

Clustering

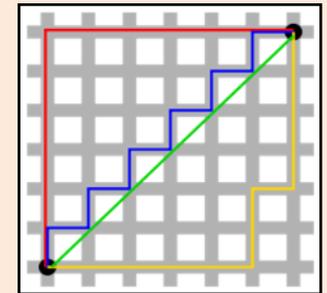
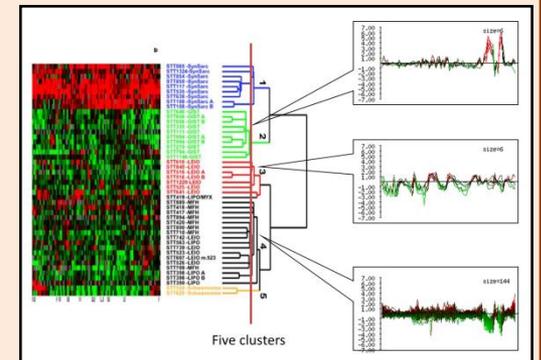
k-mean clustering

Genome 559: Introduction to Statistical and
Computational Genomics

Elhanan Borenstein

A quick review

- The clustering problem:
 - partition genes into distinct sets with high homogeneity and high separation
 - Different representations
- Homogeneity vs Separation
- Many possible distance metrics
- Method matters; metric matters; definitions matter;
- One problem, numerous solutions

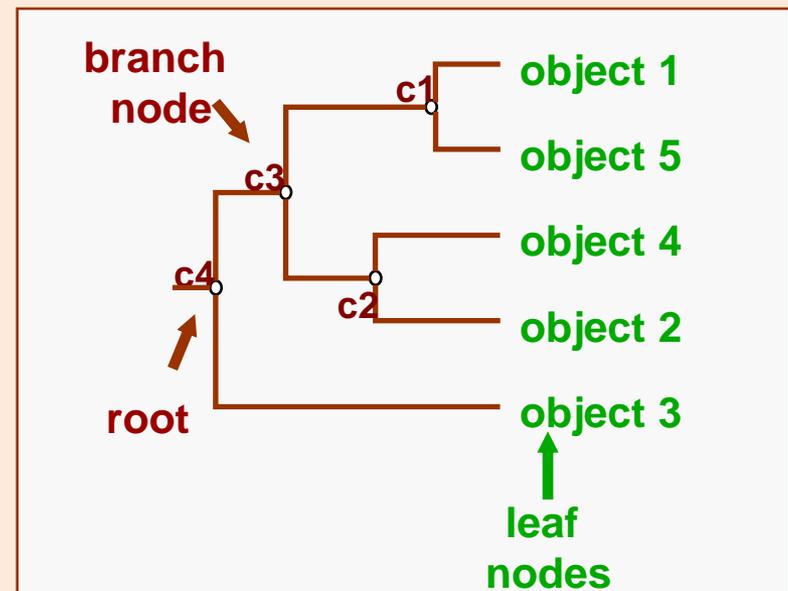


A quick review

- **Hierarchical clustering: An agglomerative method**
 - Takes as input a distance matrix
 - Progressively regroups the closest objects/groups
 - The result is a tree - intermediate nodes represent clusters
 - Branch lengths represent distances between clusters

Distance matrix

	object 1	object 2	object 3	object 4	object 5
object 1	0.00	4.00	6.00	3.50	1.00
object 2	4.00	0.00	6.00	2.00	4.50
object 3	6.00	6.00	0.00	5.50	6.50
object 4	3.50	2.00	5.50	0.00	4.00
object 5	1.00	4.50	6.50	4.00	0.00

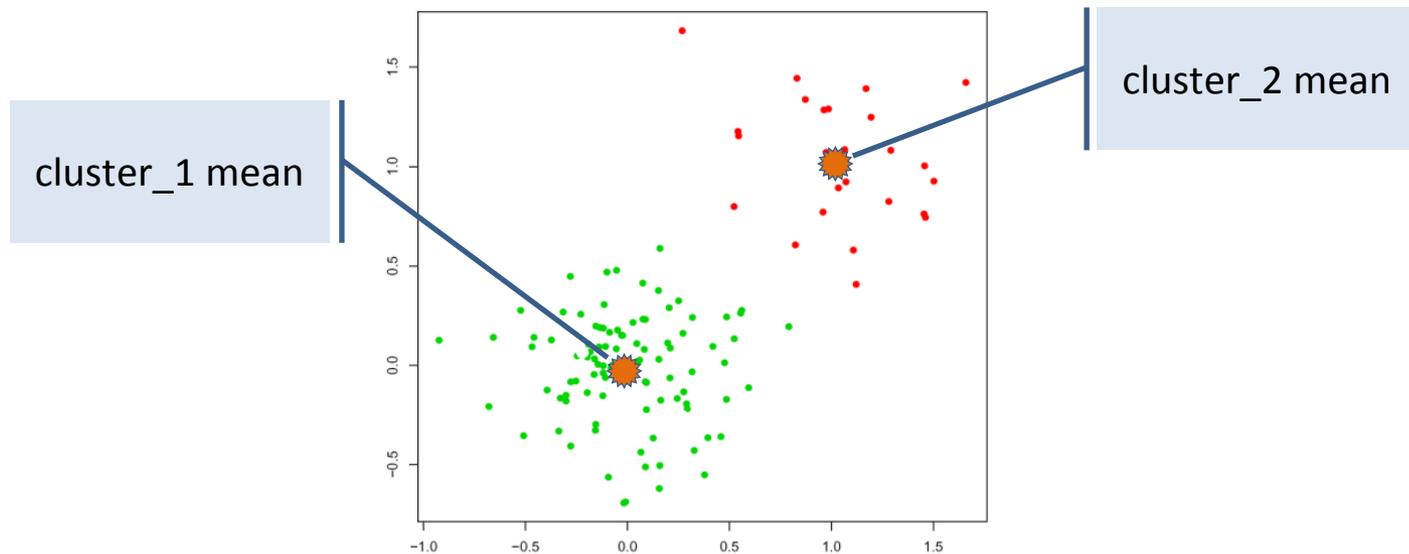


K-mean clustering

Divisive (vs. Agglomerative)

K-mean clustering

- An algorithm for partitioning n observations/points into k clusters such that each observation belongs to the cluster with the nearest mean/center



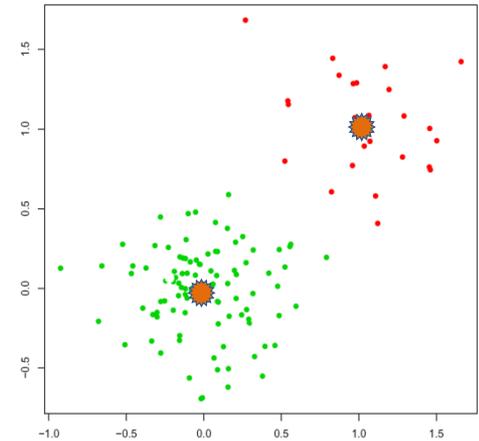
K-mean clustering: Chicken and egg

- An algorithm for partitioning n observations/points into k clusters such that each observation belongs to the cluster with the nearest mean/center

- **The chicken and egg problem:**

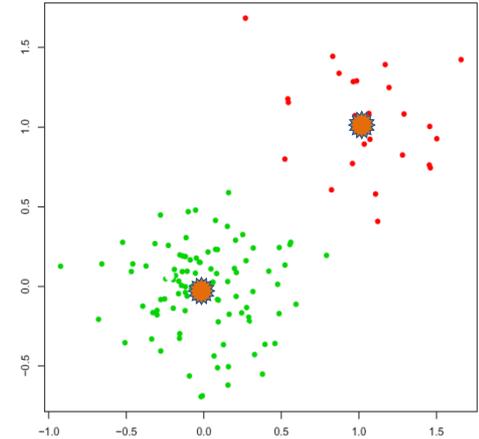
I do not know the means before I determine the partitioning into clusters

I do not know the partitioning into clusters before I determine the means



K-mean clustering: Chicken and egg

- An algorithm for partitioning n observations/points into k clusters such that each observation belongs to the cluster with the nearest mean/center



- **The chicken and egg problem:**

I do not know the means before I determine the partitioning into clusters

I do not know the partitioning into clusters before I determine the means

- **Key principle - cluster around mobile centers:**

- Start with some **random** locations of means/centers, partition into clusters according to these centers, and then correct the centers according to the clusters

[similar to EM (expectation-maximization) algorithms]

K-mean clustering algorithm

- The number of centers, k , has to be specified a-priori
- **Algorithm:**
 1. Arbitrarily select k initial centers
 2. Assign each element to the closest center
 3. Re-calculate centers (mean position of the assigned elements)
 4. Repeat 2 and 3 until ...

