

Chapter VIII: Clustering

Information Retrieval & Data Mining
Universität des Saarlandes, Saarbrücken
Winter Semester 2011/12

Chapter VIII: Clustering*

1. Basic idea

1.1. Distances between objects

2. Representative-based clustering

2.1. *k*-means

2.2. EM-clustering

3. Hierarchical clustering

3.1. Basic idea

3.2. Cluster distances

4. Co-clustering

5. Discussion and clustering applications

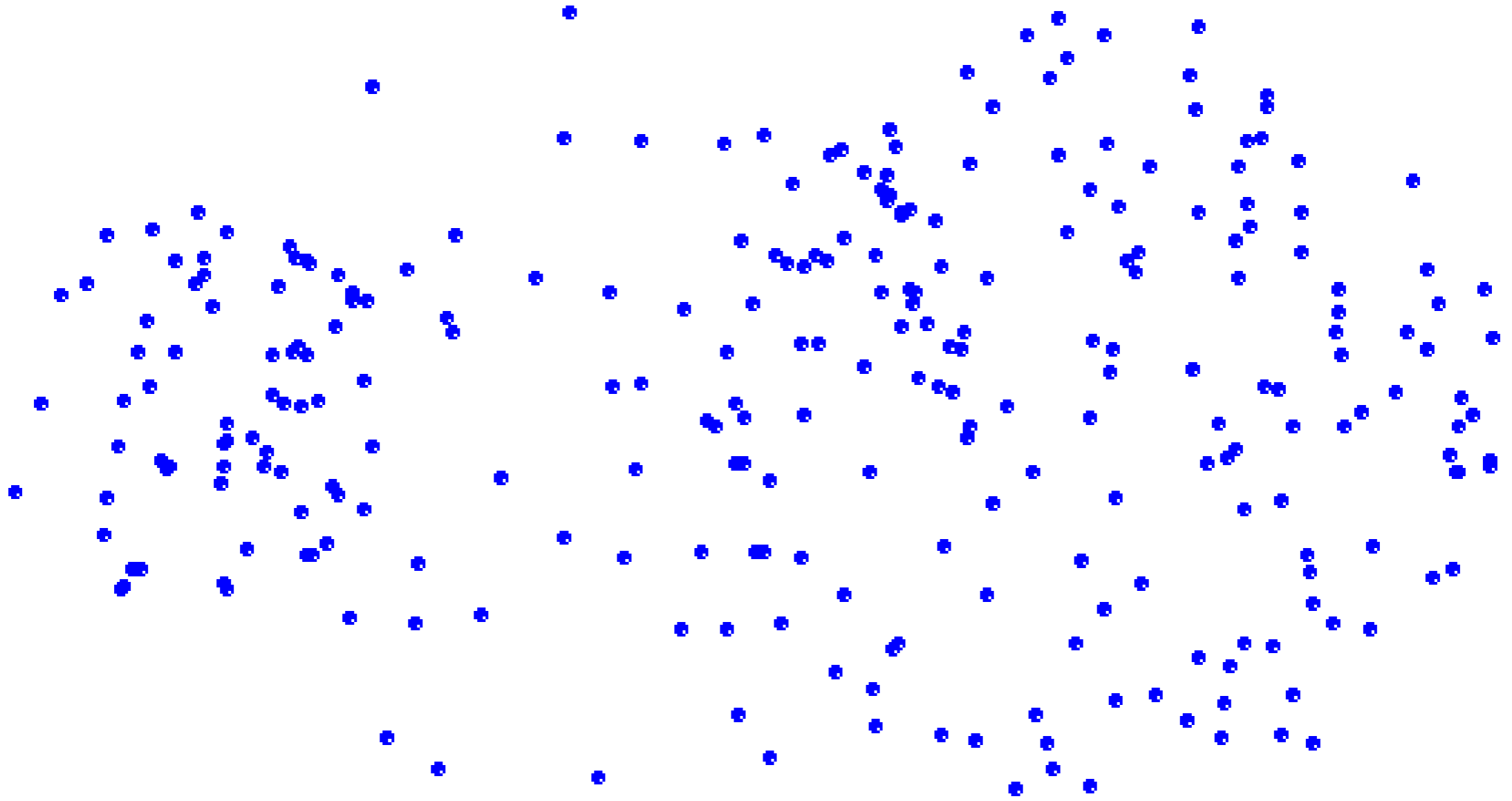
*Zaki & Meira, Chapters 16 and 17; Tan, Steinbach & Kumar, Chapter 8

1. Basic idea

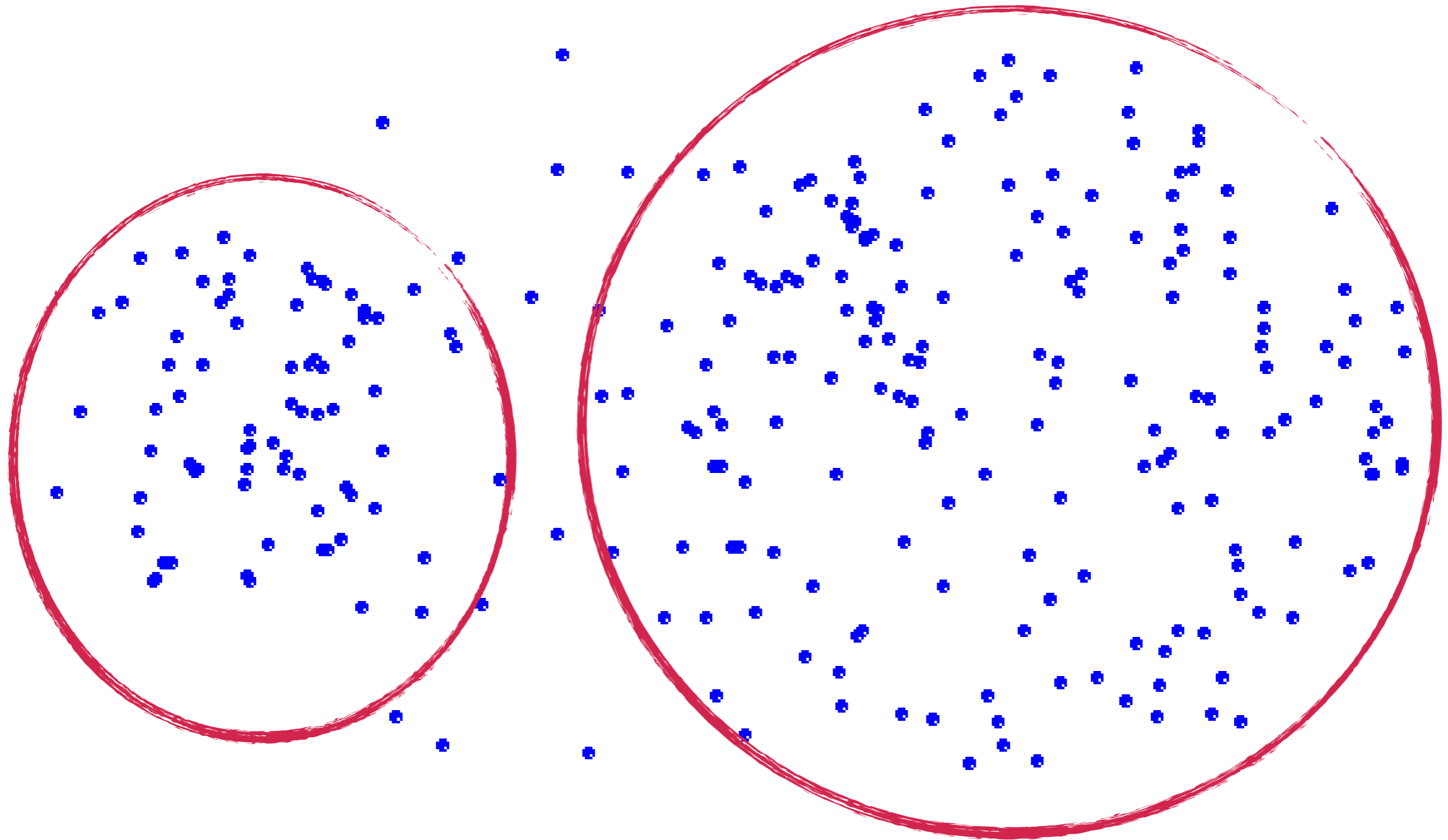
1. Example

2. Distances between objects

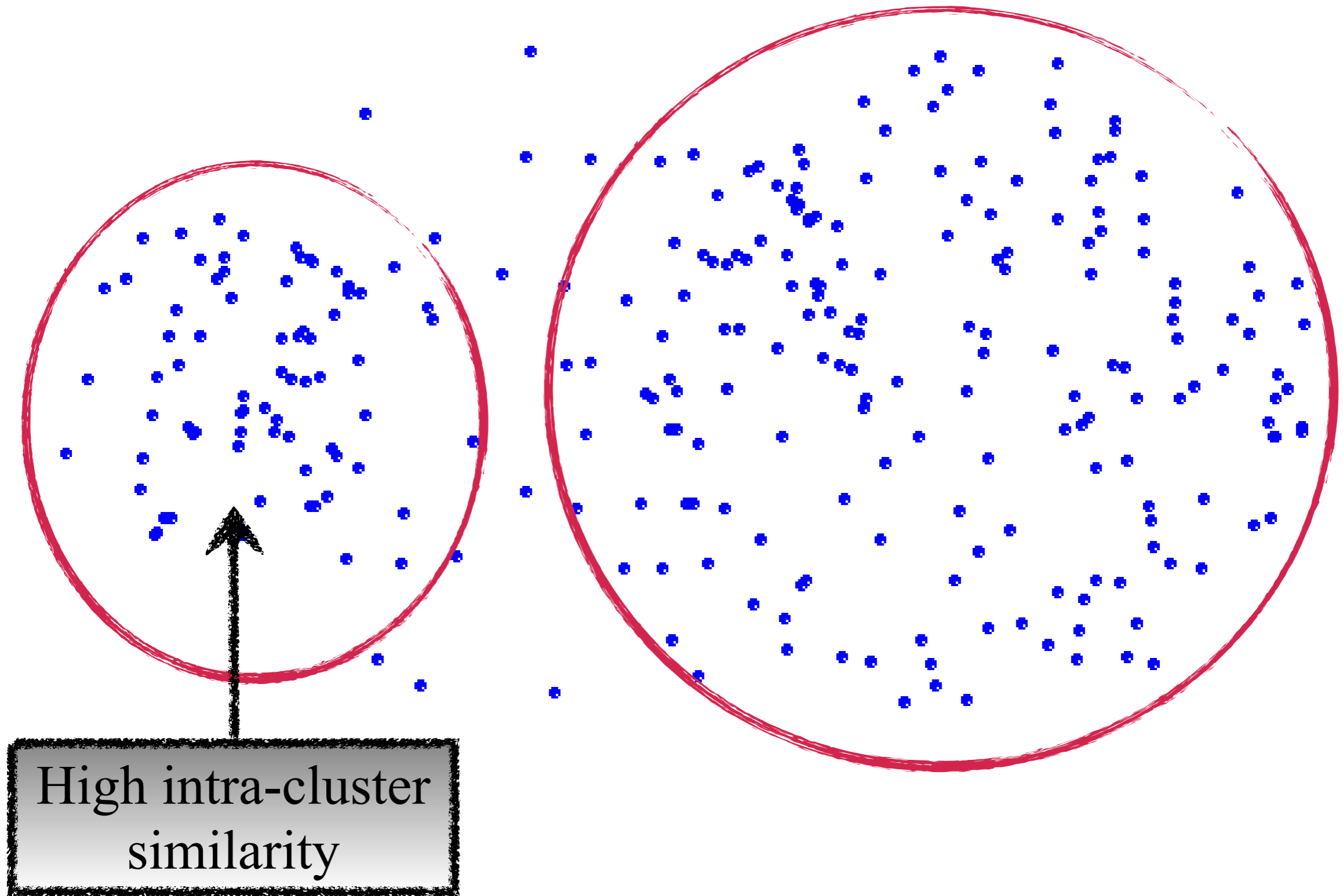
Example



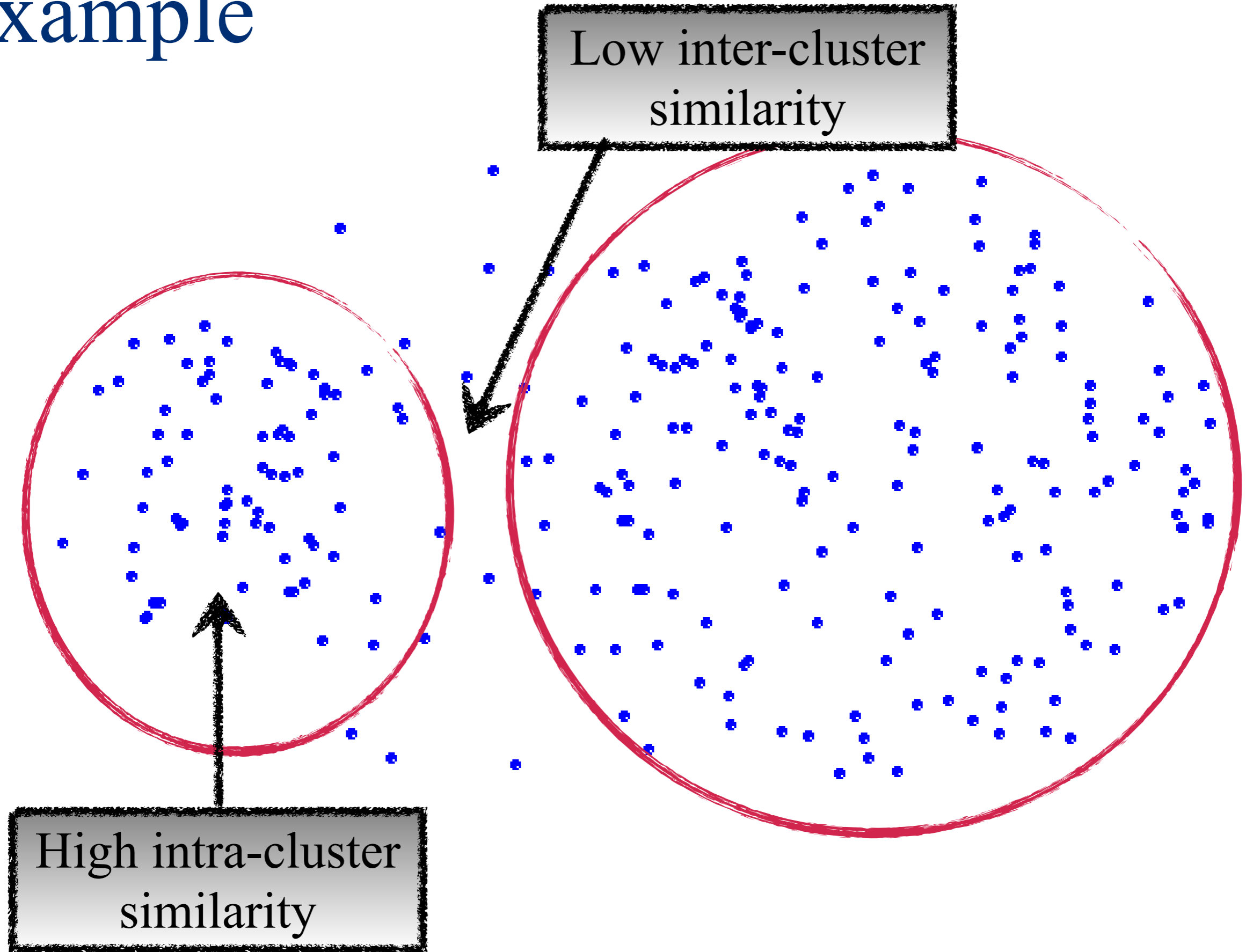
Example



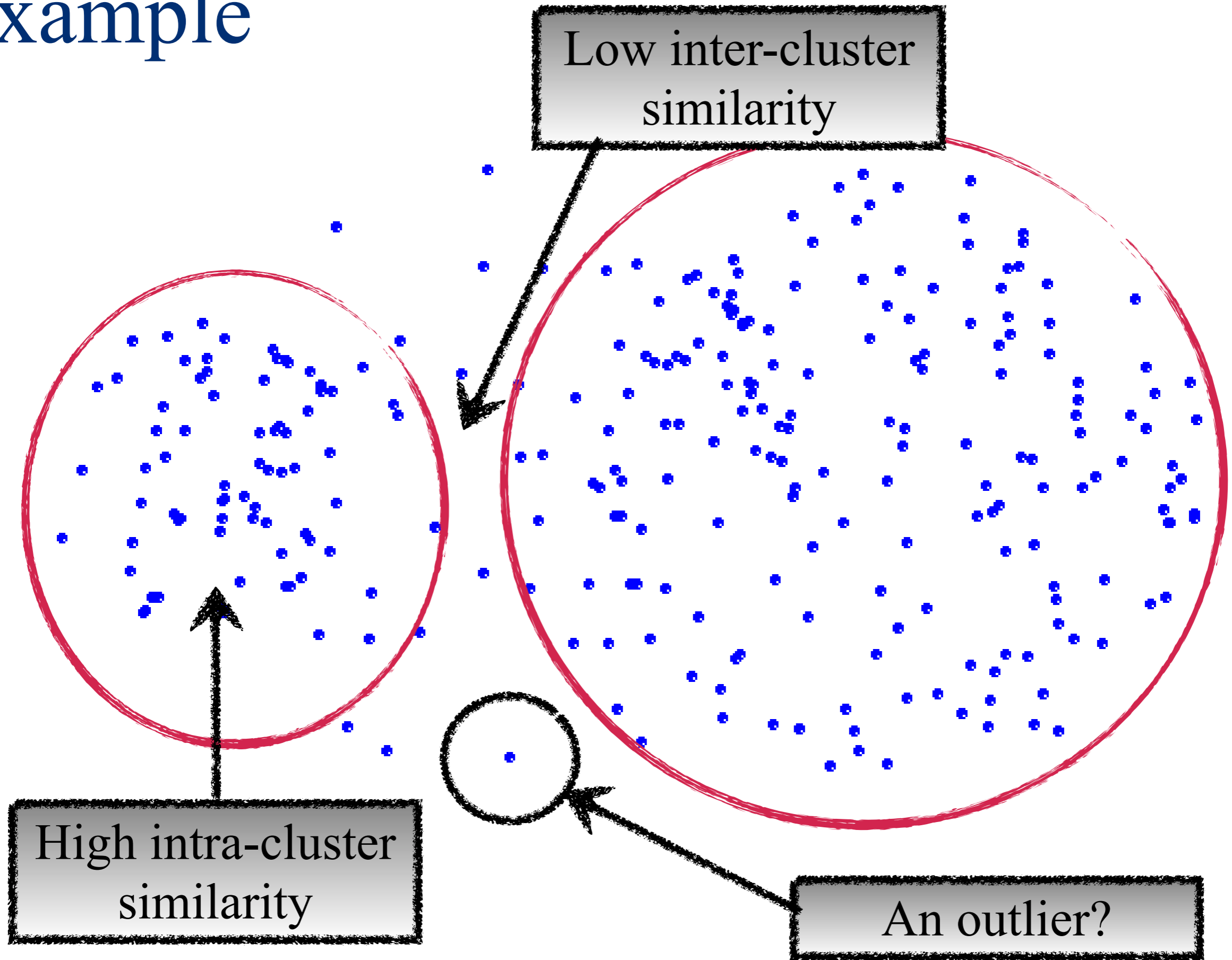
Example



Example



Example



The clustering task

- Given a set U of objects and a distance $d:U^2 \rightarrow R^+$ between them, group objects of U into clusters such that distance between points in the same cluster is low and distance between the points in different clusters is large
 - Small and large are not well defined
 - Clustering can be
 - *exclusive* (each point belongs to exactly one cluster)
 - *probabilistic* (each point-cluster pair is associated with a probability of the point belonging to that cluster)
 - *fuzzy* (each point can belong to multiple clusters)
 - Number of clusters can be pre-defined or not

