Using Topological Data Analysis for Clustering Aptamers by Site Affinity

George Clare Kennedy

September 7, 2024

University of Iowa

Outline

Why TDA?

Mapper and Its Flavors

Aptamers

Section Map (!)

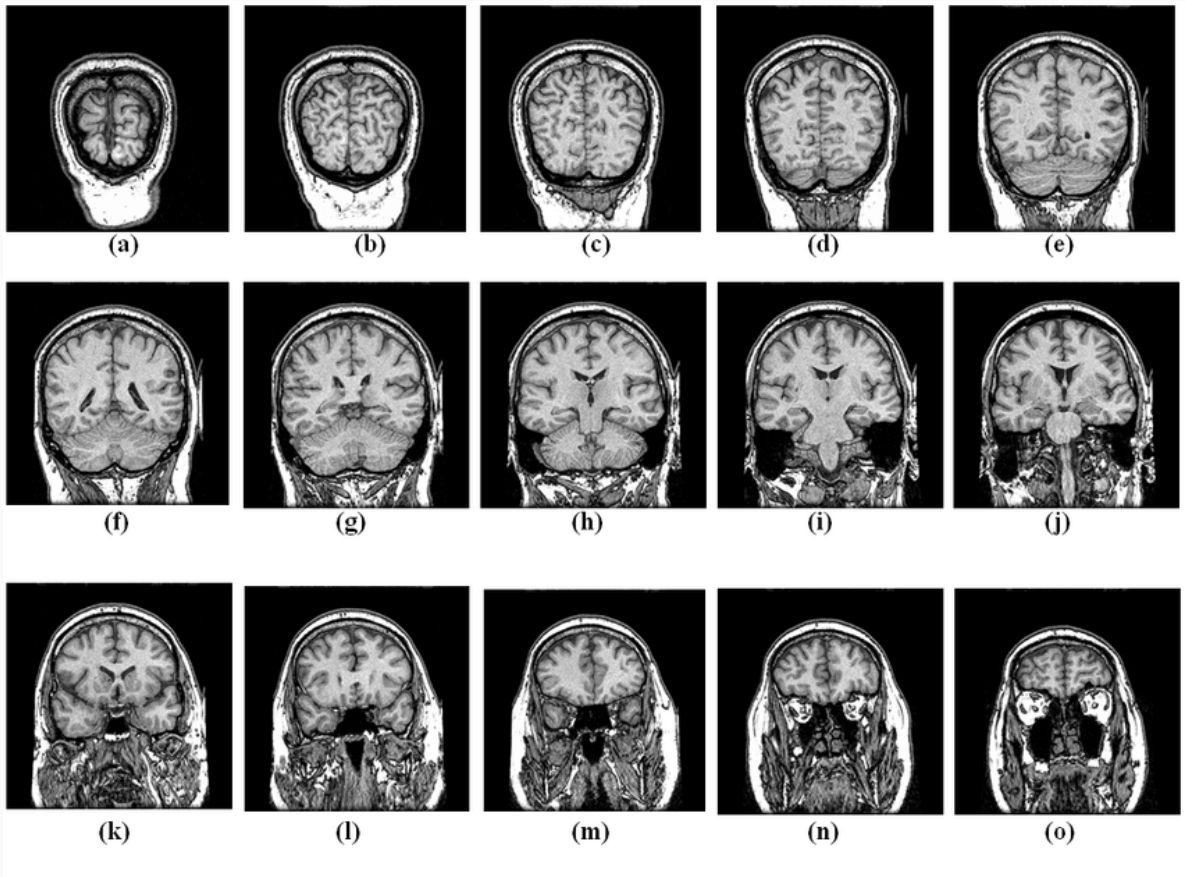
Why TDA?

Mapper and Its Flavors

Aptamers

Data is Big

- Modern techniques allow for rich data collection and storage
- Size of datasets can be enormous in both observations (rows) and variables (columns)



[7]

Data is Big

- Modern techniques allow for rich data collection and storage
- Size of datasets can be enormous in both observations (rows) and variables (columns)

[2]

Geometry is Hard

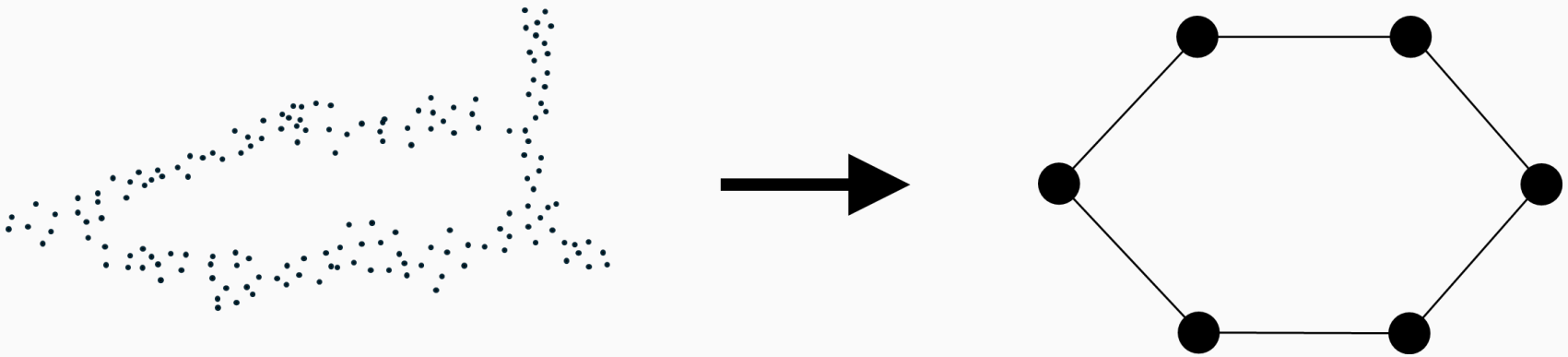
- High-dimensional space is extremely unintuitive
- If $V_n(r)$ is the volume of the n-dimensional ball with radius r, then for any $\varepsilon > 0$,

$$\lim_{n \to \infty} \frac{V_n(1-\varepsilon)}{V_n(1)} = 0$$

  i.e., the volume of balls lives almost entirely at the boundary
- Trying to analyze many characteristics creates combinatorial problems ($n!$ is big!)

Toning It Down

- Broad idea: high dimensions $\Longrightarrow$ low dimensions
- More specific idea: build a simplicial complex
- Simpler idea: build a 1-dimensional simplicial complex (that is, a graph)
- Enter: the Mapper algorithm (Singh et al, 2007) [8]



Section Map (!)

Why TDA?

Mapper and Its Flavors

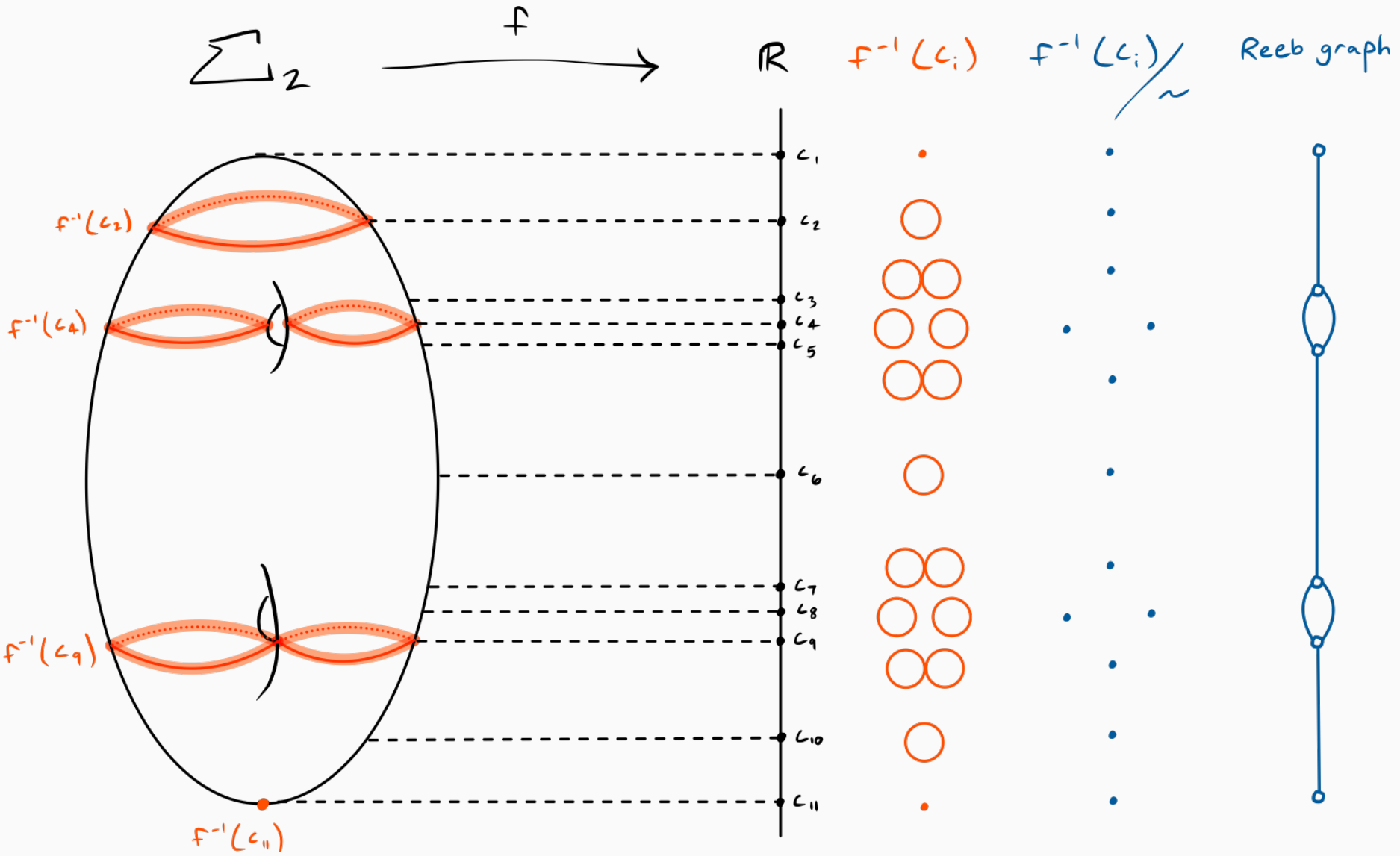
Aptamers

Motivation: Reeb Graph

- Idea: construct graph reflecting level sets of a "filter" function
- Formally, given a topological space X and a continuous function $f : X \to \mathbb{R}$, define an equivalence relation $\sim$ on X where $x \sim y$ if x and y live in the same connected component of a level set $f^{-1}(c)$ for some $c \in \mathbb{R}$.
- The **Reeb graph**[1] is $X/\sim$, taken with the quotient topology.

[1] Despite names this is not always a graph

Motivation: Reeb Graph



Mapper: Original Flavor

- How can we apply this to the discrete setting?
- Topological space $X \implies$ point cloud P (a discrete set of points in a space)
- Filter function: $f : P \to \mathbb{R}$
- Level sets of points $\implies$ level sets of overlapping intervals
- Connected components $\implies$ clusters
- Quotient space $\implies$ intersection graph

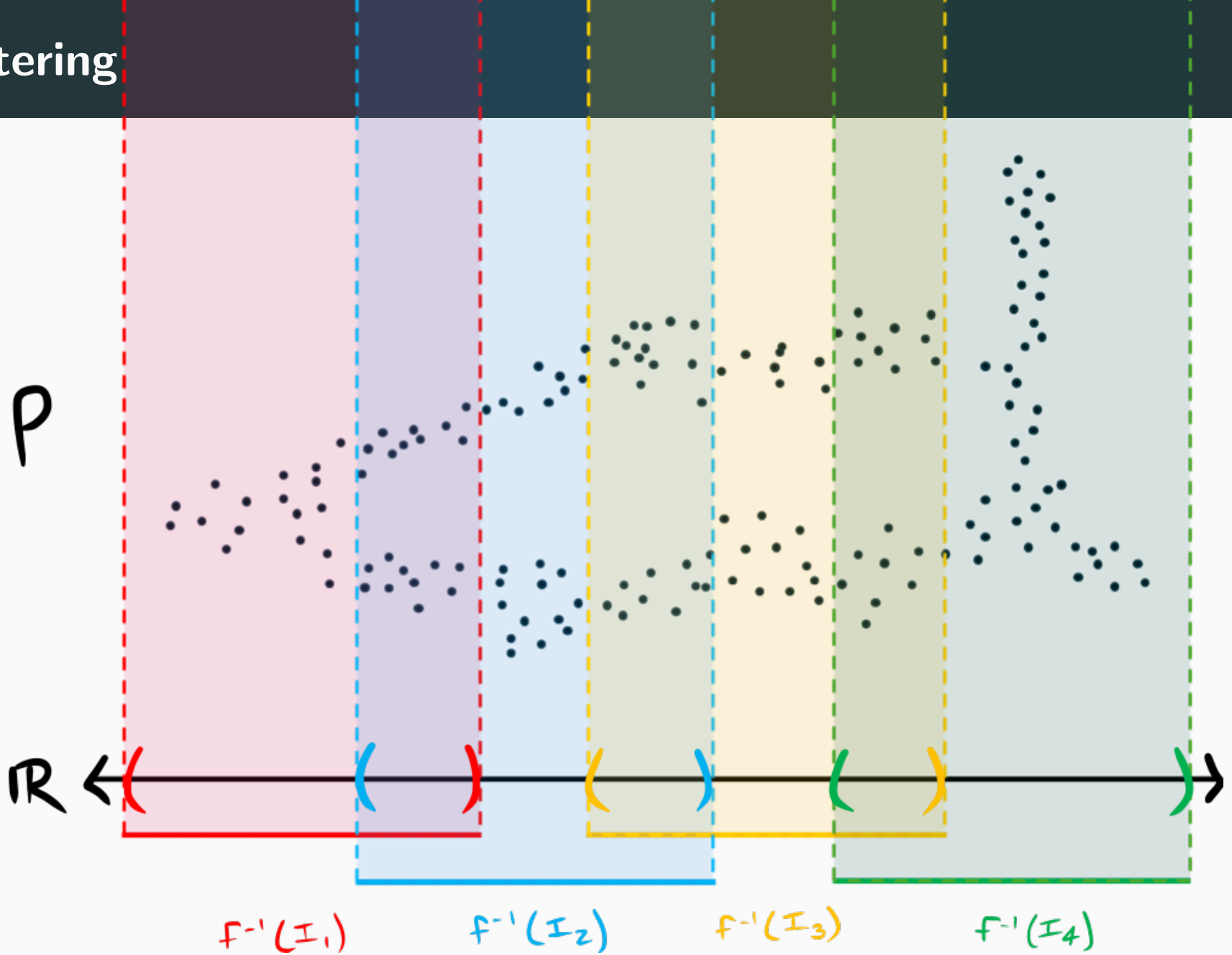
Original Mapper Algorithm: Overview

- Ingredients:
  - Point cloud P
  - Filter function $f : P \to \mathbb{R}$
  - Collection of overlapping intervals $\{I_1, \ldots, I_k\}$
  - Clustering algorithm
- Output:
  - Finite graph M