K-mean clustering algorithm

- The number of centers, k , has to be specified a-priori

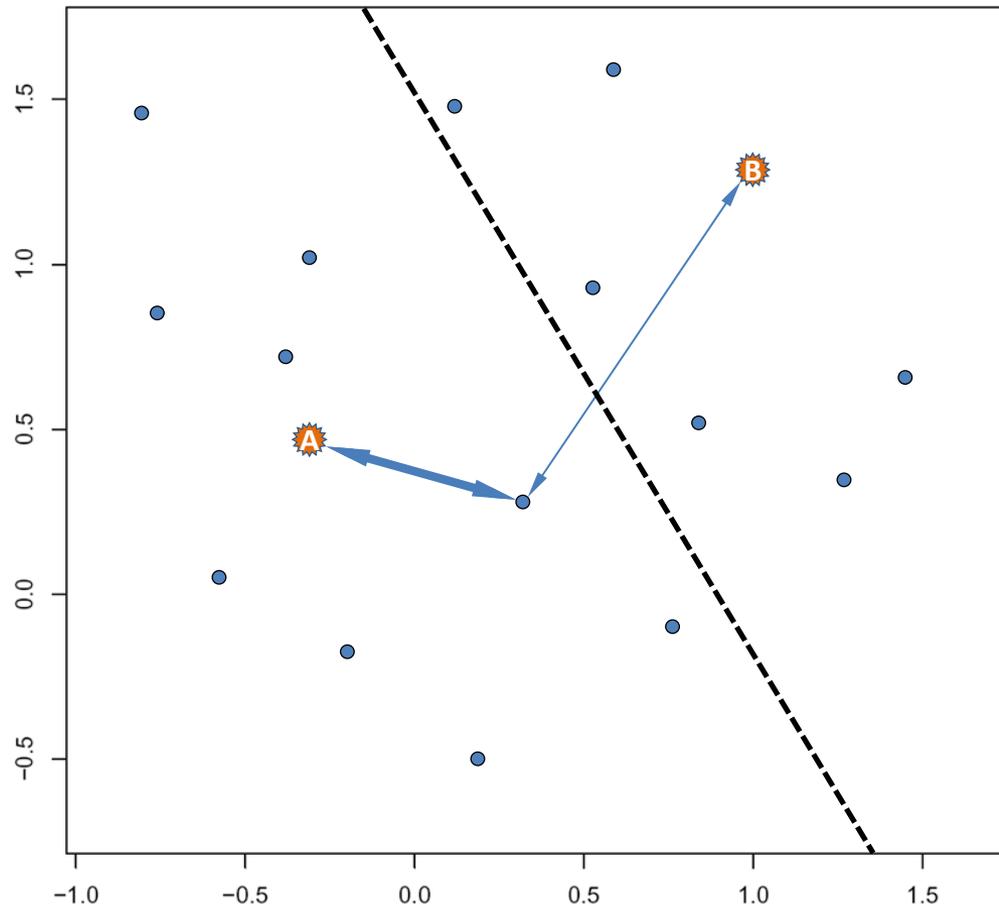
- **Algorithm:**

1. Arbitrarily select k initial centers
2. Assign each element to the closest center
3. Re-calculate centers (mean position of the assigned elements)
4. Repeat 2 and 3 until one of the following termination conditions is reached:
 - i. The clusters are the same as in the previous iteration
 - ii. The difference between two iterations is smaller than a specified threshold
 - iii. The maximum number of iterations has been reached

How can we do this efficiently?