On distances

- A function $d:U^2 \rightarrow R^+$ is a **metric** if:
 - $d(u, v) = 0$ if and only if $u = v$
 - $d(u, v) = d(v, u)$ for all $u, v \in U$
 - $d(u, v) \leq d(u, w) + d(w, v)$ for all u, v , and $w \in U$
- A metric is a *distance*; if $d:U^2 \rightarrow [0, a]$ for some positive a , then $a - d(u, v)$ is *similarity*
- Common metrics:
 - $L_p: \left(\sum_{i=1}^d |u_i - v_i|^p \right)^{\frac{1}{p}}$ for d -dimensional space
 - L_1 = Hamming = city-block; L_2 = Euclidean
 - Correlation distance: $1 - \varphi$
 - Jaccard distance: $1 - |A \cap B| / |A \cup B|$

On distances

- A function $d:U^2 \rightarrow R^+$ is a **metric** if:
 - $d(u, v) = 0$ if and only if $u = v$ **Self-similarity**
 - $d(u, v) = d(v, u)$ for all $u, v \in U$
 - $d(u, v) \leq d(u, w) + d(w, v)$ for all u, v , and $w \in U$
- A metric is a *distance*; if $d:U^2 \rightarrow [0, a]$ for some positive a , then $a - d(u, v)$ is *similarity*
- Common metrics:
 - $L_p: \left(\sum_{i=1}^d |u_i - v_i|^p \right)^{\frac{1}{p}}$ for d -dimensional space
 - L_1 = Hamming = city-block; L_2 = Euclidean
 - Correlation distance: $1 - \varphi$
 - Jaccard distance: $1 - |A \cap B| / |A \cup B|$

On distances

- A function $d:U^2 \rightarrow R^+$ is a **metric** if:
 - $d(u, v) = 0$ if and only if $u = v$ **Self-similarity**
 - $d(u, v) = d(v, u)$ for all $u, v \in U$ **Symmetry**
 - $d(u, v) \leq d(u, w) + d(w, v)$ for all u, v , and $w \in U$
- A metric is a *distance*; if $d:U^2 \rightarrow [0, a]$ for some positive a , then $a - d(u, v)$ is *similarity*
- Common metrics:
 - $L_p: \left(\sum_{i=1}^d |u_i - v_i|^p \right)^{\frac{1}{p}}$ for d -dimensional space
 - L_1 = Hamming = city-block; L_2 = Euclidean
 - Correlation distance: $1 - \varphi$
 - Jaccard distance: $1 - |A \cap B| / |A \cup B|$

On distances

- A function $d:U^2 \rightarrow R^+$ is a **metric** if:
 - $d(u, v) = 0$ if and only if $u = v$ **Self-similarity**
 - $d(u, v) = d(v, u)$ for all $u, v \in U$ **Symmetry**
 - $d(u, v) \leq d(u, w) + d(w, v)$ for all u, v , and $w \in U$ **Triangle inequality**
- A metric is a *distance*; if $d:U^2 \rightarrow [0, a]$ for some positive a , then $a - d(u, v)$ is *similarity*
- Common metrics:
 - $L_p: \left(\sum_{i=1}^d |u_i - v_i|^p \right)^{\frac{1}{p}}$ for d -dimensional space
 - L_1 = Hamming = city-block; L_2 = Euclidean
 - Correlation distance: $1 - \varphi$
 - Jaccard distance: $1 - |A \cap B| / |A \cup B|$

More on distances

- For all-numerical data, **sum of squared errors** is the most common one
 - SSE: $\sum_{i=1}^d |u_i - v_i|^2$
- For all-binary data, either Hamming or Jaccard is used
- For categorical data either
 - first convert the data to binary by adding one binary variable per category label and then use Hamming; or
 - count the agreements and disagreements of category labels with Jaccard
- For mixed data, some combination must be used

Implicit distance and distance matrix

$$\begin{pmatrix} 0 & d_{1,2} & d_{1,3} & \cdots & d_{1,n} \\ d_{1,2} & 0 & d_{2,3} & \cdots & d_{2,n} \\ d_{1,3} & d_{2,3} & 0 & \cdots & d_{3,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ d_{1,n} & d_{2,n} & d_{3,n} & \cdots & 0 \end{pmatrix}$$

- A *distance* (or *dissimilarity*) *matrix* is
 - n -by- n for n objects
 - non-negative ($d_{i,j} \geq 0$)
 - symmetric ($d_{i,j} = d_{j,i}$)
 - zero on diagonal ($d_{i,i} = 0$)

2. Representative-based clustering

1. Partitions and prototypes

2. The k -means algorithm

2.1. Basic algorithm

2.2. Analysis

2.3. The k -means++ algorithm

3. The EM clustering algorithm

3.1. 1-D Gaussian

3.2. General Gaussian

3.3. The k -means as EM

4. How to select k

Partitions and prototypes

- *Exclusive* representative-based clustering:
 - The set of objects U is partitioned into k **clusters** C_1, C_2, \dots, C_k
 - $\cup_i C_i = U$ and $C_i \cap C_j = \emptyset$ for $i \neq j$
 - Each cluster is represented by a **prototype** (also called **centroid** or **mean**) μ_i
 - Prototype does not have to be (and usually is not) one of the objects
 - Clustering quality is based on sum of squared errors between objects in cluster and cluster prototype

$$\sum_{i=1}^k \sum_{x_j \in C_i} \|x_j - \mu_i\|_2^2 = \sum_{i=1}^k \sum_{x_j \in C_i} \sum_{l=1}^d (x_{jl} - \mu_{il})^2$$

Partitions and prototypes

- *Exclusive* representative-based clustering:
 - The set of objects U is partitioned into k **clusters** C_1, C_2, \dots, C_k
 - $\cup_i C_i = U$ and $C_i \cap C_j = \emptyset$ for $i \neq j$
 - Each cluster is represented by a **prototype** (also called **centroid** or **mean**) μ_i
 - Prototype does not have to be (and usually is not) one of the objects
 - Clustering quality is based on **Over all clusters** sum of squared errors between objects in cluster and cluster prototype

$$\sum_{i=1}^k \sum_{x_j \in C_i} \|x_j - \mu_i\|_2^2 = \sum_{i=1}^k \sum_{x_j \in C_i} \sum_{l=1}^d (x_{jl} - \mu_{il})^2$$

Partitions and prototypes

- *Exclusive* representative-based clustering:
 - The set of objects U is partitioned into k **clusters** C_1, C_2, \dots, C_k
 - $\cup_i C_i = U$ and $C_i \cap C_j = \emptyset$ for $i \neq j$
 - Each cluster is represented by a **prototype** (also called **centroid** or **mean**) μ_i
 - Prototype does not have to be (and usually is not) one of the objects
 - Clustering quality is based on sum of squared errors between objects in cluster and cluster prototype

Over all objects in this cluster

$$\sum_{i=1}^k \sum_{x_j \in C_i} \|x_j - \mu_i\|_2^2 = \sum_{i=1}^k \sum_{x_j \in C_i} \sum_{l=1}^d (x_{jl} - \mu_{il})^2$$

Partitions and prototypes

- *Exclusive* representative-based clustering:
 - The set of objects U is partitioned into k **clusters** C_1, C_2, \dots, C_k
 - $\cup_i C_i = U$ and $C_i \cap C_j = \emptyset$ for $i \neq j$
 - Each cluster is represented by a **prototype** (also called **centroid** or **mean**) μ_i
 - Prototype does not have to be (and usually is not) one of the objects
 - Clustering quality is based on **Over all dimensions** sum of squared errors between objects in cluster and cluster prototype

$$\sum_{i=1}^k \sum_{x_j \in C_i} \|x_j - \mu_i\|_2^2 = \sum_{i=1}^k \sum_{x_j \in C_i} \sum_{l=1}^d (x_{jl} - \mu_{il})^2$$

Partitions and prototypes

- *Exclusive* representative-based clustering:
 - The set of objects U is partitioned into k **clusters** C_1, C_2, \dots, C_k
 - $\cup_i C_i = U$ and $C_i \cap C_j = \emptyset$ for $i \neq j$
 - Each cluster is represented by a **prototype** (also called **centroid** or **mean**) μ_i
 - Prototype does not have to be (and usually is not) one of the objects
 - Clustering quality is based on sum of squared errors between objects in cluster and cluster prototype

$$\sum_{i=1}^k \sum_{x_j \in C_i} \|x_j - \mu_i\|_2^2 = \sum_{i=1}^k \sum_{x_j \in C_i} \sum_{l=1}^d (x_{jl} - \mu_{il})^2$$

The naïve algorithm

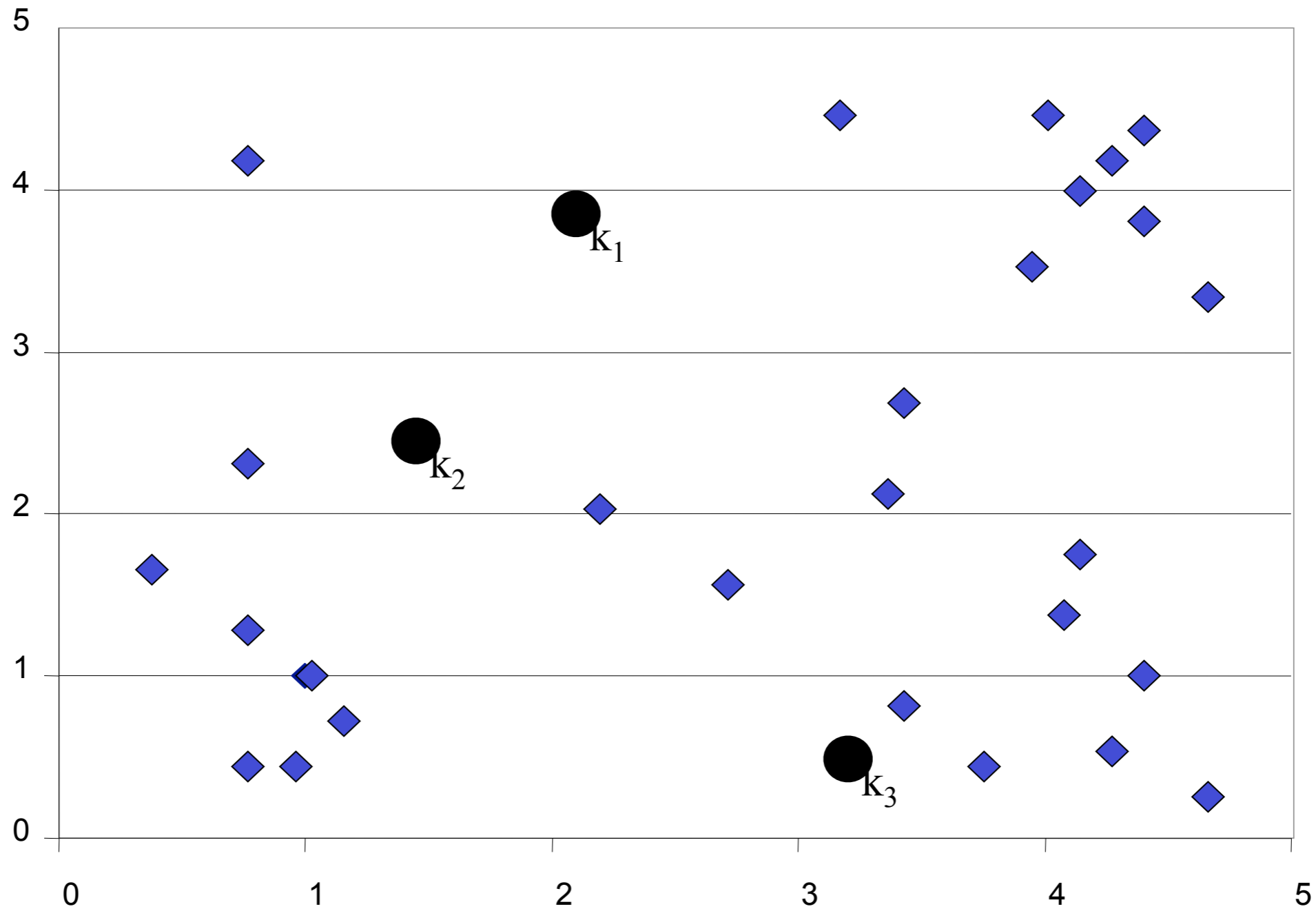
- The naïve algorithm:
 - Generate all possible clusterings one-by-one
 - Compute the squared error
 - Select the best
- But this approach is infeasible
 - There are too many possible clusterings to try
 - k^n different clusterings to k clusters (some possibly empty)
 - The number of ways to cluster n points in k *nonempty* clusters is the *Stirling number of the second kind*, $S(n, k)$,

$$S(n, k) = \left\{ \begin{matrix} n \\ k \end{matrix} \right\} = \frac{1}{k!} \sum_{j=0}^k (-1)^j \binom{k}{j} (k-j)^n$$

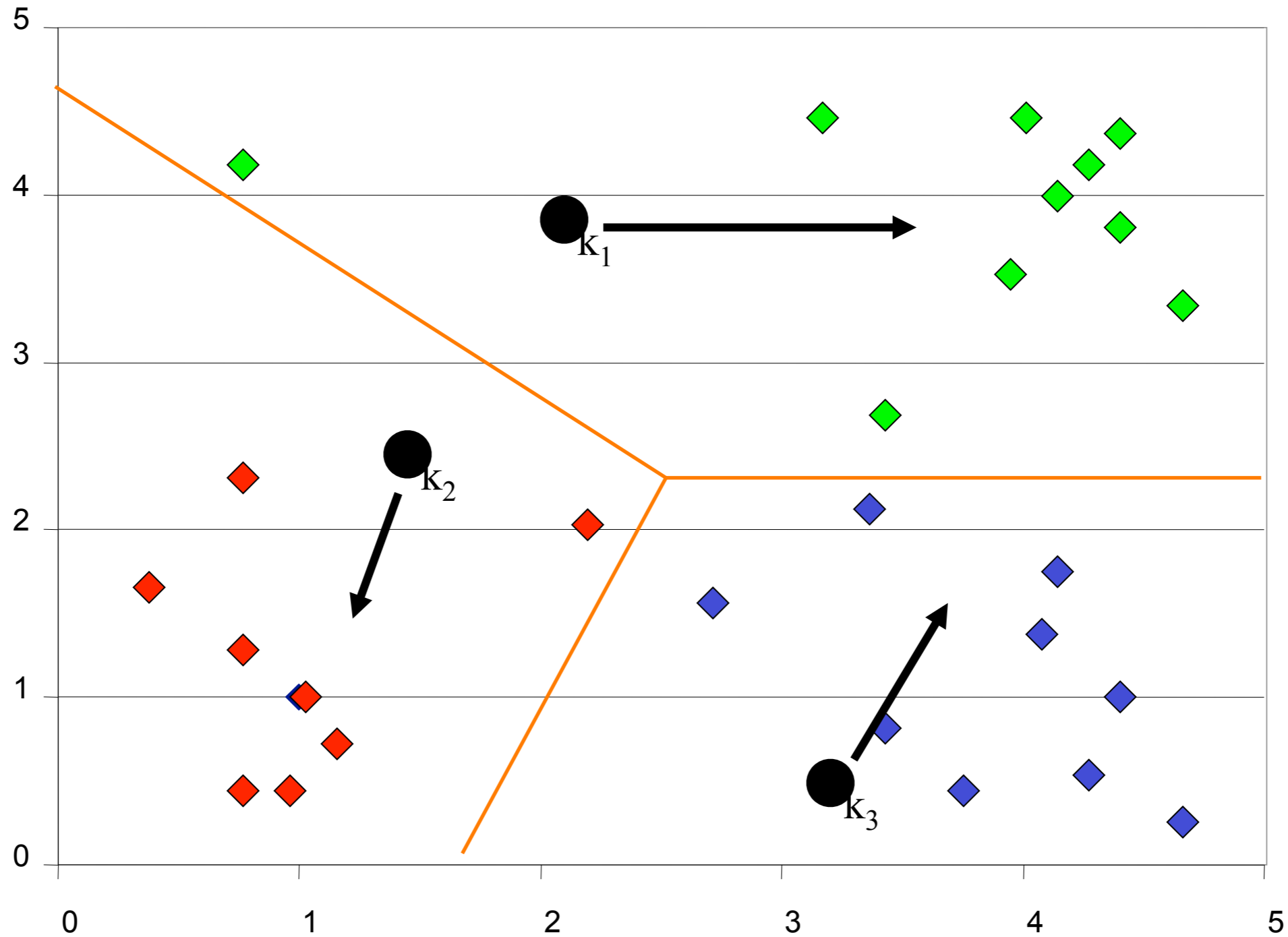
An iterative k -means algorithm

1. select k random cluster centroids
2. assign each point to its closest centroid and compute the error
3. **do**
 - 3.1. **for each** cluster C_i
 - 3.1.1. compute new centroid as $\mu_i = \frac{1}{|C_i|} \sum_{x_j \in C_i} x_j$
 - 3.2. **for each** element $x_j \in U$
 - 3.2.1. assign x_j to its closest cluster centroid
4. **while** error decreases

