# TP OUMOBIO 2: Fonctions statistiques de base et visualisation des données sous R

Mathieu Brevet

2024-09-11

Bienvenue dans ce second volet de TP sur R ! Nous allons maintenant étudier comment décrire des variables qualitatives et quantitatives sur les plans statistique et graphique.

## Décrire une variable qualitative

Dans un premier temps nous allons essayer de comprendre comment **décrire une variable qualitative** sur les plans statistique et graphique.

Reprenons notre jeu de données d'études là où nous l'avions laissé:

```
setwd("~/ATER PAU 2024/Cours modifiés/OUMOBIOS")


data_lezard = read.csv(file = "Suivi_lezard_vivipare.csv")
```

### Descriptions statistiques

Nous allons dans un premier temps décrire la variable "Population d'origine" (colonne POP) de notre jeu de données. Nous allons tout d'abord en faire une **description statistique** en essayant de comprendre la répartition des individus dans les différentes populations:

```
unique(data_lezard$POP)
```

```
## [1] "MON" "JOC" "JON" "COP" "VIA" "PIM" "BOU"
```

```
# classes (=valeurs) de cette variable catégorielle

table(data_lezard$POP)
```

```
##
## BOU COP JOC JON MON PIM VIA
##  12  15  33  12  51  30  15
```

```
# effectifs de chaque classe de la variable, aussi appelé table de contingence

table(data_lezard$POP)/length(data_lezard$POP)
```

```
##
##         BOU         COP         JOC         JON         MON         PIM         VIA
## 0.07142857 0.08928571 0.19642857 0.07142857 0.30357143 0.17857143 0.08928571
```

```r
# fréquence de chaque classe de la variable


# essayons également de trouver le mode de la variable (valeur pour laquelle on
# a l'effectif maximal):

max(table(data_lezard$POP))
```

```
## [1] 51
```

```r
# effectif maximal

names(which.max(table(data_lezard$POP)))
```

```
## [1] "MON"
```

```r
# la fonction which.max() permet de déterminer la position de la valeur
# maximale, puis names() permet de récupérer le nom associé à cette position
```

Prenons un autre exemple avec les dates de naissances des individus, nous voulons comprendre comment le nombre de naissance évolue dans le temps, nous allons donc utilisé des fréquences cumulées:

```r
cumsum(table(data_lezard$BIRTH_DATE))/length(data_lezard$BIRTH_DATE)
```

```
##  02/07/19  04/07/19  05/07/19  10/07/19  11/07/19  12/07/19  13/07/19  16/07/19
## 0.1607143 0.1845238 0.2678571 0.2916667 0.3511905 0.3928571 0.5714286 0.6607143
##  17/07/19  18/07/19  20/07/19  21/07/19  22/07/19  23/07/19  24/07/19
## 0.7500000 0.8035714 0.8750000 0.8869048 0.9107143 0.9345238 1.0000000
```

```r
# cumsum() permet de réaliser la somme cumulée d'un vecteur
```

Lorsque nous avons cherché à décrire les fréquences de classes et le mode de la variable nous avons utilisé une suite de commande. Au lieu d'utiliser cette suite de commande de manière répétée à chaque fois que nous en aurons besoin, nous pouvons construire **une fonction** nous permettant de formaliser l'opération et de pouvoir l'appeler dès que nécessaire (et ainsi gagner du temps dans l'écriture des scripts).
Une fonction s'écrit sur R sous la forme: **Nom <- function(arg)** {...} ("Nom": nom de la fonction; "arg": argument de la fonction, il peut y en avoir plusieurs séparés par des virgules; "...": commandes réalisées par la fonction). Voici deux exemples de construction de fonction:

```r
freq <- function (vect) { # argument de la fonction: vecteur de chaînes de caractères
  table(vect) /       # table des effectifs du vecteur
    length(vect)      # taille du vecteur
}
# fonction calculant la fréquence de toutes les classes d'une variable catégorielle
```

```r
mode <- function (vect) { # argument de la fonction: vecteur
  names(                          # récupère le nom sur une variable avec identifiant
    which.max(                    # récupère la (ou les) positions de la valeur maximale
      table(                      # table des effectifs du vecteur
        vect)
      )
    )
}
# fonction calculant le mode d'une variable
```

**BONNES PRATIQUES**

Il est utile de regrouper les fonctions au même endroit dans votre script dans le cas de fonctions qui sont utilisées de manière récurrentes au cours de votre script, pour qu'un lecteur (ou vous-mêmes) puisse rapidement y accéder pour les consulter ou les modifier. Une bonne pratique consiste à réaliser un **préambule** au tout début de votre script pour regrouper vos fonctions (et décrire à l'aide de commentaires leur fonctionnement et utilité).

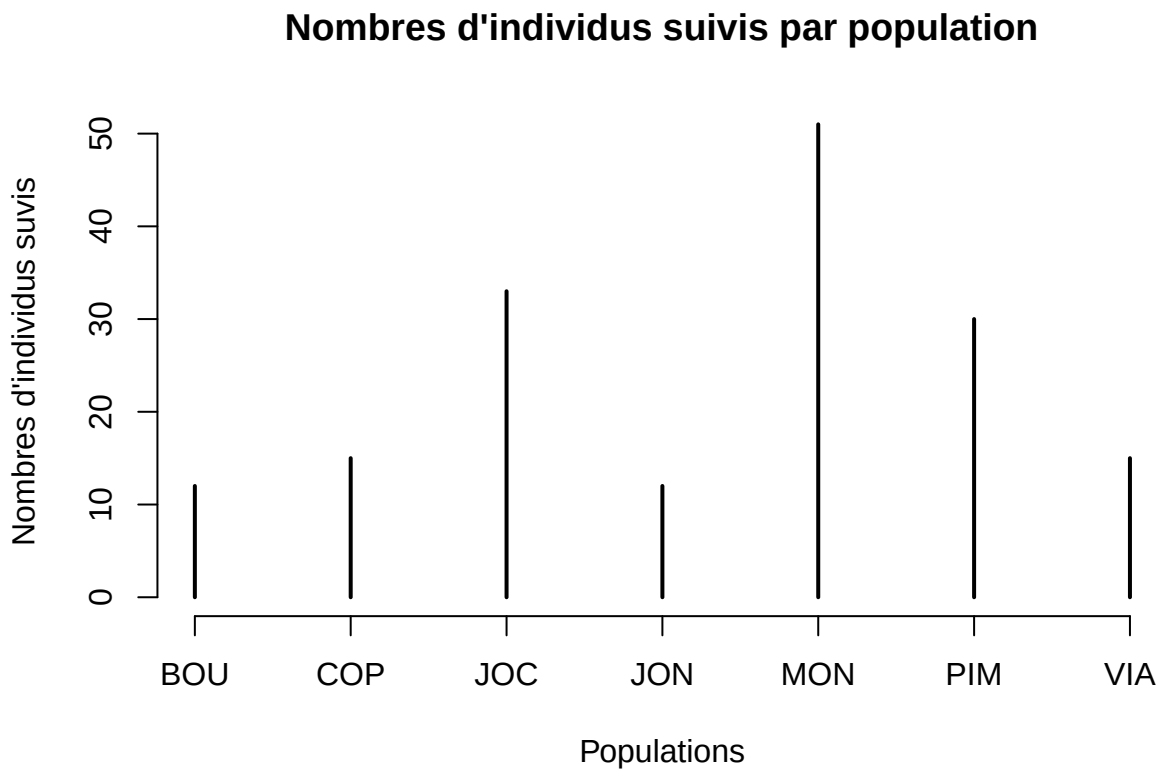
**POUR ALLER PLUS LOIN**

Comme vous avez déjà pu le constater il existe souvent plusieurs manières de réaliser une opération complexe sur R et donc d'écrire une fonction. Il peut parfois être intéressant d'identifier la solution la plus efficiente parmi les différentes possibilités. Pour cela il existe des outils dans R qui permettent de calculer le temps écoulé au cours d'une opération (system.time() ou microbenchmark() du paquet "microbenchmark" qui permet de comparer le temps de plusieurs opérations en les répétant de nombreuses fois), plus ce temps est court plus votre opération/fonction est performante. Cela peut être très utile si vous répétez de très nombreuses fois une opération/fonction ou si vous l'utilisez sur des volumes de données extrêmement important. Cela peut également vous permettre d'identifier quelles parties de votre fonction/opération ralentissent votre processus.

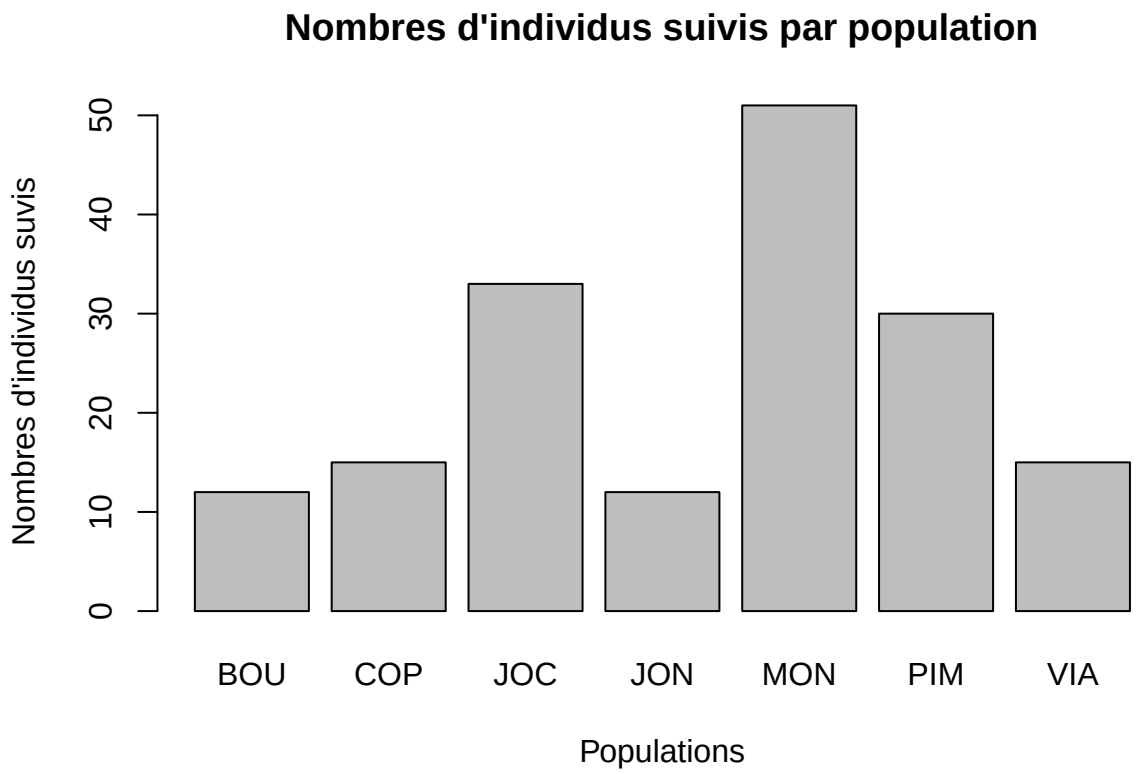
## Descriptions graphiques

Après avoir décrit la variable sur le plan statistique nous allons maintenant illustrer nos résultats graphiquement. N'oubliez pas certaines bonnes pratiques lorsque vous créez un graphique (que vous utiliserez pour un rendu): le graphique doit toujours avoir un **titre**, des **légendes** décrivant les axes et une **échelle** facilement lisible. Voici les principaux **outils graphiques** que vous pouvez utiliser pour une variable qualitative:

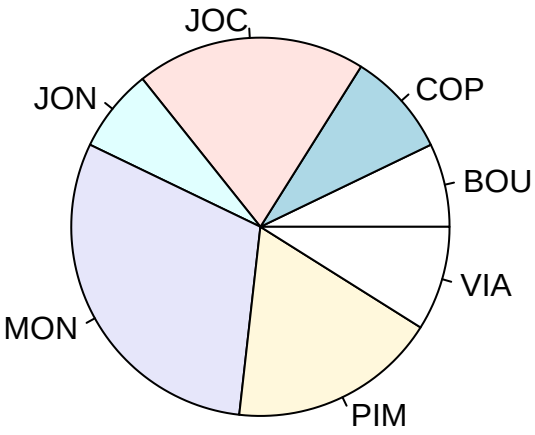
```r
plot(
  table(data_lezard$POP),
  main = "Nombres d'individus suivis par population",
  xlab = "Populations",
  ylab = "Nombres d'individus suvis"
  )
```



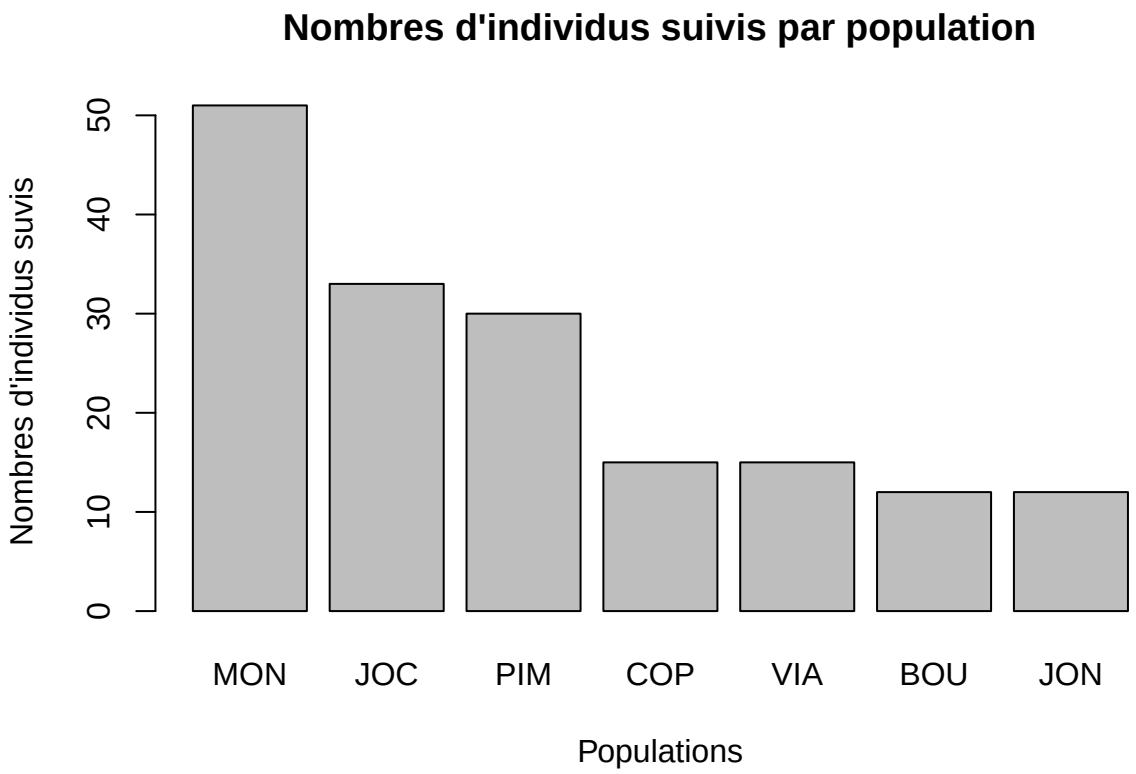

```r
# autre visualisation, la plus employée et commune pour une variable qualitative (conseillée):
barplot(
  table(data_lezard$POP),
  main = "Nombres d'individus suivis par population",
  xlab = "Populations",
  ylab = "Nombres d'individus suvis"
  )
```




```
pie(
  table(data_lezard$POP),
  main = "Proportion d'individus suivis par population"
  )
```




```r
# diagramme en "camembert" (peu utilisé)



# visualiser après un tri des valeurs:

table = table(data_lezard$POP) [order(table(data_lezard$POP), decreasing = T)]

barplot(
  table,
  main = "Nombres d'individus suivis par population",
  xlab = "Populations",
  ylab = "Nombres d'individus suvis"
  )
```




```
 # enregistrement d'un graphique dans votre dossier de travail:

png("Distribution_population.png", res=100)
# ouvre une session graphique, enregistrement en png (aussi possible en jpeg, pdf), avec
# la résolution en ppi (pixels per inch, pour les formats hors pdf), et les dimensions
# gérés par les paramètres (height, width)
barplot(
  table,
  main = "Nombres d'individus suivis par population",
  xlab = "Populations",
  ylab = "Nombres d'individus suvis"
  )
# création figure
dev.off()
```

```
## pdf
##   2
```

```
graphics.off()
# fermeture de la session graphique


rm(table)
```

**NOTE IMPORTANTE**

Les figures que vous créez sous R peuvent être enregistrées sur votre dossier de travail, soit directement en utilisant R (voir lignes de code ci-dessus), soit en utilisant l'interface Rstudio, dans le panel en bas à droite, sur l'onglet "Plots" (voir image ci-dessous). L'avantage de la première méthode est qu'elle permet de gérer la résolution de l'image, chose très utile dans le cas où le rendu doit être propre. L'export en pdf assure une qualité optimale dans tous les cas mais peut être plus dur à insérer dans certains documents.

