

Science Reproductible avec un grand R

slides : tinyurl.com/ycktp4kv

Maxime Jaunatre
UR LESSEM | INRAE
2022-03-17 (updated: 2023-01-23)

Objectif : bonnes pratiques en R

 : Construit par et pour des statisticiens.

- Majorité d'utilisateurs non informaticiens.
- Bonne pratiques de *software engineering* diffusent dans la communauté.

Objectif : bonnes pratiques en R

R : Construit par et pour des statisticiens.

- Majorité d'utilisateurs non informaticiens.
- Bonne pratiques de *software engineering* diffusent dans la communauté.

Formation personnelle : Apprentissage de R pour l'**écologie**, traitements de données.

Bonnes pratiques viennent des échanges, de la communauté et de la pratique.



Objectif : bonnes pratiques en R

R : Construit par et pour des statisticiens.

- Majorité d'utilisateurs non informaticiens.
- Bonne pratiques de *software engineering* diffusent dans la communauté.

Formation personnelle : Apprentissage de R pour l'**écologie**, traitements de données.

Bonnes pratiques viennent des échanges, de la communauté et de la pratique.



On ne va pas parler de statistiques aujourd'hui.

Reproductibilité ?

Pouvoir reproduire une analyse, une figure ou des données efficacement !

Principe FAIR :

- **F**indable
- **A**ccessible
- **I**nteroperable
- **R**eusable

Reproductibilité ?

Pouvoir reproduire une analyse, une figure ou des données efficacement !

Principe FAIR :

- Findable
- Accessible
- Interoperable
- Reusable



Reproductibilité ?

Pouvoir reproduire une analyse, une figure ou des données efficacement !

Principe FAIR :

- Findable
- Accessible
- Interoperable
- Reusable

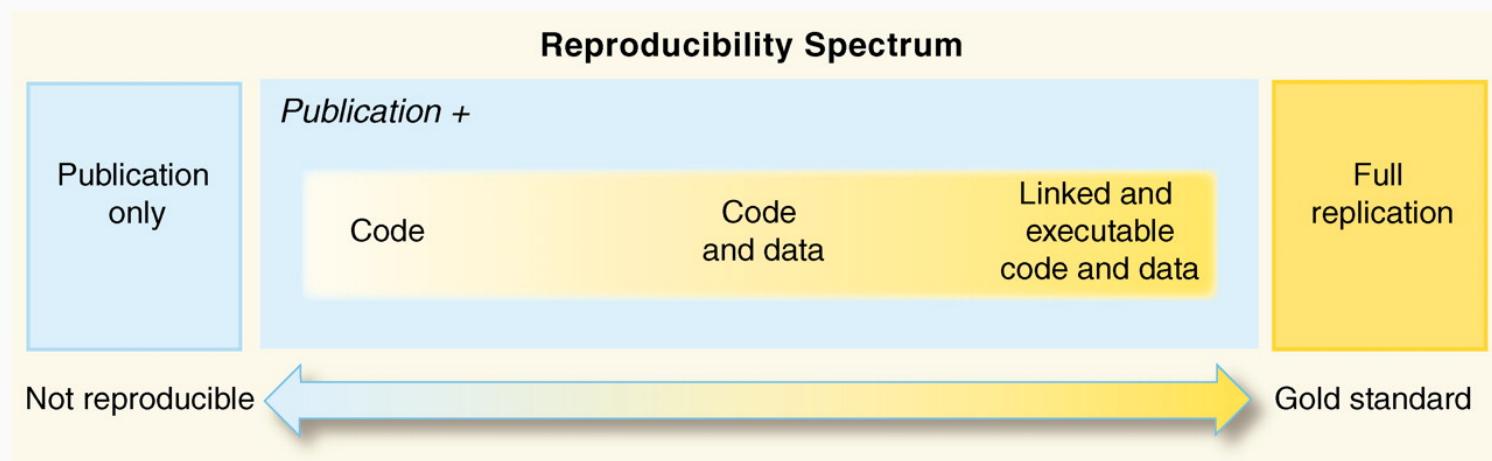
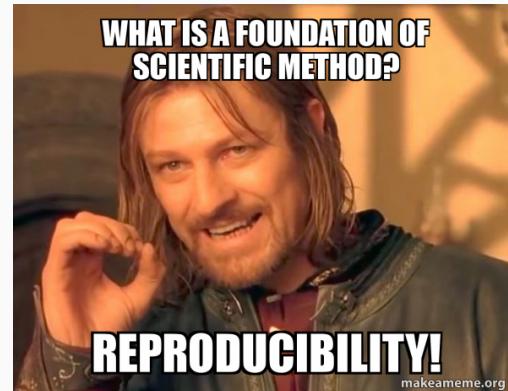


Figure : Peng. "Reproducible Research in Computational Science". In: *Science* (2011).

Findable

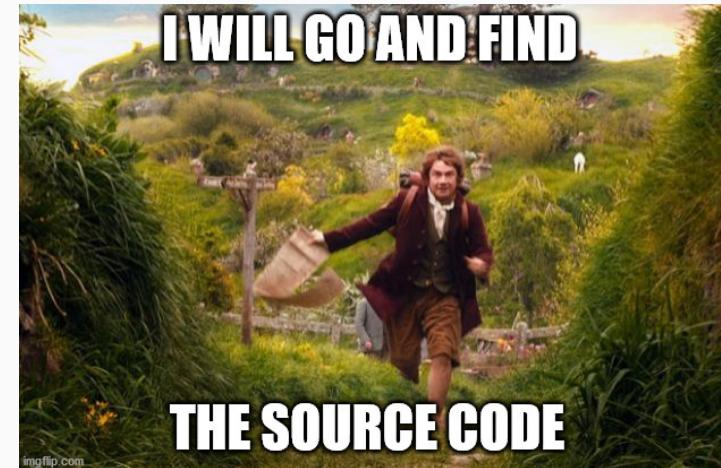


Un donnée ou une figure non liées au code qui l'a produit n'est pas reproductible.

