STATISTIQUES DESCRIPTIVES AVEC **R**

Amandine Blin
UAR 2700 2AD, Service Analyse de Données, Pôle Analyse de Données

18/10/2022

Table des matières

Chapitre 1

Quelques rappels et description des jeux de données

Des références de livres sont indiquées à la fin du document : Zar (1984), Verzani (2005), Frédéric Bertrand (2010), Sievert (2020), Wilke (2019).

1.1 Quelques définitions

Statistique

Science permettant d'effectuer des déductions sur les caractéristiques d'une population ou d'un échantillon issu de la population. Elle regroupe les techniques de la collecte au traitement et à l'analyse des résultats.

Population

Ensemble d'individus sur lequel porte l'analyse statistique.

Individus

Eléments constituant la population. C'est sur l'individu que l'on va effectuer une mesure. C'est une unité statistique.

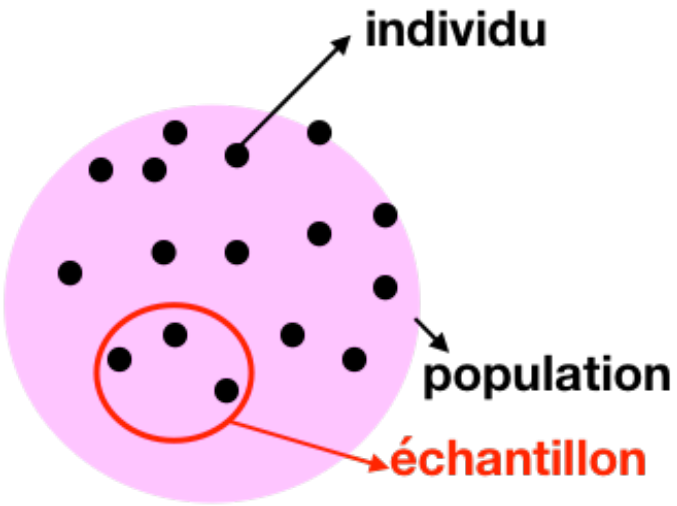
Echantillon

Sous-ensemble de la population (aléatoire ou non aléatoire). Pour qu'un échantillon soit représentatif de la population, l'échantillon doit être aléatoire. Lorsque la population est trop importante, il est impossible d'effectuer toutes les mesures. C'est donc à partir de l'échantillon que l'on va tirer des conclusions sur la population.

Inférence statistique

Induire sur les caractéristiques d'une population inconnue à partir d'un échantillon avec une marge d'erreur.

Exemple : On souhaite étudier l'invasion des criquets en Afrique de l'est. La population est l'ensemble des criquets. C'est un ensemble qui n'est pas défini et qui va varier au cours du temps.



Variable

Caractéristique commune à tous les individus qu'on souhaite étudier. Chaque observation prend une valeur différente selon l'individu. Une variable est aléatoire si on ne peut pas prédire la valeur pour un individu. Elle peut être :

- quantitative : continue ou discrète
- qualitative possédant des modalités ou classes : nominale ou ordinale

Exemples : le poids des lapins (variable quantitative continue), le nombre de personnes lors d'un congrès (quantitative discrète), la race de chien (qualitative nominale) et l'appréciation faible/moyen/bon/très bon (qualitative ordinale).

1.2 Le jeu de données *ChickWeight*

```
data(ChickWeight)
help("ChickWeight")
```

```
## démarrage du serveur d'aide httpd ... fini
```

- Quatre groupes de poussins ont été pesés à la naissance, puis tous les deux jours jusqu'au vingt-et-unième jour. Les poussins ont été soumis à un régime différent.
- Le tableau de données a 578 lignes (ou observations) et 4 colonnes (variables).
- La variable *weight* indique le poids des poussins (g).
- La variable *Time* indique le nombre de jours depuis la naissance jusqu'au jour où la mesure a été faite.
- La variable *Chick* indique l'identifiant du poussin.
- La variable *Diet* indique le régime expérimental que le poussin a reçu.

1.3 Le jeu de données *corvus* du package ade4 (Laiolo and Rolando (2003))

```
data(corvus, package="ade4")
help("corvus",package="ade4")
```

28 espèces de corbeaux ont été mesurées. Il y a 4 variables :

- La variable *wing* : la longueur d'aile en mm
- La variable *bill* : la longueur du bec en mm
- La variable *habitat* : type d'habitat (ouvert ou fermé)
- La variable *phylog* : groupe phylogéographique

1.4 Le jeu de données *iris*

```
data(iris)
help("iris")
```

Les données ont été collectées par Edgar Anderson (Anderson (1935)). 50 fleurs de 3 espèces (*Species*) différentes d'iris (*iris setosa*, *versicolor*, et *virginica*) ont été mesurées (en cm) :

- La longueur des sépales (*Sepal.Length*)
- La largeur des sépales (*Sepal.Width*)
- La longueur des pétales (*Petal.Length*)
- La largeur des pétales (*Petal.Width*)

1.5 A VOUS DE JOUER !

- Importer votre jeu de données dans R.

Chapitre 2

Statistique descriptive univariée

Cela consiste à effectuer l'analyse statistique sur une seule variable aléatoire (quantitative ou qualitative).

2.1 Cas d'une variable quantitative

Une variable quantitative est composée de valeurs numériques. Elle peut être continue ou discrète. Dans certains cas, on peut modifier une variable quantitative en variable qualitative. Exemple : la variable âge peut être une série numérique mais on peut modifier la variable avec des classes d'âge.

Prenons la série statistique suivante :

```
serie <- c(4, 5, 7, 34, 6, 2, 1, 1, 8, 8, 8, 8, 8, 4, 5, 4, 2)
```

Comment peut-on résumer une série statistique ?

— Nombre d'éléments d'une série, le nombre d'observations avec la fonction *length()*

```
length(serie)
```

```
## [1] 17
```

On peut aussi calculer l'effectif de chaque valeur avec la fonction *table()*.

```
tableau <- table(serie)
print(tableau)
```

```
## serie
##  1  2  4  5  6  7  8 34
##  2  2  3  2  1  1  5  1
```

On peut également calculer les effectifs cumulés des valeurs de la série :

```
cumsum(serie)
```

```
##  [1]   4   9  16  50  56  58  59  60  68  76  84  92 100 104 109 113 115
```

Si on souhaite calculer le tableau des fréquences de chacune des valeurs, on peut utiliser la fonction *prop.table()*.

```
prop.table(serie)
```

```
##  [1] 0.034782609 0.043478261 0.060869565 0.295652174 0.052173913 0.017391304
##  [7] 0.008695652 0.008695652 0.069565217 0.069565217 0.069565217 0.069565217
## [13] 0.069565217 0.034782609 0.043478261 0.034782609 0.017391304
```

Les indicateurs de tendance centrale

— La moyenne arithmétique : somme des valeurs divisée par la somme totale. Dans le cas de valeurs extrêmes, il est nécessaire de calculer d'autres indicateurs.

```
mean(serie)
```

```
## [1] 6.764706
```

— La médiane : valeur qui partage la série en deux sous-ensembles contenant chacun la moitié des observations

```
median(serie)
```

```
## [1] 5
```

— Le mode : valeur qui apparaît le plus fréquemment dans la série

```
max(tableau)
```

```
## [1] 5
```

```
which(tableau==max(tableau))
```

```
## 8
## 7
```

```
mode <- names(which(tableau==max(tableau)))
print(mode)
```

```
## [1] "8"
```

Les indices de position

— Le maximum

```
max(serie)
```

```
## [1] 34
```

— Le minimum

```
min(serie)
```

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

— La moyenne
— La médiane
— Le mode

Indicateur de dispersion

Plusieurs indicateurs de dispersion sont disponibles.

— Variance : moyenne du carré des écarts à la moyenne

```
var(serie)
```

```
## [1] 55.69118
```

— Ecart-type : racine carrée de la variance. L'écart-type mesure la dispersion autour de la moyenne. Dans le cas de valeurs extrêmes, il est nécessaire de calculer d'autres indicateurs.

```
sd(serie)
```

```
## [1] 7.462652
```

— Le p-ième quantile est la valeur (entre 0 et 1) qui sépare les données avec 100p % des observations en dessous de la valeur et 100(1-p)% au dessus. Les quartiles correspondent au 0%, 25%, 50%, 75% et 100% quantile.

```
quantile(serie)
```

```
##   0%  25%  50%  75% 100%
##    1    4    5    8   34
```

— Etendue

```
# La fonction range() donne le minimum et le maximum de la série.
range(serie)
```

```
## [1]  1 34
```

```
etendue <- diff(range(serie))
print(etendue)
```

```
## [1] 33
```

— Ecart interquartile : différence entre le troisième et le premier quartile

```
IQR(serie)
```

```
## [1] 4
```

— Coefficient de variation : rapport entre l'écart-type et la moyenne

```
sd(serie)/mean(serie)
```

```
## [1] 1.103175
```

La fonction *summary()*

La fonction *summary()* permet de résumer statistiquement la série statistique (minimum, premier quartile, médiane, moyenne, troisième quartile, maximum).

```
summary(serie)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   4.000   5.000   6.765   8.000  34.000
```

Indices de forme

On utilisera le package **e1071** (Meyer et al. (2019)).

Asymétrie d'une distribution

On peut calculer le coefficient d'asymétrie qui permet de mesurer l'asymétrie d'une distribution.

— Si le coefficient d'asymétrie est inférieur à zéro, la queue de distribution est étalée vers la gauche.
— Si le coefficient d'asymétrie est supérieur à zéro, la queue de distribution est étalée vers la droite.
— Si le coefficient est égal à zéro, la distribution est symétrique.
— Si la valeur absolue du coefficient est supérieure à 1, on peut considérer l'asymétrie très importante.

```
library(e1071)
skewness(serie)
```

```
## [1] 2.765247
```

Indices d'aplatissement

Le coefficient d'aplatissement (kurtosis) permet de mesurer la dispersion des valeurs extrêmes par rapport à la distribution gaussienne. Pour le calculer, on utilise la fonction *kurtosis()* du package **e1071**. Par défaut, la fonction renvoie la valeur de coefficient du kurtosis de normalité (excès d'aplatissement) c'est-à-dire le coefficient de kurtosis auquel on soustrait la valeur trois.

- Si le coefficient est égal à zéro, la distribution est normale.
- Si le coefficient est inférieur à zéro, les queues de distribution ont moins d'observations que la distribution gaussienne. Le pic est donc plus large et plus bas, il n'y a pas de courbe en forme de cloche (distribution aplatie) . Il y a également peu de valeurs extrêmes.
- Si le coefficient est supérieur à zéro, les queues de distribution ont plus d'observations que la distribution gaussienne. Le pic est donc plus haut, moins large (distribution pointue). Il y a également de nombreuses valeurs extrêmes.

```
kurtosis(serie)
```

```
## [1] 7.5007
```

Les deux indices de forme nous donnent une indication sur la normalité des données.

2.2 Cas d'une variable qualitative

Soit la série statistique *couleur*.

```
couleur <- c("bleu", "rouge", "rouge","rouge", "vert","vert","bleu",
             "rouge")
couleur <- as.factor(couleur)
```

De la même manière, on peut obtenir un résumé statistique d'une variable qualitative en utilisant la fonction *summary()* qui nous indiquera le nombre d'observations pour chaque modalité. A noter que la variable couleur doit être un objet de type facteur.

```
summary(couleur)
```

```
##  bleu rouge  vert
##     2     4     2
```

2.3 Application au jeu de données *ChickWeight*

— La variable *weight* : variable quantitative

— La variable *Time* : variable quantitative

— La variable *Chick* : variable qualitative

— La variable *Diet* : variable qualitative

```
summary(ChickWeight)
```

```
##      weight           Time           Chick     Diet
##  Min.   : 35.0   Min.   : 0.00   13     : 12   1:220
##  1st Qu.: 63.0   1st Qu.: 4.00   9      : 12   2:120
##  Median :103.0   Median :10.00   20     : 12   3:120
##  Mean   :121.8   Mean   :10.72   10     : 12   4:118
##  3rd Qu.:163.8   3rd Qu.:16.00   17     : 12
##  Max.   :373.0   Max.   :21.00   19     : 12
##                                  (Other):506
```

2.4 TP

— Décrivez les variables du jeu de données *iris*.

— Décrivez les variables du jeu de données *corvus*.

2.5 A VOUS DE JOUER !

— Décrivez chacune des variables de votre jeu de données.

Chapitre 3

Représentations graphiques et statistique descriptive univariée

3.1 Cas d'une variable quantitative

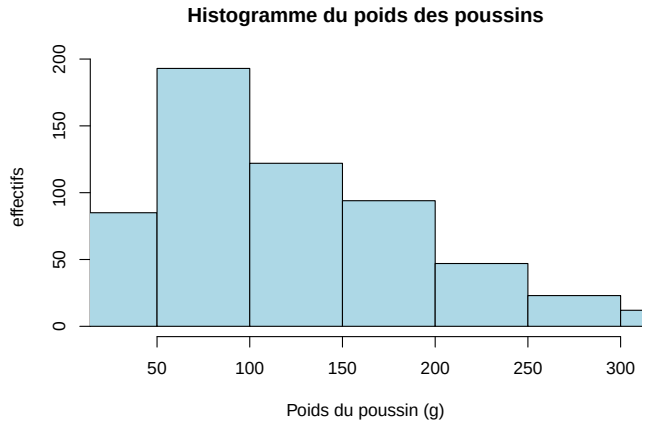
Reprendre le tableau de données *ChickWeight* et la variable quantitative *weight*.

Visualiser une distribution

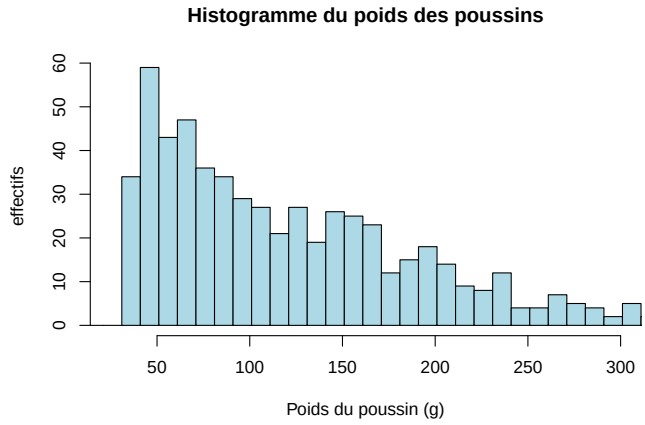
L'histogramme

L'histogramme permet de visualiser la distribution des valeurs d'une variable quantitative. On peut modifier le nombre de classes, les intervalles de classe ainsi que l'ordonnée.

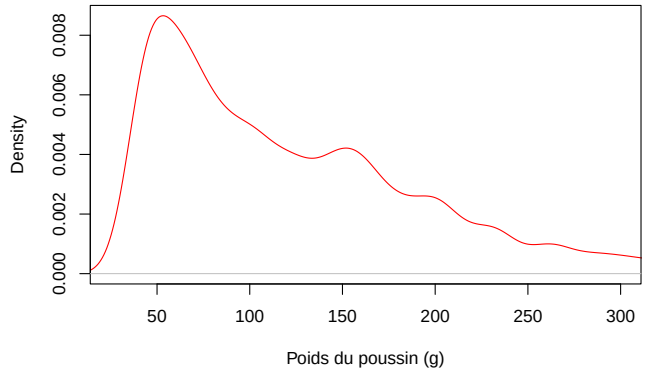
```
# Histogramme et effectifs
hist(ChickWeight$weight, xlab="Poids du poussin (g)", ylab="effectifs",
     col="lightblue", main="Histogramme du poids des poussins", freq=T,
     xlim=c(25,300))
```




```
hist(ChickWeight$weight, xlab="Poids du poussin (g)", ylab="effectifs",
     col="lightblue", main="Histogramme du poids des poussins", freq=T,
     breaks=seq(1,400,10), xlim=c(25,300))
```

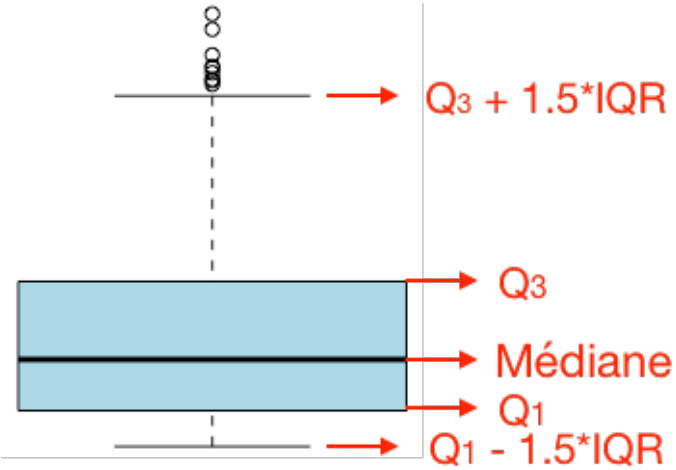



```
# Courbe de densité (noyau gaussien)
dens <- density(ChickWeight$weight, bw=10)
plot(dens, xlab="Poids du poussin (g)", main="", col="red", xlim=c(25,300))
```

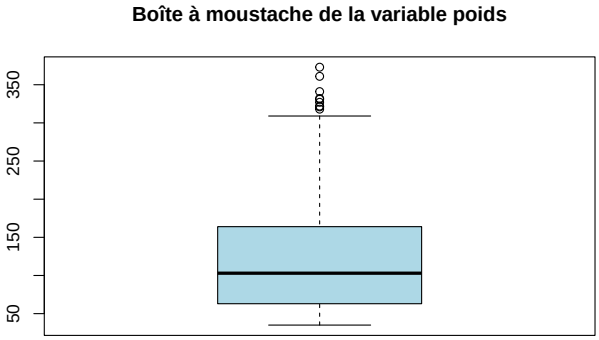


La boîte à moustaches

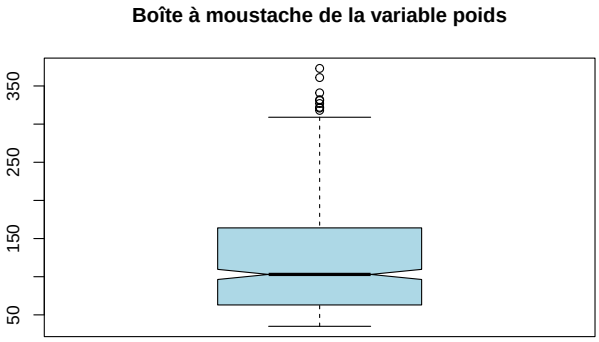
On peut représenter la dispersion d'une variable quantitative avec une boîte à moustache.




```
boxplot(ChickWeight$weight,
        main="Boîte à moustache de la variable poids", col="lightblue")
```




```
boxplot(ChickWeight$weight,
        main="Boîte à moustache de la variable poids", col="lightblue",
        notch=TRUE)
```



Si on veut extraire les statistiques du boxplot notamment les outliers, on peut utiliser la fonction *boxplot.stats()*

