


The Life & Times of a Reproducible Clinical Project in R

Jennifer L Thompson, MPH

Vanderbilt University Medical Center

Department of Biostatistics + Center for Critical Illness, Brain Dysfunction & Survivorship

R/Medicine 2018

 bit.ly/jlt-rmed2018

 jent103

Reproducibility, for Scientific Reasons

Given your data, someone else can reproduce your exact results

- Increased trustworthiness
- More rigorous, reliable science
- Learning more from one another's work

"Science should be 'show me', not 'trust me'; it should be 'help me if you can', not 'catch me if you can'."

-- Philip B. Stark, *Nature* 2018

Reproducibility, for Personal Reasons

"Did I mention that subjects with IDs > 100 are actually kangaroos and should be excluded?"

I won the lottery 🙌; now my coworker is taking over

Journal reviews back after 8 months! Time for revisions!

"How does that patient have 15 months of treatment when we only followed people for 12 months?"

Plan Ahead to Live Your Best (Research) Life

*"It's not thinking, 'This is easiest for myself right now.' It's thinking, 'When I'm working on this next week, next month, right before I graduate — **how** do I set myself up so that it's easier later?'"*

*-- Julia Stewart Lowndes in "A toolkit for data transparency takes shape,"
Nature 20 August 2018*

Plan Ahead to Live Your Best (Research) Life

*"It's not thinking, 'This is easiest for myself right now.' It's thinking, 'When I'm working on this next week, next month, right before I graduate — **how do I set myself up so that it's easier later?**'"*

*-- Julia Stewart Lowndes in "A toolkit for data transparency takes shape,"
Nature 20 August 2018*



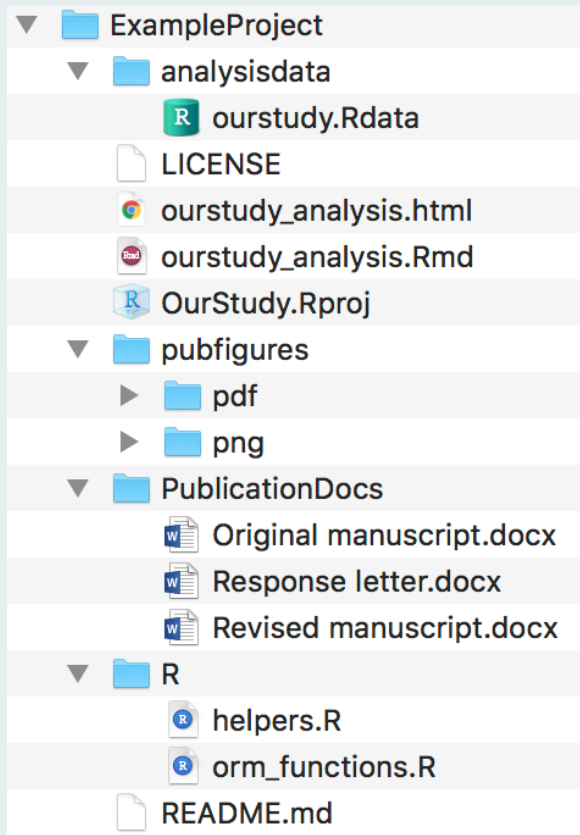
TODAY'S GOAL

Demonstrate several R tools & practices which can help us not only improve our scientific rigor, but make our lives more pleasant throughout the course of a project.

Phase I: Set Yourself Up for Success

Organized Files, Happier Life

One project, one directory



- Keep yourself organized
- Allow someone else (*inc. your future self!*) to use your code
- Reuse/adapt components for other projects

RStudio Projects + version control can help! Projects:

- Allow >1 project open at once
- Keep you in the right place
- Encourage 1:1 philosophy
- Facilitate **packrat**, version control
- "Your life will be better [if you use Projects]" - *Mine Çetinkaya-Rundel, yesterday*

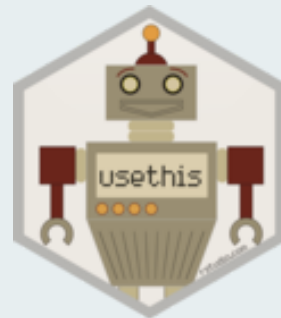
Why You, a [data person], Should Use Version Control

