

# Chapitre 9

## Statistique descriptive

### Objectif

Ce chapitre décrit les différentes commandes à taper sous **R** pour structurer vos variables, tracer des résumés graphiques classiques de vos données et calculer des résumés numériques statistiques simples sur un jeu de données. Les données utilisées pour illustrer ce chapitre sont celles du jeu de données NUTRIAGE. Quelques exemples de fonctions permettant d'obtenir des graphiques de qualité esthétique supérieure pouvant servir dans des présentations ou des rapports sont également fournis.

#### SECTION 9.1

### Introduction

Nous allons fonder tous les exemples de ce chapitre sur le fichier de données `nutriage.xls` que vous pouvez importer dans **R** en suivant l'une des méthodes exposées dans le chapitre 2. Vous pourriez par exemple utiliser les instructions suivantes (pensez à installer le fichier <http://www.biostatisticien.eu/springer/Rtools.exe> si vous travaillez sous l'environnement Microsoft) :

```
> require("gdata") # Donne accès à la fonction read.xls()
> nutriage <- read.xls(
+ "http://www.biostatisticien.eu/springer/nutriage.xls")
```

```
> attach(nutriage)
> head(nutriage)
```

	<i>sexe</i>	<i>situation</i>	<i>the</i>	<i>cafe</i>	<i>taille</i>	<i>poids</i>	<i>age</i>	<i>viande</i>	<i>poisson</i>
1	2	1	0	0	151	58	72	4	3
2	2	1	1	1	162	60	68	5	2
3	2	1	0	4	162	75	78	3	1
4	2	1	0	0	154	45	91	0	4
5	2	1	2	1	154	50	65	5	3
6	2	1	2	0	159	66	82	4	2

	<i>fruit_crus</i>	<i>fruit_legume_cuits</i>	<i>chocol</i>	<i>matgras</i>
1	1		4	5
2	5		5	1
3	5		2	5
4	4		0	3
5	5		5	3
6	5		5	1

Les données de ce tableau consistent en la mesure de treize variables sur 226 individus.

Nous supposons, dans tout ce chapitre, que les données utilisées dans les différents exemples ont été structurées comme cela est indiqué dans la section suivante.

## SECTION 9.2

# Structuration des variables suivant leur type

Les treize variables du jeu de données NUTRIAGE peuvent être classées suivant leur type, en suivant l'algorithme décrit dans la figure suivante. Rappelons que le type d'une variable  $X$  se détermine par rapport à l'ensemble  $E_X$  des modalités **susceptibles d'être observées** et non pas sur la base des modalités réellement observées. En fait, le type d'une variable est défini (fait partie intégrante) du modèle mathématique que l'on choisit, parce qu'en l'état actuel des connaissances, c'est celui qui « représente » le mieux le phénomène que l'on veut étudier.

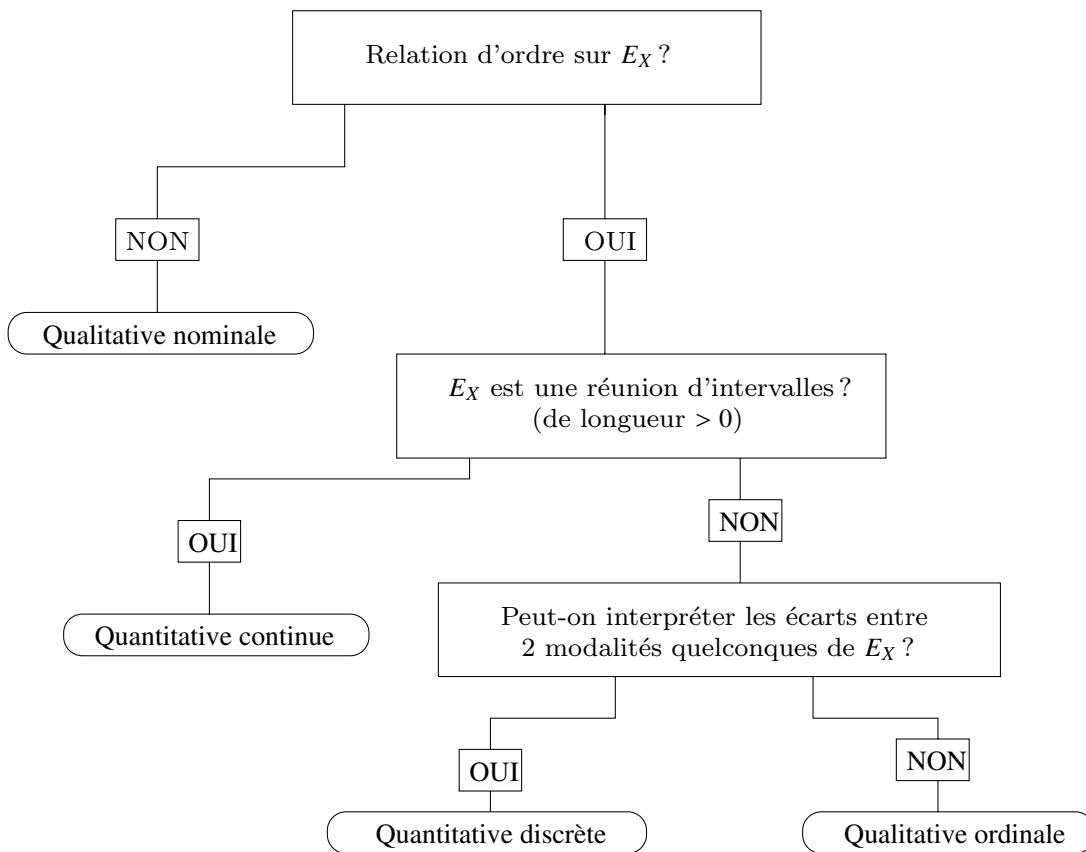


FIGURE 9.1 – Algorithme de détermination du type d'une variable.

En suivant cet algorithme, nous obtenons le tableau récapitulatif suivant :

Variables qualitatives	sexe, situation et matgras
Variables ordinales	viande, poisson, fruit_crus, fruit_legume_cuits et chocol
Variables quantitatives discrètes	the et cafe
Variables quantitatives continues	taille, poids et age

Nous allons commencer par imposer une structure **R** adaptée au type de chacune des variables précédentes.

### 9.2.1 Structurer les variables qualitatives

Pour les variables qualitatives, la structure est imposée au moyen de la fonction `as.factor()`. Il peut éventuellement être intéressant d'utiliser aussi la fonction `levels()` pour recoder les modalités d'une variable qualitative. Notez que l'association entre les codes et les *levels* se fait par ordre alphabétique (et pas par ordre de saisie).

Effectuons ces opérations sur les variables qualitatives de notre jeu de données.

```
> sexe <- as.factor(sexe)
> levels(sexe) <- c("Homme", "Femme")
> situation <- as.factor(situation)
> levels(situation) <- c("seul", "couple", "famille", "autre")
> matgras <- as.factor(matgras)
> levels(matgras) <- c("beurre", "margarine", "arachide",
+ "tournesol", "olive", "Isio4", "colza", "canard")
```

Notez que dans le cas où une variable serait codée en présence/absence, il est aussi possible d'utiliser une structuration R sous la forme d'un vecteur de logiques :

```
> fumeur <- c(1,0,0,1,0,1,0,1,0,0) # 10
                                     # fumeurs (=1)/non-fumeurs (=0) .
> fumeur
[1] 1 0 0 1 0 1 0 1 0 0
> fumeur <- as.logical(fumeur)
> fumeur
[1] TRUE FALSE FALSE TRUE FALSE TRUE FALSE TRUE FALSE
[10] FALSE
```

Si en outre vous souhaitez associer un nom à vos modalités, vous pouvez procéder ainsi :

```
> fumeur <- c(fume=fumeur)
> fumeur
  fume1  fume2  fume3  fume4  fume5  fume6  fume7  fume8  fume9
  TRUE  FALSE  FALSE  TRUE  FALSE  TRUE  FALSE  TRUE  FALSE
fume10
  FALSE
```

Nous déconseillons toutefois le codage en logique des variables du type présence/absence, car cela nuit à leur utilisation subséquente dans certaines fonctions R, notamment pour les représentations graphiques.

#### Expert

Si vous partez d'une variable déjà structurée en factor telle que celle-ci :

```
> fumeur <- as.factor(c("Fumeur", "NonFumeur", "NonFumeur",
+ "Fumeur", "NonFumeur", "Fumeur", "NonFumeur",
+ "Fumeur", "NonFumeur", "NonFumeur"))
> fumeur
[1] Fumeur NonFumeur NonFumeur Fumeur NonFumeur
[6] Fumeur NonFumeur Fumeur NonFumeur NonFumeur
Levels: Fumeur NonFumeur
```

vous pouvez procéder ainsi pour la coder en logiques :



```
> fumeur <- c(fume=as.logical(2-as.integer(fumeur)))
> fumeur
fume1  fume2  fume3  fume4  fume5  fume6  fume7  fume8  fume9
TRUE   FALSE  FALSE   TRUE   FALSE  TRUE   FALSE   TRUE   FALSE
fume10
FALSE
```

### 9.2.2 Structurer les variables ordinales

Pour les variables ordinales, la structure est imposée au moyen de la fonction `as.ordered()`. Il peut éventuellement être intéressant d'utiliser aussi la fonction `levels()` pour recoder les modalités d'une variable ordinale.

Effectuons ces opérations sur les variables ordinales de notre jeu de données :

```
> viande <- as.ordered(viande)
> poisson <- as.ordered(poisson)
> fruit_crus <- as.ordered(fruit_crus)
> fruit_legume_cuits <- as.ordered(fruit_legume_cuits)
> chocol <- as.ordered(chocol)
> niveaux <- c("jamais", "< 1/sem.", "1/sem.", "2-3/sem.",
+             "4-6/sem.", "1/jour")
> levels(chocol) <- levels(fruit_legume_cuits) <-
+             levels(fruit_crus) <- niveaux
> levels(poisson) <- levels(viande) <- niveaux
```

### 9.2.3 Structurer les variables quantitatives discrètes

Pour une variable discrète, la structure est imposée au moyen de la fonction `as.integer()`.

```
> the <- as.integer(the)
> cafe <- as.integer(cafe)
```

Cela n'est toutefois valable que si les données observées sont des entiers.

### 9.2.4 Structurer les variables quantitatives continues

Pour une variable continue, la structure est imposée au moyen de la fonction `as.double()`.

```
> taille <- as.double(taille)
> poids <- as.double(poids)
> age <- as.double(age)
```

## SECTION 9.3

## Tableaux de données

Pour un jeu de données particulier, les graphiques et les résumés numériques qu'il est possible de réaliser dépendent étroitement de la structure du tableau de données, et du type de la ou des variables en jeu. Dans ce chapitre, nous détaillons les possibilités offertes par **R** en ce qui a trait à l'organisation des données sous forme de tableaux.

## 9.3.1 Tableaux des données individuelles

Il s'agit du type d'organisation le plus courant. On dispose des mesures d'une ou de plusieurs variables pour chacun des  $N$  individus constitutifs d'une certaine population. Vous êtes priés de vous reporter au chapitre 2 pour savoir comment importer des données dans **R**. Les données sont en général organisées dans un *data.frame*.

