LORIA, CNRS, Université de Lorraine @ Nancy, France
HSE University @ Moscow, Russia

Clustering with Stable Pattern Concepts

Egor Dudyrev, Mariia Zueva, Sergei O. Kuznetsov and Amedeo Napoli

19 October 2024, FCA4AI@ECAI24, Santiago de Compostela

Outline

- Motivation
- Concepts as clusters
- Clustering pipeline
    1. Initialising Pattern Structure
    2. Enumerating Cluster Candidates
    3. Enumerating Clustering Candidates
    4. Selecting the Best Clustering
- Conclusions

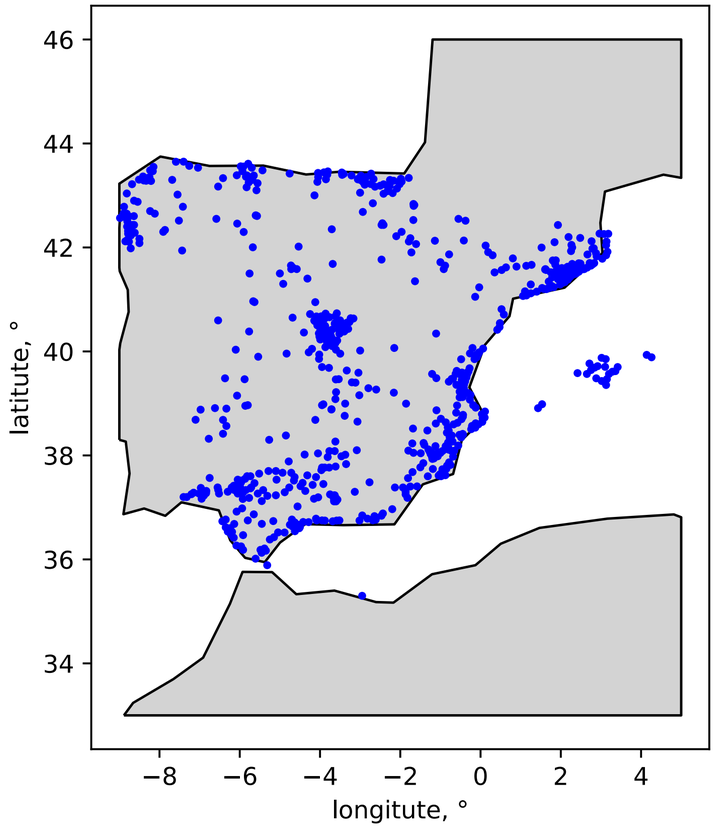
Motivation

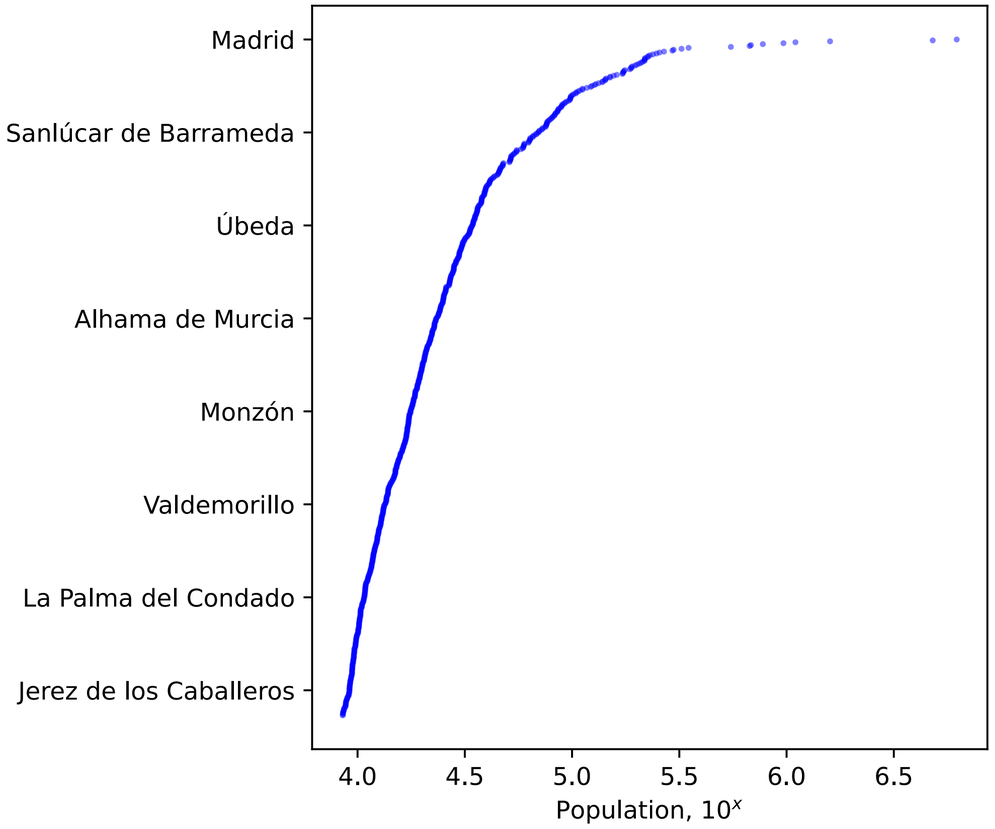
Cities of Spain

Consider a dataset of world cities from https://simplemaps.com/data/world-cities.

For simplicity, let us select 728 cities in the continental Spain with 3 dimensions:

- longitude,
- latitude, and
- population.





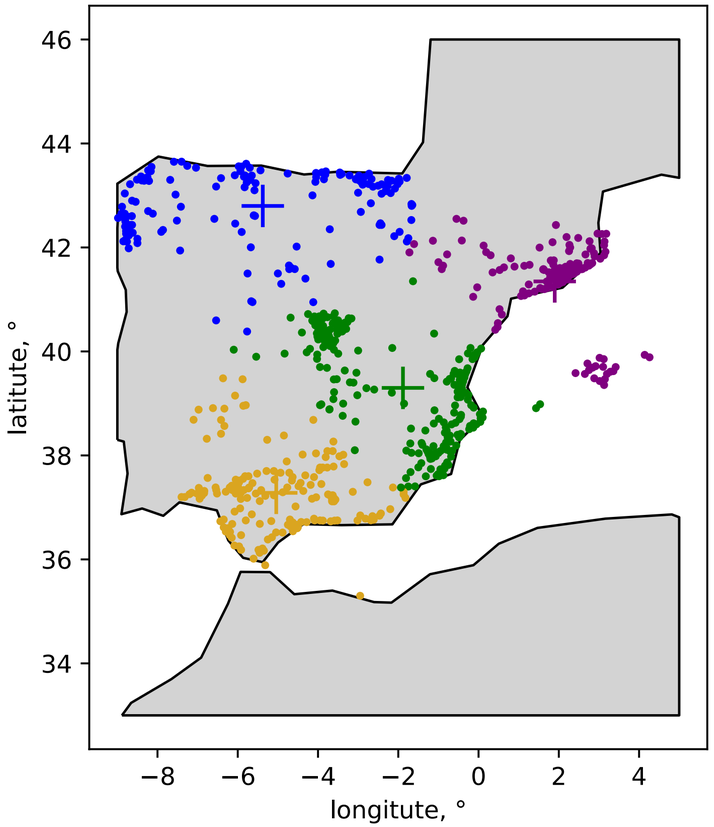
KMeans clustering

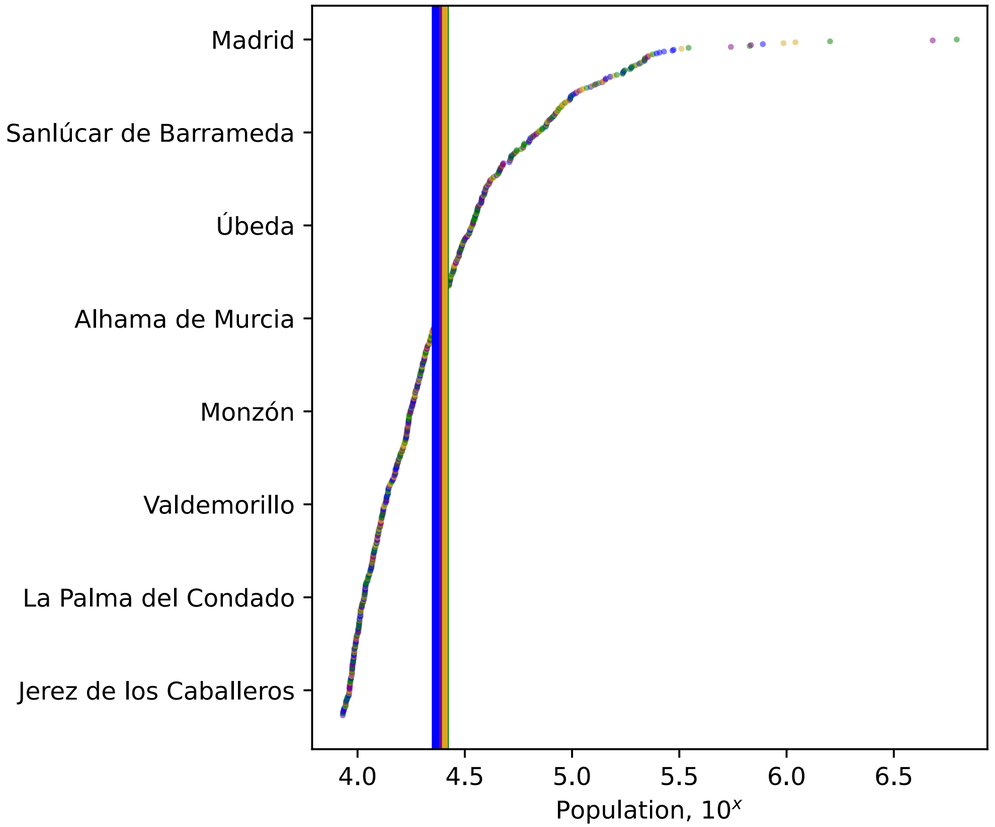
Version 1

Here is the first clustering proposed by KMeans from scikit-learn on top of MinMaxScaled data.

The clustering is purely geographical, and gives no regard to the population.

So, let us change the normalisation to make population important.





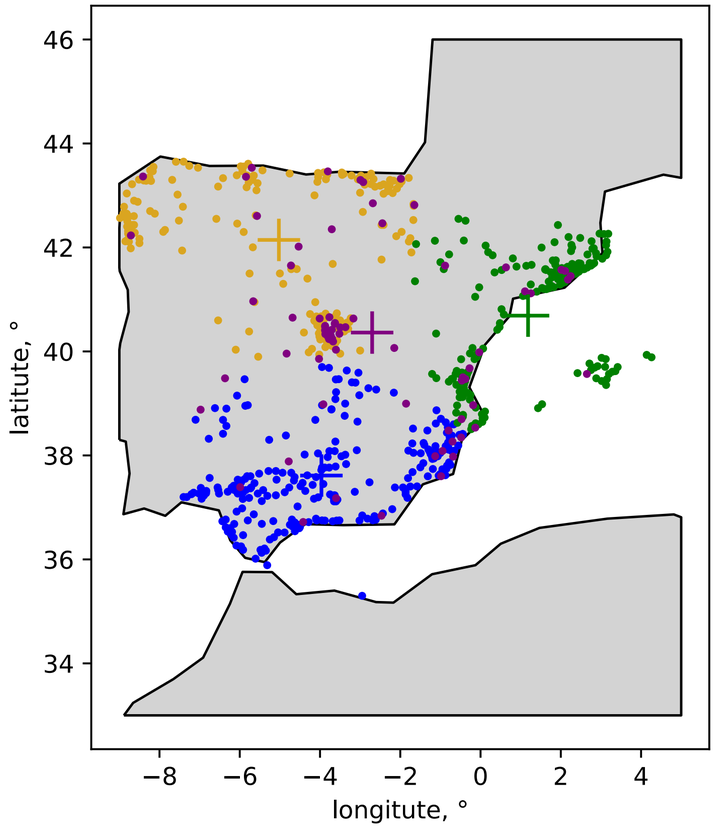
KMeans clustering

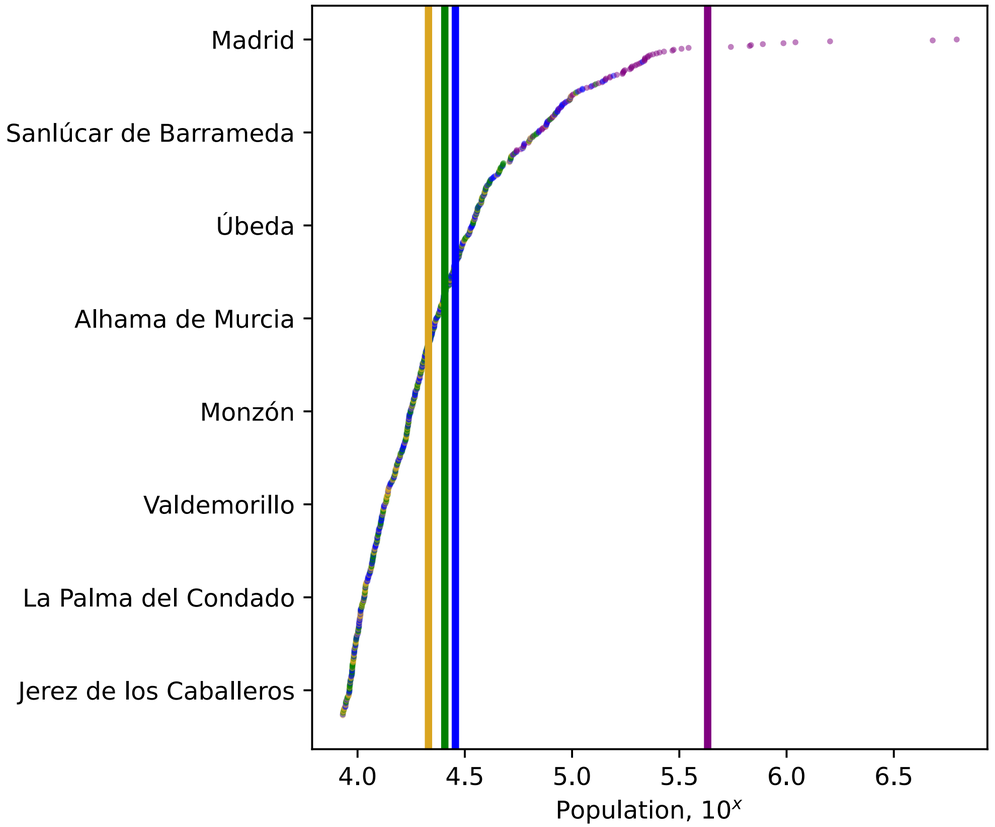
Version 2

Here is the second clustering proposed by KMeans from scikit-learn on top of renormalised data.

