Travail des Packages

GASMI Chaymae, RIDADARAJAT Zakaria, DAIF Hakim

Package KScorrect

KScorrect met en œuvre le test de Kolmogorov-Smirnov corrigé par Lilliefors pour une utilisation dans les tests de qualité d'ajustement, adapté lorsque les paramètres de population sont inconnus et doivent être estimés par des statistiques d'échantillon. Les P-values sont estimées par simulation. Codé pour compléter stats :: ks.test, il peut être utilisé avec une variété de distributions continues, y compris les distributions normales, lognormales, mélange de normales, uniformes, loguniformes, exponentielles, gamma et Weibull.

Des fonctions sont également fournies pour générer des nombres aléatoires et calculer des fonctions de densité, de distribution et de quantile pour les distributions de mélange loguniformes et normales.

```
#install.packages("devtools")
library(devtools)
```

```
install.packages("infer")
```

Les fonctions du Package KScorrect

The Log Uniform Distribution:

Densité, fonction de distribution, fonction de quantile et génération aléatoire pour la distribution log uniforme dans l'intervalle de min à max. Les paramètres doivent être des valeurs brutes (non transformées en log) et seront transformées en log en utilisant la base spécifiée.

Usage

- `dlunif(x, min, max, base = exp(1))` : donne la densité.
- `plunif(q, min, max, base = exp(1))` : donne la distribution de la fonction.
- `qlunif(p, min, max, base = exp(1))` : donne le quantile de la fonction.
- `rlunif(n, min, max, base = exp(1))` : génére des nombres aléatoires random.

Arguments

+`x`: Vecteur de quantiles.

+`q`:Vecteur de quantiles.

+`p`: Vecteur de probabilités

+`n`: Nombres d'observations.

+`min`: Limite inférieure de la distribution, en valeurs brutes (non transformées en log). Les valeurs négatives donneront un avertissement.

+`max`: Limite supérieure de la distribution, en valeurs brutes (non transformées en log). Les valeurs négatives donneront un a avertissement.

+`base`: La base sur laquelle les logarithmes sont calculés. La valeur par défaut est e = exp (1). Doit être un nombre positif.

Exemple d'applications

on va voir dans ce qui suit des exemples d'applications des fonctions du package(KScorrect):

```
plot(2:200, dlunif(2:200, exp(1), exp(20)), type="l", main="Loguniform density")
plot(log(2:200), dlunif(log(2:200), log(1), log(20)), type="l", main="Loguniform density")
plot(2:200, plunif(2:200, exp(1), exp(20)), type="l", main="Loguniform cumulative")
plot(qlunif(ppoints(200), exp(1), exp(20)), type="l", main="Loguniform quantile")

hist(rlunif(2000, exp(1), exp(20)), main="random loguniform sample")
hist(log(rlunif(20000, exp(1), exp(20))), main="random loguniform sample")
hist(log(rlunif(20000, exp(1), exp(20), base=10), base=10), main="random loguniform sample")
```

The Normal Mixture Distribution:

Densité, fonction de distribution, fonction quantile et génération aléatoire pour une distribution univariée (unidimensionnelle) composée d'un mélange de distributions normales avec des moyennes égales à la moyenne, des écarts-types égaux à sd et une proportion de mélange des composantes égale à pro.

Usage

- `dmixnorm(x,mean,sd,pro)`:donne la densité.
- `dmixnorm(q,mean,sd,pro)`:donne la distribution de la fonction.
- `dmixnorm(p,mean,sd,pro,expand=1)`:donne le quantile de la fonction.
- `dmixnorm(n,mean,sd,pro)`:génére des nombres aléatoires random.

Arguments

+`x`: Vecteur de quantiles.

+`mean`: Vecteur de moyennes, un pour chaque composant.

+`sd` :Vecteur d'écarts types, un pour chaque composant. Si une seule valeur est fournie, un modèle de mélange à variance égale est implémenté.Doit être non négatif.

+`pro`: Vecteur de proportions de mélange, un pour chaque composant. S'il manque, un modèle à proportion égale est implémenté, avec un avertissement. Si les proportions ne correspondent pas à Unity, elles sont redimensionnées pour ce faire. Doit être non négatif.

+`q`:Vecteur de quantiles.

+`p`: Vecteur de probabilités.

+`expand`:Valeur pour élargir la plage de probabilités pour l'approximation quantile. Valeur par défaut = 1,0.

+`n`: Nombres d'observations.

Exemple d'applications

on va voir dans ce qui suit des exemples d'applications des fonctions du package(KScorrect):

NB : on aura besoin du package mclust

```
mean <- c(2, 8)
pro <- c(.30, .100)
sd <- c(.9, 2)
x <- rmixnorm(n=7000, mean=mean, pro=pro, sd=sd)
hist(x, n=30, main="random bimodal sample")
plot(seq(0, 20, .1), dmixnorm(seq(0, 20, .1), mean=mean, sd=sd, pro=pro),
type="l", main="Normal mixture density")
plot(seq(0, 20, .1), pmixnorm(seq(0, 20, .1), mean=mean, sd=sd, pro=pro),
type="l", main="Normal mixture cumulative")
plot(stats::ppoints(200), qmixnorm(stats::ppoints(200), mean=mean, sd=sd, pro=pro), type="l", main="Nor
```

---

ks_test_stat:Internal KScorrect Function:

Fonction interne non destinée à être appelée directement par les utilisateurs.

+`ks.test`:fonction qui calcule uniquement la statistique du test bilatéral D.

Package Rstatix:

Le Package "Rstatix" fournit un cadre simple et intuitif, pour effectuer des tests statistiques de base, y compris le test de comparaison de moyenne, le test de Wilcoxon, l'ANOVA, Kruskal-Wallis et les analyses de corrélation.

La sortie de chaque test est automatiquement transformée en tidy data frame afin de faciliter la visualisation.

```
library(datarium)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggpubr)
library(rstatix)
```

```
##
## Attaching package: 'rstatix'
## The following object is masked from 'package:stats':
##
##     filter
```

Nous utiliserons le jeu de données sur l'estime de soi mesuré sur trois points temporels. Les données sont disponibles dans le package "datarium".

```r
# Préparation des données
# Format large
data("selfesteem", package = "datarium")
head(selfesteem, 6)
```

```
## # A tibble: 6 x 4
##      id    t1    t2    t3
##   <int> <dbl> <dbl> <dbl>
## 1     1  4.01  5.18  7.11
## 2     2  2.56  6.91  6.31
## 3     3  3.24  4.44  9.78
## 4     4  3.42  4.71  8.35
## 5     5  2.87  3.91  6.46
## 6     6  2.05  5.34  6.65
```

```r
# Rassembler les colonnes t1, t2 et t3 en format long
# Convertir l'identifiant et le temps en facteurs
selfesteem <- selfesteem %>%
  gather(key = "time", value = "score", t1, t2, t3) %>%
  convert_as_factor(id, time)
head(selfesteem, 3)
```

```
## # A tibble: 3 x 3
##   id    time  score
##   <fct> <fct> <dbl>
## 1 1     t1     4.01
## 2 2     t1     2.56
## 3 3     t1     3.24
```

Statistiques descriptives:

Calculons quelques statistiques sommaires du score d'estime de soi par groupe (temps) : moyenne et sd (écart-type) en utilisant la fonction "get_summary_stats" qui est l'une des fonctions du package "Rstatix".