L'emplacement du code doit être explicite. Le trouver ne doit pas être une aventure.

Cela passe par quelques conseils :

- Noms de scripts explicites.
(on évite les noms comme **test, test2**)
- Structure de dossier, **Compendium**
- Disque réseau et/ou en ligne
- Dossier avec noms de **projets**
(sans espaces et accents !)



- Logiciel de control de version (**git**)

```
list.files()[1:4]
```

```
## [1] "CSU_RepRo_files" "CSU_RepRo.html"   "CSU_RepRo.pdf"    "CSU_RepRo.Rmd"
```

Pourquoi utiliser un CVS ?

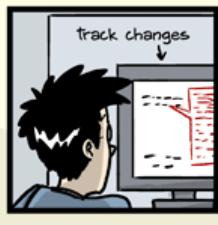
Est-ce que vous avez déjà :

- **Fait un changement de code et voulu revenir en arrière ?**
- **Perdu du code ou une sauvegarde trop ancienne ?**
- Voulu voir la différence entre 2 versions ?
- Voulu vérifier l'historique d'un script ?
- **Voulu travailler sur un script à plusieurs ?**
- **Voulu partager votre code à quelqu'un ?**
- Voulu tester une nouveauté sans modifier du code déjà utile ?

Si oui, et dans plein de cas, un système de version control aurait pu vous simplifier la vie.

Pourquoi utiliser un CVS ?

"FINAL".doc

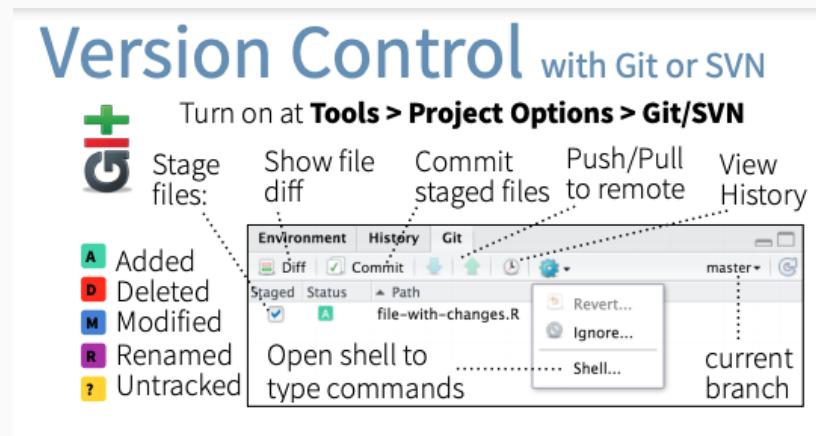


JORGE CHAM © 2012

Git : comment ça marche ?

3 commandes à retenir : commit, push, and pull

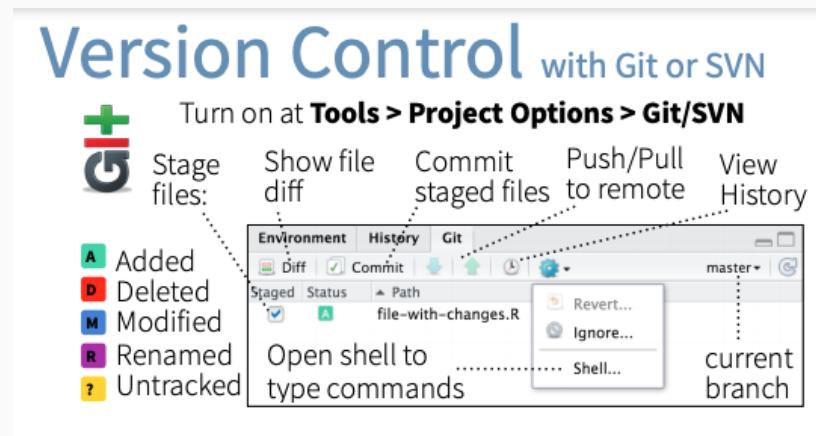
- commit: enregistre une snapshot du code à un certain point temporel.
(permet d'associer du texte pour expliquer les modifications)



Git : comment ça marche ?

3 commandes à retenir : commit, push, and pull

- commit: enregistre une snapshot du code à un certain point temporel.
(permet d'associer du texte pour expliquer les modifications)



- pull: met à jour le projet local avec la dernière version du projet*
- push: met en commun les modifications locales avec le projet principal*

[*] Si hébergé en ligne.