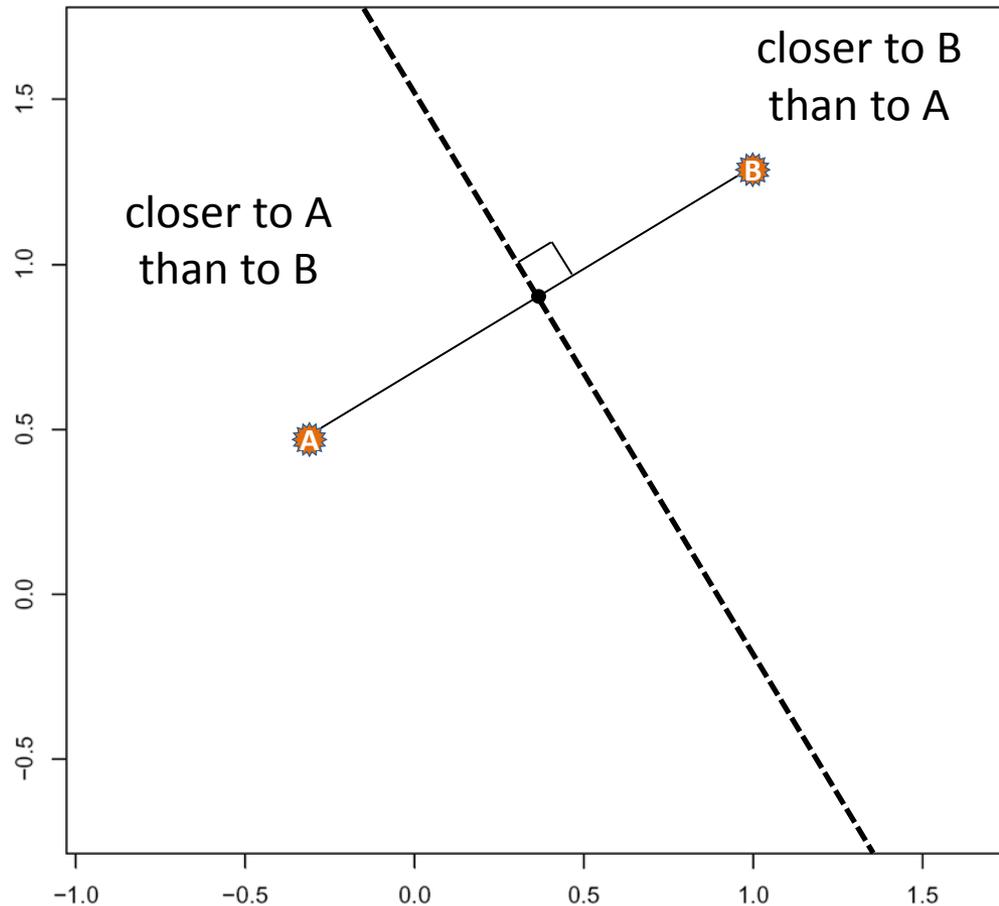
Partitioning the space

- Assigning elements to the closest center



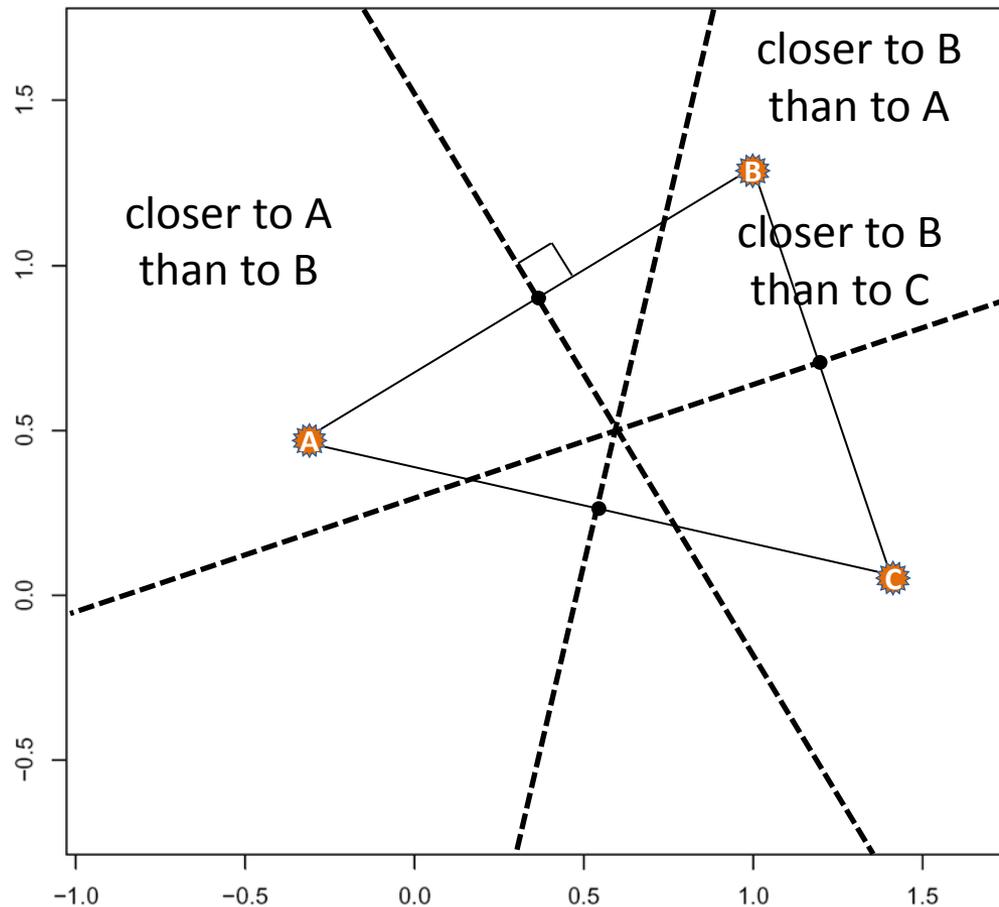
Partitioning the space

- Assigning elements to the closest center



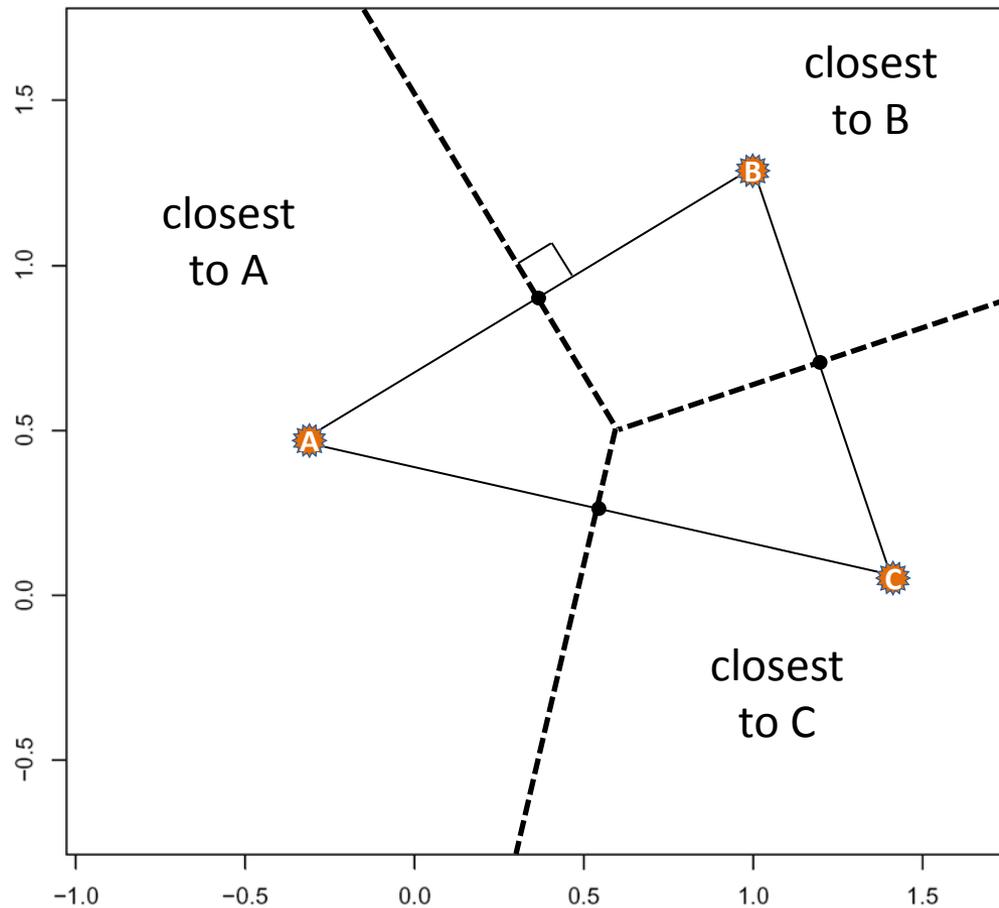
Partitioning the space

- Assigning elements to the closest center



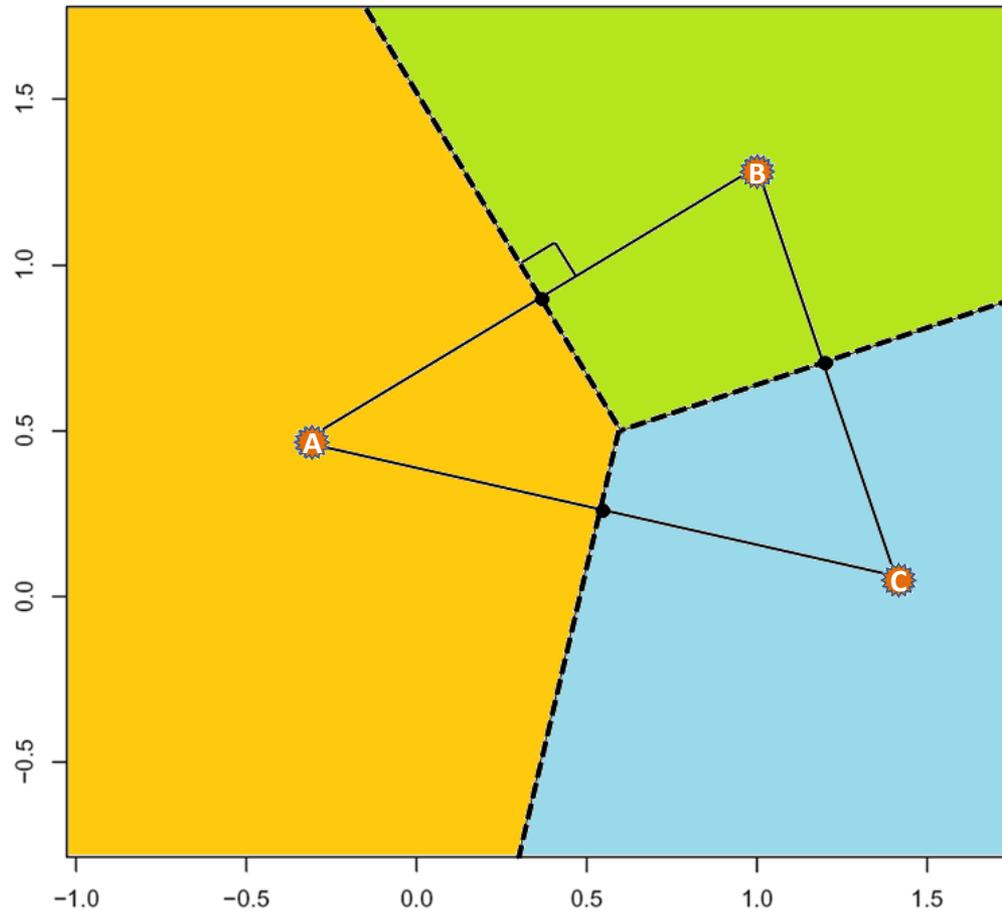
Partitioning the space

- Assigning elements to the closest center



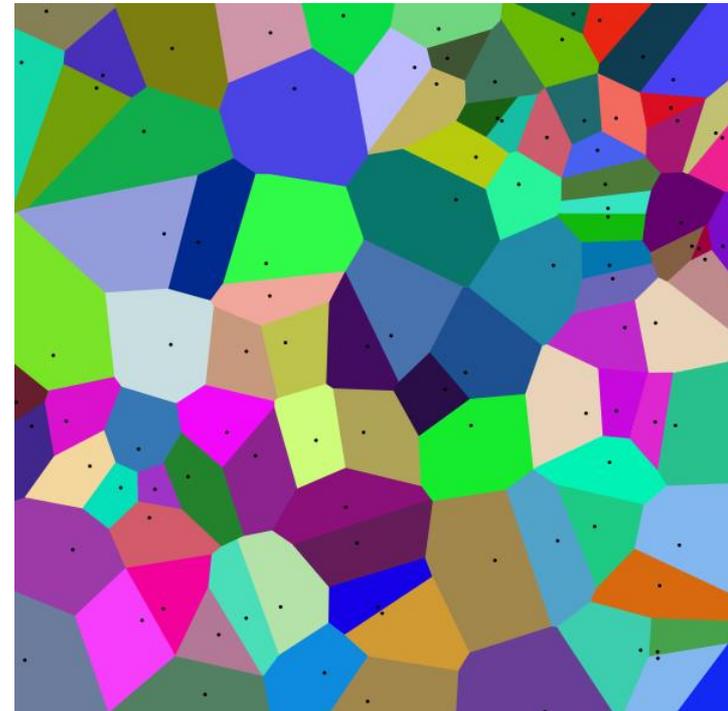
Partitioning the space

- Assigning elements to the closest center



Voronoi diagram

- Decomposition of a metric space determined by distances to a specified discrete set of “centers” in the space
- Each colored cell represents the collection of all points in this space that are closer to a specific center s than to any other center
- Several algorithms exist to find the Voronoi diagram.



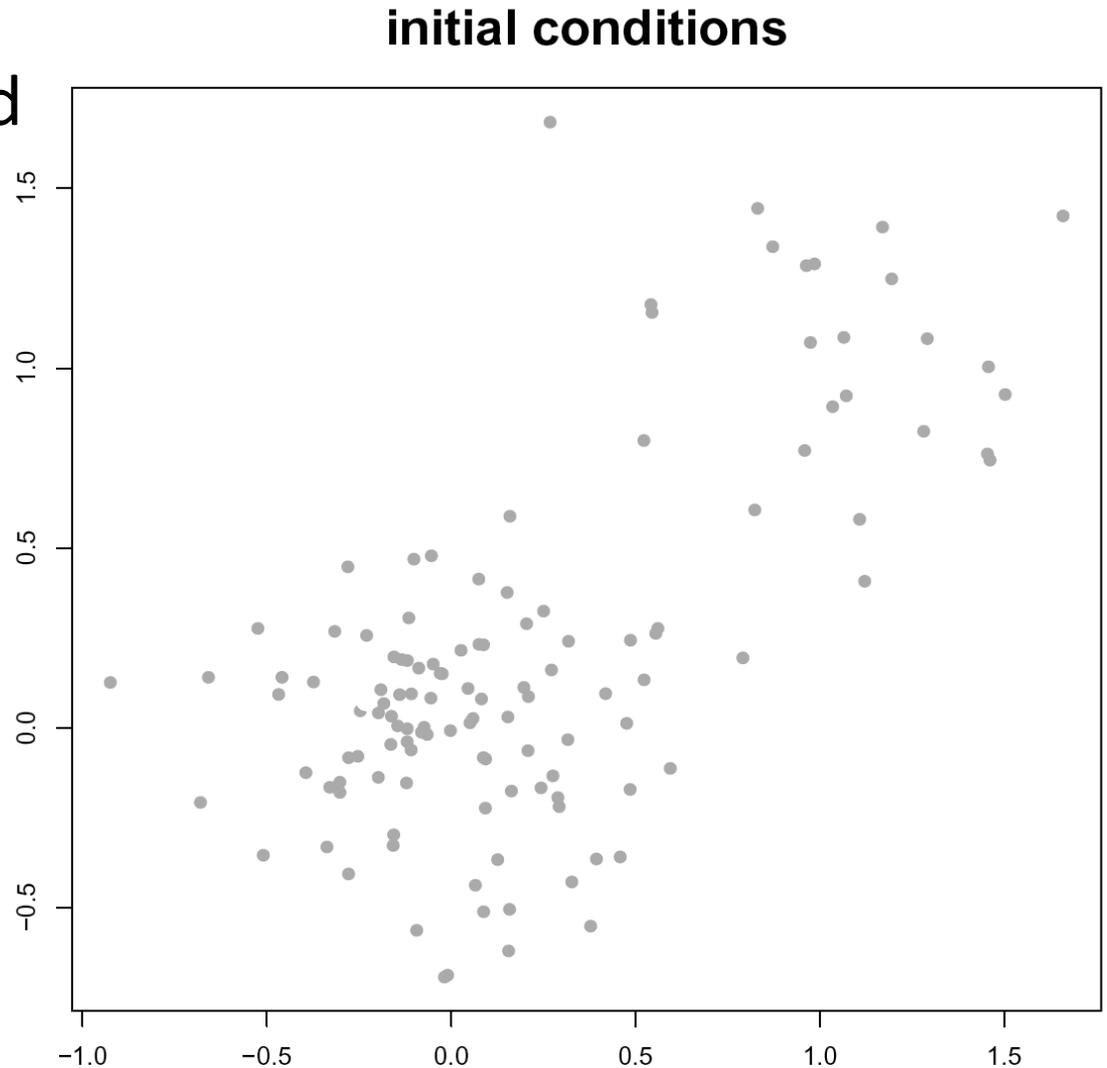
K-mean clustering algorithm

- The number of centers, k , has to be specified a priori
- **Algorithm:**
 1. Arbitrarily select k initial centers
 2. Assign each element to the closest center (**Voronoi**)
 3. Re-calculate centers (mean position of the assigned elements)
 4. Repeat 2 and 3 until one of the following termination conditions is reached:
 - i. The clusters are the same as in the previous iteration
 - ii. The difference between two iterations is smaller than a specified threshold
 - iii. The maximum number of iterations has been reached

