

cobras

interactive clustering with pairwise queries

Toon Van Craenendonck, Sebastijan Dumancic, Elia Vanwolputte, Hendrik Blockeel
26/10/2018

Overview

1. Motivation & background
 - Semi-supervised and interactive clustering
 - COBRA
2. COBRAS
3. Conclusion

1. Motivation & background

Semi-supervised and interactive clustering

COBRA

2. COBRAS

3. Conclusion

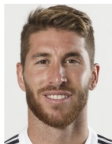
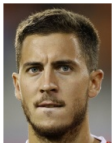
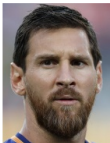
1. Motivation & background

Semi-supervised and interactive clustering

COBRA

2. COBRAS

3. Conclusion



League?



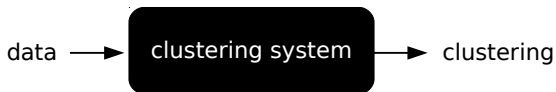
Position?





Playing style?

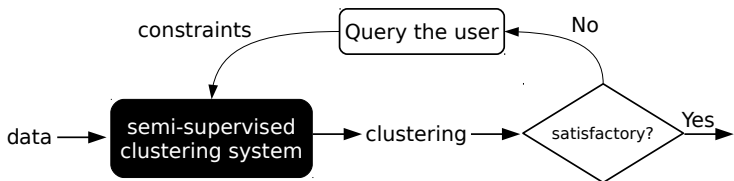


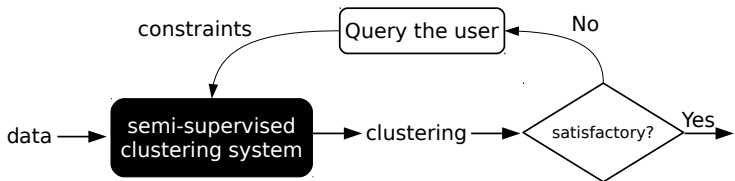
Clustering is **subjective**,
a black box system cannot work



Semi-supervised clustering systems rely on *interaction in the form of pairwise queries*

Should  and  be in the same cluster?





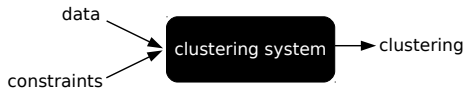
Such an *interactive* workflow requires the clustering system to be:

1. **anytime**: the user can stop and get the best result so far at any time
2. **query-efficient**: few queries before a reasonable result is obtained
3. **time-efficient**: user should not have to wait long between answering queries

Such an *interactive* workflow requires the clustering system to be:

1. **anytime**: the user can stop and get the best result so far at any time

Most existing semi-supervised clustering systems do not support this



2. **query-efficient**: few queries before a reasonable result is obtained
3. **time-efficient**: user should not have to wait long between answering queries

Such an *interactive* workflow requires the clustering system to be:

1. **anytime**: the user can stop and get the best result so far at any time
2. **query-efficient**: few queries before a reasonable result is obtained

Active clustering systems try to get a good result given as few queries as possible, but most of them require all queries to be known beforehand

3. **time-efficient**: user should not have to wait long between answering queries

Such an *interactive* workflow requires the clustering system to be:

1. **anytime**: the user can stop and get the best result so far at any time
2. **query-efficient**: few queries before a reasonable result is obtained
3. **time-efficient**: user should not have to wait long between answering queries

We can re-run existing systems each time a new constraint is given,
but this becomes slow for reasonably sized datasets

1. Motivation & background

Semi-supervised and interactive clustering

COBRA

2. COBRAS

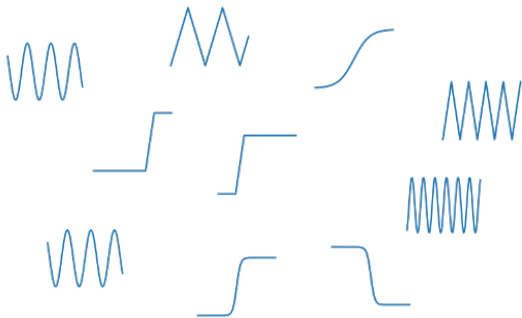
3. Conclusion

COBRA (Constraint-based Repeated Aggregation)

1. Over-cluster the data to create **super-instances**
= *intermediate layer between instances and clusters*
2. Merge these super-instances into clusters using pairwise constraints

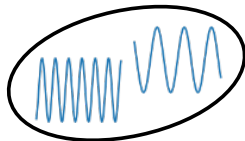
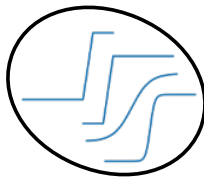
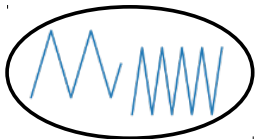
COBRA (Constraint-based Repeated Aggregation)

