Lire ; Compter ; Tester... avec R

Préparation des données / Analyse univariée / Analyse bivariée

Christophe Genolini

Table des matières

Chapitre 1

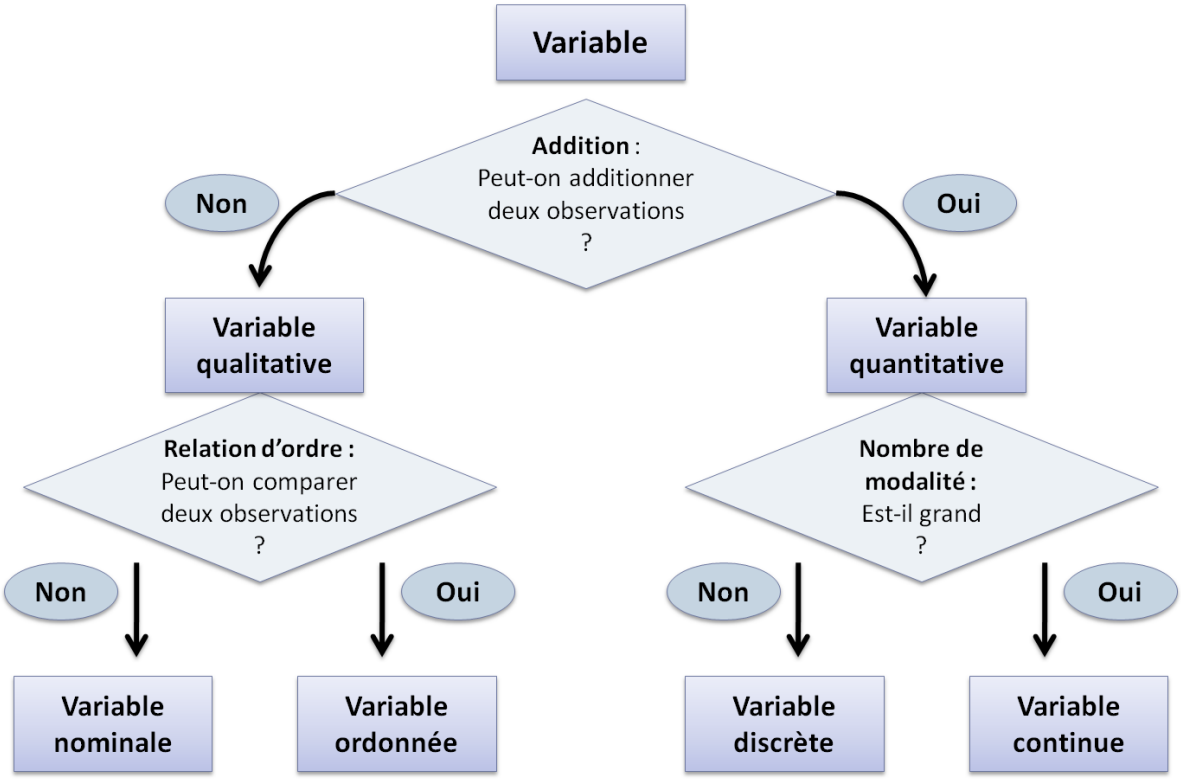
Rappels théoriques

1.1 Vocabulaire

Nom	**Définition**	**Exemple**
Individu	Objet étudié	un étudiant
Population	Ensemble des individus	tous les étudiants participant à l'expérience
Variables	Ce qui est étudié chez les individus (et qui varie d'un individu à l'autre)	`[Age]`, `[CigaretteJour]`, `[NiveauSportif]`
Modalités (d'une variable)	Liste de toutes les valeurs possibles pour une variable	Modalités de `[Age]` : de 0 ans à 120 ans
Observation	Valeurs prises par un individu particulier	Marc a 21 ans, `(21)` est une observation.

1.2 Nature d'une variable

La nature d'une variable détermine le type d'outil statistique qu'on pourra utiliser sur la variable. Pour déterminer son type, il faut se demander ce qu'on peut faire avec ses observations :



1.3 Principe de l'analyse univariée

L'analyse univariée permet de mieux appréhender une variable. Elle comporte quatre étapes :

1. Calcul des effectifs
2. Calcul de la centralité
3. Calcul de la dispersion
4. Représentation graphique

Ces étapes varient selon le type de variable. Voila le détail des étapes en fonction du type de variable :

Étape	**Nominale**	**Ordonnée**	**Discrète**	**Continue**
1. Effectifs	A faire	A faire	A faire	Inutile
2. Centralité	Mode	Médiane	Moyenne et Médiane	Moyenne et Médiane
3. Dispersion	N'existe pas	Quartile	Écart type et quartiles	Écart type et quartile
4. Graphique	Histogramme des effectifs	Histogramme des effectifs	Histogramme des effectifs, boîte à moustache	Distribution et boîte à moustache

1.4 Principe de l'analyse bivariée

L'analyse bivariée consiste à étudier deux variables conjointement, puis éventuellement à tester le lien entre les deux variables.

Deux variables sont liées si connaître l'une donne des informations sur l'autre. Par exemple, connaître le sexe d'un individu permet d'en savoir un peu plus sur la longueur de ses cheveux. Attention, cela ne permet pas de *connaître* mais juste d'avoir une information plus précise. Par exemple, dans l'UFR STAPS, 20% des individus ont les cheveux longs. Si on détaille selon les sexes, 4% des garçons ont les cheveux long contre 55% des filles. Connaître le sexe d'un individu ne donne donc aucune certitude, mais permet d'avoir un peu plus d'informations.

Pour savoir si deux variables sont liées (avec un certain risque d'erreur, imcompressible), on utilise un test. Le test à utiliser dépend du type des variables et de leur propriétés :

Variables	**Test paramétrique**	**Diagnostic**	**Test non paramétrique**
Qualitative & Qualitative	χ^2	1. Les valeurs de toutes les cases du tableau des effectifs attendus doivent être supérieures ou égales a 5.	Test exact de Fisher
Qualitative (2 classes) **& Numérique**	T de Student	1. Les écart types sont égaux 2. Pour chaque groupe, la variable numérique suit une loi normale OU les effectifs sont supérieurs à 30.	Test des rangs de Wilcoxon
Qualitative (3 classes et plus) **& Numérique**	F de Fisher (ANOVA)	1. Les écart types sont égaux 2. Pour chaque groupe, la variable numérique suit une loi normale OU les effectifs sont supérieurs à 30.	Test de Kruskal-Wallis
Numérique & Numérique	R de Pearson	1. Au moins une des deux variables suit une loi normale.	R de Spearman

Chapitre 2

Préparation des données

2.1 Télécharger

Avant de lire les données, ils vous faut R... Vous pouvez le télécharger sur le site du CRAN : `http:`
`cran.r-paoject.org` puis *Download and Install R*. Cliquez ensuite sur votre système d'exploitation (Linux, MacOS X ou Windows) puis télépcharger R en cliquant sur `base`.