We have 4 clusters that we can interpret as follows:

- **Purple cluster**: Big cities
- **Green cluster**: Cities from the East
- **Orange cluster**: Cities from the North-West
- **Blue cluster**: Cities from the South





KMeans clustering

Explanations for clusters

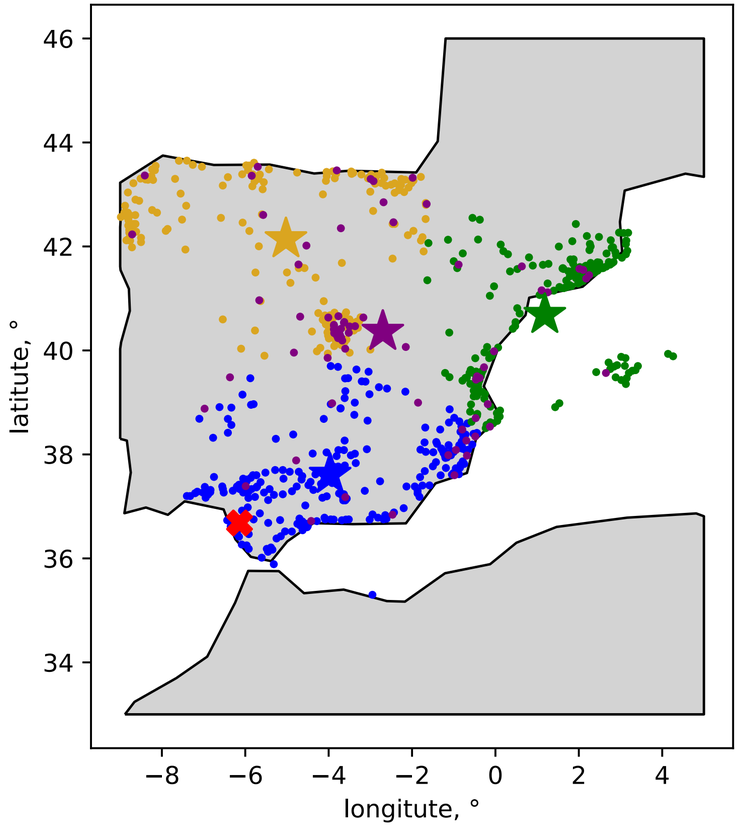
The clustering seems reasonable on a big scale.

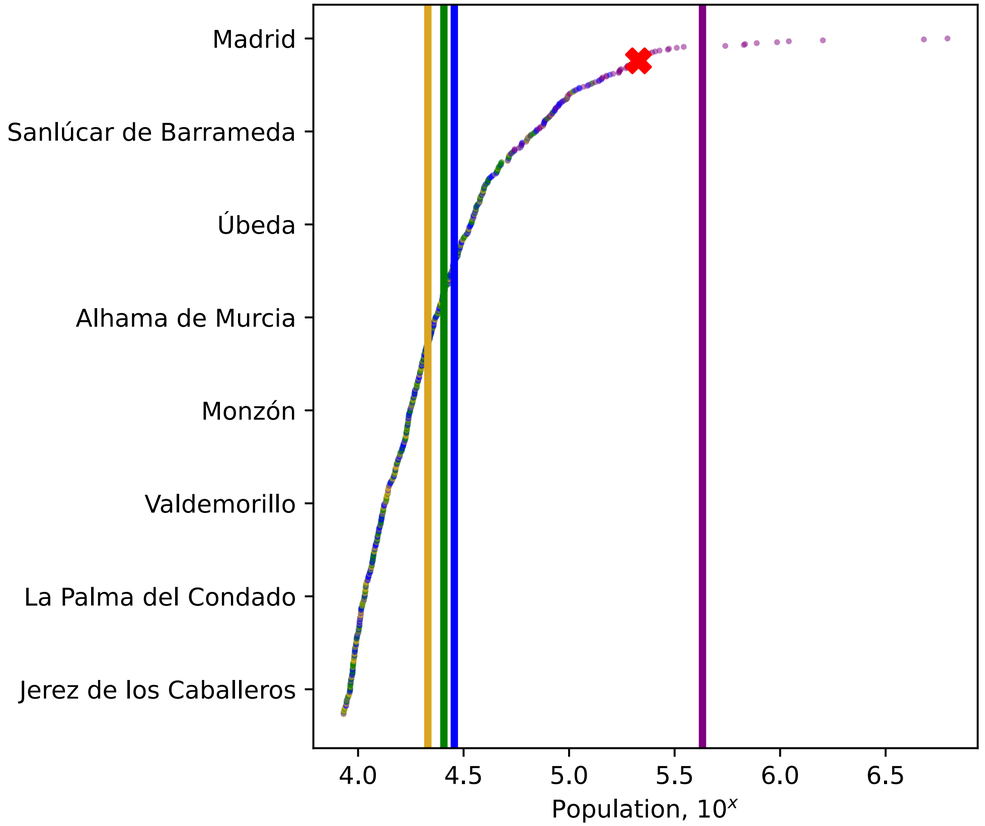
But it raises questions when we look at each city in detail.

For example:

Why is Jerez de la Frontera clustered blue ?

Jerez de la Frontera is marked with a red cross.





Concepts as Clusters

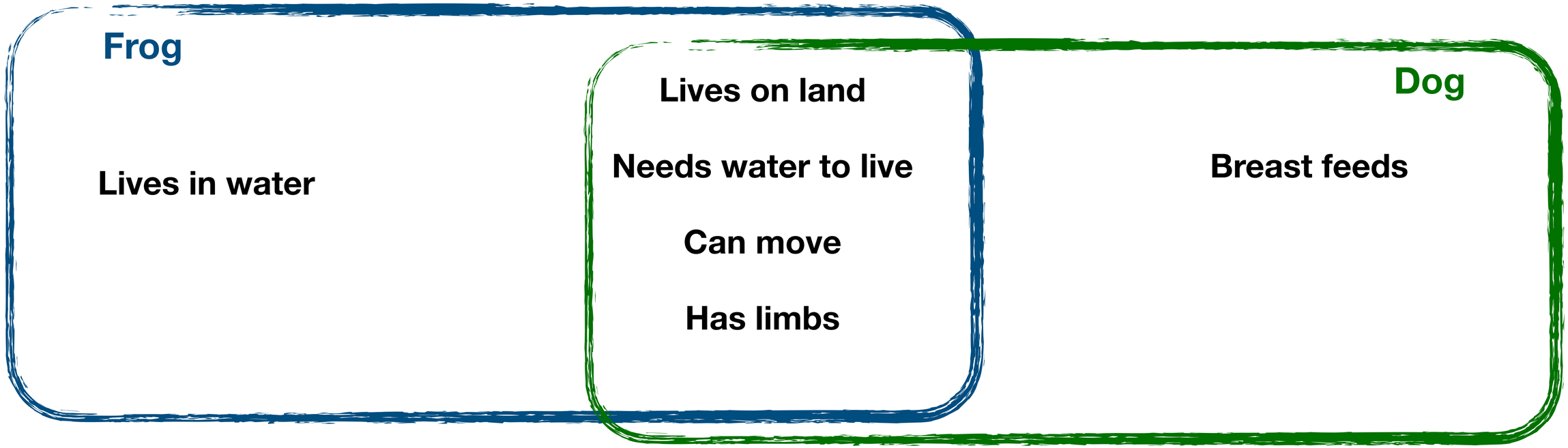
Formal Context

Formal Concept Analysis (FCA) works with a binary dataset, represented with a **Formal Context** (G, M, I):

- Objects G
- Attributes M
- Relations $I \subseteq G \times M$

	Needs water to live	Lives in water	Lives on land	Needs chlorophyll	di-cotyledon	mono-cotyledon	Can move	Has limbs	Breast feeds
Fish leech	✓	✓					✓		
Bream	✓	✓					✓	✓	
Frog	✓	✓	✓				✓	✓	
Dog	✓		✓				✓	✓	✓
Water weeds	✓	✓		✓		✓			
Reed	✓	✓	✓	✓		✓			
Bean	✓		✓	✓	✓				
Corn	✓		✓	✓		✓			

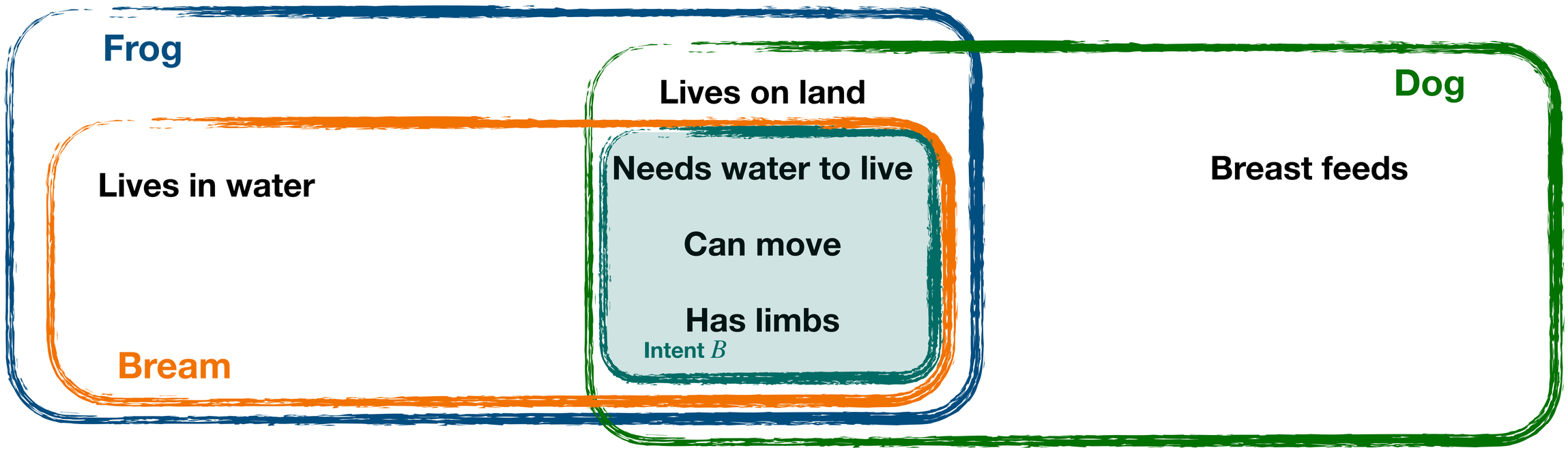
Jaccard Similarity



Within binary data, we can compare the descriptions of two objects with Jaccard similarity coefficient