1. Over-cluster the data to create **super-instances**
= *intermediate layer between instances and clusters*
2. Merge these super-instances into clusters using pairwise constraints



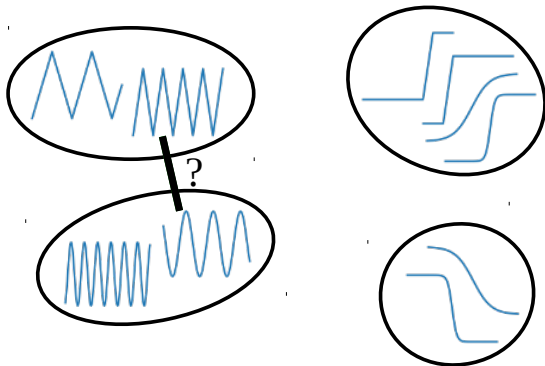
COBRA (Constraint-based Repeated Aggregation)

1. Over-cluster the data to create **super-instances**
= *intermediate layer between instances and clusters*
2. Merge these super-instances into clusters using pairwise constraints



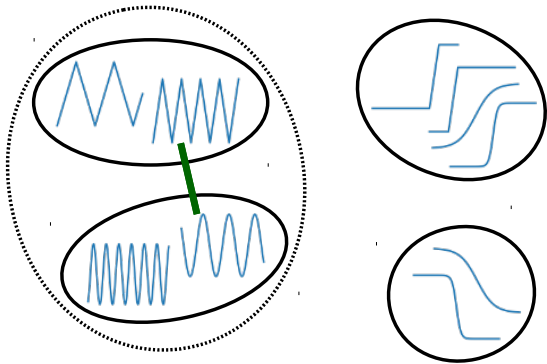
COBRA (Constraint-based Repeated Aggregation)

1. Over-cluster the data to create **super-instances**
= *intermediate layer between instances and clusters*
2. Merge these super-instances into clusters using pairwise constraints



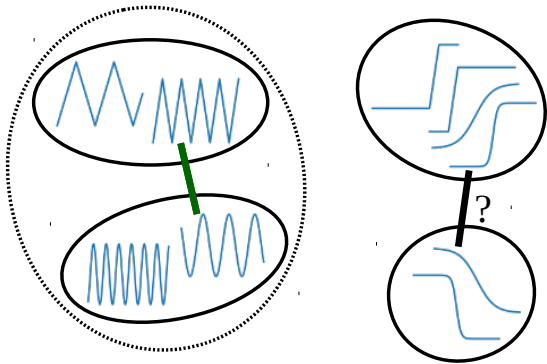
COBRA (Constraint-based Repeated Aggregation)

1. Over-cluster the data to create **super-instances**
= *intermediate layer between instances and clusters*
2. Merge these super-instances into clusters using pairwise constraints



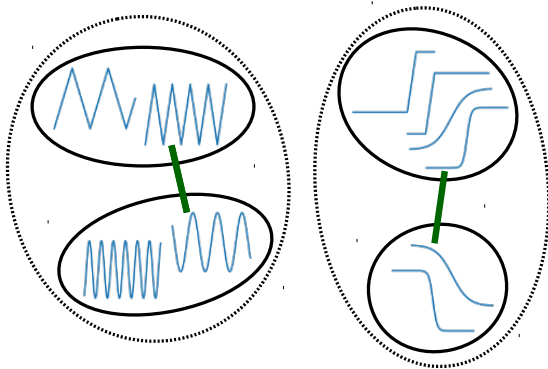
COBRA (Constraint-based Repeated Aggregation)

1. Over-cluster the data to create **super-instances**
= *intermediate layer between instances and clusters*
2. Merge these super-instances into clusters using pairwise constraints



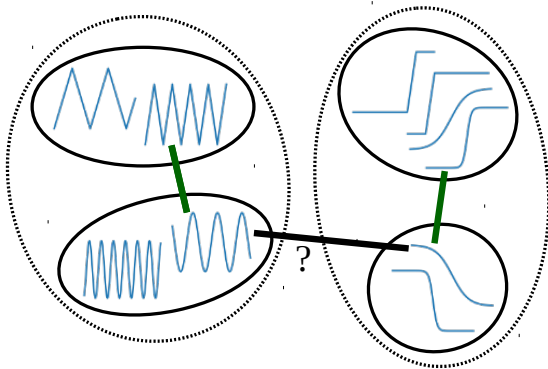
COBRA (Constraint-based Repeated Aggregation)

1. Over-cluster the data to create **super-instances**
= *intermediate layer between instances and clusters*
2. Merge these super-instances into clusters using pairwise constraints