2.2 Lecture des données

Excel étant un logiciel propriétaire, il est difficile à un autre logiciel de lire le format `.xls`. Par contre, R sait lire les fichiers au format `.csv` . Donc, nous allons préparer un fichier `.csv`.

1. Ouvrez vos données sous Excel, Open Office, SPSS, SAS...
2. Dans le menu *Fichier* ou *File*, il existe probablement une option *Enregistrer sous* ou *Exporter*. Choisissez le format `.csv`. Si votre logiciel demande des précisions, vous pouvez lui spécifier : *séparateur décimal="."* et *séparateur=";"*. S'il ne demande rien, tout va bien.

Un fichier `.csv` vient d'être créé dans votre répertoire. Pour le lire à partir de R, il faut lui préciser le repertoire de lecture. Cela se fait à partir de la fenêtre R, dans le menu *Fichier → Changer le repertoire courant.*

Il est maintenant possible de lire vos données à partir de R grâce à l'instruction :

```
> ### Lecture des données
> read.csv2("FormationR.csv")
```

```
    id sexe  age taille niveau departement   UFR frereEtSoeur
1    1    F   22    1,7     L3          75  SJAP            0
2    2    F   20   1,66     L3          92 SEGMI            0
3    3    F                 L3          78 SEGMI            0
4    4    F   25   1,65     M2          75  SJAP            0
5    5    F  340   1,62     M2          92 STAPS            0
6 ...  ...  ...    ...    ...         ...   ...          ...
  rapportRisque transAvecPres rapportAge rapportSexuel scoreConnaissance
1           Non                       19           Oui                 3
2           Non           Non         18           Oui
3           Oui           Non         15           Oui                 2
4           Non                       17           Oui                 1
5           Oui           Non         21           Oui                 3
6          ...           ...        ...           ...               ...
```

Pour pouvoir manipuler ce fichier (et faire des statistiques dessus), il faut le stocker dans une variable de type un peu spécial qu'on appelle `data.frame`. Cela se fait à l'aide de la flèche d'affectation `<-`. Pour stocker vos données dans la variable `data`, tapez :

```
> ### Lecture des données avec stocakge
> data <- read.csv2("FormationR.csv")
```

Il ne se passe rien a l'écran, mais `data` contient maintenant vos données. Pour vérifiez que c'est bien le cas, tapez simplement `data`. R affiche alors le contenu de `data`, c'est à dire vos données.

```
> ### Vérification que les données sont en mémoire
> data
```

```
    id sexe  age taille niveau departement   UFR frereEtSoeur
1    1    F   22    1,7     L3          75  SJAP            0
2    2    F   20   1,66     L3          92 SEGMI            0
3    3    F                 L3          78 SEGMI            0
4    4    F   25   1,65     M2          75  SJAP            0
5    5    F  340   1,62     M2          92 STAPS            0
6 ...  ...  ...    ...    ...         ...   ...          ...
  rapportRisque transAvecPres rapportAge rapportSexuel scoreConnaissance
1           Non                       19           Oui                 3
2           Non           Non         18           Oui
3           Oui           Non         15           Oui                 2
4           Non                       17           Oui                 1
5           Oui           Non         21           Oui                 3
6           ...           ...        ...           ...               ...
```

Le pire ennemi du statisticien, tous les enquêteurs le savent, est la *valeur manquante*. En R, les valeurs manquantes sont codées `NA` ou `<NA>`. Dans votre fichier `.csv`, le codage de la valeur manquante dépend de votre logiciel : case vide pour Excel et Open office, un point pour SAS,... Il faut donc préciser à R le type de valeur manquante qu'il va rencontrer dans le fichier. Cela se fait en ajoutant l'option `na.string="codage_Manquante"` dans la ligne de lecture. Ainsi, si votre `.csv` a été obtenu à partir d'Excel ou Open office, l'instruction de lecteure est `read.csv2("nom_de_fichier.csv", na.string = "")`

```
> ### Lecture des données en considérant les manquantes
> data <- read.csv2("FormationR.csv",na.string="")
> data
```

```
    id sexe  age taille niveau departement   UFR frereEtSoeur
1    1    F   22    1,7     L3          75  SJAP            0
2    2    F   20   1,66     L3          92 SEGMI            0
3    3    F <NA>   <NA>     L3          78 SEGMI            0
4    4    F   25   1,65     M2          75  SJAP            0
5    5    F  340   1,62     M2          92 STAPS            0
6 ...  ...  ...    ...    ...         ...   ...          ...
  rapportRisque transAvecPres rapportAge rapportSexuel scoreConnaissance
1           Non          <NA>         19           Oui                 3
2           Non           Non         18           Oui              <NA>
3           Oui           Non         15           Oui                 2
4           Non          <NA>         17           Oui                 1
5           Oui           Non         21           Oui                 3
6           ...           ...        ...           ...               ...
```

On constate que les cases vides ont été remplacées par des `NA` ou des `<NA>`. Si vous aviez utilisé SAS, l'instuction aurait été `data <- read.csv2("nom_de_fichier.csv",na.string=".")`.

2.3 Manipulation d'un data.frame

`data` est un data.frame, c'est-à-dire un tableau contenant vos données. Chaque colonne du tableau contient une variable. Chaque ligne du tableau est un individu. Pour travailler sur une colonne précise (par exemple la deuxième), tapez `data[,2]`. Vous pouvez également taper le nom du data.frame, puis le symbole $ suivi du nom de la colonne :

```
> ### Deuxième colonne
> data[,2]
```

```
 [1] F    F    <NA> F    F    F    F    F    F    F    F    F    F    F
[15] F    F    F    F    F    F    F    <NA> F    F    F    H    H    H
[29] H    H    H    H    H    H    H    H    H    H    H    H
Levels: F H
```

```
> ### Colonne sexe
> data$sexe
```

```
 [1] F    F    <NA> F    F    F    F    F    F    F    F    F    F    F
[15] F    F    F    F    F    F    F    <NA> F    F    F    H    H    H
[29] H    H    H    H    H    H    H    H    H    H    H    H
Levels: F H
```

R affiche le contenu de la colonne. Il indique également les modalités de la variable (`Levels`).