$$J(\text{Frog, Dog}) = \frac{4}{6} \sim 0.67$$

Formal Concept as a bound

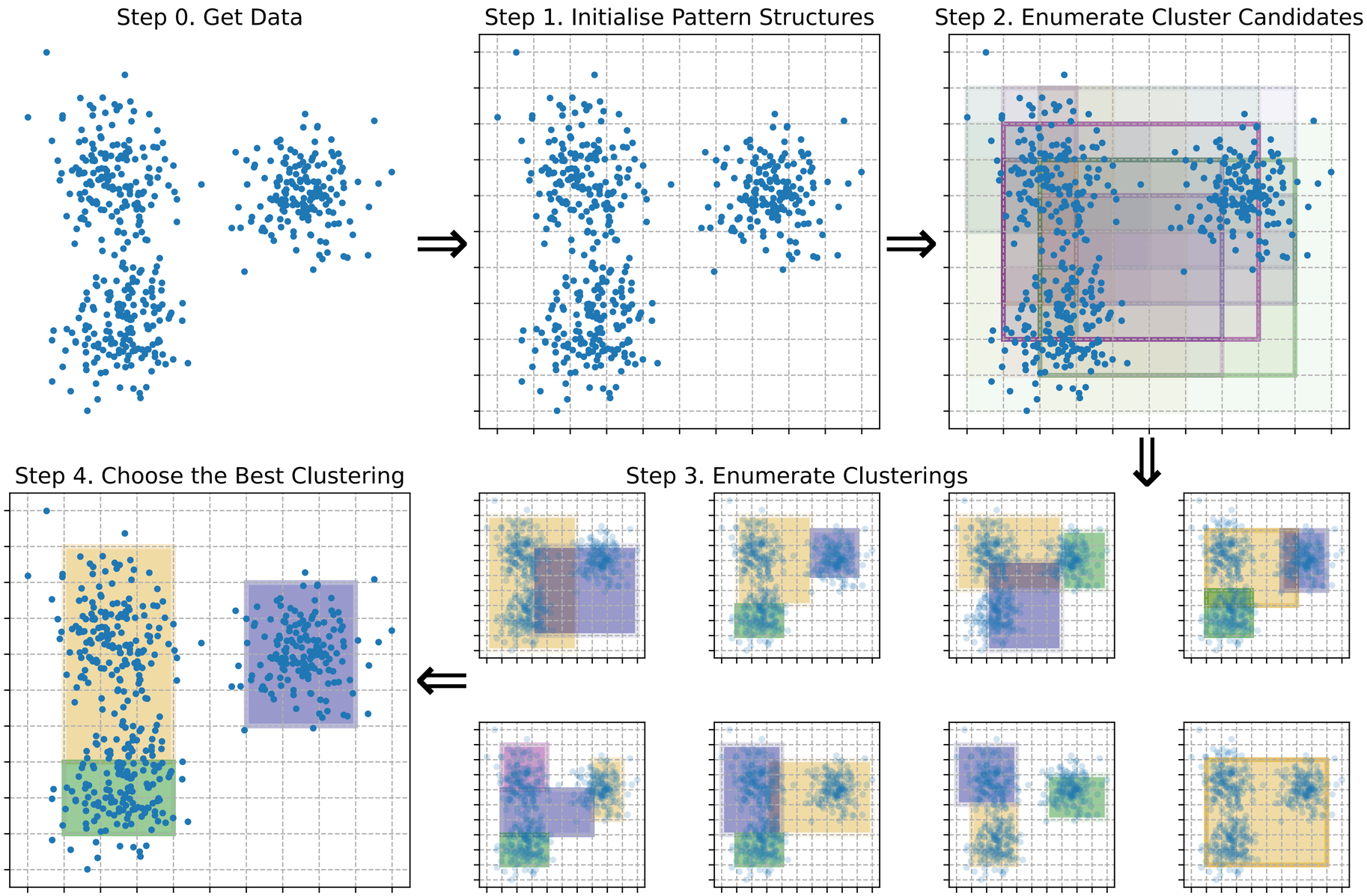


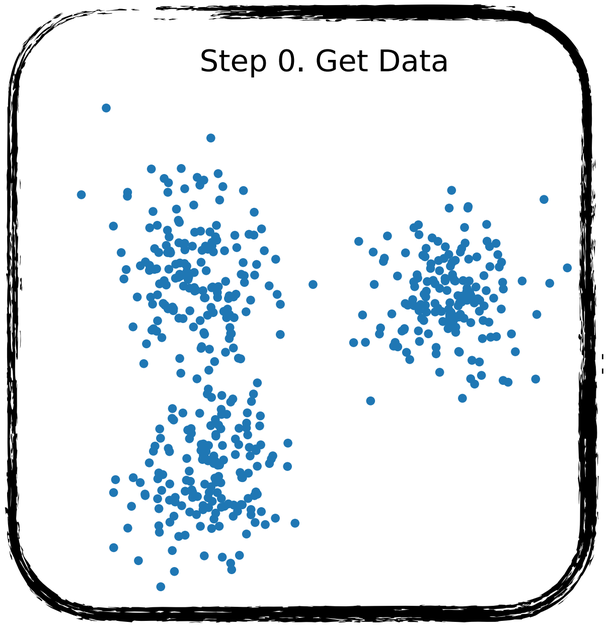
A simple way to estimate a similarity of a group of objects is to see how many attributes they have in common.

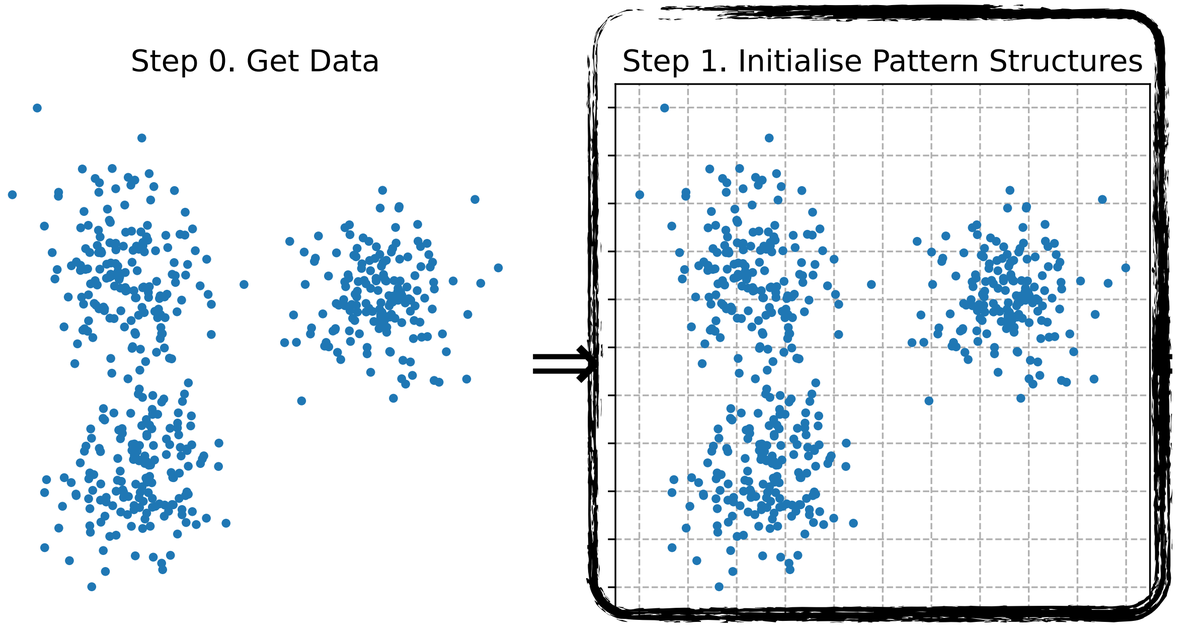
In FCA language, the maximal set of common attribute is the (concept's) **intent** of a subset of objects. The maximal set of objects with a common description is the (concept's) **extent**.

$$\frac{|B|}{|M|} = \frac{3}{9} \sim 0.33 \leq \begin{cases} J(\text{Frog, Dog}) = \frac{4}{6} \sim 0.67 \\ J(\text{Frog, Bream}) = \frac{4}{5} = 0.8 \\ J(\text{Dog, Bream}) = \frac{3}{6} = 0.5 \end{cases}$$

Clustering pipeline


Step 0. Get Data
Step 1. Initialise Pattern Structures
Step 2. Enumerate Cluster Candidates
Step 3. Enumerate Clusterings
Step 4. Choose the Best Clustering


Step 0. Get Data

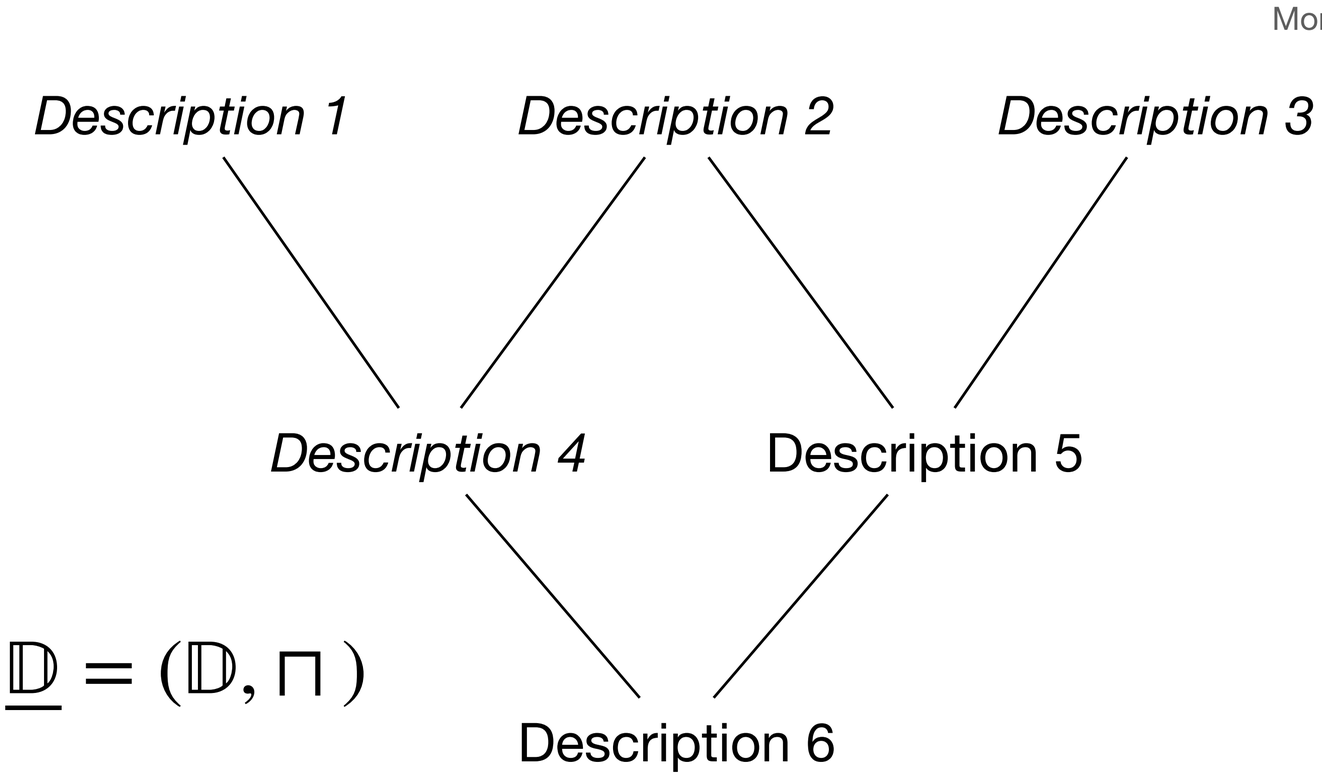

Step 0. Get Data
Step 1. Initialise Pattern Structures

Pattern Structure

General definition

Formal Context (G, M, I)
with description lattice $(2^M, \subseteq)$

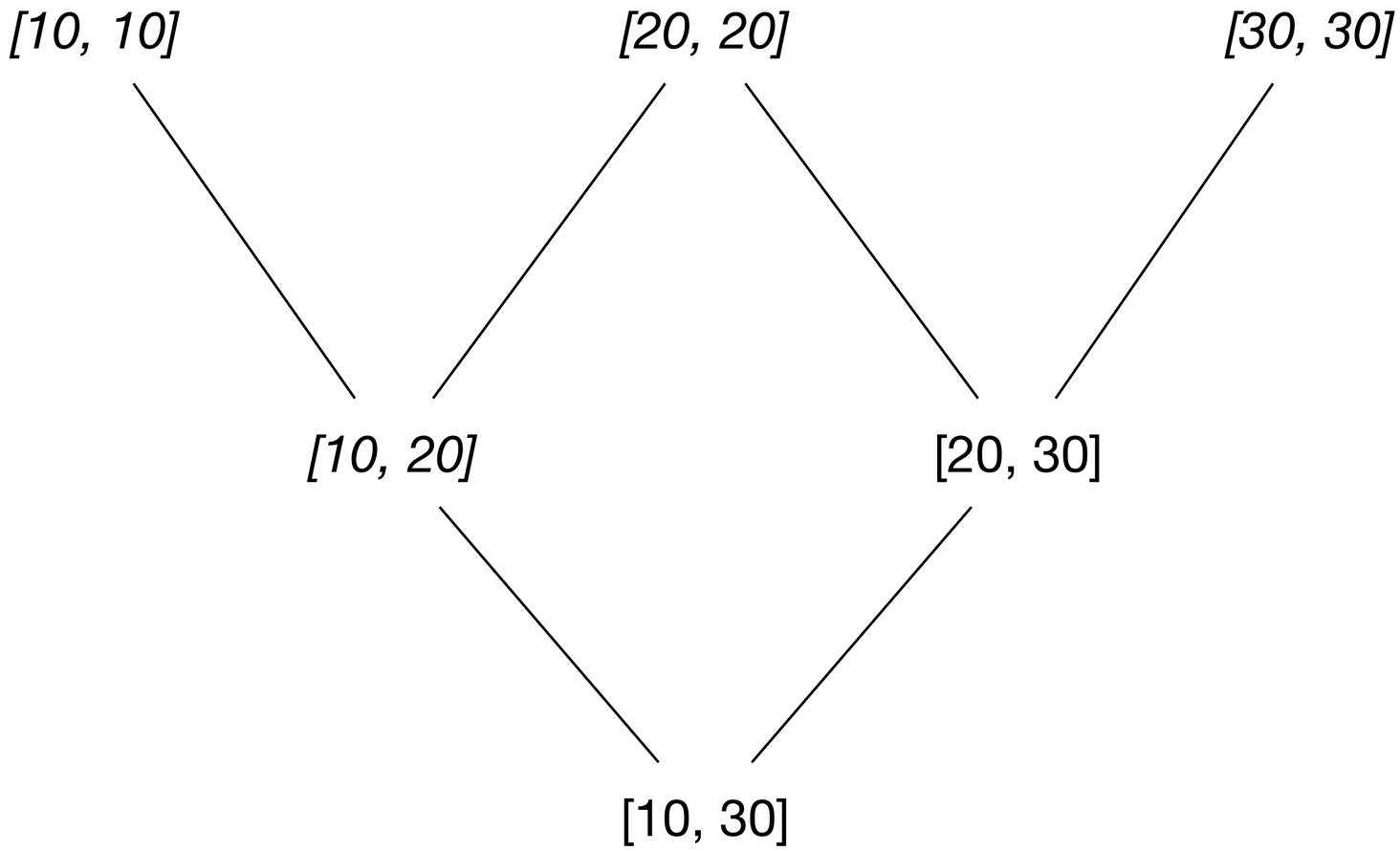
Pattern Structure $(G, \underline{\mathbb{D}}, \delta)$
with description meet semi-lattice $\underline{\mathbb{D}} = (\mathbb{D}, \sqcap)$



