

Clustering

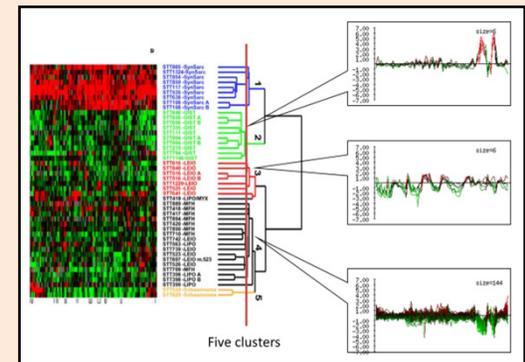
k-mean clustering

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
 - Clustering (unsupervised) vs. classification



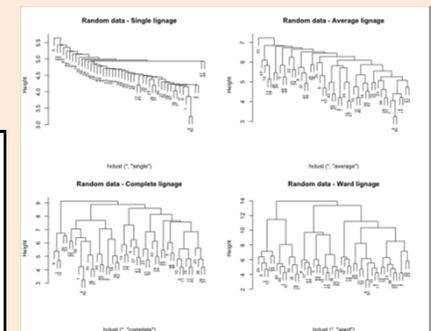
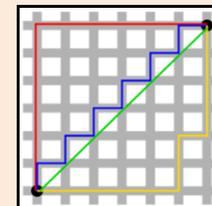
- Clustering methods:
 - Agglomerative vs. divisive; hierarchical vs. non-hierarchical

- **Hierarchical clustering algorithm:**

1. Assign each object to a separate cluster.
2. Find the pair of clusters with the shortest distance, and regroup them into a single cluster.
3. Repeat 2 until there is a single cluster.

- Many possible distance metrics

- **Metric matters**



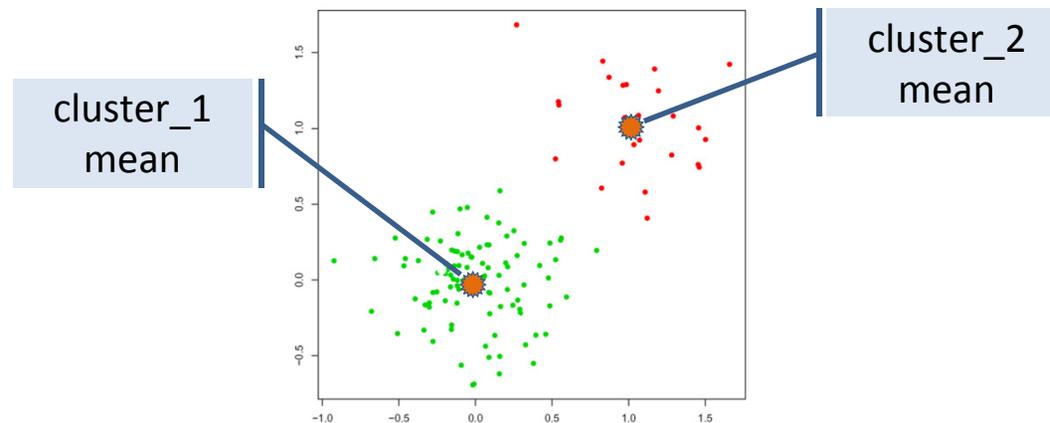
K-mean clustering

Agglomerative

Non-hierarchical

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

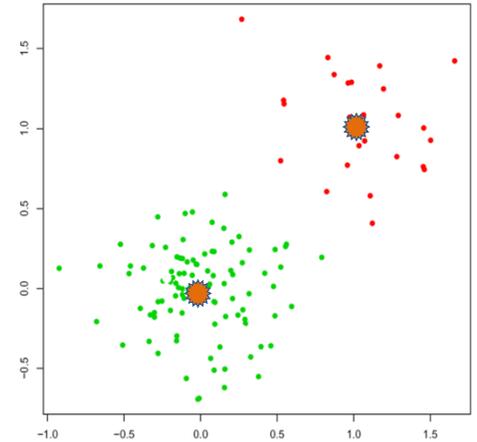


- Isn't this a somewhat circular definition?
 - Assignment of a point to a cluster is based on the proximity of the point to the cluster mean
 - But the cluster mean is calculated based on all the points assigned to the cluster.

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 (somewhat similar to expectation-maximization algorithm)



K-mean clustering algorithm

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

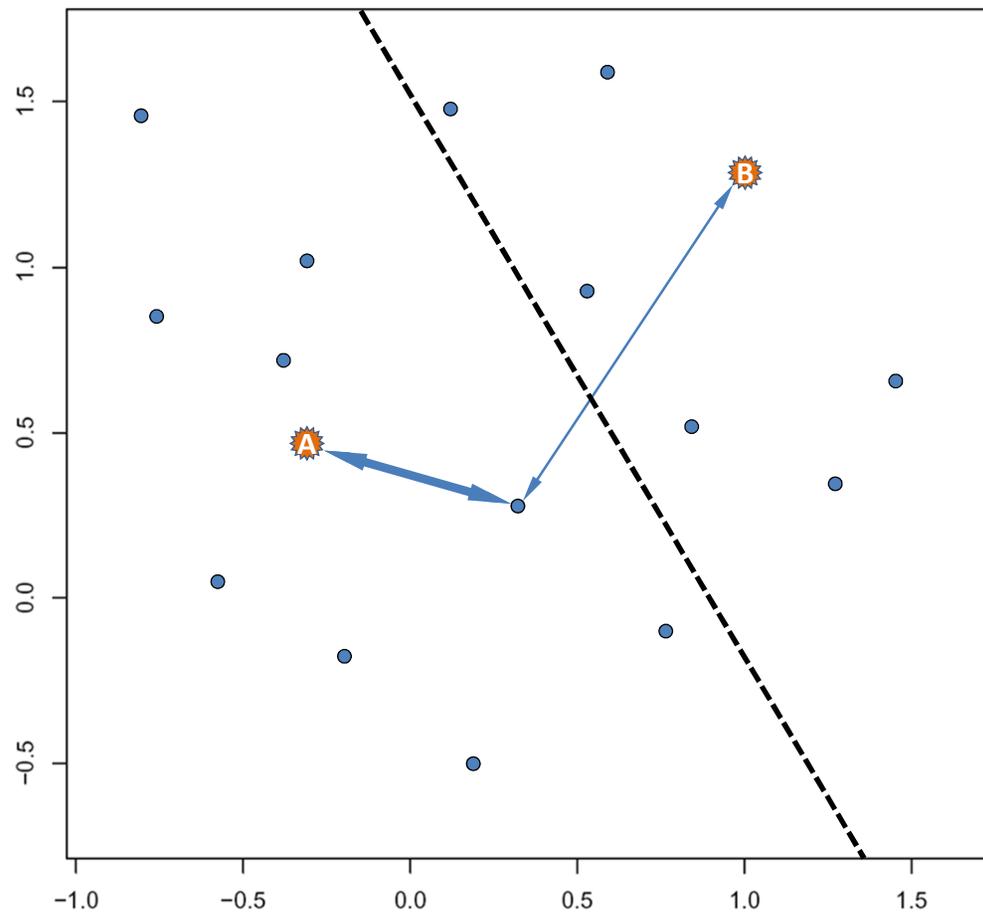
- **Algorithm:**

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?

