

# Unsupervised Learning

## Introduction to clustering

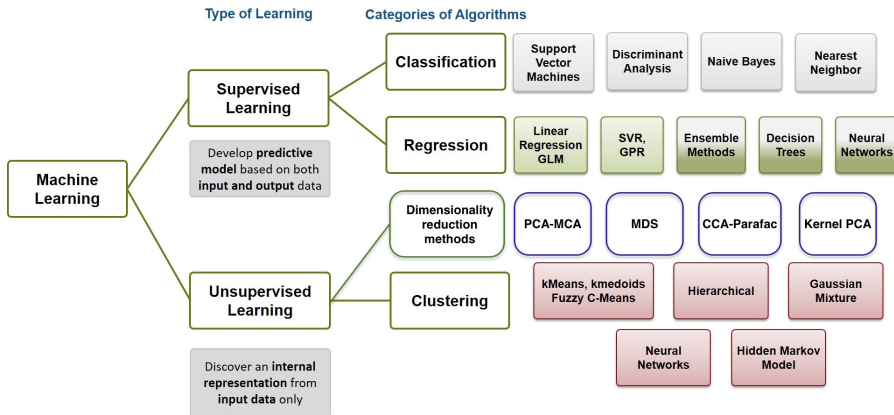
MAP 573, 2019 – Julien Chiquet

cole Polytechnique, Autumn semester, 2019

<https://github.com/jchiquet/CourseUnsupervisedLearningX>



# Machine Learning



# Supervised vs Unsupervised Learning

## Supervised Learning

- Training data  $\mathcal{D}_n = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$ ,  $X_i \sim^{\text{i.i.d}} \mathbb{P}$
- Construct a predictor  $\hat{f} : \mathcal{X} \rightarrow \mathcal{Y}$  using  $\mathcal{D}_n$
- Loss  $\ell(y, f(x))$  measures how well  $f(x)$  predicts  $y$
- Aim: minimize the generalization error
- Task: Regression, Classification

↪ The goal is clear: predict  $y$  based on  $x$  (regression, classification)

## Unsupervised Learning

- Training data  $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$
- Loss? , Aim?
- Task: Dimension reduction, Clustering

↪ The goal is less well defined, and *validation* is questionable

# Outline

## ① Clustering: introduction

- Motivating example

- Generalities

- Vocabulary

## ② Distance-based methods

- The K-means algorithm

- Hierarchical Agglomerative Clustering

## ③ Model-based approach

- Mixture models

- Expectation-Maximization algorithm

# Outline

## ① Clustering: introduction

Motivating example

Generalities

Vocabulary

## ② Distance-based methods

## ③ Model-based approach

# Packages required for reproducing the slides

```
library(tidyverse)  # opinionated collection of packages for data manipulation
library(corrplot)   # fancy plots of matrices as images
library(GGally)     # extension to ggplot vizualization system
library(ggfortify)  # extension to ggplot vizualization system
library(mclust)     # Gaussian mixture models
library(aricode)    # fast computation of clustering measures
library(animation)  # kmeans animation slides
# color and plots themes
library(RColorBrewer)
pal <- brewer.pal(10, "Set3")
theme_set(theme_bw())
```

# Outline

## ① Clustering: introduction

**Motivating example**

Generalities

Vocabulary

## ② Distance-based methods

## ③ Model-based approach

# Companion data set

## Morphological Measurements on Leptograpsus Crabs

### Description

The crabs data frame has 200 rows and 8 columns, describing 5 morphological measurements on 50 crabs each of two colour forms and both sexes, of the species *Leptograpsus variegatus* collected at Fremantle, W. Australia.

```
crabs <- MASS::crabs %>% select(-index) %>%  
  rename(sex = sex,  
         species = sp,  
         frontal_lob = FL,  
         rear_width = RW,  
         carapace_length = CL,  
         carapace_width = CW,  
         body_depth = BD)  
crabs %>% select(sex, species) %>% summary() %>% knitr::kable("latex")
```

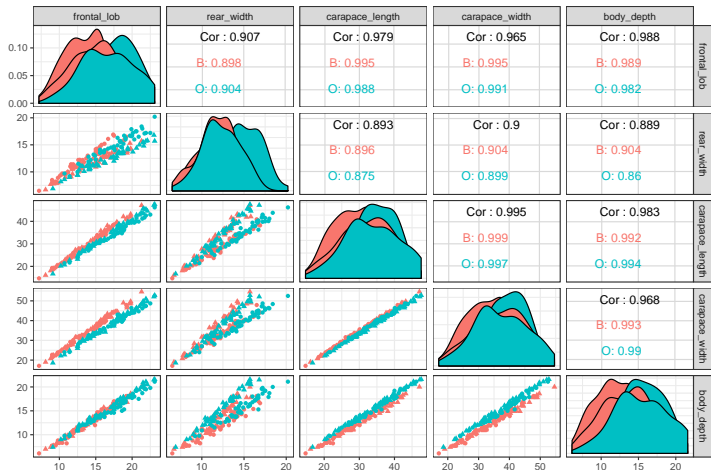
	sex	species
	F:100	B:100
	M:100	O:100



# Companion data set II

## Pairs plot of attributes

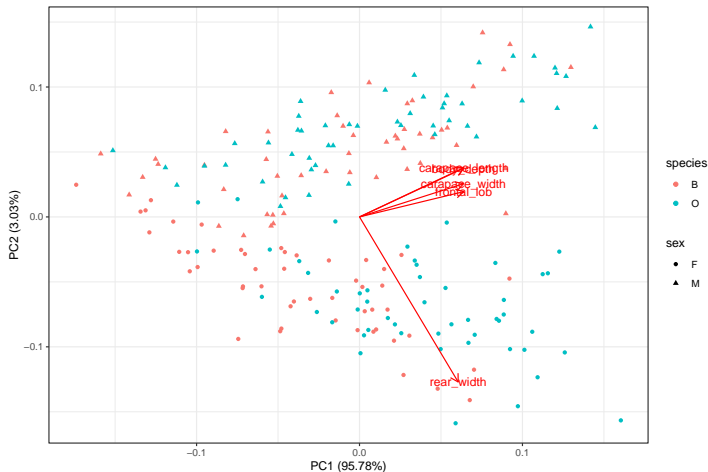
```
ggpairs(crabs, columns = 3:7, aes(colour = species, shape = sex))
```



# Companion data set III

## PCA on the attributes

```
prcomp(select(crabs, -species, -sex), scale. = TRUE) %>%  
  autoplot(loadings = TRUE, loadings.label = TRUE,  
           data = crabs, colour = 'species', shape = 'sex')
```



# Remove size effect I

Carried by the 1st principal component

PCA is solved by SVD

$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^{\top}.$$

We remove the best rank-1 approximation of  $\mathbf{X}$  to remove the *size effect*, carried by the first axis, that is,

$$\tilde{\mathbf{X}}^{(1)} = \mathbf{U}_{\bullet 1} d_{11} \mathbf{v}_{\bullet 1}^{\top}.$$

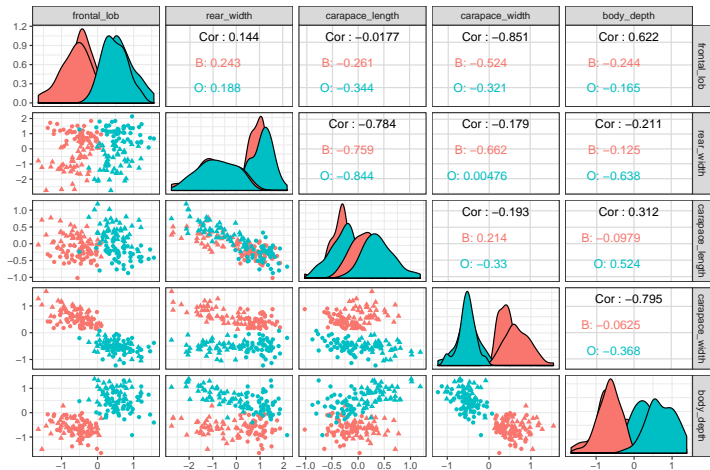
```
attributes <- select(crabs, -sex, -species)
SVD <- svd(attributes)
attributes_rank1 <- tcrossprod(SVD$u[, 1] * SVD$d[1], SVD$v[, 1])
crabs_corrected <- crabs
crabs_corrected[, 3:7] <- attributes - attributes_rank1
```

↪ Axis 1 explains a latent effect, here the size in the case at hand, common to all attributes.

# Remove size effect II

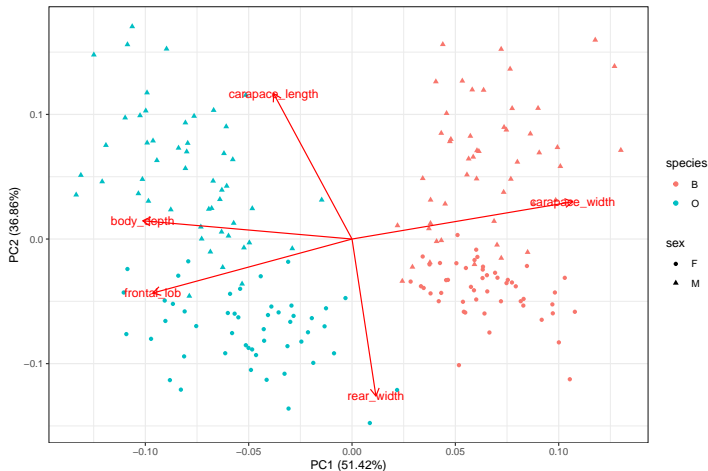
Carried by the 1st principal component

```
ggpairs(crabs_corrected, columns = 3:7, aes(colour = species, shape = sex))
```



# PCA on corrected data

```
prcomp(select(crabs_corrected, -species, -sex), scale. = TRUE) %>%  
  autoplot(loadings = TRUE, loadings.label = TRUE,  
    data = crabs_corrected, colour = 'species', shape = 'sex')
```



# Questions

- ① Could we automatically identify some grouping (**clustering**) between samples ?
- ② Would this clustering correspond to some known labels (sex, species)?
- ③ Does it matter?