Pour accéder à une ligne (par exemple la troisième), tapez `data[3,]`

```
> ### Troisième ligne
> data[3,]
```

```
  id sexe age taille niveau departement   UFR frereEtSoeur rapportRisque
3  3 <NA>  NA     NA     L3          78 SEGMI            0           Oui
  transAvecPres rapportAge rapportSexuel scoreConnaissance
3           Non         15           Oui                 2
```

Pour accéder à une colonne et une ligne, on combine les deux : `data[3,2]` nous donne la valeur du troisième individu, deuxième colonne ; `data$sexe[3]` donne la troisième valeur de la colonne `sexe`.

```
> ### Affichage d'une valeur précise
> data[3,2]
```

```
[1] <NA>
Levels: F H
```

```
> data$sexe[3]
```

```
[1] <NA>
Levels: F H
```

2.4 Modification d'une valeur

La modification d'une valeur se fait grâce à l'opérateur `<-`. L'instruction `a <- 5` a pour effet de créer la variable `a` et de placer la valeur 5 dans cette variable. Dans le cas d'un data.frame, on peut souhaiter modifier une valeur particulière. Par exemple, l'individu 5 a pour age `340`, ce qui semble plutôt improbable. Après vérification, il s'agit simplement d'une erreur de saisie, la vraie valeur est 34. Il faut donc remplacer `340` par `34`. Cela se fait avec `<-`.

```
> ### Variable age
> data$age
```

```
 [1]  22  20  NA  25 340  20  19  18  21  21  21  19  18  21  24  21  21
[18]  21  20  NA  19  NA  21  21  21  24  19  23  20  22  23  23  21  23
[35]  21  22  22  22  19  22
```

```
> ### Cinquième valeur de age
> data$age[5]
```

```
[1] 340
```

```
> ### Modification de la cinquième valeur
> data$age[5] <- 34
> ### Vérification
> data$age
```

```
 [1] 22 20 NA 25 34 20 19 18 21 21 21 19 18 21 24 21 21 21 20 NA 19 NA 21
[24] 21 21 24 19 23 20 22 23 23 21 23 21 22 22 22 19 22
```

La cinquième valeur de la colonne `age` a été corrigée.

2.5 Type de variable

Chaque colonne correspond à une variable et a donc une type. Les différents type de variables statistique décrites section 1.2 page 5 correspondent aux types R suivant :

En statistique	**Sous R**
Nominale	`factor`
Ordonnée	`ordered`
Discrète	`numeric` (ou `integer`)
Continue	`numeric` (ou `integer`)

Quand R charge un ficher en mémoire (dans `data`), il donne a chaque variable un type. Pour connaître le type d'une variable, on utilise `str`. Cela liste toutes les variables avec leur type, leurs modalités et les premières observations.

```
> ### Le type des colonnes
> str(data)
```

```
'data.frame':   40 obs. of  13 variables:
 $ id               : int  1 2 3 4 ...
 $ sexe             : Factor w/ 2 levels "F","H": 1 1 NA 1 ...
 $ age              : num  22 20 NA 25 ...
 $ taille           : num  1.7 1.66 NA 1.65 ...
 $ niveau           : Factor w/ 5 levels "L1","L2","L3",..: 3 3 3 5 ...
 $ departement      : int  75 92 78 75 ...
 $ UFR              : Factor w/ 3 levels "SEGMI","SJAP",..: 2 1 1 2 ...
 $ frereEtSoeur     : int  0 0 0 0 ...
 $ rapportRisque    : Factor w/ 2 levels "Non","Oui": 1 1 2 1 ...
 $ transAvecPres    : Factor w/ 2 levels "Non","Oui": NA 1 1 NA ...
 $ rapportAge       : int  19 18 15 17 ...
 $ rapportSexuel    : Factor w/ 2 levels "Non","Oui": 2 2 2 2 ...
 $ scoreConnaissance: int  3 NA 2 1 ...
```

Dans un certain nombre de cas, R n'a pas possibilité de donner le type correct : il n'a aucun moyen d'identifier les variables ordonnées (il les prend pour des `factor`) car il ne connait pas la relation d'ordre qui s'applique. C'est par exemple le cas de la variable `[niveau]`. De même, il ne peut pas identifier une variable nominale dont les modalités seraient des chiffres (comme les numéros de département). Nous allons donc devoir corriger ses choix.

Pour transformer une variable numérique en facteur, il faut utiliser la fonction `as.factor`. `as.factor(data$departement)` permet de considérer la colonne `data$departement` non plus comme une variable numerique mais comme une nominale. Toutefois, pour que la variable departement soit modifiée de manière durable au sein du data.frame, il faut remplacer la colonne département par la variable avec son nouveau type. Encore une fois, cela se fait avec l'opérateur d'affectation `<-` :

```
> ### Modification du type de departement
> data$departement <- as.factor(data$departement)
```

Ainsi, la colonne `departement` du data.frame `data` (à gauche de la flèche) est remplacée (la flèche) par la colonne `departement` *transformé* en `factor` (à droite de la flèche). Vérification :

```
> ### Le type des colonnes
> str(data)
```

```
'data.frame':   40 obs. of  13 variables:
 $ id               : int  1 2 3 4 ...
 $ sexe             : Factor w/ 2 levels "F","H": 1 1 NA 1 ...
 $ age              : num  22 20 NA 25 ...
 $ taille           : num  1.7 1.66 NA 1.65 ...
 $ niveau           : Factor w/ 5 levels "L1","L2","L3",..: 3 3 3 5 ...
 $ departement      : Factor w/ 9 levels "1","21","55",..: 5 8 6 5 ...
 $ UFR              : Factor w/ 3 levels "SEGMI","SJAP",..: 2 1 1 2 ...
 $ frereEtSoeur     : int  0 0 0 0 ...
 $ rapportRisque    : Factor w/ 2 levels "Non","Oui": 1 1 2 1 ...
 $ transAvecPres    : Factor w/ 2 levels "Non","Oui": NA 1 1 NA ...
 $ rapportAge       : int  19 18 15 17 ...
 $ rapportSexuel    : Factor w/ 2 levels "Non","Oui": 2 2 2 2 ...
 $ scoreConnaissance: int  3 NA 2 1 ...
```

Le type de département est bien modifié. De la même manière, l'identifiant n'est pas une variable numériques mais est nominale :

```
> ### Modification du type de id
> data$id <- as.factor(data$id)
```

Les autres changements de type fonctionnent sur le même principe. Pour la transformation d'une variable en numérique, on utilise `as.numeric`.

