Séance 2 : Statistiques fondamentales

Introduction à la sociologie quantitative, niveau 2

Samuel Coavoux

1 Statistique univariée

2 Statistique bivariée et test d'hypothèse

Section 1

Statistique univariée

Sous-section 1

Décrire une variable quantitative

Somme

```
x <- runif(50, min = 0, max = 10)
x
```

```
##  [1] 2.2545920 0.9434869 8.3604689 0.3289177
##  [5] 4.2318615 2.0570059 4.7544763 0.1795488
##  [9] 9.2106140 7.1371017 0.7814955 9.1995658
## [13] 6.8540844 8.6728645 3.0234849 0.3241510
## [17] 8.3072564 3.6477874 5.9848938 7.0177142
## [21] 4.4695677 5.4321462 5.6392160 7.5271377
## [25] 7.1139194 4.7879820 4.9230179 0.4445459
## [29] 8.8137699 2.9129888 5.7464054 7.8061068
## [33] 9.5196606 7.8030906 0.7704786 5.9857575
## [37] 3.8492693 6.7775555 9.3873744 7.0325398
## [41] 2.7710910 2.5091991 9.7728695 3.2026599
## [45] 8.5998032 7.0366410 0.3302319 8.6122161
## [49] 0.8952351 1.3120692
```

```
sum(x)
```

Tendance centrale

```
# Moyenne
mean(x)
```

```
## [1] 5.101118
```

```
# enlever les 5% de valeurs les plus hautes
# et les 5% de valeurs les plus basses
mean(x, trim=.05)
```

```
## [1] 5.114341
```

```
# Médiane
median(x)
```

```
## [1] 5.535681
```

Extrêmes

```
min(x)
```

```
## [1] 0.1795488
```

```
max(x)
```

```
## [1] 9.772869
```

```
# Nombre de valeurs supérieures à
# une valeur fixée
sum(x > max(x)/2)
```

```
## [1] 27
```

Dispersion

```
var(x)
```

```
## [1] 9.425417
```

```
sd(x)
```

```
## [1] 3.070084
```

Dispersion : quantiles

```
# Quartiles, min et max
quantile(x)
```

```
##        0%       25%       50%       75%      100%
## 0.1795488 2.5746721 5.5356811 7.7341024 9.7728695
```

```
# 1er et 9e déciles
quantile(x, c(.1, .9))
```

```
##       10%       90%
## 0.7378853 8.8523495
```

Résumé

```
summary(x)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.1795  2.5747  5.5357  5.1011  7.7341  9.7729
```

Arrondis

```
round(x, 2)
```

```
##  [1] 2.25 0.94 8.36 0.33 4.23 2.06 4.75 0.18 9.21
## [10] 7.14 0.78 9.20 6.85 8.67 3.02 0.32 8.31 3.65
## [19] 5.98 7.02 4.47 5.43 5.64 7.53 7.11 4.79 4.92
## [28] 0.44 8.81 2.91 5.75 7.81 9.52 7.80 0.77 5.99
## [37] 3.85 6.78 9.39 7.03 2.77 2.51 9.77 3.20 8.60
## [46] 7.04 0.33 8.61 0.90 1.31
```

```
round(mean(x), 2)
```

```
## [1] 5.1
```

Intervalle de confiance autour d'une moyenne

```
tt <- t.test(x)
print(tt)
```

```
##
##  One Sample t-test
##
## data :  x
## t = 11.749, df = 49, p-value = 7.33e-16
## alternative hypothesis : true mean is not equal to 0
## 95 percent confidence interval :
##  4.228610 5.973627
## sample estimates :
## mean of x
##  5.101118
```

Intervalle de confiance autour d'une moyenne

```
str(tt)
```

```
## List of 9
##  $ statistic  : Named num 11.7
##   ..- attr(*, "names")= chr "t"
##  $ parameter  : Named num 49
##   ..- attr(*, "names")= chr "df"
##  $ p.value    : num 7.33e-16
##  $ conf.int   : num [1 :2] 4.23 5.97
##   ..- attr(*, "conf.level")= num 0.95
##  $ estimate   : Named num 5.1
##   ..- attr(*, "names")= chr "mean of x"
##  $ null.value : Named num 0
##   ..- attr(*, "names")= chr "mean"
##  $ alternative : chr "two.sided"
##  $ method      : chr "One Sample t-test"
##  $ data.name  : chr "x"
##  - attr(*, "class")= chr "htest"
```

Intervalle de confiance autour d'une moyenne

```
## Pour accéder au seul intervalle de confiance
tt$conf.int
```

```
## [1] 4.228610 5.973627
## attr(,"conf.level")
## [1] 0.95
```

```
## Ou directement
t.test(x)$conf.int
```

```
## [1] 4.228610 5.973627
## attr(,"conf.level")
## [1] 0.95
```

Intervalle de confiance autour d'une moyenne

Il est possible d'ajuster le niveau de confiance souhaité. Par défaut, on fixe α à 0.05 avec l'argument `conf.level = 0.95` ; il est possible de réduire α en augmentant `conf.level`.

```
t.test(x, conf.level = .99)$conf.int
```

```
## [1] 3.937549 6.264688
## attr(,"conf.level")
## [1] 0.99
```

Gestion des valeurs manquantes

La grande majorité de ces fonctions renvoient `NA` s'il y a une valeur manquante dans le vecteur. On peut alors préciser `na.rm = TRUE` pour que les valeurs manquantes soient supprimées avant le calcul.