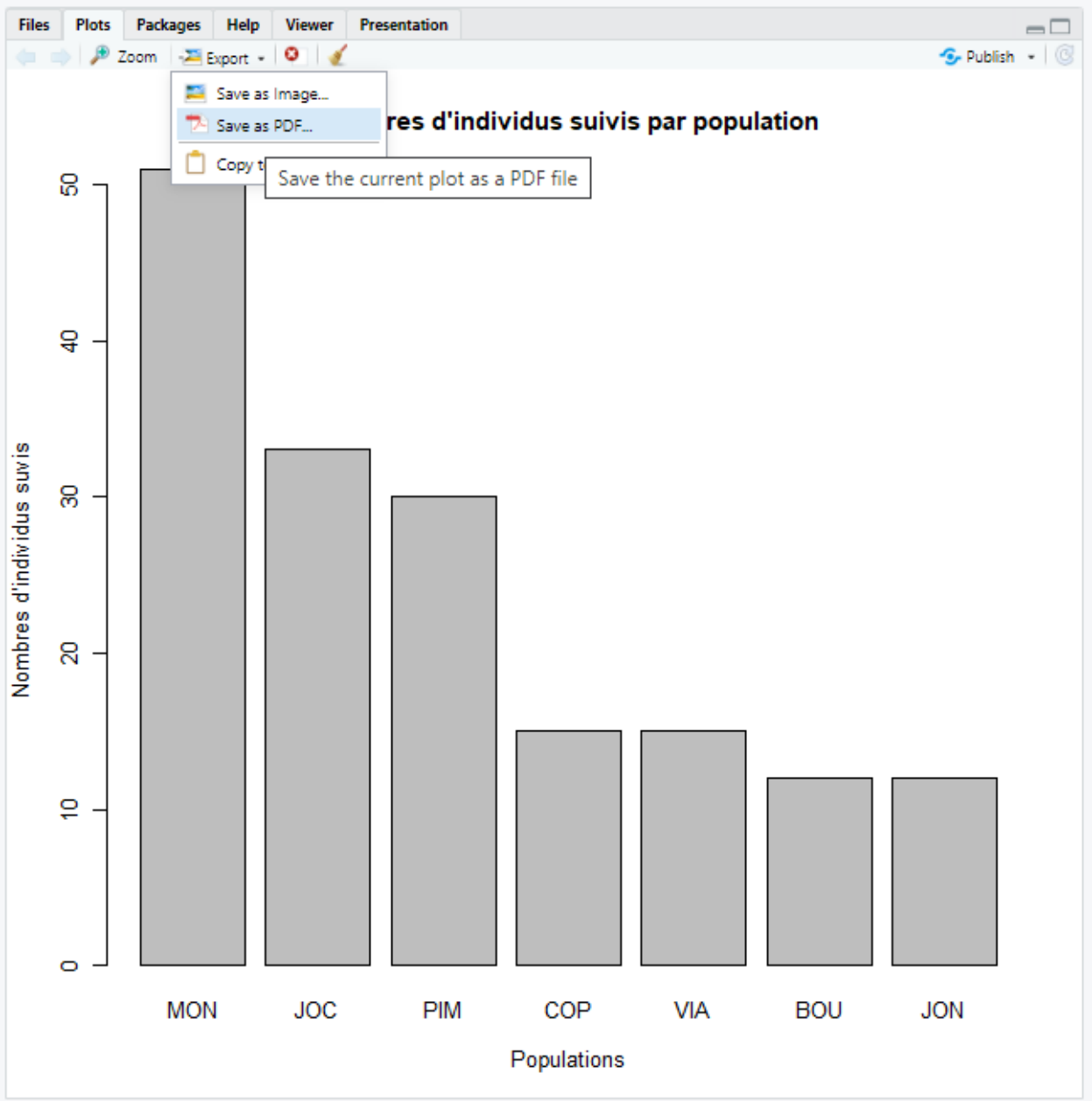



Figure 1: Enregistrement d'une image

**EXERCICE**

- Créez une fonction permettant d'obtenir des fréquences cumulées
- Convertissez la taille des individus en variable catégorielle et décrivez-la statistiquement et graphiquement

```
cumfreq <- function (vect) { # argument de la fonction: vecteur de chaînes de caractères
  cumsum(                # somme cumulée des éléments d'un vecteur
    table(vect) /        # table des effectifs du vecteur
      length(vect)       # taille du vecteur
    )
}
# fonction calculant la fréquence cumulée de toutes les classes d'une variable catégorielle




cat_taille = as.character(data_lezard$SVL_IND)

table(cat_taille)
mode(cat_taille)
cumfreq(cat_taille)

barplot(table(cat_taille),
        main = "Nombres d'individus par valeurs de taille",
        xlab = "Tailles mesurées",
        ylab = "Nombres d'individus"
  )
```

```
rm(cat_taille)
```

## Ré-échantillonnage/échantillonnage d'une variable qualitative

Imaginons que vous avez besoin de présenter vos résultats concernant ces variables catégorielles devant un public d'expert, vous rencontrez alors un problème: vos données sont sensibles et vous ne pouvez pas les transmettre telles quelles. Vous décidez alors de reproduire la variable en question via un échantillonnage aléatoire suivant les mêmes probabilités que votre distribution. Cette manipulation peut être utile dans de nombreux cas, lorsque vous avez besoin de **simuler une distribution** ou d'**échantillonner au sein d'une distribution** donnée, voici comment procéder:

```
sample(data_lezard$POP, 20)
```

```
##  [1] "JOC" "PIM" "PIM" "JOC" "PIM" "JOC" "JOC" "PIM" "JOC" "MON" "COP" "COP"
## [13] "BOU" "JOC" "MON" "PIM" "MON" "JOC" "BOU" "JON"
```

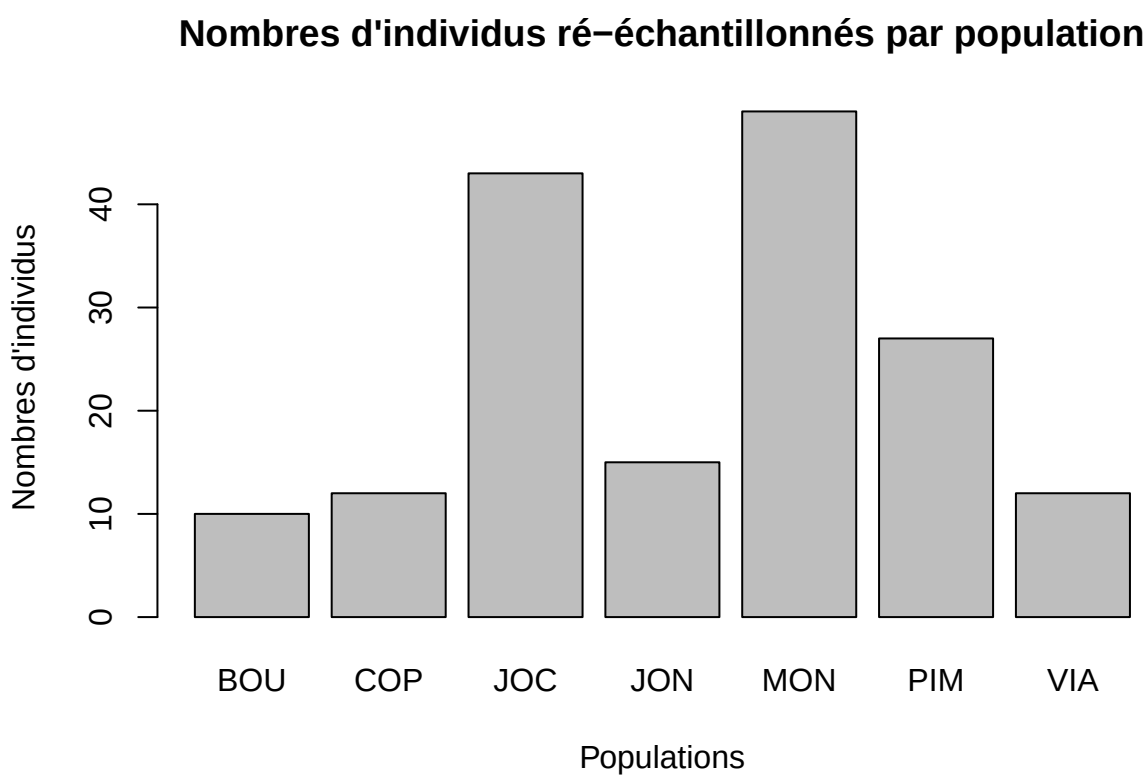
```
# échantillonnage aléatoire de la population de provenance de 20 individus
# (chaque individu a autant de chance d'être choisi qu'un autre et ne peux pas
# être tiré plusieurs fois)

sample(data_lezard$POP, 20, replace = T)
```

```
##  [1] "COP" "MON" "JOC" "JON" "COP" "PIM" "MON" "JOC" "MON" "JOC" "COP" "MON"
## [13] "JOC" "JOC" "JOC" "MON" "JON" "MON" "PIM" "MON"
```

```r
# même échantillonnage mais avec remise: un même individu peut être tiré
# plusieurs fois


resample = sample(data_lezard$POP)
# ré-échantillonnage de tous les individus, leur ordre a été permuté !

resample = sample(data_lezard$POP, replace = T)
# ré-échantillonnage 'bootstrap', très utile lorsqu'il est utilisé de manière
# répété pour estimer la sensibilité des estimateurs statistiques pour
# l'échantillon étudié !



resample = sample(
    names(table(data_lezard$POP)),
    size = length(data_lezard$POP),
    prob = table(data_lezard$POP),
    replace = T
)
# ici on réalise un échantillonnage reproduisant les caractéristiques de notre
# variable d'origine en échantillonnant parmi les valeurs possibles de
# populations avec une probabilité de tirage ('prob') égale à la fréquence de
# chaque classe, et une taille d'échantillonnage ('size') correspondante au
# nombre d'individus dans l'étude

# NB: Dans le dernier cas le vecteur de valeur doit bien être trié de la même
# manière que le vecteur de probabilité ! (d'où l'emploi de
# names(table(data_leazard$POP)), on aurait aussi pu utiliser
# sort(unique(data_lezard$POP)) pour bien ordonner les valeurs par ordre
# alphabétique, comme dans la table)


table(resample)
```

```
## resample
## BOU COP JOC JON MON PIM VIA
##  10  12  43  15  49  27  12
```

```r
barplot(
    table(resample),
    main = "Nombres d'individus ré-échantillonnés par population", xlab = "Populations",
    ylab = "Nombres d'individus"
)
```




```
# La distribution obtenue est effectivement proche de la distribution originale
```

**EXERCICE**

- Créez une variable catégorielle de taille 65 et constituée de quatre classes différentes, de probabilités respectives: 0.1, 0.2, 0.3, et 0.4.
- Vous devez réalisez une expérience complémentaire pour vérifier le comportement d'individus suivis provenant des populations "VIA" et "PIM", mais vous n'avez pas le temps de tous les faire passer en expérience, sélectionner un tiers des individus (par leurs identifiants) de chaque population, sans remise.

```
sample(c("a", "b", "c", "d"), size = 65, prob = c(0.1, 0.2, 0.3, 0.4), replace = T)

c(
  sample(
  data_lezard[data_lezard$POP == "VIA",]$ID_IND,
  length( data_lezard[data_lezard$POP == "VIA", ] )*1/3,
  replace = F ),

  sample(
    data_lezard[data_lezard$POP == "PIM",]$ID_IND,
    length( data_lezard[data_lezard$POP == "PIM", ] )*1/3,
    replace = F )
  )
```

# Décrire une variable quantitative

## Descriptions statistiques

Nous allons maintenant décrire une **variable quantitative**, en prenant ici la masse des juvéniles suivis comme exemple. Dans un premier temps nous allons décrire les **caractéristiques de position et de dispersion** de ce vecteur de valeurs:

```
# paramètres de position de la distribution:

mean(data_lezard$M_IND)
```

```
## [1] 0.1576786
```

```
# masse moyenne des juvéniles

median(data_lezard$M_IND)
```

```
## [1] 0.16
```

```
# masse médiane des juvéniles

quantile(data_lezard$M_IND)
```

```
##   0%  25%  50%  75% 100%
## 0.12 0.14 0.16 0.17 0.21
```

```
# quantile (par défaut quartile) de la distribution

quantile(data_lezard$M_IND, seq = c(0, 1, 0.1))
```

```
##   0%  25%  50%  75% 100%
## 0.12 0.14 0.16 0.17 0.21
```

```
# décile de la distribution

mode(data_lezard$M_IND)
```

```
## [1] "0.17"
```

```
# mode (valeur la plus représentée) de la distribution de masse



# exemples de distribution ne pouvant pas être différenciées sur la base de
# leurs paramètres de position:

mean(c(10, 10, 11, 12, 12)) ==
    mean(c(0, 0, 0, 0, 55))
```

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

```r
mean(c(0, 0, 50, 100, 100)) ==
    mean(c(40, 40, 50, 51, 51))
```

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

```r
mean(c(40, 50, 50, 60, 70)) ==
    mean(c(0, 0, 50, 100, 120))
```

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

```r
median(c(40, 50, 50, 60, 70)) ==
    median(c(0, 0, 50, 100, 120))
```

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

```r
# paramètres de dispersion de la distribution:


min(data_lezard$M_IND)
```

```
## [1] 0.12
```

```r
# valeur minimale de la distribution

max(data_lezard$M_IND)
```

```
## [1] 0.21
```

```r
# valeur maximale de la distribution

range(data_lezard$M_IND)
```

```
## [1] 0.12 0.21
```

```r
# min et max de la distribution

max(data_lezard$M_IND) -
    min(data_lezard$M_IND)
```

```
## [1] 0.09
```

```r
# étendue de la distribution

IQR(data_lezard$M_IND)
```

```
## [1] 0.03
```

```r
# écart inter-quartiles de la distribution
```

```r
var(data_lezard$M_IND)
```

```
## [1] 0.0003305068
```

```r
# variance de la distribution

# NB: la variance d'une distribution correspond à son moment d'ordre 2
```

```r
sd(data_lezard$M_IND)
```

```
## [1] 0.01817985
```

```r
# écart-type (standard deviation) de la distribution

mad(data_lezard$M_IND)
```

```
## [1] 0.014826
```

```r
# median absolute deviation (valeur médiane des écarts absolus à la médiane),
# cette métrique de dispersion est très peu sensible aux outliers (points
# extrêmes) contrairement à la variance ou à l'écart-type (analogue à la
# comparaison moyenne-médiane)
```

```r
summary(data_lezard$M_IND)
```