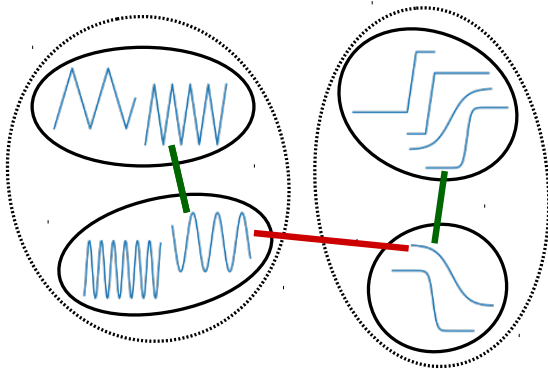
COBRA (Constraint-based Repeated Aggregation)

1. Over-cluster the data to create **super-instances**
= *intermediate layer between instances and clusters*
2. Merge these super-instances into clusters using pairwise constraints



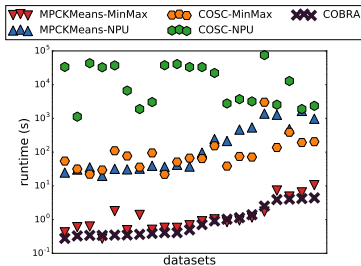
COBRA (Constraint-based Repeated Aggregation)

1. Over-cluster the data to create **super-instances**
= *intermediate layer between instances and clusters*
2. Merge these super-instances into clusters using pairwise constraints



COBRA is accurate and fast 😊

	25 super-instances		50 super-instances		100 super-instances	
COBRA	2.43	COBRA	2.14	COBRA	2.52	
MPCK-NPU	3.00	MPCK-MM*	3.00	COSC-NPU*	2.98	
MPCK-MM	3.07	COSC-NPU*	3.02	MPCK-NPU*	3.00	
COSC-MM*	3.12	COSC-MM*	3.26	MPCK-MM*	3.19	
COSC-NPU*	3.40	MPCK-NPU*	3.57	COSC-MM*	3.31	

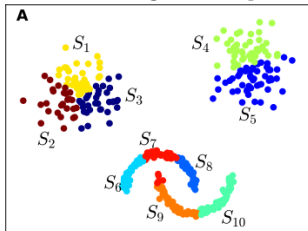


T. Van Craendonck, S. Dumancic, H. Blockeel. COBRA: A Fast and Simple Method for Active Clustering with Pairwise Constraints. IJCAI 2017

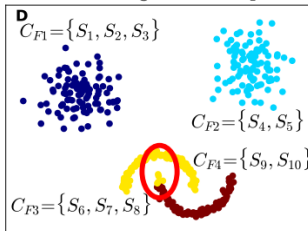
but it depends critically on the number of super-instances N_S ☹️

COBRA, $N_S = 10$

Initial clustering, after 0 queries

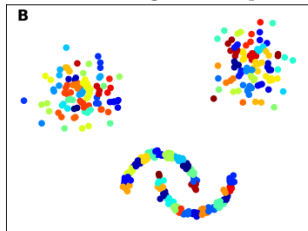


Final clustering, after 14 queries

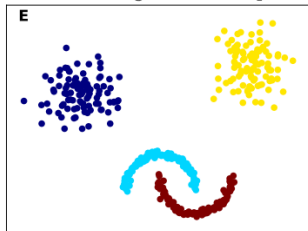


COBRA, $N_S = 100$

Initial clustering, after 0 queries



Final clustering, after 103 queries



1. Motivation & background

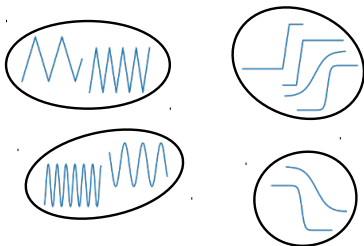
Semi-supervised and interactive clustering

COBRA

2. COBRAS

3. Conclusion

problem with COBRA: super-instances are fixed



*OK if we want to cluster based on monotonicity,
not if we want to cluster based on smoothness*

COBRAS: dynamically refine super-instances during clustering

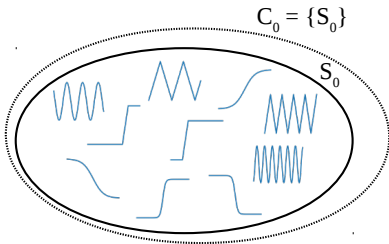
COBRAS

Constraint-based Repeated Aggregation and Splitting

Input: \mathcal{X} : a dataset, q : a query limit

Output: \mathcal{C} : a clustering of D

```
1:  $ML = \emptyset, CL = \emptyset$ 
2:  $S = \{\mathcal{X}\}, C = \{S\}, C = \{C\}$ 
3: while  $|ML| + |CL| < q$  do
4:    $S_{split}, C_{origin} = \arg \max_{S \in C, C \in C} |S|$ 
5:    $k, ML, CL =$ 
      $\text{determineSplitLevel}(S_{split}, ML, CL)$ 
6:    $S_{new_1}, \dots, S_{new_k} = \text{K-means}(S_{split}, k)$ 
7:    $C_{origin} = C_{origin} \setminus \{S_{split}\}$ 
8:    $C = C \cup \{\{S_{new_1}\}, \dots, \{S_{new_k}\}\}$ 
9:    $C, ML, CL = \text{COBRA}(C, ML, CL)$ 
10: end while
11: return  $C$ 
```



