Matematica Discreta e Applicazioni

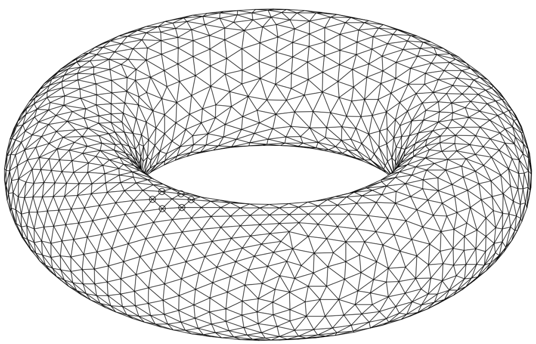
Topological Data Analysis

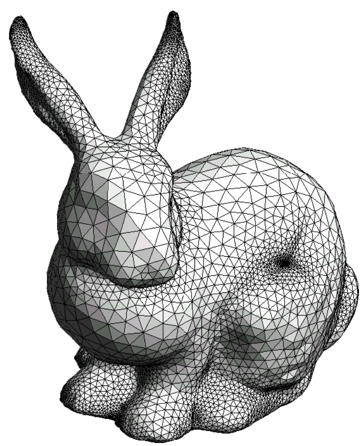
Ulderico Fugacci

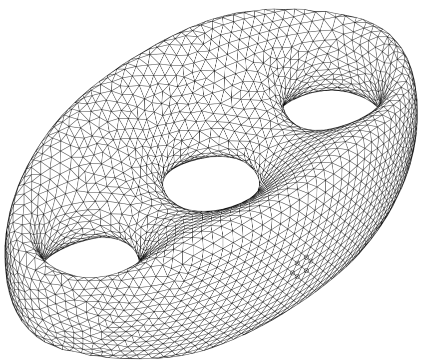
CNR - IMATI

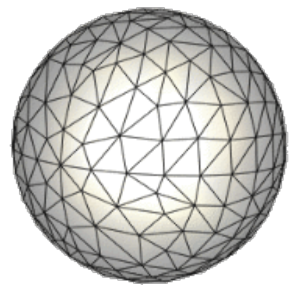
Topological Data Analysis

Topology *describes*, *characterizes,* and *discriminates* ***shapes*** by studying their properties that are preserved under ***continuous deformations***, such as ***stretching*** and ***bending***, but ***not tearing*** or ***gluing***





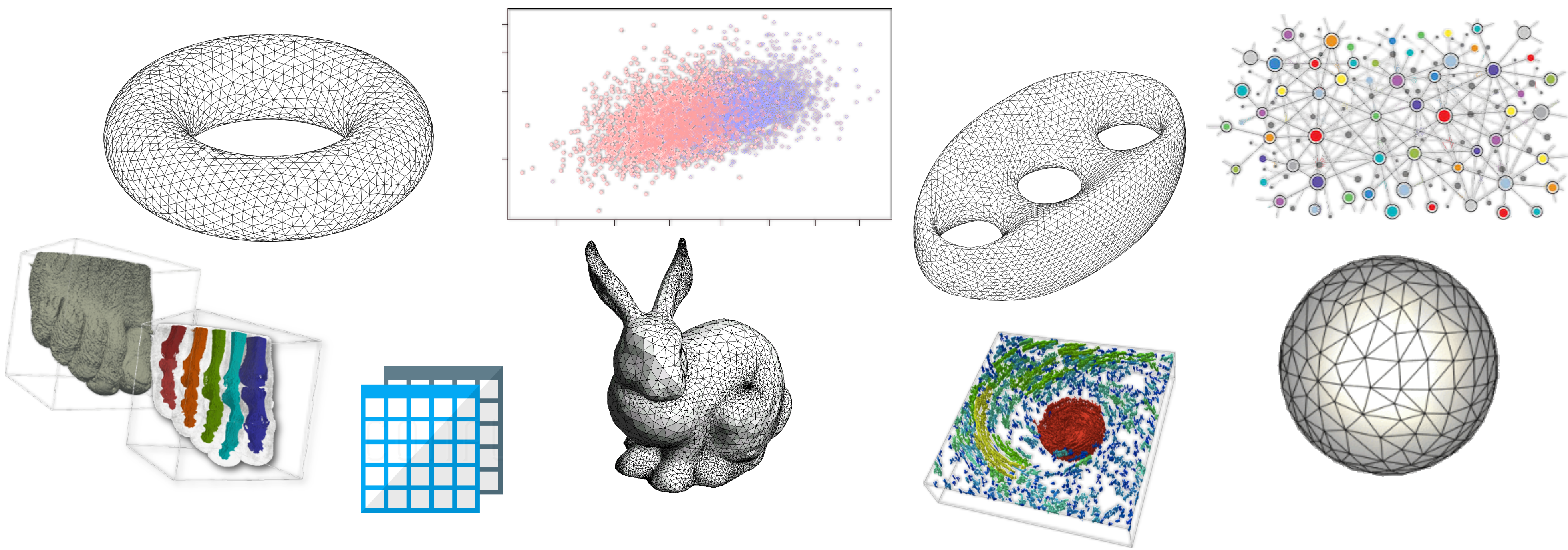




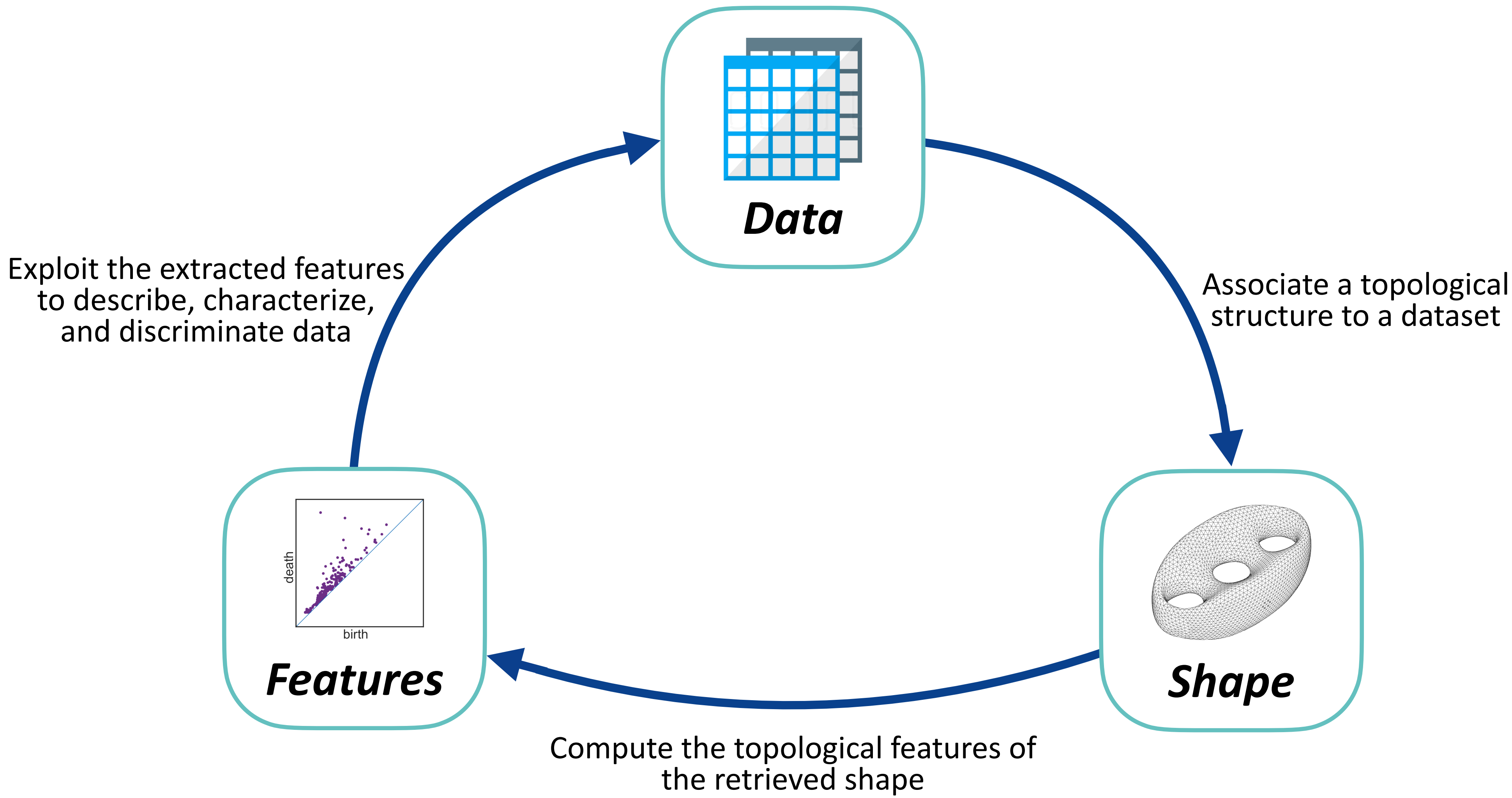
Topological Data Analysis

Assumption in TDA: ***Any data*** can be endowed with a ***shape***.

So, any data can be studied in terms of its ***topological features***



Topological Data Analysis



Topological Data Analysis

Outline:

The Notion of Shape

Simplicial Complexes

Simplicial Homology

From Data to Complexes

Persistent Homology

Visualizing Persistence

Persistence & Stability

Computing Persistence

Data Structures

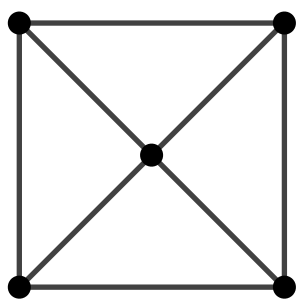
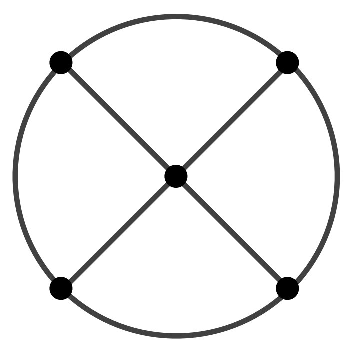
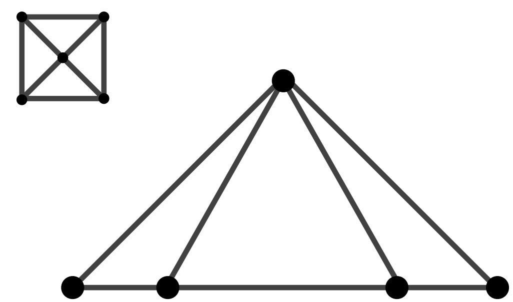
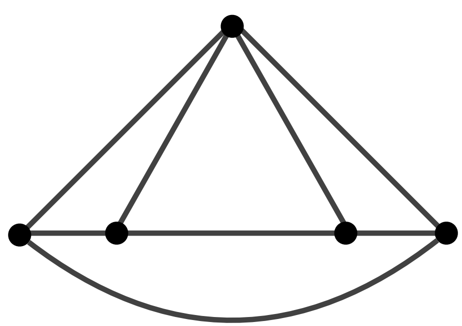
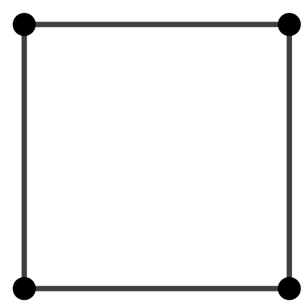
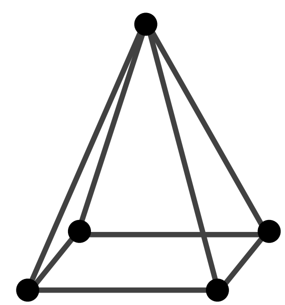
The Notion of Shape

Geometry or Topology?

Which of these domains look similar?

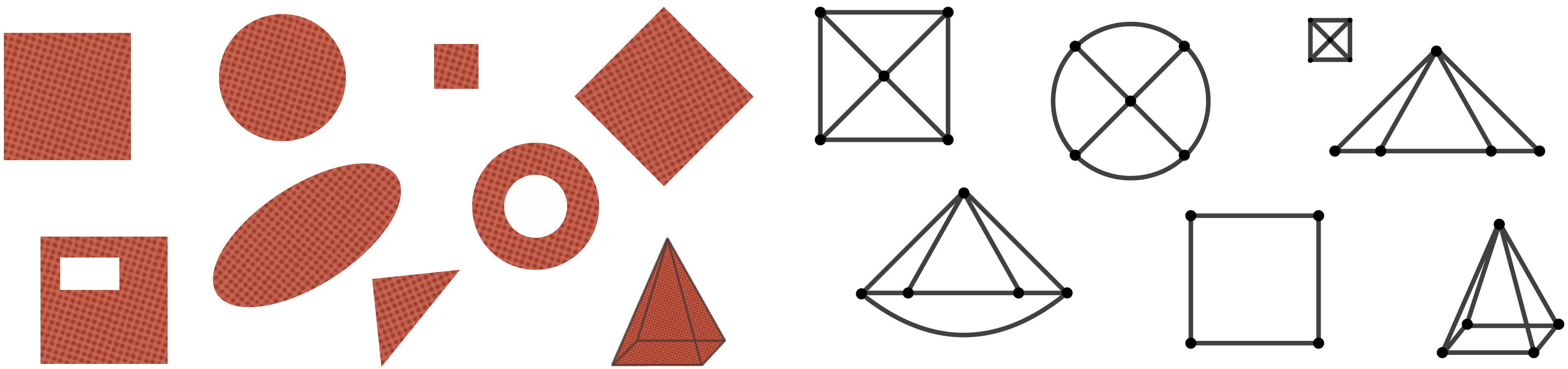
Geometry or Topology?

And what about these ones?

Geometry or Topology?

The answer depends on the *point of view* we adopt

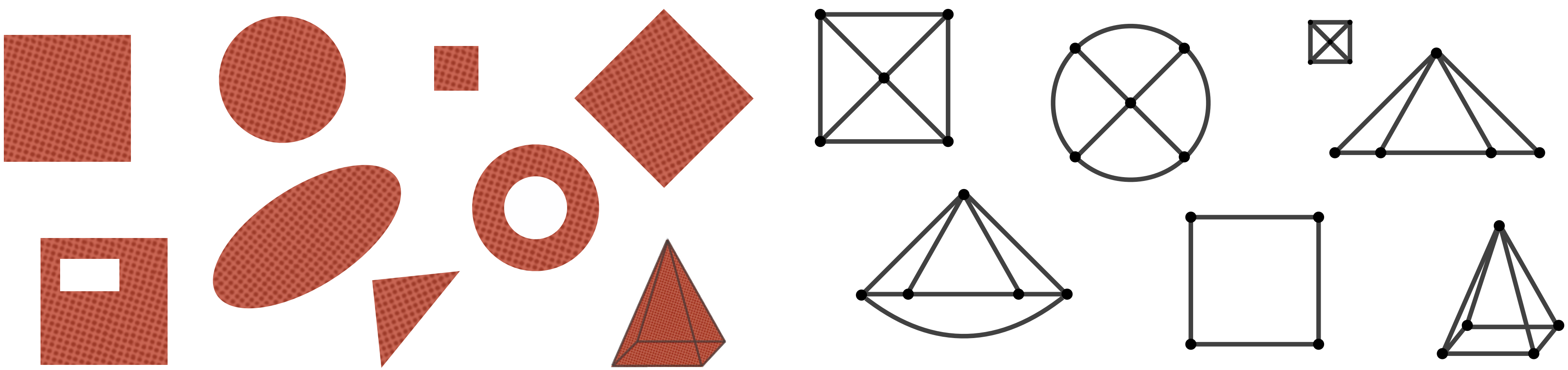


*Geometry cares about those properties which **change** when an object is continuously **deformed***

E.g. length, area, volume, angles, curvature, …

Geometry or Topology?

The answer depends on the *point of view* we adopt



Topology

~~Geometry~~ cares about those properties which *do not* ***change***
when an object is continuously ***deformed***

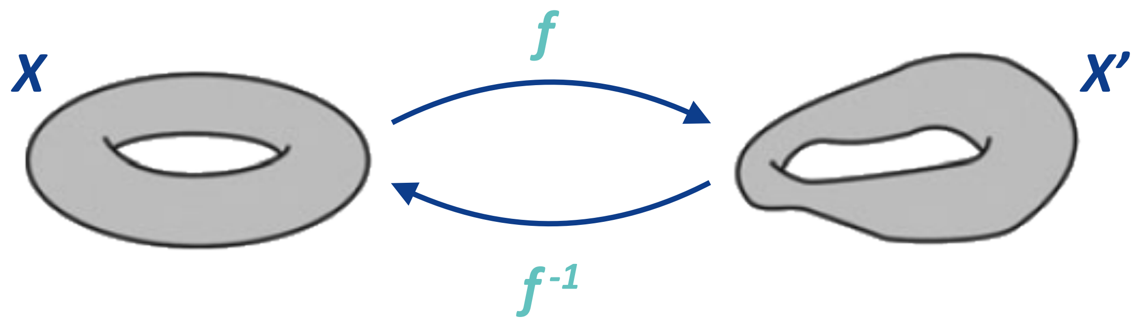
E.g. connectivity, orientation, manifoldness, …

Homeomorphisms

Definition:

Given two topological spaces (X, T) and (X', T'),
a function f: X ⟶ X' is called ***homeomorphism*** if:

- f is a ***bijection***
- f is ***continuous***
- f^{-1} is ***continuous***

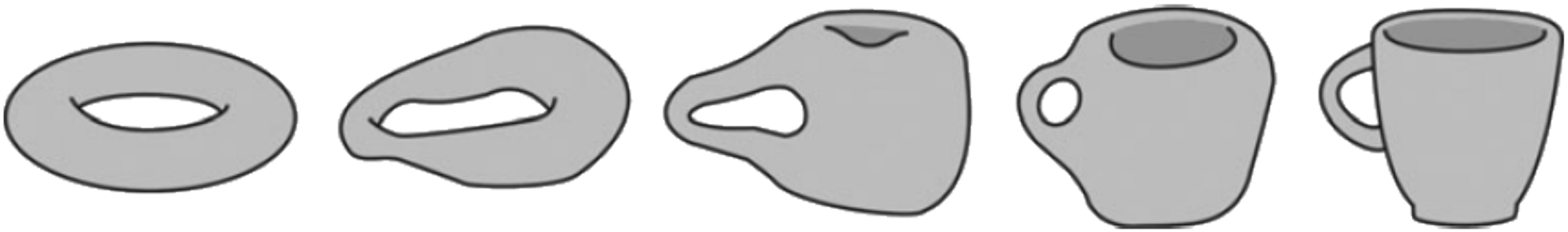


Two topological spaces (X, T) and (X', T') are ***homeomorphic*** and denoted $X \cong X'$ if there exists a homeomorphism f: X ⟶ X'

Homeomorphisms induce an ***equivalence relation*** of topological spaces partitioning them into equivalence classes

Homeomorphisms

Intuitively:



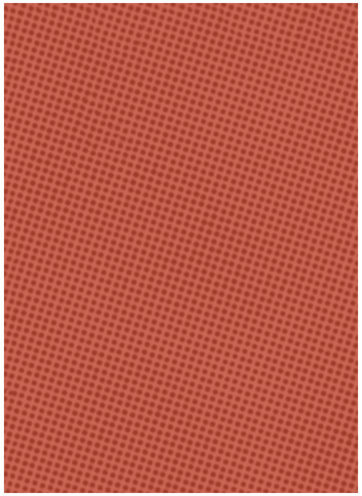
The notion of homeomorphism captures the idea of continuous deformation

$\cong$

Homeomorphisms

Intuitively:

One can:

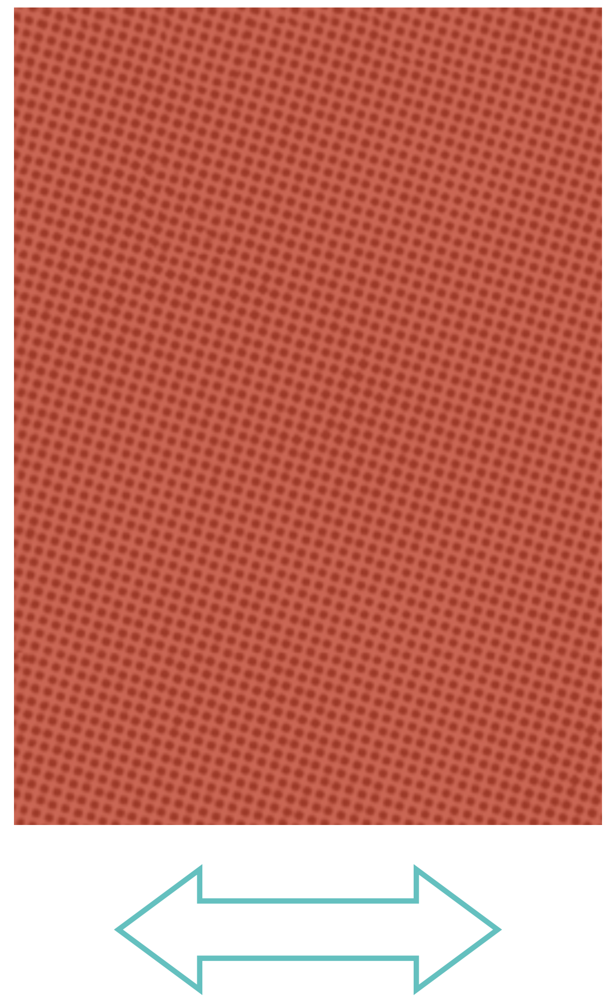


Homeomorphisms

Intuitively:

One can:

- Stretch

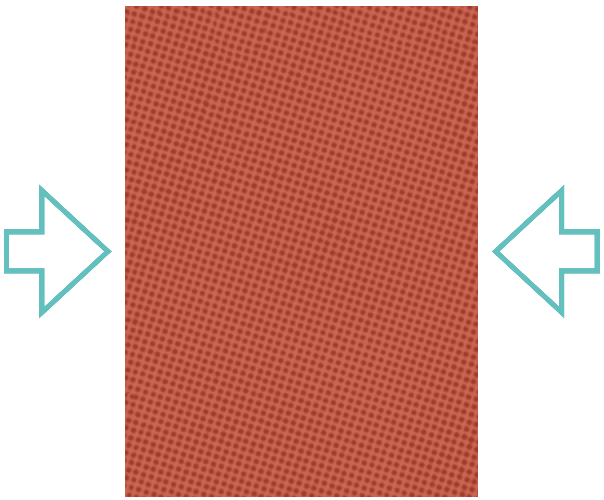


Homeomorphisms

Intuitively:

One can:

- Stretch
- Compress



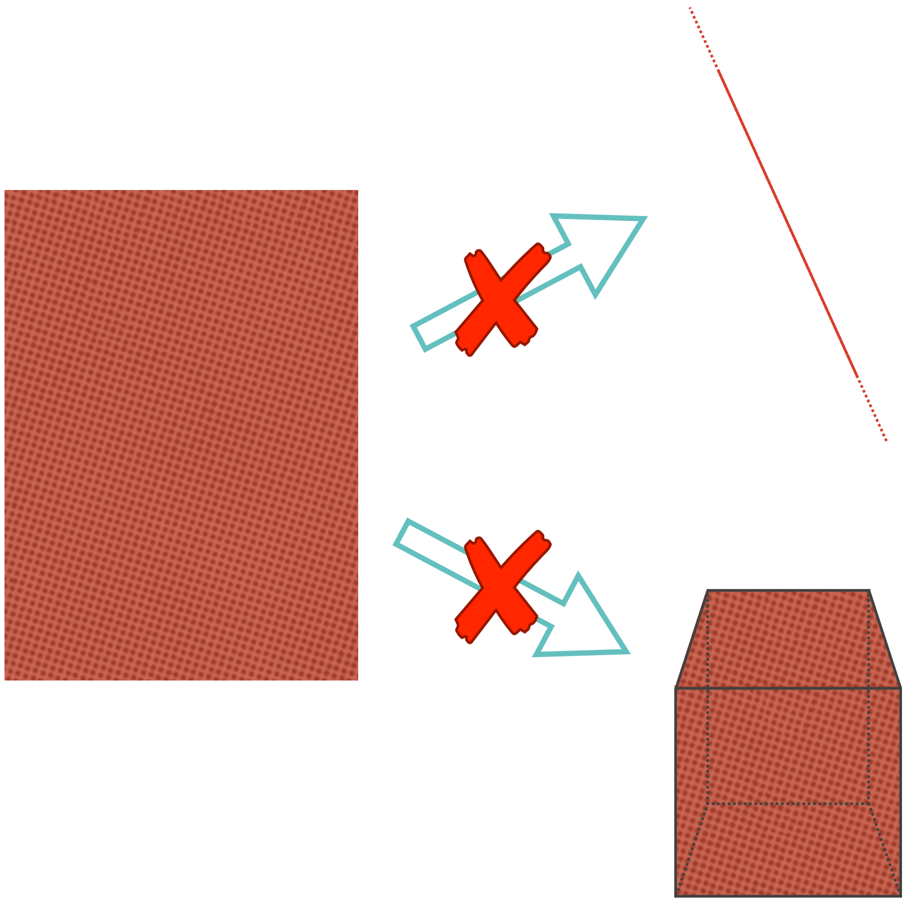
Homeomorphisms

Intuitively:

One can:

- Stretch
- Compress

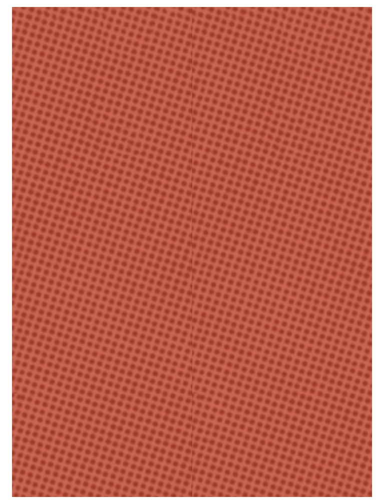
But not too much!



Homeomorphisms

Intuitively:

Moreover:

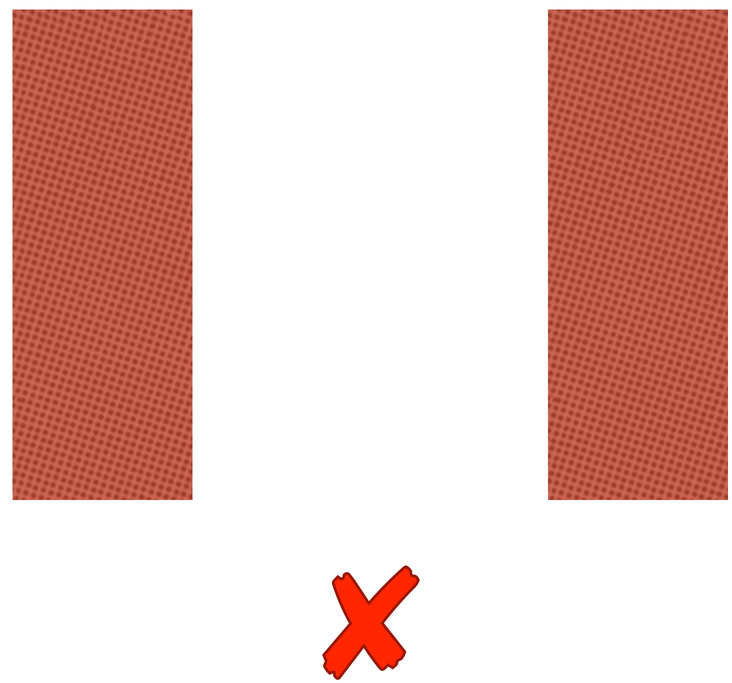


Homeomorphisms

Intuitively:

Moreover:

- No Cut

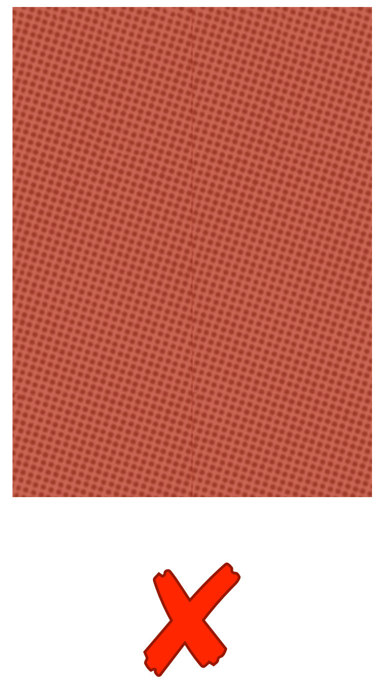


Homeomorphisms

Intuitively:

Moreover:

- No Cut
- No Glue

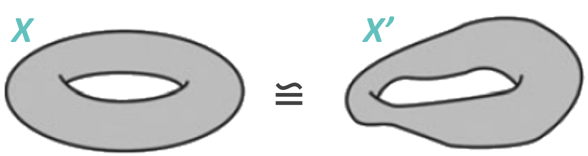


Topological Invariants

Definition:

I is a ***topological invariant*** if, given two topological spaces (X, T) and (X', T'),

X is *homeomorphic* to *X'*



X and *X'* have the *same topological invariant*

$$I(X) = I(X')$$

Some classical topological invariants:

- *Connectedness*
- *Compactness*
- *Manifoldness*
- *Orientability*
- *Euler characteristic*
- *Homology*
- *Homotopy*

Topological Invariants

Question:

Is there a "perfect" topological invariant I such that

$X \cong X'$ if and only if $I(X) = I(X')$?

Topological Invariants

Question:

Is there a "perfect" topological invariant I such that

$X \cong X'$ if and only if $I(X) = I(X')$?

Let us **simplify the question** and let focus on:

- *Considering a specific topological invariant I (e.g. the **homology**)*
- *Completely characterizing just the **spheres** $S^n := \{ x \in \mathbb{R}^n : |x| = 1 \}$*

The above question turns into the following:

If X and S^n have the same homology, then $X \cong S^n$?

Topological Invariants

Question:

Is there a "perfect" topological invariant I such that

$X \cong X'$ if and only if $I(X) = I(X')$?

Let us **simplify the question** and let focus on:

- *Considering a specific topological invariant I (e.g. the **homology**)*
- *Completely characterizing just the **spheres** $S^n := \{ x \in \mathbb{R}^n : |x| = 1 \}$*

The above question turns into the following:

If X and S^n have the same homology, then $X \cong S^n$?

NO

Topological Invariants

But:

*Replacing homology with **homotopy**, the answer is positive!*

Topological Invariants

But:

*Replacing homology with **homotopy**, the answer is positive!*

Poincaré Conjecture *(3rd Millennium Prize Problem)**:***

If X is a closed n-manifold homotopy equivalent to S^n, then $X \cong S^n$

Proven by Grigori Perelman in 2003

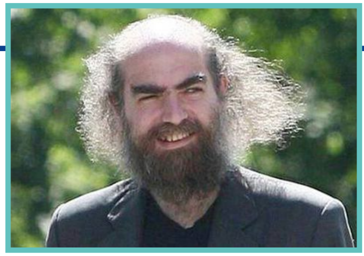
Topological Invariants

But:

*Replacing homology with **homotopy**, the answer is positive!*

Poincaré Conjecture *(3rd Millennium Prize Problem)***:**

If X is a closed n-manifold homotopy equivalent to S^n, then $X \cong S^n$



Proven by Grigori Perelman in 2003

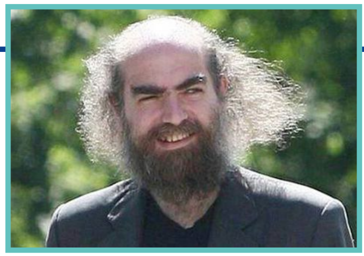
Topological Invariants

But:

Replacing homology with ***homotopy****, the answer is positive!*

Poincaré Conjecture *(3rd Millennium Prize Problem)****:***

If X is a closed n-manifold homotopy equivalent to S^n, then $X \cong S^n$



Proven by Grigori Perelman in 2003

So: *Why we will mainly focus on homology rather than homotopy?*

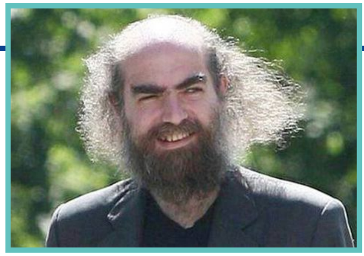
Topological Invariants

But:

*Replacing homology with **homotopy**, the answer is positive!*

Poincaré Conjecture *(3rd Millennium Prize Problem)***:**

If X is a closed n-manifold homotopy equivalent to S^n, then $X \cong S^n$



Proven by Grigori Perelman in 2003

So: *Why we will mainly focus on homology rather than homotopy?*

Because, in practice, computing homotopy groups is nearly impossible!

Bibliography

Some References:

- ***Books on TDA:***
  - A. J. Zomorodian. ***Topology for computing.*** Cambridge University Press, 2005.
  - H. Edelsbrunner, J. Harer. ***Computational topology: an introduction***. American Mathematical Society, 2010.
  - R. W. Ghrist. ***Elementary applied topology***. Seattle: Createspace, 2014.

- ***Papers on TDA:***
  - G. Carlsson. ***Topology and data***. Bulletin of the American Mathematical Society 46.2, pages 255-308, 2009.

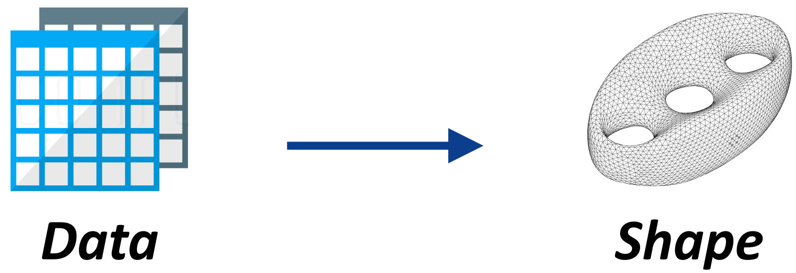
- ***Intro to (Algebraic) Topology:***
  - E. Sernesi. ***Geometria 2***. Bollati Boringhieri, Torino, 1994.
  - A. Hatcher. ***Algebraic topology.*** Cambridge University Press, 2002.

Simplicial Complexes

Complexes & Data

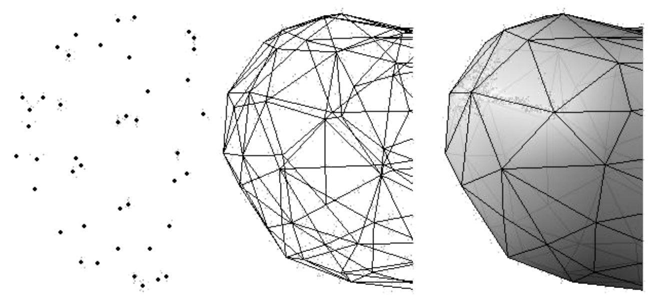
Goal:

We want to associate a topological structure to a given dataset



Due to the nature of data and to our computational ambitions, datasets will be represented by ***"discrete"*** structures

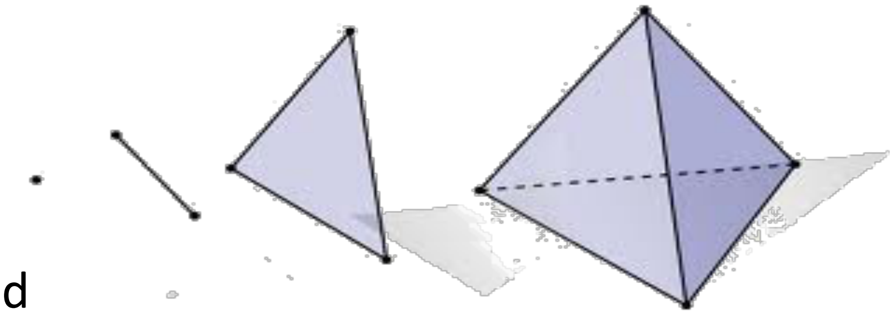
Among various possibilities, ***simplicial complexes*** represent the most suitable choice



In fact, simplicial complexes are able to deal with data:

- of ***large size*** (e.g. consisting of a huge number of samples)
- of ***high dimension*** (e.g. involving a large number of variables or parameters)
- ***unorganized*** (e.g. not arranged in a regular grid)

Simplicial Complexes



Definitions:

A set $V := \{ v_0, v_1, \ldots, v_k \}$ of points in $\mathbb{R}^n$ is called ***geometrically independent*** if vectors $v_1 - v_0, \ldots, v_k - v_0$ are ***linearly independent*** over $\mathbb{R}$

E.g. two distinct points, three non-collinear points, four non-coplanar points

The ***k-simplex*** $\boldsymbol{\sigma = v_0 v_1 \ldots v_k}$ spanned by a geometrically independent set $V = \{v_0, v_1, \ldots, v_k\}$ of in $\mathbb{R}^n$ is the ***convex hull*** of V, i.e. the set of all points $x \in \mathbb{R}^n$ such that

$$x = \sum_{i=0}^{k} t_i v_i \quad \text{where} \quad \sum_{i=0}^{k} t_i = 1 \quad \text{and } t_i \geq 0 \text{ for all } i$$

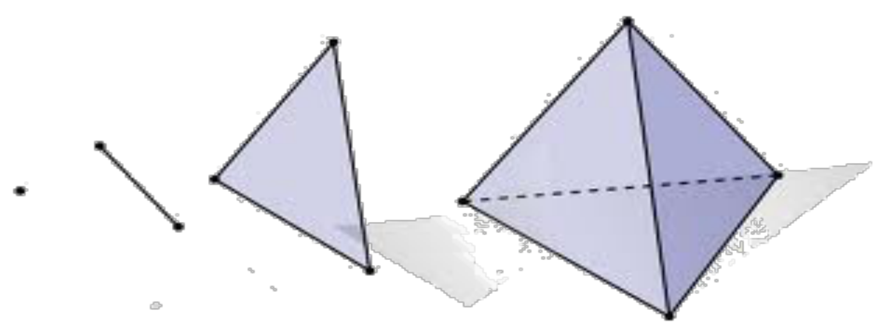
The numbers t_i are uniquely determined by x and are called ***barycentric coordinates*** of x

E.g. a 0-simplex is a vertex, a 1-simplex is an edge, a 2-simplex is a triangle, a 3-simplex is a tetrahedron

Simplicial Complexes

Definitions:

- The points v_0, v_1, ..., v_k spanning a k-simplex σ are called the ***vertices*** of σ
- k is called the ***dimension*** of σ and denoted as dim(σ)
- Any simplex τ spanned by a non-empty subset of V is called a ***face*** of σ
- Conversely, σ is called a ***coface*** of τ

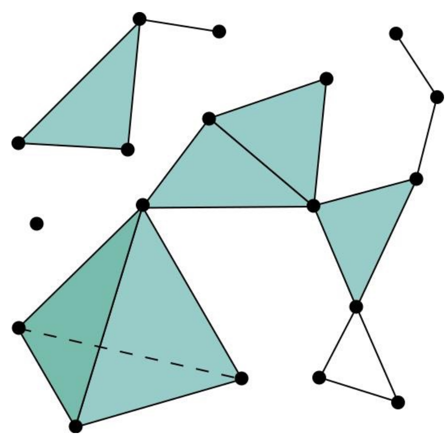


Simplicial Complexes

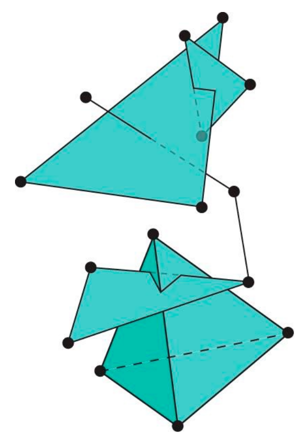
Definition:

A *(geometric)* ***simplicial complex*** K in $\mathbb{R}^n$ is a collection of simplices in $\mathbb{R}^n$ such that

- *Every face of a simplex of K is in K*
- *The non-empty intersection of any two simplices of K is a face of each of them*

