Topological Methods in Visualization

Content

- Overview and Motivation
- Morse-Smale Complexes
- Piecewise Linear Manifolds
- Persistence
- Reeb Graph
- Algorithms
  - Contour Tree
  - Morse-Smale Complex
- Simplification
- Applications
  - Bubble Counting
  - Finger Counting

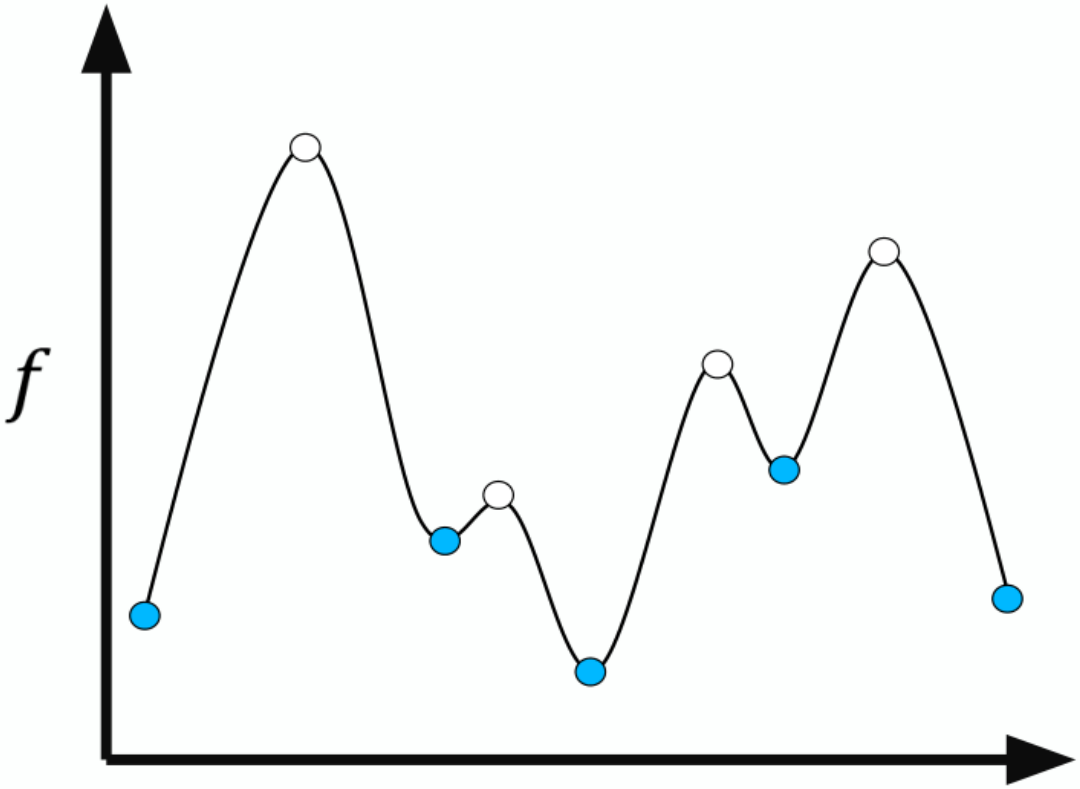
OVERVIEW AND MOTIVATION

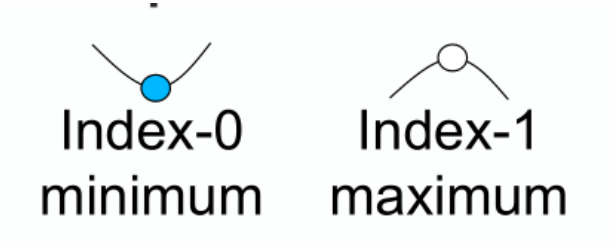
Morse Function

- Domain $\boldsymbol{D}$: $\boldsymbol{R}^{n\in\{1,2,3\}}$ or manifold (curve or surface)
- We start with a smooth function : $f: \boldsymbol{D} \mapsto \boldsymbol{R}$
- A point $\underline{\boldsymbol{p}} \in \boldsymbol{D}$ is called critical with respect to f, if gradient $\nabla f\left(\underline{\boldsymbol{p}}\right) = \vec{\mathbf{0}}$ vanishes. (All other points are called regular)
- A critical point is ordinary/degenerate, if Hessian $\boldsymbol{H}_f\left(\underline{\boldsymbol{p}}\right)$ ($n \times n$-dimensional matrix of 2nd derivatives of f) is non-singular/singular.
- The function f is called Morse, iff all its critical points are ordinary and have distinct function values.
- The index of a critical point is the number of negative eigenvalues of $\boldsymbol{H}_f\left(\underline{\boldsymbol{p}}\right)$ and separates minima, maxima, and saddles

Morse Function

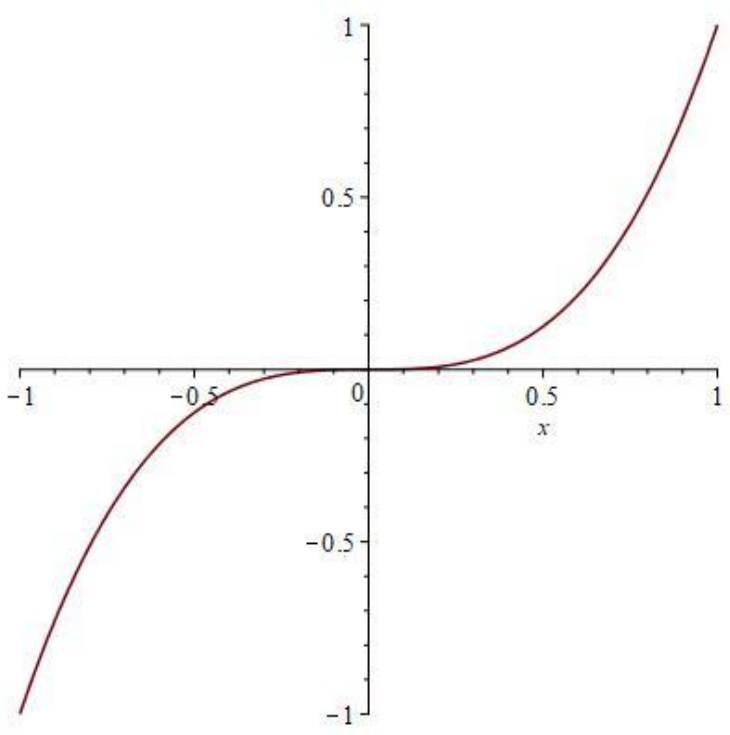
- 1D-Example where gradient is first and Hessian second derivative of function

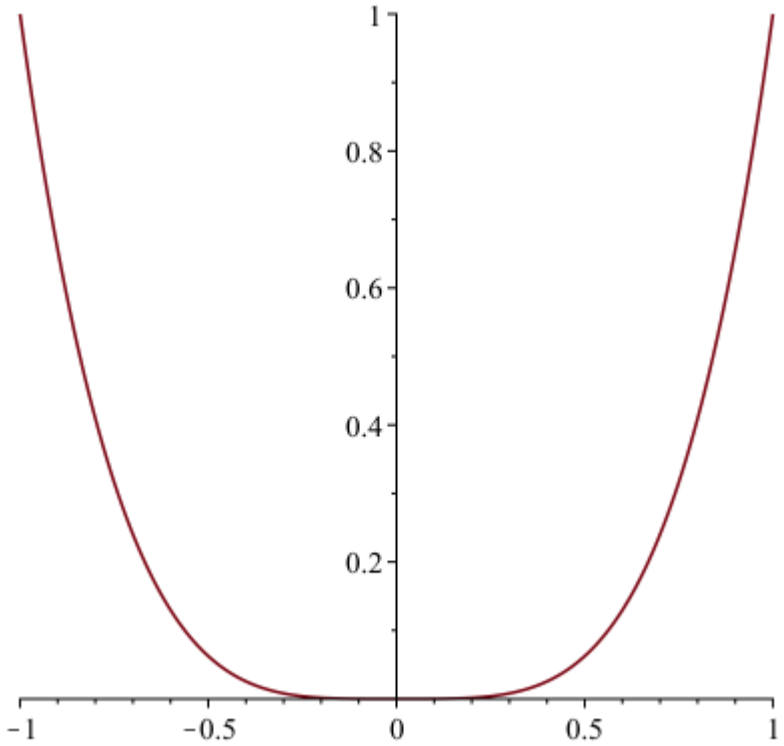




None Morse Functions

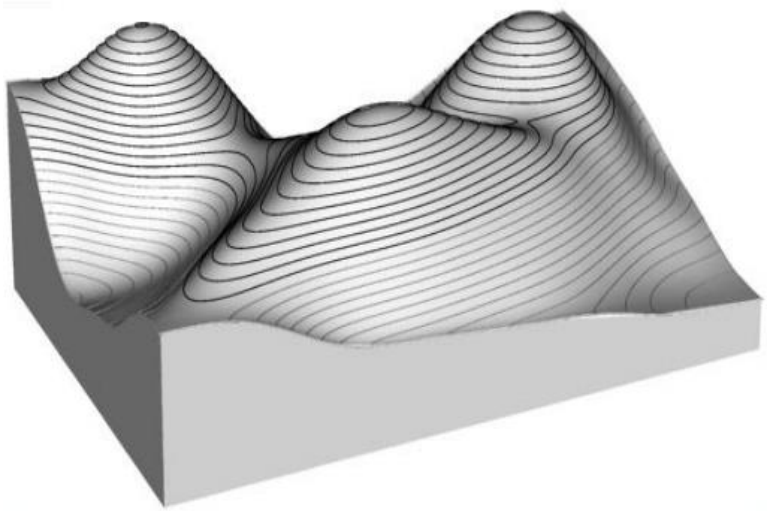
- left: $f(x) = x^3$ with degenerate critical point at $x = 0$ ($f''(0) = 0$) that is a saddle
- right: $f(x) = x^4$ with degenerate critical point at $x = 0$ ($f''(0) = 0$) that is a minimum

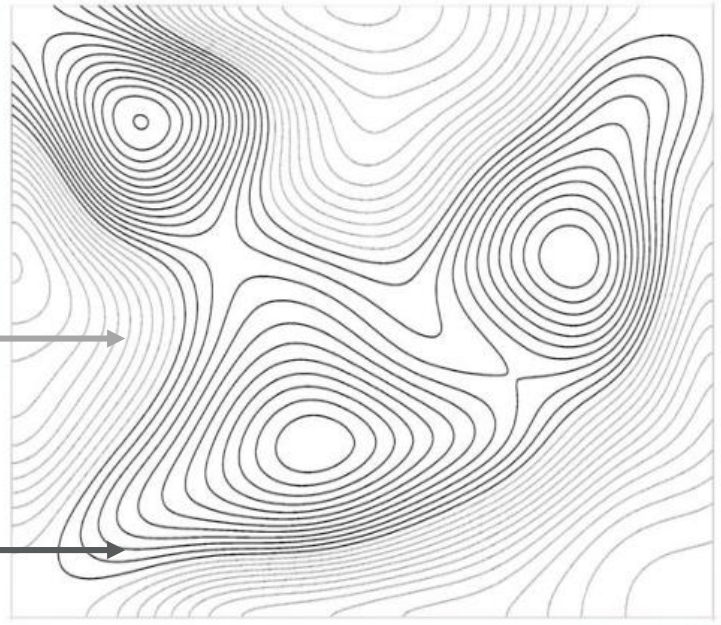




Level Sets

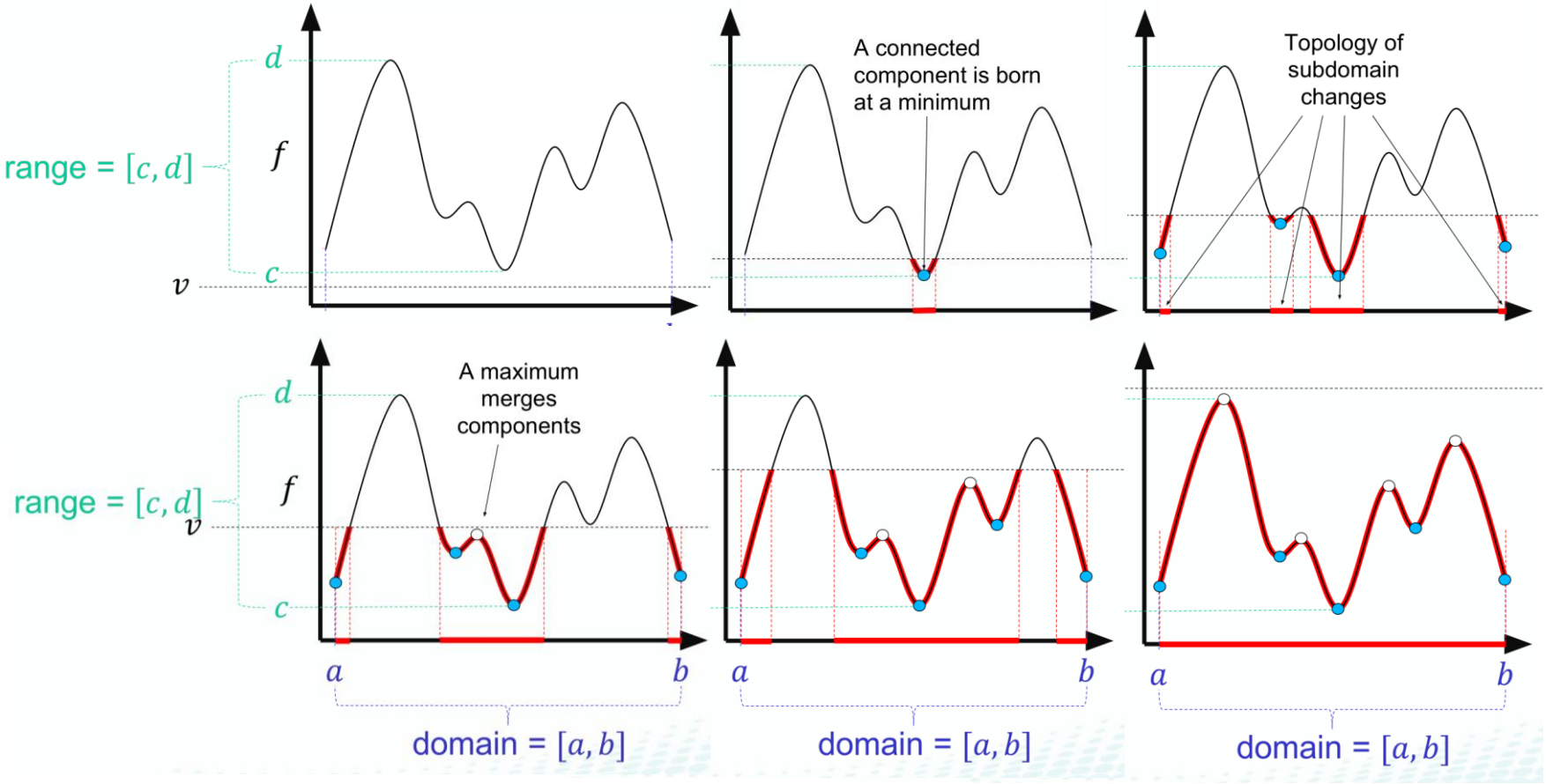
- For values v in the range of f, the preimage or level set
  $f^{-1}(v) = \left\{ \underline{\boldsymbol{p}} \in \boldsymbol{D} \middle| f\left(\underline{\boldsymbol{p}}\right) = v \right\}$
  can consist of several components that are
  - isolated points
  - simple closed contours
  - contours with touching points
- sublevel set:
  $\mathcal{L}_f^-(v) = \left\{ \underline{\boldsymbol{p}} \in \boldsymbol{D} \middle| f\left(\underline{\boldsymbol{p}}\right) \leq v \right\}$
- superlevel set:
  $\mathcal{L}_f^+(v) = \left\{ \underline{\boldsymbol{p}} \in \boldsymbol{D} \middle| f\left(\underline{\boldsymbol{p}}\right) \geq v \right\}$





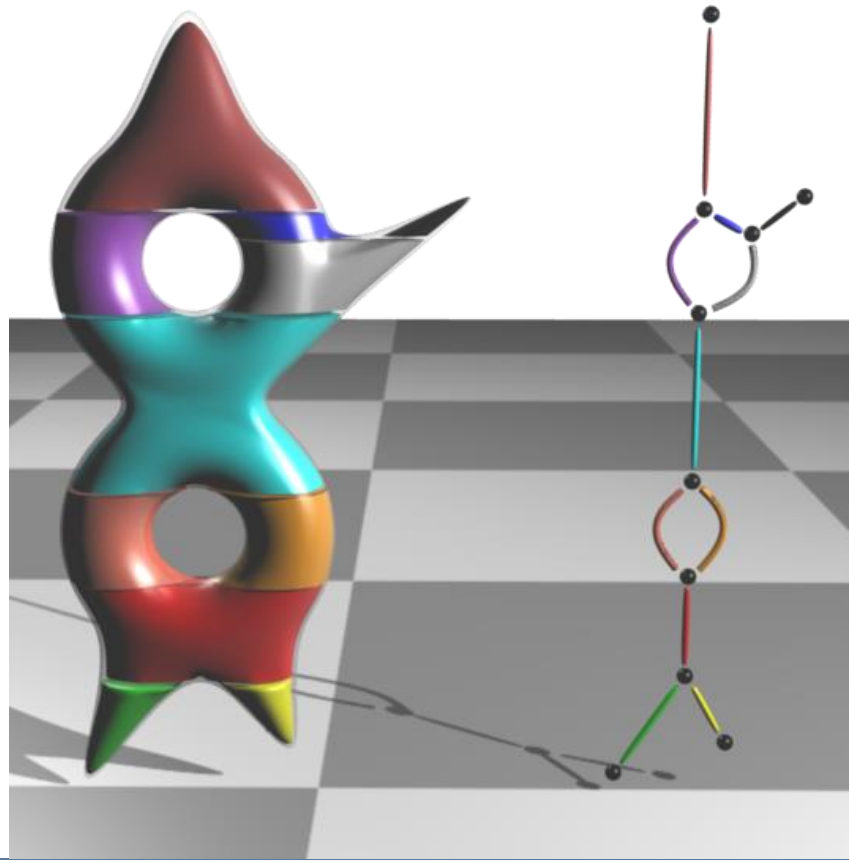
Filtration

- A sweep of the level sets through all $v = -\infty \rightarrow \infty$ is called a filtration.
- During a filtration the topological changes are tracked



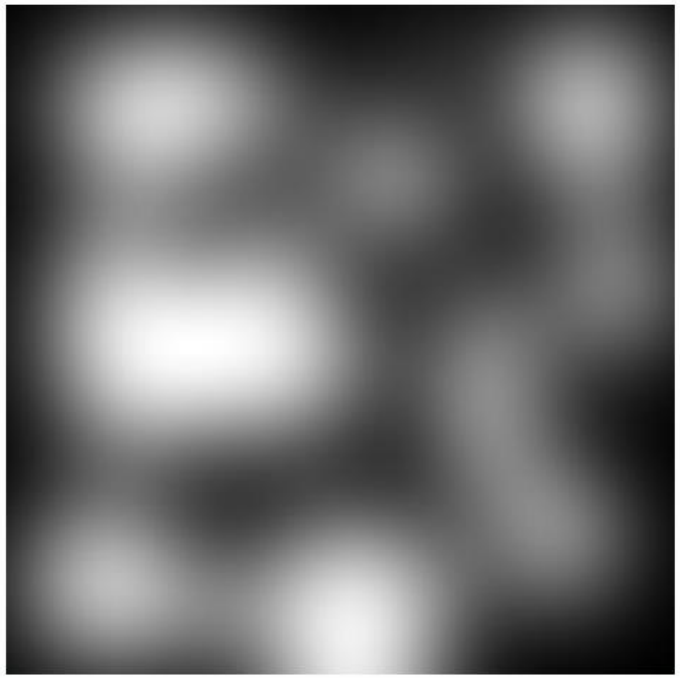
Reeb Graph

- **Def.:** Given domain $\boldsymbol{D}$ and function $f(\boldsymbol{D}) \rightarrow \boldsymbol{R}$, Reeb graph is defined as quotion space of $\boldsymbol{D}$ with respect to equivalence relation $\sim$, with $\underline{\boldsymbol{p}} \sim \underline{\boldsymbol{q}}$ iff $f(\underline{\boldsymbol{p}}) = f(\underline{\boldsymbol{q}}) = v$ and $\underline{\boldsymbol{p}}$, $\underline{\boldsymbol{q}}$ are in same connected component of $f^{-1}(v)$.
- To construct the Reeb Graph one identifies contours of all points that map to same value and splits them into connected components.
- Contour components are tracked over v and form edges of the Reeb Graph
- Topology changes (birth, split, join, death) in the contours arise at critical points, which form nodes of Reeb Graph

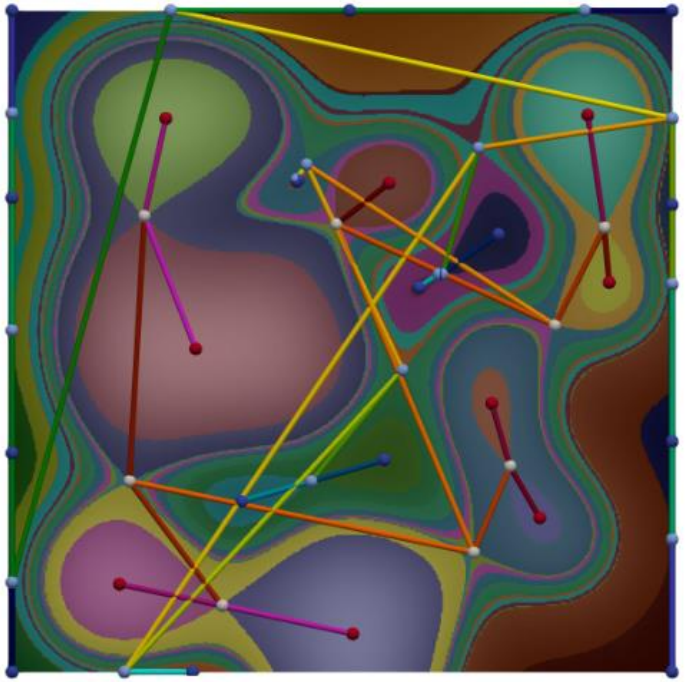


Contour Tree

- If the domain is simply connected (every closed loop can be contracted to a point), the Reeb Graph is a tree, which is called contour tree
- tree edges correspond to areas in domain



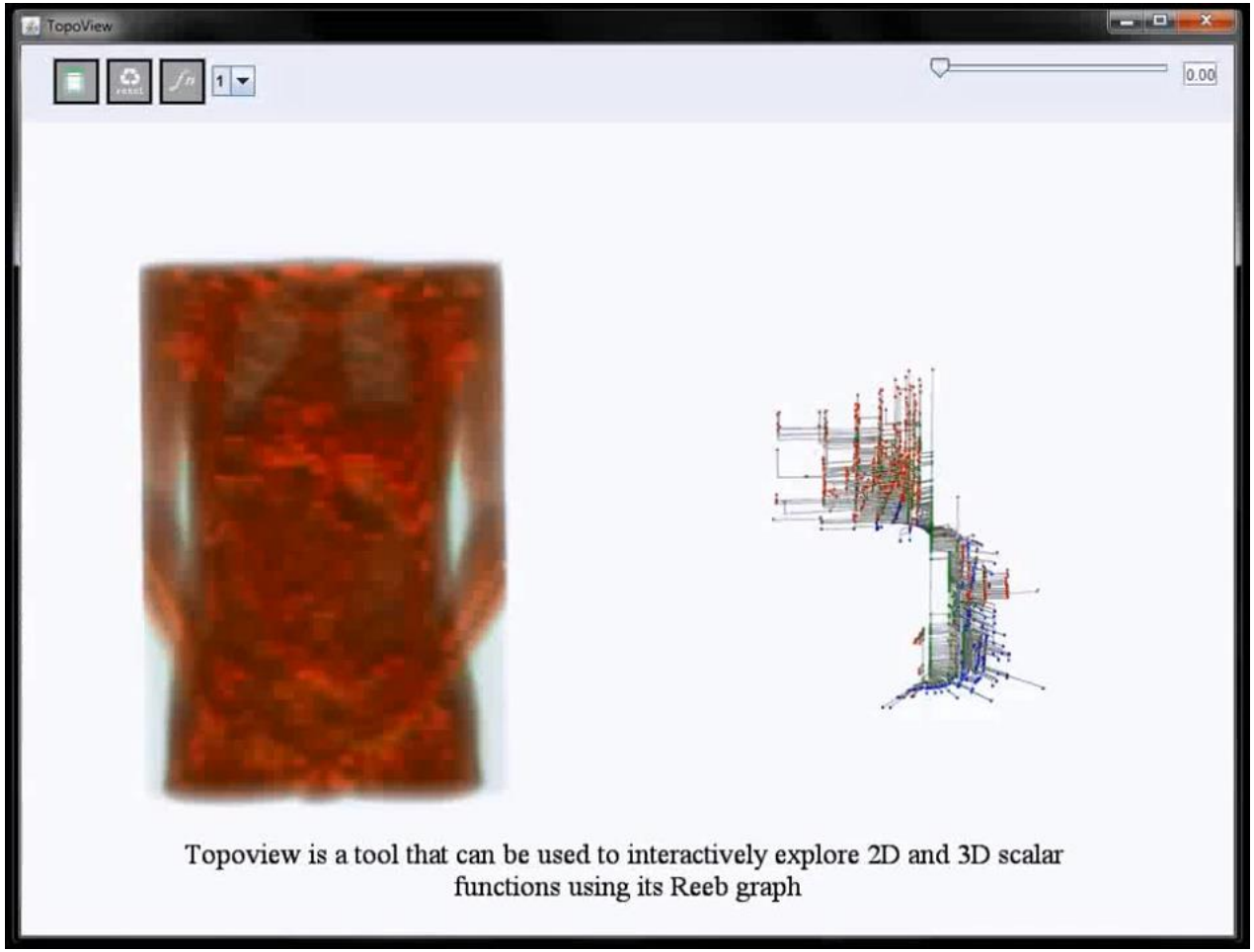
scalar density field over 2D domain



contours colored and contour tree overlayed

Volume Exploration

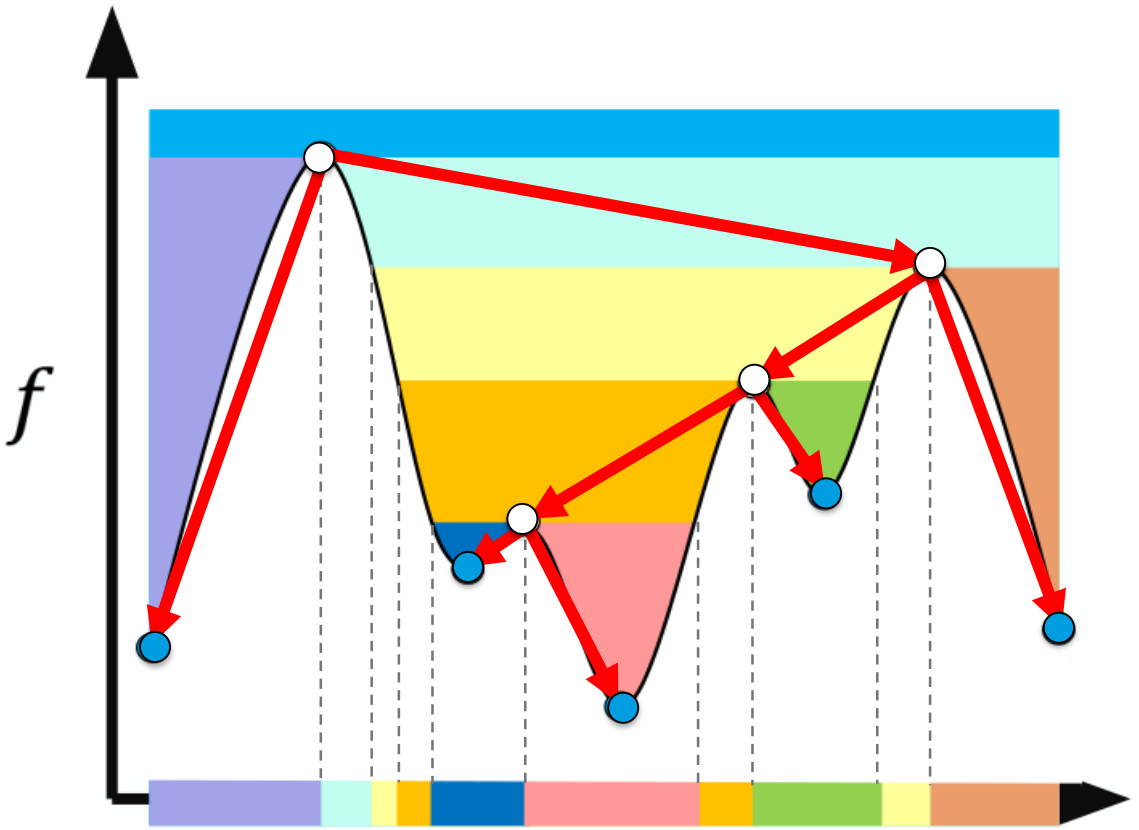
- contour Tree can be used to explore volume data set by selection of tree edges



