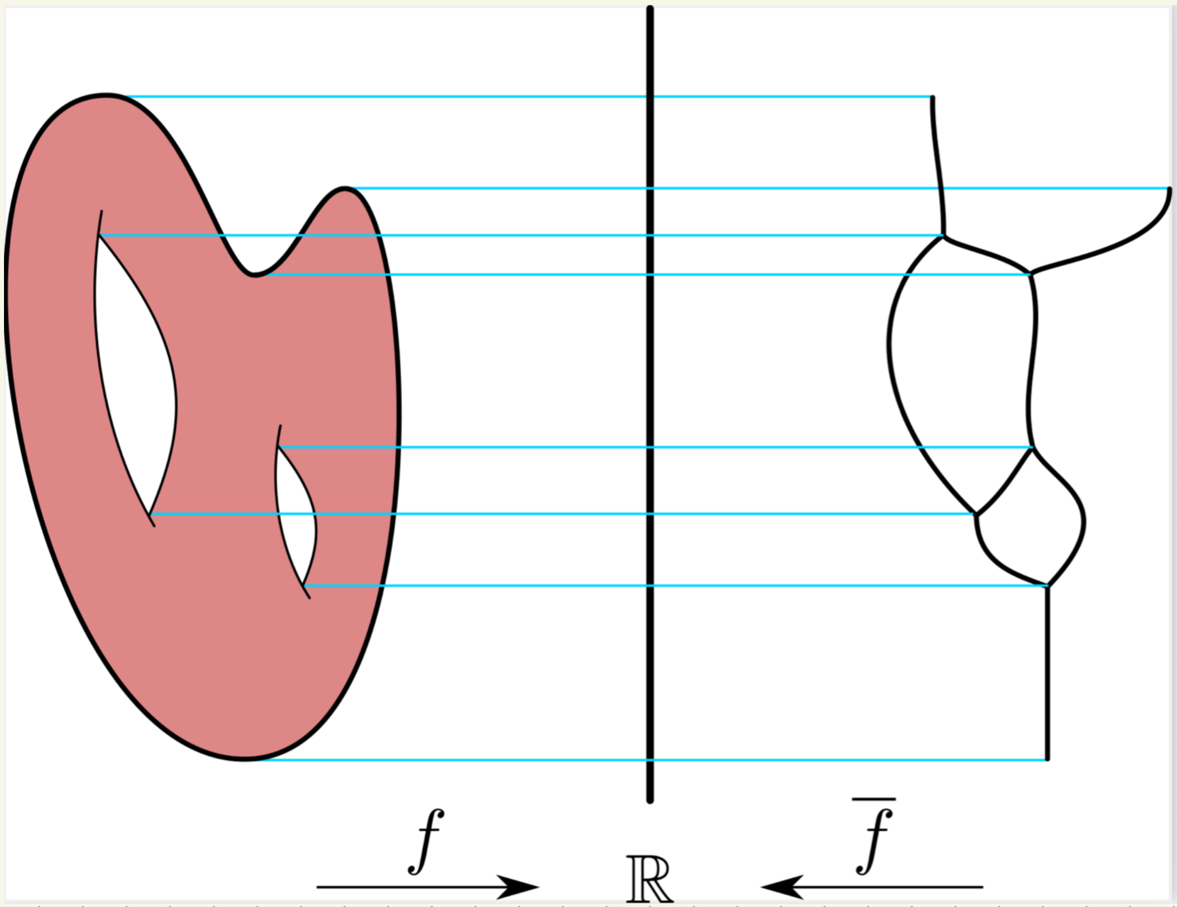
TDA - Fall 2025

More on Reeb
Graphs

Reeb Graph Definition

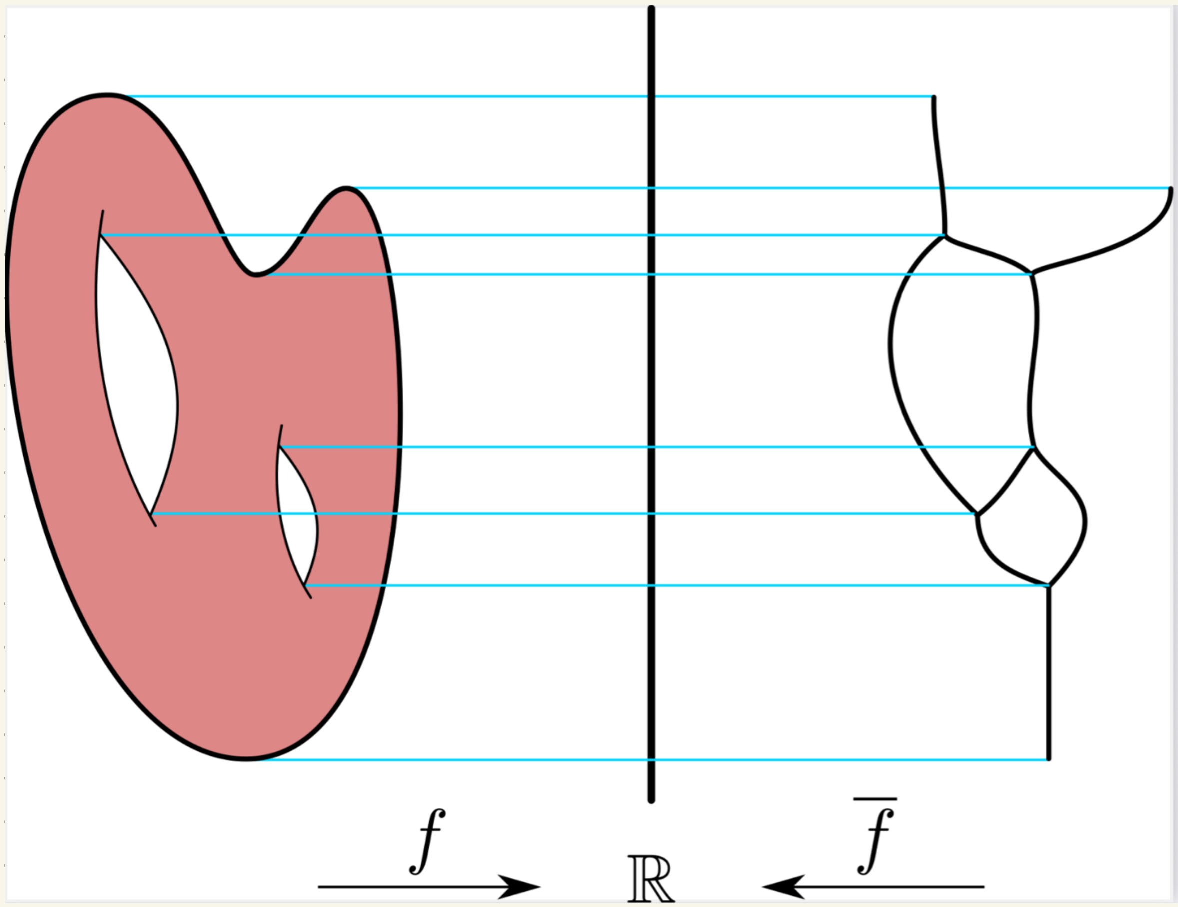
Given a function $f: X \to \mathbb{R}$, define an equivalence relation $\sim$ by $x \sim y$ iff:
- $f(x) = f(y) = a$
- x & y in same connected component of level set $f^{-1}(a)$.



Let $[x]$ be equiv. class of x:
$\{y \in X \mid y \sim x\}$

The Reeb graph R_f is the quotient space $X/\sim$.
Let $\Phi: X \to R_f$, $x \mapsto [x]$ be quotient map

Induced map $\tilde{f}: R_f \to \mathbb{R}$



Define

$\tilde{f}: R_f \to \mathbb{R}$

$[x] \longmapsto f(x)$

End result: a Reeb graph is a graph with an $\mathbb{R}$-valued function

Discrete versus continuous viewpoints

Graph $G=(V,E)$

V: vertex set

E = edges

$uv \in E \implies f(u) \neq f(v)$

Essentially, a 1-dim abstract simplicial complex

"Topological" graph

- Vertices
- Edges $\cong$ interval

Essentially, a 1-dim stratified space

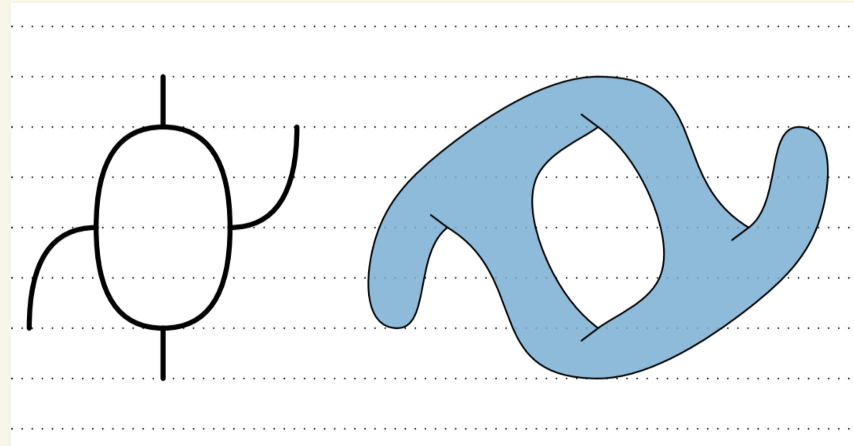
↳ 1 manifold except at singularities

function f:

Vertices in R_f

Given a node $x \in V$ (where $V = V(R_f)$),

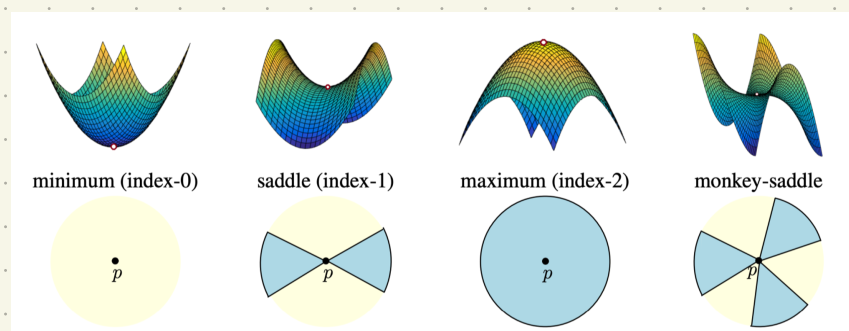
- up-degree of x:



- down degree of x:

A <u>node</u> is

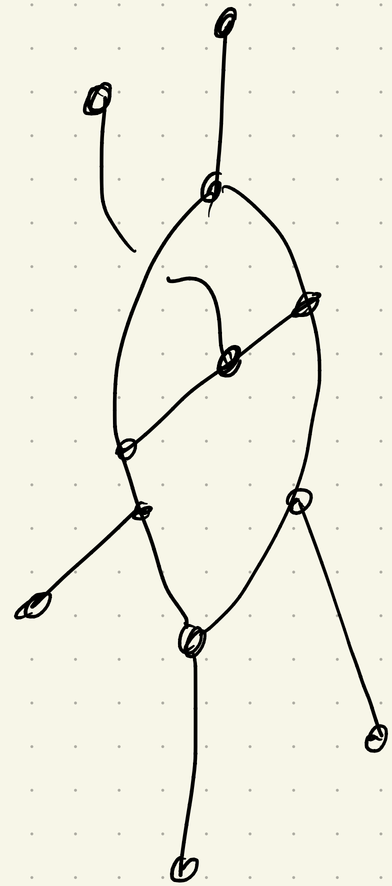
- regular if up & down degree both $= 1$
- critical otherwise



[Aside: Why?]

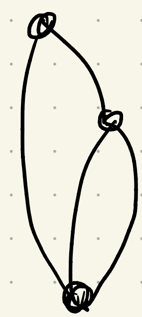
Types of critical points

A critical point is a



- minimum if it has down-degree = 0
- maximum if it has up-degree = 0
- a downfork if it has down-degree > 1
- an upfork if it has up-degree > 1

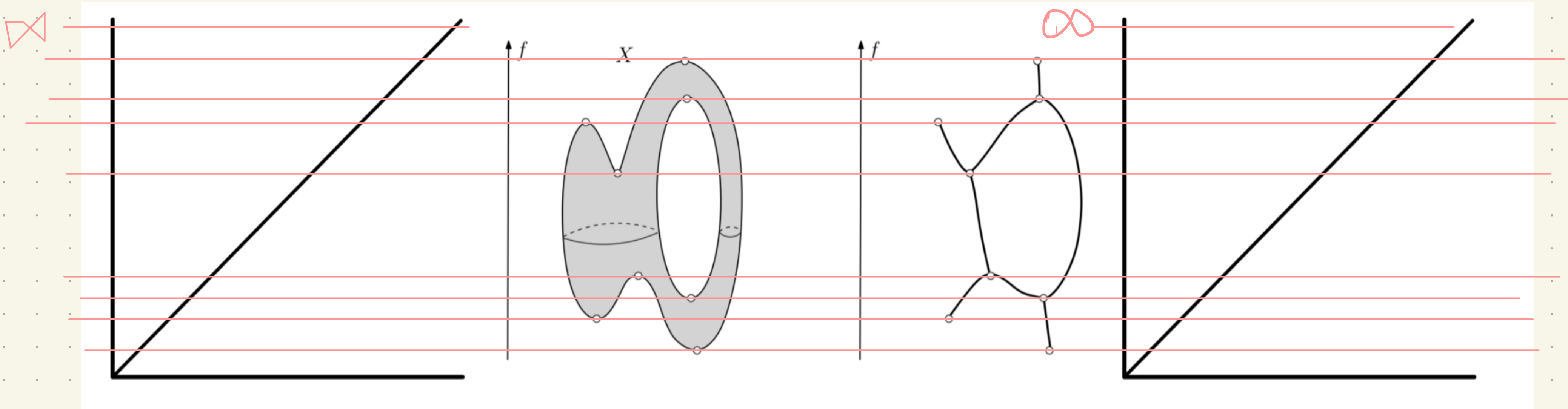
Note: a degenerate critical point may have more than one type of criticality



Persistence & Betti Numbers

Theorem: For a tame function $f: X \to \mathbb{R}$,
$\beta_0(X) = \beta_0(R_f)$ and $\beta_1(X) \geq \beta_1(R_f)$.

