Topological optimization: gradient descent and regularization

Mathieu Carrière
INRIA Sophia-Antipolis
mathieu.carriere@inria.fr
Recall: persistence diagrams and machine learning
Recall: persistence diagrams and machine learning
Recall: persistence diagrams and machine learning

\[ k(\cdot, \cdot) := \langle \Phi(\cdot), \Phi(\cdot) \rangle_{\mathcal{H}} \]

Persistence diagram
Recall: persistence diagrams and machine learning

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

Etc.

\[ k(\cdot, \cdot) := \langle \Phi(\cdot), \Phi(\cdot) \rangle_{\mathcal{H}} \]
Recall: 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?
Recall: persistence diagrams and machine learning

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

Etc.

$\Phi_k(\cdot, \cdot) := \langle \Phi(\cdot), \Phi(\cdot) \rangle_H$

What filtration to choose?

What linearization to choose?
Persistence images

Data → Diagram $B$ → Diagram $T(B)$ → Image

- Compute PD
- Rotate PD
- Pixelate + concatenate into vector

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

Discretize plane into one or several grid(s):

For each pixel $P$, compute $I(P) = \# D \cap P$

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

Stability $\rightarrow$ weight points: $w_t(x, y) = \frac{1}{t}$

$\rightarrow$ blur image

(convolve with Gaussian $\rightarrow$ details forthcoming)
Persistence images

Prop:

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

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

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

\( \nu_x^y : H(f^{-1}(-\infty, x)) \rightarrow H(f^{-1}(-\infty, y)) \) induced linear map

Rank function is defined as \( \lambda(x, y) = \text{rank} \nu_x^y \)
Persistence landscapes

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: \( \Lambda(i, t) = \lambda_{\lfloor i \rfloor}(t) \)
Persistence landscapes

Prop:

- $\|\Lambda(D) - \Lambda(D')\|_\infty \leq d_\infty(D, D')$
- $\min\{1, C(D, D')\|\Lambda(D) - \Lambda(D')\|_2\} \leq d_2(D, D')$
Sliced Wasserstein distance

Project points onto line defined with angle $\theta$

Sort projections

Take 1-norm and average over many lines

$$\langle p_1, e_\theta \rangle_{\pi(1)}$$

$$\langle p_n, e_\theta \rangle_{\pi(n)}$$

$$\langle q_1, e_\theta \rangle_{\pi'(1)}$$

$$\langle q_n, e_\theta \rangle_{\pi'(n)}$$

$$L_\theta$$

$$L'_\theta$$

$$\|L_{\theta_1} - L'_{\theta_1}\|_1$$

$$\|L_{\theta_2} - L'_{\theta_2}\|_1$$

$$\|L_{\theta_k} - L'_{\theta_k}\|_1$$
Sliced Wasserstein distance

Project points onto line defined with angle $\theta$

Sort projections

Take 1-norm and average over many lines

$\langle p_1, e_\theta \rangle$
$\vdots$
$\langle p_n, e_\theta \rangle$

$\langle q_1, e_\theta \rangle$
$\vdots$
$\langle q_n, e_\theta \rangle$

$\langle p_{\pi(1)}, e_\theta \rangle$
$\vdots$
$\langle p_{\pi(n)}, e_\theta \rangle$

$\langle q_{\pi'(1)}, e_\theta \rangle$
$\vdots$
$\langle q_{\pi'(n)}, e_\theta \rangle$

$\theta$

$L_\theta$

$\|L_{\theta_1} - L'_{\theta_1}\|_1$
$+$
$\vdots$
$+$
$\|L_{\theta_k} - L'_{\theta_k}\|_1$

Q: Show that linearizing PDs by storing their projections onto the diagonal is a stable, but not injective, method.
Q: What happens in general when one embeds PDs in Hilbert?

