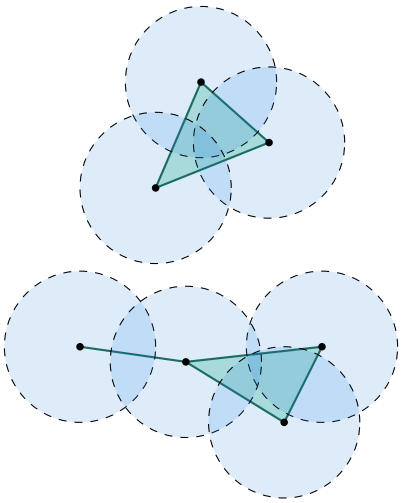
ETH zürich

Bachelor Thesis

Classification & Uniqueness of Clustering Functors

Dominique F. Garmier

November 3, 2024



Supervisor: Dr. Sara Kališnik Hintz
Department of Mathematics, ETH Zürich

Acknowledgements

I would like to thank my supervisor Dr. Sara Kališnik Hintz not only for her guidance and support throughout the writing of this thesis but also for introducing me to the fascinating subject of algebraic topology through her lectures. I would also like to thank Anna Walder, Eric Ceglie, Samuel Huber, and Sébastien Garmier for their editorial comments and suggestions.

Abstract

Clustering algorithms are a widely used tool in machine learning and data science that organize data points into "groups" by recognizing patterns. They have proven useful for dealing with ever-increasing amounts of data. We review a formalization and classification of clustering algorithms developed by Carlsson and Mémoli [CM10b]. For such formal clustering algorithms we will use the term *clustering functor*. An important theoretical result in the study of clustering algorithms is an impossibility theorem by Kleinberg [Kle02]. It states that no clustering algorithm can be *rich*, *consistent* and *scale invariant* at the same time. As identified by Carlsson and Mémoli, the so-called *Vietoris-Rips* clustering functor has some unique characterizing properties [CM10a, Thm. 18], [CM10b, Thm. 7.1]. The Vietoris-Rips clustering functor is based on the idea that for some threshold parameter $\delta > 0$, we assign two data points to the same cluster if, according to some metric, they are δ-close to each other. It was previously shown that the Vietoris-Rips clustering functor satisfies a set of modified conditions from Kleinberg's impossibility theorem [CM10b, Sec. 7.3.1]. By showing that these modified conditions imply the characterizing properties of the Vietoris-Rips clustering functor discussed by Carlsson and Mémoli, we were able to prove that the Vietoris-Rips clustering functor uniquely satisfies the modified conditions from Kleinberg's impossibility theorem.

Contents

Introduction

A central problem in machine learning and data science is the task of clustering data, i.e., "grouping" data points according to some patterns. Underlying most clustering techniques is the notion of a distance between data points. Motivated by this, we think of *data* as a finite metric space, i.e. a finite set equipped with a metric and of a clustering algorithm as a map assigning to a finite metric space either a *partition* (Figure 1a) or a *dendrogram* (Figure 1b) of the underlying set.

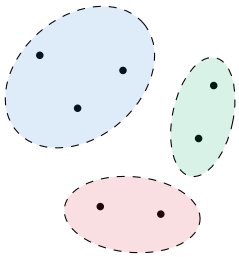


Figure 1a: A partition of seven points.

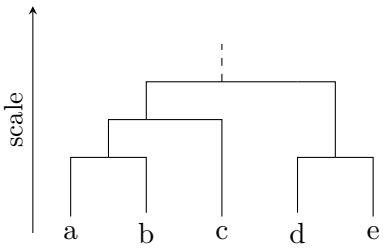


Figure 1b: A dendrogram of five points.

A dendrogram is a (rooted) tree with data points as leaves. The dendrogram represents the order in which data points are "merged" into clusters. In Figure 1b, we have a dendrogram over the points $\{a, b, c, d, e\}$. Here, we first merge a and b, as well as d and e into respective pairs. Then, we add c to the cluster containing a and b. Finally, we merge all five points into a single cluster. By stopping this merging process early, we obtain a partition of the data points. In this sense, dendrograms are more general than partitions. Depending on the clustering algorithm used, a dendrogram may need to satisfy additional properties. Clustering algorithms that produce a partition are sometimes called *classical* and those that produce a dendrogram are called *hierarchical*.

In this thesis we use the language of category theory to succinctly describe properties of clustering algorithms. In particular, clustering algorithms are seen as special functors mapping from appropriate categories of finite metric spaces to partitions or dendrograms. For this reason, we refer to such clustering algorithms as *classical* and *hierarchical clustering functors* respectively [CM10b]. We do not require the reader to be familiar with category theory; we introduce and give examples of all the necessary concepts.

First we work with classical clustering functors, where two central properties of clustering functors are considered.

- *Surjectivity*, which means that for any partition of a set of n points we can find a metric on these n points such that our clustering functor produces this partition.
- *Splitting*, which refers to a particular behaviour of clustering functors on metric spaces with two points. Namely, two points are "merged" into a single cluster if and only if they are δ-close for some $\delta > 0$.

The latter is a characterizing property of the so-called *Vietoris-Rips* clustering functor [CM10b, Thm. 6.4]. In this thesis we show that under certain technical assumptions surjectivity implies splitting. In some sense this means that a global property of a clustering functor (surjectivity) implies a local property (splitting).

For classical clustering algorithms there exists an important impossibility theorem by Kleinberg [Kle02] which states that there exists no clustering algorithm satisfying the following three properties at the same time:

- *Richness* is the term used by Kleinberg to describe surjectivity.
- *Consistency* is the property that decreasing the distance between points inside a cluster does not change the clustering.
- *Scale invariance* refers to the fact that we can rescale the metric and the clustering remains the same.

At the end we consider a modified version of the above conditions presented in [CM10b, Sec. 7.3.1] that concern hierarchical clustering functors, more precisely, *scaling* hierarchical clustering functors. We show that scaling hierarchical clustering functors are essentially the same as classical clustering functors[1]. Utilizing this trick and the tools described earlier, we can show that the modified Kleinberg conditions give rise to a unique (up to some transformation) hierarchical clustering functor.

[1] We show that there is a one-to-one correspondence between scaling hierarchical clustering functors and what we call *regular* classical clustering functors.

Chapter 1

Data Clustering

Data clustering refers to the procedure of assigning to a set of data points some kind of "grouping". This is often used as a first step in machine learning pipelines as a way to reduce data complexity. In the simplest case this "grouping" is a *partition* of the original data points. This is also called *classical clustering* (see Figure 1a). Alternatively, we could assign a so-called *dendrogram* to the data points, which is referred to as *hierarchical clustering* (see Figure 1b). The first part this chapter explains these concepts in more detail.

Next, we discuss two examples of commonly studied clustering algorithms. Namely, *k-means clustering*, a classical clustering technique and *linkage clustering*, a hierarchical clustering algorithm. By contextualizing these algorithms with computations we demonstrate how they can fall short in certain cases. This part is based on the books [ELLS11] and [SSMÁU21].

Finally, we show how clustering algorithms can be seen as remaining invariant under certain transformations of their input. One such transformation could be the introduction of noise into the data. If we model this noise as some transformation of the "ground truth" we can then try to construct a clustering algorithm that is invariant under this transformation. In other words, the output of our clustering algorithm would not depend on the noise. This idea naturally leads to Kleinberg's impossibility result [Kle02].

1.1 Partitions, Dendrograms and Clustering Algorithms

In this section, we make precise what we mean by a "partition" or a "dendrogram", which will enable us to formally define *clustering algorithms*.

Definition 1.1: Partitions of Finite Sets

Let X be a finite set. A *partition* P of X is the set of equivalence classes $X/_{\sim_P}$ for some equivalence relation $\sim_P$ on X. The set of all such partitions is denoted by $\mathfrak{P}(X)$. Furthermore:

- We interchangeably use P (the partition) and $\sim_P$ (the corresponding equivalence relation).
- A single equivalence class $X_\alpha \in P$ will be called a *block* or *part* of P.

It naturally makes sense to use partitions as outputs of (classical) clustering algorithms, where a block of a partition is to be interpreted as a *cluster*.

When describing a partition it is often easier to describe a minimal set of conditions the partition relation must satisfy. More formally, we talk about taking the *transitive closure* of a relation.

Definition 1.2

Given a relation $\sim$ on a set X its *transitive closure* is the transitive relation $\sim^+$ such that for $x, y \in X$ we have

$$x \sim^+ y$$

if there exists a sequence $x = x_0, x_1, \ldots, x_n = y$ with $x_i \sim_R x_{i+1}$ for all $i = 0, \ldots, n-1$ [LP97, p.337]. In this case we also say that $\sim^+$ is *generated* by $\sim$.

In particular, if $\sim$ is reflexive and symmetric then $\sim^+$ is an equivalence relation. Later we will use this fact to define equivalence relations as transitive closures of reflexive and symmetric relations.

Example 1.3

Let $G = (V, E)$ be a graph, then E is a binary symmetric relation on V. Two points $v, w \in V$ lie in the same connected component of G if there exists a path, i.e., a sequence $v = v_0, v_1, \ldots, v_n = w$ such that $(v_i, v_{i+1}) \in E$ for all $i = 0, \ldots, n-1$. The resulting partition of V into connected components corresponds to the transitive closure of E.

To define a *dendrogram* we need to give a precise meaning to what we previously informally described as the "merging" of clusters.

Definition 1.4: Refinement

Let $P, Q \in \mathfrak{P}(X)$. Then we write $P \preceq Q$ and say that P *refines* Q if

$$\forall x, y \in X : x \sim_P y \implies x \sim_Q y.$$

This defines a partial order on $\mathfrak{P}(X)$.

Example 1.5

Consider $X := \{a, b, c\}$ together with partitions

$$P := \{\{a\}, \{b\}, \{c\}\} \quad \text{and} \quad Q := \{\{a, b\}, \{c\}\}.$$

As the blocks in P are always contained in some block of Q we have $P \preceq Q$.

This corresponds to the idea of "merging" in the sense that if we have $P, Q \in \mathfrak{P}(X)$ such that $P \preceq Q$ we can say that Q can be obtained from P by "merging" blocks of P.

There are two extreme kinds of partitions: one where all points belong to the same equivalence class, and another where each point forms its own equivalence class. For this we introduce the following notation.

Definition 1.6

We say that $P \in \mathfrak{P}(X)$ is:

1. *discrete* if $x \sim_P y \iff x = y$.
2. *trivial* if $x \sim_P y$ for all $x, y \in X$.

Naturally, if Q is discrete and P is trivial then $Q \preceq R \preceq P$ for all $R \in \mathfrak{P}(X)$.

With this we now have everything to define a *dendrogram*.

Definition 1.7: Dendrogram [CM10b, Def. 2.2]

Let X be a finite set. A map $\theta : \mathbb{R}_{\geq 0} \to \mathfrak{P}(X)$ with

1. $\forall r, s \in \mathbb{R}_{\geq 0} : r \leq s \implies \theta(r) \preceq \theta(s)$,
2. $\exists r, s \in \mathbb{R}_{\geq 0}$ such that $\theta(r)$ is trivial and $\theta(s)$ is discrete[a],
3. $\forall r \in \mathbb{R}_{\geq 0}\, \exists \varepsilon > 0$ such that θ is constant on $[r, r + \varepsilon)$,

is called a *dendrogram* of X. Sometimes r is referred to as the *scale*.

[a] In particular, we get that $\theta(0)$ is always discrete.

This will be familiar to anyone who has seen *persistent homology* where such properties are referred to as *persistence* [Car14, Chap. 3].

As for the relevance of the last condition, consider the following remark.

Remark 1.8

Since X is a finite set, we could decide to omit the third condition in the definition of a dendrogram. Dendrograms would still have discrete scale i.e. the dendrogram would still be constant on intervals. However, it would not be clear what value the dendrogram would take at the endpoints of these intervals. This technicality will become important for some uniqueness theorems we present later.

Example 1.9

As an example of a dendrogram consider Figure 1.1.

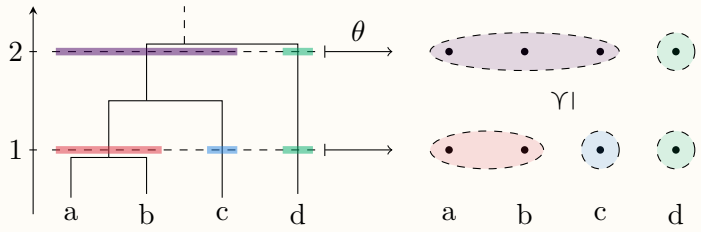


Figure 1.1: A dendrogram $\theta : \mathbb{R}_{\geq 0} \to \mathfrak{P}(X)$ with the four points $X = \{a, b, c, d\}$.

