Computing Height Persistence and Homology Generators in $\mathbb{R}^3$ Efficienetly

Tamal K. Dey

Department of Computer Science and Engineering
The Ohio State University

Persistence

- Simplicial filtration:

$$\emptyset = K_0 \subseteq K_1 \subseteq \cdots \subseteq K_n = K$$

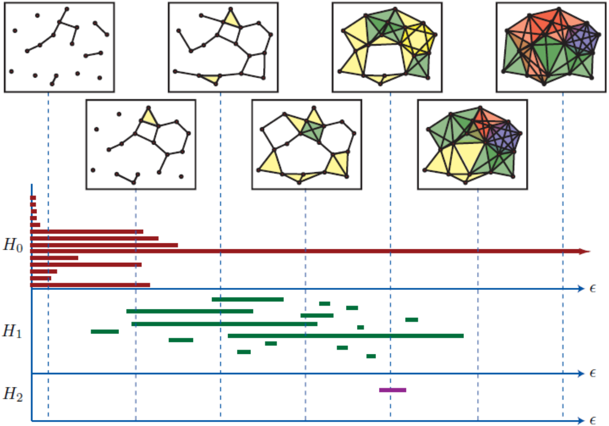
- Persistence module:

$$0 \to \mathrm{H}_p(K_1) \to \cdots \to \mathrm{H}_p(K_n) = \mathrm{H}_p(K).$$

- Birth and Death of homology classes

Bar Codes

- birth-death and bar codes

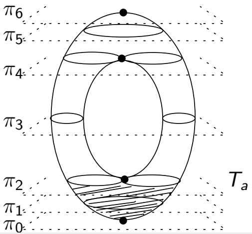


Height Persistence

- $\mathbb{T} \subseteq \mathbb{R}^3$; height $z : \mathbb{T} \to \mathbb{R}$
- $\mathbb{T}_a = z^{-1}(-\infty, a]$, the sublevel set
- $\mathbb{T}_a \subseteq \mathbb{T}_b$ for $a \leq b$ provides inclusion map $\iota : \mathbb{T}_a \to \mathbb{T}_b$
- Induced map $\iota_* : \mathsf{H}_p(\mathbb{T}_a) \to \mathsf{H}_p(\mathbb{T}_b)$ giving the sequence
  $0 \to \mathsf{H}_p(\mathbb{T}_{a_1}) \to \mathsf{H}_p(\mathbb{T}_{a_2}) \to \cdots \to \mathsf{H}_p(\mathbb{T}_{a_n}) \to \mathsf{H}_p(\mathbb{T})$
- Persistent homology classes: Image of $f_p^{ij} : \mathsf{H}_p(\mathbb{T}_{a_i}) \to \mathsf{H}_p(\mathbb{T}_{a_j})$

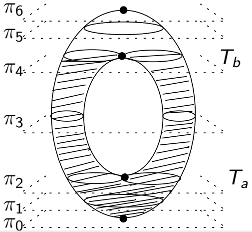
Height Persistence

- $\mathbb{T} \subseteq \mathbb{R}^3$; height $z : \mathbb{T} \to \mathbb{R}$
- $\mathbb{T}_a = z^{-1}(-\infty, a]$, the sublevel set
- $\mathbb{T}_a \subseteq \mathbb{T}_b$ for $a \leq b$ provides inclusion map $\iota : \mathbb{T}_a \to \mathbb{T}_b$
- Induced map $\iota_* : \mathrm{H}_p(\mathbb{T}_a) \to \mathrm{H}_p(\mathbb{T}_b)$ giving the sequence
  $0 \to \mathrm{H}_p(\mathbb{T}_{a_1}) \to \mathrm{H}_p(\mathbb{T}_{a_2}) \to \cdots \to \mathrm{H}_p(\mathbb{T}_{a_n}) \to \mathrm{H}_p(\mathbb{T})$
- Persistent homology classes: Image of $f_p^{ij} : \mathrm{H}_p(\mathbb{T}_{a_i}) \to \mathrm{H}_p(\mathbb{T}_{a_j})$



