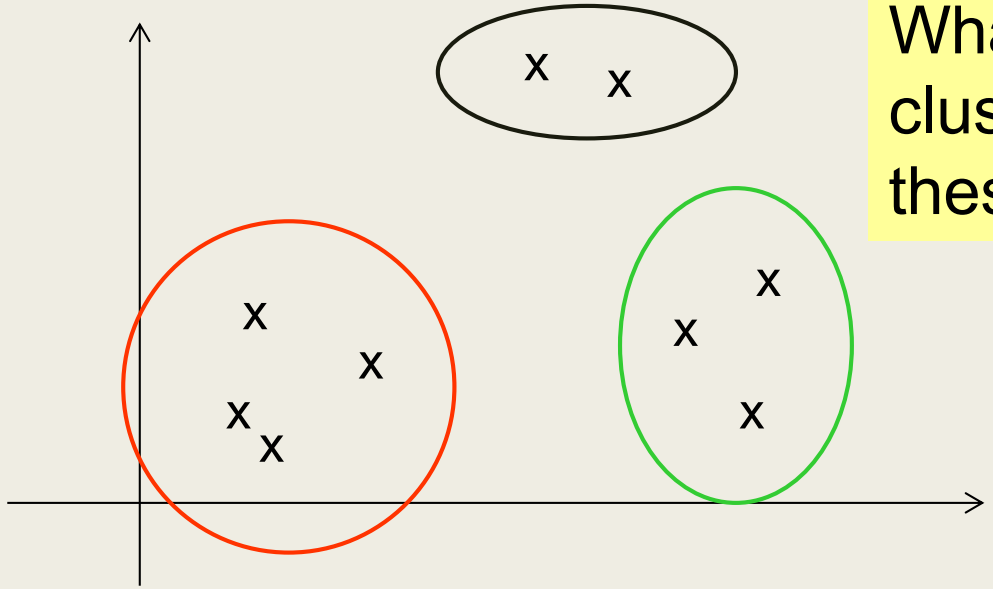
CLUSTERING

Dr. Vani Vasudevan

Introduction

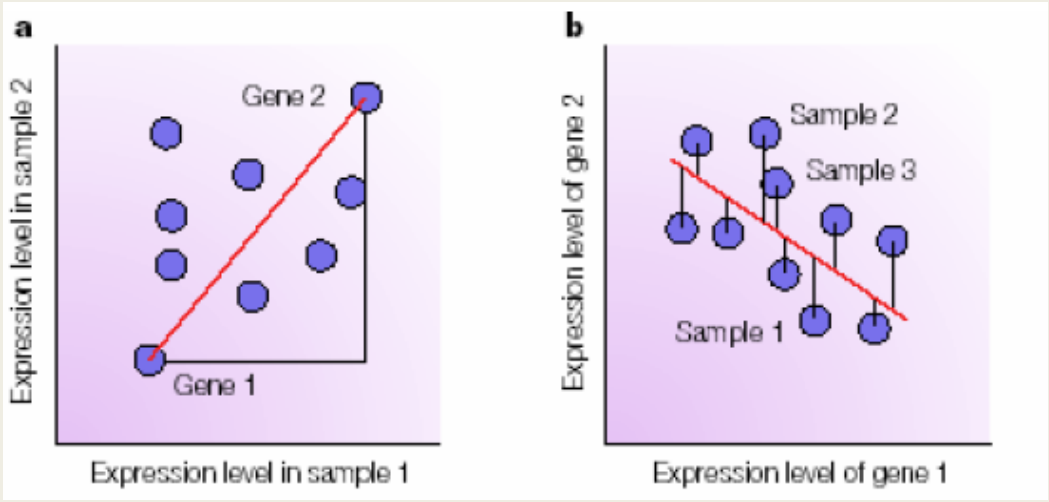
- The goal of clustering is to
  - *group data points that are close (or **similar**) to each other*
  - *identify such groupings (or clusters) in an **unsupervised** manner*
    - Unsupervised: no information is provided to the algorithm on which data points belong to which clusters
- Example



What should the clusters be for these data points?

Measuring Similarity

- When trying to group together objects that are similar, we need:
  1. ***Distance Metric*** –
     which define the meaning of similarity/dissimilarity



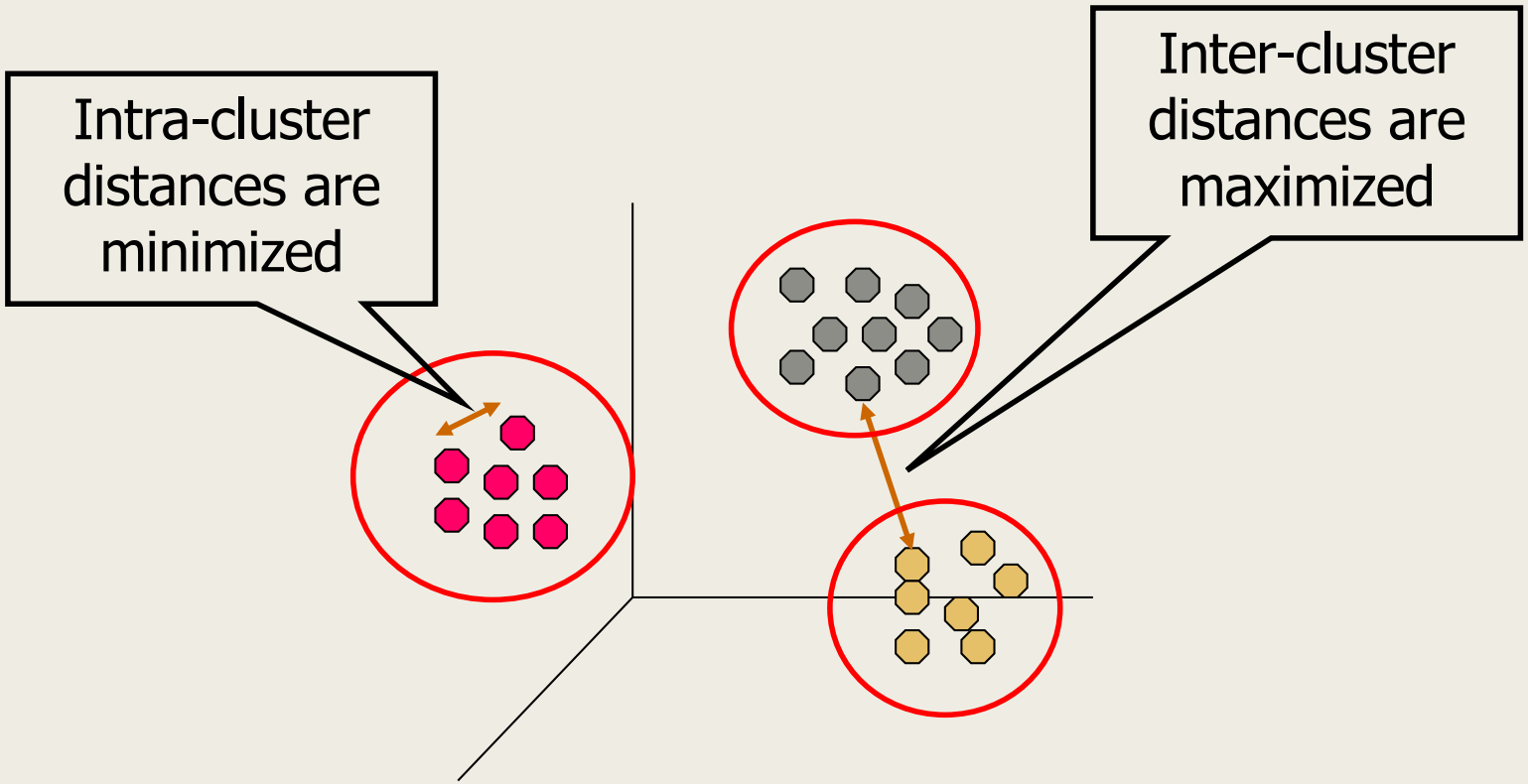
a) Two conditions and n genes

b) Two genes and n conditions

What Is Good Clustering?

- A good clustering method will produce high quality clusters with
  - *high intra-class similarity*
  - *low inter-class similarity*
- The quality of a clustering result depends on both the similarity measure used by the method and its implementation.
- The quality of a clustering method is also measured by its ability to discover some or all of the hidden patterns.

Intra-cluster and Inter-cluster distances

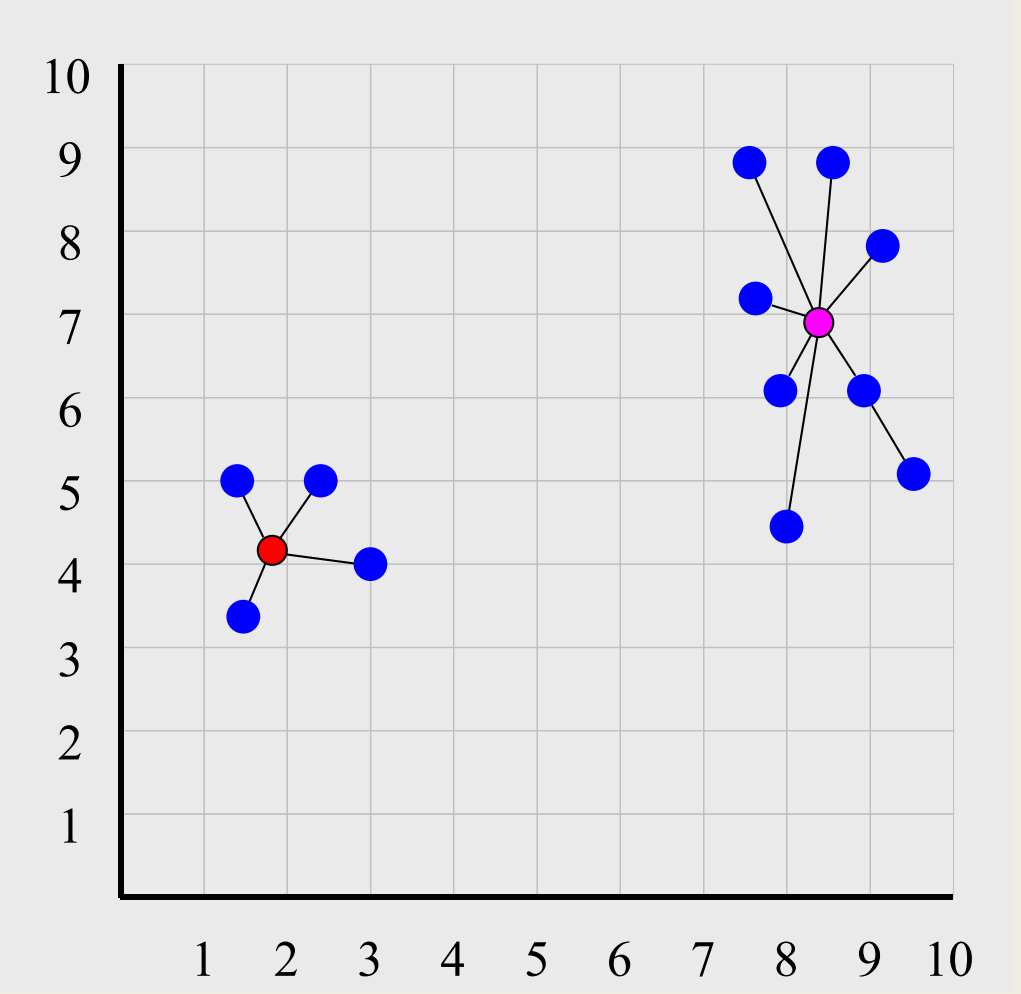


Squared Error

$$se_{K_i} = \sum_{j=1}^{m} \|t_{ij} - C_k\|^2$$

$$se_K = \sum_{j=1}^{k} se_{K_j}$$

Objective Function



Major Types of Clustering Algorithms

- Partitioning:

  Partition the database into k clusters which are represented by representative objects of them

- Hierarchical:

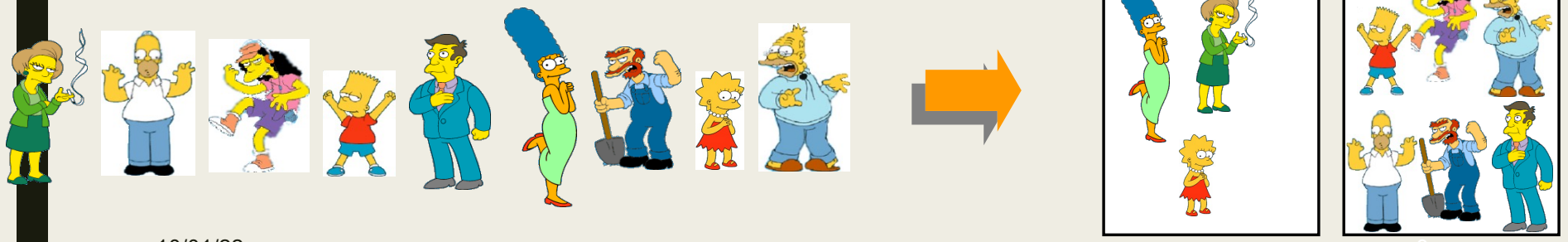
  Decompose the database into several levels of partitioning which are represented by dendrogram

- Density-based:

  Based on connectivity and density functions

Partitional Clustering

- Nonhierarchical, each instance is placed in exactly one of K nonoverlapping clusters.
- Since only one set of clusters is output, the user normally has to input the desired number of clusters K.



Partitional Clustering

Partitional algorithms determine all clusters at once.

They include:

- *K-means*
- *Nearest Neighbour*

Algorithm *k-means*

1. Decide on a value for *k*.

2. Initialize the *k* cluster centers (randomly, if necessary).

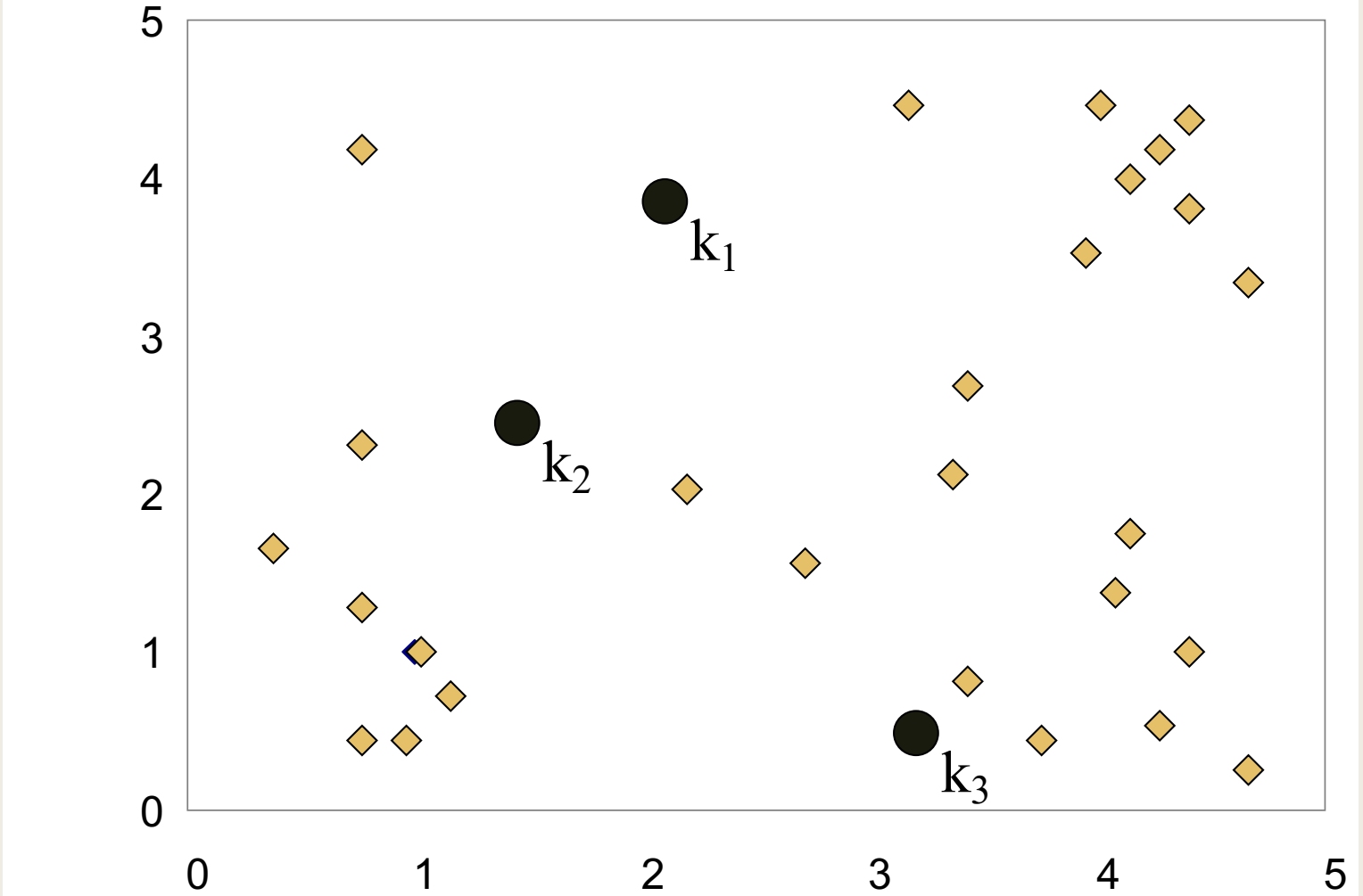
3. Decide the class memberships of the *N* objects by assigning them to the nearest cluster center.

