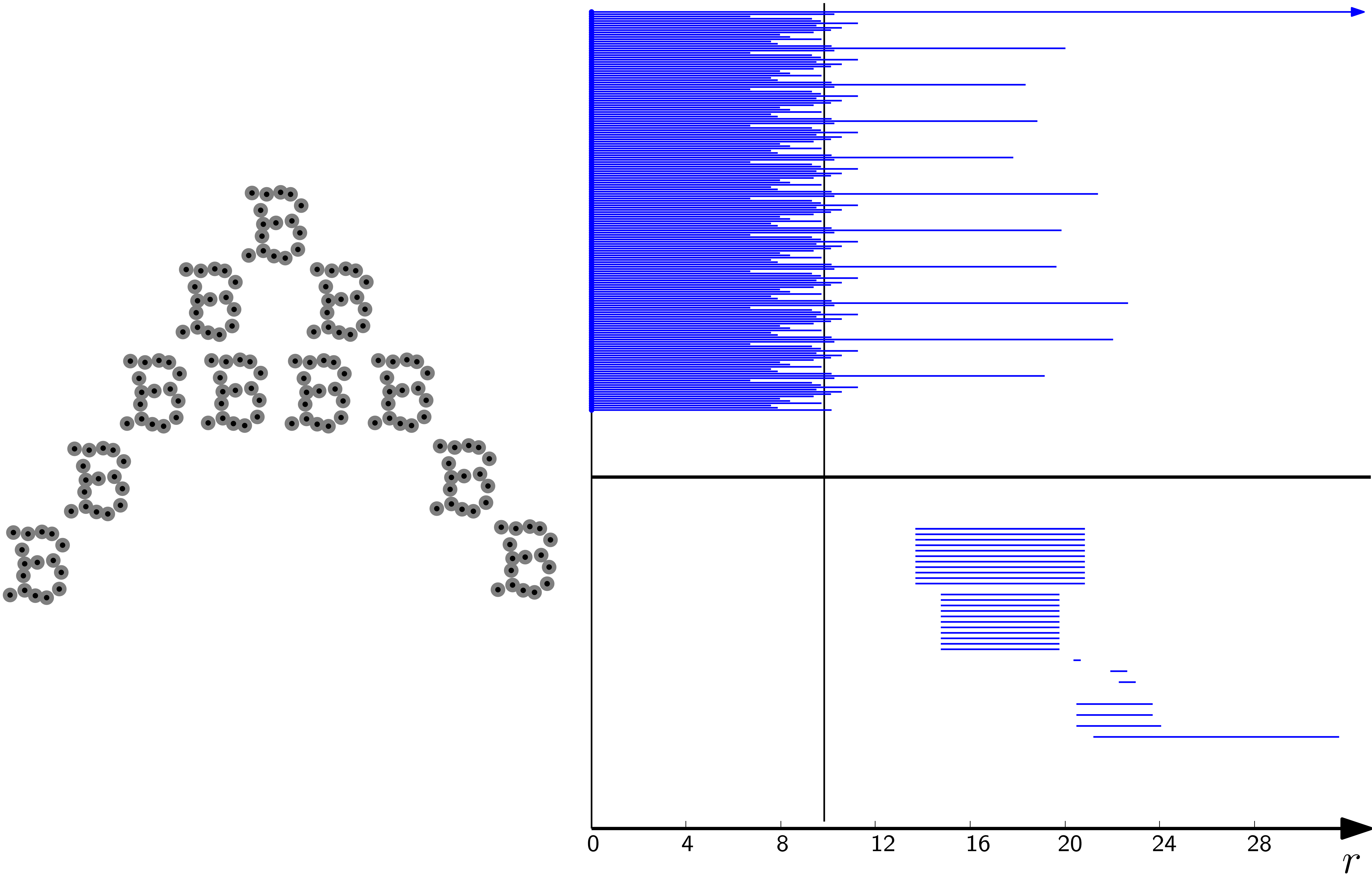


Motivation

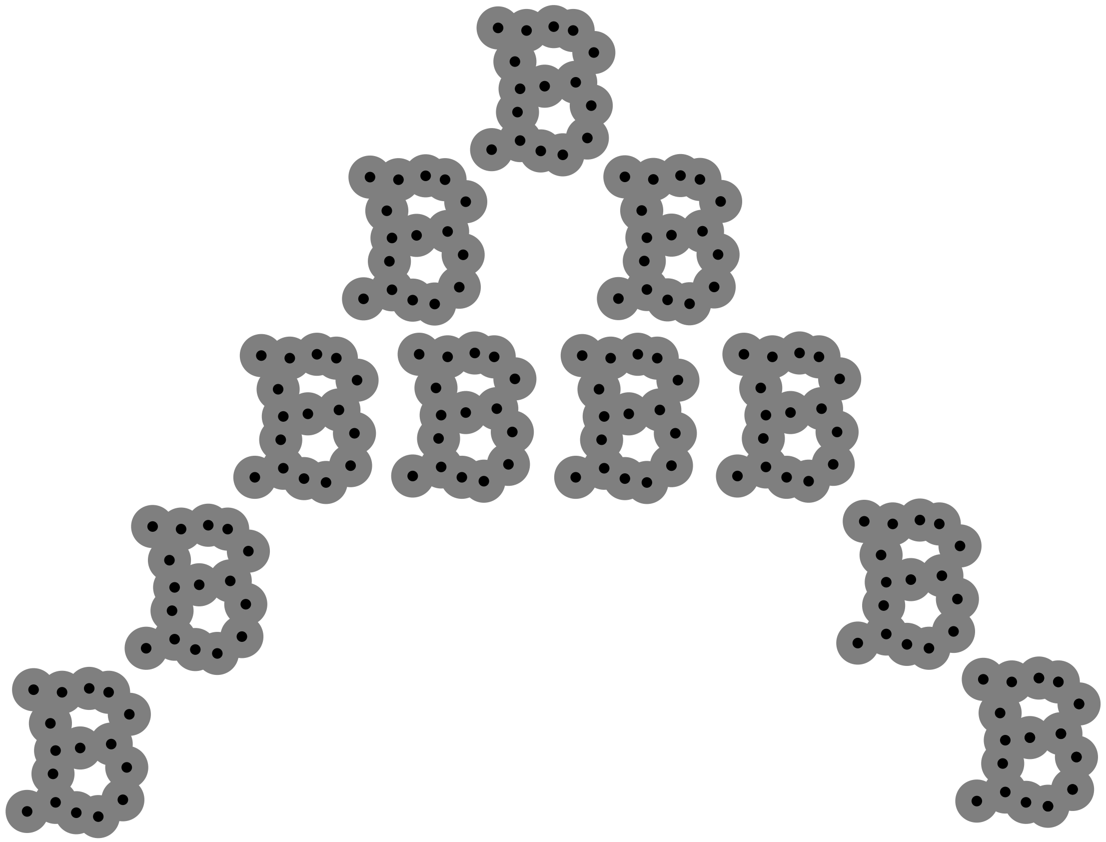


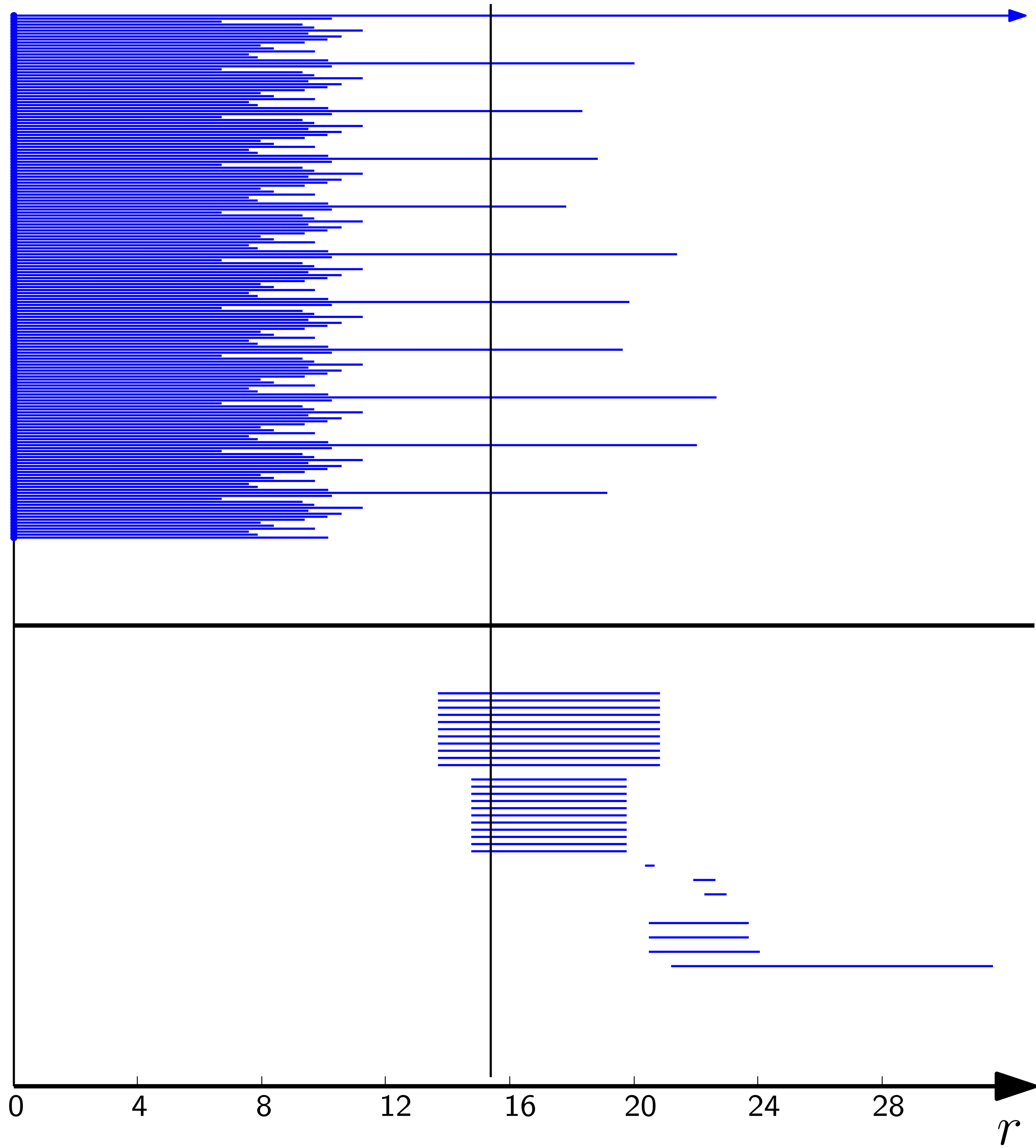


Motivation

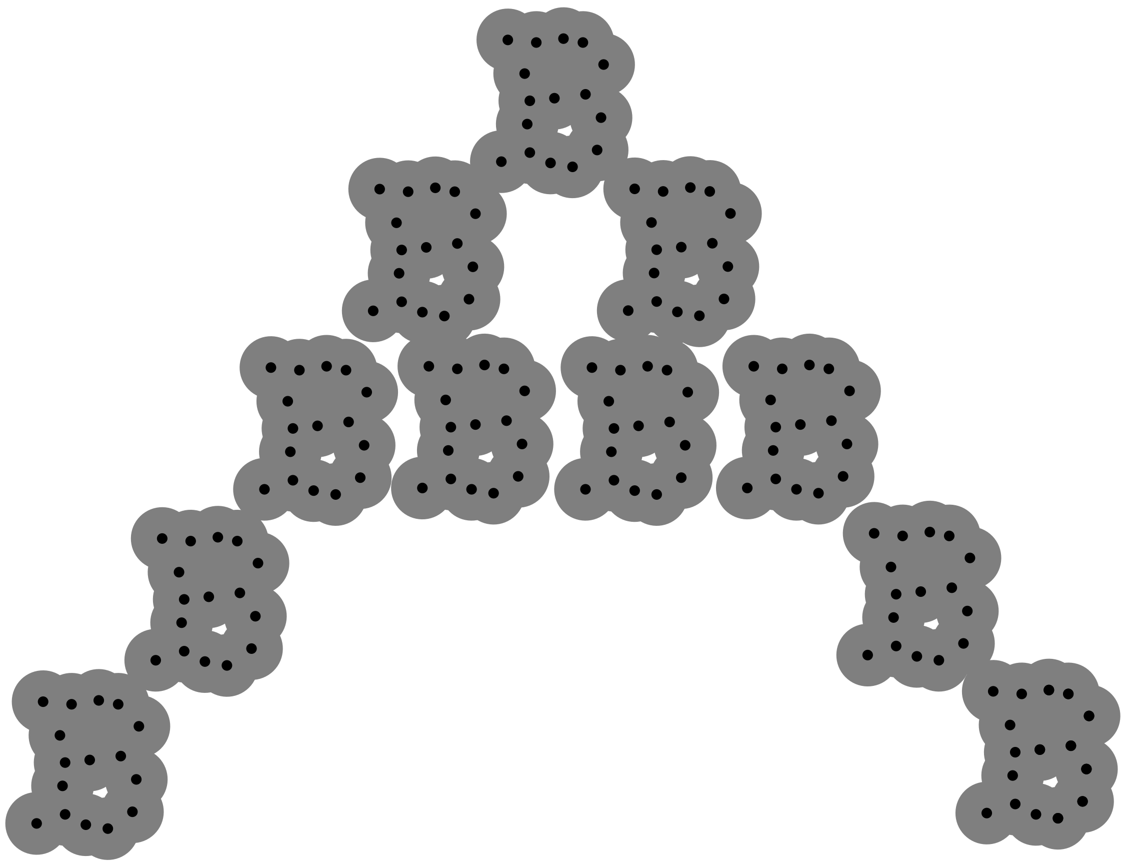


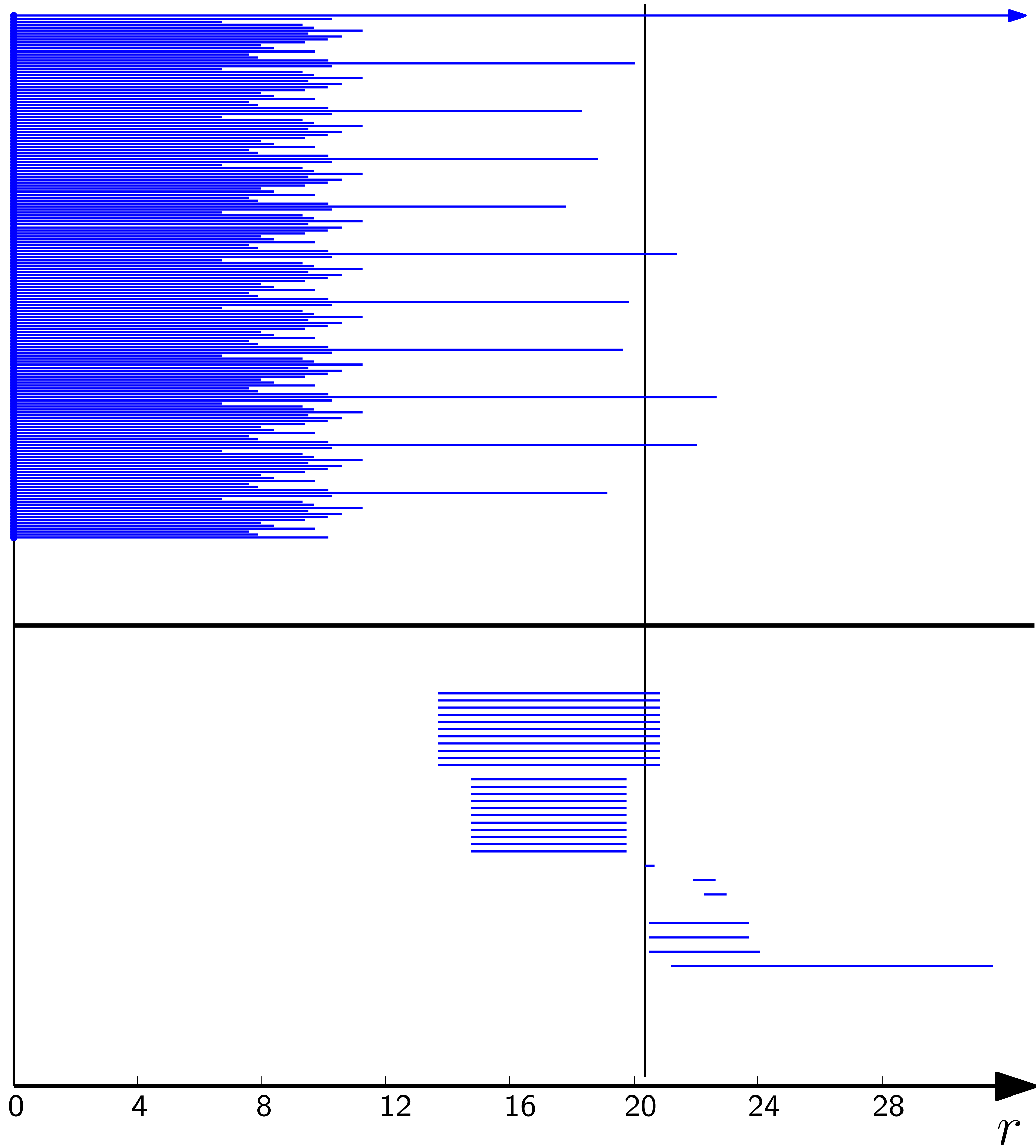
Motivation



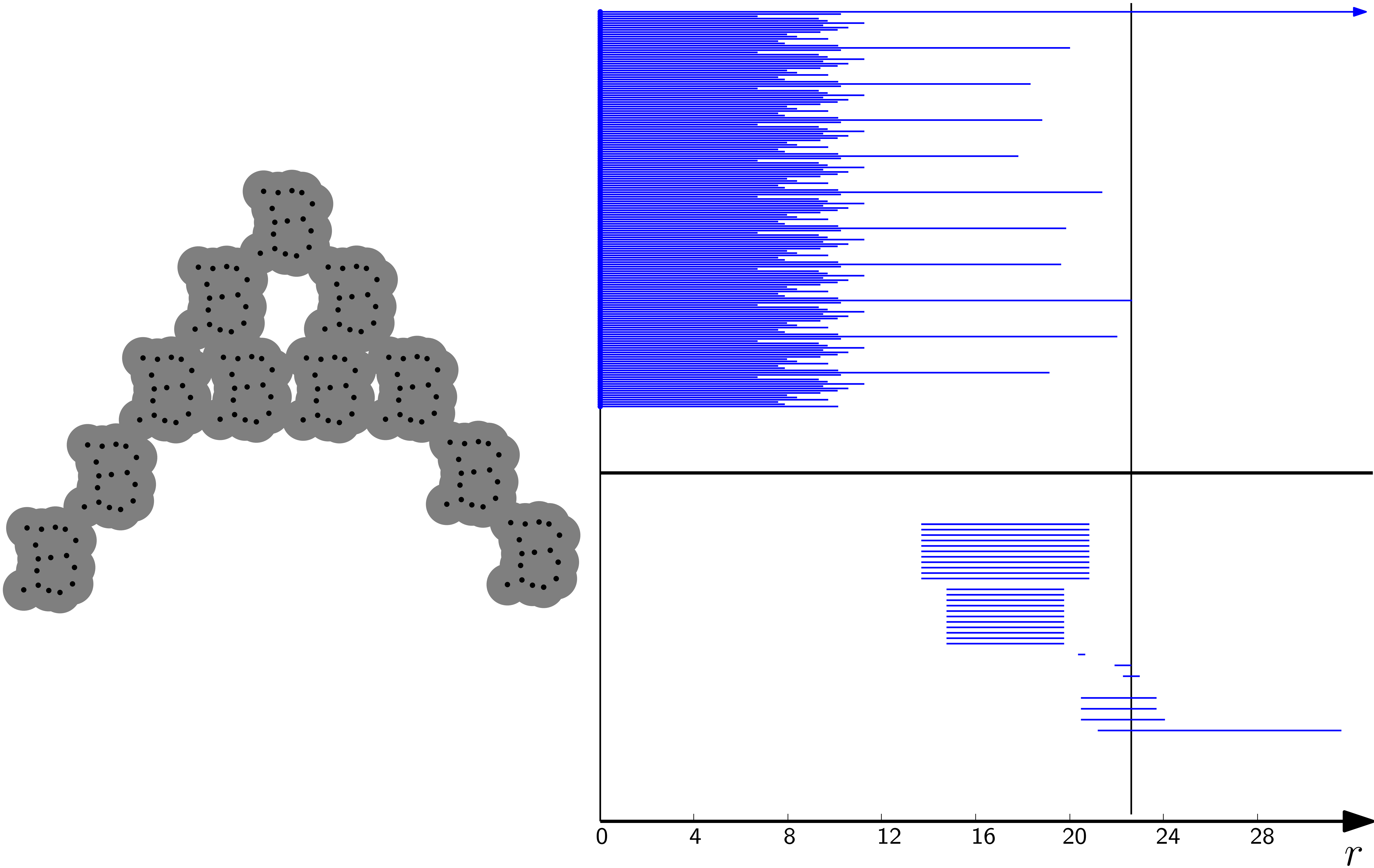


Motivation

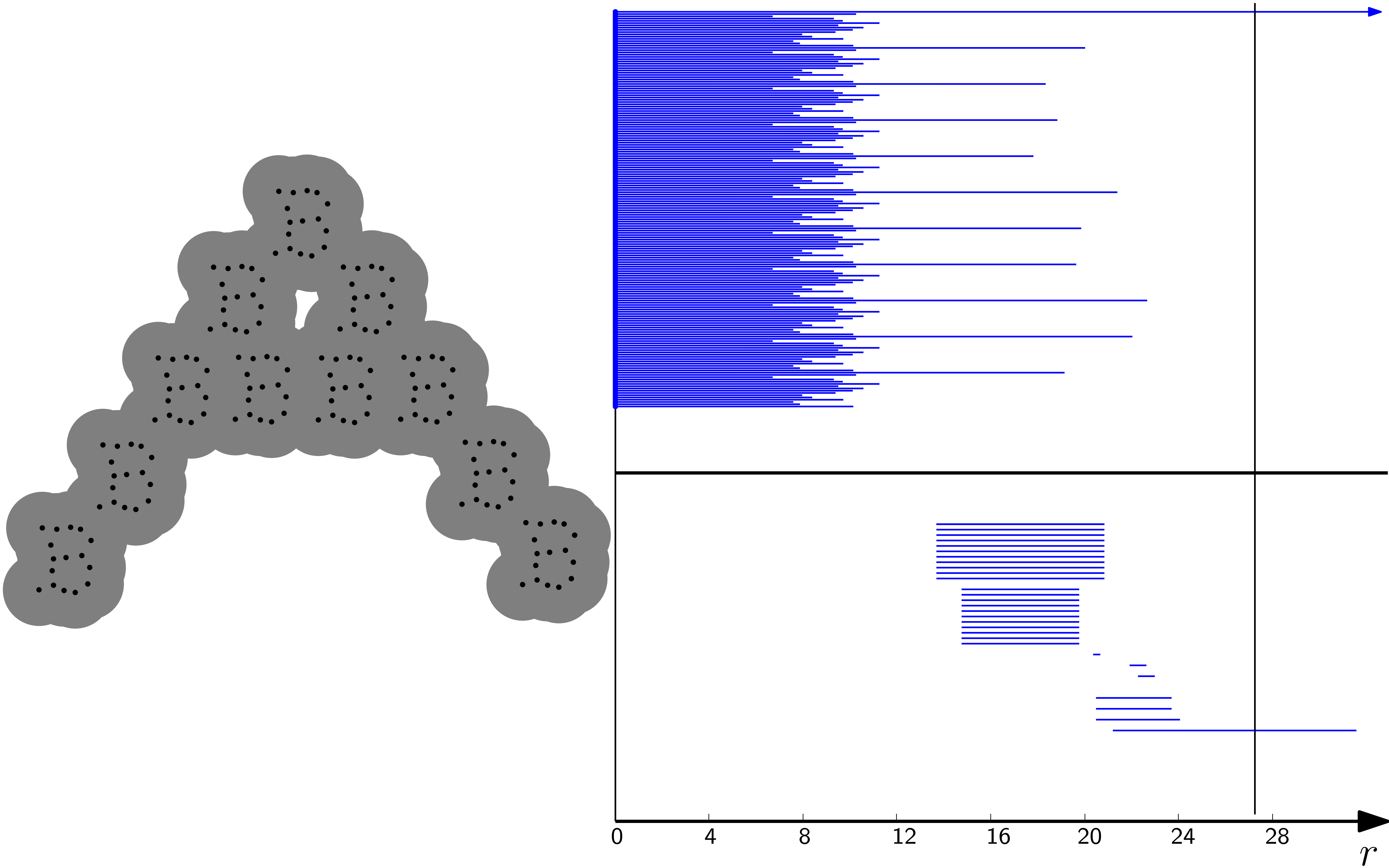




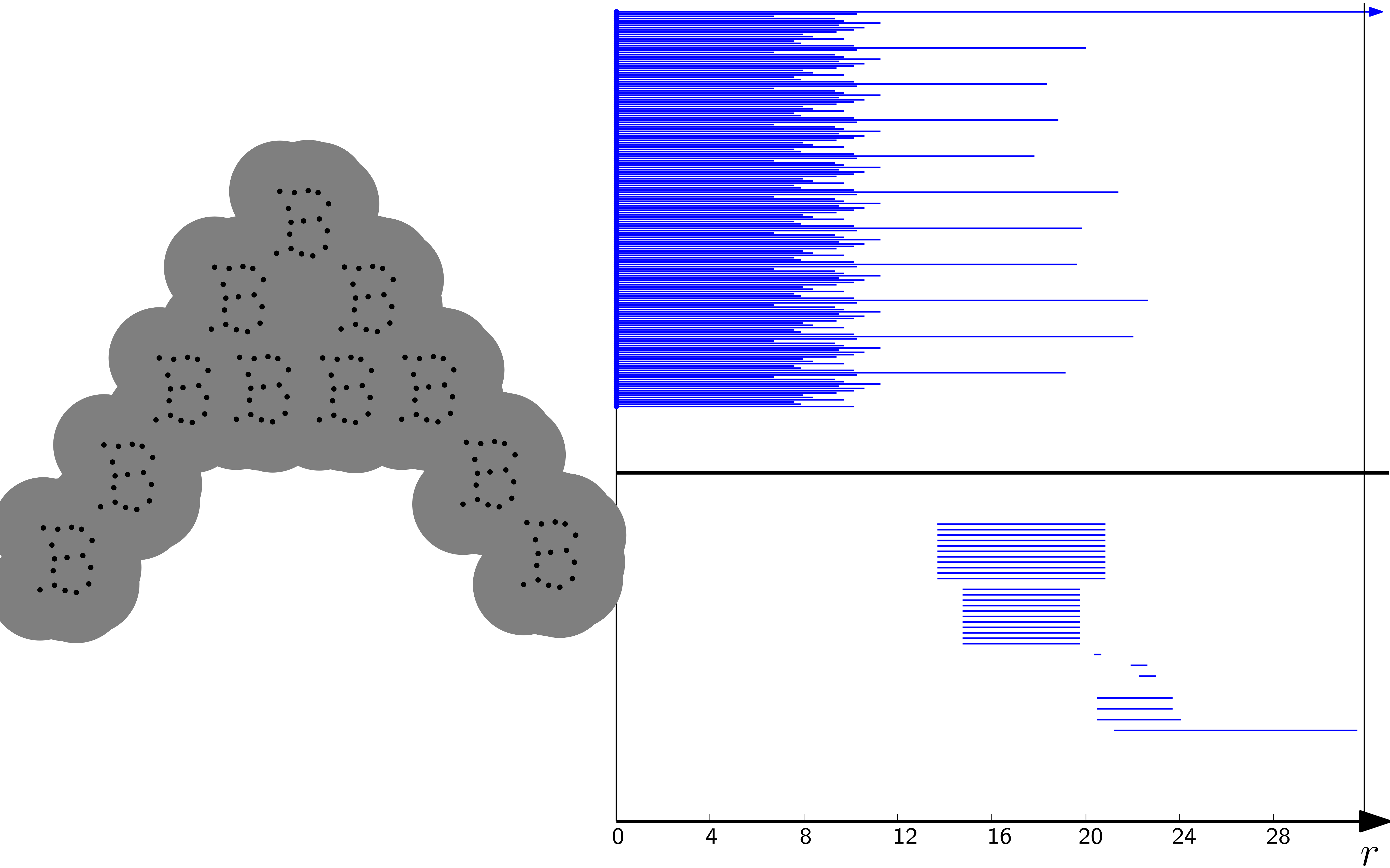
Motivation



Motivation



Motivation



Motivation

However, there are two limitations:

Motivation

However, there are two limitations:

Persistent homology computation is expensive.