Height Persistence

- $\mathbb{T} \subseteq \mathbb{R}^3$; height $z : \mathbb{T} \to \mathbb{R}$
- $\mathbb{T}_a = z^{-1}(-\infty, a]$, the sublevel set
- $\mathbb{T}_a \subseteq \mathbb{T}_b$ for $a \leq b$ provides inclusion map $\iota : \mathbb{T}_a \to \mathbb{T}_b$
- Induced map $\iota_* : \mathrm{H}_p(\mathbb{T}_a) \to \mathrm{H}_p(\mathbb{T}_b)$ giving the sequence
  $0 \to \mathrm{H}_p(\mathbb{T}_{a_1}) \to \mathrm{H}_p(\mathbb{T}_{a_2}) \to \cdots \to \mathrm{H}_p(\mathbb{T}_{a_n}) \to \mathrm{H}_p(\mathbb{T})$
- Persistent homology classes: Image of $f_p^{ij} : \mathrm{H}_p(\mathbb{T}_{a_i}) \to \mathrm{H}_p(\mathbb{T}_{a_j})$



Previous work

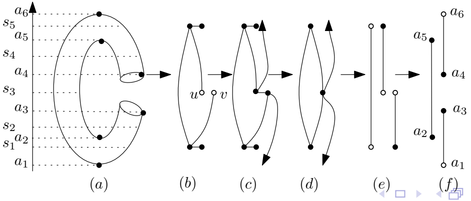
- Classical persistence algorithm [ELZ01] runs in matrix multiplication time $O(n^{\omega}) = O(n^{2.373})$ [SMJ11]
- Computing Betti numbers for 2-complexes in $\mathbb{R}^4$ is as hard as matrix rank computation [EP15]
- Special cases of graphs, surfaces, H_{p-1}-persistence for p-complex in $\mathbb{R}^p$ in $O(n \log n)$ time; reduces to min. spanning tree

This work:

- $O(n \log n)$ algorithm for height persistence $z : \mathbb{T} \to \mathbb{R}$, $\mathbb{T} \subseteq \mathbb{R}^3$
- $O(n \log n + k)$ algorithm for computing H_1-generators for $\mathcal{K} \subseteq \mathbb{R}^3$

Approach

- Use zigzag level set persistence for $z : \mathbb{T} \rightarrow \mathbb{R}$ for a subset of bars of height persisitence
  - Track level sets to construct a barcode graph B; $O(n \log n)$ time
  - Prove that reduced H_0-persistence for height on B is equivalent to H_1-persistence for height on $\mathbb{T}$
  - Extract bars for the height on B in $O(n \log n)$ time by modifying an algorithm of [AEHW05]
- Rest of the bars for $z : \mathbb{T} \rightarrow \mathbb{R}$ are computed from the Reeb graph $R_z(\mathbb{T})$; $O(n \log n)$ time [Parsa 12]



Sub-level and Zigzag level set persistence

- $z : \mathbb{T} \to \mathbb{R}$ has homological critical values $-\infty = a_0 < a_1 < a_2 < \ldots < a_m < a_{m+1} = \infty$
- $\{s_i\}$ of z interleaving with its critical values:

$$a_0 < s_0 < a_1 < s_1 < \ldots < a_m < s_m < a_{m+1}$$

- sub-level sets $\mathbb{T}_{[0,r]} := z^{-1}(-\infty, r]$. Sub-level set persistence module:

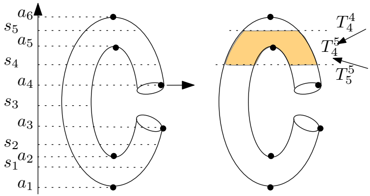
$$\begin{aligned} \mathcal{SL}(f, \mathbb{T}) &: \mathbb{T}_{[0,a_1]} \to \mathbb{T}_{[0,s_1]} \cdots \to \mathbb{T}_{[0,s_m]} \to \mathbb{T}_{[0,a_{m+1}]} \\ \mathrm{H}_p(\mathcal{SL}(f, \mathbb{T})) &: \mathrm{H}_p(\mathbb{T}_{[0,a_1]}) \to \mathrm{H}_p(\mathbb{T}_{[0,s_1]}) \cdots \to \mathrm{H}_p(\mathbb{T}_{[0,a_{m+1}]}) \end{aligned}$$

