June 25, 2010

Algebraic Topology: Methods, Computation and Science

Persistence-based Clustering and Segmentation

Primoz Skraba Frédéric Chazal Leo Guibas

Steve Oudot Maks Ovsjanikov

F. Chazal, L. J. Guibas, S. Y. Oudot, P. Skraba. *Persistence-Based Clustering in Riemannian Manifolds.* INRIA Research Report RR-6968, June 2009.

P. Skraba, M. Ovsjanikov, F. Chazal, and L. Guibas, *Persistence-based Segmentation of Deformable Shapes*, 3rd Workshop on Non-Rigid Shape Analysis and Deformable Image Alignment, Proc. CVPR, 2010

Outline

Persistence-based Clustering

Stability

Basins of Attraction

Applications/Choice of Functions

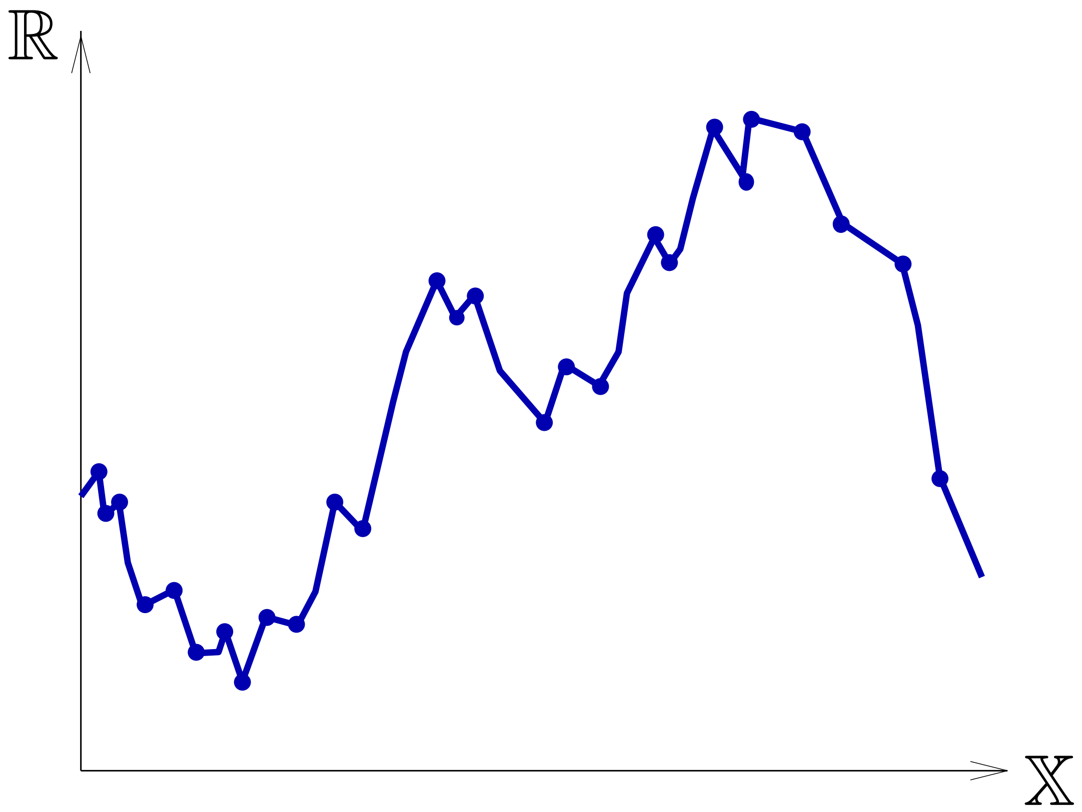
Clustering – Density Function

Mesh Segmentation – Heat Kernel Signature

Dealing with Instability

Finding unstable regions

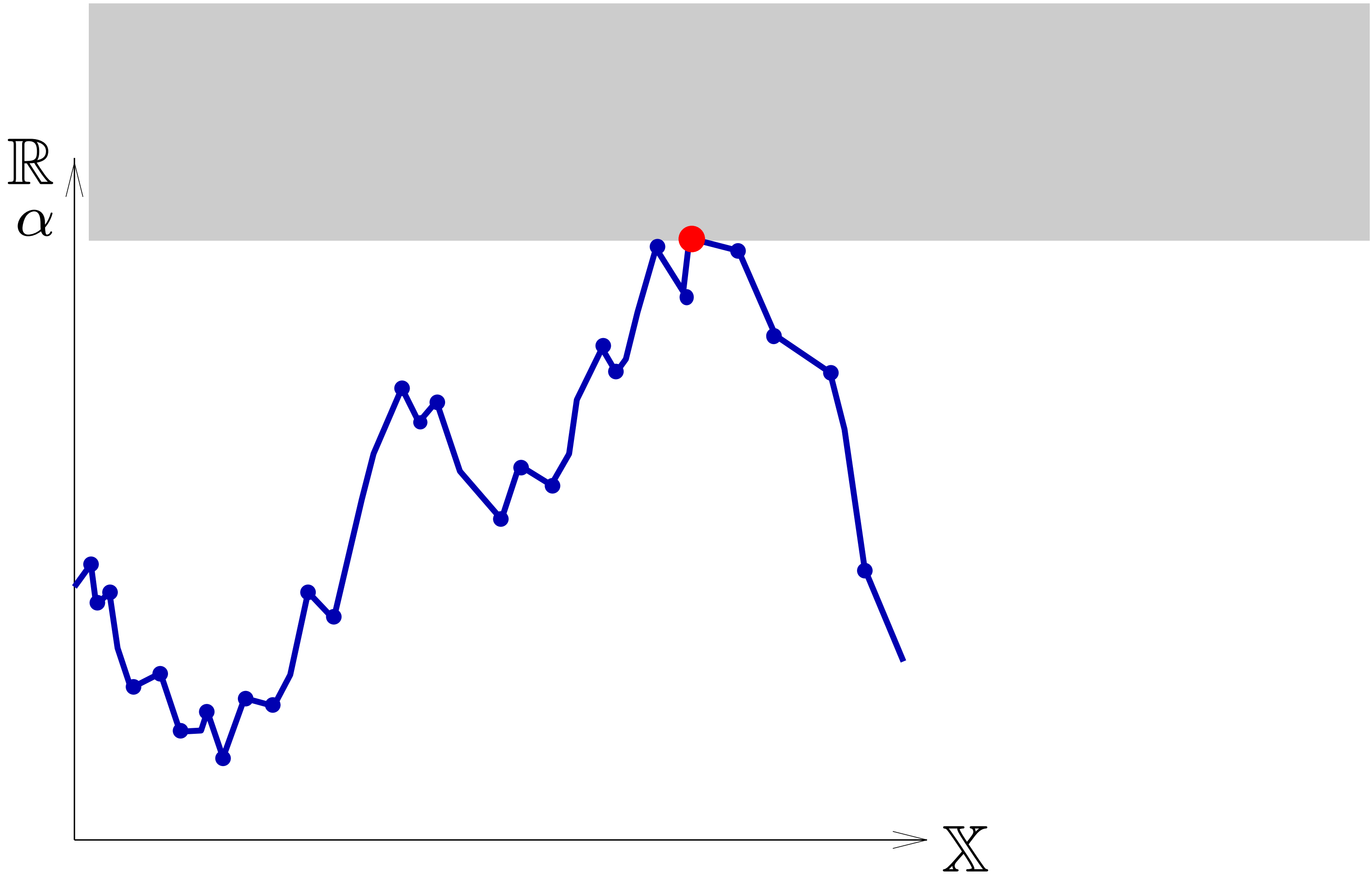
Computing Basins



How do we compute basins from a PD?

- Do not merge basins with persistence less than a threshold τ!

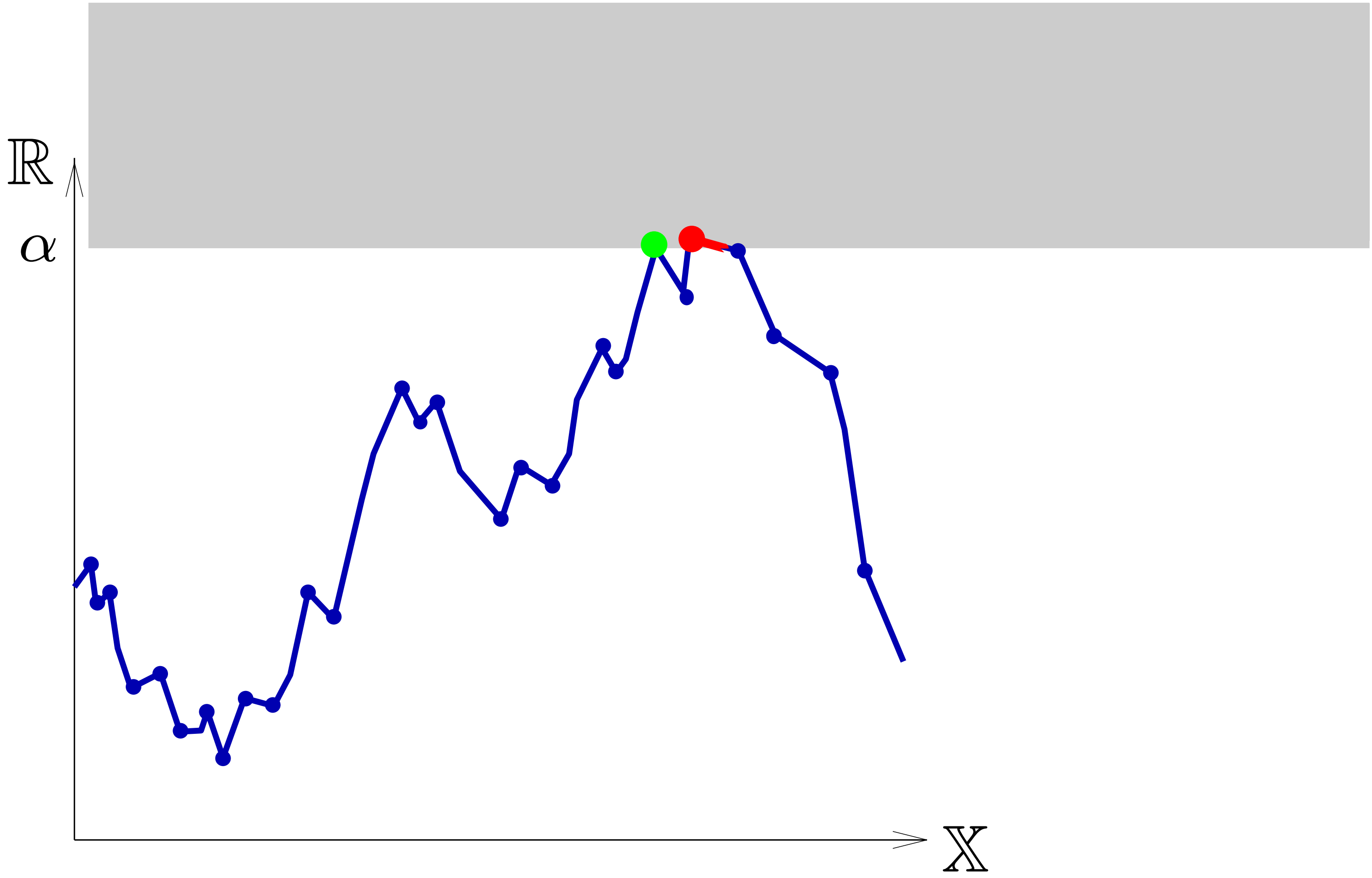
Computing Basins



How do we compute basins from a PD?

- Do not merge basins with persistence less than a threshold τ!

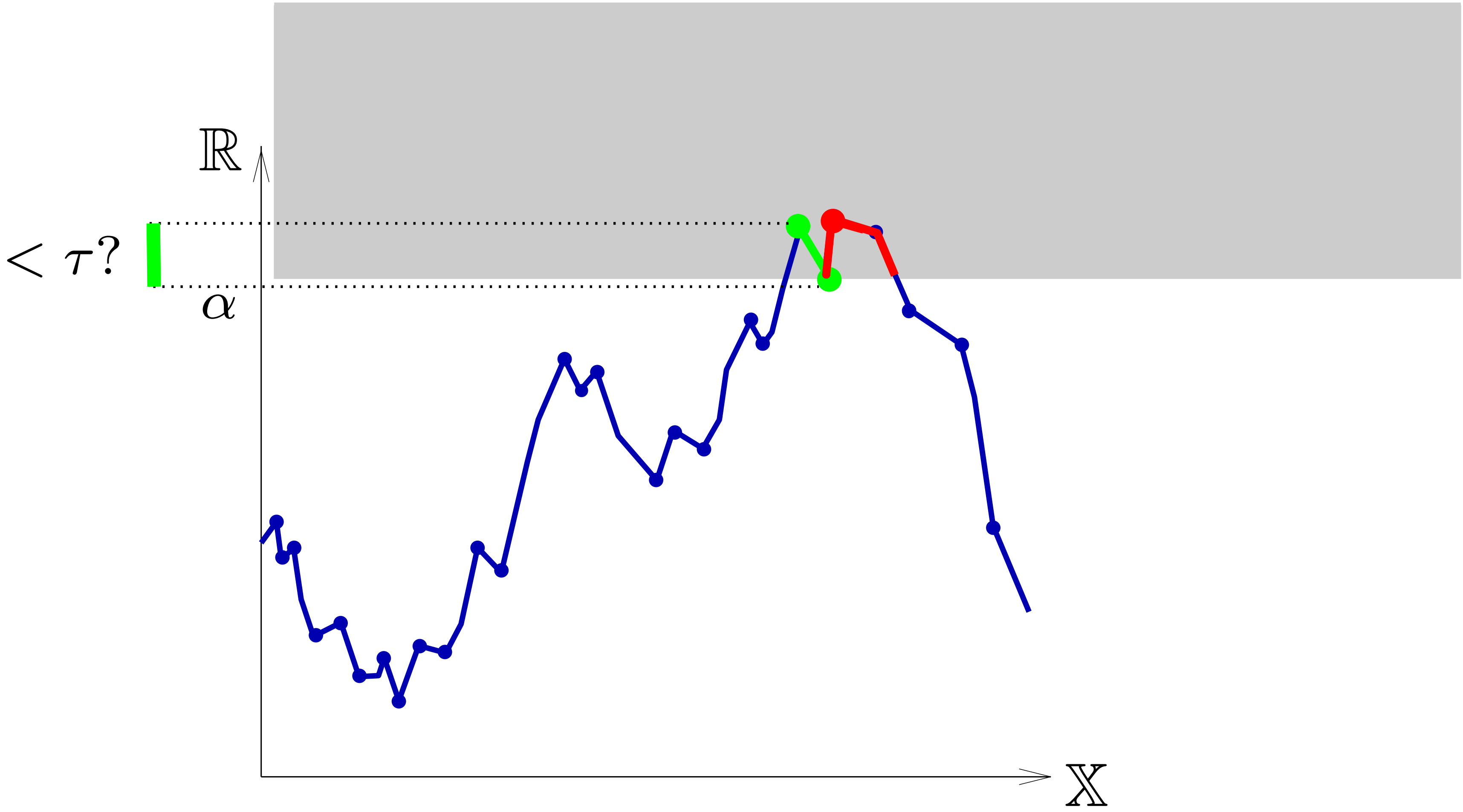
Computing Basins



How do we compute basins from a PD?

- Do not merge basins with persistence less than a threshold τ!

Computing Basins

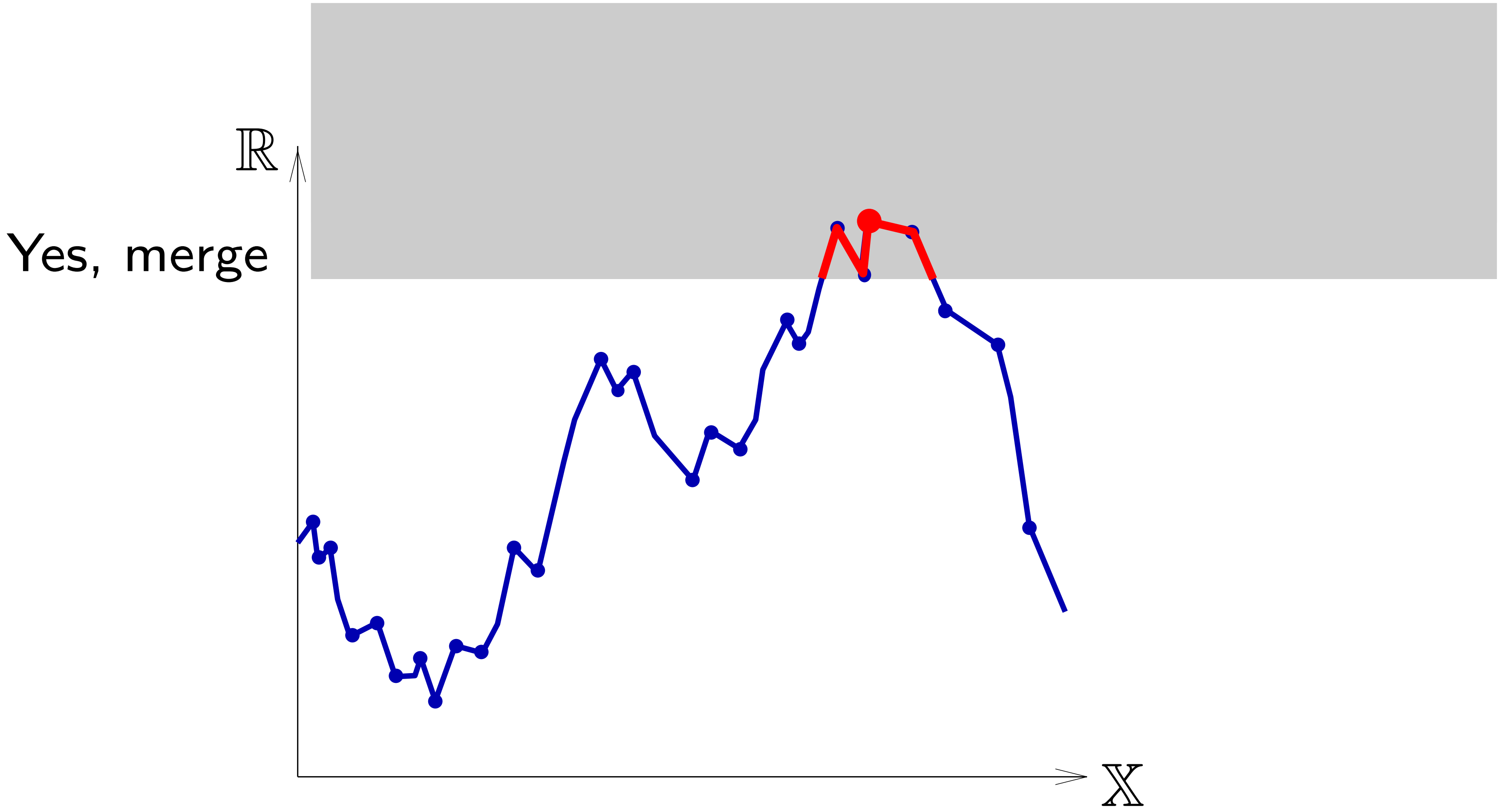


How do we compute basins from a PD?

- Do not merge basins with persistence less than a threshold τ!

$< \tau?$

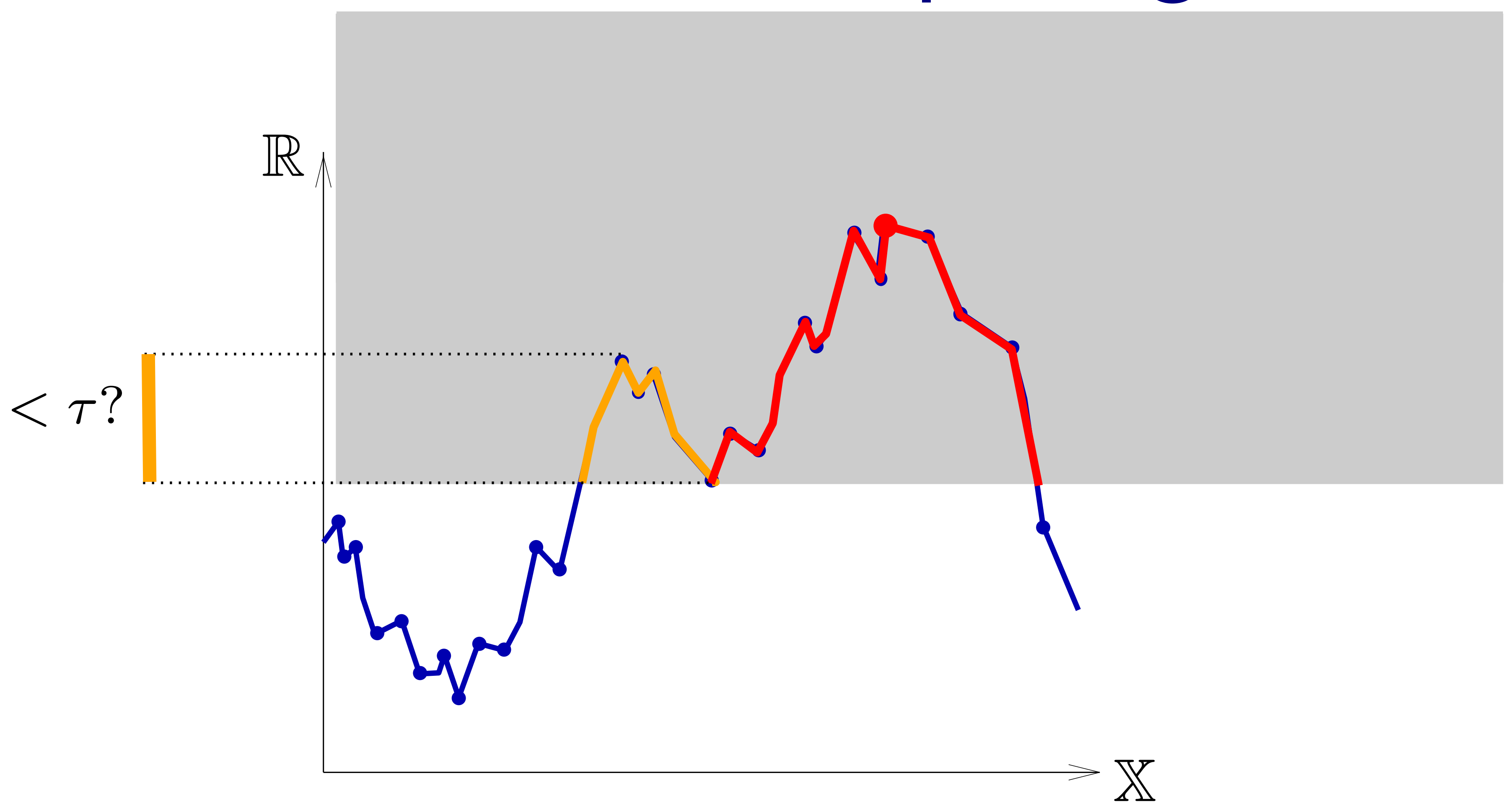
Computing Basins



How do we compute basins from a PD?

- Do not merge basins with persistence less than a threshold τ!

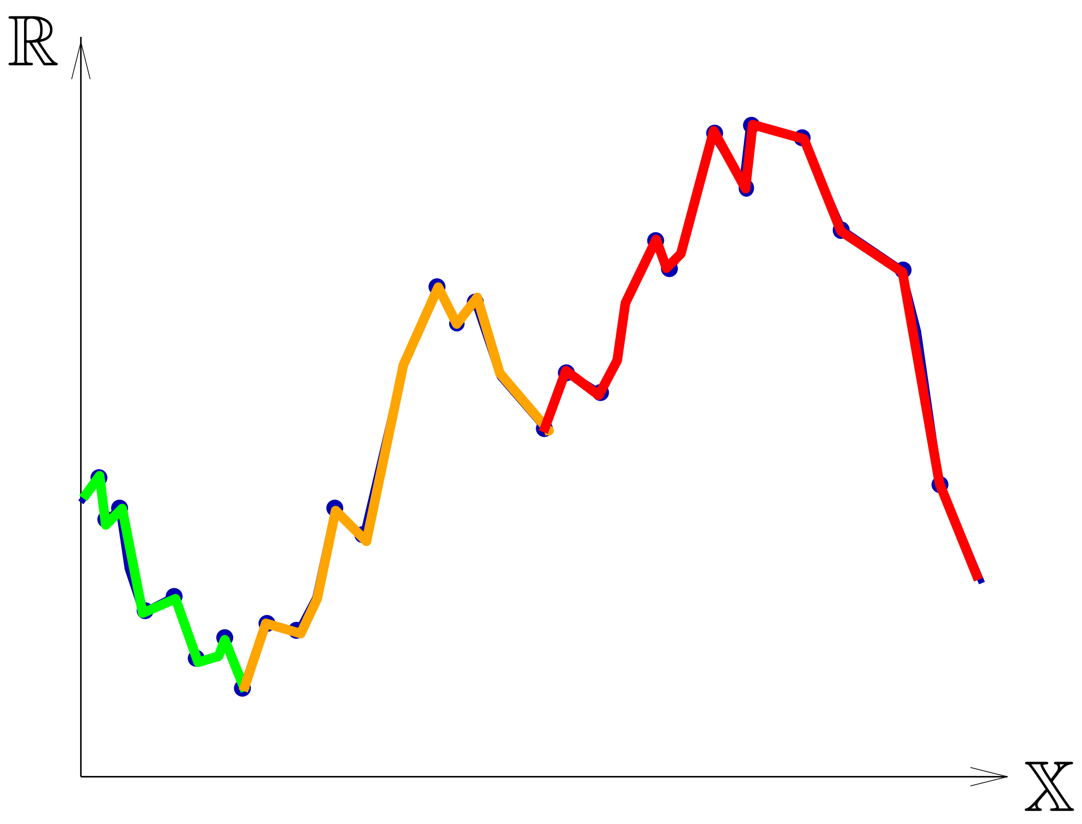
Computing Basins

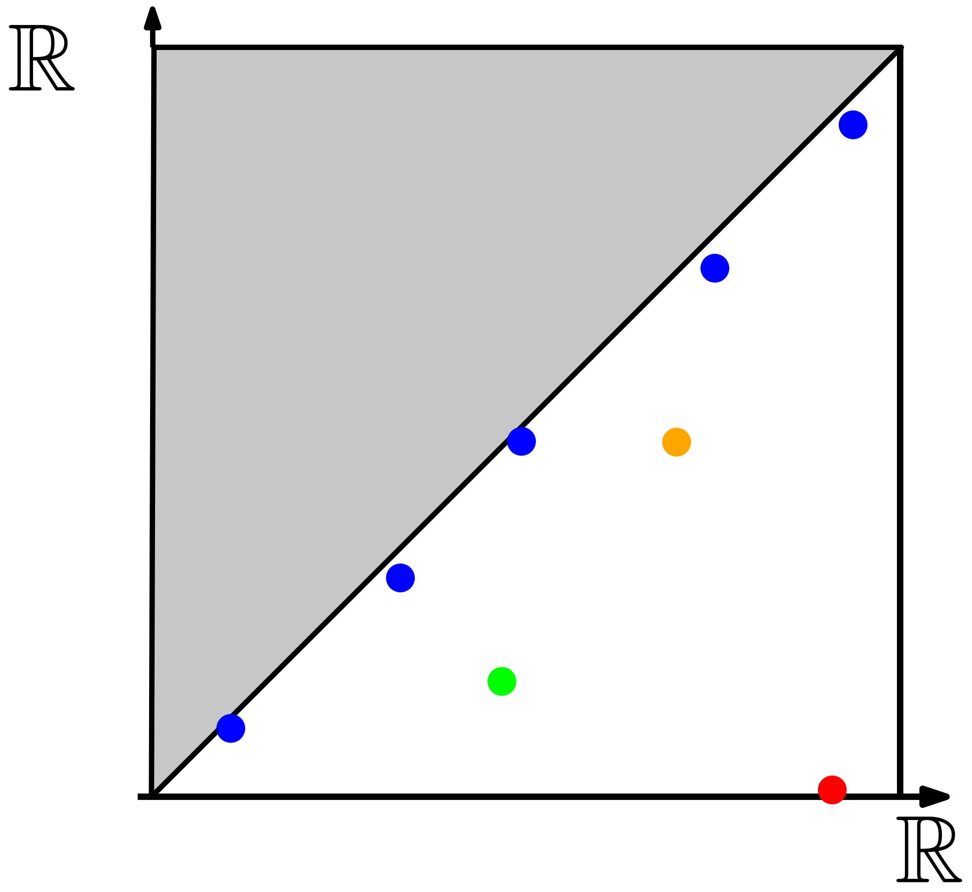


How do we compute basins from a PD?

- Do not merge basins with persistence less than a threshold τ!

Computing Basins





How do we compute basins from a PD?

- Do not merge basins with persistence less than a threshold τ!

Algorithm

- Input: $f(x), \mathcal{G}, \alpha$

Algorithm

- Input: $f(x), \mathcal{G}, \alpha$

1. Sort x according to f
2. For $x \in L$

   2a. For neighbors of x in $\mathcal{G}$

   If no higher neighbors $\Rightarrow$ new cluster

   else assign x to ∇f

   2b. For adjacent clusters y to x

   if $|f(y) - f(x)| \leq \alpha$

   merge into oldest adjacent cluster

Algorithm

- Input: $f(x), \mathcal{G}, \alpha$

1. Sort x according to f
2. For $x \in L$

   2a. For neighbors of x in $\mathcal{G}$

   If no higher neighbors $\Rightarrow$ new cluster

   else assign x to ∇f

   2b. For adjacent clusters y to x

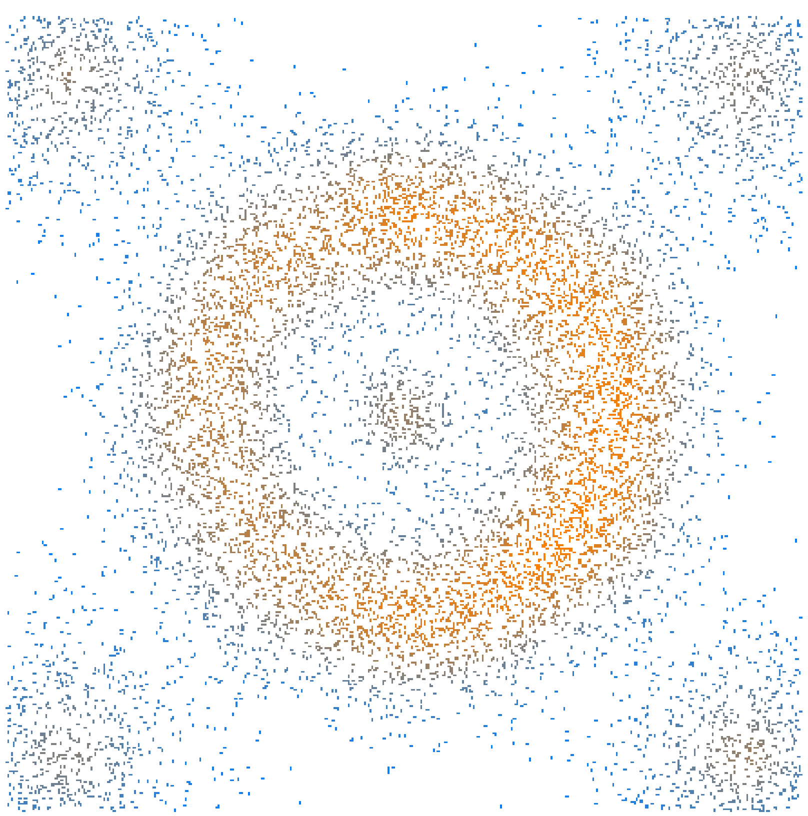
   if $|f(y) - f(x)| \leq \alpha$

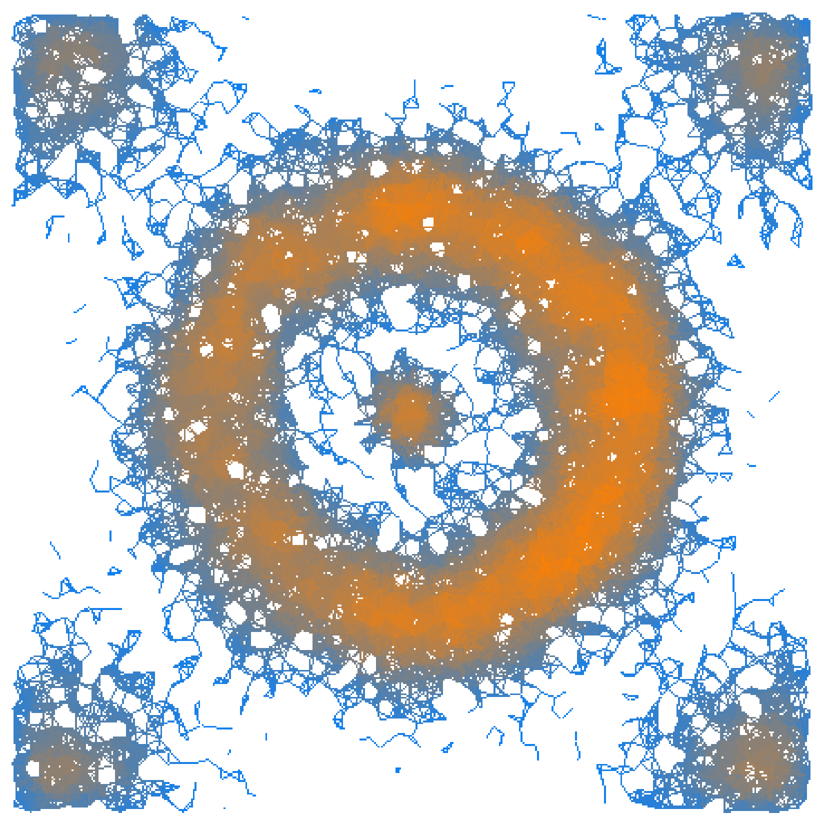
   merge into oldest adjacent cluster

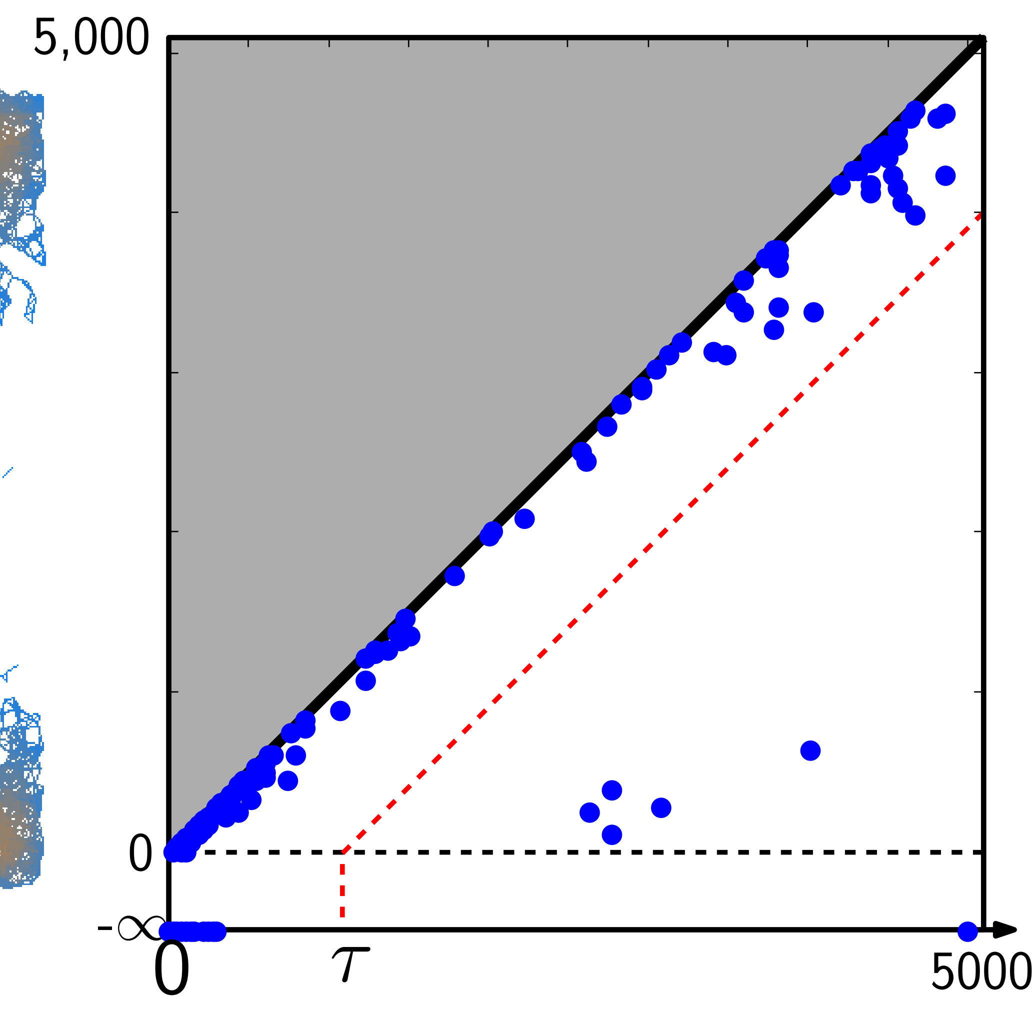
Union-Find!

