

# 01 - Boîte à outils

PRO1036 - Analyse de données scientifiques en R

Tim Bollé

September 8, 2025

# Boîte à outils

# Les outils

## Développement:

- R
- RStudio
- tidyverse
- R Markdown

## Gestion et collaboration:

- Git
- GitHub

# Objectif

# Objectif du cours

À la fin de ce cours, vous pourrez:

- Analyser des données
- Analyser des données de manière **répétable**
- Analyser des données de manière répétable, avec des **outils de programmation modernes**
- Analyser des données de manière répétable et **collaborative**, avec des outils de programmation modernes

# Répétabilité

| Que signifie conduire une analyse de donnée de manière répétable ?

À court-terme:

- Pouvons nous reproduire les tableaux et les figures à partir des données
- Est-ce que le code fait ce que nous voulons ?
- Pouvons-nous reconstruire pourquoi et comment nous avons obtenus les résultats

À long-terme:

- Peut-on réutiliser le code pour d'autres données ?
- Peut-on réutiliser le code pour faire autre chose ?

# Les outils de la répérabilité

*Scriptability* \(\rightarrow\) R

Documentation et communication \(\rightarrow\) R Markdown

Gestion et collaboration \(\rightarrow\) Git/GitHub

# R et RStudio



- R est un langage de programmation **open-source**
- R est un environnement pour faire des **calculs statistiques** et de la **visualisation**
- De nombreuses autres applications sont disponibles grâce à des ***packages***



- RStudio est un **IDE** (Environnement de Développement Intégré)
- C'est une interface pour R
- Pas nécessaire pour coder en R mais tellement pratique !



# R packages

Les packages sont les *building blocks* de la reproductibilité. Ils contiennent de nombreuses fonctions réutilisables, de la documentation et données de test ([Wickham and Bryan, 2023](#))

Nous allons en utiliser quelques une mais vous verrez que c'est tout une philosophie !

# RStudio tour

The screenshot displays the RStudio environment with the following components:

- Top Panel:** Menu bar (File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help) and a toolbar with icons for file operations and navigation.
- Source Editor:** Displays a data table named 'penguins' with columns: species, island, bill\_length\_mm, bill\_depth\_mm, flipper\_length\_mm, body\_mass\_g, sex, and year. The table shows 15 rows of data for Adelie penguins from Torgersen island.
- Environment Pane:** Shows the 'Global Environment' with a variable 'x' containing the value 2.
- Console:** Shows R code execution:
 

```
> 2 + 2
[1] 4
> x <- 2
> x * 3
[1] 6
> library(palmerpenguins)
> view(penguins)
> mean(penguins$flipper_length_mm)
[1] NA
> ?mean
> mean(penguins$flipper_length_mm, na.rm = TRUE)
[1] 200.9152
> penguins$flipper_length_mm
[1] 181 186 195 NA 193 190 181 195 193 190 186 180 182 191 198 185 195 197 184 194 174 180
[23] 189 185 180 187 183 187 172 180 178 178 188 184 195 196 190 180 181 184 182 195 186 196
```
- Documentation Pane:** Displays the R documentation for the 'mean' function, including the title 'Arithmetic Mean', a description 'Generic function for the (trimmed) arithmetic mean.', usage examples, and arguments 'x' and 'trim'.

# R 101

Les **fonctions** sont souvent des verbes, suivi de parenthèses, contenant des arguments:

```
1 fait_ca(avec_ca)
2 fait_ca(avec_ca, et_ca, et_encore_ca)
```

Les packages peuvent être installés avec **install.package** et chargés avec **library**:

```
1 install.packages("package_name")
2 library(package_name)
```

**\$** permet d'accéder aux colonnes des tableaux

```
1 dataframe$var_name
```

**?** permet d'accéder à l'aide sur les fonctions

```
1 ?mean
```

# Tidyverse



[tidyverse.org](https://tidyverse.org)

Le **Tidyverse** est une collection de packages développés pour faire de la data science

Il y a une philosophie et une grammaire commune à tous ces packages, que nous allons apprendre.

# R Markdown

[rmarkdown.rstudio.com](https://rmarkdown.rstudio.com)

**R Markdown** permet d'écrire des documents avec du code intégré (extension en **.Rmd**).

Va permettre de documenter et de communiquer directement nos analyses de données !

- Reproductible: À chaque fois qu'on génère le document, tout est exécuté depuis le début
- Syntaxe simple pour avoir des documents de qualité
- Le document se découpe en zones de texte et blocks de code



# R Markdown

The image shows a side-by-side comparison of an R Markdown document in its source and rendered states. The left pane displays the source code, which includes a YAML header, a title, author information, and R code for loading packages and filtering a dataset. The right pane shows the rendered HTML output, which formats the code as preformatted text and renders the R code blocks as executable snippets.

