

Refresher

Pratiques de la Recherche en Économie

Florentine Oliveira

2024-11-05

About me

Doctorante en Économie à PSE en 3ème année

Champs de recherche: économie de la famille, de l'éducation, du travail

Pour me contacter, si vous avez des questions ou envie/besoin de parler:

florentine.oliveira@psemail.eu

Campus Jourdan, Bureau R4-54

Objectifs du cours

- Se familiariser avec
 - les techniques **économétriques** de base
 - la **recherche en économie empirique**
- Apprendre le language de programmation R

⇒ pour pouvoir réaliser votre **mémoire** !

Roadmap

Semestre 1

1. Séance 1 (aujourd'hui): Rappels de `R`
2. Séance 2 (12/11/2024): *LATEX* et R Markdown
3. Séance 3 (19/12/2024): Régressions Linéaires et Causalité

Semestre 2

1. Séance 4 (04/02/2025): Variables de Contrôle et Matching
2. Séance 5 (18/02/2025): Méthode des Variables Instrumentales
3. Séance 6 (11/03/2025): Méthode des Doubles-Différences (DiD)
4. Séance 7 (25/03/2025): Regression Discontinuity Design (RDD)
5. Séance 8 (08/04/2025): Q & A

Validation du cours

DM

Objectif: manipulation de données sur R, intuition, quelques régressions

À faire seul(e) ou à deux

À rendre pour le 10 Janvier 23h59

Mémoire

Cette séance

1. La recherche en économie empirique

2. R: Rappels

2.1. Définition, usages, interface

2.2. Projet R

2.3. Importer des données

2.4. Manipulation des données avec dplyr

2.5. Visualisation des données avec ggplot

2.6. Maps

1. La recherche en économie empirique

Qu'est-ce que la recherche en économie empirique?

- Répondre à des **questions de recherche** pour apprendre sur les comportements humains, l'efficacité des politiques publiques, ...

1. La recherche en économie empirique

Qu'est-ce que la recherche en économie empirique?

- Répondre à des **questions de recherche** pour apprendre sur les comportements humains, l'efficacité des politiques publiques, ...
 - Quel est l'effet des bourses de scolarité **sur** la réussite scolaire?
 - Quel est l'effet d'un allongement de la durée du congé parental **sur** la participation au marché du travail des mères?
 - Quelle est l'ampleur de la discrimination dans le processus de recrutement?
 - Quel est l'effet d'un assouplissement des conditions d'éligibilité à l'assurance chômage **sur** l'emploi?
 - Quel est l'effet de la pollution atmosphérique **sur** la productivité?
 - Quel est le rôle des médias **sur** le comportement de vote?
 - Quel est l'effet de la densité de population **sur** les salaires?

1. La recherche en économie empirique

Qu'est-ce que la recherche en économie empirique?

- Répondre à des **questions de recherche** pour apprendre sur les comportements humains, l'efficacité des politiques publiques, ...
 - Grande diversité des champs de recherche: éducation, travail, famille, environnement, macroéconomie, politique, migration, genre, urban, crime, économie historique, ...
- ... en utilisant des techniques statistiques rigoureuses (= **économétrie**) ...
- ... une stratégie/un cadre qui nous permet d'identifier un **effet causal**...
- ... et des données!
- R est un logiciel qui permet de réaliser ces analyses descriptives et empiriques.

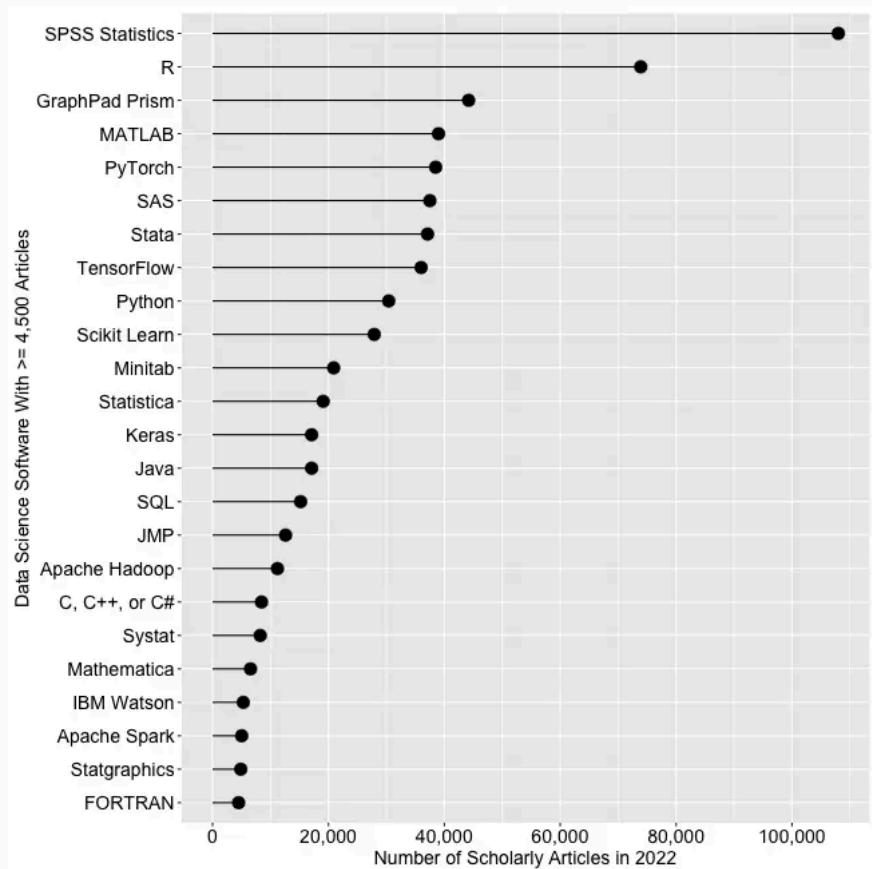
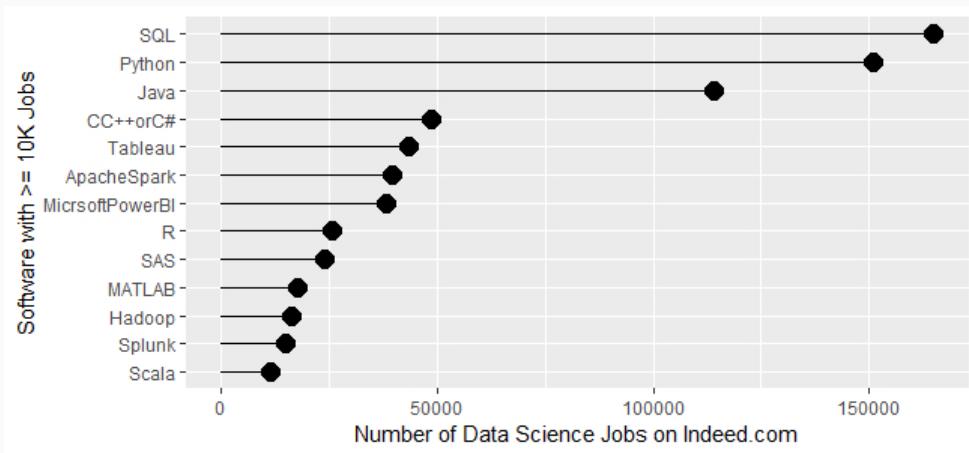
2. R: rappels

2. R: rappels

2.1. Définition, usages, interface

- D'après le site internet de R : *R is a free software environment for statistical computing and graphics.*
 - un logiciel (interface R studio)
 - un language de programmation
- R est **gratuit** et en **open source**
 - donc très utilisé dans les milieux académique et institutionnel
- Logiciel très **polyvalent**
 - statistiques descriptives, analyses économétriques, analyse de données spatiales, ML, Rmarkdown
- Compétence valorisée et de plus en plus recherchée (public & privé)

