How much k-means can be improved by using better initialization and repeats?

Pasi Fränti and Sami Sieranoja

11.4.2019

P. Fränti and S. Sieranoja, "How much k-means can be improved by using better initialization and repeats?, Pattern Recognition, 2019.
Introduction
Goal of k-means

Input $N$ points:

$$X = \{x_1, x_2, \ldots, x_N\}$$

Output partition and $k$ centroids:

$$P = \{p_1, p_2, \ldots, p_k\}$$

$$C = \{c_1, c_2, \ldots, c_k\}$$

Objective function:

$$SSE = \sum_{i=1}^{N} \left\| x_i - c_j \right\|^2$$

$SSE = \text{sum-of-squared errors}$
Goal of k-means

Input $N$ points:

$X = \{ x_1, x_2, \ldots, x_N \}$

Output partition and $k$ centroids:

$P = \{ \rho_1, \rho_2, \ldots, \rho_k \}$

$C = \{ c_1, c_2, \ldots, c_k \}$

Objective function:

$$SSE = \sum_{i=1}^{N} \left\| x_i - c_j \right\|^2$$

Assumptions:

- SSE is suitable
- $k$ is known
Using SSE objective function

Non-spherical
5 clusters

Different variance
2 clusters

Different density
8 clusters
K-means algorithm

http://cs.uef.fi/sipu/clustering/animator/

X = Data set
C = Cluster centroids
P = Partition

K-Means(X, C) → (C, P)

REPEAT

\[ C_{\text{prev}} \leftarrow C; \]

FOR \( i = 1 \) TO \( N \) DO

\[ p_i \leftarrow \text{FindNearest}(x_i, C); \]

FOR \( j = 1 \) TO \( k \) DO

\[ c_j \leftarrow \text{Average of } x_i \quad \forall \quad p_i = j; \]

UNTIL \( C = C_{\text{prev}} \)
K-means optimization steps

Assignment step:

\[ P_i = \arg \min_{1 \leq j \leq k} \| x_i - c_j \| ^2 \quad \forall i \in [1, N] \]

Centroid step:

\[ c_j = \frac{\sum_{P_i = j} x_i}{\sum_{P_i = j} 1} \quad \forall j \in [1, k] \]
Examples
Iterate by k-means

1st iteration
Iterate by k-means

2nd iteration
Iterate by k-means

3rd iteration
Iterate by k-means

16th iteration
Iterate by k-means

17th iteration
Iterate by k-means

18th iteration
Iterate by k-means

19th iteration
Final result

25 iterations
Problems of k-means

Distance of clusters

Cannot move centroids between clusters far away
Data and methodology
Clustering basic benchmark

Fränti and Sieranoja
K-means properties on six clustering benchmark datasets
*Applied Intelligence, 2018.*

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Varying</th>
<th>Size</th>
<th>Dimensions</th>
<th>Clusters</th>
<th>Per cluster</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Number of clusters</td>
<td>3000-7500</td>
<td>2</td>
<td>20-50</td>
<td>150</td>
</tr>
<tr>
<td>S</td>
<td>Overlap</td>
<td>5000</td>
<td>2</td>
<td>15</td>
<td>333</td>
</tr>
<tr>
<td>Dim</td>
<td>Dimensions</td>
<td>1024</td>
<td>32-1024</td>
<td>16</td>
<td>64</td>
</tr>
<tr>
<td>G2</td>
<td>Dimensions + overlap</td>
<td>2048</td>
<td>2-1024</td>
<td>2</td>
<td>1024</td>
</tr>
<tr>
<td>Birch</td>
<td>Structure</td>
<td>100,000</td>
<td>2</td>
<td>100</td>
<td>1000</td>
</tr>
<tr>
<td>Unbalance</td>
<td>Balance</td>
<td>6500</td>
<td>2</td>
<td>8</td>
<td>100-2000</td>
</tr>
</tbody>
</table>

G2-2-30

600,500

500,500
Centroid index
Requires ground truth

\[ \text{CI} = 4 \]

- 15 prototypes (pigeons)
- 15 real clusters (pigeon holes)

\textbf{CI = Centroid index:}

P. Fränti, M. Rezaei and Q. Zhao
“Centroid index: cluster level similarity measure”
*Pattern Recognition*, 47 (9), 3034-3045, September 2014.
Centroid index example

CI = 4

Missing centroids

Too many centroids
Success rate
How often CI = 0?