4. Re-estimate the *k* cluster centers, by assuming the memberships found above are correct.

5. If none of the *N* objects changed membership in the last iteration, exit. Otherwise goto 3.

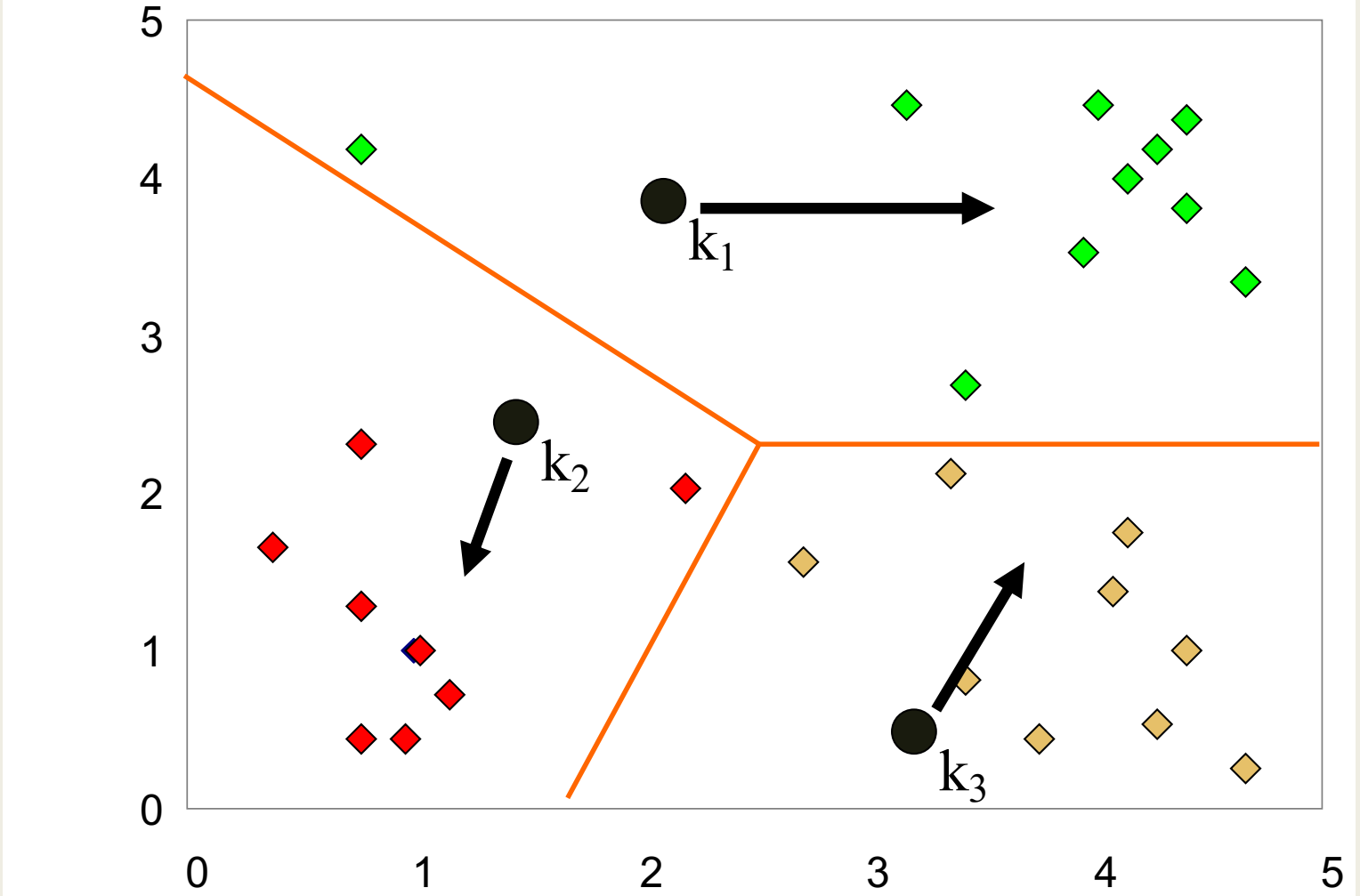
K-means Clustering: Step 1

Algorithm: k-means, Distance Metric: Euclidean Distance



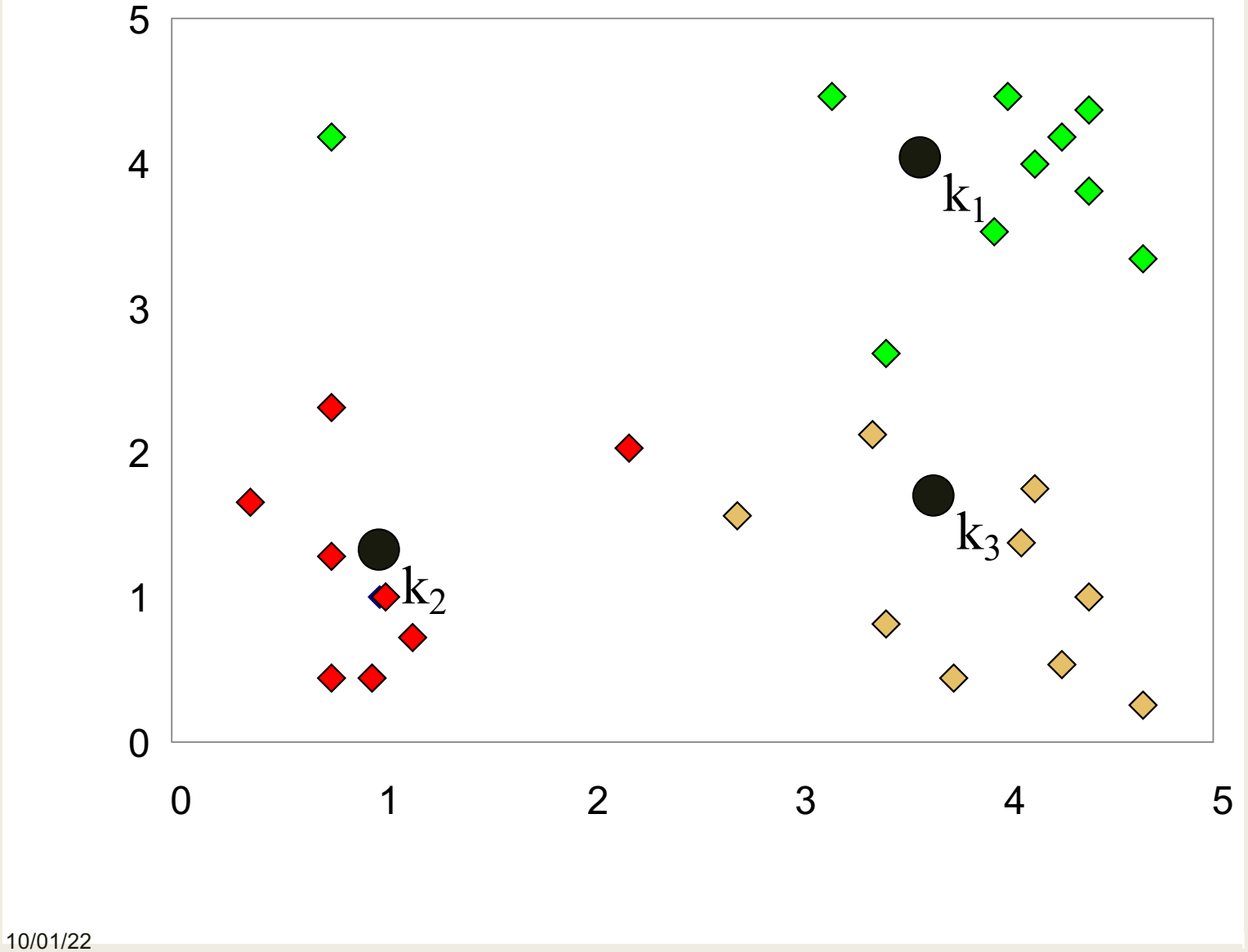
K-means Clustering: Step 2

Algorithm: k-means, Distance Metric: Euclidean Distance



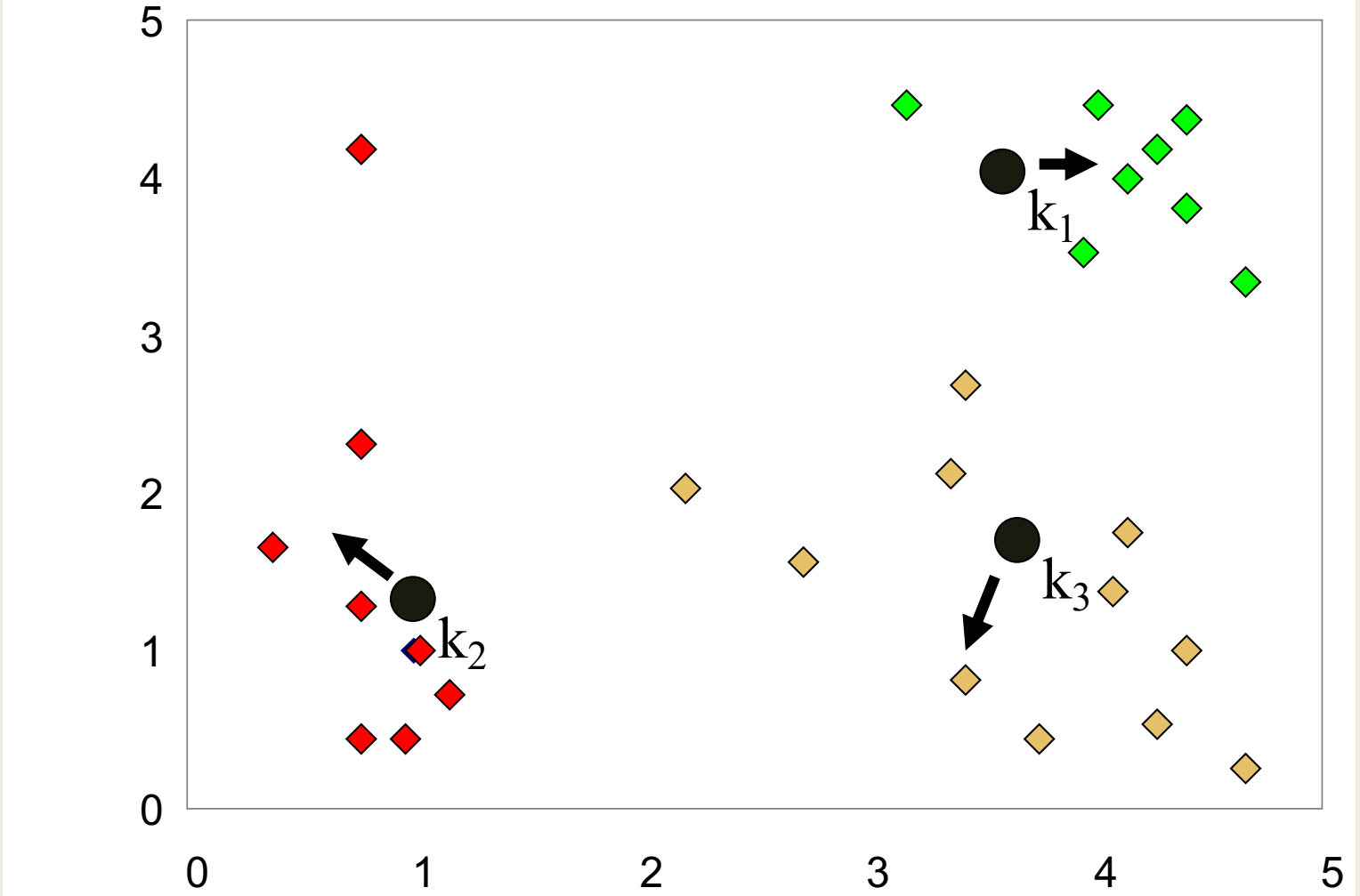
K-means Clustering: Step 3

Algorithm: k-means, Distance Metric: Euclidean Distance



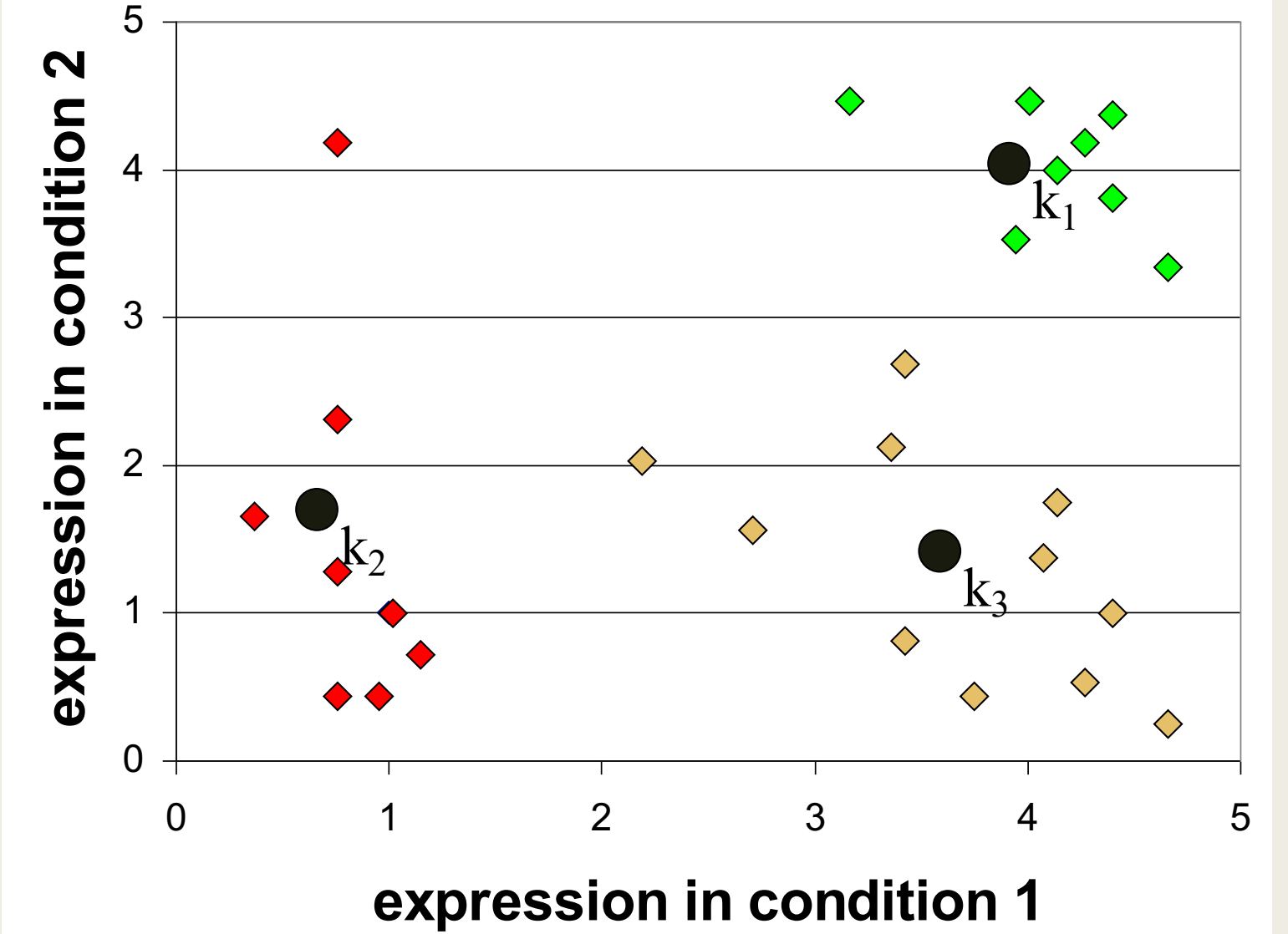
K-means Clustering: Step 4

Algorithm: k-means, Distance Metric: Euclidean Distance



K-means Clustering: Step 5

Algorithm: k-means, Distance Metric: Euclidean Distance

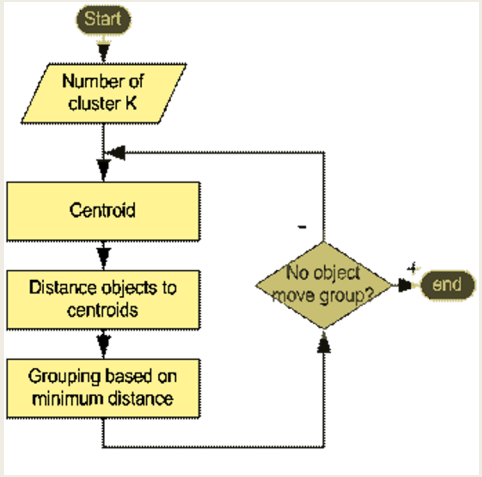


K-MEANS CLUSTERING

- The **k-means algorithm** is an algorithm to cluster *n* objects based on attributes into *k* partitions, where $k < n$.