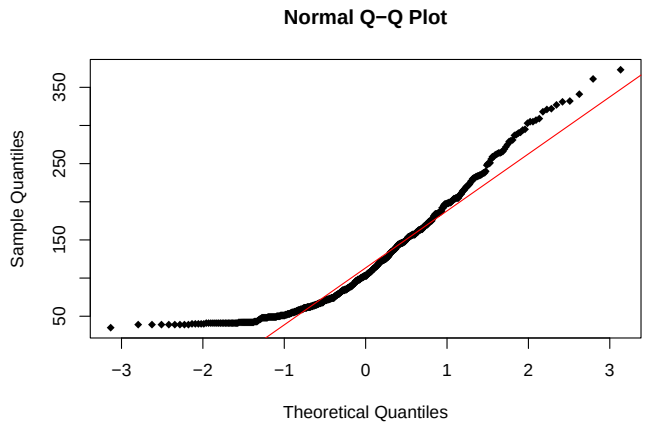
```
boxplot.stats(ChickWeight$weight)$out
```

```
## [1] 318 331 327 341 332 361 373 321 322
```

Diagramme quantile-quantile normal

Ce grahique consiste à comparer graphiquement l'ajustement d'une distribution donnée issue de la série de données à une distribution gaussienne par les quantiles des deux distributions.

```
qqnorm(ChickWeight$weight, pch=18)
qqline(ChickWeight$weight, col="red")
```

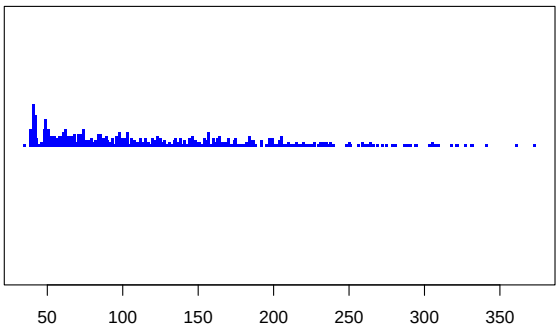


Visualiser une dispersion

Strip chart

On peut utiliser la fonction *stripchart()* pour visualiser la dispersion. Les valeurs sont ordonnées sur un axe. Lorsque les observations sont trop nombreuses, la visualisation devient compliquée. C'est pourquoi on peut utiliser l'option *method="jitter"* pour mieux visualiser les observations.

```
stripchart(ChickWeight$weight, pch=15, cex=0.3, col="blue", method="stack")
```




```
stripchart(ChickWeight$weight, pch=15, cex=0.3, col="blue", method="jitter")
```

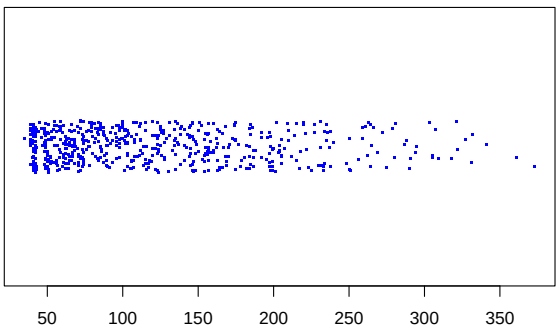


Diagramme tige-feuilles (histogramme de Tukey)

Ce graphique est très utilisé lorsqu'on a des petits effectifs. Il permet d'avoir une vision synthétique et rapide (le minimum, le maximum…).

```
stem(ChickWeight$weight, scale=3)
```

3.2 Cas d'une variable qualitative

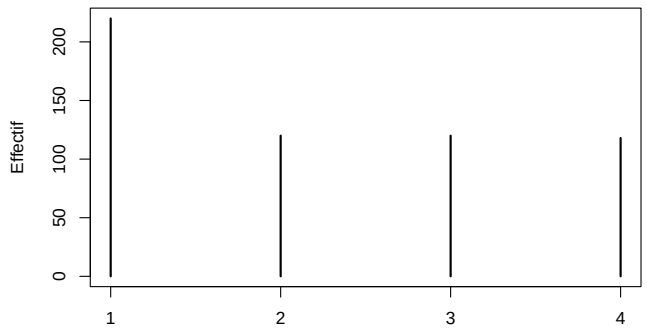
Reprenons le tableau de données *ChickWeight* et la variable qualitative *Diet*.

Visualiser une quantité

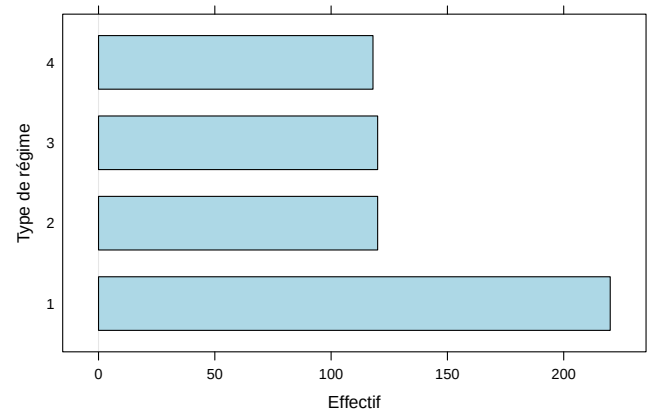
Diagramme en bâton

On va représenter la distribution des effectifs de la variable qualitative avec un diagramme en bâton. Pour chaque modalité, il y a une barre dont la hauteur est proportionnelle à la fréquence.

```
tableau_effec <- table(ChickWeight$Diet)
plot(tableau_effec, ylab="Effectif")
```




```
library(lattice)
barchart(xtabs( ~ Diet,data=ChickWeight), col="lightblue",
         ylab="Type de régime", xlab="Effectif")
```




```
barplot(xtabs( ~ Diet,data=ChickWeight), col="lightblue",
        ylab="Effectif", xlab="Type de régime")
```

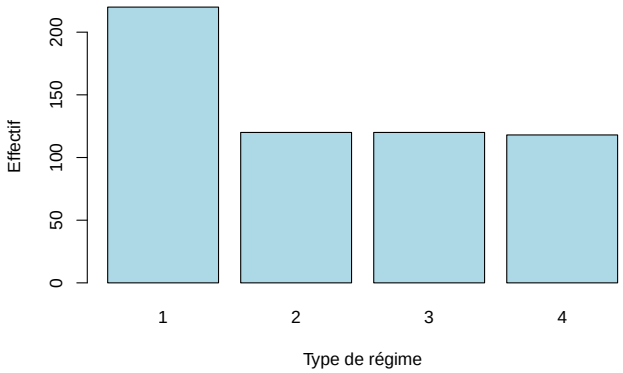
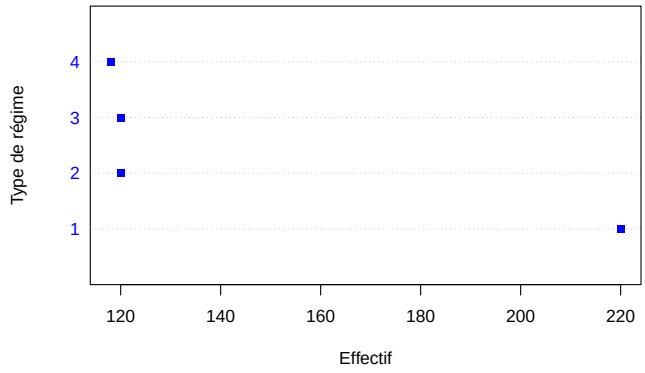


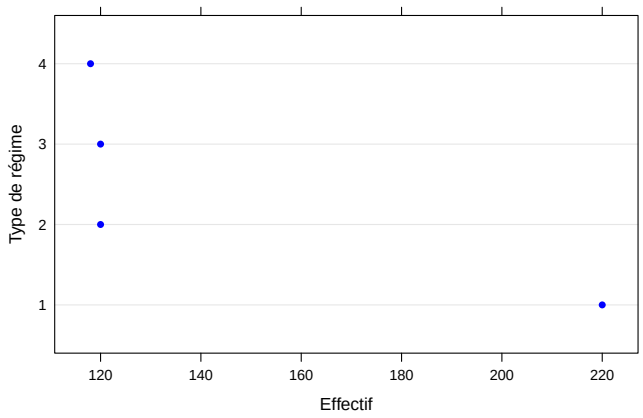
Diagramme de Cleveland

Un point correspond à l'effectif de chaque modalité avec en abscisse l'effectif et en ordonnée la modalité.

```
dotchart(as.matrix(table(sort(ChickWeight$Diet)))[,1], col="blue", pch=15,
        xlab="Effectif", ylab="Type de régime")
```




```
# Le diagramme de Cleveland avec l'utilisation de la fonction
# dotplot() du package lattice
dotplot(xtabs( ~ Diet,data=ChickWeight), col="blue",
        ylab="Type de régime", xlab="Effectif")
```

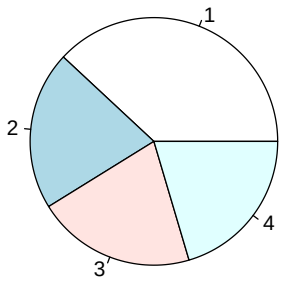


Visualiser une proportion

Diagramme circulaire

Pour effectuer un diagramme circulaire, on peut utiliser la fonction *pie()*.

```
pie(tableau_effec)
```



Ajoutons comme étiquette le pourcentage de chaque type de régime. Pour cela, il faut créer la variable pourcentage.

```
# Ajouter les pourcentages
(pourcentage <- (tableau_effec/length(ChickWeight$Diet))*100)
```

```
##
##        1        2        3        4
## 38.06228 20.76125 20.76125 20.41522
```

```
# Arrondir
(pourcentage <- round(pourcentage,2))
```

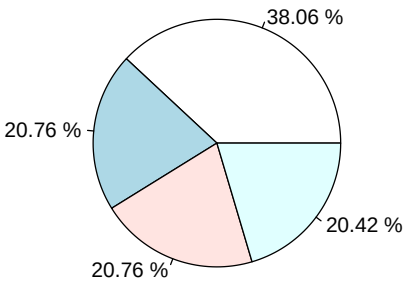
```
##
##     1     2     3     4
## 38.06 20.76 20.76 20.42
```

```
# Ajouter le symbole %
(pourcentage <- paste(pourcentage, "%"))
```

```
## [1] "38.06 %" "20.76 %" "20.76 %" "20.42 %"
```

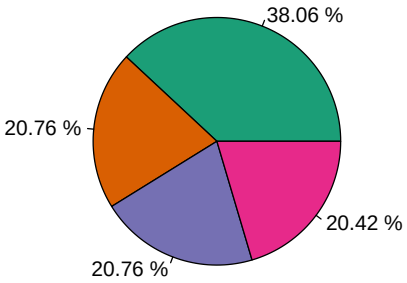
Effectuons à nouveau le diagramme circulaire.

```
pie(tableau_effec, labels=pourcentage)
```



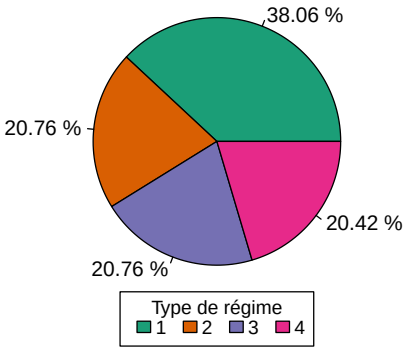
Ajoutons une couleur pour chaque type de régime.

```
library(RColorBrewer)
# display.brewer.all() pour voir les palettes
pie(tableau_effec, labels=pourcentage, col=brewer.pal(n = 4, name = "Dark2"))
```



Ajoutons une légende.

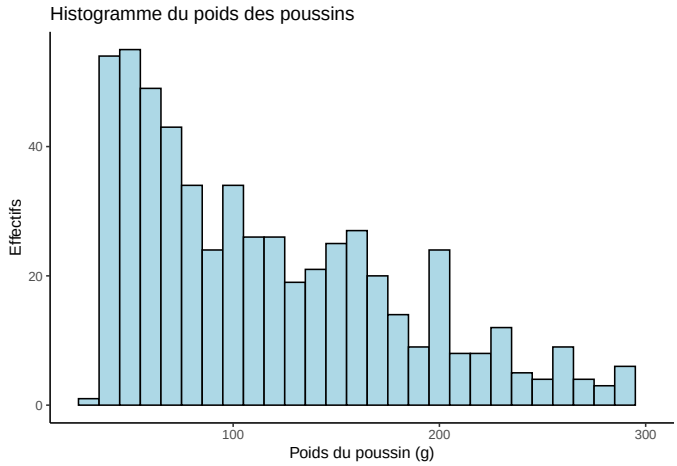
```
pie(tableau_effec, labels=pourcentage, col=brewer.pal(n = 4, name = "Dark2"))
legend("top", levels(ChickWeight$Diet), inset = c(0.1, 0.95),
       x.intersp=0.3, y.intersp=0.7, cex=0.9, horiz=TRUE, xpd = TRUE,
       title="Type de régime", fill=brewer.pal(n = 4, name = "Dark2"))
```



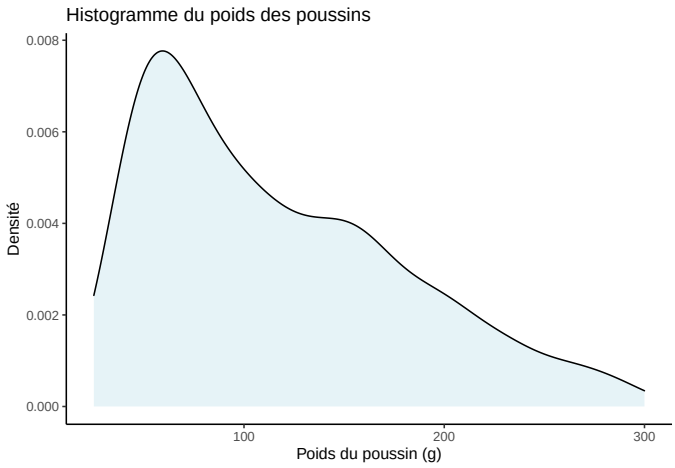
3.3 Utilisation du package tidyverse (package ggplot2)

Cas d'une variable quantitative

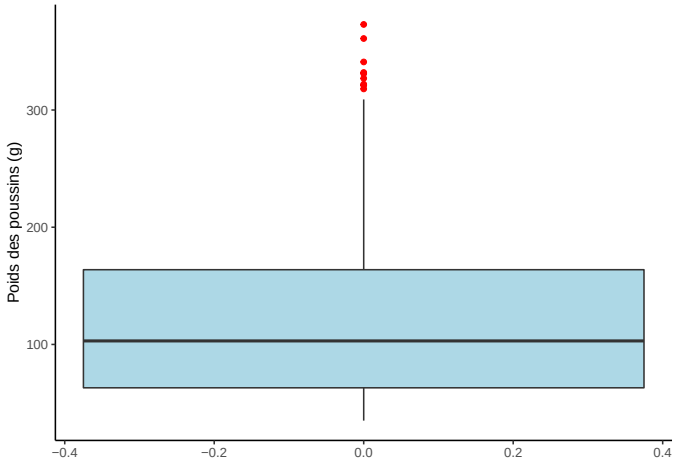
```
library(tidyverse)
# Histogramme
ggplot(data=ChickWeight, aes(weight)) +
  geom_histogram(color="black", fill="lightblue", binwidth=10) +
  xlim(25,300) + xlab("Poids du poussin (g)") + ylab("Effectifs") +
  ggtitle('Histogramme du poids des poussins') + theme_classic()
```



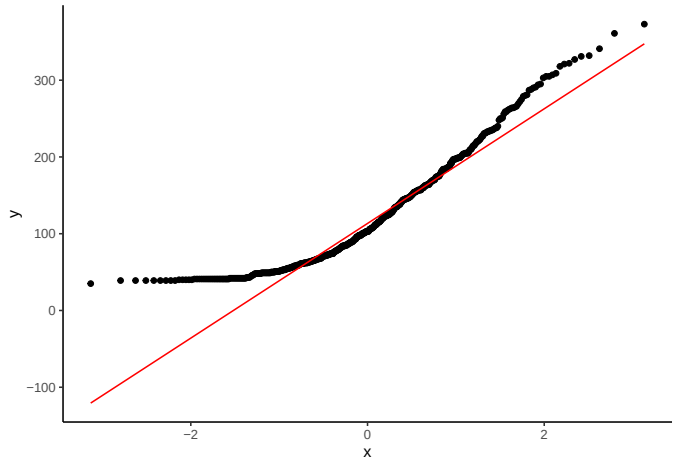
```
# Courbe de distribution
ggplot(data=ChickWeight, aes(weight)) +
  geom_density(alpha=.3, color="black", fill="lightblue") +
   xlim(25,300) +
  xlab("Poids du poussin (g)") +  ylab("Densité") +
  ggtitle('Histogramme du poids des poussins') + theme_classic()
```




```
# Boxplot
ggplot(data=ChickWeight, aes(y=weight)) +
  geom_boxplot(fill="lightblue", outlier.colour="red") +
  ylab("Poids des poussins (g)") + theme_classic()
```




```
# Graphique quantile-quantile
ggplot(ChickWeight, aes(sample = weight)) + stat_qq() +
  stat_qq_line(col="red") + theme_classic()
```



Cas d'une variable qualitative

Diagramme en bâton

Effectuons un diagramme en bâton en utilisant la couche *geom_bar* :

```
# Diagramme en bâton
ggplot(data=ChickWeight, aes(x=Diet)) +
  geom_bar(fill="lightblue") + xlab("Type de régime") + ylab("Effectifs") +
  theme_classic()
```

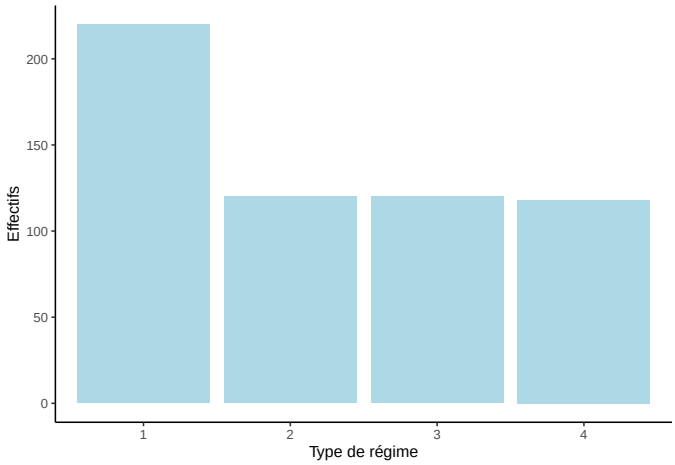


Diagramme de Cleveland

Effectuons un diagramme de Cleveland.

```
# Diagramme de Cleveland
ggplot(data=ChickWeight, aes(x=Diet)) +
  geom_point(stat="count", pch=15, col="blue") +
  xlab("Type de régime") + ylab("Effectifs") + coord_flip() + theme_classic()
```

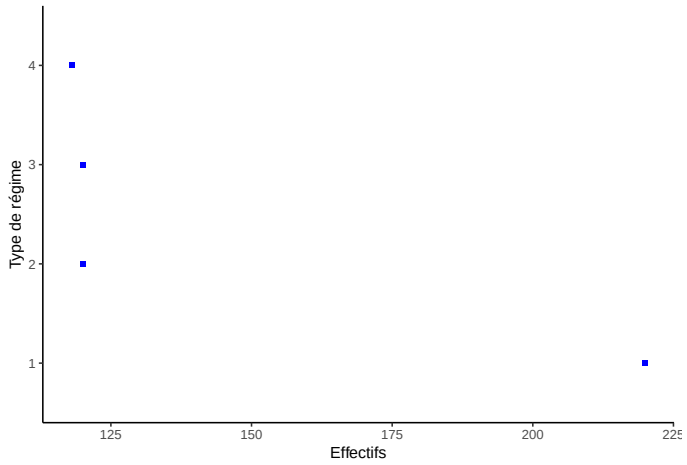


Diagramme circulaire

Essayons d'effectuons un diagramme circulaire. Pour cela, on va créer un tableau avec les effectifs et les pourcentages de chaque régime. On peut utiliser le package *dplyr* déjà chargé avec le package *tidyverse*.