La transformation d'une variable en variable ordonnée se fait -oh surprise- avec l'instruction `ordered`[1]. Il faut en outre préciser à R la relation d'ordre utilisée. Cela se fait en spécifiant l'option `levels`. Par exemple, le `niveau` est une variable ordonnée L1 puis L2 puis L3 puis M1 puis M2 :

```
> ### Ordonnancement de niveau
> data$niveau<-ordered(data$niveau,levels=c("L1","L2","L3","M1","M2"))
> data$niveau
```

```
 [1] L3   L3   L3   M2   M2   L3   L2   L3   L3   L3   L3   L1   L3   L2
[15] L3   <NA> L2   L3   L3   L3   L1   L2   L3   L3   L3   L3   L3   L3
[29] L3   L3   M1   L3   L3   L3   L3   L3   L3   L3   L2   M2
Levels: L1 < L2 < L3 < M1 < M2
```

1. Surprise parce qu'on se serait attendu a `as.ordered`. R est hélas plein de surprises...

Chapitre 3

Analyse univariée

Nos variables sont maintenant prêtes, l'analyse univariée peut commencer. L'instruction `summary` a pour effet de calculer automatiquement une partie de cette analyse en l'adaptant au type de variable : effectifs pour les `factor` et les `ordered`, moyenne et quartile pour les `numeric` :

```
> ### Résumé des données
> summary(data)
```

```
       id        sexe          age             taille       niveau
 1      : 1   F    :23   Min.   :18.00   Min.   :1.600   L1  : 2
 2      : 1   H    :15   1st Qu.:20.00   1st Qu.:1.640   L2  : 5
 3      : 1   NA's: 2   Median :21.00   Median :1.670   L3  :28
 4      : 1             Mean   :21.46   Mean   :1.676   M1  : 1
 5      : 1             3rd Qu.:22.00   3rd Qu.:1.700   M2  : 3
 6      : 1             Max.   :34.00   Max.   :1.850   NA's: 1
 (Other):34             NA's   : 3.00   NA's   :2.000
  departement     UFR       frereEtSoeur    rapportRisque transAvecPres
 92      :15   SEGMI:12   Min.   :0.0000   Non :28       Non :22
 78      :11   SJAP :14   1st Qu.:0.0000   Oui : 5       Oui :15
 75      : 7   STAPS:13   Median :1.0000   NA's: 7       NA's: 3
 1       : 1   NA's : 1   Mean   :0.8718
 21      : 1              3rd Qu.:1.5000
 (Other): 4               Max.   :3.0000
 NA's    : 1              NA's   :1.0000
   rapportAge     rapportSexuel scoreConnaissance
 Min.   :14.00   Non : 4       Min.   :0.000
 1st Qu.:15.25   Oui :33       1st Qu.:2.000
 Median :17.00   NA's: 3       Median :3.000
 Mean   :16.97                 Mean   :2.686
 3rd Qu.:18.00                 3rd Qu.:3.500
 Max.   :21.00                 Max.   :6.000
 NA's   : 6.00                 NA's   :5.000
```

Cela permet de jeter un premier oeil sur nos variables. Des instructions plus spécifiques permettent une analyse plus précise.

3.1 Effectifs

Les effectifs se calculent pour les variables nominale, ordonnée et discrète. Cela se fait grâce à l'instruction `table` :

```
> ### Effectif de sexe
> table(data$sexe)
```

```
 F  H
23 15
```

```
> ### Effectif de niveau
> table(data$niveau)
```

```
L1 L2 L3 M1 M2
 2  5 28  1  3
```

```
> ### Effectif de frereEtSoeur
> table(data$frereEtSoeur)
```

```
 0  1  2  3
18 11  7  3
```

On note au passage que le tableau des effectifs d'une variable continue est possible à calculer, mais qu'il ne donne pas d'information pertinante [1] :

```
> table(data$taille)
```

```
 1.6 1.61 1.62 1.63 1.64 1.65 1.66 1.67 1.68 1.69  1.7 1.72 1.73 1.74 1.85
   1    1    2    3    4    3    4    2    3    2    5    4    2    1    1
```

3.2 Centralité

3.2.1 Mode

Le mode s'obtient par lecture de la table des effectifs en prenant le plus grand. Si les modalités sont très nombreuse, on peut trier les effectifs avec l'instruction `sort` de manière décroissante en utilisant l'option `decreasing=TRUE` (afin que le mode soit en tête).

```
> ### Mode de niveau
> sort(table(data$niveau),decreasing=TRUE)
```

```
L3 L2 M2 L1 M1
28  5  3  2  1
```

Le mode de `taille` n'a pas d'intérêt, mais si nous devions le calculer, nous utiliserions :

```
> ### Mode de taille
> sort(table(data$taille),decreasing=TRUE)
```

```
 1.7 1.64 1.66 1.72 1.63 1.65 1.68 1.62 1.67 1.69 1.73  1.6 1.61 1.74 1.85
   5    4    4    4    3    3    3    2    2    2    2    1    1    1    1
```

3.2.2 Médiane

Médiane d'une numérique :

La médiane se calcule avec `median`. Quand la variable contient des valeurs manquante, il faut préciser à R de les supprimer en ajoutant l'option `na.rm=TRUE` :

```
> ### Mediane de taille
> median(data$taille,na.rm=TRUE)
```

```
[1] 1.67
```

1. Pour simplifier, nous travaillons sur un petit fichier de 40 lignes. Cela a pour effet de rendre les variables continue *presque* utilisable comme des nominales. En tout état de cause, avec une vrai variable continue sur 200 individus, les effectifs n'ont clairement plus aucun sens.

Médiane d'une ordonnée :

La médiane d'une variable ordonnée n'est pas calculée automatiquement par R. Il faut donc le faire "manuellement". Pour cela, trois étapes :

1. Calcul du rang de la médiane (après exclustion des manquantes).
2. Ordonnancement de la variable
3. Combinaison de 1 et 2, selection la modalité du milieu

Pour exclure les manquantes, on utilise `na.omit`.

```
> ### Exclusion des manquantes
> na.omit(data$niveau)
```

```
 [1] L3 L3 L3 M2 M2 L3 L2 L3 L3 L3 L3 L1 L3 L2 L3 L2 L3 L3 L3 L1 L2 L3 L3
[24] L3 L3 L3 L3 L3 L3 M1 L3 L3 L3 L3 L3 L3 L3 L2 M2
attr(,"na.action")
[1] 16
attr(,"class")
[1] "omit"
Levels: L1 < L2 < L3 < M1 < M2
```

Pour connaitre la longueur d'une variable, on utilise l'instruction `length`

```
> ### Nombre d'observations d'une variable
> length(na.omit(data$niveau))
```

```
[1] 39
```

Le rang de la médiane est l'observation de rang $\frac{n+1}{2}$. Si le nombre d'individu est pair, nous arrondissons à l'inférieur grâce a `round` :

```
> ### Rang de la mediane
> round( (length(na.omit(data$niveau))+1)/2 )
```

```
[1] 20
```

Ordonner une variable se fait grâce à `sort` :

```
> ### Ordonner une variable
> sort(data$niveau)
```

```
 [1] L1 L1 L2 L2 L2 L2 L2 L3 L3 L3 L3 L3 L3 L3 L3 L3 L3 L3 L3 L3 L3 L3 L3
[24] L3 L3 L3 L3 L3 L3 L3 L3 L3 L3 L3 L3 M1 M2 M2 M2
Levels: L1 < L2 < L3 < M1 < M2
```