Putting it together

- Compute function
- Run algorithm with $\alpha = \infty$
  - Standard persistence algorithm
- Use persistence diagram to choose threshold
- Re-run algorithm

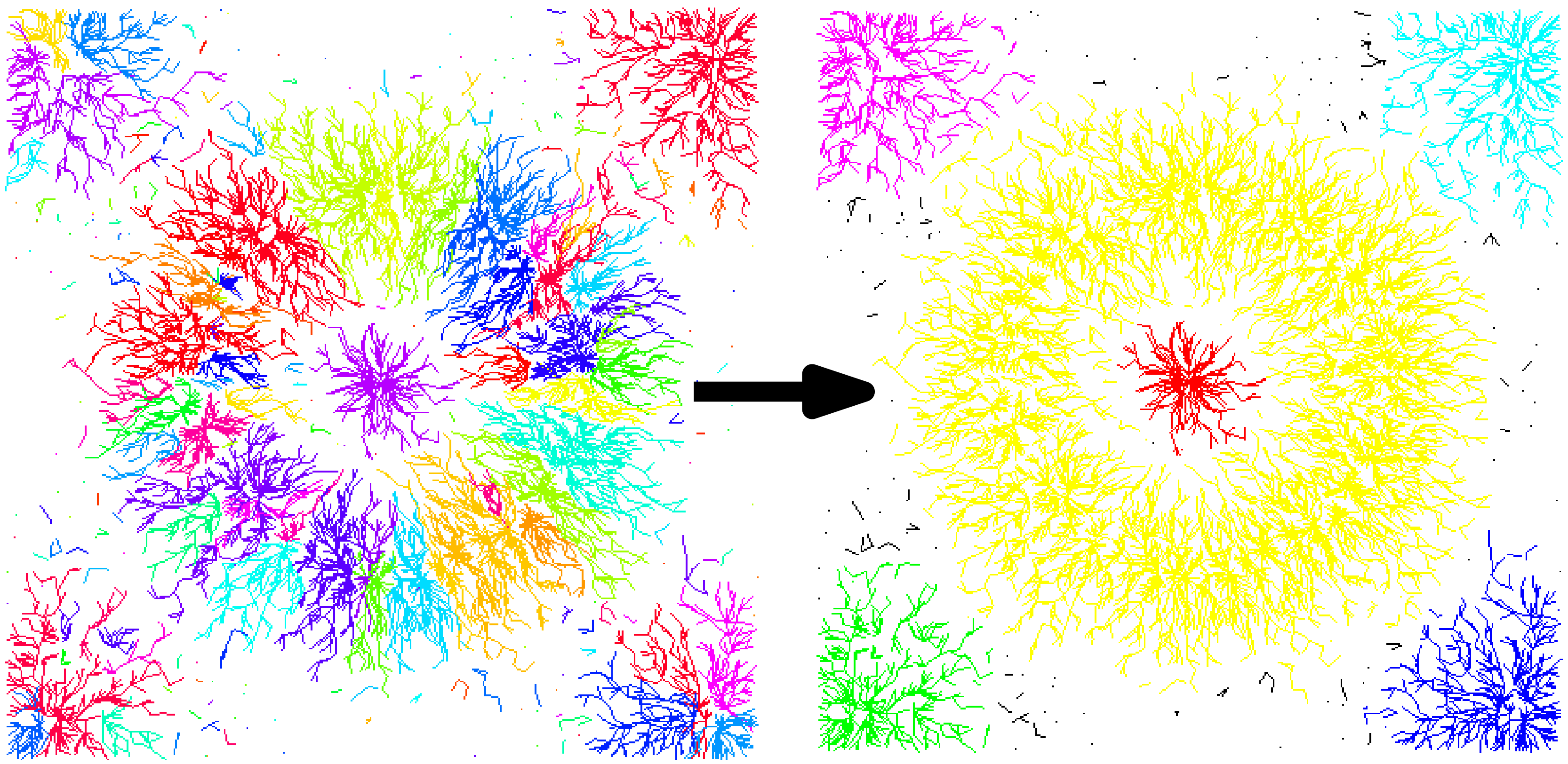


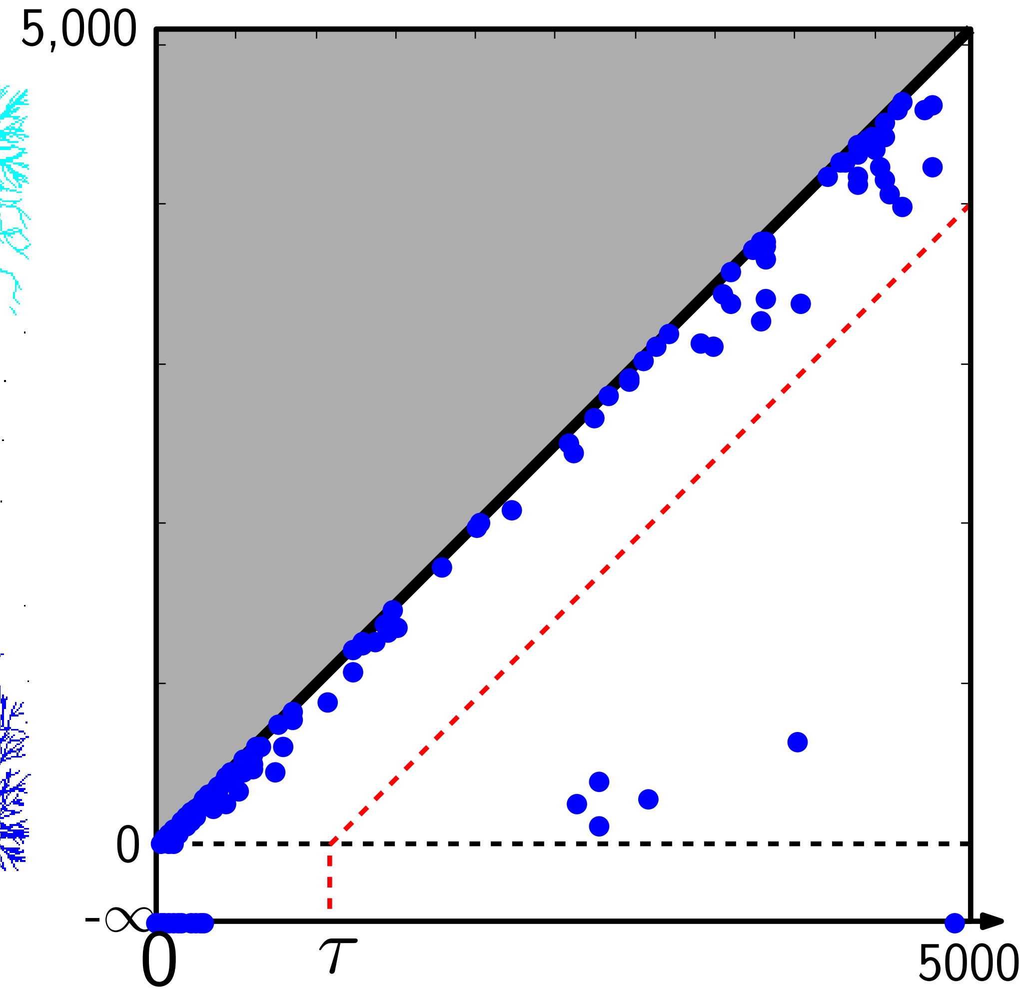




Putting it together

- Compute function
- Run algorithm with $\alpha = \infty$
  - Standard persistence algorithm
- Use persistence diagram to choose threshold
- Re-run algorithm



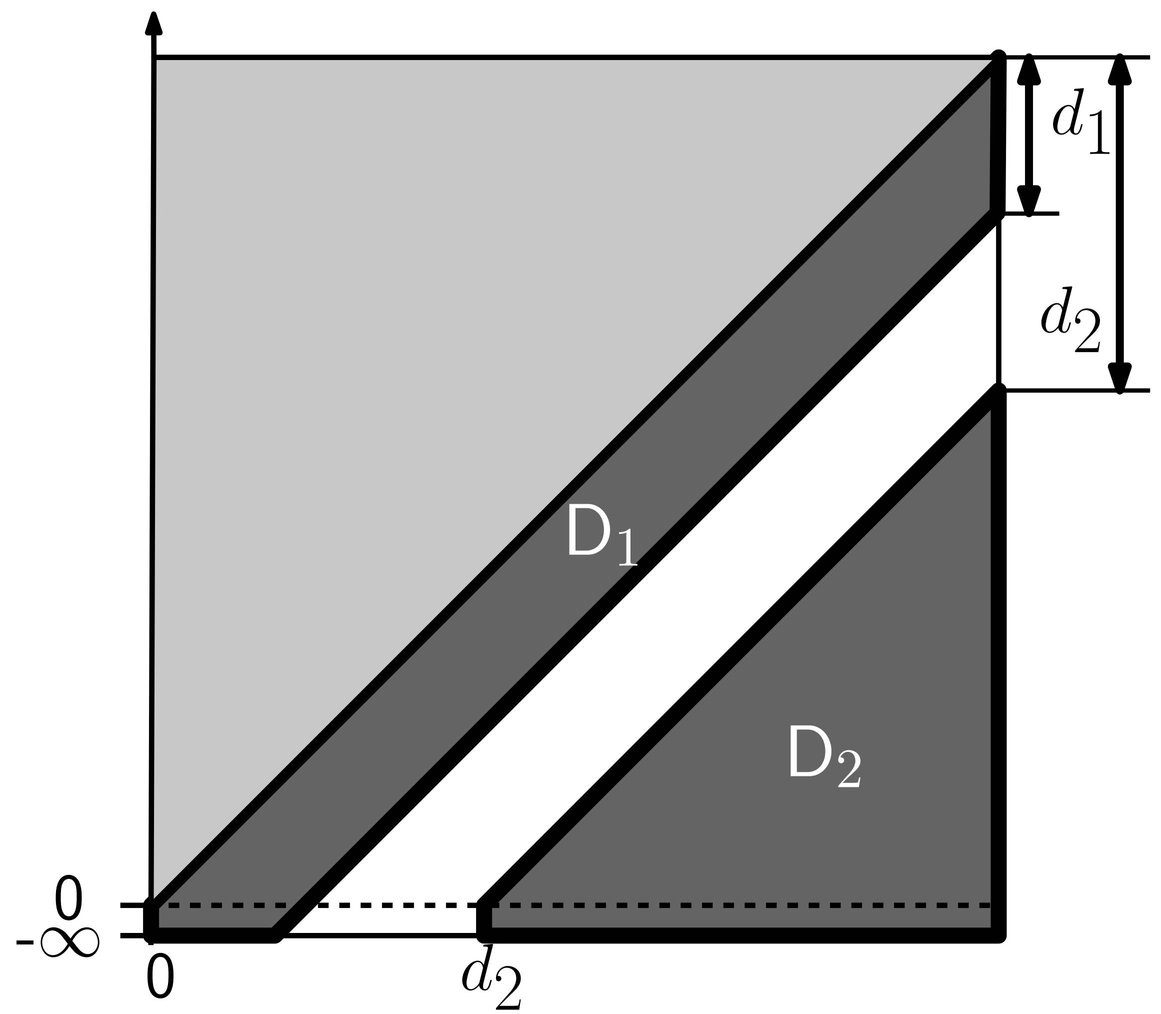


Feedback & Interpreting Diagrams

- If peaks are prominent enough, number of clusters/segments is stable
- Theoretically,
  - The number of clusters/segments is stable
  - The “denser the sampling,” the smaller the noise

Feedback & Interpreting Diagrams

- If peaks are prominent enough, number of clusters/segments is stable
- Theoretically,
  - The number of clusters/segments is stable
  - The "denser the sampling," the smaller the noise

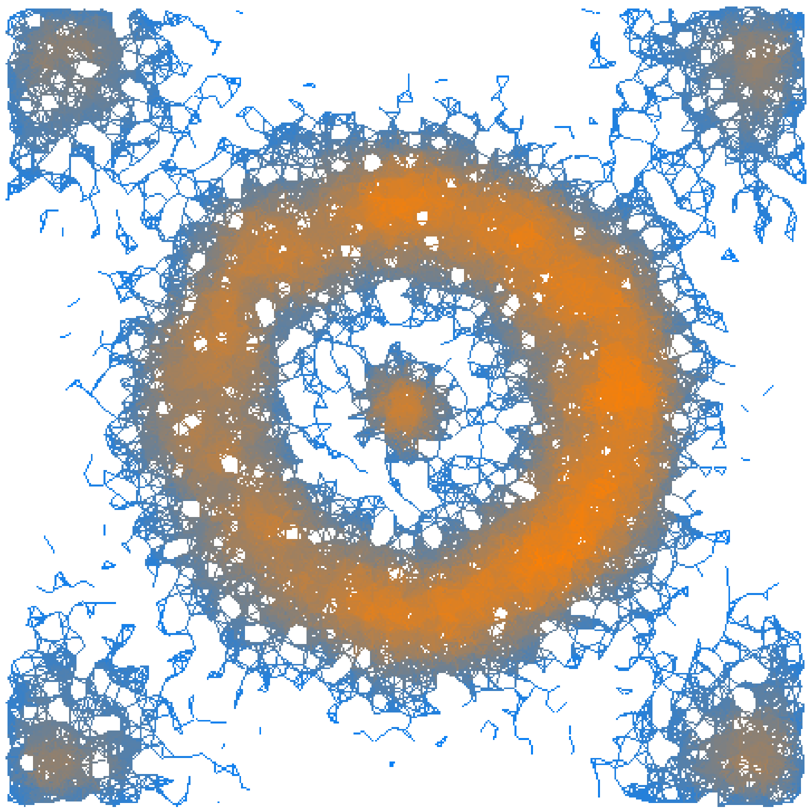


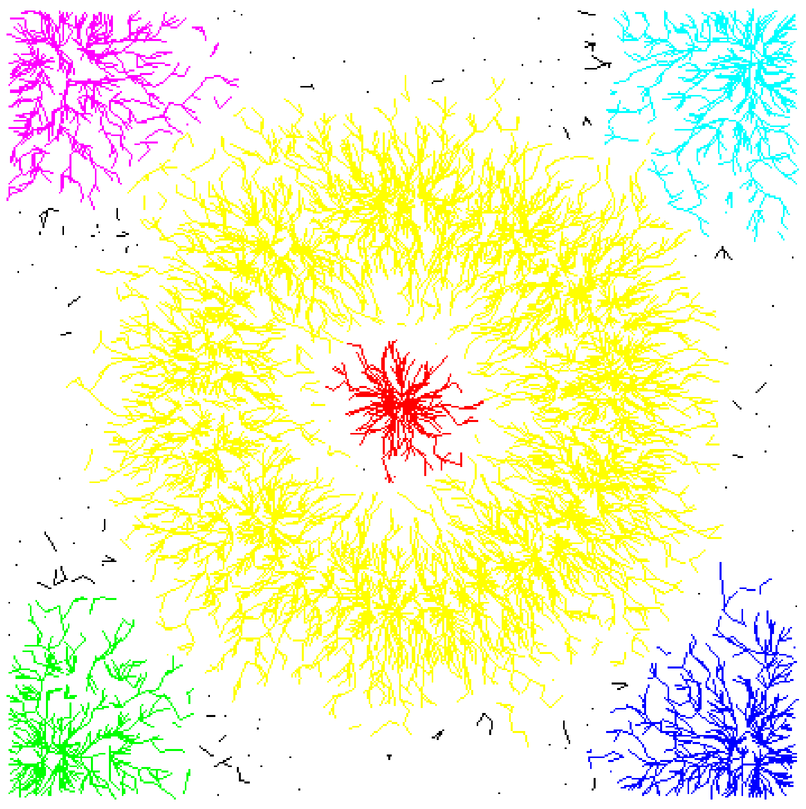
Feedback & Interpreting Diagrams

- If peaks are prominent enough, number of clusters/segments is stable
- Theoretically,
  - The number of clusters/segments is stable
  - The "denser the sampling," the smaller the noise
- Practically,
  - Gives a sense of stability of the number of basins
  - Choice of threshold transparent w.r.t. number of basins
  - Can help with the choice of other parameters

Clustering

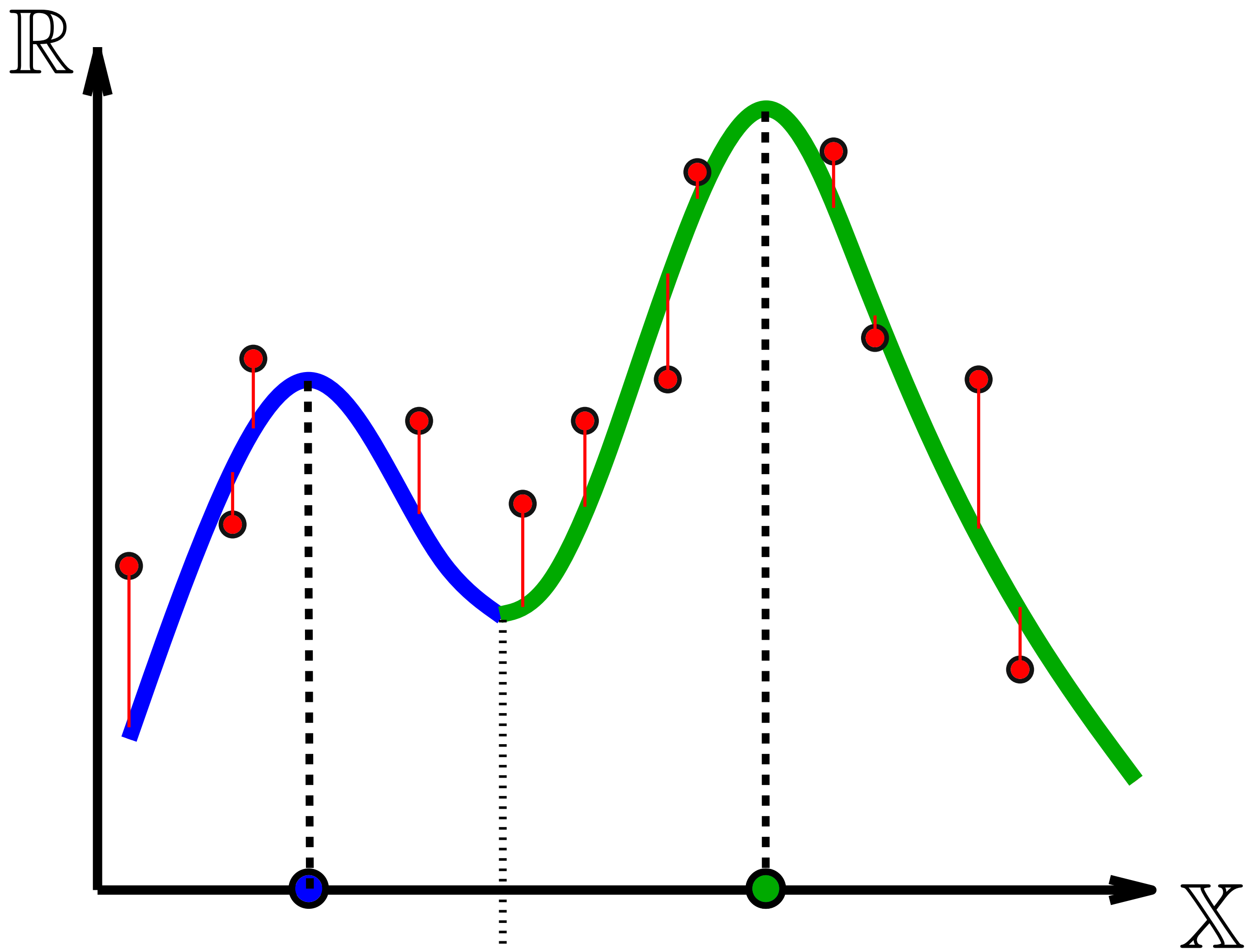
Find "important" components in a point cloud





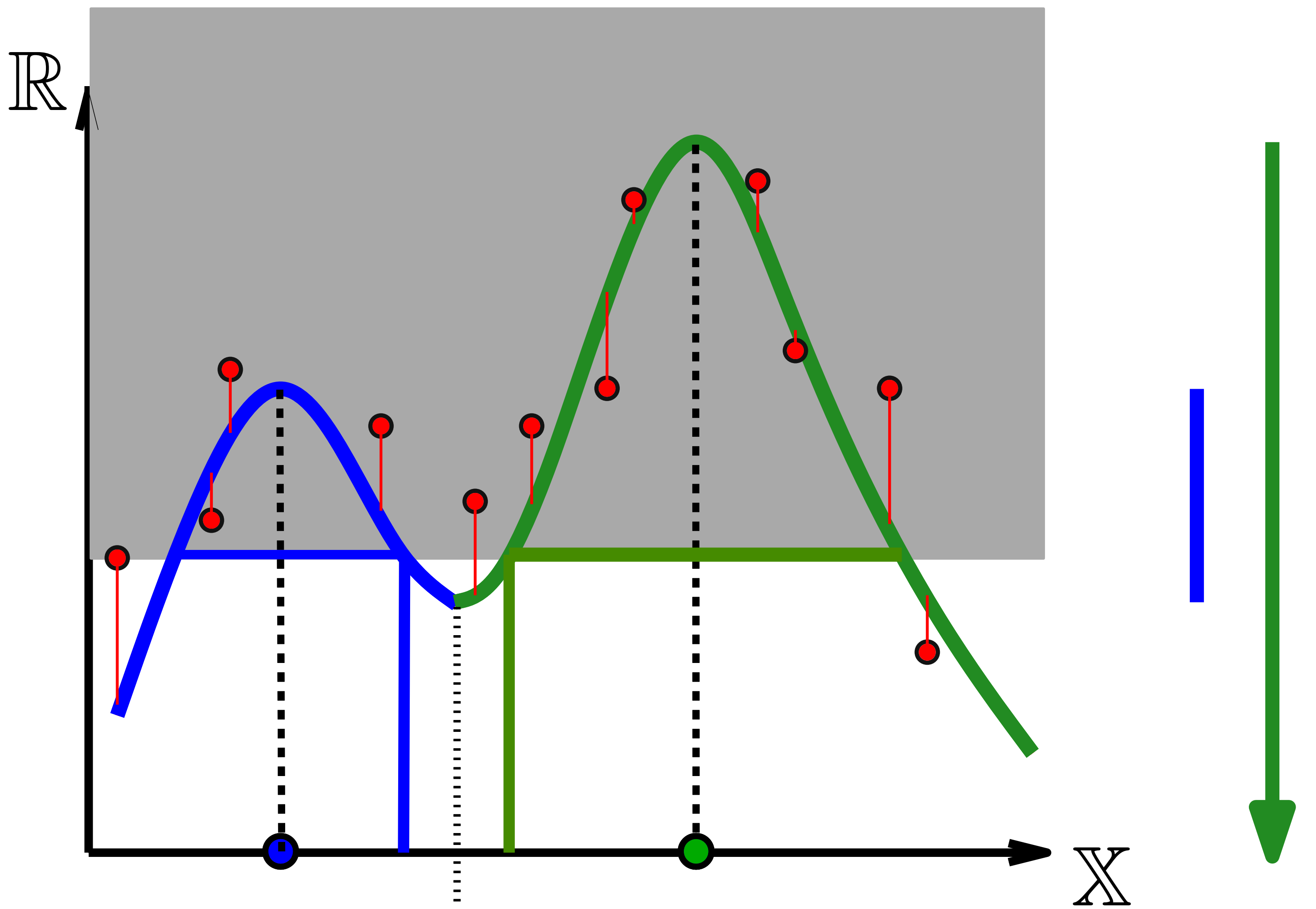
Clustering

Topological definition of a cluster



Clustering

Topological definition of a cluster



Prominent peaks correspond to persistent
connected components of the super-level set filtration of f

Density Estimation

- Gaussian kernel
  - Bandwidth parameter h

$$\hat{f}(p) = \frac{1}{|\mathcal{N}_p|} \sum_{q \in \mathcal{N}_p} e^{\frac{-d(p-q_i)^2}{2h}}$$

Density Estimation

- Gaussian kernel
  - Bandwidth parameter h

$$\hat{f}(p) = \frac{1}{|\mathcal{N}_p|} \sum_{q \in \mathcal{N}_p} e^{\frac{-d(p-q_i)^2}{2h}}$$

- Distance to a measure
  - Number of neighbors

$$\hat{f}(p) = \left(\frac{1}{k} \sum_{i=0}^{k-1} d(p, q_i)^2 \right)^{\frac{1}{2}}$$

Density Estimation

- Gaussian kernel
  - Bandwidth parameter h

$$\hat{f}(p) = \frac{1}{|\mathcal{N}_p|} \sum_{q \in \mathcal{N}_p} e^{\frac{-d(p-q_i)^2}{2h}}$$

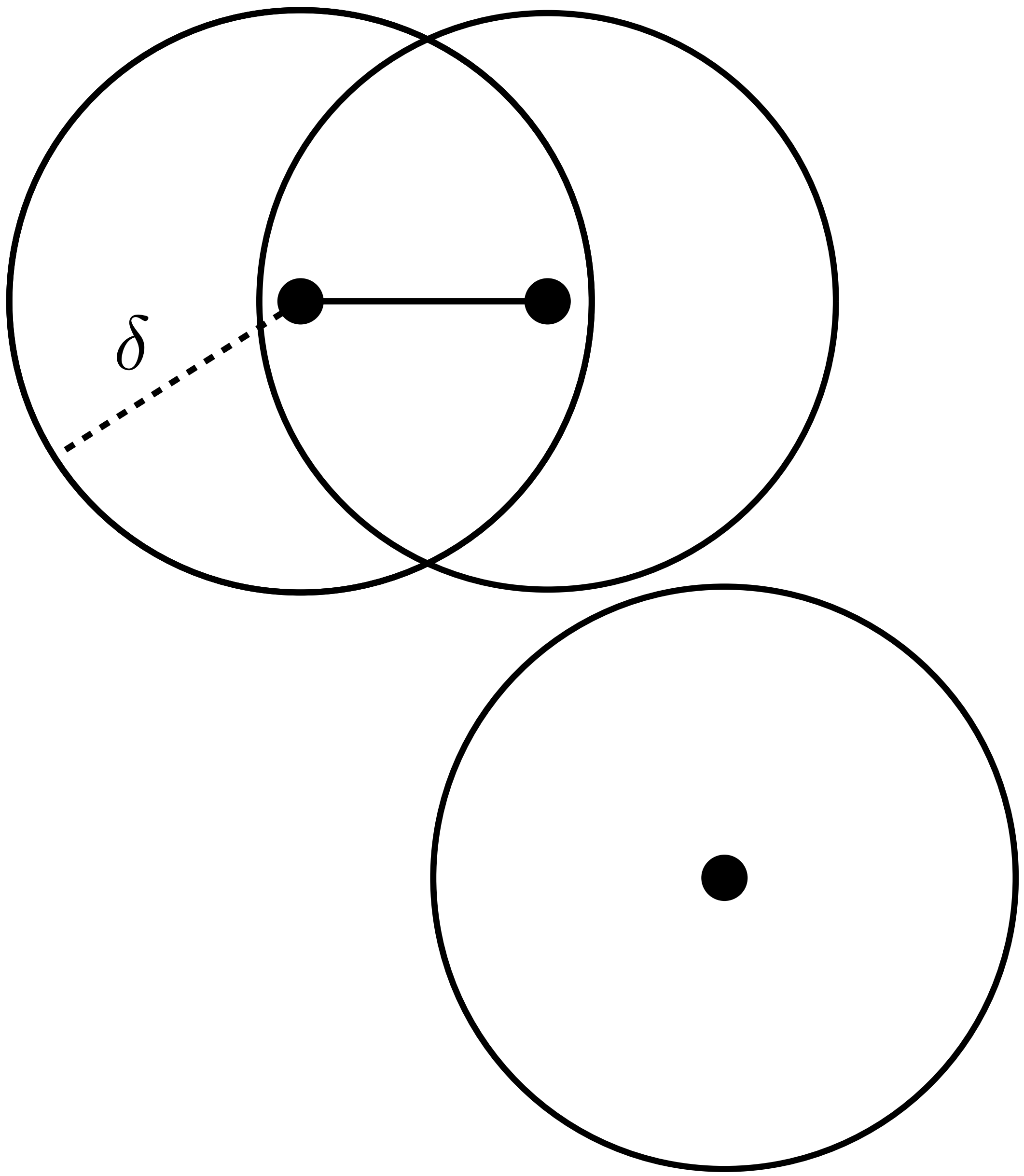
- Distance to a measure
  - Number of neighbors

$$\hat{f}(p) = \left(\frac{1}{k} \sum_{i=0}^{k-1} d(p, q_i)^2 \right)^{\frac{1}{2}}$$

$$\boxed{\text{Require } ||f - \hat{f}||_\infty \leq \epsilon}$$

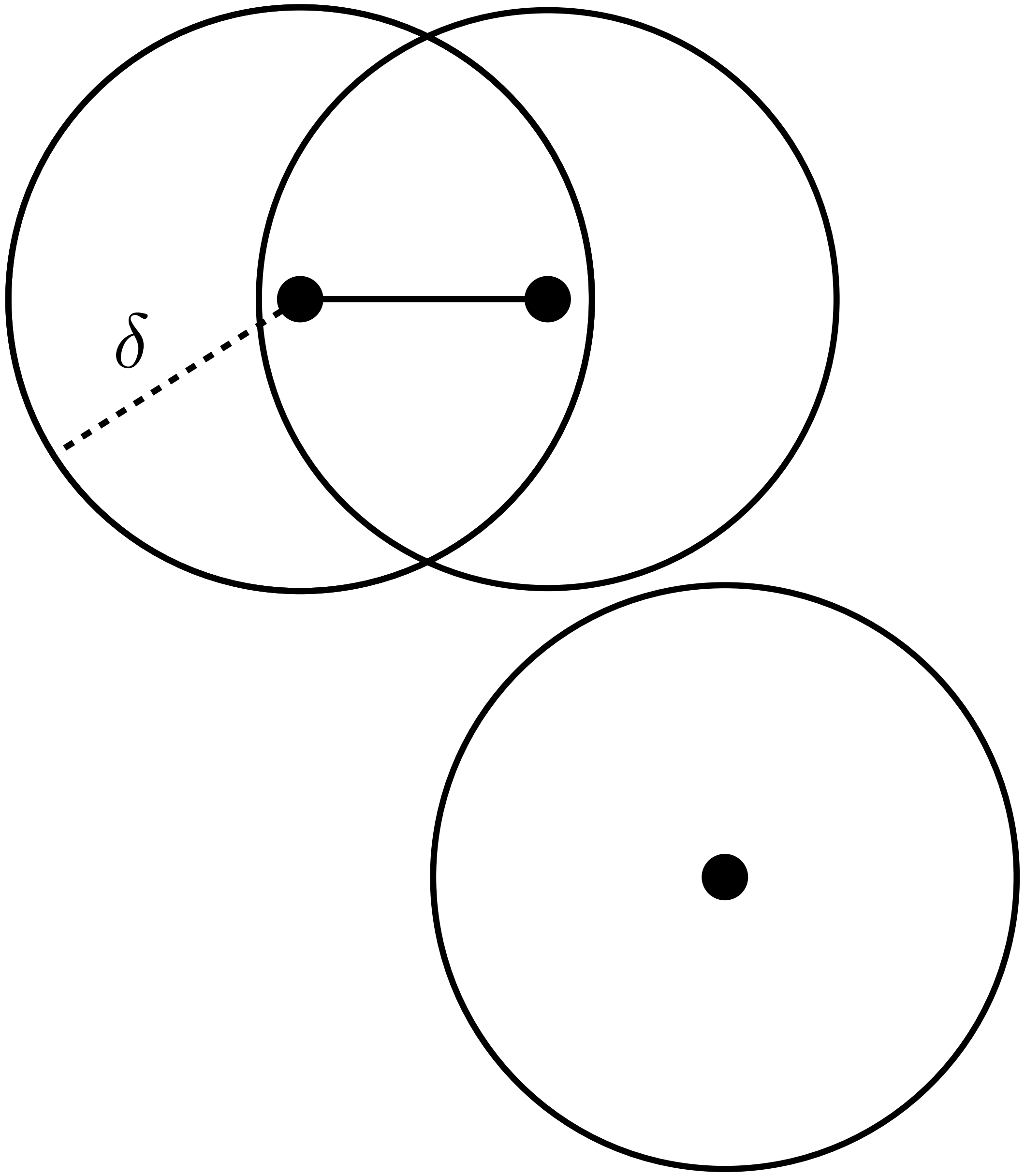
Estimating Connectivity

Vietoris-Rips Graph



Estimating Connectivity

Vietoris-Rips Graph

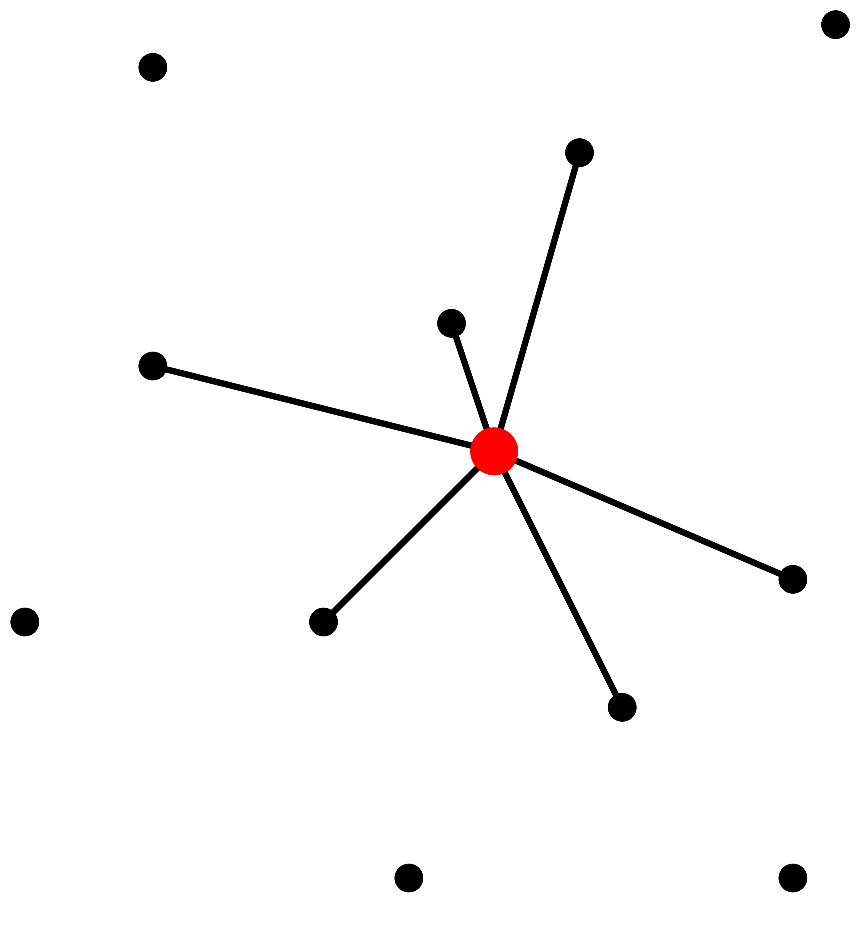


- Simple to compute
- Requires only pairwise distances
- Can be built in any metric space
- Provable reconstruction

