# Hierarchical Clustering

Isabella Cattinelli

*cattinelli@dsi.unimi.it*

2

# Introduction to the HC paradigm

… forget about partitional methods ;)

# What HC is

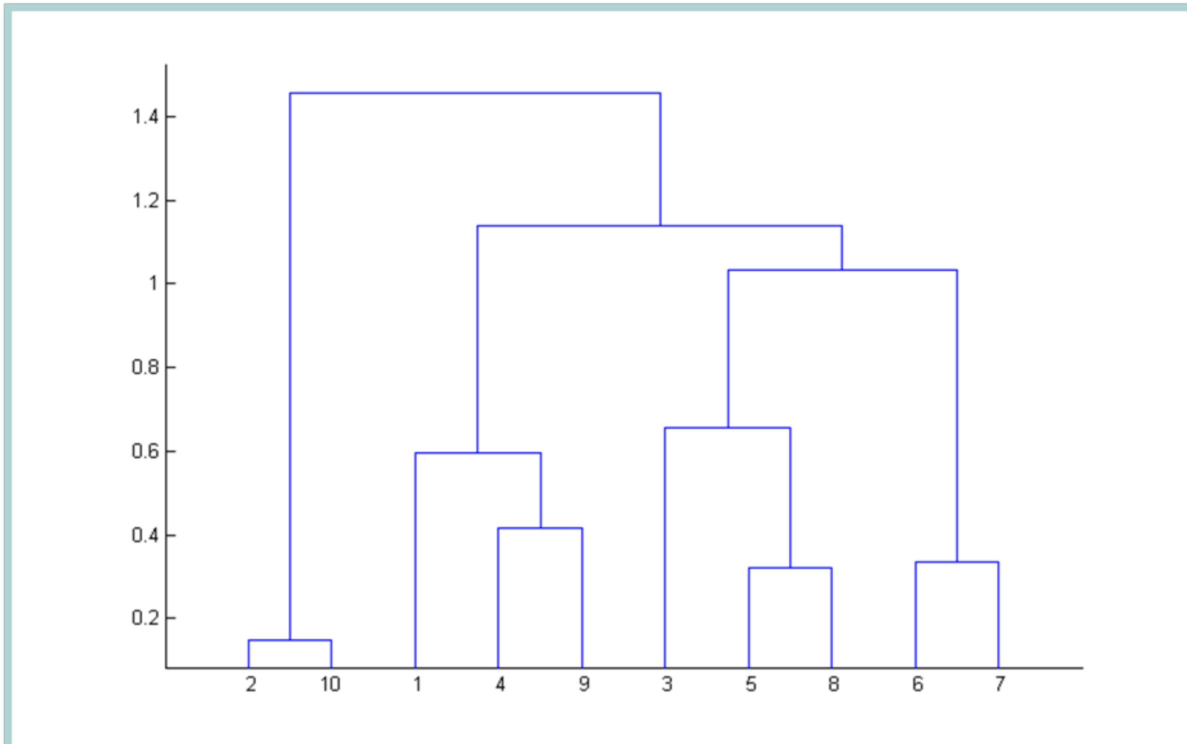
- In brief, HC algorithms build a whole hierarchy of clustering solutions
  - Solution at level k is a *refinement* of solution at level k-1
- Two main classes of HC approaches:
  - Agglomerative: solution at level k is obtained from solution at level k-1 by merging two clusters
  - Divisive: solution at level k is obtained from solution at level k-1 by splitting a cluster into two parts
    - Less used because of computational load

# Agglomerative HC

1. At start, each input pattern is assigned to a singleton cluster
2. At each step, the two *closest* clusters are merged into one
   - So the number of clusters is decreased by one at each step
3. At the last step, only one cluster is obtained