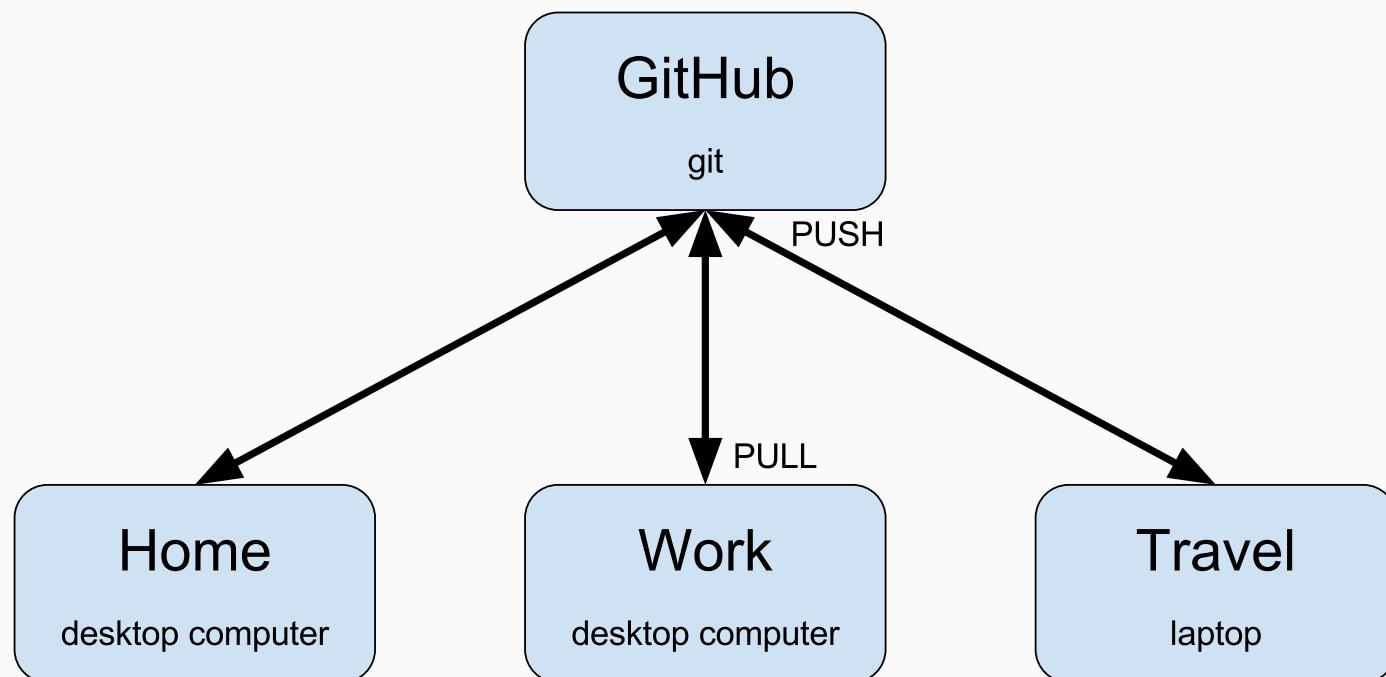
Git : comment ça marche ?

	COMMENT	DATE
O	CREATED MAIN LOOP & TIMING CONTROL	14 HOURS AGO
O	ENABLED CONFIG FILE PARSING	9 HOURS AGO
O	MISC BUGFIXES	5 HOURS AGO
O	CODE ADDITIONS/EDITS	4 HOURS AGO
O	MORE CODE	4 HOURS AGO
O	HERE HAVE CODE	4 HOURS AGO
O	AAAAAAA	3 HOURS AGO
O	ADKFJSLKDFJSDFKLJ	3 HOURS AGO
O	MY HANDS ARE TYPING WORDS	2 HOURS AGO
O	HAAAAAAAAANDS	2 HOURS AGO

AS A PROJECT DRAGS ON, MY GIT COMMIT MESSAGES GET LESS AND LESS INFORMATIVE.

Git : comment ça marche ?

Sauvegarder en ligne ?



Ressources Git

- Tutoriel ThinkR : R and Git
- Advance R , H. Wickham
- Happy Git and GitHub for the useR
- Git cheatsheet
- Quand ça part en vrille
- Réparer une erreur
- Créer une nouvelle branche avec git et merge des branches

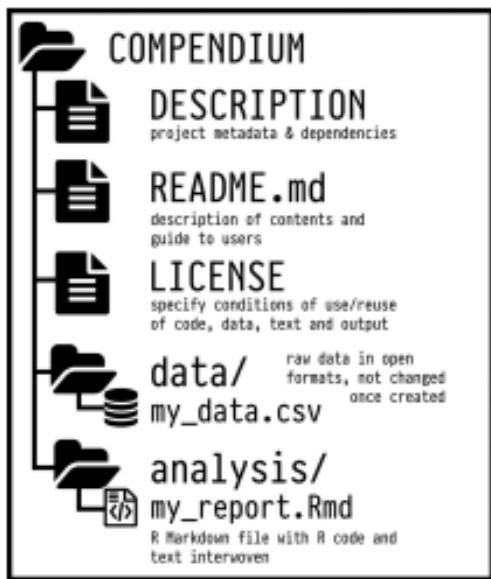
xkcd comics, CC BY-NC 2.5 license



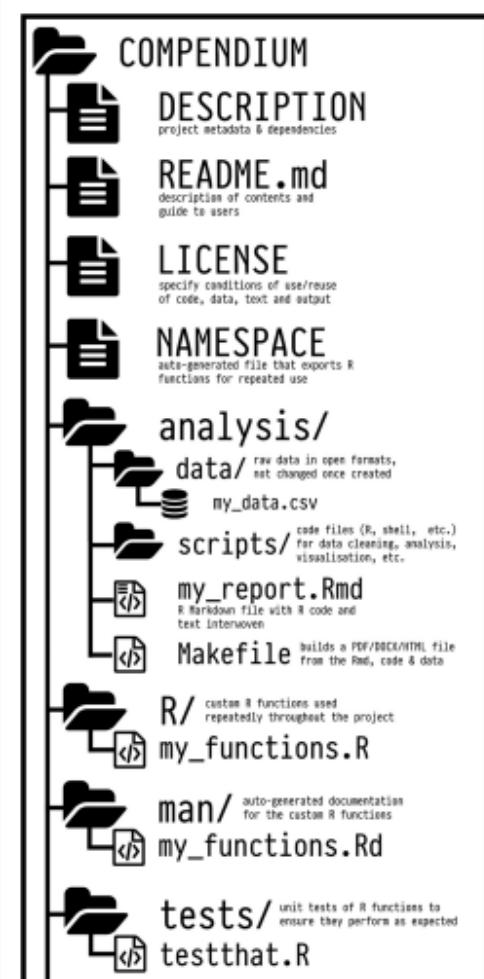
Accessible

COMPENDIUM

Structurer les dossiers en packages R.



