

# Data analysis and Unsupervised Learning

## Clustering: distance-based methods

MAP 573, 2020 – Julien Chiquet

École Polytechnique, Autumn semester, 2020

<https://jchiquet.github.io/MAP573>



## 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(FactoMineR) # PCA and oter linear method for dimension reduction
library(factoextra) # fancy plotting for FactoMineR output
library(kernlab)   # Kernel-based methods, among which spectral-clustering
library(aricode)   # fast computation of clustering measures
library(animation) # kmeans animation slides
library(igraph)    # graph manipulation
theme_set(theme_bw()) # plots themes
```

# 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.

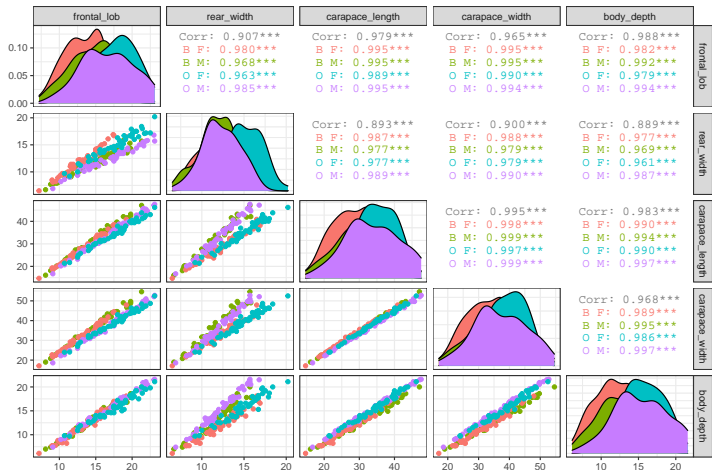
```
crabs <- MASS::crabs %>% select(-index) %>%  
  rename(sex = sex,  
         species = sp,  
         frontal_lob = FL,  
         rear_width = RW,  
         carapace_length = CL,  
         carapace_width = CW,  
         body_depth = BD)  
crabs %>% select(sex, species) %>% summary() %>% knitr::kable("latex")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

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

# Companion data set II

## Pairs plot of attributes

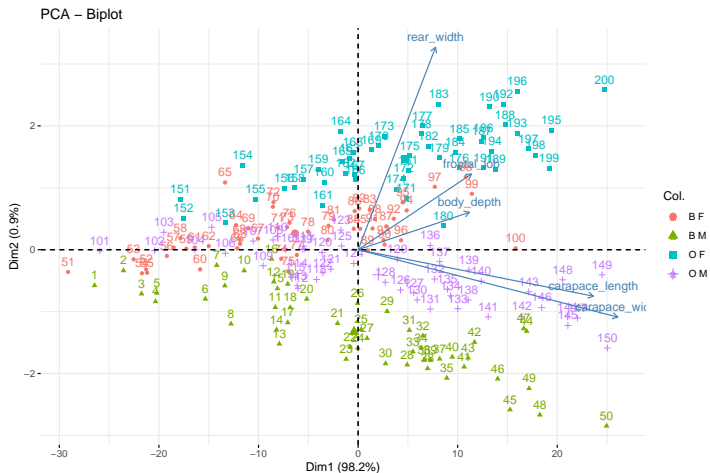
```
ggpairs(crabs, columns = 3:7, aes(colour = paste(crabs$species, crabs$sex)))
```



# Companion data set III

## PCA on the attributes

```
select(crabs, -species, -sex) %>% PCA(scale.unit = FALSE, graph = FALSE) %>%  
  fviz_pca_biplot(axes = c(1,2), col.ind = paste(crabs$species, crabs$sex))
```



# Remove size effect I

Carried by the 1st principal component

## First component

$$\mathbf{f}_1 = \mathbf{X}^c \mathbf{u}_1.$$

We extract the best rank-1 approximation of  $\mathbf{X}$  to remove the *size effect*, carried by the first axis, and return to the original space,