Sub-level and Zigzag level set persistence

- $z : \mathbb{T} \to \mathbb{R}$ has homological critical values
  $-\infty = a_0 < a_1 < a_2 < \ldots < a_m < a_{m+1} = \infty$
- $\{s_i\}$ of z interleaving with its critical values:

$$a_0 < s_0 < a_1 < s_1 < \ldots < a_m < s_m < a_{m+1}$$

- Interval sets $\mathbb{T}_i^j := \mathbb{T}_{[s_i, s_j]}$
  Zigzag level set persistence module:



Sub-level and Zigzag level set persistence

- $z : \mathbb{T} \to \mathbb{R}$ has homological critical values $-\infty = a_0 < a_1 < a_2 < \ldots < a_m < a_{m+1} = \infty$
- $\{s_i\}$ of z interleaving with its critical values:

$$a_0 < s_0 < a_1 < s_1 < \ldots < a_m < s_m < a_{m+1}$$

- Interval sets $\mathbb{T}_i^j := \mathbb{T}_{[s_i, s_j]}$
  Zigzag level set persistence module:

$$\mathcal{L}(f, \mathbb{T}) : \mathbb{T}_0^0 \to \mathbb{T}_0^1 \leftarrow \mathbb{T}_1^1 \to \mathbb{T}_1^2 \cdots \to \mathbb{T}_{m-1}^m \leftarrow \mathbb{T}_m^m$$
$$\mathsf{H}_p(\mathcal{L}(f, \mathbb{T})) : \mathsf{H}_p(\mathbb{T}_0^0) \to \mathsf{H}_p(\mathbb{T}_0^1) \leftarrow \mathsf{H}_p(\mathbb{T}_1^1) \to \cdots \leftarrow \mathsf{H}_p(\mathbb{T}_m^m)$$

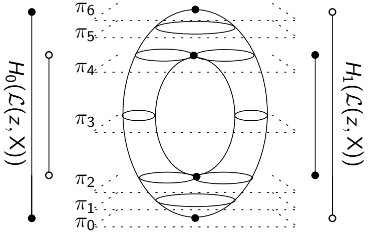
Bars

- By Quiver theory $\mathrm{H}_p(\mathcal{L}(f,\mathbb{T}))$ and $\mathrm{H}_p(\mathcal{SL}(f,\mathbb{T}))$ decomposes into intervals:

$$\mathcal{I}_{[b,d]} : I_1 \leftrightarrow I_2 \cdots \leftrightarrow I_m,\ b, d \in \{a_i, s_i\}$$

- Four types of bars:

- $[a_i, a_j]$: closed-closed
- $[a_i, s_j] \Leftrightarrow [a_i, a_{j+1})$: closed-open
- $[s_i, a_j] \Leftrightarrow (a_i, a_j]$: open-closed
- $[s_i, s_j] \Leftrightarrow (a_i, a_{j+1})$: open-open



Link between sublevel and level set persistence

Theorem (Burghlea,D.)

1. *$[a_i, a_j)$ is a bar for $\mathrm{H}_p(\mathcal{SL}(f, \mathbb{T}))$ iff it is so for $\mathrm{H}_p(\mathcal{L}(f, \mathbb{T}))$,*
2. *$[a_i, \infty)$ is a bar for $\mathrm{H}_p(\mathcal{SL}(f, \mathbb{T}))$ iff either $[a_i, a_j]$ is a closed-closed bar for $\mathrm{H}_p(\mathcal{L}(f, \mathbb{T}))$ for some $a_j > a_i$, or (a_j, a_i) is an open-open bar for $\mathrm{H}_{p-1}(\mathcal{L}(f, \mathbb{T}))$ for some $a_j < a_i$.*

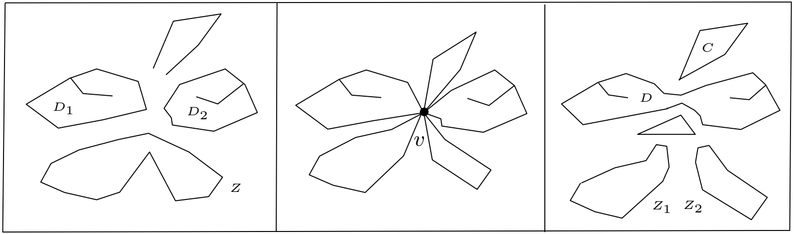
- Compute bars for $\mathrm{H}_1(\mathcal{L}(z, \mathbb{T}))$ to obtaine the closed-open and closed-closed bars.
- Compute bars for $\mathrm{H}_0(\mathcal{L}(z, \mathbb{T}))$ to obtain the open-open bars: equivalent to computing H_0-persistence on the Reeb graph $\mathrm{R}b_z(\mathbb{T})$