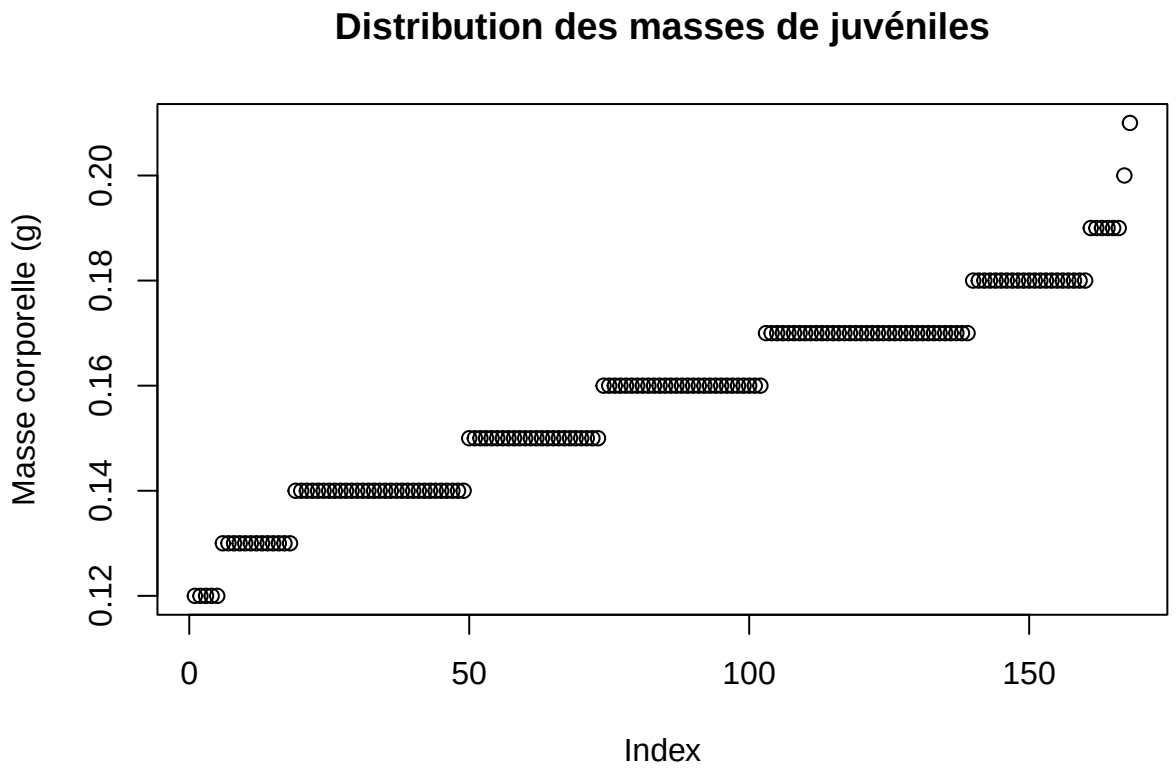
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.1200  0.1400  0.1600  0.1577  0.1700  0.2100
```

```r
# résumé des principaux paramètres de position de la distribution (en y
# ajoutant l'écart-type on obtient les principaux descripteurs usuels d'une
# distribution quantitative)
```

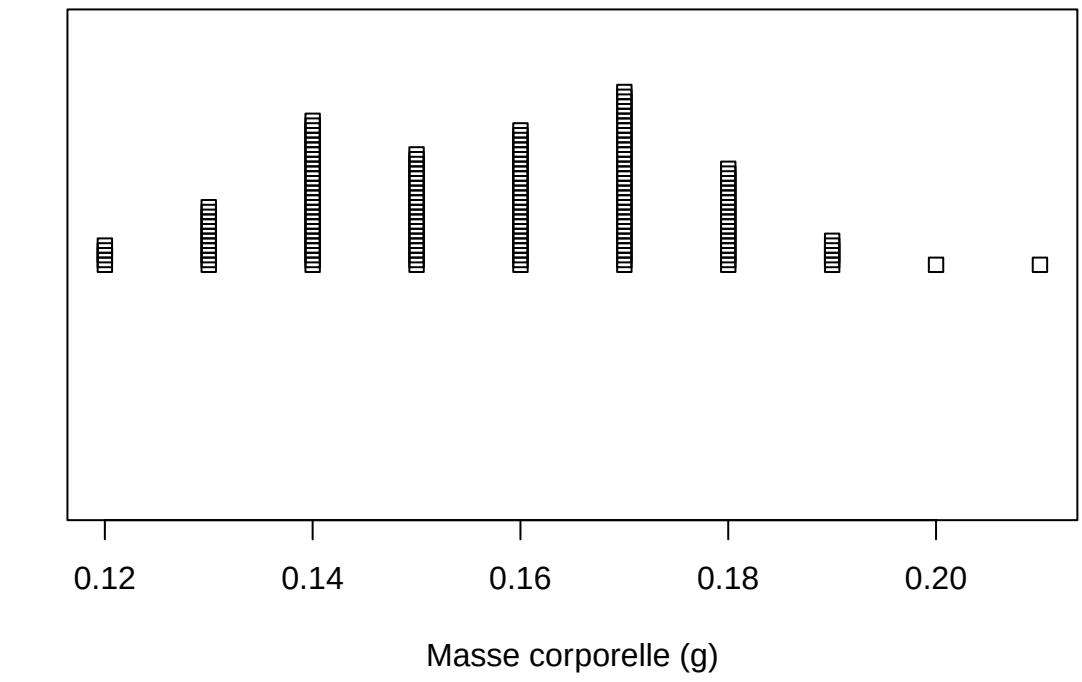
## Descriptions graphiques

Nous allons maintenant décrire graphiquement la distribution que nous venons d'étudier. Les deux modes graphiques les plus utilisés pour la visualisation de variables quantitatives sont les **histogrammes** (permettant d'avoir une idée du type de distribution de probabilité que peut suivre la variable et sa forme globale) et les **boxplots** qui permettent d'avoir une visualisation rapide des paramètres de position et de dispersion de la variable. D'autres visualisation graphiques alternatives permettent également de décrire la distribution. Voici les principaux exemples de sorties graphiques ci-dessous:

```r
plot(
  sort(data_lezard$M_IND),
  main = "Distribution des masses de juvéniles",
  ylab = "Masse corporelle (g)"
  )
```

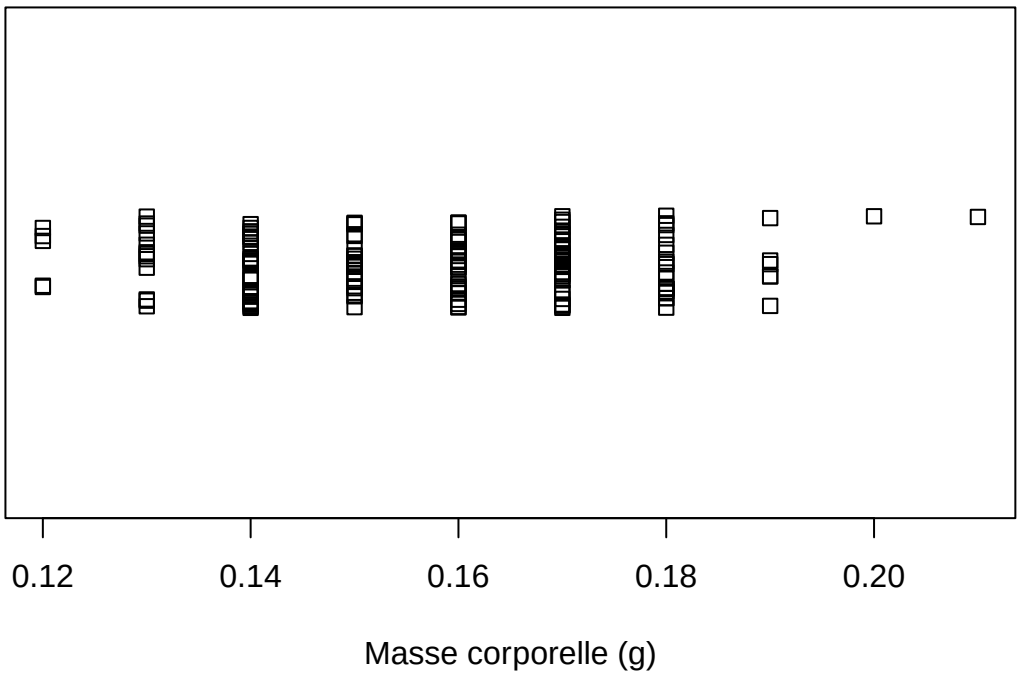



```r
# graphe des valeurs ordonnées de manière croissantes
# (Rq: ne pas oublier l'unité en légende d'axe !)



# autre mode visualisation par "dot plot" (ou "scatter plot"):

stripchart(
  data_lezard$M_IND,
  method = "stack",
  main = "Distribution des masses de juvéniles (empilements des points)",
  xlab = "Masse corporelle (g)",
  offset = 1/8  # zoom in plot
  )
```

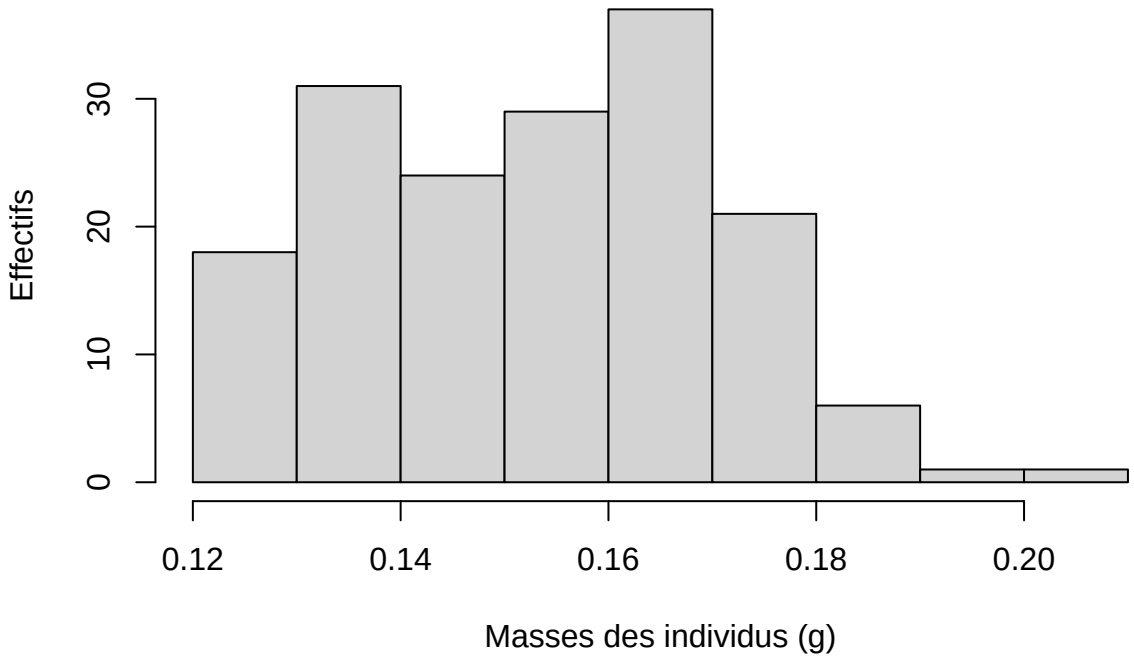



```r
stripchart(
  data_lezard$M_IND,
  method = "jitter",
  main = "Distribution des masses de juvéniles (instabilités des points)",
  xlab = "Masse corporelle (g)"
  )
```

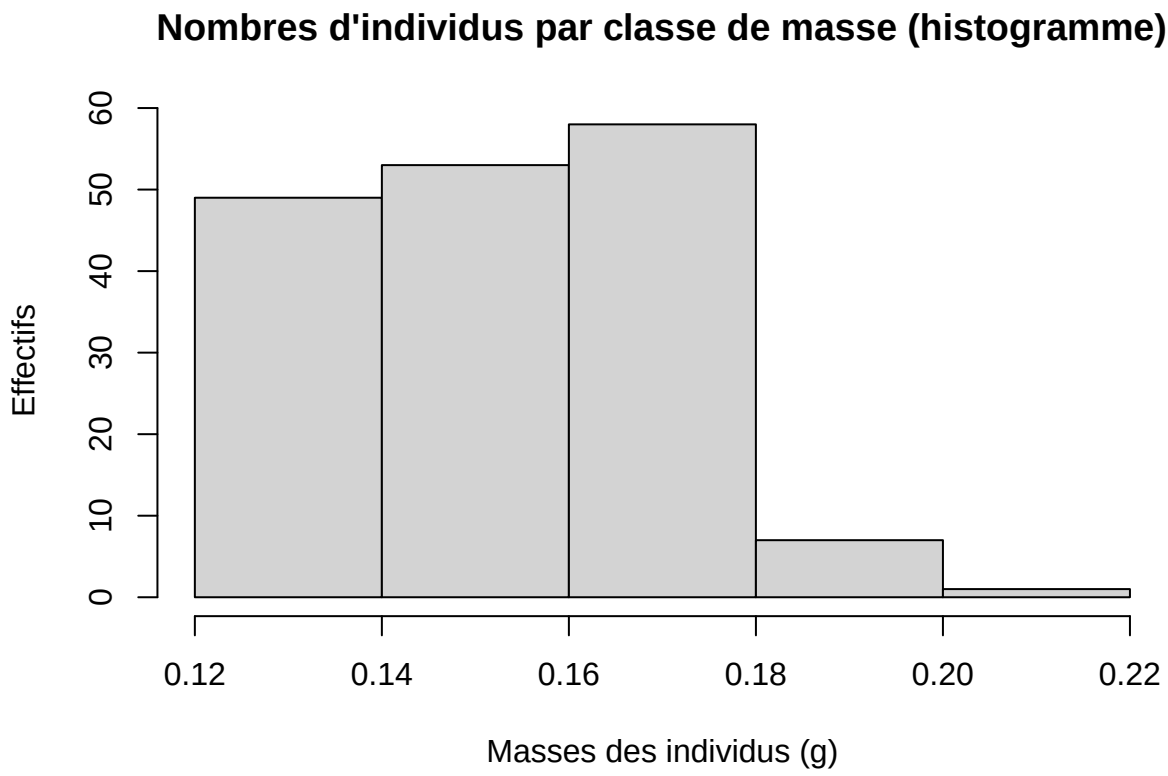
## Distribution des masses de juvéniles (instabilités des points)



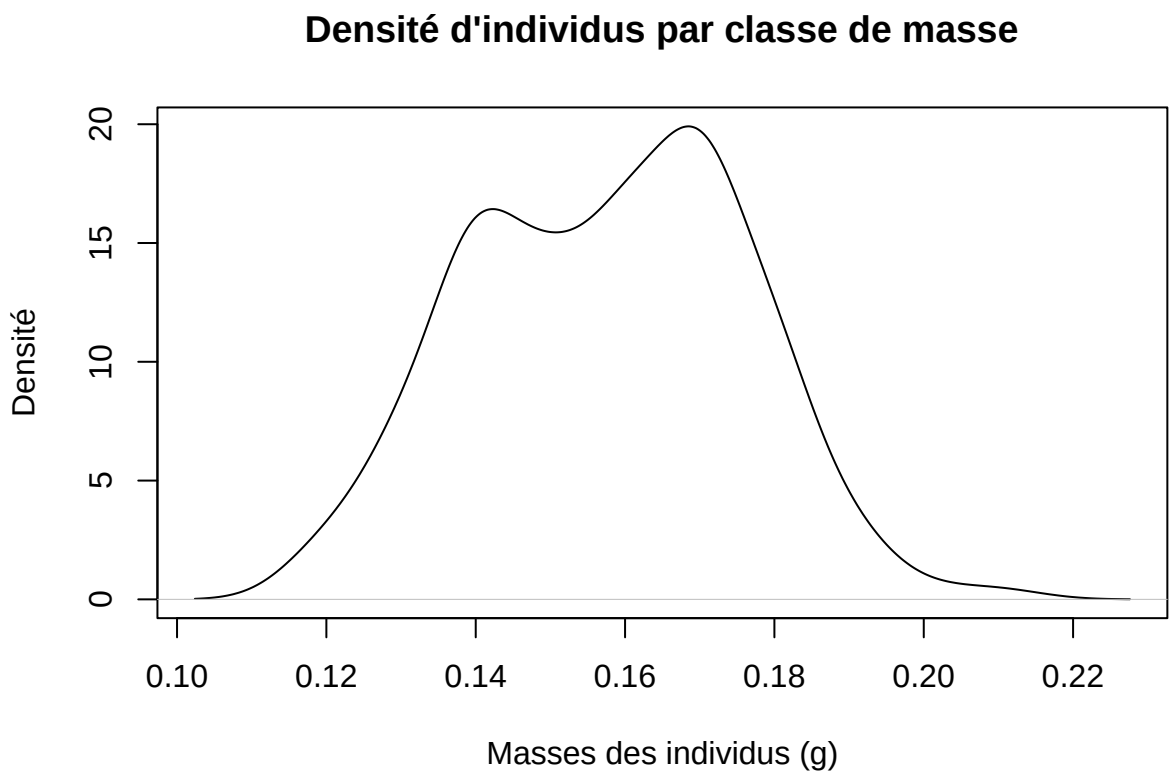

