Classifying Clustering Schemes

Facundo Mémoli,
joint with Gunnar Carlsson

IMA, 2014

What is data clustering?

A (standard/flat) **clustering scheme** assigns to any finite set X a partition P_X of that set.

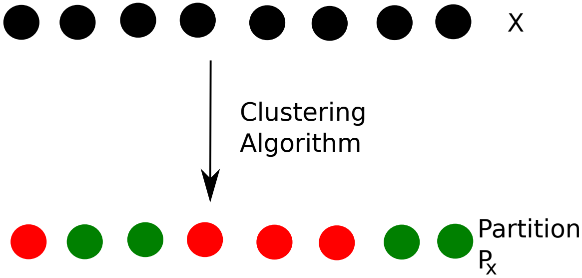


Figure : **Elements with the same color are in the same block**

Clustering: setup

- Let $\mathcal{M} = \{(X, d_X), \text{ finite metric spaces}\}$.

Clustering: setup

- Let $\mathcal{M} = \{(X, d_X)$, finite metric spaces$\}$.
- Given finite set X, $\mathbf{P}(X)$ denotes set of all **partitions** of X.

Clustering: setup

- Let $\mathcal{M} = \{(X, d_X)$, finite metric spaces$\}$.
- Given finite set X, $\mathbf{P}(X)$ denotes set of all **partitions** of X.
- Let $\mathcal{P} = \{(X, P_X), X$ finite and $P_X \in \mathbf{P}(X)\}$. Partitioned spaces.

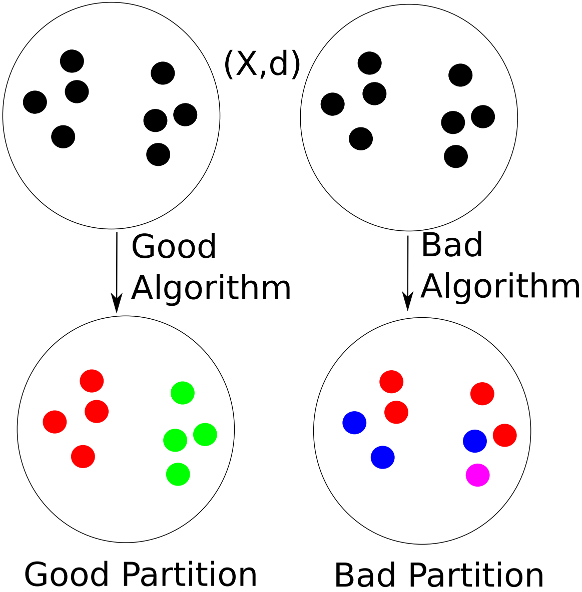
Clustering: setup

- Let $\mathcal{M} = \{(X, d_X)$, finite metric spaces$\}$.
- Given finite set X, $\mathbf{P}(X)$ denotes set of all **partitions** of X.
- Let $\mathcal{P} = \{(X, P_X), X$ finite and $P_X \in \mathbf{P}(X)\}$. Partitioned spaces.

A **standard clustering scheme** is a map:

$$\begin{aligned} \mathfrak{C} : \mathcal{M} &\longrightarrow \mathcal{P} \\ (X, d_X) &\mapsto (X, P_X) \end{aligned}$$

Goal of data clustering



A theorem of J. Kleinberg

Theorem (2002)

J. Kleinberg states: there exists no standard clustering algorithm $\mathfrak{C}$ *that can simultaneously satisfy the following three properties:*

A theorem of J. Kleinberg

Theorem (2002)

J. Kleinberg states: there exists no standard clustering algorithm $\mathfrak{C}$ that can simultaneously satisfy the following three properties:

- *Scale Invariance: For all $(X, d_X) \in \mathcal{M}$,* $\mathfrak{C}(X, d_X) = \mathfrak{C}(X, \lambda d_X), \ \forall \lambda > 0$.

A theorem of J. Kleinberg

Theorem (2002)

J. Kleinberg states: there exists no standard clustering algorithm $\mathfrak{C}$ that can simultaneously satisfy the following three properties:

- *Scale Invariance: For all* $(X, d_X) \in \mathcal{M}$, $\mathfrak{C}(X, d_X) = \mathfrak{C}(X, \lambda d_X)$, $\forall \lambda > 0$.
- *Surjectivity:* $\forall$ X *and* $P_X \in \mathbf{P}(X)$ $\exists d_X$ *s.t.* $\mathfrak{C}(X, d_X) = (X, P_X)$.

A theorem of J. Kleinberg

Theorem (2002)

J. Kleinberg states: there exists no standard clustering algorithm $\mathfrak{C}$ that can simultaneously satisfy the following three properties:

- *Scale Invariance: For all $(X, d_X) \in \mathcal{M}$,* $\mathfrak{C}(X, d_X) = \mathfrak{C}(X, \lambda d_X), \ \forall \lambda > 0.$
- *Surjectivity: $\forall$ X and $P_X \in \mathbf{P}(X)$ $\exists d_X$ s.t.* $\mathfrak{C}(X, d_X) = (X, P_X).$
- *Consistency:*

A theorem of J. Kleinberg

Theorem (2002)

J. Kleinberg states: there exists no standard clustering algorithm $\mathfrak{C}$ *that can simultaneously satisfy the following three properties:*

- *Scale Invariance: For all* $(X, d_X) \in \mathcal{M}$, $\mathfrak{C}(X, d_X) = \mathfrak{C}(X, \lambda d_X)$, $\forall \lambda > 0$.
- *Surjectivity:* $\forall$ X *and* $P_X \in \mathbf{P}(X)$ $\exists d_X$ *s.t.* $\mathfrak{C}(X, d_X) = (X, P_X)$.
- *Consistency: Partition your dataset into blocks.*

A theorem of J. Kleinberg

Theorem (2002)

J. Kleinberg states: there exists no standard clustering algorithm $\mathfrak{C}$ that can simultaneously satisfy the following three properties:

- *Scale Invariance: For all* $(X, d_X) \in \mathcal{M}$, $\mathfrak{C}(X, d_X) = \mathfrak{C}(X, \lambda d_X),\ \forall \lambda > 0.$
- *Surjectivity:* $\forall\ X$ *and* $P_X \in \mathbf{P}(X)\ \exists d_X$ *s.t.* $\mathfrak{C}(X, d_X) = (X, P_X).$
- *Consistency: Partition your dataset into blocks.*
    - *Shrink intrablock distances*

A theorem of J. Kleinberg

Theorem (2002)

J. Kleinberg states: there exists no standard clustering algorithm $\mathfrak{C}$ that can simultaneously satisfy the following three properties:

- *Scale Invariance: For all $(X, d_X) \in \mathcal{M}$, $\mathfrak{C}(X, d_X) = \mathfrak{C}(X, \lambda d_X),\ \forall \lambda > 0.$*
- *Surjectivity: $\forall\ X$ and $P_X \in \mathbf{P}(X)\ \exists d_X$ s.t. $\mathfrak{C}(X, d_X) = (X, P_X)$.*
- *Consistency: Partition your dataset into blocks.*
    - *Shrink intrablock distances*
    - *Expand interblock distances*

A theorem of J. Kleinberg

Theorem (2002)

J. Kleinberg states: there exists no standard clustering algorithm $\mathfrak{C}$ that can simultaneously satisfy the following three properties:

- *Scale Invariance: For all $(X, d_X) \in \mathcal{M}$,* $\mathfrak{C}(X, d_X) = \mathfrak{C}(X, \lambda d_X),\ \forall \lambda > 0$.
- *Surjectivity:* $\forall\ X$ *and* $P_X \in \mathbf{P}(X)\ \exists d_X$ *s.t.* $\mathfrak{C}(X, d_X) = (X, P_X)$.
- *Consistency: Partition your dataset into blocks.*
    - *Shrink intrablock distances*
    - *Expand interblock distances*
    - *invariance under this transformation*

A theorem of J. Kleinberg

Theorem (2002)

J. Kleinberg states: there exists no standard clustering algorithm $\mathfrak{C}$ that can simultaneously satisfy the following three properties:

- *Scale Invariance: For all* $(X, d_X) \in \mathcal{M}$, $\mathfrak{C}(X, d_X) = \mathfrak{C}(X, \lambda d_X)$, $\forall \lambda > 0$.
- *Surjectivity:* $\forall$ X *and* $P_X \in \mathbf{P}(X)$ $\exists d_X$ *s.t.* $\mathfrak{C}(X, d_X) = (X, P_X)$.
- *Consistency: Partition your dataset into blocks.*
    - *Shrink intrablock distances*
    - *Expand interblock distances*
    - *invariance under this transformation*

Kleinberg's theorem was the inspiration for this work.

This talk

↳ study reformulation/variation of Kleinberg's point of view

This talk

$\hookrightarrow$ study reformulation/variation of Kleinberg's point of view

Roadmap for studying data clustering:

This talk

↳ study reformulation/variation of Kleinberg's point of view

Roadmap for studying data clustering:

1. Recast input/output spaces as categories

This talk

↳ study reformulation/variation of Kleinberg's point of view

Roadmap for studying data clustering:

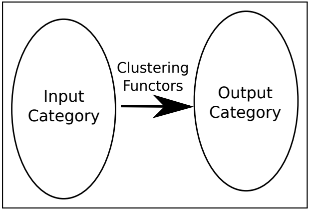
1. Recast input/output spaces as categories
2. Categories impose a structure on data

This talk

↳study reformulation/variation of Kleinberg's point of view

Roadmap for studying data clustering:

1. Recast input/output spaces as categories
2. Categories impose a structure on data
3. Study "functors" between categories (respecting this structure)

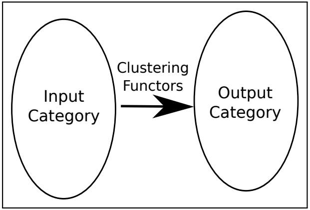


This talk

↳ study reformulation/variation of Kleinberg's point of view

Roadmap for studying data clustering:

1. Recast input/output spaces as categories
2. Categories impose a structure on data
3. Study "functors" between categories (respecting this structure)



We provide a range of input categories (small to large) and show analogues of Kleinberg's theorem in each.

Roadmap for classification

- Motivation
- **Categories**
- Framework
  - Input Categories
  - Output Category
- Functors
- Results
  - $\mathcal{M}^{iso}$ (small)
  - $\mathcal{M}^{gen}$ (large)
  - $\mathcal{M}^{inj}$ (medium)
- Conclusion

Category theory

Intuitively: category $\simeq$ directed multigraph.

Category theory

Intuitively: category $\simeq$ directed multigraph.

Definition (A category $\mathcal{C}$ consists of:)

Category theory

Intuitively: category $\simeq$ directed multigraph.