Complexity of (naive) implementation is $O(N^3)$ where N is the number of simplices in the simplicial complex.

For Rips complexes, $N = 2^{\mathrm{card}(P)} - 1$ (if r is not bounded)!

Motivation

However, there are two limitations:

Persistent homology computation is expensive.

Complexity of (naive) implementation is $O(N^3)$ where N is the number of simplices in the simplicial complex.

For Rips complexes, $N = 2^{\mathrm{card}(P)} - 1$ (if r is not bounded)!

These complexes generally suffer from lack of robustness.

Motivation

However, there are two limitations:

Persistent homology computation is expensive.

Complexity of (naive) implementation is $O(N^3)$ where N is the number of simplices in the simplicial complex.

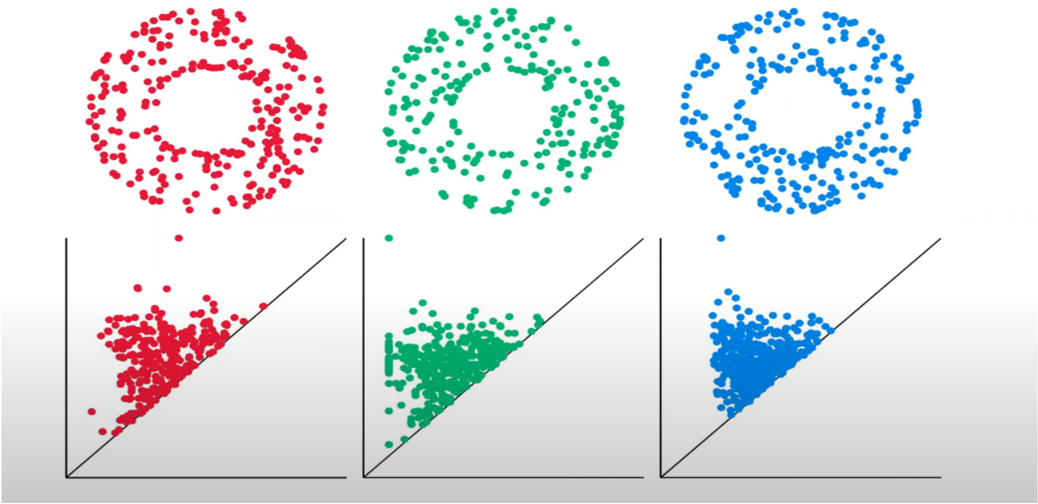
For Rips complexes, $N = 2^{\mathrm{card}(P)} - 1$ (if r is not bounded)!

These complexes generally suffer from lack of robustness.

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

$$d_B(D_{\mathrm{Rips}}(X), D_{\mathrm{Rips}}(Y)) \leq d_{GH}(X, Y).$$

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



Motivation

However, there are two limitations:

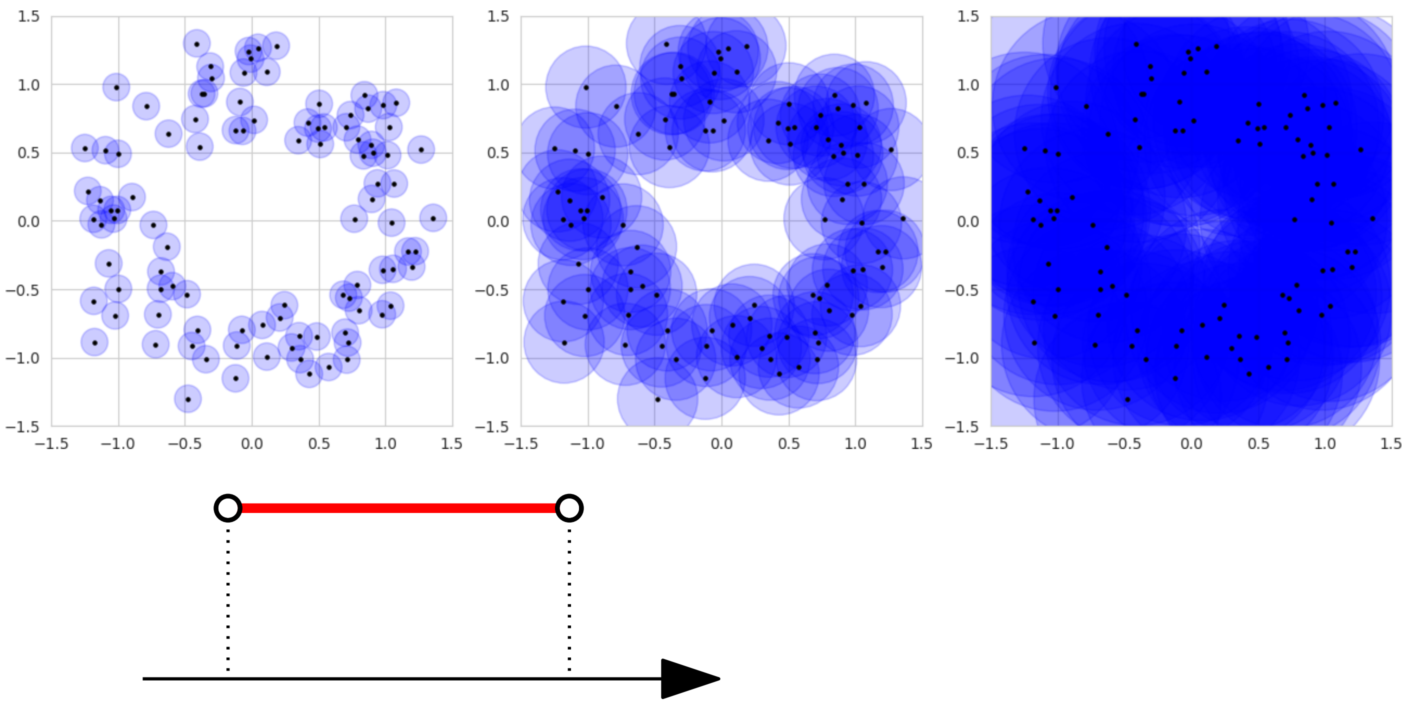
Persistent homology computation is expensive.

Complexity of (naive) implementation is $O(N^3)$ where N is the number of simplices in the simplicial complex.

For Rips complexes, $N = 2^{\mathrm{card}(P)} - 1$ (if r is not bounded)!

These complexes generally suffer from lack of robustness.

For instance, persistence diagrams of geometric complexes (and the Gromov-Hausdorff distance itself) are very sensitive to noise and outliers.



Motivation

However, there are two limitations:

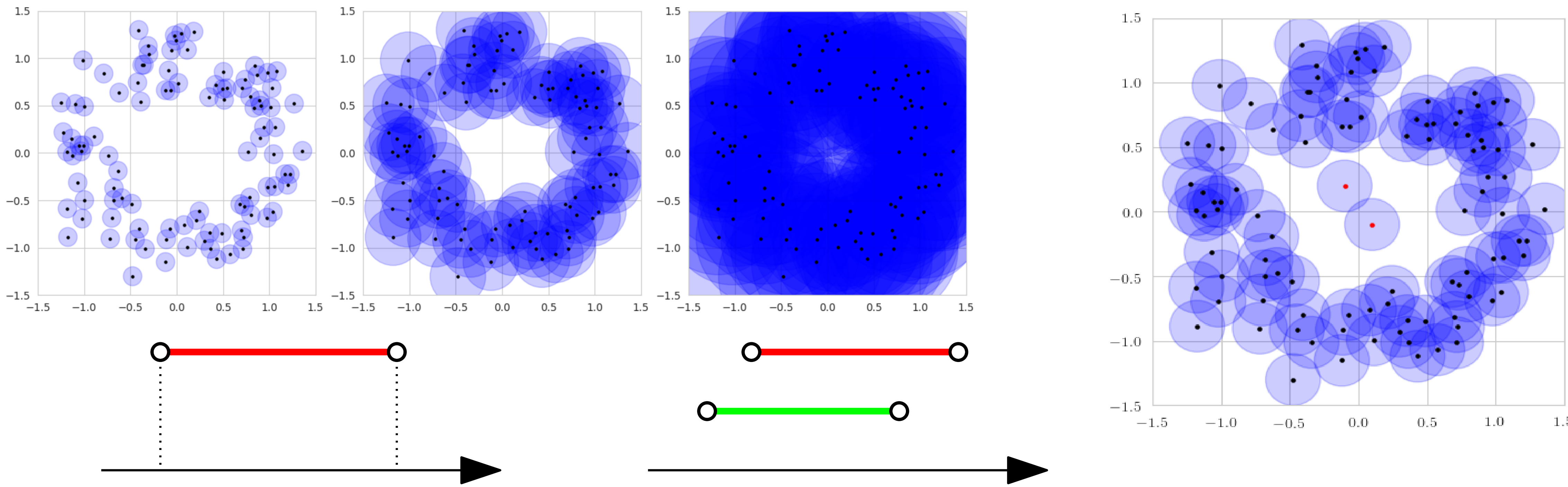
Persistent homology computation is expensive.

Complexity of (naive) implementation is $O(N^3)$ where N is the number of simplices in the simplicial complex.

For Rips complexes, $N = 2^{\mathrm{card}(P)} - 1$ (if r is not bounded)!

These complexes generally suffer from lack of robustness.

For instance, persistence diagrams of geometric complexes (and the Gromov-Hausdorff distance itself) are very sensitive to noise and outliers.



Motivation

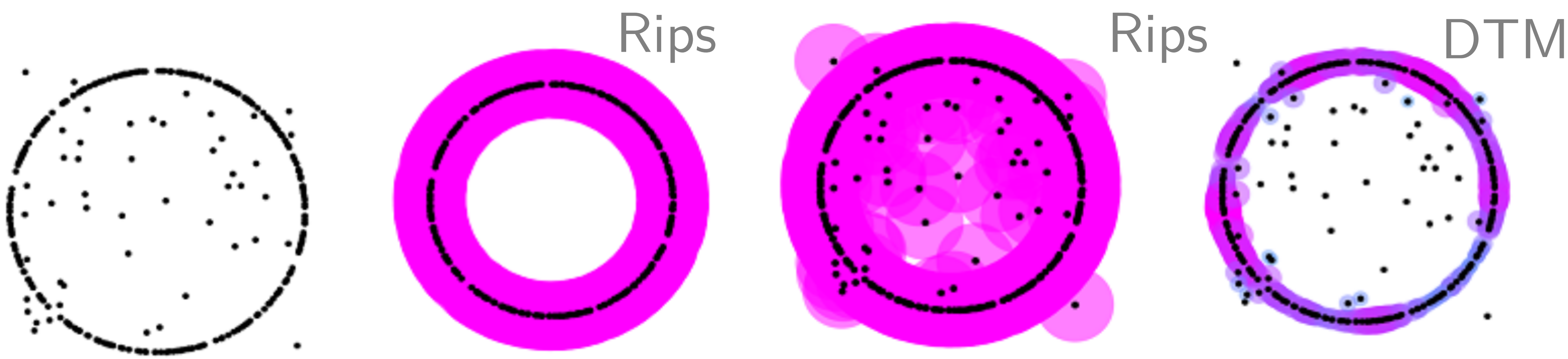
However, there are two limitations:

Persistent homology computation is expensive.

Complexity of (naive) implementation is $O(N^3)$ where N is the number of simplices in the simplicial complex.

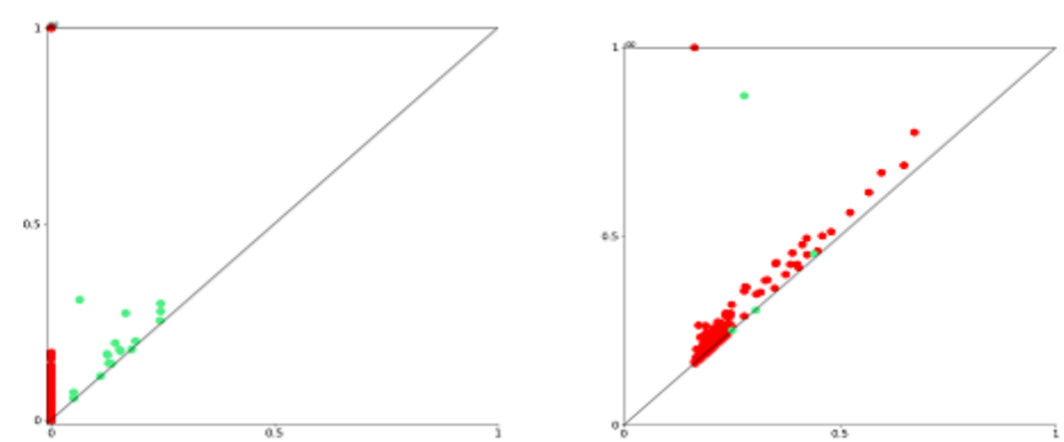
For Rips complexes, $N = 2^{\text{card}(P)} - 1$ (if r is not bounded)!

These complexes generally suffer from lack of robustness.



Some solutions have been proposed:

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



Motivation

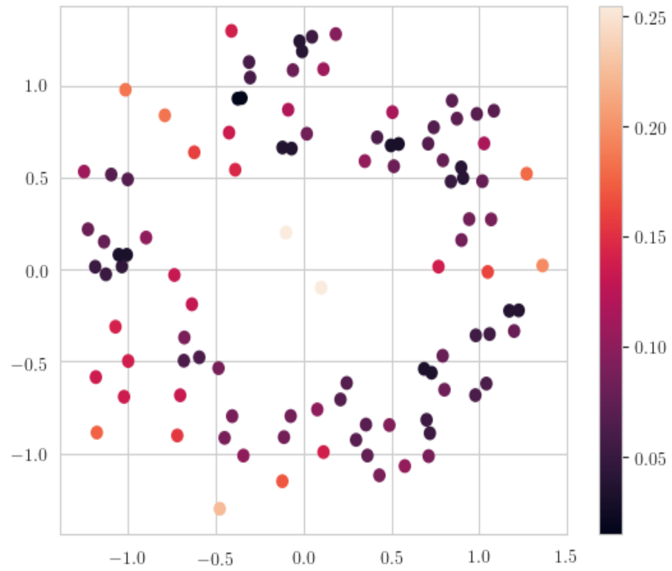
However, there are two limitations:

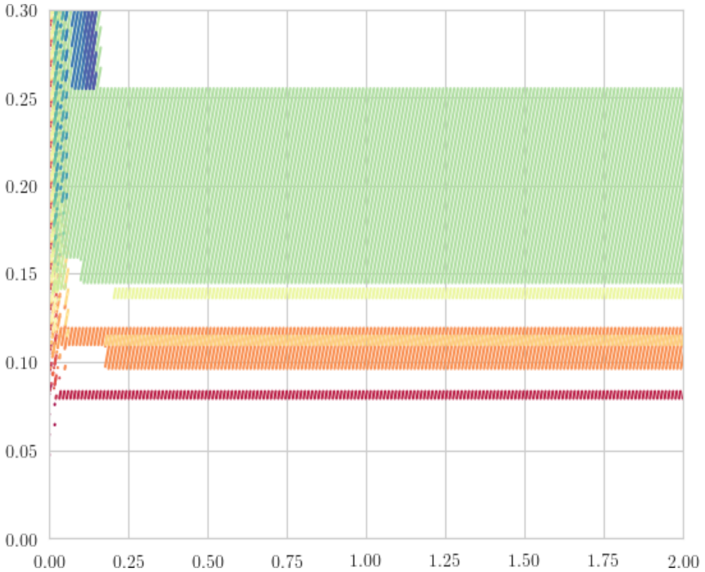
Persistent homology computation is expensive.

Complexity of (naive) implementation is $O(N^3)$ where N is the number of simplices in the simplicial complex.

For Rips complexes, $N = 2^{\mathrm{card}(P)} - 1$ (if r is not bounded)!

These complexes generally suffer from lack of robustness.





Some solutions have been proposed:

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

[*Multiparameter persistence image for topological machine learning*, C., Blumberg , NeurIPS, 2020]

[*Stability of 2-Parameter Persistent Homology*, Blumberg, Lesnick, arXiv, 2020]

Motivation

However, there are two limitations:

Persistent homology computation is expensive.

Complexity of (naive) implementation is $O(N^3)$ where N is the number of simplices in the simplicial complex.

For Rips complexes, $N = 2^{\text{card}(P)} - 1$ (if r is not bounded)!

These complexes generally suffer from lack of robustness.

We can address both problems at the same time by computing persistent homology with DeepSet neural networks.

Stability of Deep Set architectures

Thm: Let $X = \{X_1, \ldots, X_n\} \subset \mathbb{R}^d$ and $Y = \{Y_1, \ldots, Y_m\} \subset \mathbb{R}^d$ be two point clouds, and $\hat{X}_n := \frac{1}{|X|}\sum_{i=1}^{n} \delta_{X_i}$ and $\hat{Y}_m := \frac{1}{|Y|}\sum_{j=1}^{m} \delta_{Y_j}$ be the corresponding empirical measures. Let RN be a DeepSet architecture with associated functions ρ and ϕ. Then, one has:

$$\|\mathrm{RN}(X) - \mathrm{RN}(Y)\| \leq C_1 \cdot C_2 \cdot W_p(\hat{X}_n, \hat{Y}_m),$$

where C_1 and C_2 are the Lipschitz constants of ρ and ϕ respectively.

Stability of Deep Set architectures

Thm: Let $X = \{X_1, \ldots, X_n\} \subset \mathbb{R}^d$ and $Y = \{Y_1, \ldots, Y_m\} \subset \mathbb{R}^d$ be two point clouds, and $\hat{X}_n := \frac{1}{|X|} \sum_{i=1}^n \delta_{X_i}$ and $\hat{Y}_m := \frac{1}{|Y|} \sum_{j=1}^m \delta_{Y_j}$ be the corresponding empirical measures. Let RN be a DeepSet architecture with associated functions ρ and ϕ. Then, one has:

$$\|\mathrm{RN}(X) - \mathrm{RN}(Y)\| \leq C_1 \cdot C_2 \cdot W_p(\hat{X}_n, \hat{Y}_m),$$

where C_1 and C_2 are the Lipschitz constants of ρ and ϕ respectively.

In practice, we train RipsNet to minimize the following (empirical) risk:

$$\hat{\mathcal{R}}_n := \frac{1}{N} \sum_{i=1}^N \|\mathrm{RN}(X^i) - \mathrm{PV}(X^i)\|,$$

where $X^1, \ldots, X^N$ are training point clouds, and PV denote the *persistence vectorizations* (e.g., images, landscapes) of the corresponding persistence diagrams.

Stability of Deep Set architectures

Thm: Let $X = \{X_1, \ldots, X_n\} \subset \mathbb{R}^d$ and $Y = \{Y_1, \ldots, Y_m\} \subset \mathbb{R}^d$ be two point clouds, and $\hat{X}_n := \frac{1}{|X|} \sum_{i=1}^{n} \delta_{X_i}$ and $\hat{Y}_m := \frac{1}{|Y|} \sum_{j=1}^{m} \delta_{Y_j}$ be the corresponding empirical measures. Let RN be a DeepSet architecture with associated functions ρ and ϕ. Then, one has:

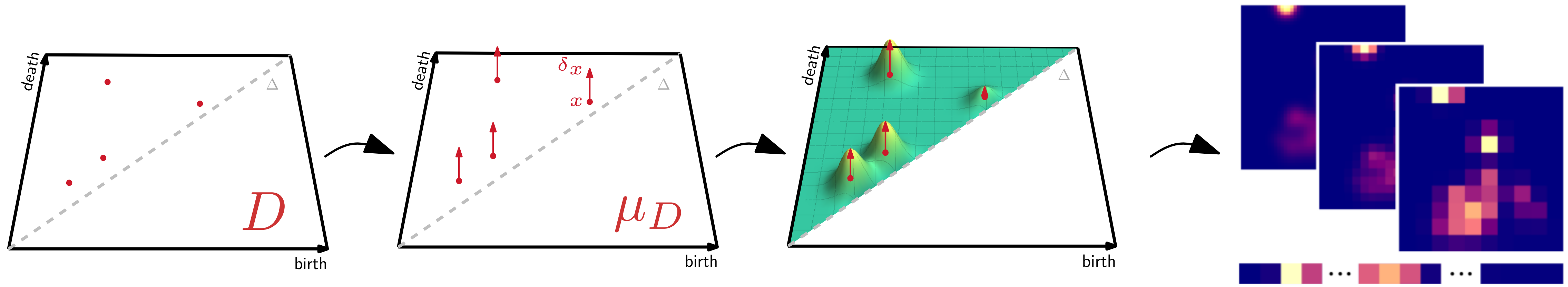
$$\|\mathrm{RN}(X) - \mathrm{RN}(Y)\| \leq C_1 \cdot C_2 \cdot W_p(\hat{X}_n, \hat{Y}_m),$$

where C_1 and C_2 are the Lipschitz constants of ρ and ϕ respectively.

In practice, we train RipsNet to minimize the following (empirical) risk:

$$\hat{\mathcal{R}}_n := \frac{1}{N} \sum_{i=1}^{N} \|\mathrm{RN}(X^i) - \mathrm{PV}(X^i)\|,$$

where $X^1, \ldots, X^N$ are training point clouds, and PV denote the *persistence vectorizations* (e.g., images, landscapes) of the corresponding persistence diagrams.