## 9.3.2 Tableaux des effectifs ou des fréquences d'une variable

Il est souvent intéressant de représenter un tableau de données individuelles (ou tableau de données brutes) sous une forme plus condensée. Ainsi, le tableau des effectifs ou des fréquences (appelé aussi tri à plat) permet d'appréhender plus facilement la distribution d'une variable, notamment qualitative ou ordinaire. Il s'obtient au moyen de la fonction `table()`.

```
> tpe <- table(matgras)           # Tri à plat en effectifs.
> tpe
matgras
      beurre margarine  arachide tournesol      olive      Isio4
        15         27         48         68         40         23
      colza      canard
         1         4
> tpf <- tpe/length(matgras)      # Tri à plat en fréquences.
> tpf
matgras
      beurre  margarine  arachide  tournesol      olive
0.066371681 0.119469027 0.212389381 0.300884956 0.176991150
      Isio4      colza      canard
0.101769912 0.004424779 0.017699115
> levels(matgras)                 # Modalités.
[1] "beurre"      "margarine"   "arachide"    "tournesol"   "olive"
[6] "Isio4"       "colza"       "canard"
> nlevels(matgras)                # Nombre de modalités.
[1] 8
```

### 9.3.3 Tableaux de données regroupées en classes

Il est parfois intéressant de représenter un tableau de données individuelles (ou tableau de données brutes), récoltées sur une ou plusieurs variables quantitatives, sous une forme plus condensée. On utilise pour cela un tableau de données regroupées en classes, en notant les effectifs (ou les fréquences) de différentes classes préalablement déterminées.

Notez que vous pouvez utiliser la fonction `hist()` en spécifiant le vecteur des bornes des classes dans son paramètre `breaks` qui renvoie alors les effectifs de chacune de ces classes. La valeur par défaut du paramètre `breaks` (qui vaut "Sturges") effectue un calcul automatique des classes.

```
> res <- hist(taille, plot=FALSE)
> nn <- as.character(res$breaks)
> x <- as.table(res$counts)
> dimnames(x) <- list(paste(nn[-length(nn)], nn[-1], sep="-"))
> x
140-145 145-150 150-155 155-160 160-165 165-170 170-175
      1       7      37      50      46      31      27
175-180 180-185 185-190
      17       4       6
> # Ou bien en utilisant la fonction cut():
> table(cut(taille, res$breaks, include.lowest=TRUE))
[140,145] [145,150] [150,155] [155,160] [160,165] [165,170]
      1       7      37      50      46      31
[170,175] [175,180] [180,185] [185,190]
      27      17       4       6
```

### 9.3.4 Tableaux croisant deux variables

#### 9.3.4.1 Tableaux de contingence

Lorsque l'on dispose du tableau des données individuelles, on peut utiliser la fonction `table()` pour obtenir le tableau de contingence observé (encore appelé tri croisé en effectifs) du couple  $(X, Y)$ .

```
> matable <- table(sexe, situation)
> matable
      situation
sexe  seul couple famille autre
Homme   20    63      2     0
Femme  78    56      7     0
```

Si l'on veut rajouter les marges à ce tableau, on peut utiliser la fonction `addmargins()`.

```
> table.complète <- addmargins(matable, FUN=sum, quiet=TRUE)
> table.complète
      situation
sexe   seul couple famille autre sum
Homme   20    63      2      0  85
Femme  78    56      7      0 141
sum    98   119      9      0 226
```

## Astuce



- Pour changer l'intitulé « sum » des marges des tableaux précédents, vous pouvez définir la fonction `Total()` (ainsi : `Total <- sum`) puis remplacer `sum` par `Total` dans l'appel de `addmargins()` ci-dessus.
- Lorsque le tableau de contingence est déjà fourni dans un fichier, nous avons vu dans le chapitre 2 comment importer ce type de fichier au moyen de la fonction `read.ftable()` et l'afficher au moyen de la fonction `ftable()`.

### 9.3.4.2 Distribution conjointe

Le tableau de la distribution conjointe (encore appelé tri croisé en fréquences relatives) du couple  $(X, Y)$  s'obtient à partir du tableau de contingence `matable` précédent.

```
> tableaufreq <- matable/sum(matable)
> tableaufreq
      situation
sexe   seul   couple   famille   autre
Homme 0.088495575 0.278761062 0.008849558 0.000000000
Femme 0.345132743 0.247787611 0.030973451 0.000000000
```

Si l'on veut aussi obtenir les marges, on peut utiliser l'une des instructions suivantes :

```
> Total <- sum
> addmargins(tableaufreq, FUN=Total, quiet=TRUE)
      situation
sexe   seul   couple   famille   autre
Homme 0.088495575 0.278761062 0.008849558 0.000000000
Femme 0.345132743 0.247787611 0.030973451 0.000000000
Total 0.433628319 0.526548673 0.039823009 0.000000000
      situation
sexe   Total
Homme 0.376106195
Femme 0.623893805
Total 1.000000000
```



```
> tableaufreqtotal <- table.complète/table.complète[
+                               length(table.complète)]
> tableaufreqtotal
      situation
sexe      seul      couple      famille      autre
Homme 0.088495575 0.278761062 0.008849558 0.000000000
Femme 0.345132743 0.247787611 0.030973451 0.000000000
sum   0.433628319 0.526548673 0.039823009 0.000000000
      situation
sexe      sum
Homme 0.376106195
Femme 0.623893805
sum   1.000000000
```

### 9.3.4.3 Distributions marginales

L'obtention des marges d'une table de distribution `tableaufreq` (ou d'une table de contingence) s'obtient au moyen de la fonction `margin.table()`.

```
> margin.table(tableaufreq,1) # Marge de droite.
sexe
      Homme      Femme
0.3761062 0.6238938
> margin.table(tableaufreq,2) # Marge du bas.
situation
      seul      couple      famille      autre
0.43362832 0.52654867 0.03982301 0.00000000
```

#### Attention

Lorsque la table de distribution (ou de contingence) est complète, c'est-à-dire qu'elle contient déjà les marges (comme pour `tableaufreqtotal`), il ne faut pas utiliser la fonction `margin.table()` pour les extraire, mais plutôt procéder ainsi :

```
> tableaufreqtotal[,ncol(tableaufreqtotal)] # Marge de droite.
      Homme      Femme      sum
0.3761062 0.6238938 1.0000000
> tableaufreqtotal[nrow(tableaufreqtotal),] # Marge du bas.
      seul      couple      famille      autre      sum
0.43362832 0.52654867 0.03982301 0.00000000 1.00000000
```



### 9.3.4.4 Distributions conditionnelles

Les tableaux des distributions conditionnelles s'obtiennent au moyen de la fonction `prop.table()`.

- Distributions conditionnelles de `situation` sachant les valeurs de `sexe` :

```
> prop.table(matable,1)
      situation
sexe      seul      couple      famille      autre
Homme 0.23529412 0.74117647 0.02352941 0.00000000
Femme 0.55319149 0.39716312 0.04964539 0.00000000
> addmargins(prop.table(matable,1),margin=2,FUN=sum) # Ajout du
                                                    # Total.

      situation
sexe      seul      couple      famille      autre      sum
Homme 0.23529412 0.74117647 0.02352941 0.00000000 1.00000000
Femme 0.55319149 0.39716312 0.04964539 0.00000000 1.00000000
```

- Distributions conditionnelles de `sexe` sachant les valeurs de `situation` :

```
> prop.table(matable,2)
      situation
sexe      seul      couple      famille      autre
Homme 0.2040816 0.5294118 0.2222222
Femme 0.7959184 0.4705882 0.7777778
> addmargins(prop.table(matable,2),margin=1,FUN=sum) # Ajout du
                                                    # Total.

      situation
sexe      seul      couple      famille      autre
Homme 0.2040816 0.5294118 0.2222222
Femme 0.7959184 0.4705882 0.7777778
sum    1.0000000 1.0000000 1.0000000
```

## SECTION 9.4

## Résumés numériques

Nous présentons tous les résumés numériques sur le vecteur  $\mathbf{x} = (x_1, \dots, x_N)^T$  pour plus de simplicité. Ce vecteur est l'ensemble des  $N$  valeurs de la variable  $X$  mesurées sur une population d'effectif  $N$  (cas standard de la statistique descriptive). Lorsque le vecteur  $\mathbf{x}$  sera plutôt considéré comme un échantillon extrait, nous noterons sa longueur  $n$ . Les exemples d'application seront principalement fondés sur la série de données du vecteur `taille`.

### Attention

Les résumés numériques ne peuvent être calculés en présence de données manquantes (NA). Si cela est nécessaire, il est possible d'utiliser la fonction `na.omit()` pour les retirer lors du calcul.

```
> x <- na.omit(taille) # Inutile ici car taille n'a pas de NA.
```



### 9.4.1 Résumés de position d'une distribution

#### 9.4.1.1 Le (ou les) mode(s)

Les modes sont les valeurs de la variable  $X$  qui apparaissent le plus fréquemment. Ils peuvent se calculer pour une variable de n'importe quel type, bien que, pour une variable quantitative continue, on présente plutôt la classe modale. Notez que le mode peut être unique, auquel cas on parle de distribution unimodale, par opposition à des variables multimodales.

```
> tabthe <- table(the)
> names(which.max(tabthe))           # Obtention d'un mode
                                     # unique.

[1] "0"
> names(tabthe)[max(tabthe)==tabthe] # Obtention de tous les
                                     # modes.

[1] "0"
```

Ici, la variable `the` (nombre de tasses de thé par jour) est unimodale.

#### Astuce

Dans le cas d'une variable quantitative, vous pouvez utiliser la fonction `as.numeric()` sur les résultats ci-dessus pour récupérer des valeurs numériques.



#### 9.4.1.2 La médiane

La médiane d'une série statistique est la valeur  $m_e$  de la variable  $X$  qui partage cette série statistique en deux parties (inférieure et supérieure à  $m_e$ ) de même effectif, les valeurs du caractère étant rangées dans l'ordre croissant. Il s'agit d'un critère de position qui ne se calcule évidemment pas pour des variables purement qualitatives. Pour la calculer, on distingue deux cas :


- l'effectif total  $N$  de la série est impair. Dans ce cas, la médiane est la valeur située à la position  $\frac{N+1}{2}$  ;
- l'effectif total  $N$  de la série est pair. Dans ce cas, n'importe quelle valeur comprise entre les valeurs aux positions  $\frac{N}{2}$  et  $\frac{N}{2} + 1$  peut être considérée comme une médiane de la série. En pratique, la médiane est généralement la moyenne de ces deux valeurs (cela exclut donc les caractères ordinaux de ce dernier calcul).