On ajoute la colonne n donnant l'effectif de chaque régime.

```
(datapie <- ChickWeight %>% count(Diet))
```

```
##   Diet   n
## 1    1 220
## 2    2 120
## 3    3 120
## 4    4 118
```

On ajoute la colonne prop donnant la proportion de chaque régime.

```
(datapie <- datapie %>% mutate(prop = prop.table(n)))
```

```
##   Diet   n      prop
## 1    1 220 0.3806228
```

```
## 2    2 120 0.2076125
## 3    3 120 0.2076125
## 4    4 118 0.2041522
```

On ajoute la colonne pourc donnant le pourcentage de chaque régime.

```
(datapie <- datapie %>% mutate(pourc=prop*100))
```

```
##   Diet   n      prop    pourc
## 1    1 220 0.3806228 38.06228
## 2    2 120 0.2076125 20.76125
## 3    3 120 0.2076125 20.76125
## 4    4 118 0.2041522 20.41522
```

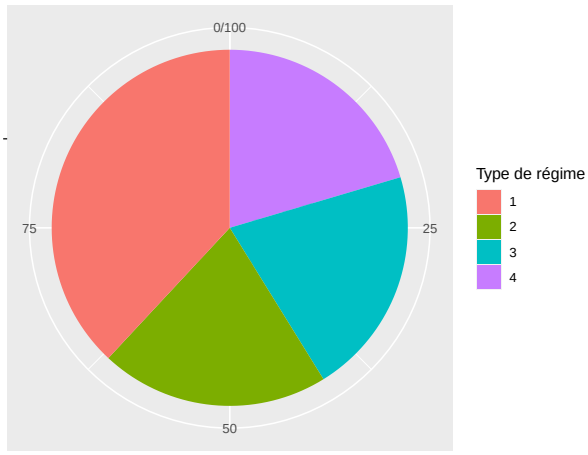
On trie le tableau par ordre croissant de la variable pourc.

```
(datapie <- datapie %>% arrange(desc(Diet)))
```

```
##   Diet   n      prop    pourc
## 1    4 118 0.2041522 20.41522
## 2    3 120 0.2076125 20.76125
## 3    2 120 0.2076125 20.76125
## 4    1 220 0.3806228 38.06228
```

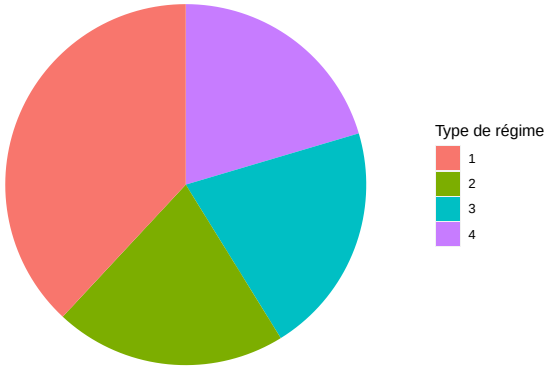
Effectuons le diagramme circulaire.

```
# Diagramme circulaire
ggplot(data = datapie, aes(x = "", y = pourc, fill = Diet)) +
  geom_bar(stat = "identity") + coord_polar("y", start = 0) +
  labs(fill="Type de régime",  x=NULL, y=NULL)
```



Enlevons les annotations grises.

```
ggplot(data = datapie, aes(x = "", y = pourc, fill = Diet)) +
  geom_bar(stat = "identity") + coord_polar("y", start = 0) +
  labs(fill="Type de régime",  x=NULL, y=NULL) +
  theme(panel.grid = element_blank(),
        axis.ticks = element_blank(),
        axis.text.x=element_blank(),
        panel.background = element_rect(fill="white"))
```

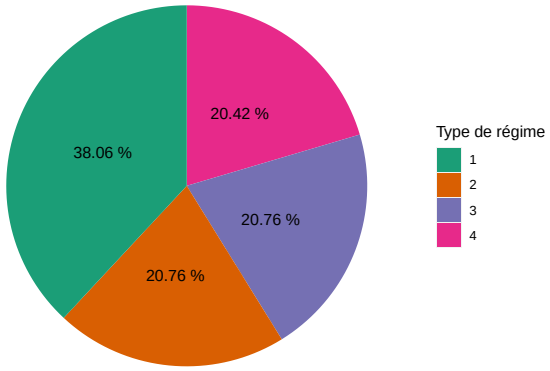


Ajoutons dans le tableau la colonne labpos correspondant à la position de chaque pourcentage sur le diagramme.

```
datapie <- datapie %>% mutate(labpos = c(10,31,52,81))
```

Effectuons le diagramme circulaire en ajoutant les pourcentages et en modifiant les couleurs associées au type de régime :

```
ggplot(data = datapie, aes(x = "", y = pourc, fill = Diet)) +
  geom_bar(stat = "identity") + coord_polar("y", start = 0) +
  scale_fill_brewer(palette = "Dark2") +
  labs(fill="Type de régime",  x=NULL, y=NULL) +
  theme(panel.grid = element_blank(),
        axis.ticks = element_blank(),
        axis.text.x=element_blank(),
        panel.background = element_rect(fill="white")) +
  geom_text(aes(y = labpos,label = paste(round(pourc,2),"%")))
```



3.4 TP

— Représenter graphiquement les variables *Time* et *Chick*.

— Reprendre le jeu de données *corvus*. Décrire les variables *habitat* et *phylog*.

— Reprendre le jeu de données *iris*. Effectuer un type de représentation pour chacune des variables.

3.5 A VOUS DE JOUER !

— Effectuer des représentations graphiques de chacune des variables de votre tableau de données.

Chapitre 4

Résumé statistique à l'aide de packages R

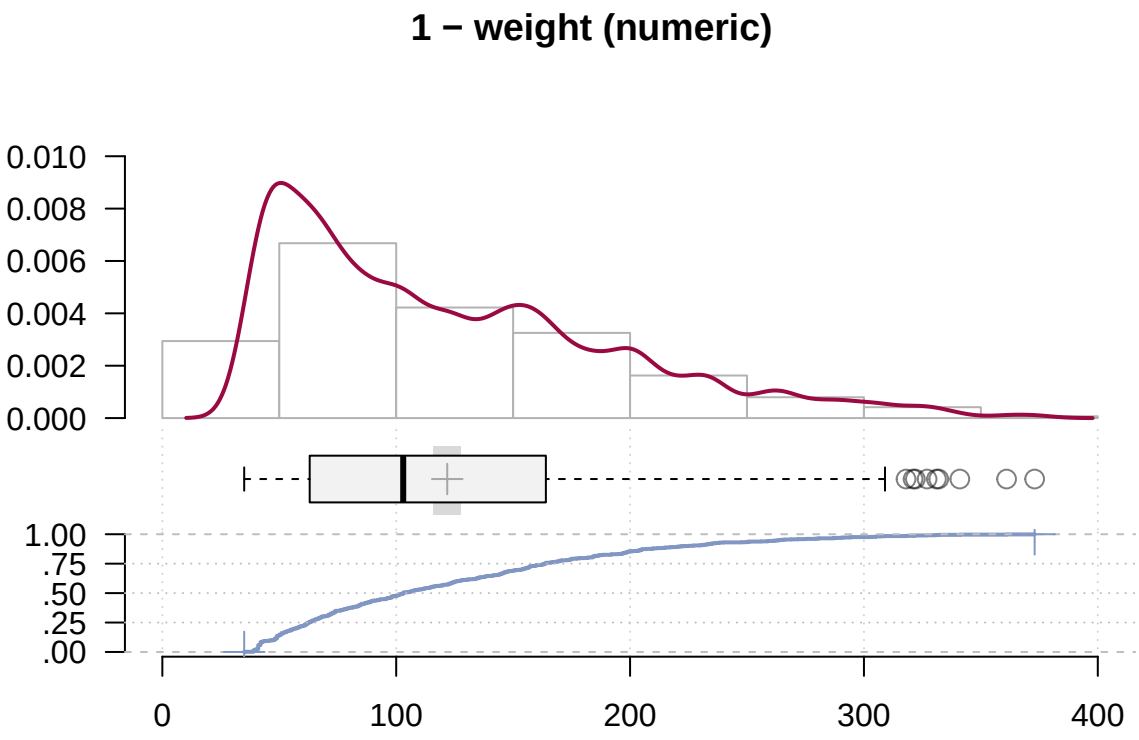
4.1 Utilisation du package DescTools et de la fonction *Desc()*

Les références relatives au package **DescTools** sont mises à la fin du document (al. (2020)).

```
library(DescTools)
print(Desc(ChickWeight))
```

```
## ------------------------------------------------------------------------
## Describe ChickWeight (nfnGroupedData, nfGroupedData, groupedData, data.frame): :
##   Time
##   Body weight
##
##
##   Body weight
##
## data frame:  578 obs. of  4 variables
##      578 complete cases (100.0%)
##
##   Nr  ColName  Class            NAs  Levels
##   1   weight   numeric          .
##   2   Time     numeric          .
##   3   Chick    ordered, factor  .    (50): 1-18, 2-16, 3-15, 4-13, 5-9, ...
##   4   Diet     factor           .    (4): 1-1, 2-2, 3-3, 4-4
```

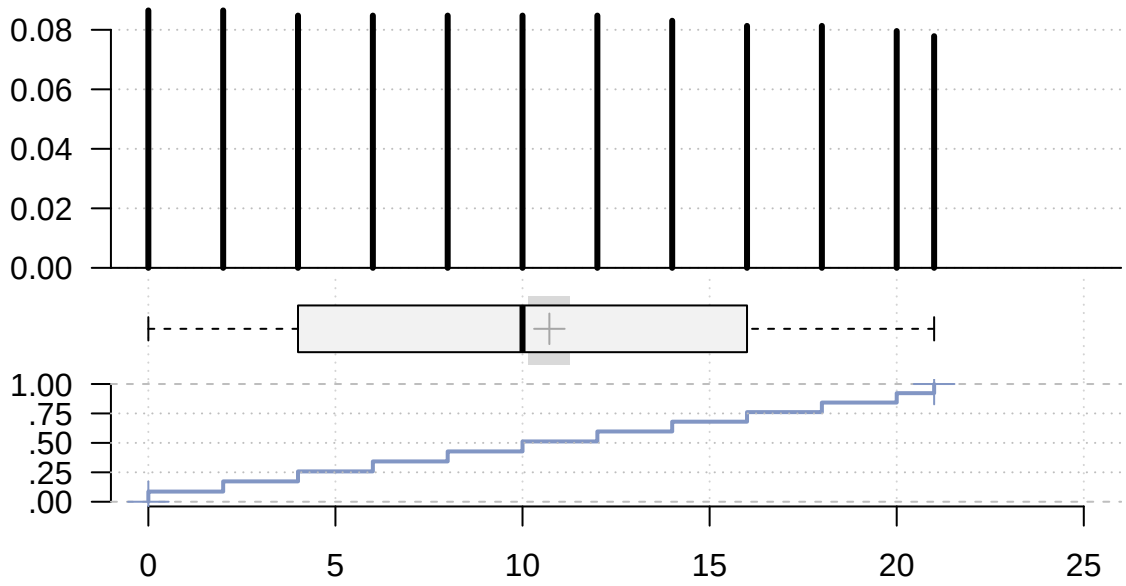
```
##
##
## ------------------------------------------------------------------------------
## 1 - weight (numeric)
##
##   length       n    NAs  unique      0s    mean  meanCI'
##      578     578      0     212       0  121.82  116.01
##           100.0%   0.0%           0.0%          127.62
##
##      .05     .10    .25  median     .75     .90     .95
##    41.00   47.70  63.00  103.00  163.75  223.60  264.00
##
##    range      sd  vcoef     mad     IQR    skew    kurt
##   338.00   71.07   0.58   69.68  100.75    0.96    0.34
##
## lowest : 35.0, 39.0 (8), 40.0 (5), 41.0 (20), 42.0 (15)
## highest: 331.0, 332.0, 341.0, 361.0, 373.0
##
## ' 95%-CI (classic)
```




```
## ------------------------------------------------------------------------------
## 2 - Time (numeric)
##
##   length       n    NAs  unique     0s   mean  meanCI'
##      578     578      0      12     50  10.72   10.17
##           100.0%   0.0%           8.7%          11.27
##
##      .05     .10     .25  median    .75    .90     .95
##     0.00    2.00    4.00   10.00  16.00  20.00   21.00
##
##    range      sd  vcoef     mad    IQR   skew    kurt
##    21.00    6.76   0.63    8.90  12.00  -0.02   -1.26
##
##
##     value  freq  perc  cumfreq  cumperc
## 1       0    50  8.7%       50     8.7%
## 2       2    50  8.7%      100    17.3%
## 3       4    49  8.5%      149    25.8%
```

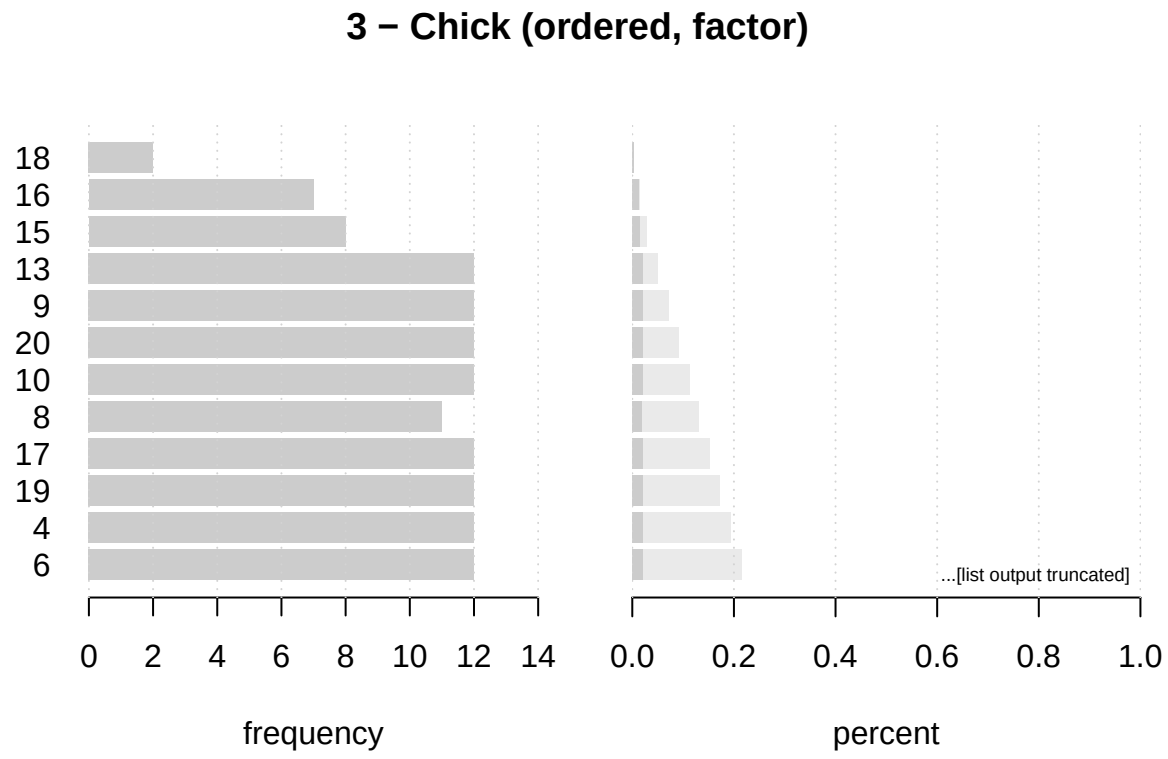
```
## 4       6    49  8.5%      198    34.3%
## 5       8    49  8.5%      247    42.7%
## 6      10    49  8.5%      296    51.2%
## 7      12    49  8.5%      345    59.7%
## 8      14    48  8.3%      393    68.0%
## 9      16    47  8.1%      440    76.1%
## 10     18    47  8.1%      487    84.3%
## 11     20    46  8.0%      533    92.2%
## 12     21    45  7.8%      578   100.0%
##
## ' 95%-CI (classic)
```

2 − Time (numeric)



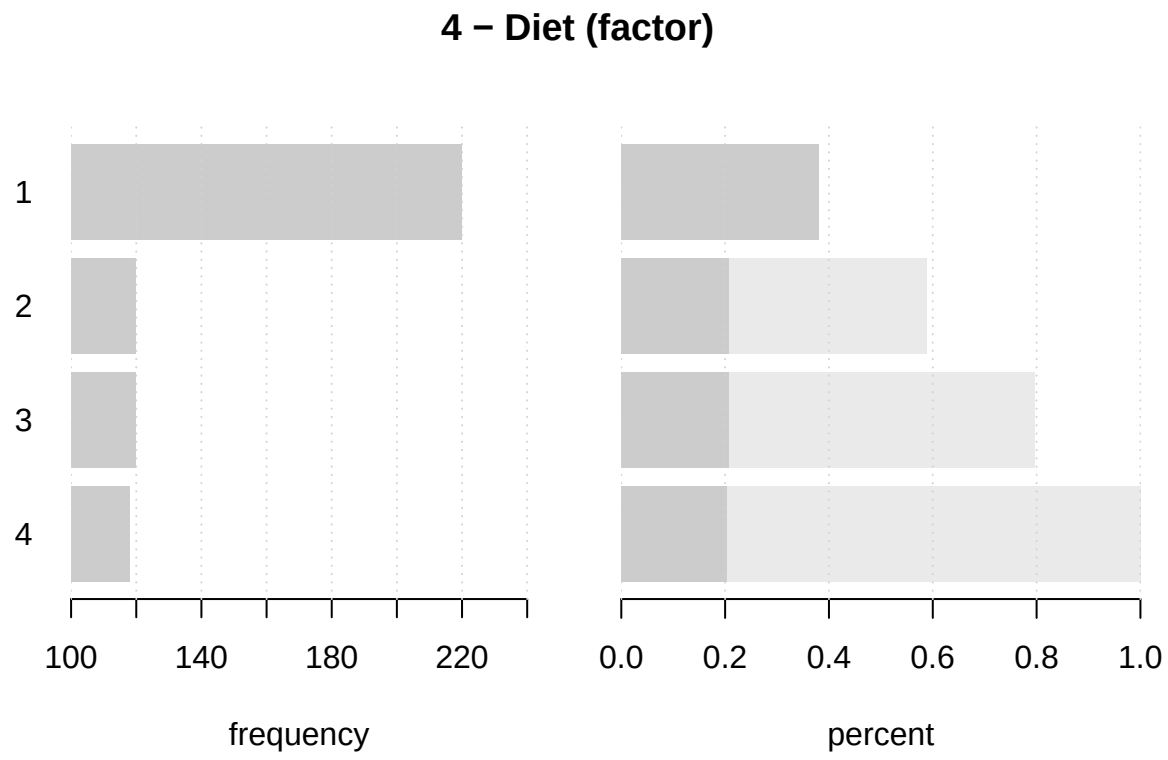

```
## ------------------------------------------------------------------------
## 3 - Chick (ordered, factor)
##
##   length      n    NAs unique levels  dupes
##      578    578      0     50     50      y
##          100.0%   0.0%
```

```
##
##     level  freq  perc  cumfreq  cumperc
## 1      18     2  0.3%        2     0.3%
## 2      16     7  1.2%        9     1.6%
## 3      15     8  1.4%       17     2.9%
## 4      13    12  2.1%       29     5.0%
## 5       9    12  2.1%       41     7.1%
## 6      20    12  2.1%       53     9.2%
## 7      10    12  2.1%       65    11.2%
## 8       8    11  1.9%       76    13.1%
## 9      17    12  2.1%       88    15.2%
## 10     19    12  2.1%      100    17.3%
## 11      4    12  2.1%      112    19.4%
## 12      6    12  2.1%      124    21.5%
## ... etc.
##  [list output truncated]
```




```
## ------------------------------------------------------------------------
```

```
## 4 - Diet (factor)
##
##   length      n    NAs unique levels  dupes
##      578    578      0      4      4      y
##          100.0%   0.0%
##
##    level  freq   perc  cumfreq  cumperc
## 1      1   220  38.1%      220    38.1%
## 2      2   120  20.8%      340    58.8%
## 3      3   120  20.8%      460    79.6%
## 4      4   118  20.4%      578   100.0%
```