```r
# histogramme et kernels de densité de la distribution:

hist(
  data_lezard$M_IND,
  main = "Nombres d'individus par classe de masse (histogramme)",
  xlab = "Masses des individus (g)",
  ylab = "Effectifs"
  )
```



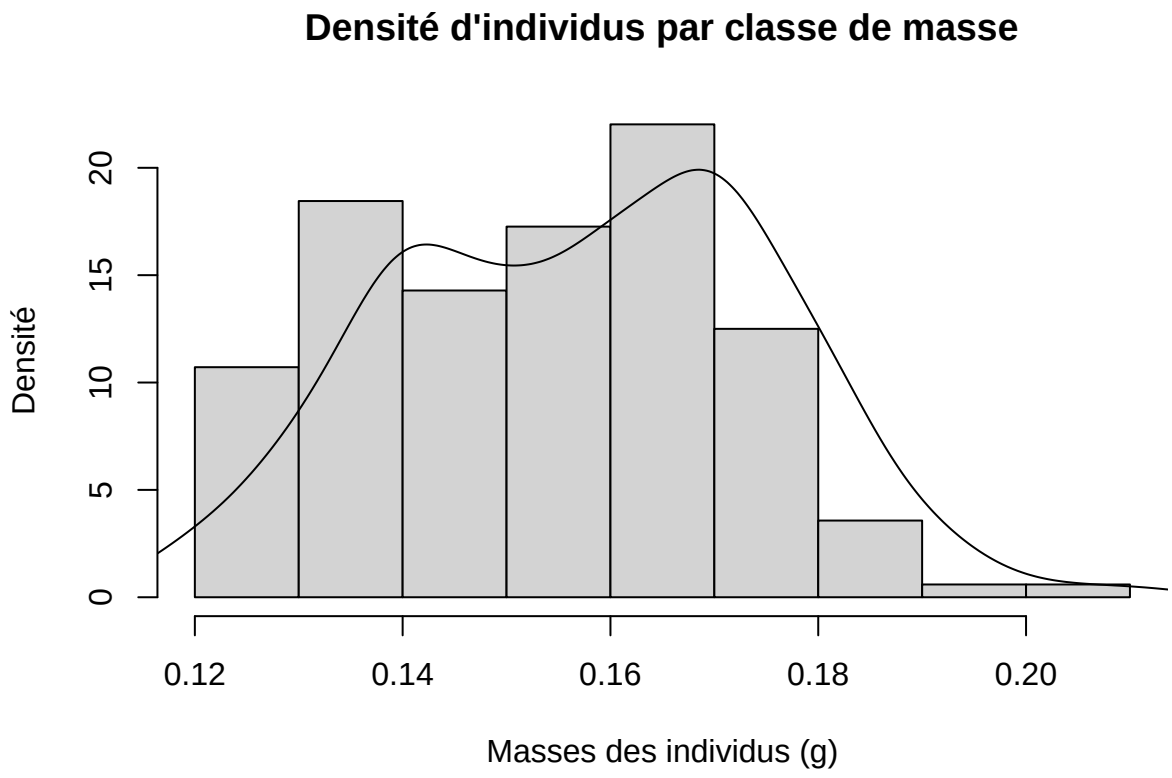

```r
# par défaut visualisation cherchant à optimiser la lisibilité

hist(
  data_lezard$M_IND,
  breaks = seq(0.12, 0.22, 0.02),
  main = "Nombres d'individus par classe de masse (histogramme)",
  xlab = "Masses des individus (g)",
  ylab = "Effectifs"
  )
```



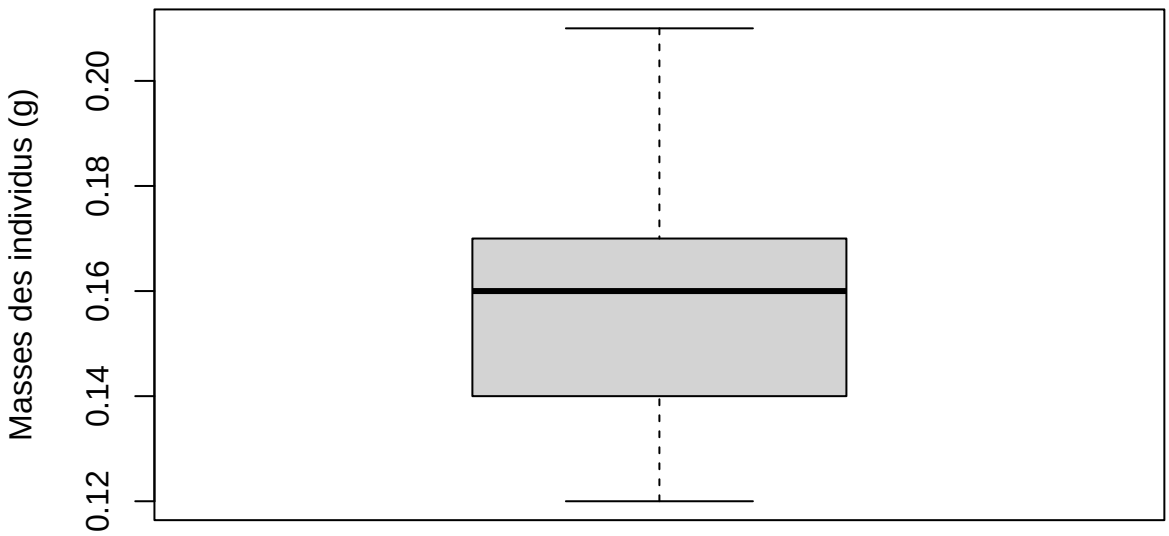

```r
# possibilité de changer manuellement les classes de regroupement pour la visualisation
# (on peut également renseigner le nombre de classe souhaité)

plot(
  density(data_lezard$M_IND),
  main = "Densité d'individus par classe de masse",
  xlab = "Masses des individus (g)",
  ylab = "Densité"
  )
```




```r
# graphe du kernel de densité
# (estimation de l'aire sous la courbe de distribution, avec une aire égale à 1)


# il est possible de combiner des graphes sur une même sortie:

hist(
  data_lezard$M_IND,
  prob = T,
  main = "Densité d'individus par classe de masse",
  xlab = "Masses des individus (g)",
  ylab = "Densité"
  )
# histogramme, avec une modification des valeurs d'effectifs en ordonnée
# tel que l'aire sous la courbe soit égale à 1 (argument "prob = T")
lines(density(data_lezard$M_IND))
```




```
# ajout d'une ligne correspondant à la courbe de densité
# (il est également possible d'ajouter des points avec la fonction "points()"
# ou des graphes entiers en utilisant l'argument "add = T" dans la fonction graphique)




# boîte à moustaches (boxplots):


boxplot(
  data_lezard$M_IND,
  main = "Distribution des masses corporelles des juvéniles (boxplot)",
  ylab = "Masses des individus (g)"
  )
```




```
# pour rappel le boxplot donne les positions (de haut en bas):
# du maximum, du troisième quartile, de la médiane, du premier quartile, du minimum.
# Lorsque les valeurs les plus extrêmes dépassent 1.5 fois la distance inter-quartile
# (en partant du premier ou troisième quartile) on considère les points comme des "outliers"
# et ils sont affichés à part, les barres horizontales les plus externes indiquent alors
# cette distance d'1.5 fois l'écart interquartile (au lieu du minimum et du maximum)
```

Les distributions quantitatives peuvent également être décrite à l'aide de **paramètres de forme**, ils visent à savoir si la distribution est **symétrique** ("skewness" en anglais) et à connaître son degré d'**aplatissement** ("kurtosis" en anglais). Ces caractéristiques de formes sont visibles graphiquement avec l'histogramme (voir paragraphe précédent) mais on peut aussi les formaliser de manière numérique à l'aide de coefficient tel que le coefficient de Fisher (voir exercice ci-dessous).

**EXERCICE**

- Re-créez les fonctions mean() et sd() par vous mêmes
  Rappels:

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \overline{x})^2}$$

- Faites une description statistique et graphique de la masse des mères suivies dans l'étude (attention: une même mère peut apparaître plusieurs fois dans le tableau car certaines ont donné naissance à plusieurs nouveau-nés, veillez à bien éliminer les duplicats !)
- Comparez les deux valeurs suivantes: mean(c(1,2,3,4,50)) et median(c(1,2,3,4,50)), que pouvez-vous en conclure ? Comment décrire (à l'aide d'outils statistiques) la différence entre ces deux distributions ?
- Créez une fonction permettant de calculer les moments d'ordre n
  Rappel:

$$\mu_n = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \overline{x})^n$$

- Créez une fonction permettant de calculez les coefficients de Fisher de symétrie (skewness) et d'aplatissement (kurtosis), sachant qu'ils sont définis de la manière suivante:

$$\gamma_{skewness} = \frac{\mu_3}{s^3}$$

$$\gamma_{kurtosis} = \frac{\mu_4}{s^4} - 3$$

- Si $\gamma_{skewness} = 0$ la distribution est symétrique, si $\gamma_{skewness} > 0$ la courbe est étalée à droite, si $\gamma_{skewness} < 0$ la courbe est étalée à gauche. Si $\gamma_{kurtosis} = 0$ la distribution est mésokurtique (typique d'une loi gaussiennne), si $\gamma_{kurtosis} > 0$ la courbe est leptokurtique (piquée), si $\gamma_{kurtosis} < 0$ la courbe est platykurtique (aplatie). Que pouvez-vous dire de la symétrie et de l'aplatissement de la distribution des masses chez les mères suivies ?

```r
mean_bis <- function(x) {
  sum(x) /
    length(x)
}

sd_bis <- function(x) {
  sqrt(
    sum((x - mean(x))^2) /
      (length(x) - 1)
  )
}



summary(data_lezard[!duplicated(data_lezard$ID_MOTHERS),]$M_MOTHERS)

sd(data_lezard[!duplicated(data_lezard$ID_MOTHERS),]$M_MOTHERS)
```

```r
hist(
  data_lezard[!duplicated(data_lezard$ID_MOTHERS),]$M_MOTHERS,
  main = "Nombres de mères par classe de masse (histogramme)",
  xlab = "Masse corporelle (g)",
  ylab = "Effectifs"
  )
```

```r
boxplot(
  data_lezard[!duplicated(data_lezard$ID_MOTHERS),]$M_MOTHERS,
  main = "Distribution des masses corporelles des mères (boxplot)",
  ylab = "Masse corporelle (g)"
  )
```

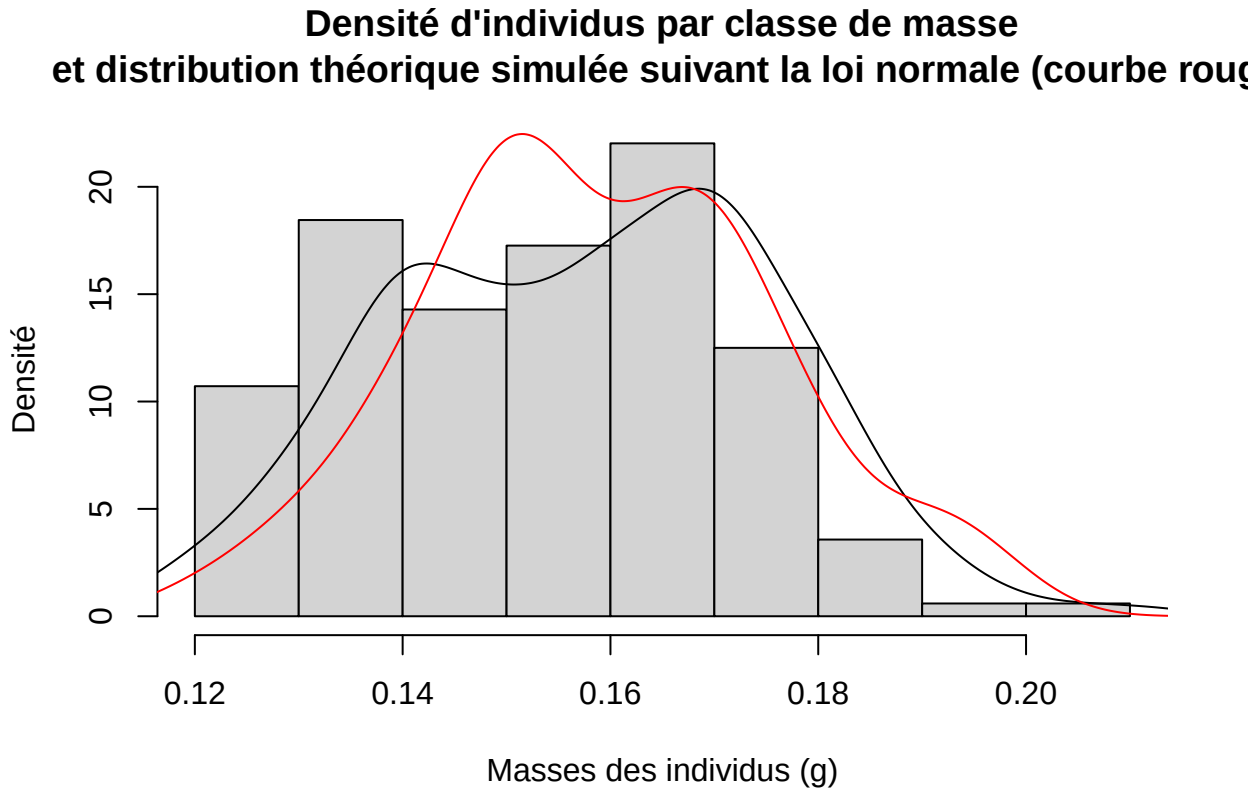
```r
mean(c(1,2,3,4,50))
median(c(1,2,3,4,50))
# sensibilité de la moyenne aux point extrêmes par rapport à la médiane



moment_n <- function(x, n) {
    sum((x - mean(x))^n) /
      (length(x) - 1)
}



gamma_skewness <- function(x) {
  moment_n(x, 3) /
    sd(x)^3
}


gamma_kurtosis <- function(x) {
  moment_n(x, 4) /
    sd(x)^4 - 3
}


gamma_skewness(data_lezard[!duplicated(data_lezard$ID_MOTHERS),]$M_MOTHERS)
gamma_kurtosis(data_lezard[!duplicated(data_lezard$ID_MOTHERS),]$M_MOTHERS)
# la courbe de distribution est asymétrique (étalée vers la droite) et platykurtique (aplatie)

hist(
  breaks=seq(2.5, 5, 0.25),
  data_lezard[!duplicated(data_lezard$ID_MOTHERS),]$M_MOTHERS,
  main = "Nombres de mères par classe de masse (histogramme)",
  xlab = "Masse corporelle (g)",
  ylab = "Effectifs"
  )
```