La fonction **R** permettant de calculer une médiane **uniquement pour des données numériques** est `median()`.

```
> median(x)
[1] 163
```

## Astuce

Nous proposons le code suivant qui permettra de calculer la médiane pour des données **individuelles** ordinales ou numériques :



```
> ma.mediane <- function(x) {
+   if (is.numeric(x)) return(median(x))
+   if (is.ordered(x)) {
+     N <- length(x)
+     if (N%%2) return(sort(x)[(N+1)/2]) else {
+       inf <- sort(x)[N/2]
+       sup <- sort(x)[N/2+1]
+       if (inf==sup) return(inf) else return(c(inf,sup))
+     }
+   }
+   stop("Calcul de médiane impossible pour ce type")
+ }
> ma.mediane(poisson)
[1] 2-3/sem.
6 Levels: jamais < 1/sem. < 1/sem. < ... < 1/jour
```

Supposons maintenant que l'on dispose non pas des données individuelles, mais uniquement du tableau des fréquences observées  $f_1, \dots, f_k, \dots, f_K$  dans les  $K$  classes  $[e_0, e_1], \dots, [e_{k-1}, e_k], \dots, [e_{K-1}, e_K]$  d'une variable **quantitative**. Il faut alors procéder par interpolation linéaire entre les deux valeurs dont les fréquences cumulées encadrent 50 %. Notons  $e_k$  et  $e_{k+1}$  les deux valeurs consécutives des bornes de classe telles que  $PFC_X(e_k) < 0.5 \leq PFC_X(e_{k+1})$ , où  $PFC_X(e_j) = f_1 + f_2 + \dots + f_j$ ,  $j = 1, \dots, K$  ( $PFC_X(x)$  est la valeur du polygone des fréquences cumulées au point  $x$ ). Alors on a :

$$m_e = e_k + (e_{k+1} - e_k) \frac{0.5 - PFC_X(e_k)}{PFC_X(e_{k+1}) - PFC_X(e_k)}.$$

Nous pouvons proposer le programme suivant permettant de calculer la médiane dans ce cas :

```
> mediane.sur.freq <- function(x) {
+   # x est le tableau des fréquences.
+   tmp <- suppressWarnings(as.double(names(x)))
+   if (!(is.null(tmp) | all(is.na(tmp)))) {
+     tab.freq.cum <- cumsum(x)
+     index <- order(tab.freq.cum < 0.5)[1]
+     fc1 <- tab.freq.cum[index]
+     fc2 <- tab.freq.cum[index-1]
+     x1 <- as.numeric(names(fc1))
+     x2 <- as.numeric(names(fc2))
+     mex <- as.numeric(x1 + (x2-x1)*(0.5-fc1)/(fc2-fc1))
+   } else {mex <- NA}
```

```
+ return(mex)
+ }
```

Voici le calcul de la médiane pour les données du vecteur **x** regroupées en classes au moyen de la fonction **hist()** :

```
> res <- hist(x, plot=FALSE, breaks=c(130, 150, 160, 170, 180, 190))
> tab.x <- table(rep(res$breaks[-1], res$counts))
> tab.x
150 160 170 180 190
  8  87  77  44  10
> mediane.sur.freq(tab.x/sum(tab.x))
[1] 162.3377
```

### 9.4.1.3 La moyenne

Elle se calcule uniquement pour des variables quantitatives.

```
> mean(x)      #  $\mu_X = \frac{1}{N} \sum_{i=1}^n x_i$ 
[1] 163.9602
```

### 9.4.1.4 Les fractiles

Le fractile d'ordre  $p$  ( $0 < p < 1$ ) est la valeur  $q_p$  de la variable  $X$  qui coupe l'échantillon en deux portions, l'une ayant un nombre d'éléments égal à  $p$  % du nombre total d'éléments dans **x** (ce sont les éléments inférieurs à  $q_p$ ), l'autre à  $(1 - p)$  % (ce sont les éléments supérieurs à  $q_p$ ). Il ne se calcule pas pour des variables purement qualitatives.

Fractiles d'ordre 10 % et 90 % :

```
> quantile(x, probs=c(0.1, 0.9))
10% 90%
153 176
```

Quartiles  $q_{1/4}, q_{1/2}, q_{3/4}$  (aussi notés  $q_1, q_2 = m_e, q_3$ ) :

```
> quantile(x, probs=c(0.25, 0.5, 0.75))
25% 50% 75%
157 163 170
```

Déciles :

```
> quantile(x, probs=1:10/10)
10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
153.0 156.0 158.5 160.0 163.0 165.0 168.0 172.0 176.0 188.0
```

## Attention



La fonction `quantile()` contient un argument `type` pouvant prendre une valeur entière (entre 1 et 9) sélectionnant l'un des neuf algorithmes qui sont détaillés dans le fichier d'aide de cette fonction.

## Astuce



La fonction `summary()` appliquée à un vecteur de données quantitatives permet de calculer le minimum, le maximum, la moyenne et les trois quartiles.

## 9.4.2 Résumés de dispersion d'une distribution

Ces résumés peuvent être calculés uniquement pour des variables quantitatives. Nous les présentons dans le tableau ci-dessous après avoir défini les trois fonctions R suivantes :

```
> # Variance  $\sigma^2$  de la population  $\sigma^2(x) = \frac{1}{N} \sum_{i=1}^N (x_i - \mu_X)^2$ .
> var.pop <- function(x) var(x) * (length(x)-1) / length(x)
> # Écart type  $\sigma$  de la population  $\sigma(x) = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - \mu_X)^2}$ .
> sd.pop <- function(x) sqrt(var.pop(x))
> # Coefficient de variation:
> #  $c_v = \frac{\sigma}{\mu_X}$ .
> co.var <- function(x) sd.pop(x) / mean(x)
```

Nom	Formule mathématique	Instruction R	Résultat
Étendue	$\max_{1 \leq i \leq N} (x_i) - \min_{1 \leq i \leq N} (x_i)$	<code>max(x)-min(x)</code> <code>diff(range(x))</code>	48.0
Intervalle inter-quartiles	$q_{3/4} - q_{1/4}$	<code>IQR(x)</code>	13.0
Variance	$\sigma_{pop}^2(x) = \frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^2$	<code>var.pop(x)</code>	80.702
Écart type	$\sigma_{pop}(x) = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^2}$	<code>sd.pop(x)</code>	8.983
Coefficient de variation	$c_v = \frac{\sigma_{pop}}{\mu_X}$	<code>co.var(x)</code>	0.055
Déviation absolue par rapport à la médiane	$\frac{1}{N} \sum_{i=1}^N  x_i - me_X $	<code>mad(x)</code>	10.378
Écart moyen	$\frac{1}{N} \sum_{i=1}^N  x_i - \bar{x} $	<code>mean(abs(x-mean(x)))</code>	7.312

## Remarque



Une estimation sans biais  $\hat{\sigma}^2$  de la variance  $\sigma^2$  de la population, fondée sur un échantillon de taille  $n$ , est calculée au moyen de la fonction `var()`. L'écart type  $\hat{\sigma}$  correspondant est calculé au moyen de la fonction `sd()`.



```

#  $O_{ij}$ .
> tab.ind <- chisq.test(sexematgras)$expected # Les  $E_{ij}$ .
> round(tab.ind)
      matgras
sexe  beurre margarine arachide tournesol olive Isio4 colza
Homme    6      10      18      26      15      9      0
Femme    9      17      30      42      25      14      1
      matgras
sexe  canard
Homme    2
Femme    2
> # (sexematgras-tab.ind)^2/tab.ind:
#  $\frac{(O_{ij}-E_{ij})^2}{E_{ij}}$ .
> tab.contr <- chisq.test(sexematgras)$residuals^2
> tab.contr
      matgras
sexe  beurre  margarine  arachide  tournesol
Homme 3.367083116 0.002361810 0.233489502 0.818473834
Femme 2.029801879 0.001423786 0.140756083 0.493406212
      matgras
sexe  olive  Isio4  colza  canard
Homme 1.632483082 1.540468053 0.376106195 1.486777720
Femme 0.984121007 0.928650954 0.226730685 0.896284441
> khi2 <- chisq.test(sexematgras)$statistic # sum(tab.contr)
> khi2
#  $\chi^2 = \sum_{i,j} \frac{(O_{ij}-E_{ij})^2}{E_{ij}}$ .
X-squared
15.15842

```

## Astuce

Une autre manière de calculer la statistique du  $\chi^2$  repose sur l'utilisation de la fonction `summary()`.

```

> khi2 <- summary(table(sexe,matgras))$statistic
> khi2
[1] 15.15842

```

9.5.1.2  $\Phi^2$ ,  $V$  de Cramér et coefficient de contingence de Pearson

Tous les indicateurs du tableau suivant se calculent à partir du coefficient du  $\chi^2$ .

```

> N <- sum(sexematgras)
> p <- nrow(sexematgras)
> q <- ncol(sexematgras)

```



Indicateur	Formule	Instruction R	Résultat
$\Phi^2$	$\Phi^2 = \frac{\chi^2}{N}$	<code>Phi2 &lt;- khi2/N</code>	0.067
$V^2$ de Cramér	$V^2 = \frac{\Phi^2}{\min(p-1, q-1)}$	<code>Phi2/(min(p,q)-1)</code>	0.067
Coefficient de contingence de Pearson	$C = \sqrt{\frac{\chi^2}{(N+\chi^2)}}$ $= \sqrt{\frac{\Phi^2}{(1+\Phi^2)}}$	<code>sqrt(khi2/(N+khi2))</code>	0.251

**Astuce**

La fonction `cramer.v()` du package `rgrs` permet aussi de calculer le  $V$  de Cramér.

```
> require("rgrs")
> cramer.v(sexematgras)^2
[1] 0.06707265
```



## 9.5.2 Mesures de liaison entre des variables ordinales (ou des rangs)

### 9.5.2.1 Le $\tau$ et le $\tau_b$ de Kendall

Ce coefficient est fondé sur la notion de concordance entre individus. Pour deux individus  $i$  et  $j$  et pour les variables ordinales  $X$  et  $Y$  ayant  $p$  et  $q$  modalités respectivement, on dit que les paires  $(x_i, y_i)$  et  $(x_j, y_j)$  sont concordantes si  $\text{sign}(x_j - x_i) = \text{sign}(y_j - y_i)$  et discordantes si  $\text{sign}(x_j - x_i) = -\text{sign}(y_j - y_i)$ . Si  $x_i = x_j$  ou  $y_i = y_j$  (ou les deux), la paire correspondante n'est ni concordante ni discordante, et nous disons qu'il s'agit d'*ex æquo*. S'il y a  $n_c$  paires concordantes,  $n_d$  paires discordantes et  $n_e$  *ex æquo*, alors  $n_c + n_d + n_e = \frac{1}{2}N(N-1)$ . On calcule alors le  $\tau_b$  de Kendall par la formule :

$$\tau_b = \frac{2(n_c - n_d)}{\sqrt{(N^2 - \sum_{k=1}^p n_{k\bullet}^2)(N^2 - \sum_{l=1}^q n_{\bullet l}^2)}}$$

avec  $2(n_c - n_d) = \sum_{k=1}^p \sum_{l=1}^q \text{sign}(x_k - x_l) \text{sign}(y_k - y_l)$ ,  $n_{k\bullet} = \sum_{l=1}^q n_{kl}$  et  $n_{\bullet l} = \sum_{k=1}^p n_{kl}$ , où  $n_{kl}$  est le nombre d'individus ayant à la fois la modalité  $k$  pour  $X$  et  $l$  pour  $Y$ .

Lorsqu'il n'y a pas d'*ex æquo*, cette formule se simplifie en ce que l'on appelle le  $\tau$  de Kendall :


$$\tau = \frac{2(n_c - n_d)}{N(N-1)}.$$

Ces deux quantités se calculent en **R** au moyen de la fonction `cor()`.

```
> cor(as.numeric(viande), as.numeric(poisson), method="kendall")
[1] -0.1583088
```

## Astuce

Notez qu'il est possible de programmer soi-même ces coefficients en traduisant simplement la formule de  $\tau_b$  en instructions R :



```
> Kendall.taub <- function(x,y) {
+   a1 <- sign(outer(as.numeric(x),as.numeric(x),"-"))
+   a2 <- sign(outer(as.numeric(y),as.numeric(y),"-"))
+   num <- sum(a1*a2)
+   N <- length(x)
+   b1 <- sum(margin.table(table(x,y),1)^2)
+   b2 <- sum(margin.table(table(x,y),2)^2)
+   denom <- sqrt((N^2-b1)*(N^2-b2))
+   taub <- num / denom
+   return(taub)
+ }
> Kendall.taub(viande,poisson)
[1] -0.1583088
```

### 9.5.2.2 Coefficient $\rho$ de corrélation des rangs de Spearman

Il faut commencer par calculer les rangs (fonction `rank()`) des individus pour la variable  $X$  (notés  $x_i$ ) et pour la variable  $Y$  (notés  $y_i$ ). En cas d'*ex æquo*, il faut assigner comme même rang aux valeurs égales la moyenne de leurs positions dans l'ordre croissant des valeurs (ce que fait la fonction `rank()` par défaut).