Definition (A category $\mathcal{C}$ consists of:)

- A collection of **objects** $\boxed{\mathrm{ob}(\mathcal{C})}$ *(think "vertices")*

Category theory

Intuitively: category $\simeq$ directed multigraph.

Definition (A category $\mathcal{C}$ consists of:)

- A collection of **objects** $\boxed{\mathrm{ob}(\mathcal{C})}$ *(think "vertices")*
- For each pair of objects X, Y a set $\boxed{\mathrm{Mor}_{\mathcal{C}}(X, Y)}$, the **morphisms** from X to Y *(think directed edges/arrows)*

Intuitively: category $\simeq$ directed multigraph.

Definition (A category $\mathcal{C}$ consists of:)

- A collection of **objects** $\boxed{\mathrm{ob}(\mathcal{C})}$ *(think "vertices")*
- For each pair of objects X, Y a set $\boxed{\mathrm{Mor}_{\mathcal{C}}(X, Y)}$, the **morphisms** from X to Y *(think directed edges/arrows)*
- **Composition operations**:
  $\circ : \mathrm{Mor}_{\mathcal{C}}(X, Y) \times \mathrm{Mor}_{\mathcal{C}}(Y, Z) \to \mathrm{Mor}_{\mathcal{C}}(X, Z)$
  such that $\circ$ is *associative (think function composition)*

Intuitively: category $\simeq$ directed multigraph.

Definition (A category $\mathcal{C}$ consists of:)

- A collection of **objects** $\boxed{\mathrm{ob}(\mathcal{C})}$ *(think "vertices")*
- For each pair of objects X, Y a set $\boxed{\mathrm{Mor}_{\mathcal{C}}(X, Y)}$, the **morphisms** from X to Y *(think directed edges/arrows)*
- **Composition operations**:
  $\circ : \mathrm{Mor}_{\mathcal{C}}(X, Y) \times \mathrm{Mor}_{\mathcal{C}}(Y, Z) \to \mathrm{Mor}_{\mathcal{C}}(X, Z)$
  such that $\circ$ is *associative (think function composition)*
- **Identity morphisms**: special morphisms $X \to X$, $\mathrm{id}_X \in \mathrm{Mor}_{\mathcal{C}}(X, X)$ for each X *(think self loops/ identity maps)* such that if $f \in \mathrm{Mor}_{\mathcal{C}}(X, Y)$ then $\mathrm{id}_Y \circ f = f \circ \mathrm{id}_X = f$

Examples of categories

Example (category $\underline{3}$)

The category $\underline{3}$ has exactly three objects A, B and C and six morphisms: the identities for A, B,C, and three more morphisms, $\mathrm{Mor}_{\underline{3}}(A, B) = f$, $\mathrm{Mor}_{\underline{3}}(B, C) = g$ and $\mathrm{Mor}_{\underline{3}}(A, C) = h$:

$$\begin{tikzcd}
A \arrow[loop left, "id_A"] \arrow[rr, "f"] \arrow[dr, "h"'] & & B \arrow[loop right, "id_B"] \arrow[dl, "g"] \\
& C \arrow[loop below, "id_C"] &
\end{tikzcd}$$

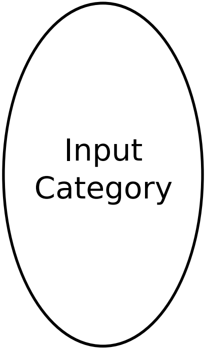
In order to satisfy composition: $h = g \circ f$.

Roadmap for classification

- Motivation
- Categories
- Framework
  - **Input Categories**
  - Output Categories
- Functors
- Results
  - $\mathcal{M}^{iso}$ (small)
  - $\mathcal{M}^{gen}$ (large)
  - $\mathcal{M}^{inj}$ (medium)
- Conclusion

Roadmap for classification

- Motivation
- Categories
- Framework
  - **Input Categories**
  - Output Categories
- Functors
- Results
  - $\mathcal{M}^{iso}$ (small)
  - $\mathcal{M}^{gen}$ (large)
  - $\mathcal{M}^{inj}$ (medium)
- Conclusion



Input categories for our model

Consider $(X, d_X), (Y, d_Y) \in \mathcal{M}$, where $\mathcal{M}$ is the collection of all finite metric spaces.

Input categories for our model

Consider $(X, d_X), (Y, d_Y) \in \mathcal{M}$, where $\mathcal{M}$ is the collection of all finite metric spaces.

Define $N(X, Y)$ the set of **distance non-increasing maps**:

$$N(X, Y) = \{f : X \to Y \mid \forall x, x' \in X, d_X(x, x') \geq d_Y(f(x), f(x'))\}$$

Input categories for our model

Consider $(X, d_X), (Y, d_Y) \in \mathcal{M}$, where $\mathcal{M}$ is the collection of all finite metric spaces.

Define $N(X, Y)$ the set of **distance non-increasing maps**:

$$N(X, Y) = \{f : X \to Y \mid \forall x, x' \in X, d_X(x, x') \geq d_Y(f(x), f(x'))\}$$

Note: composition works, $\circ : N(X, Y) \times N(Y, Z) \to N(X, Z)$, and $\mathrm{id}_X \in N(X, X)$.

Input categories for our model

Consider $(X, d_X), (Y, d_Y) \in \mathcal{M}$, where $\mathcal{M}$ is the collection of all finite metric spaces.

Define $N(X, Y)$ the set of **distance non-increasing maps**:

$$N(X, Y) = \{f : X \to Y \mid \forall x, x' \in X, d_X(x, x') \geq d_Y(f(x), f(x'))\}$$

Note: composition works, $\circ : N(X, Y) \times N(Y, Z) \to N(X, Z)$, and $\mathrm{id}_X \in N(X, X)$.
We now define three categories:

Input categories for our model

Consider $(X, d_X), (Y, d_Y) \in \mathcal{M}$, where $\mathcal{M}$ is the collection of all finite metric spaces.

Define $N(X, Y)$ the set of **distance non-increasing maps**:

$$N(X, Y) = \{f : X \to Y \mid \forall x, x' \in X, d_X(x, x') \geq d_Y(f(x), f(x'))\}$$

Note: composition works, $\circ : N(X, Y) \times N(Y, Z) \to N(X, Z)$, and $\mathrm{id}_X \in N(X, X)$.
We now define three categories:

- $\mathcal{M}^{gen}$: $\mathrm{ob}(\mathcal{M}^{gen}) = \mathcal{M}$ and $\mathrm{Mor}_{\mathcal{M}^{gen}}(X, Y) = N(X, Y)$.

Input categories for our model

Consider $(X, d_X), (Y, d_Y) \in \mathcal{M}$, where $\mathcal{M}$ is the collection of all finite metric spaces.

Define $N(X, Y)$ the set of **distance non-increasing maps**:

$$N(X, Y) = \{f : X \to Y \mid \forall x, x' \in X, d_X(x, x') \geq d_Y(f(x), f(x'))\}$$

Note: composition works, $\circ : N(X, Y) \times N(Y, Z) \to N(X, Z)$, and $\mathrm{id}_X \in N(X, X)$.
We now define three categories:

- $\mathcal{M}^{gen}$: $\mathrm{ob}(\mathcal{M}^{gen}) = \mathcal{M}$ and $\mathrm{Mor}_{\mathcal{M}^{gen}}(X, Y) = N(X, Y)$.
- $\mathcal{M}^{inj}$: the subcategory whose morphisms are also **injective**.

Input categories for our model

Consider $(X, d_X), (Y, d_Y) \in \mathcal{M}$, where $\mathcal{M}$ is the collection of all finite metric spaces.

Define $N(X, Y)$ the set of **distance non-increasing maps**:

$$N(X, Y) = \{f : X \to Y \mid \forall x, x' \in X, d_X(x, x') \geq d_Y(f(x), f(x'))\}$$

Note: composition works, $\circ : N(X, Y) \times N(Y, Z) \to N(X, Z)$, and $\mathrm{id}_X \in N(X, X)$.
We now define three categories:

- $\mathcal{M}^{gen}$: $\mathrm{ob}(\mathcal{M}^{gen}) = \mathcal{M}$ and $\mathrm{Mor}_{\mathcal{M}^{gen}}(X, Y) = N(X, Y)$.
- $\mathcal{M}^{inj}$: the subcategory whose morphisms are also **injective**.
- $\mathcal{M}^{iso}$: the subcategory whose morphisms are **isometries**:
  $f : X \to Y$ is an *isometry* if f is bijective and
  $d_Y(f(x), f(x')) = d_X(x, x')$ for all x and x'.

Input categories for our model

Consider $(X, d_X), (Y, d_Y) \in \mathcal{M}$, where $\mathcal{M}$ is the collection of all finite metric spaces.

Define $N(X, Y)$ the set of **distance non-increasing maps**:

$$N(X,Y) = \{f : X \to Y \mid \forall x, x' \in X, d_X(x,x') \geq d_Y(f(x), f(x'))\}$$

Note: composition works, $\circ : N(X,Y) \times N(Y,Z) \to N(X,Z)$, and $\mathrm{id}_X \in N(X,X)$.
We now define three categories:

- $\mathcal{M}^{gen}$: $\mathrm{ob}(\mathcal{M}^{gen}) = \mathcal{M}$ and $\mathrm{Mor}_{\mathcal{M}^{gen}}(X,Y) = N(X,Y)$.
- $\mathcal{M}^{inj}$: the subcategory whose morphisms are also **injective**.
- $\mathcal{M}^{iso}$: the subcategory whose morphisms are **isometries**:

  $f : X \to Y$ is an *isometry* if f is bijective and $d_Y(f(x), f(x')) = d_X(x, x')$ for all x and x'.

$$\mathcal{M}^{gen} \supseteq \mathcal{M}^{inj} \supseteq \mathcal{M}^{iso}$$

Input categories for our model

Consider $(X, d_X), (Y, d_Y) \in \mathcal{M}$, where $\mathcal{M}$ is the collection of all finite metric spaces.

