Developing a Workflow to Maximize Reproducibility and Research Impact: Managing Data, Computer Code, and Projects for Success

Althea A. ArchMiller & John R. Fieberg

7/12/2017

Welcome!

Developing a Workflow to Maximize Reproducibility and Research Impact: Managing Data, Computer Code, and Projects for Success

Althea A. ArchMiller *Assistant Professor, Biology, Concordia College, MN*

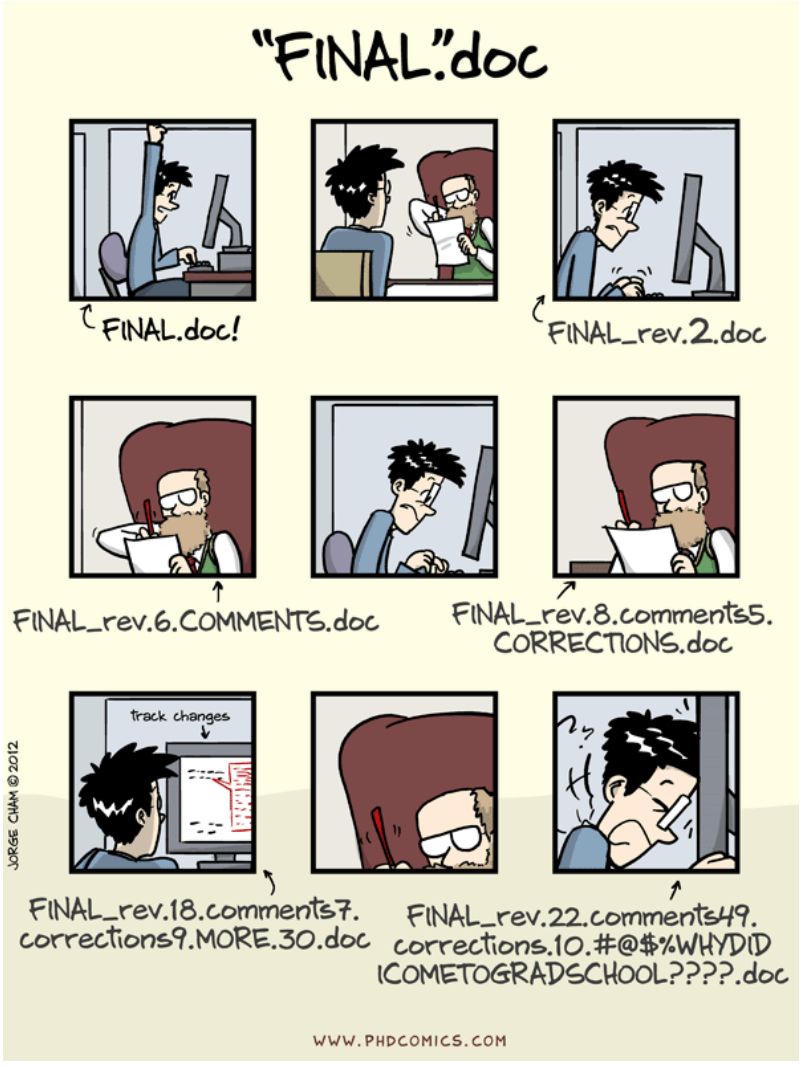
John R. Fieberg *Associate Professor of Quantitative Ecology, University of Minnesota*

Before we begin, please open up the reproducible_workshop repository in RStudio and install these packages:

- knitr
- ezknitr
- devtools

Why worry about reproducibility?

Working towards future reproducibility makes my code easier for my collaborators (and me) to read, run, and debug today,
and that's why I think reproducibility is a **win-win for all researchers**."
-Althea



Why worry about reproducibility?

"[Reproducibility] provides security, saves time, and forces me to be more thoughtful about my workflow." - Ethan Young

- make your life easier! Now, and in the future
- collaborations
- broader research impact
- increased citations
- transparency
- grant and journal requirements

Is my research reproducible?

- What formats are your research documents stored in?

Is my research reproducible?

- What formats are your research documents stored in? - .csv
  - .txt
  - .pdf
  - .html
  - .R/.Rdata
    - YES!

Is my research reproducible?

- What formats are your research documents stored in? - .csv
  - .txt
  - .pdf
  - .html
  - .R/.Rdata
    - YES!
  - .doc/.docx
  - .sas
  - .xls/.xlsx
  - any other proprietary file format
    - NO!