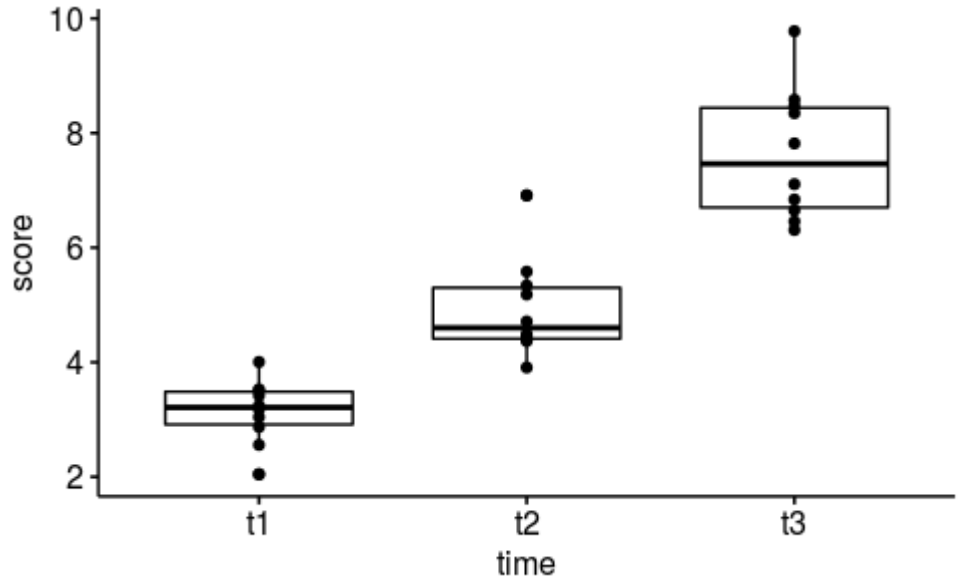
```r
selfesteem %>%
  group_by(time) %>%
  get_summary_stats(score, type = "mean_sd")
```

```
## # A tibble: 3 x 5
##   time  variable     n  mean    sd
##   <fct> <chr>    <dbl> <dbl> <dbl>
## 1 t1    score       10  3.14 0.552
## 2 t2    score       10  4.93 0.863
## 3 t3    score       10  7.64 1.14
```

Visualisation:

Nous allons créer un box plot et ajouter des points correspondant à des valeurs individuelles pour la variable "score d'estime de soi" selon les trois points temporels.

```r
bxp <- ggboxplot(selfesteem, x = "time", y = "score", add = "point")
bxp
```



Vérification des hypothèses

Valeurs aberrantes

Les valeurs aberrantes peuvent être facilement identifiées à l'aide de méthodes des boxplots, implémentées dans la fonction R identify_outliers()

```
selfesteem %>%
  group_by(time) %>%
  identify_outliers(score)
```

```
## # A tibble: 2 x 5
##   time  id    score is.outlier is.extreme
##   <fct> <fct> <dbl> <lgl>      <lgl>
## 1 t1    6      2.05 TRUE       FALSE
## 2 t2    2      6.91 TRUE       FALSE
```

on remarque qu'il n'y pas de valeurs aberrantes sur nos données [is.extreme=FALSE].

Hypothèse de normalité

L'hypothèse de normalité peut être vérifiée en calculant le test de Shapiro-Wilk pour chaque point dans le temps. Si les données sont normalement distribuées, la p-value doit être supérieure à 0,05.

```
selfesteem %>%
  group_by(time) %>%
  shapiro_test(score)
```

```
## # A tibble: 3 x 4
##   time  variable statistic     p
##   <fct> <chr>        <dbl> <dbl>
## 1 t1    score        0.967 0.859
## 2 t2    score        0.876 0.117
## 3 t3    score        0.923 0.380
```

Le score d'estime de soi est normalement distribué à chaque point dans le temps, tel qu'évalué par le test de Shapiro-Wilk ($p > 0,05$).

Hypothèse de sphéricité

L'hypothèse de sphéricité sera automatiquement vérifiée lors du calcul du test ANOVA en utilisant la fonction R anova_test() du package Rstatix. Le test de Mauchly est utilisé en interne pour évaluer l'hypothèse de sphéricité.

on utilise la fonction get_anova_table() pour extraire la table ANOVA, par ailleurs, la correction de sphéricité de Greenhouse-Geisser est automatiquement appliquée aux facteurs qui violent l'hypothèse de sphéricité.

Calcul Anova:

```
res.aov <- anova_test(data = selfesteem, dv = score, wid = id, within = time)
get_anova_table(res.aov)
```

```
## ANOVA Table (type III tests)
##
##   Effect DFn DFd      F        p p<.05   ges
## 1   time   2  18 55.469 2.01e-08     * 0.829
```

D'après les resultats du modèle Anova,le score de l'estime de soi était statistiquement significativement différent aux différents temps pendant le régime vu que la p-value est inférieure à 5%.

Package Infer:

L'objectif de ce package est d'effectuer des inférences statistiques en utilisant une grammaire statistique expressive qui est cohérente avec le tidyverse cadre de conception. Le package est centré autour de 4 verbes principaux, complétés par de nombreux utilitaires pour visualiser et extraire la valeur de leurs sorties.

- Specify() vous permet de spécifier la variable, ou la relation entre les variables, qui vous intéresse.
- hypothesize() vous permet de déclarer l'hypothèse nulle, l'endroit où nous sélectionnons l'hypothèse nulle.
- generate() vous permet de générer des données reflétant l'hypothèse nulle, crée des valeurs permutées.
- calculate() vous permet de calculer une distribution de statistiques à partir des données
- générées pour former la distribution nulle.
- visualize() trace automatiquement les valeurs permutées avec ggplot, ce qui facilite l'ajout de geoms dessus

Points forts:

- dataframe dans dataframe out
- composer des tests avec des tuyaux
- la lecture d'une chaîne inférentielle décrit une procédure inférentielle

Le package infer permet, dans un premier lieu, d'expliciter et illustrer le raisonnement sous-jacent aux tests statistiques (comme le t-test ou le test du chi-2).

Il permet en outre de s'abstraire (dans une certaine mesure) des problèmes de non-respect des hypothèses de ces tests en permettant de calculer la distribution des statistiques de test via des permutations plutôt qu'en s'appuyant sur les distributions théoriques des statistiques sous hypothèse de normalité, d'homoscedasticité, ou de taille d'échantillon ou effectifs "suffisants".

Package tidyverse:
Le package infer est particulièrement intéressant dans le cas d'une initiation à R par le tidyverse.

Tidyverse est une collection de packages R essentiels pour la science des données. Les packages sous le parapluie tidyverse nous aident à exécuter et à interagir avec les données. Il existe une multitude de choses que vous pouvez faire avec vos données, telles que le sous-ensemble, la transformation, la visualisation, etc.

Tidyverse a été créé par le grand Hadley Wickham et son équipe dans le but de fournir tous ces utilitaires pour nettoyer et travailler avec des données.

Certains des principaux packages de tidyverse sont:

Infer répond aux mêmes logiques de syntaxe et de programmation (avec des commandes "pipables" par exemple). Il est aussi également particulièrement intéressant si vous souhaitez enseigner les statistiques à des gens qui ne savent pas ou ne sont pas encore à l'aise pour réaliser des simulations "à la main".

En effet, le package prend en charge la réalisation de nombreuses permutations des données qui permettent de calculer la distribution empirique d'une statistique sous hypothèse nulle (et donc le calcul d'une p-value) même si les hypothèses permettant de déduire la distribution théorique de la statistique ne sont pas réunies.

Les créateurs du package illustrent cette idée de la manière suivante:

#installation

```
#installation et charegment du package infer
#install.packages("infer")
library(infer)

#Pour des graphs meilleurs
#install.packages("cowplot")
library(cowplot)


#chargement de tidyverse
library(tidyverse)
```

Nous utiliserons l' gss ensemble de données fournies par infer, qui contient un échantillon de données de l'Enquête sociale générale. Voir gss pour plus d'informations sur les variables incluses et leur source. Notez que ces données (et nos exemples dessus) sont à des fins de démonstration uniquement et ne fourniront pas nécessairement des estimations précises à moins d'être correctement pondérées. Pour ces exemples, supposons que cet ensemble de données soit un échantillon représentatif d'une population que nous voulons connaître: les adultes américains. Les données ressemblent à ceci:

```
data (gss)
load(url("http://bit.ly/2E65g15"))

#afficher les données (colonnes, attributs)

names(gss)
```

```
##  [1] "id"       "year"     "age"      "class"    "degree"   "sex"
##  [7] "marital"  "race"     "region"   "partyid"  "happy"    "relig"
## [13] "cappun"   "finalter" "natspac"  "natarms"  "conclerg" "confed"
## [19] "conpress" "conjudge" "consci"   "conlegis" "zodiac"   "oversamp"
## [25] "postlife" "party"    "space"    "NASA"
```

```
#selectionner uniquement party et nasa
gss %>% select(party, NASA)
```