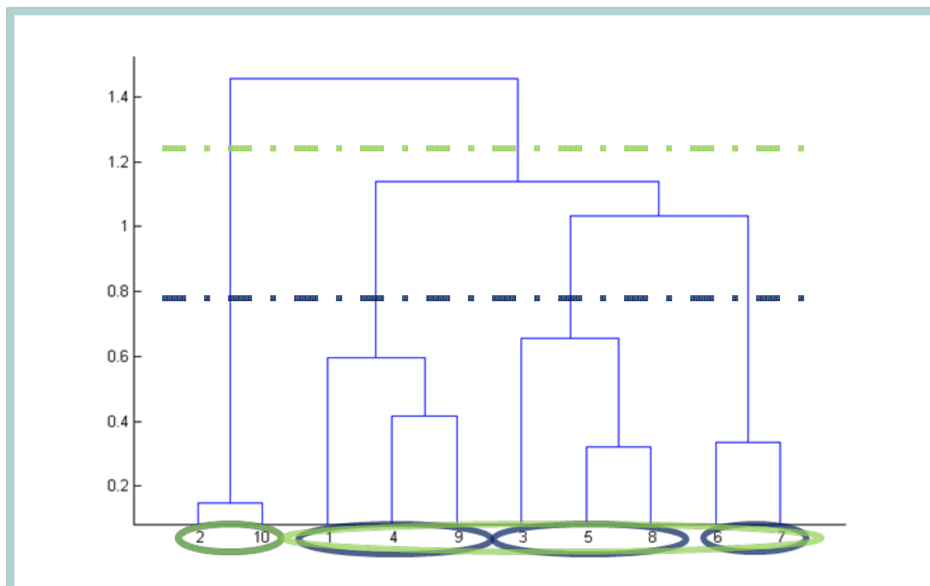
# Dendrograms

- The clustering process is represented by a *dendrogram:*



# Dendrograms

- The resulting dendrogram has to be cut at some level to get the final clustering:
  - Cut criterion: number of desired clusters, or threshold on some features of resulting clusters




6

# Computing dissimilarities

Dissimilarity between pairs of single points

- Different distances/indices of dissimilarity…
  - E.g. euclidean, city-block, correlation…

Dissimilarity between pairs of clusters

- … and agglomeration criteria: Merge clusters $C_i$ and $C_j$ such that *diss(i, j)* is minimum
  - Single linkage:
    - diss(i,j) = min d(x, y), where x is in $C_i$ , y in cluster $C_j$
  - Complete linkage:
    - diss(i,j) = max d(x, y), where x is in cluster $C_i$, y in cluster $C_j$
  - Group Average and Weighted Average Linkage:
    - diss(i, j) = $\sum_{x \in C_i} \sum_{y \in C_j} w_i w_j d(x,y) \Big/ \sum_{x \in C_i} \sum_{y \in C_j} w_i w_j$

GA: $w_i = w_j = 1$
WA: $w_i = n_i$, $w_j = n_j$

# Computing dissimilarities (cont.)

Dissimilarity between pairs of clusters

8

- Other agglomeration criteria: Merge clusters $C_i$ and $C_j$ such that *diss(i, j)* is minimum
  - Centroid Linkage:
    - diss(i, j) = $d(\mu_i, \mu_j)$
  - Median Linkage:
    - diss(i, j) = d($center_i$, $center_j$), where each $center_i$ is the average of the centers of the clusters composing $C_i$
  - Ward's Method:
    - diss(i, j) = increase in the total error sum of squares (ESS) due to the merging of $C_i$ and $C_j$

Squared Euclidean distances should be used

- Single, complete, and average linkage: *graph methods*
  - All points in clusters are considered
- Centroid, median, and Ward's linkage: *geometric methods*
  - Clusters are summed up by their centers

# Ward's criterion

- Also known as minimum variance method
- Each merging step minimizes the increase in the total ESS:

$$ESS_i = \sum_{x \in C_i} (x - \mu_i)^2 \qquad ESS = \sum_i ESS_i$$

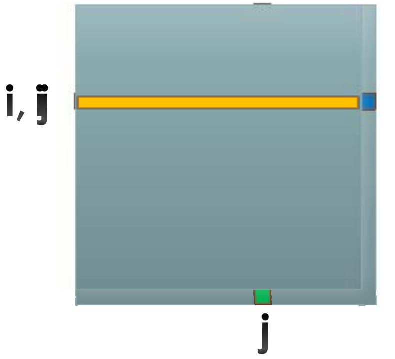
  - When merging clusters $C_i$ and $C_j$, the increase in the total ESS is

$$\Delta ESS = ESS_{i,j} - ESS_i - ESS_j$$

- Spherical, compact clusters are obtained
- The solution at each level k is an <u>approximation</u> to the optimal solution for that level (the one minimizing ESS)

