

Topological Data Analysis, PSL Week 2025

Persistent homology for ML

Frédéric Chazal

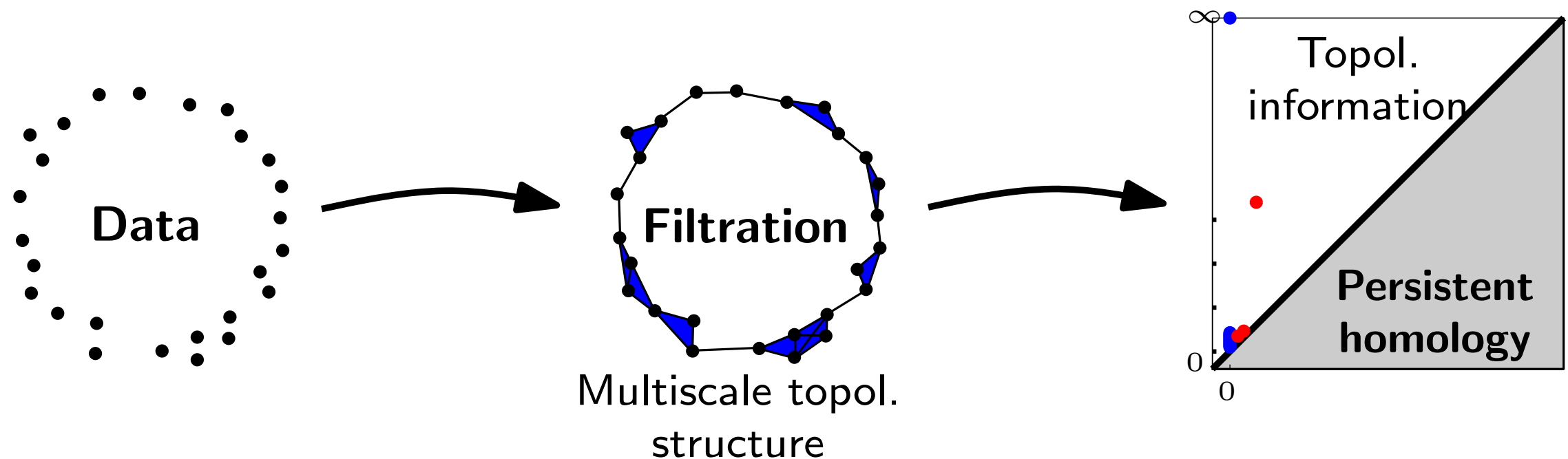
DataShape team

Inria & Laboratoire de Mathématiques d'Orsay Institut

DATAIA Université Paris-Saclay

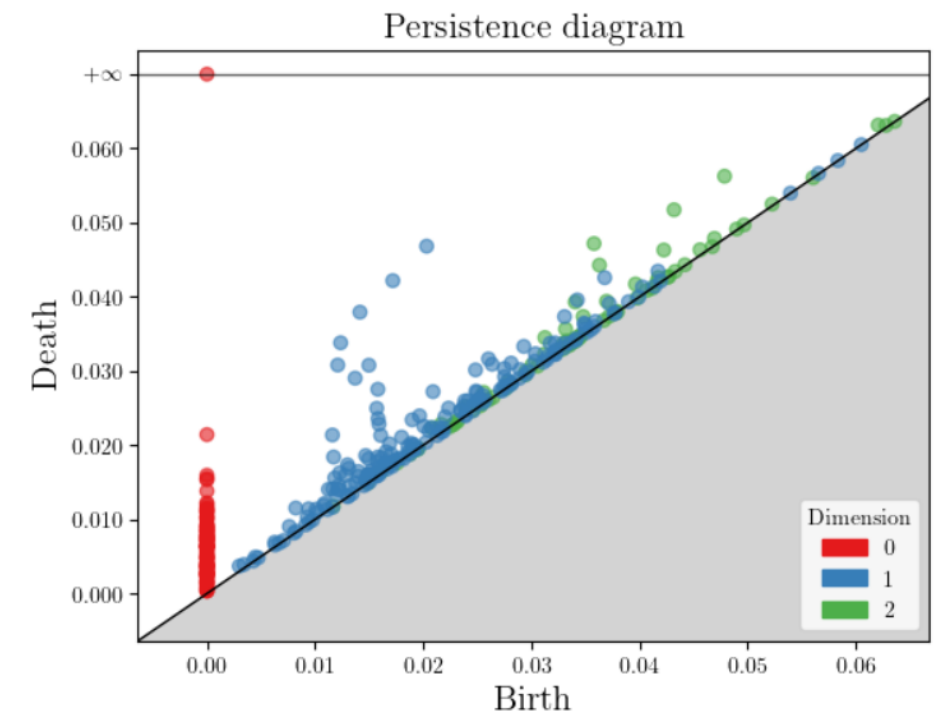


Introduction and motivation

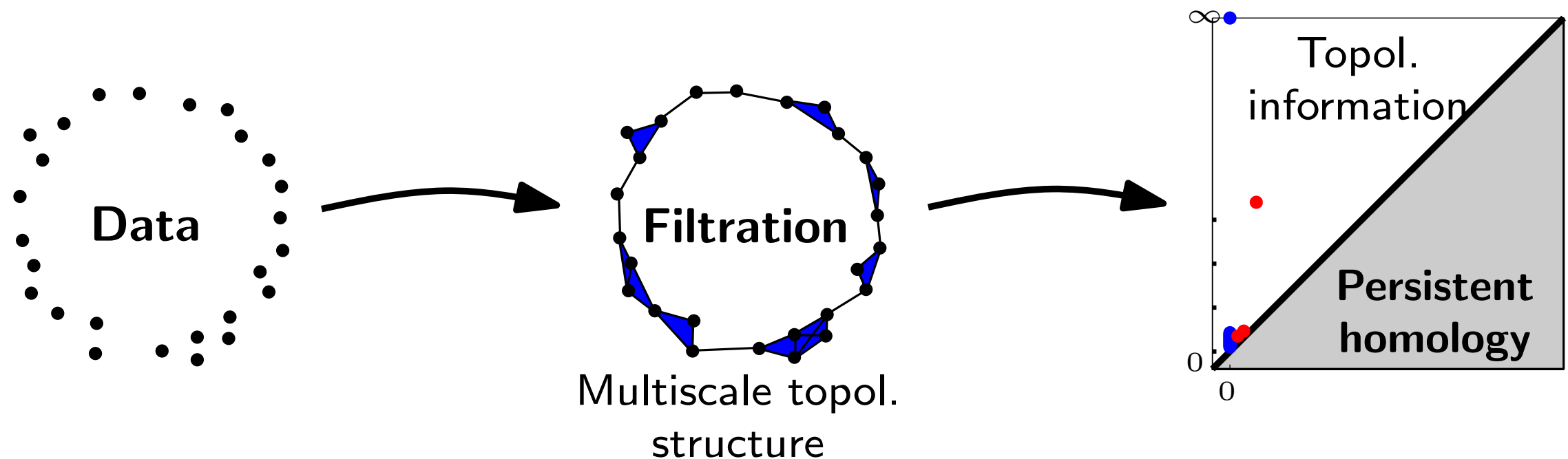


Question: How to take advantage of the topological information encoded in persistence diagrams for further data analysis or Machine Learning tasks?

- Persistence diagrams are not well-suited for direct use in ML algorithm.
- Relevant topological information (for a given task) may not be easy to extract from persistence diagrams.



Introduction and motivation



Different uses of persistence diagrams in ML

1. Persistent homology to improve ML algorithms/models (through a better understanding of the topological structure of data)
2. Persistent homology as a feature engineering tool (vectorization of persistence diagrams)
3. Persistent homology to monitor ML models (an active research domain - not addressed in this course).

Persistent homology to improve ML algorithms/models

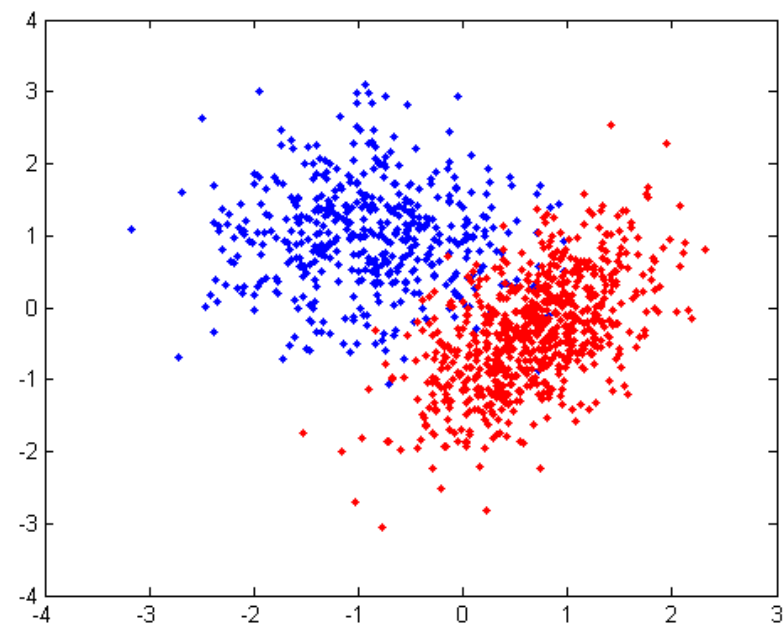
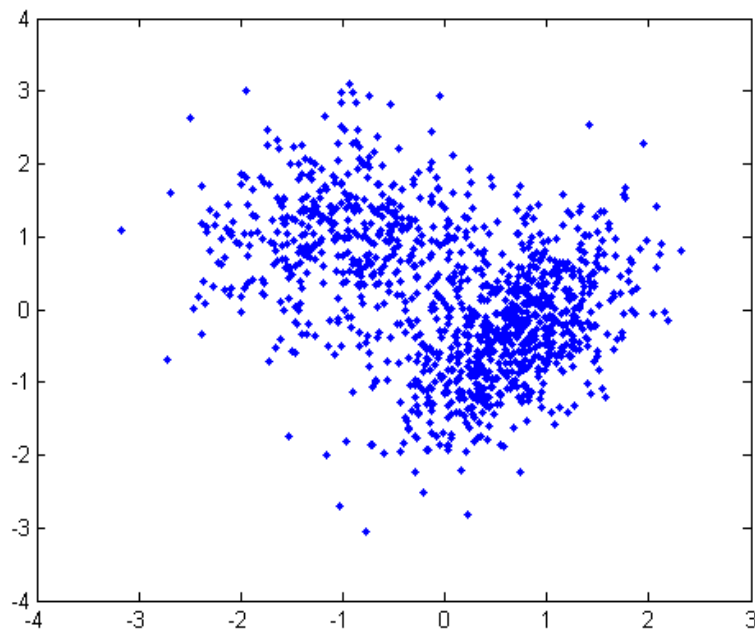
The example of clustering

An active research area

Introduction

Clustering: A partition of data into groups of similar observations. The observations in each group (cluster) are similar to each other and dissimilar to observations from other groups.


Input: a finite set of observations: point cloud embedded in an Euclidean space (with coordinates) or a more general metric space (pairwise distance/similarity) matrix.



Goal: partition the data into a relevant family of subsets (clusters).

Introduction

Clustering: A partition of data into groups of similar observations. The observations in each group (cluster) are similar to each other and dissimilar to observations from other groups.



Not a single/universal notion of cluster.

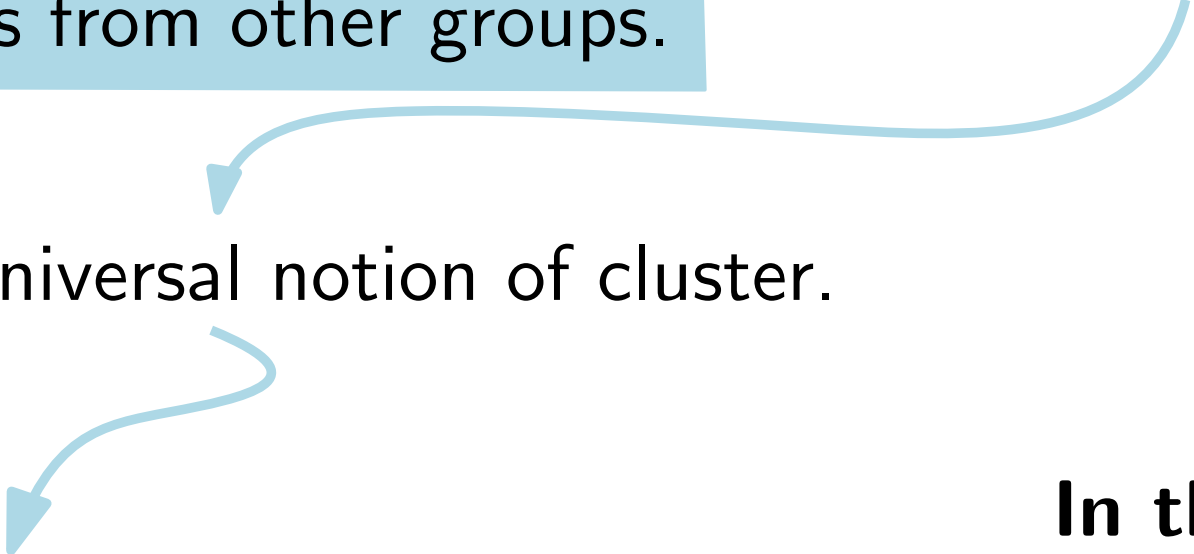


A wealth of approaches:

- Variational
- Spectral
- Density based
- Hierarchical
- etc...

Introduction

Clustering: A partition of data into groups of similar observations. The observations in each group (cluster) are similar to each other and dissimilar to observations from other groups.



```
graph TD; A["Clustering: A partition of data into groups of similar observations. The observations in each group (cluster) are similar to each other and dissimilar to observations from other groups."] --> B["Not a single/universal notion of cluster."]; B --> C["A wealth of approaches:"]; C --> D["• Variational<br/>• Spectral<br/>• Density based<br/>• Hierarchical<br/>• etc..."];
```

Not a single/universal notion of cluster.

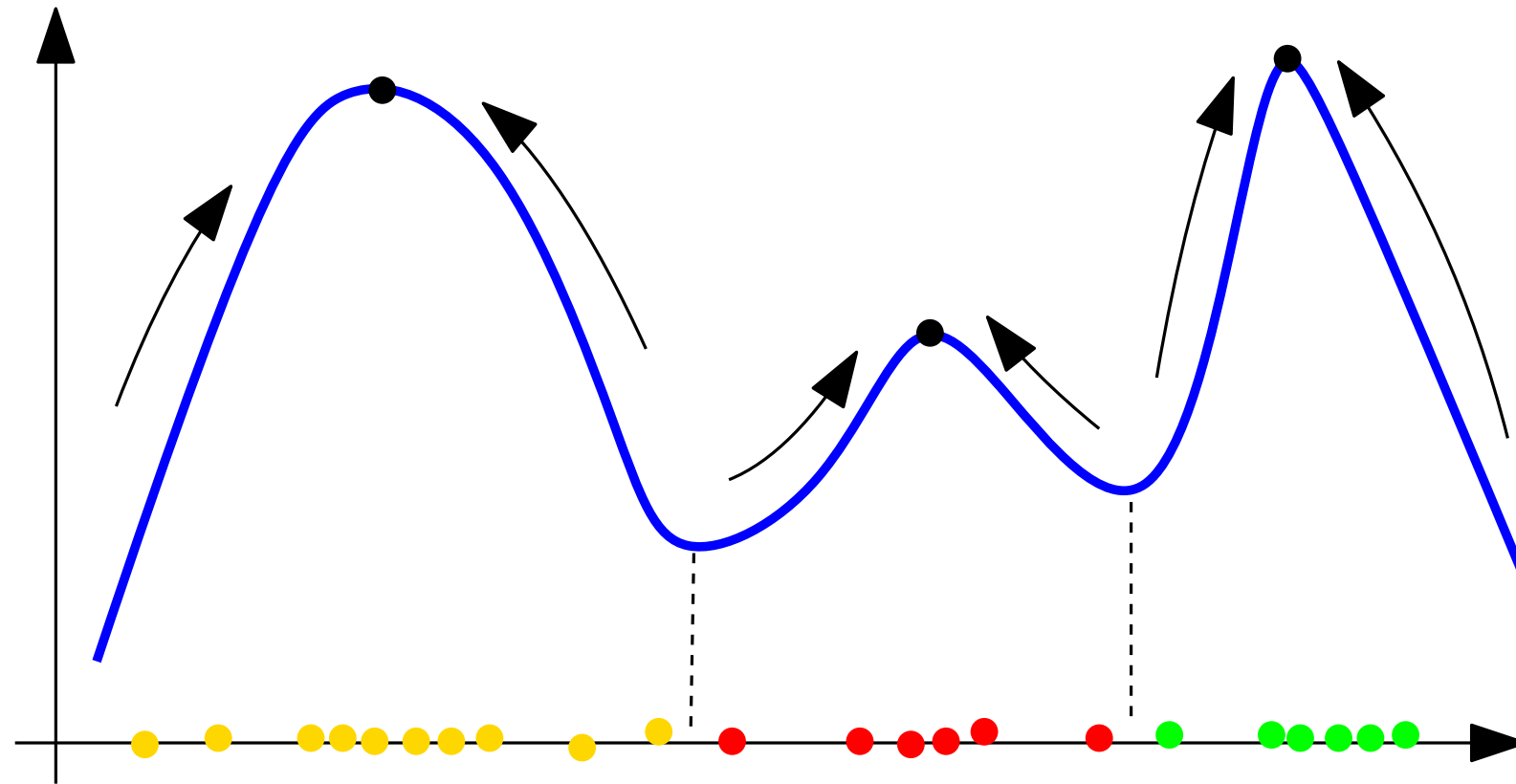
A wealth of approaches:

- Variational
- Spectral
- Density based
- Hierarchical
- etc...

In this lecture:

a persistence based mode
seeking algorithm:
ToMATo

Mode seeking clustering

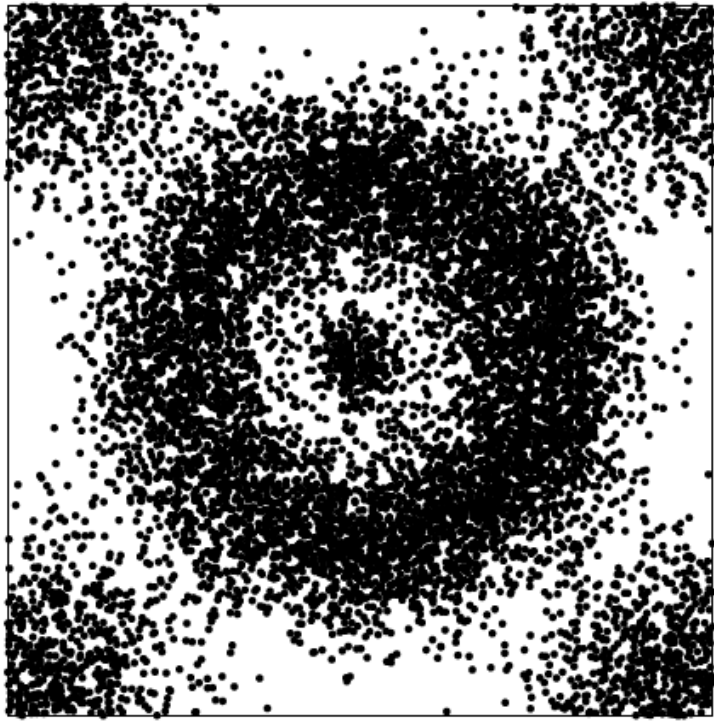


- Data points are sampled according to some (unknown) probability density.
- Clusters = the basins of attractions of the density

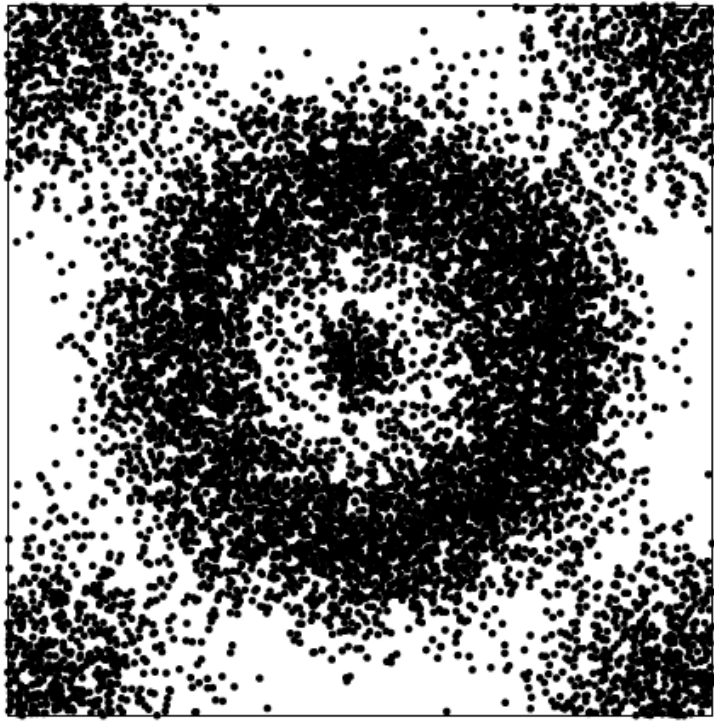
Two approaches:

- **Iterative**, such as, e.g., Mean Shift [Comaniciu et al, IEEE Trans. on Pattern Analysis and Machine Intelligence, 2002].
- **Graph-based**, such as, e.g., [Koontz et al, IEEE Trans. on Computers 1976].

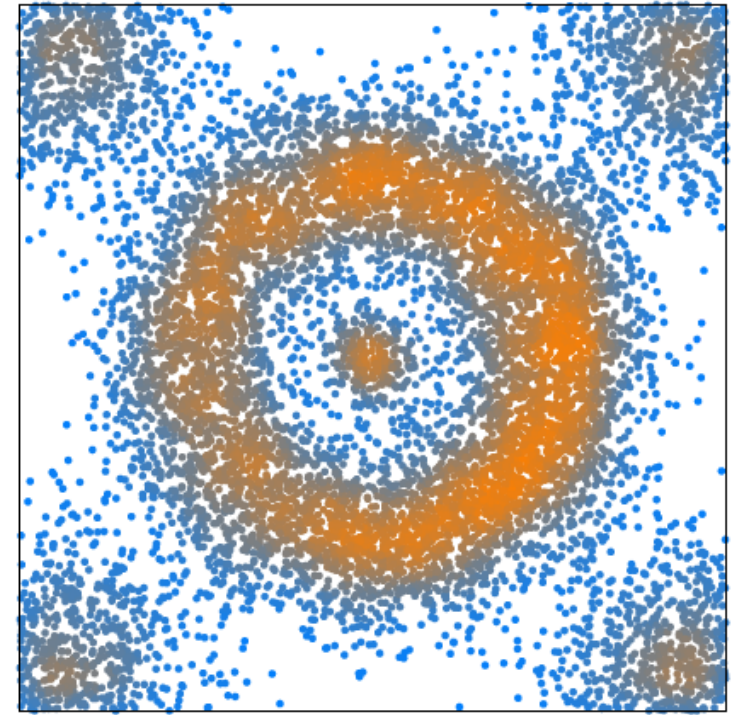
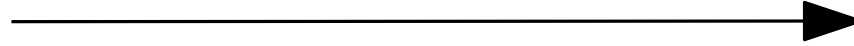
The Koonz, Narendra and Fukunaga algorithm (1976)



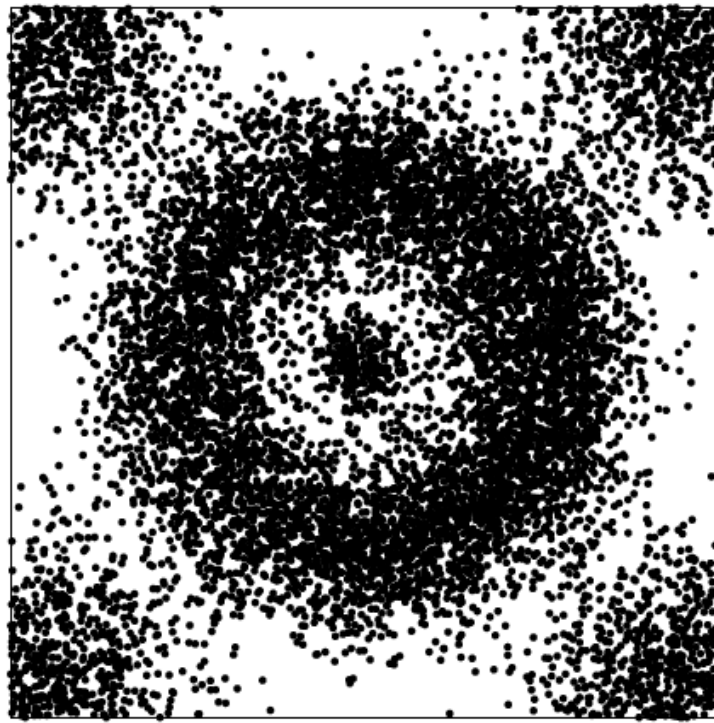
The Koonz, Narendra and Fukunaga algorithm (1976)



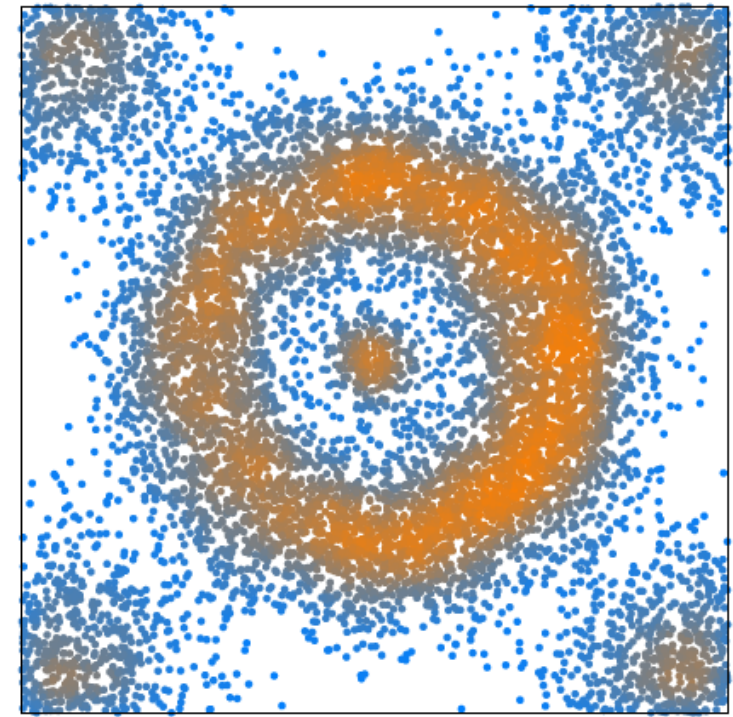
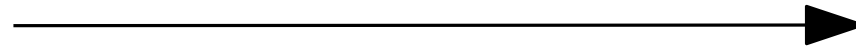
Density estimation



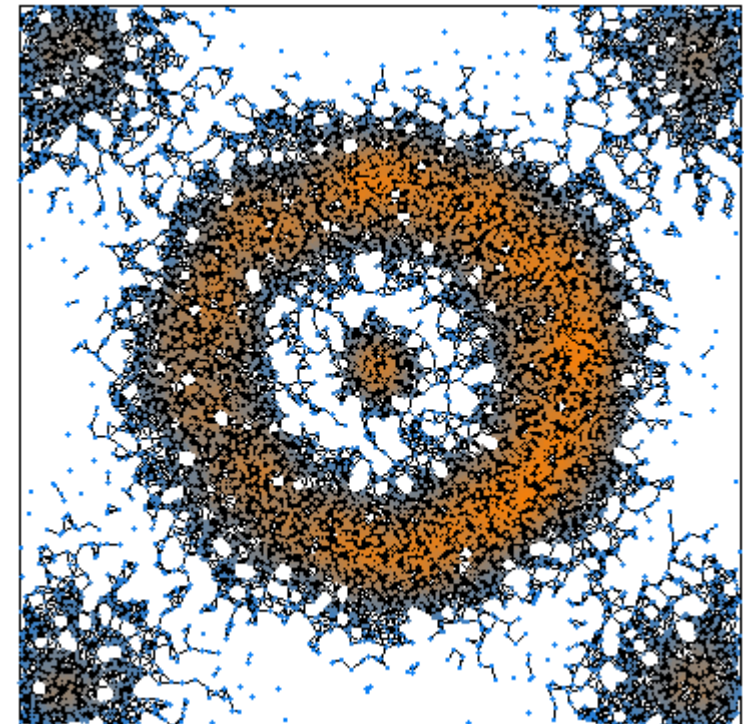
The Koonz, Narendra and Fukunaga algorithm (1976)



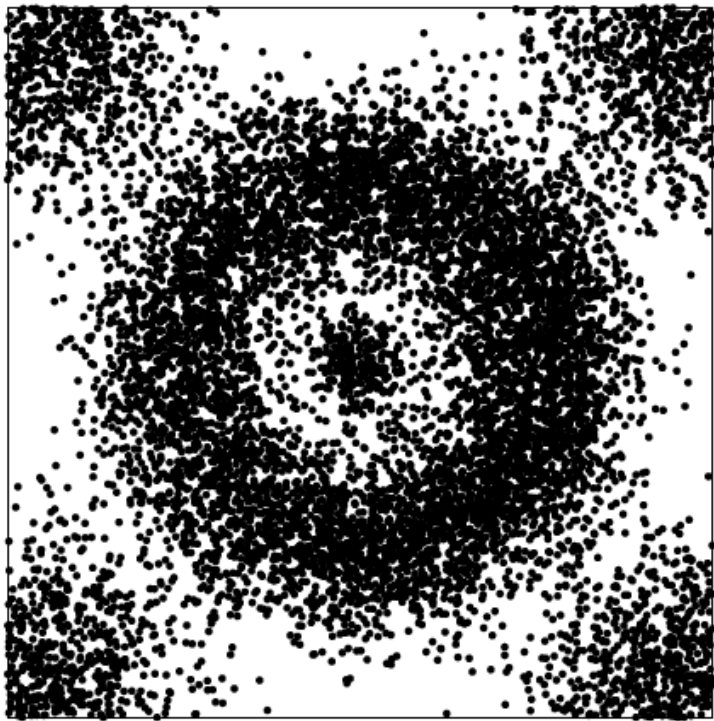
Density estimation



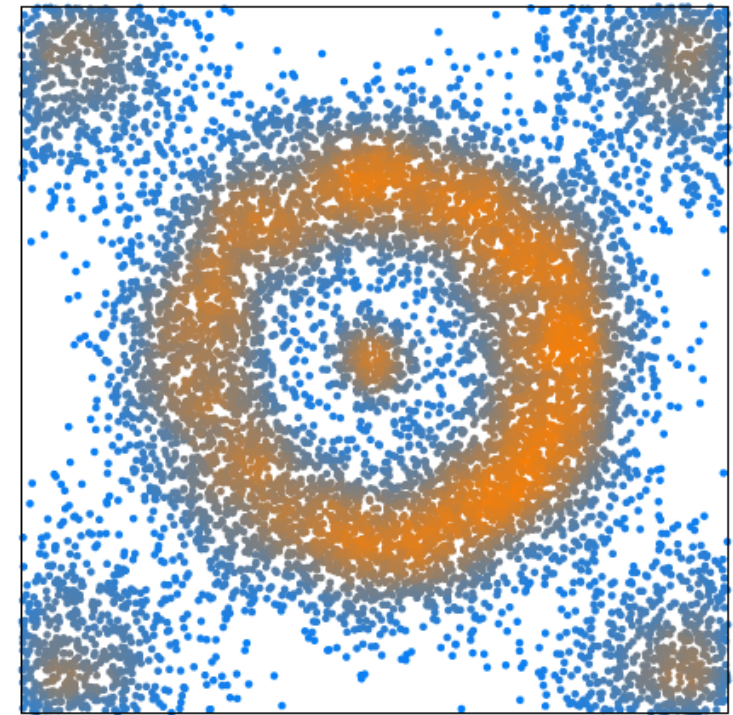
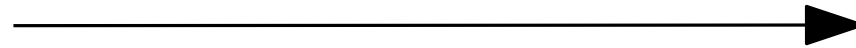
Neighborhood
graph



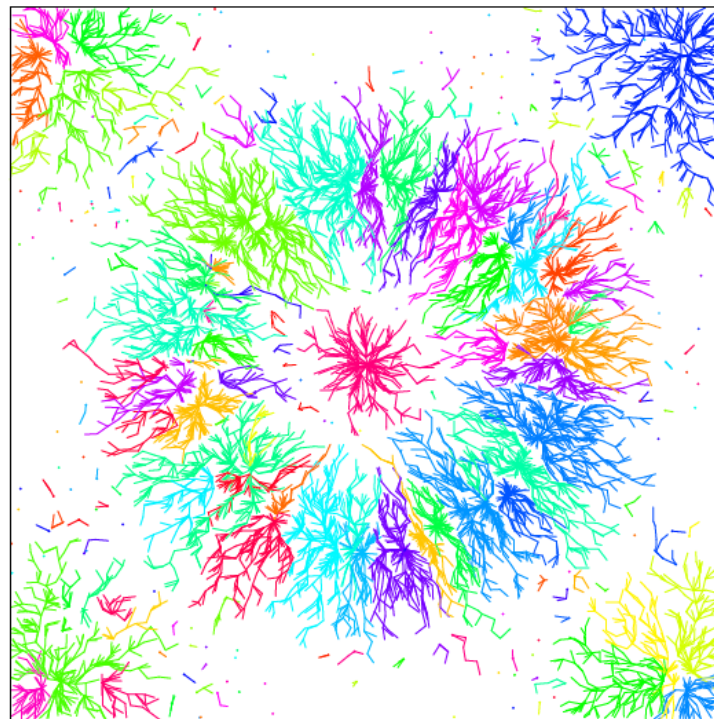
The Koonz, Narendra and Fukunaga algorithm (1976)



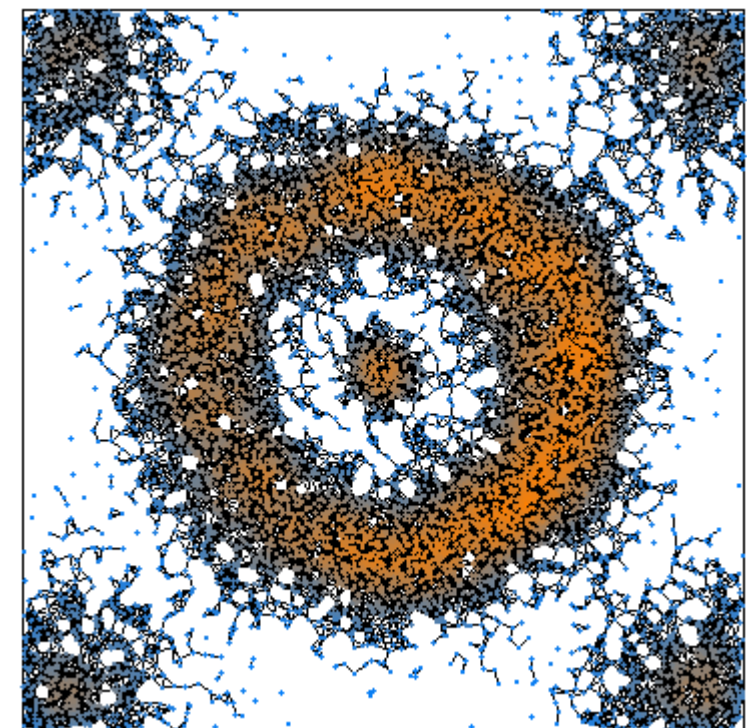
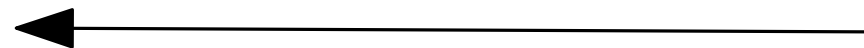
Density estimation



Neighborhood
graph



Discrete approximation of the gradient; for each vertex v , a gradient edge is selected among the edges adjacent to v .



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 $\{1, 2, \dots, n\}$ in decreasing order: $\hat{f}(1) \geq \hat{f}(2) \geq \dots \geq \hat{f}(n)$;

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;

$r(e) \leftarrow i$ // $r(e)$ stores the root vertex associated with the entry e

else

$g(i) \leftarrow \operatorname{argmax}_{j \in N} \hat{f}(j)$ // $g(i)$ stores the approximate gradient at vertex i

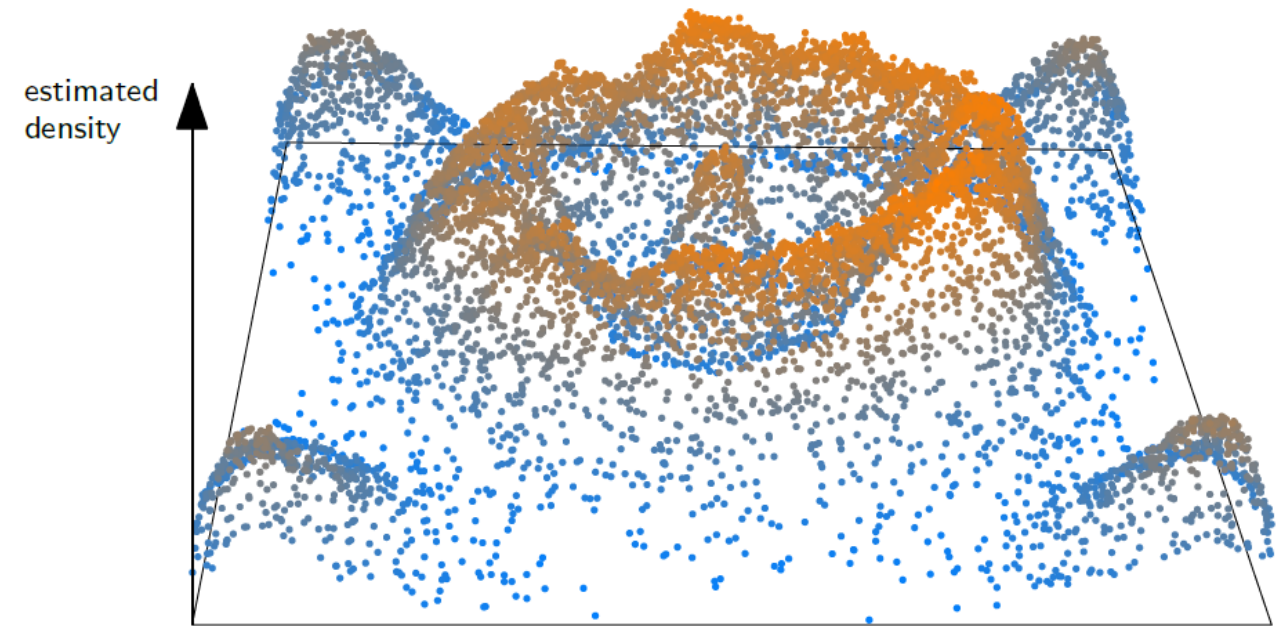
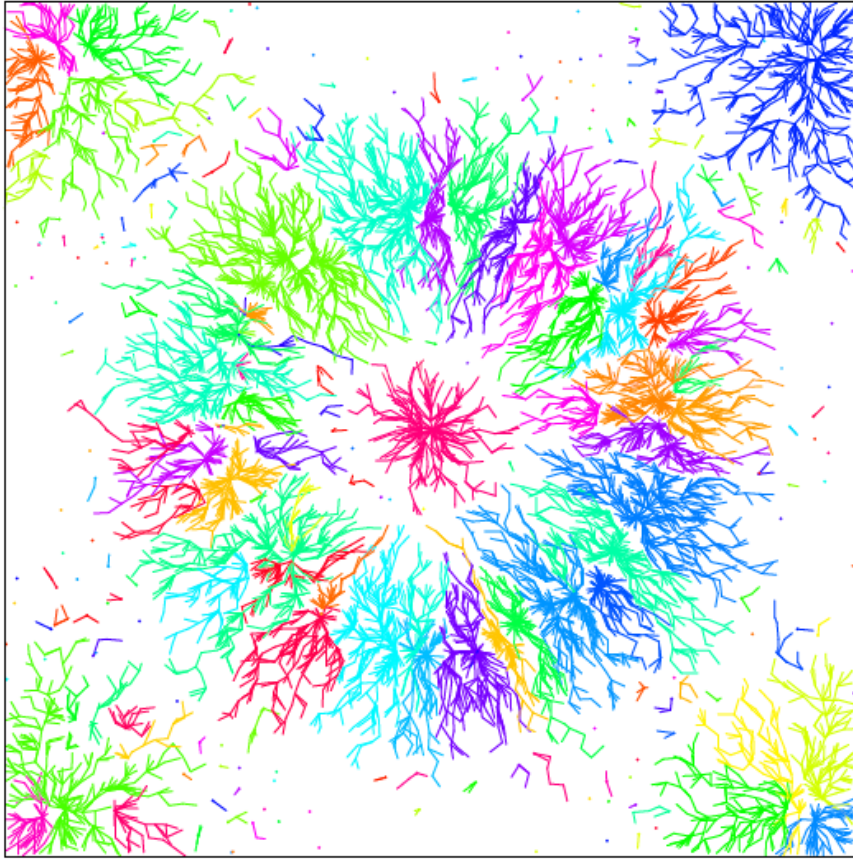
$e_i \leftarrow \mathcal{U}.\text{find}(g(i))$;

Attach vertex i to the entry e_i ;

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

The Koonz, Narendra and Fukunaga algorithm (1976)

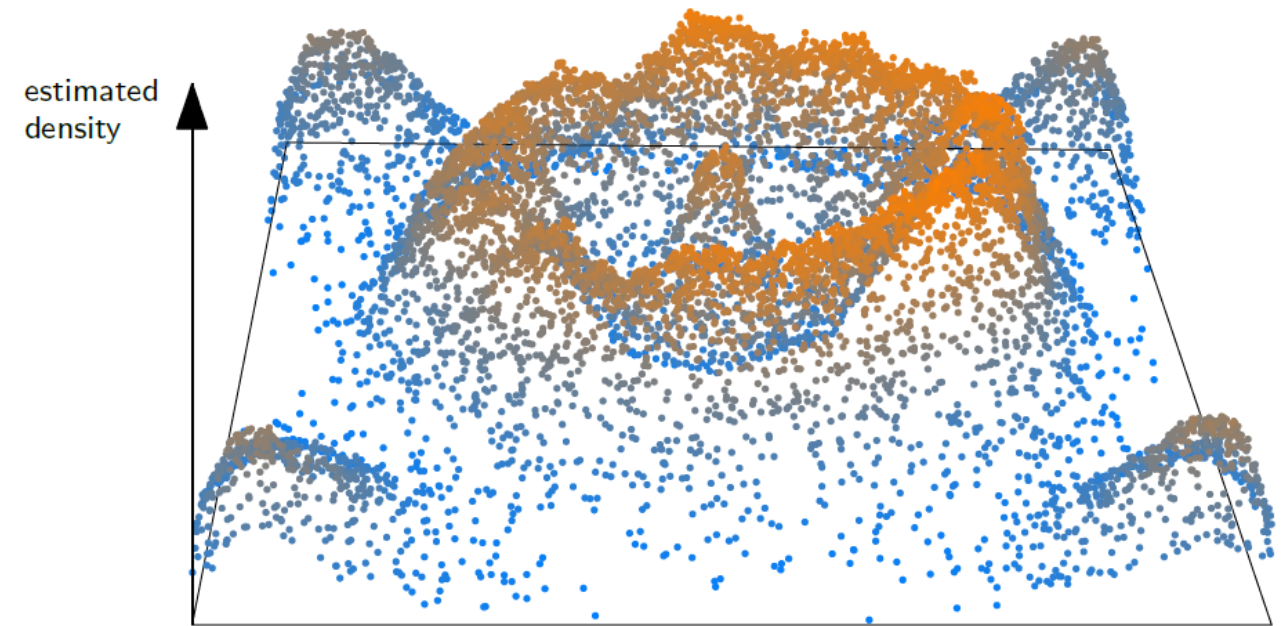
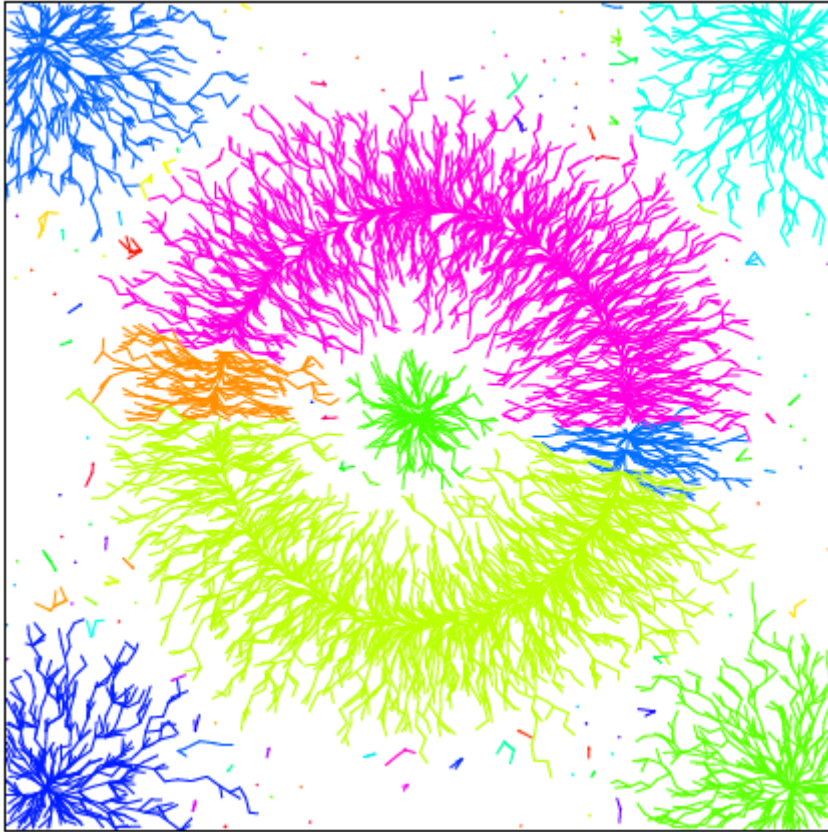
Drawbacks:



- As many clusters as local maxima of the density estimate \rightarrow sensitivity to noise!