Point Cloud

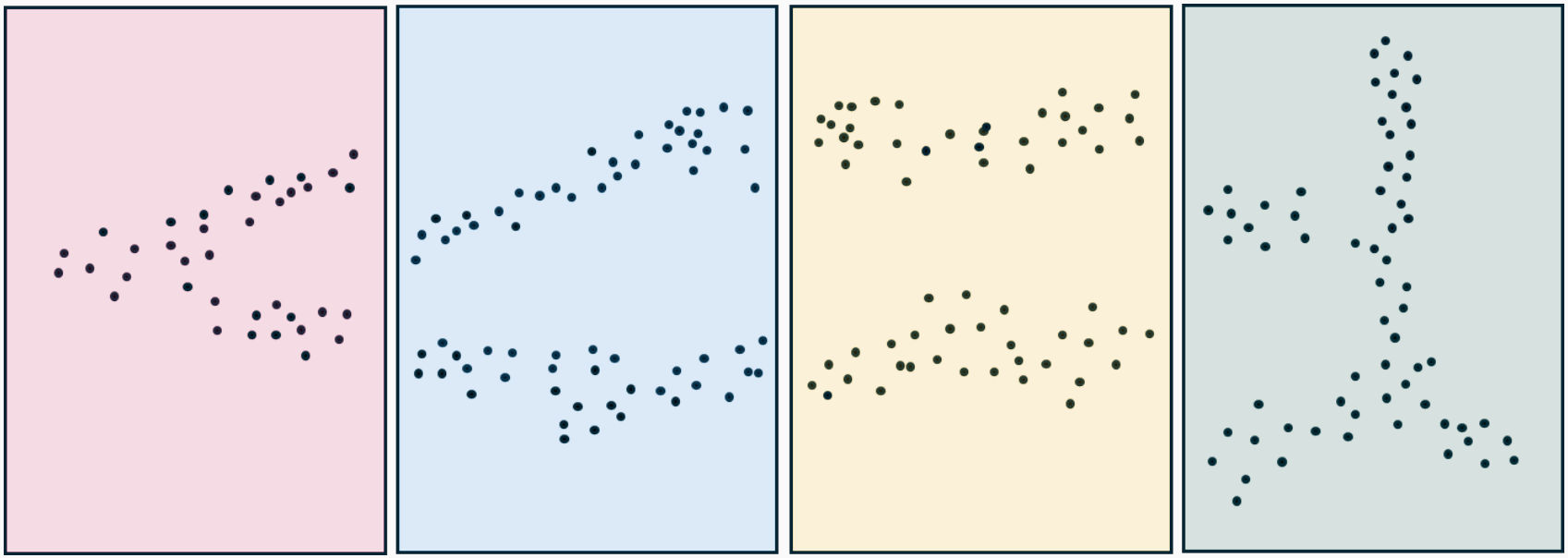
Our example point cloud will live in $\mathbb{R}^2$:

Filtering

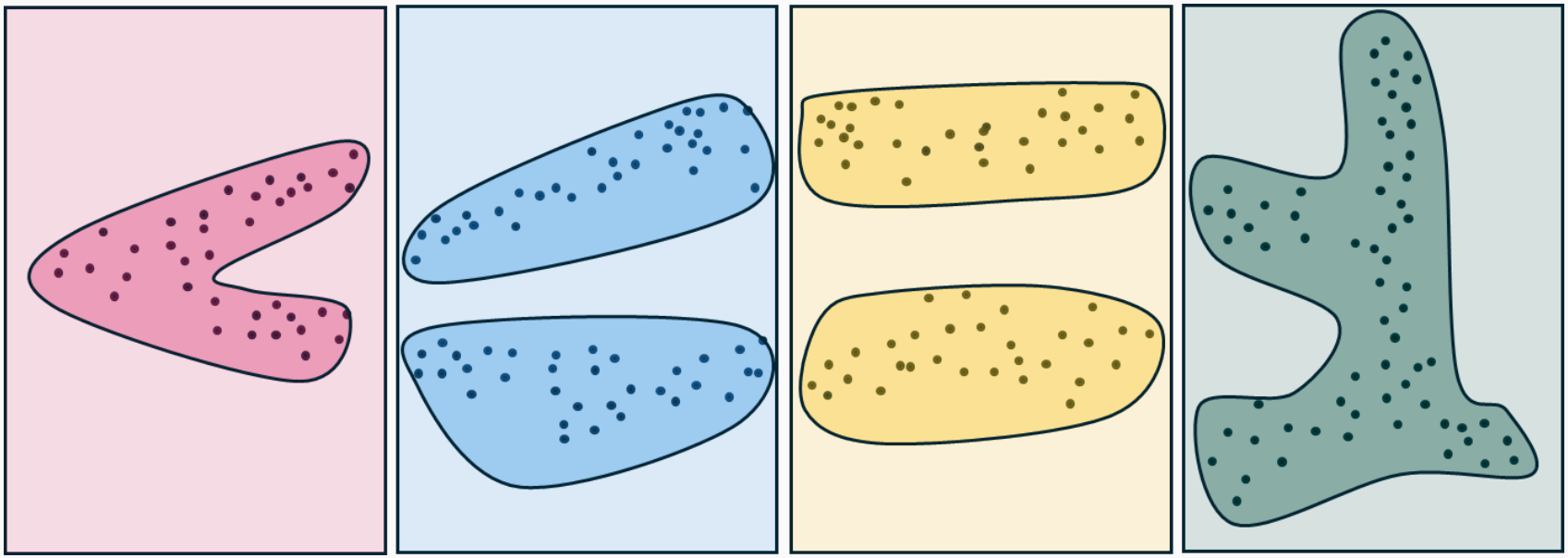


Here, $f : P \to \mathbb{R}$ is projection to the x-coordinate.

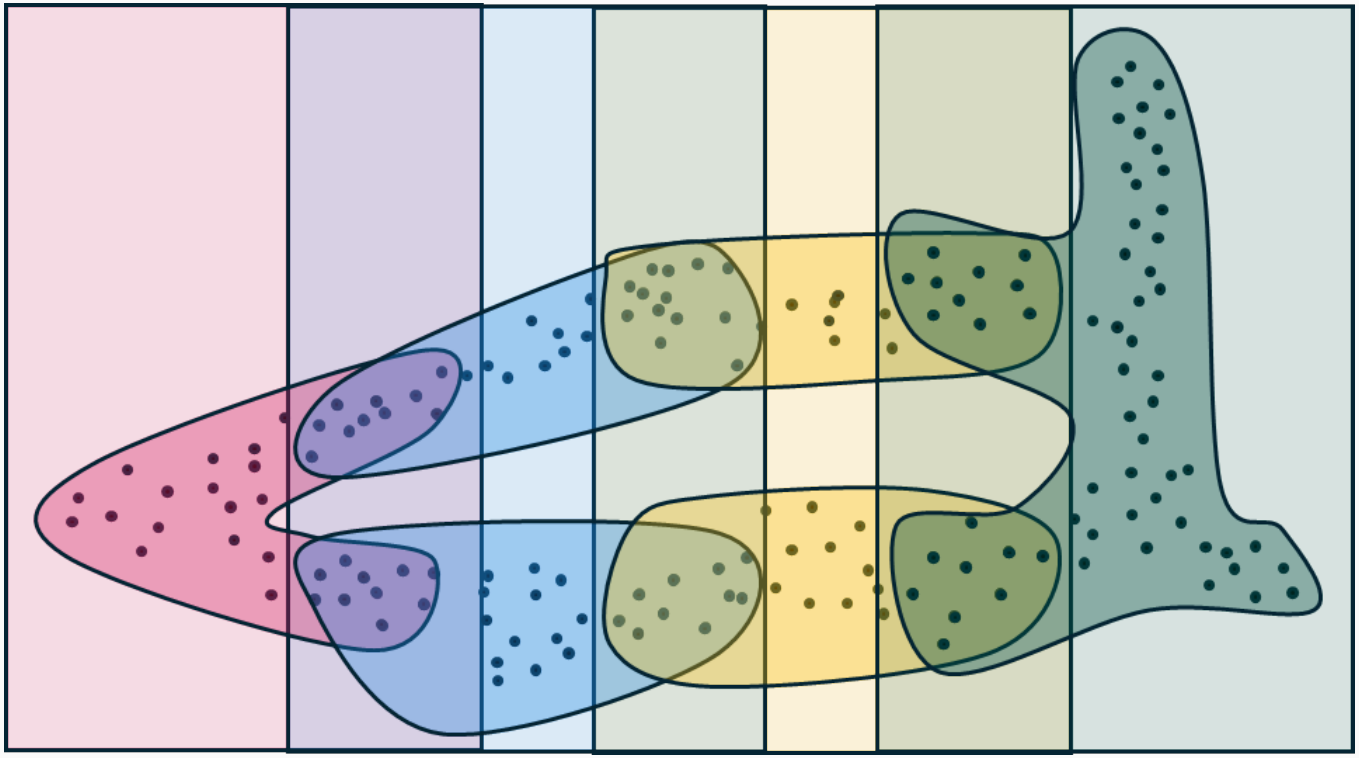
Filtering



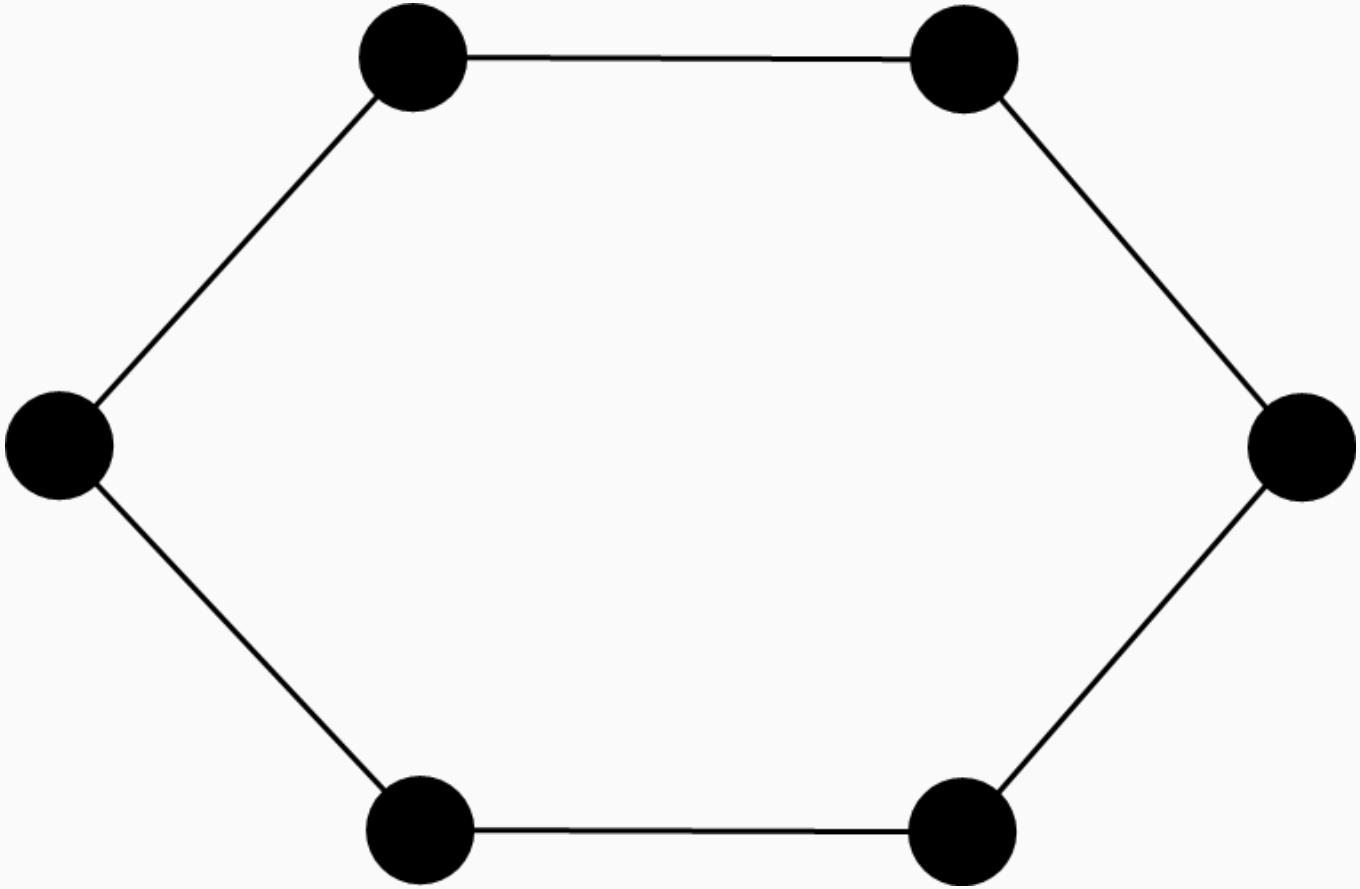
Clustering



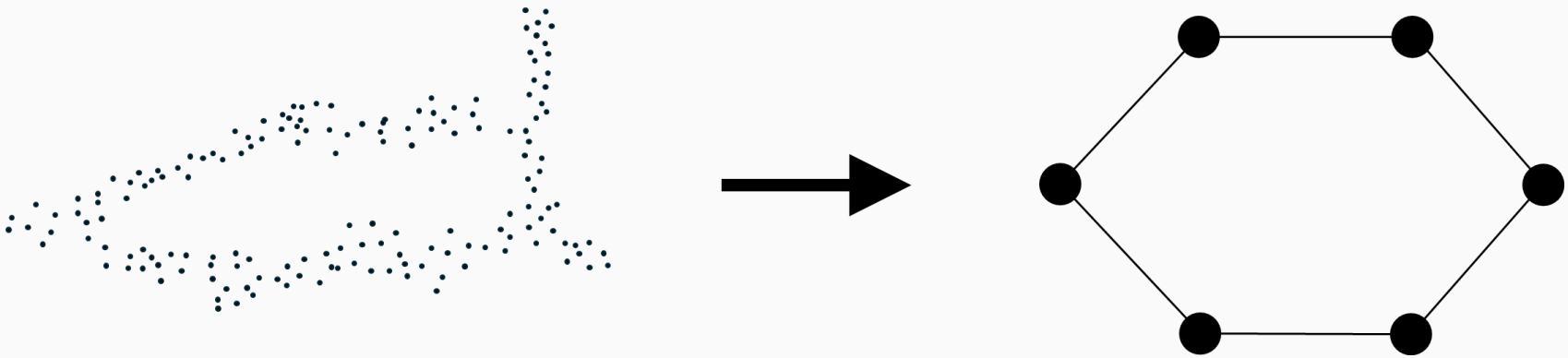
Overlapping Clusters



Output



Original Mapper: Problems



- We lost so much stuff! Clustering takes work.
- Graphs are abstract combinatorial structures; they convey no other information
- Potentially interesting features can "bypass" the filter
- Large number of parameters complicates effectiveness
- Output heavily depends on choice of clustering method