K-mean clustering example

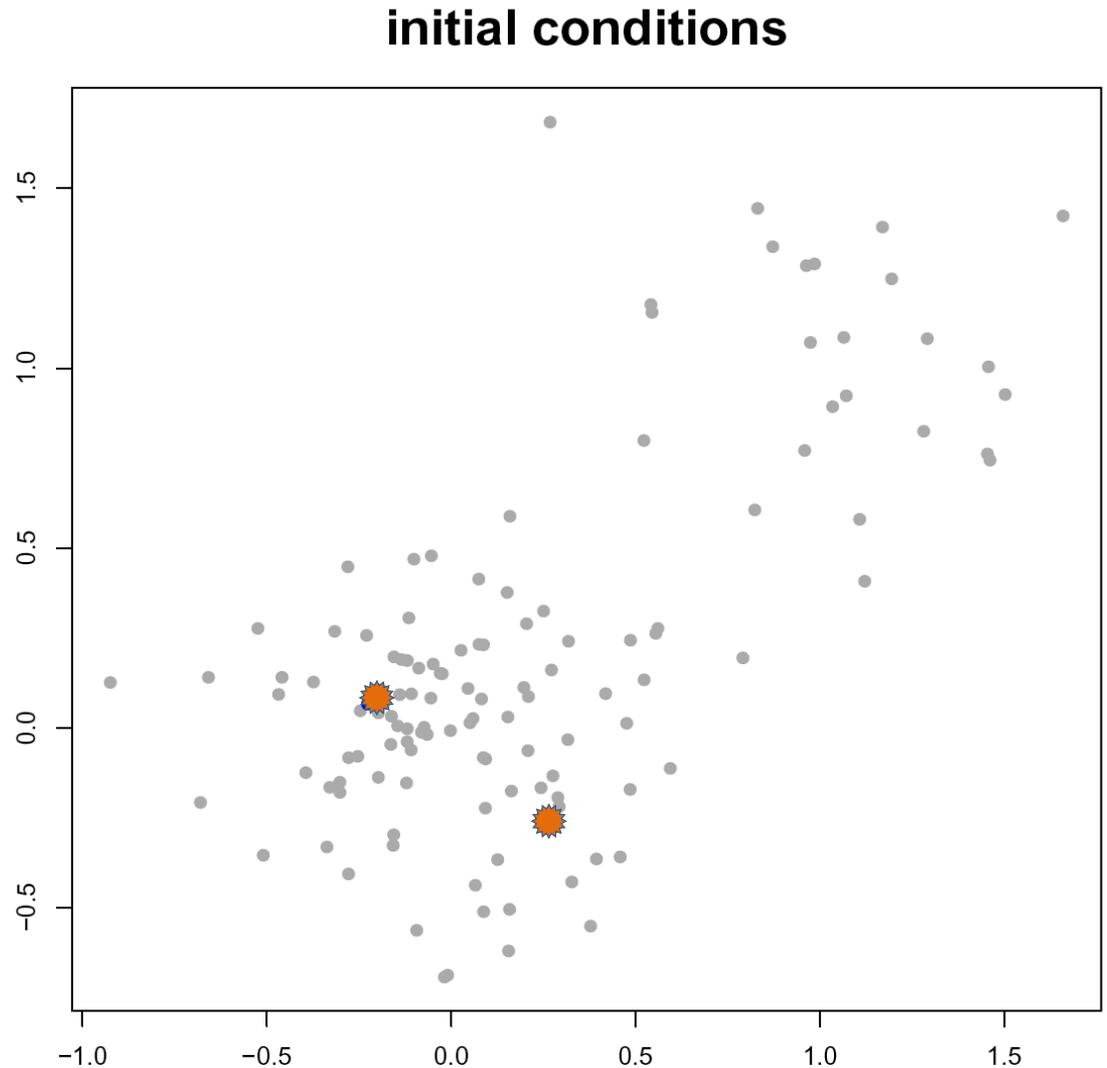
- Two sets of points randomly generated

- 200 centered on (0,0)
- 50 centered on (1,1)



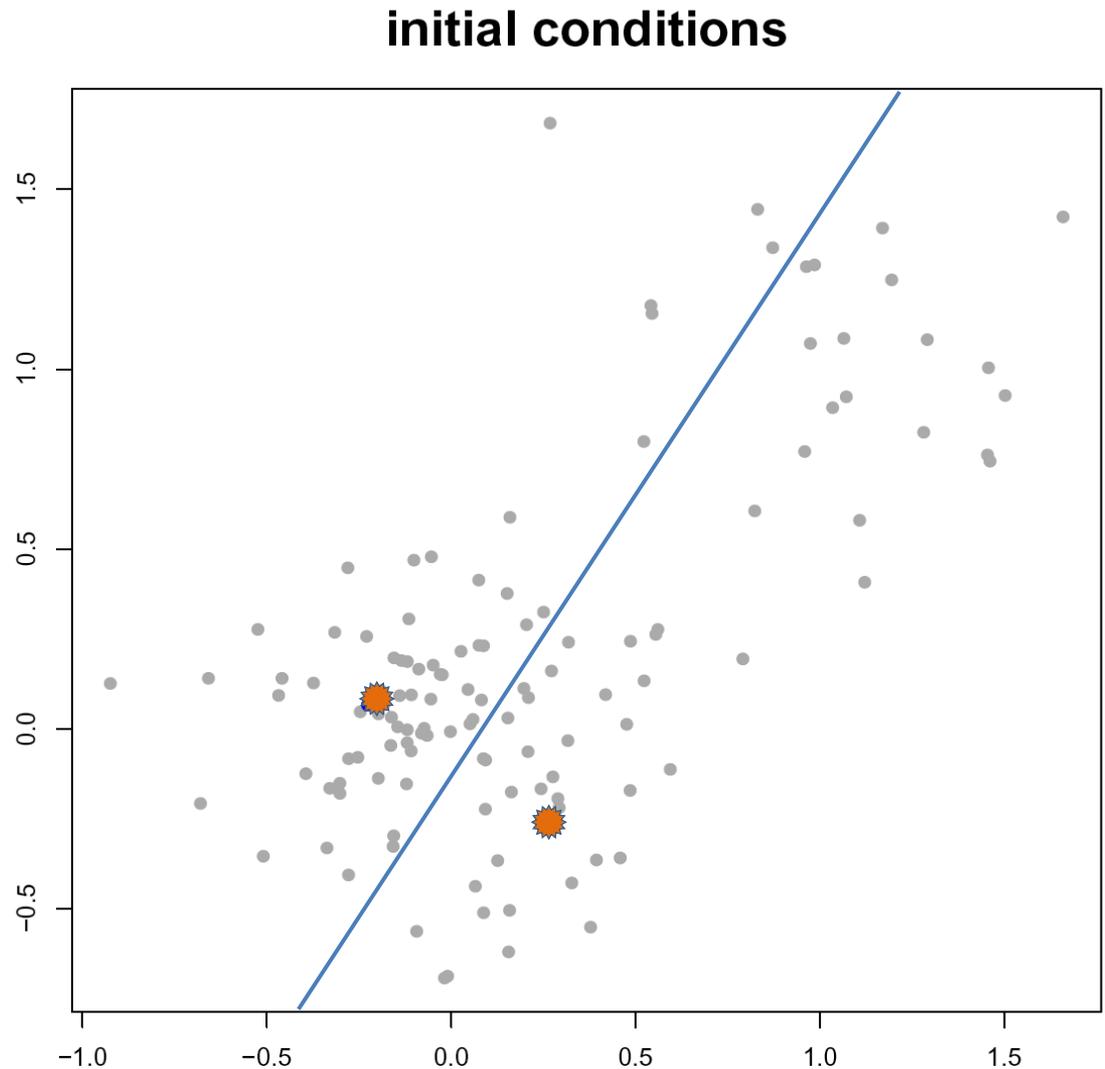
K-mean clustering example

- Two points are randomly chosen as centers (stars)



K-mean clustering example

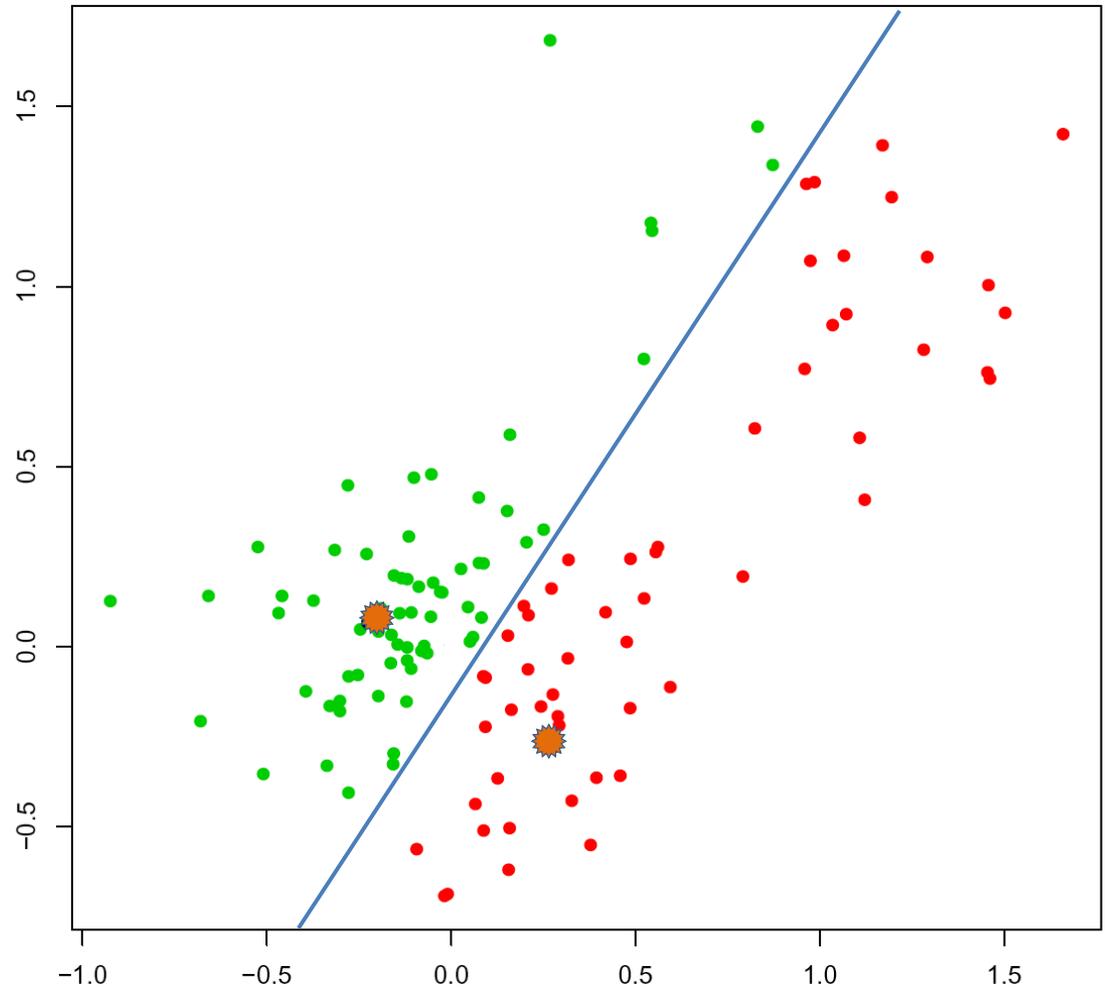
- Each dot can now be assigned to the cluster with the closest center