# Outline

## ① Clustering: introduction

Motivating example

**Generalities**

Vocabulary

## ② Distance-based methods

## ③ Model-based approach

# Clustering: general goals

**Objective:** construct a map  $f$  from  $\mathcal{D}$  to  $\{1, \dots, K\}$  where  $K$  is a fixed number of clusters.

Careful! classification  $\neq$  clustering

- Classification presupposes the existence of classes
- Clustering labels only elements of the dataset
  - $\rightsquigarrow$  no ground truth (no given labels)
  - $\rightsquigarrow$  discovers a structure "natural" to the data
  - $\rightsquigarrow$  not necessarily related to a known classification

## Motivations

- describe large masses of data in a simplified way,
- structure a set of knowledge,
- reveal structures, hidden causes,
- use of the groups in further processing,
- ...



# Clustering: challenges

## Clustering quality

No obvious measure to define the **quality** of the clusters. Ideas:

- **Inner** homogeneity: samples in the same group should be similar
- **Outer** inhomogeneity: samples in different groups should be different

## Number of clusters

Choice of the number of clusters  $K$  often complex

- No ground truth in unsupervised learning!
- Several solutions might be equally good

## Two general approaches

- **distance-based**: require a distance/dissimilarity between  $\{\mathbf{x}_i\}$
- **model-based**: require assumptions on the distribution  $\mathbb{P}$

# Outline

## ① Clustering: introduction

Motivating example

Generalities

**Vocabulary**

## ② Distance-based methods

## ③ Model-based approach

# Dissimilarity and Distance

*Clustering requires a measure of resemblance between object*

Definition ((dis)similarity)

Similarity (*resp.* Dissimilarity) measures the resemblance (*resp.* discrepancy) between objects based on several features.

For instance, two objects are similar if

- they share a certain feature
- their features are close according to a measure of proximity

Definition (distance/metric)

Dissimilarity can be measured by distances, *i.e.* a function  $d_{ij}$  between pairs in  $\{\mathbf{x}_i\}$  s.t.

- $d_{ij} \geq 0$ ,
- $d_{ij} = 0 \Leftrightarrow \mathbf{x}_i = \mathbf{x}_j$ ,
- $d_{ij} = d_{ji}$ ,
- $d_{ik} \leq d_{ij} + d_{jk}$ .

# Dissimilarity and Distance

*Clustering requires a measure of resemblance between object*

## Definition ((dis)similarity)

Similarity (*resp.* Dissimilarity) measures the resemblance (*resp.* discrepancy) between objects based on several features.

For instance, two objects are similar if

- they share a certain feature
- their features are close according to a measure of proximity

## Definition (distance/metric)

Dissimilarity can be measured by distances, *i.e.* a function  $d_{ij}$  between pairs in  $\{\mathbf{x}_i\}$  s.t.

- $d_{ij} \geq 0$ ,
- $d_{ij} = 0 \Leftrightarrow \mathbf{x}_i = \mathbf{x}_j$ ,
- $d_{ij} = d_{ji}$ ,
- $d_{ik} \leq d_{ij} + d_{jk}$ .

# Classification structures: Partition

*Clustering leads to a grouping (or classification) of individuals into homogeneous classes*

We consider two structures to describe this classification:

- partitions and
- hierarchies.

## Definition (Partition)

A partition  $\mathcal{P}$  is a decomposition  $\mathcal{P} = \{P_1, \dots, P_K\}$  of a finite ensemble  $\Omega$  such that

- $P_k \cap P_{k'} = \emptyset$  for any  $k \neq k'$
- $\bigcup_k P_k = \Omega$

In a set  $\Omega = (\mathbf{x}_1, \dots, \mathbf{x}_n)$  partitioned into  $K$  classes, each element of the set belongs to a class and only one.

# Classification structures: Hierarchy

## Definition (Hierarchy)

A hierarchy  $\mathcal{H}$  is a non empty subset of a finite ensemble  $\Omega$  such that

- $\Omega \in \mathcal{H}$ ,
- $\forall \mathbf{x} \in \Omega, \{\mathbf{x}\} \in \mathcal{H}$ ,
- $\forall H, H' \in \mathcal{H}$ , then either  $H \cap H' = \emptyset$ ,  $H \subset H'$  or  $H' \subset H$ .

## Definition (Index of a Hierarchy)

The index is a function  $i: \mathcal{H} \rightarrow \mathbb{R}_+$  such that

- if  $H \subset H'$  then  $i(H) < i(H')$ ;
- if  $\mathbf{x} \in \Omega$  then  $i(\mathbf{x}) = 0$ .

## Properties (Partition and Hierarchy)

- *Each level of an indexed hierarchy is a partition;*
- *$\{\Omega, P_1, \dots, P_K, \mathbf{x}_1, \dots, \mathbf{x}_n\}$  is a hierarchy.*

# Classification structures: Hierarchy

## Definition (Hierarchy)

A hierarchy  $\mathcal{H}$  is a non empty subset of a finite ensemble  $\Omega$  such that

- $\Omega \in \mathcal{H}$ ,
- $\forall \mathbf{x} \in \Omega, \{\mathbf{x}\} \in \mathcal{H}$ ,
- $\forall H, H' \in \mathcal{H}$ , then either  $H \cap H' = \emptyset$ ,  $H \subset H'$  or  $H' \subset H$ .

## Definition (Index of a Hierarchy)

The index is a function  $i : \mathcal{H} \rightarrow \mathbb{R}_+$  such that

- if  $H \subset H'$  then  $i(H) < i(H')$ ;
- if  $\mathbf{x} \in \Omega$  then  $i(\mathbf{x}) = 0$ .

## Properties (Partition and Hierarchy)

- *Each level of an indexed hierarchy is a partition;*
- *$\{\Omega, P_1, \dots, P_K, \mathbf{x}_1, \dots, \mathbf{x}_n\}$  is a hierarchy.*

# Classification structures: Hierarchy

## Definition (Hierarchy)

A hierarchy  $\mathcal{H}$  is a non empty subset of a finite ensemble  $\Omega$  such that

- $\Omega \in \mathcal{H}$ ,
- $\forall \mathbf{x} \in \Omega, \{\mathbf{x}\} \in \mathcal{H}$ ,
- $\forall H, H' \in \mathcal{H}$ , then either  $H \cap H' = \emptyset$ ,  $H \subset H'$  or  $H' \subset H$ .

## Definition (Index of a Hierarchy)

The index is a function  $i : \mathcal{H} \rightarrow \mathbb{R}_+$  such that

- if  $H \subset H'$  then  $i(H) < i(H')$ ;
- if  $\mathbf{x} \in \Omega$  then  $i(\mathbf{x}) = 0$ .

## Properties (Partition and Hierarchy)

- *Each level of an indexed hierarchy is a partition;*
- *$\{\Omega, P_1, \dots, P_K, \mathbf{x}_1, \dots, \mathbf{x}_n\}$  is a hierarchy.*



# Clusterings Comparison: Contingency table

## Definition

Consider two clusterings  $U$  and  $V$  of elements in  $\Omega$ , into respectively  $|U|$  and  $|V|$  classes. The  $|U| \times |V|$  contingency matrix stores at position  $(i, j)$  the number of elements that are simultaneously in cluster  $i$  of  $U$  and  $j$  of  $V$ .

$U \backslash V$	$V_1$	$V_2$	$\dots$	$V_{ V }$	Sums
$U_1$	$n_{11}$	$n_{12}$	$\dots$	$n_{1 V }$	$n_{1.}$
$U_2$	$n_{21}$	$n_{22}$	$\dots$	$n_{2 V }$	$n_{2.}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
$U_{ U }$	$n_{ U 1}$	$n_{ U 2}$	$\dots$	$n_{ U  V }$	$n_{ U .}$
Sums	$n_{.1}$	$n_{.2}$	$\dots$	$n_{. V }$	$n_{..} = n$

# Clusterings Comparison: Measures (I)

## Definition (Rand index)

Given a set  $\Omega$  of  $n$  elements and two partitions  $U$  and  $V$  to compare, define the following:

- $a$ , the number of pairs in the same subset in  $U$  and in  $V$
- $b$ , the number of pairs in different subsets in  $U$  and in  $V$

The Rand index,  $RI \in [0, 1]$  is

$$RI = \frac{a + b}{\binom{n}{2}}$$

The Rand index can be viewed as a measure of the percentage of correct decisions:

$$RI = \frac{TP + TN}{\binom{n}{2}},$$

where  $TP, TN$  are true positive and true negative decisions.

## Clusterings Comparison: Measures (II)

The ARI (most popular) is a version of the RI adjusted for chance grouping of element (i.e., the expected similarity of all pair-wise comparisons).

Definition (Adjusted Rand-index)

$$ARI(U, V) = \frac{\sum_{i,j} \binom{n_{ij}}{2} - \left[ \sum_i \binom{n_{i.}}{2} \sum_j \binom{n_{.j}}{2} \right] / \binom{n}{2}}{\frac{1}{2} \left[ \sum_i \binom{n_{i.}}{2} + \sum_j \binom{n_{.j}}{2} \right] - \left[ \sum_i \binom{n_{i.}}{2} \sum_j \binom{n_{.j}}{2} \right] / \binom{n}{2}}$$

Other popular measures:

- *NVI*, the normalized variation information
- *NID*, the normalized information distance
- *NMI*, the normalized mutual information

# Outline

- 1 Clustering: introduction
- 2 Distance-based methods
  - The K-means algorithm
  - Hierarchical Agglomerative Clustering
- 3 Model-based approach

# References



The Elements of Statistical Learning,

T. Hastie, R. Tibshirani, J. Friedman

Chapter: 14 Unsupervised Learning, Section 3: Cluster Analysis

<https://web.stanford.edu/~hastie/ElemStatLearn/>



Classification non-supervisées,

É. Lebarbier, T. Mary-Huard

Chapitre 2 - méthode de partitionnement

<https://www.agroparistech.fr/IMG/pdf/ClassificationNonSupervisee-AgroParisTech.pdf>

# Outline

- 1 Clustering: introduction
- 2 Distance-based methods
  - The K-means algorithm
  - Hierarchical Agglomerative Clustering
- 3 Model-based approach