# The dissimilarity matrix

- HC algorithms operate on a dissimilarity matrix:
  - For each pair of existant clusters, their dissimilarity value is stored
- When clusters $C_i$ and $C_j$ are merged, only dissimilarities for the new resulting cluster have to be computed
  - The rest of the matrix is left untouched



# The Lance-Williams formula

- Used for iterative implementation
- The dissimilarity value between newly formed cluster $\{C_i, C_j\}$ and every other cluster $C_k$ is computed as

$$diss(k,(i,j)) = \alpha_i\, diss(k,i) + \alpha_j\, diss(k,j) + \beta\, diss(i,j) + \\ + \gamma \left| diss(k,i) - diss(k,j) \right|$$

- Only values already stored in the dissimilarity matrix are used
- Different sets of coefficients correspond to different criteria

# The Lance-Williams formula - coefficients



$$diss(k,(i,j)) = \alpha_i\, diss(k,i) + \alpha_j\, diss(k,j) + \beta\, diss(i,j) + \gamma \left| diss(k,i) - diss(k,j) \right|$$

| Criterion | $\alpha_i$ | $\alpha_j$ | $\beta$ | $\gamma$ |
|---|---|---|---|---|
| Single Link. | ½ | ½ | 0 | -½ |
| Complete Link. | ½ | ½ | 0 | ½ |
| Group Avg. | $n_i/(n_i+n_j)$ | $n_j/(n_i+n_j)$ | 0 | 0 |
| Weighted Avg. | ½ | ½ | 0 | 0 |
| Centroid | $n_i/(n_i+n_j)$ | $n_j/(n_i+n_j)$ | $-n_in_j/(n_i+n_j)^2$ | 0 |
| Median | ½ | ½ | - ¼ | 0 |
| Ward | $(n_i+n_k)/(n_i+n_j+n_k)$ | $(n_j+n_k)/(n_i+n_j+n_k)$ | $-n_k/(n_i+n_j+n_k)$ | 0 |

e.g. for single linkage…
diss(k, (i,j)) = min(diss(k, i), diss(k, j))

# The Lance-Williams Formula and Single Linkage

| Criterion | $\alpha_i$ | $\alpha_j$ | $\beta$ | $\gamma$ |
|---|---|---|---|---|
| Single Linkage | ½ | ½ | 0 | -½ |

$$diss(k,(i,j)) = \alpha_i\, diss(k,i) + \alpha_j\, diss(k,j) + \beta\, diss(i,j) + \\ + \gamma \left| diss(k,i) - diss(k,j) \right|$$



diss(k, i) > diss(k, j)

$$diss(k,(i,j)) = \frac{1}{2}\cancel{diss(k,i)} + \frac{1}{2} diss(k,j) + \\ -\frac{1}{2}\cancel{diss(k,i)} + \frac{1}{2} diss(k,j)$$

$$diss(k,(i,j)) = \frac{1}{2} diss(k,i) + \frac{1}{2} diss(k,j) + \\ -\frac{1}{2}\left| diss(k,i) - diss(k,j) \right|$$

diss(k, i) < diss(k, j)

$$diss(k,(i,j)) = \frac{1}{2} diss(k,i) + \frac{1}{2}\cancel{diss(k,j)} + \\ -\frac{1}{2}\cancel{diss(k,j)} + \frac{1}{2} diss(k,i)$$