So, the set of descriptions $\mathbb{D}$ can be of whatever type (intervals, ngrams, graphs, convex polygons), as long as for any pair of descriptions $D_1, D_2 \in \mathbb{D}$ we have a single maximal common description $D_1 \sqcap D_2 \in \mathbb{D}$.

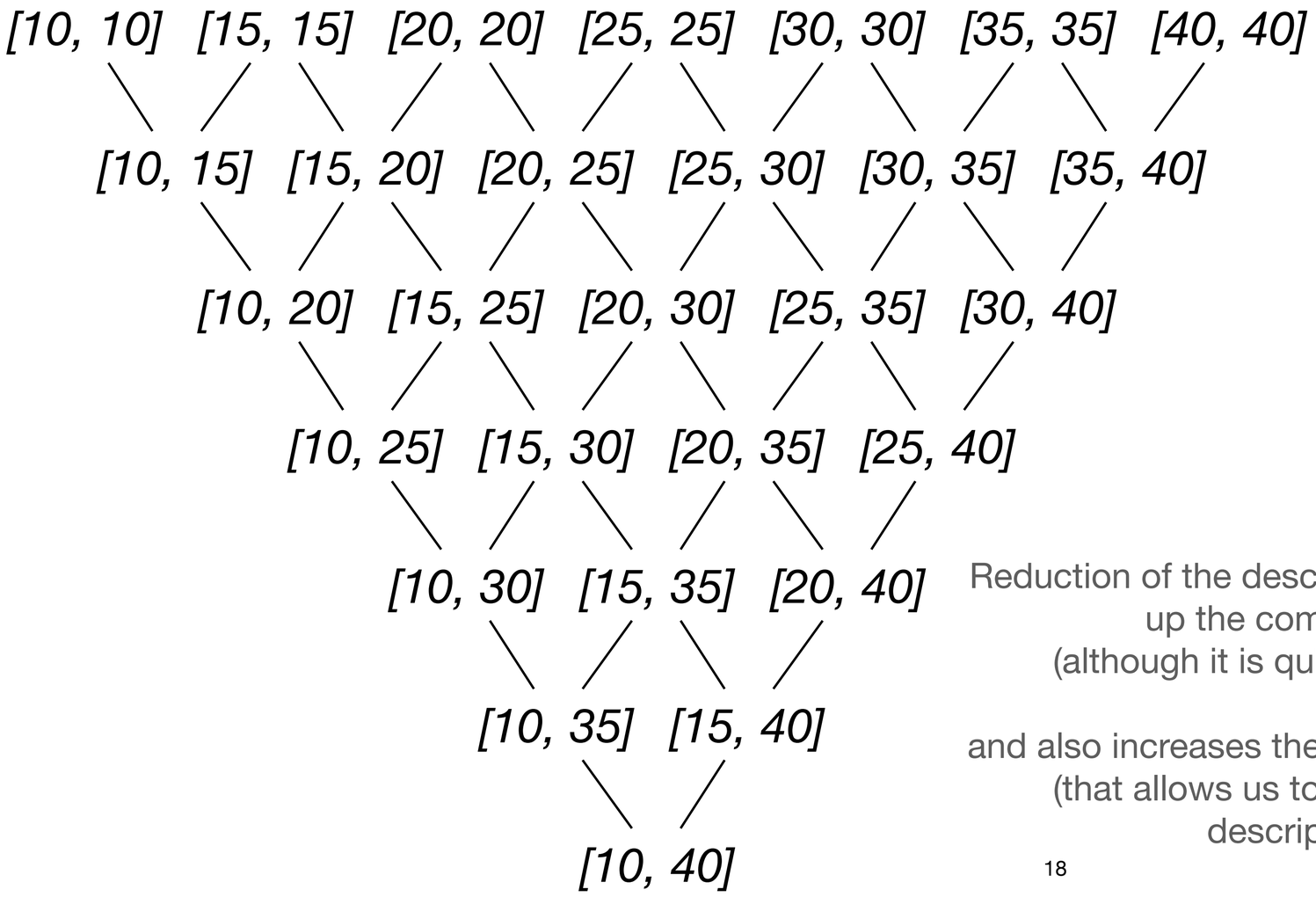
Pattern Structure

Interval PS

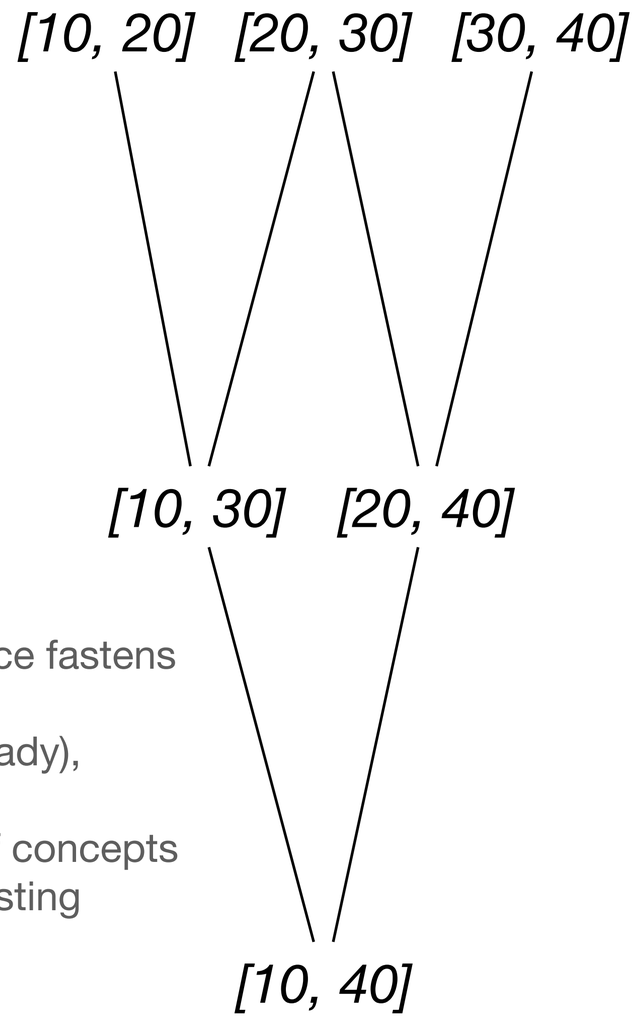


Discretisation

Before

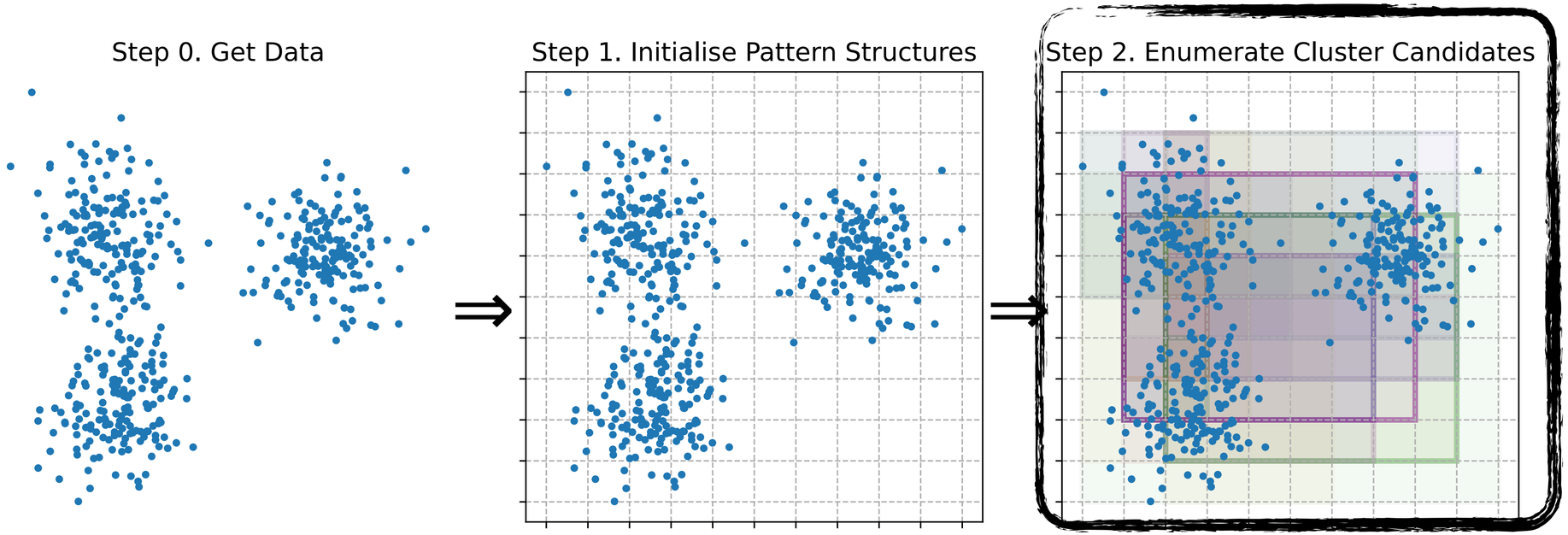


After



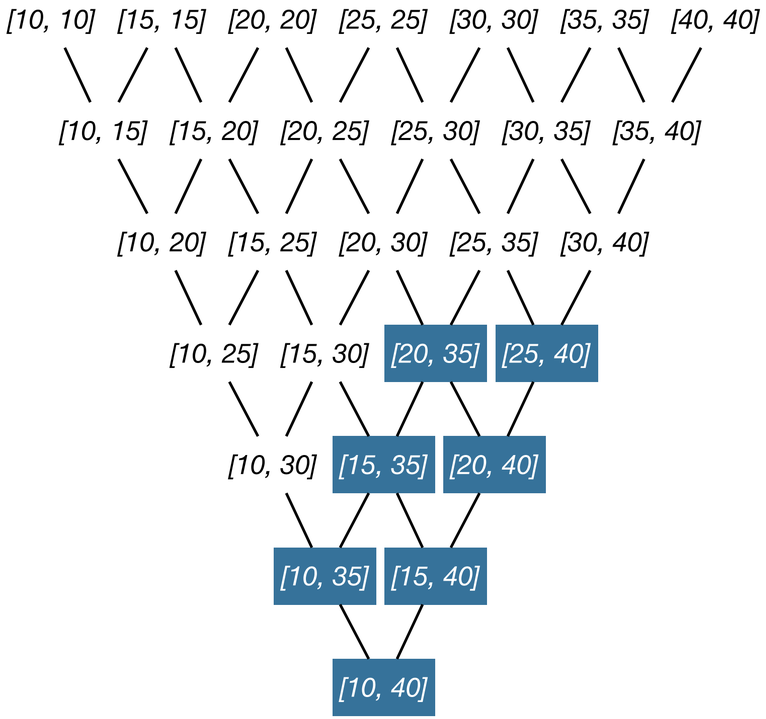
Reduction of the description space fastens up the computations
(although it is quite fast already),

and also increases the stability of concepts
(that allows us to find interesting descriptions)


Step 0. Get Data
Step 1. Initialise Pattern Structures
Step 2. Enumerate Cluster Candidates