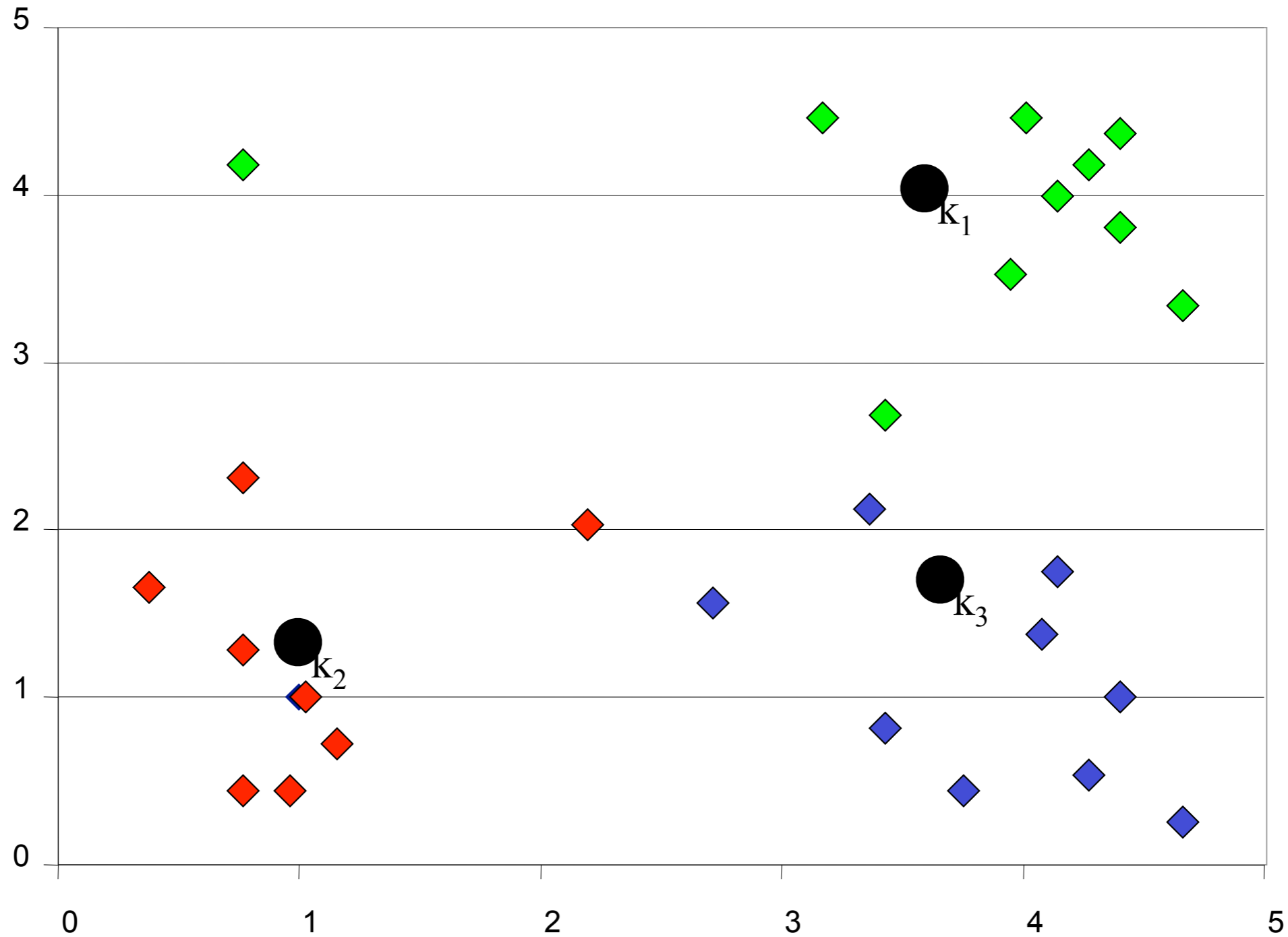
k -means example



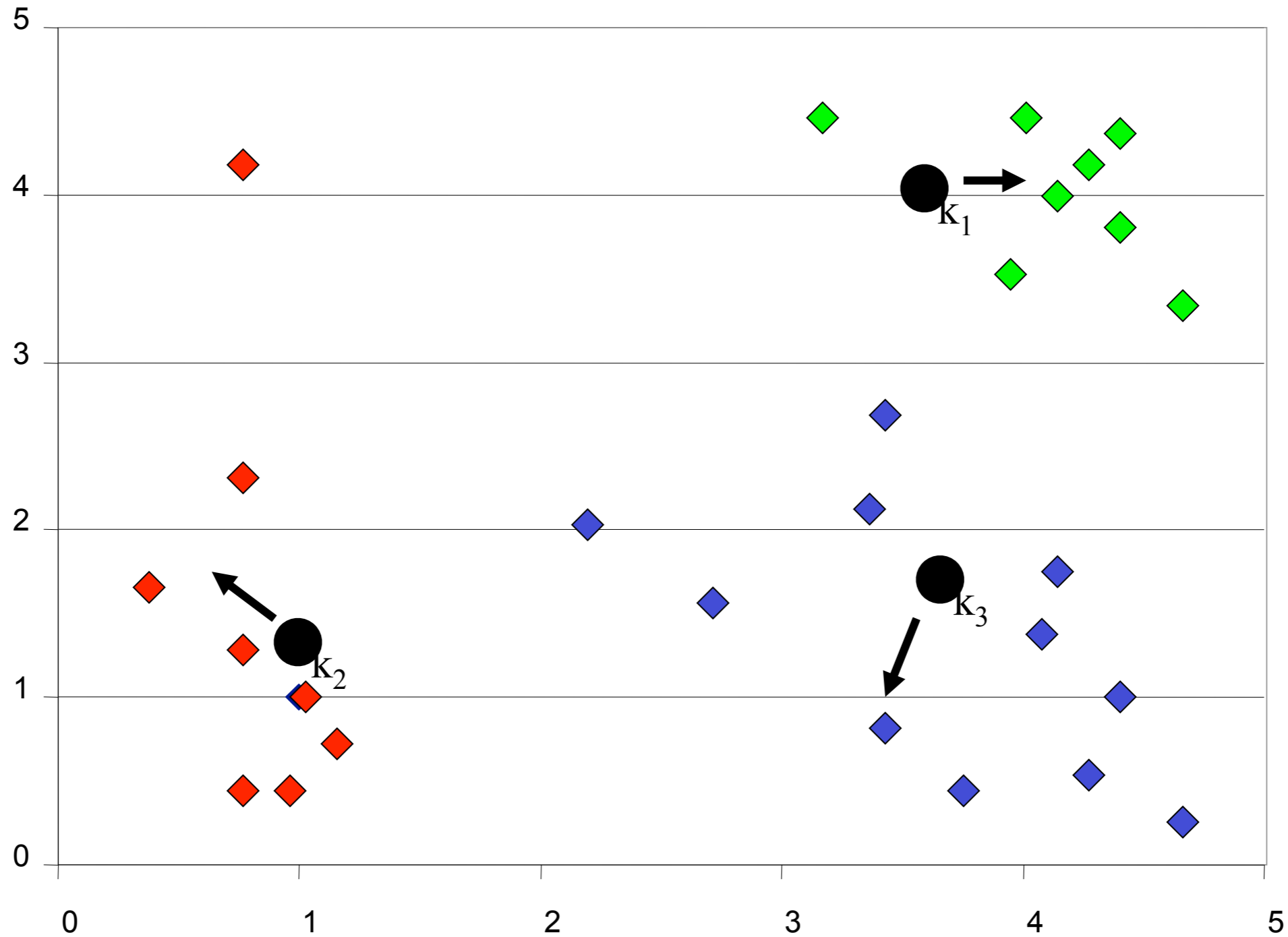
k -means example



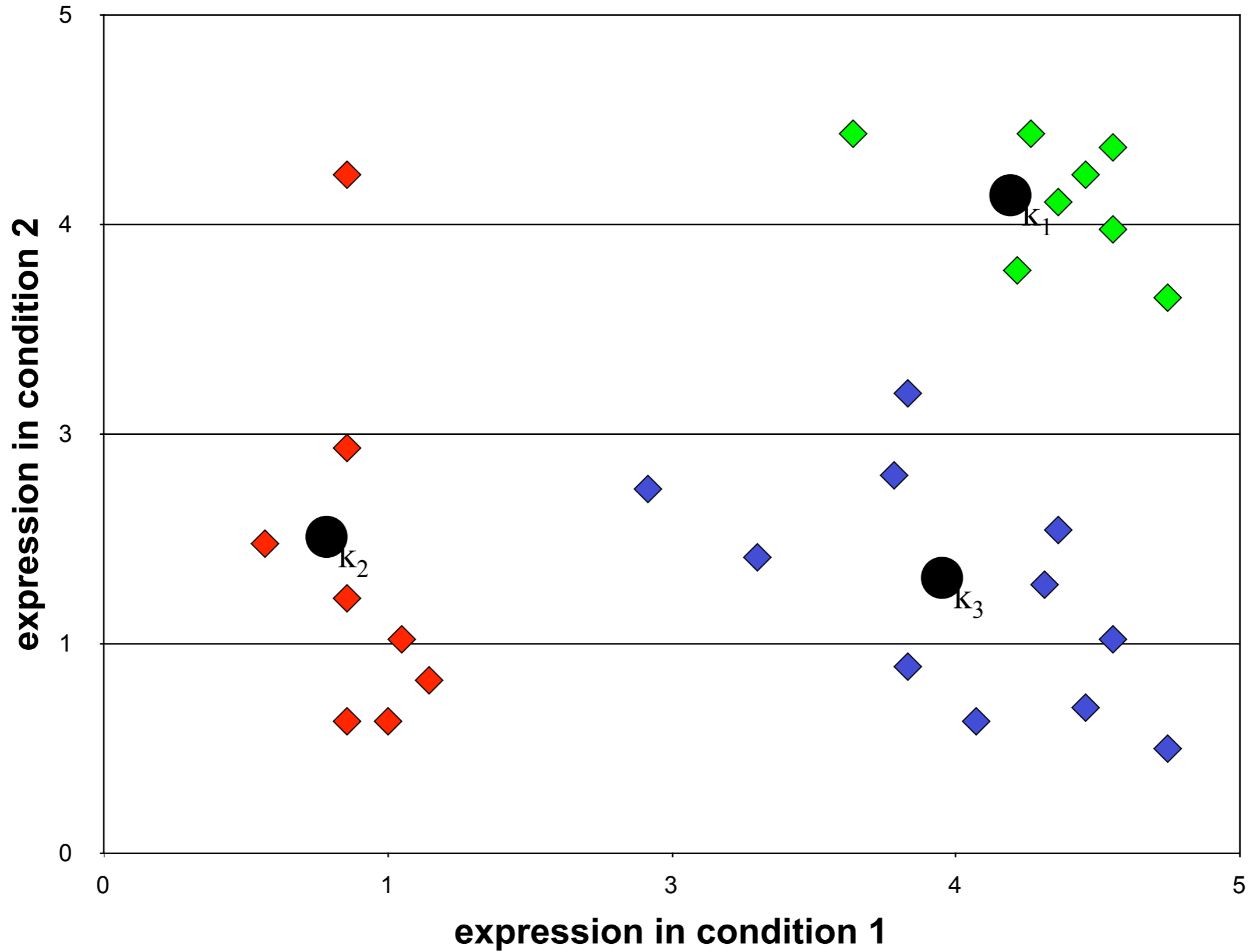
k -means example



k -means example



k -means example



Some notes on the algorithm

- Always converges eventually
 - On each step the error decreases
 - Only finite number of possible clusterings
 - Convergence to *local optimum*
- At some point a cluster can become empty
 - All points are closer to some other centroid
 - Some options:
 - Split the biggest cluster
 - Take the furthest point as a singleton cluster
- Outliers can yield bad clusterings