Define $N(X, Y)$ the set of **distance non-increasing maps**:

$$N(X,Y) = \{f : X \to Y \mid \forall x, x' \in X, d_X(x,x') \geq d_Y(f(x), f(x'))\}$$

Note: composition works, $\circ : N(X,Y) \times N(Y,Z) \to N(X,Z)$, and $\mathrm{id}_X \in N(X,X)$.
We now define three categories:

- $\mathcal{M}^{gen}$: $\mathrm{ob}(\mathcal{M}^{gen}) = \mathcal{M}$ and $\mathrm{Mor}_{\mathcal{M}^{gen}}(X,Y) = N(X,Y)$.
- $\mathcal{M}^{inj}$: the subcategory whose morphisms are also **injective**.
- $\mathcal{M}^{iso}$: the subcategory whose morphisms are **isometries**:
  $f : X \to Y$ is an *isometry* if f is bijective and $d_Y(f(x), f(x')) = d_X(x, x')$ for all x and x'.

$$\mathcal{M}^{gen} \supseteq \mathcal{M}^{inj} \supseteq \mathcal{M}^{iso}$$

$\mathcal{M}$ will denote any of these categories

Nested categories: same object set, but filter morphisms

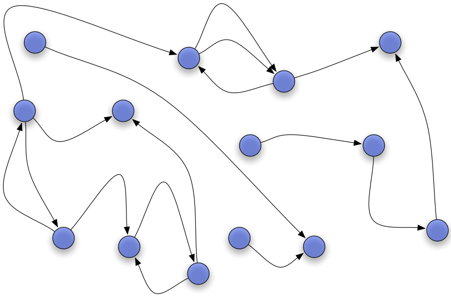


Figure : $\mathcal{M}^{gen} \supseteq \mathcal{M}^{inj} \supseteq \mathcal{M}^{iso}$. Same object set, nested morphism sets!

Nested categories: same object set, but filter morphisms

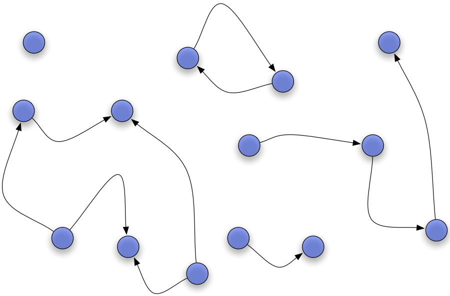


Figure : $\mathcal{M}^{gen} \supseteq \mathcal{M}^{inj} \supseteq \mathcal{M}^{iso}$. Same object set, nested morphism sets!

Nested categories: same object set, but filter morphisms

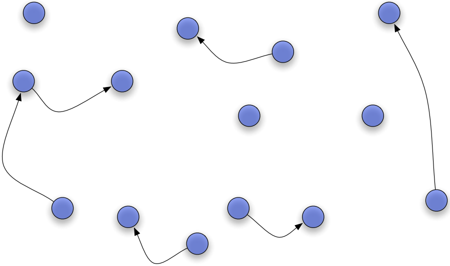


Figure : $\mathcal{M}^{gen} \supseteq \mathcal{M}^{inj} \supseteq \mathcal{M}^{iso}$. Same object set, nested morphism sets!

$\mathrm{Mor}_{\mathcal{M}^{gen}}(X, Y)$ is never empty

$\mathcal{M}^{gen}$ is special in that $\forall X, Y \ \mathrm{Mor}_{\mathcal{M}^{gen}}(X, Y) \neq \emptyset$

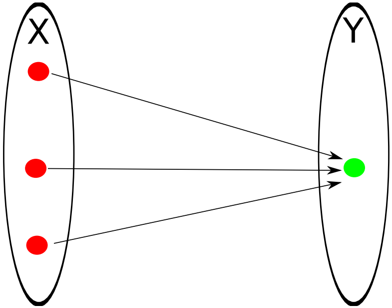


Figure : Constant maps are always distance non-increasing. Hence, $\mathcal{M}^{gen}$ is fully connected! Thus, can always have diagram $X \to Y \to X$ in $\mathcal{M}^{gen}$.

$\mathrm{Mor}_{\mathcal{M}^{gen}}(X, Y)$ is never empty

$\mathcal{M}^{gen}$ is special in that $\forall X, Y\ \mathrm{Mor}_{\mathcal{M}^{gen}}(X, Y) \neq \emptyset$

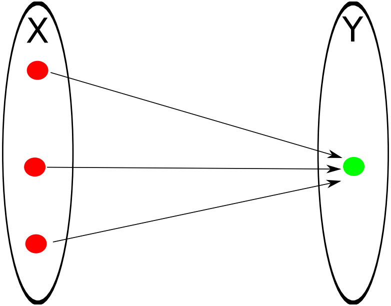
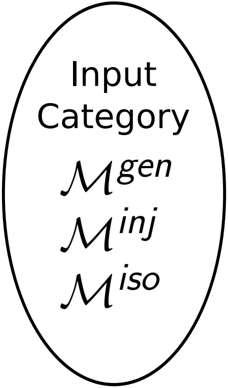


Figure : Constant maps are always distance non-increasing. Hence, $\mathcal{M}^{gen}$ is fully connected! Thus, can always have diagram $X \to Y \to X$ in $\mathcal{M}^{gen}$.

This property fails in $\mathcal{M}^{inj}$ and $\mathcal{M}^{iso}$, since morphism ϕ could not send two different elements in X, to one element in Y.

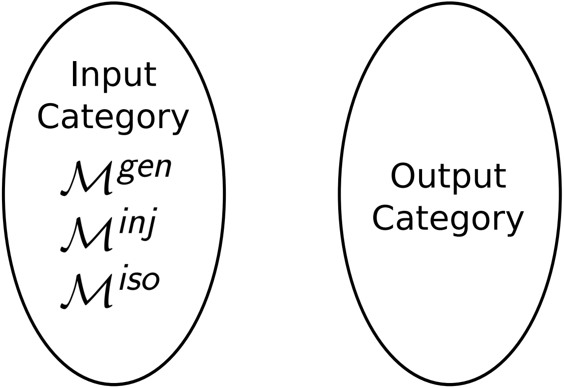
Roadmap for classification

- Motivation
- Categories
- Framework
  - Input Categories
  - **Output Category**
- Functors
- Results
  - $\mathcal{M}^{iso}$ (small)
  - $\mathcal{M}^{gen}$ (large)
  - $\mathcal{M}^{inj}$ (medium)
- Conclusion



Roadmap for classification

- Motivation
- Categories
- Framework
  - Input Categories
  - **Output Category**
- Functors
- Results
  - $\mathcal{M}^{iso}$ (small)
  - $\mathcal{M}^{gen}$ (large)
  - $\mathcal{M}^{inj}$ (medium)
- Conclusion



Output category

For a finite set X, $\mathbf{P}(X)$ denotes set of all partitions of X.

Output category

For a finite set X, $\mathbf{P}(X)$ denotes set of all partitions of X.

Definition ($\mathcal{P}$, a category of outputs of standard clustering schemes)

The category $\mathcal{P}$ has

Output category

For a finite set X, $\mathbf{P}(X)$ denotes set of all partitions of X.

Definition ($\mathcal{P}$, a category of outputs of standard clustering schemes)

The category $\mathcal{P}$ has

- $\mathrm{ob}(\mathcal{P})$ equal to all possible pairs (X, P_X) where X is a finite set and $P_X \in \mathbf{P}(X)$.

Output category

For a finite set X, $\mathbf{P}(X)$ denotes set of all partitions of X.

Definition ($\mathcal{P}$, a category of outputs of standard clustering schemes)

The category $\mathcal{P}$ has

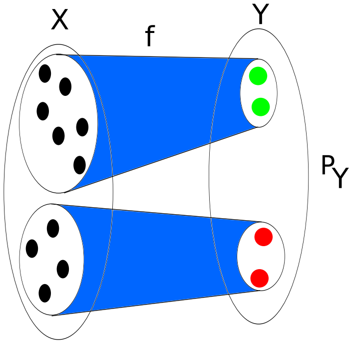
- $\mathrm{ob}(\mathcal{P})$ equal to all possible pairs (X, P_X) where X is a finite set and $P_X \in \mathbf{P}(X)$.
- $\mathrm{Mor}_{\mathcal{P}}\big((X, P_X), (Y, P_Y)\big)$ is the set of all maps $f : X \to Y$ where P_X is a refinement of $f^*(P_Y)$.

What is $f^*(P_Y)$?

What is $f^*(P_Y)$?

Definition ($f^*(P_Y)$: pullback partition)

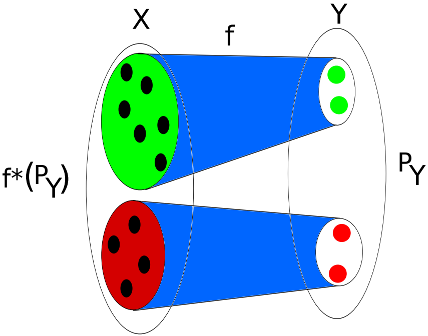
Let Y be a finite set, $P_Y \in \mathbf{P}(Y)$, and $f : X \to Y$ be a set map. We define $f^*(P_Y) = \{f^{-1}(B) : B \in P_Y\} \in \mathbf{P}(X)$.



What is $f^*(P_Y)$?

Definition ($f^*(P_Y)$: pullback partition)

Let Y be a finite set, $P_Y \in \mathbf{P}(Y)$, and $f : X \to Y$ be a set map.
We define $f^*(P_Y) = \{f^{-1}(B) : B \in P_Y\} \in \mathbf{P}(X)$.



Morphisms in $\mathcal{P}$

$\mathrm{Mor}_{\mathcal{P}}\big((X, P_X), (Y, P_Y)\big)$ is the set of all maps $f : X \to Y$ where P_X is a refinement of $f^*(P_Y)$

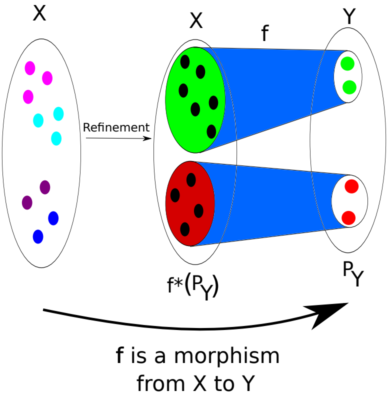


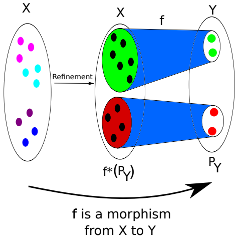
Figure : P_X refines $f^*(P_Y)$

For a finite set X we denote by $\mathbf{P}(X)$ the set of all partitions of X.

Definition ($\mathcal{P}$, a category of outputs of standard clustering schemes)

The category $\mathcal{P}$ has

- $\mathrm{ob}(\mathcal{P})$ equal to all possible pairs (X, P_X) where X is a finite set and $P_X \in \mathbf{P}(X)$.
- $\mathrm{Mor}_{\mathcal{P}}\big((X, P_X), (Y, P_Y)\big)$ is the set of all maps $f : X \to Y$ where P_X is a refinement of $f^*(P_Y)$.



Example (forcing)

- Assume that $f \in \mathrm{Mor}_{\mathcal{P}}\big((\{a,b\},\{\{a,b\}\}),(Y,P_Y)\big)$. Then,

  $f(a)$ and $f(b)$ must be in <u>same</u> block of P_Y.