```r
# plus visible sur cette figure
```

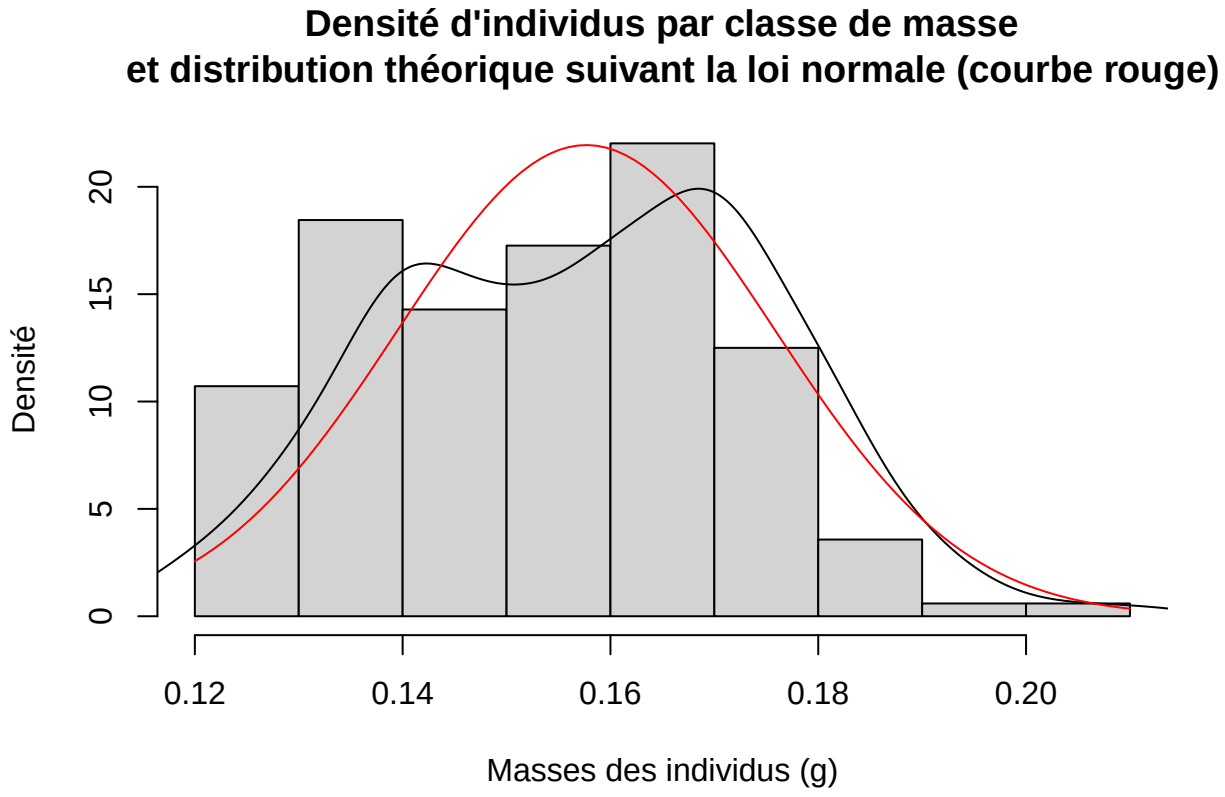
### Simulation de distribution théorique et comparaison graphiques avec des distributions existantes

Il est possible dans R de **simuler** des valeurs suivant une **loi de de probabilité** donnée. Ces simulations se réalisent en utilisant les fonction de la forme rmaloi (en remplaçant "maloi" par le nom de la distribution souhaitée: par "norm" pour une loi normale, "binom" pour une loi de Bernoulli, "pois" pour une loi de Poisson, etc.). On peut également obtenir la densité, fonction de répartition et les quantiles de la distribution à l'aide des fonctions de la forme "dmaloi", "pmaloi", "qmaloi". Ces fonctions peuvent être très utile pour comparer une distribution existante à une distribution théorique ou tout simplement pour simuler une telle distribution théorique en soi. Nous allons ici prendre un exemple à partir d'une simulation de loi normale:

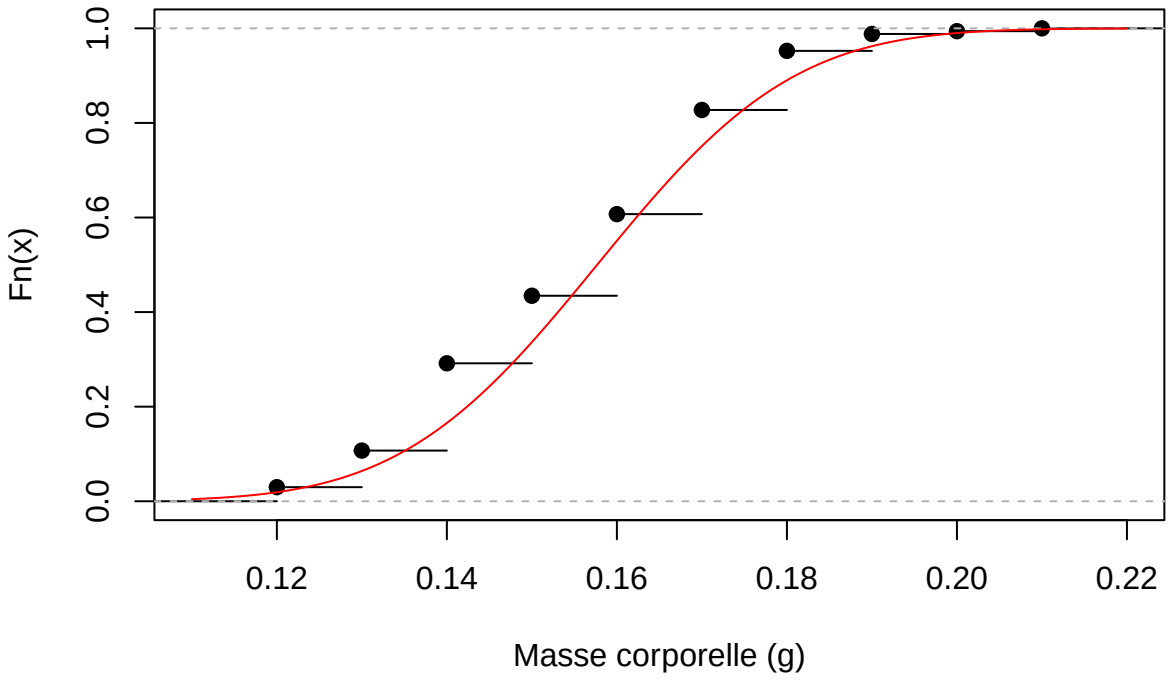
```r
# nous allons simuler un loi de distribution cherchant à se rapprocher de la distribution
# des masses de lézard:

sim_norm = rnorm(length(data_lezard$M_IND), mean(data_lezard$M_IND), sd(data_lezard$M_IND))
# simulation contenant le même nombre d'individus que dans le jeu de données, et dont les
# paramètres de distribution (moyenne, écart-type) sont les mêmes que la variable étudiée


# nous allons graphiquement comparer les deux distributions (théoriques et observés):

hist(
  data_lezard$M_IND,
  prob = T,
  main = "Densité d'individus par classe de masse
  et distribution théorique simulée suivant la loi normale (courbe rouge)",
  xlab = "Masses des individus (g)",
  ylab = "Densité"
  )
# histogramme, avec une modification des valeurs d'effectifs en ordonnée telle que
# l'aire sous la courbe soit égale à 1 (argument "prob = T")
lines(density(data_lezard$M_IND))
lines(density(sim_norm), col = "red")
```




```r
# les deux distributions sont assez proches
# NB: l'argument "col" permet de choisir la couleur du figuré principal de la figure /
# insérer un retour à la ligne dans du texte permet de faire apparaître un retour à la ligne
# (on peut aussi insérer la chaîne de caractère "\n" pour cela)


hist(
  data_lezard$M_IND,
  prob = T,
  main = "Densité d'individus par classe de masse
  et distribution théorique suivant la loi normale (courbe rouge)",
  xlab = "Masses des individus (g)",
  ylab = "Densité"
  )
lines(density(data_lezard$M_IND))
curve(dnorm(x, mean(data_lezard$M_IND), sd(data_lezard$M_IND)), add=T, col = "red")
```




```
# alternative: pas de simulation, tracé direct de la fonction de densité théorique
# NB: la fonction "curve()" permet de tracer une courbe suivant une fonction prenant
# en argument "x" l'étendue de la fenêtre graphique actuelle


plot(ecdf(data_lezard$M_IND),
     main="Fonction de répartition observé et théorique (en rouge)",
     xlab = "Masse corporelle (g)")
curve(pnorm(x, mean(data_lezard$M_IND), sd(data_lezard$M_IND)),
      add=T,
      col="red")
```



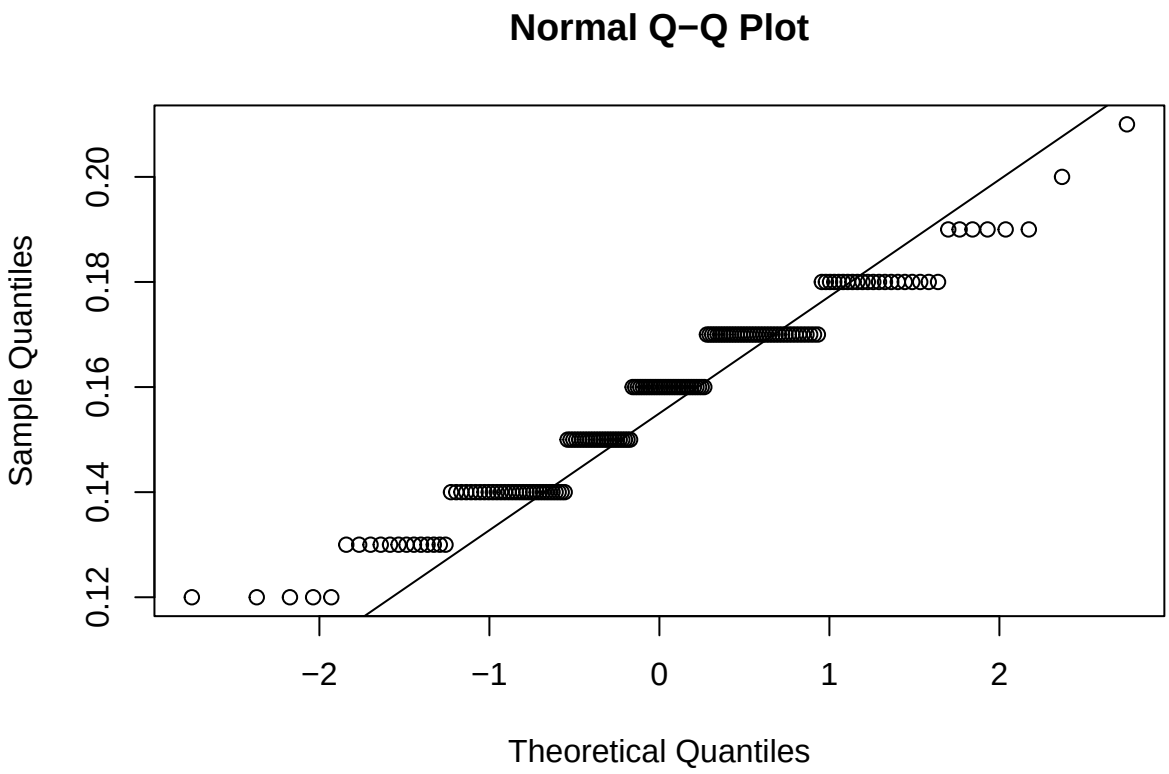

```
# la fonction "ecdf()" permet de tracer la fonction de répartition empirique
# à partir des données fournies




# la comparaison peut également se faire sur les quantiles:


qnorm(c(0.25, 0.5, 0.75), mean(data_lezard$M_IND), sd(data_lezard$M_IND))
```

```
## [1] 0.1454165 0.1576786 0.1699407
```

```
summary(data_lezard$M_IND)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.1200  0.1400  0.1600  0.1577  0.1700  0.2100
```

```
qqnorm(data_lezard$M_IND)
# graphe de comparaison des quantiles théoriques et observés
qqline(data_lezard$M_IND)
```




```
# droite théorique que devraient suivre les valeurs en cas d'adéquation avec la loi normale