Topoview is a tool that can be used to interactively explore 2D and 3D scalar functions using its Reeb graph

Join Tree

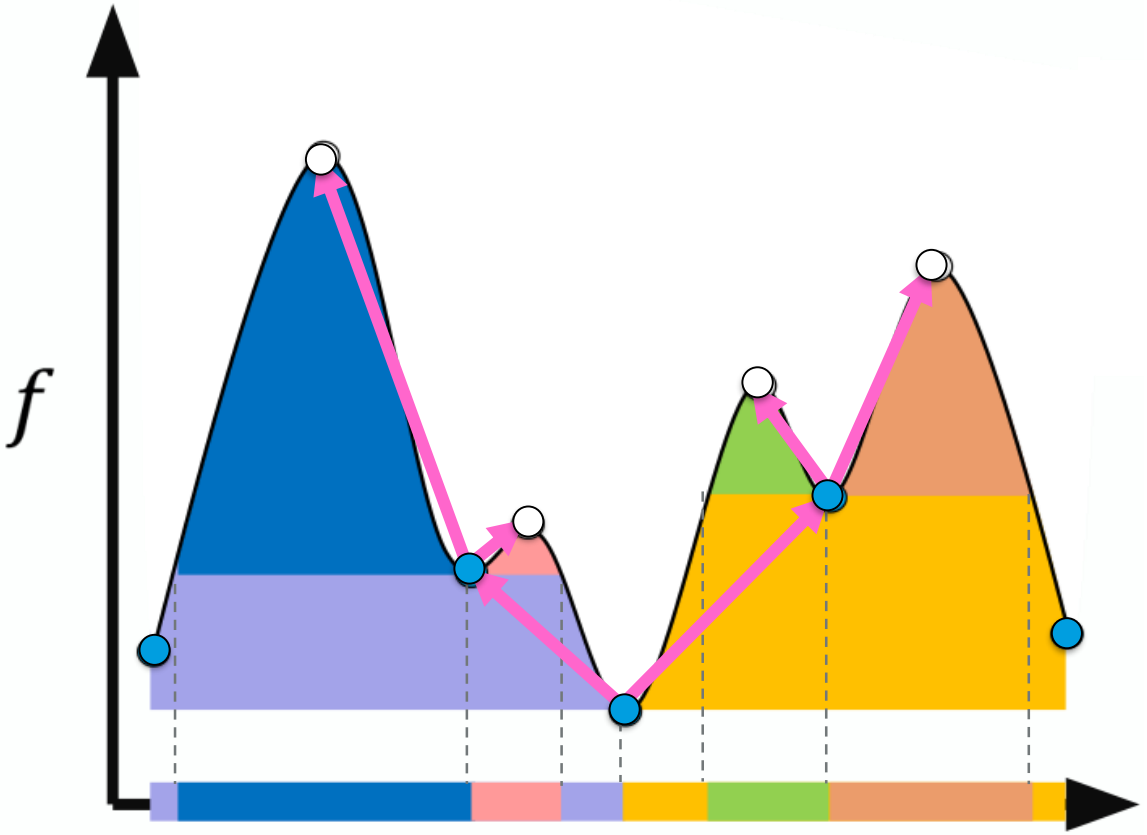
- examines the joining/merging of the superlevel sets during filtration and captures birth and join events of contours



Filteration and generated segmentation

Split Tree

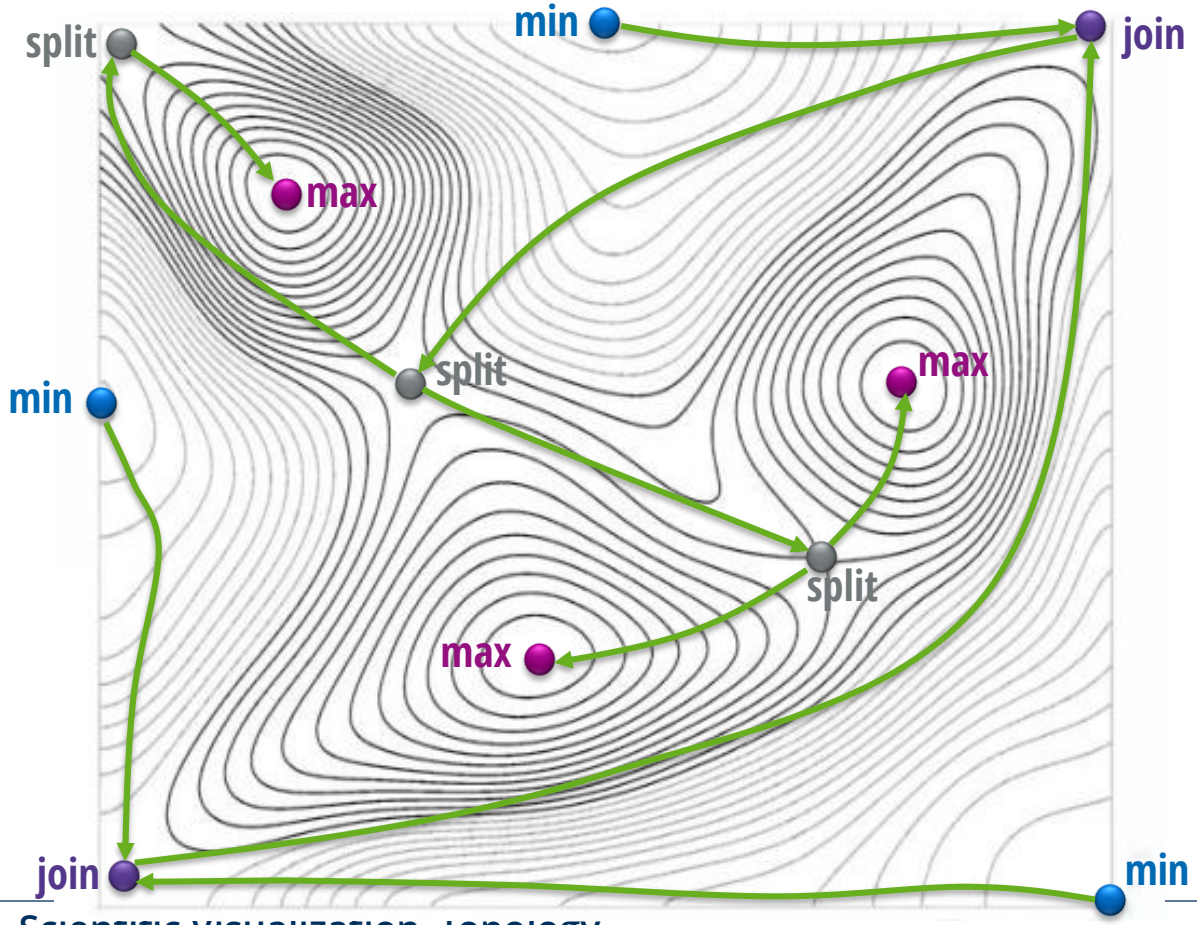
- examines the splitting of the sublevel sets during filtration and captures split and death events of contours



Filteration and generated segmentation

Contour Tree

- Contour tree coincides in 1D with join tree
- In n>1 dimensions the contour tree can be computed from split and join trees according to simple rules

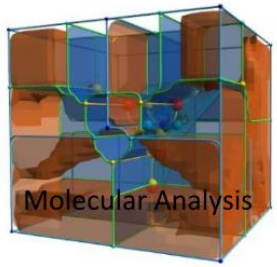


Definition of Topology-Based Vis

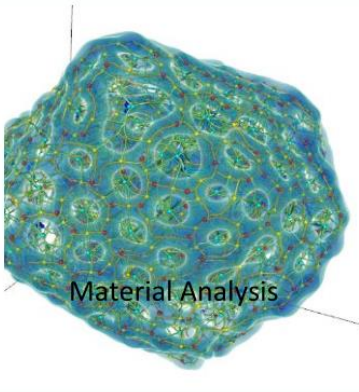
Heine, C., Leitte, H., Hlawitschka, M., Iuricich, F., De Floriani, L., Scheuermann, G., ... & Garth, C. (2016, June). A survey of topology-based methods in visualization. In *Computer Graphics Forum* (Vol. 35, No. 3, pp. 643-667).

- Topology-based visualization uses topological concepts to describe, reduce, or organize data in order to be used in visualization.
- Typical topological concepts are, e.g., topological space, cell complex, homotopy equivalence, homology, connectedness, quotient space.
- Typical visualization uses are, e.g. to highlight data subsets, to provide a structural overview, or to guide interactive exploration.

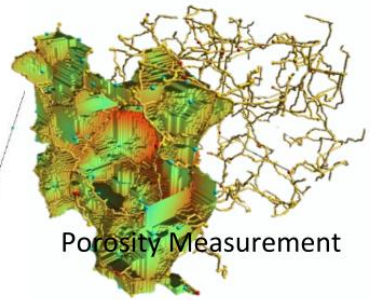
Examples



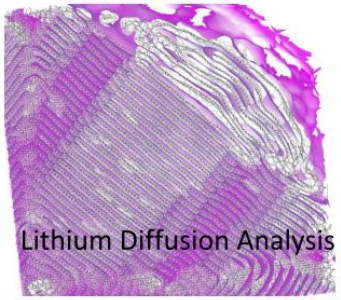
Molecular Analysis



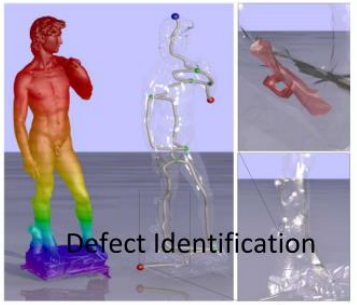
Material Analysis



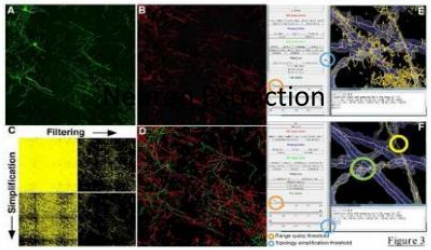
Porosity Measurement



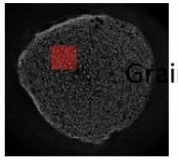
Lithium Diffusion Analysis



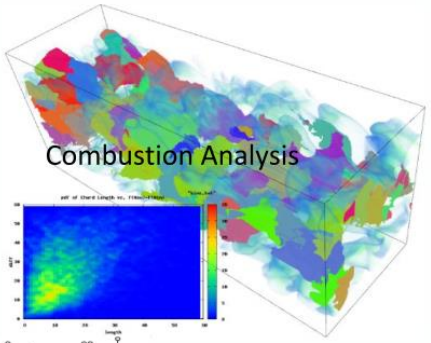
Defect Identification



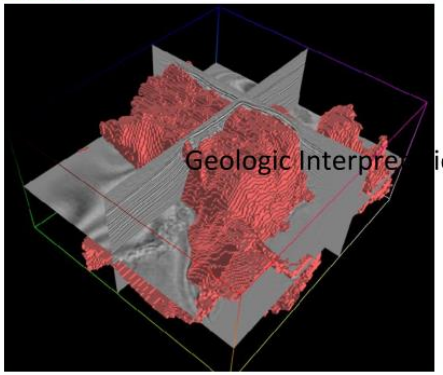
neuron extraction



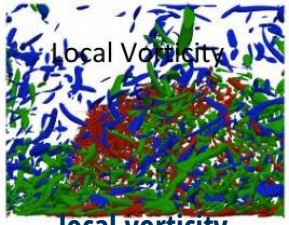
grain analysis



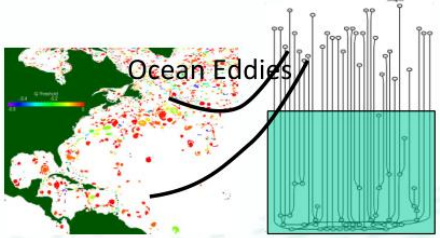
Combustion Analysis



Geologic Interpretation



local vorticity



Ocean Eddies

Classification of Topological Methods

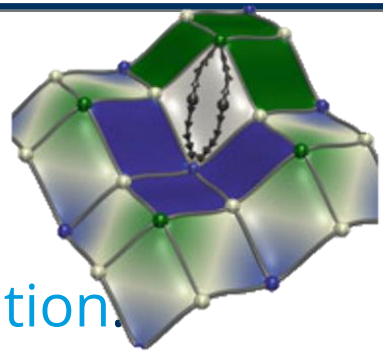
- domain: topological dimension (e.g. 2 for surfaces), discretization (piecewise const, linear, structured, unstructured, combinatorial)
- field type: scalar, vector, tensor, multi-variate (several values over the same domain)
- aspects: certainty, time dependence, topological model, algorithms
- visualization use: highlight, guidance, abstraction, feature identification & tracking, compression

Table 1: *Typology of Topological Models*

	certain		uncertain	
	time-independent	time-dependent	time-independent	time-dependent
scalar	critical points, persistent homology, merge tree, contour tree, Reeb graph, Morse-Smale complex, extremum graph	critical point tracking, dynamic merge tree, dynamic contour tree	mandatory critical points, critical point confidence region	
vector	critical points, invariant sets, separatrices, saddle connectors, Morse decomposition	critical point tracking, LCS, streak line topology, unsteady vector field topology	stable Morse decomposition, uncertain vector field toplogy	uncertain LCS
tensor	degenerate points & lines, separatrices	tensor topology tracking		
multi	Jacobi set, Reeb space joint contour net, Pareto sets			

MORSE-SMALE COMPLEX

Morse-Smale Function



- An integral line is a path $\underline{\boldsymbol{p}}(t)$ in the domain of a Morse function f such that $\frac{\partial \underline{\boldsymbol{p}}}{\partial t}(t) = \nabla f(\underline{\boldsymbol{p}}(t))$.
- $\lim_{t \to -\infty} \underline{\boldsymbol{p}}(t)$ and $\lim_{t \to \infty} \underline{\boldsymbol{p}}(t)$ are called origin and destination.
- origin and destination are critical points
- critical points are formally defined as one-point integral lines such that the domain is partitioned into integral lines
- descending (stable) / ascending (unstable) manifolds of a critical point $\underline{\boldsymbol{p}}$ of index p is p / $(n-p)$-dimensional union of integral lines with $\underline{\boldsymbol{p}}$ as destination / origin
- Union of descending / ascending manifolds forms the Morse Complex of f / $-f$.
- f is a Morse-Smale function, if Morse complexes of f and $-f$ only intersect transversally.

Morse-Smale Complex – Definition

- The Morse-Smale complex is the intersection of the Morse complexes of f and $-f$.
- It captures features that are related to the gradient.

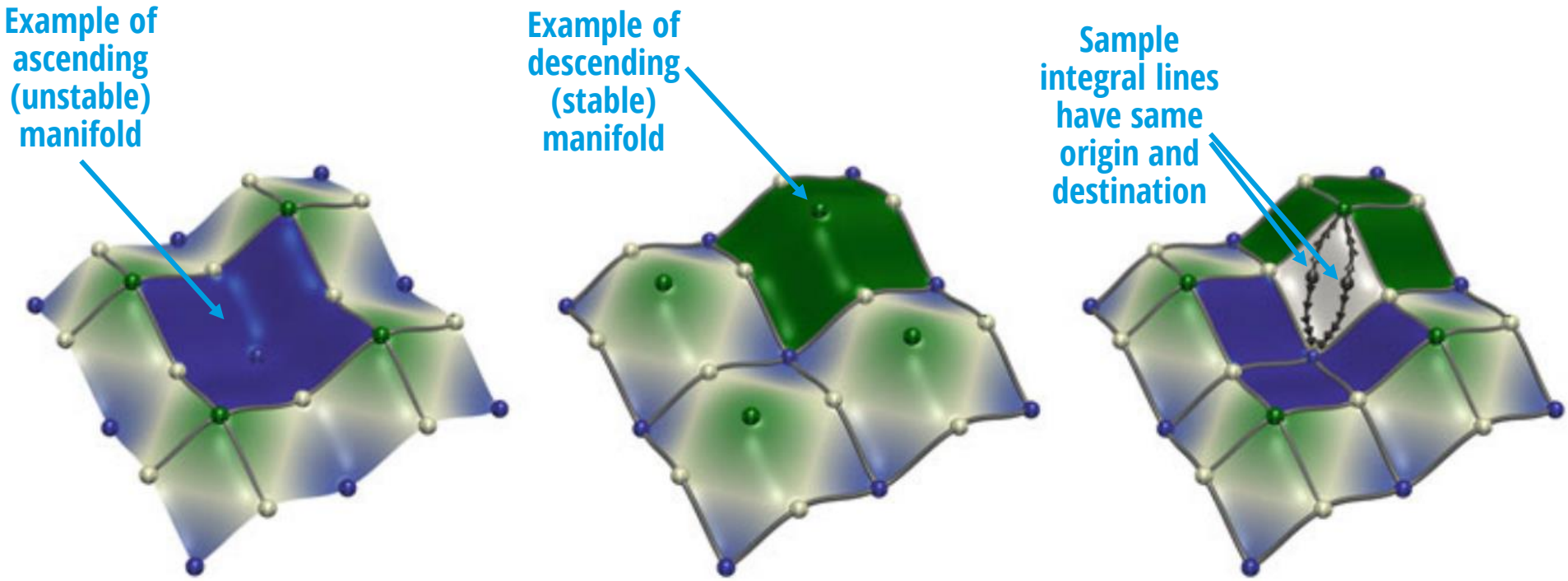
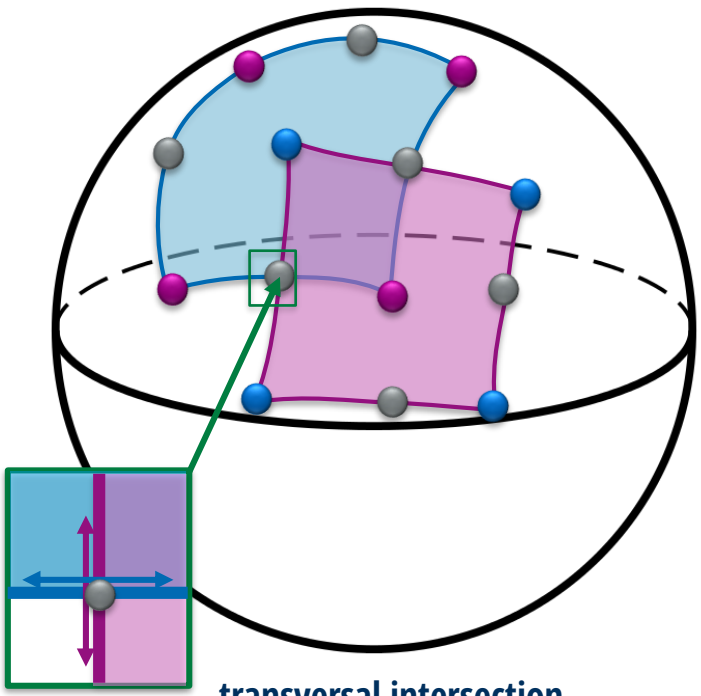


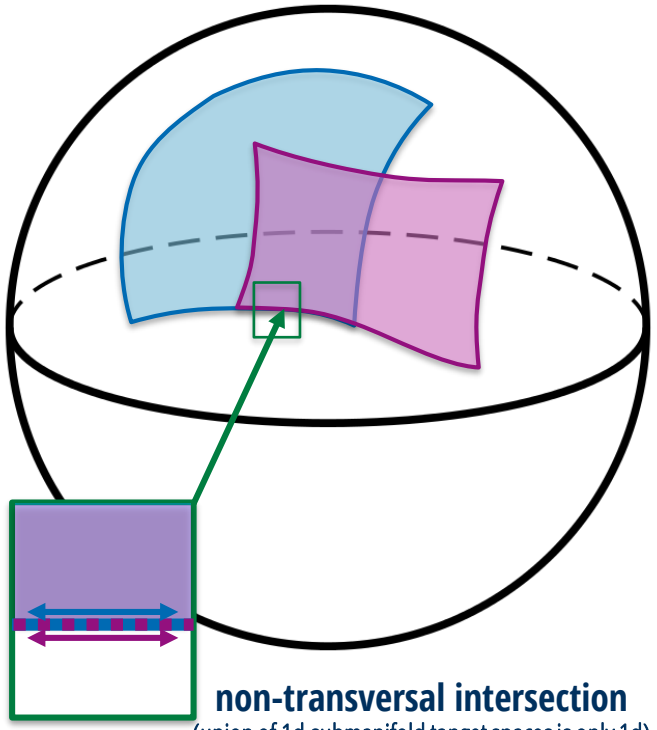
Fig. 2.18 Ascending (left) and descending (center) manifolds and Morse-Smale complex (right) of a PL Morse scalar field f defined on a PL 2-manifold

Transversal Intersection

- Two submanifolds of a given finite-dimensional smooth manifold are said to **intersect transversally** if at every point of intersection, their separate tangent spaces at that point together generate the tangent space of the ambient manifold at that point.



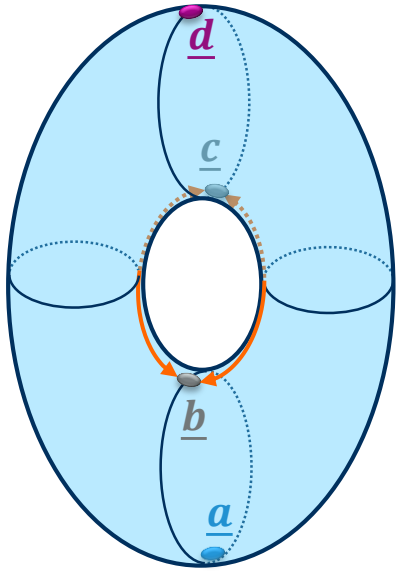
transversal intersection
(union of 1d submanifold tanget spaces is only 2d)



non-transversal intersection
(union of 1d submanifold tanget spaces is only 1d)

Non-Transversality

- The torus standing on one side with the height as function is the standard example of a function that is not a Morse-Smale function
- It has a minimum, two saddles and a maximum. The two saddle are connected together by the two inner half circles.
- The descending manifold of the upper saddle intersects with the ascending manifold of the lower saddle along these half circles yielding a non-transversal intersection



stable manifolds:
a ... isolated point a
b ... 1d circle through b and a excluding a
c ... 1d circle through c and b excluding b
d ... 2d surface of torus excluding the 2 circles

unstable manifolds:
d ... isolated point d
c ... 1d circle through c and d excluding d
b ... 1d circle through b and c excluding c
a ... 2d surface of torus excluding the 2 circles

intersection:
0d points: a , b, c and d
1d open halves of three circles
(non traversal in central circle)
2d surface of torus excluding the 3 circles

Morse-Smale Complex

- Quadrangle Lemma: For 2D manifolds, the Morse–Smale complex is composed of quadrangles with vertices of index 0, 1, 2, 1, in this order around the region. The boundary is possibly glued to itself along vertices and arcs.

