Ripser

Efficient computation of Vietoris–Rips persistence barcodes

Ulrich Bauer

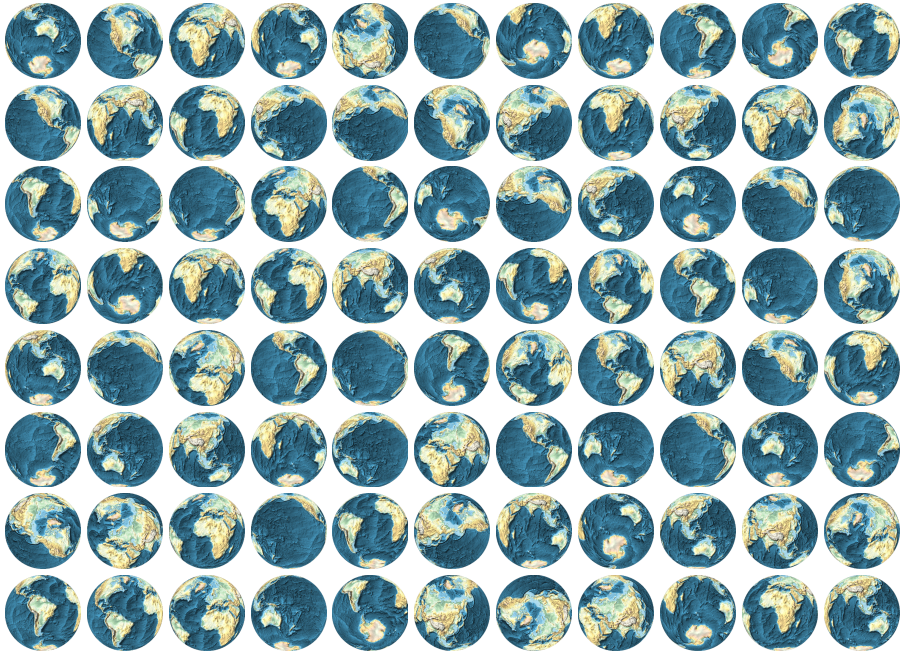
TUM

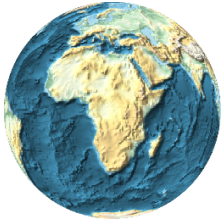
Dec 3, 2019

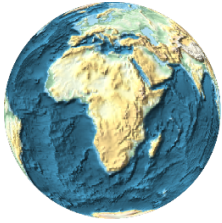
Mathematical Software Day
MPI für Mathematik in den Naturwissenschaften Leipzig

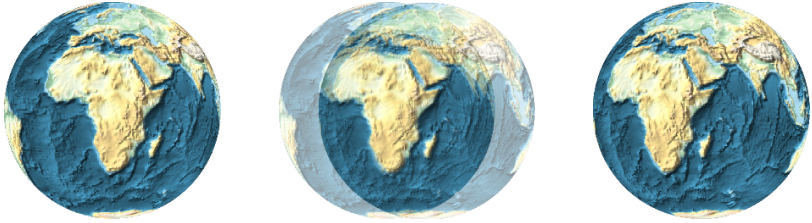
Finding topology in data sets

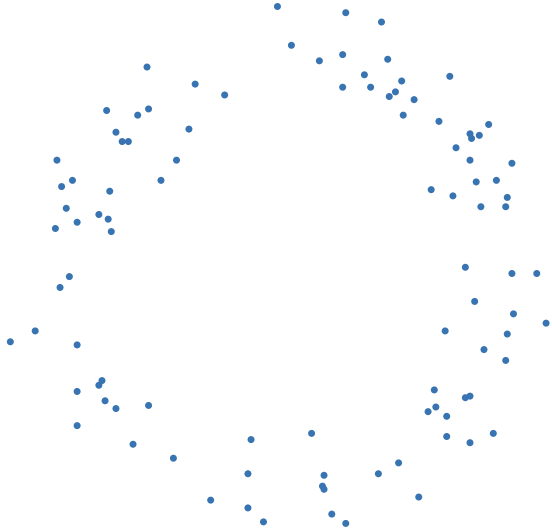
(Columbia Object Image Library)