# l'adéquation à une loi normale est plutôt bonne même si on observe des déviations
# sur les queues de distribution
```

**EXERCICE**

- Étudiez graphiquement l'adéquation de la distribution des masses corporelles des mères à une loi normale.

**ASTUCES**

Après avoir exporté les sorties graphiques d'intérêt, vous pouvez supprimer les graphiques de votre interface R (dans le panel en bas à droite). Vous pouvez faire le choix de les supprimer un à un ou en entièreté (attention: cette opération est irréversible), via les onglets en haut à gauche du panel (voir image ci-dessous pour les détails). Vous pouvez également naviguer parmi les différentes figures produites à l'aide des flèches de la fenêtre graphique. Enfin, vous pouvez zoomer sur le graphique affiché (ouvre une fenêtre à part) en cliquant sur la loupe.

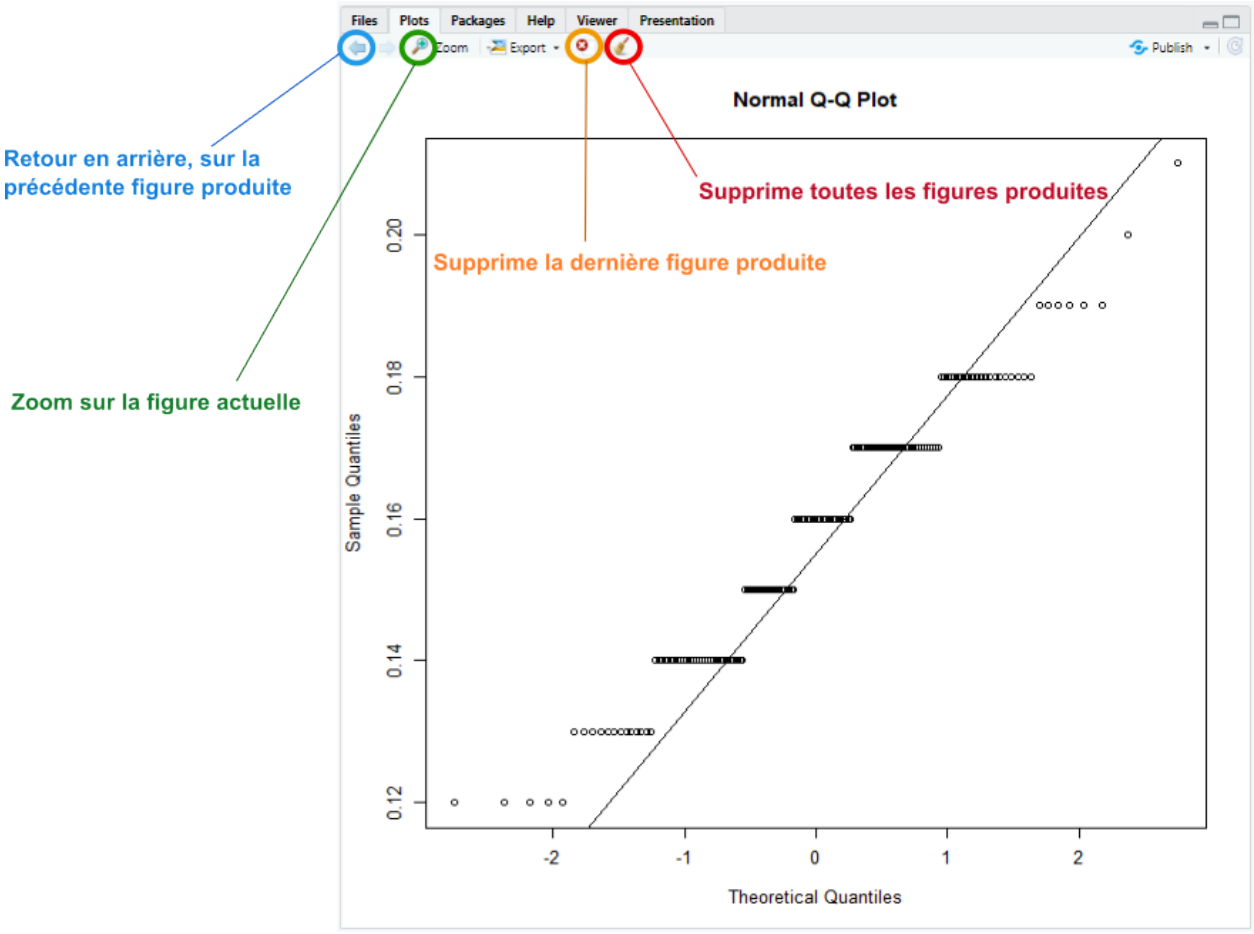
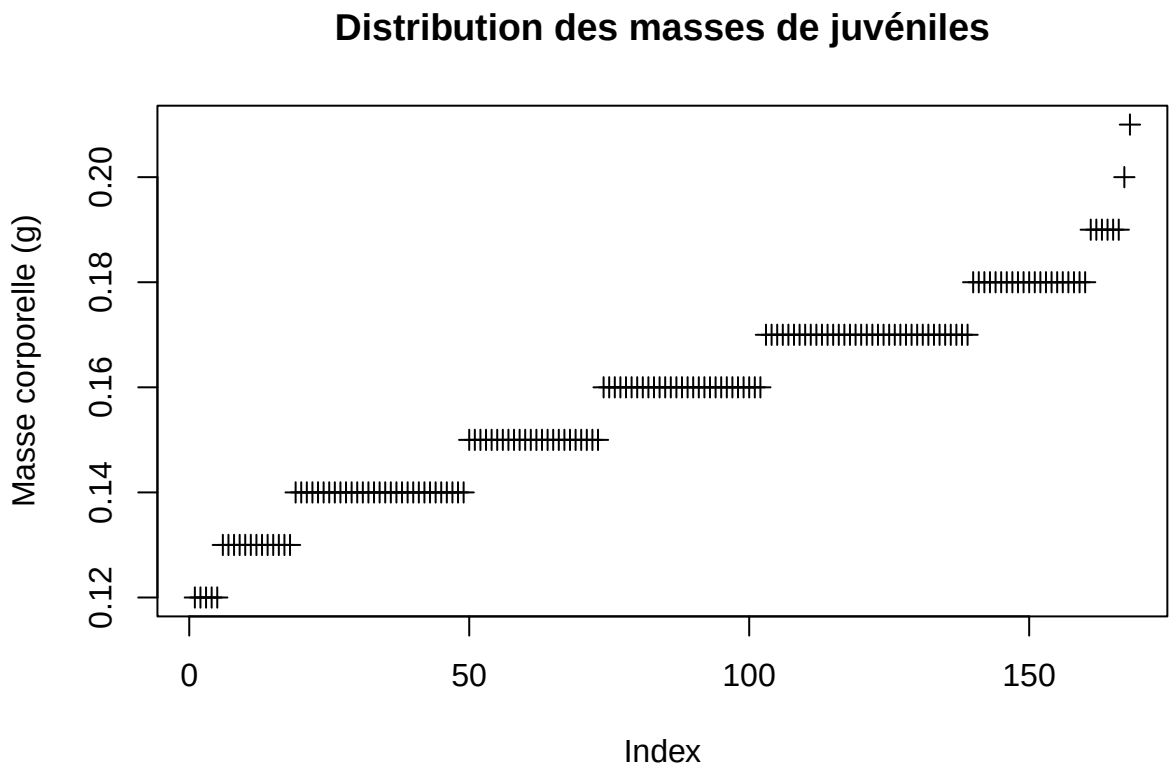



Figure 2: Description interface graphique

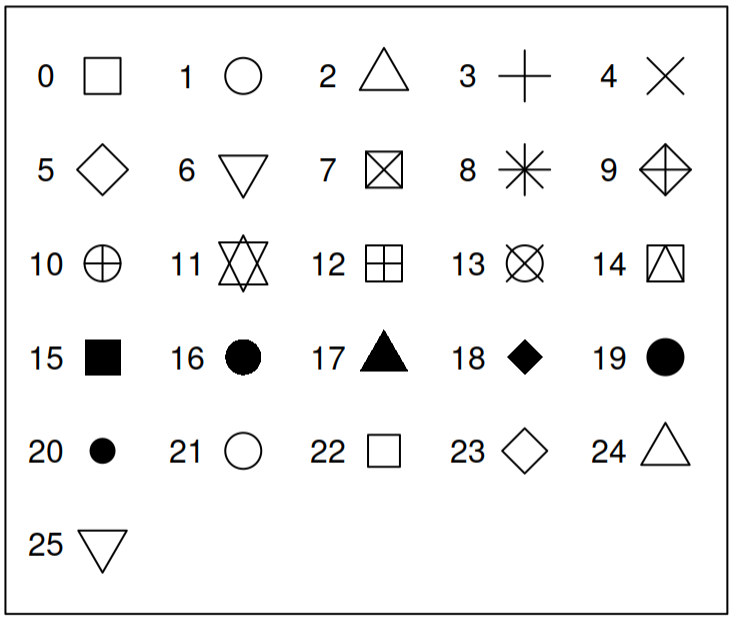
## Paramètres graphiques avancés

Les graphiques que nous avons produits jusqu'à maintenant sont relativement simples, mais il existe de nombreuses options graphiques dans R permettant de **personnaliser et mettre en forme** ses graphiques. Ces options permettent notamment de gérer la couleurs, la taille et le style des figurés, d'ajouter du texte au graphique, de mettre en forme tout texte apparaissant dans le graphique, ou encore de modifier les axes et les marges du graphique. Voici quelques exemples des principales options existantes:

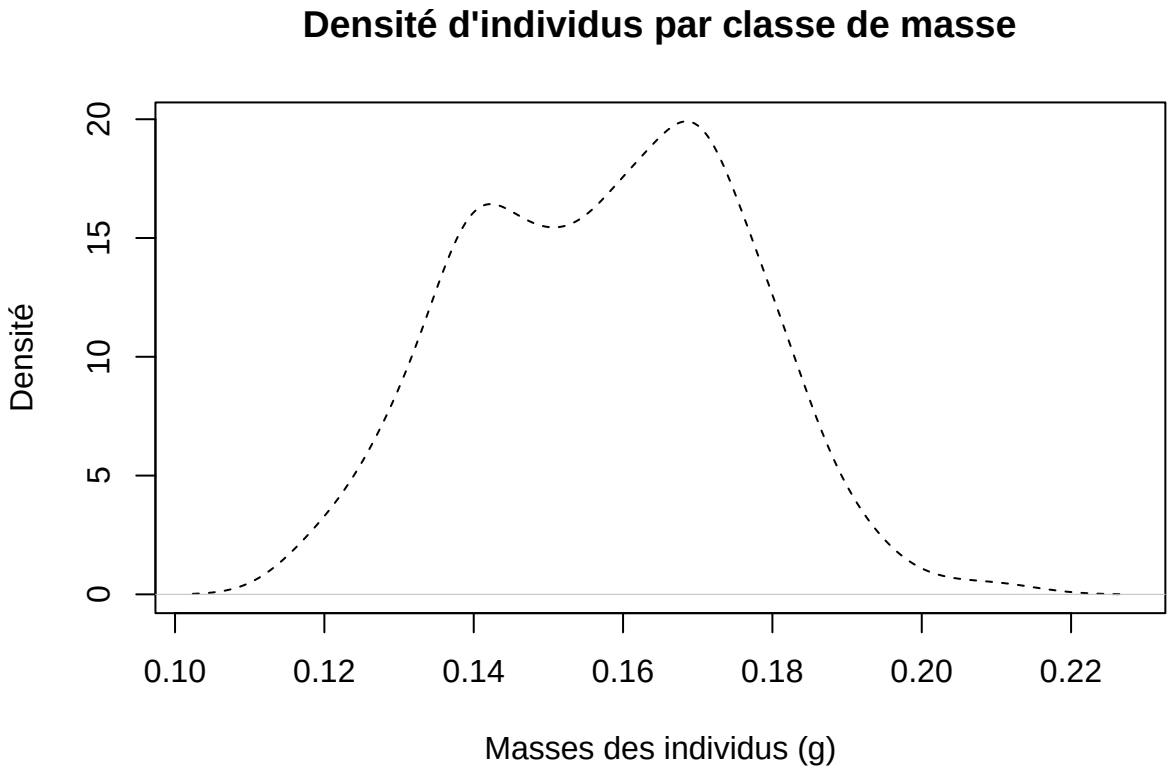
```
# modification des figurés:

## symbole:

plot(
sort(data_lezard$M_IND),
main = "Distribution des masses de juvéniles",
ylab = "Masse corporelle (g)",
pch = 3
)
```



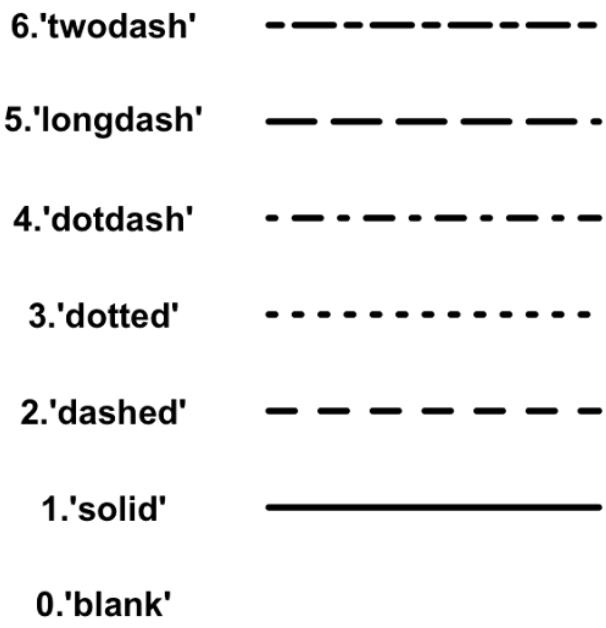

```
# pour les points l'argument "pch" permet de gérer le type de symbole,
# vous pouvez trouver ci-dessous les différents symboles utilisables:
```



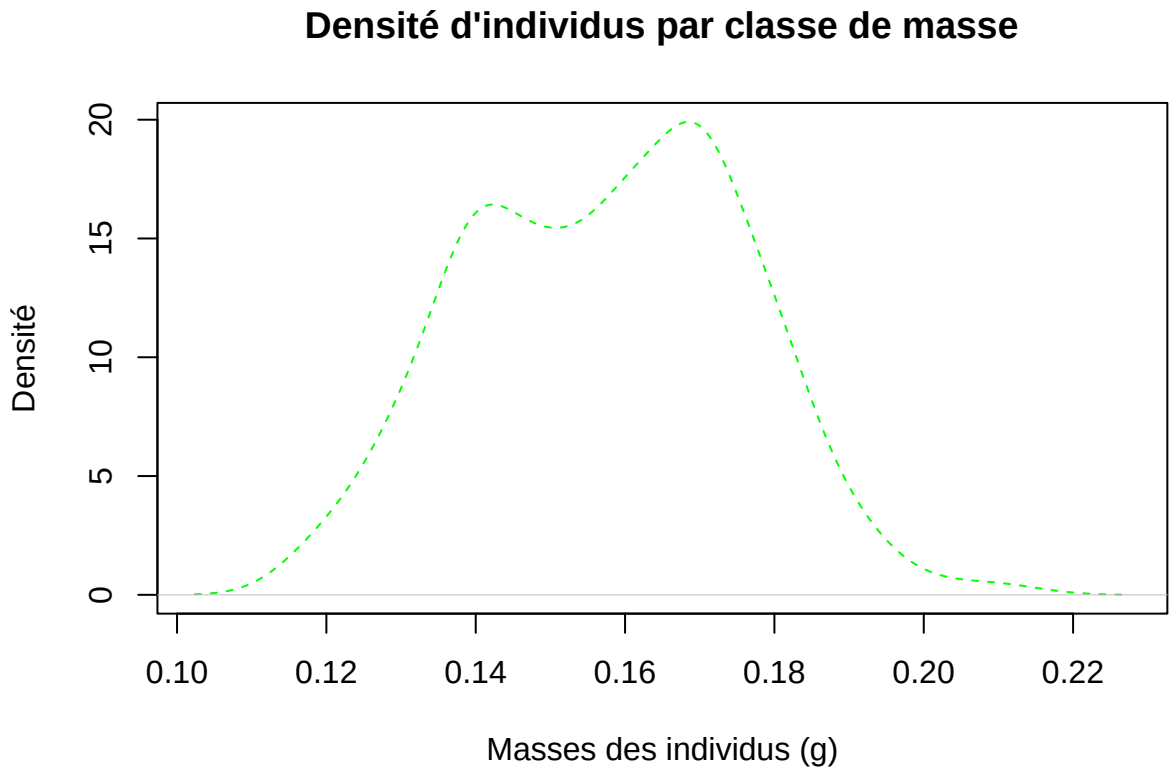

```
plot(
density(data_lezard$M_IND),
main = "Densité d'individus par classe de masse",
xlab = "Masses des individus (g)",
ylab = "Densité",
lty = "dashed"
)
```




```
# pour les lignes l'argument "lty" permet de gérer le type de symbole,
# vous pouvez trouver ci-dessous les différents symboles utilisables:
```

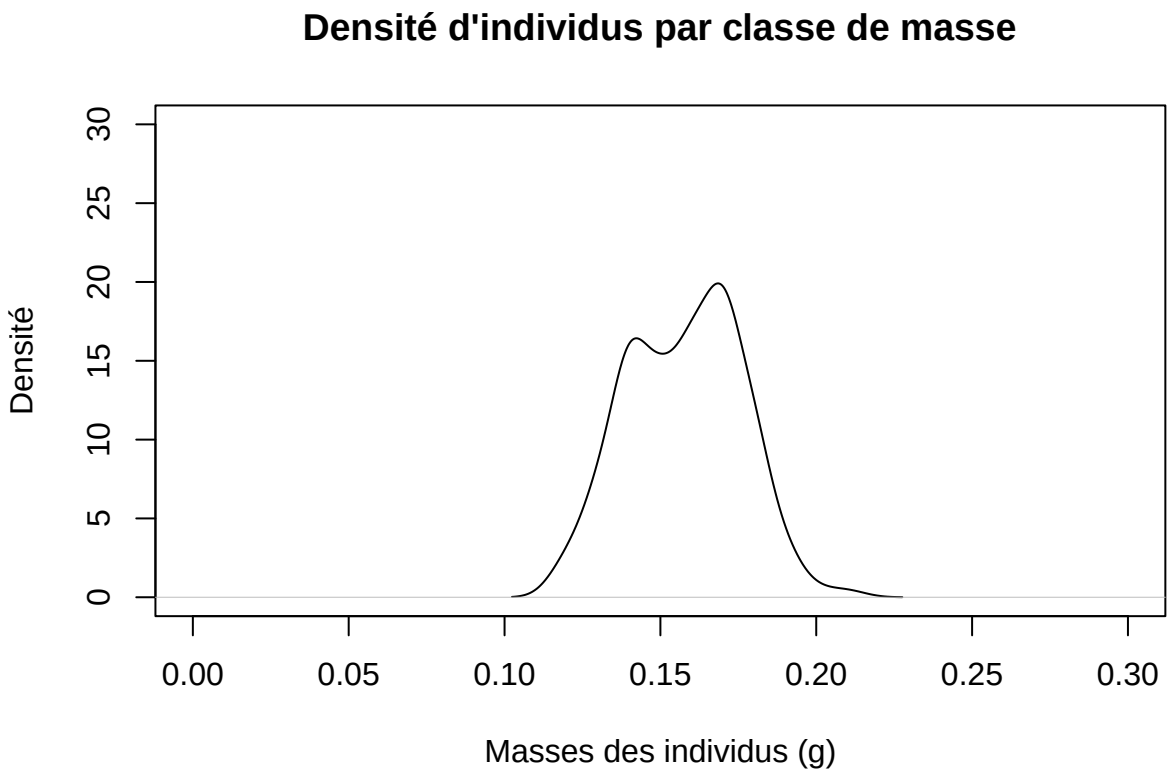


```r
# couleurs:

plot(
density(data_lezard$M_IND),
main = "Densité d'individus par classe de masse",
xlab = "Masses des individus (g)",
ylab = "Densité",
lty = "dashed",
col = "green"
)
```

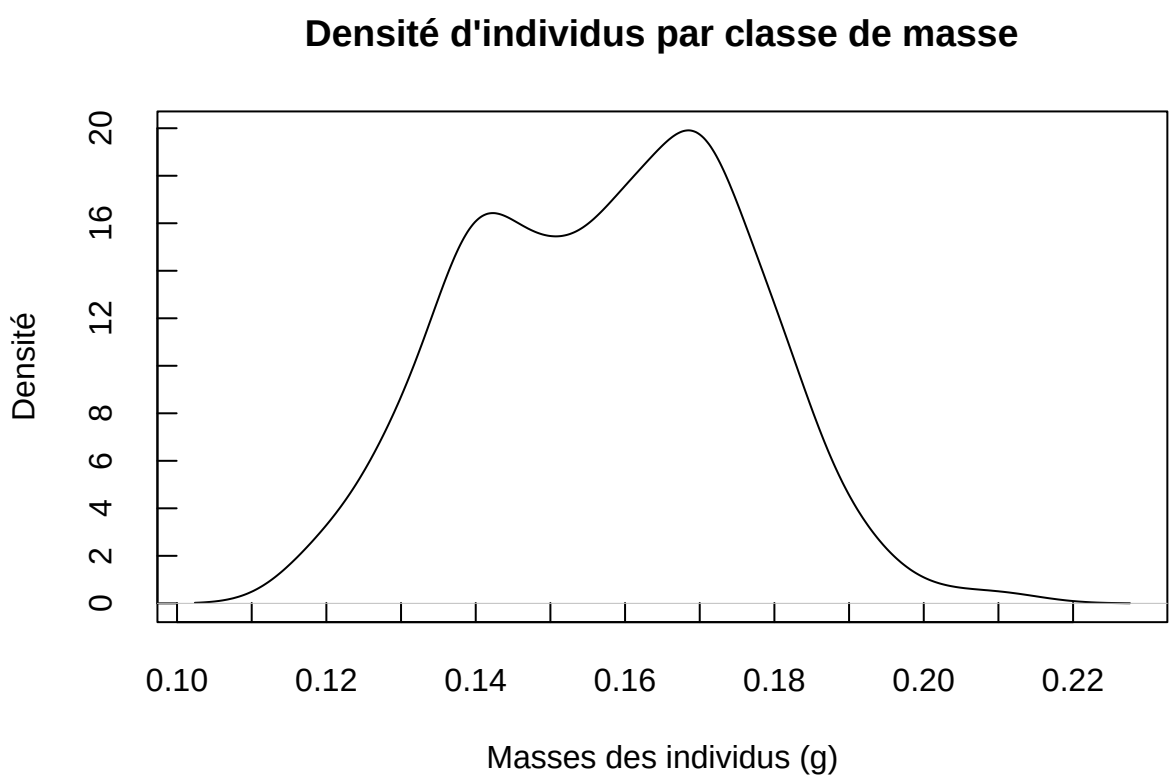