Why? Look at "standard" persistence:



0-dim:

1-dim:

2-dim:

Special settings: 2-manifolds

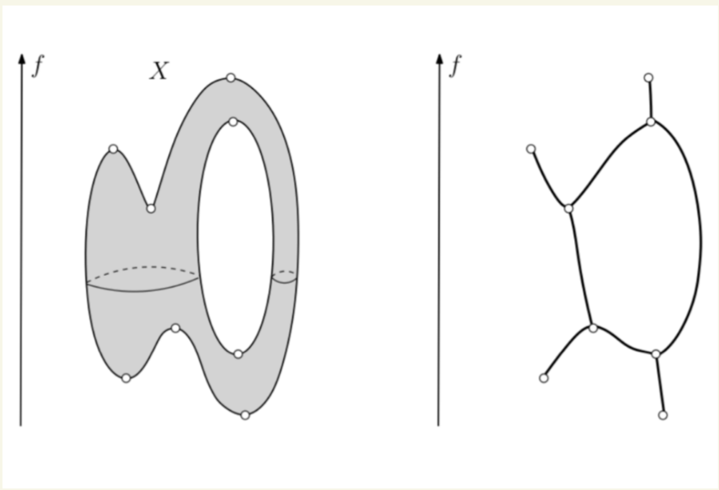
Theorem: Let $f: X \to \mathbb{R}$ be a Morse function on a connected compact 2manifold. Then if X is orientable,

$$\beta_1(R_f) = \beta_1(X)/2$$

If X is non orientable,

$$\beta_1(R_f) \leq \beta_1(X)/2$$

Recall:



Leads to an interesting classification of cycles:
vertical versus horizontal (with respect to f)

- gives improved algorithms Dey-Wang 2012
- & applications to surface reconstruction

Figure 7: *From left to right: handle and tunnel loops computed on Casting, tunnel features are identified, small tunnel features are filled.*

Dey et al 2008

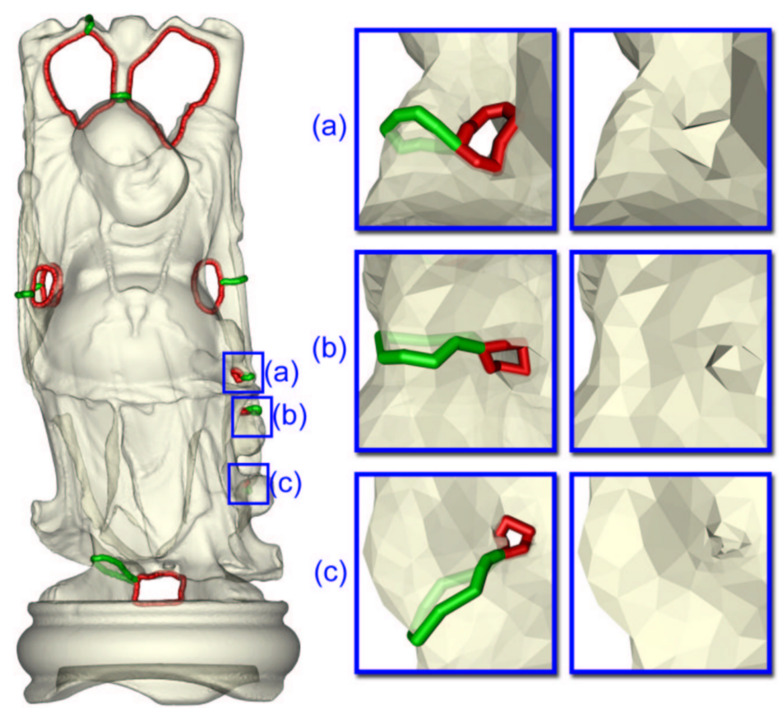
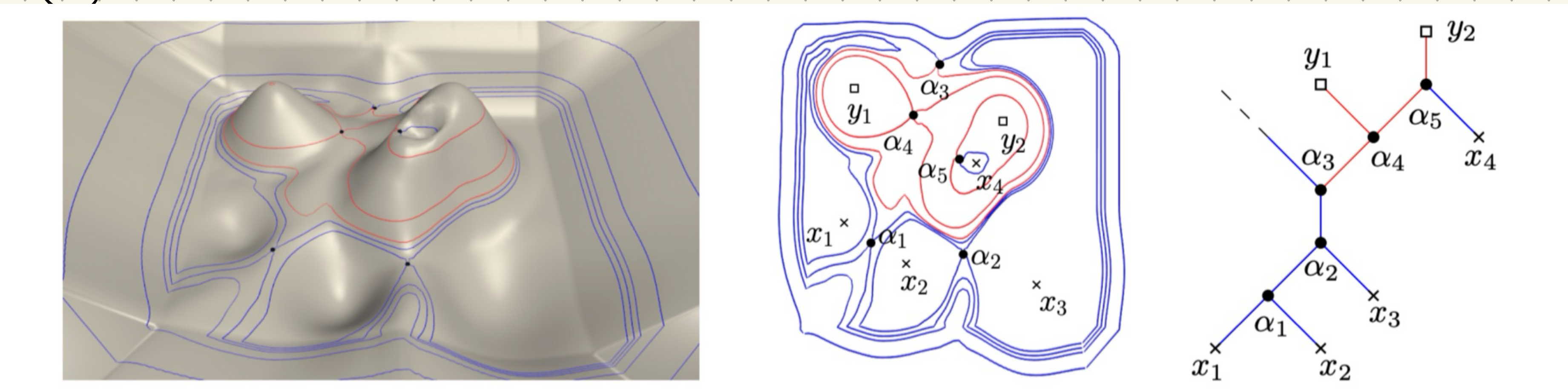


Figure 8: *Three small tunnels in Buddha are filled.*

Simplifications

If X has no H_1, then resulting graph is called a contour tree.

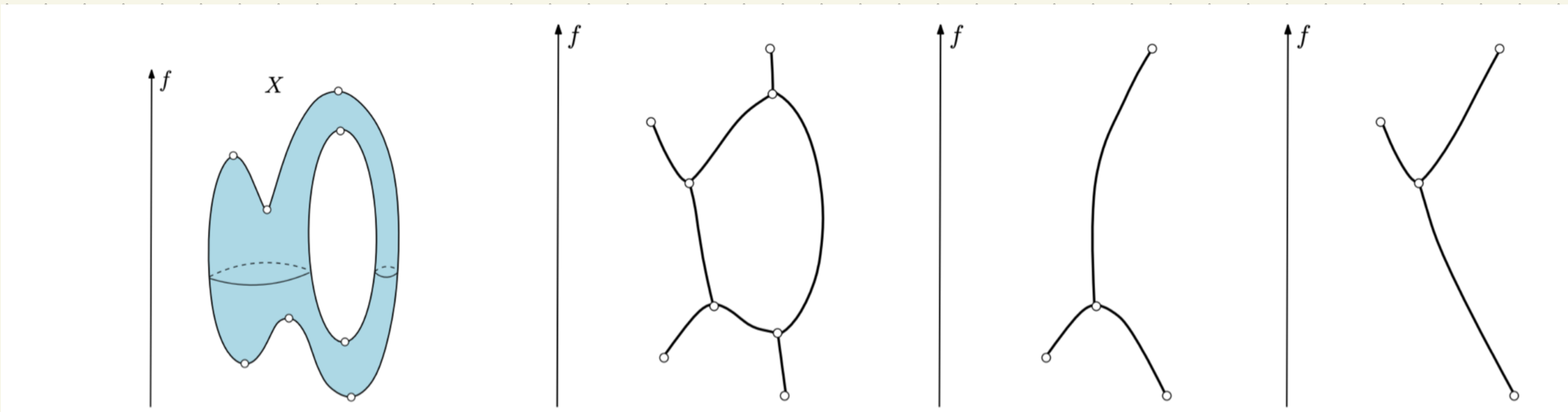


(a) (b) (c)

$\boxed{\text{Agarwal et al 2014}}$

Why we care:

In fact, simpler versions of Reeb graphs are very popular in visualization & more applied work:



(a) Input scalar field (b) Reeb graph (c) Merge tree Split tree

- Merge trees:
- Split trees:

Some examples: Yan et al 2021 ← survey

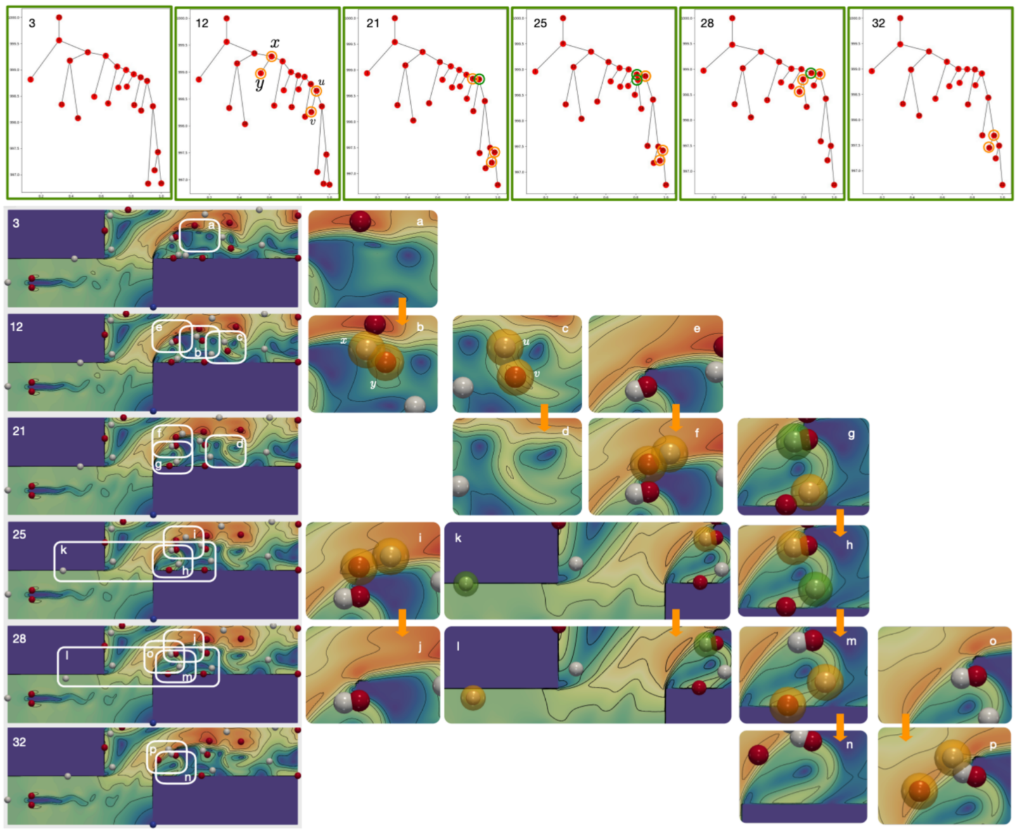


Fig. 11: Sketching the *Corner Flows* dataset with 15 basis trees with IFS: (Top) first 6 basis trees where orange circles highlight topological changes w.r.t. near basis trees, (Bottom Left) scalar fields that give rise to these basis trees, areas with critical points appearances/disappearances are shown with zoomed views in (Bottom Right).