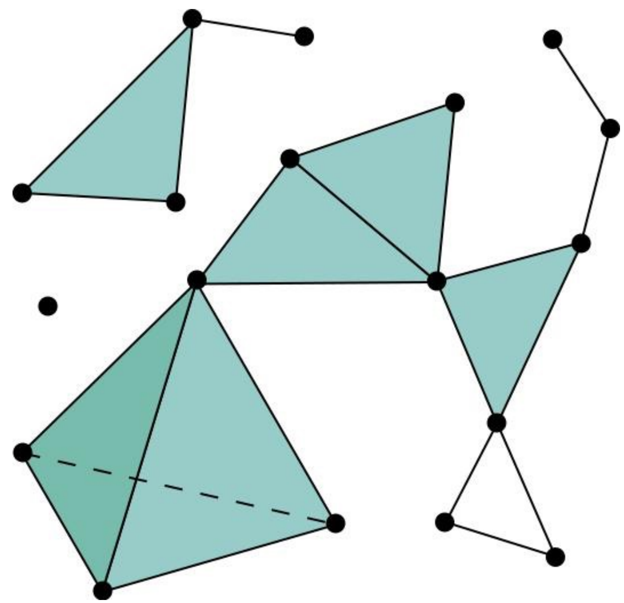


simplicial complex



non-simplicial complex

Simplicial Complexes



Definitions:

Given a (geometric) simplicial complex K in $\mathbb{R}^n$,

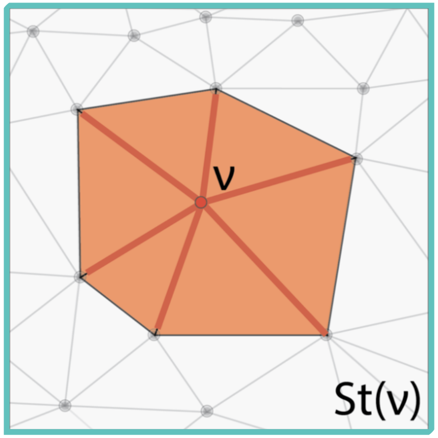
- The ***dimension*** of a simplicial complex K in $\mathbb{R}^n$, denoted as dim(K), is the supremum of the dimensions of the simplices of K
- A simplex $\boldsymbol{\sigma}$ of K such that dim($\boldsymbol{\sigma}$) = dim(K) is called ***maximal***
- A simplex $\boldsymbol{\sigma}$ of K which is not a proper face of any simplex of K is called ***top***
- A subcollection of K that is itself a simplicial complex is called a ***subcomplex*** of K

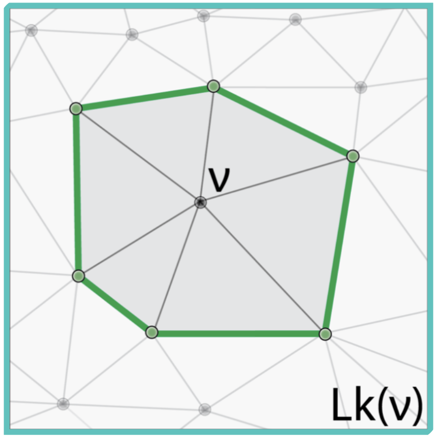
Simplicial Complexes

Definitions:

Given a simplex $\boldsymbol{\sigma}$ of a (geometric) simplicial complex K in $\mathbb{R}^n$,

- The ***star*** of $\boldsymbol{\sigma}$ is the set ***St(σ)*** of the cofaces of $\boldsymbol{\sigma}$
- The ***link*** of $\boldsymbol{\sigma}$ is the set ***Lk(σ)*** of the faces of the simplices in St($\boldsymbol{\sigma}$) such that do not intersect $\boldsymbol{\sigma}$



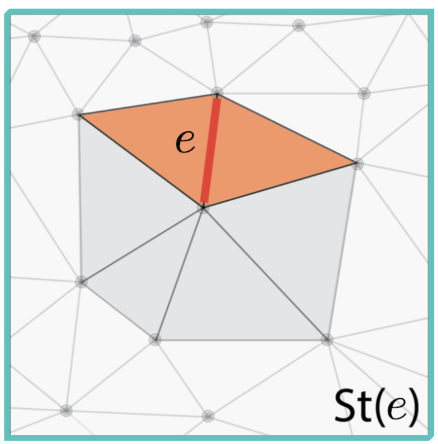


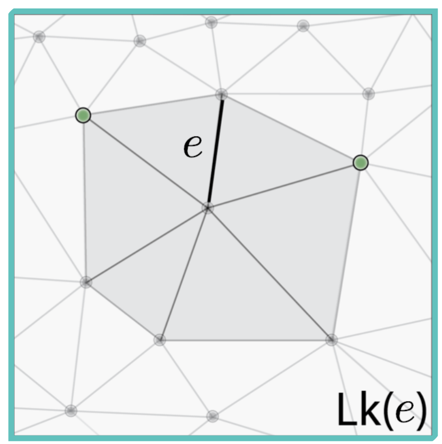
Simplicial Complexes

Definitions:

Given a simplex σ of a (geometric) simplicial complex K in $\mathbb{R}^n$,

- The ***star*** of σ is the set ***St(σ)*** of the cofaces of σ
- The ***link*** of σ is the set ***Lk(σ)*** of the faces of the simplices in St(σ) such that do not intersect σ





Simplicial Complexes

Given a (geometric) simplicial complex K in $\mathbb{R}^n$,
its ***polytope*** $|K|$ is the subset of $\mathbb{R}^n$ defined as the union of the simplices of K

The polytope $|K|$ can be endowed with ***two possible topologies*** T_1 and T_2:

- ***T_1***: A subset F of $|K|$ is a closed set of $(|K|, T_1)$ if and only if $F \cap \boldsymbol{\sigma}$ is a closed set of $(\boldsymbol{\sigma}, T_{\boldsymbol{\sigma}})$ for each $\boldsymbol{\sigma}$ in K where $T_{\boldsymbol{\sigma}}$ is the subspace topology induced on $\boldsymbol{\sigma}$ by $\mathbb{E}^n$
- ***T_2***: The subspace topology induced on $|K|$ by $\mathbb{E}^n$

In general, the two topologies T_1, T_2 are ***different***, but

Proposition: *If K is a **finite** simplicial complex, $T_1 = T_2$*

From now on, if not differently specified, we consider only *finite* simplicial complexes

Simplicial Complexes

Proposition:

*Given a simplicial complex K and a topological space (X, T), a function f from ($|K|$, T_1) to (X, T) is **continuous** if and only if $f|_\sigma$ is continuous for each $\boldsymbol{\sigma} \in K$*

Definition:

Given two simplicial complexes K and K’,

- A function f: K $\longrightarrow$ K’ is called a ***simplicial map*** if for every simplex $\boldsymbol{\sigma} = v_0v_1 \dots v_k$ in K, $f(\boldsymbol{\sigma}) = f(v_0)f(v_1)\dots f(v_k)$ is a simplex in K’
- The restriction f_V of f to the set of vertices V of K is called the ***vertex map*** of f

Simplicial Complexes

Definition:

An ***abstract simplicial complex*** K on a set V is a collection of finite non-empty subsets of V, called ***simplices***, such that if $\boldsymbol{\sigma} \in$ K and $\boldsymbol{\tau} \subseteq \boldsymbol{\sigma}$, then $\boldsymbol{\tau} \in$ K

Analogously to the case of a geometric simplicial complex,

- The elements of V are called ***vertices*** of K
- The ***dimension*** of a simplex $\boldsymbol{\sigma}$ is one less than the number of its elements
- The supremum of the dimensions of the simplices in K is called ***dimension*** of K
- Each non-empty subset $\boldsymbol{\tau}$ of a simplex $\boldsymbol{\sigma} \in$ K is called a ***face*** of $\boldsymbol{\sigma}$ and $\boldsymbol{\sigma}$ is called a ***coface*** of $\boldsymbol{\tau}$

The notions of geometric simplicial complex and abstract simplicial complex are equivalent. *More properly, it is always possible,*

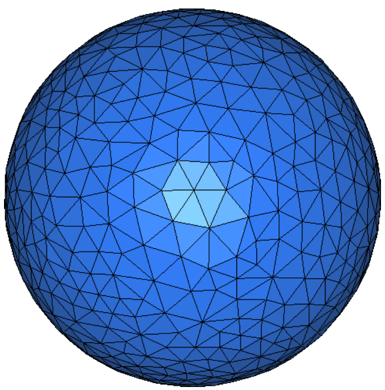
- *Given an abstract simplicial complex, to endow it with a **geometric realization***
- *Given a geometric simplicial complex, to **forget its geometry** thus obtaining an abstract simplicial complex*

Simplicial Complexes

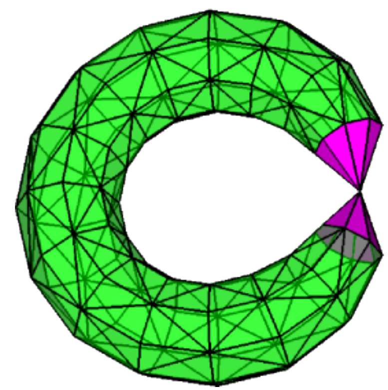
Definition: A simplicial complex K is called

- ***n-manifold [with boundary]*** if its polytope |K| is a (topological) n-manifold [with boundary]
- ***Combinatorial n-manifold [with boundary]*** if, for every vertex v, the link Lk(v) is homeomorphic to the (n − 1)-sphere S^{n-1} [or to the (n − 1)-disk $D^{n-1} := \{x \in \mathbb{R}^{n-1} : |x| \leq 1\}$]

combinatorial manifold



non-combinatorial manifold



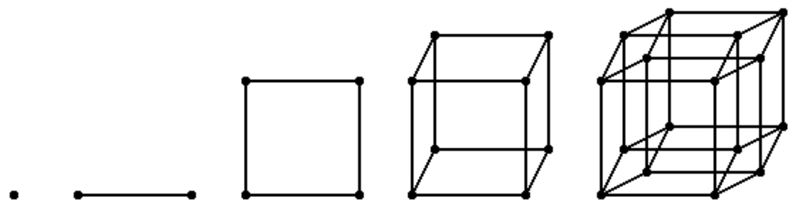
Proposition:

If K is a combinatorial n-manifold [with boundary], then K is a n-manifold [with boundary]

The converse is: ***True for n ≤ 3*** ***Open for n = 4*** ***False for n > 4***

Regular Grids

Hyper-Cube:

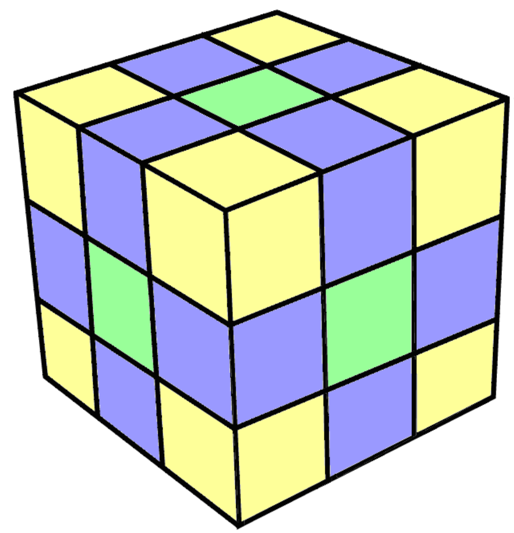


A ***k*-hyper-cube *η*** is the ***Cartesian product of k closed intervals*** of equal length

Regular Grids:

A ***regular grid H*** is a (finite) collection of hyper-cubes such that:

- *Each face of a hyper-cube of H is in H*
- *Each non-empty intersection of two hyper-cubes in H is a face of both*
- *The domain of H is a hyper-cube*

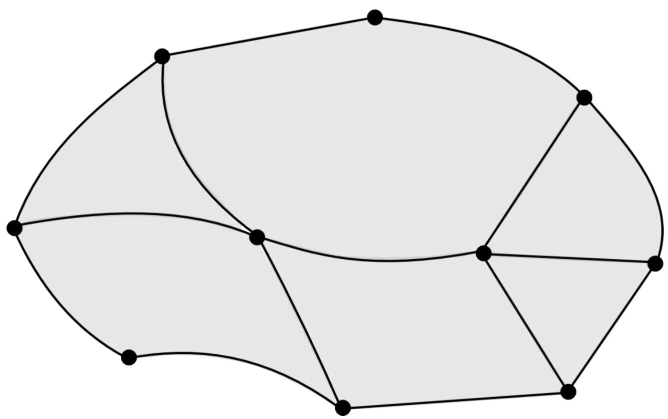


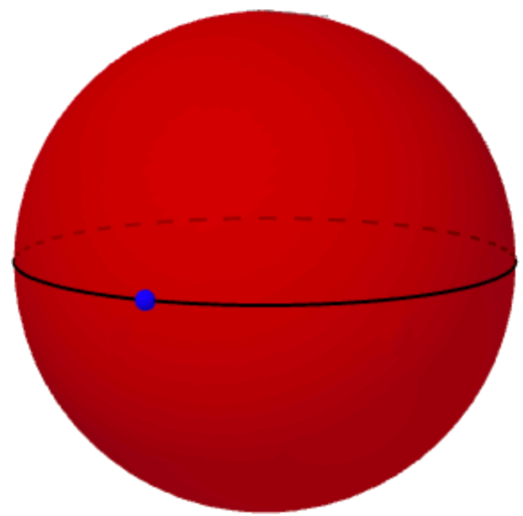
Cell Complexes

Intuitively:

Similarly to simplicial complexes and regular grids,

A ***cell complex*** Γ is a collection of cells ***"suitably glued together"***





Where a ***k-cell*** is a topological space homeomorphic to the ***k-dimensional open disk i(D^k)***

Bibliography

Some References:

- ***Simplicial Complexes:***
  - J. R. Munkres. ***Elements of algebraic topology***. CRC Press, 1984.

Simplicial Homology

Simplicial Homology

Given a topological space *X*, the ***homology of X*** is a ***topological invariant***

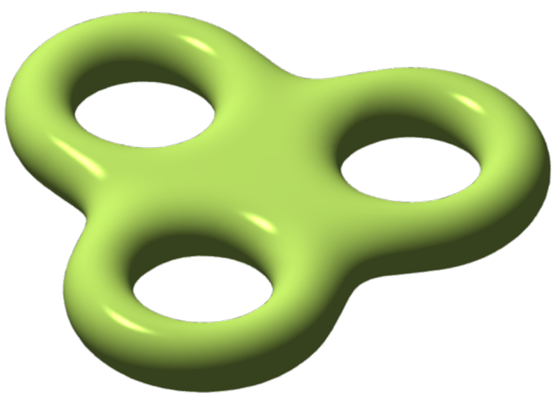
intuition

detecting the "holes" of X

capturing the independent non-bounding cycles of X

measuring how far the chain complex associated with X is from being exact

formalism



$$\longmapsto H_k(X;\mathbb{Z}) \cong \begin{cases} \mathbb{Z} & \text{for } k = 0 \\ \mathbb{Z}^6 & \text{for } k = 1 \\ \mathbb{Z} & \text{for } k = 2 \\ 0 & \text{otherwise} \end{cases}$$

Simplicial Homology

Topological Space

Algebraic Structure

Simplicial Complex K

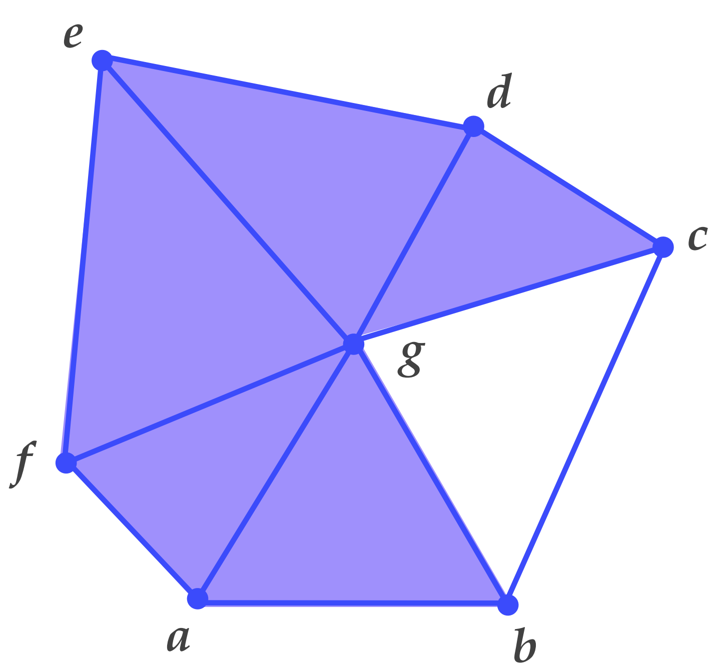
Chain Complex $C_*(K)$

Simplicial Homology $H_*(K)$

Simplicial Homology

Given a simplicial complex K,

- a ***k-chain*** is a formal sum *(with $\mathbb{Z}_2$ coefficients)* of k-simplices of K



Examples:

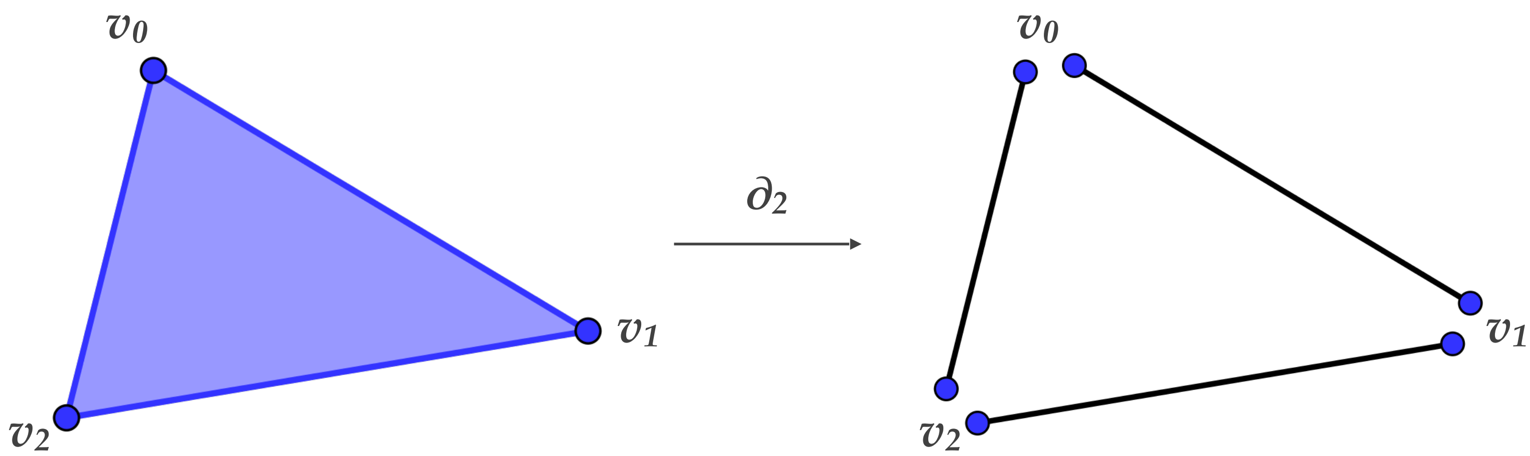
- $a + b + e$ is a 0-chain
- $fg + dg + de + eg$ is a 1-chain
- $abg + afg$ is a 2-chain

Simplicial Homology

The *chain complex* ***$C_*(K)$*** *associated with K* consists of:

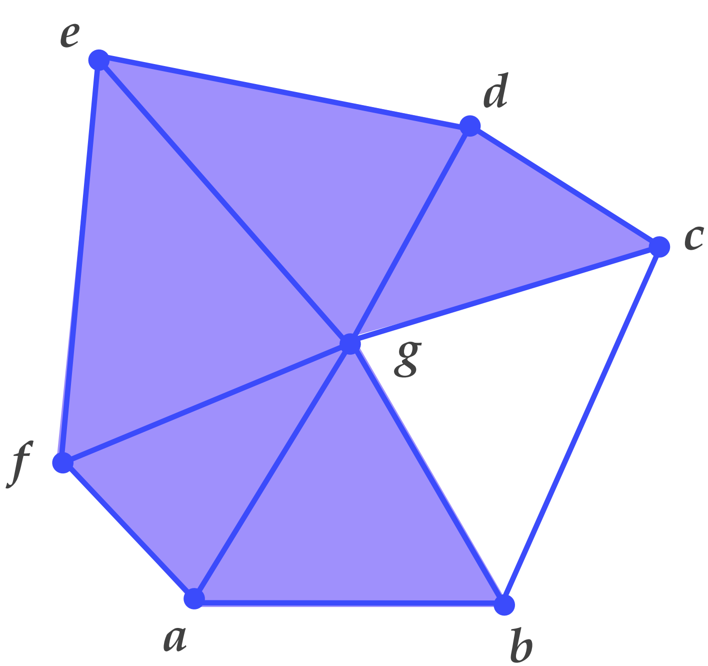
- a collection $\{C_k(K)\}_{k\in\mathbb{Z}}$ of vector spaces where **$C_k(K)$** is the *group of the k-chains* of K
- a collection $\{\partial_k\}_{k\in\mathbb{Z}}$ of linear maps where the *boundary map* **$\partial_k : C_k(K) \longrightarrow C_{k-1}(K)$** is defined by

$$\partial_k(v_0 \cdots v_k) := \sum_{i=0}^{k} v_0 \cdots \hat{v}_i \cdots v_k$$



Simplicial Homology

Examples:



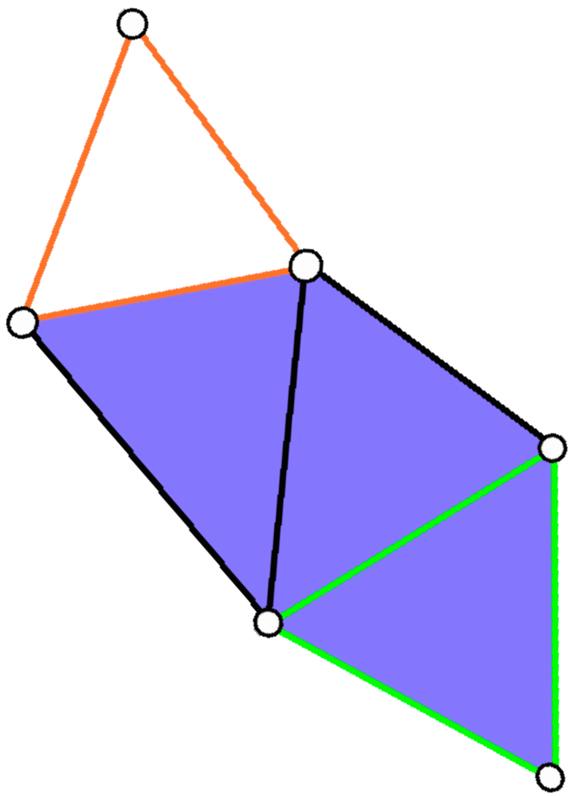
- $\partial_1(\,ab\,) = a + b$
- $\partial_1(\,ab + bc\,) = a + 2b + c = a + c$
- $\partial_2(\,afg + efg\,) = af + ag + 2fg + ef + eg =$
  $= af + ag + ef + eg$
- $\partial_1(\,af + ag + ef + eg\,) =$
  $= 2a + 2f + 2g + 2e = 0$

Simplicial Homology

Properties:

- For ***$k < 0$*** or ***$k > dim(K)$***, $C_k(K)$ is the ***null group***
- For ***$k \leq 0$*** or ***$k > dim(K)$***, ∂_k is the ***null map***
- For any $k \in \mathbb{Z}$, $\boldsymbol{\partial_k \circ \partial_{k+1} = 0}$
- For any $k \in \mathbb{Z}$, $\boldsymbol{Im(\partial_{k+1}) \subseteq Ker(\partial_k)}$

Simplicial Homology



Definition:

A k-chain c is called:

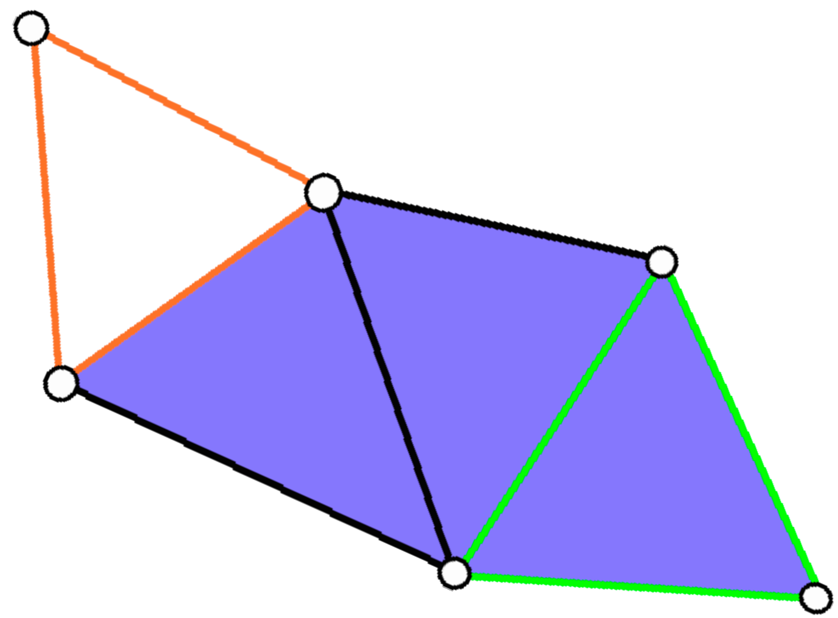
- ***k-cycle*** if $c \in \mathrm{Ker}(\partial_k)$
- ***k-boundary*** if $c \in \mathrm{Im}(\partial_{k+1})$

Each k-boundary is a k-cycle

Simplicial Homology

Given a simplicial complex K, the ***k-homology group*** ***$H_k(K)$*** of K is defined as

$$H_k(K) := Z_k(K)/B_k(K)$$

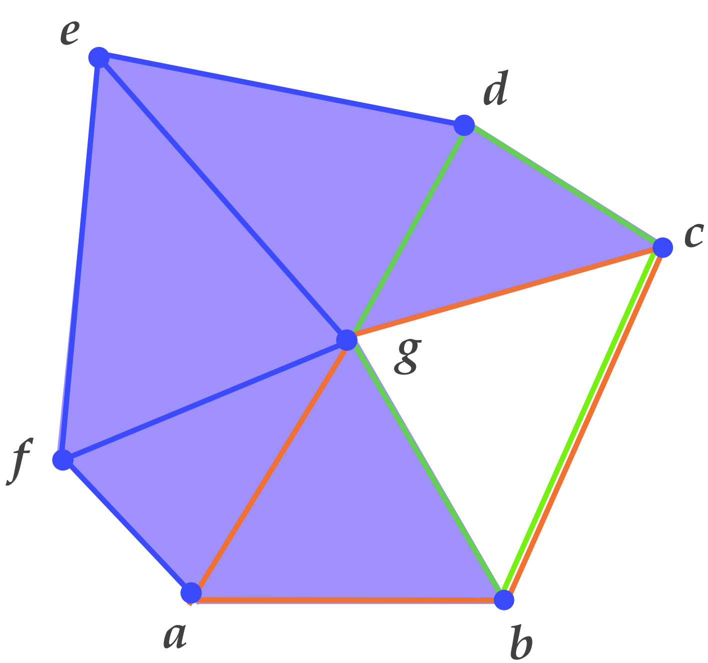


where:

- ***$Z_k(K)$*** is the ***group of k-cycles*** of K
- ***$B_k(K)$*** is the ***group of k-boundaries*** of K

Simplicial Homology

$H_k(K)$ partitions the k-cycles into equivalence classes called ***homology classes***



Definition:

Two k-cycles are said ***homologous*** if they belong to the same homology class or, equivalently, ***if their difference is a k-boundary***

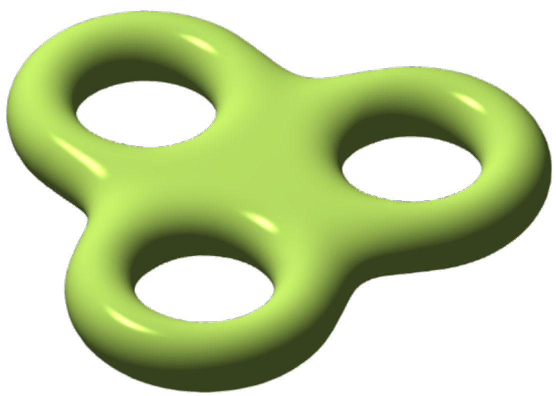
ab+ag+bc+cg is homologous to bc+bg+cd+dg

Simplicial Homology

Theorem:

Each homology group can be expressed as

$$H_k(K) \cong (\mathbb{Z}_2)^{\beta_k}$$



$$H_k(K) \cong \begin{cases} \mathbb{Z}_2 & \text{for } k = 0 \\ (\mathbb{Z}_2)^6 & \text{for } k = 1 \\ \mathbb{Z}_2 & \text{for } k = 2 \end{cases}$$

β_k is called the ***k^{th} Betti number*** of K

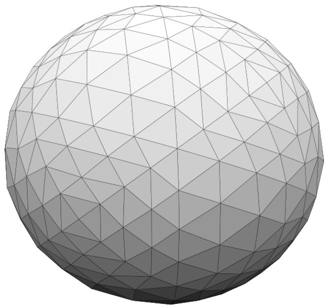
Simplicial Homology

Examples:

- ***point P***

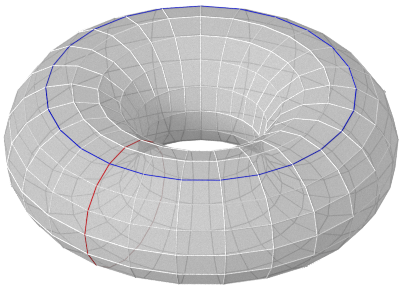
$$\beta_k(P) = \begin{cases} 1 & \text{for } k = 0 \\ 0 & \text{for } k > 0 \end{cases}$$

- ***n-dimensional sphere S^n***



$$\beta_k(S^n) = \begin{cases} 1 & \text{for } k = 0 \\ 0 & \text{for } 0 < k < n \\ 1 & \text{for } k = n \\ 0 & \text{for } k > n \end{cases}$$

- ***torus T***



$$\beta_k(T) = \begin{cases} 1 & \text{for } k = 0 \\ 2 & \text{for } k = 1 \\ 1 & \text{for } k = 2 \\ 0 & \text{for } k > 2 \end{cases}$$

Simplicial Homology

Homology groups can be defined ***in a more general way*** by choosing coefficients in $\mathbb{Z}$

Theorem:

Each homology group can be expressed as

$$H_k(K;\mathbb{Z}) \cong \mathbb{Z}^{\beta_k}\langle c_1,\ldots,c_{\beta_k}\rangle \oplus \mathbb{Z}_{\lambda_1}\langle c'_1\rangle \oplus \cdots \oplus \mathbb{Z}_{\lambda_{p_k}}\langle c'_{p_k}\rangle$$

with $\lambda_{i+1} \mid \lambda_i$

We call:

- β_k, the ***k^{th} Betti number*** of K
- $\lambda_1,\ldots,\lambda_{p_k}$, the ***torsion coefficients*** of K
- $c_1,\ldots,c_{\beta_k},c'_1,\ldots,c'_{p_k}$, the ***homology generators*** of K

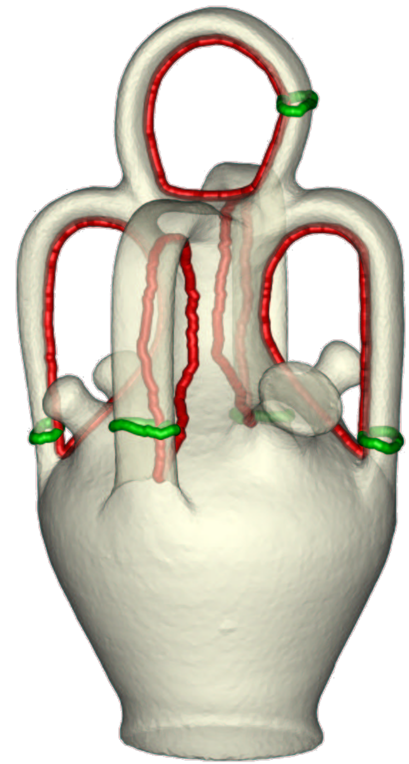


Image from [Dey et al. 2008]

Simplicial Homology

Working with coefficients in $\mathbb{Z}$:

*Up to isomorphism, the **Betti numbers** and the **torsion coefficients** of K*
***completely characterize** the **homology groups** of K*

Working with coefficients in a field $\mathbb{F}$:

*Up to isomorphism, the **Betti numbers** of K*
***completely characterize** the **homology groups** of K*

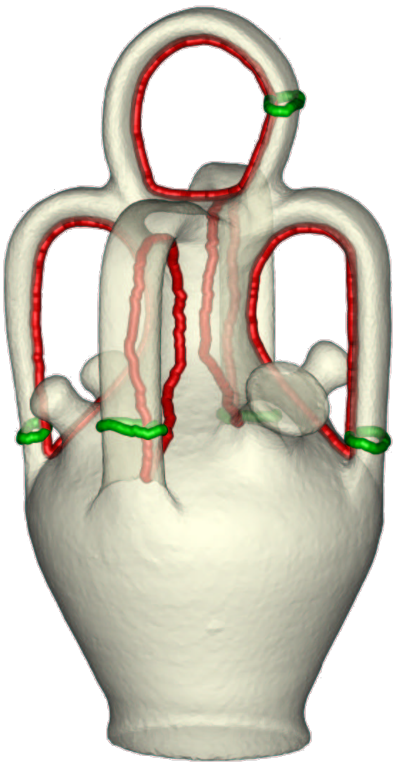
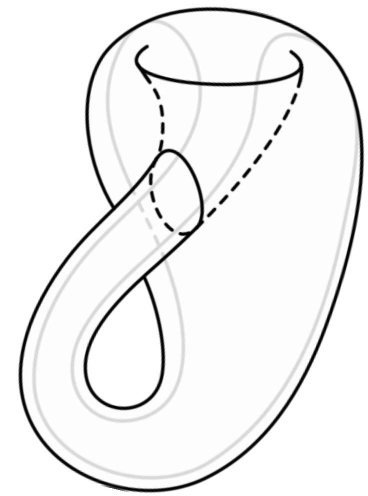


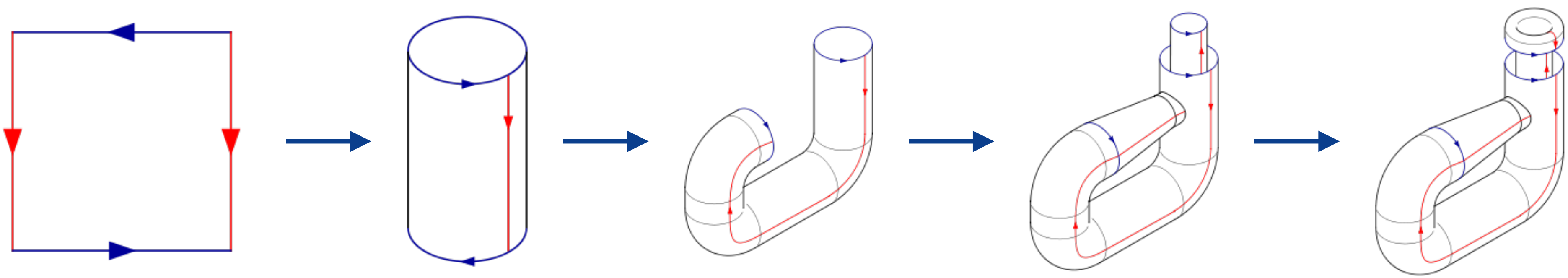
Image from [Dey et al. 2008]

Simplicial Homology

Example:

The ***Klein bottle K*** *is a non-orientable 2-dimensional manifold embeddable in* $\mathbb{R}^4$ *which can be built from a unit square by the following construction*





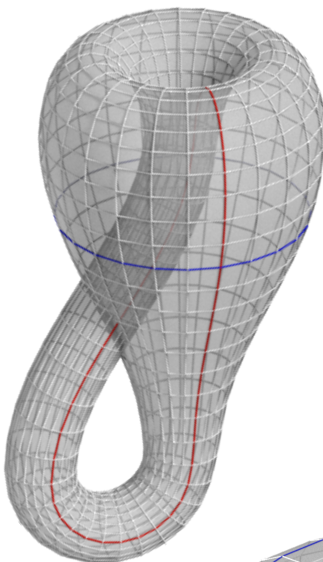
Simplicial Homology

Example:

By considering $\mathbb{Z}$ as coefficient group,

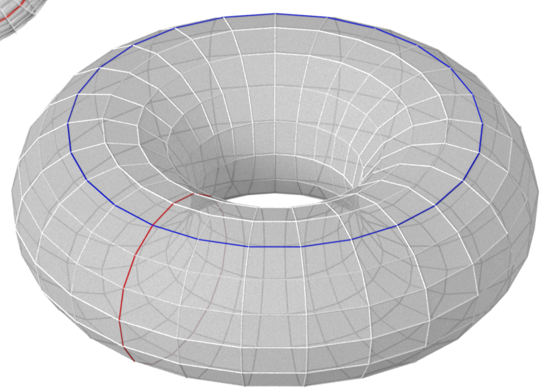
K has the following homology groups

$$H_k(K;\mathbb{Z}) \cong \begin{cases} \mathbb{Z} & \text{for } k=0 \\ \mathbb{Z}\oplus\mathbb{Z}_2 & \text{for } k=1 \\ 0 & \text{for } k\geq 2 \end{cases}$$



So, it can be distinguished from a torus T

$$H_k(T;\mathbb{Z}) \cong \begin{cases} \mathbb{Z} & \text{for } k=0 \\ \mathbb{Z}^2 & \text{for } k=1 \\ \mathbb{Z} & \text{for } k=2 \\ 0 & \text{for } k>2 \end{cases}$$

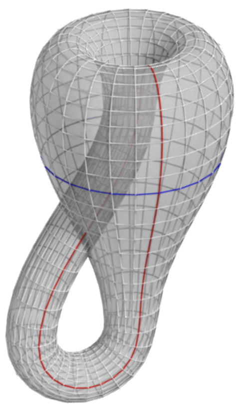


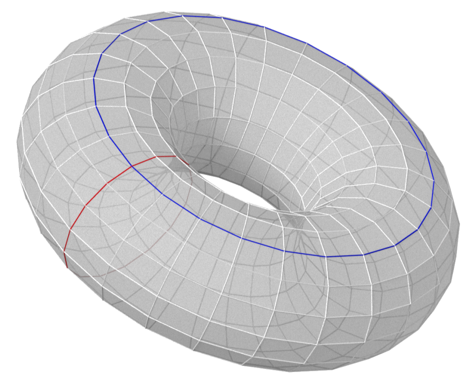
Simplicial Homology

Example:

By considering $\mathbb{Z}_2$ as coefficient group,

the Klein bottle K and the torus T have isomorphic homology groups





$$H_k(K;\mathbb{Z}_2) \cong \left\{ \begin{array}{ll} \mathbb{Z}_2 & \text{for } k=0 \\ \mathbb{Z}_2 \oplus \mathbb{Z}_2 & \text{for } k=1 \\ \mathbb{Z}_2 & \text{for } k=2 \\ 0 & \text{for } k>2 \end{array} \right\} \cong H_k(T;\mathbb{Z}_2)$$

Bibliography

Some References:

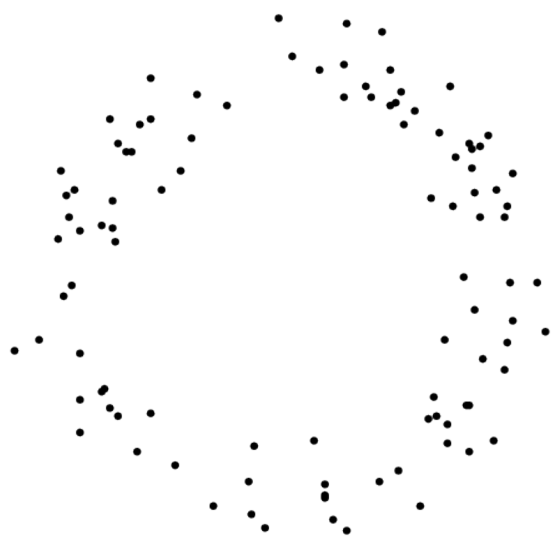
- ***Simplicial Homology:***
  - J. R. Munkres. ***Elements of algebraic topology***. CRC Press, 1984.