K-mean clustering example

- First partition into clusters

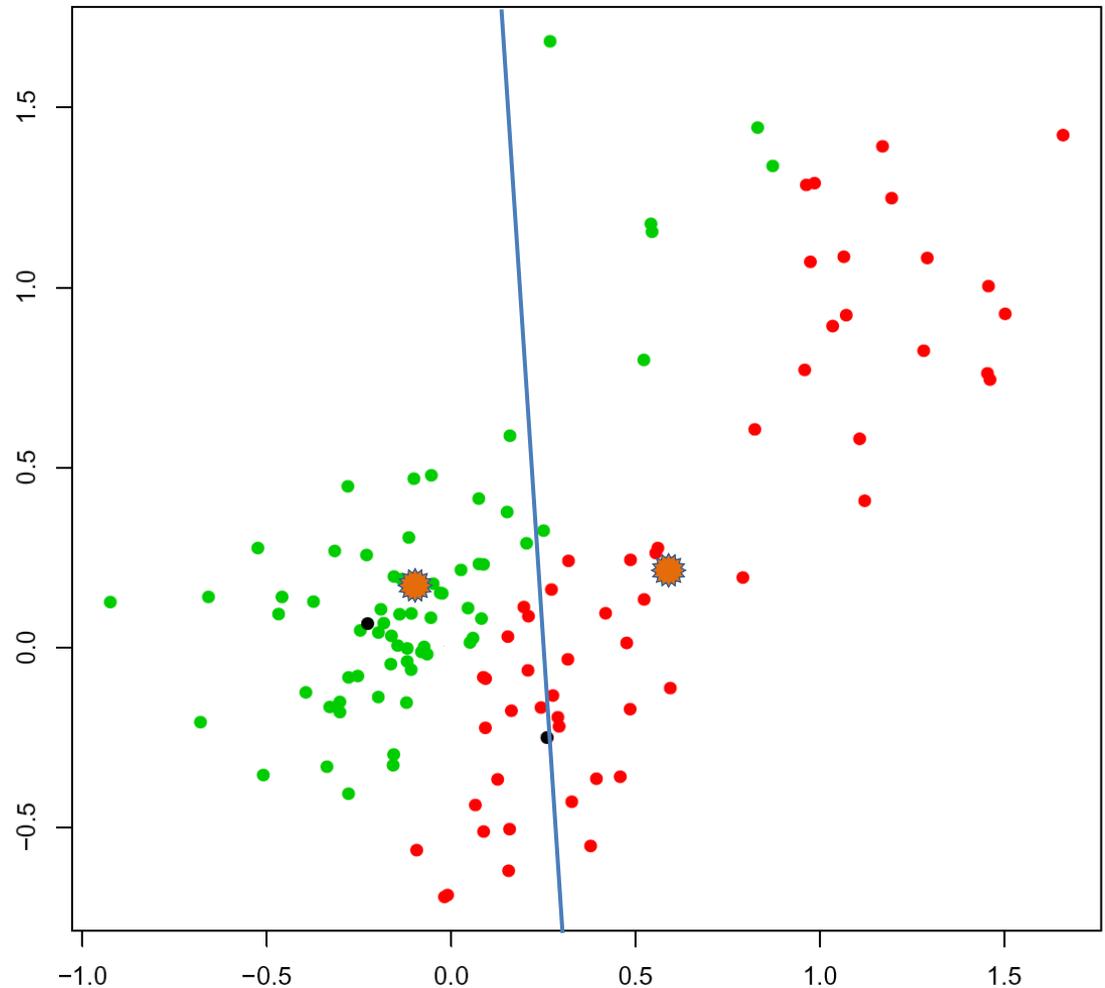
iter.max = 1 ; iterations = 1



K-mean clustering example

- And are again used to partition the points

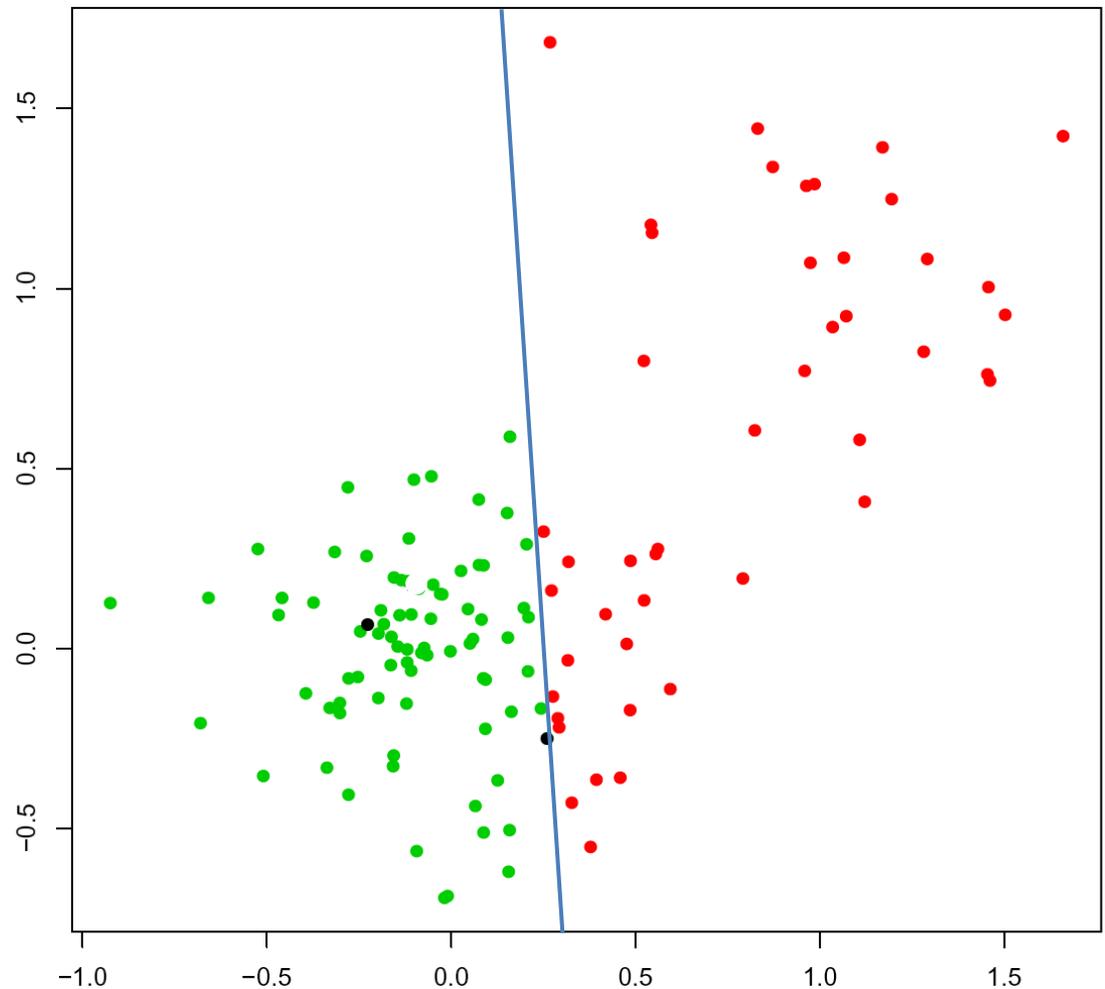
iter.max = 1 ; iterations = 1



K-mean clustering example

- Second partition into clusters

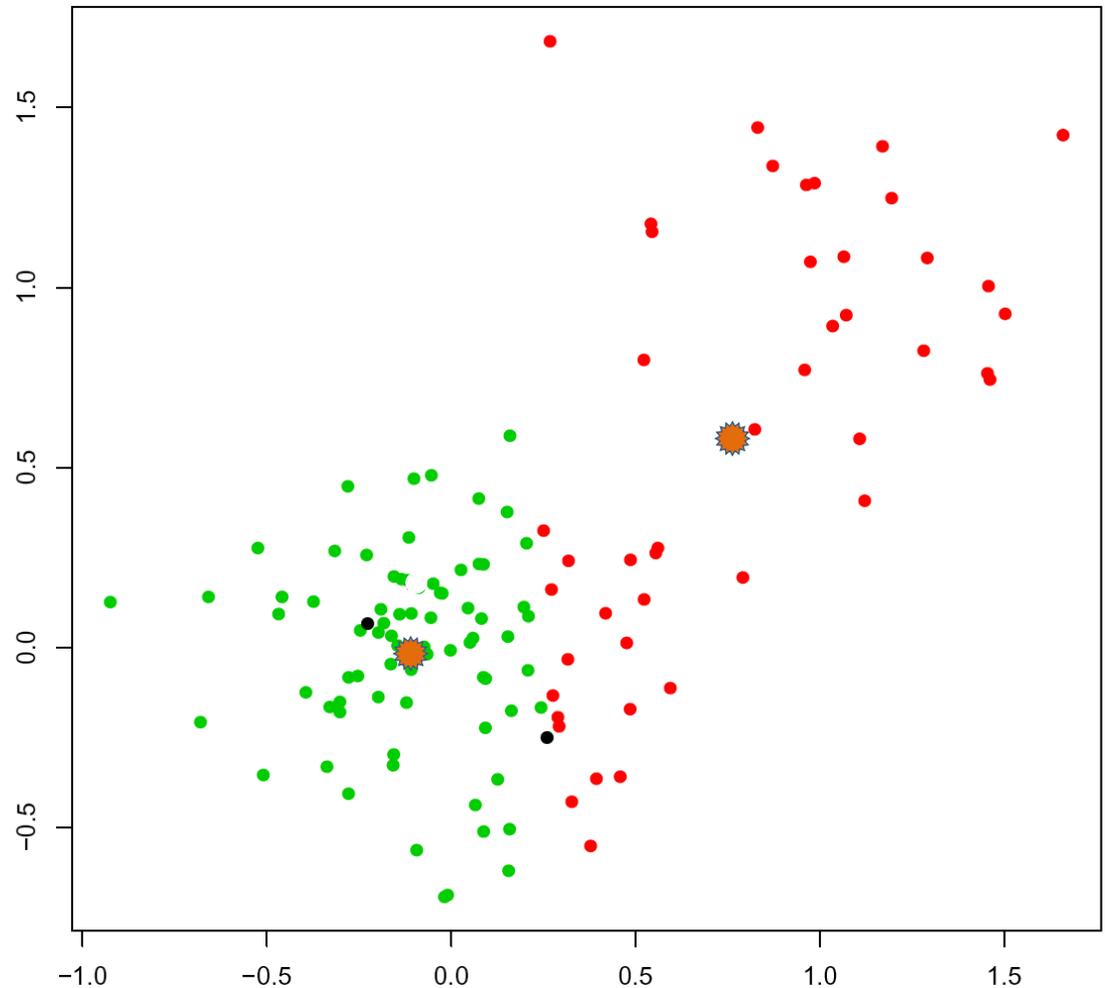
iter.max = 2 ; iterations = 2



K-mean clustering example

- Re-calculating centers again

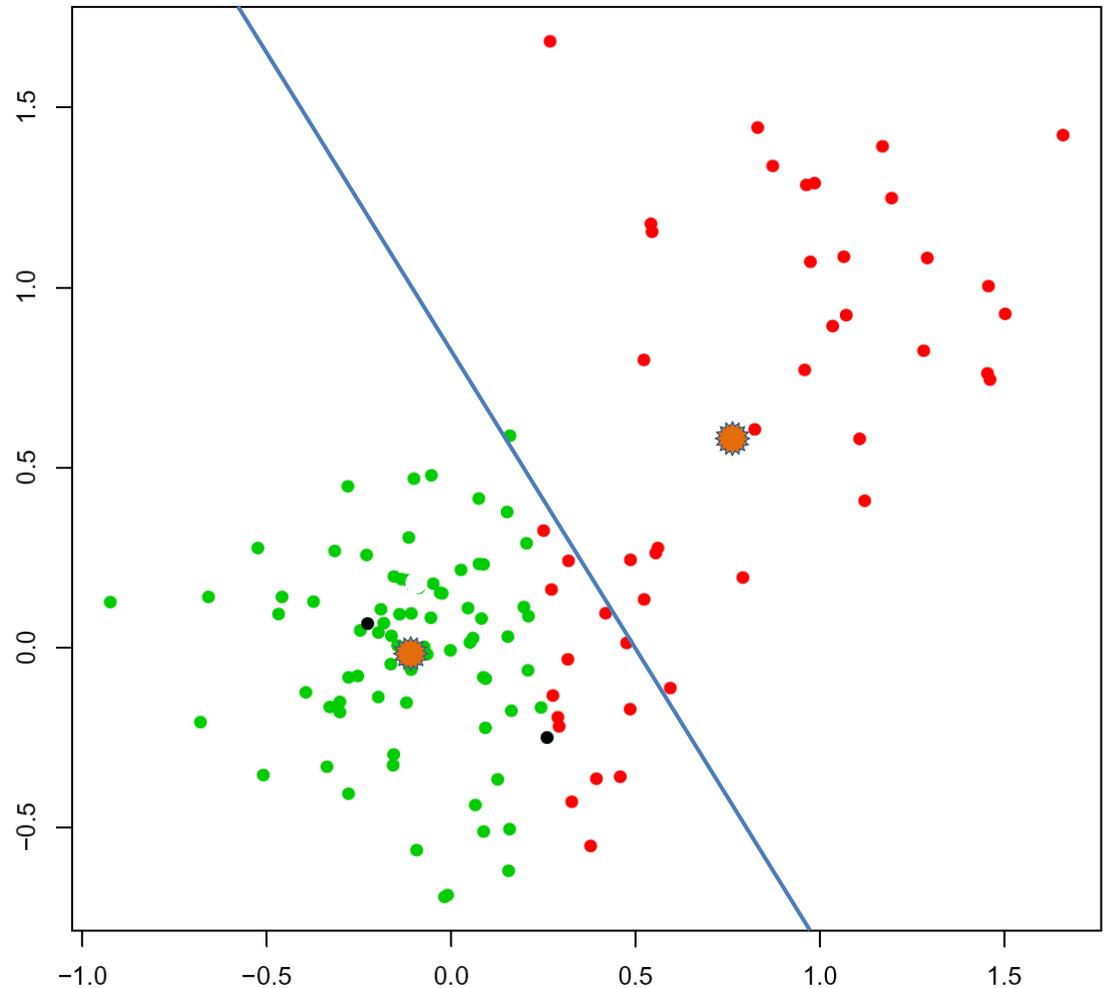
iter.max = 2 ; iterations = 2



K-mean clustering example