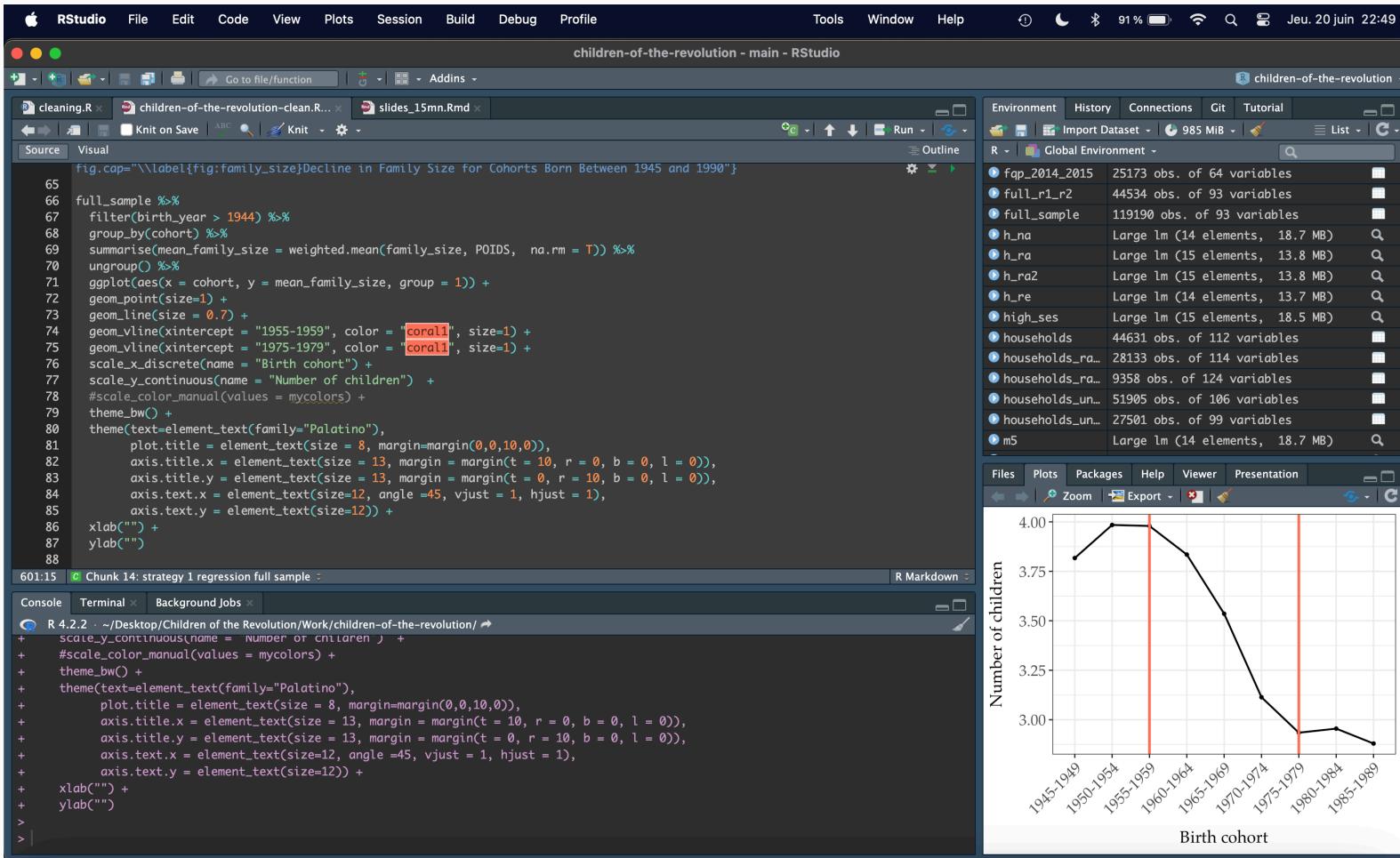
Popularité de R¹



[1]: [The Popularity of Data Science Software](#)

2. R: rappels

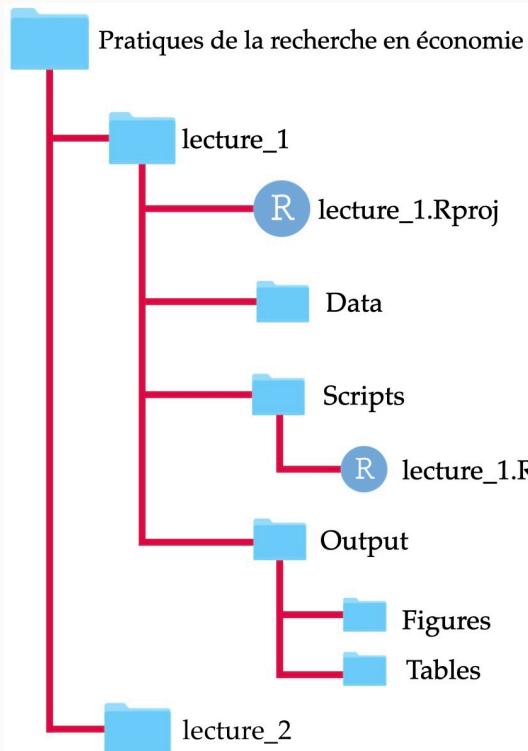
2.1. Définition, usages, interface



2. R: rappels

2.2. Projet R

Les projets R permettent une meilleure organisation de votre travail.

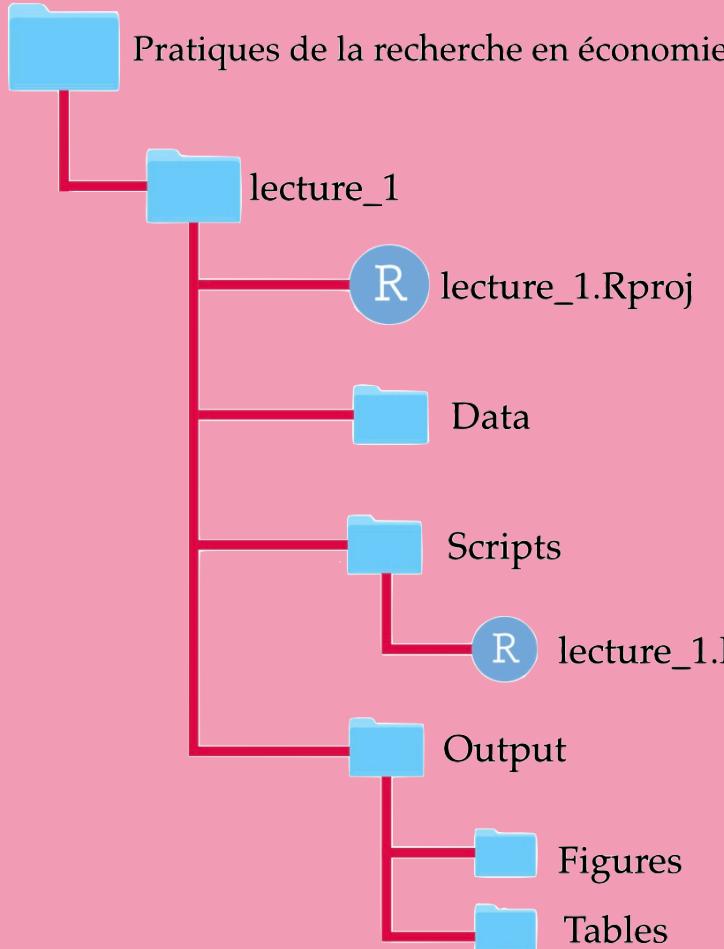


Création d'un projet R:

- Dans R-Studio
 - Fichier → Nouveau projet

Application: créer votre projet R

03:00



- Créer un projet R pour la Lecture 1 en respectant cette structure
- Créer un nouveau script R, enregistré dans le dossier lecture_1

2. R: rappels

2.3. Importer des données

Format .csv:

- `read.csv()` ou `read.csv2()` avec l'argument `sep = ','`

Formats .dta (stata), .sas7bdat (sas): package `haven` (nb: `haven` est inclus dans `tidyverse`)

- `.dta` : `read_dta()`
- `.sas7dbat` : `read_sas()`

Format .xls ou .xlsx (Excel): package `readxl`

- `read_excel()`

2. R: rappels

2.3. Importer des données

Format .rds: format de données le moins lourd et optimal pour travailler sur R, surtout quand les bases de données sont très volumineuses

- `readRDS()`

Conseil: à la fin du script de cleaning, enregistrer le ou les data.frames nettoyés en format `.RDS` avec la fonction `saveRDS()` et importer ces données nettoyées dans le script sur lequel vous ferez toutes vos analyses.

2. R: rappels

2.4. Manipulation des données avec `dplyr`

Principal package pour la manipulation de données : `dplyr`

`data.frame` $\underbrace{\%>\%}_{\text{pipe}}$ `function(your action)`



2. R: rappels

2.4. Manipulation des données avec dplyr

Principal package pour la manipulation de données : `dplyr`

`data.frame` $\underbrace{\%>\%}_{\text{pipe}}$ `function(your action)`



Principales fonctions

- `mutate()`: créer ou modifier des variables
- `summarise()`: résume les données par des statistiques descriptives
- `filter()`: conserve ou supprime les lignes
- `group_by()`: permet de grouper des observations entre elles
- `select()`: conserve ou supprime des colonnes
- `arrange()`: ordonne les observations par rapport aux valeurs d'une colonne

2. R: rappels

2.4. Manipulation des données avec `dplyr`

Opérateurs

Opérateur arithmétiques	Description
<code>+ / -</code>	Addition / Soustraction
<code>* / /</code>	Multiplication / Division
<code>^</code>	Exposant

Opérateur logiques	Description
<code>< / ></code>	Strictement inférieur / supérieur à
<code>< = / > =</code>	Supérieur / inférieur ou égal à
<code>= =</code>	Égal à
<code>! =</code>	Different de
<code>x %in% c(1,6)</code>	x dans $\{1, 6\}$
<code>x & y</code>	x et y
<code>x y</code>	x ou y

Application : Titanic

Ce jeu de données fournit des informations sur le sort des passagers du paquebot « Titanic », résumées en fonction du statut économique (class), du sexe, de l'âge et de la survie.

```
titanic = as.data.frame(Titanic)  
head(titanic)
```

```
##   Class     Sex   Age Survived Freq  
## 1  1st      Male Child     No     0  
## 2  2nd      Male Child     No     0  
## 3  3rd      Male Child     No    35  
## 4 Crew      Male Child     No     0  
## 5  1st Female Child     No     0  
## 6  2nd Female Child     No     0
```

Application : Titanic

05:00

- 1) Calculer le taux de survie par classe, par sexe, et par âge
- 2) Créer une variable `passenger` qui vaut 1 si l'individu n'est pas un membre d'équipage (*crew*), 0 si c'est un membre d'équipage
- 3) Créer une table qui résume, pour les adultes uniquement, le taux de survie par Classe x Sexe et ordonner par ordre décroissant

Solution : Titanic

1) Calculer le taux de survie par classe, par sexe, et par âge

```
titanic %>%
  group_by(Class) %>%
  summarise(tx_survie = sum(Freq[Survived == "Yes"])/sum(Freq)) %>%
  ungroup()
```

```
## # A tibble: 4 × 2
##   Class    tx_survie
##   <fct>     <dbl>
## 1 1st      0.625
## 2 2nd      0.414
## 3 3rd      0.252
## 4 Crew     0.240
```

Solution : Titanic

1) Calculer le taux de survie par classe, par sexe, et par âge

```
titanic %>%
  group_by(Class) %>%
  summarise(tx_survie = sum(Freq[Survived == "Yes"])/sum(Freq)) %>%
  ungroup()
```

```
## # A tibble: 4 × 2
##   Class    tx_survie
##   <fct>     <dbl>
## 1 1st      0.625
## 2 2nd      0.414
## 3 3rd      0.252
## 4 Crew     0.240
```

Solution : Titanic

1) Calculer le taux de survie par classe, par sexe, et par âge

```
titanic %>%
  group_by(Sex) %>%
  summarise(tx_survie = sum(Freq[Survived == "Yes"])/sum(Freq)) %>%
  ungroup()
```

```
## # A tibble: 2 × 2
##   Sex      tx_survie
##   <fct>    <dbl>
## 1 Male     0.212
## 2 Female   0.732
```

Solution : Titanic

1) Calculer le taux de survie par classe, par sexe, et par âge

```
titanic %>%
  group_by(Age) %>%
  summarise(tx_survie = sum(Freq[Survived == "Yes"])/sum(Freq)) %>%
  ungroup()
```

```
## # A tibble: 2 × 2
##   Age     tx_survie
##   <fct>    <dbl>
## 1 Child     0.523
## 2 Adult     0.313
```

Solution : Titanic

2) Créer une variable `passenger` qui vaut 1 si l'individu n'est pas un membre d'équipage (`crew`), 0 si c'est un membre d'équipage

```
# Solution 1
titanic %>%
  mutate(passenger = ifelse(Class != "Crew", 1, 0))

# Solution 2
titanic %>%
  mutate(passenger = case_when(Class != "Crew" ~ 1,
                                Class == "Crew" ~ 0))
```

Conseil: utiliser `case_when` quand il y a plus d'une condition.

