Mesh Generation and Topological Data Analysis

Don Sheehy
INRIA Saclay, France

Obvious.

Obvious.

“I could have thought of that.”

Obvious.

"I could have thought of that."

"I should have thought of that."

Outline

Outline

1 **The Obvious.**

Outline

1. **The Obvious.**
   Mesh generation as a preprocess for TDA

Outline

1. **The Obvious.**
   Mesh generation as a preprocess for TDA

2. **The Not Obvious.**

Outline

1. **The Obvious.**
   Mesh generation as a preprocess for TDA

2. **The Not Obvious.**
   Some challenges of mesh generation

Outline

1. **The Obvious.**
   Mesh generation as a preprocess for TDA

2. **The Not Obvious.**
   Some challenges of mesh generation
   (and their solution)

Outline

1. **The Obvious.**
   Mesh generation as a preprocess for TDA

2. **The Not Obvious.**
   Some challenges of mesh generation
   (and their solution)

3. **Some things that are true.**

Outline

1. **The Obvious.**
   Mesh generation as a preprocess for TDA

2. **The Not Obvious.**
   Some challenges of mesh generation
   (and their solution)

3. **Some things that are true.**
   The main theorems about approximating
   persistence diagrams using mesh filtrations

Outline

1 **The Obvious.**

Mesh generation as a preprocess for TDA

2 **The Not Obvious.**

Some challenges of mesh generation
(and their solution)

3 **Some things that are true.**

The main theorems about approximating
persistence diagrams using mesh filtrations

4 **Some things that might be true.**

Outline

1 **The Obvious.**

Mesh generation as a preprocess for TDA

2 **The Not Obvious.**

Some challenges of mesh generation
(and their solution)

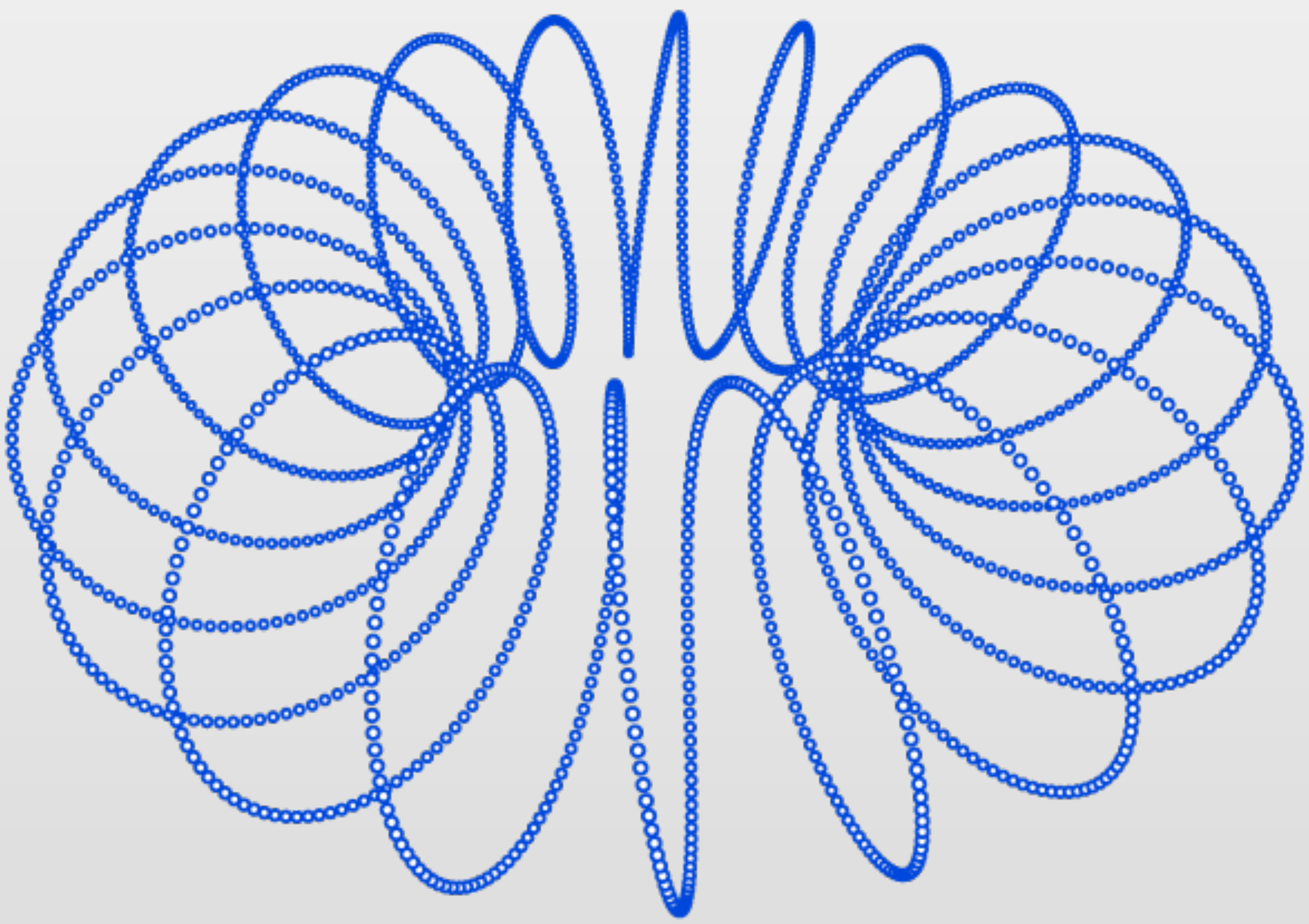
3 **Some things that are true.**

The main theorems about approximating persistence diagrams using mesh filtrations

4 **Some things that might be true.**

Wild speculation.

We consider point clouds in low-dimensional Euclidean space.

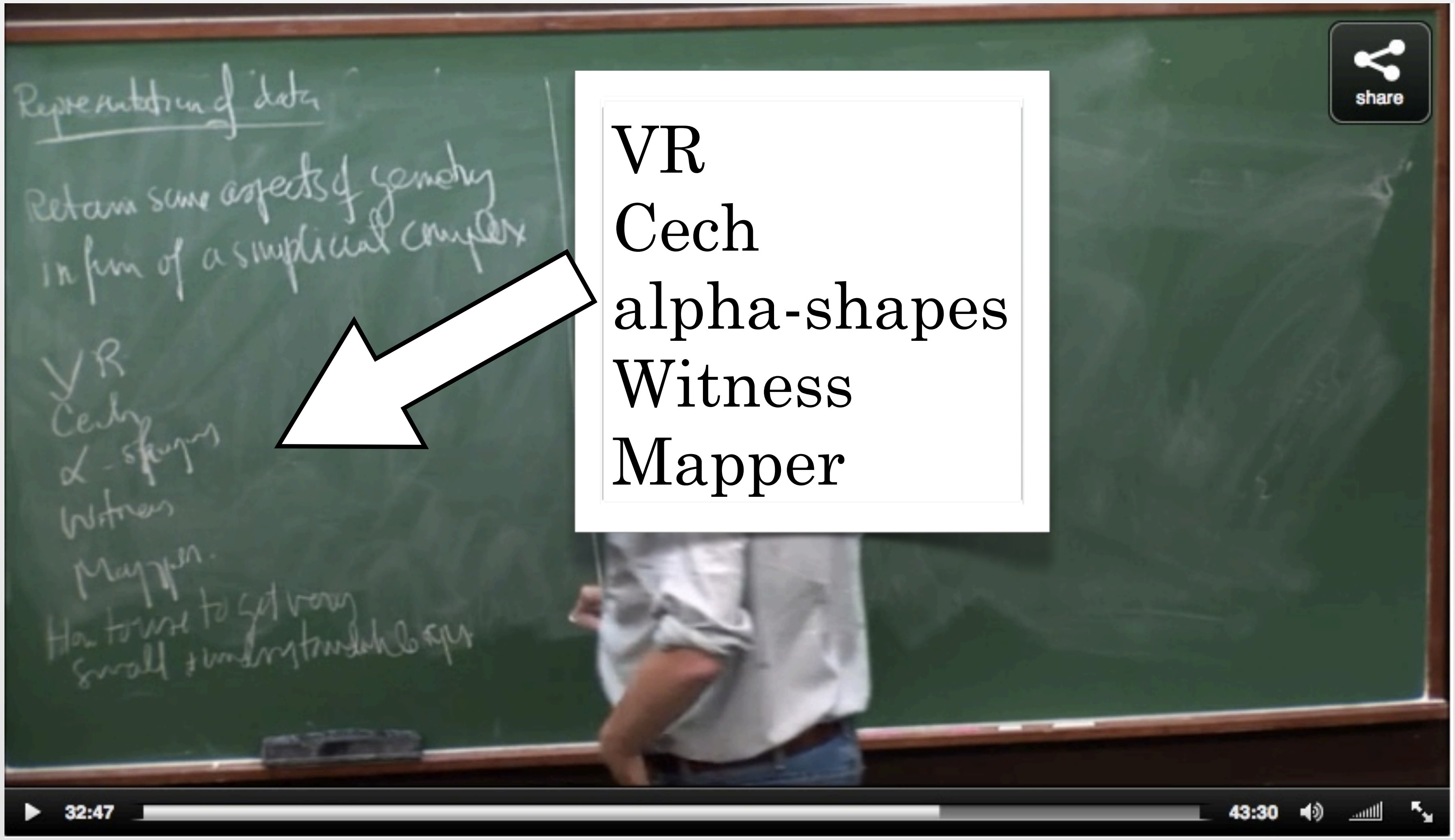


Maybe there is underlying structure, but maybe not.

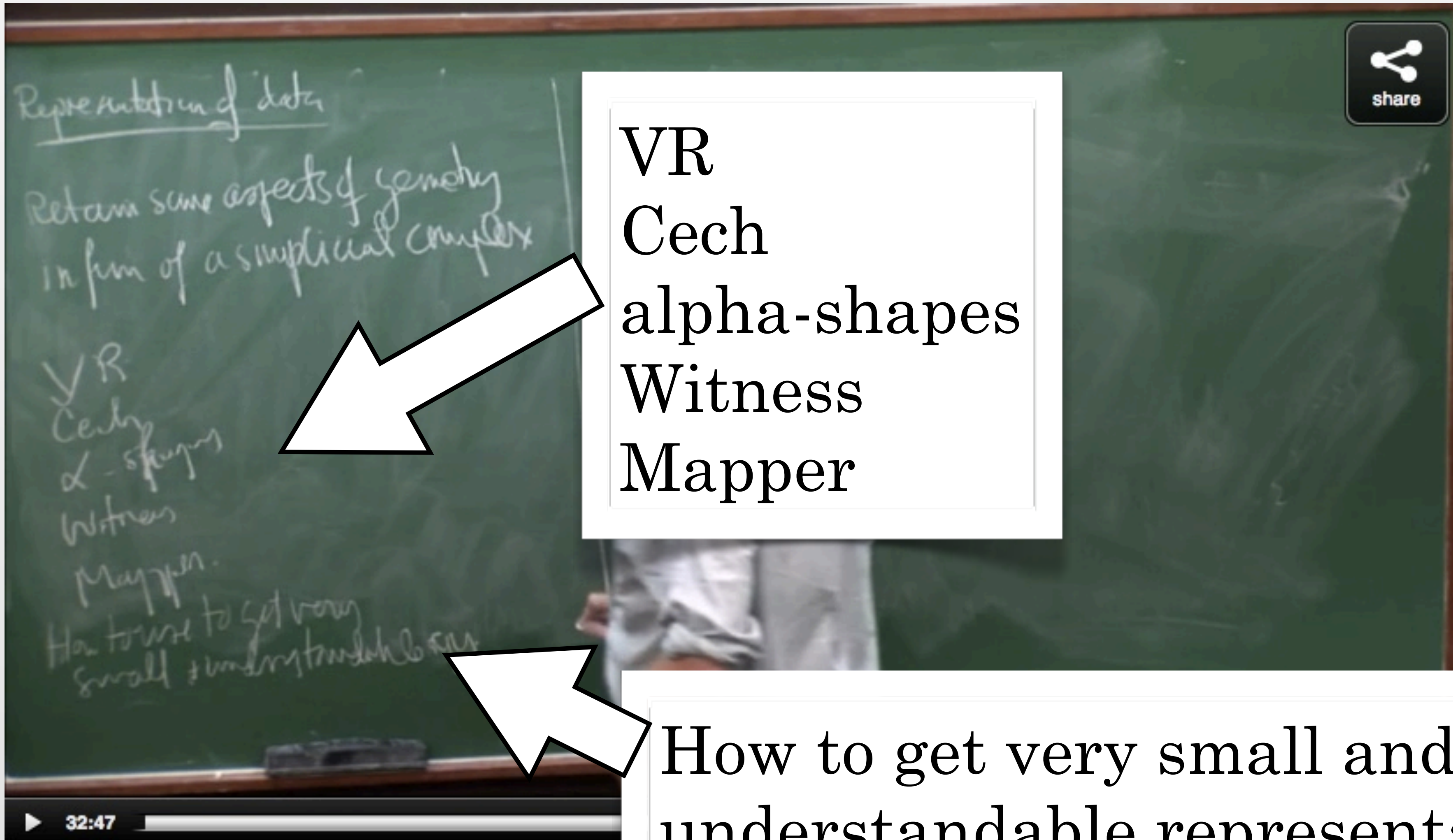
A trip down memory lane...

A trip down memory lane…

A trip down memory lane...



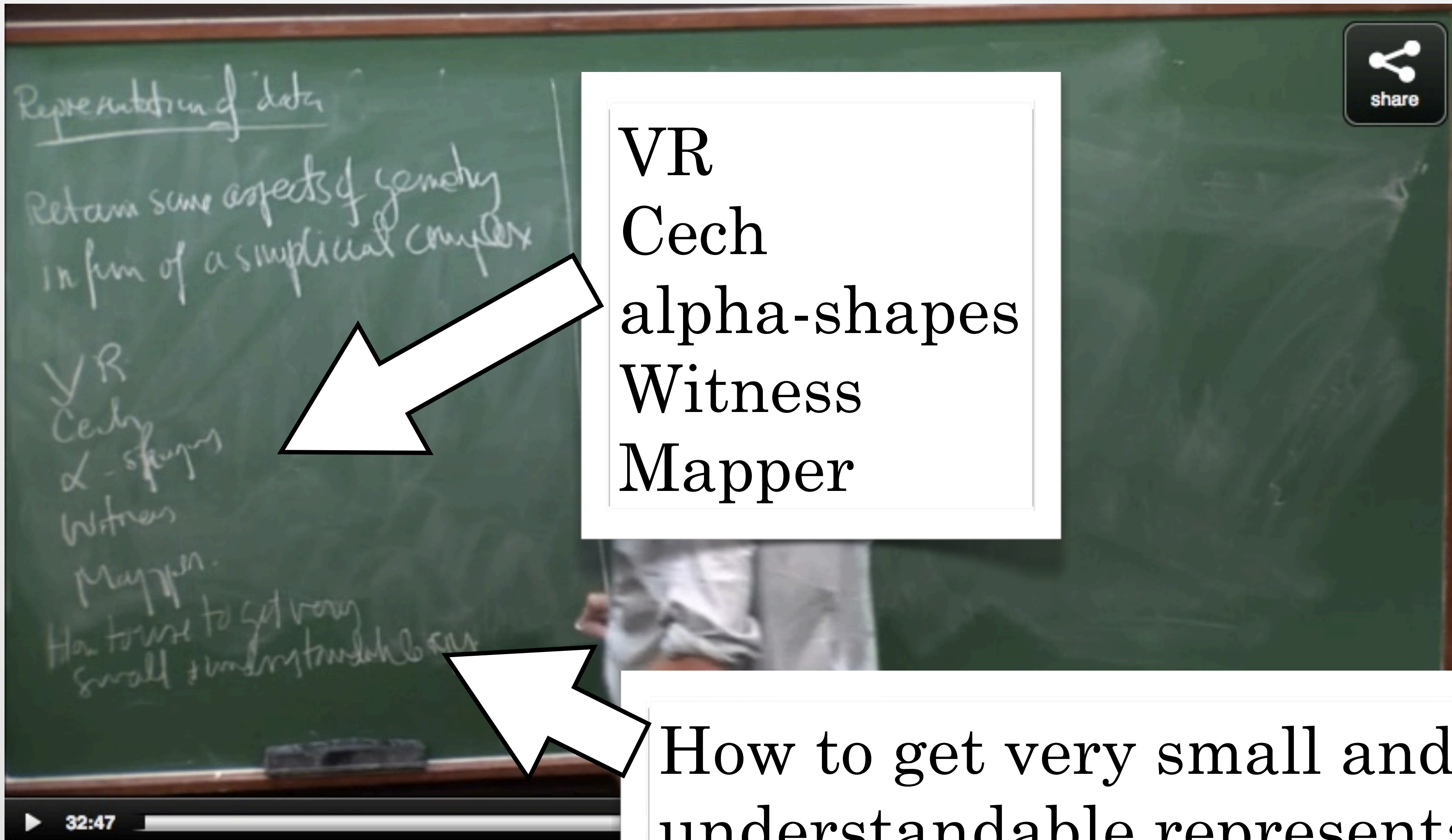
A trip down memory lane...



VR
Cech
alpha-shapes
Witness
Mapper

How to get very small and
understandable representations

A trip down memory lane...

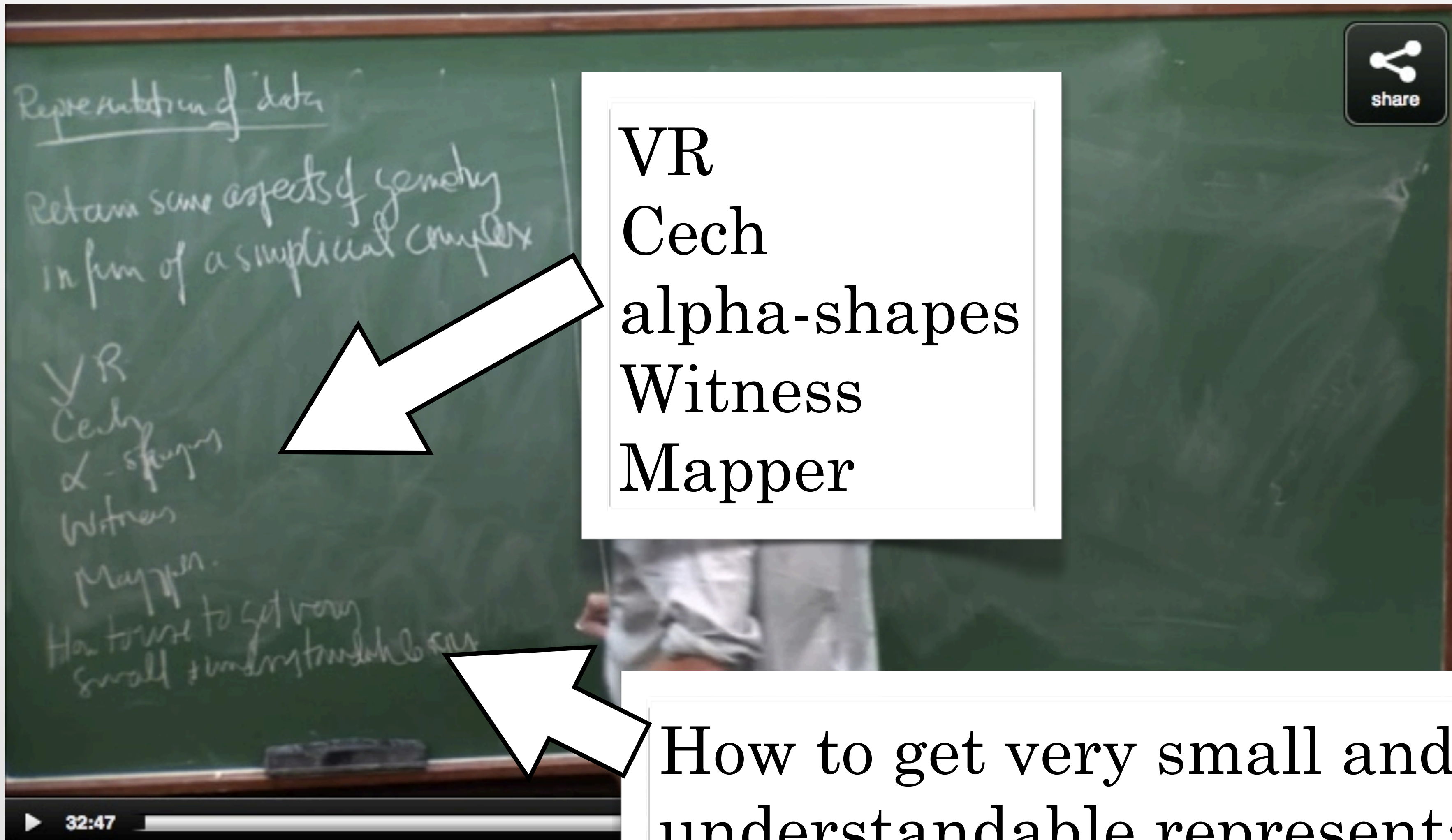


VR
Cech
alpha-shapes
Witness
Mapper

How to get very small and understandable representations

Conventional Wisdom: *If you want a smaller complex, you need to use fewer vertices.*

A trip down memory lane...



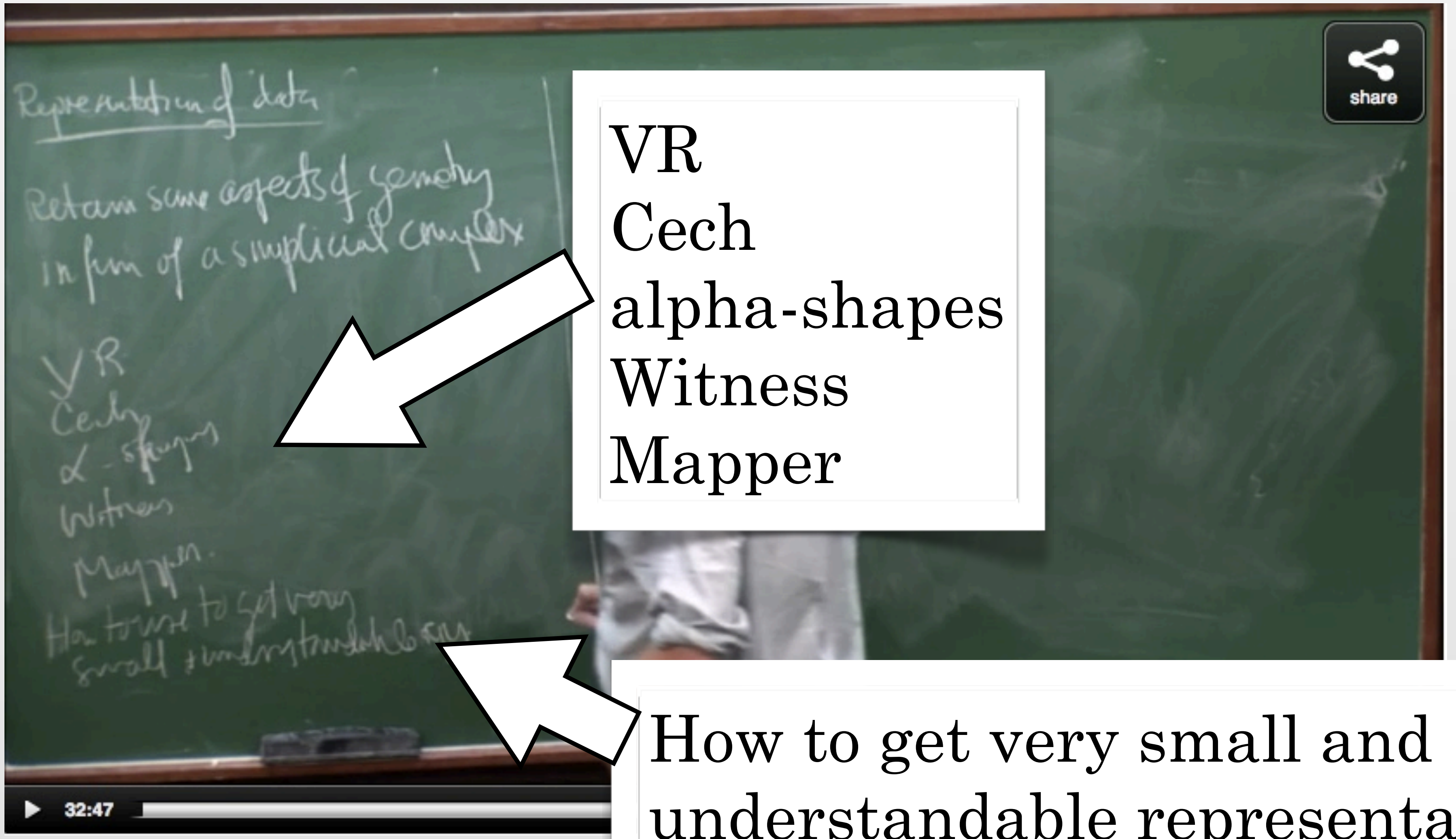
VR
Cech
alpha-shapes
Witness
Mapper

How to get very small and understandable representations

Conventional Wisdom: *If you want a smaller complex, you need to use fewer vertices.*

Obvious?

A trip down memory lane...



VR
Cech
alpha-shapes
Witness
Mapper

How to get very small and understandable representations

Conventional Wisdom: *If you want a smaller complex, you need to use fewer vertices.*

Obvious? True?

The complexity of simplicial complexes is not dominated by vertex counts.

The complexity of simplicial complexes is not dominated by vertex counts.

For simplicial polytopes: $\Omega(n) \leq \text{number of faces} \leq O(n^{\lfloor d/2 \rfloor})$

The complexity of simplicial complexes is not dominated by vertex counts.

For simplicial polytopes: $\Omega(n) \leq \text{number of faces} \leq O(n^{\lfloor d/2 \rfloor})$

For α-complexes: $\Omega(n) \leq \text{number of faces} \leq O(n^{\lceil d/2 \rceil})$

The complexity of simplicial complexes is not dominated by vertex counts.

For simplicial polytopes: $\Omega(n) \leq \text{number of faces} \leq O(n^{\lfloor d/2 \rfloor})$

For α-complexes: $\Omega(n) \leq \text{number of faces} \leq O(n^{\lceil d/2 \rceil})$

Geometry matters!

The complexity of simplicial complexes is not dominated by vertex counts.

For simplicial polytopes: $\Omega(n) \leq \text{number of faces} \leq O(n^{\lfloor d/2 \rfloor})$

For α-complexes: $\Omega(n) \leq \text{number of faces} \leq O(n^{\lceil d/2 \rceil})$

Geometry matters!

Example:
Delaunay triangulation of points sampled from 2 skew lines.

The complexity of simplicial complexes is not dominated by vertex counts.