Stability of Deep Set architectures

Thm: Let $X = \{X_1, \ldots, X_n\} \subset \mathbb{R}^d$ and $Y = \{Y_1, \ldots, Y_m\} \subset \mathbb{R}^d$ be two point clouds, and $\hat{X}_n := \frac{1}{|X|}\sum_{i=1}^{n} \delta_{X_i}$ and $\hat{Y}_m := \frac{1}{|Y|}\sum_{j=1}^{m} \delta_{Y_j}$ be the corresponding empirical measures. Let RN be a DeepSet architecture with associated functions ρ and ϕ. Then, one has:

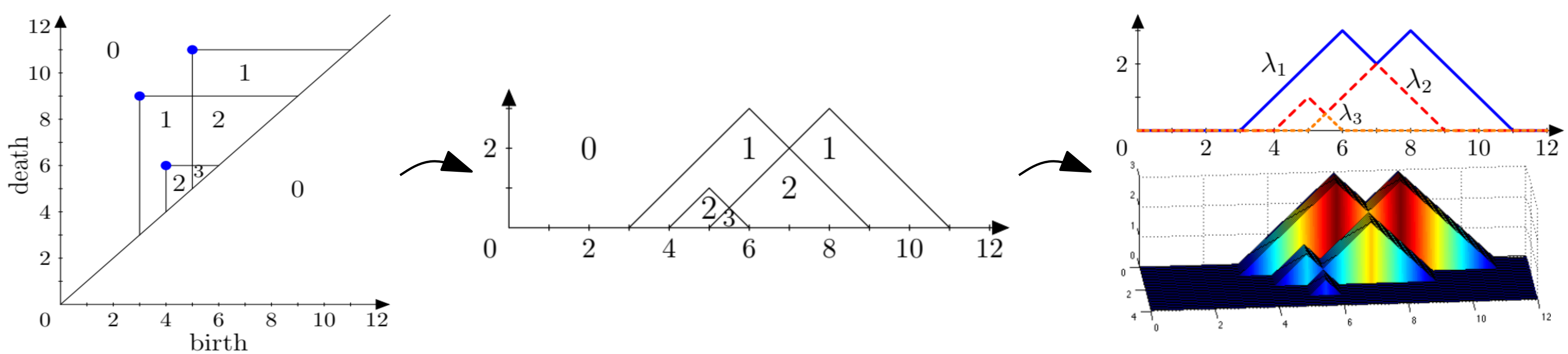
$$\|\mathrm{RN}(X) - \mathrm{RN}(Y)\| \leq C_1 \cdot C_2 \cdot W_p(\hat{X}_n, \hat{Y}_m),$$

where C_1 and C_2 are the Lipschitz constants of ρ and ϕ respectively.

In practice, we train RipsNet to minimize the following (empirical) risk:

$$\hat{\mathcal{R}}_n := \frac{1}{N}\sum_{i=1}^{N} \|\mathrm{RN}(X^i) - \mathrm{PV}(X^i)\|,$$

where $X^1, \ldots, X^N$ are training point clouds, and PV denote the *persistence vectorizations* (e.g., images, landscapes) of the corresponding persistence diagrams.



Stability of Deep Set architectures

Let P be a law on some compact set $\Omega \subset \mathbb{R}^d$, fix $n \in \mathbb{N}$, and let $\mathbb{P}$ denote $P^{\otimes n}$, that is, $X \sim \mathbb{P}$ is a random point cloud $X = \{X_1, \ldots, X_n\}$ where the X_i's are i.i.d. $\sim P$. Finally let $\mathcal{R}$ be the theoretical risk:

$$\mathcal{R} := \int \|\mathrm{RN}(X) - \mathrm{PV}(X)\| \mathrm{d}\mathbb{P}(X).$$

Stability of Deep Set architectures

Let P be a law on some compact set $\Omega \subset \mathbb{R}^d$, fix $n \in \mathbb{N}$, and let $\mathbb{P}$ denote $P^{\otimes n}$, that is, $X \sim \mathbb{P}$ is a random point cloud $X = \{X_1, \ldots, X_n\}$ where the X_i's are i.i.d. $\sim P$. Finally let $\mathcal{R}$ be the theoretical risk:

$$\mathcal{R} := \int \|\mathrm{RN}(X) - \mathrm{PV}(X)\| \mathrm{d}\mathbb{P}(X).$$

Now, we randomly replace a fraction $\lambda = \frac{n-k}{n} \in (0,1)$ of the points of X by corrupted observations distributed with respect to some law Q. Let $Y \sim Q^{\otimes n-k} =: \mathbb{Q}$ and $F(X,Y)$ denote this corrupted point cloud.

Stability of Deep Set architectures

Let P be a law on some compact set $\Omega \subset \mathbb{R}^d$, fix $n \in \mathbb{N}$, and let $\mathbb{P}$ denote $P^{\otimes n}$, that is, $X \sim \mathbb{P}$ is a random point cloud $X = \{X_1, \ldots, X_n\}$ where the X_i's are i.i.d. $\sim P$. Finally let $\mathcal{R}$ be the theoretical risk:

$$\mathcal{R} := \int \|\mathrm{RN}(X) - \mathrm{PV}(X)\| \mathrm{d}\mathbb{P}(X).$$

Now, we randomly replace a fraction $\lambda = \frac{n-k}{n} \in (0,1)$ of the points of X by corrupted observations distributed with respect to some law Q. Let $Y \sim Q^{\otimes n-k} =: \mathbb{Q}$ and $F(X,Y)$ denote this corrupted point cloud.

Thm: Let L be an upper bound on the diameters of the supports of P and Q. Then, one has:

$$\int \|\mathrm{RN}(F(X,Y)) - \mathrm{PV}(X)\| \mathrm{d}\mathbb{P}(X) \mathrm{d}\mathbb{Q}(Y) \leq \lambda \cdot C_1 \cdot C_2 \cdot L + \mathcal{R},$$

where C_1 and C_2 are the Lipschitz constants of ρ and ϕ respectively.

Stability of Deep Set architectures

Let P be a law on some compact set $\Omega \subset \mathbb{R}^d$, fix $n \in \mathbb{N}$, and let $\mathbb{P}$ denote $P^{\otimes n}$, that is, $X \sim \mathbb{P}$ is a random point cloud $X = \{X_1, \ldots, X_n\}$ where the X_i's are i.i.d. $\sim P$. Finally let $\mathcal{R}$ be the theoretical risk:

$$\mathcal{R} := \int \|\mathrm{RN}(X) - \mathrm{PV}(X)\| \mathrm{d}\mathbb{P}(X).$$

Now, we randomly replace a fraction $\lambda = \frac{n-k}{n} \in (0,1)$ of the points of X by corrupted observations distributed with respect to some law Q. Let $Y \sim Q^{\otimes n-k} =: \mathbb{Q}$ and $F(X,Y)$ denote this corrupted point cloud.

Thm: Let L be an upper bound on the diameters of the supports of P and Q. Then, one has:

$$\int \|\mathrm{RN}(F(X,Y)) - \mathrm{PV}(X)\| \mathrm{d}\mathbb{P}(X)\mathrm{d}\mathbb{Q}(Y) \leq \lambda \cdot C_1 \cdot C_2 \cdot L + \mathcal{R},$$

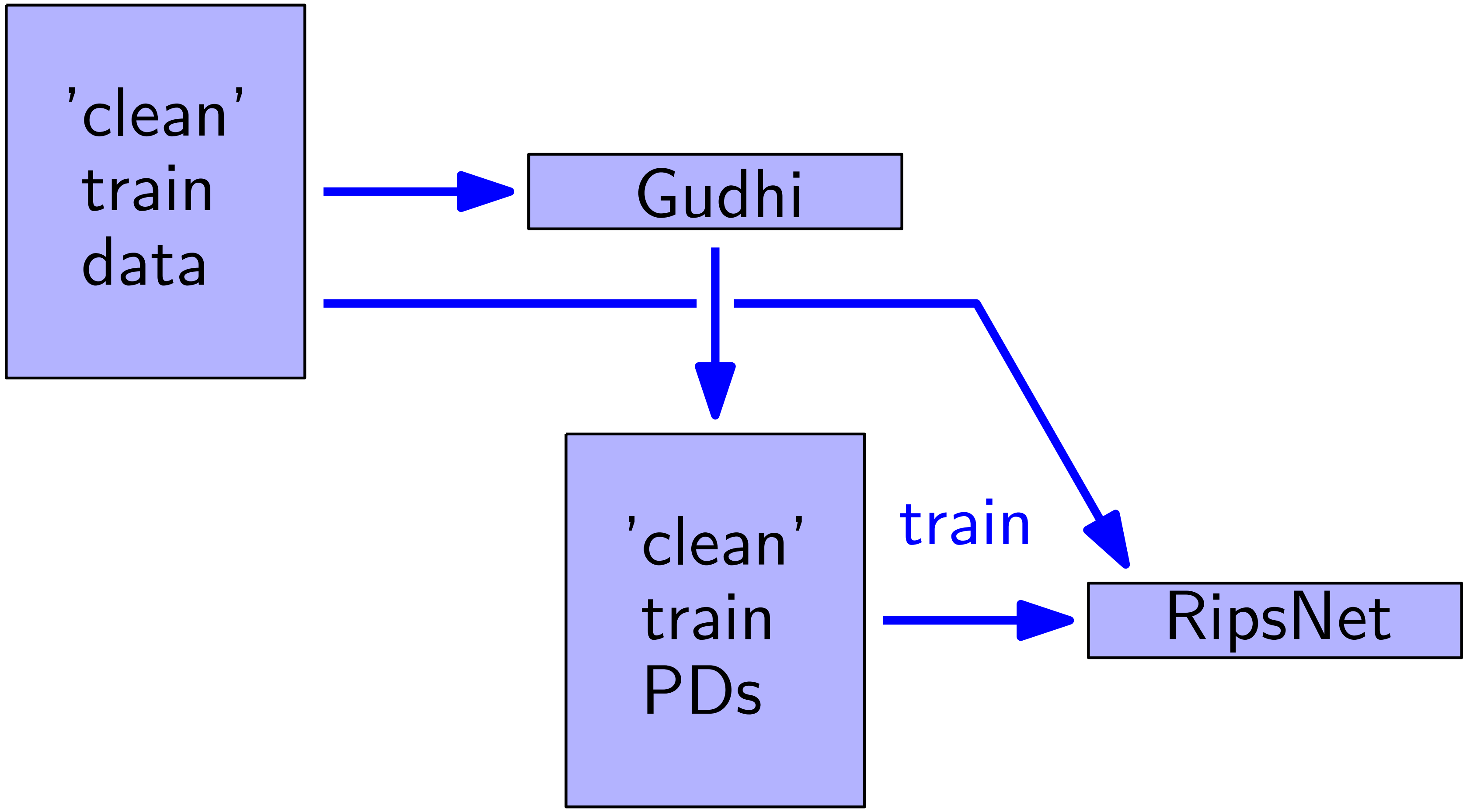
where C_1 and C_2 are the Lipschitz constants of ρ and ϕ respectively.

Note that it is easy to show that this robustness is not satisfied by persistence vectorizations:

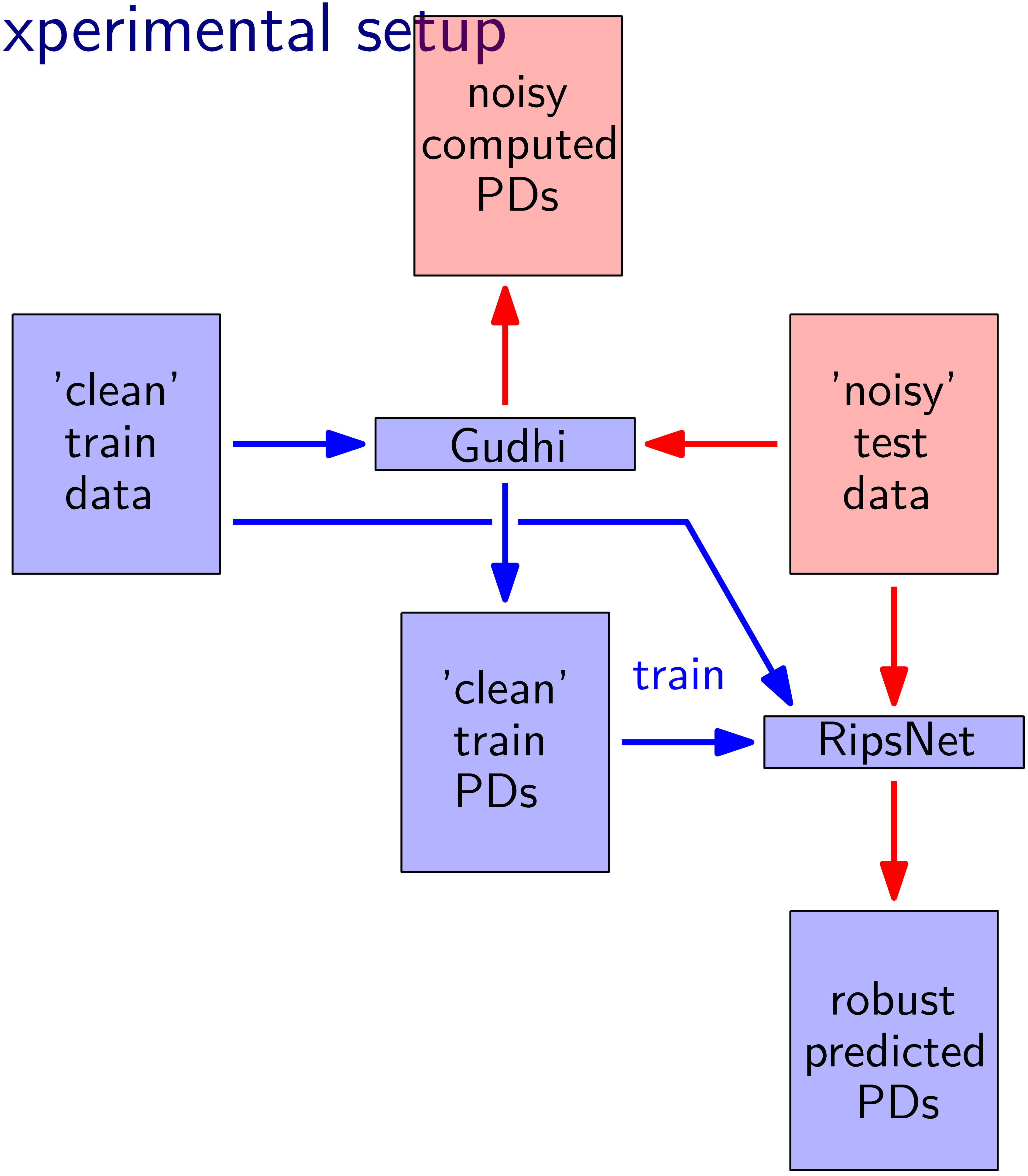
$$\int \|\mathrm{PV}(F(X,Y)) - \mathrm{PV}(X)\| \mathrm{d}\mathbb{P}(X)\mathrm{d}\mathbb{Q}(Y) \not\to 0 \text{ when } \lambda \to 0.$$