**Source (Left Pane):**

```

1 ---
2 title: "Bechdel"
3 author: "Mine Çetinkaya-Rundel"
4 format: html
5 editor: visual
6 ---
7
8 In this mini analysis we work with the data used in the FiveThirtyEight story
9 titled ["The Dollar-And-Cents Case Against Hollywood's Exclusion of
10 Women"](https://fivethirtyeight.com/features/the-dollar-and-cents-case-against-
11 t-hollywoods-exclusion-of-women/). Your task is to fill in the blanks denoted
12 by `___`.
13
14 ## Data and packages
15 We start with loading the packages we'll use.
16
17 ```{r}
18 #/ label: load-packages
19 #/ warning: false
20 #/ message: false
21
22 library(fivethirtyeight)
23 library(tidyverse)
24
25 The dataset contains information on `r nrow(bechdel)` movies released between
26 `r min(bechdel$year)` and `r max(bechdel$year)`. However we'll focus our
27 analysis on movies released between 1990 and 2013.
28
29 ```{r}
30 bechdel190_13 <- bechdel %>%
31   filter(between(year, 1990, 2013))
32
33 There are `___` such movies.
34
35 The financial variables we'll focus on are the following:
  
```

**Rendered (Right Pane):**

## Bechdel

AUTHOR  
Mine Çetinkaya-Rundel

In this mini analysis we work with the data used in the FiveThirtyEight story titled ["The Dollar-And-Cents Case Against Hollywood's Exclusion of Women"](https://fivethirtyeight.com/features/the-dollar-and-cents-case-against-t-hollywoods-exclusion-of-women/). Your task is to fill in the blanks denoted by `___`.

### Data and packages

We start with loading the packages we'll use.

```
library(fivethirtyeight)
library(tidyverse)
```

The dataset contains information on 1794 movies released between 1970 and 2013. However we'll focus our analysis on movies released between 1990 and 2013.

```
bechdel190_13 <- bechdel %>%
  filter(between(year, 1990, 2013))
```

There are `___` such movies.

The financial variables we'll focus on are the following:

# R Markdown - Aide

Cheatsheet

Help > Cheatsheet

Markdown Quick Reference

Help > Markdown Quick Reference

## R Markdown :: CHEAT SHEET

### What is R Markdown?

**Rmd files** - An R Markdown (.Rmd) file is a record of your research. It contains the code that a scientist needs to reproduce your work along with the narration that a reader needs to understand your work.

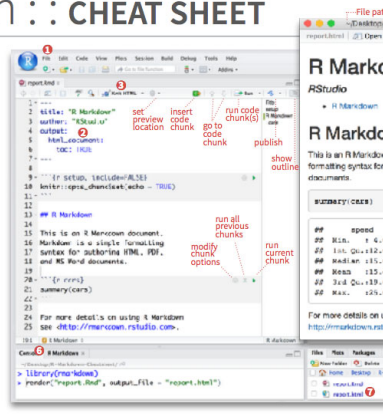
**Reproducible Research** - At the click of a button, or the type of a command, you can rerun the code in an R Markdown file to reproduce your work and export the results as a finished report.

**Dynamic Documents** - You can choose to export the finished report in a variety of formats, including html, pdf, MS Word or RTF documents, html or pdf based slides, Notebooks, and more.

### Workflow



- 1 Open a new .Rmd file at File > New File > R Markdown. Use the wizard that opens to pre-populate the file with a template
- 2 Write document by editing template
- 3 Knit document to create report; use knit button or `render()` to knit
- 4 Preview Output in IDE window
- 5 Publish (optional) to web server
- 6 Examine build log in R Markdown console
- 7 Use output file that is saved along side .Rmd



### render

Use `markdown::render()` to render/knit at cmd line. Important args:

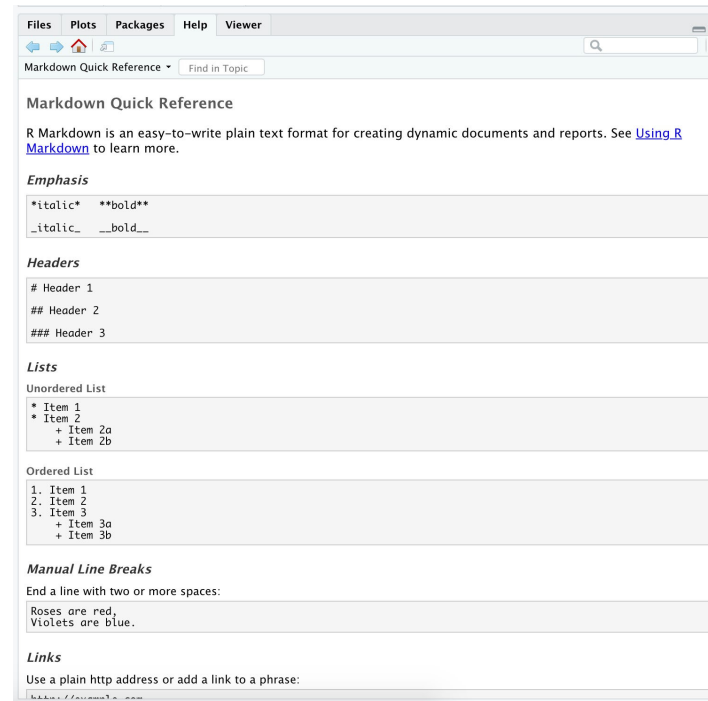
input_file to render	output_options - List of render options (as in YAML)	output_file output_dir	params - list of params to use