Il ne nous reste plus qu'à combiner les deux, sélectionner l'observation dont on a calculé le rang dans la variable classée [2] :

```
> ### Calcule de la mediane
> sort(data$niveau)[round( (length(na.omit(data$niveau))+1)/2 )]
```

```
[1] L3
Levels: L1 < L2 < L3 < M1 < M2
```

2. En pratique, les autres étapes n'étaient que pédagogiques : cette seule instruction suffit à calculer la médiane.

Médiane d'une ordonnée, deuxième version :

Une autre option consiste a transformer notre variable ordonnée en `numeric` puis calculer la médiane de cette variable et conclure grâce aux `levels` de la variable :

```
> ### Converstion en numeric
> as.numeric(data$niveau)
```

```
 [1]  3  3  3  5  5  3  2  3  3  3  3  1  3  2  3 NA  2  3  3  3  1  2  3
[24]  3  3  3  3  3  3  3  4  3  3  3  3  3  3  3  2  5
```

```
> ### Calcul de la mediane
> median(as.numeric(data$niveau),na.rm=TRUE)
```

```
[1] 3
```

```
> ### Affichage des levels :
> levels(data$niveau)
```

```
[1] "L1" "L2" "L3" "M1" "M2"
```

```
> ### Combinaison des deux
> levels(data$niveau)[median(as.numeric(data$niveau),na.rm=TRUE)]
```

```
[1] "L3"
```

3.2.3 Moyenne

Le calcul de la moyenne se fait grâce à `mean`. La encore, il faut exclure les manquantes avec `na.rm=TRUE`

```
> ### Calcul de la moyenne
> mean(data$age,na.rm=TRUE)
```

```
[1] 21.45946
```

3.3 Dispersion

3.3.1 Quartiles

Pour une variable `numeric`, les quartiles se calculent à l'aide de la fonction `quantile` avec l'option `na.rm=TRUE` s'il y a des manquantes :

```
> ### Calcul des quartiles
> quantile(data$age,na.rm=TRUE)
```

```
  0%  25%  50%  75% 100%
  18   20   21   22   34
```

Pour une variable ordonnée, la méthode est la même que pour la médiane. On peut donc calculer à la main les rangs du premier et troisième (respectivement $\frac{n+3}{4}$ et $\frac{3n+1}{4}$) :

```
> ### Premier quartile (Q1)
> rangQ1 <- round( (length(na.omit(data$niveau))+3)/4 )
> sort(data$niveau)[rangQ1]
```

```
[1] L3
Levels: L1 < L2 < L3 < M1 < M2
```

```
> ### Troisième quartile (Q3)
> rangQ3 <- round( (3*length(na.omit(data$niveau))+1)/4 )
> sort(data$niveau)[rangQ3]
```

```
[1] L3
Levels: L1 < L2 < L3 < M1 < M2
```

On peut également tranformer la variable en `numeric`, puis utiliser les `levels` :

```
> ### Calcul de tous les quartiles
> levels(data$niveau)[quantile(as.numeric(data$niveau),na.rm=TRUE)]
```

```
[1] "L1" "L3" "L3" "L3" "M2"
```

3.3.2 Écart type et variance

L'écart type et la variance se calculent respectivement à l'aide de `sd` et `var`, avec l'option `na.rm=TRUE` pour supprimer les manquantes :

```
> ### Ecart type
> sd(data$age,na.rm=TRUE)
```

```
[1] 2.683226
```

```
> ### Variance
> var(data$age,na.rm=TRUE)
```

```
[1] 7.1997
```

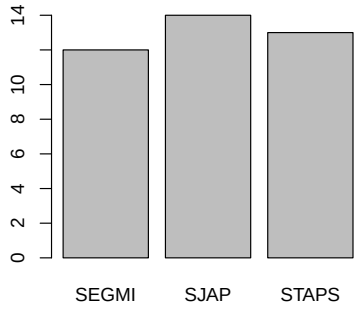
3.4 Représentation graphique

R dispose d'un grand nombre d'outils graphiques permettant de représenter des données. Là encore, la représentation graphique dépend du type de variable.

3.4.1 Diagramme en baton

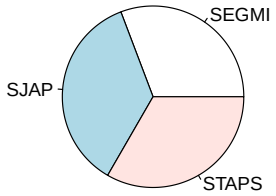
Pour les variables pour lesquelles il est possible de calculer les effectifs, on peut tracer un diagramme en baton :

```
> barplot(table(data$UFR))
```



Il est également possible de tracer des camemberts, mais cette représentation graphique est fortement déconseillée, l'oeil humain n'étant en effet pas adapté à l'évaluation des distances angulaires. Néanmoins, cela peut se faire avec `pie` (fortement déconseillé).

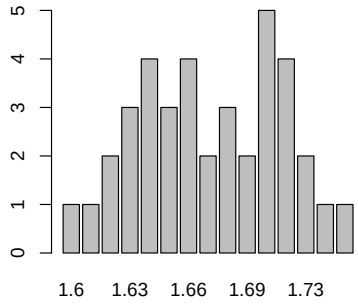
```
> pie(table(data$UFR))
```



3.4.2 Histogramme

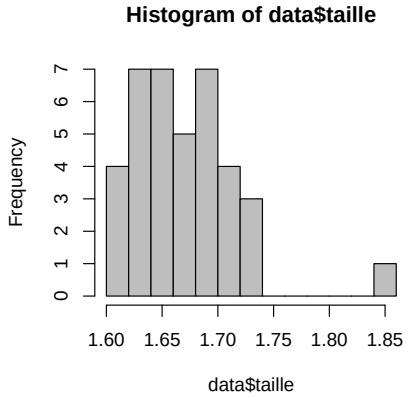
Un diagramme en baton est une représentation graphique adaptée aux variables ayant peu de modalités. Si on l'utilise sur une variable continue, on obtient un graphique peut informatif et supprimant les modalités manquantes :

```
> barplot(table(data$taille))
```



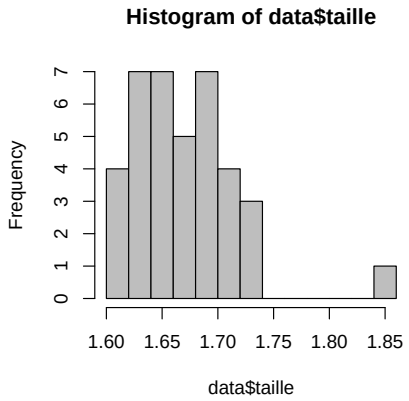
Il est donc plus intéressant de “regrouper” les modalités proches et de présenter graphiquement les regroupements. C’est ce qu’on appelle un histogramme :

```
> hist(data$taille,col="grey")
```



L’option `col="grey"` permet de préciser une couleur de coloriage. On peut également choisir d’augmenter ou de diminuer le nombre de colonne grâce à l’option `breaks` :