For simplicial polytopes: $\Omega(n) \leq \text{number of faces} \leq O(n^{\lfloor d/2 \rfloor})$

For α -complexes: $\Omega(n) \leq \text{number of faces} \leq O(n^{\lceil d/2 \rceil})$

Geometry matters!

Example:
Delaunay triangulation of points sampled from 2 skew lines.

Complexity: $O(n^2)$

The complexity of simplicial complexes is not dominated by vertex counts.

For simplicial polytopes: $\Omega(n) \leq \text{number of faces} \leq O(n^{\lfloor d/2 \rfloor})$

For α-complexes: $\Omega(n) \leq \text{number of faces} \leq O(n^{\lceil d/2 \rceil})$

Geometry matters!

Example:

Delaunay triangulation of points sampled from 2 skew lines.

Complexity: $O(n^2)$

Complexity with noise: $O(n)$

The complexity of simplicial complexes is not dominated by vertex counts.

For simplicial polytopes: $\Omega(n) \leq \text{number of faces} \leq O(n^{\lfloor d/2 \rfloor})$

For α-complexes: $\Omega(n) \leq \text{number of faces} \leq O(n^{\lceil d/2 \rceil})$

Geometry matters!

Example:
Delaunay triangulation of points sampled from 2 skew lines.

Complexity: $O(n^2)$

Complexity with noise: $O(n)$

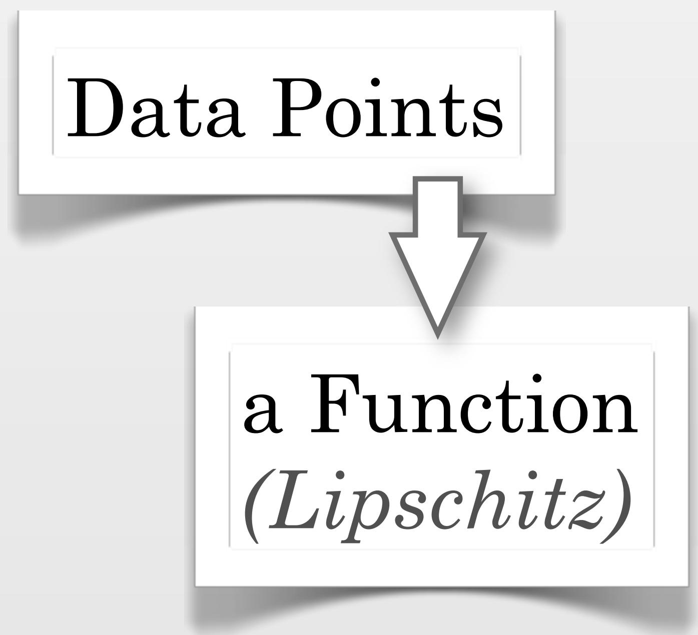
Does noise help?

A TDA Pipeline

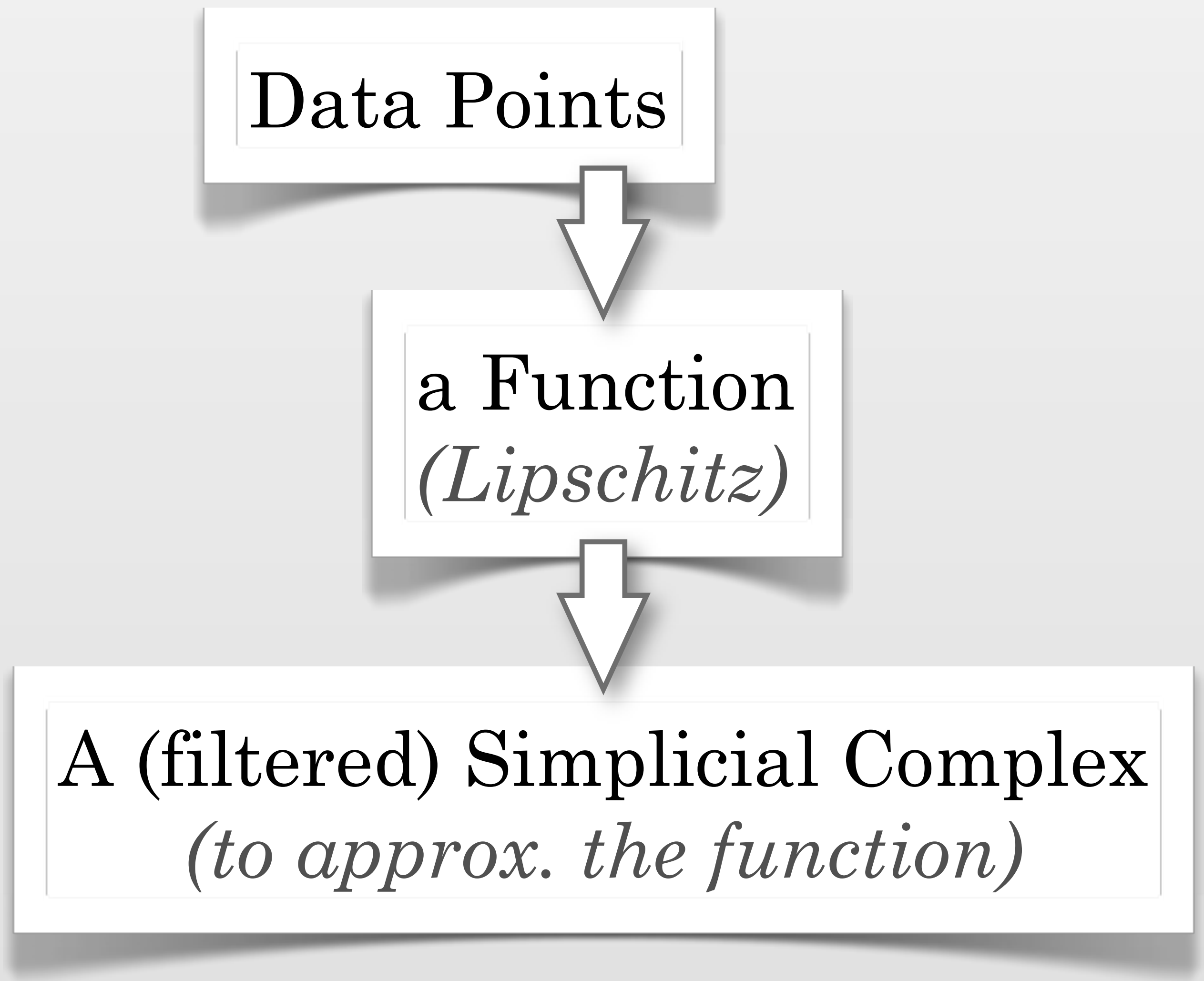
A TDA Pipeline

Data Points

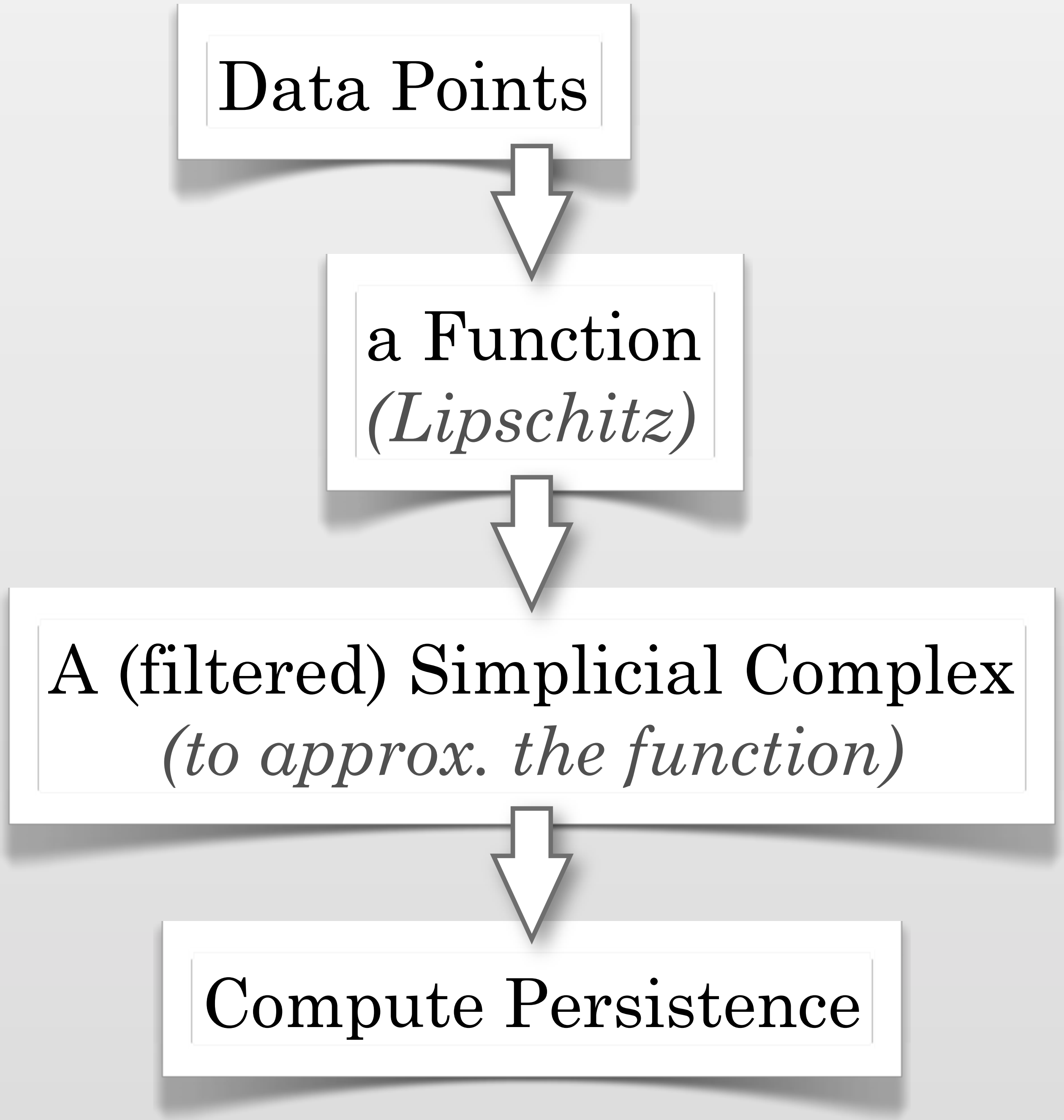
A TDA Pipeline



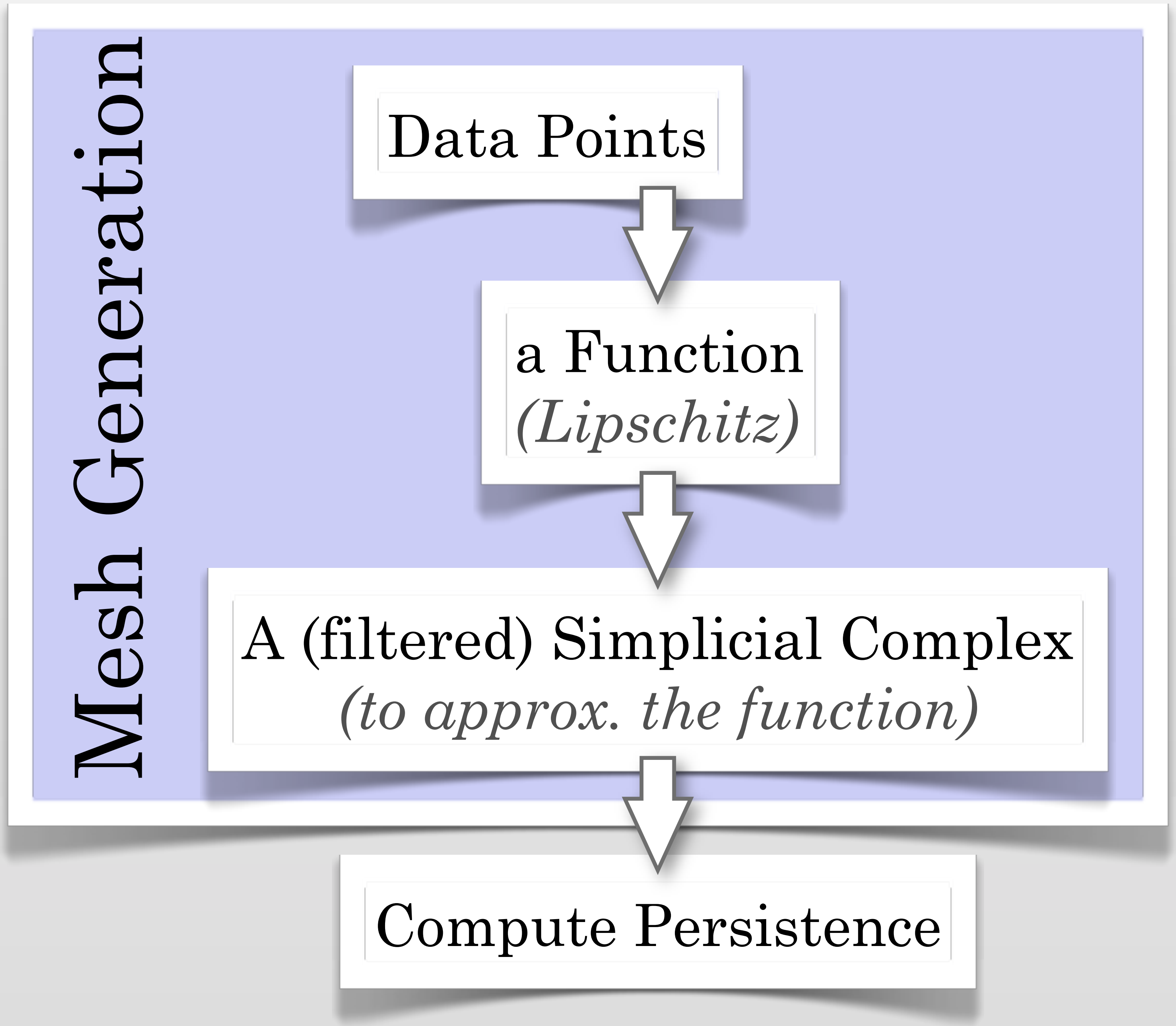
A TDA Pipeline



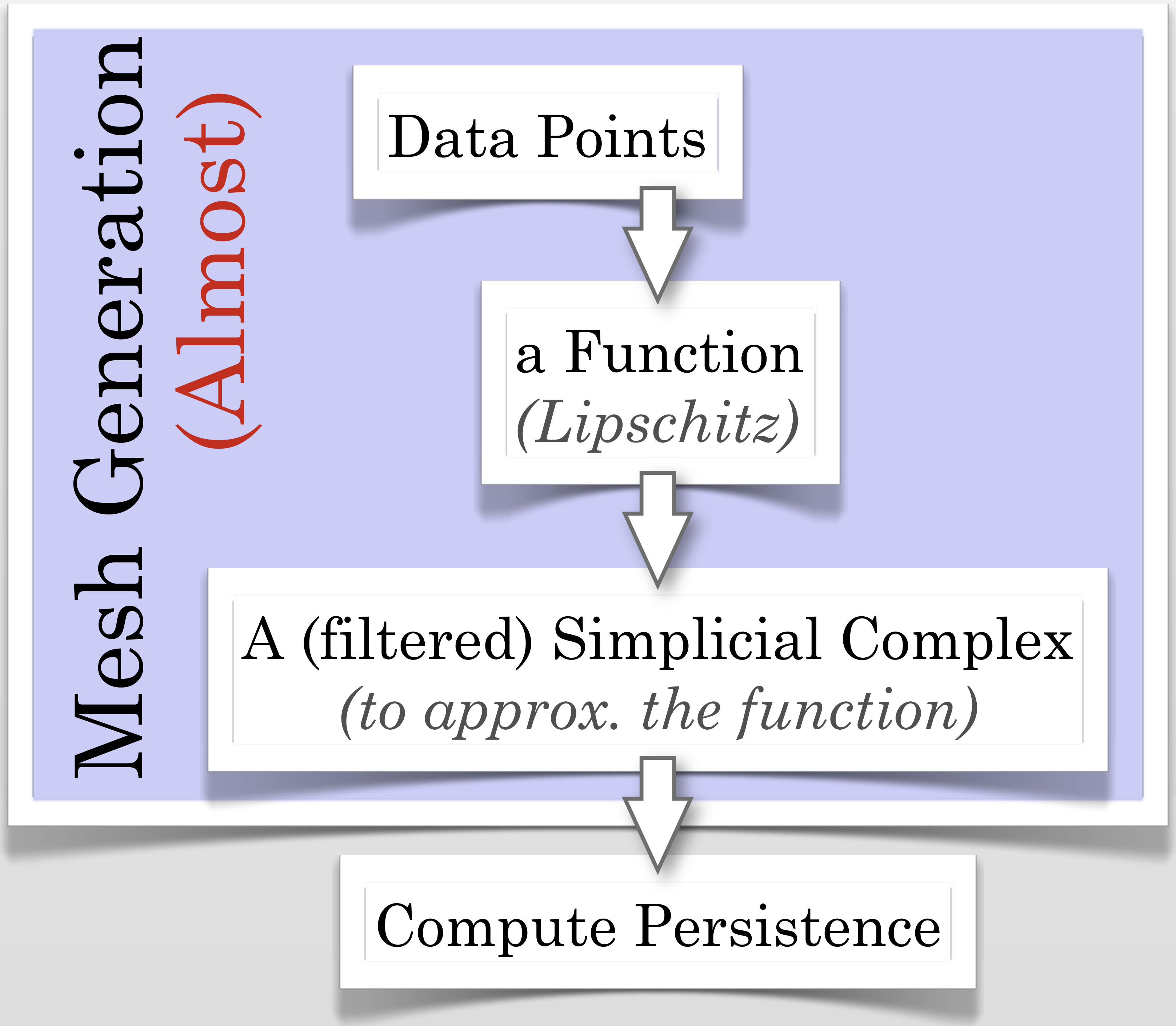
A TDA Pipeline



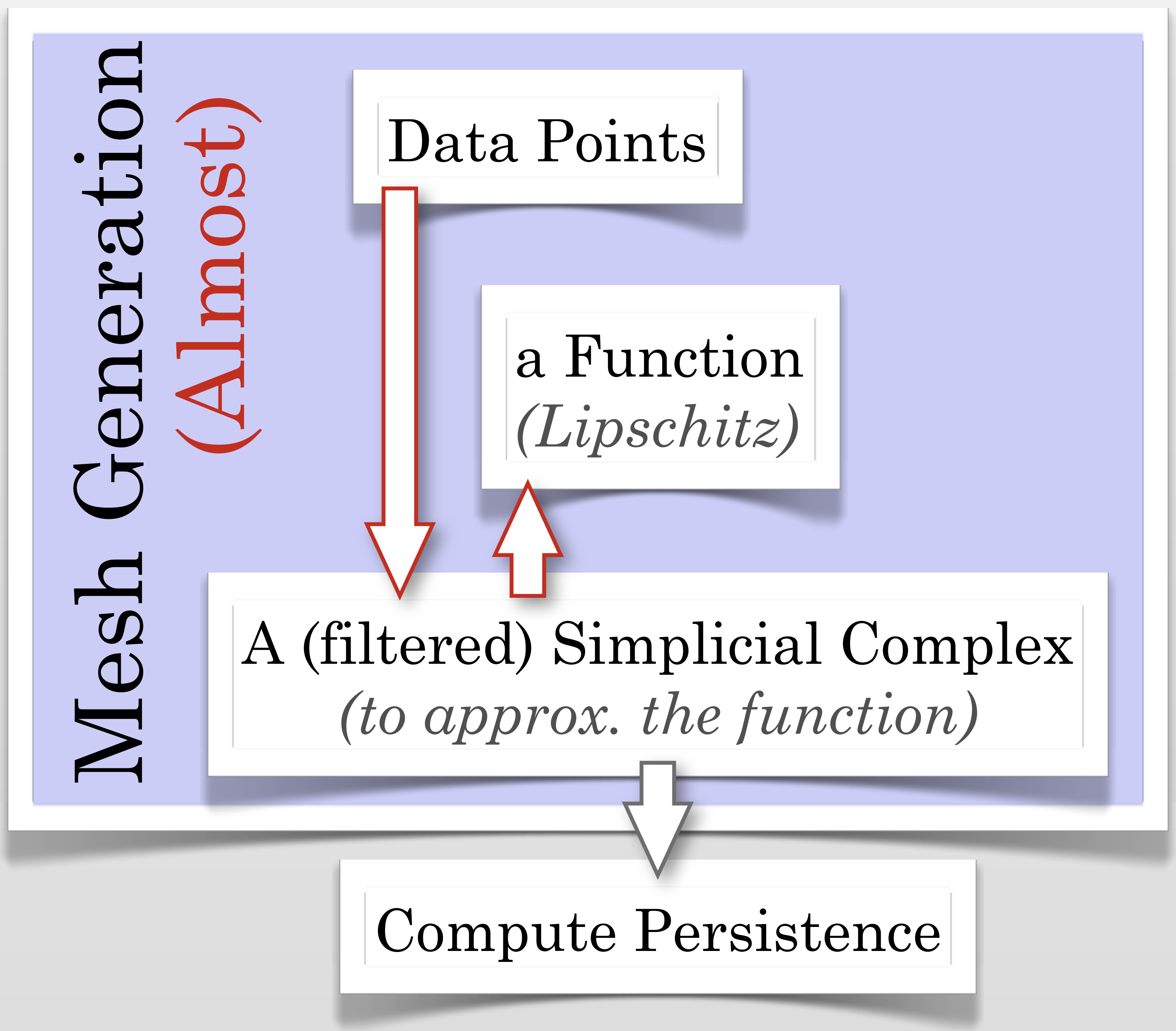
A TDA Pipeline



A TDA Pipeline



A TDA Pipeline



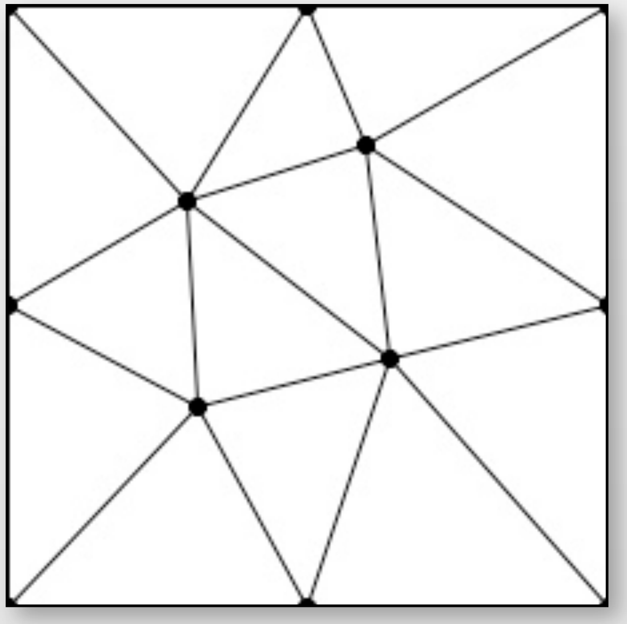
Mesh Generation

Mesh Generation

Decompose a domain into simple elements.

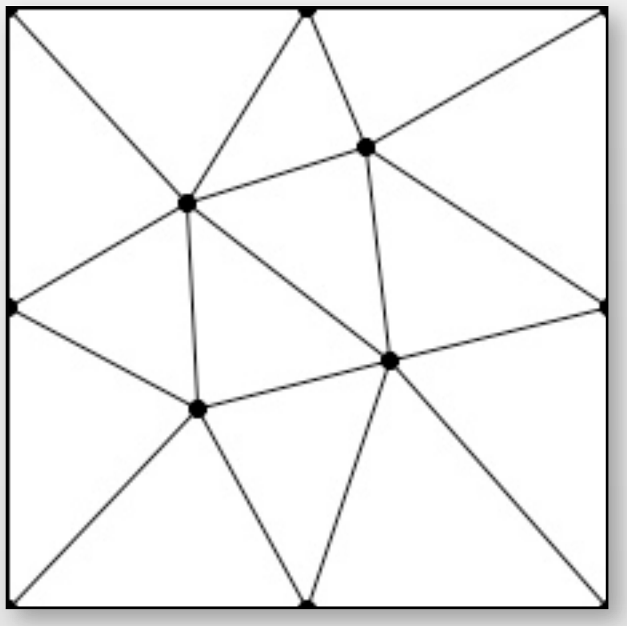
Mesh Generation

Decompose a domain into simple elements.

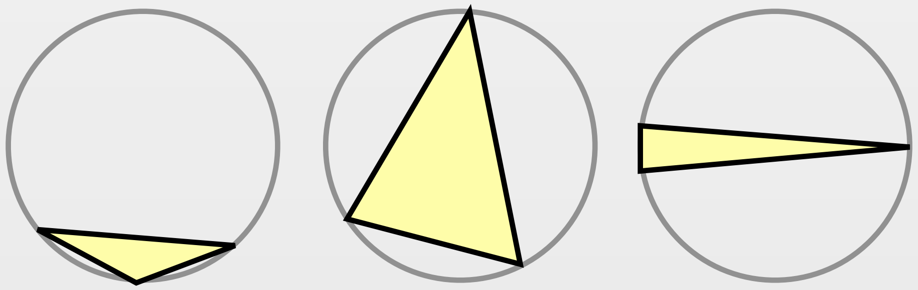


Mesh Generation

Decompose a domain into simple elements.



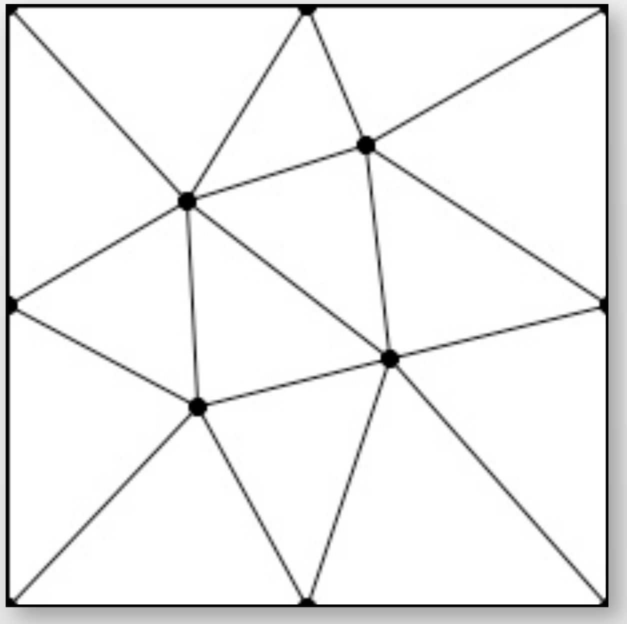
Mesh Quality



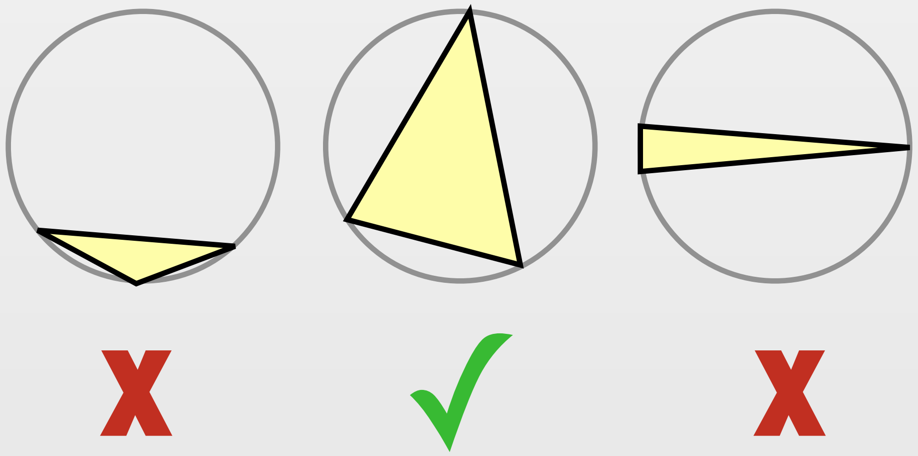
Radius/Edge < const

Mesh Generation

Decompose a domain into simple elements.