### Embed code with knitr syntax

**INLINE CODE**  
Insert with `"r<code>"`. Results appear as text without code.  
Built with `"r getRversion()"` Built with 3.2.3

**CODE CHUNKS**  
One or more lines surrounded with ````[r]<code>````. Place chunk options within curly braces, after `r`. Insert with ````[echo=TRUE] getRversion()````

**GLOBAL OP!**  
Set with `knitr::opts_chunk$set(include=FALSE)`



# Boîte à outils



# Les outils

## Développement:

- R
- RStudio
- tidyverse
- R Markdown

## Gestion et collaboration:

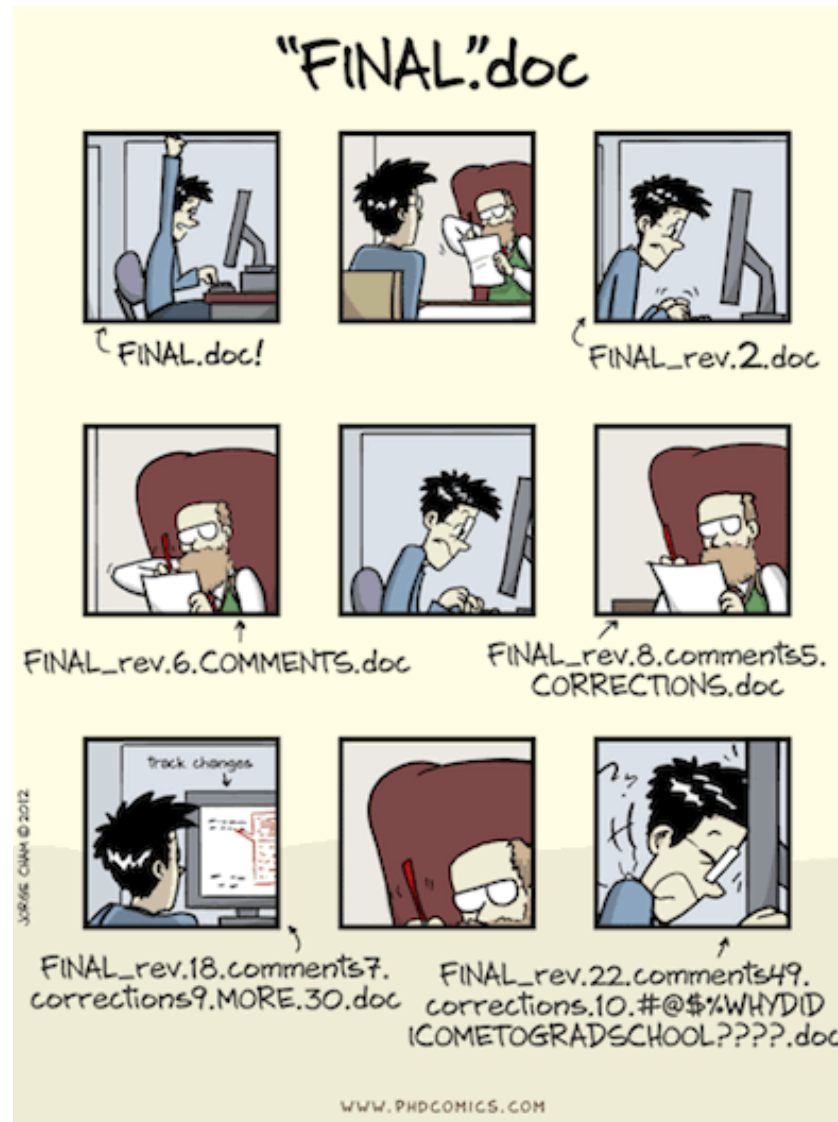
- Git
- GitHub

# Git et GitHub

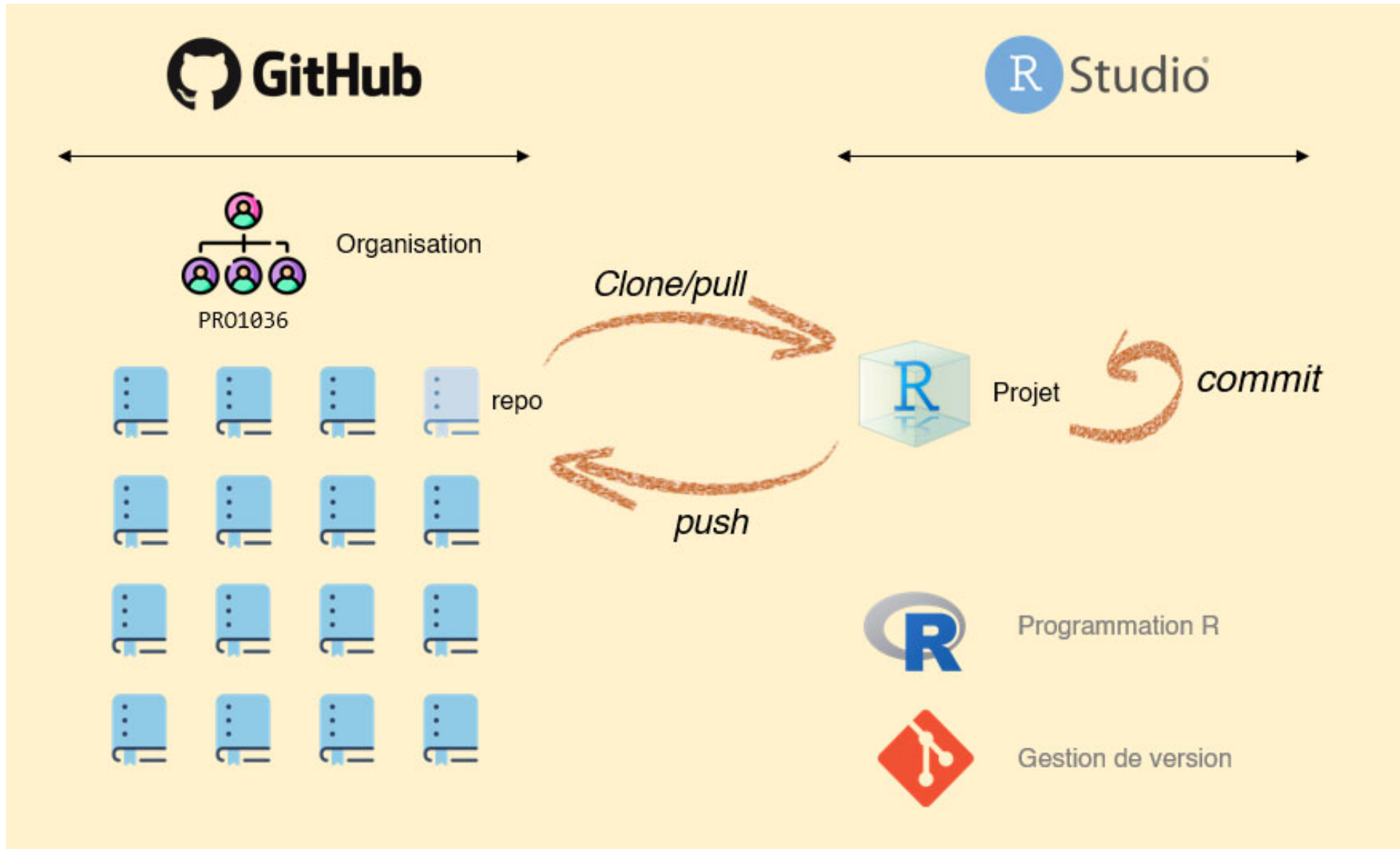


- Git est un outil de **gestion de version**
  - Comme le *track changes* sur Word
- Très populaire dans le monde de la programmation
- GitHub est un plateforme de stockage de **repo** Git
  - Comme un Onedrive/Dropbox pour Git
- Nous allons essayer de l'utiliser pour... tout !

# Pourquoi la gestion de version ?



# Fonctionnement



# Mise en place

Git peut être utilisé depuis le terminal de commande

- Utilisation plus avancée
- Nous pouvons normalement tout faire depuis R Studio

Github:

- Créez un compte avec votre adresse UQTR
- Vérifiez votre adresse courriel

# À vous de jouer !



# Votre première analyse de données

## UN Votes

Pour le faire chez vous:

- Ouvrez RStudio
- Nouveau Projet > Version Control > Git
- Entrez l'url: <https://github.com/PRO1036/unvotes>
- Choisissez un dossier sur l'ordinateur
- Ouvrez le fichier `unvotes.rmd`
- Validez l'installation des packages.
- Cliquez sur `knit`

# Références

Wickham, H. and Bryan, J. (2023). *R Packages: Organize, Test, Document, and Share Your Code* (2nd edition). O'Reilly Media.