# K-means heuristic

## Idea

- ① Clustering is defined by a partition in  $K$  classes
- ② Minimize a criteria of clustering quality
- ③ Use Euclidean distances to measure dissimilarity

Criteria: intra-class variance/ Inertia "within"

Intra-class variance measures **inner** homogeneity

$$I_W = \sum_{k=1}^K \sum_{i=1}^n c_{ik} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|_2^2,$$

where

- $\boldsymbol{\mu}_k$  are the centers (prototypes) of classes
- $c_{ik} = \mathbf{1}_{i \in \mathcal{P}_k}$  is a partition matrix

# K-means algorithm

Ideally, one would solve

$$(\hat{\mathbf{c}}, \hat{\boldsymbol{\mu}}) = \arg \min_{(\mathbf{c}, \boldsymbol{\mu})} I_w((\mathbf{c}, \boldsymbol{\mu})), \quad \text{s.t.} \quad \mathbf{c} \text{ is a partition matrix.}$$

This problem is hard to solve but can be optimized locally as follows:

## K-means algorithm (Lloyd's)

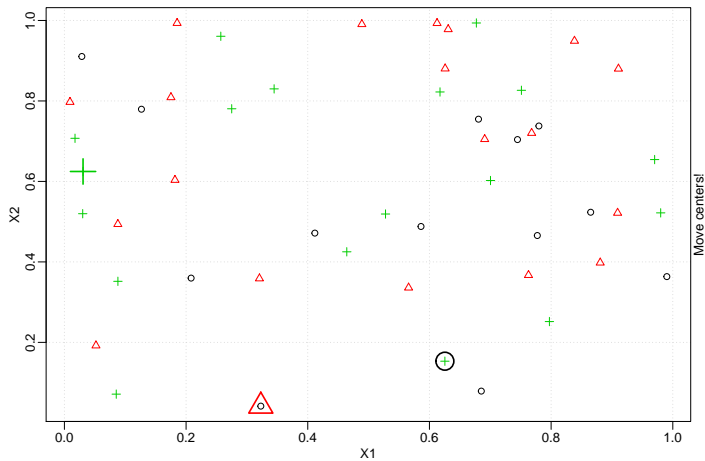
**Initialization** start by a (pseudo) random choice for the centers  $\boldsymbol{\mu}_k$

**Alternate** until convergence

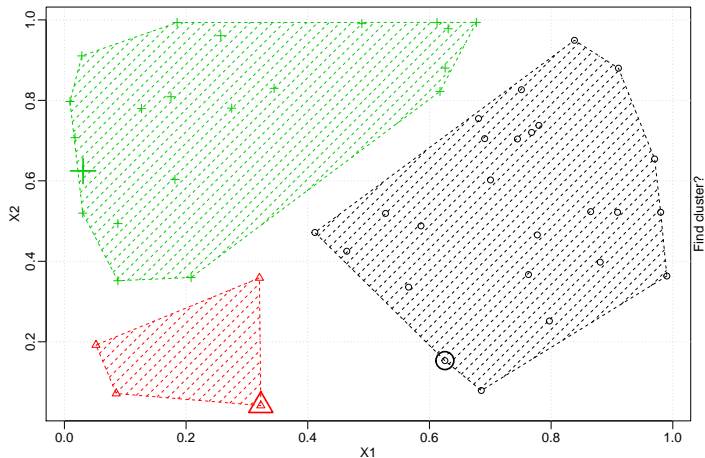
- step 1 given  $\boldsymbol{\mu}$ , chose  $\mathbf{c}$  minimizing  $I_w \equiv$  assign  $\mathbf{x}_i$  to the nearest prototype
- step 2 given  $\mathbf{c}$ , chose  $\boldsymbol{\mu}$  minimizing  $I_w \equiv$  update  $\boldsymbol{\mu}$  by the new means of classes