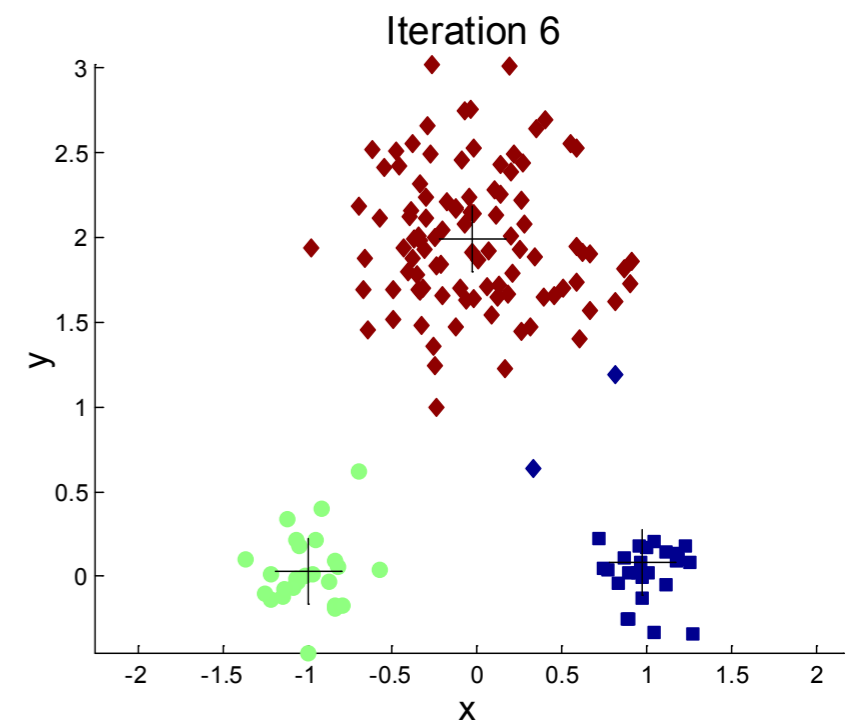
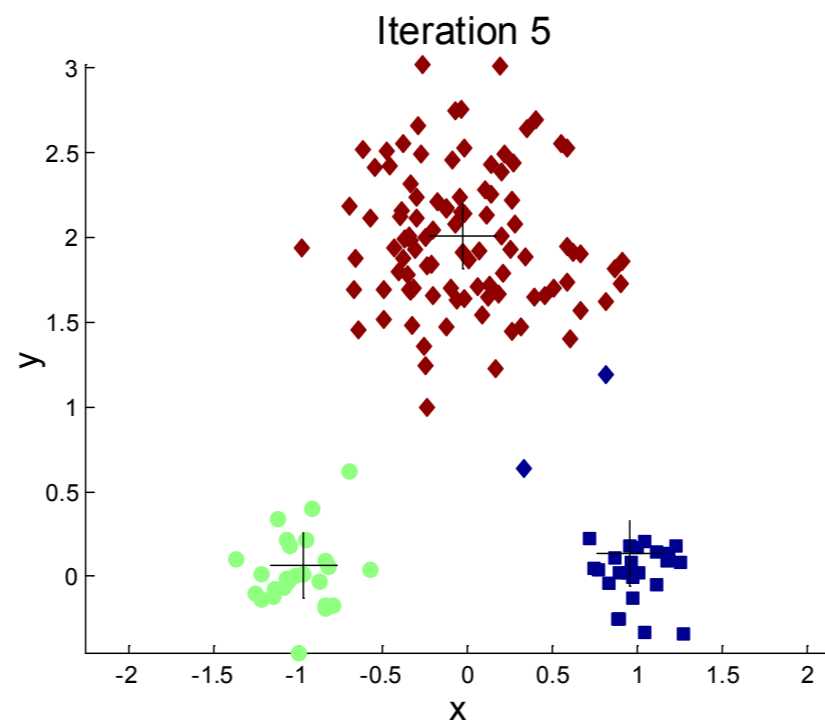
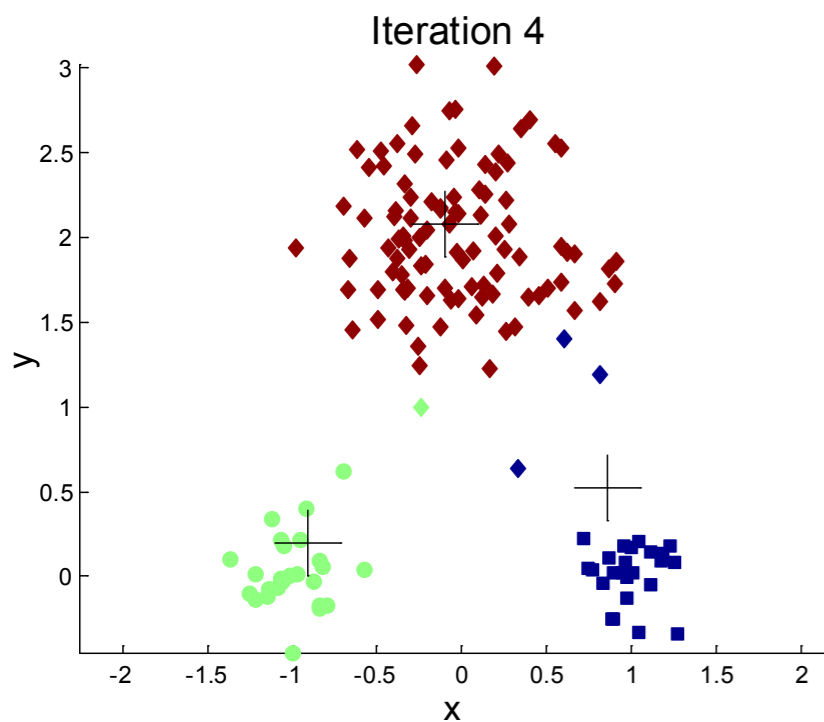
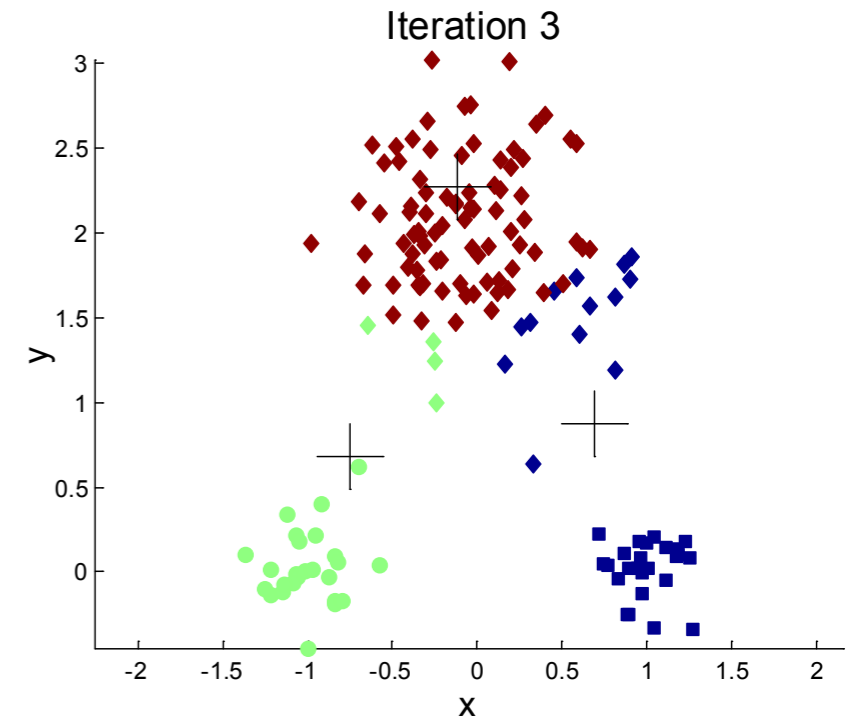
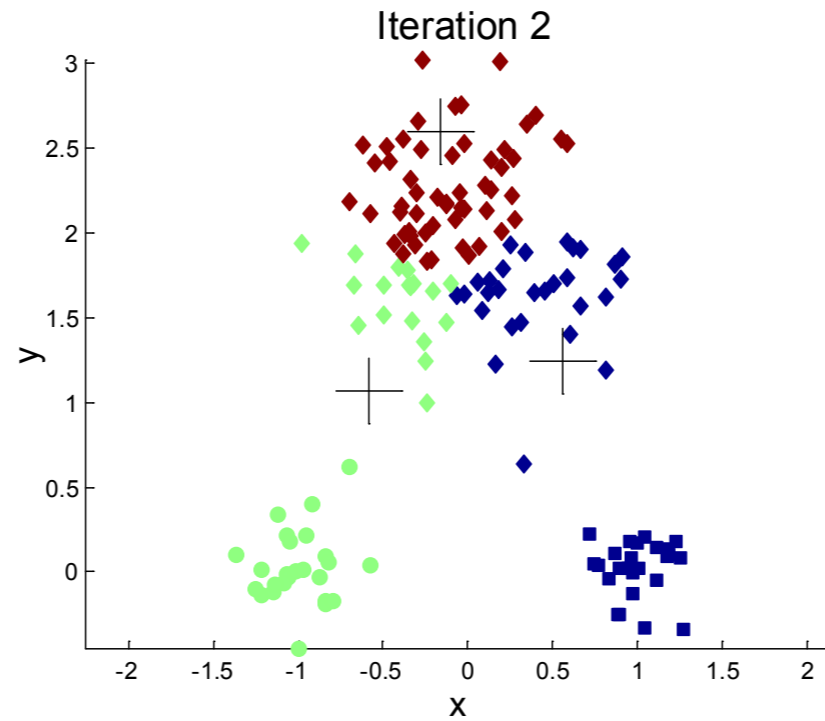
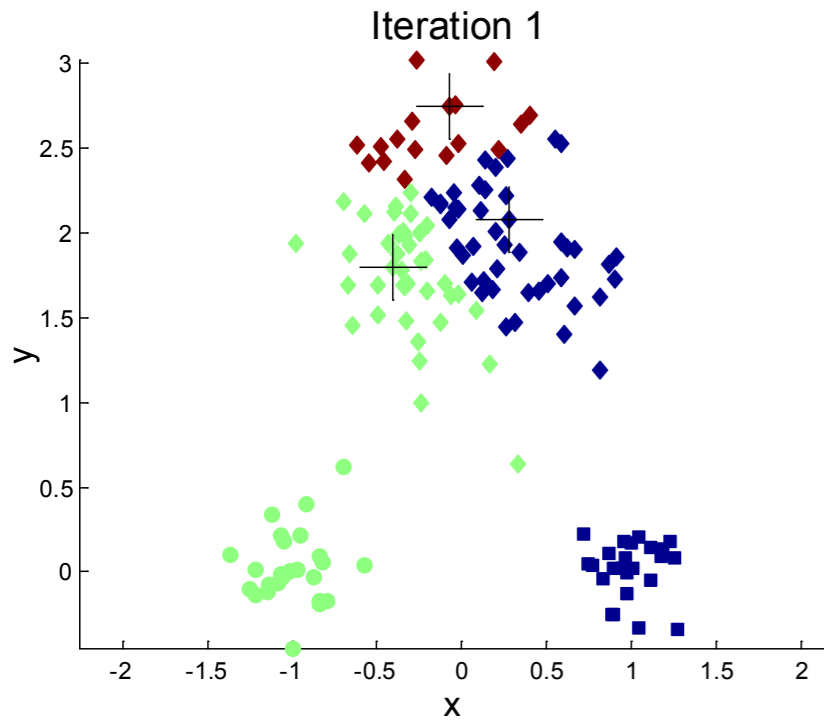
Computational complexity

- How long does the iterative k -means algorithm take?
 - Computing the centroid takes $O(nd)$ time
 - Averages over total of n points in d -dimensional space
 - Computing the cluster assignment takes $O(nkd)$ time
 - For each n points we have to compute the distance to all k clusters in d -dimensional space
 - If the algorithm takes t iterations, the total running time is $O(tnkd)$
 - But how many iterations we need?

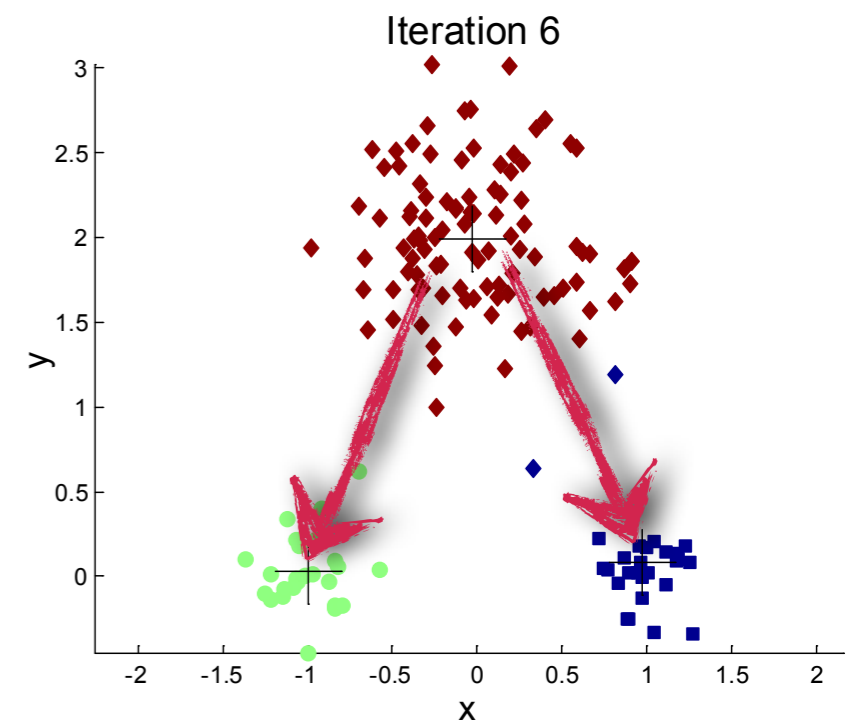
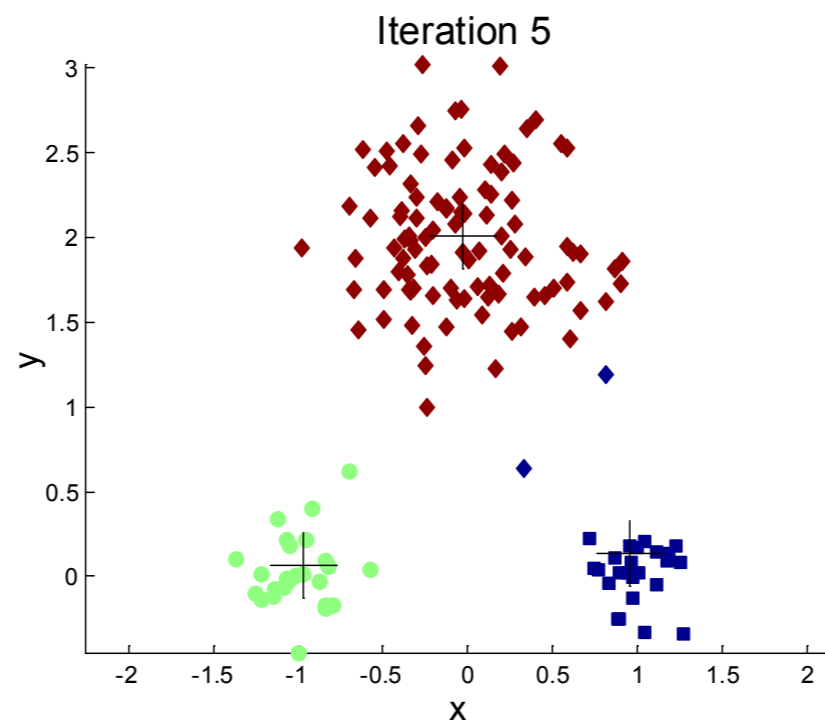
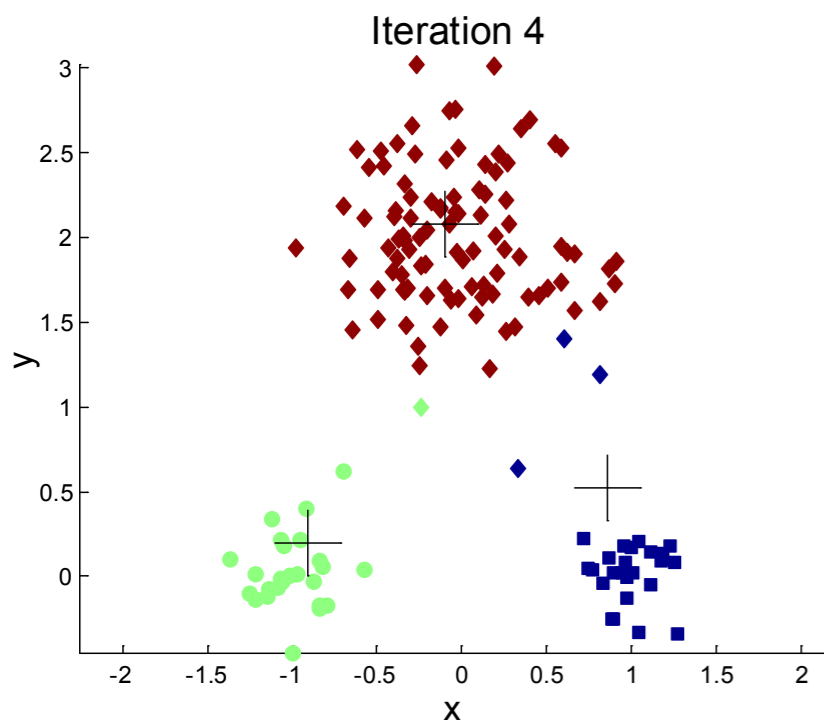
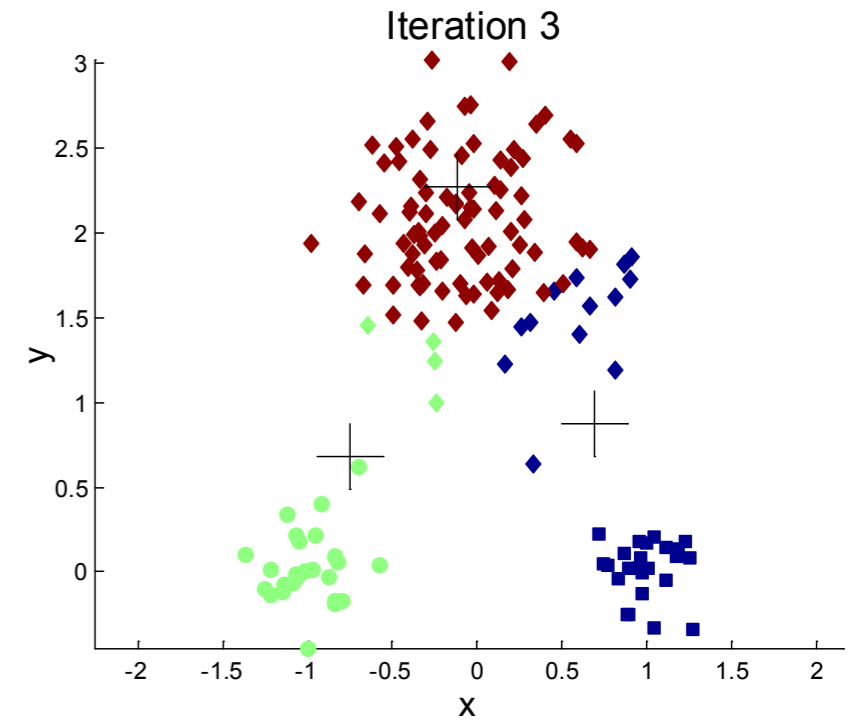
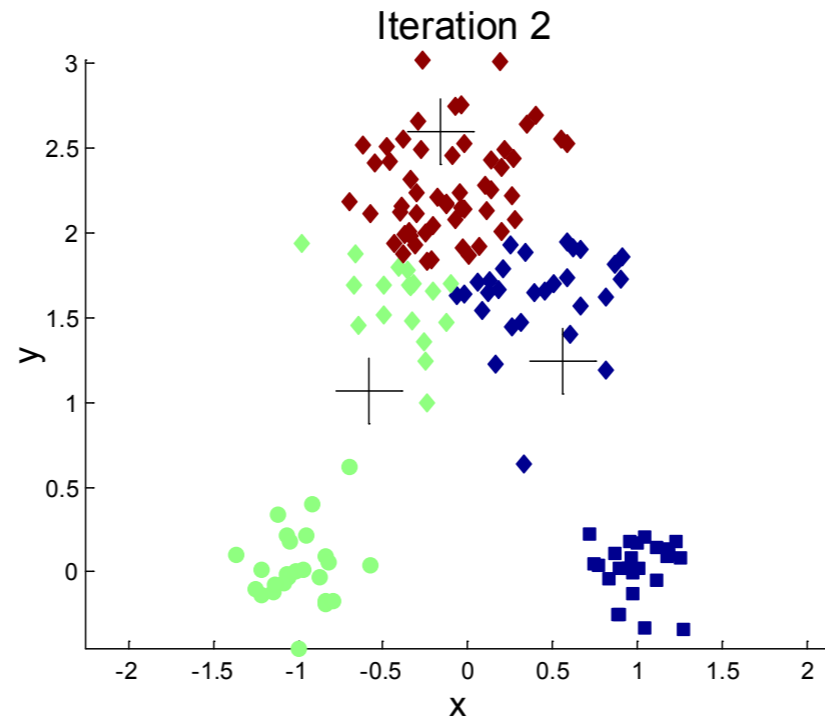
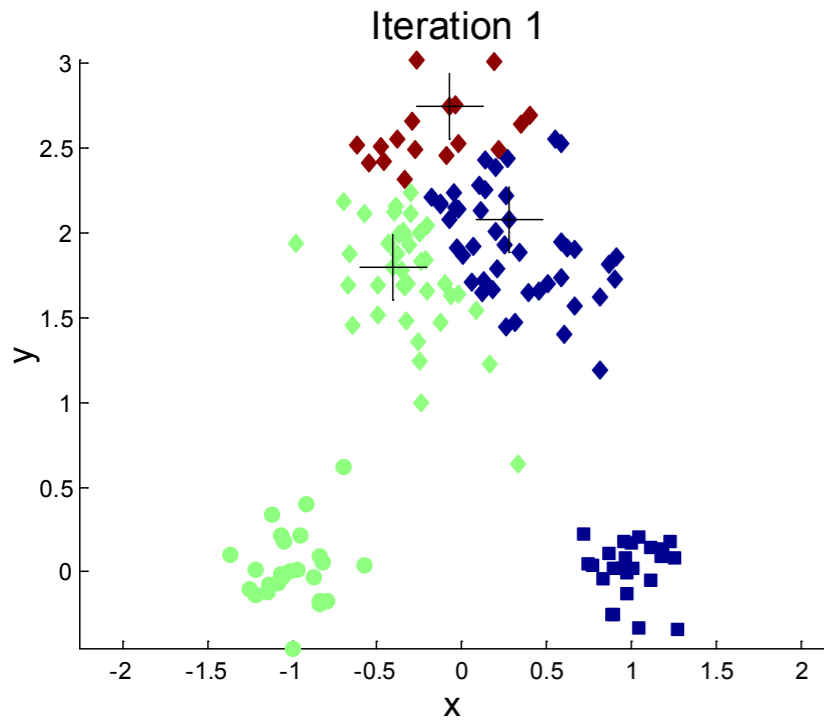
How many iterations?