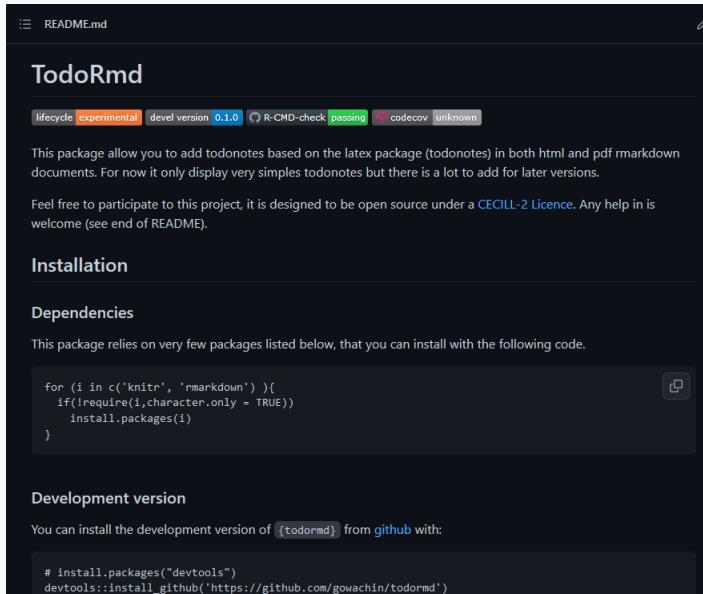
Marwick et al. "Packaging Data Analytical Work Reproducibly Using R (and Friends)". In: *The American Statistician* (2017).



Accessible

- **README** : entrée classique de documentation.
- Aide à l'installation et l'utilisation
- Pour ajouter à un projet :

```
usethis :: use_readme_md()
```



The screenshot shows the content of the `README.md` file for the `TodoRmd` package. The page has a dark theme with white text. At the top, it says "TodoRmd". Below that, there are lifecycle status badges: "experimental", "devel version 0.1.0", "R CMD check passing", and "codecov unknown". The main text reads: "This package allow you to add todonotes based on the latex package (todonotes) in both html and pdf rmarkdown documents. For now it only display very simples todonotes but there is a lot to add for later versions." It also mentions that the project is open source under the [CECILL-2 Licence](#). There are sections for "Installation", "Dependencies", and "Development version". The "Development version" section contains R code for installing the development version from GitHub.

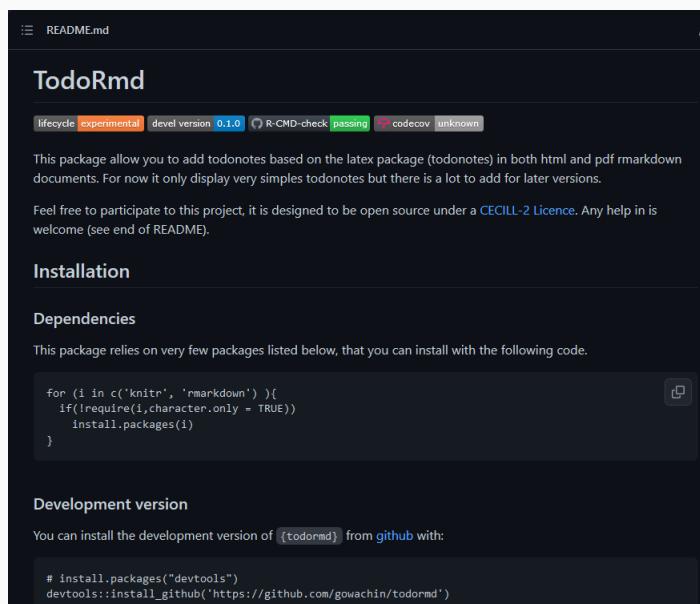
```
for (i in c('knitr', 'rmarkdown')) {
  if(!require(i, character.only = TRUE))
    install.packages(i)
}
```

```
# install.packages("devtools")
devtools::install_github('https://github.com/gowachin/todormd')
```

Accessible

- **README** : entrée classique de documentation.
- Aide à l'installation et l'utilisation
- Pour ajouter à un projet :

```
usethis :: use_readme_md()
```



The package allows you to add todo notes based on the `textrm{todo}` package in both HTML and PDF R Markdown documents. For now, it only displays very simple todo notes, but there is a lot to add for later versions.

Feel free to participate in this project; it is designed to be open source under a [CeCILL-2 Licence](#). Any help is welcome (see end of README).

Installation

Dependencies

This package relies on very few packages listed below, that you can install with the following code.

```
for (i in c('knitr', 'rmarkdown')) {  
  if(!require(i, character.only = TRUE))  
    install.packages(i)  
}
```

Development version

You can install the development version of `{todormd}` from GitHub with:

```
# install.packages("devtools")  
devtools::install_github('https://github.com/gowachin/todormd')
```

- **LICENCE** : sans licence, un code est théoriquement inutilisable.
(MIT, GNU GPL, CC)

Dicte les droits d'utilisation, de copie, de modification d'un code.

Loi Numérique n°2016-1321 impose une licence libre pour tout logiciel produit avec des fonds publics.



A nuancer avec les questions de publications, de tutelles etc.

Accessible

Projet = Package

- **DESCRIPTION :**

Package: todormd

Type: Package

Title: Using todonotes in rmarkdown package

Version: 0.1.0

Authors@R: c(

```
  person('Maxime', 'Jaunatre',
  email = "maxime.jaunatre@yahoo.fr",
  role = c('aut', 'cre'))
)
```

Description: Personnal project number x

License: CeCILL-2

Encoding: UTF-8

Imports:

```
  knitr,
  rmarkdown
```

RoxygenNote: 7.1.1

Suggests:

```
  testthat (> 3.0.0)
```

- **NEWS.md** : Fichier de suivis des mises à jour d'un projet.

Permet d'informer sur un changement majeur ou des nouveautés. Important d'avoir un cycle de version avec :

```
0.1.1 > 0.1.0
```

Why and how maintain a NEWS file for your R package?