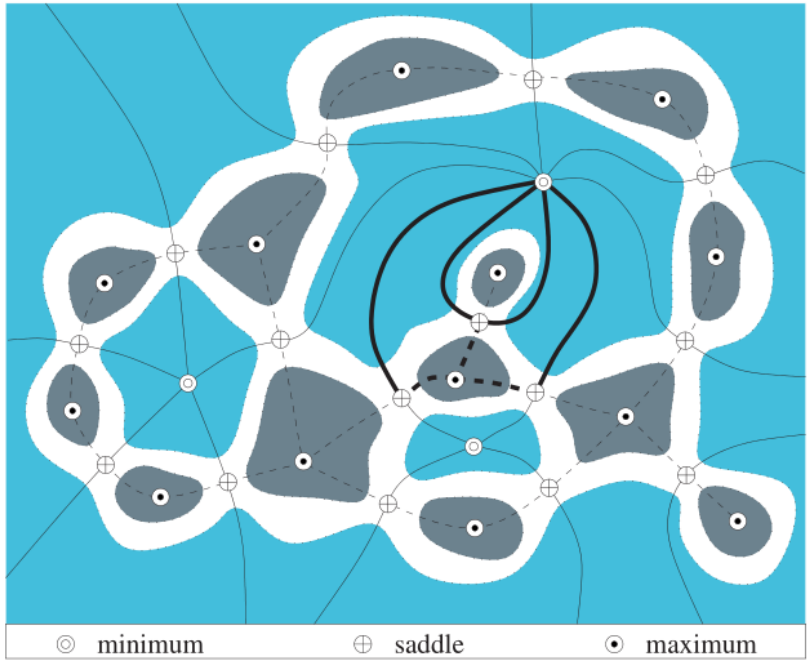


Fig. 1. A Morse–Smale complex with solid stable 1-manifolds and dashed unstable 1-manifolds. In drawing the dotted iso-lines we assume that all saddles have height between all minima and all maxima.

PIECEWISE LINEAR MANIFOLDS

Simplicial Complex

- In SciVis and Computer Graphics the most common domain discretization used for topological analysis are simplicial complexes.
- A d-simplex is the convex hull of $d + 1$ affinely independent points in $\boldsymbol{R}^n$ and is d-dimensional
- Any subset τ of points of a d-simplex σ is called a face what is denoted by $\tau \leq \sigma$.
- A simplicial complex is a finite collection of simplices that contains all faces of any simplex and where the intersection of two simplices is empty or a face of both.
- The number of d-simplices in a simplicial complex is abbreviated by $\#_d$.

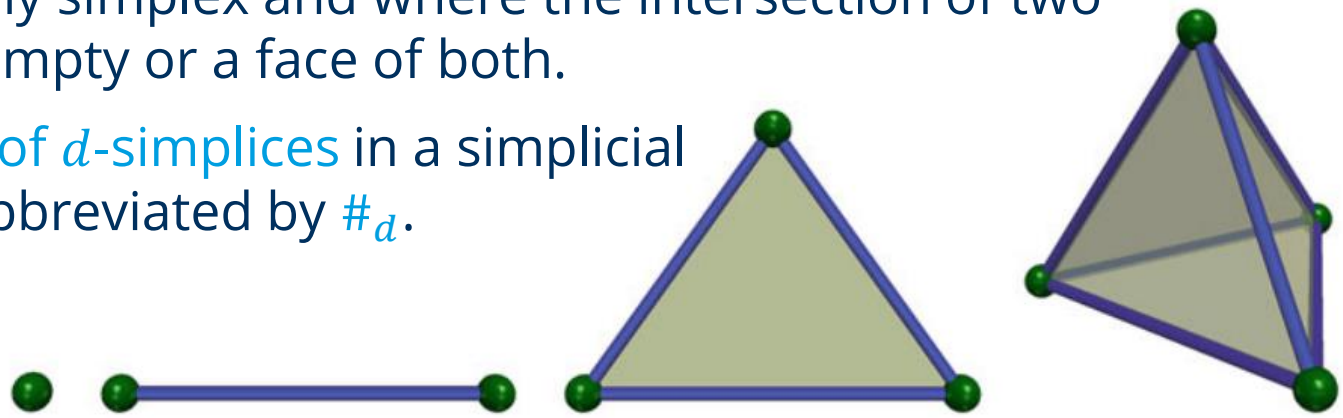


Fig. 2.3 Illustrations of 0 (green), 1 (blue), 2 (white) and 3-simplices (transparent), from left to right, along with their faces

Stars and Links

- The star $\mathrm{St}(\sigma)$ of a simplex σ in a simplicial complex $\mathbb{K}$ is the set of simplices that contain σ: $\mathrm{St}(\sigma) = \{\tau \in \mathbb{K} | \sigma \leq \tau\}$
- The link $\mathrm{Lk}(\sigma)$ of $\sigma \in \mathbb{K}$ is the set of faces of $\mathrm{St}(\sigma)$ simplices that are disjoint from σ:

$$\mathrm{Lk}(\sigma) = \{\tau \leq \rho \in \mathrm{St}(\sigma) | \tau \cap \sigma = \emptyset\}$$

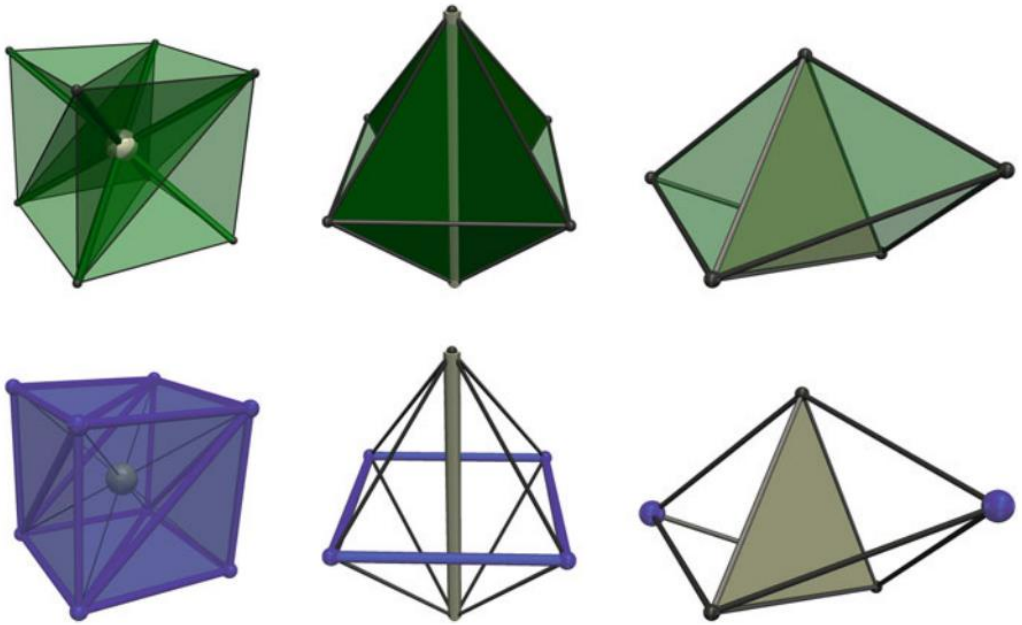


Fig. 2.4 Illustrations of stars (green, top) and links (blue, bottom) for 0, 1 and 2-simplices (white, from left to right) of a 3-dimensional simplicial complex

Piecewise Linear Manifolds

- A triangulation of a manifold is called piecewise manifold and represents a topological space

Fig. 2.6 Example of PL 3-manifold (left, right: clipped view)

Topological Analysis on PL Manifolds

- Topological analysis of linear functions over PL manifolds is again based on filtrations that form nested subsets of the domain simplices corresponding to sublevel sets
- Analysis is based on the tracking of changes in topological features of the domain subsets
- To define topological features we introduce homotopy theory that allows to compare domain subsets and defines topological features that are invariant to continuous morphing of the domain.
- The resulting feature counts are called Betti numbers, which are tracked over filtrations and are additionally related to critical point counts as well as simplex counts.

Homotopy – Basics

- A topological space $\mathbb{X}$ is connected if any two points can be connected by a path inside of the $\mathbb{X}$.
- The maximally connected subsets of a topological space $\mathbb{X}$ are called its connected components.
- A homotopy between two continuous functions $f, g: \mathbb{X} \to \mathbb{Y}$ is a continuous function $H: \mathbb{X} \times [0,1] \to \mathbb{Y}$ such that $\forall \underline{\boldsymbol{x}} \in \mathbb{X}: f(\underline{\boldsymbol{x}}) = H(\underline{\boldsymbol{x}}, 0) \wedge g(\underline{\boldsymbol{x}}) = H(\underline{\boldsymbol{x}}, 1)$.
- If there exists a homotopy then f and g are said to be homotopic.
- Basically, a homotopy is a continuous morph between two functions that does not change their topology.
- Homotopy and homotopic can also be applied to paths and surfaces in a topological space

Homotopy – Simply Connected

- A topological space is simply connected if it is connected and if for any two points, any two paths between them are homotopic, i.e. there exists a continuous mapping morphing one into the other.

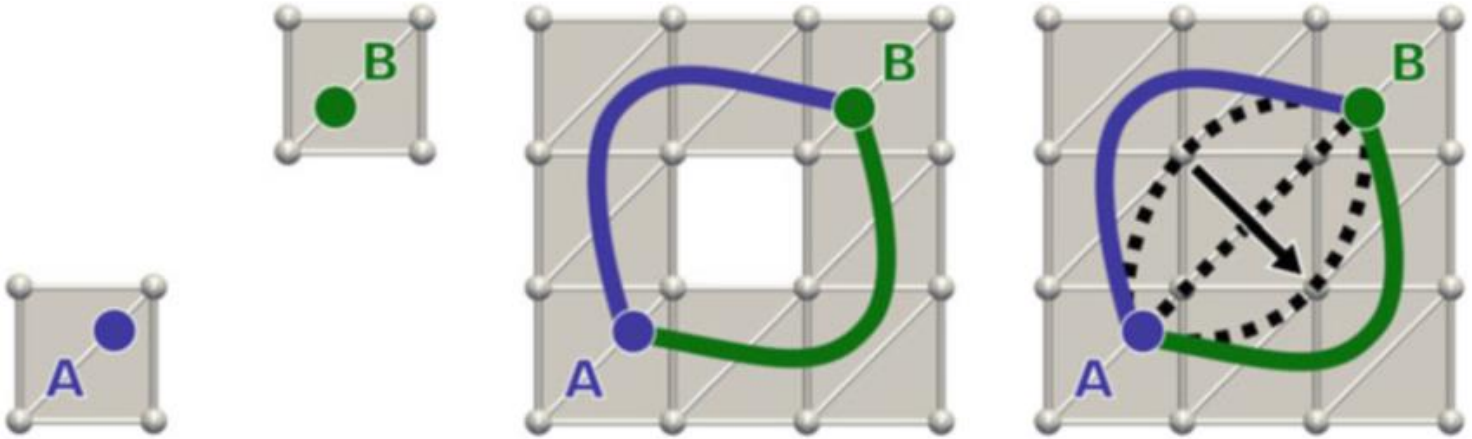
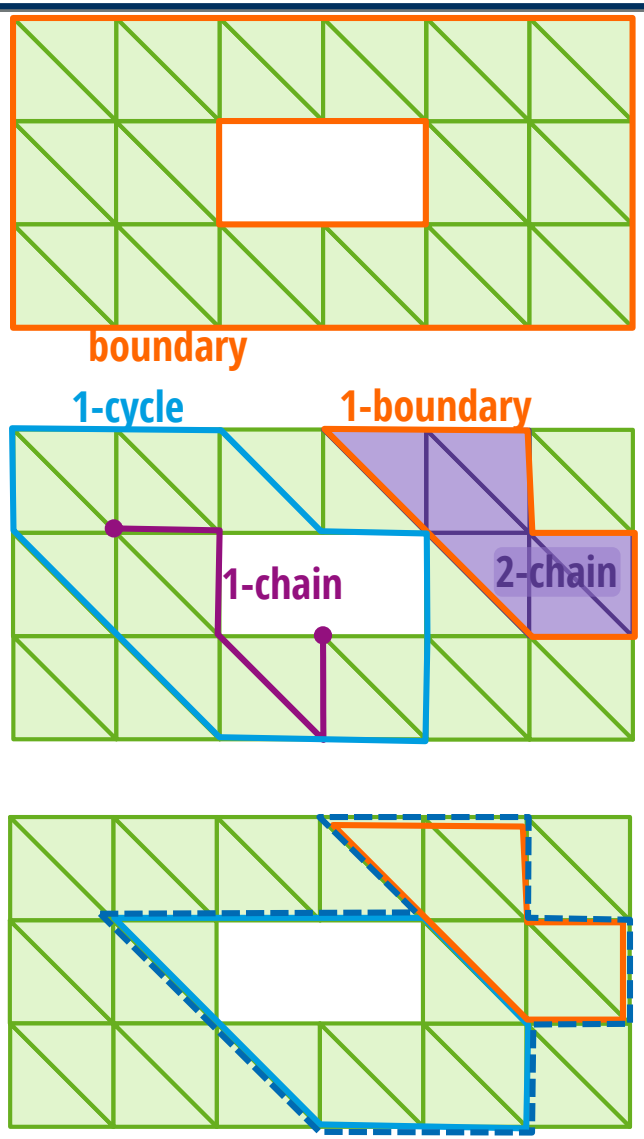


Fig. 2.7 Examples of disconnected, connected and simply connected domains (from left to right)

Homotopy – Boundary & Cycles

- The boundary $\partial\mathbb{X}$ of a topological space $\mathbb{X}$ is the complement of its interior which is composed of all elements with open neighborhood
- A d-chain is a subset of the d-simplices, represented as $\#_d$ dimensional vector of binary numbers modulo 2.
- A d-cycle is a boundary less d-chain
- A d-boundary is the boundary of a $(d+1)$-chain and always a d-cycle.
- vector addition on d-cycles | d-boundaries forms the groups $Z_d(\mathbb{X})$ | $B_d(\mathbb{X})$
- the dth homology group $H_d(\mathbb{X})$ is the factor group $H_d(\mathbb{X}) = Z_d(\mathbb{X})/B_d(\mathbb{X})$
- Intuitively, d-cycles are equivalent, or homologous, if they can be continuously transformed into each other



Betti Numbers

- The dth Betti number β_d is the rank of its dth homology group. Each group member is a topological feature that can be represented by a generator cycle (point pair, loop, surface, volume)

for piecewise linear 3-manifolds

- β_0 … number of connected components
- β_1 … number of handles
- β_2 … number of voids
- β_3 … number of non-orientable components (always 0 in 3D)

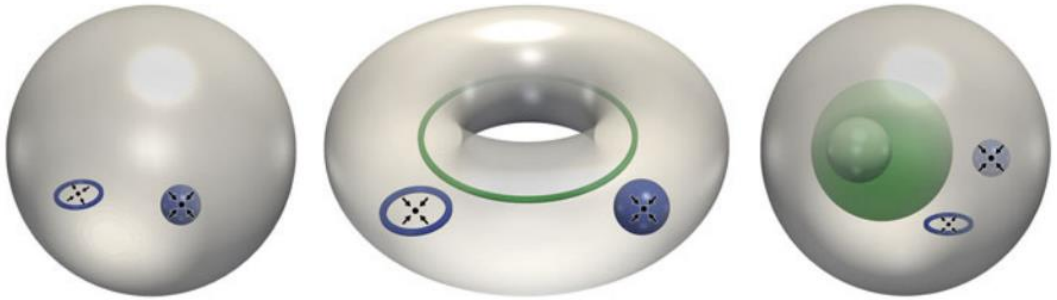


Fig. 2.8 Examples of PL 3-manifolds with varying Betti numbers. From left to right: a 3-ball, a solid torus, a 3-ball with a void. From left to right, $(\beta_0, \beta_1, \beta_2)$ is equal to $(1, 0, 0)$, $(1, 1, 0)$, and $(1, 0, 1)$). Generators are displayed in green, while examples of non-generator p-cycles are displayed in blue

Euler Characteristics

- The Euler characteristic $\chi(\mathbb{T})$ of a triangulation $\mathbb{T}$ of a topological space $\mathbb{X}$ of dimension d is the alternating sum of its Betti numbers β_i and equal to the alternating sum of its i-simplex count $\#_i$:

$$\sum_{i=0}^{d}(-1)^i\beta_i(\mathbb{T}) = \chi(\mathbb{T}) = \sum_{i=0}^{d}(-1)^i\#_i(\mathbb{T})$$

Piecewise Linear Scalar Field

- Let $\hat{f}$ be a function that maps the 0-simplices of a triangulation $\mathbb{T}$ to $\mathbb{R}$. Let f be the piecewise linear barycentric interpolated $\hat{f}$, i.e. for any point $\underline{\boldsymbol{p}}$ inside any d-simplex σ $\mathbb{T}$ we have: $f\left(\underline{\boldsymbol{p}}\right) = \sum_{i=0}^{d} \alpha_i \hat{f}\left(\tau_0^i\right)$, where τ_0^i is the ith 0-face of σ.
- f is called a piecewise linear (PL) scalar field.

Fig. 2.9 Example of PL scalar field f defined on a PL 3-manifold $\mathscr{M}$. From left to right: restriction $\hat{f}$ of f on the 0-simplices of $\mathscr{M}$, f (the color coding denotes the linear interpolation within each simplex), clipped view of f

Simulation of Simplicity

- In case a PL scalar field has several identical values at 0-simplices, it is not a Morse function
- The concept of simulation of simplicity exploits the linear order of the unique storage locations for the 0-simplex values to disambiguate the identical values
- Let $o(p)$ be the memory offset of a 0-simplex p. Then the comparison between two 0-simplices p and q is implemented according to:

$$p < q \Leftrightarrow f(p) < f(q) \vee \big(f(p) = f(q) \wedge o(p) < o(q)\big)$$

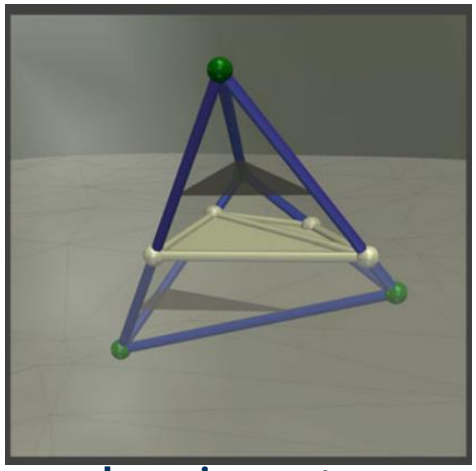
Gradient, Integral Lines & Contours

- The gradient of a piecewise linear scalar field is constant on each d-simplex:

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x_1 & x_2 & x_2 \\ y_1 & y_2 & y_3 \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \end{pmatrix}$$

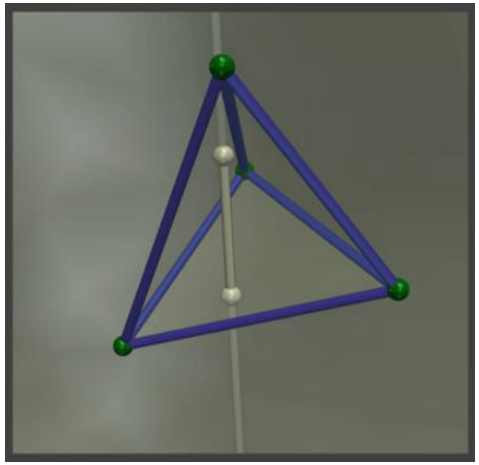
$$f = (f_0 \quad f_1 \quad f_2) \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \end{pmatrix} = (f_0 \quad f_1 \quad f_2) \begin{pmatrix} x_1 & x_2 & x_2 \\ y_1 & y_2 & y_3 \\ 1 & 1 & 1 \end{pmatrix}^{-1} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

$$\nabla f = (f_0 \quad f_1 \quad f_2) \begin{pmatrix} x_1 & x_2 & x_2 \\ y_1 & y_2 & y_3 \\ 1 & 1 & 1 \end{pmatrix}^{-1} \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix} = const$$



planar iso-contour

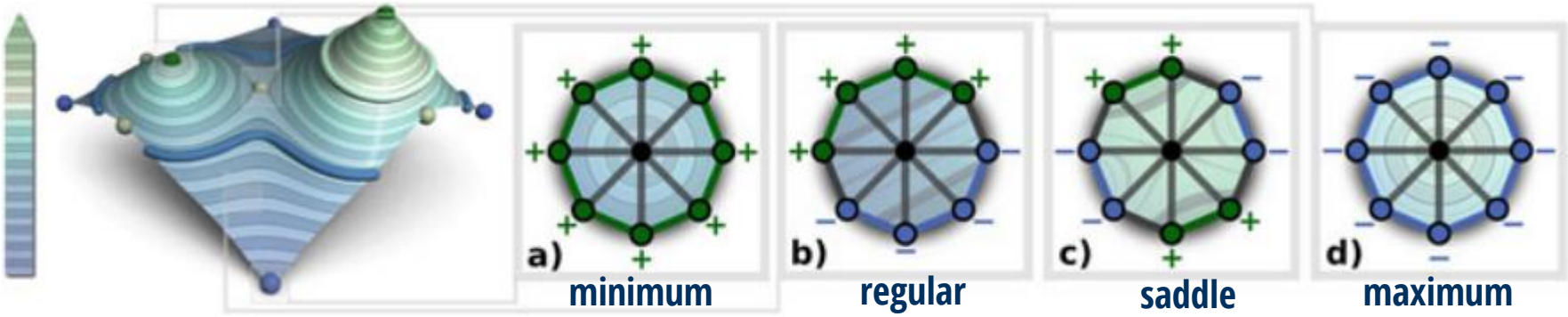
- Therefore any iso-contour restricted to a d-simplex is planar
- Furthermore, an integral line restricted to a d-simplex is straight



straight integral line

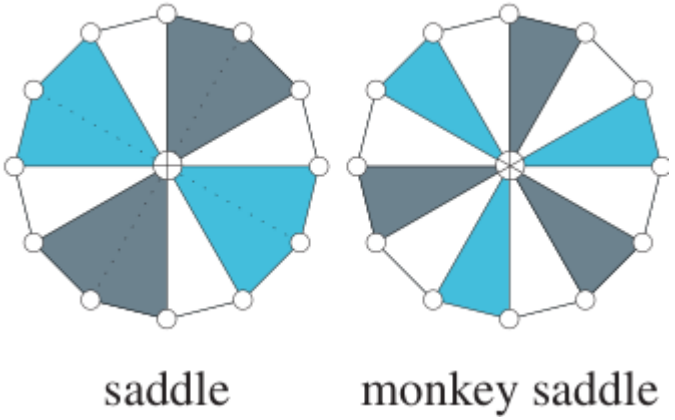
Critical Points

- lower / upper link $\mathrm{Lk}^-(\sigma)$ / $\mathrm{Lk}^+(\sigma)$ of d-simplex relative to PL function f is subset of $\mathrm{Lk}(\sigma)$ such that all 0-faces have strictly smaller / larger f-value than those of σ.
- A vertex v is regular with respect to f if $\mathrm{Lk}^-(v)$ & $\mathrm{Lk}^+(v)$ are non-empty and simply connected, else v is critical.
- Critical points only occur at vertices and their number is therefore finite
- If $\mathrm{Lk}^-(v)$ / $\mathrm{Lk}^+(v)$ is empty, v is a minimum / maximum. Otherwise it is called a saddle



Degenerate Critical Points

- The multiplicity μ of a saddle is $\mu = \max\{\beta_0(\mathrm{Lk}^-(\sigma)), \beta_0(\mathrm{Lk}^+(\sigma))\} - 1$
- Degenerate points are all saddles with $\mu > 1$ and a PL Morse scalar field has no degenerate point.



saddle monkey saddle

- Any PL scalar field can be transformed to a PL Morse scalar field by saddle unfolding:

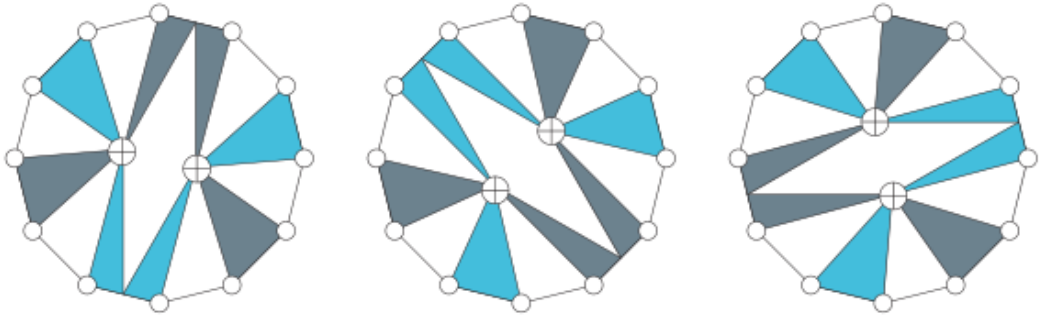


Fig. 3. A monkey saddle may be unfolded into two simple saddles in three different ways. If a wedge consists of a single edge, this edge unfolds into two copies.