Is my research reproducible?

- Is your code linear?
  - Clear environment often and at beginning of script
  - Don't save .Rdata or history
  - Each program should focus on one main task or analysis
  - Don't rely on manual commenting/uncommenting

Is my research reproducible?

- Is your code linear?
  - Clear environment often and at beginning of script
  - Don't save .Rdata or history
  - Each program should focus on one main task or analysis
  - Don't rely on manual commenting/uncommenting

So, what's wrong here?

```
# What variables are significant?
lm.out <- lm(weight ~ height, data = trial.data)
remove(lm.out) # clear previous lm.out for each
               #   new lm() definition above

# Is the relationship significant?
#     (If not, clear and try a new regressor)
summary(lm.out)
```

Is my research reproducible?

- Are your files easily shared with others?
  - Organized directory structure
  - Files relatively linked
  - Well-documented & commented
  - Consistency in coding practices

"The point of having style guidelines is to have a common vocabulary of coding so people can concentrate on *what* you are saying, rather than on *how* you are saying it." - Google's R Style Guide

Workshop Outline

The goal for this workshop is to help you develop the tools to develop a workflow to maximize reproducibility, collaborations, and research impact.

1. RStudio Projects for organizing data, code, and output

Workshop Outline

The goal for this workshop is to help you develop the tools to develop a workflow to maximize reproducibility, collaborations, and research impact.

1. RStudio Projects for organizing data, code, and output
2. R-Markdown and R-Oxygen for documenting your code and creating reproducible reports

Workshop Outline

The goal for this workshop is to help you develop the tools to develop a workflow to maximize reproducibility, collaborations, and research impact.

1. RStudio Projects for organizing data, code, and output
2. R-Markdown and R-Oxygen for documenting your code and creating reproducible reports
3. GitHub for version-control, collaborating and archiving

1. RStudio Projects

Think about a typical research project, maybe a dissertation chapter or an experiment that you've managed from data collection through publication. What are typical **folders** that you've used?

1. RStudio Projects

Think about a typical research project, maybe a dissertation chapter or an experiment that you've managed from data collection through publication. What are typical **folders** that you've used?

- Raw data
- Processed data
- Analysis scripts
- Paper/Manuscript-related documents
- Sharing documents ("transmittals")
- Metadata
- Maps or other deliverables

RStudio Projects provide an opportunity for you to organize and manage all of these types of folders in **one place** in a way that **relatively links** everything together and **eases sharing**.

1. RStudio Projects

Think about a typical research project, maybe a dissertation chapter or an experiment that you've managed from data collection through publication. What are typical **folders** that you've used?

- Raw data
- Processed data
- Analysis scripts
- Paper/Manuscript-related documents
- Sharing documents ("transmittals")
- Metadata
- Maps or other deliverables

RStudio Projects provide an opportunity for you to organize and manage all of these types of folders in **one place** in a way that **relatively links** everything together and **eases sharing**.

Up next, Activity 1!

Activity 1: Data management and updating

Here, we will read in and process three weeks of experimental data and do some preliminary analysis. Then, we will get a final (4th) week of data, which we will merge with the original data.

The goals are to:

1. Be introduced to RStudio
2. Create a framework for keeping data organized and up-to-date
3. Automatically update our analyses based on the master dataset

Context: Abundance data from ~75 invertebrate species sampled on various beaches along the Dutch coast.

Zuur, A.F., E.N. Ieno, and G.M. Smith (2007) Analysing Ecological Data. Springer, New York.

Activity 1: Data management and updating

Before we begin today, we need sync your individual versions of the workshop documents with Althea's master branch:

Activity 1: Data management and updating

Before we begin today, we need sync your individual versions of the workshop documents with Althea's master branch:

1. Open RStudio and your reproducibility_workshop.rproj. (File > Open Project...)
2. Open shell (Tools > Shell...)
3. Type in exactly, then press enter:

```
$ git fetch upstream
```

4. Type in exactly, then press enter:

```
$ git checkout master
```

5. Type in exactly, then press enter:

```
$ git merge upstream/master
```

Activity 1: Data management and updating

Now create a new folder in student_folders/ for all of today's activities. Name the folder after yourself (or an alias).

Open a new R Script file and save it to that new folder as **"activity1a_data_processing.R"**

First, we will read in first three weeks of data and combine them, process the data a little bit, and save the merged/processed data for analysis.