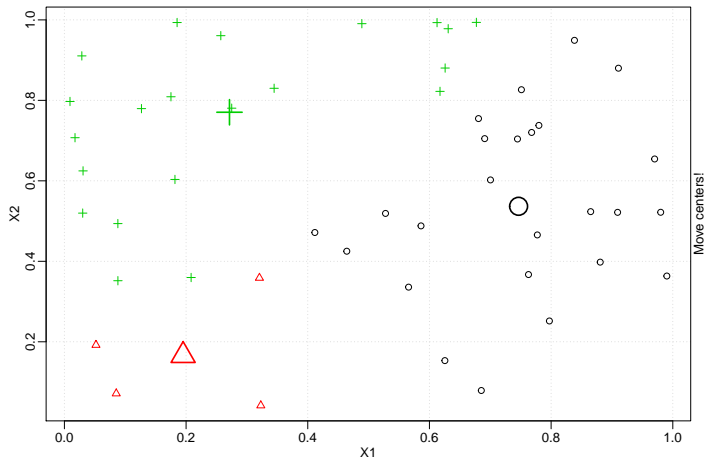
# K-means in action I



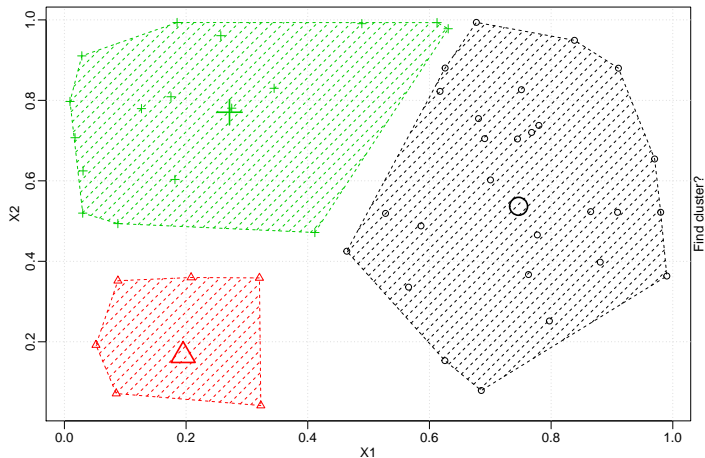
# K-means in action II



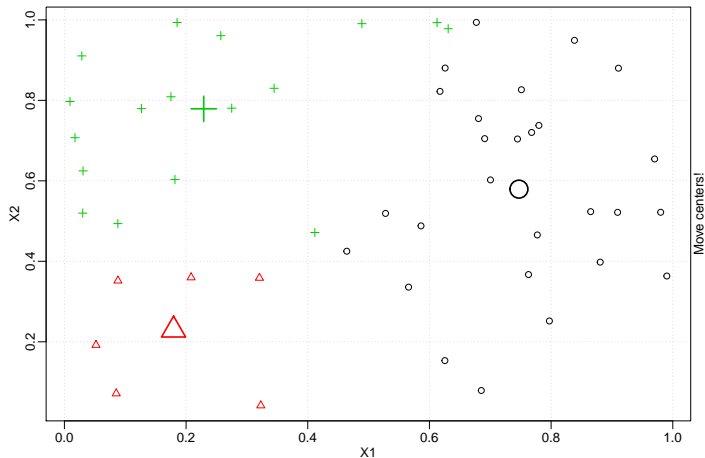
# K-means in action III



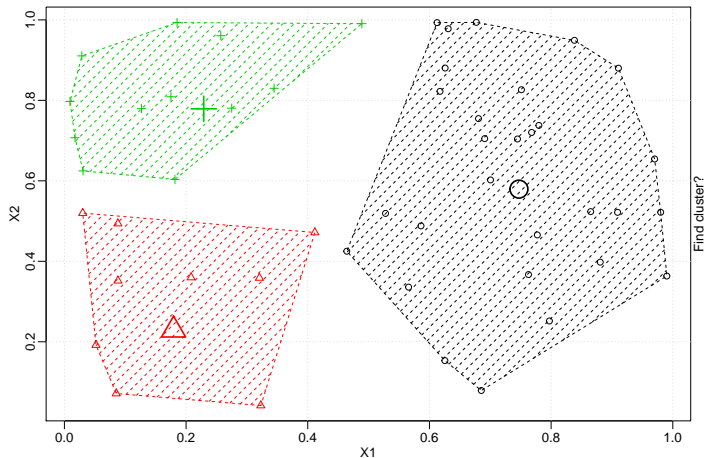
# K-means in action IV



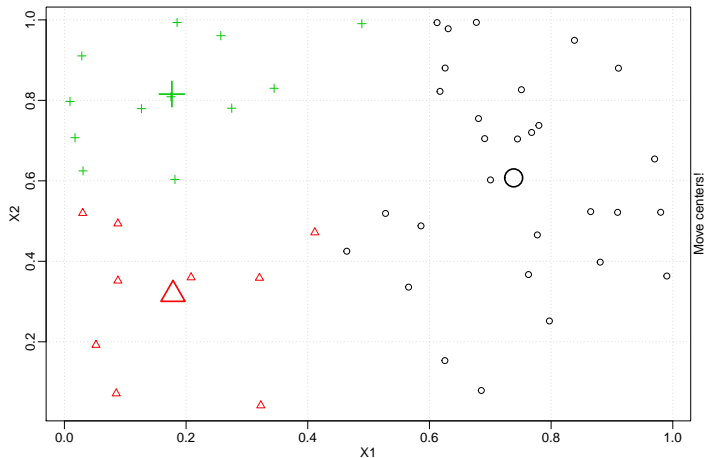
# K-means in action V



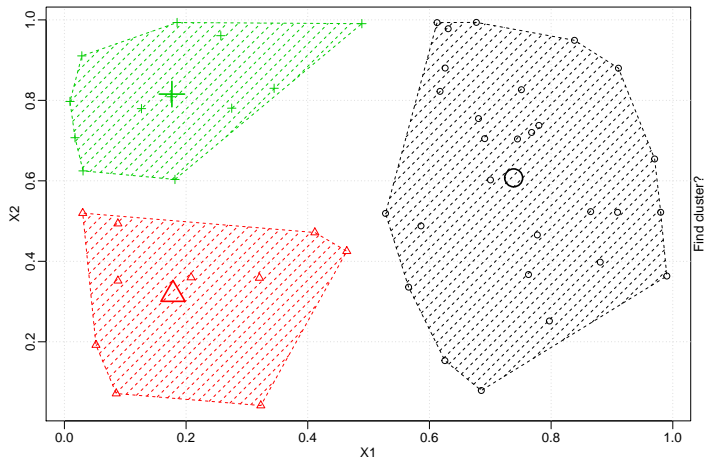
# K-means in action VI



# K-means in action VII

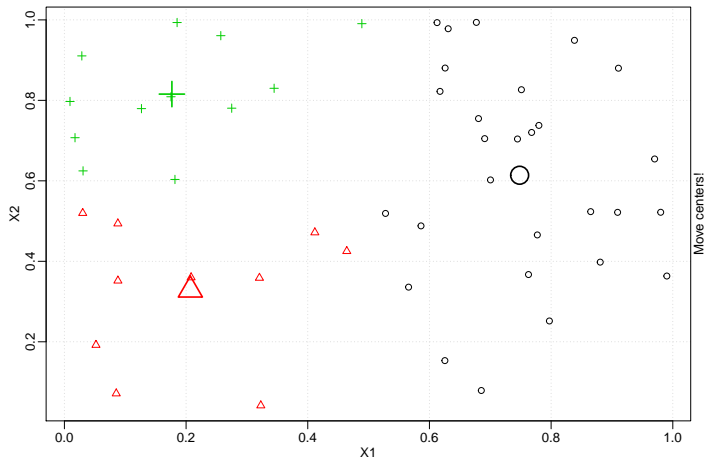


# K-means in action VIII

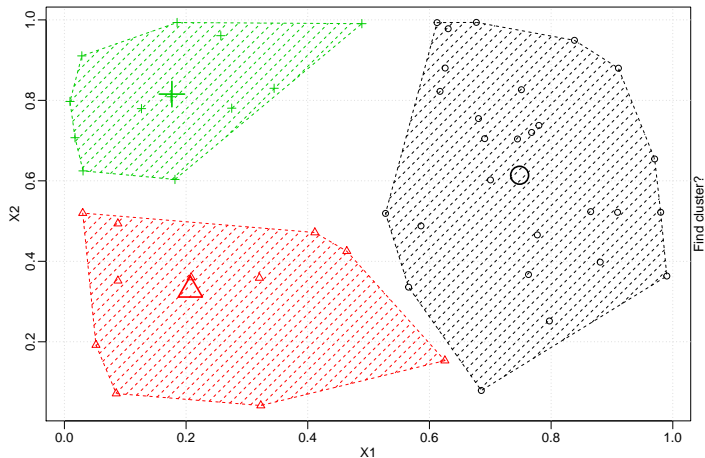




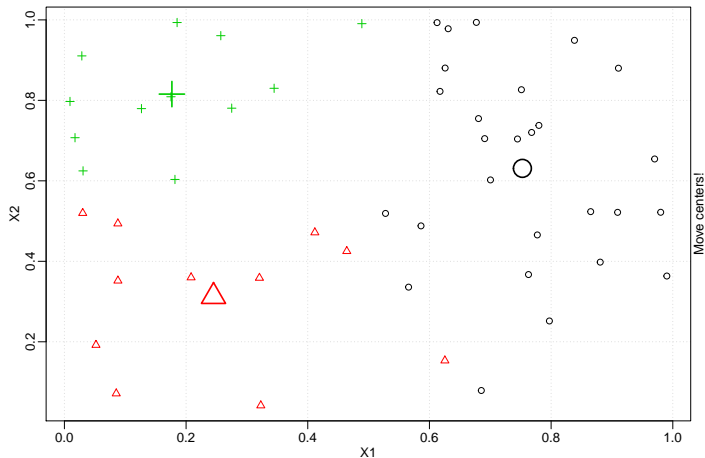
# K-means in action IX



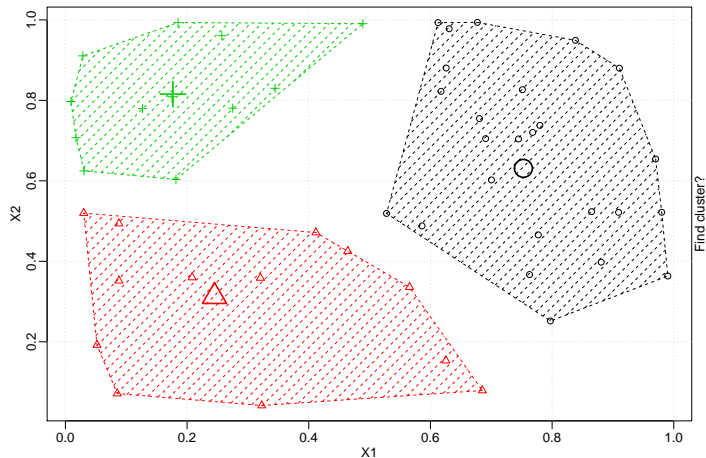
# K-means in action X



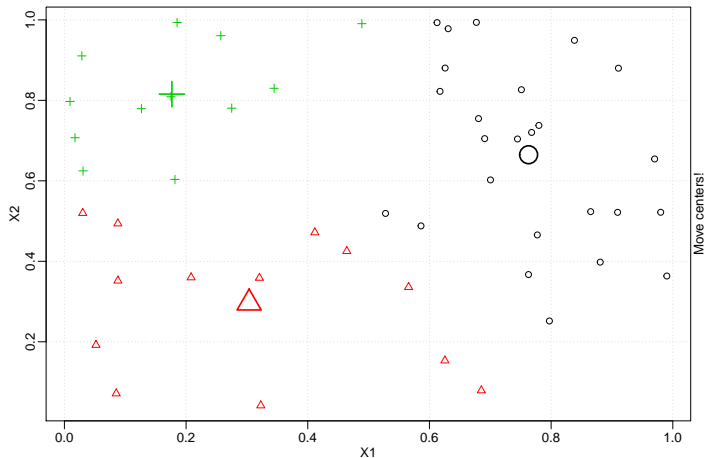
# K-means in action XI



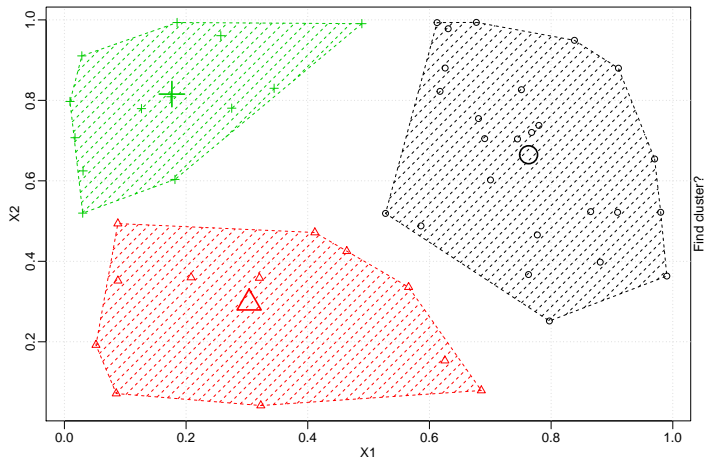
# K-means in action XII



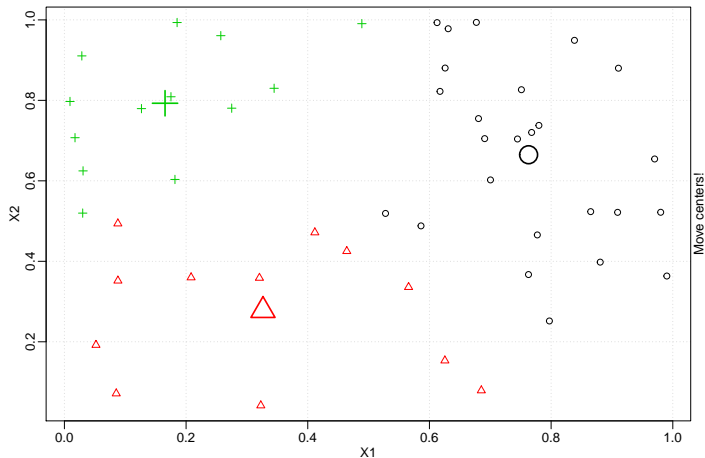
# K-means in action XIII



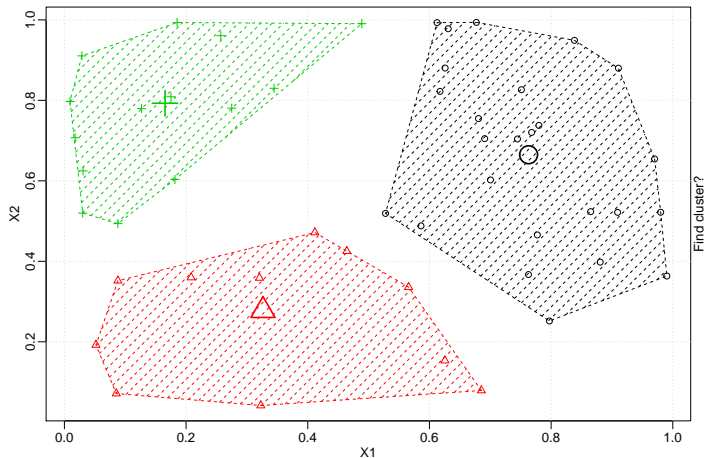
# K-means in action XIV



# K-means in action XV

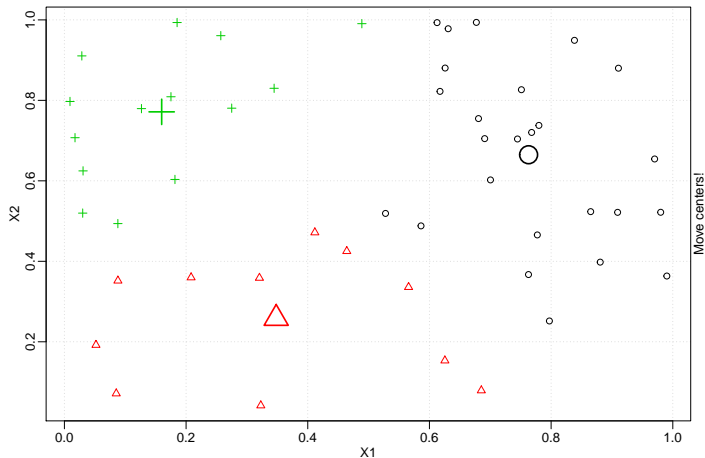


# K-means in action XVI

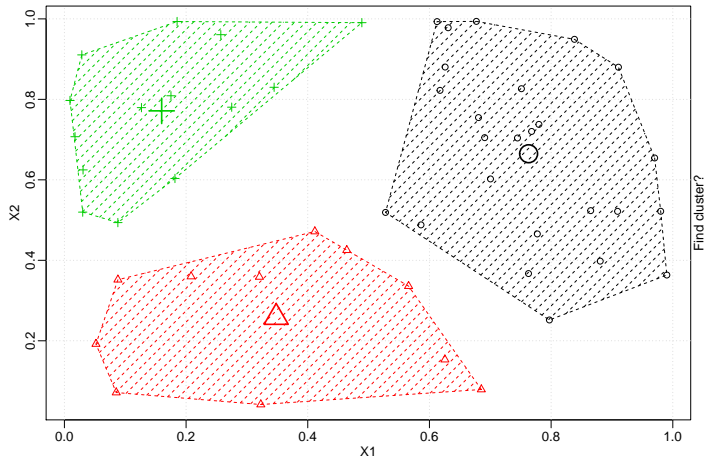




# K-means in action XVII



# K-means in action XVIII



# K-means: properties

## Other schemes

- **McQueen**: modify the mean each time a sample is assigned to a new cluster.
- **Hartigan**: modify the mean by removing the considered sample, assign it to the nearby center and recompute the new mean after assignment.