4.2 Utilisation du package summarytools

Les références relatives au package **summarytools** sont mises à la fin du document (Comtois (2020)).

```
library(summarytools)
descr(ChickWeight)
```

```
## Descriptive Statistics
## ChickWeight
## N: 578
##
##                       Time   weight
## ---------------- -------- --------
##             Mean    10.72   121.82
##          Std.Dev     6.76    71.07
##              Min     0.00    35.00
##               Q1     4.00    63.00
##           Median    10.00   103.00
##               Q3    16.00   164.00
##              Max    21.00   373.00
##              MAD     8.90    69.68
##              IQR    12.00   100.75
##               CV     0.63     0.58
##         Skewness    -0.02     0.96
##      SE.Skewness     0.10     0.10
##         Kurtosis    -1.26     0.34
##          N.Valid   578.00   578.00
##        Pct.Valid   100.00   100.00
```

```
view(dfSummary(ChickWeight))
```

De nombreux autres packages existent comme le package *DataExplorer*.

Chapitre 5

Statistiques descriptives bivariées

Le but du chapitre est d'étudier la relation entre deux variables qu'elle soit quantitative ou qualitative.

5.1 Cas de deux variables qualitatives

Il faut en premier lieu établir un tableau de contingence c'est-à-dire un tableau croisé 2*2 de deux variables qualitatives.

Distribution marginale

Distribution des effectifs ou des fréquences de chaque variable qualitative. Pour l'obtenir, on somme les effectifs ligne par ligne ou colonne par colonne du tableau de contingence. Sous R, on utilisera la fonction *table()*.

Distribution conditionnelle

Distribution d'une variable qualitative en fonction de l'autre variable qualitative réduite à une seule modalité. Elle ne concerne qu'une partie des individus de la population ou de l'échantillon. Sous R, on utilisera la fonction *prop.table()*.

Reprenons le jeu de données *corvus*.

```
# Tableau de contingence
contingence <- table(corvus$habitat, corvus$phylog)
# Somme des lignes
margin.table(contingence,1)
```

```
##
## clos open
##   11   17
```

```
# Somme des colonnes
margin.table(contingence,2)
```

```
##
##  amer orien  pale
##     6    13     9
```

```
# Somme totale
margin.table(contingence)
```

```
## [1] 28
```

```
addmargins(contingence)
```

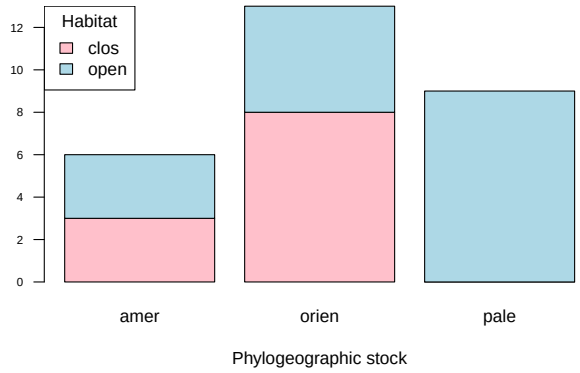
```
##
##        amer orien pale Sum
##   clos    3     8    0  11
##   open    3     5    9  17
##   Sum     6    13    9  28
```

```
# Distribution conditionnelle
prop.table(contingence)
```

```
##
##             amer     orien      pale
##   clos 0.1071429 0.2857143 0.0000000
##   open 0.1071429 0.1785714 0.3214286
```

Effectuons une représentation graphique.

```
# Tableau de contingence
contingence <- table(corvus$habitat, corvus$phylog)
barplot(contingence, col=c("pink","lightblue"), beside = F, las=1,
        cex.axis=0.7, xlab="Phylogeographic stock")
legend("topleft", legend=row.names(contingence), title="Habitat",
       fill=c("pink", "lightblue"))
```



5.2 Cas d'une variable qualitative et d'une variable quantitative

Dans ce cas, on cherche à voir la variabilité d'une variable quantitative selon chaque modalité de la variable qualitative. On peut effectivement calculer des indicateurs statistiques pour chaque modalité. Reprenons le jeu de données *corvus*.

```
tapply(corvus$wing , corvus$habitat , mean)
```

```
##     clos     open
## 300.1818 368.0588
```

```
tapply(corvus$wing , corvus$habitat , median)
```

```
## clos open
##  319  370
```

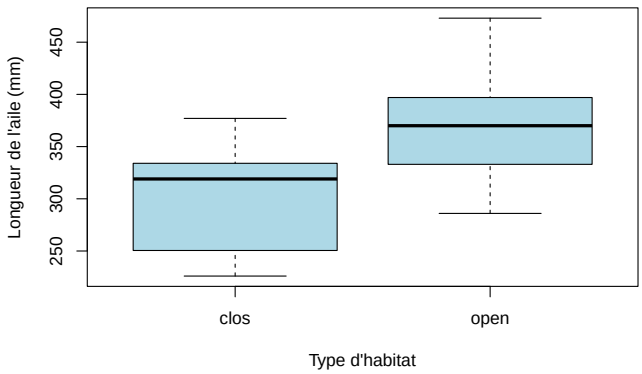
```
tapply(corvus$wing , corvus$habitat , sd)
```

```
##     clos     open
## 53.70627 52.91086
```

Visualisation de distributions

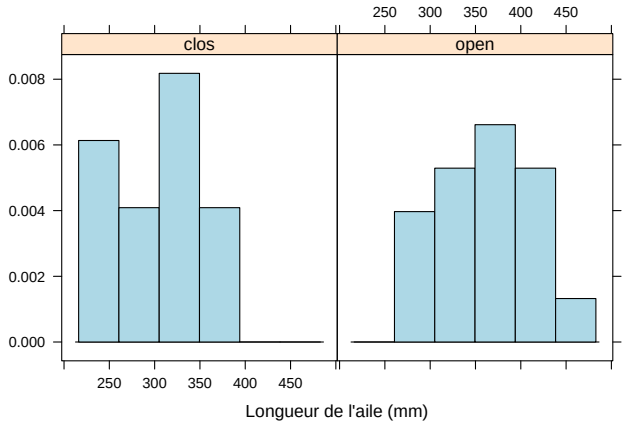
Boîtes à moustache

```
# Boite à moustache
boxplot(corvus$wing ~ corvus$habitat, col="lightblue",
        xlab="Type d'habitat", ylab="Longueur de l'aile (mm)")
```



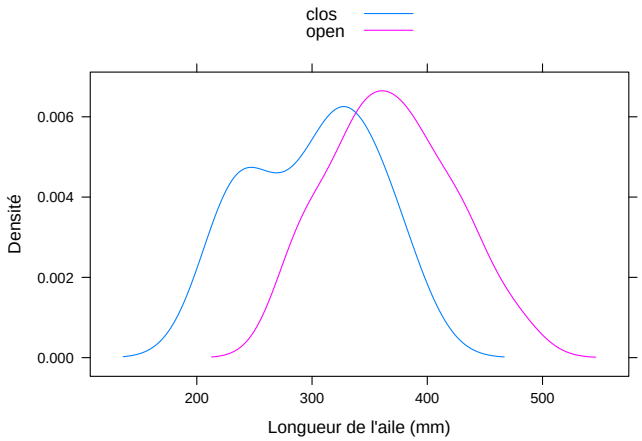
Histogrammes

```
# Histogramme de la longueur d'aile en fonction de l'habitat (package lattice)
histogram( ~ corvus$wing | corvus$habitat, type="density",
           xlab="Longueur de l'aile (mm)", ylab="", col="lightblue")
```



Courbes de densité

```
# Densité de la longueur d'aile en fonction de l'habitat
densityplot( ~ corvus$wing, groups=corvus$habitat, type="density",
            xlab="Longueur de l'aile (mm)", ylab="Densité", auto.key=TRUE)
```

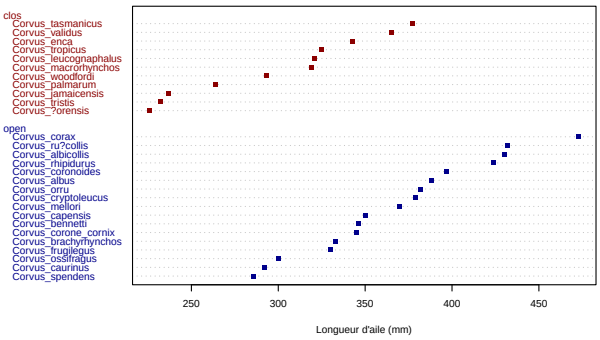


Visualisation de quantités

On peut également construire un diagramme de Cleveland en fonction de la modalité de la variable qualitative.

```
couleur <- c("darkred", "darkblue")
# Réordonner les données selon la variable wing
corvus2 <- corvus[order(corvus$wing), ]
dotchart(corvus2$wing, labels = row.names(corvus2),
```

```
        gcolor = couleur, color = couleur[corvus2$habitat],
        groups = corvus2$habitat, cex=0.6, pch = 15,
        xlab = "Longueur d'aile (mm)")
```

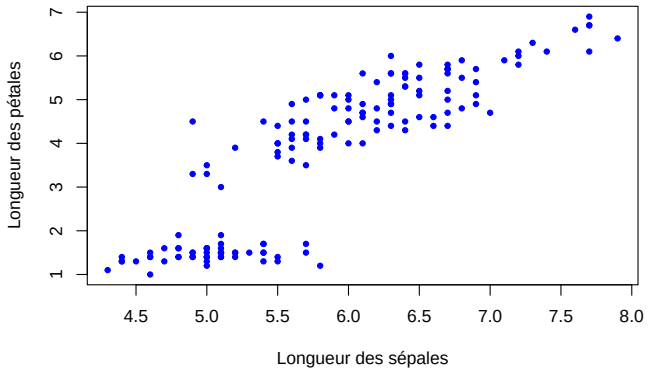


5.3 Cas de deux variables quantitatives

Le nuage de points

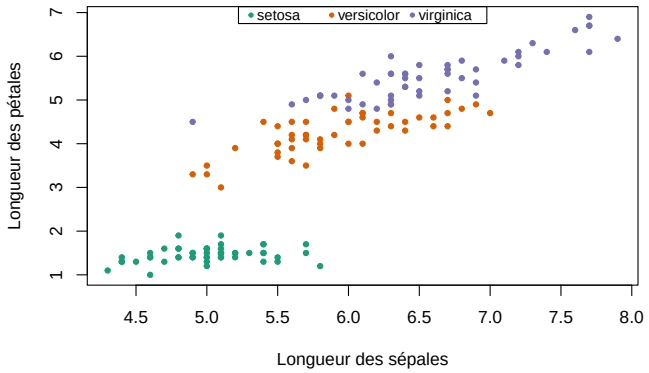
Reprenons l'exemple du jeu de données *iris*. Traçons la longueur des pétales en fonction de la longueur des sépales.

```
plot(iris$Sepal.Length,iris$Petal.Length, xlab="Longueur des sépales",
ylab="Longueur des pétales", pch=20, col="blue")
```



On peut colorier les points en fonction de l'espèce des iris.

```
couleur <- brewer.pal(n = 3, name = "Dark2")
plot(iris$Sepal.Length,iris$Petal.Length, pch=20,
     xlab="Longueur des sépales", ylab="Longueur des pétales",
     col=couleur[iris$Species])
legend("top", legend=levels(iris$Species), pch = 20, col=couleur,
       cex=0.8, horiz = T, x.intersp = 0.5, y.intersp = 0.01)
```



Pour tester si il y a une liaison (positive ou négative) entre deux variable quantitatives comme par exemple la variable *Sepal.Length* et *Petal.Length*, on peut calculer le coefficient de corrélation entre les deux variables. Le coefficient de corrélation est compris entre -1 et 1. Si le coefficient est proche de 0, il n'existe aucune corrélation. En revanche, si le coefficient est proche de 1 (resp -1), il existe une corrélation positive (resp négative).

```
cor(iris$Sepal.Length, iris$Petal.Length)
```

```
## [1] 0.8717538
```

Attention, appliquer un test de corrélation de Pearson suppose que les conditions d'application du test sont remplies. Les tests statistiques seront abordés lors du prochain atelier.

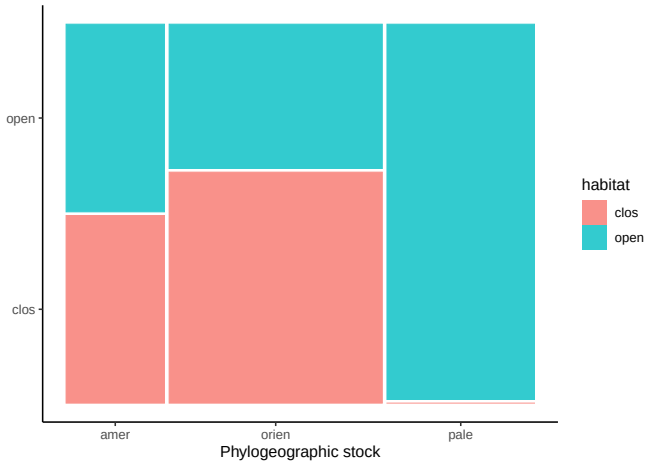
5.4 Quelques graphiques avec le package tidyverse (package ggplot2)

Les graphiques précédents peuvent être effectués avec le package **ggplot2**.

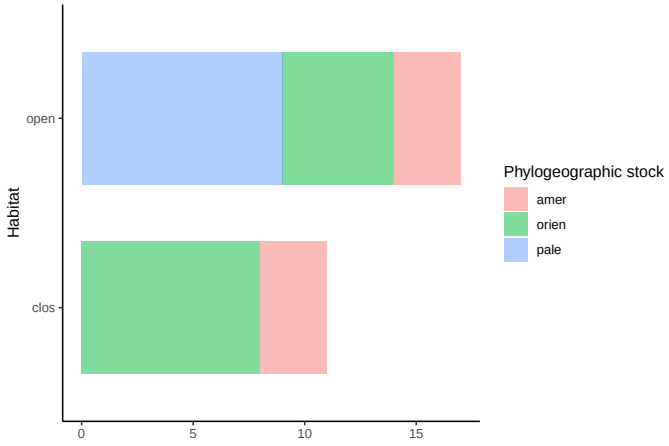
Cas de deux variables qualitatives

```
# Mosaic plot
library(ggmosaic)
ggplot(corvus)+
    geom_mosaic(aes(x=product(habitat,phylog), fill=habitat))+
     xlab("Phylogeographic stock") + ylab("") + theme_classic()
```

```
## Warning: `unite_()` was deprecated in tidyr 1.2.0.
## Please use `unite()` instead.
```

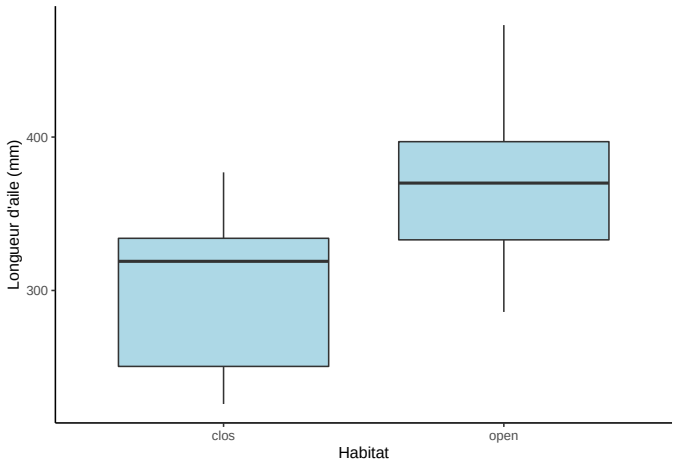



```
# Diagramme en barre
ggplot(data=corvus, aes(x=habitat, fill=phylog)) +
  geom_bar(width=0.7, alpha=0.5) + ylab("Effectif") +
  xlab("Habitat")  + ylab("") + coord_flip() +
  scale_fill_discrete("Phylogeographic stock") +
  theme_classic()
```

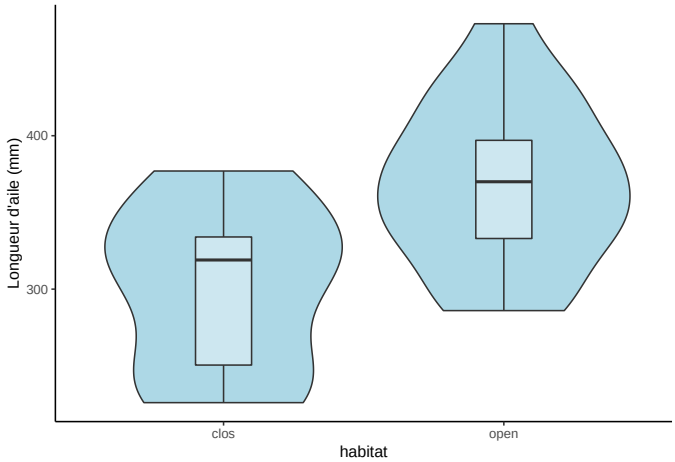


Cas d'une variable qualitative et d'une variable quantitative

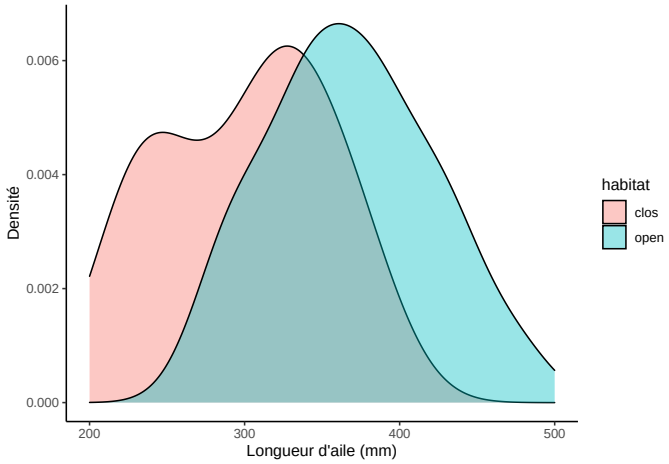
```
# Boxplot
ggplot(data=corvus, aes(x=habitat, y=wing)) +
  geom_boxplot(fill="lightblue", outlier.colour="red") +
  xlab("Habitat") + ylab("Longueur d'aile (mm)") +
  theme_classic()
```