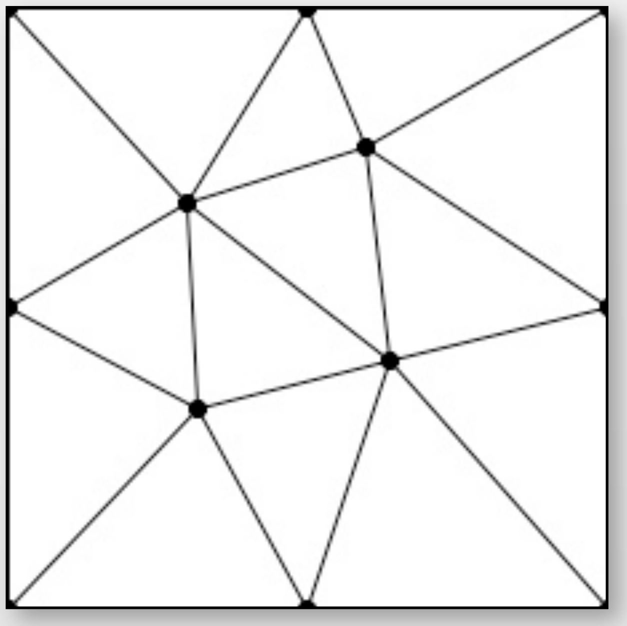
Mesh Quality



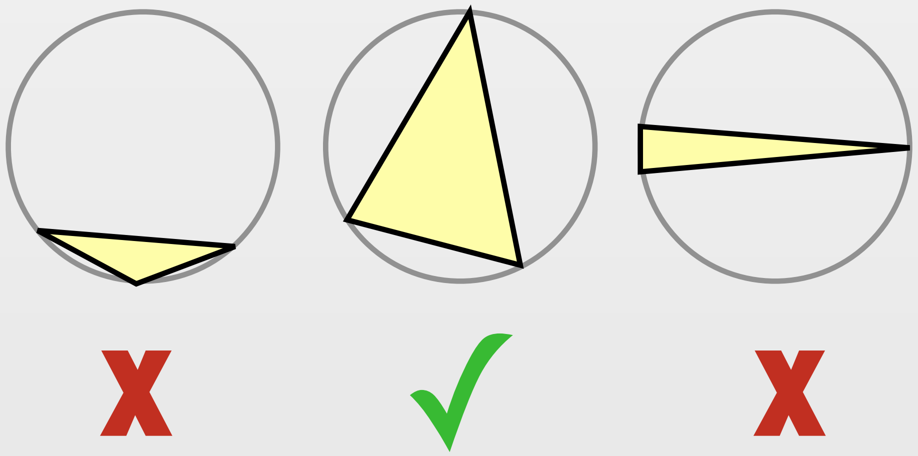
Radius/Edge < const

Mesh Generation

Decompose a domain into simple elements.



Mesh Quality

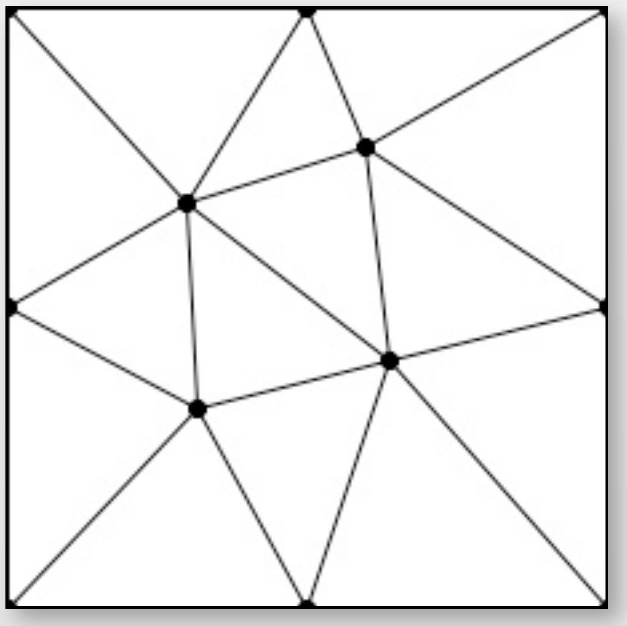


Radius/Edge < const

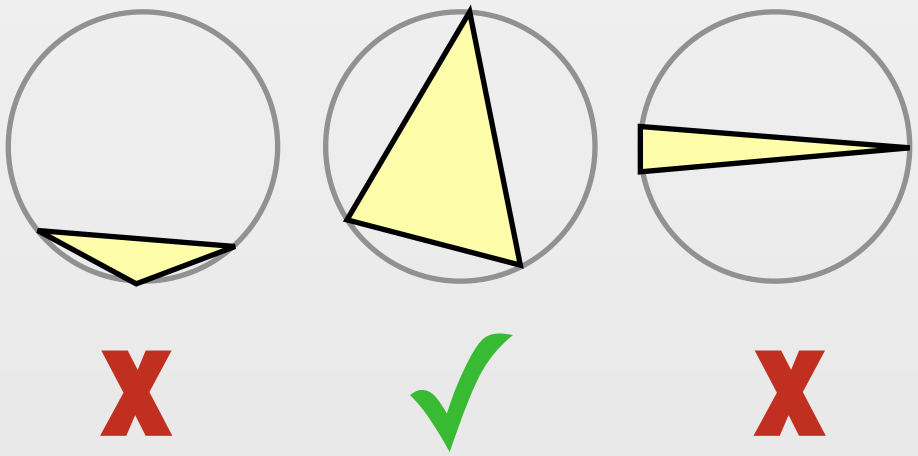
Conforming to Input

Mesh Generation

Decompose a domain into simple elements.

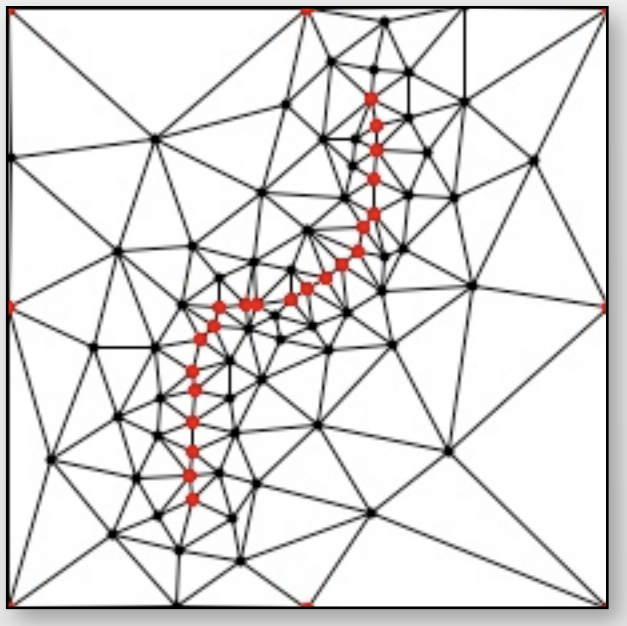


Mesh Quality



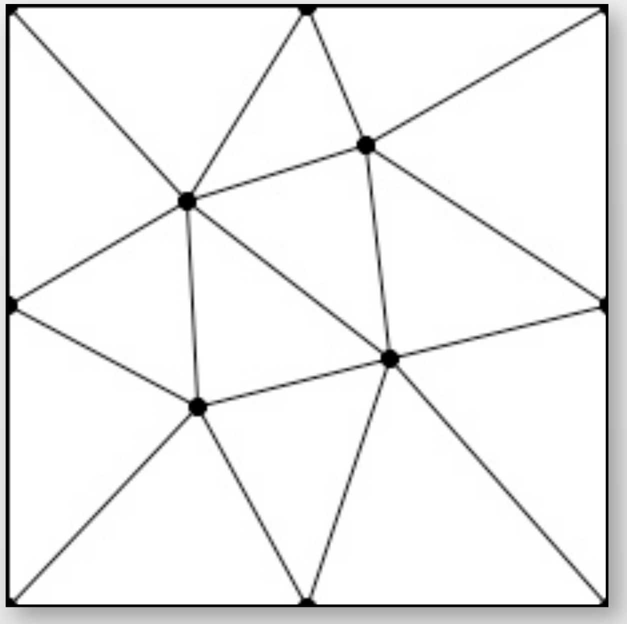
Radius/Edge < const

Conforming to Input

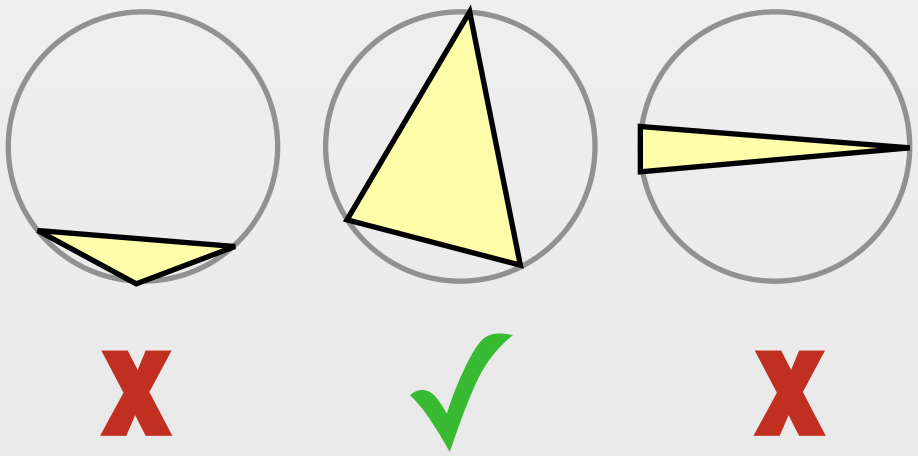


Mesh Generation

Decompose a domain into simple elements.

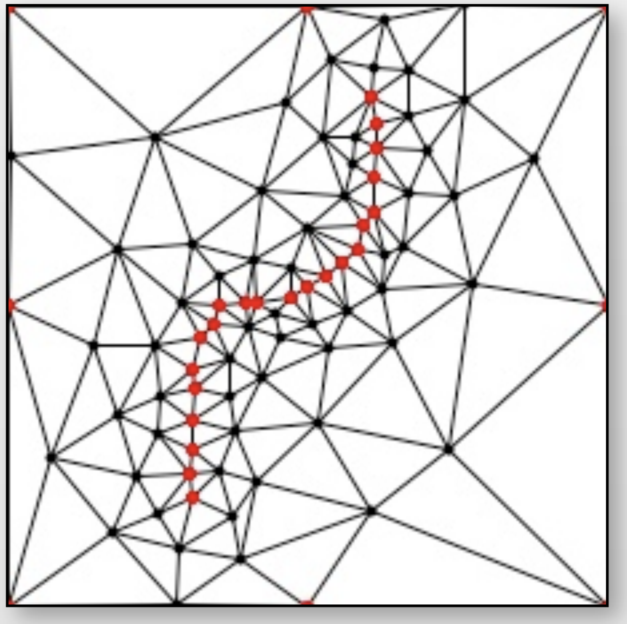


Mesh Quality



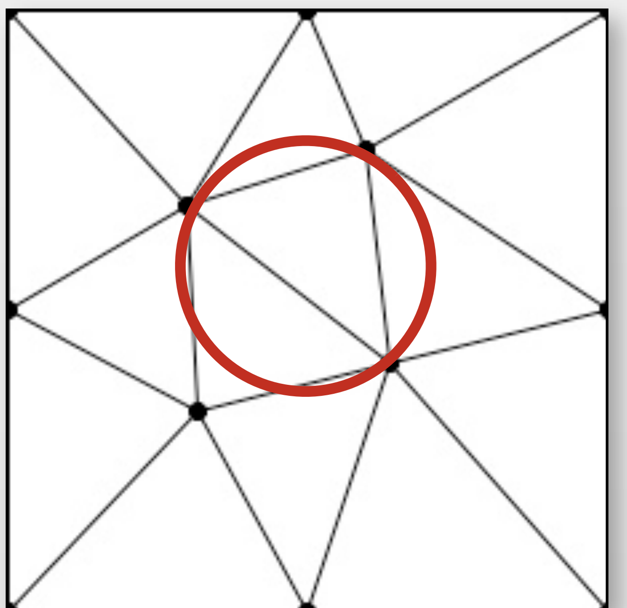
Radius/Edge < const

Conforming to Input

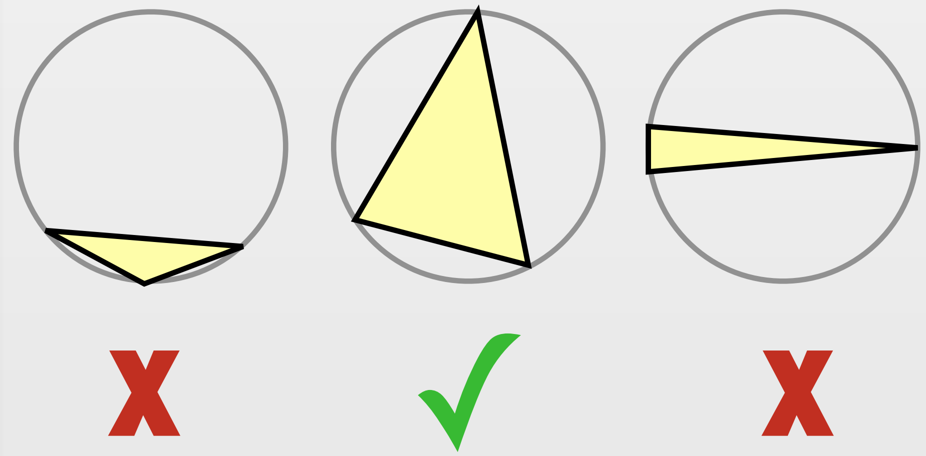


Mesh Generation

Decompose a domain into simple elements.

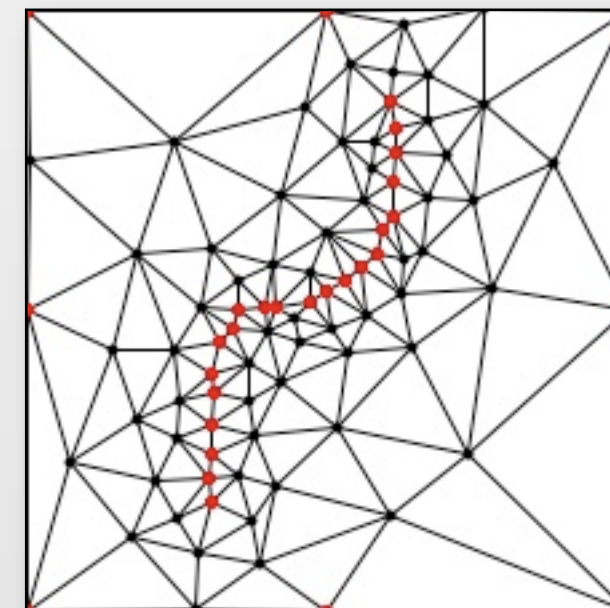


Mesh Quality



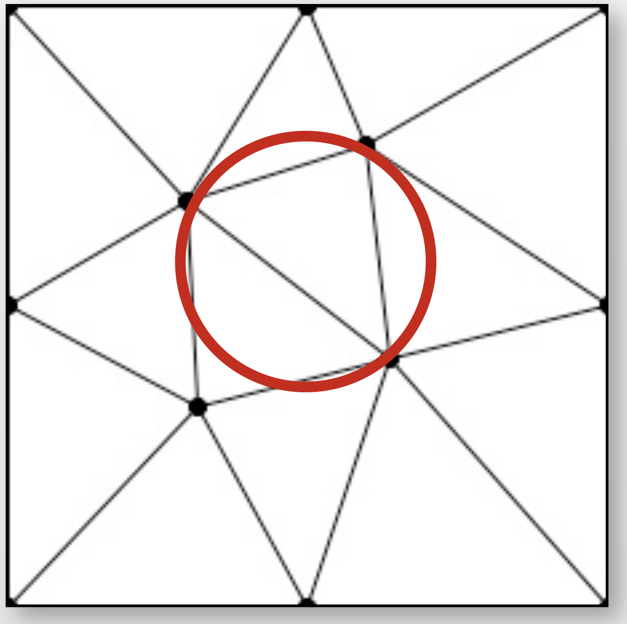
Radius/Edge < const

Conforming to Input

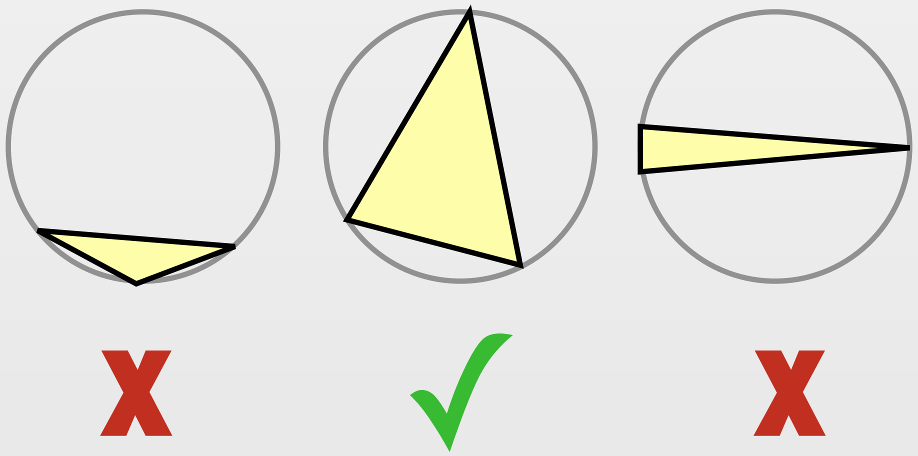


Mesh Generation

Decompose a domain into simple elements.

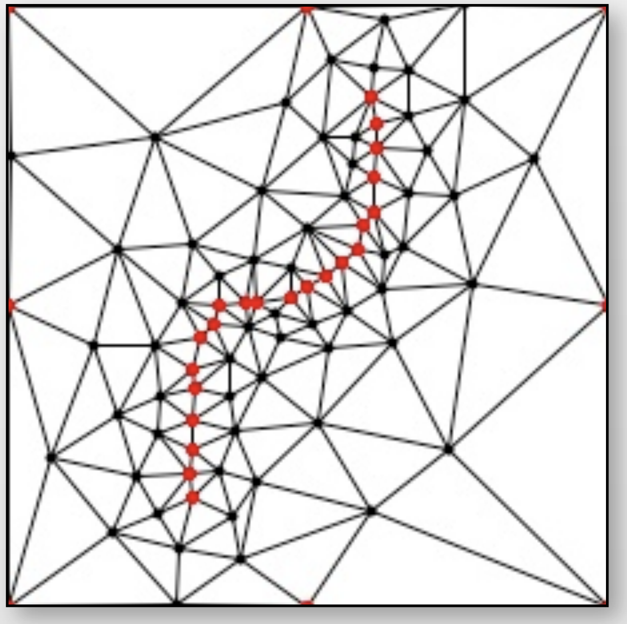


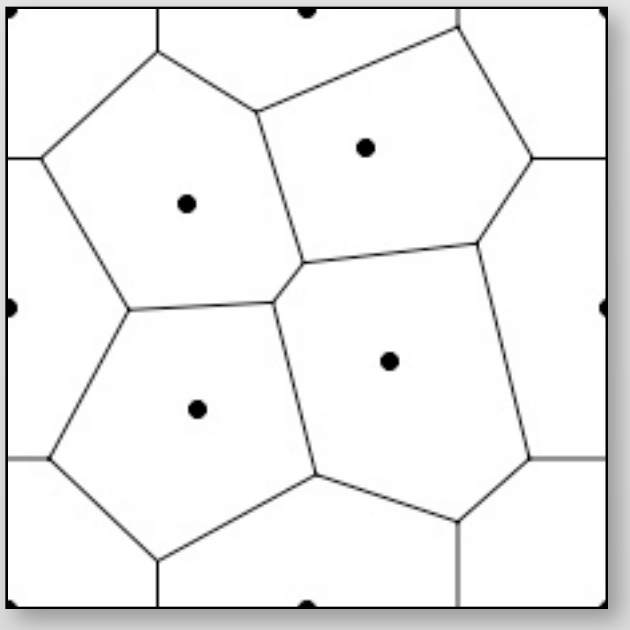
Mesh Quality



Radius/Edge < const

Conforming to Input

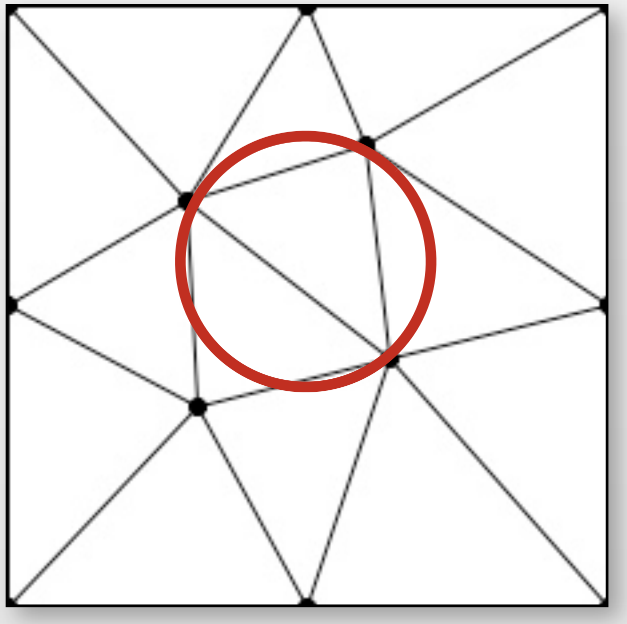




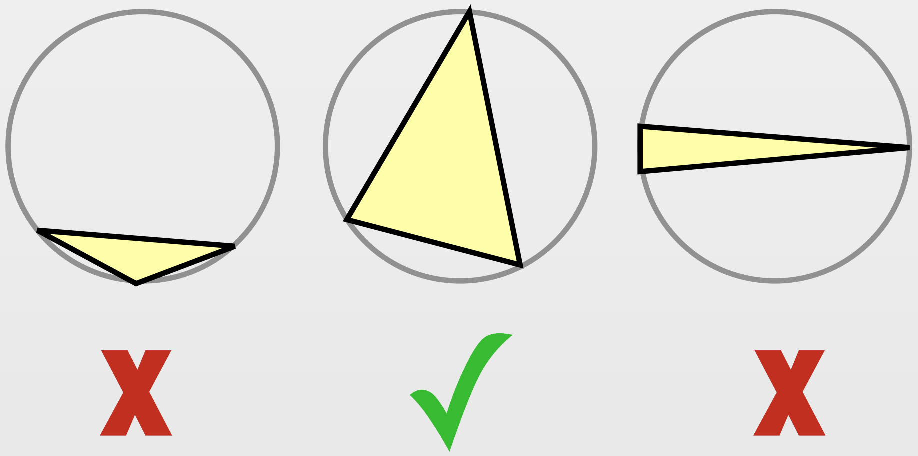
Voronoi Diagram

Mesh Generation

Decompose a domain into simple elements.