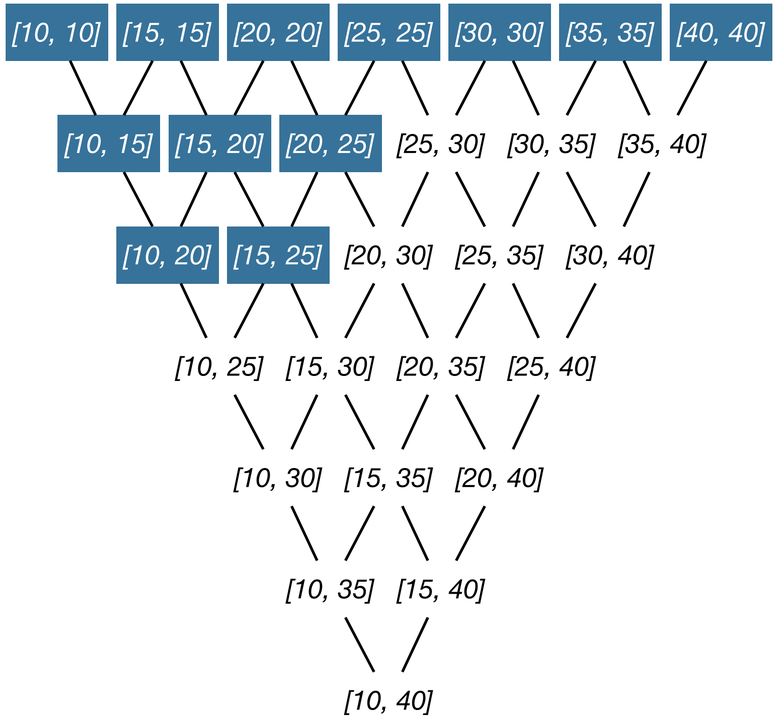
Cluster descriptions candidates

Frequent descriptions



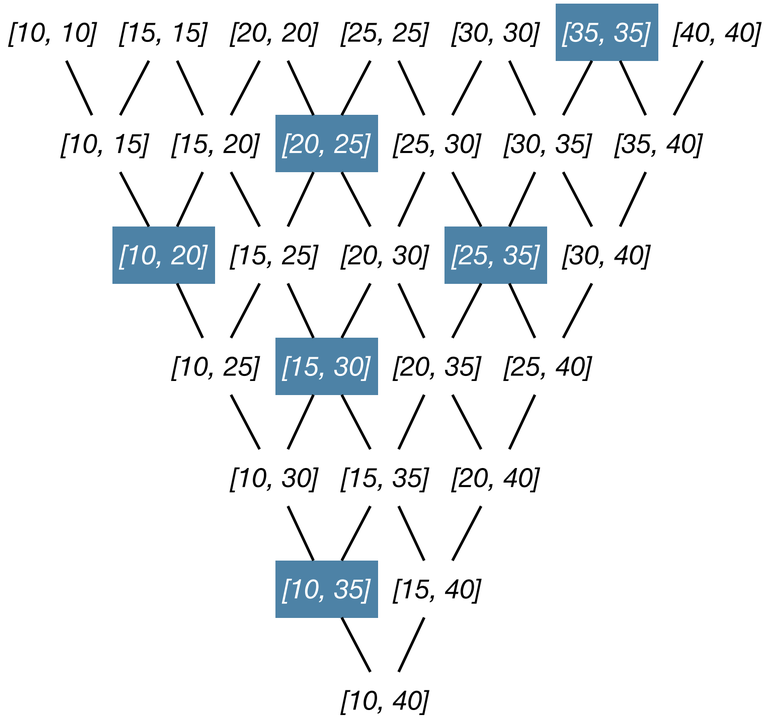
Describe many objects

Rare descriptions



Describe a few objects

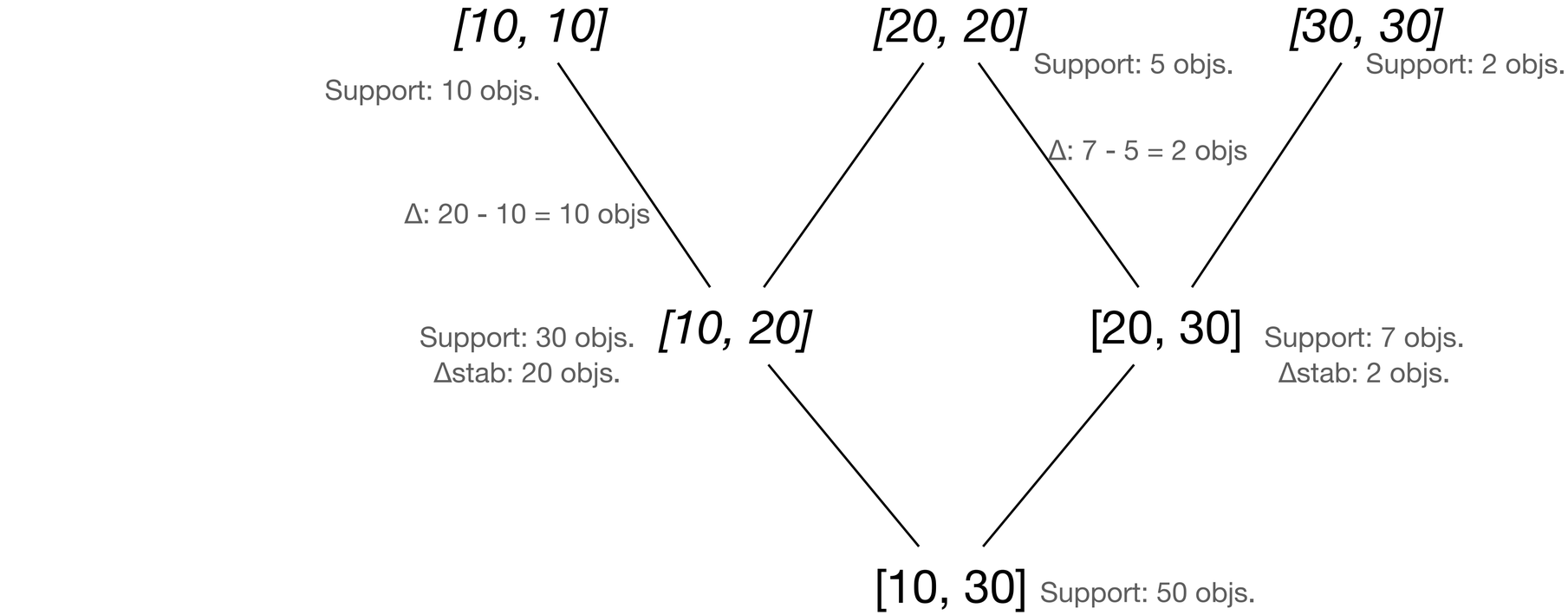
Stable descriptions



Lose many objects when made more precise

Delta-stability

$$\Delta\texttt{stab}(D) = \texttt{supp}(D) - \max_{D_2 \in \mathbb{D}} \texttt{supp}(D_2)$$
$$\text{s.t.} D \sqsubset D_2$$



Finding Stable Concepts

Pattern Structure

	x	y	z
g1	[0, 5]	[20, 30]	[0, 1]
g2	[5, 10]	[20, 30]	[1, 2]
g3	[0, 5]	[30, 40]	[2, 3]
g4	[5, 10]	[30, 40]	[1, 2]
g5	[10, 15]	[20, 30]	[0, 1]
g6	[0, 5]	[10, 20]	[1, 2]
g7	[10, 15]	[20, 30]	[2, 3]
g8	[10, 15]	[30, 40]	[1, 2]

Paspailleur →

Formal Context

	x≥0	x≥5	x≥10	x≤5	x≤10	x≤15	y≥10	y≥20	…
g1	✓			✓	✓	✓	✓	✓	
g2	✓	✓			✓	✓	✓	✓	
g3	✓			✓	✓	✓	✓	✓	
g4	✓	✓			✓	✓	✓	✓	
g5	✓	✓	✓			✓	✓	✓	
g6	✓			✓	✓	✓	✓		
g7	✓	✓	✓			✓	✓	✓	
g8	✓	✓	✓			✓	✓	✓	

Caspailleur + gSofia algorithm →

ΔStable Clusters

so their Δstability is at least Δ objects

{g1, g3, g6}

{g5, g7, g8}

{g1, g2, g5, g7}

{g2, g4, g6, g8}

Paspailleur →

ΔStable Intents

[0, 5], [10, 40], [0, 3]

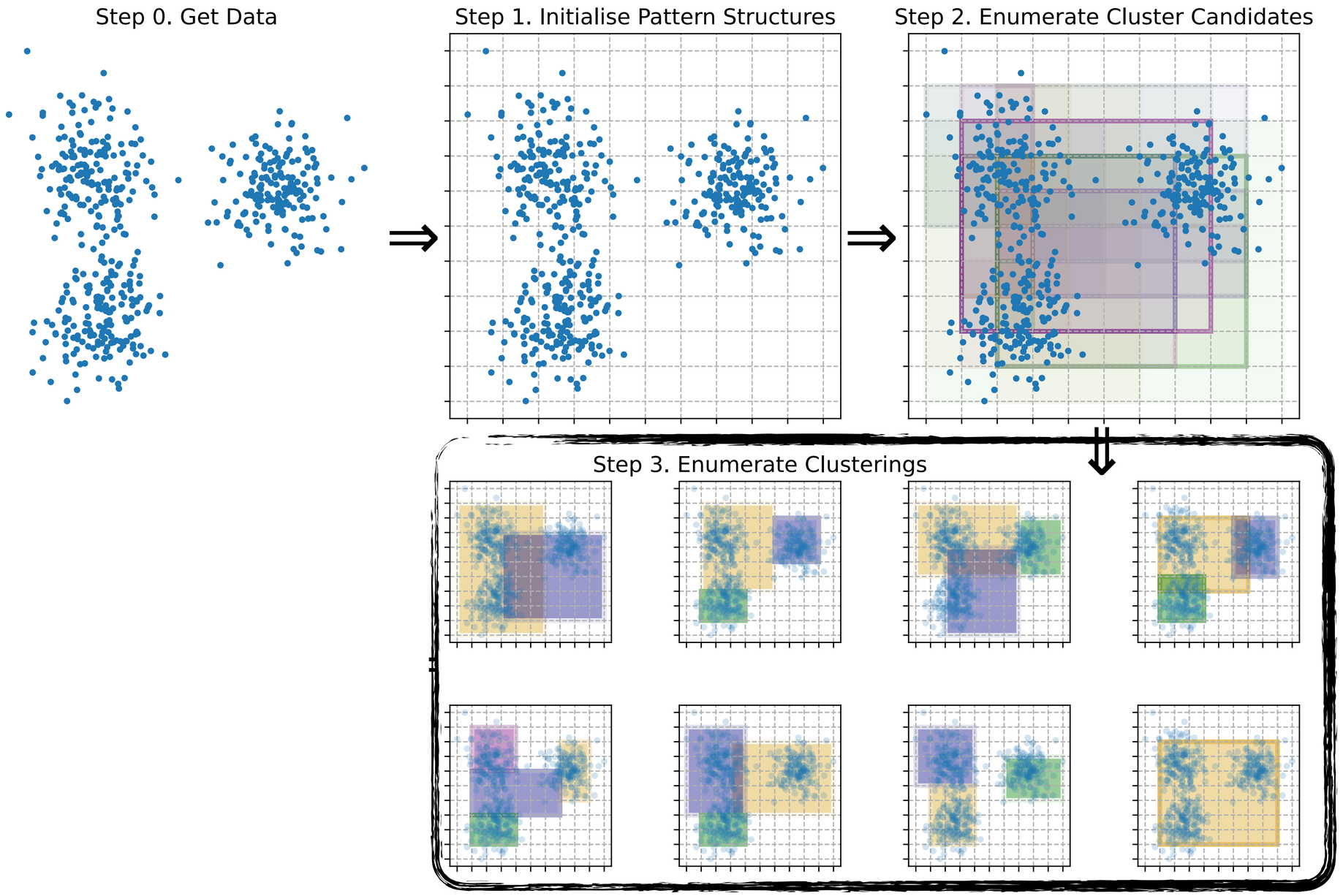
[10, 15], [20, 40], [0, 3]

[0, 15], [20, 30], [0, 3]

[0, 15], [10, 40], [1, 2]

All steps of this pipeline are already implemented in *caspailleur* and *paspailleur* Python packages. So no new code should be added.

NB: I usually set up Δ threshold to be equal to 5% of the total amount of objects $|G|$.


Step 0. Get Data
Step 1. Initialise Pattern Structures
Step 2. Enumerate Cluster Candidates
Step 3. Enumerate Clusterings

Clustering
Definition

A set of **clusters candidates** $\mathbb{C} \subseteq \wp(G)$, where every cluster $C \in \mathbb{C}$ is a closed set of objects, i.e., $C'' = C$.

A **clustering** is any subset of clusters $\mathscr{C} \subseteq \mathbb{C}$.