Tracking Basis in Level Sets

- Level set $G_r = z^{-1}(r)$ is a planar graph
- Combinatorially G_r changes passing through a vertex
- Primary and secondary cycles

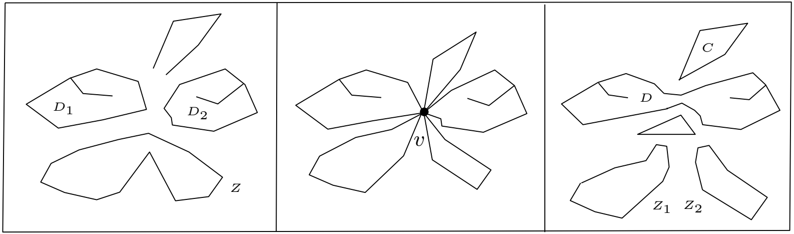
Tracking Basis in Level Sets

- Level set $G_r = z^{-1}(r)$ is a planar graph
- Combinatorially G_r changes passing through a vertex
- Primary and secondary cycles



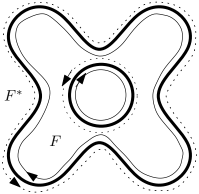
Tracking Basis in Level Sets

- Level set $G_r = z^{-1}(r)$ is a planar graph
- Combinatorially G_r changes passing through a vertex
- Primary and secondary cycles



Tracking Basis in Level Sets

- Level set $G_r = z^{-1}(r)$ is a planar graph
- Combinatorially G_r changes passing through a vertex
- Primary and secondary cycles



Primary cycle basis

Proposition

The classes of unoriented cycles $\{[C_F] \mid C_{\vec{F}}$ *is primary*$\}$ *form a basis of* $\mathrm{H}_1(G_r)$.

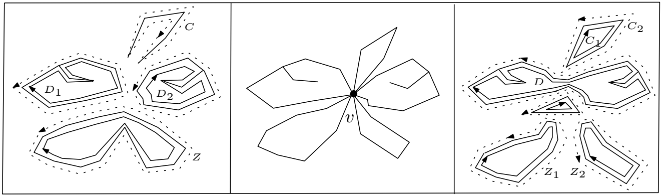
- Track primary and secondary cycles

Primary cycle basis

Proposition

The classes of unoriented cycles $\{[C_F] \mid C_{\vec{F}}$ *is primary*$\}$ *form a basis of* $\mathrm{H}_1(G_r)$.

- Track primary and secondary cycles

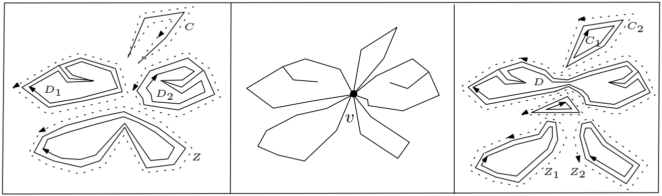


Primary cycle basis

Proposition

The classes of unoriented cycles $\{[C_F] \mid C_{\overrightarrow{F}}$ *is primary*$\}$ *form a basis of* $\mathsf{H}_1(G_r)$.

- Track primary and secondary cycles

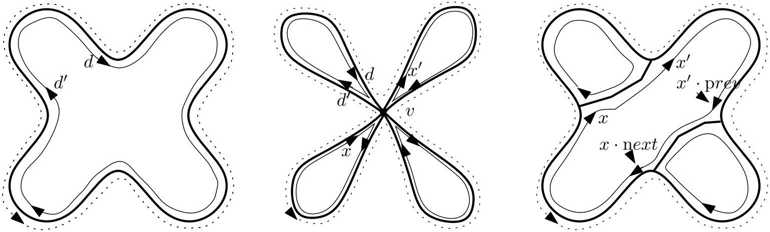


Primary cycle basis

Proposition

The classes of unoriented cycles $\{[C_F] \mid C_{\vec{F}}$ *is primary*$\}$ *form a basis of* $\mathrm{H}_1(G_r)$.