Ballmapper (Dłotko, 2019)

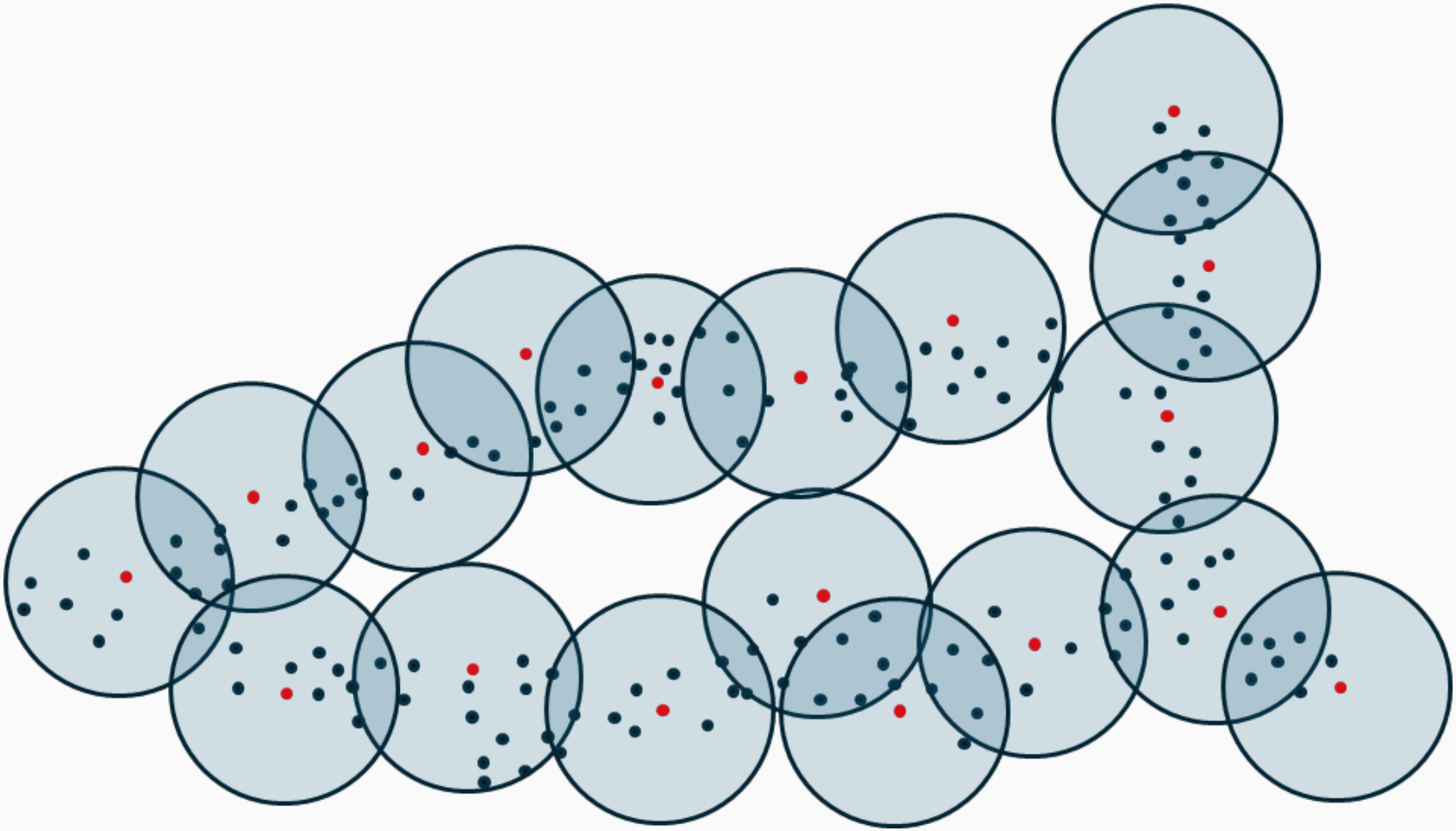
- Original flavor has a lot of choices to make
- Idea: come up with a one (ish)-parameter Mapper
- **Ballmapper** [6]: in place of a conventional filter, cover the dataset with overlapping ε-balls
- Specifically, we want a cover $C = \bigcup_i B(x_i, \varepsilon)$ such that:
  - Every datapoint x is contained in $B(x_i, \varepsilon)$ for some x_i
  - If x_j is a ball center, then the only ball containing it is $B(x_j, \varepsilon)$
- Graph construction unchanged

Ballmapper Algorithm: Overview

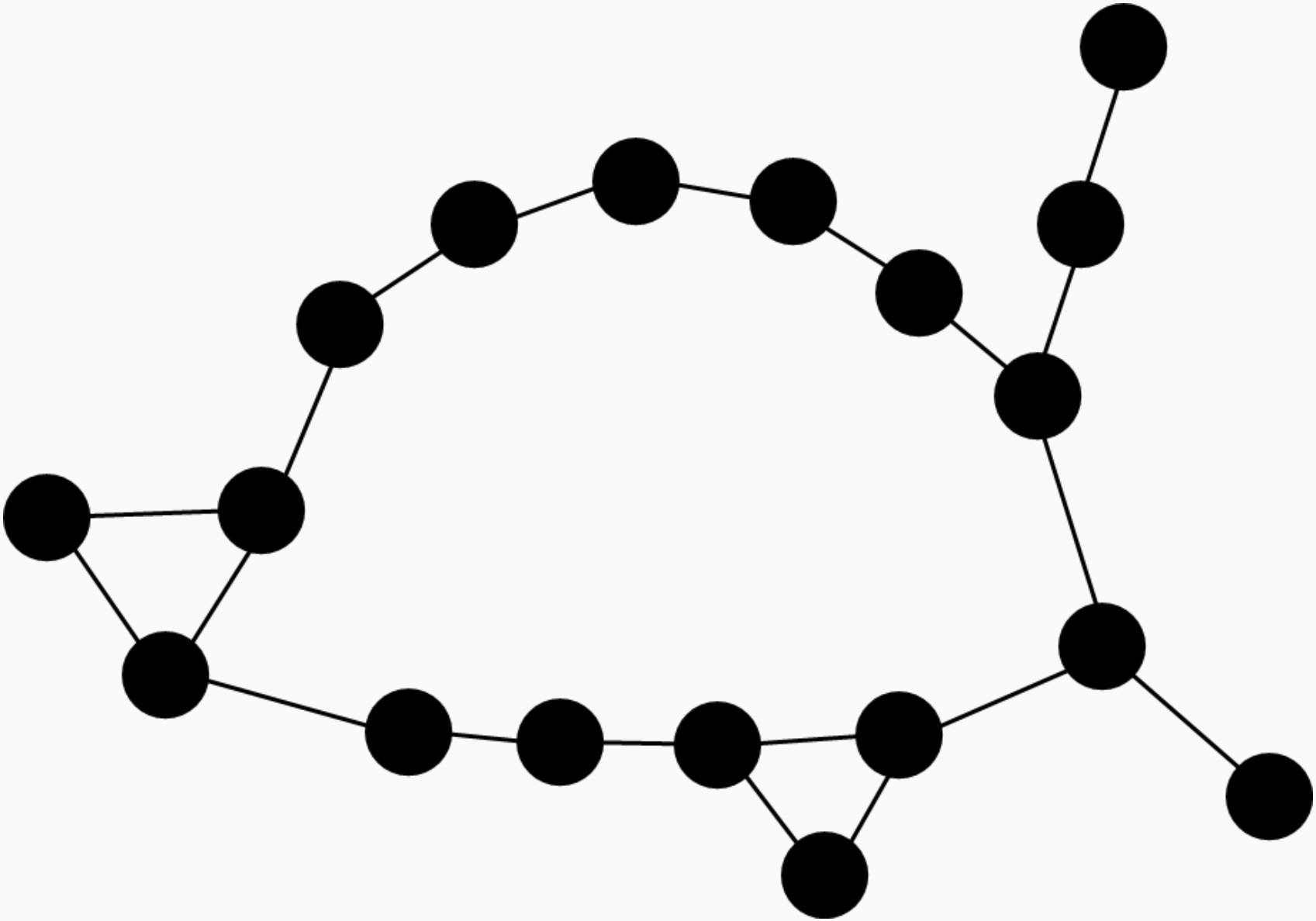
- Ingredients:
  - Point cloud P with distance function d
  - Ball radius $\varepsilon > 0$
  - Suitable cover of data $\bigcup B_\varepsilon(x_i)$
- Output:
  - Finite graph BM

Balling the Data

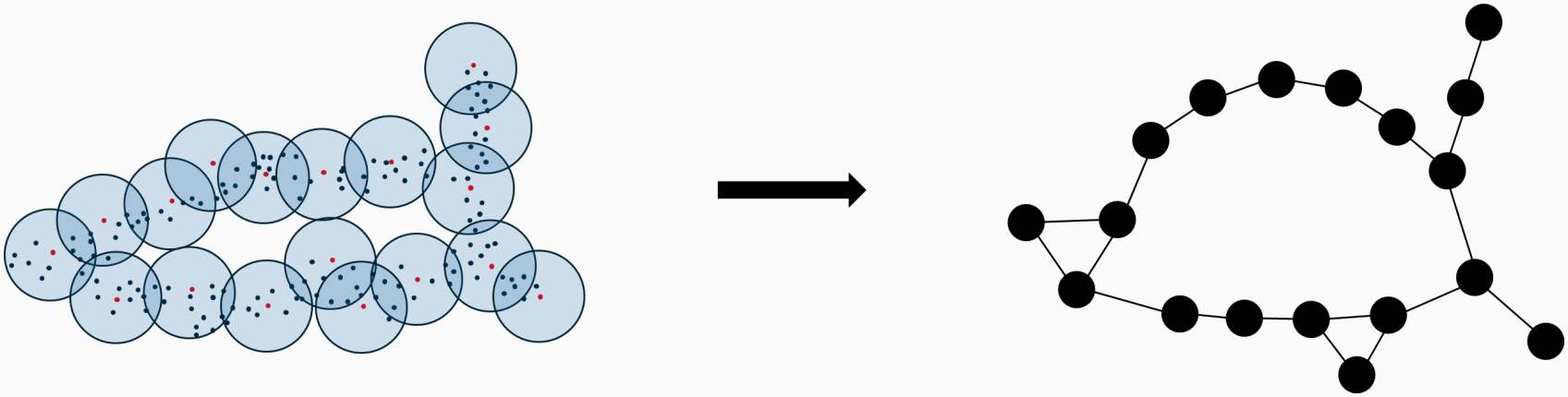
- Can be done quickly with a greedy method
- May also use k-means clustering, etc.



Output



Ballmapper: Problems



- Output is still just a graph!
- Balls become black boxes
- Can be quite noisy if ε is too small, and meaningless for ε too large

Refined Ballmapper

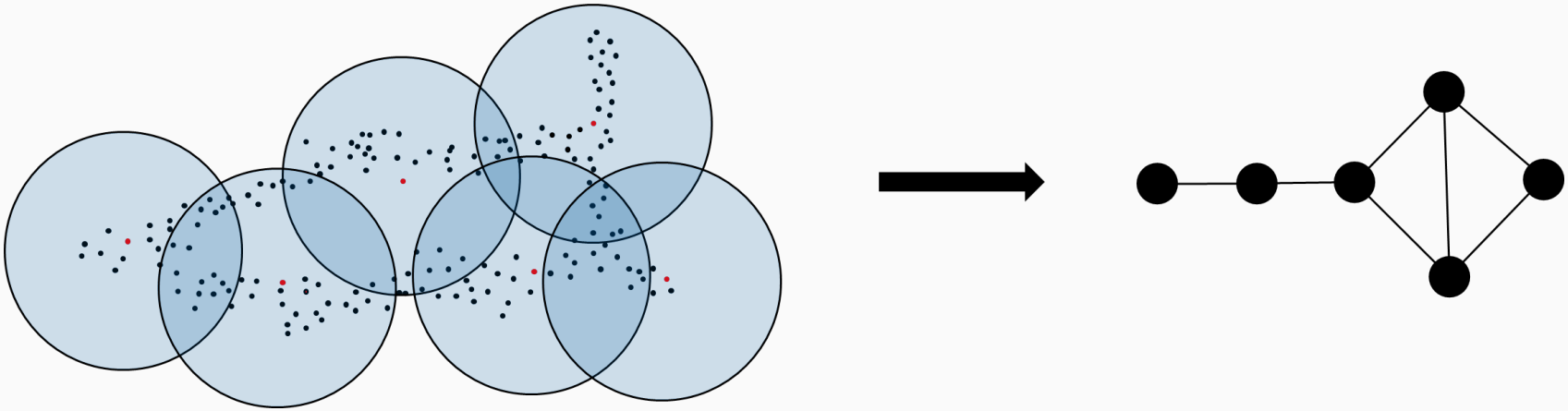
- Idea: combine Ball filtering and Original clustering
- Bin by balling, then cluster within balls as in Original
- Allows for comparison of two different metrics[2] at the same time on the same data

[2] Could be not strictly metrics!