## Initialization

No guarantee to converge to a global optimum

- Repeat and keep the best result
- k-Mean++: try to take them as separated as possible.

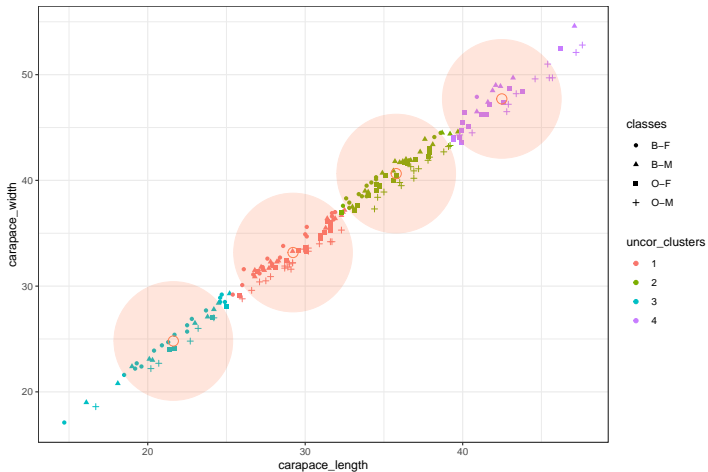
## Complexity

$O(nKT)$  where  $T$  is the number of step in the algorithm.

# K-means in R on uncorrected data set I

```
uncor_kmeans_res <- crabs %>%  
  select(-species, -sex) %>%  
  kmeans(4, nstart = 10)  
uncor_clusters <- as.factor(uncor_kmeans_res$cluster)  
uncor_centers <- as_tibble(uncor_kmeans_res$centers)  
classes <- paste(crabs_corrected$species, crabs_corrected$sex, sep = "-")  
  
crabs %>%  
  ggplot(aes(x = carapace_length, y = carapace_width, color = uncor_clusters)) +  
  geom_point(aes(shape = classes)) +  
  geom_point(data = uncor_centers, color = 'coral', size = 4, pch = 21) +  
  geom_point(data = uncor_centers, color = 'coral', size = 50, alpha = 0.2)
```

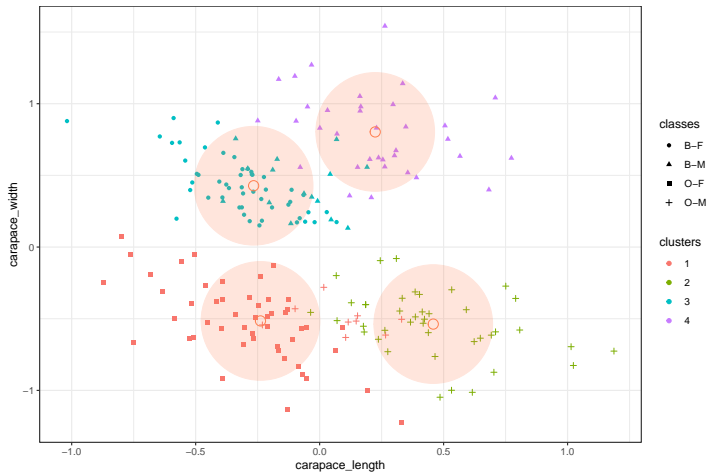
# K-means in R on uncorrected data set II



# K-means in R on corrected crabs data set I

```
kmeans_res <- crabs_corrected %>%  
  select(-species, -sex) %>%  
  kmeans(4, nstart = 10)  
clusters <- as.factor(kmeans_res$cluster)  
centers <- as.tibble(kmeans_res$centers)  
classes <- paste(crabs_corrected$species, crabs_corrected$sex, sep = "-")  
  
crabs_corrected %>%  
  ggplot(aes(x = carapace_length, y = carapace_width, color = clusters)) +  
  geom_point(aes(shape = classes)) +  
  geom_point(data = centers, color = 'coral', size = 4, pch = 21) +  
  geom_point(data = centers, color = 'coral', size = 50, alpha = 0.2)
```

# K-means in R on corrected crabs data set II



# Clustering comparison

```
aricode::ARI(clusters, classes)

## [1] 0.7223637

aricode::ARI(uncor_clusters, classes)

## [1] 0.01573617
```

```
knitr::kable(table(clusters, classes),
caption = "Estimating structure with k-means")
```

**Table:** Estimating structure with k-means

B-F	B-M	O-F	O-M
0	0	50	9
0	0	0	41
50	15	0	0
0	35	0	0



# Outline

- ① Clustering: introduction
- ② Distance-based methods
  - The K-means algorithm
  - Hierarchical Agglomerative Clustering
- ③ Model-based approach

# Agglomerative Clustering: Heuristic

## Idea

- ① Start with small clusters (e.g. one cluster  $\equiv$  one individual)
- ② Merge the most similar clusters sequentially (and greedily)
- ③ Stops when all individuals are in the same groups

## Ingredients

- ① a dissimilarity measure (distance between individuals)
- ② a merging criterion  $\Delta$  (dissimilarity between clusters)

- + Generates a hierarchy of clustering instead of a single partition
- Need to select the number of cluster afterwards

# Agglomerative Clustering: general algorithm

## Algorithm

- 1 Start with  $(\mathcal{C}_k^{(0)}) = (\{\mathbf{x}_i\})$  the collection of all singletons.
- 2 At step  $s$ , we have  $n - s$  clusters  $(\mathcal{C}_k^{(s)})$ :
  - Find the two most similar clusters according to a criterion  $\Delta$ :

$$(k, \ell) = \arg \min_{(k', \ell')} \Delta(\mathcal{C}_{k'}^{(s)}, \mathcal{C}_{\ell'}^{(s)})$$

- Merge  $\mathcal{C}_k^{(s)}$  and  $\mathcal{C}_\ell^{(s)}$  into  $\mathcal{C}_k^{(s+1)}$
  - Update the distances between  $\mathcal{C}_k^{(s+1)}$  and the remaining clusters
- 3 Repeat until there is only one cluster.

## Complexity

- In general  $O(n^3)$
- Can be reduced to  $O(n^2)$  if bounding the number of merges

# Agglomerative Clustering: general algorithm

## Algorithm

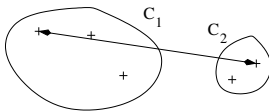
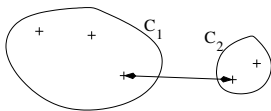
- 1 Start with  $(\mathcal{C}_k^{(0)}) = (\{\mathbf{x}_i\})$  the collection of all singletons.
- 2 At step  $s$ , we have  $n - s$  clusters  $(\mathcal{C}_k^{(s)})$ :
  - Find the two most similar clusters according to a criterion  $\Delta$ :

$$(k, \ell) = \arg \min_{(k', \ell')} \Delta(\mathcal{C}_{k'}^{(s)}, \mathcal{C}_{\ell'}^{(s)})$$

- Merge  $\mathcal{C}_k^{(s)}$  and  $\mathcal{C}_\ell^{(s)}$  into  $\mathcal{C}_k^{(s+1)}$
  - Update the distances between  $\mathcal{C}_k^{(s+1)}$  and the remaining clusters
- 3 Repeat until there is only one cluster.

## Complexity

- In general  $O(n^3)$
- Can be reduced to  $O(n^2)$  if bounding the number of merges



## Merging criterion based on the distance between points

- Single linkage (or minimum linkage):

$$\Delta(\mathcal{C}_k, \mathcal{C}_\ell) = \min_{\mathbf{x}_i \in \mathcal{C}_k, \mathbf{x}_j \in \mathcal{C}_\ell} d(\mathbf{x}_i, \mathbf{x}_j)$$

- Complete linkage (or maximum linkage):

$$\Delta(\mathcal{C}_k, \mathcal{C}_\ell) = \max_{\mathbf{x}_i \in \mathcal{C}_k} \max_{\mathbf{x}_j \in \mathcal{C}_\ell} d(\mathbf{x}_i, \mathbf{x}_j)$$

- Average linkage (or group linkage):

$$\Delta(\mathcal{C}_k, \mathcal{C}_\ell) = \frac{1}{|\mathcal{C}_k||\mathcal{C}_\ell|} \sum_{\mathbf{x}_i \in \mathcal{C}_k} \sum_{\mathbf{x}_j \in \mathcal{C}_\ell} d(\mathbf{x}_i, \mathbf{x}_j)$$