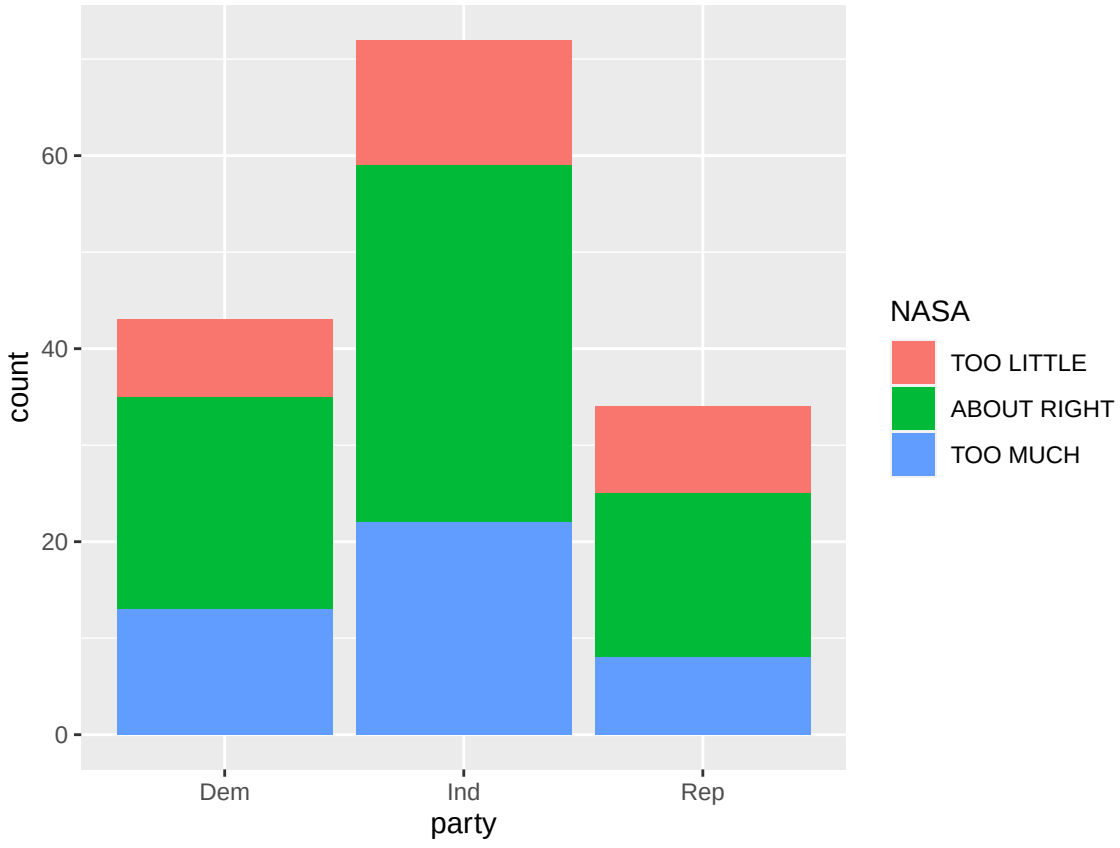
```
## # A tibble: 149 x 2
##    party NASA
##    <fct> <fct>
```

```
##  1 Ind   TOO LITTLE
##  2 Ind   ABOUT RIGHT
##  3 Dem   ABOUT RIGHT
##  4 Ind   TOO LITTLE
##  5 Ind   TOO MUCH
##  6 Ind   TOO LITTLE
##  7 Ind   ABOUT RIGHT
##  8 Dem   ABOUT RIGHT
##  9 Dem   TOO LITTLE
## 10 Ind   TOO LITTLE
## # ... with 139 more rows
```

EXEMPLE 1: EXISTE-IL UNE RELATION ENTRE PARTY(LES PARTIS POLITIQUES) ET LE DEGRE DE LEURS SOUTIENT A L'EXPLOITATION SPACIALE (NASA,TOO LITTLE, ABOUT RIGHT, TOO MUCH)

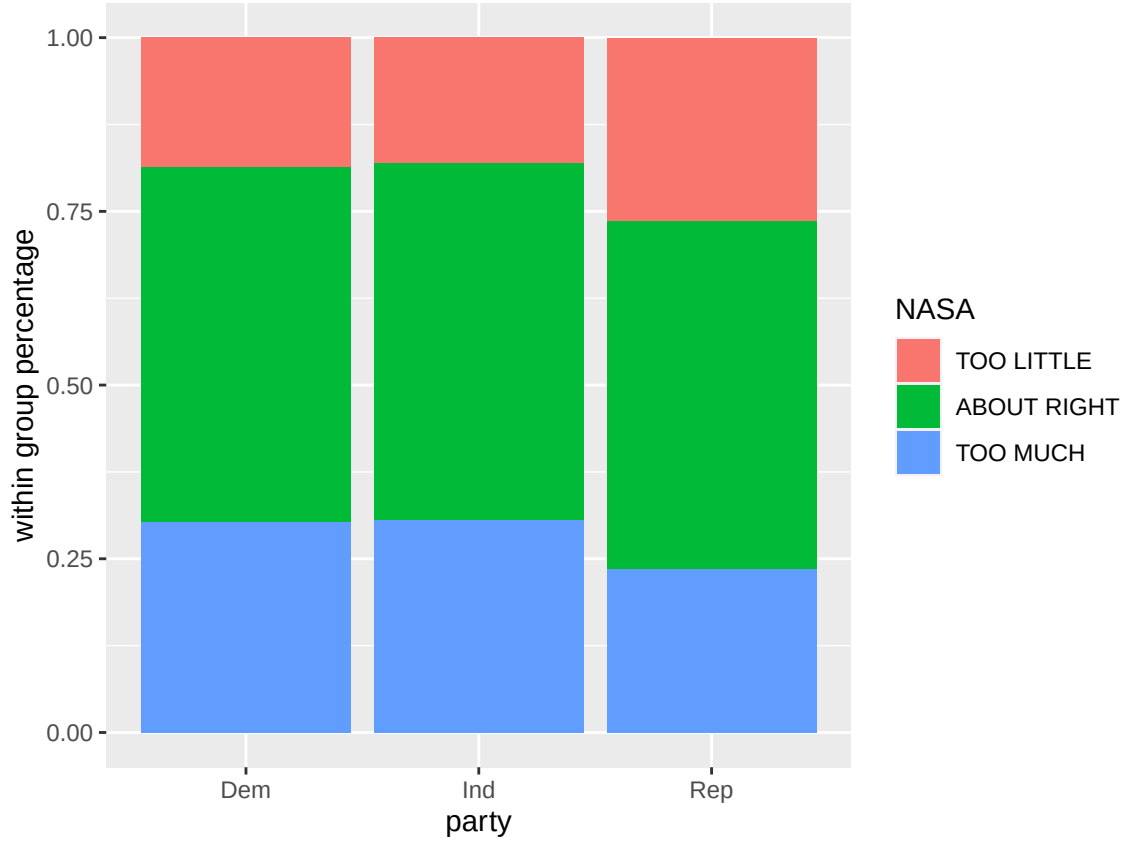
Question: Le financement de l'exploration spatiale (NASA) a-t-il une relation avec les partis politiques (les indépendants, les républicains, les démocrates) ? -Pensez-vous que le financement de l'exploration spatiale est too little, about right, too much?»

```
#interessant uniquement aux: party(les partis politiques) et le degré de leurs soutient à l'exploitatio

#visualisation en graph (geom bar)
gss %>%
  select(party, NASA) %>%
  ggplot(aes(x=party, fill = NASA)) +
  geom_bar()
```



Les données de comptage peuvent être trompeuses lorsque nous recherchons des tendances entre des variables catégorielles au sein de groupes, donc normalisons les pourcentages intra-groupes avec: position = “fill”argument dans geom_bar(). A travers ces commande on affiche un graph plus interprétable et plus signifiant

```
gss %>%
  select(party, NASA) %>%
  ggplot(aes(x=party, fill = NASA)) +
  geom_bar(position = "fill") +
  ylab("within group percentage")
```



Résultat 1 Il ne semble pas qu’il y ait beaucoup de différence dans la façon dont les démocrates, les indépendants et les républicains soutiennent l’exploration spatiale, mais allons maintenant approfondir cela avec quelques tests d’hypothèse, en comparant la base R et infer.

pour le moment et Ce que nous avons essentiellement, c’est juste un tableau de contingence des partis politiques (les indépendants, les républicains, les démocrates) et de l’attitude (too little, about right, too much) envers l’exploration spatiale (NASA), et nous voulons voir: # S’il existe une relation entre ces variables (problématique)

Solution: Le test d’indépendance du chi carré

Le test d’indépendance du Khi2 permet de déterminer si deux questions qualitatives son indépendantes ou non, ou autrement dit, si les réponses de l’une conditionnent les réponses de l’autre. Il ne permet toutefois pas de connaître le sens de la dépendance.