Example (forcing)

- Assume that $f \in \mathrm{Mor}_{\mathcal{P}}\big((\{a,b\},\{\{a,b\}\}),(Y,P_Y)\big)$. Then,

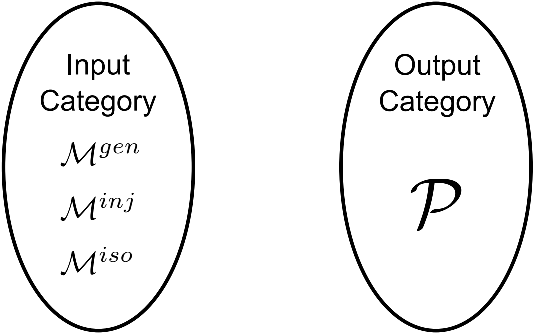
  $f(a)$ and $f(b)$ must be in <u>same</u> block of P_Y.

- Let $f \in \mathrm{Mor}_{\mathcal{P}}\big((X,P_X),(\{a,b\},\{\{a\},\{b\}\})\big)$. Then,

  $f(x) \neq f(x')$, implies that x, x' must be in <u>different</u> blocks of P_X.

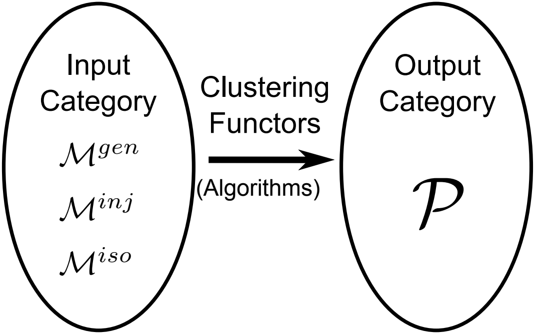
Roadmap for classification

- Motivation
- Categories
- Framework
  - Input Categories
  - Output Categories
- **Functors**
- Results
  - $\mathcal{M}^{iso}$ (small)
  - $\mathcal{M}^{gen}$ (large)
  - $\mathcal{M}^{inj}$ (medium)
- Conclusion



Roadmap for classification

- Motivation
- Categories
- Framework
  - Input Categories
  - Output Categories
- **Functors**
- Results
  - $\mathcal{M}^{iso}$ (small)
  - $\mathcal{M}^{gen}$ (large)
  - $\mathcal{M}^{inj}$ (medium)
- Conclusion



Functors

Definition (Functor)

Let $\mathcal{C}$ and $\mathcal{D}$ be categories. A **functor** from $\mathcal{C}$ to $\mathcal{D}$ consists of:

Functors

Definition (Functor)

Let $\mathcal{C}$ and $\mathcal{D}$ be categories. A **functor** from $\mathcal{C}$ to $\mathcal{D}$ consists of:

- **A mapping of the objects:** $\Phi : \mathrm{ob}(\mathcal{C}) \to \mathrm{ob}(\mathcal{D})$.

Functors

Definition (Functor)

Let $\mathcal{C}$ and $\mathcal{D}$ be categories. A **functor** from $\mathcal{C}$ to $\mathcal{D}$ consists of:

- **A mapping of the objects:** $\Phi : \mathrm{ob}(\mathcal{C}) \to \mathrm{ob}(\mathcal{D})$.
- **A mapping of the morphisms:** $\forall X, Y \in \mathrm{ob}(\mathcal{C})$ $\Phi : \mathrm{Mor}_{\mathcal{C}}(X, Y) \to \mathrm{Mor}_{\mathcal{D}}(\Phi(X), \Phi(Y))$ so that

Functors

Definition (Functor)

Let $\mathcal{C}$ and $\mathcal{D}$ be categories. A **functor** from $\mathcal{C}$ to $\mathcal{D}$ consists of:

- **A mapping of the objects:** $\Phi : \mathrm{ob}(\mathcal{C}) \to \mathrm{ob}(\mathcal{D})$.
- **A mapping of the morphisms:** $\forall X, Y \in \mathrm{ob}(\mathcal{C})$ $\Phi : \mathrm{Mor}_{\mathcal{C}}(X, Y) \to \mathrm{Mor}_{\mathcal{D}}(\Phi(X), \Phi(Y))$ so that
  1. **identities are respected:** $\Phi(\mathrm{id}_X) = \mathrm{id}_{\Phi(X)}$

Functors

Definition (**Functor**)

Let $\mathcal{C}$ and $\mathcal{D}$ be categories. A **functor** from $\mathcal{C}$ to $\mathcal{D}$ consists of:

- **A mapping of the objects:** $\Phi : \mathrm{ob}(\mathcal{C}) \to \mathrm{ob}(\mathcal{D})$.
- **A mapping of the morphisms:** $\forall X, Y \in \mathrm{ob}(\mathcal{C})$
  $\Phi : \mathrm{Mor}_{\mathcal{C}}(X, Y) \to \mathrm{Mor}_{\mathcal{D}}(\Phi(X), \Phi(Y))$ so that
  1. **identities are respected:** $\Phi(\mathrm{id}_X) = \mathrm{id}_{\Phi(X)}$
  2. **composition is respected:** $\Phi(g \circ f) = \Phi(g) \circ \Phi(f)$

- For $\mathcal{M}$ being each of our three input categories, we are going to require that for all morphisms (test functions) $f \in \mathrm{Mor}_{\mathcal{M}}(X, Y)$:

$$\begin{tikzcd}
(X, d_X) \arrow[r, "f"] \arrow[d, "\mathfrak{C}"] & (Y, d_Y) \arrow[d, "\mathfrak{C}"] \\
(X, P_X) \arrow[r, "f"] & (Y, P_Y)
\end{tikzcd}$$

Clustering methods as functors

- For $\mathcal{M}$ being each of our three input categories, we are going to require that for all morphisms (test functions) $f \in \mathrm{Mor}_{\mathcal{M}}(X, Y)$:

$$\begin{tikzcd}
(X, d_X) \arrow[r, "f"] \arrow[d, "\mathfrak{C}"] & (Y, d_Y) \arrow[d, "\mathfrak{C}"] \\
(X, P_X) \arrow[r, "f"] & (Y, P_Y)
\end{tikzcd}$$

- Take for example (X, d) and (X, d'), two metrics on X s.t. $d \geq d'$. If P and P' are the respective partitions produced by a clustering functor $\mathfrak{C}$, then if $x \sim_P x'$, it also has to happen that $x \sim_{P'} x'$ as well. Namely, reducing distances has to make it easier to cluster two points!

Clustering methods as functors

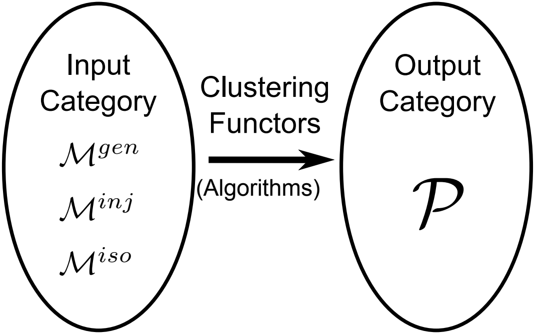
- For $\mathcal{M}$ being each of our three input categories, we are going to require that for all morphisms (test functions) $f \in \mathrm{Mor}_{\mathcal{M}}(X, Y)$:

$$\begin{tikzcd}
(X, d_X) \arrow[r, "f"] \arrow[d, "\mathfrak{C}"] & (Y, d_Y) \arrow[d, "\mathfrak{C}"] \\
(X, P_X) \arrow[r, "f"] & (Y, P_Y)
\end{tikzcd}$$

- Take for example (X, d) and (X, d'), two metrics on X s.t. $d \geq d'$. If P and P' are the respective partitions produced by a clustering functor $\mathfrak{C}$, then if $x \sim_P x'$, it also has to happen that $x \sim_{P'} x'$ as well. Namely, reducing distances has to make it easier to cluster two points!
- Notice that it is more difficult to find functors $\mathcal{M}^{gen} \to \mathcal{P}$ than to find functors $\mathcal{M}^{inj} \to \mathcal{P}$, than to find functors $\mathcal{M}^{iso} \to \mathcal{P}$.

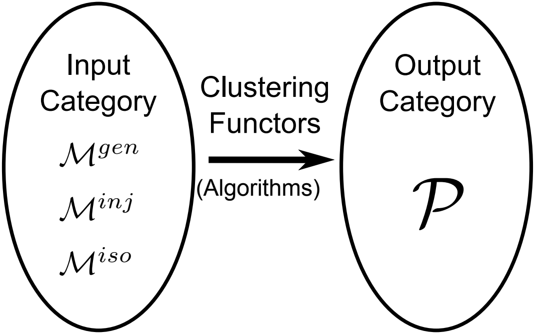
Roadmap for classification

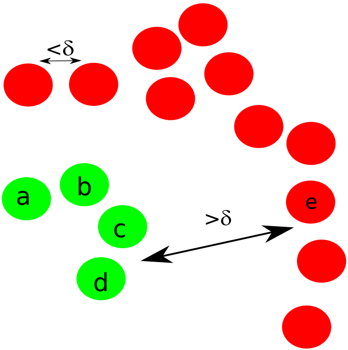
- Motivation
- Categories
- Framework
  - Input Categories
  - Output Categories
- Functors
- Results
  - $\mathcal{M}^{iso}$ **(small)**
  - $\mathcal{M}^{gen}$ (large)
  - $\mathcal{M}^{inj}$ (medium)
- Conclusion



Roadmap for classification

- Motivation
- Categories
- Framework
  - Input Categories
  - Output Categories
- Functors
- Results
  - $\mathcal{M}^{iso}$ (small)
  - $\mathcal{M}^{gen}$ **(large)**
  - $\mathcal{M}^{inj}$ (medium)
- Conclusion





Definition

On $(X, d_X) \in \mathcal{M}$, for each $\delta \geq 0$, define the **equivalence relation** $\sim_\delta$, where $x \sim_\delta x' \iff$ there is a sequence $x_0, x_1, \ldots, x_k \in X$ so that $x_0 = x, x_k = x'$, and $d_X(x_i, x_{i+1}) \leq \delta$ for all i. Let $\mathbf{P_X}(\delta)$ be the resulting partition.

$\mathcal{M}^{gen}$ functorial clustering algorithms

Definition (Vietoris-Rips clustering functor on $\mathcal{M}^{gen}$)

For each $\delta \geq 0$ define

$$\mathfrak{R}_\delta : \mathcal{M}^{gen} \to \mathcal{P}$$

by

$$\mathfrak{R}_\delta(X, d_X) = (X, P_X(\delta)).$$

A.K.A. the **single linkage clustering** on parameter δ.