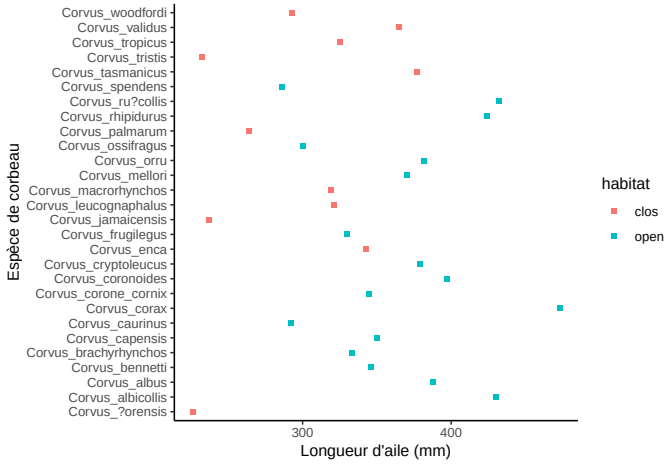

```
# Violin plot
ggplot(data=corvus, aes(x=habitat, y=wing)) +
  geom_violin(fill="lightblue") +
  geom_boxplot(width=0.2, alpha=0.4) +
  ylab("Longueur d'aile (mm)") +
  guides(fill="none") +
  theme_classic()
```



```
# Densité
ggplot(corvus, aes(x=wing, fill=habitat)) +
  geom_density(alpha=.4) +
  xlab("Longueur d'aile (mm)") + ylab("Densité") + xlim(200,500) +
  theme_classic()
```



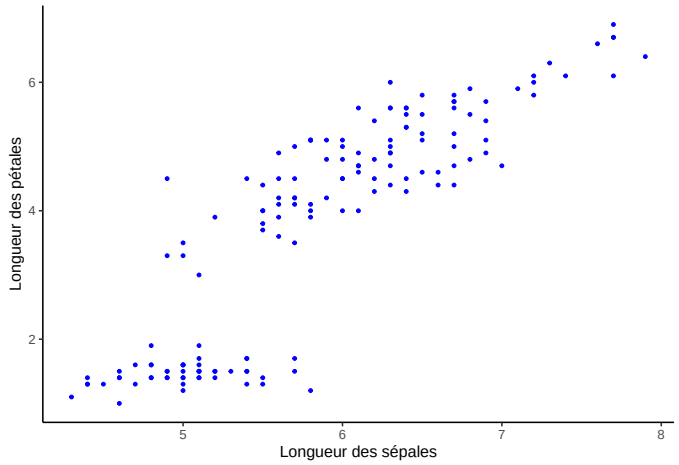

```
# Diagramme de Cleveland
ggplot(corvus2, aes(x = wing, y = rownames(corvus2), habitat)) +
  xlab("Longueur d'aile (mm)") +
  geom_point(pch = 15, aes(color = habitat)) +
  ylab("Espèce de corbeau")  + theme_classic()
```



Cas de plusieurs variables quantitatives

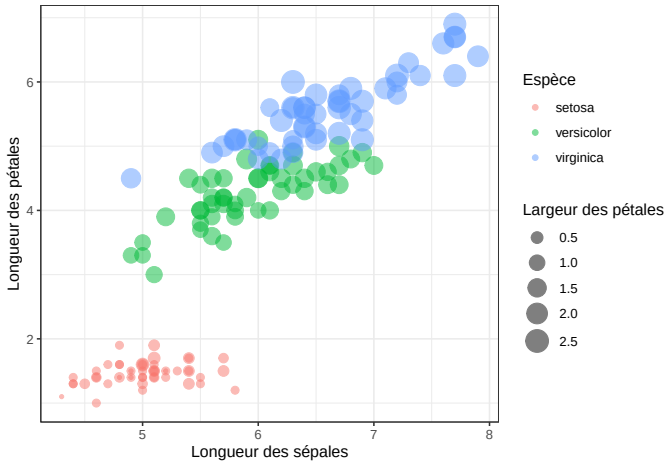
Nuage de points

```
ggplot(data=iris, aes(x=Sepal.Length, y=Petal.Length)) +
  geom_point(pch=20, col="blue") +
  xlab("Longueur des sépales") +
  ylab("Longueur des pétales") +
  theme_classic()
```



Graphique à bulles

```
ggplot(data=iris, aes(x=Sepal.Length, y=Petal.Length)) +
  geom_point(alpha=0.5, aes(size=Petal.Width, color=Species)) +
  xlab("Longueur des sépales") + ylab("Longueur des pétales") +
  scale_size(name="Largeur des pétales", range = c(1, 7)) +
  scale_color_discrete(name="Espèce") +
  theme_bw()
```



5.5 TP

— Charger le jeu de données *banque* du package **ade4**.

— Représenter graphiquement la relation entre la catégorie socio-professionnelle et l'âge.

— Charger le jeu de données *InsectSprays* disponible dans les packages de base de **R**.

— Représenter graphiquement le nombre d'insectes en fonction du type d'insecticide.

5.6 A VOUS DE JOUER !

— Effectuer l'analyse descriptive de votre jeu de données.

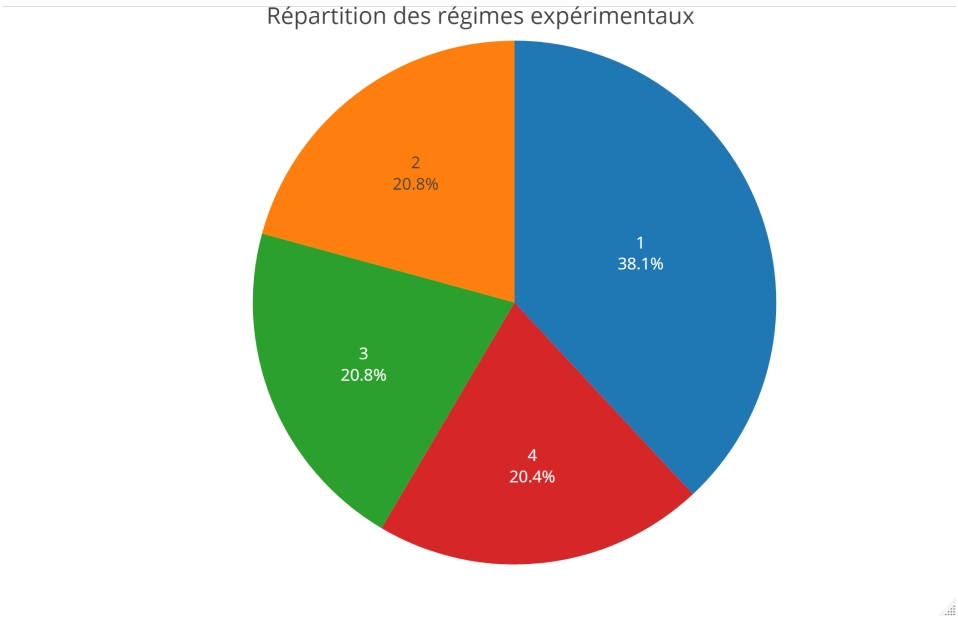
Chapitre 6

Zoom sur le package plotly

Ce package permet d'effectuer des graphiques interactifs (Sievert (2020)).

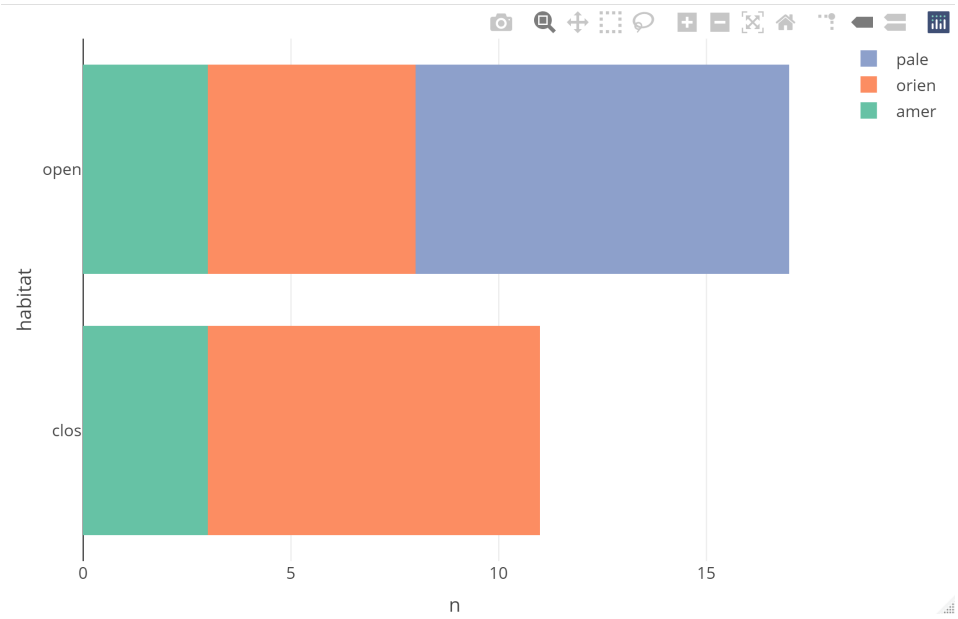
6.1 Diagramme en camembert

```
library(plotly)
pie_graph <- plot_ly(ChickWeight, labels = ~Diet, type = 'pie',
                     textinfo = 'label+percent', showlegend = FALSE) %>%
  layout(title = 'Répartition des régimes expérimentaux')
```



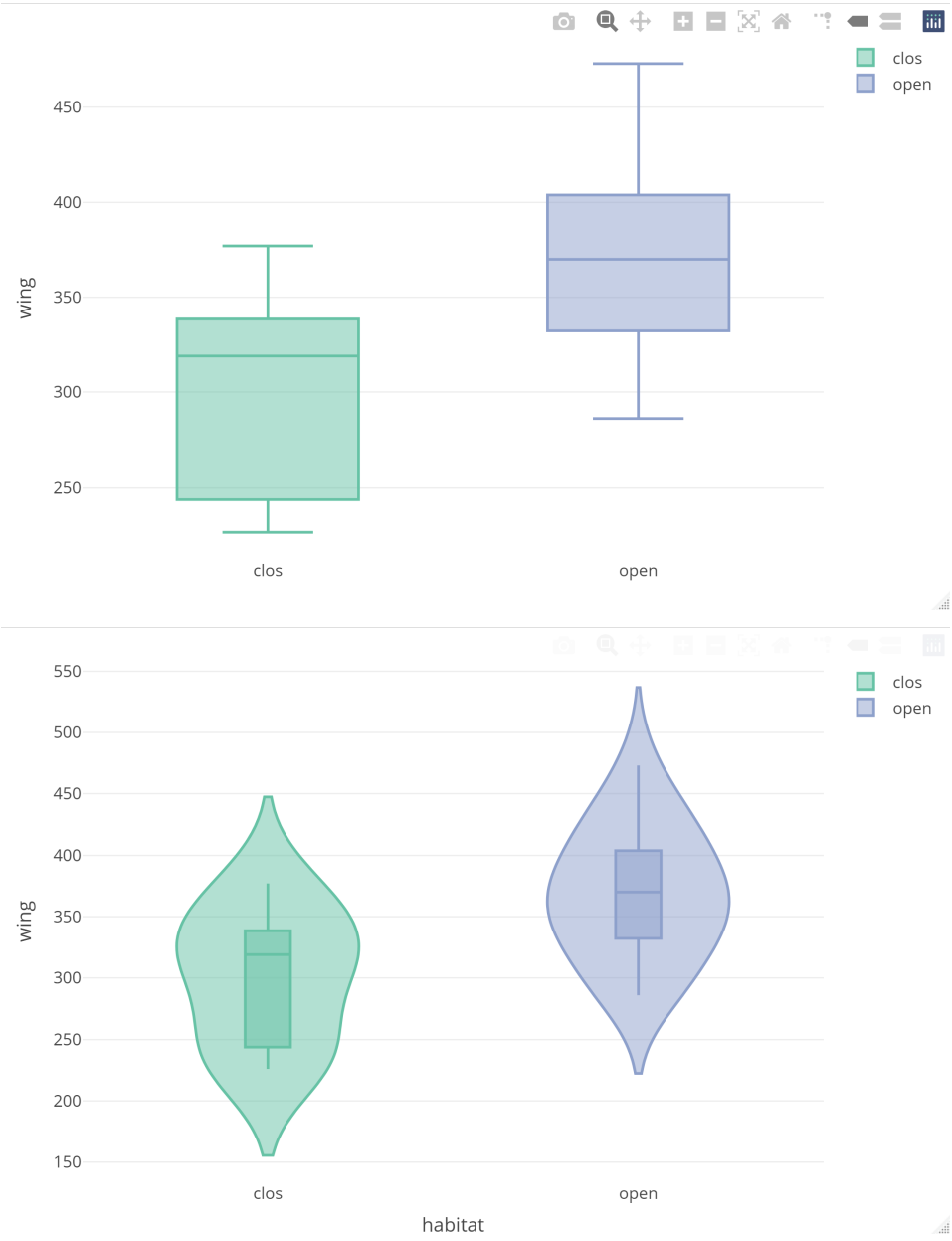
6.2 Diagramme en barre

```
bar_graph <- corvus %>%
  count(habitat, phylog) %>%
  plot_ly(y = ~habitat, x = ~n, color = ~phylog) %>%
  add_bars() %>% layout(barmode = "stack", xaxis=list(title="Effectif"),
                        yaxis=list(title="Habitat"))
```



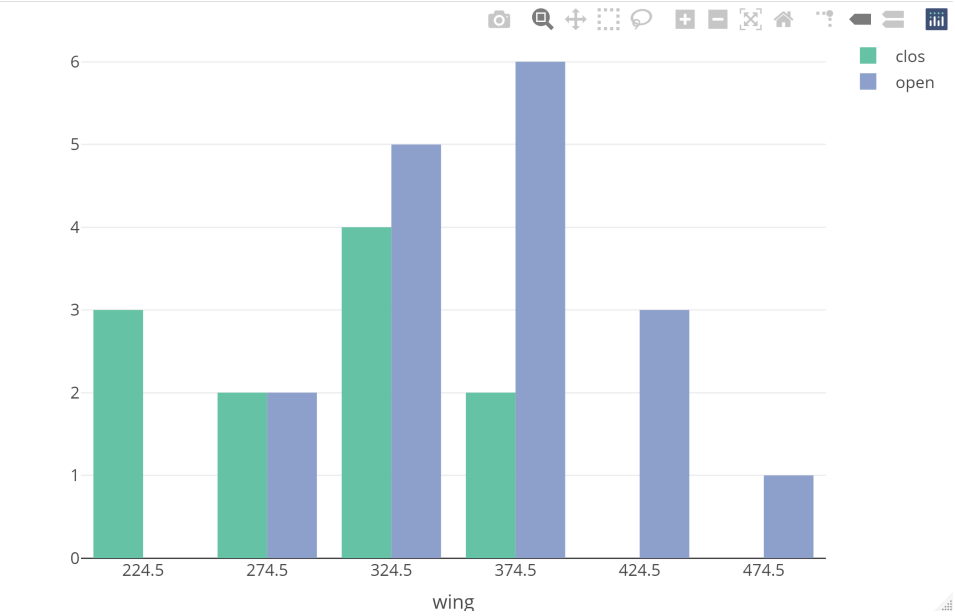
6.3 Boxplot et violin plot

```
# Boxplot
box_graph <- plot_ly(corvus, y = ~wing, color = ~habitat, type = "box")
# Violin plot
violin_graph <- plot_ly(corvus, x = ~habitat, y = ~wing, color= ~habitat,
                        type = 'violin', box = list(visible = T))
```



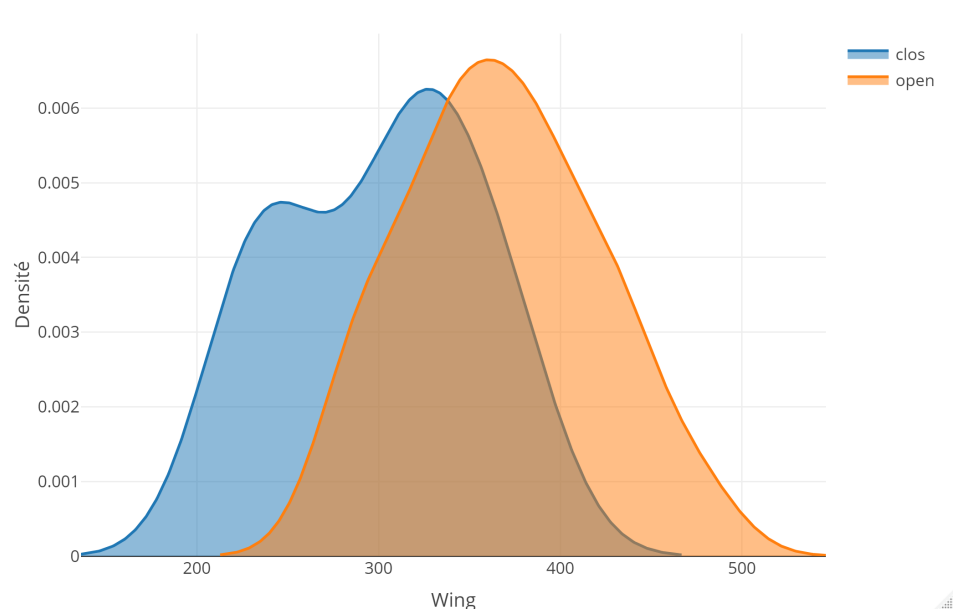
6.4 Histogramme

```
histo_graph <- plot_ly(corvus, x = ~wing, color = ~habitat) %>%
  add_histogram()
```