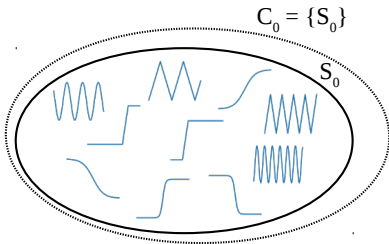
COBRAS

Constraint-based Repeated Aggregation and Splitting

Input: \mathcal{X} : a dataset, q : a query limit

Output: \mathcal{C} : a clustering of D

```
1:  $ML = \emptyset, CL = \emptyset$ 
2:  $S = \{\mathcal{X}\}, C = \{S\}, \mathcal{C} = \{C\}$ 
3: while  $|ML| + |CL| < q$  do
4:    $S_{split}, C_{origin} = \arg \max_{S \in C, C \in \mathcal{C}} |S|$ 
5:    $k, ML, CL =$ 
      $\text{determineSplitLevel}(S_{split}, ML, CL)$ 
6:    $S_{new_1}, \dots, S_{new_k} = \text{K-means}(S_{split}, k)$ 
7:    $C_{origin} = C_{origin} \setminus \{S_{split}\}$ 
8:    $C = C \cup \{\{S_{new_1}\}, \dots, \{S_{new_k}\}\}$ 
9:    $C, ML, CL = \text{COBRA}(C, ML, CL)$ 
10: end while
11: return  $\mathcal{C}$ 
```



COBRAS

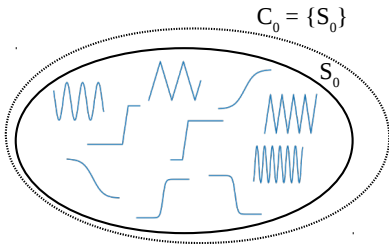
Constraint-based Repeated Aggregation and Splitting

Input: \mathcal{X} : a dataset, q : a query limit

Output: \mathcal{C} : a clustering of D

```
1:  $ML = \emptyset, CL = \emptyset$ 
2:  $S = \{\mathcal{X}\}, C = \{S\}, \mathcal{C} = \{C\}$ 
3: while  $|ML| + |CL| < q$  do
4:    $S_{split}, C_{origin} = \arg \max_{S \in C, C \in \mathcal{C}} |S|$ 
5:    $k, ML, CL =$   

   determineSplitLevel $(S_{split}, ML, CL)$ 
6:    $S_{new_1}, \dots, S_{new_k} = \text{K-means}(S_{split}, k)$ 
7:    $C_{origin} = C_{origin} \setminus \{S_{split}\}$ 
8:    $C = C \cup \{\{S_{new_1}\}, \dots, \{S_{new_k}\}\}$ 
9:    $C, ML, CL = \text{COBRA}(C, ML, CL)$ 
10: end while
11: return  $\mathcal{C}$ 
```



COBRAS

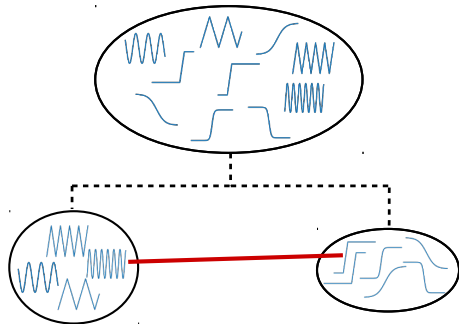
Constraint-based Repeated Aggregation and Splitting

Input: \mathcal{X} : a dataset, q : a query limit

Output: \mathcal{C} : a clustering of D

```
1:  $ML = \emptyset, CL = \emptyset$ 
2:  $S = \{\mathcal{X}\}, C = \{S\}, \mathcal{C} = \{C\}$ 
3: while  $|ML| + |CL| < q$  do
4:    $S_{split}, C_{origin} = \arg \max_{S \in C, C \in \mathcal{C}} |S|$ 
5:    $k, ML, CL =$   

   determineSplitLevel $(S_{split}, ML, CL)$ 
6:    $S_{new_1}, \dots, S_{new_k} = \text{K-means}(S_{split}, k)$ 
7:    $C_{origin} = C_{origin} \setminus \{S_{split}\}$ 
8:    $\mathcal{C} = \mathcal{C} \cup \{\{S_{new_1}\}, \dots, \{S_{new_k}\}\}$ 
9:    $\mathcal{C}, ML, CL = \text{COBRA}(\mathcal{C}, ML, CL)$ 
10: end while
11: return  $\mathcal{C}$ 
```