$\mathcal{M}^{gen}$ functorial clustering algorithms

Definition (Vietoris-Rips clustering functor on $\mathcal{M}^{gen}$)

For each $\delta \geq 0$ define

$$\mathfrak{R}_\delta : \mathcal{M}^{gen} \to \mathcal{P}$$

by

$$\mathfrak{R}_\delta(X, d_X) = (X, P_X(\delta)).$$

A.K.A. the **single linkage clustering** on parameter δ.

One checks this *is* a functor (**existence**), and prove a **uniqueness** theorem for it.

$\mathcal{M}^{gen}$ functorial clustering algorithms

Definition (Vietoris-Rips clustering functor on $\mathcal{M}^{gen}$)

For each $\delta \geq 0$ define

$$\mathfrak{R}_\delta : \mathcal{M}^{gen} \to \mathcal{P}$$

by

$$\mathfrak{R}_\delta(X, d_X) = (X, P_X(\delta)).$$

A.K.A. the **single linkage clustering** on parameter δ.

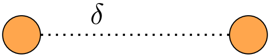
One checks this *is* a functor (**existence**), and prove a **uniqueness** theorem for it.

Functoriality is equivalent to:

$\phi \in \mathrm{Mor}_{\mathcal{M}^{gen}}(X, Y)$, then $x \sim_\delta x'$ in X, implies $\phi(x) \sim_\delta \phi(x')$ in Y.

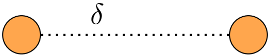
Uniqueness

Let $\Delta_2(\delta)$ be the metric space of two points, where $d(x, y) = \delta$:



Uniqueness

Let $\Delta_2(\delta)$ be the metric space of two points, where $d(x, y) = \delta$:



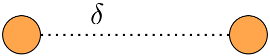
.

Theorem

Assume that $\mathfrak{C} : \mathcal{M}^{gen} \to \mathcal{P}$ *is a clustering functor for which there exists* $\delta_{\mathfrak{C}} > 0$ *with the property that*

Uniqueness

Let $\Delta_2(\delta)$ be the metric space of two points, where $d(x, y) = \delta$:



.

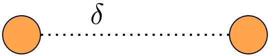
Theorem

Assume that $\mathfrak{C} : \mathcal{M}^{gen} \to \mathcal{P}$ *is a clustering functor for which there exists* $\delta_{\mathfrak{C}} > 0$ *with the property that*

1. $\mathfrak{C}(\Delta_2(\delta))$ *is in <u>one</u> piece for all* $\delta \in [0, \delta_{\mathfrak{C}}]$, *and*

Uniqueness

Let $\Delta_2(\delta)$ be the metric space of two points, where $d(x, y) = \delta$:



.

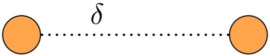
Theorem

Assume that $\mathfrak{C} : \mathcal{M}^{gen} \to \mathcal{P}$ is a clustering functor for which there exists $\delta_{\mathfrak{C}} > 0$ with the property that

1. $\mathfrak{C}(\Delta_2(\delta))$ *is in <u>one</u> piece for all $\delta \in [0, \delta_{\mathfrak{C}}]$, and*
2. $\mathfrak{C}(\Delta_2(\delta))$ *is in <u>two</u> pieces for all $\delta > \delta_{\mathfrak{C}}$.*

Uniqueness

Let $\Delta_2(\delta)$ be the metric space of two points, where $d(x, y) = \delta$:



.

Theorem

Assume that $\mathfrak{C} : \mathcal{M}^{gen} \to \mathcal{P}$ is a clustering functor for which there exists $\delta_{\mathfrak{C}} > 0$ with the property that

1. $\mathfrak{C}(\Delta_2(\delta))$ *is in one piece for all $\delta \in [0, \delta_{\mathfrak{C}}]$, and*
2. $\mathfrak{C}(\Delta_2(\delta))$ *is in two pieces for all $\delta > \delta_{\mathfrak{C}}$.*

Then, $\mathfrak{C}$ is the Vietoris-Rips functor with parameter $\delta_{\mathfrak{C}}$. i.e.
$\mathfrak{C} = \mathfrak{R}_{\delta_{\mathfrak{C}}}$.

- Great so SLC is unique in $\mathcal{M}^{gen}$!

- Great so SLC is unique in $\mathcal{M}^{gen}$!
- Recall Kleinberg's criteria:

- Great so SLC is unique in $\mathcal{M}^{gen}$!
- Recall Kleinberg's criteria:
    - *Scale Invariance:* $\mathfrak{C}(X, d) = \mathfrak{C}(X, \lambda d)\ \forall \lambda > 0$
    - *Surjectivity:* $\forall\ P_X\ \exists d_X$ such that: $\mathfrak{C}(X, d_X) = P_X$
    - *Consistency:* Partition your dataset into blocks.
        - Shrink intrablock distances
        - Expand interblock distances
        - invariance under this transformation

- Great so SLC is unique in $\mathcal{M}^{gen}$!
- Recall Kleinberg's criteria:
  - *Scale Invariance:* $\mathfrak{C}(X, d) = \mathfrak{C}(X, \lambda d)\ \forall \lambda > 0$
  - *Surjectivity:* $\forall\ P_X\ \exists d_X$ such that: $\mathfrak{C}(X, d_X) = P_X$
  - *Consistency:* Partition your dataset into blocks.
    - Shrink intrablock distances
    - Expand interblock distances
    - invariance under this transformation
- To what degree do these properties hold for the VR/SL functor?

More on the VR functor

- The VR functor is surjective, this can be checked easily.

More on the VR functor

- The VR functor is surjective, this can be checked easily.
- Functoriality is our substitute for consistency.

More on the VR functor

- The VR functor is surjective, this can be checked easily.
- Functoriality is our substitute for consistency.
- The Vietoris-Rips functor is **not** scale invariant over $\mathcal{P}$!

More on the VR functor

- The VR functor is surjective, this can be checked easily.
- Functoriality is our substitute for consistency.
- The Vietoris-Rips functor is **not** scale invariant over $\mathcal{P}$!

Theorem (Scale invariance in $\mathcal{M}^{gen}$)

More on the VR functor

- The VR functor is surjective, this can be checked easily.
- Functoriality is our substitute for consistency.
- The Vietoris-Rips functor is **not** scale invariant over $\mathcal{P}$!

Theorem (Scale invariance in $\mathcal{M}^{gen}$)

Let σ_λ be the mapping $(X, d_X) \mapsto (X, \lambda \cdot d_X)$

More on the VR functor

- The VR functor is surjective, this can be checked easily.
- Functoriality is our substitute for consistency.
- The Vietoris-Rips functor is **not** scale invariant over $\mathcal{P}$!

Theorem (Scale invariance in $\mathcal{M}^{gen}$)

Let σ_λ be the mapping $(X, d_X) \mapsto (X, \lambda \cdot d_X)$

Let $\mathfrak{C} : \mathcal{M}^{gen} \to \mathcal{P}$ be a clustering functor s.t. $\mathfrak{C} \circ \sigma_\lambda = \mathfrak{C}$ for all $\lambda > 0$. Then, either

More on the VR functor

- The VR functor is surjective, this can be checked easily.
- Functoriality is our substitute for consistency.
- The Vietoris-Rips functor is **not** scale invariant over $\mathcal{P}$!

Theorem (Scale invariance in $\mathcal{M}^{gen}$)

Let σ_λ be the mapping $(X, d_X) \mapsto (X, \lambda \cdot d_X)$

Let $\mathfrak{C} : \mathcal{M}^{gen} \to \mathcal{P}$ be a clustering functor s.t. $\mathfrak{C} \circ \sigma_\lambda = \mathfrak{C}$ for all $\lambda > 0$. Then, either

- *$\mathfrak{C}$ assigns to each finite metric space X the partition of X into singletons, or*

More on the VR functor

- The VR functor is surjective, this can be checked easily.
- Functoriality is our substitute for consistency.
- The Vietoris-Rips functor is **not** scale invariant over $\mathcal{P}$!

Theorem (Scale invariance in $\mathcal{M}^{gen}$)

Let σ_λ be the mapping $(X, d_X) \mapsto (X, \lambda \cdot d_X)$

Let $\mathfrak{C} : \mathcal{M}^{gen} \to \mathcal{P}$ be a clustering functor s.t. $\mathfrak{C} \circ \sigma_\lambda = \mathfrak{C}$ for all $\lambda > 0$. Then, either

- *$\mathfrak{C}$ assigns to each finite metric space X the partition of X into singletons, or*
- *$\mathfrak{C}$ assigns to each finite metric the partition with only one block.*

More on the VR functor

- The VR functor is surjective, this can be checked easily.
- Functoriality is our substitute for consistency.
- The Vietoris-Rips functor is **not** scale invariant over $\mathcal{P}$!

Theorem (Scale invariance in $\mathcal{M}^{gen}$)

Let σ_λ be the mapping $(X, d_X) \mapsto (X, \lambda \cdot d_X)$

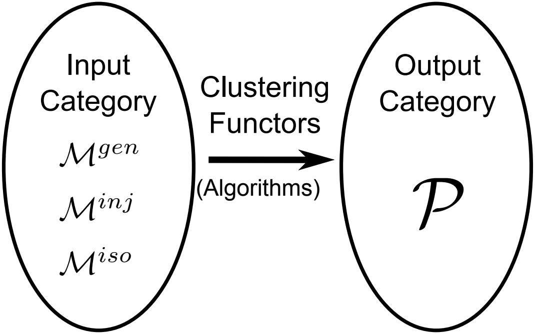
Let $\mathfrak{C} : \mathcal{M}^{gen} \to \mathcal{P}$ be a clustering functor s.t. $\mathfrak{C} \circ \sigma_\lambda = \mathfrak{C}$ for all $\lambda > 0$. Then, either

- *$\mathfrak{C}$ assigns to each finite metric space X the partition of X into singletons, or*
- *$\mathfrak{C}$ assigns to each finite metric the partition with only one block.*

A similar theorem holds in $\mathcal{M}^{inj}$.

Roadmap for classification

- Motivation
- Categories
- Framework
  - Input Categories
  - Output Categories
- Functors
- Results
  - $\mathcal{M}^{iso}$ (small)
  - $\mathcal{M}^{gen}$ (large)
  - $\mathcal{M}^{inj}$ **(medium)**
- Conclusion



A property for functorial algorithms

Definition (Excisive clustering functors)

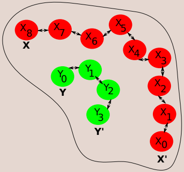
We say that a clustering functor $\mathfrak{C}$ is **excisive** if for all $(X, d_X) \in \mathrm{ob}(\mathcal{M})$, if we write $\mathfrak{C}(X, d_X) = (X, \{X_\alpha\}_{\alpha\in A})$, then $\mathfrak{C}\left(X_\alpha, d_{X|_{X_\alpha\times X_\alpha}}\right) = (X_\alpha, \{X_\alpha\})$ for all $\alpha \in A$.