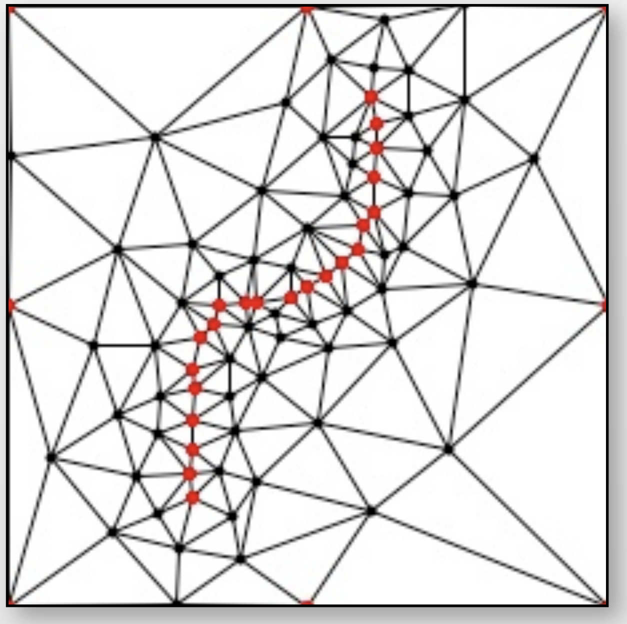


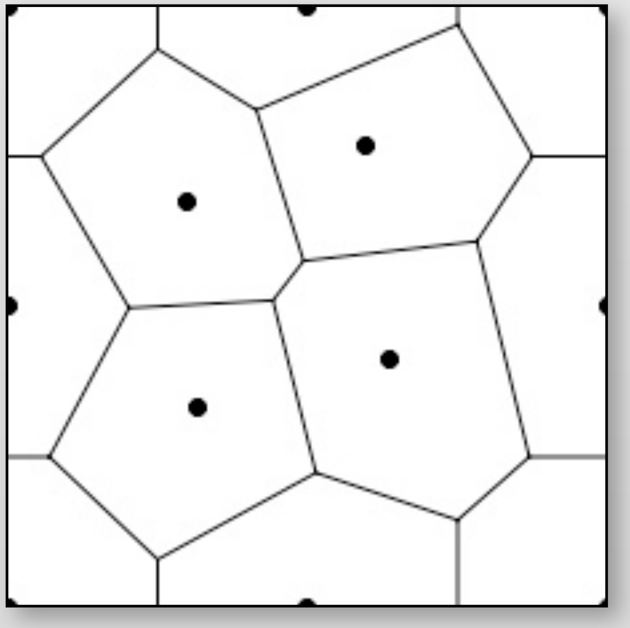
Mesh Quality



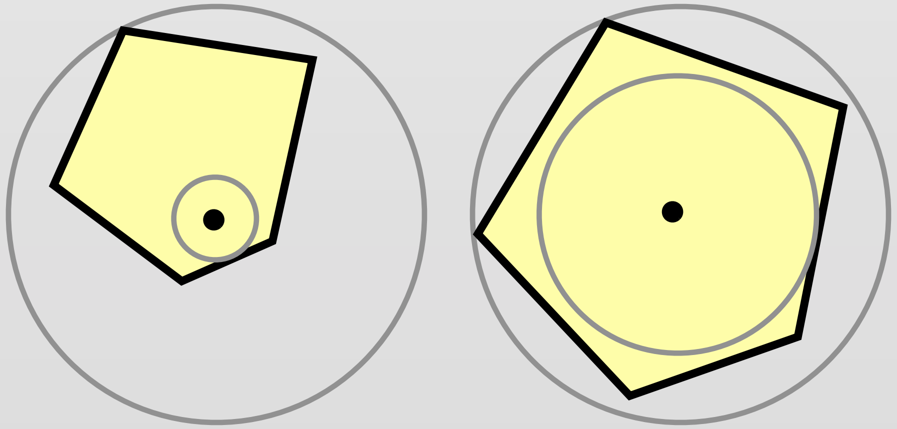
Radius/Edge < const

Conforming to Input





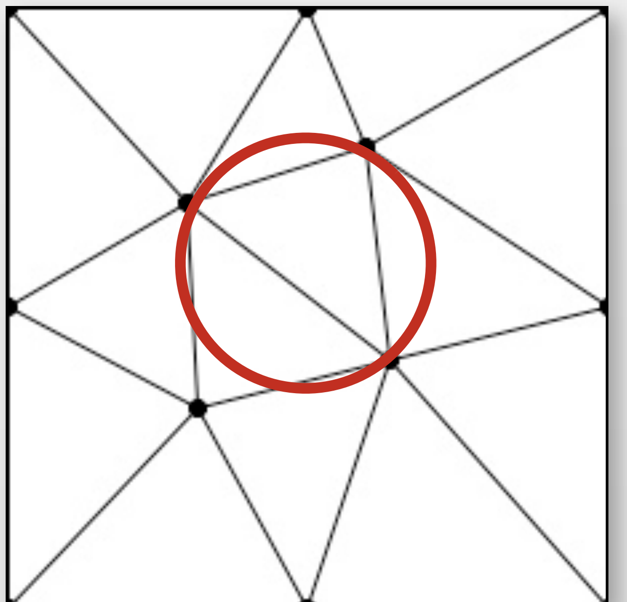
Voronoi Diagram



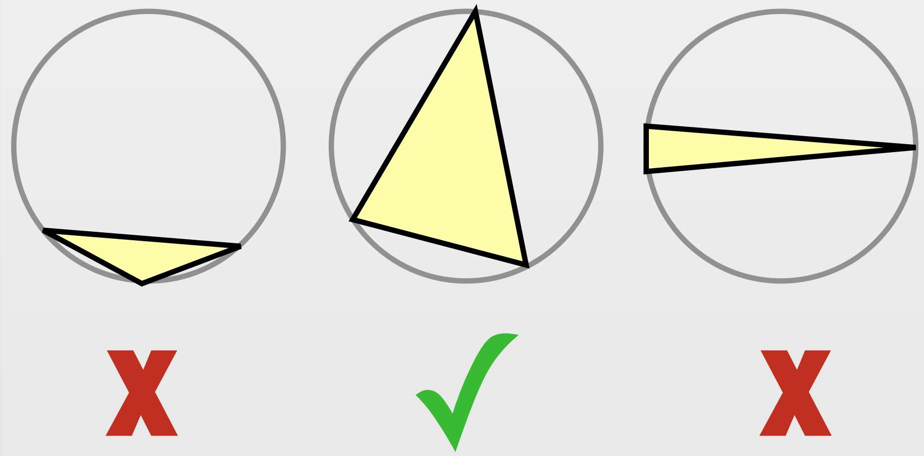
OutRadius/InRadius < const

Mesh Generation

Decompose a domain into simple elements.

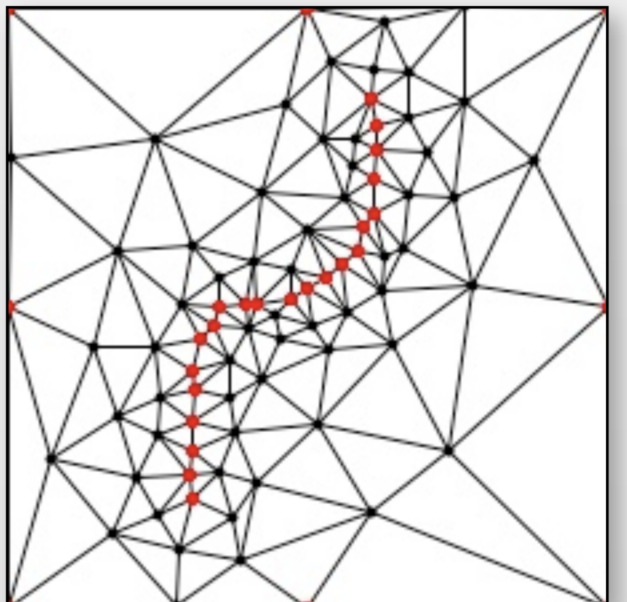


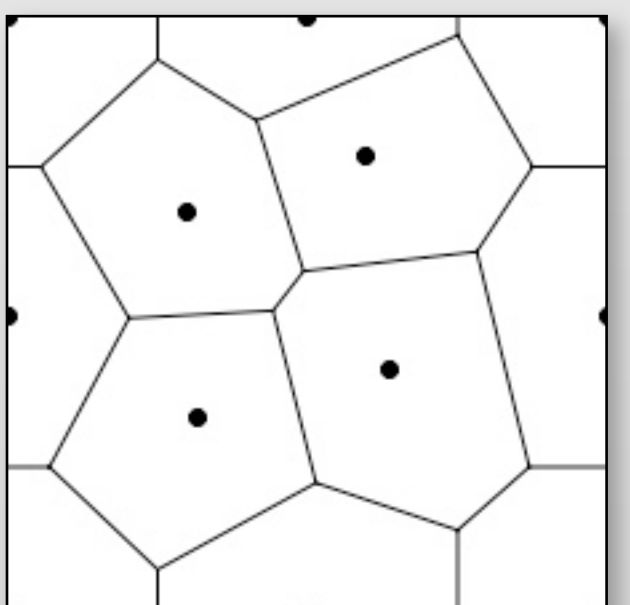
Mesh Quality



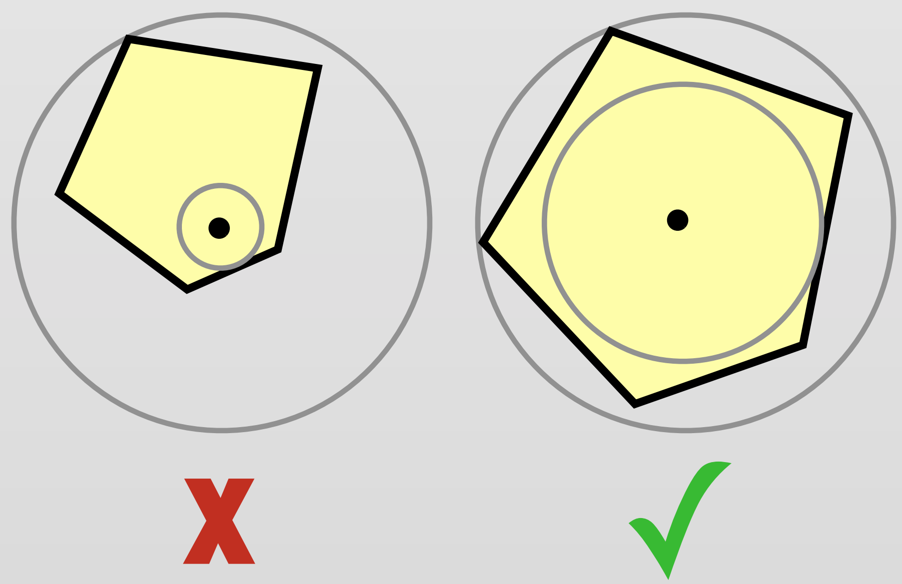
Radius/Edge < const

Conforming to Input





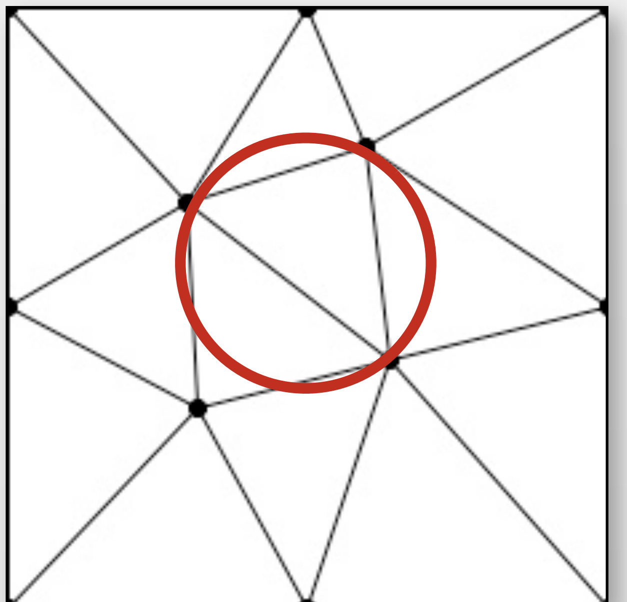
Voronoi Diagram



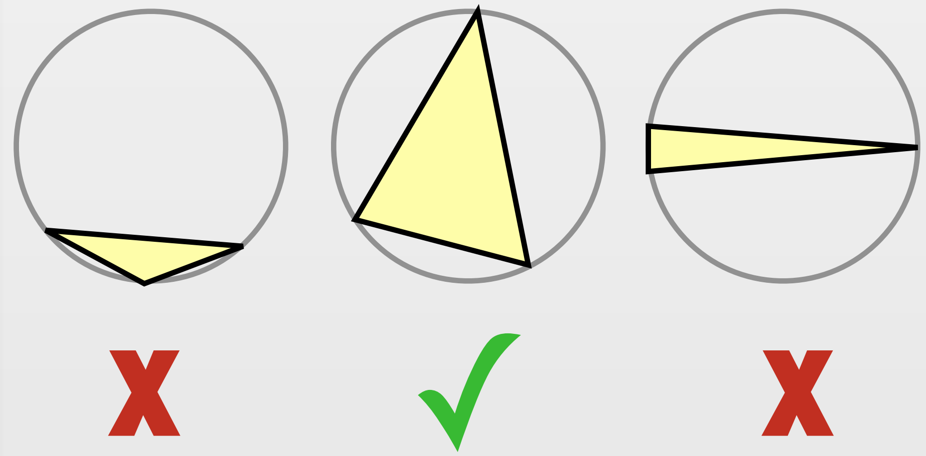
OutRadius/InRadius < const

Mesh Generation

Decompose a domain into simple elements.