Critical Point Indices

- For PL 2-Mannifolds the index of a critical point is
  - minimum … index = 2
  - 1-saddle … index = 1
  - maximum … index = 0
- For PL 3-Mannifolds:
  - minimum … index = 3
  - 2-saddle … index = 2
  - 1-saddle … index = 1
  - maximum … index = 0
- Let C_f^i be the number of critical points of f with index i, then:

$$\chi(\mathbb{T}) = \sum_{i=0}^{d} (-1)^i C_f^i(\mathbb{T})$$

Euler-Morse Relation

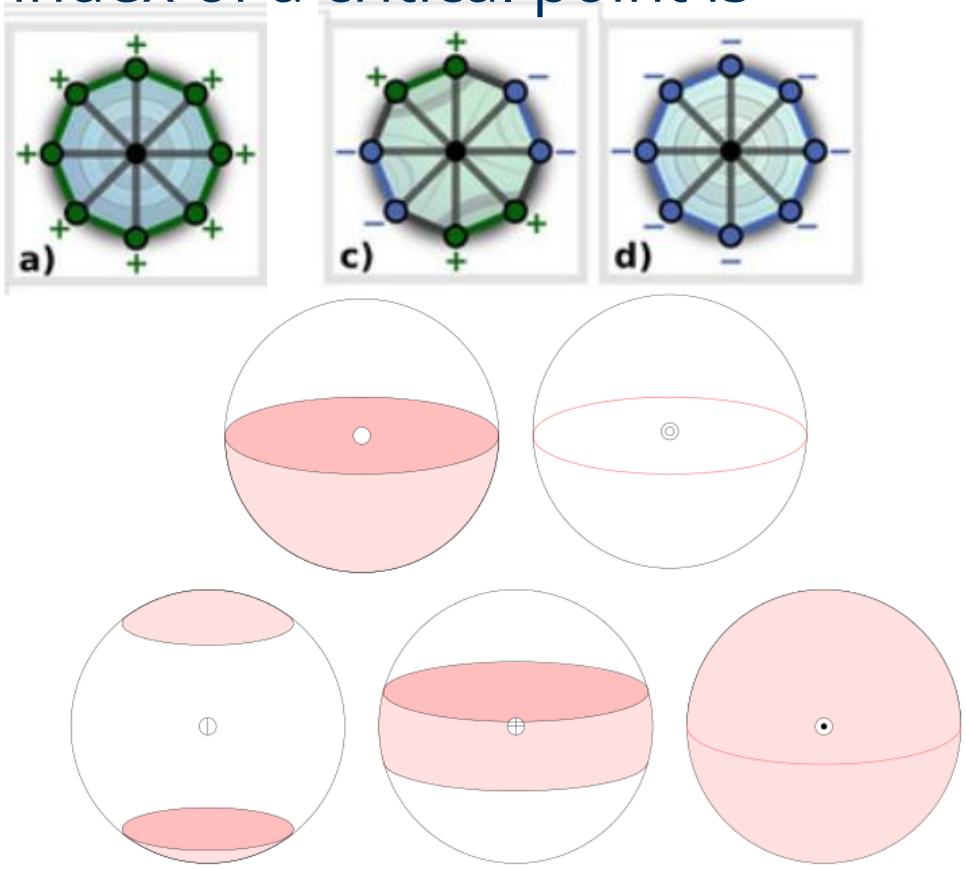


Figure 1: The local pictures with shaded oceans and white continents of a regular point, a minimum, a 1-saddle, a 2-saddle, and a maximum. Take notice of the symbols used to mark the different types of vertices at the centers of the spheres.

PERSISTENCE

Filtration

- Given a PL scalar field f over a simplicial complex $\mathbb{K}$ with $f(\tau) < f(\sigma)$ if τ is a face of σ.
- Let $i \in \mathbb{N}$ be the index of the ith f-value in the sorted list over all n simplices and $\mathcal{L}_f^-(i)$ the sublevels.
- $\mathcal{L}_f^-(0) \subset \mathcal{L}_f^-(1) \subset \cdots \subset \mathcal{L}_f^-(n-1)$ is called filtration.
- A homomorphism is a map between groups that commutes with the group operation.
- The filtration of f induces a sequence of homomorphisms between the homology groups $H_d(.)$ of the d-simplices in $\mathcal{L}_f^-(i)$:

$$H_d\left(\mathcal{L}_f^-(0)\right) \to H_d\left(\mathcal{L}_f^-(1)\right) \to \cdots \to H_d\left(\mathcal{L}_f^-(n-1)\right) = H(\mathbb{K})$$

Persistent Homology Group

- Given two indices $0 \leq i \leq j \leq n-1$ the dth persistence homology group $H_d^{i,j}$ is a factor group that tracks topological features that existed in $\mathcal{L}_f^-(i)$ and still persist in $\mathcal{L}_f^-(j)$.
- The corresponding Betti numbers $\beta_d^{i,j} = \text{rank} H_d^{i,j}$ are called persistent Betti numbers

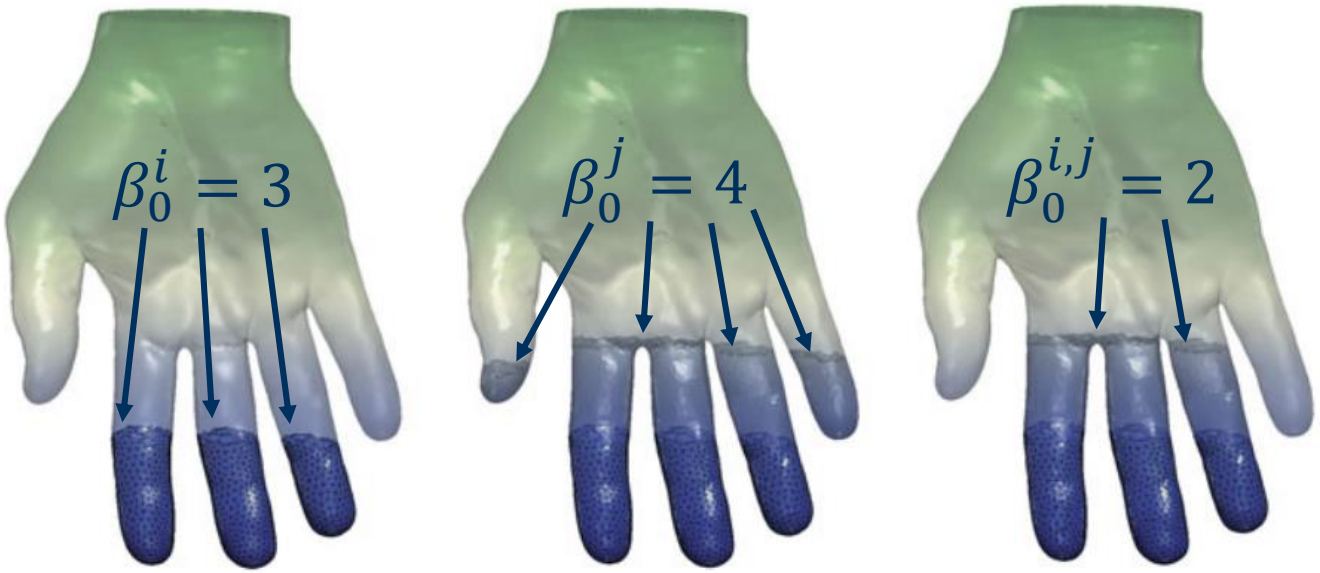


Fig. 2.14 Sub-complexes induced by the filtration of a PL scalar field defined on a PL 3-manifold (dark blue: $\mathscr{L}^-(i)$, light blue: $\mathscr{L}^-(j)$). From left to right: $\beta_0(\mathscr{L}^-(i)) = 3$, $\beta_0(\mathscr{L}^-(j)) = 4$, $\beta_0(\mathscr{L}^-(i,j)) = 2$

Elder's Rule in Persistence Diagram

- During filtration the sublevel Betti numbers change at critical points corresponding to feature death and birth events.
- If two features merge the younger dies and is merged into the older, what is called the Elder's rule.
- The life span of features induces a pairing of the critical points.
- The persistence of a feature is the f-value difference of its critical point pair
- Persistence diagrams show the life span of topological features as vertical bars, typically of connected components induced by β_0 changes.

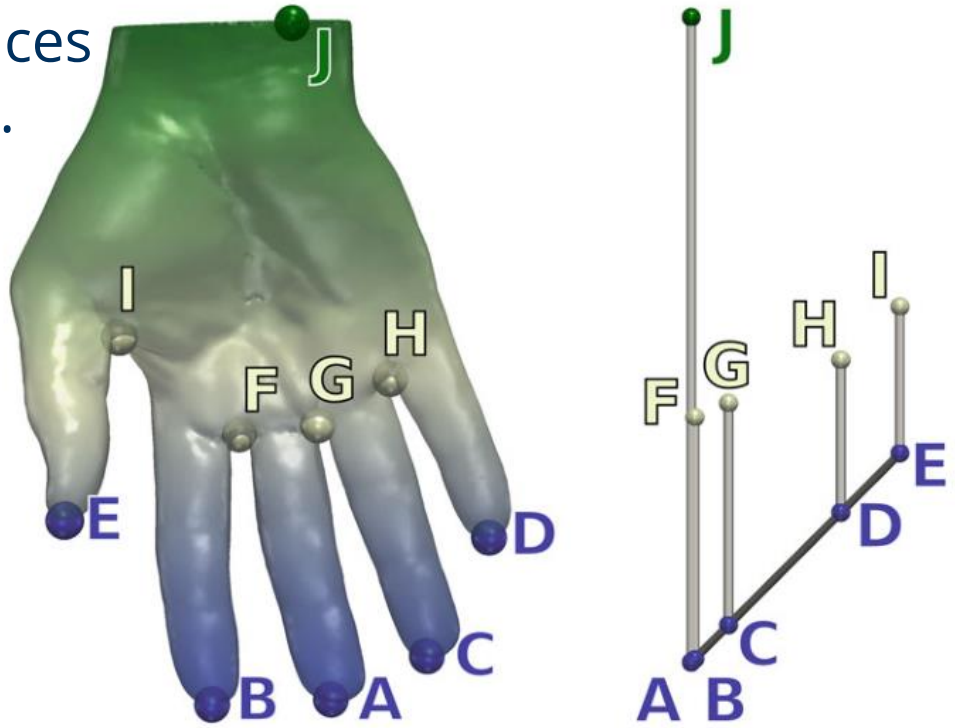


Fig. 2.15 Critical points of a PL scalar field f defined on a PL 3-manifold (left) and its persistence diagram $\mathcal{D}(f)$ (right). In the diagram, each pair of critical points is represented by a vertical bar (white) and its persistence is given by the height of the bar

Persistence Curve

- A persistence curve plots the number of critical point pairs over the logarithm of a threshold on the life span.
- Often the persistence curve shows a flat plateau separating unimportant features with low persistence from important features with high persistence.

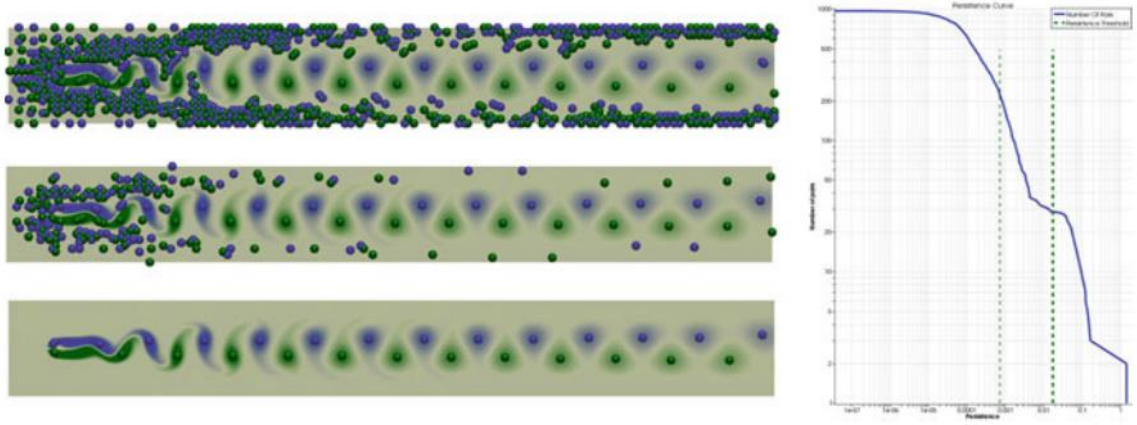


Fig. 2.13 Minima (blue) and maxima (green) of the orthogonal curl component of a flow simulation of the von Kármán street (flow turbulence behind an obstacle, here at the left of the domain). Right: persistence curve of the field. Selecting extrema involved in pairs more persistent than an increasing threshold (vertical lines, right) yields a hierarchy of critical point sets (left). Here, the light green vertical line (right) corresponds to the middle level (left) while the dark green line (right) corresponds to the bottom level (left). In practice, a flat plateau in the persistence curve (right) often indicates a separation between noise and features

REEB GRAPH

Reeb Graph

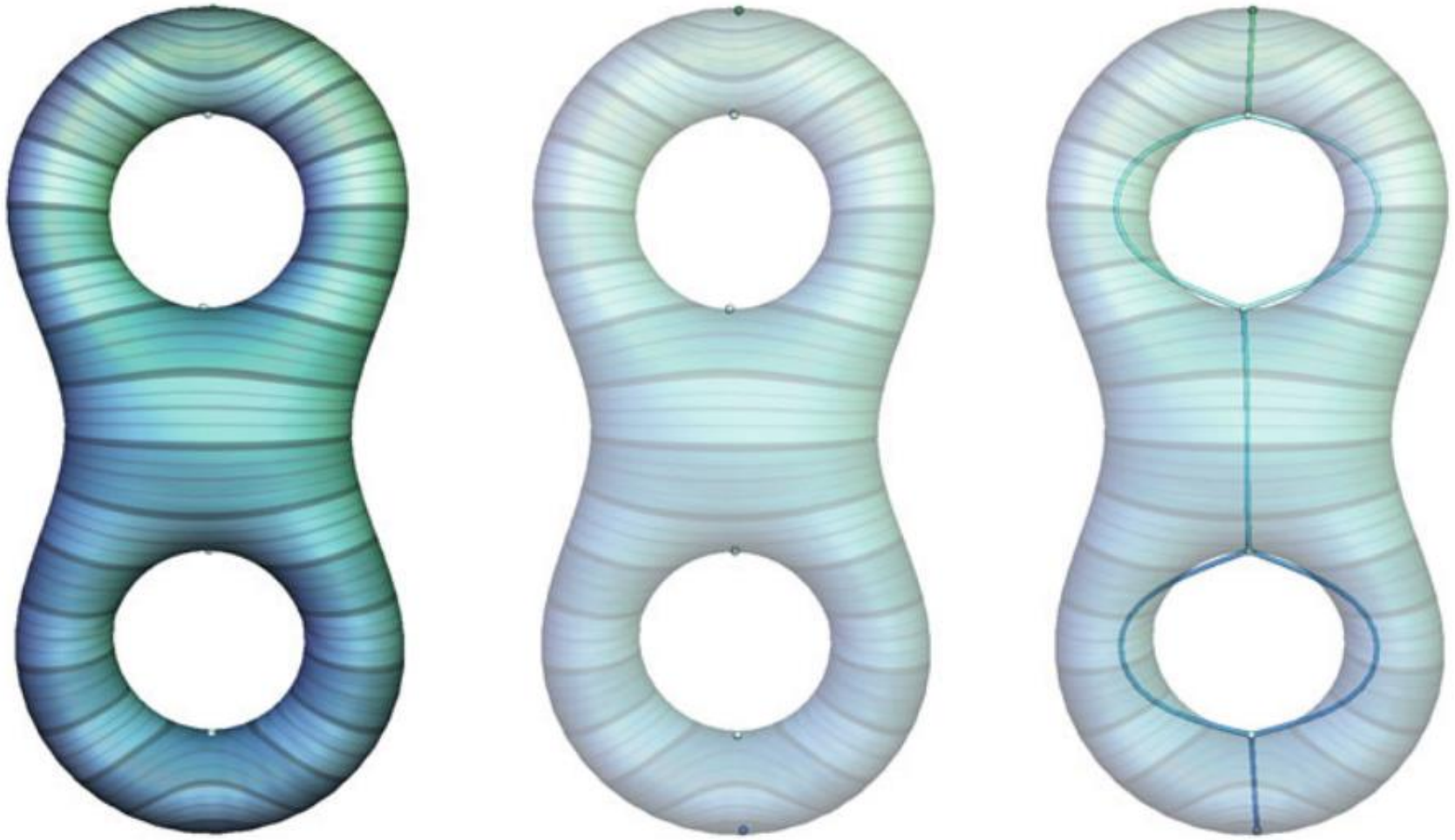


Fig. 2.16 PL Morse scalar field defined on a PL 2-manifold (left and center) and its Reeb graph (right)

Reeb Graph

- Let f be a PL Morse scalar field in PL manifold $\mathbb{M}$.
- Let $f^{-1}\left(f(\underline{\boldsymbol{x}})\right)_{\underline{\boldsymbol{x}}}$ denote the connected component of the contour that contains $\underline{\boldsymbol{x}}$.
- Two points $\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}} \in \mathbb{M}$ are equivalent with respect to f, denoted as $\underline{\boldsymbol{x}} \sim_f \underline{\boldsymbol{y}}$ if they belong to the same connected contour component, i.e. $\underline{\boldsymbol{x}} \sim_f \underline{\boldsymbol{y}} \Leftrightarrow \underline{\boldsymbol{y}} \in f^{-1}\left(f(\underline{\boldsymbol{x}})\right)_{\underline{\boldsymbol{x}}}$.
- The Reeb Graph $\mathcal{R}(f)$ of f is the one-dimensional simplicial complex defined by the quotient space of $\mathbb{M}$ with respect to the equivalence relation $\sim_f$.
- Intuitively, each edge (1-simplex) of $\mathcal{R}(f)$ corresponds to an evolving contour of constant β_0 and each point corresponds to a contour – embeddable e.g. at centroid

Critical Points in Reeb Graph

- f can be decomposed into: $\mathbb{M} \xrightarrow{\phi} \mathcal{R}(f) \xrightarrow{\psi} \mathbb{R}$, where ψ maps each contour to its f-value.
- ϕ maps
  - regular points of f to the interior of a 1-simplex of $\mathcal{R}(f)$
  - extrema of f to 0-simplices of valence 1
- on 2D-manifolds, ϕ maps saddles to 0-simplices of valence 2, 3 or 4.
- on $d \geq 3$D-manifolds, ϕ maps index 1 and index $d-1$ saddles to 0-simplices of valence 2 or 3.
- on $d > 3$D-manifolds, ϕ maps all other saddles to 0-simplices of valence 2.

Loops in Reeb Graph

- The number of loops $l(\mathcal{R}(f))$ in a Reeb Graph $\mathcal{R}(f)$ on manifold $\mathbb{M}$ is bound by $\beta_1(\mathbb{M})$: $l(\mathcal{R}(f)) \leq \beta_1(\mathbb{M})$.
- It is not equal as Reeb Graph construction can remove 1-cycles. But it cannot add new 1-cycles.
- It follows that the Reeb Graph on simply connected manifolds ($\beta_1 = 0$) is loop free, i.e. it is a tree called contour tree.
- On orientable 2D-manifolds without boundary the number of Reeb Graph loops is equal to the genus.

Reeb Graph Applications

- persistence can be applied to Reeb Graph directly and used for simplification / abstraction
- ϕ^{-1} can be stored per Reeb Graph simplex during construction and allows segmentation of the manifold

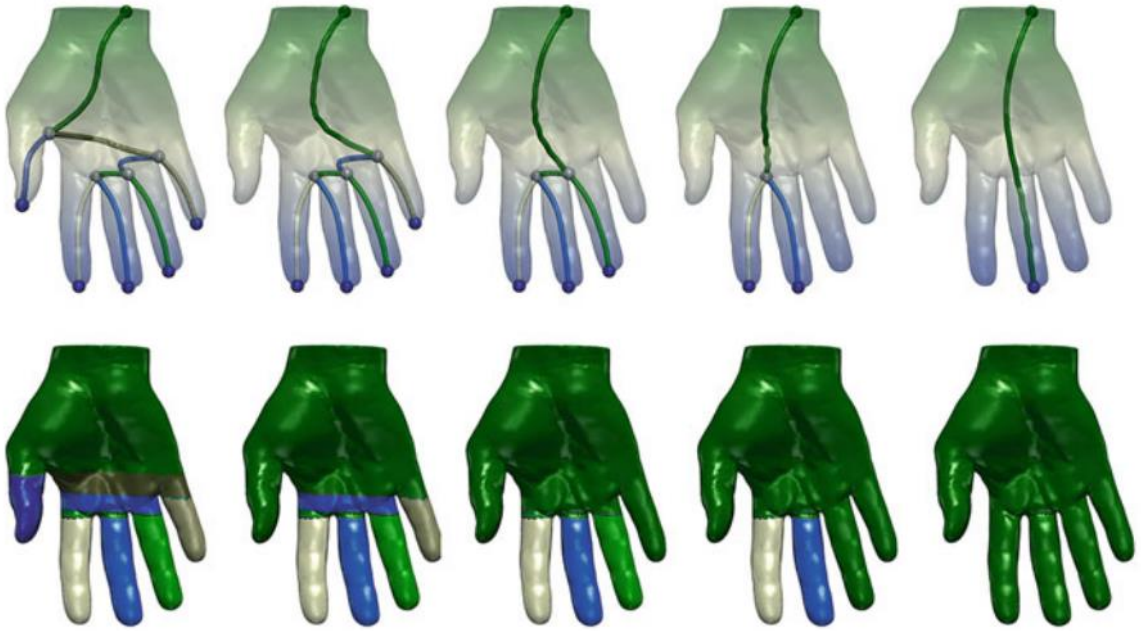


Fig. 2.17 Hierarchy of Reeb graphs obtained by repeated persistence-driven removal of their 1-simplices (top) and hierarchy of data segmentations (bottom) obtained by considering the pre-image by ϕ of each 1-simplex of the Reeb graphs (matching colors)

Algorithms

CONTOUR TREE

Contour Tree Algorithm

- **Hamish Carr, Jack Snoeyink, and Ulrike Axen. "Computing contour trees in all dimensions." Computational Geometry 24.2 (2003): 75-94.**
- Works in arbitrary dimensions
- Domain discretization: simplicial cell complex with barycentric interpolation (2D triangulation, 3D tetrahedralization, …)
- $O\left(n \cdot \log n + \ m \cdot \alpha(m)\right)$ runtime for cell complex with n cells and m vertices ($\alpha(.)$ is inverse Ackermann function)
- No explicit construction of contours such that ϕ^{-1} needs to be constructed additionally in case the goal is segmentation.

Contour Tree Algorithm

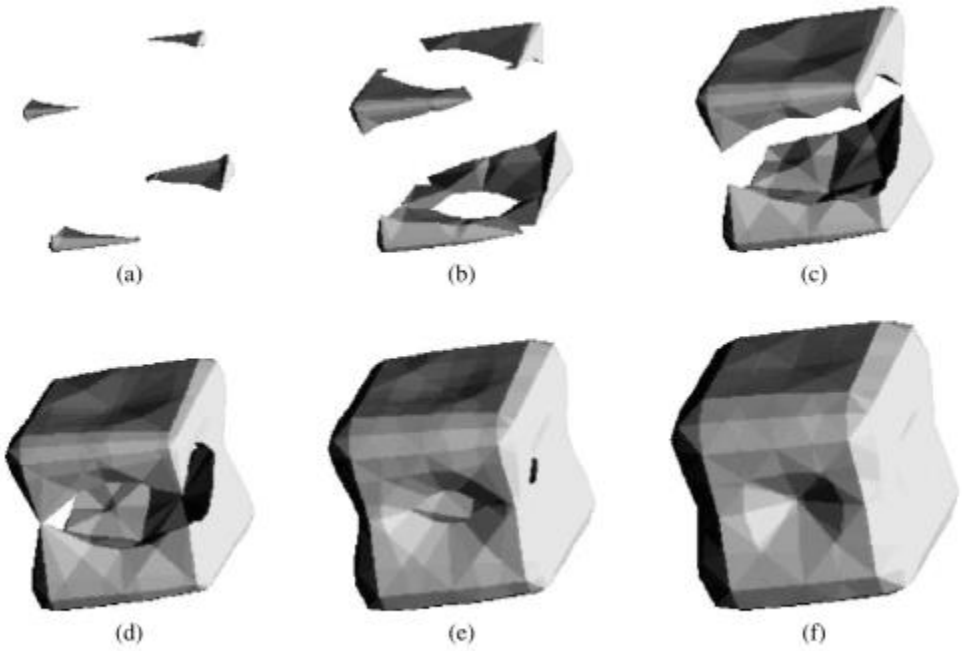


Fig. 1. Level sets of f as $f(x)$ decreases.

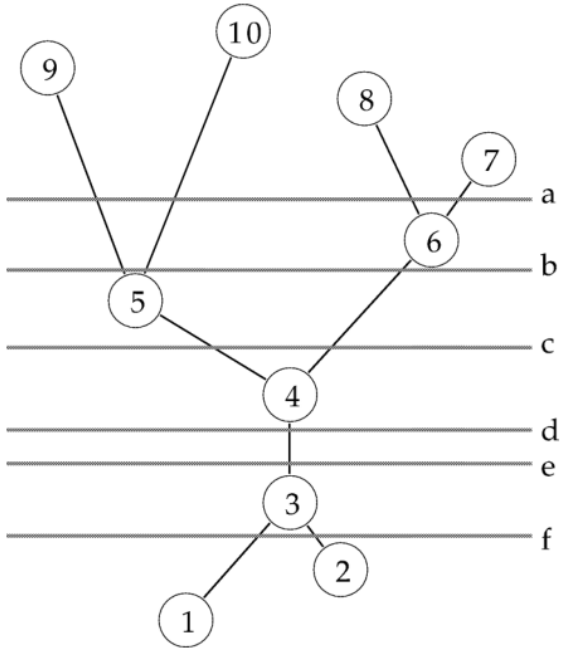
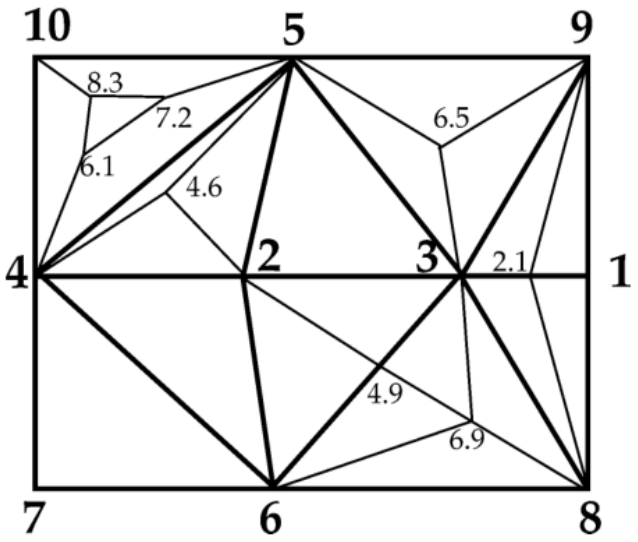
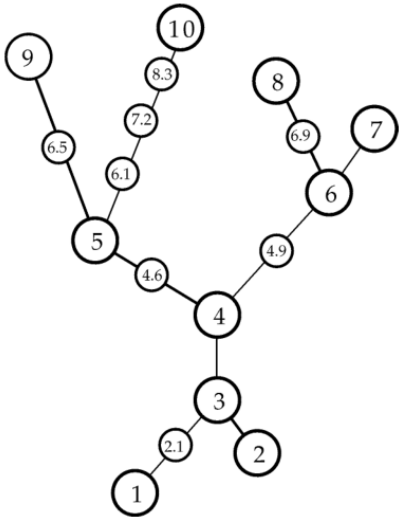


Fig. 2. Contour tree for Fig. 1.