A property for functorial algorithms

Definition (Excisive clustering functors)

We say that a clustering functor $\mathfrak{C}$ is **excisive** if for all $(X, d_X) \in \mathrm{ob}(\mathcal{M})$, if we write $\mathfrak{C}(X, d_X) = (X, \{X_\alpha\}_{\alpha \in A})$, then $\mathfrak{C}\left(X_\alpha, d_{X|_{X_\alpha \times X_\alpha}}\right) = (X_\alpha, \{X_\alpha\})$ for all $\alpha \in A$.

The Vietoris-Rips functor is excisive.



Metric Space
(X,d)

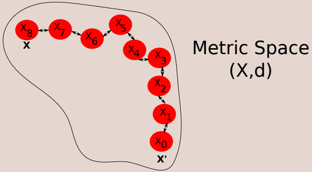
□

A property for functorial algorithms

Definition (Excisive clustering functors)

We say that a clustering functor $\mathfrak{C}$ is **excisive** if for all $(X, d_X) \in \mathrm{ob}(\mathcal{M})$, if we write $\mathfrak{C}(X, d_X) = (X, \{X_\alpha\}_{\alpha \in A})$, then $\mathfrak{C}\left(X_\alpha, d_{X|_{X_\alpha \times X_\alpha}}\right) = (X_\alpha, \{X_\alpha\})$ for all $\alpha \in A$.

The Vietoris-Rips functor is excisive.



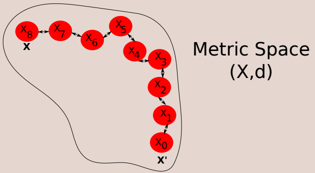
$\square$

A property for functorial algorithms

Definition (Excisive clustering functors)

We say that a clustering functor $\mathfrak{C}$ is **excisive** if for all $(X, d_X) \in \mathrm{ob}(\mathcal{M})$, if we write $\mathfrak{C}(X, d_X) = (X, \{X_\alpha\}_{\alpha \in A})$, then $\mathfrak{C}\left(X_\alpha, d_{X|_{X_\alpha \times X_\alpha}}\right) = (X_\alpha, \{X_\alpha\})$ for all $\alpha \in A$.

The Vietoris-Rips functor is excisive.



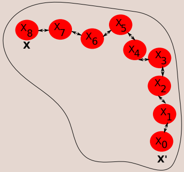
$\square$

Do non-excisive methods exist?

Definition (Excisive clustering functors)

We say that a clustering functor $\mathfrak{C}$ is **excisive** if for all $(X, d_X) \in \mathrm{ob}(\mathcal{M})$, if we write $\mathfrak{C}(X, d_X) = (X, \{X_\alpha\}_{\alpha \in A})$, then $\mathfrak{C}\left(X_\alpha, d_{X|_{X_\alpha \times X_\alpha}}\right) = (X_\alpha, \{X_\alpha\})$ for all $\alpha \in A$.

The Vietoris-Rips functor is excisive.



Metric Space
(X,d)

$\square$

Do non-excisive methods exist? Definitely not in $\mathcal{M}^{gen}$!

Excisiveness in $\mathcal{M}^{inj}$

In $\mathcal{M}^{inj}$ there exist **non-excisive** clustering functors: for each finite metric space X (with at least two points):

Excisiveness in $\mathcal{M}^{inj}$

In $\mathcal{M}^{inj}$ there exist **non-excisive** clustering functors: for each finite metric space X (with at least two points): let

$$\eta(X) := \left(\mathrm{sep}(X)\right)^{-1},$$

where $\mathrm{sep}(X) = \min_{x \neq x'} d_X(x, x')$ (separation of a metric space).

Excisiveness in $\mathcal{M}^{inj}$

In $\mathcal{M}^{inj}$ there exist **non-excisive** clustering functors: for each finite metric space X (with at least two points): let

$$\eta(X) := \left(\mathrm{sep}(X)\right)^{-1},$$

where $\mathrm{sep}(X) = \min_{x \neq x'} d_X(x, x')$ (separation of a metric space). Let $\widehat{\mathfrak{R}}$ be functor assigning to each X partition into equivalence classes of $\sim_{\eta(X)}$ (now parameter δ depends on the space!).

Excisiveness in $\mathcal{M}^{inj}$

In $\mathcal{M}^{inj}$ there exist **non-excisive** clustering functors: for each finite metric space X (with at least two points): let

$$\eta(X) := \left(\mathrm{sep}(X)\right)^{-1},$$

where $\mathrm{sep}(X) = \min_{x \neq x'} d_X(x, x')$ (separation of a metric space). Let $\widehat{\mathfrak{R}}$ be functor assigning to each X partition into equivalence classes of $\sim_{\eta(X)}$ (now parameter δ depends on the space!).

That this indeed defines a functor in $\mathcal{M}^{inj}$ follows from:

$$\mathrm{Mor}_{\mathcal{M}^{inj}}(X, Y) \neq \emptyset \Longrightarrow \mathrm{sep}(X) \geq \mathrm{sep}(Y).$$

Excisiveness in $\mathcal{M}^{inj}$

In $\mathcal{M}^{inj}$ there exist **non-excisive** clustering functors: for each finite metric space X (with at least two points): let

$$\eta(X) := \big(\mathrm{sep}(X)\big)^{-1},$$

where $\mathrm{sep}(X) = \min_{x \neq x'} d_X(x, x')$ (separation of a metric space). Let $\widehat{\mathfrak{R}}$ be functor assigning to each X partition into equivalence classes of $\sim_{\eta(X)}$ (now parameter δ depends on the space!).

That this indeed defines a functor in $\mathcal{M}^{inj}$ follows from:

$$\mathrm{Mor}_{\mathcal{M}^{inj}}(X, Y) \neq \emptyset \Longrightarrow \mathrm{sep}(X) \geq \mathrm{sep}(Y).$$

Indeed, assume that $\phi \in \mathrm{Mor}_{\mathcal{M}^{inj}}(X, Y)$, then for all $x \neq x'$:

$$d_X(x, x') \geq d_Y(\phi(x), \phi(x')) \boxed{\geq} \min_{y \neq y'} d_Y(y, y') = \mathrm{sep}(Y).$$

Excisiveness in $\mathcal{M}^{inj}$

In $\mathcal{M}^{inj}$ there exist **non-excisive** clustering functors: for each finite metric space X (with at least two points): let

$$\eta(X) := \big(\mathrm{sep}(X)\big)^{-1},$$

where $\mathrm{sep}(X) = \min_{x \neq x'} d_X(x, x')$ (separation of a metric space). Let $\widehat{\mathfrak{R}}$ be functor assigning to each X partition into equivalence classes of $\sim_{\eta(X)}$ (now parameter δ depends on the space!).

That this indeed defines a functor in $\mathcal{M}^{inj}$ follows from:

$$\mathrm{Mor}_{\mathcal{M}^{inj}}(X, Y) \neq \emptyset \Longrightarrow \mathrm{sep}(X) \geq \mathrm{sep}(Y).$$

Indeed, assume that $\phi \in \mathrm{Mor}_{\mathcal{M}^{inj}}(X, Y)$, then for all $x \neq x'$:

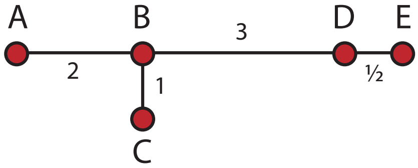
$$d_X(x, x') \geq d_Y(\phi(x), \phi(x')) \boxed{\geq} \min_{y \neq y'} d_Y(y, y') = \mathrm{sep}(Y).$$

Then, $\mathrm{sep}(X) \geq \mathrm{sep}(Y)$, and since $\mathfrak{R}_\delta$ is functorial and $\eta(Y) \geq \eta(X)$,

$$x \sim_{\eta(X)} x' \Longrightarrow \phi(x) \sim_{\eta(X)} \phi(x') \Longrightarrow \phi(x) \sim_{\eta(Y)} \phi(x').$$

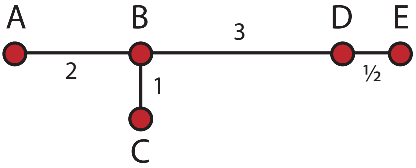
Excisiveness in $\mathcal{M}^{inj}$

Consider X to be the metric space below (graph distance):



Excisiveness in $\mathcal{M}^{inj}$

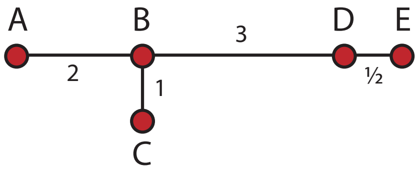
Consider X to be the metric space below (graph distance):



Note that $\eta(X) = 2$

Excisiveness in $\mathcal{M}^{inj}$

Consider X to be the metric space below (graph distance):

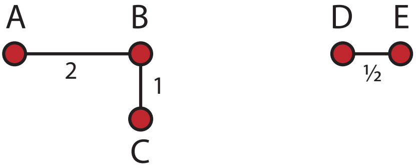


Note that $\eta(X) = 2 \Longrightarrow$

$$\widehat{\mathfrak{R}}(X, d_X) = \mathfrak{R}_2(X, d_X) = (X, \{\{A, B, C\}, \{D, E\}\}).$$

Excisiveness in $\mathcal{M}^{inj}$

Consider X to be the metric space below (graph distance):



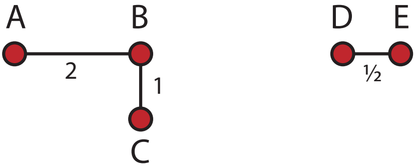
Note that $\eta(X) = 2 \implies$

$$\widehat{\mathfrak{R}}(X, d_X) = \mathfrak{R}_{2}(X, d_X) = (X, \{\{A, B, C\}, \{D, E\}\}).$$

Now, if $Y = \{A, B, C\}$, then $\eta(Y) = 1 \implies \{A, B, C\}$ gets further partitioned! Then, $\widehat{\mathfrak{R}}$ is **not-excisive**.

Excisiveness in $\mathcal{M}^{inj}$

Consider X to be the metric space below (graph distance):

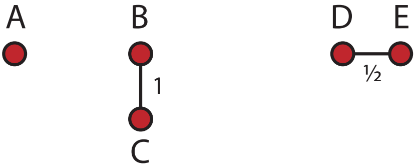


Note that $\eta(X) = 2 \implies$

$$\widehat{\mathfrak{R}}(X, d_X) = \mathfrak{R}_2(X, d_X) = (X, \{\{A, B, C\}, \{D, E\}\}).$$

Now, if $Y = \{A, B, C\}$, then $\eta(Y) = 1 \implies \{A, B, C\}$ gets further partitioned! Then, $\widehat{\mathfrak{R}}$ is **not-excisive**.