Notice how in this case the monotonicity condition is satisfied e.g. $\theta(1) \preceq \theta(2)$ as shown on the righthand side. The second condition of a dendrogram is also met as it becomes trivial at the top and discrete at the bottom. Finally, the third condition tells us what value the dendrogram takes whenever a "merge happens".

With this we can now state the definition of a *clustering algorithm*.

Definition 1.10: Clustering Algorithm

A *clustering algorithm* $\mathfrak{C}$ is a procedure that assigns to a finite metric space (X, d) either a partition or dendrogram, which we denote by

$$\mathfrak{C}(X, d).$$

In the case that $\mathfrak{C}$ outputs partitions, we refer to it as a *classical* clustering algorithm. If it outputs dendrograms, we refer to it as *hierarchical*.

In Chapter 3 we will revisit this definition, at which point we will call it a *clustering functor*.

1.2 Examples of Clustering Algorithms

Let us consider a few examples of clustering algorithms.

1.2.1 *k*-means

A common method of clustering is to find a partition of the data into k parts that minimizes some objective function. One such example is the *k-means* clustering objective [SSMÁU21, Sec. 3.1].

Definition 1.11: k-means Clustering

Let $\{x_1, \dots, x_n\} \subset \mathbb{R}^d$. For $k \in \mathbb{N}$ *k-means clustering* refers to finding a partition $C_1, \dots, C_k$ of $\{x_1, \dots, x_n\}$ such that

$$\sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|_2^2$$

is minimal, where $\mu_k := \frac{1}{|C_k|} \sum_{x \in C_k} x$ is the *center* of C_k[a].

[a] In case $C_k = \emptyset$, we set $\mu_k = 0 \in \mathbb{R}^d$.

To find such a partition we would have to check all possible partitions of $x_1, \dots, x_n$. This is computationally infeasible. In practice, we use approximate methods such as Lloyd's algorithm, which is sometimes simply referred to as *the k-means algorithm* [SSMÁU21, Sec. 3.1.2].

Definition 1.12: Lloyd's Algorithm

Given $\{x_1, \dots, x_n\} \subset \mathbb{R}^d$ and some *centers* $z_1, \dots, z_k \in \mathbb{R}^d$ Lloyd's algorithm consists of iteratively applying the following two steps:

1. **Assignment Step:** Assign each point x_i to the cluster C_j such that

$$j = \operatorname*{argmin}_{1 \le l \le k} \|x_i - z_l\|_2^2.$$

2. **Update Step:** Update the centers $z_1, \dots, z_k$ by setting

$$z_j = \frac{1}{|C_j|} \sum_{x \in C_j} x \quad \forall j \in \{1, \dots, k\},$$

where $z_j = 0$ if $C_j = \emptyset$.

One can show that Lloyd's algorithm converges after finitely many steps [SSMÁU21, Thm. 3.14]. We will see that the choice of starting centers $z_1, \dots, z_k$ can have a significant impact on the outcome of the algorithm.

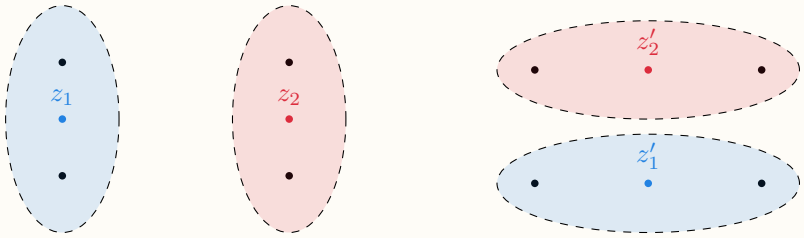
Example 1.13

Consider the set $\{(0,0),(0,1),(2,0),(2,1)\} \subset \mathbb{R}^2$, $k = 2$ and two possible choices of starting centers:

1. $z_1 = (0, 1/2)$ and $z_2 = (2, 1/2)$;
2. $z_1' = (1, 0)$ and $z_2' = (1, 1)$.

Consider the first case. In the first assignment step we assign $(0,0)$ and $(0,1)$ to the first cluster C_1 and similarly $(2,0)$ and $(2,1)$ to the second cluster C_2. In the update step we update the centers and get $z_1 = \frac{1}{2}[(0,0) + (0,1)] = (0, 1/2)$. Similarly, we get $z_2 = (2, 1/2)$. Notice that these are the same centers as we started with. Therefore, the algorithm converges after just one iteration.

Repeating the same computations for the second case we notice that the algorithm will also converge after just one step to the original centers. This will result in the red and blue clusters as seen in Figure 1.2.



starting centers (1) starting centers (2)

Figure 1.2: k-means clustering for different starting centers.

In fact the first solution is the one minimizing the k-means objective. So we see that the choice of initial centers is important for the outcome of the algorithm. For this reason in practice one might run the algorithm multiple times with different initial centers drawn from some distribution and then choose the best solution.

Another potential downside of the k-means algorithm is that its clusters are always convex.

Example 1.14

Consider the following data points in Figure 1.3 for $k = 2$.

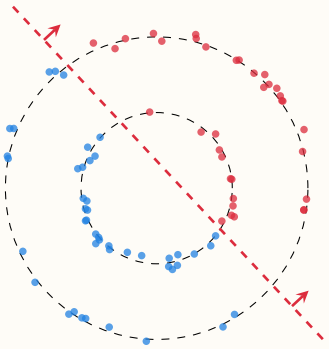


Figure 1.3: Points generated by adding noise to two circles and the result of running the k-means algorithm for $k = 2$.

In this case, the k-means algorithm will be unable to separate the two circles because it produces convex clusters, causing it to divide the data along a straight line.

1.2.2 Linkage Clustering

Let us now consider a clustering algorithm that produces a dendrogram as an output. We achieve this by consecutively merging clusters based on some distance criterion [ELLS11, Sec. 4.2.2].

Definition 1.15: Linkage Clustering

Let $X := \{x_1, \dots, x_n\} \subset \mathbb{R}^n$ and $\mathfrak{d} \colon \mathscr{P}(X) \times \mathscr{P}(X) \to \mathbb{R}_{\geq 0}$[a] a *distance function* between clusters. We start with the partition

$$B_j^{(0)} := \{x_j\}$$

for $j \in \{1, \dots, n\}$. Given a partition

$$B_1^{(k)}, \dots, B_{n-k}^{(k)} \quad \text{of} \quad x_1, \dots, x_n$$

we successively merge the clusters j and ℓ such that

$$\mathfrak{d}(B_j^{(k)}, B_\ell^{(k)})$$

is minimal for $j, \ell \in \{1, \dots, n-k\}$.

[a] $\mathscr{P}(X)$ denotes the power set of X.

The reason this is in fact a hierarchical clustering algorithm, is that we can now construct the dendrogram $\theta : \mathbb{R}_{\geq 0} \to \mathfrak{P}(X)$ given by

$$\theta(t) := \{B_1^{(k)}, \ldots, B_{n-k}^{(k)}\} \in \mathfrak{P}(X)$$

for all $t \in [k, k+1)$ and $k = 0, \ldots, n-1$.

Depending on the distance function $\mathfrak{d}$ we use we call *linkage clustering*

- *single-linkage clustering* if $\mathfrak{d}_{\min}(B_i, B_j) := \min\{d(x,y) : x \in B_i, y \in B_j\}$;
- *complete-linkage clustering* if $\mathfrak{d}_{\max}(B_i, B_j) := \max\{d(x,y) : x \in B_i, y \in B_j\}$;
- *average-linkage clustering* if $\mathfrak{d}_{\text{avg}}(B_i, B_j) := \frac{1}{|B_i||B_j|} \sum_{x \in B_i, y \in B_j} d(x,y)$.

Example 1.16

As an example, consider the data points in Figure 1.4 and their corresponding dendrogram obtained from single-linkage clustering. Here we first merge the points v, w and z, y as they are the closest. Next we add the point x to the cluster $\{v, w\}$ as x is closer to $\{v, w\}$ than $\{y, z\}$. Finally, we merge the remaining two clusters $\{v, w, x\}$ and $\{y, z\}$ to obtain a dendrogram. Notice how this order is fully described on the righthand side.

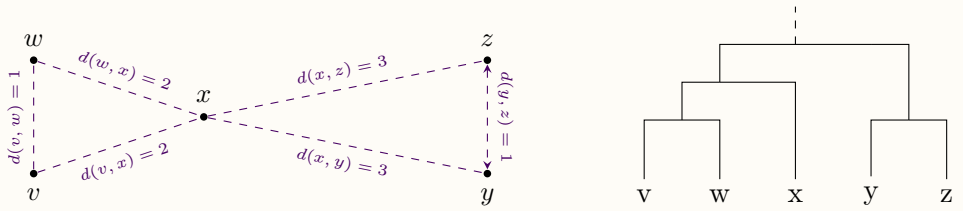


Figure 1.4: Example of single-linkage clustering with the data points $\{v, w, x, y, z\} \subset \mathbb{R}^2$ on the left and the resulting dendrogram on the right.

A downside of this definition of linkage clustering is that the output will always be a binary tree. In particular, if we have three equidistant points it is unclear which should be merged first. We can fix this by introducing a "tie-breaking" rule or by merging more than two clusters at once. We will see more about this once we define *hierarchical clustering functors* in Chapter 3.

Other problems of single-linkage clustering include its insensitivity to density and the tendency to produce long chains as clusters. Complete-linkage and average-linkage clustering, on the other hand, have nice properties with regard to density, but it has been shown that they are not stable under small perturbations of the data [CM10a, Sec. 3.6], [LW67].

1.3 Invariance of Clustering Algorithms

One desirable property of a clustering algorithm is that it remains invariant under certain transformations of the data. The reason this is desirable is that we can model noise in the data as some transformation of the underlying "ground truth". If an algorithm is invariant under such transformations we can hope to recover the ground truth. As we are going to see, it is often not realistic to expect that such an algorithm even exists. But this still motivates the description of clustering algorithms in terms of their invariances.

Example 1.17

Consider a finite set X representing guests at a gathering and a metric d measuring the distance between individual guests. When searching for a (classical) clustering algorithm that finds friendship groups it would make sense to consider algorithms that have the following properties:

1. If we rescale the metric d, i.e., we change the unit with which we measure distance, the algorithm will not change its belief.
2. No matter what the friendship groups are, the guests should be able to arrange themselves in such a way that the algorithm can detect the groups properly.
3. When guests that the algorithm has classified as belonging to the same group move even closer to one another, the algorithm should continue to classify them as part of the same group.

1.3.1 Kleinberg's Impossibility Theorem

It is apparent that as we require more invariants from a clustering algorithm it becomes harder to construct one. There exists an important result in this regard. Kleinberg showed that a generalized version of the properties we considered in Example 1.17 are impossible to satisfy simultaneously [Kle02]. To state Kleinberg's theorem we first need to precisely define these properties.

Definition 1.18: Scale Invariance

We say that a clustering algorithm $\mathfrak{C}$ is *scale invariant* if for every finite metric space (X, d) and $\lambda > 0$ we have

$$\mathfrak{C}(X, d) = \mathfrak{C}(X, \lambda \cdot d),$$

i.e., re-scaling the metric does not affect the clustering algorithm.

Definition 1.19: Richness

A clustering algorithm $\mathfrak{C}$ is *rich* if for every finite set X and every partition P of X there exists a metric d on X such that $\mathfrak{C}(X, d) = P$.

Definition 1.20: Consistency

Let $\mathfrak{C}$ be a clustering algorithm. We say that it is *consistent* if for every finite metric space (X, d) and every metric d' on X such that

- $d'(x, y) \leq d(x, y)$ if x, y are in the same part of $\mathfrak{C}(X, d)$;
- $d'(x, y) \geq d(x, y)$ if x, y are in different parts of $\mathfrak{C}(X, d)$;

we have $\mathfrak{C}(X, d') = \mathfrak{C}(X, d)$.

Kleinberg showed that in combination these properties are impossible to satisfy.

Theorem 1.21: Kleinberg [Kle02, Thm. 2.1]

There exists no clustering algorithm that is scale invariant, rich and consistent at the same time.

Furthermore, if we drop any of the requirements we can find clustering algorithms satisfying the remaining two properties. As an example, k-means clustering from Definition 1.11 is clearly not rich since we restrict ourselves to partitions with k or fewer[1] parts. It is, however, scale invariant.