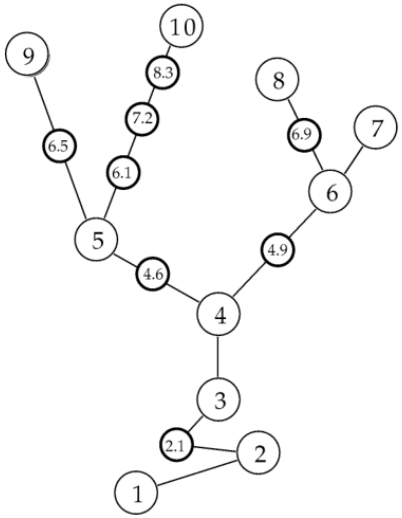
- 3D example of contour tree
- Fig. 1. (a)-(f) filtration with decreasing value v
- Fig. 2. contour tree with lines illustrating snapshots in Fig. 1

Contour Tree Algorithm

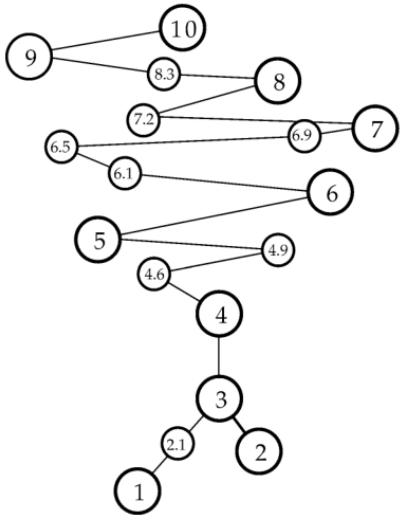




augmented contour



join tree

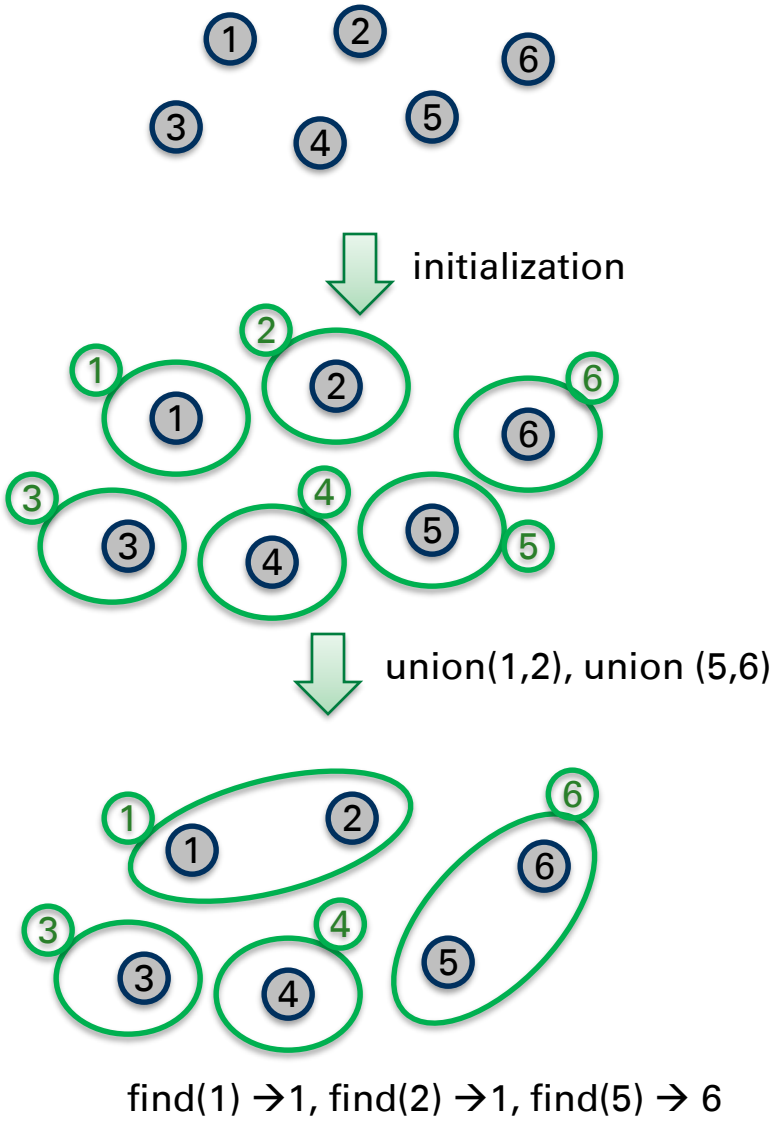


split tree

- Augmented trees contain all nodes of cell complex
- split tree is computed with join tree algorithm on reversed filtration
- Approach
  - compute augmented join and split trees
  - merge trees to form contour tree

Union Find

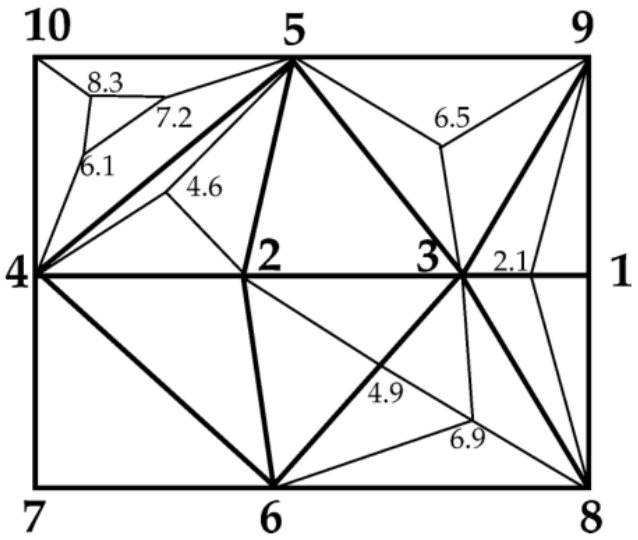
- The union-find data structure manages subsets of elements in an array
- It supports three operations:
  - initialization to one subset per element
  - union of two subsets
  - find subset of element
- All operations can be performed in amortized near constant time
- Careful: subset indices can miss indices in between
- Simple implementation



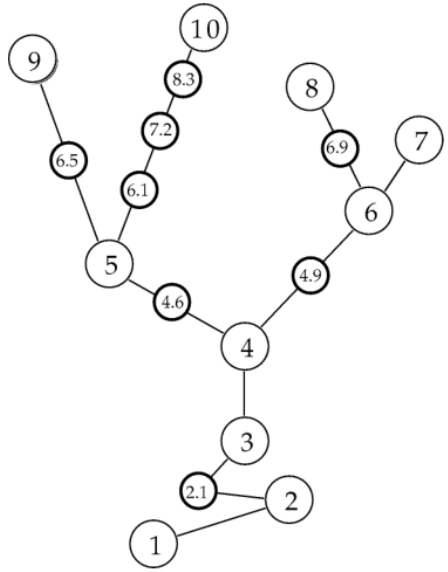
Join Tree Algorithm

Algorithm to compute $J_C = J_M$:
Input: the mesh M, with vertices $x_1 \ldots x_n$ in sorted order (i.e., $h_1 < h_2 \ldots h_n$)
Output: the join tree J_C, with vertices $y_1 \ldots y_n$

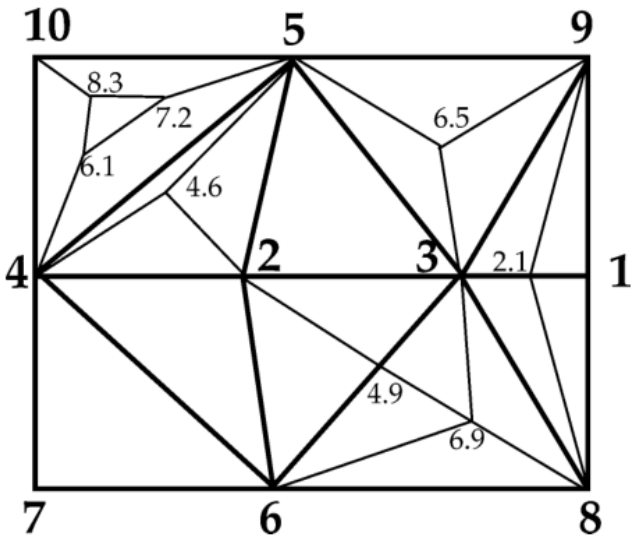
1. for i := n downto 1 do: ← process from highest to lowest value
   (a) Component[i] := i
   (b) LowestVertex[i] := y_i
   (c) for each vertex x_j adjacent to x_i
      i. if ($j < i$) or (Component[i] = Component[j]) skip x_j
      ii. UFMerge(Component [i], Component[j])
      iii. AddEdgeToJoinTree(y_i, LowestVertex[Component[j]])
      iv. LowestVertex[Component[j]] := y_i

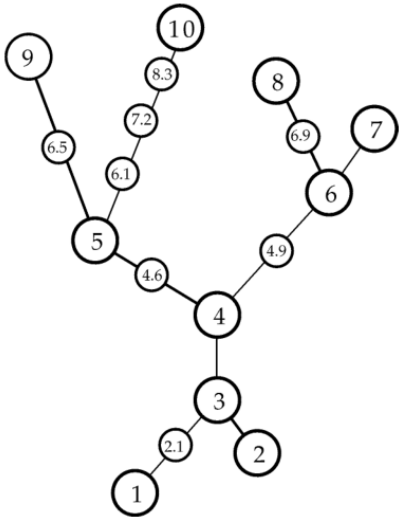


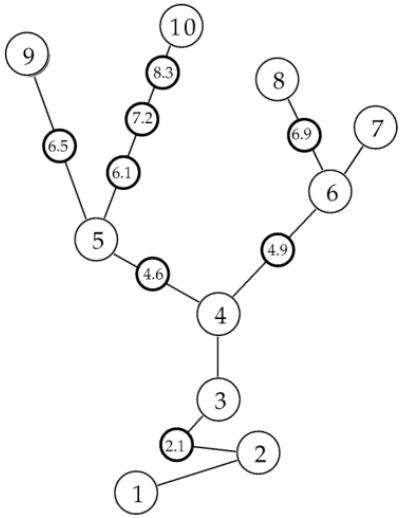
Algorithm 4.1 to construct a join tree.

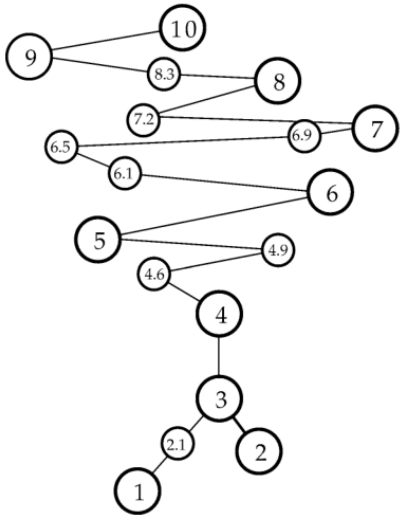


Tree Merging Algorithm









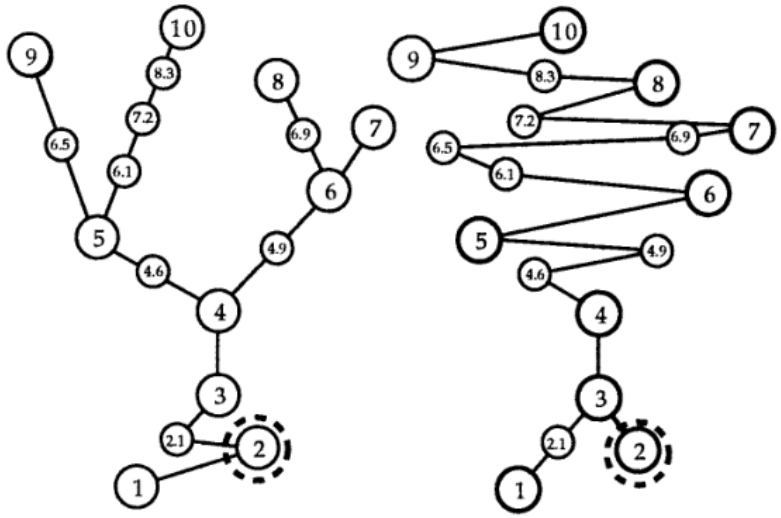
- Observation:
  - updegree (number of upward pointing edges) of contour and join trees conincide
  - downdegree of contour and split trees conincide
- Approach
  - identify upper/lower contour tree leaf and incident edge from up/downdegree of join/split tree to remove these from problem

Tree Merging Algorithm

Algorithm to compute the contour tree:
Input: the join tree J_C and split tree S_C corresponding to C, stored as adjacency lists
Output: the contour tree C

1. For each vertex x_i, if up-degree in J_C + down-degree in S_C is 1, enqueue x_i
2. Initialize C to an empty graph on $\|J_C\|$ vertices
3. While leaf queue size > 1
   (a) Dequeue the first vertex, x_i, on the leaf queue.
   (b) If x_i is an upper leaf, find incident arc $y_i y_j$ in J_C. Else find incident arc $z_i z_j$ in S_C.
   (c) Add $x_i x_j$ to C.
   (d) $J_C \leftarrow J_C \ominus y_i$, $S_C \leftarrow S_C \ominus z_i$.
   (e) If x_j is now a leaf, enqueue x_j.

Fig. 9. Algorithm 4.2 to merge the join and

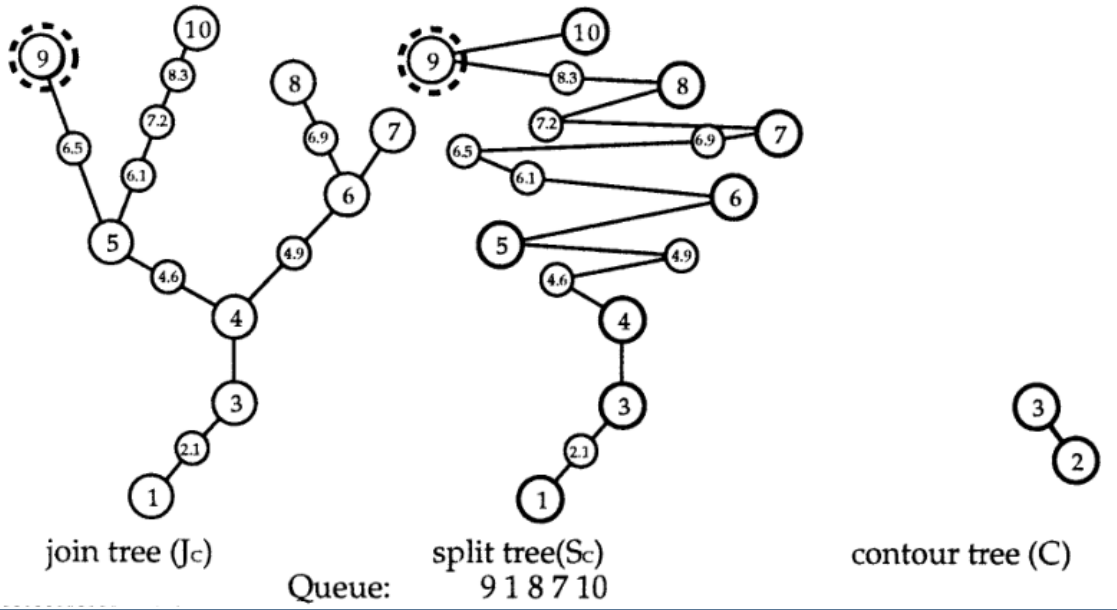


Tree Merging Algorithm

Algorithm to compute the contour tree:
Input: the join tree J_C and split tree S_C corresponding to C, stored as adjacency lists
Output: the contour tree C

1. For each vertex x_i, if up-degree in J_C+ down-degree in S_C is 1, enqueue x_i
2. Initialize C to an empty graph on $\|J_C\|$ vertices
3. While leaf queue size > 1
   (a) Dequeue the first vertex, x_i, on the leaf queue.
   (b) If x_i is an upper leaf, find incident arc $y_i y_j$ in J_C. Else find incident arc $z_i z_j$ in S_C.
   (c) Add $x_i x_j$ to C.
   (d) $J_C \leftarrow J_C \ominus y_i$, $S_C \leftarrow S_C \ominus z_i$.
   (e) If x_j is now a leaf, enqueue x_j.

Fig. 9. Algorithm 4.2 to merge the join and

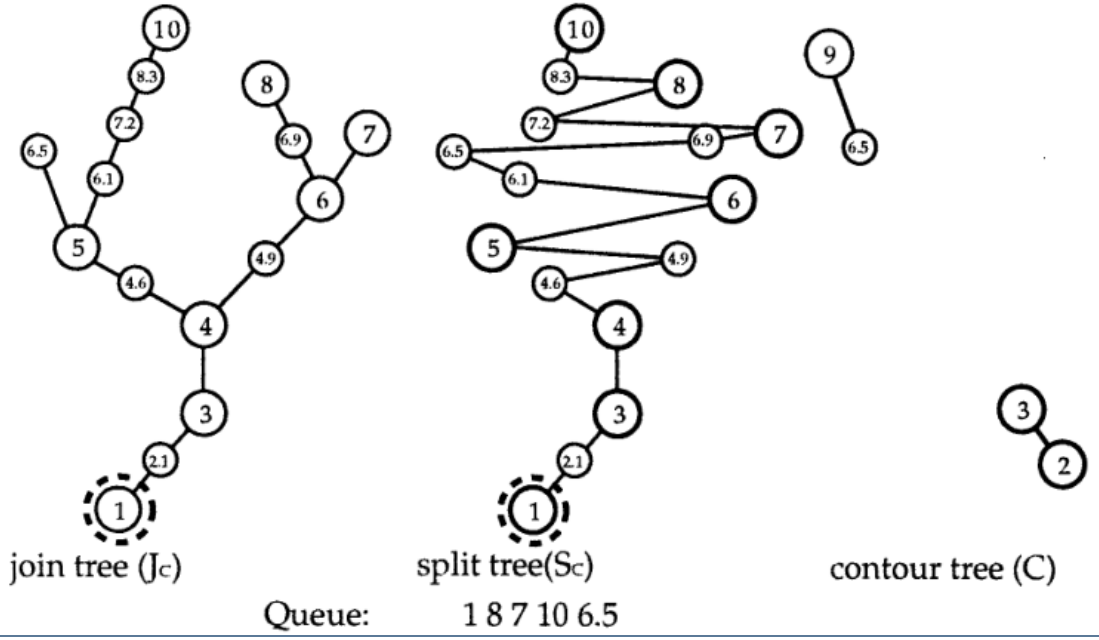


Tree Merging Algorithm

Algorithm to compute the contour tree:
Input: the join tree J_C and split tree S_C corresponding to C, stored as adjacency lists
Output: the contour tree C

1. For each vertex x_i, if up-degree in J_C + down-degree in S_C is 1, enqueue x_i
2. Initialize C to an empty graph on $\|J_C\|$ vertices
3. While leaf queue size > 1
   (a) Dequeue the first vertex, x_i, on the leaf queue.
   (b) If x_i is an upper leaf, find incident arc $y_i y_j$ in J_C. Else find incident arc $z_i z_j$ in S_C.
   (c) Add $x_i x_j$ to C.
   (d) $J_C \leftarrow J_C \ominus y_i$, $S_C \leftarrow S_C \ominus z_i$.
   (e) If x_j is now a leaf, enqueue x_j.

Fig. 9. Algorithm 4.2 to merge the join and

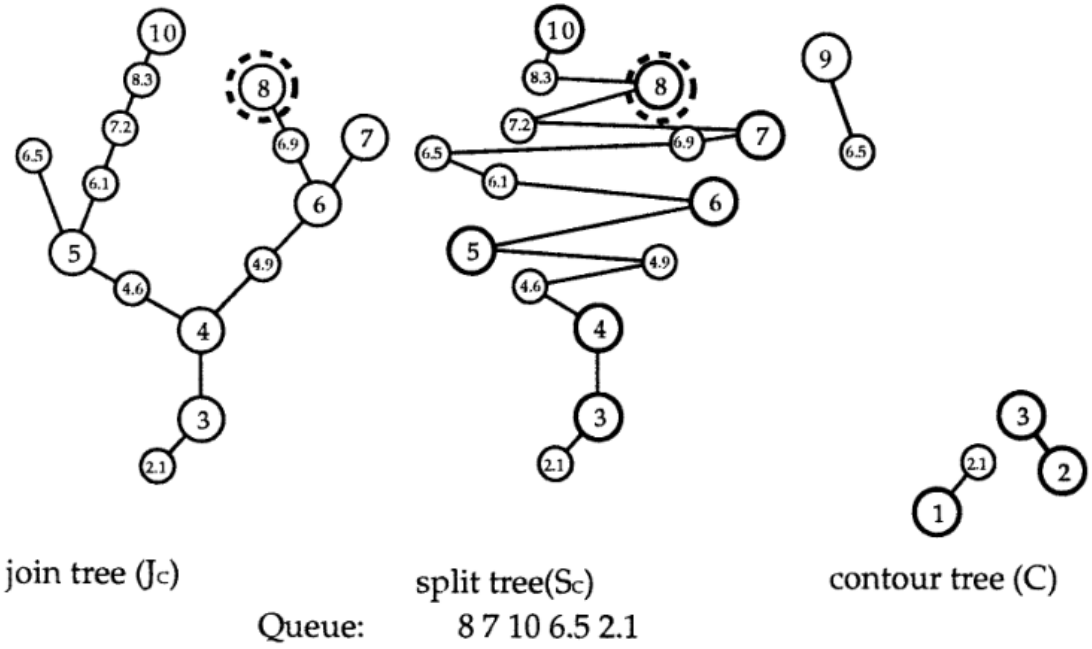


Tree Merging Algorithm

Algorithm to compute the contour tree:
Input: the join tree J_C and split tree S_C corresponding to C, stored as adjacency lists
Output: the contour tree C

1. For each vertex x_i, if up-degree in J_C + down-degree in S_C is 1, enqueue x_i
2. Initialize C to an empty graph on $\|J_C\|$ vertices
3. While leaf queue size > 1
   (a) Dequeue the first vertex, x_i, on the leaf queue.
   (b) If x_i is an upper leaf, find incident arc $y_i y_j$ in J_C. Else find incident arc $z_i z_j$ in S_C.
   (c) Add $x_i x_j$ to C.
   (d) $J_C \leftarrow J_C \ominus y_i$, $S_C \leftarrow S_C \ominus z_i$.
   (e) If x_j is now a leaf, enqueue x_j.

Fig. 9. Algorithm 4.2 to merge the join and

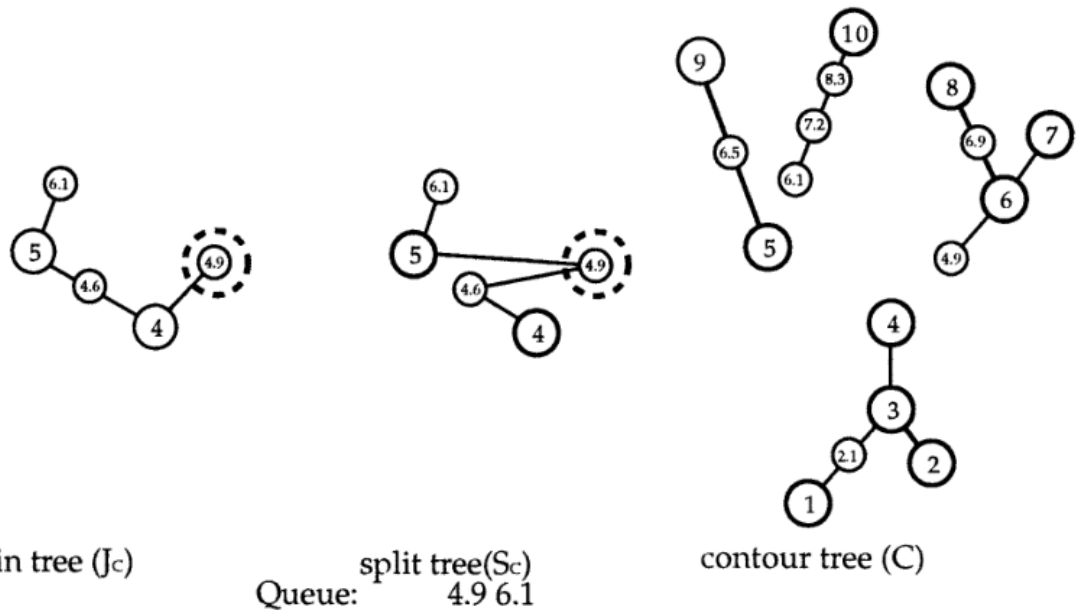


Tree Merging Algorithm

Algorithm to compute the contour tree:
Input: the join tree J_C and split tree S_C corresponding to C, stored as adjacency lists
Output: the contour tree C

1. For each vertex x_i, if up-degree in J_C + down-degree in S_C is 1, enqueue x_i
2. Initialize C to an empty graph on $\|J_C\|$ vertices
3. While leaf queue size > 1
   (a) Dequeue the first vertex, x_i, on the leaf queue.
   (b) If x_i is an upper leaf, find incident arc $y_i y_j$ in J_C. Else find incident arc $z_i z_j$ in S_C.
   (c) Add $x_i x_j$ to C.
   (d) $J_C \leftarrow J_C \ominus y_i$, $S_C \leftarrow S_C \ominus z_i$.
   (e) If x_j is now a leaf, enqueue x_j.