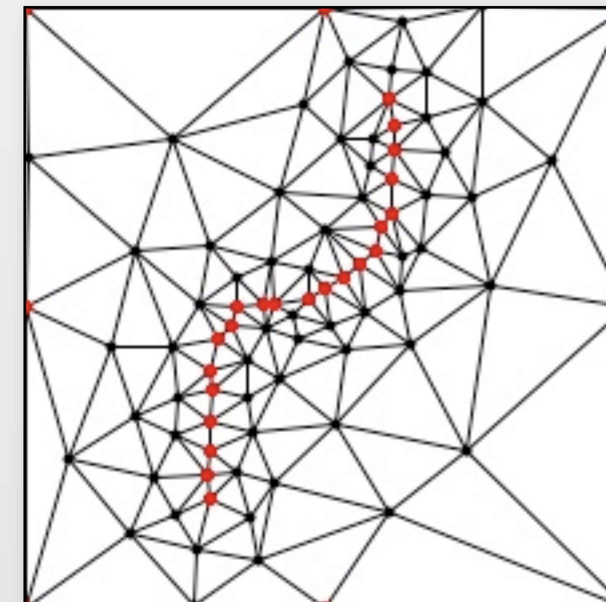


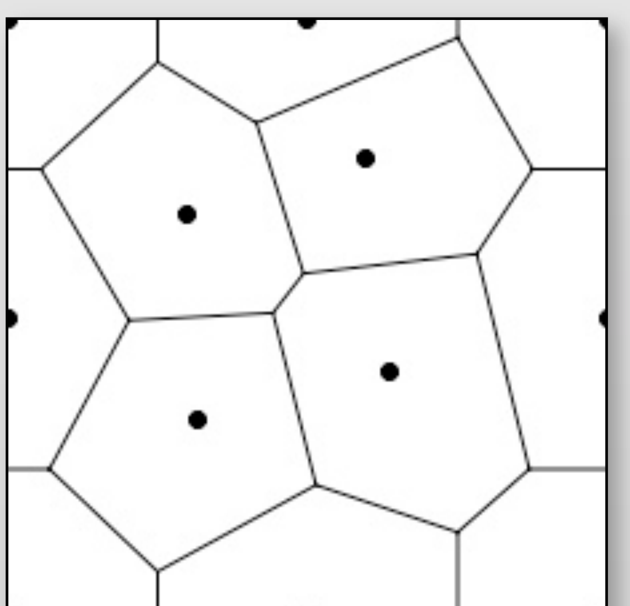
Mesh Quality



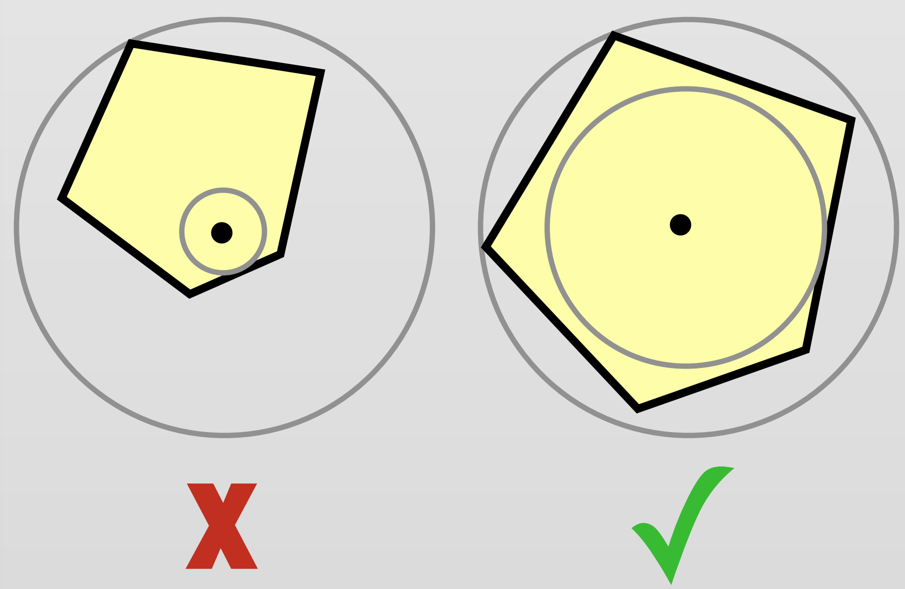
Radius/Edge < const

Conforming to Input

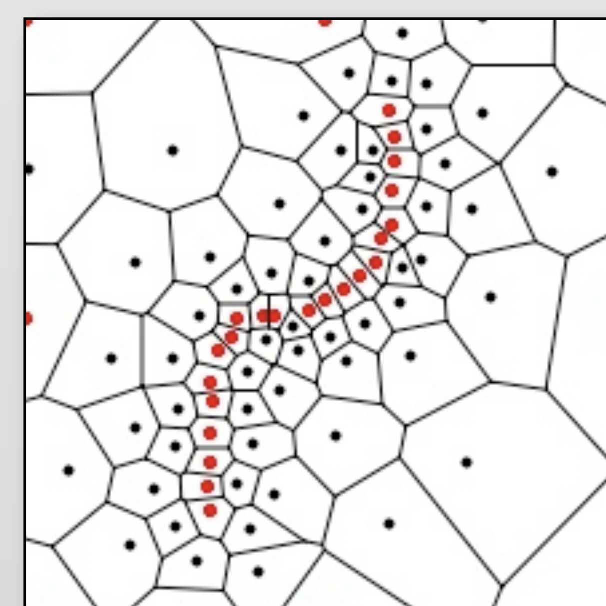




Voronoi Diagram



OutRadius/InRadius < const



Meshing Points

Input: $P \subset \mathbb{R}^d$

Output: $M \supset P$ with a "nice" Voronoi diagram

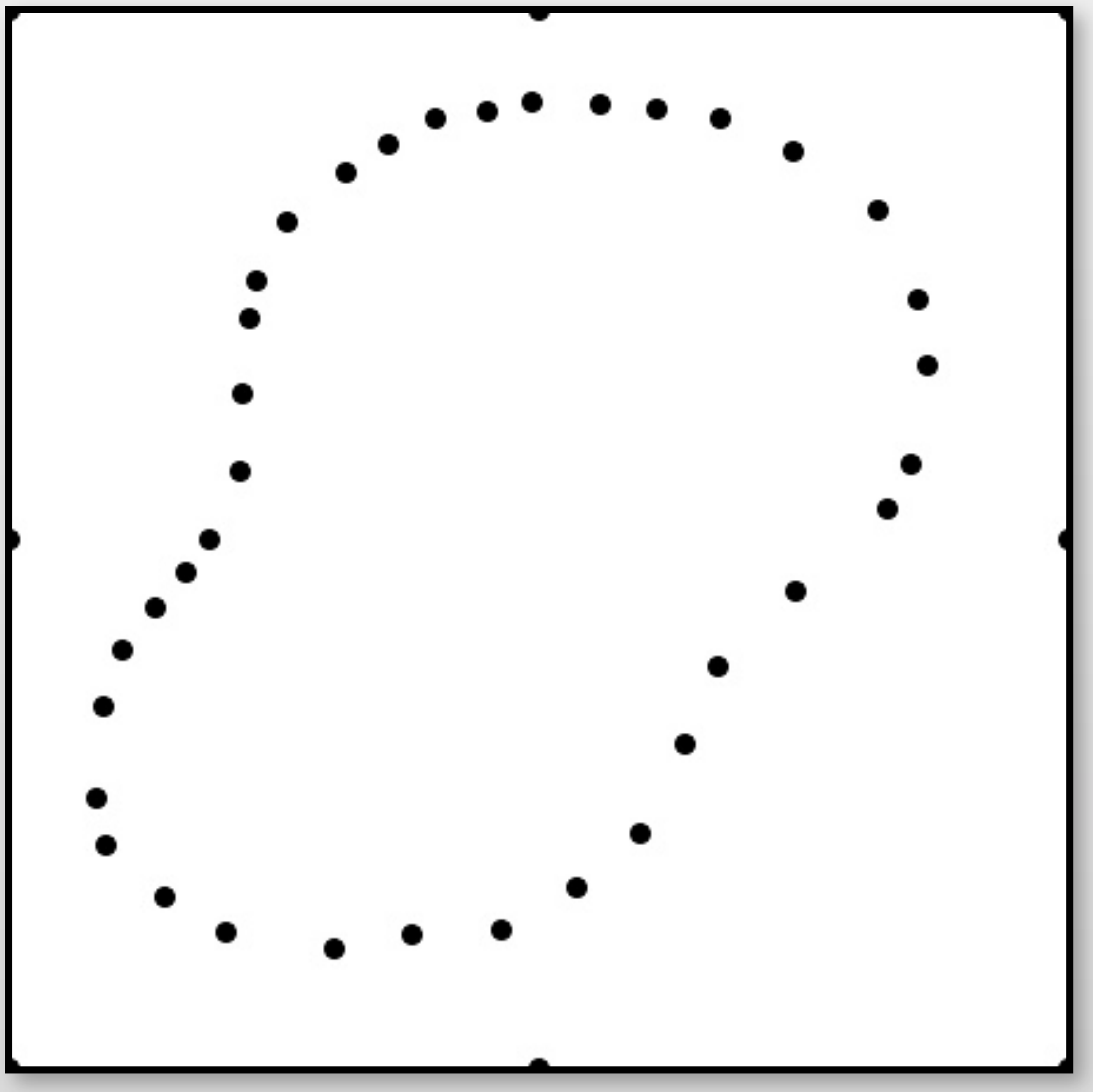
$n = |P|, m = |M|$

Meshing Points

Input: $P \subset \mathbb{R}^d$

Output: $M \supset P$ with a "nice" Voronoi diagram

$n = |P|, m = |M|$

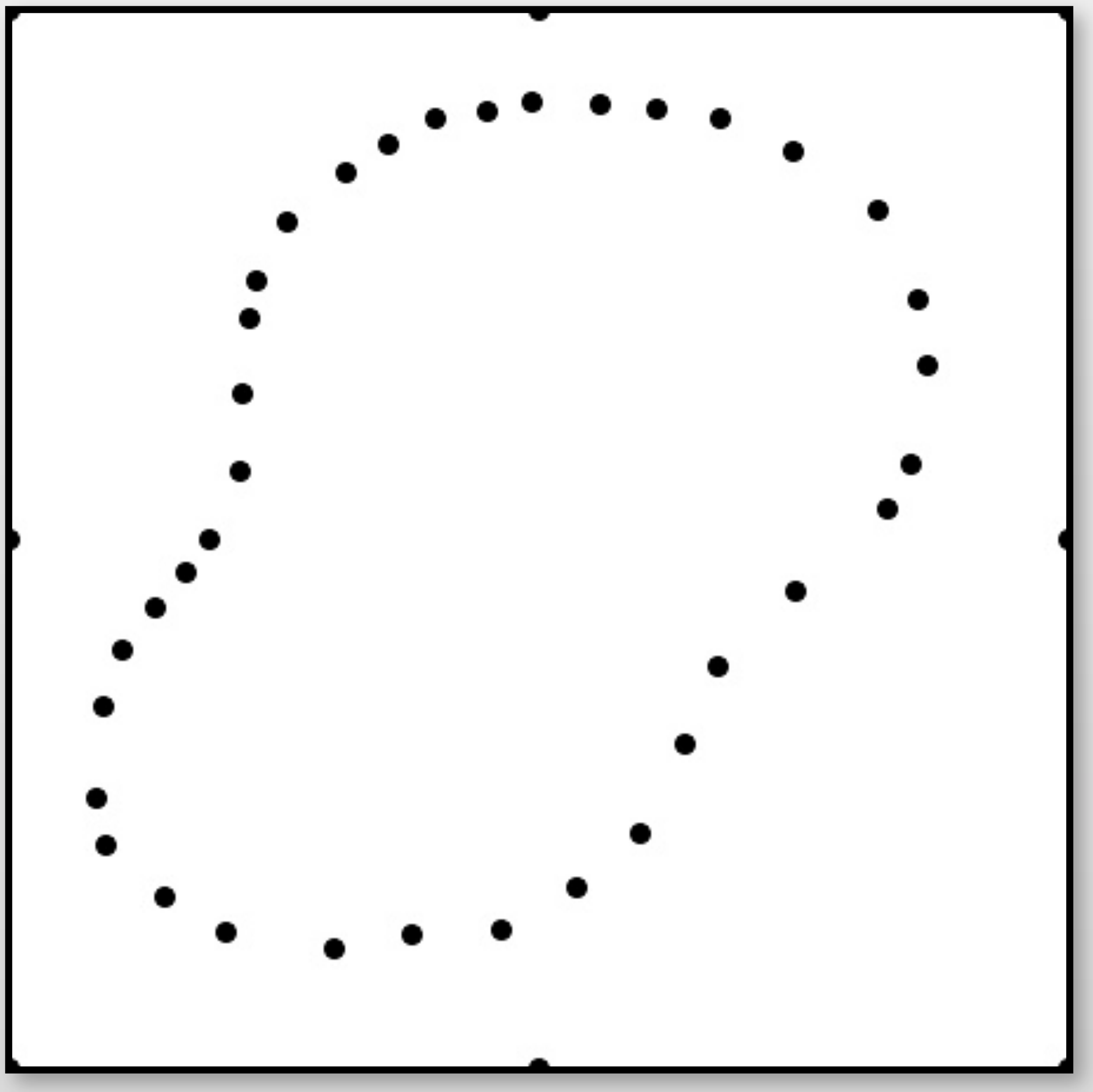


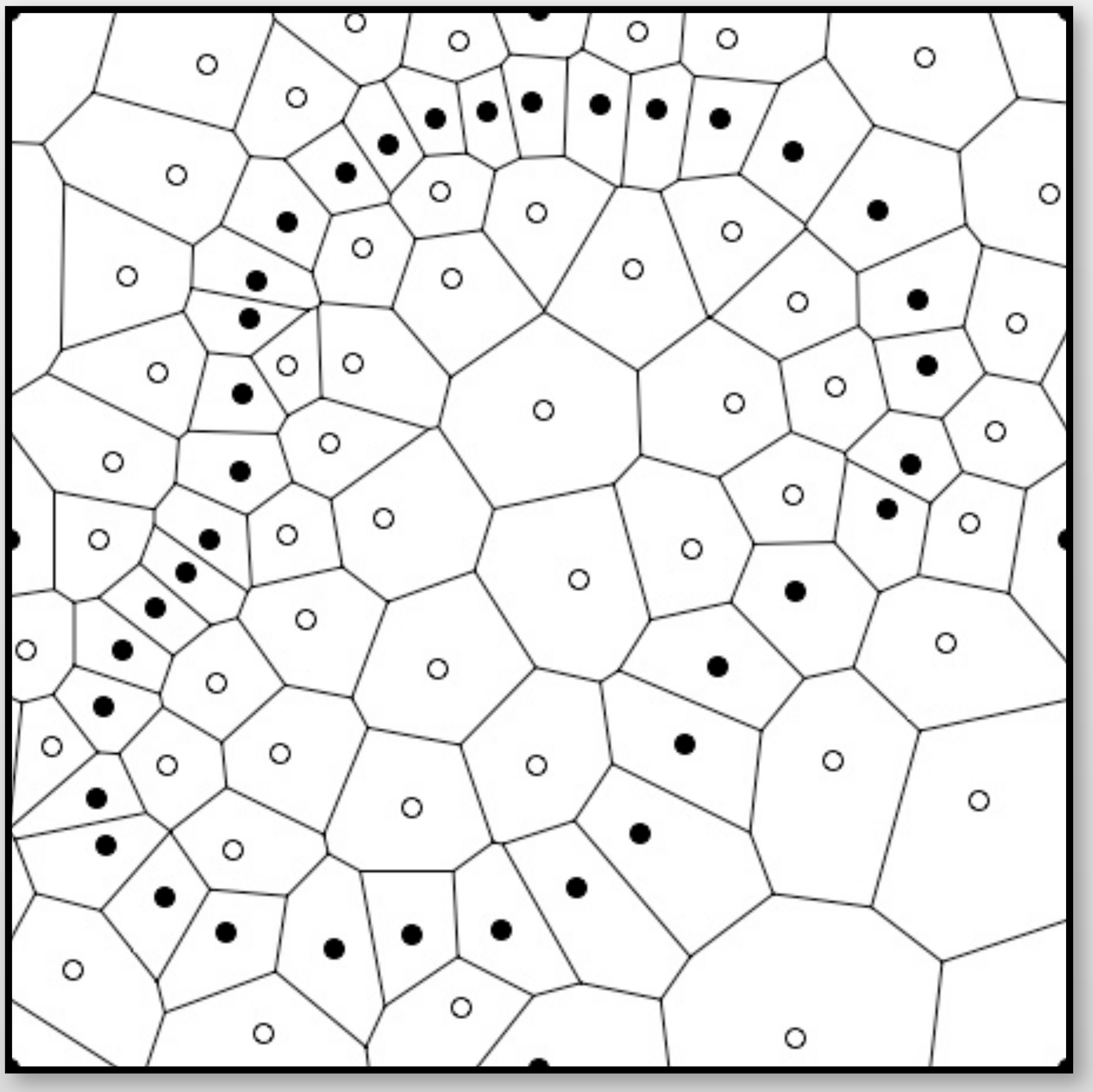
Meshing Points

Input: $P \subset \mathbb{R}^d$

Output: $M \supset P$ with a "nice" Voronoi diagram

$n = |P|, m = |M|$



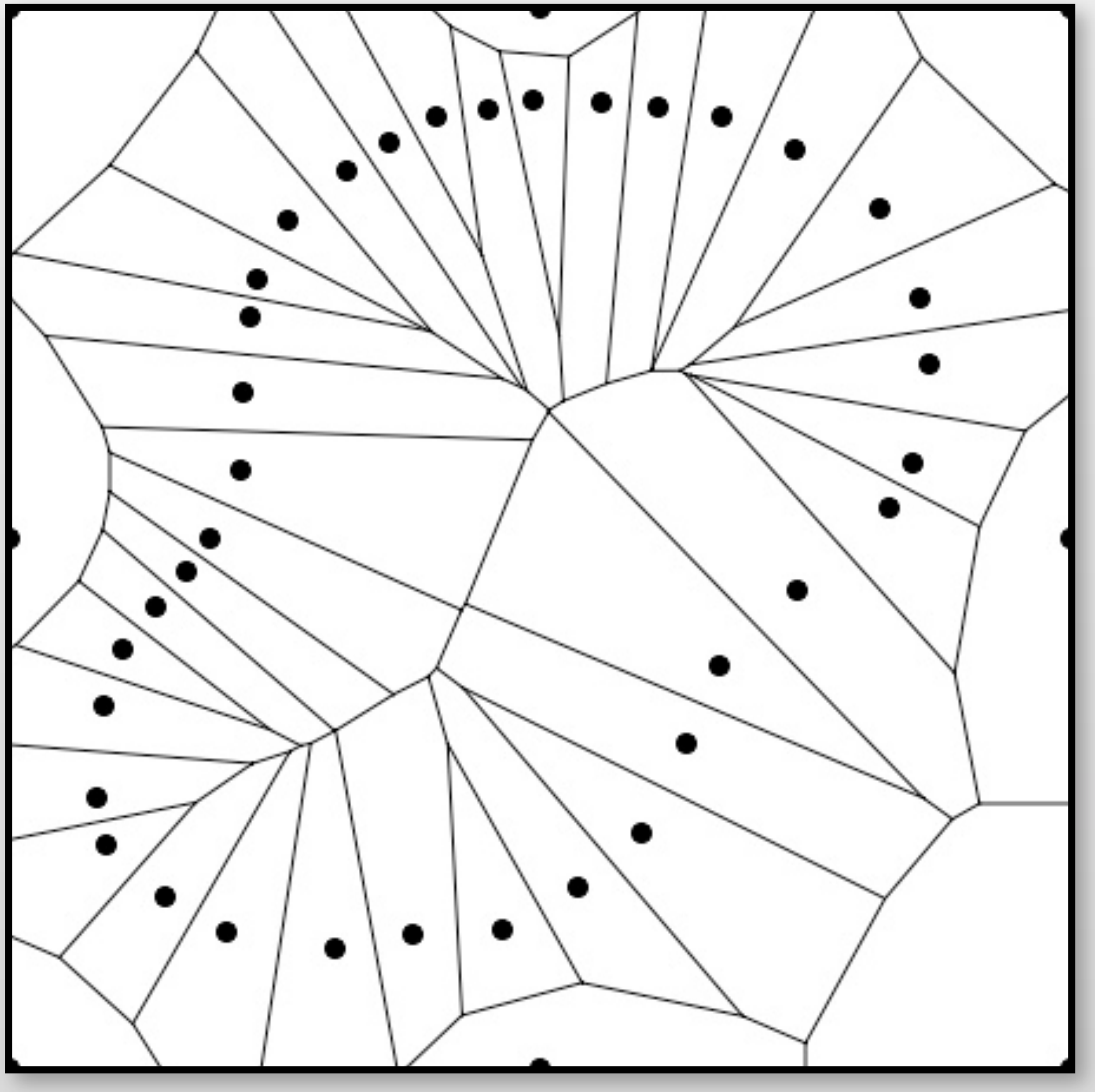


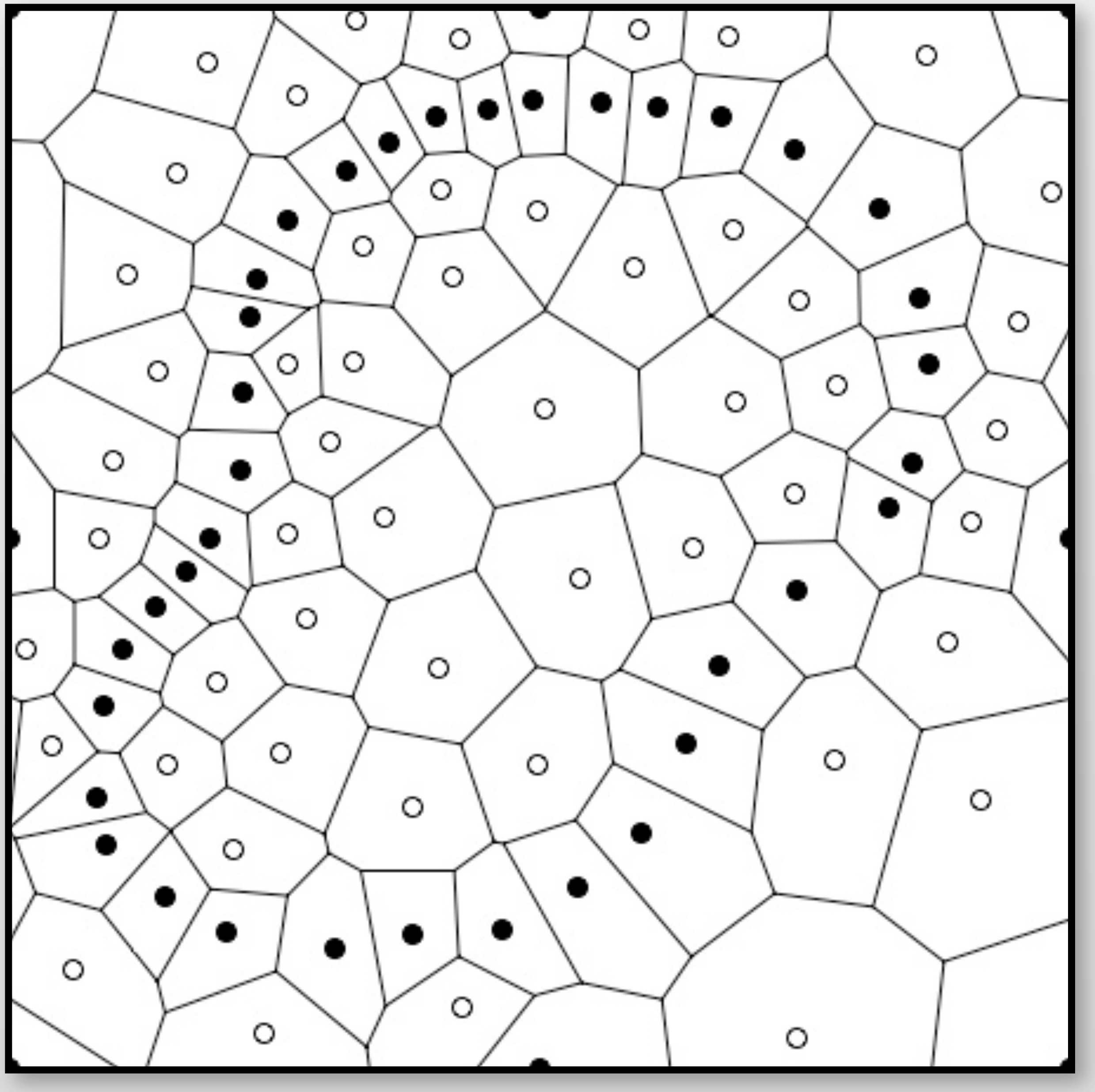
Meshing Points

Input: $P \subset \mathbb{R}^d$

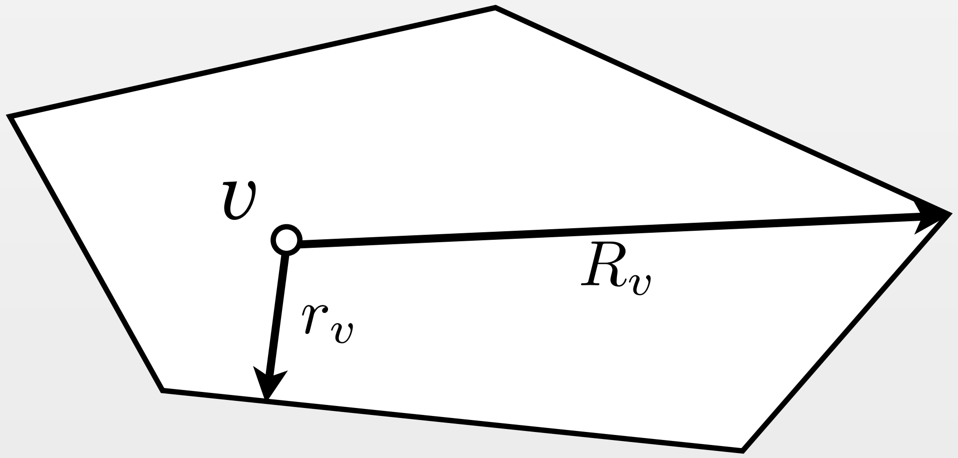
Output: $M \supset P$ with a "nice" Voronoi diagram

$n = |P|, m = |M|$





Meshing Guarantees



Aspect Ratio (quality):

$$\frac{R_v}{r_v} \leq \tau$$

Cell Sizing:

$$\operatorname{lfs}(x) := \mathbf{d}(x, P \setminus \{\operatorname{NN}(x)\})$$
$$R_v \leq \varepsilon \operatorname{lfs}(v)$$

Constant Local Complexity:

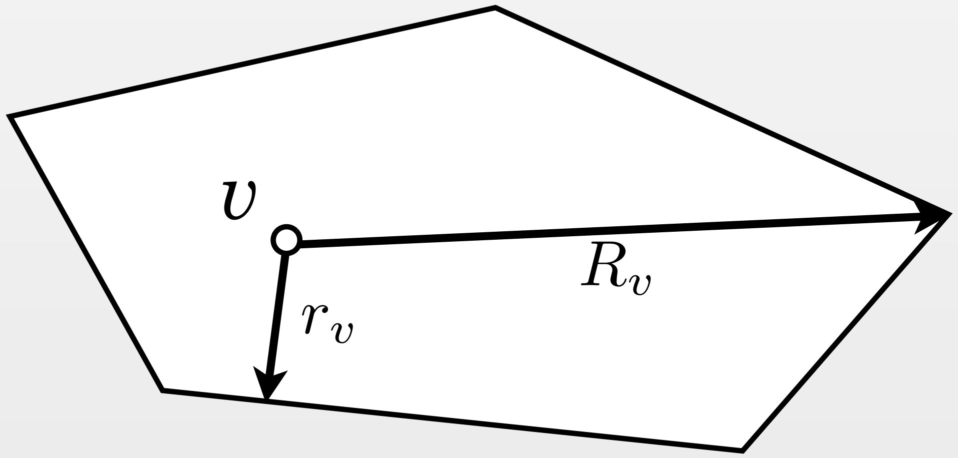
The degree of the 1-skeleton is $2^{O(d)}$.

Optimality and Running time:

$$|M| = \Theta(|\text{Optimal}|)$$

Running time: $O(n \log n + |M|)$

Meshing Guarantees



Aspect Ratio (quality):

$$\frac{R_v}{r_v} \leq \tau$$

Cell Sizing:

$$\text{lfs}(x) := \mathbf{d}(x, P \setminus \{\text{NN}(x)\})$$

ε -refined

$$R_v \leq \varepsilon \text{lfs}(v)$$

Constant Local Complexity:

The degree of the 1-skeleton is $2^{O(d)}$.

Optimality and Running time:

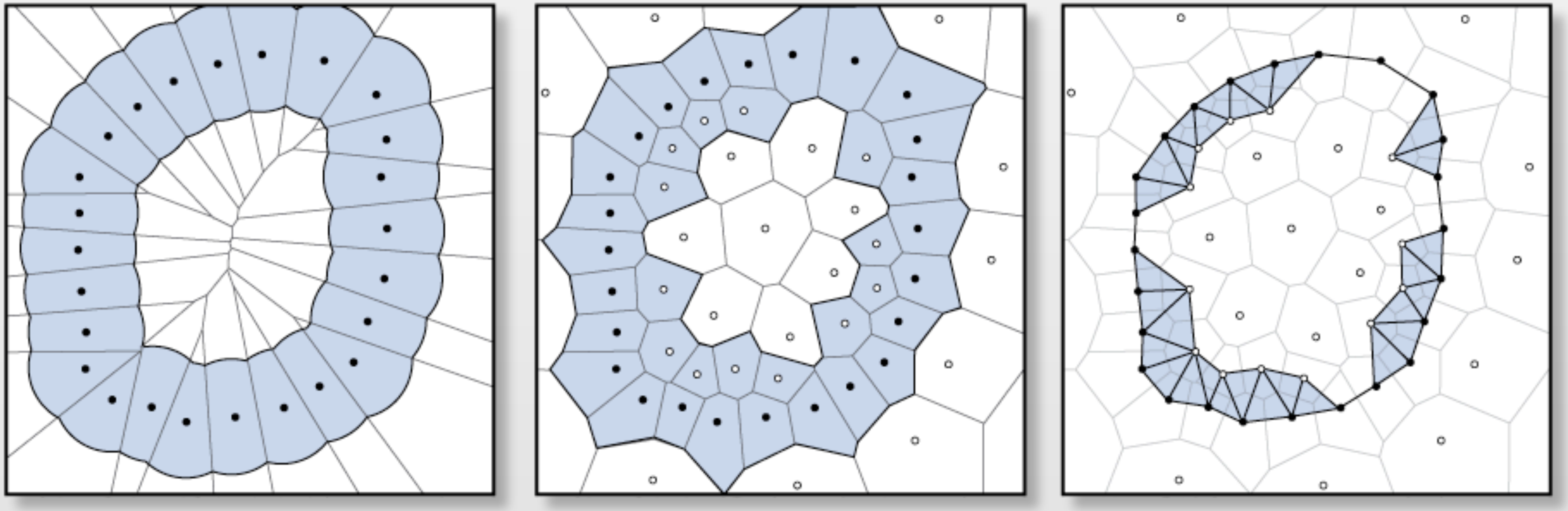
$$|M| = \Theta(|\text{Optimal}|)$$

Running time: $O(n \log n + |M|)$

Mesh Filtrations

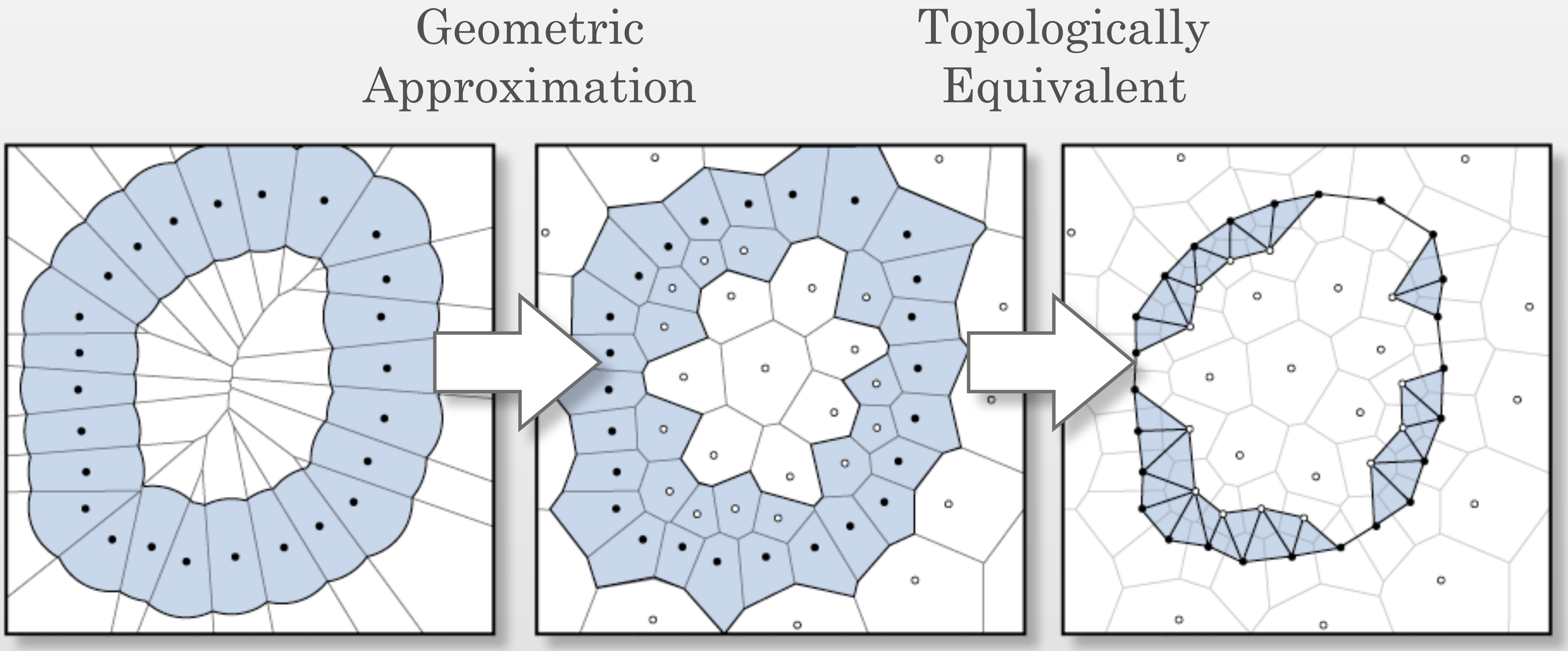
Geometric Approximation

Topologically Equivalent



1. Compute the function on the vertices.
2. Approximate the sublevel with a union of Voronoi cells.
3. Filter the Delaunay triangulation appropriately.

Mesh Filtrations

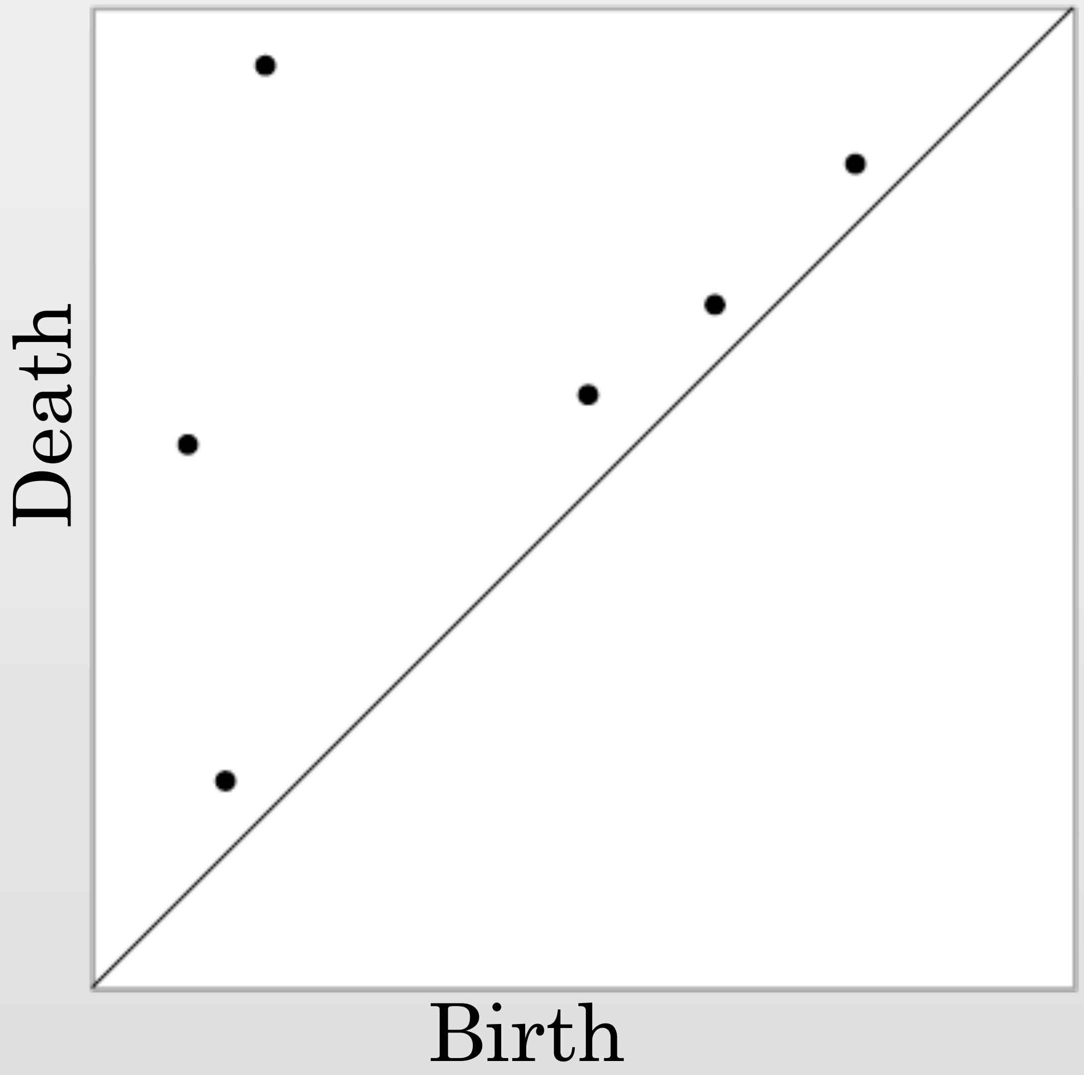


1. Compute the function on the vertices.

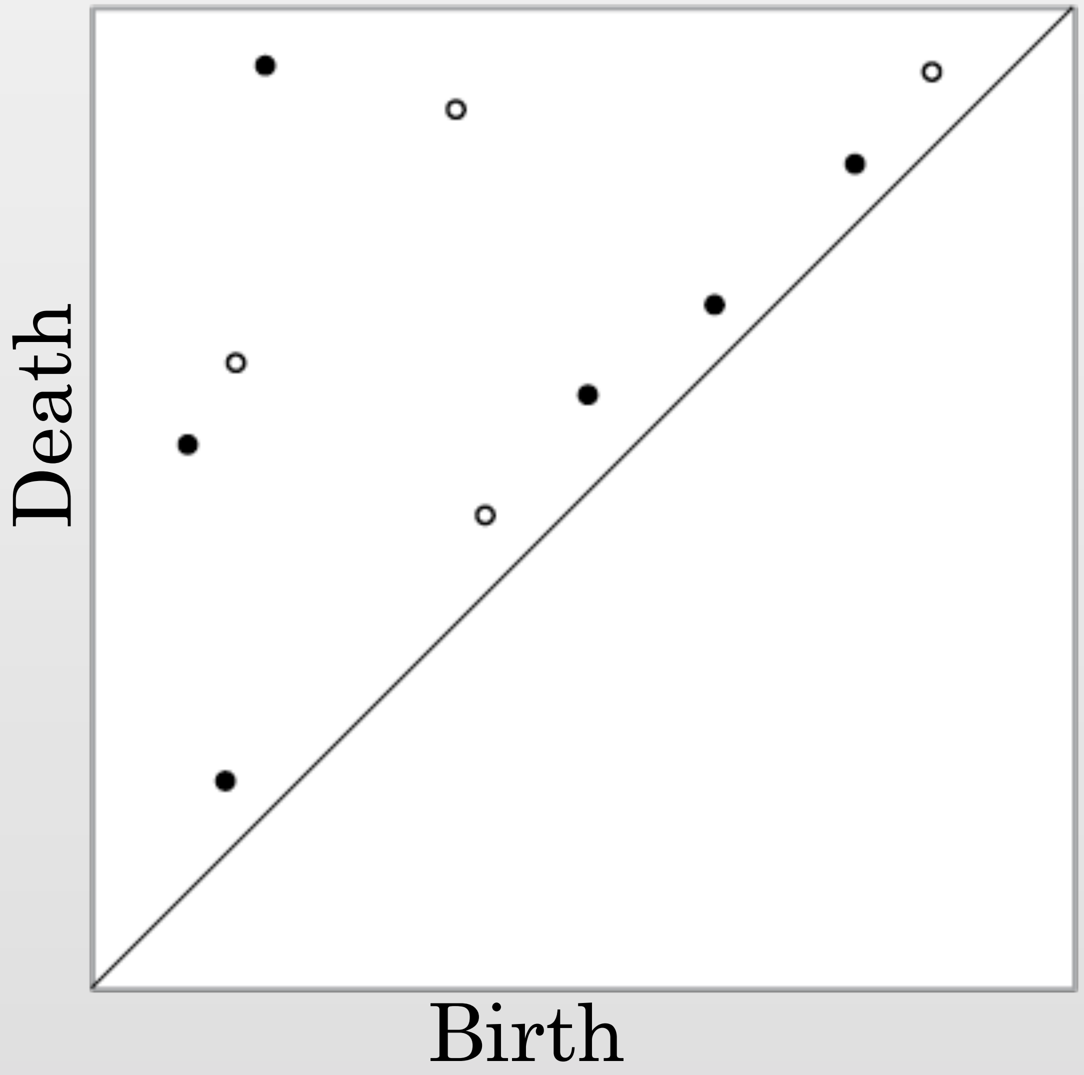
2. Approximate the sublevel with a union of Voronoi cells.