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

$$\exists 0 < A, B < +\infty \text{ s.t. } A d(\cdot, \cdot) \leq d'(\cdot, \cdot) \leq B d(\cdot, \cdot)$$
The space of persistence diagrams

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

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

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

Prop: $\mathcal{H}$ Hilbert with dot product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ and distance $\| \cdot \|_{\mathcal{H}}$. Assume $d_{\mathcal{H}}$ and $d_\infty$ or $d_p$ are equivalent.

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

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

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

when $L, N \to +\infty$
Q: What happens in general when one embeds PDs in Hilbert?

**Def:** Two metrics \(d, d'\) are **equivalent** if

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

Proof:

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

Consider \(S = \{D_u\}_{u \in \{0,1\}^N}\)

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

\(S\) is not countable with \(d_1\)
Q: What happens in general when one embeds PDs in Hilbert?

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

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

Proof:

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

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

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

Proof:

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

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

$$\forall D_u \in S, \exists D_{u'} \in S' : d_1(D_u, D_{u'}) \leq \epsilon$$
The space of persistence diagrams

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

Def: Two metrics $d, d'$ are equivalent if
\[ \exists 0 < A, B < +\infty \text{ s.t. } A d(\cdot, \cdot) \leq d'(\cdot, \cdot) \leq B d(\cdot, \cdot) \]

Proof:
\[ S = \{ D_u \}_{u \in \{0,1\}^N} \]

Indeed, let $S' \subseteq S$ be a dense set and $\epsilon > 0$
\[ \forall D_u \in S, \exists D_{u'} \in S' : d_1(D_u, D_{u'}) \leq \epsilon \]

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

[On the Metric Distortion of Embedding Persistence Diagrams into separable Hilbert Spaces, Bauer, Carrière, 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

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

**Proof:**

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

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

$$\forall D_u \in S, \exists D_{u'} \in S' : d_1(D_u, D_{u'}) \leq \epsilon$$

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

$$\Rightarrow \text{card}(S') \geq \text{card}(S/ \sim)$$

where $D_u \sim D_v \Leftrightarrow \text{supp}(u) \triangle \text{supp}(v) < \infty$
The space of persistence diagrams

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

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

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

Proof:

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

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

$$\forall D_u \in S, \exists D_{u'} \in S' : d_1(D_u, D_{u'}) \leq \epsilon$$

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

$$\Rightarrow \text{card}(S') \geq \text{card}(S/\sim)$$

uncountable!

where $D_u \sim D_v \Leftrightarrow \text{supp}(u) \triangle \text{supp}(v) < \infty$
Q: What happens in general when one embeds PDs in Hilbert?

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

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

Ex: Persistence surface

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

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

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

Ex: Persistence surface

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

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

If $\alpha \geq 2$, $S$ is in the domain of $\Phi$
The space of persistence diagrams

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

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

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

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

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

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

$\dim_A(D, d_p) = +\infty$ 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

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

Proof:

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

The number of such diagrams increases to $+\infty$ as $\beta$ 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

[On the Metric Distortion of Embedding Persistence Diagrams into separable Hilbert Spaces, Bauer, Carri`ere, SoCG, 2019]
The space of persistence diagrams

Illustrations:

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

Idea: Stay in Euclidean space $\mathbb{R}^d$ but learn best vectorization with Neural Net

[On the Metric Distortion of Embedding Persistence Diagrams into separable Hilbert Spaces, Bauer, Carrière, SoCG, 2019]
Neural Networks

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

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

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

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

Final classifier: \(F_\theta = f_{\theta_n}^{(n)} \circ \cdots \circ f_{\theta_0}^{(0)}\)
Neural Networks

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

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

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

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

Backpropagation: for each $k$:
1. compute $\nabla \ell(\theta_k)$ with chain rule
2. update $\theta_k := \theta_k - \eta \nabla \ell(\theta_k)$
Neural Networks

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

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

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. $\theta_k$ and $x$
The Deep Set architecture

Deep Set is a novel neural net architecture that is able to handle sets instead of finite dimensional vectors

Input: \( \{x_1, ..., x_n\} \subset \mathbb{R}^d \) instead of \( x \in \mathbb{R}^d \)
Deep Set is a novel neural net architecture that is able to handle sets instead of finite dimensional vectors.