[1] Depending on the implementation it is possible that k-means produces partitions with less than k parts.

Chapter 2

Categories and Functors

> *"Category theory starts with the observation that many properties of mathematical systems can be unified and simplified by a presentation with diagrams of arrows"* Saunders Mac Lane, 1978 [ML78, p. 1].

Category theory is a branch of mathematics developed to unify common methods and structures used in different areas of mathematics. In this thesis, it will serve as a way to reason about concepts that we define in later chapters. For this chapter we follow books by Roman [Rom17] and Leinster [Lei14].

2.1 Categories

In mathematics, it is common to study maps between objects, e.g.: continuous functions between topological spaces, linear maps between vector spaces, etc. Categories create a framework for this type of mathematical structure.

Definition 2.1: Category [Rom17, Sec. 1.2]

A category $\mathcal{C}$ consists of

1. **Objects**: A class[a] $\mathrm{Ob}(\mathcal{C})$.
2. **Morphisms**: For every $A, B \in \mathrm{Ob}(\mathcal{C})$ we have a set $\mathrm{Mor}_{\mathcal{C}}(A, B)$. An element $f \in \mathrm{Mor}_{\mathcal{C}}(A, B)$ is called a morphism from A to B, and we often write
$$f \colon A \to B,$$
even though f is not always a map. Furthermore, $\mathrm{Mor}_{\mathcal{C}}(A, B)$ and $\mathrm{Mor}_{\mathcal{C}}(C, D)$ are disjoint unless $A = C$ and $B = D$.

3. **Composition**: For every $f \in \mathrm{Mor}_{\mathcal{C}}(A, B)$ and $g \in \mathrm{Mor}_{\mathcal{C}}(B, C)$ there exists a $g \circ f \in \mathrm{Mor}_{\mathcal{C}}(A, C)$ called the composition. The composition operation is associative:
$$f \circ (g \circ h) = (f \circ g) \circ h.$$

4. **Identity**: For all objects $A, B \in \mathrm{Ob}(\mathcal{C})$ there exists a morphism $e_A \in \mathrm{Mor}_C(A, A)$ such that for $f \in \mathrm{Mor}_C(A, B)$ and $g \in \mathrm{Mor}_C(B, A)$ we have
$$f \circ e_A = f \quad \text{and} \quad e_A \circ g = g.$$

[a] A class is a collection of mathematical objects.

One reason why $\mathrm{Ob}(\mathcal{C})$ is not generally a set is that sets require the axiom of regularity, disallowing such notions as "the set of all sets". In particular, classes can be much larger than sets [Rom17, p. 1]. Nonetheless, we will often define the class of objects using the following notation:
$$\mathrm{Ob}(\mathcal{C}) = \{A \mid A \text{ fulfills some condition}\},$$
where it is understood that this need not be a proper set.

In the definition of a category we never required any morphisms other than the identity morphism to exist. This means that we could construct a category with arbitrary objects and only the identity morphism for each object. Such a category would not be very interesting to study. It is, however, common to study categories where certain objects have no morphisms between them.

Interestingly, we can encode many mathematical structures as categories, even some where it is not immediately obvious that they form a category [Rom17, Chap. 1 Ex. 1-7], [Lei14, Sec. 1.1].

The most familiar way of thinking about a category is to see morphisms as maps and the composition as the composition of maps. There are many examples of such categories.

Example 2.2: Category of Sets

Consider the category SET consisting of

- $\mathrm{Ob}(\mathrm{SET}) := \{X \mid X \text{ is a set}\}$,
- $\mathrm{Mor}_{\mathrm{SET}}(X, Y) := \{f \colon X \to Y \mid f \text{ is a map}\}$.

In this case composition is simply the composition of maps and the identity is given by the identity map $\mathrm{id}_A \colon x \in A \mapsto x \in A$.

Instead of taking arbitrary sets and maps between them, we can also study categories where the objects have additional structure and the morphisms preserve that structure.

Example 2.3: Topological Spaces

We can define the category of topological spaces TOP with

- $\mathrm{Ob}(\mathrm{TOP}) := \{(X, \tau) \mid X \text{ is a set with a topology } \tau\}$,
- $\mathrm{Mor}_{\mathrm{TOP}}(X, Y) := \{f \colon X \to Y \mid f \text{ is continuous}\}$

and the usual composition of maps and identity function. The reason this is in fact a category is that the composition of continuous maps is continuous.

Similarly, we can define the category of *based topological* spaces $\mathrm{TOP}_\bullet$. In this case the objects are tuples (X, x_0) with $x_0 \in X$ and morphisms are called *based maps* which are continuous maps $f \colon X \to Y$ such that $f(x_0) = y_0$.

Example 2.4: Vector Spaces

Another very common category is the category of vector spaces over a field $\mathbb{K}$ denoted by $\mathrm{VEC}_{\mathbb{K}}$ where

- $\mathrm{Ob}(\mathrm{VEC}_{\mathbb{K}}) := \{V \mid V \text{ is a vector space over } \mathbb{K}\}$,
- $\mathrm{Mor}_{\mathrm{VEC}_{\mathbb{K}}}(V, W) := \{f \colon V \to W \mid f \text{ is a } \mathbb{K}\text{-linear map}\}$.

This is again a category for similar reasons as in the previous example.

Example 2.5: Groups

We can also consider the category GRP where

1. $\mathrm{Ob}(\mathrm{GRP}) := \{G \mid G \text{ is a group}\}$,
2. $\mathrm{Mor}_{\mathrm{GRP}}(G, H) := \{f \colon G \to H \mid f \text{ is a group homomorphism}\}$.

Instead of considering the category of all groups we could restrict the objects to be abelian groups.

These examples are perhaps the first categories one encounters in mathematics.

We can also consider categories whose morphisms are not maps. In this case it is important to be clear what the composition $\circ$ should be.

Example 2.6: Category of Ordering

Consider the category $\mathrm{ORD}_{\mathbb{R}}$ with

- $\mathrm{Ob}(\mathrm{ORD}_{\mathbb{R}}) := \mathbb{R}$,

- $\mathrm{Mor}_{\mathrm{ORD}_{\mathbb{R}}}(a,b) := \begin{cases} \{(a,b)\} & \text{if } a \geq b \\ \emptyset & \text{else} \end{cases}$.

We define the composition to be $(a,b) \circ (b,c) := (a,c)$ if $a \geq b \geq c$ together with the identity (a,a). Notice how this category encodes the full information of the total ordering $\geq$ on $\mathbb{R}$. In particular, the composition $\circ$ encodes transitivity and the identity encodes reflexivity.

Example 2.7: Group as a Category

Not to be confused with the category of groups GRP, we can also consider a group as a category itself. Let G be a group. Then we can define a corresponding category $\mathcal{G}$ where

- $\mathrm{Ob}(\mathcal{G}) := \{\epsilon\}$ where ϵ is an arbitrary element,
- $\mathrm{Mor}_{\mathcal{G}}(\epsilon, \epsilon) := G$

and the composition is given by $g \circ h = gh$ and identity by $e \in G$.

In the previous two examples it might not be intuitive why these compositions form a category. It is, however, a good exercise to verify that they do.

2.2 Functors

We would now like to study maps between categories. Intuitively, a map like this sends objects from one category to objects in another and morphisms from one category to morphisms in the other in a way that preserves composition and the identity morphism.

Definition 2.8: Functor [Rom17, Sec. 1.3]

Let $\mathcal{A}, \mathcal{B}$ be two categories. A (*covariant*) *functor*, which we denote by

$$\mathfrak{F} \colon \mathcal{A} \to \mathcal{B},$$

consists of the following maps, all denoted with the same symbol:

1. A map between the objects of the two categories:

$$\mathfrak{F} \colon \mathrm{Ob}(\mathcal{A}) \to \mathrm{Ob}(\mathcal{B});$$

2. For all $X, Y \in \mathrm{Ob}(\mathcal{A})$ we have a map between morphisms:

$$\mathfrak{F} \colon \mathrm{Mor}_{\mathcal{A}}(X,Y) \to \mathrm{Mor}_{\mathcal{B}}(\mathfrak{F}(X), \mathfrak{F}(Y)). \tag{2.2.1}$$

A functor must also satisfy

$$\mathfrak{F}(e_X) = e_{\mathfrak{F}(X)}, \tag{2.2.2}$$

where e_X denotes the identity morphism on an object X. Furthermore, for every $f \in \mathrm{Mor}_{\mathcal{A}}(X, Y)$ and $g \in \mathrm{Mor}_{\mathcal{A}}(Y, Z)$

$$\mathfrak{F}(g \circ f) = \mathfrak{F}(g) \circ \mathfrak{F}(f). \tag{2.2.3}$$

The additional requirements (2.2.2) and (2.2.3) are sometimes referred to as *functoriality*.

Remark 2.9: Contravariant Functors

Other than covariant functors we can also consider *contravariant* functors, where the direction of morphisms are "flipped", i.e., instead of (2.2.1) we have

$$\mathfrak{F}\colon \mathrm{Mor}_{\mathcal{A}}(X, Y) \to \mathrm{Mor}_{\mathcal{B}}(\mathfrak{F}(Y), \mathfrak{F}(X))$$

and the composition (2.2.3) is also flipped:

$$\mathfrak{F}(g \circ f) = \mathfrak{F}(f) \circ \mathfrak{F}(g).$$

Similar to functions, we can also compose functors.

Definition & Proposition 2.10: [Rom17, Sec. 1.3.1]

Let $\mathcal{A}, \mathcal{B}, \mathcal{C}$ be three categories and

$$\mathfrak{F}\colon \mathcal{A} \to \mathcal{B}, \quad \mathfrak{G}\colon \mathcal{B} \to \mathcal{C}$$

two functors. Then $\mathfrak{G}$ and $\mathfrak{F}$ can be composed, i.e., we compose the maps between the objects

$$\mathfrak{G}\mathfrak{F}(X) := \mathfrak{G}(\mathfrak{F}(X)) \in \mathrm{Ob}(\mathcal{C}) \quad \forall X \in \mathrm{Ob}(\mathcal{A}),$$

and for all $X, Y \in \mathrm{Ob}(\mathcal{A})$ the maps between the morphisms

$$\mathfrak{G}\mathfrak{F}(f) := \mathfrak{G}(\mathfrak{F}(f)) \in \mathrm{Mor}_{\mathcal{C}}(\mathfrak{G}\mathfrak{F}(X), \mathfrak{G}\mathfrak{F}(Y)) \quad \forall f \in \mathrm{Mor}_{\mathcal{A}}(X, Y).$$

This yields a new functor

$$\mathfrak{G}\mathfrak{F}\colon \mathcal{A} \to \mathcal{C}.$$

Moreover, the composition of functors is associative.

Here we have a few functors from different areas of mathematics, namely linear algebra and algebraic topology.

Example 2.11: The Adjoint Functor [Lei14, Ex. 1.2.12]

Consider the category $\mathrm{VEC}_{\mathbb{K}}$. Then we can define the contravariant functor

$$^*\colon \mathrm{VEC}_{\mathbb{K}} \to \mathrm{VEC}_{\mathbb{K}},$$

which maps a vector space V to its (algebraic) dual V^* and maps a linear map $f\colon V \to W$ to its adjoint map $f^*\colon W^* \to V^*$. In particular, this functor is contravariant since it "flips the direction" of the morphisms.

Example 2.12: Fundamental Group Functor [Hat01, Sec. 1.1.2]

The map that assigns to a based topological space (X, x_0) its fundamental group $\pi_1(X, x_0)$ and maps continuous functions $f\colon (X, x_0) \to (Y, y_0)$ to the group homomorphism

$$\pi_1(X, x_0) \ni [\alpha] \mapsto [f \circ \alpha] \in \pi_1(Y, y_0)$$

is a functor

$$\pi_1\colon \mathrm{TOP}_\bullet \to \mathrm{GRP}.$$

Building on the category $\mathcal{G}$ from Example 2.7, we can now translate the familiar concept of a group homomorphism into a functor between categories.

Example 2.13: Group Homomorphism as a Functor

Let G_1, G_2 be groups, $\phi\colon G_1 \to G_2$ be a group homomorphism and $\mathcal{G}_1, \mathcal{G}_2$ the corresponding categories from Example 2.7. The goal is to construct a corresponding functor

$$\mathfrak{F}_\phi\colon \mathcal{G}_1 \to \mathcal{G}_1.$$

Since both $\mathcal{G}_1$ and $\mathcal{G}_2$ contain a single object ϵ it is clear that $\mathfrak{F}_\phi(\epsilon) = \epsilon$. We still need to define $\mathfrak{F}_\phi$ on the morphisms, for which we simply set

$$\mathfrak{F}_\phi(g) = \phi(g) \in \mathrm{Mor}_{\mathcal{G}_2}(\epsilon, \epsilon) = G_2$$

for every $g \in \mathrm{Mor}_{\mathcal{G}_1}(\epsilon, \epsilon) = G_1$.