Solution : Titanic

3) Créer une table qui résume, pour les adultes uniquement, le taux de survie par Classe x Sexe et ordonner par ordre décroissant

```
titanic %>%
  filter(Age == "Adult") %>%
  group_by(Class, Sex) %>%
  summarise(tx_survie = sum(Freq[Survived == "Yes"]) / sum(Freq)) %>%
  arrange(desc(tx_survie))
```

Solution : Titanic

3) Créer une table qui résume, **pour les adultes uniquement**, le taux de survie par Classe x Sexe et ordonner par ordre décroissant

```
titanic %>%
  filter(Age == "Adult") %>%
  group_by(Class, Sex) %>%
  summarise(tx_survie = sum(Freq[Survived == "Yes"]) / sum(Freq)) %>%
  arrange(desc(tx_survie))
```

Solution : Titanic

3) Créer une table qui résume, pour les adultes uniquement, le taux de survie par **Classe x Sexe** et ordonner par ordre décroissant

```
titanic %>%
  filter(Age == "Adult") %>%
  group_by(Class, Sex) %>%
  summarise(tx_survie = sum(Freq[Survived == "Yes"]) / sum(Freq)) %>%
  arrange(desc(tx_survie))
```

Solution : Titanic

3) Créer une table qui résume, pour les adultes uniquement, le **taux de survie** par Classe x Sexe et ordonner par ordre décroissant

```
titanic %>%
  filter(Age == "Adult") %>%
  group_by(Class, Sex) %>%
  summarise(tx_survie = sum(Freq[Survived == "Yes"]) / sum(Freq)) %>%
  arrange(desc(tx_survie))
```

Solution : Titanic

3) Créer une table qui résume, pour les adultes uniquement, le taux de survie par Classe x Sexe et **ordonner par ordre décroissant**

```
titanic %>%
  filter(Age == "Adult") %>%
  group_by(Class, Sex) %>%
  summarise(tx_survie = sum(Freq[Survived == "Yes"]) / sum(Freq)) %>%
  arrange(desc(tx_survie))
```

```
## # A tibble: 8 × 3
## # Groups:   Class [4]
##   Class Sex     tx_survie
##   <fct> <fct>     <dbl>
## 1 1st   Female    0.972
## 2 Crew  Female    0.870
## 3 2nd   Female    0.860
## 4 3rd   Female    0.461
## 5 1st   Male      0.326
## 6 Crew  Male      0.223
## 7 3rd   Male      0.162
## 8 2nd   Male      0.0833
```

2. R: rappels

2.4. Manipulation des données avec `dplyr`

The diagram illustrates the addition of two data frames, **a** and **b**, to produce a result. Table **a** has columns **x1** and **x2**, with rows A (1), B (2), and C (3). Table **b** has columns **x1** and **x3**, with rows A (T), B (F), and D (T). The addition is represented by a plus sign (+) between the two tables, followed by an equals sign (=) and a question mark (?).

a		b		
x1	x2	x1	x3	
A	1	A	T	
B	2	B	F	
C	3	D	T	

- Très souvent, les données nécessaires à l'analyse sont issues de plusieurs sources
- Joindre deux tables de données peut permettre d'ajouter
 - des variables supplémentaires
 - des observations supplémentaires
- L'ajout de variables peut se faire sur la base:
 - de la position des observations dans les deux jeux de données: **binding joints**
 - relativement aux valeurs d'une ou plusieurs autres colonnes, les clés: **mutating joints**

2. R: rappels

2.4. Manipulation des données avec `dplyr`

Binding joints

Binding Rows

The diagram illustrates the process of binding rows from two data frames, **a** and **b**, to produce a new data frame. Data frame **a** has columns **x1** and **x2**, with rows A, B, and C having values 1, 2, and 3 respectively. Data frame **b** has columns **x1** and **x3**, with rows A, B, and D having values T, F, and T respectively. The operation is represented by a plus sign (+) between the two data frames, followed by an equals sign (=) and the resulting data frame.

x1	x2
A	1
B	2
C	3

+

x1	x3
A	T
B	F
D	T

=

x1	x2	x3
A	1	NA
B	2	NA
C	3	NA
A	NA	T
B	NA	F
D	NA	T

`bind_rows(a, b)`: dans une jointure de lignes, les colonnes sont associées à leur nom et toute colonne manquante est remplacée par NA.

2. R: rappels

2.4. Manipulation des données avec `dplyr`

Binding joints

Binding Columns

The diagram illustrates the binding of columns from two data frames, **a** and **b**, resulting in a third data frame. Data frame **a** has columns **x1** and **x2**, with rows A, B, and C. Data frame **b** has columns **x1** and **x3**, with rows A, B, and D. The operation is represented by a plus sign (+) between the two data frames, followed by an equals sign (=) and the resulting data frame.

a		b	
x1	x2	x1	x3
A	1	A	T
B	2	B	F
C	3	D	T

+

=

x1	x2	x1	x3
A	1	A	T
B	2	B	F
C	3	D	T

`bind_cols(a, b)`: dans une jointure de colonnes, les lignes sont mises en correspondance en fonction de leur **position**

2. R: rappels

2.4. Manipulation des données avec `dplyr`

Mutating joints

Left-joint

a

x1	x2
A	1
B	2
C	3

+

b

x1	x3
A	T
B	F
D	T

=

x1	x2	x3
A	1	T
B	2	F
C	3	NA

`left_join(a, b, by = "x1")`: pour chaque ligne de **a** on a ajouté les colonnes de **b** pour lesquelles la valeur de la clé **x1** est la même

2. R: rappels

2.4. Manipulation des données avec `dplyr`

Mutating joints

Right-joint

a		b		=
x1	x2	x1	x3	
A	1	A	T	
B	2	B	F	
C	3	D	T	

x1	x3	x2
A	T	1
B	F	2
D	T	NA

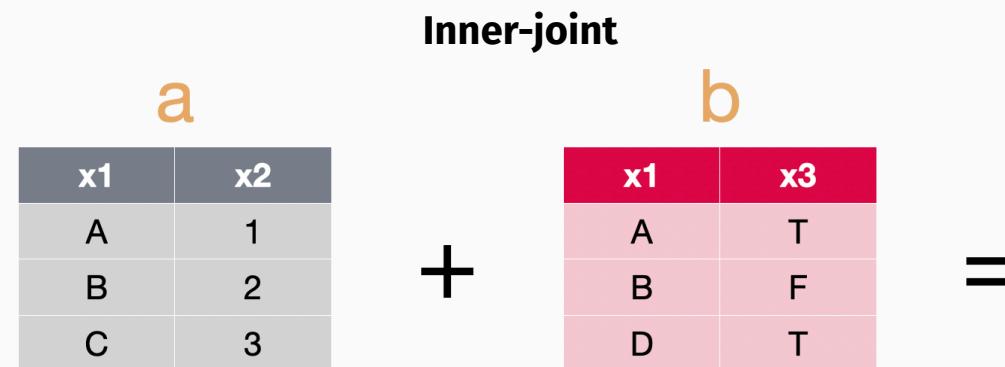
`right_join(a, b, by = "x1")`: pour chaque ligne de **b** on a ajouté les colonnes de **a** pour lesquelles la valeur de la clé **x1** est la même

2. R: rappels

2.4. Manipulation des données avec `dplyr`

Mutating joints

Inner-joint



a		b		=
x1	x2	x1	x3	
A	1	A	T	
B	2	B	F	
C	3	D	T	

x1	x2	x3
A	1	T
B	2	F

`inner_join(a, b, by = "x1")`: seules les lignes présentes à la fois dans **a** et **b** sont jointes

2. R: rappels

2.4. Manipulation des données avec `dplyr`

Mutating joints

Full-joint

a		b	
x1	x2	x1	x3
A	1	A	T
B	2	B	F
C	3	D	T

+

=

x1	x2	x3
A	1	T
B	2	F
C	3	NA
D	NA	T

`full_join(a, b, by = "x1")`: toutes les lignes de **a** et toutes les lignes de **b** sont jointes (avec des NA ajoutés si nécessaire) même si elles sont absentes de l'autre table

2. R: rappels

2.4. Manipulation des données avec dplyr

Formats wide et long

Pour **rassembler des colonnes** et passer en format **long**: pivot_longer

The diagram illustrates the transformation of a wide-format data frame 'a' into a long-format data frame 'b'. An arrow points from 'a' to 'b', indicating the direction of the conversion.