- It assumes that the object attributes form a vector space.

How the K-Mean Clustering algorithm works?



How the K-Mean Clustering algorithm works?

- **Step 1:** Begin with a decision on the value of k = number of clusters .
- **Step 2**: Put any initial partition that classifies the data into k clusters. You may assign the training samples randomly, or systematically as the following:

  1. Take the first k training sample as single-element clusters
  2. Assign each of the remaining (N-k) training sample to the cluster with the nearest centroid. After each assignment, recompute the centroid of the gaining cluster.

How the K-Mean Clustering algorithm works?

- **Step 3:** Take each sample in sequence and compute its distance from the centroid of each of the clusters. If a sample is not currently in the cluster with the closest centroid, switch this sample to that cluster and update the centroid of the cluster gaining the new sample and the cluster losing the sample.
- **Step 4 .** Repeat step 3 until convergence is achieved, that is until a pass through the training sample causes no new assignments.

K-Means

- K-Means is relatively an efficient method. However, we need to specify the number of clusters, in advance and the final results are sensitive to initialization and often terminates at a local optimum
- Unfortunately, there is no global theoretical method to find the optimal number of clusters.
- A practical approach is to compare the outcomes of multiple runs with different *k* and choose the best one based on a predefined criterion.
- In general, a large *k* probably decreases the error but increases the risk of overfitting.

Example 1…

- Suppose we want to group the visitors to a website using just their age (a one-dimensional space) as follows:

15,15,16,19,19,20,20,21,22,28,35,40,41,42,43,44,60,61,65

Initial clusters:

Centroid (C1) = 16 [16]

Centroid (C2) = 22 [22]

Iteration 1:

C1 = 15.33 [15,15,16]

C2 = 36.25 [19,19,20,20,21,22,28,35,40,41,42,43,44,60,61,65]

Iteration 2:

C1 = 18.56 [15,15,16,19,19,20,20,21,22]

C2 = 45.90 [28,35,40,41,42,43,44,60,61,65]

Example 1

Iteration 3:

C1 = 19.50 [15,15,16,19,19,20,20,21,22,28]

C2 = 47.89 [35,40,41,42,43,44,60,61,65]

Iteration 4:

C1 = 19.50 [15,15,16,19,19,20,20,21,22,28]

C2 = 47.89 [35,40,41,42,43,44,60,61,65]

No change between iterations 3 and 4 has been noted. By using clustering, 2 groups have been identified 15-28 and 35-65. The initial choice of centroids can affect the output clusters, so the algorithm is often run multiple times with different starting conditions in order to get a fair view of what the clusters should be.

Example 2 (using K=2)

Individual	Variable 1	Variable 2
1	1.0	1.0
2	1.5	2.0
3	3.0	4.0
4	5.0	7.0
5	3.5	5.0
6	4.5	5.0
7	3.5	4.5

Example 2...

Step 1:

Initialization: Randomly we choose following two centroids (k=2) for two clusters.In this case the 2 centroid are: m1=(1.0,1.0) and m2=(5.0,7.0).

Individual	Variable 1	Variable 2
1	1.0	1.0
2	1.5	2.0
3	3.0	4.0
4	5.0	7.0
5	3.5	5.0
6	4.5	5.0
7	3.5	4.5

	Individual	Mean Vector
Group 1	1	(1.0, 1.0)
Group 2	4	(5.0, 7.0)

Example 2…

Step 2:

- Thus, we obtain two clusters containing:

  {1,2,3} and {4,5,6,7}.

- Their new centroids are:

$$m_1 = \left(\frac{1}{3}(1.0+1.5+3.0), \frac{1}{3}(1.0+2.0+4.0)\right) = (1.83, 2.33)$$

$$m_2 = \left(\frac{1}{4}(5.0+3.5+4.5+3.5), \frac{1}{4}(7.0+5.0+5.0+4.5)\right)$$

$$= (4.12, 5.38)$$

Individual	Centroid 1	Centroid 2
1	0	7.21
2 (1.5, 2.0)	1.12	6.10
3	3.61	3.61
4	7.21	0
5	4.72	2.5
6	5.31	2.06
7	4.30	2.92

$$d(m_1,2) = \sqrt{|1.0-1.5|^2 + |1.0-2.0|^2} = 1.12$$
$$d(m_2,2) = \sqrt{|5.0-1.5|^2 + |7.0-2.0|^2} = 6.10$$

Example 2…

Step 3:

- Now using these centroids we compute the Euclidean distance of each object, as shown in table.

Individual	Centroid 1	Centroid 2
1	1.57	5.38
2	0.47	4.28
3	2.04	1.78
4	5.64	1.84
5	3.15	0.73
6	3.78	0.54
7	2.74	1.08

- Therefore, the new clusters are:

  {1,2} and {3,4,5,6,7}

- Next centroids are: m1=(1.25,1.5) and m2 = (3.9,5.1)

Example 2

- Step 4 :

  The clusters obtained are:

  {1,2} and {3,4,5,6,7}

- Therefore, there is no change in the cluster.
- Thus, the algorithm comes to a halt here and final result consist of 2 clusters {1,2} and {3,4,5,6,7}.

Individual	Centroid 1	Centroid 2
1	0.56	5.02
2	0. 56	3.92
3	3.05	1.42
4	6.66	2.20
5	4.16	0.41
6	4.78	0.61
7	3.75	0.72

Comments on the *K-Means* Method

Strength

- *Relatively easy algorithm*
- *Relatively efficient: O(tkn), where n is # objects, k is # clusters, and t is # iterations. Normally, k, t << n.*

Weakness

- *Applicable only when mean is defined, then what about categorical data?*
- *Need to specify k, the number of clusters, in advance*
- *Unable to handle noisy data and outliers*

Validity of clusters

- Why validity of clusters?
  - *Given some data, any clustering algorithm generates clusters*
  - *So, we need to make sure the clustering results are valid and meaningful.*
- Measuring the validity of clustering results usually involve
  - *Optimality of clusters*
  - *Verification of meaning of clusters*

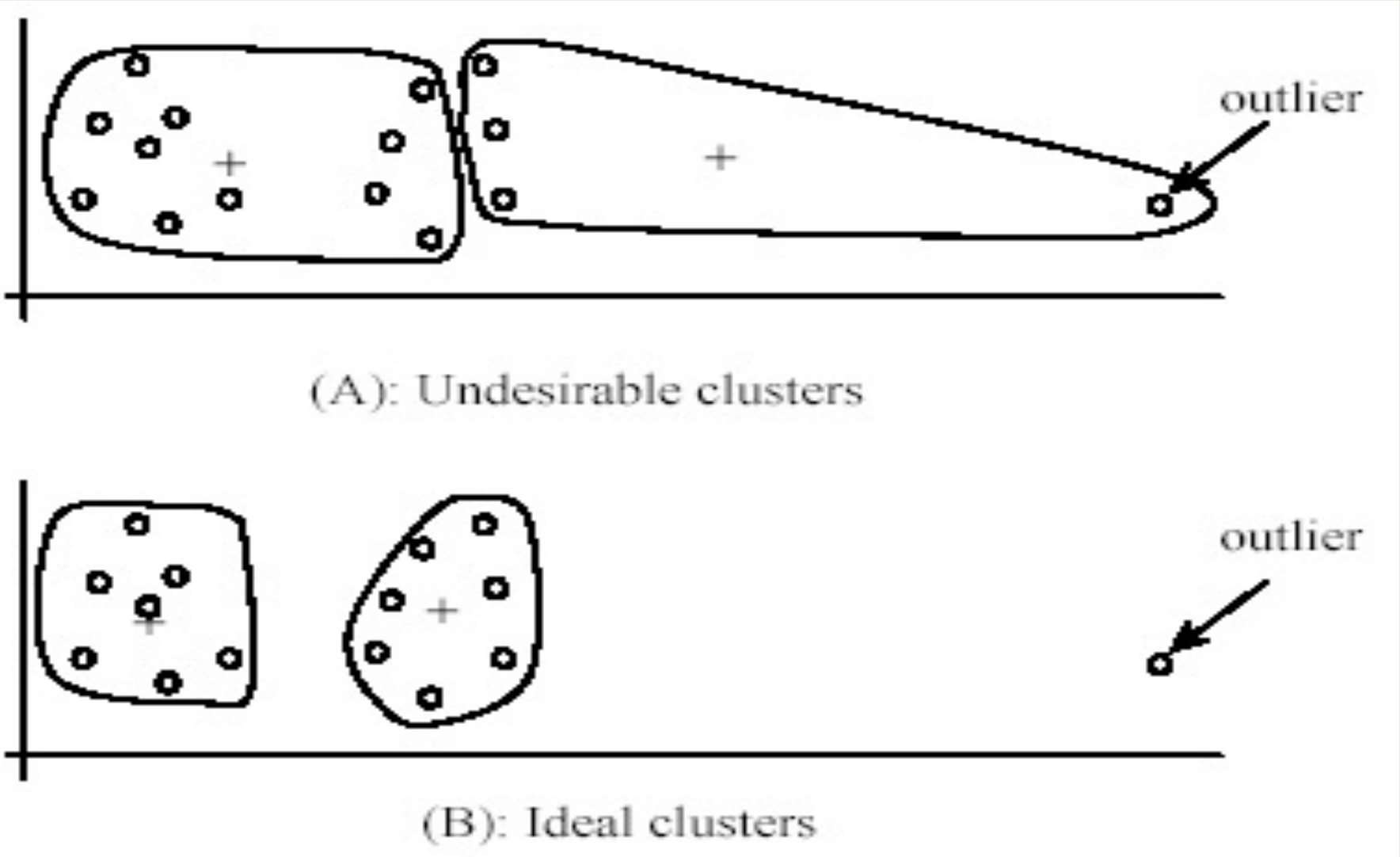
Optimality of clusters

- Optimal clusters should
  - *minimize distance **within** clusters (intracluster)*
  - *maximize distance **between** clusters (intercluster)*
- Example of intracluster measure
  - Squared error *se*

    where m_i is the mean of all instances in cluster c_i

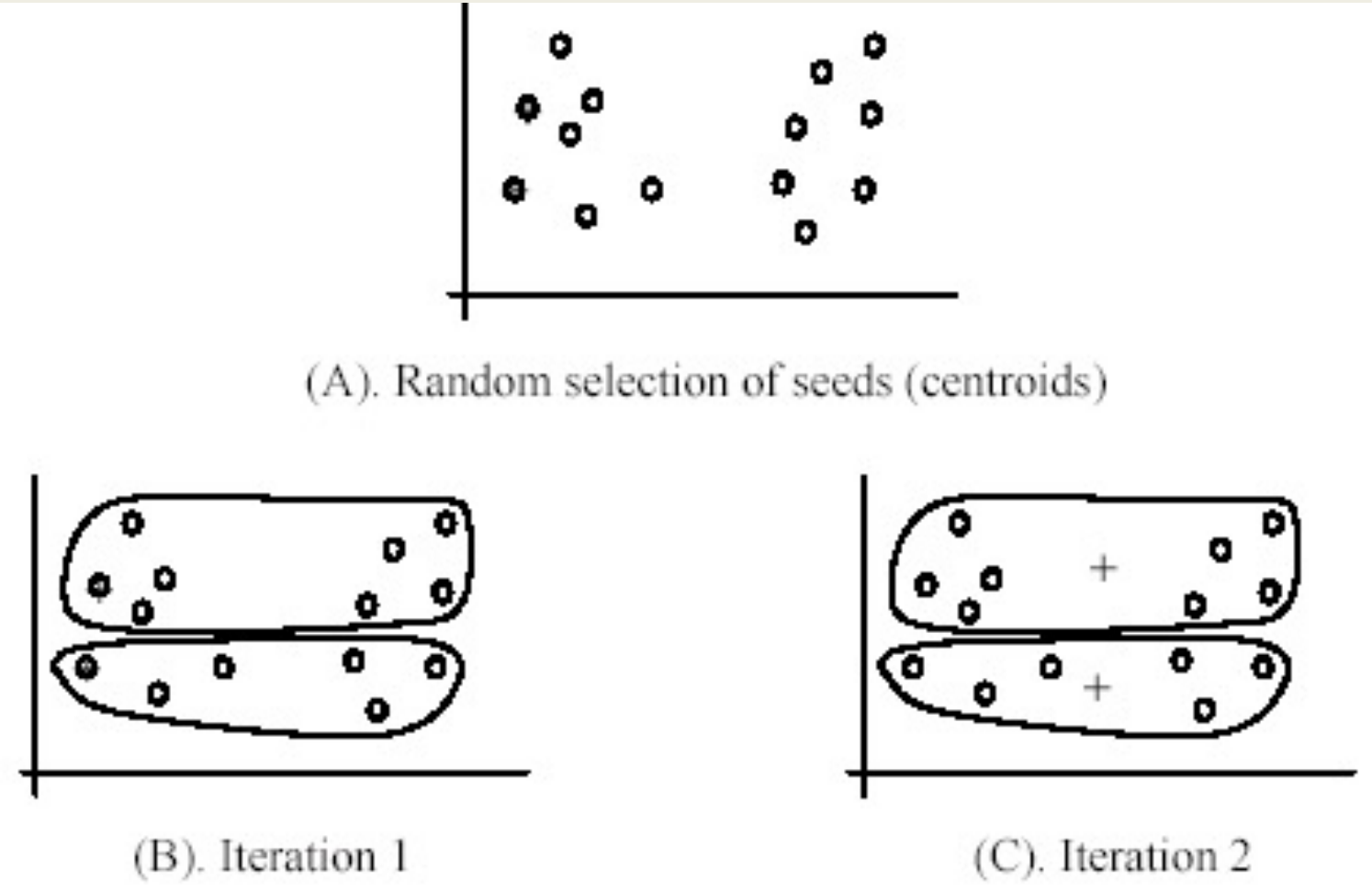
$$se = \sum_{i=1}^{k} \sum_{p \in c_i} \|p - m_i\|^2$$

Weaknesses of k-means: Problems with outliers



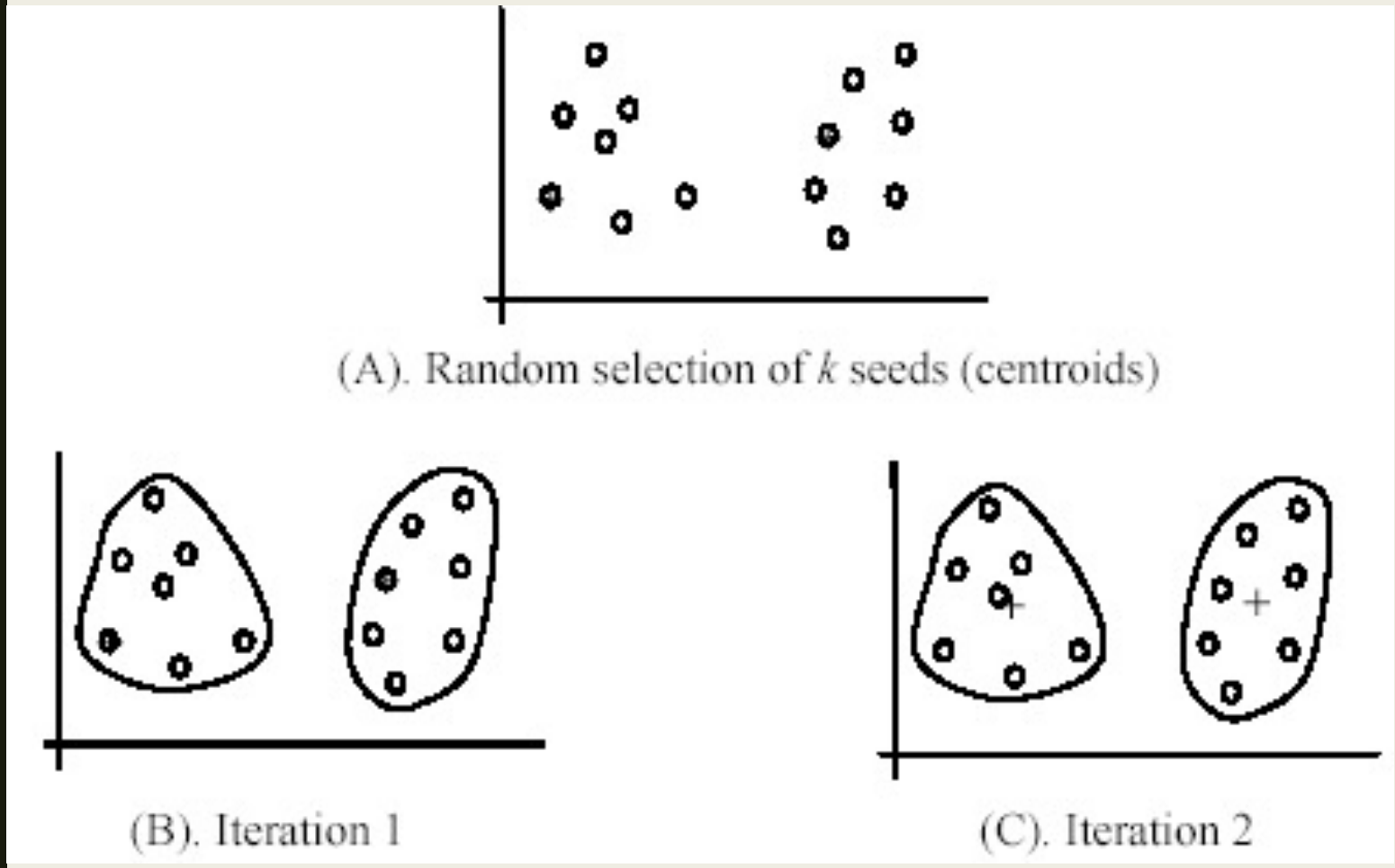
Weaknesses of k-means (cont ...)