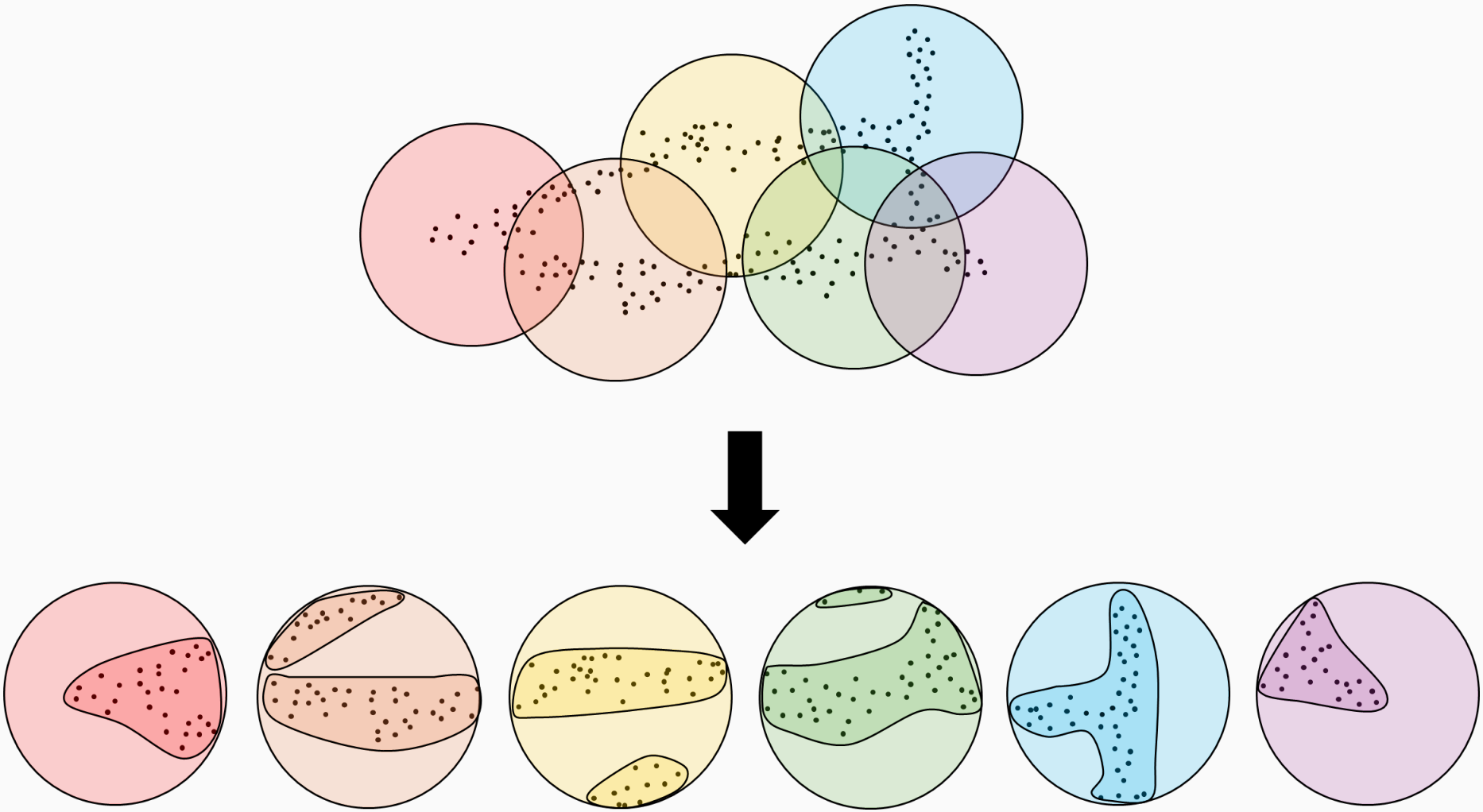
Refined Ballmapper: Overview

- Ingredients:
  - Point cloud P with distance function d
  - Ball radius $\varepsilon > 0$
  - Suitable cover of data $\bigcup B_\varepsilon(x_i)$
  - Clustering algorithm
- Output:
  - Finite graph *RBM*

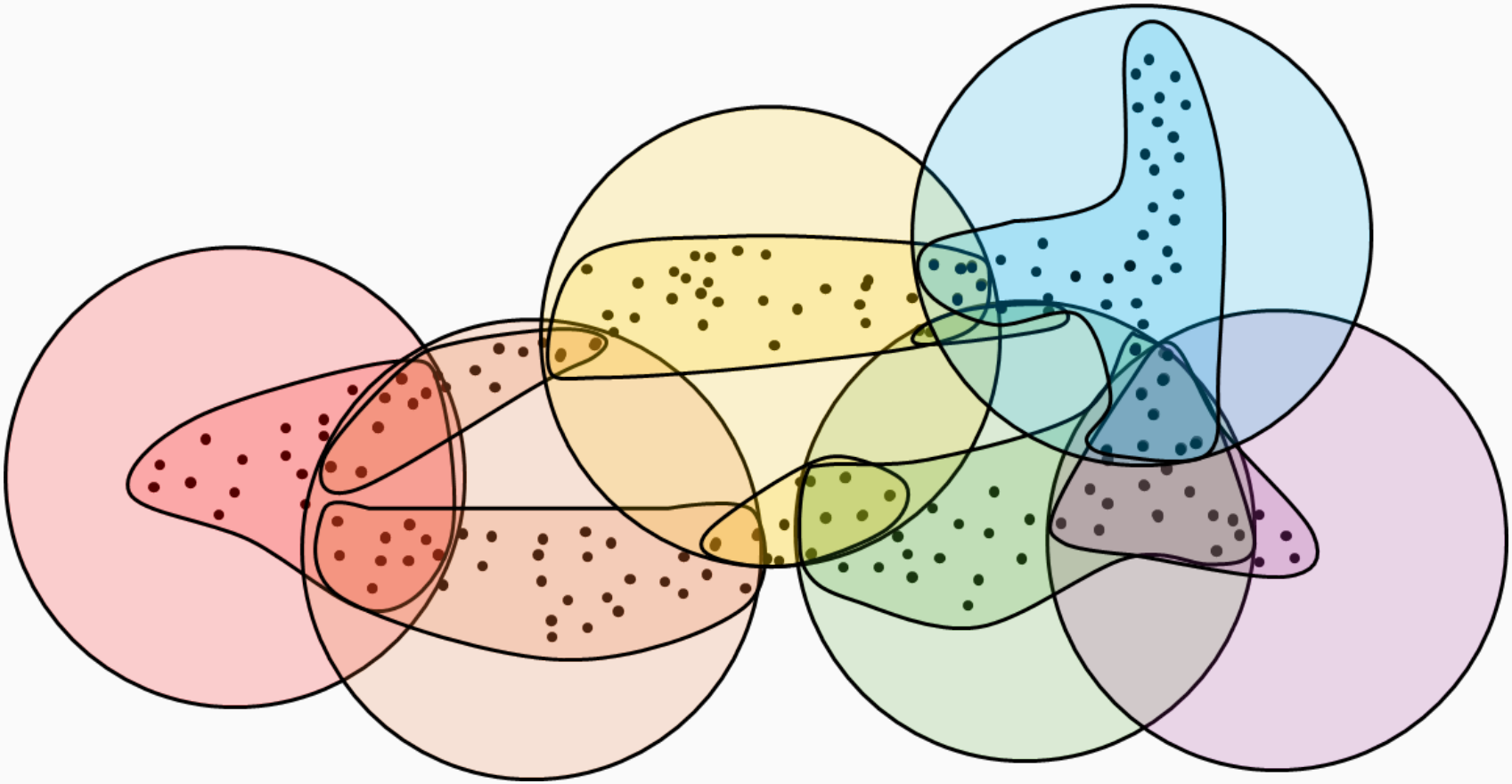
Standard Ballmapper



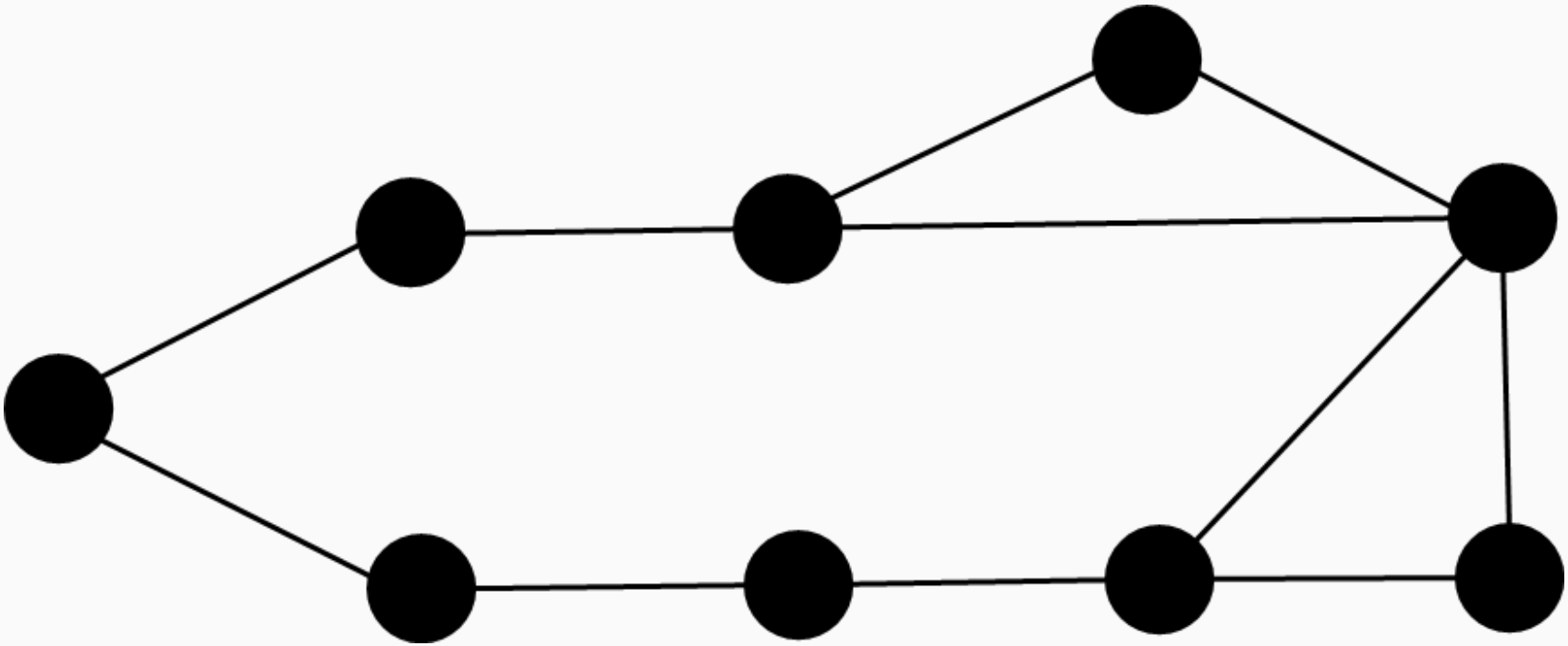
Balls as Bins



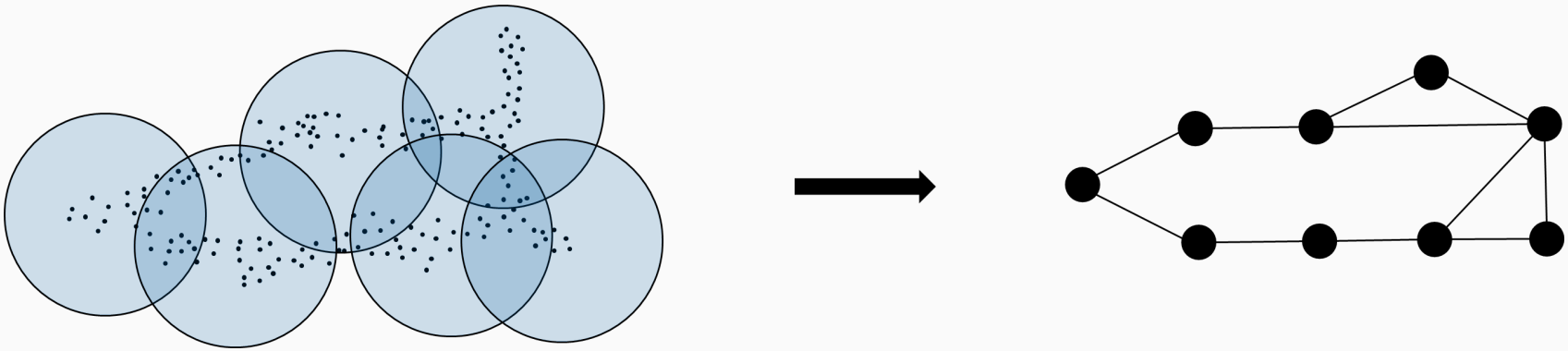
Balls Together



Output



Refined Ballmapper: Problems



- Choice of cover now strongly influences meaningfulness of clusters
- Choice of clustering algorithm is still an issue
- Still have that graph problem!

Section Map (!)

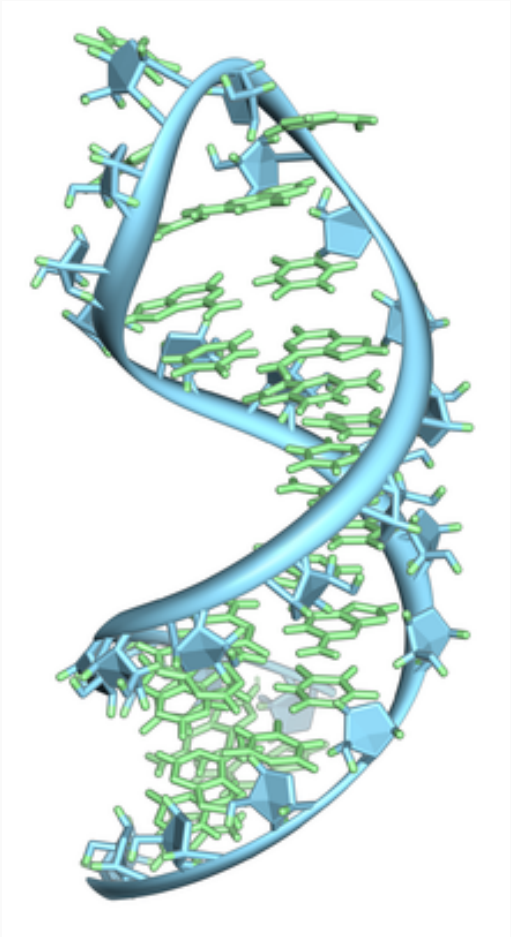
Why TDA?