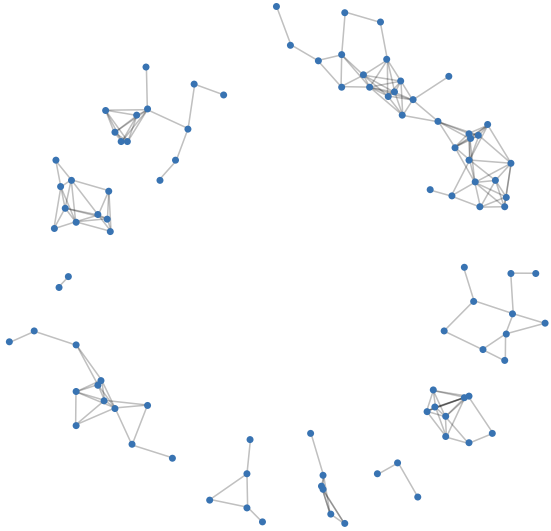


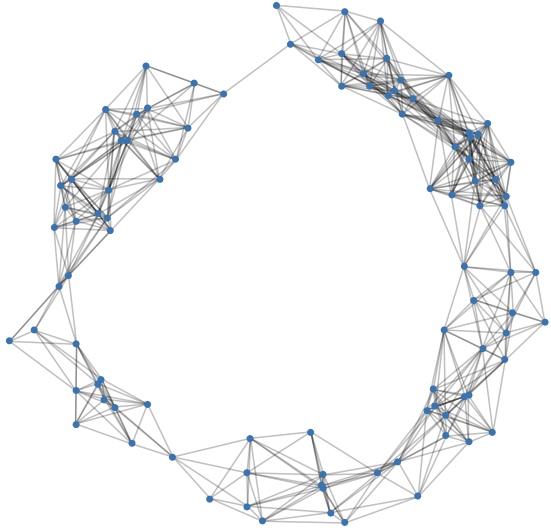


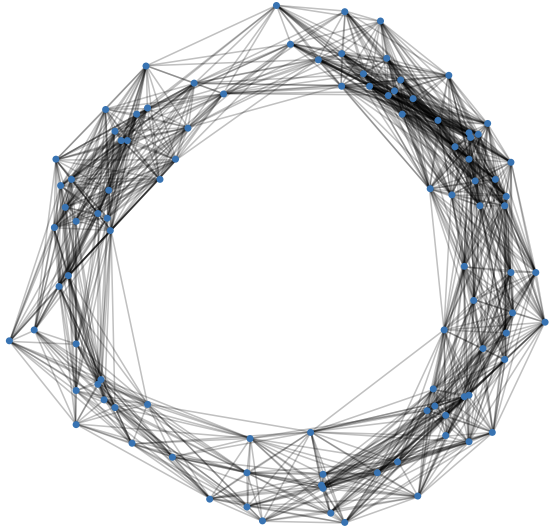


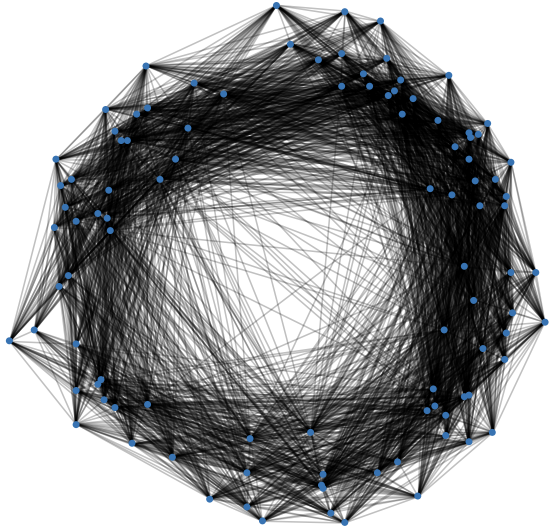


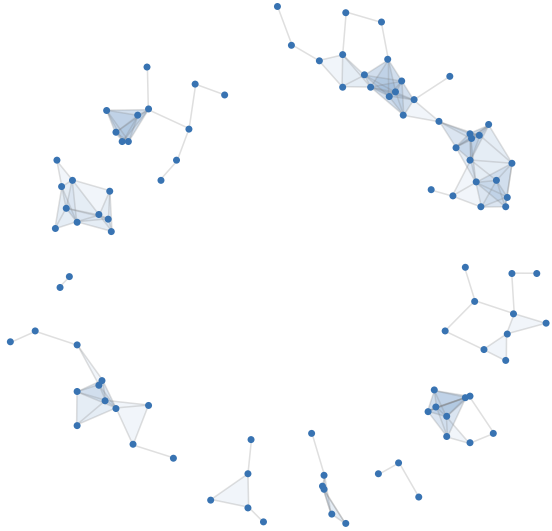


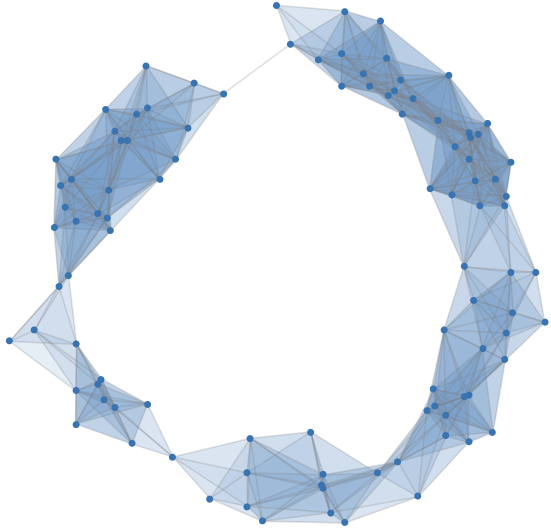


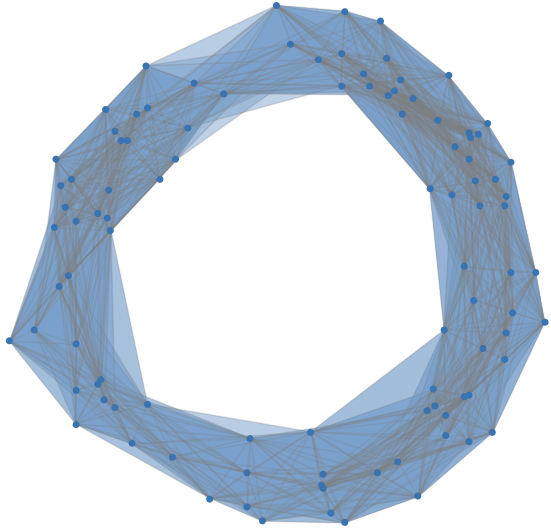


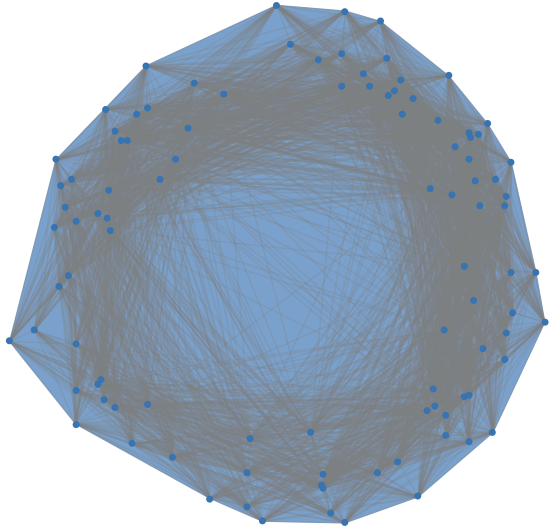












Vietoris–Rips complexes

Consider a finite metric space (X, d).
The *Vietoris–Rips complex* is the simplicial complex

$$\mathrm{Rips}_t(X) = \{S \subseteq X \mid \operatorname{diam} S \leq t\}$$

- all edges with pairwise distance $\leq t$
- all possible higher simplices

Vietoris–Rips complexes

Consider a finite metric space (X, d).

The *Vietoris–Rips complex* is the simplicial complex

$$\mathrm{Rips}_t(X) = \{S \subseteq X \mid \operatorname{diam} S \leq t\}$$

- all edges with pairwise distance $\leq t$
- all possible higher simplices

For large t, $\mathrm{Rips}_t(X)$ is the full simplex with vertices X

- Number of d-simplices is $\binom{|X|}{d+1}$
- Computation is one of the most important challenges in applied topology!

An example computation

Example data set:

- 192 points on $\mathbb{S}^2$
- persistent homology barcodes up to dimension 2
- over 56 mio. simplices in 3-skeleton

An example computation

Example data set:

- 192 points on $\mathbb{S}^2$
- persistent homology barcodes up to dimension 2
- over 56 mio. simplices in 3-skeleton

Some previous software:

- javaplex (Stanford): 3200 seconds, 12 GB
- Dionysus (Duke): 615 seconds, 3.4 GB
- DIPHA (IST Austria): 50 seconds, 6 GB
- GUDHI (INRIA): 60 seconds, 3 GB

An example computation

Example data set:

- 192 points on $\mathbb{S}^2$
- persistent homology barcodes up to dimension 2
- over 56 mio. simplices in 3-skeleton

Some previous software:

- javaplex (Stanford): 3200 seconds, 12 GB
- Dionysus (Duke): 615 seconds, 3.4 GB
- DIPHA (IST Austria): 50 seconds, 6 GB
- GUDHI (INRIA): 60 seconds, 3 GB

Demo: live.ripser.org

An example computation

Example data set:

- 192 points on $\mathbb{S}^2$
- persistent homology barcodes up to dimension 2
- over 56 mio. simplices in 3-skeleton

Some previous software:

- javaplex (Stanford): 3200 seconds, 12 GB
- Dionysus (Duke): 615 seconds, 3.4 GB
- DIPHA (IST Austria): 50 seconds, 6 GB
- GUDHI (INRIA): 60 seconds, 3 GB

Demo: live.ripser.org

- Ripser: 1.2 seconds, 160 MB

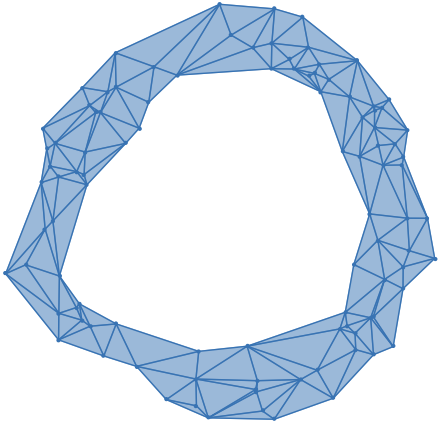
Ripser

A software for computing Vietoris–Rips persistent homology

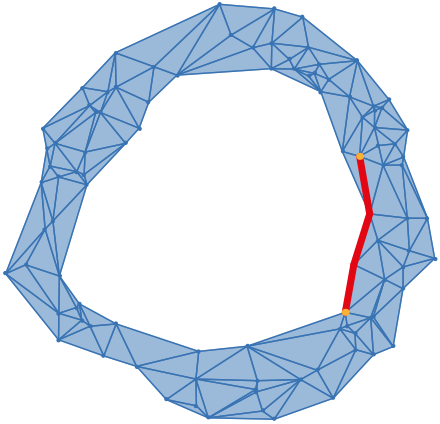
- around 1000 lines of C++ code, no external dependencies
- support for
  - coefficients in a prime field $\mathbb{Z}/p\mathbb{Z}$
  - sparse distance matrices (for distance threshold)
- open source (`http://ripser.org`)
- online version (`http://live.ripser.org`), based on
  - WebAssembly (C++ within the browser)
  - D3.js (JavaScript visualization toolbox)
- 2016 ATMCS Best New Software Award (joint with RIVET by M. Lesnick and M. Wright)

Persistent homology

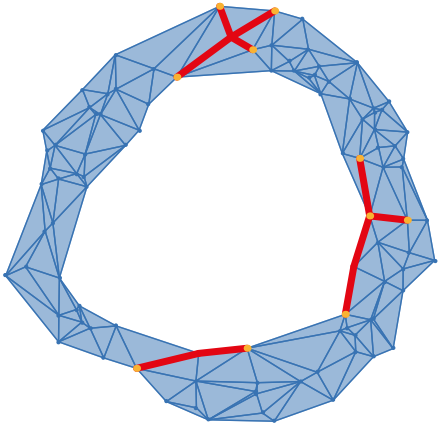
What is homology?



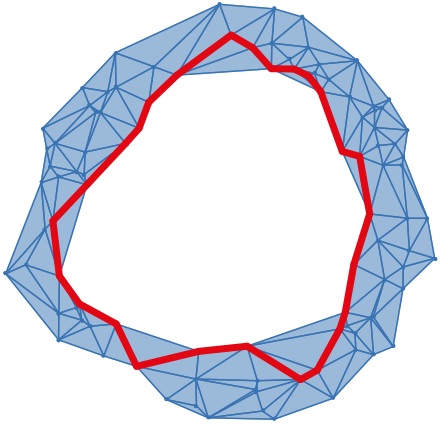
What is homology?



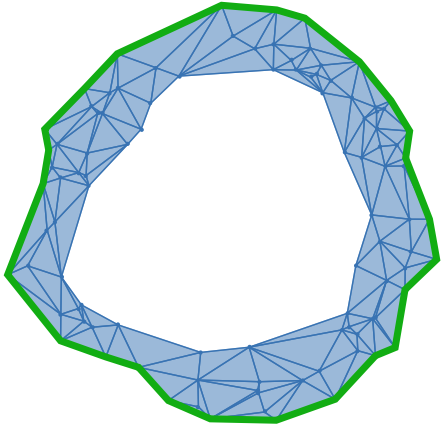
What is homology?



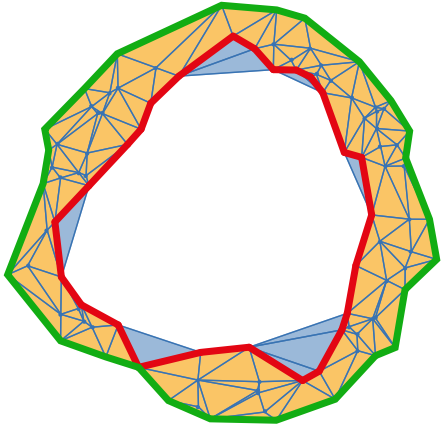
What is homology?



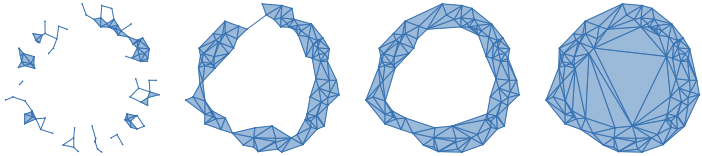
What is homology?



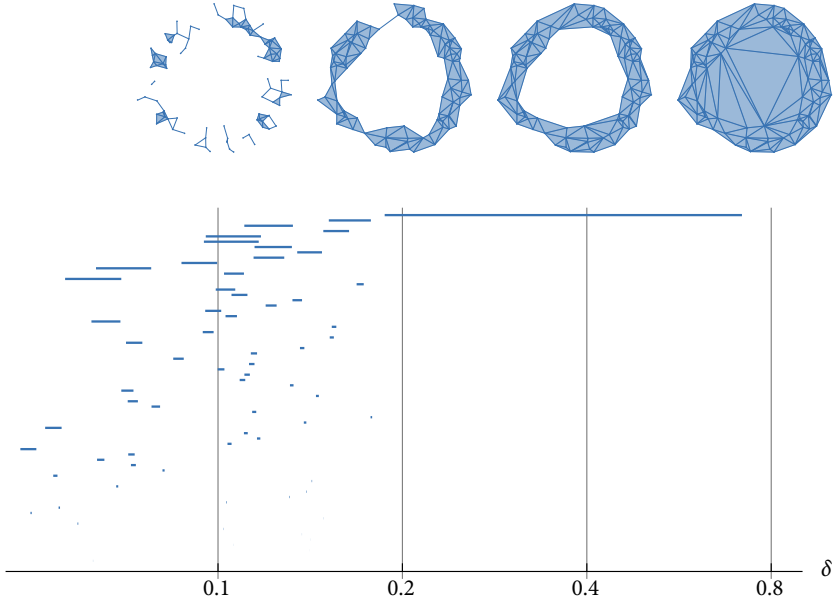
What is homology?