**Input:** \( \{x_1, \ldots, x_n\} \subset \mathbb{R}^d \) instead of \( x \in \mathbb{R}^d \)

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

\[ \Rightarrow F(\{x_1, \ldots, x_n\}) = F(\{x_{\sigma(1)}, \ldots, x_{\sigma(n)}\}), \forall \sigma \]

In practice: \( \phi(x_i) = W \cdot x_i + b \)
The Deep Set architecture

Deep Set is a novel neural net architecture that is able to handle sets instead of finite dimensional vectors

Input: \( \{x_1, \ldots, x_n\} \subset \mathbb{R}^d \) instead of \( x \in \mathbb{R}^d \)

Network is permutation invariant: \( F(X) = \rho \left( \sum_i \phi(x_i) \right) \)

Universality theorem

**Thm:**

A function \( f \) is permutation invariant iif \( f(X) = \rho \left( \sum_i \phi(x_i) \right) \)

for some \( \rho \) and \( \phi \), whenever \( X \) is included in a *countable* space
Application to PDs
Application to PDs

Permutation invariant layers generalize several TDA approaches
Application to PDs

Permutation invariant layers generalize several TDA approaches

→ persistence images
Application to PDs

Permutation invariant layers generalize several TDA approaches

→ persistence images  → landscapes
Application to PDs

Permutation invariant layers generalize several TDA approaches

→ persistence images → landscapes → Betti curves

Application to PDs

Permutation invariant layers generalize several TDA approaches

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

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

Permutation invariant layers generalize several TDA approaches

→ persistence images → landscapes → Betti curves

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

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

→ persistence images → landscapes → Betti curves

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

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

$$\text{PersLay}(D) = \rho \left( \text{op}\{w(p) \cdot \phi(p)\}_{p \in D} \right)$$
Permutation invariant layers generalize several TDA approaches

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

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

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

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

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

Application to PDs

Permutation invariant layers generalize several TDA approaches

→ persistence images → landscapes → Betti curves

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

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

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

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

Application to PDs

Permutation invariant layers generalize several TDA approaches

→ persistence images → landscapes → Betti curves

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

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

$$\text{PersLay}(D) = \rho \left( \text{op} \left\{ w(p) \cdot \phi(p) \right\}_{p \in D} \right)$$
Application to PDs

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

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

[PerSLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures, Carrière, Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]
Application to PDs

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

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

$$\text{op} = \text{sum}$$

[PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures, Carrière, Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]
Application to PDs

Parameters $\Delta_1, \ldots, \Delta_q \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$

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

$\phi_L : p \mapsto \left\lceil \langle p, e_{\Delta_1} \rangle + b_{\Delta_1} \right\rceil$

$\ldots$

$\left\lceil \langle p, e_{\Delta_q} \rangle + b_{\Delta_q} \right\rceil$

$w(p) = 1$

$op = top-k$

PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures, Carrière, Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019
Application to PDs

[PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures, Carrère, Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]
Application to PDs

from perslay.perslay import perslay_channel

perslay_parameters = {
    "layer": "im",
    "image_size": (20, 20),
    "perm_op": "sum"
}

perslay_channel(output = list_v, name = "perslay", diag = YOUR_DIAGS, **self.perslay_parameters)

# outputs
# name of this layer
# diagrams

[PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures, Carrière, Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]
Application to graph classification

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

$$D \text{ its degree matrix}$$

and $L_w(G) = I - D^{-1/2}AD^{-1/2}$ its normalized Laplacian.
Application to graph classification

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

$$D \text{ its degree matrix}$$

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 $0 \leq \lambda_1 \leq \cdots \leq \lambda_n \leq 2$
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.