- The algorithm is sensitive to initial seeds.



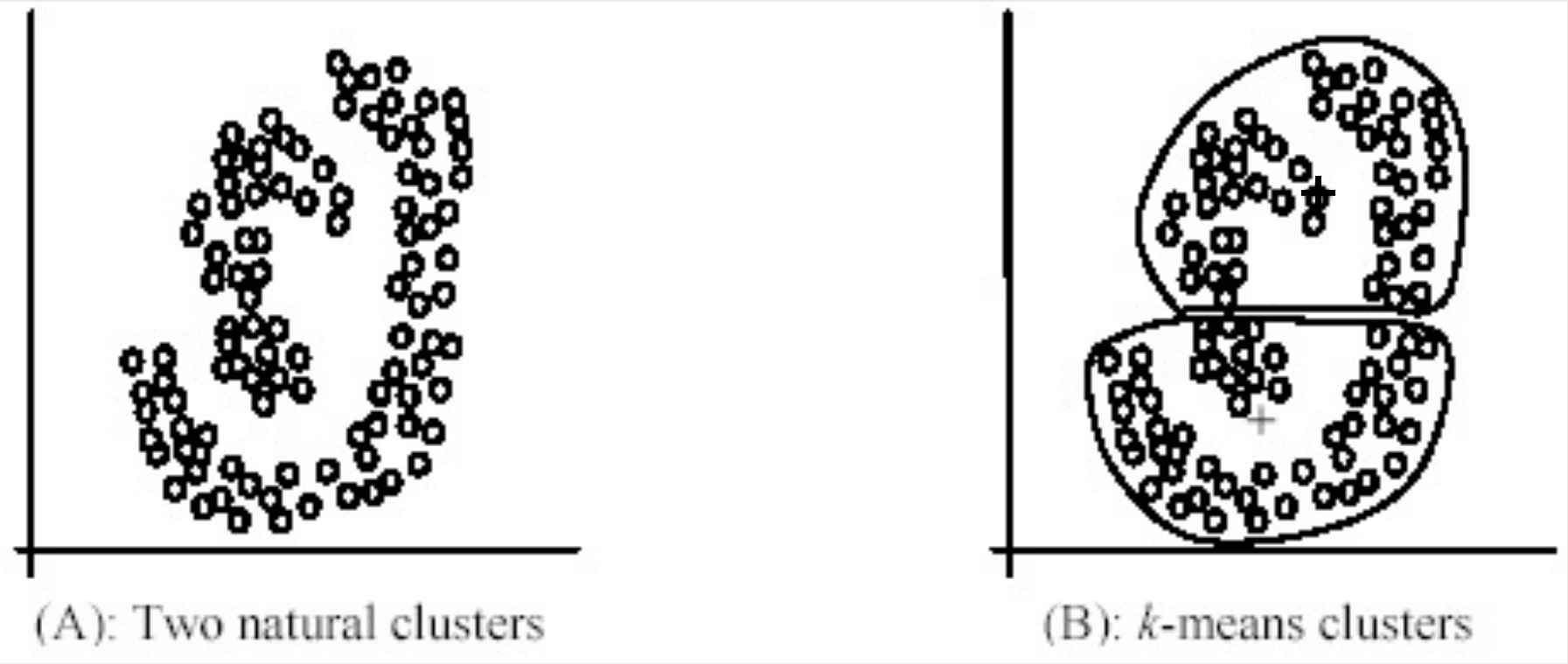
Weaknesses of k-means (cont ...)

- If we use different seeds: good results



Weaknesses of k-means (cont ...)

- The *k*-means algorithm is not suitable for discovering clusters that are not hyper-ellipsoids (or hyper-spheres).

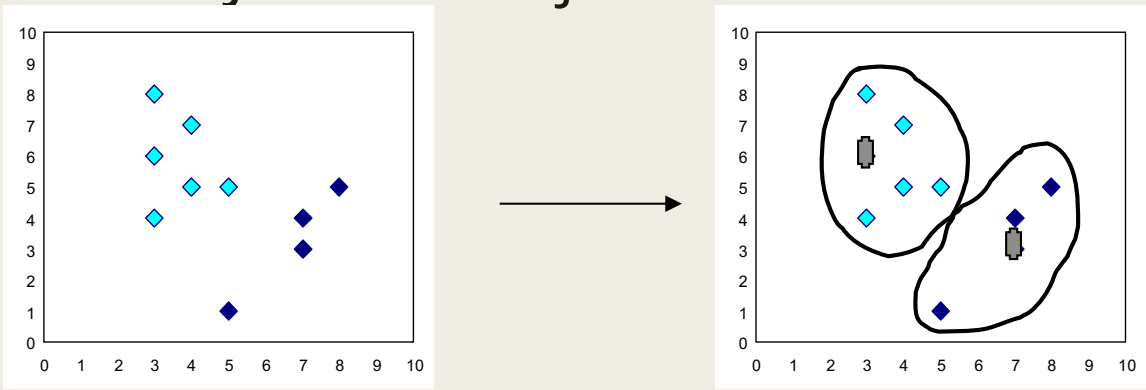


(A): Two natural clusters

(B): *k*-means clusters

What is the problem of k-Means Method?

- The k-means algorithm is sensitive to outliers !
  - *Since an object with an extremely large value may substantially distort the distribution of the data.*
- K-Medoids: Instead of taking the **mean** value of the object in a cluster as a reference point, **medoids** can be used, which is the **most centrally located** object in a cluster.



Termination conditions

- Several possibilities, e.g.,
  - *A fixed number of iterations.*
  - *Cluster partition unchanged.*
  - *Centroid positions don't change.*

K-means: summary

- Algorithmically, very simple to implement
- *K*-means converges, but it finds a local minimum of the cost function
- Works only for numerical observations
- *K* is a user input;
- Outliers can be considerable trouble to *K*-means

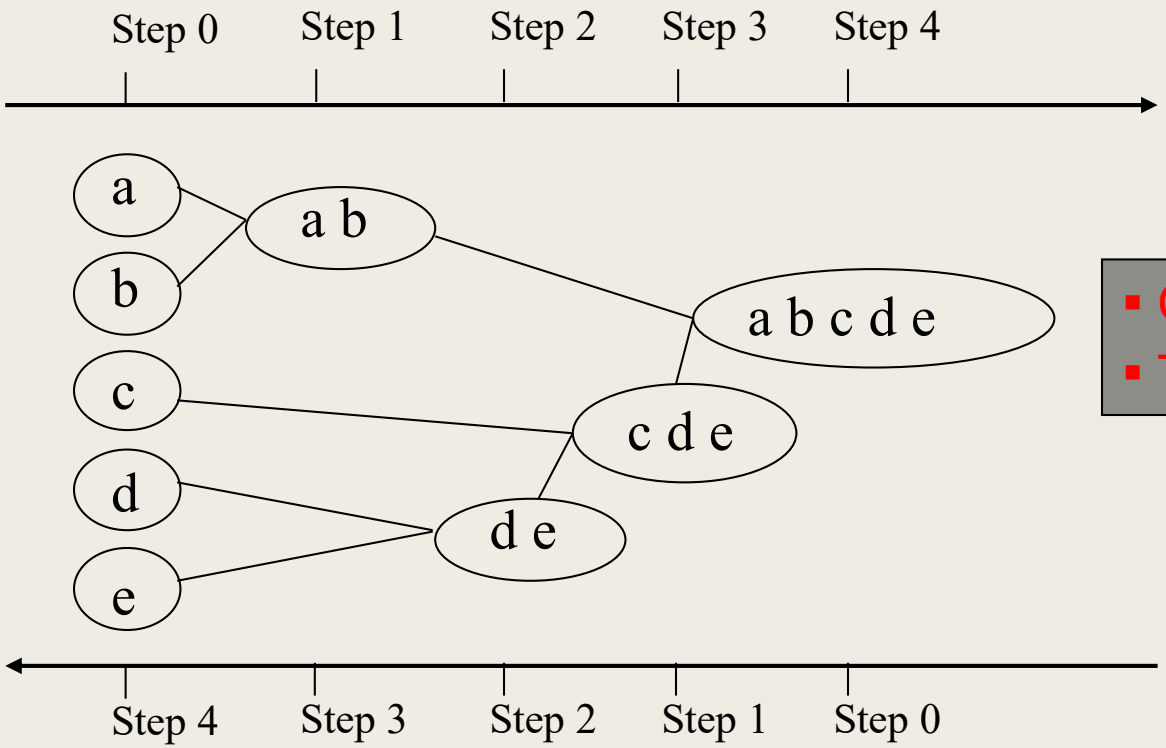
Hierarchical Clustering

- Two main types of hierarchical clustering
  - *Agglomerative:*
    - Start with the points as individual clusters
    - At each step, merge the closest pair of clusters until only one cluster (or k clusters) left
  - *Divisive:*
    - Start with one, all-inclusive cluster
    - At each step, split a cluster until each cluster contains a point (or there are k clusters)
- Traditional hierarchical algorithms use a similarity or distance matrix
  - *Merge or split one cluster at a time*

ntroduction

Illustrative Example

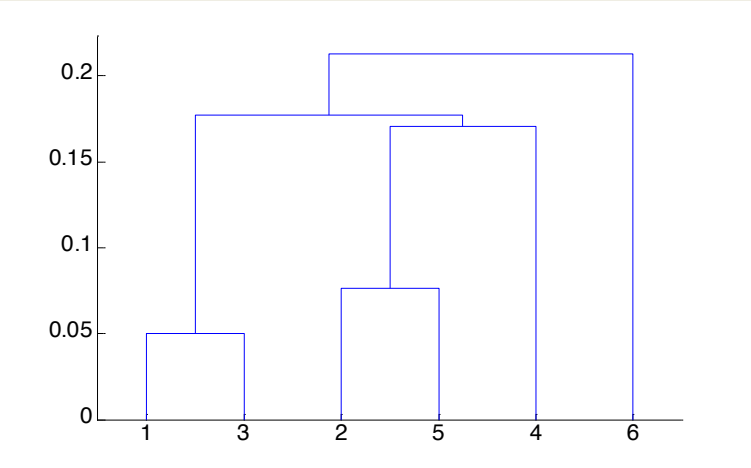
Agglomerative and divisive clustering on the data set {a, b, c, d ,e }

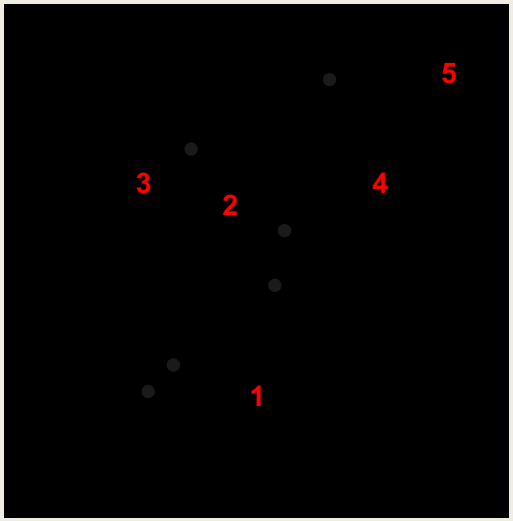


- Cluster distance
- Termination condition

Hierarchical Clustering

- Produces a set of nested clusters organized as a hierarchical tree
- Can be visualized as a dendrogram
  - *A tree like diagram that records the sequences of merges or splits*





Agglomerative Clustering Algorithm

- More popular hierarchical clustering technique

- Basic algorithm is straightforward

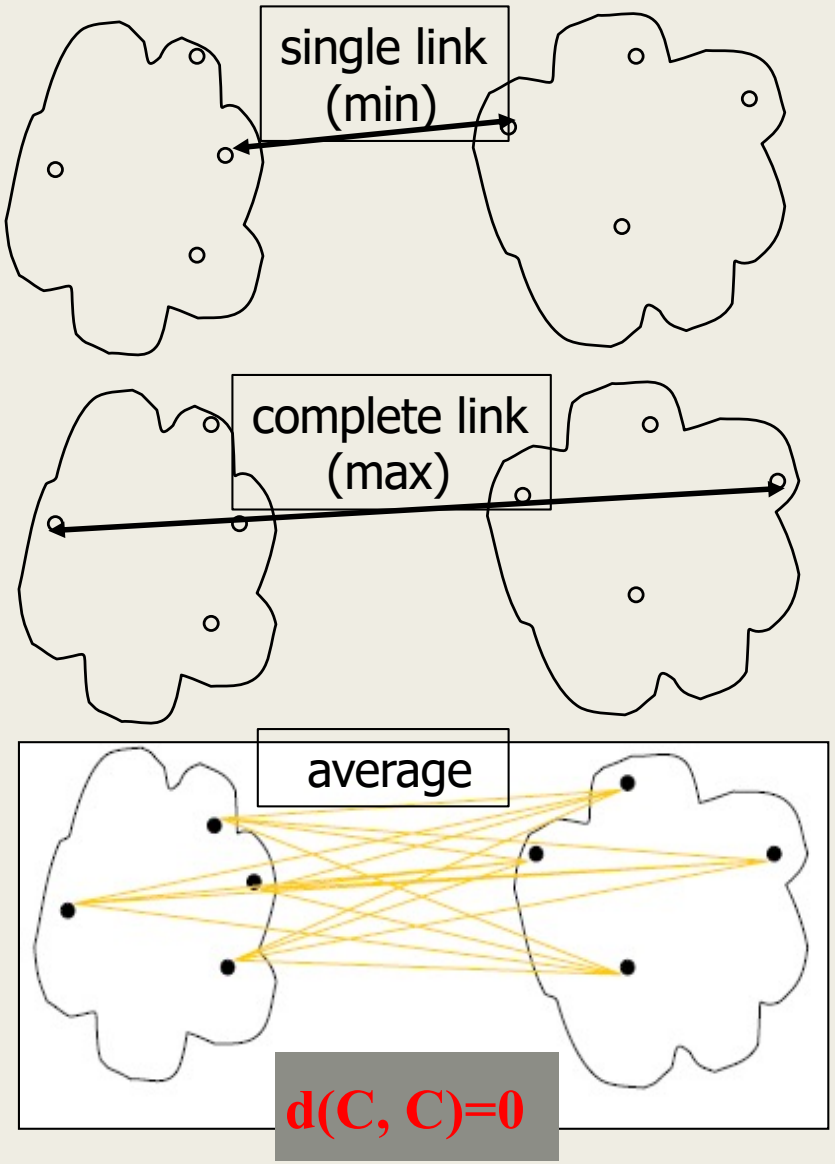
1. *Compute the proximity matrix*
2. *Let each data point be a cluster*
3. ***Repeat***
4. *Merge the two closest clusters*
5. *Update the proximity matrix*
6. ***Until** only a single cluster remains*