Data Frame a:

student	L1	L2
A	13.7	12.4
B	10.5	14.1
C	16.2	15.8

Data Frame b:

student	grade	mean
A	L1	13.7
B	L1	10.5
C	L1	16.2
A	L2	12.4
B	L2	14.1
C	L2	15.8

```
b = pivot_longer(a, cols = 2:3, names_to = "grade", values_to = "mean")
```

2. R: rappels

2.4. Manipulation des données avec dplyr

Formats wide et long

Pour **dispercer des lignes** et obtenir un format **wide**: pivot_wider

The diagram illustrates the transformation of a wide data frame 'a' into a long data frame 'b' using the `pivot_wider` function.

Data Frame a:

student	grade	subject	mean
A	L1	statistics	12.3
A	L1	macro	10.9
A	L2	statistics	19
A	L2	macro	13.6
B	L1	statistics	11.2
B	L1	macro	17.1
B	L2	statistics	8.2
B	L2	macro	14.7
C	L1	statistics	12.8
C	L1	macro	13.5
C	L2	statistics	14.1
C	L2	macro	13.8

Data Frame b:

student	grade	statistics	macro
A	L1	12.3	10.9
A	L2	19	13.6
B	L1	11.2	17.1
B	L2	8.2	14.7
C	L1	12.8	13.5
C	L2	14.1	13.8

```
b = pivot_wider(a, names_from = subject, values_from = mean)
```

Application : Législatives 2024

Utilisation de 3 bases de données:

- résultats au premier tour: `resultats-definitifs-par-circonscription-t1.csv`
- résultats au second tour: `resultats-definitifs-par-circonscription-t2.csv`
- population par circonscription en 2019: `population_2019.csv`

Chaque observation = une circonscription

Les bases de données des résultats aux élections sont en format wide:

- chaque circonscription à un nombre fini de candidats (jusqu'à 19 candidats)
- les informations relatives aux différents candidats (nom, sexe, parti politique, etc) sont renseignées dans différentes variables
- les variables relatives au même candidat ont le même suffixe (eg Nom.candidat.1, Nuance.candidat.1, ...)

Application : Législatives 2024

⭐ Encodage du libellé des circonscriptions

```
head(resultats_t1[1:4,c(2,4)])
```

```
##   Libellé.département Libellé.circonscription.législative
## 1               Ain           1ère circonscription
## 2               Ain           2ème circonscription
## 3               Ain           3ème circonscription
## 4               Ain           4ème circonscription
```

```
head(pop_2019)[1:4, ]
```

```
##   X. Circonscriptions Population.20191 ... hab ..
## 1 1               Ain, 1re           122750
## 2 2               Ain, 2e           137975
## 3 3               Ain, 3e           146110
## 4 4               Ain, 4e           128896
```

Variables Clés: `Libellé.département` qui comprend uniquement le nom du département, et `Libellé.circonscription` qui contient uniquement le numéro de la circonscription.

Application : Législatives 2024

⭐ Encodage du libellé des circonscriptions

```
resultats_t1 = resultats_t1 %>%
  mutate(Libellé.circonscription.législative = gsub("\\D", "", Libellé.circonscription.législative))

unique(resultats_t1$Libellé.circonscription.législative)

## [1] "1"   "2"   "3"   "4"   "5"   "6"   "7"   "8"   "9"   "10"  "11"  "12"  "13"  "14"  "15"
## [16] "16"  "17"  "18"  "19"  "20"  "21"
```

$\underbrace{\text{gsub}}_{\text{remplace tout ce qui n'est pas numérique}} \left(\underbrace{"\\D"}_{\text{par une chaîne de caractères vide}}, \underbrace{""}_{\text{dans la variable Libellé.circonscription.législative}}, \underbrace{\text{Libellé.circonscription.législative}}_{\text{Libellé.circonscription.législative}} \right)$

Application : Législatives 2024

⭐ Encodage du libellé des circonscriptions

```
pop_2019 = pop_2019 %>%
  separate(Circonscriptions, into = c("Libellé.département", "Libellé.circonscription.législative"), sep = ", ")
  mutate(Libellé.circonscription.législative = gsub("\\D", "", Libellé.circonscription.législative))

unique(pop_2019$Libellé.circonscription.législative)
```

```
## [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10" "11" "12" "13" "14" "15"
## [16] "16" "17" "18" "19" "20" "21"
```

$\underbrace{\text{separate}}_{\text{sépare}} \underbrace{(\text{"Circonscriptions"})}_{\text{les caractères de la variable Circonscriptions}}, \underbrace{\text{into} = \text{c}(\text{"A"}, \text{"B"})}_{\text{en deux chaînes de caractères qui définissent ces deux variables}}, \underbrace{\text{sep} = \text{",",}}_{\text{la séparation étant faite à l'endroit de la virgule}}$

Application : Législatives 2024

10:00

- 1) Importer les données (nommer les dataframes `resultats_t1`, `resultats_t2` et `pop_2019`)
- 2) Effectuer sur chaque jeu de données les opérations nécessaires pour harmoniser l'encodage du libellé de circonscription
 - **pop-19:** `separate(Circonscriptions, into = c("Libellé.département", "Libellé.circonscription.législative"), sep = ", ")`
 - **resultats_t1 et resultats_t2 et pop_2019:** `gsub("\\"D", "", Libellé.circonscription.législative)`
- 3) Reformatter `resultats_t1` et `resultats_t2` en format long (chaque observation = un candidat)
- 4) Ajouter une variable `tour` qui vaut 1 dans `resultats_t1` et 2 dans `resultats_t2`
- 5) Ajouter les observations de `resultats_t2` à `resultats_t1` et stocker le tout dans `resultats`
- 6) Ne conserver que les observations des candidats élus
- 7) Joindre `pop_2019` à `resultats` et renommer `Population.20191 ... hab ..` en `pop`

Solution

1) Importer les données

```
resultats_t1 = read.csv("data/resultats-definitifs-par-circonscription-t1.csv", sep = ";", na.strings=c("", "NA"))
resultats_t2 = read.csv("data/resultats-definitifs-par-circonscription-t2.csv", sep = ";", na.strings=c("", "NA"))
pop_2019 = read.csv("data/population_2019.csv", sep = ";")
```

2) Effectuer sur chaque jeu de données les opérations nécessaires pour harmoniser l'encodage du libellé de circonscription

```
resultats_t1 = resultats_t1 %>%
  mutate(Libellé.circonscription.législative = gsub("\\\\D", "", Libellé.circonscription.législative))

resultats_t2 = resultats_t2 %>%
  mutate(Libellé.circonscription.législative = gsub("\\\\D", "", Libellé.circonscription.législative))

pop_2019 = pop_2019 %>%
  separate(Circonscriptions, into = c("Libellé.département", "Libellé.circonscription.législative"), sep = ", ") %>%
  mutate(Libellé.circonscription.législative = gsub("\\\\D", "", Libellé.circonscription.législative))
```

Solution

3) Reformatter `resultats_t1` et `resultats_t2` en format long

4) Ajouter une variable `tour` qui vaut 1 dans `resultats_t1` et 2 dans `resultats_t2`

```
resultats_t1_long = resultats_t1 %>%  
  pivot_longer(  
    cols = matches(".*\\.[1-9]$|.*\\.[1][0-9]$"),  
    names_to = c(".value", "candidate_number"),  
    names_pattern = "(.*)(\\d+)"  
  ) %>%  
  mutate(tour = 1)
```

Explication:

- `cols = matches(".*\\.[1-9]$|.*\\.[1][0-9]$")`: Sélectionne les colonnes dont le nom termine par un point suivi d'un nombre de 1 à 19

Solution

3) Reformatter `resultats_t1` et `resultats_t2` en format long

4) Ajouter une variable `tour` qui vaut 1 dans `resultats_t1` et 2 dans `resultats_t2`

```
resultats_t1_long = resultats_t1 %>%  
  pivot_longer(  
    cols = matches(".*\\.[1-9]$|.*\\.[1][0-9]$"),  
    names_to = c(".value", "candidate_number"),  
    names_pattern = "(.*\\.(\\d+))"  
  ) %>%  
  mutate(tour = 1)
```

Explications:

- `names_to = c(".value", "candidate_number")`: définit le **nom** des nouvelles variables
 - `.value` crée de nouvelles variables dont le nom correspond à ce qui précède le point pour chaque variable ayant un suffixe numérique
 - `candidate_number` crée une nouvelle colonne qui contient les suffixes numériques (1 à 19) indiquant le numéro du candidat

Solution

3) Reformatter `resultats_t1` et `resultats_t2` en format long

4) Ajouter une variable `tour` qui vaut 1 dans `resultats_t1` et 2 dans `resultats_t2`

```
resultats_t1_long = resultats_t1 %>%  
  pivot_longer(  
    cols = matches(".*\\.[1-9]$|.*\\.[1][0-9]$"),  
    names_to = c(".value", "candidate_number"),  
    names_pattern = "(.*\\.(\\d+))"  
  ) %>%  
  mutate(tour = 1)
```

Explications:

- `names_pattern = "(.*\\.(\\d+))"`: utilise une expression régulière pour **séparer** les noms de colonnes en deux parties:
 - `(.*)` capture tout ce qui précède le point et l'assigne à `.value`
 - `(\\d+)` capture les nombres après le point et les assigne à `candidate_number`

Solution

3) Reformatter `resultats_t1` et `resultats_t2` en format long

4) Ajouter une variable `tour` qui vaut 1 dans `resultats_t1` et 2 dans `resultats_t2`

On reproduit les mêmes opérations sur `resultats_t2` (sauf que cette fois-ci il y a au maximum 4 candidats par circonscription)

```
resultats_t2_long = resultats_t2 %>%
  pivot_longer(
    cols = matches(".*\\.[1-4]$"),
    names_to = c(".value", "candidate_number"),
    names_pattern = "(.*\\\\.(\\\\d+)"
  ) %>%
  mutate(tour = 2)
```

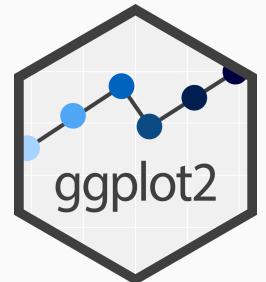
Solution

- 5) Ajouter les observations de `resultats_t2` à `resultats_t1` et stocker le tout dans `resultats`
- 6) Ne conserver que les observations des candidats élus
- 7) Joindre `pop_2019` à `resultats` et renommer `Population.20191 ... hab..` en `pop`

```
resultats = bind_rows(resultats_t1_long, resultats_t2_long) %>%  
  filter(Elu == "élu") %>%  
  left_join(pop_2019 %>% rename(pop = Population.20191 ... hab..), by = c("Libellé.département", "Libellé.circonscripti
```

2. R: rappels

2.5. Visualisation des données avec ggplot2



ggplot2 est un package R qui permet de **visualiser des données** sous forme de graphiques.