Li et al 2025

Fasy et al 2024

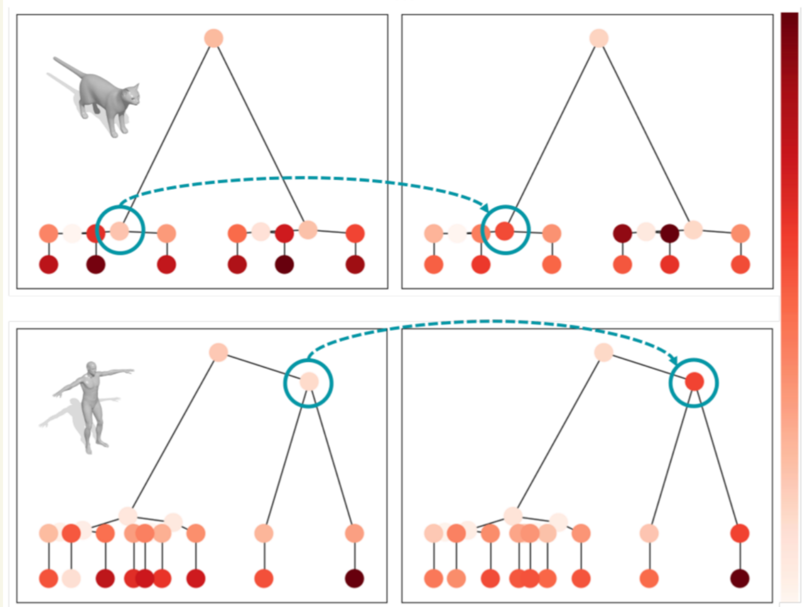


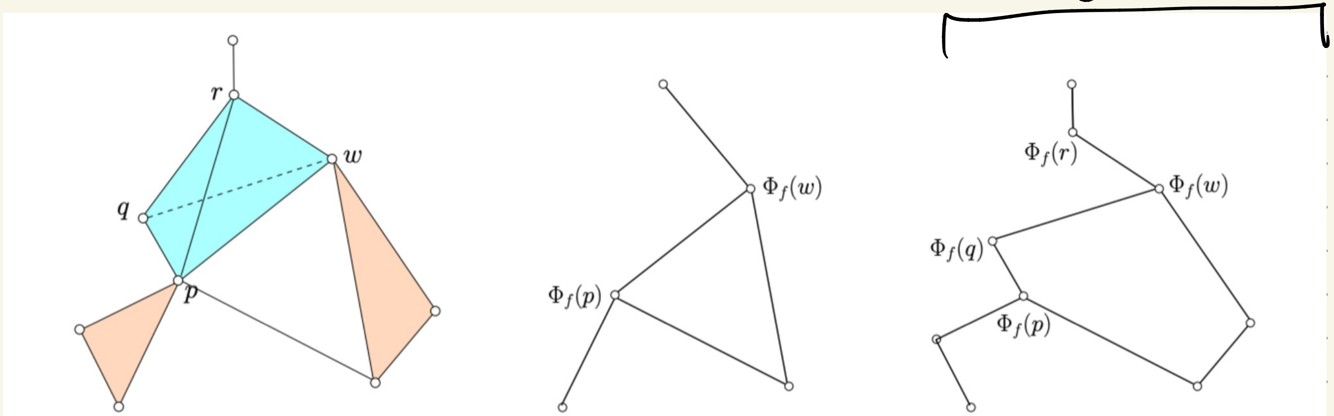
Fig. 13: Topological attention on two merge trees from the TOSCA dataset. Each row displays the merge tree visualized by the relative importance, with example inserts within the merge trees. Left: node importance without topological attention, using the GIN model. The node colors indicate their relative importance when comparing the two examples (the darker with red, the higher the weight). Right: node importance with topological attention using the MTNN model. The GIN model already emphasizes structural differences between the two merge trees. But, the introduction of topological attention further emphasizes these differences, re-weighing the nodes in the high persistence feature highlighted in cyan.

Computation

Given a simplicial complex K & a PL function $f: |K| \to \mathbb{R}$, R_f depends only on 2-skeleton.

Why?

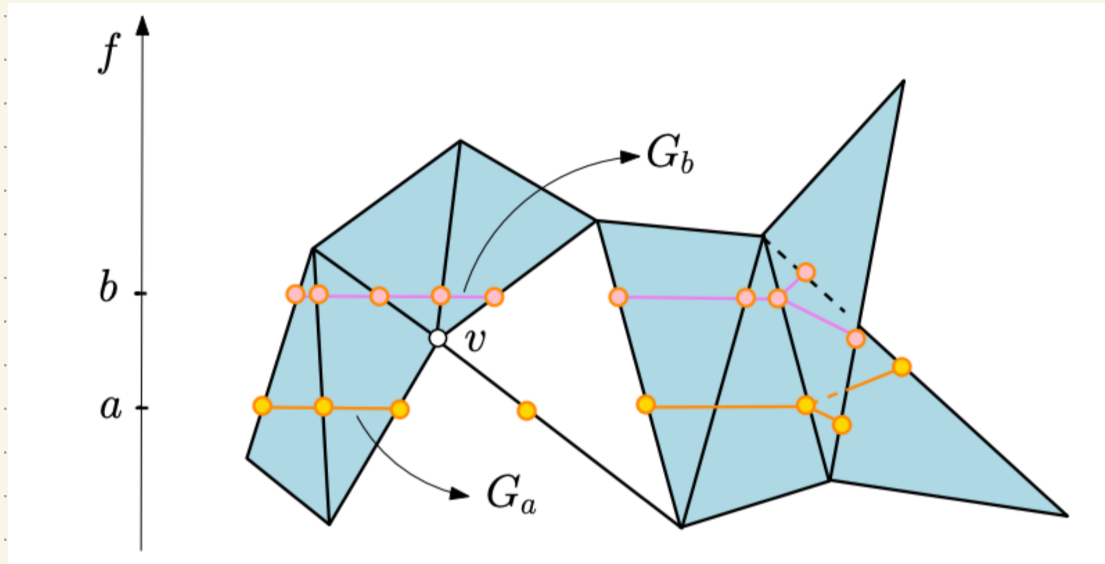
Augmented Reeb graph:



(a) (b) (c)

Sweep-line algorithm:

- Sort vertices by f
- Track connected components
- Graph changes only at vertices; + then only at simplices adjacent to vertex



Data structure:
Track "forest", allowing edge insertion + removal
↳ dynamic trees, so $O(\log m)$ each update

Question: Given 2 Reeb graphs, how to compare them?

Goal: At least an extended pseudo metric:

1) $d(X, X) = 0$
2) $d(X, Y) = d(Y, X)$
3) $d(X, Z) \leq d(X, Y) + d(Y, Z)$

Weaker than metric:

- extended: could be ∞
- pseudo: $d(x, y) = 0 \not\Rightarrow x = y$

More considerations:

1. **Stability**: "Perturbed" spaces should have small distance: $d(R_f, R_G) \leq \|f-g\|$
2. **Discriminativity**: How well can it capture differences compared to alternatives, or some baseline?
3. **Isomorphism invariance**: Can we have $R_g = R_f \iff d(R_g, R_f) = 0$?
4. **Path component tolerance**: For some measures, $d(R_g, R_f) = \infty$ $\iff$ different # of path components.
5. **Universality**: (more later...)

Options Several have been studied.

Edit distances

Based on more combinatorial algorithms, modified to use weights

Interleaving distances

Based on more categorical approach.

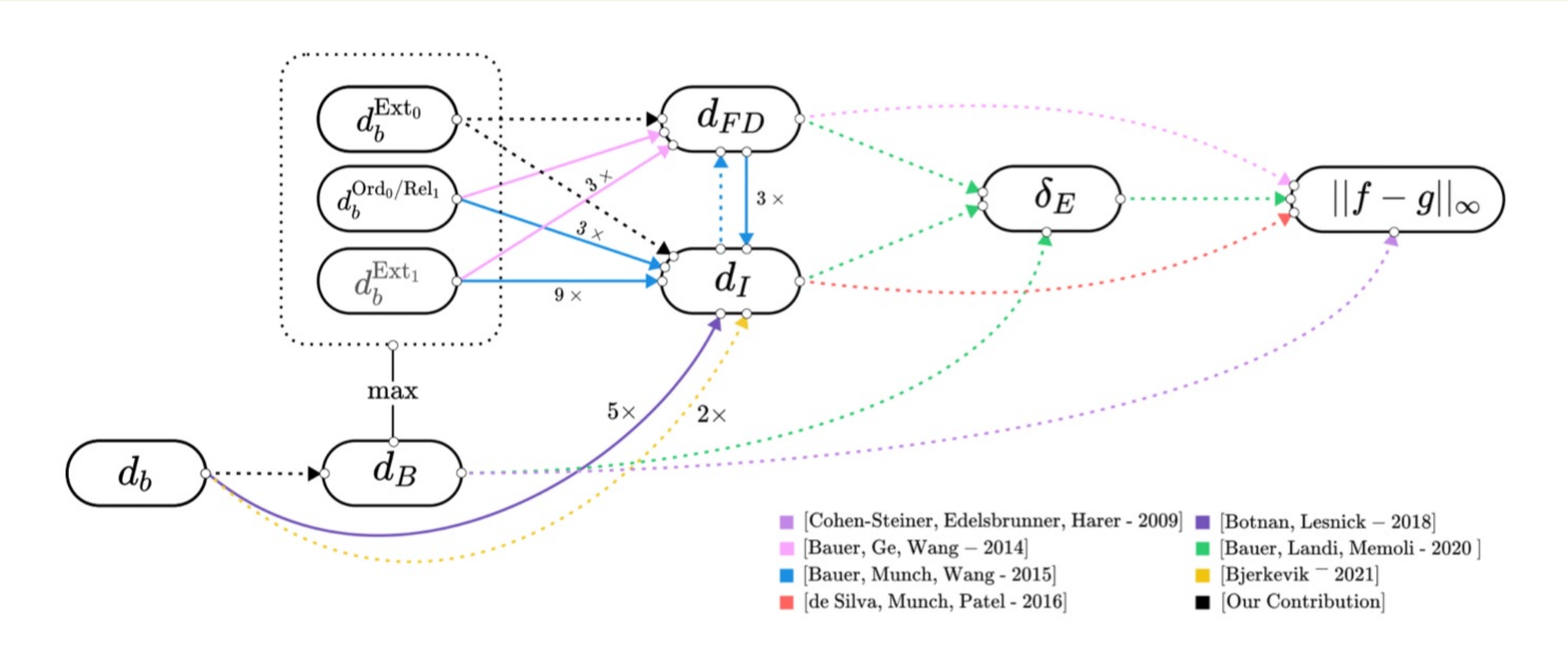
Functional distortion

Based on ideas from Gromov-Hausdorff distances.

Bottleneck distances

Based on persistent homology of the Reeb graph.

Surprisingly, these are often related!

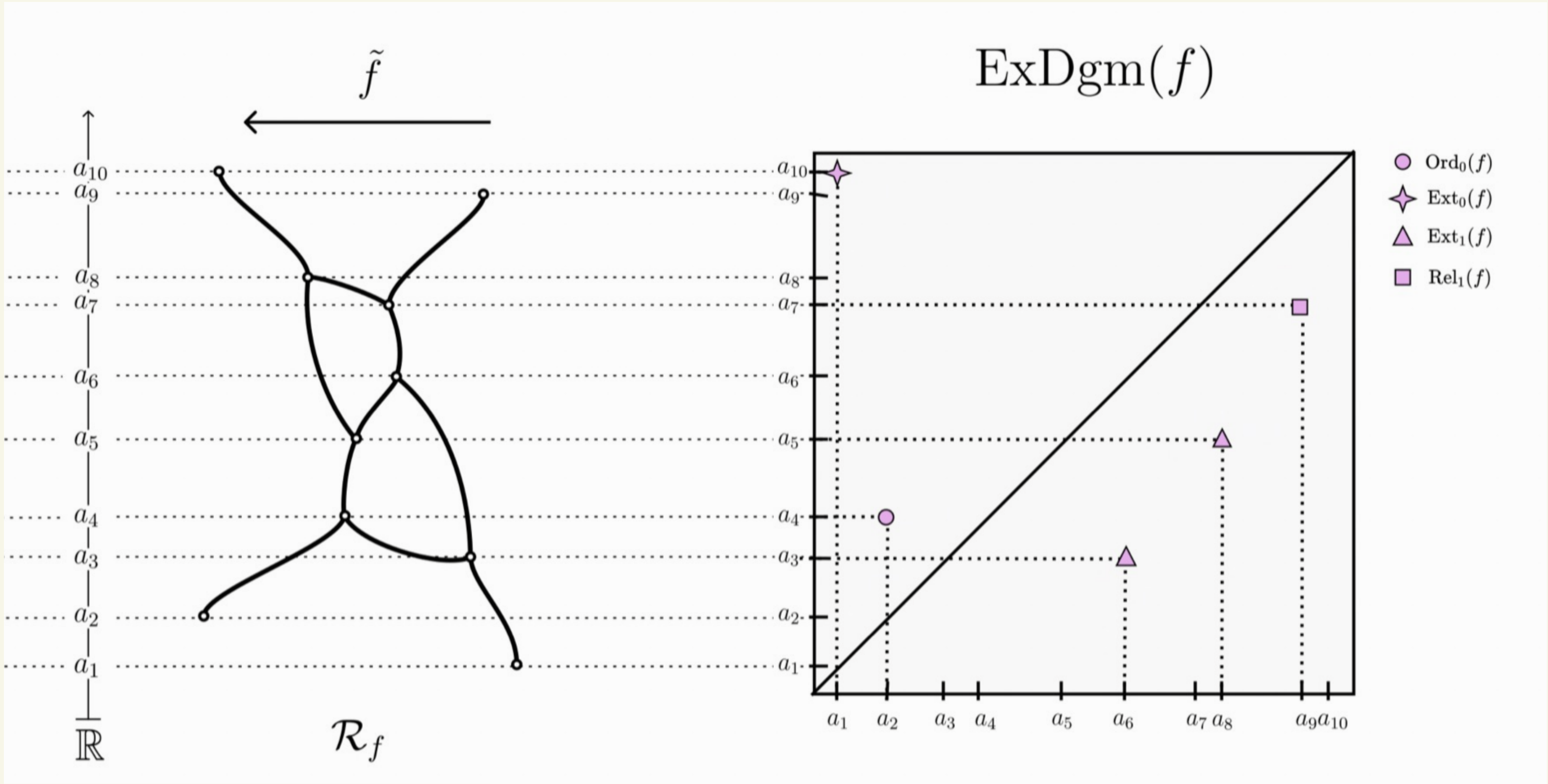


Lets look more carefully...