$$\tilde{\mathbf{X}}^{(1)} = \mathbf{f}_1 \mathbf{u}_1^\top.$$

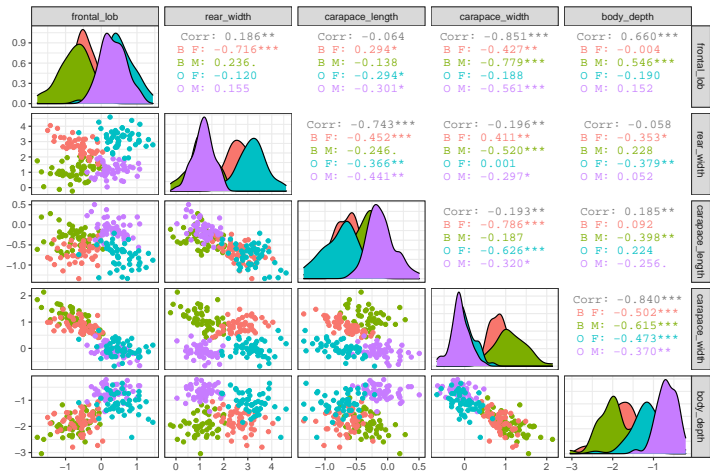
```
attributes <- select(crabs, -sex, -species) %>% as.matrix()
u1 <- eigen(cov(attributes))$vectors[, 1, drop = FALSE]
attributes_rank1 <- attributes %*% u1 %*% t(u1)
crabs_corrected <- crabs
crabs_corrected[, 3:7] <- attributes - attributes_rank1
```

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

# Remove size effect II

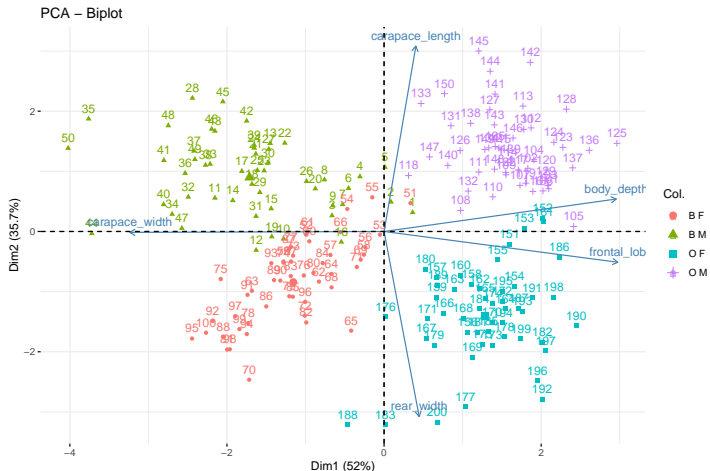
Carried by the 1st principal component

```
ggpairs(crabs_corrected, columns = 3:7, aes(colour = paste(crabs$species, crabs$sex))
```



# PCA on corrected data

```
select(crabs_corrected, -species, -sex) %>% FactoMineR::PCA(graph = FALSE) %>%  
fviz_pca_biplot(col.ind = paste(crabs_corrected$species, crabs_corrected$sex))
```





# Questions

- ① Could we automatically identify some grouping (**clustering**) between samples?
- ② Would this clustering correspond to some known labels (sex, species)?
- ③ Do we need to transform the data before we perform clustering?

# Clustering: general goals

**Objective:** construct a map

$$f : \mathcal{D} = \{1, \dots, n\} \mapsto \{1, \dots, K\}$$

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  $\mathbb{P}$

## Part II

### Distance-based method

# Outline

## Distance-based method

- 1 Clustering: introduction
- 2 The K-means algorithm
- 3 Hierarchical Agglomerative Clustering
- 4 Spectral Clustering

# 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  $V$ .

$U \setminus V$	$V_1$	$V_2$	$\dots$	$V_{ V }$	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

## Distance-based method

- 1 Clustering: introduction
- 2 The K-means algorithm**
- 3 Hierarchical Agglomerative Clustering
- 4 Spectral Clustering

# 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:

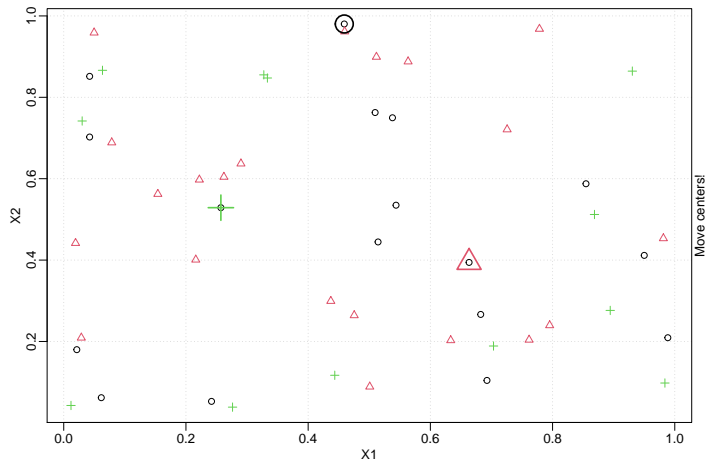
## K-means algorithm (Loyds)

