# The Workflow of Your Analysis [1]

Michael Höhle[x]


m_hoehle

[x]Department of Mathematics, Stockholm University, Sweden

Statistical Consulting Course
14 Dec 2018

Stockholm University

# Outline

# Motivation: The Setup

- So far you learned how to talk to your clients, identify research questions and fit statistical models
- You know R, got data from a client and are ready to start...
- However, the data you got consists of several Excel files and they don't fit into the structure that your R function needs...
- How to get your analysis pipeline running from this point?

# Good Enough Practices...

**Jenny Bryan** @JennyBryan · 22. Juni 2017

Don't let the perfect be the enemy of the ... **Good enough practices** in scientific computing! Now in @PLOSCompBiol doi.org/10.1371/journa...

Tweet übersetzen

1 99 131

Source: Tweet by Jenny Bryan on 2017-06-22.

# Outline

# Project Organization

Recommendations of Wilson et al. (2017):

- Put each project in its own directory, which is named after the project
- Put text documents associated with the project in the doc directory
- Put raw data and metadata in a data directory and files generated during cleanup and analysis in a results directory
- Put project source code in the R/src directory
- Name all files to reflect their content or function
  - For files sent to external collaborators: Use a date stamp as part of the file name

# Project Folder Layout

Adapted from Wilson et al. (2017):

```
.
|--README
|--LICENSE
|--data
| |--birds_count_table-2018-12-12.csv
| |--measurement-locations-2018-12-12.csv
|--doc
| |--email-collaborator-2018-10-10.txt
| |--todo.txt
|--results
| |--birdlocations.RData
| |--summarized_results.RData
|--R
| |--sightings_analysis.R
| |--runall.R
|--manuscript
| |--howbirdsmove.Rmd
| |--howbirdsmove.docx
| |--submission
|   |--howbirdsmove-nature-2018-12-13.pdf
```

# Data Management (1)

Recommendations from Wilson et al. (2017):

- Save the raw data
- Ensure that raw data are backed up in more than one location
- Create the data you wish to see in the world
- Create analysis-friendly data
- Record all the steps used to process data
- Anticipate the need to use multiple tables, and use a unique identifier for every record
- (Submit data to a reputable DOI-issuing repository so that others can access and cite it)

# Data Management (2)

- Most likely, you will spend 80% of the project time wrangling your data into the desired shape
- Some problems repeat: date formatting, encoding, ...
- In R: Use the `tidyverse` and pipes for these steps - a great resource explaining this is Wickham and Grolemund (2017):

`https://r4ds.had.co.nz`

# Writing your Programs

Recommendations from Wilson et al. (2017):

- Place a brief explanatory comment at the start of every program
- Decompose programs into functions
- Be ruthless about eliminating duplication
- Always search for well-maintained software libraries that do what you need
- Test libraries before relying on them
- Give functions and variables meaningful names
- Make dependencies and requirements explicit
- Do not comment and uncomment sections of code to control a program's behavior
- Provide a simple example or test data set
- (Submit code to a reputable DOI-issuing repository)

# Keeping track of your changes

- As a service to your future self: Use a version control program, e.g., `git`
- `git` does not require to upload files to the cloud
  - origin on a shared local network drive
  - local origin and transfer bundles by email
  - go cloudy using a private github repo or use a shared cloud resource like Dropbox, Google Drive, ...
- RStudio provides seamless `git` integration, but one can also use modern web interfaces such as GitHub or GitLab.
- Other `git` GUI clients are listed at

  `https://git-scm.com/downloads/guis`

# Writing the Manuscript (1)

- I'm going to take a bet that you collaboration partner will want to use Word for this! (Warning: limited version control!)
- Note: It is possible to convert your Rmd file to Word – this is done by `rmarkdown::render` with the `word_document()` renderer and is easily available through RStudio

  Example: `analysis.Rmd`
- Better alternative if your contribution is beyond providing 2 figures: Use a collaborative writing platform, e.g., GoogleDocs, Overleaf (LaTeX) or Word Online

# Writing the Manuscript (2)

- The `officer` package provides helpful functionality to dynamically produce PowerPoint presentations
- Even modifiable Windows Graphics can be generated
- A project worth following is the `redoc` package allowing for a two-way R-Markdown $\leftrightarrow$ Microsoft Word workflow