- Key operation is the computation of the proximity of two clusters
  - *Different approaches to defining the distance between clusters distinguish the different algorithms*

Cluster Distance Measures

- Single link: smallest distance between an element in one cluster and an element in the other, i.e., $d(C_i, C_j) = \min\{d(x_{ip}, x_{jq})\}$
- Complete link: largest distance between an element in one cluster and an element in the other, i.e., $d(C_i, C_j) = \max\{d(x_{ip}, x_{jq})\}$
- Average: avg distance between elements in one cluster and elements in the other, i.e.,

$d(C_i, C_j) = \text{avg}\{d(x_{ip}, x_{jq})\}$



Cluster Distance Measures

Example: Given a data set of five objects characterised by a single continuous feature, assume that there are two clusters: C_1: {a, b} and C_2: {c, d, e}.

	a	b	c	d	e
Feature	1	2	4	5	6

1. Calculate the distance matrix.

	a	b	c	d	e
a	0	1	3	4	5
b	1	0	2	3	4
c	3	2	0	1	2
d	4	3	1	0	1
e	5	4	2	1	0

2. Calculate three cluster distances between C_1 and C_2.

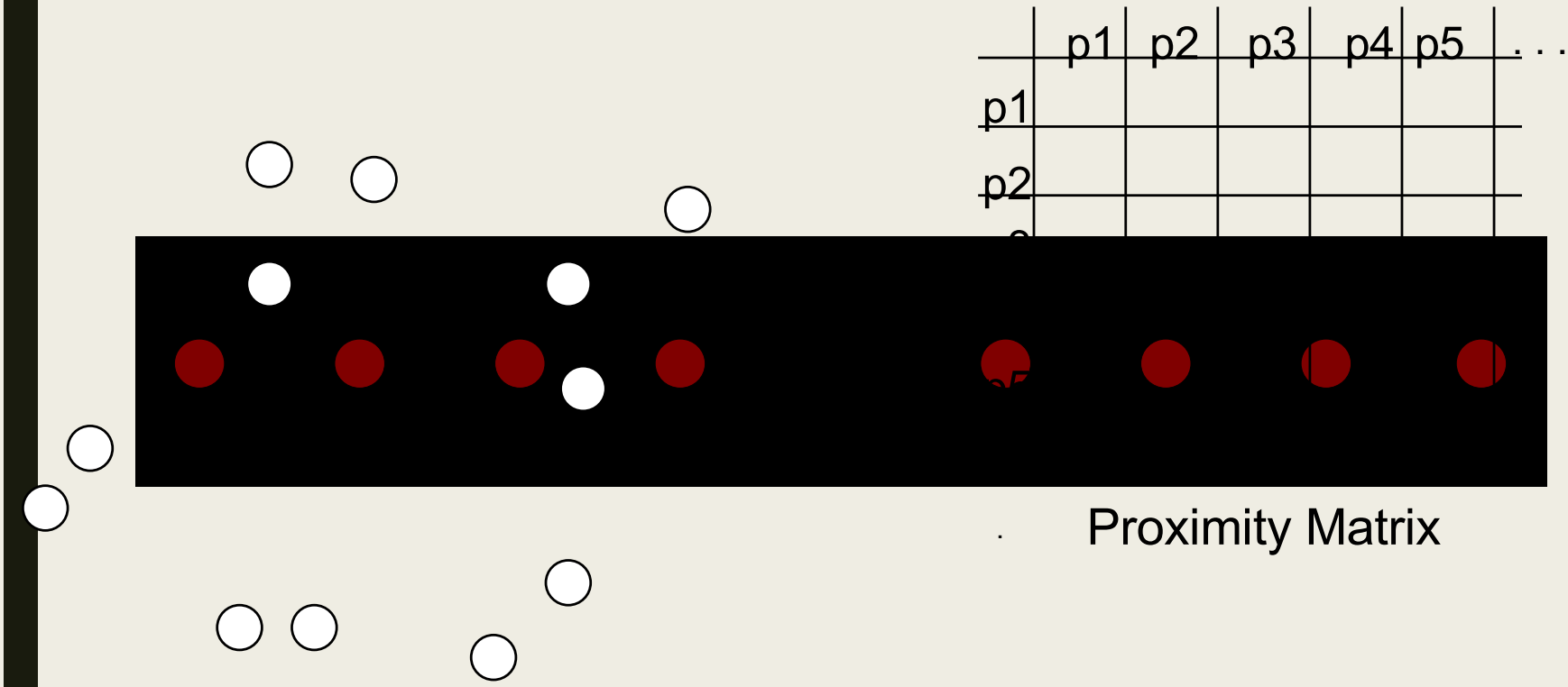
Single link

Complete link

Average

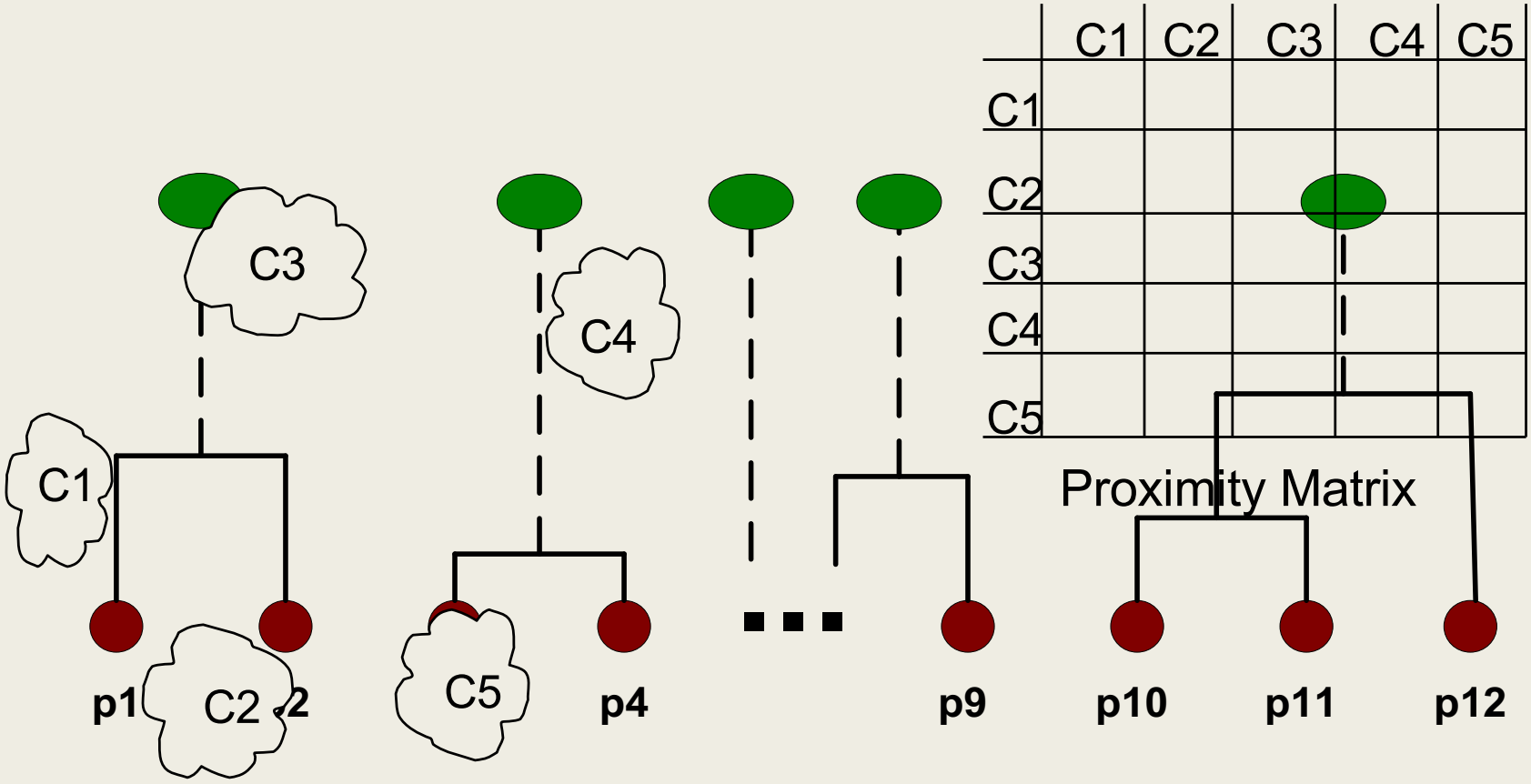
Starting Situation

- Start with clusters of individual points and a proximity matrix



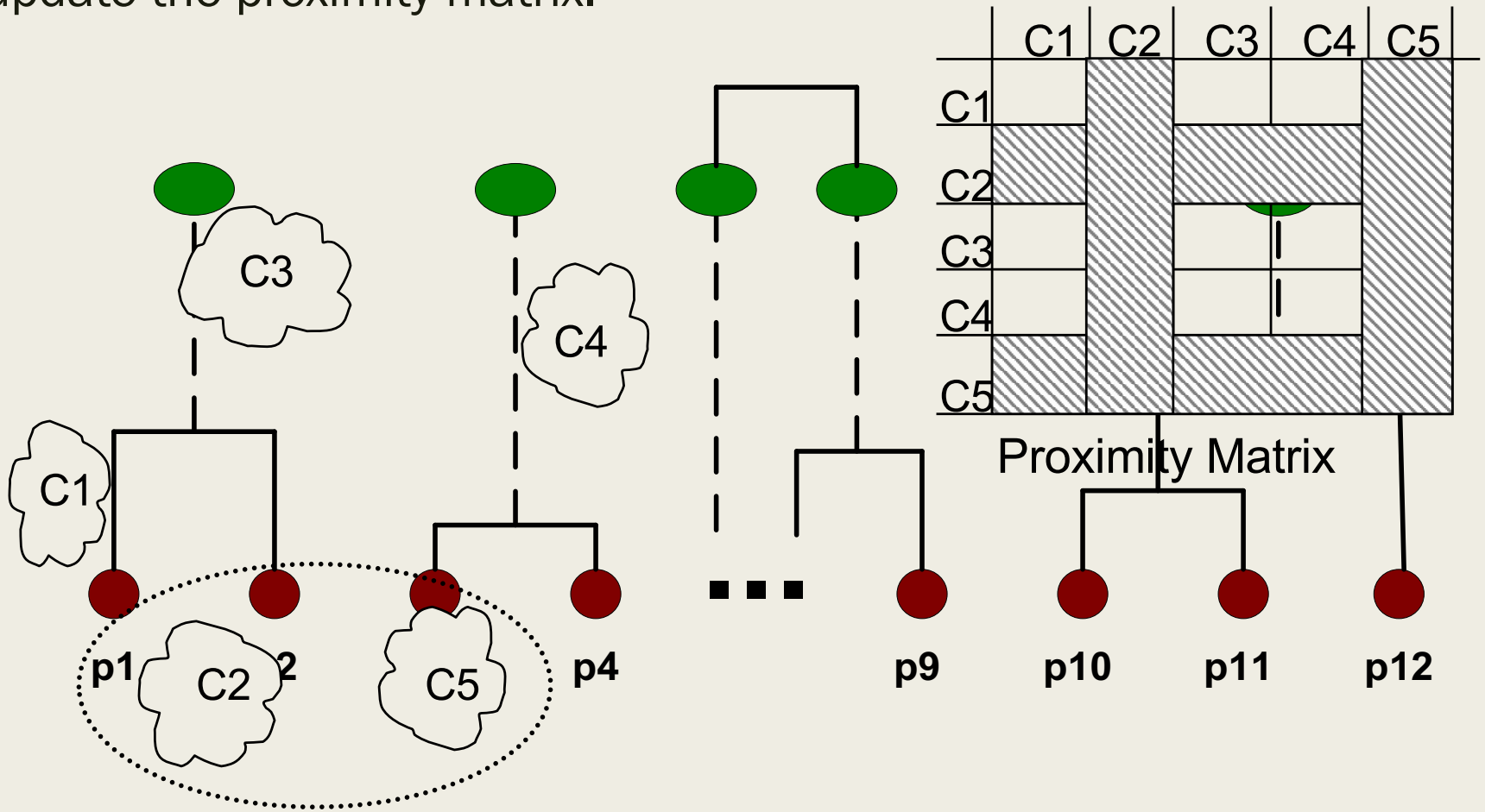
Intermediate Situation

- After some merging steps, we have some clusters



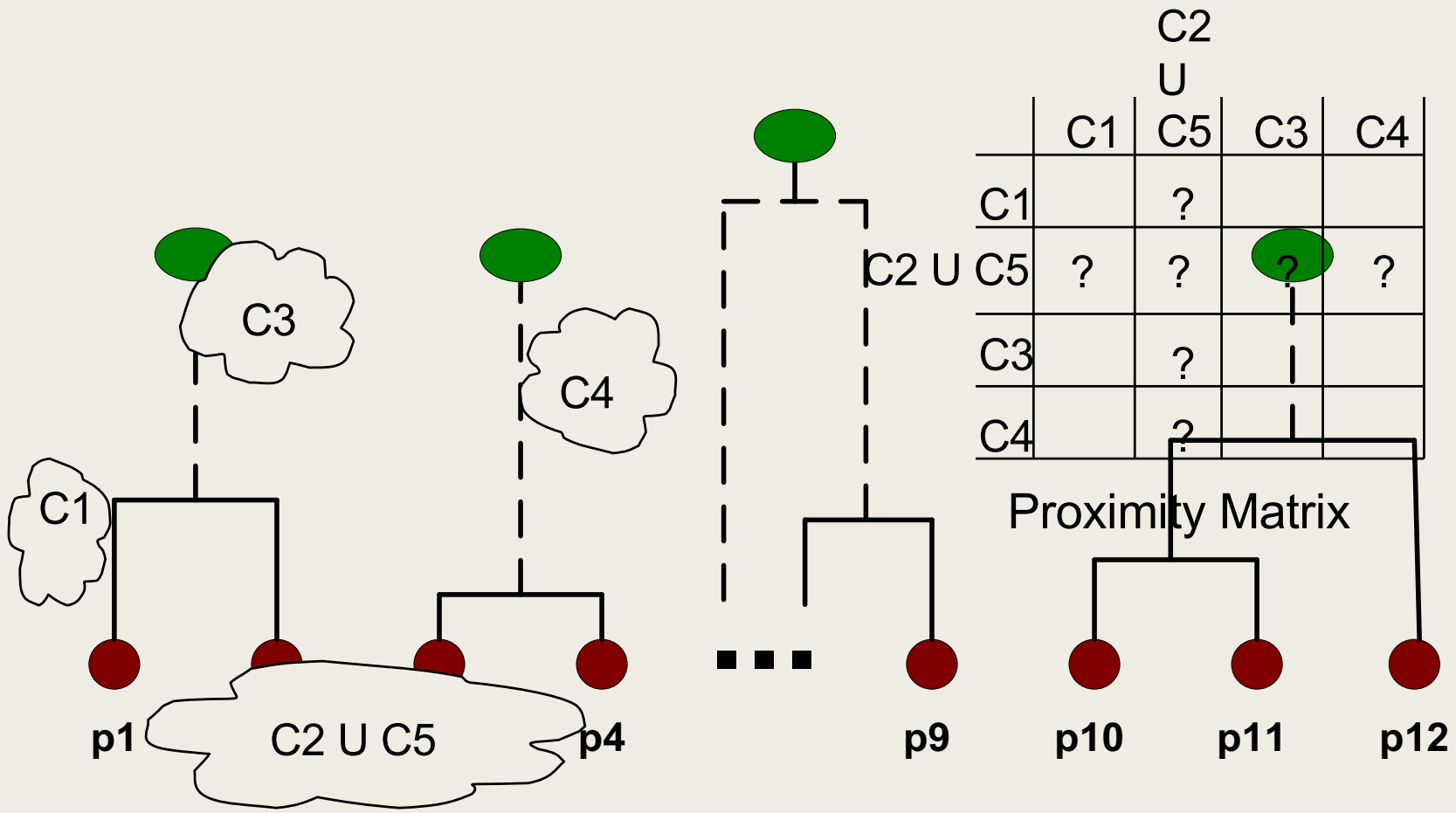
Intermediate Situation

- We want to merge the two closest clusters (C2 and C5) and update the proximity matrix.