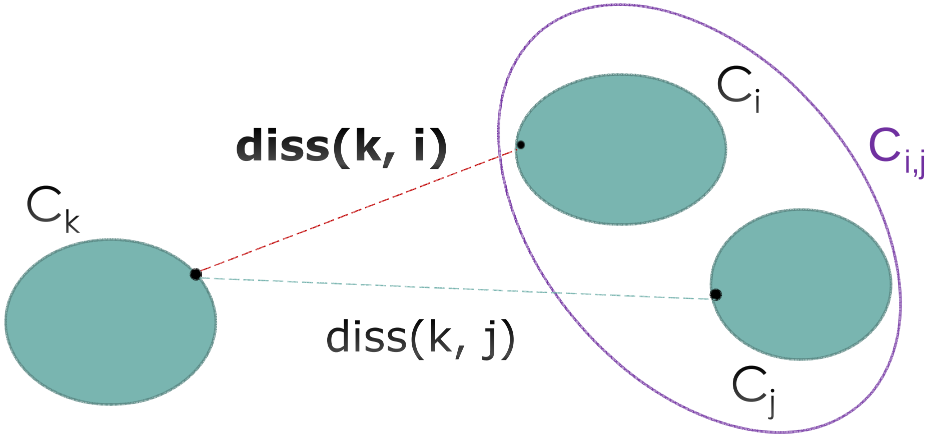
… and therefore: diss(k, (i,j)) = min(diss(k, i), diss(k, j))

# The Lance-Williams Formula and Single Linkage

… and therefore:

diss(k, (i,j)) = min(diss(k, i), diss(k, j))



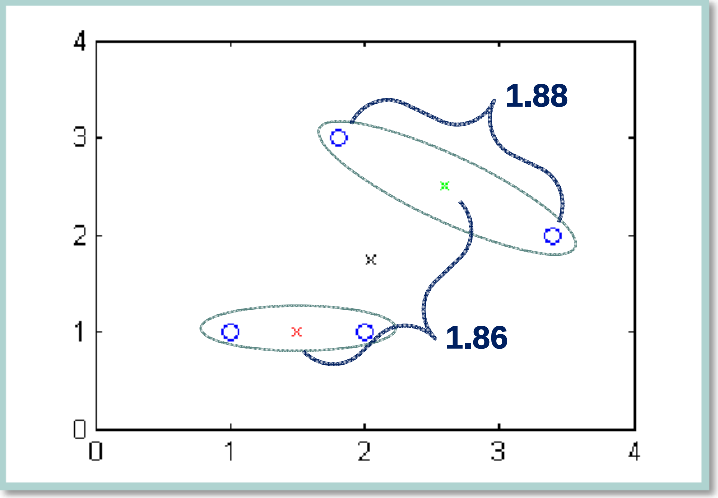


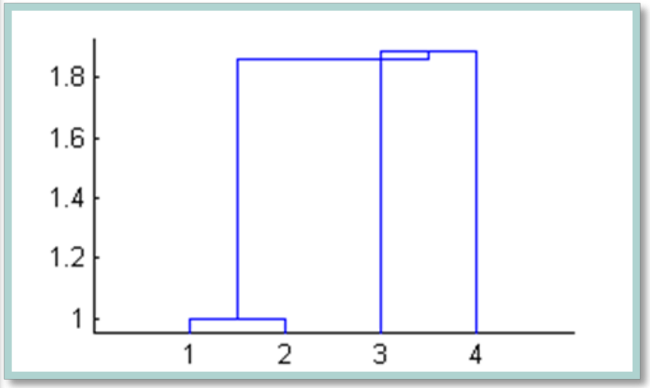
# Pros and cons of HC algorithms

- Pros:
  - Indipendence from initialization
  - No need to specify a desired number of clusters from the beginning
- Cons:
  - Computational complexity at least $O(N^2)$
  - Sensitivity to outliers
  - No reconsideration of possibly misclassified points
  - Possibility of inversion phenomena and multiple solutions

# Inversions

- We have an inversion when the sequence of dissimilarity values selected by the HC algorithm is nonmonotonic

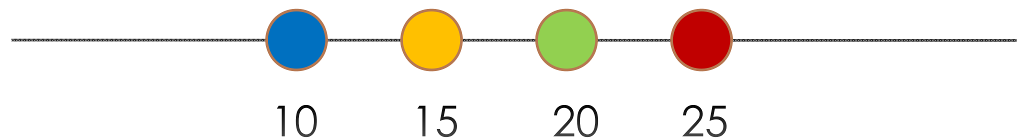



- Inversions may be produced when using the centroid or the median criterion

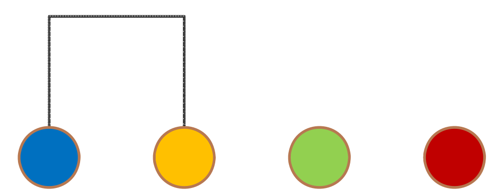
# Non-uniqueness

- « This problem "certainly is not *widely known*" »
  (van der Kloot et al., 2005)