1. Encourages modular thinking and organization
2. Easier to know when and why something changed
3. Documents/timestamps your work
4. Easier to collaborate/work on multiple machines

Why You, a [data person], Should Use Version Control

1. Encourages modular thinking and organization
2. Easier to know when and why something changed
3. Documents/timestamps your work
4. Easier to collaborate/work on multiple machines





Use usethis for This

usethis is particularly helpful for creating R packages, but has plenty of functions that will help you set up a general project:

- Projects (`create_project()`)
- Standardized file structures (`use_r()`, `use_data_raw()`)
- git and GitHub (`use_git()`, `use_github()`)
- And more!

Bonus points: Use **usethis** to turn your project into a fully transportable package!

And also read these: Jenny Bryan on

- Project-oriented workflow
- Naming things
- Version control:
 - "Excuse me, do you have a moment to talk about version control?"
 - Happy Git with R *with STAT545 TAs*

Phase II:
Access data
programmatically

Tired

Many files, potentially with different versions, from disparate sources, manually created/exported

- No one (*including future you*) knows for certain what you did
- Potential organizational nightmare
- Manually exporting from data source is a potential pain point with plenty of room for error



Wired

Accessing your data programmatically
(*REDCap API, SQL database...*)

- Track exactly what data you access + all steps taken to manipulate it
- Raw data remains in original state/location
- Rerun same script(s) throughout data collection -> use most current data to monitor, clean, explore



Example: REDCap API

```
library(httr)

monthly_post <- httr::POST(
  url = "https://redcap.vanderbilt.edu/api/",
  body = list(
    token = Sys.getenv("MYTOKEN"), ## API token gives you permission
    content = "record",             ## export *records*
    format = "csv",                 ## export as *CSV*
    forms = "monthly_data",         ## which form(s)?
    fields = c("study_id"),         ## additional fields
    events = paste(sprintf("month_%s_arm_1", 1:3), collapse = ","),
    ## all 3 monthly visit events
    rawOrLabel = "label"           ## export factor *labels* v codes
  )
)

monthly_df <- read.csv(
  text = as.character(monthly_post),
  stringsAsFactors = FALSE, na.strings = ""
)
```

Example: REDCap API

```
library(httr)

monthly_post <- httr::POST(
  url = "https://redcap.vanderbilt.edu/api/",
  body = list(
    token = Sys.getenv("MYTOKEN"), ## API token gives you permission
    content = "record",             ## export *records*
    format = "csv",                 ## export as *CSV*
    forms = "monthly_data",         ## which form(s)?
    fields = c("study_id"),          ## additional fields
    events = paste(sprintf("month_%s_arm_1", 1:3), collapse = ","),
    ## all 3 monthly visit events
    rawOrLabel = "label"            ## export factor *labels* v codes
  )
)

monthly_df <- read.csv(
  text = as.character(monthly_post),
  stringsAsFactors = FALSE, na.strings = ""
)
```

Example: REDCap API

```
library(httr)

monthly_post <- httr::POST(
  url = "https://redcap.vanderbilt.edu/api/",
  body = list(
    token = Sys.getenv("MYTOKEN"), ## API token gives you permission
    content = "record",             ## export *records*
    format = "csv",                 ## export as *CSV*
    forms = "monthly_data",         ## which form(s)?
    fields = c("study_id"),         ## additional fields
    events = paste(sprintf("month_%s_arm_1", 1:3), collapse = ","),
    ## all 3 monthly visit events
    rawOrLabel = "label"           ## export factor *labels* v codes
  )
)

monthly_df <- read.csv(
  text = as.character(monthly_post),
  stringsAsFactors = FALSE, na.strings = ""
)
```


Now We Can...

Once the initial script or report is built, we can - **with one line of code*** - do the following, always assured that we're using the most recently updated data:

1. Frequently monitor accuracy & completion during prospective data collection ([tutorial on GitHub](#))
2. Use API + RMarkdown to easily run analyses that happen repeatedly *during* prospective data collection:
 - NIH or DSMB reports
 - Study monitoring (We [use flexdashboards](#))
 - Are you executing the study you planned to do?
 - Enrollment targets, protocol pain points...