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

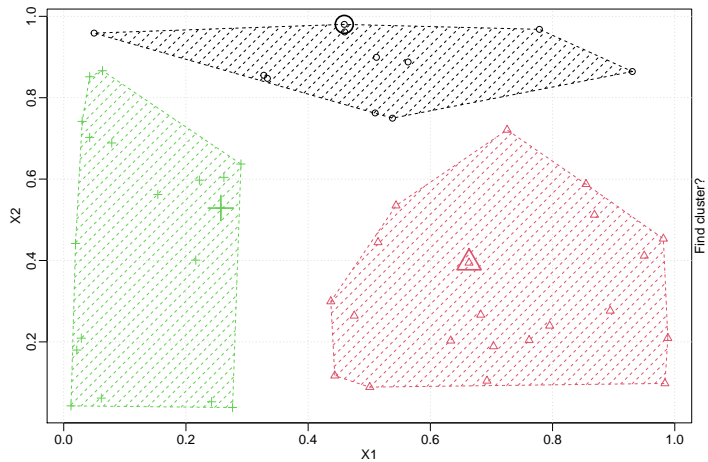
**Alternate** until convergence

- 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

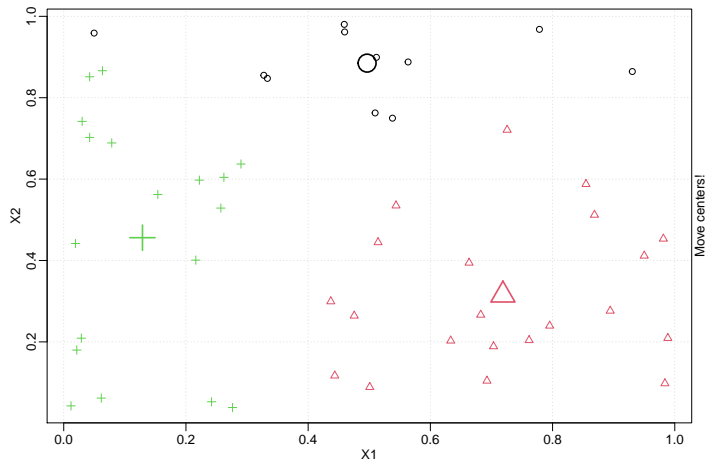
# K-means in action I



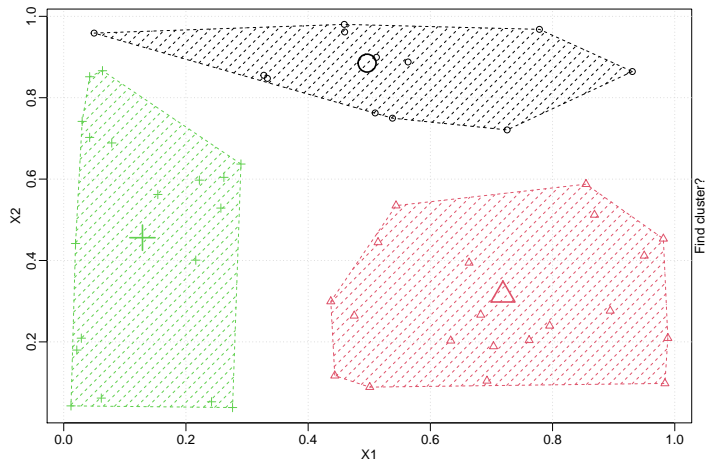
# K-means in action II



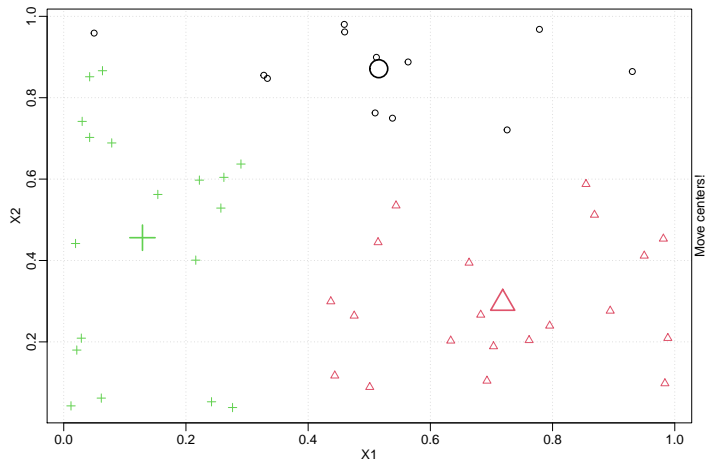
## K-means in action III



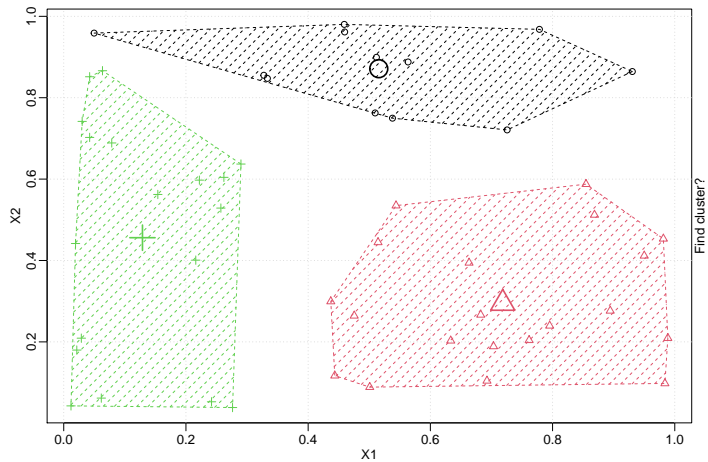