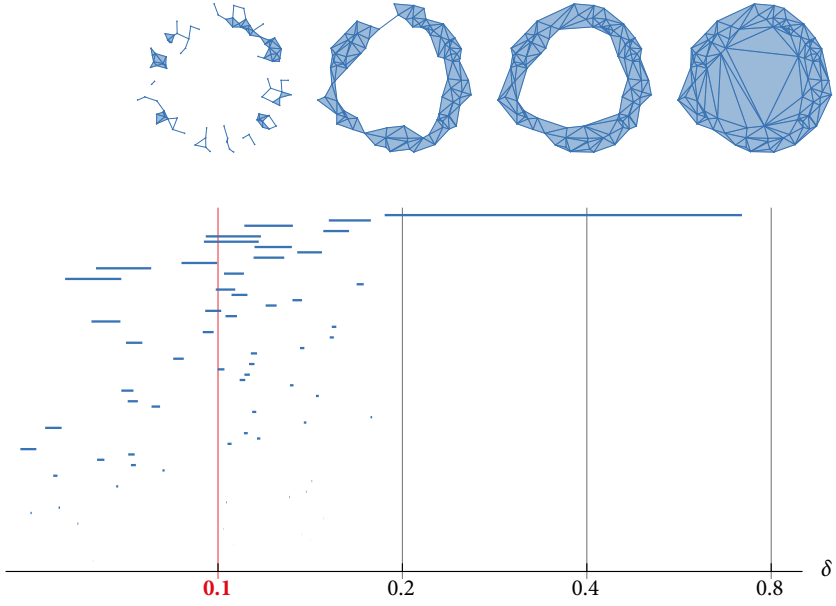
What is persistent homology?



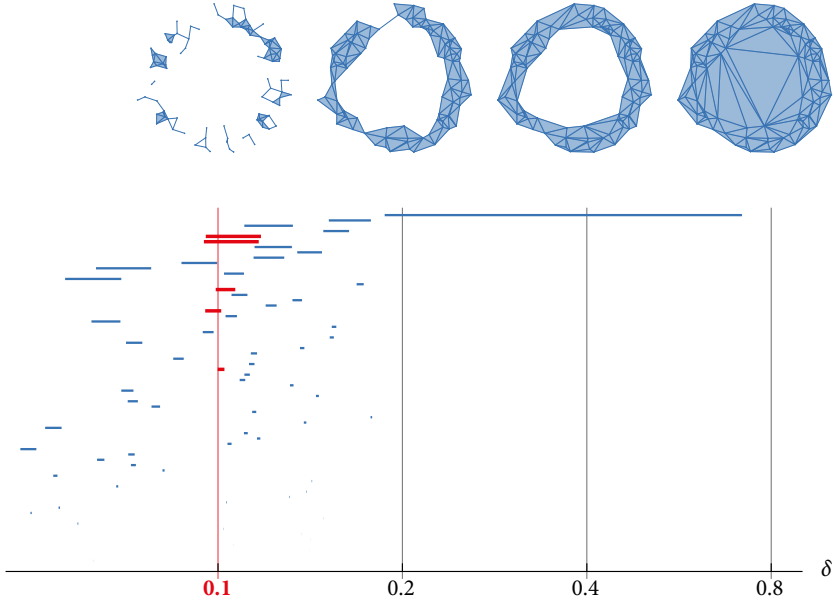
What is persistent homology?



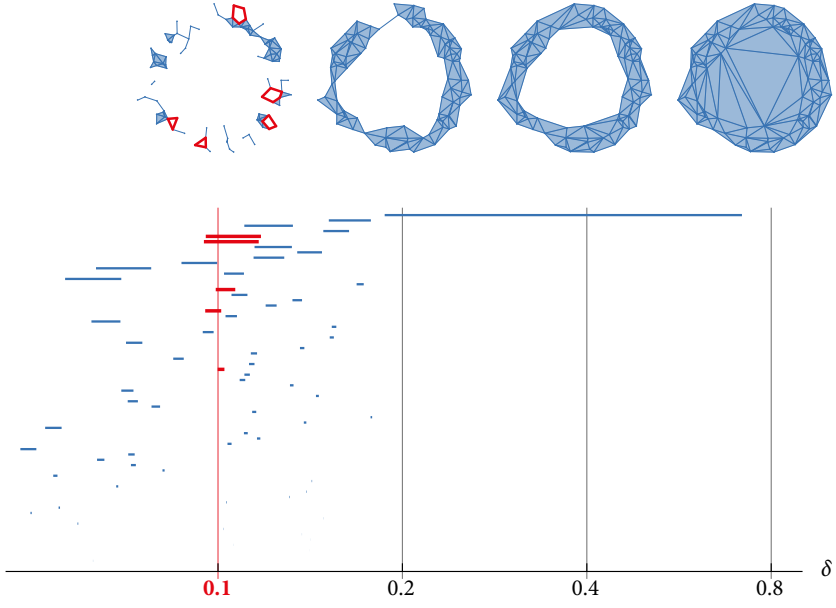
What is persistent homology?



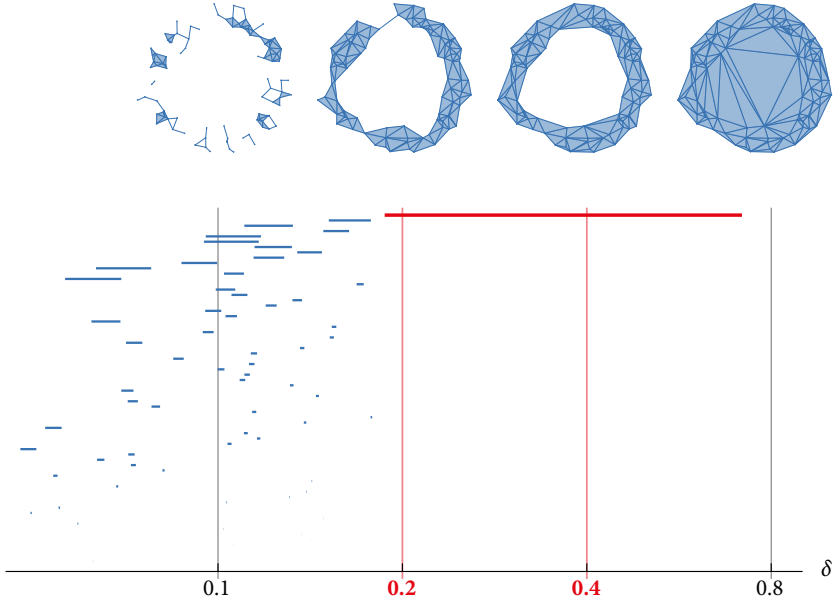
What is persistent homology?



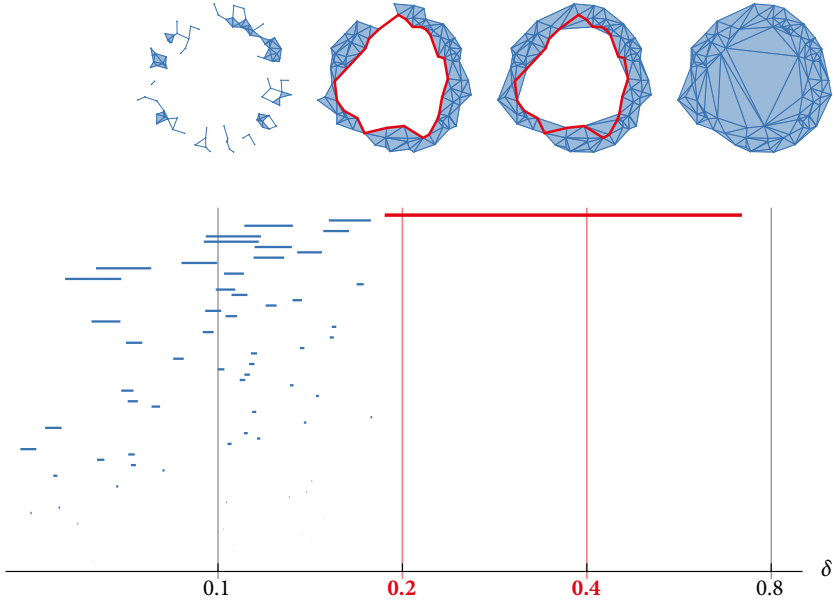
What is persistent homology?

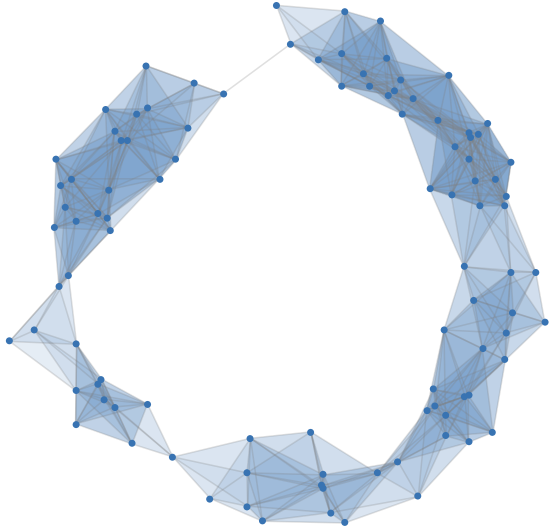


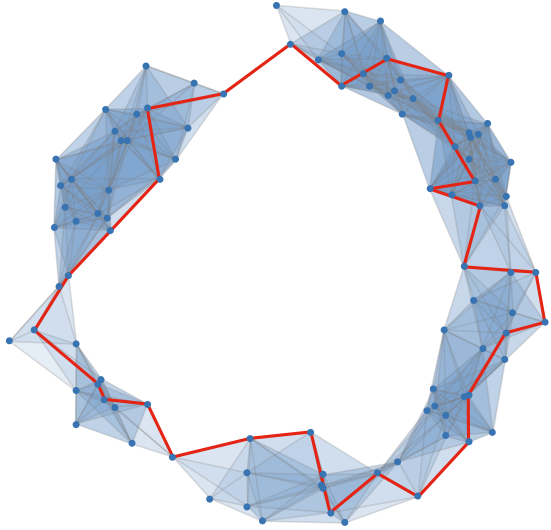
What is persistent homology?