It remains to check that this is indeed a functor. It holds that

$$\begin{aligned}\mathfrak{F}_\phi(g \circ h) &= \phi(g \circ h) = \phi(g)\phi(h) \\ &= \phi(g) \circ \phi(h) = \mathfrak{F}_\phi(g) \circ \mathfrak{F}_\phi(h)\end{aligned}$$

for all $g, h \in \mathrm{Mor}_{\mathcal{G}_1}(\epsilon, \epsilon) = G_1$. Since group homomorphisms map the neutral element to the neutral element we get

$$\mathfrak{F}_\phi(e_{G_1}) = \phi(e_{G_1}) = e_{G_2}.$$

The *forgetful functor* will later be used in the definition of clustering functors.

Definition 2.14: [Rom17, Chap. 1 Ex. 10]

Let $\mathcal{C}$ be a category whose objects are sets (potentially with some additional structure) and whose morphisms are certain maps between these sets. Then we can define the *forgetful functor*

$$\alpha_{\mathcal{C}} : \mathcal{C} \to \mathrm{SET}$$

where $\alpha_{\mathcal{C}}$ maps objects to their underlying set and morphisms get mapped to their underlying functions. In this sense, the forgetful functor *forgets* the additional structure of the category.

Example 2.15

Consider the category of topological spaces TOP from Example 2.3. In this case objects $(X, \tau), (Y, \sigma) \in \mathrm{Ob}(\mathrm{TOP})$ are sets equipped with topologies and morphisms $f : (X, \tau) \to (Y, \sigma)$ are continuous maps. In this case the forgetful functor α_{TOP} *forgets* the topology τ and the fact that f is continuous, i.e., it maps (X, τ) to X and f to f.

In particular, we will use the following notation related to the forgetful functor.

Definition 2.16

Let $\mathcal{A}, \mathcal{B}$ be categories with forgetful functors $\alpha_{\mathcal{A}}, \alpha_{\mathcal{B}}$ respectively. We say that a functor $\mathfrak{F} : \mathcal{A} \to \mathcal{B}$ *factorizes the forgetful functors* if

$$\alpha_A = \alpha_B \mathfrak{F}.$$

This means that $\mathfrak{F}$ preserves the underlying set of an object and the underlying map of a morphism.

We now have all the required tools to define clustering functors in the next chapter.

Chapter 3

Clustering Functors

In this chapter we introduce the notion of a *clustering functor*. Ultimately, a clustering functor will be a functor, with certain properties, from a category of finite metric spaces to a category of partitions. Functoriality gives us a nice way of talking about certain invariances described in Section 1.3. This chapter is based on work done by Carlsson and Mémoli [CM10b].

3.1 Finite Metric Spaces

First we need to define the category of "inputs" of clustering functors. As we have seen before that it is natural to think of *data* as a finite metric space. This motivates the following definition.

Definition 3.1: Finite Metric Spaces [CM10b, Sec. 3.2]

We define three categories $\mathcal{M}_{\text{iso}}, \mathcal{M}_{\text{inj}}$ and $\mathcal{M}_{\text{gen}}$, all sharing the same objects:

$$\text{Ob}(\mathcal{M}) := \{(X, d) \mid X \text{ is a finite non-empty set and } d \text{ a metric on } X\}$$

for $\mathcal{M} \in \{\mathcal{M}_{\text{iso}}, \mathcal{M}_{\text{inj}}, \mathcal{M}_{\text{gen}}\}$. The three categories are distinguished by their morphisms. For $A, B \in \text{Ob}(\mathcal{M})$:

- $\text{Mor}_{\mathcal{M}_{\text{gen}}}(A, B)$ consists of distance non-increasing functions $f\colon A \to B$;
- $\text{Mor}_{\mathcal{M}_{\text{inj}}}(A, B)$ consists of distance non-increasing injective functions $f\colon A \to B$;
- $\text{Mor}_{\mathcal{M}_{\text{iso}}}(A, B)$ consists of isometries $f\colon A \to B$.

The composition of morphisms is given by the composition of functions, and the identity morphism is the identity function.

Depending on what kind and the amount of "structure" we would like our clustering algorithms to "preserve", we can choose one of the three categories. By construction, we have the inclusions

$$\mathcal{M}_{\text{iso}} \subset \mathcal{M}_{\text{inj}} \subset \mathcal{M}_{\text{gen}}. \tag{3.1.1}$$

Furthermore, this inclusion yields functors $\mathcal{M}_{\text{iso}} \to \mathcal{M}_{\text{inj}} \to \mathcal{M}_{\text{gen}}$ [Lei14, Def. 1.2.18]. As such, $\mathcal{M}_{\text{iso}}$ is the category with the fewest morphisms. Indeed, between most objects there are no morphisms. All morphisms are between spaces of the same isometry class and have an inverse. In contrast, $\mathcal{M}_{\text{gen}}$ has the most morphisms. In particular, for any metric spaces $A, B \in \text{Ob}(\mathcal{M}_{\text{gen}})$ we have the morphism

$$\text{const}_b \colon A \to B,\ a \mapsto b$$

for some $b \in B$.

3.2 Outputs of Clustering Functors

In this section we define the category of "outputs" of clustering functors. To do this, we recall the definition of a partition and dendrogram introduced in Section 1.1 (see Definitions 1.1 and 1.7). We need additional structure that will allow us to define categories based on partitions or dendrograms respectively.

Definition 3.2

Let $f \colon X \to Y$ be a map. For some partition $P \in \mathfrak{P}(Y)$ we define $f^*(P) \in \mathfrak{P}(X)$ to be the partition such that

$$\forall x, y \in X : x \sim_{f^*(P)} y \iff f(x) \sim_P f(y).$$

We call $f^*(P)$ the *pullback partition of* P *w.r.t.* f.

Example 3.3

Consider $X := \{a, b, c\}$, $Y := \{a, b\}$ and $f \colon X \to Y$ such that

$$f(a) = a \quad \text{and} \quad f(b) = f(c) = b.$$

If we have a partition

$$Q := \{\{a\}, \{b\}\}$$

of Y, then $f^*(Q) = \{\{a\}, \{b, c\}\}$ is the pullback partition of Q w.r.t f. Notice how $f^*(Q)$ consists of the preimages of the parts of Q.

We would like to combine the idea of a pullback partition defined above with the concept of refinement from Definition 1.4.

Definition 3.4

Let $f\colon X \to Y$ be a map $P \in \mathfrak{P}(X)$ and $Q \in \mathfrak{P}(Y)$. Then we write

$$P \preceq_f Q$$

if $P \preceq f^*(Q)$ and say that P *refines* Q *via* f.

Example 3.5

Consider $X := \{a, b, c\}$ and $Y := \{a, b\}$ and $f\colon X \to Y$ as in Example 3.3. If we have the partition

$$Q := \{\{a\}, \{b\}\}$$

of Y then we have already seen that $f^*(Q) = \{\{a\}, \{b, c\}\}$. Thus, for the partition

$$P := \{\{a\}, \{b\}, \{c\}\}$$

of X we have $P \preceq_f Q$.

A useful fact to remember is that if $\mathrm{id}\colon X \to X$ is the identity then

$$P \preceq_{\mathrm{id}} Q \iff P \preceq Q. \tag{3.2.1}$$

With this, we can finally define the categories which will be the "outputs" of clustering functors.

Definition 3.6: Classical Clustering Outputs[CM10b, Def. 3.2]

The category $\mathcal{C}$ of *classical clustering outputs* is defined by

$$\mathrm{Ob}(\mathcal{C}) := \{(X, P) \mid X \text{ finite non-empty and } P \in \mathfrak{P}(X)\}$$

and for all $(X, P), (Y, Q) \in \mathrm{Ob}(\mathcal{C})$ we have the morphisms

$$\mathrm{Mor}_{\mathcal{C}}((X, P), (Y, Q)) := \{f\colon X \to Y \mid P \preceq_f Q\}.$$

In short, we write $(X, P) \preceq_f (Y, Q)$ for such a morphism. As before, the composition is given by composition of maps and the identity is the identity map.

Definition 3.7: Hierarchical Clustering Outputs[CM10b, Def. 3.3]

Similarly, we define the category $\mathcal{H}$ of *hierarchical clustering outputs*, given by

$$\mathrm{Ob}(\mathcal{H}) := \Big\{(X, \theta_X) \;\Big|\; X \text{ finite non-empty} \\ \text{and } \theta_X : \mathbb{R}_{\geq 0} \to \mathfrak{P}(X) \text{ a dendrogram}\Big\}$$

and for all $(X, \theta_X), (Y, \theta_Y) \in \mathrm{Ob}(\mathcal{H})$ we have the morphisms

$$\mathrm{Mor}_{\mathcal{H}}((X, \theta_X), (Y, \theta_Y)) := \{f \colon X \to Y \mid \forall r \in \mathbb{R}_{\geq 0} : \theta_X(r) \preceq_f \theta_Y(r))\}.$$

Again, we write $(X, \theta_X) \preceq_f (Y, \theta_Y)$ for such a morphism, and the composition and identity are defined as before.

Another way of thinking about morphisms in $\mathcal{C}$ (or $\mathcal{H}$) is that given $(X, P), (Y, Q) \in \mathrm{Ob}(\mathcal{C})$ a morphism $f \in \mathrm{Mor}_{\mathcal{C}}((X, P), (Y, Q))$ is simply a map $f \colon X \to Y$ such that

$$\forall x, y \in X : x \sim_P y \implies f(x) \sim_Q f(y).$$

3.3 Clustering Functors

We now have all the tools to define a *clustering functor*.

Definition 3.8: Clustering Functor [CM10b, Sec. 4.1]

Let $\mathcal{M} \in \{\mathcal{M}_{\mathrm{iso}}, \mathcal{M}_{\mathrm{inj}}, \mathcal{M}_{\mathrm{gen}}\}$ and $\mathcal{A} \in \{\mathcal{C}, \mathcal{H}\}$. An *$\mathcal{M}$-functorial clustering functor* (or *$\mathcal{M}$ clustering functor*) is a functor from $\mathcal{M}$ to $\mathcal{A}$

$$\mathfrak{C} : \mathcal{M} \longrightarrow \mathcal{A}$$

such that $\mathfrak{C}$ factorizes the forgetful functors (see Definition 2.16). If $\mathcal{A} = \mathcal{C}$ we say that $\mathfrak{C}$ is *classical* and otherwise if $\mathcal{A} = \mathcal{H}$ we say it is *hierarchical*.

We can express the functoriality of a clustering functor $\mathfrak{C}$ by the following commutative diagram.

$$\begin{array}{ccc} (X, d) & \xrightarrow{f} & (Y, d) \\ \Big\Downarrow \mathfrak{C} & & \Big\Downarrow \mathfrak{C} \\ \mathfrak{C}(X, d) & \xrightarrow{\preceq_f} & \mathfrak{C}(Y, d) \end{array}$$

For a hierarchical clustering functor $\mathfrak{H}$ we will use the simplified notation:

$$\mathfrak{H}(X, d; r) := (X, \theta_X(r)).$$

Moreover, for any $r \in \mathbb{R}_{\geq 0}$ a hierarchical clustering functor $\mathfrak{H}$ induces a classical clustering functor $\mathfrak{H}(\,\cdot\,; r)$.

Remark 3.9

Recall the inclusions (3.1.1) and their induced functors. Given a clustering functor on a larger category, say $\mathfrak{C}\colon \mathcal{M}_{\text{gen}} \to \mathcal{C}$, this immediately induces clustering functors on the smaller categories by pre-composition with the inclusion functors $\mathcal{M}_{\text{iso}} \to \mathcal{M}_{\text{inj}} \to \mathcal{M}_{\text{gen}}$. We will use the same symbol for the induced functors.

With this in mind, it makes sense to think of the categories $\mathcal{M}_{\text{iso}}, \mathcal{M}_{\text{inj}}, \mathcal{M}_{\text{gen}}$ as being different levels of "structure" a clustering functor can "preserve" where $\mathcal{M}_{\text{iso}}$ is the least restrictive and $\mathcal{M}_{\text{gen}}$ the most restrictive.