$L_w(G)$ decomposes on an 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 $t$:

$$hks_{G,t}: v \mapsto \sum_{k=1}^{n} \exp(-\lambda_k t)\phi_k(v)^2$$
Application to graph classification

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

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

[PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures, Carrière, Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]
Application to graph classification

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

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

Sublevel graphs (increasing values of hks)

[PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures, Carrière, Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]
**Def:** Let $t \geq 0$, and define the *Heat Kernel Signature* of param $t$:

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

**Application to graph classification**

[PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures, Carrière, Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]
Application to graph classification

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

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

[PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures, Carrière, Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]
Application to graph classification

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

$$hks_{G,t} : v \mapsto \sum_{k=1}^{n} \exp(-\lambda_k t)\phi_k(v)^2$$
Application to graph classification

Graph from the PROTEINS dataset

Corresponding extended persistence diagram

[PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures, Carrière, Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019]
Application to graph classification

PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures, Carrière, Chazal, Ike, Lacombe, Royer, Umeda, AISTATS, 2019

Weight function learnt (after training on the MUTAG dataset)
Recall: persistence diagrams and machine learning

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

Etc.

$k(\cdot, \cdot) := \langle \Phi(\cdot), \Phi(\cdot) \rangle_{\mathcal{H}}$

What filtration to choose?

What linearization to choose?
Problem setting

Q: How to define $\nabla D$?
Problem setting

**Q:** How to define $\nabla D$?

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

**Q:** Given a point cloud $X \subseteq \mathbb{R}^d$, how to define $\nabla_X D_k(\text{Rips}(X))$?
Problem setting

Q: How to define $\nabla D$?

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

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

Idea: Let’s go back to the PD construction...
Computation with matrix reduction

Input: simplicial filtration

(Persistent) homology can be computed by using the fact that each simplex is either:
	negative, i.e., it destroys an homology class

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

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Computation with matrix reduction

Input: simplicial filtration
given as boundary matrix

\[
\begin{array}{ccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 \\
\hline
1 & & & & & & \\
2 & & & & & & \\
3 & & & & & & \\
4 & & & & & & \\
5 & & & & & & \\
6 & & & & & & \\
7 & & & & & & \\
\end{array}
\]
Computation with matrix reduction

Input: simplicial filtration

given as *boundary matrix*

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td>♦</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>♦</td>
<td>♦</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>♦</td>
<td></td>
<td></td>
<td>♦</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Diagram of a triangle](image_url)
Computation with matrix reduction

Input: simplicial filtration
given as *boundary matrix*

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>⋄</td>
<td>⋄</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>⋄</td>
<td>⋄</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>⋄</td>
<td>⋄</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Diagram of a simplicial complex](image)
Computation with matrix reduction

Input: simplicial filtration
given as *boundary matrix*

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>•</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>•</td>
<td>•</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td>•</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td>•</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td>•</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>•</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Diagram:

```
    3
   /   
  6    5
   
  7
```

Diagram:

```
    3
   /   
  6    5
   
  7
```
Computation with matrix reduction

Input: simplicial filtration
given as boundary matrix

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

for j=1 to m do:
while \( \exists k < j \) s.t. \( \text{low}(k) = \text{low}(j) \) do:
\[ \text{col}(j) = \text{col}(j) + \text{col}(k) \]
Computation with matrix reduction

Input: simplicial filtration

given as boundary matrix

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

for \( j = 1 \) to \( m \) do:

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

\[
\text{col}(j) = \text{col}(j) + \text{col}(k)
\]

\( \text{low}(j) = j' \)
Computation with matrix reduction

Input: simplicial filtration
given as boundary matrix

\[
\begin{array}{ccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 \\
1 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
2 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
3 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
4 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
5 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
6 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
7 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
\end{array}
\]

for \( j = 1 \) to \( m \) do:
\[
\text{while } \exists k < j \text{ s.t. } \text{low}(k) = \text{low}(j) \text{ do:}
\]
\[
\text{col}(j) = \text{col}(j) + \text{col}(k)
\]
\[
6 = 6 + 5
\]
\[
\text{low}(j) = j'
\]
Computation with matrix reduction

Input: simplicial filtration

given as boundary matrix

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

6 = 6+5

low(j) = j'