Mapper and Its Flavors

Aptamers

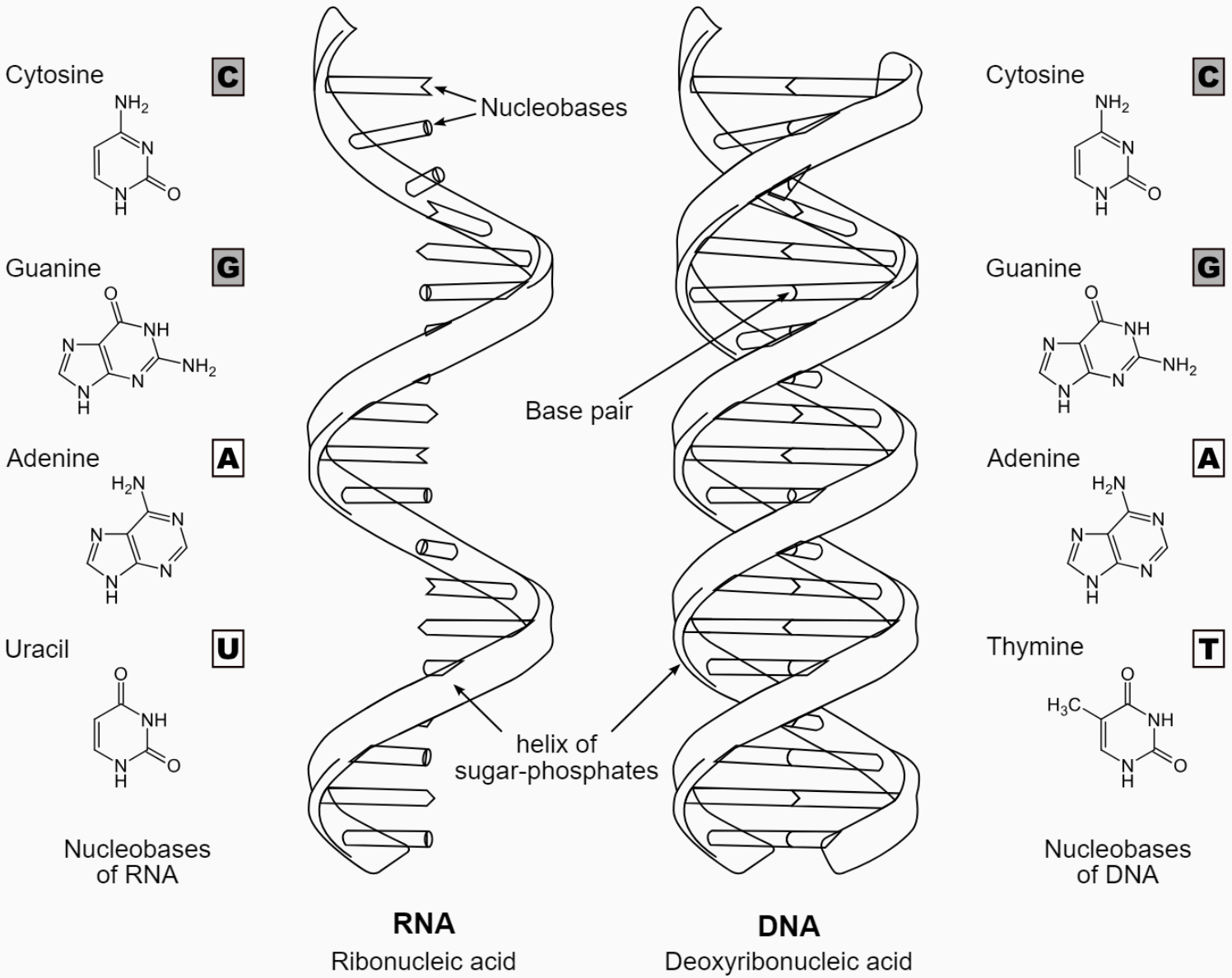
Biology CliffNotes CliffNotes

- RNA (ribonucleic acid) molecules are polymers which carry genetic information and have additional structure



[5]

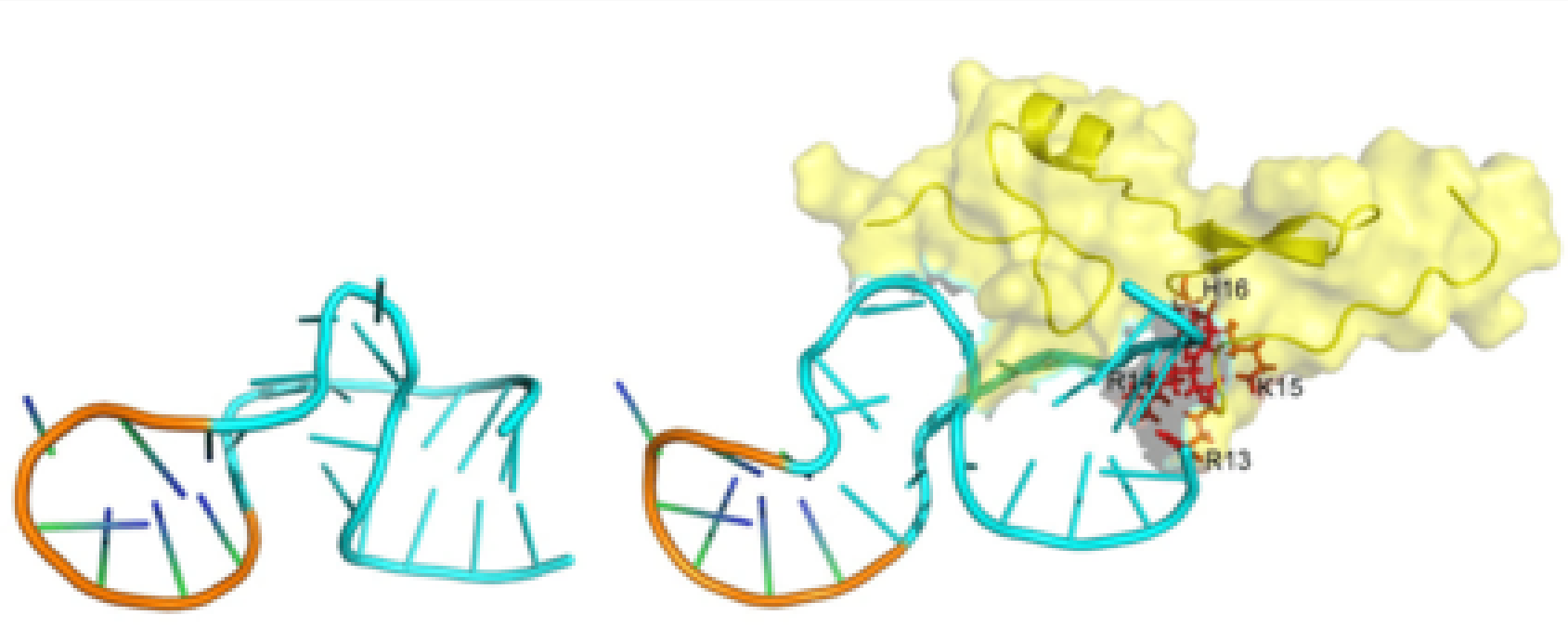
RNA: Structure



[3]

What Is an Aptamer?

- Aptamers are synthetic RNA molecules that bind to a specific target
- Similar function to antibodies, but much smaller
- **Genetic information not expressed**



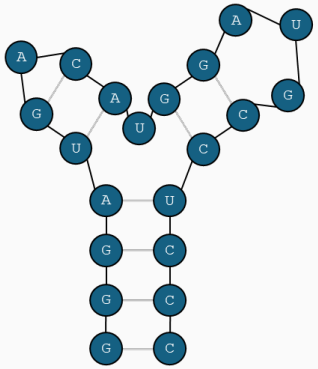
[4]

Comparing Aptamers

- For TDA to work we need a notion of distance among aptamers
- Aptamers have two characteristics: their genetic information and their structure
- Distance between sequences: Levenshtein distance
- Distance between structures: tree distance

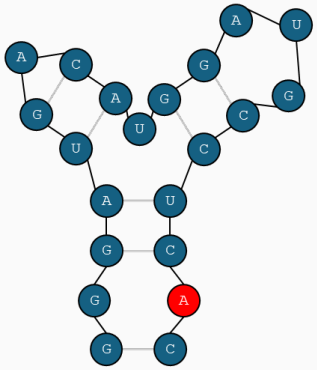
Aptamer Metrics

Aptamer X



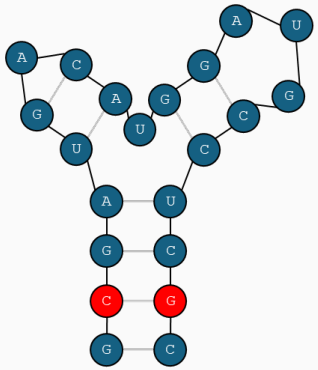
GGGAUGACAUGGAUGCCUCCC
((((((.)).((...))))))

Aptamer Y



GGGAUGACAUGGAUGCCUCAC
(.((((.)).((...)))).)

Aptamer Z



GGGCUGACAUGGAUGCCGCCC
((((((.)).((...))))))

δ_{Lev}: Levenshtein distance

δ_{Tree}: Tree distance

$\delta_{Lev}(X, Y) = 1$

$\delta_{Tree}(X, Y) = 1$

$\delta_{Lev}(X, Z) = 2$

$\delta_{Tree}(X, Z) = 0$

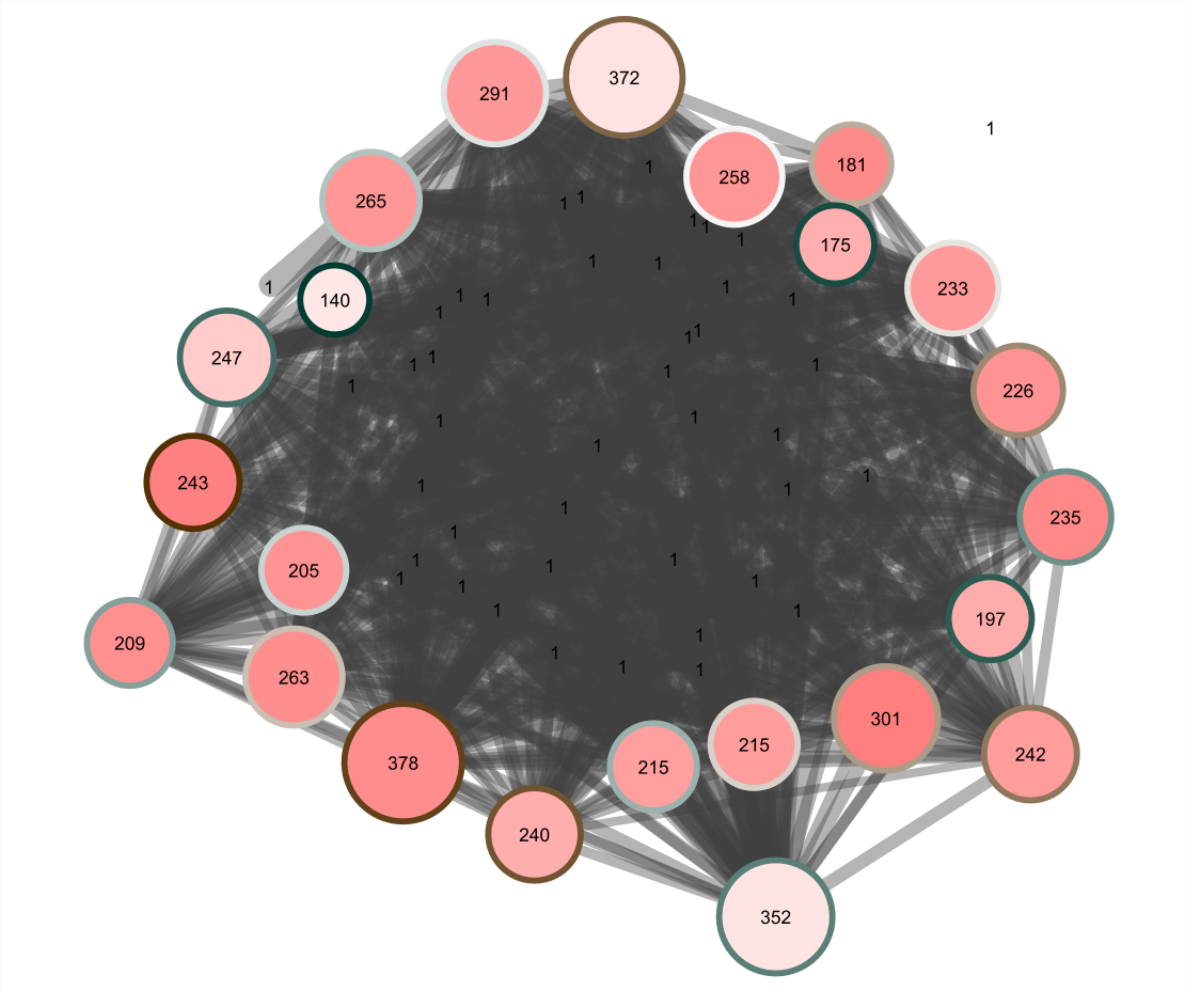
[9]

Aptamer Clustering With Mapper

- Flavor: Refined Ballmapper
- Idea: Ball using tree distance, cluster using Levenshtein distance
- Clustering method: single linkage hierarchical
- Vertices of the graph are clusters of aptamers related in both sequence and structure
- Graph structure may highlight families of aptamers or reveal other insights