Secondly, we will save another new R Script file as **"activity1b_data_analysis.R"** and do (preliminary) regression analysis.

Finally, we will pretend to have just gotten the final week's data in and update everything in a "reproducible" way.

1. RStudio Projects

Other links

`https://swcarpentry.github.io/r-novice-gapminder/02-project-intro/`

Data Mangement Tips

- Treat data as read-only
  - Don't use Excel, etc, to manipulate raw data
  - Use a single R program for all manipulation
  - Save "cleaned" or "processed" data in easily loadable formats
- Differentiate data types with folders *raw* versus *processed* versus *output* (e.g., linear regression objects, etc)
- Write dates in YYYYMMDD or equivalent format

Why R-Markdown for manuscripts?

"I can do reproducible work in R (making me happy) and format the output report in Word (making my collaborators happy)" - Richard Layton `http://rmarkdown.rstudio.com/articles_docx.html`

Documenting Code

Native R Scripts (.R extensions) (or any analysis code) are generally not designed for reading, but the **knitr** library has been designed for converting R scripts into readable reports, such as Word, PDF, and/or html documents.

Not only do these types of reports help with collaborating, they provide a great framework for archiving your analyses and results.

Example:
`https://conservancy.umn.edu/handle/11299/181607`

Documenting Code

General tips for better coding:

- Consistent and meaningful naming conventions
  - a = b*c
  - weekly.pay = hours.worked*pay.rate (not cross-compatible)
  - weekly_pay = hours_worked*pay_rate
  - weeklyPay = hoursWorked*payRate
- Using YYYYMMDD or equivalent for dates
- When possible, bundle data, code, figures/tables together with **knitr** package using R-Markdown or R-Oxygen languages

Documenting Code: R-Markdown

R-Markdown combines markdown language, which is "an easy-to-write plain text format" and embedded R code chunks that are "run so their output can be included in the final document" [1]

Documenting Code: R-Markdown

R-Markdown combines markdown language, which is "an easy-to-write plain text format" and embedded R code chunks that are "run so their output can be included in the final document" [1]

```
14  This is an R Markdown document. Markdown is a
    simple formatting syntax for authoring HTML,
    PDF, and MS Word documents. For more details
    on using R Markdown see
    <http://rmarkdown.rstudio.com>.
15
16  When you click the **Knit** button a document
    will be generated that includes both content
    as well as the output of any embedded R code
    chunks within the document. You can embed an
    R code chunk like this:
17
18  ```{r cars}
19  summary(cars)
20  ```
21
```

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```
##      speed           dist
##  Min.   : 4.0   Min.   :  2.00
##  1st Qu.:12.0   1st Qu.: 26.00
##  Median :15.0   Median : 36.00
##  Mean   :15.4   Mean   : 42.98
##  3rd Qu.:19.0   3rd Qu.: 56.00
##  Max.   :25.0   Max.   :120.00
```

[1] www.rmarkdown.rstudio.com

Documenting code: R-Markdown

Exercise 2a: Introduction to R-Markdown

- File > New File > R Markdown...
- Choose "html" - optionally put in a title and press "OK"
- This R-Markdown template is ready to "knit" into an html as-is
  - Click the blue Knit button
  - Save in your student_project folder as "activity2a_intro_rmarkdown.Rmd"
  - See the resultant html
- Take a few minutes to modify the .Rmd and view how the changes appear in the knit html document.

`https://www.rstudio.com/wp-content/uploads/2015/02/rmarkdown-cheatsheet.pdf`

Documenting code: ezknitr

What folder did the html end up in?

Documenting code: ezknitr

What folder did the html end up in?

Now imagine if you wanted to keep the programs/scripts in a folder separate from reports (highly recommended!). You can easily direct the output html file into a different folder using **ezknitr** package.