Avant d’effectuer ce test il est très important de vérifier et respecter ces hypothèses:

- Hypothèse n°1: les variables sont catégoriques. Il existe 3 types de données catégorielles: -Dichotomique : 2 groupes (par exemple - Homme et Femme) -Nominal : 3 groupes catégoriels ou plus (p. Ex. - premier cycle, professeur, étudiant diplômé, chercheur postdoctoral) -Ordinal : groupes ordonnés (par exemple - Niveau de douleur 1, Niveau de douleur 2, Niveau de douleur 3,. . .)
- Hypothèse n°2: Les observations sont indépendantes les unes des autres (par exemple, aucune relation entre les cas).
- Hypothèse n°3: Les regroupements de variables catégorielles doivent être mutuellement exclusifs. En d'autres termes, nous ne pouvons pas avoir un seul participant comme «démocrate» et «indépendant».
- Hypothèse n°4: Il doit y avoir au moins 5 fréquences attendues dans chaque groupe de votre variable catégorielle (seulement important pour la solution analytique) n>5.

Hypothèse nulle H0: il n'y a pas de relation entre le parti (démocrate, indépendant, républicain) et l'attitude envers l'exploration spatiale (too little, about right, too much).

Hypothèse alternative H1: il existe une relation entre le parti et l'attitude envers l'exploration spatiale.

seuil critique (alpha)= 5% une statistique chi-carré plus grande suggère des preuves plus solides pour le rejet de notre hypothèse nulle. Si nous observons une valeur p <= .05, nous rejetterions notre hypothèse nulle.

Que signifierait accepter notre hypothèse alternative H1? Dans le cas de notre exemple, si nous vivions dans un univers complètement aléatoire, moins de 5% de chance nous arriverions à la combinaison particulière de parti et d'attitude envers l'exploration spatiale que nous observons dans nos données. En d'autres termes, la relation entre le parti et l' attitude envers l'exploration spatiale que nous voyons dans nos données est significative.

Test de chi-2 sans le package R (solution classique en d'autres termes solution analytique)

```
chisq.test(gss$party, gss$NASA)
```

```
##
##  Pearson's Chi-squared test
##
## data:  gss$party and gss$NASA
## X-squared = 1.3, df = 4, p-value = 0.9
```

```
observed_stat <- chisq.test(gss$party, gss$NASA)$stat
```

Interpretation du résultat (test de chi-2, solution analytique sans le package infer)

On remarque que p-value est largement supérieure au seuil critique alpha de 5%,on accepte l'hypothèse alternative H1: il existe une relation entre le parti et l'attitude envers l'exploration spatiale Ou bien,la relation entre le parti et l' attitude envers l'exploration spatiale que nous voyons dans nos données est significative .

Interprétation du résultat (test de chi-2, solution analytique sans le package infer)

On remarque que p-value est largement supérieure au seuil critique alpha de 5%,on accepte l'hypothèse alternative H1: il existe une relation entre le parti et l'attitude envers l'exploration spatiale Ou bien, la relation entre le parti et l' attitude envers l'exploration spatiale que nous voyons dans nos données est significative .

Test de chi-2 en utilisant le package infer Une autre façon de tester s'il existe une relation significative dans nos données est d'adopter une approche programmatique en utilisant le package infer, cela nous permettra de faire face à la problématique illustrée dans un mode purement aléatoire, revenons au premier paragraphe de définition du package infer et son utilité:

Il permet en outre de s'abstraire (dans une certaine mesure) des problèmes de non-respect des hypothèses de ces tests en permettant de calculer la distribution des statistiques de test via des permutations plutôt qu'en s'appuyant sur les distributions théoriques des statistiques sous hypothèse de normalité, d'homoscédasticité, ou de taille d'échantillon ou effectifs"suffisants"

Permutation des données(prenons une des colonnes et brouillons-la)

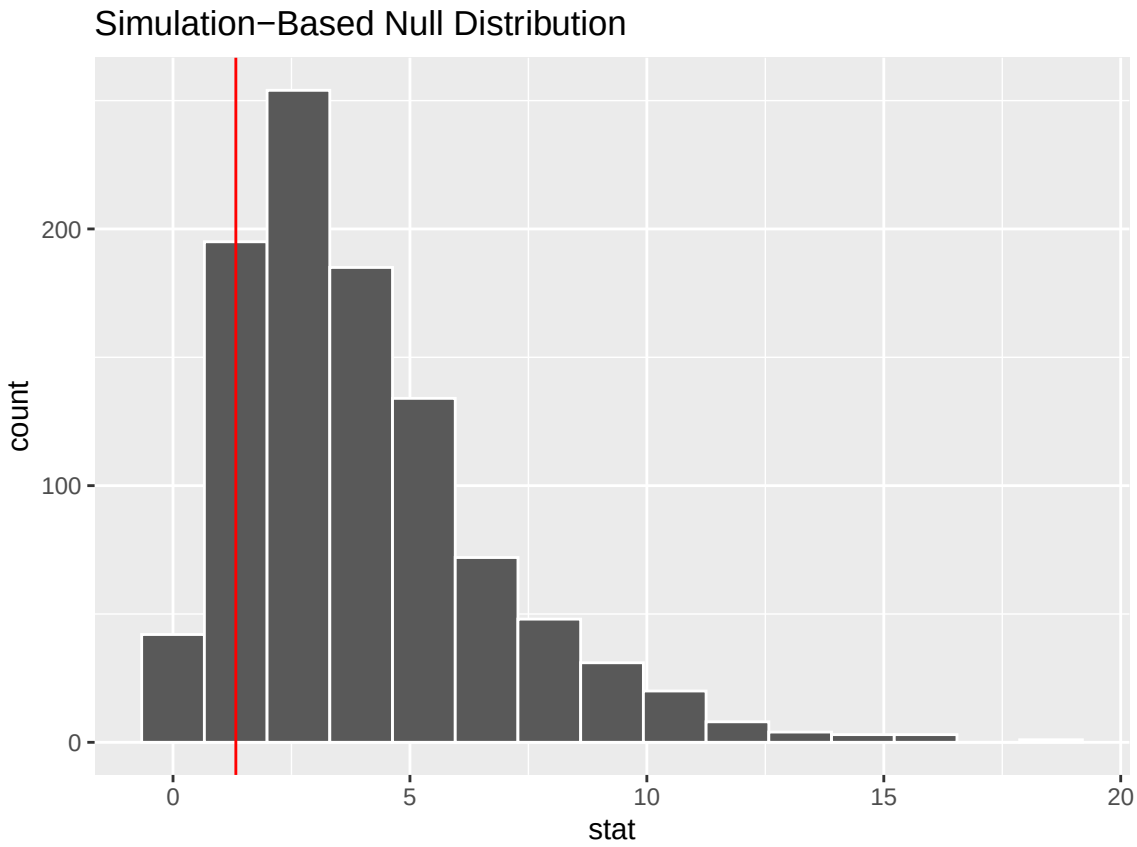
```
gss %>% select(party, NASA) %>%
  mutate(permutation_1 = sample(NASA),
         permutation_2 = sample(NASA))
```

```
## # A tibble: 149 x 4
##    party NASA        permutation_1 permutation_2
##    <fct> <fct>       <fct>         <fct>
##  1 Ind   TOO LITTLE  ABOUT RIGHT   ABOUT RIGHT
##  2 Ind   ABOUT RIGHT ABOUT RIGHT   TOO LITTLE
##  3 Dem   ABOUT RIGHT TOO MUCH      TOO LITTLE
##  4 Ind   TOO LITTLE  TOO MUCH      ABOUT RIGHT
##  5 Ind   TOO MUCH    TOO MUCH      TOO MUCH
##  6 Ind   TOO LITTLE  TOO LITTLE    ABOUT RIGHT
##  7 Ind   ABOUT RIGHT TOO MUCH      TOO MUCH
##  8 Dem   ABOUT RIGHT ABOUT RIGHT   ABOUT RIGHT
##  9 Dem   TOO LITTLE  ABOUT RIGHT   TOO MUCH
## 10 Ind   TOO LITTLE  TOO MUCH      TOO MUCH
## # ... with 139 more rows
```