3. Filter the Delaunay triangulation appropriately.

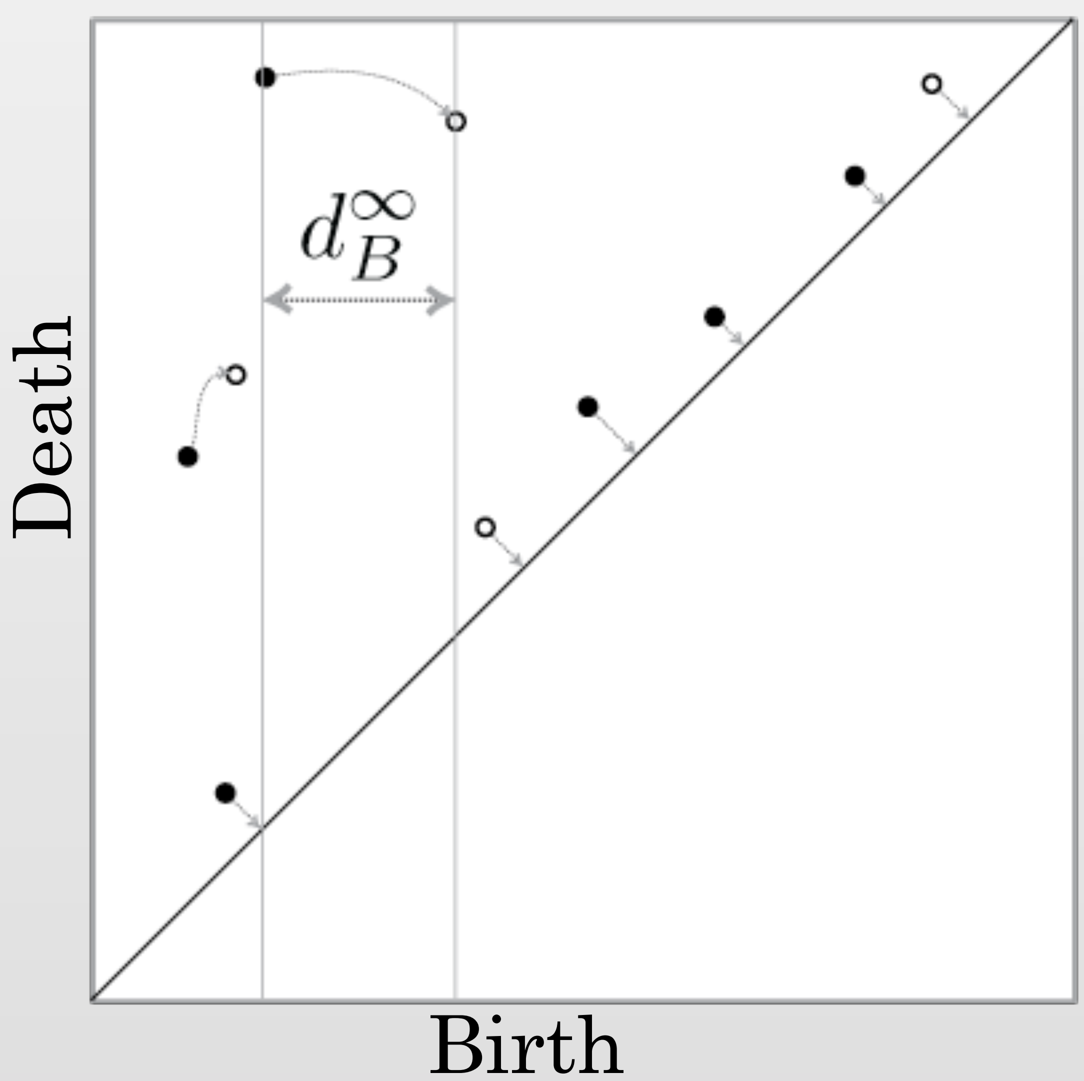
Persistence Diagrams



Persistence Diagrams



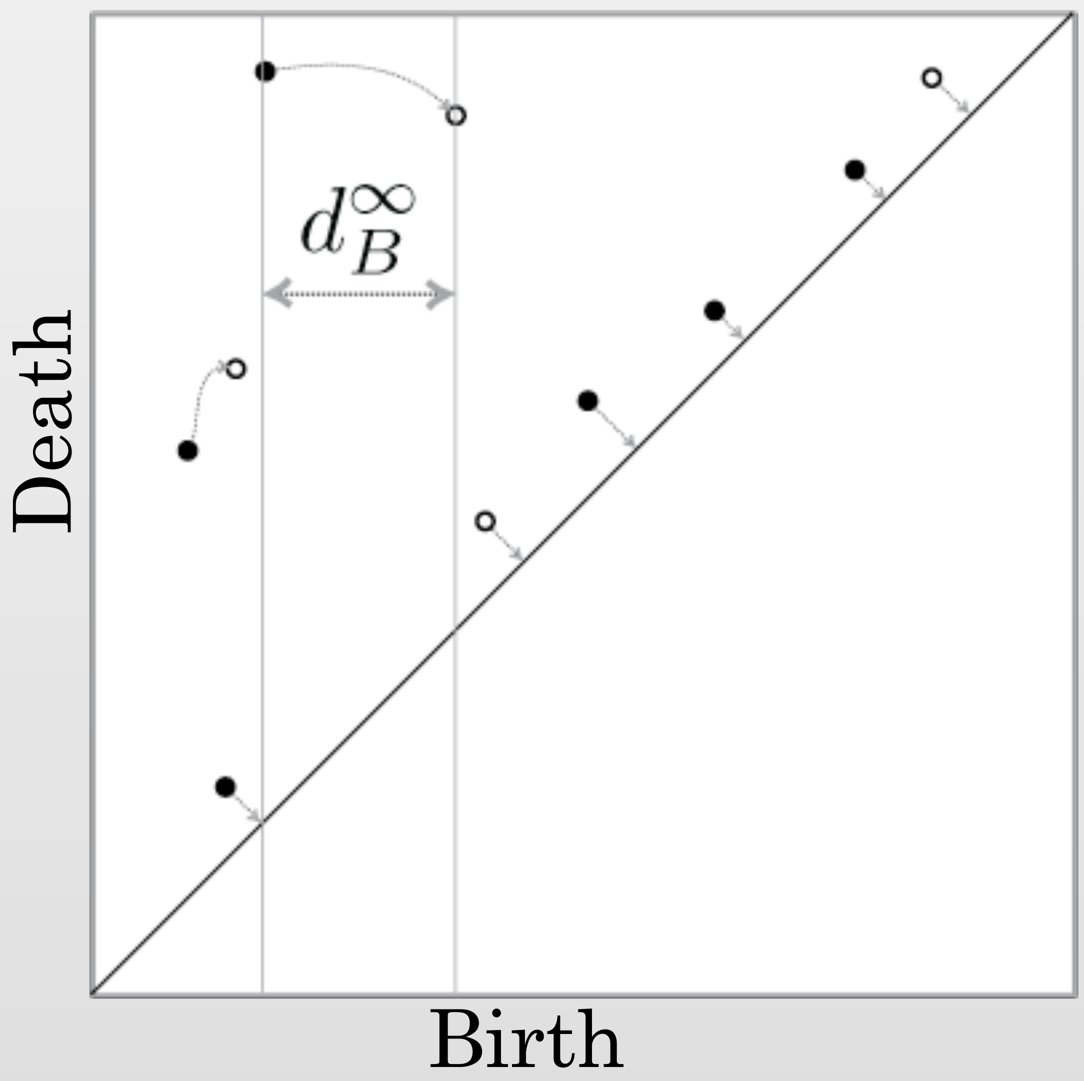
Persistence Diagrams



Bottleneck Distance

$$d_B^\infty = \max_i |p_i - q_i|_\infty$$

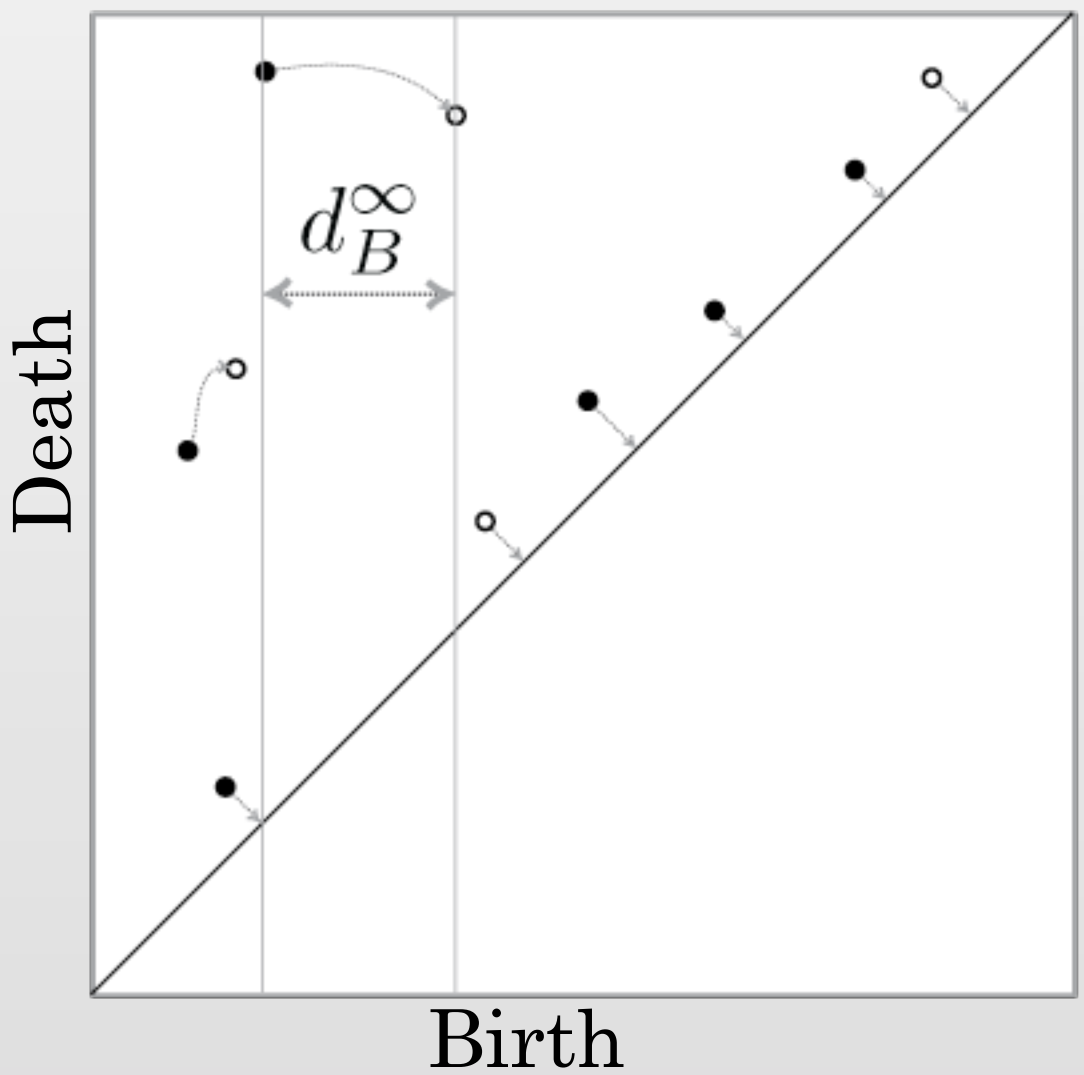
Approximate
Persistence Diagrams



Bottleneck Distance

$$d_B^\infty = \max_i |p_i - q_i|_\infty$$

Approximate
Persistence Diagrams

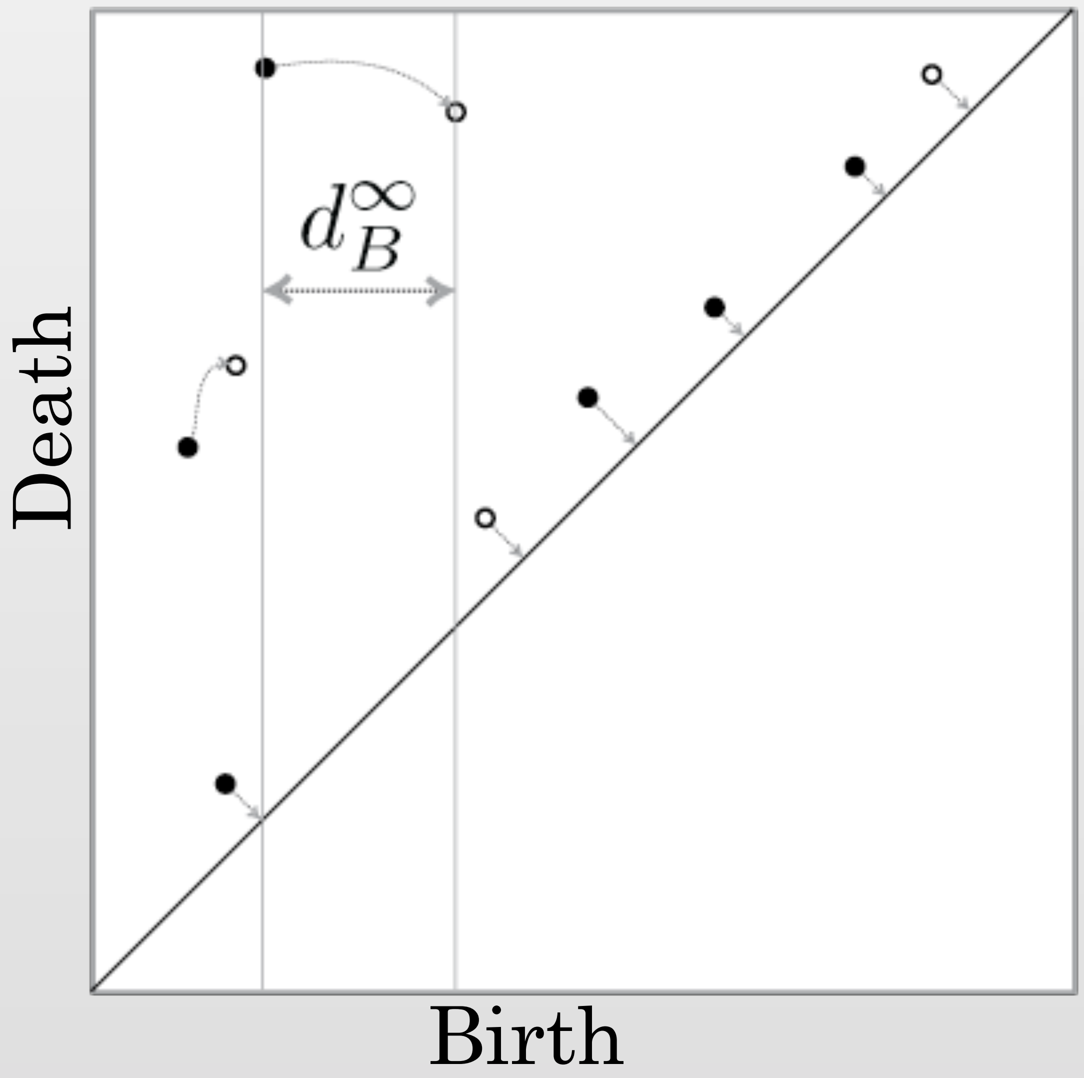


Bottleneck Distance

$$d_B^\infty = \max_i |p_i - q_i|_\infty$$

Birth and Death times differ by a constant factor.

Approximate Persistence Diagrams



Bottleneck Distance

$$d_B^\infty = \max_i |p_i - q_i|_\infty$$

Birth and Death times differ by a constant factor.

This is just the bottleneck distance of the log-scale diagrams.

Stability theorems can be viewed as approximation theorems.

Stability theorems can be viewed as approximation theorems.

This is just a change in perspective.

Stability theorems can be viewed as approximation theorems.

This is just a change in perspective.

Define the **log-scale filtration** of $\mathcal{F} = \{F_\alpha\}$ as

$$\mathcal{F}^{\log} := \{F_\alpha^{\log}\} \text{ where } F_{\log \alpha}^{\log} := F_\alpha$$

Stability theorems can be viewed as approximation theorems.

This is just a change in perspective.

Define the **log-scale filtration** of $\mathcal{F} = \{F_\alpha\}$ as

$$\mathcal{F}^{\log} := \{F^{\log}_\alpha\} \text{ where } F^{\log}_{\log\alpha} := F_\alpha$$

Given filtrations $\mathcal{F}$ and $\mathcal{G}$, we say
Dgm $\mathcal{F}$ is a γ**-approximation** to Dgm $\mathcal{G}$ iff

$$\mathbf{d}_B(\mathrm{Dgm}\ \mathcal{F}^{\log}, \mathrm{Dgm}\ \mathcal{G}^{\log}) \leq \log\gamma.$$

Stability theorems can be viewed as approximation theorems.

This is just a change in perspective.

Define the **log-scale filtration** of $\mathcal{F} = \{F_\alpha\}$ as

$$\mathcal{F}^{\log} := \{F^{\log}_\alpha\} \text{ where } F^{\log}_{\log\alpha} := F_\alpha$$

Given filtrations $\mathcal{F}$ and $\mathcal{G}$, we say
$\mathrm{Dgm}\,\mathcal{F}$ is a γ**-approximation** to $\mathrm{Dgm}\,\mathcal{G}$ iff

$$\mathbf{d}_B(\mathrm{Dgm}\,\mathcal{F}^{\log}, \mathrm{Dgm}\,\mathcal{G}^{\log}) \leq \log\gamma.$$

Lemma. *Let $\mathcal{F} = \{F_\alpha\}$ and $\mathcal{G} = \{G_\alpha\}$ be filtrations. If $F_{\alpha/\gamma} \subseteq G_\alpha \subseteq F_{\alpha\gamma}$ for all $\alpha \geq 0$, then $\mathrm{Dgm}\,\mathcal{F}$ is a γ-approximation to $\mathrm{Dgm}\,\mathcal{G}$.*

The main idea

The main idea

Given a function $f : \mathbb{R}^d \to \mathbb{R}$, let $\mathcal{F} = \{F_\alpha\}$ be its sublevel filtration.

$$F_\alpha := f^{-1}[0, \alpha]$$

The main idea

Given a function $f : \mathbb{R}^d \to \mathbb{R}$, let $\mathcal{F} = \{F_\alpha\}$ be its sublevel filtration.

$$F_\alpha := f^{-1}[0, \alpha]$$

Let M be a quality mesh.

The main idea

Given a function $f : \mathbb{R}^d \to \mathbb{R}$, let $\mathcal{F} = \{F_\alpha\}$ be its sublevel filtration.

$$F_\alpha := f^{-1}[0, \alpha]$$

Let M be a quality mesh.

Let $\mathcal{V} = \{V_\alpha\}$ be the Voronoi filtration of f on M.

$$V_\alpha := \bigcup_{\substack{v \in M \\ f(v) \leq \alpha}} \mathrm{Vor}(v)$$

The main idea

Given a function $f : \mathbb{R}^d \to \mathbb{R}$, let $\mathcal{F} = \{F_\alpha\}$ be its sublevel filtration.

$$F_\alpha := f^{-1}[0, \alpha]$$

Let M be a quality mesh.

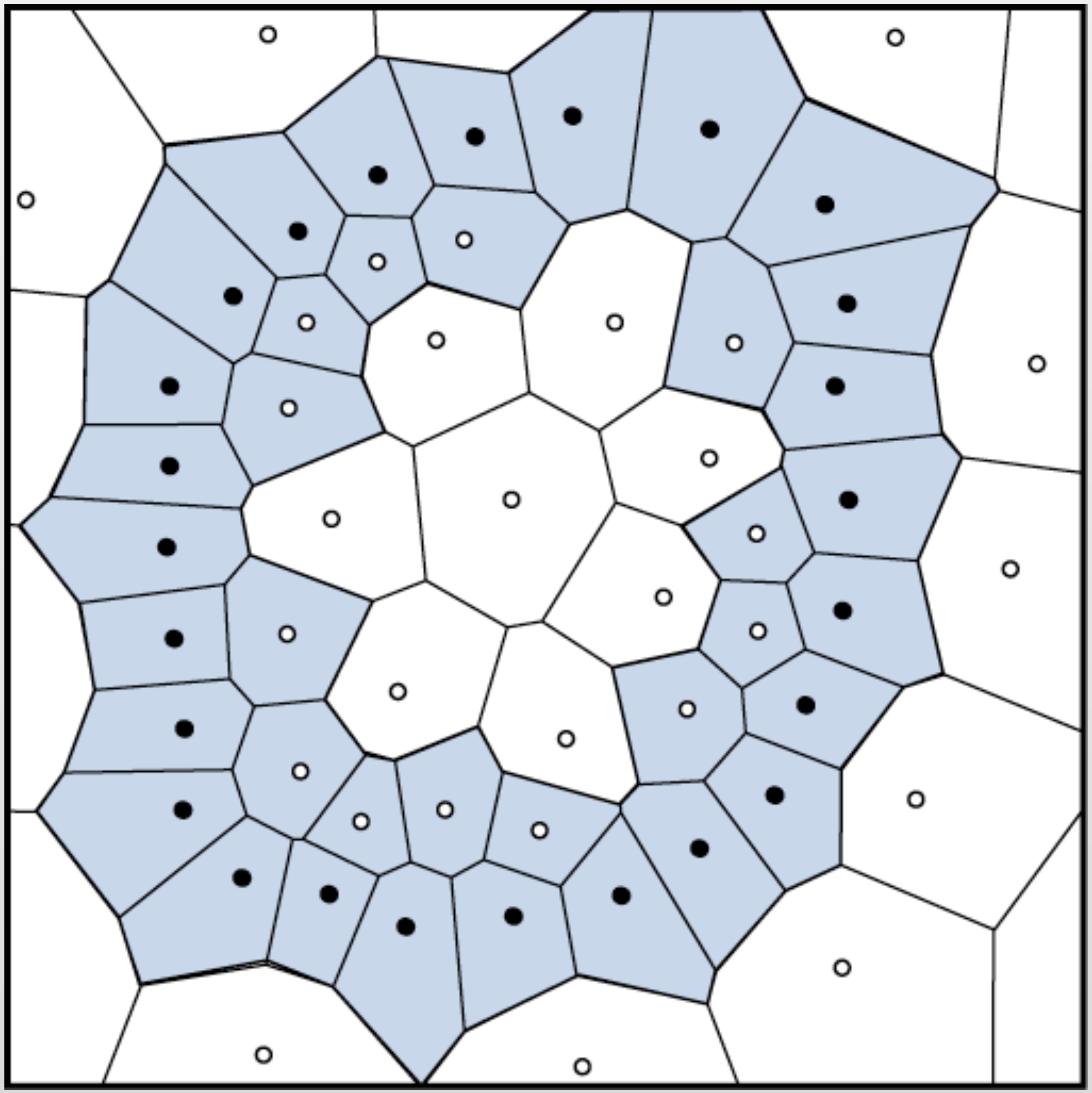
Let $\mathcal{V} = \{V_\alpha\}$ be the Voronoi filtration of f on M.

$$V_\alpha := \bigcup_{\substack{v \in M \\ f(v) \leq \alpha}} \mathrm{Vor}(v)$$

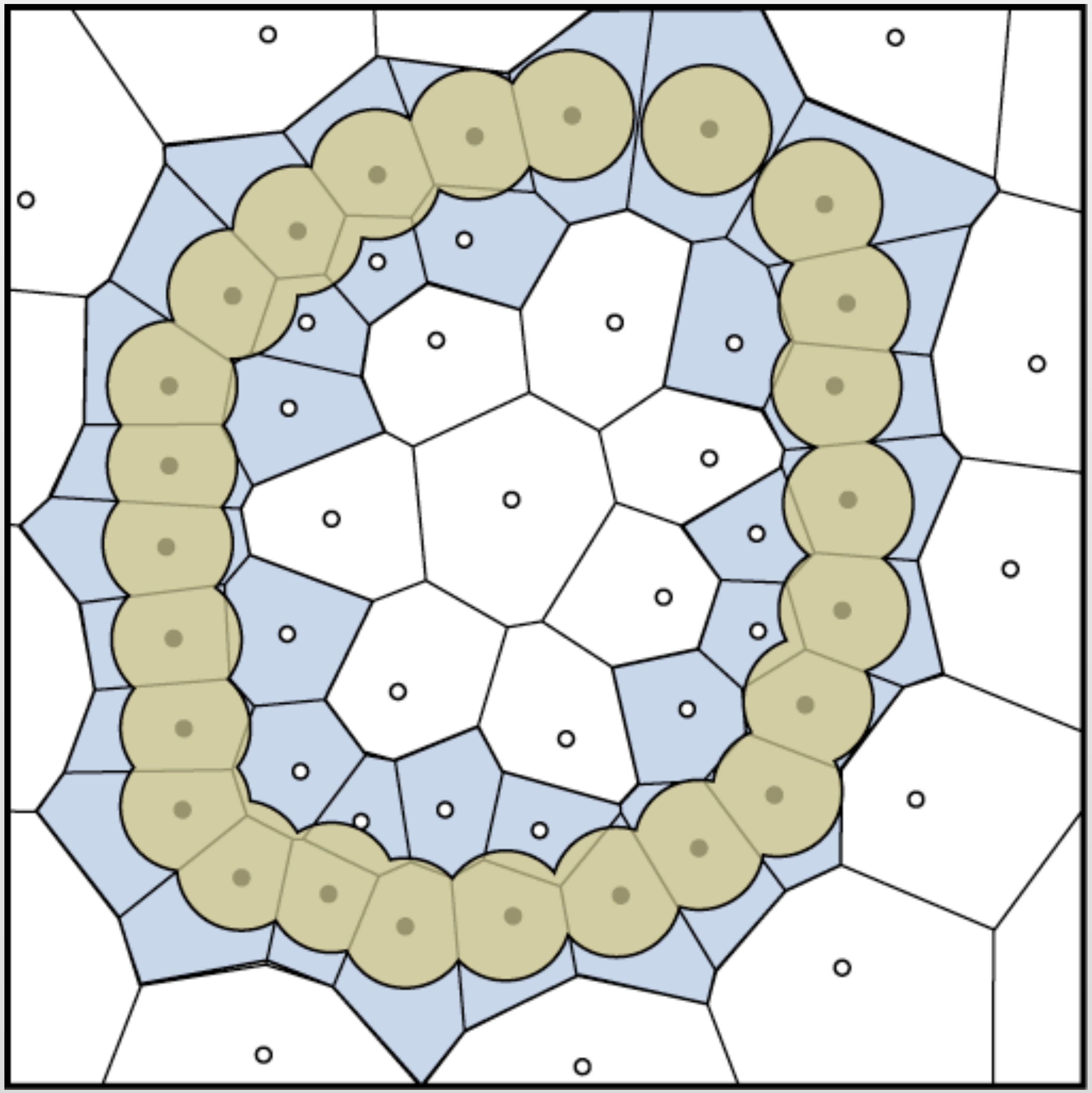
We will show $\mathrm{Dgm}\, \mathcal{V}$ is a good approximation to $\mathrm{Dgm}\, \mathcal{F}$.