COBRAS

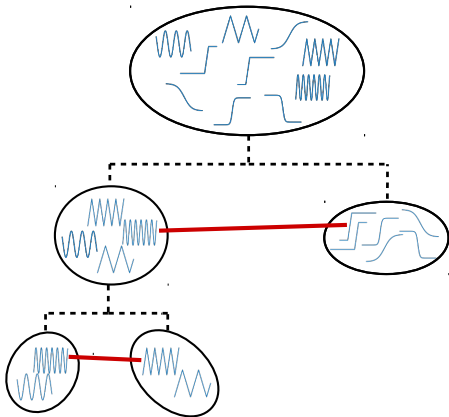
Constraint-based Repeated Aggregation and Splitting

Input: \mathcal{X} : a dataset, q : a query limit

Output: \mathcal{C} : a clustering of D

```
1:  $ML = \emptyset, CL = \emptyset$ 
2:  $S = \{\mathcal{X}\}, C = \{S\}, \mathcal{C} = \{C\}$ 
3: while  $|ML| + |CL| < q$  do
4:    $S_{split}, C_{origin} = \arg \max_{S \in C, C \in \mathcal{C}} |S|$ 
5:    $k, ML, CL =$   

   determineSplitLevel $(S_{split}, ML, CL)$ 
6:    $S_{new_1}, \dots, S_{new_k} = \text{K-means}(S_{split}, k)$ 
7:    $C_{origin} = C_{origin} \setminus \{S_{split}\}$ 
8:    $C = C \cup \{\{S_{new_1}\}, \dots, \{S_{new_k}\}\}$ 
9:    $C, ML, CL = \text{COBRA}(C, ML, CL)$ 
10: end while
11: return  $\mathcal{C}$ 
```



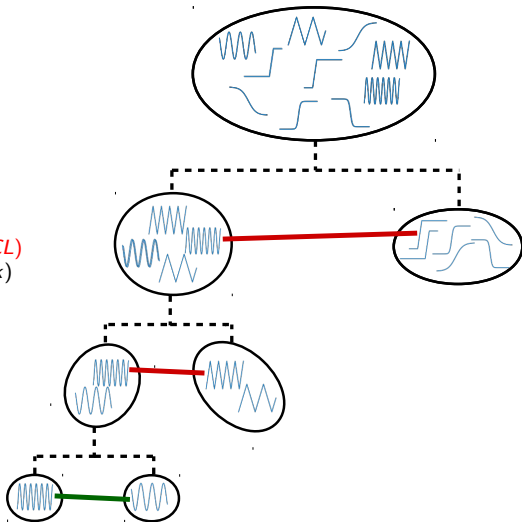
COBRAS

Constraint-based Repeated Aggregation and Splitting

Input: \mathcal{X} : a dataset, q : a query limit

Output: \mathcal{C} : a clustering of D

- 1: $ML = \emptyset, CL = \emptyset$
- 2: $S = \{\mathcal{X}\}, C = \{S\}, \mathcal{C} = \{C\}$
- 3: **while** $|ML| + |CL| < q$ **do**
- 4: $S_{split}, C_{origin} = \arg \max_{S \in C, C \in \mathcal{C}} |S|$
- 5: $k, ML, CL =$
 determineSplitLevel (S_{split}, ML, CL)
- 6: $S_{new_1}, \dots, S_{new_k} = \text{K-means}(S_{split}, k)$
- 7: $C_{origin} = C_{origin} \setminus \{S_{split}\}$
- 8: $\mathcal{C} = \mathcal{C} \cup \{\{S_{new_1}\}, \dots, \{S_{new_k}\}\}$
- 9: $\mathcal{C}, ML, CL = \text{COBRA}(\mathcal{C}, ML, CL)$
- 10: **end while**
- 11: **return** \mathcal{C}



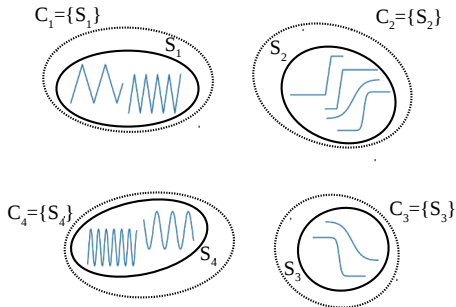
COBRAS

Constraint-based Repeated Aggregation and Splitting

Input: \mathcal{X} : a dataset, q : a query limit

Output: \mathcal{C} : a clustering of D

```
1:  $ML = \emptyset, CL = \emptyset$ 
2:  $S = \{\mathcal{X}\}, C = \{S\}, \mathcal{C} = \{C\}$ 
3: while  $|ML| + |CL| < q$  do
4:    $S_{split}, C_{origin} = \arg \max_{S \in C, C \in \mathcal{C}} |S|$ 
5:    $k, ML, CL =$ 
    $\text{determineSplitLevel}(S_{split}, ML, CL)$ 
6:    $S_{new_1}, \dots, S_{new_k} = \text{K-means}(S_{split}, k)$ 
7:    $C_{origin} = C_{origin} \setminus \{S_{split}\}$ 
8:    $C = C \cup \{\{S_{new_1}\}, \dots, \{S_{new_k}\}\}$ 
9:    $C, ML, CL = \text{COBRA}(C, ML, CL)$ 
10: end while
11: return  $\mathcal{C}$ 
```