Bottleneck distance:

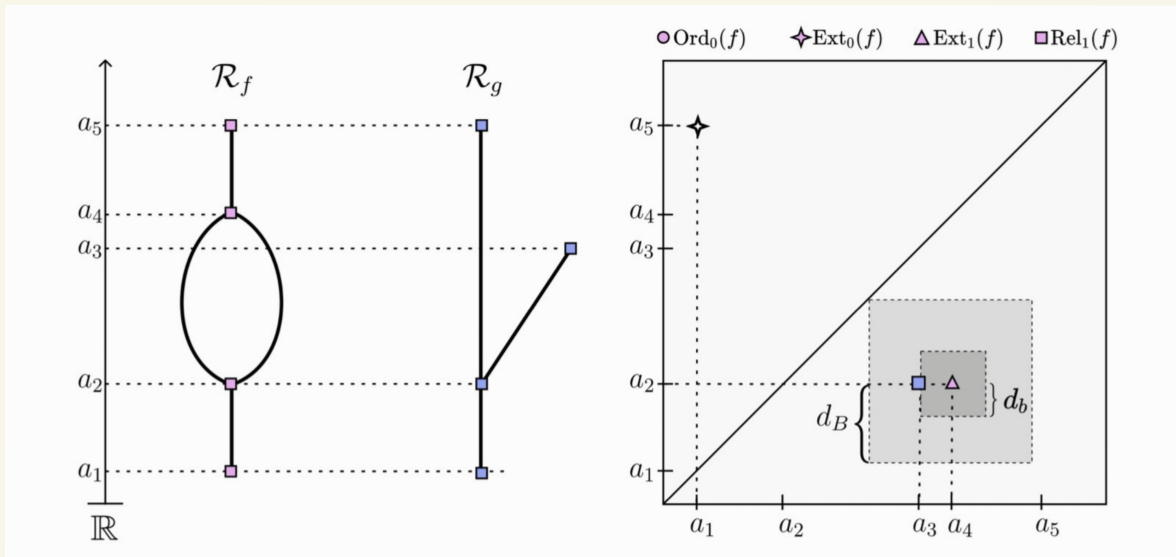
Based on persistence or extended persistence of the Reeb graph (NOT input space):

Recall:



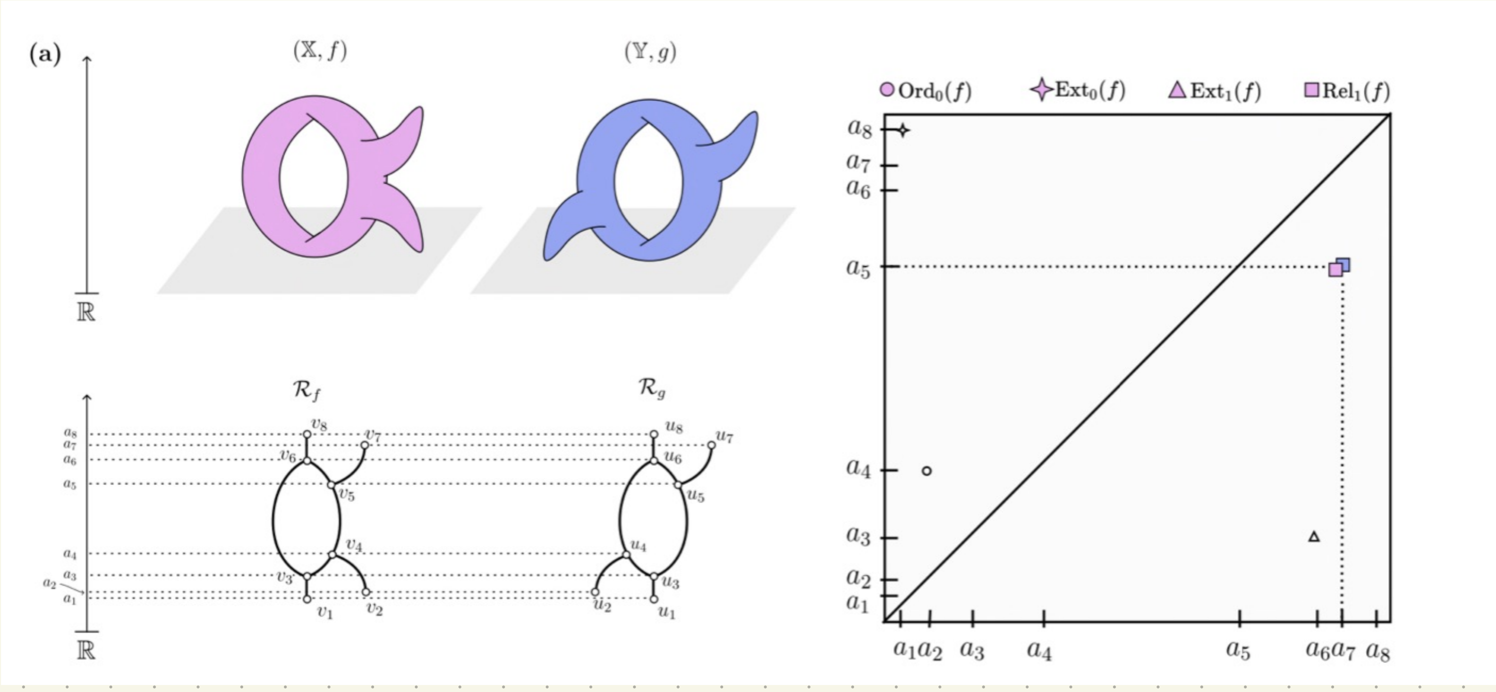
Back to bottleneck distance:

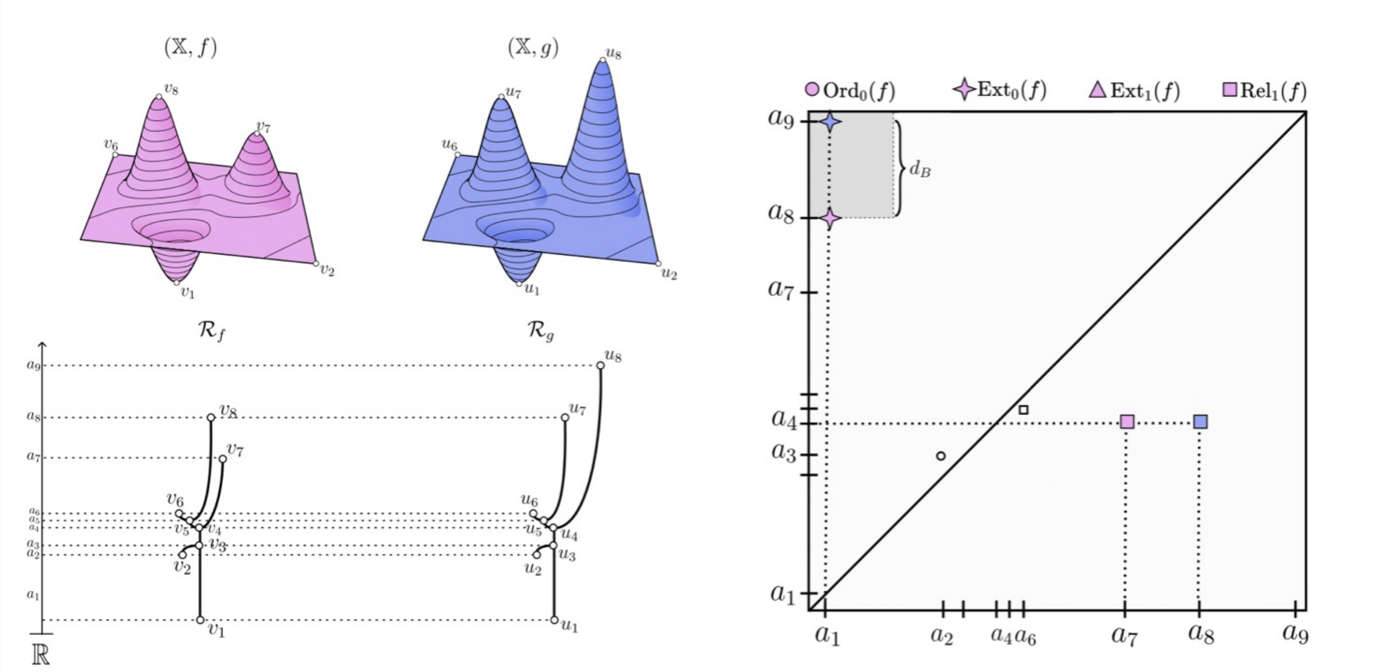
To compare two Reeb graphs, can simplify to their diagrams & compare those:



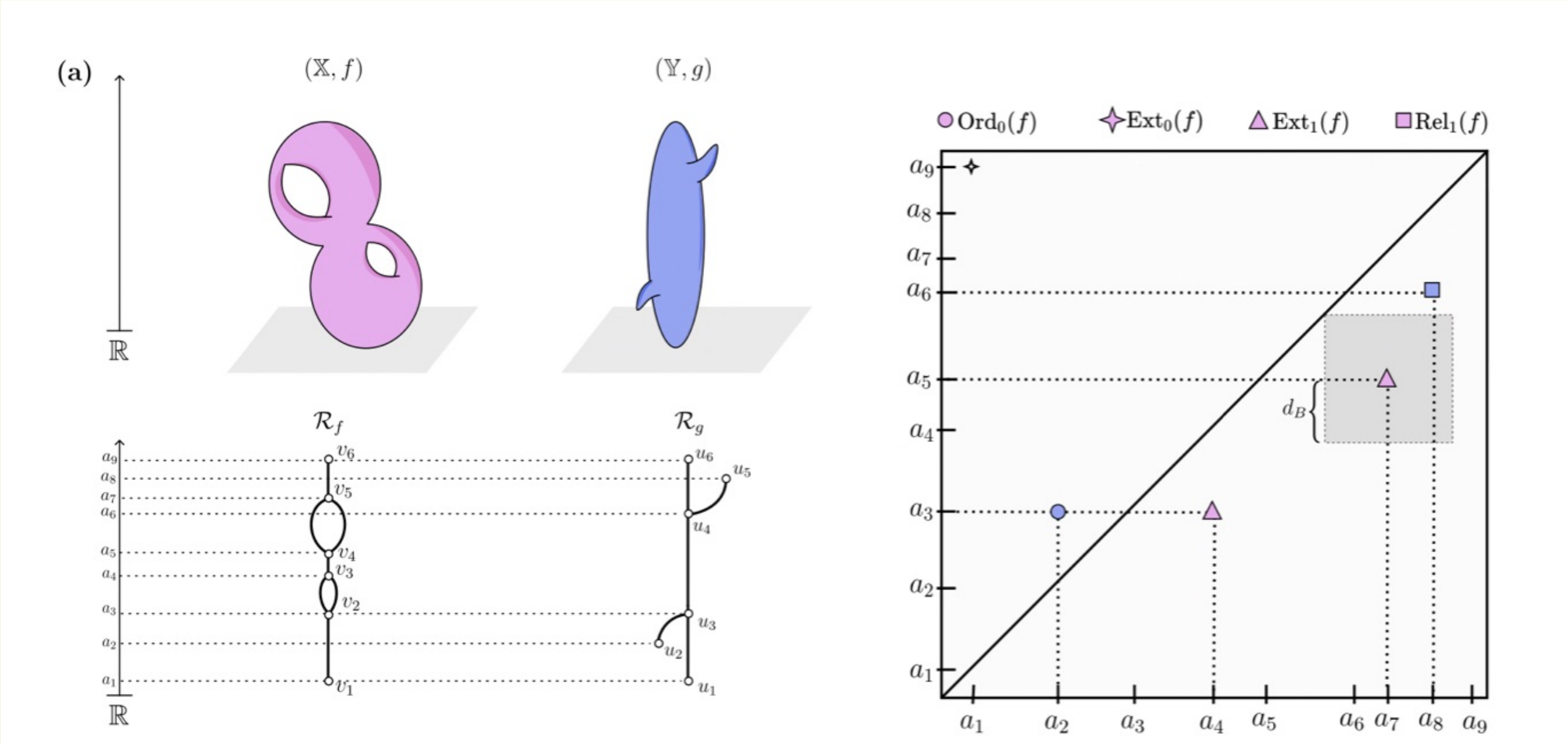
Note: - Allowed to match any point to diagonal.
- Types can either matter (graded d_B) or not matter (ungraded d_b).