Consider X to be the metric space below (graph distance):



Note that $\eta(X) = 2 \Longrightarrow$

$$\widehat{\mathfrak{R}}(X, d_X) = \mathfrak{R}_2(X, d_X) = (X, \{\{A, B, C\}, \{D, E\}\}).$$

Now, if $Y = \{A, B, C\}$, then $\eta(Y) = 1 \Longrightarrow \{A, B, C\}$ gets further partitioned! Then, $\widehat{\mathfrak{R}}$ is **not-excisive**.

a factory of monsters

Let $\iota : \mathcal{M}^{inj} \to (\mathbb{R}, \geq)$ by any non-constant contra-variant functor:

$$\mathrm{Mor}_{\mathcal{M}^{inj}}(X, Y) \neq \emptyset \Longrightarrow \iota(X) \leq \iota(Y).$$

For each of them know how to build a non-excisive monster: $\mathfrak{R}_{\iota(\cdot)}$.

Let $\iota : \mathcal{M}^{inj} \to (\mathbb{R}, \geq)$ by any <u>non-constant</u> contra-variant functor:

$$\mathrm{Mor}_{\mathcal{M}^{inj}}(X, Y) \neq \emptyset \Longrightarrow \iota(X) \leq \iota(Y).$$

For each of them know how to build a non-excisive monster: $\mathfrak{R}_{\iota(\cdot)}$.

We saw that no such ι exist on $\mathcal{M}^{gen}$: uniqueness of VR.

a factory of monsters

Let $\iota : \mathcal{M}^{inj} \to (\mathbb{R}, \geq)$ by any non-constant contra-variant functor:

$$\mathrm{Mor}_{\mathcal{M}^{inj}}(X, Y) \neq \emptyset \Longrightarrow \iota(X) \leq \iota(Y).$$

For each of them know how to build a non-excisive monster: $\mathfrak{R}_{\iota(\cdot)}$.

We saw that no such ι exist on $\mathcal{M}^{gen}$: uniqueness of VR.

Indirectly: $X \to Y \to X$ is always a possible diagram on $\mathcal{M}^{gen}$.
Apply ι and get $\iota(X) = \iota(Y)$.

And now something completely different..

Generalize single linkage clustering

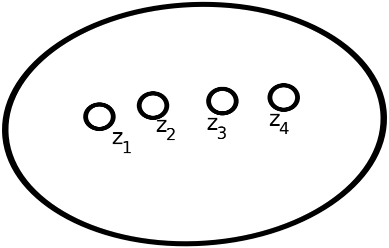
Notice the following: given $\delta \geq 0$, $(X, d_X) \in \mathcal{M}$ and $x, x' \in X$,

$$d_X(x, x') \leq \delta \iff \exists \phi \in \mathrm{Mor}_{\mathcal{M}}(\Delta_2(\delta), X) \text{ s.t. } \{x, x'\} \in \mathrm{im}(\phi).$$

Generalize single linkage clustering

Notice the following: given $\delta \geq 0$, $(X, d_X) \in \mathcal{M}$ and $x, x' \in X$,

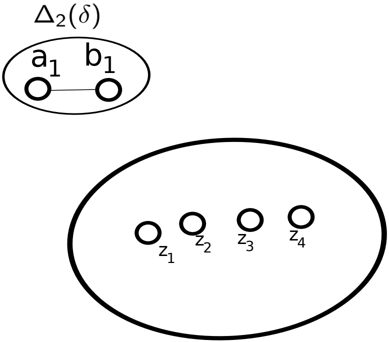
$$d_X(x, x') \leq \delta \Longleftrightarrow \exists \phi \in \mathrm{Mor}_{\mathcal{M}}(\Delta_2(\delta), X) \,\text{s.t.}\; \{x, x'\} \in \mathrm{im}(\phi).$$



Generalize single linkage clustering

Notice the following: given $\delta \geq 0$, $(X, d_X) \in \mathcal{M}$ and $x, x' \in X$,

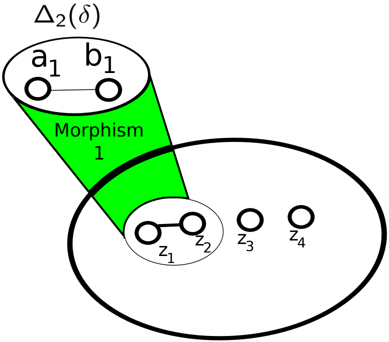
$$d_X(x, x') \leq \delta \iff \exists \phi \in \mathrm{Mor}_{\mathcal{M}}(\Delta_2(\delta), X) \text{ s.t. } \{x, x'\} \in \mathrm{im}(\phi).$$



Generalize single linkage clustering

Notice the following: given $\delta \geq 0$, $(X, d_X) \in \mathcal{M}$ and $x, x' \in X$,

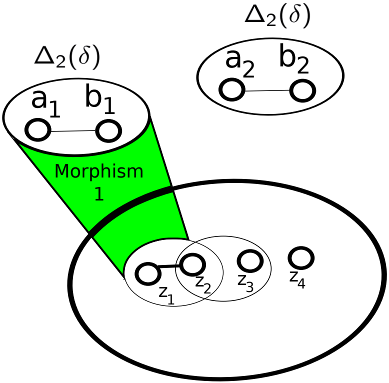
$$d_X(x, x') \leq \delta \iff \exists \phi \in \mathrm{Mor}_{\mathcal{M}}(\Delta_2(\delta), X) \text{ s.t. } \{x, x'\} \in \mathrm{im}(\phi).$$



Generalize single linkage clustering

Notice the following: given $\delta \geq 0$, $(X, d_X) \in \mathcal{M}$ and $x, x' \in X$,

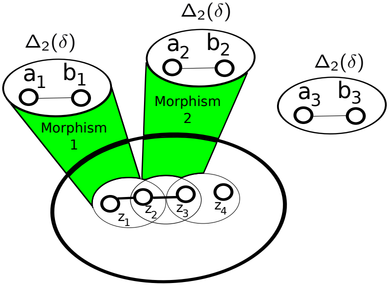
$$d_X(x, x') \leq \delta \iff \exists \phi \in \mathrm{Mor}_{\mathcal{M}}(\Delta_2(\delta), X) \text{ s.t. } \{x, x'\} \in \mathrm{im}(\phi).$$



Generalize single linkage clustering

Notice the following: given $\delta \geq 0$, $(X, d_X) \in \mathcal{M}$ and $x, x' \in X$,

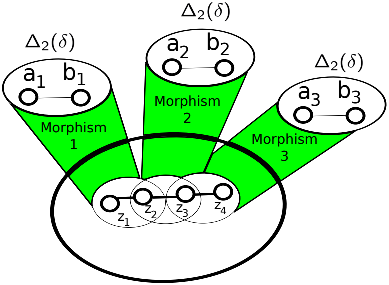
$$d_X(x, x') \leq \delta \iff \exists \phi \in \mathrm{Mor}_{\mathcal{M}}(\Delta_2(\delta), X) \text{ s.t. } \{x, x'\} \in \mathrm{im}(\phi).$$



Generalize single linkage clustering

Notice the following: given $\delta \geq 0$, $(X, d_X) \in \mathcal{M}$ and $x, x' \in X$,

$$d_X(x, x') \leq \delta \iff \exists \phi \in \mathrm{Mor}_{\mathcal{M}}(\Delta_2(\delta), X) \text{ s.t. } \{x, x'\} \in \mathrm{im}(\phi).$$



What if we use something else instead of $\Delta_2(\delta)$?

Generalize single linkage clustering

Definition (representable functors)

We say a functor $\mathfrak{C}$ is **representable** if there is a collection Ω of finite metric spaces such that: points x and x' in X are in the same block of $\mathfrak{C}(X, d_X)$ if and only if there exist:

Generalize single linkage clustering

Definition (representable functors)

We say a functor $\mathfrak{C}$ is **representable** if there is a collection Ω of finite metric spaces such that: points x and x' in X are in the same block of $\mathfrak{C}(X, d_X)$ if and only if there exist:

- a sequence of points $z_0, \ldots, z_k \in X$ with $z_0 = x$ and $z_k = x'$,

Generalize single linkage clustering

Definition (representable functors)

We say a functor $\mathfrak{C}$ is **representable** if there is a collection Ω of finite metric spaces such that: points x and x' in X are in the same block of $\mathfrak{C}(X, d_X)$ if and only if there exist:

- a sequence of points $z_0, \ldots, z_k \in X$ with $z_0 = x$ and $z_k = x'$,
- a sequence of metric spaces $\omega_1, \ldots, \omega_k \in \Omega$,

Generalize single linkage clustering

Definition (representable functors)

We say a functor $\mathfrak{C}$ is **representable** if there is a collection Ω of finite metric spaces such that: points x and x' in X are in the same block of $\mathfrak{C}(X, d_X)$ if and only if there exist:

- a sequence of points $z_0, \ldots, z_k \in X$ with $z_0 = x$ and $z_k = x'$,
- a sequence of metric spaces $\omega_1, \ldots, \omega_k \in \Omega$,
- for each $i = 1, \ldots, k$, pairs of points $(\alpha_i, \beta_i) \in \omega_i$ and morphisms $f_i \in \mathrm{Mor}_{\mathcal{M}}(\omega_i, X)$ s.t. $f_i(\alpha_i) = z_{i-1}$ and $f_i(\beta_i) = z_i$.

Generalize single linkage clustering

Definition (representable functors)

We say a functor $\mathfrak{C}$ is **representable** if there is a collection Ω of finite metric spaces such that: points x and x' in X are in the same block of $\mathfrak{C}(X, d_X)$ if and only if there exist:

- a sequence of points $z_0, \ldots, z_k \in X$ with $z_0 = x$ and $z_k = x'$,
- a sequence of metric spaces $\omega_1, \ldots, \omega_k \in \Omega$,
- for each $i = 1, \ldots, k$, pairs of points $(\alpha_i, \beta_i) \in \omega_i$ and morphisms $f_i \in \mathrm{Mor}_{\mathcal{M}}(\omega_i, X)$ s.t. $f_i(\alpha_i) = z_{i-1}$ and $f_i(\beta_i) = z_i$.

If Ω is finite, then we say that $\mathfrak{C}$ is **finitely representable**. We write $\mathfrak{C}^{\Omega}$ for a functor that is represented by Ω.