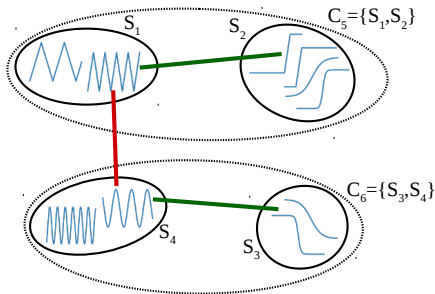
COBRAS

Constraint-based Repeated Aggregation and Splitting

Input: \mathcal{X} : a dataset, q : a query limit

Output: \mathcal{C} : a clustering of D

```
1:  $ML = \emptyset, CL = \emptyset$ 
2:  $S = \{\mathcal{X}\}, C = \{S\}, \mathcal{C} = \{C\}$ 
3: while  $|ML| + |CL| < q$  do
4:    $S_{split}, C_{origin} = \arg \max_{S \in C, C \in \mathcal{C}} |S|$ 
5:    $k, ML, CL =$ 
    $\text{determineSplitLevel}(S_{split}, ML, CL)$ 
6:    $S_{new_1}, \dots, S_{new_k} = \text{K-means}(S_{split}, k)$ 
7:    $C_{origin} = C_{origin} \setminus \{S_{split}\}$ 
8:    $\mathcal{C} = \mathcal{C} \cup \{\{S_{new_1}\}, \dots, \{S_{new_k}\}\}$ 
9:    $C, ML, CL = \text{COBRA}(\mathcal{C}, ML, CL)$ 
10: end while
11: return  $\mathcal{C}$ 
```



COBRAS

Constraint-based Repeated Aggregation and Splitting

Input: \mathcal{X} : a dataset, q : a query limit

Output: \mathcal{C} : a clustering of D

```
1:  $ML = \emptyset, CL = \emptyset$ 
2:  $S = \{\mathcal{X}\}, C = \{S\}, \mathcal{C} = \{C\}$ 
3: while  $|ML| + |CL| < q$  do
4:    $S_{split}, C_{origin} = \arg \max_{S \in C, C \in \mathcal{C}} |S|$ 
5:    $k, ML, CL =$ 
      $\text{determineSplitLevel}(S_{split}, ML, CL)$ 
6:    $S_{new_1}, \dots, S_{new_k} = \text{K-means}(S_{split}, k)$ 
7:    $C_{origin} = C_{origin} \setminus \{S_{split}\}$ 
8:    $\mathcal{C} = \mathcal{C} \cup \{\{S_{new_1}\}, \dots, \{S_{new_k}\}\}$ 
9:    $C, ML, CL = \text{COBRA}(C, ML, CL)$ 
10: end while
11: return  $\mathcal{C}$ 
```



COBRAS

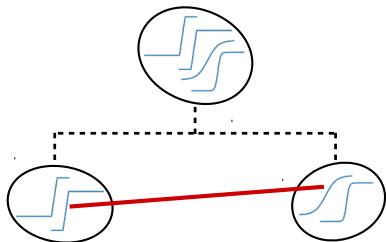
Constraint-based Repeated Aggregation and Splitting

Input: \mathcal{X} : a dataset, q : a query limit

Output: \mathcal{C} : a clustering of D

```
1:  $ML = \emptyset, CL = \emptyset$ 
2:  $S = \{\mathcal{X}\}, C = \{S\}, \mathcal{C} = \{C\}$ 
3: while  $|ML| + |CL| < q$  do
4:    $S_{split}, C_{origin} = \arg \max_{S \in C, C \in \mathcal{C}} |S|$ 
5:    $k, ML, CL =$   

   determineSplitLevel( $S_{split}, ML, CL$ )
6:    $S_{new_1}, \dots, S_{new_k} = \text{K-means}(S_{split}, k)$ 
7:    $C_{origin} = C_{origin} \setminus \{S_{split}\}$ 
8:    $\mathcal{C} = \mathcal{C} \cup \{\{S_{new_1}\}, \dots, \{S_{new_k}\}\}$ 
9:    $\mathcal{C}, ML, CL = \text{COBRA}(\mathcal{C}, ML, CL)$ 
10: end while
11: return  $\mathcal{C}$ 
```



COBRAS

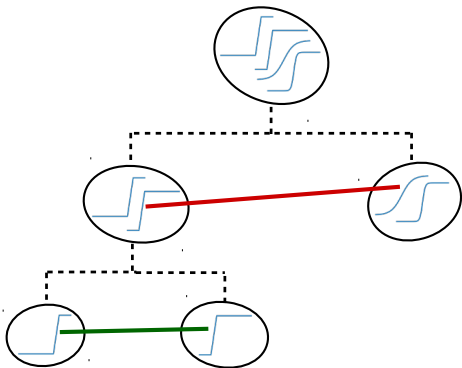
Constraint-based Repeated Aggregation and Splitting