Examples:


(a)
$(\mathbb{X}, f)$
$(\mathbb{Y}, g)$
$\mathbb{R}$
$\mathcal{R}_f$
$\mathcal{R}_g$
$\mathrm{Ord}_0(f)$
$\mathrm{Ext}_0(f)$
$\mathrm{Ext}_1(f)$
$\mathrm{Rel}_1(f)$
a_1 a_2 a_3 a_4 a_5 a_6 a_7 a_8


$(\mathbb{X}, f)$
$(\mathbb{X}, g)$
$\mathcal{R}_f$
$\mathcal{R}_g$
$\mathbb{R}$
$\mathrm{Ord}_0(f)$
$\mathrm{Ext}_0(f)$
$\mathrm{Ext}_1(f)$
$\mathrm{Rel}_1(f)$
d_B
a_1 a_2 a_3 a_4 a_6 a_7 a_8 a_9

Examples :



(a)
$(\mathbb{X}, f)$
$(\mathbb{Y}, g)$
$\mathbb{R}$
$\mathcal{R}_f$
$\mathcal{R}_g$
a_1 a_2 a_3 a_4 a_5 a_6 a_7 a_8 a_9
v_1 v_2 v_3 v_4 v_5 v_6
u_1 u_2 u_3 u_4 u_5 u_6
$\mathrm{Ord}_0(f)$
$\mathrm{Ext}_0(f)$
$\mathrm{Ext}_1(f)$
$\mathrm{Rel}_1(f)$
d_B

Bottleneck pros + cons:

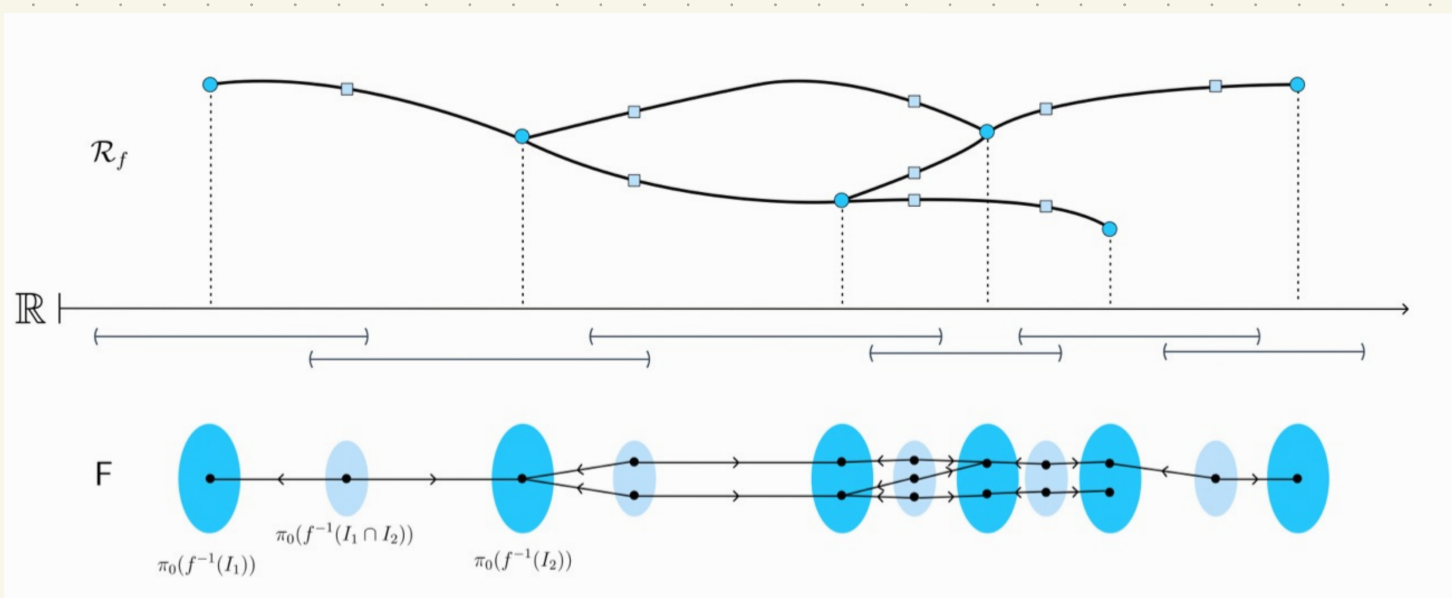
- Stable [Cohen-Steiner et al 2007]
- Can compute extended persistence quickly - then it's geometric matchings. (subquadratic - see [Efrat et al 2001] and [Kerber et al 2016])
- It is not isomorphism invariant
- All other metrics are more discriminative [Bauer et al 2014]
- It is not path component sensitive

Interleaving distance:

Takes a categorical point of view:
Reeb graph is a set valued cosheaf,

$$F : \mathrm{Int} \longrightarrow \mathrm{Set}$$

$$\begin{tikzcd}
I = (a,b) \arrow[r] \arrow[d, hook] & \pi_0\left(f^{-1}((a,b))\right) \arrow[d, "{F[I \subseteq J]}"] \\
J = (c,d) \arrow[r] & \pi_0\left(f^{-1}((c,d))\right)
\end{tikzcd}$$

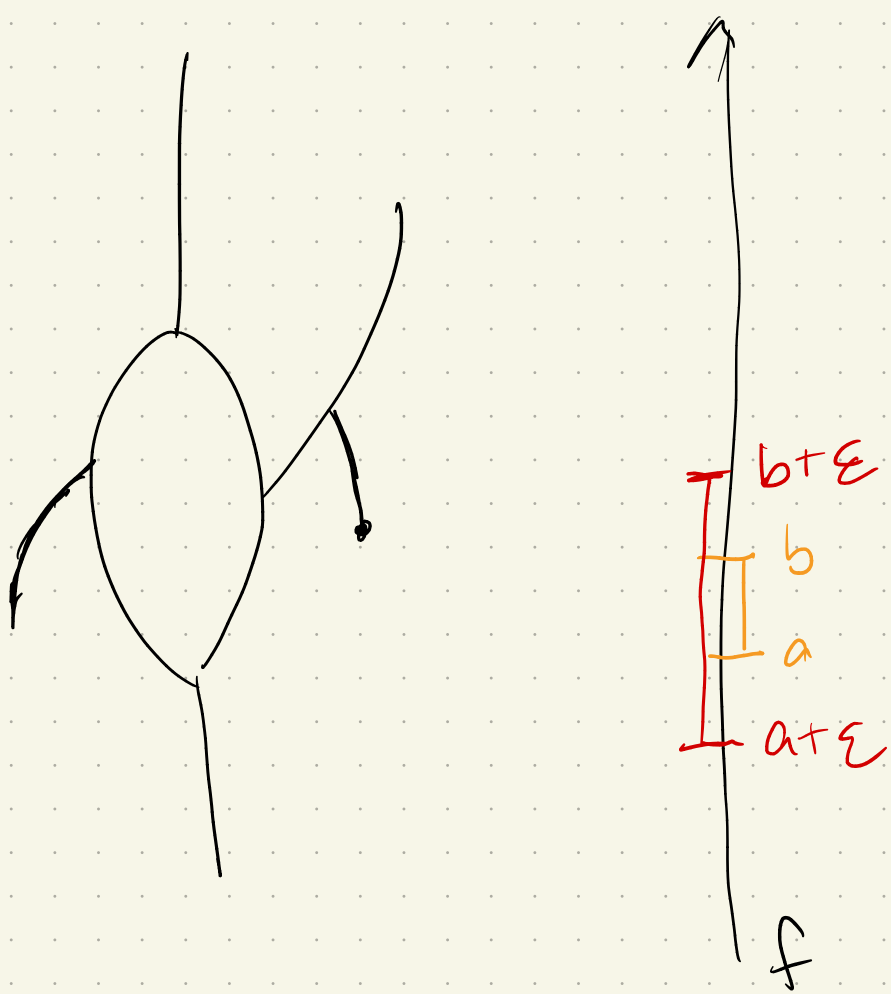


Smoothing a Reeb co-sheaf:

Given $F : \mathrm{Int} \longrightarrow \mathrm{Set}$

return $S_\varepsilon(F) : \mathrm{Int} \longrightarrow \mathrm{Set}$

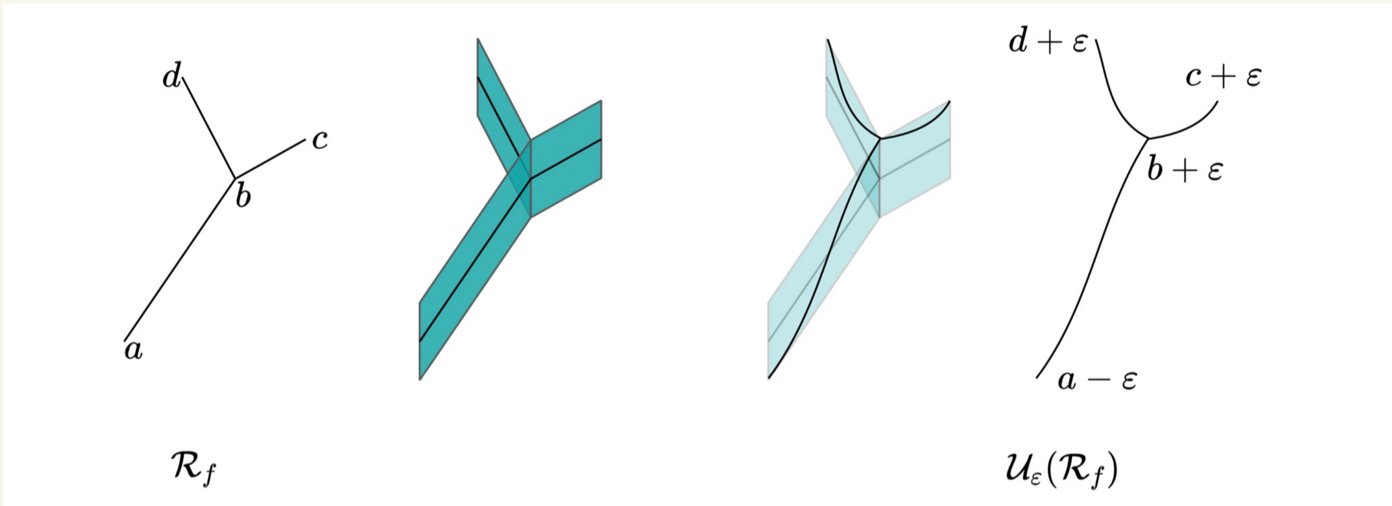
$$(a,b) \longrightarrow f^{-1}(\pi_0(a-\varepsilon, b+\varepsilon))$$



Result:

a different Reeb co-sheaf, related to original

Result: In [de Silva et al 2016], they connect this to the thickening functor:



Result: Nice connection to a geometric notion of similarity.

Interleaving distance: Why do we care??

Given 2 Reeb cosheaves

$$F, G : \mathrm{Int} \to \mathrm{Set}$$

an ε-interleaving is φ and ψ such that the diagram commutes:

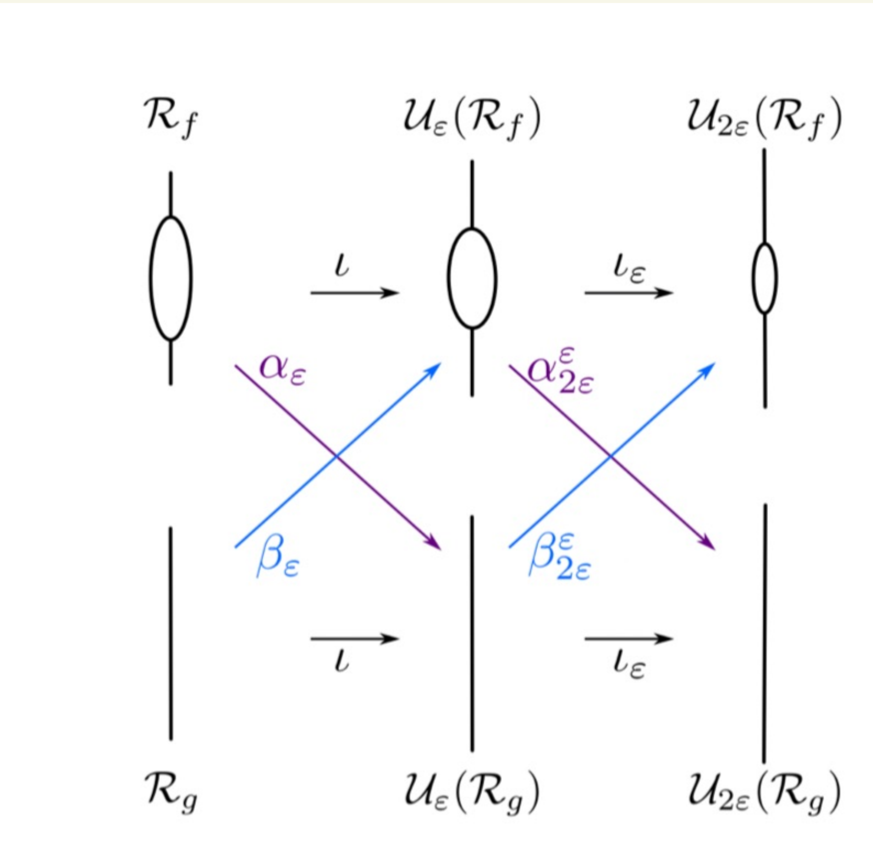
$$\begin{tikzcd}
F \arrow[r] \arrow[dr, "\varphi"] & S_\varepsilon F \arrow[r] \arrow[dr, "\varphi"] & S_{2\varepsilon} F \\
G \arrow[r] \arrow[ur, "\psi"] & S_\varepsilon G \arrow[r] \arrow[ur, "\psi"] & S_{2\varepsilon} G
\end{tikzcd}$$