The Koonz, Narendra and Fukunaga algorithm (1976)

Drawbacks:



- As many clusters as local maxima of the density estimate \rightarrow sensitivity to noise!
- The choice of the neighborhood graph may result in wide changes in the output.

The Koonz, Narendra and Fukunaga algorithm (1976)

Drawbacks:

- As many clusters as local maxima of the density estimate → sensitivity to noise!
- The choice of the neighborhood graph may result in wide changes in the output.

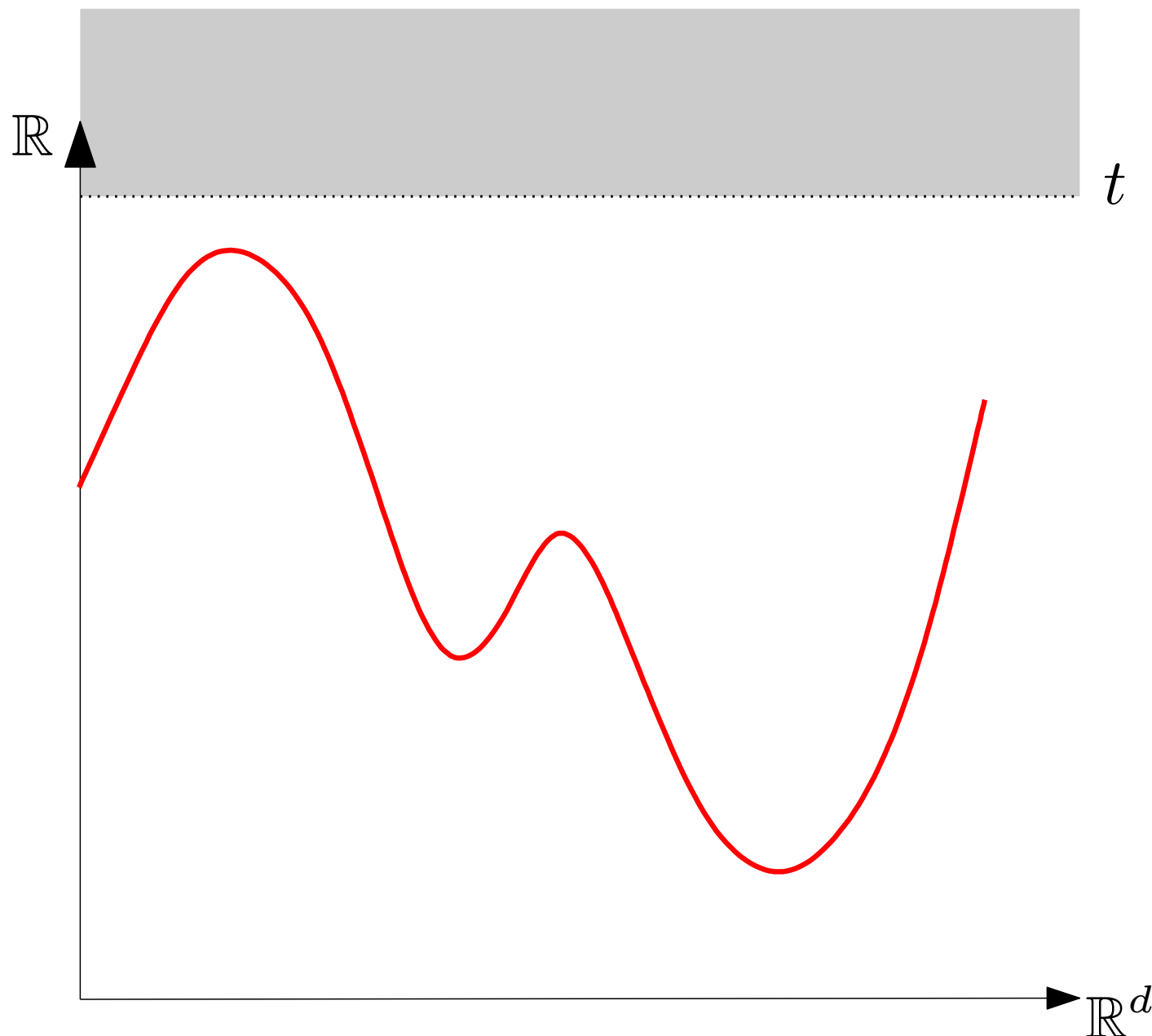
Approaches to overcome these issues:

- Smooth out the density estimate (e.g. mean-shift)... But pb of choice of a smoothing param.
- Merge clusters: various way to do that. In the following: **use persistent homology!**

Superlevel set persistence

Given a probability density f :

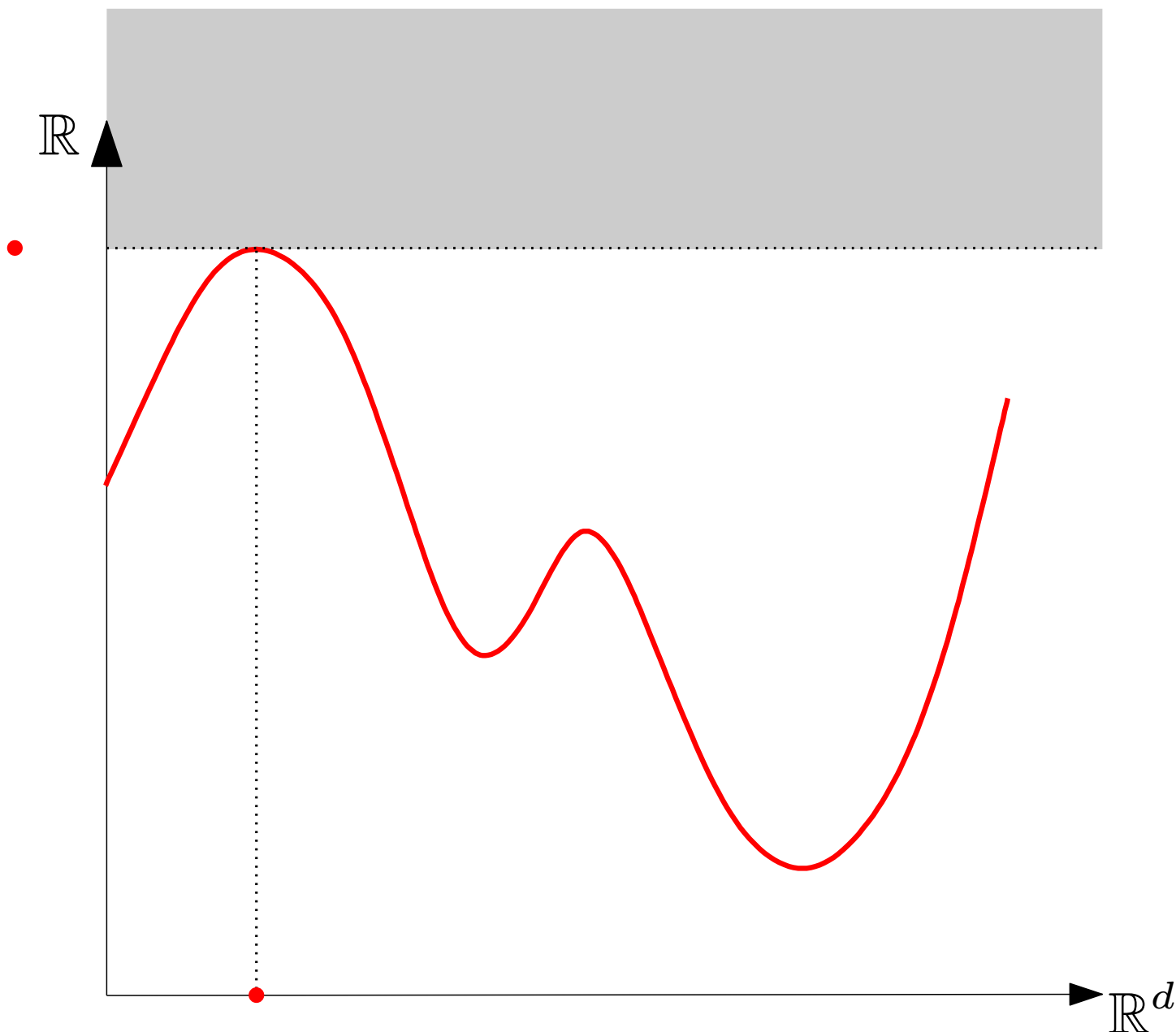
- Consider the superlevel-sets filtration $f^{-1}([t, +\infty))$ for t from $+\infty$ to $-\infty$, instead of the sublevel-sets filtration.
- Persistence is defined in the same way



Superlevel set persistence

Given a probability density f :

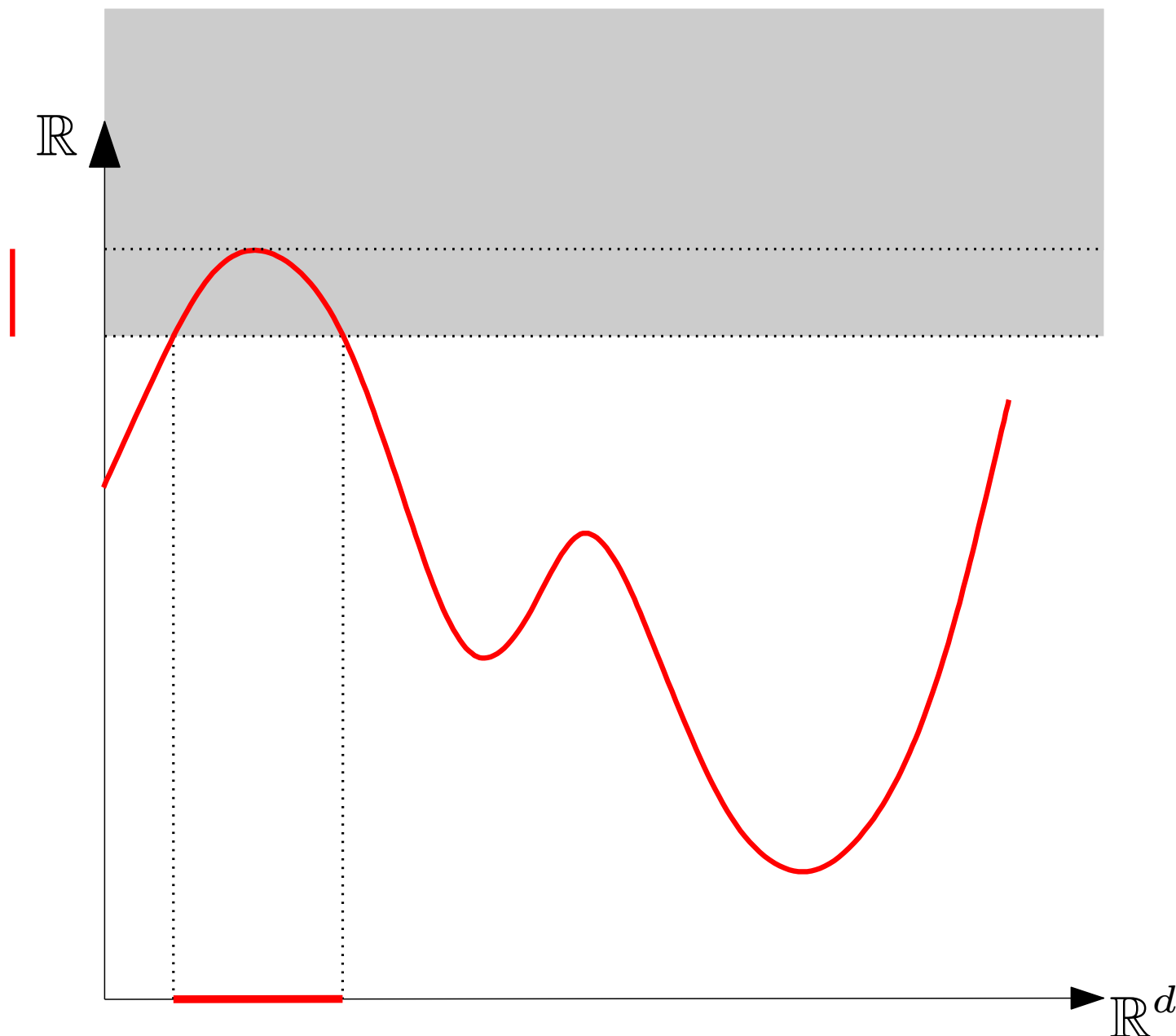
- Consider the superlevel-sets filtration $f^{-1}([t, +\infty))$ for t from $+\infty$ to $-\infty$, instead of the sublevel-sets filtration.
- Persistence is defined in the same way



Superlevel set persistence

Given a probability density f :

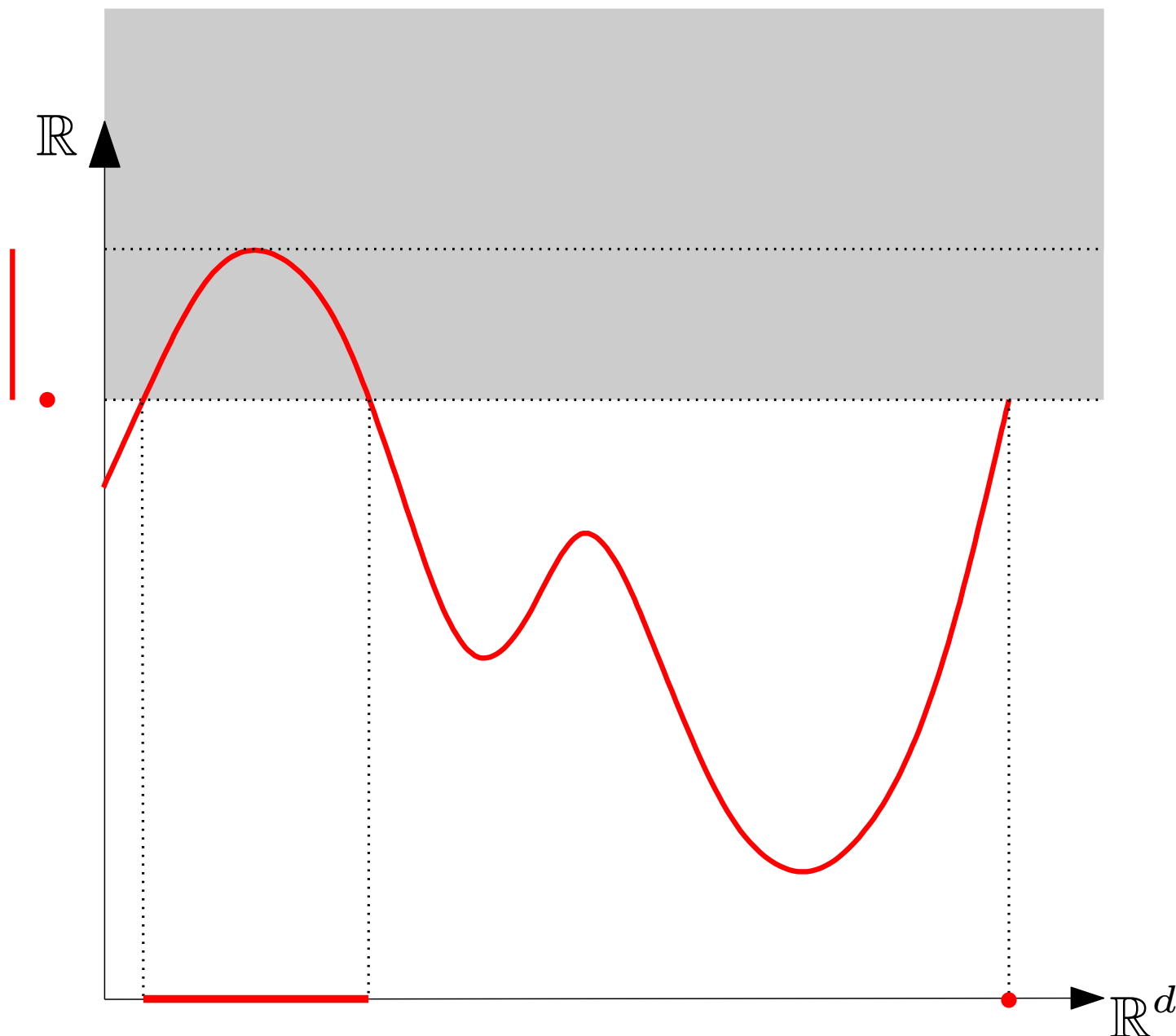
- Consider the superlevel-sets filtration $f^{-1}([t, +\infty))$ for t from $+\infty$ to $-\infty$, instead of the sublevel-sets filtration.
- Persistence is defined in the same way



Superlevel set persistence

Given a probability density f :

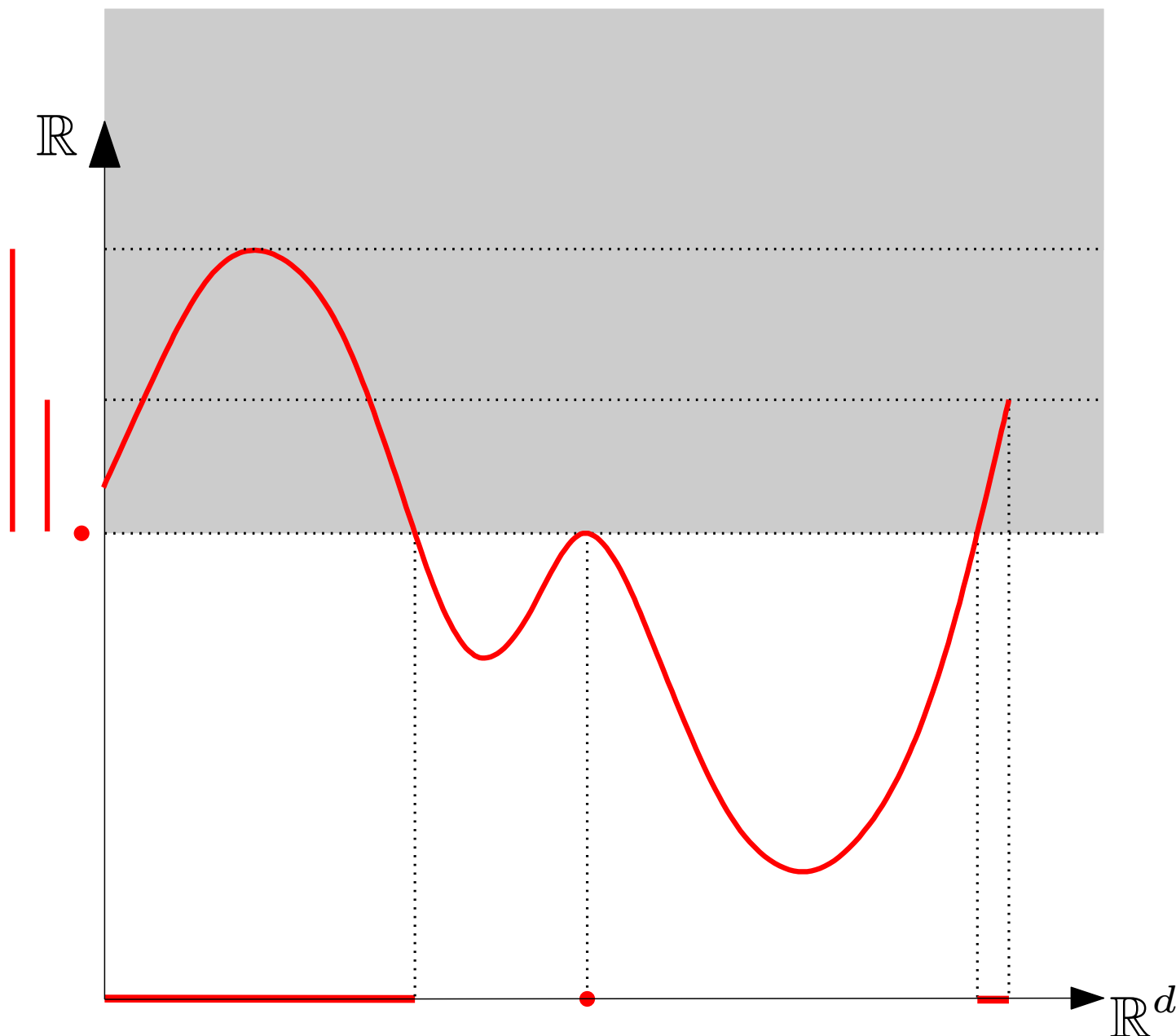
- Consider the superlevel-sets filtration $f^{-1}([t, +\infty))$ for t from $+\infty$ to $-\infty$, instead of the sublevel-sets filtration.
- Persistence is defined in the same way



Superlevel set persistence

Given a probability density f :

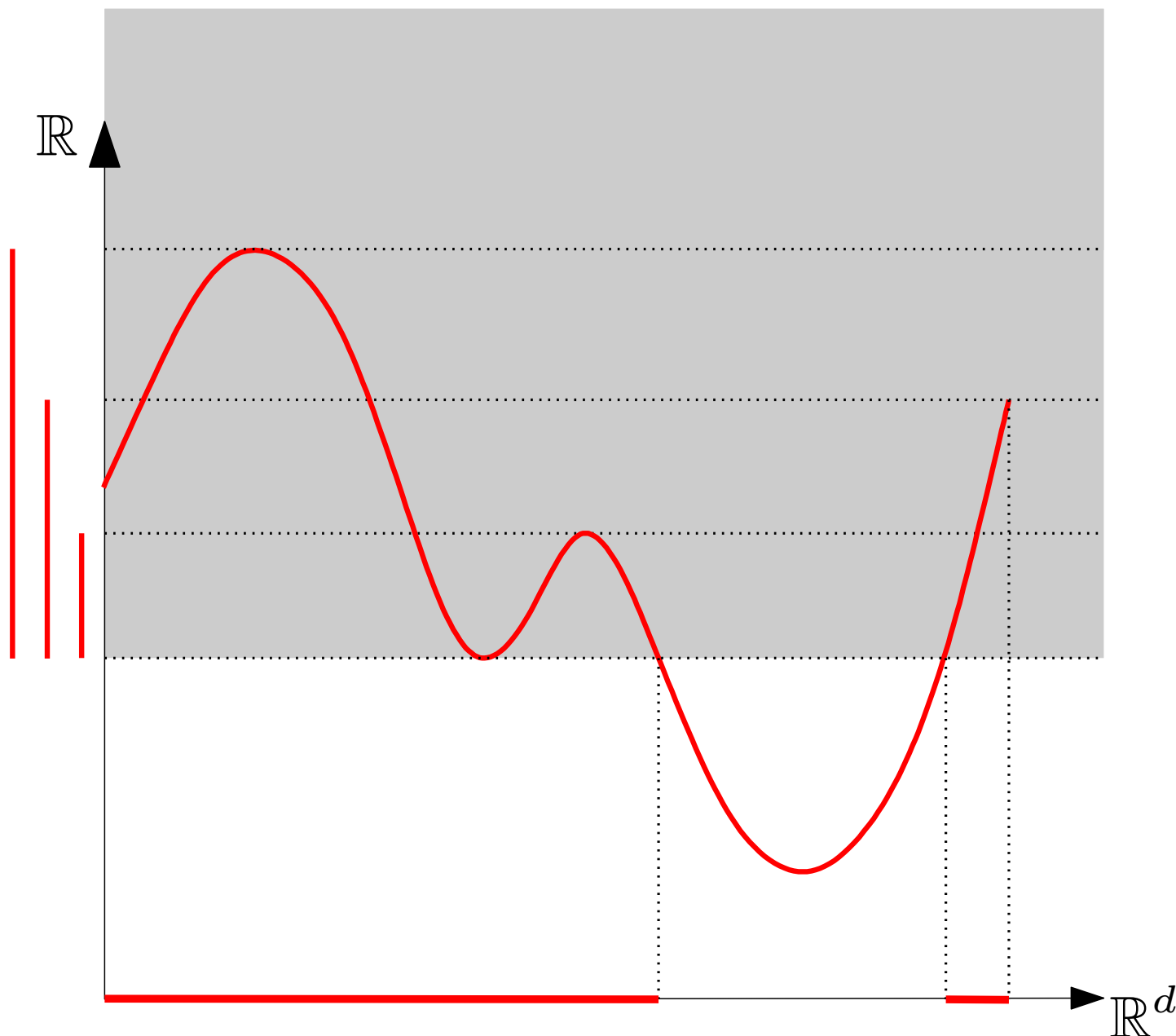
- Consider the superlevel-sets filtration $f^{-1}([t, +\infty))$ for t from $+\infty$ to $-\infty$, instead of the sublevel-sets filtration.
- Persistence is defined in the same way



Superlevel set persistence

Given a probability density f :

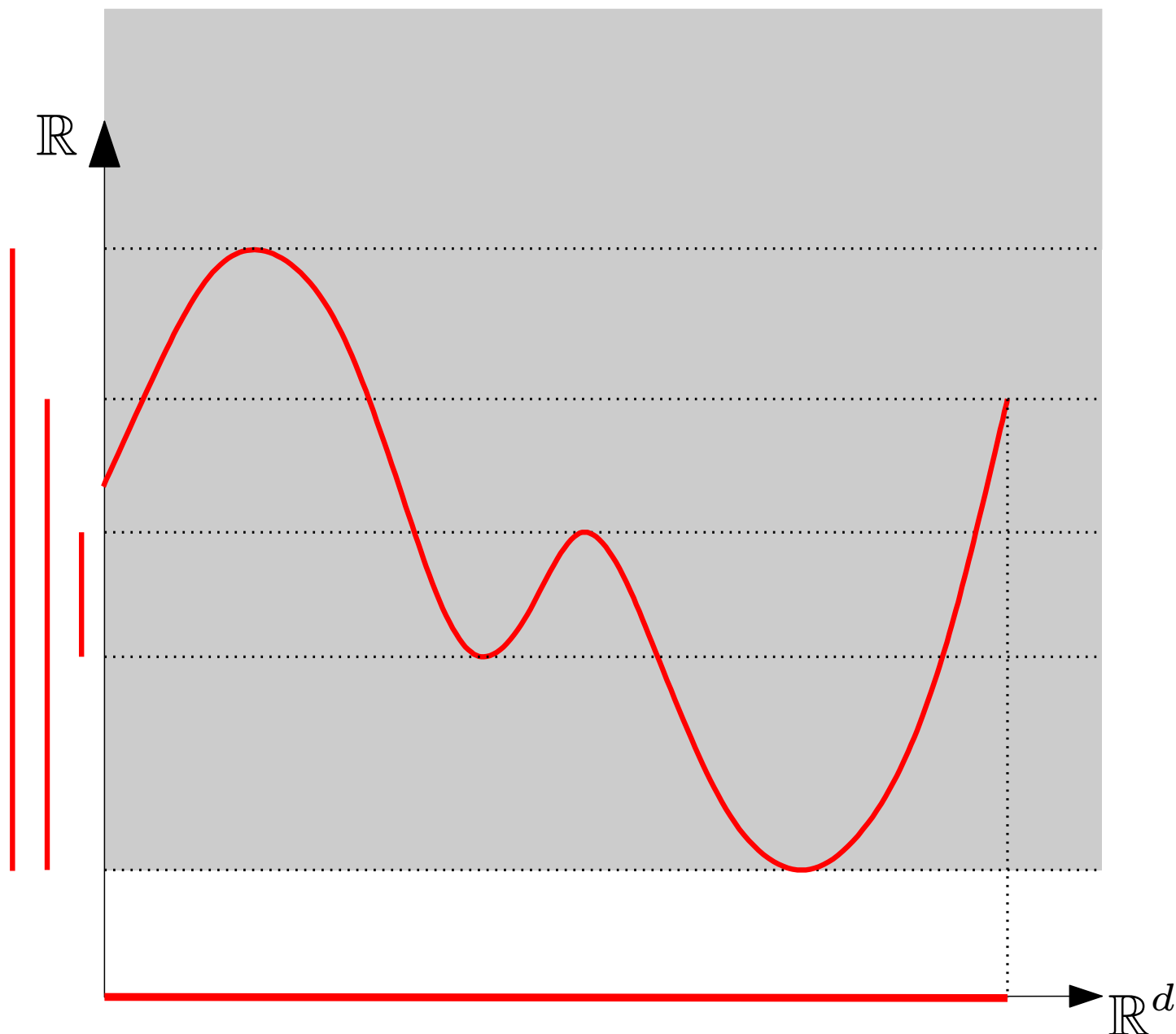
- Consider the superlevel-sets filtration $f^{-1}([t, +\infty))$ for t from $+\infty$ to $-\infty$, instead of the sublevel-sets filtration.
- Persistence is defined in the same way



Superlevel set persistence

Given a probability density f :

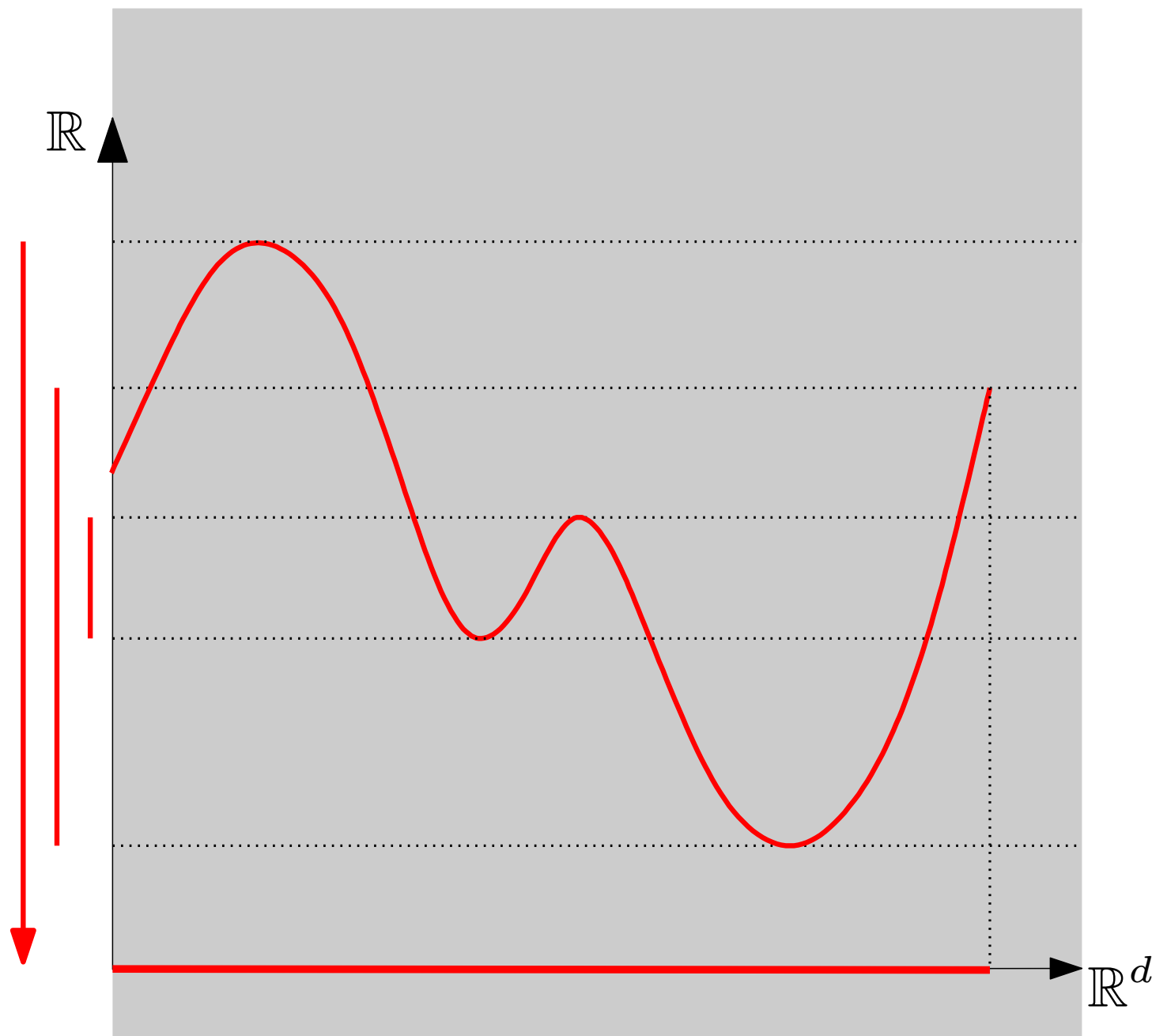
- Consider the superlevel-sets filtration $f^{-1}([t, +\infty))$ for t from $+\infty$ to $-\infty$, instead of the sublevel-sets filtration.
- Persistence is defined in the same way



Superlevel set persistence

Given a probability density f :

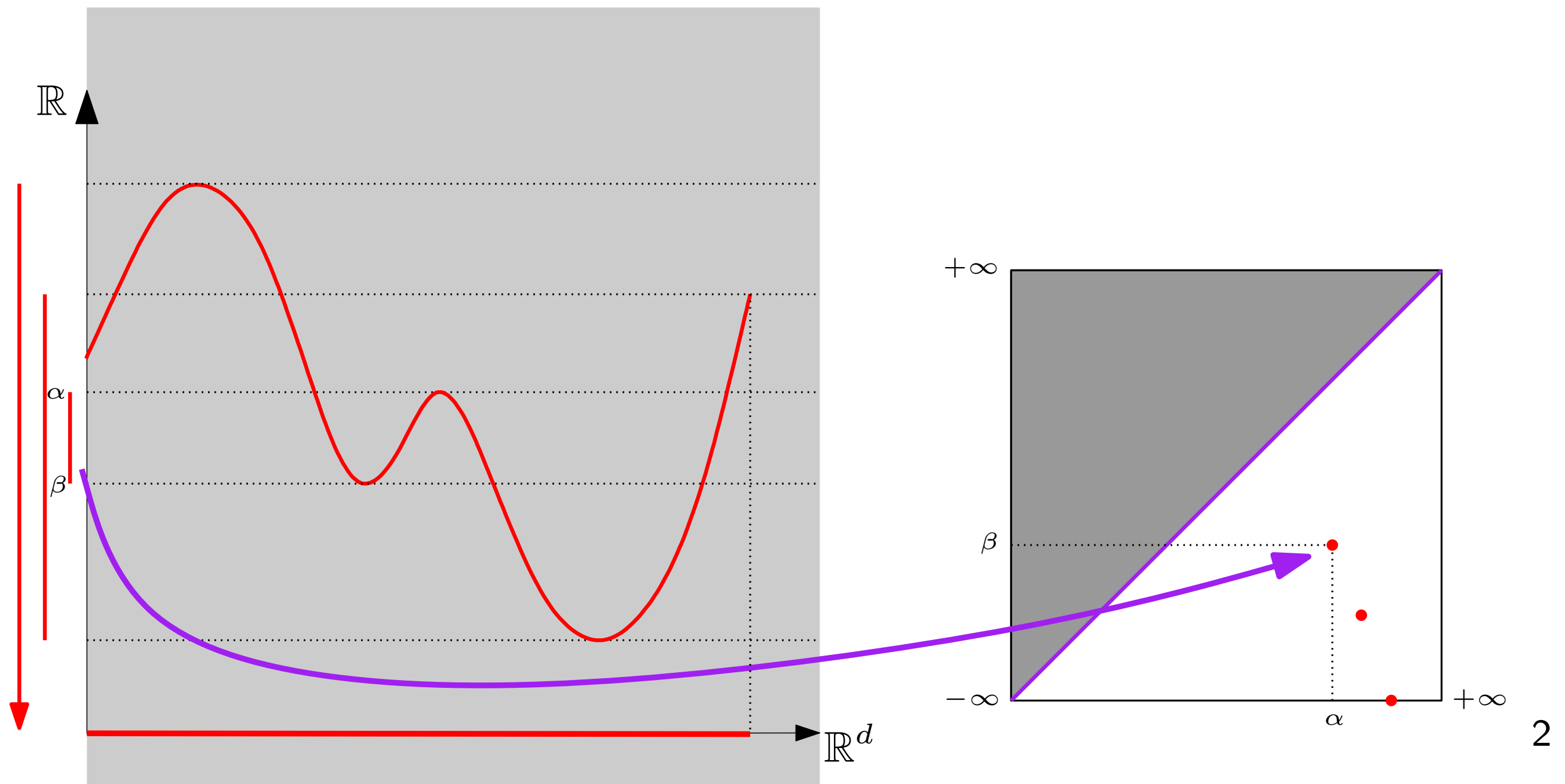
- Consider the superlevel-sets filtration $f^{-1}([t, +\infty))$ for t from $+\infty$ to $-\infty$, instead of the sublevel-sets filtration.
- Persistence is defined in the same way



Superlevel set persistence

Given a probability density f :

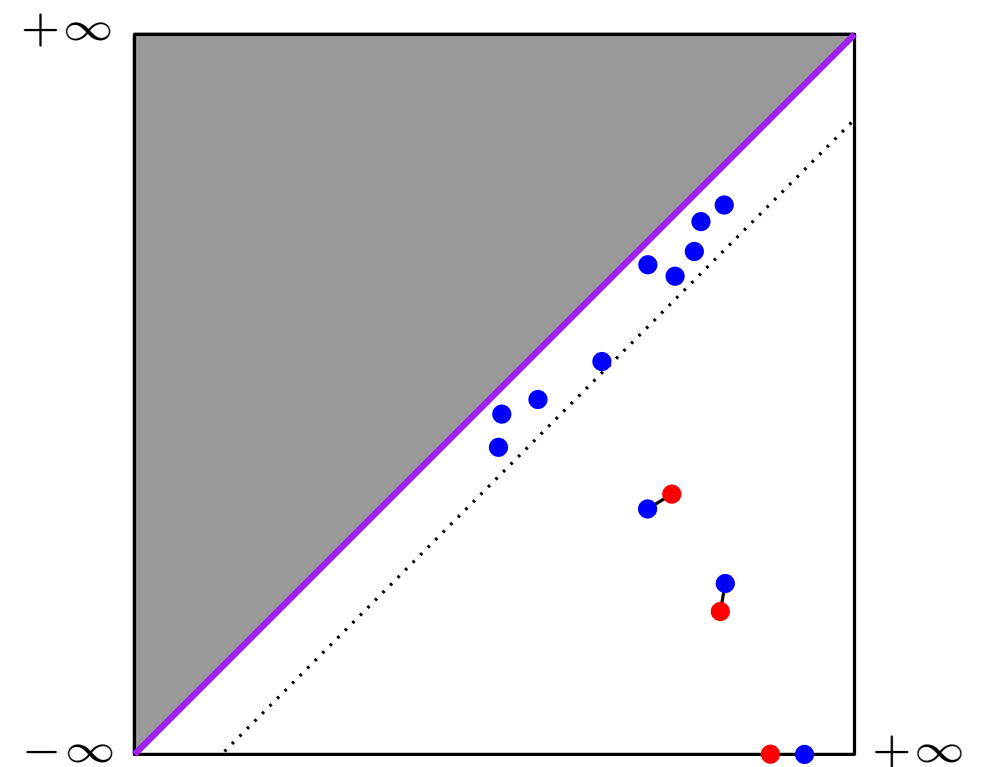
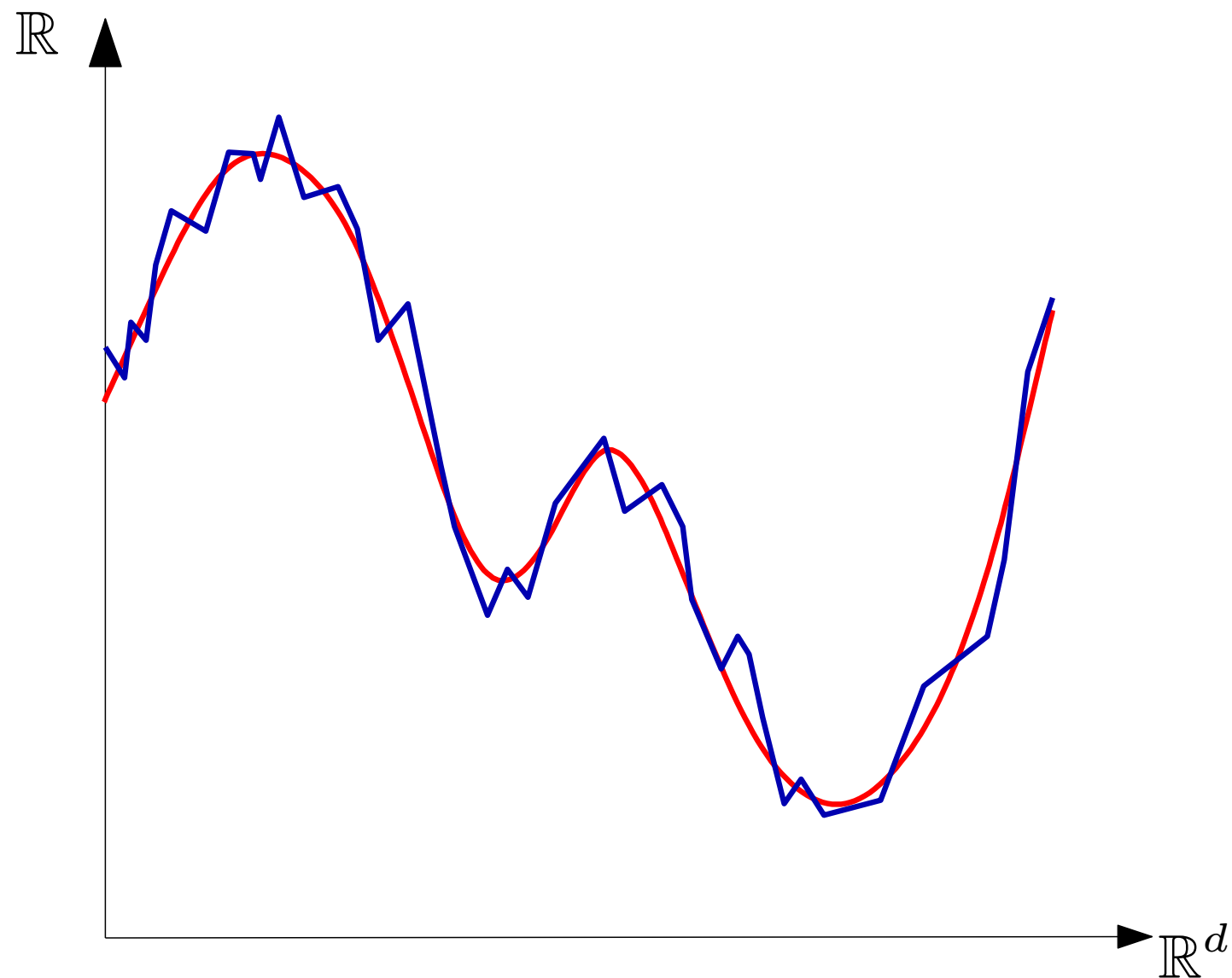
- Consider the superlevel-sets filtration $f^{-1}([t, +\infty))$ for t from $+\infty$ to $-\infty$, instead of the sublevel-sets filtration.
- Persistence is defined in the same way



Superlevel set persistence

Given an estimator \hat{f} :

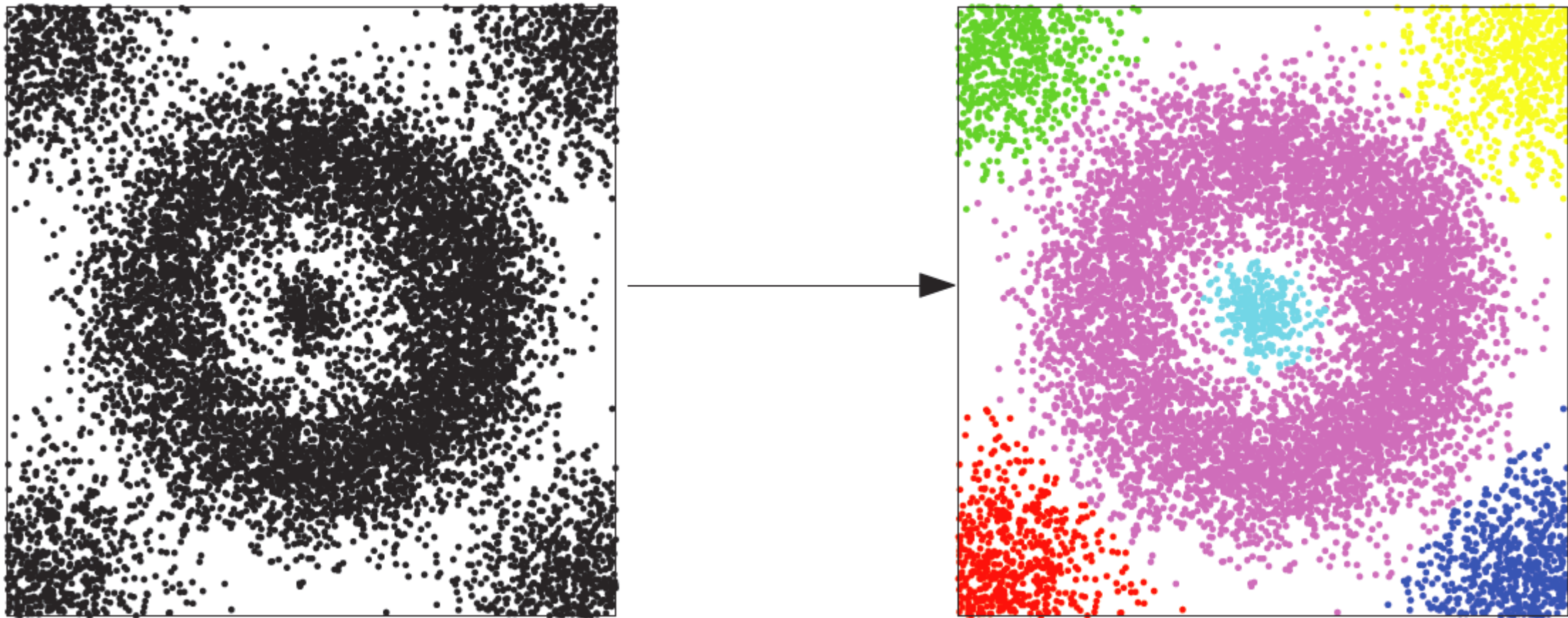
$$\text{Stability Theorem} \Rightarrow d_B(Df, D\hat{f}) \leq \|f - \hat{f}\|_\infty.$$



Persistence-based clustering (ToMATo)

Combine a mode seeking approach with (0-dim) persistence computation.