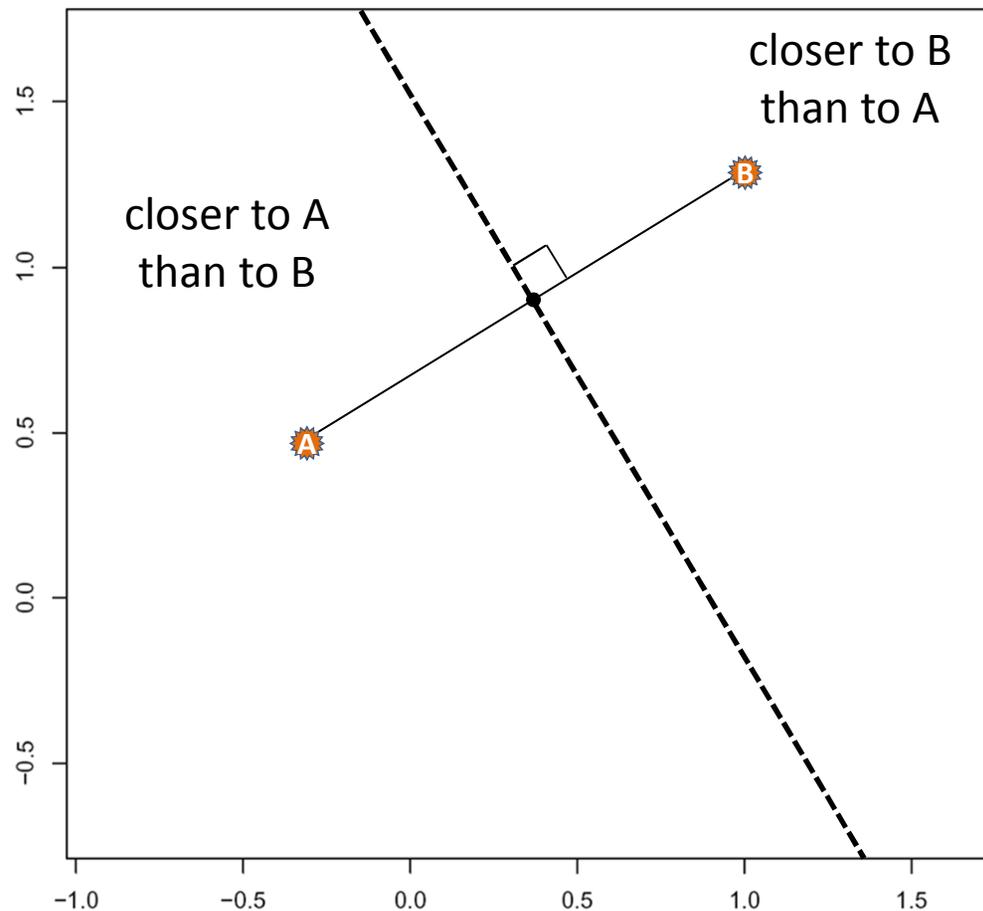
Partitioning the space

- Assigning elements to the closest center



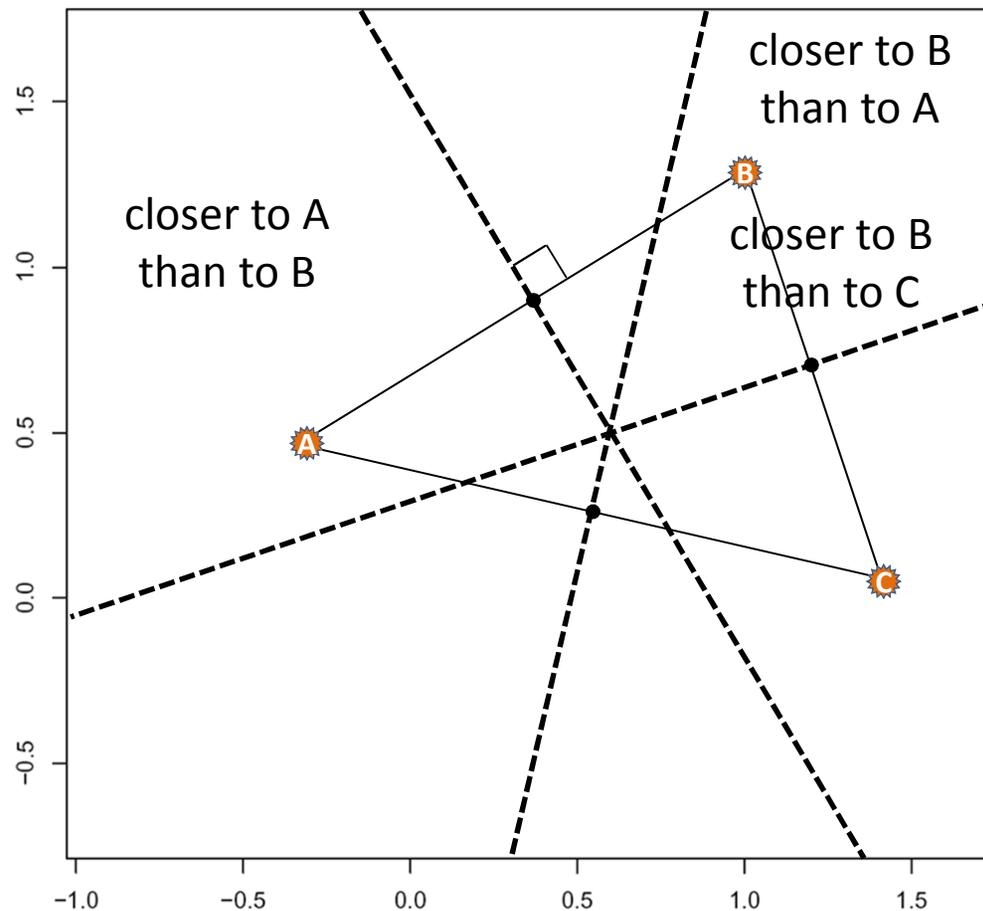
Partitioning the space

- Assigning elements to the closest center



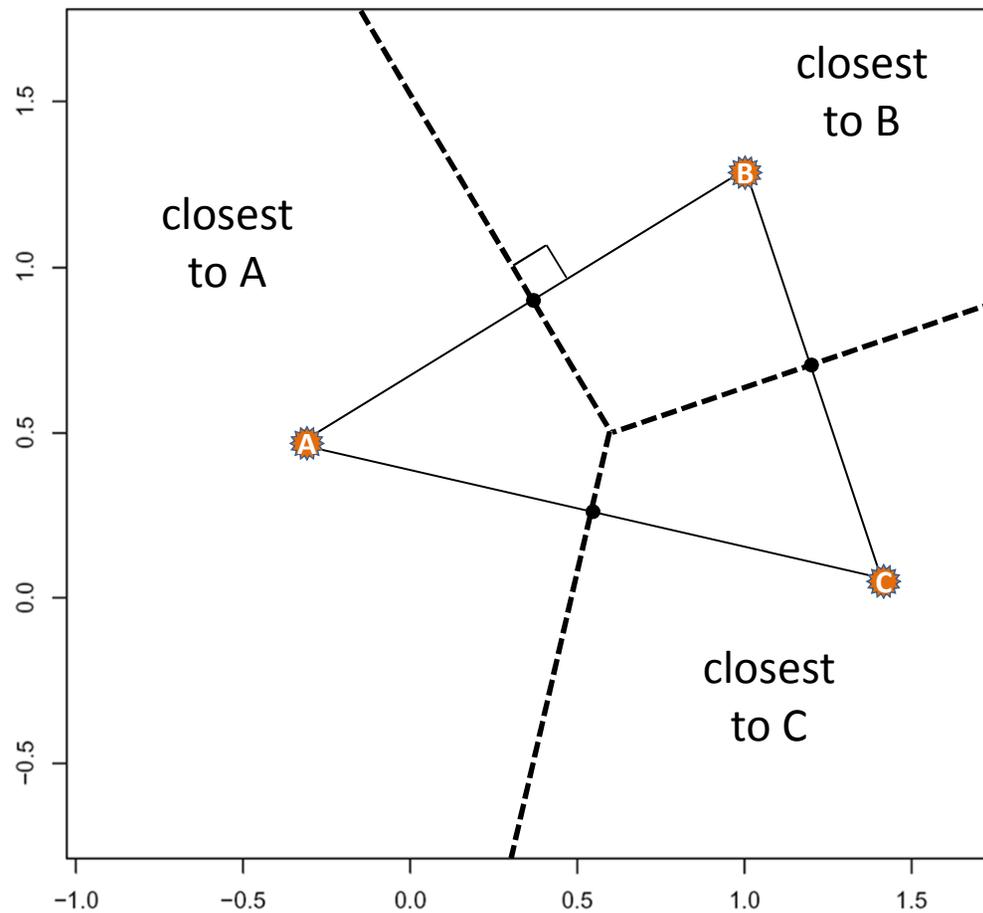
Partitioning the space

- Assigning elements to the closest center



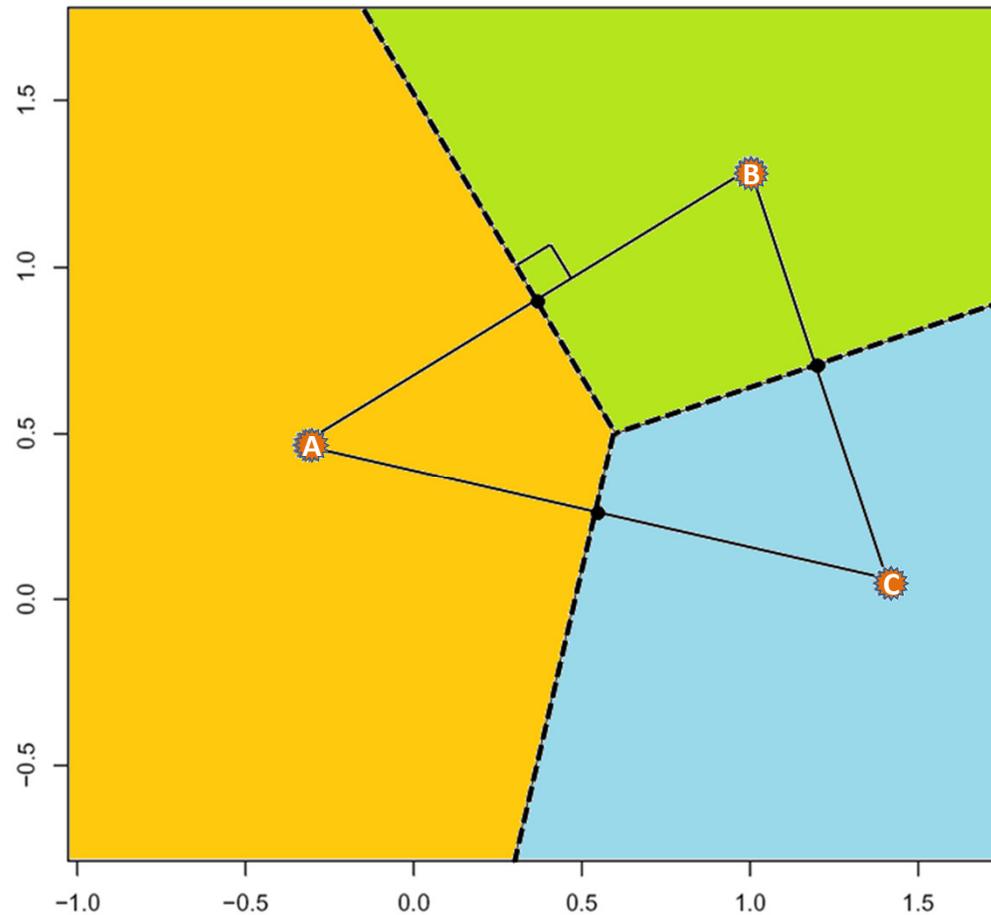
Partitioning the space

- Assigning elements to the closest center



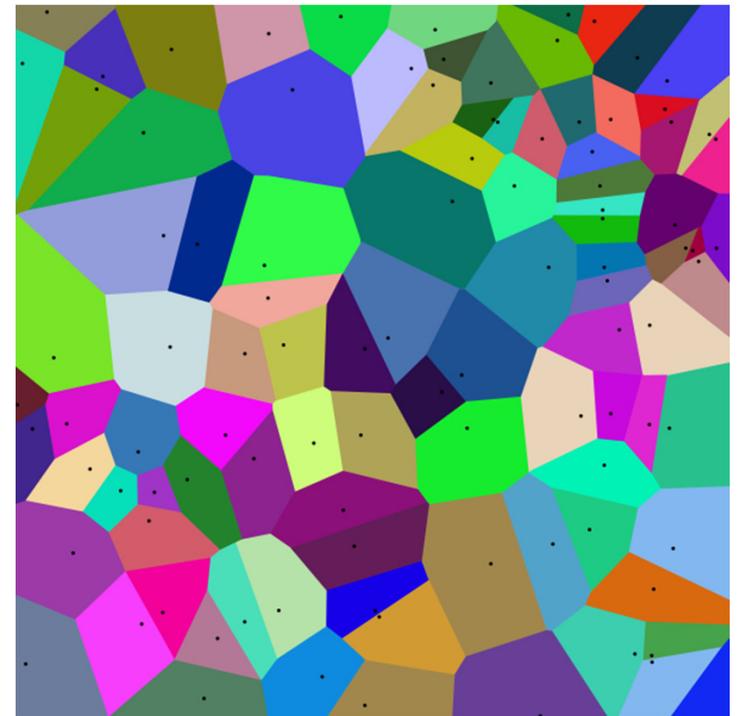
Partitioning the space

- 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.