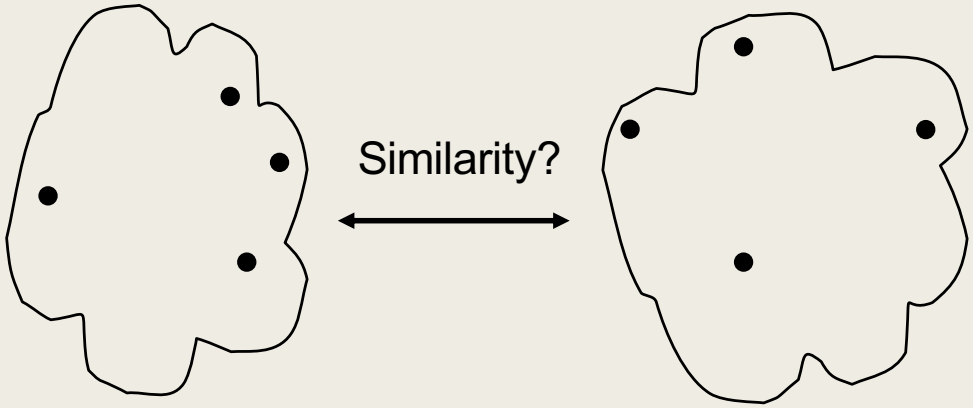


After Merging

- The question is “How do we update the proximity matrix?”



How to Define Inter-Cluster Similarity?

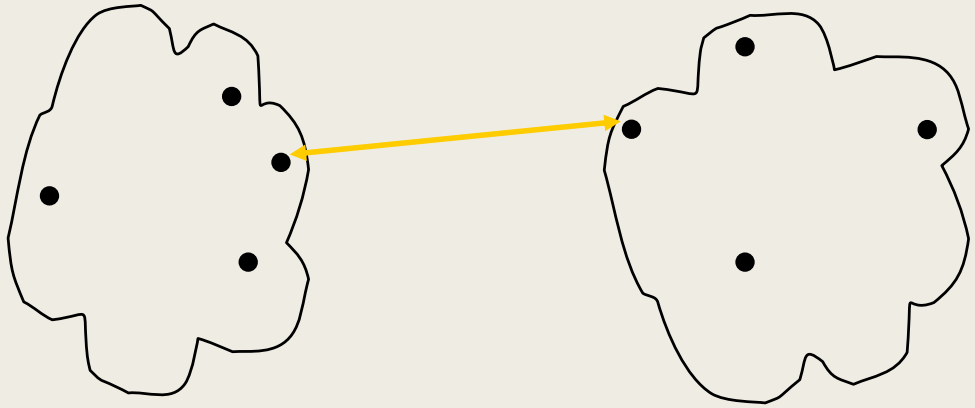


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

How to Define Inter-Cluster Similarity

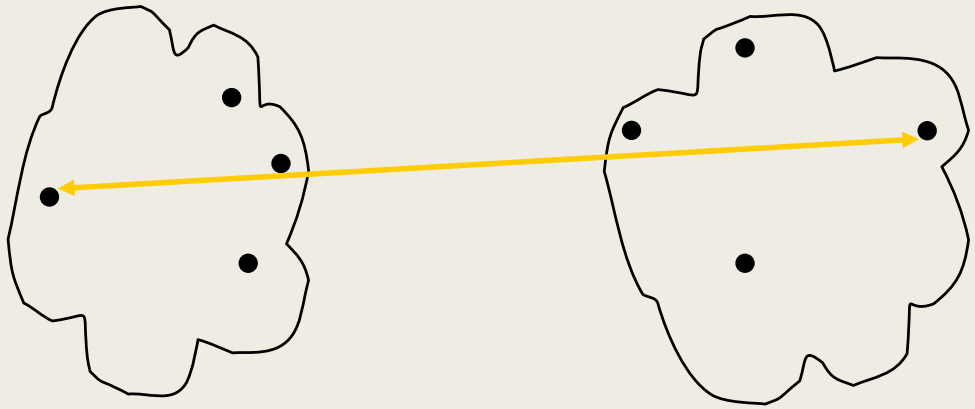


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

How to Define Inter-Cluster Similarity

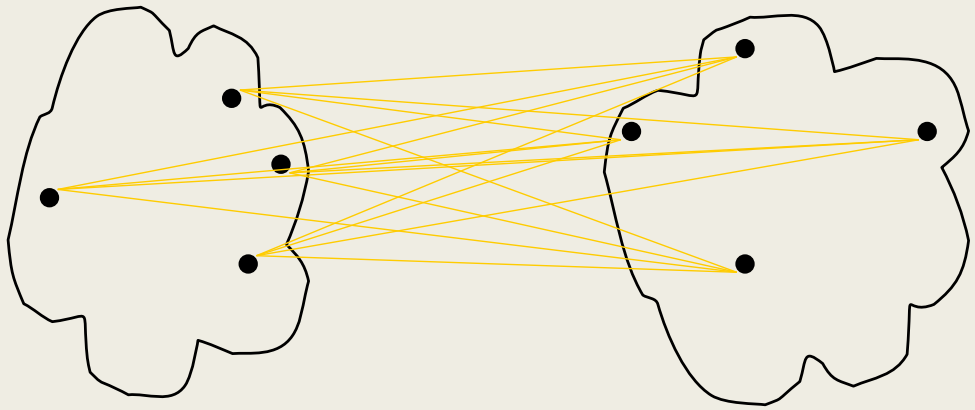


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

How to Define Inter-Cluster Similarity

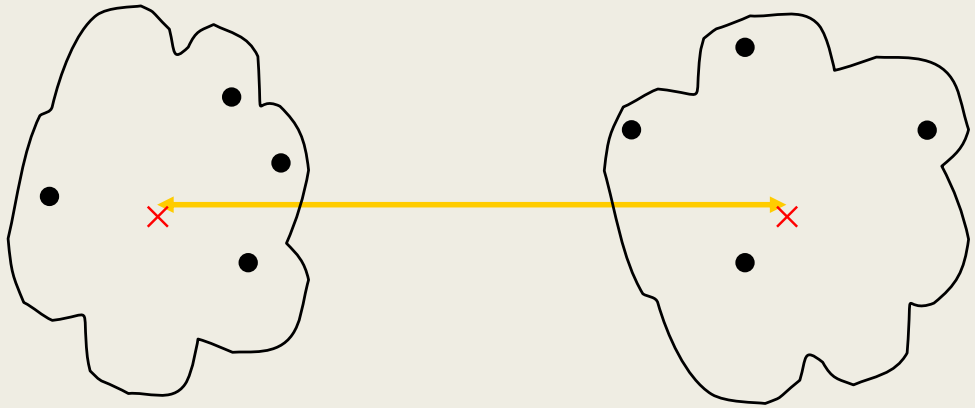


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

How to Define Inter-Cluster Similarity



	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

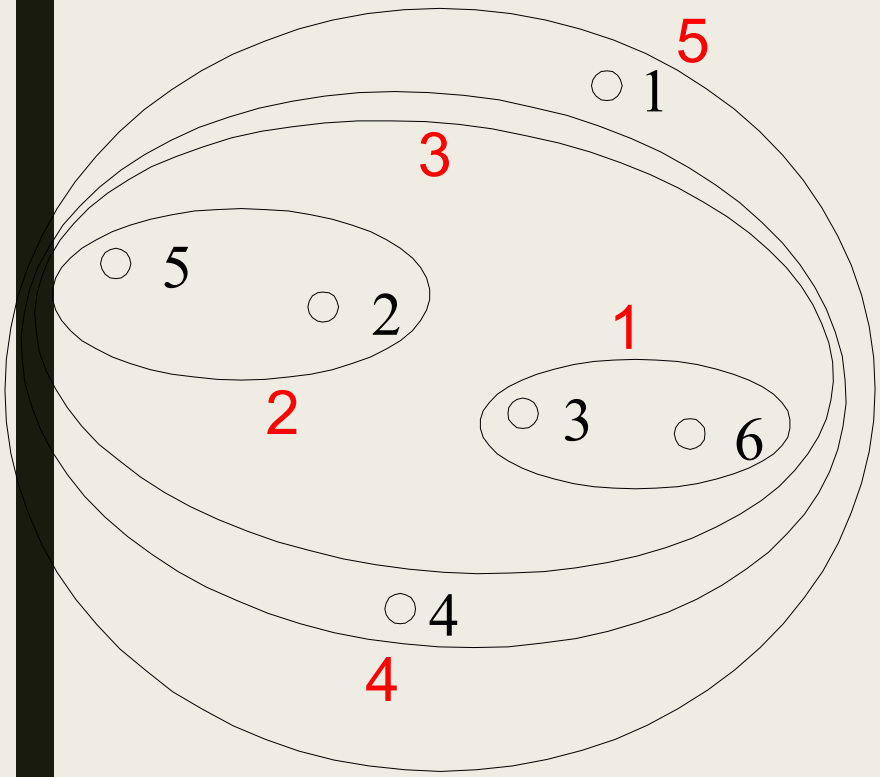
Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

Single Link – Complete Link

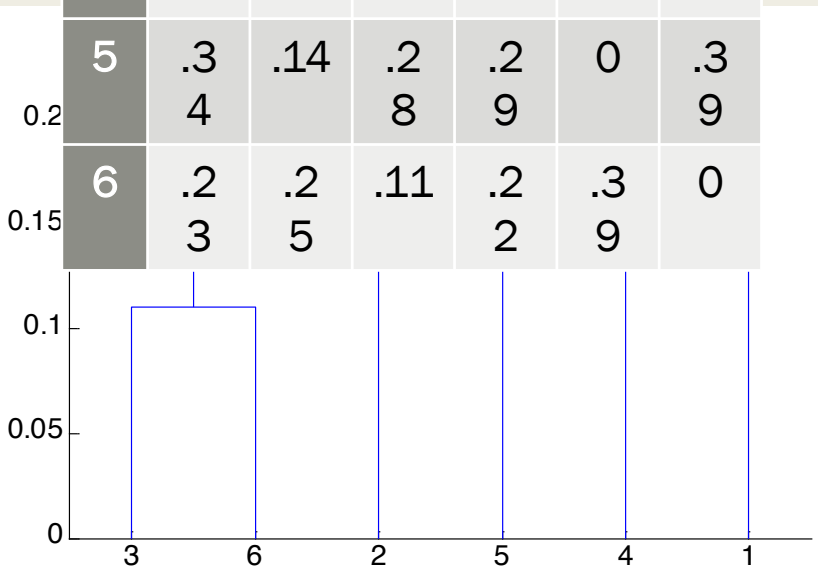
- Another way to view the processing of the hierarchical algorithm is that we create links between their elements in order of increasing distance
  - *The MIN – Single Link, will merge two clusters when a single pair of elements is linked*
  - *The MAX – Complete Linkage will merge two clusters when all pairs of elements have been linked.*

Hierarchical Clustering: MIN



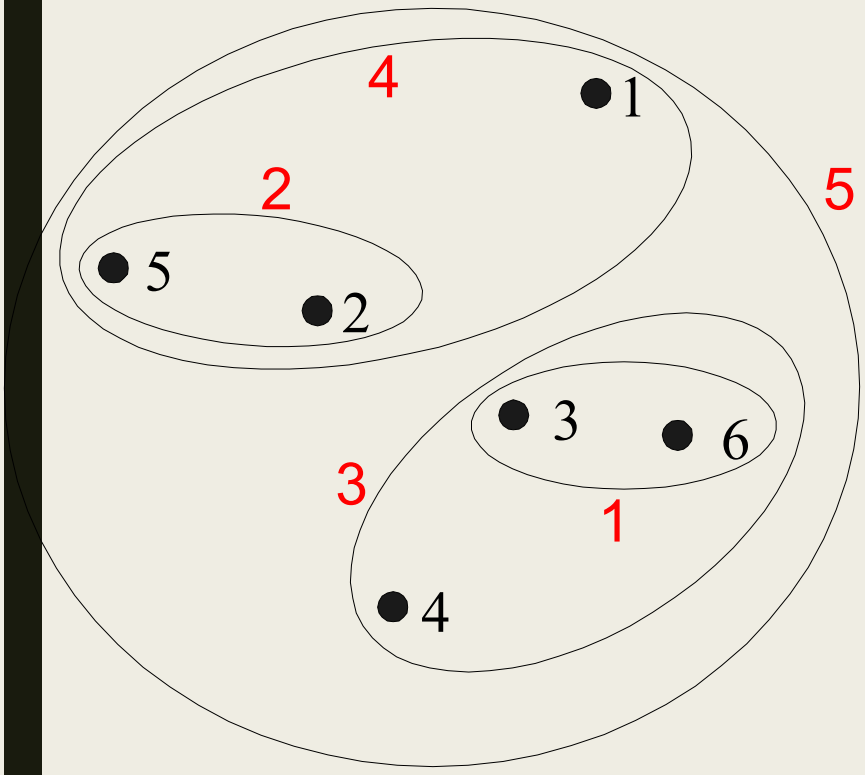
Nested Clusters

	1	2	3	4	5	6
1	0	.24	.22	.37	.34	.23
2	.24	0	.15	.20	.14	.25
3	.22	.15	0	.15	.28	.11
4	.37	.20	.15	0	.29	.22
5	.34	.14	.28	.29	0	.39
6	.23	.25	.11	.22	.39	0



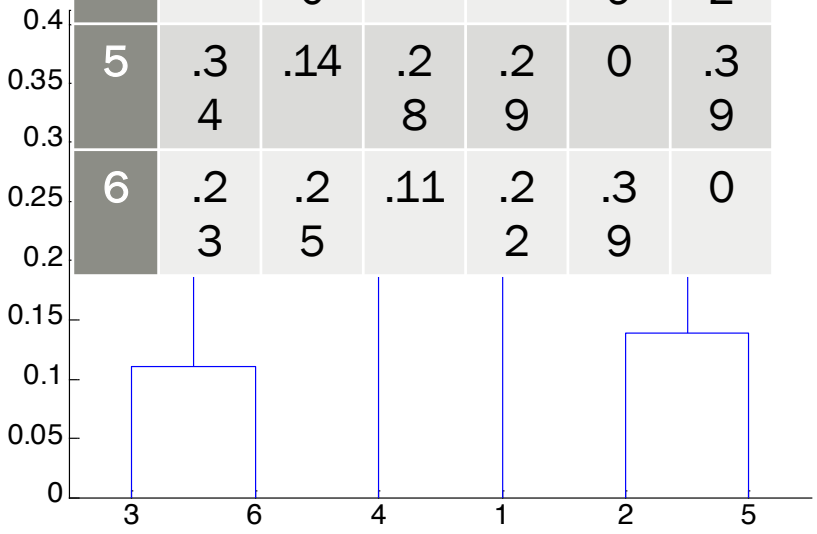
Dendrogram

Hierarchical Clustering: MAX



Nested Clusters

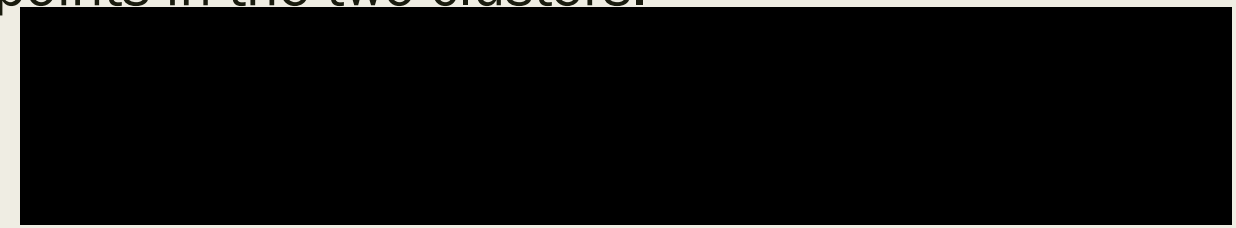
	1	2	3	4	5	6
1	0	.24	.22	.37	.34	.23
2	.24	0	.15	.20	.14	.25
3	.22	.15	0	.15	.28	.11
4	.37	.20	.15	0	.29	.22
5	.34	.14	.28	.29	0	.39
6	.23	.25	.11	.22	.39	0



Dendrogram

Cluster Similarity: Group Average

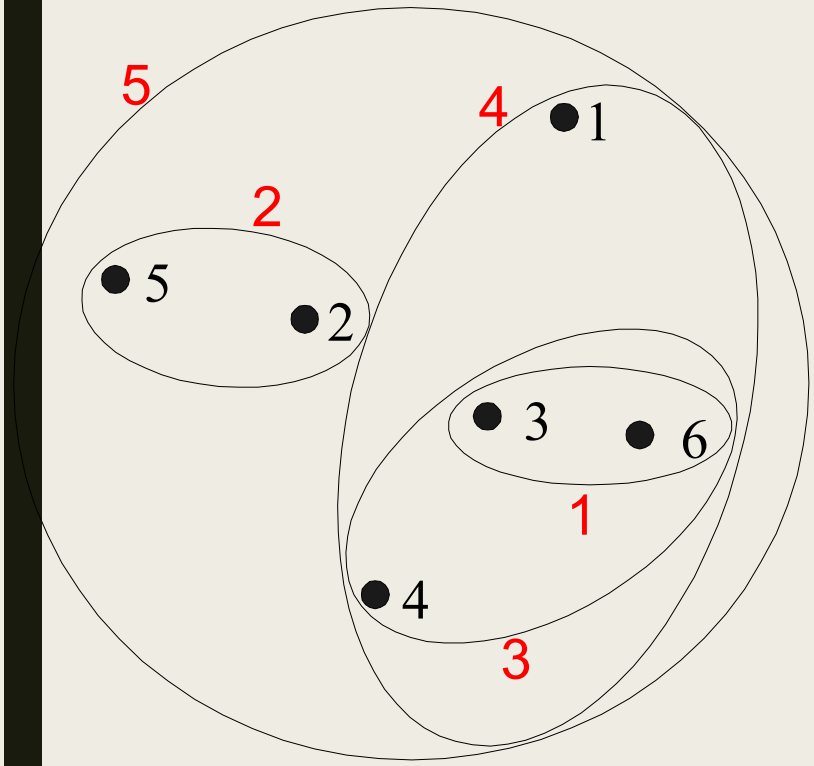
- Proximity of two clusters is the average of pairwise proximity between points in the two clusters.