[C., Guibas, Oudot, Skraba - J. ACM 2013]



Input:

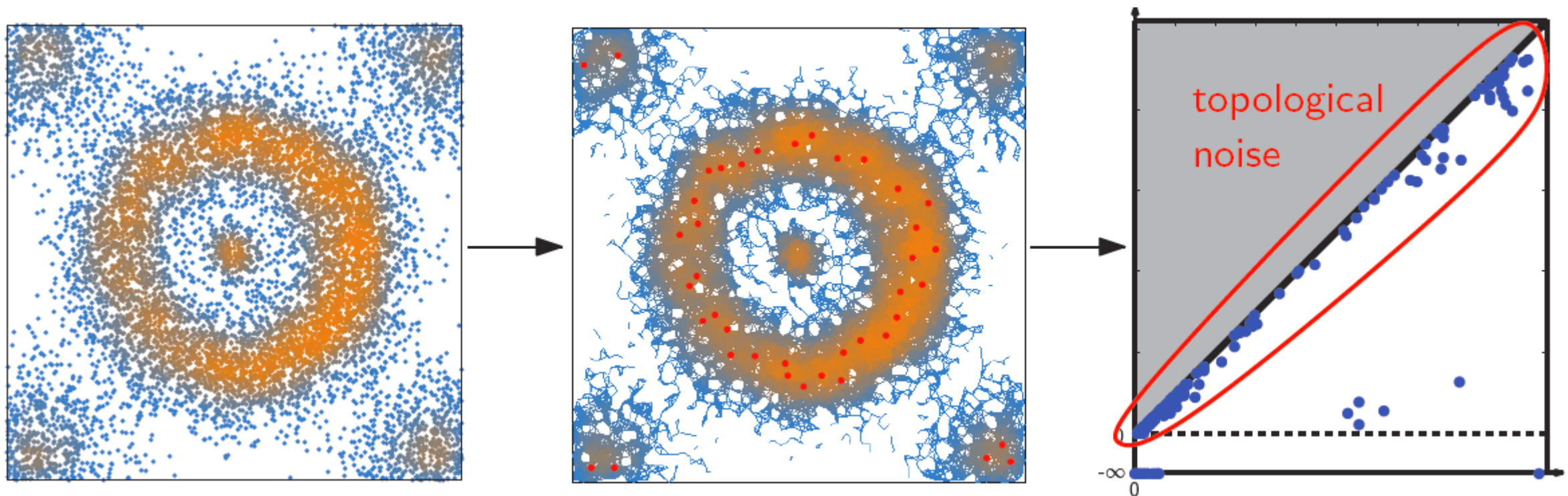
1. A finite set X of observations (point cloud with coordinates or pairwise distance matrix),
2. A real valued function f defined on the observations (e.g. density estimate).

Goal: Partition the data according to the basins of attraction of the peaks of f

Persistence-based clustering (ToMATo)

Combine a mode seeking approach with (0-dim) persistence computation.

[C., Guibas, Oudot, Skraba - J. ACM 2013]

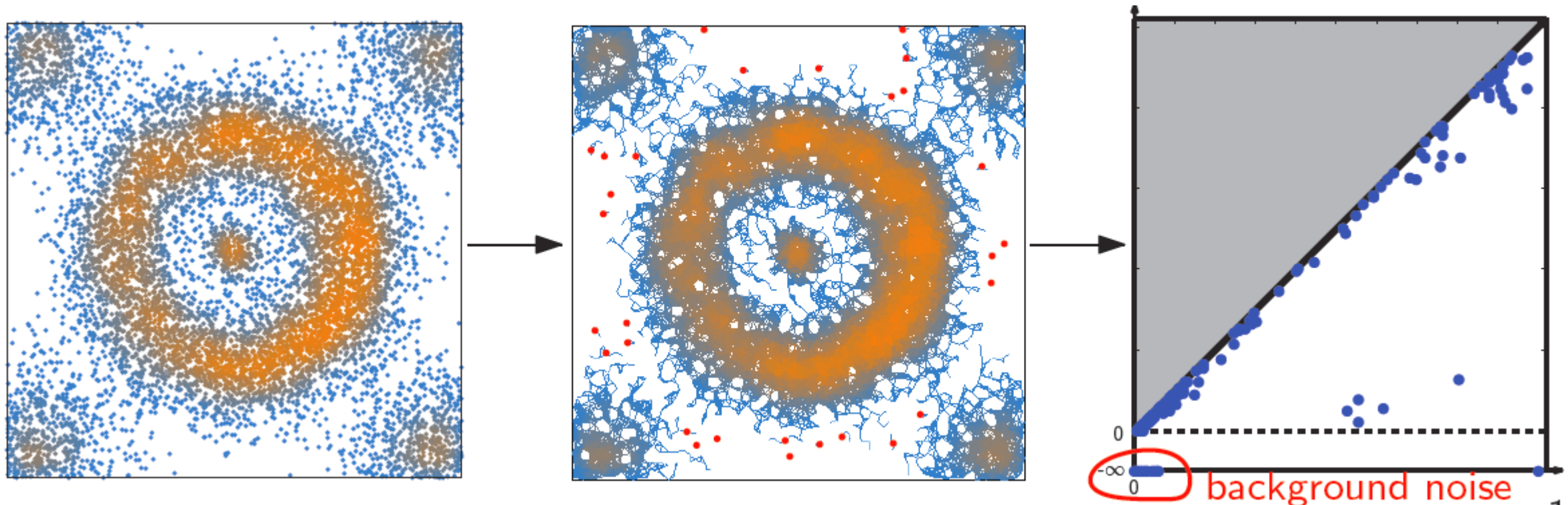


1. Build a neighboring graph G on top of X .
2. Compute the (0-dim) persistence of f to identify prominent peaks \rightarrow number of clusters (union-find algorithm).

Persistence-based clustering (ToMATo)

Combine a mode seeking approach with (0-dim) persistence computation.

[C., Guibas, Oudot, Skraba - J. ACM 2013]

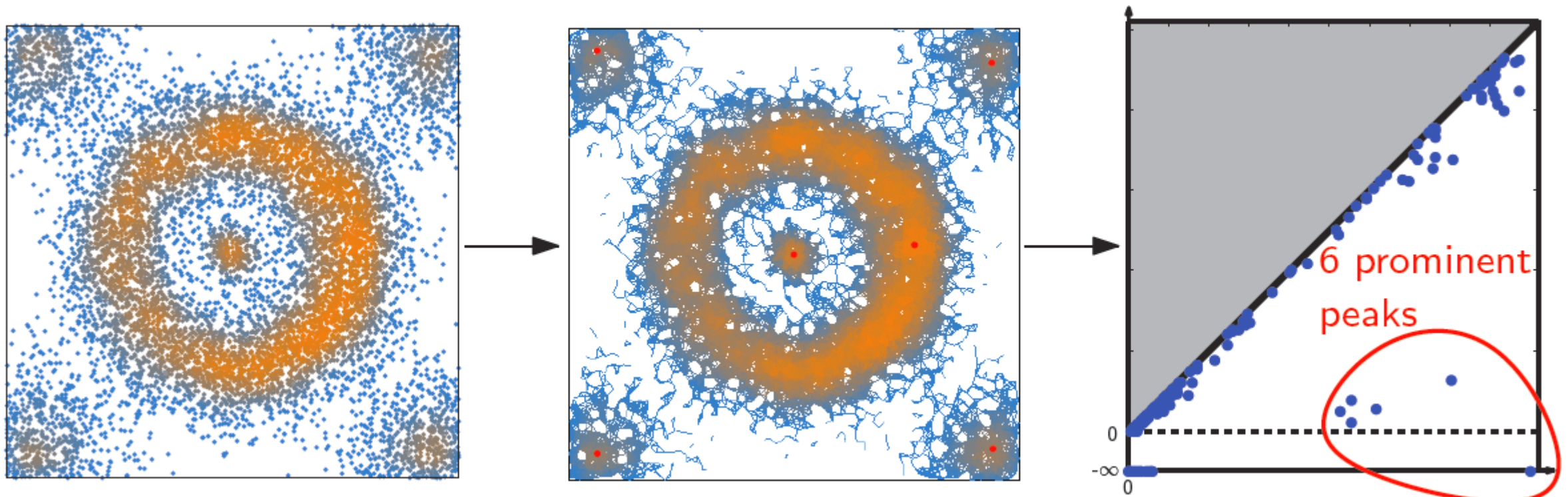


1. Build a neighboring graph G on top of X .
2. Compute the (0-dim) persistence of f to identify prominent peaks \rightarrow number of clusters (union-find algorithm).

Persistence-based clustering (ToMATo)

Combine a mode seeking approach with (0-dim) persistence computation.

[C., Guibas, Oudot, Skraba - J. ACM 2013]

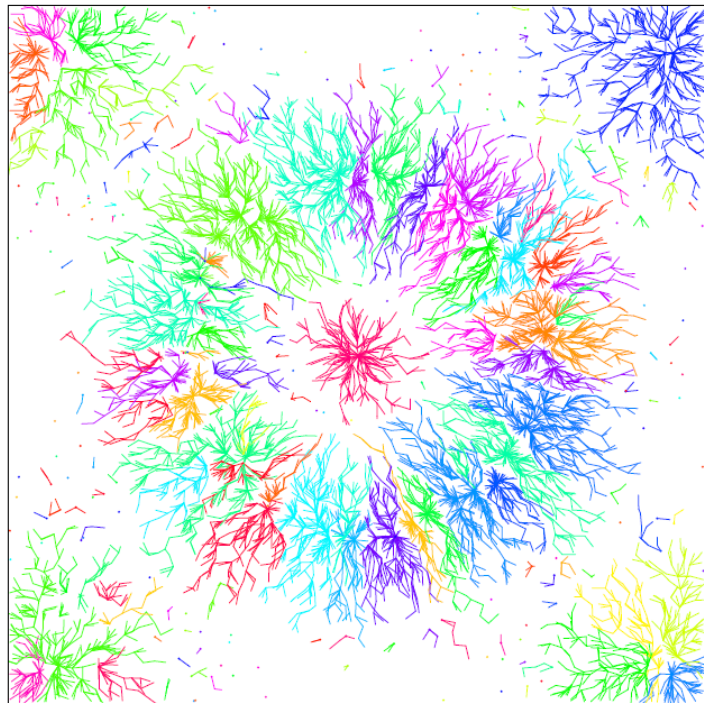


1. Build a neighboring graph G on top of X .
2. Compute the (0-dim) persistence of f to identify prominent peaks \rightarrow number of clusters (union-find algorithm).

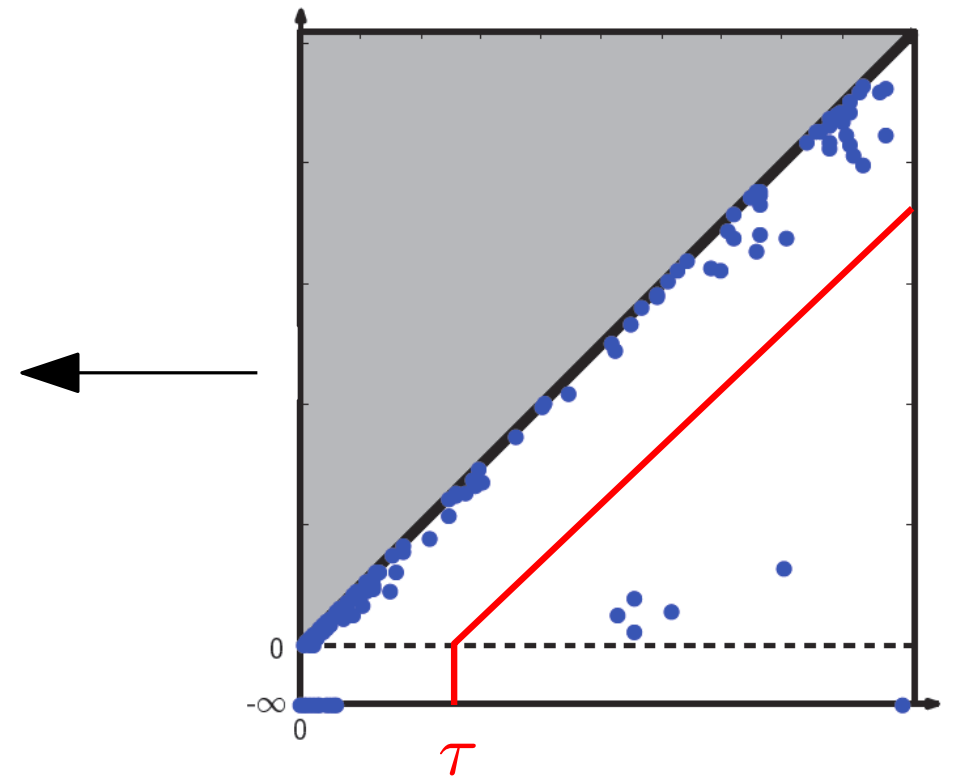
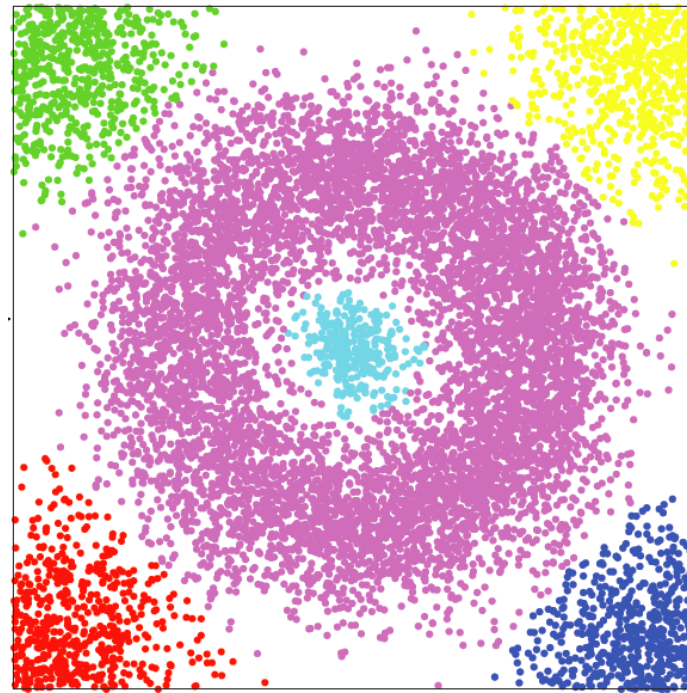
Persistence-based clustering (ToMATo)

Combine a mode seeking approach with (0-dim) persistence computation.

[C., Guibas, Oudot, Skraba - J. ACM 2013]



$\tau = 0$

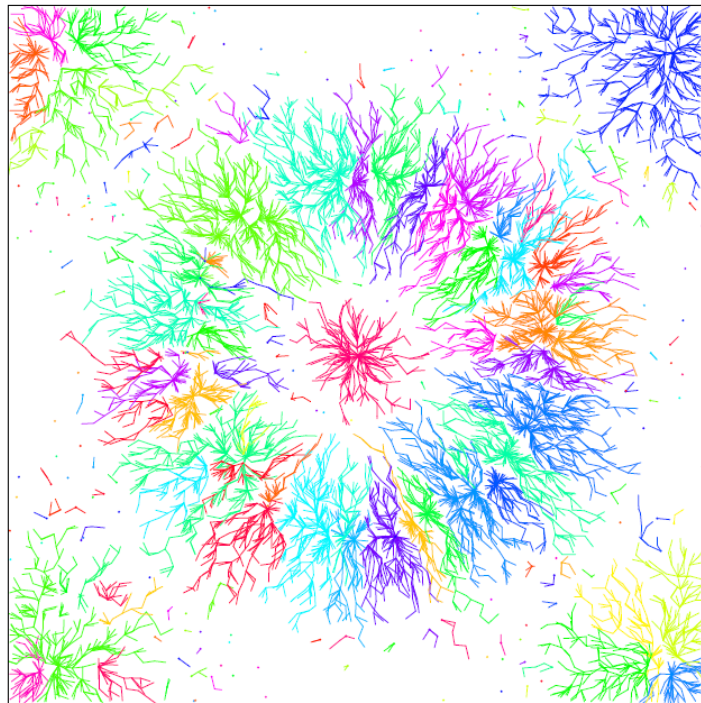


1. Build a neighboring graph G on top of X .
2. Compute the (0-dim) persistence of f to identify prominent peaks \rightarrow number of clusters (union-find algorithm).
3. Chose a threshold $\tau > 0$ and use the persistence algorithm to merge components with prominence less than τ .

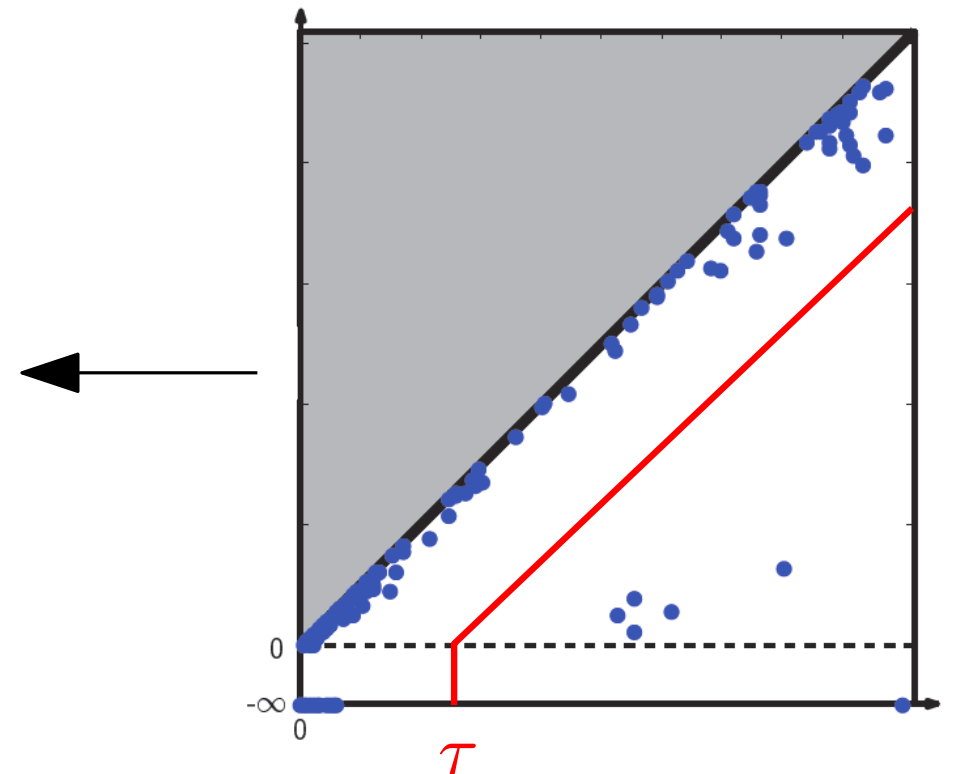
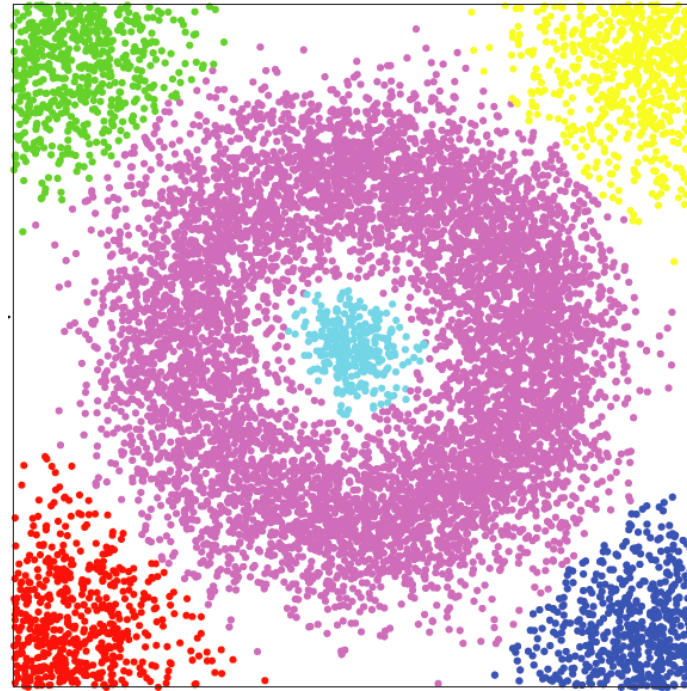
Persistence-based clustering (ToMATo)

Combine a mode seeking approach with (0-dim) persistence computation.

[C., Guibas, Oudot, Skraba - J. ACM 2013]



$\tau = 0$



Complexity of the algorithm: $O(n \log n)$

Theoretical guarantees:

- Stability of the number of clusters (w.r.t. perturbations of X and f).
- Partial stability of clusters: well identified stable parts in each cluster.

→ “soft ” clustering

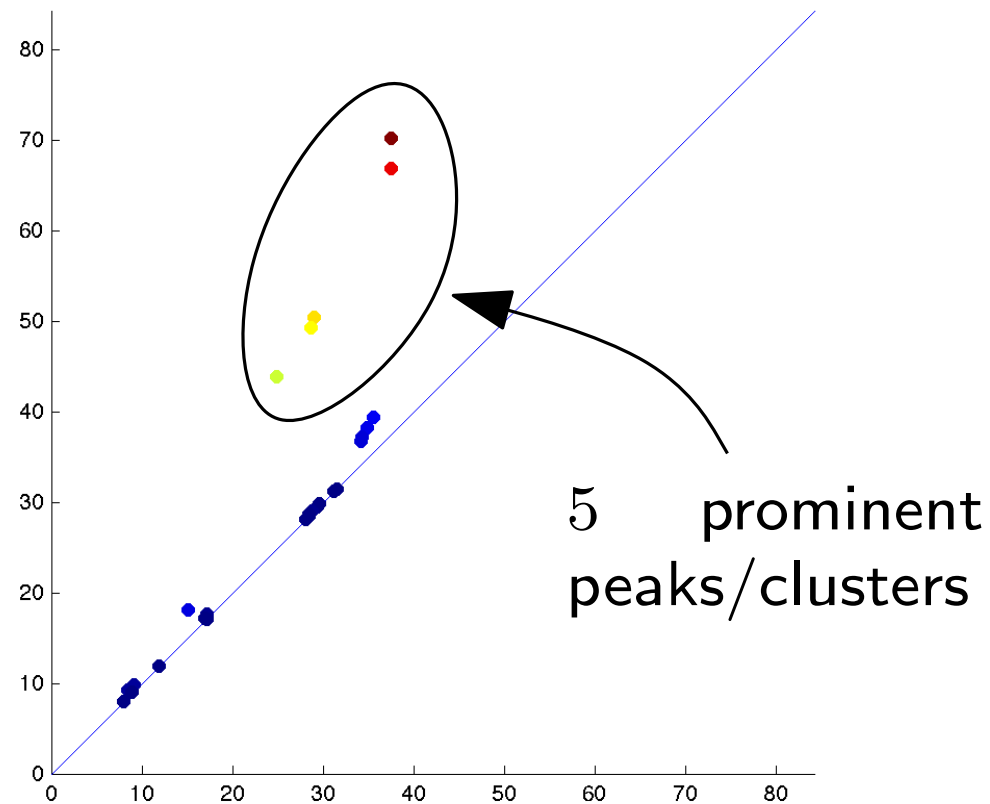
Application to non-rigid shape segmentation

[Skraba, Ovsjanikov, C., Guibas, NORDIA 10]



X : a 3D shape
 $f = \text{HKS}$ function on X

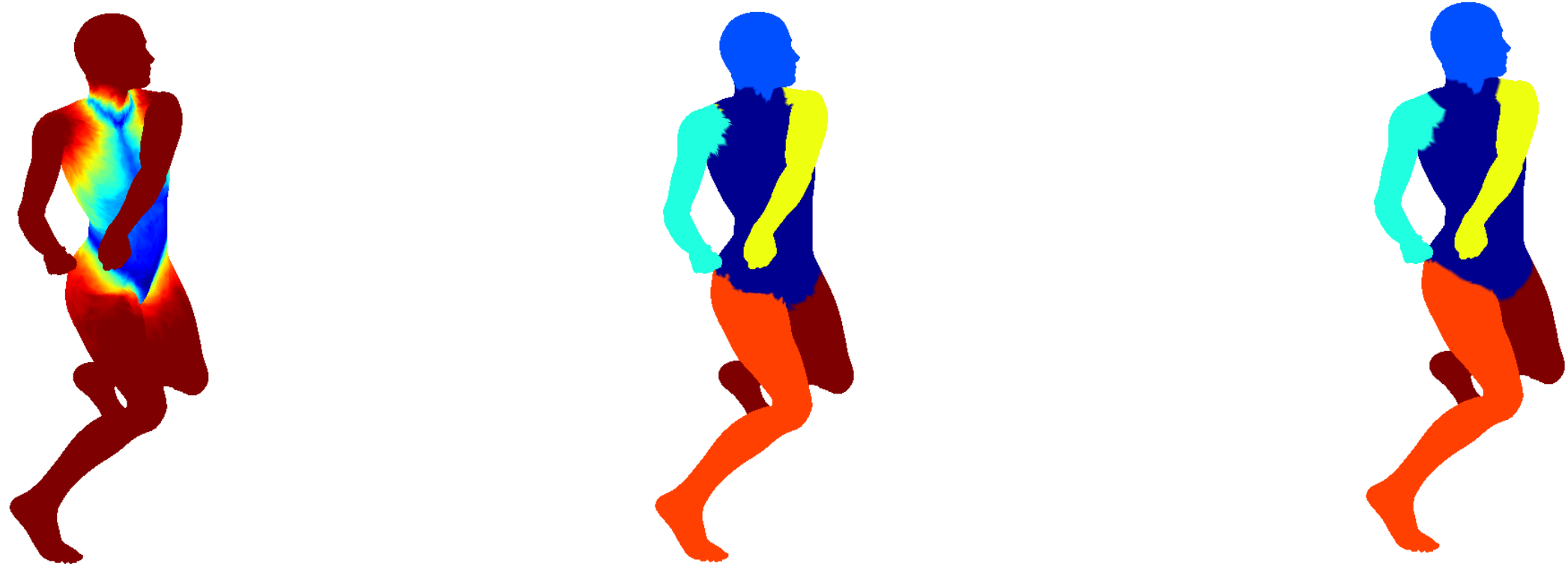
Persistence diagram for david1 with $f = \text{HKS}(0.1)$



Problem: some part of clusters are unstable \rightarrow dirty segments

Application to non-rigid shape segmentation

[Skraba, Ovsjanikov, C., Guibas, NORDIA 10]



Problem: some part of clusters are unstable \rightarrow dirty segments

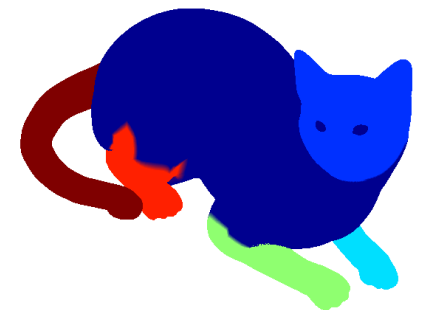
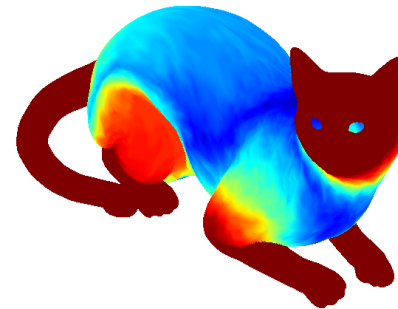
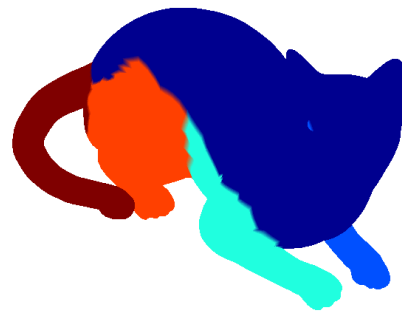
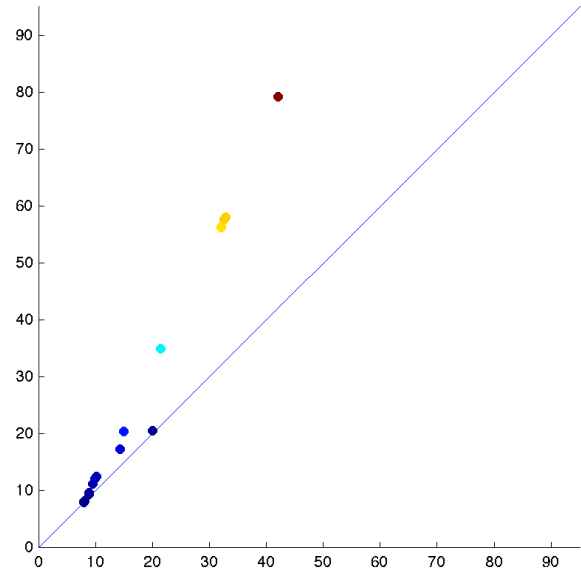
Idea:

- Run the persistence based algorithm several times on random perturbations of f (size bounded by the “persistence” gap).
- Partial stability of clusters allows to establish correspondences between clusters across the different runs \rightarrow for any $x \in X$, a vector giving the probability for x to belong to each cluster.

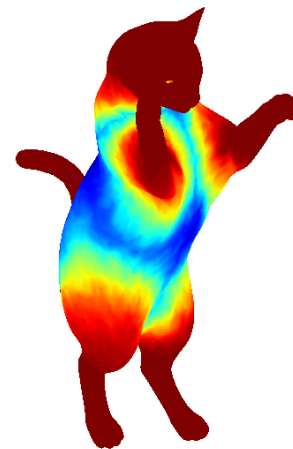
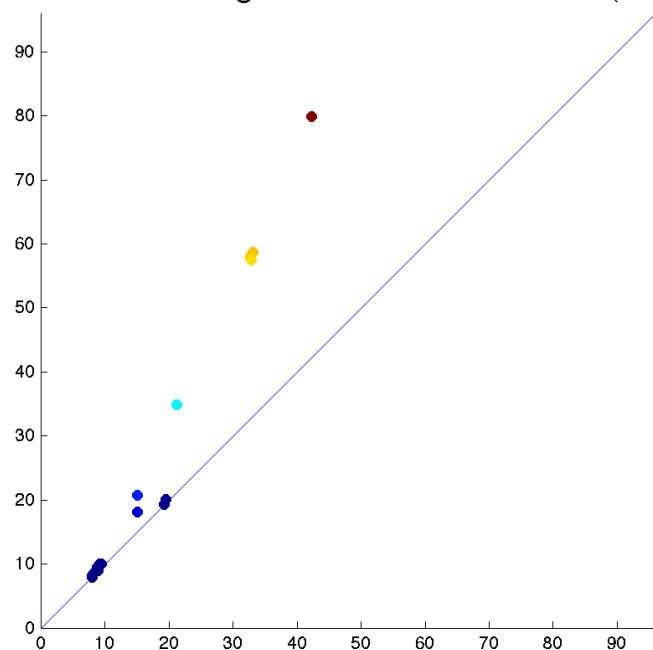
Application to non-rigid shape segmentation

[Skraba, Ovsjanikov, C., Guibas, NORDIA 10]

Persistence diagram for cat7 with $f = \text{HKS}(0.1)$



Persistence diagram for cat1 with $f = \text{HKS}(0.1)$



An example of application

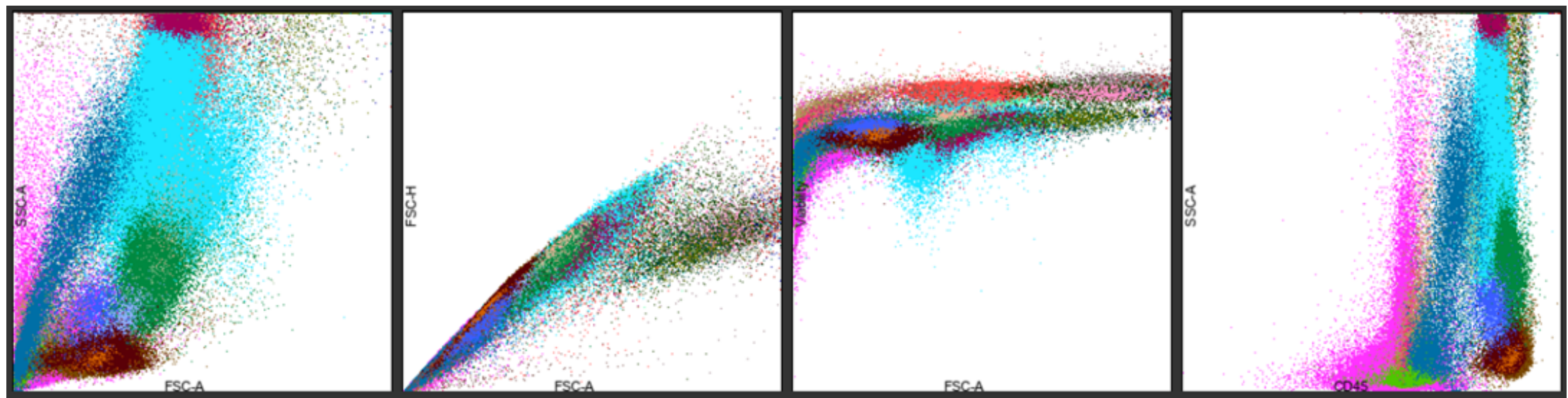
Topology-based unsupervised classification [C, Guibas, Oudot, Skraba 2013]

Segmentation of cytometry data for medical diagnosis

[M. Glisse, L. Pujol et al 2020]



An innovative start-up specialized in biological diagnosis from cytometry data.

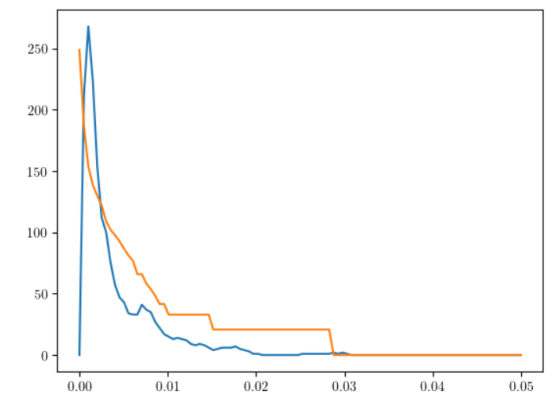
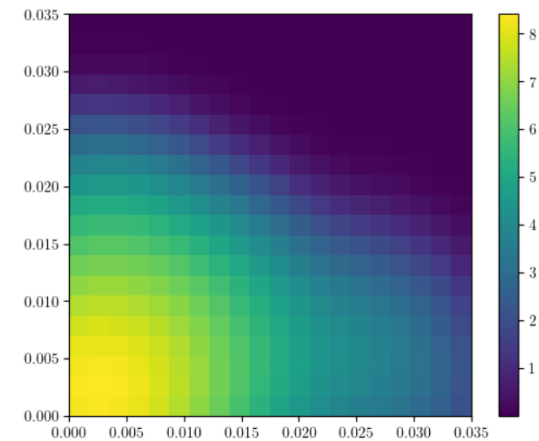
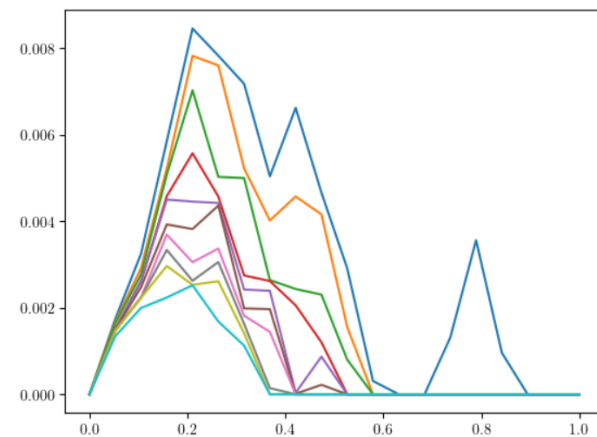
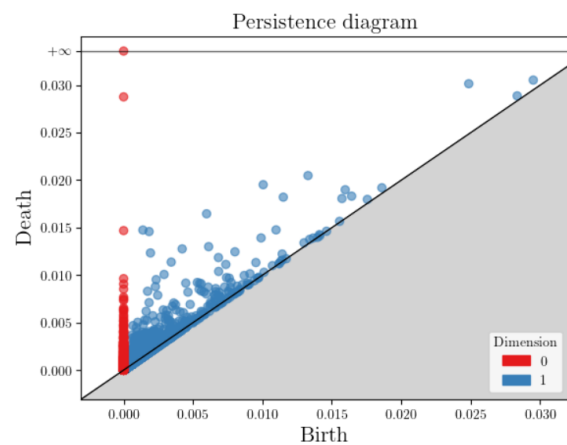


Objective: unsupervised learning in large point clouds (several millions) in medium/high dimensions ($\approx 4 \rightarrow 80$)

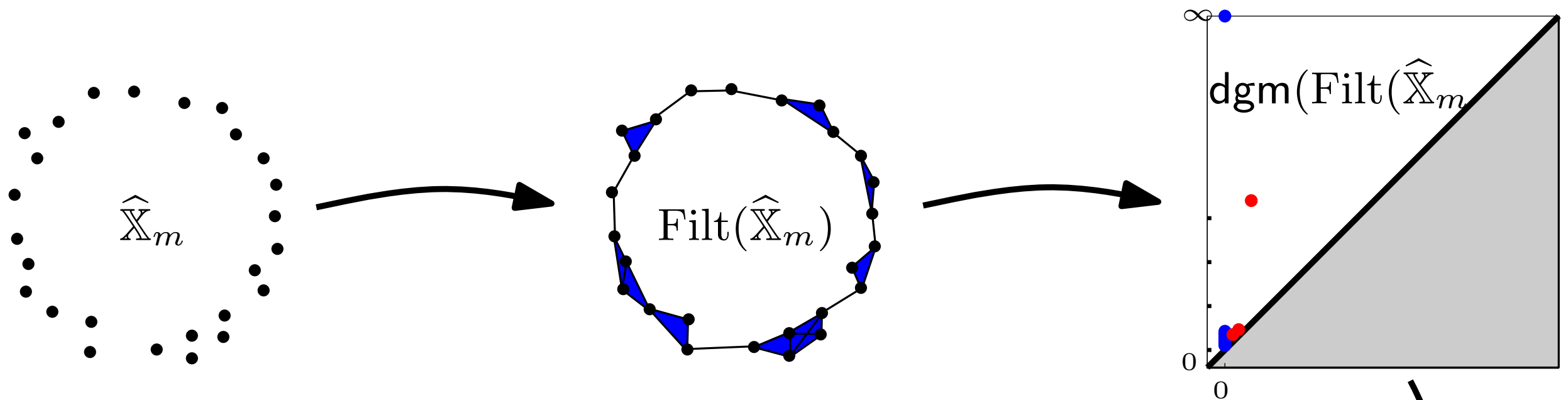
Applications: medical diagnosis from blood samples (1 point = 1 blood cell)

Methodology: Persistence-based clustering to robustly identify relevant clusters.