The Voronoi filtration interleaves with the offset filtration.

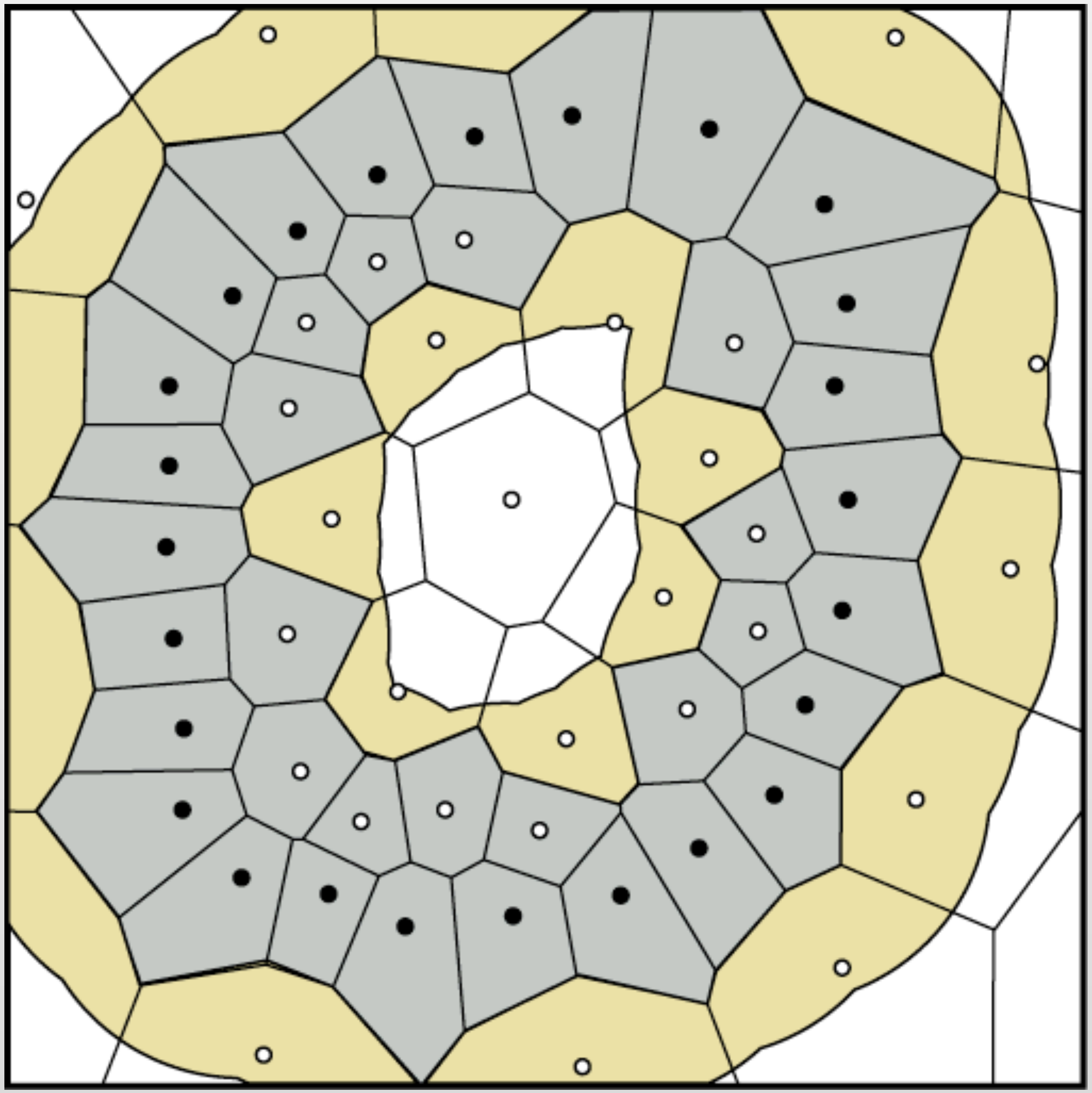
The Voronoi filtration interleaves with the offset filtration.



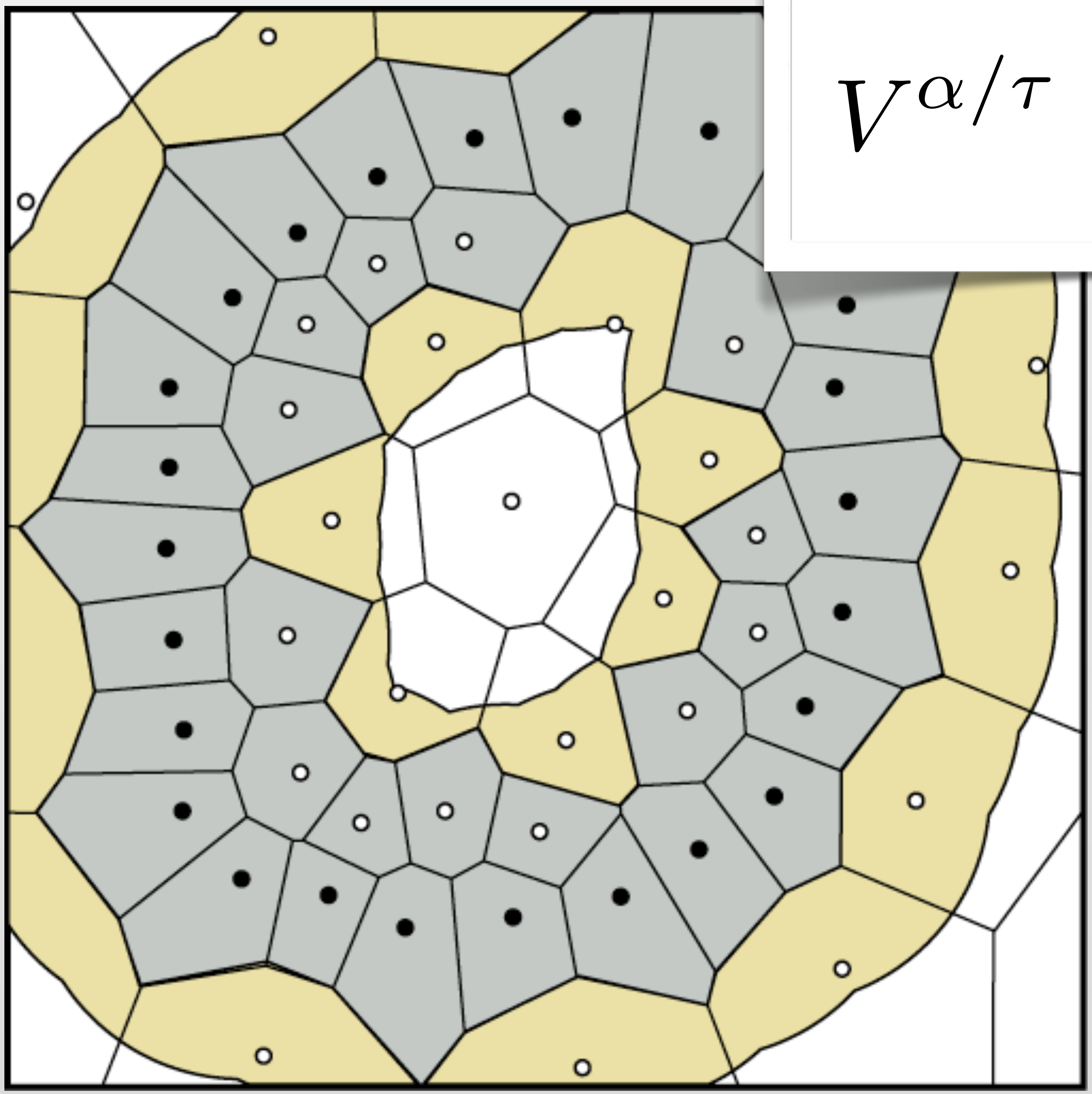
The Voronoi filtration interleaves with the offset filtration.



The Voronoi filtration interleaves with the offset filtration.

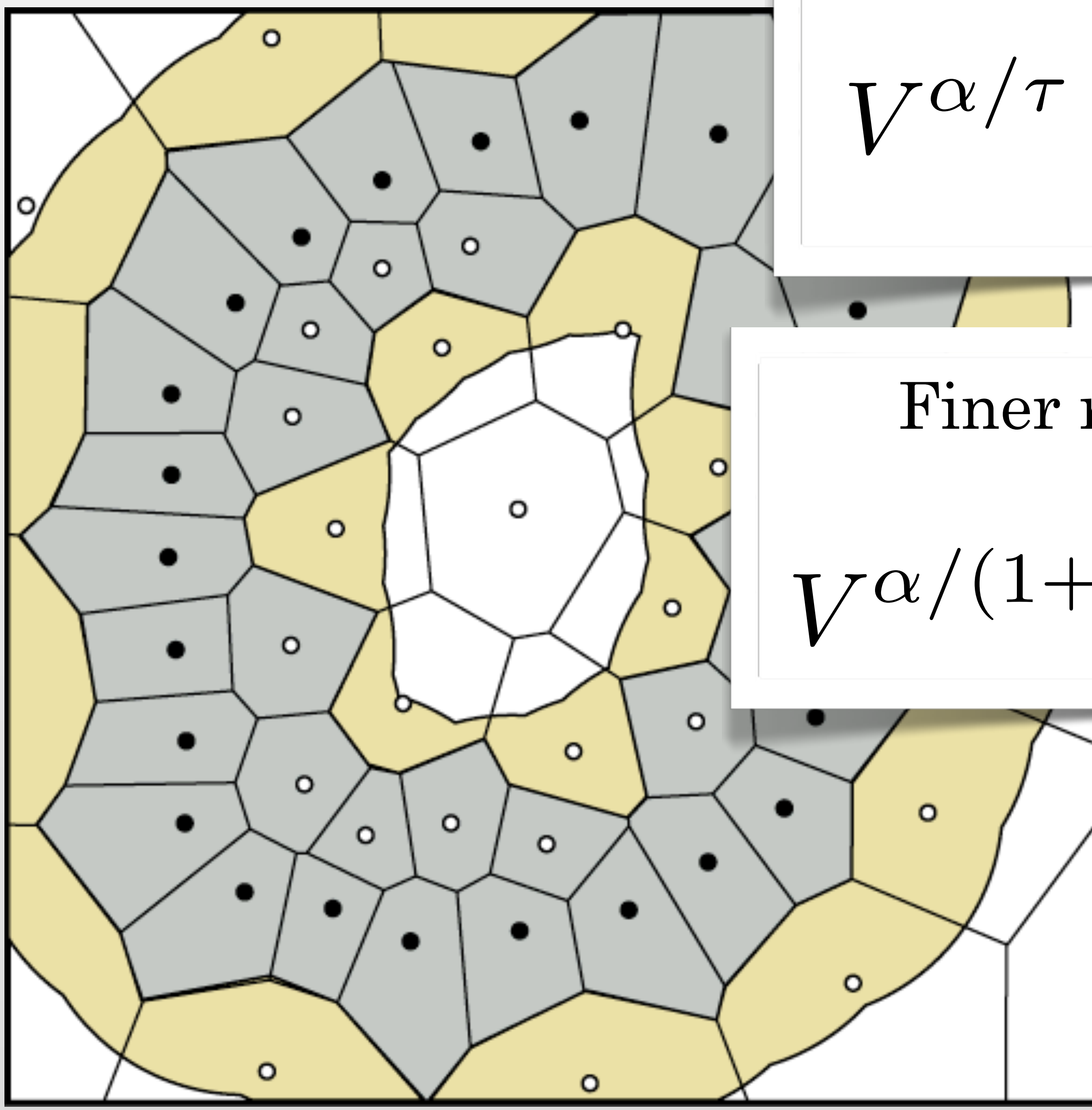


The Voronoi filtration interleaves with the offset filtration.



$$V^{\alpha/\tau} \subseteq P^{\alpha} \subseteq V^{\alpha\tau}$$

The Voronoi filtration interleaves with the offset filtration.

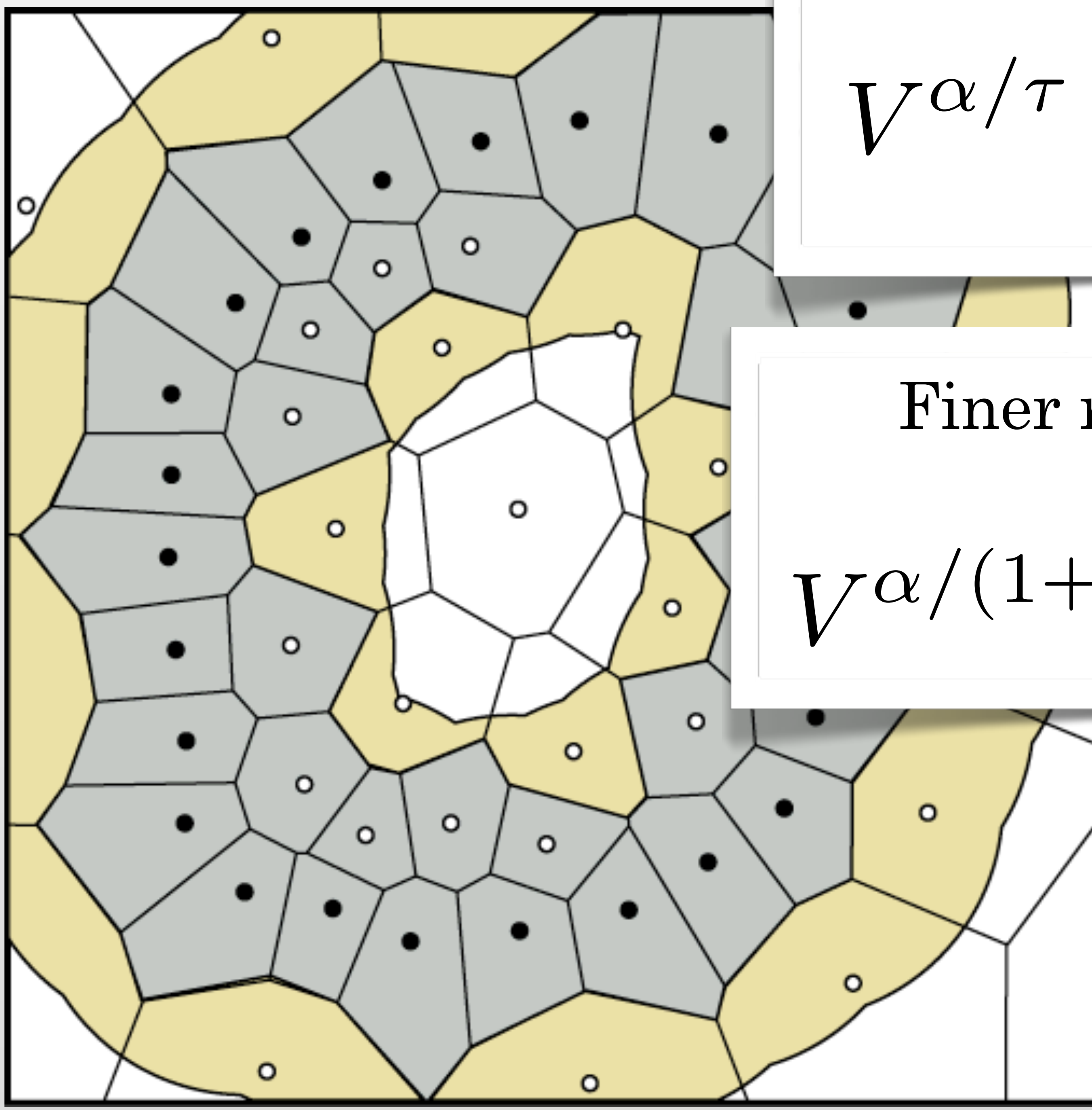


$$V^{\alpha/\tau} \subseteq P^{\alpha} \subseteq V^{\alpha\tau}$$

Finer refinement yields a tighter interleaving.

$$V^{\alpha/(1+\varepsilon)} \subseteq P^{\alpha} \subseteq V^{\alpha(1+\varepsilon)}$$

The Voronoi filtration interleaves with the offset filtration.



$$V^{\alpha/\tau} \subseteq P^{\alpha} \subseteq V^{\alpha\tau}$$

Finer refinement yields a tighter interleaving.

$$V^{\alpha/(1+\varepsilon)} \subseteq P^{\alpha} \subseteq V^{\alpha(1+\varepsilon)}$$

caveat: Special case for small scales.

The Not Obvious.

The Not Obvious.

1 Why meshing is not obviously a good idea for TDA.

The Not Obvious.

1. Why meshing is not obviously a good idea for TDA.

2. Why meshing is a nonobvious good idea for TDA.

The Not Obvious.

1 Why meshing is not obviously a good idea for TDA.

2 Why meshing is a nonobvious good idea for TDA.

3 Why meshing is obviously not a good idea for TDA.

The Not Obvious.

1. Why meshing is not obviously a good idea for TDA.

2. Why meshing is a nonobvious good idea for TDA.

3. ~~Why meshing is obviously not a good idea for TDA.~~

Meshing codes are hard to write.

image credit: Pointwise

Meshing codes are hard to write.

1 Numerical Robustness

image credit: Pointwise

Meshing codes are hard to write.

1 Numerical Robustness

2 Understanding tradeoffs

image credit: Pointwise

Meshing codes are hard to write.

1 Numerical Robustness

2 Understanding tradeoffs

3 “Necessary” heuristics

image credit: Pointwise

Meshing codes are hard to write.

1 Numerical Robustness

2 Understanding tradeoffs

3 “Necessary” heuristics

image credit: Pointwise

On the other hand, meshing for FEA often has stricter requirements than those for TDA.

Meshing codes are hard to write.

1 Numerical Robustness

2 Understanding tradeoffs

3 “Necessary” heuristics

image credit: Pointwise

On the other hand, meshing for FEA often has stricter requirements than those for TDA.

- Approximate gradients

Meshing codes are hard to write.

1 Numerical Robustness

2 Understanding tradeoffs

3 "Necessary" heuristics

image credit: Pointwise

On the other hand, meshing for FEA often has stricter requirements than those for TDA.

- Approximate gradients
- Slivers

The size of an optimal mesh is given by the *feature size measure*.

The size of an optimal mesh is given by the *feature size measure*.

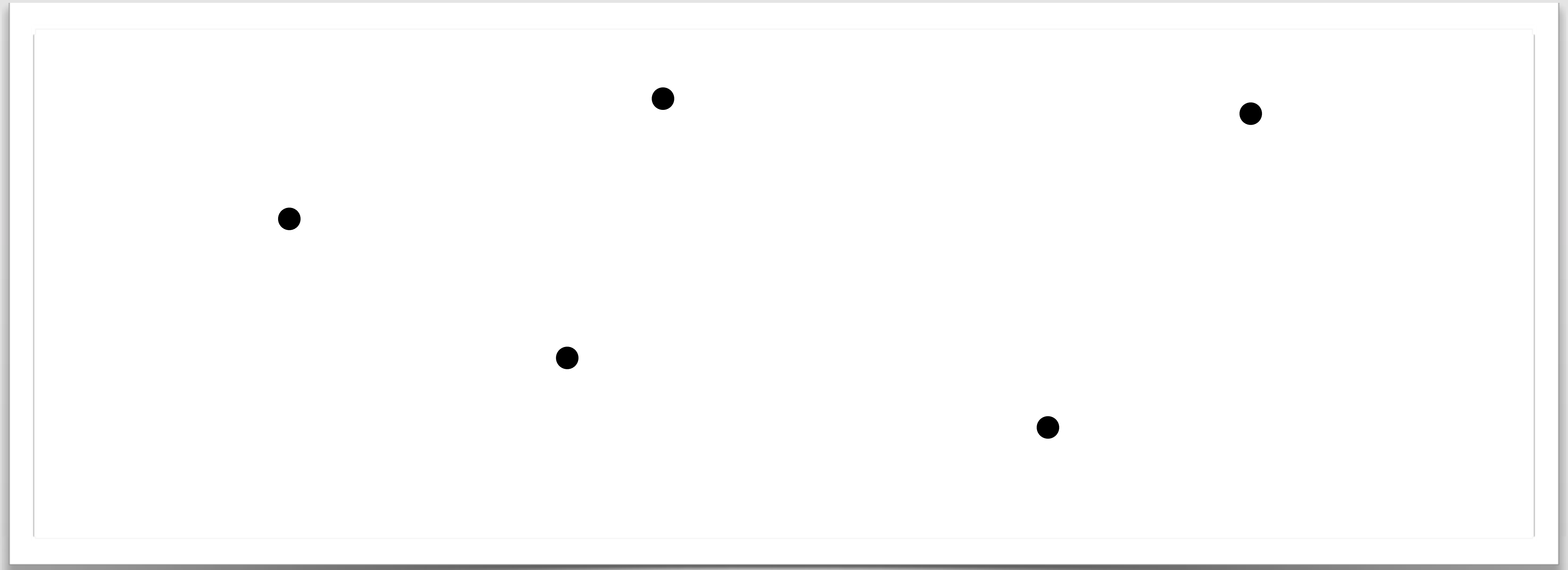
$$\mathrm{lfs}_P(x) := \text{Distance to second nearest neighbor in } P.$$

The size of an optimal mesh is given by the *feature size measure*.

$\mathrm{lfs}_P(x) :=$ Distance to second nearest neighbor in P.

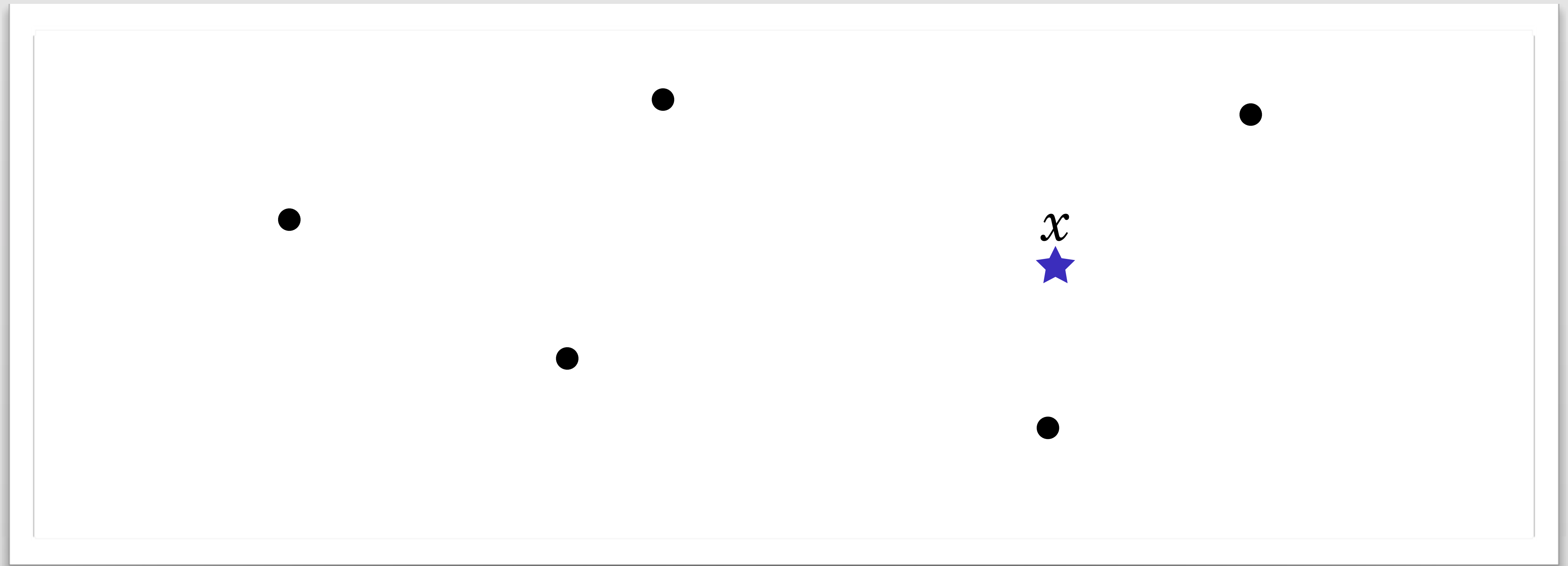
The size of an optimal mesh is given by the *feature size measure*.

$\mathrm{lfs}_P(x) :=$ Distance to second nearest neighbor in P.



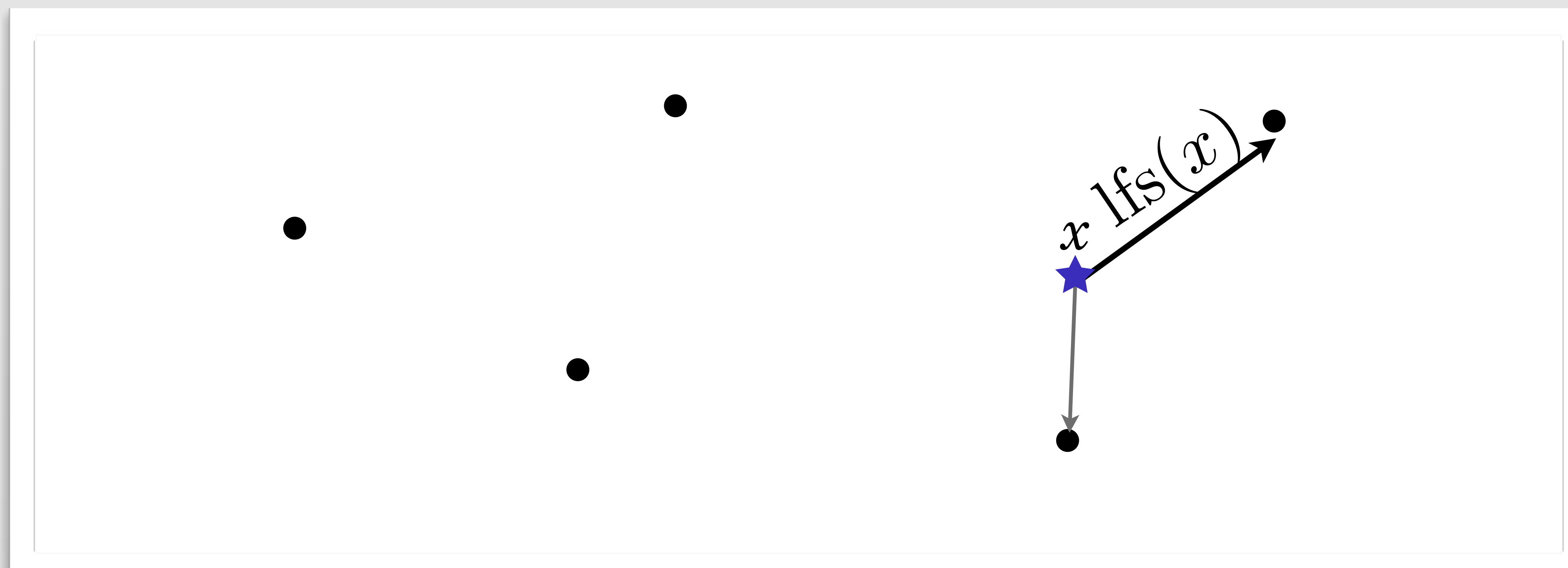
The size of an optimal mesh is given by the *feature size measure*.