Experimental setup

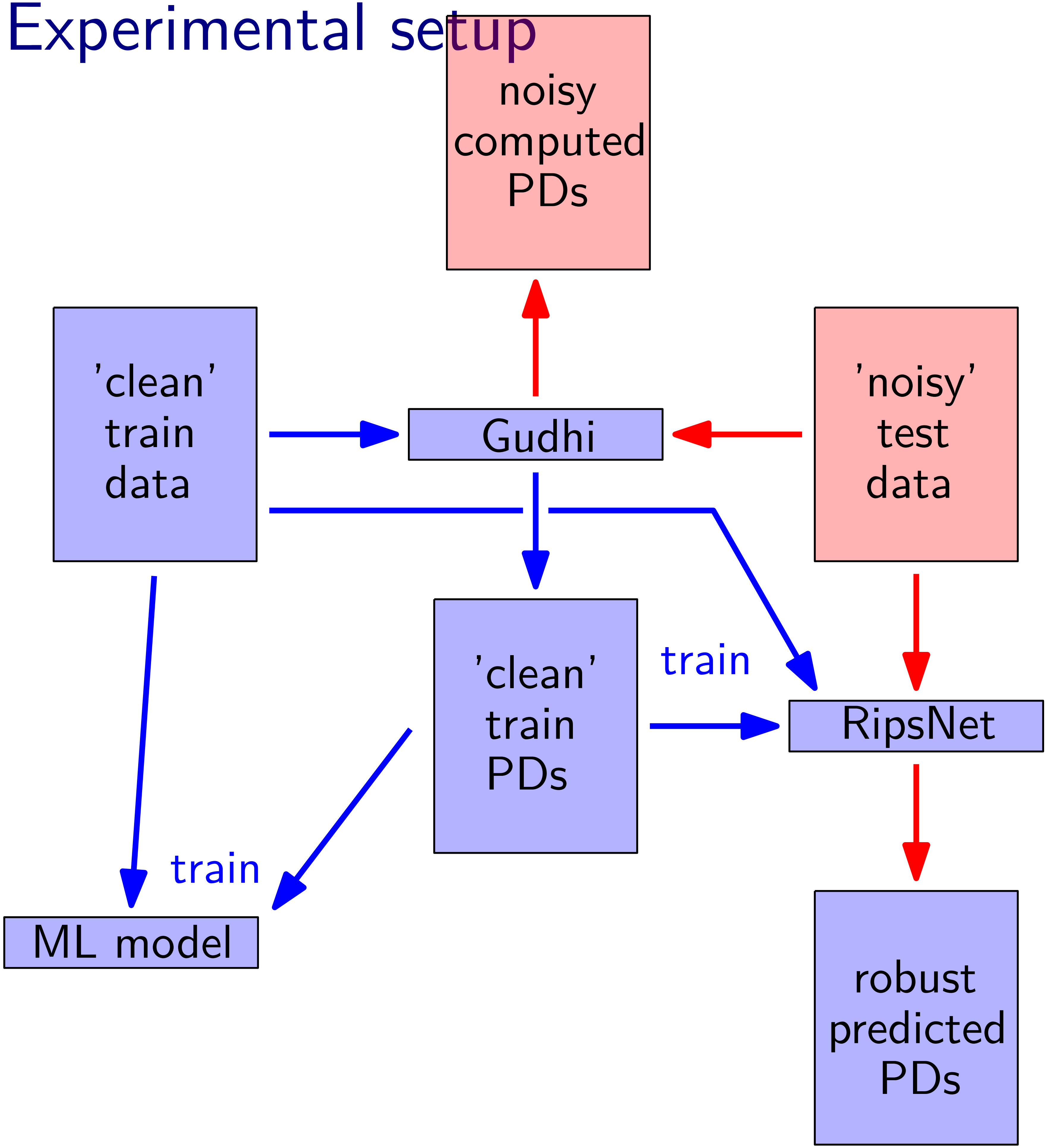
Experimental setup



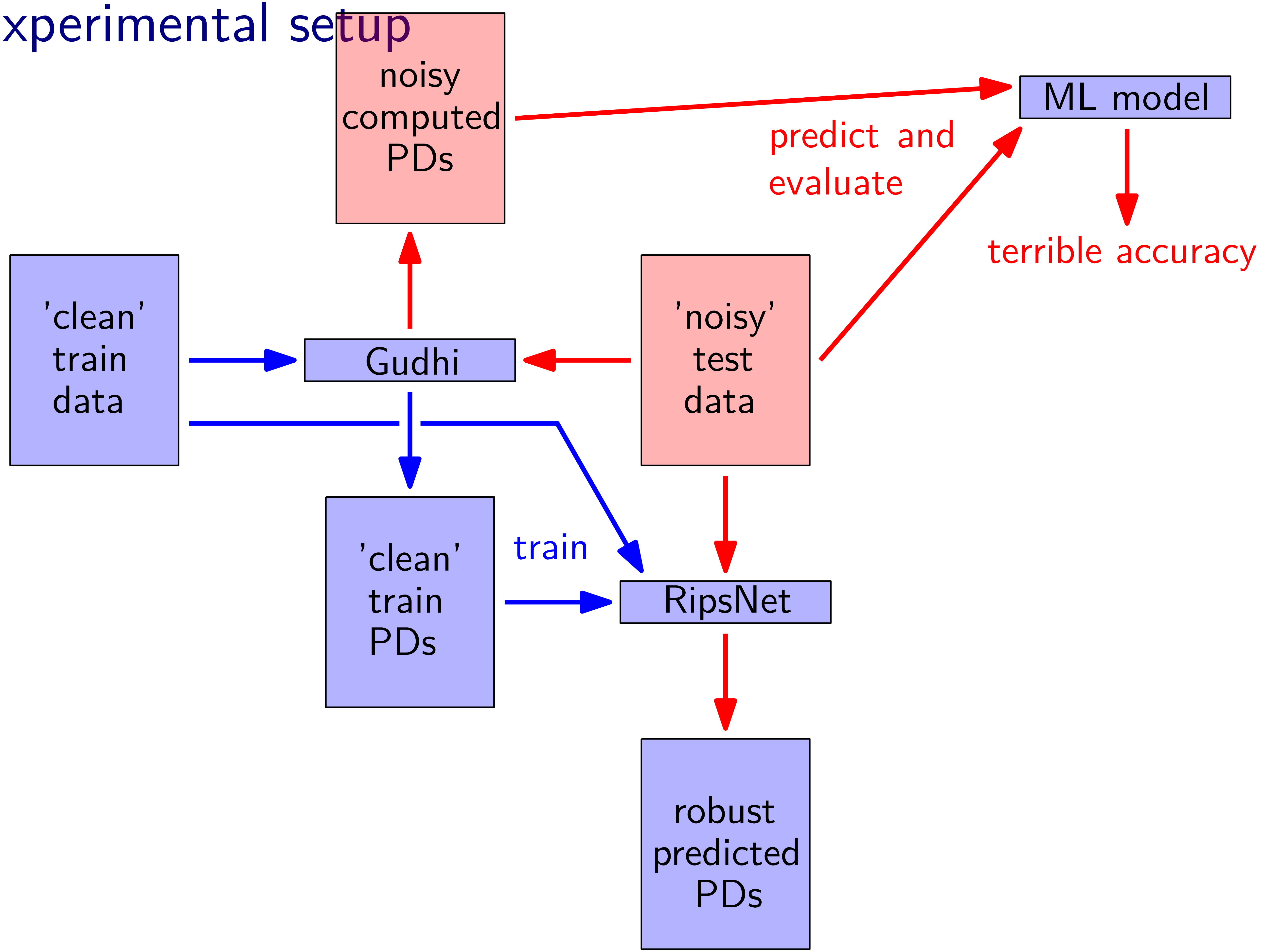
Experimental setup



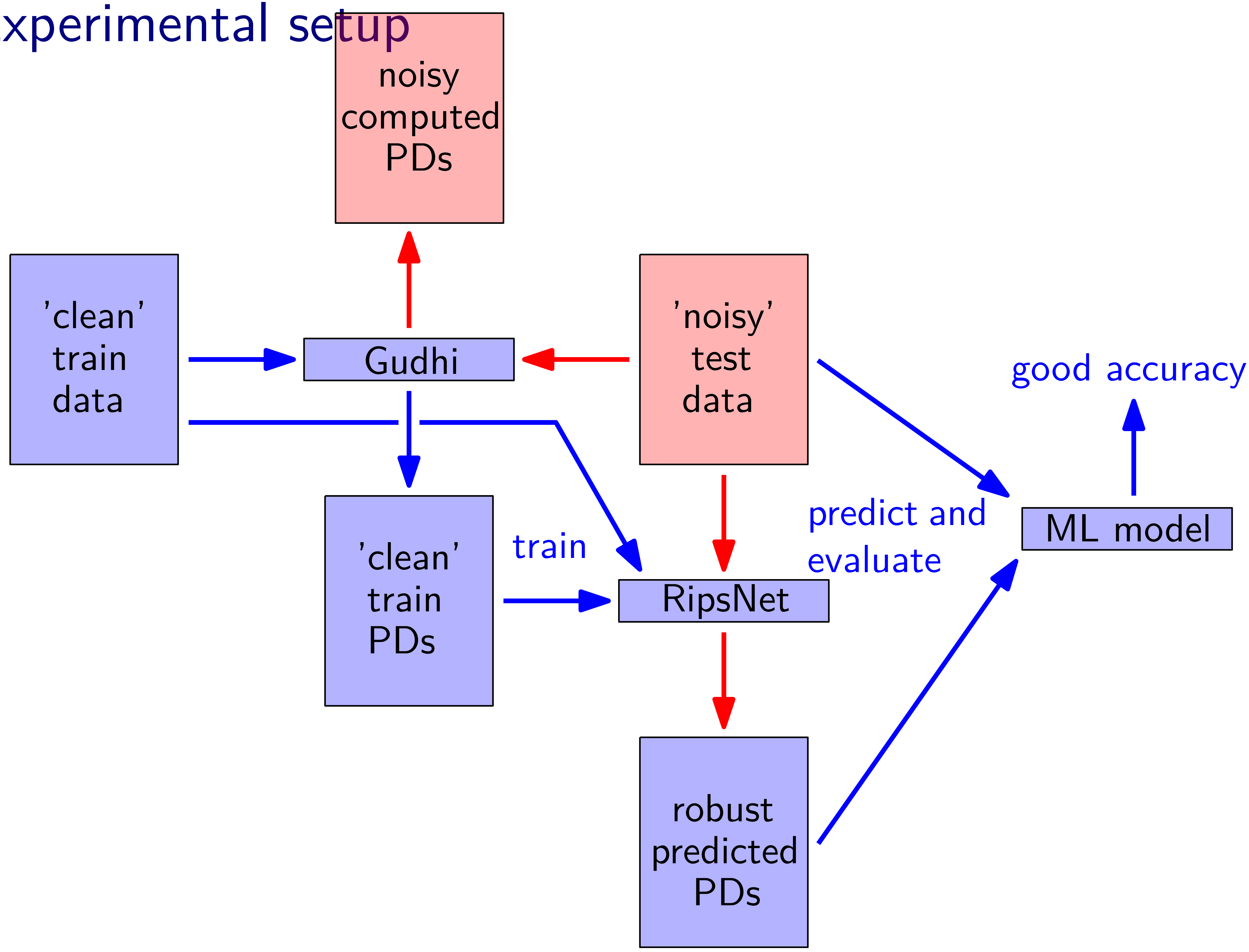
Experimental setup



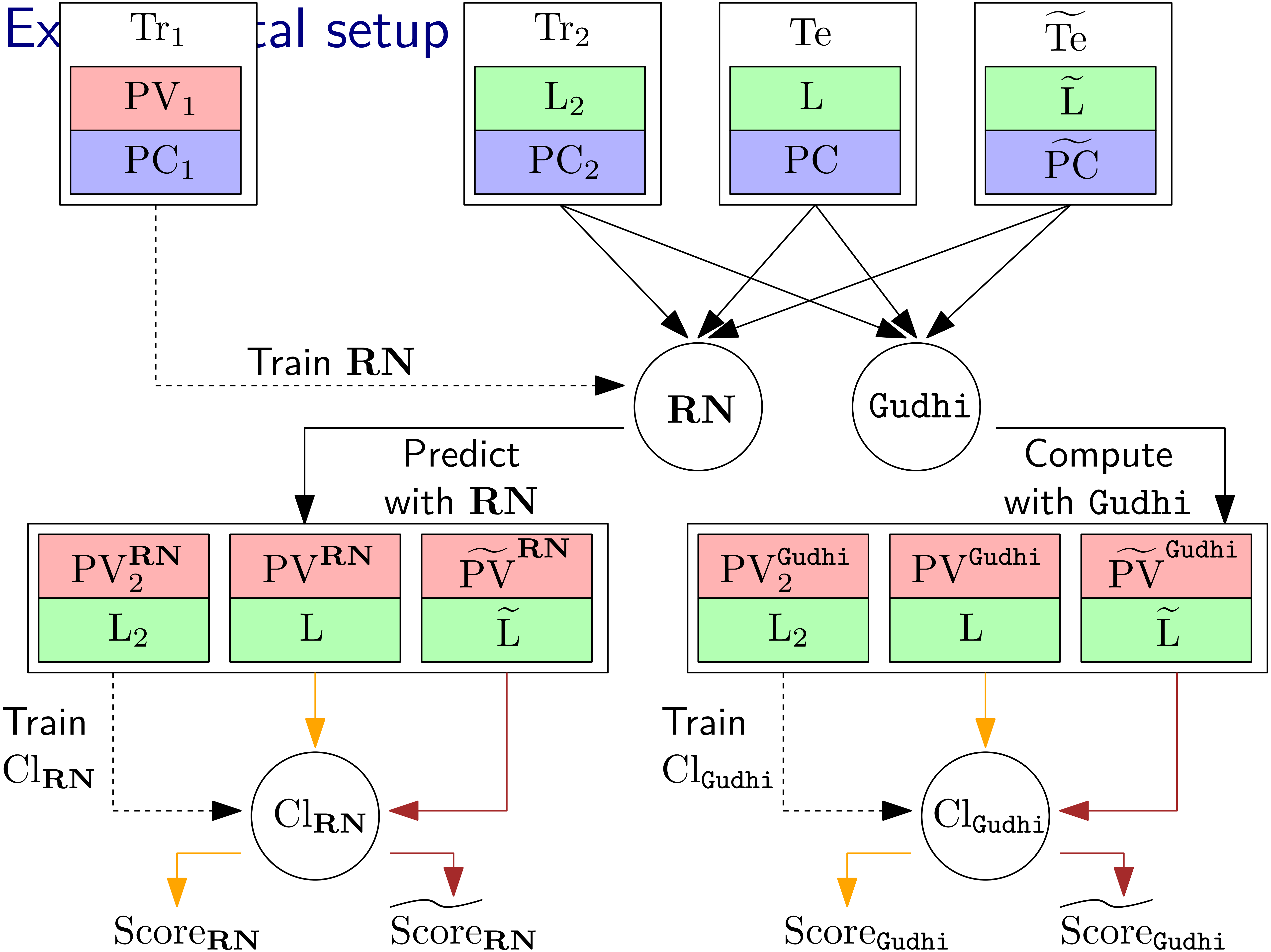
Experimental setup



Experimental setup

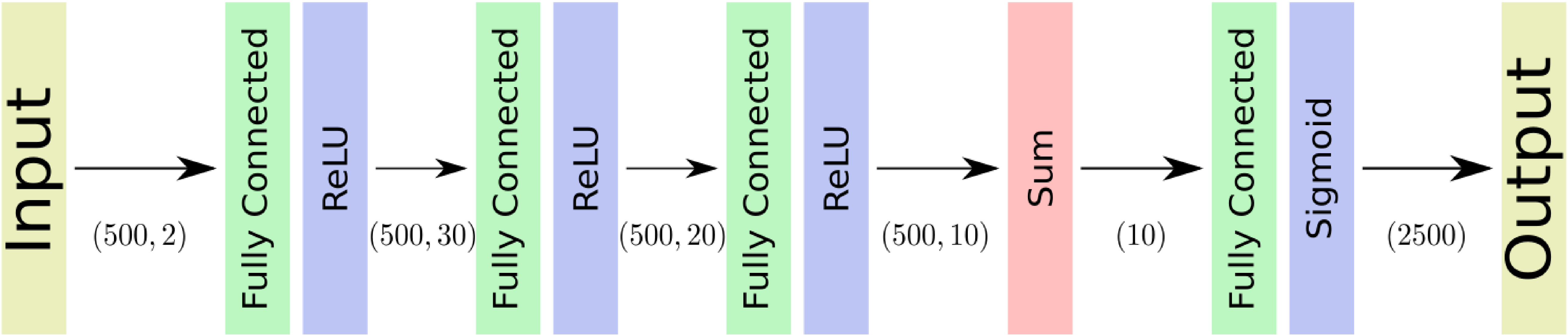


Ex tal setup



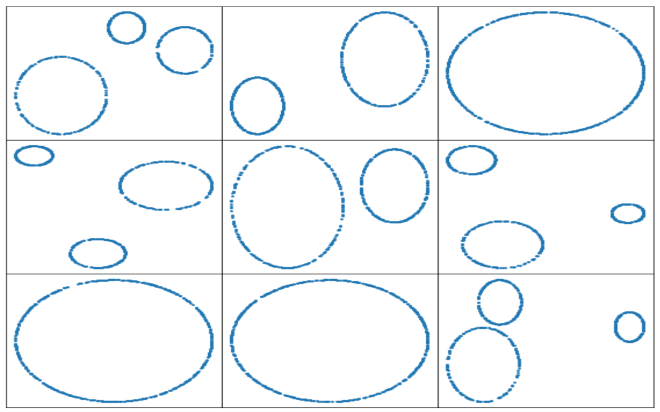
Tr_1
PV_1
PC_1
Tr_2
L_2
PC_2
Te
L
PC
$\widetilde{\mathrm{Te}}$
$\widetilde{\mathrm{L}}$
$\widetilde{\mathrm{PC}}$
Train $\mathbf{RN}$
$\mathbf{RN}$
Gudhi
Predict with $\mathbf{RN}$
Compute with Gudhi
$\mathrm{PV}_2^{\mathbf{RN}}$
L_2
$\mathrm{PV}^{\mathbf{RN}}$
L
$\widetilde{\mathrm{PV}}^{\mathbf{RN}}$
$\widetilde{\mathrm{L}}$
$\mathrm{PV}_2^{\mathtt{Gudhi}}$
L_2
$\mathrm{PV}^{\mathtt{Gudhi}}$
L
$\widetilde{\mathrm{PV}}^{\mathtt{Gudhi}}$
$\widetilde{\mathrm{L}}$
Train $\mathrm{Cl}_{\mathbf{RN}}$
$\mathrm{Cl}_{\mathbf{RN}}$
Train $\mathrm{Cl}_{\mathtt{Gudhi}}$
$\mathrm{Cl}_{\mathtt{Gudhi}}$
$\mathrm{Score}_{\mathbf{RN}}$
$\widetilde{\mathrm{Score}}_{\mathbf{RN}}$
$\mathrm{Score}_{\mathtt{Gudhi}}$
$\widetilde{\mathrm{Score}}_{\mathtt{Gudhi}}$

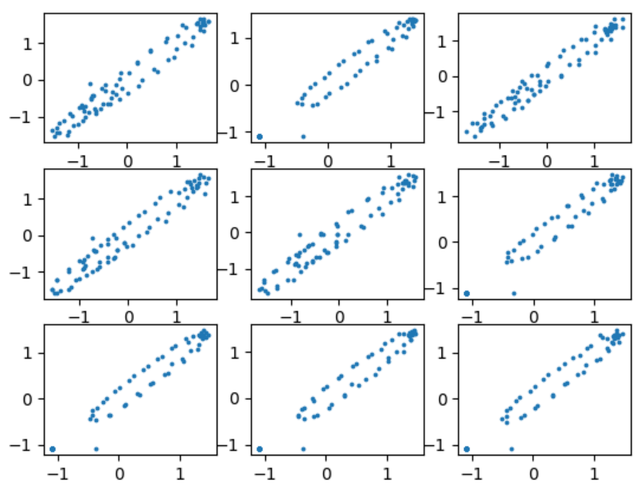
Experimental setup

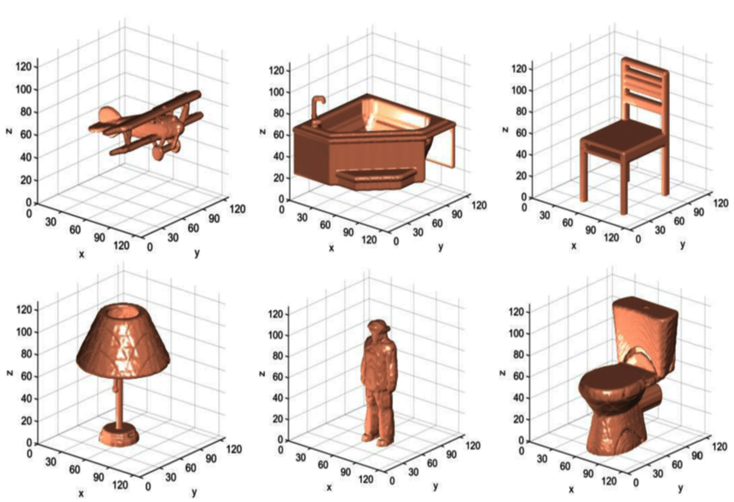


Data	Gudhi (s)	Gudhi$^{\text{DTM}}$ (s)	**RN** (s)
LS	56.3 ± 1.5	155.9 ± 8.1	$\mathbf{0.3 \pm 0.0}$
PI	69.5 ± 3.1	173.7 ± 13.3	$\mathbf{0.4 \pm 0.0}$
P	5.3 ± 1.4	44.7 ± 6.6	$\mathbf{0.2 \pm 0.0}$
UMD	8.0 ± 1.4	55.7 ± 3.6	$\mathbf{0.2 \pm 0.0}$
$\lambda = 2\%$	118.4 ± 4.7	178.5 ± 8.1	$\mathbf{0.2 \pm 0.0}$
$\lambda = 5\%$	117.8 ± 4.5	180.0 ± 9.2	$\mathbf{0.2 \pm 0.0}$

Experimental setup

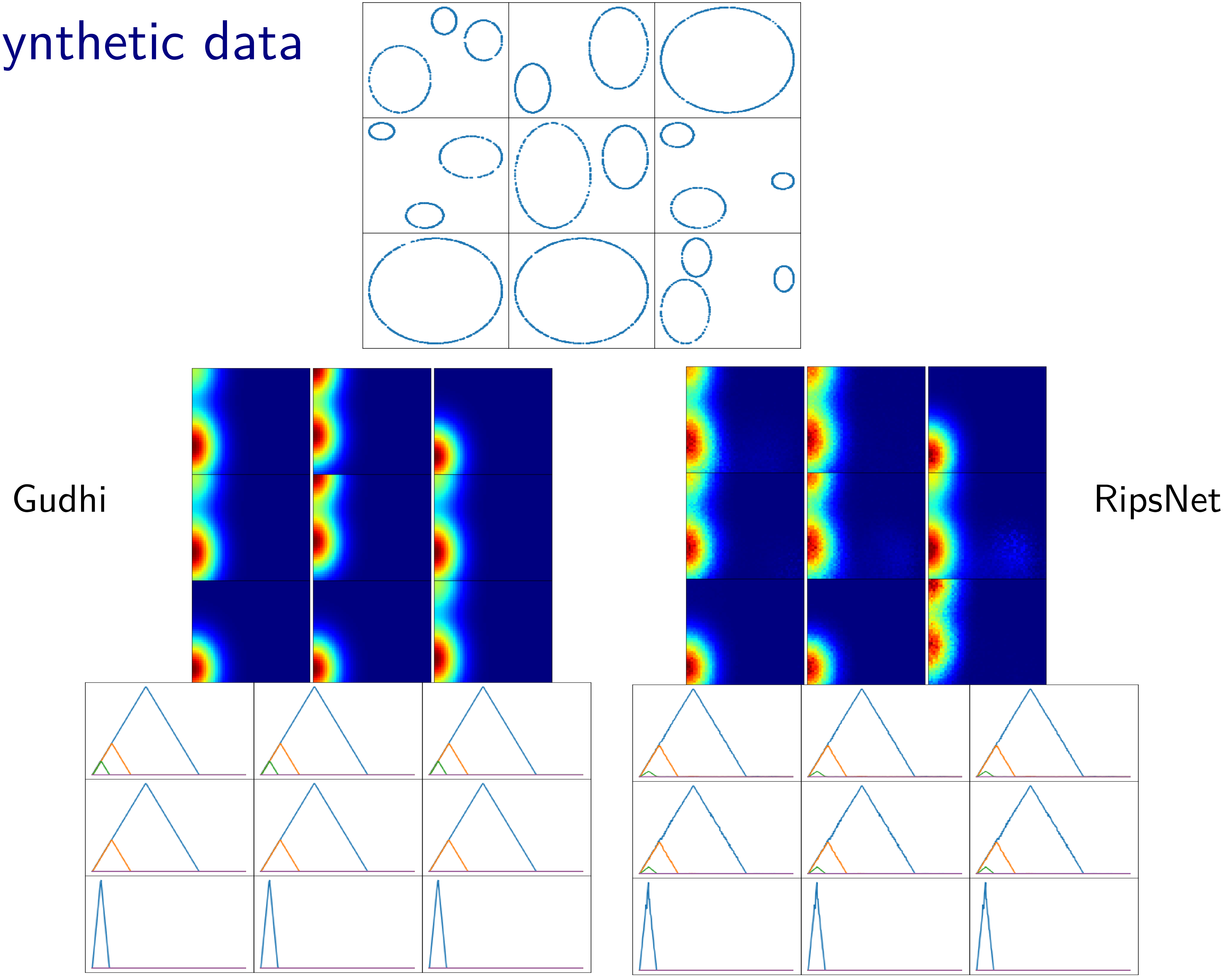




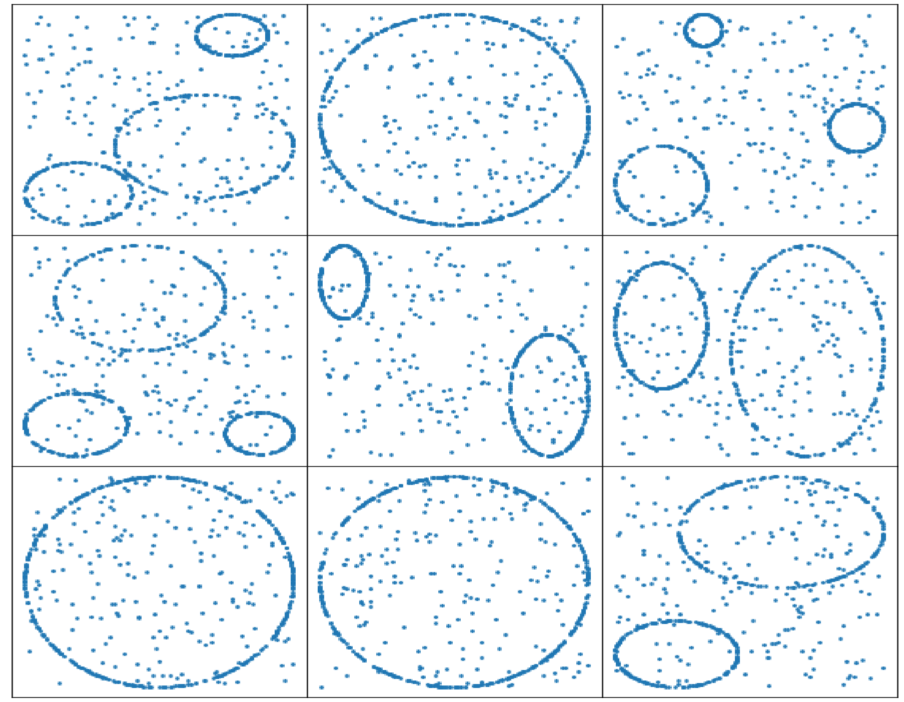


Synth. Data	$\mathrm{Cl}^{XGB}_{\texttt{Gudhi}}$	$\mathrm{Cl}^{XGB}_{\texttt{Gudhi}\,\mathrm{DTM}}$	$\mathrm{Cl}^{XGB}_{\mathbf{RN}}$	$\mathbf{DS}_1$	$\mathbf{DS}_2$
LS	99.9 ± 0.1	99.9 ± 0.1	80.7 ± 3.0	66.4 ± 2.3	66.0 ± 2.4
PI	100.0 ± 0.0	100.0 ± 0.1	81.6 ± 5.3	-	-
$\widetilde{\texttt{LS}}$	66.7 ± 0.0	66.7 ± 0.0	76.3 ± 2.3	66.8 ± 1.0	66.6 ± 2.3
$\widetilde{\texttt{PI}}$	33.3 ± 0.0	65.0 ± 1.3	$\mathbf{77.4 \pm 4.4}$	-	-
UCR Data	$\mathrm{Cl}^{XGB}_{\texttt{Gudhi}}$	$\mathrm{Cl}^{XGB}_{\texttt{Gudhi}\,\mathrm{DTM}}$	$\mathrm{Cl}^{XGB}_{\mathbf{RN}}$	$\mathbf{kNN}_{\mathrm{D}}$	$\mathbf{kNN}_{\mathrm{E}}$
P	70.5 ± 0.0	56.2 ± 0.0	$\mathbf{88.4 \pm 4.1}$	82.9 ± 0.0	78.1 ± 0.0
$\widetilde{\texttt{P}}$	22.5 ± 2.6	53.9 ± 2.5	43.0 ± 7.9	$\mathbf{82.9 \pm 0.0}$	78.1 ± 0.6
SAIBORS2	63.6 ± 0.0	66.2 ± 0.0	$\mathbf{80.2 \pm 5.2}$	73.8 ± 0.0	72.4 ± 0.0
$\widetilde{\texttt{SAIBORS2}}$	56.8 ± 0.8	60.0 ± 1.2	$\mathbf{75.6 \pm 6.6}$	73.7 ± 0.9	72.4 ± 0.4
ECG5000	84.2 ± 0.0	86.2 ± 0.0	90.2 ± 0.2	$\mathbf{93.0 \pm 0.0}$	92.8 ± 0.0
$\widetilde{\texttt{ECG5000}}$	68.9 ± 0.8	71.6 ± 1.0	75.8 ± 4.7	$\mathbf{93.1 \pm 0.3}$	92.8 ± 0.1
UMD	55.6 ± 0.0	54.2 ± 0.0	$\mathbf{71.1 \pm 6.5}$	68.8 ± 0.0	61.1 ± 0.0
$\widetilde{\texttt{UMD}}$	51.8 ± 1.9	48.9 ± 1.6	$\mathbf{69.2 \pm 6.4}$	68.3 ± 1.7	61.1 ± 0.4
GPOVY	98.4 ± 0.0	97.8 ± 0.0	90.4 ± 19.0	$\mathbf{100.0 \pm 0.0}$	100.0 ± 0.0
$\widetilde{\texttt{GPOVY}}$	54.8 ± 0.7	54.3 ± 0.6	82.4 ± 20.7	$\mathbf{100.0 \pm 0.0}$	100.0 ± 0.0
λ (%)	$\mathrm{Cl}^{NN}_{\texttt{Gudhi}}$	$\mathrm{Cl}^{NN}_{\texttt{Gudhi}\,\mathrm{DTM}}$	$\mathrm{Cl}^{NN}_{\mathbf{RN}}$	pointnet	
0	30.4 ± 4.0	30.9 ± 2.0	53.9 ± 2.4	$\mathbf{81.6 \pm 1.1}$	
2	30.3 ± 3.2	31.0 ± 2.7	53.2 ± 2.5	$\mathbf{74.5 \pm 1.6}$	
5	29.9 ± 4.0	31.0 ± 2.7	55.1 ± 3.3	$\mathbf{63.4 \pm 1.6}$	
10	25.2 ± 3.2	29.5 ± 3.1	$\mathbf{51.0 \pm 2.1}$	50.6 ± 1.5	
15	22.9 ± 4.6	25.7 ± 3.1	$\mathbf{46.9 \pm 3.0}$	44.9 ± 1.7	
25	14.4 ± 4.0	18.1 ± 2.6	$\mathbf{42.6 \pm 2.5}$	11.0 ± 0.2	
50	14.0 ± 3.4	13.1 ± 1.9	$\mathbf{31.6 \pm 3.3}$	10.9 ± 0.0	