6 = 6+5
Computation with matrix reduction

Input: simplicial filtration

given as boundary matrix

\[
\begin{array}{cccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 \\
1 & & & & & & \\
2 & & & & & & \\
3 & & & & & & \\
4 & & & & & & \\
5 & & & & & & \\
6 & & & & & & \\
7 & & & & & & \\
\end{array}
\]

\[
\begin{array}{cccc}
5 & 6 & & \\
4 & 6 & & \\
& & & \\
& & & \\
& & & \\
& & & \\
& & & \\
\end{array}
\]

\[
\begin{array}{cccc}
6 & 5 & & \\
6 & 4 & & \\
& & & \\
& & & \\
& & & \\
& & & \\
& & & \\
\end{array}
\]

for \( j=1 \) to \( m \) do:

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

\[
\text{col}(j) = \text{col}(j) + \text{col}(k)
\]

\[
\text{low}(j) = j'
\]

\[
6 = 6+5 \\
6 = 6+4
\]
Computation with matrix reduction

Input: simplicial filtration

given as *boundary matrix*

\[
\begin{array}{ccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 \\
1 & & & & & & \\
2 & & & & & & \\
3 & & & & & & \\
4 & & & & & & \\
5 & & & & & & \\
6 & & & & & & \\
7 & & & & & & \\
\end{array}
\]

for \( j = 1 \) to \( m \) do:

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

\[
\text{col}(j) = \text{col}(j) + \text{col}(k)
\]

\[
6 = 6 + 5
\]

\[
6 = 6 + 4
\]

\[
\text{low}(j) = j'
\]
Computation with matrix reduction

Input: simplicial filtration
Output: boundary matrix

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td>*</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>*</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td>*</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>*</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>*</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Computation with matrix reduction

Input: simplicial filtration

Output: boundary matrix
reduced to column-echelon form

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>*</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>*</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td>*</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td>*</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>*</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>*</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>*</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>*</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>*</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Computation with matrix reduction

Input: simplicial filtration
Output: boundary matrix

- Simplex pairs give finite intervals: [2, 4), [3, 5), [6, 7)
- Unpaired simplices give infinite intervals: [1, +∞)

\[
\begin{array}{ccccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
1 & & & * & * & & & \\
2 & & * & * & & & & \\
3 & & & * & * & & & \\
4 & & & & * & & & \\
5 & & & & & * & & \\
6 & & & & & & * & \\
7 & & & & & & & \\
\end{array}
\]

\[
\begin{array}{ccccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
1 & 1 & & * & & & & \\
2 & & 1 & & * & & & \\
3 & & & 1 & & & & \\
4 & & & & 1 & & & \\
5 & & & & & 1 & & \\
6 & & & & & & 1 & \\
7 & & & & & & & \\
\end{array}
\]
Computation with matrix reduction

Input: simplicial filtration

Output: boundary matrix

reduced to column-echelon form

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

Box: unpaired simplices give infinite intervals: [1, +∞)

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

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

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

Point cloud $\hat{X}_n$

Persistence barcode
Example: Vietoris-Rips gradient

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

$$\mathcal{F}(\sigma) = \max_{i,j} \|v_i - v_j\|$$
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 $p = (\mathcal{F}(\sigma_+), \mathcal{F}(\sigma_-)) \in D_k(\text{Rips}(X))$
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 $p = (\mathcal{F}(\sigma_+), \mathcal{F}(\sigma_-)) \in D_k(\text{Rips}(X))$

with $\sigma_+ = \{v_0, \ldots, v_k\}$ and $\sigma_- = \{w_0, \ldots, w_{k+1}\}$
Example: Vietoris-Rips gradient

Let \( p = (\mathcal{F}(\sigma_+), \mathcal{F}(\sigma_-)) \in D_k(\text{Rips}(X)) \)

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

\[
\nabla X p = \left[ \frac{\partial}{\partial X} \|v_i^* - v_j^*\|, \frac{\partial}{\partial X} \|w_a^* - w_b^*\| \right]
\]
Example: Vietoris-Rips gradient