Dissimilarity matrix

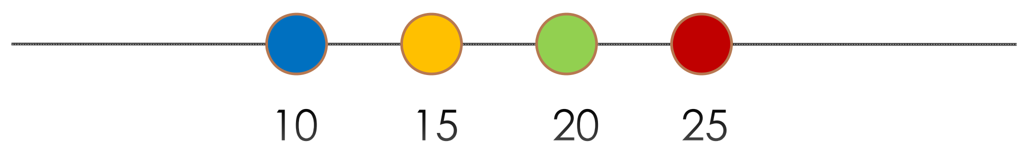
| | | | |
|---|---|---|---|
| | 5 | 10 | 15 |
| | | 5 | 10 |
| | | | 5 |



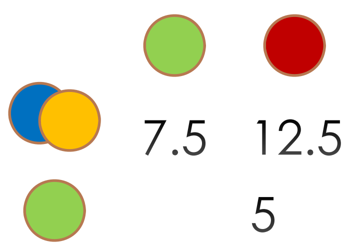
# Non-uniqueness

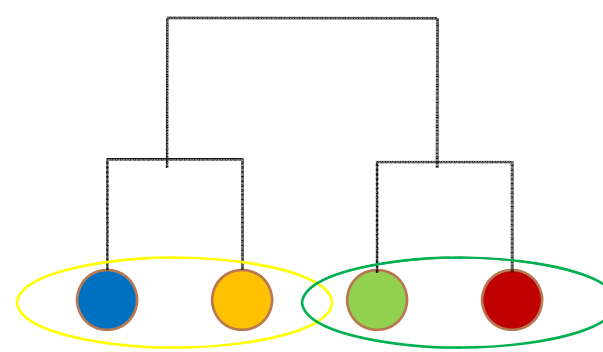
- « This problem "certainly is not *widely known*" »
  (van der Kloot et al., 2005)





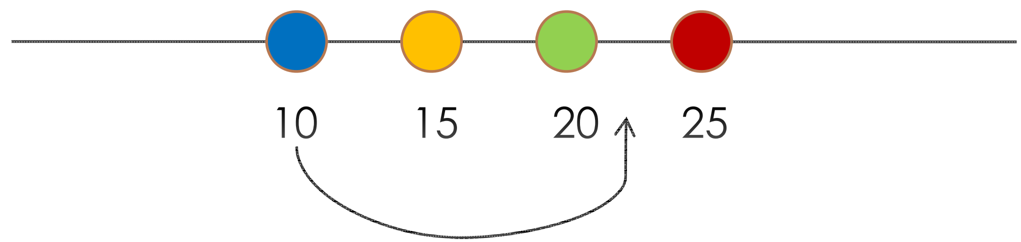

Dissimilarity matrix



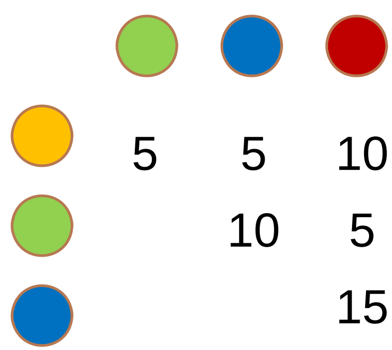



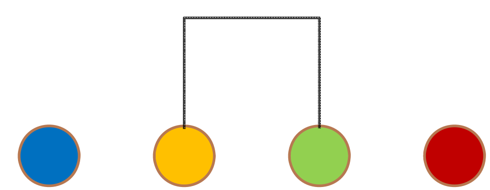
# Non-uniqueness

- « This problem "certainly is not *widely known*" »
  (van der Kloot et al., 2005)