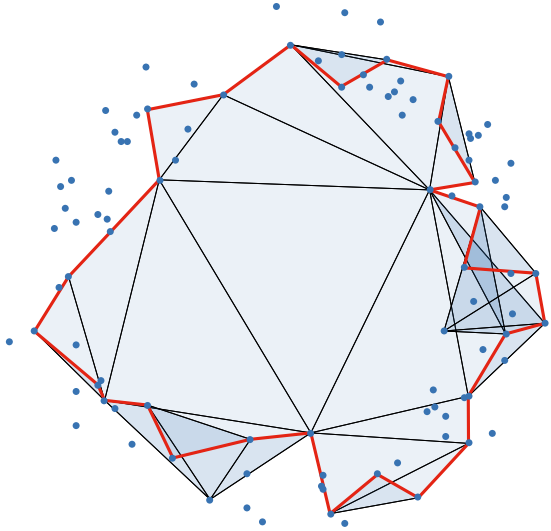


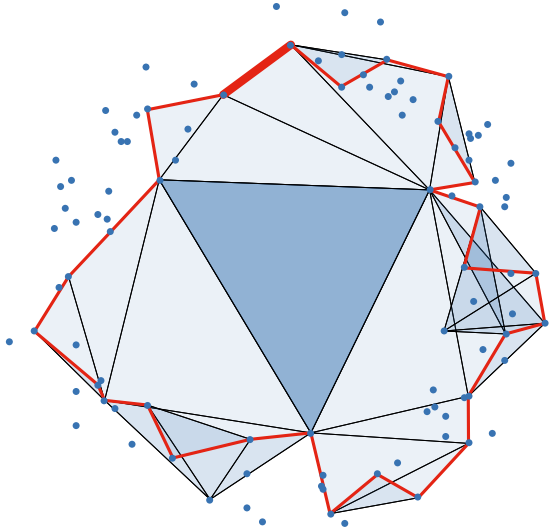
What is persistent homology?











What is persistent homology?

What is persistent homology?

Persistent homology is the homology of a filtration.

What is persistent homology?

Persistent homology is the homology of a filtration.

- A filtration is a certain diagram $K : \mathbb{R} \to \mathbf{Top}$ of topological spaces, indexed over the reals:

$$\cdots\cdots\rightarrow K_s \hookrightarrow K_t \cdots\cdots\rightarrow$$

  - a topological space K_t for each $t \in \mathbb{R}$
  - an inclusion map $K_s \hookrightarrow K_t$ for each $s \leq t \in \mathbb{R}$

What is persistent homology?

Persistent homology is the homology of a filtration.

- A filtration is a certain diagram $K : \mathbb{R} \to \mathbf{Top}$ of topological spaces, indexed over the reals:

$$\cdots\cdots\rightarrow K_s \hookrightarrow K_t \cdots\cdots\rightarrow$$

  - a topological space K_t for each $t \in \mathbb{R}$
  - an inclusion map $K_s \hookrightarrow K_t$ for each $s \leq t \in \mathbb{R}$
- Apply homology $H_* : \mathbf{Top} \to \mathbf{Vect}$

What is persistent homology?

Persistent homology is the homology of a filtration.

- A filtration is a certain diagram $K : \mathbb{R} \to \mathbf{Top}$ of topological spaces, indexed over the reals:

$$\cdots\cdots\cdots\rightarrow K_s \hookrightarrow K_t \cdots\cdots\cdots\rightarrow$$

  - a topological space K_t for each $t \in \mathbb{R}$
  - an inclusion map $K_s \hookrightarrow K_t$ for each $s \leq t \in \mathbb{R}$
- Apply homology $H_* : \mathbf{Top} \to \mathbf{Vect}$
- Persistent homology is a diagram $M : \mathbb{R} \to \mathbf{Vect}$ *(persistence module)*.

In this talk, all vector spaces will be finite dimensional.

Persistence modules

A *persistence module* M is a diagram $M : \mathbb{R} \to \mathbf{vect}$ of vector spaces, indexed over the reals:

$$\cdots\cdots\rightarrow\ M_s \longrightarrow M_t\ \cdots\cdots\rightarrow$$

- a vector space M_t for each $t \in \mathbb{R}$
- an linear map $M_s \to M_t$ for each $s \leq t \in \mathbb{R}$

Persistence modules

A *persistence module* M is a diagram $M : \mathbb{R} \to \mathbf{vect}$ of vector spaces, indexed over the reals:

$$\cdots\cdots\rightarrow\ M_s \longrightarrow M_t\ \cdots\cdots\rightarrow$$

- a vector space M_t for each $t \in \mathbb{R}$
- an linear map $M_s \to M_t$ for each $s \leq t \in \mathbb{R}$

The maps (*morphisms*) between persistence modules are *natural transformations*:

$$\begin{array}{ccccc} \cdots\cdots\rightarrow & M_s & \longrightarrow & M_t & \cdots\cdots\rightarrow \\ & \downarrow{\scriptstyle f_s} & & \downarrow{\scriptstyle f_t} & \\ \cdots\cdots\rightarrow & N_s & \longrightarrow & N_t & \cdots\cdots\rightarrow \end{array}$$

- a linear map $f_t : M_t \to N_t$ for each $t \in \mathbb{R}$
- horizontal and vertical maps commute (for all $s \leq t$)

Barcodes: the structure of persistence modules

Theorem (Crawley-Boevey 2015)

Any persistence module $M : \mathbb{R} \to \mathbf{vect}$ *(of finite dim. vector spaces over some field* $\mathbb{F}$*) decomposes as a direct sum of* interval modules

$$0 \to \cdots \to 0 \to \mathbb{F} \to \cdots \to \mathbb{F} \to 0 \to \cdots \to 0.$$

Barcodes: the structure of persistence modules

Theorem (Crawley-Boevey 2015)

Any persistence module $M : \mathbb{R} \to \mathbf{vect}$ *(of finite dim. vector spaces over some field* $\mathbb{F}$*) decomposes as a direct sum of* interval modules

$$0 \to \cdots \to 0 \to \mathbb{F} \to \cdots \to \mathbb{F} \to 0 \to \cdots \to 0.$$

- The supporting intervals form the *persistence barcode* $B(M)$.

Barcodes: the structure of persistence modules

Theorem (Crawley-Boevey 2015)

Any persistence module $M : \mathbb{R} \to \mathbf{vect}$ *(of finite dim. vector spaces over some field* $\mathbb{F}$*) decomposes as a direct sum of* interval modules

$$0 \to \cdots \to 0 \to \mathbb{F} \to \cdots \to \mathbb{F} \to 0 \to \cdots \to 0.$$

- The supporting intervals form the *persistence barcode* $B(M)$.
- The decomposition is not unique, but the barcode is.

Barcodes: the structure of persistence modules

Theorem (Crawley-Boevey 2015)

Any persistence module $M : \mathbb{R} \to \mathbf{vect}$ *(of finite dim. vector spaces over some field* $\mathbb{F}$*) decomposes as a direct sum of* interval modules

$$0 \to \cdots \to 0 \to \mathbb{F} \to \cdots \to \mathbb{F} \to 0 \to \cdots \to 0.$$

- The supporting intervals form the *persistence barcode* $B(M)$.
- The decomposition is not unique, but the barcode is.
- The barcode completely describes the persistence module (up to isomorphism).

Barcodes: the structure of persistence modules

Theorem (Crawley-Boevey 2015)

Any persistence module $M : \mathbb{R} \to \mathbf{vect}$ *(of finite dim. vector spaces over some field* $\mathbb{F}$*) decomposes as a direct sum of* interval modules

$$0 \to \cdots \to 0 \to \mathbb{F} \to \cdots \to \mathbb{F} \to 0 \to \cdots \to 0.$$

- The supporting intervals form the *persistence barcode* $B(M)$.
- The decomposition is not unique, but the barcode is.
- The barcode completely describes the persistence module (up to isomorphism).
- This is why we use homology with coefficients in a field.

Barcodes: the structure of persistence modules

Theorem (Crawley-Boevey 2015)

Any persistence module $M : \mathbb{R} \to \mathbf{vect}$ *(of finite dim. vector spaces over some field* $\mathbb{F}$*) decomposes as a direct sum of* interval modules

$$0 \to \cdots \to 0 \to \mathbb{F} \to \cdots \to \mathbb{F} \to 0 \to \cdots \to 0.$$

- The supporting intervals form the *persistence barcode* $B(M)$.
- The decomposition is not unique, but the barcode is.
- The barcode completely describes the persistence module (up to isomorphism).
- This is why we use homology with coefficients in a field.
- We rarely have a similarly clean structure for other parameter spaces, like $\mathbb{R}^2 \to \mathbf{vect}$ (two-parameter persistence modules)

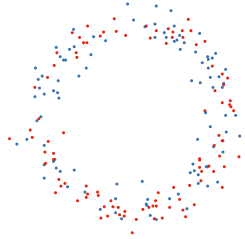
Stability

Stability of Vietoris–Rips persistence barcodes

If two finite metric spaces are close, then their barcodes are also close:

Theorem (Cohen-Steiner, Edelsbrunner, Harer 2005; Chazal et al. 2009)

Let X, Y be finite metric spaces with Gromov–Hausdorff distance $d_{GH}(X, Y) = \frac{\delta}{2}$.