Lorsqu'il n'y a pas d'*ex æquo*, le coefficient de corrélation des rangs de Spearman est donné par la formule suivante :

$$\rho = 1 - \frac{6 \sum_{i=1}^N d_i^2}{N(N^2 - 1)} \quad \text{avec } d_i = x_i - y_i.$$

En présence de valeurs *ex æquo*, il faut utiliser le coefficient de corrélation classique de Pearson **entre les rangs** :

$$\rho = \frac{N(\sum_{i=1}^N x_i y_i) - (\sum_{i=1}^N x_i)(\sum_{i=1}^N y_i)}{\sqrt{N(\sum_{i=1}^N x_i^2) - (\sum_{i=1}^N x_i)^2} \sqrt{N(\sum_{i=1}^N y_i^2) - (\sum_{i=1}^N y_i)^2}}.$$

Pour l'obtenir sous R, on peut donc utiliser les fonctions `rank()` et `cor()`, ou bien directement la fonction `cor()` avec la valeur d'entrée "spearman" de son paramètre `method`.

```
> cor(rank(matgras), rank(situation))
[1] 0.008787643
> cor(as.numeric(matgras), as.numeric(situation), method="spearman")
[1] 0.008787643
```

### 9.5.3 Mesures de liaison entre deux variables quantitatives

#### 9.5.3.1 Covariance et coefficient de corrélation de Pearson

L'indicateur de liaison approprié dans le cas de deux variables quantitatives est la corrélation. Il est défini comme le rapport entre la covariance des deux variables et le produit de leurs écarts types respectifs. Il se calcule au moyen de la fonction `cor()`.

```
> cor(taille,poids)
[1] 0.6306576
```

La covariance se calcule au moyen de la fonction `cov()`.

```
> cov(taille,poids)
[1] 68.32596
```

#### Astuce

Ce coefficient peut aussi se calculer avec la fonction `cor.test()`.

```
> cor.test(taille,poids)$estimate
      cor
0.6306576
```



#### Attention

Cet indicateur est pertinent si la relation unissant les deux variables est de type linéaire. Dans le cas contraire, il faut utiliser une mesure de dépendance plus générale, comme par exemple la *distance correlation* disponible dans le *package* `IndependenceTests`.



### 9.5.4 Mesures de liaison entre une variable quantitative et une variable qualitative

#### 9.5.4.1 Le rapport de corrélation $\eta_{Y|X}^2$

Le rapport de corrélation  $\eta_{Y|X}^2$  indique dans quelle mesure les variations d'une variable quantitative  $Y$  sont expliquées par les modalités d'une variable qualitative  $X$  à  $p$  modalités. En effet, on peut considérer que la variable  $X$  définit des groupes dans la population. Le rapport de corrélation est alors défini comme le rapport entre la variance inter-groupes et la variance intra-groupe. Il se calcule au moyen de la formule suivante :

$$\eta_{Y|X}^2 = \frac{\sum_{k=1}^p n_k (\bar{y}_k - \bar{y})^2}{\sum_{i=1}^N (y_i - \bar{y})^2}$$

dans laquelle  $n_k$  désigne le nombre d'observations  $y_i$  correspondant à la  $k$ -ième modalité de  $X$ .

Voici le programme R permettant de le calculer :

```
> eta2 <- function(x, gpe) {
+   moyennes <- tapply(x, gpe, mean)
+   effectifs <- tapply(x, gpe, length)
+   varinter <- (sum(effectifs * (moyennes - mean(x))^2))
+   vartot <- (var(x) * (length(x) - 1))
+   res <- varinter/vartot
+   return(res)
+ }
```

Calculons-le par exemple pour les variables `poids` et `sexe`.

```
> eta2(poids, sexe)
[1] 0.3325501
```

#### Astuce



Vous pouvez aussi utiliser l'instruction R suivante pour calculer  $\eta^2_{Y|X}$  :

```
> summary(lm(poids~sexe))$r.squared
[1] 0.3325501
```

## SECTION 9.6

# Représentations graphiques

Rappelons qu'il convient toujours de choisir adéquatement le mode de représentation graphique d'une variable adapté à son type. En effet, **le type d'une variable est souvent traduit par des caractéristiques particulières d'un graphique** donné. Par exemple, l'ajout d'une pointe de flèche au bout d'un axe indique un ordre sur les modalités correspondantes. Il conviendra donc d'indiquer le sens de l'axe des modalités pour toutes les variables sauf pour celles qui sont du type qualitatif. Nous avons par conséquent défini, et intégré au *package* associé à ce livre, la fonction `fleches()` qui nous permettra d'ajouter, lorsque cela sera nécessaire, une flèche sur les axes d'un graphique. Par ailleurs, nous avons choisi de présenter côte à côte, pour quelques graphiques, une version basique et une version esthétiquement plus élaborée de ceux-ci. Bien que nous ne conseillions pas nécessairement l'utilisation de ces versions élaborées dans une phase d'exploration des données, ces dernières ont l'avantage d'illustrer les grandes possibilités offertes par R pour modifier à sa guise un graphique.

## 9.6.1 Graphiques pour les variables qualitatives

### 9.6.1.1 Diagramme en croix

Le diagramme en croix affiche pour chaque observation une petite barre horizontale dans la colonne de la modalité correspondante. Il n'est pas intégré au logiciel **R**, mais nous pouvons le programmer au moyen de la fonction `plot()` et de son paramètre `pch`. La fonction obtenue, que nous avons nommée `diagcroix()`, est intégrée au *package* associé au livre.

```
> diagcroix(situation, col=c("orange", "darkgreen", "black", "tan"))
```

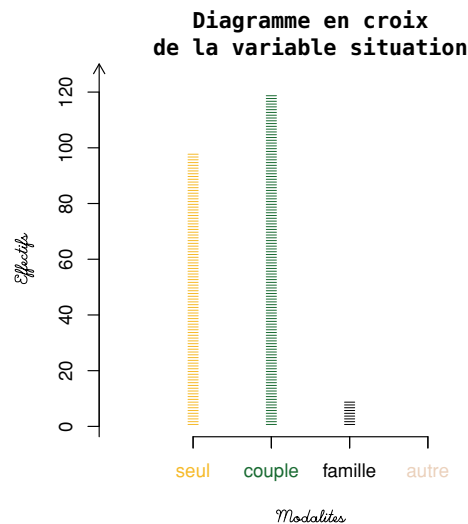


FIGURE 9.2 – Diagramme en croix pour une variable qualitative.

Notons au passage l'existence de la fonction `dotchart()` dont l'utilisation, couplée avec la fonction `table()`, offre un type d'affichage légèrement similaire, si ce n'est qu'il représente plutôt le tableau des effectifs de la variable.

```
> dotchart(table(situation), col=c("orangered", "darkgreen",
+ "turquoise", "tan"), pch=15, main=paste("Diagramme en points des
+ effectifs", "de la variable situation", sep="\n"), lcolor="red")
```

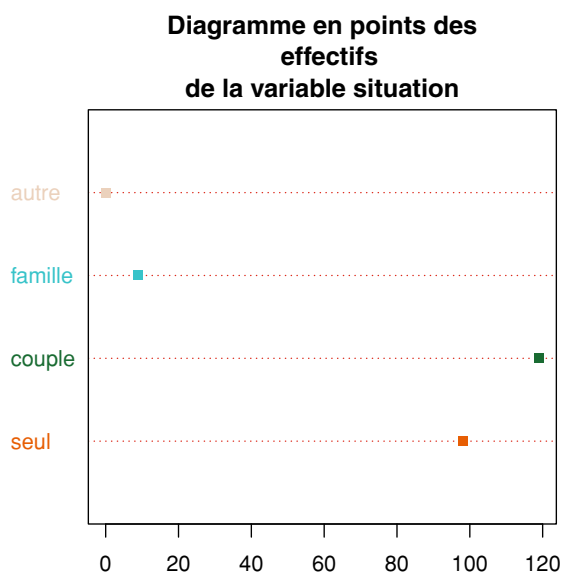


FIGURE 9.3 – Diagramme en points pour une variable qualitative.

### 9.6.1.2 Diagramme en tuyaux d'orgue

Son obtention se fait au moyen de la fonction `barplot()`. Afin d'améliorer un peu la qualité esthétique du graphique, nous proposons la fonction `tuyauxorgue()`, présente dans le *package* associé à ce livre.

```
> col <- c("gray", "orangered", "lightgoldenrodyellow", "red")
> barplot(table(situation), col=col)
```

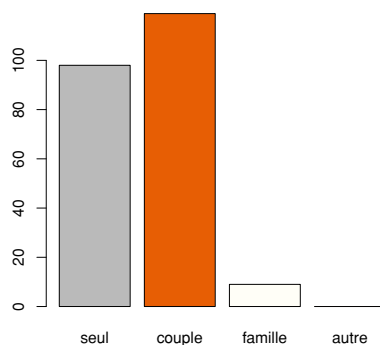
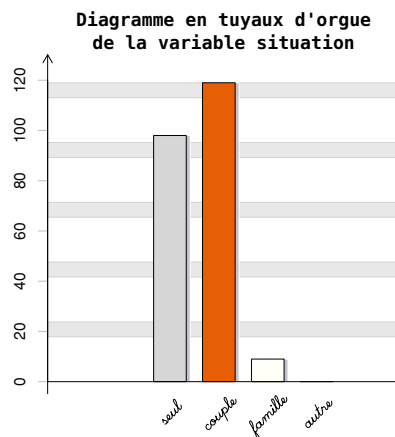


FIGURE 9.4 – Diagramme en tuyaux d'orgue pour une variable qualitative.

```
> tuyauxorgue(situation,col)
```



### 9.6.1.3 Diagramme de Pareto

Son obtention se fait également au moyen de la fonction `barplot()`, puisqu'il s'agit d'un diagramme en tuyaux d'orgue dont les tuyaux sont représentés par hauteur décroissante.

```
> col <- c("yellow","yellow2","sandybrown","orange",
+         "darkolivegreen","green","olivedrab2","green4")
```

```
> barplot(sort(table(matgras),TRUE),col=col)
```

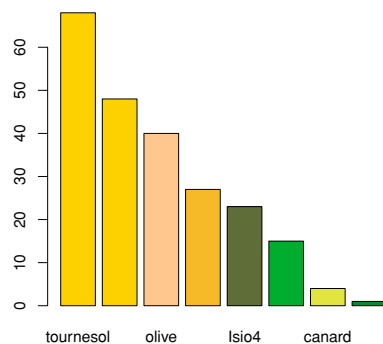
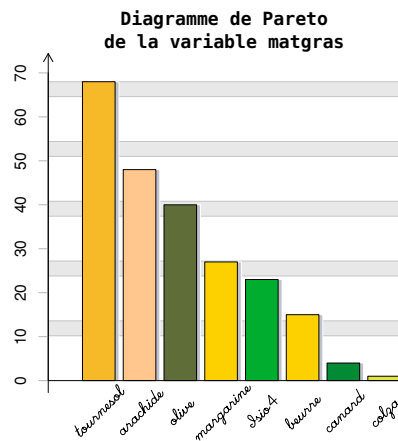


FIGURE 9.5 – Diagramme de Pareto pour une variable qualitative.

```
> tuyauxorgue(matgras,col,pareto=TRUE)
```



#### 9.6.1.4 Diagramme empilé

Il s'obtient au moyen de la fonction `barplot()` en fournissant un objet du type `matrix` comme premier paramètre effectif.

```
> nbh <- table(sexe)[1]
> nbf <- table(sexe)[2]
> tabfreq.matgras.hommes <- table(matgras[sexe=="Homme"])/nbh
> tabfreq.matgras.femmes <- table(matgras[sexe=="Femme"])/nbf
> barplot(cbind(tabfreq.matgras.hommes,tabfreq.matgras.femmes),
+         main="Diagramme empilé de la variable matgras",col=
+         c("yellow","yellow2","sandybrown","orange",
+         "darkolivegreen","green","olivedrab2","green4"),xlim=
+         c(0,1),width=0.15,space=1,names.arg=
+         c("Hommes","Femmes"),legend=TRUE,density=40)
> fleches(x=FALSE,y=TRUE)
```

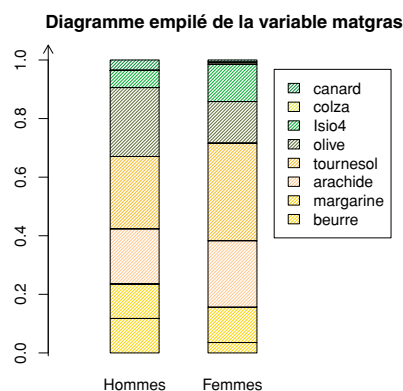


FIGURE 9.6 – Diagramme empilé pour une variable qualitative.