```r
# les couleurs sont gérées à l'aide l'argument "col",
# voici les couleurs pouvant être utilisés dans R:
```


```
# modification des axes:


plot(
density(data_lezard$M_IND),
main = "Densité d'individus par classe de masse",
xlab = "Masses des individus (g)",
ylab = "Densité",
xlim = c(0, 0.3),
ylim = c(0, 30)
)
```

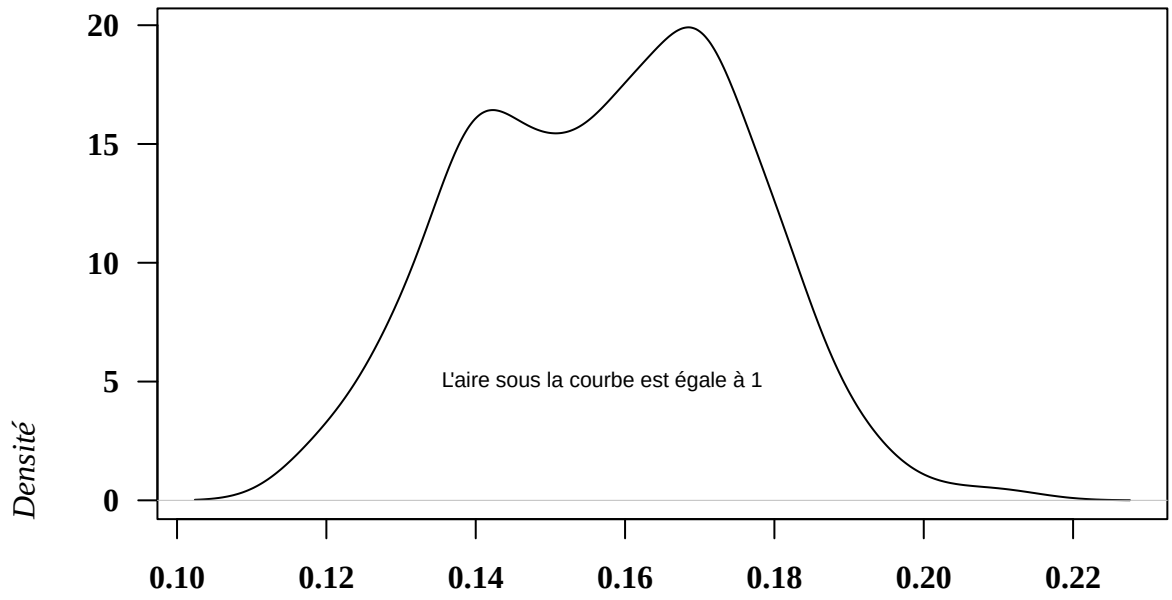



```r
# xlim et ylim définissent les intervalles des axes horizontaux et verticaux
# (peut aussi être géré avec l'argument "asp" qui gère le ratio d'apparence entre axes)



plot(
density(data_lezard$M_IND),
main = "Densité d'individus par classe de masse",
xlab = "Masses des individus (g)",
ylab = "Densité",
xaxp = c(0.1, 0.22, 12),
yaxp = c(0, 20, 10),
tcl = 0.5
)
```



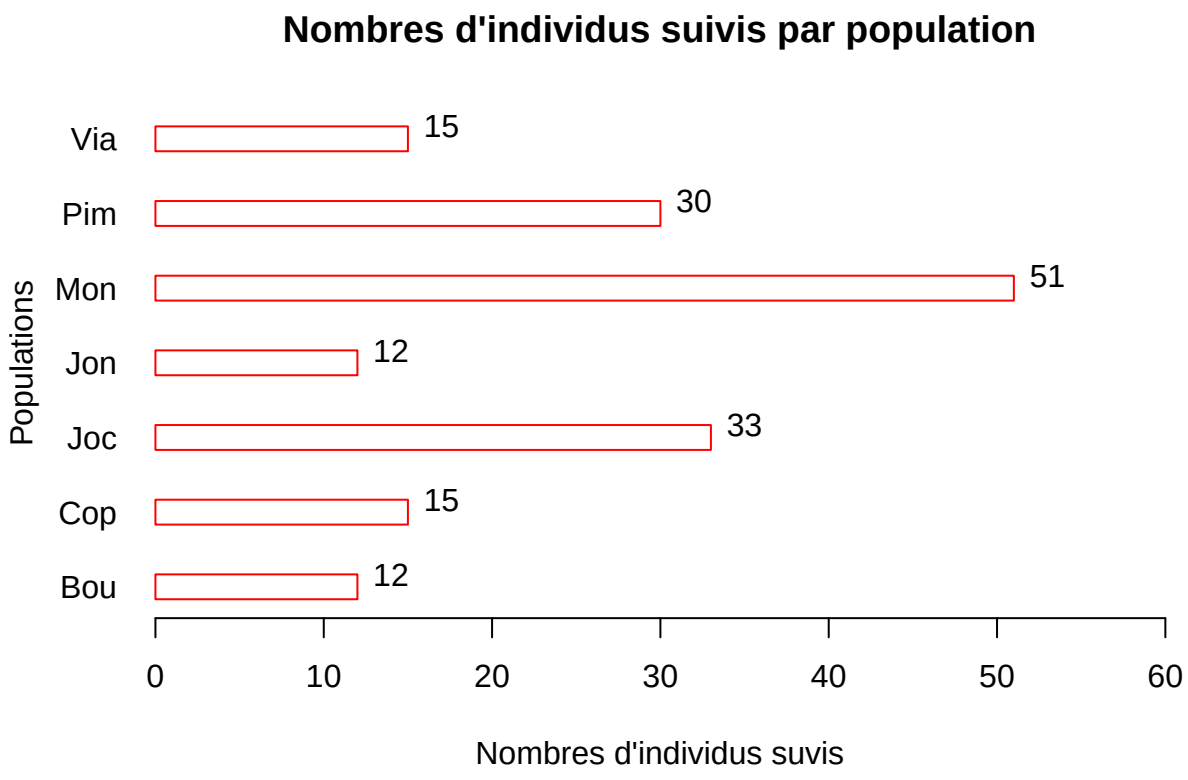

```r
# les arguments "xaxp" et "yaxp" permettent de gérer les positions des encoches
# et légendes associées sur les axes (on indique dans l'ordre: début, fin, nombre d'encoches),
# l'argument "tcl" permet de gérer la taille des encoches
# (par défaut: -0.5; supprime les encoches si égal à 0)




# ajout/gestion texte:


plot(
    density(data_lezard$M_IND),
    main = "Densité d'individus par classe de masse",
    xlab = "Masses des individus (g)",
    ylab = "Densité",
    cex.main = 1.5,
    font.lab = 3,
    font.axis = 2,
    family = "serif",
    las = 1,
    adj = 0
)
# les différents arguments présentés ici permettent de gérer l'aspect du texte contenu
# dans le graphique (soit pour tous les textes du graphique si on n'indique pas de point médian,
```

```r
# soit sur le titre / sous-titre / légendes / axes en indiquant l'élément après le point):
# cex.main/sub/lab/axis: taille de la police
# font.main/sub/lab/axis: effet de police
# (1: normal, 2: gras, 3: italique, 4: gras et italique)
# family.main/sub/lab/axis: type de police
# las: orientation du texte sur les axes (0: parallèle à l'axe, 1: horizontal,
# 2: perpendiculaire à l'axe, 3: vertical)
# adj: positionnement du texte des légendes et titres
# (0: à gauche, 0.5: centré, 1: à droite)


mtext(side = 1,
      line = 4,
      adj = 1,
      "La masse a été mesuré à l'aide d'une balance de précision.")
text(0.157, 5 , "L'aire sous la courbe est égale à 1", cex = 0.7)
```

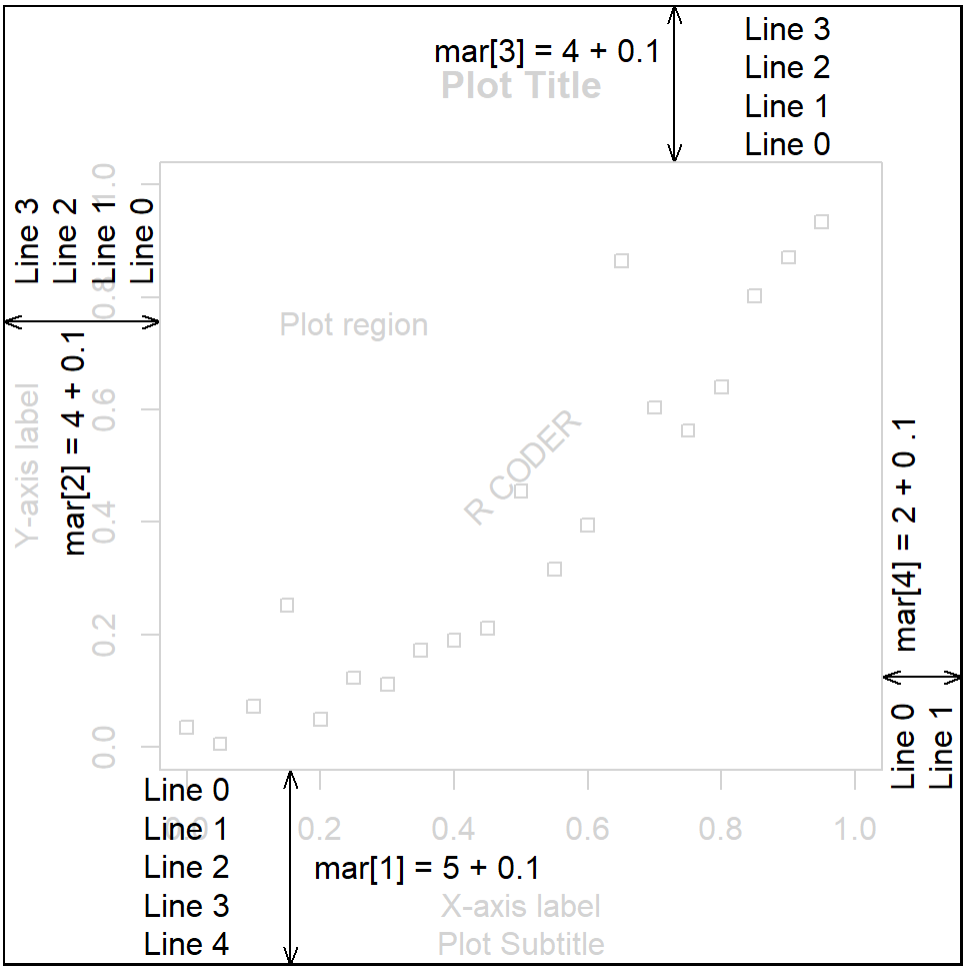



```r
# ajout de texte autour du graphe (mtext: "side" indique la région de localisation,
# en bas: 1, à gauche: 2, en haut: 3, à droite: 4; "line" indique la distance,
# en nombre de lignes, depuis l'axe) ou dans le graphe (text: indication de
# la ou les positions du ou des textes, puis du texte)
```

```r
# gestion barplot / boxplot :

barplot(
  table(data_lezard$POP),
  main = "Nombres d'individus suivis par population",
  xlab = "Nombres d'individus suvis",
  ylab = "Populations",
  xlim = c(0,60),
  space = 2,
  horiz = T,
  border = "red",
  col = "white",
  names.arg = c("Bou", "Cop", "Joc", "Jon", "Mon", "Pim", "Via"),
  las = 1)
# "space" permet de gérer l'espacement entre les barres / "horiz" permet de faire
# apparaître le graphique à l'horizontale / "border" permet de gérer la couleur des
# lignes / "col" permet de gérer la couleur du remplissage / "names.arg" permet de
# gérer le nom des légendes de catégories
# NB: pour boxplot remplacer "horiz" par "horizontal", "names.arg" par "names" et
# utiliser "width" et non pas "space" pour gérer la largeur des boxplots

text(x = table(data_lezard$POP)+2,
     y = 1:7*3,
     labels = table(data_lezard$POP)
       )
```

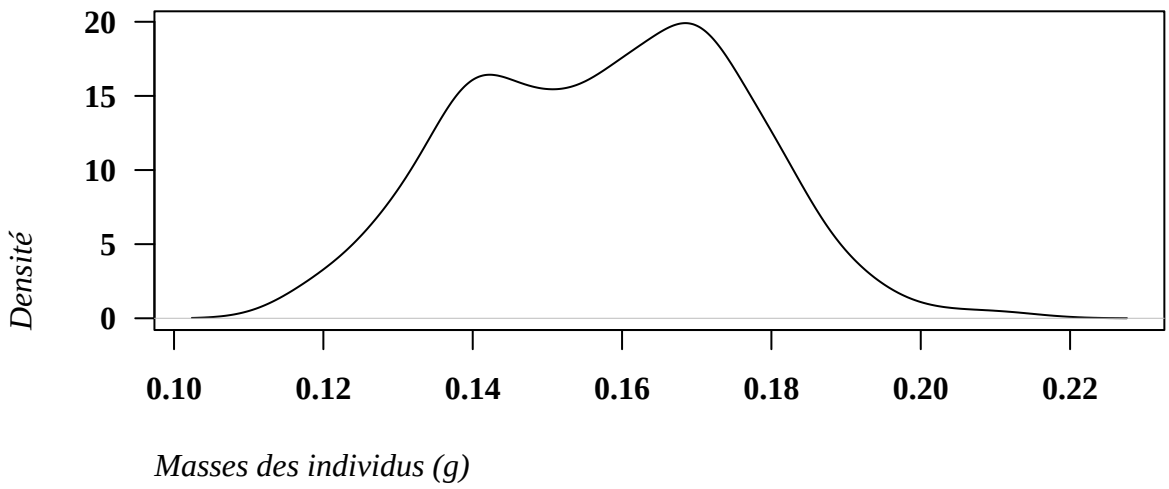


```r
# autre exemple d'ajout de textes (multiples)




# marge de fenêtre graphique:


# La fonction "par()" vous permet d'afficher tous les paramètres graphiques actuels:
# nous allons particulièrement nous intéresser à l'argument "mar", les marges du graphique
# (voir illustration ci-dessous)
```