Fig. 9. Algorithm 4.2 to merge the join and

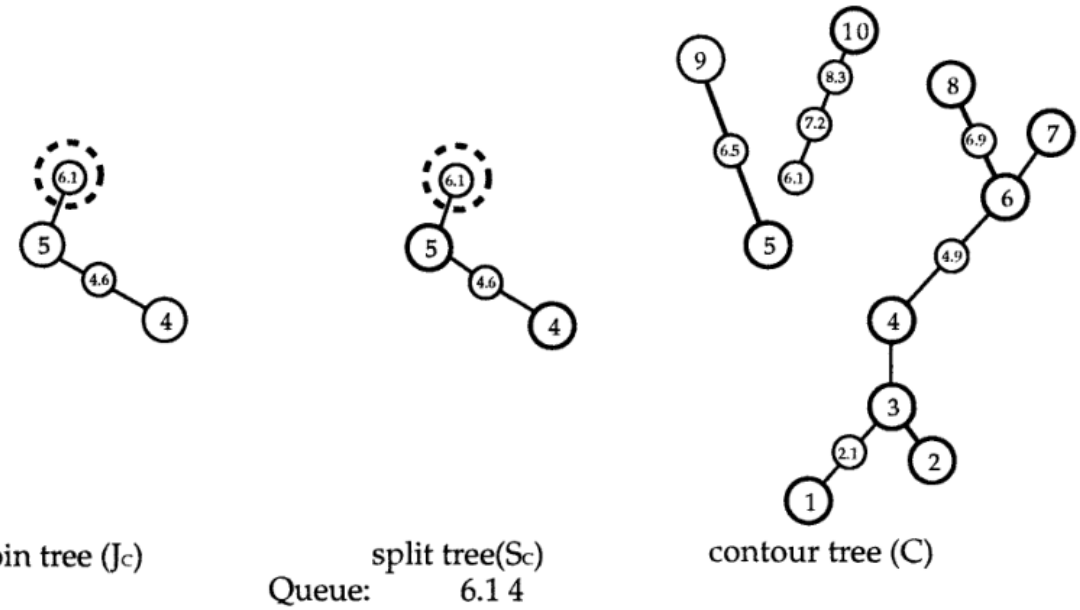


Tree Merging Algorithm

Algorithm to compute the contour tree:
Input: the join tree J_C and split tree S_C corresponding to C, stored as adjacency lists
Output: the contour tree C

1. For each vertex x_i, if up-degree in J_C + down-degree in S_C is 1, enqueue x_i
2. Initialize C to an empty graph on $\|J_C\|$ vertices
3. While leaf queue size > 1
   (a) Dequeue the first vertex, x_i, on the leaf queue.
   (b) If x_i is an upper leaf, find incident arc $y_i y_j$ in J_C. Else find incident arc $z_i z_j$ in S_C.
   (c) Add $x_i x_j$ to C.
   (d) $J_C \leftarrow J_C \ominus y_i$, $S_C \leftarrow S_C \ominus z_i$.
   (e) If x_j is now a leaf, enqueue x_j.

Fig. 9. Algorithm 4.2 to merge the join and

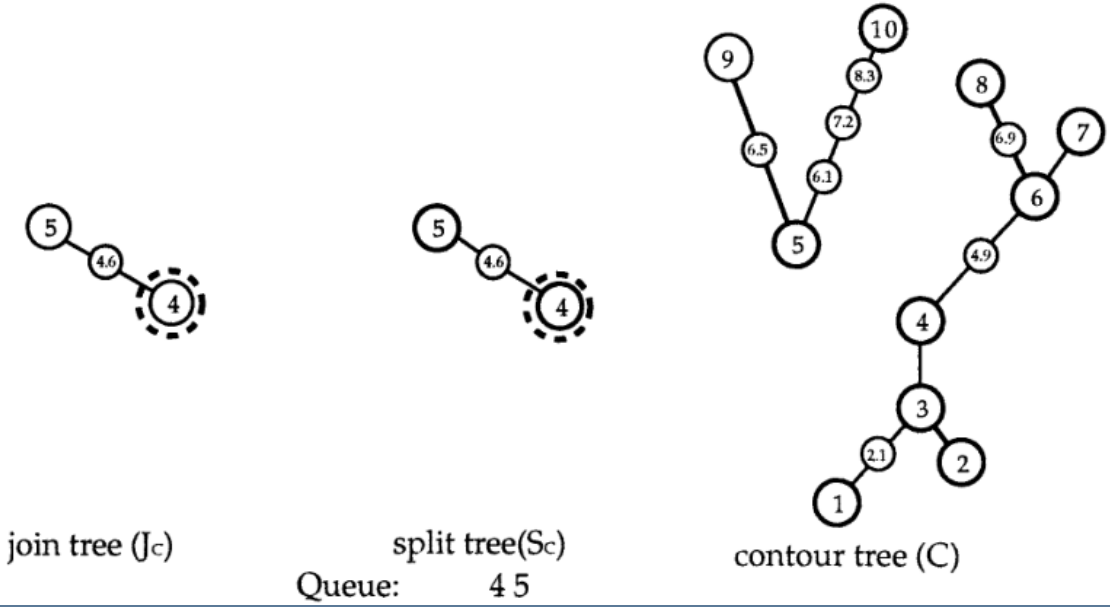


Tree Merging Algorithm

Algorithm to compute the contour tree:
Input: the join tree J_C and split tree S_C corresponding to C, stored as adjacency lists
Output: the contour tree C

1. For each vertex x_i, if up-degree in J_C + down-degree in S_C is 1, enqueue x_i
2. Initialize C to an empty graph on $\|J_C\|$ vertices
3. While leaf queue size > 1
   (a) Dequeue the first vertex, x_i, on the leaf queue.
   (b) If x_i is an upper leaf, find incident arc $y_i y_j$ in J_C. Else find incident arc $z_i z_j$ in S_C.
   (c) Add $x_i x_j$ to C.
   (d) $J_C \leftarrow J_C \ominus y_i$, $S_C \leftarrow S_C \ominus z_i$.
   (e) If x_j is now a leaf, enqueue x_j.

Fig. 9. Algorithm 4.2 to merge the join and

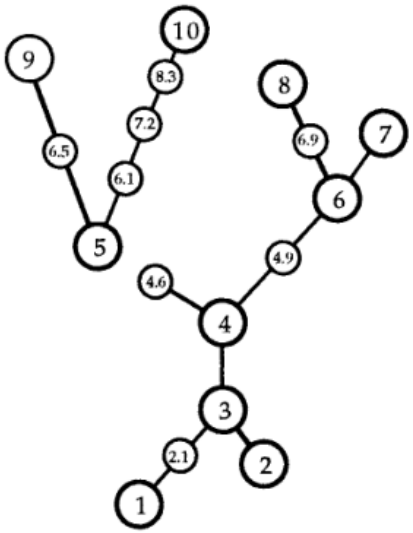


Tree Merging Algorithm

Algorithm to compute the contour tree:
Input: the join tree J_C and split tree S_C corresponding to C, stored as adjacency lists
Output: the contour tree C

1. For each vertex x_i, if up-degree in J_C + down-degree in S_C is 1, enqueue x_i
2. Initialize C to an empty graph on $\|J_C\|$ vertices
3. While leaf queue size > 1
   (a) Dequeue the first vertex, x_i, on the leaf queue.
   (b) If x_i is an upper leaf, find incident arc $y_i y_j$ in J_C. Else find incident arc $z_i z_j$ in S_C.
   (c) Add $x_i x_j$ to C.
   (d) $J_C \leftarrow J_C \ominus y_i$, $S_C \leftarrow S_C \ominus z_i$.
   (e) If x_j is now a leaf, enqueue x_j.

Fig. 9. Algorithm 4.2 to merge the join and



join tree (J_C) split tree(S_C) contour tree (C)

Queue: 5 4.6

TOPOLOGICAL SIMPLIFICATION

Motivation

- In 1D a pair of adjacent extrema can be eliminated by changing the function in a way to move values of extrema closer together till they vanish
- persistence based pairing of extrema yields nested intervals on the domain
- A critical point pair is adjacent after all point pairs contained in their domain interval have been contracted

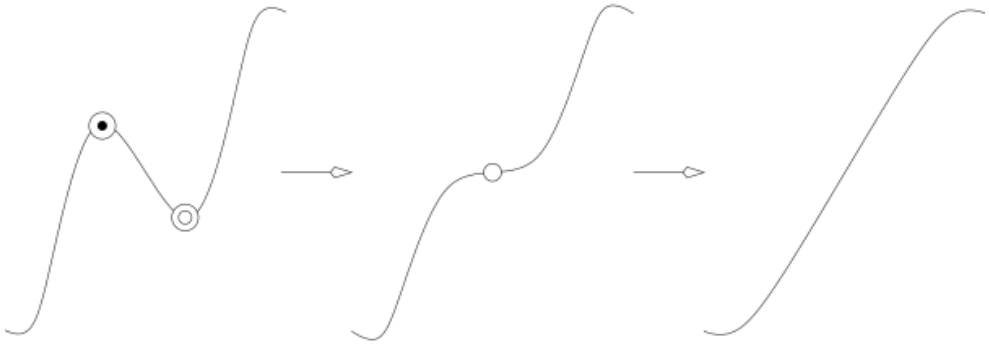


Fig. 15. The cancellation of a minimum–maximum pair.

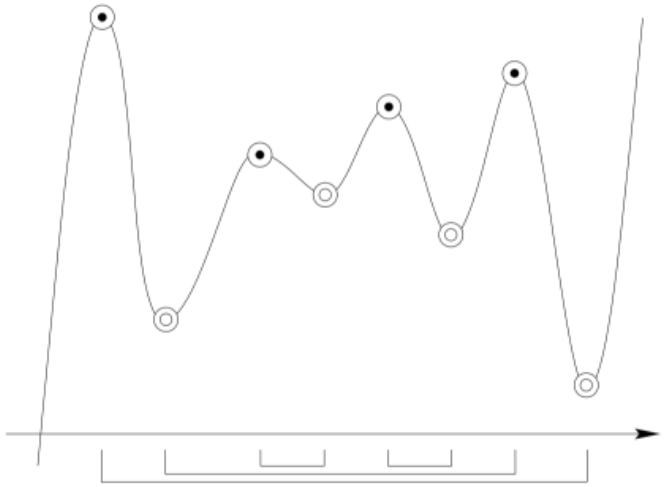


Fig. 16. The intervals defined by critical point pairs are either disjoint or nested.

Hierarchy Construction

- On a MS Complex over a PL 2D manifold persistence pairs minima with merging saddles and maxima with splitting saddles.
- A hierarchy of complexes can be built by removing critical point pairs according to the following contraction operation:

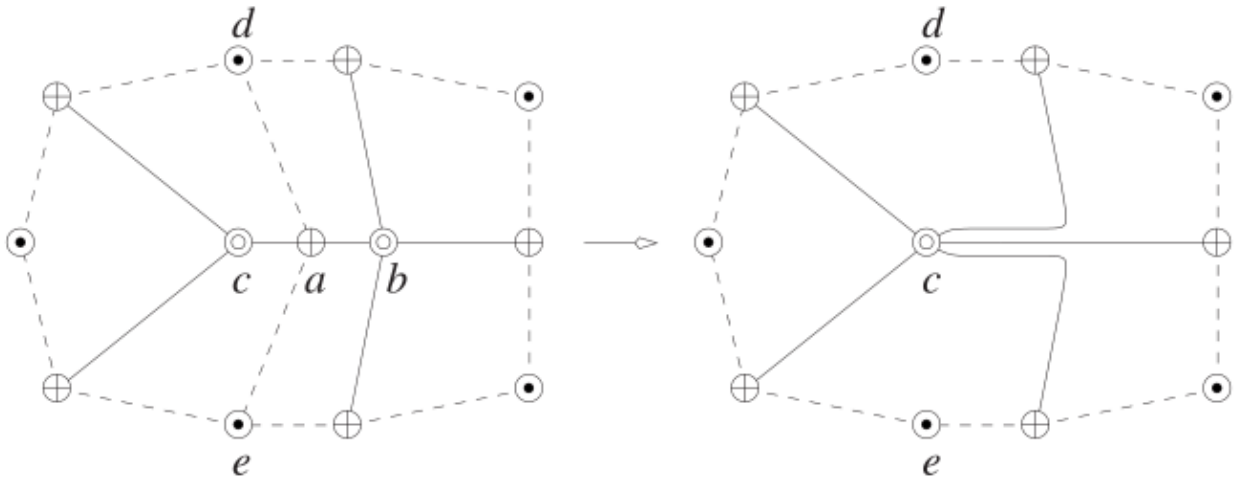


Fig. 17. The cancellation of a and b deletes the arcs ad and ae and contracts the arcs ca and ab. The contraction effectively extends the remaining arcs of b to c.

Simplification Results

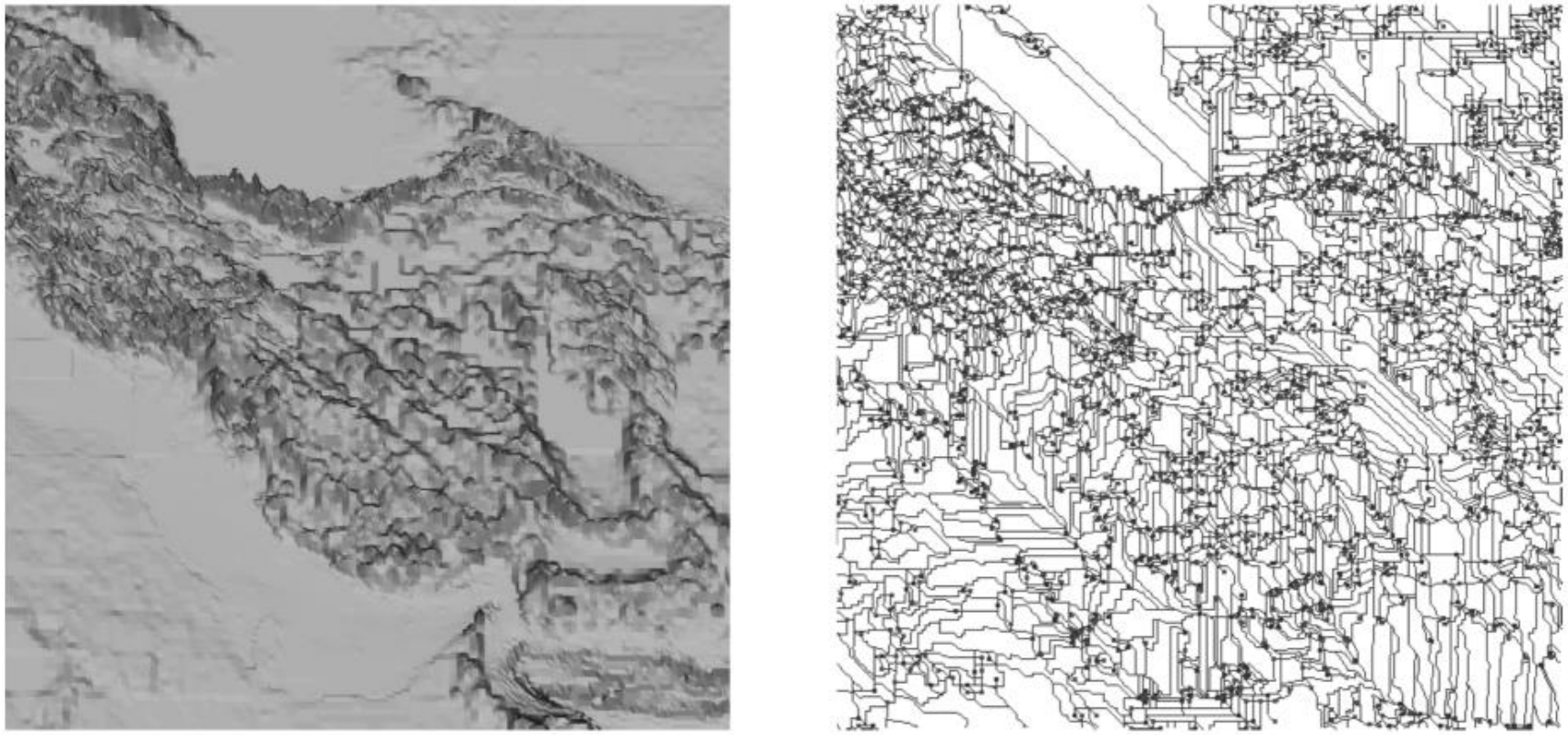


Fig. 19. Iran's Alburz mountain range borders the Caspian sea (top flat area), and its Zagros mountain range shapes the Persian Gulf (left bottom). We show a rendering of the terrain and its quasi MS-complex.

- But no approach for computation of simplified function

Topological Simplification

Tierny, Julien, and Valerio Pascucci. "Generalized topological simplification of scalar fields on surfaces." ***IEEE transactions on visualization and computer graphics*** **18.12 (2012): 2005-2013.**

- Very simple algorithm that takes function f and to be preserved extrema and outputs approximation g with to be preserved extrema minimizing L_∞ norm to f.
- Algorithm directly works on function values and does not need to construct Reeb Graph or MS complex

Algorithm overview

- repeat until convergence
  - enforce minima with algorithm 1 on next slide
  - enforce maxima with algorithm 1 in reversed order
- Runtime is $O(n \cdot \log n)$ due to construction of filtration.

Algorithm 1: Sub-level set constrained reconstruction

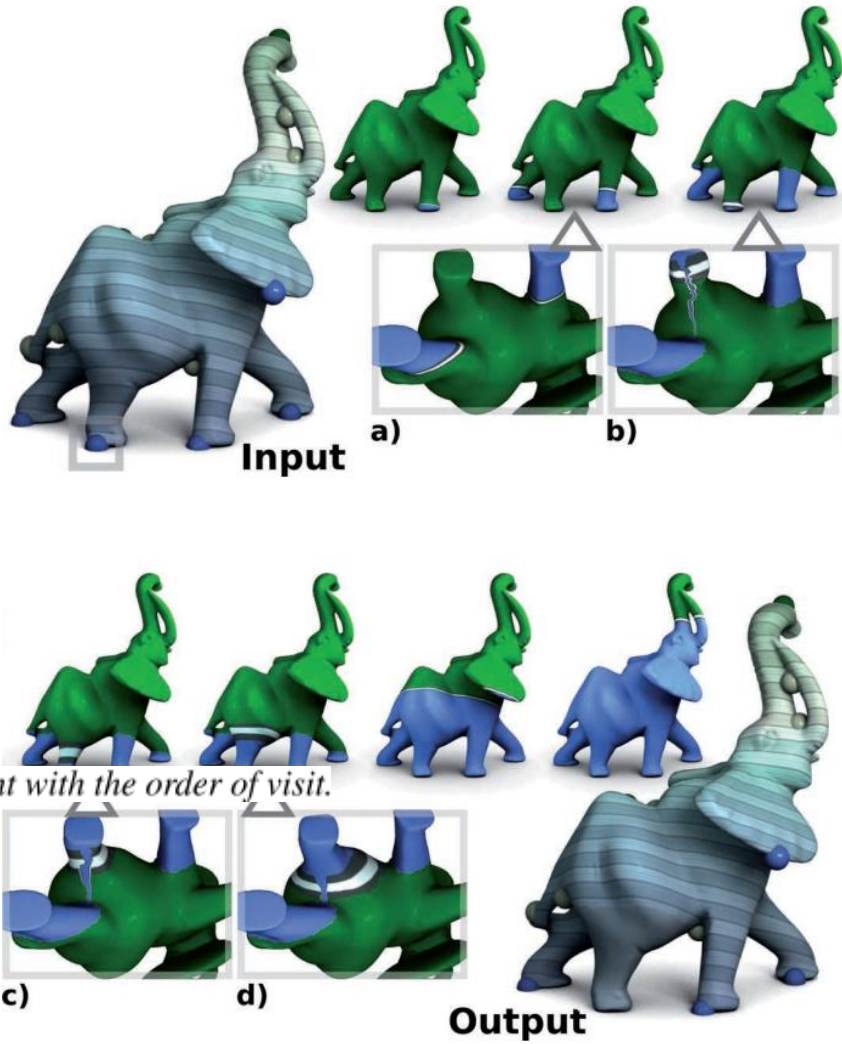
input : Scalar field $f : \mathcal{S} \rightarrow \mathbb{R}$ (with n scalar (f) and offset ($\mathcal{O}$) values);
input : Set of minima constraints to enforce $\mathcal{C}_g^0$;
output: Scalar field $g : \mathcal{S} \rightarrow \mathbb{R}$ with enforced minima in $\mathcal{C}_g^0$.

begin
  // $\mathcal{T}$: set of vertices (self-balancing binary search tree).
  $\mathcal{T} \leftarrow \emptyset$;
  // i: time (integer) when a vertex was last processed.
  $i \leftarrow 0$;

  // Initialize $\mathcal{T}$ with the minima constraints.
  foreach $m \in \mathcal{C}_g^0$ **do** $\mathcal{T} \leftarrow \{\mathcal{T} + m\}$;

  repeat
    $v \leftarrow argmin_{x \in \mathcal{T}} \; f(x)$;
    $\mathcal{T} \leftarrow \{\mathcal{T} - \{v\}\}$;
    mark v as **visited**;
    // Add unvisited neighbors.
    $\mathcal{T} \leftarrow \{\mathcal{T} \cup \{v_n \in Lk(v) \mid v_n$ is not **visited**$\}\}$;
    $\mathcal{A}[i] \leftarrow v$;
    $i \leftarrow i + 1$;
  until $\mathcal{T} = \emptyset$;

  // Scalar and offset value update, for all the vertices.
  // Make the ordering on g (scalar and offset values) consistent with the order of visit.
  for $j \leftarrow 0$ ***to*** n **do**
    if $j \neq 0$ && $f(\mathcal{A}[j]) < g(\mathcal{A}[j-1])$ **then**
      $g(\mathcal{A}[j]) \leftarrow g(\mathcal{A}[j-1])$;
    else
      $g(\mathcal{A}[j]) \leftarrow f(\mathcal{A}[j])$;
    $\mathcal{O}(\mathcal{A}[j]) \leftarrow j$;
end



Topological Simplification

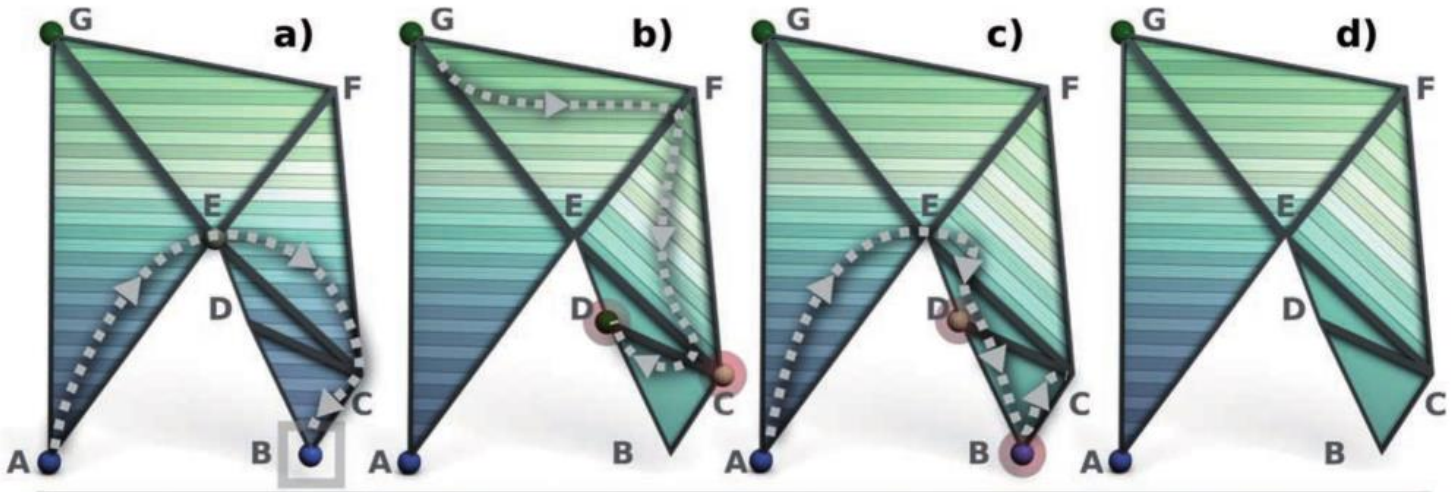


Fig. 6. Sub-level set constrained reconstruction can introduce residual maxima (red spheres): in (a), all the neighbors of D are visited before it, hence yielding a maximum (b). Symmetrically, in (b), all the neighbors of B are visited before it, yielding a minimum (c). Alternating sub- and sur-level set reconstruction reduces the (offset) function difference between the residual extrema and their corresponding saddle (cf. vertex ordering, bottom), and converges to the removal of all the residuals.