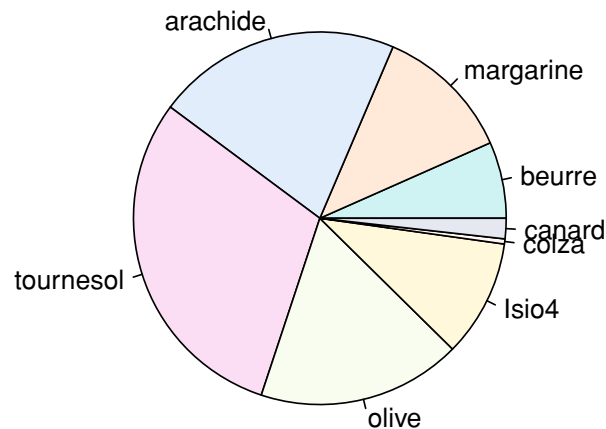


### 9.6.1.5 Diagramme circulaire

Il s'obtient au moyen de la fonction `pie()`. Afin d'améliorer un peu la qualité esthétique du graphique, nous proposons d'utiliser la fonction `camembert()`, présente dans le *package* associé à ce livre.

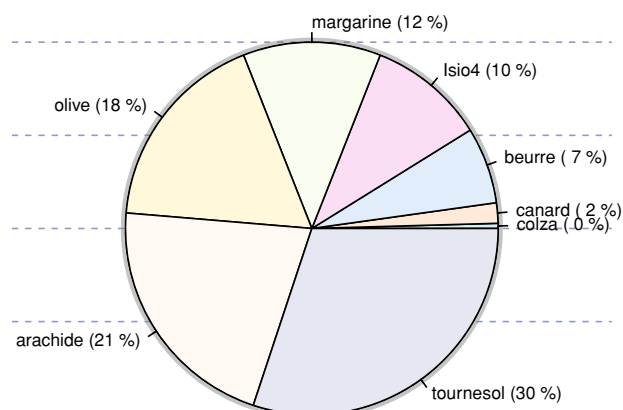
```
> require("RColorBrewer")
> col <- brewer.pal(8, "Pastel12")

> pie(table(matgras), col=col)
```



```
> camembert(matgras, col)
```

**Diagramme circulaire  
de la variable matgras**



## 9.6.2 Graphiques pour les variables ordinales

### 9.6.2.1 Diagramme en tuyaux d'orgue avec courbe des fréquences cumulées

Il s'obtient au moyen des fonctions `barplot()` et `points()`. Ici encore, l'utilisation de la fonction `tuyauxorgue()` présente dans le *package* associé à ce livre permet une représentation différente (avec notamment la courbe des fréquences cumulées apparaissant en relief avec une ombre projetée).

```
> require("RColorBrewer")
> col <- brewer.pal(6, "Blues")
> tx <- table(poisson)
> tx <- tx/sum(tx)
> r <- barplot(tx, ylim=c(0, 1), col=col)
> points(r, cumsum(tx), type="l")
```

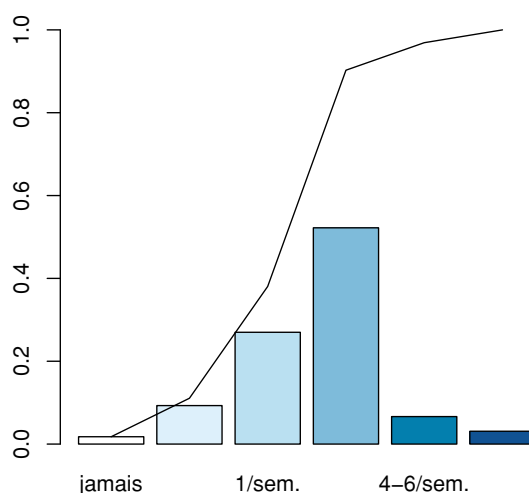


FIGURE 9.7 – Tuyaux d'orgue avec courbe des fréquences cumulées pour une variable ordinaire.

## 9.6.3 Graphiques pour les variables quantitatives discrètes

### 9.6.3.1 Diagramme en croix

Il s'obtient au moyen de la fonction `diagcroix()`, présente dans le *package* associé à ce livre.

## Renvoi

Voir la partie sur les variables qualitatives, page 387.



### 9.6.3.2 Diagramme en bâtons

Il s'obtient au moyen de la fonction `plot()` appliquée à une table de contingence.

```
> plot(tabthe/length(unique(the)),ylab="",
+       col="darkolivegreen",lwd=5,
+       main="Diagramme en bâtons de la variable thé")
> fleches()
```

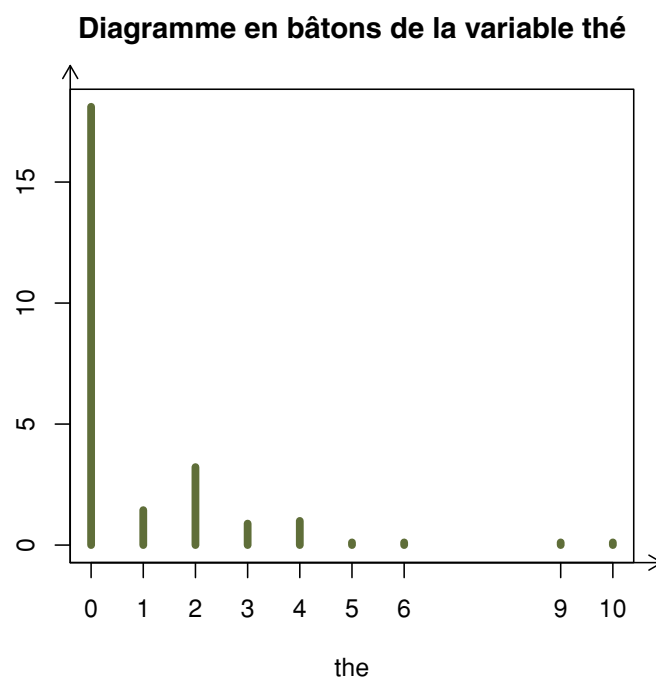


FIGURE 9.8 – Diagramme en bâtons pour une variable quantitative discrète.

### 9.6.3.3 Graphe de la fonction de répartition empirique

Il s'obtient au moyen des fonctions `plot()` et `ecdf()`.

```
> plot(ecdf(na.omit(cafe)),main=paste("Fonction de répartition
+ empirique", "de la variable café", sep="\n"),verticals=TRUE,
+ ylab=expression(F[n](x)),col.l0line="#89413A",col.points=
+ "#6D1EFF",col.hor='
> fleches(x=TRUE,y=TRUE)
```

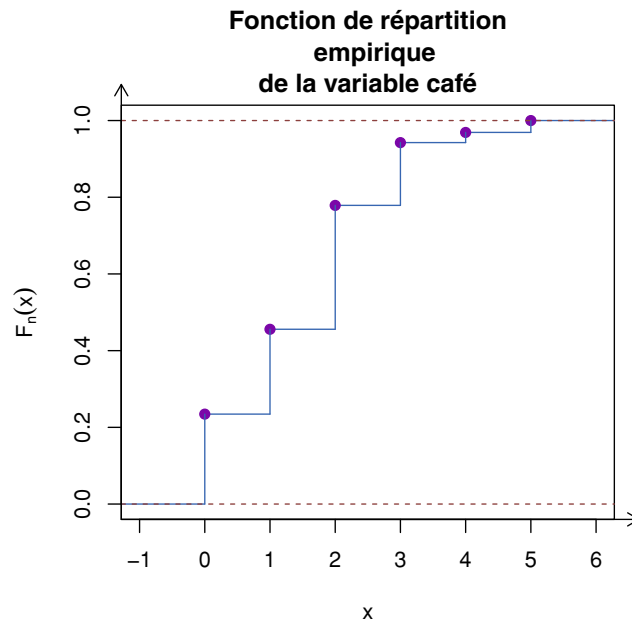


FIGURE 9.9 – Graphe de la fonction de répartition empirique pour une variable quantitative discrète.

#### 9.6.3.4 Diagramme en tiges et feuilles

Il s'obtient au moyen de la fonction `stem.leaf()` du package `aplpack`.

##### Renvoi

Voir la partie sur les variables quantitatives continues, page 397.