API Resources

- General: [httr](#); intro to APIs by Lucy D'Agostino McGowan
- REDCap: [redcapAPI](#) (Benjamin Nutter), [REDCapR](#) (Will Beasley)
- [ropensci.org/packages](#) for specific web APIs

Phase III: Test Your (Data) Assumptions

Testing Data with assertr

Sinking feeling (n):

Going far too long without realizing you've made a major yet sneaky error in your data wrangling.



Testing Data with assertr

Sinking feeling (n):

Going far too long without realizing you've made a major yet sneaky error in your data wrangling.



With **assertr** (author: Tony Fischetti), we can

- Include assertions in scripts that
 - Load raw data - *is it what we expect?*
 - Create analysis datasets - *is our code doing what we expect?*
- Discover problems **early**
- Be more **confident** in your data wrangling + analysis

Checking Raw Data

```
library(assertr)

## Creatinine must be <=20
monthly_df %>%
  verify(creat_m <= 20)
```

Checking Raw Data

```
library(assertr)

## Creatinine must be <=20
monthly_df %>%
  verify(creat_m <= 20)
```

Result: 🙅

Pipeline stops
and prints an
informative
message

```
verification [creat_m <= 20] failed! (1 failure)
```

	verb	redux_fn	predicate	column	index	value
1	verify	NA	creat_m <= 20	NA	5	NA

```
monthly_df[5, c("study_id", "redcap_event_name", "c
```

	study_id	redcap_event_name	creat_m
5	3	Month 2	25

Checking Raw Data

```
monthly_df %>%  
  ## Visit date *must* be entered;  
  ## HDL, LDL must be within range  
  assert(not_na, date_visit_m) %>%  
  assert(within_bounds(25, 95), hdl_m) %>%  
  assert(within_bounds(1, 200), ldl_m)
```

Checking Raw Data

```
monthly_df %>%  
  ## Visit date *must* be entered;  
  ## HDL, LDL must be within range  
  assert(not_na, date_visit_m) %>%  
  assert(within_bounds(25, 95), hdl_m) %>%  
  assert(within_bounds(1, 200), ldl_m)
```

Result: 👍

Returns our original data frame so we can move on down the pipeline!



You Decide

How Strict Your Assertions Are

Q: What if you
want to test
assumptions,
but keep
going no
matter what?

A: **error_fun**

You Decide

How Strict Your Assertions Are

Q: What if you want to test assumptions, but keep going no matter what?

```
## Maybe it's OK if creatinine is super high
newdf <- monthly_df %>%
  verify(creat_m <= 20, error_fun = error_append)

attr(newdf, "assertr_errors")
```

A: **error_fun**

```
[[1]]
```

```
verification [creat_m <= 20] failed! (1 failure)
```

	verb	redux_fn	predicate	column	index	value
1	verify	NA	creat_m <= 20	NA	5	NA

Result: 😐

For More: assertr tutorial

ABOUT BLOG PACKAGES COMMUNITY DISCUSS

assertr tutorial

for v2.0.2.2

Phase IV: Take Advantage of That Workflow

drake



Built to encourage and enable efficient, reproducible workflows

1. Set up a workflow made of separate components
2. Describe the **plan** that will create/update these **targets**
3. **make()** the **plan**
 - **drake** knows which components are up to date, which to update
 - Full reproducibility without unnecessary rebuilding

Much more at:

- ropensci.github.io/drake
- **drake** manual
- Will Landau's R/Pharma talk: wlandau.github.io/drake-talk