- Pour ajouter une dépendances au projet :

```
devtools::use_package("tidyverse")
```



Accessible

Documentation i

- Rédiger en amont (`{Roxygen2}`)

```
#' half_life
#'
#' compute half_life percentage.
#'
#' @param period period in minute
#' @param time time elapsed (min)
#'
#' @return percentage.
#' @author M. Jaunatre <mail>
#'
#' @examples
#' half_life(20, 20)
#' half_life(20, 40)
#'
half_life <- function(period, time){
  return((1 - 2^(-time/period)) * 100)
}
```

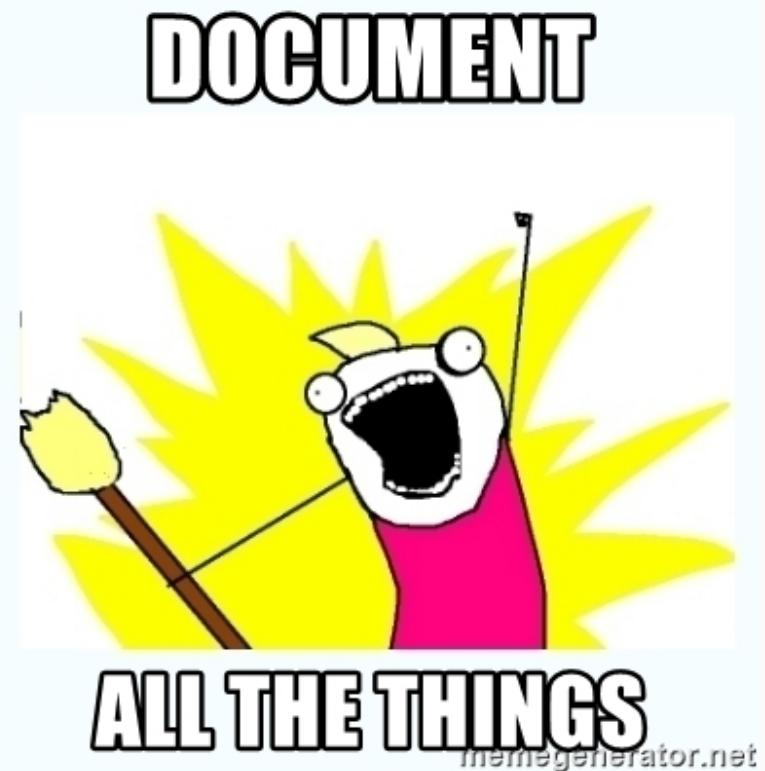
Accessible

Documentation i

- Rédiger en amont (`{Roxygen2}`)

```
#' half_life
#'
#' compute half_life percentage.
#'
#' @param period period in minute
#' @param time time elapsed (min)
#'
#' @return percentage.
#' @author M. Jaunatre <mail>
#'
#' @examples
#' half_life(20, 20)
#' half_life(20, 40)
#'

half_life <- function(period, time){
  return((1 - 2^(-time/period)) * 100)
}
```

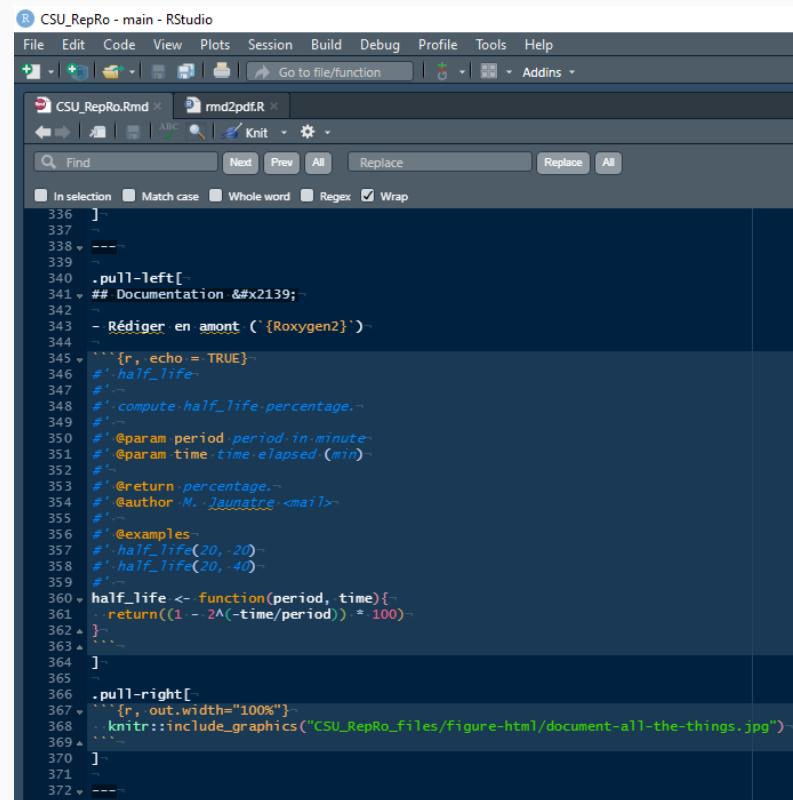


Accessible

Documentation i

Forme longue de documentation

- {Rmarkdown} ! Permet de mélanger texte (**md**, *LATEX*) et code **R**.
- Vignettes pour les packages
- Rédaction d'articles -> {rticles}
- Diapos -> {xaringan}
- Livre -> {bookdown}
- Site web -> {pkgdown}



```
R CSU_RepRo - main - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
+ - Go to file/function Addins ...
CSU_RepRo.Rmd rmd2pdf.R ...
← → ABC Knit ...
Find Next Prev All Replace All
In selection Match case Whole word Regex Wrap
336 ]
337
338 ---
339
340 .pull-left[
341 ## Documentation &#x2139;
342 -
343 - Rédiger en amont (`{Rxygen2}`)-
344
345 ````[r, echo = TRUE]-
346 `#`-
347 `#`-
348 `#` compute_half_life percentage.-
349 `#`-
350 `#` @param period period in minute-
351 `#` @param time time elapsed (min)-
352 `#`-
353 `#` @return percentage.-
354 `#` @author M. Jaunatre <mailto:>-
355 `#`-
356 `#` @examples-
357 `#` half_life(20, 20)-
358 `#` half_life(30, 40)-
359 `#`-
360 half_life <- function(period, time){-
361   return((1 - 2^( -time / period)) * 100)-
362 }-
363 ]-
364
365
366 .pull-right[
367 ````[r, out.width="100%"]-
368 knitr:::include_graphics("CSU_RepRo_files/figure-html/document-all-the-things.jpg")-
369 ]-
370
371
372 ---
```