#### 9.6.3.5 Boîte à moustaches (*boxplot*)

Afin de tracer un diagramme en boîte à moustaches, il faut utiliser la fonction `boxplot()` qui produit le graphique ci-après. Le schéma en explicite la lecture.

```
> boxplot(cafe,col="orange",  
+ main="Boxplot de la variable café")  
> fleches(x=FALSE,y=TRUE)
```

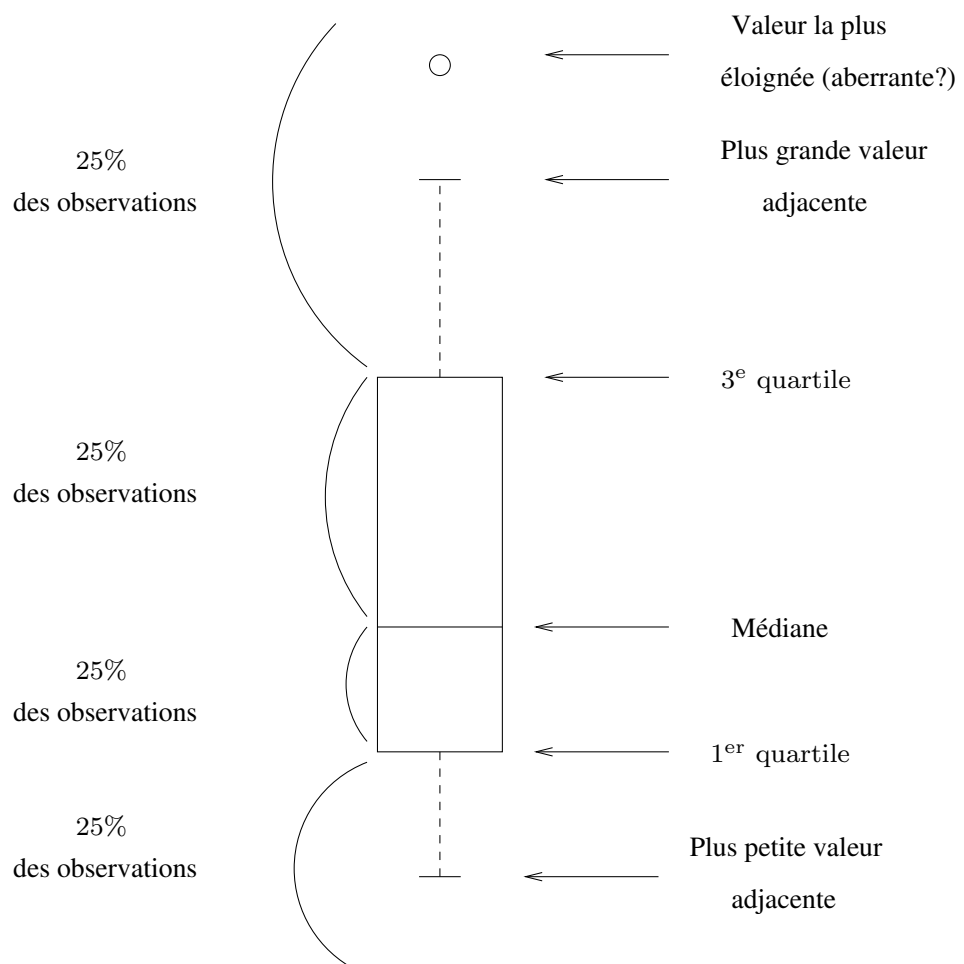
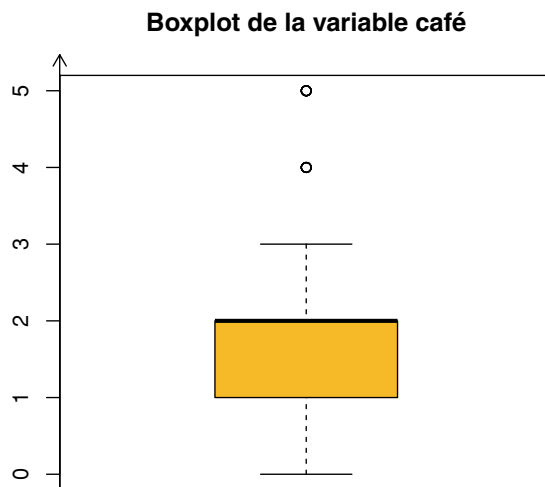


FIGURE 9.10 – Boîte à moustaches et explications associées.

La boîte est tracée en se servant des valeurs des trois quartiles. Notez que les valeurs repérées par des petits cercles sont des valeurs hors norme, éventuellement suspectes ou aberrantes (`boxplot(cafe)$out`). Ces valeurs extrêmes sont celles qui se situent à l'extérieur de la boîte, au-delà d'une distance de 1.5 fois l'intervalle inter-quartiles (le paramètre `range` permet de modifier cette valeur par défaut 1.5). Notez également que les valeurs se situant à l'extérieur de la boîte, mais à une distance en deçà de 1.5 fois l'intervalle inter-quartile sont des valeurs dites adjacentes. Les deux moustaches sont tracées respectivement à la plus grande et à la plus petite valeur adjacente.

**Astuce**

Tapez `example(bxp)` ou `example(boxplot)` pour d'autres exemples de ce type de diagramme.

## 9.6.4 Graphiques pour les variables quantitatives continues

Nous présentons maintenant quelques graphiques utiles pour l'exploration de données quantitatives.

### 9.6.4.1 Graphe de la fonction de répartition empirique

Il faut utiliser les fonctions `plot()` et `ecdf()`.

```
> plot(ecdf(na.omit(age)),main=paste("Fonction de répartition
+ empirique","de la variable age",sep="\n"),col.hor='# 3971FF',
+ col.points='#3971FF')
> fleches()
```

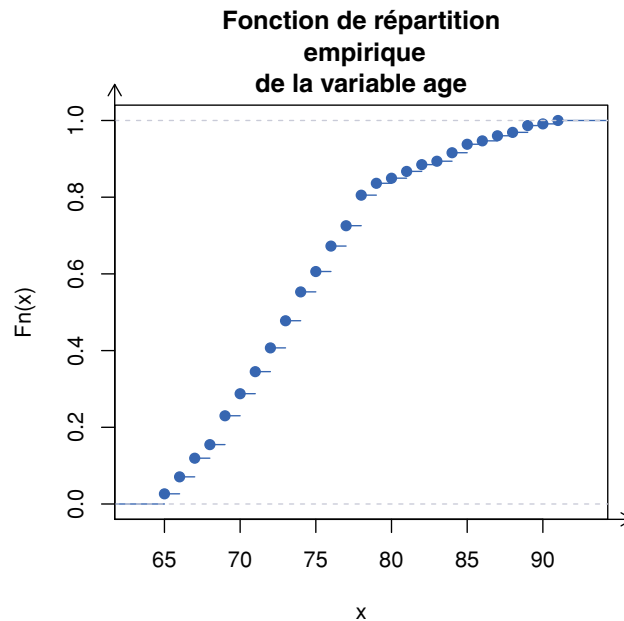


FIGURE 9.11 – Graphe de la fonction de répartition empirique pour une variable quantitative continue.

#### 9.6.4.2 Diagramme en tiges et feuilles

On peut utiliser la fonction `stem()`, mais elle est assez limitée et nous conseillons plutôt l'utilisation de la fonction plus évoluée `stem.leaf()` du package `aplpack`. La construction du diagramme se fait en trois étapes :

- choix d'une unité de tige au moyen du paramètre `unit` ;
- nombre de parties (1, 2 ou 5) en lesquelles chaque tige sera divisée via le paramètre `m` ;
- choix d'un style de représentation en utilisant le paramètre `style` qui peut prendre les valeurs "Tukey" ou "bare".

```
> require("aplpack")
> stem.leaf(taille,m=5,style="bare")
1 | 2: represents 12
leaf unit: 1
      n: 226
  1   14 | 0
      14 |
      14 |
      14 |
  3   14 | 88
 12   15 | 000001111
```

```

24    15 | 222223333333
45    15 | 44444444444455555555
60    15 | 666666666667777
73    15 | 8888888899999
97    16 | 0000000000000000000011
(22)  16 | 2222222222333333333333
107   16 | 4444444455555555555555
85    16 | 6666777
78    16 | 88888888888999
64    17 | 00000000011111
49    17 | 222222222333
36    17 | 44455555
27    17 | 6666666777
17    17 | 88889
12    18 | 0011
8      18 | 22
      18 |
6      18 | 666
3      18 | 888

```

#### 9.6.4.3 Boîte à moustaches

Elle s'obtient au moyen de la fonction `boxplot()`.

##### Renvoi

Voir la partie sur les variables quantitatives discrètes, page 394.

#### 9.6.4.4 Histogramme en densité à amplitudes de classes égales ou inégales

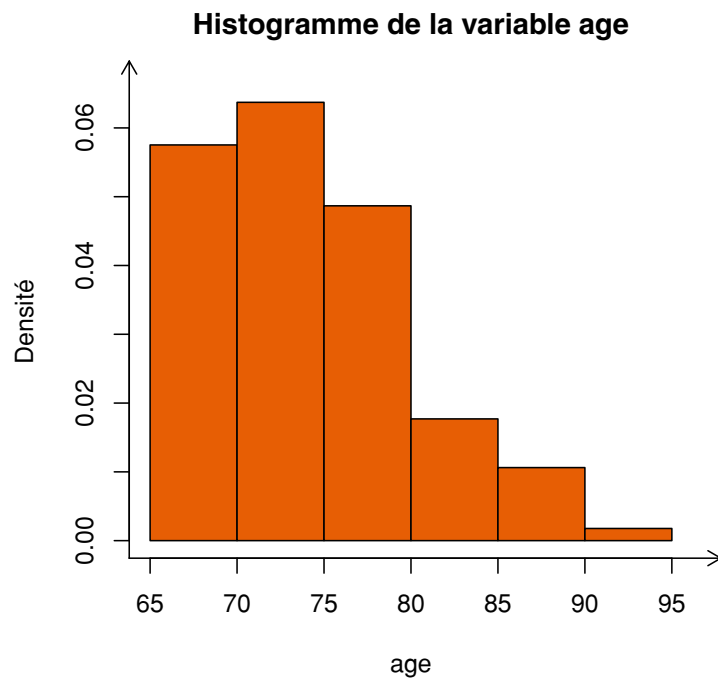
Il faut utiliser la fonction `hist()`. Notez que par défaut, **R** construit des classes de la forme  $[a; b]$  (« standard » anglo-saxon), alors qu'en France l'habitude est plutôt de faire des classes  $[a; b[$ .

```

> classes <- hist(age, right=TRUE, freq=FALSE, ylab="Densité",
+   main="Histogramme de la variable age", col="orangered")
> fleches()

```





```
> classes <- hist(poids, right=TRUE, freq=FALSE,
+ main="Histogramme de la variable poids",
+ ylab="Densite", breaks=c(min(poids), 50, 80,
+ 90, max(poids)), col="olivedrab")
> fleches()
```

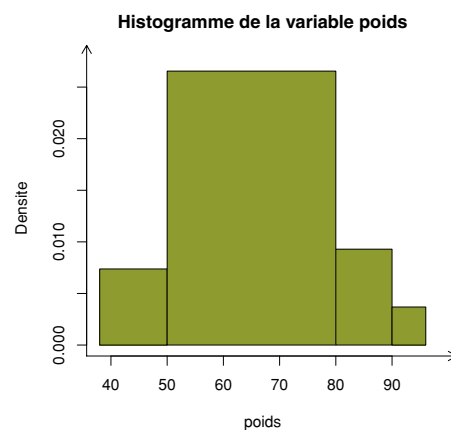


FIGURE 9.12 – Histogramme en densité à amplitudes de classes égales ou in-égales.

#### 9.6.4.5 Polygone des fréquences

On utilise les fonctions `hist()` et `segments()`.

```
> classes <- hist(taille, right=TRUE, freq=FALSE,  
+ main=paste("Histogramme et polygone des fréquences",  
+           "de la variable taille", sep="\n"), col="orangered")  
> milieux <- classes$mid ; mlon <- length(milieux)  
> densites <- classes$density  
> segments(milieux[1:mlon-1], densites[1: mlon-1],  
+          milieux[2:mlon], densites[2:mlon], col=  
+          rgb(0.4196078, 0.4196078, 0.1372549, 0.9), lwd=3)  
> fleches()
```

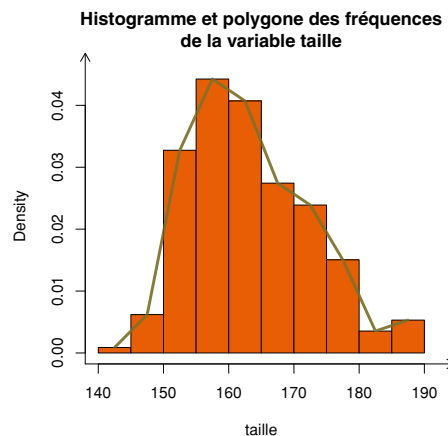


FIGURE 9.13 – Polygone des fréquences.