Sous-section 2

Décrire une variable catégorielle

Données : Artist Community Survey

```
load("./data/ACS_artists.Rdata")
```

Tri à plat

```
table(dt$sexe)
```

```
##
## Female   Male
##  61185  62838
```

Tri à plat

```
table(dt$sexe)/nrow(dt)
```

```
##
##    Female      Male
## 0.4933359 0.5066641
```

```
prop.table(table(dt$sexe))
```

```
##
##    Female      Male
## 0.4933359 0.5066641
```

Tri à plat

```
library(questionr)
freq(dt$sexe)
```

```
##             n    % val%
## Female  61185 49.3 49.3
## Male    62838 50.7 50.7
```

Arguments de `freq()`

- `digits`: nombre de chiffres après la virgule (1 par défaut)
- `cum`: calculer les pourcentages cumulés (FALSE par défaut)
- `sort`: trier les modalités ("inc" : dans l'ordre croissant, "dec" : dans l'ordre décroissant ; défaut : ordre normal des modalités)

```
freq(dt$eng, cum = TRUE)
```

```
##                      n     %  val%  %cum  val%cum
## (1) Very well    12857  10.4  71.3  10.4     71.3
## (2) Well          3401   2.7  18.9  13.1     90.2
## (3) Not well      1453   1.2   8.1  14.3     98.2
## (4) Not at all     319   0.3   1.8  14.5    100.0
## NA              105993  85.5    NA 100.0       NA
```

Test de proportion

```
pt <- prop.test(x = sum(dt$sexe == "Female"),
                n = nrow(dt),
                conf.level = .99)
pt$conf.int
```

```
## [1] 0.4896756 0.4969970
## attr(,"conf.level")
## [1] 0.99
```

Section 2

Statistique bivariée et test d'hypothèse

Sous-section 1

Deux variables quantitatives

Analyse de corrélation

```
var(dt$age)
```

```
## [1] 229.5101
```

```
var(dt$income)
```

```
## [1] 3326444615
```

```
cov(dt$age, dt$income)
```

```
## [1] 128716.8
```

```
cor(dt$age, dt$income)
```

```
## [1] 0.1473141
```

Sous-section 2

Deux variables qualitatives

Tableau croisé

```
tb <- table(dt$dipl_c, dt$sexe)
tb
```

```
##
##                 Female  Male
##   Aucun           2132  2244
##   HS degree      18624 20223
##   College        31177 30324
##   Graduate ed.    9252 10047
```

Tableau croisé : structure

```
str(tb)
```

```
##  'table' int [1 :4, 1 :2] 2132 18624 31177 9252 2244 20223 30
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1 :4] "Aucun" "HS degree" "College" "Graduate ed
##   ..$ : chr [1 :2] "Female" "Male"
```

Tableau croisé : marges

```
addmargins(tb)
```

```
##
##                 Female   Male    Sum
##    Aucun          2132   2244   4376
##    HS degree     18624  20223  38847
##    College       31177  30324  61501
##    Graduate ed.   9252  10047  19299
##    Sum           61185  62838 124023
```

Tableau croisé : fréquences

```
# Fréquences du total ;
# Ajouter margin = 1 pour % ligne
# Ajouter margin = 2 pour % colonne
prop.table(tb)
```

```
##
##                    Female       Male
##    Aucun          0.01719036 0.01809342
##    HS degree      0.15016570 0.16305846
##    College        0.25138079 0.24450304
##    Graduate ed.   0.07459907 0.08100917
```

Tableau croisé : fréquences ligne

```
library(questionr)
lprop(tb)
```

```
##
##                Female Male  Total
##   Aucun          48.7   51.3 100.0
##   HS degree      47.9   52.1 100.0
##   College        50.7   49.3 100.0
##   Graduate ed.   47.9   52.1 100.0
##   Ensemble       49.3   50.7 100.0
```

Tableau croisé : fréquences colonne

```
cprop(tb)
```

```
##
##                Female Male  Ensemble
##   Aucun           3.5    3.6   3.5
##   HS degree      30.4   32.2  31.3
##   College        51.0   48.3  49.6
##   Graduate ed.   15.1   16.0  15.6
##   Total         100.0  100.0 100.0
```

Test du χ^2

```
ct <- chisq.test(tb)
ct
```

```
##
##  Pearson's Chi-squared test
##
## data :  tb
## X-squared = 91.248, df = 3, p-value <
## 2.2e-16
```