Estimating Connectivity

Vietoris-Rips Graph

k-NN Graph

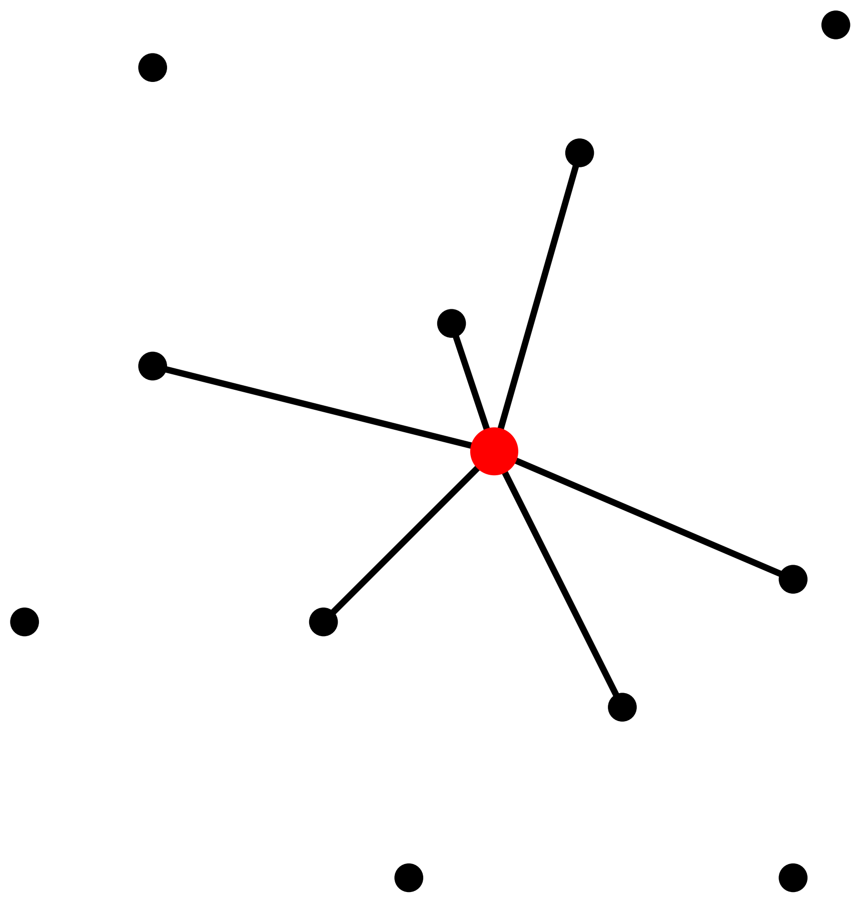


Estimating Connectivity

Vietoris-Rips Graph

k-NN Graph

- Simple to compute
- Requires only pairwise distances
- Can be built in any metric space
- Sparse

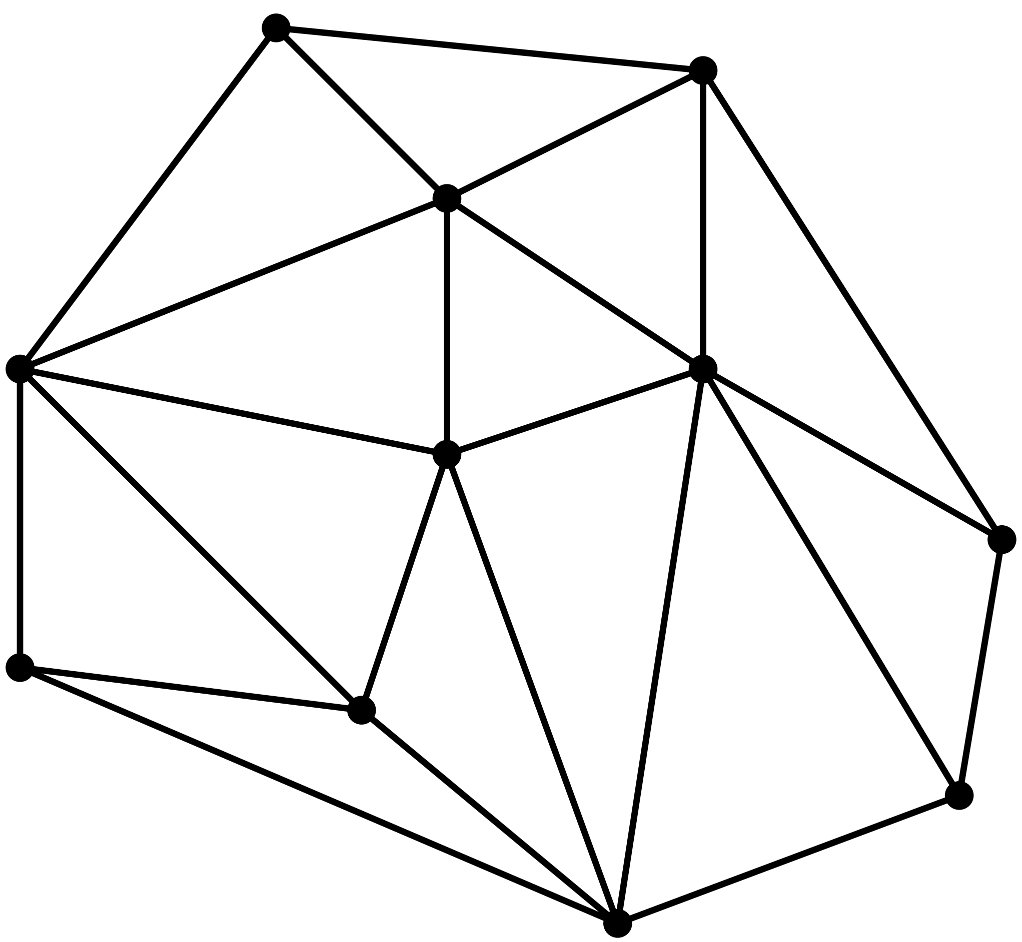


Estimating Connectivity

Vietoris-Rips Graph

k-NN Graph

Delaunay Graph



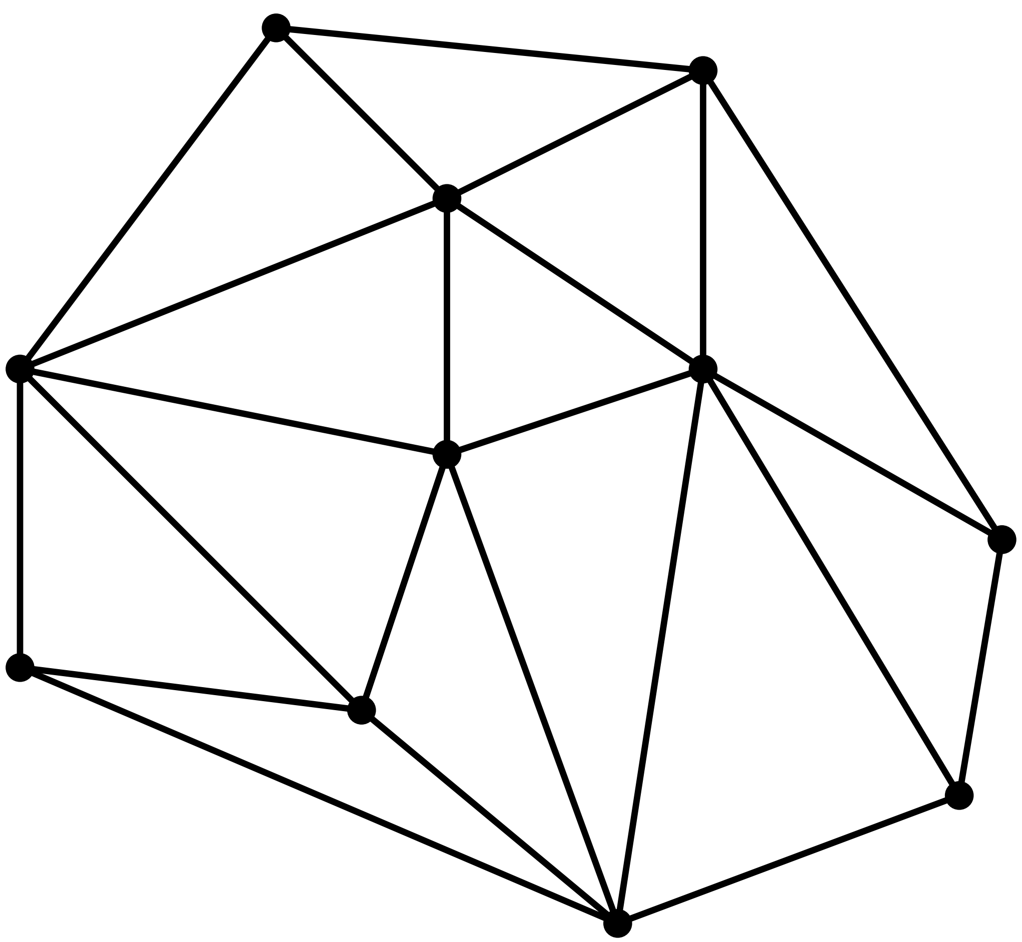
Estimating Connectivity

Vietoris-Rips Graph

k-NN Graph

Delaunay Graph

- Parameter-free
- Sparse
- Fast computation in low-dimensional Euclidean space



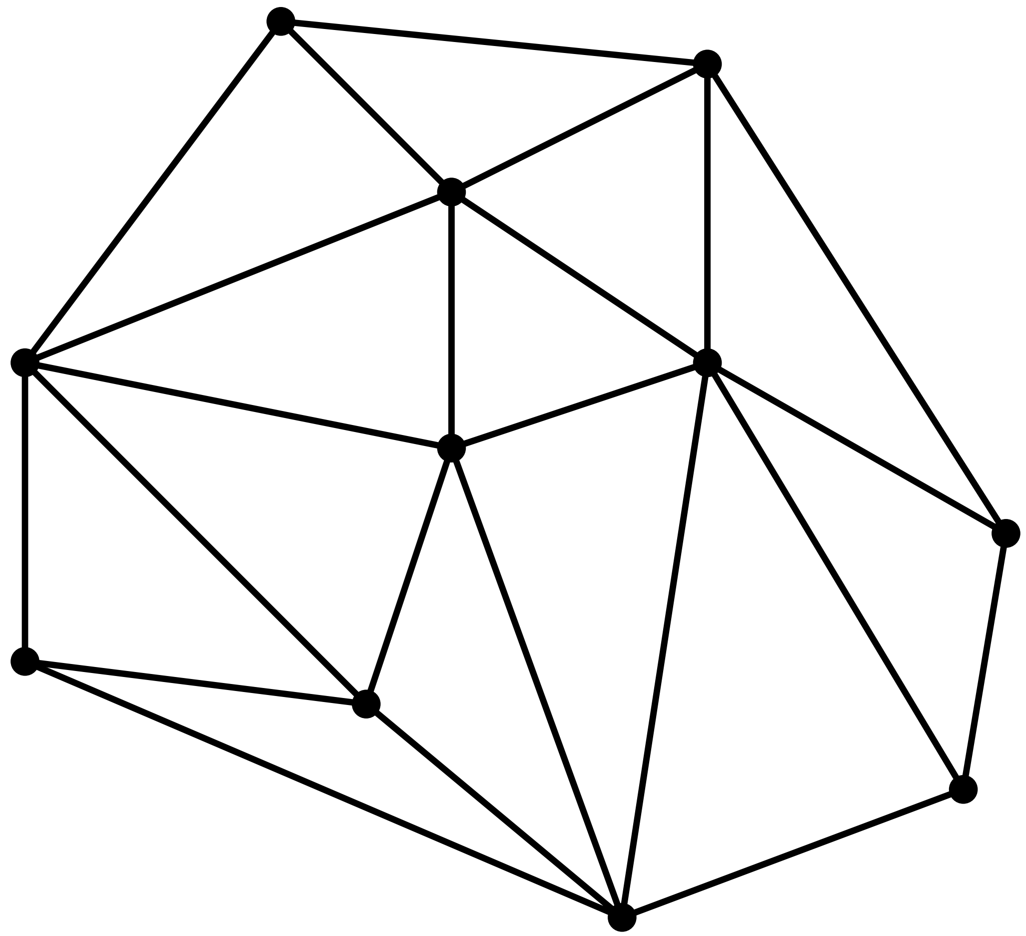
Estimating Connectivity

Vietoris-Rips Graph

k-NN Graph

Delaunay Graph

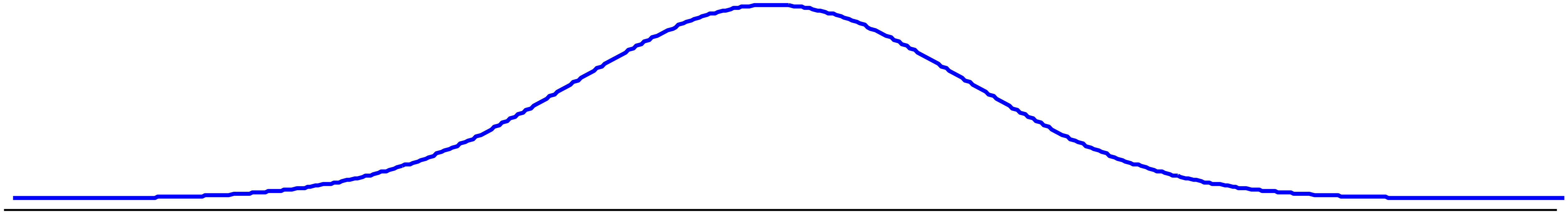
- Parameter-free
- Sparse
- Fast computation in low-dimensional Euclidean space



Sampling

- Whole space is **not** uniformly sampled

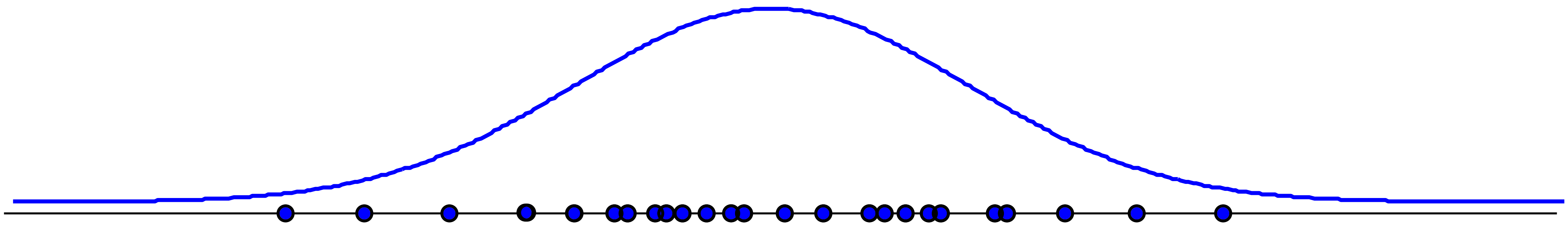
Approximation depends on $c\delta$



Sampling

- Whole space is **not** uniformly sampled

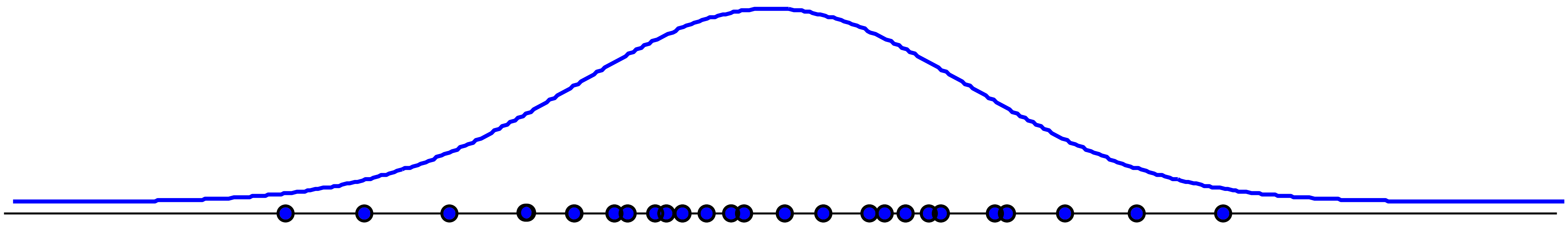
Approximation depends on $c\delta$



Sampling

- Whole space is **not** uniformly sampled

Approximation depends on $c\delta$

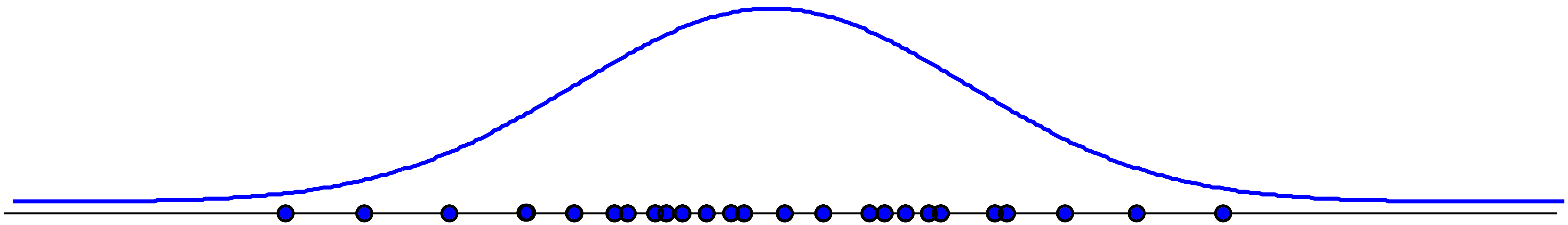


- Approximation result holds in well-sampled regions w.h.p.

Sampling

- Whole space is **not** uniformly sampled

Approximation depends on $c\delta$



- Approximation result holds in well-sampled regions w.h.p.
- Rips parameter $\delta =$ spatial scale
  - Trade-off

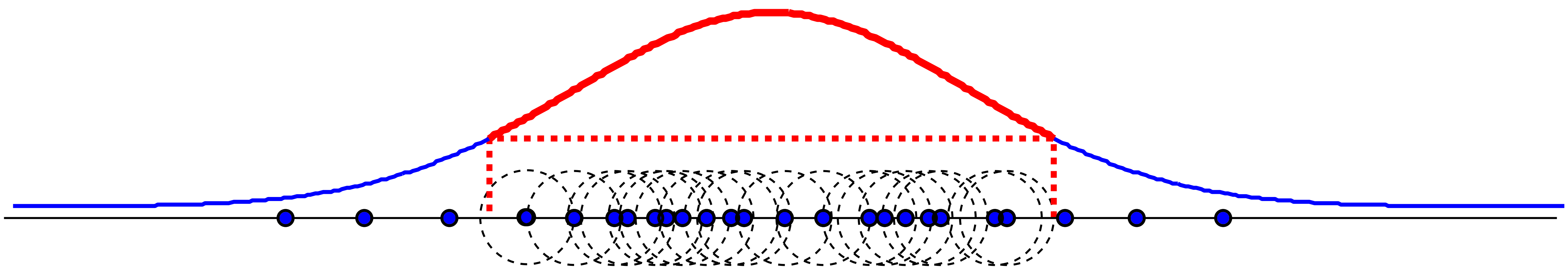
Small $\delta =$ good approximation

Large $\delta =$ holds over a larger part of the space

Sampling

- Whole space is **not** uniformly sampled

Approximation depends on $c\delta$



- Approximation result holds in well-sampled regions w.h.p.
- Rips parameter $\delta =$ spatial scale
  - Trade-off

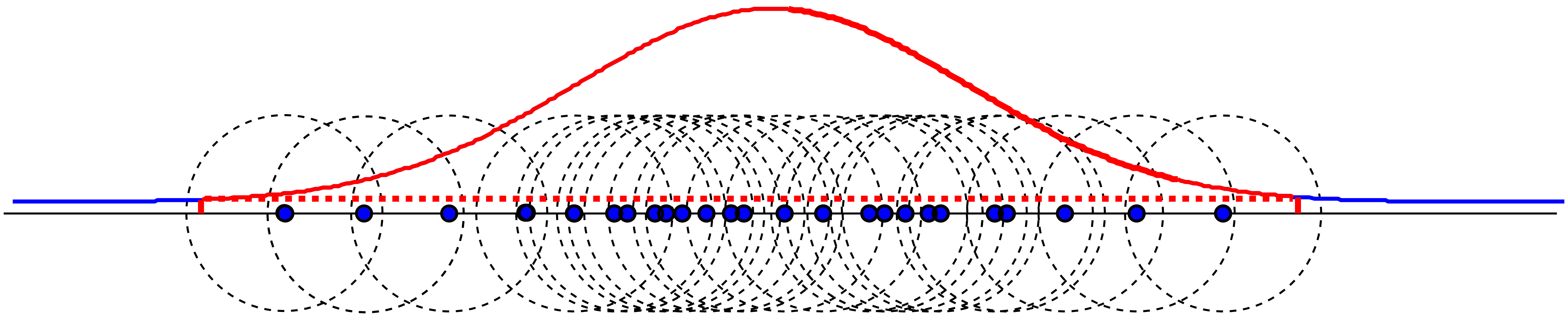
Small $\delta =$ good approximation

Large $\delta =$ holds over a larger part of the space

Sampling

- Whole space is **not** uniformly sampled

Approximation depends on $c\delta$



- Approximation result holds in well-sampled regions w.h.p.

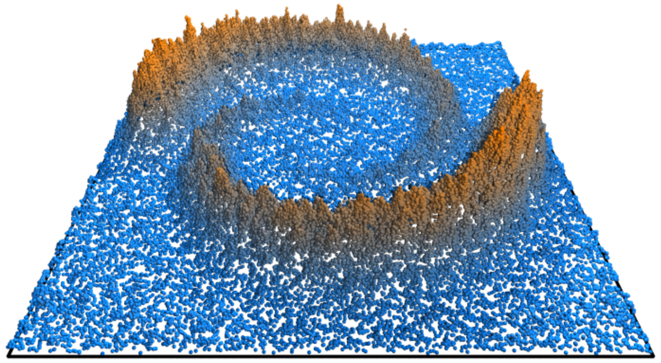
- Rips parameter δ = spatial scale

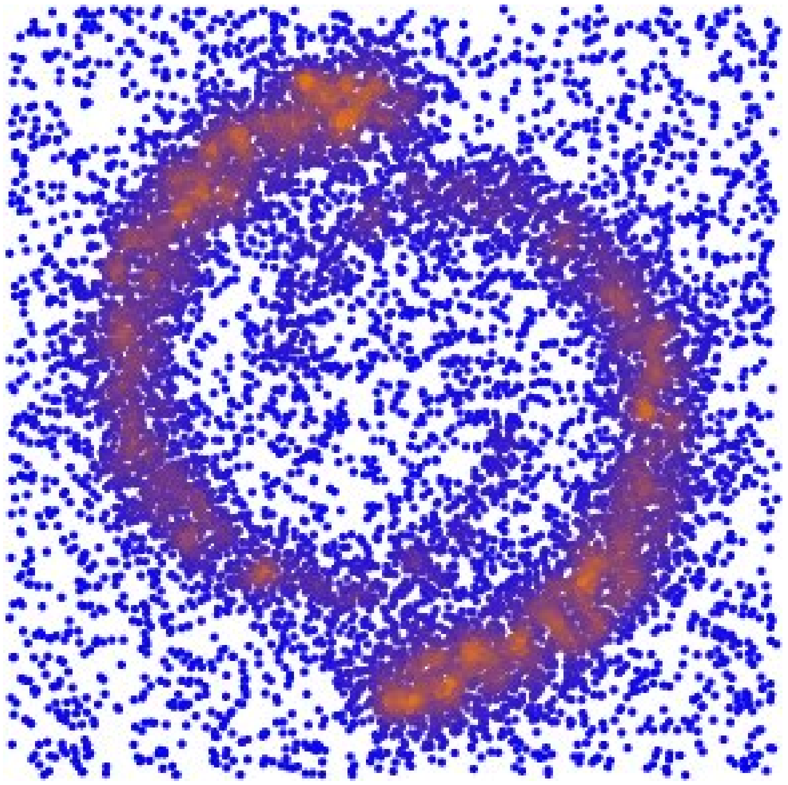
  - Trade-off

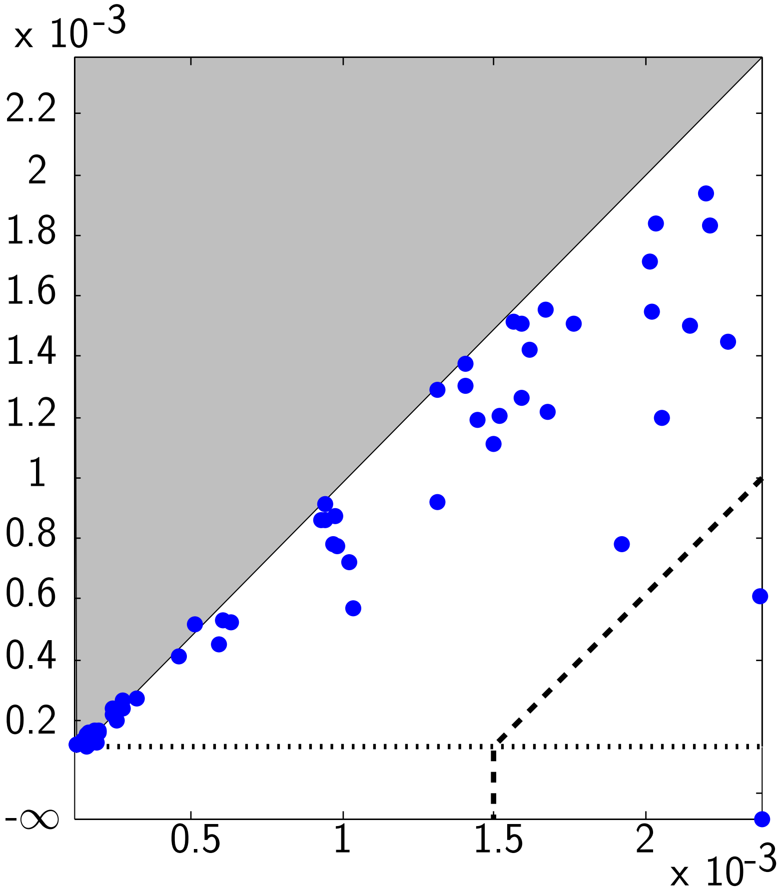
Small δ = good approximation

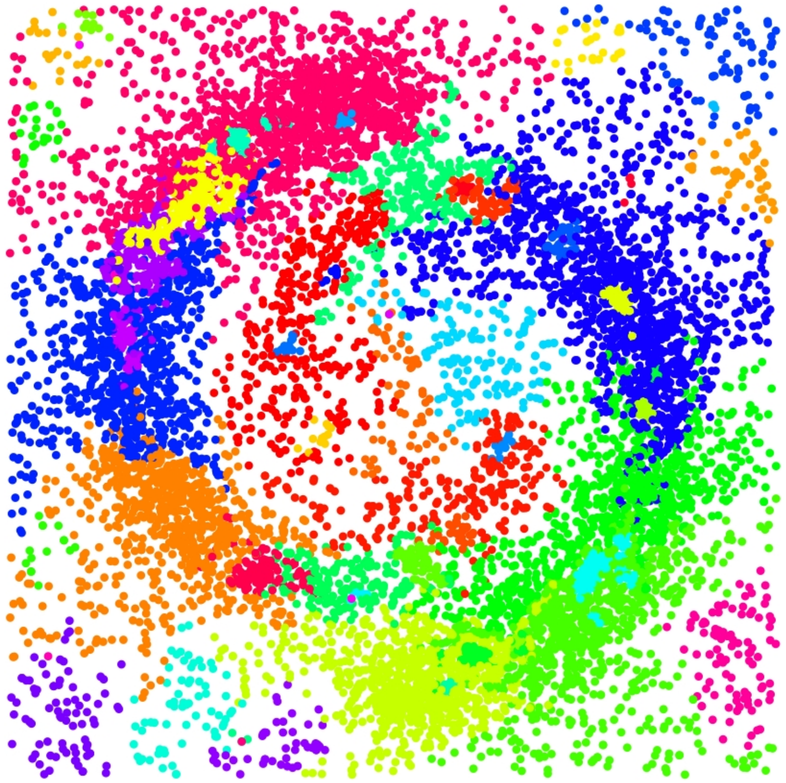
Large δ = holds over a larger part of the space

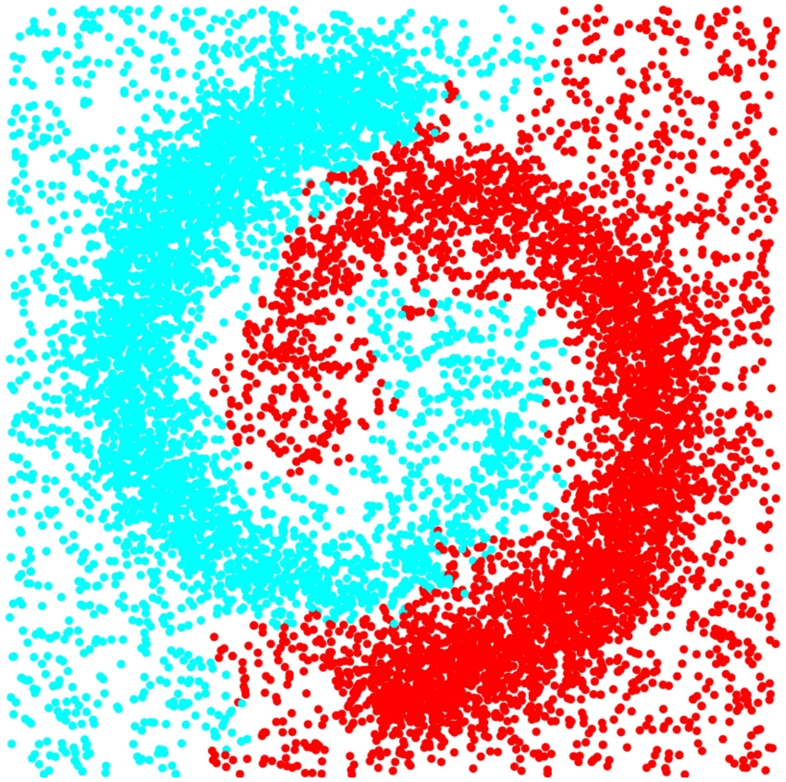
Spirals



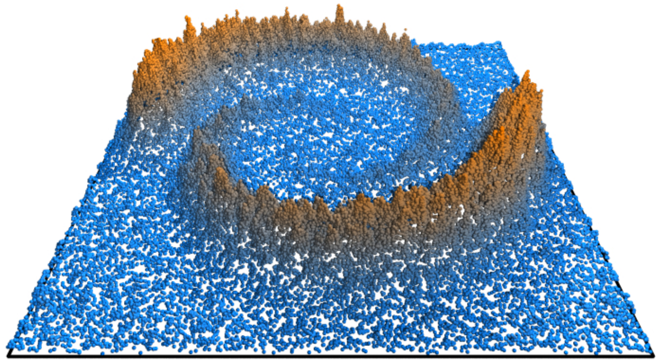


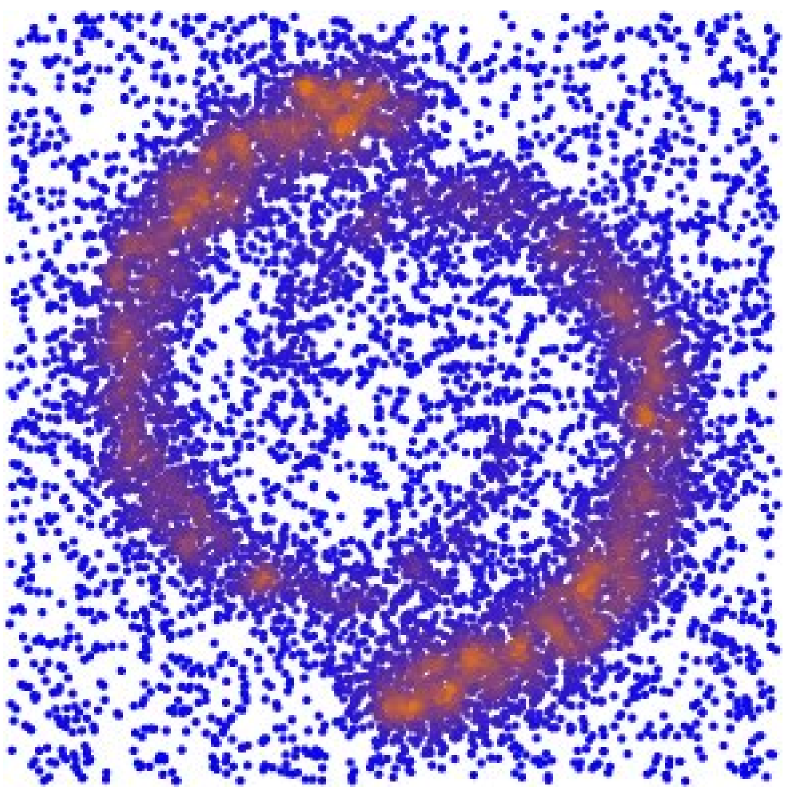


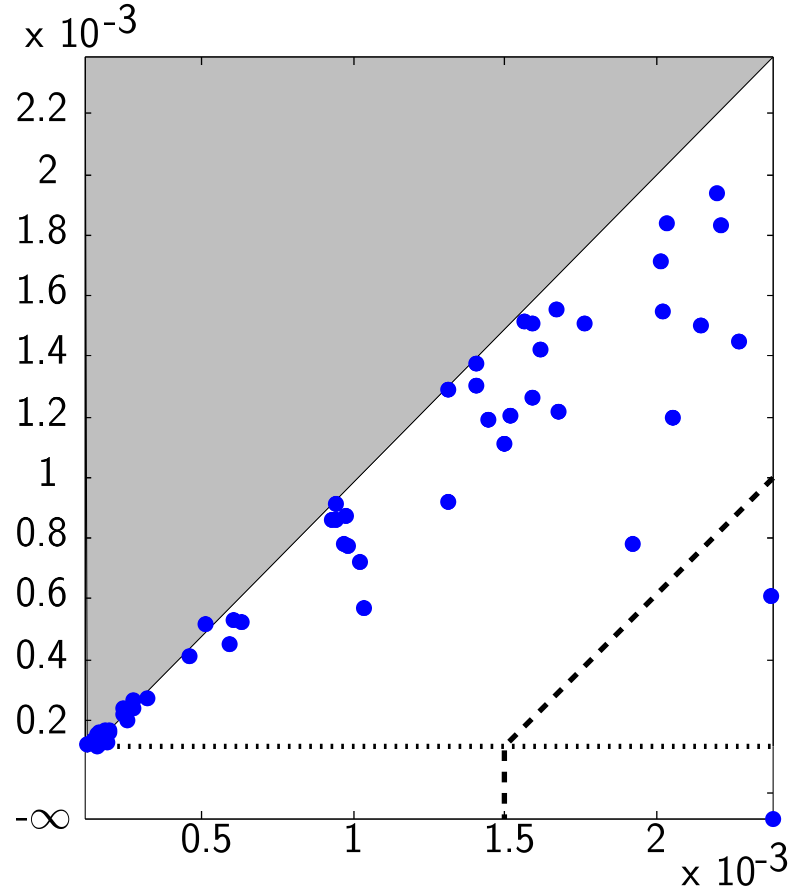


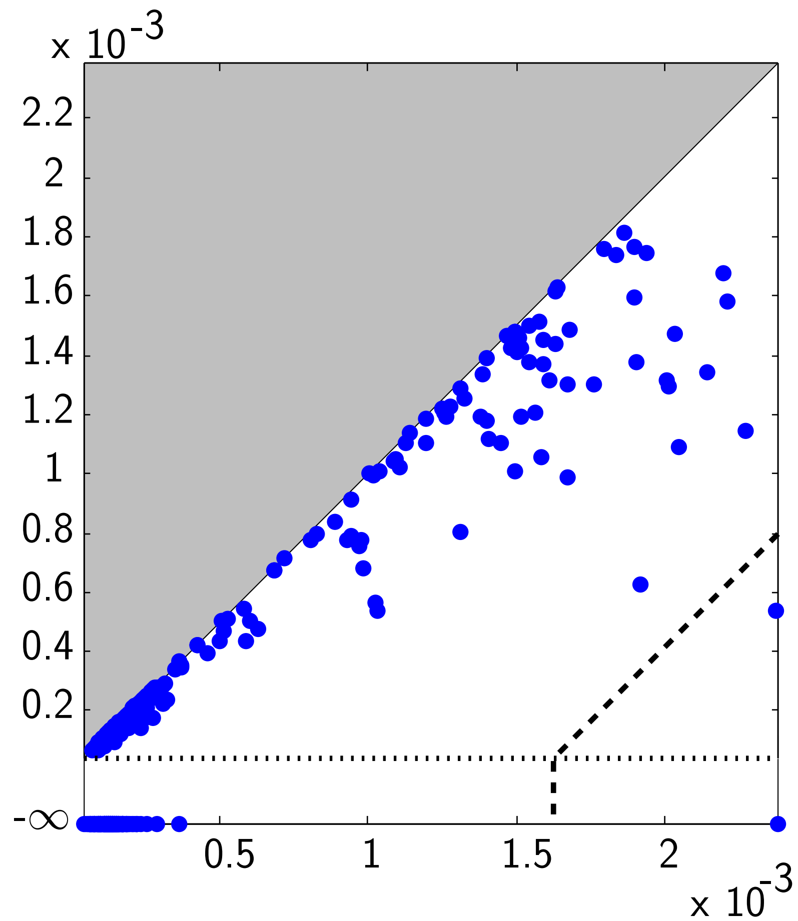


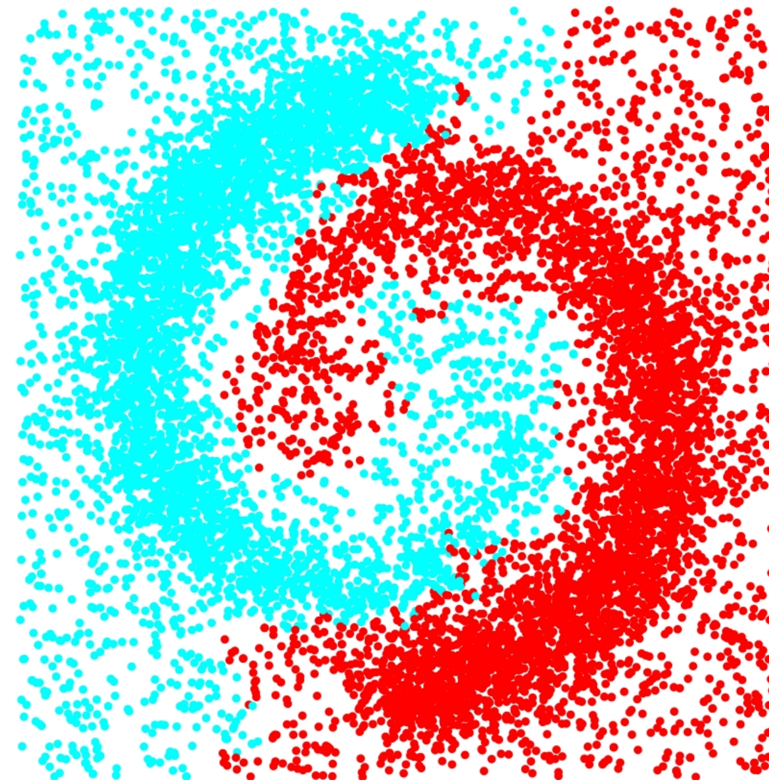
Spirals

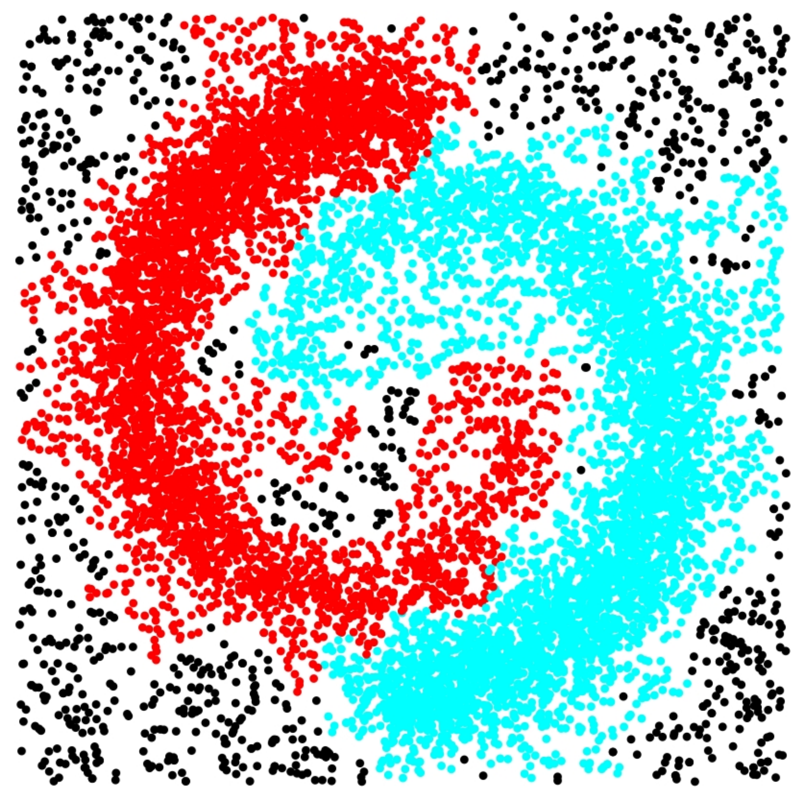




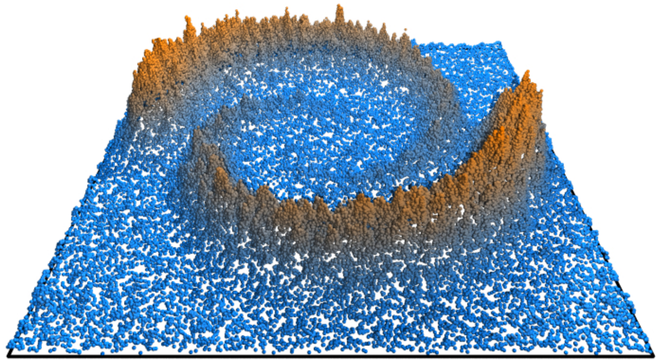

x 10^-3
2.2
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
-∞
0.5
1
1.5
2
x 10^-3