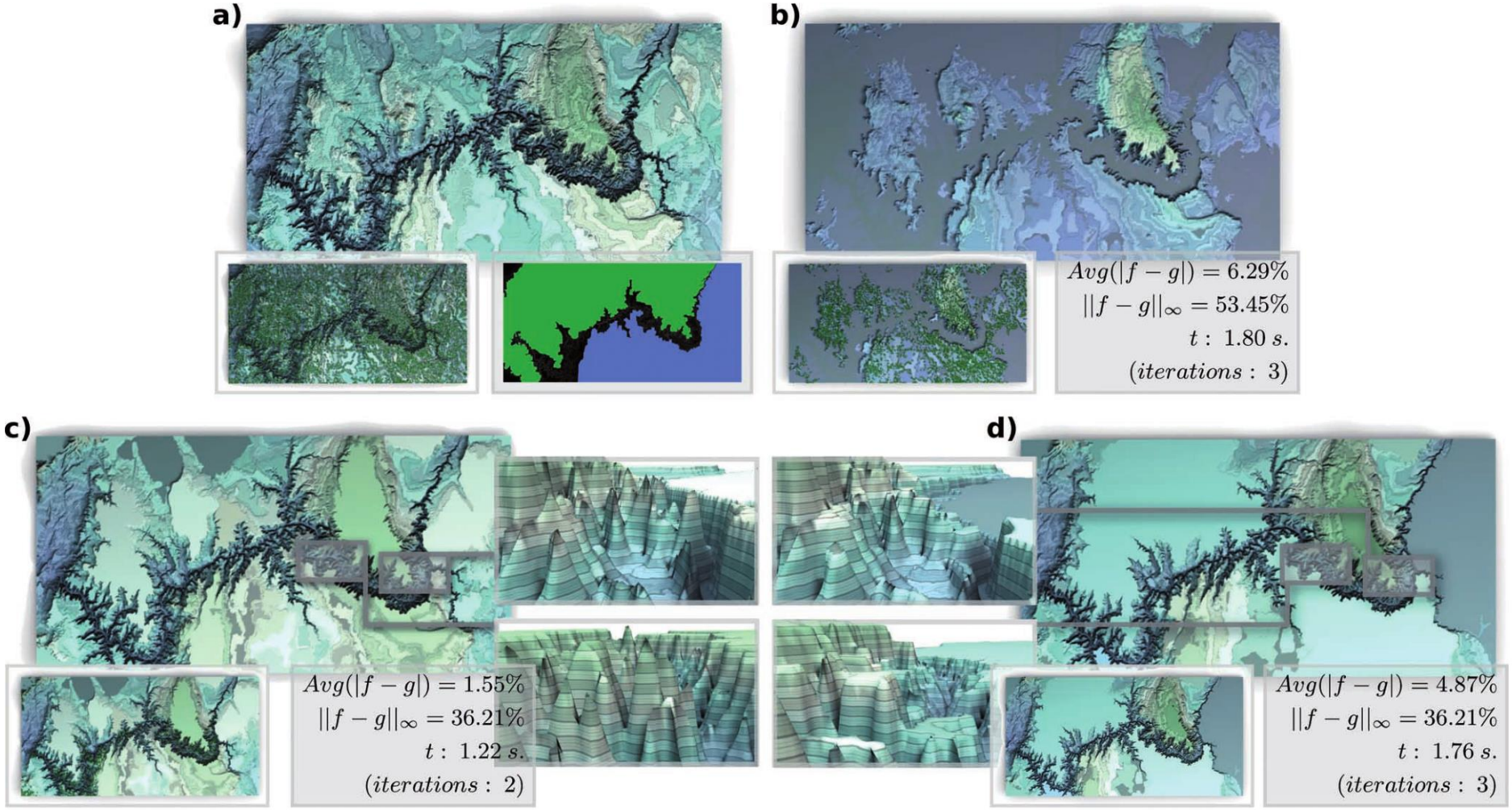


Fig. 10. Simplifying the Grand Canyon ((a), 500k vertices, 65,526 critical points, bottom) with a location driven feature selection. The terrain is decomposed into three geographically meaningful regions: the canyon, its north rim and its south rim (in black, green, blue, (a) bottom). (b) Maintaining only one minimum and removing all the critical points from the canyon (16,756 critical points remain) emphasizes the topological features of the rims and simulates a massive flooding of the canyon. (c) Removing all the critical points from the rims (2,296 remaining) emphasizes the topological features inside the canyon in a way that is suited for an interactive fly-through within the canyon. An ϵ-simplification with compatible L_∞ norm (d) completely discards the features irrespectively of their location (zoom insets) while yielding a worse average data fitting ($Avg(|f-g|)$).

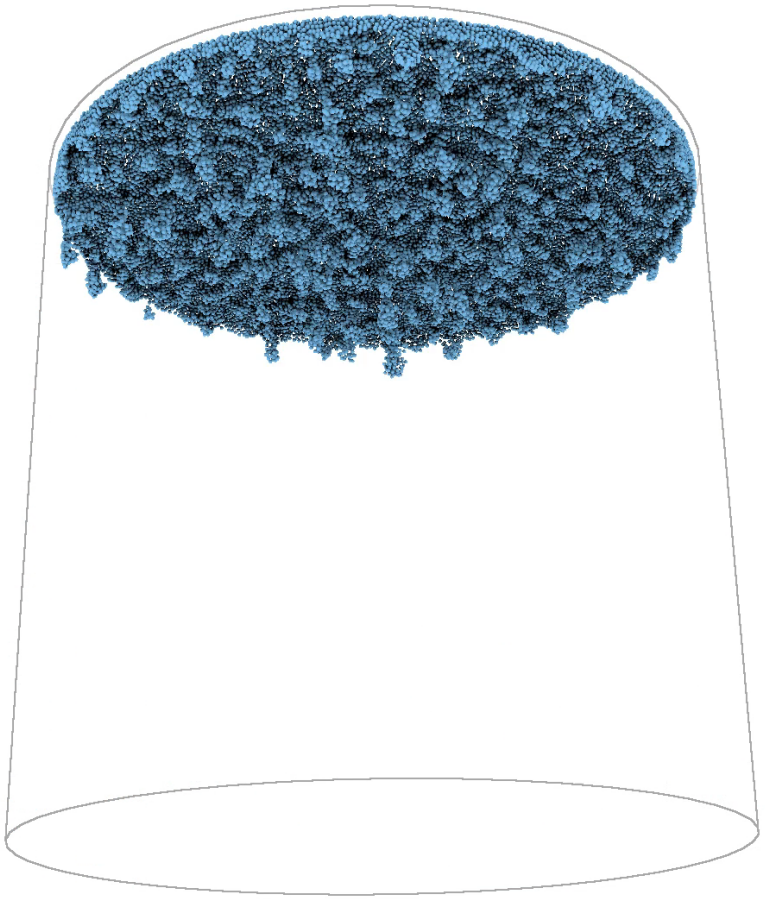
Applications

FINGER COUNTING

Extraction of Viscous Fingers

SciVis Contest 2016

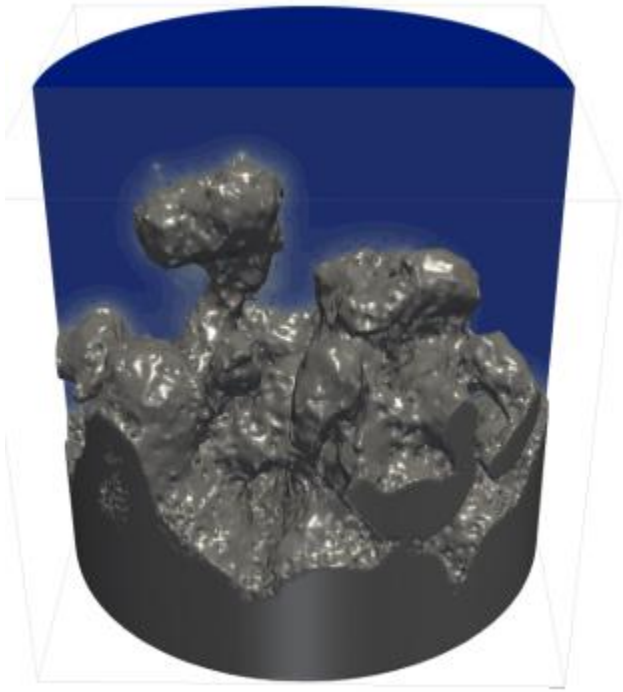
P. Gralka, S. Grottel, J. Staib, K. Schatz, G. Karch, M. Hirschler, M. Krone, G. Reina, S. Gumhold, and T. Ertl, "2016 IEEE Scientific Visualization Contest Winner: Visual and Structural Analysis of Point-based Simulation Ensembles," IEEE Computer Graphics and Applications, vol. 38, no. 3, pp. 106–117, May 2018.



Extraction of Viscous Fingers

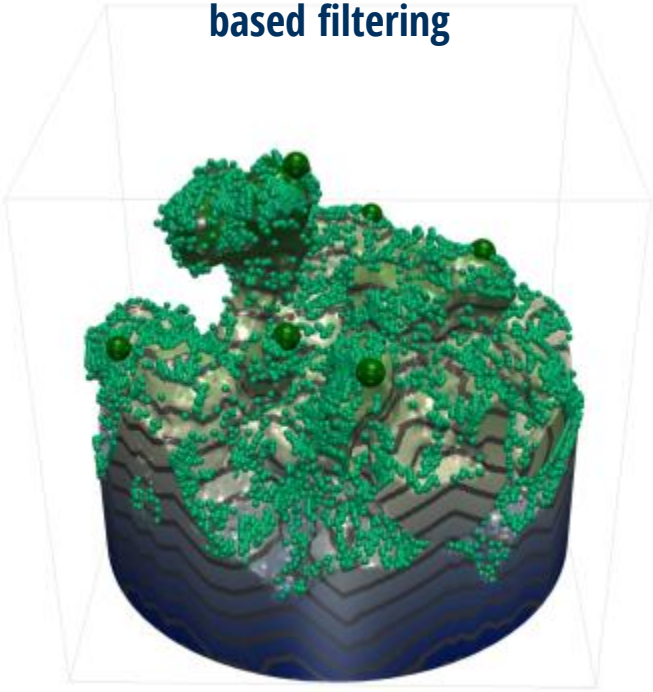
- Favelier, Guillaume, Charles Gueunet, and Julien Tierny. "Visualizing ensembles of viscous fingers." 2016. (video)

largest connected component of super levelset of salt density



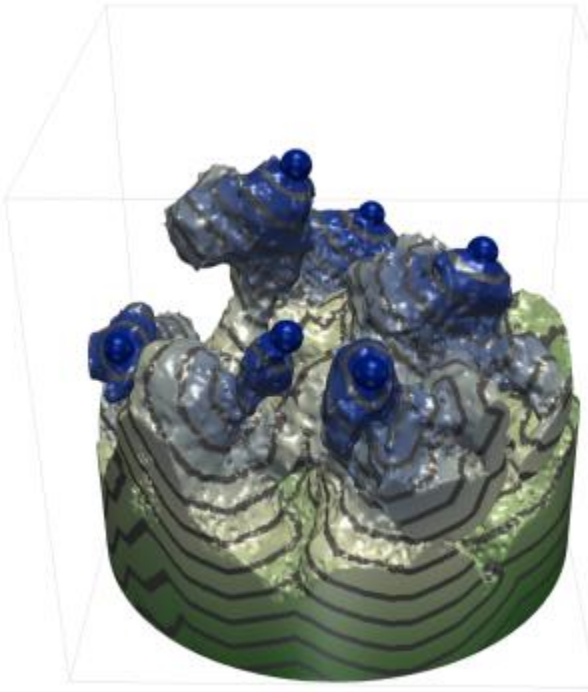
(a)

extraction of maxima of geodesic distance, maxima extraction and persistence based filtering



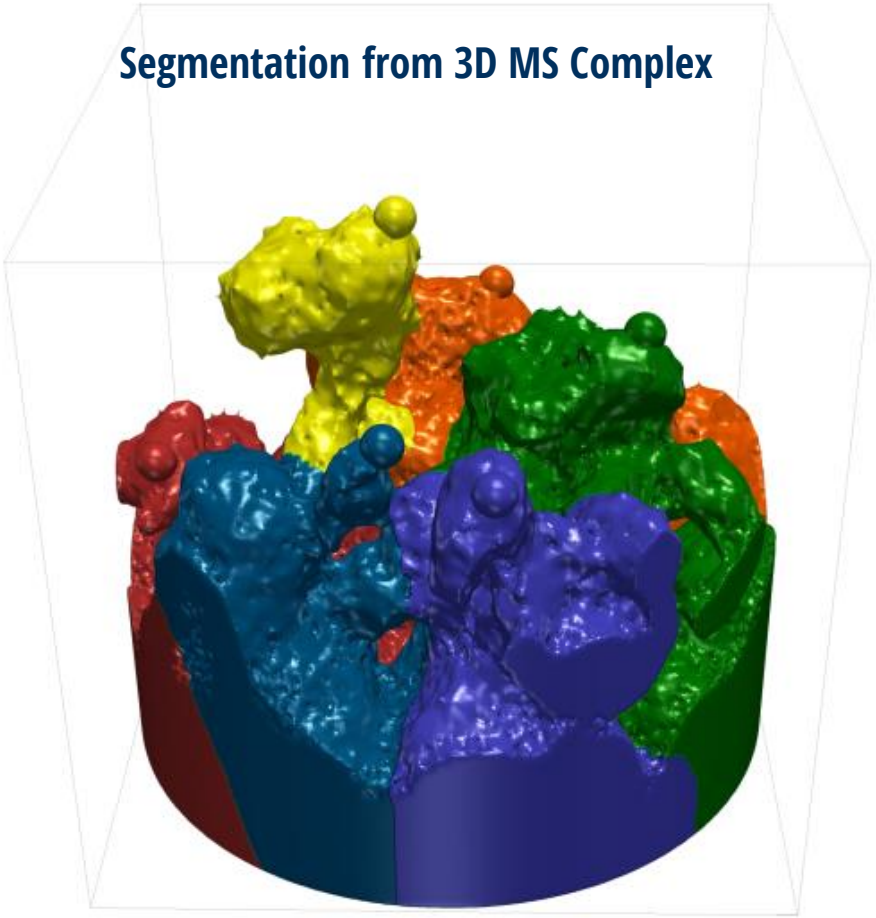
(b)

simplified distance function



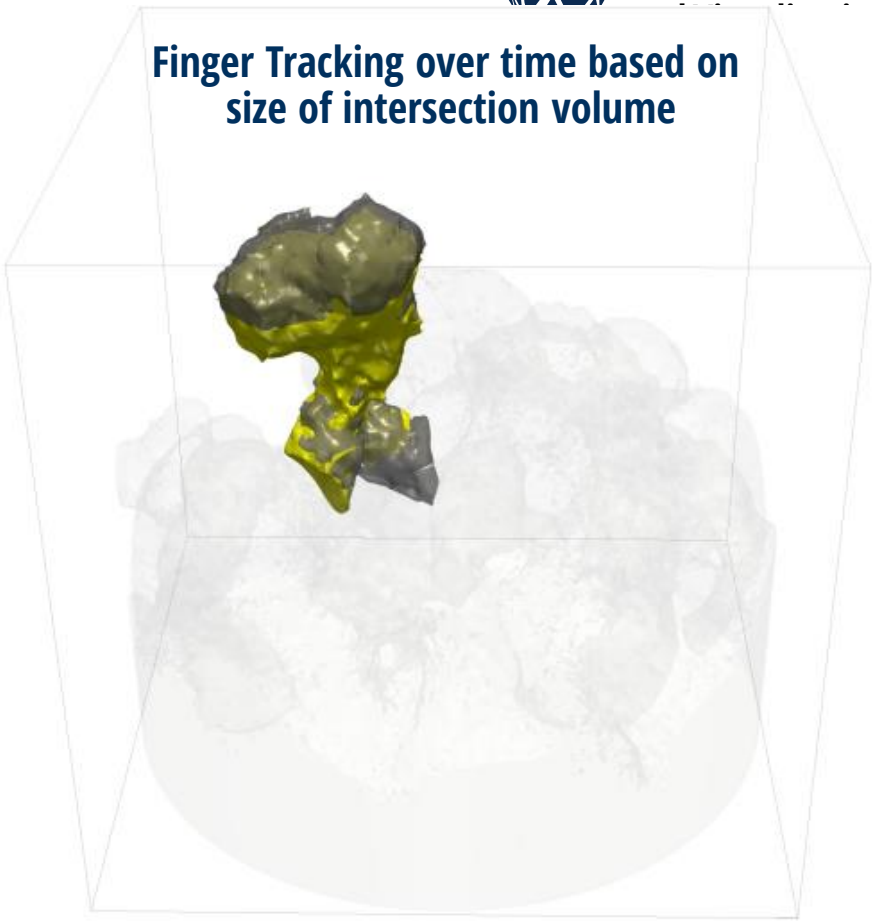
(c)

Segmentation from 3D MS Complex



(d)

Finger Tracking over time based on size of intersection volume



(e)

Figure 3: Overview of our topological data analysis pipeline. (a) The dissolving salt is first isolated from the ambient water by considering the largest connected component (noted $\mathscr{S}$, in gray) of the sur-level set of salt concentration (Sec. 3.1). (b) Finger tips are identified as local maxima (small light green spheres) of the geodesic distance $f_d : \mathscr{S} \to \mathbb{R}$ (color gradient and level lines) from the top of the cylinder. Restricting the identification to the most persistent maxima (larger dark green spheres) enables the identification of the most prominent fingers. (c) Geodesic distance field from the most persistent maxima $f_t : \mathscr{S} \to \mathbb{R}$. (d) The Morse complex of f_t decomposes $\mathscr{S}$ into fingers (Sec. 3.2). (e) Each finger at time step t is connected to the finger at time step $t+1$ which maximizes the volume of their intersection (Sec. 3.3). An example finger is shown in yellow while the corresponding maximizer at time step $t+1$ is shown in transparent black. Data-sets are shown upside down to reduce occlusion.

References

- Edelsbrunner, Herbert, John Harer, and Afra Zomorodian. "Hierarchical Morse-Smale complexes for piecewise linear 2-manifolds." *Discrete and Computational Geometry* 30.1 (2003): 87-107.
- Carr, H., Snoeyink, J., & Axen, U. (2003). Computing contour trees in all dimensions. *Computational Geometry*, *24*(2), 75-94.
- Tierny, Julien. "Introduction to Topological Data Analysis." master course notes, webpage 2016 ff
- Tierny, J. (2017). *Topological Data Analysis for Scientific Visualization*. Springer International Publishing. SLUB
- Tierny, J., Favelier, G., Levine, J. A., Gueunet, C., & Michaux, M. (2017). The Topology ToolKit. *IEEE transactions on visualization and computer graphics*, *24*(1), 832-842.
- Heine, C., Leitte, H., Hlawitschka, M., Iuricich, F., De Floriani, L., Scheuermann, G., Hagen, H. & Garth, C. (2016, June). A survey of topology-based methods in visualization. In *Computer Graphics Forum* (Vol. 35, No. 3, pp. 643-667).

References

- De Floriani, Leila, et al. "Morse complexes for shape segmentation and homological analysis: discrete models and algorithms." Computer Graphics Forum. Vol. 34. No. 2. 2015.
- Tierny, Julien, and Valerio Pascucci. "Generalized topological simplification of scalar fields on surfaces." IEEE transactions on visualization and computer graphics 18.12 (2012): 2005-2013.
- Favelier, Guillaume, Charles Gueunet, and Julien Tierny. "Visualizing ensembles of viscous fingers." 2016.
- P. Gralka, S. Grottel, J. Staib, K. Schatz, G. Karch, M. Hirschler, M. Krone, G. Reina, S. Gumhold, and T. Ertl, “2016 IEEE Scientific Visualization Contest Winner: Visual and Structural Analysis of Point-based Simulation Ensembles,” IEEE Computer Graphics and Applications, vol. 38, no. 3, pp. 106–117, May 2018.
- Edelsbrunner, H., Harer, J., Natarajan, V., & Pascucci, V. (2003). Morse-Smale Complexes for Piecewise Linear 3-Manifolds.
- Laney, Daniel, et al. "Understanding the structure of the turbulent mixing layer in hydrodynamic instabilities." IEEE Transactions on Visualization and Computer Graphics 12.5 (2006): 1053-1060.

Algorithms

MORSE-SMALE COMPLEX

Morse-Smale Complex Algorithm

- Edelsbrunner et. al propose in 2003 an algorithm to construct the Morse-Smale Complex on a PL **2D-manifold** which is composed of critical vertices, edges, and quadrangular faces.
- The runtime is $O(n \cdot \log n)$ to construction a filtration.
- The central difficulty is that integral lines can split and merge in the PL setting.

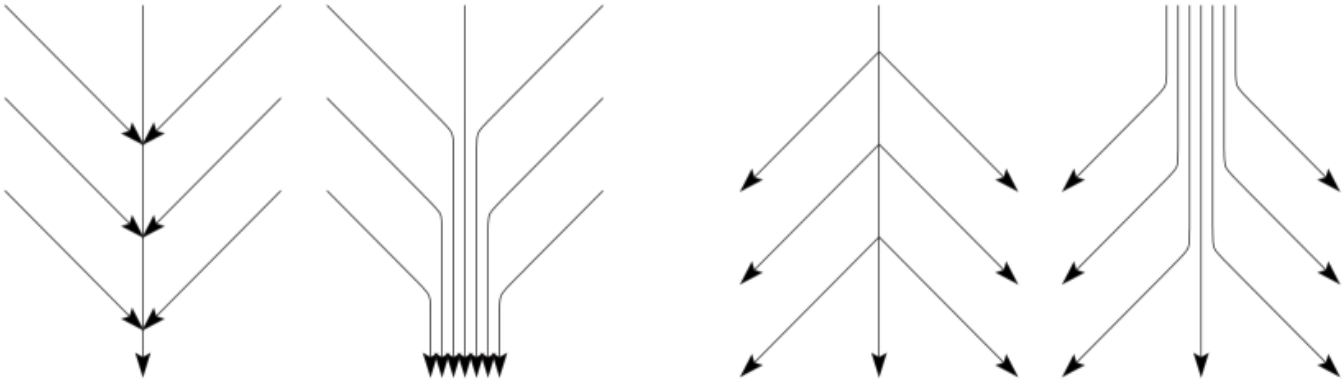


Fig. 4. Merging and forking PL curves and their corresponding smooth flow pictures.

Quasi Morse-Smale Complex

- A Quasi Morse-Smale is a complex with vertices at critical points, quadrangular faces and all paths strictly ascending or descending
- The Morse-Smale complex is the unique Quasi Morse-Smale complex where all paths ascend/descend with maximal slope.

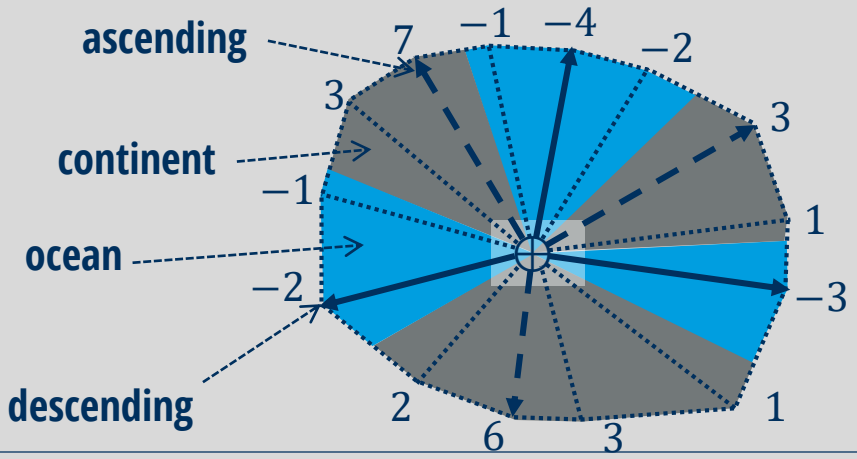
Algorithm Overview

- Construct a Quasi Morse-Smale Complex in 3 Stages
- Perform local transformations with re-rooting of paths until Morse-Smale Complex has been found

Construction of Quasi MS Complex

Stage 1 (complex with junctions)

- find and classify all critical points
- trace k+1 ascending and k+1 descending paths from extreme valued edges in each wedge of each k-fold saddle only along mesh edges, until
  a) extremum (regular case)
  b) hitting a previously traced path (create [multi-]junction)
  c) another saddle (transversal connection)



Construction of Quasi MS Complex

Stage 2 (path extension)

1. junction removal (regular points)
   - iterate once ascending and once descending through all junctions
   - extent paths through junctions by duplication of edges
   - find non crossing concatenation of paths

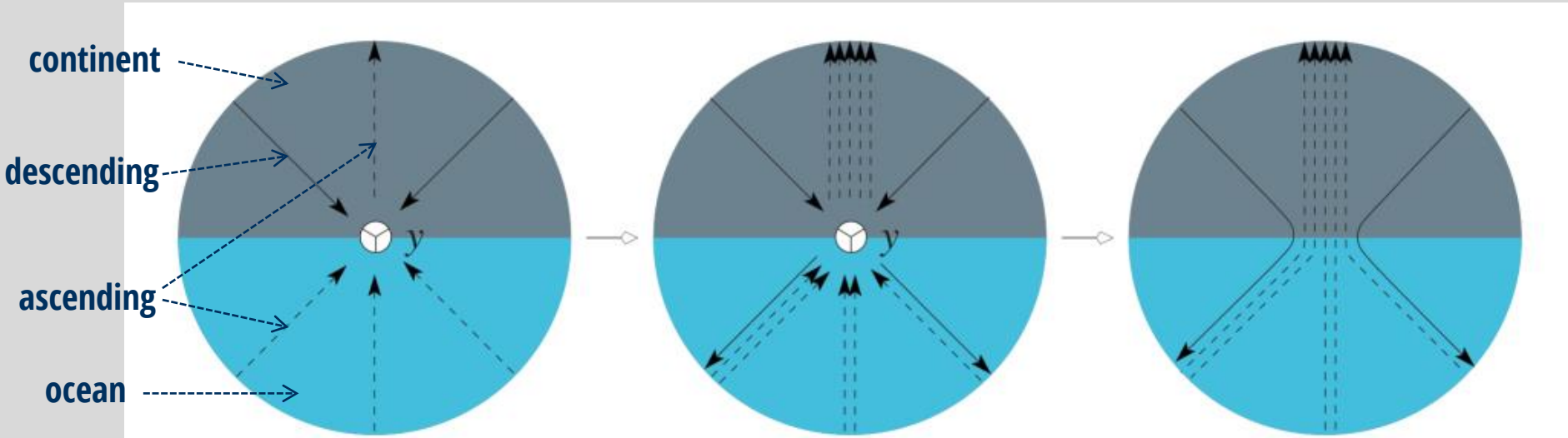


Fig. 6. Paths ending at junctions are extended by duplication and concatenation.