17%
Properties of k-means
Cluster overlap

Definition

\[
\text{overlap} = \frac{1}{N} \cdot \sum \text{ov}(d_1, d_2)
\]

\[
\text{ov}(d_1, d_2) = \begin{cases} 
1, & d_1 > d_2 \\
0, & d_1 \leq d_2 
\end{cases}
\]

d_1 = \text{distance to nearest centroid}

d_2 = \text{distance to nearest in other cluster}

Points = 2048
Evidence = 332
Overlap = 332 / 2048

16 %
Dependency on overlap

Success rates and CI-values:

- \( S_1 \): 3% (CI = 1.8)
- \( S_2 \): 11% (CI = 1.4)
- \( S_3 \): 12% (CI = 1.3)
- \( S_4 \): 26% (CI = 0.9)
Why overlap helps?

Overlap = 7%
13 iterations

Overlap = 22%
90 iterations
Linear dependency on clusters \((k)\)

**Birch2** dataset

![Graph showing relative CI-value vs. number of clusters (k)]

- **K-means**
- **Repeated k-means**
- **Birch2 subsets**

16%
Dependency on dimensions

**DIM datasets**

Dimensions increase:

- 32
- 64
- 128
- 256
- 512
- 1024

CI:

- 3.6
- 3.5
- 3.8
- 3.8
- 3.9
- 3.7

Success rate: 0%
Lack of overlap is the cause!

**G2 datasets**

<table>
<thead>
<tr>
<th>σ \ dim</th>
<th>2</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>32</th>
<th>64</th>
<th>128</th>
<th>256</th>
<th>512</th>
<th>1024</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>3%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>16%</td>
<td>5%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td></td>
<td>26%</td>
<td>19%</td>
<td>3%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>32%</td>
<td>31%</td>
<td>13%</td>
<td>1%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>39%</td>
<td>43%</td>
<td>27%</td>
<td>4%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>43%</td>
<td>50%</td>
<td>40%</td>
<td>11%</td>
<td>0.1%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>70</td>
<td>46%</td>
<td>58%</td>
<td>50%</td>
<td>20%</td>
<td>0.8%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>47%</td>
<td>61%</td>
<td>60%</td>
<td>31%</td>
<td>2.0%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>47%</td>
<td>66%</td>
<td>64%</td>
<td>39%</td>
<td>4.4%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>47%</td>
<td>67%</td>
<td>58%</td>
<td>41%</td>
<td>4.7%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Dimensions increases

- **Overlap**
- **Success**

Correlation: **0.91**
Effect of unbalance

Unbalance datasets

K-means tend to put too many clusters here ...

... and too few here

Success: 0%

Average CI: 3.9

Problem originates from the random initialization.
Summary of k-means properties

1. Overlap
   - Good!

2. Number of clusters
   - Linear dependency

3. Dimensionality
   - No direct effect

4. Unbalance of cluster sizes
   - Bad!
How to improve?
Repeated k-means (RKM)

Repeat 100 times

**Initialize**

**K-means**

Must include randomness
How to initialize?

Some obvious heuristics:
• Furthest point
• Sorting
• Density
• Projection

Clear state-of-the-art is missing:
• No single technique outperforms others in all cases.
• Initialization not significantly easier than the clustering itself.
• K-means can be used as fine-tuner with almost anything.