Persistent homology as a (robust) feature engineering tool: vectorization of persistence diagrams



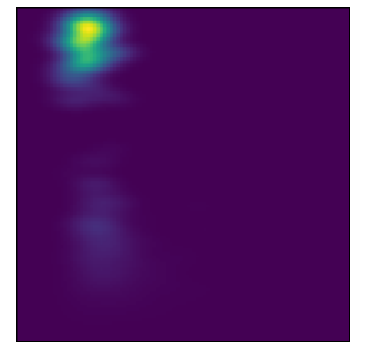
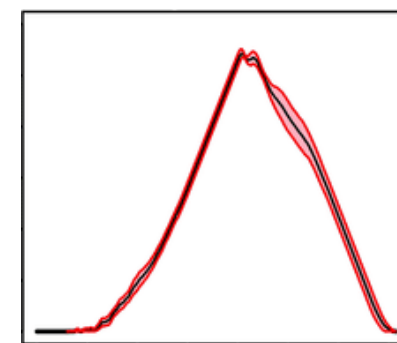
The problem of representation of persistence



Persistence diagrams are not well-suited for classical ML algorithms (the space of PD is highly non linear)

Not always clear which part of the diagrams carries the relevant information.

Machine
Learning / AI



**Representations of
persistence**

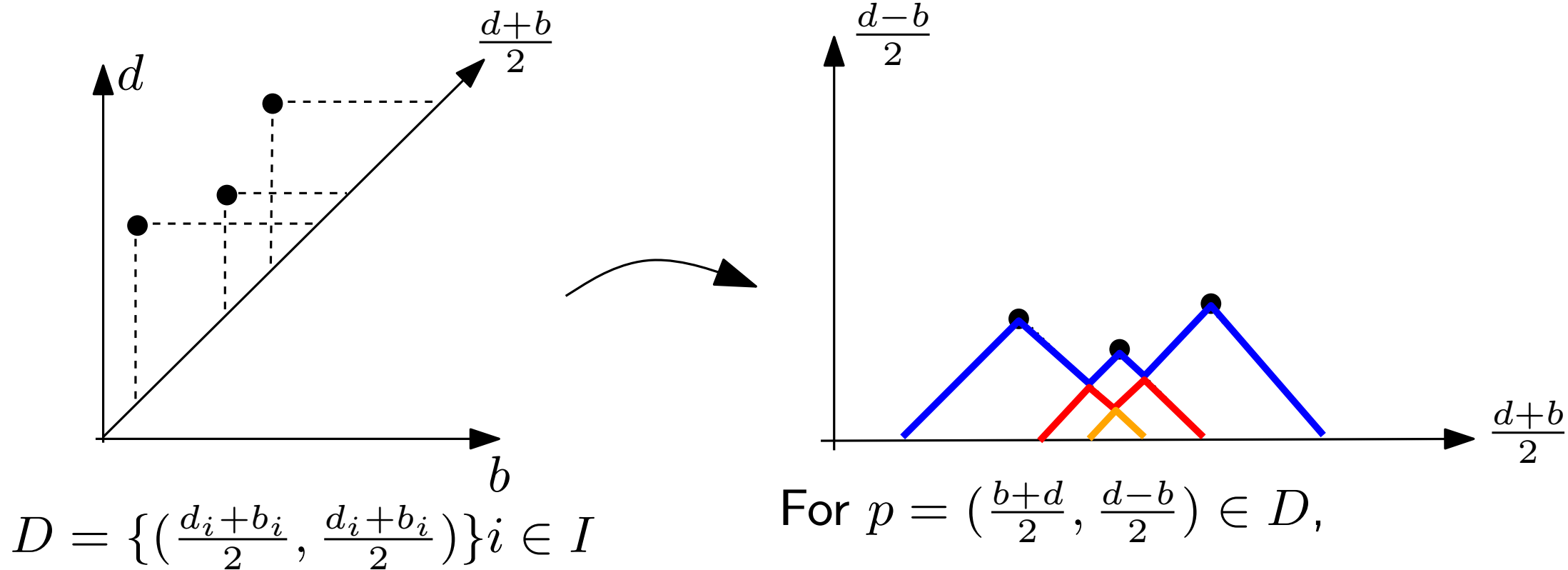
A zoo of representations of persistence

(non exhaustive list - see also Gudhi representations)

- Collections of 1D functions
 - landscapes [Bubenik 2012]
 - Betti curves [Umeda 2017]
- discrete measures point of view: (interesting statistical properties [C., Divoi 2018])
 - persistence images [Adams et al 2017]
 - ATOL [C. et al 2021]
 - convolution with Gaussian kernel [Reininghaus et al. 2015] [Chepushtanova et al. 2015] [Kusano Fukumisu Hiraoka 2016-17] [Le Yamada 2018]
 - sliced on lines [Carrière Oudot Cuturi 2017]
- finite metric spaces [Carrière Oudot Ovsjanikov 2015]
- polynomial roots or evaluations [Di Fabio Ferri 2015] [Kališnik 2016]

Persistence landscapes

[Bubenik 2012]



$$\Lambda_p(t) = \begin{cases} t - b & t \in [b, \frac{b+d}{2}] \\ d - t & t \in (\frac{b+d}{2}, d] \\ 0 & \text{otherwise.} \end{cases}$$

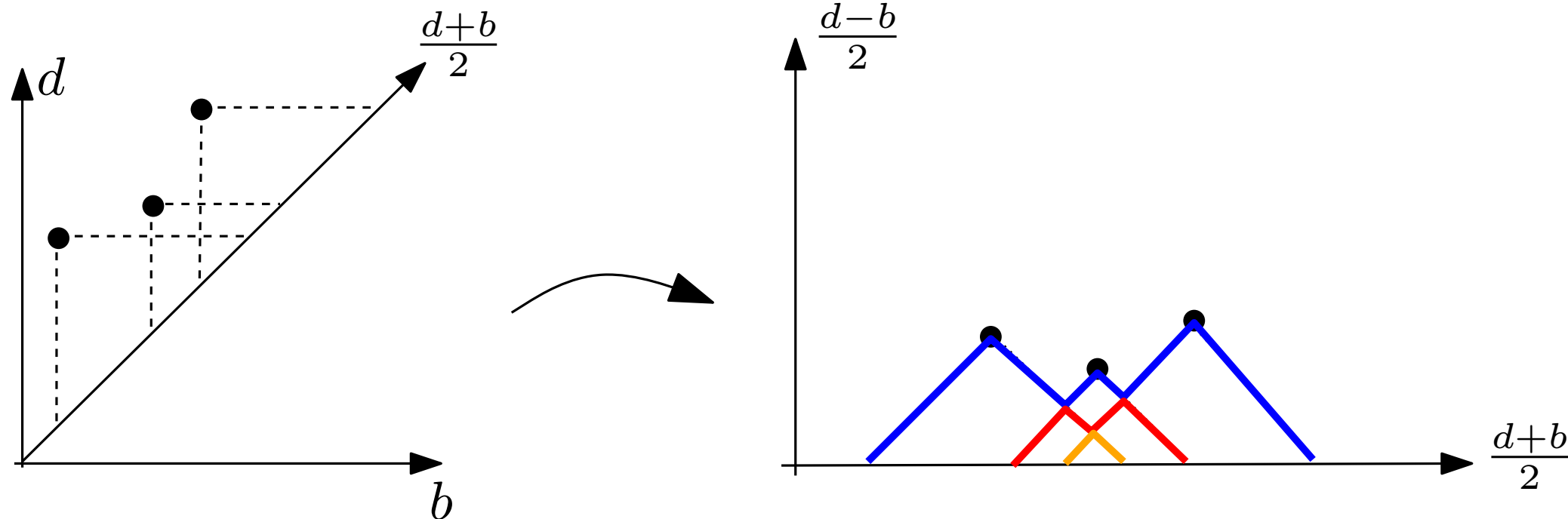
Persistence landscape [Bubenik 2012]:

$$\lambda_D(k, t) = \text{kmax}_{p \in \text{dgm}} \Lambda_p(t), \quad t \in \mathbb{R}, k \in \mathbb{N},$$

where kmax is the k th largest value in the set.

Persistence landscapes

[Bubenik 2012]



Persistence landscape [Bubenik 2012]:

$$\lambda_D(k, t) = k \max_{p \in \text{dgm}} \Lambda_p(t), \quad t \in \mathbb{R}, k \in \mathbb{N},$$

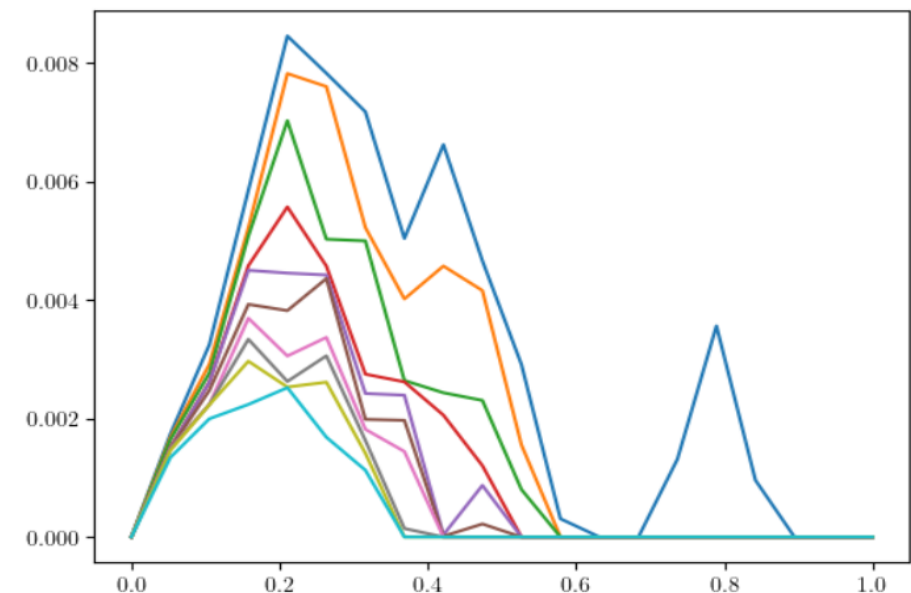
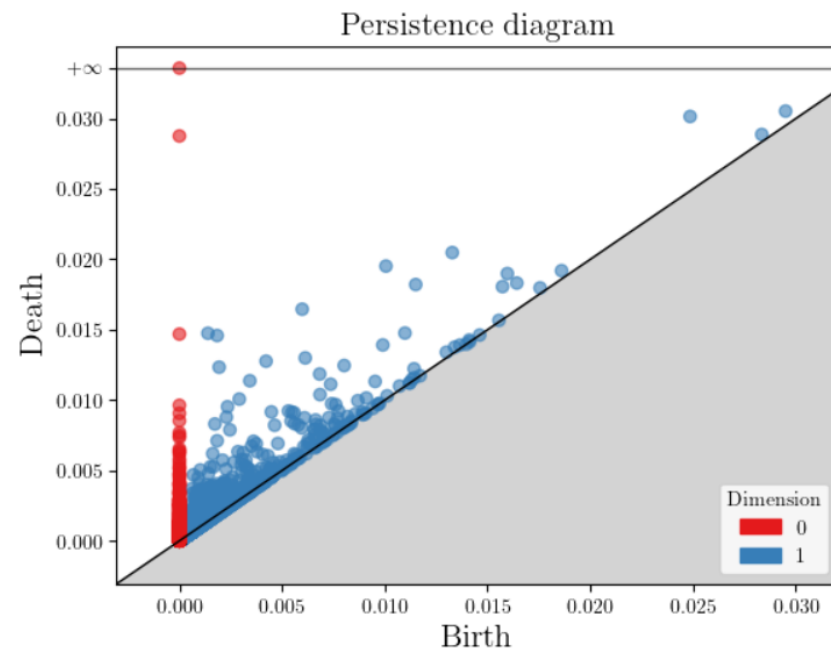
Properties

- For any $t \in \mathbb{R}$ and any $k \in \mathbb{N}$, $0 \leq \lambda_D(k, t) \leq \lambda_D(k+1, t)$.
- For any $t \in \mathbb{R}$ and any $k \in \mathbb{N}$, $|\lambda_D(k, t) - \lambda_{D'}(k, t)| \leq d_B(D, D')$ where $d_B(D, D')$ denotes the bottleneck distance between D and D' .

stability properties of persistence landscapes

Persistence landscapes

[Bubenik 2012]



- Persistence encoded as an element of a functional space (vector space!).
- Expectation of distribution of landscapes is well-defined and can be approximated from average of sampled landscapes.
- Process point of view: convergence results and convergence rates → confidence intervals can be computed using bootstrap.

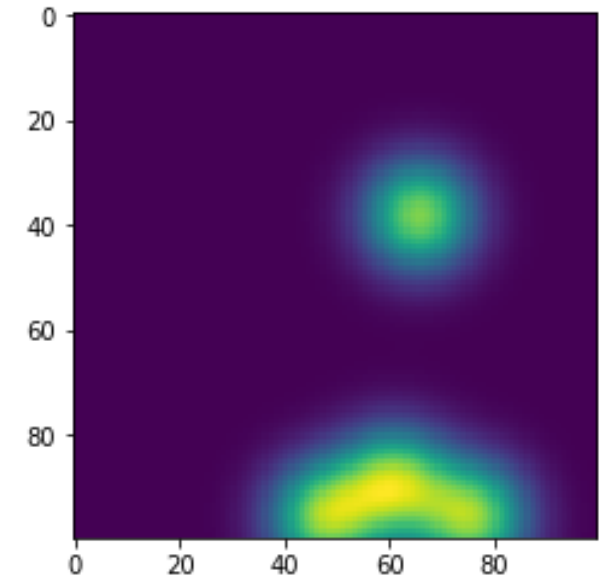
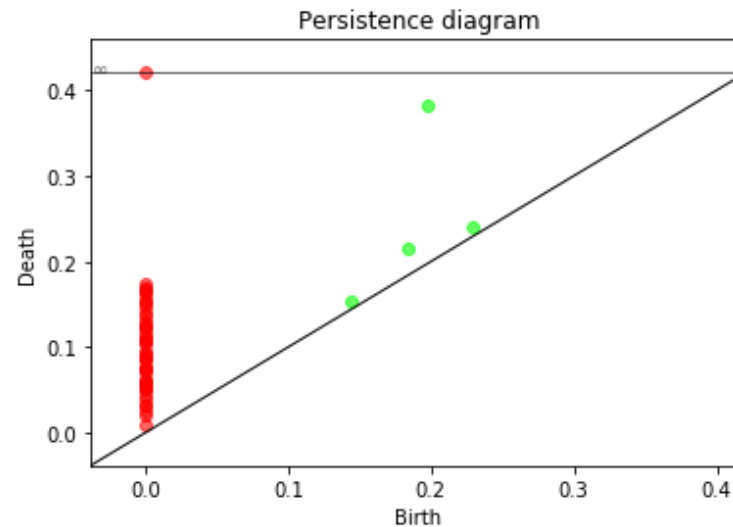
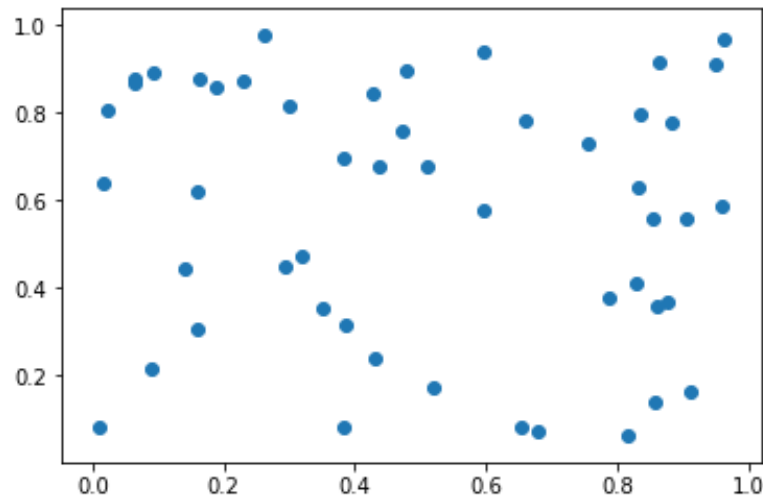
[C., Fasy, Lecci, Rinaldo, Wasserman SoCG 2014]

- Provide a convenient way to process persistence information in deep neural networks.

[Kim, Kim, Zaheer, Kim, C., Wasserman NeurIPS 2020,
Carrière, C., Ike, Lacombe, Royer, Umeda AISTAT 2020]

Persistence images

[Adams et al, JMLR 2017]



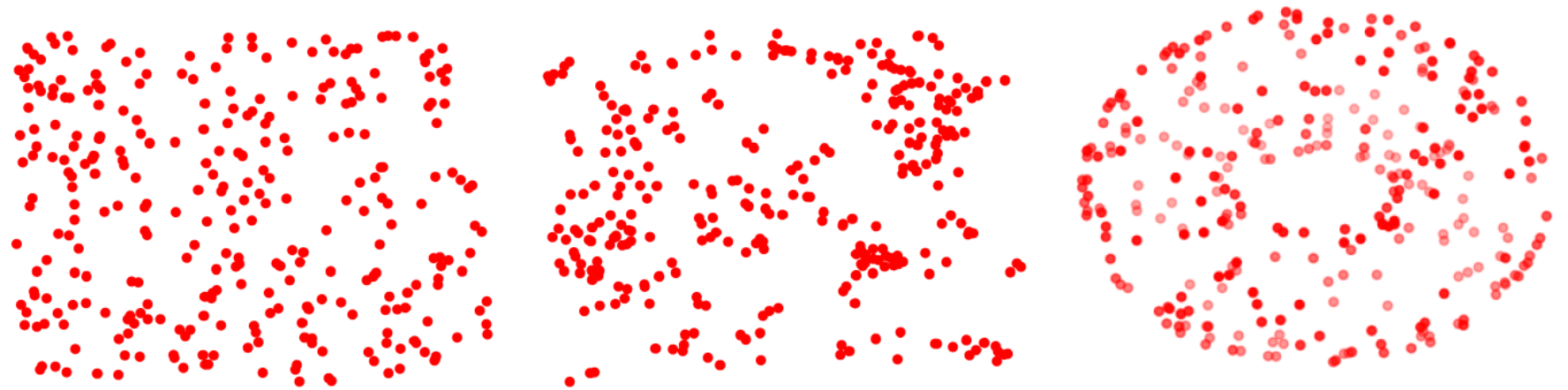
For $K : \mathbb{R}^2 \rightarrow \mathbb{R}$ a kernel and H a bandwidth matrix (e.g. a symmetric positive definite matrix), pose for $u \in \mathbb{R}^2$, $K_H(u) = |H|^{-1/2} K(H^{-1/2} \cdot u)$

For $D = \sum_i \delta_{p_i}$ a diagram, $K : \mathbb{R}^2 \rightarrow \mathbb{R}$ a kernel, H a bandwidth matrix and $w : \mathbb{R}^2 \rightarrow \mathbb{R}_+$ a weight function, one defines the **persistence surface** of D with kernel K and weight function w by:

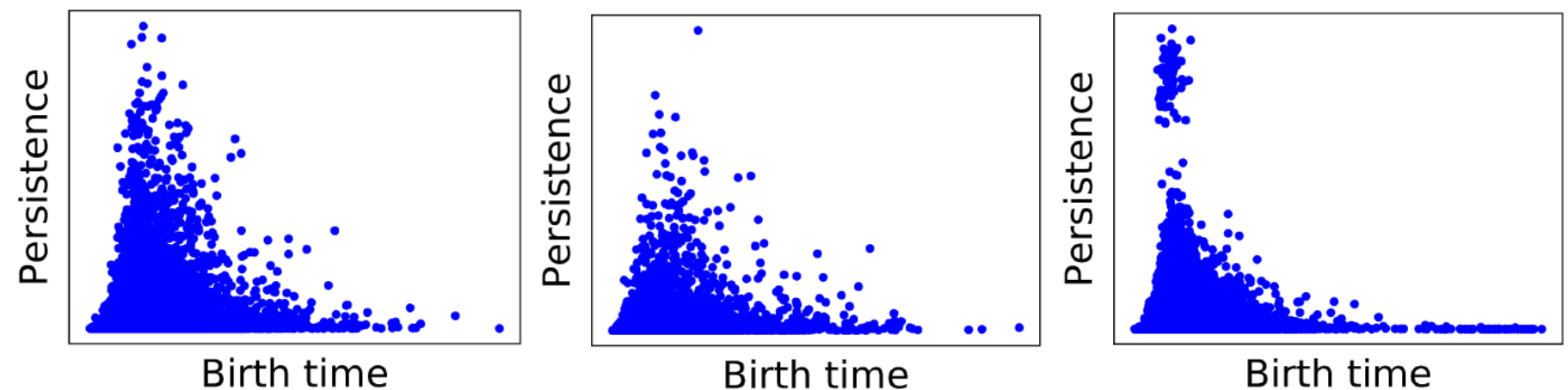
$$\forall u \in \mathbb{R}^2, \quad \rho(D)(u) = \sum_i w(p_i) K_H(u - p_i) = D(w K_H(u - \cdot))$$

Persistence images

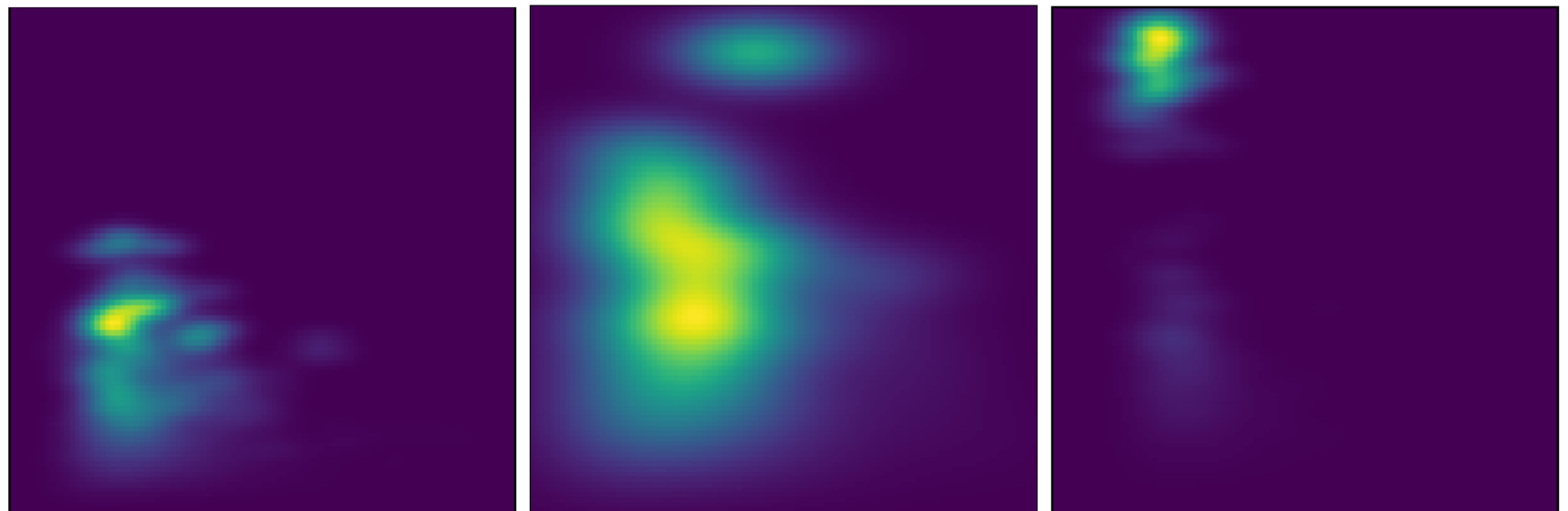
The realization of 3
different processes



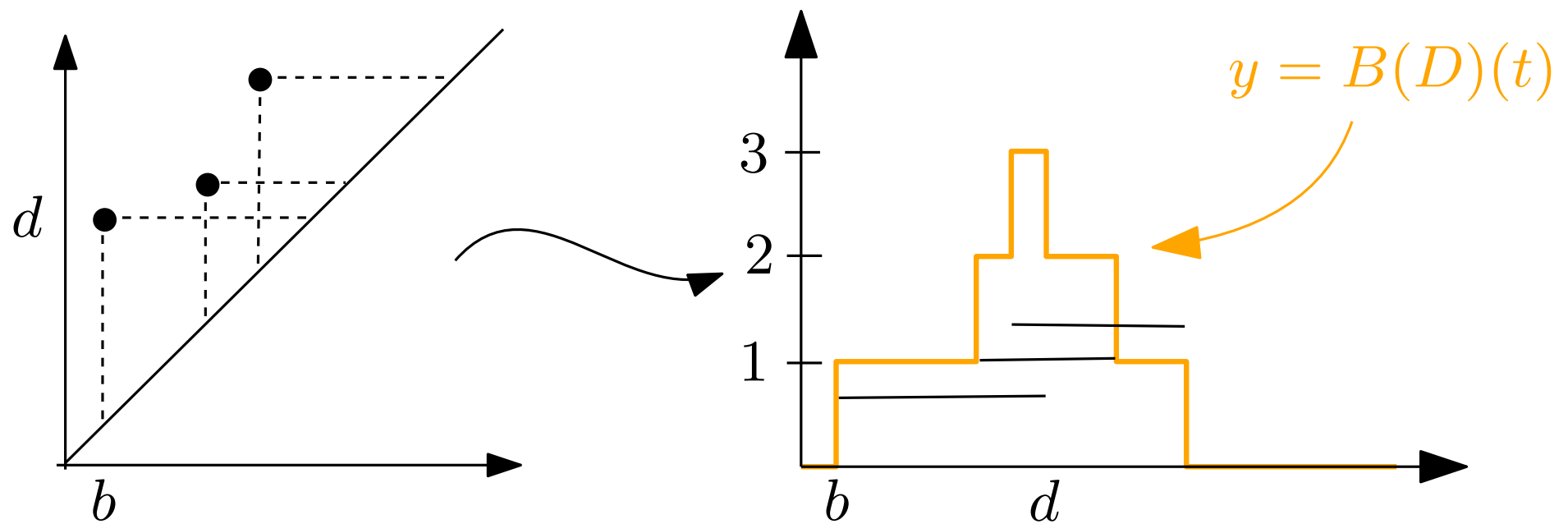
The overlay of 40
different persistence
diagrams



The persistence images
with weight function
 $w(\mathbf{r}) = (r_2 - r_1)^3$ and
bandwidth selected using
cross-validation.



Betti curves



Betti curve: The integer-valued piecewise constant function defined by :

$$D \rightarrow B(D) : \mathbb{R} \rightarrow \mathbb{N}$$

where

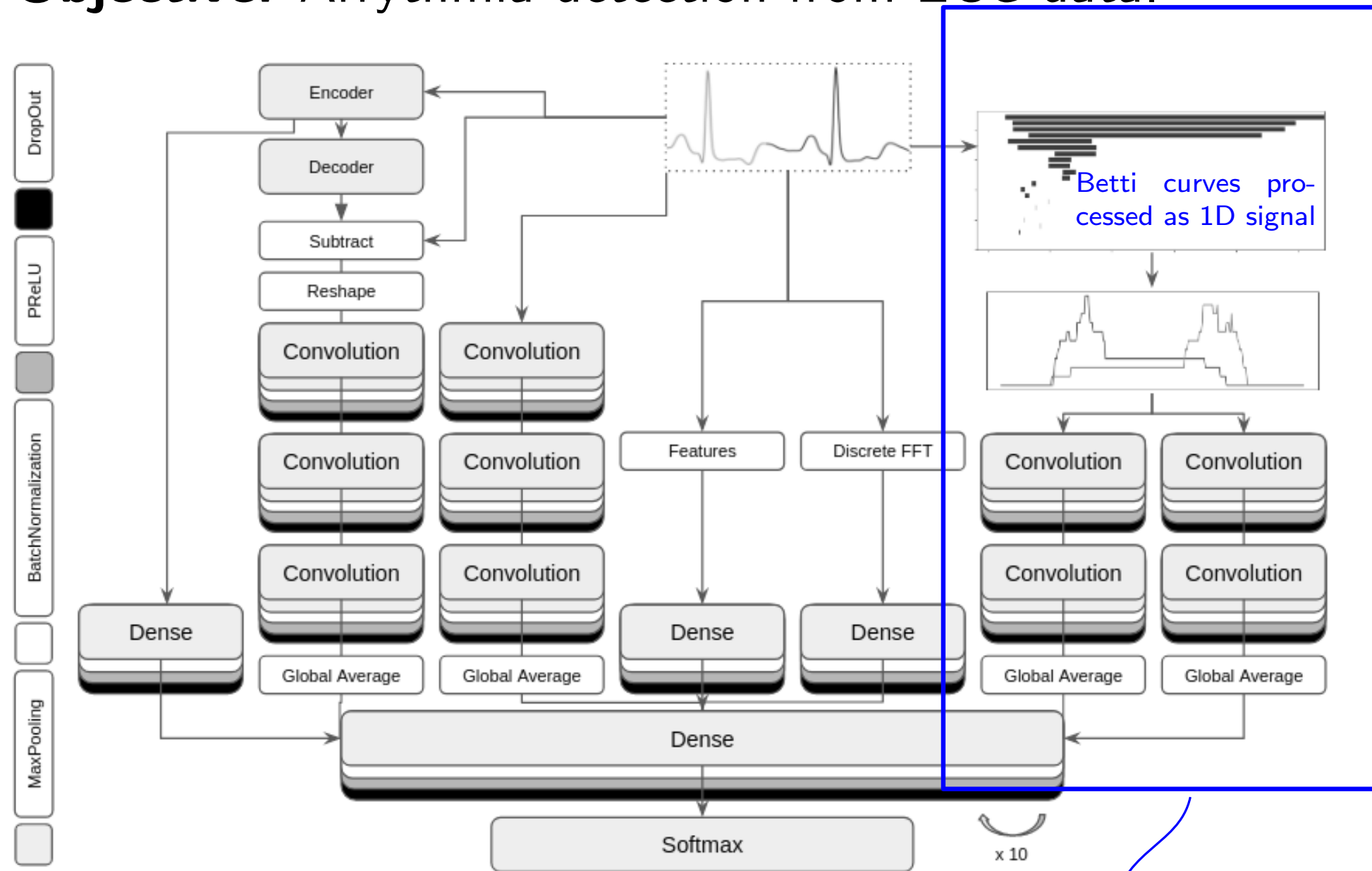
$$B(D)(t) = \#\{p = (b, d) \in D : b \leq t \leq d\}$$

Betti curves provides the betti numbers of each subspaces/subcomplexes of the considered filtration.

Example of application: arrhythmia detection

[Dindin, Umeda, C. Can. Conf. AI 2020]

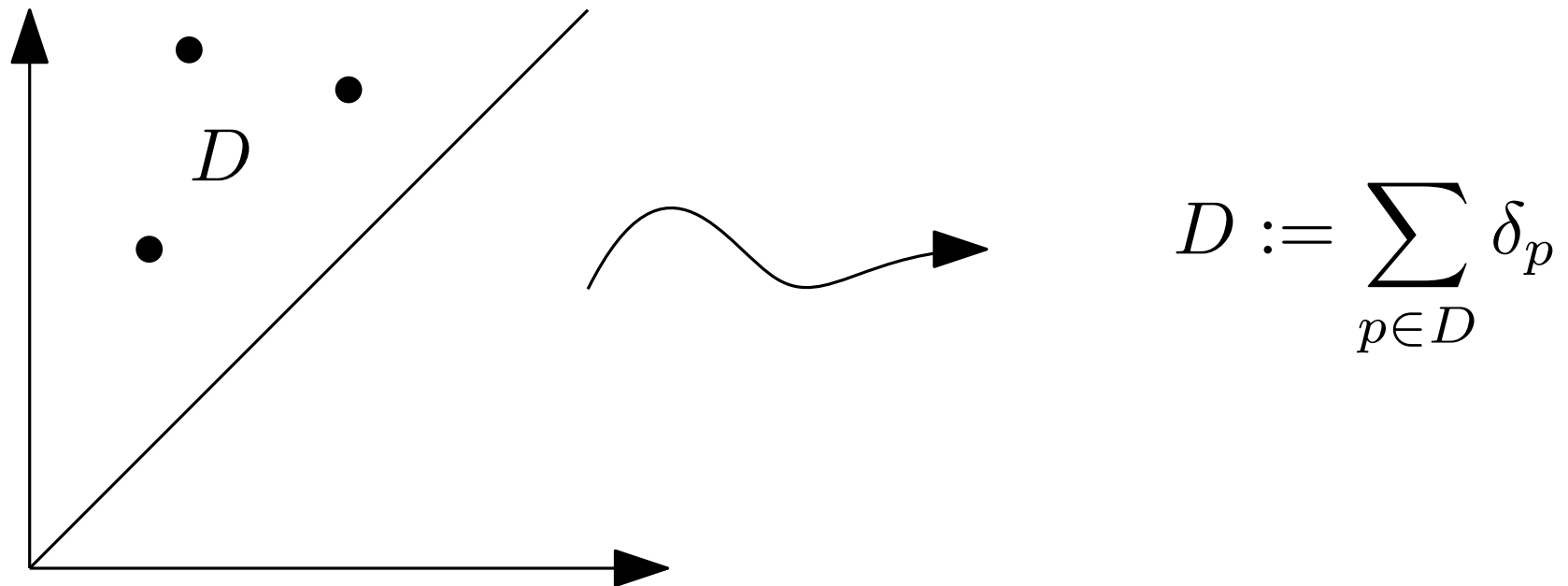
Objective: Arrhythmia detection from ECG data.



- Improvement over state-of-the-art.
- Better generalization.

	Accuracy[%]
UCLA (2018)	93.4
Li et al. (2016)	94.6
Inria-Fujitsu (2018)*	98.6

Persistence diagrams as discrete measures



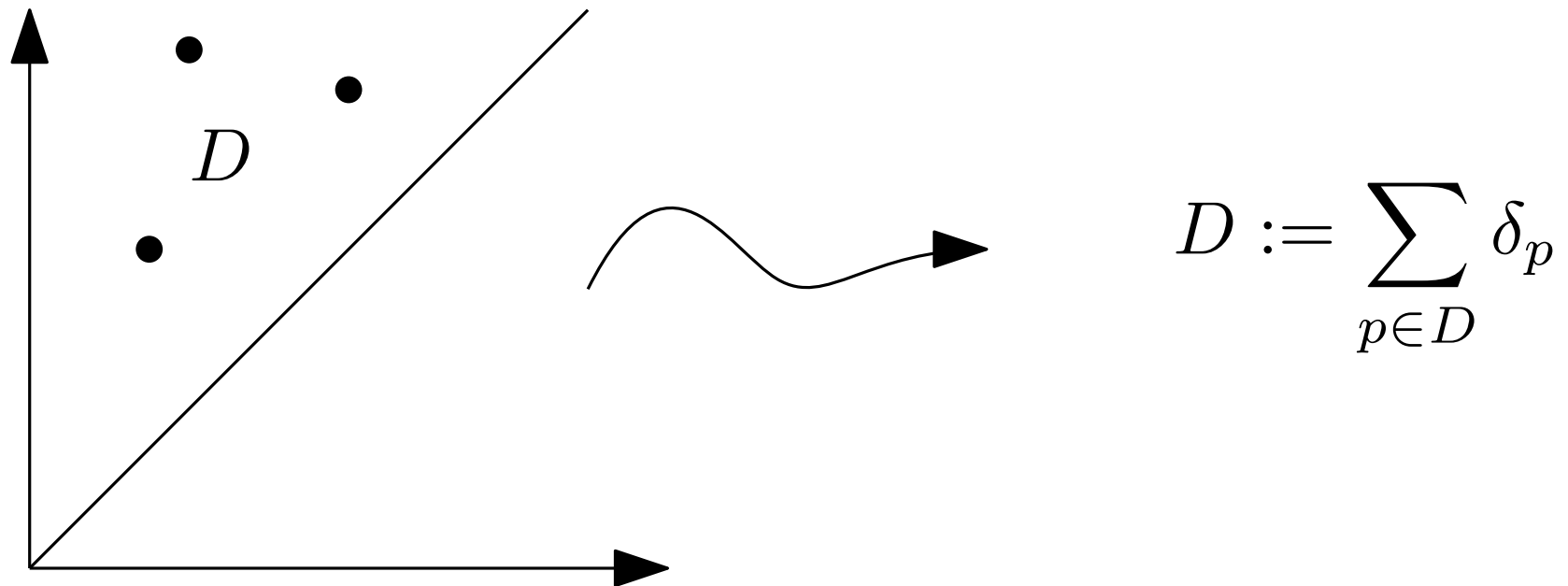
Motivations:

- The space of measures is much nicer than the space of P. D. !
- In the general algebraic persistence theory, persistence diagrams naturally appear as discrete measures in the plane. [C., de Silva, Glisse, Oudot 16]
- Many persistence representations can be expressed as linear representations

$$D(f) = \sum_{p \in D} f(p) = \int f dD$$

for well-chosen functions $f : \mathbb{R}^2 \rightarrow \mathcal{H}$.

Persistence diagrams as discrete measures

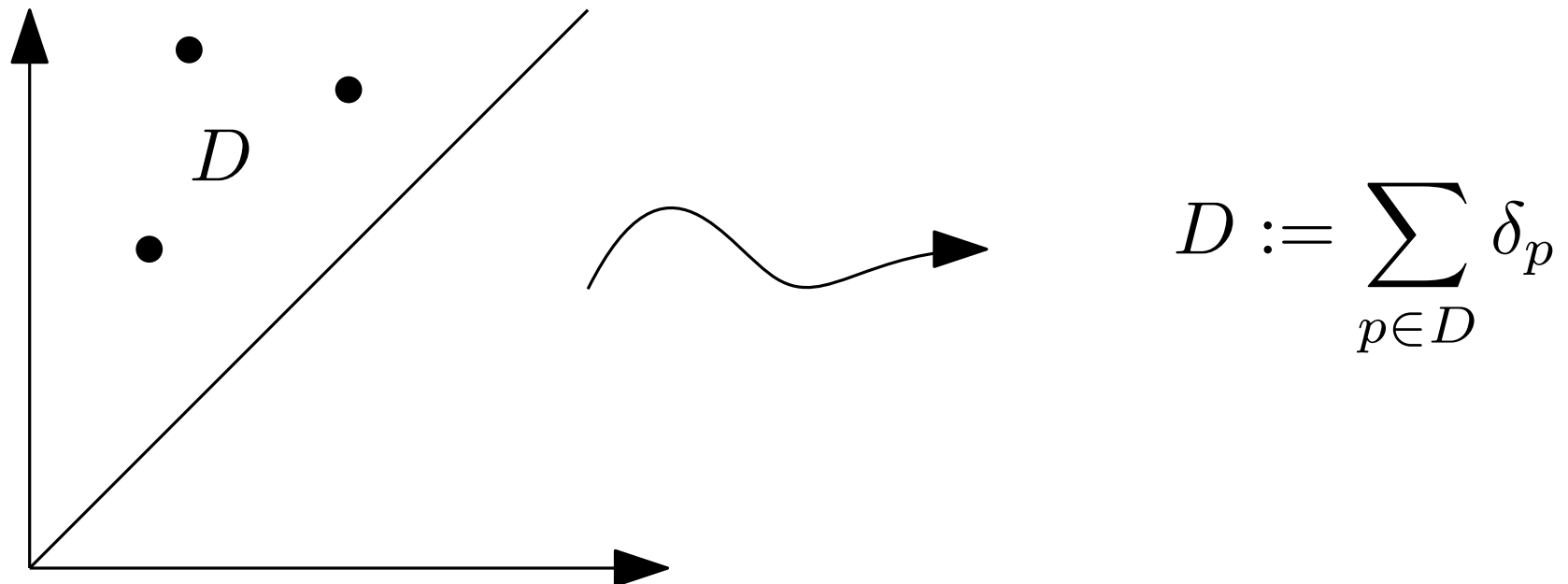


Benefits:

- Interesting statistical properties
- Data-driven selection of well-adapted representations (supervised and unsupervised, coming with guarantees)
- Optimisation of persistence-based functions

Many tools available and implemented in the GUDHI library

Persistence diagrams as discrete measures



Example/exercise:

- Persistence images are linear representation of persistence diagrams
- Betti curves are linear representation of persistence diagrams
- Imagine other representations
- Are persistence landscapes linear representations?

A zoo of representations of persistence

(non exhaustive list - see also Gudhi representations)

- Collections of 1D functions
 - landscapes [Bubenik 2012]
 - Betti curves [Umeda 2017]
- discrete measures point of view: (interesting statistical properties [C., Divoi 2018])
 - persistence images [Adams et al 2017]
 - ATOL [C. et al 2021]
 - convolution with Gaussian kernel [Reininghaus et al. 2015] [Chepushtanova et al. 2015] [Kusano Fukumisu Hiraoka 2016-17] [Le Yamada 2018]
 - sliced on lines [Carrière Oudot Cuturi 2017]
- finite metric spaces [Carrière Oudot Ovsjanikov 2015]
- polynomial roots or evaluations [Di Fabio Ferri 2015] [Kališnik 2016]

A zoo of representations of persistence

(non exhaustive list - see also Gudhi representations)

- Collections of 1D functions
 - landscapes [Bubenik 2012]
 - Betti curves [Umeda 2017]
- discrete measures point of view: (interesting statistical properties [C., DivoI 2018])
 - persistence images [Adams et al 2017]
 - ATOL [C. et al 2021]
 - convolution with Gaussian kernel [Reininghaus et al. 2015] [Chepushtanova et al. 2015] [Kusano Fukumisu Hiraoka 2016-17] [Le Yamada 2018]
 - sliced on lines [Carrière Oudot Cuturi 2017]
- finite metric spaces [Carrière Oudot Ovsjanikov 2015]
- polynomial roots or evaluations [Di Fabio Ferri 2015] [Kališnik 2016]

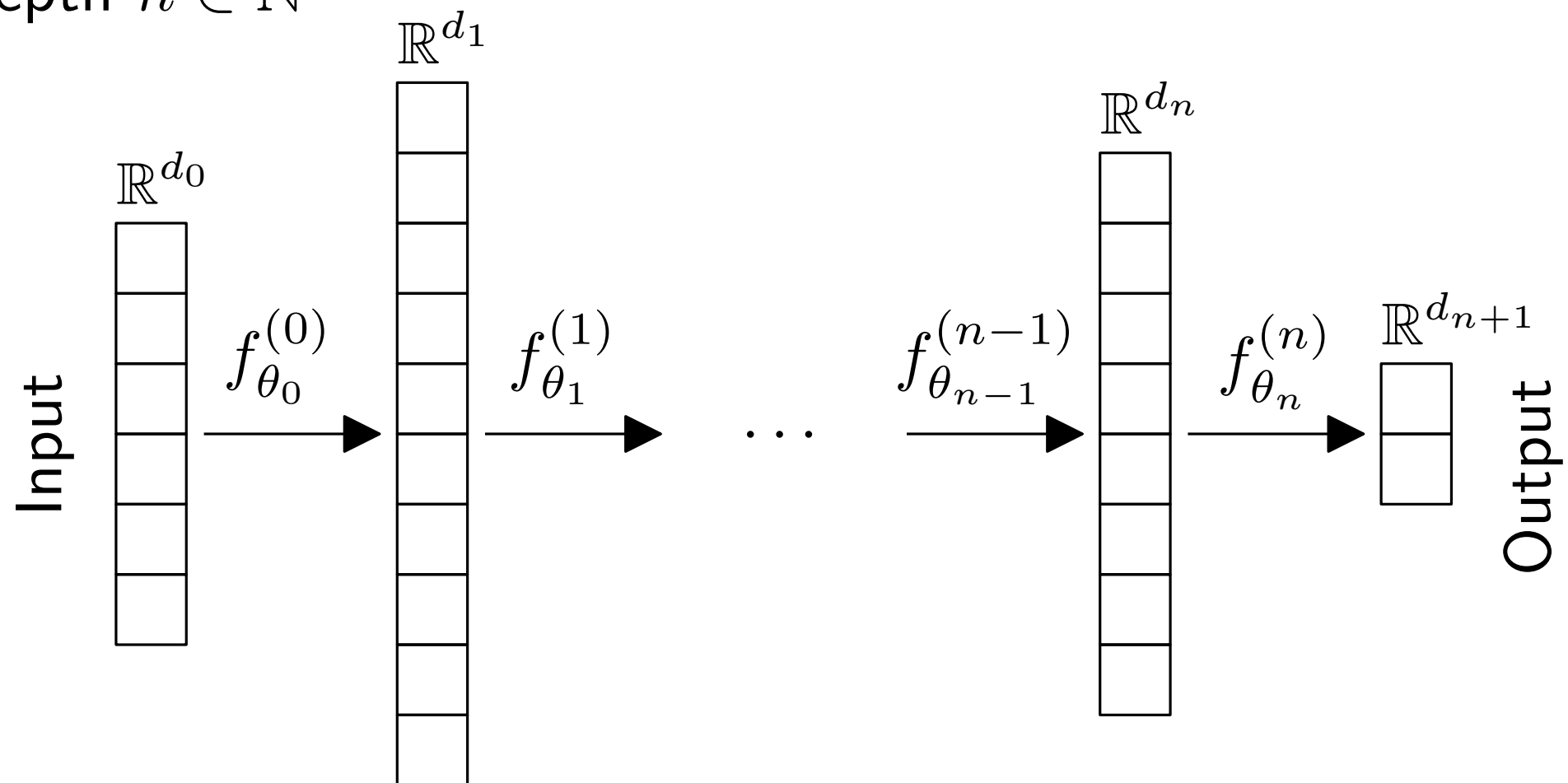
Problem: How to chose the right representation?