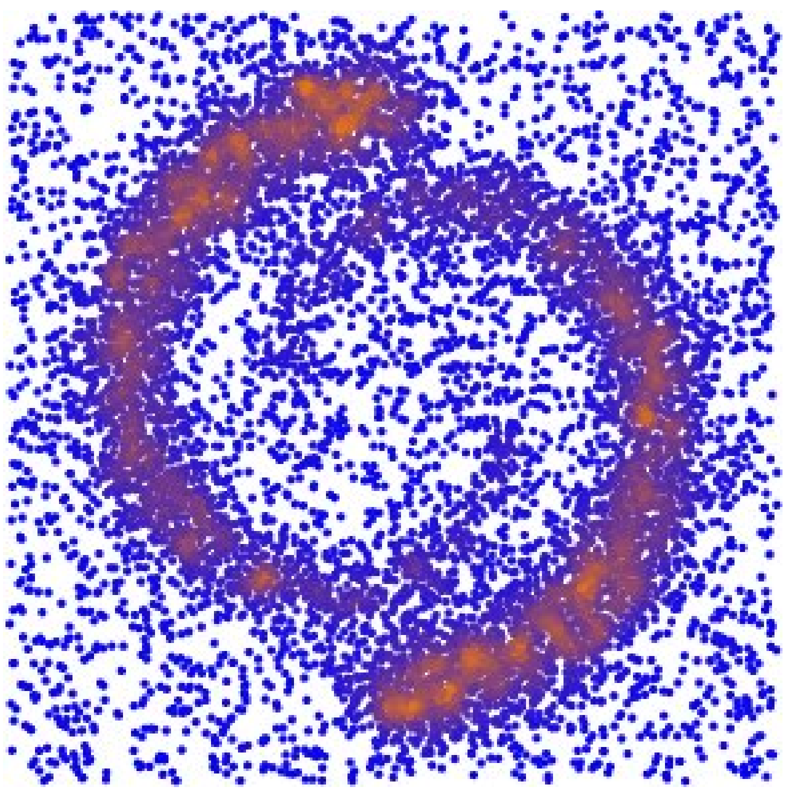

x 10^-3
2.2
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
-∞
0.5
1
1.5
2
x 10^-3

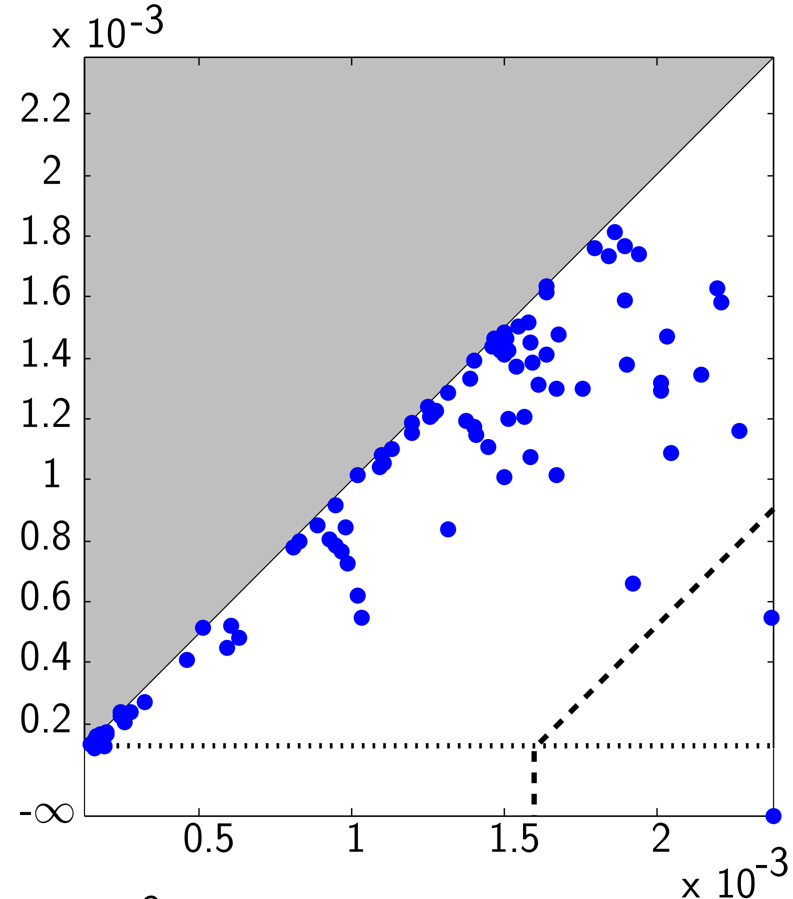


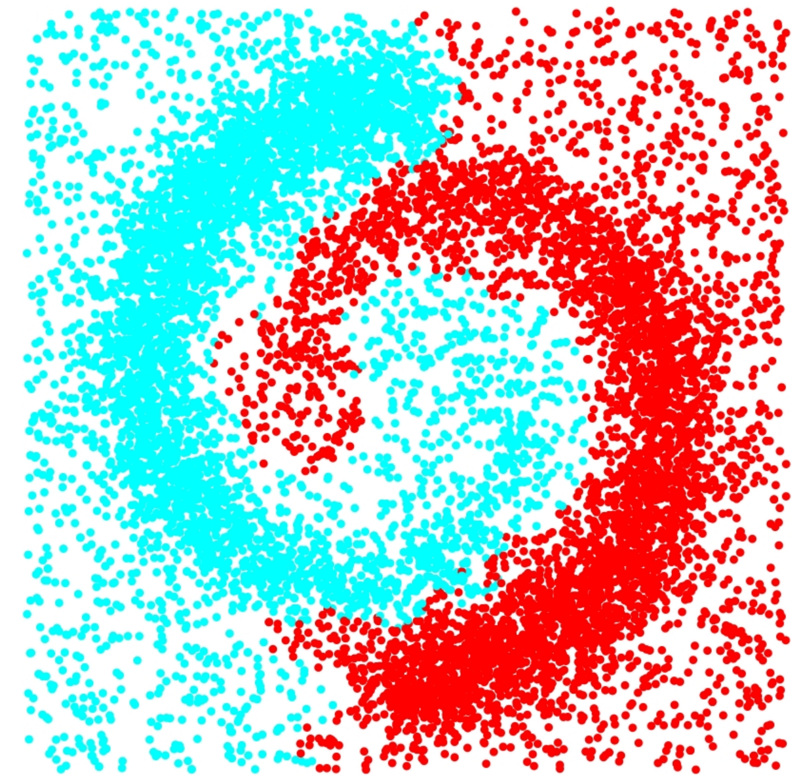


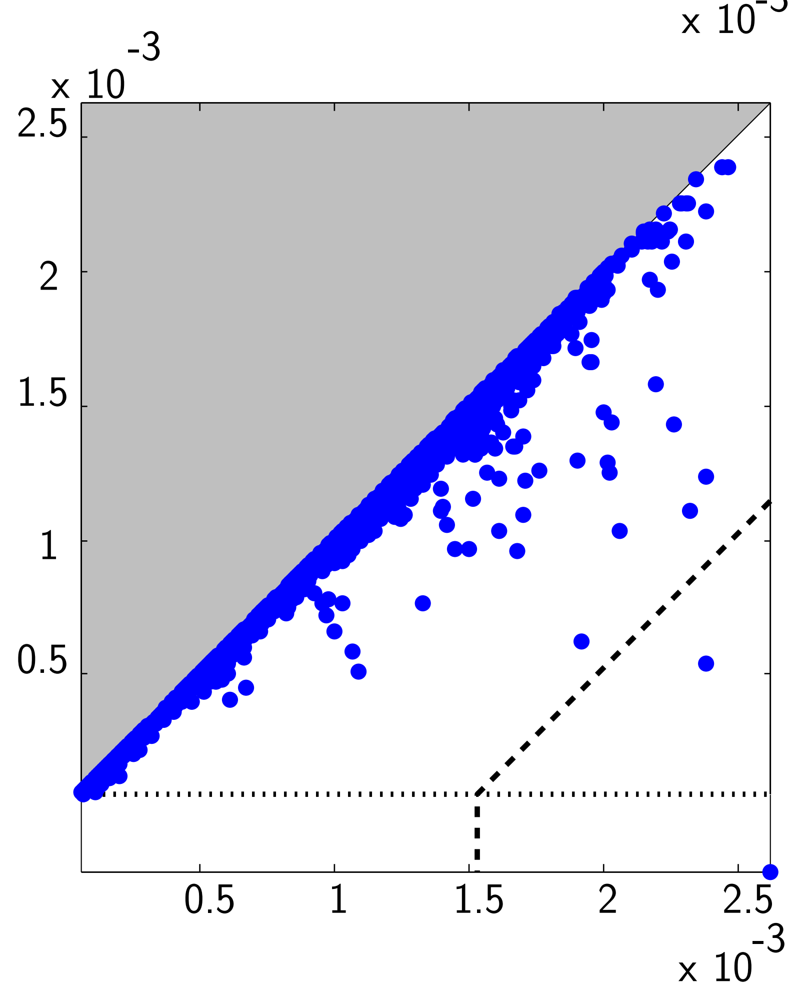
Spirals

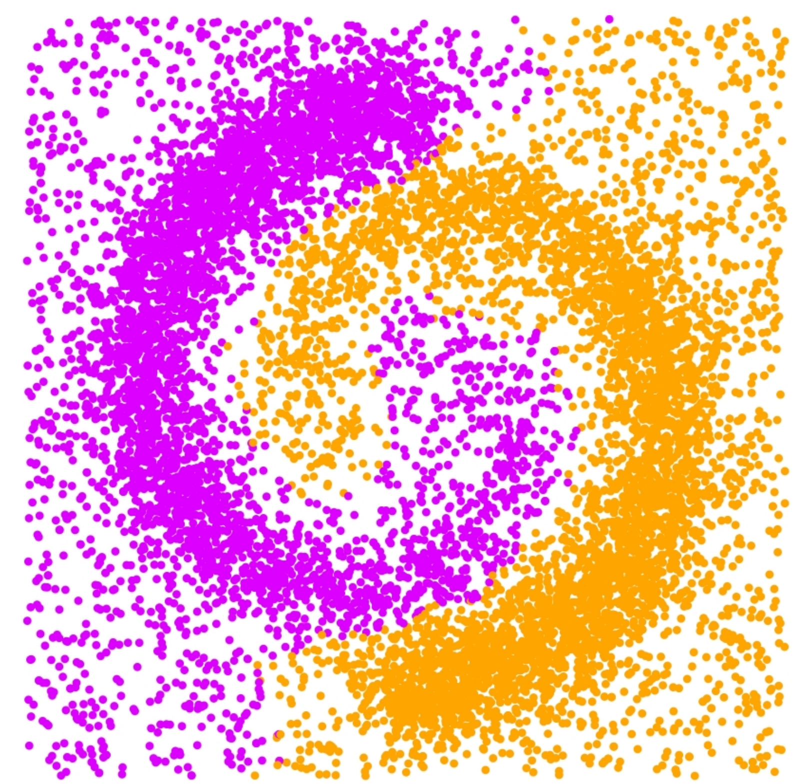





x 10^-3
2.2
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
-∞
0.5
1
1.5
2
x 10^-3

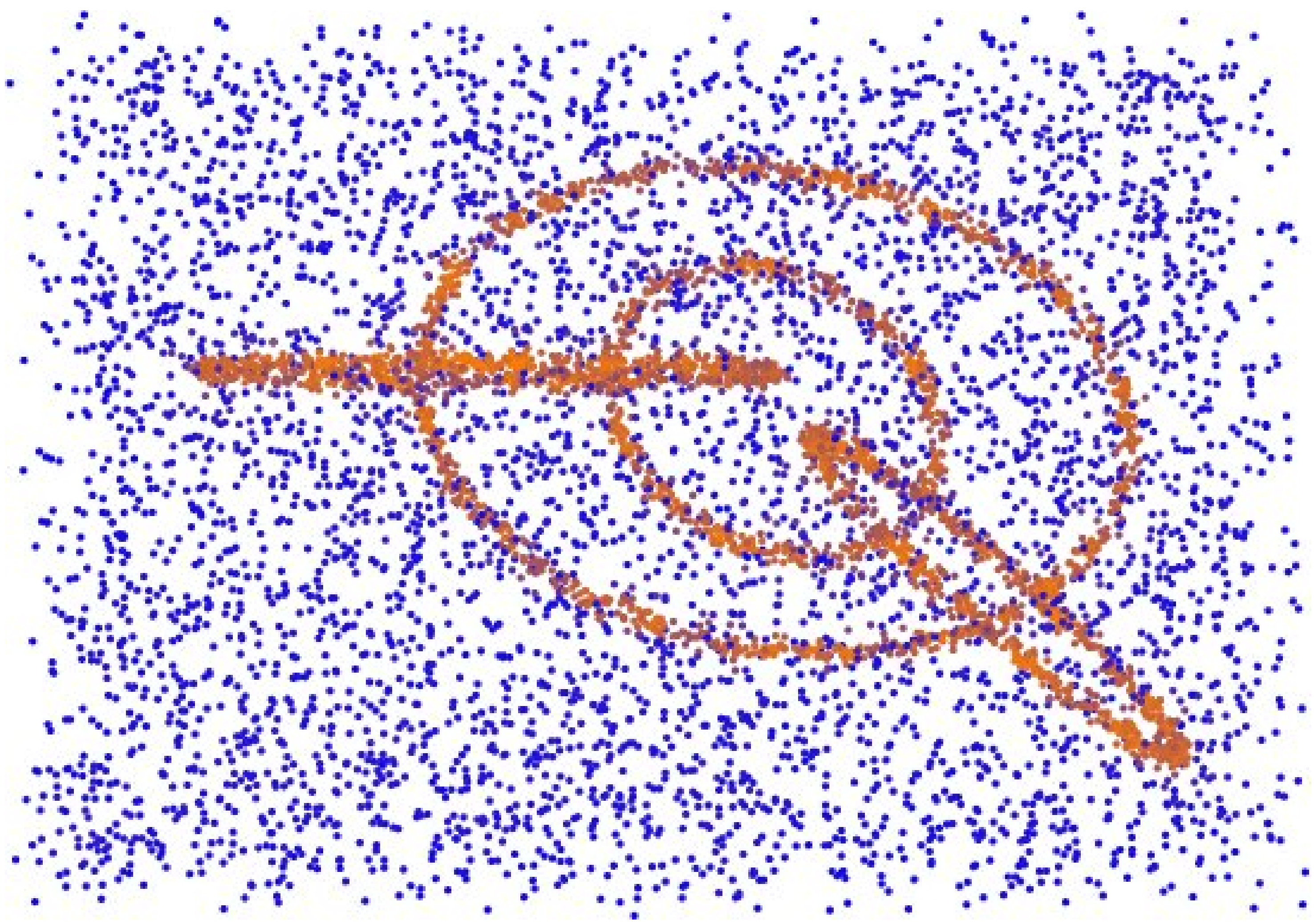



x 10^-3
2.5
2
1.5
1
0.5
0.5
1
1.5
2
2.5
x 10^-3

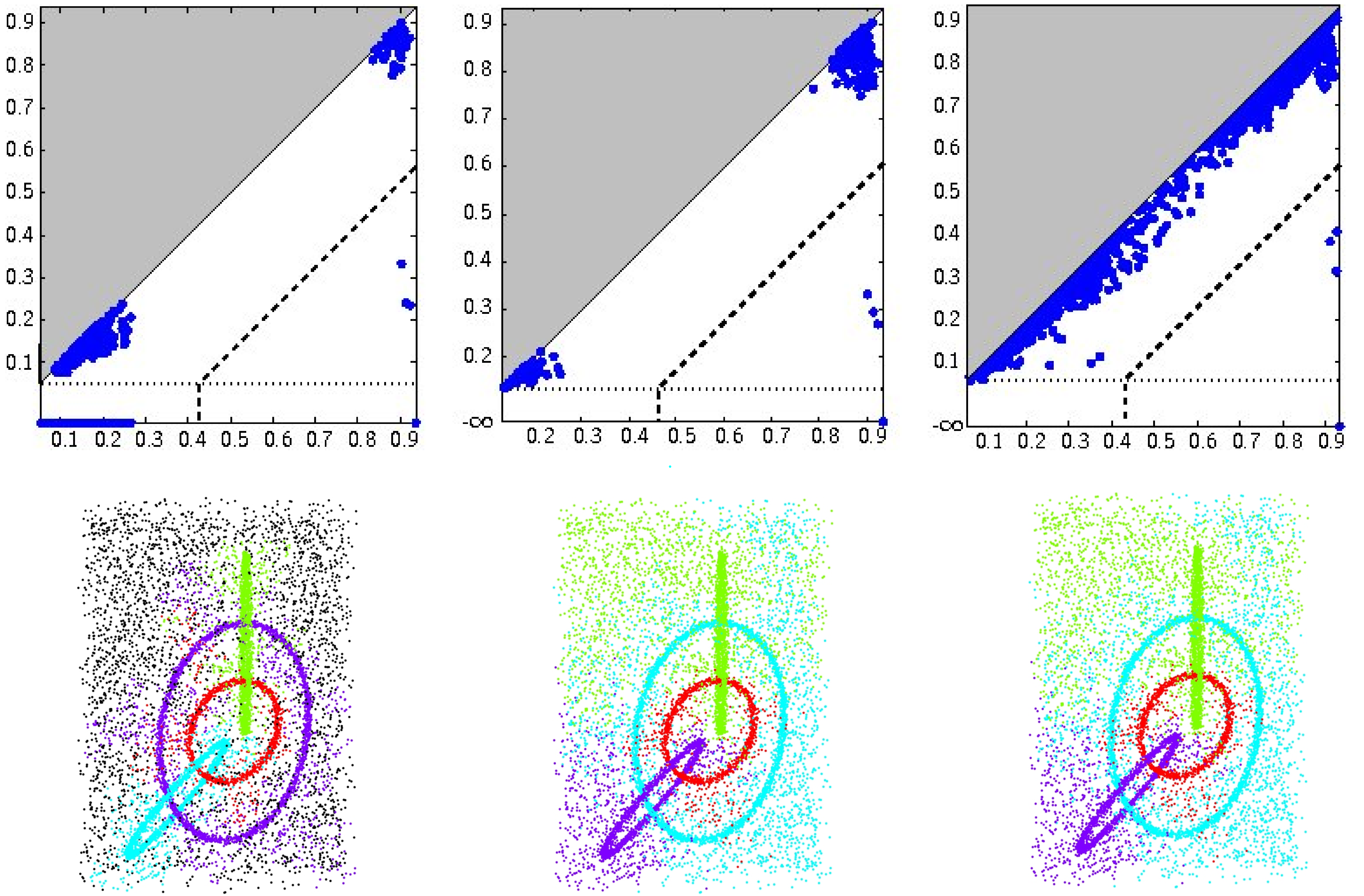


4 Rings

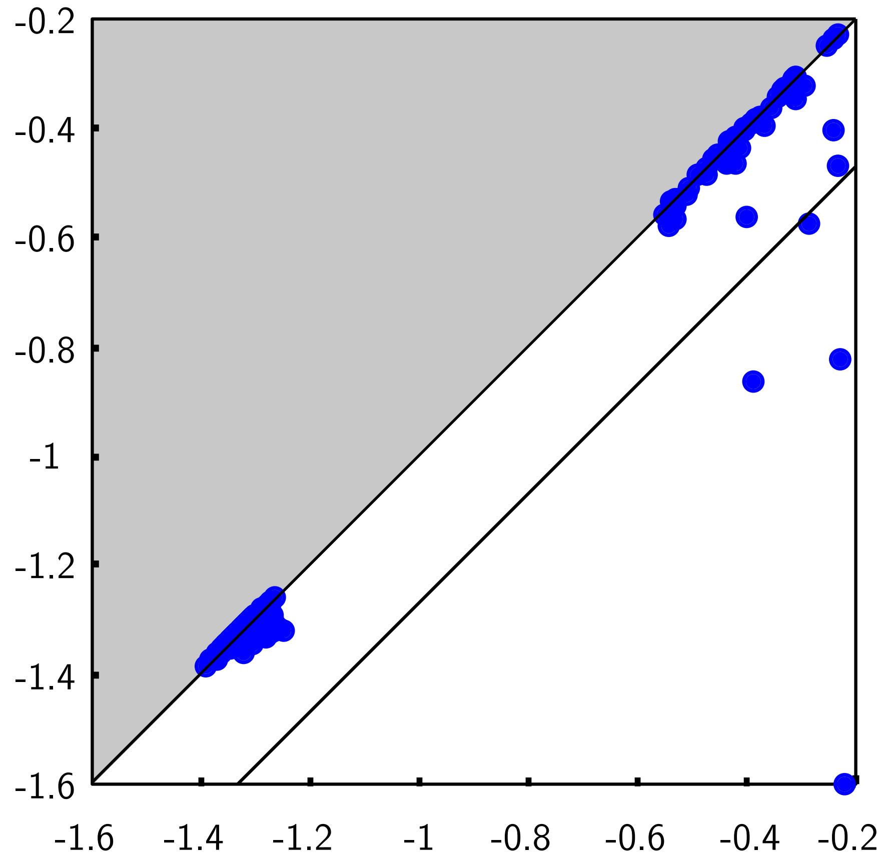
- Interlocking rings in $\mathbb{R}^3$
- 12K points total