- Track primary and secondary cycles

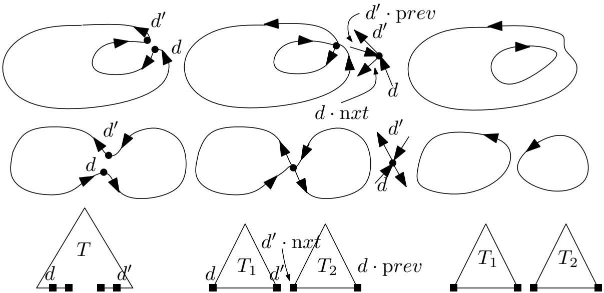


Tracking through vertex

- $a_i = z(v_i)$;
- Track primary/secondary cycles going from $G_{s_{i-1}}$ to G_{a_i} and then to G_{s_i}.
- Directed edges d constitute cycles
- Cycles are maintained by cycle trees (AVL or 2-3 trees)

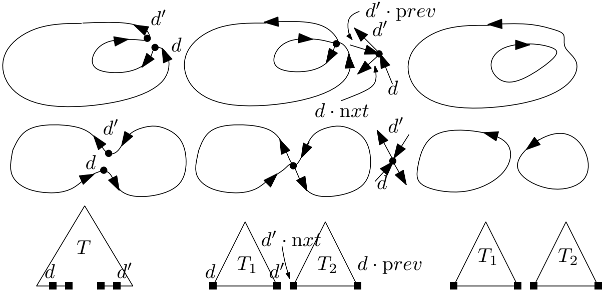
Tracking through vertex

- $a_i = z(v_i)$;
- Track primary/secondary cycles going from $G_{s_{i-1}}$ to G_{a_i} and then to G_{s_i}.
- Directed edges d constitute cycles
- Cycles are maintained by cycle trees (AVL or 2-3 trees)



Tracking through vertex

- $a_i = z(v_i)$;
- Track primary/secondary cycles going from $G_{s_{i-1}}$ to G_{a_i} and then to G_{s_i}.
- Directed edges d constitute cycles
- Cycles are maintained by cycle trees (AVL or 2-3 trees)

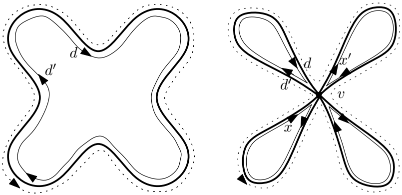


Cycle tree operations

- Operations Split, Merge, Find in $O(\log n)$ time
  - $\textsc{Find}(d)$ returns T
  - $\textsc{Split}(T, d)$ splits T into T_1 and T_2 at d
  - $\textsc{Join}(T_1, T_2)$ merges T_1 and T_2 into a single tree
  - $\textsc{Insert}(T, d)$, $\textsc{Delete}(T, d)$...
- Sweeping through $v_1, v_2, \ldots, v_n$ of $\mathcal{K}$ takes $O(n \log n)$ time

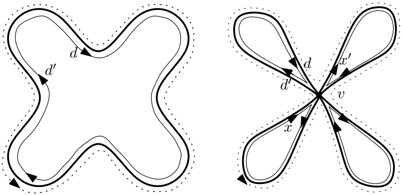
Updating $G_{s_{i-1}}$ to G_{a_i}

- Edges of $G_{s_{i-1}}$ get contracted to vertex v_i.
- Primary/secondary cycle c may split reaching a_i



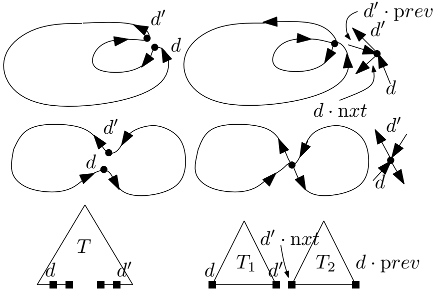
Updating $G_{s_{i-1}}$ to G_{a_i}

- Edges of $G_{s_{i-1}}$ get contracted to vertex v_i.
- Primary/secondary cycle c may split reaching a_i
  - Depending on cases, we can decide types of new cycles