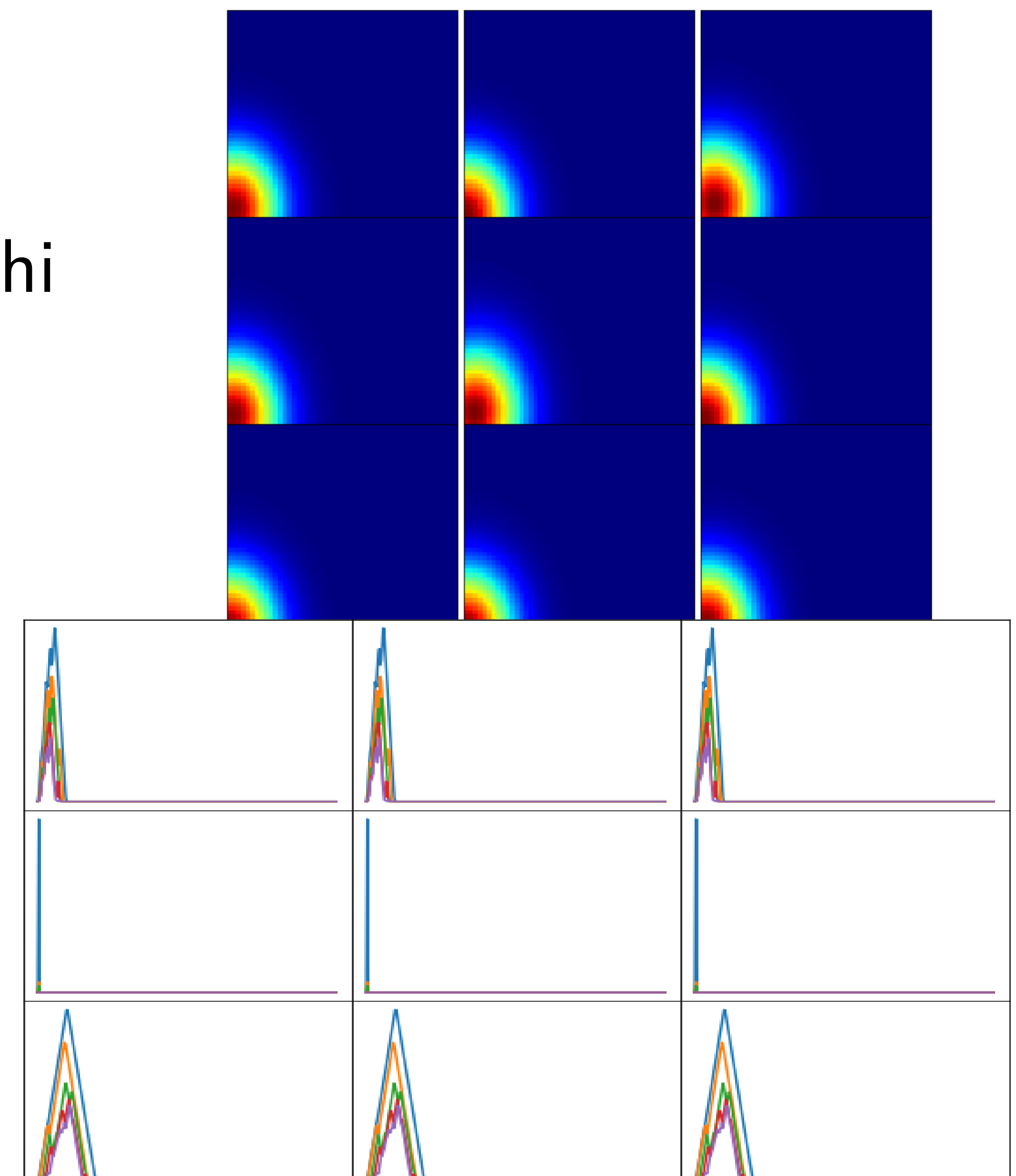
Synthetic data

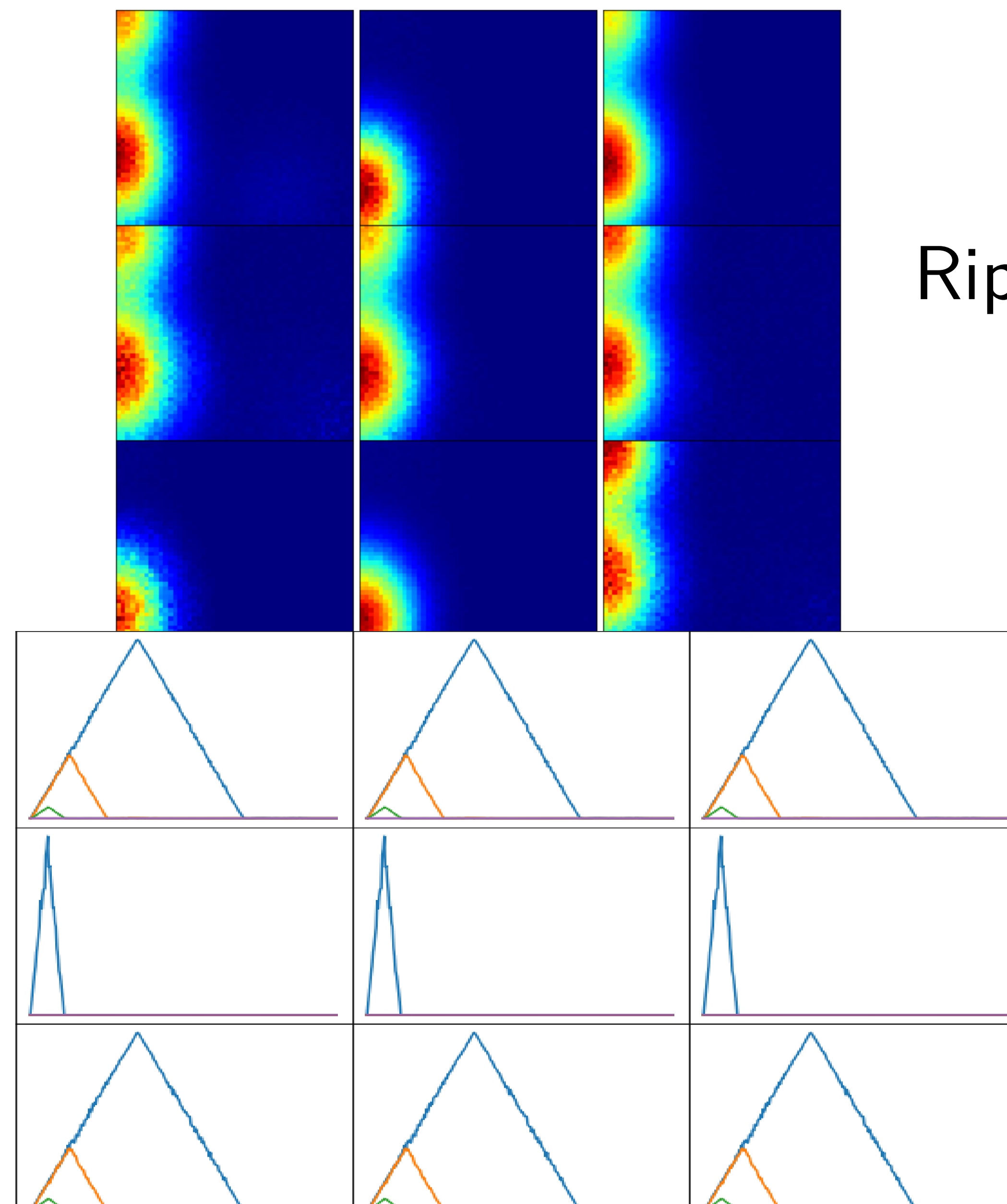


Synthetic data



Gudhi



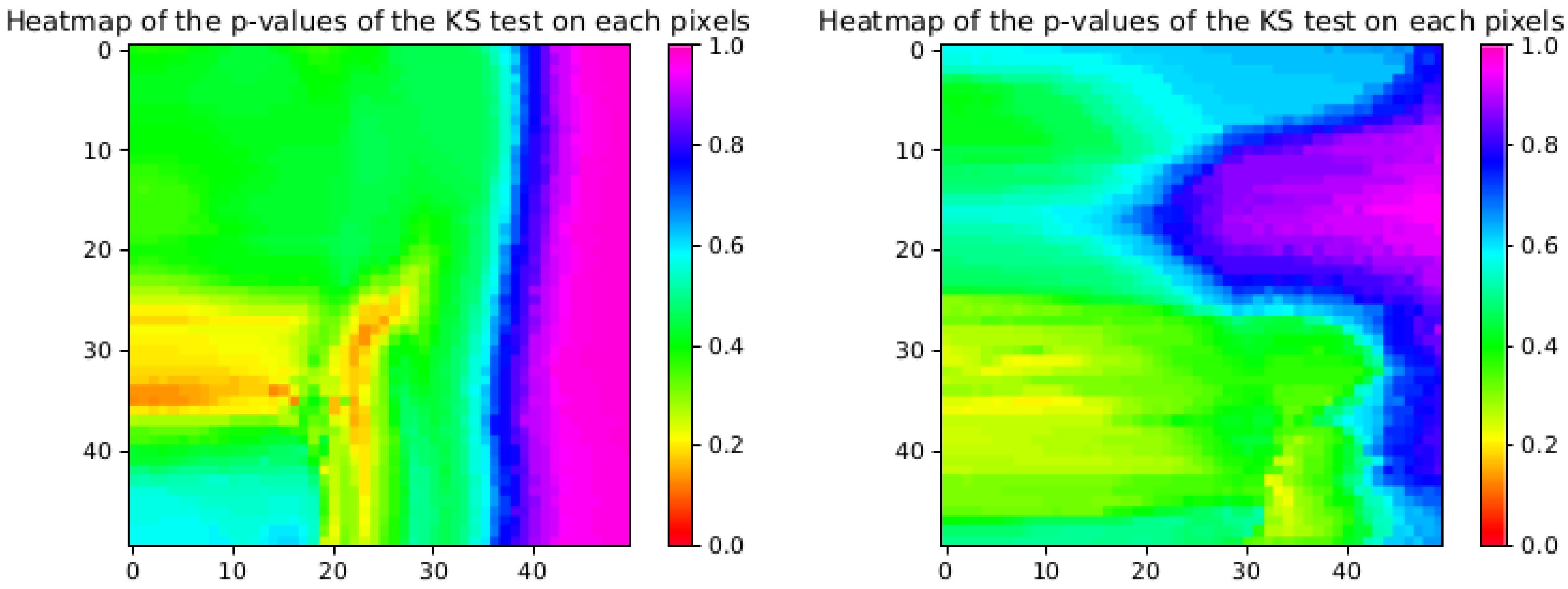


RipsNet

Interpretation

As a measure of where RipsNet reliably predicts the persistence image values, we also run two-sample Kolmogorov-Smirnov tests on each pixel p, and show the heatmap of the p-values (computed with permutations) of the test.

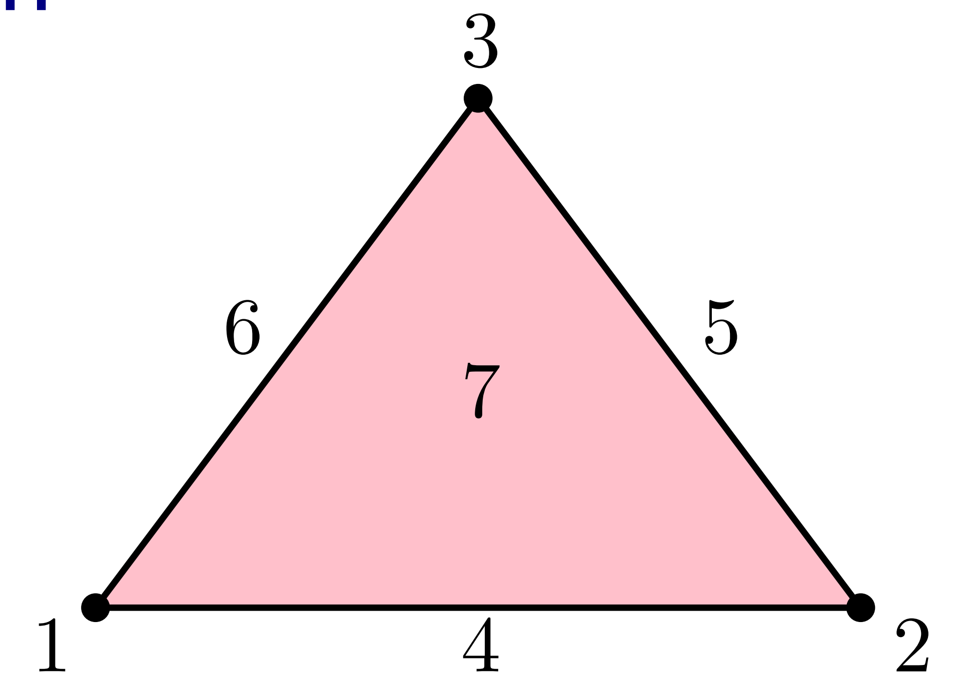
$$\hat{D}^p = \|\hat{F}^p_{\mathrm{Gudhi}} - \hat{F}^p_{\mathrm{RipsNet}}\|_\infty$$



Persistence Computation with Boundary Matrix

Computation with matrix reduction

Input: simplicial filtration



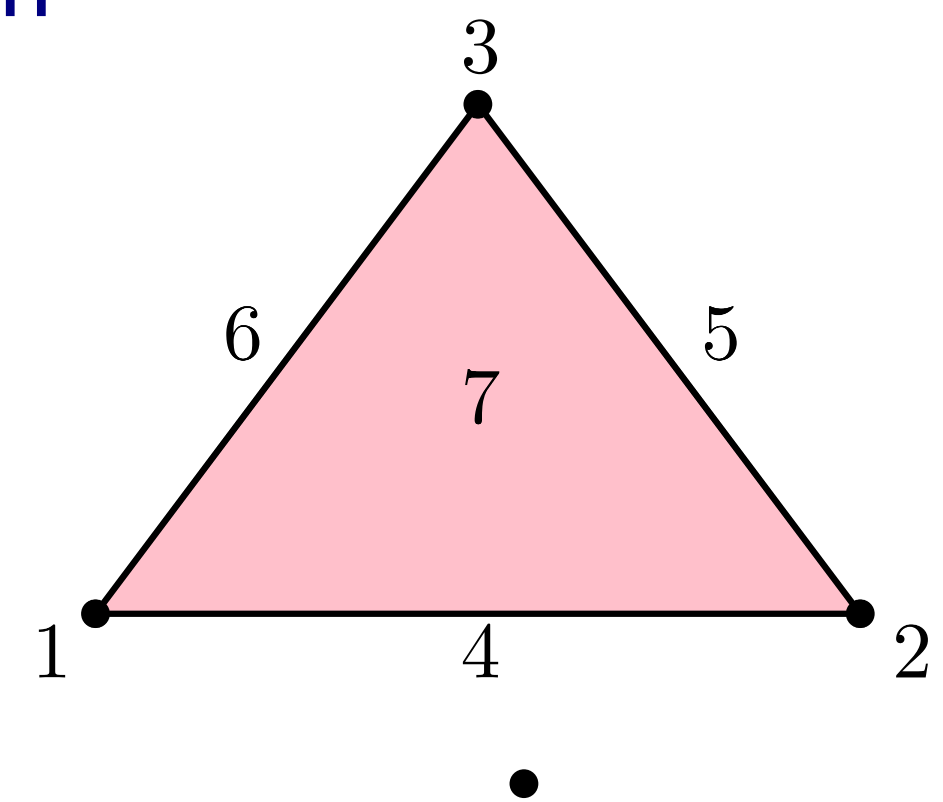
(Persistent) homology can be computed by using the fact that each simplex is either:

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

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

Computation with matrix reduction

Input: simplicial filtration



(Persistent) 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*

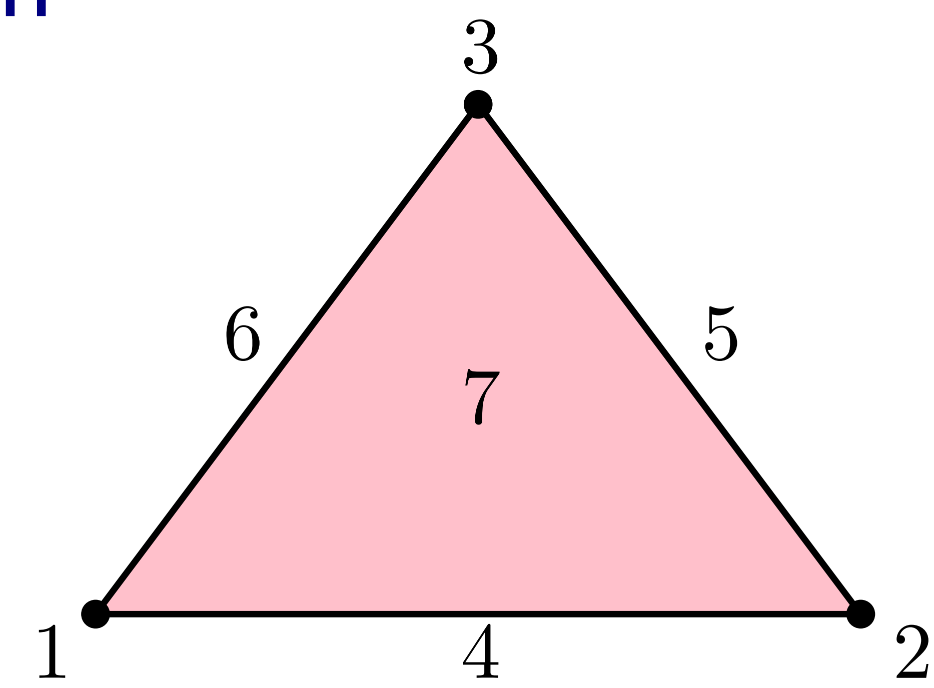
Computation with matrix reduction

Input: simplicial filtration

(Persistent) homology can be computed by using the fact that each simplex is either:

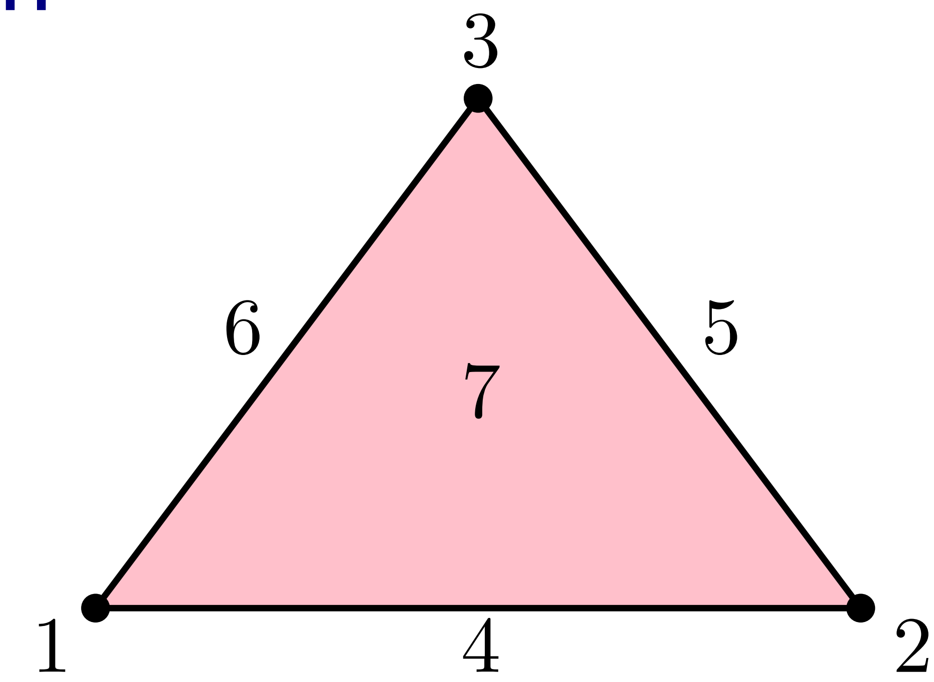
positive, i.e., it *creates a new homology class*

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



Computation with matrix reduction

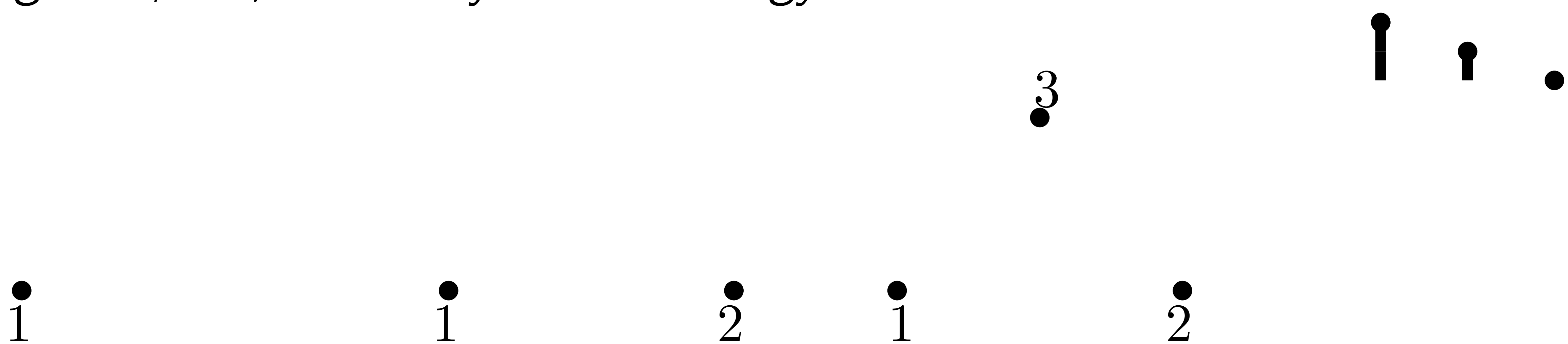
Input: simplicial filtration



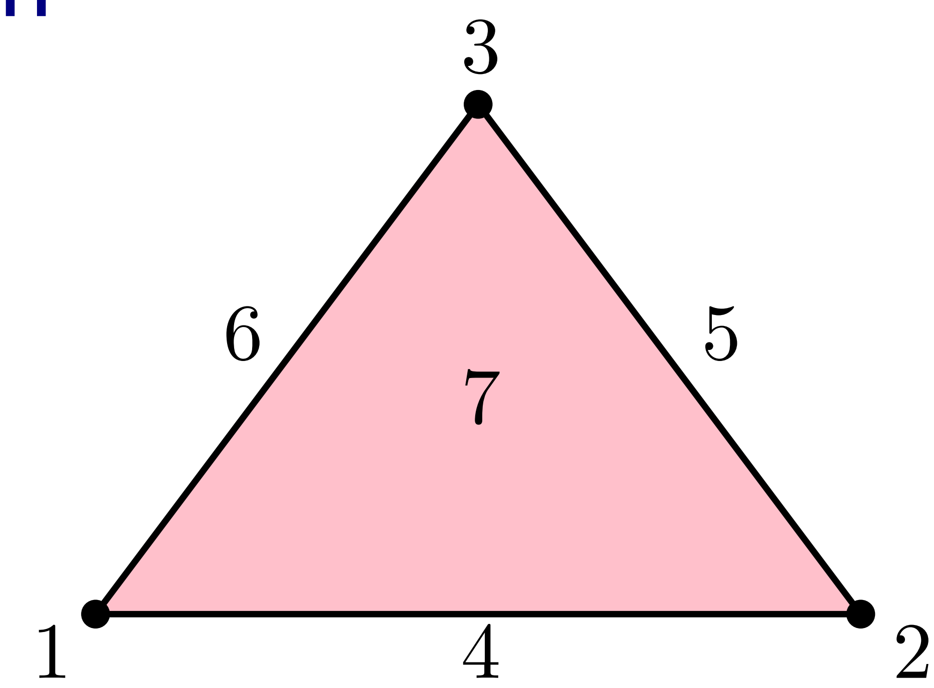
(Persistent) homology can be computed by using the fact that each simplex is either:

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

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



Computation with matrix reduction

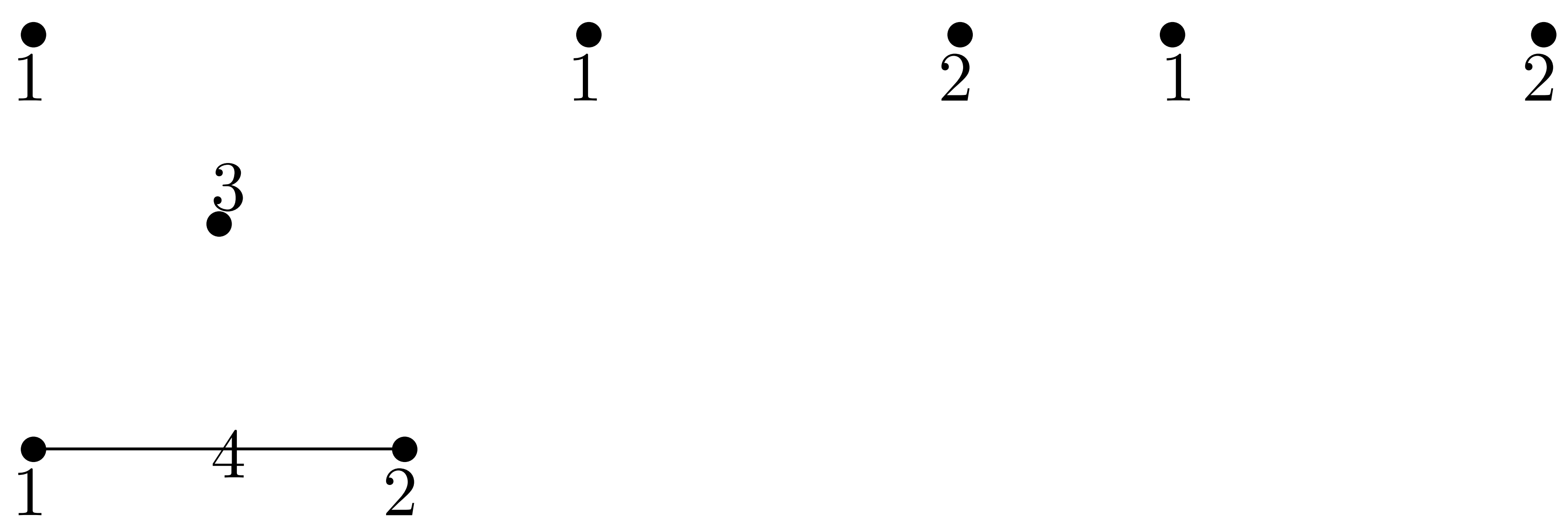


Input: simplicial filtration

(Persistent) homology can be computed by using the fact that each simplex is either:

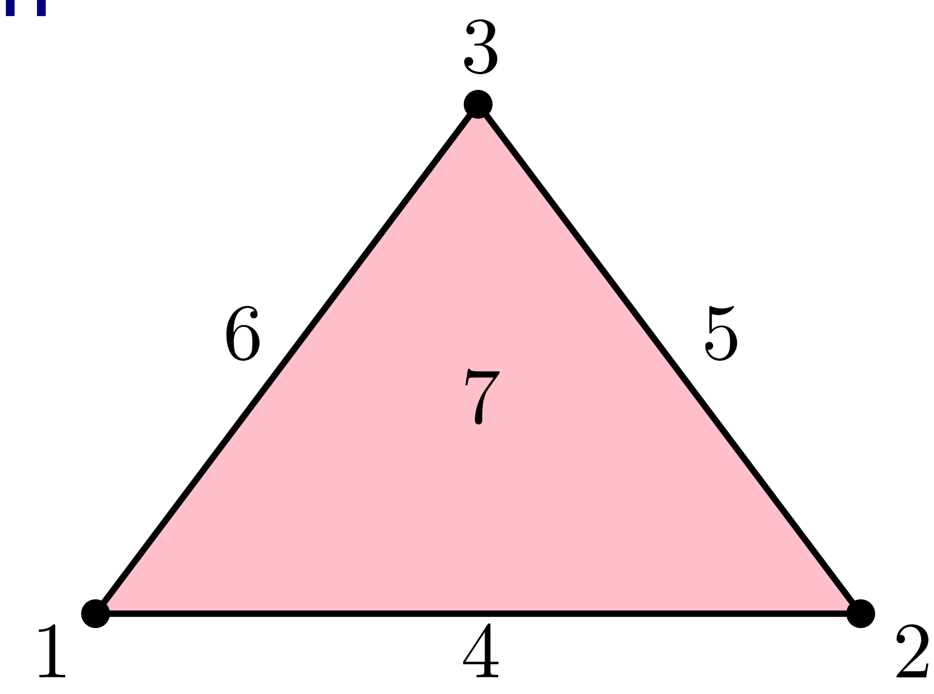
positive, i.e., it *creates a new homology class*