- And we can again partition the points

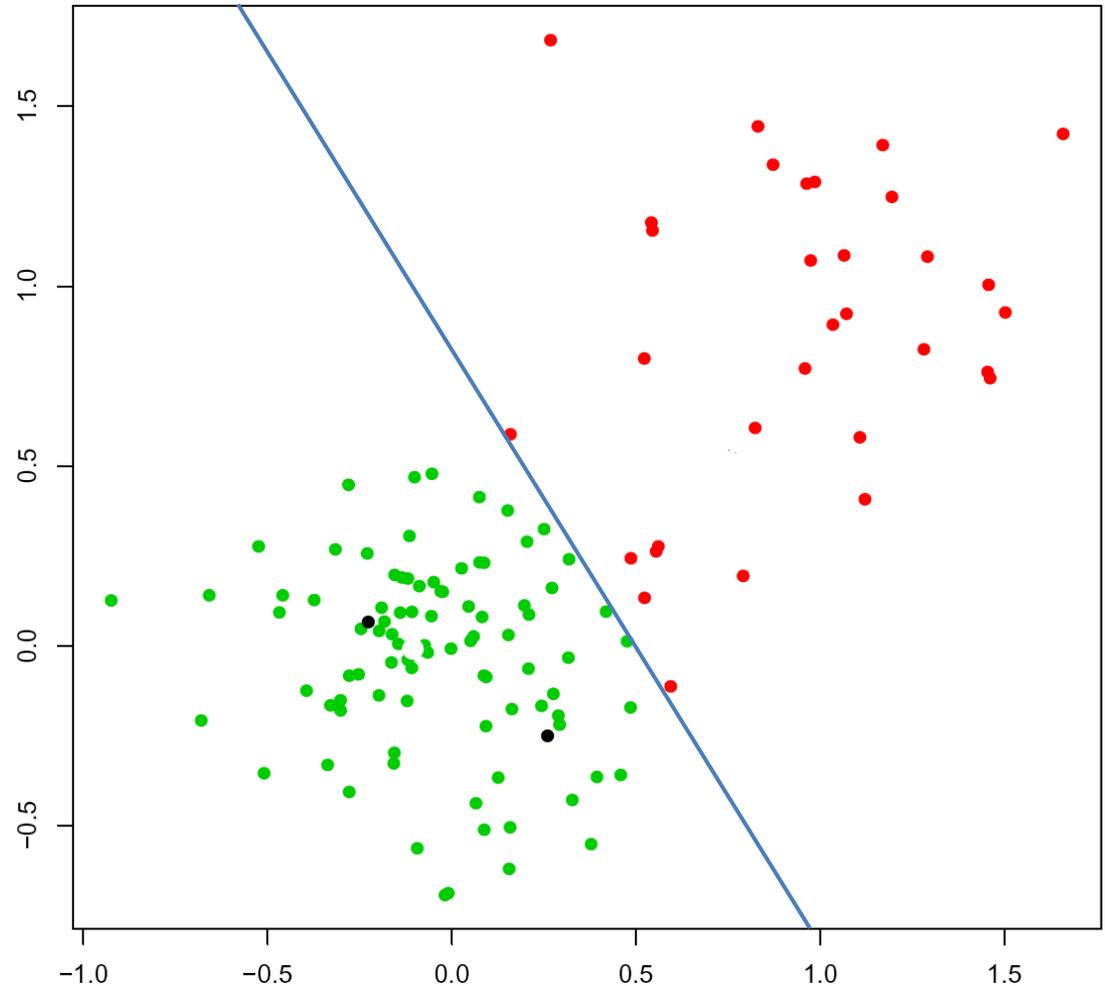
iter.max = 2 ; iterations = 2



K-mean clustering example

- Third partition into clusters

iter.max = 3 ; iterations = 3

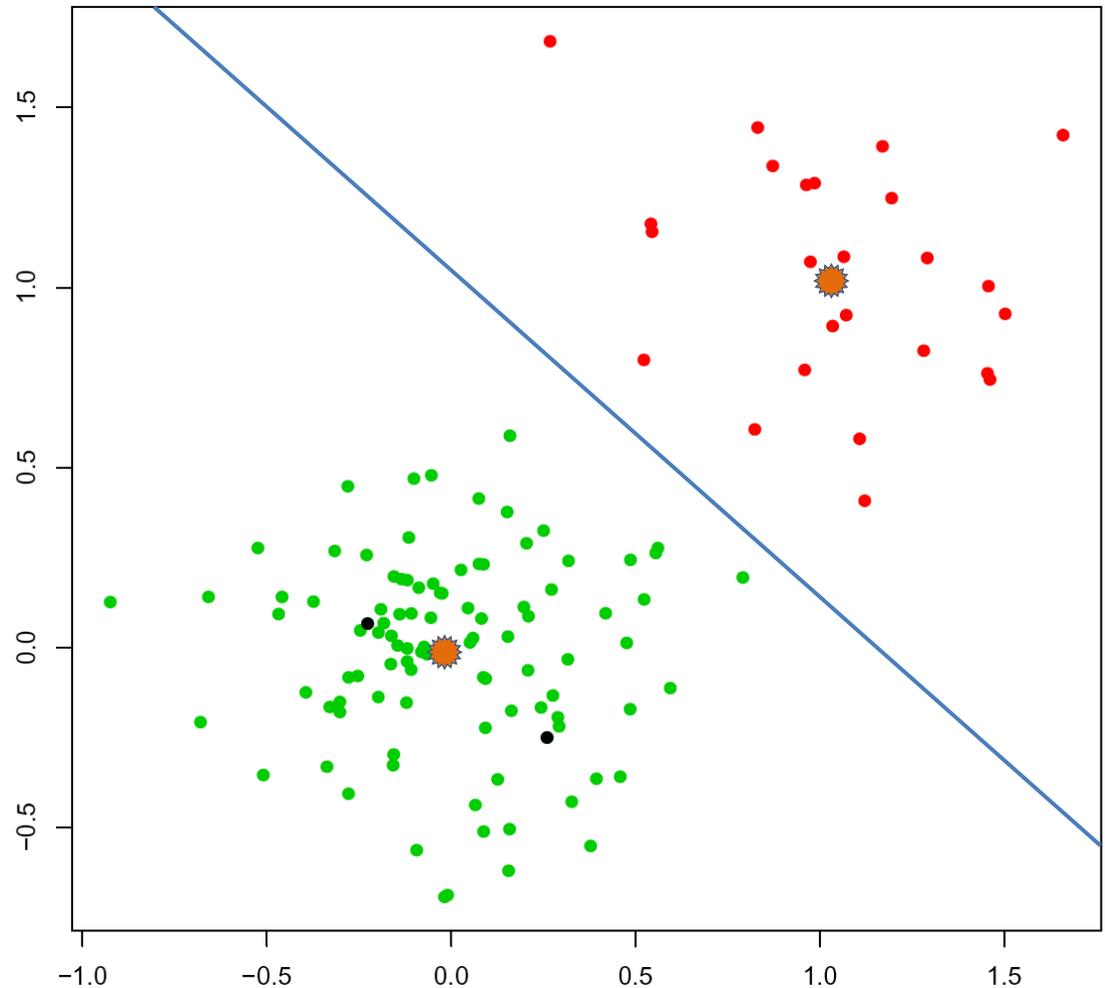


K-mean clustering example

- After 6 iterations:

iter.max = 6 ; iterations = 6

- The calculated centers remains stable



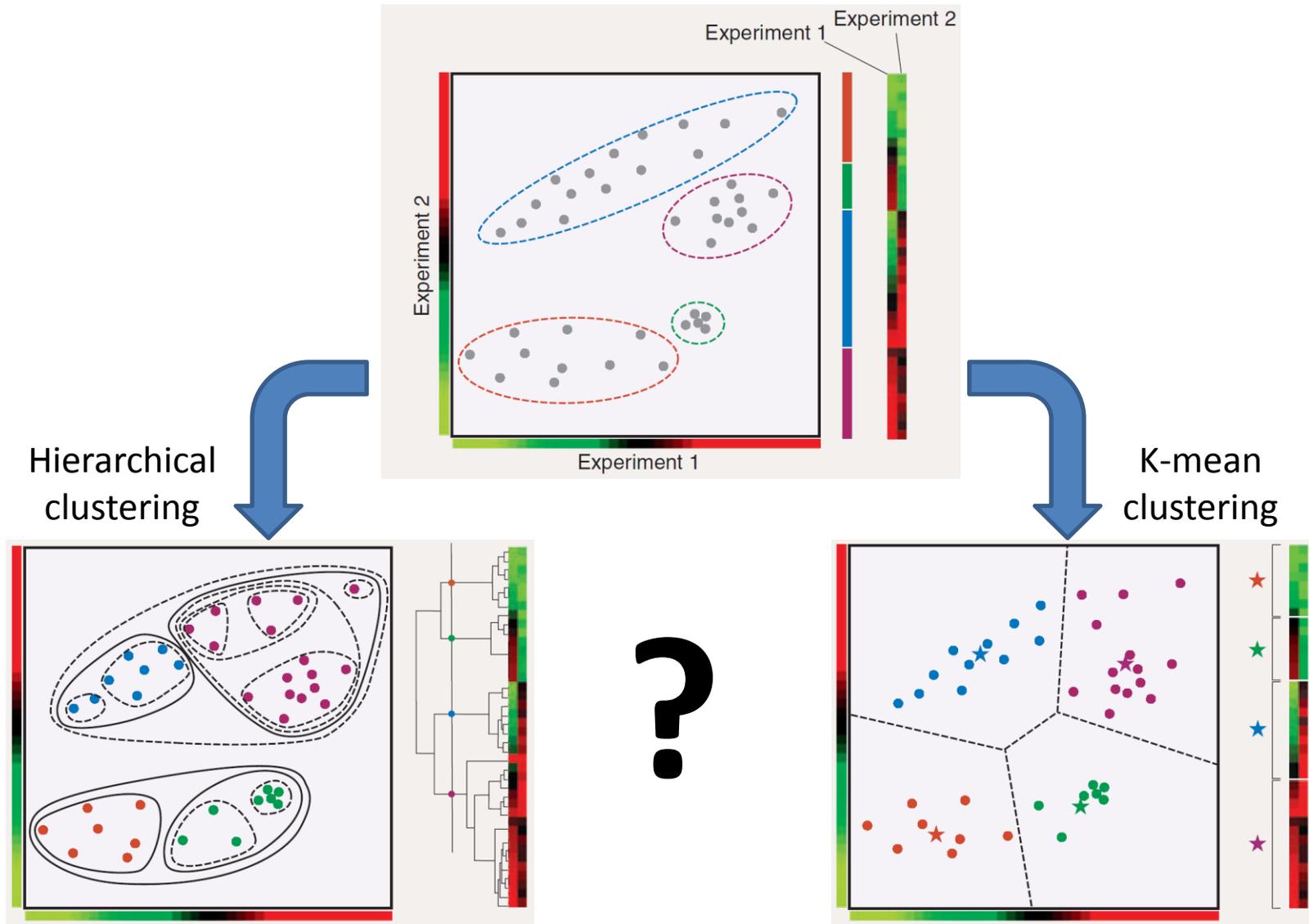
K-mean clustering: Summary

- The convergence of k-mean is usually quite fast (sometimes 1 iteration results in a stable solution)
- K-means is time- and memory-efficient
- **Strengths:**
 - Simple to use