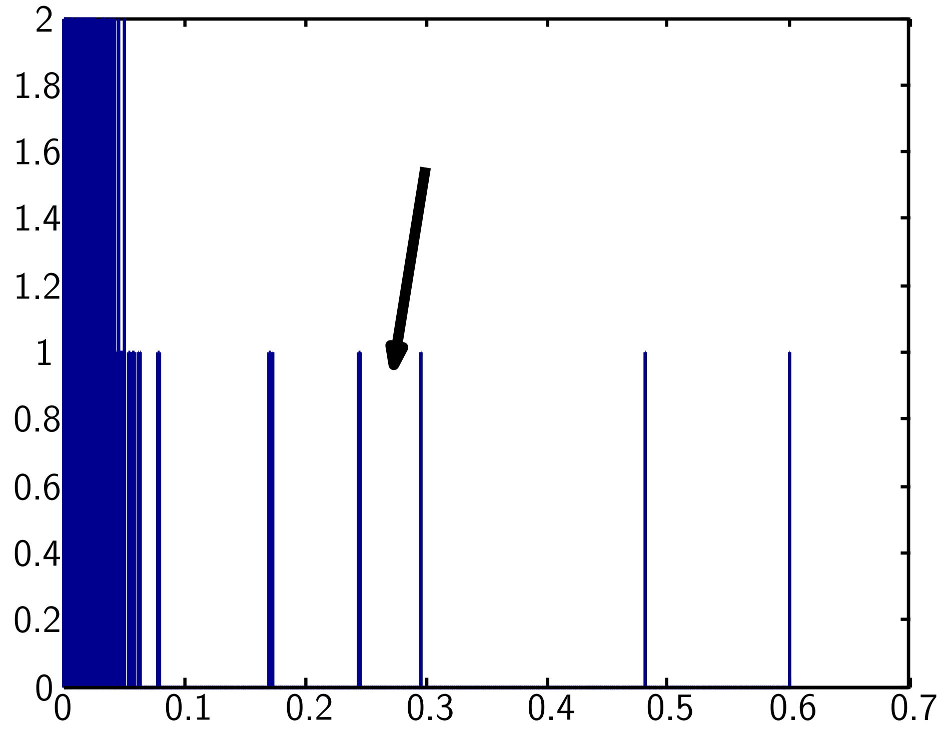


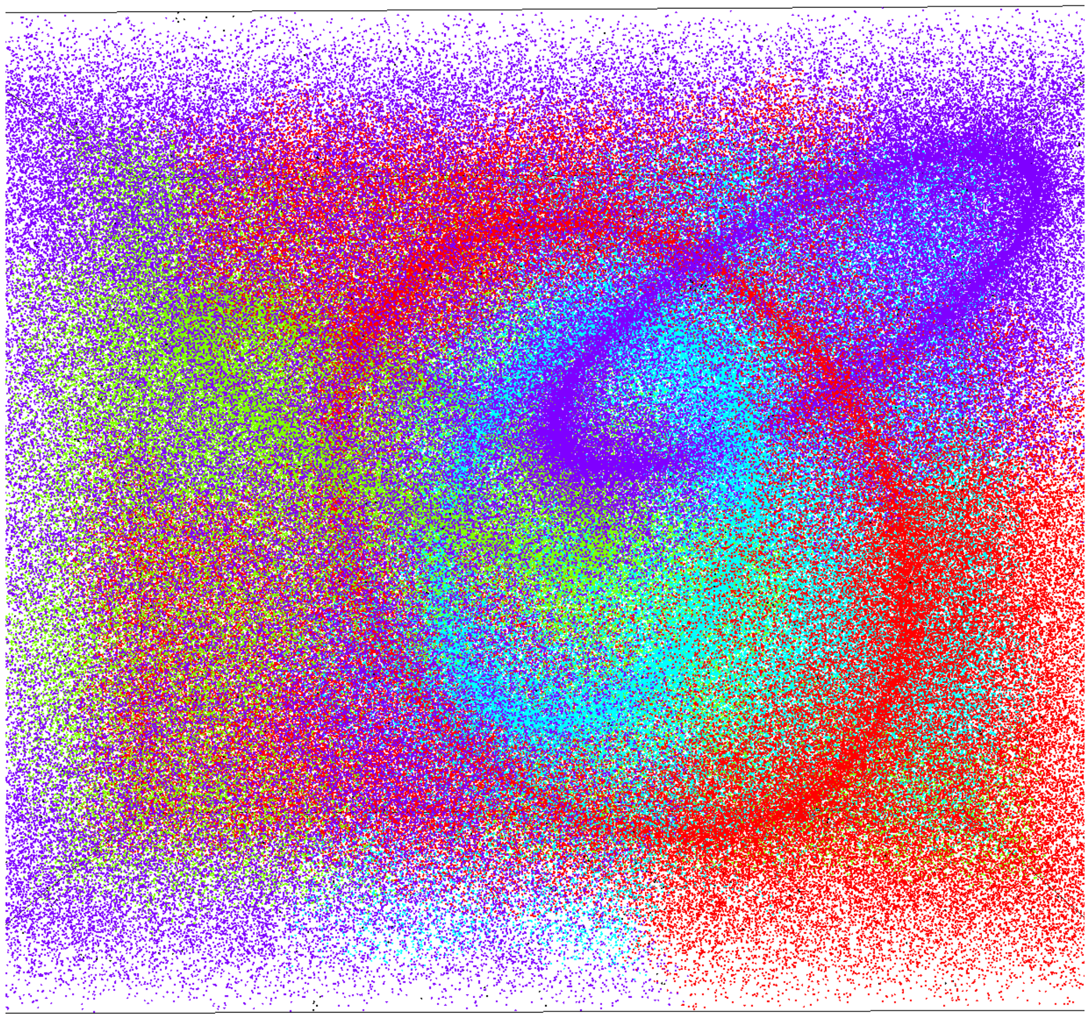
4 Rings



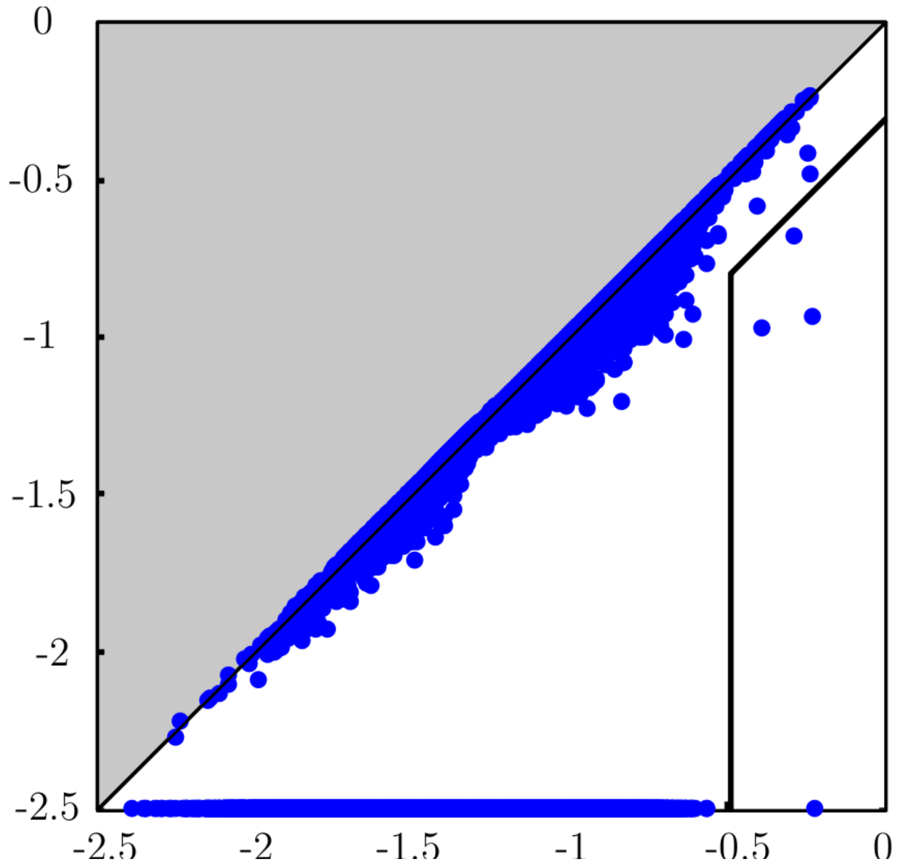
Poorly Sampled 4 Rings

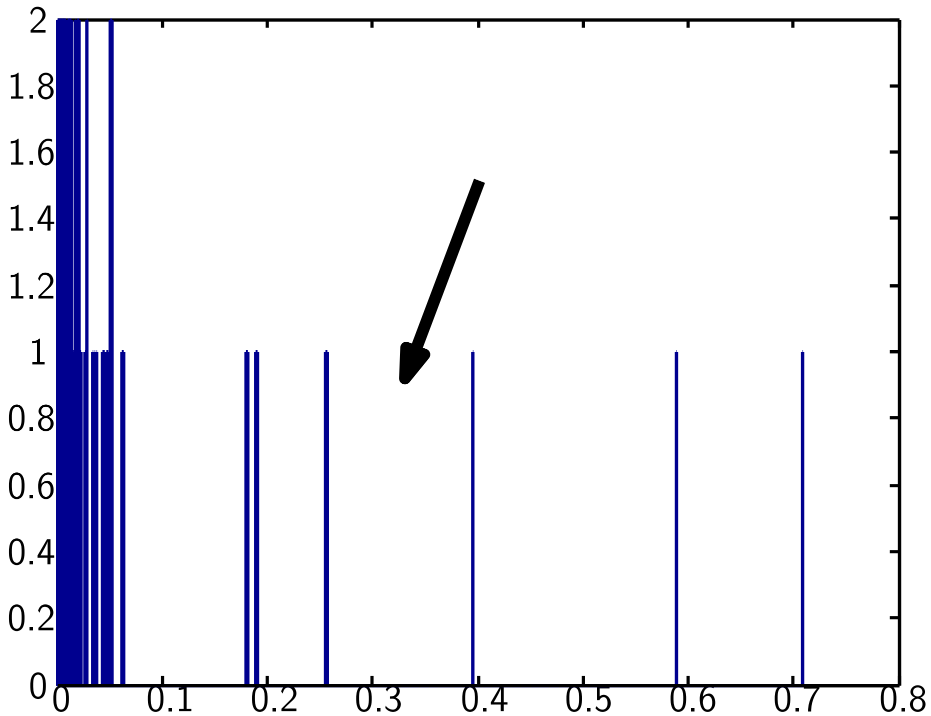


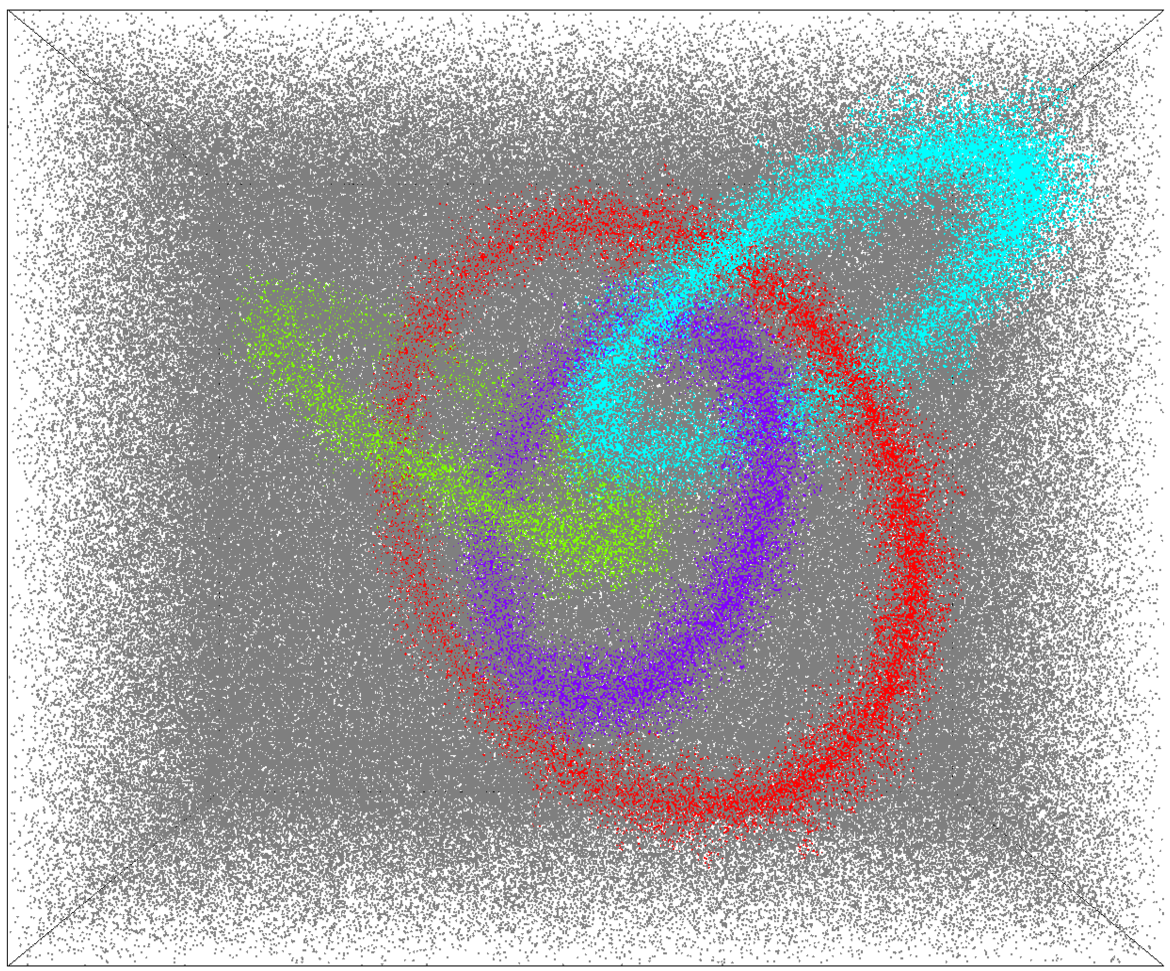




Poorly Sampled 4 Rings

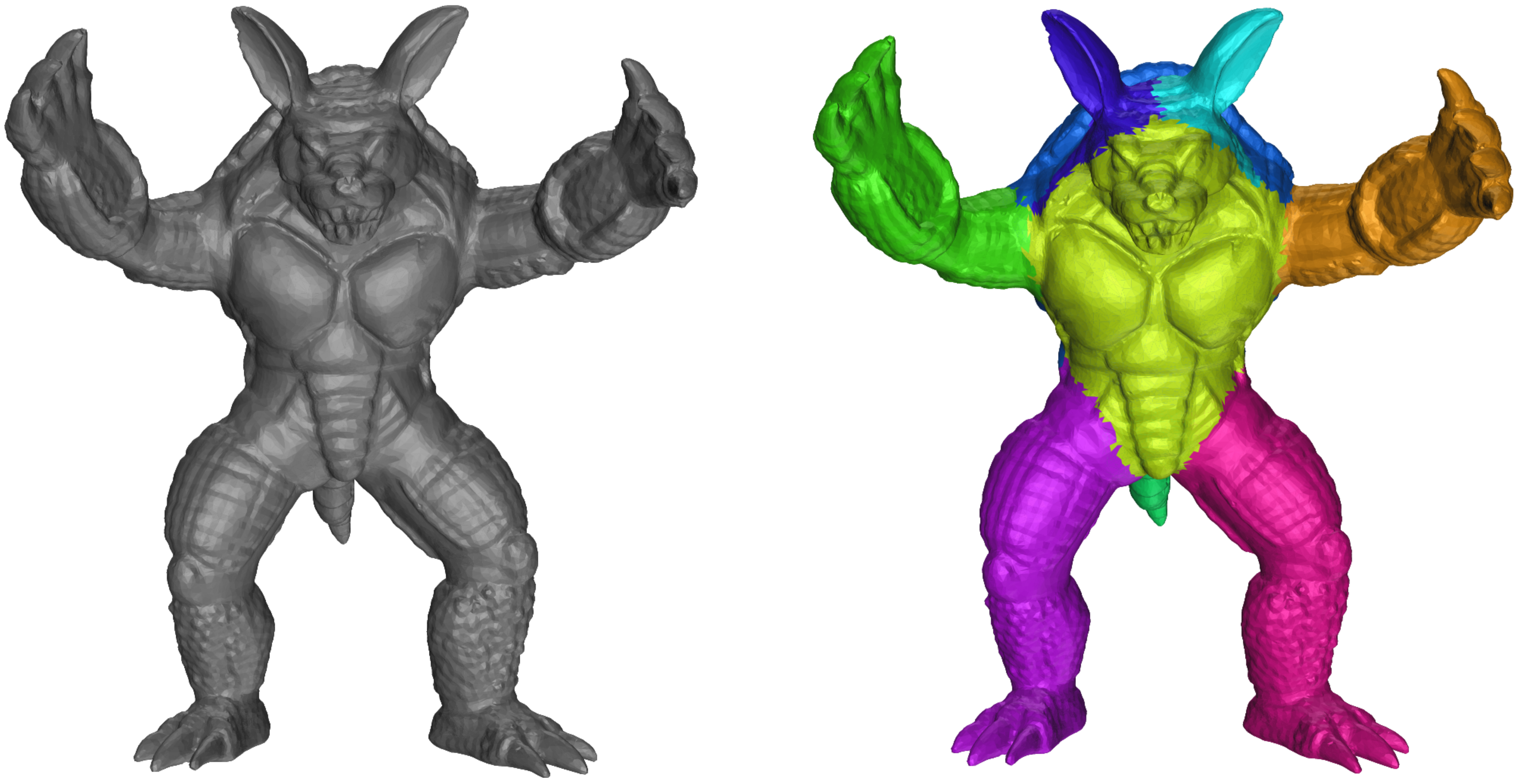






Segmentation

Segment a 3d shape into “meaningful” components



Goal

Which is more “meaningful”?

Goal

Which is more “meaningful”?

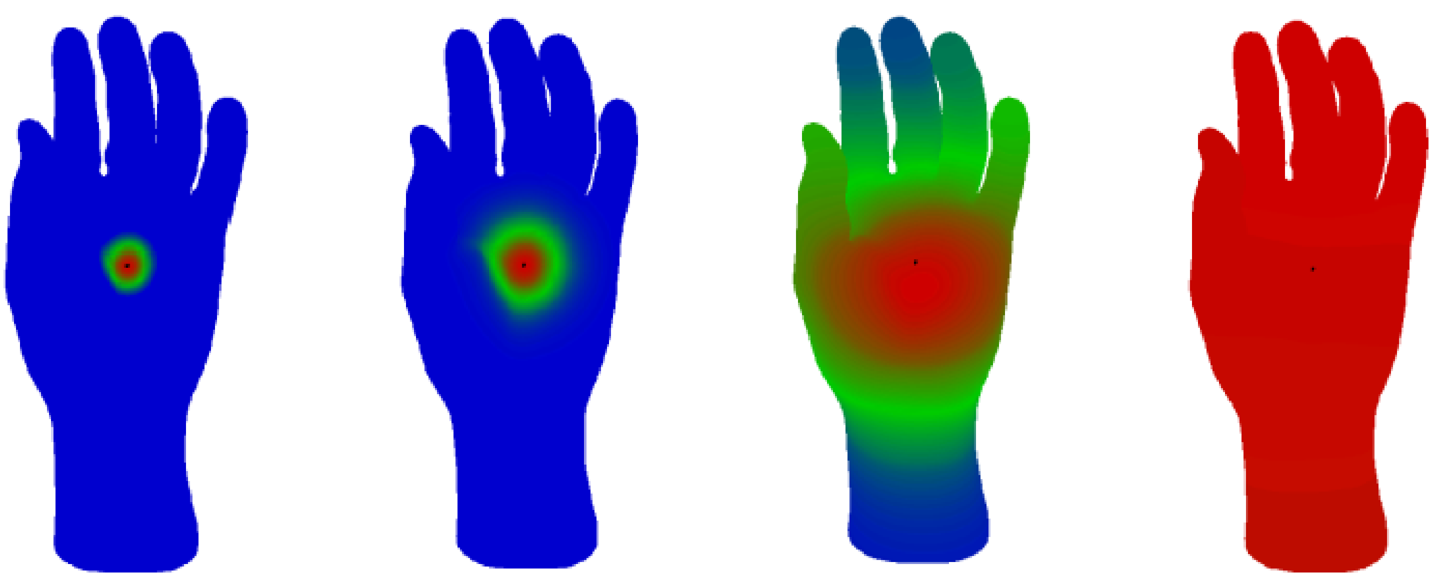
Persistence-based Clustering + Heat Kernal Signature

Heat Kernel Signature

Heat kernel: $k_t(x, y) : \mathcal{M} \times \mathcal{M} \times \mathbb{R}^+ \rightarrow \mathbb{R}^+$

$$f(x,t) = \int_{\mathcal{M}} f(y,0) k_t(x,y) dy$$

$k_t(x, y)$: intuitively, amount of heat transfered from x to y in time t.



$t = 0.001$ $t = 0.02$ $t = 3$ $t = 10$

Heat Kernel Signature

For a point $x \in \mathcal{M}$ let:

$$f(x) = k_t(x, x)$$

Heat Kernel Signature (HKS) *at scale* t.

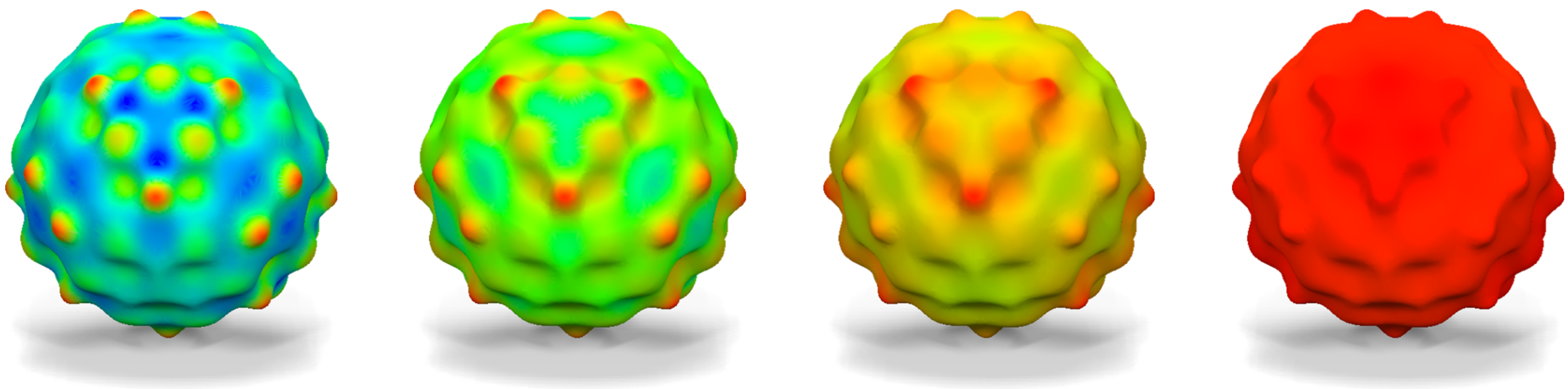
Main properties:

- Isometry-invariant: $k_t(x, x) = k_t(T(x), T(y))$.
- Robust under small perturbations.
  - $k_t(\cdot, \cdot)$ is the p.d.f. of Brownian motion on the manifold. Weighted average over all possible paths.
- Multi-scale
  - As t increases heat diffuses to larger neighborhoods.

Jian Sun, Maks Ovsjanikov, and Leonidas Guibas, *A Concise and Provably Informative Multi-scale Signature Based on Heat Diffusion*, SGP 2009.

Heat Kernel Signature

Can be interpreted as a multi-scale, intrinsic curvature.



$t = 0.004$ $\quad$ $t = 0.008$ $\quad$ $t = 0.02$ $\quad$ $t = 2$

Computing the Heat Kernel Signature

On a compact manifold:

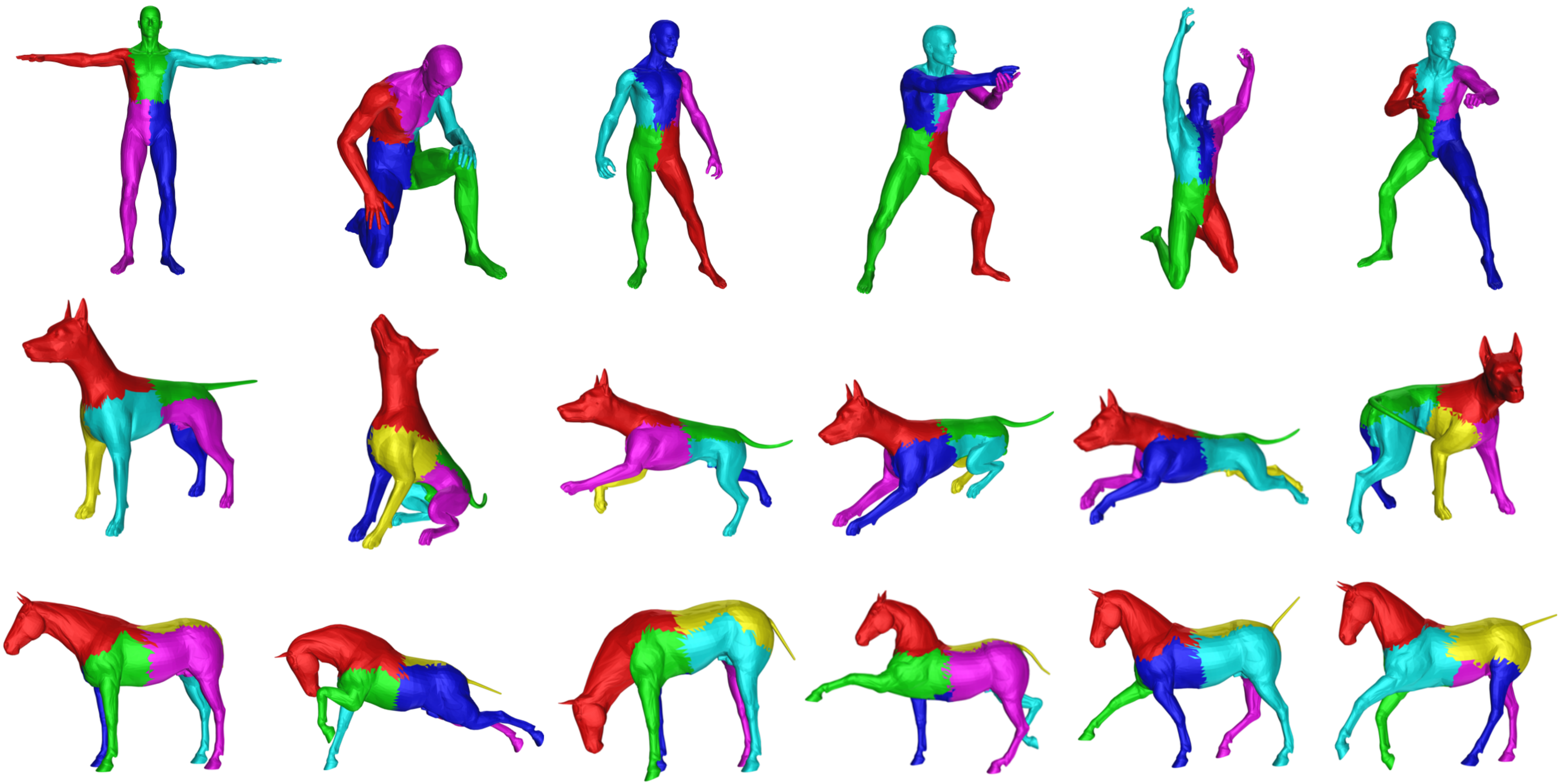
$$k_t(x,x) = \sum_{i=0}^{\infty} e^{-t\lambda_i} \phi_i(x)^2$$

λ_i, ϕ_i: i^{th} eigenvalue/function of the Laplace-Beltrami operator.

Laplace-Beltrami operator on a mesh is a matrix. Use eigenvalues & eigenvectors of the matrix.

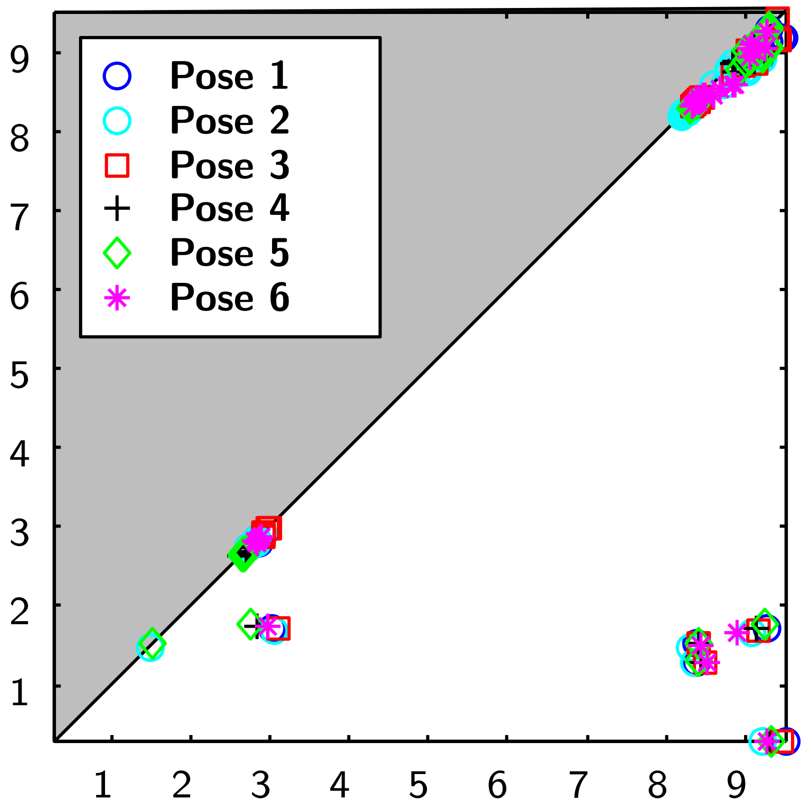
Once eigen-decomposition is computed, can obtain HKS at any scale.

Segmentations

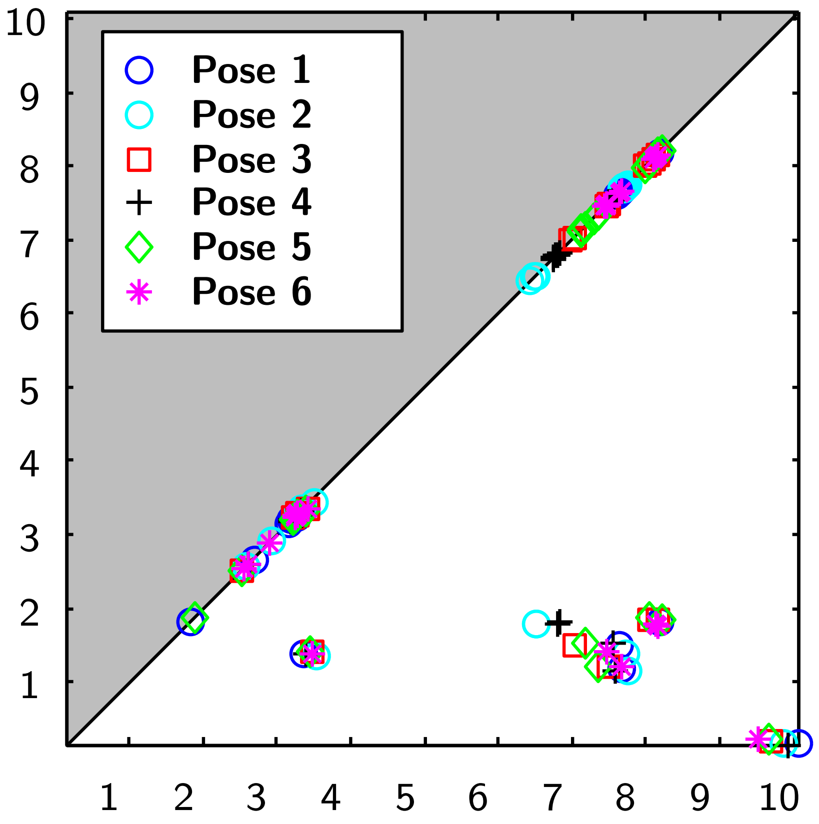


$$t = 0.1$$

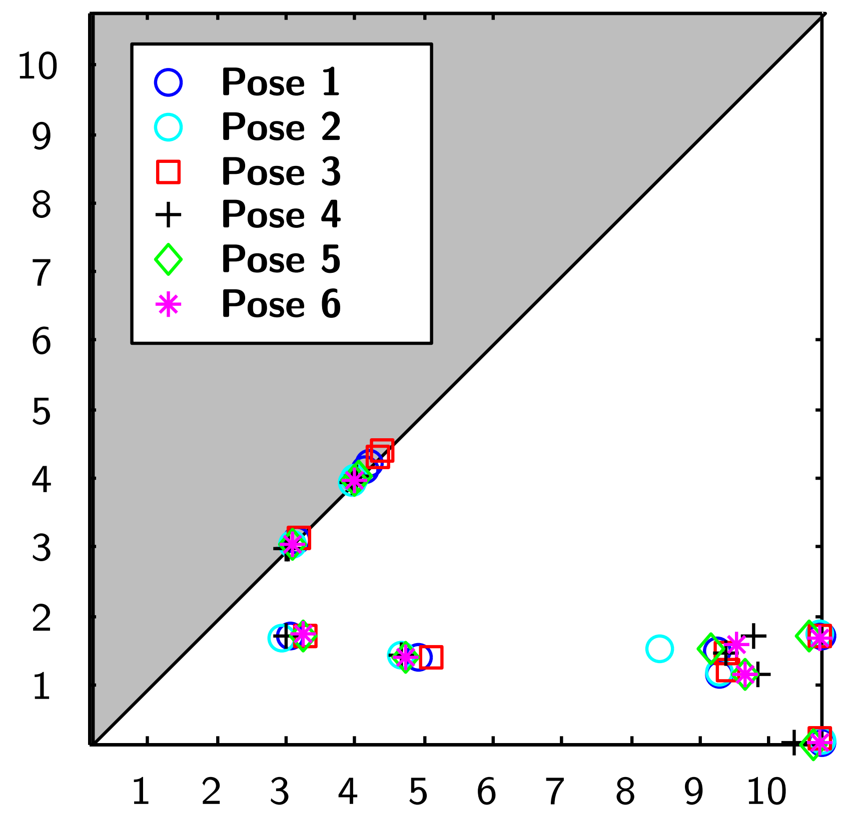
Stable Diagrams



Human

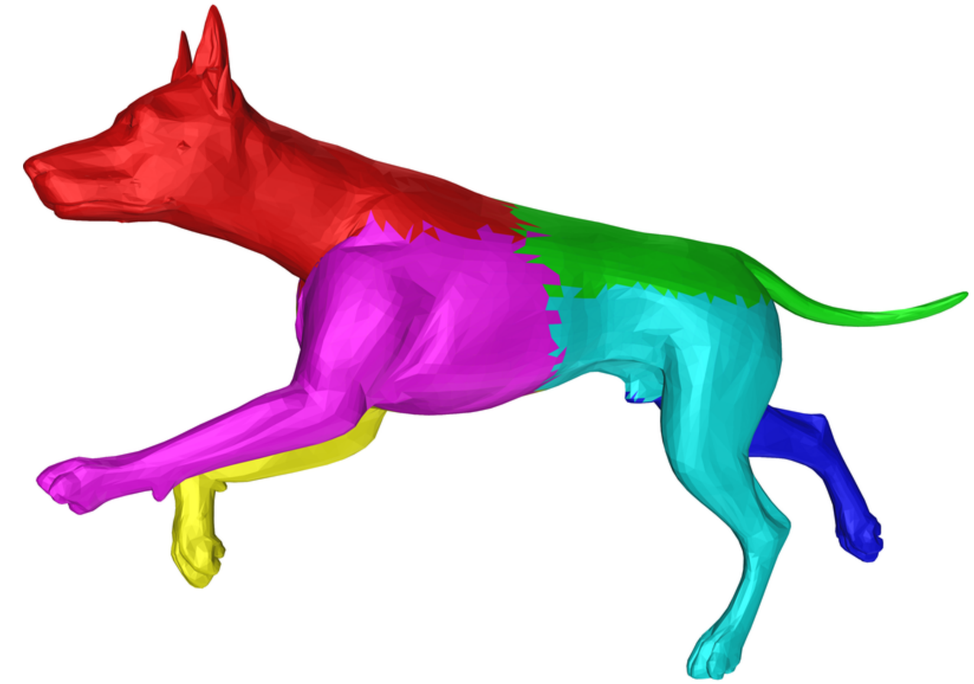


Dog

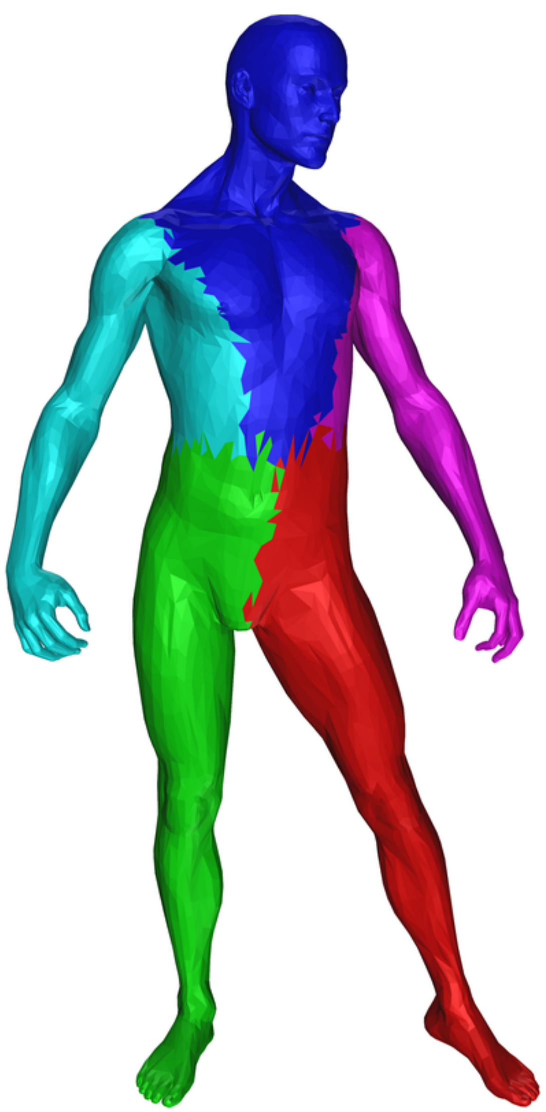


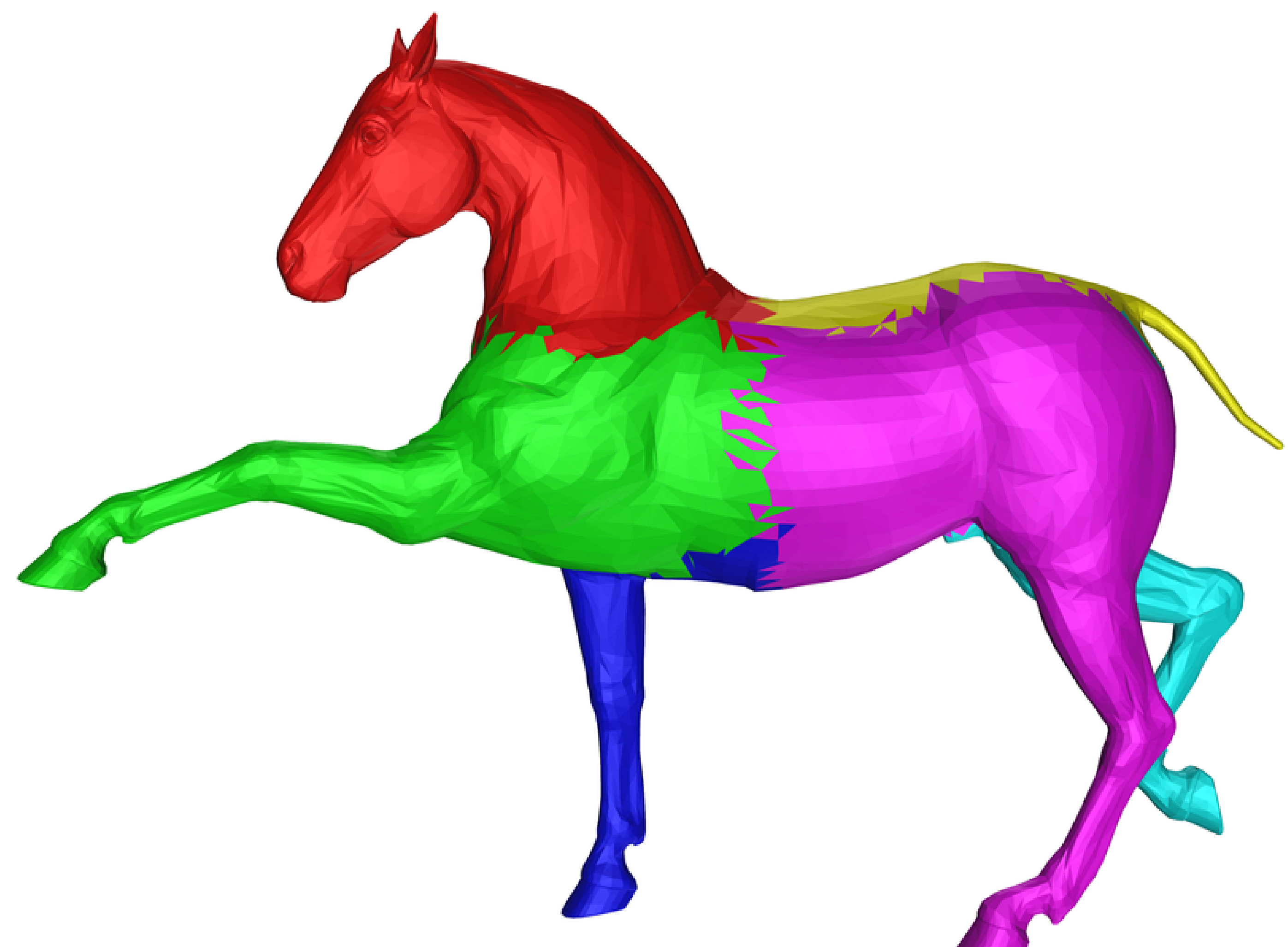
Horse

Regions without Features



- Problem with feature-based methods
  - inherently unstable regions
  - no features create plateaus





Spatial Stability

- Can we say anything about the clusters themselves?

Spatial Stability

- Can we say anything about the clusters themselves?

  Idea: Prominent clusters have a minimum size under c-Lipschitz assumption

Spatial Stability

- Can we say anything about the clusters themselves?

  Idea: Prominent clusters have a minimum size under c-Lipschitz assumption

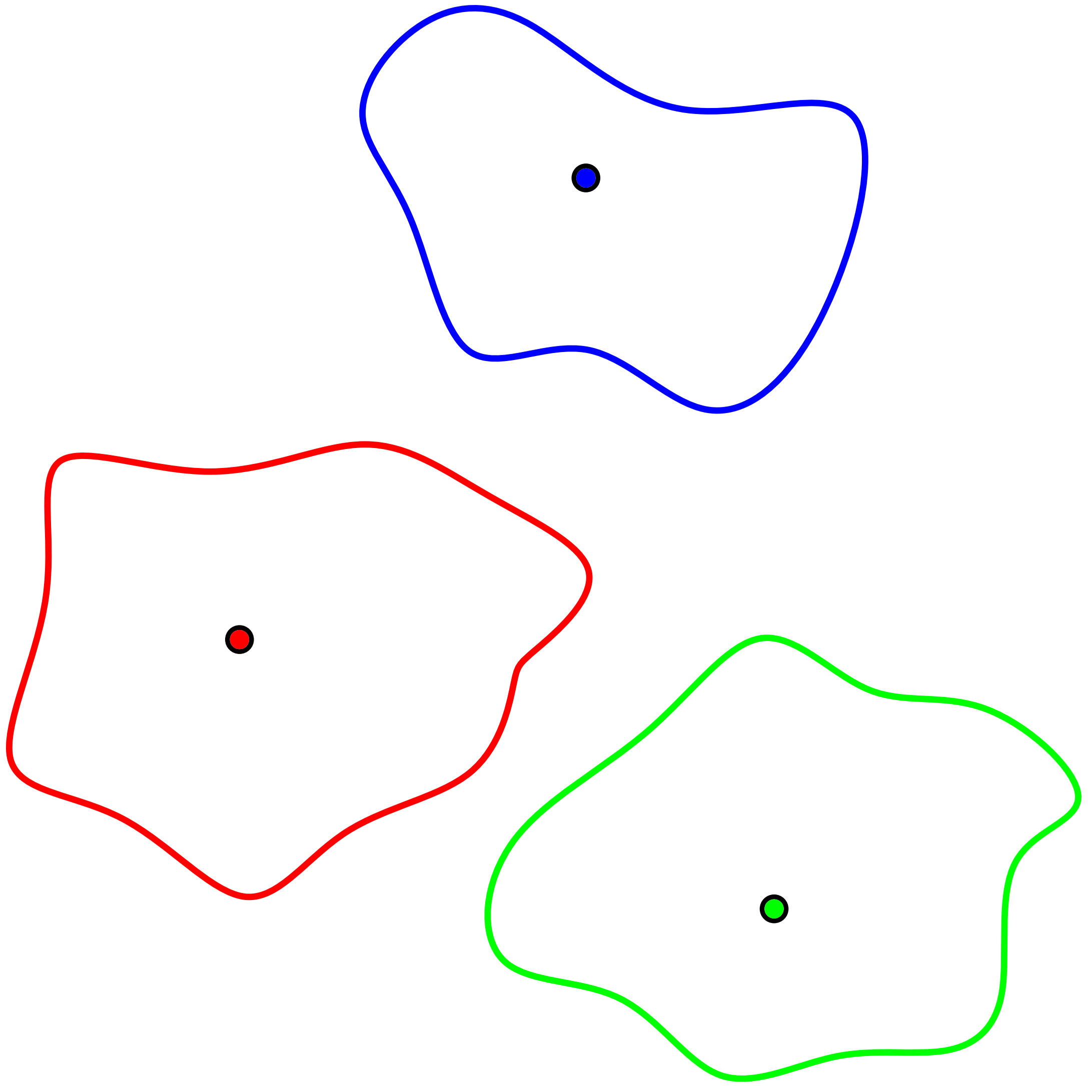
- Under small pertubations, prominent peak must be part of the "same" cluster

Spatial Stability

- Can we say anything about the clusters themselves?