Accessible

Lisibilité

```
# Title #####
# Author, date, contact (?) etc
# Description

## Depends #####
library(IAM)
# remotes::install_github(
#   "github.com/gowachin/todormd"
# )

# sources
sources("R/hello_world.R")
# functions
foo ← function(){cat("Don't panic !")}

## Datasets #####
read.csv(file = "raw_data/Picea_abies.

## Edit Dataset #####
## Plots #####
## Export #####
```

Accessible

Lisibilité

```
# Title #####
# Author, date, contact (?) etc
# Description

## Depends #####
library(IAM)
# remotes::install_github(
#   "github.com/gowachin/todormd"
# )

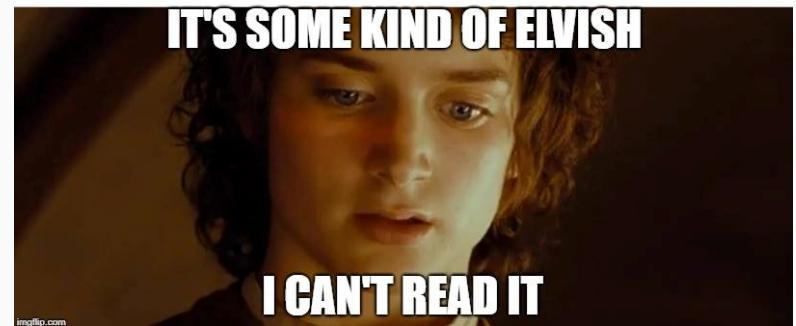
# sources
sources("R/hello_world.R")
# functions
foo ← function(){cat("Don't panic !")}

## Datasets #####
read.csv(file = "raw_data/Picea_abies.

## Edit Dataset #####
## Plots #####
## Export #####
```

- **DRY** (Don't Repeat Yourself)
- **KISS** (Keep it Simple, Stupid)
plein de petits fichiers > script infini
- `{cleanR}`: nettoyer les lignes inutiles.
- `{styleR}`: reformate le code.

When you trying to look at
the code you wrote a month ago



Accessible

Lisibilité

Bad

```
if(T){print(10)}  
T ← FALSE  
mean ← function(x) sum(x)  
  
if (y < 0 && debug)  
  message("y is negative")  
  stop("error for in y test")
```

```
function_with_many_argument("that", ma  
function_with_many_argument(x = "that"  
  many)
```

```
x = 5  
i = 0 ; y=12
```

Nice

```
if(TRUE){print(10)} # Full logical  
testing ← FALSE # name conflict  
use_sum ← function(x) sum(x)
```

```
if (y < 0 && debug) {  
  message("y is negative")  
  stop("error for in y test")  
} # Braces
```

```
function_with_many_argument(  
  x = "that",  
  y = many # name args  
) # line limit = 80
```

```
x ← 5 # arrow  
i ← 0  
y ← 12
```

"Code needs a lot of whitespace. That is how it breathes." Roger Peng

Interopérable



Un projet ne doit pas dépendre de l'ordinateur qu'il utilise -> Portabilité

Cela signifie l'oubli des commandes maudites :

```
setwd("C:/home/maxime/Documents/Projet magnifique/mouette/analyse/bob/modèle/Tset")
```

```
◀ | ▶
```

```
rm(list = ls()) # → Supprime uniquement les objects "utilisateurs"  
# → Ne supprime pas des dépendances chargées.  
# → Indique une session R ouverte depuis 30 ans  
# → Vide l'environnement des copains qui veulent aider  
# → Fait aussi disparaître les bébés phoques
```

Interopérable



Un projet ne doit pas dépendre de l'ordinateur qu'il utilise -> Portabilité

Cela signifie l'oubli des commandes maudites :

```
setwd("C:/home/maxime/Documents/Projet magnifique/mouette/analyse/bob/modèle/Tset/
```

◀ | ▶

```
rm(list = ls()) # → Supprime uniquement les objects "utilisateurs"  
# → Ne supprime pas des dépendances chargées.  
# → Indique une session R ouverte depuis 30 ans  
# → Vide l'environnement des copains qui veulent aider  
# → Fait aussi disparaître les bébés phoques
```

Si votre script possède ces lignes, **Jenny Bryan viendra bruler** votre ordinateur. 🔥

Solution ? Utiliser les **Projets Rstudio** ou {here}

A noter l'espace, et l'accent dans ce chemin !

Interopérable

File → New Project → New Directory

New Project

Back Create New Project



Directory name:
my_awesome_project

Create project as subdirectory of:
~/Documents/workflows [Browse...](#)