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



Computation with matrix reduction

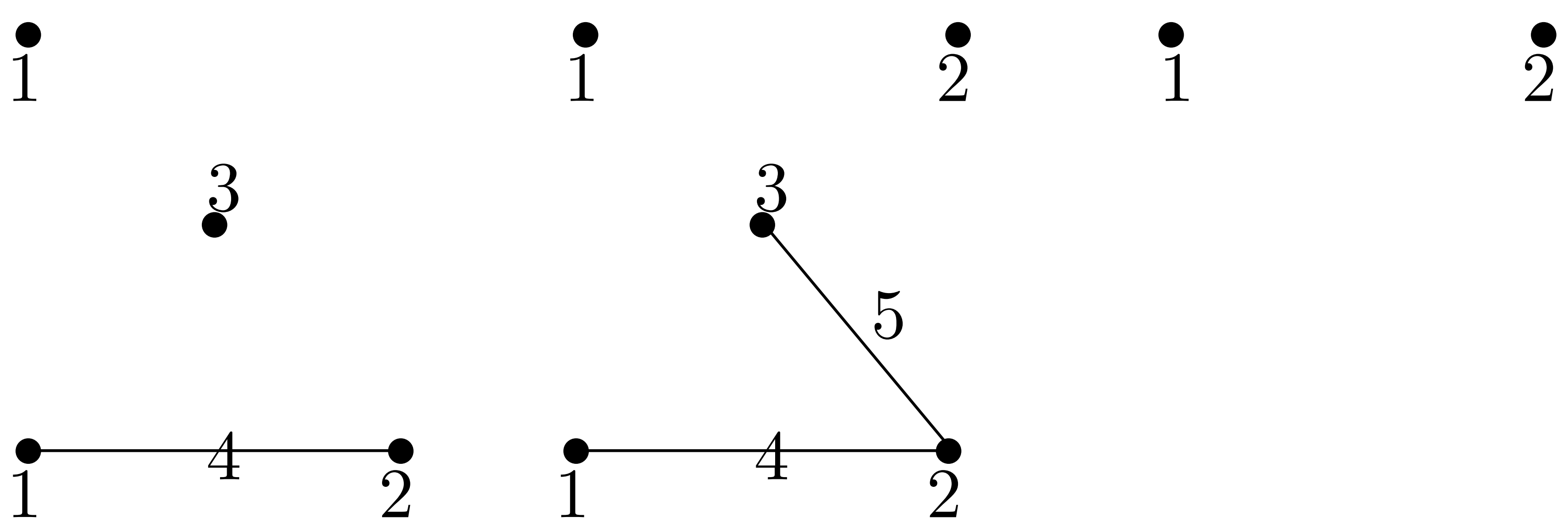
Input: simplicial filtration



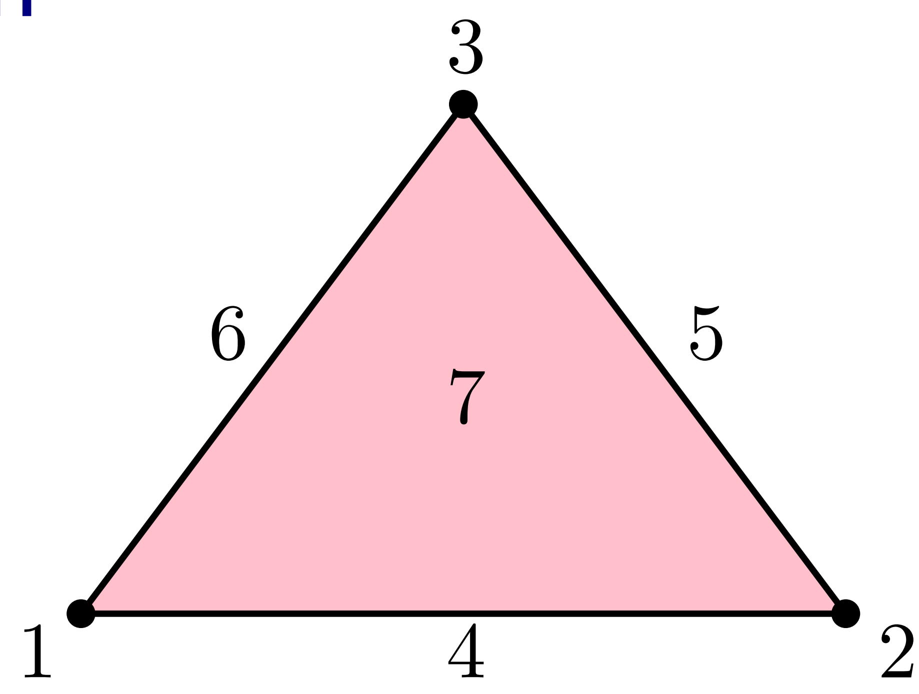
(Persistent) homology can be computed by using the fact that each simplex is either:

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

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



Computation with matrix reduction

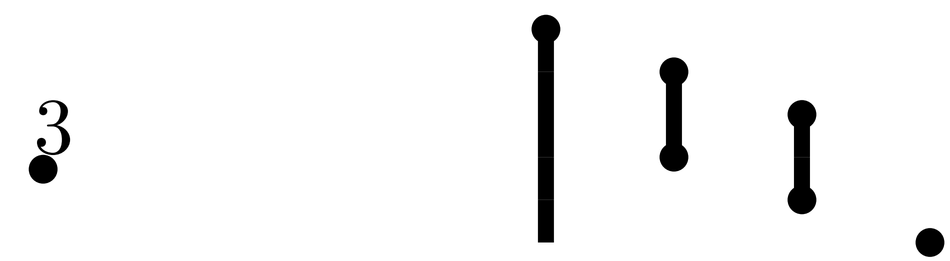


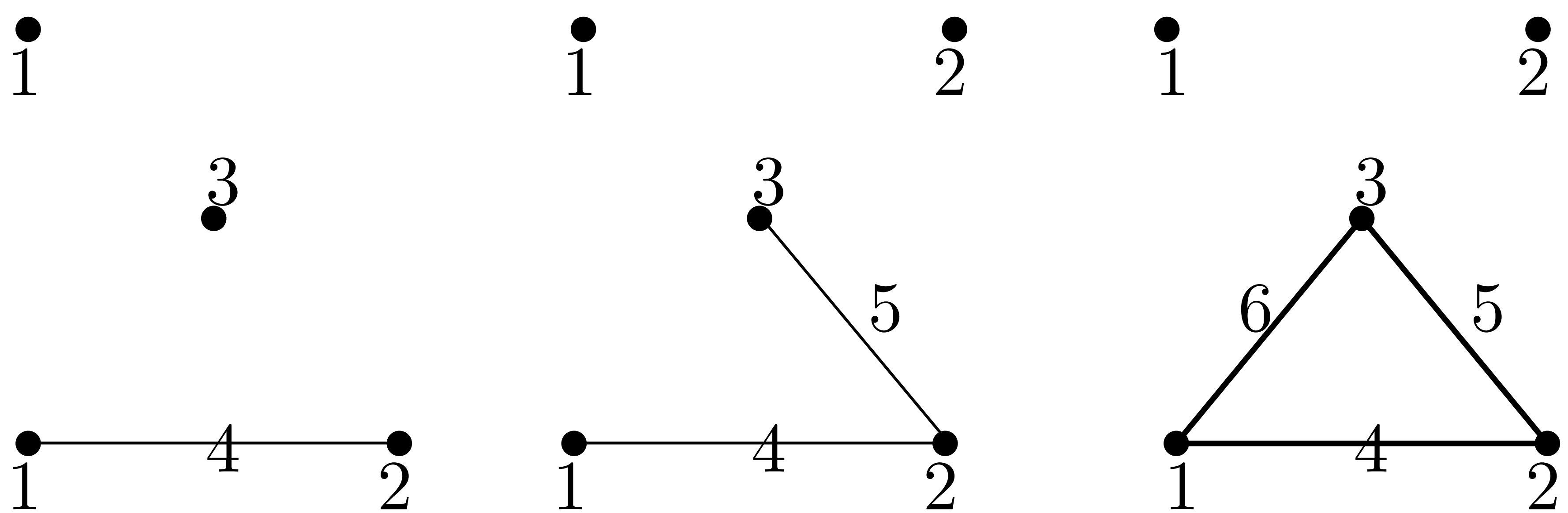
Input: simplicial filtration

(Persistent) homology can be computed by using the fact that each simplex is either:

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

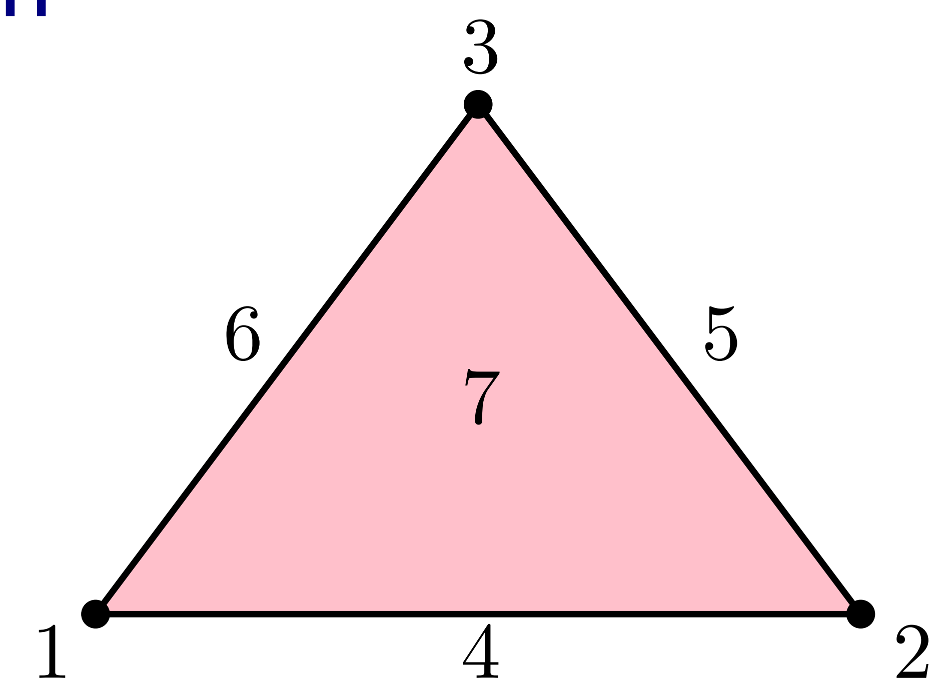
negative, i.e., it *destroys an homology class*





Computation with matrix reduction

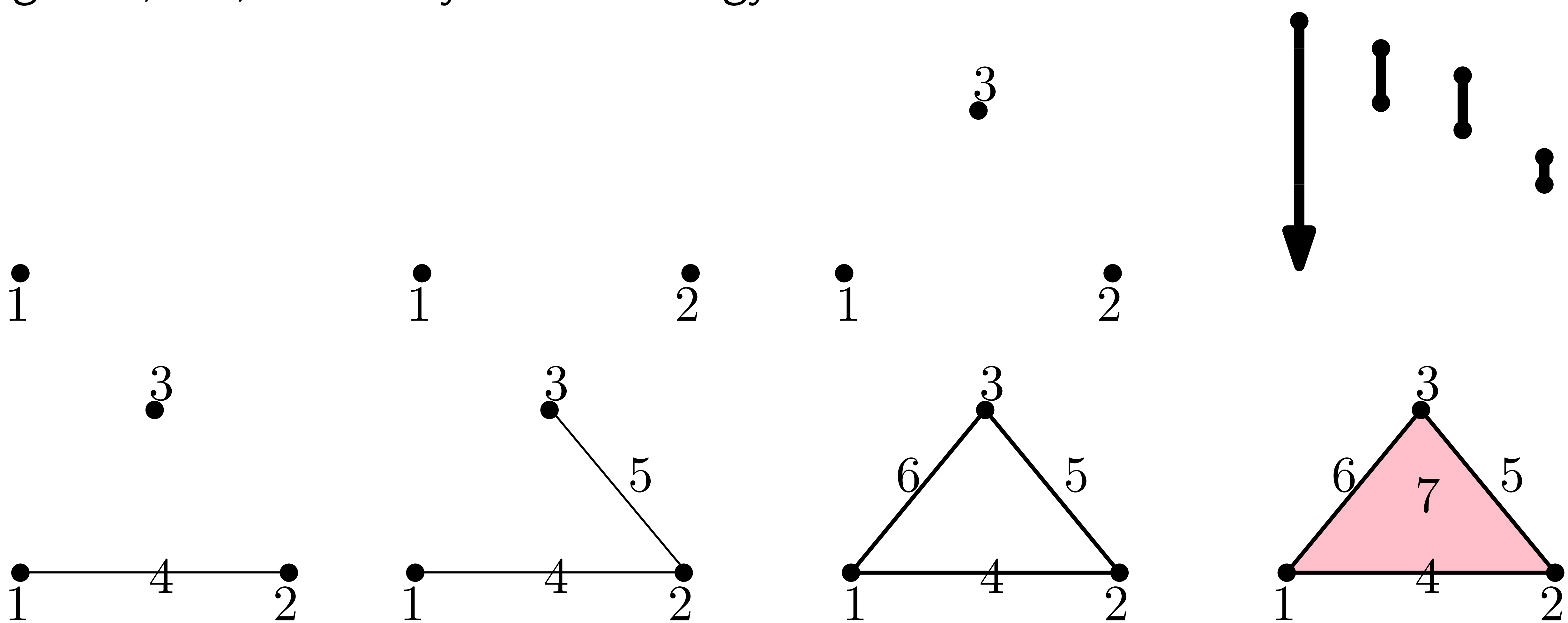
Input: simplicial filtration



(Persistent) 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*

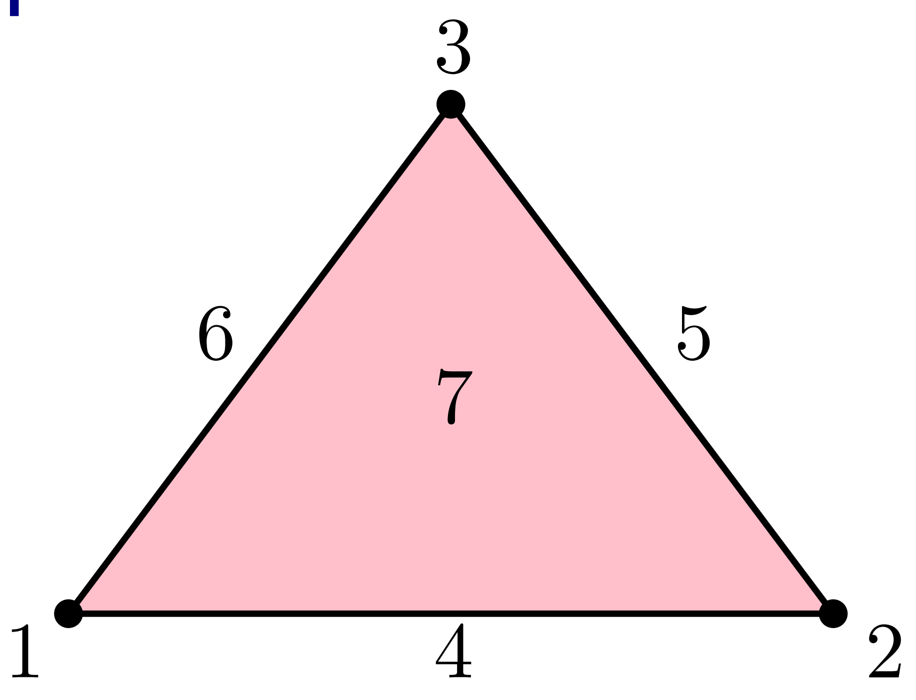


Computation with matrix reduction

Input: simplicial filtration

given as *boundary matrix*

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

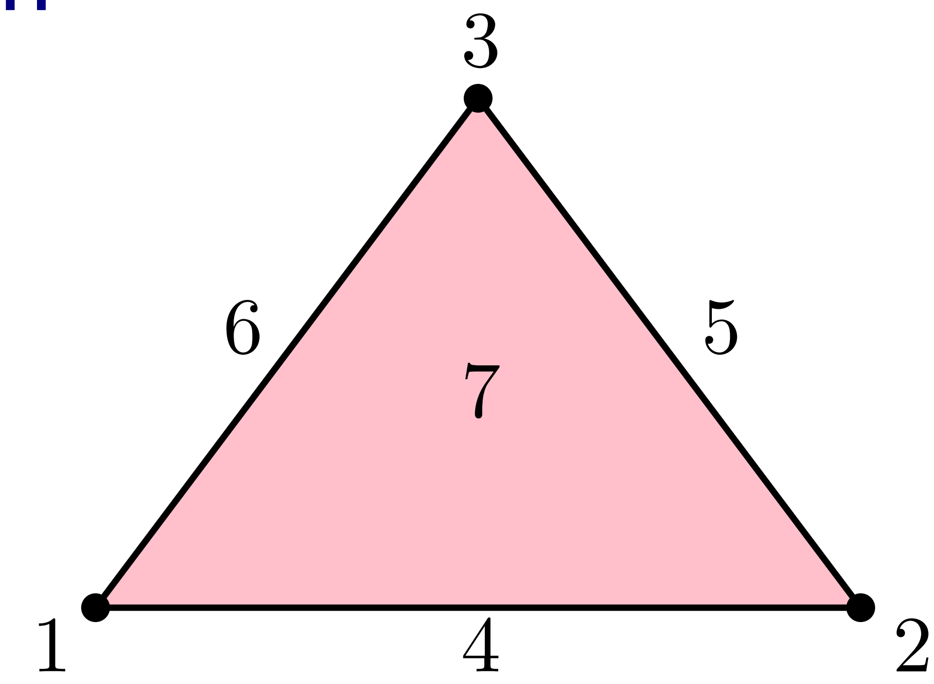


Computation with matrix reduction

Input: simplicial filtration

given as *boundary matrix*

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

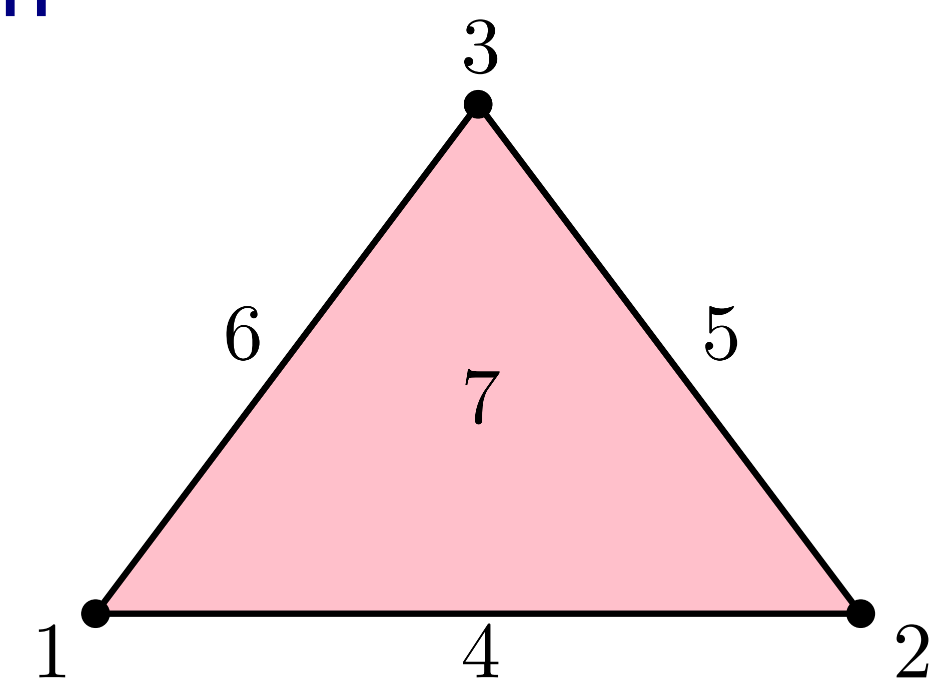


Computation with matrix reduction

Input: simplicial filtration

given as *boundary matrix*

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

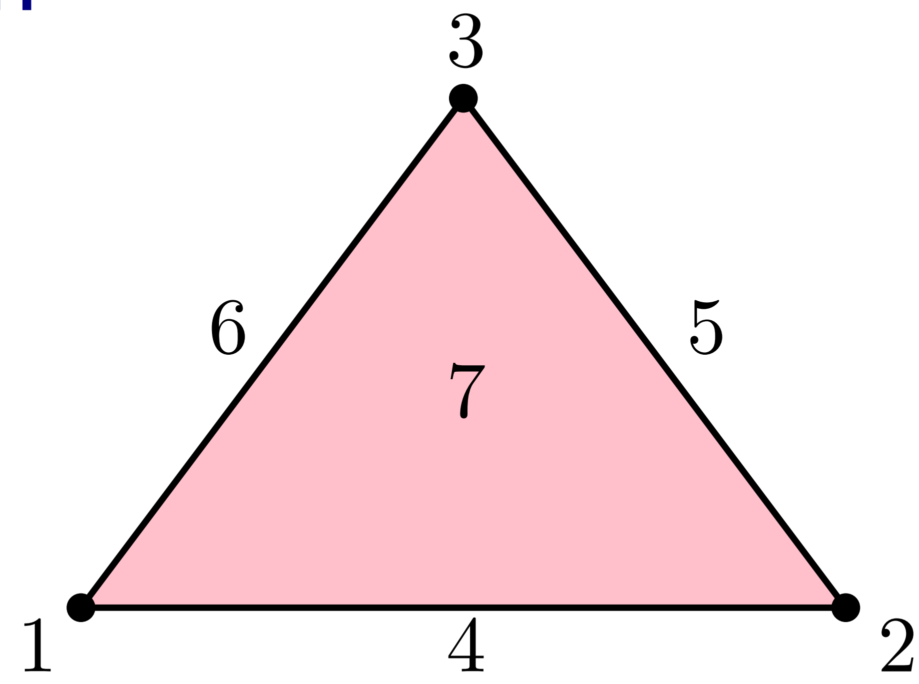


Computation with matrix reduction

Input: simplicial filtration

given as *boundary matrix*

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

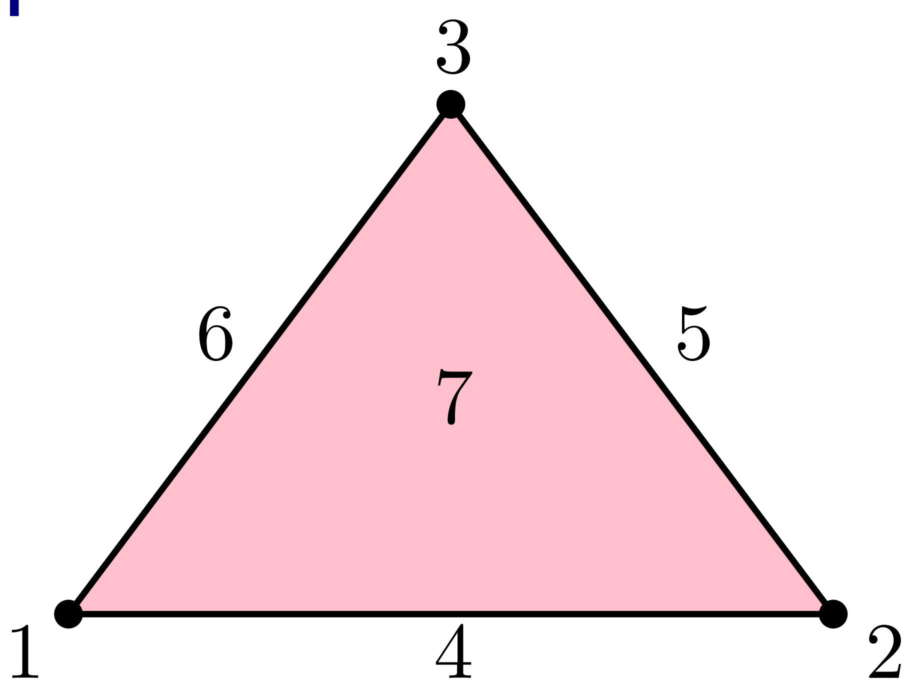


Computation with matrix reduction

Input: simplicial filtration

given as *boundary matrix*

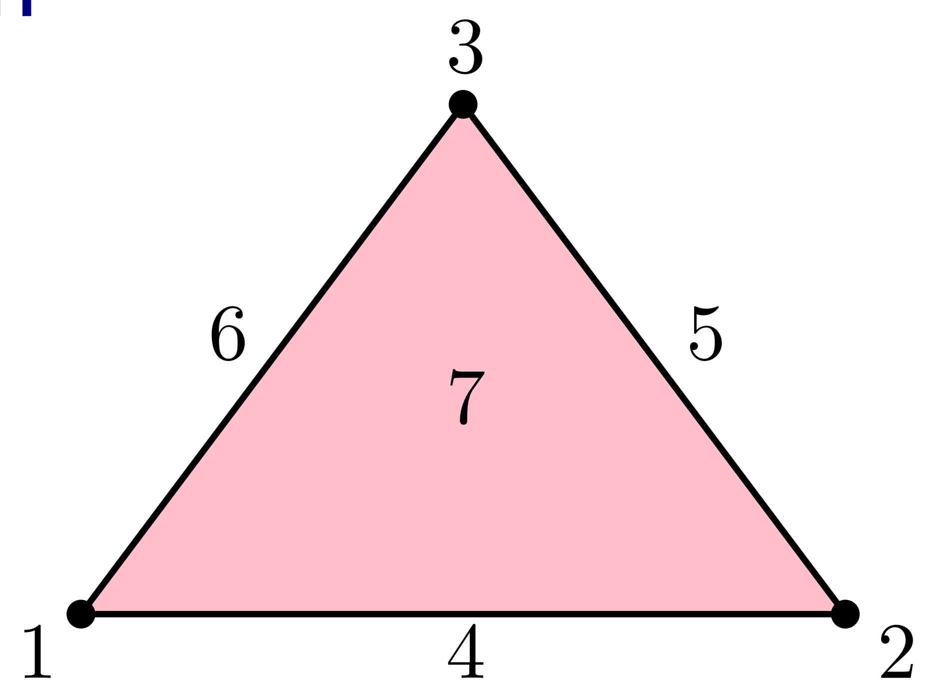
	1	2	3	4	5	6	7
1				•		•	
2				•	•		
3					•	•	
4							•
5							•
6							•
7							