Afin de *chaîner* des opérations, c'est à dire d'**ajouter des couches au graphique, on utilise les +** (l'équivalent de %>% avec dplyr).

Pour construire n'importe quel graphique, il faut:

- une base de données
- lier les variables aux éléments graphiques (axes, couleurs, tailles) via l'argument aes()
- définir un type de graphique (ligne, point, histogramme, densité, etc)

Structure de base

```
ggplot(data,  
        aes(x = x, y = y)) +  
geom_point()
```

2. R: rappels

2.5. Visualisation des données avec `ggplot2`

Principales fonctions

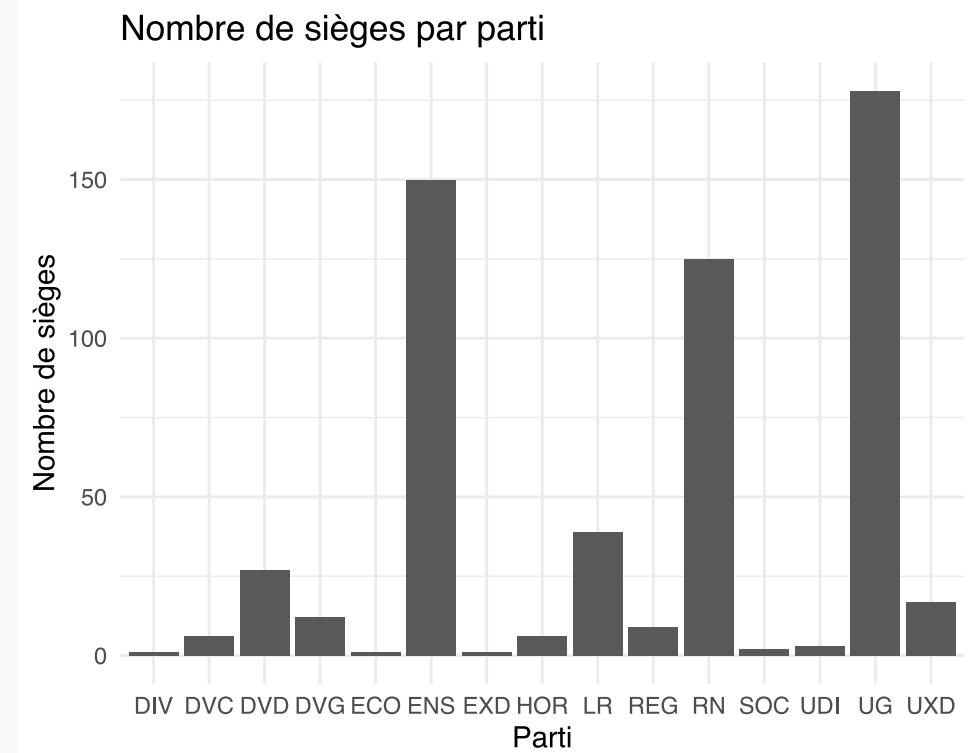
- `ggplot()` : Initialiser un graphique en spécifiant les données et les aesthetics
- `geom_` : Représente les types de graphiques:
 - `geom_point()` pour un nuage de points
 - `geom_line()` pour des lignes
 - `geom_bar()` pour un histogramme/barplot
 - `geom_density()` pour une densité
- `labs()` : Ajouter des titres et légendes
- `theme()` : Personnaliser l'apparence du graphique

2. R: rappels

2.5. Visualisation des données avec `ggplot2`

Variable discrète

```
ggplot(resultats,  
       aes(x = Nuance.candidat)) +  
  geom_bar() +  
  labs(  
    title = "Nombre de sièges par parti",  
    x = "Parti",  
    y = "Nombre de sièges") +  
  theme_minimal()
```



2. R: rappels

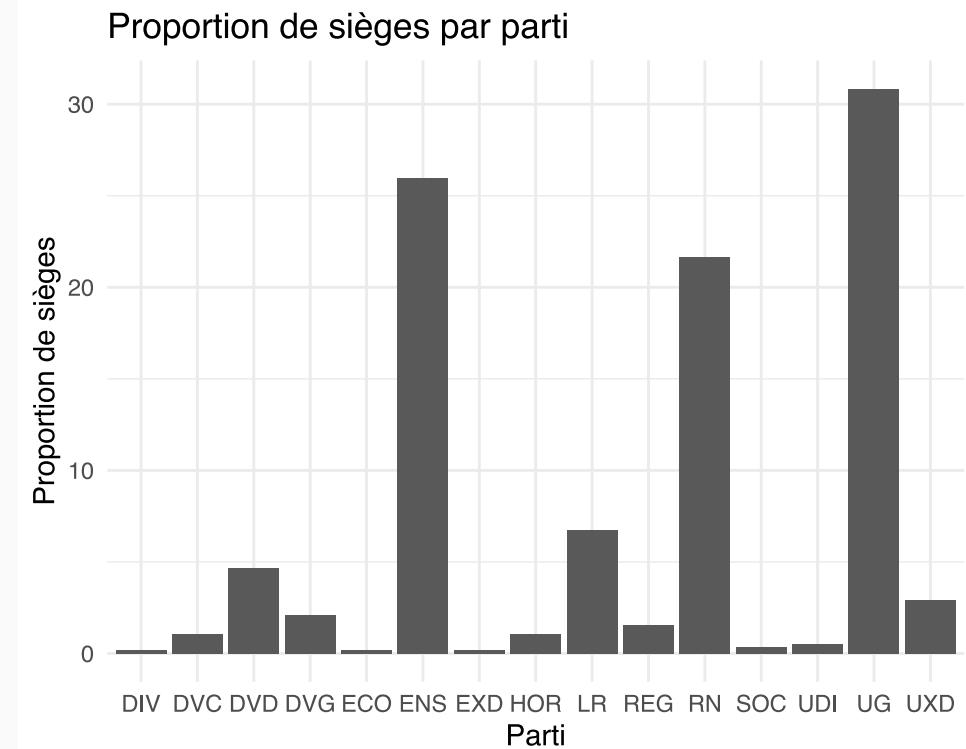
2.5. Visualisation des données avec ggplot2

Variable discrète

```
resultats %>%
  count(Nuance.candidat) %>%
  mutate(percentage = n/sum(n)*100) %>%
  ggplot(aes(x = Nuance.candidat,
              y = percentage)) +
  geom_bar(stat = "identity") +
  labs(
    title = "Proportion de sièges par parti",
    x = "Parti",
    y = "Proportion de sièges") +
  theme_minimal()
```

La fonction `count()` compte le nombre d'observation par groupe `Nuance.candidat` et stocke cette information dans une nouvelle variable, `n` (équivaut à

```
group_by(Nuance.candidat) %>% summarize(n = n())
```

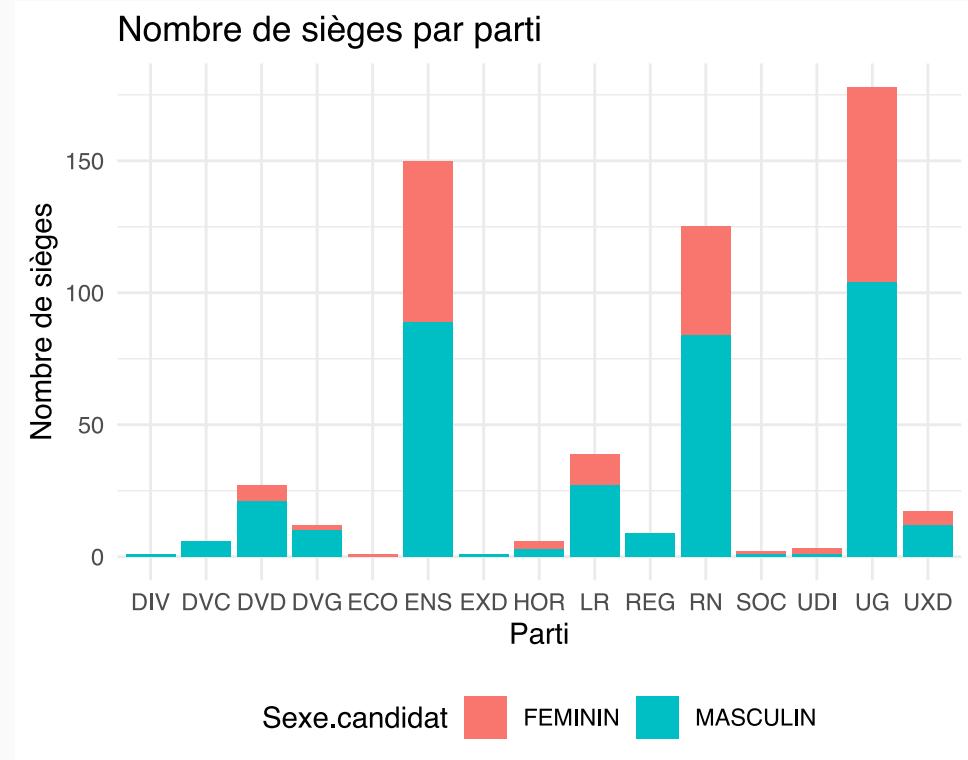


2. R: rappels

2.5. Visualisation des données avec ggplot2

Variable discrète

```
ggplot(resultats,  
       aes(x = Nuance.candidat, fill = Sexe.candidat)) +  
  geom_bar() +  
  labs(  
    title = "Nombre de sièges par parti",  
    x = "Parti",  
    y = "Nombre de sièges") +  
  theme_minimal() +  
  theme(legend.position="bottom")
```