- Need to use average connectivity for scalability since total proximity favors large clusters

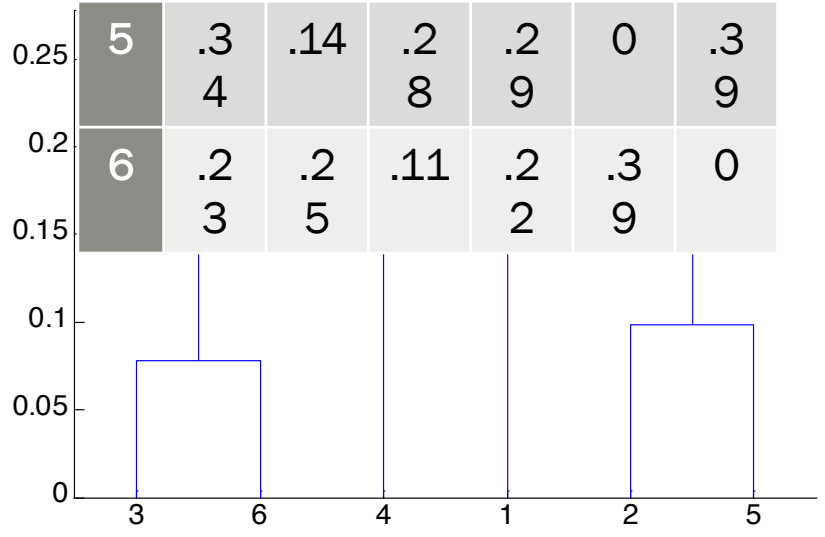
	1	2	3	4	5	6
1	0	.24	.22	.37	.34	.23
2	.24	0	.15	.20	.14	.25
3	.22	.15	0	.15	.28	.11
4	.37	.20	.15	0	.29	.22

Hierarchical Clustering: Group Average



Nested Clusters

	1	2	3	4	5	6
1	0	.24	.22	.37	.34	.23
2	.24	0	.15	.20	.14	.25
3	.22	.15	0	.15	.28	.11
4	.37	.20	.15	0	.29	.22
5	.34	.14	.28	.29	0	.39
6	.23	.25	.11	.22	.39	0



Dendrogram

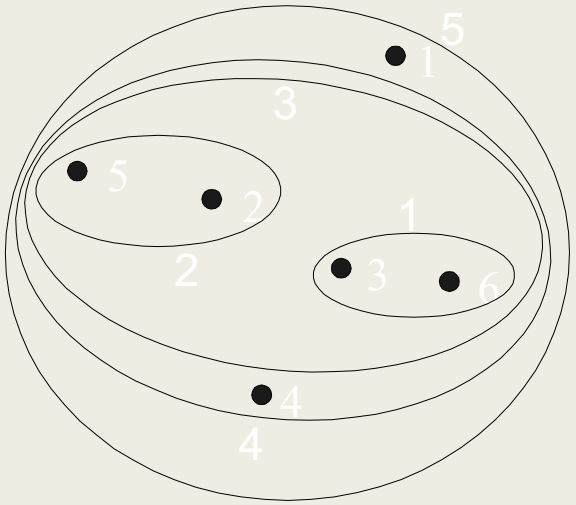
Hierarchical Clustering: Group Average

- Compromise between Single and Complete Link
- Strengths
  - *Less susceptible to noise and outliers*
- Limitations
  - *Biased towards globular clusters*

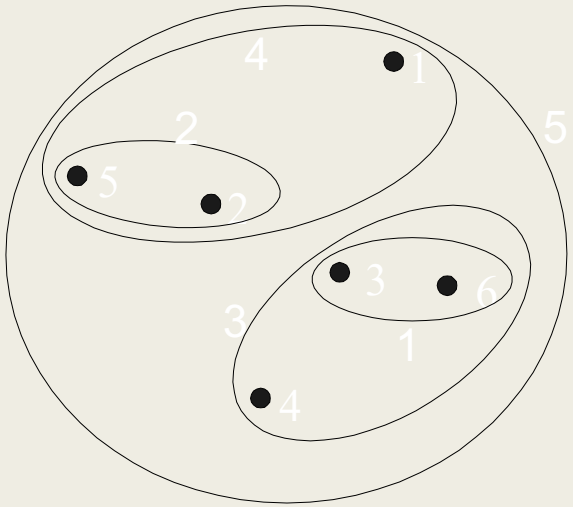
Cluster Similarity: Ward's Method

- Similarity of two clusters is based on the increase in squared error (SSE) when two clusters are merged
  - *Similar to group average if distance between points is distance squared*
- Less susceptible to noise and outliers
- Biased towards globular clusters
- Hierarchical analogue of K-means
  - *Can be used to initialize K-means*

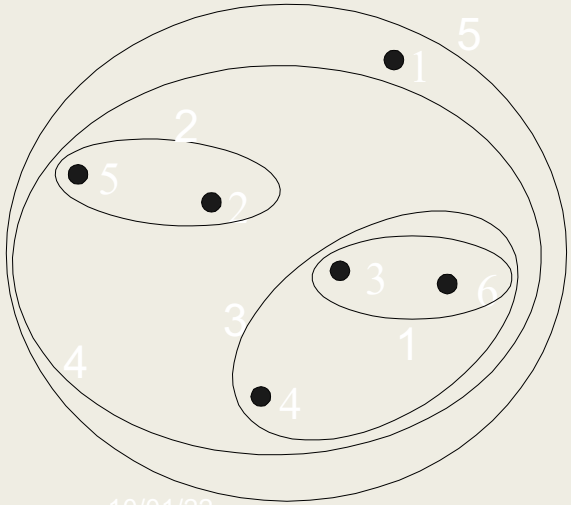
Hierarchical Clustering: Comparison



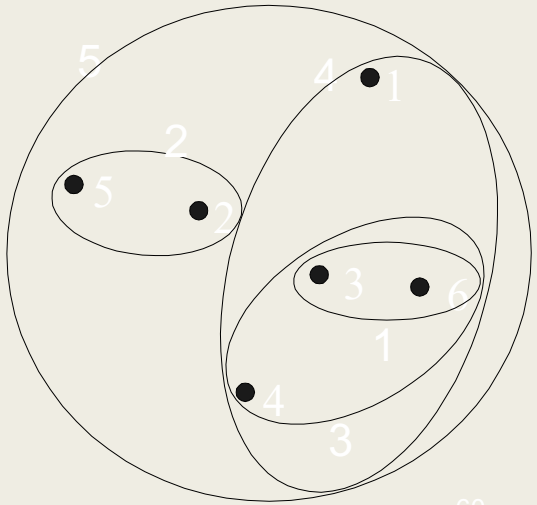
MIN



MAX



Group Average

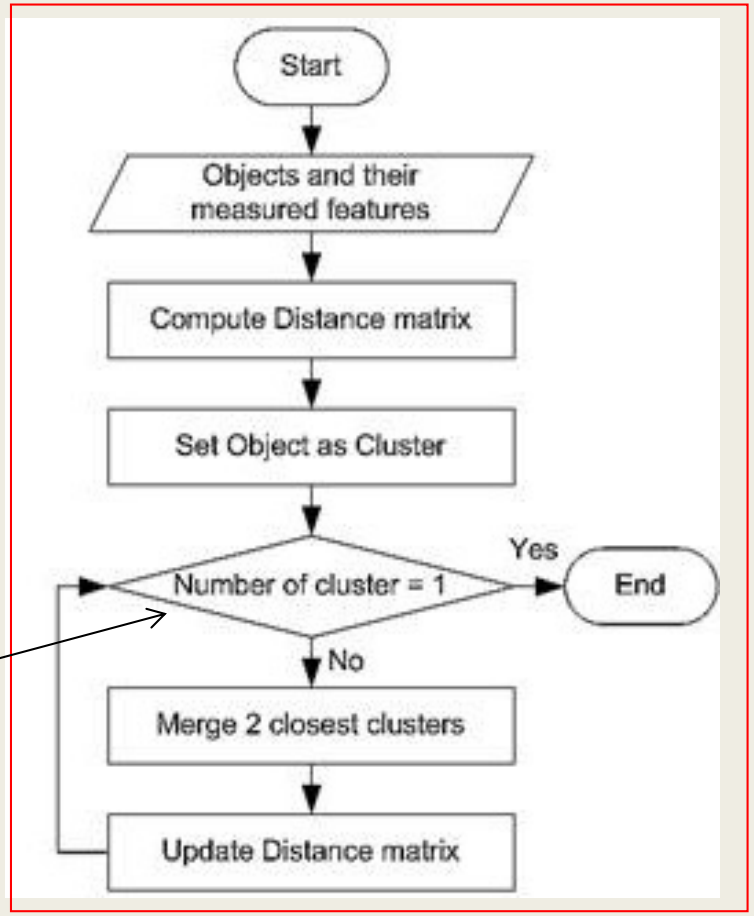


Ward's Method

Agglomerative Algorithm

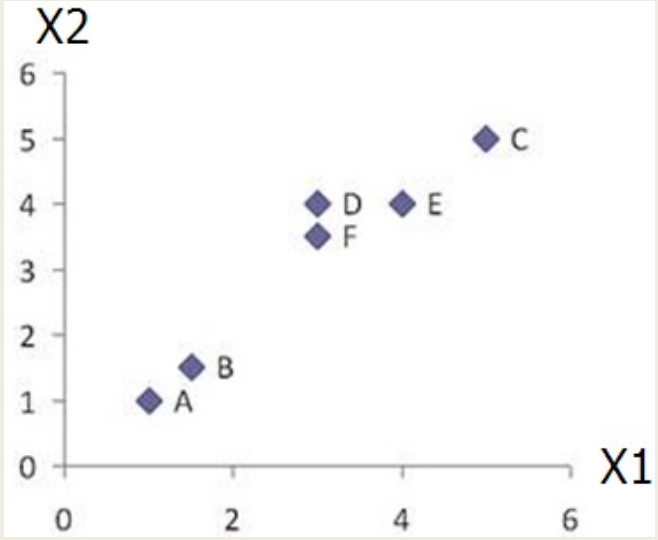
- The *Agglomerative* algorithm is carried out in three steps:

1) Convert all object features into a distance matrix
2) Set each object as a cluster (thus if we have *N* objects, we will have *N* clusters at the beginning)
3) Repeat until number of cluster is one (or known # of clusters)
   - Merge two closest clusters
   - Update "distance matrix"



Example

- Problem: clustering analysis with agglomerative algorithm



	X1	X2
A	1	1
B	1.5	1.5
C	5	5
D	3	4
E	4	4
F	3	3.5

data matrix

$$d_{AB} = \left((1-1.5)^2 + (1-1.5)^2\right)^{\frac{1}{2}} = \sqrt{\frac{1}{2}} = 0.7071$$

$$d_{DF} = \left((3-3)^2 + (4-3.5)^2\right)^{\frac{1}{2}} = 0.5$$

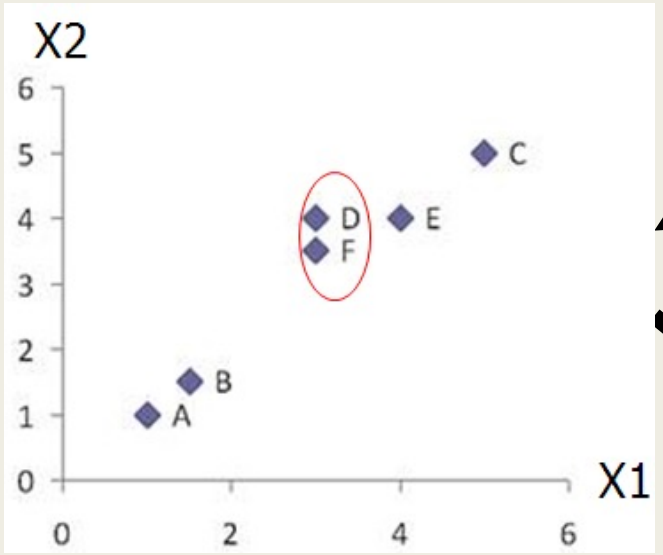
Euclidean distance

Dist	A	B	C	D	E	F
A	0.00	0.71	5.66	3.61	4.24	3.20
B	0.71	0.00	4.95	2.92	3.54	2.50
C	5.66	4.95	0.00	2.24	1.41	2.50
D	3.61	2.92	2.24	0.00	1.00	0.50
E	4.24	3.54	1.41	1.00	0.00	1.12
F	3.20	2.50	2.50	0.50	1.12	0.00

distance matrix

Example

- Merge two closest clusters (iteration 1)



Dist	A	B	C	D	E	F
A	0.00	0.71	5.66	3.61	4.24	3.20
B	0.71	0.00	4.95	2.92	3.54	2.50
C	5.66	4.95	0.00	2.24	1.41	2.50
D	3.61	2.92	2.24	0.00	1.00	0.50
E	4.24	3.54	1.41	1.00	0.00	1.12
F	3.20	2.50	2.50	0.50	1.12	0.00

Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	?	4.24
B	0.71	0.00	4.95	?	3.54
C	5.66	4.95	0.00	?	1.41
D, F	?	?	?	0.00	?
E	4.24	3.54	1.41	?	0.00

Example

- Update distance matrix (iteration 1)

Dist	A	B	C	D	E	F
A	0.00	0.71	5.66	3.61	4.24	3.20
B	0.71	0.00	4.95	2.92	3.54	2.50
C	5.66	4.95	0.00	2.24	1.41	2.50
D	3.61	2.92	2.24	0.00	1.00	0.50
E	4.24	3.54	1.41	1.00	0.00	1.12
F	3.20	2.50	2.50	0.50	1.12	0.00

$$d_{(D,F)\to A} = \min\left(d_{DA}, d_{FA}\right) = \min\left(3.61, 3.20\right) = 3.20$$

$$d_{(D,F)\to B} = \min\left(d_{DB}, d_{FB}\right) = \min\left(2.92, 2.50\right) = 2.50$$

$$d_{(D,F)\to C} = \min\left(d_{DC}, d_{FC}\right) = \min\left(2.24, 2.50\right) = 2.24$$

$$d_{E\to(D,F)} = \min\left(d_{ED}, d_{EF}\right) = \min\left(1.00, 1.12\right) = 1.00$$

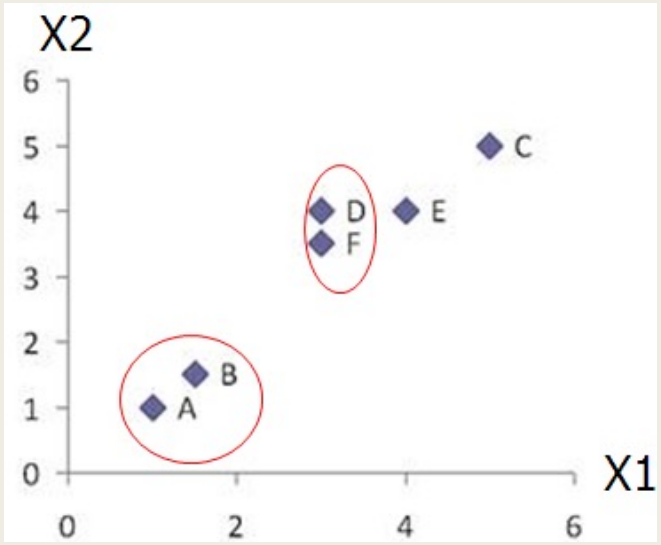
Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	?	4.24
B	0.71	0.00	4.95	?	3.54
C	5.66	4.95	0.00	?	1.41
D, F	?	?	?	0.00	?
E	4.24	3.54	1.41	?	0.00