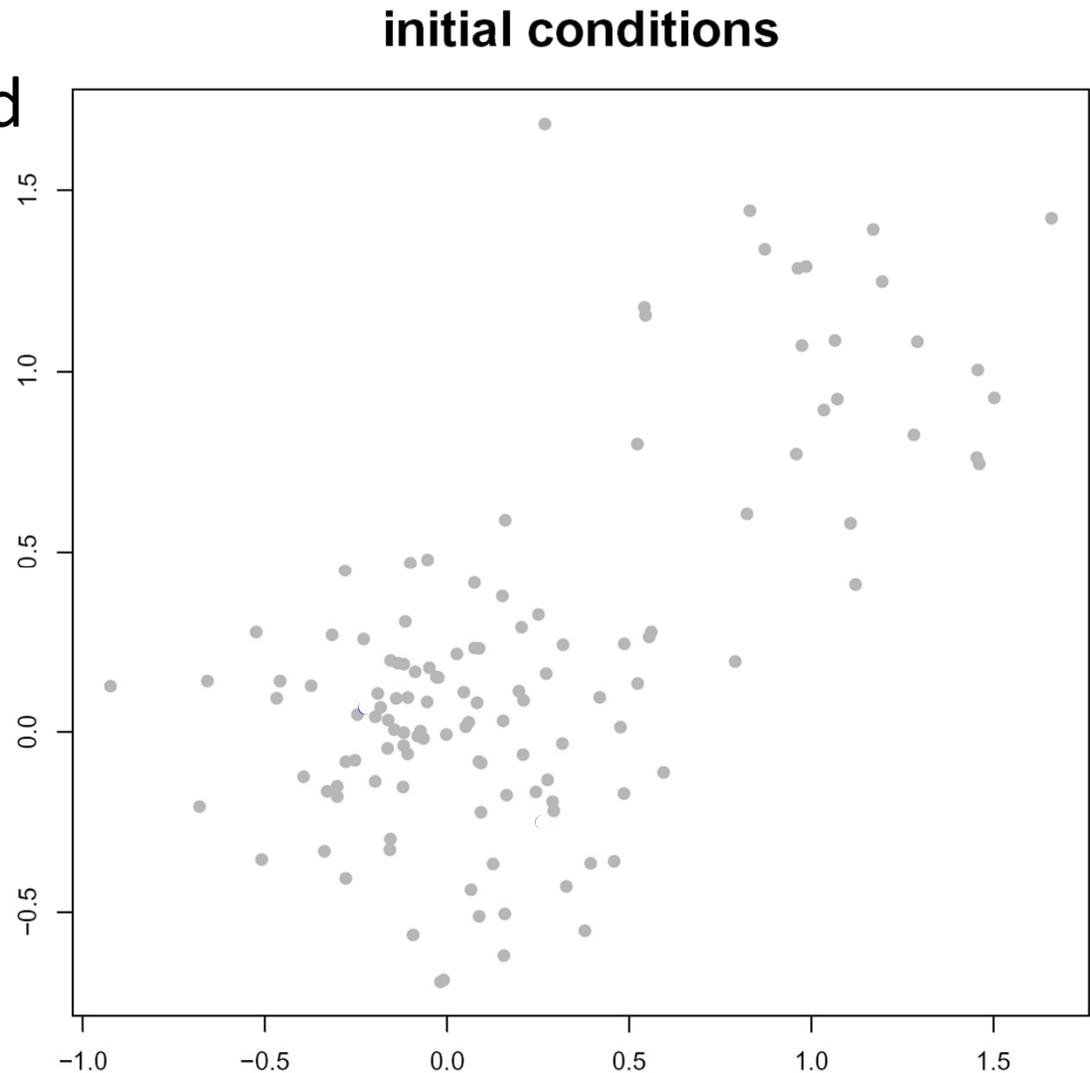
K-mean clustering algorithm

- 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

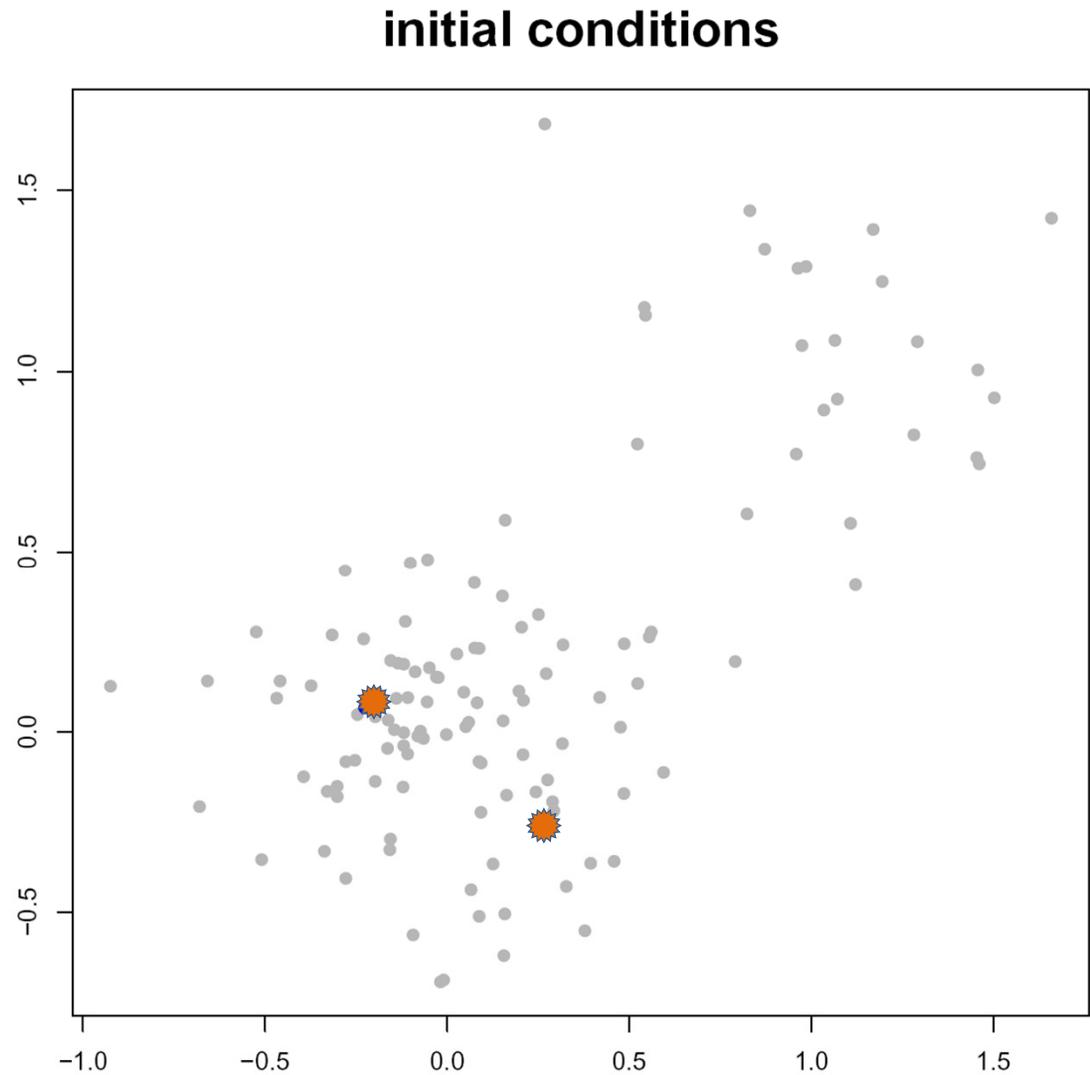
- Two sets of points randomly generated

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