# Ward's criteria

Merging criterion based on distance to the mean

Ward's criterion:

$$\begin{aligned}\Delta(\mathcal{C}_k, \mathcal{C}_\ell) = & \sum_{\mathbf{x}_i \in \mathcal{C}_k} (d^2(\mathbf{x}_i, \boldsymbol{\mu}_{\mathcal{C}_k \cup \mathcal{C}_\ell}) - d^2(\mathbf{x}_i, \boldsymbol{\mu}_{\mathcal{C}_k})) \\ & + \sum_{\mathbf{x}_j \in \mathcal{C}_\ell} (d^2(\mathbf{x}_j, \boldsymbol{\mu}_{\mathcal{C}_k \cup \mathcal{C}_\ell}) - d^2(\mathbf{x}_j, \boldsymbol{\mu}_{\mathcal{C}_\ell}))\end{aligned}$$

Inertia Intra-cluster

If  $d$  is the Euclidean distance, then

$$\Delta(\mathcal{C}_k, \mathcal{C}_\ell) = \frac{2|\mathcal{C}_k||\mathcal{C}_\ell|}{|\mathcal{C}_k| + |\mathcal{C}_\ell|} d^2(\boldsymbol{\mu}_{\mathcal{C}_k}, \boldsymbol{\mu}_{\mathcal{C}_\ell})$$

## Ward's criteria: details

Recall that the intra-class inertia measures the homogeneity of the size-K clustering

$$I_W = \sum_{k=1}^K \sum_{\mathbf{x}_i \in \mathcal{C}_k} \|\mathbf{x}_i - \boldsymbol{\mu}_{\mathcal{C}_k}\|_2^2$$

Consider the following two partitions

- $\mathcal{P} = (\mathcal{C}_1, \dots, \mathcal{C}_K)$  at one level of the hierarchy  $\Omega$
- $\mathcal{P}'$  is  $\mathcal{P}$  once  $\mathcal{C}_k, \mathcal{C}_\ell$  merged

Then

$$I_W(\mathcal{P}) - I_W(\mathcal{P}') = \frac{|\mathcal{C}_k||\mathcal{C}_\ell|}{|\mathcal{C}_k| + |\mathcal{C}_\ell|} d^2(\boldsymbol{\mu}_{\mathcal{C}_k}, \boldsymbol{\mu}_{\mathcal{C}_\ell}) = \frac{1}{2} \Delta(\mathcal{C}_k, \mathcal{C}_\ell).$$

- ↪ At each step, Ward limits the increase of the intra class variance
- ↪ Defines an indexed hierarchy (height of the dendrogram)
- ↪ Same criteria as in the K-means algorithm

## Ward's criteria: details

Recall that the intra-class inertia measures the homogeneity of the size-K clustering

$$I_W = \sum_{k=1}^K \sum_{\mathbf{x}_i \in \mathcal{C}_k} \|\mathbf{x}_i - \boldsymbol{\mu}_{\mathcal{C}_k}\|_2^2$$

Consider the following two partitions

- $\mathcal{P} = (\mathcal{C}_1, \dots, \mathcal{C}_K)$  at one level of the hierarchy  $\Omega$
- $\mathcal{P}'$  is  $\mathcal{P}$  once  $\mathcal{C}_k, \mathcal{C}_\ell$  merged

Then

$$I_W(\mathcal{P}) - I_W(\mathcal{P}') = \frac{|\mathcal{C}_k||\mathcal{C}_\ell|}{|\mathcal{C}_k| + |\mathcal{C}_\ell|} d^2(\boldsymbol{\mu}_{\mathcal{C}_k}, \boldsymbol{\mu}_{\mathcal{C}_\ell}) = \frac{1}{2} \Delta(\mathcal{C}_k, \mathcal{C}_\ell).$$

- ↪ At each step, Ward limits the increase of the intra class variance
- ↪ Defines an indexed hierarchy (height of the dendrogram)
- ↪ Same criteria as in the K-means algorithm



## Ward's criteria: details

Recall that the intra-class inertia measures the homogeneity of th size-K clustering

$$I_W = \sum_{k=1}^K \sum_{\mathbf{x}_i \in \mathcal{C}_k} \|\mathbf{x}_i - \boldsymbol{\mu}_{\mathcal{C}_k}\|_2^2$$

Consider the following two partitions

- $\mathcal{P} = (\mathcal{C}_1, \dots, \mathcal{C}_K)$  at one level of the hierarchy  $\Omega$
- $\mathcal{P}'$  is  $\mathcal{P}$  once  $\mathcal{C}_k, \mathcal{C}_\ell$  merged

Then

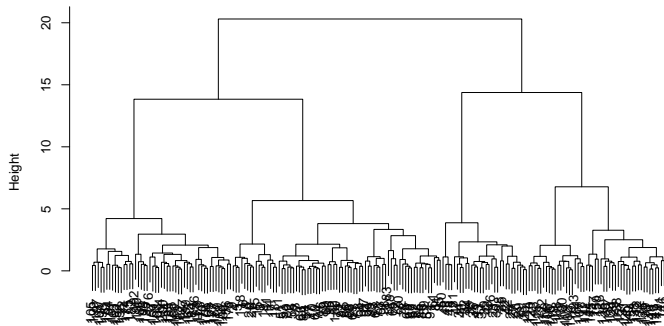
$$I_W(\mathcal{P}) - I_W(\mathcal{P}') = \frac{|\mathcal{C}_k||\mathcal{C}_\ell|}{|\mathcal{C}_k| + |\mathcal{C}_\ell|} d^2(\boldsymbol{\mu}_{\mathcal{C}_k}, \boldsymbol{\mu}_{\mathcal{C}_\ell}) = \frac{1}{2} \Delta(\mathcal{C}_k, \mathcal{C}_\ell).$$

- ↪ At each step, Ward limits the increase of the intra class variance
- ↪ Defines an indexed hierarchy (height of the dendrogram)
- ↪ Same criteria as in the K-means algorithm

# Ward agglomerative clustering in R

```
Ward <- crabs_corrected %>%  
  select(-sex, -species) %>%  
  dist(method = "euclidean") %>%  
  hclust(method = "ward.D2")  
plot(Ward)
```

Cluster Dendrogram



hclust (\*, "ward.D2")

# Ward agglomerative clustering in R: comparison I

Compare with out reference classification and k-means

```
aricode::ARI(cutree(Ward, 4), classes)
```

```
## [1] 0.6829729
```

```
aricode::ARI(cutree(Ward, 4), clusters)
```

```
## [1] 0.7999974
```

```
knitr::kable(table(clusters, cutree(Ward,4)),  
caption = "k-means vs Ward")
```

Table: k-means vs Ward

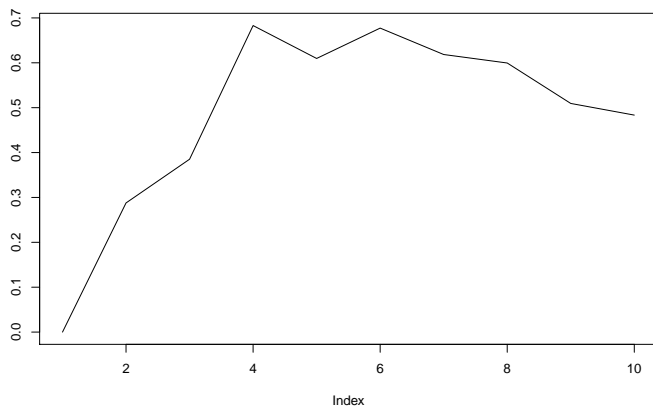
1	2	3	4
2	0	49	8
1	0	0	40
65	0	0	0
5	30	0	0

# Ward agglomerative clustering in R: comparison II

Optimize over a range of values

```
Ward %>% cutree(k = 1:10) %>% as.data.frame() %>% as.list() %>%  
  sapply(aricode::ARI, classes) %>% plot(type = "l")
```

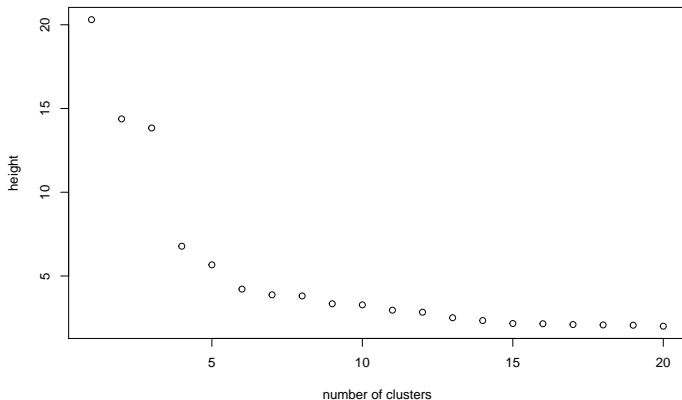
# Ward agglomerative clustering in R: comparison III



Look at Ward intra-class variance

# Ward agglomerative clustering in R: comparison IV

```
plot(rev(Ward$height)[1:20], xlab = "number of clusters", ylab = "height")
```

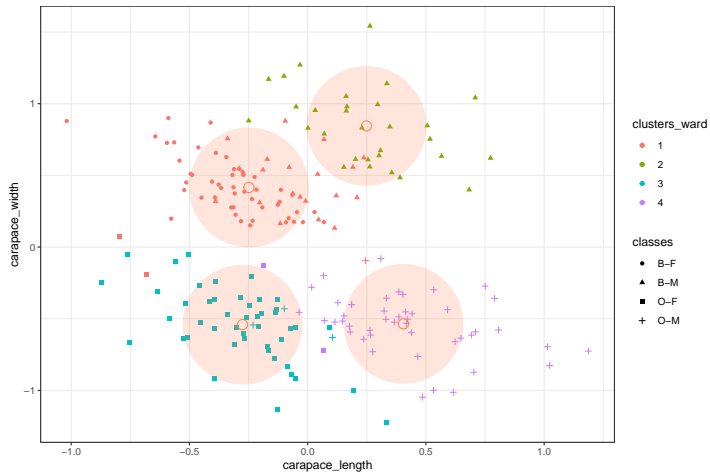


# Ward agglomerative clustering in R: projection I

```
clusters_ward <- as.factor(cutree(Ward, 4))
centers_ward  <- select(crabs_corrected, -sex, -species) %>%
  aggregate(list(cutree(Ward, 4)), mean) %>% as_tibble() %>% select(-Group.1)

crabs_corrected %>%
  ggplot(aes(x = carapace_length, y = carapace_width, color = clusters_ward)) +
  geom_point(aes(shape = classes)) +
  geom_point(data = centers_ward, color = 'coral', size = 4, pch = 21) +
  geom_point(data = centers_ward, color = 'coral', size = 50, alpha = 0.2)
```