$\mathrm{lfs}_P(x) :=$ Distance to second nearest neighbor in P.



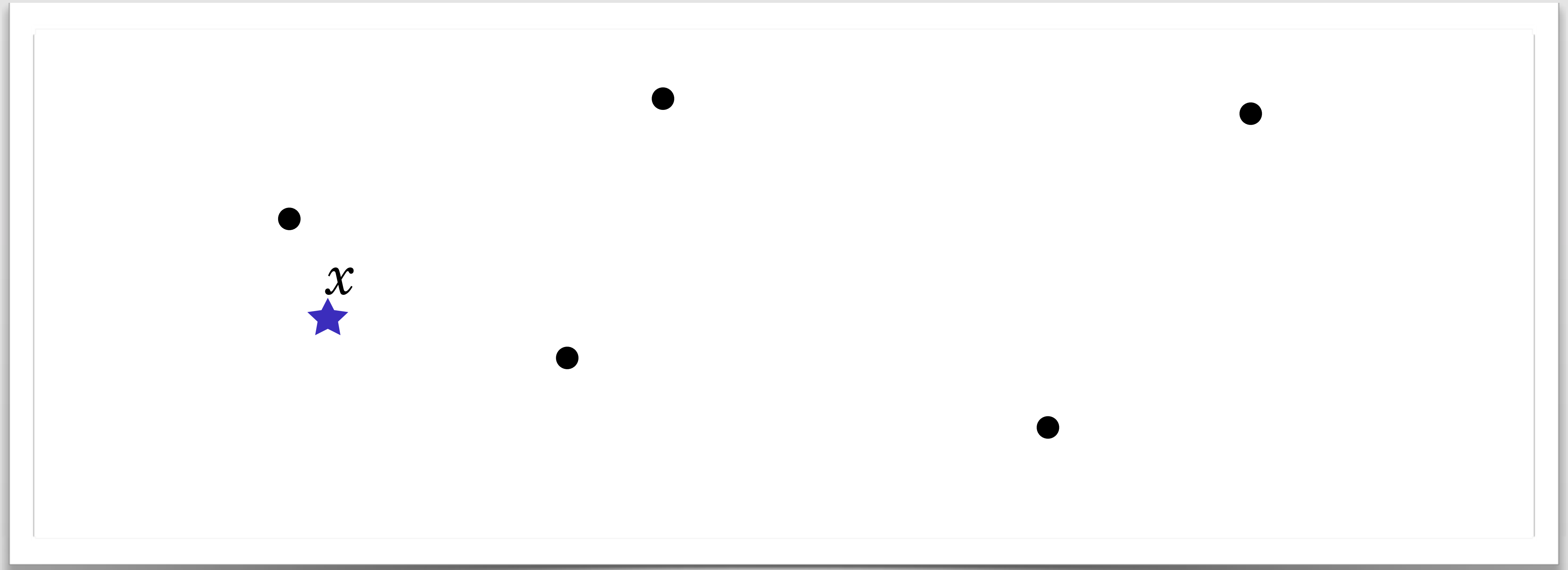
The size of an optimal mesh is given by the *feature size measure*.

$\mathrm{lfs}_P(x) :=$ Distance to second nearest neighbor in P.



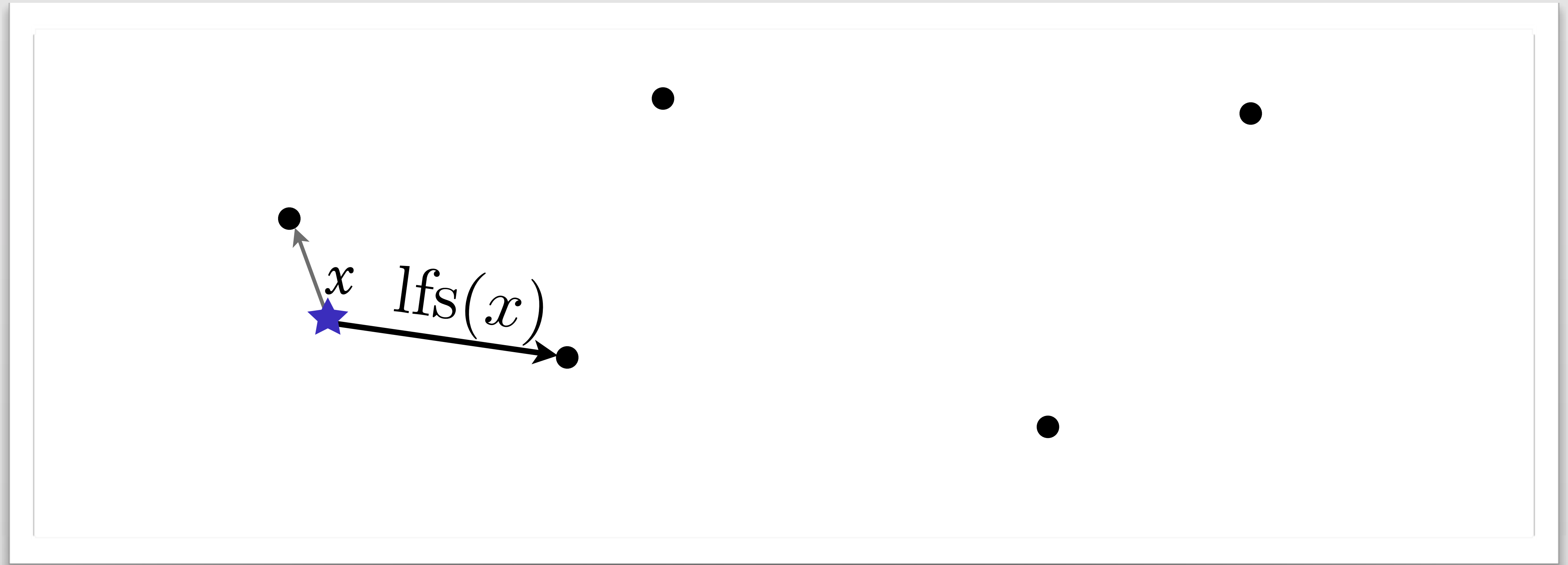
The size of an optimal mesh is given by the *feature size measure*.

$\mathrm{lfs}_P(x) :=$ Distance to second nearest neighbor in P.



The size of an optimal mesh is given by the *feature size measure*.

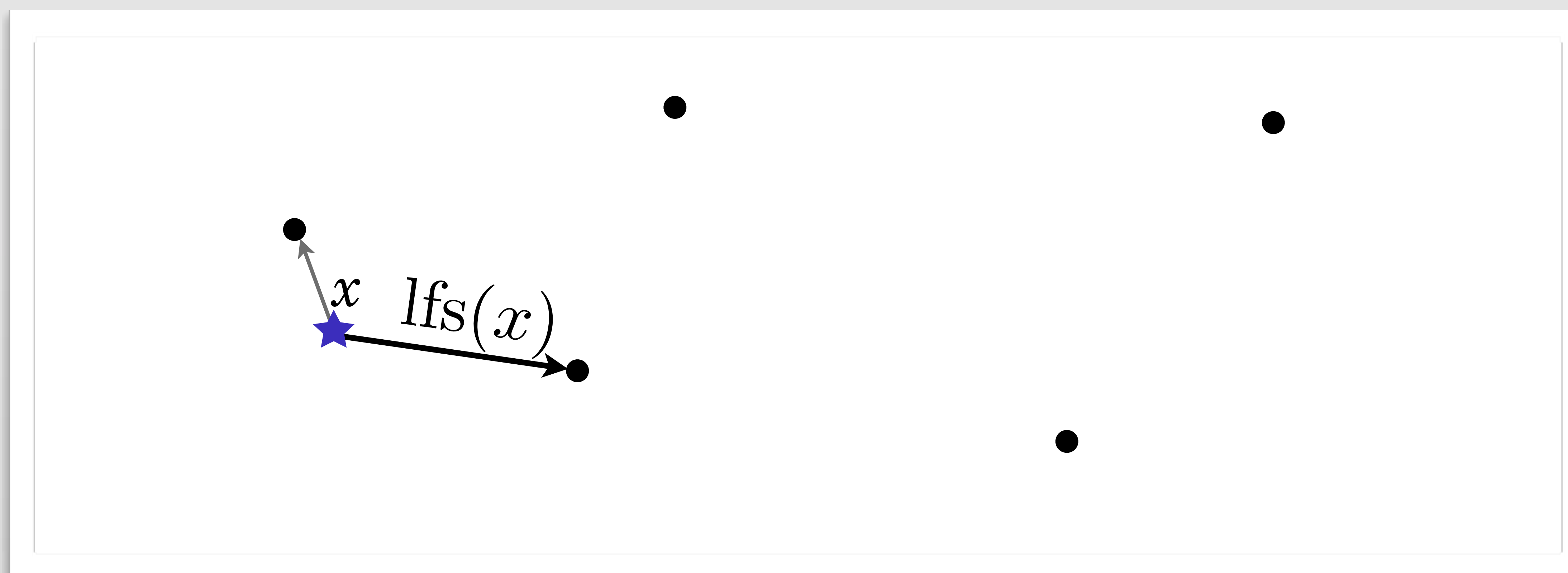
$\mathrm{lfs}_P(x) :=$ Distance to second nearest neighbor in P.



The size of an optimal mesh is given by the *feature size measure*.

$\text{lfs}_P(x) :=$ Distance to second nearest neighbor in P.

$$\text{Optimal Mesh Size} = \Theta\left(\int_\Omega \frac{dx}{\text{lfs}(x)^d}\right)$$

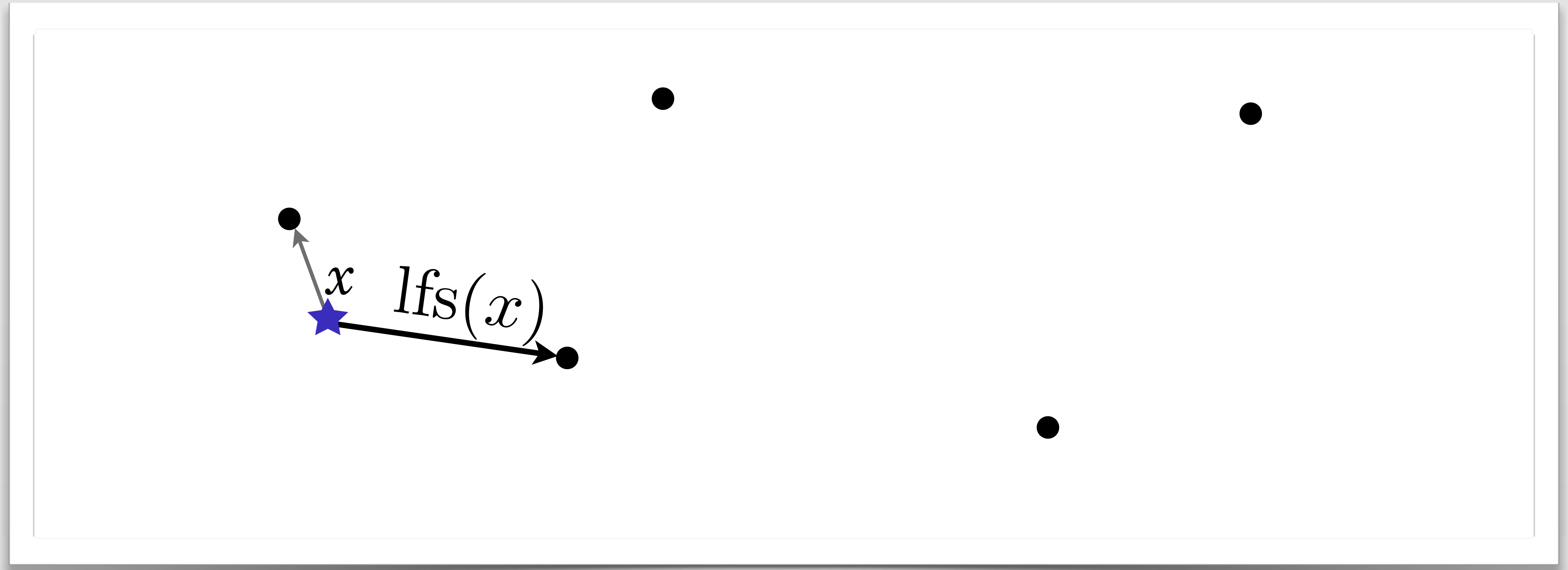


The size of an optimal mesh is given by the *feature size measure*.

$\mathrm{lfs}_P(x) :=$ Distance to second nearest neighbor in P.

number of vertices

$$\text{Optimal Mesh Size} = \Theta\left(\int_\Omega \frac{dx}{\mathrm{lfs}(x)^d}\right)$$



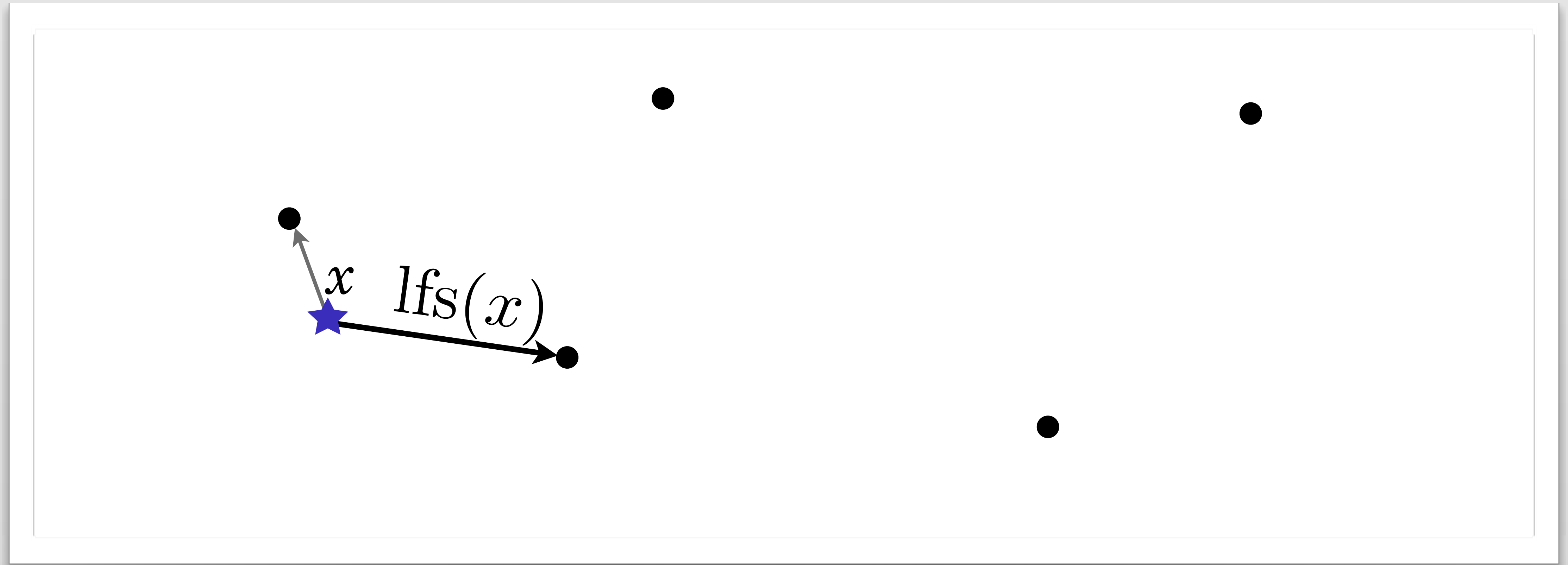
The size of an optimal mesh is given by the *feature size measure*.

$\text{lfs}_P(x) :=$ Distance to second nearest neighbor in P.

number of vertices

$$\text{Optimal Mesh Size} = \Theta\left(\int_\Omega \frac{dx}{\text{lfs}(x)^d}\right)$$

hides simple exponential in d



The size of an optimal mesh is given by the *feature size measure*.

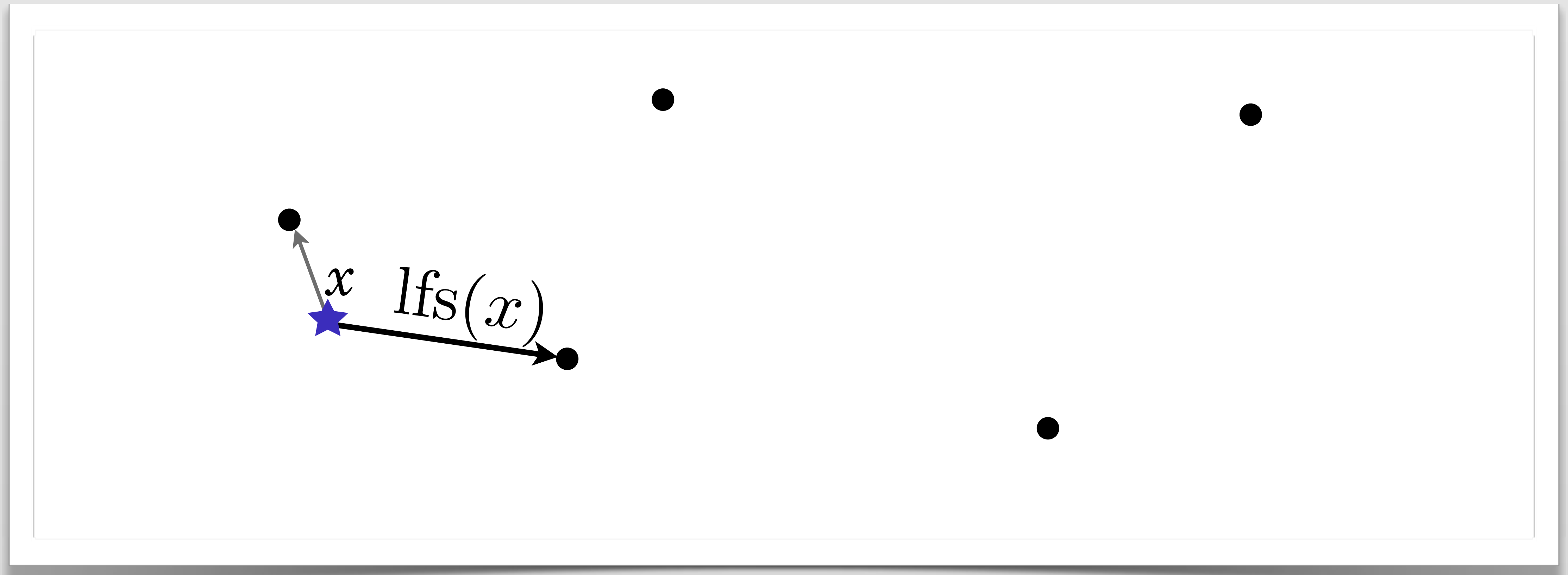
$\mathrm{lfs}_P(x) :=$ Distance to second nearest neighbor in P.

number of vertices

$$\text{Optimal Mesh Size} = \Theta\left(\int_\Omega \frac{dx}{\mathrm{lfs}(x)^d}\right)$$

hides simple exponential in d

The Feature Size Measure: $\mu_P(\Omega) = \displaystyle\int_\Omega \frac{dx}{\mathrm{lfs}_P(x)^d}$



The size of an optimal mesh is given by the *feature size measure*.

$\mathrm{lfs}_P(x) :=$ Distance to second nearest neighbor in P.

number of vertices

$$\text{Optimal Mesh Size} = \Theta\left(\int_\Omega \frac{dx}{\mathrm{lfs}(x)^d}\right)$$

hides simple exponential in d

The Feature Size Measure: $\mu_P(\Omega) = \displaystyle\int_\Omega \frac{dx}{\mathrm{lfs}_P(x)^d}$

When is $\mu_P(\Omega) = O(n)$?

The size of an optimal mesh is given by the *feature size measure*.

$\mathrm{lfs}_P(x) :=$ Distance to second nearest neighbor in P.

number of vertices

$$\text{Optimal Mesh Size} = \Theta\left(\int_\Omega \frac{dx}{\mathrm{lfs}(x)^d}\right)$$

hides simple exponential in d

The Feature Size Measure: $\mu_P(\Omega) = \int_\Omega \frac{dx}{\mathrm{lfs}_P(x)^d}$

When is $\mu_P(\Omega) = O(n)$?

Pacing and the Empty Annulus Condition [S. 2012]

The Main Approximation Theorem

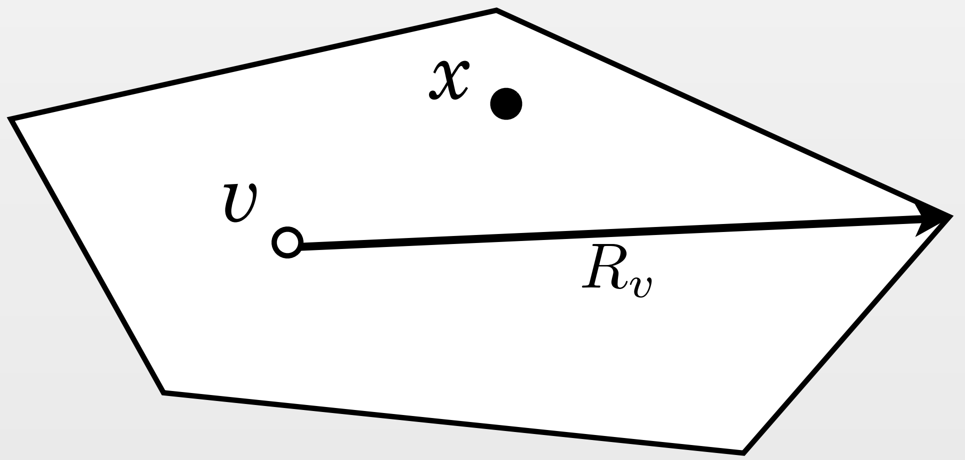
Theorem. *Let P be a point cloud and let M be an ε-refined mesh of P. Let $f \geq \frac{1}{c}\mathrm{lfs}_P$ be a t-Lipschitz function $\mathbb{R}^d \to \mathbb{R}$ for some constant $c > 0$. Let $\mathcal{F}$ be the sublevel filtration of f and let $\mathcal{V}$ be the Voronoi filtration of f on M. Then* $\mathrm{Dgm}\,\mathcal{V}$ *is a* $\left(1 + \frac{ct\varepsilon}{1-\varepsilon}\right)$*-approximation to* $\mathrm{Dgm}\,\mathcal{F}$.

Proof of the main theorem

Fix any $v \in M$ and any $x \in \mathrm{Vor}(v)$

$$f \geq \frac{1}{c}\mathrm{lfs}$$

$$R_v \leq \varepsilon \mathrm{lfs}(v)$$



$$\|v - x\| \leq R_v \leq \varepsilon \mathrm{lfs}(v) \leq c\varepsilon f(v)$$

$$f(x) \leq f(v) + t\|v - x\| \leq (1 + ct\varepsilon) f(v)$$

$$\forall v \in M, \mathrm{Vor}(v) \subseteq F_{(1+ct\varepsilon)f(v)}$$

$$\forall \alpha \geq 0, V_\alpha \subseteq F_{(1+ct\varepsilon)\alpha}$$

$$\|v - x\| \leq \frac{c\varepsilon}{1-\varepsilon} f(x)$$

$$f(v) \leq f(x) + t\|v - x\| \leq \left(1 + \frac{ct\varepsilon}{1-\varepsilon}\right) f(x)$$

$$\forall \alpha \geq 0, F_\alpha \subseteq F_{\left(1+\frac{ct\varepsilon}{1-\varepsilon}\right)\alpha}$$

$$\forall \alpha \geq 0, \quad V_{\alpha/\gamma} \subseteq F_\alpha \subseteq V_{\alpha\gamma} \quad \text{where } \gamma = 1 + \frac{ct\varepsilon}{1-\varepsilon}$$

Dgm $\mathcal{V}$ is a γ-approximation to Dgm $\mathcal{F}$.

The Main Approximation Theorem

Theorem. *Let P be a point cloud and let M be an ε-refined mesh of P. Let $f \geq \frac{1}{c}\mathrm{lfs}_P$ be a t-Lipschitz function $\mathbb{R}^d \to \mathbb{R}$ for some constant $c > 0$. Let $\mathcal{F}$ be the sublevel filtration of f and let $\mathcal{V}$ be the Voronoi filtration of f on M. Then* $\mathrm{Dgm}\,\mathcal{V}$ *is a* $\left(1 + \frac{ct\varepsilon}{1-\varepsilon}\right)$*-approximation to* $\mathrm{Dgm}\,\mathcal{F}$.

The Main Approximation Theorem

Theorem. *Let P be a point cloud and let M be an ε-refined mesh of P. Let $f \geq \frac{1}{c}\mathrm{lfs}_P$ be a t-Lipschitz function $\mathbb{R}^d \to \mathbb{R}$ for some constant $c > 0$. Let $\mathcal{F}$ be the sublevel filtration of f and let $\mathcal{V}$ be the Voronoi filtration of f on M. Then* $\mathrm{Dgm}\ \mathcal{V}$ *is a* $\left(1 + \frac{ct\varepsilon}{1-\varepsilon}\right)$*-approximation to* $\mathrm{Dgm}\ \mathcal{F}$.

Observations:
M can be made to have size *O(n)*.
The construction of *M* does not depend on *f*.
One mesh works for many functions.

We know how to do many things with meshes.

(Some of these should be useful for TDA).

We know how to do many things with meshes.

(Some of these should be useful for TDA).

Dealing with anisotropy.

We know how to do many things with meshes.

(Some of these should be useful for TDA).

Dealing with anisotropy.

Approximation of gradients.

We know how to do many things with meshes.

(Some of these should be useful for TDA).

Dealing with anisotropy.

Approximation of gradients.

Mesh Coarsening.

We know how to do many things with meshes.

(Some of these should be useful for TDA).

Dealing with anisotropy.

Approximation of gradients.

Mesh Coarsening.

Adaptive/Dynamic/Kinetic

We know how to do many things with meshes.

(Some of these should be useful for TDA).

Dealing with anisotropy.

Approximation of gradients.

Mesh Coarsening.

Adaptive/Dynamic/Kinetic

Geometric Separators

Mesh generation is a natural preprocess for TDA in low-dimensional Euclidean space.

Thank You!