Computation with matrix reduction

Input: simplicial filtration
given as *boundary matrix*

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

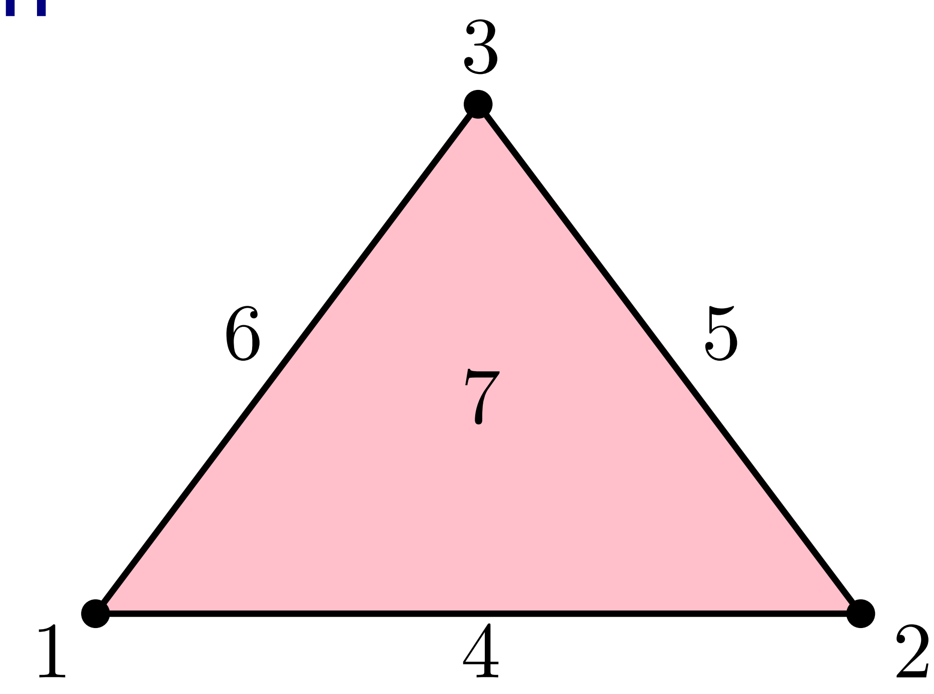



```
for j=1 to m do:
    while ∃k < j s.t. low(k) ==low(j) do:
        col(j) =col(j)+col(k)
```

Computation with matrix reduction

Input: simplicial filtration

given as *boundary matrix*



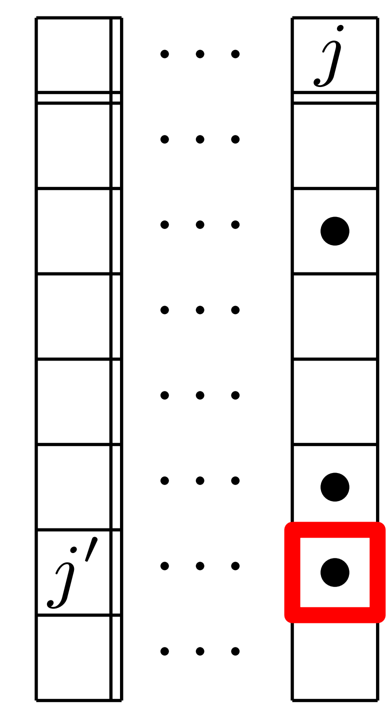
	1	2	3	4	5	6	7
1				●		●	
2				●	●		
3					●	●	
4							●
5							●
6							●
7							

`for` j=1 `to m do:`

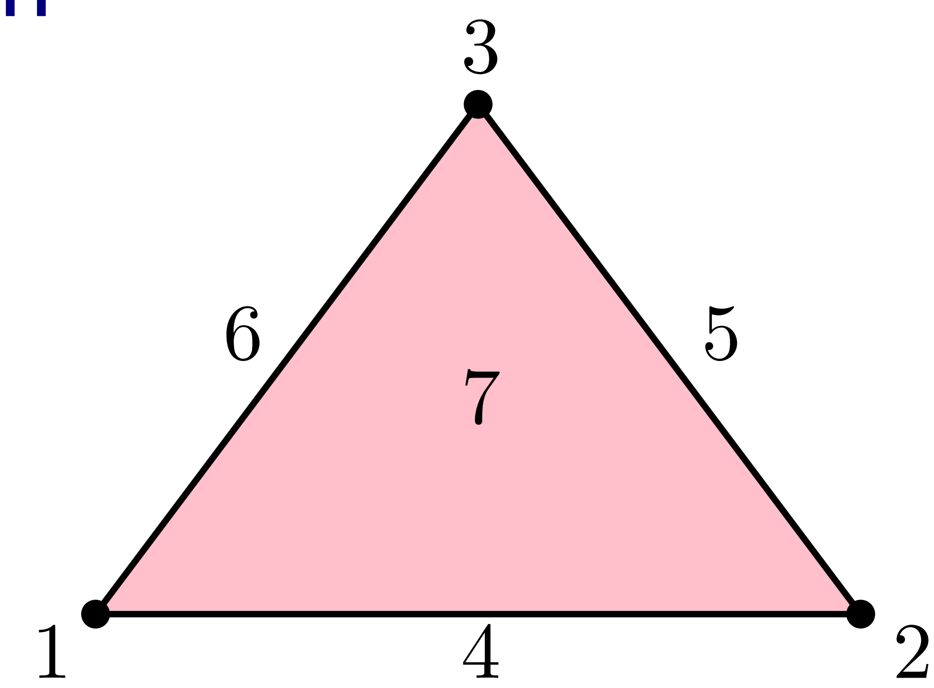
`while` $\exists k < j$ `s.t.` low(k) ==low(j) `do:`

col(j) =col(j)+col(k)

low$(j) = j'$



Computation with matrix reduction



Input: simplicial filtration

given as *boundary matrix*

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

for j=1 to m do:

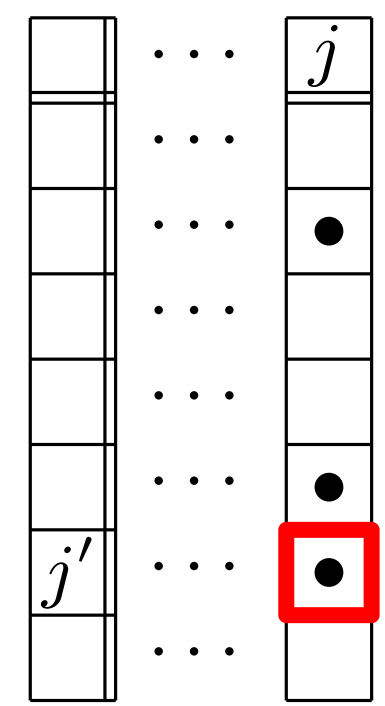
while $\exists k < j$ s.t. low(k) ==low(j) do:

col(j) =col(j)+col(k)

5	6
	•
•	
•	•

6 = 6+5

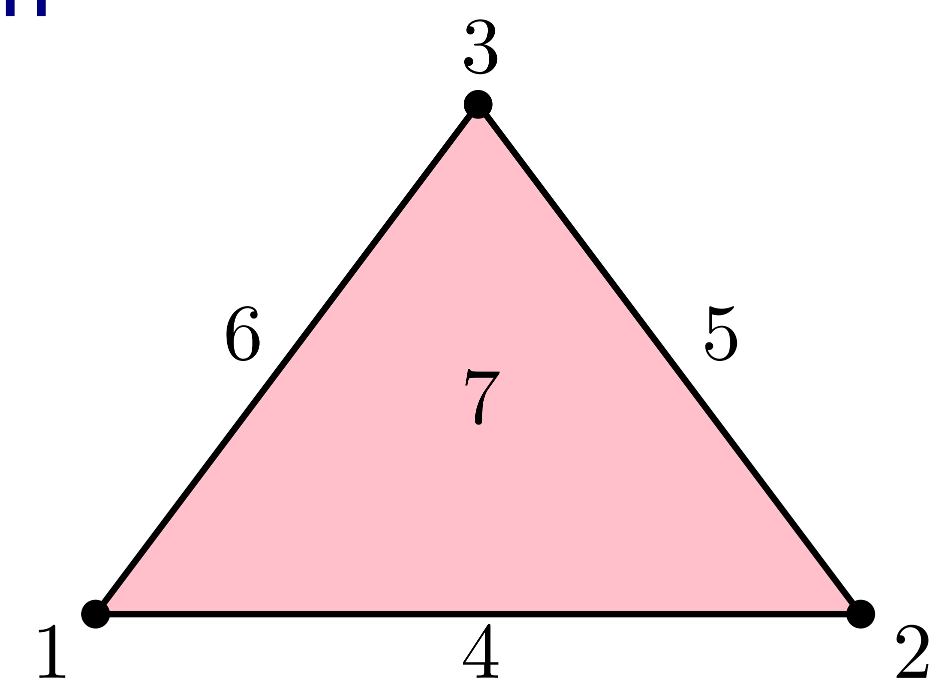
low$(j) = j'$



Computation with matrix reduction

Input: simplicial filtration

given as *boundary matrix*



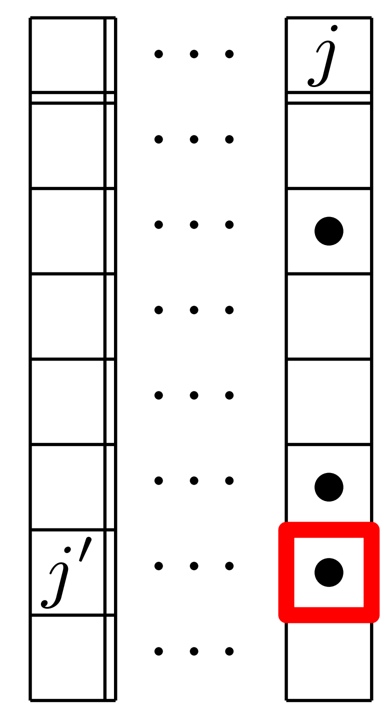
	1	2	3	4	5	6	7
1				•		•	
2				•	•	•	
3					•		
4							•
5							•
6							•
7							

```
for j=1 to m do:
    while ∃k < j s.t. low(k) ==low(j) do:
        col(j) =col(j)+col(k)
```

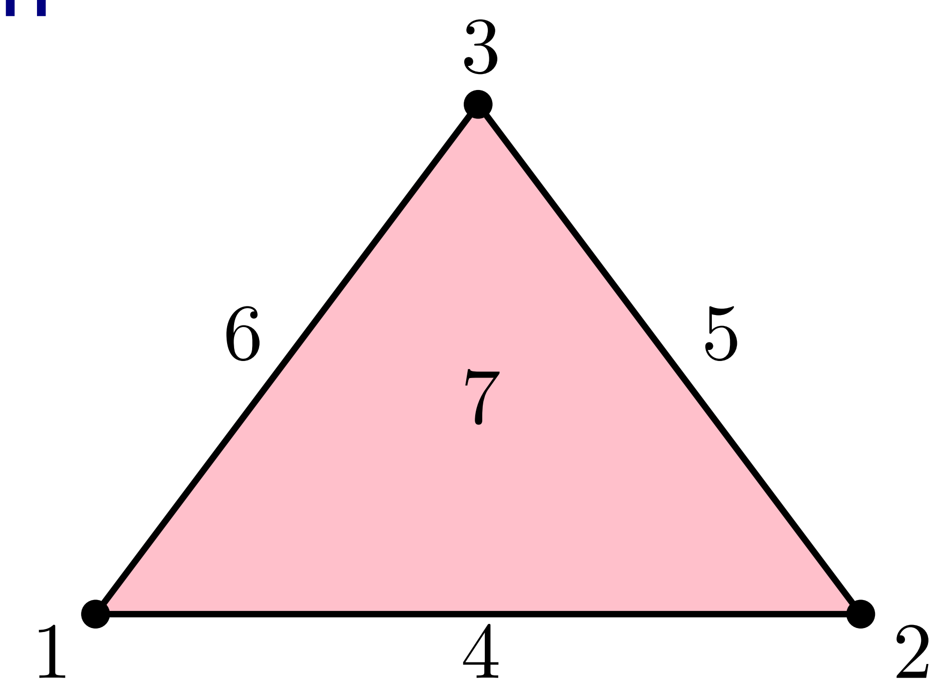
5	6
	•
•	•
•	

6 = 6+5

low$(j) = j'$



Computation with matrix reduction



Input: simplicial filtration

given as *boundary matrix*

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

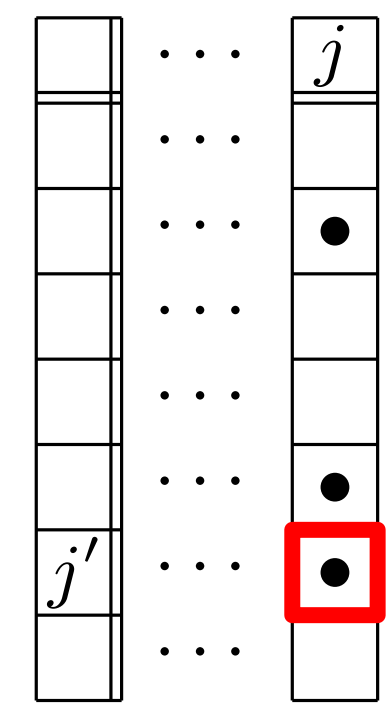
```
for j=1 to m do:
    while ∃k < j s.t. low(k) == low(j) do:
        col(j) = col(j) + col(k)
```

5	6
	•
•	•
•	

4		6
•		•
•		•

6 = 6+5
6 = 6+4

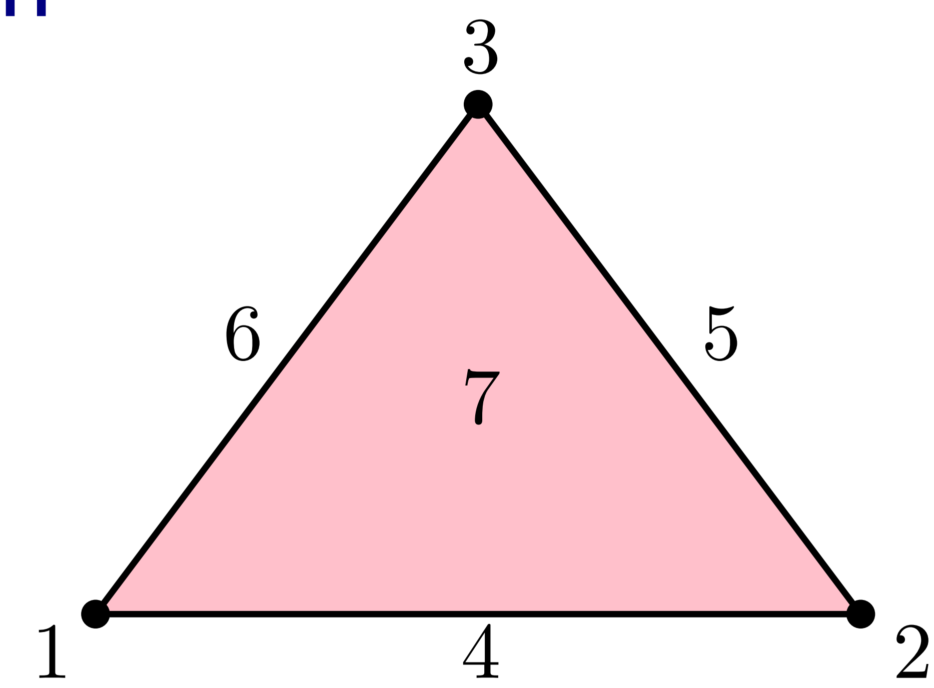
$\mathsf{low}(j) = j'$



Computation with matrix reduction

Input: simplicial filtration

given as *boundary matrix*



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

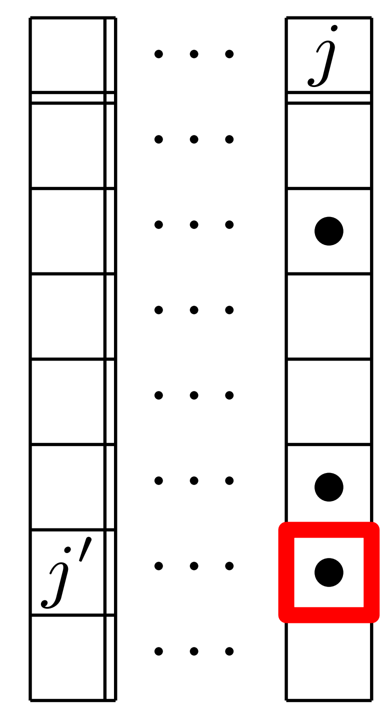
```
for j=1 to m do:
    while ∃k < j s.t. low(k) ==low(j) do:
        col(j) =col(j)+col(k)
```

5	6
	●
●	●
●	

4		6
●		
●		

6 = 6+5
6 = 6+4

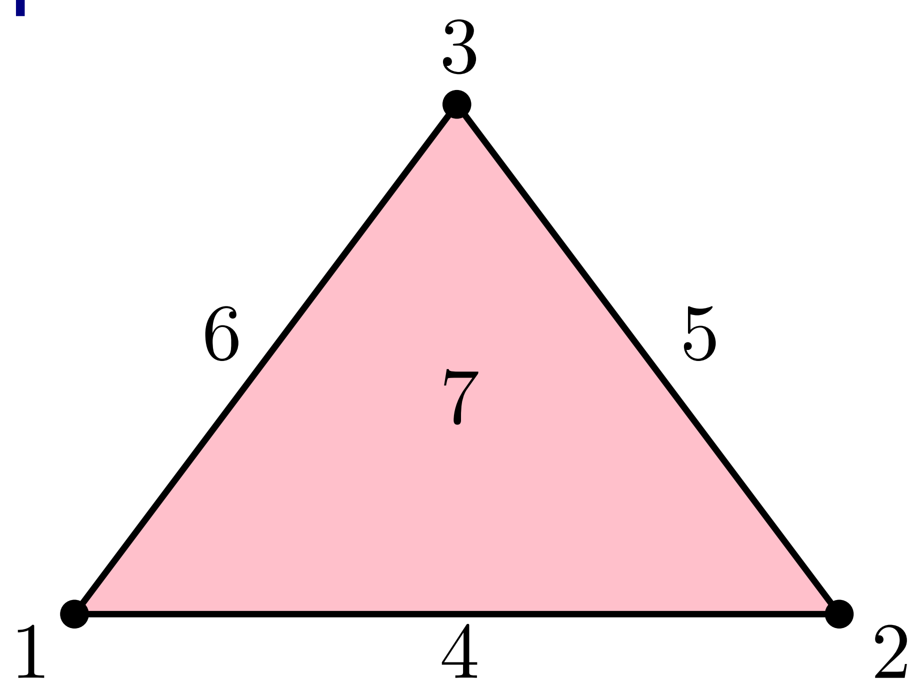
$\mathrm{low}(j) = j'$



Computation with matrix reduction

Input: simplicial filtration

Output: boundary matrix

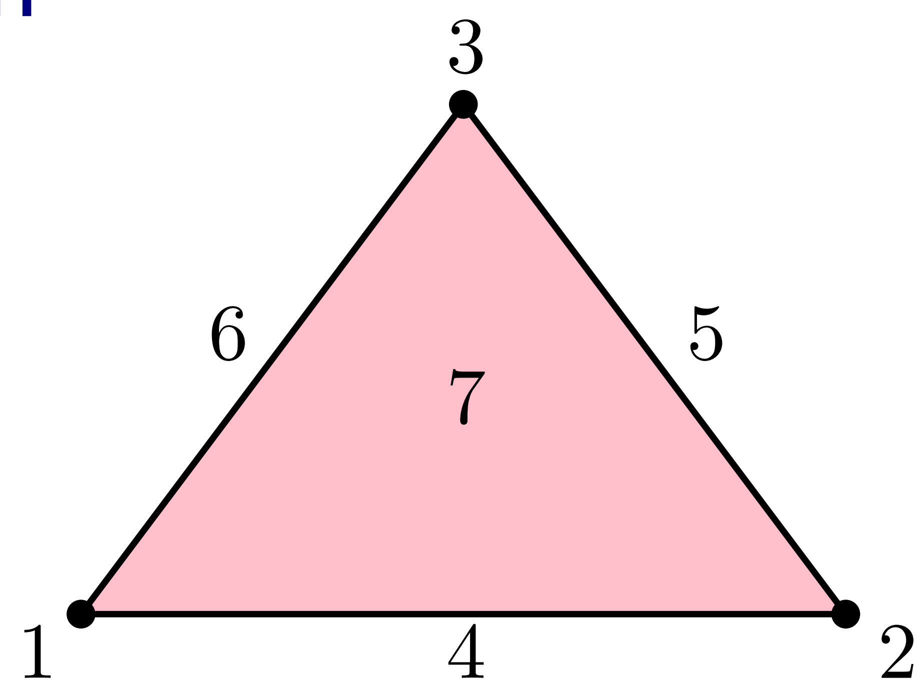


	1	2	3	4	5	6	7
1				*		*	
2				*	*		
3					*	*	
4							*
5							*
6							*
7							

Computation with matrix reduction

Input: simplicial filtration

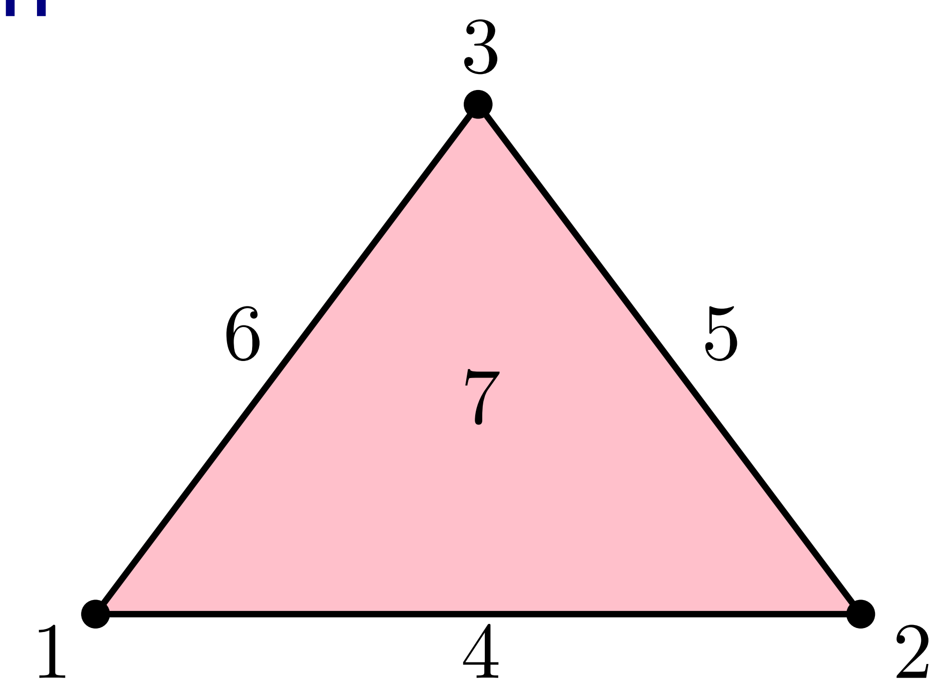
Output: boundary matrix reduced to column-echelon form



	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 matrix reduction



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: $[1, +\infty)$

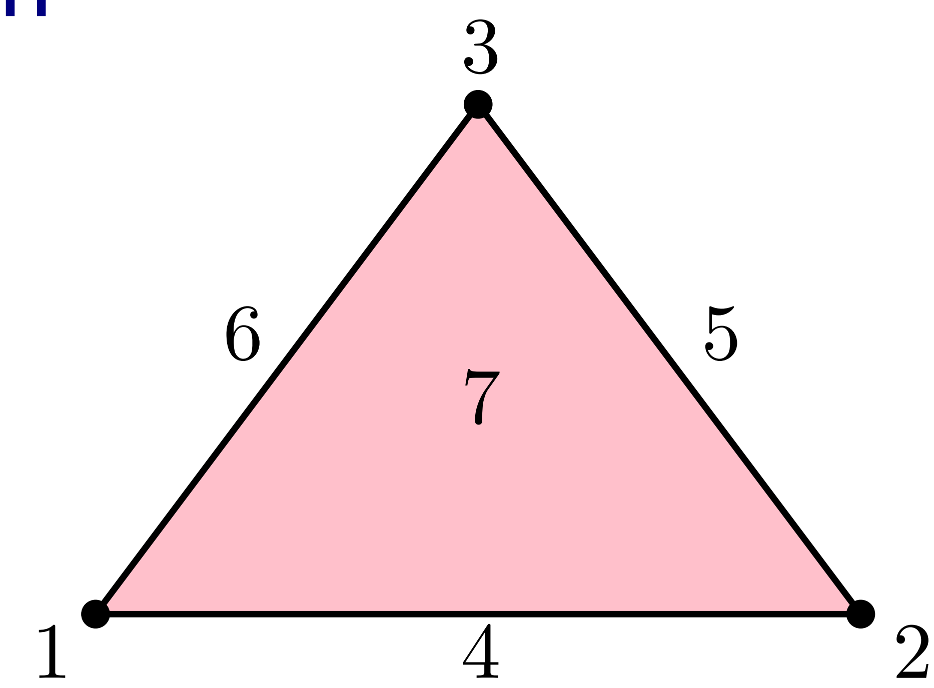
	1	2	3	4	5	6	7
1				*		*	
2				*	*		
3					*	*	
4							*
5							*
6							*
7							

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

Computation with matrix reduction

Input: simplicial filtration

Output: boundary matrix
reduced to column-echelon form



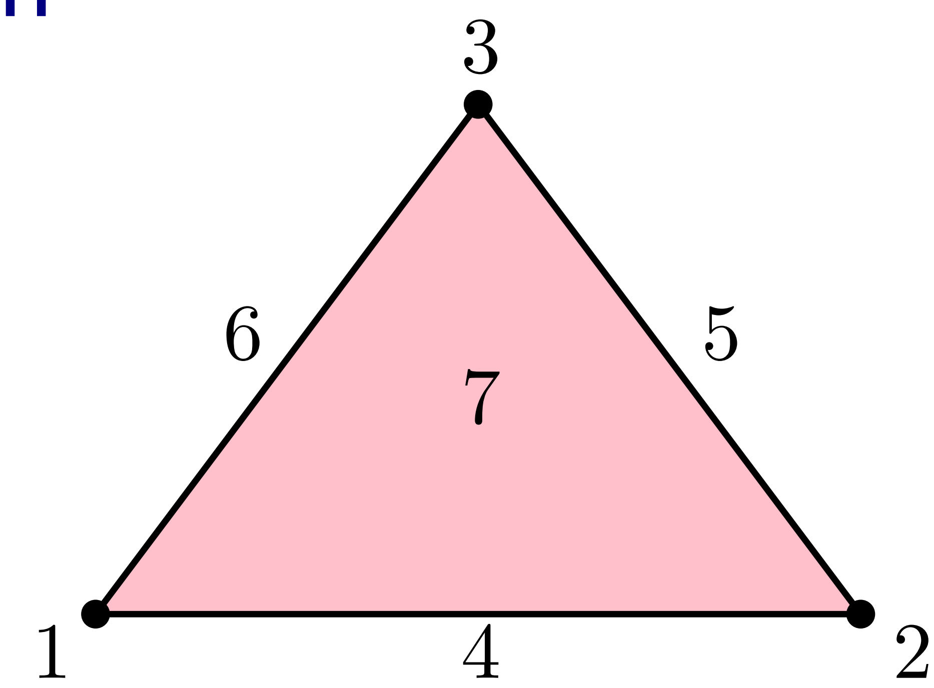
○ simplex pairs give finite intervals:
$[2,4)$, $[3,5)$, $[6,7)$

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

A persistence diagram D is made of all $(\mathcal{F}(\sigma_+),\mathcal{F}(\sigma_-)) \in \mathbb{R}^2$ where σ_+ (resp. σ_-) is positive (resp. negative), and $\mathcal{F}$ is the filtration function.