Supervised learning of persistence representations: the example of PersLay

- Learn, from training data, a well-suited persistence representation for a given learning task.
- PersLay: a neural network architecture to learn persistence representations.

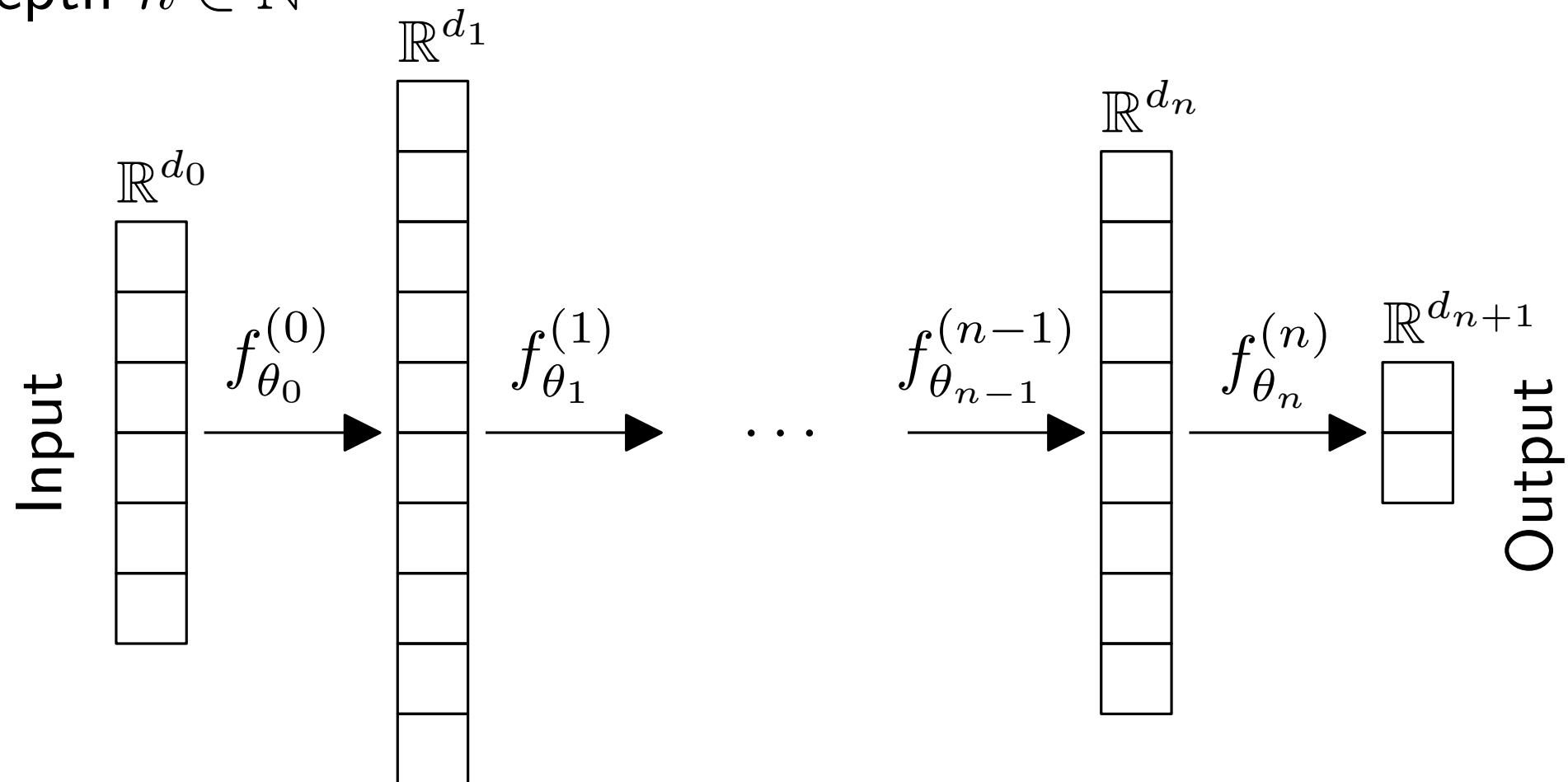
Reminder: Neural Net

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



Reminder: Neural Net

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



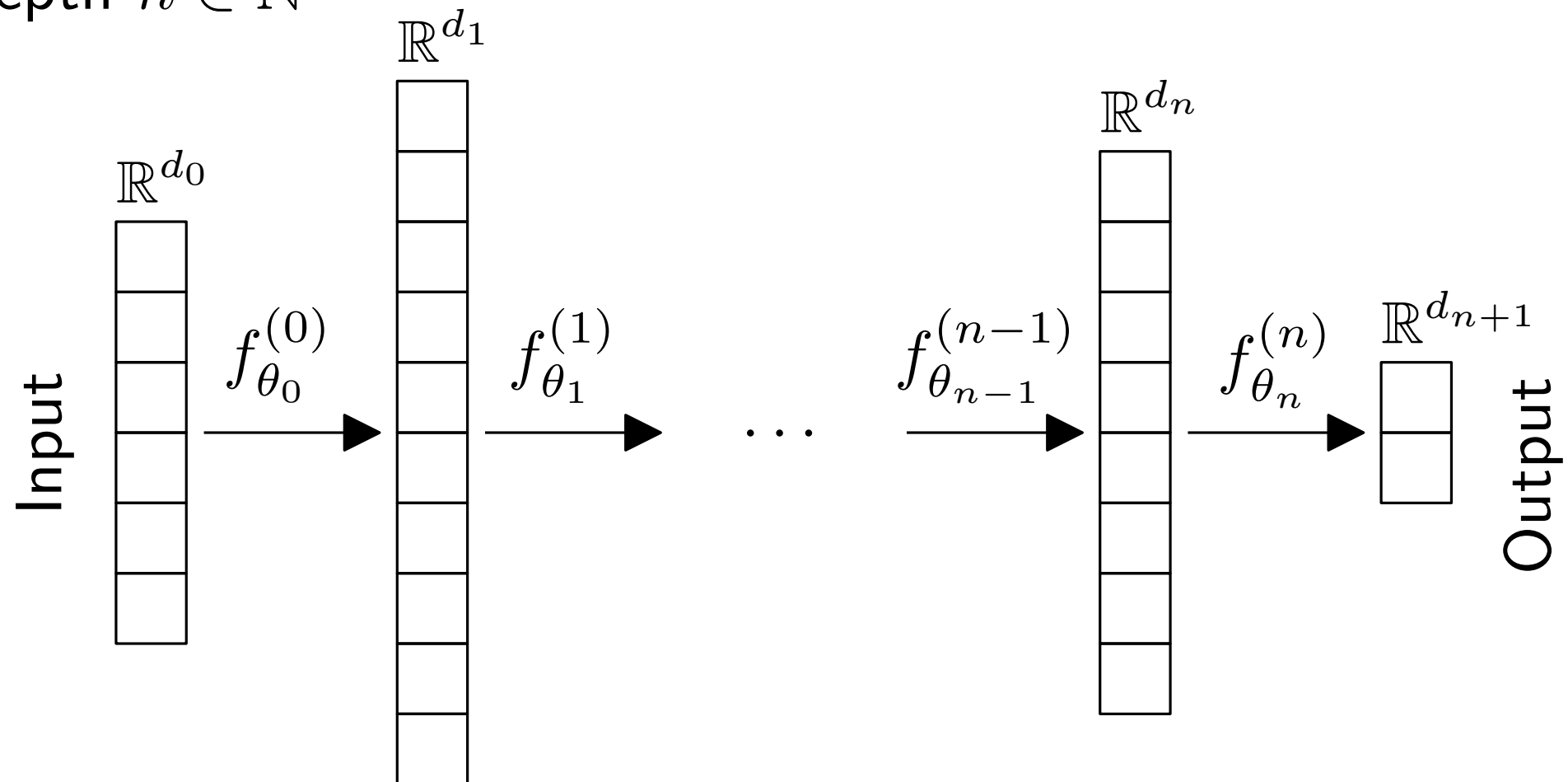
$$\theta_k = (W_k \in \mathbb{R}^{d_{k+1} \times d_k}, b_k \in \mathbb{R}^{d_{k+1}}), \quad \sigma : x \mapsto \max(0, x) \text{ or } (1 + e^{-x})^{-1}$$

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

$$\text{Final classifier: } F_\theta = f_{\theta_n}^{(n)} \circ \dots \circ f_{\theta_0}^{(0)}$$

Reminder: Neural Net

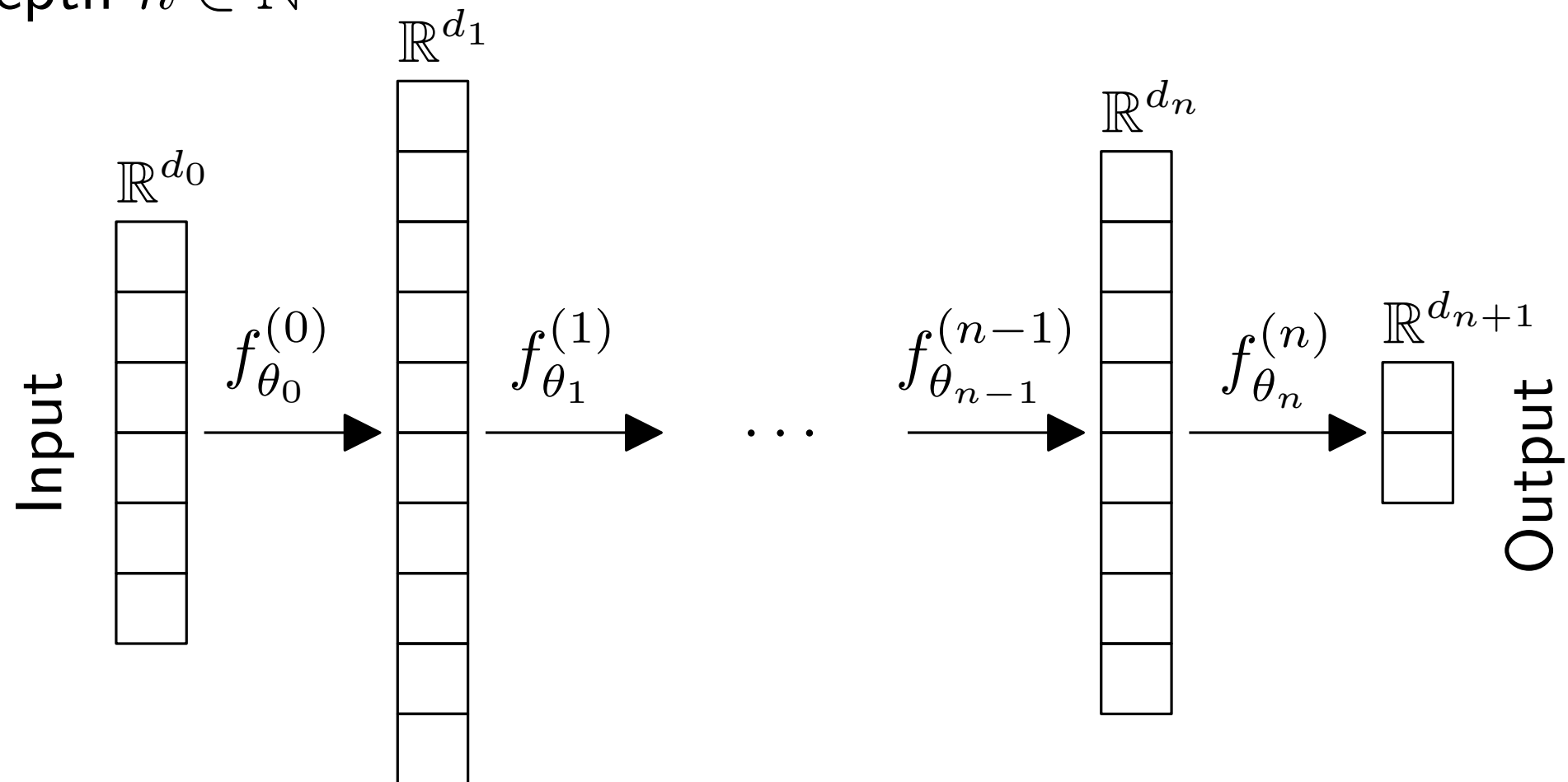
NN with depth $n \in \mathbb{N}^*$



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

Reminder: Neural Net

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



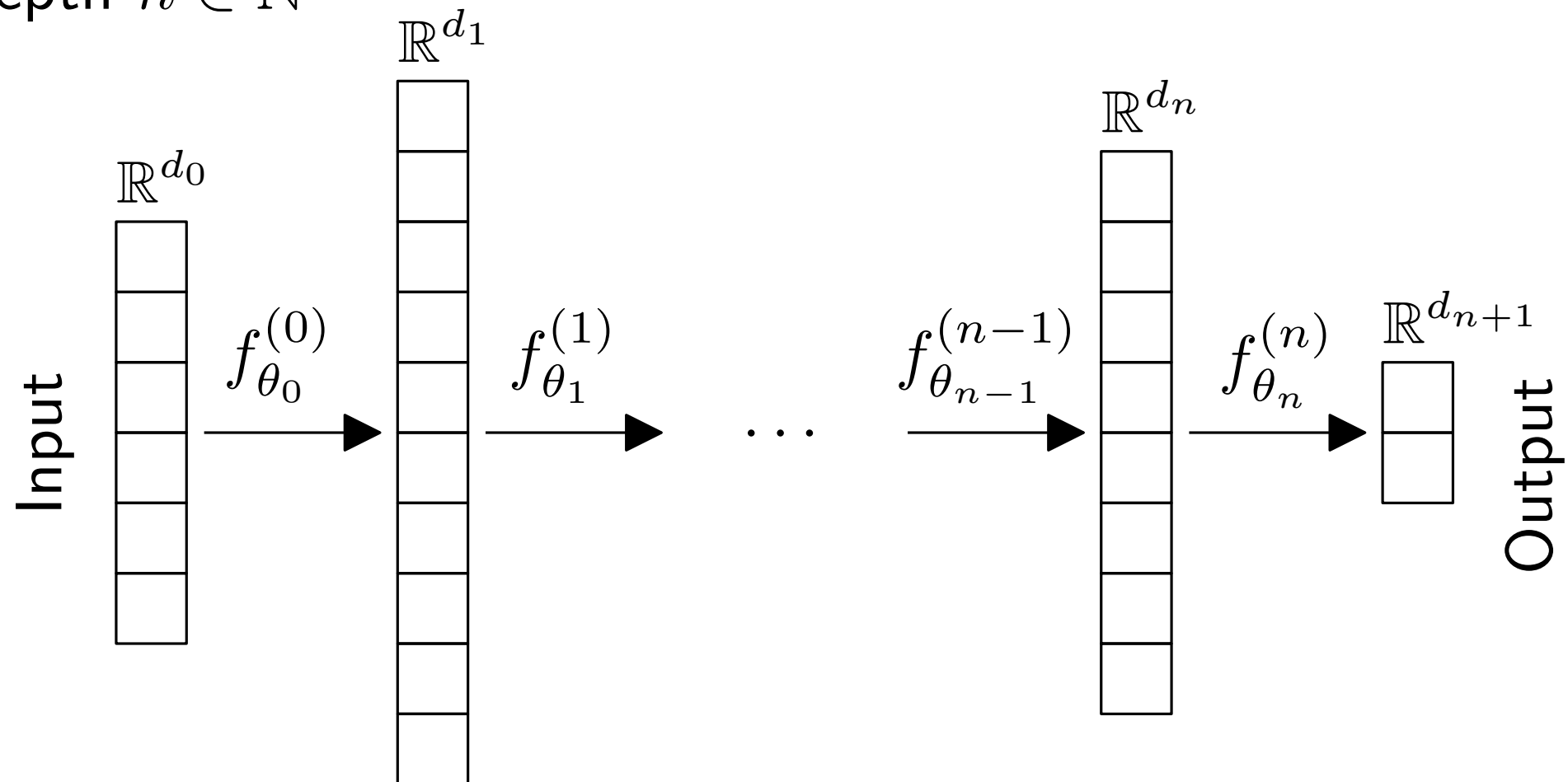
Goal: Minimize $\ell(\theta) = \sum_i \|f_{\theta}(x_i) - y_i\|_2^2$ w.r.t. θ

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 Net

NN 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: $f_{\theta_k}^{(k)}$ needs to be **differentiable** w.r.t. θ_k and x

Deep Set Architecture

Originally defined in [Zaheer et al. 2017]

Tailored to handle sets instead of finite dimensional vectors

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

Deep Set Architecture

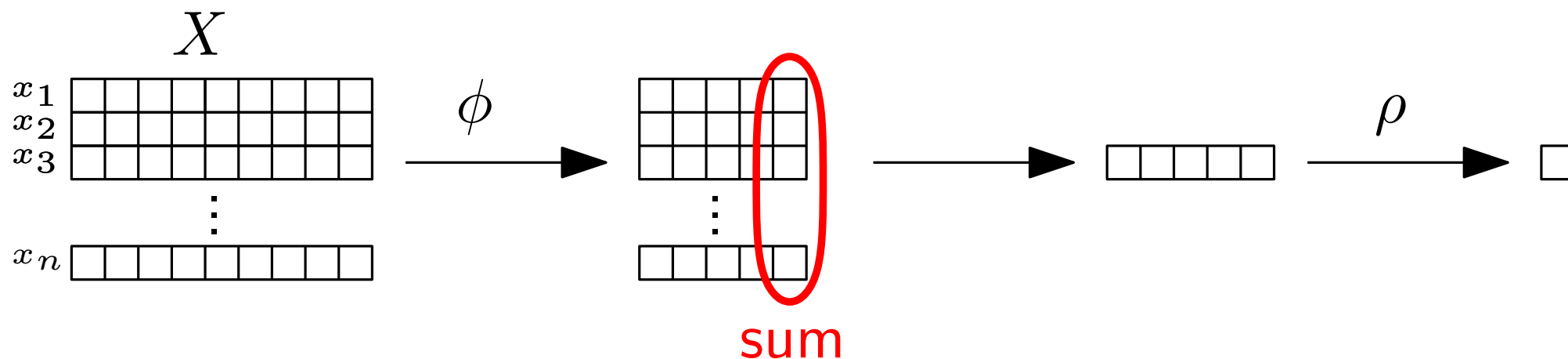
Originally defined in [Zaheer et al. 2017]

Tailored to handle sets instead of finite dimensional vectors

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

Network is *permutation invariant*: $F(X) = \rho(\sum_i \phi(x_i))$

$$\Rightarrow F(\{x_1, \dots, x_n\}) = F(\{x_{\sigma(1)}, \dots, x_{\sigma(n)}\}), \forall \sigma$$



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

Deep Set Architecture

Originally defined in [Zaheer et al. 2017]

Tailored to handle sets instead of finite dimensional vectors

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

Network is *permutation invariant*: $F(X) = \rho(\sum_i \phi(x_i))$

Universality theorem

Th: [Zaheer et al. 2017]

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

Deep Set Architecture

Originally defined in [Zaheer et al. 2017]

Tailored to handle sets instead of finite dimensional vectors

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

Deep Set Architecture

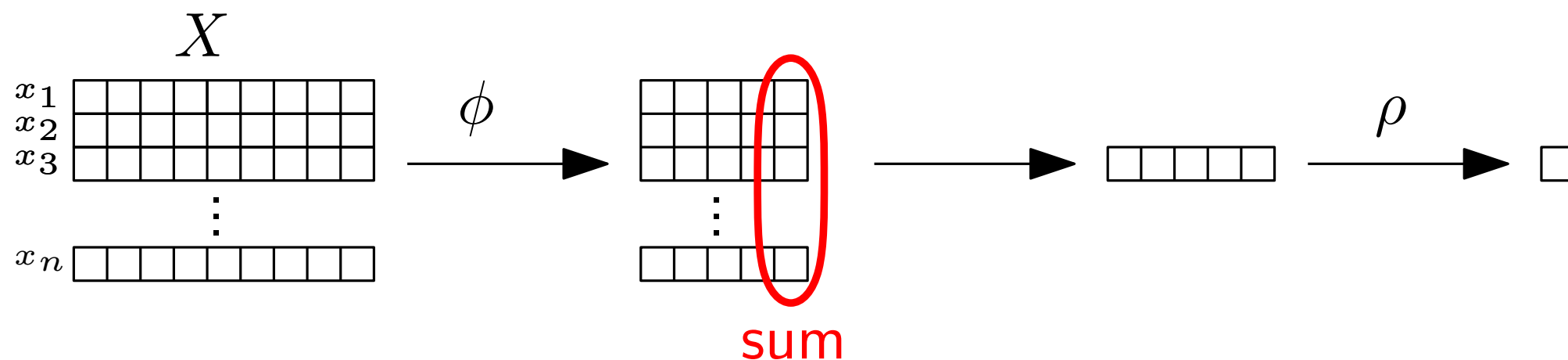
Originally defined in [Zaheer et al. 2017]

Tailored to handle sets instead of finite dimensional vectors

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

Network is *permutation invariant*: $F(X) = \rho(\sum_i \phi(x_i))$

$$\Rightarrow F(\{x_1, \dots, x_n\}) = F(\{x_{\sigma(1)}, \dots, x_{\sigma(n)}\}), \forall \sigma$$



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

Deep Set Architecture

Originally defined in [Zaheer et al. 2017]

Tailored to handle sets instead of finite dimensional vectors

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

Network is *permutation invariant*: $F(X) = \rho(\sum_i \phi(x_i))$

Universality theorem

Th: [Zaheer et al. 2017]

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

Deep Set Architecture

Originally defined in [Zaheer et al. 2017]

Tailored to handle sets instead of finite dimensional vectors

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

Deep Set Architecture

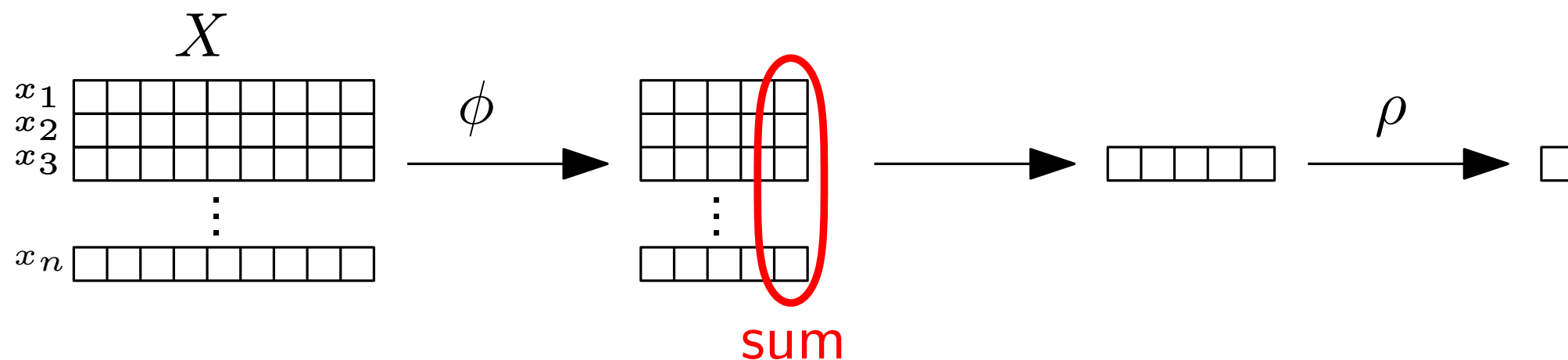
Originally defined in [Zaheer et al. 2017]

Tailored to handle sets instead of finite dimensional vectors

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

Network is *permutation invariant*: $F(X) = \rho(\sum_i \phi(x_i))$

$$\Rightarrow F(\{x_1, \dots, x_n\}) = F(\{x_{\sigma(1)}, \dots, x_{\sigma(n)}\}), \forall \sigma$$



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

Deep Set Architecture

Originally defined in [Zaheer et al. 2017]

Tailored to handle sets instead of finite dimensional vectors

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

Network is *permutation invariant*: $F(X) = \rho(\sum_i \phi(x_i))$

Universality theorem

Th: [Zaheer et al. 2017]

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

Adaptation to persistence diagrams: PersLay

[Carrière et al 2019]

Permutation invariant layers generalize several TDA approaches

→ persistence images → silhouettes → Betti curves

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

$$\text{PersLay}(\text{dgm}) = \rho(\text{op}\{w(p) \cdot \phi(p)\}_{p \in \text{dgm}})$$

Permutation-invariant
operation

Weight function

Point transformation
 $\phi : \mathbb{R}^2 \rightarrow \mathbb{R}^k$

https://gudhi.inria.fr/python/latest/representations_tflow_itf_ref.html (Available through Gudhi)

Adaptation to persistence diagrams: PersLay

[Carrière et al 2019]

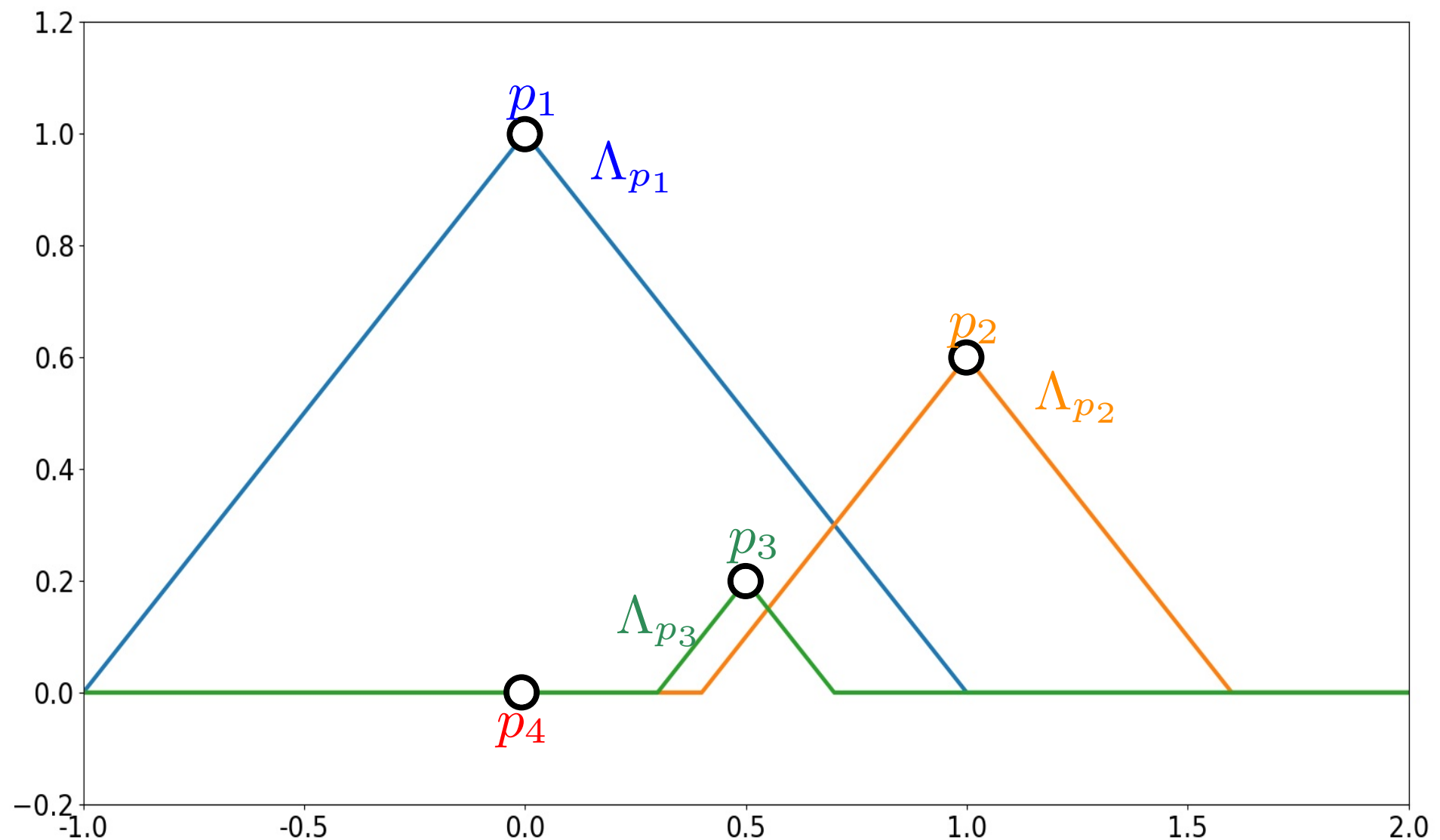
Parameters $t_1, \dots, t_q \in \mathbb{R}$

$$w(p) = 1$$

$$\phi_\Lambda : p \mapsto$$

$$\begin{bmatrix} \Lambda_p(t_1) \\ \Lambda_p(t_2) \\ \vdots \\ \Lambda_p(t_q) \end{bmatrix}$$

op = top- k



Persistence landscape

Adaptation to persistence diagrams: PersLay

[Carrière et al 2019]

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

$$w(p) = w_t((x, y))$$

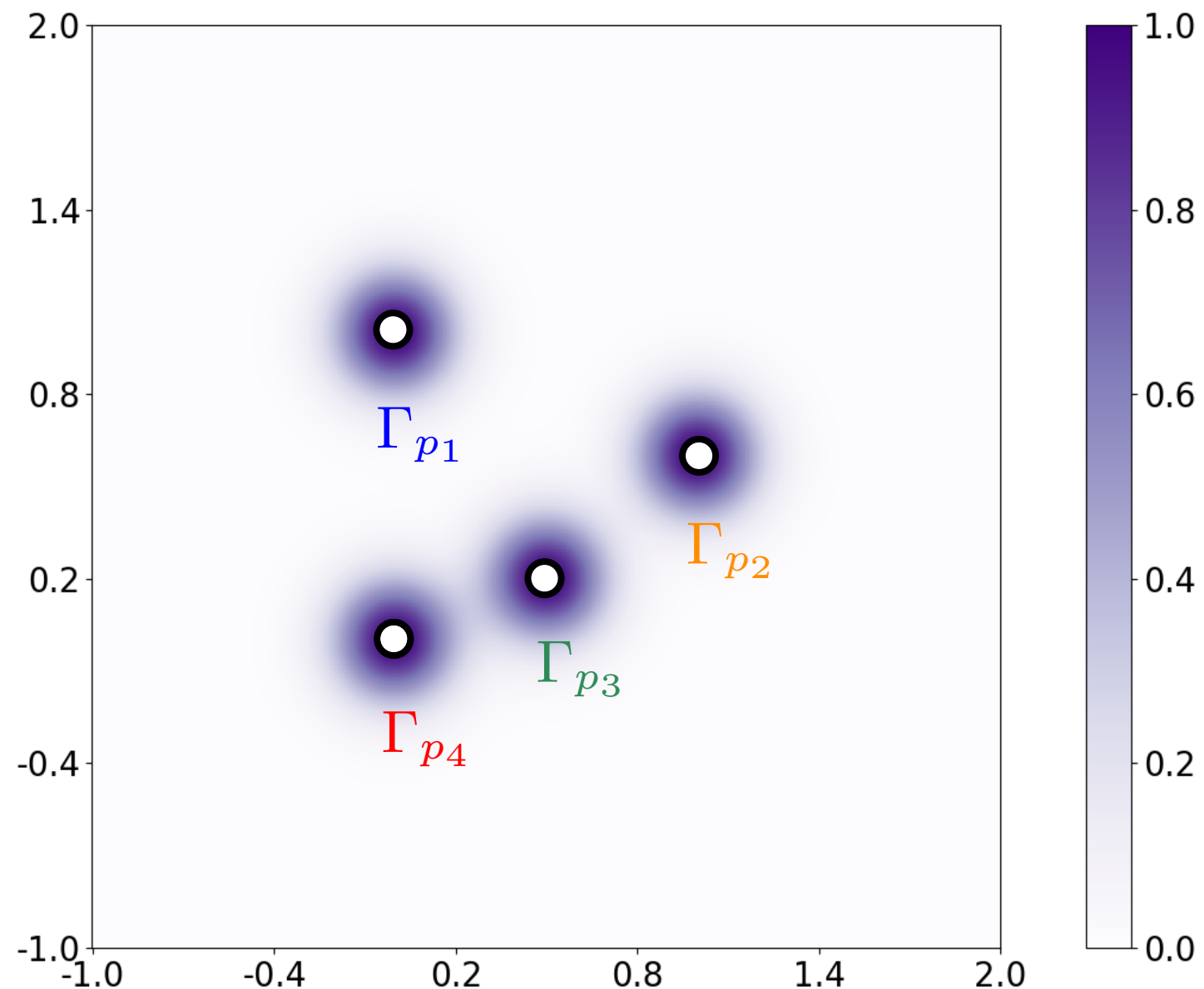
$$\phi_\Gamma : p \mapsto$$

$$\begin{bmatrix} \Gamma_p(t_1) \\ \Gamma_p(t_2) \\ \vdots \\ \Gamma_p(t_q) \end{bmatrix}$$

op = sum

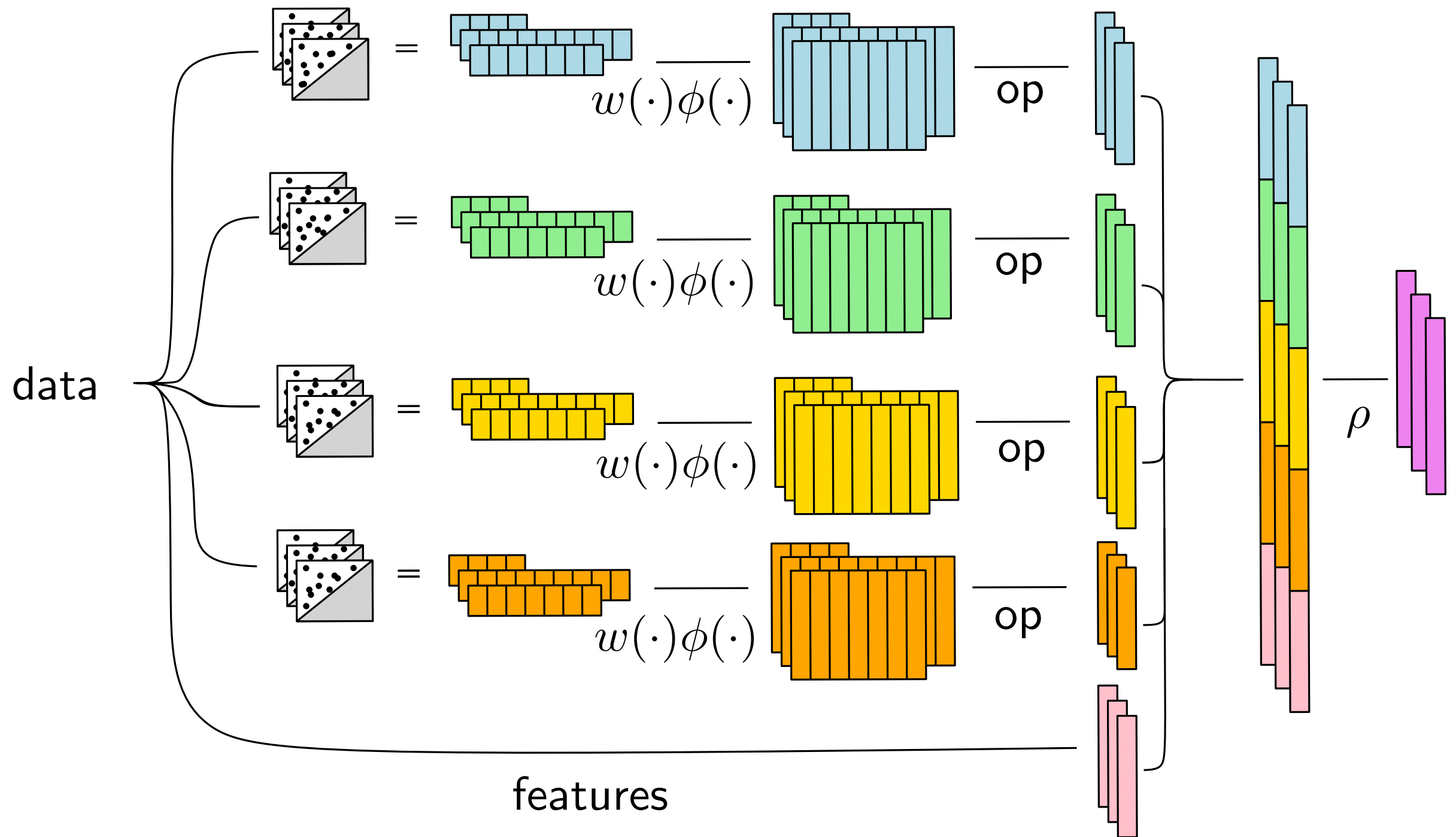
$$\Gamma_p : t \mapsto \exp(-\|p - t\|_2^2 / (2\sigma^2))$$

Persistence surface



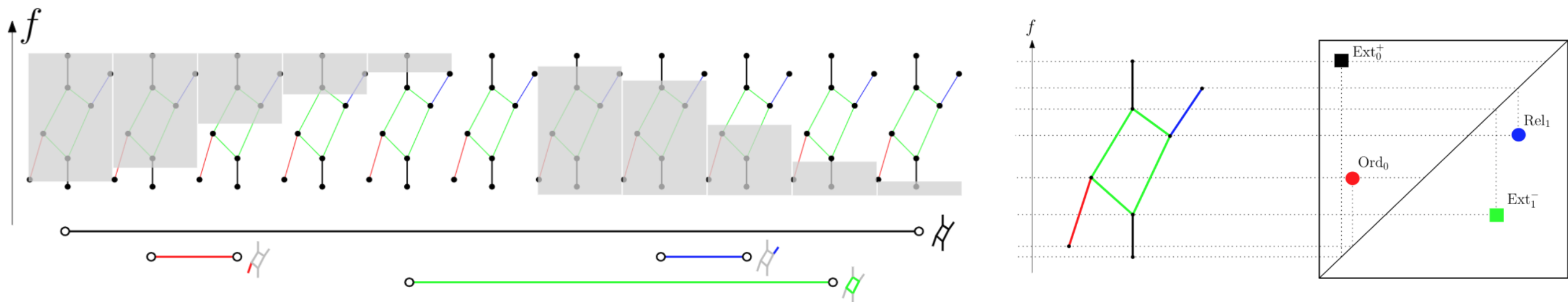
Adaptation to persistence diagrams: PersLay

[Carrière et al 2019]



Adaptation to persistence diagrams: PersLay

[Carrière et al 2019]



Dataset	ScaleVariant ¹	RetGK1 * ²	RetGK11 * ²	FGSD ³	GCNN ⁴	Spectral + HKS ⁵	PersLay
REDDIT5K	—	56.1(±0.5)	55.3(±0.3)	47.8	52.9	49.7(±0.3)	56.6(±0.3)
REDDIT12K	—	48.7(±0.2)	47.1(±0.3)	—	46.6	39.7(±0.1)	47.7(±0.2)
COLLAB	—	81.0(±0.3)	80.6(±0.3)	80.0	79.6	67.8(±0.2)	76.4(±0.4)
IMDB-B	72.9	71.9(±1.0)	72.3(±0.6)	73.6	73.1	67.6(±0.6)	70.9(±0.7)
IMDB-M	50.3	47.7(±0.3)	48.7(±0.6)	52.4	50.3	44.5(±0.4)	48.7(±0.6)
BZR *	86.6	—	—	—	—	80.8(±0.8)	87.2(±0.7)
COX2 *	78.4	80.1(±0.9)	81.4(±0.6)	—	—	78.2(±1.3)	81.6(±1.0)
DHFR *	78.4	81.5(±0.9)	82.5(±0.8)	—	—	69.5(±1.0)	81.8(±0.8)
MUTAG *	88.3	90.3(±1.1)	90.1(±1.0)	92.1	86.7	85.8(±1.3)	89.8(±0.9)
PROTEINS *	72.6	75.8(±0.6)	75.2(±0.3)	73.4	76.3	73.5(±0.3)	74.8(±0.3)
NCI1 *	71.6	84.5(±0.2)	83.5(±0.2)	79.8	78.4	65.3(±0.2)	72.8(±0.3)
NCI109 *	70.5	—	—	78.8	—	64.9(±0.2)	71.7(±0.3)
FRANKENSTEIN	69.4	—	—	—	—	62.9(±0.1)	70.7(±0.4)

Average scores from 10 times 10-folds
cross-validation

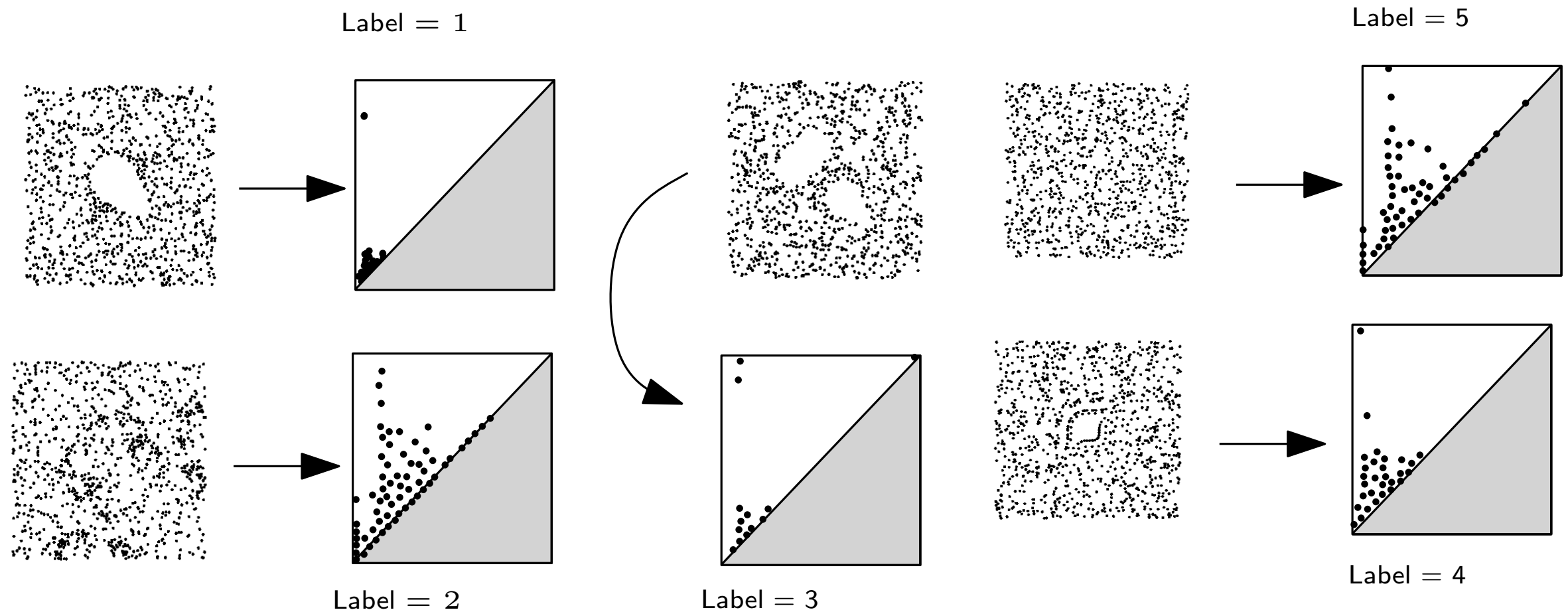
Adaptation to persistence diagrams: PersLay