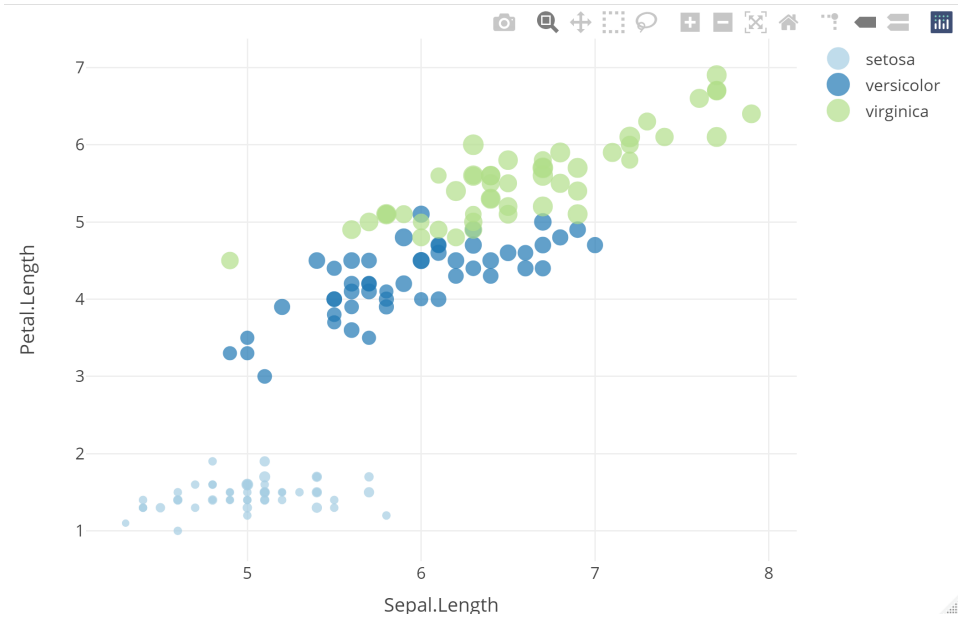
6.5 Densité

```
densite1 <- density(corvus$wing[corvus$habitat=="clos"])
densite2 <- density(corvus$wing[corvus$habitat=="open"])
densite_graph <- plot_ly(x = ~densite1$x, y = ~densite1$y, type = 'scatter',
                         mode = 'lines', fill = 'tozeroy', name="clos") %>%
  add_trace(x = ~densite2$x, y = ~densite2$y, fill = 'tozeroy', name="open") %>%
  layout(xaxis = list(title = 'Wing'), yaxis = list(title = 'Densité'))
```



6.6 Graphique à bulles

```
bubble <- plot_ly(iris, x = ~Sepal.Length, y = ~Petal.Length, type = 'scatter',
                  mode = 'markers', size = ~Petal.Width, color = ~Species,
                  colors = 'Paired', marker = list(opacity = 0.7))
```



Chapitre 7

Pour aller plus loin : l'analyse en composantes principales

Pour effectuer une analyse en composantes principales sous **R**, on peut utiliser les packages **ade4** (Dray and Dufour (2007), Bougeard and Dray (2018), Daniel Chessel, Dufour, and Thioulouse (2004), Dray, Dufour, and Chessel (2007)) et **FactoMineR** (Lê, Josse, and Husson (2008)). Pour effectuer des graphiques issus de l'ACP, on pourra utiliser le package **factoextra** (Kassambara and Mundt (2019)).

```
library(ade4)
library(FactoMineR)
library(factoextra)
```

7.1 Les jeux de données

Le jeu de données *doubs* du package ade4 (D. Chessel, Lebreton, and Yoccoz (1987))

Différentes variables ont été mesurées dans 30 sites situés le long du Doubs (variables environnementales, abondance des espèces de poissons et espèces de poissons).

```
data(doubs, package="ade4")
```

Ici, on va s'intéresser aux données physico-chimiques mesurées dans chaque site.

```
datadoubs <- doubs$env[,5:11]
```

Le tableau *datadoubs* a 30 lignes (sites) et 7 variables :

- La variable *pH* : pH
- La variable *har* : dureté de l'eau
- La variable *pho* : phosphates (mg/l * 100)
- La variable *nit* : nitrate (mg/l * 100)
- La variable *amm* : ammoniaque
- La variable *oxy* : oxygène (mg/l * 10)
- La variable *bdo* : demande biologique d'oxygène (mg/l * 10)

On effectuera l'ACP sur les variables (variables quantitatives).

Le jeu de données *tortues* du package ade4 (Jolicoeur and Mosimann (1960))

Il décrit la morphologie de 48 tortues. Le tableau de données a 48 lignes (tortues) et 4 variables :

- La variable *long* : longueur (mm)
- La variable *larg* : profondeur maximum (mm)
- La variable *haut* : hauteur (mm)
- La variable *sexe* : sexe

Chargeons le jeu de données *tortues* dans **R** :

```
data(tortues, package = "ade4")
```

7.2 Quelques notions

Les ressources

- Livres : François Husson and Pagès (2016), Ludovic Lebard and Morineau (2006) (plus technique), Saporta (2006) (plus technique)
- Internet : Vidéos You Tube de F. Husson, https://husson.github.io/MOOC.html, http://factominer.free.fr/factomethods/analyse-en-composantes-principales.html

But d'une ACP

L'ACP est une méthode d'analyse de données (statistique descriptive multidimensionnelle) utilisée notamment pour visualiser et réduire les données. Le jeu de données contient en lignes les individus et en colonnes les variables.

Principe d'une ACP

On dispose d'observations dans un espace de dimension égale au nombre de variables. L'ACP consiste à réduire le nombre de variables et donc d'arriver à un sous espace de dimension restreinte tel que la projection sur ce sous-espace contienne le plus d'information possible. Les variables constituent les composantes principales qui sont des combinaisons linéaires de variables du jeu de données initial. Les valeurs propres permettent de choisir la dimension de ce sous-ensemble. L'ACP va alors permettre également de décorréler des variables entre elles. D'un point de vue plus mathématique, la méthode consiste à trouver un axe issu de combinaisons linéaires des variables du jeu de données tel que la variance autour de cet axe soit maximum.

Centrage et réduction des données

Les variables quantitatives peuvent être dans des unités différentes. Il est donc important de :

- Centrer les données en soustrayant à chaque observation la moyenne afin de recentrer le nuage de points vers l'origine.
- Réduire les données en divisant les observations centrées par l'écart type.

Variables supplémentaires

Ces variables ne sont pas intégrées à l'ACP. Cela permet de voir comment elles sont liées aux axes de l'ACP. Typiquement, l'ACP n'utilisant que des variables quantitatives, on peut mettre en variables supplémentaires les variables qualitatives.

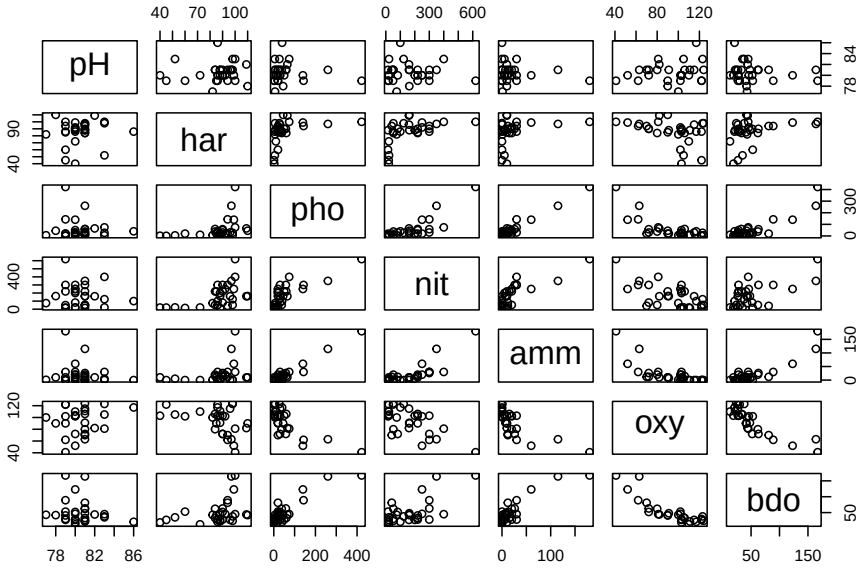
7.3 Première application : le jeu de données *datadoubs*

Le package ade4

Avant d'effectuer une ACP, il est important d'examiner les relations entre chaque variable en calculant la matrice de corrélation.

```
cor(datadoubs)
```

```
plot(datadoubs)
```



Dans le package **ade4**, on uilise la fonction *dudi.pca()* pour effectuer une ACP. L'option *center=TRUE* permet de centrer les données, l'option *scale=TRUE* permet de réduire les données et l'option *scannf=FALSE* permet de ne pas tracer le diagramme des valeurs propres, diagramme permettant de savoir combien d'axes on va garder. Pour ce faire, on fera un graphique à part. Si on dispose de valeurs manquantes, il faut soit les imputer préalablement, soit les supprimer si le pourcentage de données manquantes est négligeable. On peut utiliser la fonction *na.omit()* pour supprimer les lignes du tableau de données où existent des données manquantes.

Commençons par effectuer une ACP.

```
# Réalisation de l'ACP avec le package ade4
ACP <- dudi.pca(datadoubs, center=TRUE, scale=TRUE, scannf=FALSE)
```

Variance expliquée (inertie)

Choisissons la dimension du sous-espace. En effet, il s'agit de savoir combien d'axes on va retenir en traçant l'éboulis des valeurs propres (règle du coude par exemple). Les axes retenus représentent la quantité maximum de variation du nuage de points.

```
# Valeurs propres (variance expliquée ou inertie)
ACP$eig
```

```
## [1] 4.39589524 1.10002343 0.76061244 0.43077968 0.23247150 0.05448059 0.02573712
```

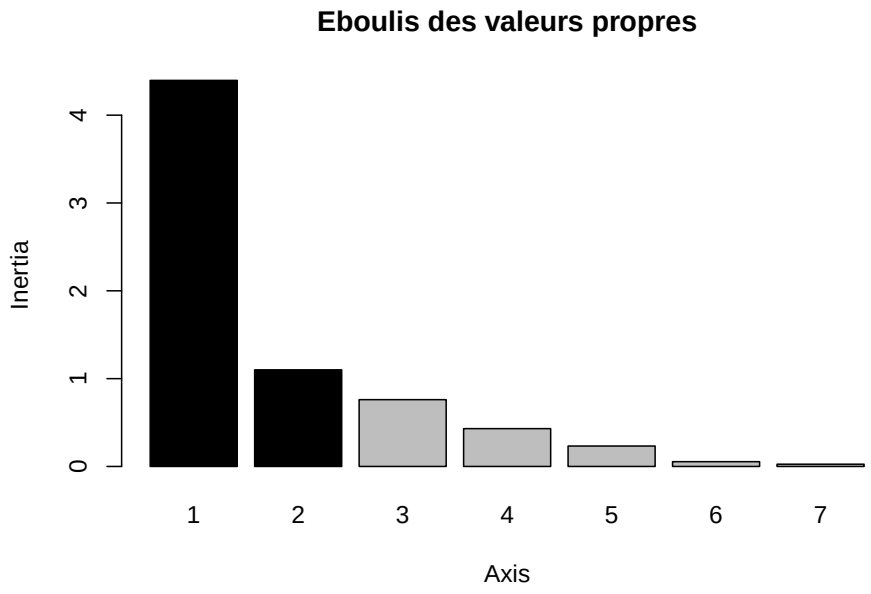
```
# Variance expliquée (en %)
(ACP$eig/sum(ACP$eig))*100
```

```
## [1] 62.7985034 15.7146204 10.8658920  6.1539954  3.3210214  0.7782941  0.3676732
```

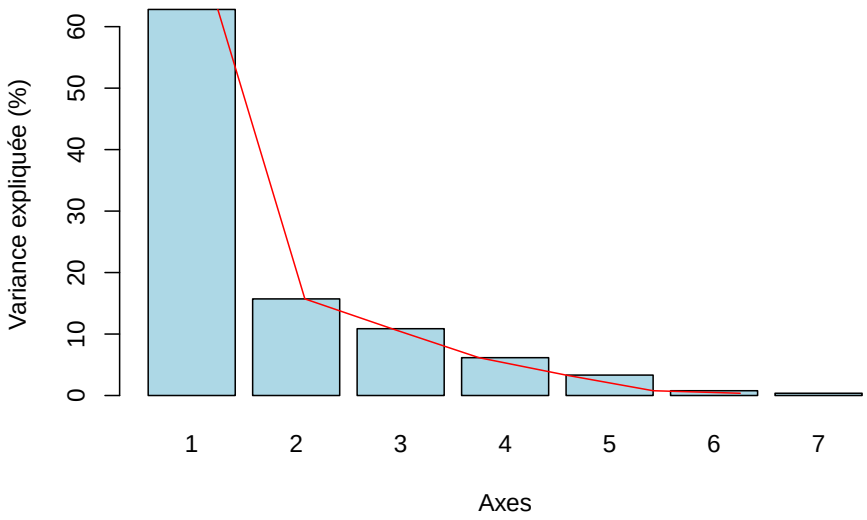
```
# Somme cumulée de la variance expliquée (en %)
cumsum((ACP$eig/sum(ACP$eig))*100)
```

```
## [1]  62.79850  78.51312  89.37902  95.53301  98.85403  99.63233 100.00000
```

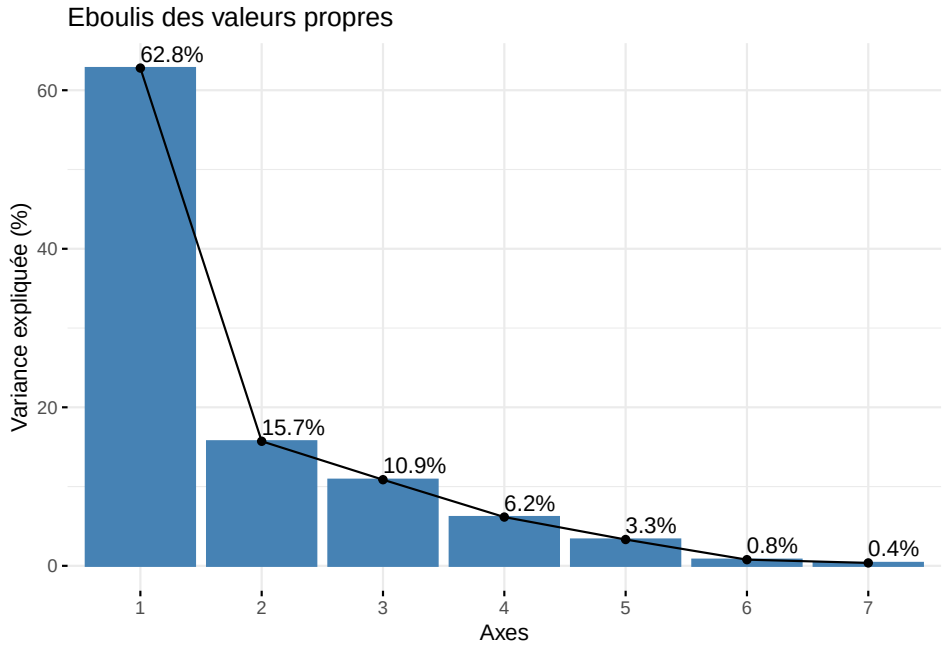
```
# Eboulis des valeurs propres
screeplot(ACP, main="Eboulis des valeurs propres")
```




```
# Autre solution
barplot((ACP$eig/sum(ACP$eig))*100,
        names.arg=as.character(c(1:7)), col="lightblue",
        xlab="Axes", ylab="Variance expliquée (%)")
lines(x=c(1:7), y=(ACP$eig/sum(ACP$eig))*100, col="red")
```




```
# Autre solution avec le package factoextra
fviz_eig(ACP, addlabels = TRUE,
         main="Eboulis des valeurs propres", xlab="Axes",
         ylab="Variance expliquée (%)")
```



Représentation graphique des variables

On va tout d'abord examiner la contribution des variables à la formation des axes. Sous **R**, on utilisera la fonction *inertia.dudi()*.

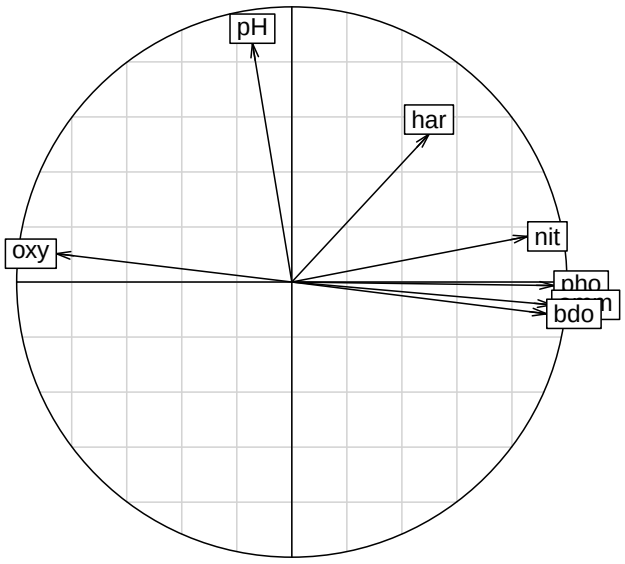
```
# Contribution des variables
inertia.dudi(ACP, col.inertia = T)$col.abs
# Contribution relative des variables (qualité de représentation)
inertia.dudi(ACP, col.inertia = T)$col.rel
```

Pour obtenir les coordonnées des variables, on utilisera la sortie de l'ACP. Dans le cas d'une ACP normée, les coordonnées des variables sont les corrélations entre les variables et les axes.

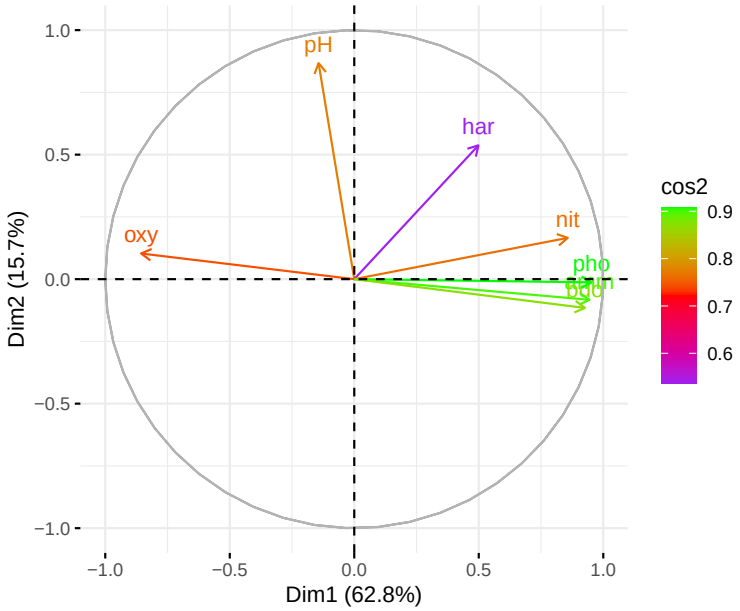
```
# Coordonnées des variables
ACP$co
```

Le cercle des corrélations nous permet de visualiser les relations entre les variables. Graphiquement, plus la variable est proche de la circonférence du cercle des corrélations, plus la variable est bien représentée (cos2 plus grand). A l'inverse, si la variable est proche du centre du cercle des corrélations, la variable est mal représentée (cos2 faible). Pour visualiser les corrélations, on examinera le sens et la direction des vecteurs. Les variables corrélées positivement vont dans la même direction.