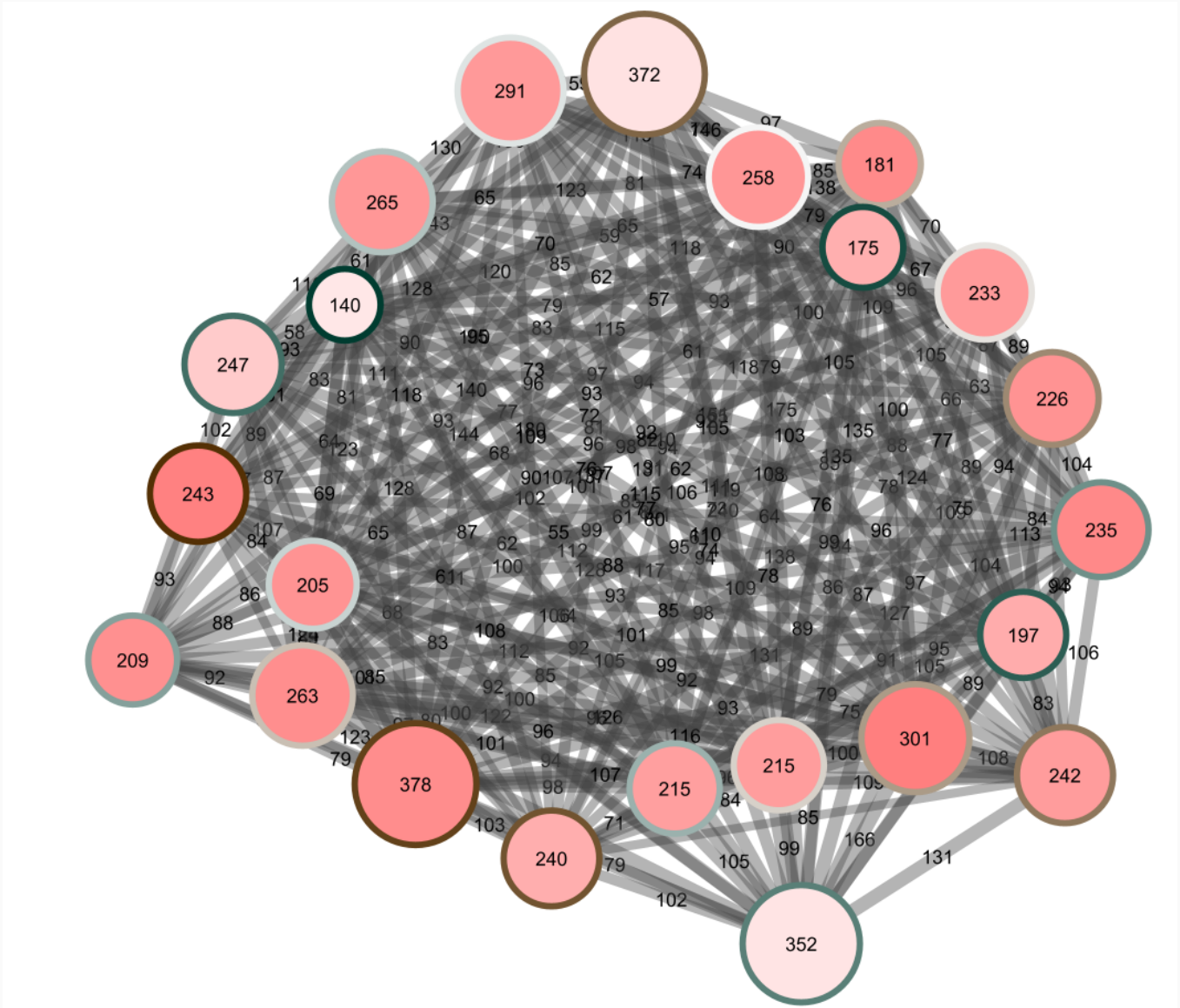
Big Mapptamer Graph

- Number of points: 800
- Ball radius ε: 10
- Clustering method: single linkage



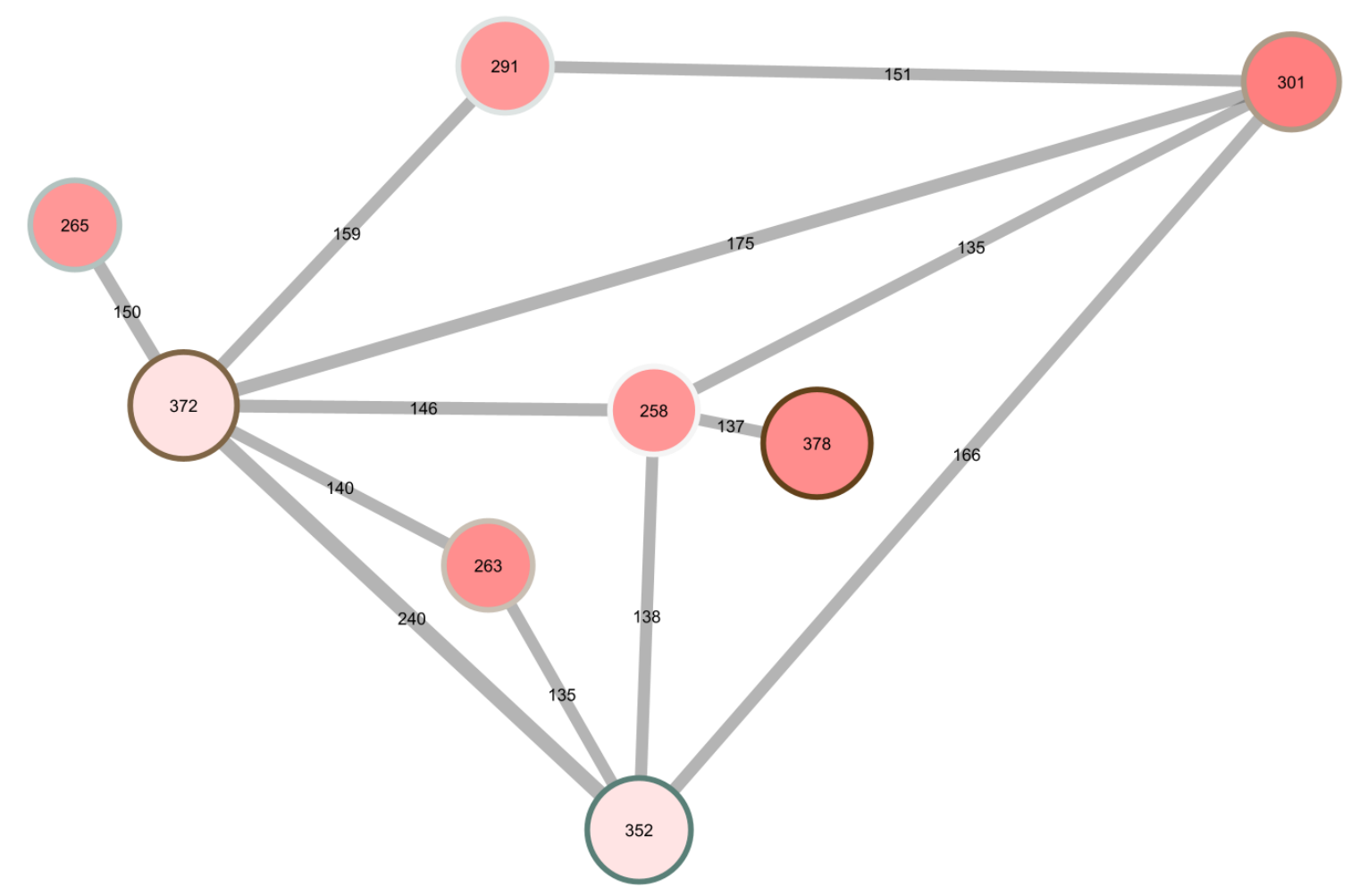
Vertex-pruned Mapptamer Graph

Vertices with < 100 datapoints eliminated



Vertex and Edge-pruned Mapptamer Graph

Vertices with < 250 datapoints and edges with $< 50\%$ overlap strength eliminated



[1] F. Chazal et al. **"Persistence-based clustering in Riemannian manifolds.".** In: (2013).

[2] Wikimedia Commons. **Climate Station (21123613446).** 2014. URL: https://commons.wikimedia.org/wiki/File:Climate_Station_%2821123613446%29.jpg.

[3] Wikimedia Commons. **Difference DNA RNA-EN BW.** 2010. URL: https://commons.wikimedia.org/wiki/File:Difference_DNA_RNA-EN_BW.svg.

[4] Wikimedia Commons. **Pegaptanib induced fit bindin.** 2021. URL: https://commons.wikimedia.org/wiki/File:Pegaptanib_induced_fit_binding.png.

[5] Wikimedia Commons. **Pre-mRNA-1ysv-tubes.** 2009. URL: https://commons.wikimedia.org/wiki/File:Pre-mRNA-1ysv-tubes.png.

[6] Paweł Dłotko. **Ball mapper: a shape summary for topological data analysis.** 2019. arXiv: 1901.07410

More Aptamer Work

- Is this a useful idea?
- Can we find an "optimal" set of parameters?
- Compare results versus those obtained by state-of-the-art aptamer algorithms (AptamerRunner)
- More kinds of aptamers (ssDNA)?

Other Clustering Algorithms

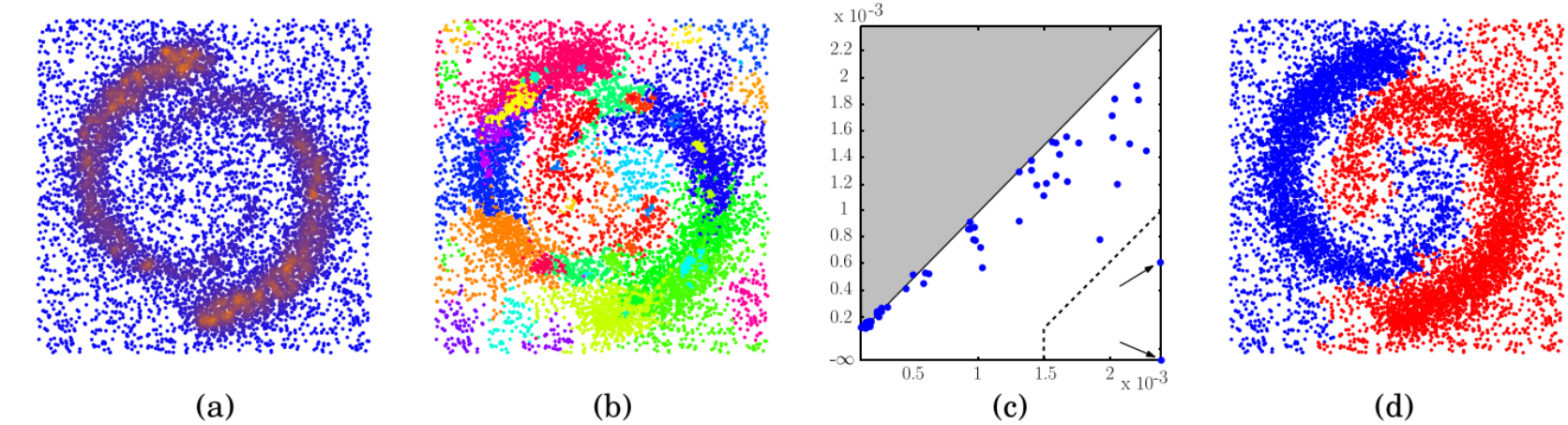
- Hierarchical:
  - Complete linkage:

$$\ell(A, B) = \max_{a \in A, b \in B} \{d(a, b)\}$$

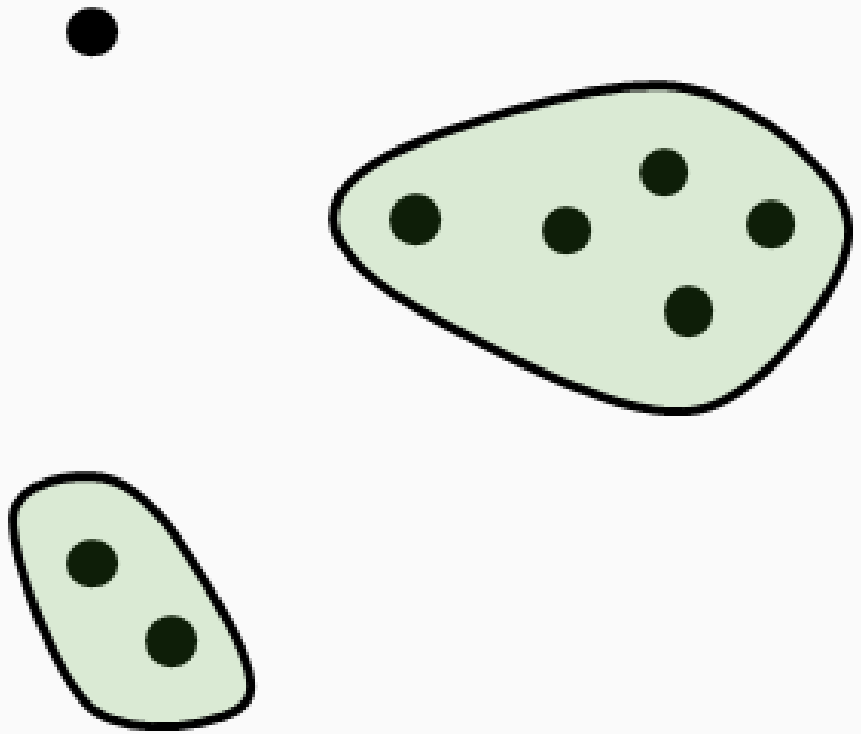
  - (Unweighted) average linkage:

$$\ell(A, B) = \frac{1}{|A||B|} \sum_{a \in A} \sum_{b \in B} d(a, b)$$

- k-means
- Topological Mode Analysis Tool (ToMATo) [1]



Speaking of Clustering Methods...



There are a large variety of clustering algorithms, including:

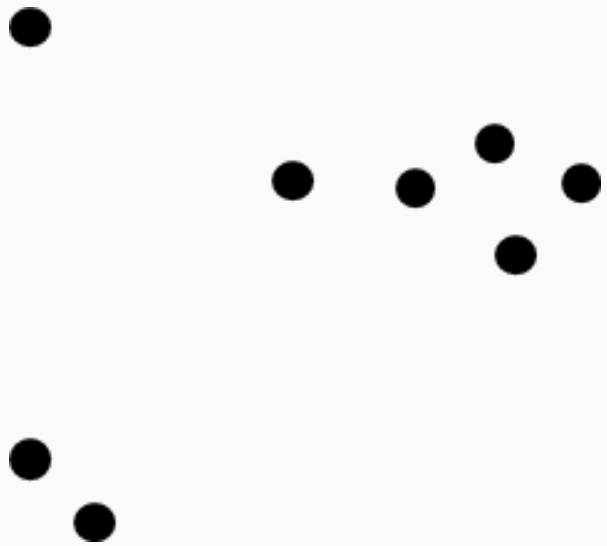
- Hierarchical clustering
- k-means clustering (centroid based)
- DBSCAN (density based)
- Topological clustering

How to Cluster, Hierarchically

- Two ingredients:
  - A discrete set of points X, equipped with a distance function $d : X \times X \to \mathbb{R}$
  - A **linking criterion**, a (partial) function $\ell : \mathscr{P}(X) \times \mathscr{P}(X) \to \mathbb{R}$ which measures the distance between disjoint sets of points
- Process:
  1. Begin by considering each point as an individual cluster as part of a collection $\mathcal{C}$.
  2. Find the two $A, B \in \mathcal{C}$ that minimize ℓ. Merge these clusters.
  3. Repeat step 2 until $\mathcal{C}$ consists of up to a single cluster.
- We record the resulting hierarchy of clusters using a **dendrogram**, which records information about the merging process.