 - Fast
 - Can be used with very large data sets
- **Weaknesses:**
 - **The number of clusters has to be predetermined**
 - The results may vary depending on the initial choice of centers

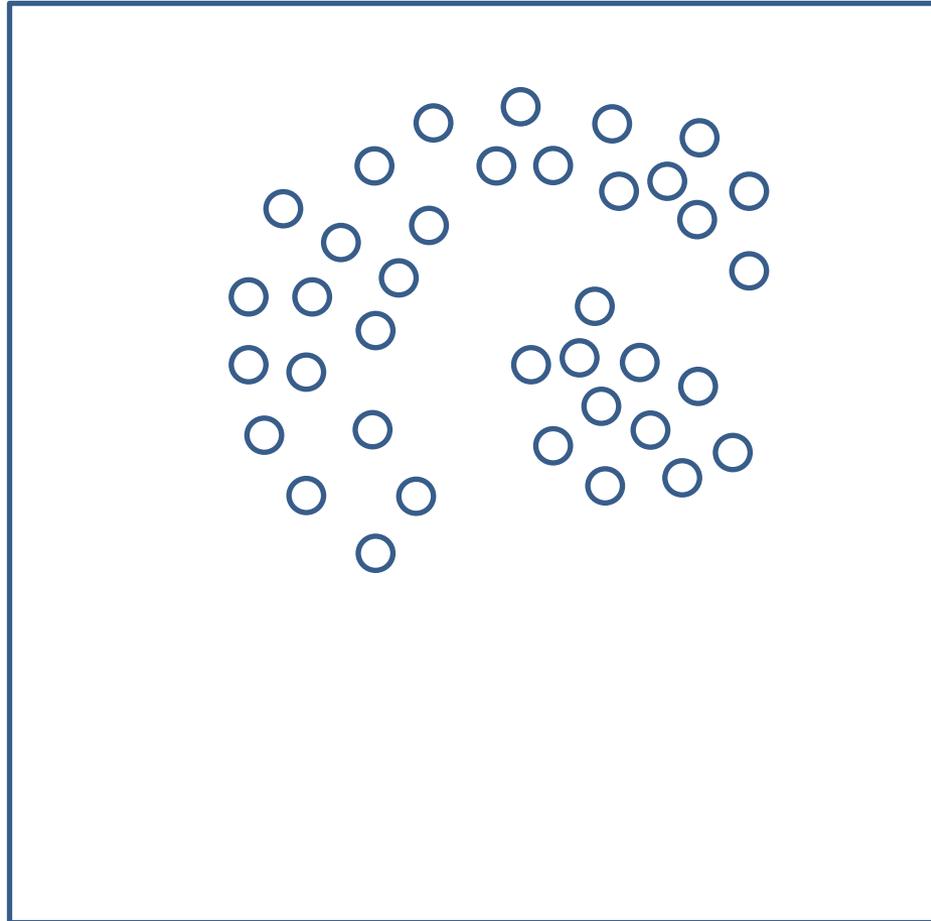
K-mean clustering: Variations

- Expectation-maximization (**EM**): maintains **probabilistic** assignments to clusters, instead of deterministic assignments, and multivariate Gaussian distributions instead of means.
- k-means++: attempts to choose better starting points.
- Some variations attempt to escape local optima by swapping points between clusters