Ces prémutations représentent ce à quoi on s'attendrait: Si la relation entre les variables était complètement aléatoire Nous pourrions générer de très nombreuses permutations, calculer une statistique Chi-carré pour chacune, et nous nous attendrions à ce que leur distribution se rapproche des fonctions de densité indiquées ci-dessus. Ensuite, nous pourrions tracer nos données sur cette distribution et voir où elle est tombée. Si l'aire sous la courbe à droite du point était inférieure à 5%, nous pourrions rejeter l'hypothèse nulle

INFER REND CETTE APPROCHE PROGRAMMATIQUE ET TRES SIMPLE

```
#Utilisation du package infer (avec 1000 permutations et calculer à chaque fois la statistique chi-carr
gss %>%
  specify(NASA ~ party)  %>%
  hypothesize(null = "independence") %>%
  generate(reps = 1000, type = "permute") %>%
  calculate(stat = "Chisq") %>%
  visualize() +
  # ajouter une ligne verticale pour les données gss
  geom_vline(aes(xintercept = observed_stat), color = "red")
```



Si nous voulions obtenir une valeur p à partir de cette approche programmatique, nous pouvons calculer l'aire sous la courbe à droite de la statistique observée en ajoutant la fonction samurise (p-val...) qui nous permet d'afficher la p-value

```
#obtenir la p-value (générale)
gss %>%
  specify(NASA ~ party)  %>%
  hypothesize(null = "independence") %>%
  generate(reps = 1000, type = "permute") %>%
  calculate(stat = "Chisq") %>%
  summarise(p_val = mean(stat > observed_stat))
```

```
## # A tibble: 1 x 1
##   p_val
##   <dbl>
## 1 0.866
```

Pour avoir le data frame des valeur permutées (les p-values des 1000 permutations) on enleve la fonction visualise.

```
#dataframe du gss
gss %>%
  specify(NASA ~ party)  %>%
  hypothesize(null = "independence") %>%
  generate(reps = 1000, type = "permute") %>%
  calculate(stat = "Chisq") #%>%
```

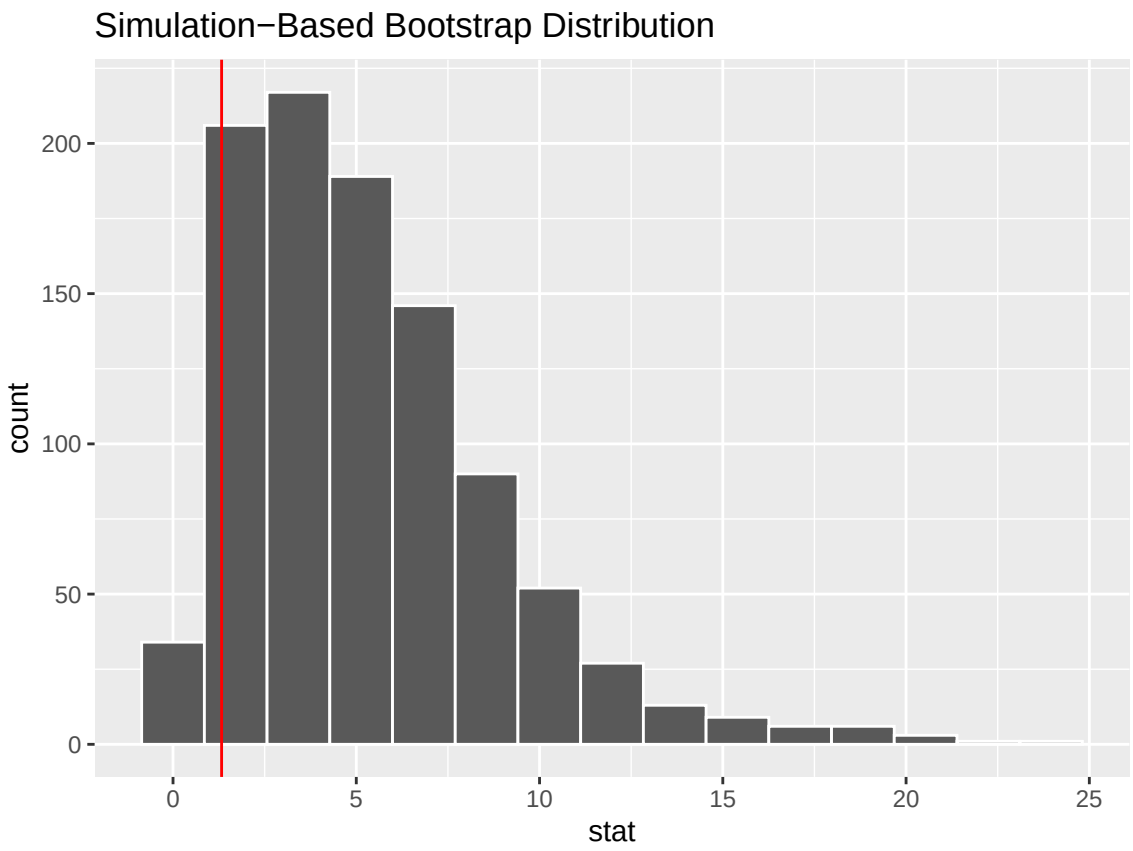
```
## # A tibble: 1,000 x 2
```

```
##    replicate  stat
##        <int> <dbl>
##  1         1  5.47
##  2         2  6.87
##  3         3  6.95
##  4         4  1.17
##  5         5 10.2
##  6         6  2.25
##  7         7  7.18
##  8         8  6.19
##  9         9  1.54
## 10        10  3.37
## # ... with 990 more rows
```

```
  #visualize() +
  #geom_vline(aes(xintercept = observed_stat), color = "red")
```

Si nous omettons d'émettre des hypothèses, nous pouvons amorcer des échantillons à partir de nos données on enlève la fonction hypothesize.

```
#Simulation-Based Boostrap Distribution
gss %>%
  specify(NASA ~ party)  %>%
  #hypothesize(null = "independence") %>%
  generate(reps = 1000, type = "bootstrap") %>%
  calculate(stat = "Chisq") %>%
  visualize() +
  geom_vline(aes(xintercept = observed_stat), color = "red")
```



EXEMPLE 2 les données broceliande

Dans cet exempel on va utiliser les données broceliande pour illustrer la réalisation d'un t-test à l'aide du package infer.

Le jeu de données broceliande recense (en terme d'individus) un certain nombre d'arbres de la forêt de Brocéliande ainsi que (en terme de variables):

-age: leur âge, en années -espece: leur espèce (chêne, châtaignier, hêtre ou sapin) -hauteur: leur hauteur, en cm -largeur: leur largeur, en cm -gui: le nombre de touffes de gui qui les affecte -enchantement: la présence d'un enchantement sur cet arbre (TRUE ou FALSE) -fees: le nombre de fées qui habitent cet arbre -lutins: le nombre de lutins qui habitent cet arbre -perlimpinpin: la quantité de poudre de perlimpinpin dans sa sève (en g/L)