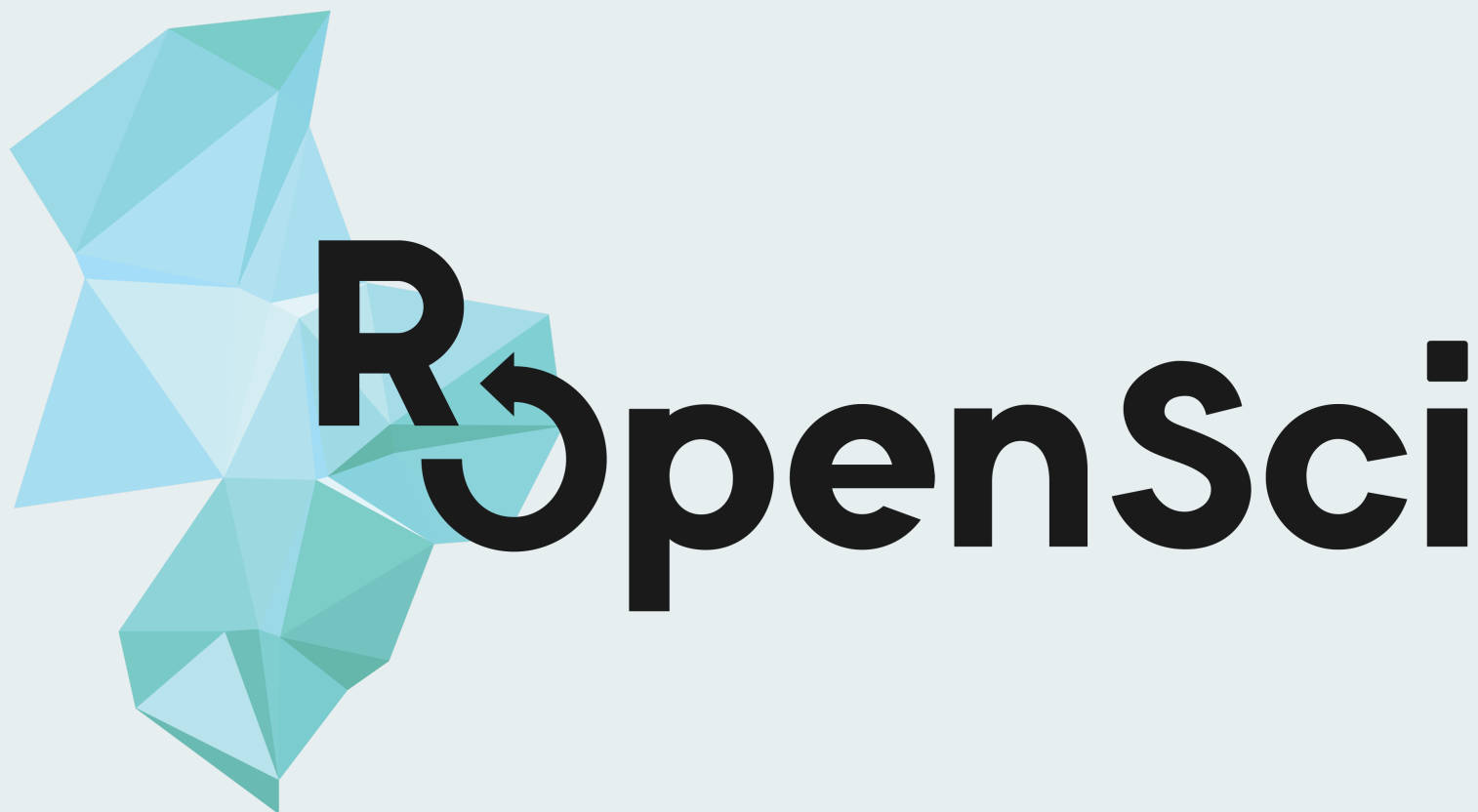
Scale up the work you need.



Skip the work you don't.



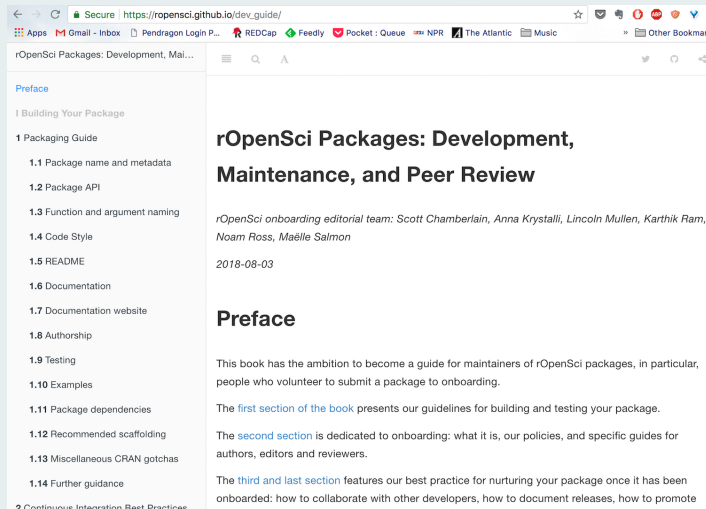
See evidence of reproducibility.