Example: Single Linkage Clustering

- Data: 8 points in $\mathbb{R}^2$, with the usual metric



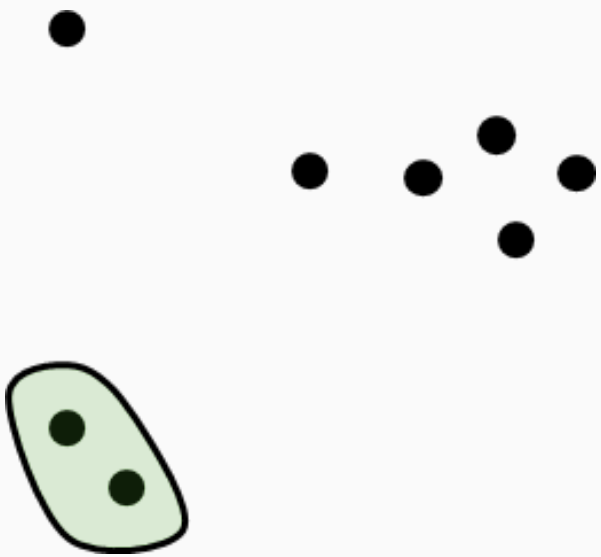
- Linking criterion:

$$\ell(A, B) = \min_{a \in A, b \in B} \{d(a, b)\}$$

- Starting dendrogram:

Single Linkage Clustering In Action

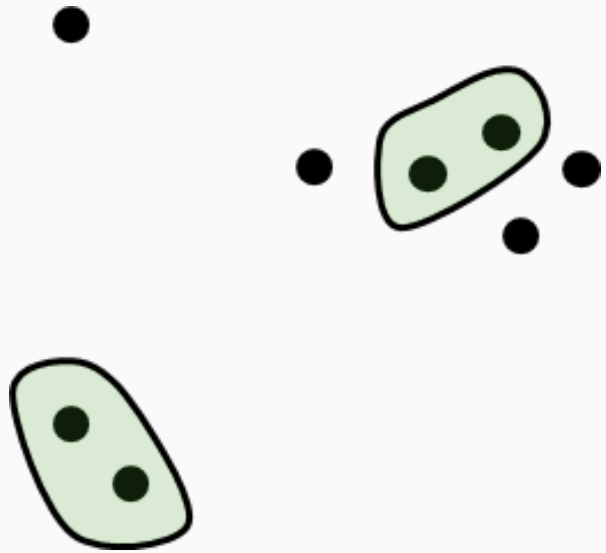
$$\ell(A, B) = \min_{a \in A, b \in B} \{d(a, b)\}$$

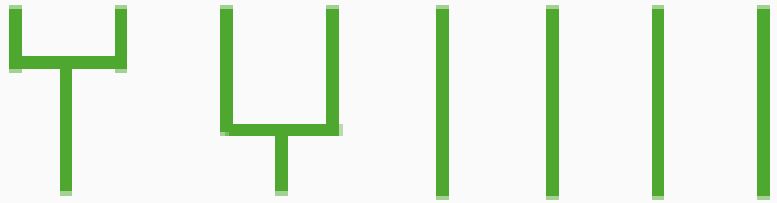


We merge edges at a height proportional to the corresponding value of ℓ (the merge height):

Single Linkage Clustering In Action

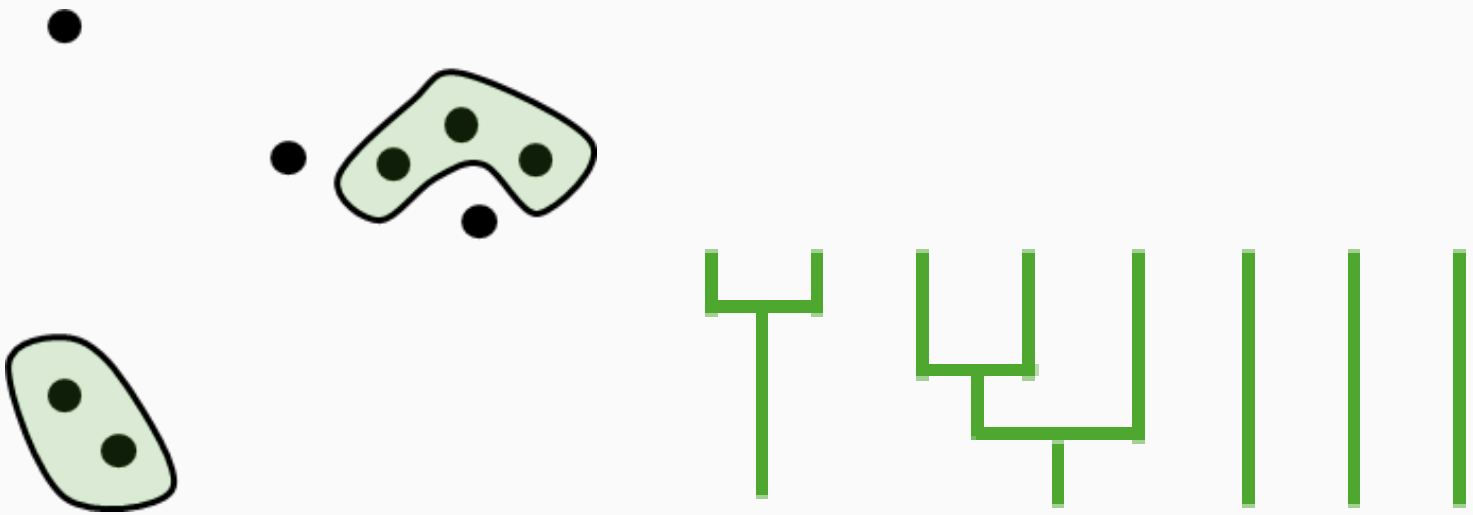
$$\ell(A, B) = \min_{a \in A, b \in B} \{d(a, b)\}$$