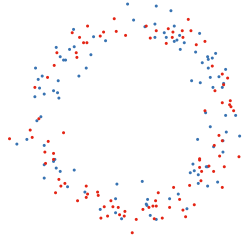


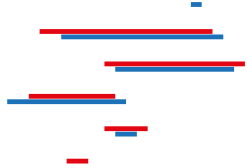
Stability of Vietoris–Rips persistence barcodes

If two finite metric spaces are close, then their barcodes are also close:

Theorem (Cohen-Steiner, Edelsbrunner, Harer 2005; Chazal et al. 2009)

Let X, Y be finite metric spaces with Gromov–Hausdorff distance $d_{GH}(X, Y) = \frac{\delta}{2}$.
*Then there exists a δ-*matching *between the intervals in the*
Vietoris–Rips persistence barcodes for X and Y:





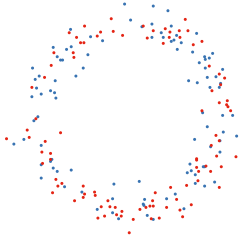
Stability of Vietoris–Rips persistence barcodes

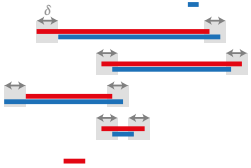
If two finite metric spaces are close, then their barcodes are also close:

Theorem (Cohen-Steiner, Edelsbrunner, Harer 2005; Chazal et al. 2009)

Let X, Y be finite metric spaces with Gromov–Hausdorff distance $d_{GH}(X, Y) = \frac{\delta}{2}$.
*Then there exists a δ-*matching *between the intervals in the*
Vietoris–Rips persistence barcodes for X and Y:

- *matched intervals have endpoints within distance $\leq \delta$, and*





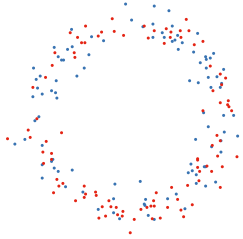
Stability of Vietoris–Rips persistence barcodes

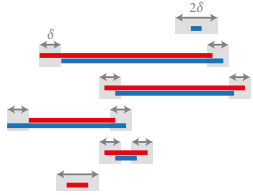
If two finite metric spaces are close, then their barcodes are also close:

Theorem (Cohen-Steiner, Edelsbrunner, Harer 2005; Chazal et al. 2009)

Let X, Y be finite metric spaces with Gromov–Hausdorff distance $d_{GH}(X, Y) = \frac{\delta}{2}$.
*Then there exists a δ-*matching *between the intervals in the*
Vietoris–Rips persistence barcodes for X and Y:

- *matched intervals have endpoints within distance $\leq \delta$, and*
- *unmatched intervals have length $\leq 2\delta$.*





Matrix reduction

Computing homology

Computing homology $H_* = Z_*/B_*$:

- compute basis for boundaries $B_* = \operatorname{im} \partial_*$
- extend to basis for cycles $Z_* = \ker \partial_*$
- new (non-boundary) basis cycles generate quotient Z_*/B_*

Computing homology

Computing homology $H_* = Z_*/B_*$:

- compute basis for boundaries $B_* = \operatorname{im} \partial_*$
- extend to basis for cycles $Z_* = \ker \partial_*$
- new (non-boundary) basis cycles generate quotient Z_*/B_*

Computing *persistent* homology $H_* = Z_*/B_*$ (for a simplexwise filtration $K_i \subseteq K$):

- compute *filtered* basis for boundaries $B_* = \operatorname{im} \partial_*$
- extend to basis for cycles $Z_* = \ker \partial_*$
- all basis cycles generate *persistent* homology

Persistence by matrix reduction

Given:

- D: matrix of boundary ∂_d for a simplexwise filtration $(K_i)_i$ (for canonical basis in filtration order, indexed by $I \times J$)

Persistence by matrix reduction

Given:

- D: matrix of boundary ∂_d for a simplexwise filtration $(K_i)_i$ (for canonical basis in filtration order, indexed by $I \times J$)

Wanted:

- persistence barcode of homology $H_d(K_i; \mathbb{F})$ for some (prime) field $\mathbb{F}$, in dimensions $d = 0, \ldots, k$

Persistence by matrix reduction

Given:

- D: matrix of boundary ∂_d for a simplexwise filtration $(K_i)_i$ (for canonical basis in filtration order, indexed by $I \times J$)

Wanted:

- persistence barcode of homology $H_d(K_i; \mathbb{F})$ for some (prime) field $\mathbb{F}$, in dimensions $d = 0, \ldots, k$

Computation: barcode is obtained by *matrix reduction* of D

- $R = D \cdot V$ *reduced* (non-zero columns have distinct pivots)
- V is regular upper triangular

Persistence by matrix reduction

Given:

- D: matrix of boundary ∂_d for a simplexwise filtration $(K_i)_i$ (for canonical basis in filtration order, indexed by $I \times J$)

Wanted:

- persistence barcode of homology $H_d(K_i; \mathbb{F})$ for some (prime) field $\mathbb{F}$, in dimensions $d = 0, \ldots, k$

Computation: barcode is obtained by *matrix reduction* of D

- $R = D \cdot V$ *reduced* (non-zero columns have distinct pivots)
- V is regular upper triangular

Notation:

- M_j: column j of M; m_{ij}: entry in row i and column j
- $\operatorname{PivotIndex} M_j = \min\{i \in I : m_{kj} = 0 \text{ for all } k > i\}$.

Matrix reduction

$$
\begin{array}{c|ccccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
\hline
1 & & & 1 & & 1 & & \\
2 & & & 1 & & & 1 & \\
3 & & & & & & & 1 \\
4 & & & & & 1 & 1 & \\
5 & & & & & & & 1 \\
6 & & & & & & & 1 \\
7 & & & & & & &
\end{array}
= D \cdot
\begin{array}{c|ccccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
\hline
1 & 1 & & & & & & \\
2 & & 1 & & & & & \\
3 & & & 1 & & & & \\
4 & & & & 1 & & & \\
5 & & & & & 1 & & \\
6 & & & & & & 1 & \\
7 & & & & & & & 1
\end{array}
$$

R $\qquad$ V

Matrix reduction

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

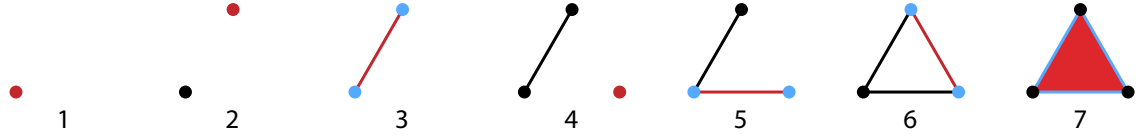
R

$= D \cdot$

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

V

Matrix reduction



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

R

$= D \cdot$

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

V

Matrix reduction

	1	2	3	4	5	6	7
1			1		1	1	
2			1			1	
3							1
4					1	0	
5							1
6							1
7							

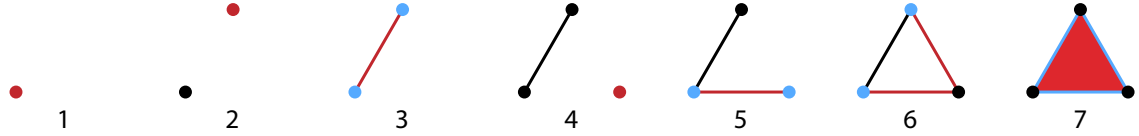
R

$= D \cdot$

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

V

Matrix reduction



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

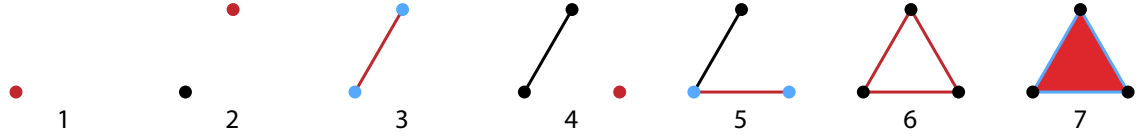
R

$= D \cdot$

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

V

Matrix reduction



	1	2	3	4	5	6	7
1			1		1	0	
2			1			0	
3							1
4					1		
5							1
6							1
7							

R

$= D \cdot$

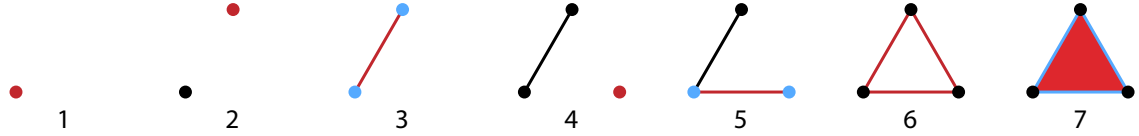
	1	2	3	4	5	6	7
1	1						
2		1					
3			1			1	
4				1			
5					1	1	
6						1	
7							1

V

Matrix reduction

$$
\underbrace{\begin{array}{c|ccccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \hline
1 & & & 1 & & 1 & & \\
2 & & & 1 & & & & \\
3 & & & & & & & 1 \\
4 & & & & & 1 & & \\
5 & & & & & & & 1 \\
6 & & & & & & & 1 \\
7 & & & & & & &
\end{array}}_{R}
= D \cdot
\underbrace{\begin{array}{c|ccccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \hline
1 & 1 & & & & & & \\
2 & & 1 & & & & & \\
3 & & & 1 & & & 1 & \\
4 & & & & 1 & & & \\
5 & & & & & 1 & 1 & \\
6 & & & & & & 1 & \\
7 & & & & & & & 1
\end{array}}_{V}
$$