Software

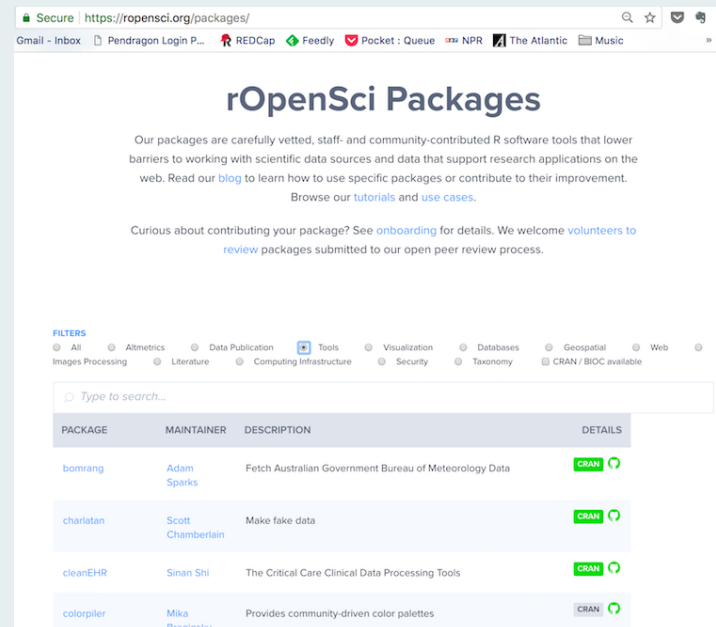
Guidelines & resources for developers

ropensci.github.io/dev_guide



Discoverability

- Blog posts, social media
- Searchable database





Peer Review

Goal: Bring the best facets of academic peer review to software, in a transparent, non-adversarial process

- **Developers:**
Feedback on design; visibility
- **Reviewers:**
Improve a package; learn about design & development

ropensci / onboarding

Watch 26

Code Issues 29 Pull requests 0 Projects 0 Wiki Insights

Skimr #175

Closed elinw opened this issue on Dec 20, 2017 · 61 comments

elinw commented on Dec 20, 2017 · edited

Summary

- What does this package do? (explain in 50 words or less):
Creates compact and flexible data summaries that are pipeable and display nicely in the console. For a given data frame or vector the skim function provides a useful set of summary statistics (based on the class of the individual vector/column) that allows users to skim their data to get an overall sense of what is included, extent of missing values, and similar information. The skim generic can be extended by users to other data structures.



Community

Welcoming, diverse community that

- Builds capacity of users & developers
- Fosters pride in members' work
- Advocates for data sharing, reusable software
- My experience: This culture and expectation leads to
 - More & better collaboration
 - Better products
 - Stronger research



Community

Welcoming, diverse community that

- Builds capacity of users & developers
- Fosters pride in members' work
- Advocates for data sharing, reusable software
- My experience: This culture and expectation leads to
 - More & better collaboration
 - Better products
 - Stronger research

"rOpenSci combines expertise and approachability, and its community inspires people to collaborate as the best versions of themselves."

-- *Will Landau, rOpenSci community member & **drake** developer*

Info/Get Involved



ropensci.org

discuss.ropensci.org

ropensci.org/packages

Contribute a package via the
onboarding process
or sign up to review!

 [@ropensci](https://twitter.com/ropensci)

Thank you!

Slides/contact

 bit.ly/jlt-rmed2018

 jenthompson.me

Find rOpenSci:

 ropensci.org

 [@ropensci](https://twitter.com/ropensci)



Jeff Leek Made This JSM 2018 talk

Or, Use RMarkdown to Record Your Decisions