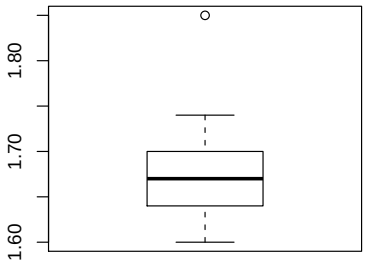
```
> hist(data$taille,col="grey",breaks=10)
```



3.4.3 Boîte à moustaches

Les boîtes à moustaches sont des représentations graphiques utilisable pour des variables numériques. La “boîte” est délimitée par le premier et le troisième quartiles, elle contient donc 50% de la population. Les moustaches encadrent les individus “proches” du centre. Au-delà des moustaches, on trouve soit les valeurs abérrantes (erreur de saisie), soit les valeurs éloignées du centre (valeurs extrèmes).

```
> boxplot(data$taille)
```



3.4.4 Export d'un graphique

R permet de sauvegarder les graphiques sous plusieurs format. Pour cela, il suffit de cliquer sur le graphique (bouton gauche) puis d'aller dans *Fichier* → *Sauver sous*. Il est également possible de faire directement un *Copier-Coller* vers un autre document. Pour cela, cliquez sur le graphique (bouton droit) puis selectionnez *Copier comme bitmap*. Vous pouvez ensuite faire un *Coller* sous Open Office ou sous Word.

Chapitre 4

Analyse bivariée

4.1 Effectifs, centralité et dispersion

Les effectifs s'obtiennent avec l'instruction `table` à laquelle on doit maintenant fournir les deux variables au lieu d'une seule. Comme pour l'analyse univariée, parler d'effectif n'a pas vraiment de sens avec les variables continues ; seules les nominales, ordonnées et discrètes sont concernées.

```
> ### Effectifs croisés de sexe & UFR
> table(data$sexe,data$UFR)
```

```
   SEGMI SJAP STAPS
F      6    9      7
H      5    5      5
```

Les indices de centralité et dispersion n'existent pas en bivariée. Par contre, il est possible de les calculer *relativement* a une autre variable. Par exemple, si on considère les variables `[sexe]` et `[age]`, il est possible de calculer les moyennes / variances de l'âge des hommes, puis des femmes. Cela se fait en selectionnant une partie de la colonne au lieu de la considérer dans son intégralité, puis en appliquant l'indice désiré. Plus précisément, nous avons vu section 2.3 page 11 qu'il était possible de selectionner seulement une ligne dans un `data.frame`. Pour mémoire :

```
> ### Selection de la deuxième valeur
> data$rapportAge[2]
```

```
[1] 18
```

Il est également possible d'en selectionner plusieurs :

```
> ### Selection des lignes 2 et 4
> data$rapportAge[c(2,4)]
```

```
[1] 18 17
```

Enfin, il est possible de sélectionner toutes les lignes vérifiant une certaine condition. Dans notre cas, nous voulons toutes les lignes pour lesquelle `sexe` prend la valeur `H`

```
> ### Selection des hommes
> data$sexe=="H"
```

```
 [1] FALSE FALSE    NA FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[12] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE    NA
[23] FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
[34]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
```

```
> ### Variable age du premier rapport
> data$rapportAge
```

```
 [1] 19 18 15 17 21 19 NA NA 16 20 19 NA 14 15 18 16 17 14 15 NA 16 18 19
[24] NA 20 17 17 14 17 14 17 18 NA 15 18 18 15 16 16 19
```

```
> ### Selection de la variable premier rapport pour les hommes
> data$rapportAge[data$sexe=="H"]
```

```
 [1] NA NA 17 17 14 17 14 17 18 NA 15 18 18 15 16 16 19
```

```
> ### Moyenne d'age du premier rapport homme
> mean(data$rapportAge[data$sexe=="H"],na.rm=TRUE)
```

```
[1] 16.5
```

Le calcul de la moyenne des femmes et des écarts types hommes / femmes répond au même principe :

```
> ### Moyenne d'age du premier rapport femme
> sd(data$rapportAge[data$sexe=="F"],na.rm=TRUE)
```

```
[1] 2.145827
```

```
> ### Ecart type des hommes
> mean(data$rapportAge[data$sexe=="H"],na.rm=TRUE)
```

```
[1] 16.5
```

```
> ### Ecart type des femmes
> sd(data$rapportAge[data$sexe=="F"],na.rm=TRUE)
```

```
[1] 2.145827
```

4.2 Représentation graphique bivariée

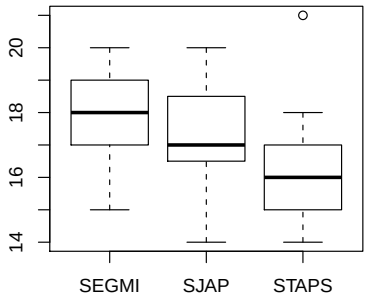
4.2.1 Deux qualitatives

Il n'existe pas vraiment de représentation graphique canonique pour deux variables qualitatives.

4.2.2 Qualitative & numérique

Pour une qualitative et une numérique, il est intéressant de graphiquement représenter des boites à moustache côte à côte, une pour chaque modalité de la qualitative :

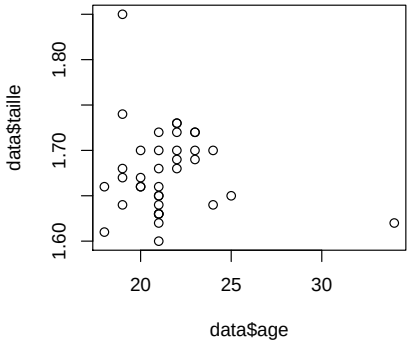
```
> ### Premiere relation selon les UFR
> boxplot(data$rapportAge~data$UFR)
```



4.2.3 Deux numériques

Pour deux numériques, on peut tracer un nuage de points :

```
> ### Age et taille
> plot(data$age,data$taille)
```



4.3 Tests

Tester, c'est répondre à la question : "y a-t-il un lien entre mes deux variables ?". Pour répondre à cette question, il existe deux types de tests. Les tests paramétriques sont des tests puissants mais ils nécessitent que les variables aient certaines propriétés. Les tests non-paramétriques sont moins puissants, mais n'imposent pas de condition d'application. C'est un peu comme une Ferrari et une Range rover : la première est très rapide mais uniquement sur route. La deuxième est moins rapide, mais elle passe partout...

Le choix d'un test se fait donc en deux étapes :

1. Le type des variables restreint le choix à un test paramétrique ou un test non paramétrique ;
2. Les propriétés des variables permettent de choisir entre le paramétrique et le non paramétrique.

4.3.1 Qualitative & Qualitative

Pour deux variables qualitatives, le test à utiliser est le test du χ^2 (paramétrique) ou le test exact de Fisher (non paramétrique). La condition nécessaire pour pouvoir utiliser

le test du χ^2 est la suivante :

1. les valeurs de toutes les cases du tableau des effectifs attendus doivent être suppérieures à 5.