Dissimilarity matrix

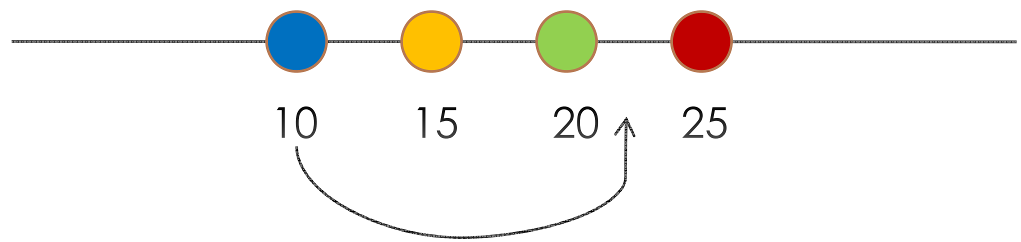


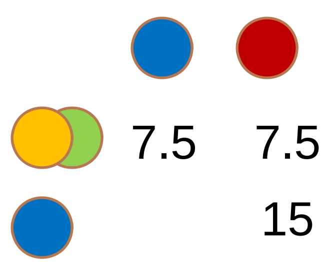
# Non-uniqueness

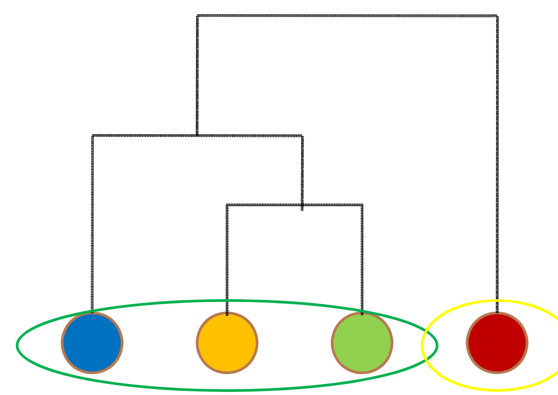
- « This problem "certainly is not *widely known*" »
  (van der Kloot et al., 2005)






Dissimilarity matrix



# Non-uniqueness

To make a long story short:

*Different permutations of the input data*

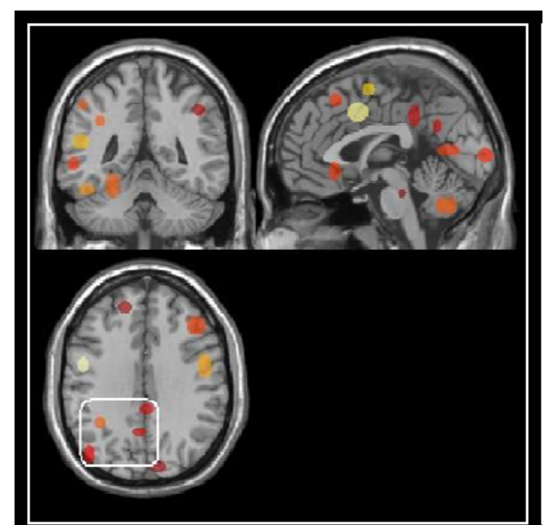
*can produce different clustering solutions!*



- More than one pair of objects having minimum distance: ties
- The first one according to the given input order is selected
  - In other words, the non-uniqueness problem is usually not taken into account, but:
- It is highly desirable to have a unique clustering solution for the same dataset!
  - Replicability of results
  - Different solutions may lead to different conclusions

# Non-uniqueness: effects

- Example of application: metanalysis of neuroimaging data
  - Input: activation coordinates on the cerebral volume
  - Output: set of clusters whose functional role has to be determined
  - Running an HC algorithm on a real dataset actually produced different solutions depending on input data order!