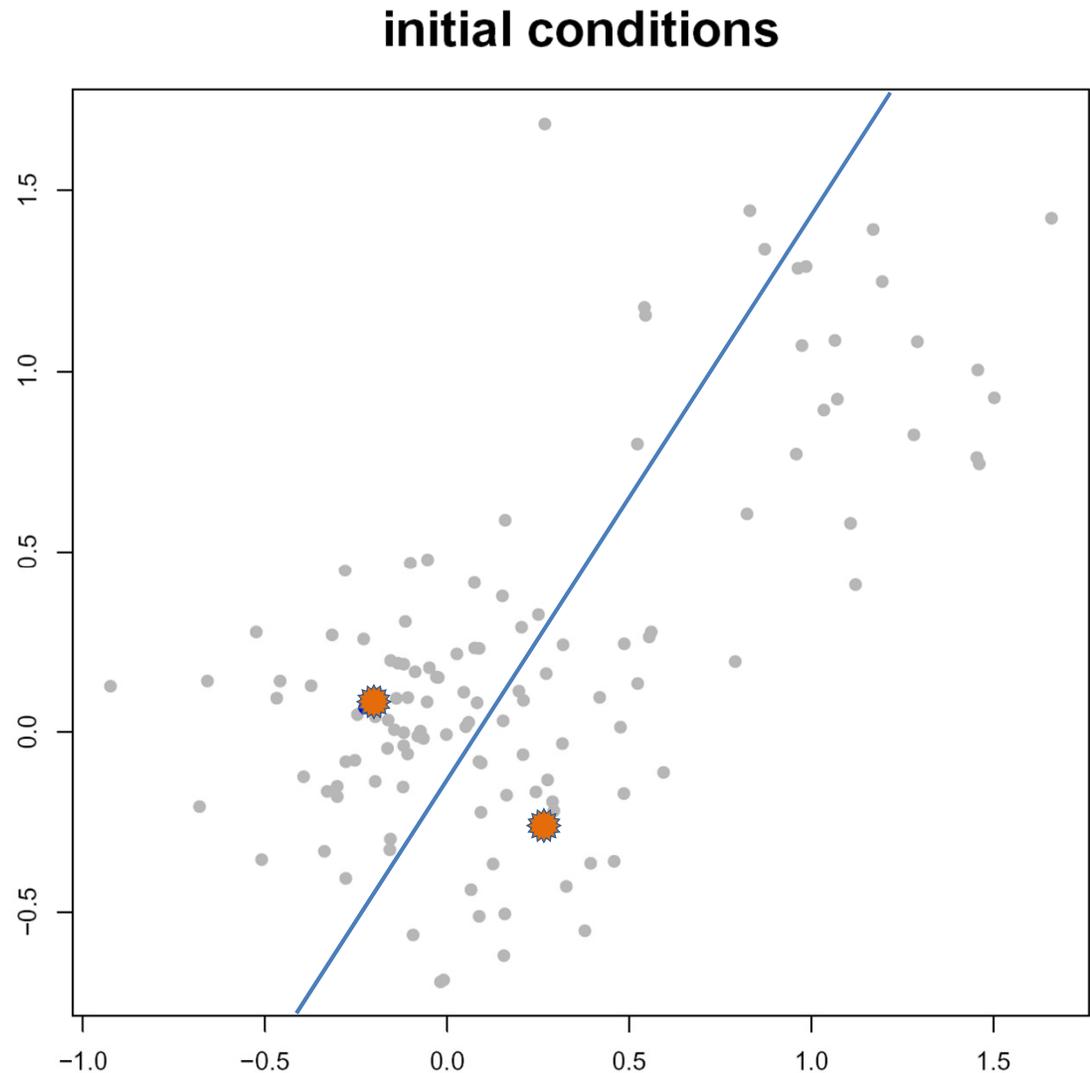
K-mean clustering example

- Two points are randomly chosen as centers (stars)



K-mean clustering example

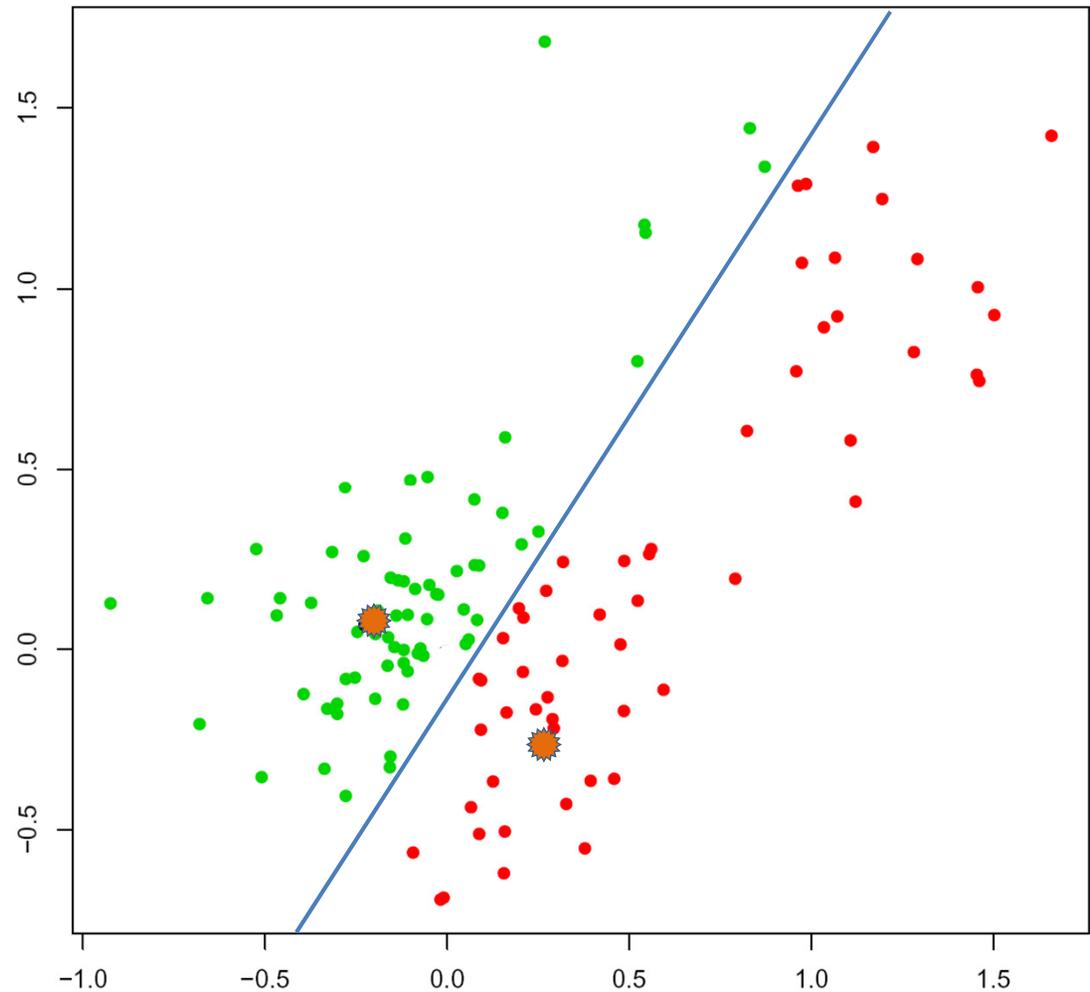
- Each dot can now be assigned to the cluster with the closest center