# K-means in action IV



# K-means in action V



# K-means in action VI



# K-means: 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.

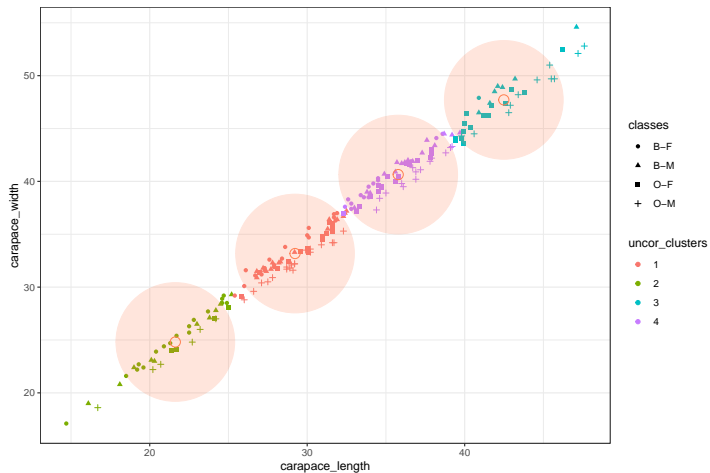


## K-means in R on uncorrected data set I

```
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)
```

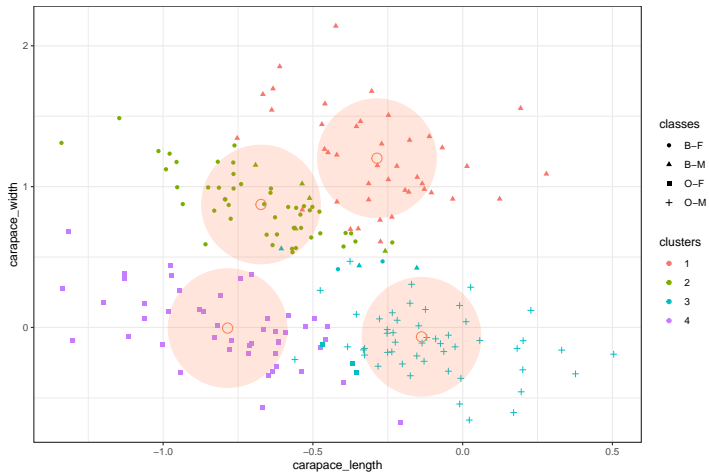
# K-means in R on uncorrected data set II