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

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

Let \( p = (\mathcal{F}(\sigma_+), \mathcal{F}(\sigma_-)) \in D_k(\text{Rips}(X)) \)

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

With this gradient rule, one can do gradient descent with any function of persistence!
Example: Vietoris-Rips gradient

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 \( p \in \mathcal{D}_1(\text{Rips}(X)) \)
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

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

with $p \in D_1(\text{Rips}(X))$.
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

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

with \( p \in D_1(\text{Rips}(X)) \) and \( C \) unit square.
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

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

with \( p \in D_1(\text{Rips}(X)) \) and \( C \) unit square.
Example: Sublevel sets

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

$$F(\sigma) = \max_i f_\theta(v_i)$$
Example: Sublevel sets

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

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

$$\nabla_\theta p = \left[ \frac{\partial}{\partial \theta} f_\theta(v_{i^*}), \frac{\partial}{\partial \theta} f_\theta(w_{\alpha^*}) \right]$$
Example: Sublevel sets

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

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

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

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

$$F(\sigma) = \max_i f_\theta(v_i)$$

$$\nabla_\theta p = \left[ \frac{\partial}{\partial \theta} f_\theta(v_i^*), \frac{\partial}{\partial \theta} f_\theta(w_a^*) \right]$$

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

We can use gradient descent to minimize loss

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

with $p \in D_0(I)$
Example: Sublevel sets

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

$$F(\sigma) = \max_i f_\theta(v_i)$$

$$\nabla_\theta p = \left[ \frac{\partial}{\partial \theta} f_\theta(v_i^*), \frac{\partial}{\partial \theta} f_\theta(w_a^*) \right]$$

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

We can use gradient descent to minimize loss

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

with $p \in D_0(I)$
Example: Sublevel sets

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

$$ F(\sigma) = \max_i f_\theta(v_i) $$

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

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

We can use gradient descent to minimize loss

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

with $p \in D_0(I)$
Example: Sublevel sets

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

$$F(\sigma) = \max_i f_\theta(v_i)$$

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

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 $p \in D_0(I)$
Topological gradient descent

[Optimizing persistent homology based functions, Carrière, Chazal, Glisse, Ike, Kanna, Umeda, 2021]
Topological gradient descent

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

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

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 $\Phi : A \to \mathbb{R}^{\vert K \vert}$ a (parameterized) filtration of $K$. There exists a partition $A = S \sqcup O_1 \sqcup \cdots \sqcup O_k$ s.t. all the restrictions $\Phi : O_i \to \mathbb{R}^{\vert K \vert}$ 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

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 $\Phi : A \rightarrow \mathbb{R}^{|K|}$ a (parameterized) filtration of $K$. There exists a partition $A = S \sqcup O_1 \sqcup \cdots \sqcup O_k$ s.t. all the restrictions $\Phi : O_i \rightarrow \mathbb{R}^{|K|}$ 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.

**Q:** What is $S$ for Vietoris-Rips? Sublevel sets?
Topological gradient descent

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

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

where $\text{conv}$ denotes the convex hull.
Topological gradient descent

Let \( \{\alpha_k\}_k, \{\zeta_k\}_k \) s.t.

\[
\alpha_k \geq 0, \sum_k \alpha_k = +\infty \text{ and } \sum_k \alpha_k^2 < +\infty
\]

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

**Thm:** As long as \( \mathcal{L} \circ \text{Pers} \circ \Phi \) is locally Lipschitz, the sequence

\[
a_{k+1} = a_k - \alpha_k (g_k + \zeta_k),
\]

where \( g_k \in \partial_{\alpha_k} (\mathcal{L} \circ \text{Pers} \circ \Phi) \), converges to a critical point of \( \mathcal{L} \circ \text{Pers} \circ \Phi \).
Example: filter selection

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

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

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:

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

one can also use Sliced Wasserstein for speedup.
Example: filter selection

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:

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

one can also use Sliced Wasserstein for speedup.

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

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

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

Topological Data Analysis is:

a mathematically grounded framework...

\[ H_k = \frac{Z_k}{B_k} \]

...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