K-mean clustering example

- First partition into clusters

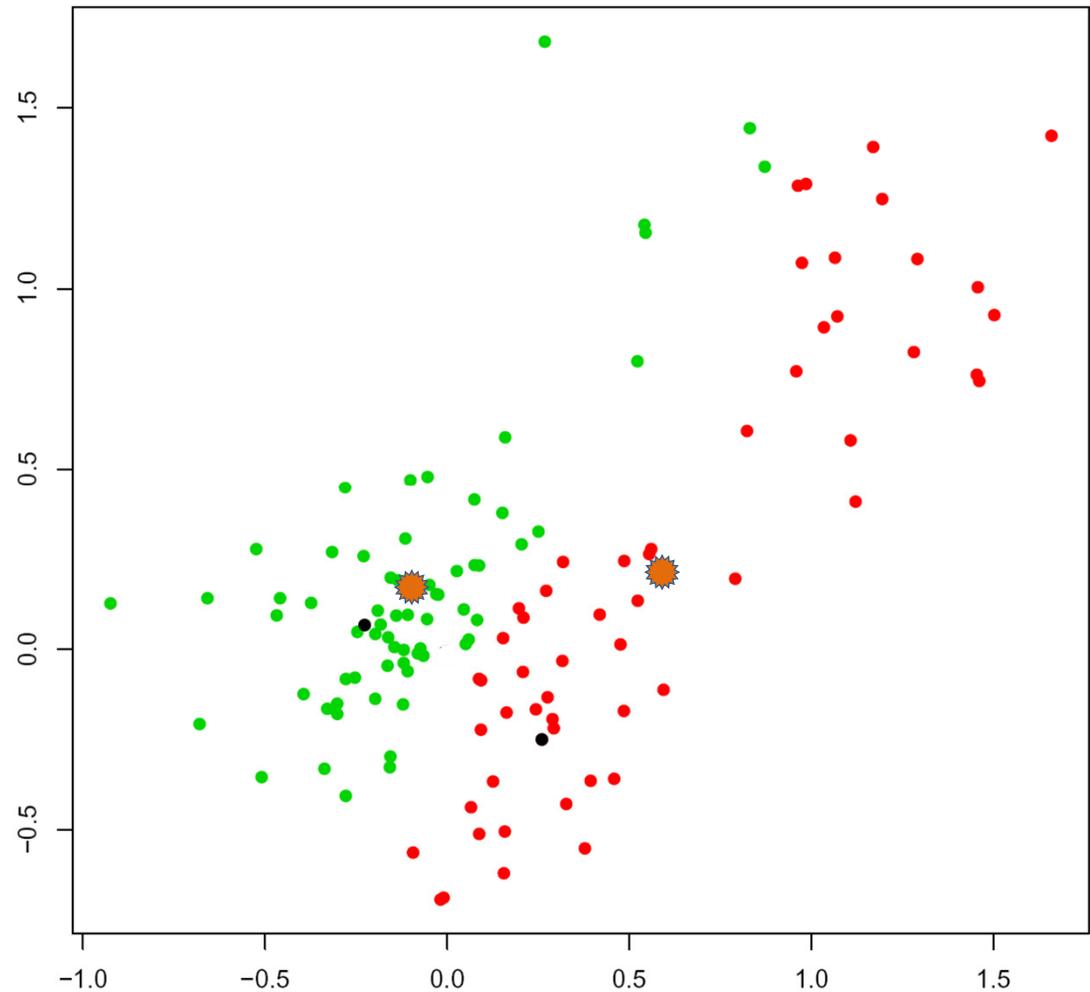
iter.max = 1 ; iterations = 1



K-mean clustering example

- Centers are re-calculated

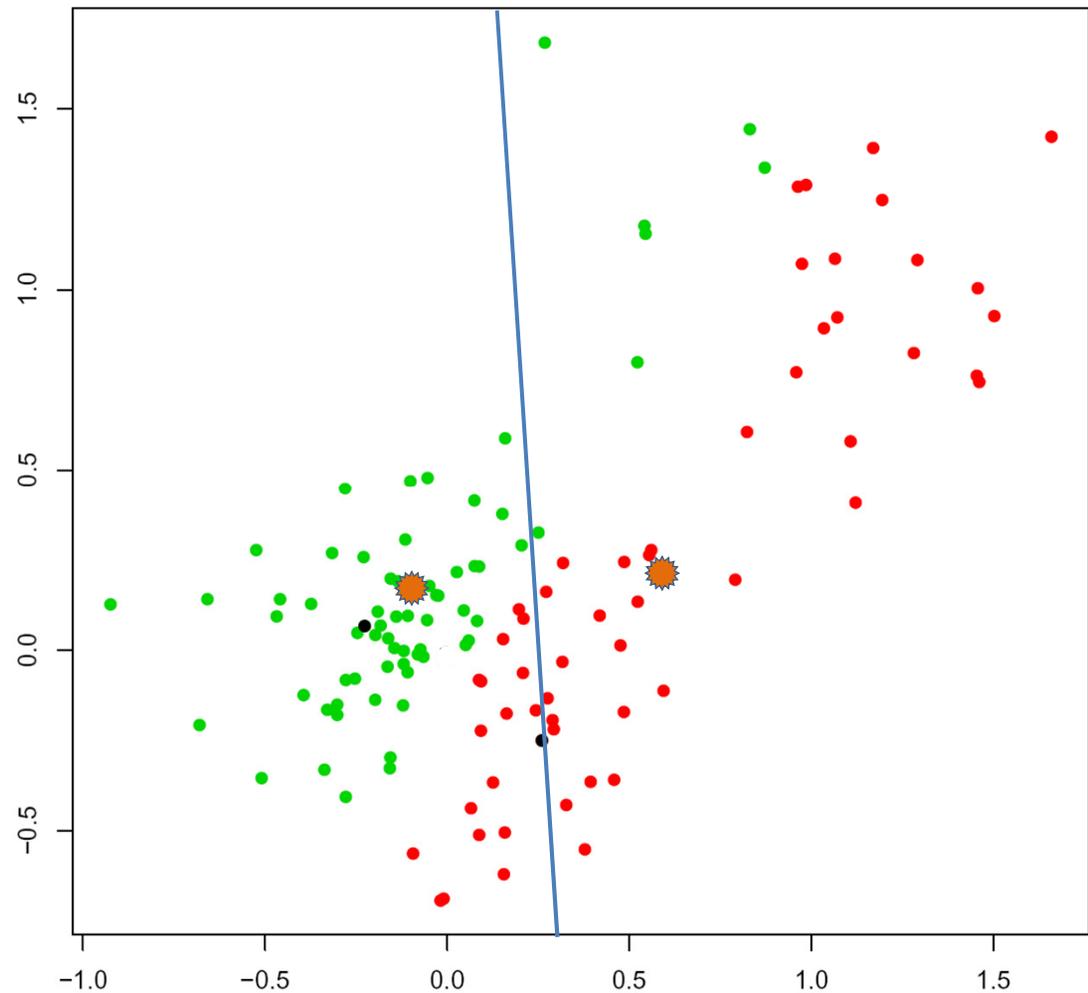
iter.max = 1 ; iterations = 1



K-mean clustering example

- And are again used to partition the points