```
library(ezknitr)
ezknit("student_folders/yours/activity2a_intro_rmarkdown.Rmd",
       out_dir = "student_folders/archmiller_a/reports",
       fig_dir = "figures",
       keep_md = F)
```

Documenting Code: R-Oxygen

Instead of using the R-Markdown language, you can also use **pure R scripts** plus **Roxygen comments** (#') to create fully reproducible reports!

Documenting Code: R-Oxygen

Instead of using the R-Markdown language, you can also use **pure R scripts** plus **Roxygen comments** (#') to create fully reproducible reports!

Benefit: The entire program can be written and run in the familiar R Script file, then "spun" into an html/Word/pdf document at any point. + Learning to code with R-Oxygen is arguably more natural since we already use # for commenting

Documenting Code: R-Oxygen

Instead of using the R-Markdown language, you can also use **pure R scripts** plus **Roxygen comments** (#') to create fully reproducible reports!

Benefit: The entire program can be written and run in the familiar R Script file, then "spun" into an html/Word/pdf document at any point. + Learning to code with R-Oxygen is arguably more natural since we already use # for commenting

Additionally, use #+ to define and label R chunks like we did with the "'{r ...} code in the R-Markdown language.

Documenting Code: R-Oxygen

Instead of using the R-Markdown language, you can also use **pure R scripts** plus **Roxygen comments** (#') to create fully reproducible reports!

Benefit: The entire program can be written and run in the familiar R Script file, then "spun" into an html/Word/pdf document at any point. + Learning to code with R-Oxygen is arguably more natural since we already use # for commenting

Additionally, use #+ to define and label R chunks like we did with the "'{r ...} code in the R-Markdown language.

Documenting Code: R-Oxygen

Activity 2b: Introduction to R-Oxygen

Here, we'll quickly convert "activity1b_data_analysis.R" into an html document!

1. Open "activity1b_data_analysis.R"
2. Save As...
   "student_folder/yours/activity2b_intro_roxygen.R"

LaTeX

Another benefit of using knitr/Rmd/Roxygen for creating statistical reports is the nice interface with LaTeX equation syntax.

Activity 3. LaTeX Equations

1. Create a new .Rmd PDF document
2. Save it as "student_folders/yours/activity3_latex.Rmd"
3. Create the following in the output PDF
   - $\alpha + \beta = 2\theta$
   - $\pi^2 = 9.86$
   - $\sum_{i=1}^{n} \sqrt{i} = 42$ (advanced)
4. When you "knit" remember to use ezknitr:

```
library(ezknitr)
ezknit("student_folders/yours/activity3_latex.Rmd",
       out_dir = "student_folders/archmiller_a/reports",
       fig_dir = "figures",
       keep_md = F)
```

https://tobi.oetiker.ch/lshort/lshort.pdf (hint: tables on p75)

Project Workflow w/ RStudio & knitr

Example project directory

- data/
  - raw_data/
  - processed_data/
  - output_data/
- manuscript/
  - ms_figures/
  - transmissions/
  - submission/
- output/
  - figures/
- programs/
- project_file.Rproj

Project Workflow w/ RStudio & knitr

- **data/**
  - **raw_data/**
    - survey_data20161227.csv
    - survey_data20161230.csv
    - survey_data20170103.csv
  - **processed_data/**
    - survey_data_all.Rdata
  - **output_data/**
    - model_out.Rdata

Project Workflow w/ RStudio & knitr

- **programs/**
  - a_data_processing.R
  - b_data_analysis.R
  - c_plots.R

Project Workflow w/ RStudio & knitr

- **output/**
  - a_data_processing.html
  - b_data_analysis.html
  - c_plots.html
  - **figures/**
    - eda1.jpg
    - scatter1.jpg

Project Workflow w/ RStudio & knitr

- **manuscript/**
  - ms.Rmd
  - ms.pdf
  - ms.docx
  - **ms_figures/**
    - fig1.jpg
    - fig2.jpg

Project Workflow w/ RStudio & knitr

- **manuscript/**
  - ms.Rmd
  - ms.pdf
  - ms.docx
  - **ms_figures/**
    - fig1.jpg
    - fig2.jpg
  - **transmittals/**
    - **from_john/**
    - ms20170523.docx
    - ms20170625.docx
    - **from_bob/**
    - ms20170626.docx

Project Workflow w/ RStudio & knitr

- **manuscript/**
  - ms.Rmd
  - ms.pdf
  - ms.docx
  - **ms_figures/**
    - fig1.jpg
    - fig2.jpg
  - **transmittals/**
    - **from_john/**
    - ms20170523.docx
    - ms20170625.docx
    - **from_bob/**
    - ms20170626.docx
  - **submission/**
    - ms.pdf
    - fig1.pdf
    - fig2.pdf
    - coverletter.docx

Tips

- Don't use github with large files :-(
- Create new projects in GitHub first, then sync them with RStudio