Le tableau des effectifs attendus est un des tableaux construits quand on fait un χ^2 à la main. En pratique, nous n'aurons pas besoin de calculer le tableau des effectifs attendus, R le fera pour nous. Il vérifiera au passage si toutes les cases sont bien suppérieures à 5. Si ce n'est pas le cas, il nous le signalera sous la forme d'un `warning`.

```
> ### Lien entre sexe et transAvecPres
> chisq.test(data$sexe,data$transAvecPres)
```

```
        Pearson's Chi-squared test with Yates' continuity correction

data:  data$sexe and data$transAvecPres
X-squared = 4.4941, df = 1, p-value = 0.03401
```

Il n'y a pas de warning, cela veut dire que le test du χ^2 est valide. Avec un petit p de 0.03, on peut conclure qu'il y a un lien entre les variables. Un bref examen du tableau des effectifs nous donne le sens du lien :

```
> ### Table croisé
> table(data$sexe,data$transAvecPres)
```

```
    Non Oui
  F  15   5
  H   5  10
```

Il y a un lien entre le sexe et le fait de penser qu'on peut contracter le SIDA même en utilisant un préservatif ; les femmes ont une plus grande confiance dans les préservatifs.

Si test un lien entre les variables `sexe` et `rapportSexuel`, on obtient :

```
> ### Lien entre sexe et rapportRisque
> chisq.test(data$sexe,data$rapportRisque)
```

```
        Pearson's Chi-squared test with Yates' continuity correction

data:  data$sexe and data$rapportRisque
X-squared = 0.0424, df = 1, p-value = 0.8368

Warning message:
In chisq.test(data$sexe, data$rapportRisque) :
  l'approximation du Chi-2 est peut-être incorrecte
```

Ce warning nous indique que le test du χ^2 n'est pas valable dans le cas présent et nous devons utiliser le *test exact de Fisher*. La syntaxe est exactement la même :

```
> fisher.test(data$sexe,data$rapportRisque)
```

```
        Fisher's Exact Test for Count Data

data:  data$sexe and data$rapportRisque
p-value = 0.6207
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
 0.00734162 6.14218525
sample estimates:
odds ratio
 0.4276964
```

Avec un petit p de 0.84, on peut conclure :

Il n'y a pas de lien entre le sexe et la prise de risque : les hommes et les femmes se comportent de la même manière vis-à-vis du risque.

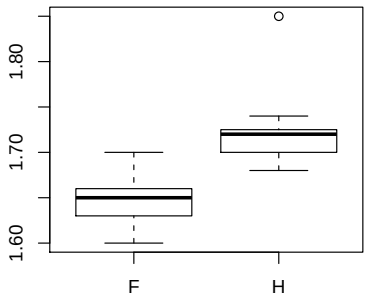
4.3.2 Qualitative (2 classes) & Numérique

La variable qualitative définit deux groupes sur lesquels on a effectué une mesure continue. Les tests possibles sont le *T de Student* (paramétrique) et le *test des rangs de Wilcoxon* (non paramétrique). Les conditions d'application sont :

1. Les écart types des deux groupes sont égaux
2. Pour chaque groupe, la variable numérique suit une loi normale OU les effectifs sont supérieurs à 30.

La vérification de l'égalité des variances peut se faire à la main (comme précisée section 4.1). On peut également tracer des boites à moustaches :

```
> ### Boites à moustaches
> boxplot(data$taille~data$sexe)
```



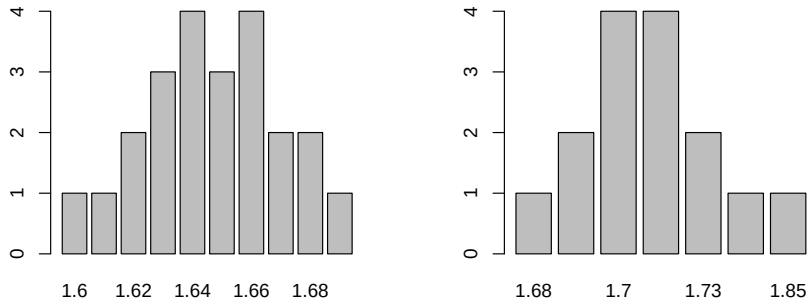
On peut également utiliser le *F de Fisher* pour comparaison des variances (à ne pas confondre avec le *test exact de Fisher*).

```
> ### Comparaison des variances des ages selon le sexe
> summary(aov(data$age~data$sexe))
```

```
             Df  Sum Sq Mean Sq F value Pr(>F)
data$sexe     1   1.892   1.892  0.2574 0.6151
Residuals    35 257.297   7.351
3 observations deleted due to missingness
```

Le petit p étant élevé, les variances peuvent être considérées comme égales. Reste à vérifier la distribution. Notre population étant de petite taille, il est plus pertinent de tracer des `barplot` que des histogrammes :

```
> ### Permet de placer 2 graphiques cote à cote
> par(mfrow=c(1,2))
> ### Trace les histogramme de age selon les sexes
> barplot(table(data$taille[data$sexe=="F"]))
> barplot(table(data$taille[data$sexe=="H"]))
```



Les deux variables suivent une loi normale, on peut donc utiliser le T de Student.

```
> ### T de Student
> t.test(data$taille~data$sexe,var.equal=TRUE)
```

```
        Two Sample t-test

data:  data$taille by data$sexe
t = -6.9189, df = 36, p-value = 4.195e-08
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.09302990 -0.05085416
sample estimates:
mean in group F mean in group H
       1.647391        1.719333
```

Dans le cas où les variances ne sont pas égales MAIS où on a tout de même la normalité, il est possible d'utiliser un T de Student *adapté* aux variances non égales. Il faut alors préciser que les variances sont différentes en utilisant l'option `var.equal=FALSE` :

```
> ### T de Student pour variances différentes
> t.test(data$taille~data$sexe,var.equal=FALSE)
```

```
        Welch Two Sample t-test

data:  data$taille by data$sexe
t = -6.2501, df = 20.682, p-value = 3.608e-06
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.09590200 -0.04798206
sample estimates:
mean in group F mean in group H
       1.647391        1.719333
```

Dans notre cas, on peut conclure :

La taille des garçons est significativement plus grande que celle des filles.

Si on souhaite tester le lien entre l'âge et la prise de risque, on constate que le groupe ayant pris des risques est tellement petit (4 personnes) qu'il ne peut pas suivre une loi normale. On doit donc appliquer un test des rangs de Wilcoxon :

```
> wilcox.test(data$age~data$rapportRisque)
```

```
        Wilcoxon rank sum test with continuity correction

data:  data$age by data$rapportRisque
W = 31, p-value = 0.2004
alternative hypothesis: true location shift is not equal to 0
```

Avec un petit p a 0.20, on peut conclure :

Il n'y a pas de lien entre la prise de risque et l'âge.