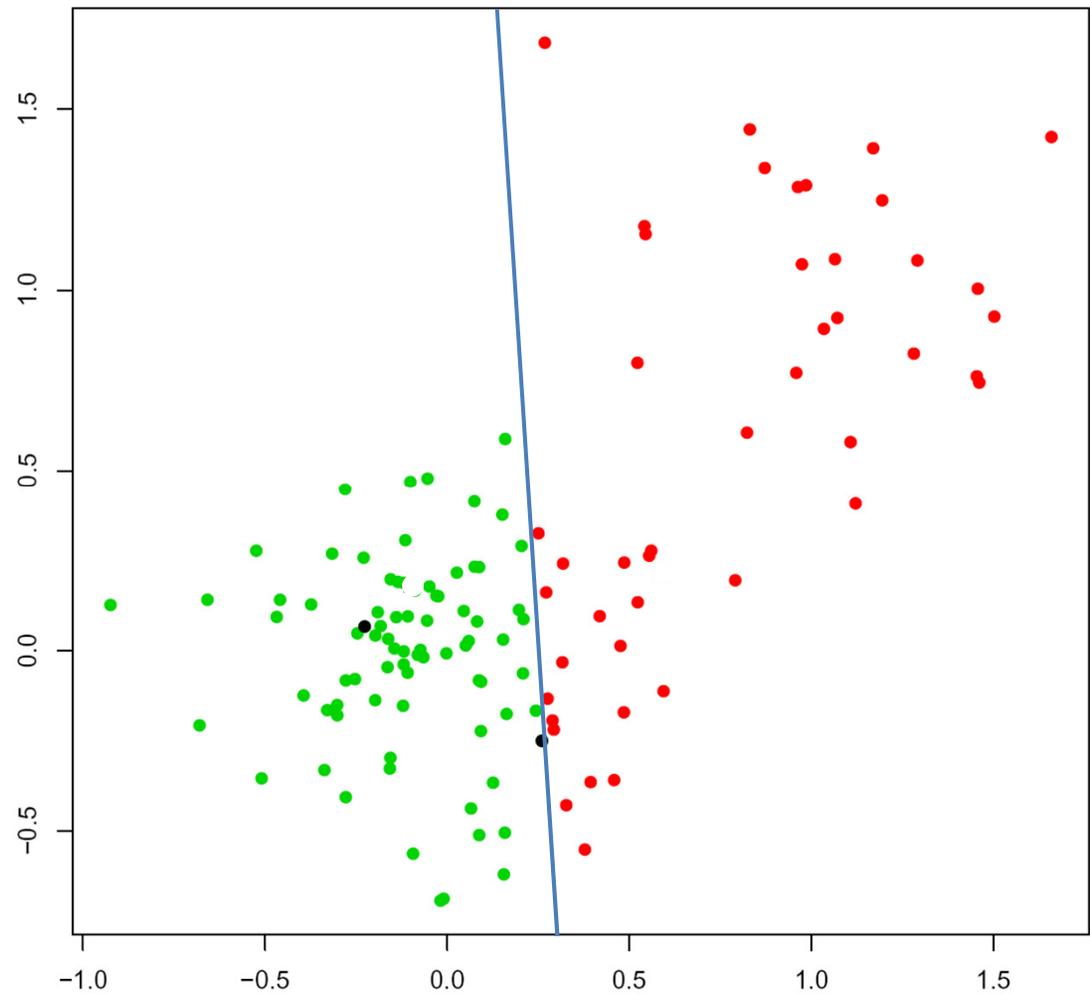
iter.max = 1 ; iterations = 1



K-mean clustering example

- Second partition into clusters

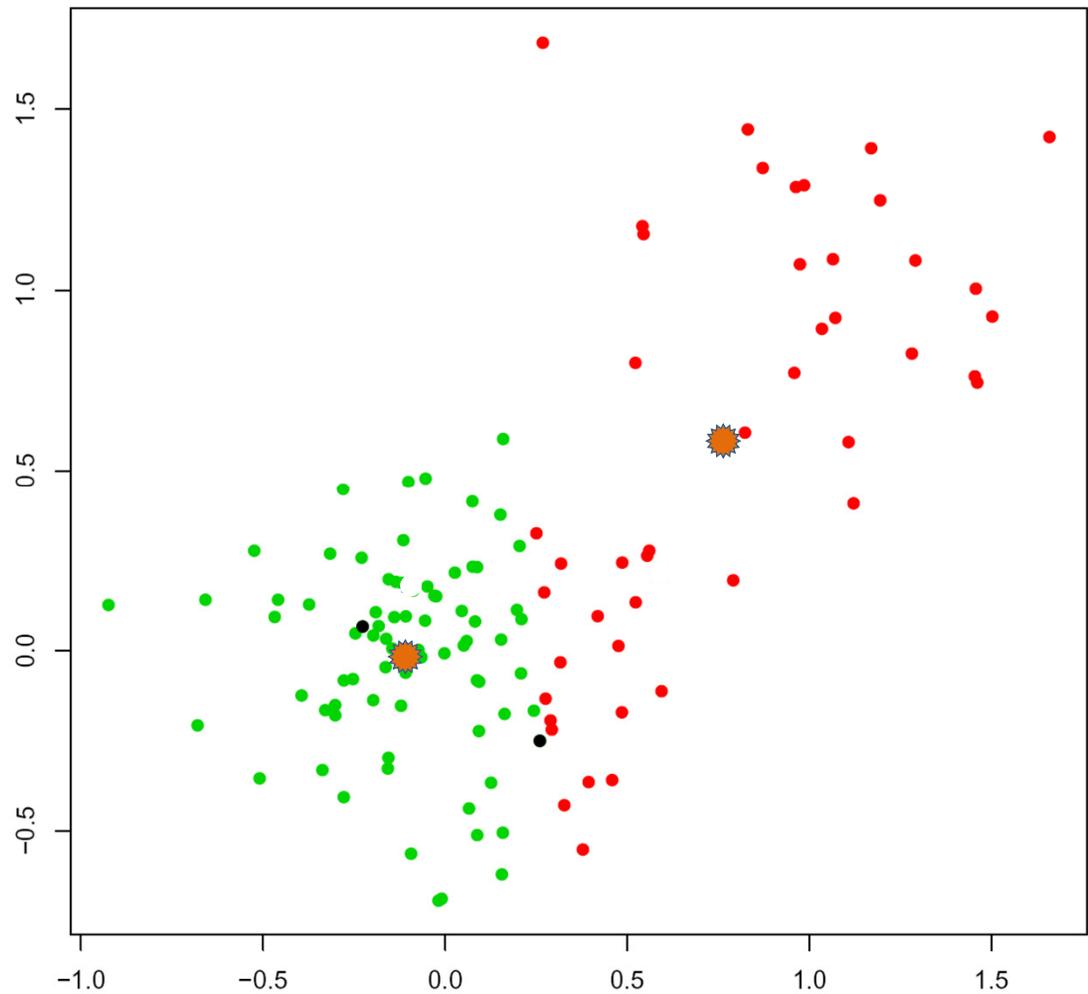
iter.max = 2 ; iterations = 2



K-mean clustering example

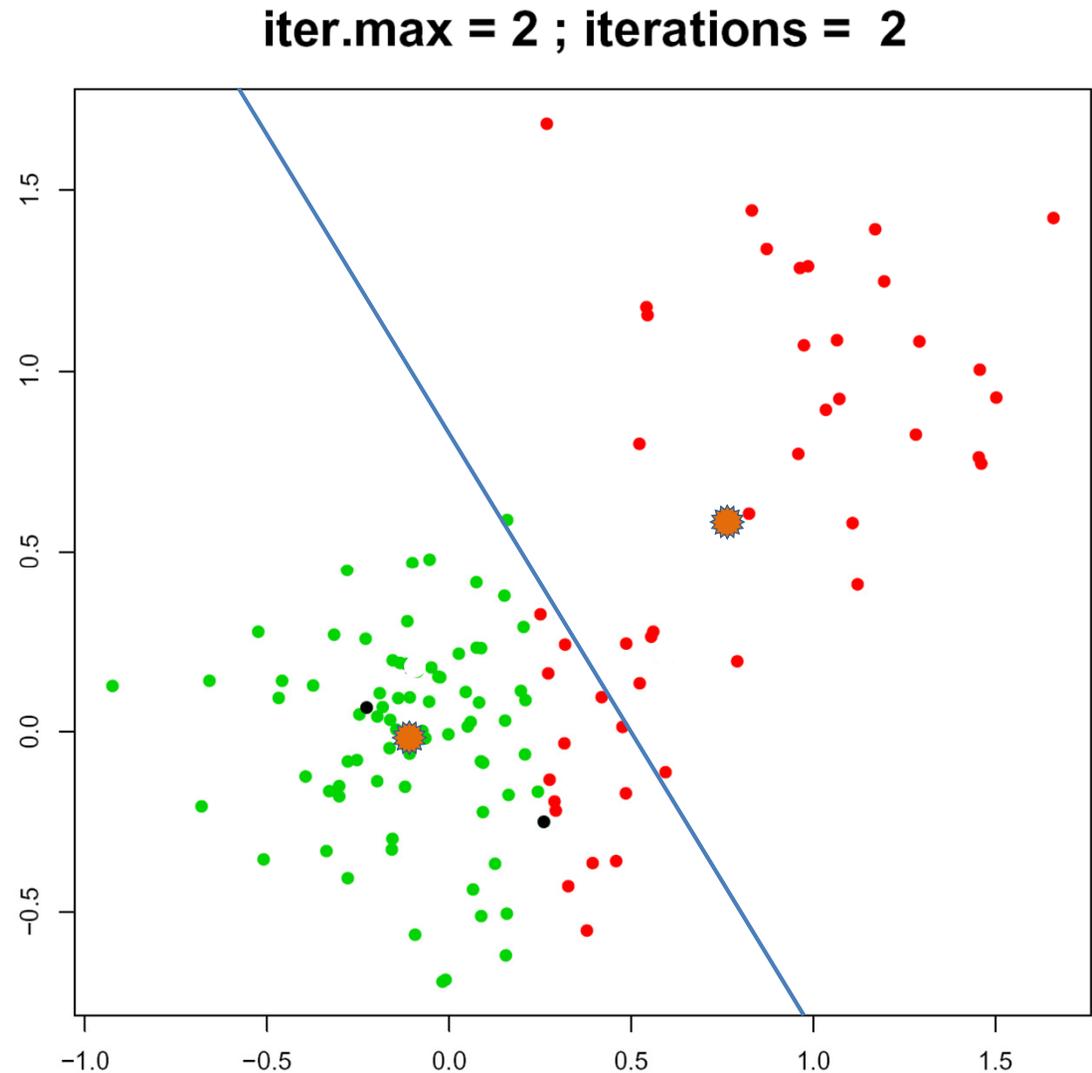