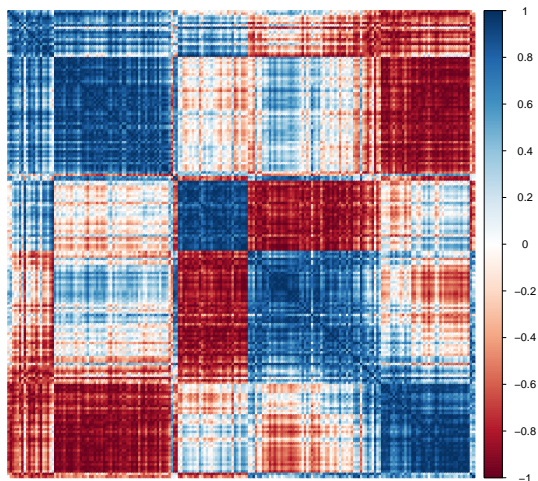
# Ward agglomerative clustering in R: projection II





# Reordered correlation matrix between individuals

```
C <- cor(t(select(crabs_corrected, -sex, -species)))  
C <- C[order(clusters_ward),order(clusters_ward)]  
corrplot(C, method = "color", tl.pos = "n")
```



# Outline

- ① Clustering: introduction
- ② Distance-based methods
- ③ Model-based approach
  - Mixture models
  - Expectation-Maximization algorithm

# References



Pattern recognition and machine learning,  
Christopher Bishop

Chapter 9: Mixture Models and EM

<http://users.isr.ist.utl.pt/~wurmd/Livros/school/>



Models with Hidden Structure with Applications in Biology and  
Genomics,

Stéphane Robin

Master MathSV Course

[https:](https://www6.inra.fr/mia-paris/content/download/4587/42934/version/1/file/ModelsHiddenStruct-Biology.pdf)

[//www6.inra.fr/mia-paris/content/download/4587/42934/version/1/file/ModelsHiddenStruct-Biology.pdf](https://www6.inra.fr/mia-paris/content/download/4587/42934/version/1/file/ModelsHiddenStruct-Biology.pdf)



Classification non-supervisées,

É. Lebarbier, T. Mary-Huard

Chapitre 3 - méthode probabiliste: le modèle de mélange

<https://www.agroparistech.fr/IMG/pdf/ClassificationNonSupervisee-AgroParisTech.pdf>

# Outline

- ① Clustering: introduction
- ② Distance-based methods
- ③ Model-based approach
  - Mixture models**
  - Expectation-Maximization algorithm

# Latent variables models

## Definition

A **latent variable model** is a statistical model that relates, for  $i = 1, \dots, n$  individuals,

- a set of **manifest** (observed) variables  $\mathbf{X} = (X_i, i = 1, \dots, n)$  to
- a set of **latent** (unobserved) variables  $\mathbf{Z} = (Z_i, i = 1, \dots, n)$ .

Common assumption: conditional independence

$$\mathbb{P}((X_1, \dots, X_n) | (Z_1, \dots, Z_n)) = \prod_{i=1}^n \mathbb{P}(X_i | Z_i).$$

Famous examples

- $(Z_i, i \geq 1)$  is Markov chain: **Markov models**
- $Z_i$  categorical and independent: **mixture models**

# Latent variables models

## Definition

A **latent variable model** is a statistical model that relates, for  $i = 1, \dots, n$  individuals,

- a set of **manifest** (observed) variables  $\mathbf{X} = (X_i, i = 1, \dots, n)$  to
- a set of **latent** (unobserved) variables  $\mathbf{Z} = (Z_i, i = 1, \dots, n)$ .

Common assumption: conditional independence

$$\mathbb{P}((X_1, \dots, X_n) | (Z_1, \dots, Z_n)) = \prod_{i=1}^n \mathbb{P}(X_i | Z_i).$$

Famous examples

- $(Z_i, i \geq 1)$  is Markov chain: **Markov models**
- $Z_i$  categorical and independent: **mixture models**

## Mixture models: the latent variables

When  $(Z_1, \dots, Z_n)$  are independent categorical variables, they give a **natural (latent) classification of the observations**  $(X_1, \dots, X_n)$  – or **labels**.

### Notations

Let  $(Z_1, \dots, Z_n)$  be *iid* categorical variables with distribution

$$\mathbb{P}(i \in q) = \mathbb{P}(Z_i = q) = \alpha_q, \quad \text{s.t.} \sum_{q=1}^Q \alpha_q = 1.$$

### Alternative (equivalent) notation

Let  $Z_i = (Z_{i1}, \dots, Z_{iQ})$  be an indicator vector of label for  $i$ :

$$\mathbb{P}(i \in q) = \mathbb{P}(Z_{iq} = 1) = \alpha_q, \quad \text{s.t.} \sum_{q=1}^Q \alpha_q = 1.$$

By definition,  $Z_i \sim \mathcal{M}(1, \alpha)$ , with  $\alpha = (\alpha_1, \dots, \alpha_Q)$ .

# Mixture models: the latent variables

When  $(Z_1, \dots, Z_n)$  are independent categorical variables, they give a **natural (latent) classification of the observations**  $(X_1, \dots, X_n)$  – or **labels**.

## Notations

Let  $(Z_1, \dots, Z_n)$  be *iid* categorical variables with distribution

$$\mathbb{P}(i \in q) = \mathbb{P}(Z_i = q) = \alpha_q, \quad \text{s.t.} \sum_{q=1}^Q \alpha_q = 1.$$

Alternative (equivalent) notation

Let  $Z_i = (Z_{i1}, \dots, Z_{iQ})$  be an indicator vector of label for  $i$ :

$$\mathbb{P}(i \in q) = \mathbb{P}(Z_{iq} = 1) = \alpha_q, \quad \text{s.t.} \sum_{q=1}^Q \alpha_q = 1.$$

By definition,  $Z_i \sim \mathcal{M}(1, \alpha)$ , with  $\alpha = (\alpha_1, \dots, \alpha_Q)$ .



# Mixture models: the latent variables

When  $(Z_1, \dots, Z_n)$  are independent categorical variables, they give a **natural (latent) classification of the observations**  $(X_1, \dots, X_n)$  – or **labels**.

## Notations

Let  $(Z_1, \dots, Z_n)$  be *iid* categorical variables with distribution

$$\mathbb{P}(i \in q) = \mathbb{P}(Z_i = q) = \alpha_q, \quad \text{s.t.} \sum_{q=1}^Q \alpha_q = 1.$$

## Alternative (equivalent) notation

Let  $Z_i = (Z_{i1}, \dots, Z_{iQ})$  be an indicator vector of label for  $i$ :

$$\mathbb{P}(i \in q) = \mathbb{P}(Z_{iq} = 1) = \alpha_q, \quad \text{s.t.} \sum_{q=1}^Q \alpha_q = 1.$$

By definition,  $Z_i \sim \mathcal{M}(1, \alpha)$ , with  $\alpha = (\alpha_1, \dots, \alpha_Q)$ .

# Mixture models: the manifest variables

A mixture model represents the **presence of subpopulations** within an overall population as follows:

$$\mathbb{P}(X_i) = \sum_{z_i \in \mathcal{Z}_i} \mathbb{P}(X_i, Z_i) = \sum_{Z_i \in \mathcal{Z}_i} \mathbb{P}(X_i | Z_i) \mathbb{P}(Z_i).$$

## Conditional distribution of the manifest variables

We assume a **parametric distribution** of  $X$  in each subpopulation

$$X_i | \{Z_i = q\} \sim \mathbb{P}_{\theta_q} \quad \left( \Leftrightarrow X_i | \{Z_{iq}\} = 1 \sim \mathbb{P}_{\theta_q} \right)$$

The specificity of each class is handled by  $\{\theta_q\}_{q=1}^Q$ .

# Mixture models: likelihoods

## The complete-data likelihood

It is the joint distribution of  $(X_i, Z_i)$ :

$$\mathbb{P}(X_i, Z_i) = \alpha_{Z_i} \mathbb{P}_{\theta_{Z_i}}(X_i)$$

## The incomplete-data likelihood

It is the marginal distribution of  $X_i$  once  $Z_i$  integrated:

$$\mathbb{P}(X_i) = \sum_{q=1}^Q \mathbb{P}(X_i, Z_i = q) = \sum_{q=1}^Q \alpha_q \mathbb{P}_{\theta_q}(X_i)$$

↪ A mixture model is a sum of distributions weighed by the proportion of each subpopulation.

# Mixture models: likelihoods

## The complete-data likelihood

It is the joint distribution of  $(X_i, Z_i)$ :

$$\mathbb{P}(X_i, Z_i) = \alpha_{Z_i} \mathbb{P}_{\boldsymbol{\theta}_{Z_i}}(X_i)$$

## The incomplete-data likelihood

It is the marginal distribution of  $X_i$  once  $Z_i$  integrated:

$$\mathbb{P}(X_i) = \sum_{q=1}^Q \mathbb{P}(X_i, Z_i = q) = \sum_{q=1}^Q \alpha_q \mathbb{P}_{\boldsymbol{\theta}_q}(X_i)$$

↪ A **mixture model** is a sum of distributions weighed by the proportion of each subpopulation.

# Outline

- ① Clustering: introduction
- ② Distance-based methods
- ③ Model-based approach
  - Mixture models
  - Expectation-Maximization algorithm

# Intractability of the Likelihood

## Maximum Likelihood Estimator

The MLE aims to maximize the (marginal) likelihood of the observations:

$$L(\boldsymbol{\theta}; \mathbf{X}) = \mathbb{P}_{\boldsymbol{\theta}}((X_1, \dots, X_n)) = \int_{\mathbf{Z} \in \mathcal{Z}} \mathbb{P}_{\boldsymbol{\theta}}(\mathbf{X}, \mathbf{Z}) d\mathbf{Z}$$

Integrations are summation over  $\{1, \dots, Q\}$ : we have  $Q^n$  terms !