Reeb Interleaving distance is

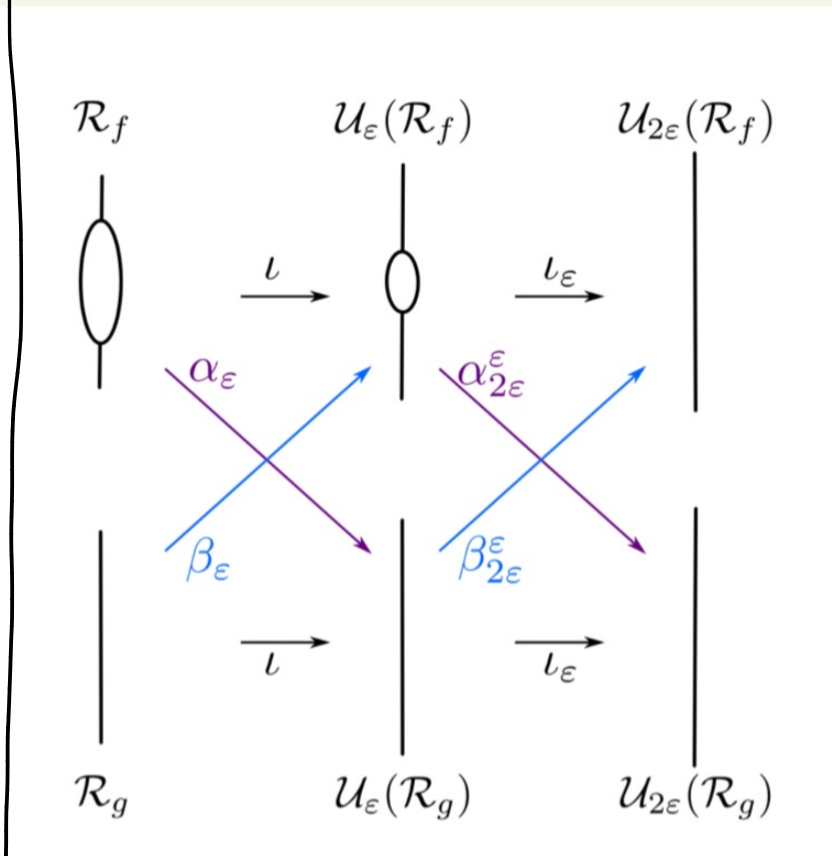
$$d_I(F,G) = \inf \{ \varepsilon \geq 0 \mid \exists\ \varepsilon\text{-interleaving} \}$$

Example:

Bad ε:



Good ε:



Interleaving distance pros + cons:

It is: - discriminative [Bauer et al 2015]

- stable
- path component sensitive
- isomorphism invariant

[de Silva et al, 2016]

Unfortunately, it's Graph-Isomorphism hard to compute.

→ Essentially, can use 0-interleaved to check for graph isomorphism.

Edit distances:

Edit distances are well studied for strings and abstract graphs.

↳ see $\boxed{\text{[Bille 2005]}}$ for the many variants on graphs.

In a graph, usually have:

- vertex insertion / deletion
- edge insertion / deletion

and some cost associated with each operation.

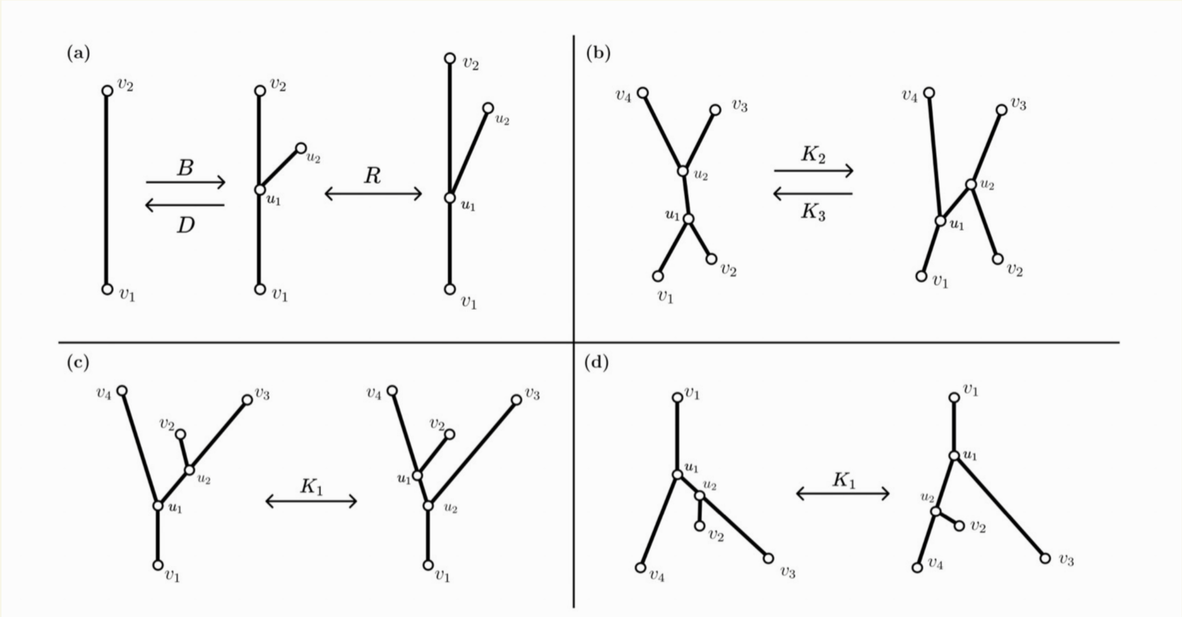
$$\text{Total Edit distance} = \min\left\{\sum(\text{edit costs})\right\}$$

Edits:

In Reeb graphs, only certain edits will correspond to topological deformations on original space.

[Di Fabio & Landi, 2012 and 2016]

- insert
- delete
- relabel
- K-type

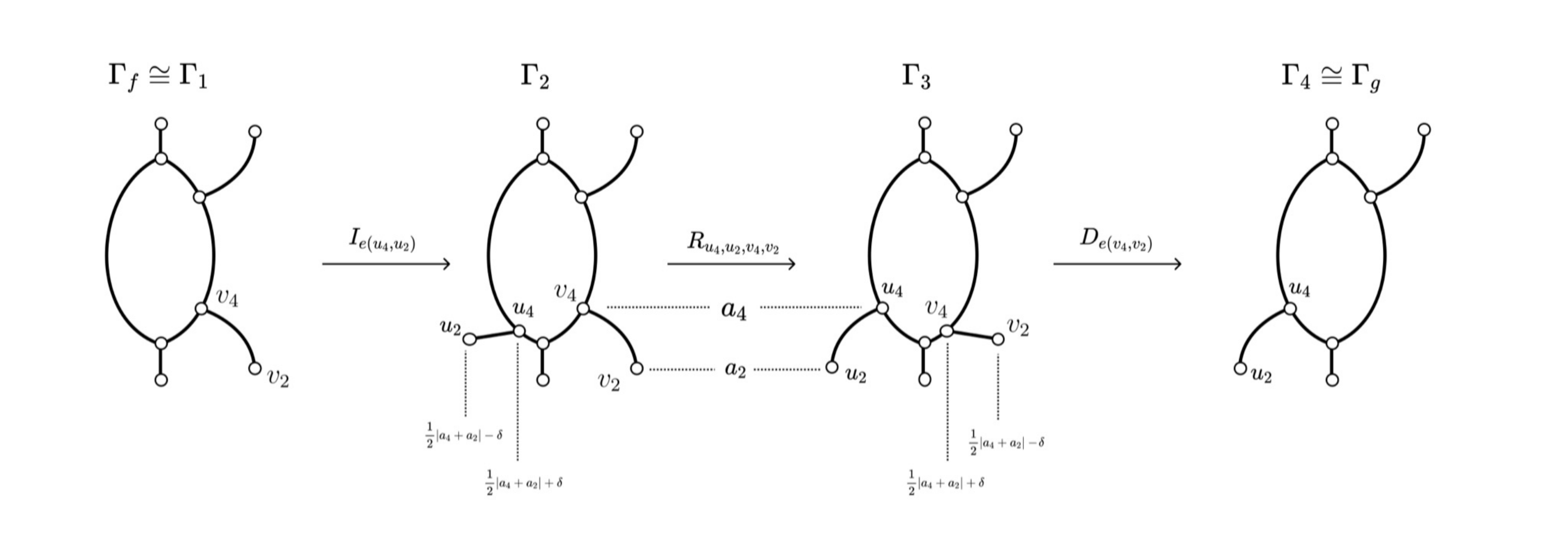


(Reason: Morse theory!)

Downside

Could only compare homeomorphic spaces, originally.

Also a more subtle flaw: can pay less by infinitesmally shifting vertices, since relabels cost max of operations.



Categorical edit distance

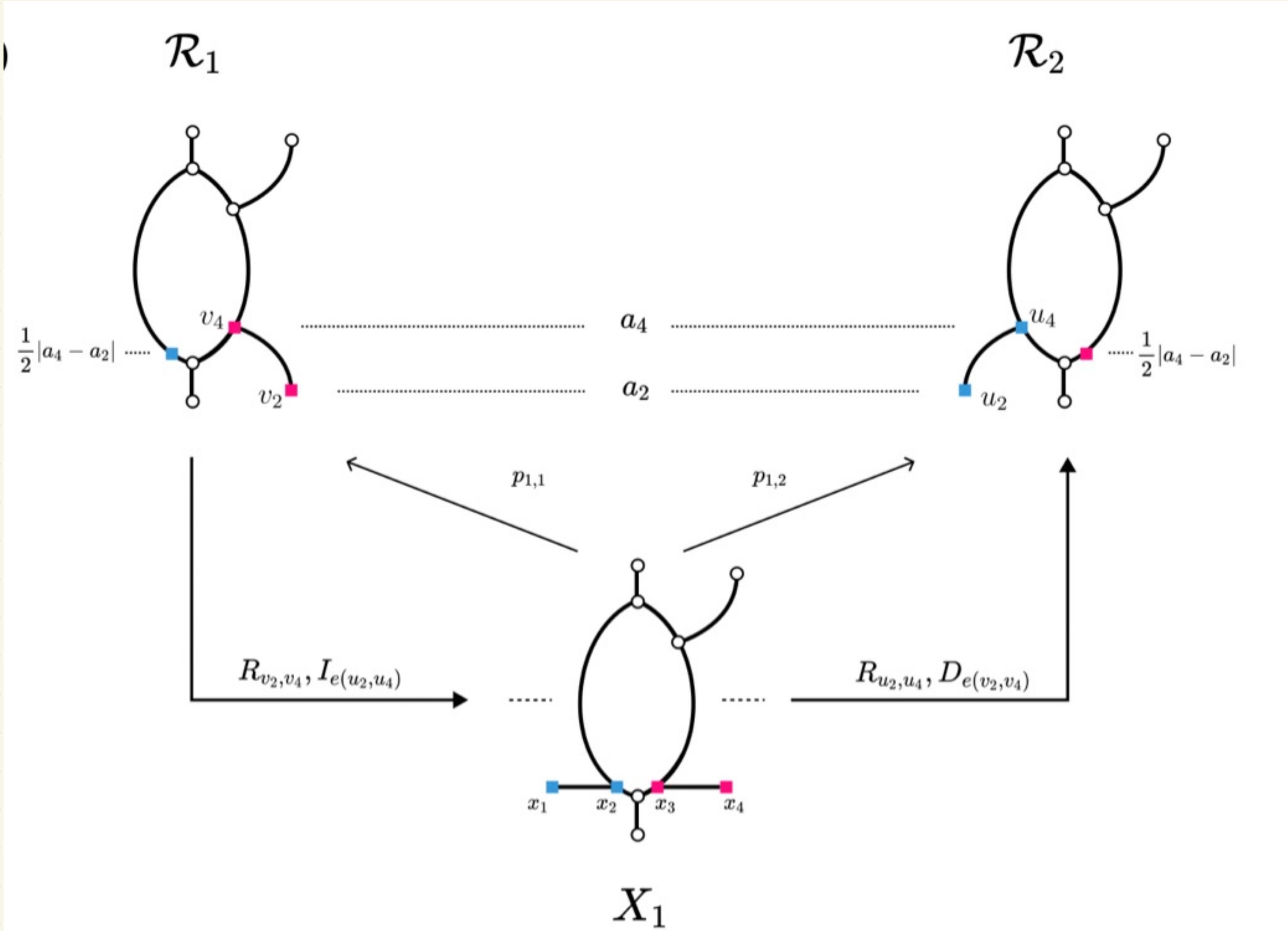
[Bauer et al 2021]

Instead of "direct" edits to Reeb graphs, goal it to find a sequence of topological spaces making this diagram commute.