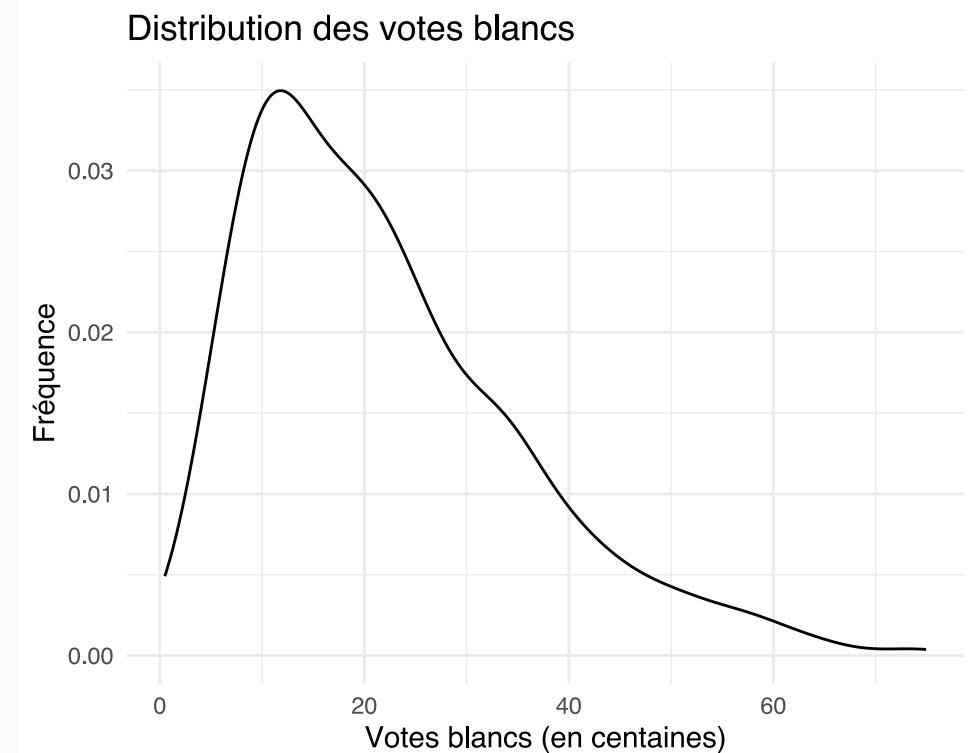


2. R: rappels

2.5. Visualisation des données avec ggplot2

Variable continue

```
resultats %>%
  mutate(Blancs = as.numeric(Blancs)/100) %>%
  ggplot(aes(x = Blancs)) +
  geom_density() +
  labs(
    title = "Distribution des votes blancs",
    x = "Votes blancs (en centaines)",
    y = "Fréquence") +
  theme_minimal()
```

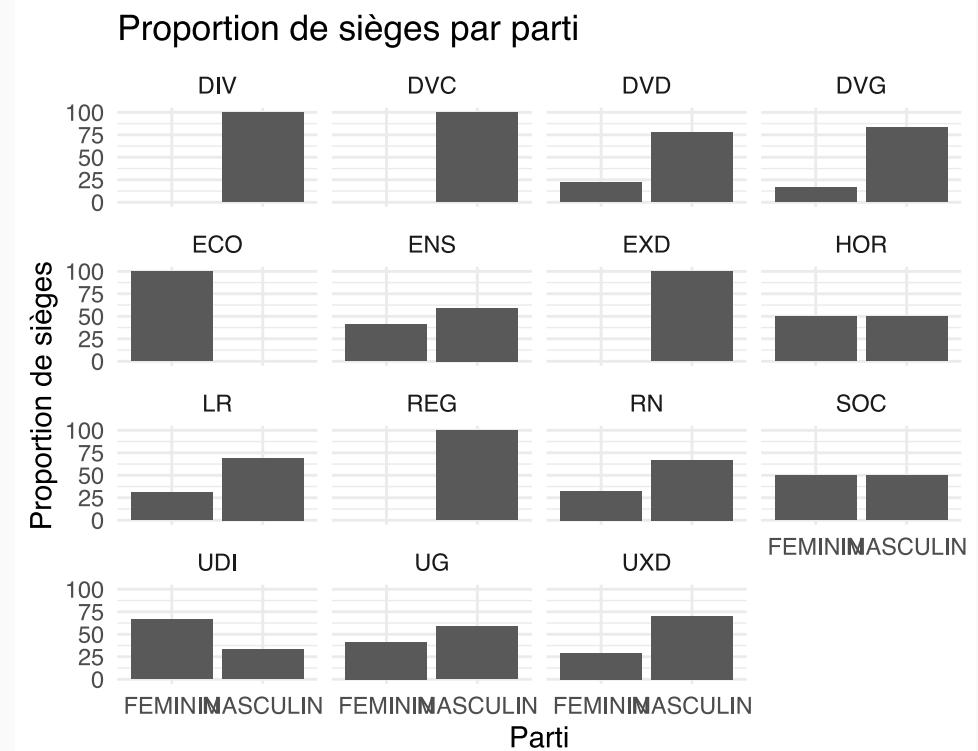


2. R: rappels

2.5. Visualisation des données avec ggplot2

facet_wrap

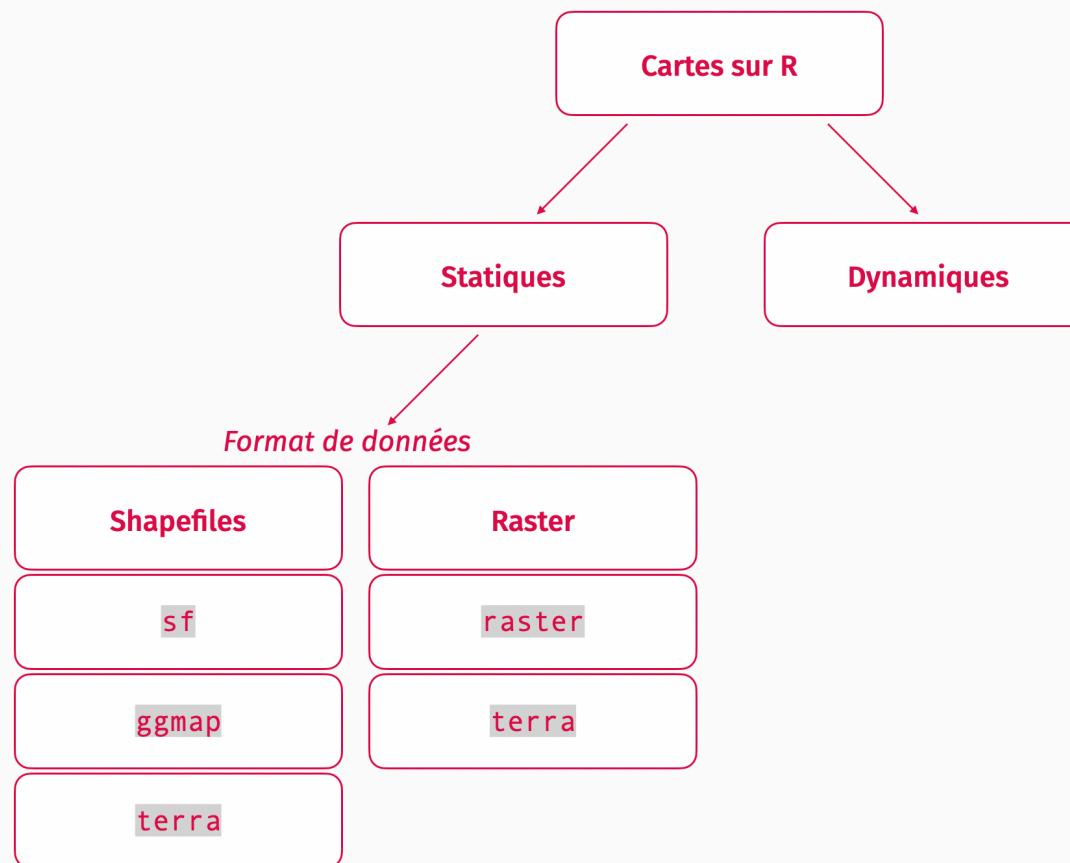
```
resultats %>%
  group_by(Nuance.candidat) %>%
  count(Sexe.candidat) %>%
  mutate(percentage = n/sum(n)*100) %>%
  ggplot(aes(x = Sexe.candidat,
             y = percentage)) +
  geom_bar(stat = "identity") +
  labs(
    title = "Proportion de sièges par parti",
    x = "Parti",
    y = "Proportion de sièges") +
  facet_wrap(~ Nuance.candidat) +
  theme_minimal()
```