Create a git repository

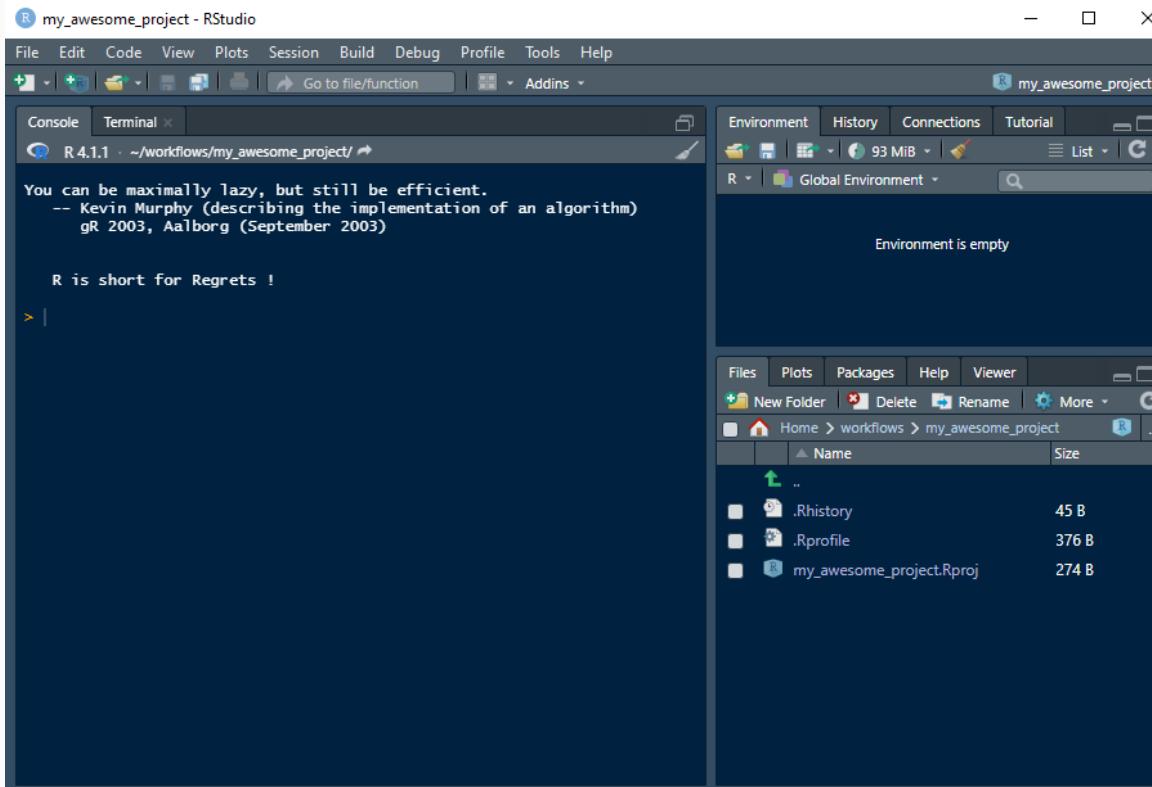
Use packrat with this project

Open in new session

[Create Project](#) [Cancel](#)

Interopérable

Dans un project, tout les chemins sont basés sur la racine du projet (où ce situe `.Rproj`)



Désactiver la sauvegarde automatique !

Tools → Global options → General → un-tick "always save history"



Données

Un script ne doit dépendre que des données présentes dans son projet.

raw_data/ = données intouchables

data/ = données temporaires



Données

Un script ne doit dépendre que des données présentes dans son projet.

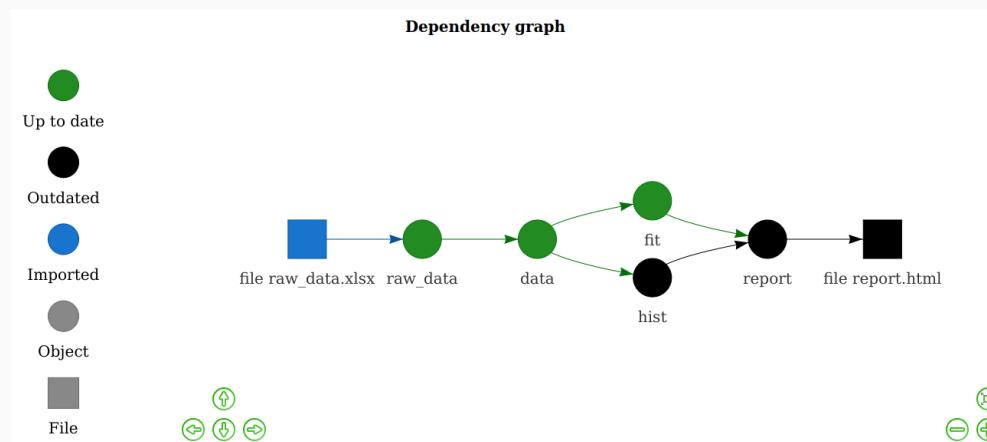
raw_data/ = données intouchables

data/ = données temporaires

- Eviter les `.RData` ! Indice de données trop complexes et de script pas clair.

Seule exception : les données de type cache : faire tourner les scripts plus facilement.

Package `{targets}` pour refaire tourner que les parties modifiées ou dépendantes d'un script.



Reusable



Parfois les données sont complexes et **non distribuables**.

- **ReprEx** : Avoir un **exemple** reproductible simplifié pour chaque script.

Reusable



Parfois les données sont complexes et **non distribuables**.

- **ReprEx** : Avoir un **exemple** reproduitible simplifié pour chaque script.

Exemple : ici la fonction prend un object en entrée...mais quel format doit-il avoir ?

```
rm_dups <- function(data){  
  data <- data[! duplicated(data$x), ]  
  data  
}
```

Reusable



Parfois les données sont complexes et **non distribuables**.

- **ReprEx** : Avoir un **exemple** reproduitible simplifié pour chaque script.

Exemple : ici la fonction prend un object en entrée...mais quel format doit-il avoir ?

```
rm_dups <- function(data){  
  data <- data[! duplicated(data$x), ]  
  data  
}  
  
df <- data.frame(x = c(0,1,1,0),  
                  y = c(2,1,0,0))  
rm_dups(df)  
  
##   x y  
## 1 0 2  
## 2 1 1
```

Reusable



Parfois les données sont complexes et **non distribuables**.

- **ReprEx** : Avoir un **exemple** reproductible simplifié pour chaque script.

Exemple : ici la fonction prend un object en entrée...mais quel format doit-il avoir ?

```
rm_dups <- function(data){  
  data <- data[! duplicated(data$x), ]  
  data  
}  
  
df <- data.frame(x = c(0,1,1,0),  
                  y = c(2,1,0,0))  
rm_dups(df)  
  
##   x y  
## 1 0 2  
## 2 1 1
```

Fournir un exemple est un très bon début de documentation.