Min Distance (Single Linkage)

Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	3.20	4.24
B	0.71	0.00	4.95	2.50	3.54
C	5.66	4.95	0.00	2.24	1.41
D, F	**3.20**	**2.50**	**2.24**	0.00	1.00
E	4.24	3.54	1.41	**1.00**	0.00

Example

- Merge two closest clusters (iteration 2)



Min Distance (Single Linkage)

Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	3.20	4.24
B	0.71	0.00	4.95	2.50	3.54
C	5.66	4.95	0.00	2.24	1.41
D, F	**3.20**	**2.50**	**2.24**	0.00	1.00
E	4.24	3.54	1.41	**1.00**	0.00

Dist	A,B	C	(D, F)	E
A,B	0	?	?	?
C	?	0	2.24	1.41
(D, F)	?	2.24	0	1.00
E	?	1.41	1.00	0

Example

- Update distance matrix (iteration 2)

Min Distance (Single Linkage)

Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	3.20	4.24
B	0.71	0.00	4.95	2.50	3.54
C	5.66	4.95	0.00	2.24	1.41
D, F	**3.20**	**2.50**	**2.24**	0.00	1.00
E	4.24	3.54	1.41	**1.00**	0.00

$$d_{C\to(A,B)} = \min\left(d_{CA}, d_{CB}\right) = \min\left(5.66, 4.95\right) = 4.95$$

$$d_{(D,F)\to(A,B)} = \min\left(d_{DA}, d_{DB}, d_{FA}, d_{FB}\right) = \min\left(3.61, 2.92, 3.20, 2.50\right) = 2.50$$

$$d_{E\to(A,B)} = \min\left(d_{EA}, d_{EB}\right) = \min\left(4.24, 3.54\right) = 3.54$$

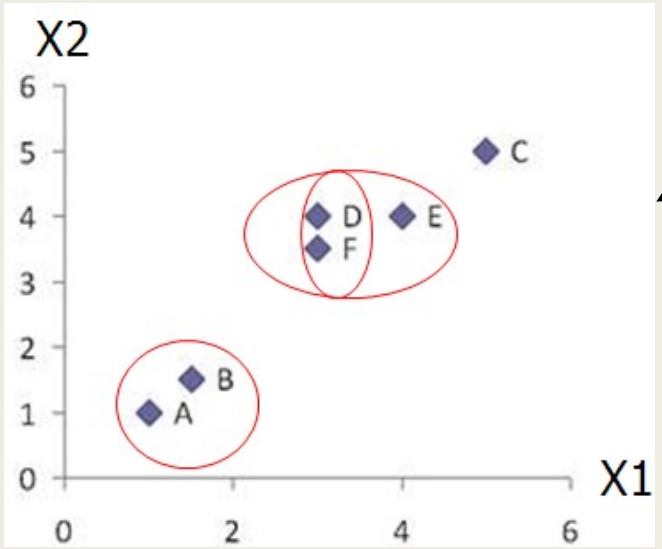
Dist	A,B	C	(D, F)	E
A,B	0	?	?	?
C	?	0	2.24	1.41
(D, F)	?	2.24	0	1.00
E	?	1.41	1.00	0

Min Distance (Single Linkage)

Dist	A,B	C	(D, F)	E
A,B	0	4.95	2.50	3.54
C	**4.95**	0	2.24	1.41
(D, F)	**2.50**	2.24	0	1.00
E	**3.54**	1.41	1.00	0

Example

Merge two closest clusters/update distance matrix (iteration 3)



Min Distance (Single Linkage)

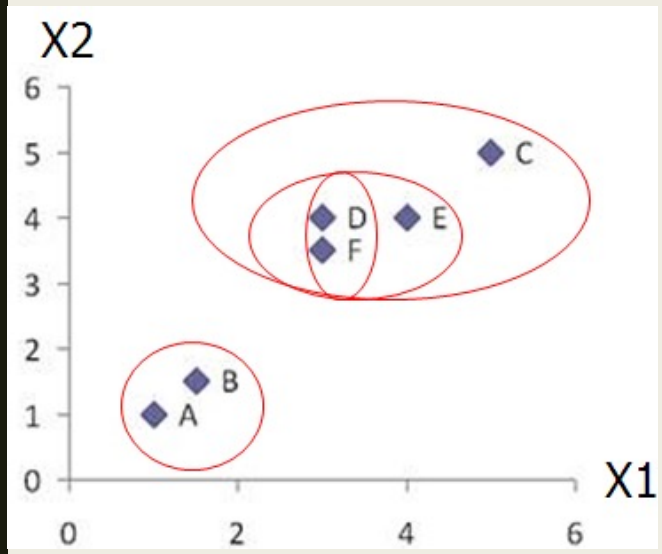
Dist	A,B	C	(D, F)	E
A,B	0	4.95	2.50	3.54
C	**4.95**	0	2.24	1.41
(D, F)	**2.50**	2.24	0	1.00
E	**3.54**	1.41	1.00	0

Min Distance (Single Linkage)

Dist	(A,B)	C	(D, F), E
(A,B)	0.00	4.95	2.50
C	4.95	0.00	1.41
(D, F), E	**2.50**	**1.41**	0.00

Example

- Merge two closest clusters/update distance matrix (iteration 4)



Min Distance (Single Linkage)

Dist	(A,B)	C	(D, F), E
(A,B)	0.00	4.95	2.50
C	4.95	0.00	1.41
(D, F), E	**2.50**	**1.41**	0.00

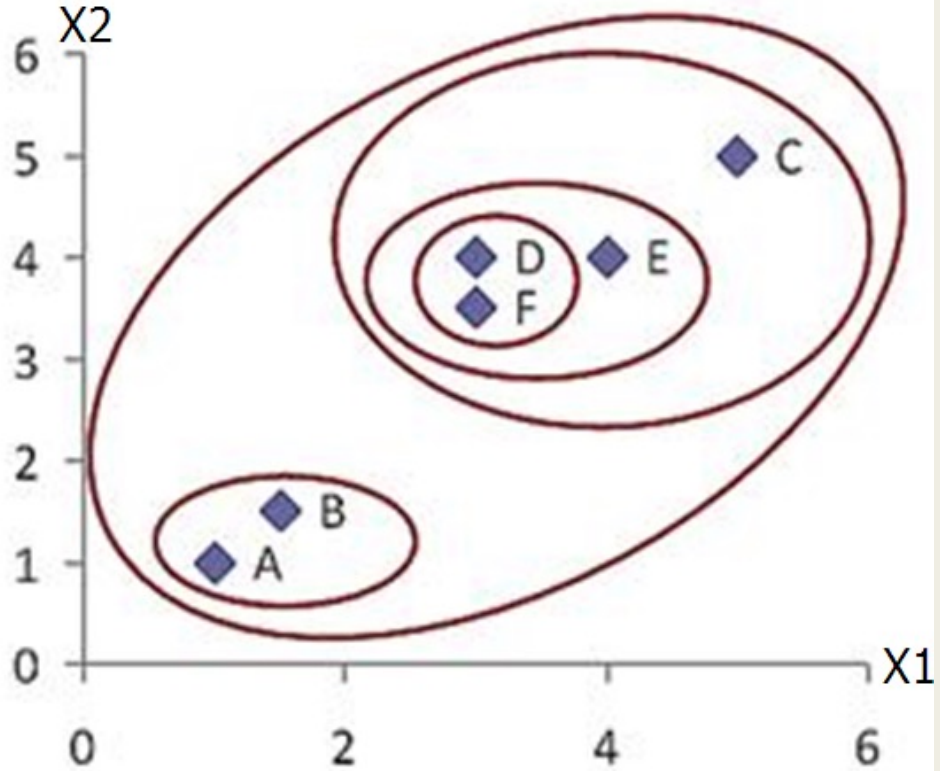
Min Distance (Single Linkage)

Dist	(A,B)	((D, F), E),C
(A,B)	0.00	2.50
((D, F), E),C	**2.50**	0.00

Example

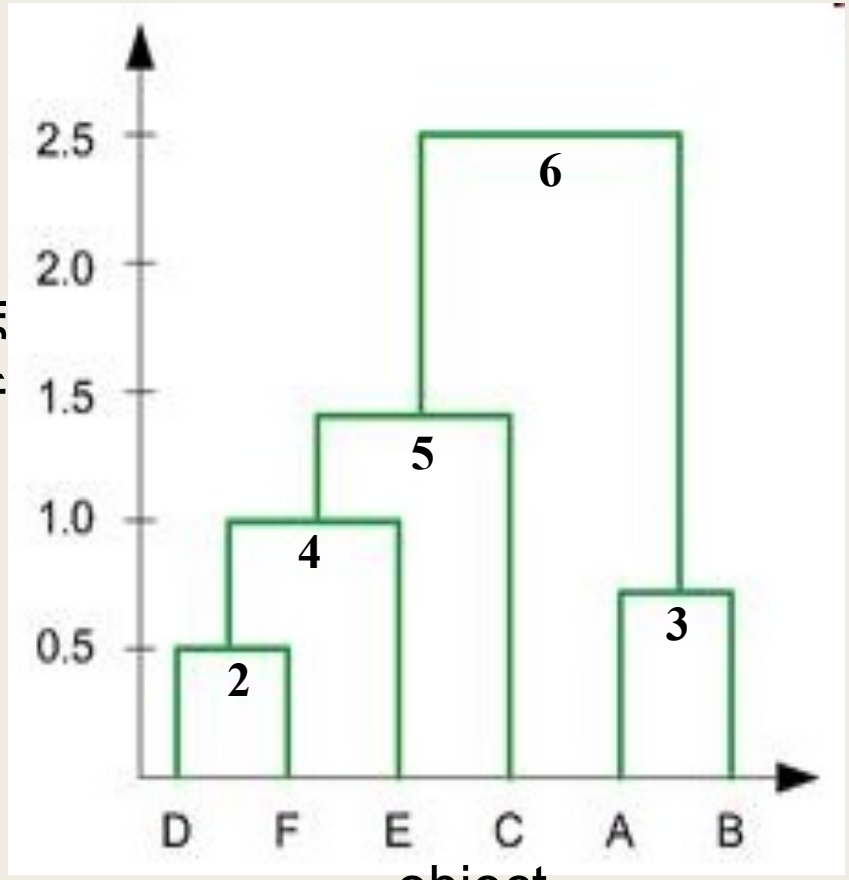
- Final result (meeting termination condition)

	X1	X2
A	1	1
B	1.5	1.5
C	5	5
D	3	4
E	4	4
F	3	3.5



Example

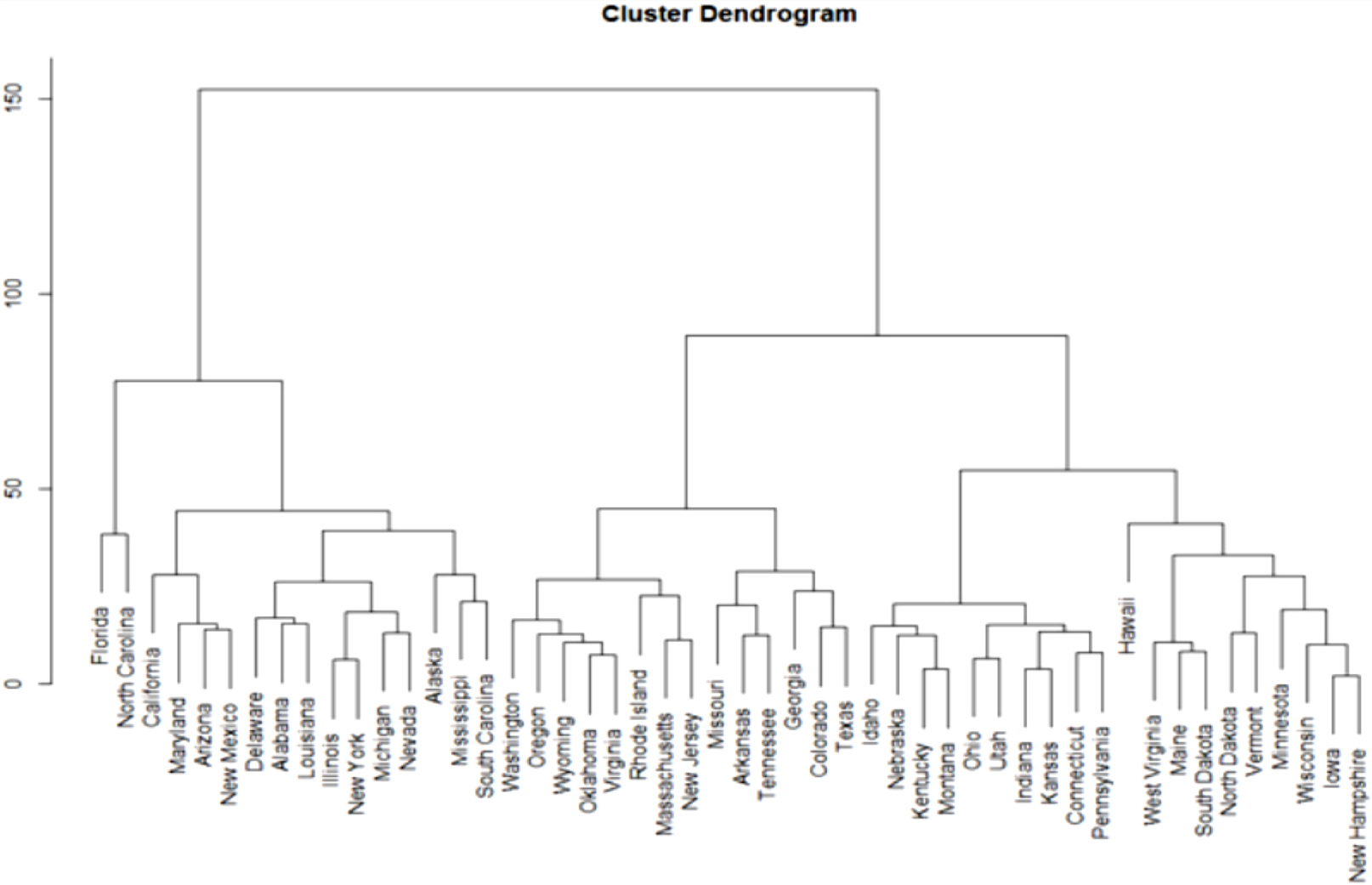
- Dendrogram tree representation



1. In the beginning we have 6 clusters: A, B, C, D, E and F
2. We merge clusters D and F into cluster (D, F) at distance 0.50
3. We merge cluster A and cluster B into (A, B) at distance 0.71
4. We merge clusters E and (D, F) into ((D, F), E) at distance 1.00
5. We merge clusters ((D, F), E) and C into (((D, F), E), C) at distance 1.41
6. We merge clusters (((D, F), E), C) and (A, B) into ((((D, F), E), C), (A, B)) at distance 2.50
7. The last cluster contain all the objects, thus conclude the computation

Example

- Dendrogram tree representation: “clustering” USA states



Strengths of Hierarchical Clustering

- Do not have to assume any particular number of clusters
  - *Any desired number of clusters can be obtained by 'cutting' the dendrogram at the proper level*

- They may correspond to meaningful taxonomies
  - *Example in biological sciences (e.g., animal kingdom, phylogeny reconstruction, …)*

Hierarchical Clustering: Problems and Limitations

- Computational complexity in time and space
- Once a decision is made to combine two clusters, it cannot be undone
- No objective function is directly minimized
- Different schemes have problems with one or more of the following:
  - *Sensitivity to noise and outliers*
  - *Difficulty handling different sized clusters and convex shapes*
  - *Breaking large clusters*

Exercise

Given a data set of five objects characterised by a single continuous feature:

	a	b	C	d	e
Feature	1	2	4	5	6

Apply the agglomerative algorithm with single-link, complete-link and averaging cluster distance measures to produce three dendrogram trees, respectively.

	a	b	c	d	e
a	0	1	3	4	5
b	1	0	2	3	4
c	3	2	0	1	2
d	4	3	1	0	1
e	5	4	2	1	0

Summary

- Hierarchical algorithm is a sequential clustering algorithm
  - *Use distance matrix to construct a tree of clusters (dendrogram)*
  - *Hierarchical representation without the need of knowing # of clusters (can set termination condition with known # of clusters)*
- Major weakness of agglomerative clustering methods
  - *Can never undo what was done previously*
  - *Sensitive to cluster distance measures and noise/outliers*
  - *Less efficient: O (n^2 logn), where n is the number of total objects*
- There are several variants to overcome its weaknesses
  - *BIRCH: scalable to a large data set*
  - *ROCK: clustering categorical data*
  - *CHAMELEON: hierarchical clustering using dynamic modelling*