[Carrière et al 2019]

Goal: classify orbits of *linked twisted map*

Orbits described by (depending on parameter r):

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

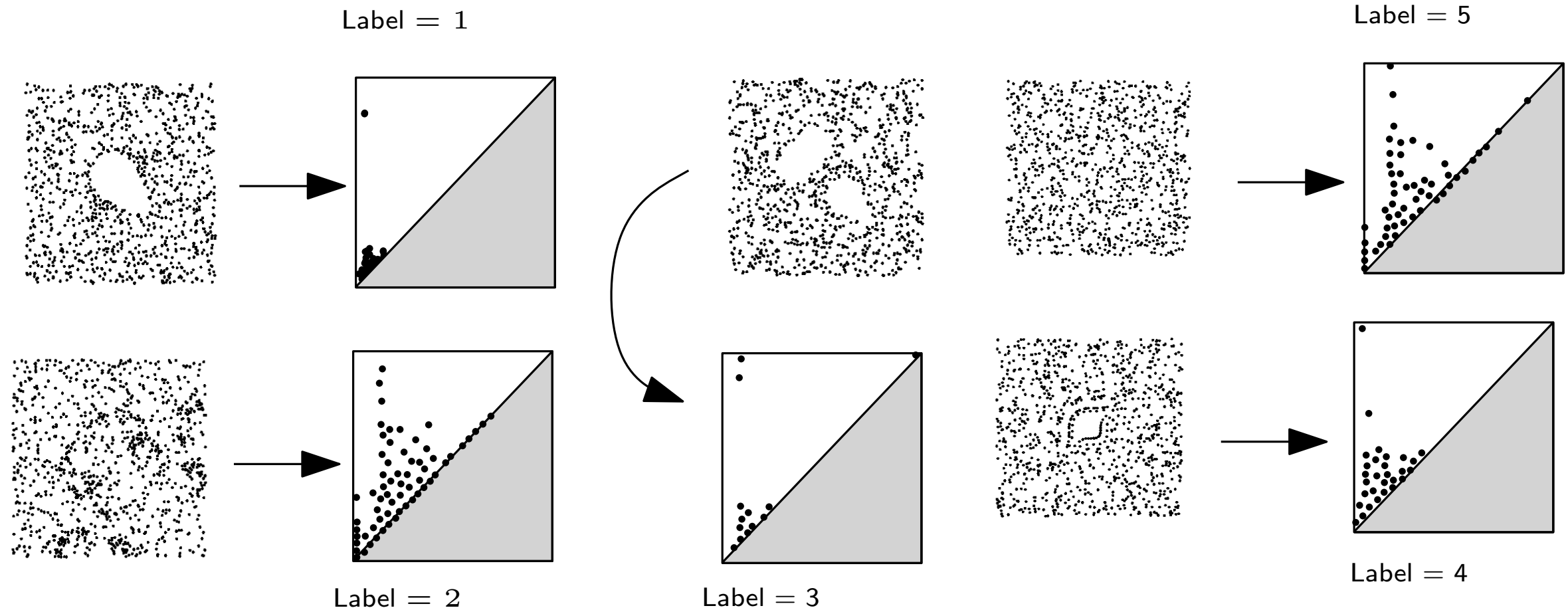


Adaptation to persistence diagrams: PersLay

[Carrière et al 2019]

Goal: classify orbits of *linked twisted map*

Dataset	PSS-K	PWG-K	SW-K	PF-K	PersLay
ORBIT5K	72.38(± 2.4)	76.63(± 0.7)	83.6(± 0.9)	85.9(± 0.8)	87.7(± 1.0)
ORBIT100K	—	—	—	—	89.2(± 0.3)

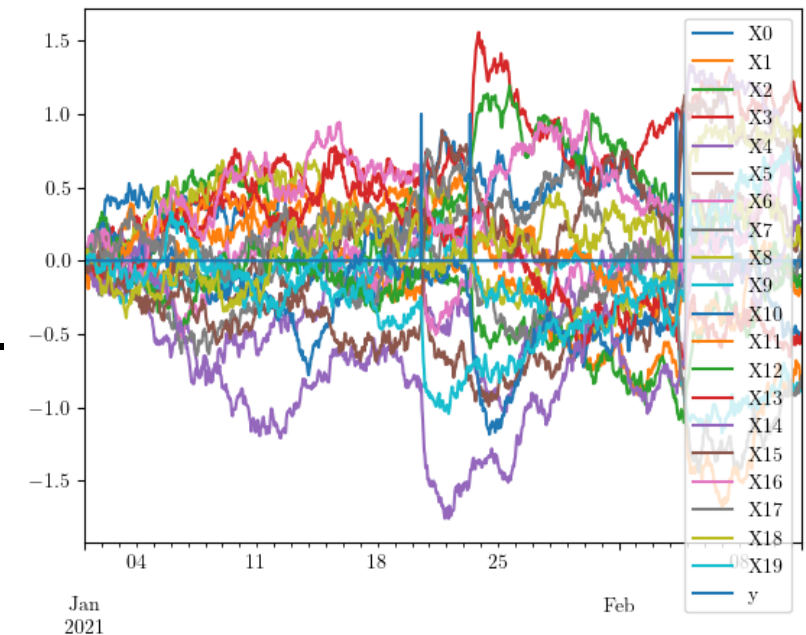


Unsupervised learning of persistence representations: the example of ATOL

- Learn, from a distribution of persistence diagrams, a well-suited persistence representation.
- ATOL: a simple and efficient data-driven method (inspired from the classical k -means algorithm) to learn persistence representations.
- Application: Anomaly detection in multivariate time series.

Unsupervised learning of linear representations of PD through a concrete example

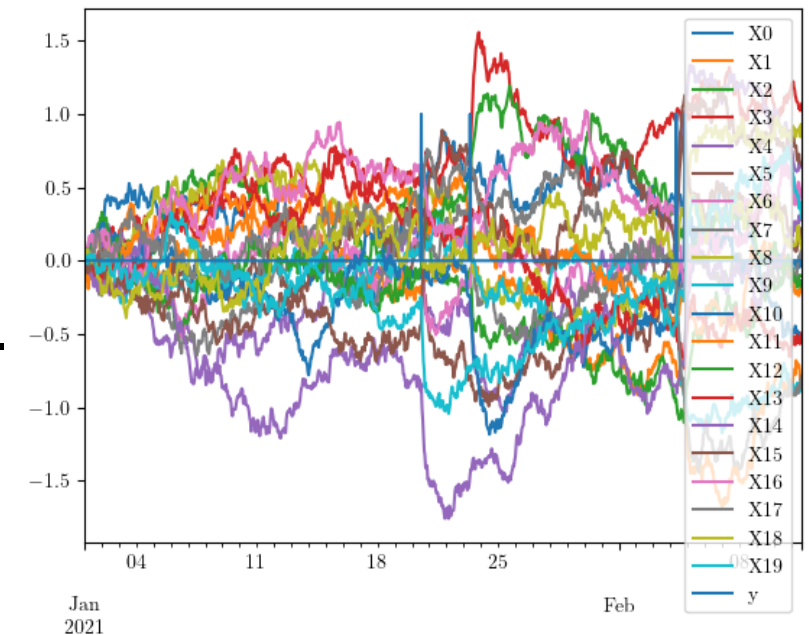
Data: A multivariate time series $(Y_t)_{t \in [0, L]} \in \mathbb{R}^D \times [0, L]$
e.g. given by a set of sensors monitoring a physical system.



General problem: Detect anomalies in (Y_t)

Unsupervised learning of linear representations of PD through a concrete example

Data: A multivariate time series $(Y_t)_{t \in [0, L]} \in \mathbb{R}^D \times [0, L]$
e.g. given by a set of sensors monitoring a physical system.



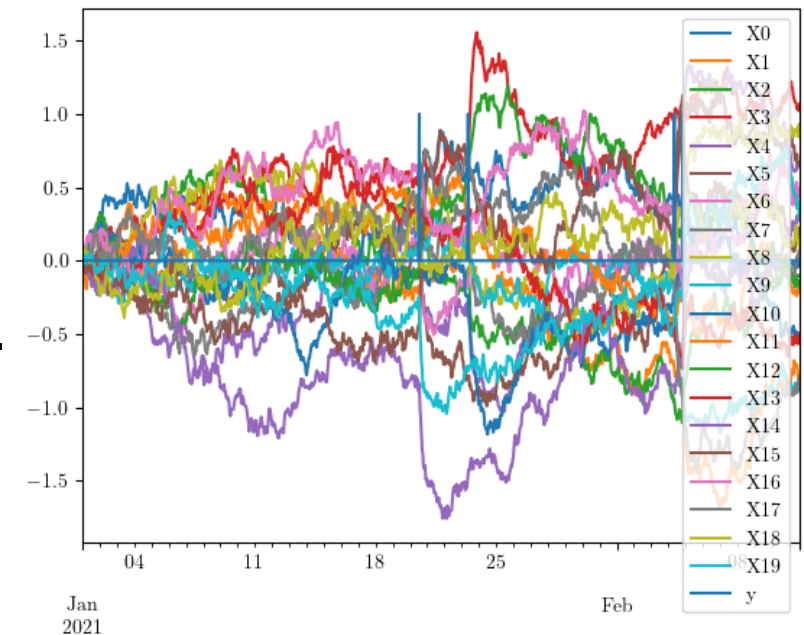
General problem: Detect anomalies in (Y_t)

→ A classical, not new, but practically important problem!

→ (Usually) Not a well-posed problem!

Unsupervised learning of linear representations of PD through a concrete example

Data: A multivariate time serie $(Y_t)_{t \in [0, L]} \in \mathbb{R}^D \times [0, L]$
e.g. given by a set of sensors monitoring a physical system.



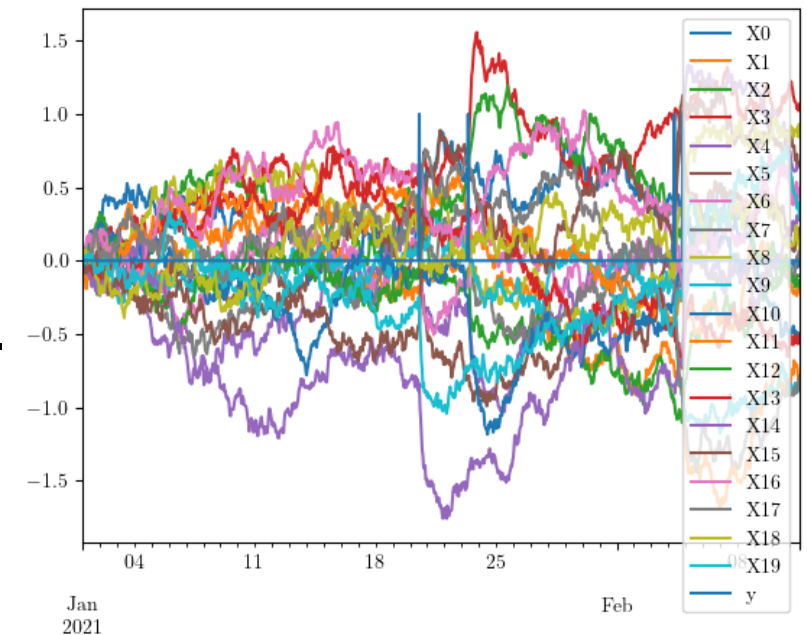
General problem: Detect anomalies in (Y_t)

Objective:

- A method to detect anomalies in the **global dependency structure** of the signal,
- based upon a **topological approach**,
- **flexible, easy to implement** and coming with (some) **mathematical guarantees**.

Unsupervised learning of linear representations of PD through a concrete example

Data: A multivariate time series $(Y_t)_{t \in [0, L]} \in \mathbb{R}^D \times [0, L]$
e.g. given by a set of sensors monitoring a physical system.



General problem: Detect anomalies in (Y_t)

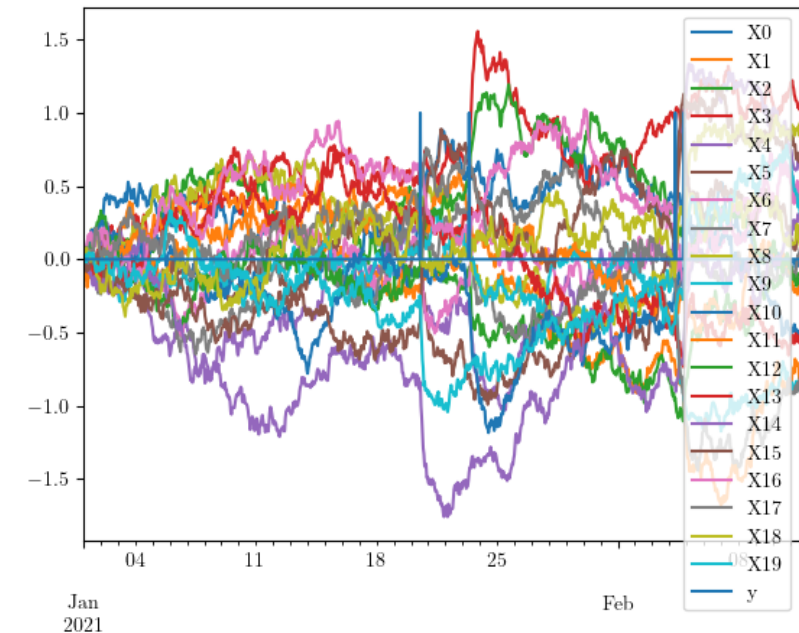
The overall pipeline:

1. Convert (Y_t) into a time dependent sequence of weighted graphs encoding the **dependency structure** between the components of (Y_t)
2. Extract (time-dependent) **topological information** from the sequence of graphs (**persistent homology**)
3. Unsupervised vectorization of topological information with **math. guarantees**.
4. Build a topological score function (classical stuff)

From multivariate time series to time-dependent graphs

Data: A multivariate time series $(Y_t)_{t \in [0, L]} \in \mathbb{R}^D \times [0, L]$

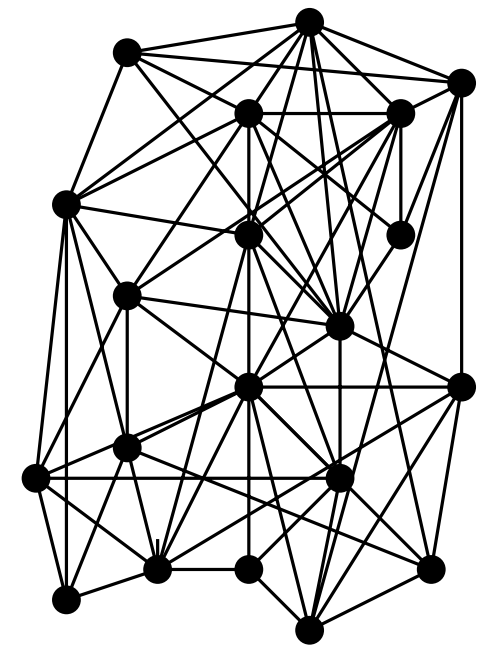
(Non-oriented) weighted graph G : a vertex set V and real valued edge weight function $s : V \times V \rightarrow \mathbb{R}$, $(v, v') \mapsto s_{v, v'}$, satisfying $s_{v, v'} := s_{v', v}$ for any pair of vertices (v, v') .



Input: Δ bandwidth, s stride

For t in $\llbracket 0, \lfloor (L - \Delta)/s \rfloor \rrbracket$, compute the weighted graph G_t with vertex set $V = \{v_0, \dots, v_{D-1}\}$, the components of (Y_t) , and weight matrix on the slice $[st, st + \Delta]$,

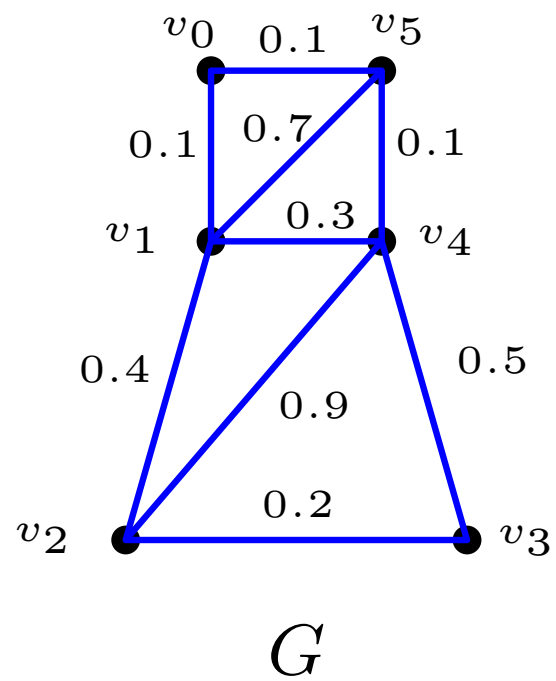
$$S_t = 1 - \text{Corr}(Y_{[st, st+\Delta]})$$



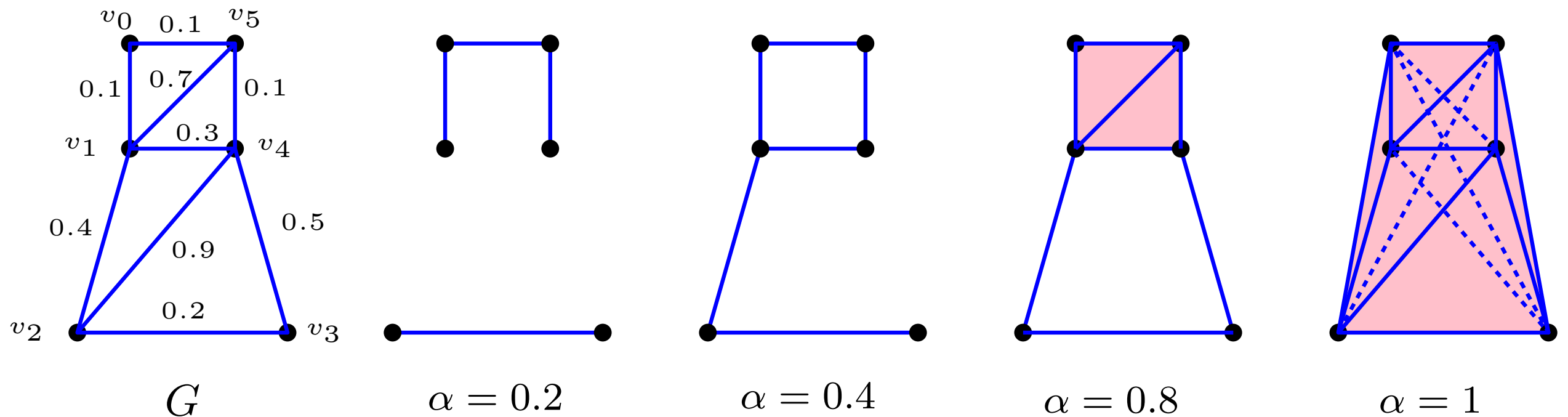
G_t encodes the correlations between the components of the time series

How to understand and exploit the global/topological structure of G_t ?

Filtrations associated to weighted graphs



Filtrations associated to weighted graphs

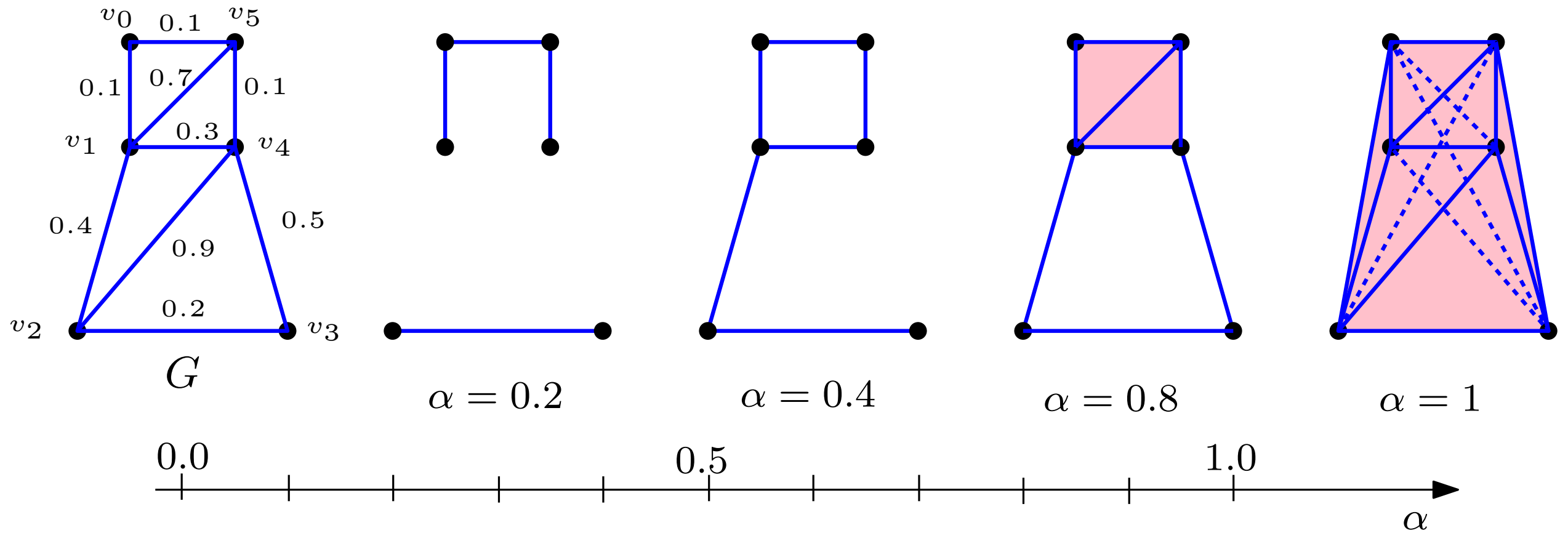


The [Vietoris-Rips filtration](#) associated to G is defined by

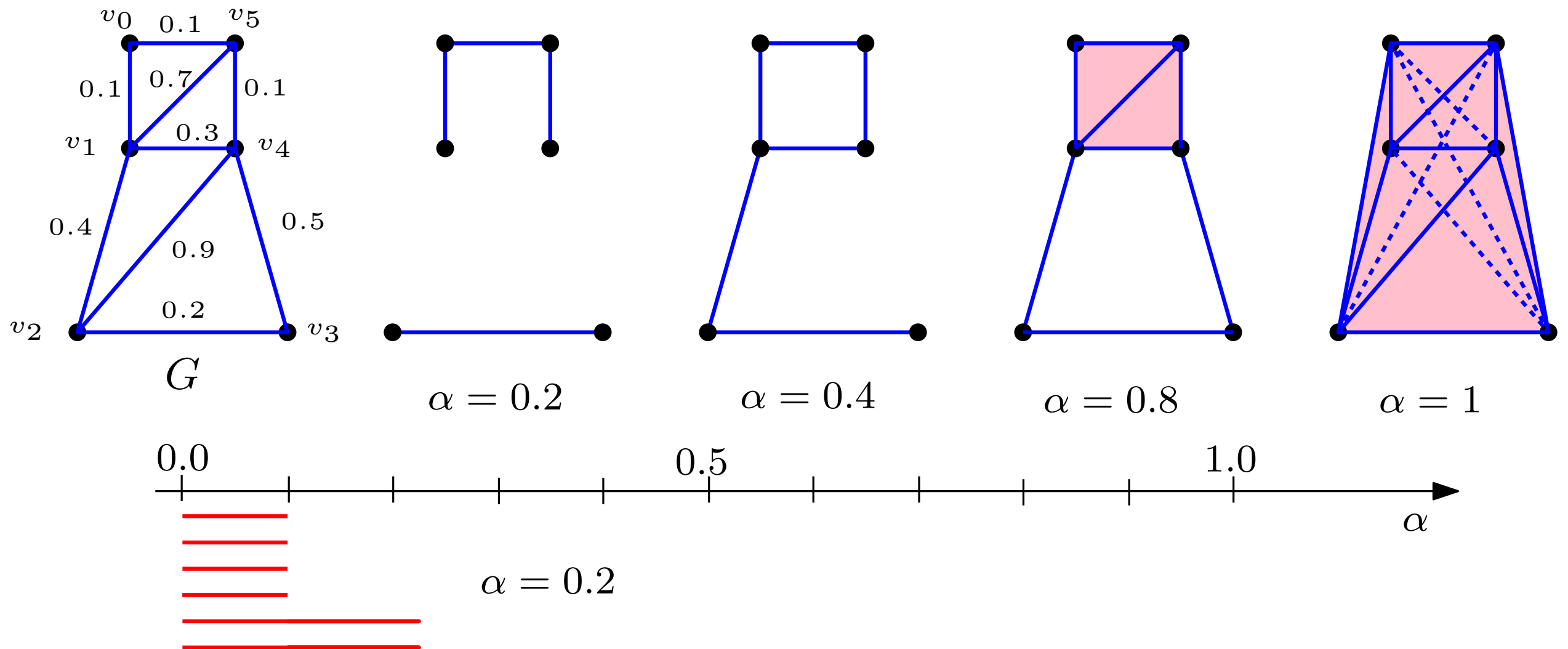
$$\sigma = [v_0, \dots, v_k] \in Rips_\alpha(G) \iff s_{v_i, v_j} \leq \alpha, \text{ for all } i, j \in \llbracket 0, k \rrbracket,$$

for $k > 1$, and $[v] \in Rips_\alpha(G)$ for any $v \in V$ and any $\alpha \geq 0$.

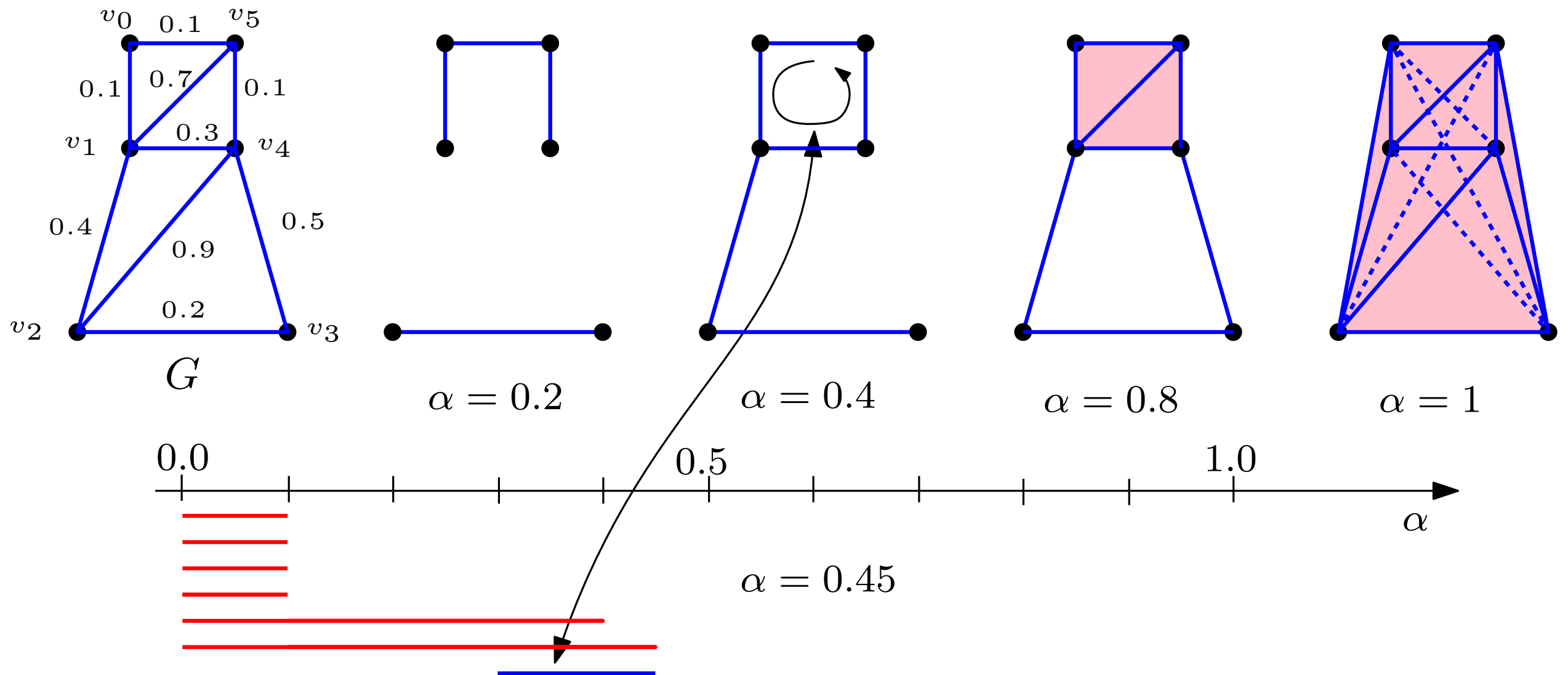
Persistent homology (in a nutshell)



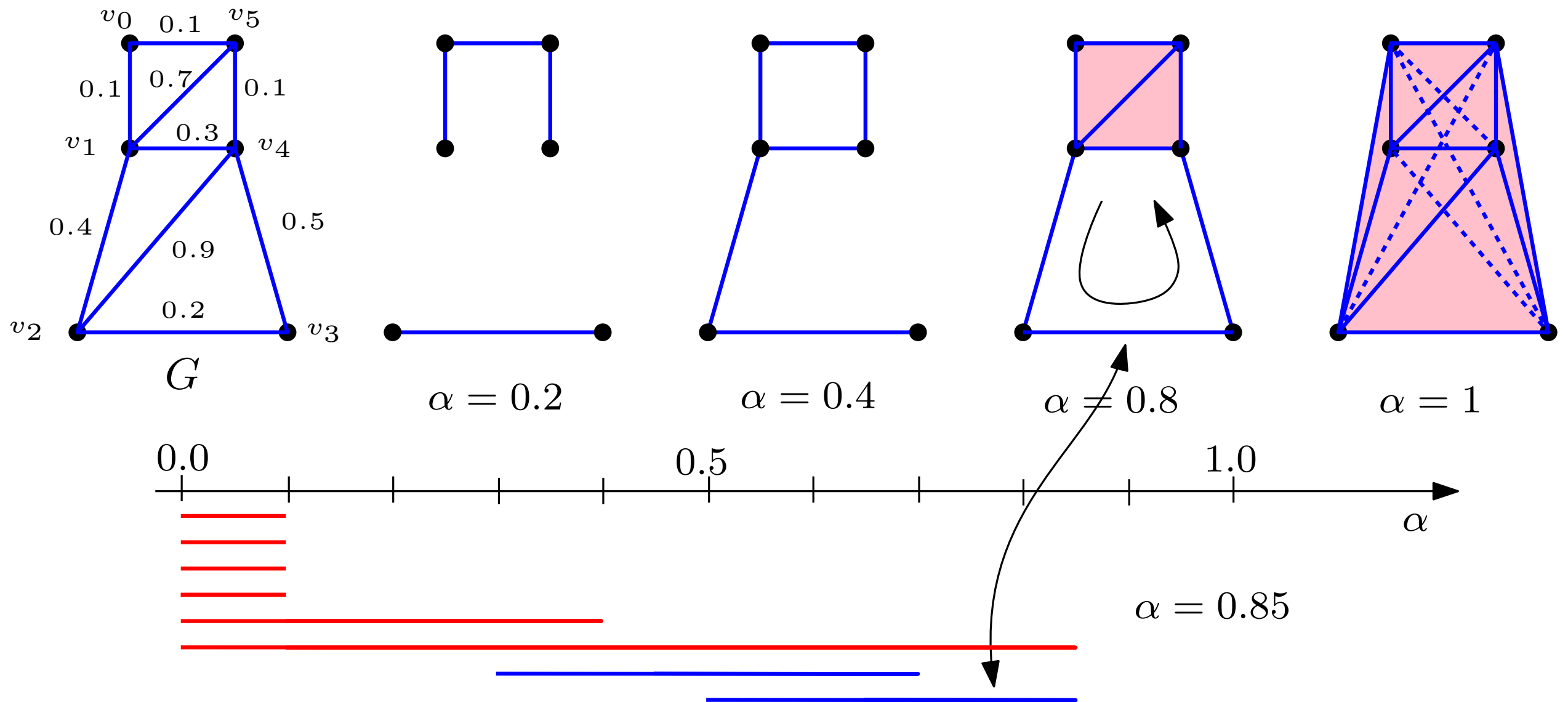
Persistent homology (in a nutshell)



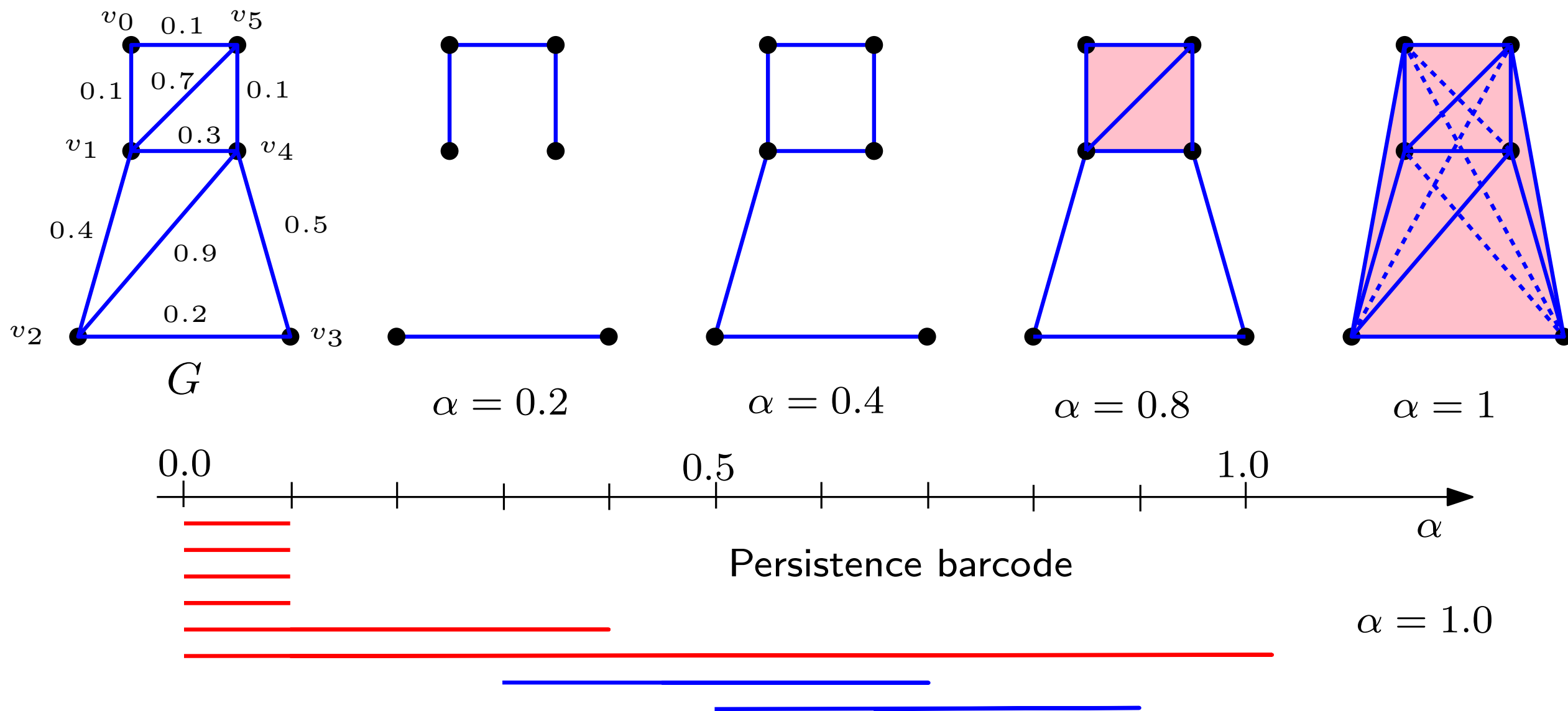
Persistent homology (in a nutshell)



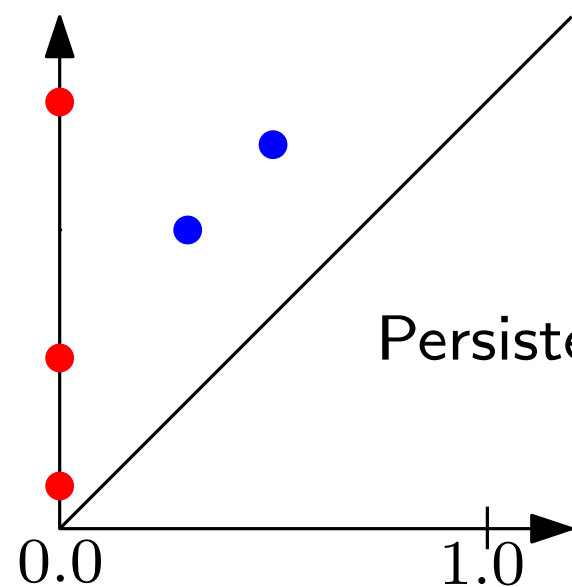
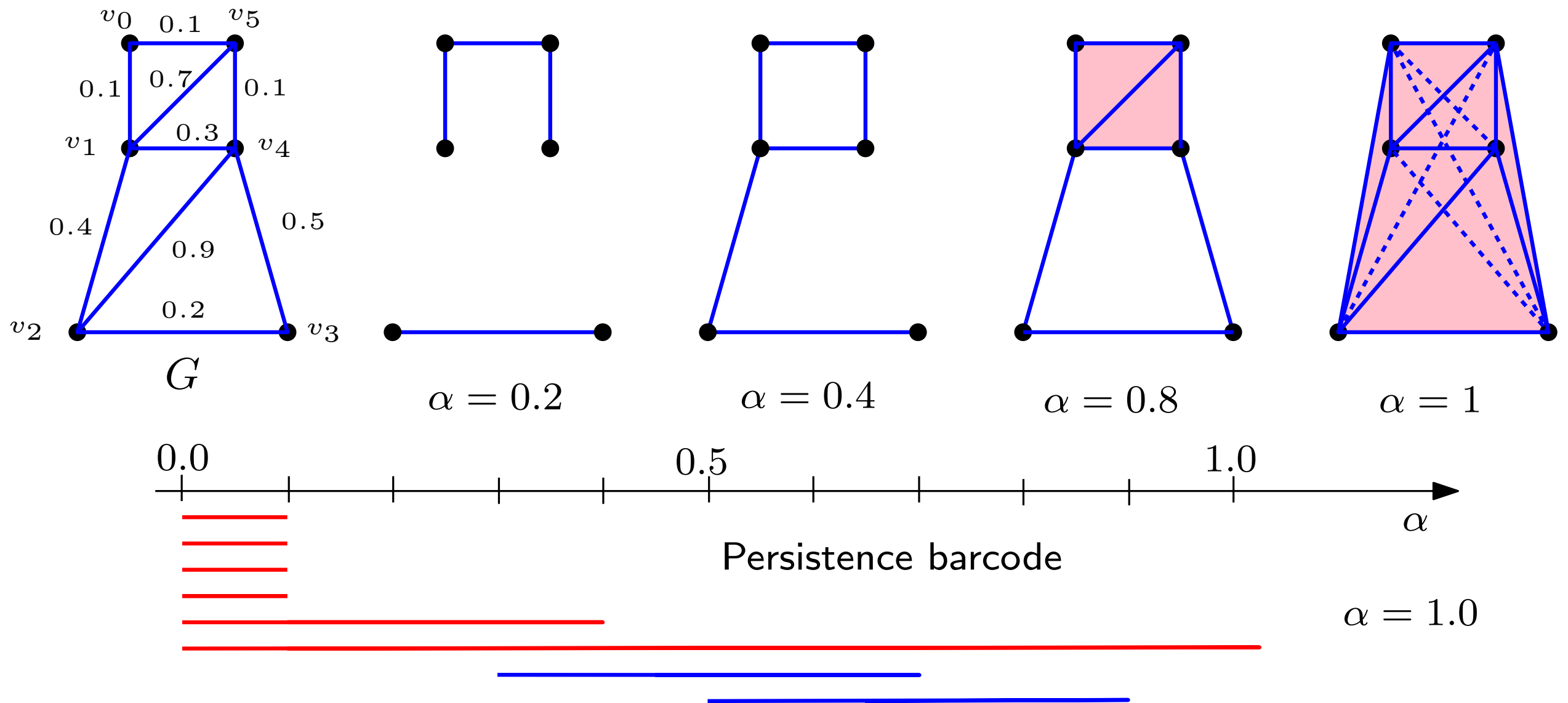
Persistent homology (in a nutshell)



Persistent homology (in a nutshell)



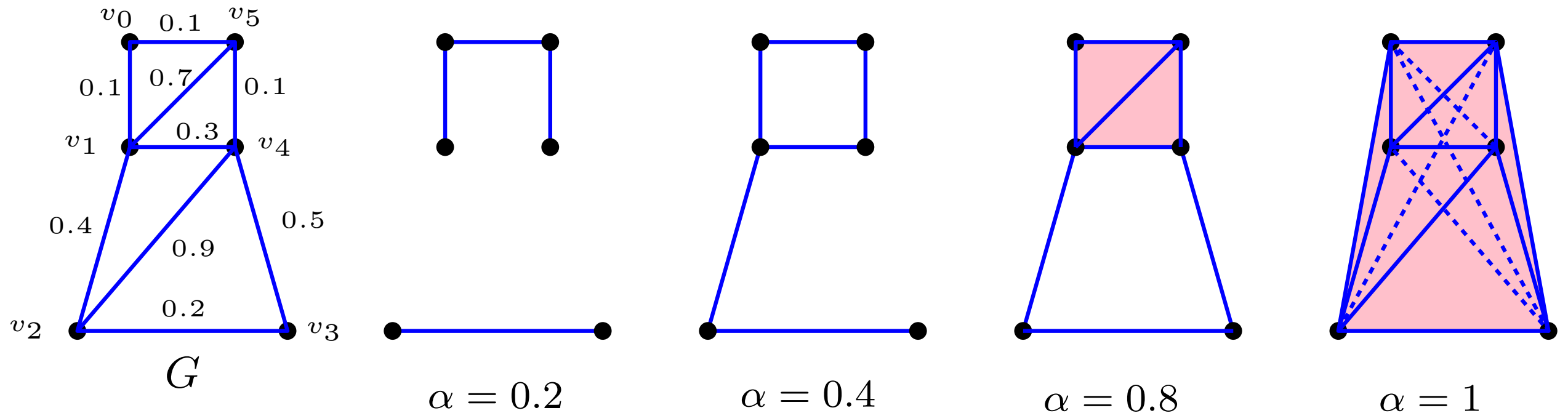
Persistent homology (in a nutshell)



$$X(G) := \sum_{(b,d) \in D_d(G)} \omega(b,d) \delta_{(b,d)},$$

(In the following experiments, $\omega(b,d) \equiv 1$)

Persistent homology (in a nutshell)

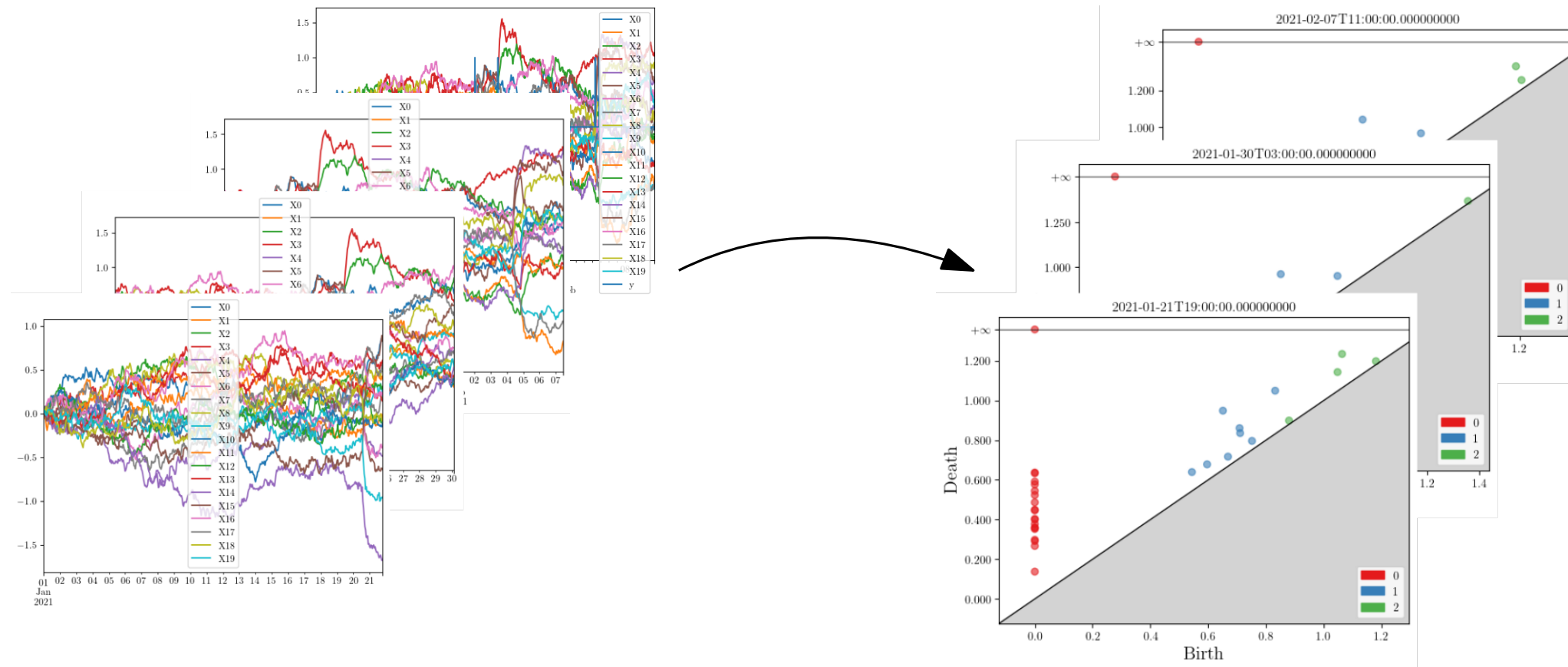


- **Topological information:** the persistence dgm of $Rips_\alpha(G)$ encodes the multi-scale topological structure of G
- **“Interpretability”:** Each point/atom in the persistence diagram represents a topological feature.
- **Stability properties** [C., de Silva, Glisse, Oudot 2013]: if G, G' are two weighted graphs with same vertex set V and edge weight functions $s, s' : V \times V \rightarrow \mathbb{R}$, then

$$d_b(D(G), D(G')) \leq \|s - s'\|_\infty := \sup_{v, v' \in V} |s_{v, v'} - s'_{v, v'}|$$

where d_B , is the so-called bottleneck distance.