  Idea: Prominent clusters have a minimum size under c-Lipschitz assumption

- Under small pertubations, prominent peak must be part of the "same" cluster

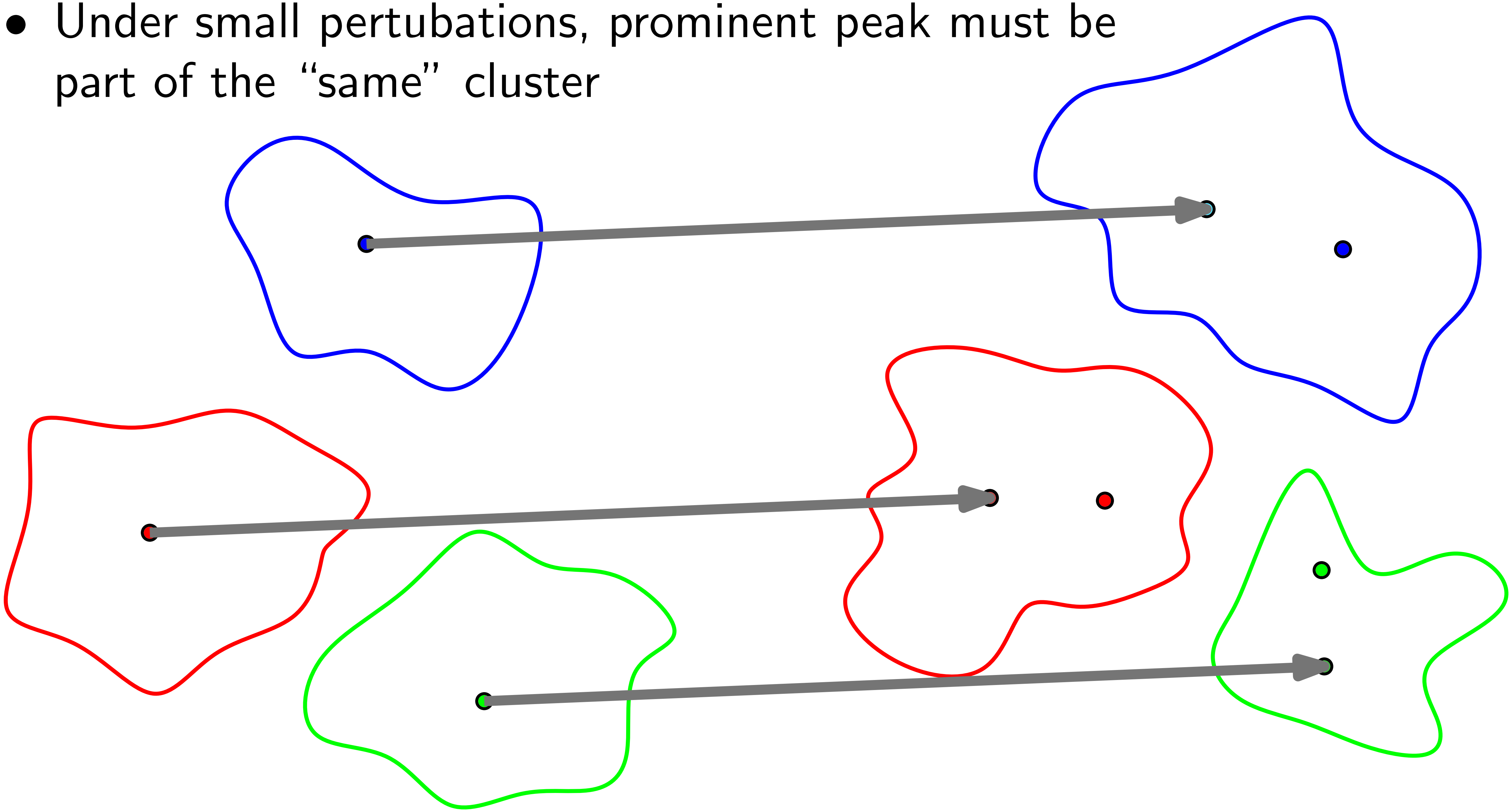


Spatial Stability

- Can we say anything about the clusters themselves?

  Idea: Prominent clusters have a minimum size under c-Lipschitz assumption

- Under small pertubations, prominent peak must be part of the "same" cluster

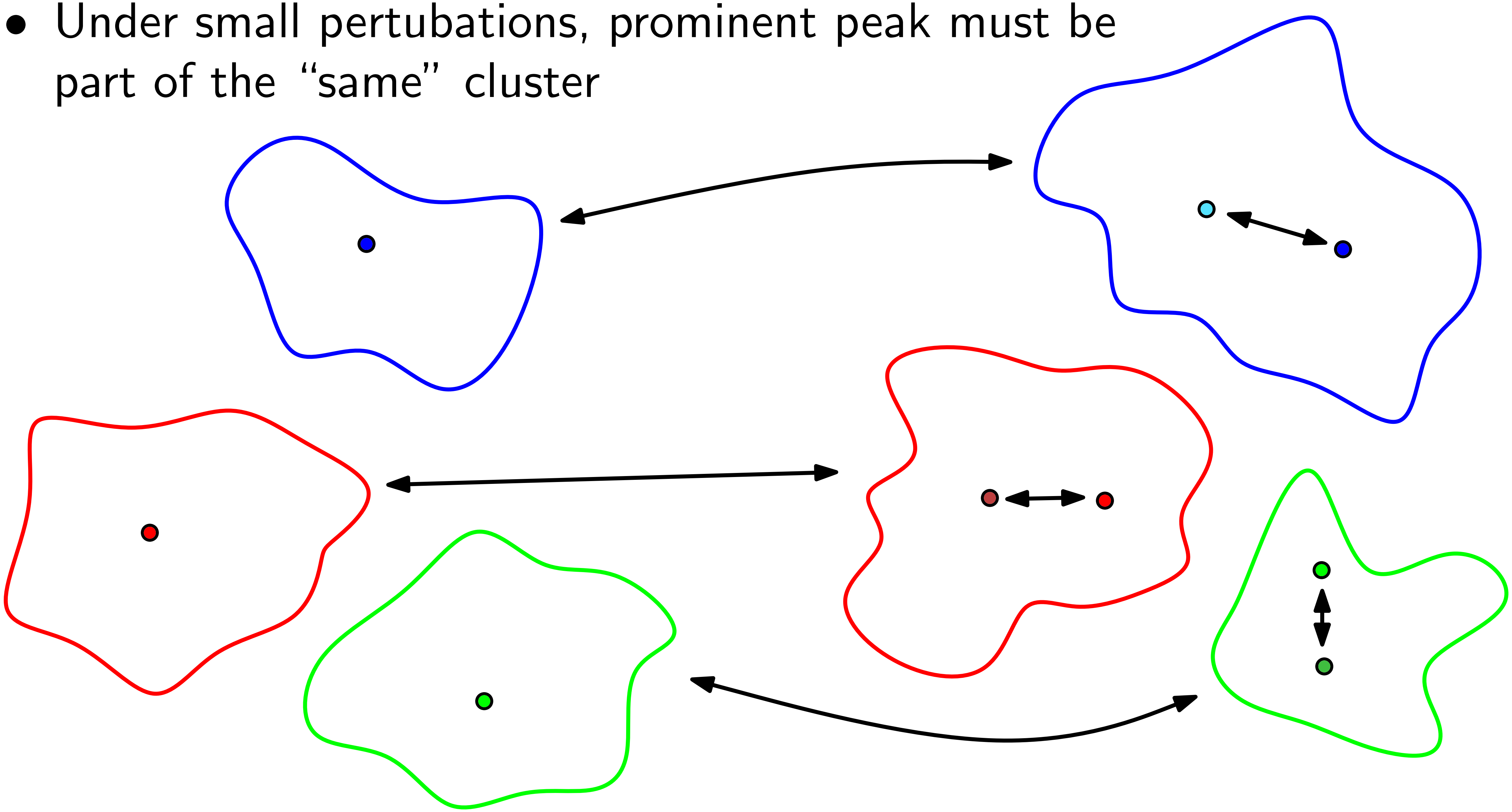


Spatial Stability

- Can we say anything about the clusters themselves?

  Idea: Prominent clusters have a minimum size under c-Lipschitz assumption

- Under small pertubations, prominent peak must be part of the "same" cluster



Regions without Features

- Solution: use spatial stability

Regions without Features

- Solution: use spatial stability

- Plan

Regions without Features

- Solution: use spatial stability

- Plan

    1. Number of segments stable
    2. Unique identification between segments under perturbations
    3. Find unstable parts

Regions without Features

- Solution: use spatial stability

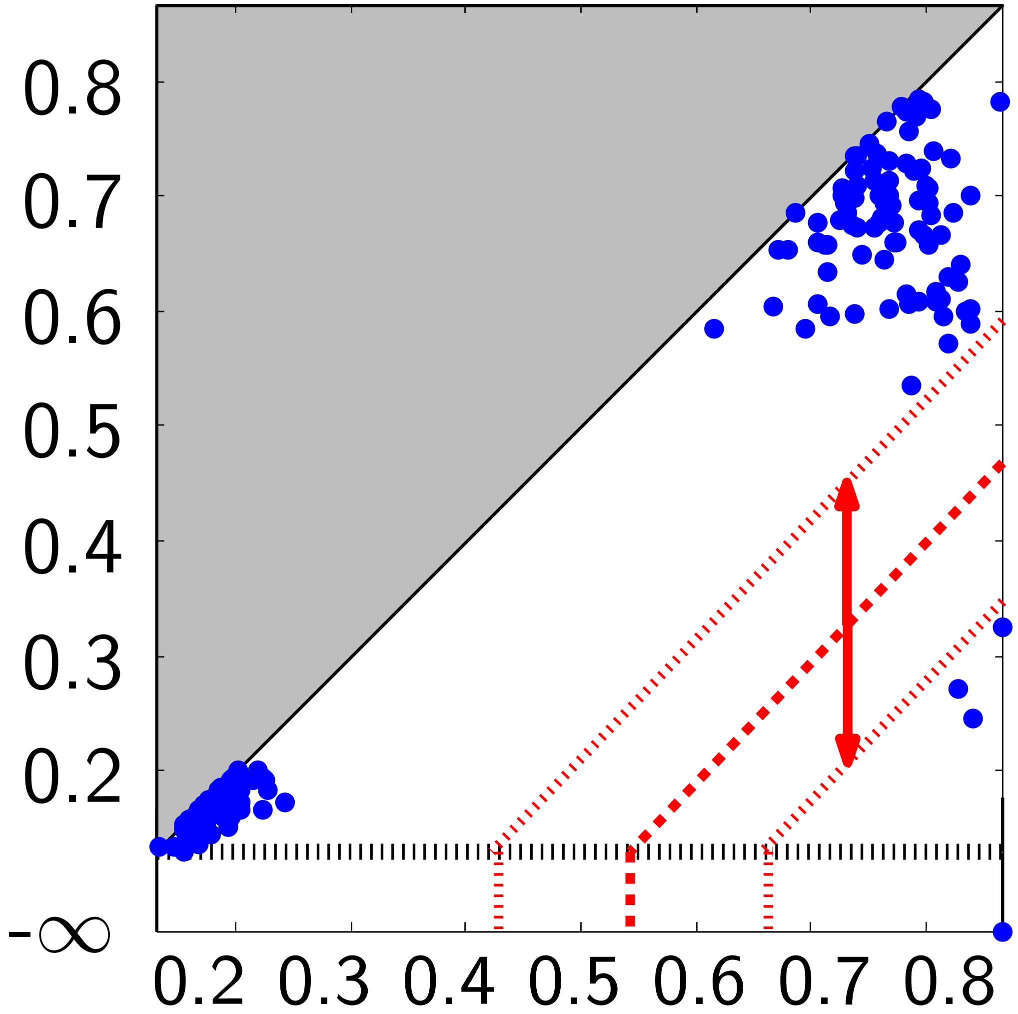
- Plan
  1. Number of segments stable
  2. Unique identification between segments under perturbations
  3. Find unstable parts

Segment unstable parts separately

Extended Algorithm

1. Run the algorithm to obtain persistence diagram

2. Choose threshold and perturbation amount



Extended Algorithm

1. Run the algorithm to obtain persistence diagram

2. Choose threshold and perturbation amount

3. For $i = 1 \ldots N$

    a. Perturb function values

    b. Run clustering algorithm

    c. Find one-to-one correspondance between segments

Extended Algorithm

1. Run the algorithm to obtain persistence diagram

2. Choose threshold and perturbation amount

3. For $i = 1 \ldots N$

    a. Perturb function values

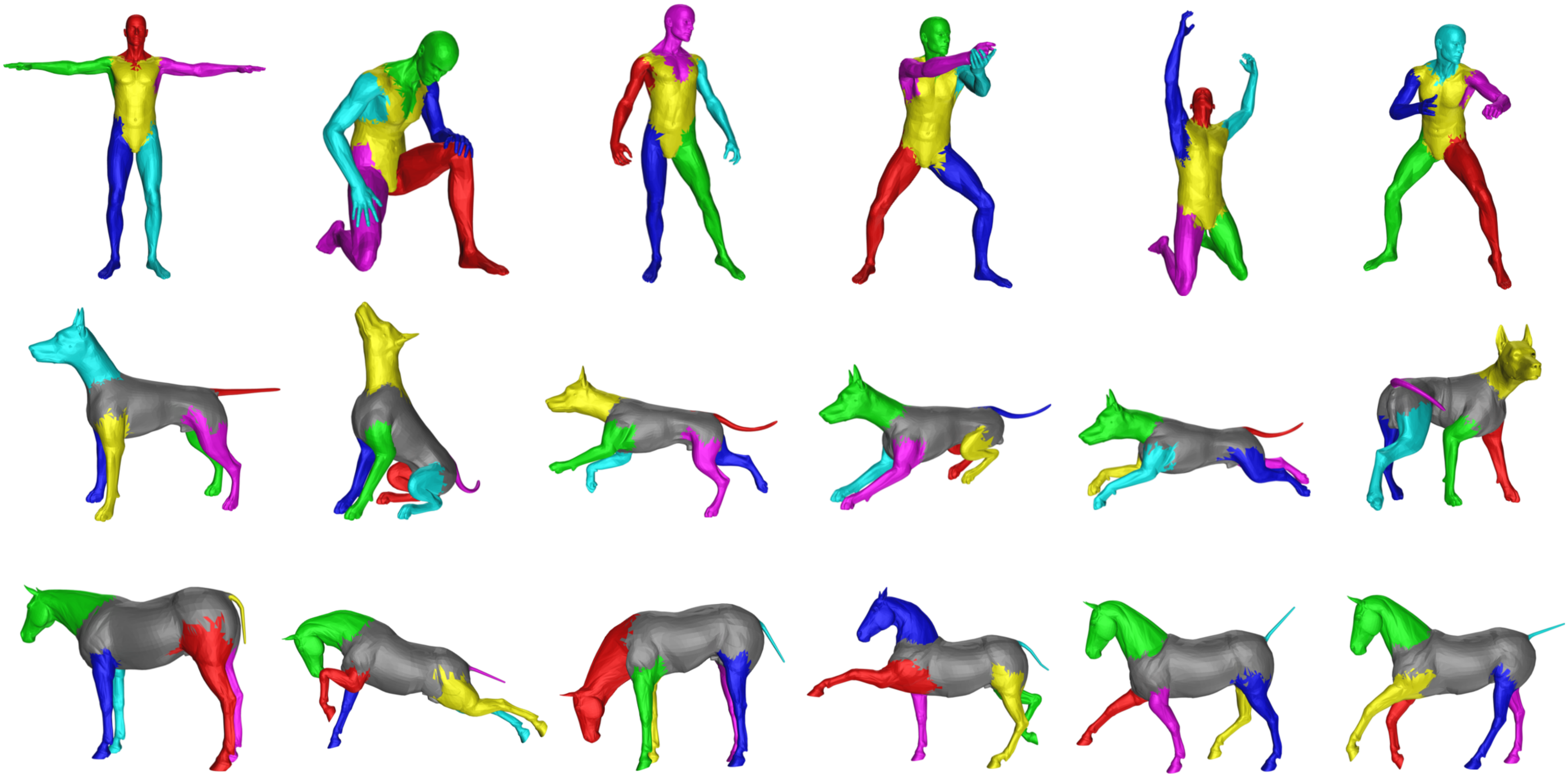
    b. Run clustering algorithm

    c. Find one-to-one correspondance between segments

4. Find stable and unstable parts

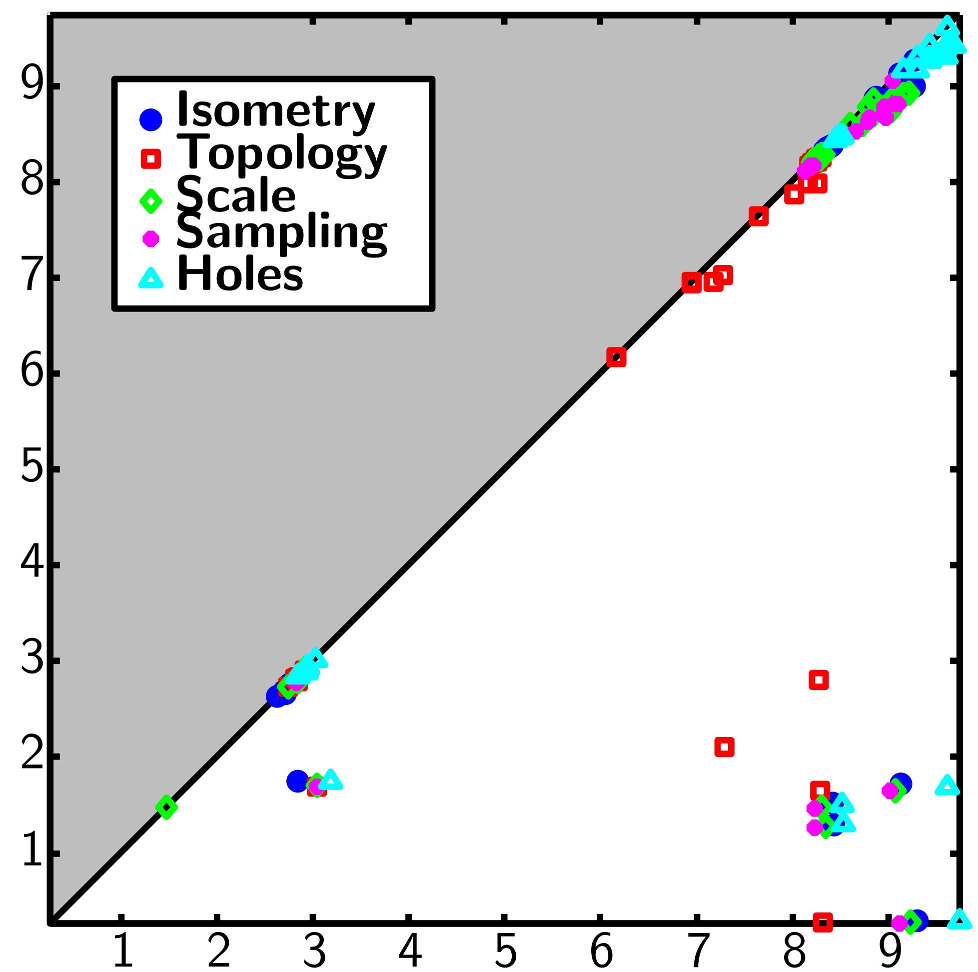
    Each point has a distribution over possible segments