Another desperate effort:
• Repeat it **LOTS OF** times
Initialization techniques

Criteria

1. Simple to implement
2. Lower (or equal) time complexity than k-means
3. No additional parameters
4. Include randomness
Requirements for initialization

1. **Simple** to implement
   - Random centroids has 2 functions + **26 lines** of code
   - Repeated k-means 5 functions + **162 lines** of code
   - Simpler than k-means itself

2. **Faster** than k-means
   - K-means real-time (<**1 s**) up to \( N \approx 10,000 \)
   - \( O(IkN) \approx 25 \cdot N^{1.5} \) --- assuming \( I \approx 25 \) and \( k = \sqrt{N} \)
   - Must be faster than \( O(N^2) \)
Simplicity of algorithms

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>0</td>
<td>2</td>
<td>26</td>
</tr>
<tr>
<td>Rep. K-Means</td>
<td>1</td>
<td>5</td>
<td>162</td>
</tr>
<tr>
<td>Random swap</td>
<td>1</td>
<td>7</td>
<td>226</td>
</tr>
<tr>
<td>Agglomerative</td>
<td>0</td>
<td>12</td>
<td>317</td>
</tr>
<tr>
<td>SPLIT</td>
<td>0</td>
<td>22</td>
<td>947</td>
</tr>
<tr>
<td>GA</td>
<td>2</td>
<td>21</td>
<td>573</td>
</tr>
<tr>
<td>FCM</td>
<td>2</td>
<td>11</td>
<td>295</td>
</tr>
<tr>
<td>GMM</td>
<td>2</td>
<td>44</td>
<td>1111</td>
</tr>
</tbody>
</table>

Alternative: A better algorithm

Random Swap (RS)

Random Swap($X$) $\rightarrow C, P$

$C \leftarrow$ Select random representatives($X$);
$P \leftarrow$ Optimal partition($X, C$);
REPEAT $T$ times

$(C^\text{new}, j)$ $\leftarrow$ Random swap($X, C$);
$P^\text{new} \leftarrow$ Local repartition($X, C^\text{new}, P, j$);
$C^\text{new}, P^\text{new} \leftarrow$ Kmeans($X, C^\text{new}, P^\text{new}$);
IF $f(C^\text{new}, P^\text{new}) < f(C, P)$ THEN
$(C, P) \leftarrow C^\text{new}, P^\text{new}$,
RETURN ($C, P$);

P. Fränti, "Efficiency of random swap clustering", Journal of Big Data, 2018

Genetic Algorithm (GA)

GeneticAlgorithm($X$) $\rightarrow (C, P)$
FOR $i \leftarrow 1$ TO $Z$
$C_i \leftarrow$ RandomCodebook($X$);
$P_i \leftarrow$ OptimalPartition($X, C_i$);
SortSolutions($C, P$);
REPEAT
$\{C, P\} \leftarrow$ CreateNewSolutions($\{C, P\}$);
SortSolutions($C, P$);
UNTIL no improvement;

CreateNewSolutions($\{C, P\}$) $\rightarrow \{C^\text{new}, P^\text{new}\}$

$C^\text{new}^{-1}, P^\text{new}^{-1} \leftarrow C^1, P^1$;
FOR $i \leftarrow 2$ TO $Z$
$(a, b) \leftarrow$ SelectNextPair;
$C^\text{new}^{-i}, P^\text{new}^{-i} \leftarrow$ Cross($C^a, P^a, C^b, P^b$);
IterateK-Means($C^\text{new}^{-i}, P^\text{new}^{-i}$);
END-FOR

Cross($C^1, P^1, C^2, P^2$) $\rightarrow \{C^\text{new}, P^\text{new}\}$

$C^\text{new} \leftarrow$ CombineCentroids($C^1, C^2$);
$P^\text{new} \leftarrow$ CombinePartitions($P^1, P^2$);
$C^\text{new} \leftarrow$ UpdateCentroids($C^\text{new}, P^\text{new}$);
RemoveEmptyClusters($C^\text{new}, P^\text{new}$);