4.3.3 Qualitative (3 classes et plus) & Numérique

La variable qualitative définit plusieurs groupes sur lesquels on a effectué une mesure continue. Les tests possibles sont l'*Analyse de variance* ou *ANOVA* qui se conclut par un *F de Fisher* (paramétrique) et le *test de Kruskal Wallis* (non paramétrique). Les conditions d'application sont les mêmes que pour le T de Student.

1. Les écart types des deux groupes sont égaux
2. Pour chaque groupe, la variable numérique suit une loi normale OU les effectifs sont supérieurs à 30.

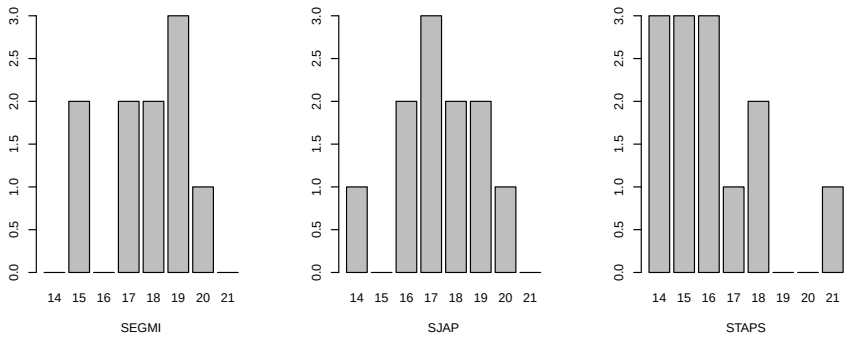
Pour comparer l'âge du premier rapport selon les UFR :

```
> ### Age du premier rapport selon les UFR
> summary(aov(data$rapportAge~data$UFR))
```

```
             Df  Sum Sq Mean Sq F value  Pr(>F)
data$UFR      2  17.402   8.701  2.6044 0.09005 .
Residuals    31 103.569   3.341
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
6 observations deleted due to missingness
```

Le petit p étant de 0.10, il serait tentant de conclure. Cela nous est formellement interdit puisque nous avons oublié (honte à nous !) de vérifier la normalité de la variable `rapportAge`...

```
> ### Normalité de rapportAge selon les UFR
> par(mfrow=c(1,3))
> barplot(table(c(data$rapportAge[data$UFR=="SEGMI"],14:21))-1,xlab="SEGMI")
> barplot(table(c(data$rapportAge[data$UFR=="SJAP"],14:21))-1,xlab="SJAP")
> barplot(table(c(data$rapportAge[data$UFR=="STAPS"],14:21))-1,xlab="STAPS")
```



La normalité n'est vraiment pas respectée ! Il est donc nécessaire d'utiliser un test non paramétrique :

```
> ### Age du premier rapport selon les UFR
> kruskal.test(data$rapportAge~data$UFR)
```

```
        Kruskal-Wallis rank sum test

data:  data$rapportAge by data$UFR
Kruskal-Wallis chi-squared = 5.5283, df = 2, p-value = 0.06303
```

Avec un petit p de 0.06, on peut conclure :

Il n'y a pas de lien entre l'âge de la première relation sexuelle et l'appartenance à un UFR.

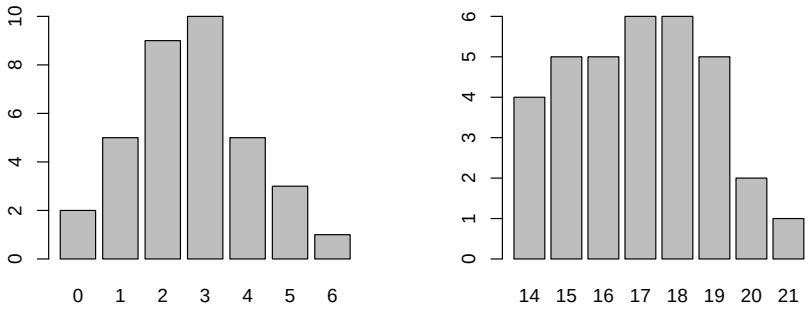
4.3.4 Numérique & Numérique

Les tests possibles sont la *corrélation de Pearson* (paramétrique) et la *corrélation de Spearman* (non paramétrique). La condition d'application est :

1. Au moins une des deux variables doit suivre une loi normale.

On veut étudier le score de connaissance du SIDA et l'âge du premier rapport :

```
> ### Vérification de la normalité
> par(mfrow=c(1,2))
> barplot(table(data$scoreConnaissance))
> barplot(table(c(data$rapportAge,14:21))-1)
```



La variable `scoreConnaissance` suit une loi normale, on peut donc utiliser le R de Pearson :

```
> ### Correlation de Pearson
> cor.test(data$scoreConnaissance,data$rapportAge)
```

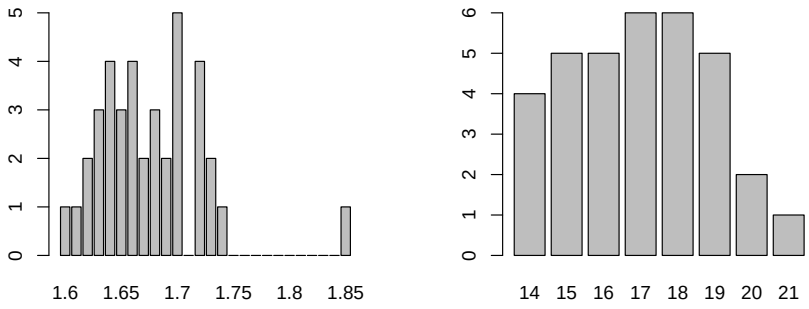
```
        Pearson's product-moment correlation

data:  data$scoreConnaissance and data$rapportAge
t = 1.2473, df = 28, p-value = 0.2226
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.1426373  0.5446716
sample estimates:
      cor
0.2294214
```

Le petit p étant de 0.22,

Il n'y a pas de lien entre l'âge de la première relation et la connaissance du SIDA

On veut maintenant étudier la taille et l'âge du premier rapport :

```
> ### Vérification de la normalité
> par(mfrow=c(1,2))
> barplot(table(c(data$taille,(160:185)/100))-1)
> barplot(table(c(data$rapportAge,14:21))-1)
```



Aucune ne suit une loi normale, on doit donc utiliser le R de Spearman :

```
> ### Correlation de Spearman
> cor.test(data$scoreConnaissance,data$rapportAge,method="spearman")
```

```
        Spearman's rank correlation rho

data:  data$scoreConnaissance and data$rapportAge
S = 3383.521, p-value = 0.1877
alternative hypothesis: true rho is not equal to 0
sample estimates:
    rho
0.24727
```

Le petit p étant de 0.18,

Il n'y a pas de lien entre l'âge de la première relation et la taille.