## 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] 0.8317615

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	42	0	0
48	5	0	0
2	3	3	50
0	0	47	0

# How about a "spectral" k-means? I

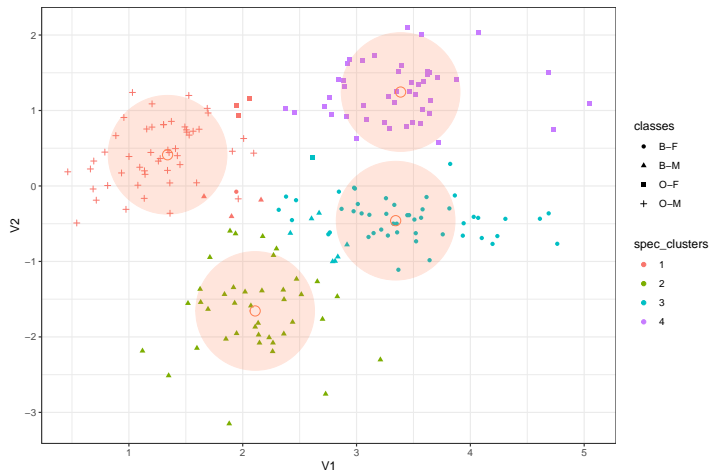
## PCA + k-means

```
SVD <- svd(select(crabs_corrected, -species, -sex))
spec_crabs <- as.tibble(SVD$u[,1:2] %*% diag(SVD$d[1:2]))
spec_kmeans_res <- spec_crabs %>%
  kmeans(4, nstart = 10)
spec_clusters <- as.factor(spec_kmeans_res$cluster)
spec_centers <- as.tibble(spec_kmeans_res$centers)
classes <- paste(crabs_corrected$species, crabs_corrected$sex, sep = "-")

ggplot(spec_crabs, aes(V1, V2, color = spec_clusters)) +
  geom_point(aes(shape = classes)) +
  geom_point(data = spec_centers, color = 'coral', size = 4, pch = 21) +
  geom_point(data = spec_centers, color = 'coral', size = 50, alpha = 0.2)
```

# How about a "spectral" k-means? II

PCA + k-means



# How about a "spectral" k-means? III

PCA + k-means

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

```
## [1] 0.8090372
```

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

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

B-F	B-M	O-F	O-M
1	3	3	50
0	40	0	0
49	7	1	0
0	0	46	0



# Outline

## Distance-based method

- 1 Clustering: introduction
- 2 The K-means algorithm
- 3 Hierarchical Agglomerative Clustering**
- 4 Spectral Clustering

# 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

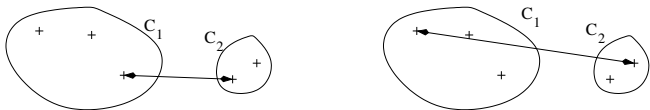
- 1 Start with  $(\mathcal{C}_k^{(0)}) = (\{\mathbf{x}_i\})$  the collection of all singletons.
- 2 At step  $s$ , we have  $n - s$  clusters  $(\mathcal{C}_k^{(s)})$ :
  - Find the two most similar clusters according to a criterion  $\Delta$ :

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

- 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:

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

Euclidean case

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 inertia measures the homogeneity of the size-K clustering

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

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 loss (increase) of the intra (inter) 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 inertia measures the homogeneity of the size-K clustering

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

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 loss (increase) of the intra (inter) 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 inertia measures the homogeneity of the size-K clustering

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

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

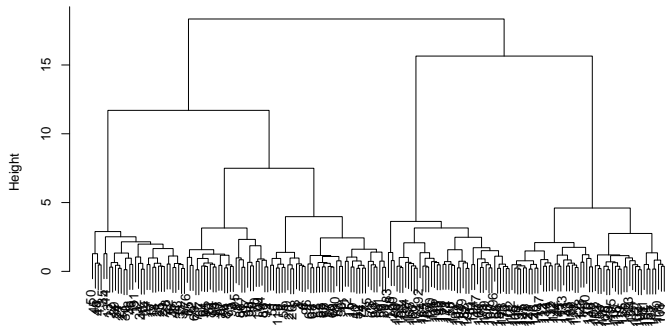
$$I_W(\mathcal{P}') - I_W(\mathcal{P}) = \frac{|\mathcal{C}_k| |\mathcal{C}_\ell|}{|\mathcal{C}_k| + |\mathcal{C}_\ell|} d^2(\boldsymbol{\mu}_{\mathcal{C}_k}, \boldsymbol{\mu}_{\mathcal{C}_\ell}) = \frac{1}{2} \Delta(\mathcal{C}_k, \mathcal{C}_\ell).$$