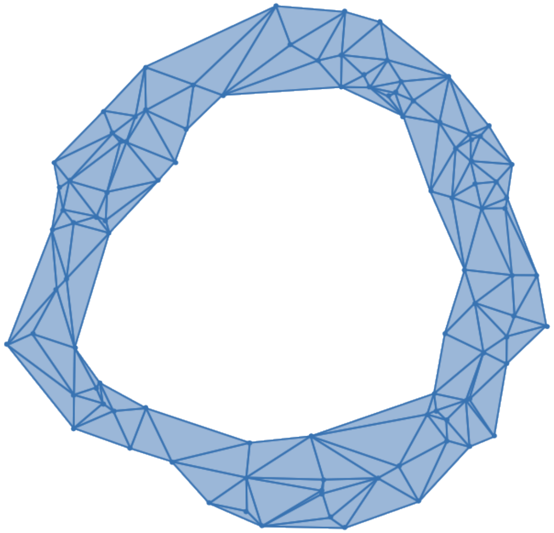
From Data to Complexes

From Data to Complexes

Let us consider a dataset represented by a ***finite point cloud V in*** $\mathbb{R}^n$

*Studying the shape of V just by considering the space consisting of its **points does not provide any relevant topological information***





*The **"real" shape** of the dataset can be captured by properly constructing a **complex connecting together close points through simplices***

From Data to Complexes

Standard Constructions:

A number of possible choices have been introduced in the literature:

- ***Delaunay* triangulations**
  - ***Voronoi*** *diagrams*
- ***Čech* complexes**
- ***Vietoris-Rips* complexes**
- ***Alpha-shapes***
- ***Witness* complexes**

Most of the above constructions are based on the notion of ***Nerve* complex**

From Data to Complexes

A First Classification: Given a finite point cloud V in $\mathbb{R}^n$,

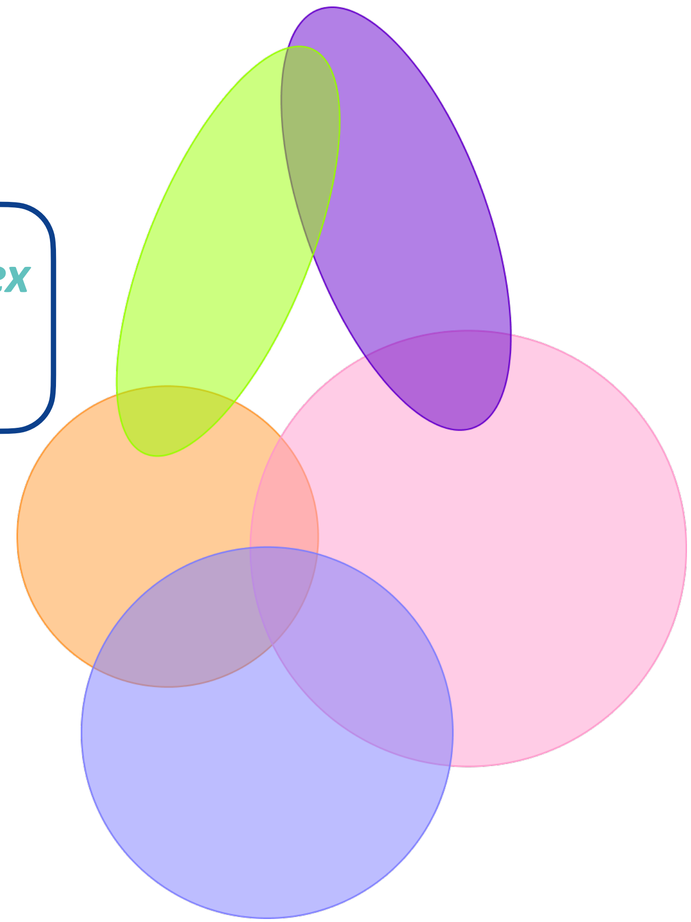
	Output Complex	Dimension	Dependence on a Parameter
Delaunay triangulation	*Geometric*	*n*	✗
Čech complex	*Abstract*	*Arbitrary (up to |V| - 1)*	✓
Vietoris-Rips complex	*Abstract*	*Arbitrary (up to |V| - 1)*	✓
Alpha-shapes	*Geometric*	*n*	✓
Witness complexes	*Abstract*	*Arbitrary (up to |V| - 1)*	✓

Nerve Complexes

Definition:

Given a finite collection S of sets in $\mathbb{R}^n$,

The ***nerve Nrv(S)*** of S is the ***abstract simplicial complex*** generated by the ***non-empty common intersections***



Formally,

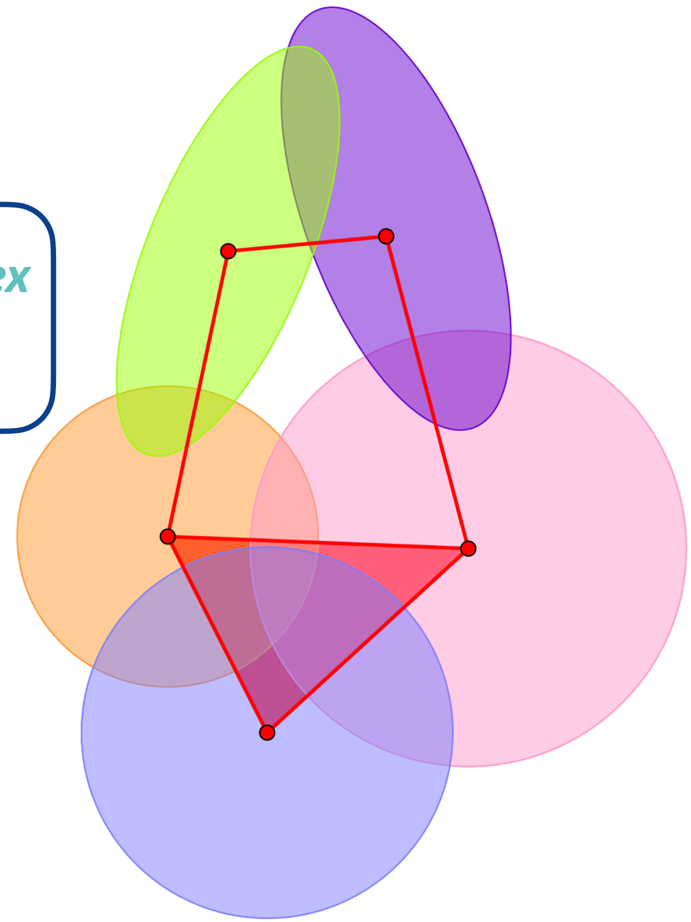
$$Nrv(S) := \{\sigma \subseteq S \mid \bigcap_{s \in \sigma} s \neq \emptyset\}$$

Nerve Complexes

Definition:

Given a finite collection S of sets in $\mathbb{R}^n$,

The ***nerve Nrv(S)*** of S is the ***abstract simplicial complex*** generated by the ***non-empty common intersections***



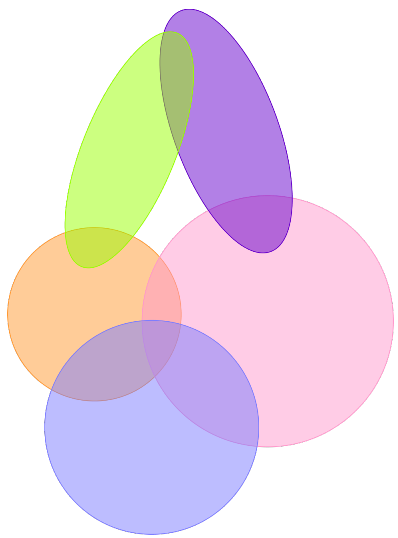
Formally,

$$Nrv(S) := \{\sigma \subseteq S \mid \bigcap_{s \in \sigma} s \neq \emptyset\}$$

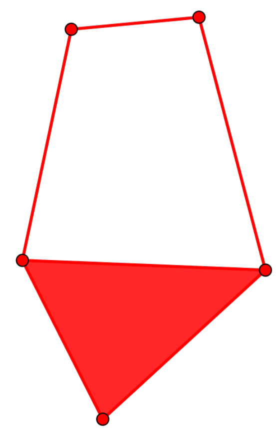
Nerve Complexes

Nerve Theorem:

If S is a finite collection of ***convex*** *sets in* $\mathbb{R}^n$*, then the* ***nerve of S*** *and the* ***union of the sets in S*** *are* ***homotopy equivalent*** *(and so they have the same homology)*



$\simeq$



Nerve Complexes

Nerve Theorem can be ***generalized*** by replacing the ***convexity*** of sets in S with the request that all non-empty common intersections are ***contractible*** *(i.e. that can be continuously shrunk to a point)*

Original Nerve Theorem:

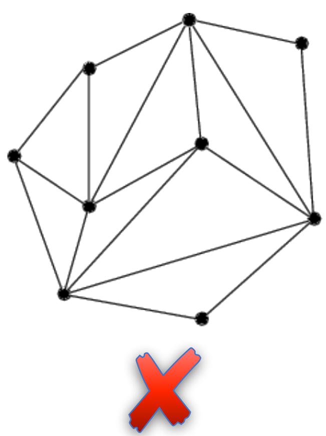
*If S is an open cover of a (para)**compact** space X such that every non-empty intersection of finitely many sets in S is **contractible**, then **X** is **homotopy equivalent** to the nerve **Nrv(S)***

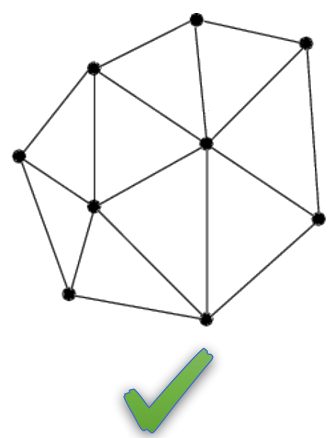
Delaunay Triangulations

Given a finite point cloud V in $\mathbb{R}^n$,

The ***Delaunay triangulation*** of V is a classic notion in Computational Geometry:

- *Producing a **“nice” triangulation** of V*
  - *free of long and skinny triangles*
- *Named after **Boris Delaunay** for his work on this topic from 1934*
- *Originally defined for sets of points in $\mathbb{R}^2$ but generalizable to arbitrary dimensions*





Images from [De Floriani 2003]

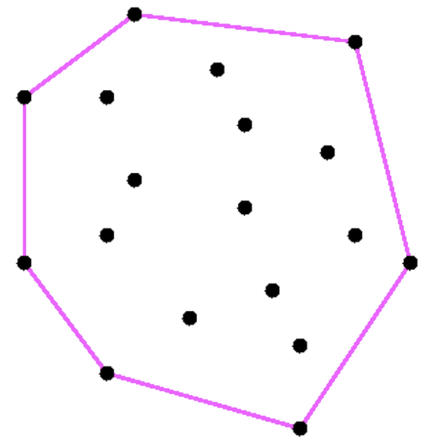
Delaunay Triangulations

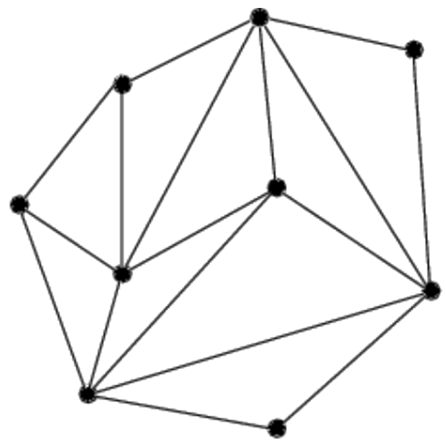
Definitions:

Given a finite point cloud V in $\mathbb{R}^2$,

- The ***convex hull*** of V is the ***smallest convex*** subset ***CH(V)*** of $\mathbb{R}^2$ containing all the points of V

- A ***triangulation*** of V is A ***2-dimensional simplicial complex K*** such that:
  - The domain of K is CH(V)
  - The 0-simplices of K are the points in V





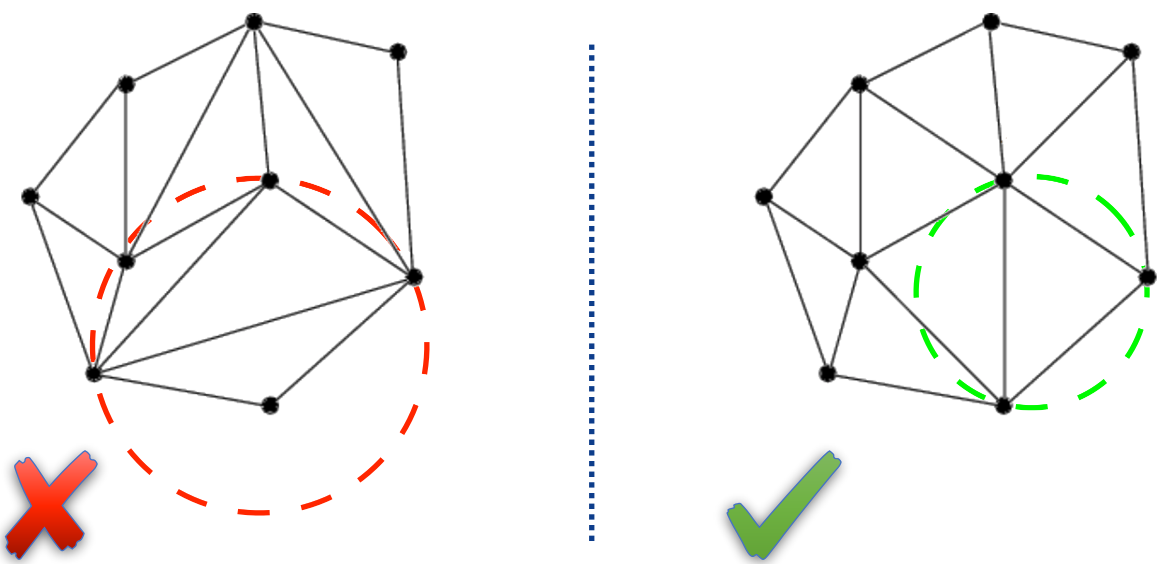
Images from [De Floriani 2003]

Delaunay Triangulations

Definition:

A ***Delaunay triangulation*** is a triangulation ***Del(V)*** of V such that:

the ***circumcircle of any triangle*** does ***not contain any point*** of V in its interior



Images from [De Floriani 2003]

Delaunay Triangulations

Definition:

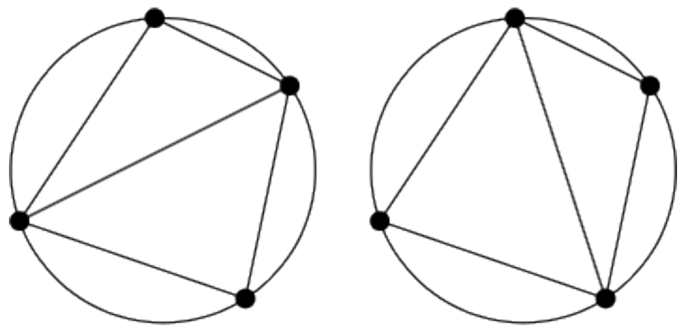
A finite set of points V in $\mathbb{R}^n$ is ***in general position*** if no n + 2 of the points lie on a common (n − 1)-sphere

*E.g. , **for n = 2**,*

V in general position ⟺ *if and only if* ⟺ *No four or more points are co-circular*

Theorem:

*If V is in general position, then Del(V) is **unique***



Images from [De Floriani 2003]

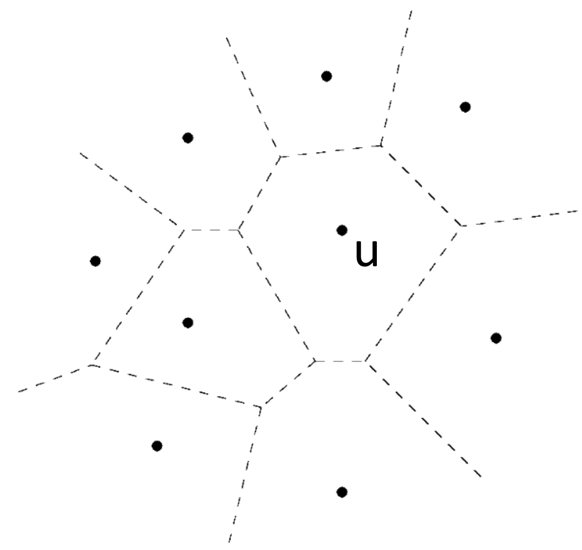
Delaunay Triangulations

Definitions:

The ***Voronoi region*** of u in V is the set of points of $\mathbb{R}^2$ for which u is the closest

$$R_V(u) := \{x \in \mathbb{R}^2 \,|\, \forall v \in V, d(x,u) \leq d(x,v)\}$$

- *Any Voronoi region is a **convex** closed subset of* $\mathbb{R}^2$
- *A Voronoi region is **not necessarily bounded***



The ***Voronoi diagram*** is the collection ***Vor(V)*** of the Voronoi regions of the points of V

Images from [De Floriani 2003]

Delaunay Triangulations

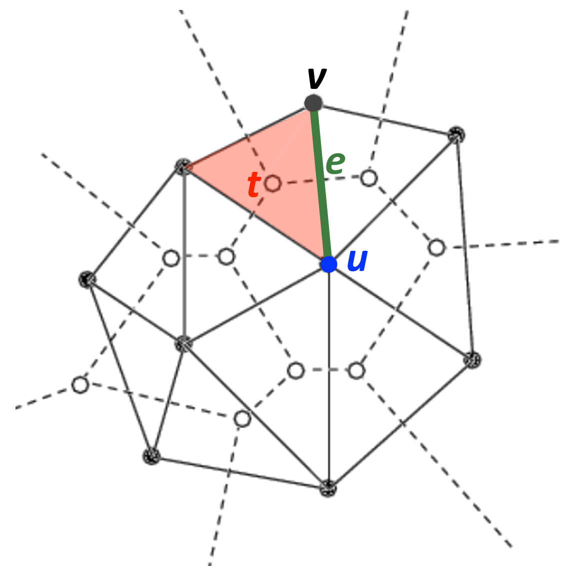
Duality Property:

If V is in general position, then

*the **Delaunay triangulation coincides** with the **nerve of the Voronoi diagram***

$$Del(V) = \{\sigma \subseteq V \mid \bigcap_{u \in \sigma} R_V(u) \neq \emptyset\}$$

- *Each **point u** of V corresponds to a Voronoi region $R_V(u)$*
- *Each **triangle t** of Del(V) correspond to a vertex in Vor(V)*
- *Each **edge e=(u,v)** in Del(V) corresponds to an edge shared by the two Voronoi regions $R_V(u)$ and $R_V(v)$*



Images from [De Floriani 2003]

Delaunay Triangulations

Algorithms:

- ***Two-step algorithms***:
  - *Computation of an arbitrary triangulation K'*
  - *Optimization of K' to produce a Delaunay triangulation*
- ***Incremental algorithms*** *[Guibas, Stolfi 1983; Watson 1981]:*
  - *Modification of an existing Delaunay triangulation while adding a new vertex at a time*
- ***Divide-and-conquer algorithms*** *[Shamos 1978; Lee, Schacter 1980]:*
  - *Recursive partition of the point set into two halves*
  - *Merging of the computed partial solutions*
- ***Sweep-line algorithms*** *[Fortune 1989]:*
  - *Step-wise construction of a Delaunay triangulation while moving a sweep-line in the plane*

Delaunay Triangulations

Watson's Algorithm:

A Delaunay triangulation is computed by **incrementally adding a single point** to an existing Delaunay triangulation

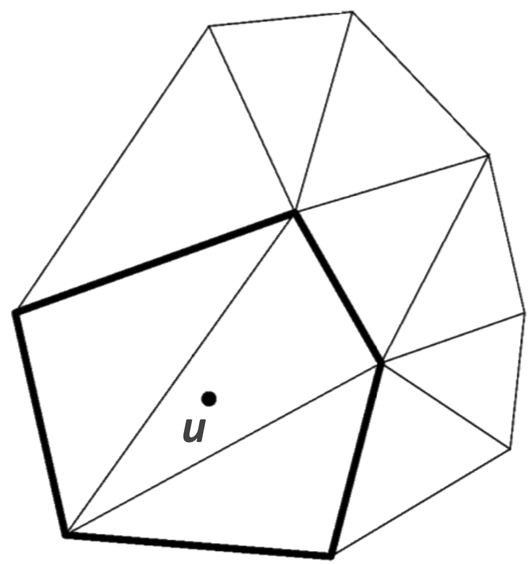
Let V_i be a subset of V and let u be a point in $V \setminus V_i$,

Input:

Del(V_i), a Delaunay triangulation of V_i

Output:

Del(V_{i+1}), a Delaunay triangulation of $V_{i+1} := V_i \cup \{u\}$



Images from [De Floriani 2003]

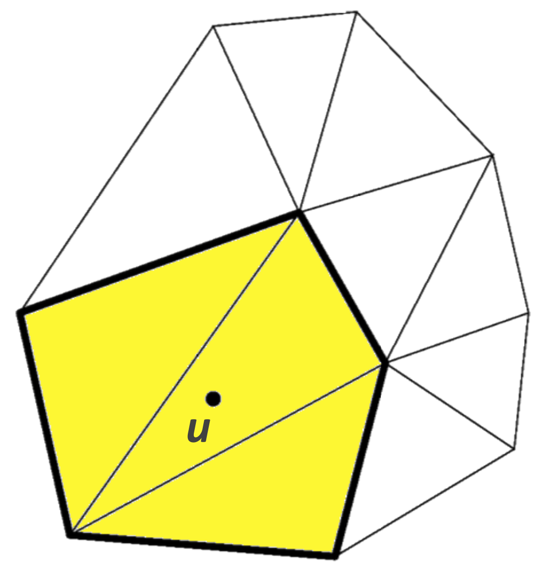
Delaunay Triangulations

Watson's Algorithm:

Given a Delaunay triangulation Del(V_i) of V_i and a point u in $V \setminus V_i$,

- *The **influence region R_u** of a point u is the region in the plane formed by the union of the triangles in Del(V_i) whose circumcircle contains u in its interior*

- *The **influence polygon P_u** of u is the polygon formed by the edges of the triangles of Del(V_i) which bound R_u*



Images from [De Floriani 2003]

Delaunay Triangulations

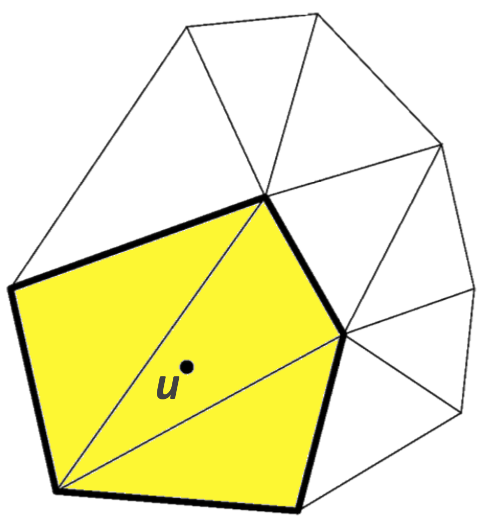
Watson's Algorithm:

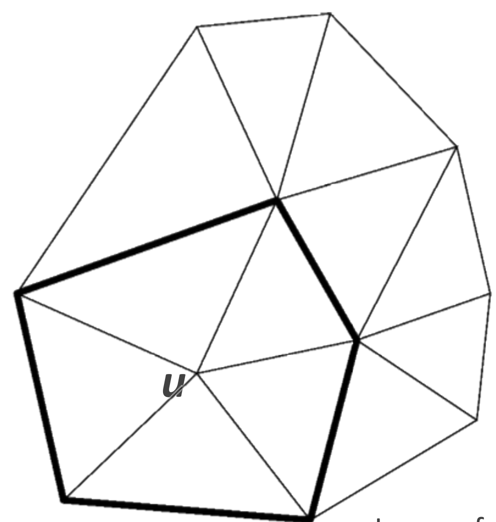
- ***Step 1:***

  Deletion of the triangles of Del(V_i) forming the ***influence region R_u***

- ***Step 2:***

  Re-triangulation of R_u by joining u to the vertices of the influence polygon P_u





Images from [De Floriani 2003]

Delaunay Triangulations

Watson's Algorithm:

Let $N_i = |V_i|$

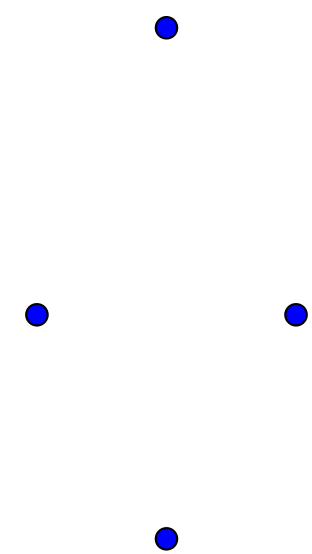
- *Detection of a triangle of Del(V_i) containing the new point u: O(N_i) in the worst case*
- *Detection of the triangles forming the region of influence through a breadth-first search: O($|R_u|$)*
- *Re-triangulation of P_u is in O($|P_u|$)*

- ***Inserting a point** u in a triangulation with N_i vertices: **O(N_i)** in the worst case*
- ***Inserting all points** of V: **O(N^2)** in the worst case, where N =|V|*

Čech Complexes

Definition:

Given a finite set of points V in $\mathbb{R}^n$, let us consider:

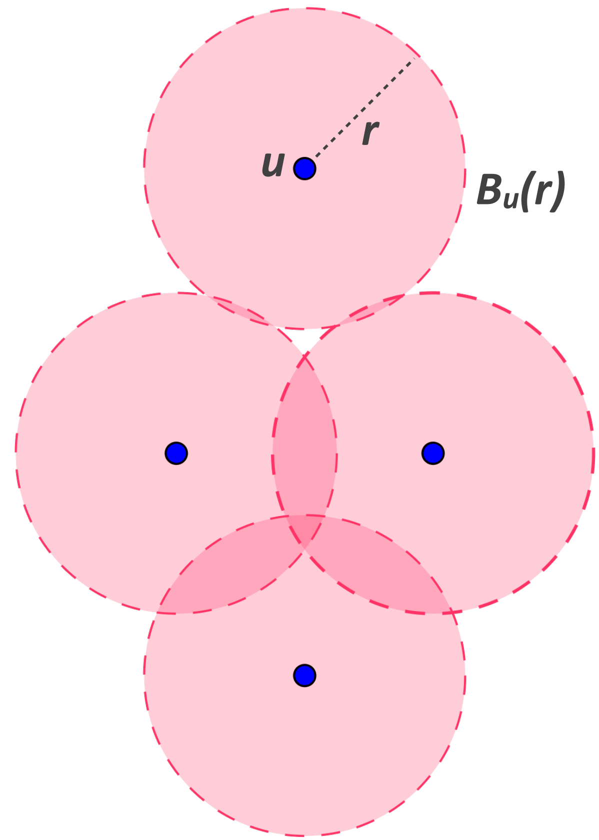


Čech Complexes

Definition:

Given a finite set of points V in $\mathbb{R}^n$, let us consider:

- ***$B_u(r)$**, the **closed ball** with **center u** $\in$ V and **radius r***
- ***S**, the collection of these balls*

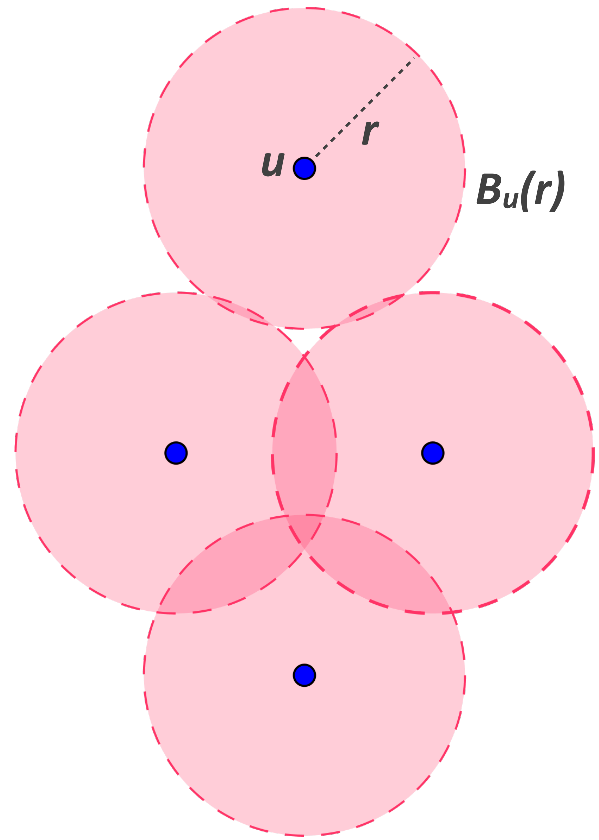


Čech Complexes

Definition:

Given a finite set of points V in $\mathbb{R}^n$, let us consider:

- ***$B_u(r)$**, the **closed ball** with **center $u \in V$** and **radius r***
- ***S**, the collection of these balls*



The ***Čech complex Čech(r)*** of V of radius r is the ***nerve of S***

$$\check{C}ech(r) := \{\sigma \subseteq V \mid \bigcap_{u\in\sigma} B_u(r) \neq \emptyset\}$$

Čech Complexes

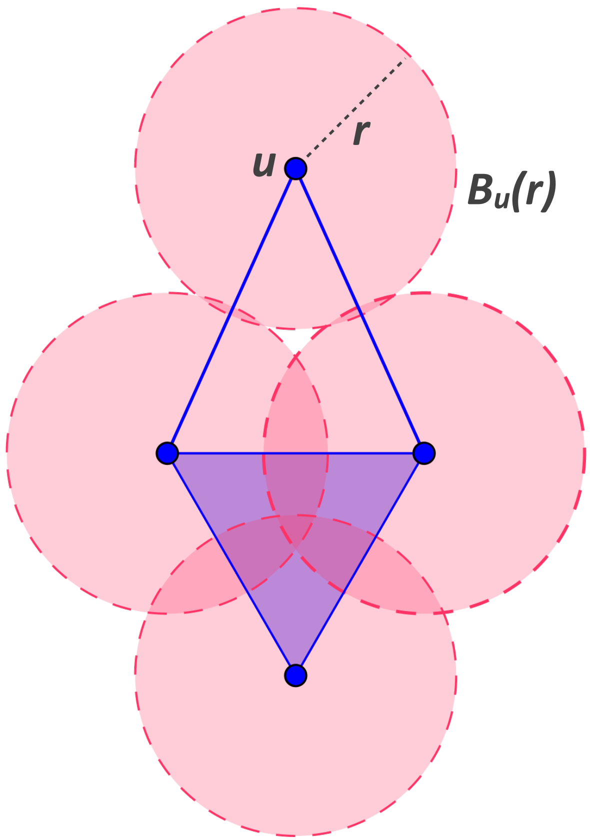
Definition:

Given a finite set of points V in $\mathbb{R}^n$, let us consider:

- ***$B_u(r)$***, *the **closed ball** with **center u** $\in$ V and **radius r***
- ***S***, *the collection of these balls*

The ***Čech complex Čech(r)*** of V of radius r is the ***nerve of S***

$$\check{C}ech(r) := \{\sigma \subseteq V \mid \bigcap_{u \in \sigma} B_u(r) \neq \emptyset\}$$

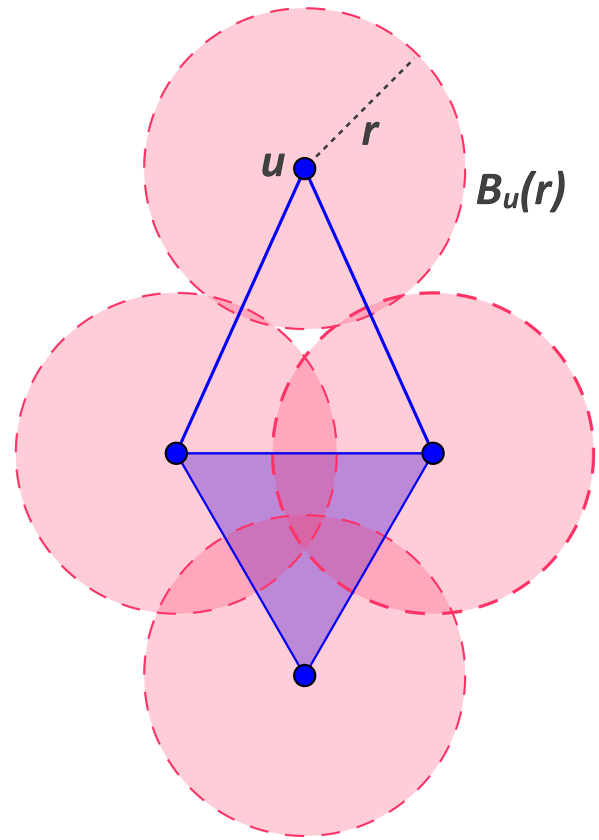


Čech Complexes

Definition:

Given a finite set of points V in $\mathbb{R}^n$, let us consider:

- ***$B_u(r)$***, *the* ***closed ball*** *with* ***center u*** *$\in$ V and* ***radius r***
- ***S***, *the collection of these balls*



The ***Čech complex Čech(r)*** of V of radius r is the ***nerve of S***

$$\check{C}ech(r) := \{\sigma \subseteq V \mid \bigcap_{u \in \sigma} B_u(r) \neq \emptyset\}$$

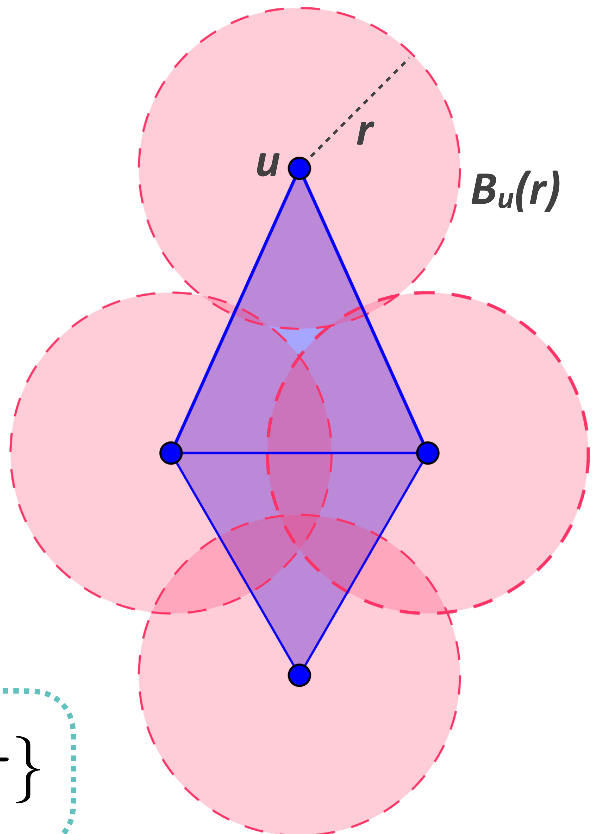
In practice, **infeasible construction**

Vietoris-Rips Complexes

Definition:

Given a finite set of points V in $\mathbb{R}^n$,

The ***Vietoris-Rips complex VR(r)*** of V and r is the ***abstract simplicial complex*** consisting of all ***subsets of diameter at most 2r***



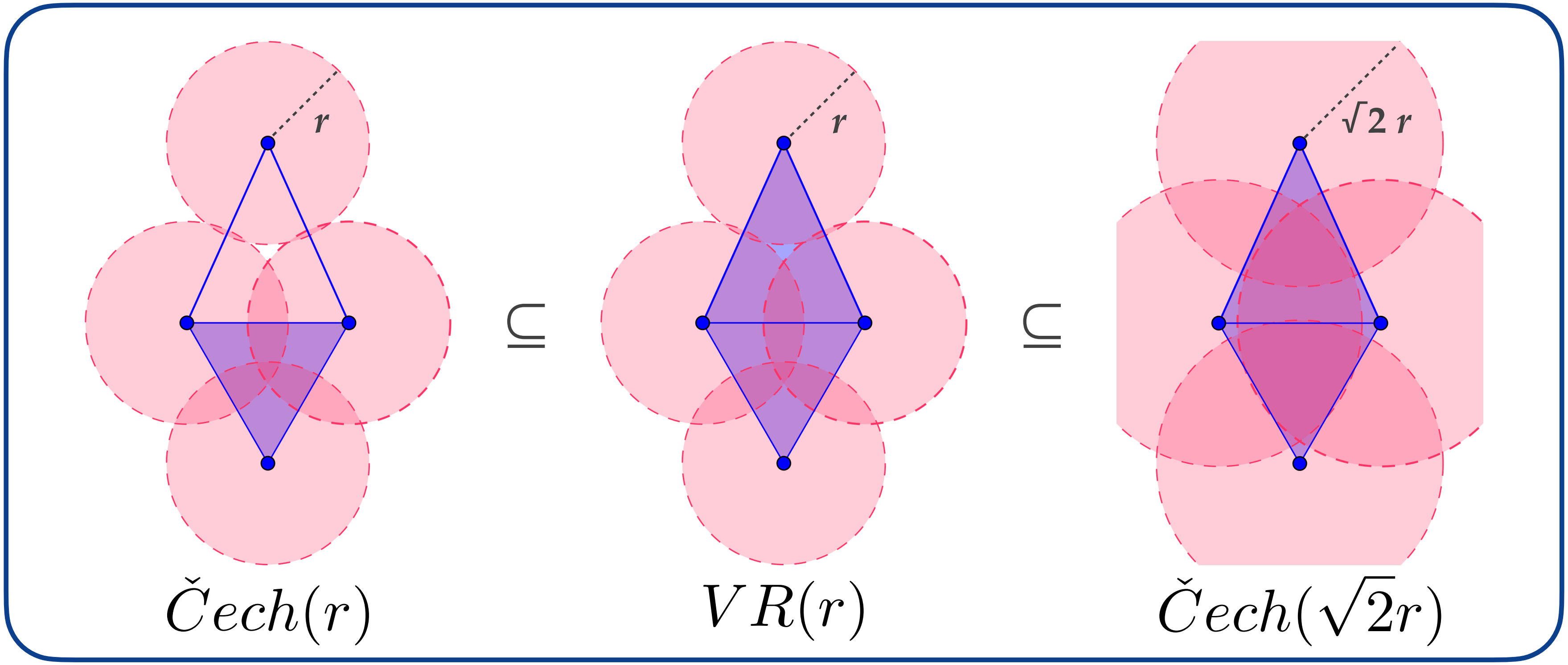
Formally,

$$VR(r) := \{\sigma \subseteq V \mid d(u,v) \leq 2r, \forall\, u,v \in \sigma\}$$

Vietoris-Rips Complexes

Properties:

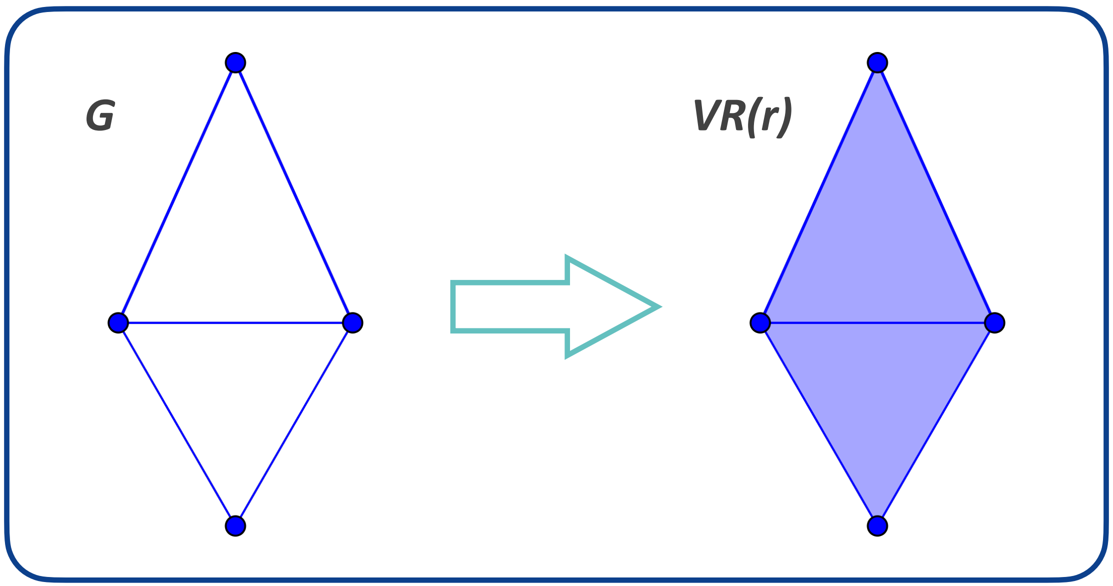
- $\check{C}ech(r) \subseteq VR(r) \subseteq \check{C}ech(\sqrt{2}r)$