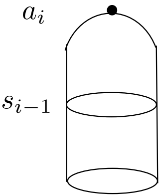
Updating $G_{s_{i-1}}$ to G_{a_i}

- Edges of $G_{s_{i-1}}$ get contracted to vertex v_i.
- Primary/secondary cycle c may split reaching a_i
  - Depending on cases, we can decide types of new cycles



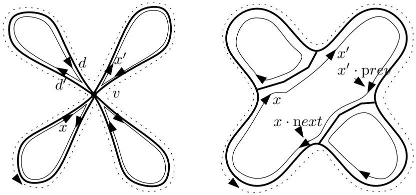
Updating $G_{s_{i-1}}$ to G_{a_i}

- Edges of $G_{s_{i-1}}$ get contracted to vertex v_i.
- Primary/secondary cycle c may split reaching a_i
    - Depending on cases, we can decide types of new cycles
- c ceases to exist



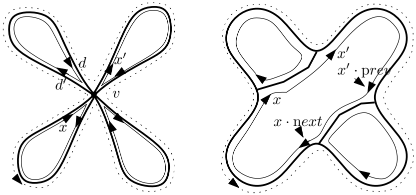
Updating G_{a_i} to G_{s_i}

- Vertex v_i expands into edges of G_{s_i}
- Primary/secondary cycle c may merge after a_i



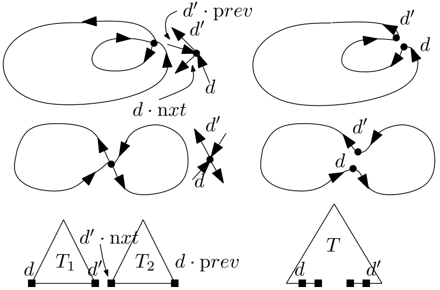
Updating G_{a_i} to G_{s_i}

- Vertex v_i expands into edges of G_{s_i}
- Primary/secondary cycle c may merge after a_i
  - Depending on cases, we can decide types of new cycles



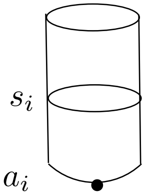
Updating G_{a_i} to G_{s_i}

- Vertex v_i expands into edges of G_{s_i}
- Primary/secondary cycle c may merge after a_i
  - Depending on cases, we can decide types of new cycles



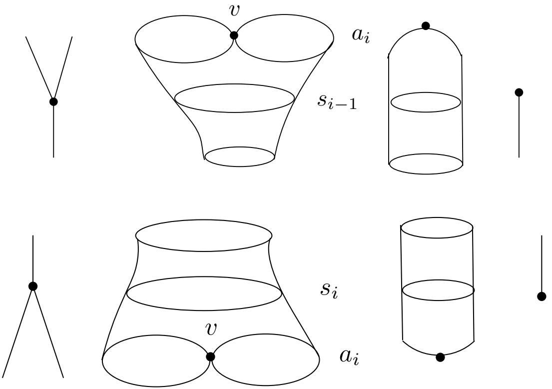
Updating G_{a_i} to G_{s_i}

- Vertex v_i expands into edges of G_{s_i}
- Primary/secondary cycle c may merge after a_i
  - Depending on cases, we can decide types of new cycles
- New cycle c is born



Updates of Barcode Graph

- $G_{s_{i-1}}$ to G_{a_i}:
  - Split
  - Death
- G_{a_i} to G_{s_i}:
  - Merge
  - Birth

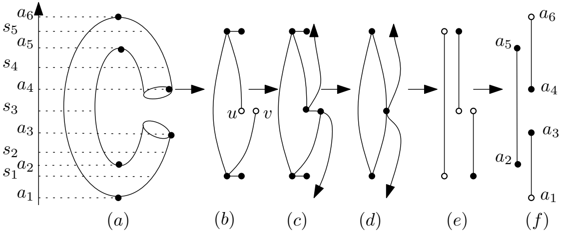


Barcode graph and zigzag persistence

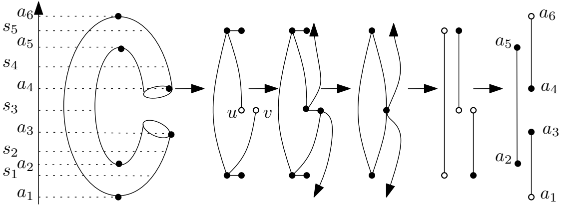
- critical values of $z : R \to \mathbb{R}$ are $\{s_1, s_2, \ldots, s_{m-1}\}$; same for $\mathcal{K}$ are $\{a_1, \ldots, a_m\}$.
- $R_i^j = R_{[a_i, a_j]}$ and $\mathcal{K}_i^j = |\mathcal{K}|_{[s_i, s_j]}$