- In practice the algorithm doesn't usually take many iterations
 - Some hundred iterations is usually enough
- Worst-case upper bound is $O(n^{dk})$
- Worst-case lower bound is superpolynomial: $2^{\Omega(\sqrt{n})}$
- The discrepancy between practice and worst-case analysis can be (somewhat) explained with *smoothed analysis* [Arthur & Vassilvitskii '06]:
 - If the data is sampled from independent d -dimensional normal distributions with same variance, iterative k -means algorithm will terminate in time $O(n^k)$ with high probability.

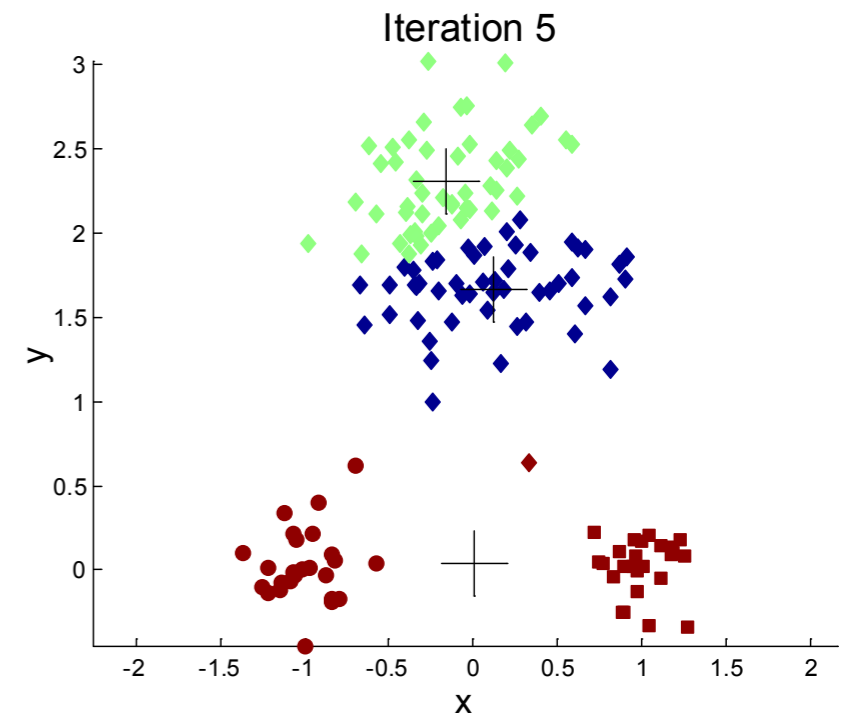
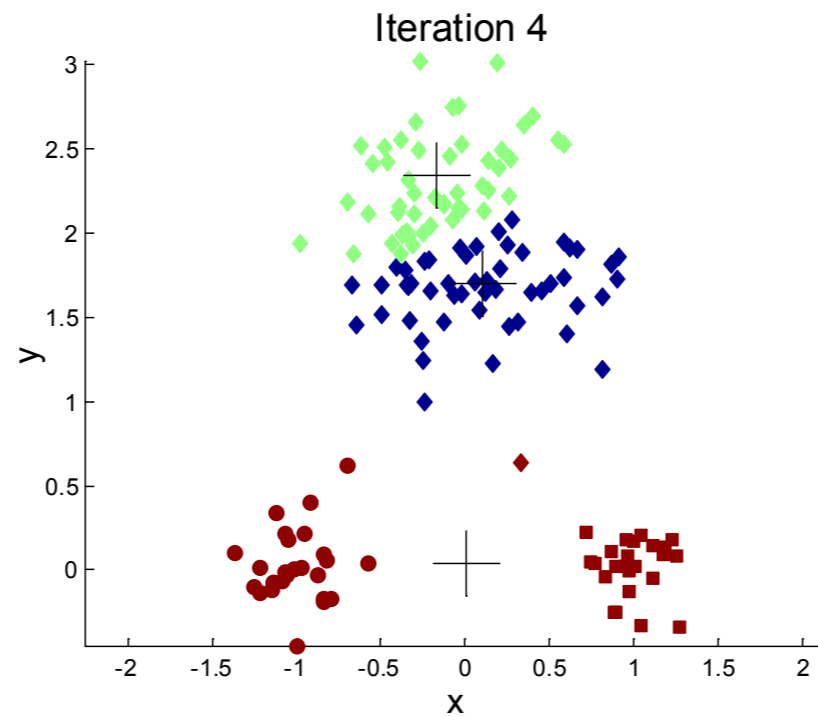
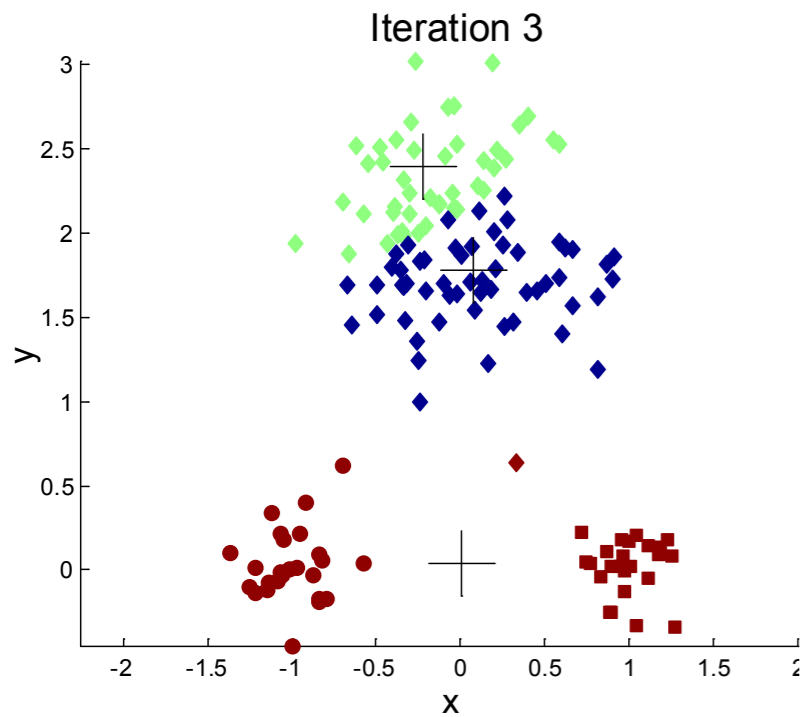
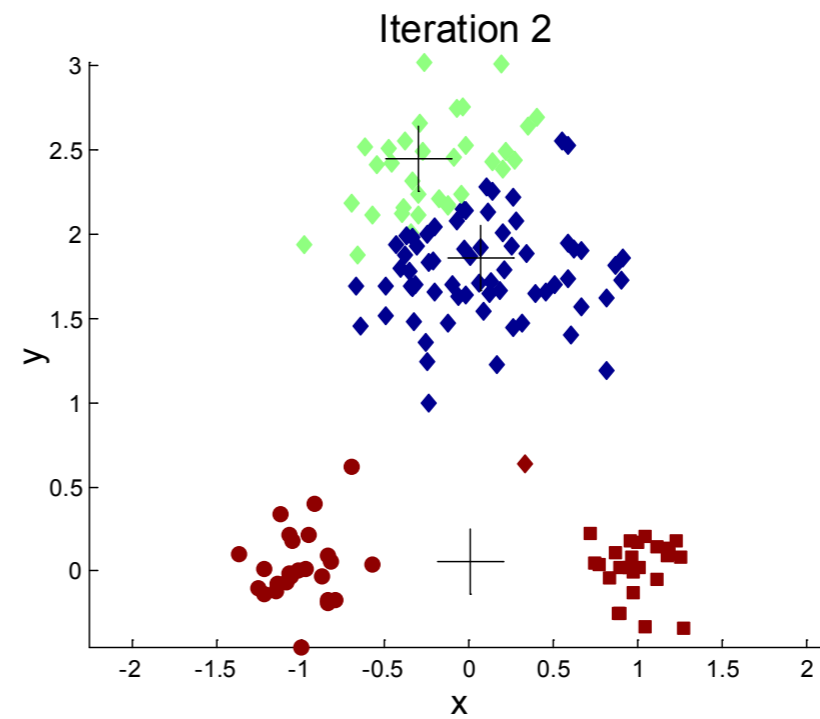
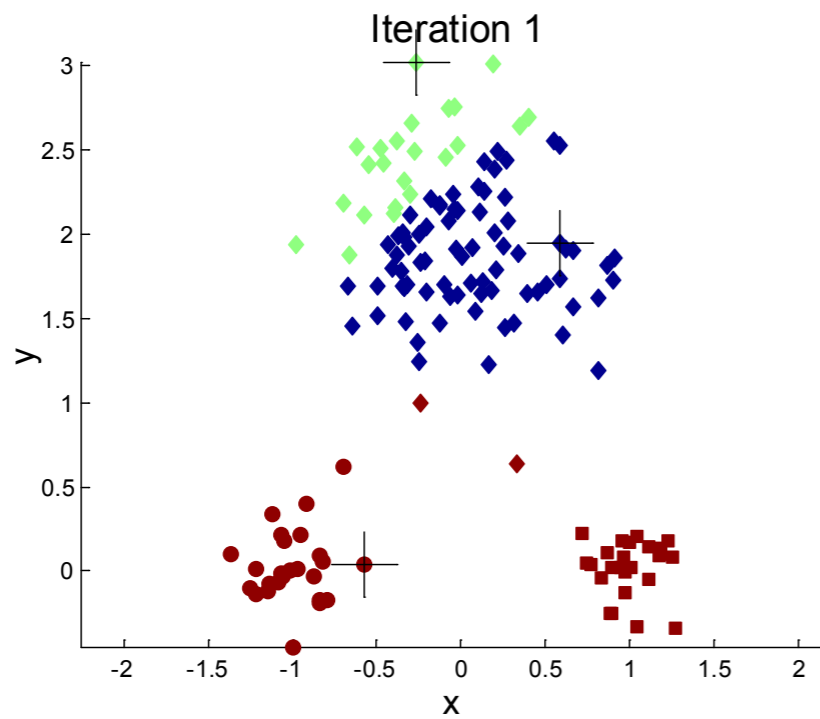
On the importance of initial centroids



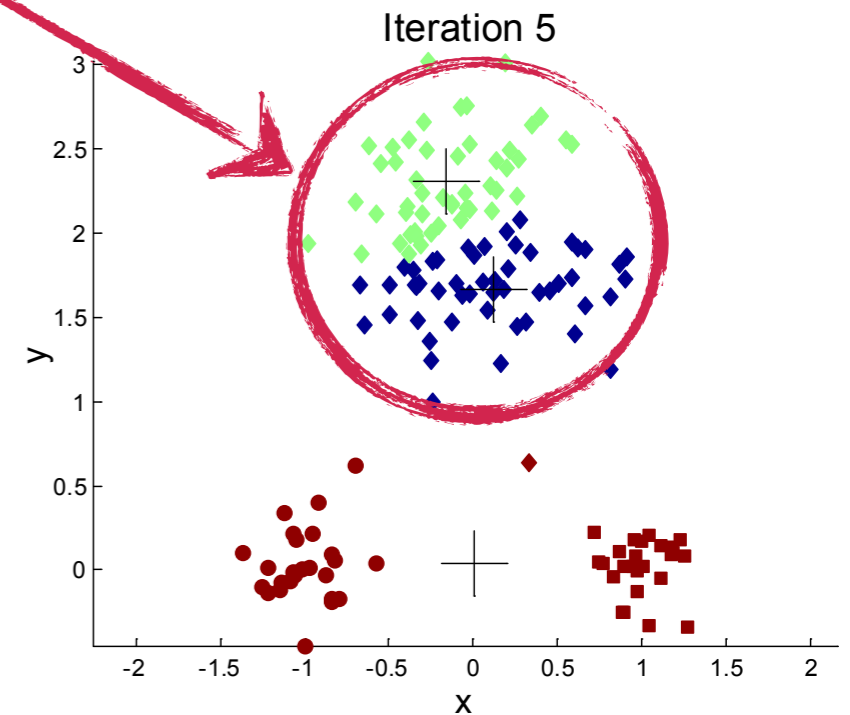
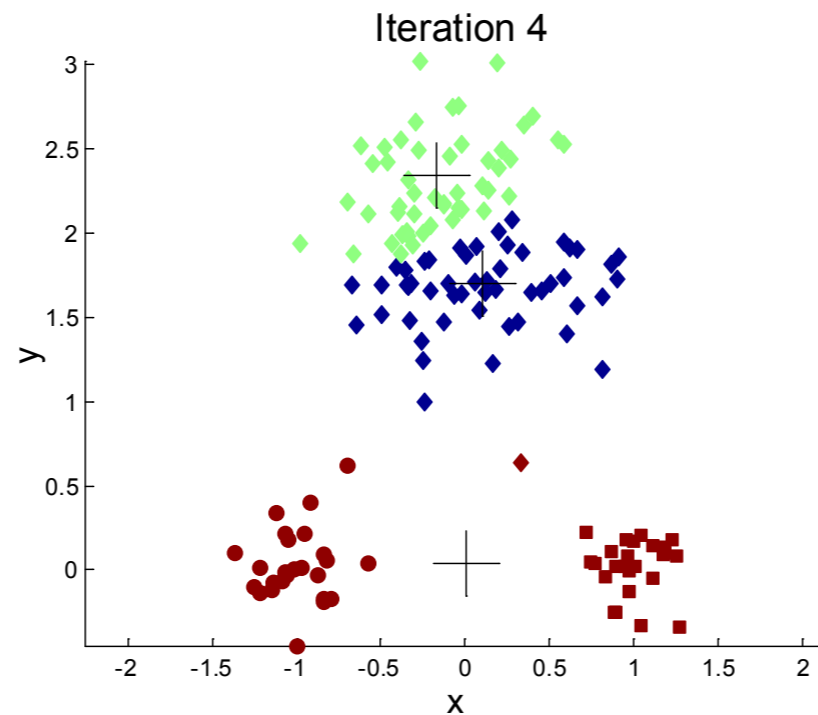
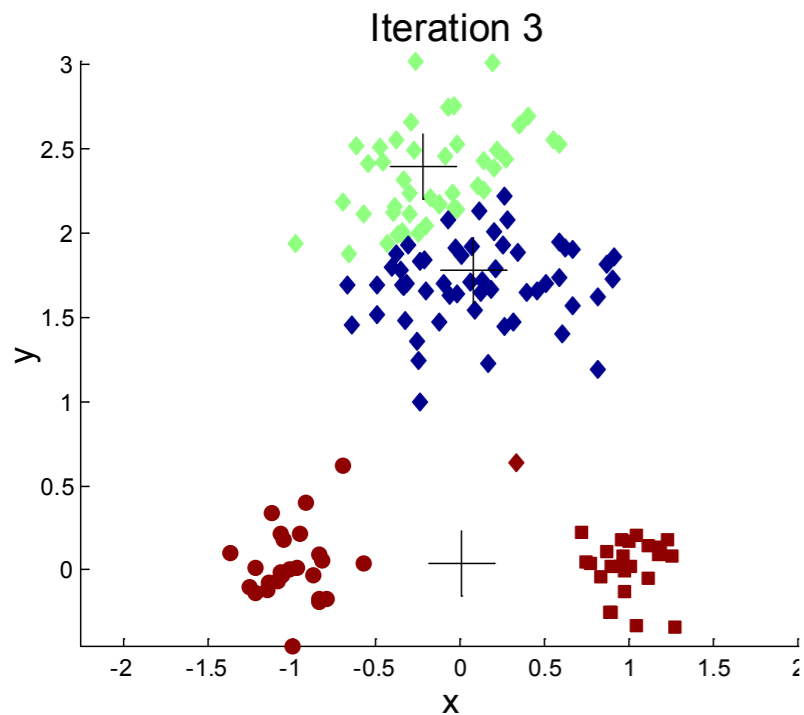
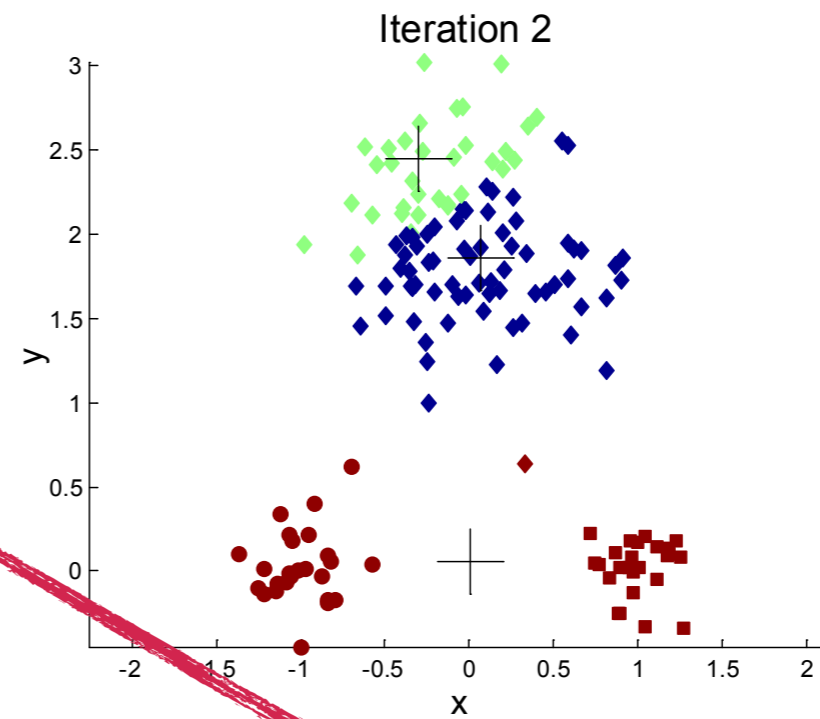
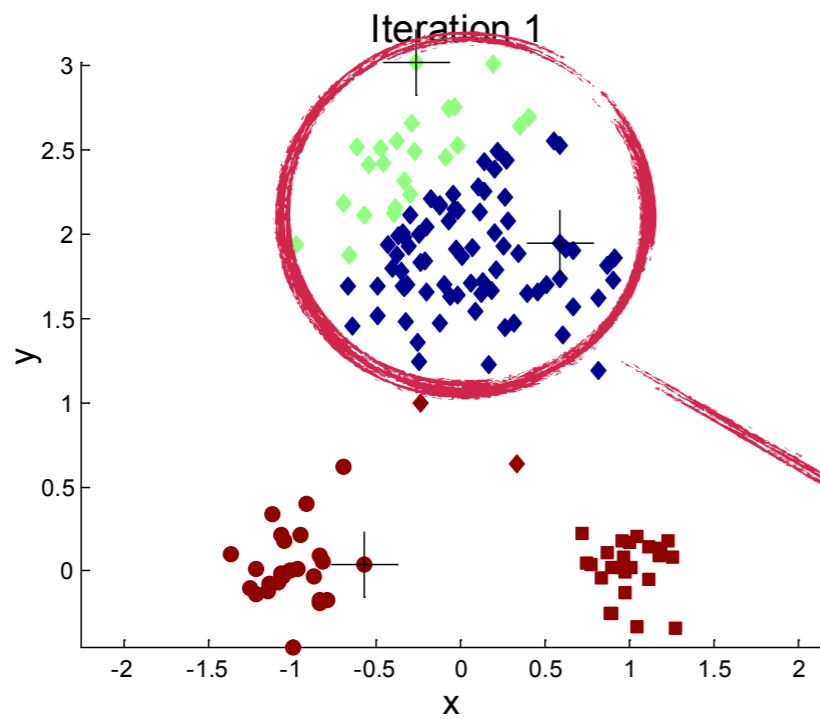
On the importance of initial centroids