Extended Segmentation



Robustness to Noise

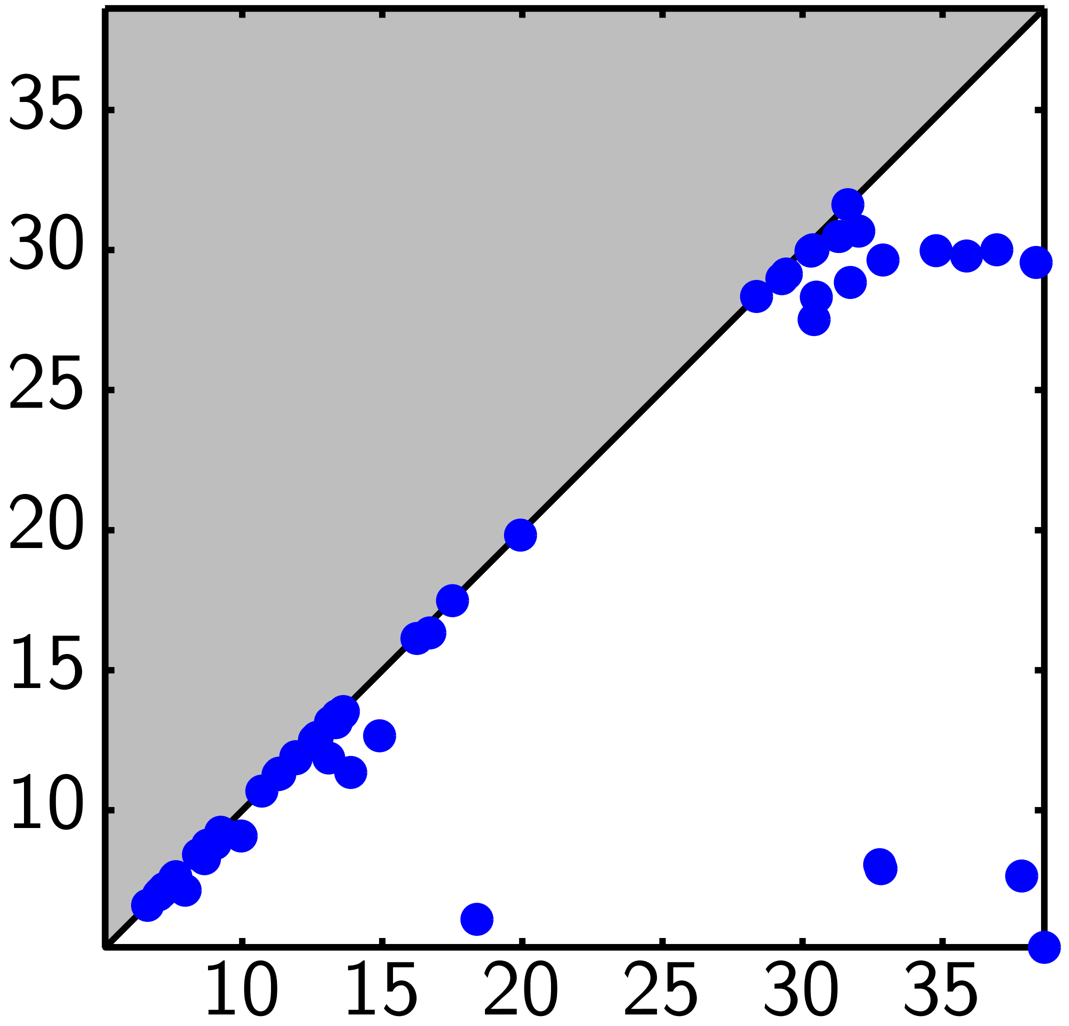
Additive Noise

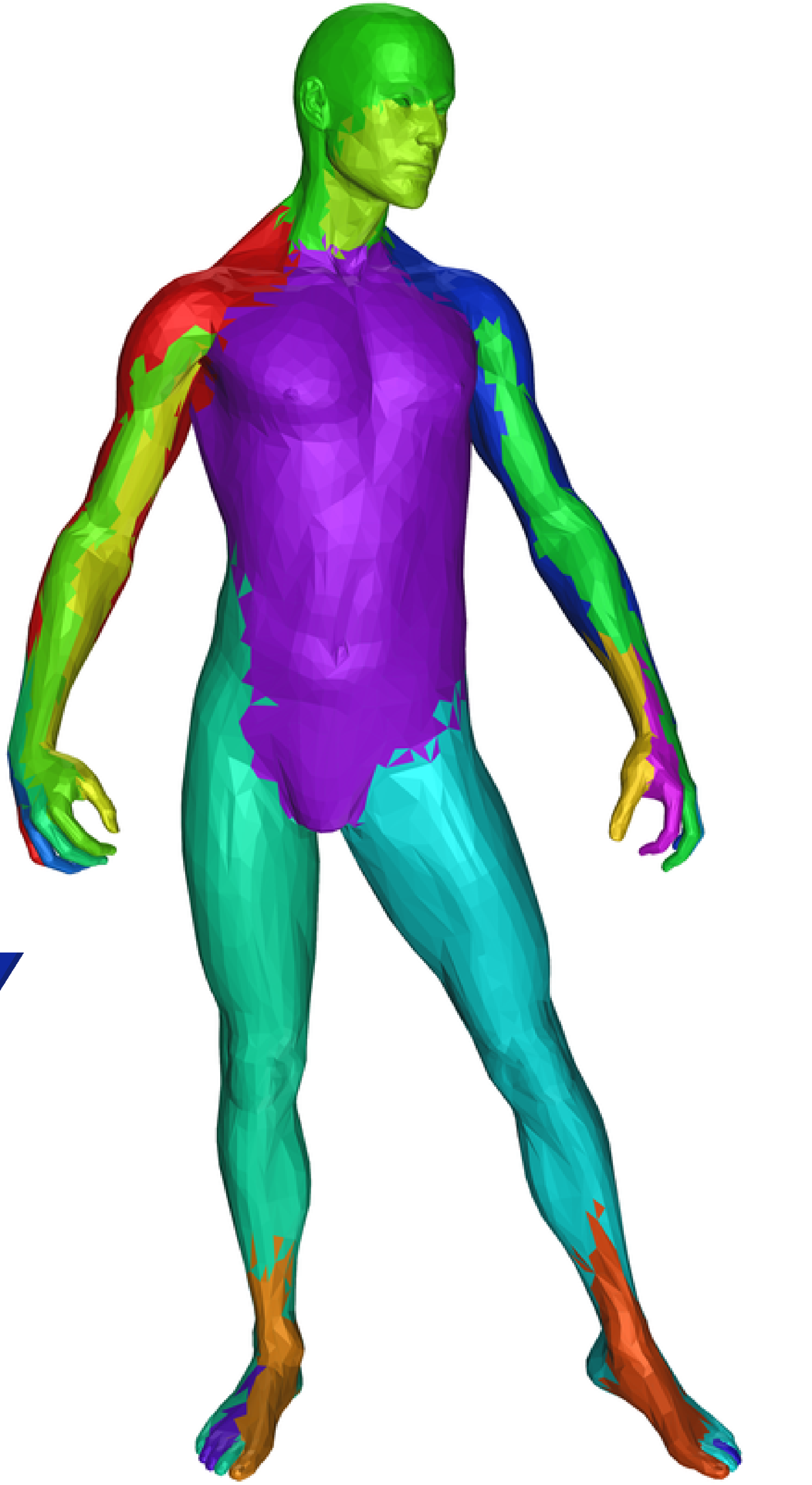


Topological Noise

Holes added

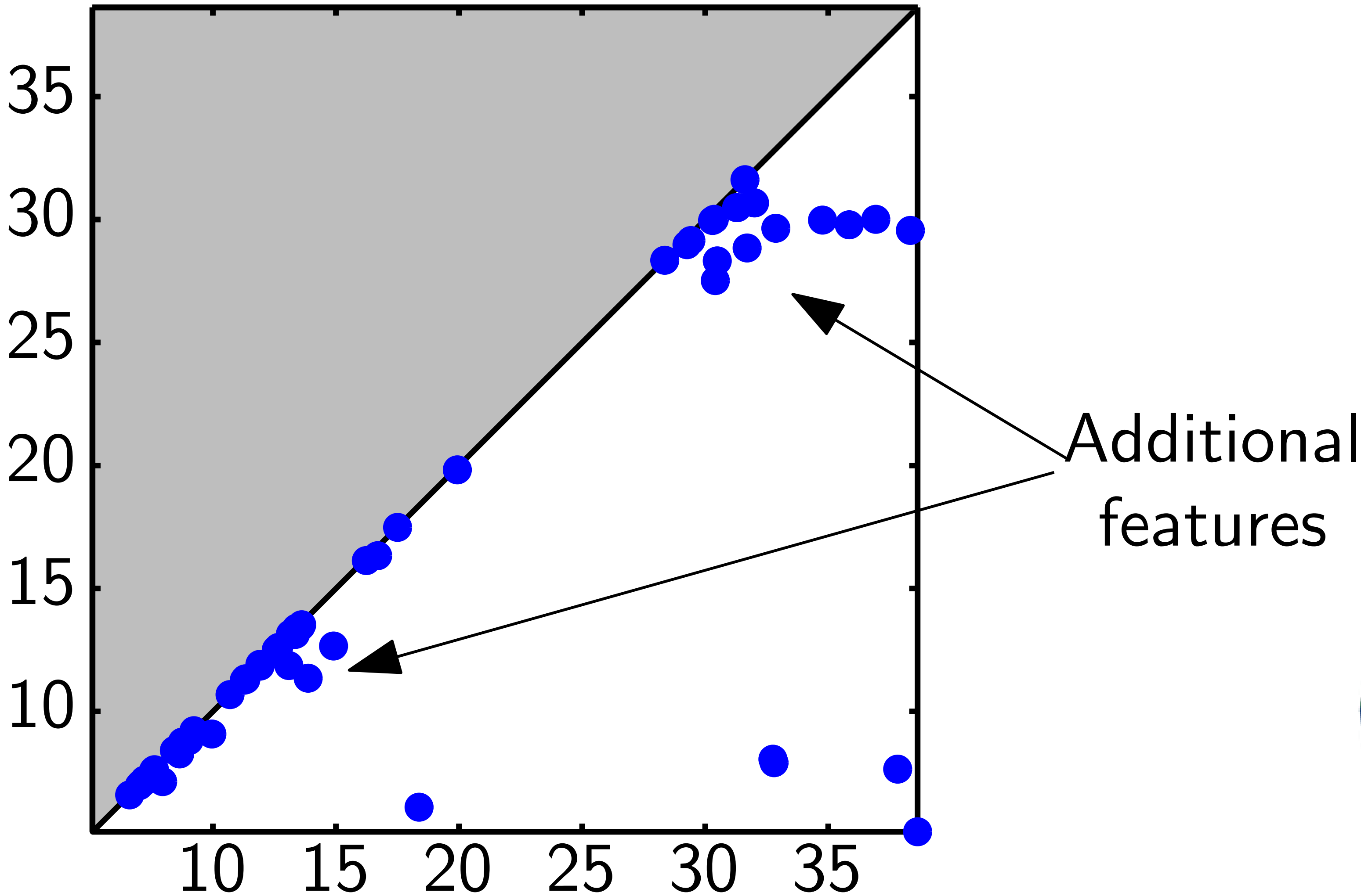
Scale of Features: Finer Scale

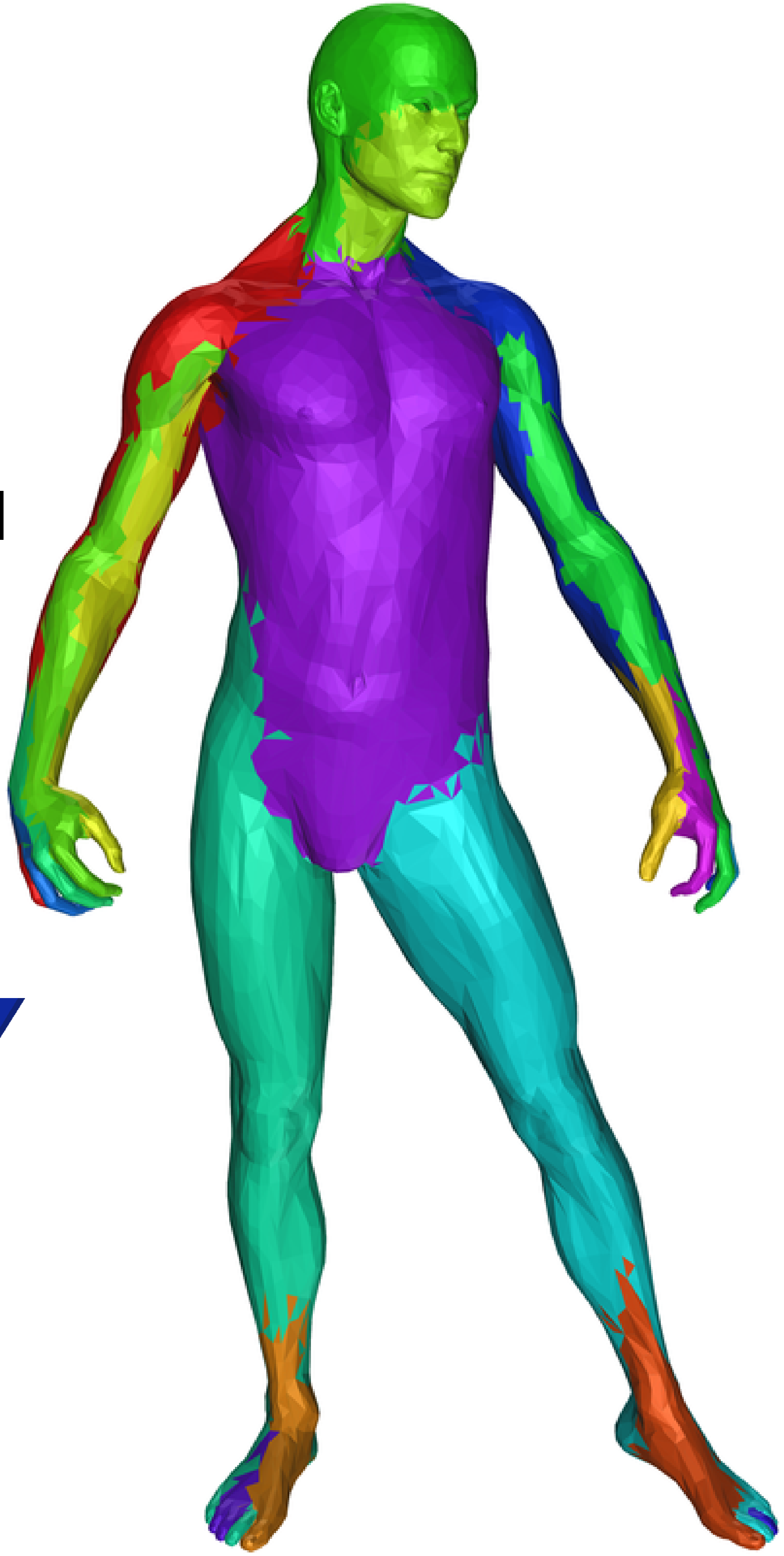




Extended Algorithm
$t = 0.01$

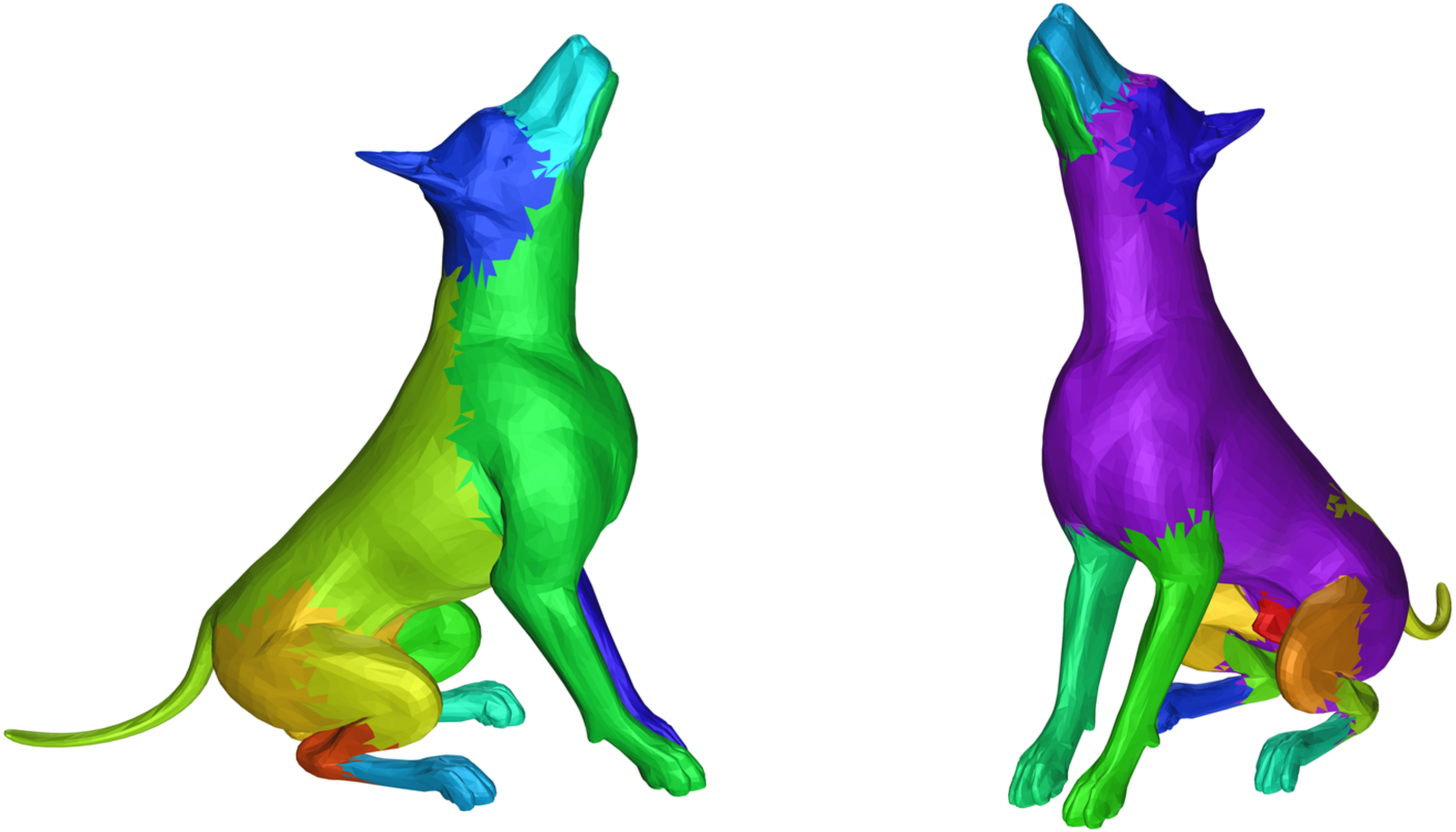
Scale of Features: Finer Scale





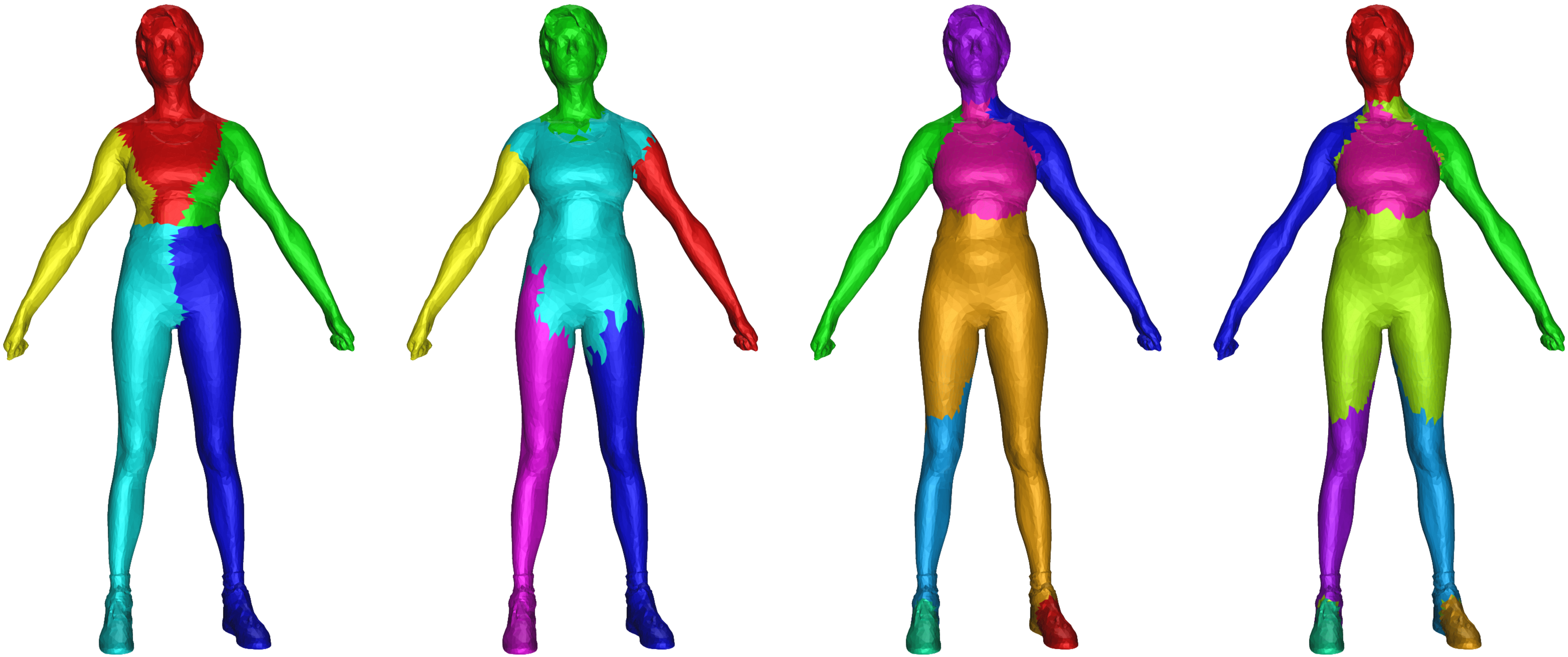
Extended Algorithm
$t = 0.01$

Scale of Features: Finer Scale



Standard Extended

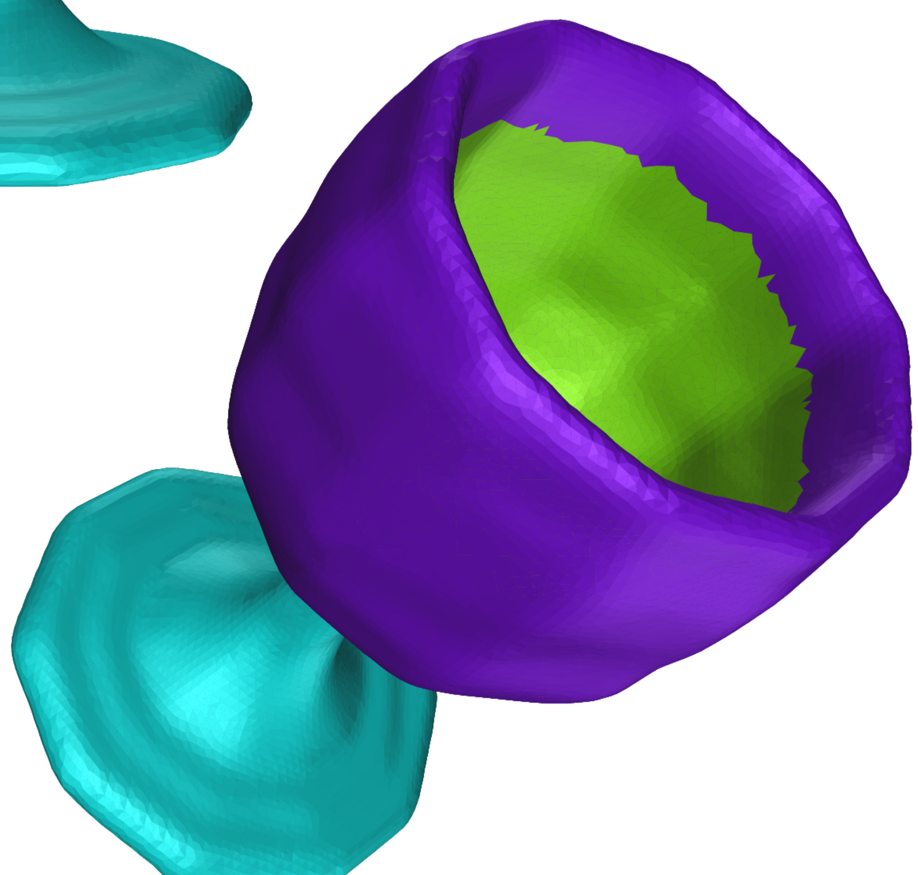
A Few Results

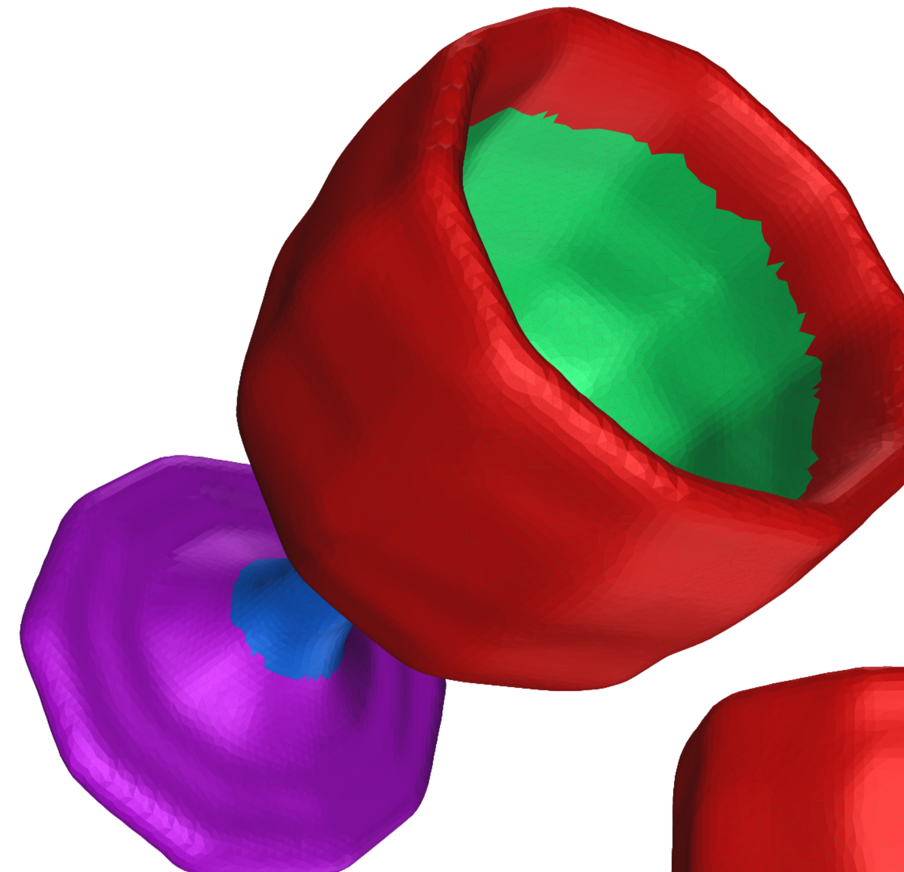


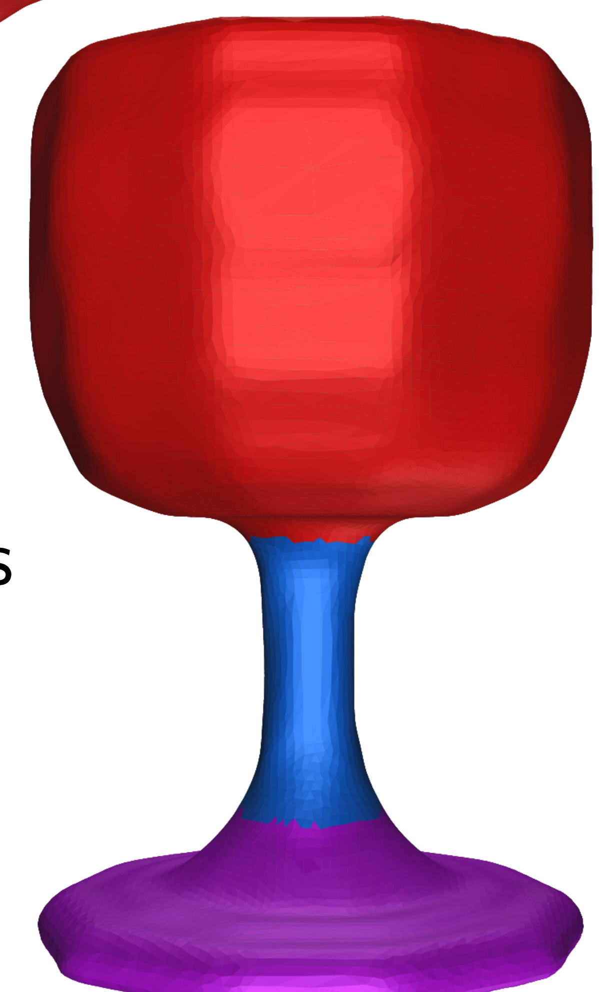
Standard $t = 0.1$ | Extended $t = 0.1$ | Standard $t = 0.01$ | Extended $t = 0.01$

A Few Results

3 Segments
$t = 0.1$

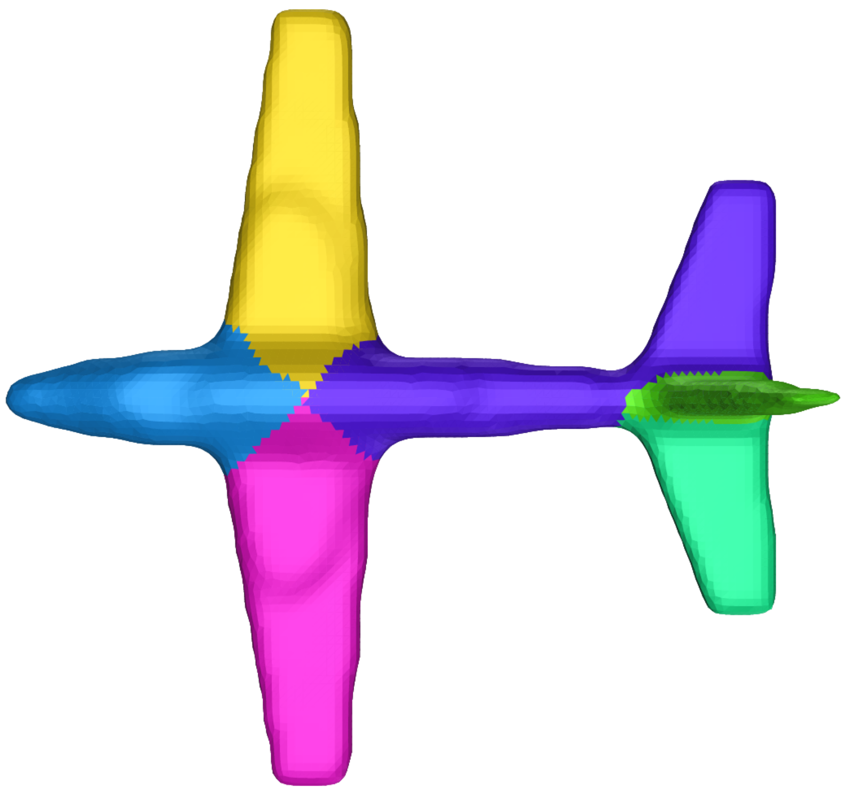




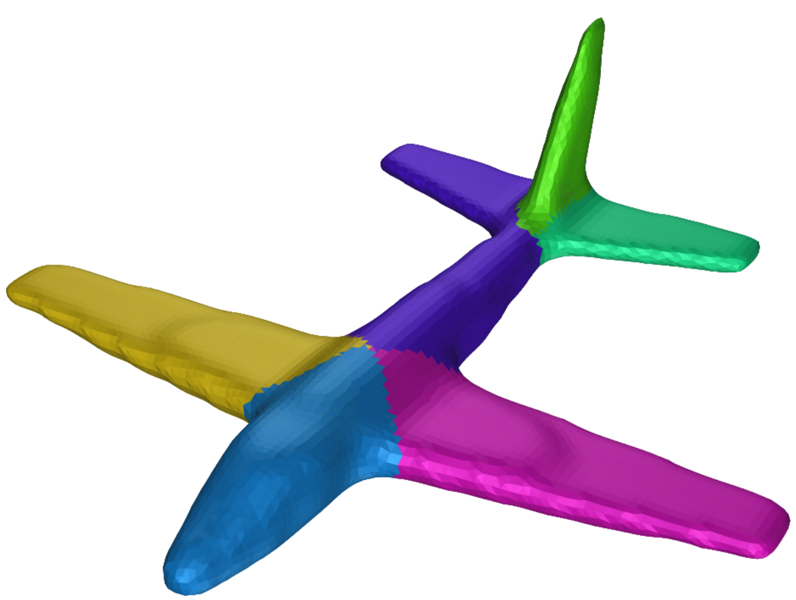


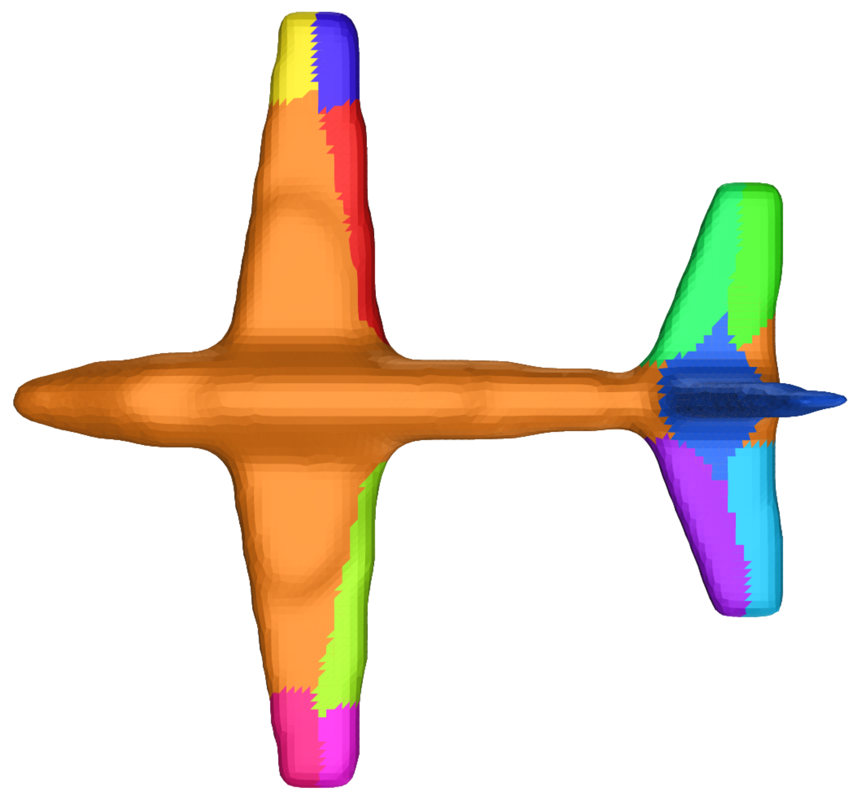
4 Segments
$t = 0.1$

A Few Results

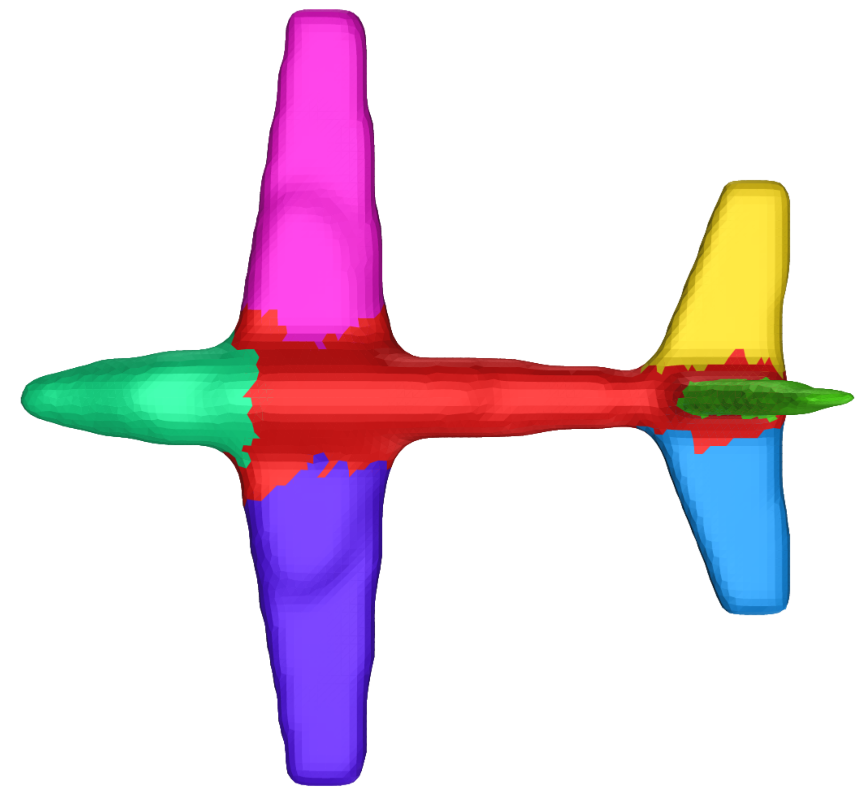


Standard
$t = 0.1$

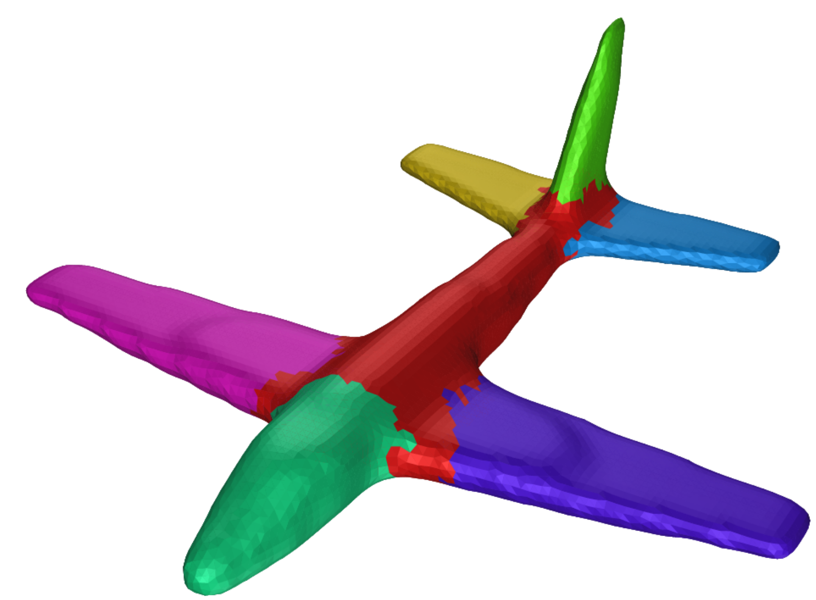




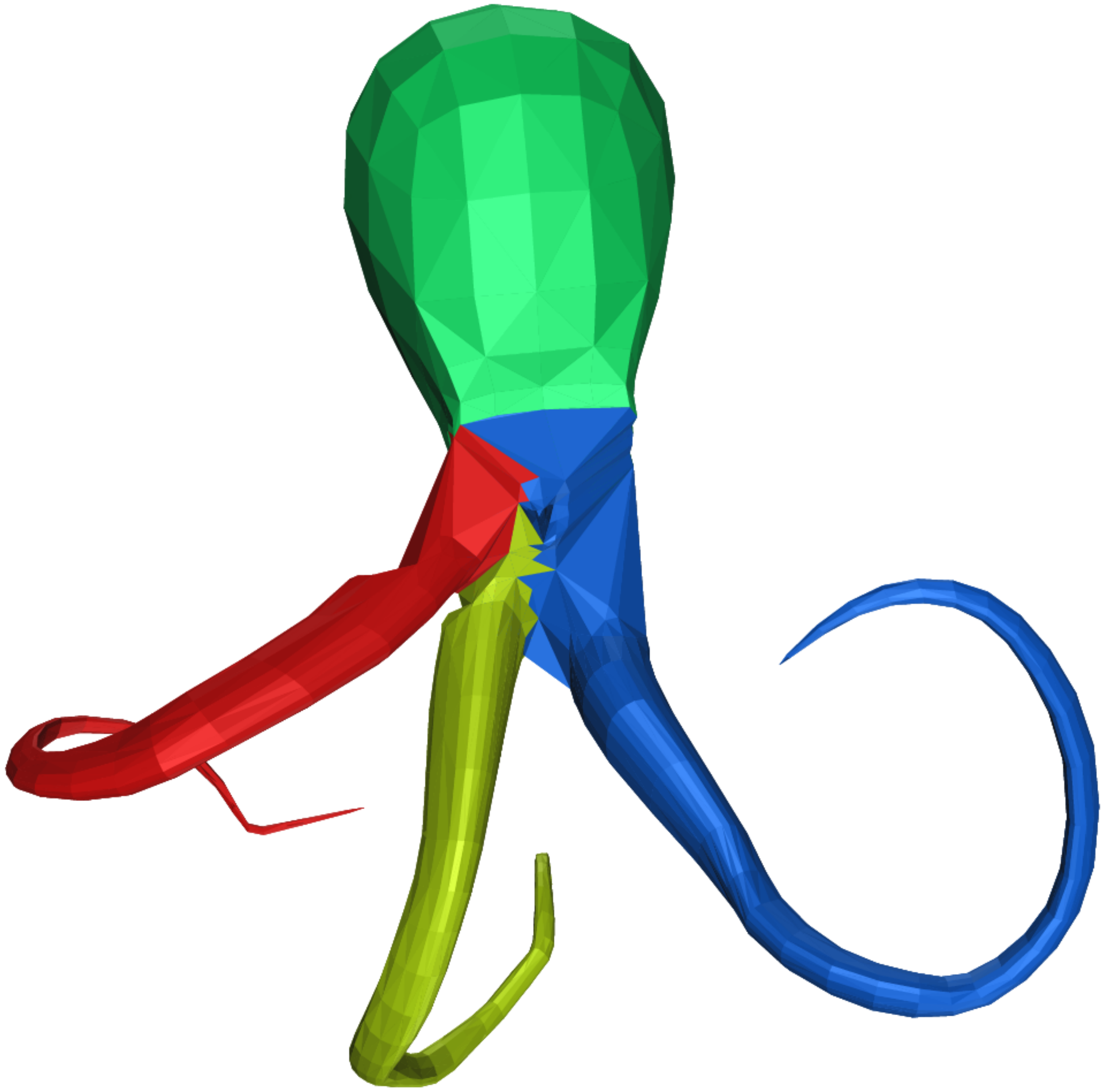
Standard
$t = 0.01$



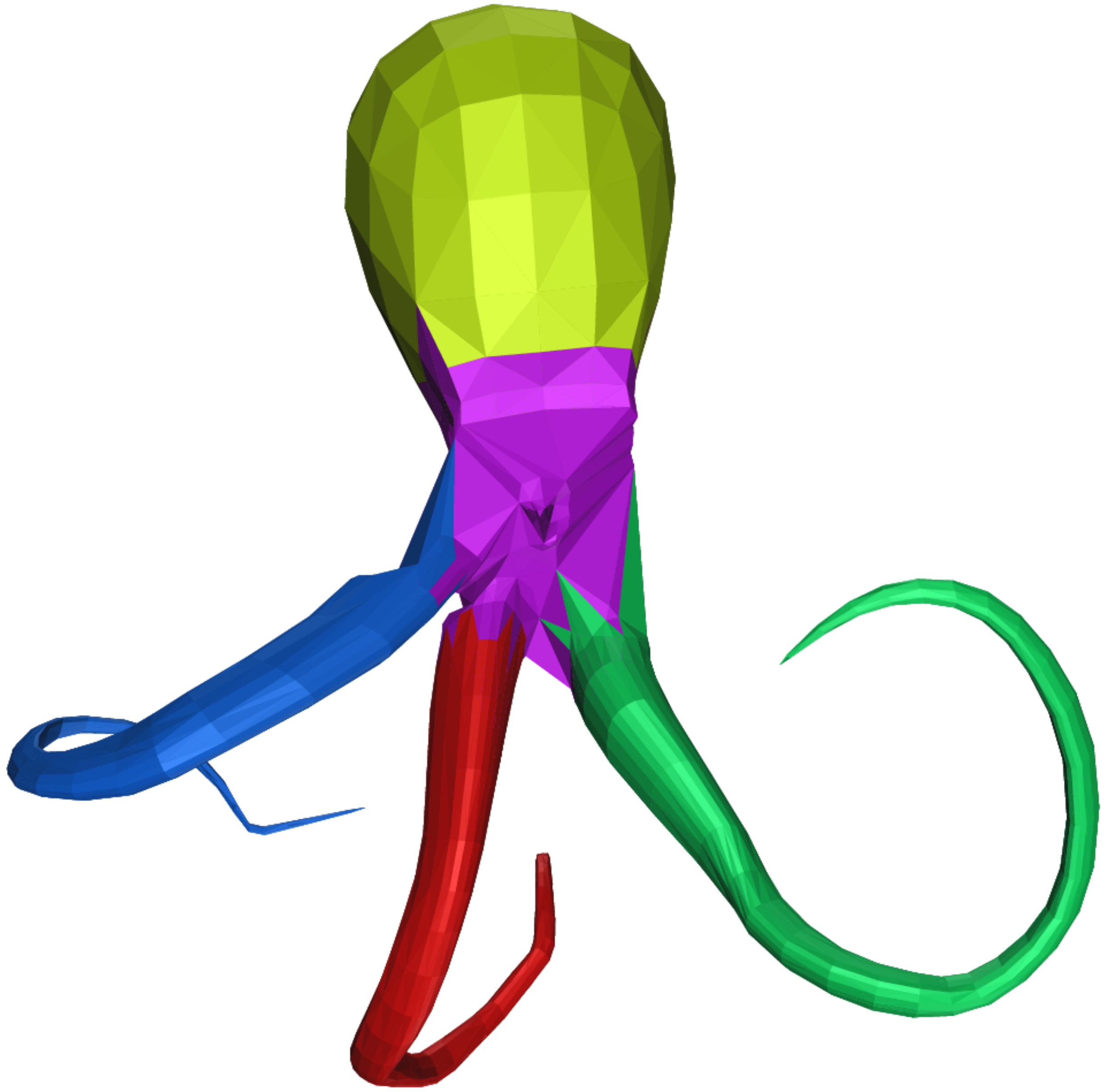
Extended
$t = 0.1$



A Few Results



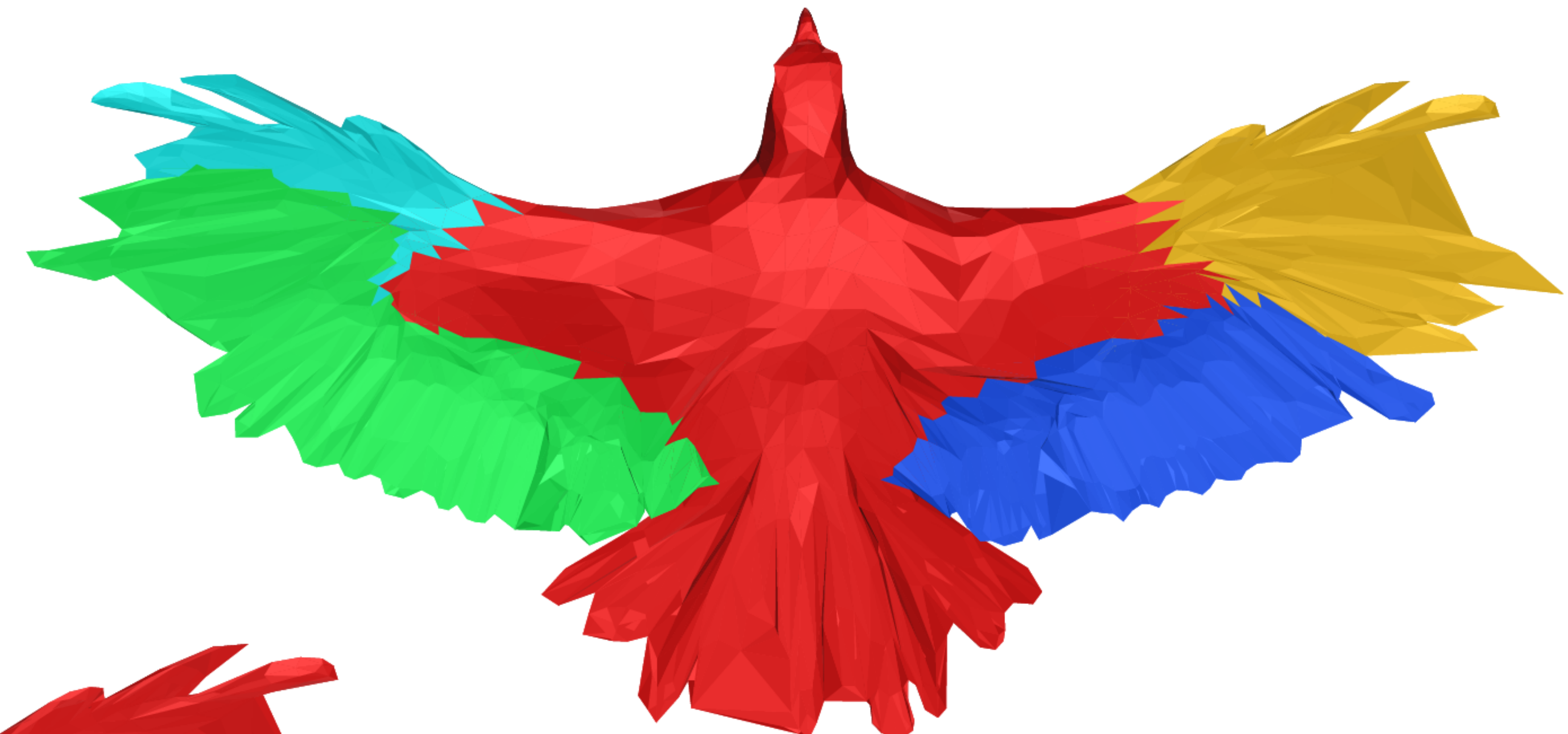
Standard
$t = 0.1$



Extended
$t = 0.1$

A Few Results

Standard
$t = 0.1$



Standard
$t = 0.01$

Extended
$t = 0.1$

Some Remarks

- We find all "likely" unstable points

Some Remarks

- We find all "likely" unstable points
- For a fixed set of sample points
  - Each point is assigned a (stochastic) vector in $[0,1]^d$
  - i-th entry is the likelihood that the point is assigned to cluster i

Some Remarks

- We find all "likely" unstable points
- For a fixed set of sample points
  - Each point is assigned a (stochastic) vector in $[0, 1]^d$
  - i-th entry is the likelihood that the point is assigned to cluster i
- The probability vector of each point quickly converges

Some Remarks

- We find all “likely” unstable points
- For a fixed set of sample points
  - Each point is assigned a (stochastic) vector in $[0, 1]^d$
  - i-th entry is the likelihood that the point is assigned to cluster i
- The probability vector of each point quickly converges
- Over different samplings
  - How do these vectors behave?