Remark 3.10

We can extend the partial order $\preceq$ on $\mathfrak{P}(X)$ to a partial order on clustering functors. In particular, if $\mathfrak{C}, \mathfrak{D}\colon \mathcal{M} \to \mathcal{C}$ are classical clustering functors we write $\mathfrak{C} \preceq \mathfrak{D}$ if

$$\forall (X,d) \in \mathrm{Ob}(\mathcal{M}) : \mathfrak{C}(X,d) \preceq \mathfrak{D}(X,d).$$

And in case of hierarchical clustering functors $\mathfrak{C}, \mathfrak{D}\colon \mathcal{M} \to \mathcal{H}$ we write $\mathfrak{C} \preceq \mathfrak{D}$ if

$$\forall (X,d) \in \mathrm{Ob}(\mathcal{M})\, \forall r \in \mathbb{R}_{\geq 0} : \mathfrak{C}(X,d;r) \preceq \mathfrak{D}(X,d;r).$$

3.3.1 The Vietoris-Rips Clustering Functor

A very natural example of a clustering functor is the so-called *Vietoris-Rips clustering functor*. It can be interpreted as both a classical and hierarchical clustering functor. The next two chapters will be dedicated to studying the unique properties of this clustering functor.

Example 3.11: Vietoris-Rips Functor [CM10b, Def. 6.1]

Let $\delta > 0$ and $\mathcal{M} \in \{\mathcal{M}_{\text{iso}}, \mathcal{M}_{\text{inj}}, \mathcal{M}_{\text{gen}}\}$. The *classical Vietoris-Rips clustering functor*

$$\mathfrak{R}_\delta\colon \mathcal{M} \to \mathcal{C}$$

assigns to each metric space $(X,d) \in \mathcal{M}$ the partition (X,P) where $\sim_P$ is the equivalence relation generated[a] by:

$$\forall x, y \in X : d(x,y) \leq \delta \implies x \sim_P y. \tag{3.3.1}$$

Let us now show that $\mathfrak{R}_\delta$ is indeed a clustering functor. For this, it is sufficient to show that $\mathfrak{R}_\delta$ is $\mathcal{M}_{\text{gen}}$-functorial. By Remark 3.9, functoriality over $\mathcal{M}_{\text{iso}}$ and

$\mathcal{M}_{\text{inj}}$ will follow.

Let $(X,d),(Y,d') \in \text{Ob}(\mathcal{M}_{\text{gen}})$ and $(X,P) := \mathfrak{R}_\delta(X)$ as well as $(Y,Q) := \mathfrak{R}_\delta(Y)$. Take any $f \in \text{Mor}_{\mathcal{M}_{\text{gen}}}(X,Y)$, recall that f is distance non-increasing. We have to show that

$$P \preceq_f Q.$$

Indeed, let $x,y \in X$ such that $d(x,y) \leq \delta$. Then, we have

$$d'(f(x),f(y)) \leq d(x,y) \leq \delta$$

and therefore $f(x) \sim_Q f(y)$. By taking the transitive closure we get that $P \preceq_f Q$, and we are done.

[a] *Generated* is to be understood in the sense of Definition 1.2.

Since we take the transitive closure of the condition of (3.3.1) two points $x,y \in X$ are in the same cluster if and only if there is a path of points $x = x_1, x_2, \ldots, x_n = y$ such that $d(x_k, x_{k+1}) \leq \delta$ for all $k = 1,\ldots,n-1$.

Remark 3.12

One could use a proper inequality in (3.3.1). The entire theory we are going to present would still hold, provided we tweak certain definitions accordingly, in particular, Definition 1.7.

By varying the parameter δ we can obtain the hierarchical Vietoris-Rips clustering functor.

Example 3.13: Vietoris-Rips Functor [CM10b, Ex. 7.1]

For $\mathcal{M} \in \{\mathcal{M}_{\text{gen}}, \mathcal{M}_{\text{inj}}, \mathcal{M}_{\text{iso}}\}$ we can define the *hierarchical Vietoris-Rips clustering functor* by

$$\mathfrak{R}\colon \mathcal{M} \to \mathcal{H}$$

where for $(X,d) \in \text{Ob}(\mathcal{M})$ and $\delta > 0$ we set

$$\mathfrak{R}(X,d;\delta) := \mathfrak{R}_\delta(X,d).$$

Notice that since $\mathfrak{R}_\delta$ is a classical clustering functor and $\mathfrak{R}_\delta \preceq \mathfrak{R}_{\delta'}$ for $\delta \leq \delta'$, $\mathfrak{R}$ is indeed a hierarchical clustering functor.

To get an understanding for how the Vietoris-Rips functor works, consider the following concrete example.

Example 3.14

Consider the seven points $\{a, b, c, d, e, f, g\} \subset \mathbb{R}^2$ shown in the Figure 3.1. $\mathfrak{R}_\delta$ creates the clusters $\{a, b, c\}$ and $\{d, e, f, g\}$, drawn in red and blue.

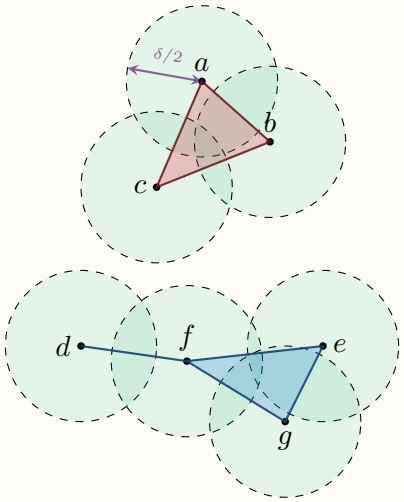


Figure 3.1: Clusters (red and blue) produced by the Vietoris-Rips functor $\mathfrak{R}_\delta$ for on the points $\{a, b, c, d, e, f, g\} \subset \mathbb{R}^2$.

For clarity, we draw the circles with radius $\delta/2$, i.e. two points are in the same cluster if circles around them with radius $\delta/2$ intersect.

Chapter 4

Classification & Uniqueness of Classical Clustering Functors

In this chapter all clustering functors are of the classical type. Our goal in this section is to prove a uniqueness result for the Vietoris-Rips clustering functor $\mathfrak{R}_\delta$ from Example 3.11. We start by presenting a useful way of constructing clustering functors. Namely, we show that any *excessive* clustering functor can be *represented* by a family of metric spaces. This will give an alternative view on the Vietoris-Rips functor. Next we look at the concept of scale invariance as presented in Theorem 1.21 and show that it is too restrictive for our purposes. After this, we are ready to tackle the task of characterizing the Vietoris-Rips functor by defining properties such as *surjectivity*, *spanning*, and *splitting*. Depending on the setting, it will turn out that these conditions are all equivalent and unique to the Vietoris-Rips functor. This chapter is largely based on results presented in [CM10b].

In this section we restrict ourselves to $\mathcal{M} \in \{\mathcal{M}_{\text{inj}}, \mathcal{M}_{\text{gen}}\}$ as it turns out that $\mathcal{M}_{\text{iso}}$ is too general for our purpose (recall Remark 3.9). This is also the setting in which the Vietoris-Rips functor has unique properties.

Remark 4.1: [CM10b, Thm. 6.1]

Let $\mathcal{I}$ denote a collection of representatives of isometry classes of finite metric spaces. For each $X \in \mathcal{I}$ consider the isometry group $\mathrm{Iso}(X)$. This group acts on $\mathfrak{P}(X)$ via $(\phi, P) \mapsto \phi_*(P)$. Let $\Xi_X \subseteq \mathfrak{P}(X)$ denote the set of fixed points of this group action. A clustering functor $\mathfrak{C}\colon \mathcal{M}_{\text{iso}} \to \mathcal{C}$ is uniquely determined by a choice of $P_X \in \Xi_X$ for each $X \in \mathcal{I}$.

On the one hand, it is clear that if $(X, d) \cong (Y, d')$ are isometric then $\mathfrak{C}(X, d) = \mathfrak{C}(Y, d')$. On the other hand, if $(X, d) \ncong (Y, d')$ then there exist no morphisms between X and Y and thus $\mathfrak{C}(X, d)$ and $\mathfrak{C}(Y, d')$ can be independently chosen from Ξ_X and Ξ_Y respectively.

In some sense, $\mathcal{M}_{\text{iso}}$ clustering functors can be thought of as any algorithm that "does not consider any particular ordering" of the data points. This is the reason that algorithms like single linkage clustering (with "tie-breaking") are not even $\mathcal{M}_{\text{iso}}$-functorial (see Section 1.2.2).

4.1 Excessive and Representable Clustering Functors

We will quickly talk about a simple yet, as we will see, very general way of constructing clustering functors.

Definition 4.2: Excessive [CM10b, Def. 6.2]

Let $\mathcal{M} \in \{\mathcal{M}_{\text{inj}}, \mathcal{M}_{\text{gen}}\}$. A clustering functor $\mathfrak{C}\colon \mathcal{M} \to \mathcal{C}$ is called *excessive* if for every $(X,d) \in \text{Ob}(\mathcal{M})$ and $(X,P) = \mathfrak{C}(X,d)$ we have that for every block $X_\alpha \in P$:

$$\mathfrak{C}(X_\alpha, d|_{X_\alpha \times X_\alpha}) = (X_\alpha, \{X_\alpha\})$$

Let Ω be a family of finite non-empty metric spaces. Such Ω's can be used to construct clustering functors. We think of Ω as being a collection of "patterns" and our clustering algorithm as detecting these "patterns".

Definition 4.3: Representable [CM10b, Sec. 6.2]

Let $\mathcal{M} \in \{\mathcal{M}_{\text{inj}}, \mathcal{M}_{\text{gen}}\}$. We define the clustering functor *represented by* Ω as

$$\mathfrak{C}^\Omega\colon \mathcal{M} \to \mathcal{C}$$

where $\mathfrak{C}^\Omega(X,d) = (X,P)$ such that $\sim_P$ is the equivalence relation generated by

$$\forall x,y \in X:\ \exists \omega \in \Omega\, \exists \phi \in \text{Mor}_{\mathcal{M}}(\omega,(X,d)) \text{ s.t. } \{x,y\} \subset \text{im}(\phi) \implies x \sim_P y.$$

Additionally, $\mathfrak{C}$ is said to be *representable* if there exists some Ω such that $\mathfrak{C} = \mathfrak{C}^\Omega$.

Let us define a simple metric space that will come in handy on several occasions.

Definition 4.4

Let $k \in \mathbb{N}$ and $\delta > 0$. Then, we define the *k-simplex* with size δ as the metric space $\Delta_k(\delta) := (X,d)$ where $X = \{1,\dots,k\}$ and the metric is given by

$$d(i,j) := \begin{cases} 0 & \text{if } i = j \\ \delta & \text{otherwise} \end{cases} \qquad \forall i,j \in X.$$

Example 4.5

With this notation we get a new way of defining the Vietoris-Rips functor, i.e.,

$$\mathfrak{R}_\delta = \mathfrak{C}^{\{\Delta_2(\delta)\}}.$$

Indeed, if $(X,d) \in \mathcal{M}_{\text{gen}}$ and $x, y \in X$ then there exists a morphism $\phi\colon \Delta_2(\delta) \to (X,d)$ with $\{x,y\} \subset \text{im}(\phi)$ if and only if $d(x,y) \leq \delta$.

The fact that $\mathfrak{R}_\delta$ is representable is not a coincidence as we will see in the remainder of this section.

Remark 4.6: [CM10b, Rem. 6.3]

Notice that for $(X,d) \in \Omega$ we have $\mathfrak{C}^\Omega(X,d) = (X, \{X\})$.

This leads to an alternative characterization of $\mathfrak{C}^\Omega$.

Proposition 4.7

Let $\mathcal{M} \in \{\mathcal{M}_{\text{inj}}, \mathcal{M}_{\text{gen}}\}$. For any $\mathcal{M}$-clustering functor

$$\mathfrak{C}\colon \mathcal{M} \to \mathcal{C}$$

such that $\mathfrak{C}(X,d)$ is trivial for all $(X,d) \in \Omega$ we have

$$\mathfrak{C}^\Omega \preceq \mathfrak{C}.$$

Proof. Let $(X,d) \in \text{Ob}(\mathcal{M})$ and $(X,P) = \mathfrak{C}^\Omega(X,d)$ and $(X,Q) = \mathfrak{C}(X,d)$. We want to show that $P \preceq Q$.

Let $x, y \in X$ such that the generating condition of P holds, i.e., there exists $\omega \in \Omega$ and $\phi \in \text{Mor}_{\mathcal{M}}(\omega, (X,d))$ such that $\{x,y\} \subset \text{im}(\phi)$.