23

# A quest for uniqueness

Work in progress…

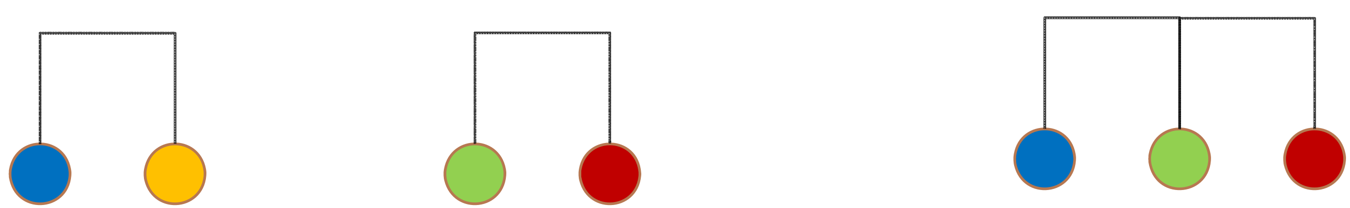
# Quest for uniqueness: first approach



- Given a set of minimal distance pairs, select for merging the “best” one
  - How to define best?
- Greedy approach: the choice of the best pair at step *k* does not guarantee the solution to be the best one overall
- Note: we are not really interested in the quality of the whole dendrogram
  - We want the final clustering after cutting the dendrogram to be the best one!

# Quest for uniqueness: second approach

- Let us develop all the possible dendrograms for a dataset, and look for the best solution they provide
  - At each step, for each minimal distance pair, we generate the dendrogram resulting from the choice of that pair
  - But we have a slight problem here… can you guess what it is???
- Note: not all minimal distance pairs are equal
  - Some are *critical*, some are not



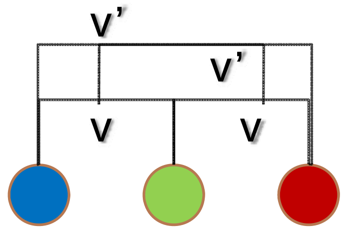
Noncritical pairs

Critical pairs

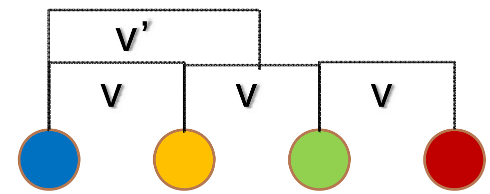
# Quest for uniqueness: third approach

- Let us develop ~~all the possible~~ all *significantly different* dendrograms for a dataset, and look for the best solution they provide
  - At each step, for each critical pair, we generate the dendrogram resulting from the choice of that pair
  - First, noncritical pairs are merged, in a random order
- The number of dendrograms to be handled drops…
  - … but not enough!
  - E.g. on a dataset of about 1200 points, after 100,000 dendrograms (and a couple of days of computing) MATLAB ran out of memory

# All critical pairs are equal, but some critical pairs are more equal than others



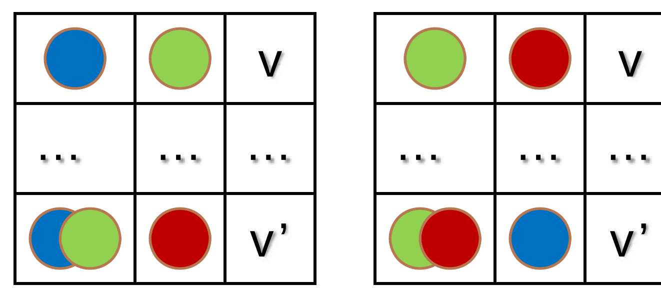

Equivalent pairs




Nonequivalent pairs



- Equivalent pairs produce equivalent trees…
- How to check for equivalence?
  - If in both scenarios, the closest point to the new cluster is the excluded extreme (and vice versa), the two pairs are equivalent



# Quest for uniqueness: fourth approach (hopefully, the last one)



- Let us develop ~~all the possible~~ ~~all significantly different~~ *all nonequivalent* dendrograms for a dataset, and look for the best solution they provide
  - At each step, for each nonequivalent pair, we generate the dendrogram resulting from the choice of that pair
  - First, noncritical pairs are merged, in a random order
- Finally, the problem seems treatable!
  - E.g. we go from an out of memory failure to the production of 128 dendrograms
  - Note: we get something more than just nonequivalent dendrograms (due to some extreme configuration of data)

# Quest for uniqueness: finding the best solution

- After getting the set of nonequivalent dendrograms, we cut all of them using the same criterion
  - And we get the corresponding final clusterings, one for each dendrogram
- We define the best clustering to be the one having maximum between-cluster variance:

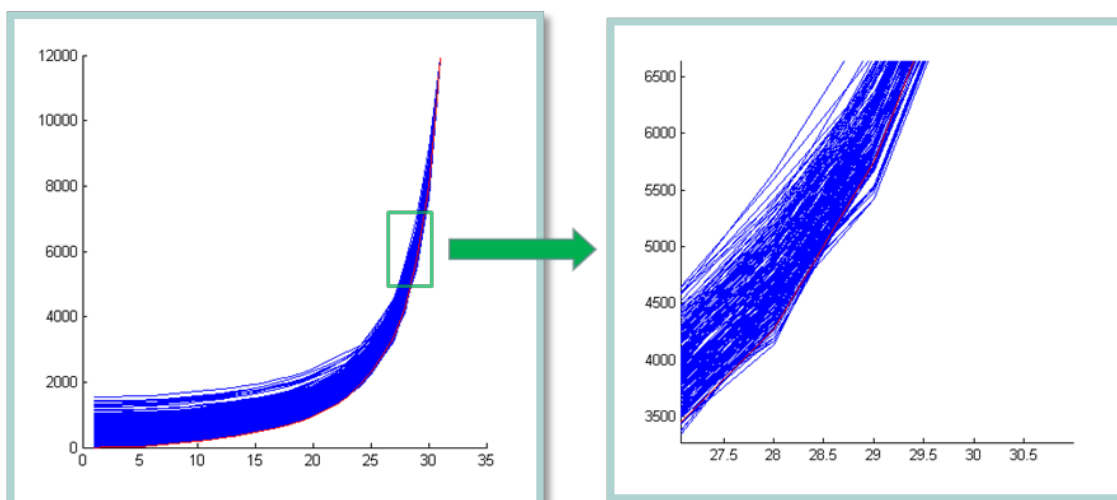
$$bcv = \sum_i n_i (m_i - M)^2$$

$n_i$ = cardinality of cluster $C_i$
$m_i$ = mean of cluster $C_i$
$M$ = grand mean

  - … which means that clusters are well-separated
- Therefore the whole process gives us a unique clustering, independent from input order, up to equivalences

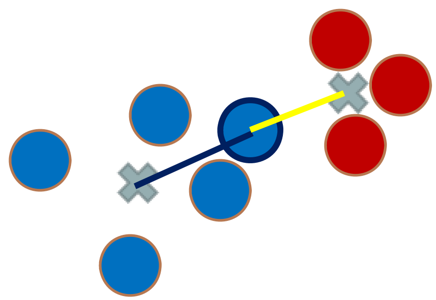
# A quest for optimality???

- HC algorithms are not optimal
  - We would like to have a method that gives us a hierarchy of partitions $P_k$, each of them optimal wrt the objective function (e.g. for Ward's method, $V(P_k) = \Sigma_{i=1\ldots k} ESS_i$)
  - But even if the single merging steps are optimal, the resulting partitions are not necessarily optimal



# A quest for optimality???

- HC algorithms can produce misclassifications:



The marked blue point is closer to the centroid of the red cluster than to the centroid of the blue cluster it belongs to

- These may be corrected by employing k-means as a postprocessing step…
  - Starting from the clusters produced by the HC algorithm, each point is reconsidered and possibly moved to the "right" cluster (the one whose centroid is closest to the point)
- … but the resulting solution is still not guaranteed to be optimal.

# A quest for optimality???



- Is it possible to design a truly optimal clustering algorithm?
  - No, exhaustive enumeration of all possible partitions is not an admissible answer ;)

# References


- Reviews:
  - R.M. Cormack. *A review of classification*. Journal of the Royal Statistical Society 134(3): 321-367, 1971.
  - F. Murtagh. *A Survey of Recent Advances in Hierarchical Clustering Algorithms*. The computer Journal 26(4):354-359, 1983.
  - R. Xu and D.C.Wunsch. *Clustering*. Wiley, 2008.
- J. H. Jr.Ward. *Hierarchical grouping to optimize an objective function*. Journal of the American StatisticalAssociation, 58:236–244, 1963.
- B. J. T. Morgan and A. P. G. Ray. *Non-uniqueness and inversions in cluster analysis*. Applied Statistics,44(1):117–134, 1995.

- For running HC algorithms in MATLAB: *linkage.m* in stats toolbox