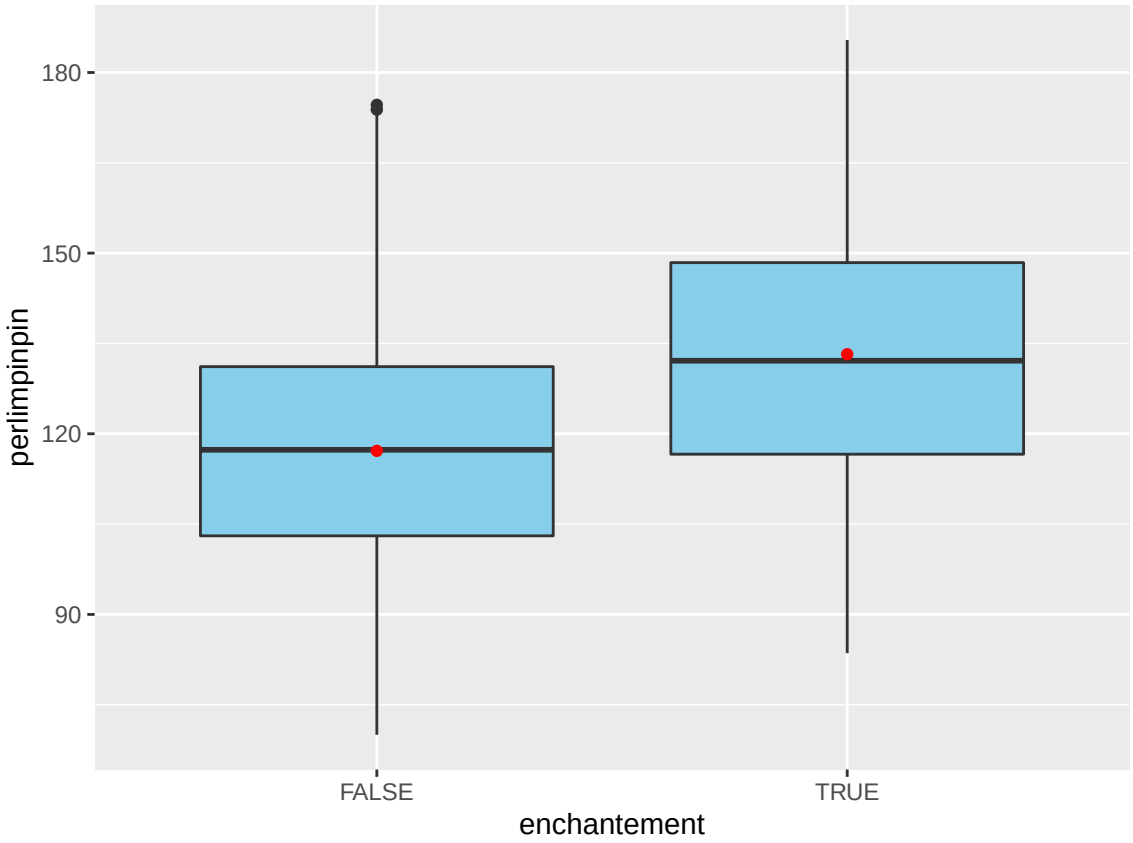
Le jeu de données broceliande est disponible en ligne à cette adresse: "http://perso.ens-lyon.fr/lise.vaudor/grimoireStat/datasets/"

T-test

On va tester l'effet de l'enchantement sur la quantité de poudre de perlimpinpin produite par les arbres.

```
#Charger les données
datasets_path="http://perso.ens-lyon.fr/lise.vaudor/grimoireStat/datasets/"
broceliande=readr::read_delim(paste0(datasets_path,'broceliande.csv'),
                              delim=';')

#Visualiser le graph
ggplot(broceliande, aes(x=enchantement, y=perlimpinpin))+
  geom_boxplot(fill="skyblue")+
  geom_point(data= broceliande %>%
```

```
    group_by(enchantement) %>%
    summarise(perlimpinpin=mean(perlimpinpin)),
  color="red")
```




```
#1er test t-test
montest1 <- broceliande %>%
  t_test(perlimpinpin ~ enchantement,
         order=c(TRUE, FALSE))

t_obs=montest1 %>%
  select(statistic)
```

L'argument order nous permet de dire dans quel sens on effectue la comparaison de moyenne (ici on considère la moyenne des arbres enchantés (enchantement est TRUE) moins la moyenne des arbres non-enchantés (enchantement est FALSE).

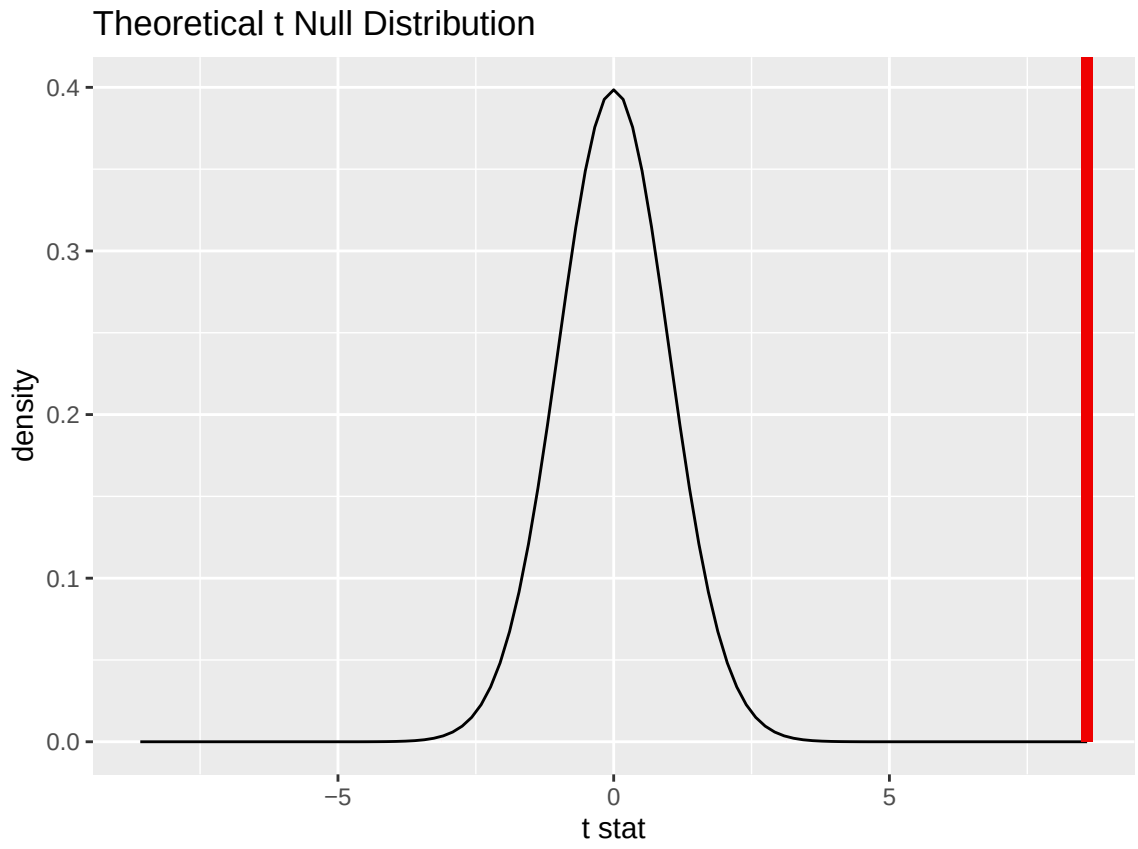
Packages utilisant %>% L'opérateur de pipe est défini dans le package magrittr , mais il a gagné en visibilité et en popularité avec le package dplyr (qui importe la définition de magrittr). Maintenant, il fait partie de tidyverse , une collection de paquets qui "fonctionnent en harmonie car ils partagent des représentations de données communes et la conception de l'API" .

Le package magrittr fournit également plusieurs variantes de l'opérateur de canalisation pour ceux qui souhaitent plus de flexibilité dans les canalisations, tels que le canal de magrittr composé %<>% , le canal d'exposition %$% et l'opérateur de départ %T>% . Il fournit également une suite de fonctions d'alias pour remplacer les fonctions communes qui ont une syntaxe spéciale (+ , [, [[, etc.) afin qu'elles puissent être facilement utilisées dans une chaîne de canaux.

on peut décomposer le processus et produire une visualisation montrant l'emplacement de la statistique

observée par rapport à la distribution théorique attendue sous hypothèse d'indépendance via les verbes du package:

```
#Utilisation du package Infer
broceliande %>%
  specify(perlimpinpin ~ enchantement) %>%
  hypothesize(null = 'independence') %>%
  calculate(stat = 't') %>%
  visualize(method = 'theoretical',
            obs_stat=t_obs,
            direction = 'two_sided')
```



Dans ce cas d'exemple, on se contente de considérer la distribution théorique sous hypothèse nulle car les conditions permettant de supposer que la statistique T suit cette distribution sont respectées. De ce fait, on se sert pas pas du verbe generate()...

En d'autre terme ca ne sert a rien de faire la permutation vu que les conditions dans ce cas sont respectées

Exemple 2: test du Chi-2 (dans ce cas les conditions sont respectées aussi)

DATA Châteaux et Boulots

Le jeu de données chateauxEtBoulots recense (en terme d'individus) un certain nombre de personnes du pays Fantaisie ainsi que (en terme de variables)

activite: leur activité (la royauté, la chevalerie, les enchantements ou la magie noire), sexe: leur sexe (féminin ou masculin), region: leur région (Bois-Jolis,Montage-Sombre ou Flots-Blancs) tenue: leur couleur de tenue (noire, grise, bleue, verte, ou rose). Le jeu de données chateauxEtBoulots est disponible en ligne à cette adresse.

Intéressons-nous maintenant au jeu de données fantaisie, et au lien entre sexe et activite.