Since $\mathfrak{C}(\omega)$ is trivial by assumption and by functoriality of $\mathfrak{C}(\omega) \preceq_\phi \mathfrak{C}(X,d)$, we get that $x \sim_Q y$. Taking the transitive closure gives the statement. □

Importantly, $\mathfrak{C}^\Omega$ is the finest clustering functor such that $\mathfrak{C}^\Omega(X,d)$ is trivial for all $(X,d) \in \Omega$. The existence of this minimal clustering functor follows from our initial construction in Definition 4.3.

Theorem 4.8: [CM10b, Thm. 6.2]

Let $\mathcal{M} \in \{\mathcal{M}_{\text{inj}}, \mathcal{M}_{\text{gen}}\}$. A clustering functor $\mathfrak{C}\colon \mathcal{M} \to \mathcal{C}$ is representable if and only if it is excessive.

Proof. First we show that $\mathfrak{C}^\Omega$ is excessive. So let $(X,d) \in \mathrm{Ob}(\mathcal{M})$ and $(X,P) = \mathfrak{C}^\Omega(X,d)$ with $X_\alpha \in P$.

Let $x, y \in X_\alpha$ such that there exists $\omega \in \Omega$ and $\phi \in \mathrm{Mor}_\mathcal{M}(\omega, (X,d))$ such that $\{x,y\} \subset \mathrm{im}(\phi)$. But now since X_α is a block of P we must have that $\mathrm{im}(\phi) \subset X_\alpha$. Therefore, we can restrict ϕ, and we have $\phi|_{X_\alpha} \in \mathrm{Mor}_\mathcal{M}(\omega, (X_\alpha, d|_{X_\alpha \times X_\alpha}))$. By taking the transitive closure, we conclude that $\mathfrak{C}^\Omega(X_\alpha, d|_{X_\alpha \times X_\alpha}) = (X_\alpha, \{X_\alpha\})$ and $\mathfrak{C}^\Omega$ is excessive.

It remains to show that any excessive clustering functor is representable. For this consider

$$\Omega := \{(X_\alpha, d|_{X_\alpha \times X_\alpha}) : X_\alpha \in P \text{ for } (X,P) = \mathfrak{C}(X,d) \text{ and } (X,d) \in \mathrm{Ob}(\mathcal{M})\}.$$

We will show that $\mathfrak{C} = \mathfrak{C}^\Omega$. By Proposition 4.7 and since $\mathfrak{C}$ is by definition trivial for all $(X,d) \in \Omega$ we have that $\mathfrak{C}^\Omega \preceq \mathfrak{C}$.

So it remains to show $\mathfrak{C} \preceq \mathfrak{C}^\Omega$. To this end let $(X,d) \in \mathrm{Ob}(\mathcal{M})$ and $(X,P) = \mathfrak{C}(X,d)$ and $(X,Q) = \mathfrak{C}^\Omega(X,d)$. Assume that $x, y \in X$ are such that $x \sim_P y$. By definition there exists $\omega \subset X$ and $\omega \in \Omega$ such that $\{x,y\} \subset \omega$. Consider the inclusion $\iota : \omega \hookrightarrow X$ which is a morphism in $\mathrm{Mor}_{\mathcal{M}_{\mathrm{inj}}}(\omega, (X,d))$. Thus by definition of $\mathfrak{C}^\Omega$, we have $x \sim_Q y$. □

4.2 Scale Invariant Clustering Functors

Let us quickly talk about scale invariance inspired by Kleinberg's conditions from Theorem 1.21, discussed in [CM10b, Sec. 6.6].

Definition 4.9: Scale Invariance

A clustering functor $\mathfrak{C}\colon \mathcal{M} \to \mathcal{C}$ is called *scale invariant* if for all $\lambda > 0$ and $(X,d) \in \mathrm{Ob}(\mathcal{M})$ we have

$$\mathfrak{C}(X,d) = \mathfrak{C}(X, \lambda \cdot d).$$

We now get two interesting distinct behaviors of scale invariant clustering functors. First we consider the case of $\mathcal{M}_{\mathrm{gen}}$.

Proposition 4.10: [CM10b, Thm. 6.5]

Let $\mathfrak{C}\colon \mathcal{M}_{\mathrm{gen}} \to \mathcal{C}$ be a scale invariant clustering functor then one of the following holds:

1. $\mathfrak{C}(X,d)$ is trivial for all $(X,d) \in \mathrm{Ob}(\mathcal{M}_{\mathrm{gen}})$
2. $\mathfrak{C}(X,d)$ is discrete for all $(X,d) \in \mathrm{Ob}(\mathcal{M}_{\mathrm{gen}})$

Proof. Let $|X| \geq 2$ otherwise the statement is clear. Then for any $x \neq x'$ in X we can find $\delta, \delta' > 0$ and morphisms in $\mathcal{M}_{\text{gen}}$

$$\Delta_2(\delta) \xrightarrow{f} X \xrightarrow{g} \Delta_2(\delta')$$

such that $f(1) = x, f(2) = x', g(x) = 1$ and $g(x') = 2$. By functoriality we get

$$\mathfrak{C}(\Delta_2(\delta)) \preceq_f \mathfrak{C}(X, d) \preceq_g \mathfrak{C}(\Delta_2(\delta')).$$

Recall that by scale invariance we either have that $\mathfrak{C}(\Delta_2(\delta))$ is trivial or discrete for all $\delta > 0$. So since $x \neq x'$ was arbitrary in X, we get that if

- $\mathfrak{C}(\Delta_2(\delta))$ is trivial, then $\mathfrak{C}(X)$ is trivial;
- $\mathfrak{C}(\Delta_2(\delta))$ is discrete, then $\mathfrak{C}(X)$ is discrete.

□

In the case of $\mathcal{M}_{\text{inj}}$ we have a more interesting behavior.

Proposition 4.11: [CM10b, Thm. 6.6]

Let $\mathfrak{C}\colon \mathcal{M}_{\text{inj}} \to \mathcal{C}$ be a scale invariant functor then there exists a $k \in \mathbb{N} \sqcup \{\infty\}$ such that for all $(X, d) \in \text{Ob}(\mathcal{M}_{\text{inj}})$:

- If $|X| > k$ then $\mathfrak{C}(X, d)$ is trivial.
- If $|X| \leq k$ then $\mathfrak{C}(X, d)$ is discrete.

Notice that if $k \in \{0, \infty\}$ then we recover the behavior from $\mathcal{M}_{\text{gen}}$.

Proof. Notice that for $n \leq n'$ there exists a morphism in $\mathcal{M}_{\text{inj}}$

$$\Delta_n(\delta) \longrightarrow \Delta_{n'}(\delta).$$

Therefore, we get that

$$\mathfrak{C}(\Delta_n(\delta)) \preceq \mathfrak{C}(\Delta_{n'}(\delta)).$$

On the other hand, any permutation of $\Delta_n(\delta)$ is also a morphism in $\mathcal{M}_{\text{inj}}$. This gives us that $\mathfrak{C}(\Delta_n(\delta))$ is either discrete or trivial.

So together we get that there exists some $k \in \mathbb{N} \sqcup \{\infty\}$ such that

- $\forall n > k : \mathfrak{C}(\Delta_n(\delta))$ is trivial.
- $\forall n \leq k : \mathfrak{C}(\Delta_n(\delta))$ is discrete.

By scale invariance, this does not depend on δ. We can now repeat the same argument as in the previous proposition to get the statement[1]. □

Following this, we immediately see that scale invariant clustering functors cannot be surjective.

4.3 Surjective Clustering Functors

Another condition in Kleinberg's impossibility theorem (Theorem 1.21) was richness (Definition 1.19). We now extend this notion to clustering functors.

Definition 4.12: Surjective

A classical clustering functor $\mathfrak{C}\colon \mathcal{M} \to \mathcal{C}$ is called *surjective* if for every finite set X and every $P \in \mathfrak{P}(X)$ there exists a metric d on X such that

$$\mathfrak{C}(X, d) = (X, P).$$

Proposition 4.13: [CM10b, Rem. 6.1]

The Vietoris-Rips functor $\mathfrak{R}_\delta$ is surjective.

Proof. Let X be a finite set and P a partition of X. Define the metric d on X by

$$d(x, y) := \begin{cases} 0 & \text{if } x = y \\ \delta & x \neq y \text{ and } x \sim_P y \\ 2\delta & \text{otherwise} \end{cases}.$$

It is straightforward to check that this is indeed a metric and that $\mathfrak{R}_\delta(X, d) = (X, P)$. □

Example 4.14

Let $\delta > 0$ consider $X := \{a, b, c\}$. We would like to check that the Vietoris-Rips functor is indeed able to reach any partition of X. Up to permutation, there are three possible partitions of X:

- $P_1 = \{\{a, b, c\}\}$,
- $P_2 = \{\{a\}, \{b\}, \{c\}\}$,
- $P_3 = \{\{a, b\}, \{c\}\}$.

[1] By using $n = |X|$, we can find morphisms $\Delta_n(\delta) \to X \to \Delta_n(\delta')$ in $\mathcal{M}_{\text{inj}}$.

First, let us consider P_1. For this we can define the metric d_1 on X such that $d_1(i,j) = \delta$ for all $i \neq j$. Then, $\mathfrak{R}_\delta(X, d_1) = (X, P_1)$. For P_2 we can do something similar and use the metric d_2 such that $d_2(i,j) = 2\delta$ for all $i \neq j$, and we get $\mathfrak{R}_\delta(X, d_2) = (X, P_2)$. Finally, for P_3 we can use the metric d_3 with $d_3(a,b) = \delta$ and $d_3(a,c) = d_3(b,c) = 2\delta$. This gives us $\mathfrak{R}_\delta(X, d_3) = (X, P_3)$.

The following property of surjective clustering functors hints that surjectivity and the Vietoris-Rips functor are closely related.

Definition 4.15: Spanning

We say that a clustering functor $\mathfrak{C}\colon \mathcal{M} \to \mathcal{C}$ is *spanning* if for every $(X,d) \in \mathrm{Ob}(\mathcal{M})$ we can find $\lambda_0, \lambda_1 > 0$ such that

1. $\mathfrak{C}(X, \lambda \cdot d)$ is trivial for all $0 < \lambda \leq \lambda_0$,
2. $\mathfrak{C}(X, \lambda \cdot d)$ is discrete for all $\lambda \geq \lambda_1$.

The concept of spanning clustering functors, a term which we introduced here, was previously discussed by Carlsson and Mémoli as part of the assumptions of a theorem [CM10b, Thm. 6.4].

Lemma 4.16

Let $\mathcal{M} \in \{\mathcal{M}_{\mathrm{inj}}, \mathcal{M}_{\mathrm{gen}}\}$ and $\mathfrak{C}\colon \mathcal{M} \to \mathcal{C}$ be a surjective clustering functor. Then $\mathfrak{C}$ is spanning.

For the proof of this lemma we need to introduce the following notation.

Definition 4.17

The *separation* of a metric space (X,d) is given by

$$\mathrm{sep}(X,d) := \mathrm{sep}(X) := \begin{cases} 0 & \text{if } |X| \leq 1 \\ \inf\{d(x,y) \mid x,y \in X, x \neq y\} & \text{otherwise} \end{cases}$$

and the *diameter*

$$\mathrm{diam}(X,d) := \mathrm{diam}(X) := \sup\{d(x,y) \mid x,y \in X\}.$$

Notice that for any metric space (X,d) with $1 < |X| < \infty$ we have

$$0 < \mathrm{sep}(X) \leq \mathrm{diam}(X) < \infty.$$

Proof of Lemma 4.16. Let $(X,d) \in \mathrm{Ob}(\mathcal{M})$, we assume that $|X| > 1$ otherwise the statement follows directly. Since $\mathfrak{C}$ is surjective there exists metrics d_0, d_1 on X such

that $\mathfrak{C}(X, d_0)$ is trivial and $\mathfrak{C}(X, d_1)$ is discrete.