```r
# il peut être parfois utile de jouer avec ce paramètre lorsque vous utilisez des portions
# de textes longues ou placées dans les parties externes du graphique,
# voici un exemple ci-dessous:


par(mar = c(10.1, 4.1, 4.1, 2.1))
# changement des paramètres graphiques portant sur la taille des marges,
# de la forme mar=c(bas, gauche, haut, droite)
```

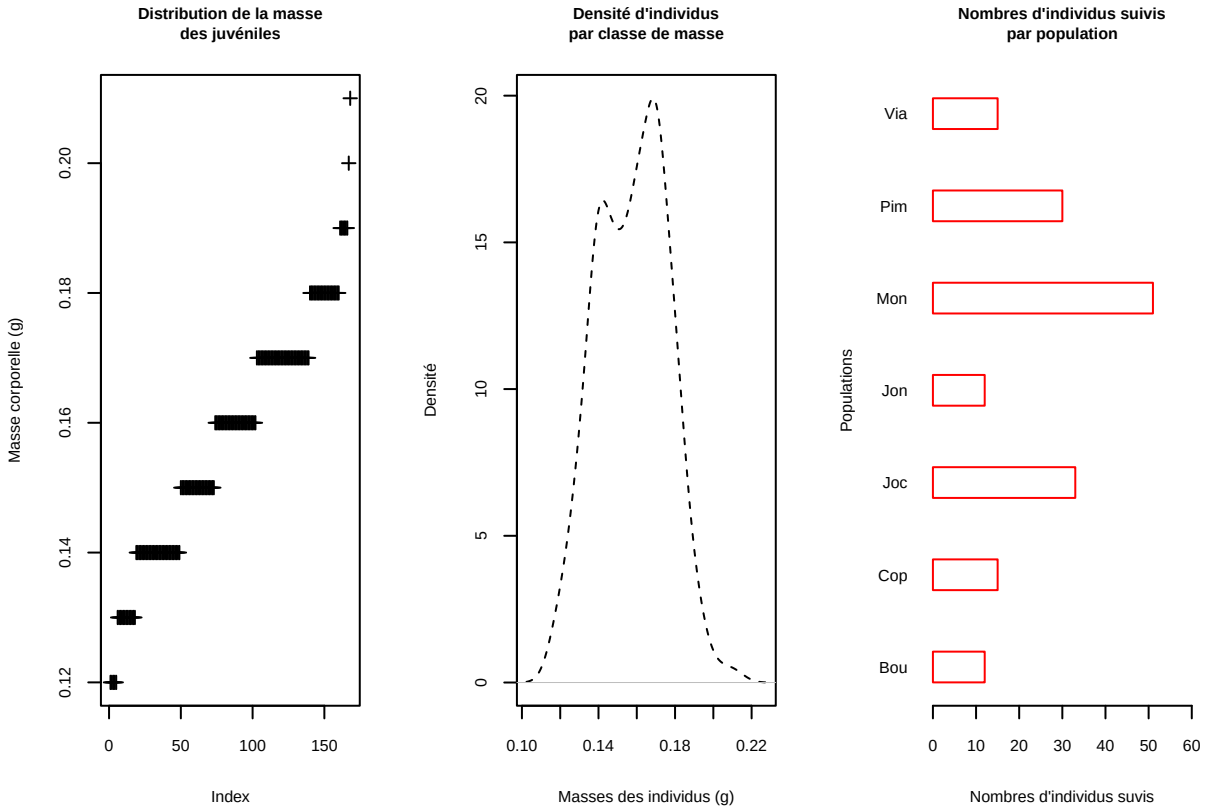
```r
plot(
    density(data_lezard$M_IND),
    main = "Densité d'individus par classe de masse",
    xlab = "Masses des individus (g)",
    ylab = "Densité",
    cex.main = 1.5,
    font.lab = 3,
    font.axis = 2,
    family = "serif",
    las = 1,
    adj = 0
)

mtext(side = 1,
      line = 8,
      adj = 1,
      "La masse a été mesuré à l'aide d'une balance de précision.")
```



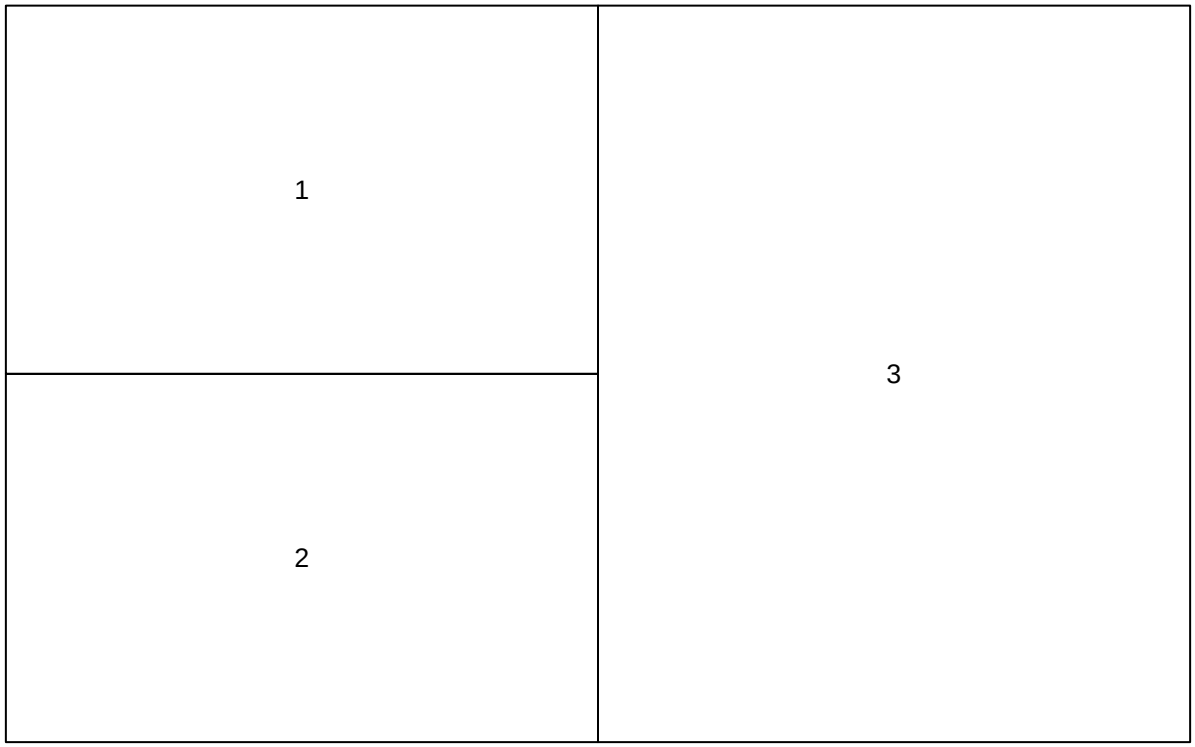

La masse a été mesuré à l'aide d'une balance de précision.

```r
par(mar = c(5.1, 4.1, 4.1, 2.1))
# ne pas oublier de remettre la valeur par défaut pour éviter d'avoir des graphiques mal
# mis en forme sur les prochaines sorties
```

```r
# panels de figures:

par(mfrow=c(1, 3))
# modification du nombre de figures par sortie graphique (nombre de ligne de figure,
# nombre de figure par ligne), ici on définit donc que toutes les prochaines sorties
# contiendront une ligne de trois graphiques

plot(
sort(data_lezard$M_IND),
main = "Distribution de la masse\ndes juvéniles",
ylab = "Masse corporelle (g)",
pch = 3,
cex.axis = 0.8,
cex.lab=0.8,
cex.main=0.8)

plot(
density(data_lezard$M_IND),
main = "Densité d'individus\npar classe de masse",
xlab = "Masses des individus (g)",
ylab = "Densité",
lty = "dashed",
cex.axis = 0.8,
cex.lab=0.8,
cex.main=0.8)


barplot(
  table(data_lezard$POP),
  main = "Nombres d'individus suivis\npar population",
  xlab = "Nombres d'individus suvis",
  ylab = "Populations",
  xlim = c(0,60),
  space = 2,
  horiz = T,
  border = "red",
  col = "white",
  names.arg = c("Bou", "Cop", "Joc", "Jon", "Mon", "Pim", "Via"),
  las = 1,
  cex.names = 0.8,
  cex.axis = 0.8,
  cex.lab=0.8,
  cex.main=0.8)
```



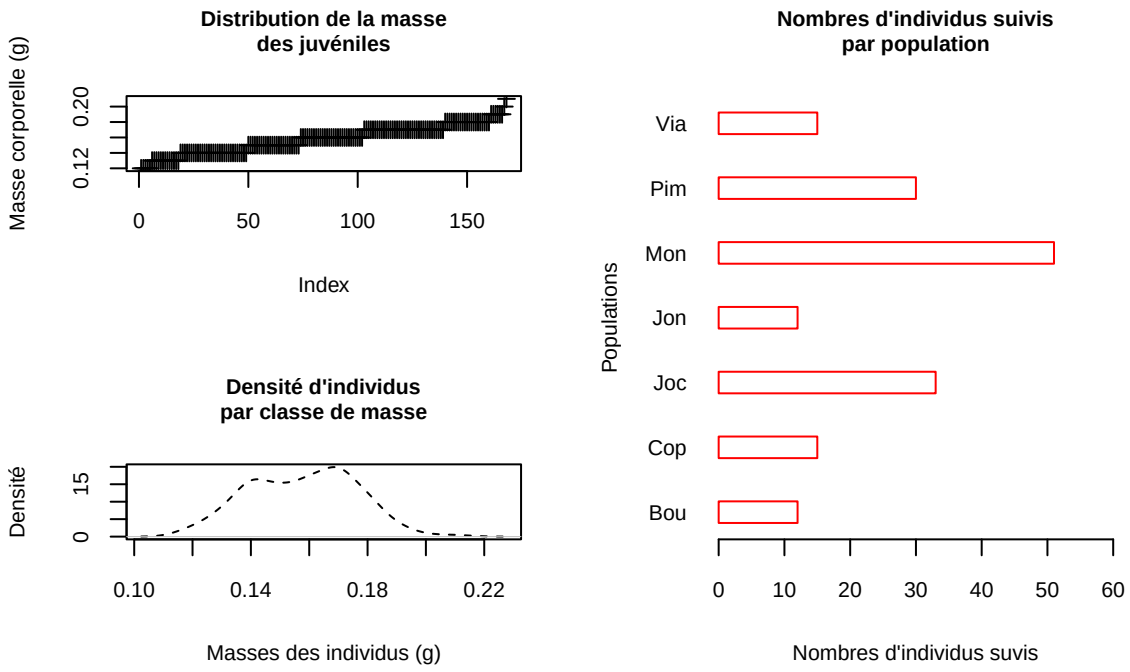

```r
par(mfrow=c(1, 1))



# on peut aussi gérer finement l'agencement des figures et les marges externes
# du panel de figure


par(mfrow = c(2, 2), oma = c(1, 1, 3, 1))
# ici on indique qu'on va réaliser un panel prenant 2 lignes et 2 colonnes, puis on change
# les marges extérieures du panel (argument "oma=c(bas, gauche, haut, droite)",
# en nombre de ligne)

layout(mat = matrix(c(1, 2, 3, 3), 2, 2, byrow = F))
# ici on indique l'arrangement de nos trois figures, les figures 1 et 2 seront dans la
# première colonne et la figure 3 prendra toute la deuxième colonne (la matrice étant
# définie par colonne, avec 2 lignes et 2 colonnes)
layout.show(n=3)
```




```r
# teste si notre arrangement graphique est bien le bon, en indiquant en entrée le nombre
# de figures qu'on souhaite voir apparaître dans notre panel

plot(
sort(data_lezard$M_IND),
main = "Distribution de la masse\ndes juvéniles",
ylab = "Masse corporelle (g)",
pch = 3,
cex.axis = 0.8,
cex.lab=0.8,
cex.main=0.8)

plot(
density(data_lezard$M_IND),
main = "Densité d'individus\npar classe de masse",
xlab = "Masses des individus (g)",
ylab = "Densité",
lty = "dashed",
cex.axis = 0.8,
cex.lab=0.8,
cex.main=0.8)


barplot(
  table(data_lezard$POP),
  main = "Nombres d'individus suivis\npar population",
```

```
  xlab = "Nombres d'individus suvis",
  ylab = "Populations",
  xlim = c(0,60),
  space = 2,
  horiz = T,
  border = "red",
  col = "white",
  names.arg = c("Bou", "Cop", "Joc", "Jon", "Mon", "Pim", "Via"),
  las = 1,
  cex.names = 0.8,
  cex.axis = 0.8,
  cex.lab=0.8,
  cex.main=0.8)


title("Distribution de la masse des juvéniles et population d'origine",
      line = 1,
      outer = T,
      col.main="darkred",
      cex.main = 1.5)
```

## Distribution de la masse des juvéniles et population d'origine




```
# ajout d'un titre dans la marge externe du panel (indiqué par "outer=T", puis une ligne au-dessus
# des limites du panel), on grossit aussi le titre (cex.main)

par(mfrow = c(1, 1), oma = c(0, 0, 0, 0))
```

```
# ré-initialisation des paramètres graphiques
```

**ASTUCES**

Il existe plusieurs aides-mémoires et tutoriels pour gérer les paramètres graphiques sur R. En voici quelques exemples: https://nicolascasajus.fr/files/pdfs/graphr.pdf, http://publish.illinois.edu/johnrgallagher/files/2015/10/BaseGraphicsCheatsheet.pdf, http://www.sthda.com/english/wiki/graphical-parameters.

**MISE EN APPLICATION**

Reprenez les graphiques réalisés au cours de la séance et apportez leur les améliorations graphiques que vous jugerez pertinentes.

## Bilan

Nous avons étudié comment décrire statistiquement et graphiquement des variables qualitatives et quantitatives sous R.
Les fonctions clés pour décrire une variable **qualitative** sont les suivantes:

- **table()**: table d'effectifs
- **barplot()**: graphe en barre

Les fonctions clés pour décrire une variable **quantitative** sont les suivantes:

- **summary()**: résumé statistique de la distribution avec moyenne (fonction **mean**), médiane (fonction **median**), quartile (fonction **quantile**) et min/max
- **sd()**: écart-type de la distribution
- **hist()**: histogramme de la distribution
- **boxplot()**: graphe "boîte à moustache"

Nous avons également étudié comment faire un **échantillonnage** au sein d'une variable donnée (fonction "**sample()**") et comment **simuler** une distribution théorique (fonction "**rnorm**" pour une loi normale). Ces différentes applications nous ont permis d'aborder les sorties graphiques sous R, et d'apprendre les mettre en forme et les personnaliser. Les principaux arguments (options) **graphiques** à retenir sont les suivants:

- **main / xlab / ylab**: noms du titre / axe des abscisses / axe des ordonnées
- **col**: couleurs des figurés
- **pch / lty**: types de figuré (points / lignes)
- **cex**: taille de police

**MISE EN APPLICATION**

Dans l'espace E-learn vous trouverez un autre jeu de données appelé "Interactions_dauphins_bateaux.txt". Cette table de données décrit le comportement de dauphins à proximité de bateaux (colonne boat.dist: "no"= pas de réponse, "approach"= s'approche du bateau, "avoidance"= s'éloigne du bateau, "response"= interagit avec le bateau) et peut être utile à des gestionnaires pour comprendre le potentiel dérangement généré par ces interactions. L'objectif est pour vous d'importer ce jeu de données dans R et d'utiliser les outils qui vous ont été présentés au cours de cette séance pour explorer ces données et vous les approprier. Voici quelques exemples ci-dessous d'objectifs que vous pouvez chercher à réaliser lors de votre exploration.

- Comment sont distribués les comportements et les réponses au cours de l'étude ? Comment en faire un ré-échantillonnage ?
- Quel est la distribution des tailles de groupes ? Comment peut-elle être décrite ? Y a-t-il une adéquation à une loi de Poisson ? (loi permettant de décrire des données de comptage, utilisez la moyenne de la distribution comme paramètre de la loi théorique)
- Comment faire une présentation graphique claire de l'ensemble des principales variables ?

Pensez bien à respectez les bonnes pratiques lorsque vous écrivez votre script pour explorer ce jeu de données, en gardant un espace de travail réduit au nécessaire et propre, en nommant correctement vos variables et objets R, en commentant bien votre code et en le structurant clairement.

**POUR ALLER PLUS LOIN**

Il existe une très grande communauté internationale travaillant sur le logiciel R, il est donc assez facile d'obtenir de l'aide en cas de blocage sur R. Une première source d'entraide sont les forums et tout particulièrement le forum "Stack Overflow" (en anglais), très actif sur le sujet. Il existe également certains documents résumant toutes les fonctions à connaître pour mener à bien des analyses statistiques comme le site [STHDA]http://www.sthda.com/french/ ou cet aide-mémoire pour les statistiques appliquées à la biologie: https://cran.r-project.org/doc/contrib/Herve-Aide-memoire-statistique.pdf (qui propose notamment un arbre de décision assez didactique pour orienter les choix d'analyses statistiques).