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

Computation with matrix reduction



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: $[1,+\infty)$

A persistence diagram D is made of all $(\mathcal{F}(\sigma_+),\mathcal{F}(\sigma_-)) \in \mathbb{R}^2$ where σ_+ (resp. σ_-) is positive (resp. negative), and $\mathcal{F}$ is the filtration function.

Thus we can define the gradient of a point $p = (\mathcal{F}(\sigma_+),\mathcal{F}(\sigma_-)) \in D$ as

$$\nabla p = [\nabla \mathcal{F}(\sigma_+), \nabla \mathcal{F}(\sigma_-)]$$

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

Persistence Diagram Embeddings into Hilbert Spaces

[*On the Metric Distortion of Embedding Persistence Diagrams into separable Hilbert Spaces*, Bauer, C., SoCG, 2019]

The space of persistence diagrams

Q: What happens in general when one embeds PDs in Hilbert?

Def: Two metrics d, d' are *equivalent* if

$$\exists 0 < A, B < +\infty \text{ s.t. } A\,\mathrm{d}(\cdot,\cdot) \leq \mathrm{d}'(\cdot,\cdot) \leq B\,\mathrm{d}(\cdot,\cdot)$$

The space of persistence diagrams

Q: What happens in general when one embeds PDs in Hilbert?

Def: Two metrics d, d' are *equivalent* if

$$\exists 0 < A, B < +\infty \text{ s.t. } A\,\mathrm{d}(\cdot,\cdot) \leq \mathrm{d}'(\cdot,\cdot) \leq B\,\mathrm{d}(\cdot,\cdot)$$

Prop: $\mathcal{H}$ Hilbert with dot product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ and distance $\| \cdot \|_{\mathcal{H}}$. Assume $\mathrm{d}_{\mathcal{H}}$ and d_B or d_q are equivalent.

$$(i)\ \mathcal{H} = \mathbb{R}^d \Rightarrow \textbf{Impossible}$$

even if the PDs are included in $[-L, L]^2$ and have less than N points

$$(ii)\ \mathcal{H} \text{ separable, } p = 1 \Rightarrow \text{either } A \to 0 \text{ or } B \to +\infty$$

$$\text{when } L, N \to +\infty$$

The space of persistence diagrams

Q: What happens in general when one embeds PDs in Hilbert?

Def: Two metrics d, d' are *equivalent* if

$$\exists 0 < A, B < +\infty \text{ s.t. } A\,\mathrm{d}(\cdot,\cdot) \le \mathrm{d}'(\cdot,\cdot) \le B\,\mathrm{d}(\cdot,\cdot)$$

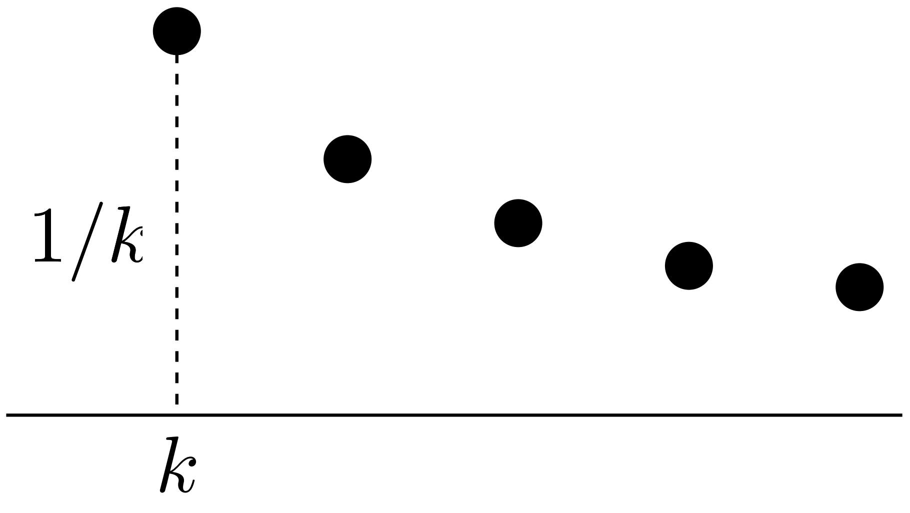
Proof:

(ii) The space of PDs with possibly infinite number of points is not separable with respect to d_1

Consider $S = \{D_u\}_{u \in \{0,1\}^{\mathbb{N}}}$

where $D_u = \left\{ \left(k, k + \frac{1}{k}\right) \, : \, u_k = 1 \right\}$

S is not countable with d_1



The space of persistence diagrams

Q: What happens in general when one embeds PDs in Hilbert?

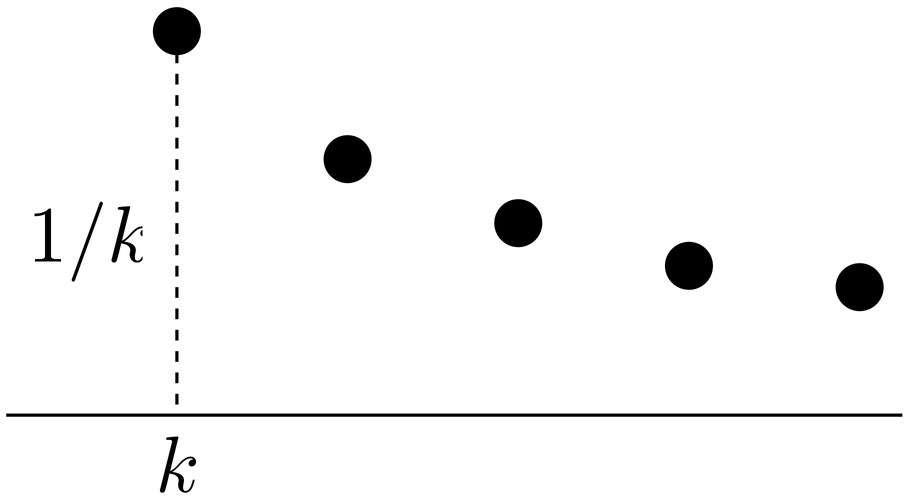
Def: Two metrics d, d' are *equivalent* if

$$\exists 0 < A, B < +\infty \text{ s.t. } A\,\mathrm{d}(\cdot,\cdot) \leq \mathrm{d}'(\cdot,\cdot) \leq B\,\mathrm{d}(\cdot,\cdot)$$

Proof:

$$S = \{D_u\}_{u \in \{0,1\}^{\mathbb{N}}}$$

Indeed, let $S' \subseteq S$ be a dense set and $\epsilon > 0$



The space of persistence diagrams

Q: What happens in general when one embeds PDs in Hilbert?

Def: Two metrics d, d' are *equivalent* if

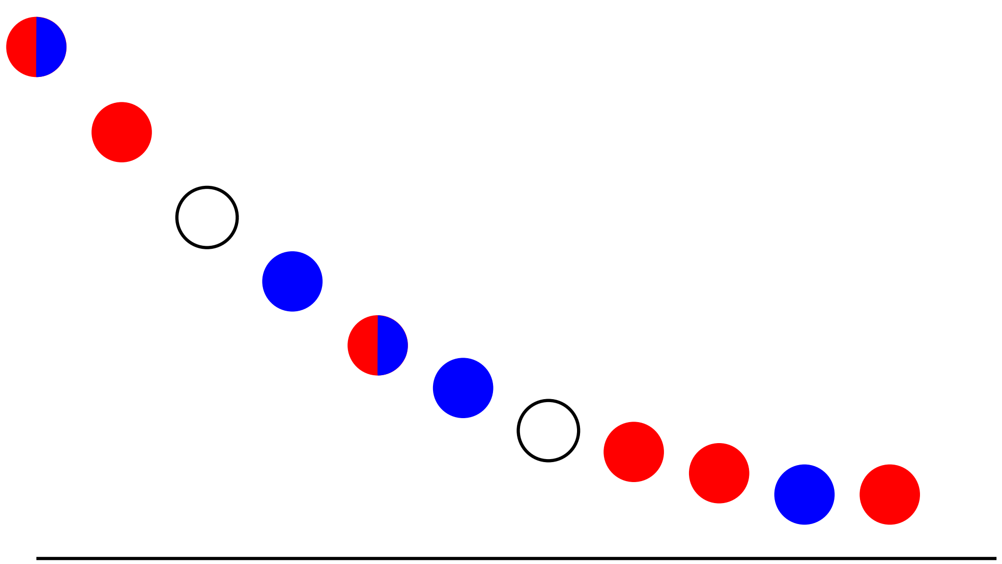
$$\exists 0 < A, B < +\infty \text{ s.t. } A\,\mathrm{d}(\cdot,\cdot) \leq \mathrm{d}'(\cdot,\cdot) \leq B\,\mathrm{d}(\cdot,\cdot)$$

Proof:

$$S = \{D_u\}_{u \in \{0,1\}^{\mathbb{N}}}$$

Indeed, let $S' \subseteq S$ be a dense set and $\epsilon > 0$

$$\forall D_u \in S,\ \exists D_{u'} \in S' : \mathrm{d}_1(D_u, D_{u'}) \leq \epsilon$$



The space of persistence diagrams

Q: What happens in general when one embeds PDs in Hilbert?

Def: Two metrics d, d' are *equivalent* if

$$\exists 0 < A, B < +\infty \text{ s.t. } A\,\mathrm{d}(\cdot,\cdot) \leq \mathrm{d}'(\cdot,\cdot) \leq B\,\mathrm{d}(\cdot,\cdot)$$

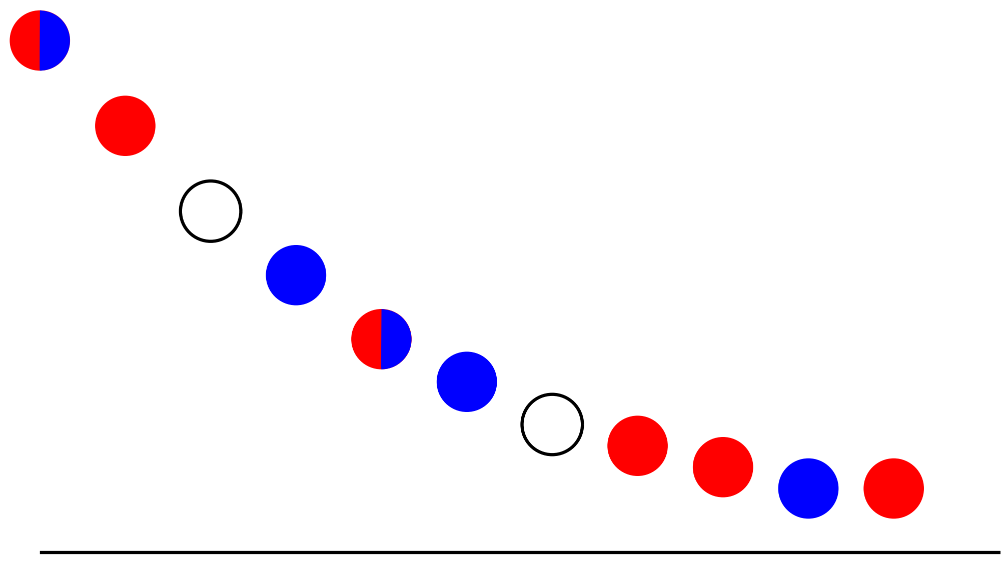
Proof:

$$S = \{D_u\}_{u \in \{0,1\}^{\mathbb{N}}}$$

Indeed, let $S' \subseteq S$ be a dense set and $\epsilon > 0$

$$\forall D_u \in S,\ \exists D_{u'} \in S' : \mathrm{d}_1(D_u, D_{u'}) \leq \epsilon$$

Supports of u' and u must differ on a finite number of terms only



The space of persistence diagrams

Q: What happens in general when one embeds PDs in Hilbert?

Def: Two metrics d, d' are *equivalent* if

$$\exists 0 < A, B < +\infty \text{ s.t. } A\,\mathrm{d}(\cdot,\cdot) \leq \mathrm{d}'(\cdot,\cdot) \leq B\,\mathrm{d}(\cdot,\cdot)$$

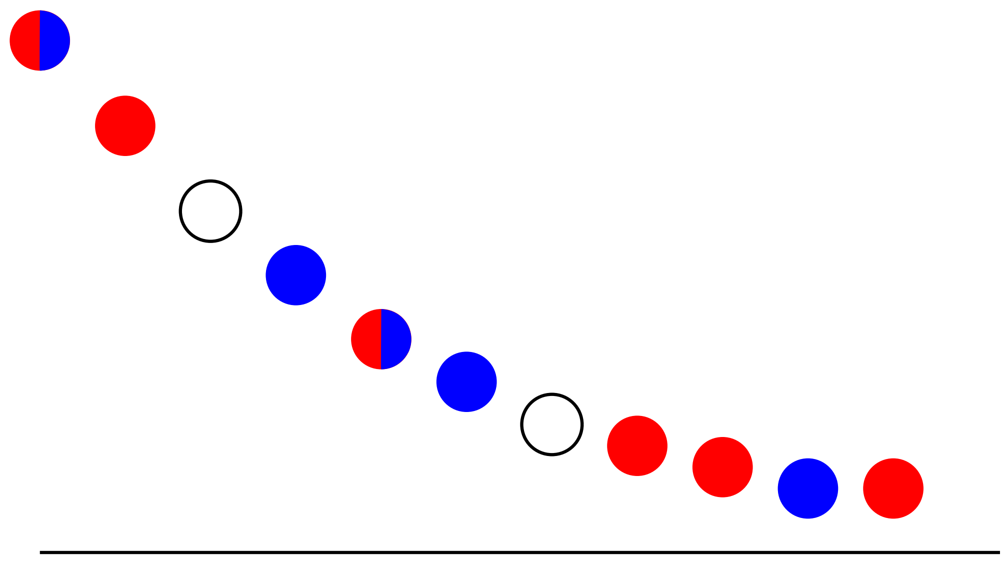
Proof:

$$S = \{D_u\}_{u\in\{0,1\}^{\mathbb{N}}}$$

Indeed, let $S' \subseteq S$ be a dense set and $\epsilon > 0$

$$\forall D_u \in S,\ \exists D_{u'} \in S' : \mathrm{d}_1(D_u, D_{u'}) \leq \epsilon$$

Supports of u' and u must differ on a finite number of terms only



$\Rightarrow$ card$(S') \geq$ card$(S/\sim)$

where $D_u \sim D_v \Leftrightarrow \text{supp}(u) \bigtriangleup \text{supp}(v) < \infty$

The space of persistence diagrams

Q: What happens in general when one embeds PDs in Hilbert?

Def: Two metrics d, d' are *equivalent* if

$$\exists 0 < A, B < +\infty \text{ s.t. } A\,\mathrm{d}(\cdot,\cdot) \leq \mathrm{d}'(\cdot,\cdot) \leq B\,\mathrm{d}(\cdot,\cdot)$$

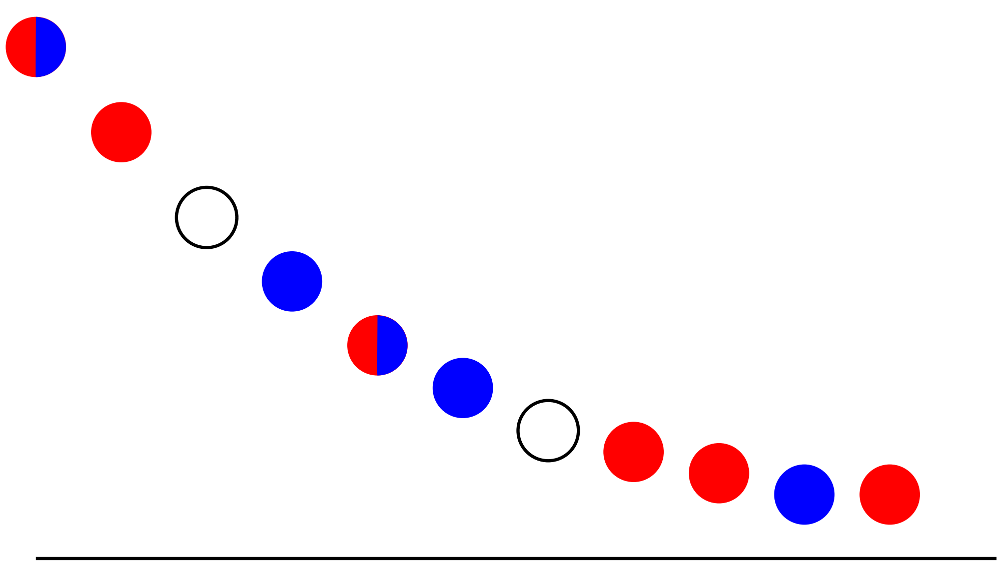
Proof:

$$S = \{D_u\}_{u \in \{0,1\}^{\mathbb{N}}}$$

Indeed, let $S' \subseteq S$ be a dense set and $\epsilon > 0$

$$\forall D_u \in S,\ \exists D_{u'} \in S' : \mathrm{d}_1(D_u, D_{u'}) \leq \epsilon$$

Supports of u' and u must differ on a finite number of terms only



$\Rightarrow$ card$(S') \geq$ card$(S/\sim)$ uncountable!

where $D_u \sim D_v \Leftrightarrow \mathrm{supp}(u) \triangle \mathrm{supp}(v) < \infty$

The space of persistence diagrams

Q: What happens in general when one embeds PDs in Hilbert?

Def: Two metrics d, d' are *equivalent* if

$$\exists 0 < A, B < +\infty \text{ s.t. } A\,\mathrm{d}(\cdot,\cdot) \leq \mathrm{d}'(\cdot,\cdot) \leq B\,\mathrm{d}(\cdot,\cdot)$$

Example: persistence image

$$\Phi(D) = \sum_{p \in D} w(p) \cdot \exp\left(-\frac{\|\cdot - p\|_2^2}{2\sigma^2}\right)$$

where $w((x,y)) = \arctan\left(C|y-x|^\alpha\right)$ with $C, \alpha > 0$

The space of persistence diagrams

Q: What happens in general when one embeds PDs in Hilbert?

Def: Two metrics d, d' are *equivalent* if

$$\exists 0 < A, B < +\infty \text{ s.t. } A\,\mathrm{d}(\cdot,\cdot) \leq \mathrm{d}'(\cdot,\cdot) \leq B\,\mathrm{d}(\cdot,\cdot)$$

Example: persistence image

$$\Phi(D) = \sum_{p\in D} w(p) \cdot \exp\left(-\frac{\|\cdot - p\|_2^2}{2\sigma^2}\right)$$

where $w((x,y)) = \arctan\left(C|y-x|^\alpha\right)$ with $C, \alpha > 0$

If $\alpha \geq 2$, S is in the domain of Φ

and metric equivalence is impossible.

The space of persistence diagrams

Q: What happens in general when one embeds PDs in Hilbert?

Def: Two metrics d, d' are *equivalent* if

$$\exists 0 < A, B < +\infty \text{ s.t. } A\,\mathrm{d}(\cdot,\cdot) \leq \mathrm{d}'(\cdot,\cdot) \leq B\,\mathrm{d}(\cdot,\cdot)$$

Proof:

(i) is a little more tricky

Def: Let (X, d) be a metric space. Given a subset $E \subset X$ and $r > 0$, let $N_r(E)$ be the least number of open balls of radius $\leq r$ that can cover E. The *Assouad dimension* of (X, d) is:

$$\dim_A(X,d) = \inf\{\alpha : \exists C \text{ s.t. } \sup_x N_{\beta r}(B(x,r)) \leq C\beta^{-\alpha},\ 0 < \beta \leq 1\}$$

The space of persistence diagrams

Q: What happens in general when one embeds PDs in Hilbert?

Def: Two metrics d, d' are *equivalent* if

$$\exists 0 < A, B < +\infty \text{ s.t. } A\,\mathrm{d}(\cdot,\cdot) \leq \mathrm{d}'(\cdot,\cdot) \leq B\,\mathrm{d}(\cdot,\cdot)$$

Proof:

(i) is a little more tricky

Def: Let (X, d) be a metric space. Given a subset $E \subset X$ and $r > 0$, let $N_r(E)$ be the least number of open balls of radius $\leq r$ that can cover E. The *Assouad dimension* of (X, d) is:

$$\dim_A(X, d) = \inf\{\alpha : \exists C \text{ s.t. } \sup_x N_{\beta r}(B(x, r)) \leq C\beta^{-\alpha},\ 0 < \beta \leq 1\}$$

$\dim_A$ is preserved for equivalent metrics

$\dim_A(\mathcal{D}, d_p) = +\infty$ whereas $\dim_A(\mathbb{R}^d) = d$

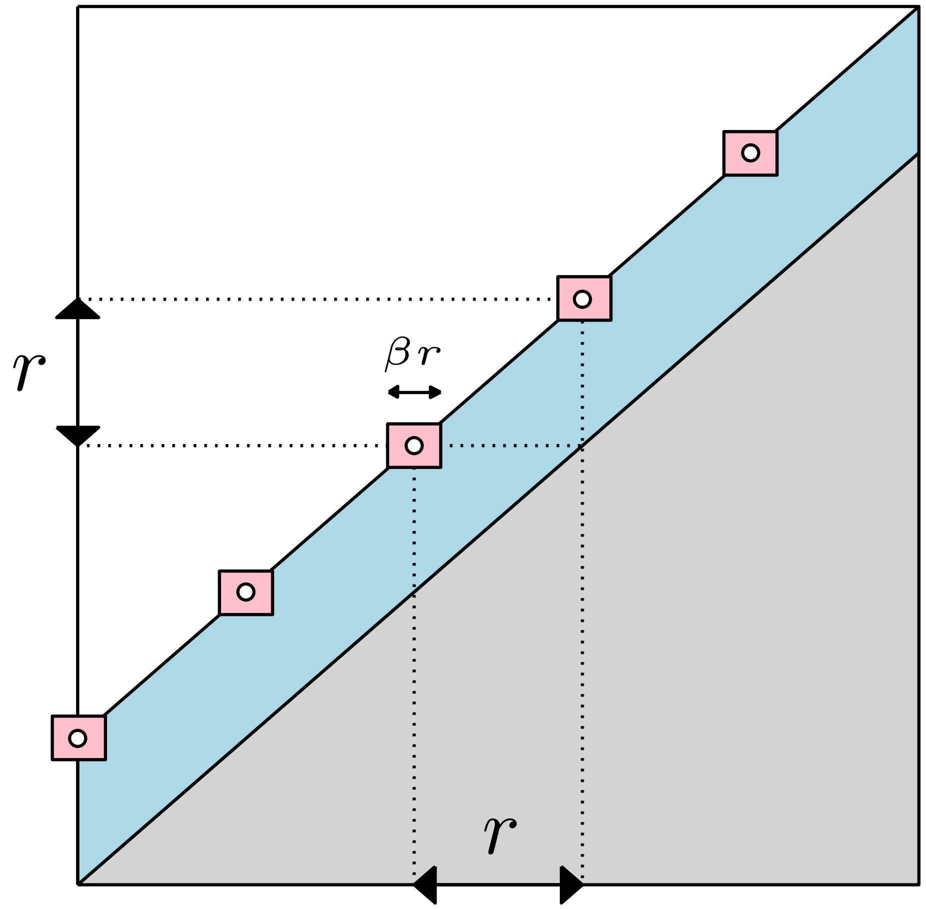
The space of persistence diagrams

Q: What happens in general when one embeds PDs in Hilbert?

Def: Two metrics d, d' are *equivalent* if

$$\exists 0 < A, B < +\infty \text{ s.t. } A\,\mathrm{d}(\cdot,\cdot) \leq \mathrm{d}'(\cdot,\cdot) \leq B\,\mathrm{d}(\cdot,\cdot)$$

Proof:



Idea: Consider the ball of radius r around the empty diagram and diagrams with single points at distance r from Δ and from each other

The number of such diagrams increases to $+\infty$ as β goes to 0

$\dim_A$ is preserved for equivalent metrics

$\dim_A(\mathcal{D}, d_p) = +\infty$ whereas $\dim_A(\mathbb{R}^d) = d$

The space of persistence diagrams

Illustrations:

We generate diagrams by uniformly sampling into the upper unit half-square