The take-home message

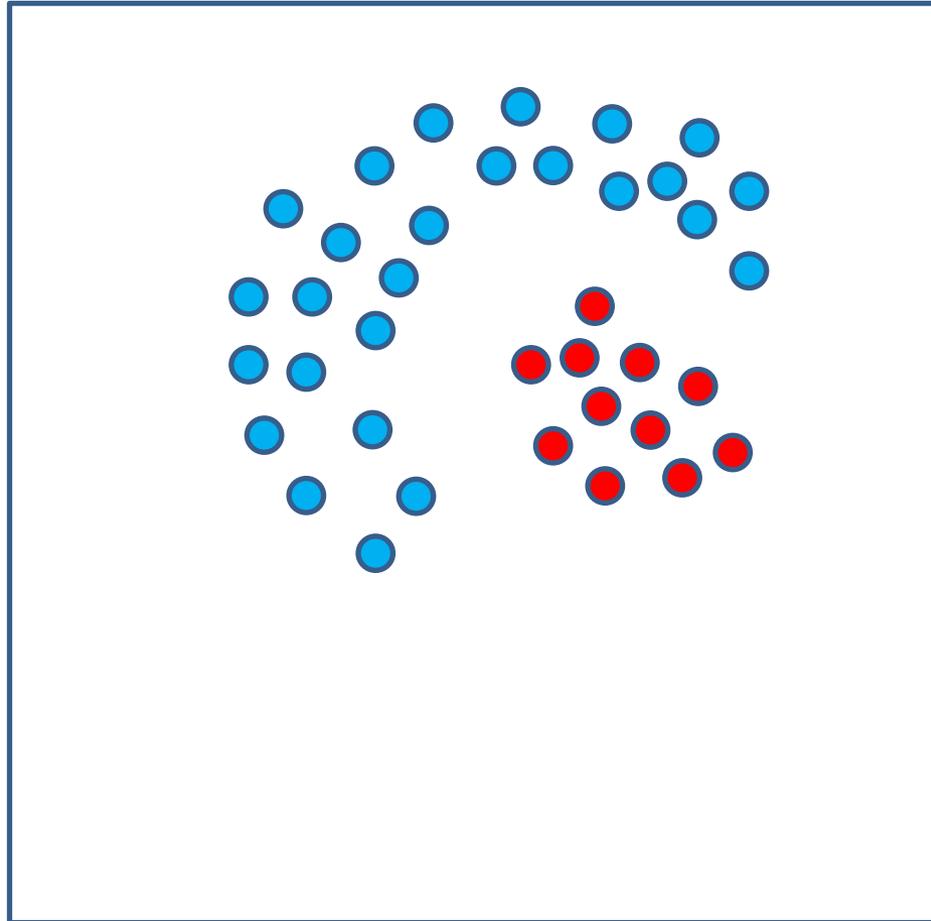


What else are we missing?



What else are we missing?

- What if the clusters are not “linearly separable”?



Cell cycle

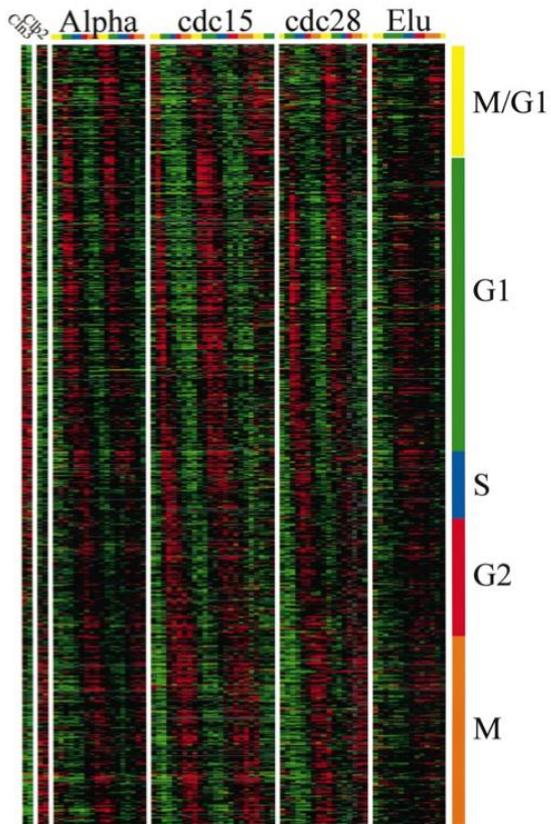
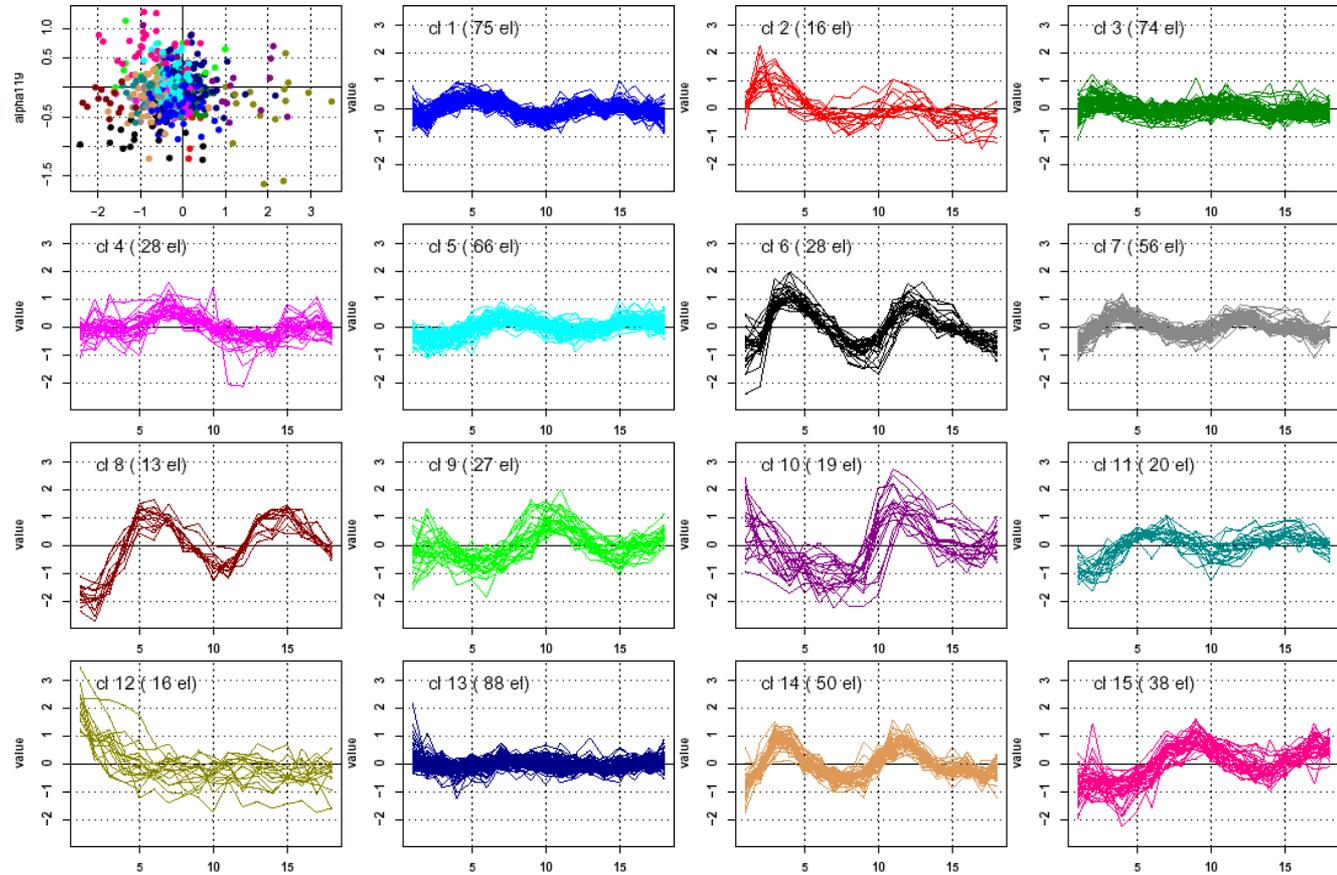


Figure 1.



Clustering methods

- We can distinguish between two types of clustering methods:
 1. **Agglomerative:** These methods build the clusters by examining small groups of elements and merging them in order to construct larger groups.
 2. **Divisive:** A different approach which analyzes large groups of elements in order to divide the data into smaller groups and eventually reach the desired clusters.
 - There is another way to distinguish between clustering methods:
 1. **Hierarchical:** Here we construct a hierarchy of clusters in order to examine the relationship between entities.
 2. **Non-Hierarchical:** In non-hierarchical methods, the elements are partitioned into non-overlapping groups.
-
- The diagram consists of two colored boxes on the right side of the slide. The top box is light blue and contains the text 'Hierarchical clustering'. The bottom box is light green and contains the text 'K-mean clustering'. A blue line starts from the word 'Agglomerative' in the first list item and points to the top box. A green line starts from the word 'Divisive' in the first list item and points to the bottom box. A blue line starts from the word 'Hierarchical' in the second list item and points to the top box. A green line starts from the word 'Non-Hierarchical' in the second list item and points to the bottom box.