# Outline

# 10 Rules for Reproducible Analyses (1)

Sandve et al. (2013) introduces the following 10 rules to "foster a culture of reproducibility":

1. For Every Result, Keep Track of How It Was Produced
2. Avoid Manual Data Manipulation Steps
3. Archive the Exact Versions of All External Programs Used
4. Version Control All Custom Scripts
5. Record All Intermediate Results, When Possible in Standardized Formats

# 10 Rules for Reproducible Analyses (2)

6. For Analyses That Include Randomness, Note Underlying Random Seeds
7. Always Store Raw Data behind Plots
8. Generate Hierarchical Analysis Output, Allowing Layers of Increasing Detail to Be Inspected
9. Connect Textual Statements to Underlying Results
10. Provide Public Access to Scripts, Runs, and Results

# Outline

# Outline

# Make reproducible analyses

- Use dynamic report generation in R → `knitr` based on LaTeX/Rmarkdown files (i.e. Rnw/Rmd)
- Make your scripts generic:
  - make the programs readable and portable
  - make a pacakge wrapping the functionality (forces you to generalize, document and test)
- Ensure that you store and can reproduce the version of the R packages used in your analysis!
  - Minimum requirement: Have `sessionInfo()` at the end of your Rnw/Rmd file.
- For better debugging: Name your chunks. If you didn't, the pkg `namer` can help.

# Make your programs portable

- Try to avoid `setwd` with hard coded paths in your program – this is fragile and your collaborator's hard disk likely doesn't have that path!

```
setwd("/Users/hoehle/Teaching/StatisticalConsulting2018/Workflow/Foils")
```

- Instead use the package here for your project:

```
here::here()
```

```
## [1] "/Users/hoehle/Teaching/StatisticalConsulting2018/Workflow/Foils"
```

```
rmarkdown::render(here::here("analysis.Rmd"), rmarkdown::word_document())
```

```
##
##
## processing file: analysis.Rmd
## output file: analysis.knit.md
##
## Output created: analysis.docx
```

- Simpler alternative: Use the `file.path` function with relative paths, e.g., `file.path(".", "R", "foo.R")`

# Managing your R packages

- The R package `packrat` provides a package management for your projects.
- See this tutorial on how to get started:

```
packrat::init(".")
```

- You can view which package are available in your packrat using

```
packrat::status()
```

- And possibly dump a snapshot of any package changes with

```
packrat::snapshot()
```

# Outline

# Critical Analyses

- If your results have substantial impact, you need to guarantee that the results are correct (according to the selected method)
- Several approaches are possible:
  - Single player mode: Write **unit tests**
  - Pair programming
  - Four-eye principle as part of a **code review**
  - Independent approaches by two persons and comparing results
- Collaborate with your future self: Redo your analyses after 6 weeks...
- Once you are stable: Track any errors you find in a database

# Check your results

- Comparison of results is supported by the `daff` package, which is a package for the `diff` of `data.frame` objects.
- A use case for a control calculation ping-pong is presented in the blog post Pair Programming Statistical Analyses

```
daff::diff_data(ada, bob)$get_data()
```

```
##    @@ Region NoOfUnits   Sales Volume   Staff Costs People
## 1 +++   <NA>         1          19500          7609      2
## 2  ->      A       3->2   59623->42123  43103->30103 16->13
## 3  ->      B       1->2 119500->239000 95691->185691     19
## 4  ->      D       1->2   45860->70860  32555->32655      9
## 5          E         1          33020         25010      7
```

# Outline

# Discussion

- The suggestions in this lecture create overhead
- Many an analysis happens in the conflict zone of meeting the deadline and living up to the ideals of reproducibility
- In 19 out of 20 times, nobody will ever ask what you did. But in the last case? What if your analysis results suddenly change a way people eat or that the government spend millions on vaccines?
- It is never too late to refactor your code!

# References I

Sandve, G. K., Nekrutenko, A., Taylor, J., and Hovig, E. (2013). Ten simple rules for reproducible computational research. PLOS Computational Biology, 9(10):1–4.

Wickham, H. and Grolemund, G. (2017). R for Data Science. O'Reilley.

Wilson, G., Bryan, J., Cranston, K., Kitzes, J., Nederbragt, L., and Teal, T. K. (2017). Good enough practices in scientific computing. PLOS Computational Biology, 13(6):1–20.