#hypothèses H0: il existe un lien entre sexe et activite. H1: il existe pas un lien entre sexe et activite.

```
#charger les données chateauxEtBoulots
chateauxEtBoulots=read.csv("http://perso.ens-lyon.fr/lise.vaudor/grimoireStat/datasets/chateauxEtBoulot

#afficher la structure des donnée
fantaisie=readr::read_delim(paste0(datasets_path,'chateauxEtBoulots.csv'),delim=';')
str(chateauxEtBoulots)
```

```
## 'data.frame':    72 obs. of  4 variables:
##  $ activite: Factor w/ 4 levels "chevalerie","enchantements",..: 1 1 1 1 1 1 1 1 1 ...
##  $ sexe    : Factor w/ 2 levels "feminin","masculin": 2 2 2 2 2 2 2 2 2 ...
##  $ region  : Factor w/ 3 levels "bois-jolis","flots-blancs",..: 1 1 1 1 1 1 2 2 2 ...
##  $ tenue   : Factor w/ 5 levels "bleue","grise",..: 1 1 2 4 5 5 1 1 2 3 ...
```

```
#Réoraganiser les données pour faciliter l'interpretation
fantaisie %>%
  janitor::tabyl(sexe,activite)
```

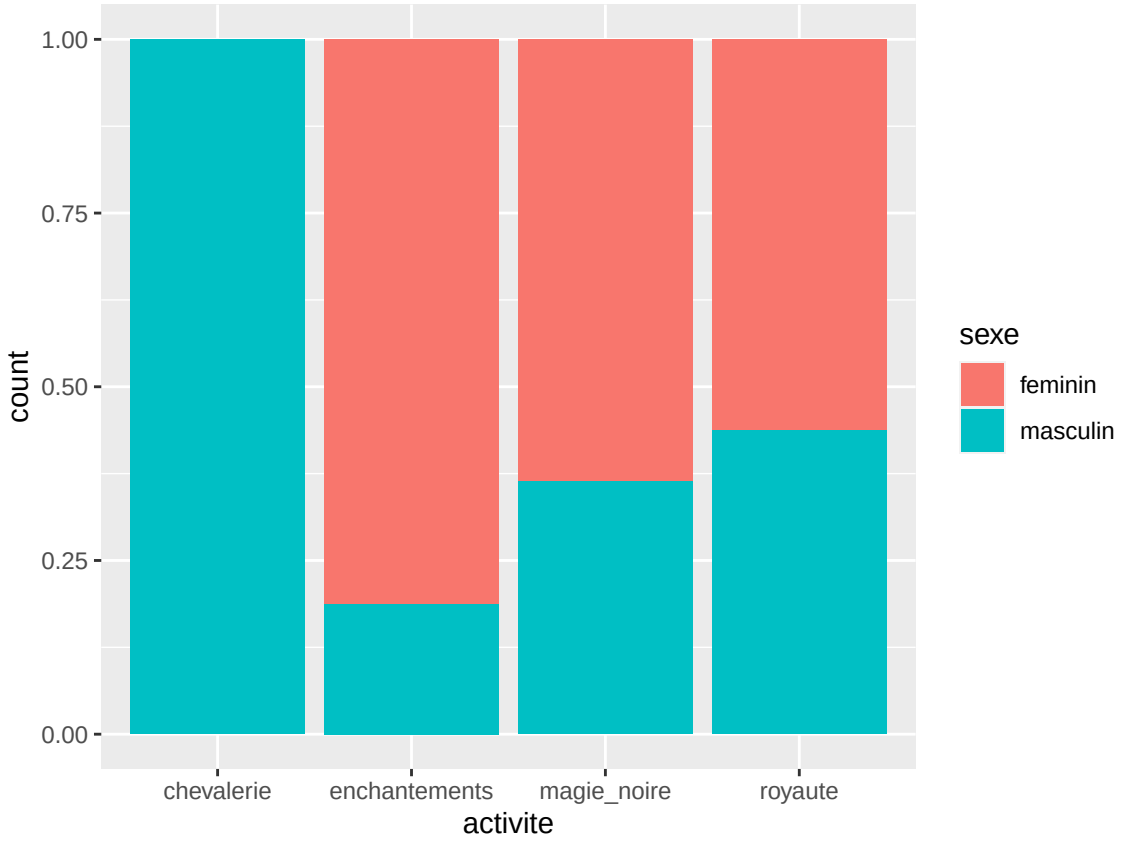
```
##      sexe chevalerie enchantements magie_noire royaute
##   feminin          0            13          14       9
##  masculin         18             3           8       7
```

Le package janitor permet de transformer ce tableau d'effectifs pour afficher à la fois les pourcentages (ici par colonne) et les effectifs, ce qui en facilite l'interprétation.

```
#réorganiser les données
fantaisie %>%
  janitor::tabyl(sexe,activite) %>%
  janitor::adorn_percentages("col") %>%
  janitor::adorn_pct_formatting(digits=2) %>%
  janitor::adorn_ns()
```

```
##      sexe   chevalerie enchantements magie_noire    royaute
##   feminin   0.00%  (0)   81.25% (13) 63.64% (14) 56.25% (9)
##  masculin 100.00% (18)   18.75%  (3) 36.36%  (8) 43.75% (7)
```

```
#Afficher le graph
ggplot(fantaisie,aes(x=activite))+
  geom_bar(aes(fill=sexe),position="fill")
```



La fonction de infer pour réaliser un test du 2 est chisq_test()

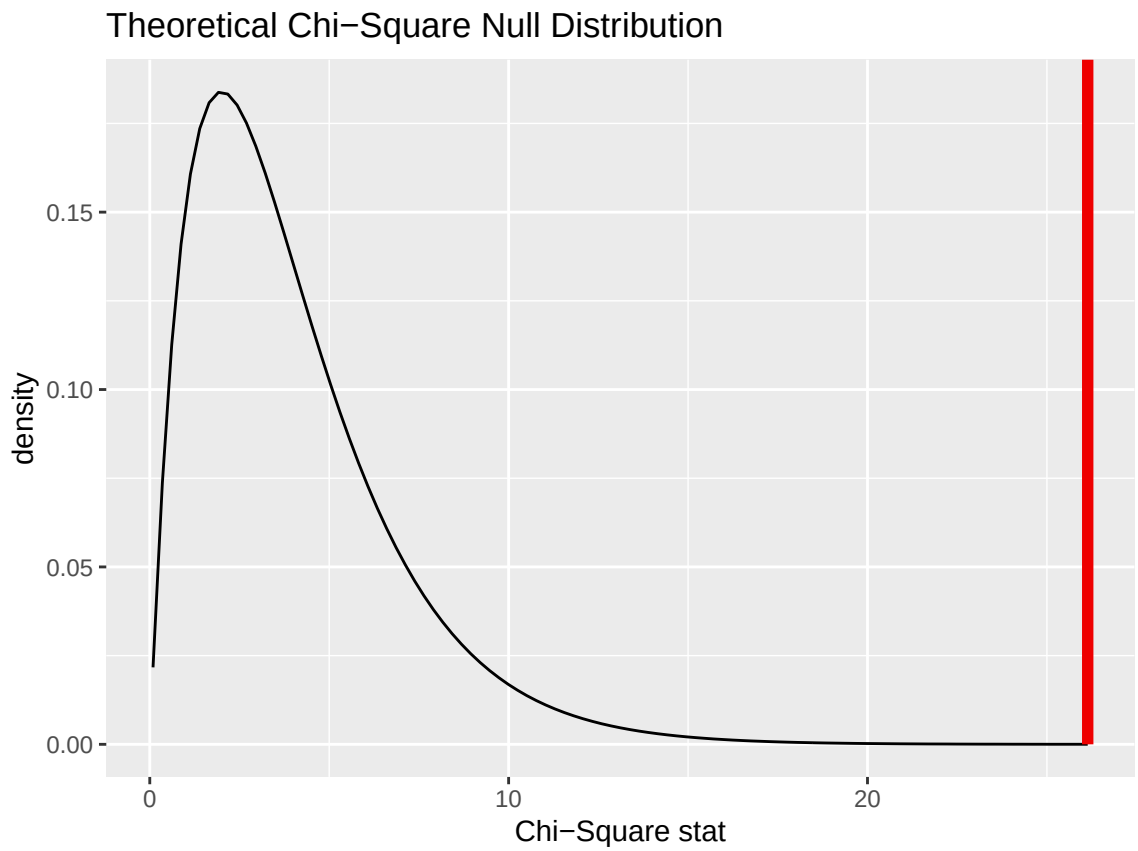
```
#Fonction pour réaliser un test de chi-2

montest2 <- fantaisie %>%
#test chi-2 pour verifier s'il existe une relation entre sexe et l'activité
  chisq_test(formula=activite~sexe)
print(montest2)
```

```
## # A tibble: 1 x 3
##   statistic chisq_df    p_value
##       <dbl>    <int>      <dbl>
## 1      26.1        3 0.00000893
```

```
chi_obs=montest2 %>%
  pull(statistic)
```