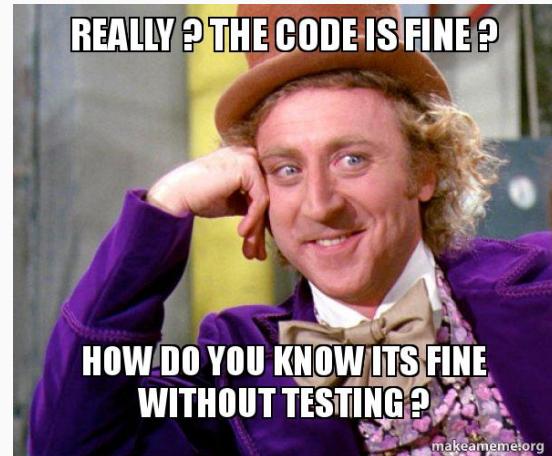
Exemple = niveau 0 du test !



Test !

Nous testons tous nos codes à la main, et ça marche sur le coup...mais 2 semaines après ?

- Ecrire des tests unitaires qui font tourner le code tout seul.
-> Dossier `test/`
- `usethis :: use_testthat()` charge automatiquement `{testthat}`



Il vaut mieux prévenir que guérir !



Test !

Utiliser `{usethis}` pour simplifier la vie !

```
library(testthat)
add <- function(y,x){
  x+y
}
```

- `usethis::use_test("add")`

```
tests
└── testthat
    └── test-add.R
└── testthat.R
```

Reusable



Test !

Utiliser `{usethis}` pour simplifier la vie !

```
library(testthat)
add <- function(y,x){
  x+y
}
```

- `usethis::use_test("add")`

```
tests
└── testthat
    └── test-add.R
└── testthat.R
```

```
test_that("add works", {
  expect_equal(add(39, 3), 42)
})
```

Test passed 🎉

```
test_that("add str error", {
  err = paste("non-numeric argument",
             "to binary operator")
  expect_error(
    add("hello", "world"), err
  )
})
```

Test passed 🎉



Test !

Utiliser `{usethis}` pour simplifier la vie !

```
library(testthat)
add <- function(y,x){
  x+y
}
```

- `usethis::use_test("add")`

```
tests
└── testthat
    └── test-add.R
└── testthat.R
```

```
test_that("add works", {
  expect_equal(add(39, 3), 42)
})
```

Test passed 🎉

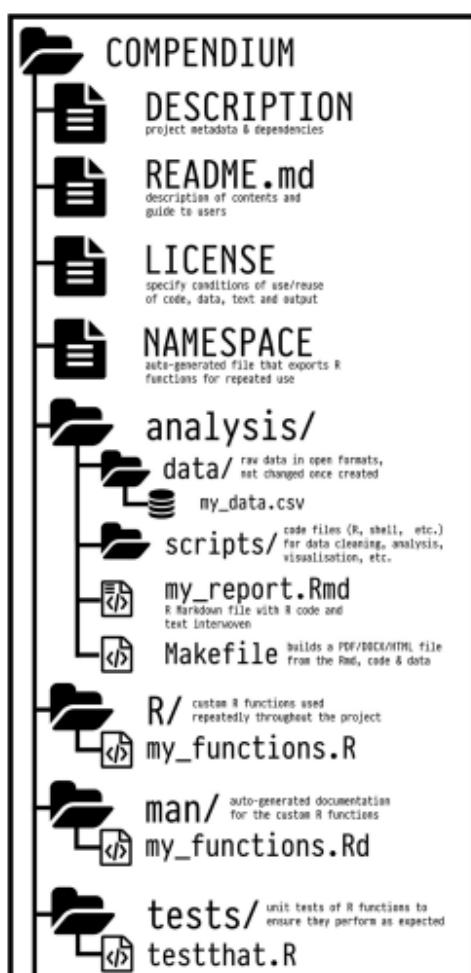
```
test_that("add str error", {
  err = paste("non-numeric argument",
             "to binary operator")
  expect_error(
    add("hello", "world"), err
  )
})
```

Test passed 🎉

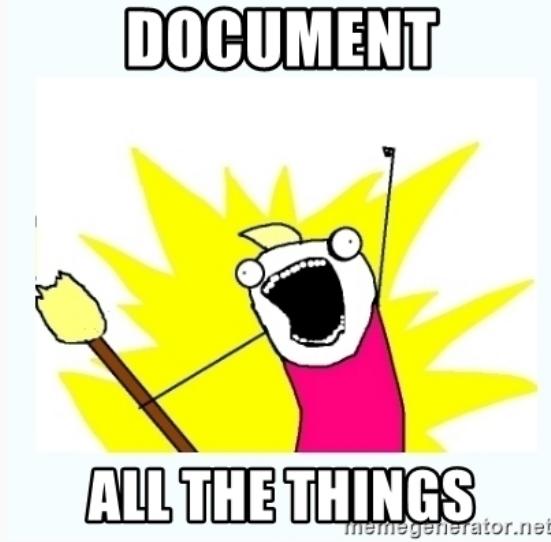
Rédiger des tests n'est jamais agréable mais sauve du temps plus tard.

Retour rapide lors de la modification du code + assurance de rien casser.

A retenir



- **Projets** Rstudio (`setwd()` = 🔥)
- **Lisibilité** (structure script)
- **ReprEx** (exemples)
- **Tests**



Trouver de l'aide pour R



RTFM*

- utiliser les aides `help(sum)` ou `?sum`
- lire les **manuels** de packages `ggplot2`
- avoir les **cheatsheet** quelque part !

Stack overflow



Issue Github



Communauté francophone de R :

- **frrrenchies** : list de doc fr
- **slack grrr** : question, news...

[*] Read The Fucking Manual.

Livres et blogs

- Advance R, Hadley Wickham
- Git et Rstudio, ThinkR
- Project-oriented workflow, Jenny Bryan

Conférences

- Code smells and feels, Jenny Bryan
- Toutes les conférences UseRs

Merci! Des questions ?

Slides en ligne https://gowachin.github.io/R_presentation/CSU_RepRo.html

Fichiers source  | Fichier pdf 

 gowachin  gowachin