From multivariate time series to sequences of measures



Data: A multivariate time series $(Y_t)_{t \in [0, L]} \in \mathbb{R}^D \times [0, L]$

Input: Δ bandwidth, s stride

For t in $\llbracket 0, \lfloor (L - \Delta)/s \rfloor \rrbracket$,

compute the persistence diagrams X_t of the weighted graphs G_t

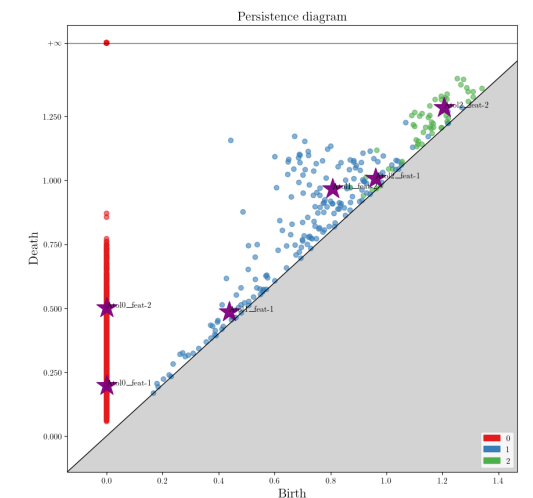
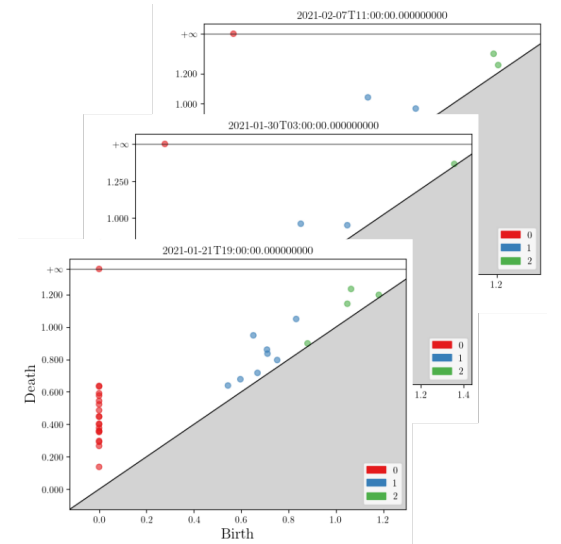
→ A sequence (X_t) of discrete measures in the plane, encoding the topological structure of the correlation graphs.

Vectorizing persistence: the ATOL procedure

An adaptation of the classical k-means to sets of measures

[C., Levrard, Royer 2021]

- X_1, \dots, X_n discrete measures where the X_i 's are sampled according to a random variable $X \in \mathcal{M}(\mathbb{R}^2)$ where $\mathcal{M}(\mathbb{R}^2)$ is the space of measures on \mathbb{R}^2 (not of constant total mass).
- An integer k (dimension of the vectorization).



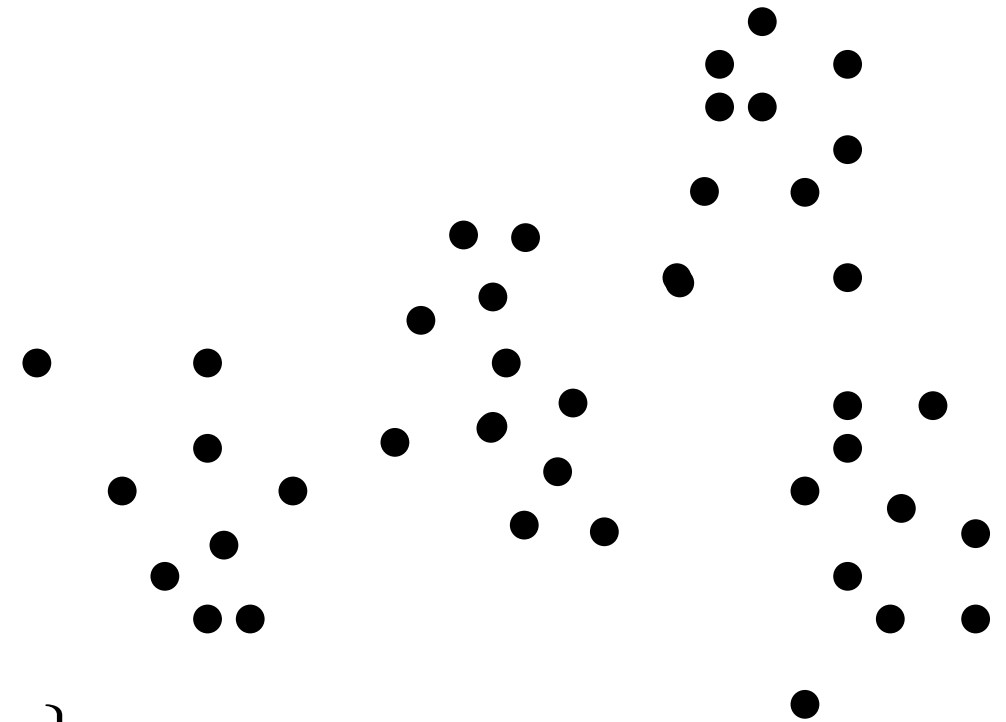
1. Optimal codebook:

$$\mathbf{c}^* \in \arg \min_{\mathbf{c} \in (\mathbb{R}^D)^k} \int \min_{j=1, \dots, k} \|u - c_j\|^2 \mathbb{E}(X)(du) = \arg \min_{\mathbf{c} \in (\mathbb{R}^D)^k} W_2^2(\mathbb{E}(X), P_{\mathbf{c}})$$

2. **Kernel based vectorization:** X_i is vectorized by integrating a kernel centered on the k points of the codebook.

(Reminder:) the k-means algorithm

Input: A (large) set of N points X and an integer $k < N$.

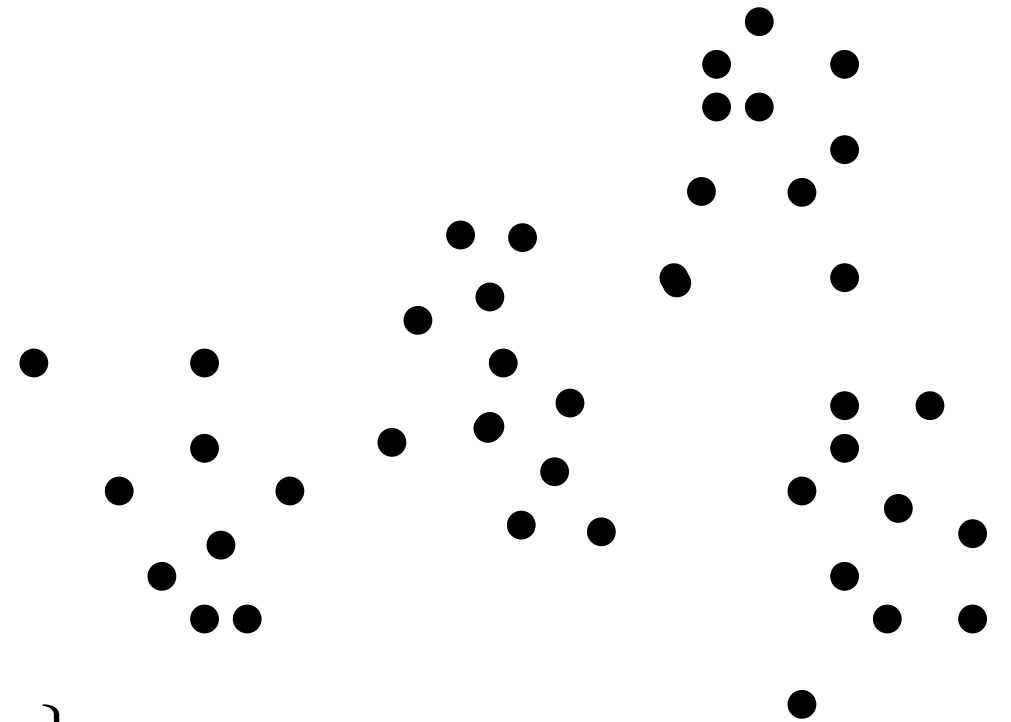


Goal: Find a set of k points $L = \{y_1, \dots, y_k\}$ that minimizes

$$E = \sum_{i=1}^N d(x_i, L)^2$$

(Reminder:) the k-means algorithm

Input: A (large) set of N points X and an integer $k < N$.



Goal: Find a set of k points $L = \{y_1, \dots, y_k\}$ that minimizes

$$E = \sum_{i=1}^N d(x_i, L)^2$$

This is a NP hard problem!

The Lloyd's algorithm: a very simple local search algorithm \rightarrow local minimum.

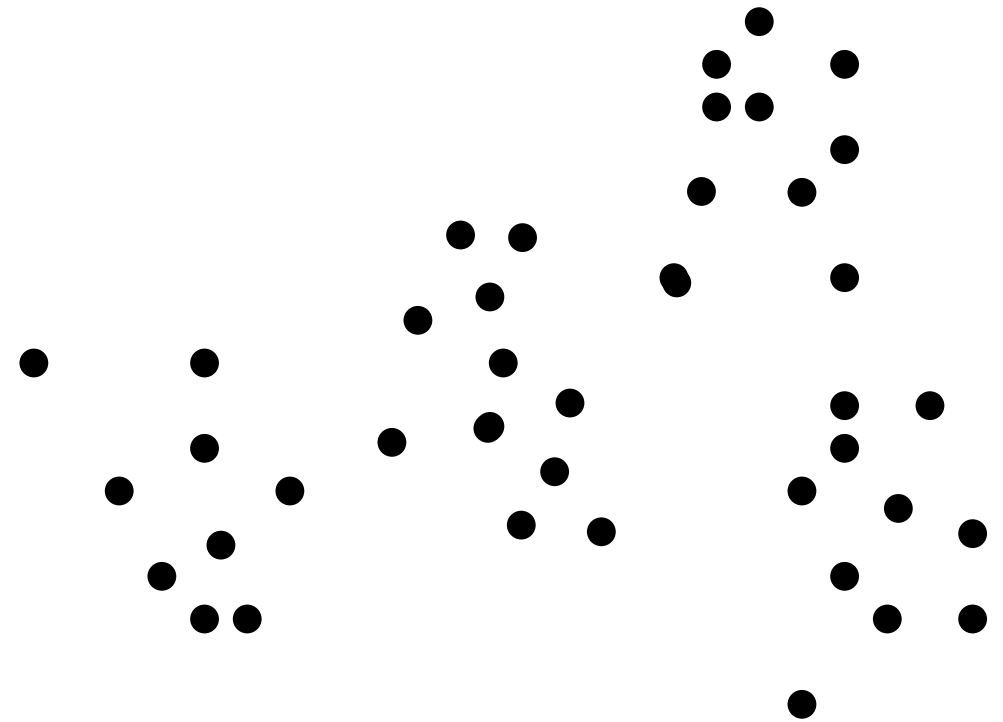
(Reminder:) the k-means algorithm

The Lloyd's algorithm

- Select $L^1 = \{y_1^1, \dots, y_k^1\}$ initial "seeds";
- $i = 1$;
- Repeat
 - For $(j = 1; j \leq k; j++)$ $S_j^i = \{x \in X : d(x, y_j^i) = d(x, L^i)\}$;
 - For $(j = 1; j \leq k; j++)$

$$y_j^{i+1} = \frac{1}{|S_j^i|} \sum_{x \in S_j^i} x$$

- $i++$;
- Until convergence



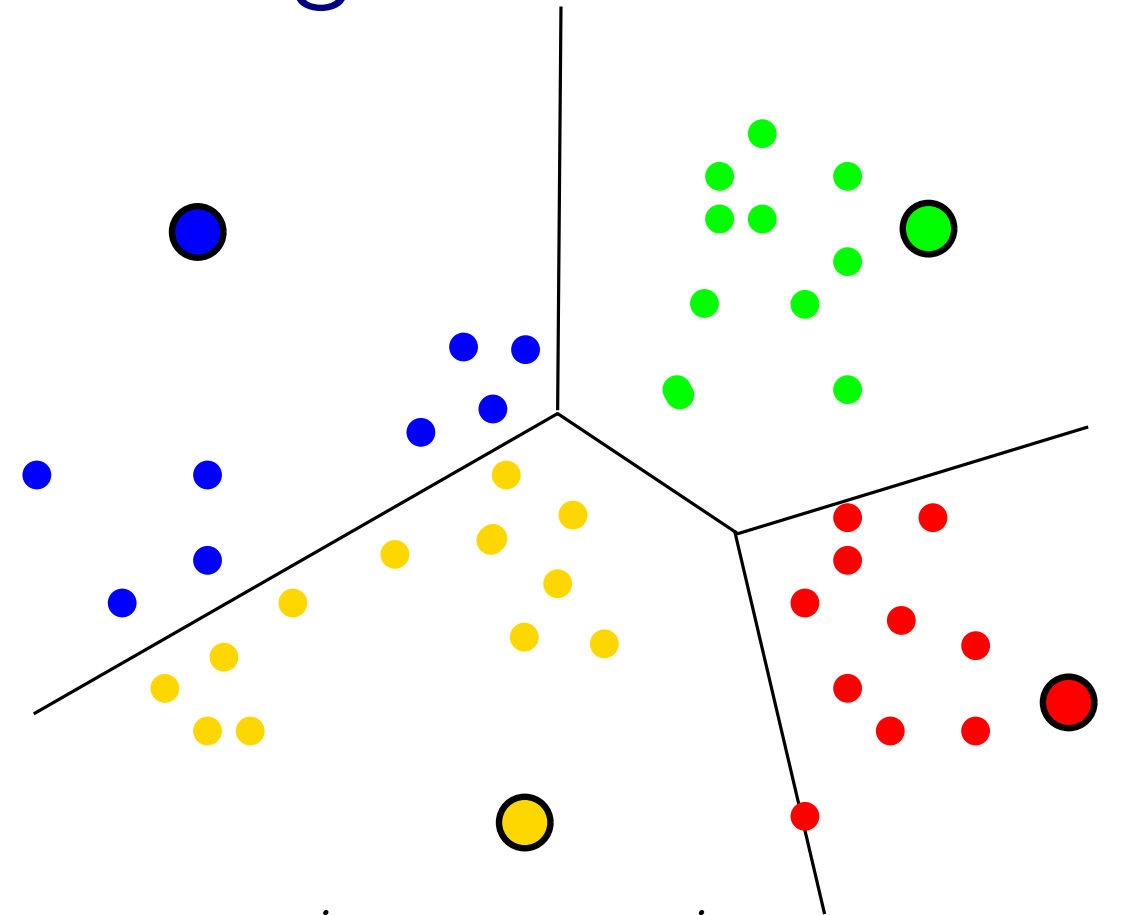
(Reminder:) the k-means algorithm

The Lloyd's algorithm

- Select $L^1 = \{y_1^1, \dots, y_k^1\}$ initial "seeds";
- $i = 1$;
- Repeat
 - For $(j = 1; j \leq k; j++)$ $S_j^i = \{x \in X : d(x, y_j^i) = d(x, L^i)\}$;
 - For $(j = 1; j \leq k; j++)$

$$y_j^{i+1} = \frac{1}{|S_j^i|} \sum_{x \in S_j^i} x$$

- $i++$;
- Until convergence



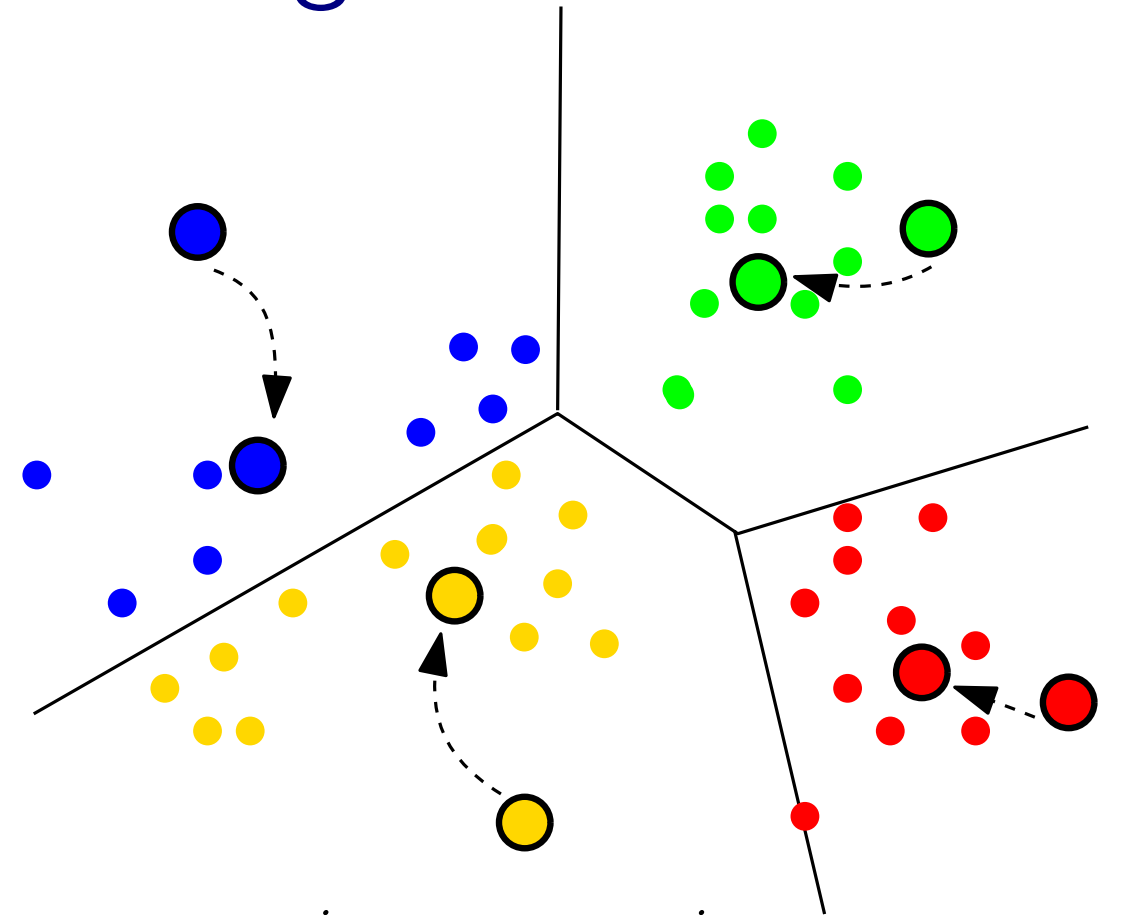
(Reminder:) the k-means algorithm

The Lloyd's algorithm

- Select $L^1 = \{y_1^1, \dots, y_k^1\}$ initial "seeds";
- $i = 1$;
- Repeat
 - For $(j = 1; j \leq k; j++)$ $S_j^i = \{x \in X : d(x, y_j^i) = d(x, L^i)\}$;
 - For $(j = 1; j \leq k; j++)$

$$y_j^{i+1} = \frac{1}{|S_j^i|} \sum_{x \in S_j^i} x$$

- $i++$;
- Until convergence



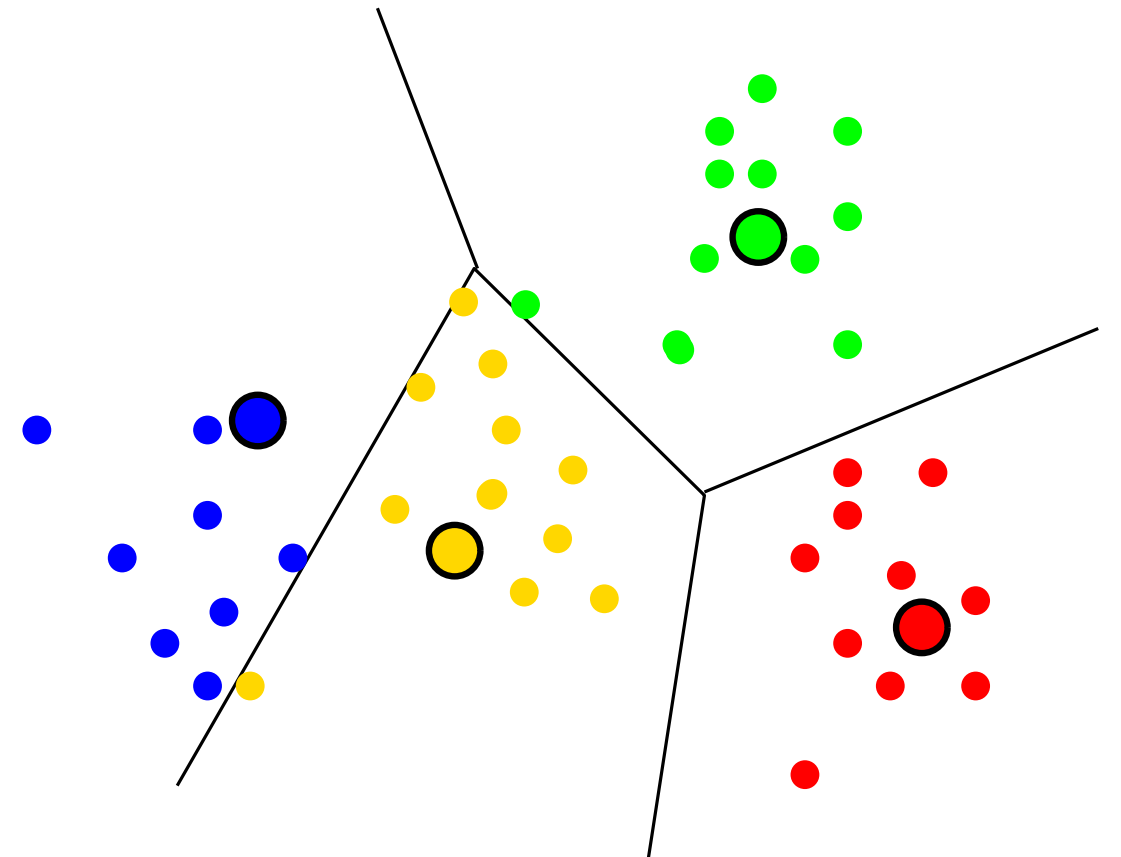
(Reminder:) the k-means algorithm

The Lloyd's algorithm

- Select $L^1 = \{y_1^1, \dots, y_k^1\}$ initial "seeds";
- $i = 1$;
- Repeat
 - For $(j = 1; j \leq k; j++)$ $S_j^i = \{x \in X : d(x, y_j^i) = d(x, L^i)\}$;
 - For $(j = 1; j \leq k; j++)$

$$y_j^{i+1} = \frac{1}{|S_j^i|} \sum_{x \in S_j^i} x$$

- $i++$;
- Until convergence



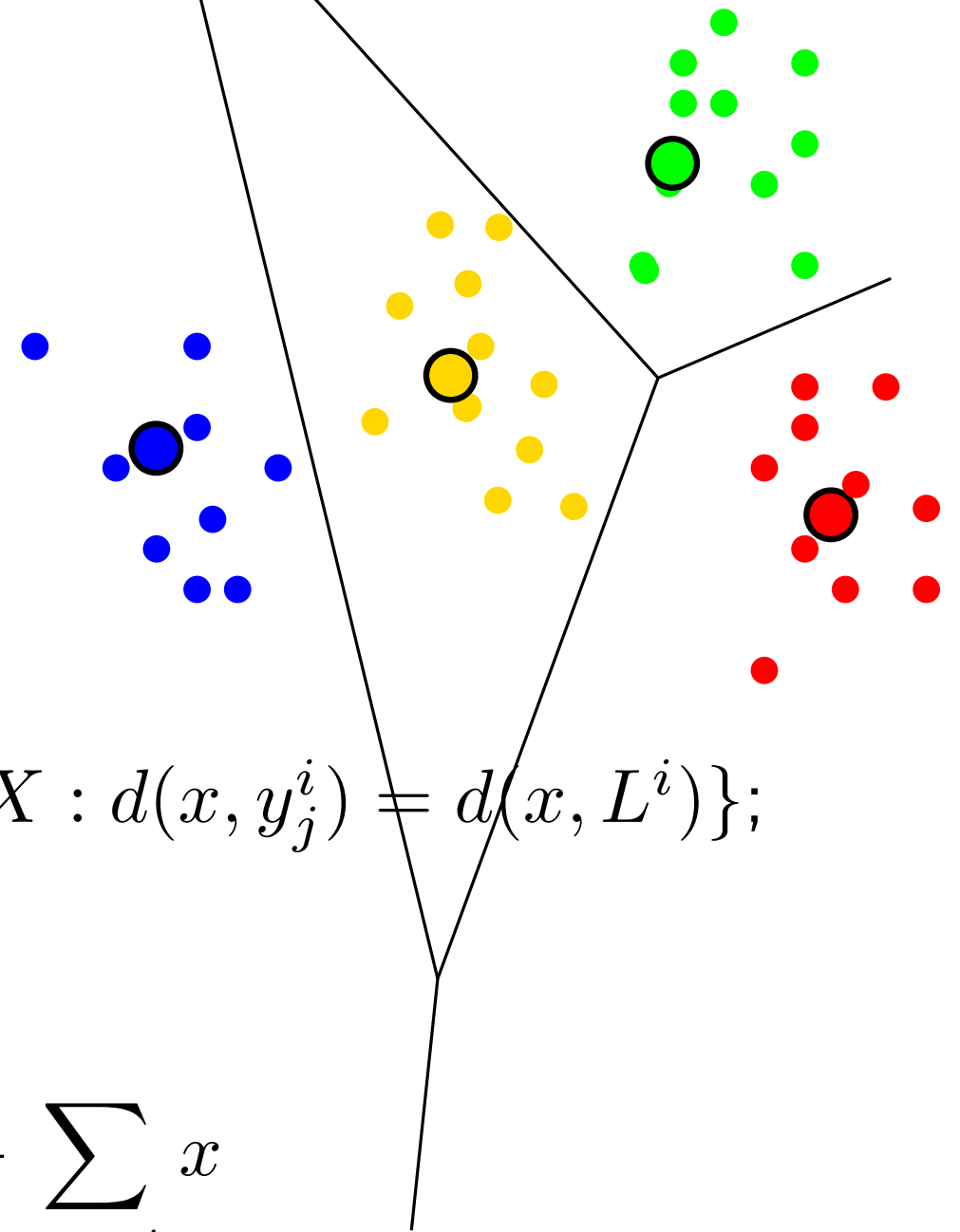
(Reminder:) the k-means algorithm

The Lloyd's algorithm

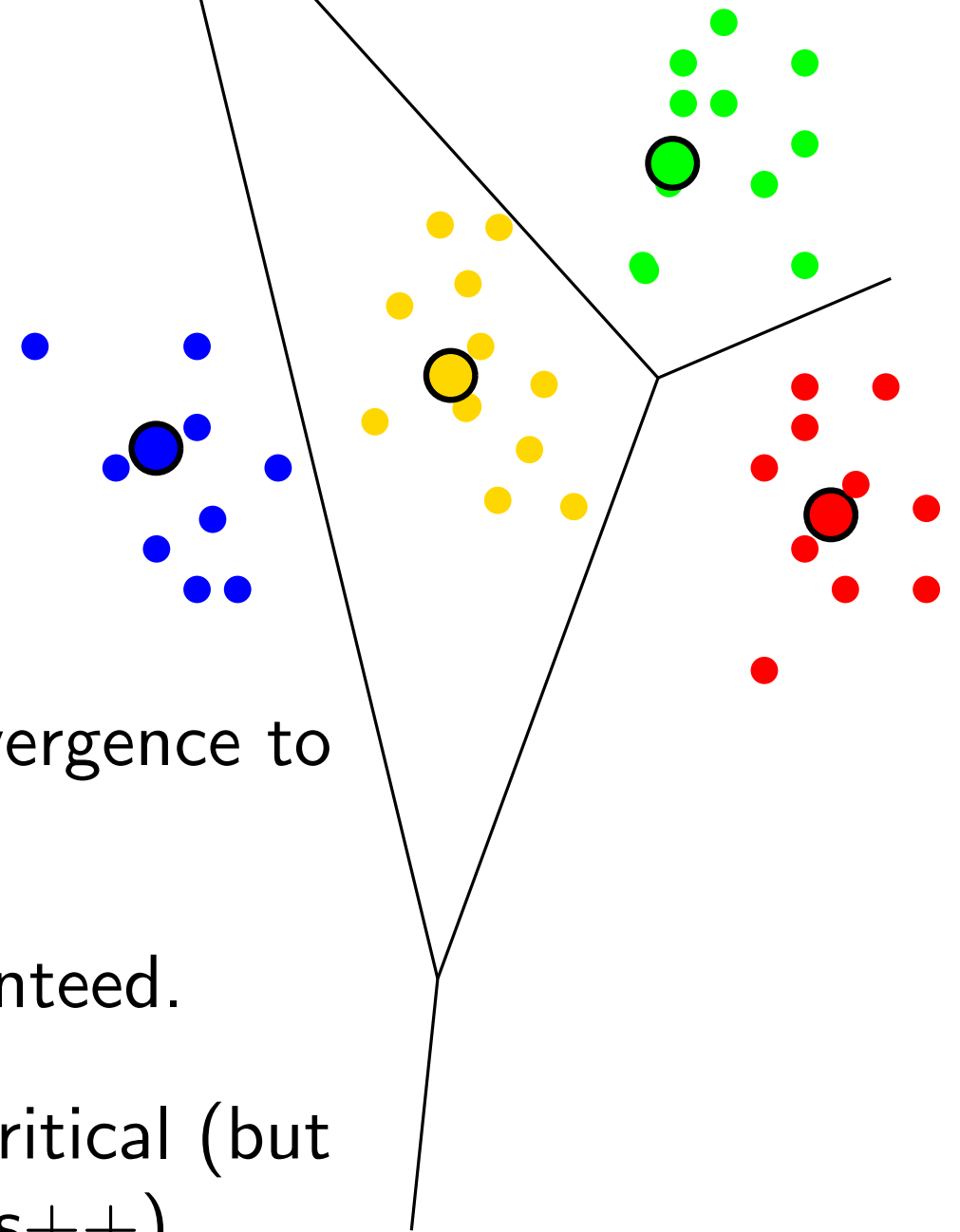
- Select $L^1 = \{y_1^1, \dots, y_k^1\}$ initial "seeds";
- $i = 1$;
- Repeat
 - For $(j = 1; j \leq k; j++)$ $S_j^i = \{x \in X : d(x, y_j^i) = d(x, L^i)\}$;
 - For $(j = 1; j \leq k; j++)$

$$y_j^{i+1} = \frac{1}{|S_j^i|} \sum_{x \in S_j^i} x$$

- $i++$;
- Until convergence



(Reminder:) the k-means algorithm



Warning:

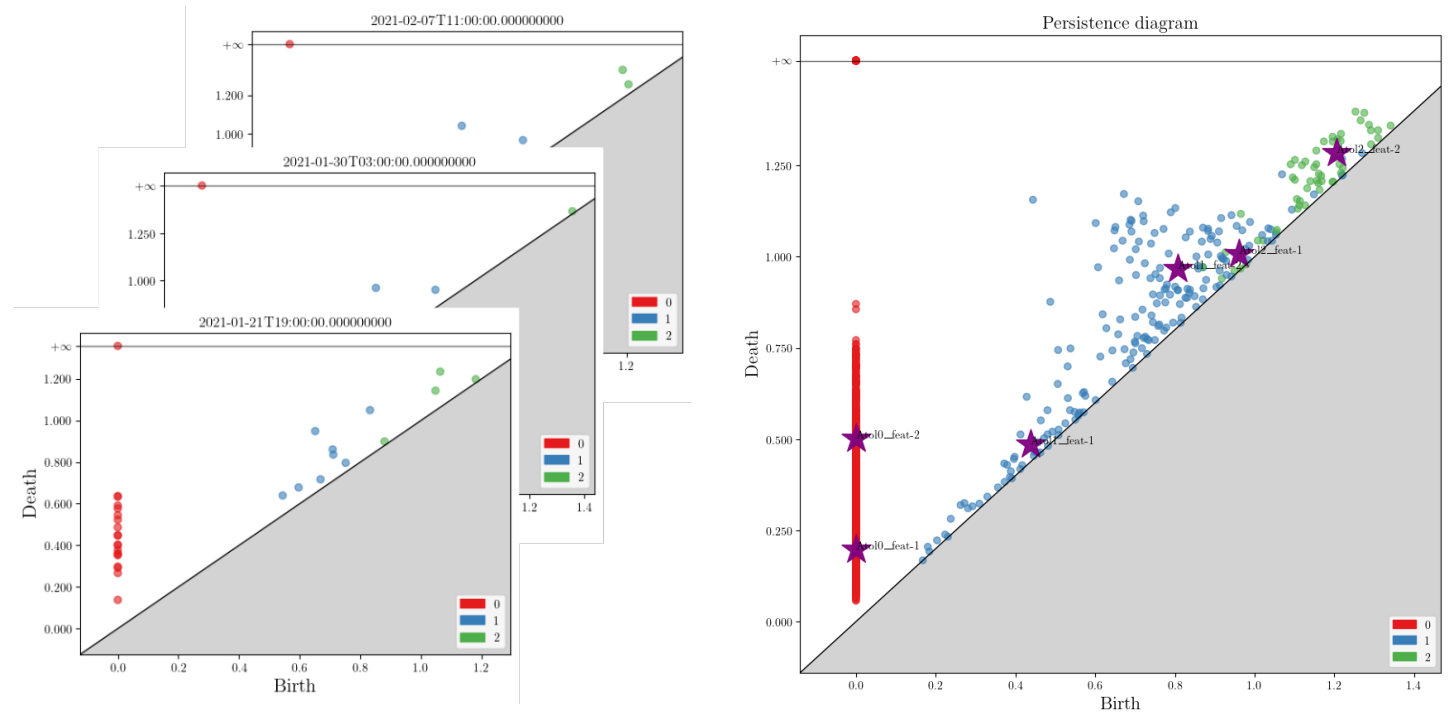
- Lloyd's algorithm does not ensure convergence to a global minimum!
- The speed of convergence is not guaranteed.
- the choice of the initial seeds may be critical (but there exists some strategies \rightarrow k-means++).

Vectorizing persistence: the ATOL procedure

An adaptation of the classical k-means to sets of measures

[C., Levrard, Royer 2021]

Centroids computation



Data: X_1, \dots, X_n discrete measures. $\bar{X}_n := \frac{1}{n} \sum_{i=1}^n X_i$

Input: K : number of centroids. T : stopping time.

Initialization: $\mathbf{c}^{(0)} = (c_1^{(0)}, \dots, c_K^{(0)})$ randomly chosen from \bar{X}_n

At each iteration $t = 1, \dots, T$,

$$W_{j,t-1} \leftarrow \{x \in \mathbb{R}^2 \mid \forall i \neq j \quad \|x - c_j^{(t-1)}\| \leq \|x - c_i^{(t-1)}\|\} \text{ (Voronoi cell).}$$

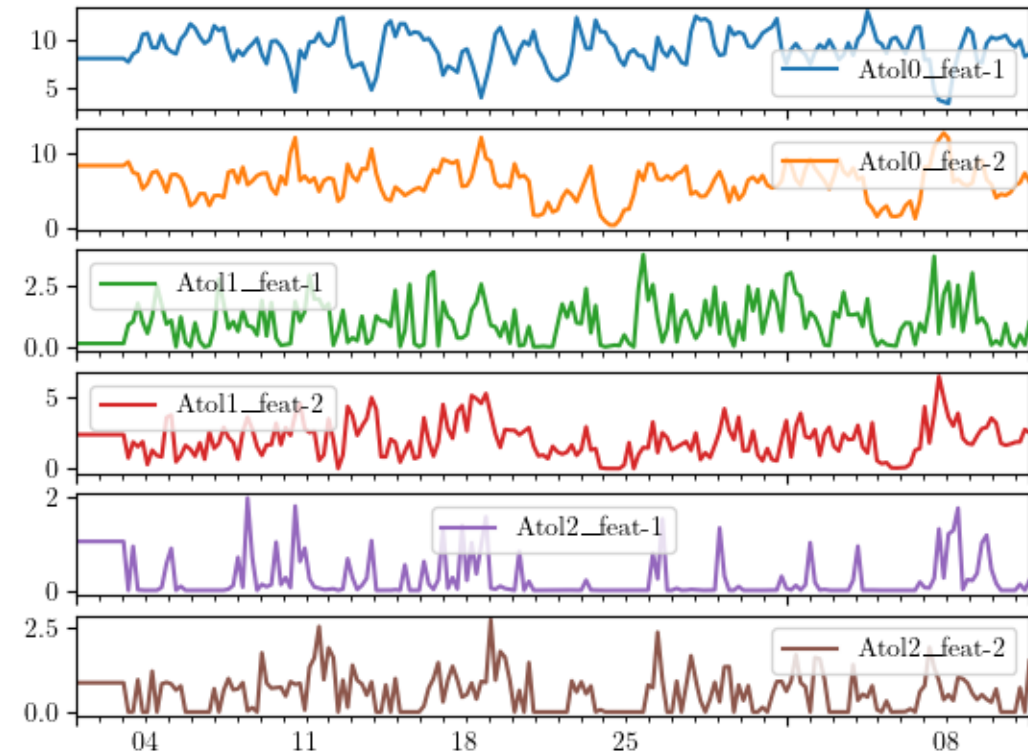
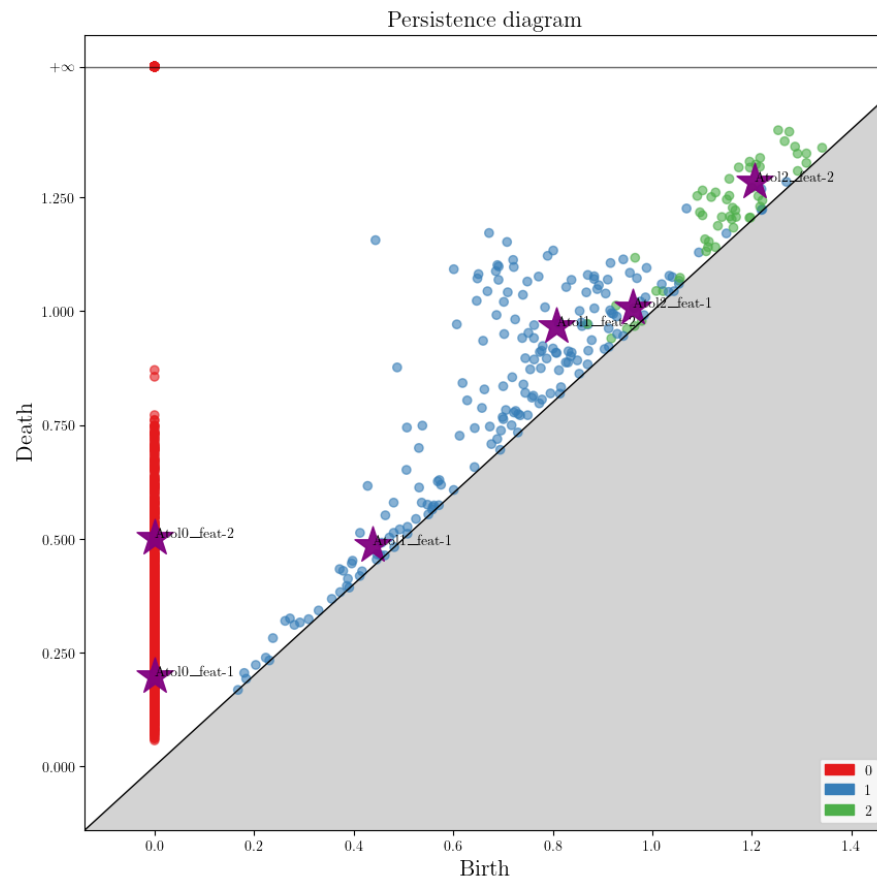
$$c_j^{(t)} \leftarrow (\bar{X}_n(du) (u \mathbb{1}_{W_{j,t-1}}(u))) / \bar{X}_n(W_{j,t-1})$$

Output: Centroids $\mathbf{c}^{(T)} = (c_1^{(T)}, \dots, c_K^{(T)})$.