2. R: rappels

2.6. Visualisation des données spatiales

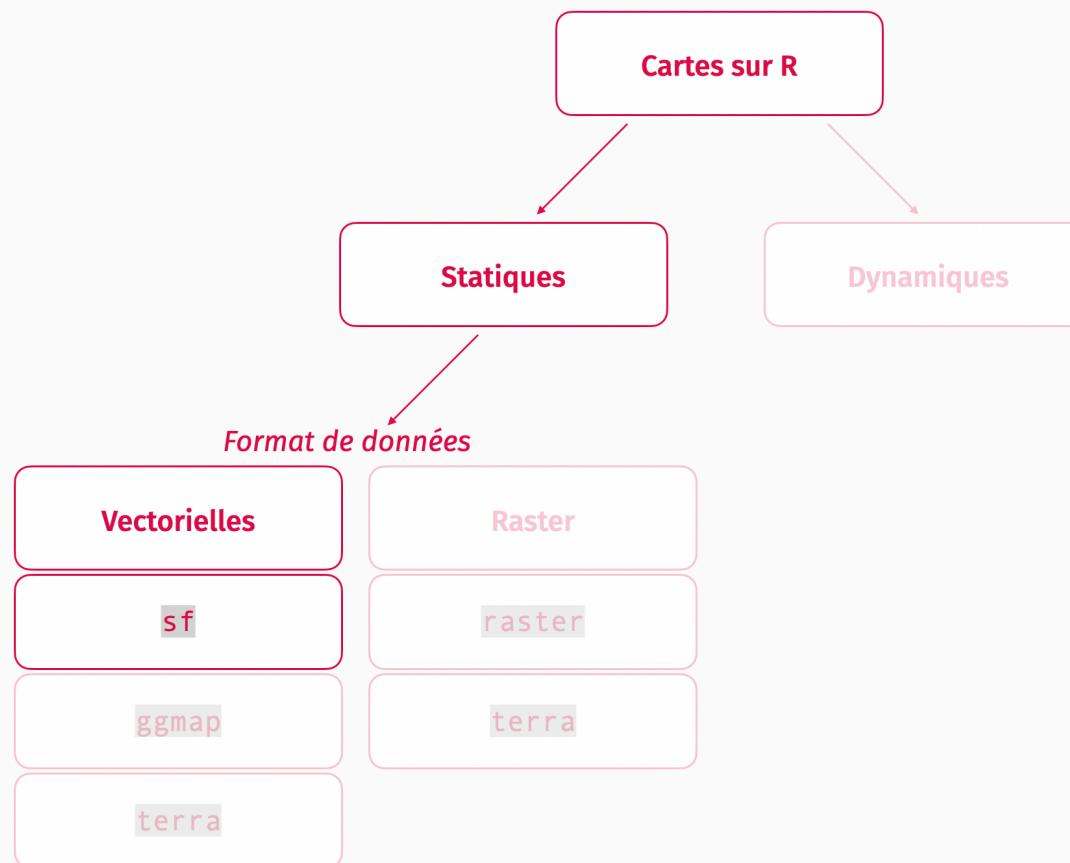
Format de cartes et données



2. R: rappels

2.6. Visualisation des données spatiales

Format de cartes et données



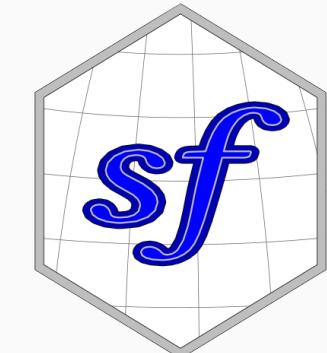
2. R: rappels

2.6. Visualisation des données spatiales

Package `sf`

Shapefile: format de fichier géospatial utilisé pour représenter des entités géographiques

- contient des informations sur la forme (polygones, points, lignes) et des attributs.
- composé de plusieurs fichiers `.shp`, `.shx`, `.dbf`, tous nécessaires pour la réalisation de cartes



Une fois importés sur `R`, les objets `sf` sont des `data.frame` dont l'une des colonnes contient des géométries.

```
head(map[ , -c(5,6)],3)
```

```
## Simple feature collection with 3 features and 4 fields
## Geometry type: MULTIPOLYGON
## Dimension: XY
## Bounding box: xmin: -0.595231 ymin: 44.80615 xmax: 5.889757 ymax: 50.7899
## Geodetic CRS: WGS 84
##      ID Code.département code_reg nom_dpt           geometry
## 1 33004            33      75 GIRONDE MULTIPOLYGON (((-0.454946 4 ...
## 2 38001            38      84  ISERE MULTIPOLYGON (((5.805288 45 ...
## 3 59010            59      32    NORD MULTIPOLYGON (((3.058745 50 ...
```

2. R: rappels

2.6. Visualisation des données spatiales

Réaliser une carte en 3 étapes

1. Importer les données

- Le jeu de données qui contient l'information à visualiser
- Les fichiers `shapefiles` qui contiennent le fond de carte, etc, avec la fonction `st_read()`
- 🚫: on importe sur `R` uniquement le fichier `.shp`, mais tous les autres fichiers (`.shx`, `.dbf`, etc) téléchargés avec le *shapefile* doivent être localisés au même endroit que le `.shp`

2. Lier le jeu de données qui contient l'information à visualiser avec les données géographiques

- Pour ajouter de l'information à un fond de carte, il est nécessaire de disposer d'une **variable (key) géographique commune entre les deux fichiers**

3. Représenter la carte avec la fonction `geom_sf()` sur le même principe que `geom_line()` par exemple avec `ggplot2`

- Ici, l'esthétique du graphique (`aes`) comprend un nouvel argument, `geometry`, qui doit être égal à la variable contenant les données géographiques

Application: carte résultats législatives

10:00

Importer les données

- 1) Importer `france-circonscriptions-legislatives-2012.shp`, que vous stockerez dans `maps`

Cleaning

- 2) Renommer `code_dpt` en `Code.département` et `num_circ` en `Libellé.circonscription`
- 3) Modifier la variable `Code.département`: si le premier caractère est un `0`, `Code.département` prend la valeur du deuxième caractère. Sinon pas de modification

Joindre les bases de données

- 4) Ajouter les informations de `map` à `resultats` et stocker le tout dans `resultats_map`

Réaliser une carte

- 5) Réaliser la carte des résultats des élections législatives de 2024

NB: ne pas sélectionner les départements 971, 972, 973, 974, 975, 976, 986, 987, 988, ZX, ZZ qui correspondent aux DROM et qui ne sont pas sur ce fond de carte

Solution

Importer les données

1) Importer `france-circonscriptions-legislatives-2012.shp`, que vous stockerez dans `maps`

Cleaning

2) Renommer `code_dpt` en `Code.département` et `num_circ` en `Libellé.circonscription`

3) Modifier la variable `Code.département`: si le premier caractère est un `0`, `Code.département` prend la valeur du deuxième caractère. Sinon pas de modification

```
# Import data
map = st_read("data/france-circonscriptions-legislatives-2012.shp", quiet = TRUE) %>%
  rename(Code.département = code_dpt,
         Libellé.circonscription.législative = num_circ) %>%
  mutate(Code.département = ifelse(substr(Code.département, 1, 1) == "0", substr(Code.département, 2, 2), Code.département))
```

Solution

Importer les données

1) Importer `france-circonscriptions-legislatives-2012.shp`, que vous stockerez dans `maps`

Cleaning

2) Renommer `code_dpt` en `Code.département` et `num_circ` en `Libellé.circonscription`

3) Modifier la variable `Code.département`: si le premier caractère est un `0`, `Code.département` prend la valeur du deuxième caractère. Sinon pas de modification

```
# Import data
map = st_read("data/france-circonscriptions-legislatives-2012.shp", quiet = TRUE) %>%
  rename(Code.département = code_dpt,
         Libellé.circonscription.législative = num_circ) %>%
  mutate(Code.département = ifelse(substr(Code.département, 1, 1) == "0", substr(Code.département, 2, 2), Code.département))
```

Solution

Importer les données

1) Importer `france-circonscriptions-legislatives-2012.shp`, que vous stockerez dans `maps`

Cleaning

2) Renommer `code_dpt` en `Code.département` et `num_circ` en `Libellé.circonscription`

3) Modifier la variable `Code.département`: si le premier caractère est un `0`, `Code.département` prend la valeur du deuxième caractère. Sinon pas de modification

```
# Import data
map = st_read("data/france-circonscriptions-legislatives-2012.shp", quiet = TRUE) %>%
  rename(Code.département = code_dpt,
         Libellé.circonscription.législative = num_circ) %>%
  mutate(Code.département = ifelse(substr(Code.département, 1, 1) == "0", substr(Code.département, 2, 2), Code.département))
```

Solution

Importer les données

1) Importer `france-circonscriptions-legislatives-2012.shp`, que vous stockerez dans `maps`

Cleaning

2) Renommer `code_dpt` en `Code.département` et `num_circ` en `Libellé.circonscription`

3) Modifier la variable `Code.département` : si le premier caractère est un `0`, `Code.département` prend la valeur du deuxième caractère. Sinon pas de modification

```
# Import data
map = st_read("data/france-circonscriptions-legislatives-2012.shp", quiet = TRUE) %>%
  rename(Code.département = code_dpt,
         Libellé.circonscription.législative = num_circ) %>%
  mutate(Code.département = ifelse(substr(Code.département, 1, 1) = "0", substr(Code.département, 2, 2), Code.département))
```

Solution

Joindre les bases de données

- 4) Ajouter les informations de `map` à `resultats` et stocker le tout dans `resultats_map`

```
# Merge geographic data and results data
resultats_map = resultats %>%
  left_join(map , by = c("Code.département", "Libellé.circonscription.législative"))
```