Vietoris-Rips Complexes

Properties:

- $\check{C}ech(r) \subseteq VR(r) \subseteq \check{C}ech(\sqrt{2}r)$
- ***VR(r)*** is completely determined by its 1-skeleton
  - I.e. the graph ***G*** of its vertices and its edges



Vietoris-Rips Complexes

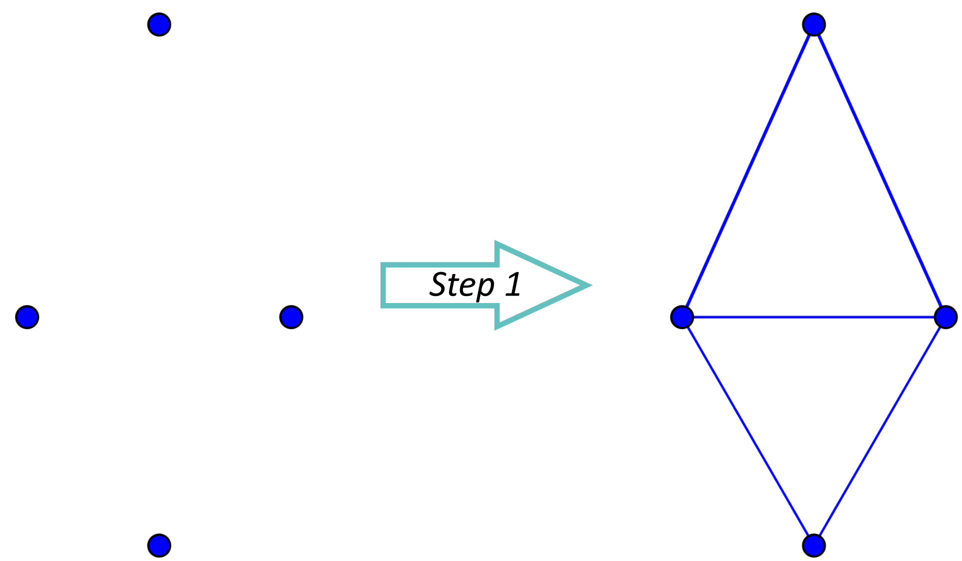
Algorithms:

<u>*Input:*</u> A finite set of points V in $\mathbb{R}^n$ and a real positive number r

<u>*Output:*</u> The Vietoris-Rips complex VR(r)

A ***two-step*** approach is typically adopted:

- ***Step 1 - Skeleton Computation:***
  - *Exact ($O(|V|^2)$ time complexity)*
  - *Approximate*
  - *Randomized*
  - *Landmarking*
- ***Step 2 - Vietoris-Rips Expansion:***
  - *Inductive*
  - *Incremental*
  - *Maximal*



Vietoris-Rips Complexes

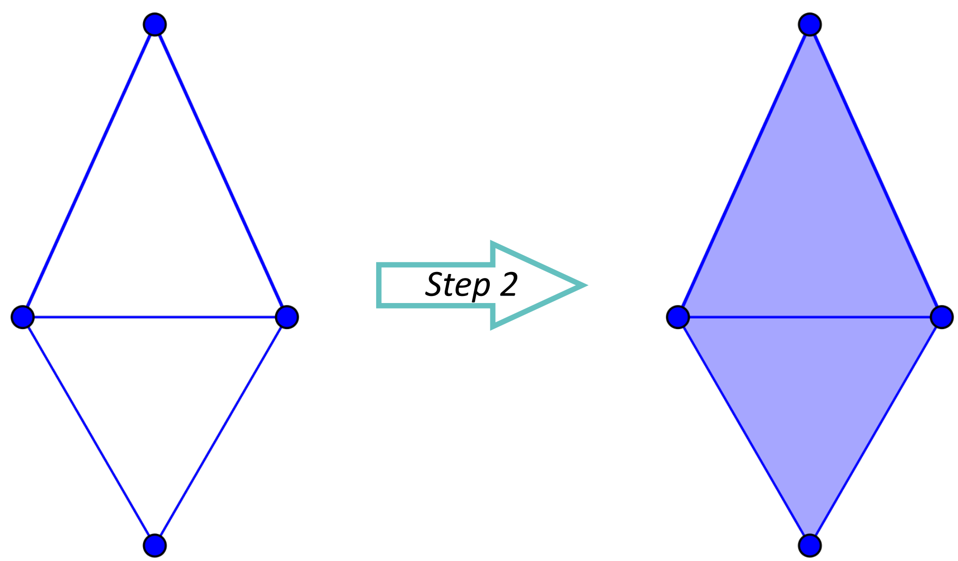
Algorithms:

Input: A finite set of points V in $\mathbb{R}^n$ and a real positive number r

Output: The Vietoris-Rips complex VR(r)

A ***two-step*** approach is typically adopted:

- ***Step 1 - Skeleton Computation:***
  - *Exact ($O(|V|^2)$ time complexity)*
  - *Approximate*
  - *Randomized*
  - *Landmarking*
- ***Step 2 - Vietoris-Rips Expansion:***
  - *Inductive*
  - *Incremental*
  - *Maximal*



Vietoris-Rips Complexes

Inductive VR expansion:

Input: The 1-skeleton G = (V, E) of VR(r)

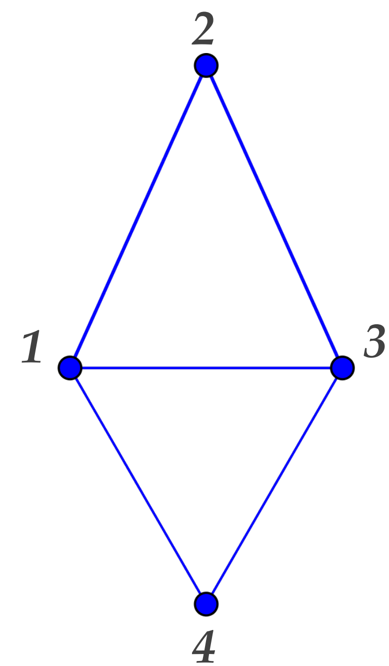
Output: The k-skeleton K of the Vietoris-Rips complex VR(r)

INDUCTIVE-VR(G, k)

K = *V* ∪ *E*
for *i* = 1 ***to*** *k*
  foreach *i-simplex* $\sigma \in K$
    $N = \cap_{u \in \sigma}$ *LOWER-NBRS(G, u)*
      foreach $v \in N$
        $K = K \cup \{ \sigma \cup \{v\}\}$
  return *K*

LOWER-NBRS(G, u)

  return $\{v \in V \mid v < u , (u, v) \in E\}$



Vietoris-Rips Complexes

Inductive VR expansion:

Input: The 1-skeleton G = (V, E) of VR(r)

Output: The k-skeleton K of the Vietoris-Rips complex VR(r)

INDUCTIVE-VR(G, k)

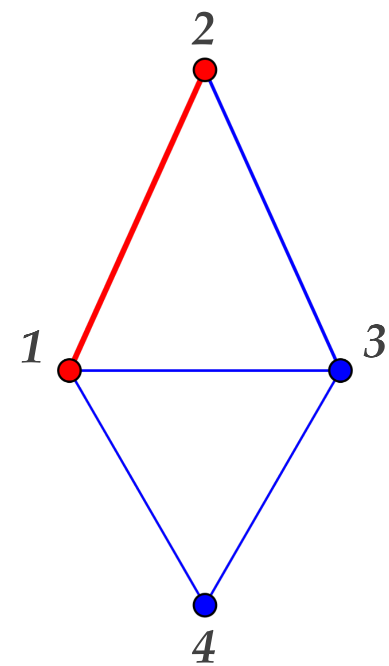
```
K = V ∪ E
for i = 1 to k
    foreach i-simplex σ ∈ K
        N = ∩_{u∈σ} LOWER-NBRS(G, u)
        foreach v ∈ N
            K = K ∪ { σ ∪ {v}}
    return K
```

LOWER-NBRS(G, u)

```
    return {v ∈ V | v < u , (u, v) ∈ E}
```

$\sigma = (1, 2)$

$N = \{\ \}$



Vietoris-Rips Complexes

Inductive VR expansion:

Input: The 1-skeleton G = (V, E) of VR(r)

Output: The k-skeleton K of the Vietoris-Rips complex VR(r)

INDUCTIVE-VR(G, k)

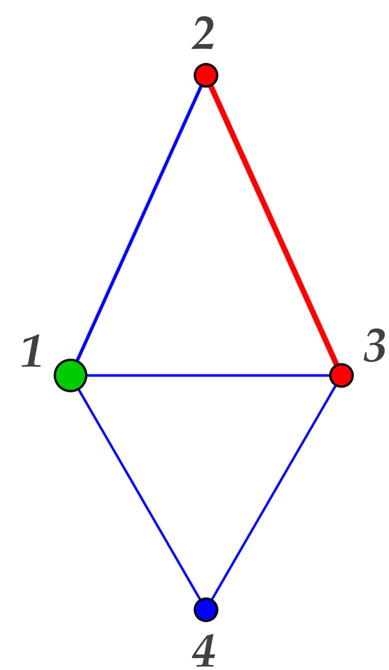
$K = V \cup E$
for $i = 1$ **to** k
  foreach i-simplex $\sigma \in K$
    $N = \cap_{u \in \sigma}$ LOWER-NBRS(G, u)
      foreach $v \in N$
        $K = K \cup \{\sigma \cup \{v\}\}$
  return K

LOWER-NBRS(G, u)

  return $\{v \in V \mid v < u, (u, v) \in E\}$

$\sigma = (2, 3)$

$N = \{1\}$



Vietoris-Rips Complexes

Inductive VR expansion:

Input: The 1-skeleton $G = (V, E)$ of $VR(r)$

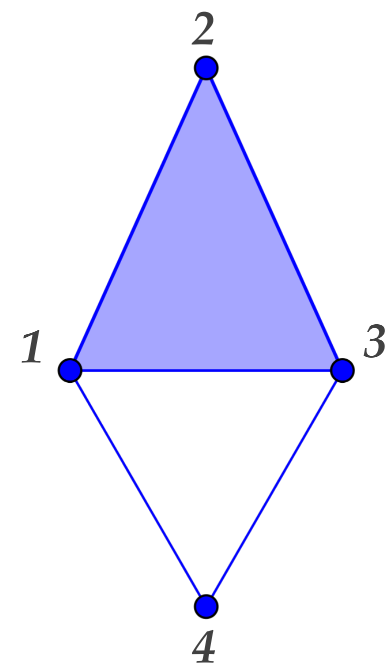
Output: The k-skeleton K of the Vietoris-Rips complex $VR(r)$

INDUCTIVE-VR(G, k)

```
K = V ∪ E
for i = 1 to k
   foreach i-simplex σ ∈ K
      N = ∩_{u∈σ} LOWER-NBRS(G, u)
         foreach v ∈ N
            K = K ∪ { σ ∪ {v}}
   return K
```

LOWER-NBRS(G, u)

```
   return {v ∈ V | v < u , (u, v) ∈ E}
```



Vietoris-Rips Complexes

Inductive VR expansion:

Input: The 1-skeleton G = (V, E) of VR(r)

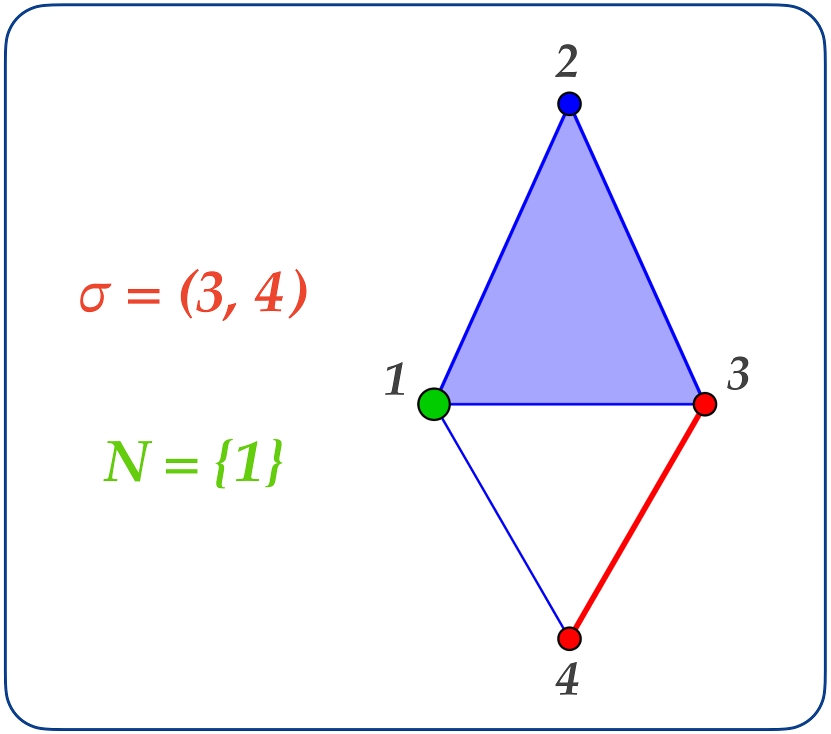
Output: The k-skeleton K of the Vietoris-Rips complex VR(r)

INDUCTIVE-VR(G, k)

$K = V \cup E$
for $i = 1$ **to** k
  foreach i-simplex $\sigma \in K$
    $N = \cap_{u\in\sigma}$ LOWER-NBRS(G, u)
      foreach $v \in N$
        $K = K \cup \{ \sigma \cup \{v\}\}$
  return K

LOWER-NBRS(G, u)

  return $\{v \in V \mid v < u , (u, v) \in E\}$



Vietoris-Rips Complexes

Inductive VR expansion:

Input: The 1-skeleton G = (V, E) of VR(r)

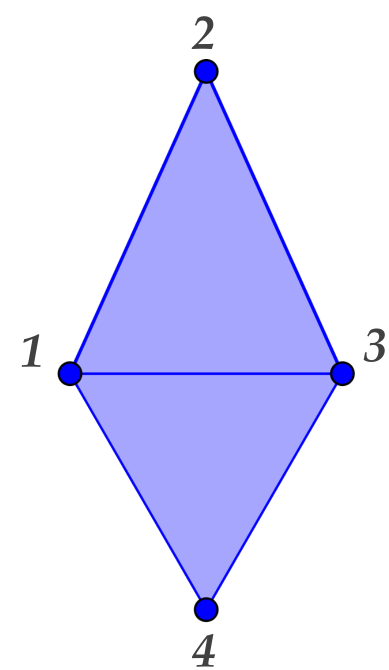
Output: The k-skeleton K of the Vietoris-Rips complex VR(r)

INDUCTIVE-VR(G, k)

$K = V \cup E$
for $i = 1$ ***to*** k
 foreach *i-simplex* $\sigma \in K$
  $N = \cap_{u\in\sigma}$ *LOWER-NBRS(G, u)*
   foreach $v \in N$
    $K = K \cup \{ \sigma \cup \{v\}\}$
 return K

LOWER-NBRS(G, u)

 return $\{v \in V \mid v < u , (u, v) \in E\}$



From Data to Complexes

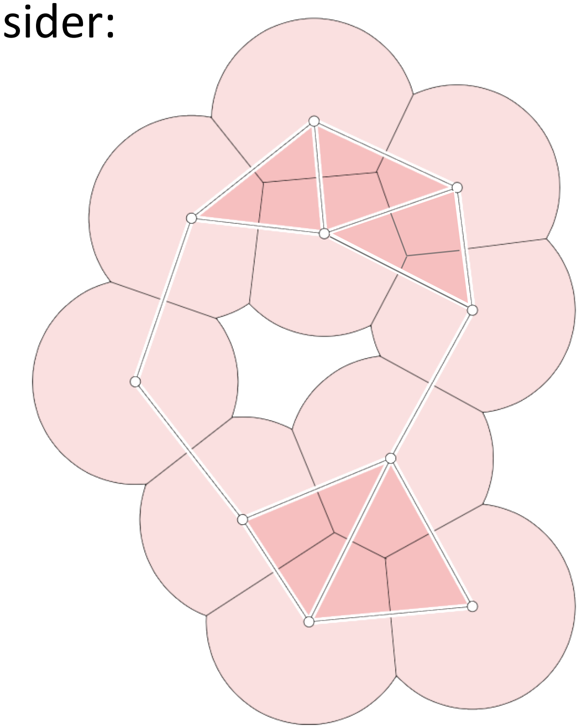
		
Delaunay triangulation	Bounded Dimension	Trivial Homology
Čech/VR complex	“Real” Homology	High Dimension Large Size

Alpha-Shapes

Definition:

Given a finite set of points V in general position of $\mathbb{R}^n$, let us consider:

- ***$A_u(r) := B_u(r) \cap R_V(u)$**, the **intersection** of the **closed ball** with **center u** $\in V$ and **radius r** and the **Voronoi region** of u*
- ***S**, the collection of these convex sets*



The ***alpha-shape Alpha(r)*** of V of radius r is the ***nerve of S***

Formally,

$$Alpha(r) := \{\sigma \subseteq V \mid \bigcap_{u \in \sigma} A_u(r) \neq \emptyset\}$$

$$A_u(r) \subseteq B_u(r) \Rightarrow Alpha(r) \subseteq \check{C}ech(r)$$

Image from [Edelsbrunner, Harer 2010]

Witness Complexes

Motivation:

The "shape" of a point cloud can be captured ***without considering all the input points***

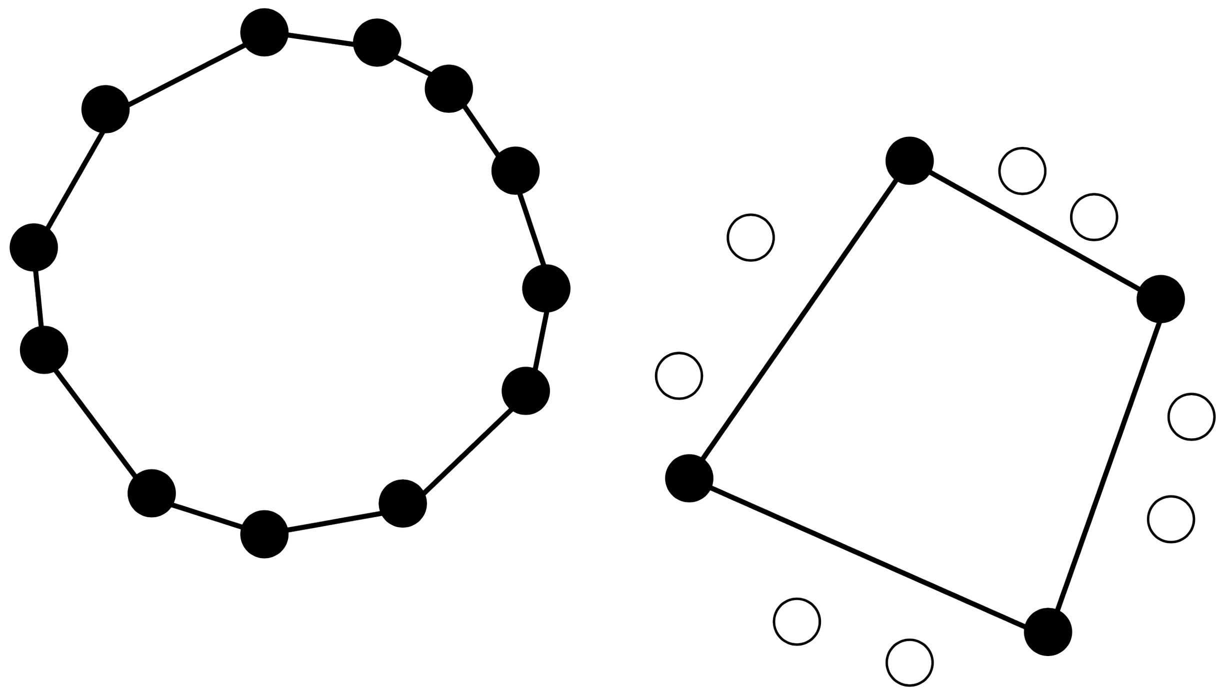
Definitions:

- ***Landmarks:***

  Selected points

- ***Witnesses:***

  Remaining points

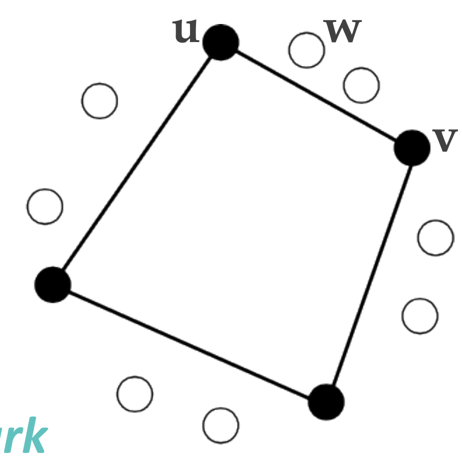


Images from [de Silva, Carlsson 2004]

Witness Complexes

Definition:

The ***witness complex W(r)*** of radius *r* is defined by:



- *u is in W(r) if u is a landmark*
- *(u, v) is in W(r) if there exists a witness w such that*

$$max\{d(u,w), d(v,w)\} \leq m_w + r$$

where ***m_w*** *: = the distance of w from the* ***2nd closest landmark***

- *the i-simplex σ is in W(r) if all its edges belong to W(r)*

$W_0(r)$ is defined by setting $m_w = 0$ for any witness w

$$W_0(r) \subseteq VR(r) \subseteq W_0(2r)$$

From Data to Complexes

Not Only Point Clouds in $\mathbb{R}^n$

Most of the presented constructions can be ***generalized/adapted*** to the case of

a finite collection of elements endowed with a notion of proximity*

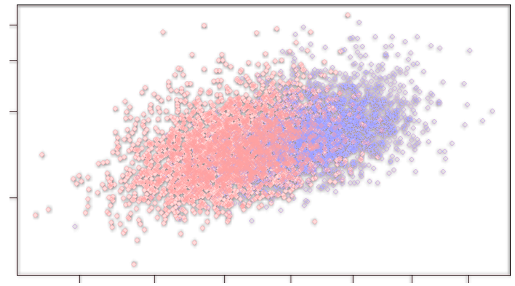
enabling to cover a **wide plethora of datasets**

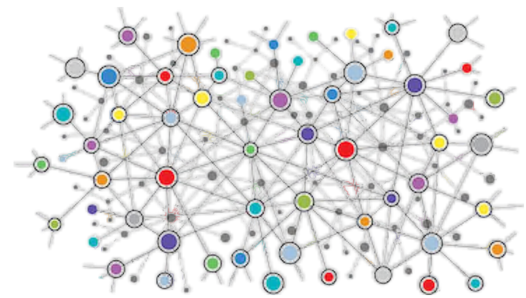
More properly, a **semi-metric, i.e. a distance not necessarily satisfying the triangle inequality*

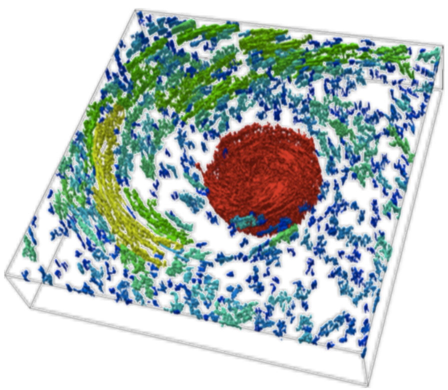
From Data to Complexes

Not Only Point Clouds in $\mathbb{R}^n$

- **Point Clouds:**
  - Delaunay triangulation
  - Čech complexes
  - Vietoris-Rips complexes
  - Alpha-shapes
  - Witness complexes complexes
- **Graphs and Complex Networks:**
  - Flag complexes
- **Functions:**
  - Sublevel sets





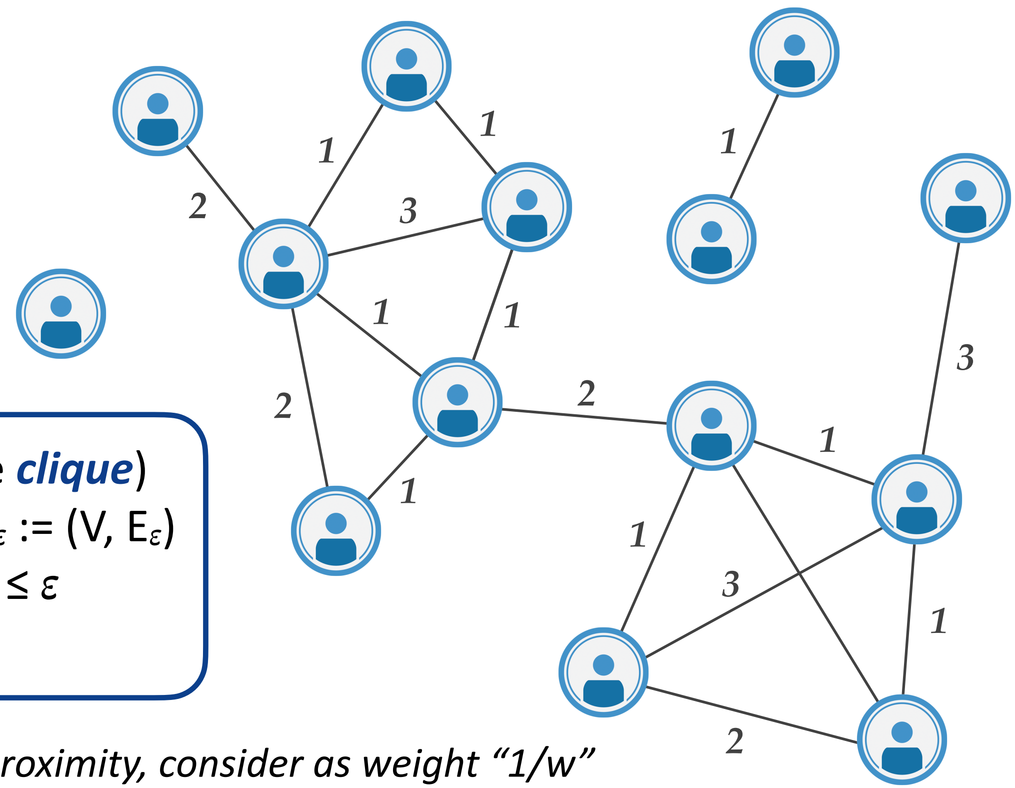


From Data to Complexes

Flag Complex of a Weighted Network:

Let G := (V, E, w: E ⟶ ℝ) be a ***weighted undirected graph*** representing a ***network***:

Nodes of V	⟺	*Entities or individuals*
Arcs of E		*Ties between entities*
Weights of w		*Proximity of entities**

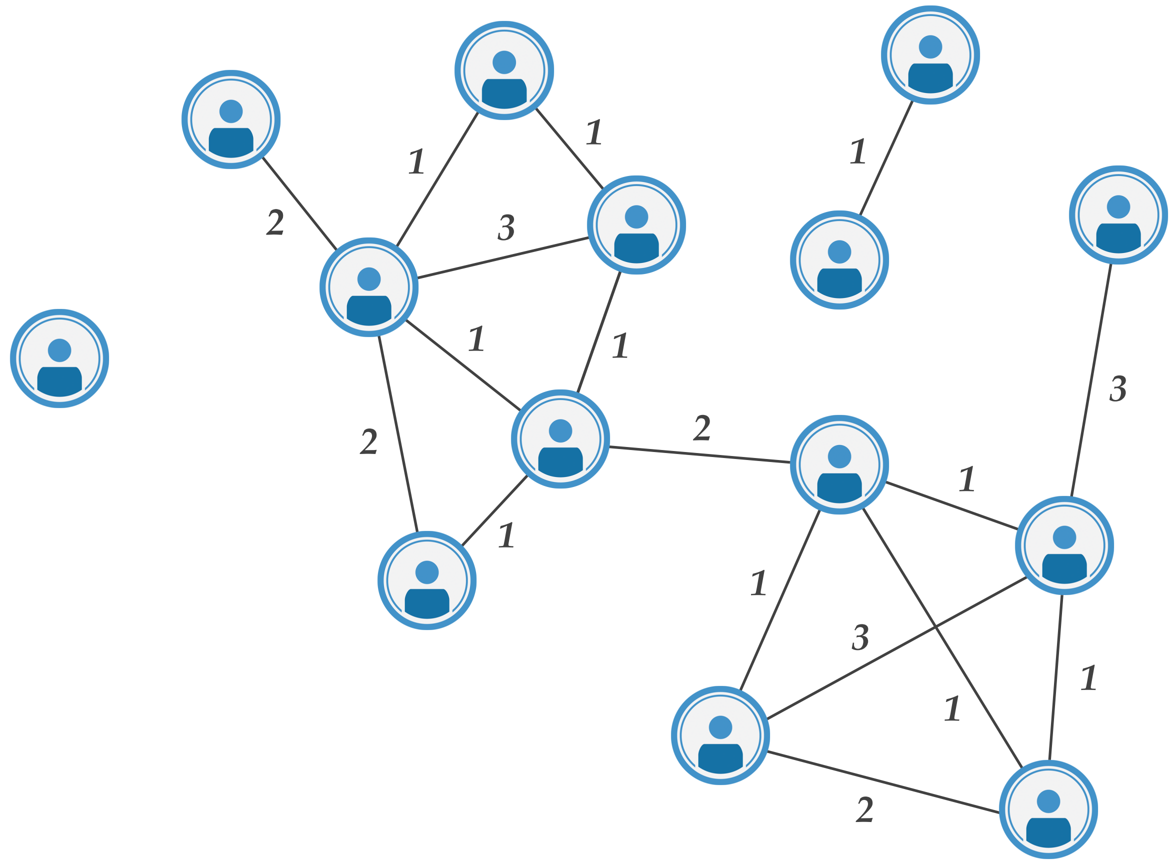


Fixed a ***weight threshold*** ε, the ***flag*** (or the ***clique***) ***complex*** is the ***VR expansion*** of the graph $G_\varepsilon := (V, E_\varepsilon)$ where E_ε are the arcs of E with weight $\leq \varepsilon$

**If w represents tie strengths rather than node proximity, consider as weight "1/w"*

From Data to Complexes

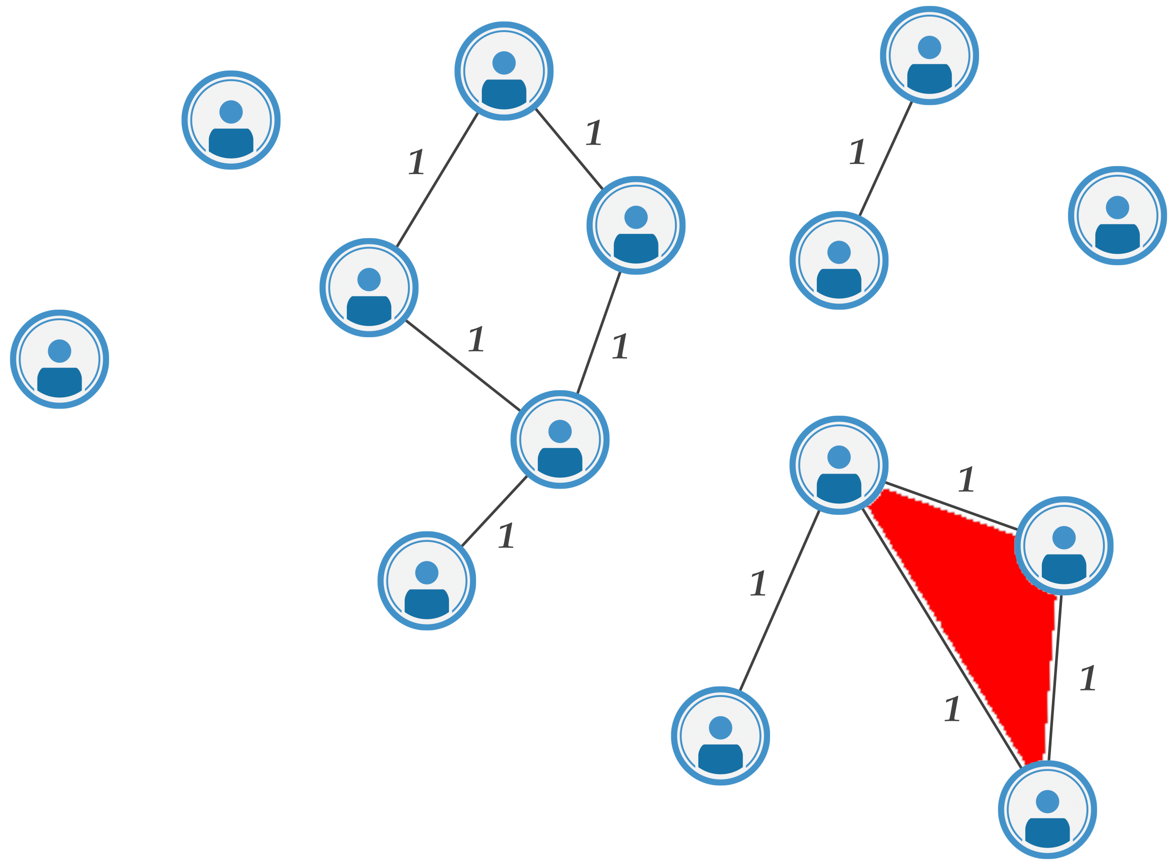
Flag Complex of a Weighted Network:



From Data to Complexes

Flag Complex of a Weighted Network:

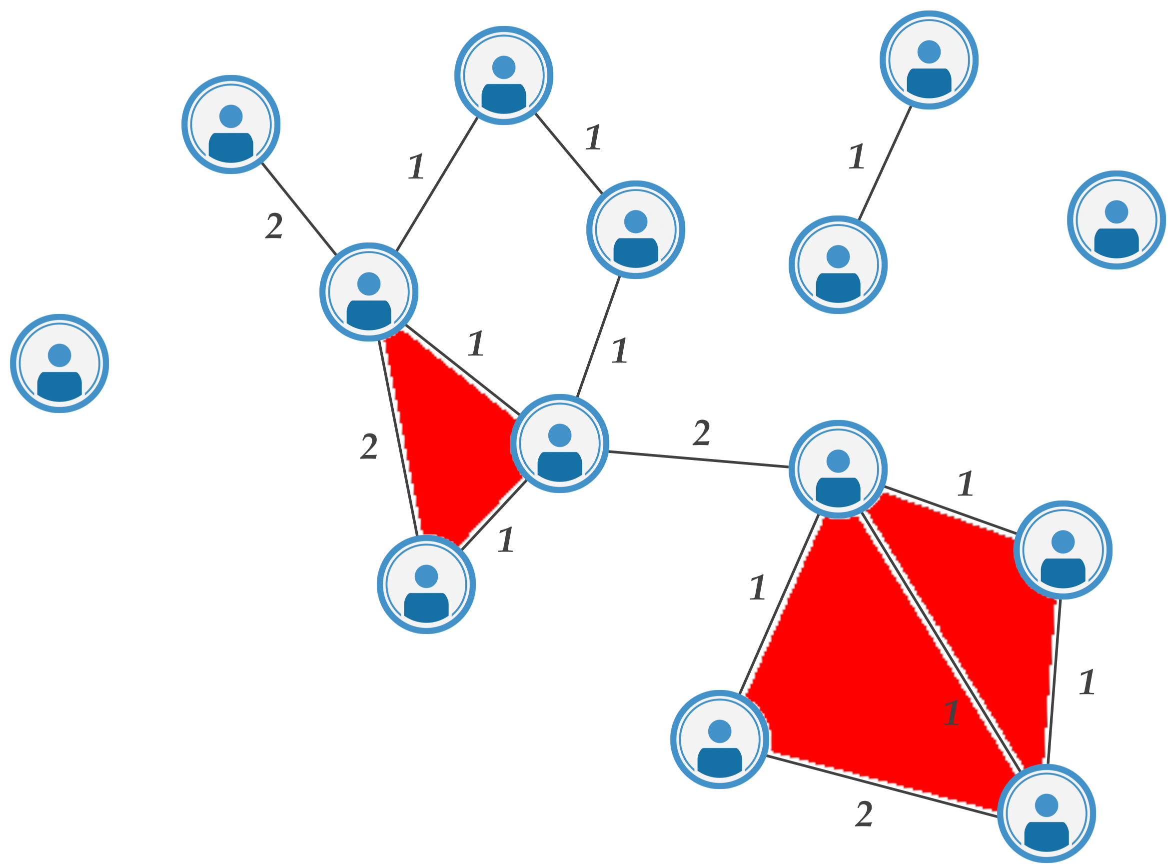
$\varepsilon = 1$



From Data to Complexes

Flag Complex of a Weighted Network:

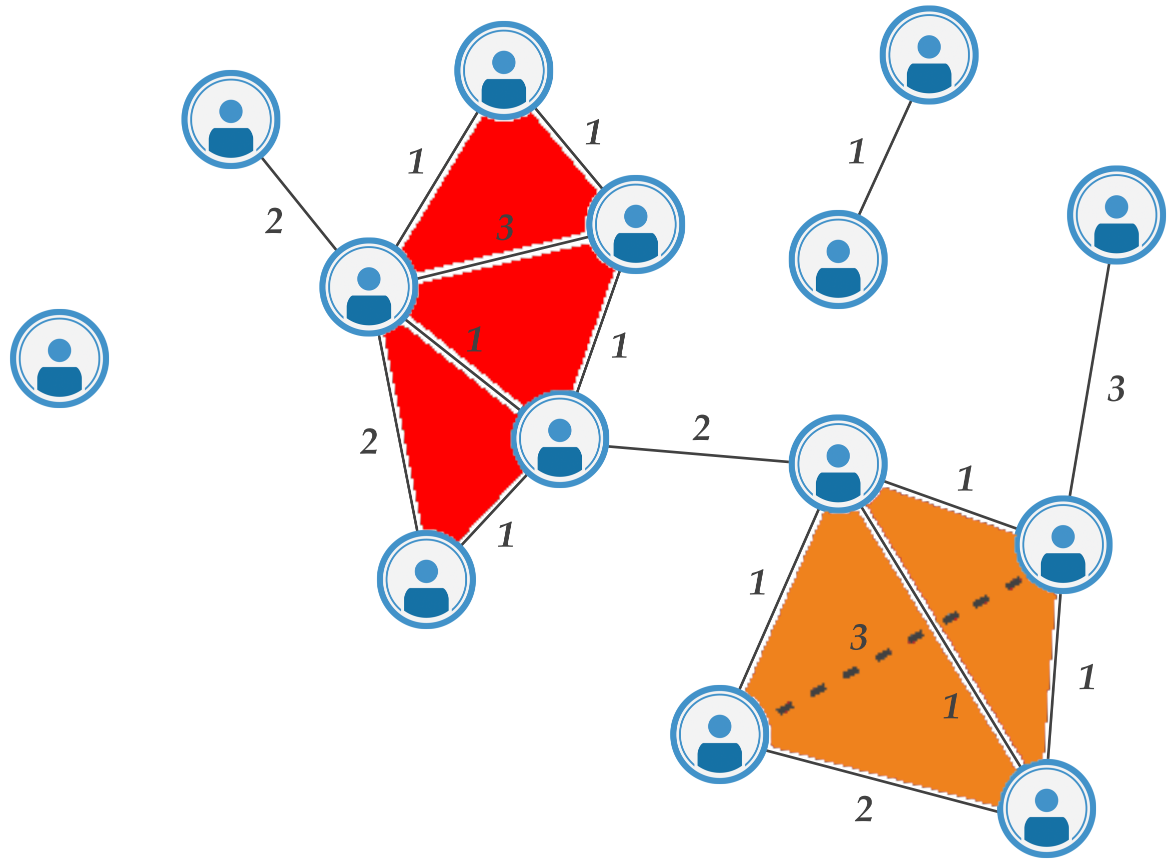
$\varepsilon = 2$



From Data to Complexes

Flag Complex of a Weighted Network:

$\varepsilon = 3$



From Data to Complexes

Sublevel Sets of Functions

Given a ***function*** f: D $\longrightarrow \mathbb{R}$,

- ***Step 1:***

  Transform f: D $\longrightarrow \mathbb{R}$ into a function **F: K** $\longrightarrow \mathbb{R}$ ***defined on a simplicial complex K***