A clustering $\mathscr{C}$ is called **broad** if it covers more than θ_{cov} objects:

$$\texttt{cov}(\mathscr{C}) = |\bigcup_{C_i \in \mathscr{C}} C_i| > \theta_{cov}$$

A clustering $\mathscr{C}$ is called **minimal broad** clustering if it is a broad clustering, and all its proper subsets are not broad clusterings:

$$\texttt{cov}(\mathscr{C}) > \theta_{cov} \text{ and } \forall \mathscr{C}_2 \subset \mathscr{C}, \texttt{cov}(\mathscr{C}_2) \leq \theta_{cov}.$$

A clustering $\mathscr{C}$ is called θ_{ol}-**non-overlapping** if every pair of clusters overlaps for at most θ_{ol} objects:

$$|C_i \cap C_j| \leq \theta_{ol}, \forall C_i, C_j \in \mathscr{C}.$$

Our task consists in enumerating minimal broad non-overlapping clusterings built from the set of clusters $\mathbb{C}$.

Broad Clusterings and Rare Itemsets

Broad Clustering enumeration task is a complex one. Luckily, we can reuse the algorithms from Rare Itemset Mining.

From De Morgan's laws:

$$a \vee b \vee c \quad \equiv \quad \overline{\overline{a} \wedge \overline{b} \wedge \overline{c}}$$

Writing in pseudomath*:

Broad Clustering

$$\texttt{cov}(a, b, c) > \theta_{cov}$$

$$|a \vee b \vee c| > \theta_{cov} \quad \equiv \quad |\overline{\overline{a} \wedge \overline{b} \wedge \overline{c}}| > \theta_{cov}$$

$$|\overline{a} \wedge \overline{b} \wedge \overline{c}| < |G| - \theta_{cov}$$

Rare Itemset

$$\texttt{supp}(\overline{a}, \overline{b}, \overline{c}) < \theta_{min}, \quad \theta_{min} = |G| - \theta_{cov}$$

*A formal proof is in the paper in Proposition 3.1

So, Broad Clusterings and Rare Itemsets are somewhat "dual" "analogical" notions.

Mining Clusterings

Via Rare Itemset Mining

Cluster context

	C1	C2	C3	C4	C5	C6	C7	C8	C9
Fish leech	✓	✓					✓		
Bream	✓	✓					✓	✓	
Frog	✓	✓	✓				✓	✓	
Dog	✓		✓				✓	✓	✓
Water	✓	✓		✓		✓			
Reed	✓	✓	✓	✓		✓			
Bean	✓		✓	✓	✓				
Corn	✓		✓	✓		✓			

Invert the context

Inverted Cluster context

	C1	C2	C3	C4	C5	C6	C7	C8	C9
Fish leech			✗	✗	✗	✗		✗	✗
Bream			✗	✗	✗	✗			✗
Frog				✗	✗	✗			✗
Dog		✗		✗	✗	✗			
Water			✗		✗		✗	✗	✗
Reed					✗		✗	✗	✗
Bean		✗				✗	✗	✗	✗
Corn		✗			✗		✗	✗	✗

Mine Rare Itemsets

Clusterings

{C4, C7},

{C6, C7},

{C2, C3, C4}

…

The pair (*Bream, C1)* means that object *Bream* is included in cluster $C1$.

This stage can be run with any Minimal Rare Itemset mining software. For example, using Coron system http://coron.loria.fr/

Added restrictions

Additional necessary properties of a good clustering $\mathscr{C} \subseteq G$:

Maximal size $\eta_{\texttt{size}}$:

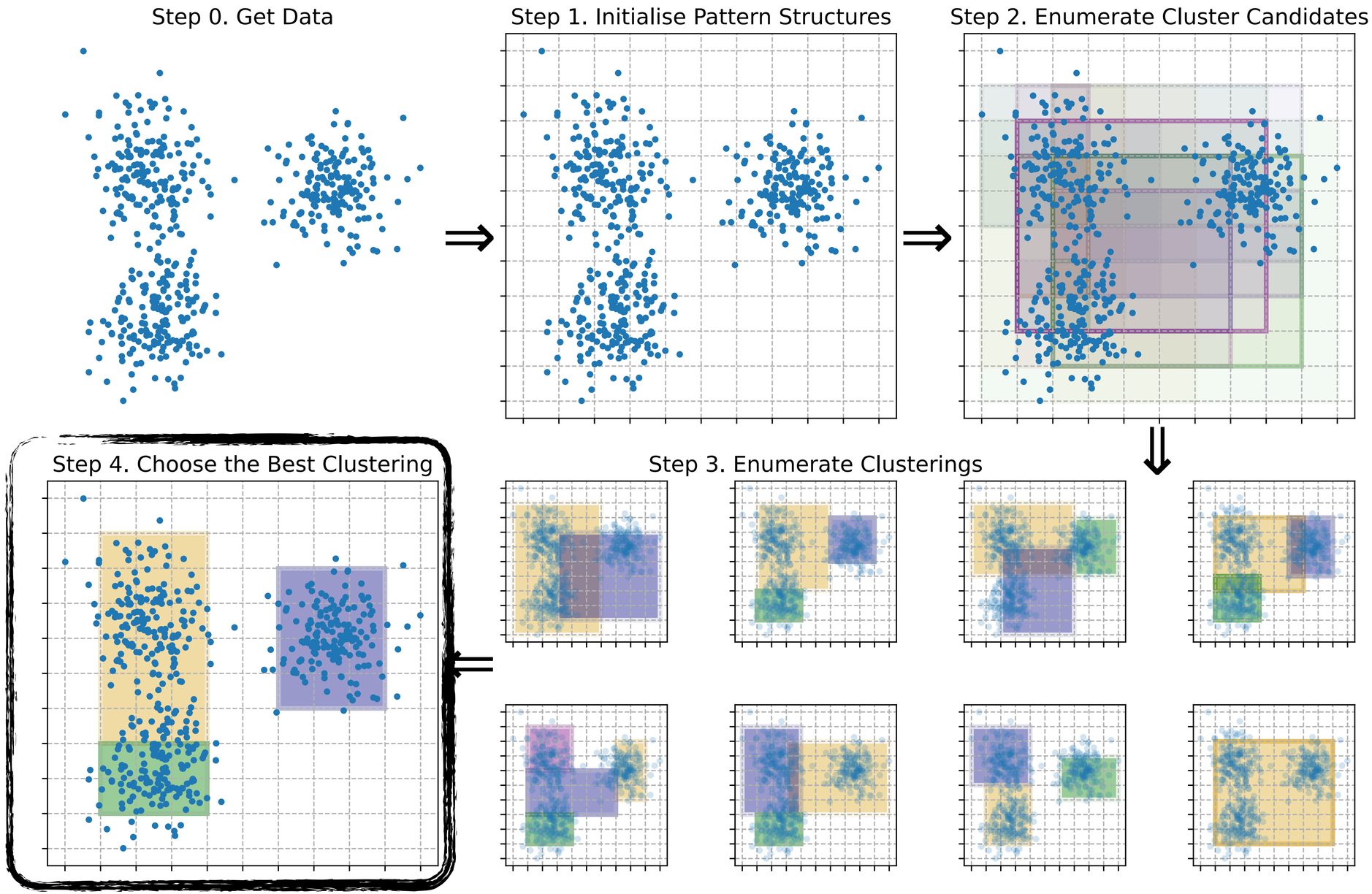
$$|\mathscr{C}| \leq \eta_{\texttt{size}}$$

Minimal added coverage $\eta_{\texttt{cov}}$:

$$\forall C_i \in \mathscr{C},\ \texttt{cov}(\mathscr{C}) - \texttt{cov}(\mathscr{C} \setminus \{C_i\}) \geq \eta_{\texttt{cov}}$$

Final algorithm

1. Take MRG-Exp algorithm from (Szathmary et al, 2007);
2. Replace conjunctions $\wedge$ with disjunctions $\vee$;
3. Replace conditions $< \theta_{min}$ with conditions $> \theta_{cov}$;
4. Skip clusterings that have more than $\eta_{\texttt{size}}$ clusters;
5. Add tests on minimal added coverage $\eta_{\texttt{cov}}$


Step 0. Get Data
Step 1. Initialise Pattern Structures
Step 2. Enumerate Cluster Candidates
Step 3. Enumerate Clusterings
Step 4. Choose the Best Clustering

Clustering measures

Coverage: $\texttt{cov}(\mathscr{C}) = |\bigcup_{C_i \in \mathscr{C}} C_i|$ i.e. how many objects are covered by the clustering

Overlap: $\texttt{ovlap}(\mathscr{C}) = \sum_{C_i, C_j \in \mathscr{C}} |C_i \cap C_j|$ i.e. how much is the pairwise overlap between the clusters

Size: $\texttt{size}(\mathscr{C}) = |\mathscr{C}|$ i.e. the number of clusters in the clustering

Imbalance: $\texttt{imb}(\mathscr{C}) = \texttt{std}\big(\langle |C_1|, |C_2|, \ldots, |C_{|\mathscr{C}|}| \rangle\big)$ i.e. how different are the sizes of clusters in the clustering

Stability: $\texttt{stab}(\mathscr{C}) = \frac{1}{|\mathscr{C}|} \sum_{C_i \in \mathscr{C}} \Delta\texttt{stab}(C_i)$ i.e. how stable are the clusters in the clusterings

Density: $\texttt{density}(\mathscr{C}) = \sum_{(A,D) \in \mathscr{C}} \texttt{density}\big((A,D)\big), \quad \texttt{density}\big((A,D)\big) = |A| / \prod_{j=1}^{n} (r_j - r_i)$ i.e. how dense are the clusters in the clusterings (since every cluster here is a hyperrectangle)

Reward function: weighted normalised average of everything above

The choice of measures and their weights depends on the goals and preferences of a data analyst. The analyst can also invent their own measures if needed.

Experiments

Rectangular clusters

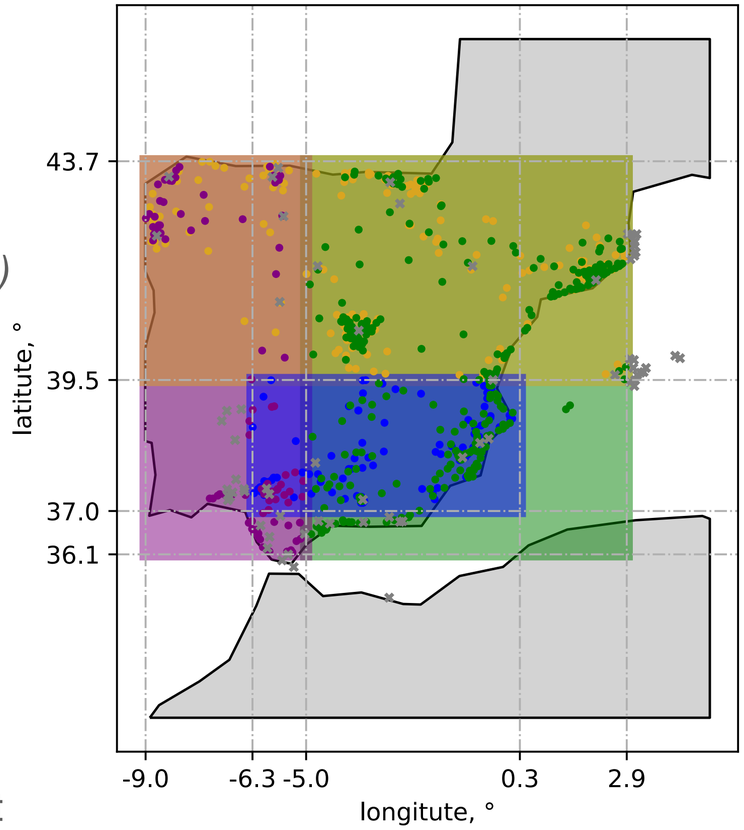
With no overlap

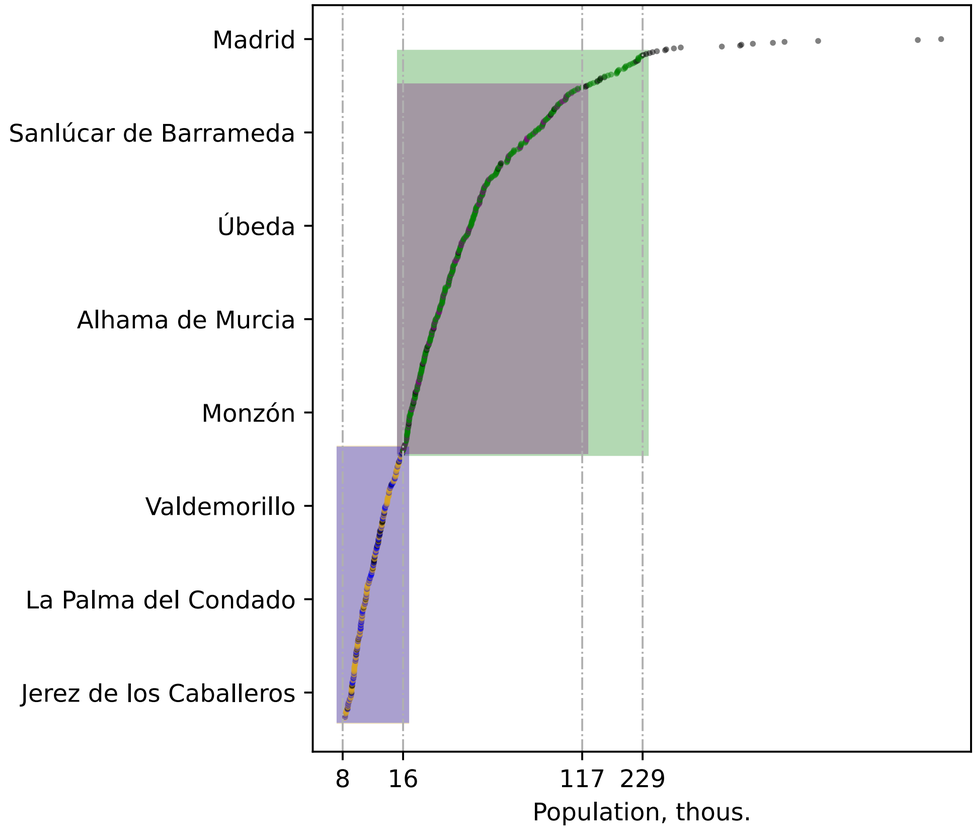
Here is the best clustering of 4 clusters with no-overlaps.