Matrix reduction



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

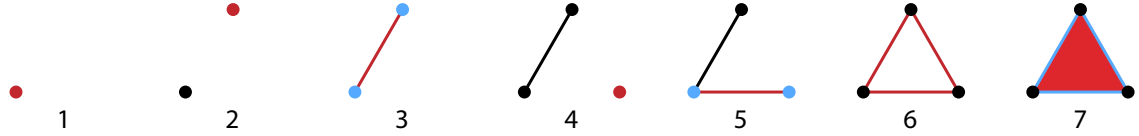
R

$= D \cdot$

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

V

Matrix reduction



R

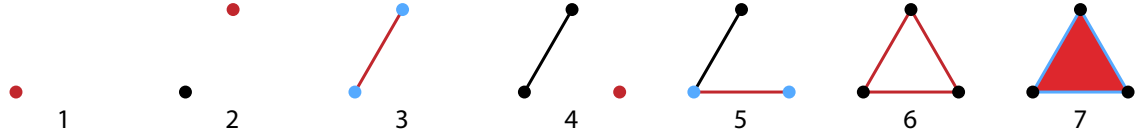
	1	2	3	4	5	6	7
1			1		1		
2			1				
3							1
4					1		
5							1
6							1
7							

$= D \cdot$

V

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

Matrix reduction



R

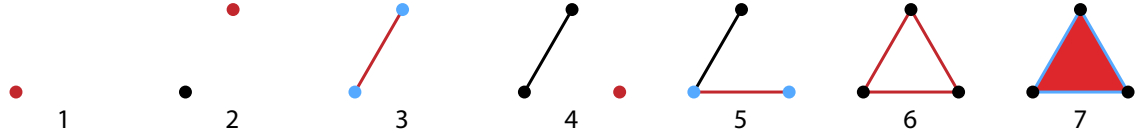
	1	2	3	4	5	6	7
1			1		1		
2			1				
3							1
4					1		
5							1
6							1
7							

$= D \cdot$

V

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

Matrix reduction



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

R

$= D \cdot$

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

V

Matrix reduction algorithm

Require: D: $I \times J$ matrix

Ensure: $R = D \cdot V$: reduced, V: regular upper triangular, P: persistence pairs, E: essential

$P := \varnothing$

$R := D$

$V := I$

for $j \in J$ in increasing order **do**

  while $\exists k < j$ with $\operatorname{PivotIndex} R_k = \operatorname{PivotIndex} R_j$ **do**

    $\lambda := \operatorname{PivotEntry} R_j / \operatorname{PivotEntry} R_k$

    $R_j := R_j - \lambda \cdot R_k$ ▷ elimintates $\operatorname{PivotEntry} R_j$

    $V_j := V_j - \lambda \cdot V_k$

  if $(i := \operatorname{PivotIndex} R_j) \neq 0$ **then**

    append (i, j) to P ▷ persistence interval $[i, j)$

  else

    append j to E ▷ persistence interval $[j, \infty)$

Compatible basis cycles

For a reduced boundary matrix $R = D \cdot V$, call

$$I_b = \{i : R_i = 0\} \qquad \textit{birth} \text{ indices ,}$$

$$I_d = \{j : R_j \neq 0\} \qquad \textit{death} \text{ indices,}$$

$$I_e = I_b \setminus \operatorname{PivotIndices} R \qquad \textit{essential} \text{ indices.}$$

(note: $i = \operatorname{PivotIndex} R_j$ implies $R_i = 0$, thus $\operatorname{PivotIndices} R \subseteq I_b$). Then

Compatible basis cycles

For a reduced boundary matrix $R = D \cdot V$, call

$$I_b = \{i : R_i = 0\} \qquad \textit{birth} \text{ indices ,}$$
$$I_d = \{j : R_j \neq 0\} \qquad \textit{death} \text{ indices,}$$
$$I_e = I_b \setminus \operatorname{PivotIndices} R \qquad \textit{essential} \text{ indices.}$$

(note: $i = \operatorname{PivotIndex} R_j$ implies $R_i = 0$, thus $\operatorname{PivotIndices} R \subseteq I_b$). Then

$$\Sigma_B = \{R_j \mid j \in I_d\} \qquad \text{is a basis of } B_*,$$

Compatible basis cycles

For a reduced boundary matrix $R = D \cdot V$, call

$$I_b = \{i : R_i = 0\} \qquad \textit{birth} \text{ indices ,}$$
$$I_d = \{j : R_j \neq 0\} \qquad \textit{death} \text{ indices,}$$
$$I_e = I_b \setminus \operatorname{PivotIndices} R \qquad \textit{essential} \text{ indices.}$$

(note: $i = \operatorname{PivotIndex} R_j$ implies $R_i = 0$, thus $\operatorname{PivotIndices} R \subseteq I_b$). Then

$$\Sigma_B = \{R_j \mid j \in I_d\} \qquad \text{is a basis of } B_*,$$
$$\Sigma_Z = \Sigma_B \cup \{V_i \mid i \in I_e\} \qquad \text{is a } \textit{compatible} \text{ basis of } Z_*.$$

Compatible basis cycles

For a reduced boundary matrix $R = D \cdot V$, call

$$I_b = \{i : R_i = 0\} \qquad \textit{birth} \text{ indices ,}$$
$$I_d = \{j : R_j \neq 0\} \qquad \textit{death} \text{ indices,}$$
$$I_e = I_b \setminus \operatorname{PivotIndices} R \qquad \textit{essential} \text{ indices.}$$

(note: $i = \operatorname{PivotIndex} R_j$ implies $R_i = 0$, thus $\operatorname{PivotIndices} R \subseteq I_b$). Then

$$\Sigma_B = \{R_j \mid j \in I_d\} \qquad \text{is a basis of } B_*,$$
$$\Sigma_Z = \Sigma_B \cup \{V_i \mid i \in I_e\} \qquad \text{is a } \textit{compatible} \text{ basis of } Z_*.$$

- Persistent homology is generated by the basis cycles Σ_Z.

Compatible basis cycles

For a reduced boundary matrix $R = D \cdot V$, call

$$I_b = \{i : R_i = 0\} \qquad \textit{birth} \text{ indices ,}$$
$$I_d = \{j : R_j \neq 0\} \qquad \textit{death} \text{ indices,}$$
$$I_e = I_b \setminus \operatorname{PivotIndices} R \qquad \textit{essential} \text{ indices.}$$

(note: $i = \operatorname{PivotIndex} R_j$ implies $R_i = 0$, thus $\operatorname{PivotIndices} R \subseteq I_b$). Then

$$\Sigma_B = \{R_j \mid j \in I_d\} \qquad \text{is a basis of } B_*,$$
$$\Sigma_Z = \Sigma_B \cup \{V_i \mid i \in I_e\} \qquad \text{is a } \textit{compatible} \text{ basis of } Z_*.$$

- Persistent homology is generated by the basis cycles Σ_Z.
- Persistence intervals: $\{[i,j) \mid i = \operatorname{PivotIndex} R_j\} \cup \{[i,\infty) \mid i \in I_e\}$

Compatible basis cycles

For a reduced boundary matrix $R = D \cdot V$, call

$$I_b = \{i : R_i = 0\} \qquad \textit{birth} \text{ indices ,}$$
$$I_d = \{j : R_j \neq 0\} \qquad \textit{death} \text{ indices,}$$
$$I_e = I_b \setminus \operatorname{PivotIndices} R \qquad \textit{essential} \text{ indices.}$$

(note: $i = \operatorname{PivotIndex} R_j$ implies $R_i = 0$, thus $\operatorname{PivotIndices} R \subseteq I_b$). Then

$$\Sigma_B = \{R_j \mid j \in I_d\} \qquad \text{is a basis of } B_*,$$
$$\Sigma_Z = \Sigma_B \cup \{V_i \mid i \in I_e\} \qquad \text{is a } \textit{compatible} \text{ basis of } Z_*.$$

- Persistent homology is generated by the basis cycles Σ_Z.
- Persistence intervals: $\{[i,j) \mid i = \operatorname{PivotIndex} R_j\} \cup \{[i,\infty) \mid i \in I_e\}$
- Columns with indices $\operatorname{PivotIndices} R$ (non-essential birth indices) not used at all

Optimizations

The four special ingredients of Ripser

Main improvements to the standard algorithm:

- Clearing inessential columns [Chen, Kerber 2011]
- Computing cohomology [de Silva et al. 2011]
- Implicit matrix reduction
- Apparent pairs

The four special ingredients of Ripser

Main improvements to the standard algorithm:

- Clearing inessential columns [Chen, Kerber 2011]
- Computing cohomology [de Silva et al. 2011]
- Implicit matrix reduction
- Apparent pairs

Lessons from previous work:

- Clearing and cohomology yield considerable speedup,
- but only when *both* are used in conjuction!