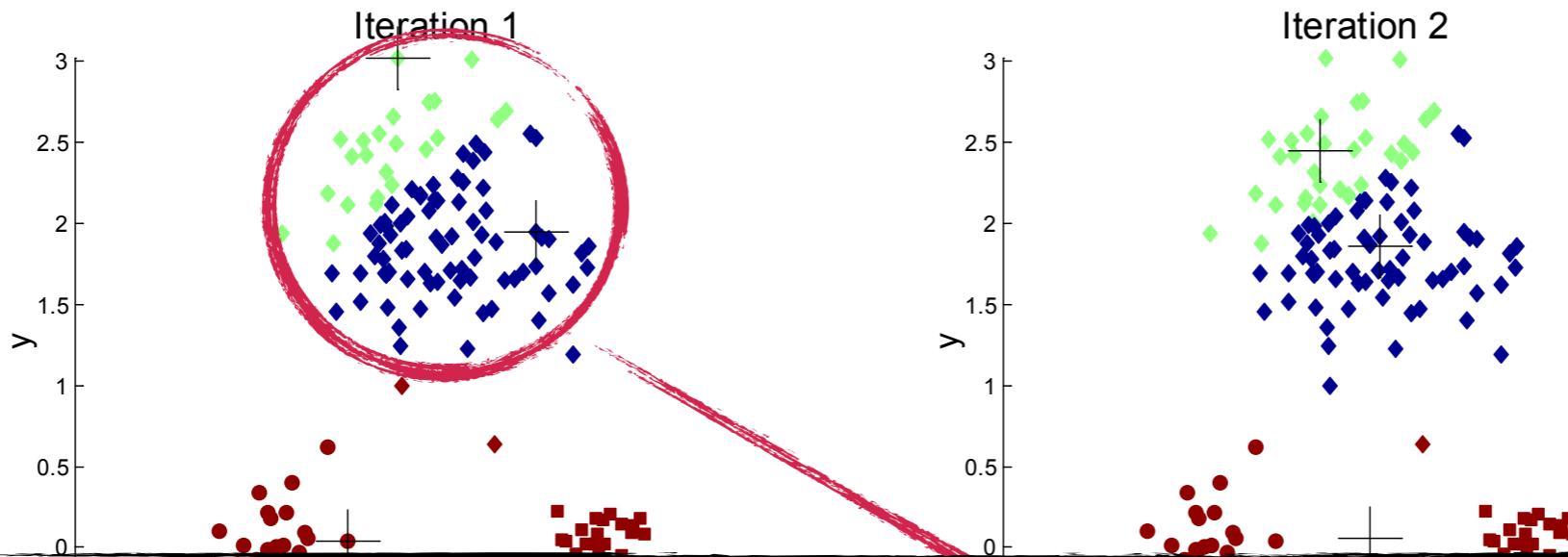
On the importance of initial centroids



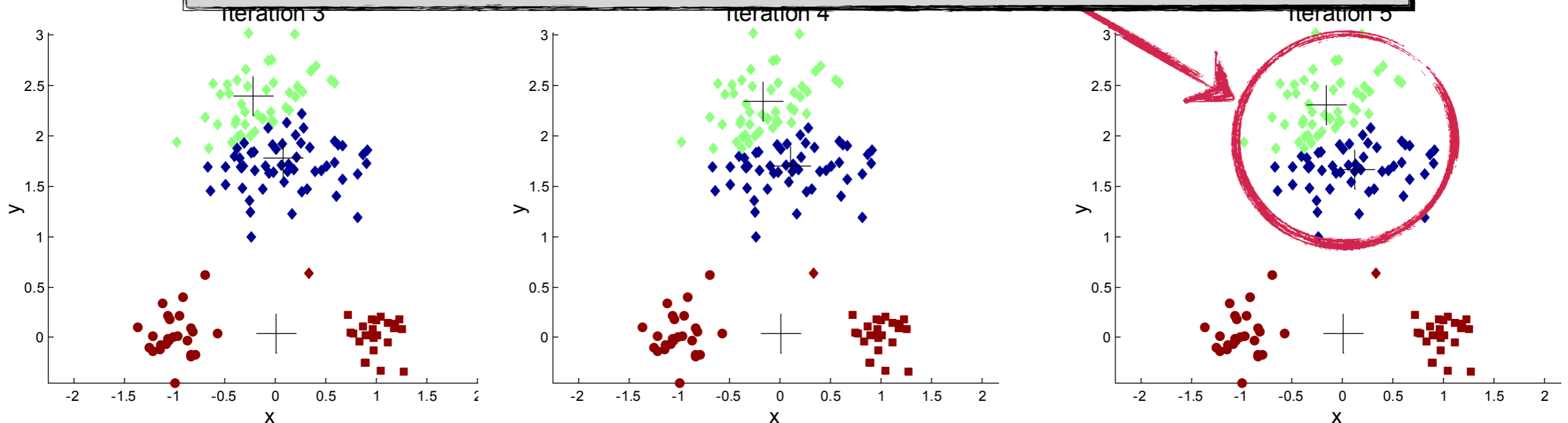
On the importance of initial centroids



On the importance of initial centroids



The k-means algorithm converges to local optimum which can be arbitrary bad vs. the global optimum.



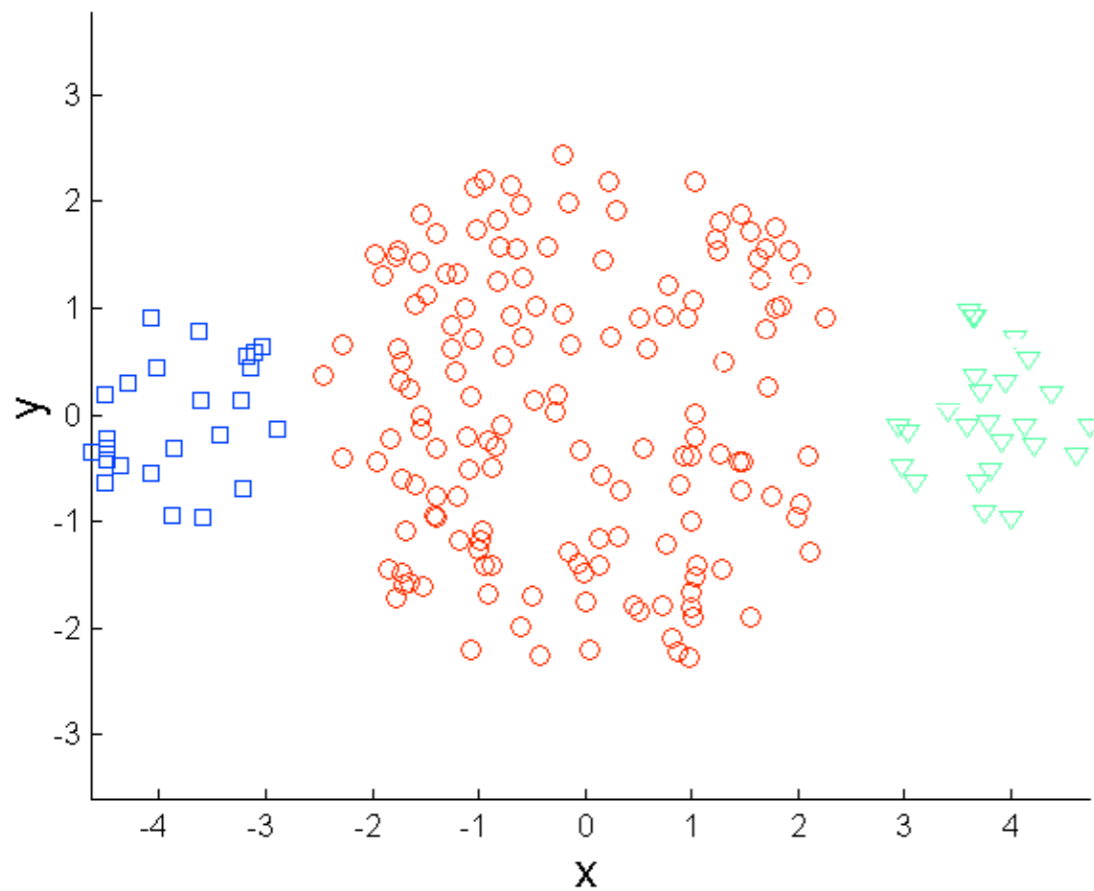
The k -means++ algorithm

- Careful initial seeding [Arthur & Vassilvitskii '07]:
 - Choose first centroid u.a.r. from data points
 - Let $D(x)$ be the shortest distance from x to any already-selected centroid
 - Choose next centroid to be x' with probability $\frac{D(x')^2}{\sum_{x \in X} D(x)^2}$
 - Points that are further away are selected more probably
 - Repeat until k centroids have been selected and continue as normal iterative k -means algorithm
- The k -means++ algorithm achieves $O(\log k)$ approximation ratio on expectation
 - $\mathbf{E}[\text{cost}] \leq 8(\ln k + 2)\text{OPT}$
- The k -means++ algorithm converges fast in practice

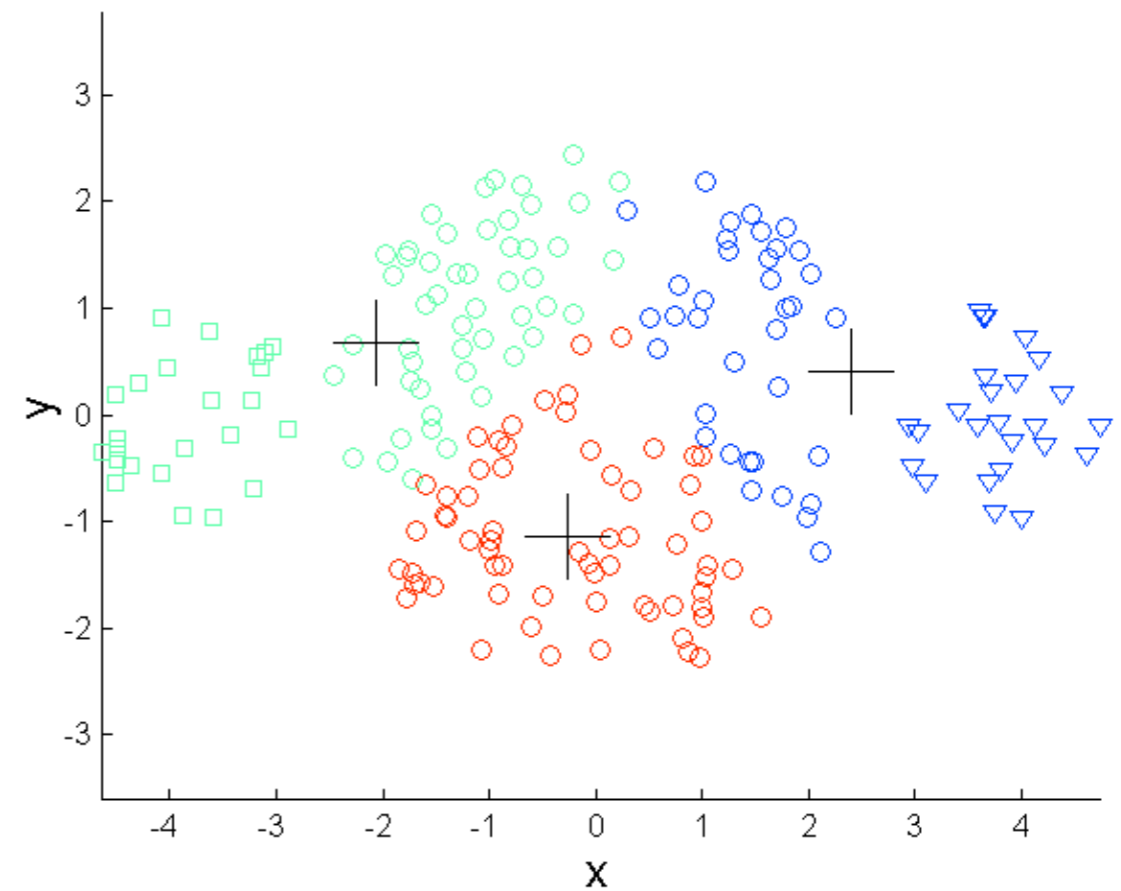
Limitations of cluster types for k -means

Limitations of cluster types for k -means

- The clusters have to be of roughly equal size



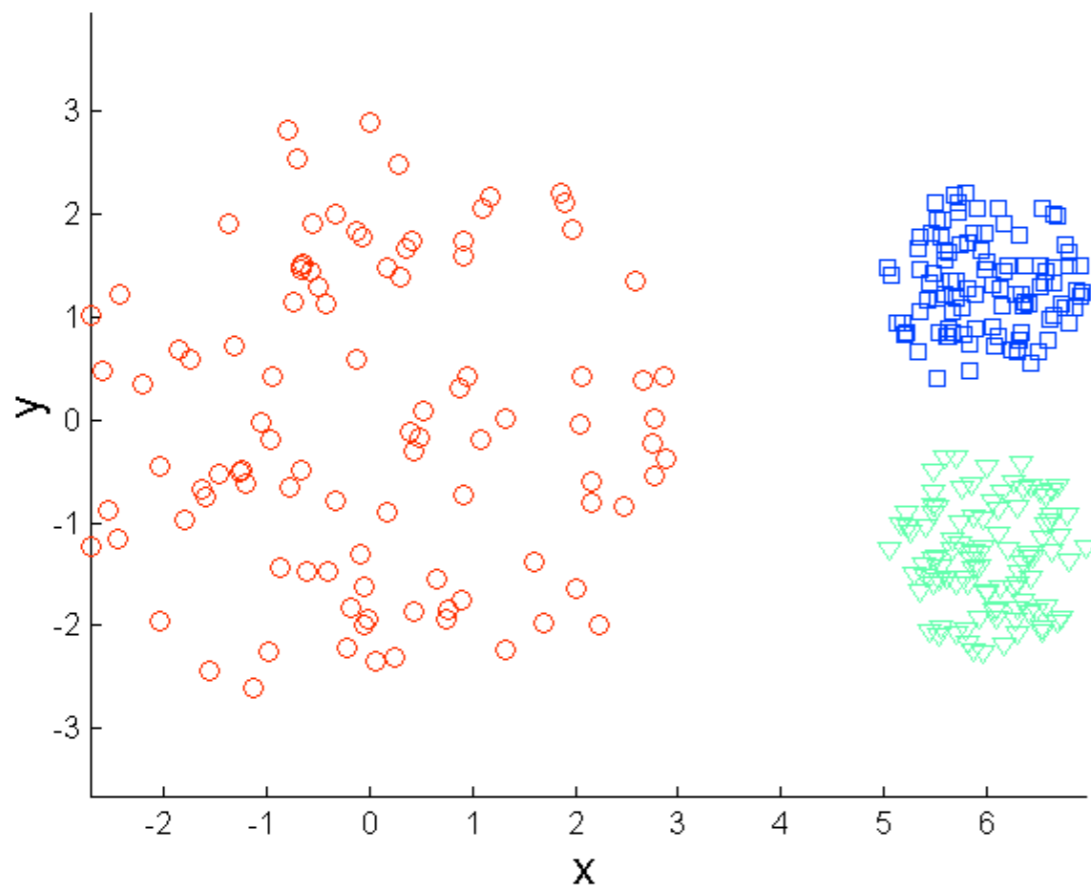
Original Points



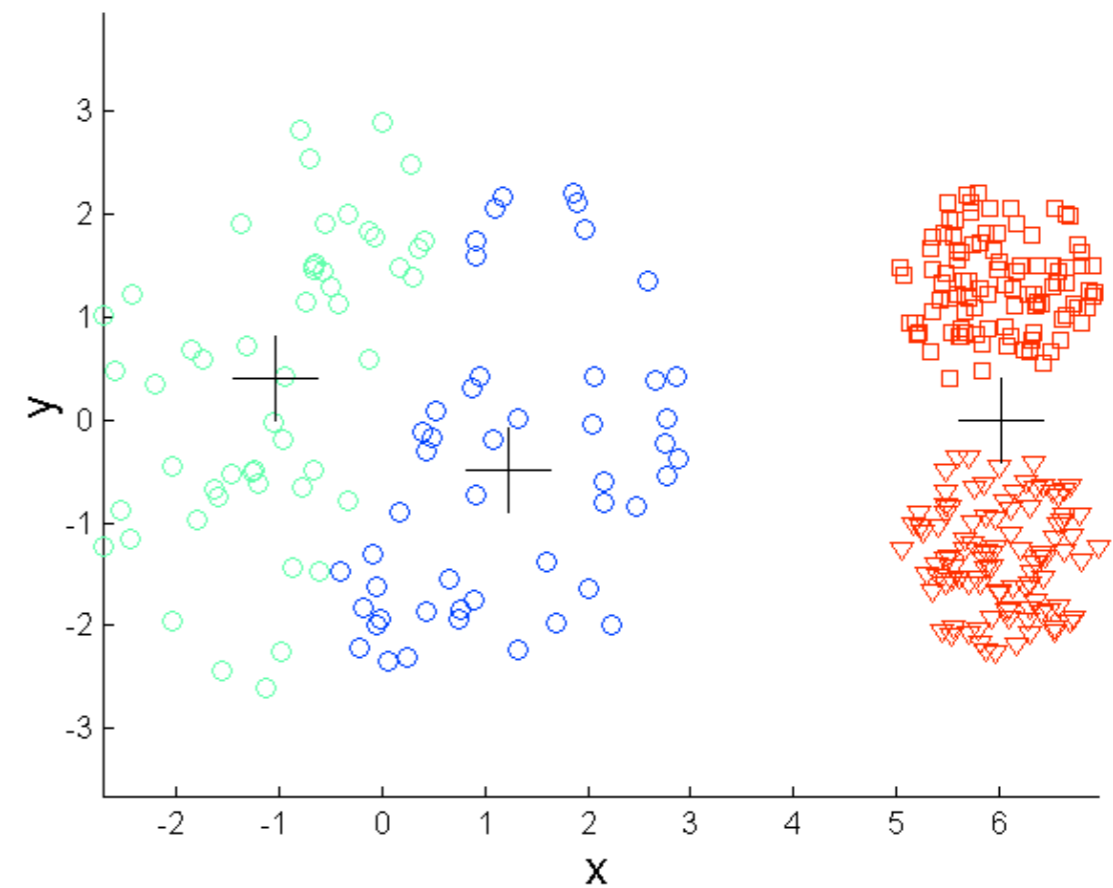
K-means (3 Clusters)