- ↪ At each step, Ward limits the loss (increase) of the intra (inter) 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



# Ward agglomerative clustering in R: comparison I

Compare with out reference classification and k-means

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

```
## [1] 0.7071894
```

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

```
## [1] 0.7538279
```

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

Table: k-means vs Ward

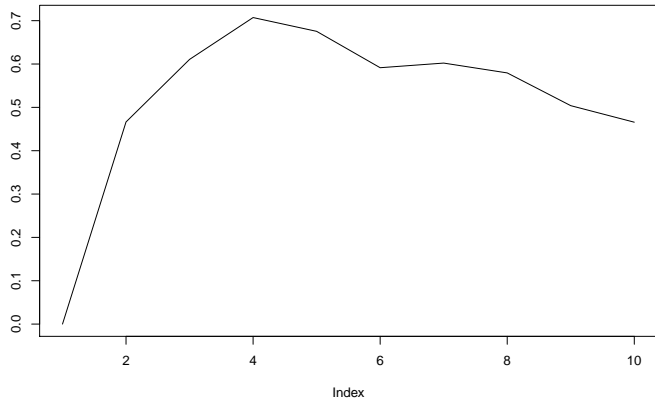
1	2	3	4
9	33	0	0
53	0	0	0
6	0	52	0
2	0	2	43

# 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")
```

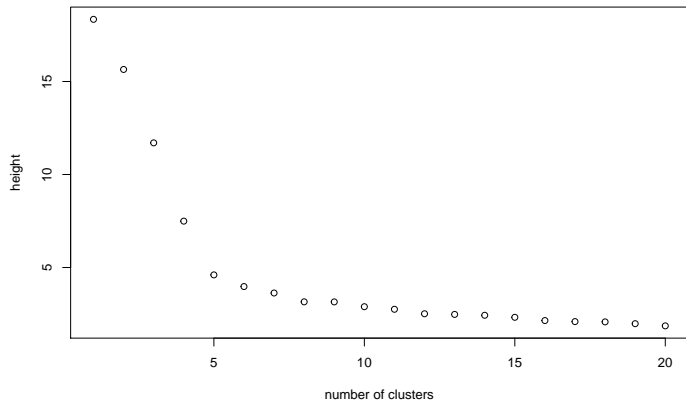
## Ward agglomerative clustering in R: comparison III



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")
```