Clearing

Clearing non-essential positive columns

Recall:

- Columns with indices $\mathrm{PivotIndices}\, R$ (non-essential birth indices) not used at all

Idea [Chen, Kerber 2011]:

- Don't reduce those columns
- Use the fact that every boundary is a cycle

Clearing non-essential positive columns

Recall:

- Columns with indices $\mathrm{PivotIndices}\, R$ (non-essential birth indices) not used at all

Idea [Chen, Kerber 2011]:

- Don't reduce those columns
- Use the fact that every boundary is a cycle
  - Reduce matrices for $\partial_d : C_d \to C_{d-1}$ for $d = k+1, \ldots, 1$

Clearing non-essential positive columns

Recall:

- Columns with indices $\mathrm{PivotIndices}\, R$ (non-essential birth indices) not used at all

Idea [Chen, Kerber 2011]:

- Don't reduce those columns
- Use the fact that every boundary is a cycle
  - Reduce matrices for $\partial_d : C_d \to C_{d-1}$ for $d = k+1, \ldots, 1$
  - Whenever $i = \mathrm{PivotIndex}\, R_j$ (in the matrix for ∂_d):
    - Set V_i to R_j (in the matrix for ∂_{d-1})
    - Then $R_i = D \cdot V_i = D \cdot R_j = 0$

Clearing non-essential positive columns

Recall:

- Columns with indices $\mathrm{PivotIndices}\, R$ (non-essential birth indices) not used at all

Idea [Chen, Kerber 2011]:

- Don't reduce those columns
- Use the fact that every boundary is a cycle
  - Reduce matrices for $\partial_d : C_d \to C_{d-1}$ for $d = k+1, \dots, 1$
  - Whenever $i = \mathrm{PivotIndex}\, R_j$ (in the matrix for ∂_d):
    - Set V_i to R_j (in the matrix for ∂_{d-1})
    - Then $R_i = D \cdot V_i = D \cdot R_j = 0$

Note:

- reducing *birth* columns typically harder than death columns:
  - $O(j^2)$ (birth) vs. $O((j-i)^2)$ (death)
- clearing avoids all non-essential birth columns (many, hard),
- reducing only death columns (easy) and *essential* birth columns (few)

Counting homology column reductions

Number of columns in boundary matrix (n vertivces, homology up to dimension k):

$$\sum_{d=1}^{k+1} \underbrace{\binom{n}{d+1}}_{\text{total}} = \sum_{d=1}^{k+1} \underbrace{\binom{n-1}{d}}_{\text{death}} + \sum_{d=1}^{k+1} \underbrace{\binom{n-1}{d+1}}_{\text{birth}}$$

Counting homology column reductions

Number of columns in boundary matrix (n vertivces, homology up to dimension k):

$$\underbrace{\sum_{d=1}^{k+1}\binom{n}{d+1}}_{\text{total}} = \underbrace{\sum_{d=1}^{k+1}\binom{n-1}{d}}_{\text{death}} + \underbrace{\sum_{d=1}^{k+1}\binom{n-1}{d+1}}_{\text{birth}}$$

$$= \underbrace{\sum_{d=1}^{k+1}\binom{n-1}{d}}_{\text{death}} + \underbrace{\binom{n-1}{k+2}}_{\text{essential}} + \underbrace{\sum_{d=1}^{k}\binom{n-1}{d+1}}_{\text{cleared}}$$

Counting homology column reductions

Number of columns in boundary matrix (n vertivces, homology up to dimension k):

$$\underbrace{\sum_{d=1}^{k+1}\binom{n}{d+1}}_{\text{total}} = \underbrace{\sum_{d=1}^{k+1}\binom{n-1}{d}}_{\text{death}} + \underbrace{\sum_{d=1}^{k+1}\binom{n-1}{d+1}}_{\text{birth}}$$

$$= \underbrace{\sum_{d=1}^{k+1}\binom{n-1}{d}}_{\text{death}} + \underbrace{\binom{n-1}{k+2}}_{\text{essential}} + \underbrace{\sum_{d=1}^{k}\binom{n-1}{d+1}}_{\text{cleared}}$$

$k = 2, n = 192$: $\quad 56\,050\,096 = 1\,161\,471 + 53\,727\,345 + 1\,161\,280$

- Clearing didn't help much!

Cohomology

Persistent (co)homology

- Cohomology is the vector space dual of homology.
- Persistent homology and persistent cohomology have the same barcode.
- Persistent cohomology can be computed by reducing the transposed boundary matrix.

Persistent (co)homology

- Cohomology is the vector space dual of homology.
- Persistent homology and persistent cohomology have the same barcode.
- Persistent cohomology can be computed by reducing the transposed boundary matrix.

What's the deal?

Cohomology and clearing

- Cohomology allows for clearing starting from dimension 0

Cohomology and clearing

- Cohomology allows for clearing starting from dimension 0
- Persistence in dimension 0 has special algorithms
  - Kruskal's algorithm for minimum spanning tree
  - union-find data structure

Counting cohomology column reductions

Consider K: $k+1$-skeleton of $n-1$-simplex, Rips filtration

- Number of columns in coboundary matrix:

$$\underbrace{\sum_{d=0}^{k} \binom{n}{d+1}}_{\text{total}} = \underbrace{\sum_{d=0}^{k} \binom{n-1}{d+1}}_{\text{death}} + \underbrace{\sum_{d=0}^{k} \binom{n-1}{d}}_{\text{birth}}$$

Counting cohomology column reductions

Consider K: $k+1$-skeleton of $n-1$-simplex, Rips filtration

- Number of columns in coboundary matrix:

$$\underbrace{\sum_{d=0}^{k}\binom{n}{d+1}}_{\text{total}} = \underbrace{\sum_{d=0}^{k}\binom{n-1}{d+1}}_{\text{death}} + \underbrace{\sum_{d=0}^{k}\binom{n-1}{d}}_{\text{birth}}$$

$$= \underbrace{\sum_{d=0}^{k}\binom{n-1}{d+1}}_{\text{death}} + \underbrace{\binom{n-1}{0}}_{\text{essential}} + \underbrace{\sum_{d=1}^{k}\binom{n-1}{d}}_{\text{cleared}}$$

Counting cohomology column reductions

Consider K: $k+1$-skeleton of $n-1$-simplex, Rips filtration

- Number of columns in coboundary matrix:

$$\underbrace{\sum_{d=0}^{k}\binom{n}{d+1}}_{\text{total}} = \underbrace{\sum_{d=0}^{k}\binom{n-1}{d+1}}_{\text{death}} + \underbrace{\sum_{d=0}^{k}\binom{n-1}{d}}_{\text{birth}}$$

$$= \underbrace{\sum_{d=0}^{k}\binom{n-1}{d+1}}_{\text{death}} + \underbrace{\binom{n-1}{0}}_{\text{essential}} + \underbrace{\sum_{d=1}^{k}\binom{n-1}{d}}_{\text{cleared}}$$

- $k=2, n=192$: $1\,179\,808 = 1\,161\,471 + 1 + 18\,336$

Observations

When reducing $R = D \cdot V$, for a typical input:

- V has very few off-diagonal entries
- most death columns of D are already reduced

Observations

When reducing $R = D \cdot V$, for a typical input:

- V has very few off-diagonal entries
- most death columns of D are already reduced

Previous example ($k = 2, n = 192$):

- Only $79 + 42 + 1 = 122$ out of $192 + 18\,145 + 1\,143\,135 = 1\,161\,471$ columns are actually modified in matrix reduction

Implicit matrix reduction

Implicit matrix reduction

Standard approach:

- Boundary matrix D for filtration-ordered basis
  - Explicitly generated and stored in memory

Implicit matrix reduction

Standard approach:

- Boundary matrix D for filtration-ordered basis
  - Explicitly generated and stored in memory
- Matrix reduction: store only reduced matrix R
  - Initialize matrix as D, reduce to R by column operations

Implicit matrix reduction

Standard approach:

- Boundary matrix D for filtration-ordered basis
  - Explicitly generated and stored in memory
- Matrix reduction: store only reduced matrix R
  - Initialize matrix as D, reduce to R by column operations

Approach for Ripser:

- Boundary matrix D for lexicographically ordered basis
  - Columns recomputed on the fly

Implicit matrix reduction

Standard approach:

- Boundary matrix D for filtration-ordered basis
  - Explicitly generated and stored in memory
- Matrix reduction: store only reduced matrix R
  - Initialize matrix as D, reduce to R by column operations

Approach for Ripser:

- Boundary matrix D for lexicographically ordered basis
  - Columns recomputed on the fly
- Matrix reduction: store only coefficient matrix V
  - Columns of $R_j = D \cdot V_j$ recomputed on the fly
  - Typically, V is much sparser and smaller than R

Implicit boundary matrix reduction algorithm

Require: D: $I \times J$ matrix

Ensure: $R = D \cdot V$: reduced, V: regular upper triangular, P: persistence pairs, E: essential

$P := \varnothing$

$R := D$

$V := I$

for $j \in J$ in increasing order **do**

  while $\exists k < j$ with $\operatorname{PivotIndex} R_k = \operatorname{PivotIndex} R_j$ **do**

    $\lambda := \operatorname{PivotEntry} R_j / \operatorname{PivotEntry} R_k$

    $R_j := R_j - \lambda \cdot R_k$ ▷ elimintates $\operatorname{PivotEntry} R_j$

    $V_j := V_j - \lambda \cdot V_k$

  if $(i := \operatorname{PivotIndex} R_j) \neq 0$ **then**

    append (i, j) to P ▷ persistence interval $[i, j)$

  else

    append j to E ▷ persistence interval $[j, \infty)$

Implicit boundary matrix reduction algorithm

Require: D: $I \times J$ matrix

Ensure: $R = D \cdot V$: reduced, V: regular upper triangular, P: persistence pairs, E: essential

$P := \varnothing$

$R := D$

$V := I$

for $j \in J$ in increasing order **do**

  while $\exists k < j$ with $\operatorname{PivotIndex} R_k = \operatorname{PivotIndex} R_j$ **do**

    $\lambda := \operatorname{PivotEntry} R_j / \operatorname{PivotEntry} R_k$

    $R_j := R_j - \lambda \cdot R_k$ ▷ elimintates $\operatorname{PivotEntry} R_j$

    $V_j := V_j - \lambda \cdot V_k$

  if $(i := \operatorname{PivotIndex} R_j) \neq 0$ **then**

    append (i, j) to P ▷ persistence interval $[i, j)$

  else

    append j to E ▷ persistence interval $[j, \infty)$

Implicit boundary matrix reduction algorithm

Require: D: $I \times J$ matrix

Ensure: $R = D \cdot V$: reduced, V: regular upper triangular, P: persistence pairs, E: essential

$P := \varnothing$

$R := D$

$V := I$

for $j \in J$ in increasing order **do**

  while $\exists k < j$ with $\operatorname{PivotIndex} R_k = \operatorname{PivotIndex} R_j$ **do**

    $\lambda := \operatorname{PivotEntry} R_j / \operatorname{PivotEntry} R_k$

    $R_j := R_j - \lambda \cdot D \cdot V_k$ ▷ elimintates $\operatorname{PivotEntry} R_j$

    $V_j := V_j - \lambda \cdot V_k$

  if $(i := \operatorname{PivotIndex} R_j) \neq 0$ **then**

    append (i, j) to P ▷ persistence interval $[i, j)$

  else

    append j to E ▷ persistence interval $[j, \infty)$

Filtrations and refinements

Simplexwise filtrations

Call a filtration $K_\bullet = (K_i)_{i \in \{1,\dots,n\}}$ of simplicial complexes *simplexwise* if all $K_i \neq \varnothing$ are of the form $K_i = K_{i-1} \cup \{\sigma_i\}$ for some $\sigma_i \in K$.