Take-Home Message(s)

- Interpretation of persistence diagrams

Take-Home Message(s)

- Interpretation of persistence diagrams
- Practical algorithm
  - Reasonable time complexity
  - Tunable with a small number of parameters
  - Robust

Take-Home Message(s)

- Interpretation of persistence diagrams
- Practical algorithm
    - Reasonable time complexity
    - Tunable with a small number of parameters
    - Robust
- Understanding limits of stability
    - Extended algorithm

Segmentation Benchmark

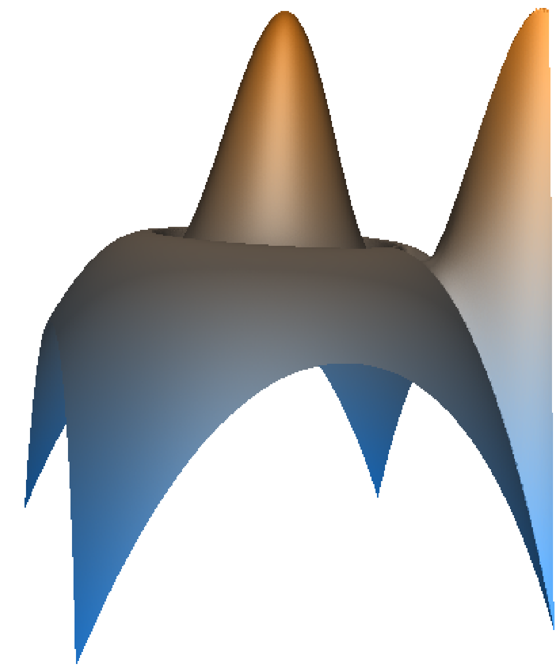
- Benchmark of human-segmentations
- Compared to curvature function

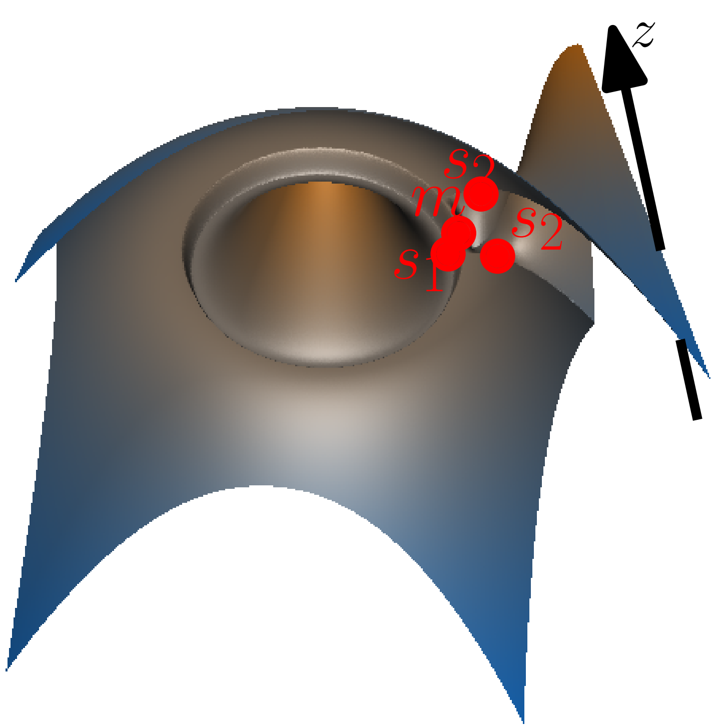
Method	CD	RI	HD	CE	
				1	2
Curvature Basic	0.348	0.218	0.257	0.255	0.253
Curvature Extended	0.322	0.221	0.220	0.232	0.201
HKS Basic	0.213	0.124	0.105	0.129	0.067
HKS Extended	0.164	0.120	0.097	0.121	0.061

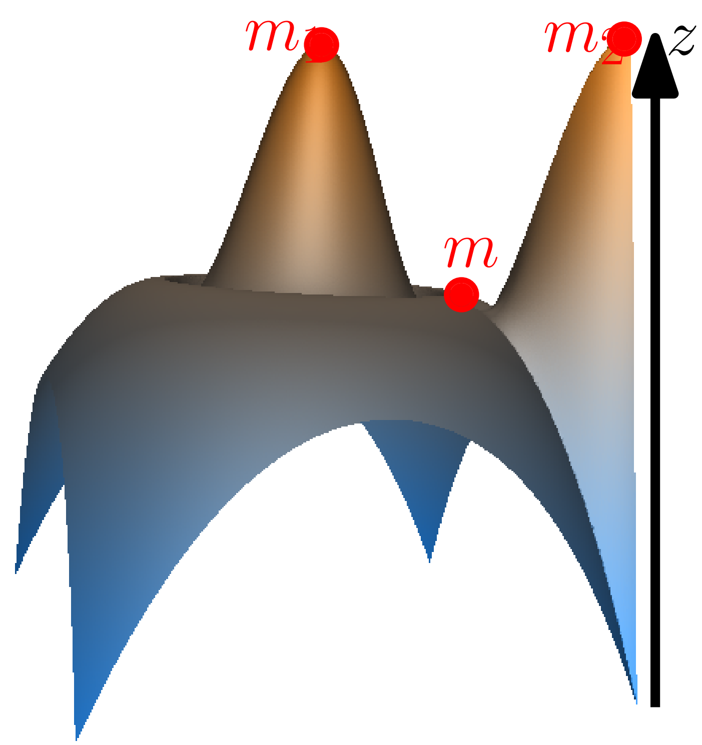
- Comparable to other methods

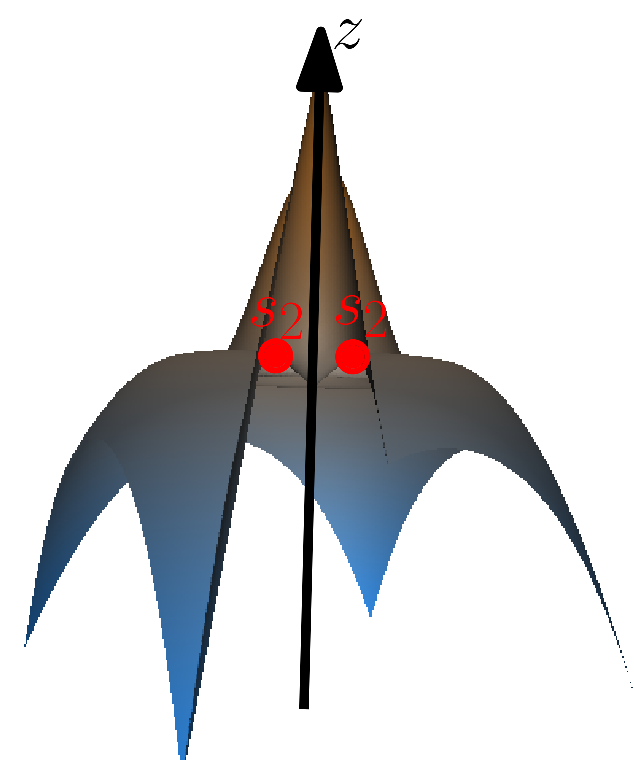
Spatial Stability

- Unstable part can be arbitrarily large



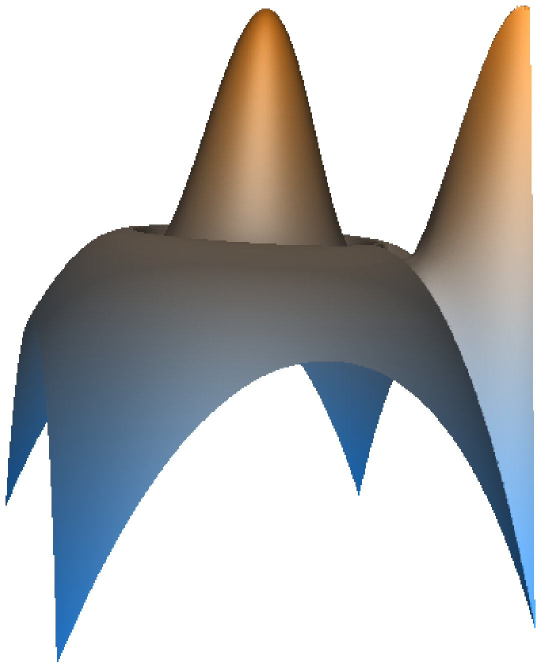


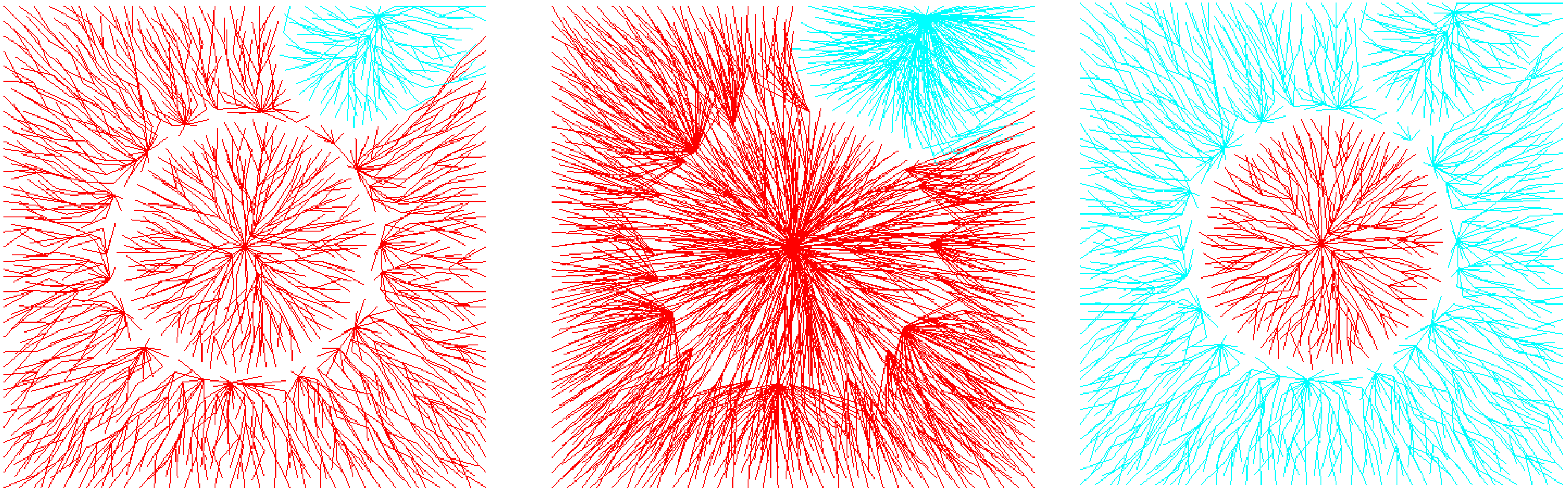




Spatial Stability

- Unstable part can be arbitrarily large





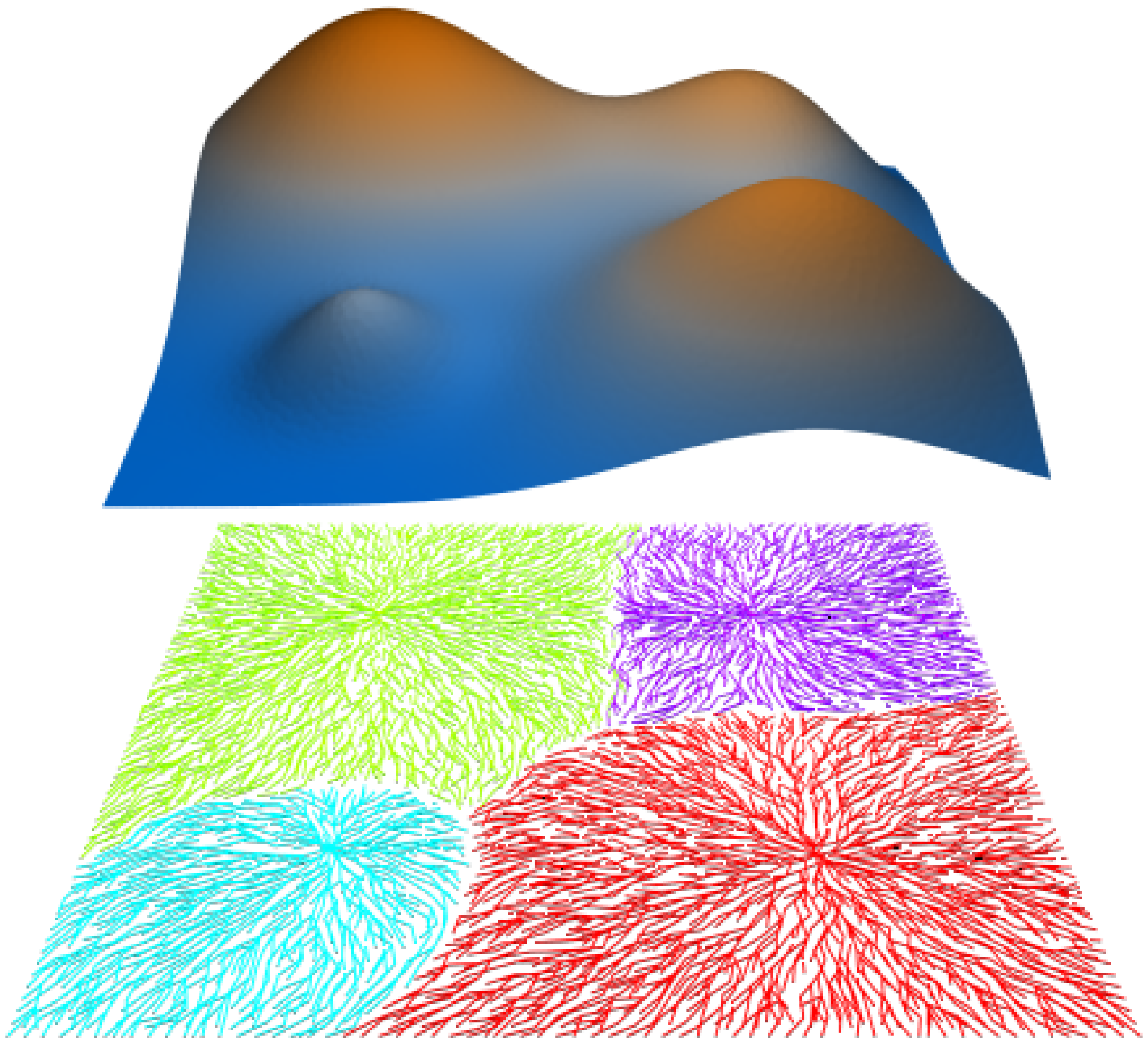
Segmentation Benchmark

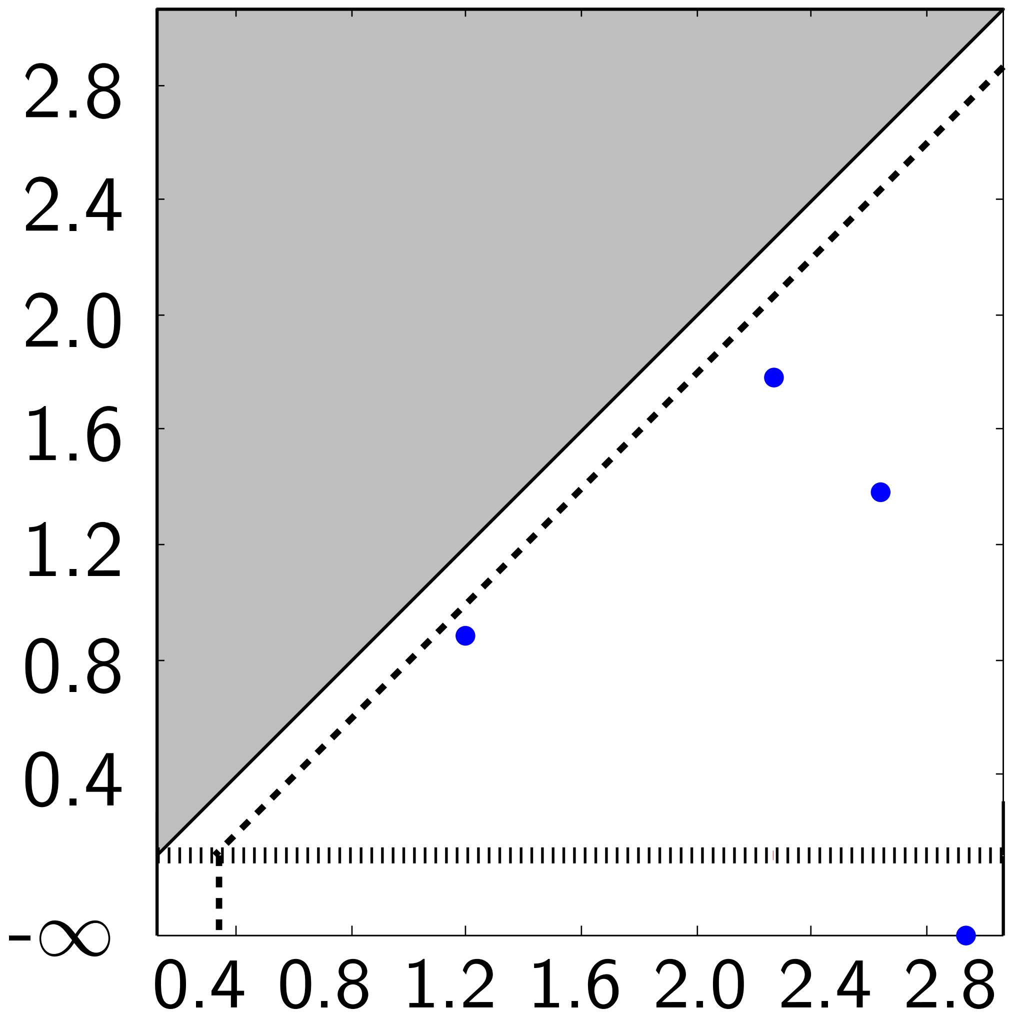
- Benchmark of human-segmentations
- Compared to curvature function

Method	CD	RI	HD	CE	
				1	2
Curvature Basic	0.348	0.218	0.257	0.255	0.253
Curvature Extended	0.322	0.221	0.220	0.232	0.201
HKS Basic	0.213	0.124	0.105	0.129	0.067
HKS Extended	0.164	0.120	0.097	0.121	0.061

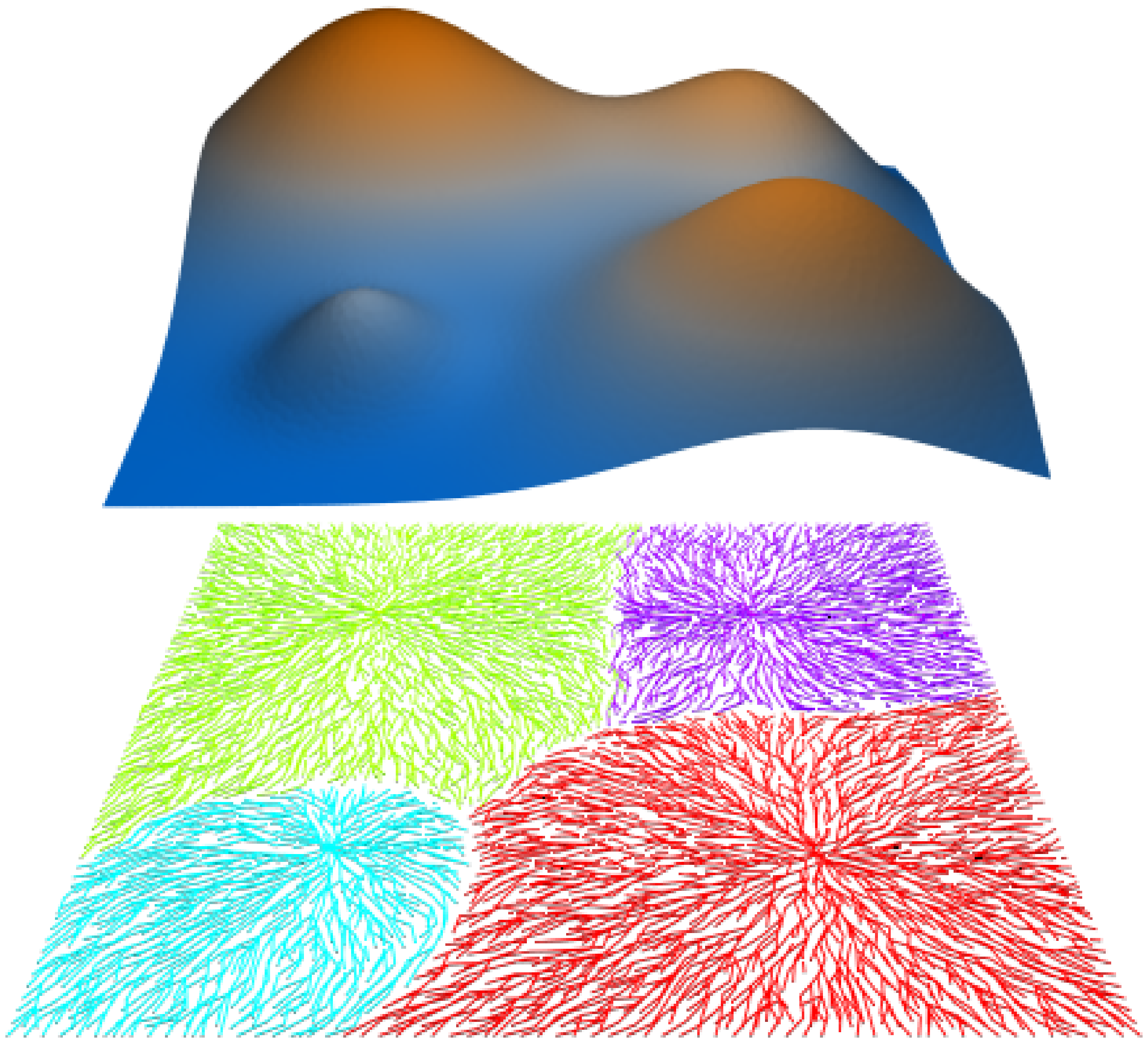
- Comparable to other methods

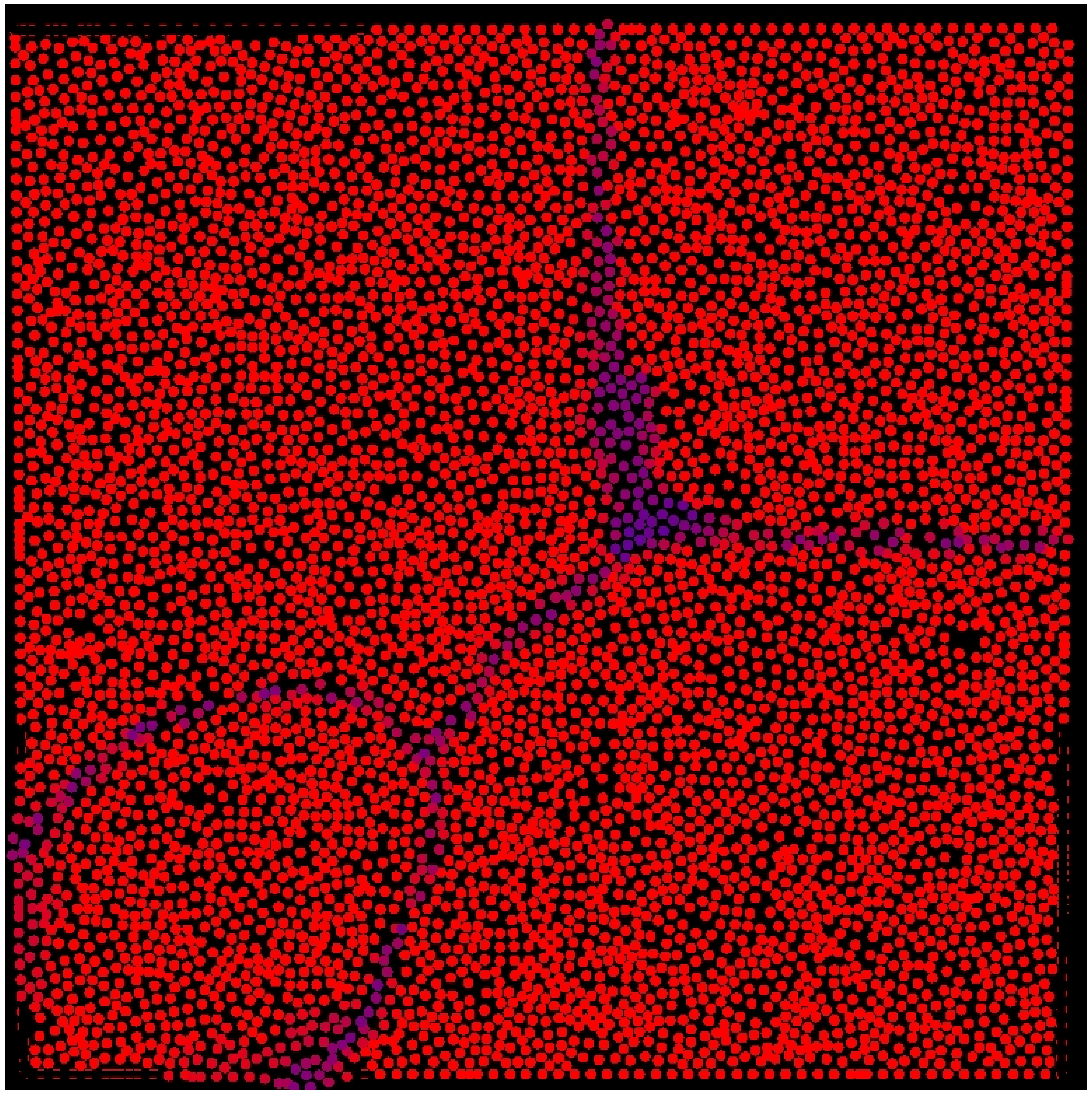
4 Gaussians



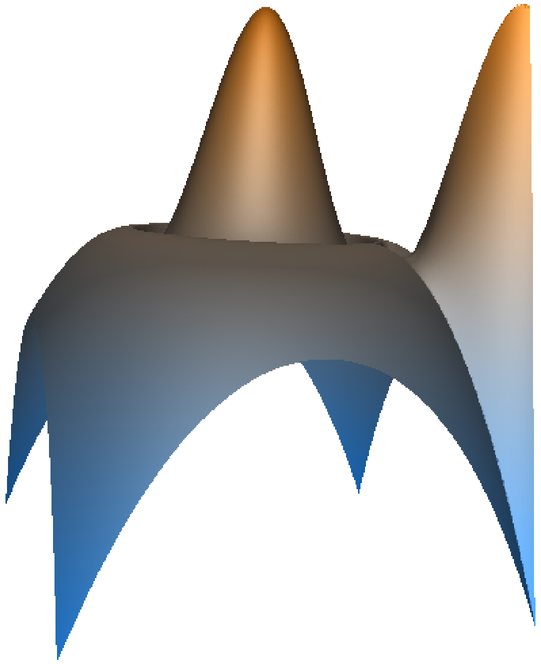


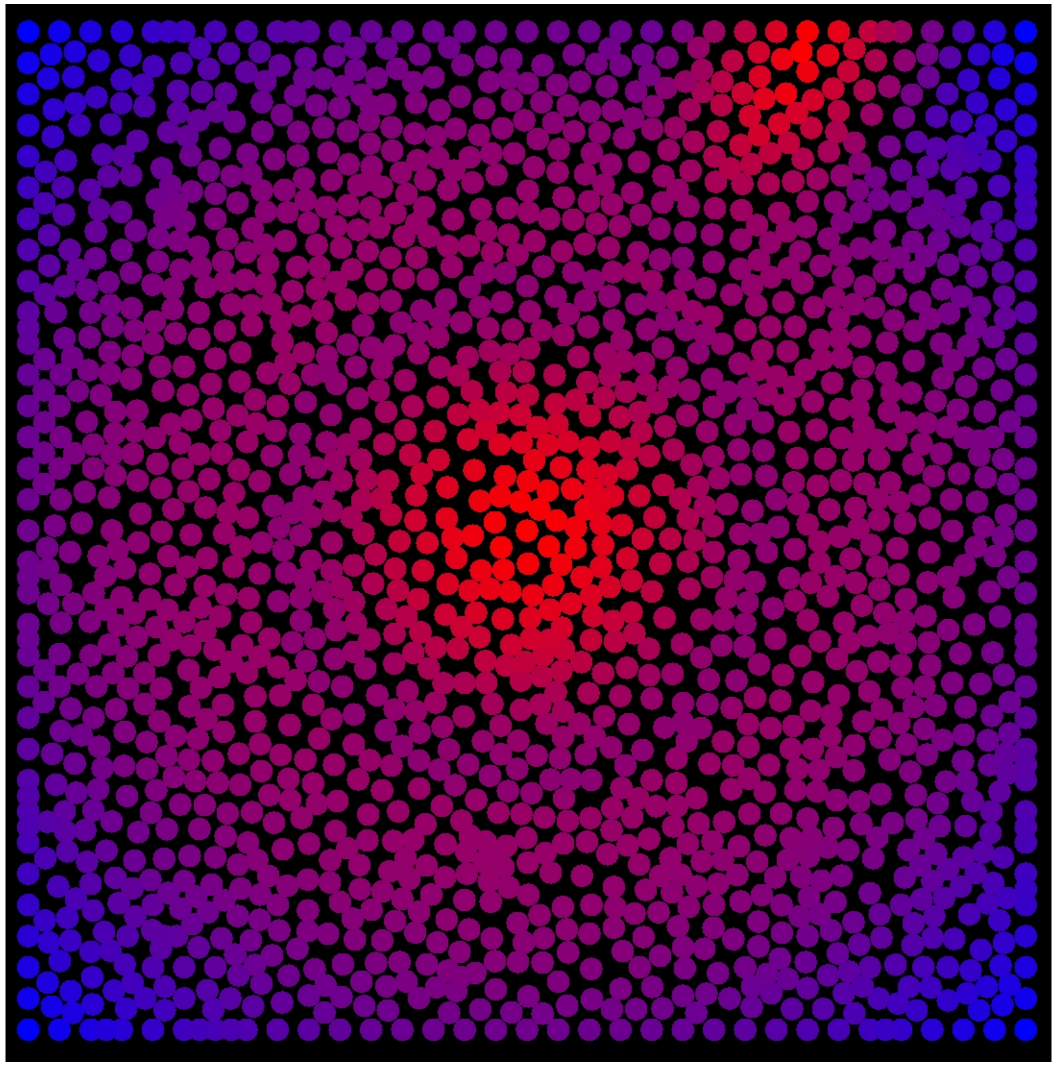
4 Gaussians

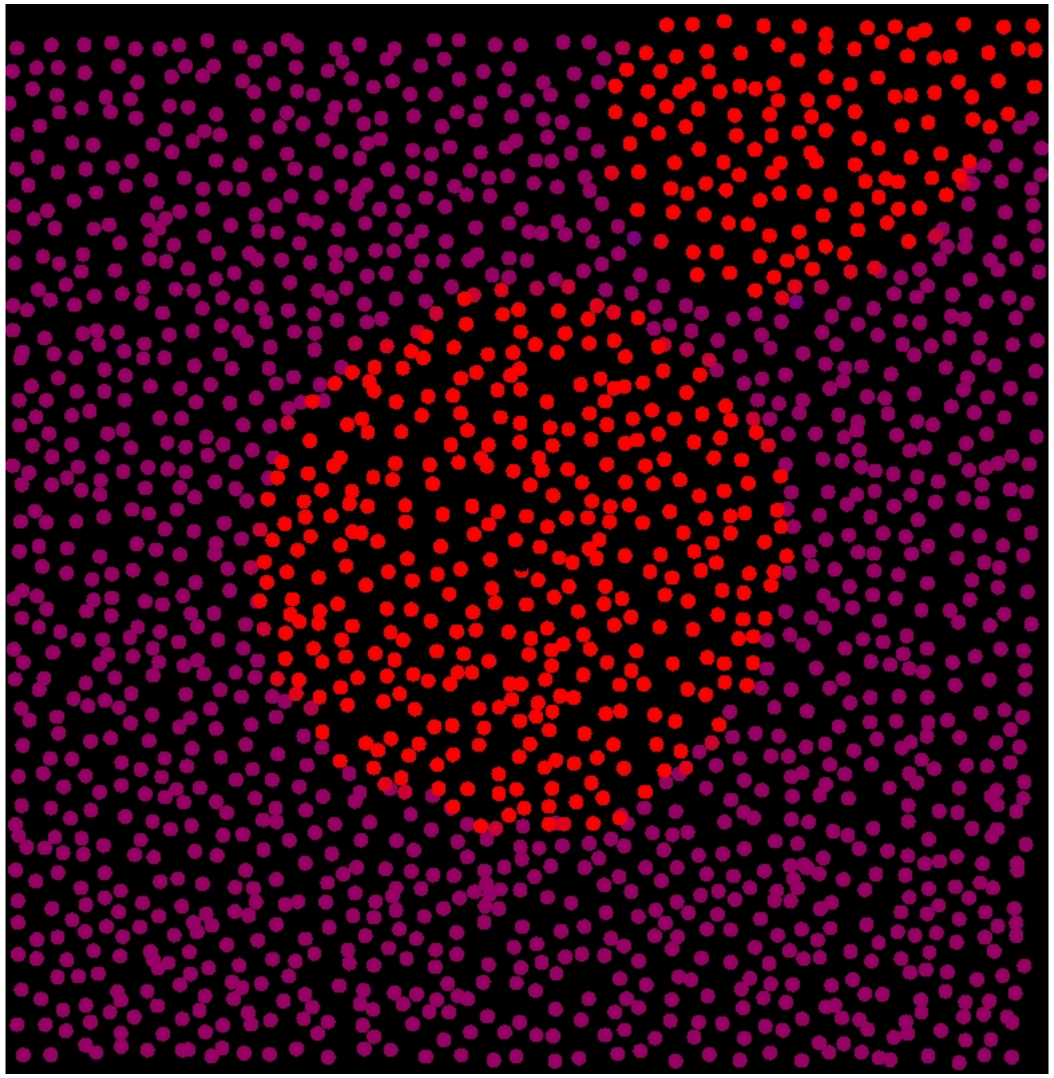




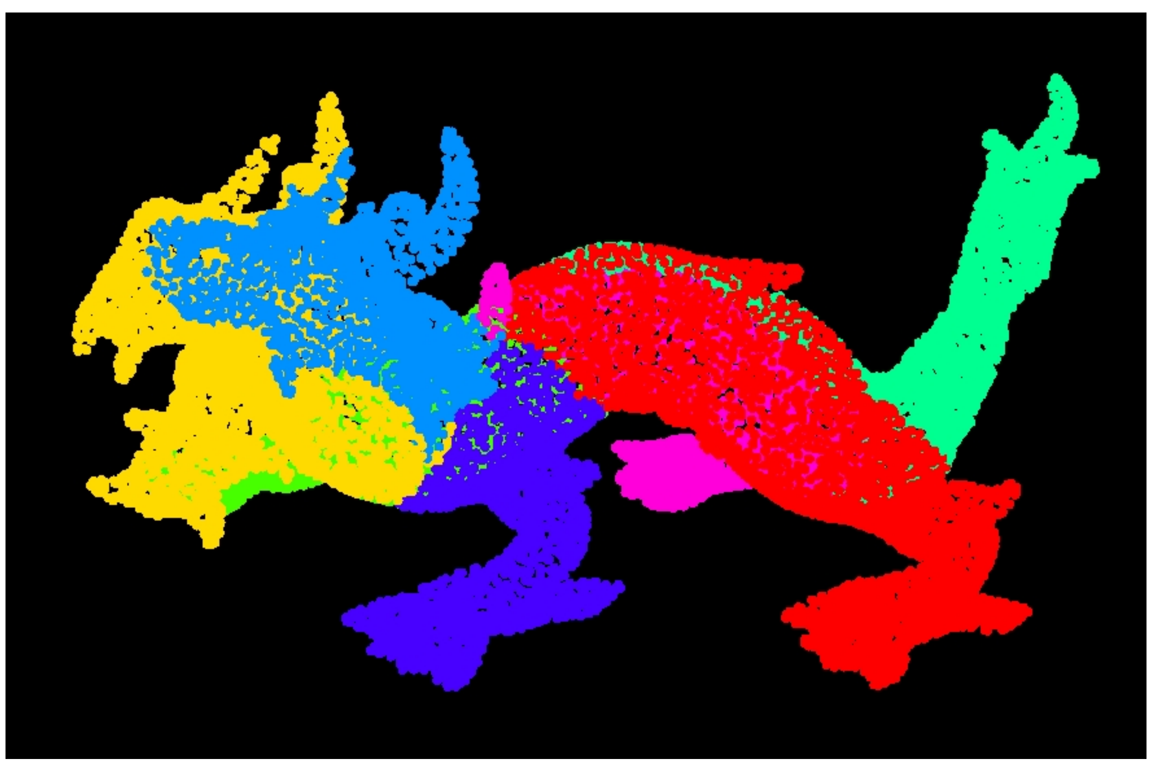
Unstable Example

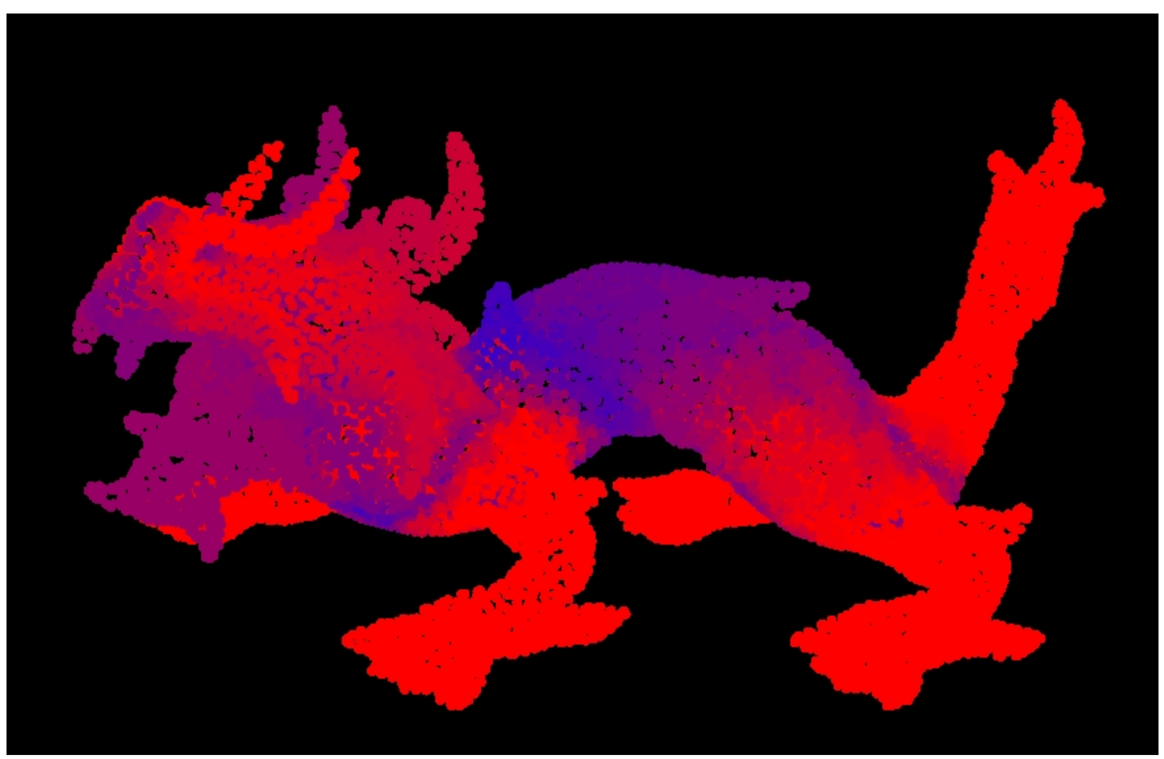






Dragon





(Input courtesy of Maks Ovsanjikov)