- Re-calculating centers again

iter.max = 2 ; iterations = 2



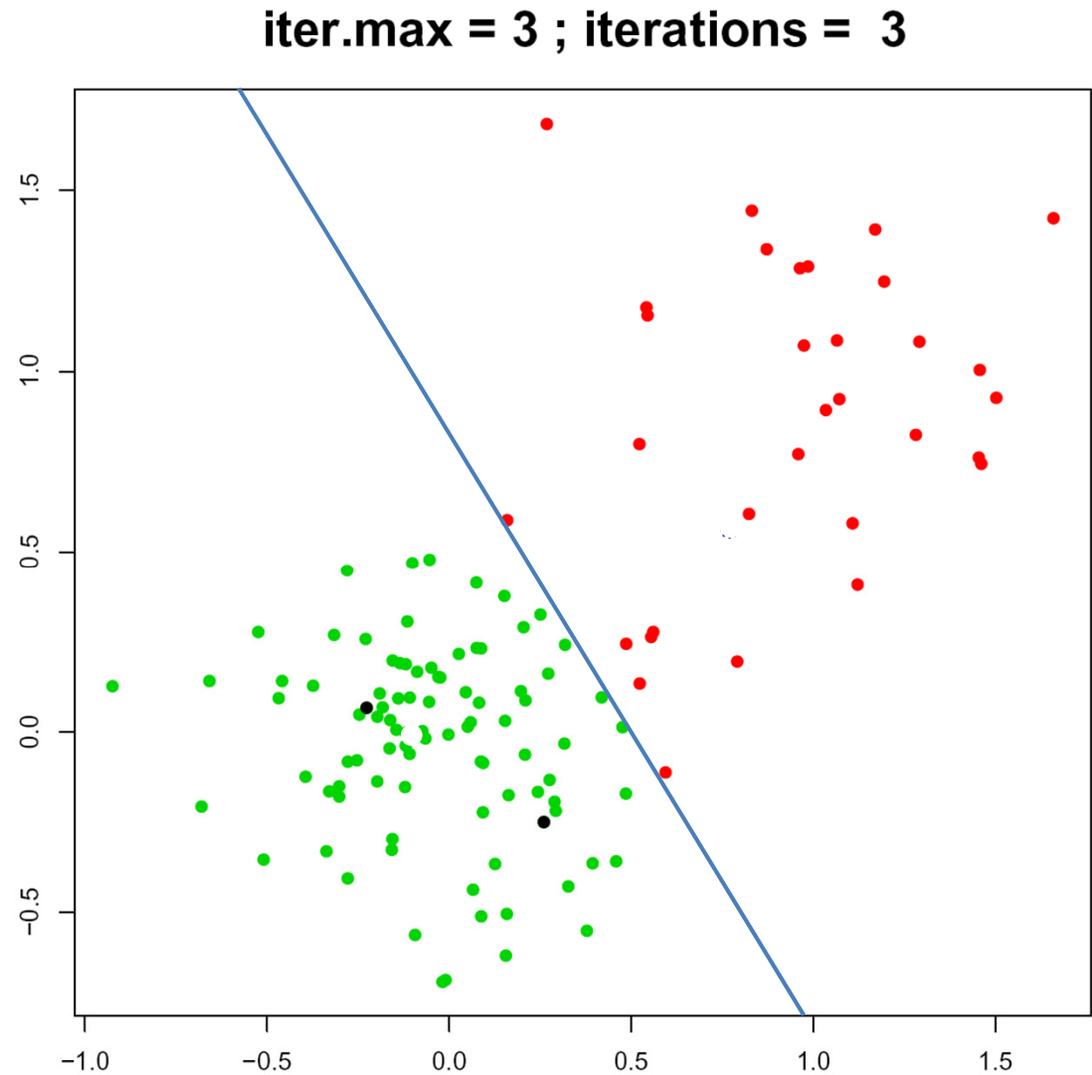
K-mean clustering example

- And we can again partition the points



K-mean clustering example

- Third partition into clusters

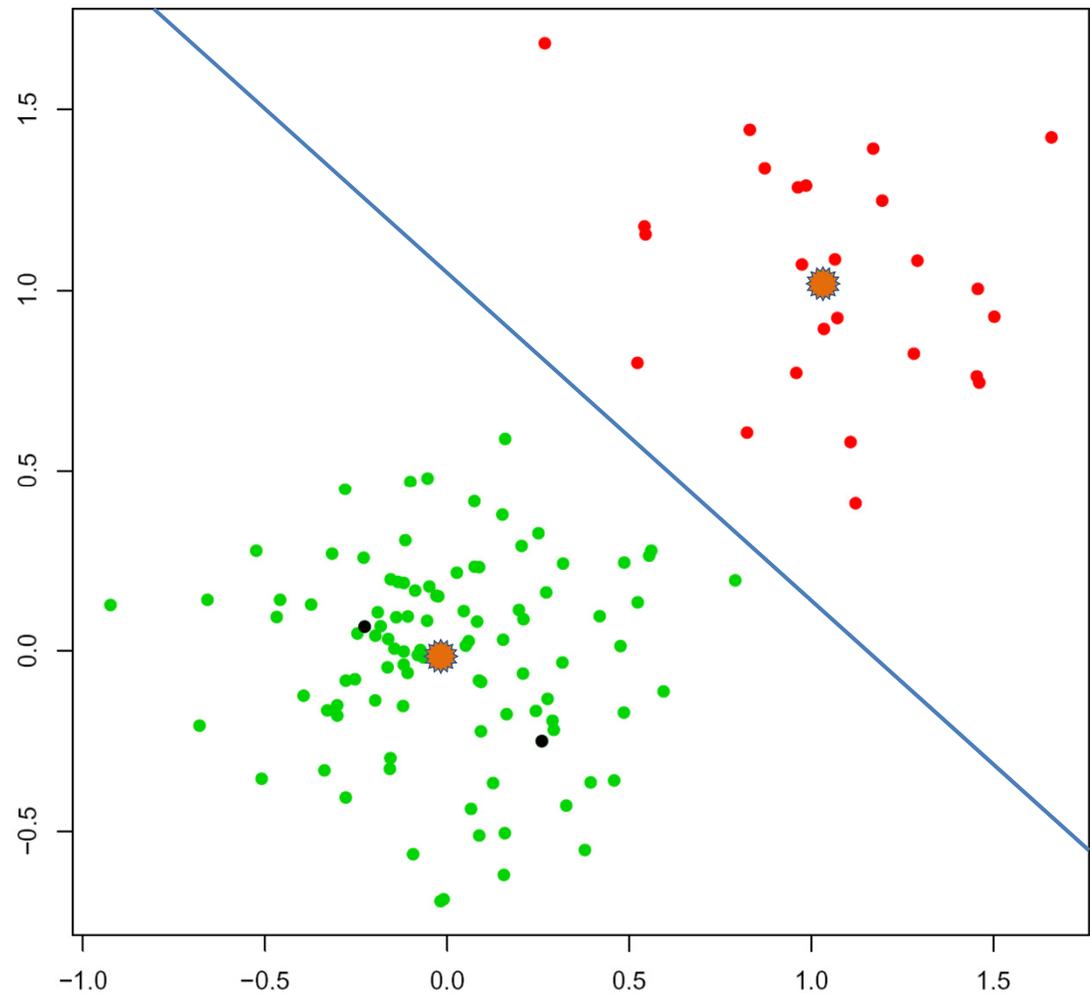


K-mean clustering example

- After 6 iterations:

iter.max = 6 ; iterations = 6

- The calculated centers remains stable



Cell cycle

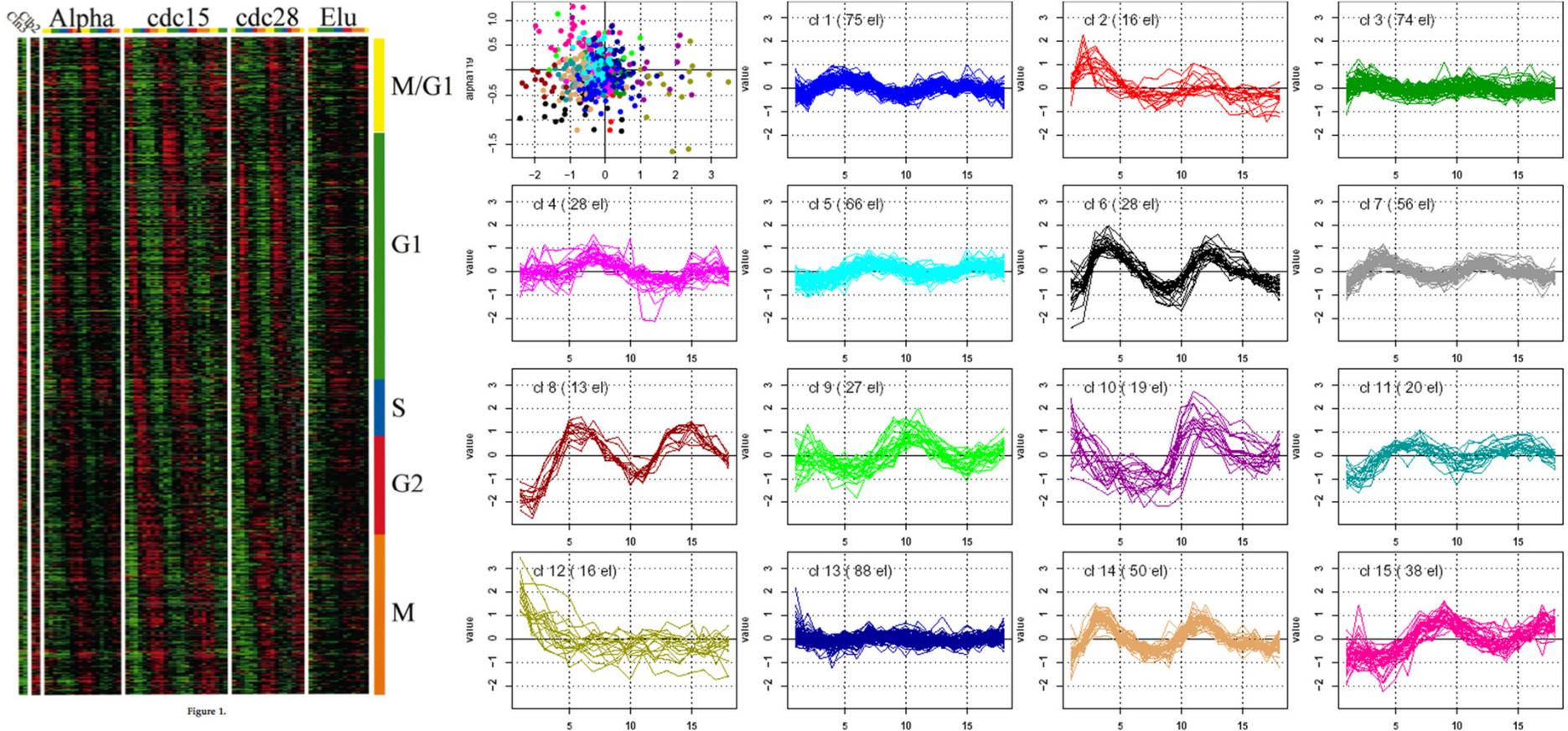


Figure 1.

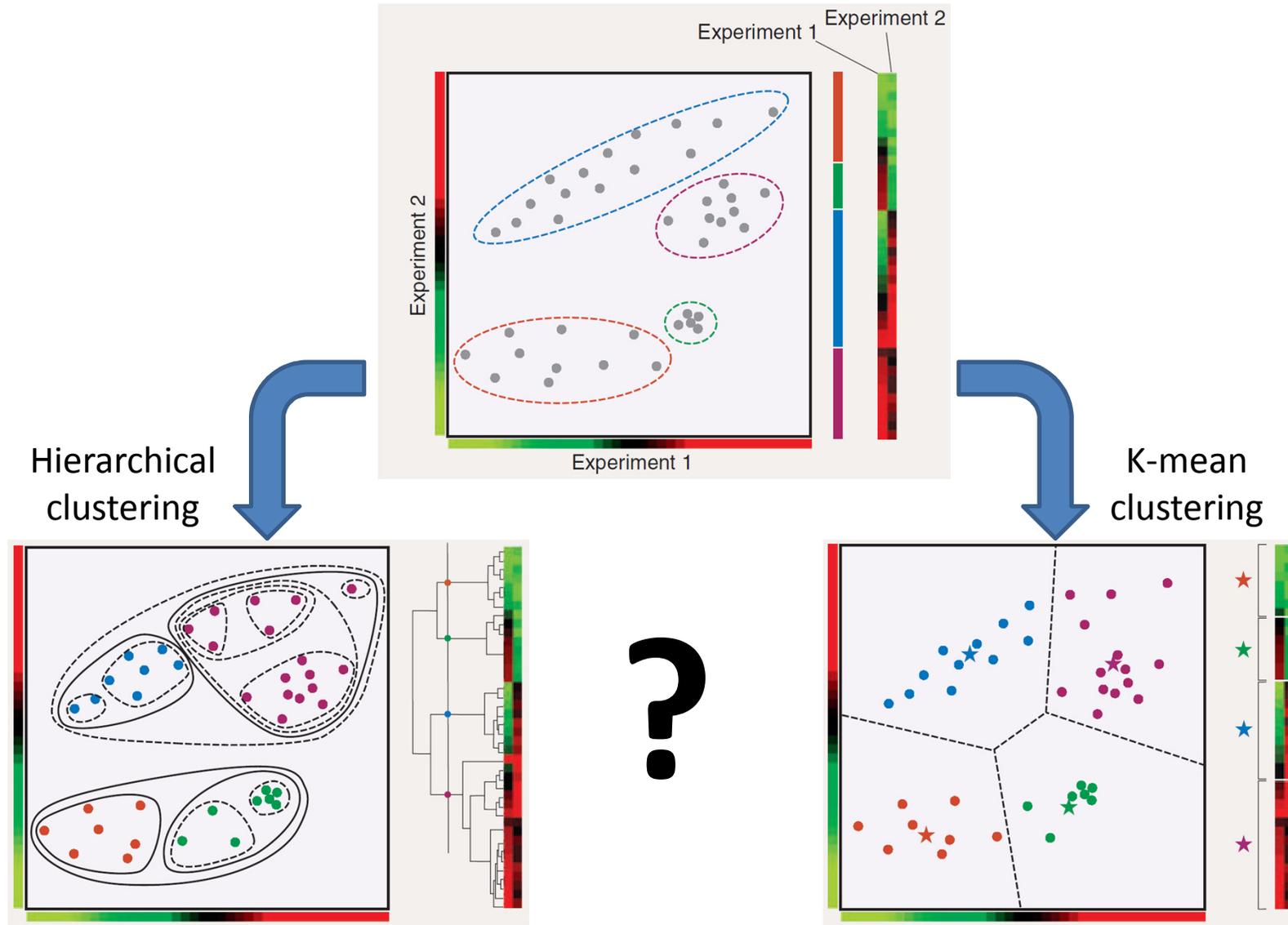
K-mean clustering: Summary

- 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

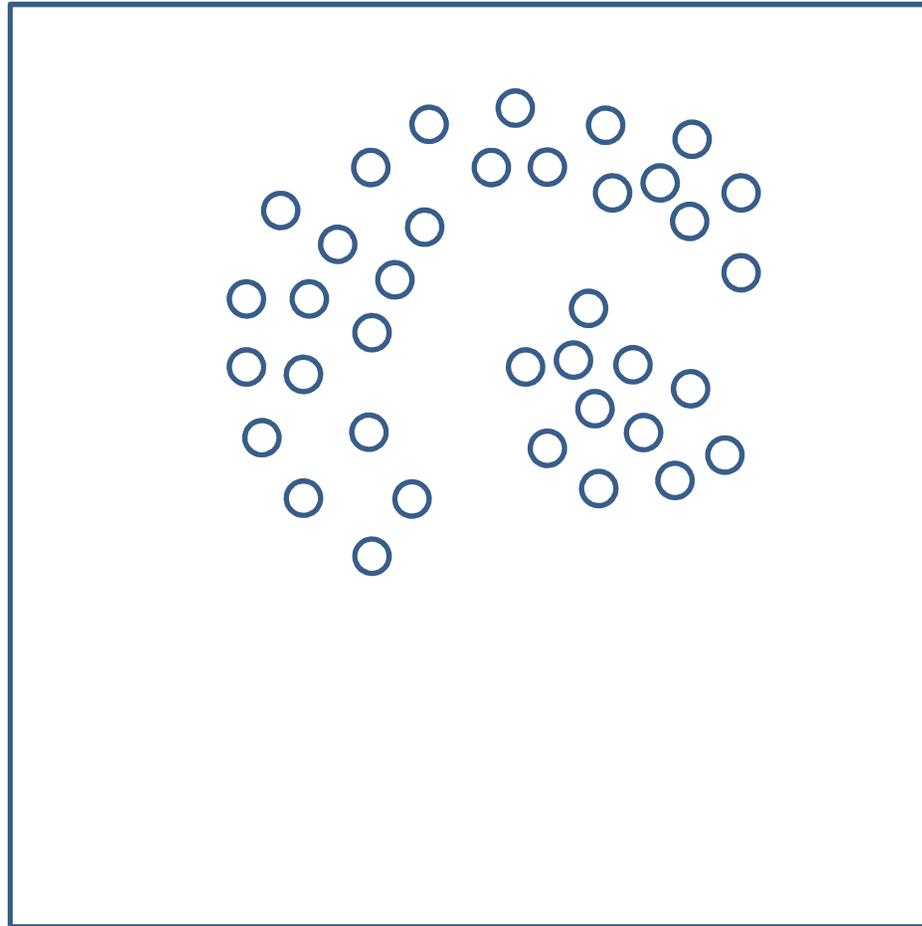
K-mean clustering: Variations

- 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”?