Generalize single linkage clustering

Definition (representable functors)

We say a functor $\mathfrak{C}$ is **representable** if there is a collection Ω of finite metric spaces such that: points x and x' in X are in the same block of $\mathfrak{C}(X, d_X)$ if and only if there exist:

- a sequence of points $z_0, \ldots, z_k \in X$ with $z_0 = x$ and $z_k = x'$,
- a sequence of metric spaces $\omega_1, \ldots, \omega_k \in \Omega$,
- for each $i = 1, \ldots, k$, pairs of points $(\alpha_i, \beta_i) \in \omega_i$ and morphisms $f_i \in \mathrm{Mor}_{\mathcal{M}}(\omega_i, X)$ s.t. $f_i(\alpha_i) = z_{i-1}$ and $f_i(\beta_i) = z_i$.

If Ω is finite, then we say that $\mathfrak{C}$ is **finitely representable**. We write $\mathfrak{C}^{\Omega}$ for a functor that is represented by Ω.

Notice if $\Omega = \{\Delta_2(\delta)\}$ then we see that $\mathfrak{C}^{\Omega}$ is the VR functor, thus it is representable.

Generalize single linkage clustering

Definition (representable functors)

We say a functor $\mathfrak{C}$ is **representable** if there is a collection Ω of finite metric spaces such that: points x and x' in X are in the same block of $\mathfrak{C}(X, d_X)$ if and only if there exist:

- a sequence of points $z_0, \ldots, z_k \in X$ with $z_0 = x$ and $z_k = x'$,
- a sequence of metric spaces $\omega_1, \ldots, \omega_k \in \Omega$,
- for each $i = 1, \ldots, k$, pairs of points $(\alpha_i, \beta_i) \in \omega_i$ and morphisms $f_i \in \mathrm{Mor}_{\mathcal{M}}(\omega_i, X)$ s.t. $f_i(\alpha_i) = z_{i-1}$ and $f_i(\beta_i) = z_i$.

If Ω is finite, then we say that $\mathfrak{C}$ is **finitely representable**. We write $\mathfrak{C}^{\Omega}$ for a functor that is represented by Ω.

Notice if $\Omega = \{\Delta_2(\delta)\}$ then we see that $\mathfrak{C}^{\Omega}$ is the VR functor, thus it is representable. Representability means: parametric description of clustering methods.

Clustering in $\mathcal{M}^{inj}$ (and $\mathcal{M}^{gen}$)

Theorem (Excisive $=$ Representable)

A clustering functor is excisive if and only if it is representable.

Clustering in $\mathcal{M}^{inj}$ (and $\mathcal{M}^{gen}$)

Theorem (Excisive = Representable)

A clustering functor is excisive if and only if it is representable.

Theorem (Factorization)

If $\mathcal{M}$ is either $\mathcal{M}^{inj}$ or $\mathcal{M}^{gen}$ then if $\mathfrak{C}$ is any $\mathcal{M}$-functorial finitely represented functor, represented by Ω,

Then $\mathfrak{C} = \mathfrak{R}_1 \circ \boxed{\mathfrak{T}^{\Omega}}$

Clustering in $\mathcal{M}^{inj}$ (and $\mathcal{M}^{gen}$)

Theorem (Excisive = Representable)

A clustering functor is excisive if and only if it is representable.

Theorem (Factorization)

If $\mathcal{M}$ is either $\mathcal{M}^{inj}$ or $\mathcal{M}^{gen}$ then if $\mathfrak{C}$ is any $\mathcal{M}$-functorial finitely represented functor, represented by Ω,

Then $\mathfrak{C} = \mathfrak{R}_1 \circ \boxed{\mathfrak{T}^\Omega} \Leftarrow$ a certain transformation that changes the metric:

$$\mathfrak{T}^\Omega : \mathcal{M} \to \mathcal{M}$$

.

Excisive clustering functors in $\mathcal{M}^{inj}$

Our factorization theorem suggests how to change metrics to account for **density**– ameliorate SL's **chaining effect**.

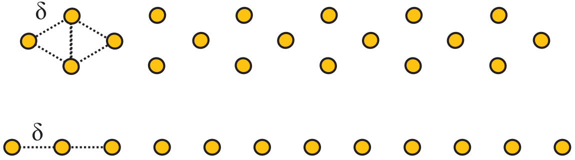


Figure : $\mathfrak{C}^{\Omega}$, with $\Omega = \{\Delta_n(\delta)\}$ for increasing n, requires more and more density.

Excisive clustering functors in $\mathcal{M}^{inj}$

Our factorization theorem suggests how to change metrics to account for **density**– ameliorate SL's **chaining effect**.

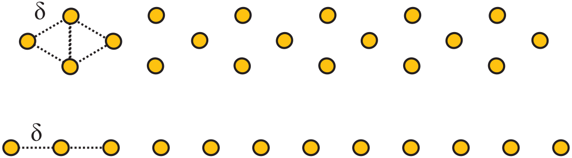


Figure : $\mathfrak{C}^{\Omega}$, with $\Omega = \{\Delta_n(\delta)\}$ for increasing n, requires more and more density.

Related to **DBSCAN** (network clustering, and database mining)!

More generality in $\mathcal{M}^{inj}$

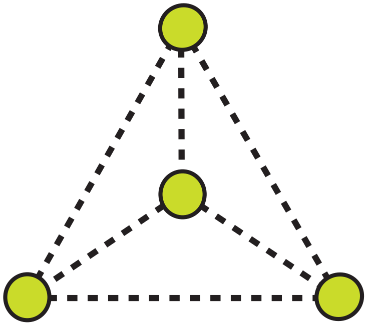


Figure : A metric space that encodes a notion of "density" different from that provided by $\Delta_3(\delta)$.

Same thing for HC methods

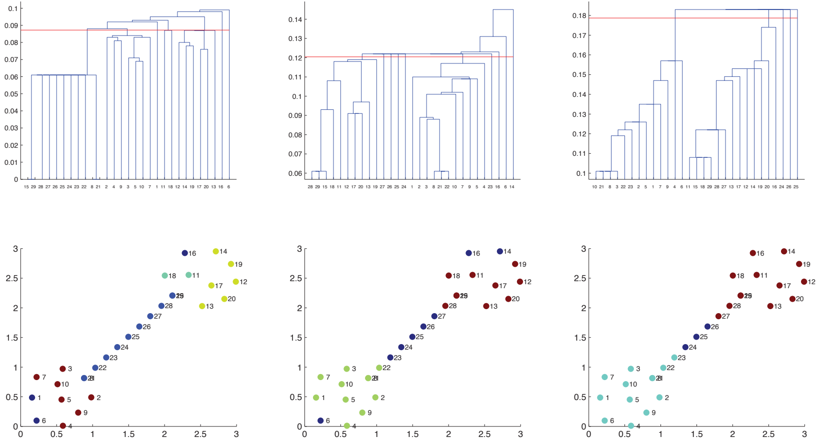
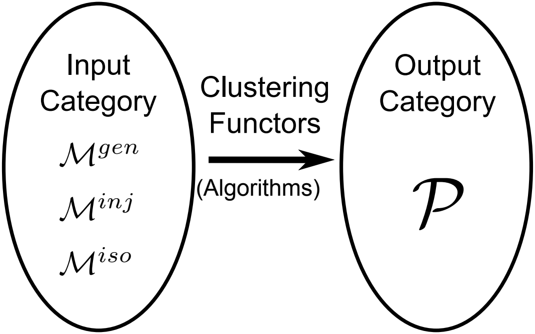


Figure : Dendrograms arising from $\mathfrak{R}^{\Delta_m}$ applied to a randomly generated points in $\mathbb{R}^2$. Top: from left to right we show dendrograms corresponding to $\mathbf{m} = \mathbf{2}, \mathbf{3}$ and $\mathbf{4}$. Bottom: partitioning induced on data by each dendrogram using parameters corresponding to red lines shown over dendrograms.

Roadmap for classification

- Motivation
- Categories
- Framework
  - Input Categories
  - Output Categories
- Functors
- Results
  - $\mathcal{M}^{iso}$ (small)
  - $\mathcal{M}^{gen}$ (large)
  - $\mathcal{M}^{inj}$ (medium)
- **Discussion**



Conclusion

- Motivation
- Categories
- Framework
  - Input Categories
  - Output Categories
- Functors
- Results
  - $\mathcal{M}^{iso}$ (small)
  - $\mathcal{M}^{gen}$ (large)
  - $\mathcal{M}^{inj}$ (medium)
- Conclusion

Conclusion

- Motivation
- Categories
- Framework
  - Input Categories
  - Output Categories
- Functors
- Results
  - $\mathcal{M}^{iso}$ (small)
  - $\mathcal{M}^{gen}$ (large)
  - $\mathcal{M}^{inj}$ (medium)
- Conclusion

- Kleinberg's nonexistence result.

Conclusion

- Motivation
- Categories
- Framework
  - Input Categories
  - Output Categories
- Functors
- Results
  - $\mathcal{M}^{iso}$ (small)
  - $\mathcal{M}^{gen}$ (large)
  - $\mathcal{M}^{inj}$ (medium)
- Conclusion

- Kleinberg's nonexistence result.
- category theory: language for **reasoning** about clustering.

Conclusion

- Motivation
- Categories
- Framework
  - Input Categories
  - Output Categories
- Functors
- Results
  - $\mathcal{M}^{iso}$ (small)
  - $\mathcal{M}^{gen}$ (large)
  - $\mathcal{M}^{inj}$ (medium)
- Conclusion

- Kleinberg's nonexistence result.
- category theory: language for **reasoning** about clustering.
- Future: better understanding of non-excisive functors in $\mathcal{M}^{inj}$.

Conclusion

- Kleinberg's nonexistence result.
- category theory: language for **reasoning** about clustering.
- Future: better understanding of non-excisive functors in $\mathcal{M}^{inj}$.
- note: also established results for **hierarchical clustering**

Conclusion

- Kleinberg's nonexistence result.
- category theory: language for **reasoning** about clustering.
- Future: better understanding of non-excisive functors in $\mathcal{M}^{inj}$.
- note: also established results for **hierarchical clustering**
- In context of HC: metric structures on $\mathcal{M}$ lead to **stability** and **convergence** (Gromov-Hausdorff distances).

Conclusion

- Kleinberg's nonexistence result.
- category theory: language for **reasoning** about clustering.
- Future: better understanding of non-excisive functors in $\mathcal{M}^{inj}$.
- note: also established results for **hierarchical clustering**
- In context of HC: metric structures on $\mathcal{M}$ lead to **stability** and **convergence** (Gromov-Hausdorff distances).
- Paper in FoCM, 2013. Also JMLR, 2010.
- Ongoing work on extending to **networks**. Joint with Carlsson, Ribeiro, Segarra. Some unexpected results.

Acknowledgements

- DARPA grant HR0011-05-1-0007
- ONR grant N00014-09-1-0783
- Stanford University, The Ohio State University.
- **Ryan Lewis** for help preparing the slides!

Any questions?