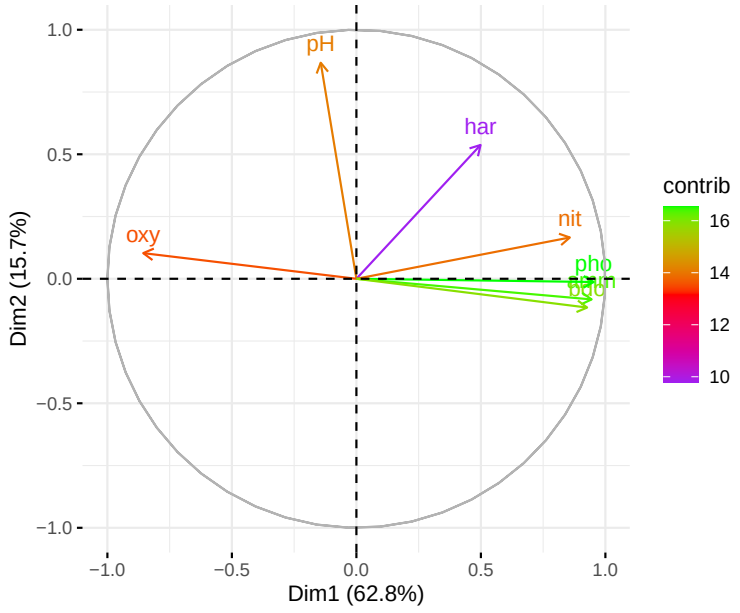
```
s.corcircle(ACP$co, xax=1, yax=2)
```




```
fviz_pca_var(ACP, col.var = "cos2",
             gradient.cols=c("purple","red","green")) +
  labs(title="")
```




```
fviz_pca_var(ACP, col.var = "contrib",
             gradient.cols=c("purple","red","green")) +
  labs(title="")
```



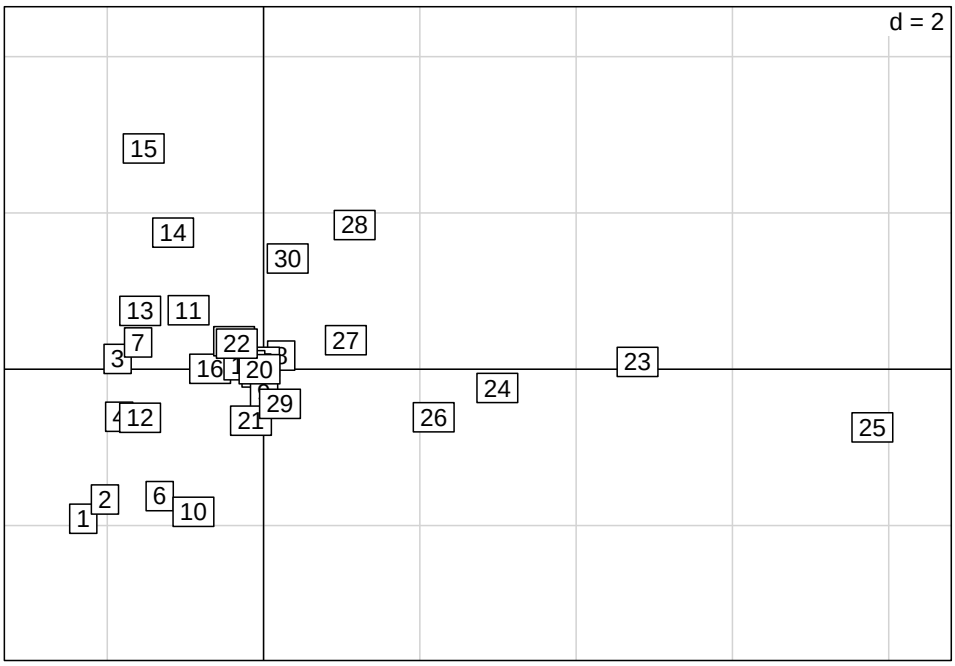
Représentation graphique des individus

De la même manière, on va examiner d'une part la contribution des individus à la formation des axes et d'autre part la qualité de représentation des individus. Sous **R**, on utilisera également la fonction *inertia.dudi()*.

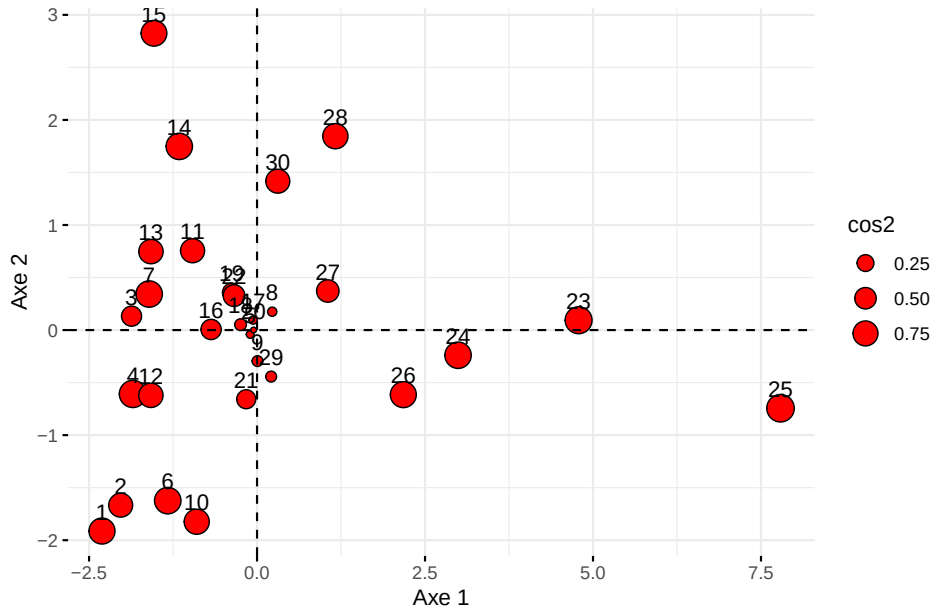
```
# Contribution des individus
inertia.dudi(ACP, row.inertia = T)$row.contrib
# Qualité de représentation des individus
inertia.dudi(ACP, row.inertia = T)$row.rel
# Coordonnées des individus
ACP$li
```

On peut également représenter graphiquement les individus. Les individus similaires sont regroupés.

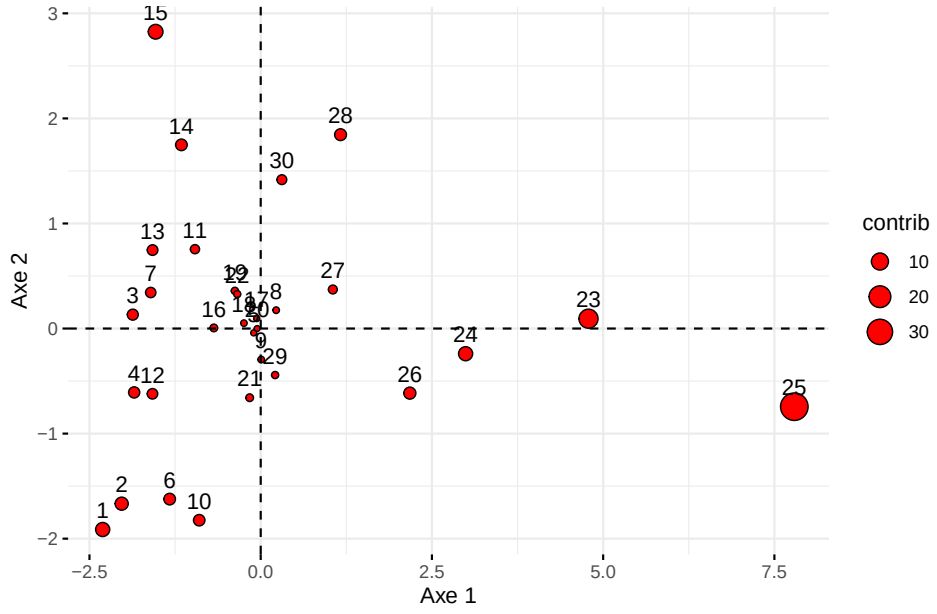
```
# Représentation des individus (composantes 1 et 2)
s.label(ACP$li, xax=1, yax=2)
```




```
fviz_pca_ind(ACP, pointsize="cos2", pointshape=21, fill="red") +
  labs(title="", x="Axe 1", y="Axe 2")
```



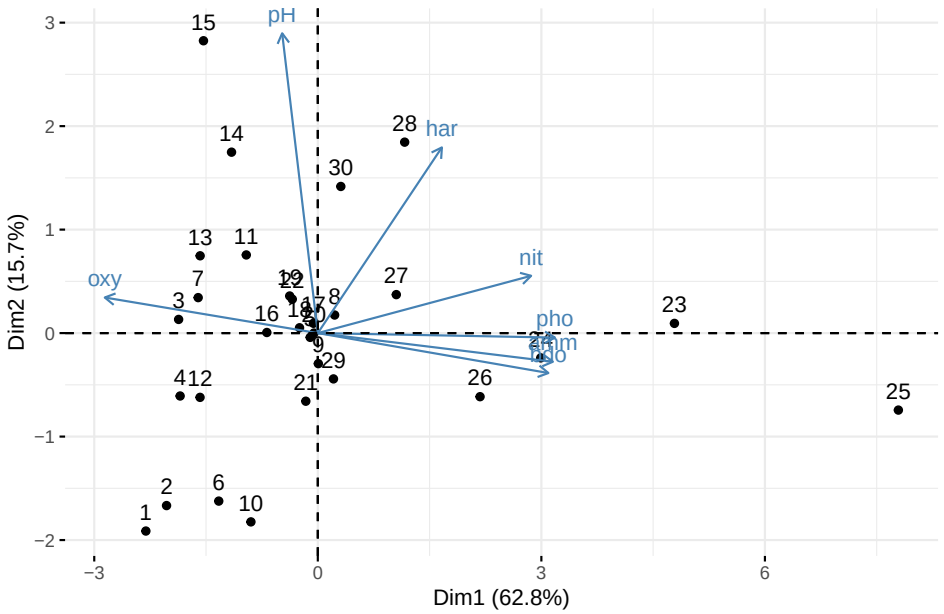

```
fviz_pca_ind(ACP, pointsize="contrib", pointshape=21, fill="red") +
  labs(title="", x="Axe 1", y="Axe 2")
```



Représentation simultanée des individus et des variables

On peut également représenter simultanément les individus et les variables.

```
# Représentation simultanée des individus et des
# variables (composantes 1 et 2)
fviz_pca_biplot(ACP, title="")
```

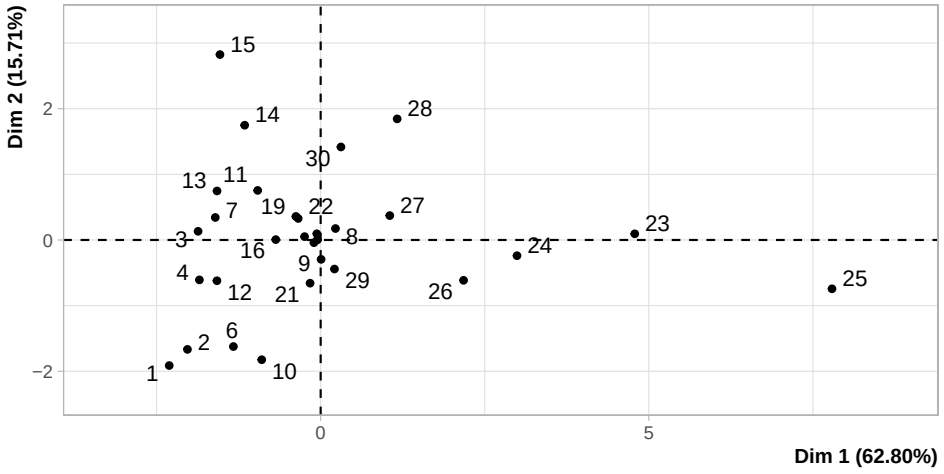


Le package FactoMineR

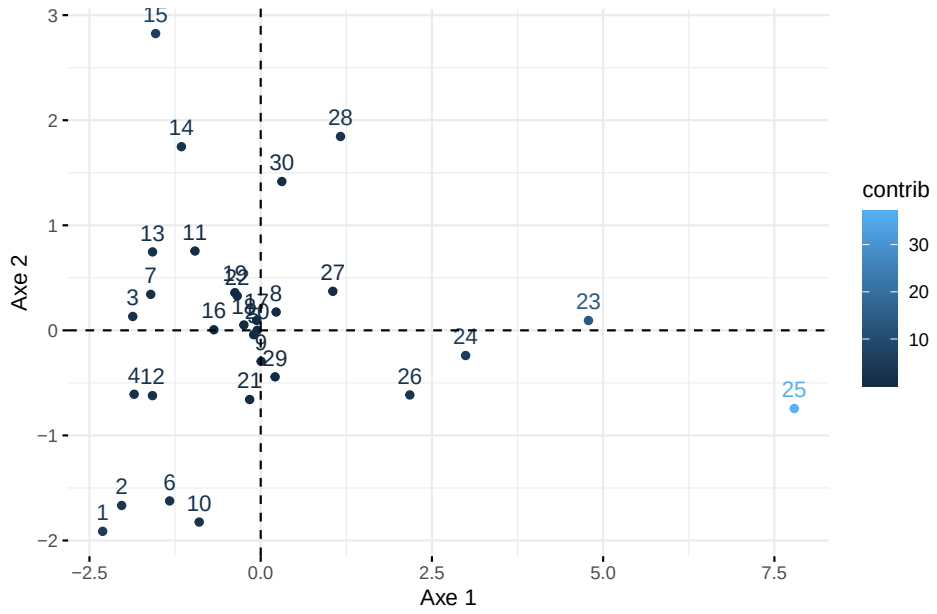
Pour efffectuer une ACP avec le package **FactoMineR**, on utilisera la fonction *PCA()*. L'option *scale.unit=TRUE* permet de centrer-réduire les données (par défaut).

```
ACP2 <- PCA(datadoubs , graph=F, scale.unit = TRUE)
summary(ACP2)
```

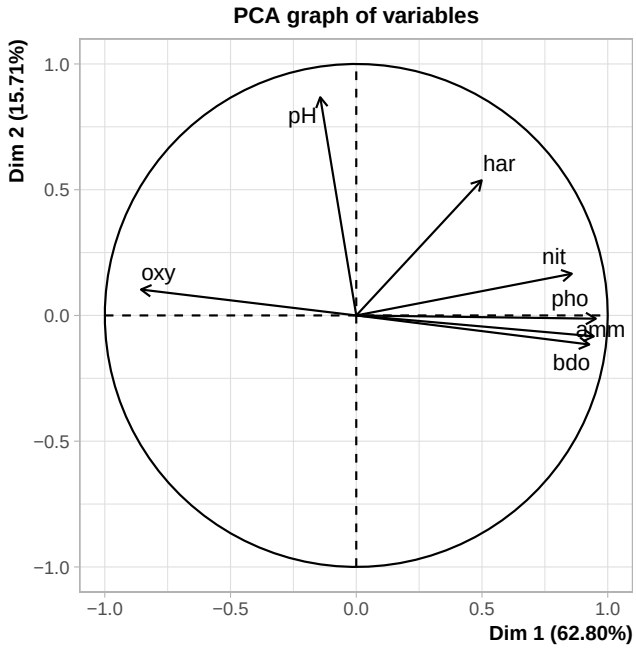
```
# Représentation des individus
plot.PCA(ACP2, axes=c(1,2),choix="ind", title="")
```



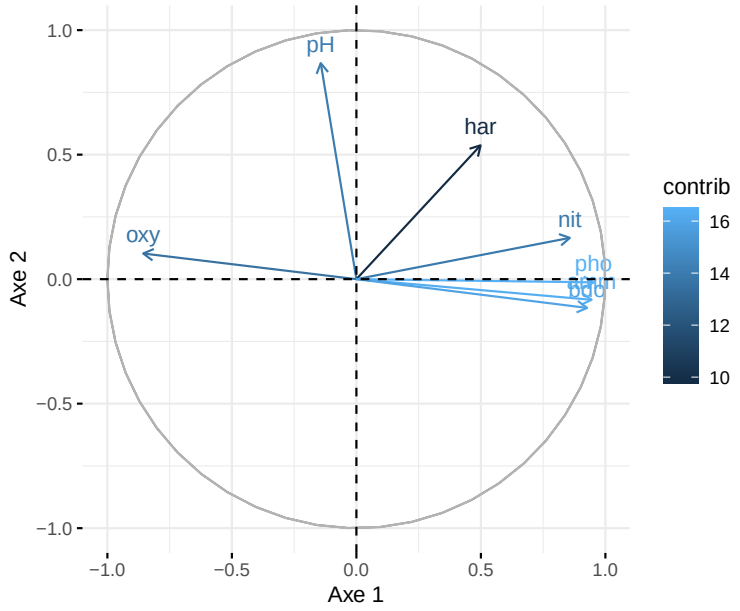
```
fviz_pca_ind(ACP2, col.ind="contrib") +
  labs(title="", x="Axe 1", y="Axe 2")
```



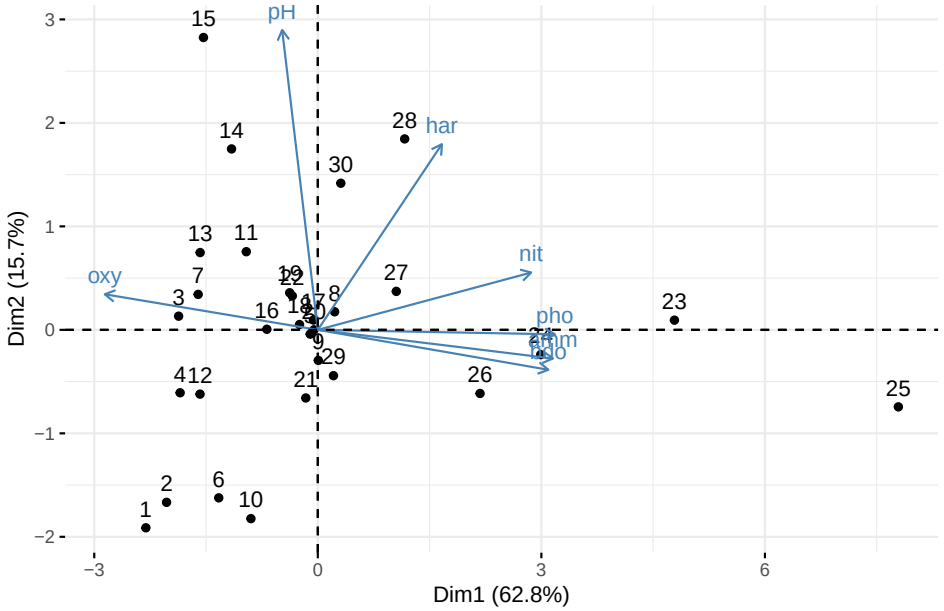

```
# Représentation des variables
plot.PCA(ACP2, axes=c(1,2),choix="var")
```




```
fviz_pca_var(ACP2, col.var = "contrib") +
  labs(title="", x="Axe 1", y="Axe 2")
```




```
# Représentation simultanée des individus et des
#variables (composantes 1 et 2)
fviz_pca_biplot(ACP2, title="")
```

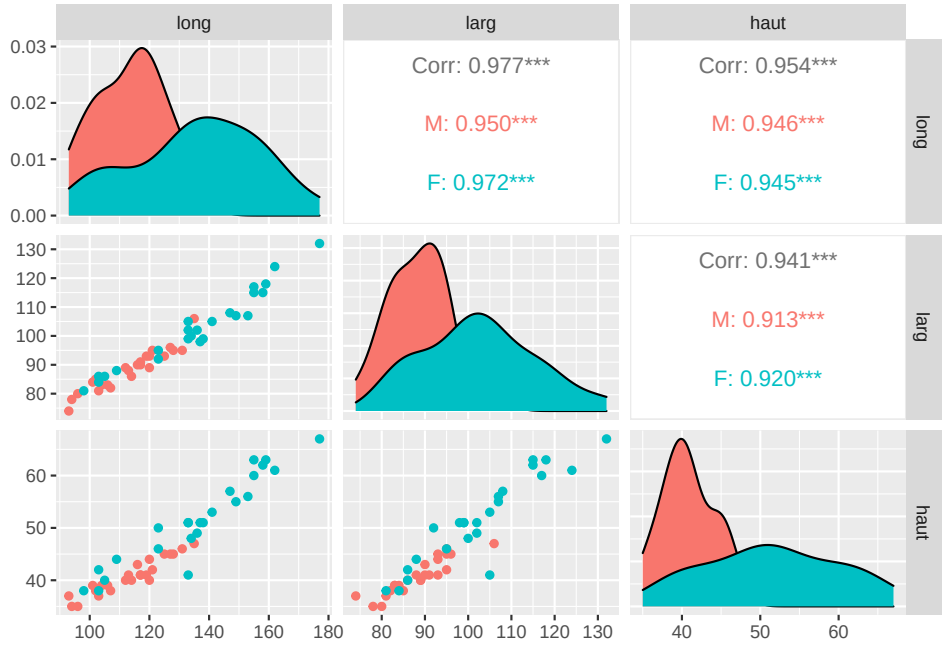


7.4 Deuxième application : le jeu de données *tortues*

Avant d'effectuer une ACP, il est important d'examiner les relations entre chaque variable en calculant la matrice de corrélation. On peut utiliser le package *GGally* pour effectuer les graphiques.

```
cor(tortues[,-4])
```

```
library(GGally)
ggpairs(tortues, columns = 1:3, mapping= ggplot2::aes(color=sexe))
```



Le jeu de données *tortues* a 3 variables quantitatives et une variable qualitative. L'ACP s'effectue sur des variables quantitatives et on ajoute une variable qualitative en variable supplémentaire.