## Intractable summation

With mixture models, for  $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_Q)$  we have

$$\log L(\boldsymbol{\theta}; \mathbf{X}) = \sum_{i=1}^n \log \left\{ \sum_{q=1}^Q \alpha_q \mathbb{P}_{\boldsymbol{\theta}_q}(X_i) \right\}.$$

↪ Direct maximization of the likelihood is impossible in practice

# Intractability of the Likelihood

## Maximum Likelihood Estimator

The MLE aims to maximize the (marginal) likelihood of the observations:

$$L(\boldsymbol{\theta}; \mathbf{X}) = \mathbb{P}_{\boldsymbol{\theta}}((X_1, \dots, X_n)) = \int_{\mathbf{Z} \in \mathcal{Z}} \mathbb{P}_{\boldsymbol{\theta}}(\mathbf{X}, \mathbf{Z}) d\mathbf{Z}$$

Integrations are summation over  $\{1, \dots, Q\}$ : we have  $Q^n$  terms !

## Intractable summation

With mixture models, for  $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_Q)$  we have

$$\log L(\boldsymbol{\theta}; \mathbf{X}) = \sum_{i=1}^n \log \left\{ \sum_{q=1}^Q \alpha_q \mathbb{P}_{\boldsymbol{\theta}_q}(X_i) \right\}.$$

⇒ Direct maximization of the likelihood is impossible in practice

# Bayes decision rule / Maximum *a posteriori*

## Principle

Affect an individual  $i$  to the subpopulation which is the most likely according to the data:

$$\tau_{iq} = \mathbb{P}(Z_{iq} = 1 | X_i = x_i)$$

This is the **posterior probability** for  $i \in q$ .

## Application of the Bayes Theorem

It is straightforward to show that

$$\tau_{iq} = \frac{\alpha_q \mathbb{P}_{\theta_q}(x_i)}{\sum_{q=1}^Q \alpha_q \mathbb{P}_{\theta_q}(x_i)}$$



# Principle of the EM algorithm

If  $\theta$  were known

...estimating the **posterior probability**  $\mathbb{P}(Z_i|\mathbf{X})$  of  $\mathbf{Z}$  should be easy

*By means of the Bayes decision rule*

If  $\mathbf{Z}$  were known...

...estimating the **best set of parameter**  $\theta$  should be easy

*This is close to usual maximum likelihood estimation*

EM principle

Maximize the marginal likelihood iteratively:

- ① Initialize  $\theta$
- ② Compute the probability of  $\mathbf{Z}$  given  $\theta$
- ③ Get a better  $\theta$  with the new  $\mathbf{Z}$
- ④ Iterate until convergence

# Principle of the EM algorithm

If  $\theta$  were known

...estimating the **posterior probability**  $\mathbb{P}(Z_i|\mathbf{X})$  of  $\mathbf{Z}$  should be easy

*By means of the Bayes decision rule*

If  $\mathbf{Z}$  were known...

...estimating the **best set of parameter**  $\theta$  should be easy

*This is close to usual maximum likelihood estimation*

## EM principle

Maximize the marginal likelihood iteratively:

- 1 Initialize  $\theta$
- 2 Compute the probability of  $\mathbf{Z}$  given  $\theta$
- 3 Get a better  $\theta$  with the new  $\mathbf{Z}$
- 4 Iterate until convergence

# Formal algorithm

**Initialization:** start from a good guess either of  $\mathbf{Z}$  or  $\boldsymbol{\theta}$ , then iterate 1-2

## 1. Expectation step

Calculate the expected value of the loglikelihood under the current  $\boldsymbol{\theta}$

$$Q\left(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)}\right)=\mathbb{E}_{\mathbf{Z}|\mathbf{X};\boldsymbol{\theta}^{(t)}}\left[\log L(\boldsymbol{\theta};\mathbf{X},\mathbf{Z})\right] \quad (\text{needs } \mathbb{P}_{\boldsymbol{\theta}^{(t)}}(\mathbf{Z}|\mathbf{X}))$$

## 2. Maximization step

Find the parameters that maximize this quantity

$$\boldsymbol{\theta}^{(t+1)}=\arg\max_{\boldsymbol{\theta}}Q\left(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)}\right)$$

Stop when  $\|\boldsymbol{\theta}^{(t+1)}-\boldsymbol{\theta}^{(t)}\|<\varepsilon$  or  $\|Q^{(t+1)}-Q^{(t)}\|<\varepsilon$

# (Basic) Convergence analysis

## Theorem

*At each step of the EM algorithm, the loglikelihood increases. EM thus reaches a local optimum.*

## Proof.

On board.



# Choosing the number of component

## Reminder: Bayesian Information Criterion

The BIC is a model selection criterion which penalizes the adjustment to the data by the number of parameter in model  $\mathcal{M}$  as follows:

$$\text{BIC}(\mathcal{M}) = \log L(\hat{\boldsymbol{\theta}}; \mathbf{X}) - \frac{1}{2} \log(n) \text{df}(\mathcal{M}).$$

## Integrated Classification Criterion

It is an adaptation working with the complete-data likelihood:

$$\begin{aligned} \text{ICL}(\mathcal{M}) &= \log L(\hat{\boldsymbol{\theta}}; \mathbf{X}, \hat{\mathbf{Z}}) + \frac{1}{2} \log(n) \text{df}(\mathcal{M}) \\ &= \text{BIC} - \mathcal{H}(\mathbb{P}(\hat{\mathbf{Z}}|\mathbf{X})), \end{aligned}$$

where the entropy  $\mathcal{H}$  measures the separability of the subpopulations.

⇒ We choose  $\mathcal{M}(Q)$  that maximizes either BIC or ICL

# Choosing the number of component

## Reminder: Bayesian Information Criterion

The BIC is a model selection criterion which penalizes the adjustment to the data by the number of parameter in model  $\mathcal{M}$  as follows:

$$\text{BIC}(\mathcal{M}) = \log L(\hat{\boldsymbol{\theta}}; \mathbf{X}) - \frac{1}{2} \log(n) \text{df}(\mathcal{M}).$$

## Integrated Classification Criterion

It is an adaptation working with the complete-data likelihood:

$$\begin{aligned} \text{ICL}(\mathcal{M}) &= \log L(\hat{\boldsymbol{\theta}}; \mathbf{X}, \hat{\mathbf{Z}}) + \frac{1}{2} \log(n) \text{df}(\mathcal{M}) \\ &= \text{BIC} - \mathcal{H}(\mathbb{P}(\hat{\mathbf{Z}}|\mathbf{X})), \end{aligned}$$

where the entropy  $\mathcal{H}$  measures the separability of the subpopulations.

⇒ We choose  $\mathcal{M}(Q)$  that maximizes either BIC or ICL

# Choosing the number of component

## Reminder: Bayesian Information Criterion

The BIC is a model selection criterion which penalizes the adjustment to the data by the number of parameter in model  $\mathcal{M}$  as follows:

$$\text{BIC}(\mathcal{M}) = \log L(\hat{\boldsymbol{\theta}}; \mathbf{X}) - \frac{1}{2} \log(n) \text{df}(\mathcal{M}).$$

## Integrated Classification Criterion

It is an adaptation working with the complete-data likelihood:

$$\begin{aligned} \text{ICL}(\mathcal{M}) &= \log L(\hat{\boldsymbol{\theta}}; \mathbf{X}, \hat{\mathbf{Z}}) + \frac{1}{2} \log(n) \text{df}(\mathcal{M}) \\ &= \text{BIC} - \mathcal{H}(\mathbb{P}(\hat{\mathbf{Z}}|\mathbf{X})), \end{aligned}$$

where the entropy  $\mathcal{H}$  measures the separability of the subpopulations.

⇒ We choose  $\mathcal{M}(Q)$  that maximizes either BIC or ICL

## Popular model: Gaussian Multivariate mixture models

The distribution of  $X_i$  conditional on the label of  $i$  is assumed to be a multivariate Gaussian distribution with unknown parameters:

$$X_i | i \in q \sim \mathcal{N}(\boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)$$

### Complete Likelihood ( $\mathbf{X}, \mathbf{Z}$ )

The model complete loglikelihood is

$$\log L(\boldsymbol{\mu}, \boldsymbol{\Sigma}; \mathbf{X}, \mathbf{Z}) = \sum_{i=1}^n \sum_{q=1}^Q Z_{iq} \left( \log \alpha_q - \frac{1}{2} \log \det(\boldsymbol{\Sigma}_q) - \frac{1}{2} \|\mathbf{x}_i - \boldsymbol{\mu}_q\|_{\boldsymbol{\Sigma}_q^{-1}}^2 \right) + c$$

↪ Implementation of the univariate case during the labs.



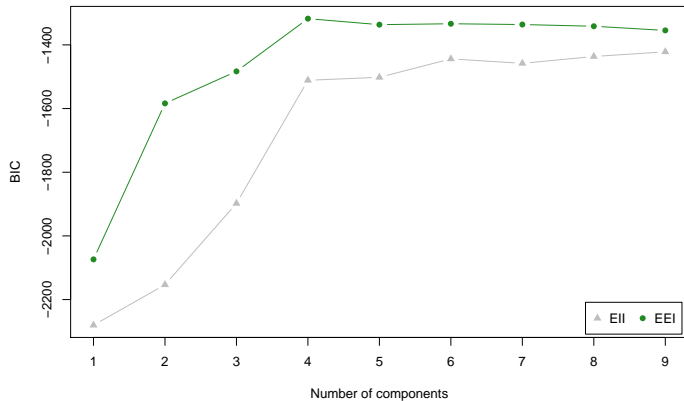
# Gaussian mixture model in R I

The package Mclust is a great reference

See <https://cran.r-project.org/web/packages/mclust/vignettes/mclust.html>

```
GMM <- crabs_corrected %>%  
  select(-sex, -species) %>%  
  Mclust(modelNames = c("EII", "EEI"))  
plot(GMM, 'BIC')
```

## Gaussian mixture model in R II



# Gaussian mixture model in R III

```
aricode::ARI(GMM$classification, classes)

## [1] 0.6451662

aricode::ARI(GMM$classification, clusters)

## [1] 0.8746812

aricode::ARI(GMM$classification, clusters_ward)

## [1] 0.8209783

plot(GMM, 'classification')
```

# Gaussian mixture model in R IV