Cl = 0

Initialization techniques
<table>
<thead>
<tr>
<th>Technique</th>
<th>Complexity</th>
<th>Time</th>
<th>Randomized</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random partitions</td>
<td>O(N)</td>
<td>10 ms</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Random centroids</td>
<td>O(N)</td>
<td>13 ms</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Maxmin</td>
<td>O(kN)</td>
<td>16 ms</td>
<td>Modified</td>
<td></td>
</tr>
<tr>
<td>kmeans++</td>
<td>O(kN)</td>
<td>19 ms</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Bradley</td>
<td>O(kN+Rk^2)</td>
<td>41 ms</td>
<td>Yes</td>
<td>R=10, s=10%</td>
</tr>
<tr>
<td>Sorting heuristic</td>
<td>O(N log N)</td>
<td>13 ms</td>
<td>Modified</td>
<td></td>
</tr>
<tr>
<td>Projection-based</td>
<td>O(N log N)</td>
<td>14 ms</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Luxburg</td>
<td>O(kN log k)</td>
<td>29 ms</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Split</td>
<td>O(N log N)</td>
<td>67 ms</td>
<td>Yes</td>
<td>k=2</td>
</tr>
</tbody>
</table>
Random centroids

**Rand-C**
INIT

**Rand-C**
FINAL
Random partitions

Initial  Steinley  Final
Furthest points (maxmin)

Maxmin
INIT

KM++
INIT

\[ p_i = \frac{d_{NN(i)}}{\sum d_{NN(j)}} \]
Furthest points (maxmin)
Projection-based initialization

Most common projection axis:
- Diagonal
- Principal axis (PCA)
- Principle curves

Initialization:
- Uniform partition along axis

Used in divisive clustering:
- Iterative split

\[ \text{PCA} = O(DN) - O(D^2N) \]
Furthest point projection

Furthest point

Projection axis

Projected points

Initial clustering

After k-means
Projection example (1)

Good projection!

**Birch2**
Projection example (2)

Bad projection!

Birch1
Projection example (3)

Bad projection!

Unbalance

Projection

INIT
More complex projections

More complex projections


Travelling salesman problem!
Sorting heuristic
Density-based heuristics
Luxburg

Luxburg’s technique:

• Selects $k \log(k)$ preliminary clusters using k-means
• Eliminate the smallest.
• Furthest point heuristic to select $k$ centroids.

Initial solution

After k-means
Splitting algorithm

- Select biggest cluster
- Select two random points
- Re-allocate points
- Tune by k-means locally
Results
## Success rates
K-means (without repeats)

<table>
<thead>
<tr>
<th>Method</th>
<th>s1</th>
<th>s2</th>
<th>s3</th>
<th>s4</th>
<th>a1</th>
<th>a2</th>
<th>a3</th>
<th>unb</th>
<th>b1</th>
<th>b2</th>
<th>dim32</th>
<th>Aver.</th>
<th>Fails</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rand-P</td>
<td>0%</td>
<td>47%</td>
<td>5%</td>
<td>63%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>10%</td>
<td>8</td>
</tr>
<tr>
<td>Rand-C</td>
<td>3%</td>
<td>11%</td>
<td>12%</td>
<td>26%</td>
<td>1%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>5%</td>
<td>6</td>
</tr>
<tr>
<td>Maxmin</td>
<td>37%</td>
<td>16%</td>
<td>36%</td>
<td>9%</td>
<td>15%</td>
<td>1%</td>
<td>0%</td>
<td>22%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>22%</td>
<td>3</td>
</tr>
<tr>
<td>kmeans++</td>
<td>21%</td>
<td>24%</td>
<td>18%</td>
<td>30%</td>
<td>7%</td>
<td>0%</td>
<td>0%</td>
<td>51%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>22%</td>
<td>4</td>
</tr>
<tr>
<td>Bradley</td>
<td>21%</td>
<td>46%</td>
<td>49%</td>
<td>64%</td>
<td>7%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>2%</td>
<td>17%</td>
</tr>
<tr>
<td>Sorting</td>
<td>12%</td>
<td>20%</td>
<td>22%</td>
<td>36%</td>
<td>10%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>12%</td>
<td>15%</td>
<td>4</td>
</tr>
<tr>
<td>Projection</td>
<td>16%</td>
<td>29%</td>
<td>30%</td>
<td>42%</td>
<td>18%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>92%</td>
<td>34%</td>
<td>4</td>
</tr>
<tr>
<td>Luxburg</td>
<td>52%</td>
<td>60%</td>
<td>45%</td>
<td>61%</td>
<td>45%</td>
<td>33%</td>
<td>31%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>17%</td>
<td>40%</td>
<td>2</td>
</tr>
<tr>
<td>Split</td>
<td>78%</td>
<td>75%</td>
<td>62%</td>
<td>64%</td>
<td>51%</td>
<td>17%</td>
<td>5%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>10%</td>
<td>42%</td>
<td>2</td>
</tr>
</tbody>
</table>