Construction of Quasi MS Complex

Stage 2 (path extension):

1. junction removal (regular points)
2. Ensure transversality at saddles
   - consider each saddle sector between two extremal edges
   - root paths of interior edges of sector through duplicated sector edge of same direction
   - find non crossing concatenation of paths

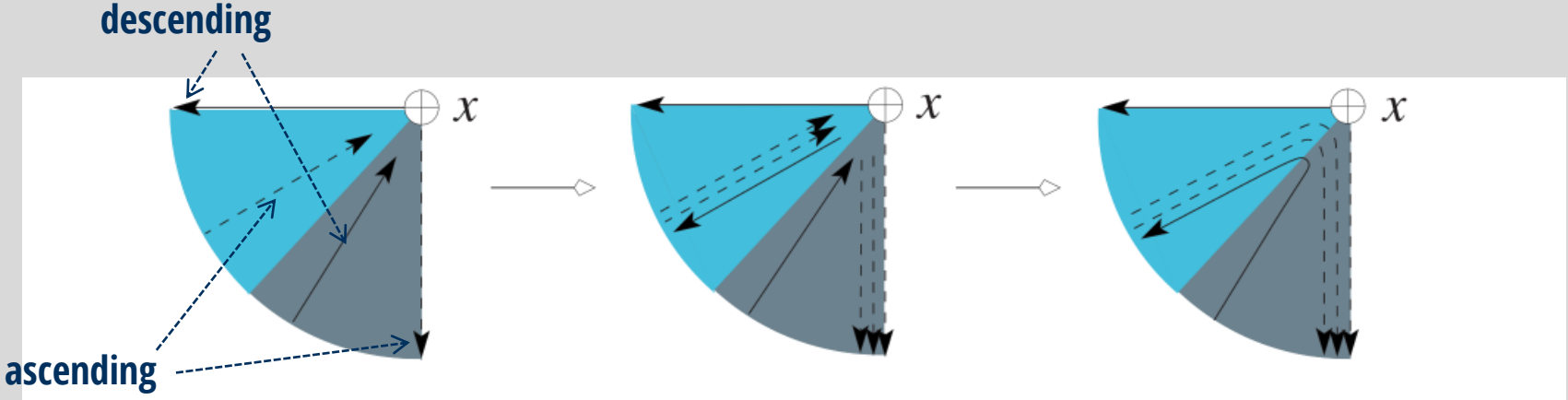


Fig. 7. Paths that end at a saddle by case (c) are extended by duplication and concatenation.

Construction of Quasi MS Complex

Stage 3 (unfolding of multi-saddles)

- Each k-fold saddle is unfolded into k simple saddles by k-1 times splitting off a single saddle through duplication of an ascending and a descending path

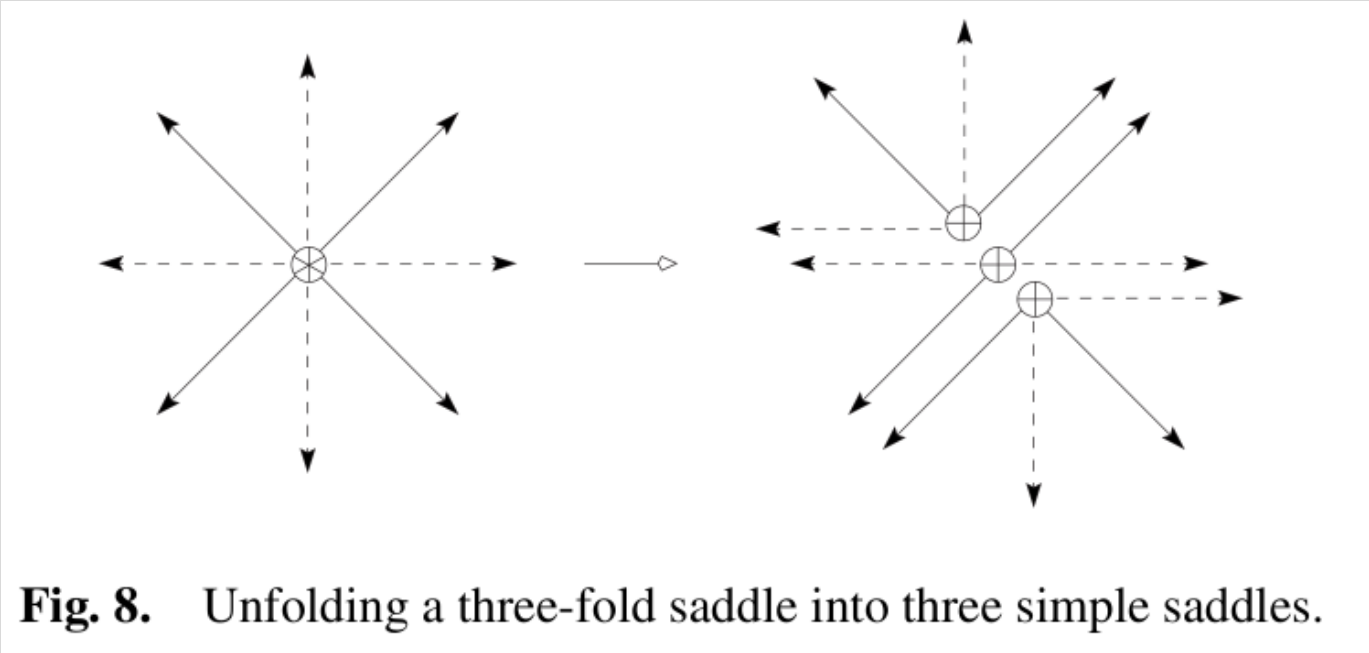
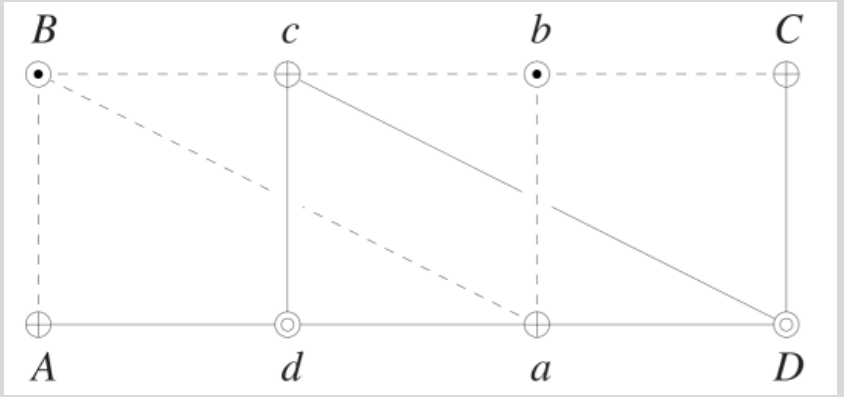


Fig. 8. Unfolding a three-fold saddle into three simple saddles.

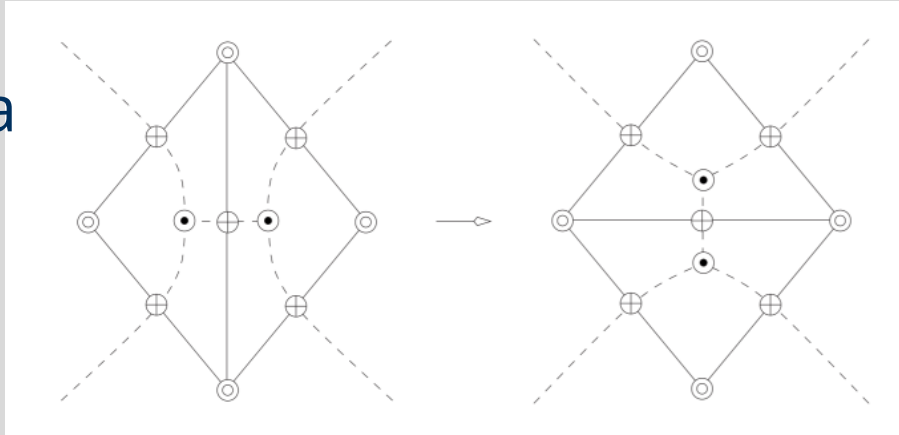
- The resulting complex is a quasi Morse-Smale Complex

Local Transformations

- Local transformations are performed through handle slide operation illustrated on right
- A handle slide is performed if the re-rooting of paths Ba and cD yields steeper ascends.
- The minima can be triangulated with the maxima as triangle centers and the saddles on edges
- A triangle flip here corresponds to two handle slide operations.



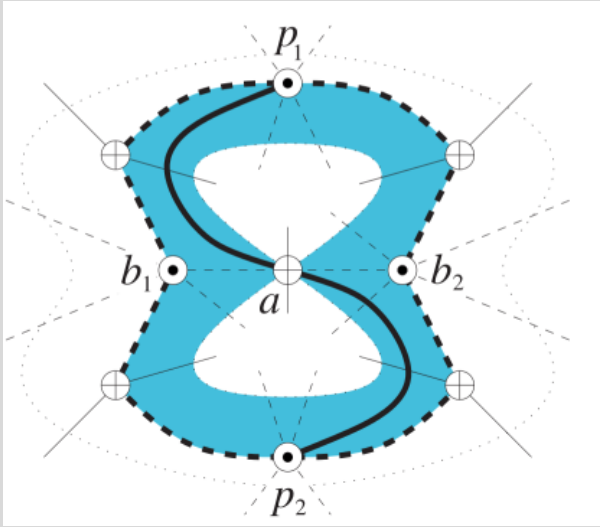
handle slide operation replaces dcBA,abcd,baDC with BAda,BaDc,cDCb

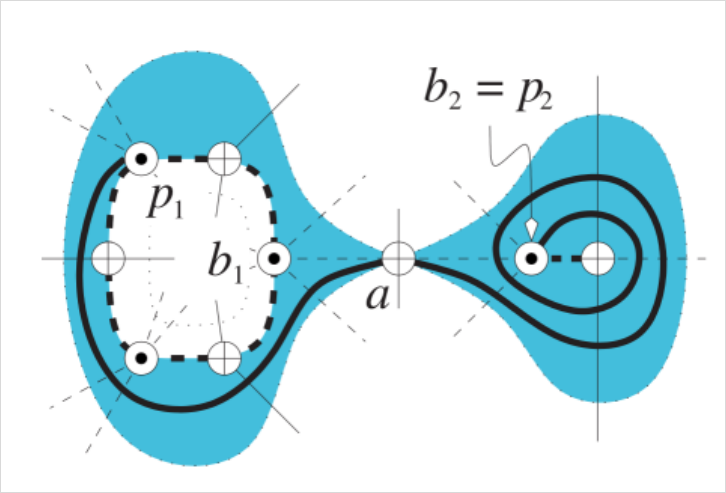


two handle slide operations can be interpreted as edge flip on triangulation of minima

Local Transformations

- There are two situations around vertex a for the handle slide operation:
  - One annulus touches itself at a
  - Two annuli touch at a
- Path re-rooting is only possible if no other path is crossed
- Potential path crossings are resolved by first performing handle slide operations on the to be crossed path.





Morse-Smale Complex Algorithm

- Edelsbrunner et. al also propose in 2003 an algorithm to construct the Morse-Smale Complex on a PL **3D-manifold** which is composed of critical vertices, edges, quadrangular faces and crystals which are cubic in the regular cases.
- The runtime is $O(n \cdot \log n)$ and due to the construction of a filtration.
- Algorithm overview:
  - Construct complex of descending manifolds
  - Construct pieces of ascending manifolds inside of descending manifolds
- Currently no stable implementation of this algorithm exists. Alternatives are the use of discrete Morse Theory or approximation via watershed transformation

Applications

BUBBLE COUNTING

Analyzing Rayleigh-Taylor Instability

https://www.youtube.com/watch?v=bW4526vHnY0

Counting Bubbles – Z-Maxima

Laney, Daniel, et al. "Understanding the structure of the turbulent mixing layer in hydrodynamic instabilities." ***IEEE Transactions on Visualization and Computer Graphics*** **12.5 (2006): 1053-1060.**

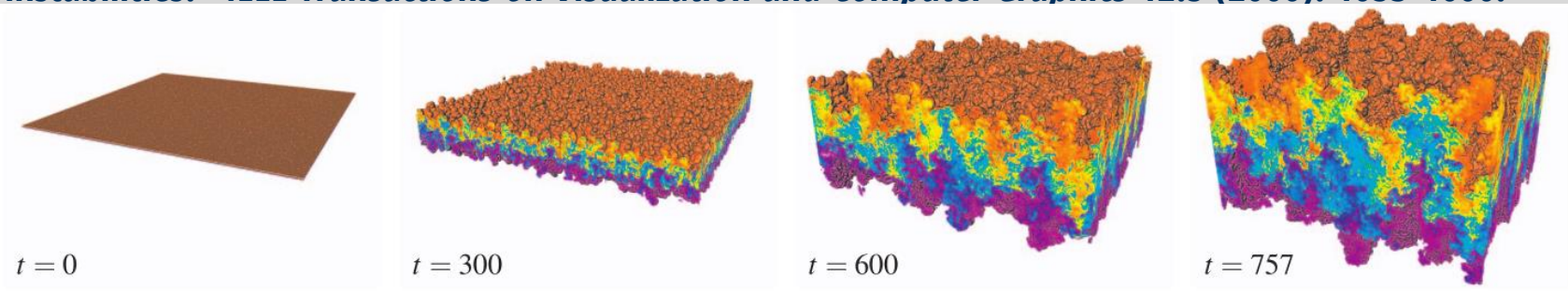


Fig. 1. An overview of a simulation (periodic in x and y) of Rayleigh-Taylor instability at four time steps on a 1152^3 grid. The light fluid has a density of 1.0, the heavy fluid has a density of 3.0. Two envelope surfaces (at densities 1.02 and 2.98) capture the mixing region. The boundaries of the box show the density field in pseudocolor. The heavy fluid is red and the light fluid is blue. Other colors represent intermediate compositions of mixed fluid. We analyze the upper envelope (red) to study bubble structures and the midplanes to study mixing trends.

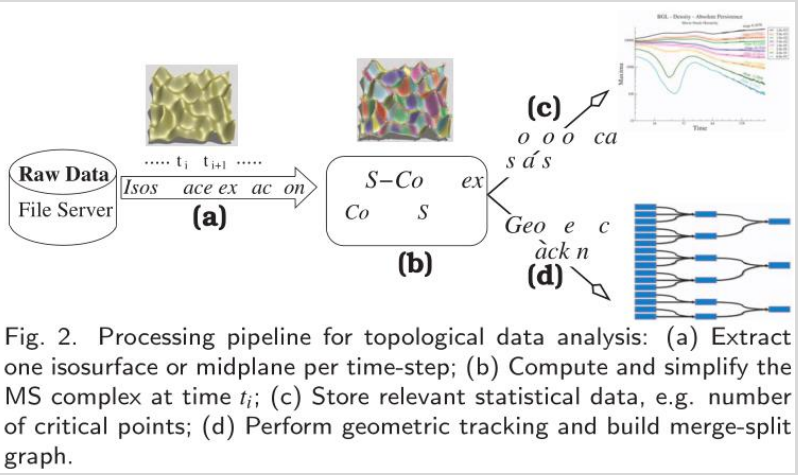
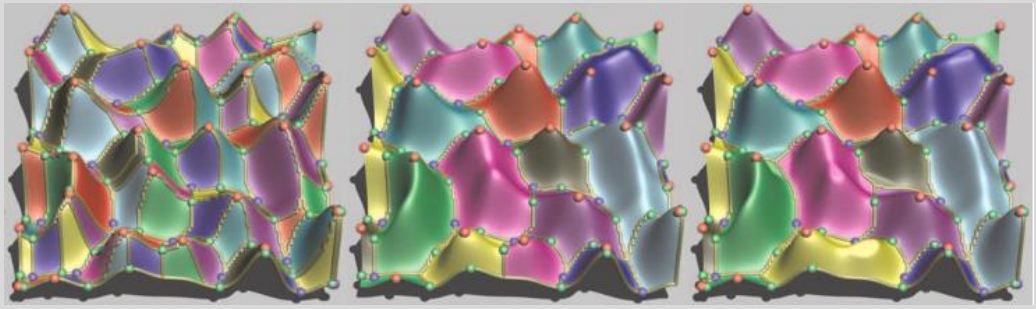


Fig. 2. Processing pipeline for topological data analysis: (a) Extract one isosurface or midplane per time-step; (b) Compute and simplify the MS complex at time t_i; (c) Store relevant statistical data, e.g. number of critical points; (d) Perform geometric tracking and build merge-split graph.



MS Complex

Stable manifold of simplified MS complex

simplified iso-surface with simplified MS complex

Counting Bubbles – Other Functions

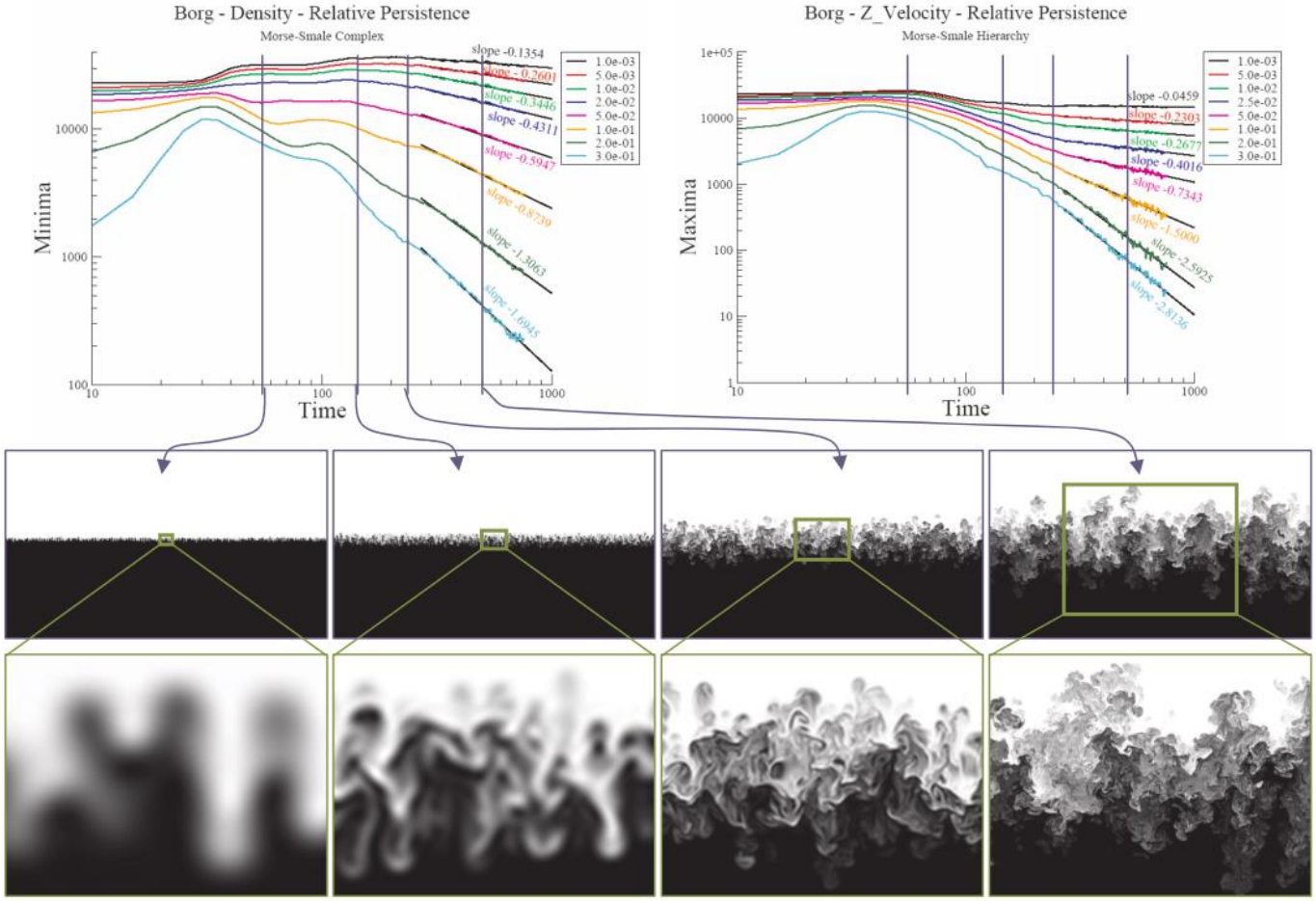


Fig. 5. Critical point counts for the 1152^3 Borg simulation, showing density minima (left) and Z velocity maxima (right) at the midplane versus time. Density minima indicate the locations where light fluid is intersecting the midplane. At left, the four phases of the mixing are suggested by four lines at distinct times in the middle of each phase and arrows indicating the corresponding XZ slices of the density field, where white denotes the heavy fluid. The linear regions of the high persistence curves suggest power-law behavior. Recall that reductions in the number of high persistence critical points is correlated with the structures in the flow becoming larger. Thus, larger structures are indicative of growth of the mixing region. The asymptotic behavior is fitted for late time.

Persistence Threshold Adjustment

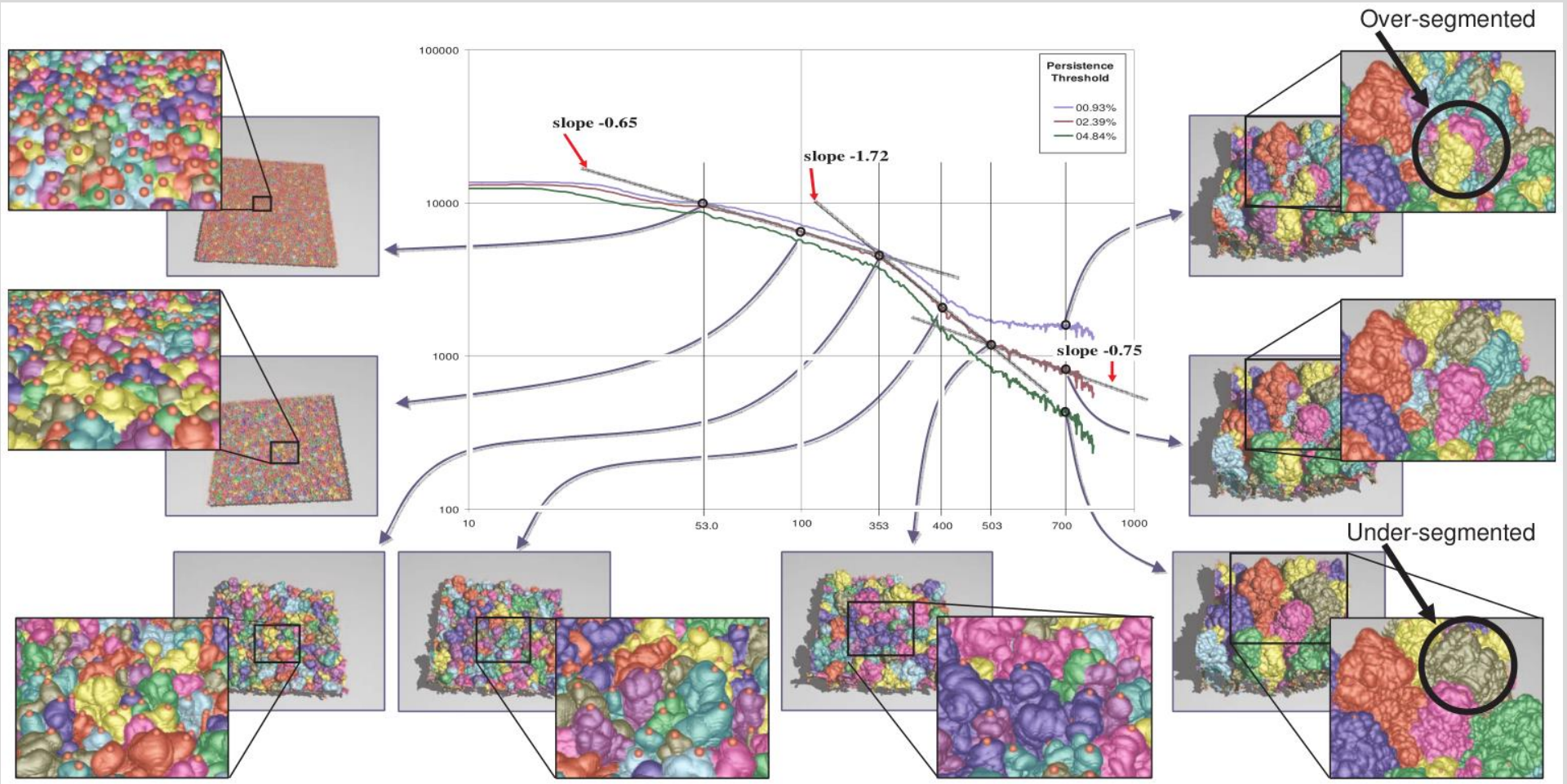


Fig. 7. The plot depicts bubble counts of the envelope surface for three persistence values. Three regions of power-law behavior are shown by the curve fits in gray to the 2.39% persistence curve. To the right of the plot, the MS-segmentation at three persistence values for time 700. To the left and below the plot, the bubble segmentation along the 2.39% persistence curve at various times. Each maximum along with the Morse cells of its child-maxima are colored the same.