$$\mathrm{H}_0(\mathcal{L}(z,R)) : \mathrm{H}_0(R_1^1) \to \mathrm{H}_0(R_1^2) \leftarrow \mathrm{H}_0(R_2^2) \to \cdots \leftarrow \mathrm{H}_0(R_m^m)$$

$$\mathrm{H}_1(\mathcal{L}(z,|\mathcal{K}|)) : \mathrm{H}_1(\mathcal{K}_0^0) \to \mathrm{H}_1(\mathcal{K}_0^1) \leftarrow \mathrm{H}_1(\mathcal{K}_1^1) \to \cdots \leftarrow \mathrm{H}_1(\mathcal{K}_m^m)$$



Barcode graph and zigzag persistence



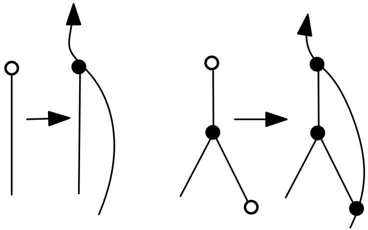
(a) (b) (c) (d) (e) (f)

Proposition

$\mathsf{H}_1(|\mathcal{K}|_{a_i}) \cong \mathsf{H}_0(R_{a_i})$ *and* $\mathsf{H}_1(|\mathcal{K}|_{s_i}) \cong \mathsf{H}_0(R_{s_i})$ *for* $i = 1, \ldots, m$.

Threading of Barcode Graph

- $\mathrm{H}_1(\mathcal{K}_i^{i+1}) = \mathrm{H}_1(z^{-1}(a_{i+1}))$; same fails for R
- Remedy: use $\tilde{\mathrm{H}}_0$, reduced H_0
- A thread connects all open ends; $\tilde{\mathrm{H}}_0(R_i^{i+1}) = \tilde{\mathrm{H}}_0(z^{-1}(s_i))$



Equivalence of zigzag modules

Proposition

$\mathsf{H}_1(\mathcal{L}(z,\mathcal{K})) \cong \tilde{\mathsf{H}}_0(\mathcal{L}(z,R))$.

With $\tilde{\mathsf{H}}_0(z^{-1}(v)) = \mathbb{U}_v$ and $\mathsf{H}_1(z^{-1}(v)) = \mathbb{V}_v$,

$$
\begin{array}{lccccccccc}
\mathsf{H}_1(\mathcal{L}(z,|\mathcal{K}|)): & \mathbb{V}_{a_1} & \longleftarrow & \mathbb{V}_{s_1} & \longrightarrow & \mathbb{V}_{a_2} & \cdots \longleftarrow & \mathbb{V}_{s_{m-1}} & \longrightarrow & \mathbb{V}_{a_m} \\
 & \| & & \| & & \| & & \| & & \| \\
\tilde{\mathsf{H}}_0(\mathcal{L}(z,R)): & \mathbb{U}_{a_1} & \longrightarrow & \mathbb{U}_{s_1} & \longleftarrow & \mathbb{U}_{a_2} & \cdots \longrightarrow & \mathbb{U}_{s_{m-1}} & \longleftarrow & \mathbb{U}_{a_m}
\end{array}
$$

Equivalence of zigzag modules

Proposition

$\mathrm{H}_1(\mathcal{L}(z,\mathcal{K})) \cong \tilde{\mathrm{H}}_0(\mathcal{L}(z,R))$.