  E.g. if D is a point cloud, construct from it a simplicial complex K and define F as

$$F(\sigma) := \max\{f(v) \mid v \text{ is a vertex of } \sigma\}$$

- ***Step 2:***

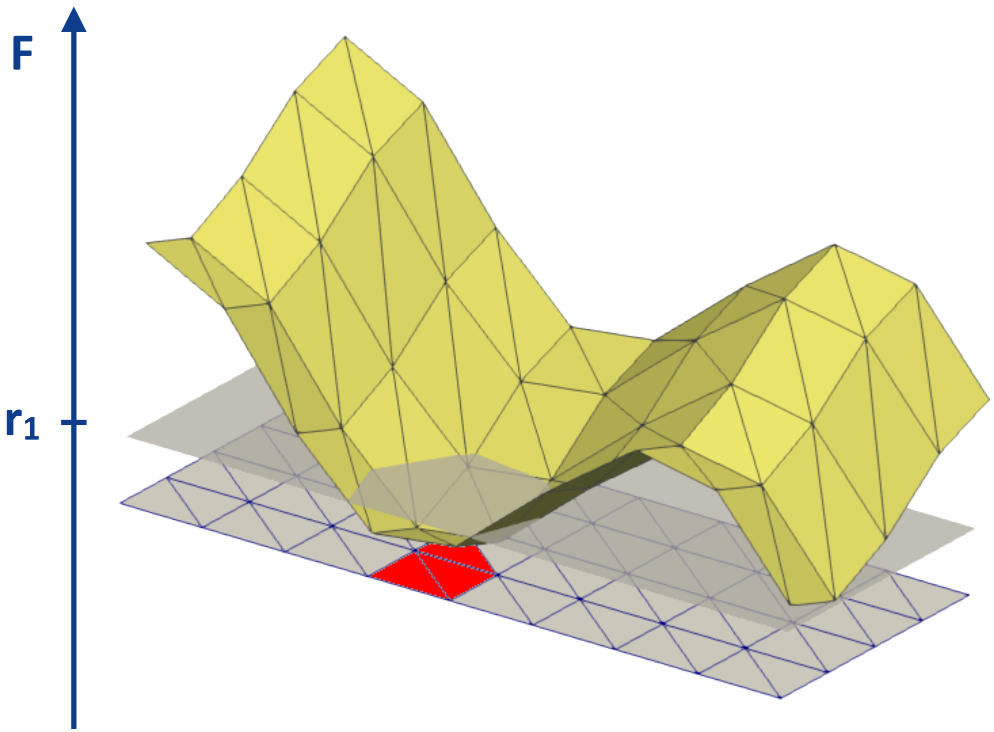
  Build the collection $\{ K^r \}_{r\in\mathbb{R}}$ of the ***sublevel sets of F*** defined as

$$K^r := \{\sigma \in K \mid F(\sigma) \leq r\}$$

  Notice that K^r is a simplicial complex whenever: if τ is a face of σ then $F(\tau) \leq F(\sigma)$

From Data to Complexes

Sublevel Sets of Functions

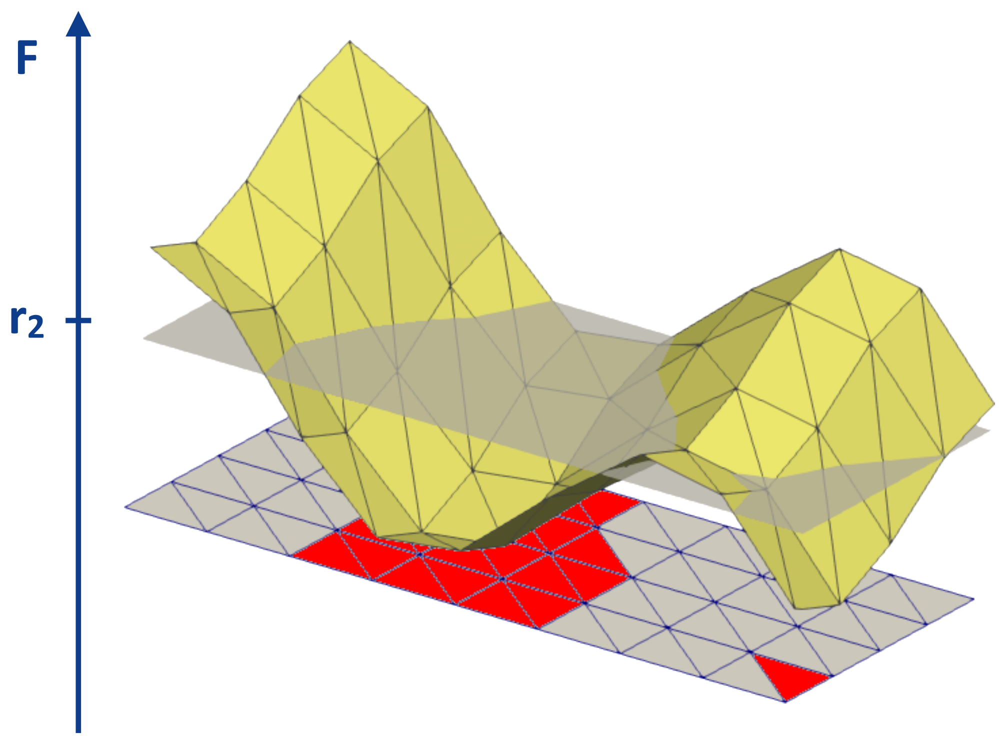


Given a function F: K ⟶ ℝ,

$$K^r := \{\sigma \in K \mid F(\sigma) \leq r\}$$

From Data to Complexes

Sublevel Sets of Functions

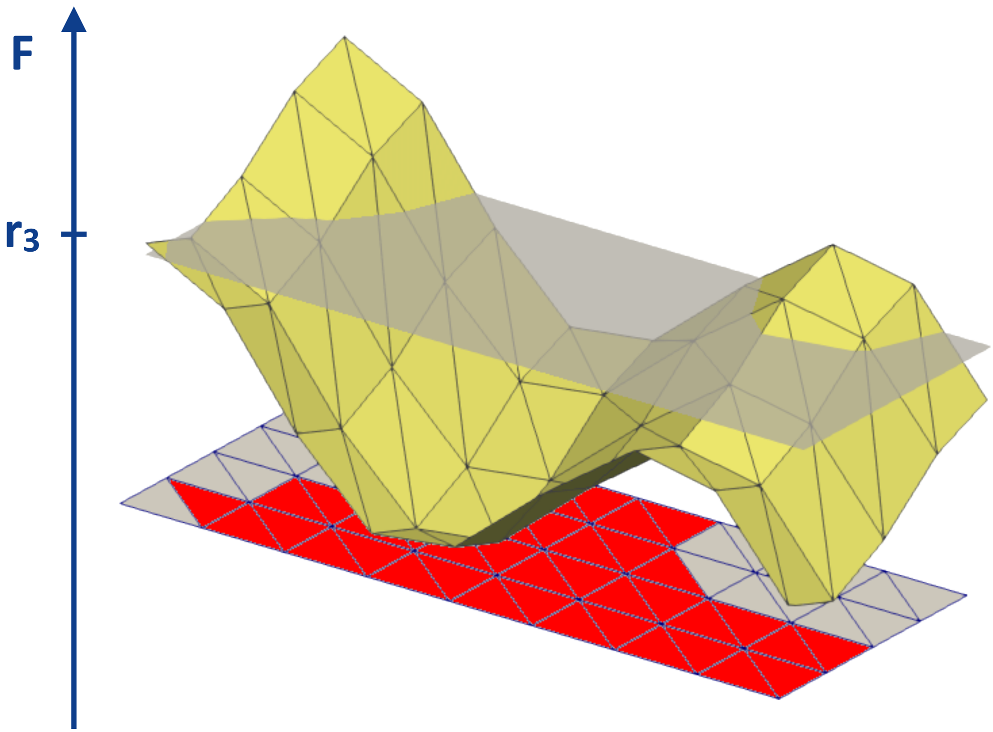


Given a function F: K $\longrightarrow \mathbb{R}$,

$$K^r := \{\sigma \in K \mid F(\sigma) \leq r\}$$

From Data to Complexes

Sublevel Sets of Functions



Given a function F: K $\longrightarrow \mathbb{R}$,

$$K^r := \{\sigma \in K \mid F(\sigma) \leq r\}$$

Bibliography

Some References:

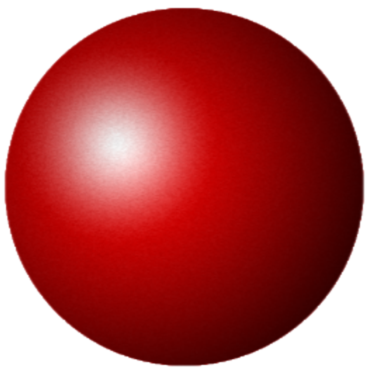
- ***From Data to Complexes:***

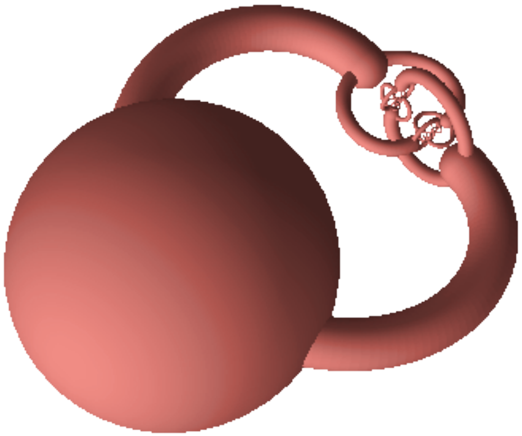
  - H. Edelsbrunner, ***Geometry and Topology for Mesh Generation***. Cambridge University Press, 2001.
  - V. de Silva, G. Carlsson. ***Topological estimation using witness complexes***. *SPBG 4*, pages 157-166, 2004.
  - A. Zomorodian, ***Fast construction of the Vietoris-Rips complex***. Computers & Graphics 34.3, pages 263-271, 2010.
  - H. Edelsbrunner. ***Algorithms in Combinatorial Geometry***. Springer Science & Business Media, 2012.

Persistent Homology

Persistent Homology

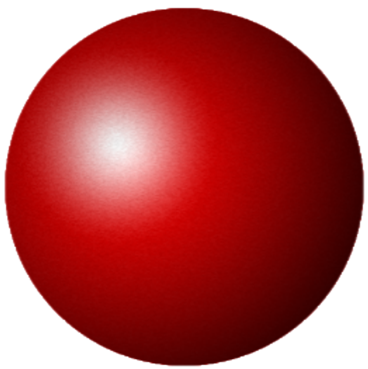
- *Do they have the same shape?*

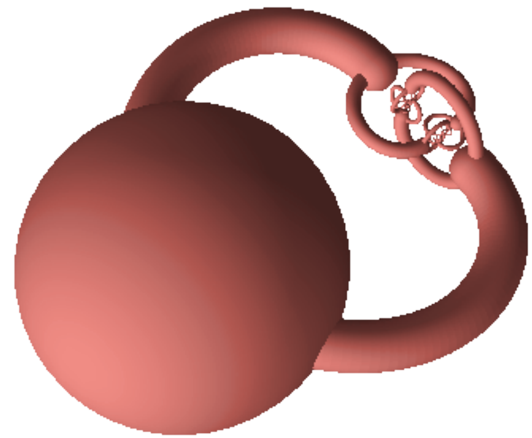




Persistent Homology

✦ *Do they have the same shape?*

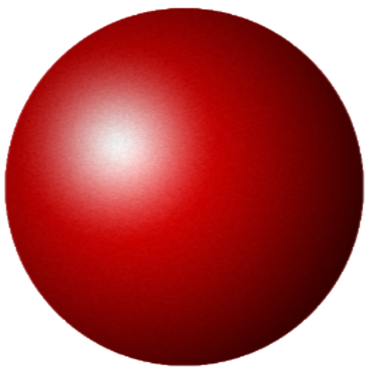


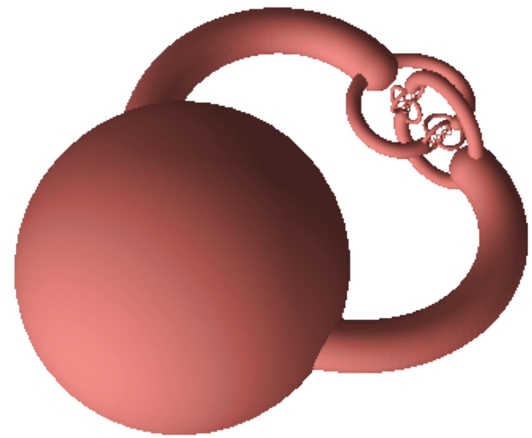


In Practice?	In Theory?

Persistent Homology

Do they have the same shape?





In Practice?

In Theory?

*They are **homeomorphic***

Persistent Homology

- *Do they have the same shape?*

Persistent Homology

✦ *Do they have the same shape?*

In Practice?

In Theory?

Persistent Homology

Do they have the same shape?

In Practice?

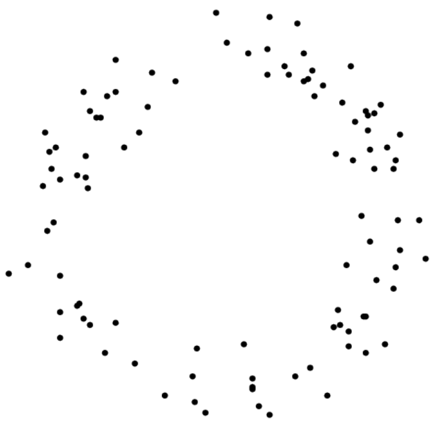
In Theory?

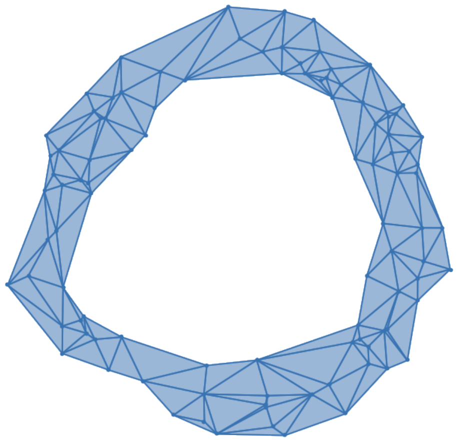
*They are **not homeomorphic***

Persistent Homology

✦ ***Which is the shape of a given data?***

Persistent homology allows for the retrieval of the ***"actual"*** homological information of a data





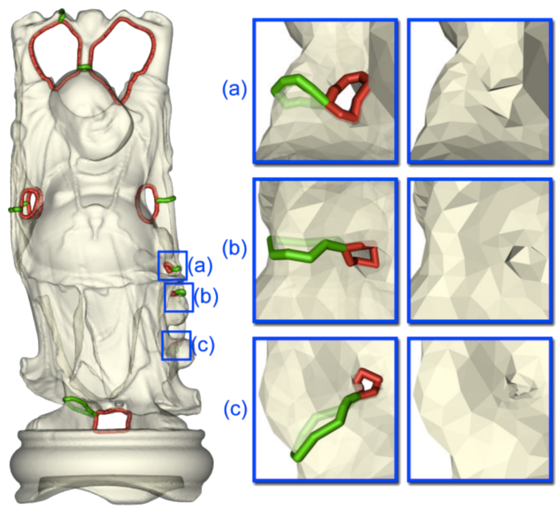
Point Cloud Dataset → ***Topological Nature of the "Underlying" Shape***

Image from [Bauer 2015]

Persistent Homology

✦ *Which is the shape of a given data?*

Persistent homology allows for the retrieval of the ***"actual"*** homological information of a data



Noisy Dataset → ***Relevant Homological Information***

Image from [Dey et al. 2008]

Persistent Homology

In a Nutshell:

Persistent homology allows for
describing the changes in the shape of an evolving object

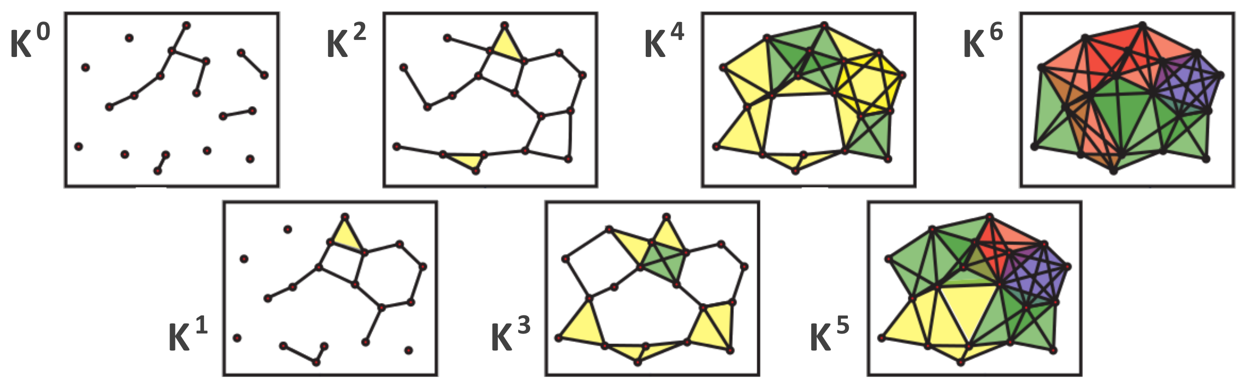


Image from [Ghrist 2008]

Persistent Homology

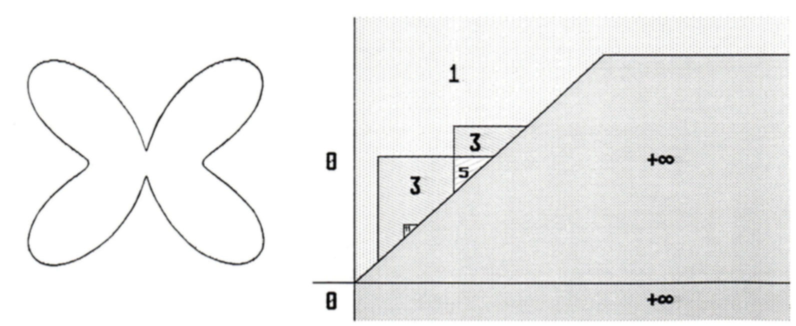
An Evolving Notion:

1990

Frosini

Size Functions:

- ***Estimation of natural pseudo-distance*** between shapes endowed with a function *f*
- Tracking of the ***connected components*** of a shape along its evolution induced by *f*



Actually, this coincides with ***persistent homology in degree 0***

Image from [Frosini 1992]

Persistent Homology

An Evolving Notion:

Incremental Algorithm for Betti Numbers:

- Introduction of the notion of ***filtration***
- De facto computation of ***persistence pairs***

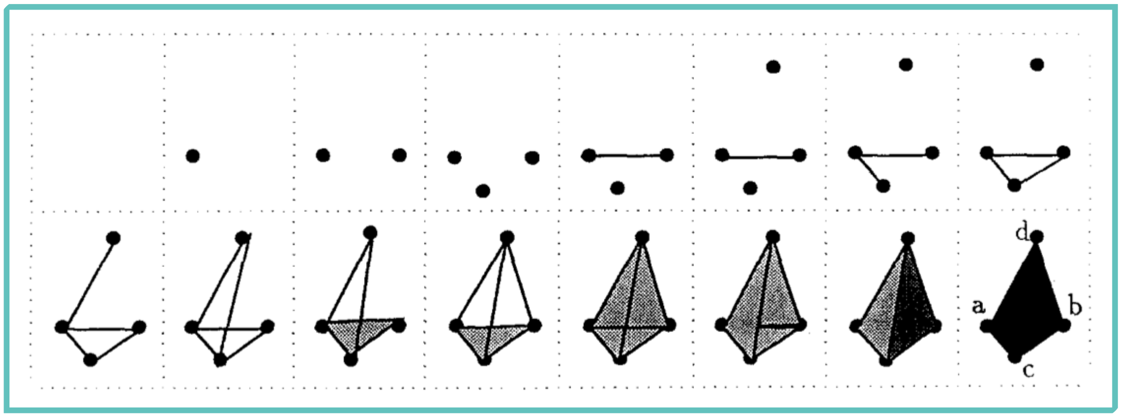


Image from [Delfinado, Edelsbrunner 1995]

Persistent Homology

An Evolving Notion:

1990 — Frosini

1994 — Delfinado, Edelsbrunner

1999 — Robins

Homology from Finite Approximations:

- ***Extrapolation of the homology*** of a metric space from a ***finite point-set approximation***
- Introduction of ***persistent Betti numbers***

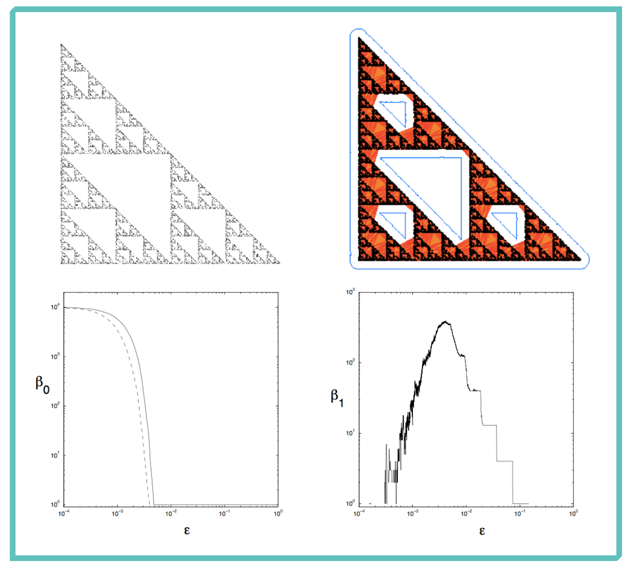


Image from [Robins 1999]

Persistent Homology

An Evolving Notion:

1990 — Frosini

1994 — Delfinado, Edelsbrunner

1999 — Robins

2002 — Edelsbrunner, Letscher, Zomorodian

Topological Persistence:

- Introduction and algebraic formulation of the notion of ***persistent homology***
- ***Description of an algorithm*** for computing persistent homology

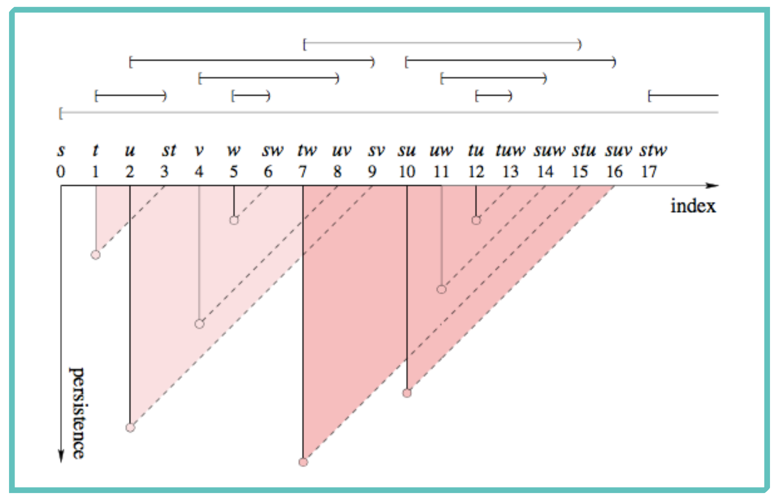
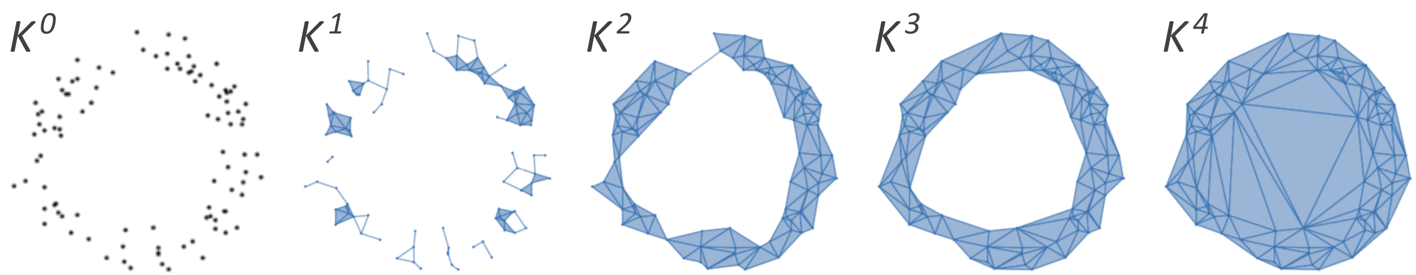


Image from [Edelsbrunner et al. 2002]

Persistent Homology

Definition:

Intuitively, a ***filtration*** $\mathcal{F}$ is a finite ***"growing" sequence*** of simplicial complexes

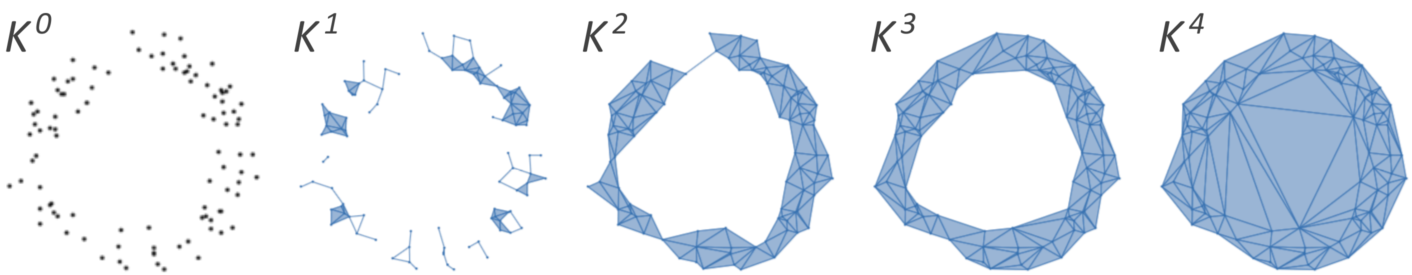


Formally, a ***filtration*** $\mathcal{F}$ *of a simplicial complex K is a collection of subcomplexes* $\{ K^p \}_{p \in \mathbb{R}}$ *of K for which,* ***given any p, q*** $\in \mathbb{R}$ ***such that*** $p \leq q$,

$$K^p \subseteq K^q$$

Persistent Homology

Most of the techniques ***transforming a dataset into a simplicial complex depending on the choice of a parameter*** *actually produce a filtration* $\{ K^p \}_{p \in \mathbb{R}}$



Working Assumption:

We can always pretend that parameter p varies over $\mathbb{N}$

Persistent Homology

Definition:

Given a filtration $\mathcal{F} := \{ K^p \}_{p \in \mathbb{N}}$*, a value* $i \in \mathbb{N}$*, and a field* $\mathbb{F}$*, the* ***i^{th} persistence module M of*** $\mathcal{F}$ ***over*** $\mathbb{F}$ *is defined as the* ***finitely generated graded $\mathbb{F}[x]$-module***

$$M := \bigoplus_{p \in \mathbb{N}} M_p$$

where:

- ***$M_p := H_i(K^p; \mathbb{F})$****, the set of* ***homogeneous elements of grade p***
- *The* ***action x^{q-p} h over an element h of grade p*** *is defined as* ***$\mu_i^{p,q}(h)$****, where:*
  - ***$\mu_i^{p,q}(h): H_i(K^p; \mathbb{F}) \longrightarrow H_i(K^q; \mathbb{F})$*** *is the linear map induced by the inclusion* $K^p \subseteq K^q$

Persistent Homology

Theorem (structure for finitely generated graded modules over a PID):

Any persistence module M can be expressed as

$$M \cong \bigoplus_{k=1}^{n} \mathbb{F}[x](-r_k) \oplus \bigoplus_{j=1}^{m} \Big(\mathbb{F}[x]/(x^{q_j - p_j})\Big)(-p_j)$$

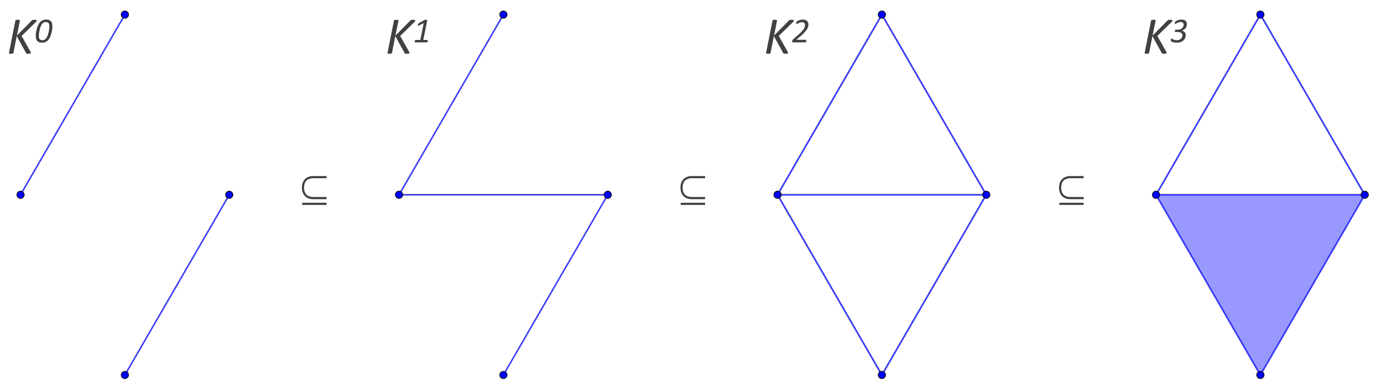
So, ***M is completely determined by the collection of values r_k and of pairs (p_j, q_j)***

Such descriptors are typically expressed as pairs, called ***persistence pairs*** of M, of the kind (r_k, ∞) and (p_j, q_j)

Persistent Homology

Intuitively:

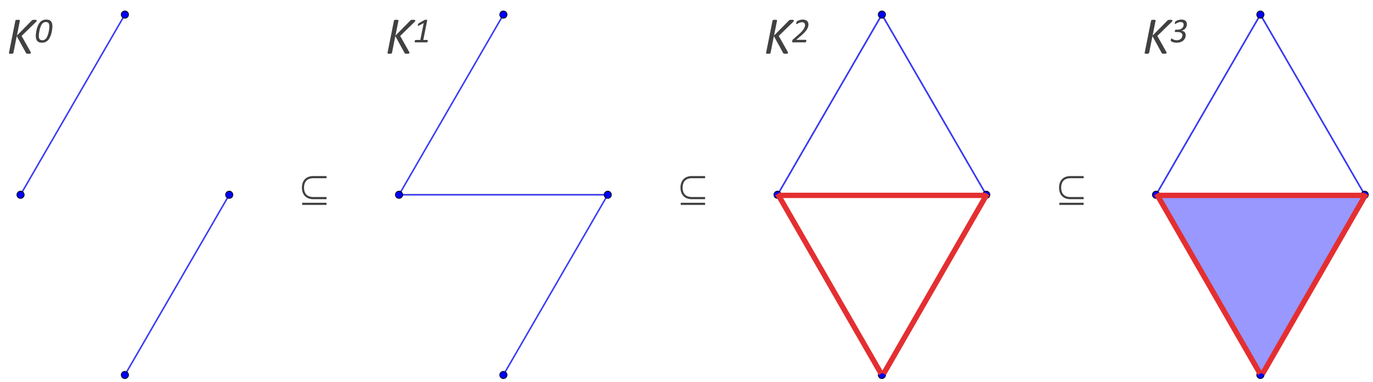
Given a filtration $\mathcal{F} := \{ K^p \}_{p \in \mathbb{N}}$, a ***persistence pair (p,q)*** $\in \mathbb{N} \times (\mathbb{N} \cup \{\infty\})$ *with $p < q$ represents a* ***homological class*** *that is* ***born at step p*** *and* ***dies at step q***



Persistent Homology

Intuitively:

Given a filtration $\mathcal{F} := \{ K^p \}_{p \in \mathbb{N}}$, a ***persistence pair (p,q)*** $\in \mathbb{N} \times (\mathbb{N} \cup \{\infty\})$ *with $p < q$ represents a* ***homological class*** *that is* ***born at step p*** *and* ***dies at step q***

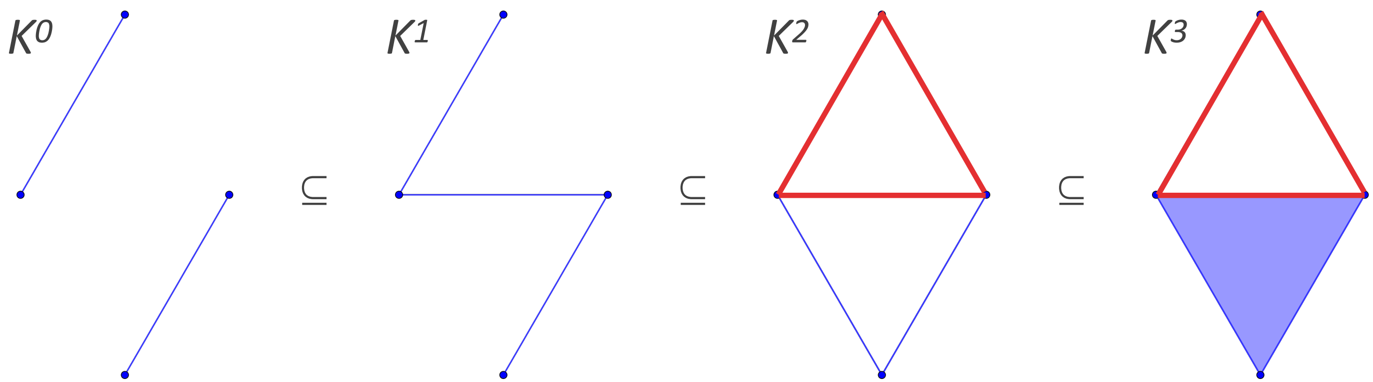


(2, 3)

Persistent Homology

Intuitively:

Given a filtration $\mathcal{F} := \{ K^p \}_{p \in \mathbb{N}}$, a ***persistence pair (p,q)*** $\in \mathbb{N} \times (\mathbb{N} \cup \{\infty\})$ *with $p < q$ represents a* ***homological class*** *that is* ***born at step p*** *and* ***dies at step q***



(2, ∞) essential pair

Persistent Homology

Differently from homology, persistent homology provides
a notion of "shape" closer to our everyday perception

It is possible to ***compare two shapes*** by comparing their ***homology groups***

Persistent Homology

Differently from homology, persistent homology provides
a notion of "shape" closer to our everyday perception

It is possible to ***compare two shapes*** by comparing their ***homology groups***

PERSISTENCE PAIRS

Persistent Homology

Differently from homology, persistent homology provides
a notion of "shape" closer to our everyday perception

It is possible to ***compare two shapes*** by comparing their ***homology groups***

PERSISTENCE PAIRS

In order to better perform the above task, we need:

- ***Visual*** and ***descriptive representations*** for persistence pairs
- Notions of ***distance*** between sets of persistence pairs and ***stability results***

Bibliography

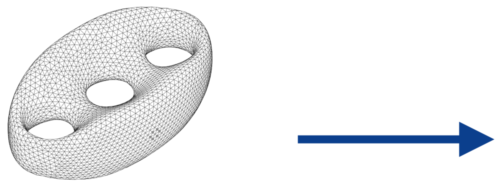
Some References:

- ***Persistent Homology:***
  - U. Fugacci, S. Scaramuccia, F. Iuricich, L. De Floriani. ***Persistent homology: a step-by-step introduction for newcomers***. Eurographics Italian Chapter Conference, pages 1-10, 2016.