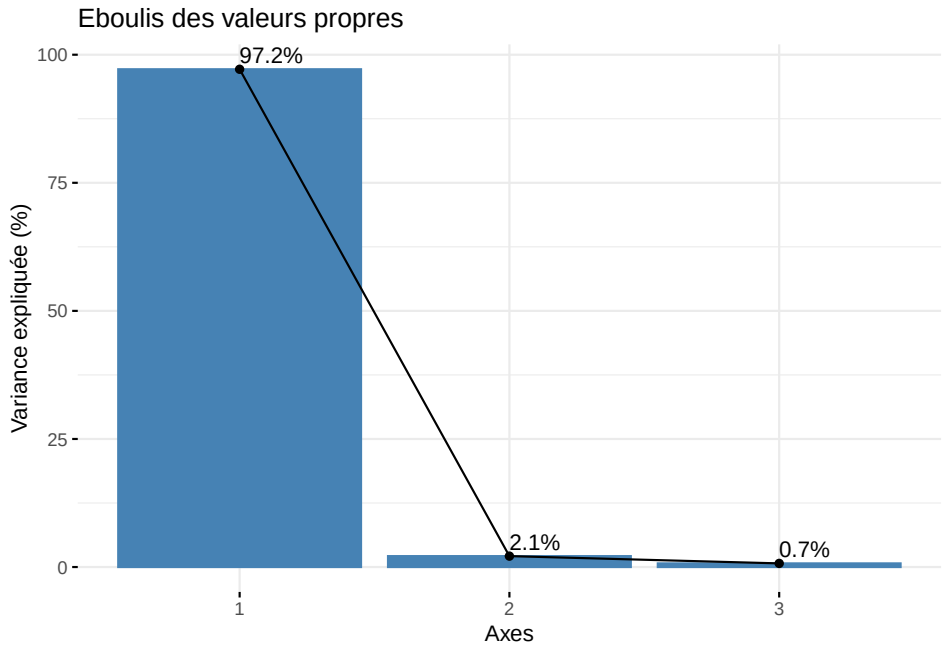
La package *ade4*

Utilisons la fonction *dudi.pca* pour effectuer l'ACP. Il faut exclure la quatrième colonne correspondant à la variable qualitative.

```
ACPtortues = dudi.pca(tortues[,-4],scannf=F)
```

Effectuons l'éboulis des valeurs propres.

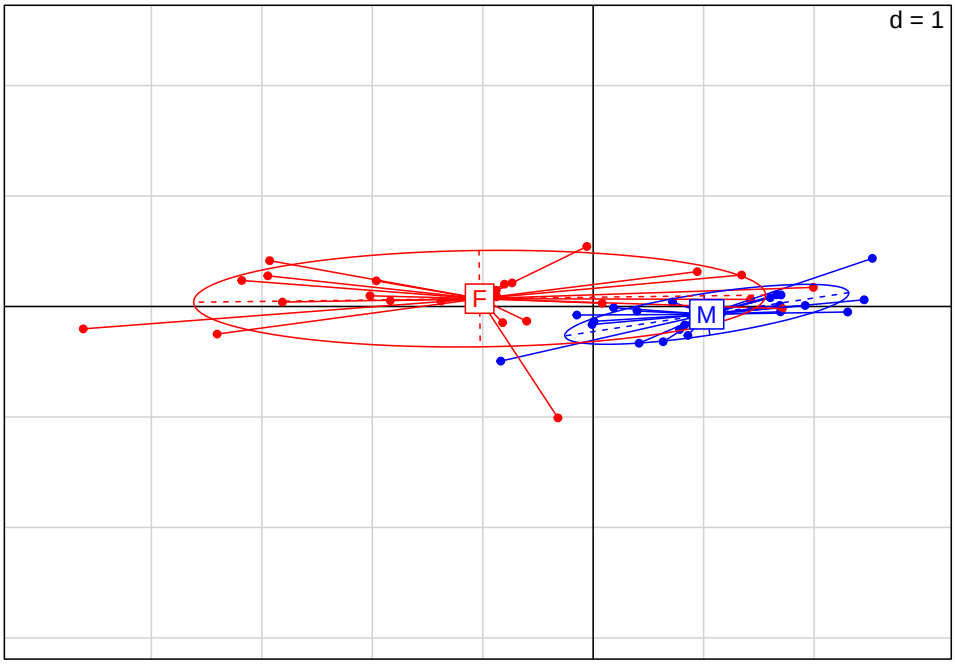
```
fviz_eig(ACPtortues, addlabels = TRUE,
         main="Eboulis des valeurs propres", xlab="Axes",
         ylab="Variance expliquée (%)")
```



On conserve les deux premiers axes.

Représentons les individus en ajoutant la variable sexe en tant que variable supplémentaire. On utilise la fonction *s.class()*.

```
s.class(ACPtortues$li, tortues$sexe, col=c("blue","red"))
```

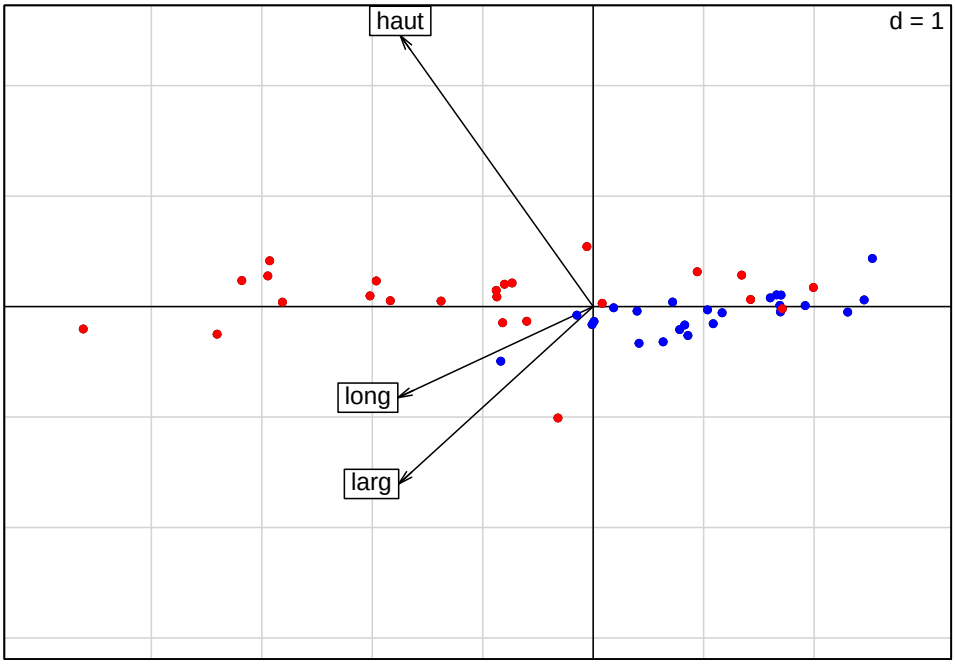


Représentons simultanément les variables et les individus en ajoutant la variable sexe en tant que variable supplémentaire.

```
scatter(ACPtortues, posieig = "none", clab.row = 0)
```

```
## NULL
```

```
s.class(ACPtortues$li, tortues$sexe, col = c("blue","red"),
        add.plot = TRUE, cstar = 0, cellipse = 0, clabel = 0)
```



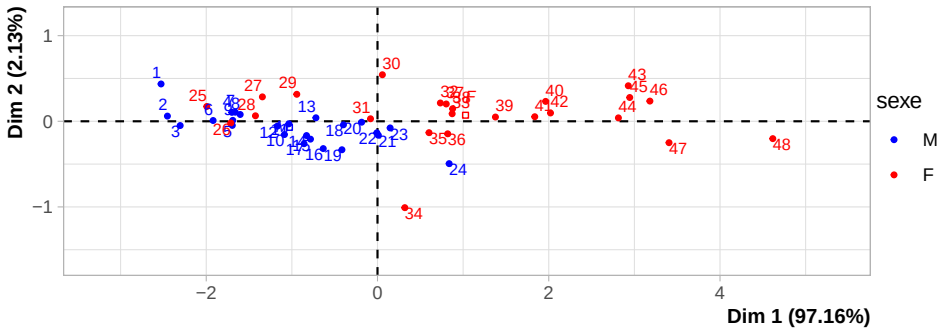
La package *factomineR*

Dans le package *factomineR*, on utilise l'option *quali.sup* de la fonction *PCA()* pour ajouter une variable qualitative supplémentaire.

```
ACPtortues2 <- PCA(tortues, quali.sup=4, graph=F, scale.unit = TRUE)
summary(ACPtortues2)
```

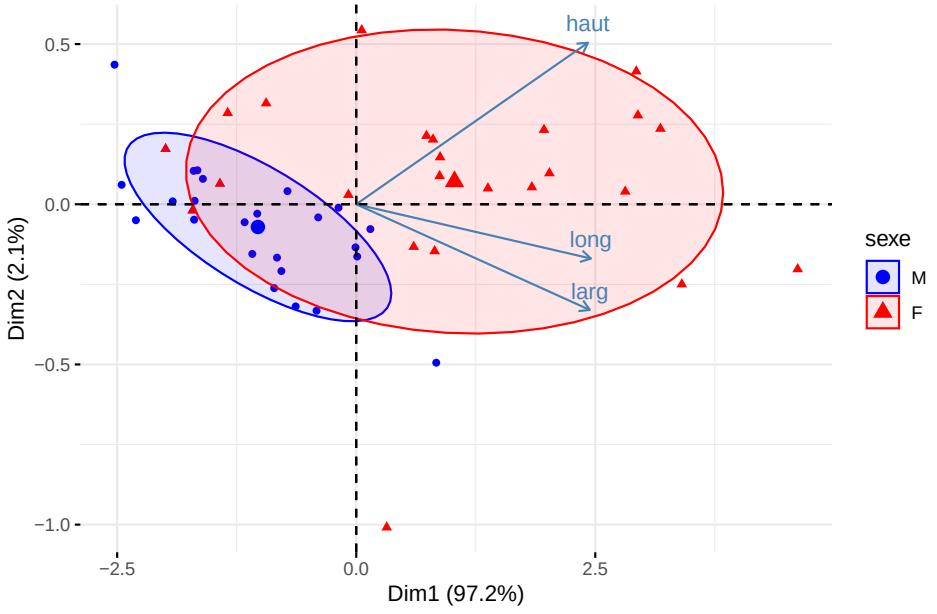
Représentons les individus.

```
plot.PCA(ACPtortues2, axes=c(1,2),choix="ind",
         habillage=4, cex=0.7, title="", palette = c("blue","red"))
```



Représentons simultanément les individus et les variables.

```
biplot <- fviz_pca_biplot(ACPtortues2, title="",
                          label="var", habillage=4,
                          addEllipses=TRUE,
                          ellipse.level=c(0.7))
biplot +
  scale_color_manual(values=c("blue", "red"))+
  scale_fill_manual(values=c("blue", "red"))
```



7.5 Les données manquantes

Le jeu de données peut comporter des données manquantes. Pour effectuer l'ACP, la fonction *PCA()* du package **FactoMineR** remplace les NA par la moyenne de la variable (imputation simple). Pour imputer des données, il existe le package **missMDA** et la fonction *imputePCA()* (Josse and Husson (2016)).

7.6 Quelques compléments

Des packages complémentaires pour une prise en main interactive et automatique existent. Il s'agit :

- du package *FactoInvestigate* : effectuer des rapports automatisés (Thuleau and Husson (2020))
- du package *Factoshiny* : ACP par interface graphique interactive Shiny (Vaissie, Monge, and Husson (2020))

```
library(FactoInvestigate)
# rapport <- Investigate(ACP2, file = "rapportDoubs.pdf",
# document = "pdf_document")
```

```
library(Factoshiny)
# sortieshiny=PCAshiny(datadoubs)
```

7.7 A VOUS DE JOUER !

- Charger le jeu de données decathlon du package FactoMineR. Effectuer une ACP. On exclura les variables Rank et Points. Effectuer l'interprétation.
- Effectuer une ACP sur votre jeu de données.

Références

al., Andri Signorell et mult. 2020. *DescTools : Tools for Descriptive Statistics*. https://cran.r-project.org/package=DescTools.

Anderson, Edgar. 1935. “The Irises of the Gaspe Peninsula.” *Bulletin of the American Iris Society* 59 : 2–5.

Bougeard, Stéphanie, and Stéphane Dray. 2018. “Supervised Multiblock Analysis in R with the ade4 Package.” *Journal of Statistical Software* 86 (1) : 1–17. https://doi.org/10.18637/jss.v086.i01.

Chessel, Daniel, Anne-Béatrice Dufour, and Jean Thioulouse. 2004. “The ade4 Package – I : One-Table Methods.” *R News* 4 (1) : 5–10. https://cran.r-project.org/doc/Rnews/.

Chessel, D., J. D. Lebreton, and N. Yoccoz. 1987. “Propriétés de l’analyse Canonique Des Correspondances ; Une Illustration En Hydrobiologie.” *Revue de Statistique Appliquée* 35 (4) : 55–71.

Comtois, Dominic. 2020. *Summarytools : Tools to Quickly and Neatly Summarize Data*. https://CRAN.R-project.org/package=summarytools.

Dray, Stéphane, and Anne–Béatrice Dufour. 2007. “The ade4 Package : Implementing the Duality Diagram for Ecologists.” *Journal of Statistical Software* 22 (4) : 1–20. https://doi.org/10.18637/jss.v022.i04.

Dray, Stéphane, Anne-Béatrice Dufour, and Daniel Chessel. 2007. “The ade4 Package – II : Two-Table and K-Table Methods.” *R News* 7 (2) : 47–52. https://cran.r-project.org/doc/Rnews/.

François Husson, Sébastien Lê, and Jérôme Pagès. 2016. *Analyse de Données Avec r, 2e Édition Revue Et Augmentée*. Pratique de La Statistique. Presses universitaires de Rennes.

Frédéric Bertrand, Myriam Maumy-Bertrand. 2010. *Initiation à La Statistique Avec R : Cours, Exemples, Exercices Et Problèmes Corrigés*. Dunod.

Jolicoeur, P., and J. Mosimann. 1960. “Size and Shape Variation in the Painted Turtle. A Principal Component Analysis.” *Growth* 24 : 339–54.

Josse, Julie, and François Husson. 2016. “missMDA : A Package for Handling Missing Values in Multivariate Data Analysis.” *Journal of Statistical Software* 70 (1) : 1–31. https://doi.org/10.18637/jss.v070.i01.

Kassambara, Alboukadel, and Fabian Mundt. 2019. *Factoextra : Extract and Visualize the Results of Multivariate Data Analyses*. https://CRAN.R-project.org/package=factoextra.

Laiolo, Paola, and Antonio Rolando. 2003. “The Evolution of Vocalisations in the Genus Corvus : Effects of

Phylogeny, Morphology and Habitat." *Evolutionary Ecology* 17 (March) : 111–23. https://doi.org/10.1023/A:1023003110969.

Lê, Sébastien, Julie Josse, and François Husson. 2008. "FactoMineR : A Package for Multivariate Analysis." *Journal of Statistical Software* 25 (1) : 1–18. https://doi.org/10.18637/jss.v025.i01.

Ludovic Lebard, Marie Piron, and Alain Morineau. 2006. *Statistique Exploratoire Multidimensionnelle - 4e Édition : Visualisation Et Inférence En Fouille de Données*. Dunod.

Meyer, David, Evgenia Dimitriadou, Kurt Hornik, Andreas Weingessel, and Friedrich Leisch. 2019. *E1071 : Misc Functions of the Department of Statistics, Probability Theory Group (Formerly : E1071), TU Wien*. https://CRAN.R-project.org/package=e1071.

Saporta, Gilbert. 2006. *Probabilités, Analyse Des Données Et Statistique*. Editions Technip.

Sievert, Carson. 2020. *Interactive Web-Based Data Visualization with r, Plotly, and Shiny*. Chapman ; Hall/CRC. https://plotly-r.com.

Thuleau, Simon, and Francois Husson. 2020. *FactoInvestigate : Automatic Description of Factorial Analysis*. https://CRAN.R-project.org/package=FactoInvestigate.

Vaissie, Pauline, Astrid Monge, and Francois Husson. 2020. *Factoshiny : Perform Factorial Analysis from 'FactoMineR' with a Shiny Application*. https://CRAN.R-project.org/package=Factoshiny.

Verzani, John. 2005. *Using R for Introductory Statistics*. Chapman & Hall/CRC.

Wilke, C. O. 2019. *Fundamentals of Data Visualization : A Primer on Making Informative and Compelling Figures*. O'Reilly Media.

Zar, Jerrold H. 1984. *Biostatistical Analysis*. Prentice Hall ; 2nd edition.