Vectorizing persistence: the ATOL procedure

An adaptation of the classical k-means to sets of measures

[C., Levrard, Royer 2021]



Vectorization

$$\psi : u \mapsto \exp(-u^2)$$

$$v_i = \left(\int \psi(\|u - c_1^{(T)}\|/\sigma_1) X_i(du), \dots, \int \psi(\|u - c_K^{(T)}\|/\sigma_K) X_i(du) \right),$$

where the bandwidths σ_j 's are defined by

$$\sigma_j = \min_{\ell \neq j} \|c_\ell^{(T)} - c_j^{(T)}\|/2,$$

Convergence guarantees

[C., Levard, Royer 2021]:

In the i.i.d. case (X_1, \dots, X_n are i.i.d sampled according to some random variable X) the output of ATOL with $T = 2 \log(n)$ iterations returns a statistically optimal approximation of

$$\mathbf{c}^* \in \mathcal{C}_{opt} = \operatorname{argmin}_{\mathbf{c} \in (\mathbb{R}^2)^K} \mathbb{E}(X)(du) \min_{j=1, \dots, K} \|u - c_j\|^2 := F(\mathbf{c}),$$

where $\mathbb{E}(X)$ is the mean measure $\mathbb{E}(X) : A \in \mathcal{B}(\mathbb{R}^2) \mapsto \mathbb{E}(X(A))$ (and under some margin condition on $\mathbb{E}(X)$).

Convergence guarantees

[C., Levard, Royer 2021]:

In the i.i.d. case (X_1, \dots, X_n are i.i.d sampled according to some random variable X) the output of ATOL with $T = 2 \log(n)$ iterations returns a statistically optimal approximation of

$$\mathbf{c}^* \in \mathcal{C}_{opt} = \operatorname{argmin}_{\mathbf{c} \in (\mathbb{R}^2)^K} \mathbb{E}(X)(du) \min_{j=1, \dots, K} \|u - c_j\|^2 := F(\mathbf{c}),$$

where $\mathbb{E}(X)$ is the mean measure $\mathbb{E}(X) : A \in \mathcal{B}(\mathbb{R}^2) \mapsto \mathbb{E}(X(A))$ (and under some margin condition on $\mathbb{E}(X)$).

What about the non i.i.d. case?

Convergence guarantees

Assumptions:

- $\dots, X_1, \dots, X_n, \dots$ is stationary, with distribution X (in the space of bounded measures supported on a compact set).
- $\mathbb{E}(X)$ satisfies some margin condition.

Convergence guarantees

Assumptions:

- $\dots, X_1, \dots, X_n, \dots$ is stationary, with distribution X (in the space of bounded measures supported on a compact set).
- $\mathbb{E}(X)$ satisfies some margin condition.

Mixing coefficients: For $t \in \mathbb{Z}$ we denote by $\sigma(-\infty, t)$ (resp. $\sigma(t, +\infty)$) the σ -fields generated by \dots, X_{t-1}, X_t (resp. $X_t, X_{t+1} \dots$). The *beta-mixing* coefficient of order q is then defined by

$$\beta(q) = \sup_{t \in \mathbb{Z}} \mathbb{E} \left[\sup_{B \in \sigma(t+q, +\infty)} |\mathbb{P}(B \mid \sigma(-\infty, t)) - \mathbb{P}(B)| \right].$$

Remark (Proposition): The mixing coefficients of X_t can be controlled by the mixing coefficients of the initial signal Y_t .

Convergence guarantees

Assumptions:

- $\dots, X_1, \dots, X_n, \dots$ is stationary, with distribution X (in the space of bounded measures supported on a compact set).
- $\mathbb{E}(X)$ satisfies some margin condition.

Mixing coefficients: For $t \in \mathbb{Z}$ we denote by $\sigma(-\infty, t)$ (resp. $\sigma(t, +\infty)$) the σ -fields generated by \dots, X_{t-1}, X_t (resp. $X_t, X_{t+1} \dots$). The *beta-mixing* coefficient of order q is then defined by

$$\beta(q) = \sup_{t \in \mathbb{Z}} \mathbb{E} \left[\sup_{B \in \sigma(t+q, +\infty)} |\mathbb{P}(B \mid \sigma(-\infty, t)) - \mathbb{P}(B)| \right].$$

Theorem: Choosing $T \geq \lceil \frac{\log(n/q)}{\log(4/3)} \rceil$, if q is such that $\beta(q)/q \leq n^{-1}$ and $\mathbf{c}^{(0)}$ is not too far (can be made explicit) from \mathbf{c}^* , then

$$\mathbb{E} \left(\inf_{\mathbf{c}^* \in \mathcal{C}_{opt}} \|\mathbf{c}^{(T)} - \mathbf{c}^*\|^2 \right) \leq C \frac{q}{n}$$

where C is a (explicit) constant.

Score function (classical)

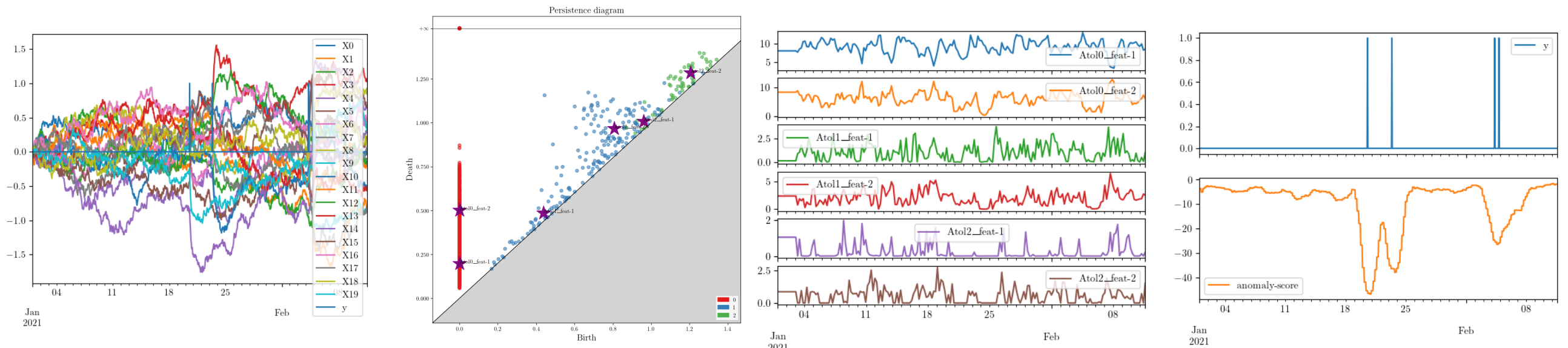
A sequence of vector-valued vectors

$$v_i = \left(\int \psi(\|u - c_1^{(T)}\|/\sigma_1) X_i(du), \dots, \int \psi(\|u - c_K^{(T)}\|/\sigma_K) X_i(du) \right) \in \mathbb{R}^K$$

has been built encoding the persistent homology of the correlation graphs G_i 's

1. Compute a (robust) estimation of the **mean** $\hat{\mu}$ and **covariance matrix** $\hat{\Sigma}$ of the distribution of the v_i 's (e.g. using the Minimum Covariance Determinant Estimator).
2. for a new vector v , a detection score is built via

$$s^2(v) = (v - \hat{\mu})^T \hat{\Sigma}^{-1} (v - \hat{\mu}),$$



Example on a simulated (Ornstein-Uhlenbeck) process with anomalies

Score function (classical)

A sequence of vector-valued vectors

$$v_i = \left(\int \psi(\|u - c_1^{(T)}\|/\sigma_1) X_i(du), \dots, \int \psi(\|u - c_K^{(T)}\|/\sigma_K) X_i(du) \right) \in \mathbb{R}^K$$

has been built encoding the persistent homology of the correlation graphs G_i 's

1. Compute a (robust) estimation of the **mean** $\hat{\mu}$ and **covariance matrix** $\hat{\Sigma}$ of the distribution of the v_i 's (e.g. using the Minimum Covariance Determinant Estimator).
2. for a new vector v , a detection score is built via

$$s^2(v) = (v - \hat{\mu})^T \hat{\Sigma}^{-1} (v - \hat{\mu}),$$

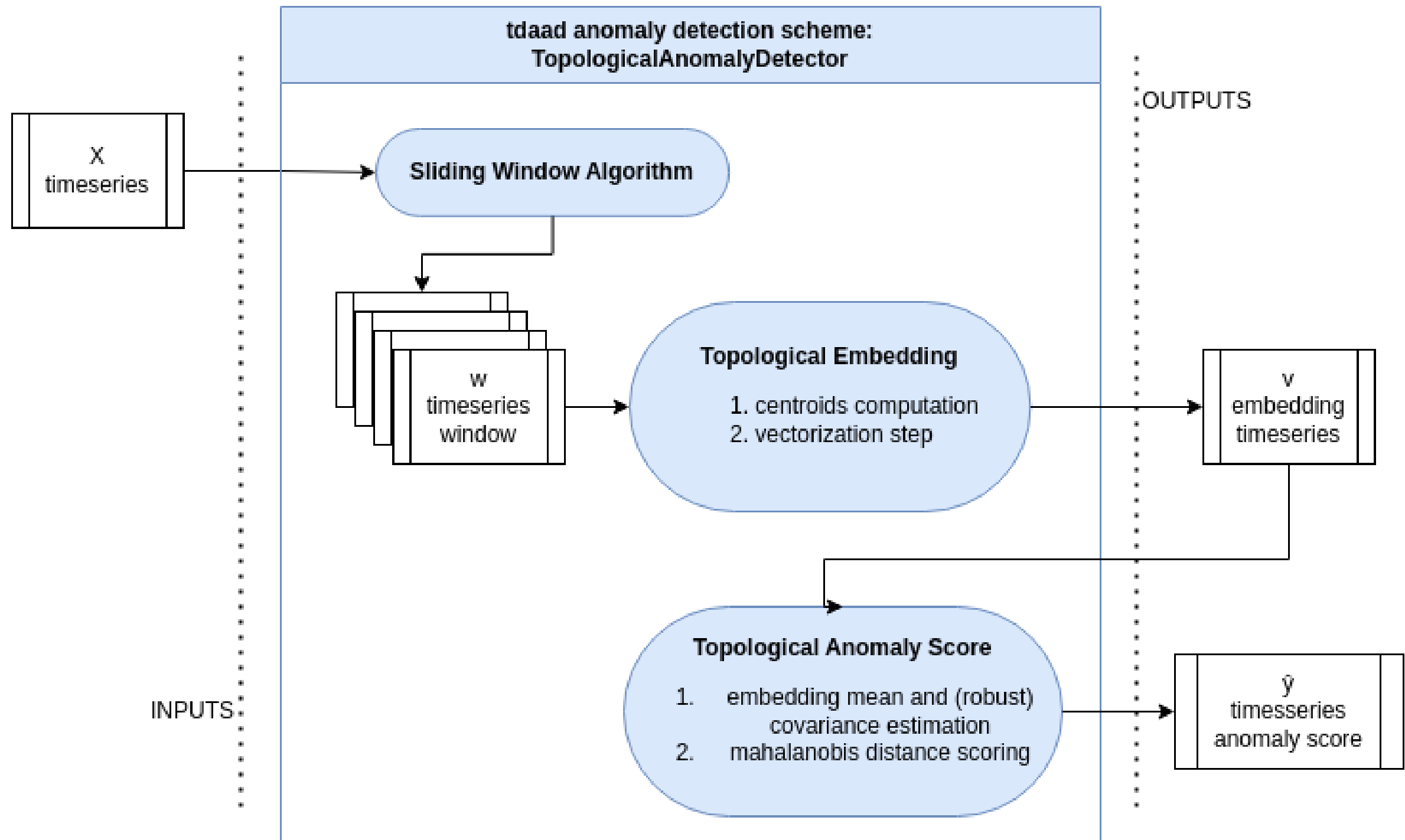
Denoting \hat{s} the score function based on some training data $(Y_t)_{t \in [0, T]}$, then anomaly detection tests of the form

$$T_\alpha(v) = \mathbb{1}_{\hat{s}(v) \geq t_\alpha}$$

may be (classically) built.

To summarize (overview)

Topological Analysis for Detecting Anomalies (TADA)



(Some) Experiments

Two kinds of experiments:

1. Challenging synthetic data set with prescribed underlying topological structure (Topological Wheels Dataset)
2. Evaluation on a large benchmark of publicly available synthetic and real-world datasets
3. Industrial use cases.

Evaluation criteria:

1. Area under the ROC curve
2. Area under the PR curve

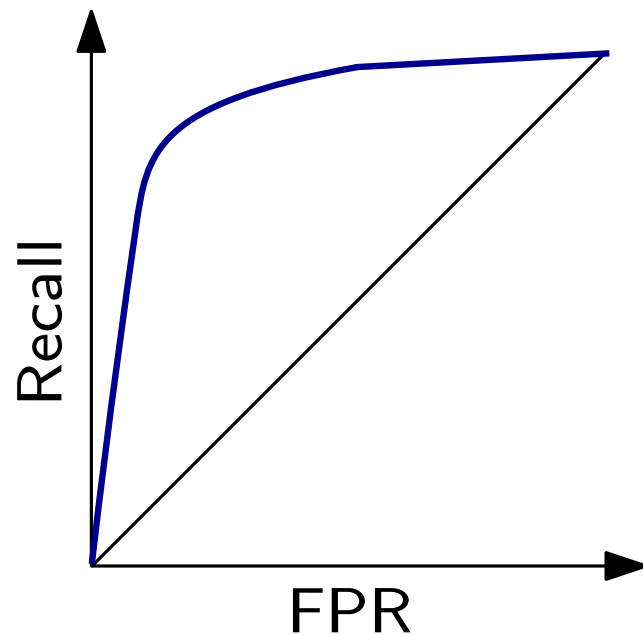
(Some) Experiments

ROC and PR curves (reminder):

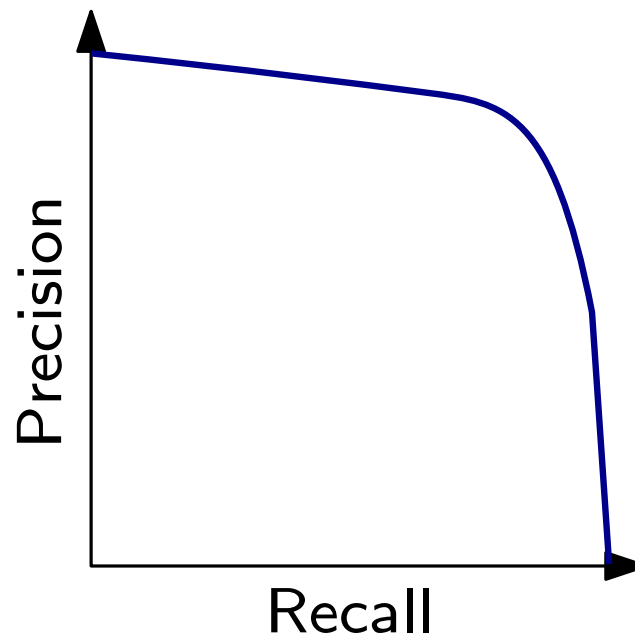
$$Precision = \frac{TP}{TP + FP}$$

$$Recall(Truth\ Positive\ Rate) = \frac{TP}{TP + FN}$$

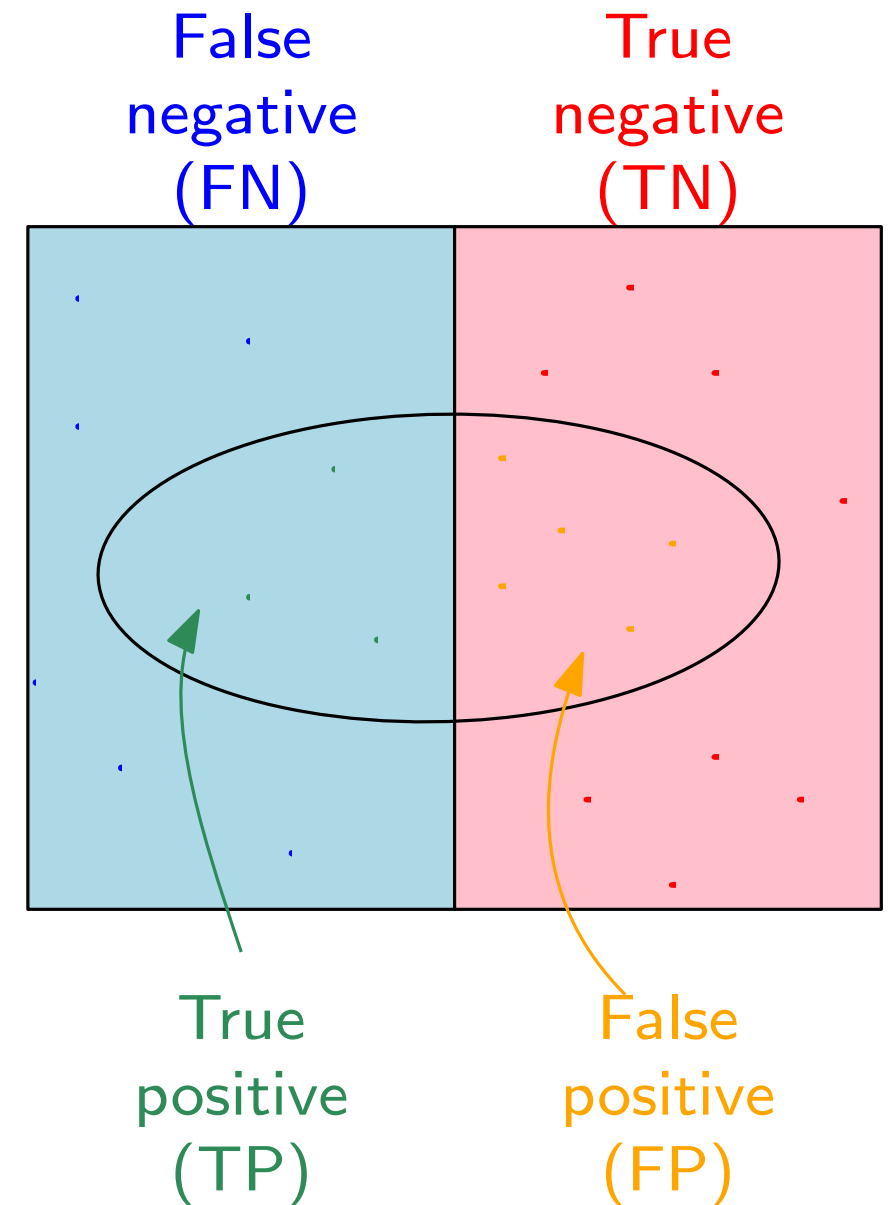
$$False\ Positive\ Rate\ (FPR) = \frac{FP}{FP + TN}$$



ROC curve



PR curve



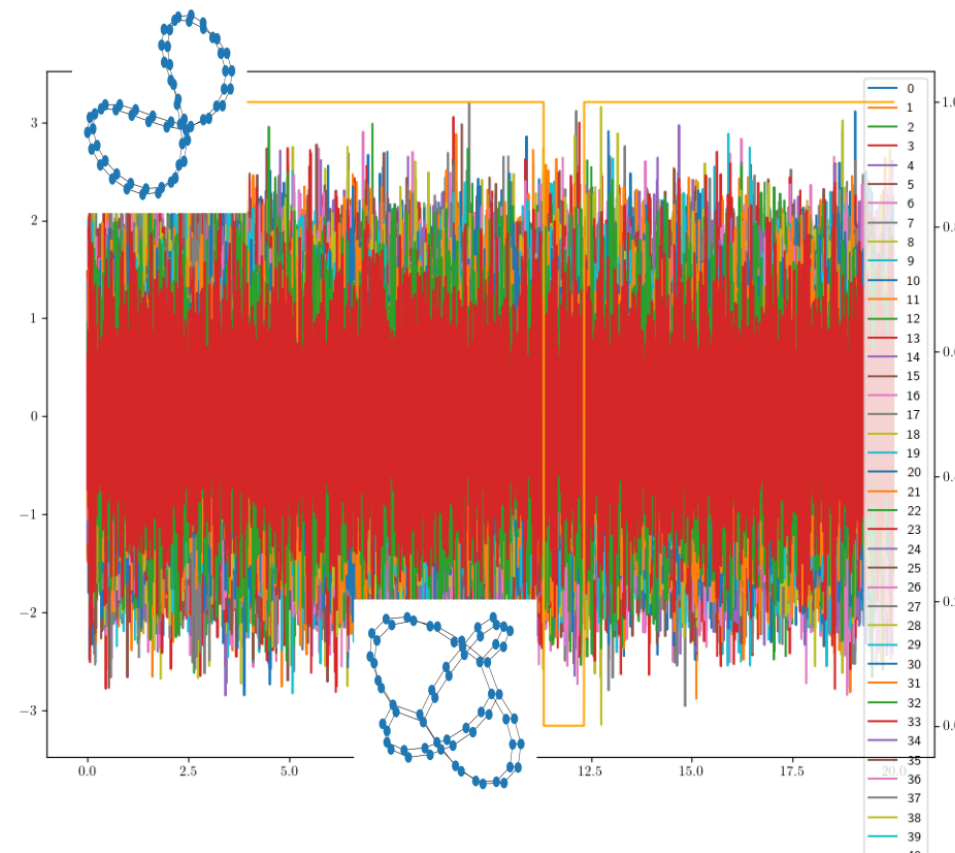
(Some) Experiments

Compared methods:

- ABATS: our
- ATOL-RF: random forest on ATOL topol. vectorization (supervised).
- ATOL-IF: isolation forest on ATOL topol. vectorization.
- Spectral-RF: random forest on spectral features (supervised).
- SubKNN, TorsKAD, KMeansAD... : some of the best performing methods on the used benchmark datasets.

The Topological wheels experiment

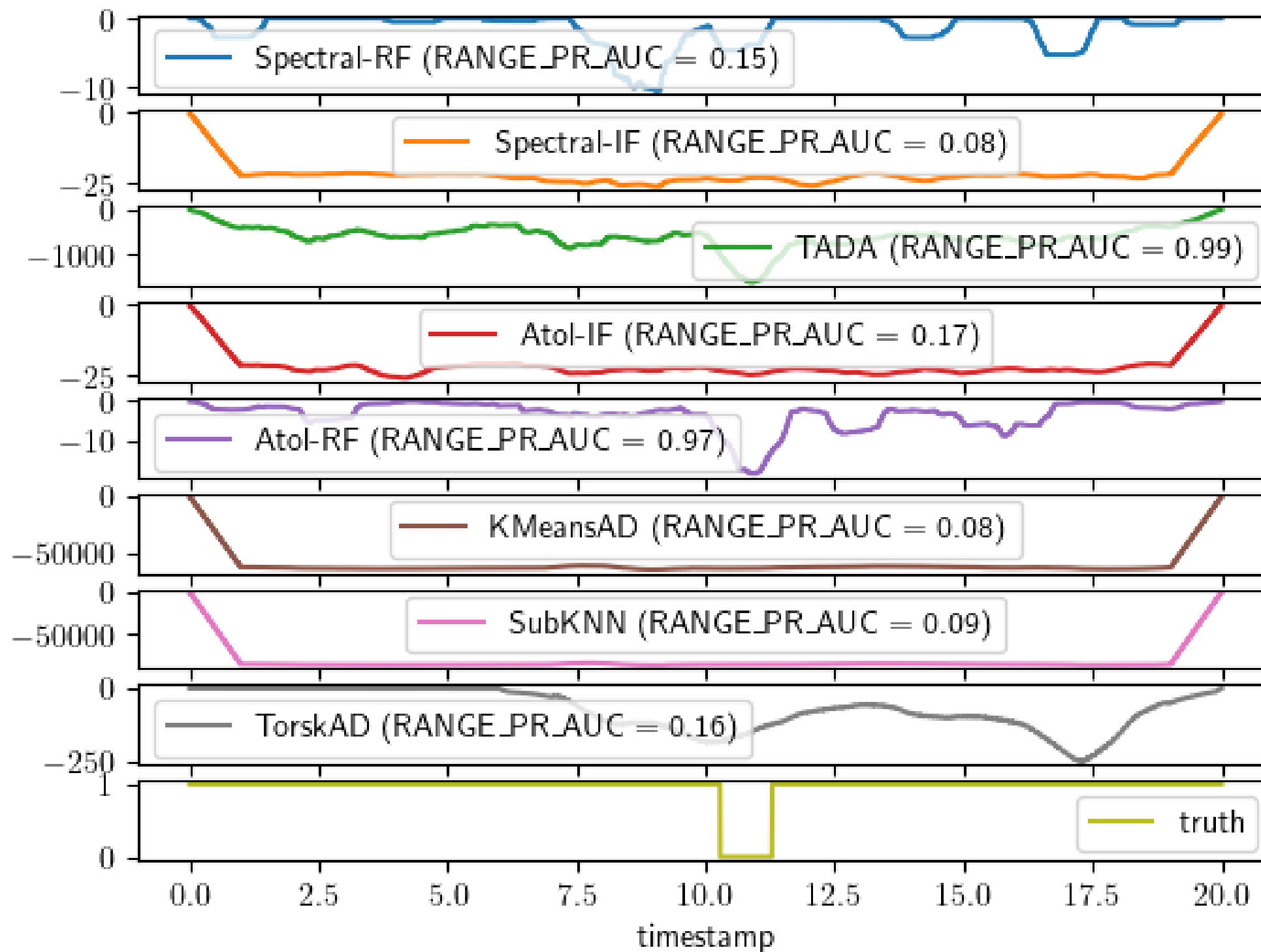
Inspired from [Bourakna, Chung, Ombao 2022]



- Generated through an auto-regressive process (of order 2) with prescribed underlying dependency structure between the channels (Bourakna et al 2022).
- Emulate network structures of the brain
- 10 realizations in dimension 40 with 10,000 timestamps and a randomly located segment (of length 500) abnormalous change in the dependency structure.

The Topological wheels experiment

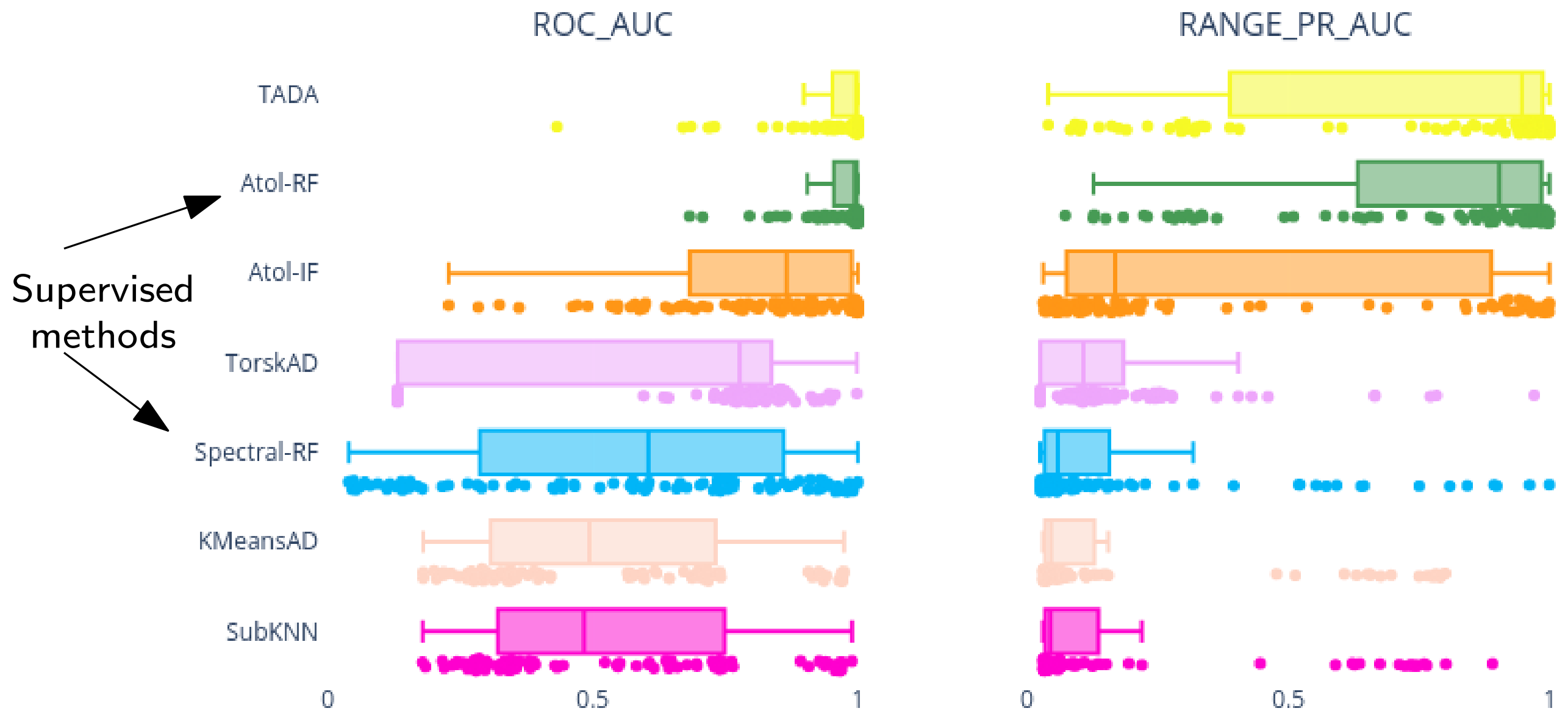
Result on an example:



Supervised
methods

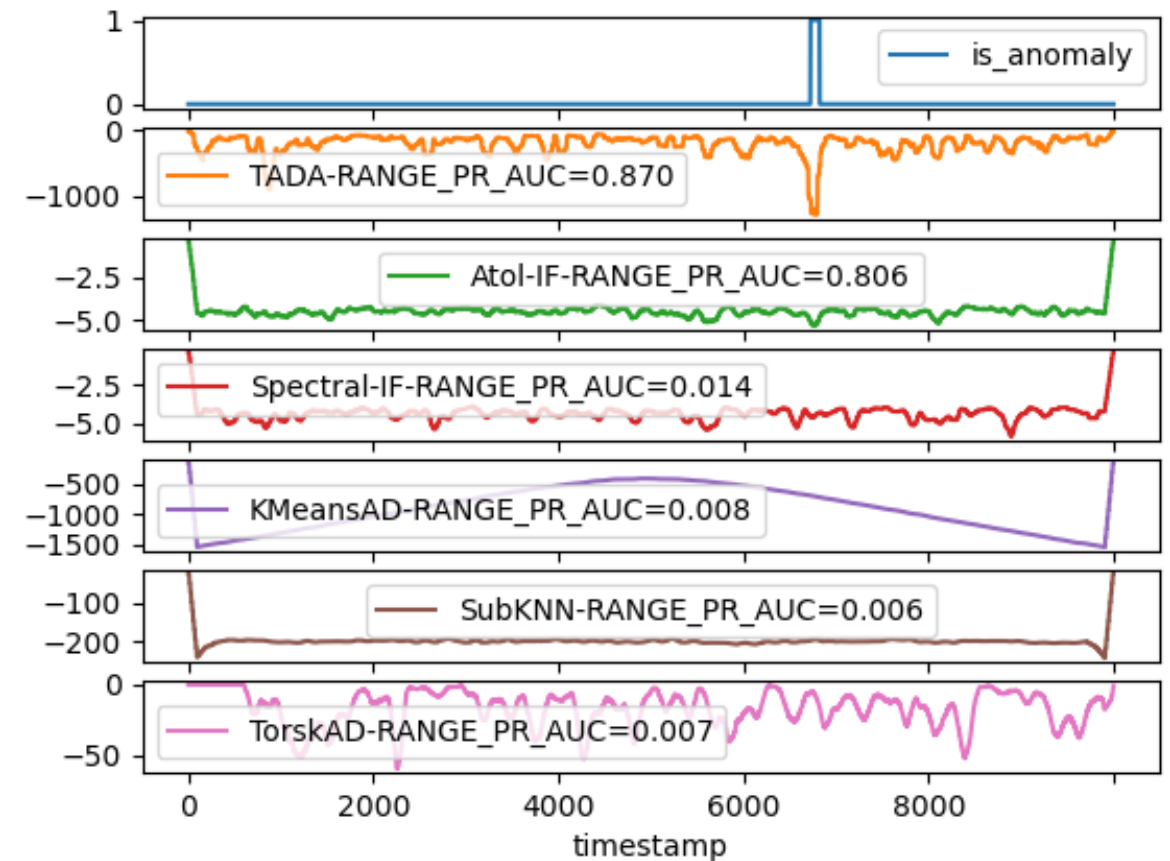
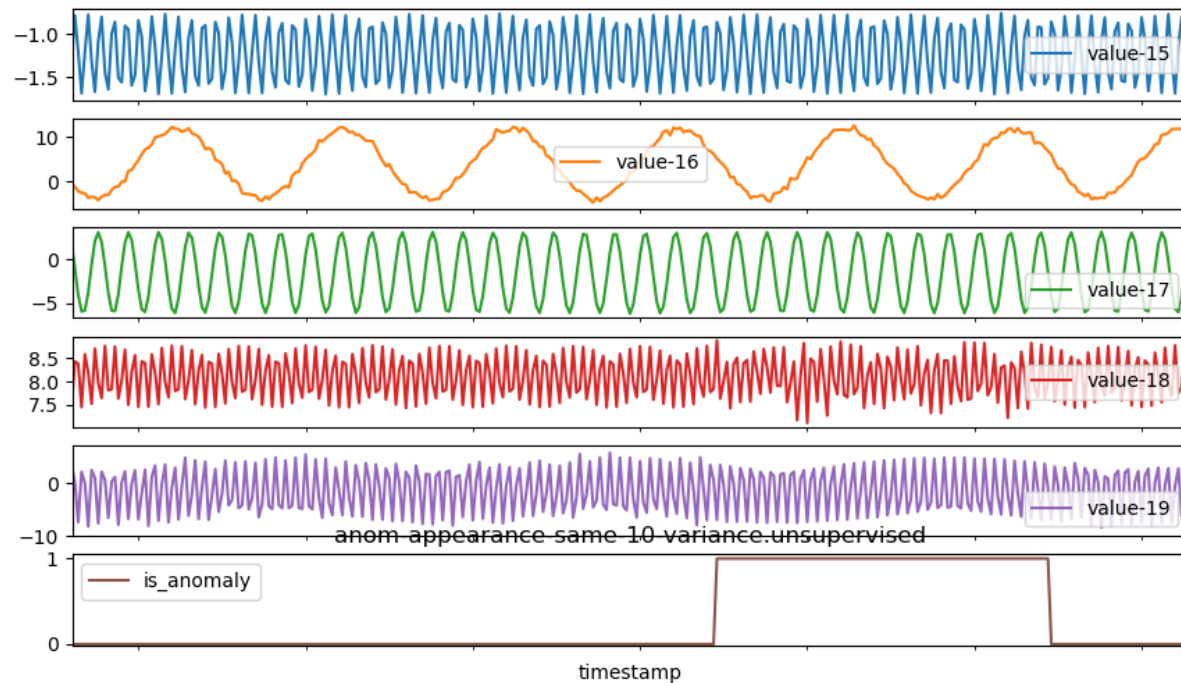
The Topological wheels experiment

Result for training on 1 dataset and evaluating on the 9 others:



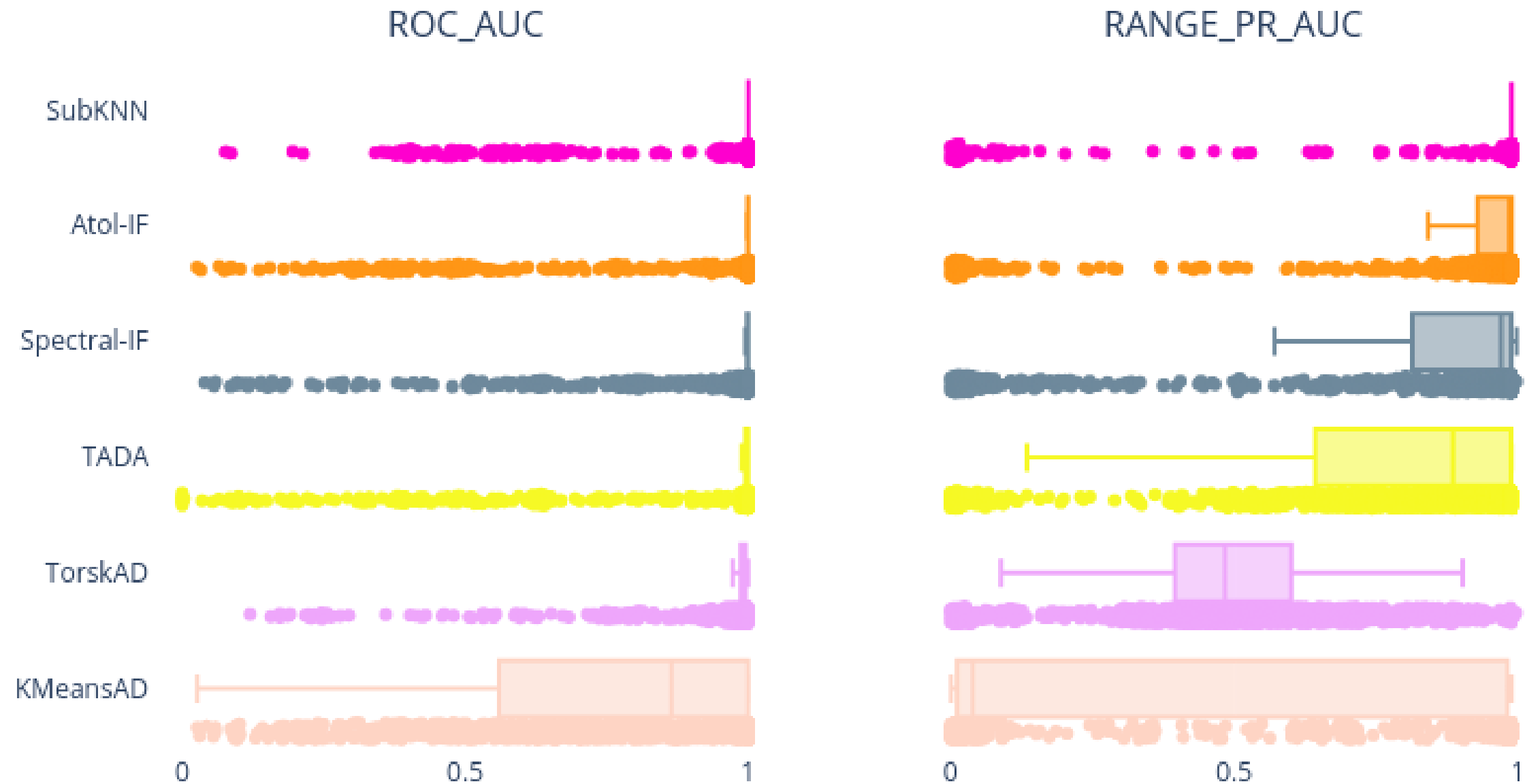
The GutenTAG benchmark

1084 synthetic datasets from [Wenig, Schmidl, Papenbrock 2022]



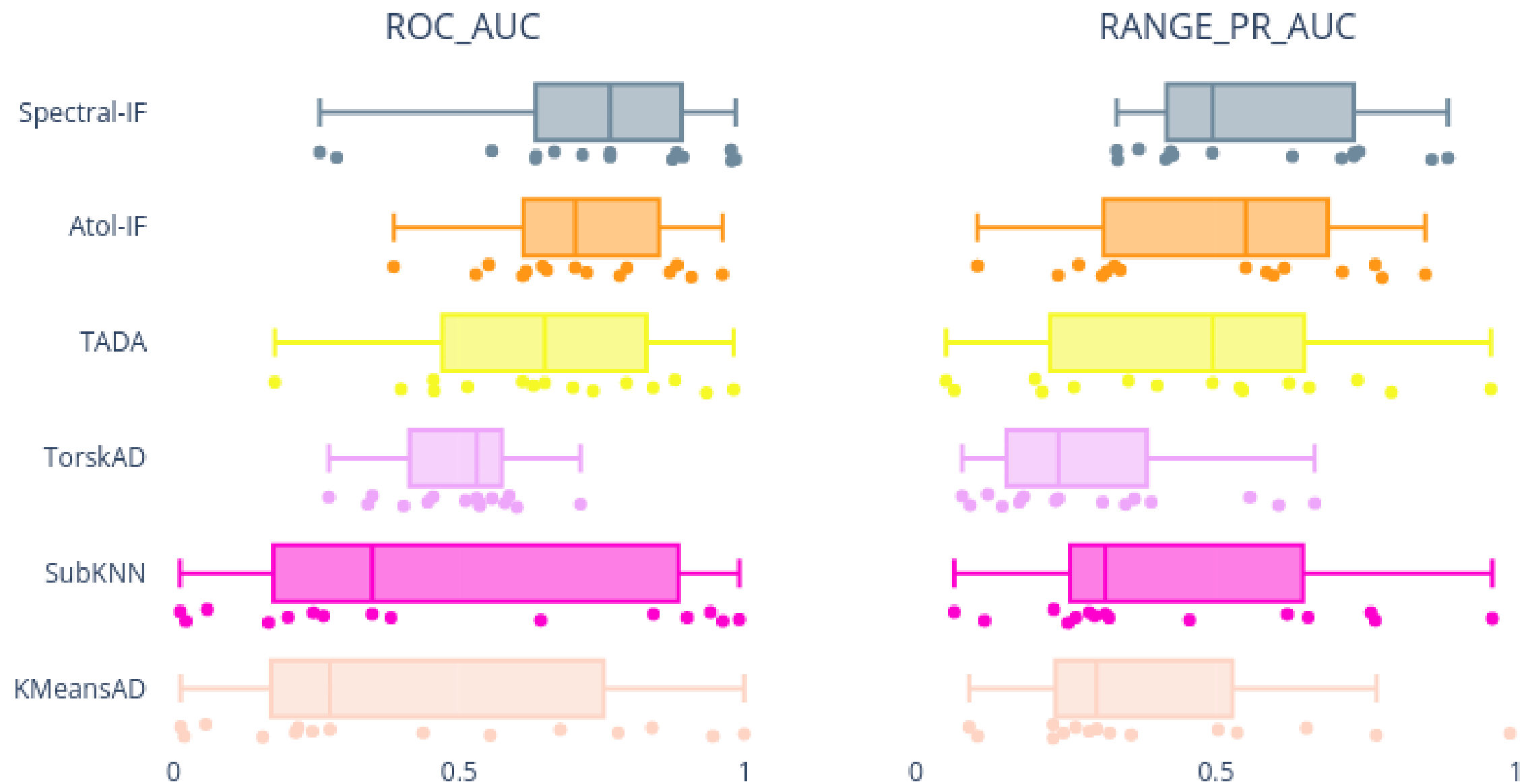
The GutenTAG benchmark

1084 synthetic datasets from [Wenig, Schmidl, Papenbrock 2022]



Exathlon real datasets

15 datasets (traces from repeated executions of large-scale stream processing jobs on a cluster) from [Jacob et al 2021]



ATOL and TADA references

References:

- [1] M. Royer, F.Chazal, C.Levrard, Y. Umeda, Y. Ike. ATOL: Measure Vectorization for Automatic Topologically-Oriented Learning. AISTAT 2021
- [2] F. Chazal, C. Levrard and M. Royer. Clustering of measures via mean measure quantization. Electronic Journal of Statistics 2021.
- [3] F. Chazal, C. Levrard and M. Royer. Topological Analysis for Detecting Anomalies in dependent sequences: application to Time Series. Journal of Machine Learning Research (JMLR). 25(365):1-49, 2024.

Software:

ATOL: available through the GUDHI library

TADA: code (<https://github.com/IRT-SystemX/tdaad/>) + doc (<https://irt-systemx.github.io/tdaad/>)

Extra slides