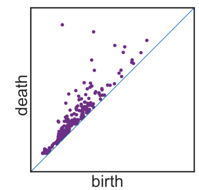
Visualizing Persistence

Persistent Homology

(Persistent) Homology allows for assigning to any (filtered) simplicial complex
topological information ***expressed in terms of*** ***algebraic structures***



Shape



Features

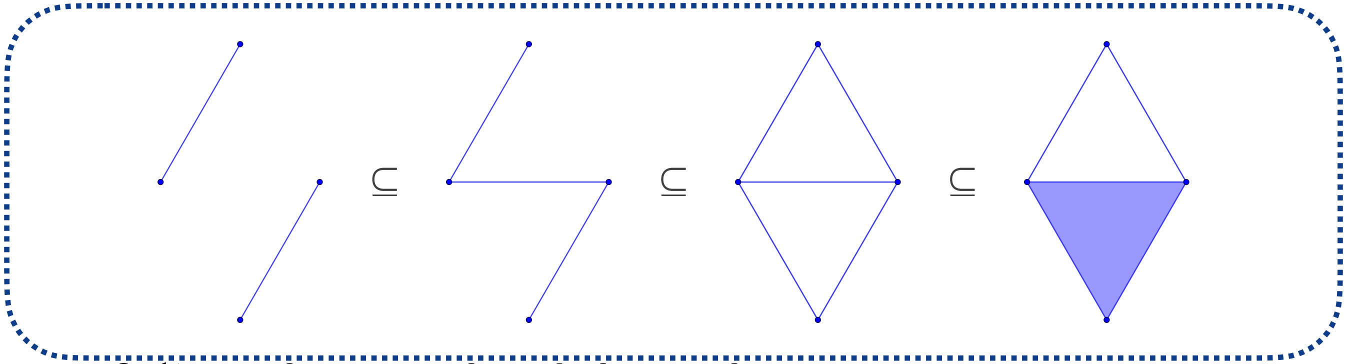
Goal:

We address two main questions:

- ***Can this topological information be characterized in a simpler and "more visualizable" way?***
- ***Is this information stable under small perturbations of the input data?***

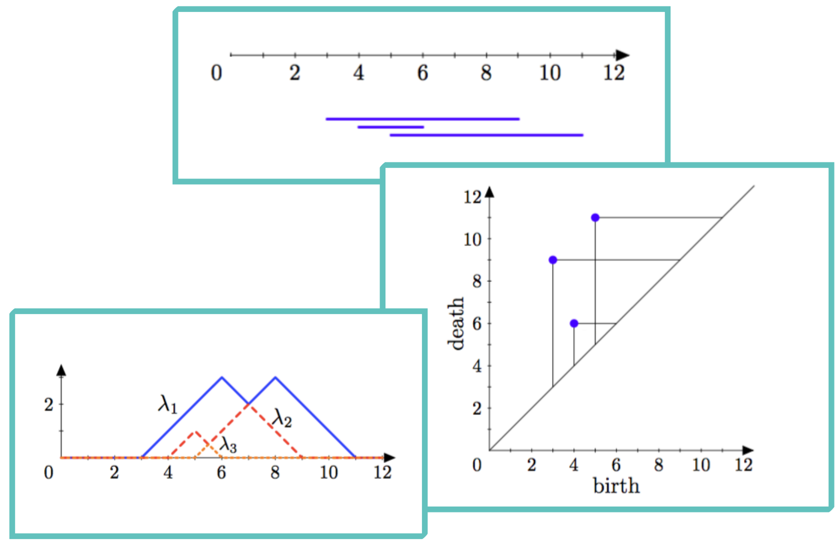
Visualizing Persistence

Given a filtration $\mathcal{F}$,



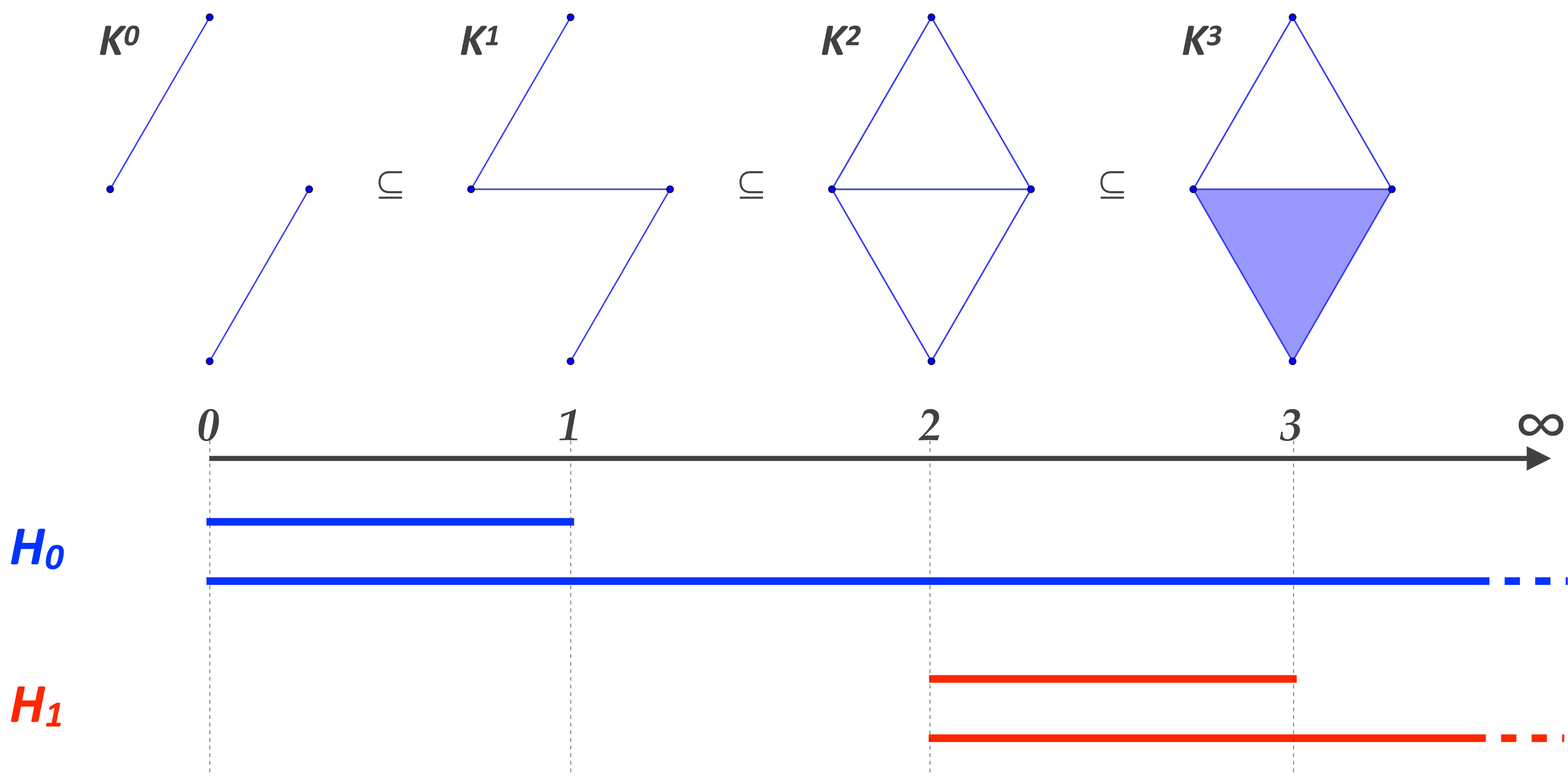
Persistent pairs of $\mathcal{F}$ can be visualized through:

- ***Barcodes*** [Carlsson et al. 2005; Ghrist 2008]
- ***Persistence diagrams*** [Edelsbrunner, Harer 2008]
- ***Persistence landscapes*** [Bubenik 2015]
- *Corner points and lines* [Frosini, Landi 2001]
- *Half-open intervals* [Edelsbrunner et al. 2002]
- *k-triangles* [Edelsbrunner et al. 2002]



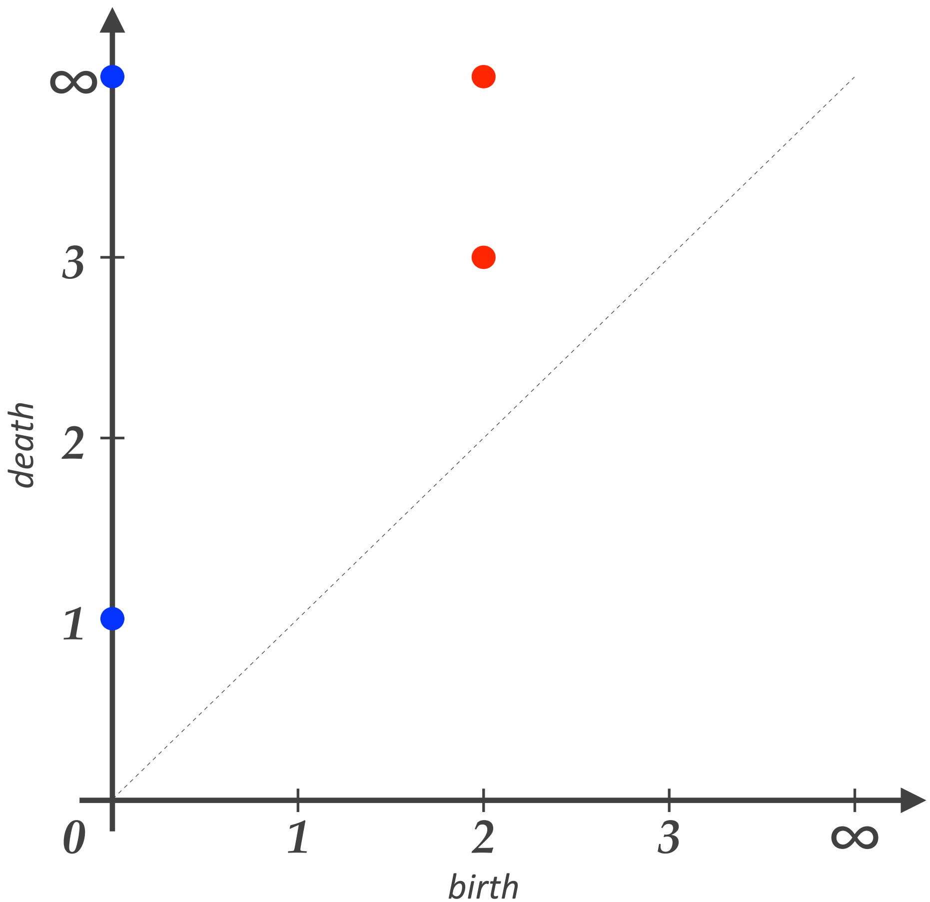
Visualizing Persistence

Barcodes: *Persistence pairs are represented as* ***intervals in*** $\mathbb{R}$



Visualizing Persistence

Persistence Diagrams: *Persistence pairs are represented as* ***points in*** $\mathbb{R}^2$

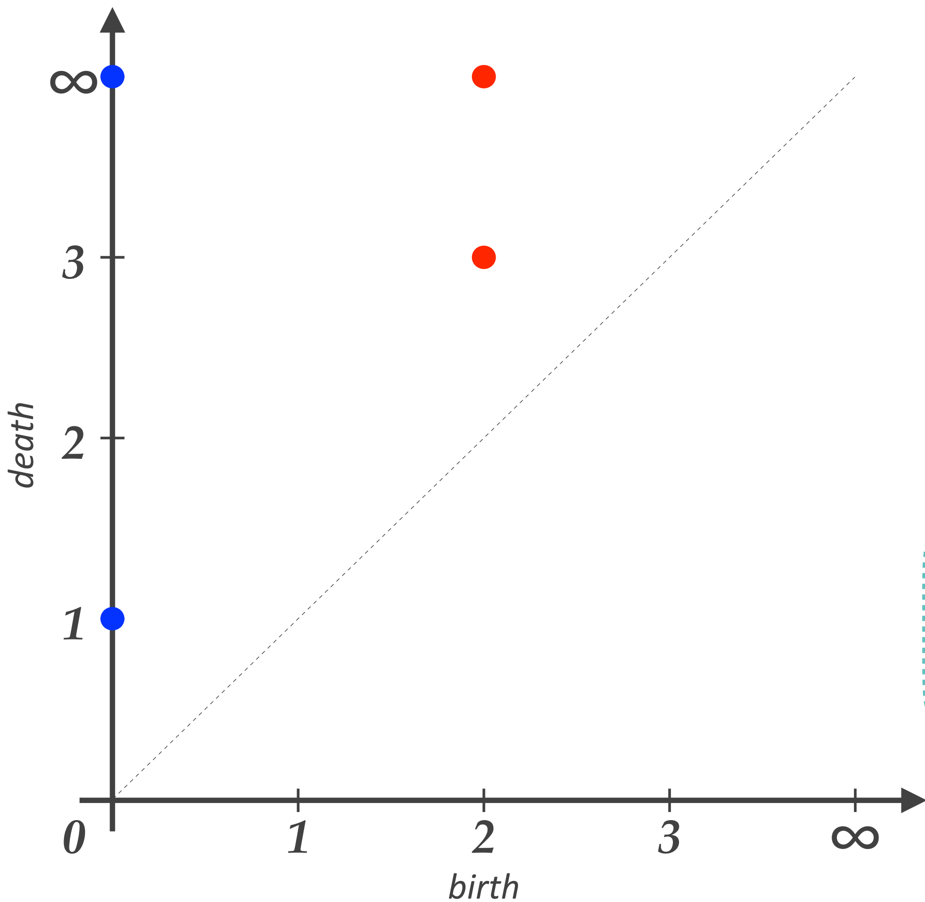


H_0 (0, 1) (0, ∞)

H_1 (2, 3) (2, ∞)

Visualizing Persistence

Persistence Diagrams: *Persistence pairs are represented as* ***points in*** $\mathbb{R} \times (\mathbb{R} \cup \{\infty\})$



H_0 (0, 1) (0, ∞)

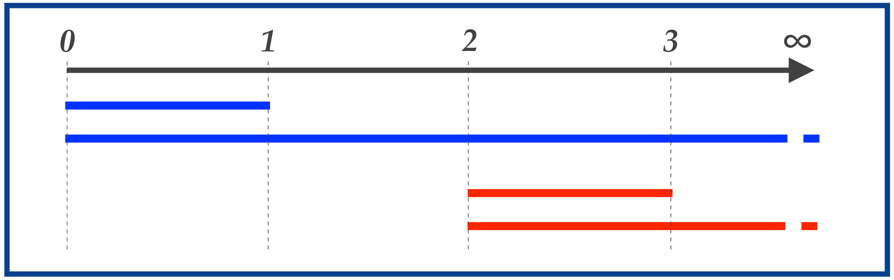
H_1 (2, 3) (2, ∞)

Formally, a persistence diagram is a ***multiset***

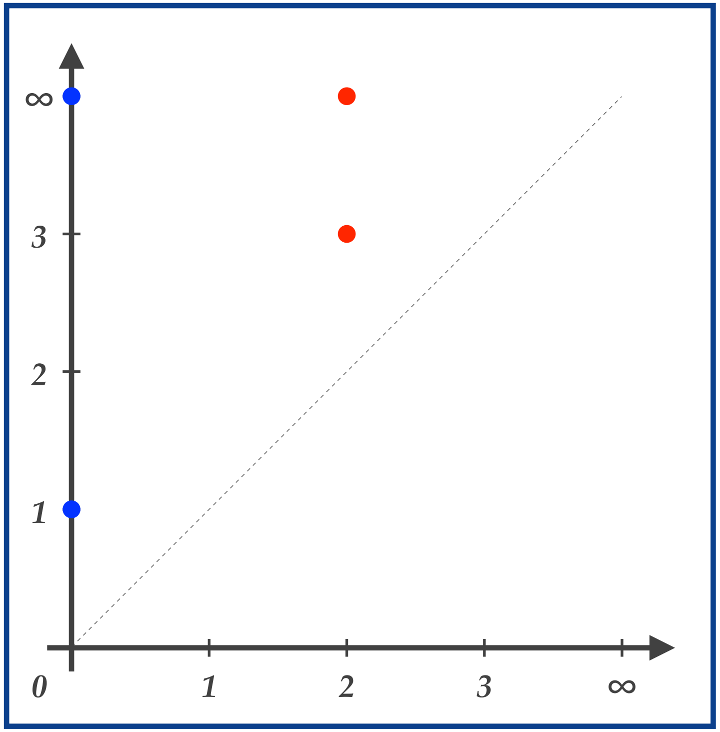
- Points are endowed with **multiplicity**

Visualizing Persistence

*Both tools **visually represent** the **lifespan** of the homology classes:*



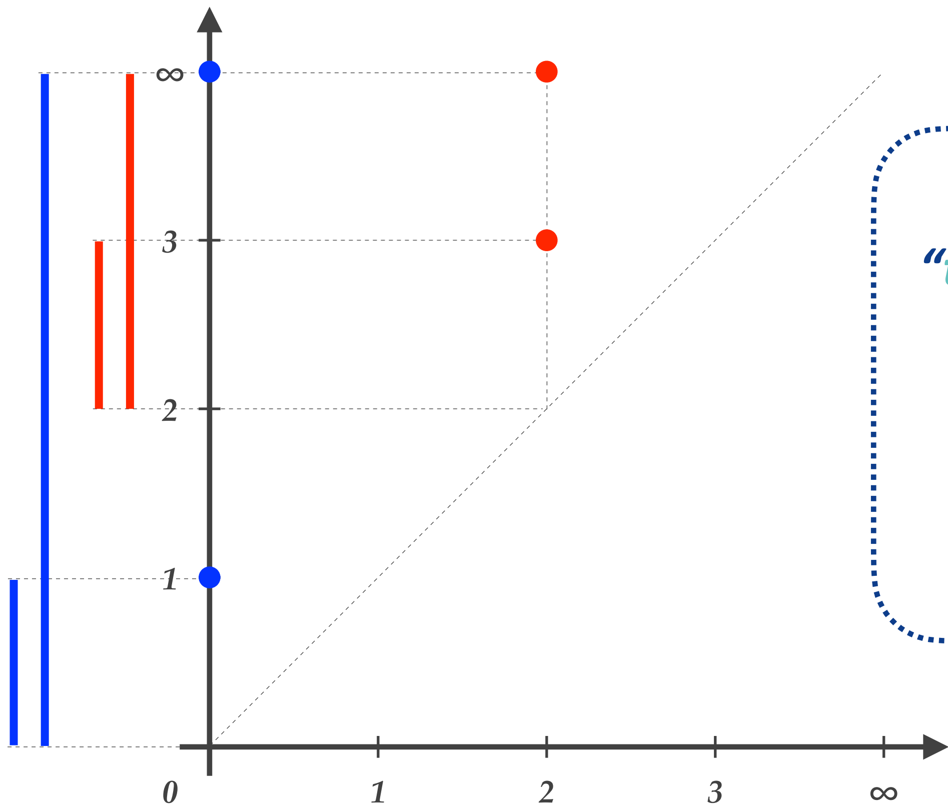
- **Barcode:** ***length of the intervals***
- **Persistence Diagram:** ***distance from the diagonal***



Barcodes and Persistence Diagrams ***encode equivalent information***

Visualizing Persistence

Barcodes and Persistence Diagrams ***encode equivalent information***



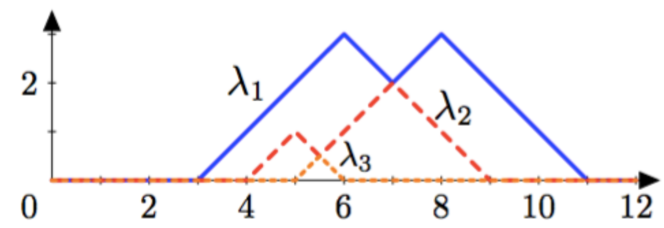
A visualization can be easily ***"translated"*** into the other one:

[p, q]	⟷	***(p, q)***
[p, ∞)		***(p, ∞)***

Visualizing Persistence

Persistence Landscapes:

***Persistence landscapes** are statistics-friendly representations of persistence pairs*



Given a persistence module M, persistence landscapes

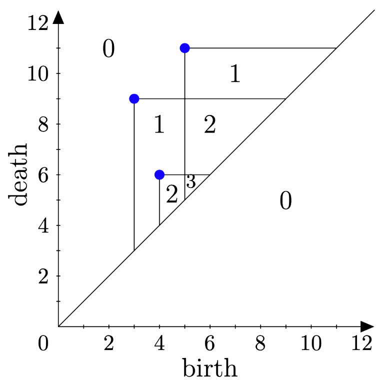
- Consist of a collection of ***1-Lipschitz functions***
- Lie in a ***vector space***
- Are ***stable*** (under small perturbations of the input filtration)

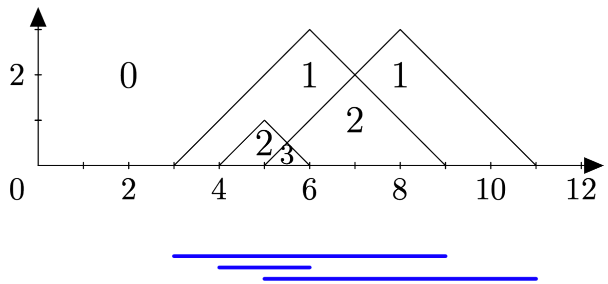
Image from [Bubenik 2015]

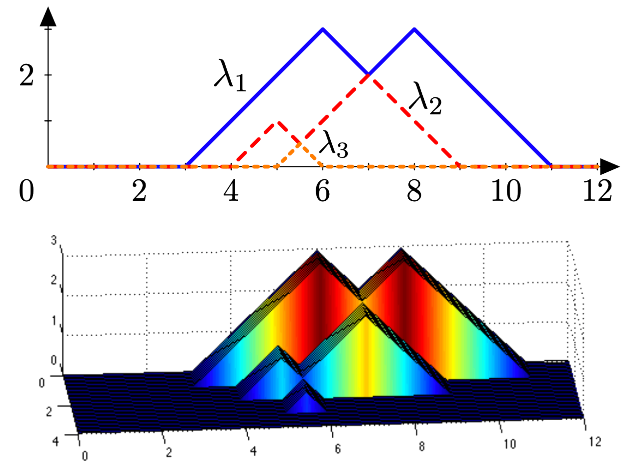
Visualizing Persistence

Persistence Landscapes:

Given a persistence module M,







Images from [Bubenik 2015]

Formally,

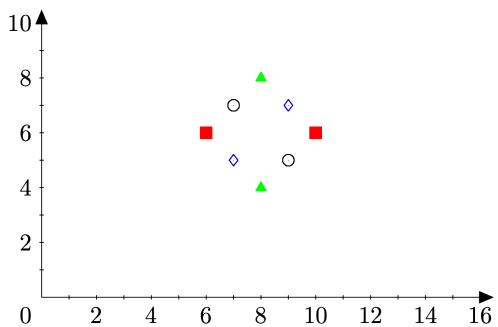
$$\lambda_i(x) := \sup\{m \geq 0 \mid \beta^{x-m,x+m} \geq i\}$$

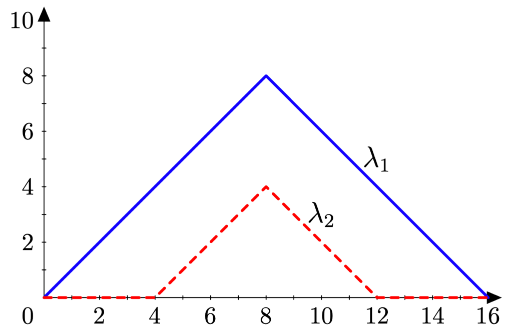
where $\beta^{p,q} := \dim(\mathrm{Im}(\mu^{p,q} : M_p \to M_q))$

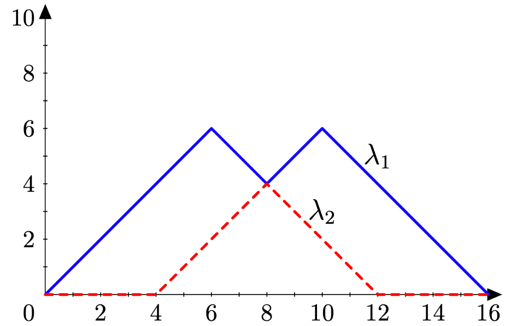
Visualizing Persistence

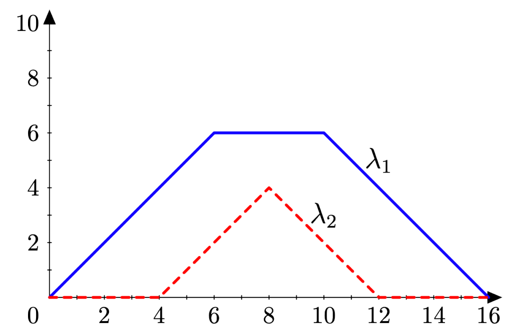
Persistence Landscapes: *Mean of persistence diagrams is **not unique**, but …*

*Mean of persistence landscapes is **well-defined***









Images from [Bubenik 2015]

Bibliography

Some References:

- ***Persistent Homology:***

  - U. Fugacci, S. Scaramuccia, F. Iuricich, L. De Floriani. ***Persistent homology: a step-by-step introduction for newcomers***. Eurographics Italian Chapter Conference, pages 1-10, 2016.

Persistence & Stability

Stability of Persistence

In order to be adopted in real applicative domains, it is crucial that

persistent homology is not affected by noisy data and small perturbations

Stability Result:

By defining ***distances**** *for both domains,*

Similar Data → ***Similar Persistent Homology***

**The term "distance" is intended in a broad sense, including pseudo-metrics and dissimilarity measures*

Stability of Persistence

Distances:

- ***For the Data in Input:***
  - ***Natural pseudo-distance** of shapes*
  - *L_∞-**distance** of filtering functions*
  - ***Gromov-Hausdorff distance** of metric spaces/point clouds*
- ***For the Retrieved Persistent Homology Information:***
  - ***Interleaving distance** of persistence modules*
  - ***Bottleneck** (a.k.a. Matching) **distance** of persistence diagrams*
  - ***Hausdorff distance** of persistence diagrams*
  - *Wasserstein distances of persistence diagrams*

Stability of Persistence

Distances for Input Data:

Let (X, f) be a *pair* such that:

- X is a ***(triangulable) topological space***
- f: X $\longrightarrow \mathbb{R}$ is a ***continuous function***

A pair (X, f) induces a *filtration*:

- $X^t := f^{-1}((-\infty, t])$

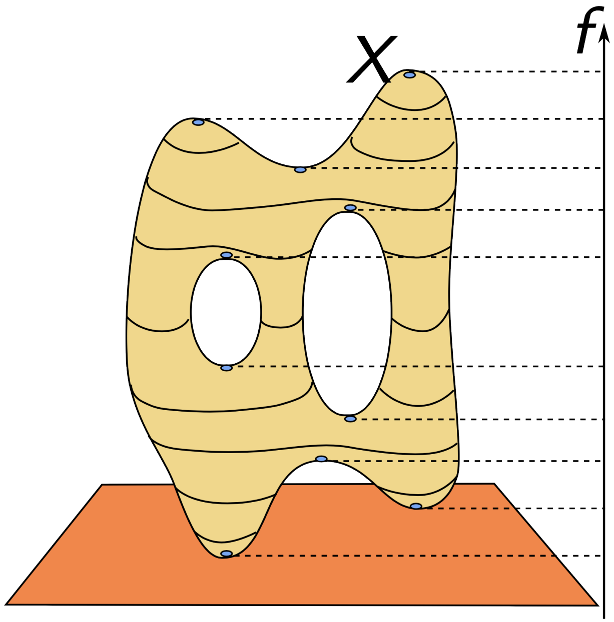


Image from [Ferri et al. 2015]

Definition:

*The function f is called **tame** if:*

- *f has a **finite number of homological critical values** (i.e. the "time" steps in which homology changes)*
- *For any $k \in \mathbb{N}$ and $t \in \mathbb{R}$, the **homology group $H_k(X^t, \mathbb{F})$ has finite dimension***

Stability of Persistence

Distances for Input Data:

Definition:

Given two pairs (X, f) and (Y, g), their ***natural pseudo-distance*** $\boldsymbol{d_N}$ *is defined as:*

$$d_N\Big((X,f),(Y,g)\Big) := \begin{cases} \inf_{h\in H(X,Y)}\{\max_{x\in X}\{|f(x)-g\circ h(x)|\}\} \\ +\infty \qquad\qquad\qquad\qquad\qquad \text{if } H(X,Y)=\emptyset \end{cases}$$

where ***H(X, Y)*** *is the set of all the* ***homeomorphisms between X and Y***

Stability of Persistence

Distances for Input Data:

Working with two functions f, g: X $\longrightarrow \mathbb{R}$ defined on the same topological space X, one can simply consider the ***L_∞-distance*** between f and g

$$\|f-g\|_\infty := \sup_{x\in X}\{|f(x)-g(x)|\}$$

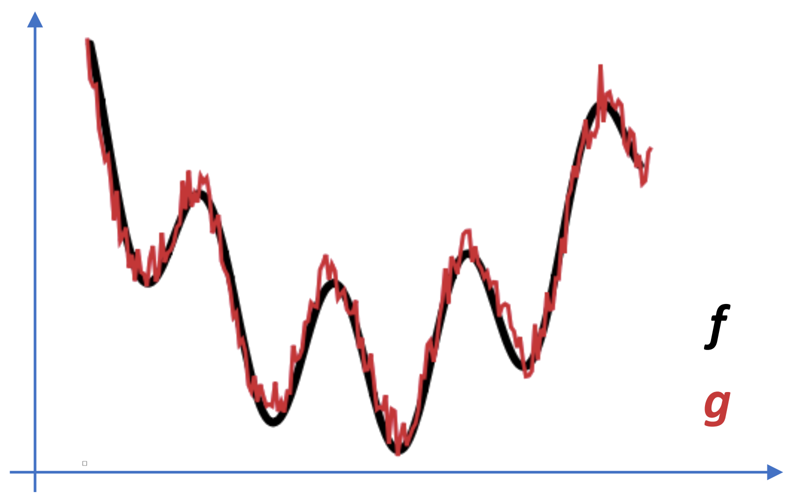


Image from [Rieck 2016]

Stability of Persistence

Distances for Input Data:

Given two ***finite metric spaces*** (X, d_X), (Y, d_Y) (e.g. two finite point clouds in $\mathbb{R}^n$),

Definitions:

*A **correspondence** $C: X \rightrightarrows Y$ from X to Y is a subset of $X \times Y$ such that the **canonical projections** $\pi_X: C \longrightarrow X$ and $\pi_Y: C \longrightarrow Y$ are both **surjective***

*The **distortion dis(C)** of a correspondence $C: X \rightrightarrows Y$ is defined as:*

$$dis(C) := \sup \left\{ |d_X(x,x') - d_Y(y,y')| : (x,y),(x',y') \in C \right\}$$

*The **Gromov-Hausdorff distance d_{GH}** between (X, d_X) and (Y, d_Y) is defined as:*

$$d_{GH}(X,Y) := \frac{1}{2} \inf\{dis(C) \,|\, C : X \rightrightarrows Y \text{ is a correspondence}\}$$

Stability of Persistence

Distances for Persistent Homology Information:

Two persistence modules M and N are called ***ε-interleaved*** with ε ≥ 0 if there exist f and g such that, for any p, q ∈ ℝ with p ≤ q, the following **diagrams commute**

$$
\begin{tikzcd}
 & M_p \arrow[dr, "f_p"] & \\
N_{p-\varepsilon} \arrow[ur, "g_{p-\varepsilon}"] \arrow[rr] & & N_{p+\varepsilon}
\end{tikzcd}
\qquad
\begin{tikzcd}
M_p \arrow[rr] \arrow[dr, "f_p"'] & & M_q \arrow[dr, "f_q"] & \\
 & N_{p+\varepsilon} \arrow[rr] & & N_{q+\varepsilon}
\end{tikzcd}
$$

$$
\begin{tikzcd}
M_{p-\varepsilon} \arrow[rr] \arrow[dr, "f_{p-\varepsilon}"'] & & M_{p+\varepsilon} \\
 & N_p \arrow[ur, "g_p"'] &
\end{tikzcd}
\qquad
\begin{tikzcd}
 & M_{p+\varepsilon} \arrow[rr] & & M_{q+\varepsilon} \\
N_p \arrow[ur, "g_p"] \arrow[rr] & & N_q \arrow[ur, "g_q"'] &
\end{tikzcd}
$$

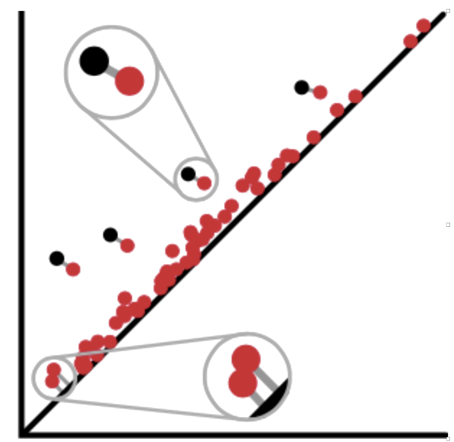
Definition:

*Given two persistence modules M and N, their **interleaving distance d_I** is defined as:*

$$d_I(M, N) := \inf\{\varepsilon \geq 0 \,|\, \text{M and N are } \varepsilon\text{-interleaved}\}$$

Stability of Persistence

Distances for Persistent Homology Information:



Definitions:

Given two persistence diagrams D_1 and D_2,

*their **bottleneck distance** d_B and **Hausdorff distance** d_H are defined as:*

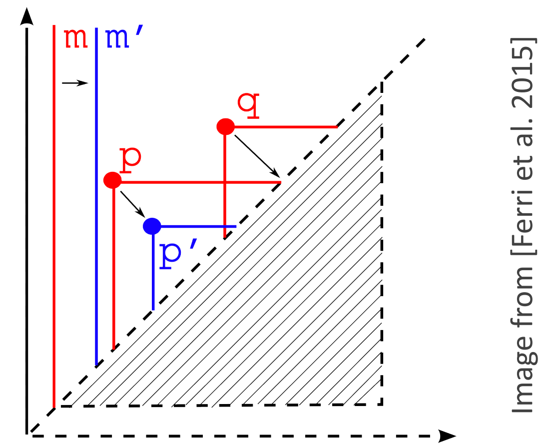
$$d_B(D_1, D_2) := \inf_{\gamma} \left\{ \sup_{x \in D_1} \{\|x - \gamma(x)\|_\infty\} \right\}$$

$$d_H(D_1, D_2) := \max \left\{ \sup_{x \in D_1} \left\{ \inf_{y \in D_2} \{\|x - y\|_\infty\} \right\}, \sup_{y \in D_2} \left\{ \inf_{x \in D_1} \{\|y - x\|_\infty\} \right\} \right\}$$

where γ ranges over all bijections from D_1 to D_2

Stability of Persistence

Distances for Persistent Homology Information:



Definitions:

Given two persistence diagrams D_1 and D_2,

*their **bottleneck distance d_B** and **Hausdorff distance d_H** are defined as:*

$$d_B(D_1, D_2) := \inf_{\gamma} \left\{ \sup_{x \in D_1} \{ \|x - \gamma(x)\|_\infty \} \right\}$$

$$d_H(D_1, D_2) := \max \left\{ \sup_{x \in D_1} \left\{ \inf_{y \in D_2} \{ \|x - y\|_\infty \} \right\}, \sup_{y \in D_2} \left\{ \inf_{x \in D_1} \{ \|y - x\|_\infty \} \right\} \right\}$$

where γ ranges over all bijections from D_1 to D_2

Stability of Persistence

Stability Results:

Given two pairs (X, f), (Y, g) of topological spaces and ***tame*** functions and $k \in \mathbb{N}$, let M, N be the induced k^{th} persistence modules and let D_1, D_2 be the corresponding persistence diagrams

- $d_H(D_1, D_2) \leq d_B(D_1, D_2)$
- $d_I(M, N) = d_B(D_1, D_2)$

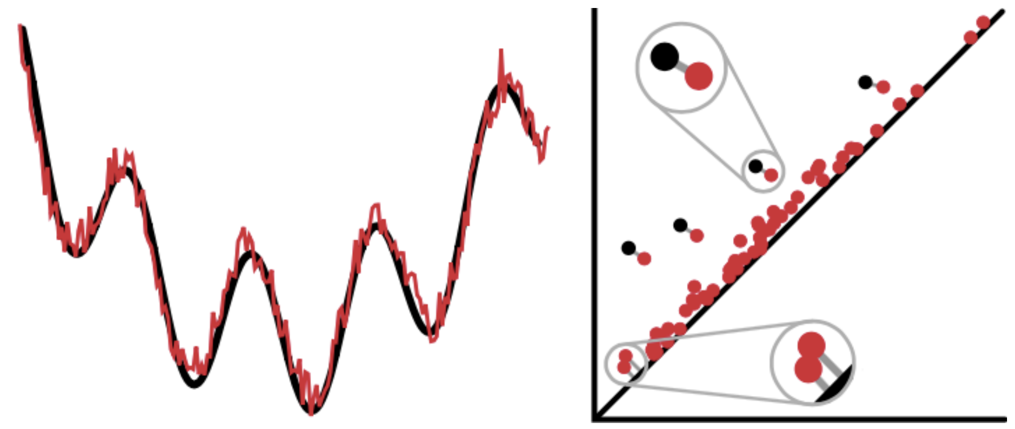
Theorem:

*Under the above hypothesis, the following **optimal lower bound** holds*

$$d_I(M, N) \leq d_N\Big((X, f), (Y, g)\Big)$$

Stability of Persistence

Stability Results:



Theorem:

*Given two **tame** continuous functions f, g: $X \longrightarrow \mathbb{R}$*

on a topological space X, $k \in \mathbb{N}$, and D_f, D_g the induced k^{th} persistence diagrams,

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

Images from [Rieck 2016]

Stability of Persistence

Stability Results:

Theorem:

Given two finite metric spaces (X, d_X), (Y, d_Y), $k \in \mathbb{N}$, and D_X, D_Y the k^{th} persistence diagrams of the ***filtrations of the Vietoris-Rips complexes generated by X and Y****,*

$$d_B(D_X, D_Y) \leq d_{GH}(X, Y)$$

Bibliography

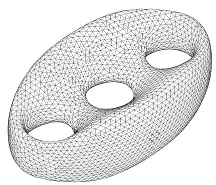
Some References:

- ***Stability Results:***
  - D. Cohen-Steiner, H. Edelsbrunner, J. Harer. ***Stability of persistence diagrams***. Discrete & Computational Geometry 37.1, pages 103-120, 2007.
  - F. Chazal, D. Cohen-Steiner, M. Glisse, L. J. Guibas, S. Y. Oudot. ***Proximity of persistence modules and their diagrams***. Proc. of the 35 annual symposium on Computational Geometry, pages 237-246, 2009.
  - F. Chazal, D. Cohen-Steiner, L. J. Guibas, F. Mémoli, S. Y. Oudot. ***Gromov-Hausdorff stable signatures for shapes using persistence***. Computer Graphics Forum 28.5, pages 1393-1403, 2009.

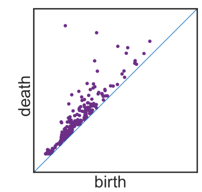
Computing Persistence

Persistent Homology Computation

Topological Data Analysis allows for assigning to (almost) ***any dataset*** a collection of features representing a ***topological summary*** of the input data



Shape



Features

Goal:

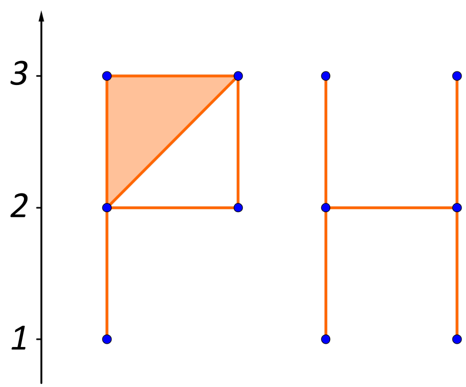
- ***How to efficiently compute (persistent) homology?***
- ***How to compactly encode simplicial complexes of high dimension and large size?***

Persistent Homology Computation

Standard Algorithm:

[Zomorodian & Carlsson 2005]

From:



To:

H_0 $[1, 2]$ $[1, \infty)$ $[1, \infty)$

H_1 $[3, \infty)$

$i\backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1			1											
3										1		1											
4								1			1						1	1					
5											1												
6									1											1			
7										1											1		
8																							
9																							
10																							
11																							
12																							
13																	1						
14																		1					
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	3					13	14		15	16		22

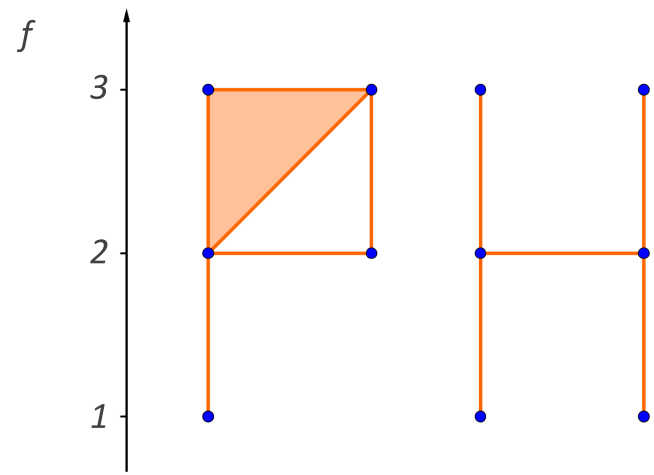
Compute a ***reduced boundary matrix*** for $\{ K^p \}_p$ from which easily read the persistence pairs

Persistent Homology Computation

Given a filtered simplicial complex, let us consider its *filtering function* ***f***:

$$f(\sigma) := \min \{ p \mid \sigma \in K^p \}$$

Conversely, $K^p := \{ \sigma \in K \mid f(\sigma) \leq p \}$



Total Ordering on $\{ K^p \}_p$:

A sequence $\sigma_1, \sigma_2, \ldots, \sigma_n$ of the simplices of K such that:

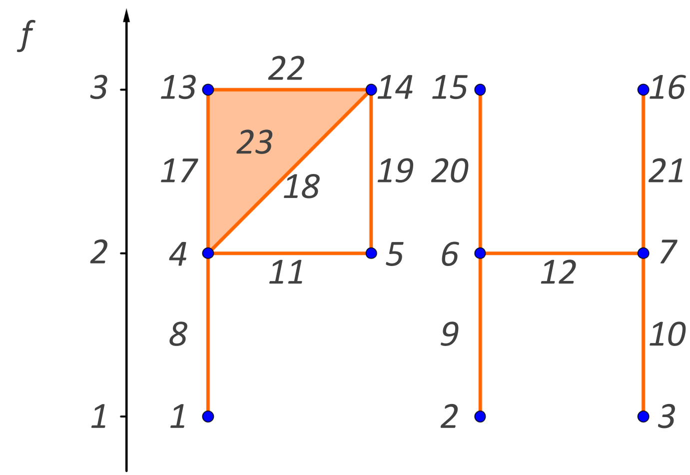
- *if $f(\sigma_i) < f(\sigma_j)$, then $i < j$*
- *if σ_i is a proper face of σ_j, then $i < j$*

Persistent Homology Computation

Given a filtered simplicial complex, let us consider its *filtering function* f:

$$f(\sigma) := \min \{ p \mid \sigma \in K^p \}$$

Conversely, $K^p := \{ \sigma \in K \mid f(\sigma) \leq p \}$



A Possible Choice:

Set $\sigma < \sigma'$ if:

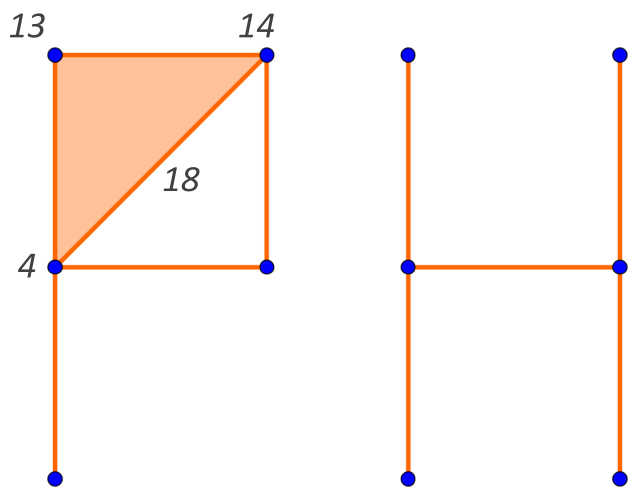
- $f(\sigma) < f(\sigma')$
- $f(\sigma) = f(\sigma')$ and $\dim(\sigma) < \dim(\sigma')$
- $f(\sigma) = f(\sigma')$, $\dim(\sigma) = \dim(\sigma')$, *and σ precedes σ' w.r.t. the **lexicographic order** of their vertices*

Persistent Homology Computation

Boundary Matrix:

A square matrix ***D*** of size *n* x *n* defined by

$$D_{i,j} := \begin{cases} 1 & \text{if } \sigma_i \text{ is a face of } \sigma_j \text{ s.t. } \dim(\sigma_i) = \dim(\sigma_j) - 1 \\ 0 & \text{otherwise} \end{cases}$$



E.g.