**Most problems:** a2, a3, unbalance, Birch1, Birch2

**Average success rate**

**No. of datasets never solved**
Cluster overlap
High cluster overlap

High overlap

- Rand-P
- Rand-C
- MaxMin
- KMplus
- Bradley
- Project
- Sorting
- ProjRP
- Split

After K-means

Initial
Cluster overlap

Low cluster overlap

Low overlap

After K-means

Initial

Rand-P  Rand-C  MaxMin  KPlus  Bradley  Project  Sorting  ProjRP  Split

G2
Number of clusters

Birch2 subsets

![Graph showing relative Cl-value against clusters (k)]
Unbalance

Rand-C
INIT

Maximin
INIT

Rand-C
FINAL

KM++
INIT

Projection
INIT

Unbalance
### Success rates

**Repeated k-means**

<table>
<thead>
<tr>
<th>Method</th>
<th>s1</th>
<th>s2</th>
<th>s3</th>
<th>s4</th>
<th>a1</th>
<th>a2</th>
<th>a3</th>
<th>unb</th>
<th>b1</th>
<th>b2</th>
<th>dim32</th>
<th>Aver.</th>
<th>Fails</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rand-P</td>
<td>0%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>27%</td>
<td>8</td>
</tr>
<tr>
<td>Rand-C</td>
<td>96%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>56%</td>
<td>2%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>2%</td>
<td>41%</td>
<td>4</td>
</tr>
<tr>
<td>Maxmin</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>58%</td>
<td>36%</td>
<td>100%</td>
<td>0%</td>
<td>0%</td>
<td>100%</td>
<td>72%</td>
<td>2</td>
</tr>
<tr>
<td>kmeans++</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>98%</td>
<td>20%</td>
<td>0%</td>
<td>100%</td>
<td>0%</td>
<td>0%</td>
<td>100%</td>
<td>65%</td>
<td>3</td>
</tr>
<tr>
<td>Bradley</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>4%</td>
<td>4%</td>
<td>4%</td>
<td>0%</td>
<td>0%</td>
<td>84%</td>
<td>54%</td>
<td>2</td>
</tr>
<tr>
<td>Sorting</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>24%</td>
<td>0%</td>
<td>0%</td>
<td>2%</td>
<td>100%</td>
<td>100%</td>
<td>66%</td>
<td>2</td>
</tr>
<tr>
<td>Projection</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>18%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>100%</td>
<td>65%</td>
<td>3</td>
</tr>
<tr>
<td>Luxburg</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>0%</td>
<td>46%</td>
<td>100%</td>
<td>100%</td>
<td>86%</td>
<td>1</td>
</tr>
<tr>
<td>Split</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>0%</td>
<td>36%</td>
<td>100%</td>
<td>100%</td>
<td>85%</td>
<td>1</td>
</tr>
</tbody>
</table>

**Furthest point approaches** solve unbalance

**Still problems:** Birch1, Birch2

**Average success rate**

**No. of datasets never solved**
## How many repeats needed?

A3

<table>
<thead>
<tr>
<th>Initialization</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rand-P</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Rand-C</td>
<td>2</td>
<td>4</td>
<td>11</td>
<td>54</td>
<td>428</td>
<td>11111</td>
<td>-</td>
</tr>
<tr>
<td>Maxmin</td>
<td>1</td>
<td>3</td>
<td>14</td>
<td>216</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Kmeans++</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>14</td>
<td>138</td>
<td>8696</td>
<td>-</td>
</tr>
<tr>
<td>Bradley</td>
<td>1</td>
<td>2</td>
<td>8</td>
<td>58</td>
<td>1058</td>
<td>33333</td>
<td>-</td>
</tr>
<tr>
<td>Sorting</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>13</td>
<td>73</td>
<td>1143</td>
<td>-</td>
</tr>
<tr>
<td>Projection</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>9</td>
<td>46</td>
<td>581</td>
<td>18182</td>
</tr>
<tr>
<td>Luxburg</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Split</td>
<td>1</td>
<td>2</td>
<td>9</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
## How many repeats needed?