Limitations of cluster types for k -means

- The clusters have to be of roughly equal size
- The clusters have to be of roughly equal density



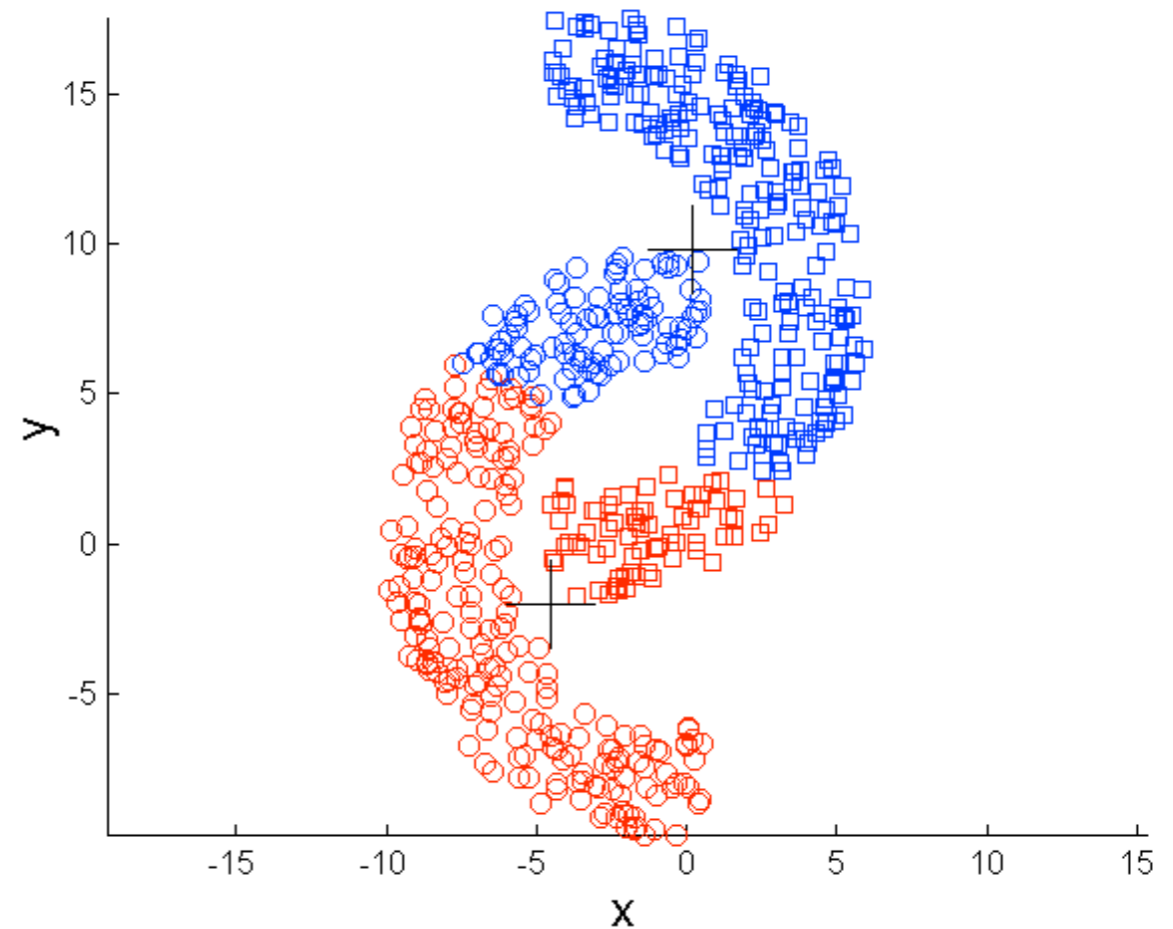
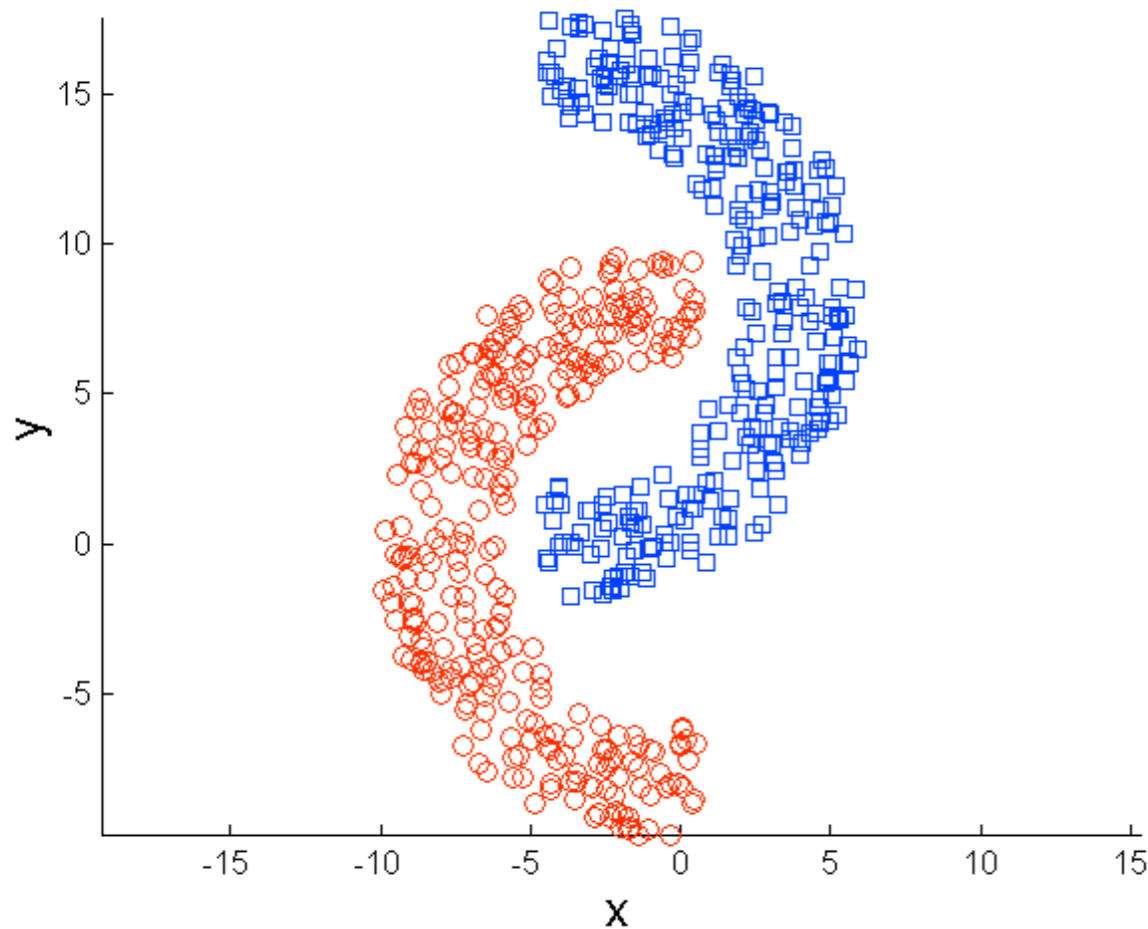
Original Points



K-means (3 Clusters)

Limitations of cluster types for k -means

- The clusters have to be of roughly equal size
- The clusters have to be of roughly equal density
- The clusters have to be of roughly spherical shape



The EM clustering algorithm

- *Probabilistic* clustering
 - I.e. not exclusive
- *Representative* in a way
 - Each cluster is represented by some *parameters*
 - The parameters can include cluster centroid
- Requires us to assume something about the distribution of the points
 - For now, each cluster is independent Gaussian
- We use the expectation-maximization approach

The basics

- We aim at finding parameters $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ for each Gaussian cluster plus k *mixture parameters* $P(C_i)$ (all together denoted by $\boldsymbol{\theta}$)

– pdf of point \boldsymbol{x} in cluster i is

$$f_i(\boldsymbol{x}) = f(\boldsymbol{x} \mid \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = (2\pi)^{-\frac{d}{2}} |\boldsymbol{\Sigma}_i|^{-\frac{1}{2}} \exp \left\{ -\frac{(\boldsymbol{x} - \boldsymbol{\mu}_i)^\top \boldsymbol{\Sigma}_i^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_i)}{2} \right\}$$

– Total pdf of \boldsymbol{x} is a *mixture model* of the k cluster Gaussians:

$$f(\boldsymbol{x}) = \sum_{i=1}^k f_i(\boldsymbol{x}) P(C_i) = \sum_{i=1}^k f(\boldsymbol{x} \mid \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) P(C_i)$$

– The log-likelihood of the data D given parameters $\boldsymbol{\theta}$ is then

$$\ln P(D \mid \boldsymbol{\theta}) = \sum_{j=1}^n \ln \left(\sum_{i=1}^k f(\boldsymbol{x}_j \mid \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) P(C_i) \right)$$

The general EM clustering algorithm

- **Initialization**

- Initialize parameters θ randomly

- **Expectation step**

- Compute the posterior probability $P(C_i | \mathbf{x}_j)$

- Per Bayes's theorem

$$P(C_i | \mathbf{x}_j) = \frac{P(\mathbf{x}_j | C_i)P(C_i)}{\sum_{a=1}^k P(\mathbf{x}_j | C_a)P(C_a)}$$

- **Maximization step**

- Re-estimate θ given $P(C_i | \mathbf{x}_j)$

- **Repeat E and M steps until convergence**

EM with Gaussians in 1-D

- Now pdf is $f(x | \mu_i, \sigma_i^2) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp \left\{ -\frac{(x-\mu_i)^2}{2\sigma_i^2} \right\}$

- **Initialization step**

- Mean μ is sampled u.a.r. from possible values, $\sigma^2 = 1$, and $P(C_i) = 1/k$ (each cluster is equiprobable)

- **Expectation step**

$$w_{ij} = P(C_i | x_j) = \frac{f(x_j | \mu_i, \sigma_i^2) P(C_i)}{\sum_{a=1}^k f(x_j | \mu_a, \sigma_a^2) P(C_a)}$$

- **Maximization step**

$$\mu_i = \frac{\sum_{j=1}^n w_{ij} x_j}{\sum_{j=1}^n w_{ij}} \quad \sigma_i^2 = \frac{\sum_{j=1}^n w_{ij} (x_j - \mu_i)^2}{\sum_{j=1}^n w_{ij}} \quad P(C_i) = \frac{\sum_{j=1}^n w_{ij}}{n}$$

EM with Gaussians in 1-D

- Now pdf is $f(x | \mu_i, \sigma_i^2) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp\left\{-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right\}$
- **Initialization step**
 - Mean μ is sampled u.a.r. from possible values, $\sigma^2 = 1$, and $P(C_i) = 1/k$ (each cluster is equiprobable)

- **Expectation step**

$$w_{ij} = P(C_i | x_j) = \frac{f(x_j | \mu_i, \sigma_i^2)P(C_i)}{\sum_{a=1}^k f(x_j | \mu_a, \sigma_a^2)P(C_a)}$$

- **Weighted mean** on step

$$\mu_i = \frac{\sum_{j=1}^n w_{ij} x_j}{\sum_{j=1}^n w_{ij}} \quad \sigma_i^2 = \frac{\sum_{j=1}^n w_{ij} (x_j - \mu_i)^2}{\sum_{j=1}^n w_{ij}} \quad P(C_i) = \frac{\sum_{j=1}^n w_{ij}}{n}$$

EM with Gaussians in 1-D

- Now pdf is $f(x | \mu_i, \sigma_i^2) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp\left\{-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right\}$
- **Initialization step**
 - Mean μ is sampled u.a.r. from possible values, $\sigma^2 = 1$, and $P(C_i) = 1/k$ (each cluster is equiprobable)
- **Expectation step**

$$w_{ij} = P(C_i | x_j) = \frac{f(x_j | \mu_i, \sigma_i^2)P(C_i)}{\sum_{a=1}^k f(x_j | \mu_a, \sigma_a^2)P(C_a)}$$

• **Weighted mean**

on step

Weighted variance

$$\mu_i = \frac{\sum_{j=1}^n w_{ij} x_j}{\sum_{j=1}^n w_{ij}}$$

$$\sigma_i^2 = \frac{\sum_{j=1}^n w_{ij} (x_j - \mu_i)^2}{\sum_{j=1}^n w_{ij}}$$

$$P(C_i) = \frac{\sum_{j=1}^n w_{ij}}{n}$$

EM with Gaussians in 1-D

- Now pdf is $f(x | \mu_i, \sigma_i^2) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp\left\{-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right\}$
- **Initialization step**
 - Mean μ is sampled u.a.r. from possible values, $\sigma^2 = 1$, and $P(C_i) = 1/k$ (each cluster is equiprobable)
- **Expectation step**

$$w_{ij} = P(C_i | x_j) = \frac{f(x_j | \mu_i, \sigma_i^2)P(C_i)}{\sum_{a=1}^k f(x_j | \mu_a, \sigma_a^2)P(C_a)}$$

• **Weighted mean** on step

Weighted variance

Fraction of weight in cluster i

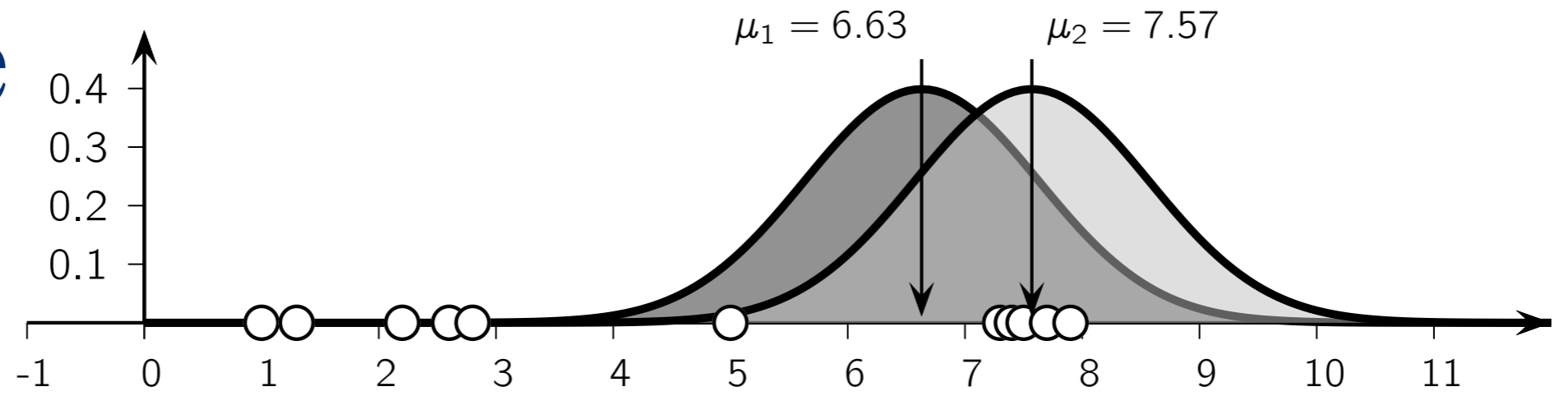
$$\mu_i = \frac{\sum_{j=1}^n w_{ij} x_j}{\sum_{j=1}^n w_{ij}}$$