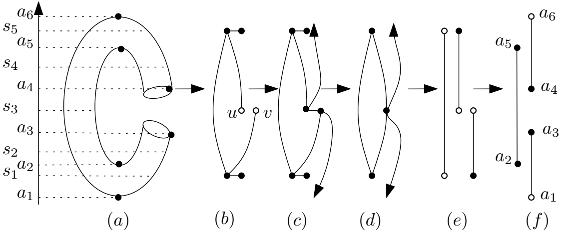
With $\tilde{\mathrm{H}}_0(z^{-1}(v)) = \mathbb{U}_v$ and $\mathrm{H}_1(z^{-1}(v)) = \mathbb{V}_v$,

$$\begin{array}{lccccccccc}
\mathrm{H}_1(\mathcal{L}(z,|\mathcal{K}|)) : & \mathbb{V}_{a_1} & \longleftarrow & \mathbb{V}_{s_1} & \longrightarrow & \mathbb{V}_{a_2} & \cdots \longleftarrow & \mathbb{V}_{s_{m-1}} & \longrightarrow & \mathbb{V}_{a_m} \\
& \| & & \| & & \| & & \| & & \| \\
\tilde{\mathrm{H}}^0(\mathcal{L}(z,R)) : & \mathbb{U}^*_{a_1} & \longleftarrow & \mathbb{U}^*_{s_1} & \longrightarrow & \mathbb{U}^*_{a_2} & \cdots \longleftarrow & \mathbb{U}^*_{s_{m-1}} & \longrightarrow & \mathbb{U}^*_{a_m}
\end{array}$$

- Output bars extracted from modified barcode graph R
- Use algorithm of [AEHW06] with the mergable tree data structure of [GTW06] to extract bars; $O(n \log n)$ time

$O(n \log n)$ Algorithm

- Compute barcode graph R; $O(n \log n)$
- Thread R; $O(n)$
- Extract bars; $O(n \log n)$
- Flip the bar ends; $O(n)$
- Compute open-open bars from Reeb graph; $O(n \log n)$

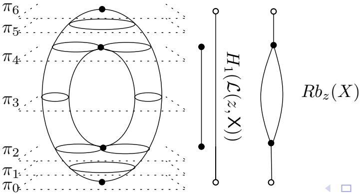


Open-open bars

Proposition

$\mathsf{H}_0(\mathcal{L}(z, \mathcal{K})) \cong \mathsf{H}_0(\mathcal{L}(z, \mathrm{R}b_z(\mathcal{K})))$.

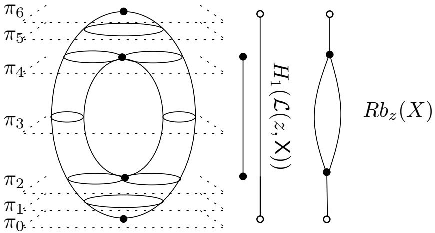
$$
\begin{tikzcd}
\mathsf{H}_0(\mathcal{L}(z,\mathcal{K})) : & \mathsf{H}_0(\mathcal{K}_0^0) \arrow[r] \arrow[d, equal] & \mathsf{H}_0(\mathcal{K}_0^1) \arrow[d, equal] & \mathsf{H}_0(\mathcal{K}_1^1) \arrow[l] \arrow[d, equal] \cdots & \mathsf{H}_0(\mathcal{K}_m^m) \arrow[l] \arrow[d, equal] \\
\mathsf{H}_0(\mathcal{L}(z,\mathrm{R}b_z(\mathcal{K}))) : & \mathsf{H}_0(\mathbb{R}_0^0) \arrow[r] & \mathsf{H}_0(\mathbb{R}_0^1) & \mathsf{H}_0(\mathbb{R}_1^1) \arrow[l] \cdots & \mathsf{H}_0(\mathbb{R}_m^m) \arrow[l]
\end{tikzcd}
$$



Open-open bars

Proposition

$\mathsf{H}_0(\mathcal{L}(z, \mathcal{K})) \cong \mathsf{H}_0(\mathcal{L}(z, \mathrm{R}b_z(\mathcal{K})))$.



- Compute the open-open bars of $\mathsf{H}_0(\mathcal{L}(z, \mathcal{K}))$ from the Reeb graph; compute $\mathrm{R}b_z(\mathcal{K})$ in $O(n \log n)$ time and then extract bars from it in another $O(n \log n)$ time

Generators

- $\mathsf{H}_1(|\mathcal{K}|) \cong \check{\mathsf{B}}_0(z, |\mathcal{K}|) \oplus \bar{\mathsf{B}}_1(z, |\mathcal{K}|)$
- generators for open-open bars are given by cycles in Reeb graph; $O(n \log n + k)$ time
- generators for closed-closed bars can be computed from tracking the level sets; $O(n \log n + k)$ time

Thank You