Input: \mathcal{X} : a dataset, q : a query limit

Output: \mathcal{C} : a clustering of D

```
1:  $ML = \emptyset, CL = \emptyset$ 
2:  $S = \{\mathcal{X}\}, C = \{S\}, \mathcal{C} = \{C\}$ 
3: while  $|ML| + |CL| < q$  do
4:    $S_{split}, C_{origin} = \arg \max_{S \in C, C \in \mathcal{C}} |S|$ 
5:    $k, ML, CL =$   

   determineSplitLevel( $S_{split}, ML, CL$ )
6:    $S_{new_1}, \dots, S_{new_k} = \text{K-means}(S_{split}, k)$ 
7:    $C_{origin} = C_{origin} \setminus \{S_{split}\}$ 
8:    $C = C \cup \{\{S_{new_1}\}, \dots, \{S_{new_k}\}\}$ 
9:    $C, ML, CL = \text{COBRA}(C, ML, CL)$ 
10: end while
11: return  $\mathcal{C}$ 
```



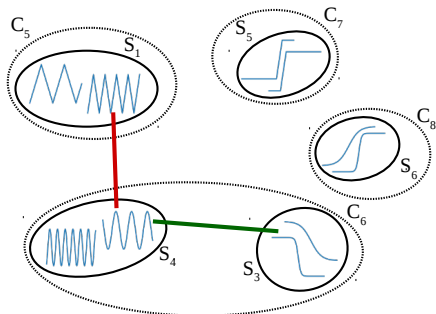
COBRAS

Constraint-based Repeated Aggregation and Splitting

Input: \mathcal{X} : a dataset, q : a query limit

Output: \mathcal{C} : a clustering of D

- 1: $ML = \emptyset, CL = \emptyset$
- 2: $S = \{\mathcal{X}\}, C = \{S\}, \mathcal{C} = \{C\}$
- 3: **while** $|ML| + |CL| < q$ **do**
- 4: $S_{split}, C_{origin} = \arg \max_{S \in C, C \in \mathcal{C}} |S|$
- 5: $k, ML, CL =$
 $\text{determineSplitLevel}(S_{split}, ML, CL)$
- 6: $S_{new_1}, \dots, S_{new_k} = \text{K-means}(S_{split}, k)$
- 7: $C_{origin} = C_{origin} \setminus \{S_{split}\}$
- 8: $\mathcal{C} = \mathcal{C} \cup \{\{S_{new_1}\}, \dots, \{S_{new_k}\}\}$
- 9: $C, ML, CL = \text{COBRA}(\mathcal{C}, ML, CL)$
- 10: **end while**
- 11: **return** \mathcal{C}



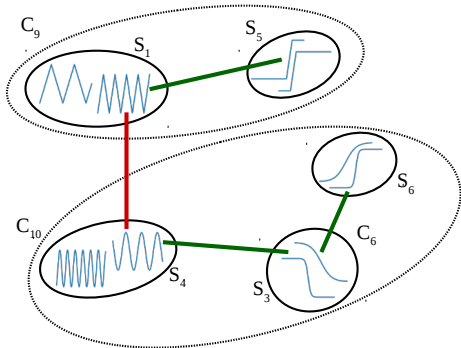
COBRAS

Constraint-based Repeated Aggregation and Splitting

Input: \mathcal{X} : a dataset, q : a query limit

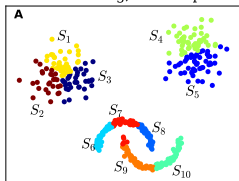
Output: \mathcal{C} : a clustering of D

- 1: $ML = \emptyset, CL = \emptyset$
- 2: $S = \{\mathcal{X}\}, C = \{S\}, \mathcal{C} = \{C\}$
- 3: **while** $|ML| + |CL| < q$ **do**
- 4: $S_{split}, C_{origin} = \arg \max_{S \in C, C \in \mathcal{C}} |S|$
- 5: $k, ML, CL =$
 $\text{determineSplitLevel}(S_{split}, ML, CL)$
- 6: $S_{new_1}, \dots, S_{new_k} = \text{K-means}(S_{split}, k)$
- 7: $C_{origin} = C_{origin} \setminus \{S_{split}\}$
- 8: $\mathcal{C} = \mathcal{C} \cup \{\{S_{new_1}\}, \dots, \{S_{new_k}\}\}$
- 9: $C, ML, CL = \text{COBRA}(C, ML, CL)$
- 10: **end while**
- 11: **return** \mathcal{C}