#### 9.6.4.6 Polygone des fréquences cumulées

On utilise les fonctions `hist()`, `ecdf()` et `plot()`.

```
> bornes <- hist(taille, right=TRUE, plot=FALSE)$breaks
> plot(bornes, ecdf(taille)(bornes), type="l", main=paste("Polygone
+   des fréquences cumulées", "de la variable taille", sep="\n"),
+   ylab="Fréquences", col="darkolivegreen", lwd=3)
> fleches()
```

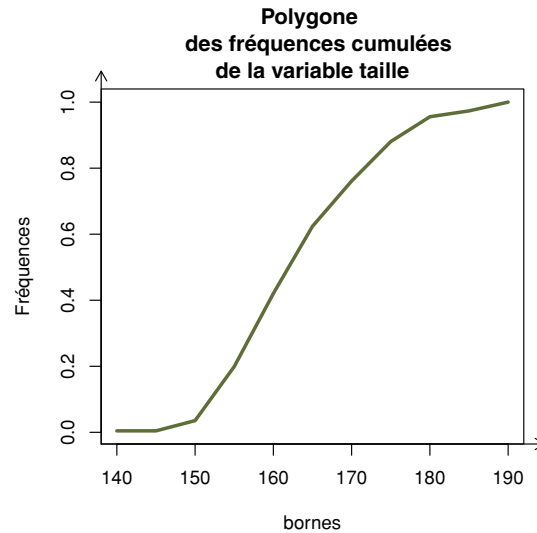


FIGURE 9.14 – Polygone des fréquences cumulées.

**Astuce**

Il est possible d'évaluer graphiquement la médiane sur ce graphique. Pour cela, il suffit de rajouter une ligne horizontale à la hauteur  $h = 0.5$  au moyen de l'instruction `abline(h=0.5)`. Ensuite, vous entrez l'instruction `locator(1)$x` puis vous cliquez sur le point d'intersection entre cette ligne horizontale et la courbe du polygone des fréquences cumulées. On peut bien entendu obtenir les autres quantiles en remplaçant la valeur 0.5 correspondant à la médiane par la valeur désirée.



## 9.6.5 Représentations graphiques dans un cadre bivarié

Nous présentons ici quelques représentations utiles dans un cadre bivarié.

### 9.6.5.1 Croisement de deux variables qualitatives

Il est possible de superposer deux diagrammes en tuyaux d'orgue comme on peut le voir sur les deux figures suivantes.

```
> tss <- prop.table(table(sexe, situation), 1)
```

```
> barplot(tss,bes=TRUE,leg=TRUE)
> title(paste("Diagrammes en tuyaux d'orgue de la situation",
+ "en fonction du sexe",sep="\n"))
> fleches(FALSE,TRUE)
```

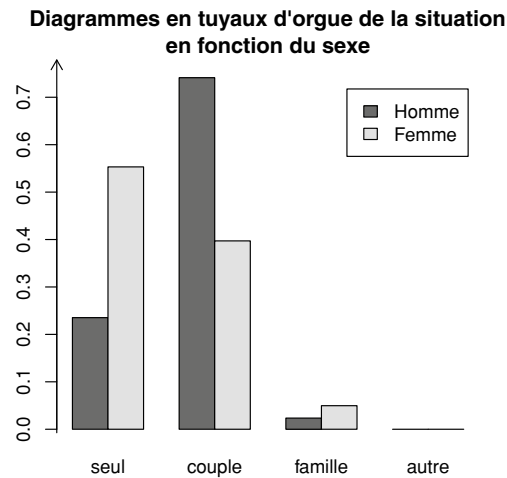


FIGURE 9.15 – Diagramme en tuyaux d'orgue pour deux variables qualitatives.

Le diagramme mosaïque peut aussi être utile pour le croisement de deux variables qualitatives.

```
> par(las=1) # Écriture horizontale des modalités.
> mosaicplot(sexe~matgras,color=brewer.pal(5,"Set1"),
+           main="Mosaicplot de matgras en fonction de age")
```

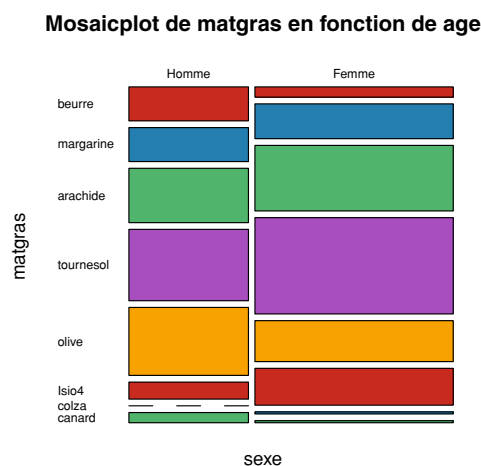


FIGURE 9.16 – Diagramme mosaïque pour le croisement de deux variables qualitatives.

## Astuce

Une autre fonction intéressante dans ce contexte est la fonction `assocplot()` qui produit un graphique d'association de Cohen-Friendly indiquant les déviations par rapport à l'indépendance dans une table de contingence  $2 \times 2$ . Mentionnons aussi la fonction `spineplot()`.



```
> assocplot(table(sexe, matgras))
```

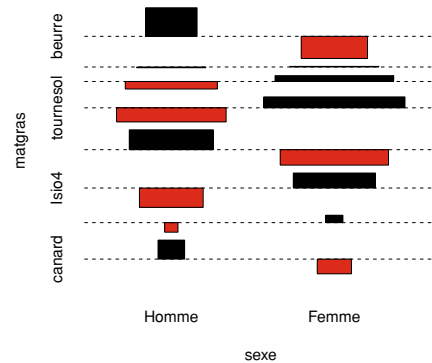


FIGURE 9.17 – Graphique d'association de Cohen-Friendly croisant deux variables qualitatives.

Une dernière fonction intéressante dans ce contexte est la fonction `table.cont()` du package `ade4`.

```
> require("ade4")
> sexematgras <- table(sexe, matgras)
> table.cont(sexematgras, row.labels=rownames(sexematgras),
+           col.labels=colnames(sexematgras))
```

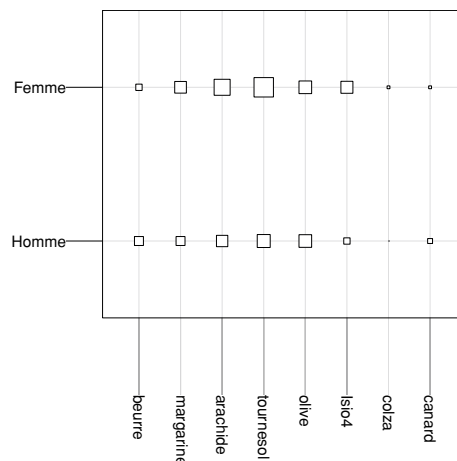
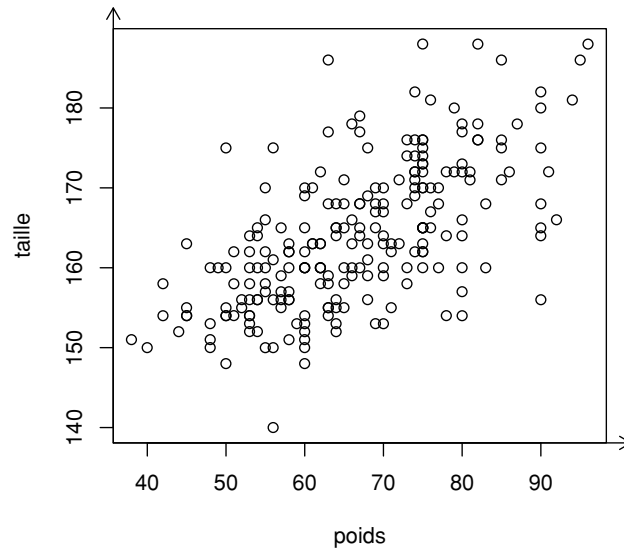


FIGURE 9.18 – Graphique `table.cont` croisant deux variables qualitatives.

### 9.6.5.2 Croisement de deux variables quantitatives

La fonction à utiliser dans ce contexte est la fonction `plot()`.

```
> plot(taille~poids)
> fleches()
```



Nous pouvons agrémenter le graphique ci-dessus en utilisant ce que nous avons vu dans le chapitre 5 pour créer une fonction `flashy.plot()`.

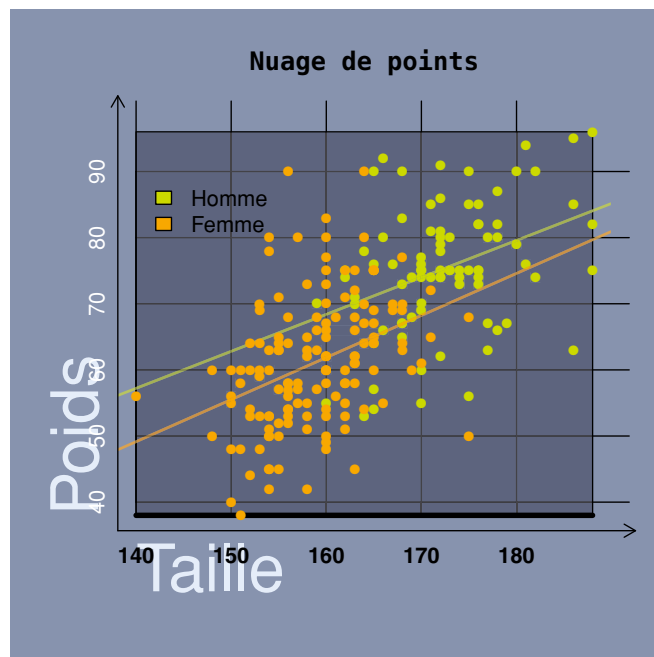


FIGURE 9.19 – Graphique croisant deux variables quantitatives.

### 9.6.5.3 Croisement d'une variable qualitative et d'une variable quantitative

Dans ce contexte, il est intéressant de tracer des diagrammes en boîte à moustaches (*boxplots*) de la variable quantitative pour chaque modalité de la variable qualitative. Si les variables ont été correctement structurées dans **R**, il suffit d'utiliser la fonction `plot()`.

```
> par(bty="n")
> plot(cafe~sexe,col=brewer.pal(5,"Set2"),notch=TRUE,varwidth=TRUE,
+      boxwex=0.3)
> title(paste("Boxplot de la consommation de café",
+ "en fonction du sexe",sep="\n"),family="Courier")
> fleches(FALSE,TRUE)
```

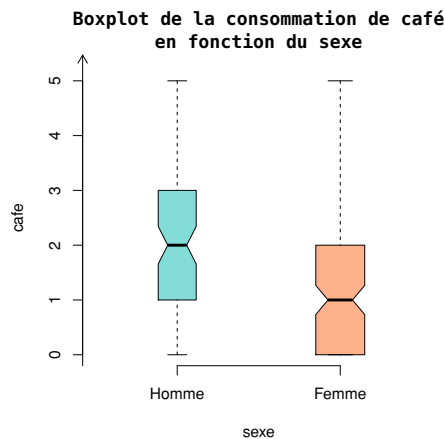


FIGURE 9.20 – *Boxplots* d'une variable quantitative selon les niveaux d'une variable qualitative.

Notons également l'existence de la fonction `stripchart()` qui permet d'obtenir le graphique suivant :

```
> stripchart(fruit_crus~age,data=nutriage)
> fleches(y=TRUE)
```

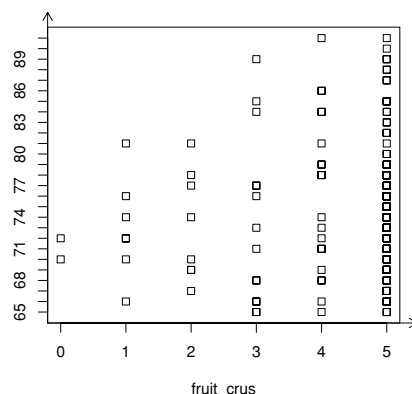


FIGURE 9.21 – Diagramme `stripchart` croisant une variable quantitative avec une variable qualitative.