The clusters can be interpreted as:

- **Blue cluster**
  Southern small cities
  (since Latitude ≤ 39.5, Population ≤16)
- **Orange cluster**
  Northern small cities
  (since Lat. ≥ 39.5, Population ≤ 16)
- **Purple cluster**
  Western big cities
  (since Long. ≤ -5, Population ≥16)
- **Green cluster**
  Eastern big cities
  (since Long. ≥ -5, Population ≥ 16)

The clusters make sense, but somewhat “trivial”. Let us see something more interesting.





NB: The clusters overlap on the two dimensional figures, so as their colours

Rectangular clusters

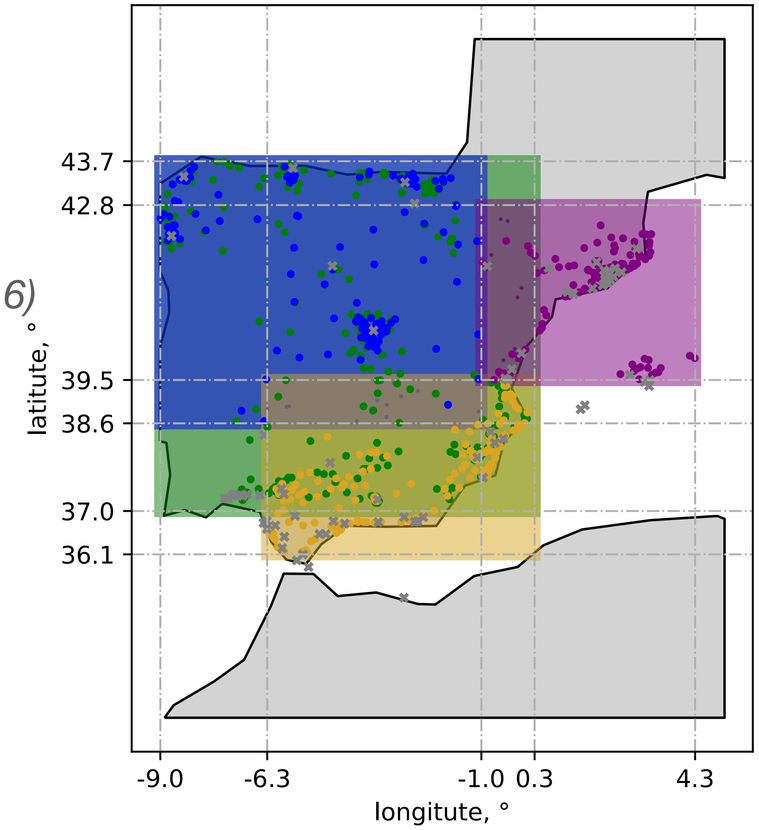
With small overlap

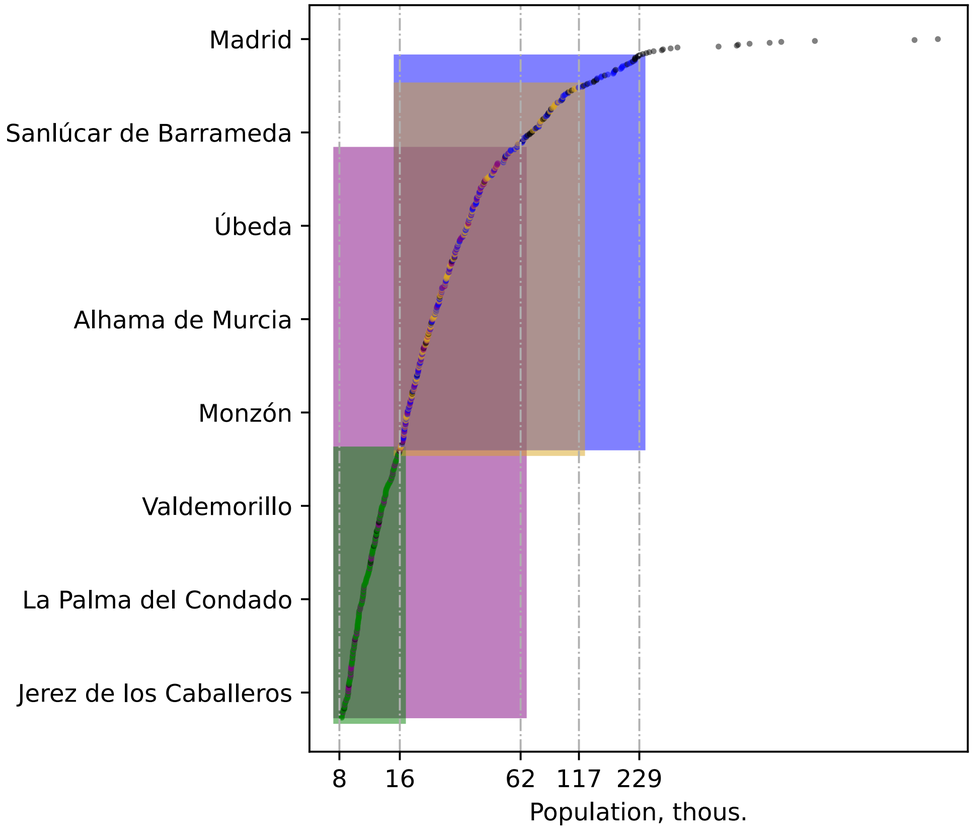
Here is the clustering of 4 clusters with the smallest non-zero overlap.

The clusters can be interpreted as:

- **Blue cluster**
  Big cities on the North-West
  (since Latitude ≥ 38.6, Population ≥ 16)
- **Orange cluster**
  Big cities on the South-East
  (since Lat. ≤ 39.5, Population ≥ 16)
- **Green cluster**
  Small cities
  (since Population ≤ 16)
- **Purple cluster**
  Catalonia and Mallorca
  (since Long. ≥ -1, Latitude ≥ 39.5)

These clusters still make sense, but are less trivial than the previous one.





Rectangular clusters

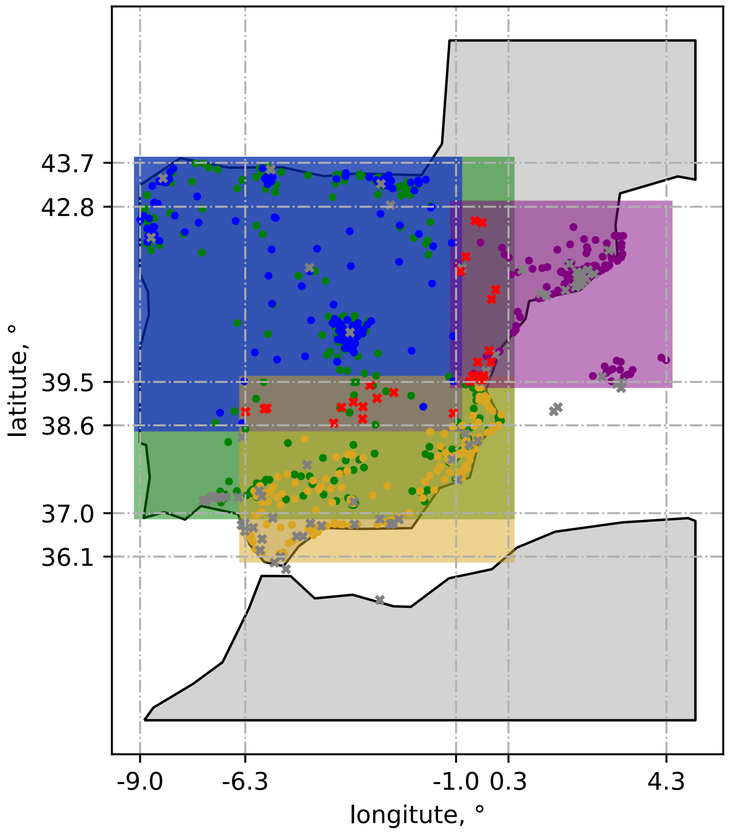
Highlighted overlap

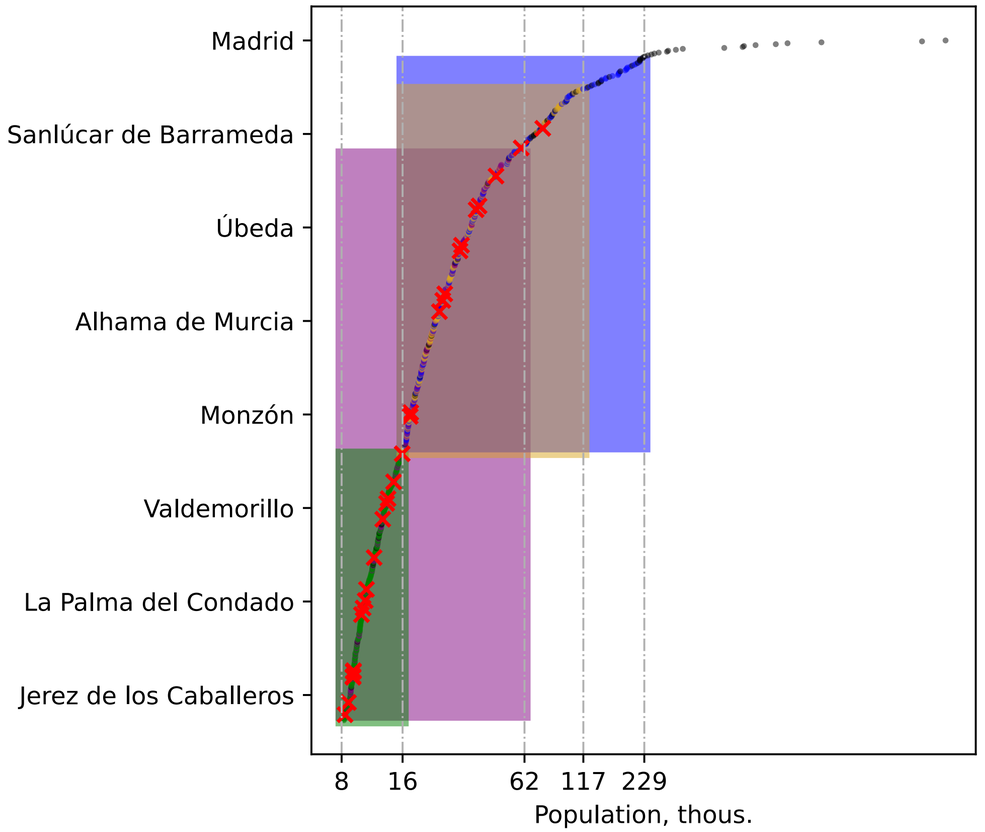
Note that the current clustering has overlaps (shown with red crosses) and does not cover all cities.

The overlaps happen at two regions:

- **Between 38.6 and 39.6 latitudes** where we cannot decide if a city belongs to the orange cluster (the Southern one) or to the blue cluster (the Northern one)
- **Between -1 and 0.3 longitudes** where we cannot decide if a city belongs to the green cluster (small cities) or to the purple cluster (Catalonia).

We believe it is the task of the data analyst to resolve such problems.