- $D_{4,18} = 1$
- $D_{14,18} = 1$
- $D_{13,18} = 0$

Persistent Homology Computation

Reduced Matrix:

Given a non-null column *j* of a boundary matrix *D*,

$$low(j) := \max \{ i \mid D_{i,j} \neq 0 \}$$

A matrix ***R*** *is called* ***reduced*** *if, for each pair of non-null columns j_1, j_2,*

$$low(j_1) \neq low(j_2)$$

Equivalently, if low function is ***injective*** on its domain of definition

Persistent Homology Computation

$i \backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1														
3										1													
4								1			1						1	1					
5											1								1				
6									1			1								1			
7										1		1									1		
8																							
9																							
10																							
11																							
12																							
13																	1					1	
14																		1	1			1	
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	7					13	14	14	15	16	14	22

low(10) = 7 = low(12) → *D* is **not** reduced

Persistent Homology Computation

Reduction Algorithm:

```
Matrix R = D
for j = 1, ..., n do
  while ∃ j'' < j with low(j') = low(j) do
    R.column(j) = R.column(j) + R.column(j')
  endwhile
endfor
return R
```

Time Complexity:

At most n^2 column additions ⇒ *$O(n^3)$ in the worst case*

$i\backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1														
3										1													
4								1			1						1	1					
5											1								1				
6									1			1								1			
7										1		1									1		
8																							
9																							
10																							
11																							
12																							
13																	1					1	
14																		1	1			1	
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	7					13	14	14	15	16	14	22

Initialize ***R*** to ***D*****,** where

D is the ***boundary matrix*** of *K*

expressed according with a ***total ordering*** of its simplices *j < 12*

$i\backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1														
3										1													
4								1			1						1	1					
5											1								1				
6									1			1								1			
7										1		1									1		
8																							
9																							
10																							
11																							
12																							
13																	1					1	
14																		1	1			1	
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	7					13	14	14	15	16	14	22

For each ***j < 12***,

there is **no *j' < j*** such that

low(j') = low(j)

So, **increase *j* by *1***

j' j

$i\backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1														
3										1													
4								1			1						1	1					
5											1								1				
6									1			1								1			
7										1		1									1		
8																							
9																							
10																							
11																							
12																							
13																	1					1	
14																		1	1			1	
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	7					13	14	14	15	16	14	22

For ***j* = 12**, ***low*(12) = 7**

***column j'*=10** is such that ***low*(*j'*) = *low*(*j*) = 7**

So, set

column 12* := *column 12* + *column 10

j

$i \backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1														
3										1		1											
4								1			1						1	1					
5											1								1				
6									1			1								1			
7										1											1		
8																							
9																							
10																							
11																							
12																							
13																	1					1	
14																		1	1			1	
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	6					13	14	14	15	16	14	22

For ***j* = 12**, ***low*(12) = 7**

column j'=10 is such that ***low(j') = low(j) = 7***

So, set

column 12 **:=** ***column 12*** **+** ***column 10*** $\longrightarrow$ ***low*(12) = 6**

									j'			j											
$i\backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1														
3										1		1											
4								1			1						1	1					
5											1								1				
6									1			1								1			
7										1											1		
8																							
9																							
10																							
11																							
12																							
13																	1					1	
14																		1	1			1	
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	6					13	14	14	15	16	14	22

For ***j* = 12**, ***low*(12) = 6**

column j' = 9 is such that ***low(j') = low(j) = 6***

So, set

column 12* := *column 12* + *column 9

j

$i\backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1			1											
3										1		1											
4								1			1						1	1					
5											1								1				
6									1											1			
7										1											1		
8																							
9																							
10																							
11																							
12																							
13																	1					1	
14																		1	1			1	
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	3					13	14	14	15	16	14	22

For ***j = 12***, ***low(12) = 6***

column j' = 9 is such that ***low(j') = low(j) = 6***

So, set

column 12 := ***column 12*** + ***column 9*** ⟶ ***low(12) = 3***

j

$i\backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1			1											
3										1		1											
4								1			1						1	1					
5											1								1				
6									1											1			
7										1											1		
8																							
9																							
10																							
11																							
12																							
13																	1					1	
14																		1	1			1	
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							

low								4	6	7	5	3					13	14	14	15	16	14	22

For each ***j = 12***,

there is **no *j' < j*** such that

low(j') = low(j) = 3

So, **increase *j* by *1***

12 < j < 19

$i\backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1			1											
3										1		1											
4								1			1						1	1					
5											1								1				
6									1											1			
7										1											1		
8																							
9																							
10																							
11																							
12																							
13																	1					1	
14																		1	1			1	
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	3					13	14	14	15	16	14	22

For each ***12 < j < 19***,

there is **no *j' < j*** such that

low(j') = low(j)

So, **increase *j* by *1***

																		j'	j				
$i \backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1			1											
3										1		1											
4								1			1						1	1					
5											1								1				
6									1											1			
7										1											1		
8																							
9																							
10																							
11																							
12																							
13																	1					1	
14																		1	1			1	
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	3					13	14	14	15	16	14	22

For ***j* = 19, *low*(19) = 14**

column j'= 18 is such that ***low(j') = low(j) = 14***

So, set

column 19* := *column 19* + *column 18

j

$i \backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1			1											
3										1		1											
4								1			1						1	1	1				
5											1								1				
6									1											1			
7										1											1		
8																							
9																							
10																							
11																							
12																							
13																	1					1	
14																		1				1	
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	3					13	14	5	15	16	14	22

For ***j* = 19, *low*(19) = 14**

column j'= 18 is such that ***low(j') = low(j) = 14***

So, set

column 19* := *column 19* + *column 18 $\longrightarrow$ ***low(19) = 5***

										j'								*j*				
$i \backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1			1											
3										1		1											
4								1			1						1	1	1				
5											1								1				
6									1											1			
7										1											1		
8																							
9																							
10																							
11																							
12																							
13																	1					1	
14																		1				1	
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	3					13	14	5	15	16	14	22

For ***j* = 19, *low*(19) = 5**

column j'= 11 is such that ***low(j') = low(j) = 5***

So, set

column 19* := *column 19* + *column 11

j

$i\backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1			1											
3										1		1											
4								1			1						1	1					
5											1												
6									1											1			
7										1											1		
8																							
9																							
10																							
11																							
12																							
13																	1					1	
14																		1				1	
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	3					13	14		15	16	14	22

For ***j* = 19, *low*(19) = 5**

column j'= 11 is such that ***low(j') = low(j) = 5***

So, set

column 19 := column 19 + column 11 $\longrightarrow$ ***low(19) undefined***

j

$i\backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1			1											
3										1		1											
4								1			1						1	1					
5											1												
6									1											1			
7										1											1		
8																							
9																							
10																							
11																							
12																							
13																	1					1	
14																		1				1	
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	3					13	14		15	16	14	22

For each ***j = 19***,

there is **no *j' < j*** such that

low(j') = low(j)

So, **increase *j* by *1***

19 < j < 22

$i\backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1			1											
3										1		1											
4								1			1						1	1					
5											1												
6									1											1			
7										1											1		
8																							
9																							
10																							
11																							
12																							
13																	1					1	
14																		1				1	
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	3					13	14		15	16	14	22

For each ***19 < j < 22***,

there is **no *j' < j*** such that

low(j') = low(j)

So, **increase *j* by *1***

																	j'				*j*	
$i\backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1			1											
3										1		1											
4								1			1						1	1					
5											1												
6									1											1			
7										1											1		
8																							
9																							
10																							
11																							
12																							
13																	1					1	
14																		1				1	
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	3					13	14		15	16	14	22

For ***j* = 22**, ***low*(22) = 14**

column j'= 18 is such that ***low(j') = low(j) = 14***

So, set

column 22* := *column 22* + *column 18

j

$i\backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1			1											
3										1		1											
4								1			1						1	1				1	
5											1												
6									1											1			
7										1											1		
8																							
9																							
10																							
11																							
12																							
13																	1					1	
14																		1					
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	3					13	14		15	16	13	22

For ***j* = 22**, ***low*(22) = 14**

column j'= 18 is such that ***low(j') = low(j) = 14***

So, set

column 22** := **column 22** + **column 18 $\longrightarrow$ ***low(22) = 13***

																	j’					*j*	
$i\backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1			1											
3										1		1											
4								1			1						1	1				1	
5											1												
6									1											1			
7										1											1		
8																							
9																							
10																							
11																							
12																							
13																	1					1	
14																		1					
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	3					13	14		15	16	13	22

For ***j* = 22**, ***low*(22) = 13**

column j’= 17 is such that ***low(j’) = low(j) = 13***

So, set

column 22 := column 22 + column 17

$i\backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1			1											
3										1		1											
4								1			1						1	1					
5											1												
6									1											1			
7										1											1		
8																							
9																							
10																							
11																							
12																							
13																	1						
14																		1					
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	3					13	14		15	16		22

For ***j* = 22**, ***low*(22) = 13**

column j'= 17 is such that ***low(j') = low(j) = 13***

So, set

column 22 := column 22 + column 17 $\longrightarrow$ ***low(22) undefined***

j

$i\backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1			1											
3										1		1											
4								1			1						1	1					
5											1												
6									1											1			
7										1											1		
8																							
9																							
10																							
11																							
12																							
13																	1						
14																		1					
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	3					13	14		15	16		22

For each ***j* = *22***,

there is **no *j' < j*** such that

low(j') = low(j)

So, **increase *j* by *1***

j

$i \backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1			1											
3										1		1											
4								1			1						1	1					
5											1												
6									1											1			
7										1											1		
8																							
9																							
10																							
11																							
12																							
13																	1						
14																		1					
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							

low								4	6	7	5	3					13	14		15	16		22

For each ***j* = 23**,

there is **no *j'* < *j*** such that

low(j') = low(j) = 22

So, **matrix *R* is reduced**

$i\backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1			1											
3										1		1											
4								1			1						1	1					
5											1												
6									1											1			
7										1											1		
8																							
9																							
10																							
11																							
12																							
13																	1						
14																		1					
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	3					13	14		15	16		22

The algorithm returns the above **reduced matrix *R***

Persistent Homology Computation

Retrieving Persistence Pairs:

- For each $i = 1, \ldots, n$,

  if there exists j such that $low(j) = i$ ⇨ **$[i, j]$ is a pair for R**

- Once every i has been parsed,

  if i is an **unpaired** value ⇨ **$[i, \infty)$ is a pair for R**

From pairs of R to the **"actual" persistence pairs of $\{K^p\}_p$**:

$[i, j]$ corresponds to $[f(\sigma_i), f(\sigma_j)]$

$[i, \infty)$ corresponds to $[f(\sigma_i), \infty)$

(homological degree = $dim(\sigma_i)$)

Persistent Homology Computation

H_0

[1, ∞)

[2, ∞)

[3, 12]

[4, 8]

[5, 11]

[6, 9]

[7, 10]

[13, 17]

[14, 18]

[15, 20]

[16, 21]

H_1

[19, ∞)

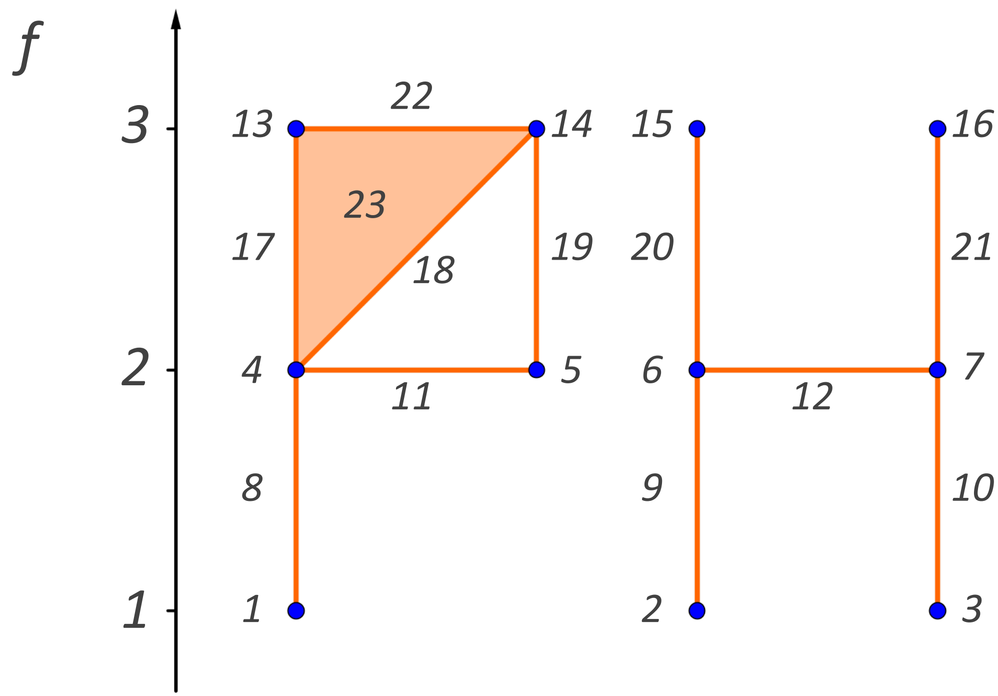
[22, 23]

$i \backslash j$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1								1															
2									1			1											
3										1		1											
4								1			1						1	1					
5											1												
6									1											1			
7										1											1		
8																							
9																							
10																							
11																							
12																							
13																	1						
14																		1					
15																				1			
16																					1		
17																							1
18																							1
19																							
20																							
21																							
22																							1
23																							
low								4	6	7	5	3					13	14		15	16		22

Persistent Homology Computation

H_0

[1, ∞)	**[1, ∞)**
[2, ∞)	**[1, ∞)**
[3, 12]	**[1, 2]**
[4, 8]	[2, 2]
[5, 11]	[2, 2]
[6, 9]	[2, 2]
[7, 10]	[2, 2]
[13, 17]	[3, 3]
[14, 18]	[3, 3]
[15, 20]	[3, 3]
[16, 21]	[3, 3]



H_1

[19, ∞)	**[3, ∞)**
[22, 23]	[3, 3]

Persistent Homology Computation

Standard algorithm to compute (persistent) homology [Zomorodian & Carlsson 2005]:

- Based on a ***matrix reduction***
- ***Linear complexity*** in practical cases
- ***Cubic complexity*** in the worst case

Several different strategies:

Direct approaches:

- ***Zigzag persistent homology*** [Milosavljević et al. ’05]
- ***Computation with a twist*** [Chen, Kerber ’11]
- ***Dual algorithm*** [De Silvia et al. ’11]
- ***Output-sensitive algorithm*** [Chen, Kerber ’13]
- ***Multi-field algorithm*** [Boissonnat, Maria ’14]
- ***Annotation-based methods*** [Boissonnat et al. ’13; Dey et al. ’14]

Distributed approaches:

- ***Spectral sequences*** [Edelsbrunner, Harer ’08; Lipsky et al. ’11]
- ***Constructive Mayer-Vietoris*** [Boltcheva et al. ’11]
- ***Multicore coreductions*** [Murty et al. ’13]
- ***Multicore homology*** [Lewis, Zomorodian ’14]
- ***Persistent homology in chunks*** [Bauer et al. ‘14a]
- ***Distributed persistent computation*** [Bauer et al. ‘14b]

Coarsening approaches:

- ***Topological operators and simplifications*** [Mrozek, Wanner ’10; Dłotko, Wagner ’14]
- ***Morse-based approaches*** [Robins et al. ’11; Harker et al. ’14; Fugacci et al. ’14]

Persistent Homology Computation

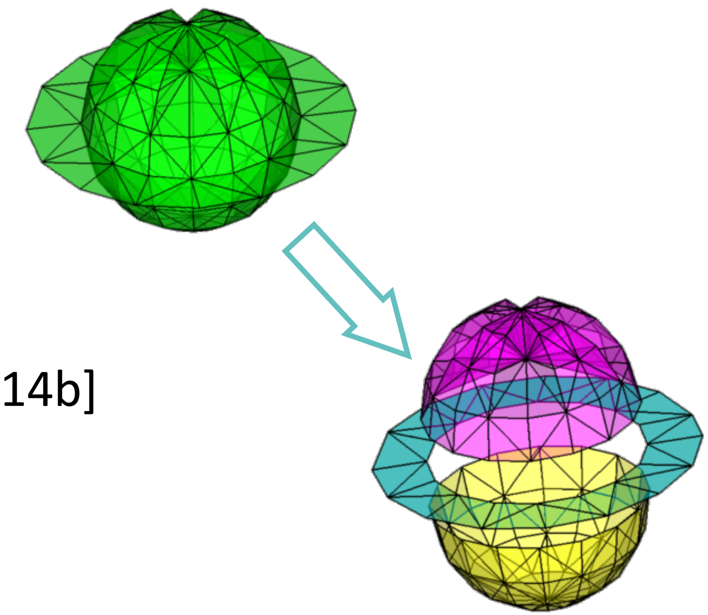
Direct Approaches:

- ***Zigzag persistent homology*** [Milosavljević et al. ’05]
- ***Computation with a twist*** [Chen, Kerber ’11]
- ***Dual algorithm*** [De Silvia et al. ’11]
- ***Output-sensitive algorithm*** [Chen, Kerber ’13]
- ***Multi-field algorithm*** [Boissonnat, Maria ’14]
- ***Annotation-based methods*** [Boissonnat et al. ’13; Dey et al. ’14]

Persistent Homology Computation

Distributed Approaches:

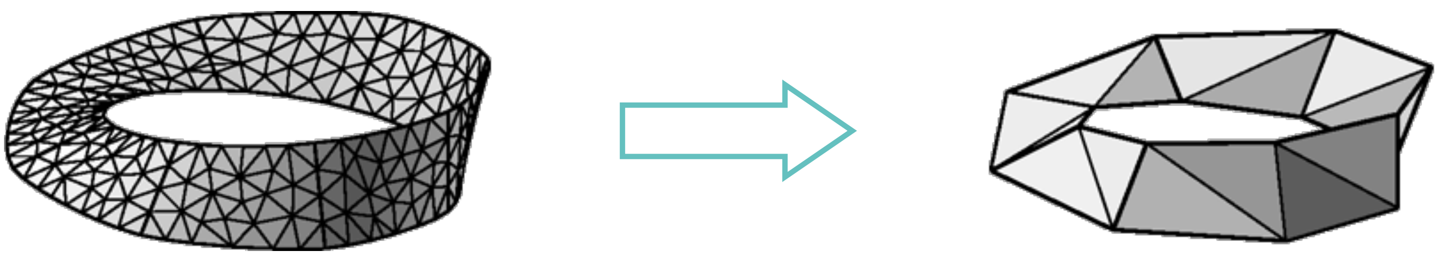
- *Spectral sequences* [Edelsbrunner, Harer ’08; Lipsky et al. ’11]
- *Constructive Mayer-Vietoris* [Boltcheva et al. ’11]
- *Multicore coreductions* [Murty et al. ’13]
- *Multicore homology* [Lewis, Zomorodian ’14]
- *Persistent homology in chunks* [Bauer et al. ‘14a]
- *Distributed persistent computation* [Bauer et al. ‘14b]



Persistent Homology Computation

Coarsening Approaches:

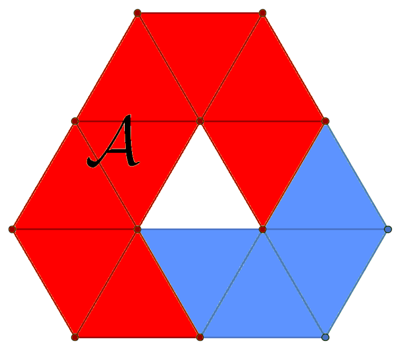
- ***Topological operators and simplifications*** [Dłotko, Wagner '14]
  - Acyclic subcomplexes [Mrozek et al. '08]
  - Reductions and coreductions [Mrozek et al. '10]
  - Edge contractions [Attali et al. '11]
- ***Morse-based approaches*** [Robins et al. '11; Harker et al. '14; Fugacci et al. '14]



Persistent Homology Computation

Coarsening Approaches:

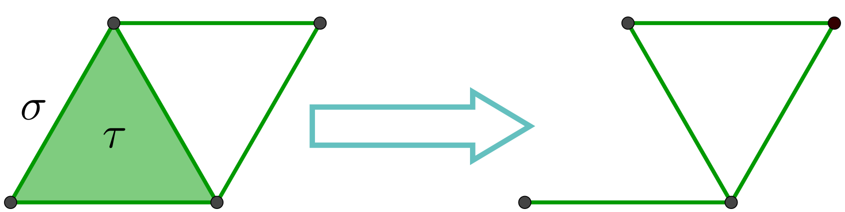
- ***Topological operators and simplifications*** [Dłotko, Wagner '14]
  - *Acyclic subcomplexes* [Mrozek et al. '08]
  - Reductions and coreductions [Mrozek et al. '10]
  - Edge contractions [Attali et al. '11]
- ***Morse-based approaches*** [Robins et al. '11; Harker et al. '14; Fugacci et al. '14]

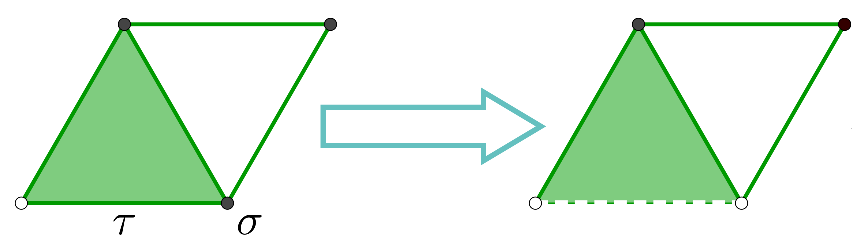


Persistent Homology Computation

Coarsening Approaches:

- ***Topological operators and simplifications*** [Dłotko, Wagner '14]
  - Acyclic subcomplexes [Mrozek et al. '08]
  - *Reductions and coreductions* [Mrozek et al. '10]
  - Edge contractions [Attali et al. '11]
- ***Morse-based approaches*** [Robins et al. '11; Harker et al. '14; Fugacci et al. '14]

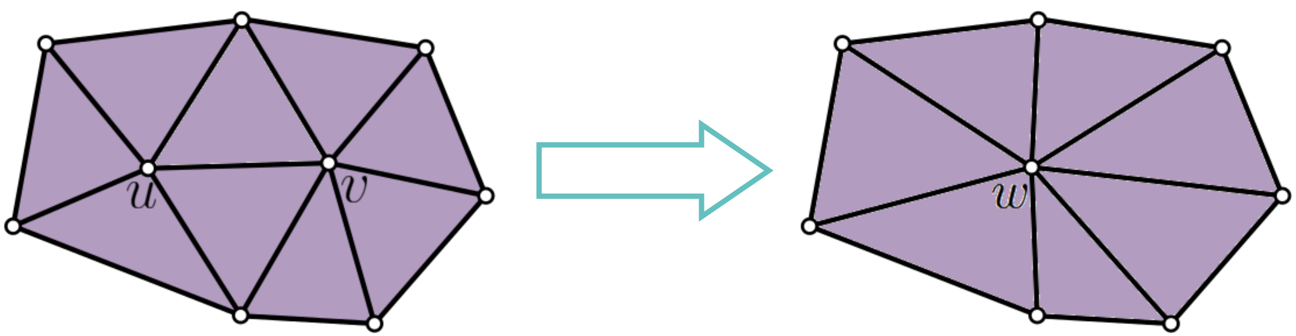




Persistent Homology Computation

Coarsening Approaches:

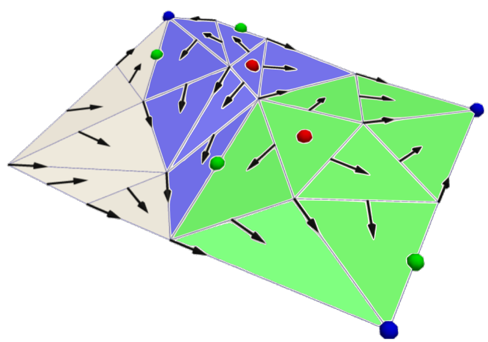
- ***Topological operators and simplifications*** [Dłotko, Wagner '14]
  - Acyclic subcomplexes [Mrozek et al. '08]
  - Reductions and coreductions [Mrozek et al. '10]
  - *Edge contractions* [Attali et al. '11]
- ***Morse-based approaches*** [Robins et al. '11; Harker et al. '14; Fugacci et al. '14]



Persistent Homology Computation

Coarsening Approaches:

- ***Topological operators and simplifications*** [Dłotko, Wagner ’14]
  - Acyclic subcomplexes [Mrozek et al. ‘08]
  - Reductions and coreductions [Mrozek et al. ‘10]
  - Edge contractions [Attali et al. ’11]
- ***Morse-based approaches*** [Robins et al. ’11; Harker et al. ’14; Fugacci et al. ’14]



Bibliography

Some References:

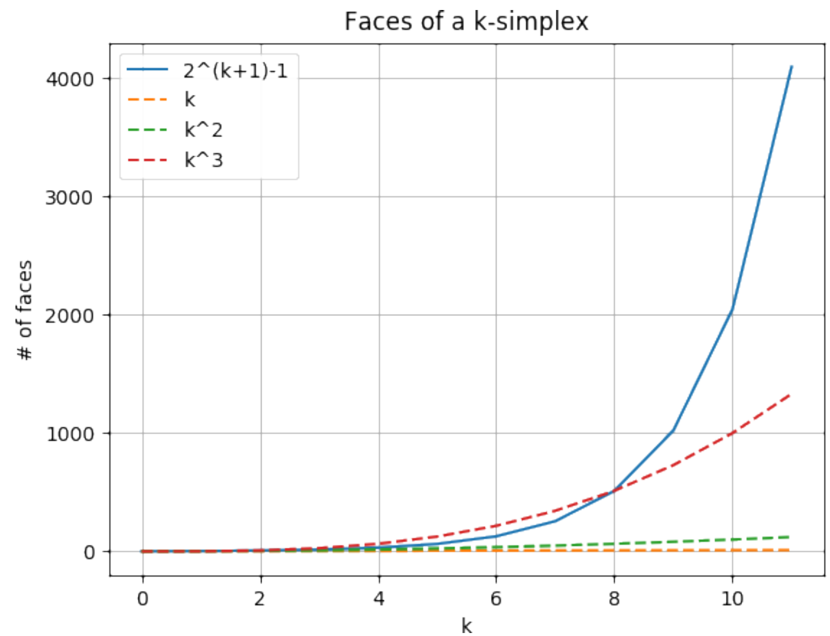
- ***Persistent Homology Computation:***
  - A. Zomorodian, G. Carlsson. ***Computing persistent homology.*** Discrete & Computational Geometry, 33.2, pages 249-274, 2005.
  - N. Otter, M.A. Porter, U. Tillmann, P. Grindrod, H.A. Harrington. ***A roadmap for the computation of persistent homology.*** EPJ Data Science, 6.1, 2017.

Data Structures

Encoding Simplicial Complexes

Issue:

It is enough to have a point cloud consisting of at least ***30 points*** *for having to deal with an associated filtered simplicial complex of more than a* ***billion*** *of simplices*



Solution:

Development of ***compact*** *and* ***efficient data structures*** *for encoding arbitrary simplicial complexes*

Encoding Simplicial Complexes

Outline:

- **Which info to be stored?**
- **Data Structures**
  - *Simplex-based representations*
  - *Top-based representations*
  - *Operator-driven representations*
- **Comparisons**
- **Issues and solutions in adopting top-based representations**

Out Of Scope:

- **Data structures for specific classes of complexes**
  - *E.g. manifold or complexes of low dimension*

Encoding Simplicial Complexes

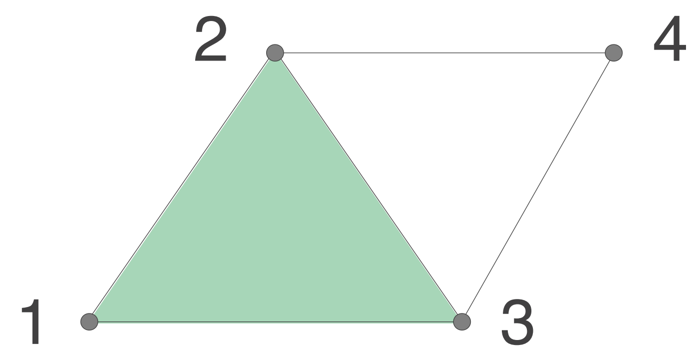
Data Structure:

The *entities* which a simplicial complex consists of are:

- its ***simplices***

$$K = K_0 \cup K_1 \cup \ldots \cup K_d$$

where K_i is the collection of the i-simplices of K



- the ***topological relations***

$$R_{i,j} \subseteq K_i \times K_j$$

between the simplices of K encoding the (co-)boundary of each simplex

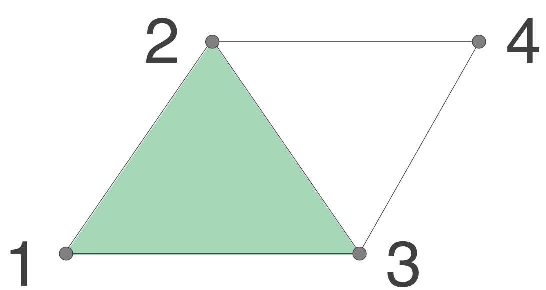
*A **data structure** for K has to explicitly **store** a portion of the above information and to (efficiently) **retrieve** the remaining part*

Encoding Simplicial Complexes

Topological Relations:

Given an i-simplex σ and a j-simplex τ of K,

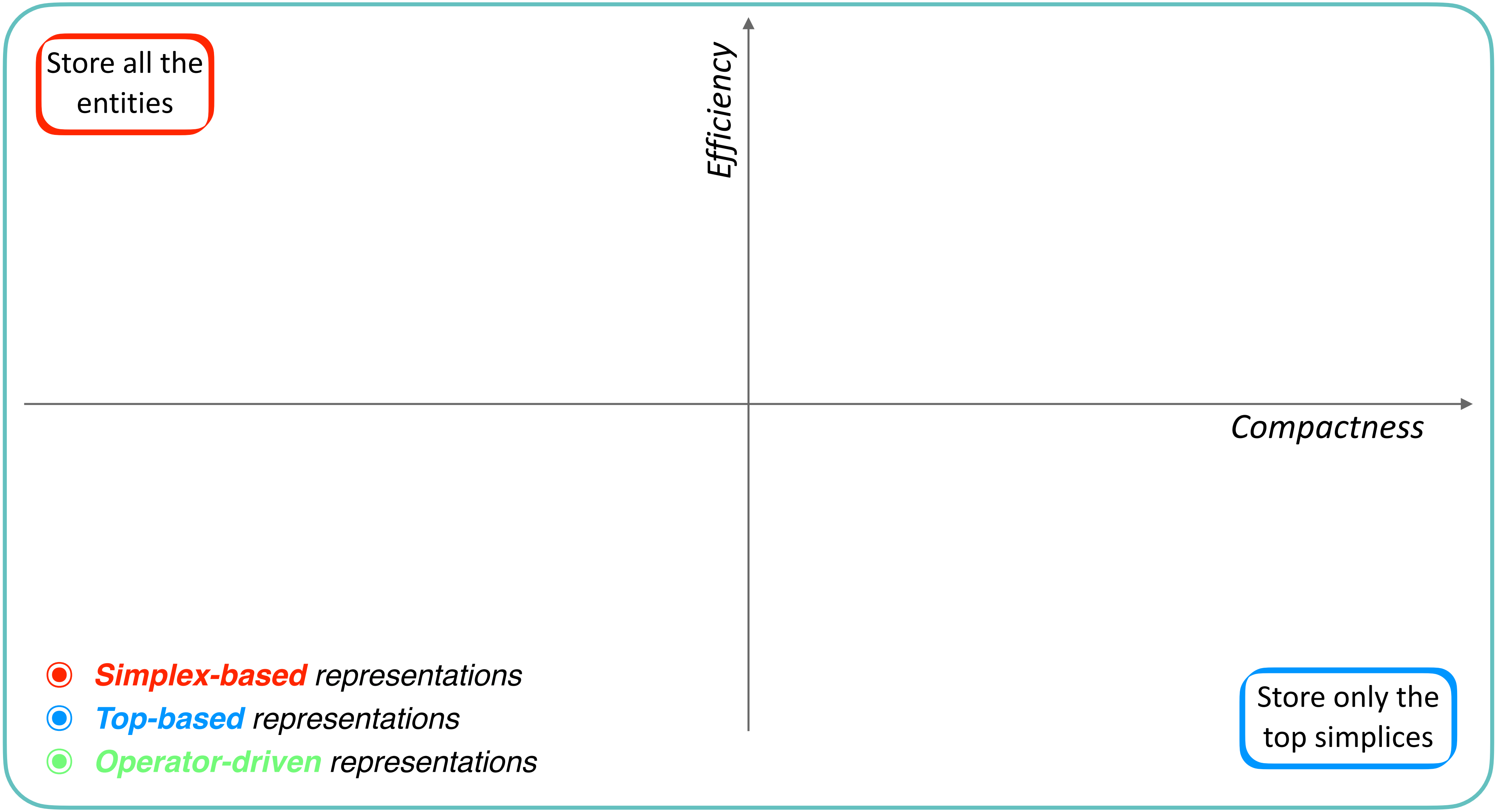
$$(\sigma, \tau) \in R_{i,j} \Longleftrightarrow \begin{cases} \sigma \subseteq \tau & \text{for } i < j \\ |\sigma \cap \tau| = i \ \text{(equivalently, } \sigma \cap \tau \in K_{i-1}\text{)} & \text{for } i = j \\ \tau \subseteq \sigma & \text{for } i > j \end{cases}$$



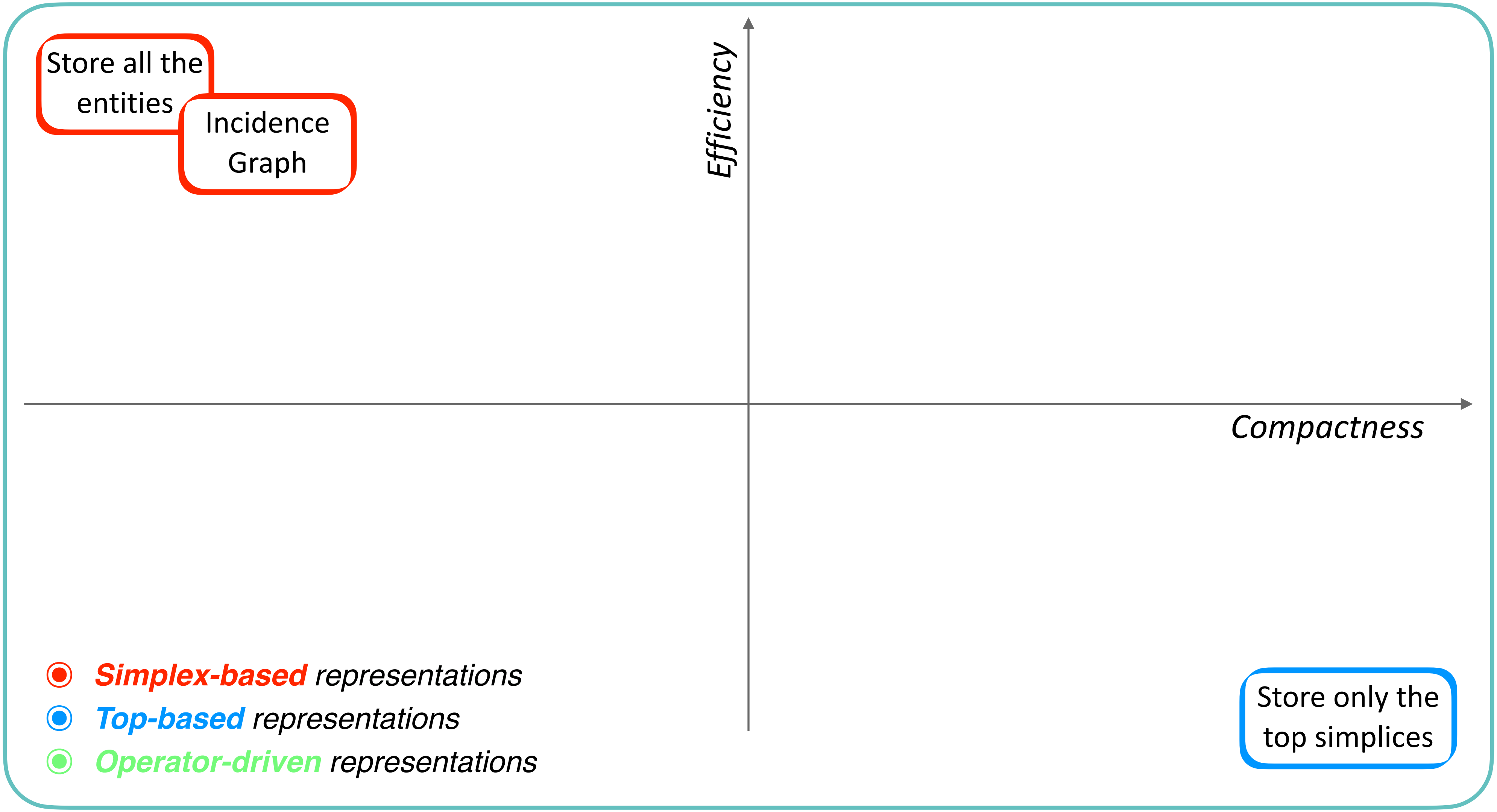
$(12, 123) \in R_{1,2}$ $(12, 24) \in R_{1,1}$ $(12, 1) \in R_{1,0}$

An i-simplex σ is called a ***top simplex*** *of K if there is no simplex τ of K such that $(\sigma, \tau) \in R_{i,i+1}$*