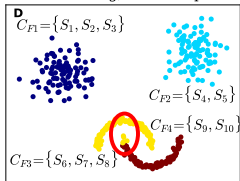


COBRA, $N_S = 10$

Initial clustering, after 0 queries

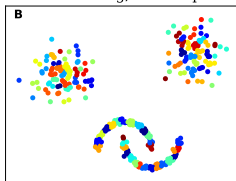


Final clustering, after 14 queries

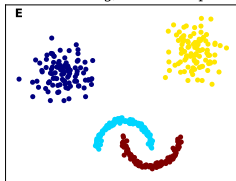


COBRA, $N_S = 100$

Initial clustering, after 0 queries

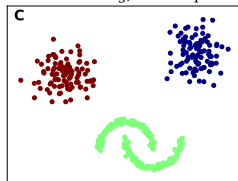


Final clustering, after 103 queries

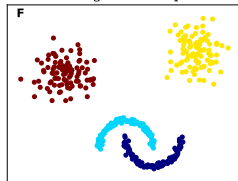


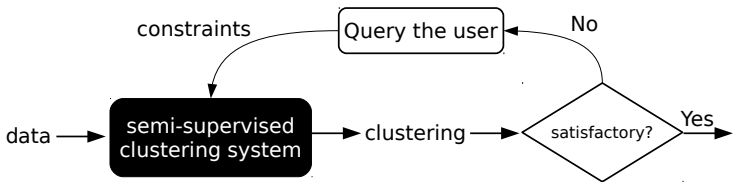
COBRAS

Initial clustering, after 5 queries



Clustering after 36 queries





An *interactive* workflow requires the clustering system to be:

1. **anytime**: the user can stop and get the best result so far at any time
2. **query-efficient**: few queries before a reasonable result is obtained
3. **time-efficient**: user should not have to wait long between answering queries

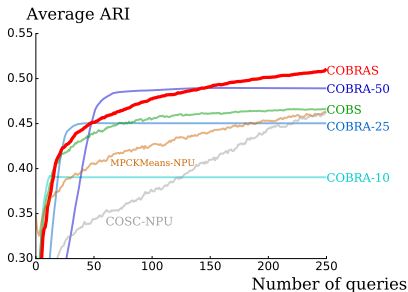
Does COBRAS satisfy these three requirements?

Experimental setup

- 21 clustering tasks (15 UCI datasets, 4 tasks on CMU faces, 2 tasks on 20 newsgroups text data)
- 10-fold CV, evaluating clustering quality on the test set using the adjusted Rand index (ARI)
- Comparing to
 - ▶ COBRA, with different numbers of super-instances
 - ▶ Constraint-based Clustering selection (COBS)
 - ▶ NPU instantiated with
 - ★ MPCKMeans
 - ★ Constrained spectral clustering (COSC)

COBRAS is:

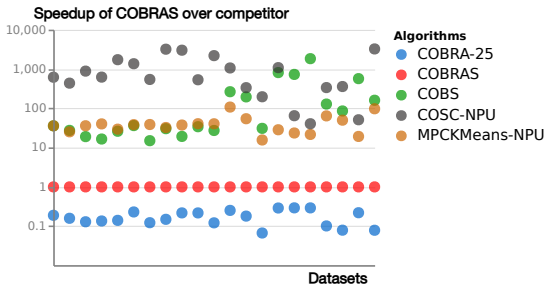
1. **anytime**: the user can stop and get the best result so far at any time ✓
2. **query-efficient**: few queries before a reasonable result is obtained ✓



3. **time-efficient**: user should not have to wait long between answering queries ✓

COBRAS is:

1. **anytime**: the user can stop and get the best result so far at any time ✓
2. **query-efficient**: few queries before a reasonable result is obtained ✓
3. **time-efficient**: user should not have to wait long between answering queries ✓



Can also be used for time series clustering,
if we plug in e.g. DTW + spectral clustering

The screenshot shows a web browser window with the URL `localhost:5000/webapp`. The page title is **COBRAS-TS** and it indicates `# queries answered: 10`.

The full dataset

This section displays a dense time series plot with a blue background and a blue line. It includes standard plot controls: a zoom-in icon, a zoom-out icon, a refresh icon, a close icon, and a color selection icon.

Should these two instances be in the same cluster?

This section shows two time series plots side-by-side. Below them are two buttons for user input:

- Yes (must-link)** (green button)
- No (cannot-link)** (orange button)

The (intermediate) clustering

This section displays three time series plots where data points are colored in blue, orange, and cyan to represent different clusters. Below each plot are two input fields:

- Field 1: `This cluster is pure.`
- Field 2: `This cluster is pure and complete.`

At the bottom of the interface is a red button labeled `Show me more queries!`

1. Motivation & background

Semi-supervised and interactive clustering

COBRA

2. COBRAS

3. Conclusion

Conclusion

Many semi-supervised methods, but none suited for *interactive* clustering

COBRAS can be used for this, it is **anytime**, **query-efficient** and **time-efficient**

cobras

<https://dtai.cs.kuleuven.be/software/cobras/>