### Unbalance

<table>
<thead>
<tr>
<th>Initialization</th>
<th>CI-value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6</td>
</tr>
<tr>
<td>Rand-P</td>
<td>1</td>
</tr>
<tr>
<td>Rand-C</td>
<td>1</td>
</tr>
<tr>
<td>Maxmin</td>
<td>1</td>
</tr>
<tr>
<td>Kmeans++</td>
<td>1</td>
</tr>
<tr>
<td>Bradley</td>
<td>1</td>
</tr>
<tr>
<td>Sorting</td>
<td>1</td>
</tr>
<tr>
<td>Projection</td>
<td>1</td>
</tr>
<tr>
<td>Luxburg</td>
<td>1</td>
</tr>
<tr>
<td>Split</td>
<td>1</td>
</tr>
</tbody>
</table>
Summary of results

**CI-values**

- K-means: $\text{CI} = 4.5$  15%
- Repeated K-means: $\text{CI} = 2.0$  6%
- Maxmin initialization: $\text{CI} = 2.1$  6%
- Both: $\text{CI} = 0.7$  1%

- Most application: Good enough!
- Accuracy vital: Find better method! (random swap)

- Cluster overlap most important factor
### Effect of different factors

<table>
<thead>
<tr>
<th>Method</th>
<th>Overlap</th>
<th>Clusters</th>
<th>Dimension</th>
<th>Unbalance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rand-P</td>
<td>No effect</td>
<td>Constant</td>
<td>No effect</td>
<td>Very bad</td>
</tr>
<tr>
<td>Rand-C</td>
<td>No effect</td>
<td>Constant</td>
<td>No effect</td>
<td>Very bad</td>
</tr>
<tr>
<td>Maxmin</td>
<td>Bad</td>
<td>Constant</td>
<td>No effect</td>
<td>A bit worse</td>
</tr>
<tr>
<td>kmeans++</td>
<td>A bit worse</td>
<td>Constant</td>
<td>No effect</td>
<td>A bit worse</td>
</tr>
<tr>
<td>Bradley</td>
<td>Good</td>
<td>Constant</td>
<td>No effect</td>
<td>Bad</td>
</tr>
<tr>
<td>Sorting</td>
<td>A bit worse</td>
<td>Constant</td>
<td>No effect</td>
<td>Very bad</td>
</tr>
<tr>
<td>Projection</td>
<td>A bit worse</td>
<td>Constant</td>
<td>No effect</td>
<td>Very bad</td>
</tr>
<tr>
<td>Luxburg</td>
<td>A bit worse</td>
<td>Minor effect</td>
<td>No effect</td>
<td>Very bad</td>
</tr>
<tr>
<td>Split</td>
<td>A bit worse</td>
<td>Constant</td>
<td>No effect</td>
<td>Very bad</td>
</tr>
<tr>
<td>KM iterations</td>
<td>Good</td>
<td>Constant</td>
<td>No effect</td>
<td>No effect</td>
</tr>
</tbody>
</table>
Conclusions

How effective:
• Repeats + Maxmin reduces error $4.5 \rightarrow 0.7$

Is it enough:
• For most applications: YES
• If accuracy important: NO

Important factors:
• Cluster overlap critical for k-means
• Dimensions does not matter
Random swap
Random swap (RS)

Initialize

Swap

K-means

Repeat 5000 times

Only 2 iterations

CI = 0

P. Fränti, "Efficiency of random swap clustering", *Journal of Big Data*, 2018
How many repeats?

P. Fränti, "Efficiency of random swap clustering", Journal of Big Data, 2018
The end