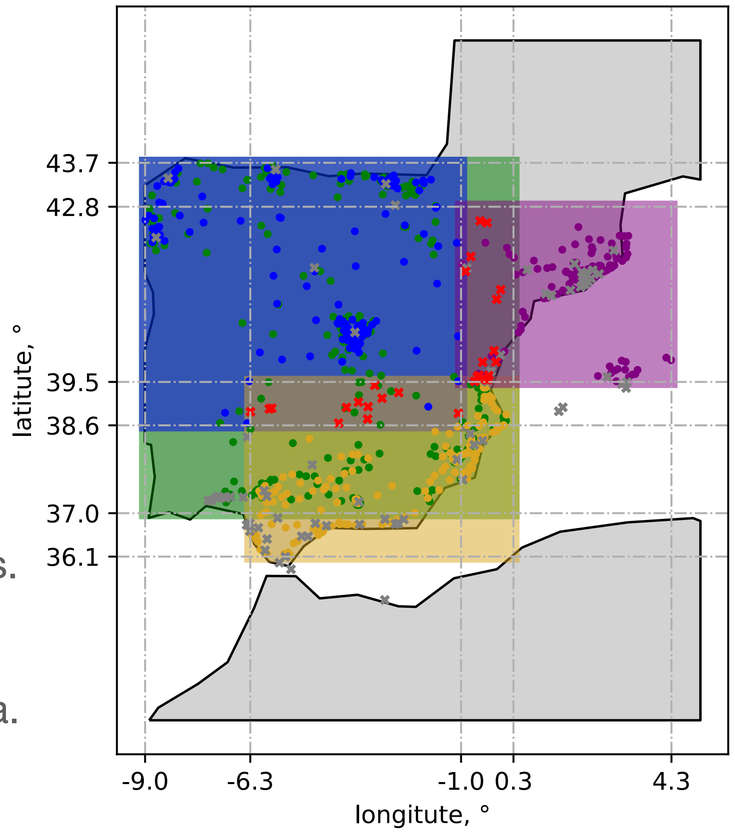
Rectangular clusters

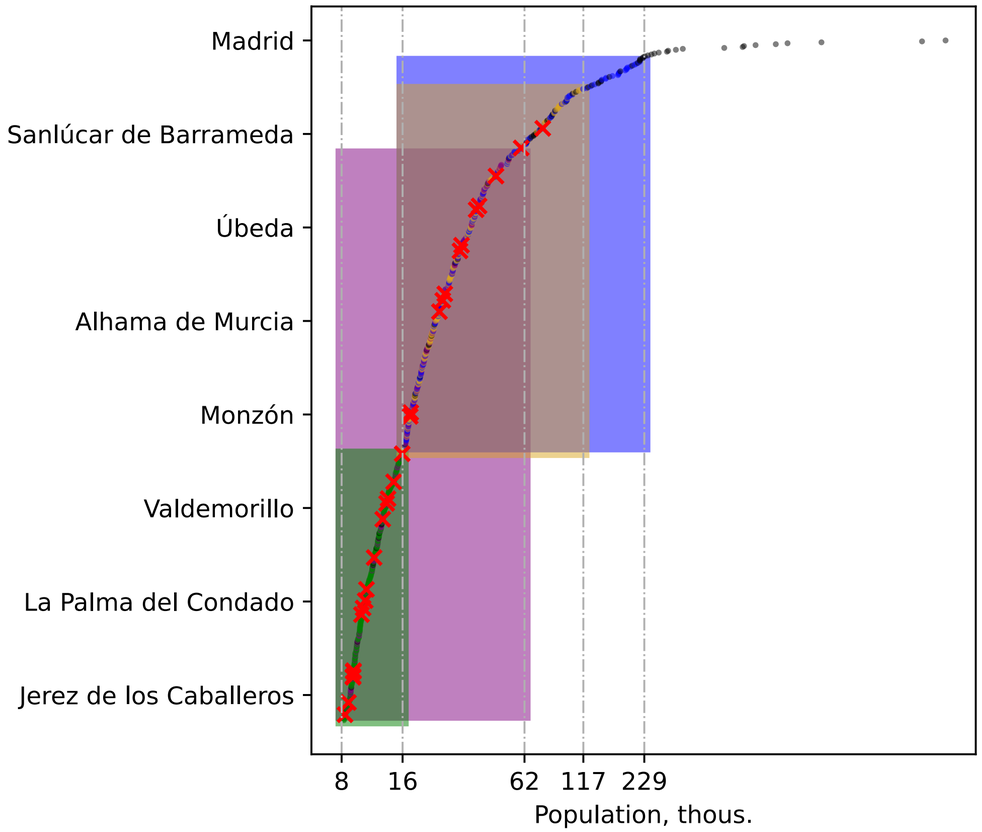
Non-clustered cities

Many non-clustered city (marked with grey colour) are the ones with high population.

This is because there are only 18 cities whose population is higher than 229 thousands people.
So the cluster of highly populated cities is too small to be found automatically. However, a data analyst can add it manually to the set of cluster candidates.

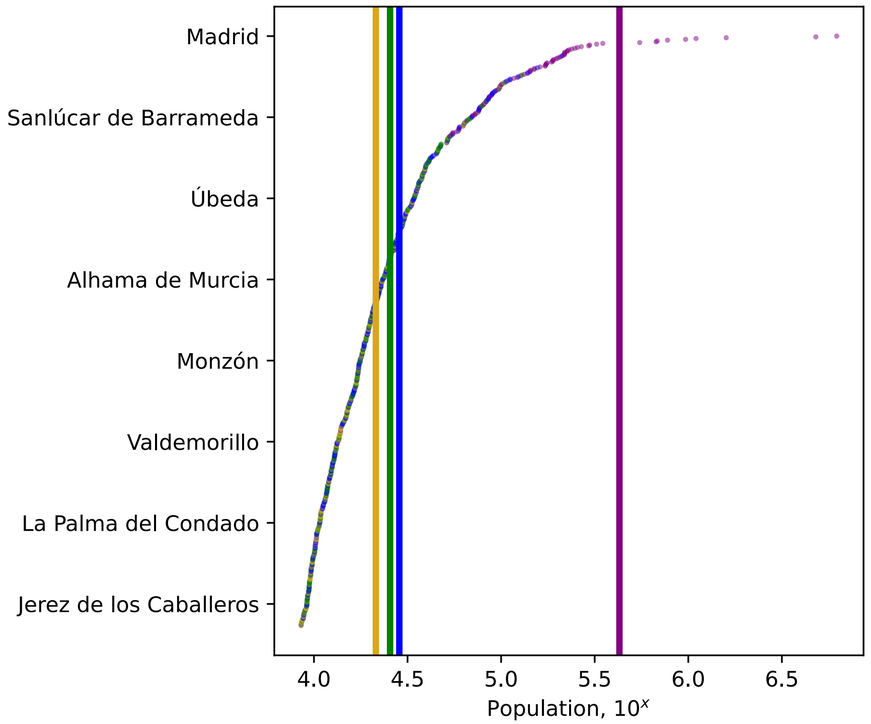
Among the other non-covered cities are the cities on Ibiza and in the North Africa. These regions form very particular clusters on their own.





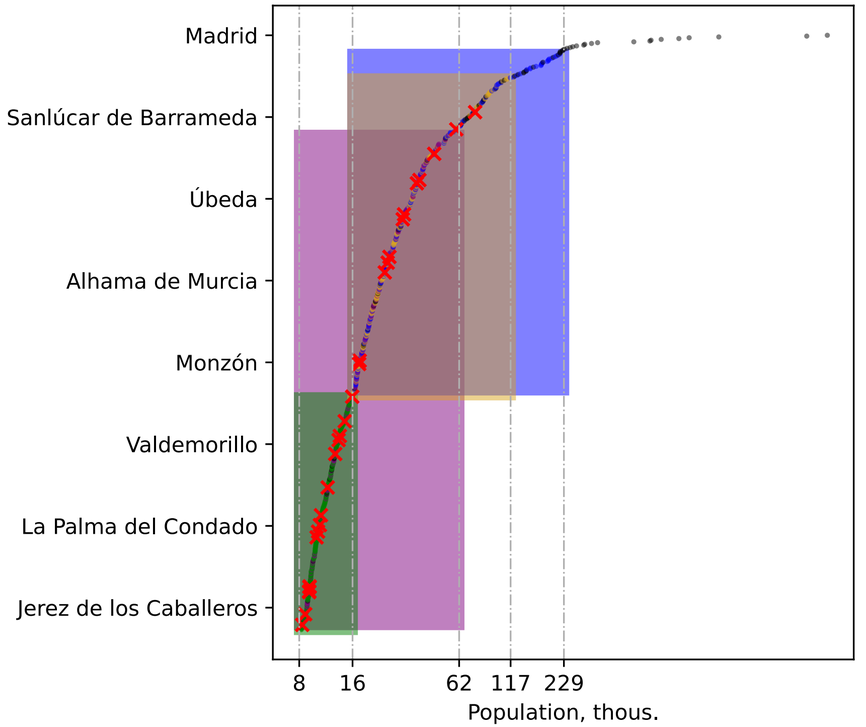
Power-Law distributed features

KMeans clustering



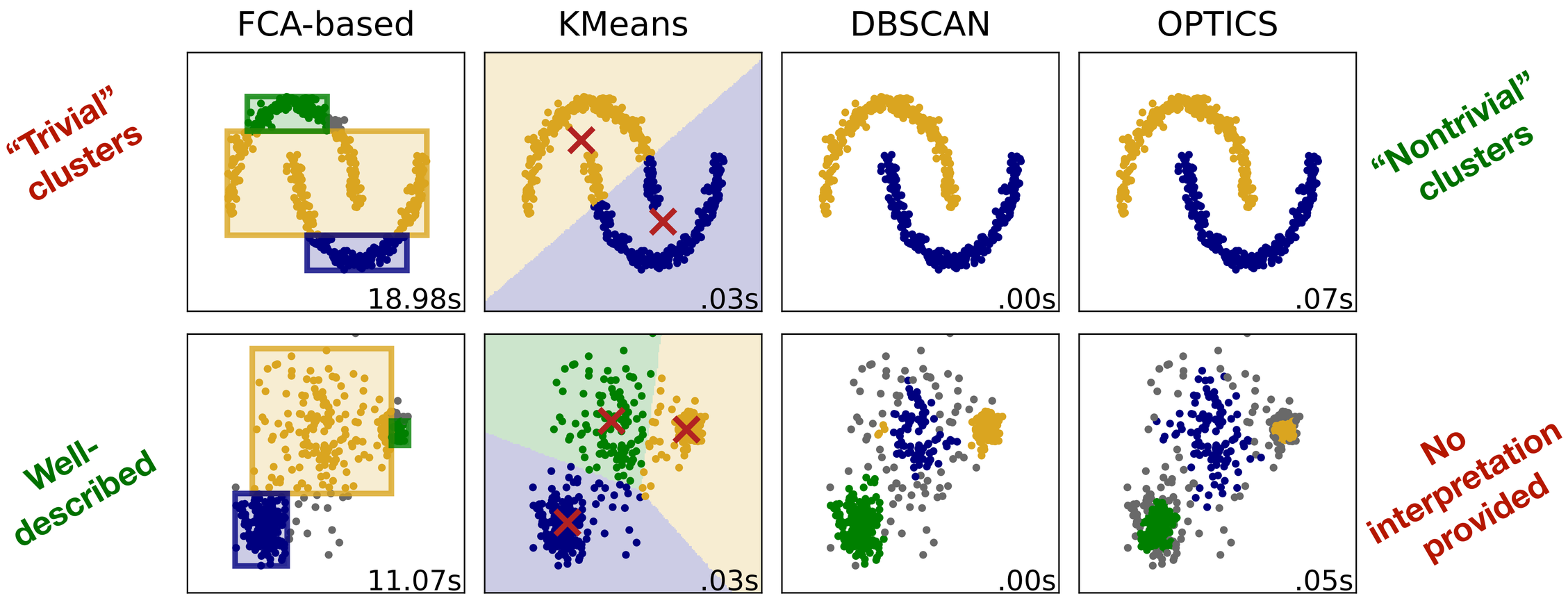
To make KMeans (and Euclidean distances) work, we should reduce the population difference among the big cities and increase the difference among small cities. For example, we can take a logarithm of the population values.
So we find clusters w.r.t. the dimension of “logarithm’ed people”.

FCA-based clustering



FCA-based clustering (with Interval Pattern Structure) is only concerned about the order of values.
Thus, order-preserving mappings do not affect the clustering results.
So we find clusters w.r.t. the dimension of “people”, which is easy to interpret.

Toy data comparison



Running time

The bottleneck

dataset	Step 2		Step 3		Step 4	total time (s)
	# stable concepts	stable concepts time (s)	# clusterings	clusterings time (s)	statistics time (s)	
noisy_circles	1 150	0.06	129 629	84.73	4.28	89.07
noisy_moons	636	0.04	99 082	15.86	3.08	18.98
varied	564	0.04	71 696	8.77	2.26	11.07
aniso	342	0.03	21 353	1.55	0.96	2.54
blobs	554	0.04	51 796	7.17	2.37	9.57
no_structure	1 139	0.05	96 914	84.18	3.19	87.42

Table 1
The time and the size of the output for every step of the proposed clustering pipeline.

Conclusion

Conclusions

We have presented an **original pipeline for clustering** numerical data using FCA and PS.

The pipeline consists of four steps:

1. we encode the data via **Interval and Cartesian Pattern Structures**,
2. we **find the set of stable cluster candidates** thanks to the gSofia algorithm,
3. we **enumerate the set of** minimal broad non-overlapping **clustering candidates**, and
4. we **select the best clustering candidate** based on a set of interestingness measures.

We show that this approach outputs some **reasonable clusterings**, while running in a **matter of seconds**.

Future work

We are planning to mainly **improve the third step of the pipeline**, by reducing the space of the clustering candidates.

We will also run experiments over **real-world complex datasets** with numerical, categorical, and textual elements.

Finally, our research raises the question of the type of clusters that can be found when using an FCA framework, i.e., how to define a pattern structure able to describe dense **continuous clusters**, or **rotated hyperrectangles**, or any **polygons in multidimensional space**.

Clustering with Stable Pattern Concepts

Egor Dudyrev, Mariia Zueva, Sergei O. Kuznetsov and Amedeo Napoli

19 October 2024, FCA4AI@ECAI24, Santiago de Compostela