1. We take
$$\lambda_0 := \frac{\operatorname{sep}(X, d_0)}{\operatorname{diam}(X, d)}.$$
Notice that for any $0 < \lambda \leq \lambda_0$ we have that $\operatorname{diam}(X, \lambda \cdot d) \leq \operatorname{sep}(X, d_0)$. Because of this the function
$$\begin{aligned} f : (X, d_0) &\longrightarrow (X, \lambda \cdot d), \\ x &\longmapsto x \end{aligned}$$
is distance non-increasing (and injective), i.e. $f \in \operatorname{Mor}_{\mathcal{M}}((X, d_0), (X, \lambda \cdot d))$. By functoriality of $\mathfrak{C}$, it follows that $\mathfrak{C}(X, d_0) \preceq_f \mathfrak{C}(X, \lambda \cdot d)$. But, since f is the identity on the set X we have $\mathfrak{C}(X, d_0) \preceq \mathfrak{C}(X, \lambda \cdot d)$ by (3.2.1). Therefore, $\mathfrak{C}(X, \lambda \cdot d)$ must be trivial.

2. An analogous argument works if we consider $\lambda \geq \lambda_1$ for
$$\lambda_1 := \frac{\operatorname{diam}(X, d_1)}{\operatorname{sep}(X, d)}$$
and the function
$$\begin{aligned} f : (X, \lambda \cdot d) &\longrightarrow (X, d_1), \\ x &\longmapsto x. \end{aligned}$$
This concludes the proof since we have $\mathfrak{C}(X, \lambda \cdot d) \preceq \mathfrak{C}(X, d_1)$ for the same reason as above.

□

4.4 Splitting Clustering Functors

As observed by [CM10b, Thm. 6.4], a characterizing property of the Vietoris-Rips functor is its behavior on the space $\Delta_2(\lambda)$. In particular, we have that $\mathfrak{R}_\delta(\Delta_2(\lambda))$ is trivial if $\lambda \leq \delta$ and discrete otherwise. This motivates the following definition.

Definition 4.18: Splitting

A clustering functor $\mathfrak{C}\colon \mathcal{M} \to \mathcal{C}$ is called *splitting* at $\delta_0 > 0$ if we have

1. $\mathfrak{C}(\Delta_2(\delta))$ is trivial for all $\delta \leq \delta_0$,
2. $\mathfrak{C}(\Delta_2(\delta))$ is discrete for all $\delta > \delta_0$.

We can now formulate two results that show to what extent splitting is a characterizing property of the Vietoris-Rips functor for $\mathcal{M} \in \{\mathcal{M}_{\text{inj}}, \mathcal{M}_{\text{gen}}\}$ [CM10b, Thm. 6.4].

Proposition 4.19

Let $\mathfrak{C}\colon \mathcal{M}_{\text{inj}} \to \mathcal{C}$ be a clustering functor that splits at $\delta > 0$. Then we have

$$\mathfrak{R}_\delta \preceq \mathfrak{C}.$$

Proposition 4.20

Let $\mathfrak{C}\colon \mathcal{M}_{\text{gen}} \to \mathcal{C}$ be a clustering functor. Then $\mathfrak{C}$ is splitting at $\delta > 0$ if and only if

$$\mathfrak{C} = \mathfrak{R}_\delta.$$

Proof of Proposition 4.19. Let $(X, d) \in \mathrm{Ob}(\mathcal{M}_{\text{inj}})$ with $(X, P) = \mathfrak{C}(X, d)$ and $x \neq y$ in X with $d(x, y) \leq \delta$. Then there is a map $f \in \mathrm{Mor}_{\mathcal{M}_{\text{inj}}}(\Delta_2(\delta), (X, d))$ with $\{x, y\} = \mathrm{im}(f)$. Since $\mathfrak{C}(\Delta_2(\delta))$ is trivial we have by functoriality that $x \sim_P y$.

Taking the transitive closure we get that $\mathfrak{R}_\delta(X, d) \preceq P$ and the statement follows since $(X, d) \in \mathrm{Ob}(\mathcal{M}_{\text{inj}})$ was arbitrary. □

Proof of Proposition 4.20. For the converse we notice that clearly $\mathfrak{R}_\delta$ is splitting at δ.

Let $\mathfrak{C}$ be splitting at δ. In view of the previous proposition it remains to show that $\mathfrak{C} \preceq \mathfrak{R}_\delta$.

Let $(X, d) \in \mathrm{Ob}(\mathcal{M}_{\text{gen}})$ with $(X, P) = \mathfrak{C}(X, d)$ and $(X, R) = \mathfrak{R}_\delta(X, d)$. We show that $P \preceq R$, more precisely, we show that for all $x, y \in X$ we have $x \not\sim_R y \implies x \not\sim_P y$. For this we define

$$\delta_0 := \min\{d(x, y) \mid x, y \in X \text{ and } x \not\sim_R y\}.$$

Let now $x, y \in X$ such that $x \not\sim_R y$. Then we can find a map $f \in \mathrm{Mor}_{\mathcal{M}_{\text{gen}}}((X, d), \Delta_2(\delta_0))$ such that $f(x) \neq f(y)$. One particular way to construct such an f would be to send every $x' \sim_R x$ to one point in $\Delta_2(\delta_0)$ and every $x' \not\sim_R x$ to the other point. By the way we defined δ_0, this map is in fact distance non-increasing.

Since $\delta_0 > \delta$ and $\mathfrak{C}$ is splitting we get $f(x) \not\sim f(y)$ so by functoriality we have $x \not\sim_P y$. □

Example 4.21

In some sense, the two previous propositions are the best we can get. For $\delta > 0$ consider the clustering functor $\bar{\mathfrak{R}}_\delta \colon \mathcal{M}_{\text{inj}} \to \mathcal{C}$ defined by

$$\bar{\mathfrak{R}}_\delta(X,d) := \mathfrak{R}_{\frac{\delta|X|}{2}}(X,d).$$

$\bar{\mathfrak{R}}_\delta$ is splitting at δ.

To see that $\bar{\mathfrak{R}}_\delta$ is indeed a clustering functor consider metric spaces $(X,d),(Y,d') \in \text{Ob}(\mathcal{M}_{\text{inj}})$ and a morphism $f \in \text{Mor}_{\mathcal{M}_{\text{inj}}}((X,d),(Y,d'))$. By injectivity of f we immediately get that $|X| \leq |Y|$. And we have

$$\bar{\mathfrak{R}}_\delta(X,d) = \mathfrak{R}_{\frac{\delta|X|}{2}}(X,d) \overset{(1)}{\preceq} \mathfrak{R}_{\frac{\delta|Y|}{2}}(X,d) \overset{(2)}{\preceq_f} \mathfrak{R}_{\frac{\delta|Y|}{2}}(Y,d') = \bar{\mathfrak{R}}_\delta(Y,d'),$$

where (1) follows from the fact that $|X| \leq |Y|$ and (2) follows from the fact that $\mathfrak{R}_{\delta'}$ is $\mathcal{M}_{\text{inj}}$ functorial for any $\delta' > 0$. Therefore, $\bar{\mathfrak{R}}_\delta$ is an $\mathcal{M}_{\text{inj}}$ clustering functor.

Using a similar argument to the one we used to show that $\mathfrak{R}_{\delta'}$ is surjective, we can also show that $\bar{\mathfrak{R}}_\delta$ is surjective.

4.5 Uniqueness of the Vietoris-Rips Functor

As previously mentioned, the Vietoris-Rips functor is surjective. In this section we present the new result that under certain technical assumptions the Vietoris-Rips is the unique surjective $\mathcal{M}_{\text{gen}}$ clustering functor.

In Chapter 5 we will discuss the implications this has on Kleinberg's theorem (Theorem 1.21).

Definition 4.22: Regularity

A clustering functor $\mathfrak{C} \colon \mathcal{M} \to \mathcal{C}$ is *regular* if for all $(X,d) \in \text{Ob}(\mathcal{M})$ there exists some $\lambda \in (0,1)$ such that

$$\mathfrak{C}(X,d) = \mathfrak{C}(X, \lambda \cdot d).$$

Notice that $\mathfrak{R}_\delta$ is regular. The reason we ask for $\lambda \in (0,1)$ and not $\lambda \in (1,\infty)$ is that this corresponds to the regularity condition 1.8 for dendrograms.

Remark 4.23

It might seem overly restrictive to ask for regularity. However, if we take some

$(X, d) \in \mathrm{Ob}(\mathcal{M})$ and consider the function

$$\lambda \mapsto \mathfrak{C}(X, \lambda \cdot d),$$

which is piecewise constant and takes only finitely many values. Regularity ensures that this function is constant on intervals of the form $(a, b]$ for some $a < b$.
This ensures compatibility with the regularity condition for dendrograms from Definition 1.7.

Assuming regularity and building on Lemma 4.16, we can now finally show that the Vietoris-Rips functor is the only surjective regular $\mathcal{M}_{\mathrm{gen}}$ clustering functor.

Theorem 4.24: Uniqueness Theorem (Classical)

Let $\mathfrak{C}\colon \mathcal{M}_{\mathrm{gen}} \to \mathcal{C}$ then the following are equivalent:

- $\mathfrak{C}$ is surjective and regular,
- $\mathfrak{C} = \mathfrak{R}_\delta$ for some $\delta > 0$.

For the proof of this theorem we use the following lemma.

Lemma 4.25

Let $\mathfrak{C}\colon \mathcal{M} \to \mathcal{C}$ be a surjective regular clustering functor. Then $\mathfrak{C}$ is splitting at some $\delta > 0$.

Proof of Lemma. Let $\lambda \geq \lambda'$, we consider the identity

$$\begin{aligned} \mathrm{id} : (X, \lambda \cdot d) &\longrightarrow (X, \lambda' \cdot d), \\ x &\longmapsto x, \end{aligned}$$

which is distance non-increasing, so by functoriality and by refinement via the identity (3.2.1) we have

$$\mathfrak{C}(X, \lambda \cdot d) \preceq_{\mathrm{id}} \mathfrak{C}(X, \lambda' \cdot d) \implies \mathfrak{C}(X, \lambda \cdot d) \preceq \mathfrak{C}(X, \lambda' \cdot d) \tag{4.5.1}$$

for all $\lambda \geq \lambda'$.

Consider now $(X, d) := \Delta_2(\lambda)$. It is clear that

$$\begin{aligned} \mathbb{R}_{\geq 0} &\longrightarrow \mathfrak{P}(X), \\ \lambda &\longmapsto \mathfrak{C}(\Delta_2(\lambda)) \end{aligned}$$

is piecewise constant and can take at most two values (either discrete or trivial), but by Lemma 4.16 this function takes exactly two values and by (4.5.1) this function is also

monotonically decreasing (with respect to $\leq$ on $\mathbb{R}_{\geq 0}$ and $\preceq$ on $\mathfrak{P}(X)$). So we can find some $\delta_0 > 0$ such that $\mathfrak{C}(\Delta_2(\delta))$ is trivial and $\mathfrak{C}(\Delta_2(\delta'))$ is discrete for all $\delta < \delta_0 < \delta'$.

As for the value at δ_0 we recall Remark 4.23 and conclude that $\mathfrak{C}(\Delta_2(\delta_0))$ is trivial. Therefore, $\mathfrak{C}$ is splitting at δ_0. □

Proof of Theorem. For the first implication, recall that in Proposition 4.13 we have already shown that $\mathfrak{R}_\delta$ is surjective. Moreover, it is also regular.

Assuming that $\mathfrak{C}$ is regular and surjective, from the above lemma we get that $\mathfrak{C}$ is splitting at some $\delta > 0$ so by Proposition 4.20 we have that $\mathfrak{C} = \mathfrak{R}_\delta$. □

Chapter 5

Hierarchical Clustering Functors

In this chapter, we first establish a link between *scaling* hierarchical clustering functors and regular spanning classical clustering functors. With this, we can then tackle the modified Kleinberg conditions presented in [CM10b, Sec. 7.3.1]. We prove that these conditions uniquely characterize the hierarchical Vietoris-Rips clustering functor $\mathfrak{R}$ from Example 3.13. Carlsson and Mémoli already proved a similar result where Kleinberg's assumption of surjectivity was replaced by what we termed splitting [CM10a, Thm. 18].

Similar to classical clustering functors, we also want to define some properties for hierarchical clustering functors.

Definition 5.1: Surjective

A hierarchical clustering functor $\mathfrak{H}$ is said to be *surjective* if for every $(X, \theta) \in \mathrm{Ob}(\mathcal{H})$ there exists a metric d on X such that $\mathfrak{H}(X, d) = (X, \theta)$.

Remark 5.2

Notice that given a surjective hierarchical clustering functor $\mathfrak{H}$, any $t \in \mathbb{R}_{\geq 0}$ induces a surjective classical clustering functor

$$\mathfrak{C}_t(X, d) := \mathfrak{H}(X, d; t).$$

To discuss a similar notion as scale invariance we need to introduce the following functor.