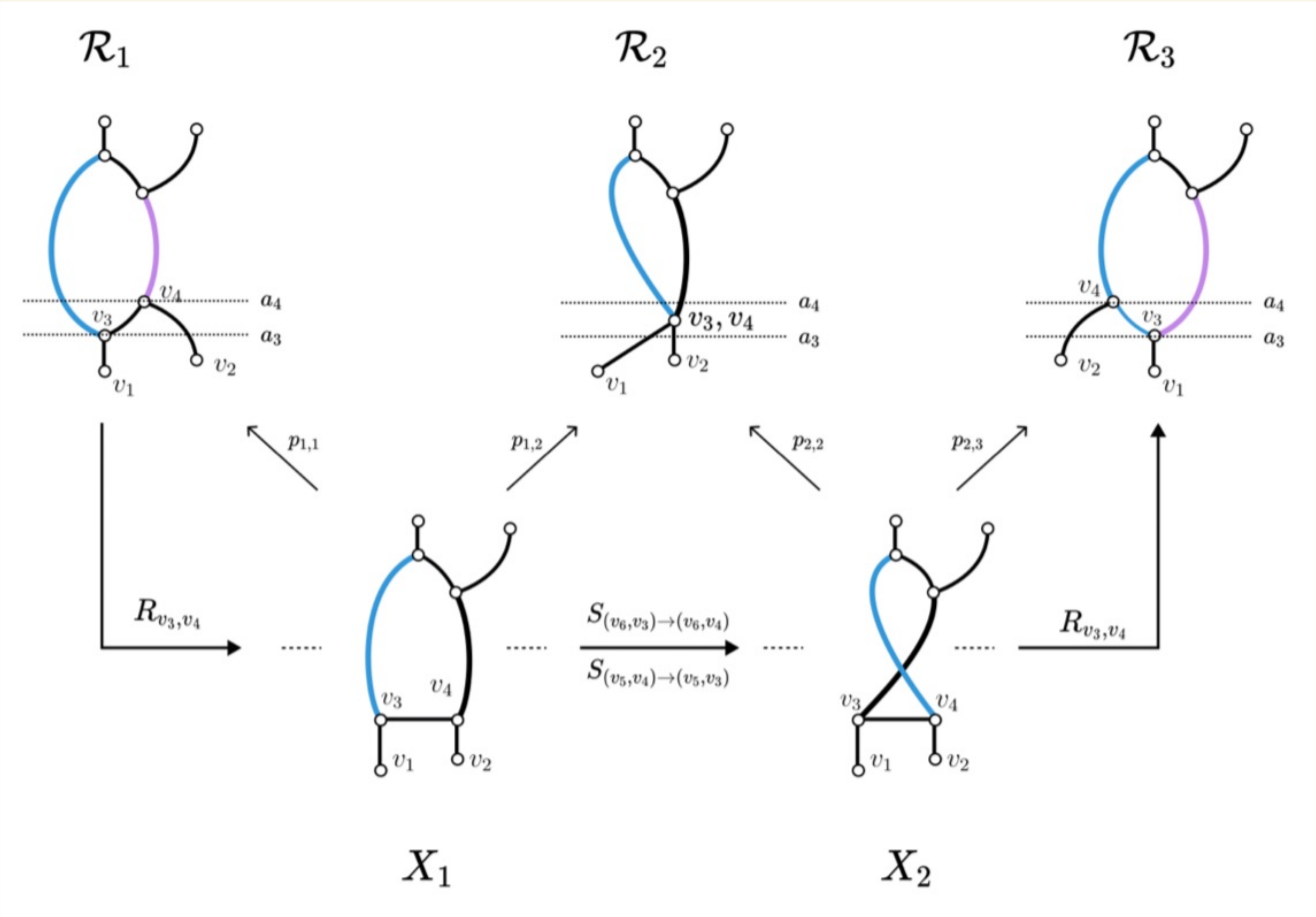
$$\begin{tikzcd}
\mathbb{R} & & \mathbb{R} & & & & \mathbb{R} & & \mathbb{R} \\
\mathcal{R}_f = \mathcal{R}_1 \arrow[u, "\tilde{f}_1"] & & \mathcal{R}_2 \arrow[u, "\tilde{f}_2"] & & \cdots & & \mathcal{R}_{n-1} \arrow[u, "\tilde{f}_{n-1}"] & & \mathcal{R}_n = \mathcal{R}_g \arrow[u, "\tilde{f}_n"] \\
& X_1 \arrow[ul, "p_{1,1}"] \arrow[ur, "p_{1,2}"] & & X_2 \arrow[ul, "p_{2,2}"] \arrow[ur, dotted] & & X_{n-2} \arrow[ul, dotted] \arrow[ur, "p_{n-2,n-1}"] & & X_{n-1} \arrow[ul, "p_{n-1,n-1}"] \arrow[ur, "p_{n-1,n}"] &
\end{tikzcd}$$

Then, pullback (or cone) of this diagram must exist, + can measure how much things more across the maps.

Example:


$\mathcal{R}_1$
$\mathcal{R}_2$
v_4
v_2
u_4
u_2
$\frac{1}{2}|a_4 - a_2|$
$\frac{1}{2}|a_4 - a_2|$
a_4
a_2
$p_{1,1}$
$p_{1,2}$
$R_{v_2,v_4}, I_{e(u_2,u_4)}$
$R_{u_2,u_4}, D_{e(v_2,v_4)}$
x_1
x_2
x_3
x_4
X_1

Alternative:



Categorical edit distance pros + cons:

It is:
- stable
- discriminative
- isomorphism invariant
- path component sensitive

and universal:
for any stable distance S,

$$d_S(R_f, R_g) \leq d_E(R_f, R_g)$$

$\boxed{\text{[Bauer et al 2020]}}$

Unfortunately, no idea how to compute it!

Functional distortion

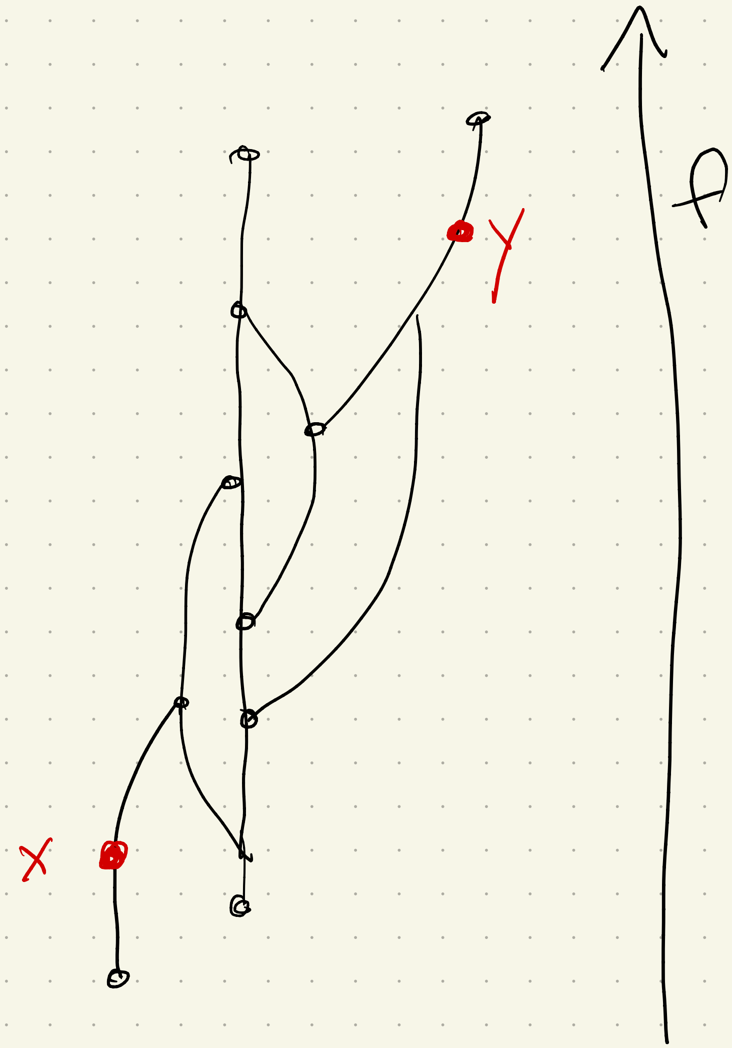
Based on Gromov-Hausdorff idea:

Height of a path:

given path π,

$$\max_{x \in \pi} f(x) - \min_{x \in \pi} f(x)$$

$d(x,y)$ = height of smallest such π



Point distortion:

Fix maps $\Psi : R_f \longrightarrow R_g$

$\Phi : R_g \longrightarrow R_f$

(continuous, but not function preserving)

$$G(\Psi, \Phi) = \{ (x, \Psi(x)) \mid x \in R_f \} \cup \{ (\Phi(y), y) \mid y \in R_g \}$$

Point distortion

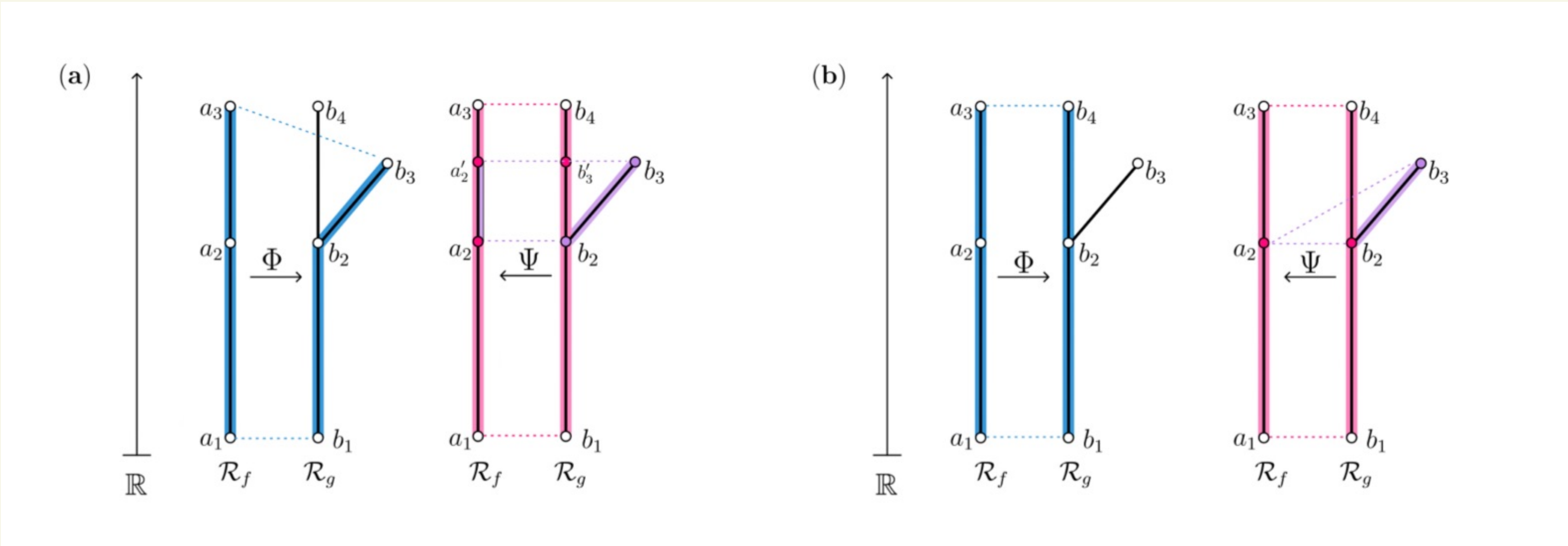
$$\lambda((x,y),(x',y')) = \frac{1}{2} \left| d_f(x,x') - d_g(y,y') \right|$$

Map distortion:

supremum over all pairs of points of the point distortion

Example

Two possible Φ, Ψ maps:



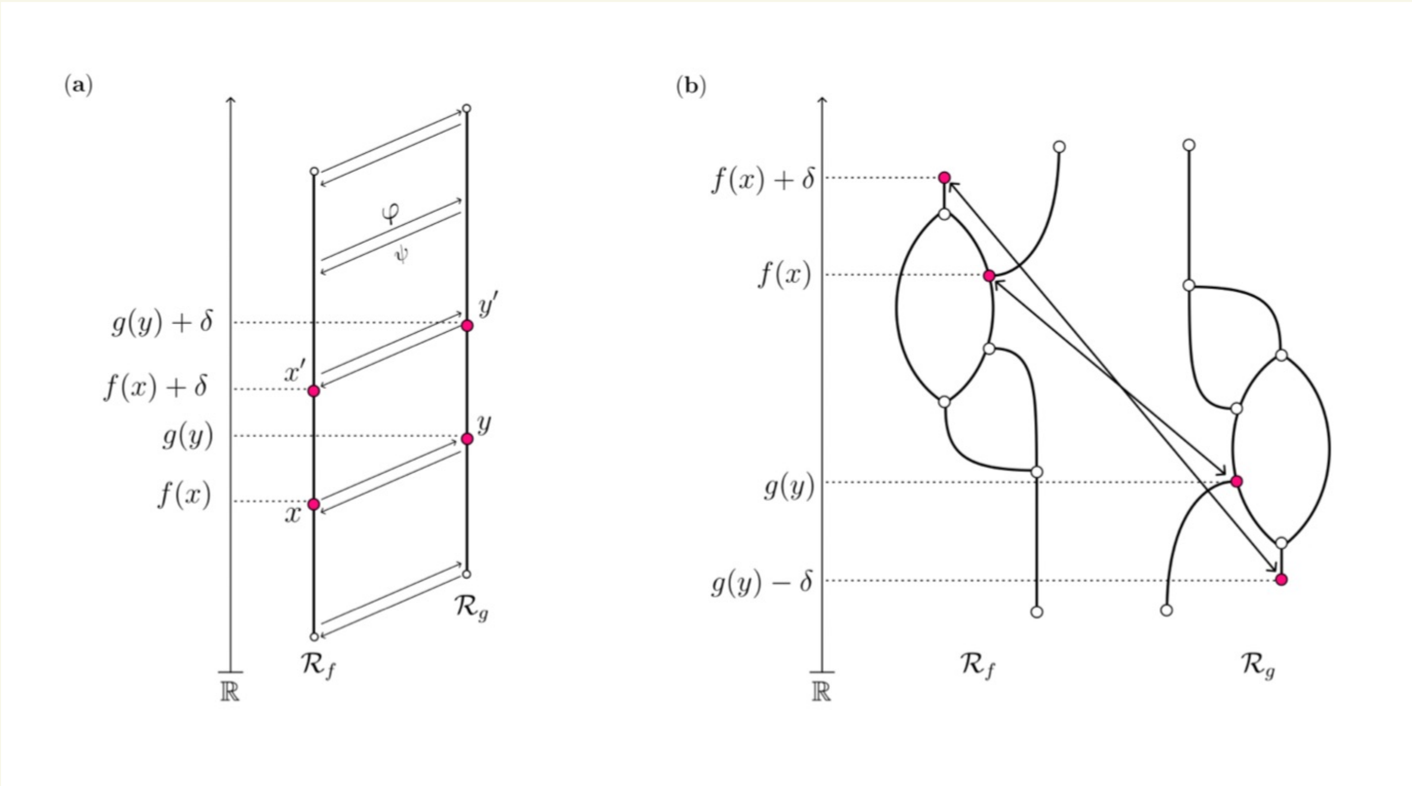
worst pair:
(a_3, b_3) + (a_2, b_2)

worst pair:
(a_2, b_3) and (a_2, b_2)

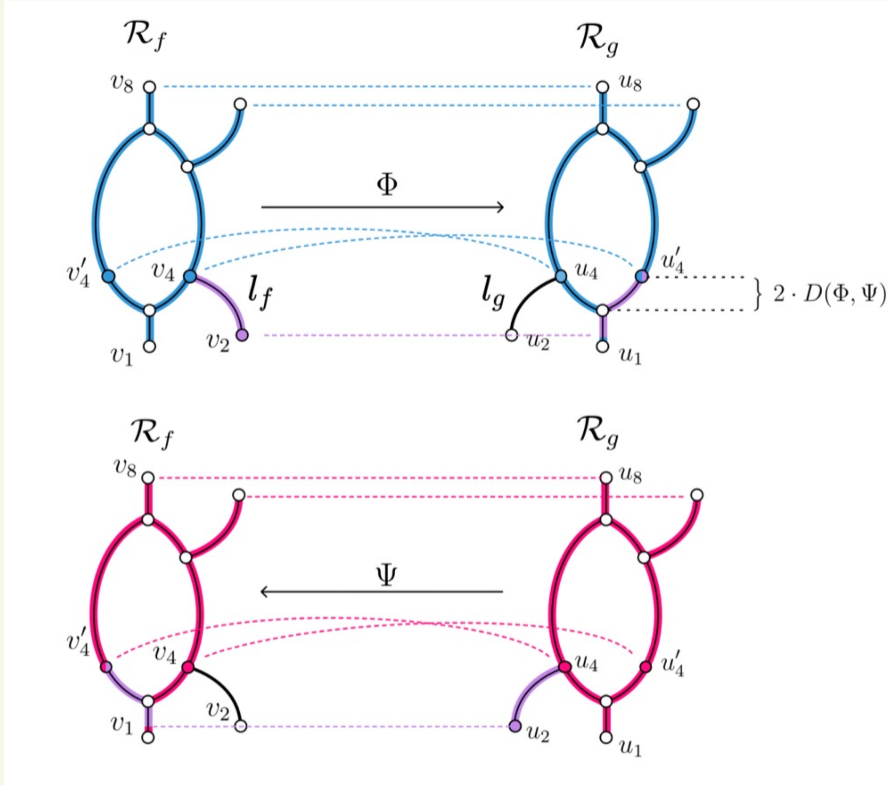
Functional distortion

$$d_{FD}(R_f, R_g) =$$

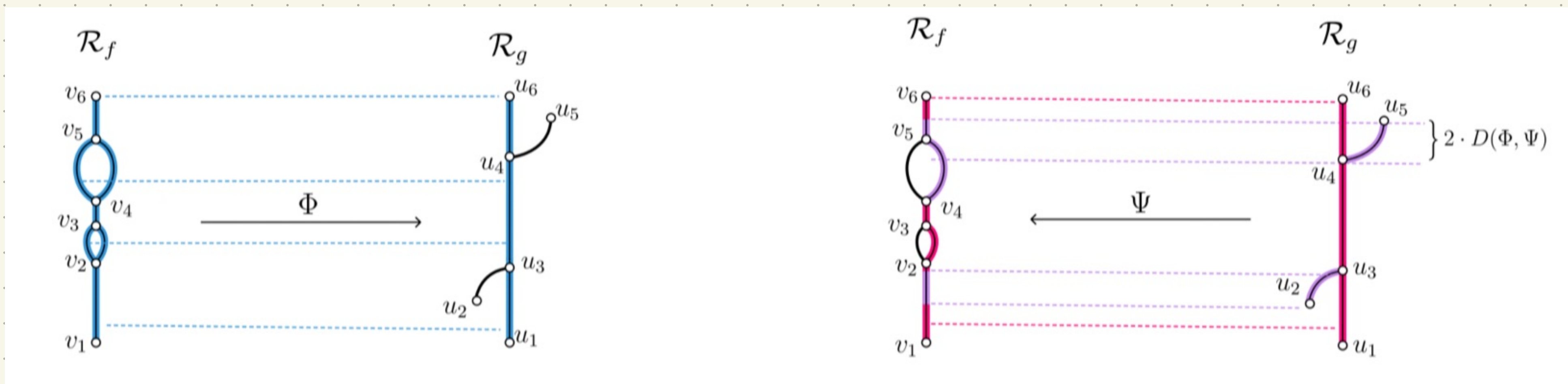
$$\inf_{\Phi, \Psi} \max \{ D(\Phi, \Psi), \|f - g \circ \Phi\|_\infty, \|f \circ \Psi - g\|_\infty \}$$



Examples:



$\mathcal{R}_f$
$\mathcal{R}_g$
v_8
u_8
Φ
v_4'
v_4
l_f
l_g
u_4
u_4'
$2 \cdot D(\Phi, \Psi)$
v_2
v_1
u_2
u_1
$\mathcal{R}_f$
$\mathcal{R}_g$
v_8
u_8
Ψ
v_4'
v_4
u_4
u_4'
v_1
v_2
u_2
u_1



$\mathcal{R}_f$
$\mathcal{R}_g$
v_6
u_6
u_5
v_5
u_4
v_4
Φ
v_3
v_2
u_3
u_2
v_1
u_1
$\mathcal{R}_f$
$\mathcal{R}_g$
v_6
u_6
u_5
v_5
$2 \cdot D(\Phi, \Psi)$
u_4
v_4
Ψ
v_3
v_2
u_3
u_2
v_1
u_1

Functional distortion pros + cons

It is:
- stable
- discriminative

[Bauer et al 2014]

- isomorphism invariant
- path component sensitive

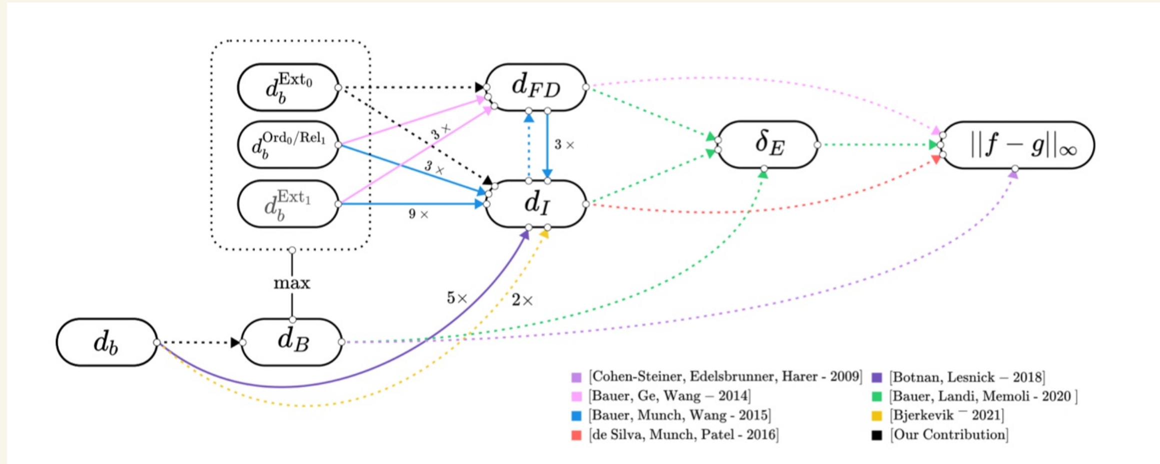
[Bauer et al 2016]

No idea if it's computable in any efficient way.

(or even at all, except on trees!)

Take-aways:

① This are very connected:



In fact, we conclude our survey with a conjecture:

Conjecture 9.3. *The functional distortion distance, interleaving distance, and universal edit distance are equivalent on the space of PL Reeb graphs where the domain $\mathbb{X}$ is simply connected. That is*

$$d_B \leq d_I = d_{FD} = \delta_E.$$

(up to constant factors)

② They do capture different types of features, even though there are connections!

	d_B	d_I	d_I^m	d_{FD}	d_E	δ_E
Stable	[27]	[62]	-	[4]	[35]	[5]
Discriminative	-	[6], 7.24	[22], C.3	[4], 7.20	[35]	[5]
Isomorphism Invariant	-	[62]	[22]	[6], 7.10	[35]	[5]
Path Component Sensitive	-	[62]	[22]	[6]	-	[5], 7.39
Universal	-	-	-	-	-	[5]

Table 1: Table of distance properties. Entry corresponds to a citation where the distance was proved. We supply additional references to statements we contributed in this work which solidify these properties. We denote disproven properties or properties that are not applicable to the given distance with "-".

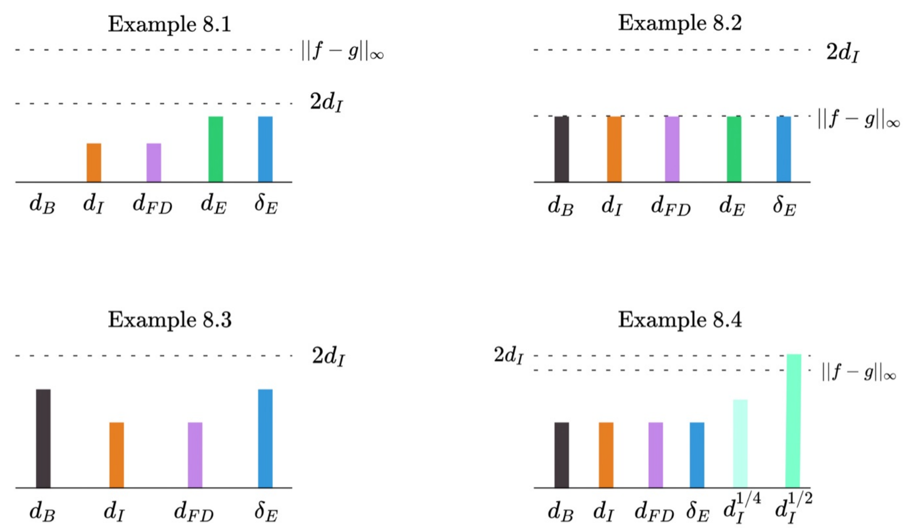


Figure 16: Visual summary of the distance values attained for each example. For Example 4, we show two different choices of m for the truncated interleaving distance to illustrate the affect that m plays on the distance values.

(3) Computation of metrics:

The real negative:
all are "Hard" on general Reeb graphs.