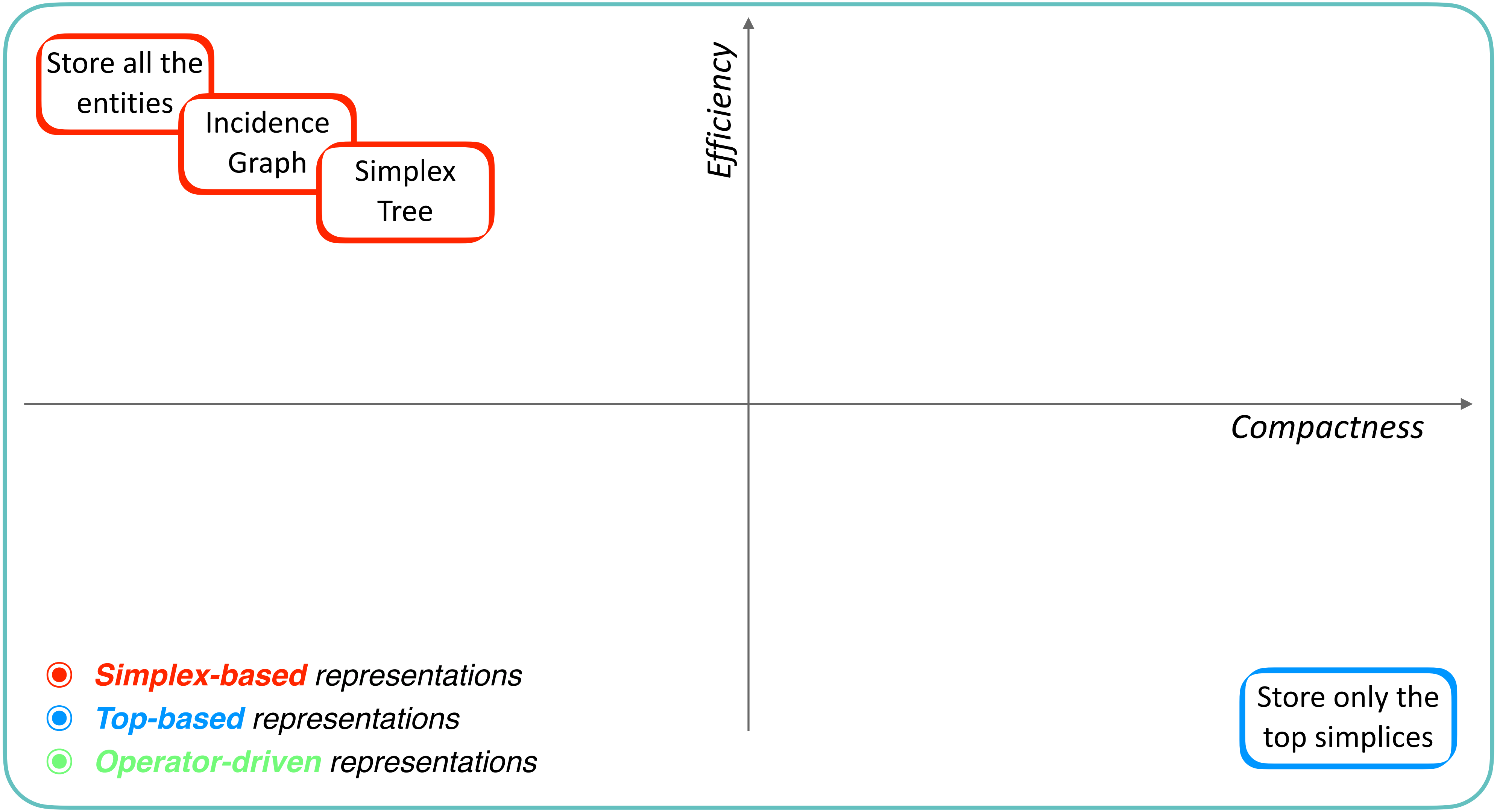
Encoding Simplicial Complexes



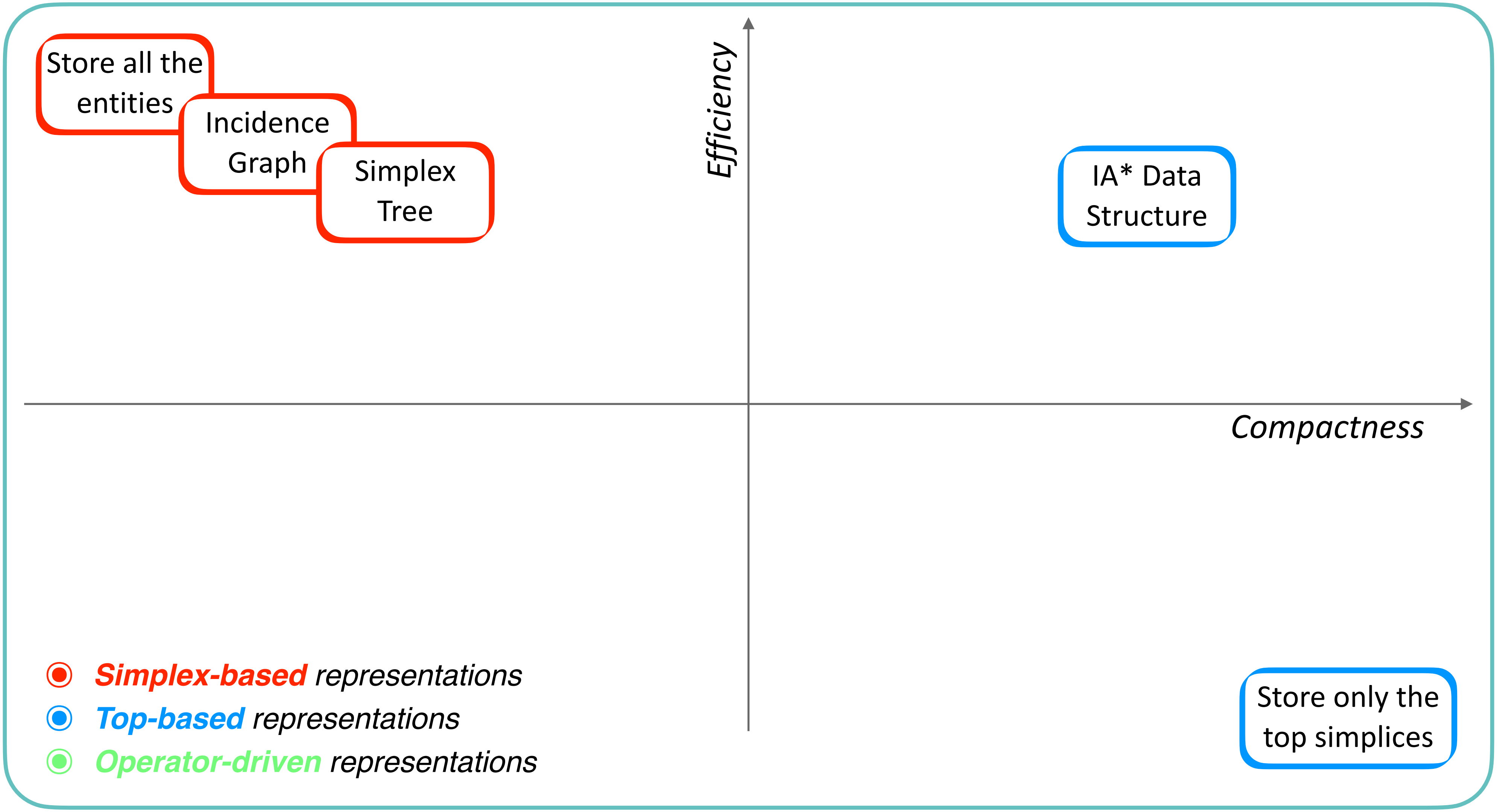
Encoding Simplicial Complexes



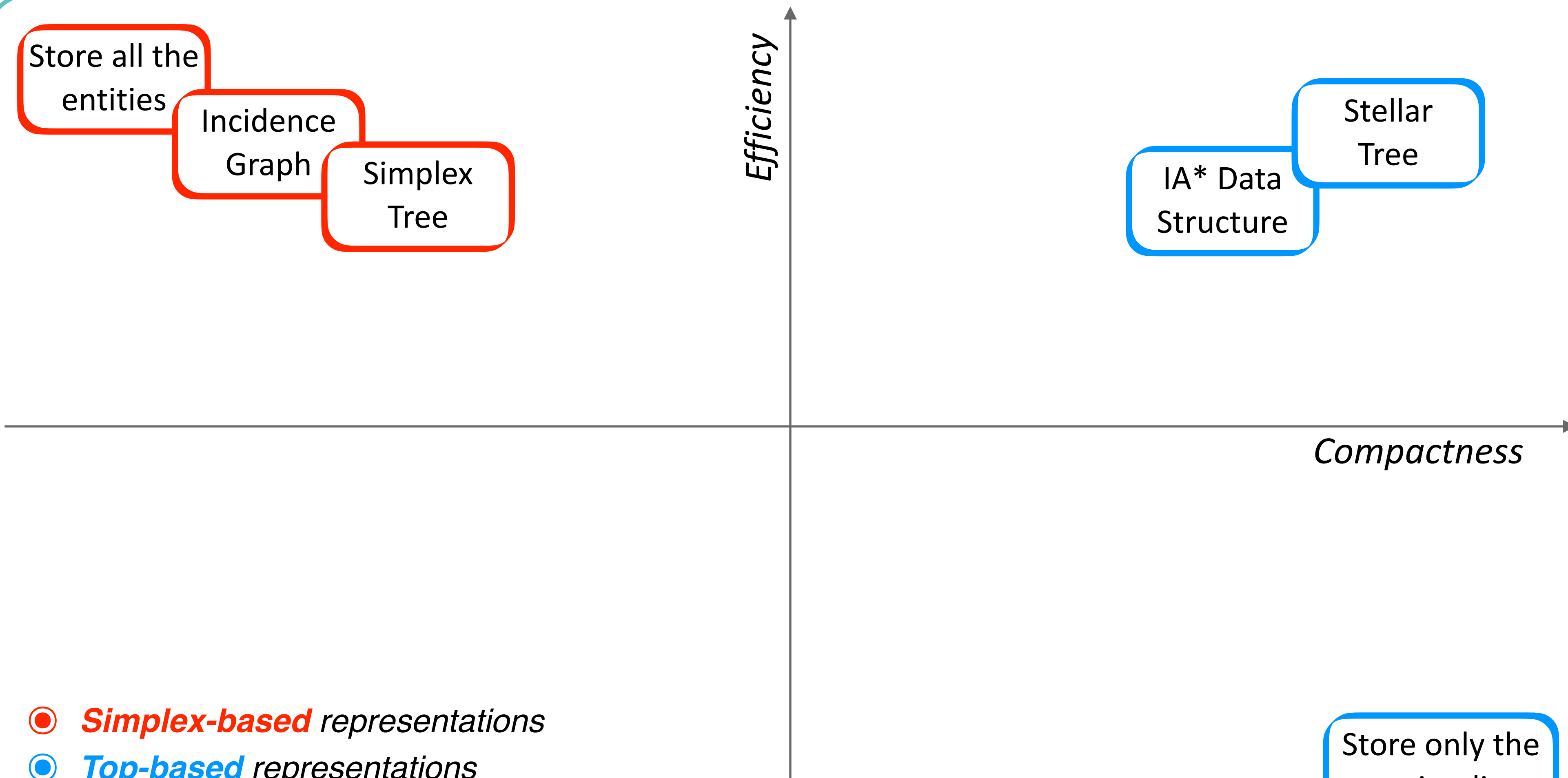
Encoding Simplicial Complexes



Encoding Simplicial Complexes

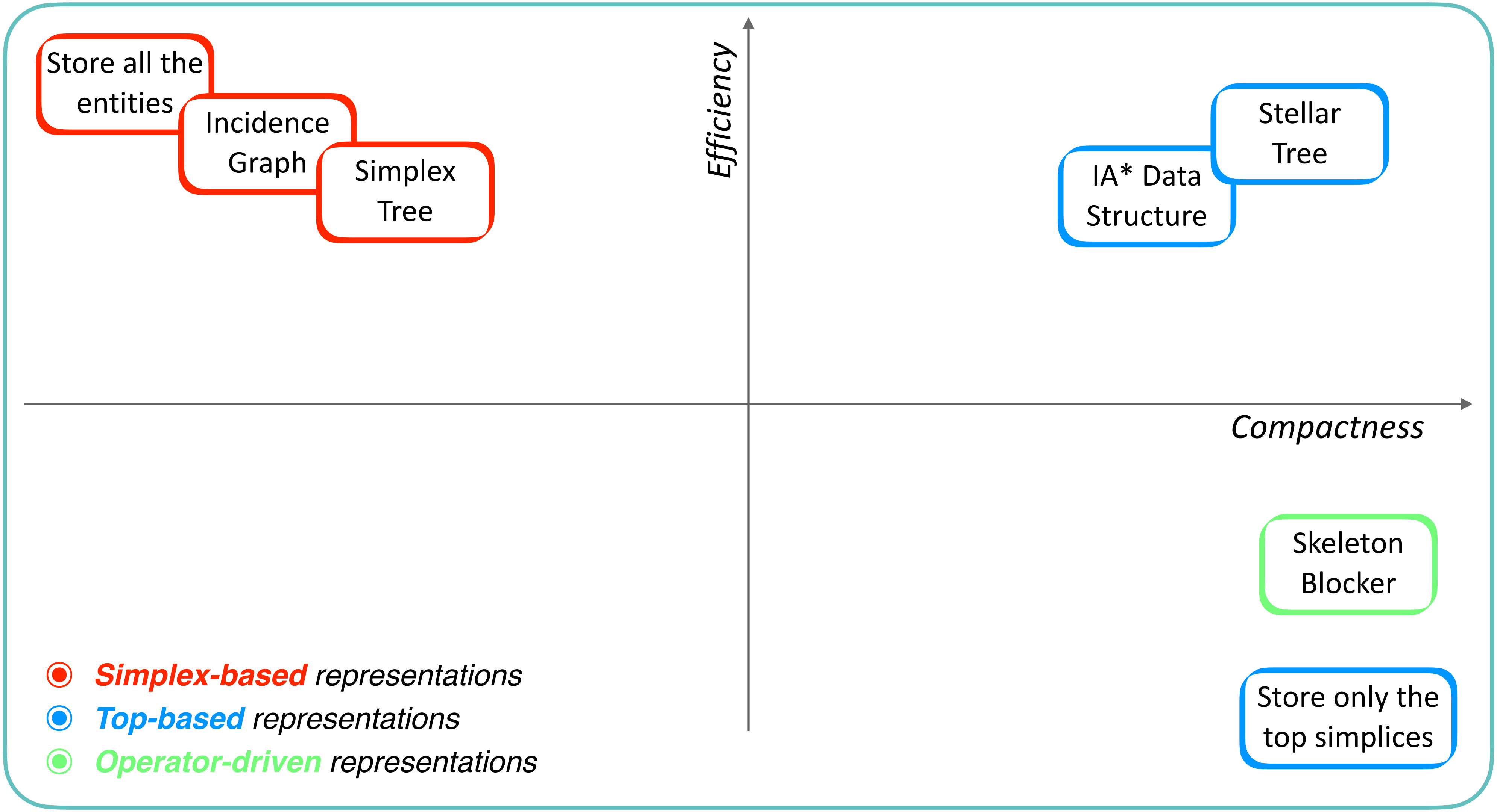


Encoding Simplicial Complexes



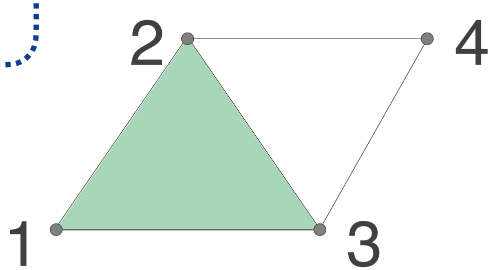
- ***Simplex-based*** *representations*
- ***Top-based*** *representations*
- ***Operator-driven*** *representations*

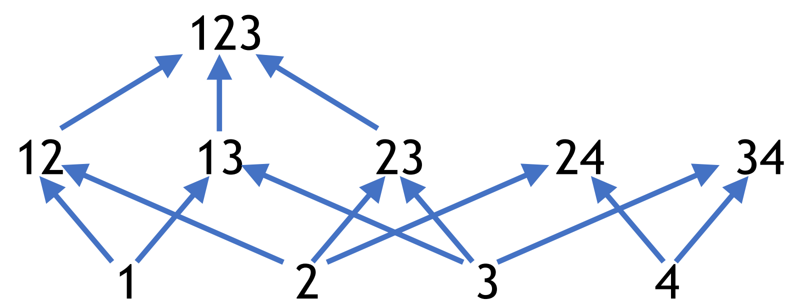
Encoding Simplicial Complexes



Simplex-based Representations

Incidence Graph:





The simplicial complex K is encoded via a ***directed graph*** ***G = (N, A)***:

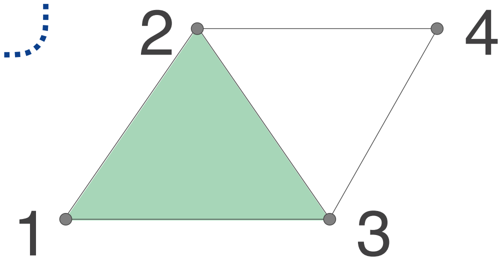
$\boldsymbol{N \longleftrightarrow K}$

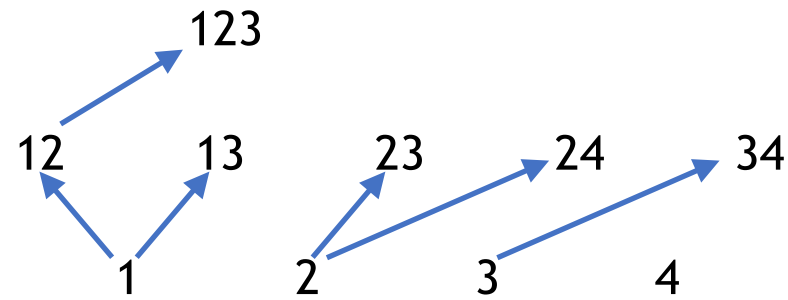
$\boldsymbol{(\sigma, \tau) \in A \longleftrightarrow (\sigma, \tau) \in R_{i,i+1}}$

- *All the relations between simplices can be immediately retrieved*
- *The representation* ***size exponentially increases*** *with the complex dimension*

Simplex-based Representations

Simplex Tree:





The simplicial complex K is encoded via a ***directed graph*** ***G = (N, A)***:

$$N \longleftrightarrow K$$

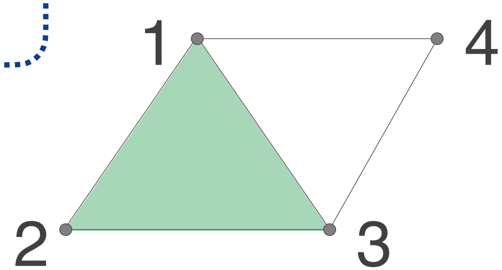
$$(\sigma, \tau) \in A \longleftrightarrow (\sigma, \tau) \in R_{i,i+1} \text{ and } I(\sigma) < I(\tau)$$

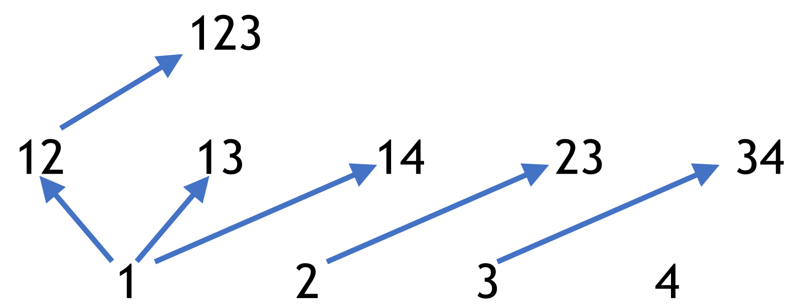
where $I(\sigma)$ denotes the ***maximum value*** taken by the vertices of σ w.r.t. a ***total order*** on K_0

*Graph is **not uniquely determined** but it depends on the chosen vertex order*

Simplex-based Representations

Simplex Tree:





The simplicial complex K is encoded via a ***directed graph G = (N, A)***:

$N \longleftrightarrow K$

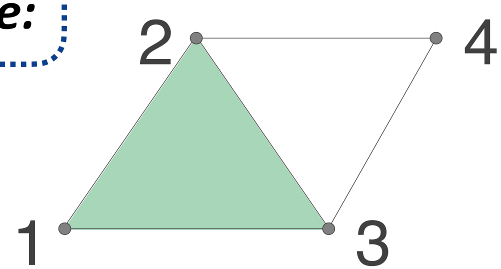
$(\sigma, \tau) \in A \longleftrightarrow (\sigma, \tau) \in R_{i,i+1}$ *and* $I(\sigma) < I(\tau)$

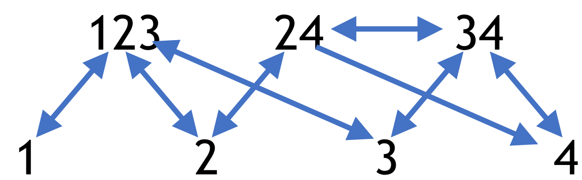
where $I(\sigma)$ denotes the ***maximum value*** taken by the vertices of σ w.r.t. a ***total order*** on K_0

*Graph is **not uniquely determined** but it depends on the chosen vertex order*

Top-based Representations

IA* Data Structure:





The simplicial complex K is encoded via a *directed graph* ***G = (N, A)***:

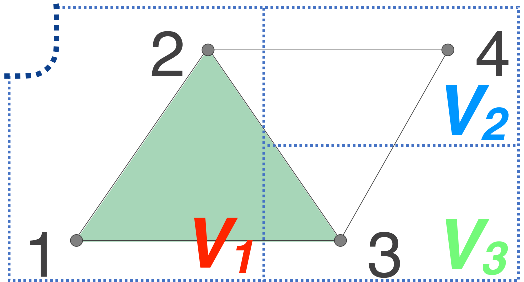
$$N \longleftrightarrow K_0 \cup K_{top}$$

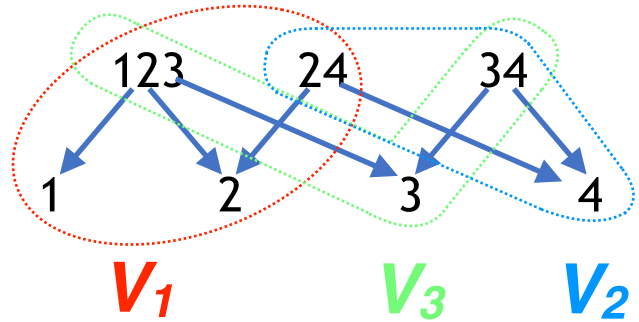
$$(\sigma, \tau) \in A \longleftrightarrow \begin{cases} \sigma \in K_{top} \text{ and } (\sigma, \tau) \in R_{i,0} \\ \sigma, \tau \in K_{top} \text{ and } (\sigma, \tau) \in R_{i,i} \\ \tau \in K_{top} \text{ and } (\sigma, \tau) \in \end{cases}$$

- ***Compact**: it explicitly stores just a fraction of the entities of a simplicial complex*
- ***Not** all the relations between simplices are **immediately available***

Top-based Representations

Stellar Tree:





Given a decomposition of K_0, the simplicial complex K is encoded via a ***directed graph*** $\boldsymbol{G = (N, A)}$:

$$N \longleftrightarrow (K_0 = V_1 \cup V_2 \cup \ldots \cup V_n) \cup K_{top}$$

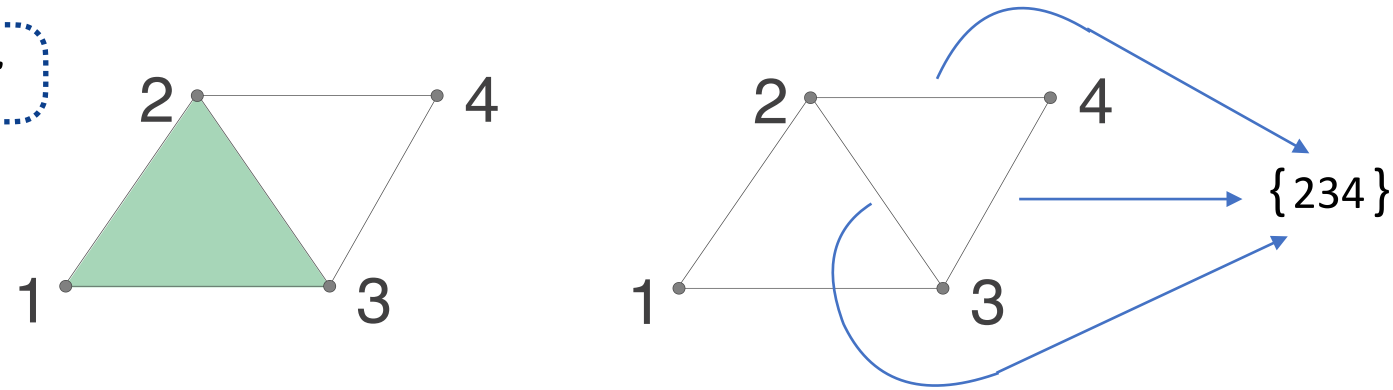
$$(\sigma, \tau) \in A \longleftrightarrow \sigma \in K_{top} \text{ and } (\sigma, \tau) \in R_{i,0}$$

plus a ***map*** returning, for each j, the vertices of K in V_j and the top simplices with at least one vertex in V_j

- ***Compact*** *and* ***highly adjustable*** *(e.g. choice of the decomposition, of the maximum number of vertices in each region)*
- ***Not*** *all the relations between simplices are* ***immediately available***

Operator-driven Representations

Skeleton Blocker:



The simplicial complex K is encoded by storing its ***1-skeleton*** (i.e. the graph consisting of the 0- and the 1-simplices) and a ***map*** returning, for each 1-simplex σ, the blockers of K containing σ, where:

*A simplex τ is a **blocker** if τ does not belong to K but all its faces do*

- ***Designed for*** flag complexes (e.g. ***VR complexes***) and edge contraction
- Too specific: ***inefficient in any other task***

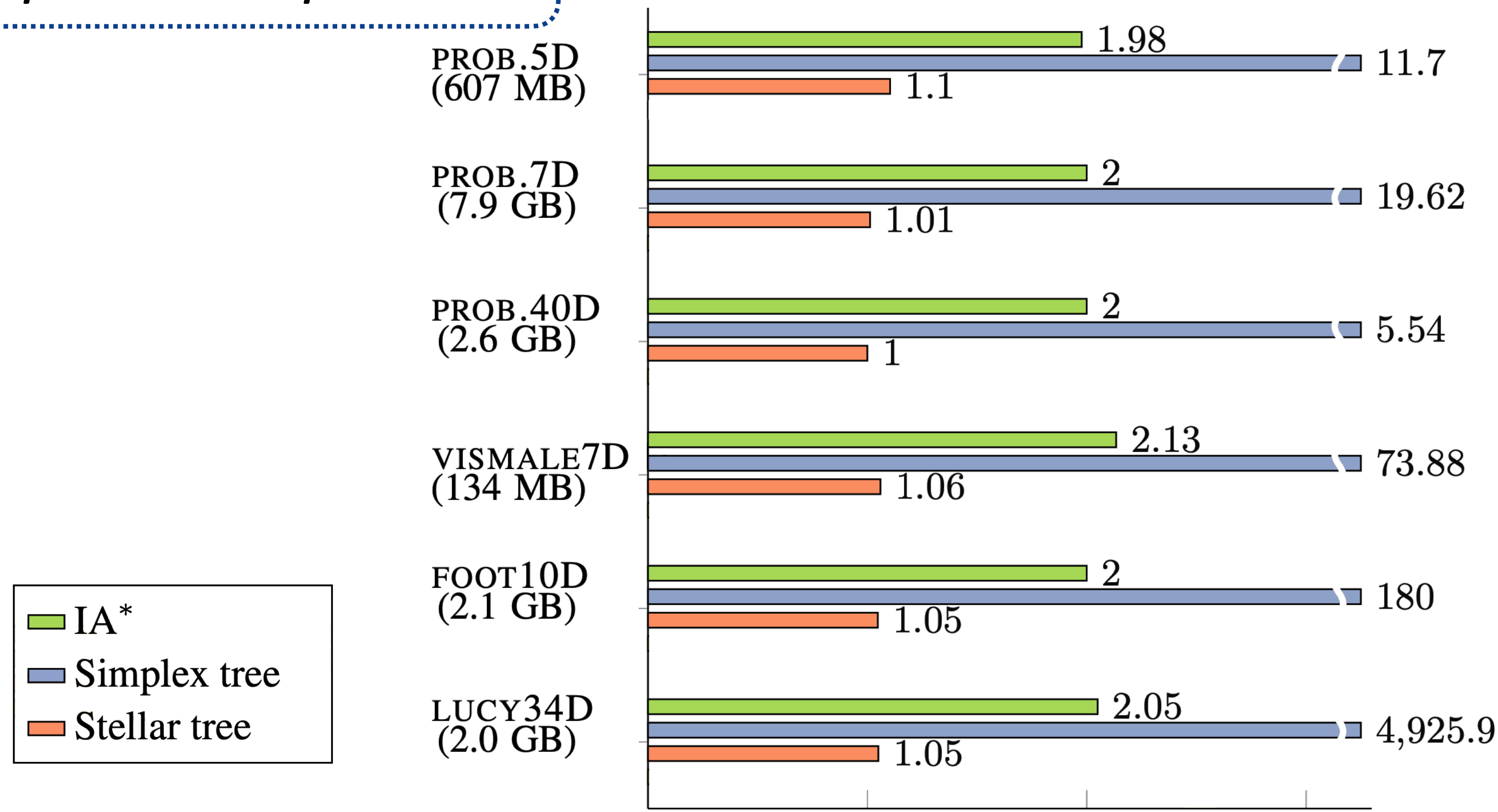
Encoding Simplicial Complexes

Top-based vs Simplex-based:

Dataset	d	$\|\Sigma_0\|$	$\|\Sigma_{top}\|$	$\|\Sigma\|$	Storage Cost		
					IA^*	IG	ST
DTI-SCAN	3	0.9M	5.5M	24M	0.97	11.9	2.4
VISMALE	3	4.6M	26M	118M	4.7	-	9.7
ACKLEY4	4	1.5M	32M	204M	6.8	-	12.8
AMAZON01	6	0.2M	0.4M	2.2M	0.12	1.6	0.3
AMAZON02	7	0.4M	1.0M	18.4M	0.28	9.8	1.5
ROADNET	3	1.9M	2.5M	4.8M	0.8	3.3	1.0
SPHERE-1.0	16	100	224	0.6M	0.003	0.9	0.04
SPHERE-1.2	21	100	285	26M	0.0032	-	1.5
SPHERE-1.3	23	100	382	197M	0.0034	-	11.01

Encoding Simplicial Complexes

Top-based vs Simplex-based:



Encoding Simplicial Complexes

Top-based vs Operator-driven:

data	ω		contr. edges	timings check	timings contr.	timings tot	memory peak gen.	memory peak simpl.
CHICAGO	28	weak	6.38K	9.15h	2.27m	9.19h	5.6	57.2K
		top		0.01s	0.02s	0.09s		7.6
		Skel.		0.00s	0.15s	0.15s	7.8	7.8
	56	weak	7.99K	out-of-memory			6.2	–
		top		0.04s	0.06s	0.23s		10.8
		Skel.		0.00s	0.71s	0.71s	14.1	14.1
ATHENS	63	weak	27.9K	out-of-memory			11.6	–
		top		0.08s	0.11s	0.38s		14.9
		Skel.		0.00s	0.74s	0.75s	26.4	26.8
	126	weak	31.2K	out-of-memory			10.0	–
		top		0.40s	0.49s	1.36s		25.9
		Skel.		0.01s	7.73s	7.74s	66.1	66.7
VISMALE	3.5	weak	4.23M	34.3m	1.28m	40.4m	1.0K	2.0K
		top		4.34m	0.89m	7.20m		2.0K
		Skel.		0.76m	3.34h	3.35h	8.0K	8.0K
FOOT	4.5	weak	4.69M	killed after 25 hours			7.5K	–
		top		2.89h	26.0m	3.32h		10.7K
		Skel.		killed after 25 hours			19.4K	–
LUCY	1.5	weak	14.0M	killed after 25 hours			7.5K	–
		top		11.9m	14.8m	32.0m		15.4K
		Skel.		23.19s	14.6h	14.6h	50.9K	52.1K

Encoding Simplicial Complexes

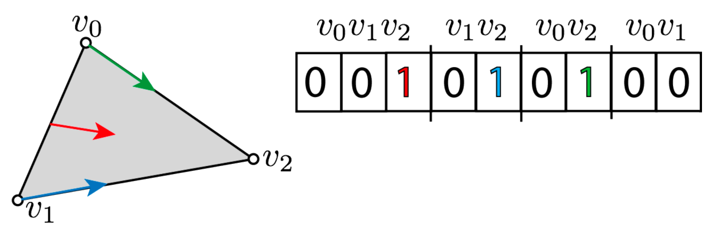
Possible Issues in Top-based Representations:

Top-based representations are promising data structures for encoding a simplicial complex K

but, how to …

- ***Store information associated to each simplex of K (e.g. labels, gradient, …)?***

Attach information to the top simplices only



- ***Efficiently perform operators having explicitly stored a fraction of the entities of K?***

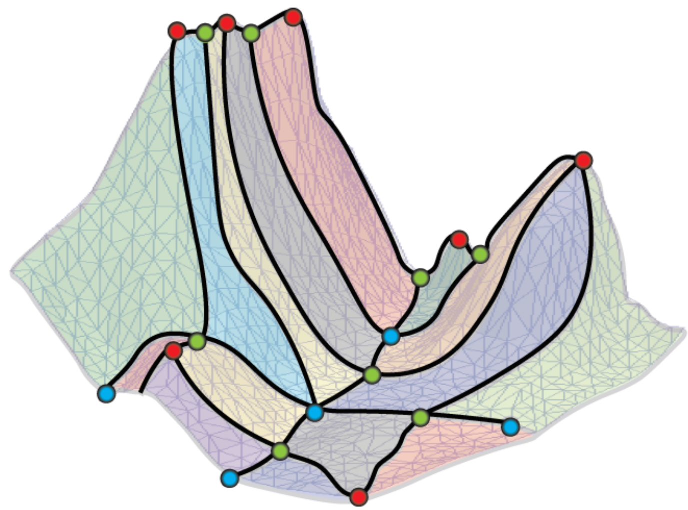
Re-define the algorithms performing the operators trying to extract the lowest possible amount of non-explicitly stored entities

Bibliography

Some References:

- ***Data Structures for Arbitrary Simplicial Complexes:***
  - D. Canino, L. De Floriani, K. Weiss. ***IA*: an adjacency-based representation for non-manifold simplicial shapes in arbitrary dimensions.*** Computers & Graphics, 35.3, pages 747-753, 2011.
  - D. Attali, A. Lieutier, D. Salinas. ***Efficient data structure for representing and simplifying simplicial complexes in high dimensions.*** International Journal of Computational Geometry & Applications, 22.4, pages 279-303, 2012.
  - J.D. Boissonnat, C. Maria. ***The simplex tree: An efficient data structure for general simplicial complexes.*** Algorithmica, 70.3, pages 406-427, 2014.
  - R. Fellegara, K. Weiss, L. De Floriani. **The Stellar tree: a compact representation for simplicial complexes and beyond.** arXiv preprint:1707.02211, 2017.
  - U. Fugacci, F. Iuricich, L. De Floriani. ***Computing discrete Morse complexes from simplicial complexes.*** Graphical models, 103, 101023, 2019.
  - R. Fellegara, F. Iuricich, L. De Floriani, U. Fugacci. ***Efficient Homology-Preserving Simplification of High-Dimensional Simplicial Shapes.*** Computer Graphics Forum, 39.1, pages 244-259, 2020.

Possible Topics for Seminars



Discrete Morse Theory

Study the shape of a space by studying the behavior of a function defined on it

Possible Topics for Seminars

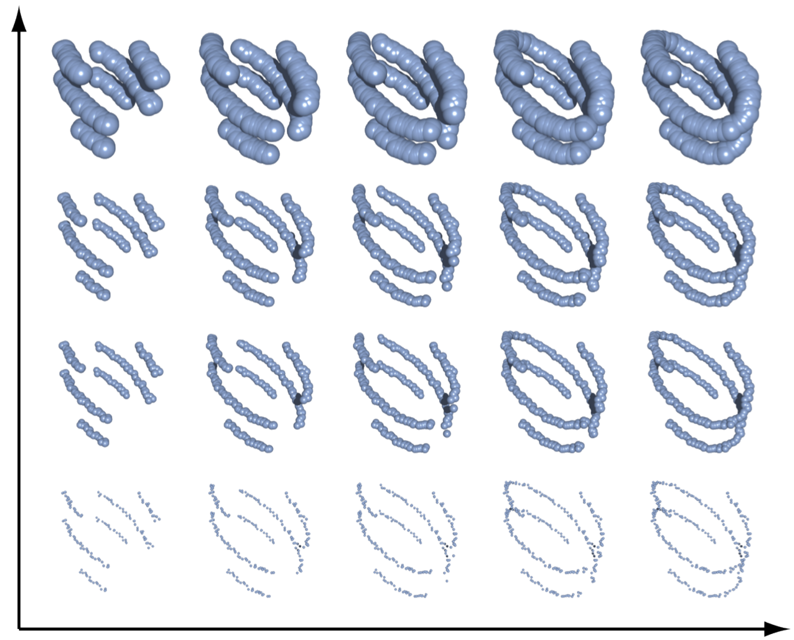
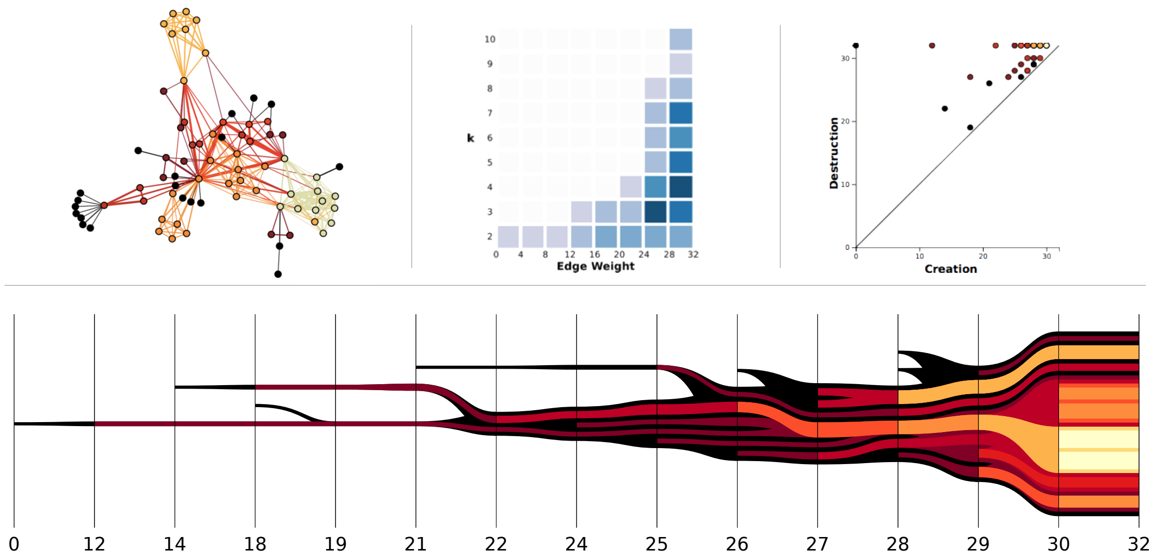


Image courtesy of
[Carlsson & Zomorodian 2009]

Multi-Parameter Persistent Homology

What if we consider multiple filtering functions?

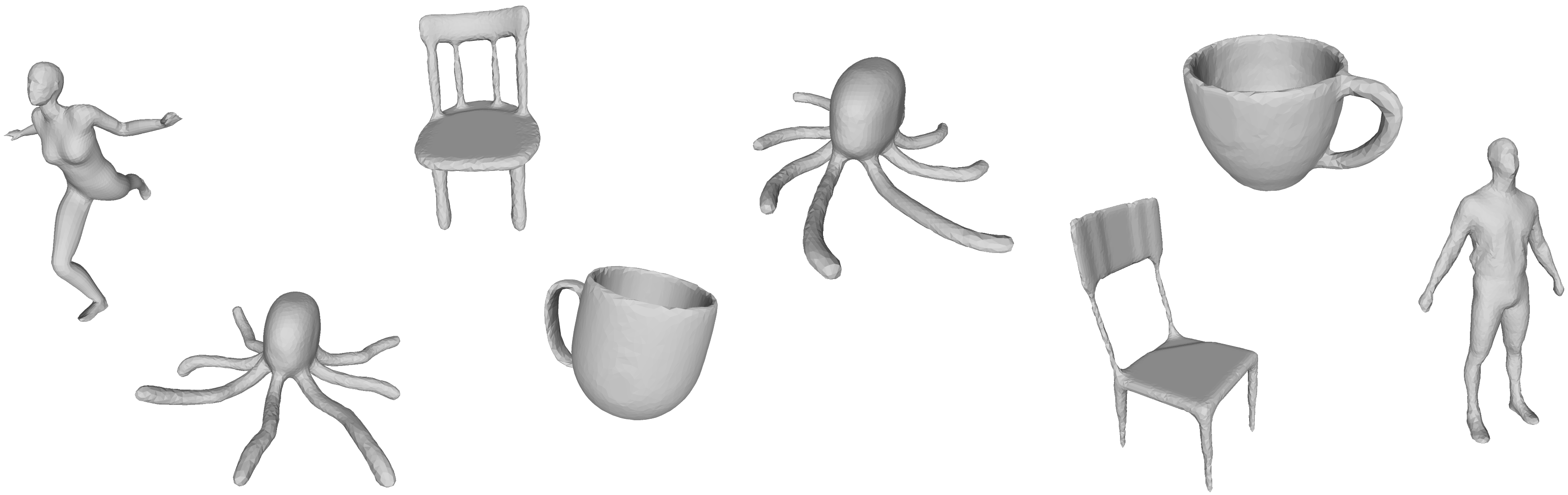
Possible Topics for Seminars



Persistent Homology & Networks

- ***Homological Scaffolds:*** *Topological summaries of weighted graphs*
- ***Clique Community Persistence:*** *Tracking the evolution of network communities*

Possible Topics for Seminars



Algorithms & Implementation

- *Efficient computation of **Vietoris-Rips complexes** and other data-to-complex strategies*
- *Focus on a specific **algorithm for speed-up persistent homology computation***
- *Use of available **software tools** for testing persistent homology on various datasets*