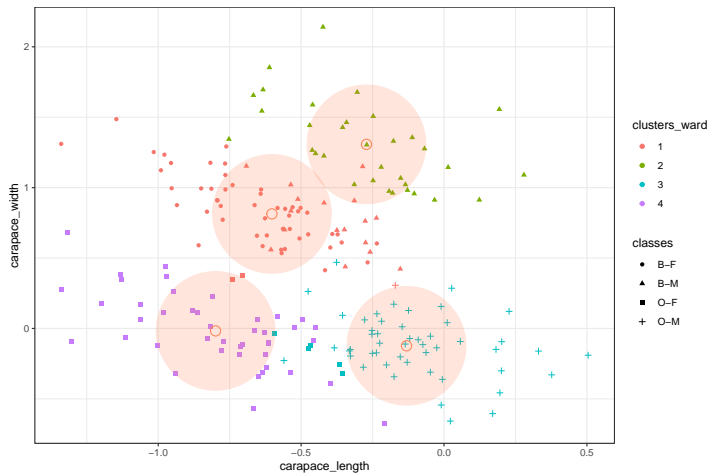


# Ward agglomerative clustering in R: projection I

```
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







# Outline

## Distance-based method

- 1 Clustering: introduction
- 2 The K-means algorithm
- 3 Hierarchical Agglomerative Clustering
- 4 Spectral Clustering**

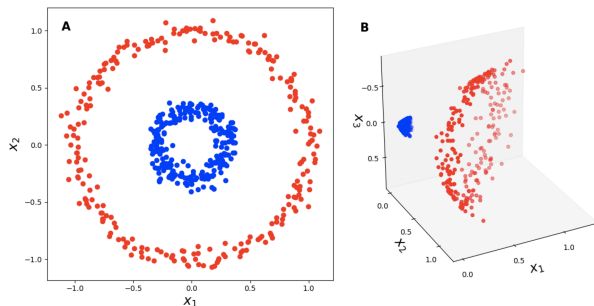
# References

-  DS David Sontag's Lecture  
<http://people.csail.mit.edu/dsontag/courses/ml13/slides/lecture16.pdf>
-  A Tutorial on Spectral Clustering,  
Ulrike von Luxburg

# Spectral Clustering

Principle: graph-based transformation prior to clustering

- 1 Build a similarity with a weighted graph of the data
- 2 Use the spectral property of this similarity ( $\rightsquigarrow$  kernel)
- 3 Apply clustering (e.g., k-means) to the projected data



**Figure:** Performing clustering after transformation + dimension reduction of the data

# Creating the graph

## Many choices

- K-nearest neighbor graph
- any distance-based similarity (fully connected graph)
- any kernel-based similarity (e.g., Gaussian kernel)

The connectivity of  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  is captured by the (weighted) adjacency matrix  $\mathbf{A}$ :

$$(\mathbf{A})_{ij} = \begin{cases} w_{ij} > 0 & \text{if } i \sim j, \\ 0 & \text{otherwise.} \end{cases}$$

## Proposition

*The degrees of  $\mathcal{G}$  are then simply obtained as the row-wise and/or column-wise sums of  $\mathbf{A}$ .*

# Incidence matrix

## Definition (Incidence matrix)

The connectivity of  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  is captured by the  $|\mathcal{V}| \times |\mathcal{E}|$  matrix  $\mathbf{B}$ :

$$(\mathbf{B})_{ij} = \begin{cases} \sqrt{w_{ij}} & \text{if } i \text{ is incident to edge } j, \\ 0 & \text{otherwise.} \end{cases}$$

## Proposition (Relationship)

Let  $\tilde{\mathbf{B}}$  be a modified *signed* version of  $\mathbf{B}$  where  $\tilde{B}_{ij} = +/ - \sqrt{w_{ij}}$  if  $i$  is incident to  $j$  as tail/head. Then

$$\tilde{\mathbf{B}}\tilde{\mathbf{B}}^\top = \mathbf{D} - \mathbf{A},$$

where  $\mathbf{D} = \text{diag}(\{d_i, i \in \mathcal{V}\})$  is the diagonal matrix of degrees.

# Graph Laplacian

## Definition ((Un-normalized) Laplacian)

The Laplacian matrix  $\mathbf{L}$ , resulting from the modified incidence matrix  $\tilde{\mathbf{B}}$   $\tilde{B}_{ij} = 1/ -1$  if  $i$  is incident to  $j$  as tail/head, is defined by

$$\mathbf{L} = \tilde{\mathbf{B}}\tilde{\mathbf{B}}^T = \mathbf{D} - \mathbf{A},$$

where  $\mathbf{D} = \text{diag}(d_i, i \in \mathcal{V})$  is the diagonal matrix of degrees.

## Remark

- $\mathbf{L}$  is called the graph Laplacian (by analogy to continuous Laplacian).
- Spectrum of  $\mathbf{L}$  has much to say about the structure of the graph  $\mathcal{G}$ .

# Graph Laplacian: spectrum

## Proposition (Spectrum of $\mathbf{L}$ )

*The  $n \times n$  matrix  $\mathbf{L}$  has the following properties:*

$$\mathbf{x}^\top \mathbf{L} \mathbf{x} = \frac{1}{2} \sum_{i,j} A_{ij} (x_i - x_j)^2, \quad \forall \mathbf{x} \in \mathbb{R}^n.$$

- $\mathbf{L}$  is a symmetric, positive semi-definite matrix,
- the smallest eigenvalue is 0 with associated eigenvector  $\mathbf{1}$ .
- $\mathbf{L}$  has  $n$  positive eigenvalues  $0 = \lambda_1 < \dots < \lambda_n$ .

## Corollary (Spectrum and Graph)

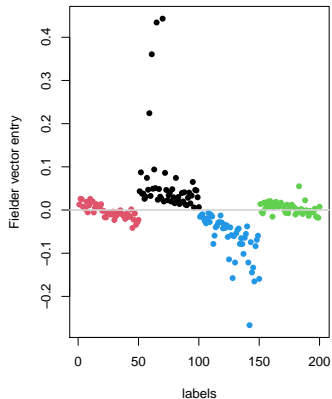
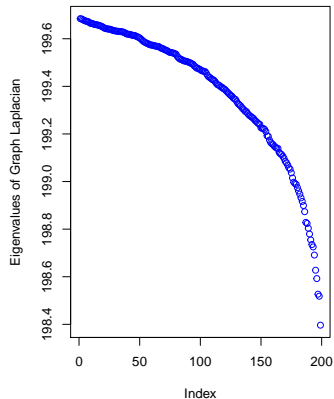
- The multiplicity of the first eigen value (0) of  $\mathbf{L}$  determines the number of connected components in the graph.
- The larger the second non trivial eigenvalue, the higher the connectivity of  $\mathcal{G}$ .

# Crabs: Fielder vector and eigenvalue I