Pour visualiser la distribution théorique de la statistique et l'emplacement de la statistique observée, la logique est exactement la même que dans le cas du t-test (sauf qu'on adapte la nature de la statistique calculée!):

```
fantaisie %>%
  specify(tenue~sexe) %>%
  hypothesize(null='independence') %>%
  calculate(stat='Chisq') %>%
  visualize(method = 'theoretical',
            obs_stat=chi_obs,
            direction = 'greater')
```



Exemple 3: test du Chi-2 en cas de non-respect des conditions d'application du test classique

Un test du 2 classique (c'est-à-dire reposant sur la distribution théorique du 2) est construit en supposant que les effectifs croisés sont 'suffisants'. Ainsi, quand certaines cases du tableau de contingence comprennent trop peu d'individus, appliquer un test du 2 peut causer un 'warning' stipulant que l'approximation du 2 peut être incorrecte.

data Châteaux et Boulots

Le jeu de données chateauxEtBoulots recense (en terme d'individus) un certain nombre de personnes du pays Fantaisie ainsi que (en terme de variables)

activite: leur activité (la royauté, la chevalerie, les enchantements ou la magie noire), sexe: leur sexe (féminin ou masculin), region: leur région (Bois-Jolis,Montage-Sombre ou Flots-Blancs) tenue: leur couleur de tenue (noire, grise, bleue, verte, ou rose). Le jeu de données chateauxEtBoulots est disponible en ligne à cette adresse.

```
#chargement de données Châteaux et Boulots
chateauxEtBoulots=read.csv("http://perso.ens-lyon.fr/lise.vaudor/grimoireStat/datasets/chateauxEtBoulots
#afficher la structure des données
str(chateauxEtBoulots)
```

```
## 'data.frame':    72 obs. of  4 variables:
##  $ activite: Factor w/ 4 levels "chevalerie","enchantements",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ sexe    : Factor w/ 2 levels "feminin","masculin": 2 2 2 2 2 2 2 2 2 2 ...
##  $ region  : Factor w/ 3 levels "bois-jolis","flots-blancs",..: 1 1 1 1 1 1 2 2 2 2 ...
##  $ tenue   : Factor w/ 5 levels "bleue","grise",..: 1 1 2 4 5 5 1 1 2 3 ...
```

Considérons par exemple un sous-jeu de données (fantaisie) rassemblant uniquement les individus de noble extraction, et intéressons-nous au lien entre leur sexe et leur couleur de tenue:

```
#filtrer et réorganiser
fantaisie_princiere=filter(fantaisie, activite=='royaute')

fantaisie_princiere %>%
  janitor::tabyl(sexe,tenue) %>%
  janitor::adorn_percentages("col") %>%
  janitor::adorn_pct_formatting(digits=2) %>%
  janitor::adorn_ns()
```

```
##      sexe      bleue      noire        rose      verte
##   feminin 75.00% (3) 25.00% (1) 100.00% (2) 50.00% (3)
##  masculin 25.00% (1) 75.00% (3)   0.00% (0) 50.00% (3)
```

Je réalise un test du 2 “classique”:

```
montest3 <- fantaisie_princiere %>%
  chisq_test(tenue~sexe)
montest3
```

```
## # A tibble: 1 x 3
##   statistic chisq_df p_value
##       <dbl>    <int>   <dbl>
## 1      3.81        3   0.283
```

Ici j’ai un warning car, de fait, certains niveaux croisés sexe*tenue comprennent très peu d’individus! Dans ce cas le recours à des permutations pour calculer la distribution de la statistique sous hypothèse d’indépendance s’avère très utile:

VOILA L’UTLITE DE INFER dans le cas ou les condition ne sont pas respectées (dans ce cas n est trop petit)

```
#utilistaion du package infer pour effectuer le trest KHi-2 avec 1000 Permutation
sim_princiere <- fantaisie_princiere %>%
  specify(tenue~sexe) %>%
  hypothesize(null='independence') %>%
  generate(reps=1000, type='permute') %>%
  calculate(stat='Chisq')
```

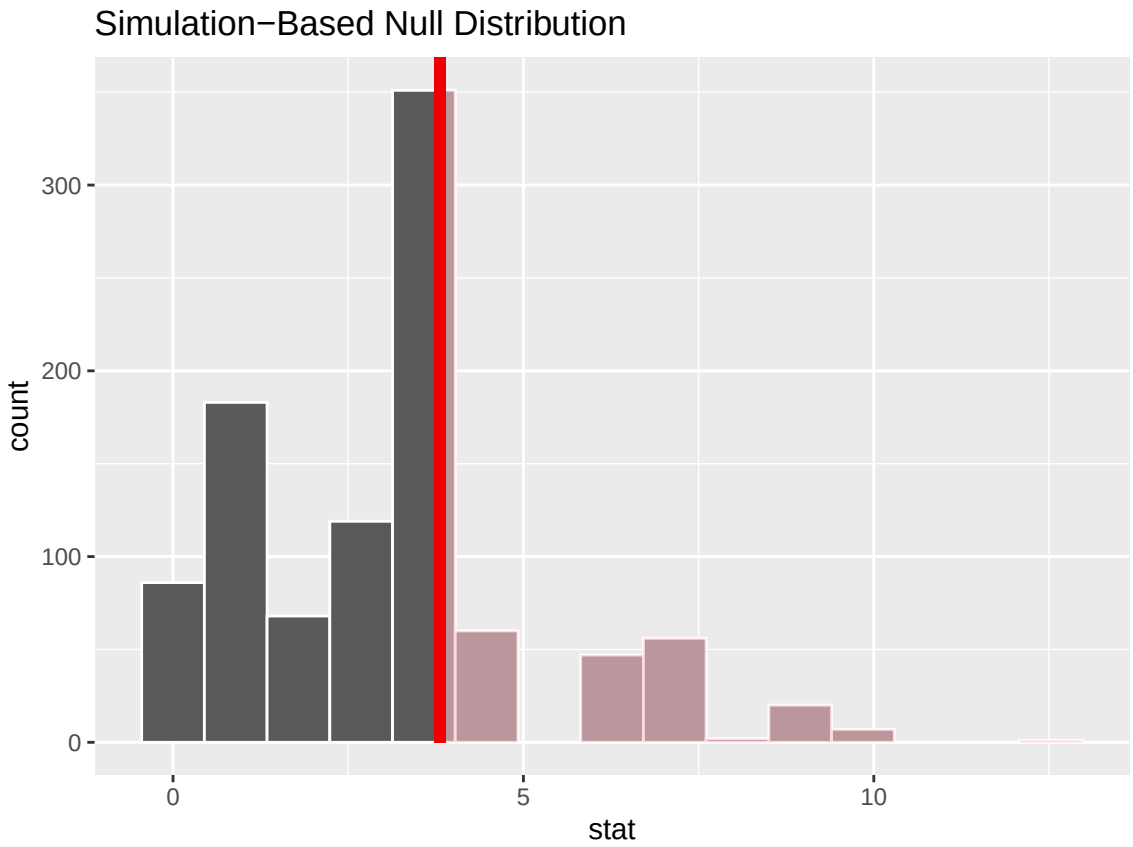
Ici la p-value correspond à la proportion des cas (parmi les 1000 permutations) où la valeur de statistique observée sur données permutées a été supérieure à la statistique observée sur nos vraies données:

```
#obtenir la p-value
sim_princiere %>%
  get_pvalue(obs_stat = montest3$statistic,
             direction = "greater")
```

```
## # A tibble: 1 x 1
##   p_value
##     <dbl>
## 1   0.264
```

La distribution de la statistique du 2 pour ces permutations est montrée par l’histogramme.

```
sim_princiere%>%
  visualize(obs_stat=montest3$statistic,
            direction='greater')
```



De fait, ici, on obtient une valeur de p-value extrêmement proche de celle que l'on avait obtenue en considérant la distribution théorique du 2…