Definition 5.3: Shift Functor [CM10b, Ex. 4.3]

We define the *shift* functor $s_\lambda \colon \mathcal{H} \to \mathcal{H}$ for some $\lambda > 0$. Given $(X, \theta) \in \mathrm{Ob}(\mathcal{H})$ we set

$$s_\lambda(X, \theta) := (X, \theta^\lambda)$$

where $\theta^\lambda(r) := \theta(\frac{r}{\lambda})$ for all $r \in \mathbb{R}_{\geq 0}$. By checking the condition on morphisms in Definition 3.7, we see that this induces a functor.

Instead of scale invariance we can ask that this shift functor behaves nicely with scaling of the metric.

Definition 5.4: Scaling

Let $\mathcal{M} \in \{\mathcal{M}_{\text{iso}}, \mathcal{M}_{\text{inj}}, \mathcal{M}_{\text{gen}}\}$. A hierarchical clustering functor $\mathfrak{H}\colon \mathcal{M} \to \mathcal{H}$ is called *scaling* if for all $(X, d) \in \text{Ob}(\mathcal{M})$ and $\lambda > 0$ we have

$$\mathfrak{H}(X, \lambda \cdot d) = s_\lambda \mathfrak{H}(X, d).$$

In other words, $\mathfrak{H}$ is scaling if for all $r \in \mathbb{R}_{\geq 0}, \lambda > 0$ and $(X, d) \in \text{Ob}(\mathcal{M})$ we have

$$\mathfrak{H}(X, \lambda \cdot d; r) = \mathfrak{H}(X, d; r\lambda^{-1}).$$

Proposition 5.5: [CM10b, Sec. 7.3.1]

The Vietoris-Rips functor $\mathfrak{R}\colon \mathcal{M} \to \mathcal{H}$ is surjective and scaling.

Proof. Let X be a finite set and $\theta : \mathbb{R}_{\geq 0} \to \mathfrak{P}(X)$ be a dendrogram. For any $x, y \in X$ we can define

$$\mathfrak{d}(x, y) := \inf\{r \in \mathbb{R}_{\geq 0} \mid x \sim_{\theta(r)} y\}.$$

Using this we define the metric:

$$d(x, y) := \inf\left\{ \sum_{k=1}^{n-1} \mathfrak{d}(x_k, x_{k+1}) \;\middle|\; \forall n \in \mathbb{N}\ \forall x = x_1, \ldots, x_n = y \right\}.$$

It is easy to verify that $\mathfrak{R}(X, d) = (X, \theta)$. Thus, $\mathfrak{R}$ is surjective. The fact that $\mathfrak{R}$ is scaling follows directly from its definition. □

Extending the Vietoris-Rips functor to a hierarchical clustering functor can be done more generally for any regular spanning classical clustering functor.

Proposition 5.6

Let $\mathcal{M} \in \{\mathcal{M}_{\text{gen}}, \mathcal{M}_{\text{inj}}\}$. Then there exists a one to one correspondence between scaling hierarchical clustering functors and regular spanning[a] classical clustering functors.

More precisely given a regular spanning classical clustering functor $\mathfrak{C}\colon \mathcal{M} \to \mathcal{C}$ there exists a unique scaling hierarchical clustering functor $\mathfrak{H}_\mathfrak{C}\colon \mathcal{M} \to \mathcal{H}$ such that

we have

$$\mathfrak{H}_{\mathfrak{C}}(X, d; 1) = \mathfrak{C}(X, d)$$

for all $(X, d) \in \mathrm{Ob}(\mathcal{M})$. Moreover, any scaling hierarchical clustering functor is of this form.

[a] Recall Definitions 4.22 and 4.15.

Proof. Given a regular spanning classical clustering functor $\mathfrak{C}\colon \mathcal{M} \to \mathcal{C}$ we can define

$$\mathfrak{H}_{\mathfrak{C}}(X, d; r) := \mathfrak{C}(X, r^{-1} \cdot d)$$

for all $(X, d) \in \mathrm{Ob}(\mathcal{M})$ and $r \in \mathbb{R}_{\geq 0}$.

To show that this is indeed a hierarchical clustering functor it remains to show that $\mathfrak{H}_{\mathfrak{C}}(X, d; r)$ is a dendrogram (recall Definition 1.7).

1. Since for $r \leq r'$ the identity map $\mathrm{id} : (X, r^{-1} \cdot d) \to (X, r'^{-1} \cdot d)$ is distance non-increasing and thus by functoriality of $\mathfrak{C}$, we have

$$\mathfrak{H}_{\mathfrak{C}}(X, d; r) = \mathfrak{C}(X, r^{-1} \cdot d) \preceq \mathfrak{C}(X, r'^{-1} d) = \mathfrak{H}_{\mathfrak{C}}(X, d; r').$$

2. Since by assumption $\mathfrak{C}$ is spanning we can find $r, s \in \mathbb{R}_{\geq 0}$ such that

$$\mathfrak{H}_{\mathfrak{C}}(X, d; r) := \mathfrak{C}(X, rd) \quad \text{and} \quad \mathfrak{H}_{\mathfrak{C}}(X, d; s) := \mathfrak{C}(X, sd)$$

   are trivial and discrete respectively.

3. For this we use the regularity of $\mathfrak{C}$ and recall Remark 4.23.

It is clear that $\mathfrak{H}_{\mathfrak{C}}$ is scaling since for all $r \in \mathbb{R}_{\geq 0}, \lambda > 0$ and $(X, d) \in \mathrm{Ob}(\mathcal{M})$ we have

$$\mathfrak{H}_{\mathfrak{C}}(X, \lambda \cdot d; r) = \mathfrak{C}(X, r^{-1}\lambda \cdot d) = \mathfrak{H}_{\mathfrak{C}}(X, d; r\lambda^{-1}).$$

On the other hand, given any scaling hierarchical clustering functor $\mathfrak{H}\colon \mathcal{M} \to \mathcal{H}$ we can take $\mathfrak{C}(X, d) := \mathfrak{H}(X, d; 1)$ which is a regular spanning classical clustering functor such that $\mathfrak{H} = \mathfrak{H}_{\mathfrak{C}}$. □

Example 5.7

Using this notation, we get

$$\mathfrak{R} = \mathfrak{H}_{\mathfrak{R}_1}.$$

Moreover, for $\lambda > 0$ and the shift functor s_λ we get that

$$s_\lambda \mathfrak{R} = s_\lambda \mathfrak{H}_{\mathfrak{R}_1} = \mathfrak{H}_{\mathfrak{R}_{1/\lambda}}.$$

5.1 Kleinberg's Conditions

Carlsson and Mémoli noticed that Kleinberg's impossibility conditions can be interpreted in the context of hierarchical clustering functors [CM10b, Sec. 7.3.1].

Definition 5.8: Modified Kleinberg Conditions [CM10b, Sec. 7.3.1]

We say that a hierarchical clustering $\mathfrak{H}\colon \mathcal{M} \to \mathcal{H}$ functor fulfills the *modified Kleinberg conditions* if all the following holds:

1. $\mathfrak{H}$ is $\mathcal{M}_{\text{gen}}$ functorial (i.e. $\mathcal{M} = \mathcal{M}_{\text{gen}}$),
2. $\mathfrak{H}$ is surjective,
3. $\mathfrak{H}$ is scaling.

We have seen in the above statements that $\mathfrak{R}$ fulfills the modified Kleinberg conditions. Using the uniqueness result of Theorem 4.24, we can now show that the modified Kleinberg conditions uniquely characterize the Vietoris-Rips functor.

Theorem 5.9: Uniqueness Theorem (Hierarchical)

Let $\mathfrak{H}\colon \mathcal{M} \to \mathcal{H}$ be a hierarchical clustering functor that fulfills the modified Kleinberg conditions. Then we have

$$\mathfrak{H} = s_\delta \mathfrak{R}$$

for some $\delta > 0$.

Proof. Let $\mathfrak{H}\colon \mathcal{M} \to \mathcal{H}$ be a hierarchical clustering functor fulfilling the modified Kleinberg conditions. In particular, we have $\mathcal{M} = \mathcal{M}_{\text{gen}}$.

By proposition 5.6, there exists a regular spanning classical functor $\mathfrak{C}\colon \mathcal{M}_{\text{gen}} \to \mathcal{C}$ such that $\mathfrak{H} = \mathfrak{H}_{\mathfrak{C}}$. Since $\mathfrak{H}$ is surjective and by Remark 5.2, we have that $\mathfrak{C}$ is also surjective. Applying Theorem 4.24, we see that $\mathfrak{C} = \mathfrak{R}_\lambda$ for some $\lambda > 0$.

Recall Example 5.7. Using $\delta := 1/\lambda$, we can then see that

$$\mathfrak{H} = \mathfrak{H}_{\mathfrak{R}_\lambda} = s_\delta \mathfrak{H}_{\mathfrak{R}_1} = s_\delta \mathfrak{R}.$$

□

Appendix A

Notation

$\mathbb{R}_{\geq 0}$	the set of non-negative real numbers
(X, d)	a (usually finite non-empty) metric space
$\mathrm{sep}(X)$	the separation of a metric space X
$\mathrm{diam}(X)$	the diameter of a metric space X
$\mathfrak{P}(X)$	the set containing all partitions of X
$P \in \mathfrak{P}(X)$	a partition of a set X
$\sim_P$	equivalence relation corresponding to the partition $P \in \mathfrak{P}(X)$
$\theta : \mathbb{R}_{\geq 0} \to \mathfrak{P}(X)$	a dendrogram on the set X
$\preceq, \preceq_f$	refinement and refinement via f
$\mathcal{M}$	a category of metric spaces $\mathcal{M} \in \{\mathcal{M}_{\mathrm{gen}}, \mathcal{M}_{\mathrm{inj}}, \mathcal{M}_{\mathrm{iso}}\}$
$\mathcal{C}$	the category of outputs of classical clustering functors
$\mathcal{H}$	the category of outputs of hierarchical clustering functors
$\mathfrak{C}$	a classical clustering functor $\mathfrak{C}\colon \mathcal{M} \to \mathcal{C}$
$\mathfrak{R}_\delta$	the Vietoris-Rips classical clustering functor at scale $\delta > 0$
$\mathfrak{C}^\Omega$	the classical clustering functor represented by Ω
$\mathfrak{H}$	a hierarchical clustering functor $\mathfrak{H}\colon \mathcal{M} \to \mathcal{H}$
$\mathfrak{H}_\mathfrak{C}$	the scaling hierarchical clustering functor corresponding to $\mathfrak{C}$
$\mathfrak{R}$	the Vietoris-Rips hierarchical clustering functor

Bibliography

[Car14] Gunnar Carlsson. Topological pattern recognition for point cloud data. *Acta Numerica*, 23:289–368, May 2014.

[CM10a] Gunnar Carlsson and Facundo Mémoli. Characterization, stability and convergence of hierarchical clustering methods. *Journal of Machine Learning Research*, 11(47):1425–1470, 2010.

[CM10b] Gunnar Carlsson and Facundo Memoli. Classifying clustering schemes, 2010.

[ELLS11] Brian S Everitt, Sabine Landau, Morven Leese, and Daniel Stahl. *Cluster Analysis*. Wiley Series in Probability and Statistics. Wiley-Blackwell, Hoboken, NJ, 5 edition, January 2011.

[Hat01] Allen Hatcher. *Algebraic Topology*. Cambridge University Press, Cambridge, England, December 2001.

[Kle02] Jon Kleinberg. An impossibility theorem for clustering. In Suzanna Becker, Sebastian Thrun, and Klaus Obermayer, editors, *Advances in Neural Information Processing Systems*, volume 15. MIT Press, 2002.

[Lei14] Tom Leinster. *Cambridge studies in advanced mathematics: Basic category theory series number 143*. Cambridge University Press, Cambridge, England, July 2014.

[LP97] Rudolf Lidl and Günter Pilz. *Applied abstract algebra*. Undergraduate Texts in Mathematics. Springer, New York, NY, 2 edition, November 1997.

[LW67] G N Lance and W T Williams. A general theory of classificatory sorting strategies: 1. hierarchical systems. *Comput. J.*, 9(4):373–380, February 1967.

[ML78] Saunders Mac Lane. *Categories for the Working Mathematician*. Springer New York, 1978.

[Rom17] Steven Roman. *An introduction to the language of category theory.* Compact Textbooks in Mathematics. Birkhauser, Basel, Switzerland, 1 edition, January 2017.

[SSMÁU21] Rudolf Scitovski, Kristian Sabo, Francisco Martínez-Álvarez, and Šime Ungar. *Cluster Analysis and Applications.* Springer International Publishing, 2021.