Test du χ^2: structure

```
str(ct)
```

```
## List of 9
##  $ statistic : Named num 91.2
##   ..- attr(*, "names")= chr "X-squared"
##  $ parameter : Named int 3
##   ..- attr(*, "names")= chr "df"
##  $ p.value   : num 1.18e-19
##  $ method    : chr "Pearson's Chi-squared test"
##  $ data.name : chr "tb"
##  $ observed : 'table' int [1 :4, 1 :2] 2132 18624 31177 9252
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1 :4] "Aucun" "HS degree" "College" "Graduate
##   .. ..$ : chr [1 :2] "Female" "Male"
##  $ expected : num [1 :4, 1 :2] 2159 19165 30341 9521 2217 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1 :4] "Aucun" "HS degree" "College" "Graduate
##   .. ..$ : chr [1 :2] "Female" "Male"
```

Test du χ^2: effectifs théoriques

```
ct$expected
```

```
##
##                    Female      Male
##    Aucun          2158.838  2217.162
##    HS degree     19164.620 19682.380
##    College       30340.652 31160.348
##    Graduate ed.   9520.890  9778.110
```

Test du χ^2: p value

```
ct$p.value
```

```
## [1] 1.181377e-19
```

Test de Fisher

```r
# On produit un tableau croisé imaginaire avec
# de très petits effectifs
tb <- as.table(matrix(c(10, 9, 4, 1), nrow = 2))
ft <- fisher.test(tb)
ft
```

```
##
##  Fisher's Exact Test for Count Data
##
## data :  tb
## p-value = 0.3577
## alternative hypothesis : true odds ratio is not equal to 1
## 95 percent confidence interval :
##  0.005091836 3.718776118
## sample estimates :
## odds ratio
##  0.2917113
```

Test de Fisher

```
str(ft)
```

```
## List of 7
##  $ p.value     : num 0.358
##  $ conf.int    : num [1 :2] 0.00509 3.71878
##   ..- attr(*, "conf.level")= num 0.95
##  $ estimate    : Named num 0.292
##   ..- attr(*, "names")= chr "odds ratio"
##  $ null.value : Named num 1
##   ..- attr(*, "names")= chr "odds ratio"
##  $ alternative : chr "two.sided"
##  $ method      : chr "Fisher's Exact Test for Count Data"
##  $ data.name   : chr "tb"
##  - attr(*, "class")= chr "htest"
```

```
ft$p.value
```

```
## [1] 0.3577075
```

Sous-section 3

Une variable quantitative, une variable qualitative

tapply()

La fonction `tapply()` permet d'appliquer une fonction à un vecteur en le découpant selon les modalités d'un deuxième vecteur. Par exemple, si l'on souhaite appliquer la fonction `mean()` au revenus des artistes en les différenciant par niveau d'éducation, on peut écrire :

```
# Les arguments sont dans le bon ordre
# Vous pouvez omettre les noms d'arguments
# qui sont rappelés parce que leur ordre
# n'est pas intuitif
tapply(X = dt$income, INDEX = dt$dipl_c, FUN = mean)
```

```
##         Aucun    HS degree      College
##      19307.13     31941.02     49261.19
## Graduate ed.
##      62959.98
```

`tapply()` renvoie par défaut un vecteur unidimensionnel si le résultat de la fonction est une valeur unique (un vecteur de taille 1) ; il renvoie une liste avec des objets dont le nom correspond aux modalités de la variable

Test de student

```
tt <- t.test(dt$income ~ dt$sexe)
tt
```

```
##
##  Welch Two Sample t-test
##
## data :  dt$income by dt$sexe
## t = -68.681, df = 109640, p-value < 2.2e-16
## alternative hypothesis : true difference in means is not equa
## 95 percent confidence interval :
##  -22599.59 -21345.50
## sample estimates :
## mean in group Female   mean in group Male
##             33778.15             55750.69
```

Test de student

On remarque que, par défaut, `t.test()` fait un test de Welch. On peut forcer un test de Student en ajoutant `var.equal = TRUE` (à condition évidemment que les variances soient bien égales).

```
tapply(dt$income, dt$sexe, var)
```

```
##     Female       Male
## 1955312594 4423377631
```

```
# Elles ne le sont pas...
# t.test(dt$income ~ dt$sexe, var.equal=TRUE)
```

Test de student

Par défaut, le test est bilatéral. On peut réaliser un test unilatéral avec `alternative` ("less" pour un test unilatéral à gauche, "greater" pour un test unilatéral à droite)

```
t.test(dt$income ~ dt$sexe, alternative = "less")
```

```
##
##  Welch Two Sample t-test
##
## data :  dt$income by dt$sexe
## t = -68.681, df = 109640, p-value < 2.2e-16
## alternative hypothesis : true difference in means is less tha
## 95 percent confidence interval :
##       -Inf -21446.31
## sample estimates :
## mean in group Female   mean in group Male
##             33778.15             55750.69
```

Test de student

```
tt$p.value
```

```
## [1] 0
```

Test de student

On peut enfin comparer une moyenne à une moyenne théorique μ.

```
t.test(dt$income, mu=45000)
```

```
##
##  One Sample t-test
##
## data :  dt$income
## t = -0.54436, df = 124020, p-value = 0.5862
## alternative hypothesis : true mean is not equal to 45000
## 95 percent confidence interval :
##  44589.86 45231.84
## sample estimates :
## mean of x
##  44910.85
```