$$\sigma_i^2 = \frac{\sum_{j=1}^n w_{ij} (x_j - \mu_i)^2}{\sum_{j=1}^n w_{ij}}$$

$$P(C_i) = \frac{\sum_{j=1}^n w_{ij}}{n}$$

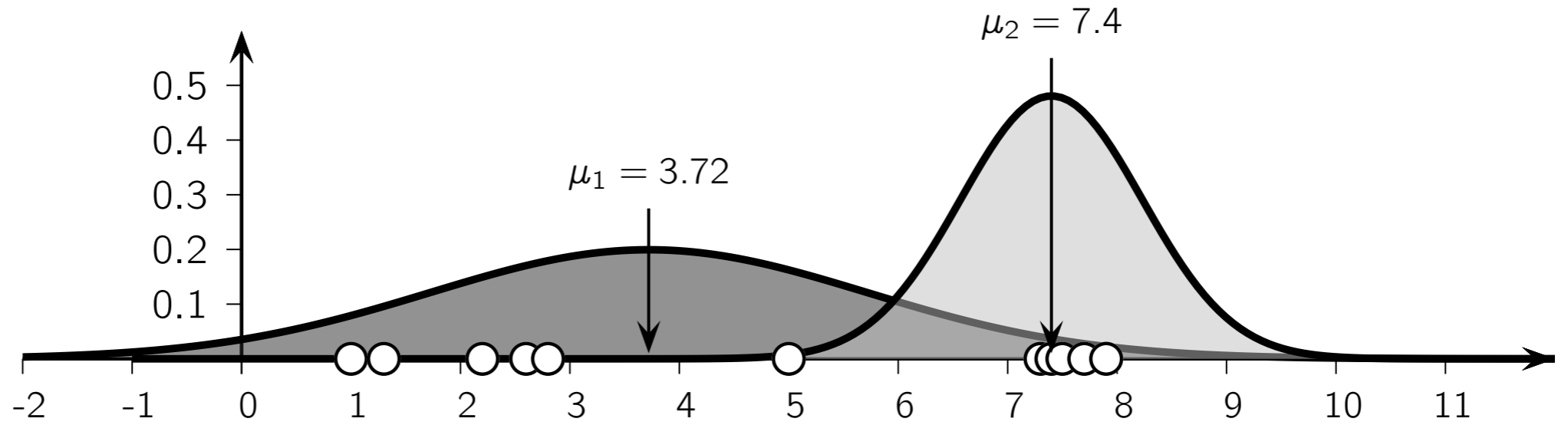
Example

Initialization



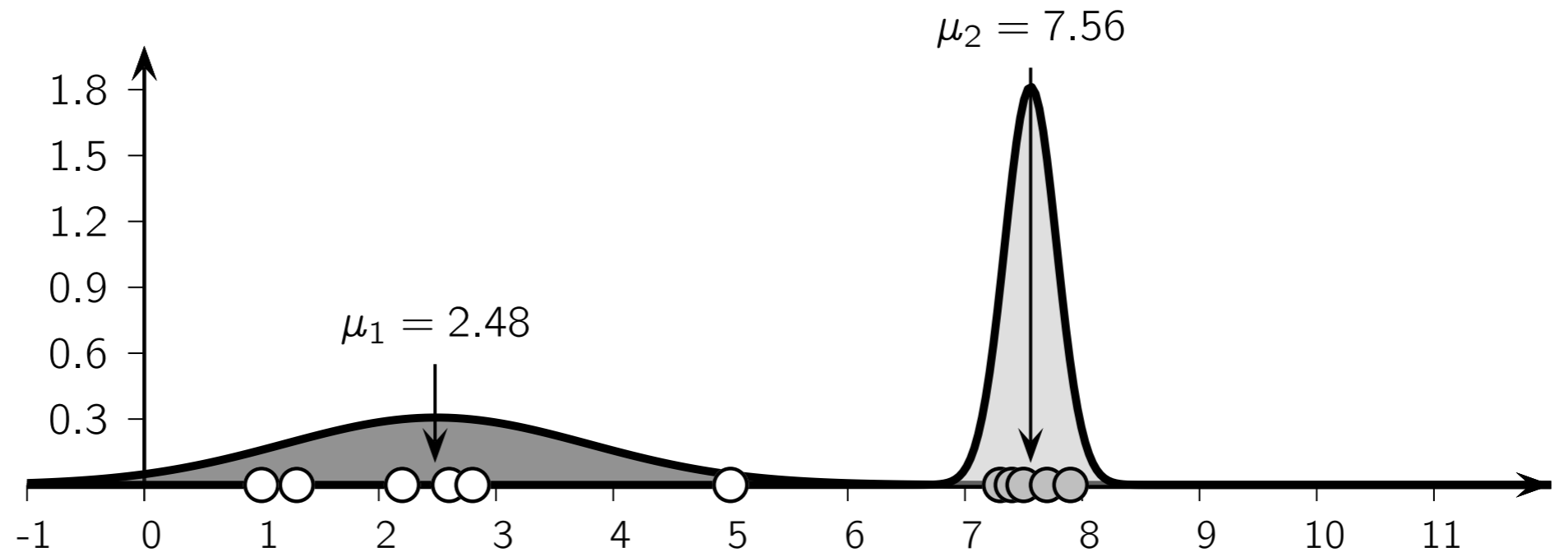
(a) Initialization: $t = 0$

Iteration 1



(b) Iteration: $t = 1$

Iteration 5



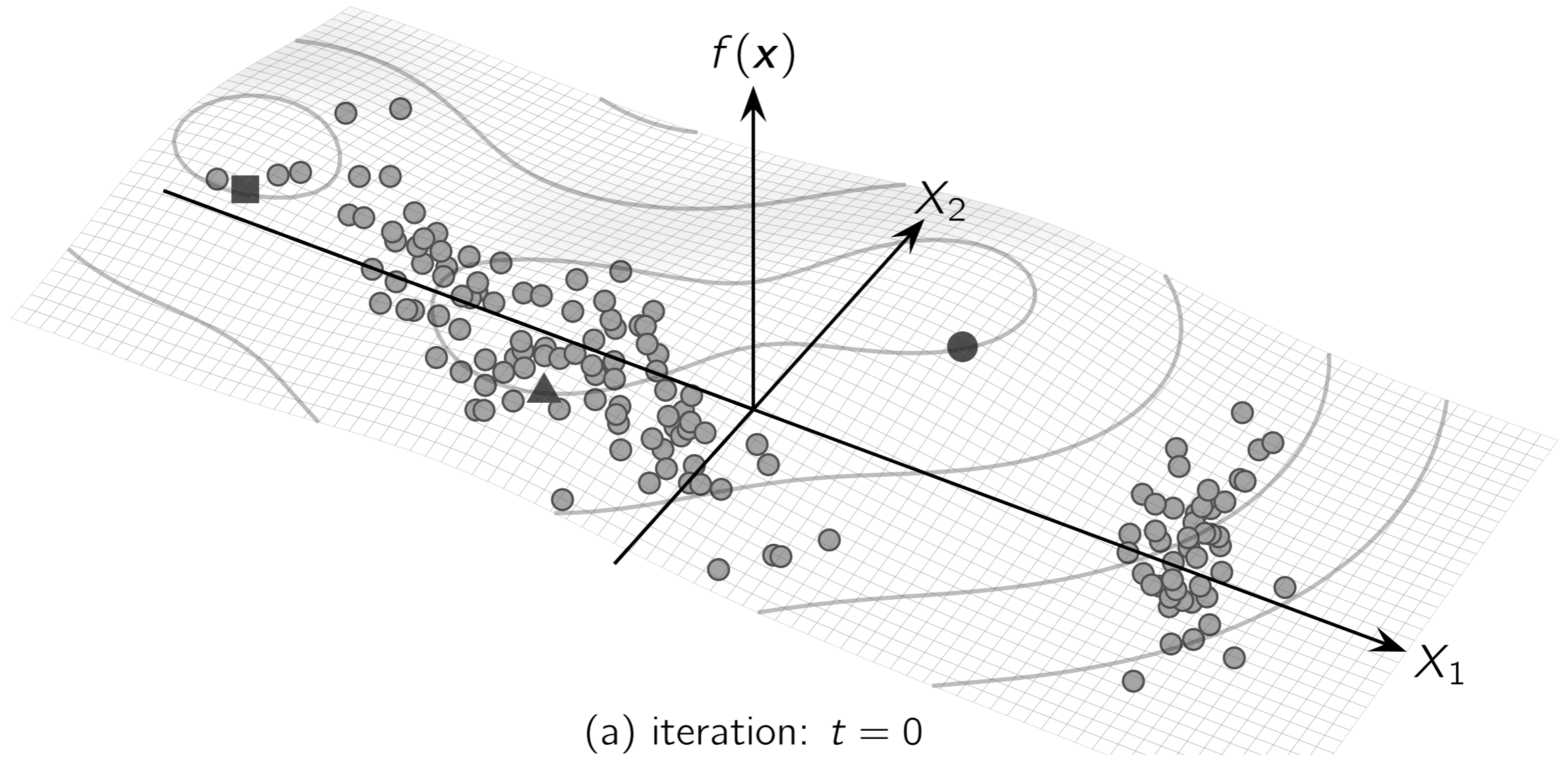
(c) Iteration: $t = 5$ (converged)

EM in d -dimensions

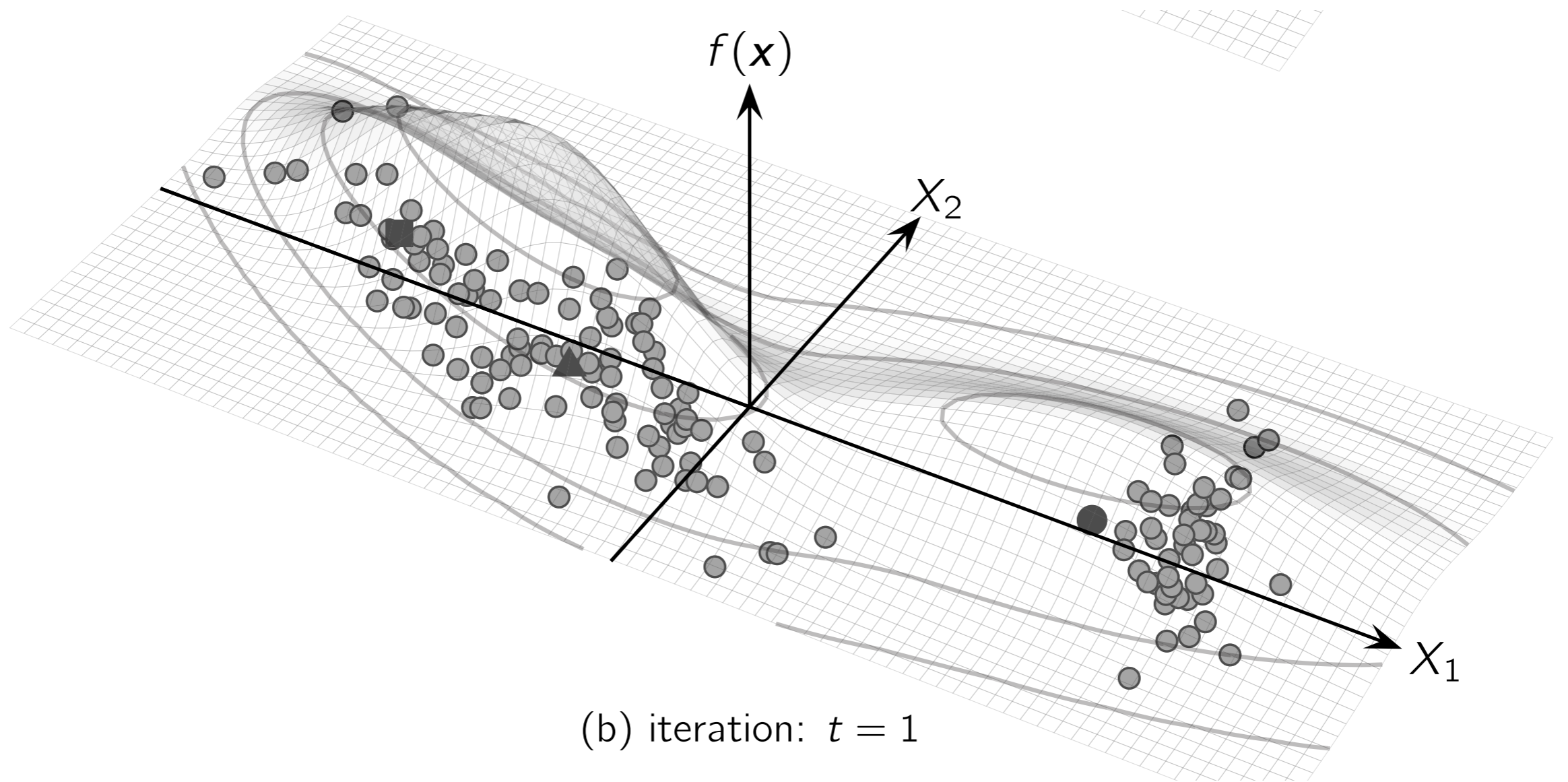
- The covariance matrix requires $d(d+1)/2$ parameters to be estimated
 - Often all dimensions are assumed independent, yielding d parameters
- The expectation step is as in 1-D
- The mean and prior $P(C_i)$ are estimated as in 1-D
- The variance of cluster i in dimension a is

$$(\sigma_{aa}^i)^2 = \frac{\sum_{j=1}^n w_{ij} (\mathbf{x}_{ja} - \mu_{ia})^2}{\sum_{j=1}^n w_{ij}}$$

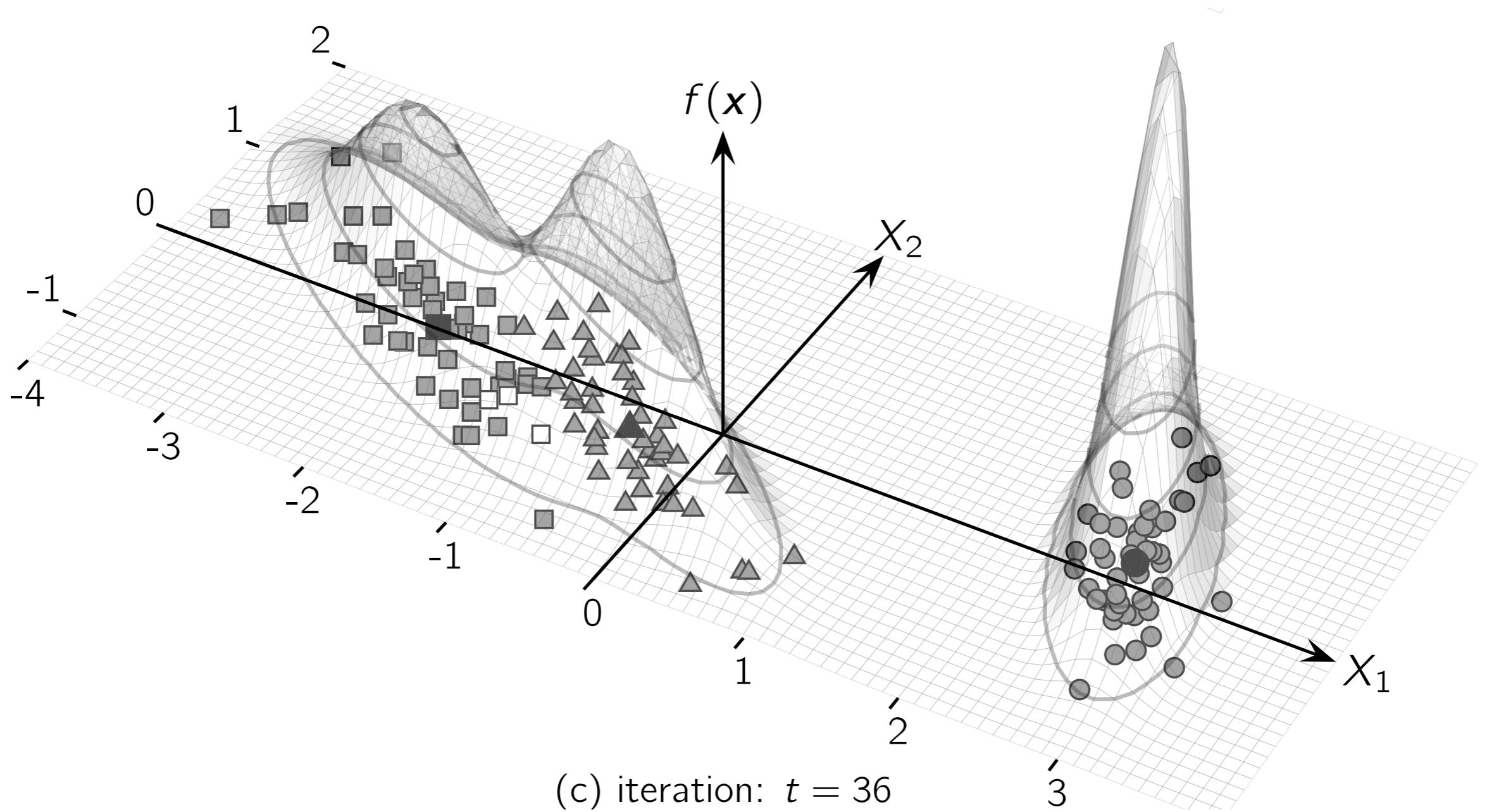
Example



Example



Example



k -means as EM

- The iterative k -means algorithm can be seen as a special case of EM algorithm using different cluster density function
 - $P(\mathbf{x}_j | C_i) = 1$ iff centroid i is the closest to point \mathbf{x}_j
- The posterior probability is then
 - $P(C_i | \mathbf{x}_j) = 1$ iff point \mathbf{x}_j belongs to cluster i
- The parameters are the centroids and $P(C_i)$
 - The covariance matrix can be ignored

How to select k

- Both k -means and EM require user to define k **before** the algorithm is run
 - But what if we don't know the k ?
- The larger the k ,
 - the smaller the error
 - the more complex the model
 - the higher the risk for over-fitting

Cross-validation

- As with regression:
 - Hold out some random points (test set)
 - Run clustering on the remaining points (training set)
 - Compute the error with test set included
 - Re-iterate with different values of k and select the one with least overall error
- Normally N -fold cross validation

AIC and BIC

- Let $\ln(L)$ be the maximized log-likelihood of the clustering (obtained e.g. via EM algorithm)
- Let $p(k)$ be the number of parameters we need for k clusters
 - For Gaussian with independent dimensions, $p(k) = k(d+2)$
 - k clusters, d variances, mean, and mixture parameter $P(C_i)$
- **Idea:** We need to pay for each new parameter in our model
- In *Akaike Information Criterion* (AIC) we select k that minimizes $AIC = 2p(k) - 2\ln(L)$
- In *Bayesian Information Criterion* (BIC) we select k that minimizes $BIC = p(k)\ln(n) - 2\ln(L)$