```
graph_crabs <- crabs %>% select(-species, -sex) %>%  
  t() %>% cor() %>% graph_from_adjacency_matrix(weighted = TRUE)  
eigen_crabs <- graph.laplacian(graph_crabs) %>% eigen()  
  
fielder_vector <- eigen_crabs$vectors[, nrow(crabs) - 1]  
faction <- factor(paste(crabs$species, crabs$sex, sep="-"))  
  
par(mfrow = c(1,2))  
plot(eigen_crabs$values[-nrow(crabs)], col = "blue", ylab = "Eigenvalues of Graph I")  
plot(fielder_vector, pch = 16, xlab = "labels",  
      ylab = "Fielder vector entry", col = faction)  
abline(0, 0, lwd = 2, col = "lightgray")
```



# Crabs: Fielder vector and eigenvalue II



## Some variants

### Definition ((Normalized) Laplacian)

The normalized Laplacian matrix  $\mathbf{L}$  is defined by

$$\mathbf{L}_N = \mathbf{D}^{-1/2} \mathbf{L} \mathbf{D}^{-1/2} = \mathbf{I} - \mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{-1/2}.$$

### Definition ((Absolute) Graph Laplacian)

The absolute Laplacian matrix  $\mathbf{L}_{abs}$  is defined by

$$\mathbf{L}_{abs} = \mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{-1/2} = \mathbf{I} - \mathbf{L}_N,$$

with eigenvalues  $1 - \lambda_n \leq \dots \leq 1 - \lambda_2 \leq 1 - \lambda_1 = 1$ , where  $0 = \lambda_1 \leq \dots \leq \lambda_n$  are the eigenvalues of  $\mathbf{L}_N$ .

# Normalized Spectral Clustering

by Ng, Jordan and Weiss (2002)

**Input:** Adjacency matrix and number of classes  $Q$

Compute the normalized graph Laplacian  $\mathbf{L}$

Compute the eigen vectors of  $\mathbf{L}$  associated with the  $Q$  **smallest eigenvalues**

Define  $\mathbf{U}$ , the  $n \times Q$  matrix that encompasses these  $Q$  vectors

Define  $\tilde{\mathbf{U}}$ , the row-wise normalized version of  $\mathbf{U}$ :  $\tilde{u}_{ij} = \frac{u_{ij}}{\|\mathbf{U}_i\|_2}$

Apply k-means to  $(\tilde{\mathbf{U}}_i)_{i=1,\dots,n}$

**Output:** vector of classes  $\mathbf{C} \in \mathcal{Q}^n$ , such as  $C_i = q$  if  $i \in q$

# Absolute Spectral Clustering

by Rohe et al. (2011)

**Input:** Adjacency matrix and number of classes  $Q$

Compute the graph Laplacian  $\mathbf{L}_{abs}$

Compute the eigen vectors of  $\mathbf{L}_{abs}$  associated with the  $Q$  largest absolute eigenvalues

Define  $\mathbf{U}$ , the  $p \times Q$  matrix that encompasses these  $Q$  vectors

Apply k-means to  $(\mathbf{U}_i)_{i=1,\dots,p}$

**Output:** vector of classes  $\mathbf{C} \in \mathcal{Q}^p$ , such as  $C_i = q$  if  $i \in q$