Simplexwise filtrations

Call a filtration $K_\bullet = (K_i)_{i \in \{1,\dots,n\}}$ of simplicial complexes *simplexwise* if all $K_i \neq \varnothing$ are of the form $K_i = K_{i-1} \cup \{\sigma_i\}$ for some $\sigma_i \in K$.

Note:

- These properties are natural assumptions for computation.
- The *Vietoris–Rips filtration* (indexed by $\mathbb{R}$) is not simplexwise.
- To compute Vietoris–Rips persistence, we have to reindex and refine.

Simplexwise filtrations

Call a filtration $K_\bullet = (K_i)_{i\in\{1,\dots,n\}}$ of simplicial complexes *simplexwise* if all $K_i \neq \varnothing$ are of the form $K_i = K_{i-1} \cup \{\sigma_i\}$ for some $\sigma_i \in K$.
Note:

- These properties are natural assumptions for computation.
- The *Vietoris–Rips filtration* (indexed by $\mathbb{R}$) is not simplexwise.
- To compute Vietoris–Rips persistence, we have to reindex and refine.

Ripser use the *lexicographic refinement*: simplices ordered by

- diameter, then
- dimension, then
- (decreasing) lexicographic order of vertices $\{v_{i_k}, \dots, v_{i_0}\}$
  (induced by fixed total order $v_0 < \cdots < v_{n-1}$ on vertices)

Enumerating (co)faces in lexicographic order

There is an order-preserving bijection

$$\begin{array}{ccccccccc} [v_2, v_1, v_0] & [v_3, v_1, v_0] & [v_3, v_2, v_0] & [v_3, v_2, v_1] & [v_4, v_1, v_0] & \cdots & [v_{n-3}, v_{n-2}, v_{n-1}] \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & & \downarrow \\ 0 & 1 & 2 & 3 & 4 & \cdots & \binom{n}{k+1} - 1 \end{array}$$

from k-simplices (ordered lexicographically) to natural numbers (called *combinatorial number system*).

Enumerating (co)faces in lexicographic order

There is an order-preserving bijection

$$\begin{array}{ccccccccc} [v_2, v_1, v_0] & [v_3, v_1, v_0] & [v_3, v_2, v_0] & [v_3, v_2, v_1] & [v_4, v_1, v_0] & \cdots & [v_{n-3}, v_{n-2}, v_{n-1}] \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & & \downarrow \\ 0 & 1 & 2 & 3 & 4 & \cdots & \binom{n}{k+1} - 1 \end{array}$$

from k-simplices (ordered lexicographically) to natural numbers (called *combinatorial number system*).

- One direction is given by

$$[v_{i_k}, \ldots, v_{i_0}] \mapsto \sum_{j=0}^{k} \binom{i_j}{j+1}$$

- The other direction can also be computed quickly
- Using this bijection, faces and cofaces can be efficiently enumerated in (decreasing) lexicographic order

Apparent pairs

Apparent pairs

Definition

In a simplexwise filtration, a pair of simplices (σ, τ) is an *apparent* pair if

- σ is the youngest facet of τ, and
- τ is the oldest cofacet of σ.

Apparent pairs

Definition

In a simplexwise filtration, a pair of simplices (σ, τ) is an *apparent* pair if

- σ is the youngest facet of τ, and
- τ is the oldest cofacet of σ.

Proposition

The apparent pairs are persistence pairs.

Apparent pairs

Definition

In a simplexwise filtration, a pair of simplices (σ, τ) is an *apparent* pair if

- σ is the youngest facet of τ, and
- τ is the oldest cofacet of σ.

Proposition

The apparent pairs are persistence pairs.

- The apparent pairs also form a *discrete gradient* (in the sense of discrete Morse theory)

Apparent pairs

Definition

In a simplexwise filtration, a pair of simplices (σ, τ) is an *apparent* pair if

- σ is the youngest facet of τ, and
- τ is the oldest cofacet of σ.

Proposition

The apparent pairs are persistence pairs.

- The apparent pairs also form a *discrete gradient* (in the sense of discrete Morse theory)
- Can be identified *without* enumerating all cofaces of σ (shortcut for computation)

Speedup from apparent pairs shortcut

Previous example ($k = 2, n = 192$):

- Only $53 + 601 = 654$ out of $18\,337 + 1\,179\,808$ columns fully constructed (without shortcut)
- Only $164\,214 + 3\,392\,039 = 3\,556\,253$ out of $3\,447\,550 + 216\,052\,515 = 219\,500\,065$ nonzero entries of the coboundary matrix are enumerated

Speedup: $1.2\,s$ vs. $5.6\,s$ (factor 4.7)

	n	d	non-zero pairs	non-shortcut pairs	persistence pairs
sphere3	192	1	53	53	18 145
		2	1	601	1 143 135
o3	4 096	1	2466	2466	230 051
		2	811	15 829	4 112 971
		3	33	123 035	39 738 338
torus4	50 000	1	14 006	14 006	2 192 209
		2	136	60 603	37 145 478
dragon	2 000	1	576	576	1 386 646
fractal-r	512	1	438	438	122 324
		2	659	6612	18 979 158

	n	d	non-zero pairs	non-shortcut pairs	persistence pairs
sphere3	192	1	53	53	18 145
		2	1	601	1 143 135
o3	4 096	1	2466	2466	230 051
		2	811	15 829	4 112 971
		3	33	123 035	39 738 338
torus4	50 000	1	14 006	14 006	2 192 209
		2	136	60 603	37 145 478
dragon	2 000	1	576	576	1 386 646
fractal-r	512	1	438	438	122 324
		2	659	6612	18 979 158

Theorem

Let (X, d) be a finite metric space with distinct pairwise distances, and consider the simplexwise refinement of the Vietoris–Rips filtration for X. In dimension 1, the apparent pairs are precisely the zero persistence pairs.

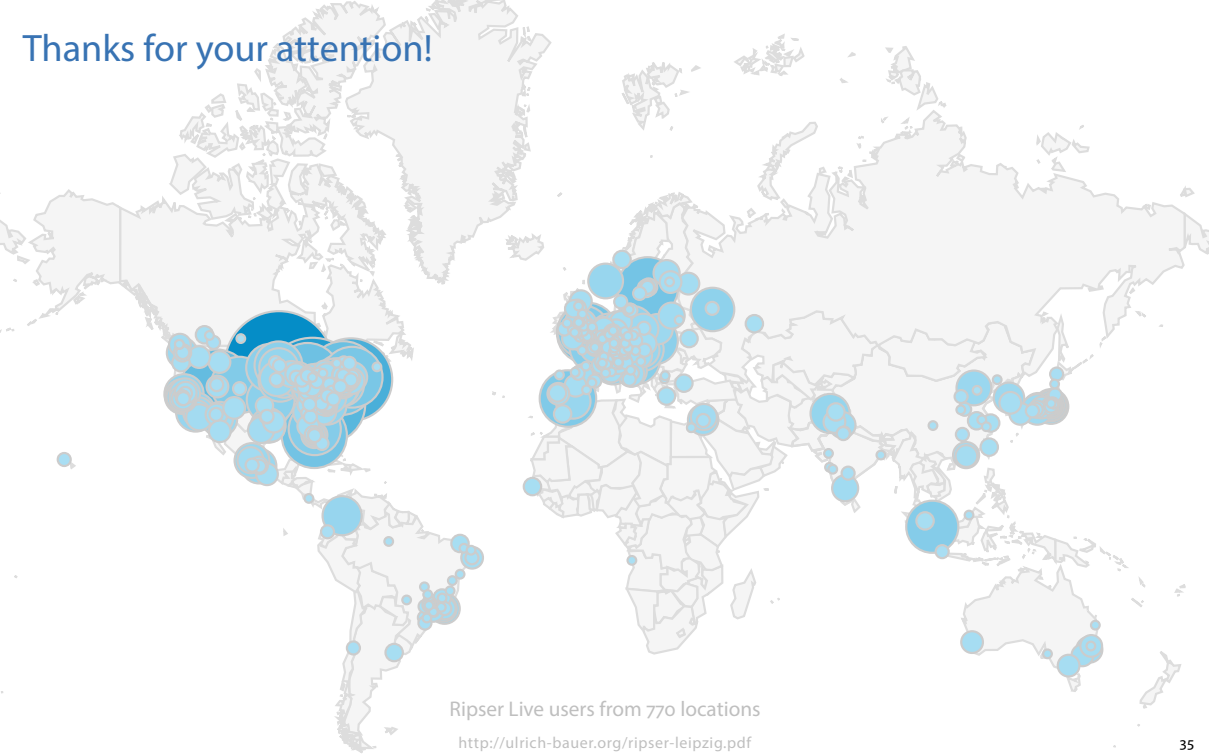
	n	d	non-zero pairs	non-shortcut pairs	persistence pairs
sphere3	192	1	53	53	18 145
		2	1	601	1 143 135
o3	4 096	1	2466	2466	230 051
		2	811	15 829	4 112 971
		3	33	123 035	39 738 338
torus4	50 000	1	14 006	14 006	2 192 209
		2	136	60 603	37 145 478
dragon	2 000	1	576	576	1 386 646
fractal-r	512	1	438	438	122 324
		2	659	6612	18 979 158

Theorem

Let (X, d) be a finite metric space with distinct pairwise distances,
and consider the simplexwise refinement of the Vietoris–Rips filtration for X.
In dimension 1, the apparent pairs are precisely the zero persistence pairs.

- The computation is *output-sensitive*:
  Only the simplices that contribute to the barcode also incur a computational cost.

Thanks for your attention!



Ripser Live users from 770 locations

http://ulrich-bauer.org/ripser-leipzig.pdf