## Termes à retenir

`as.factor()` : transforme une variable en facteurs  
`levels()` : affiche ou affecte les niveaux d'un facteur  
`as.ordered()` : transforme une variable en facteurs ordonnés  
`as.integer()` : pour structurer une variable discrète  
`as.double()` : pour structurer une variable continue  
`table()` : renvoie le tri à plat d'une variable ou crée une table de contingence entre deux variables  
`addmargins()` : rajoute les marges à une table de contingence  
`margin.table()` : renvoie les distributions marginales d'une table de contingence  
`prop.table()` : calcule les distributions conditionnelles à partir d'une table de contingence  
`na.omit()` : supprime les valeurs manquantes (NA) d'une variable  
`median()` : renvoie la médiane d'un vecteur  
`mean()` : renvoie la moyenne d'un vecteur  
`quantile()` : renvoie les quantiles d'un vecteur  
`summary()` : appliquée à une série numérique, renvoie minimum, maximum, quartiles et moyenne  
`range()` : renvoie le minimum et le maximum  
`IQR()` : renvoie l'intervalle inter-quartiles  
`var()` : renvoie la variance de l'échantillon  
`sd()` : renvoie l'écart type de l'échantillon  
`mad()` : déviation absolue par rapport à la médiane  
`chisq.test()` : calcul de la statistique du khi-2  
`cor.test()` : calcul du coefficient de corrélation de Pearson, du  $\tau$  de Kendall ou du  $\rho$  de Spearman  
`cov()` : renvoie la covariance  
`cor()` : renvoie la corrélation  
`barplot()` : trace un diagramme en tuyaux d'orgue  
`pie()` : trace un diagramme circulaire  
`plot.ecdf()` : trace la courbe de la fonction de répartition empirique  
`stem()` : affiche un diagramme en tiges et feuilles  
`boxplot()` : trace un diagramme en boîte à moustaches  
`hist()` : trace un histogramme  
`plot()` : permet de tracer un nuage de points



## Exercices

- 9.1-** Donnez l'instruction permettant d'obtenir le tableau des fréquences de la variable qualitative `x`.
- 9.2-** Donnez l'instruction permettant d'obtenir la table de contingence des variables qualitatives `x` et `y`.



- 9.3- Quelle est la fonction permettant d'obtenir les distributions marginales à partir d'une table de contingence ?
- 9.4- Quelle est la fonction permettant d'obtenir les distributions conditionnelles à partir d'une table de contingence ?
- 9.5- Donnez l'instruction permettant d'obtenir le mode d'une distribution.
- 9.6- Donnez l'instruction permettant de calculer l'étendue du vecteur  $\mathbf{x}$ .
- 9.7- Donnez l'instruction permettant de calculer l'intervalle inter-quartiles du vecteur  $\mathbf{x}$ .
- 9.8- Donnez l'instruction permettant de calculer la variance (non empirique) du vecteur  $\mathbf{x}$ .
- 9.9- Donnez le code d'un programme permettant de calculer le coefficient de variation.
- 9.10- Donnez l'instruction permettant de calculer l'écart moyen.
- 9.11- Quel *package* contient des fonctions permettant de calculer le *skewness* et le *kurtosis* ?
- 9.12- Donnez l'instruction permettant de calculer le  $\Phi^2$  de Cramér.
- 9.13- Donnez le code d'un programme permettant de calculer le rapport de corrélation  $\eta_{Y|X}^2$ .
- 9.14- Quelle est la fonction à utiliser pour obtenir un diagramme de Pareto ?
- 9.15- Quelle est la fonction à utiliser pour obtenir un diagramme empilé ?
- 9.16- Quelle est la fonction à utiliser pour obtenir un diagramme circulaire ?
- 9.17- Quelle est la fonction à utiliser pour obtenir un diagramme en boîte à moustaches ?
- 9.18- Quelle est la fonction à utiliser pour obtenir un histogramme ?



## Fiche de TP

### Études descriptives de données

#### A - Réflexion sur l'indépendance en statistique descriptive

- 9.1- Importez le fichier <http://www.biostatisticien.eu/springer/snee74.txt> dans un objet **R** nommé `snee`.
- 9.2- Affichez les premières et les dernières lignes de `snee` à l'aide des fonctions `head()` et `tail()`. Combien y a-t-il d'individus ? de variables ? De quel type sont les variables ?
- 9.3- Utilisez la fonction `attach()` sur votre *data.frame* puis vérifiez au moyen des fonctions `class()` et `levels()` que vos variables sont correctement structurées. Quelles sont les modalités des variables ?

- 9.4-** Faites une étude descriptive univariée de chacune des trois variables prises séparément : résumés numériques et représentations graphiques adéquats.
- 9.5-** Nous allons maintenant étudier la dépendance entre les variables **yeux** et **cheveux**. Créez le tableau de contingence **yeuxcheveux** (effectifs observés) des variables **yeux** et **cheveux**.
- 9.6-** Calculez les fréquences de chacune des modalités de la variable **cheveux** (profils colonnes). Nous obtenons donc la répartition **fchev** des différentes couleurs de cheveux dans la population.
- 9.7-** Faisons maintenant intervenir le deuxième caractère : la couleur des yeux. Calculez le nombre d'individus **nbleus** ayant les yeux bleus dans toute la population.
- 9.8-** On suppose qu'il y a indépendance entre la couleur des yeux et la couleur des cheveux. Autrement dit, le fait que des individus ont les yeux de cette couleur (bleu) est sans relation avec la couleur de leurs cheveux. Par conséquent, parmi la sous-population des gens qui ont les yeux bleus, les proportions calculées en 9.6 devraient être respectées. Calculez alors maintenant, sous cette hypothèse d'indépendance, le nombre de personnes aux yeux bleus qui devraient avoir les cheveux blonds (respectivement marron, noirs et roux).
- 9.9-** Faites de même pour chacune des autres couleurs d'yeux. Vous devez donc obtenir le tableau **tab.ind1** dit des effectifs théoriques sous l'hypothèse d'indépendance, entre la couleur des yeux et la couleur des cheveux.
- 9.10-** Recommençons toute l'opération, mais en inversant les deux caractères, c'est-à-dire en partant cette fois-ci de la variable **yeux**. Calculez, à partir du tableau **yeuxcheveux**, les fréquences de chacune des modalités de la variable **yeux** (profils lignes). Nous obtenons donc la répartition **fyeux** des différentes couleurs d'yeux dans la population.
- 9.11-** Faisons maintenant intervenir le deuxième caractère : la couleur des cheveux. Calculez le nombre d'individus **nblonds** ayant les cheveux blonds dans toute la population.
- 9.12-** On suppose qu'il y a indépendance entre la couleur des cheveux et la couleur des yeux. Autrement dit, le fait que des individus ont les cheveux de cette couleur (blonds) est sans relation avec la couleur de leurs yeux. Par conséquent, parmi la sous-population des gens qui ont les cheveux blonds, les proportions calculées en 9.10 devraient être respectées. Calculez alors maintenant, sous cette hypothèse d'indépendance, le nombre de personnes qui devraient avoir les yeux bleus (respectivement marron, noisette et verts).
- 9.13-** Faites de même pour chacune des autres couleurs de cheveux. Vous devez donc obtenir le tableau **tab.ind2** dit des effectifs théoriques sous l'hypothèse d'indépendance, entre la couleur des cheveux et la couleur des yeux.

- 9.14-** Comparez, à l'aide de la fonction `all.equal()`, les deux tableaux d'effectifs théoriques obtenus. Que constatez-vous ?
- 9.15-** Comparez maintenant le tableau des effectifs observés `yeuxcheveux` à celui des effectifs théoriques (pour chacune des cases, on calcule le carré de la différence).
- 9.16-** Calculez le tableau des contributions au  $\chi^2$ .
- 9.17-** Calculez tous les indicateurs de liaison appropriés. Concluez.
- 9.18-** L'indépendance est aussi définie (et c'est même sa définition première) comme étant l'égalité de toutes les distributions conditionnelles. Calculez la distribution conditionnelle de la variable `cheveux` sachant que la couleur des yeux est bleue (c'est-à-dire les répartitions des différentes couleurs de cheveux sachant que la couleur des yeux est fixée à bleue). Calculez les trois autres distributions conditionnelles de la variable `cheveux`. Calculez les distributions conditionnelles de la variable `yeux` connaissant chacune des modalités de la variable `cheveux`. Concluez.
- 9.19-** Faites une étude descriptive de la variable `cheveux` en fonction de la variable `sexe`.
- 9.20-** Faites une étude descriptive de la variable `yeux` en fonction de la variable `sexe`.
- 9.21-** Analysez maintenant la dépendance entre la couleur des yeux et la couleur des cheveux pour le tableau de contingence suivant :

		Cheveux				
		Blonds	Bruns	Noirs	Roux	Total
Yeux	Bleus	1 768	807	189	47	2 811
	Gris-vert	946	1 387	746	53	3 132
	Bruns	115	438	288	16	857
	Total	2 829	2 632	1 223	116	6 800

## B- Analyse descriptive du jeu de données NUTRIAGE

Nous vous proposons de résoudre ces questions diverses sur le jeu de données NUTRIAGE, dans lesquelles des erreurs ont été délibérément introduites.

- 9.1-** Importez le fichier de données `nutriage.xls`.
- 9.2-** Donnez le mode absolu de la variable `situation`, de la variable `chocol` et de la variable `taille`.
- 9.3-** Choisissez des classes pour la variable `taille` et donnez la classe modale.
- 9.4-** Calculez la médiane de la variable `chocol`.
- 9.5-** Faites un tri à plat de la variable `chocol` et de la variable `fruit_crus`.

- 9.6- En vous fondant **uniquement** sur ces tris à plat, donnez la médiane de ces deux variables.
- 9.7- Calculez les quartiles de la variable `taille` en utilisant les classes choisies précédemment.
- 9.8- Tracez le polygone des fréquences cumulées de la variable `taille` et évaluez, sur ce graphique, les quartiles de la distribution.
- 9.9- Calculez en utilisant les données individuelles la moyenne des variables `taille`, `poids` et `age`.
- 9.10- Faites le tri à plat de la variable `the`, et en vous fondant sur ce tri à plat, calculez-en la moyenne.
- 9.11- Calculez la moyenne de la variable `taille` en utilisant les classes choisies précédemment.
- 9.12- Calculez l'étendue de la variable `poids`.
- 9.13- Tracez un *boxplot* de la variable `poids`.
- 9.14- Calculez, en utilisant les données individuelles, l'écart type de la variable `taille`.
- 9.15- En vous fondant **uniquement** sur le tri à plat de la variable `the`, calculez-en le coefficient de variation.
- 9.16- Calculez la variance totale, la variance *inter* et la variance *intra* de la variable `cafe` en considérant que la population est partitionnée en deux sous-groupes : les hommes et les femmes. Calculez le coefficient  $\eta^2$ .

### C- Analyse descriptive de jeux de données

- 9.1- Faites une analyse de statistique descriptive du jeu de données POIDS.-  
NAISSANCE.
- 9.2- Faites une analyse de statistique descriptive du jeu de données INFARC-  
TUS.