Solution

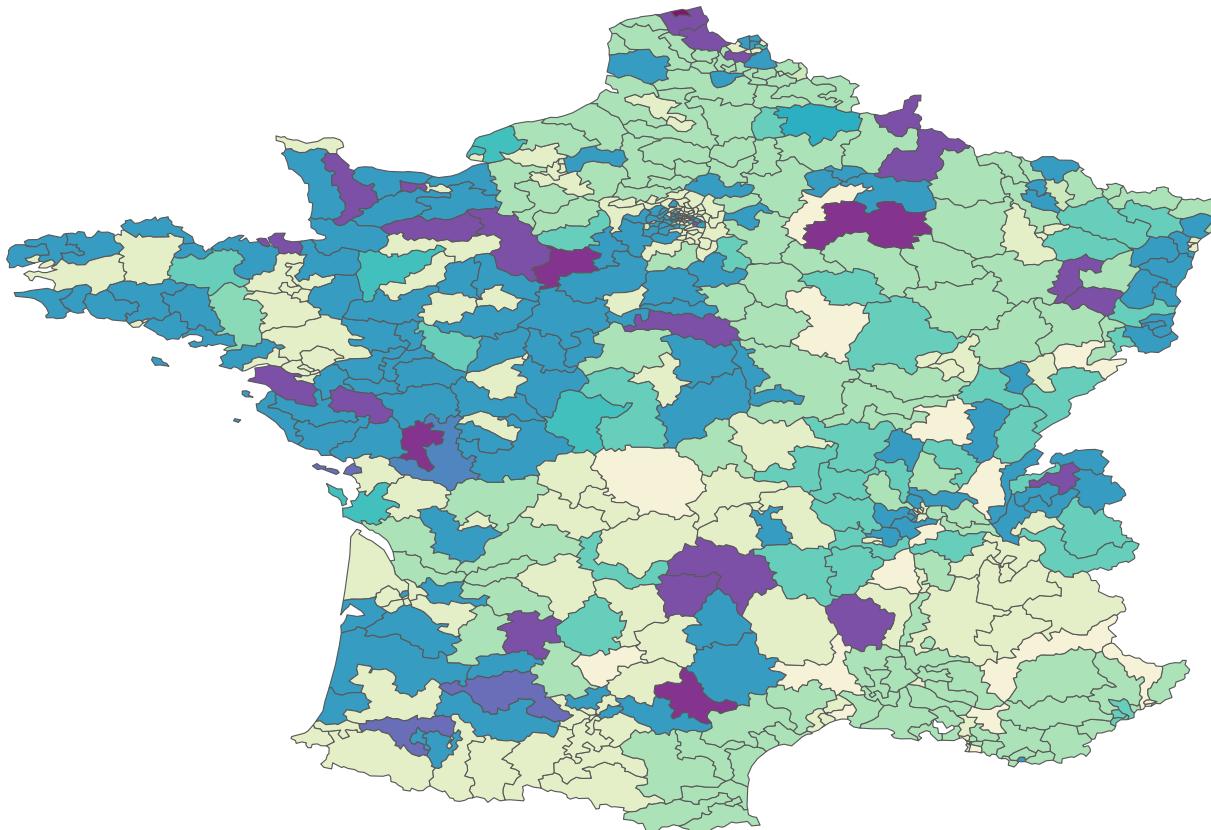
Réaliser une carte

5) Réaliser la carte des résultats des élections législatives de 2024

NB: ne pas sélectionner les départements 971, 972, 973, 974, 975, 976, 986, 987, 988, ZX, ZZ qui correspondent aux DROM et qui ne sont pas sur ce fond de carte

```
# Draw map
ggplot(resultats_map %>% filter(! (Code.département %in% c("971", "972", "973", "974", "975", "976", "986", "987", "988", "ZX", "ZZ")))
       geom_sf(aes(fill = Nuance.candidat, geometry = geometry)) +
       theme_minimal() +
       scale_fill_manual(values = hcl.colors(14, "Purple-Yellow")) +
       labs(title = "Résultats des élections législatives par circonscription",
            fill = "Parti") +
       theme(legend.position = "bottom",
             axis.text = element_blank(),
             axis.ticks = element_blank(),
             panel.grid = element_blank()) +
       guides(fill = guide_legend(nrow = 1))
```

Solution



Parti DIV DVC DVD DVG ECO ENS EXD HOR LR REG RN UDI UG UXD

Miscellaneous

Vos 4 meilleurs amis

1. La **commande** `help(.)` ou `?`.

Vos 4 meilleurs amis

1. La **commande** `help(.)` ou `?`.

2. Les ***cheatsheets***

- [Exemple](#)

Vos 4 meilleurs amis

1. La **commande** `help(.)` ou `?`

2. Les **cheatsheets**

- [Exemple](#)

3. **Internet** (stackoverflow):

- vous ne serez jamais la première personne à rencontrer les problèmes auxquels vous faites face
- copier/coller le message d'erreur peut suffire, mais essayez de comprendre par vous même d'abord

Vos 4 meilleurs amis

1. La **commande** `help(.)` ou `?`

2. Les **cheatsheets**

- [Exemple](#)

3. **Internet** (stackoverflow):

- vous ne serez jamais la première personne à rencontrer les problèmes auxquels vous faites face
- copier/coller le message d'erreur peut suffire, mais essayez de comprendre par vous même d'abord

4. **ChatGPT**: devient de plus en plus performant pour résoudre les problèmes de codes.

- ! souvent les packages ne sont pas à jour, les fonctions proposées inventées

Vos 4 meilleurs amis

1. La **commande** `help(.)` ou `?`

2. Les **cheatsheets**

- [Exemple](#)

3. **Internet** (stackoverflow):

- vous ne serez jamais la première personne à rencontrer les problèmes auxquels vous faites face
- copier/coller le message d'erreur peut suffire, mais essayez de comprendre par vous même d'abord

4. **ChatGPT**: devient de plus en plus performant pour résoudre les problèmes de codes.

- ⚠️ souvent les packages ne sont pas à jour, les fonctions proposées inventées

👉 Soyez autonomes: ne demander de l'aide à vos camarades (ou à moi) uniquement après avoir demandé à vos 4 meilleurs amis!

Bonnes pratiques

Organisation du travail

Un dossier pour chaque projet de recherche et différents sous-dossiers pour:

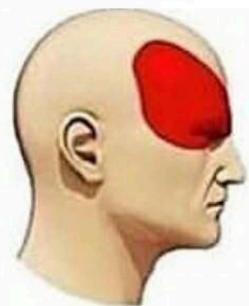
- Scripts
 - un scripts par tâche: cleaning, statistiques descriptives, analyse empirique
- Data
 - .rds
- Figures
- (Admin)
 - par exemple les documents relatifs à l'accès à une base de données

Commenter son code pour soi...¹

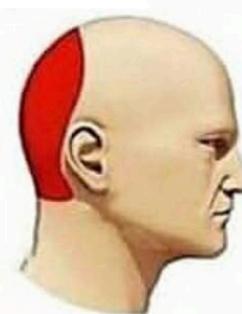
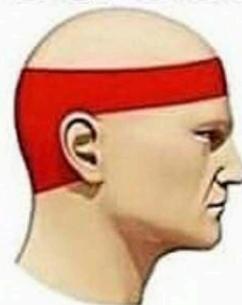


... et ses collègues¹

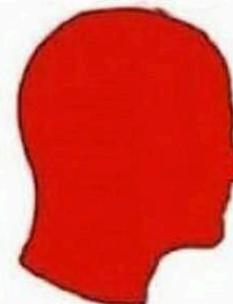
Types of Headaches



Stress



**READING SOMEONE ELSE'S
UNCOMMENTED CODE**



Essayer de ne pas s'énerver¹



L'apprentissage de R est long et fastidieux, mais les rendements sont croissants!

[1]: R Memes For Statistical Fiends

References

[Introduction à l'analyse d'enquêtes avec R et RStudio](#)

[Introduction to Econometrics & R Programming, Louis Sirugue](#)

[Econometrics, Florian Oswald](#)

[Causal inference: The Mixtape, Scott Cunningham](#)

[Géomatique sur R](#)

Appendix

Cheatsheets

- [R Studio Environment](#)
- [All cheatsheets](#)

RStudio IDE :: CHEATSHEET

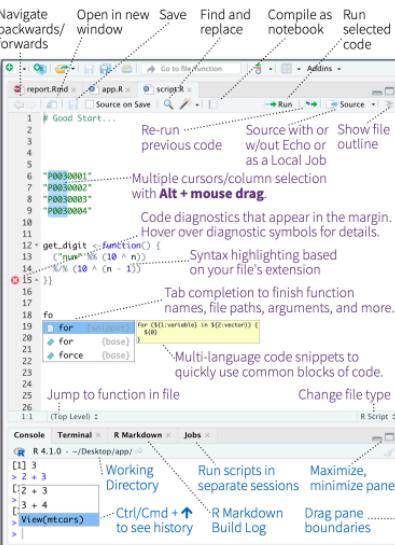
Documents and Apps



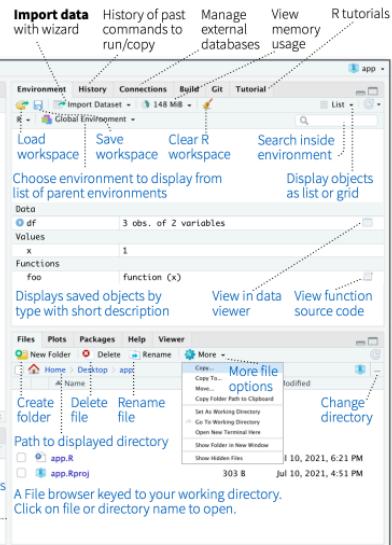
Package Development



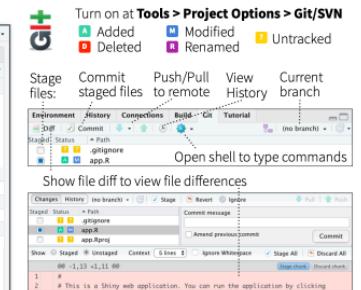
Source Editor



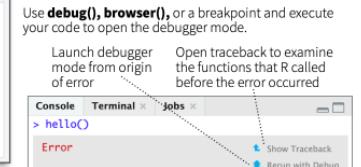
Tab Panes



Version Control



Debug Mode



[Back](#)