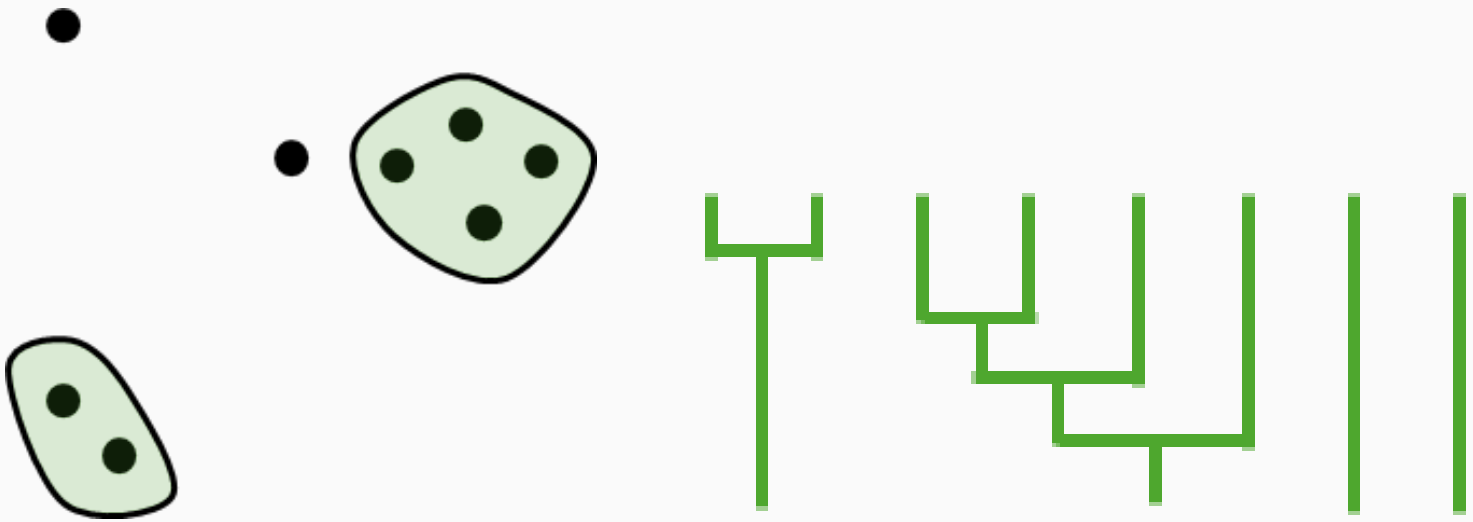
Single Linkage Clustering In Action

$$\ell(A, B) = \min_{a \in A, b \in B} \{d(a, b)\}$$



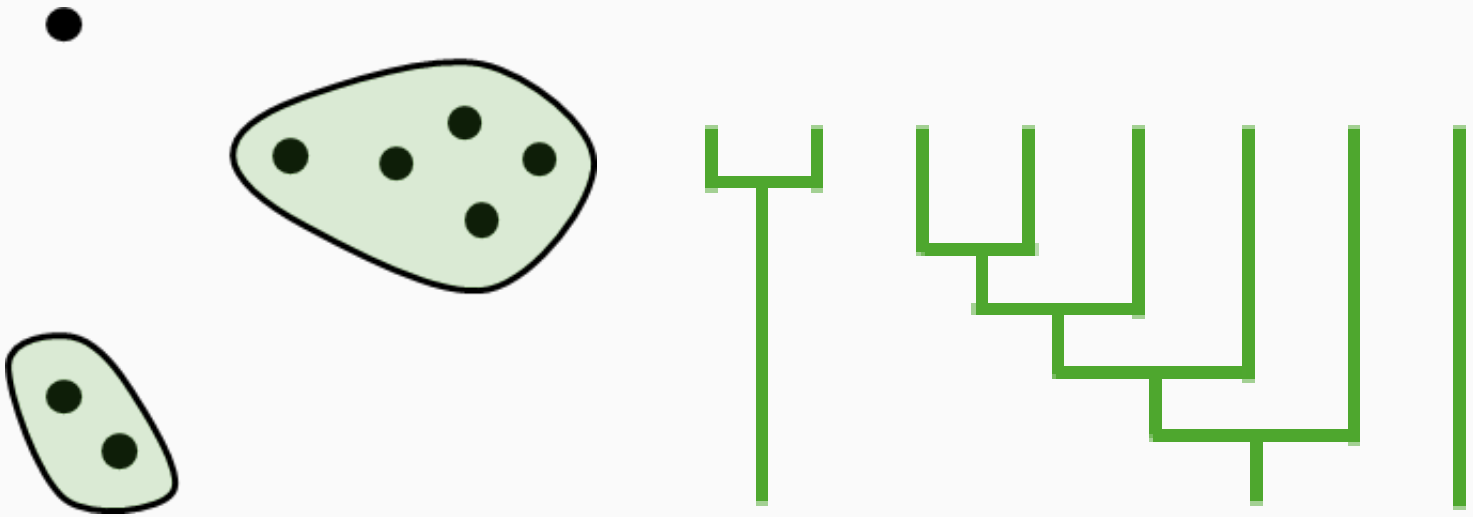
Single Linkage Clustering In Action

$$\ell(A, B) = \min_{a \in A, b \in B} \{d(a, b)\}$$



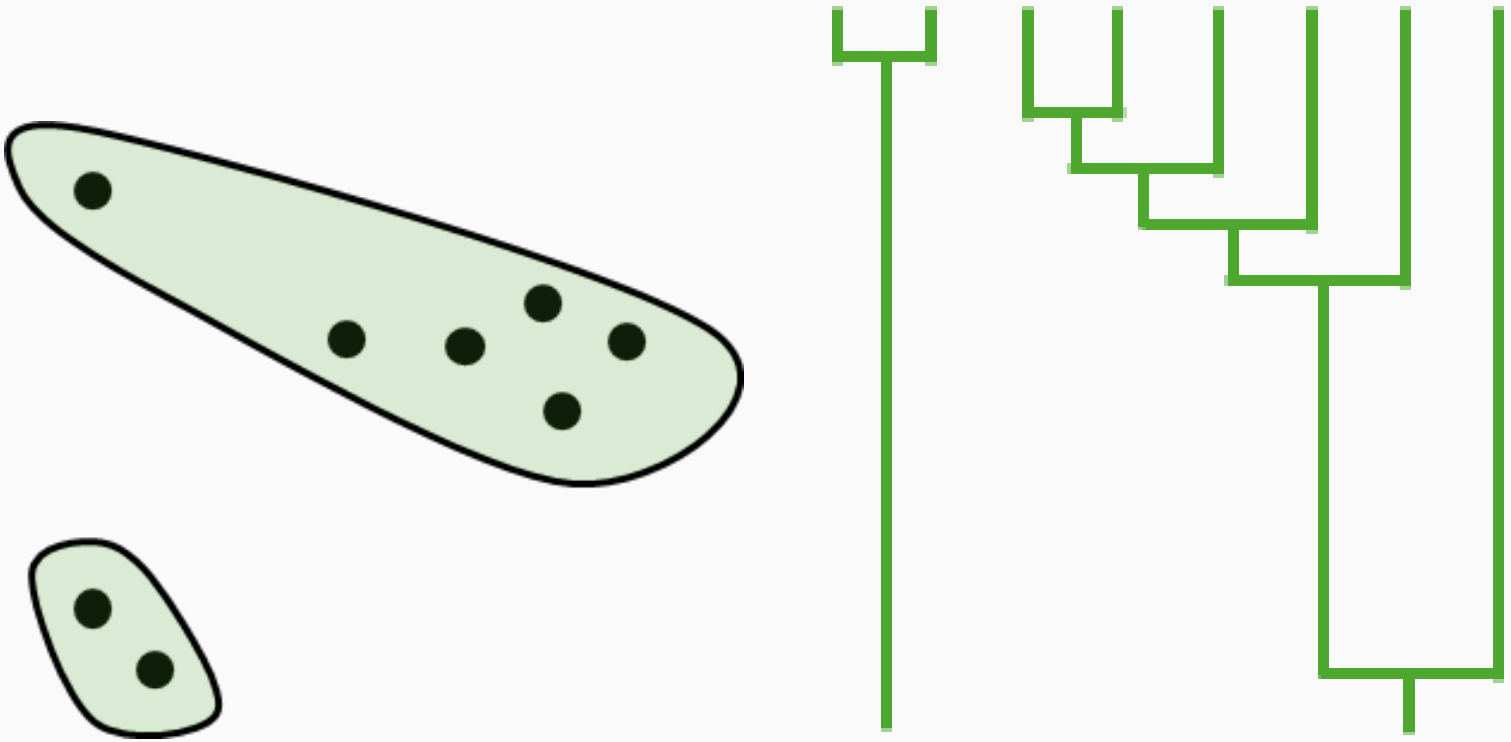
Single Linkage Clustering In Action

$$\ell(A, B) = \min_{a \in A, b \in B} \{d(a, b)\}$$



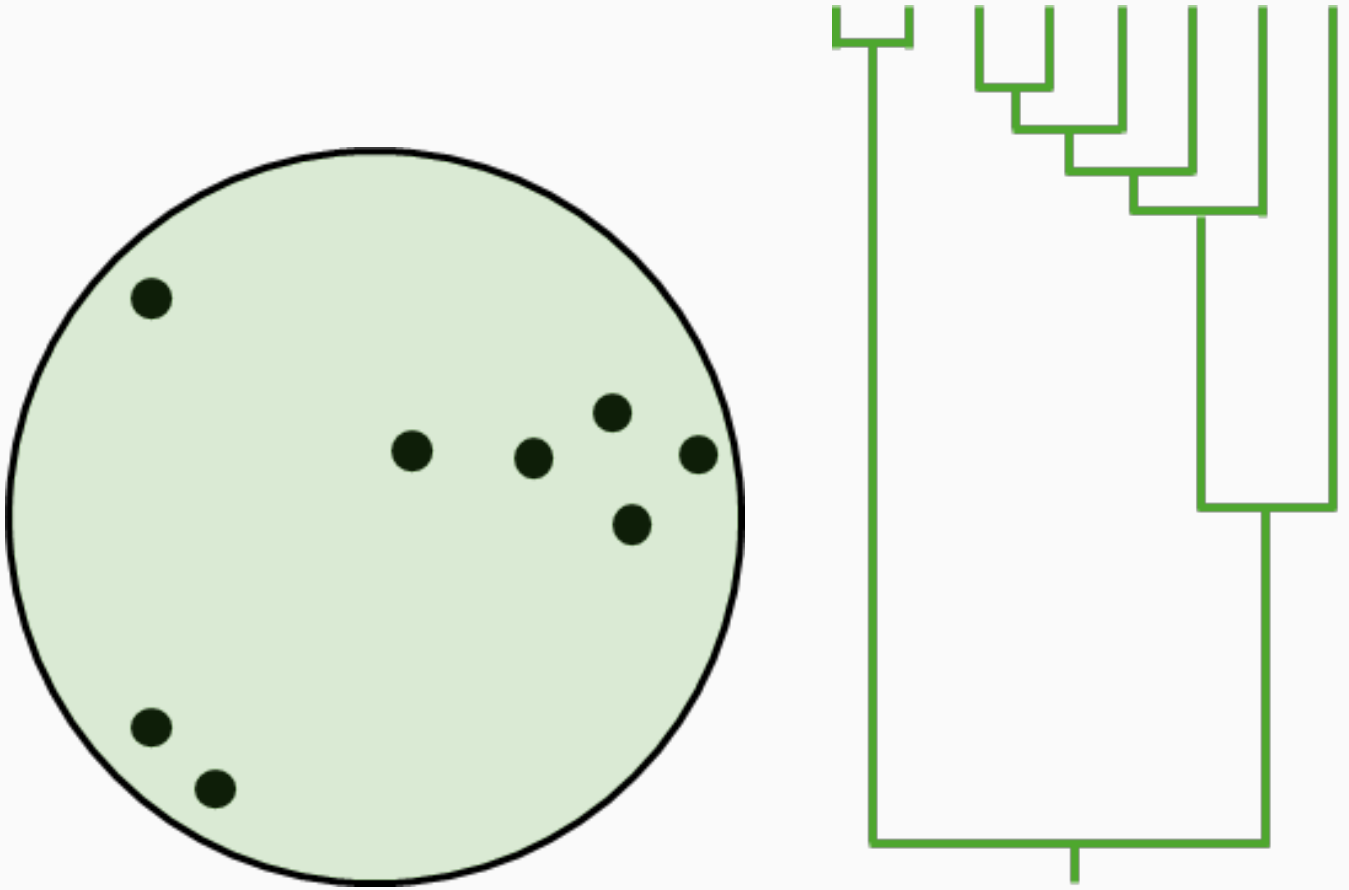
Single Linkage Clustering In Action

$$\ell(A, B) = \min_{a \in A, b \in B} \{d(a, b)\}$$



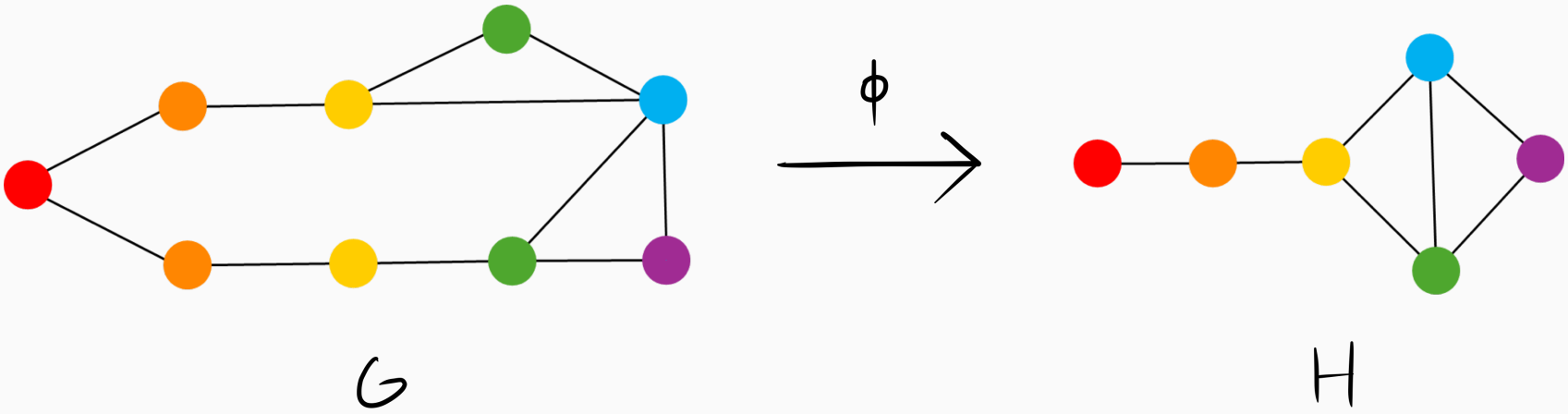
Single Linkage Clustering In Action

$$\ell(A, B) = \min_{a \in A, b \in B} \{d(a, b)\}$$



Refined Ballmapper and Graph Theory

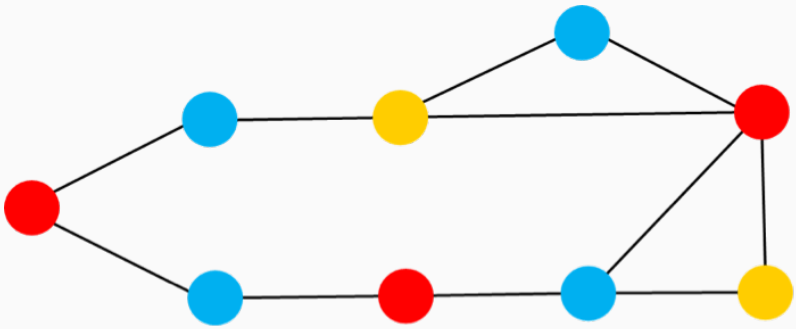
- A function ϕ between the vertices of two graphs G and H is called a **graph homomorphism** if $uv \in E(G)$ implies $\phi(u)\phi(v) \in E(H)$.
- G and H are called **homomorphically equivalent** (hom-equivalent) if there exist graph homomorphisms $f : G \to H$ and $g : H \to G$.



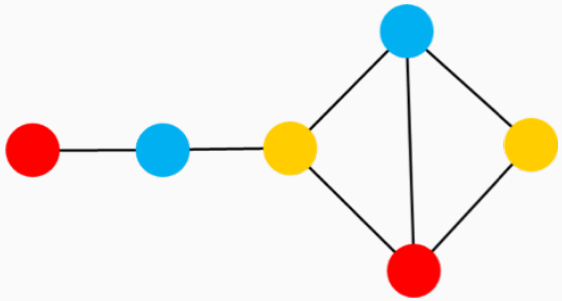
Why Might We Care? Possibility: Cores

- A **core** C of a graph G is a graph such that G and C are hom-equivalent, and C is the smallest such graph.
  - Complete graphs, odd cycles, etc
- Every finite graph has a core, and it is unique (up to isomorphism).
- Graphs with the same cores are necessarily hom-equivalent, and vice versa.
- Core-finding complexity: NP-complete :(
- Applications to relational algebra

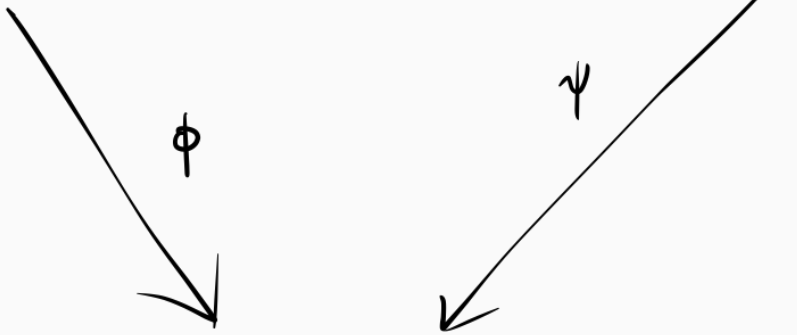
Example Equivalence via Core

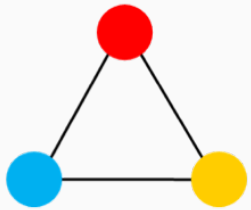


G



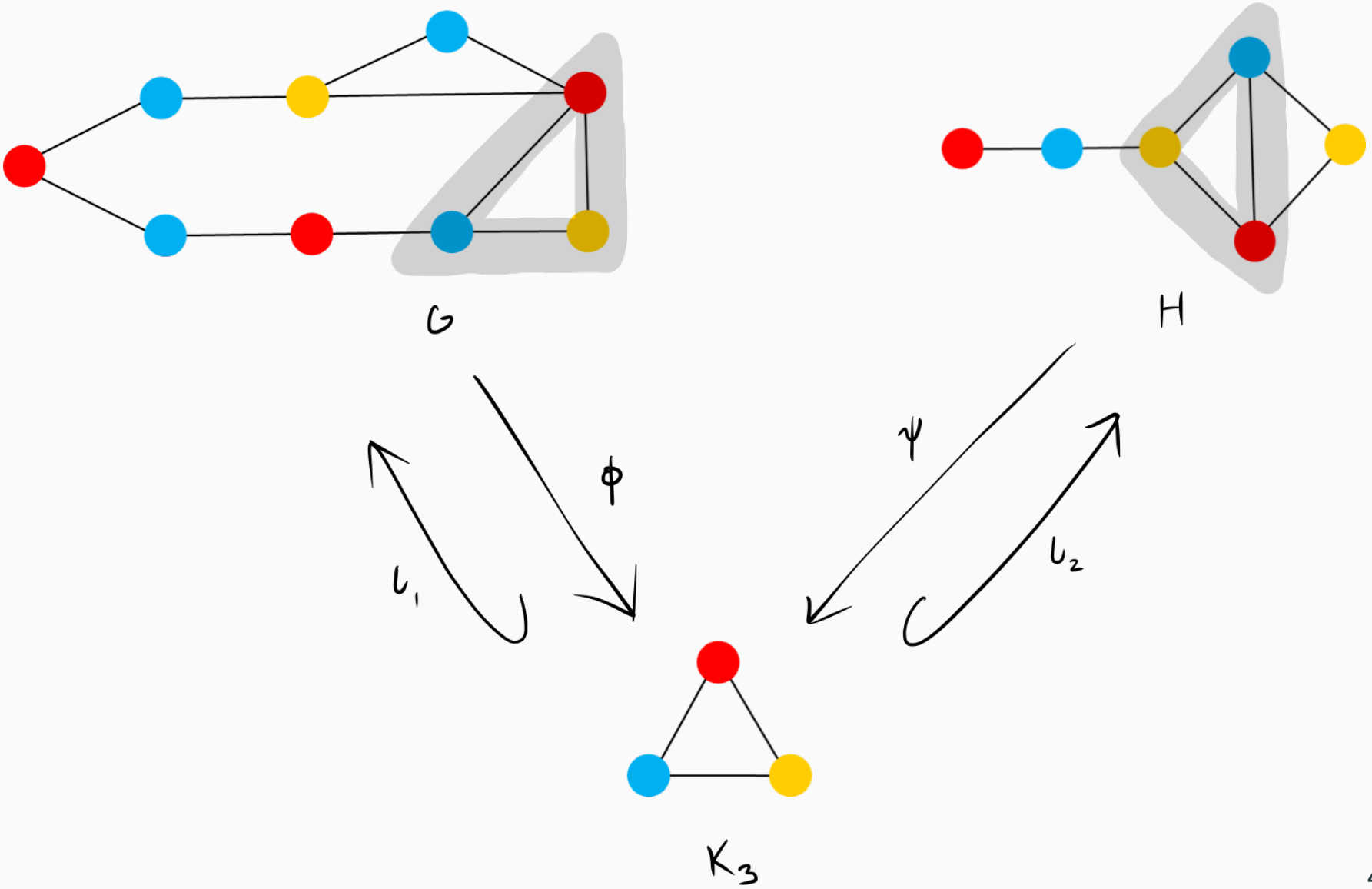
H





K_3

Example Equivalence via Core



Example Equivalence via Core