**Reproducible Research with R and RStudio**

Christopher Gandrud

# Chapter 1 – Introducing Reproducible Research

## I. Introducing Reproducible Research

1. Research vs. Advertising
2. Advertising = presentation documents announce findings and try to convince that they results are correct.
3. Research is “the full software environment, code, and data that produce the results” (Buckheit and Donoho, 1995; Donoho, 2010, 385)
4. Gandrud’s book seeks to provide tools to dynamically combine research with the presentation of the findings.
5. The tools that the author explains and advocates for are

1. A workflow for reproducible research that incorporates the principle of reproducibility throughout the research project.
2. Computer tools to make this workflow possible:
3. R – statistical language for gathering data and performing analysis.
4. LaTeX and Markdown –markup languages used to create documents for presenting results. Slide shows articles, books, and webpages.
5. *knitr* and *rmarkdown* packages, Git, and other tools – dynamically tie data work, analysis, and presentation together so they can be easily reproduced.
6. RStudio – a program that brings all the tools into one place

## II. What is Reproducible Research?

1. *Replicable* is not *reproducible*.
2. For research relying on experiments, replicable is when there is sufficient information for an independent researcher to rerun the experiment with new data and come to comparable results and conclusions.
3. In computational and statistical sciences, replicable means when independent researchers can recreate findings by:
4. Following original procedures to gather data, and
5. Running the computer code.
6. “Really reproducible research” (Peng, 2011, 1226)

In computational sciences, it can be difficult to replicate the original data due to limited resources or because the sample covers the full population of cases. As a “minimum standard,” reproducible (quantitative) research in computational sciences means

**The data and code used to make a finding are available and they are sufficient for an independent researcher to recreate the finding** (Gandrud p. 4)

To me, this statement identifies 3 components/attributes to reproducible research:

1. Data
2. Code
3. Sufficient to recreate the findings.

Gandrud wisely points out that, in practice, it needs to be *easy* to reproduce given the above. Making it easy to reproduce is where the tools that he describes will prove valuable.

## III. Why Should Research Be Reproducible?

Replicability is a key part of scientific inquiry and is what distinguishes science from non-science (Braude, 1979, 2).

1. For science
2. Standard to judge scientific claims
3. Replication – the “ultimate standard” for evaluating scientific claims. Verification through replication. This “requires the complete and open exchange of data, procedures, and materials.” (Stodden, 2009b, 38)
4. Reproducibility enhances replicability – “minimum standard” for judging scientific claims.
   1. Note that “a study can be reproducible and still be wrong.” (Peng, 2014) For example, an original finding could be due to noise in the data and erroneous but also be fully reproducible.
5. Avoiding effort duplication and encourage cumulative knowledge development
6. For you
7. Better work habits
8. Better teamwork
9. Changes are easier
10. Higher research impact

I think reproducible research would also work to maintain credibility and integrity of the work. By withstanding scrutiny, the reputation of the person or the institution upheld.

## IV. Tools of Reproducible Research

1. Reproducible research environment
2. R – programming language for statistics and visualization
3. RStudio – integrated developer environment (IDE) for R that combines R, *knitr*, *rmarkdown*, and markup languages.
4. Cloud storage and versioning – DropBox or Git/GitHub can store data, code, and presentation files; and make this widely available
5. Reproducible research publisher
6. *knitr* and *rmarkdown* – R packages for literate programming. Allow you to combine statistical analysis with presentation of results in one place
7. Literate programming = readable for humans not computers. Combines programming language with documentation language. Think documentation with code inserted rather than code with documentation.
8. Markup languages – formatting syntax; instructions for how to format a document for presentation. The text covers LaTex, Markdown, and some HTML.

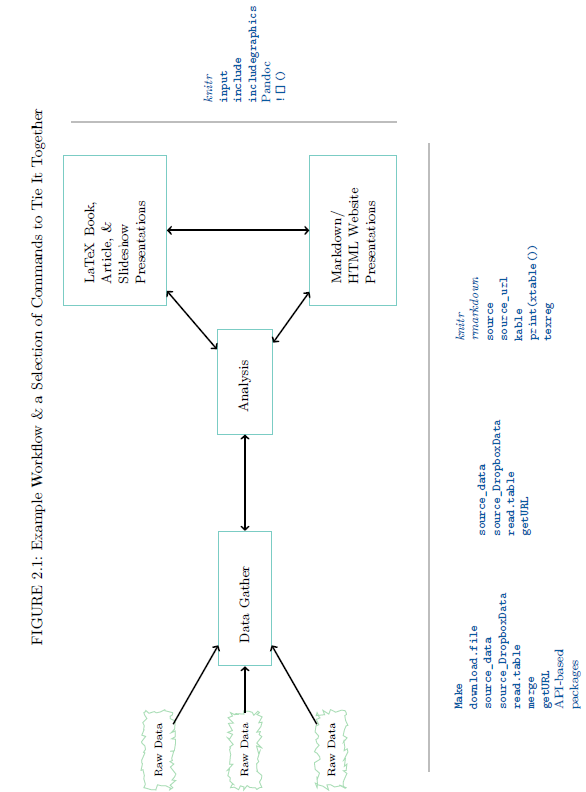
See Gandrud pp. 11–13 for his reasons for advocating for these tools. Mostly it’s that the documentation becomes an active part of the programming process, where other approaches require extra steps and more work to document steps (example SPSS and MS Word). In the author’s view, R/RStudio integrate the tools in such a way to encourage and facilitate reproducible research.

# Chapter 2 – Getting Started with Reproducible Research

“…reproducibility is not an afterthought—it is something that must be built into the project from the beginning.” (Donoho, 2010, 386)

## I. Big Picture of the Workflow for Reproducible Research

1. Data gathering,
2. Data analysis, and
3. Results presentation.



## II. Practical Tips for Reproducible Research

1. Document everything!
2. Tell your reader how you gathered your data, analyzed it, and presented the results.
3. Document your R session info – functionality can differ from version to version. Also, possible in SAS or Stata or others with software updates.
4. Everything is a (text) file
5. The author advocates storing data, analysis, code, write-ups, explanations, etc. in the simplest file format possible – the text file.
6. Comma-separated values (.csv) format, .R files, LaTeX (.tex), and Markdown (.md) files can all be opened by any program that can read text files. Other researchers will always be able to open and view your research.
7. Future-proof especially compared to Word or Excel that change and may lose compatibility.
8. All files should be human readable
9. Comment your code with the goal of communicating its design and purpose.
10. Comment frequently
11. Comment before a block of code to describe what it does
12. Comment on any line of code that is unclear or open to interpretation
13. Comment header should at least include: (1) description of what the file does, (2) the date that it was last updated, and (3) the name of the file’s creator
14. Examples using R syntax (comments start with # or hash mark symbol):

# An entire comment line

2 + 2 # An inline comment after R code

[1] 4

b.

###################################################################

# R source code file used to provide an example of comments

# and headers in R.

# Created by Emile Latour

# Last updated on 8-3-2016

###################################################################

1. Style guides – like in normal language, it’s easier to follow if you use consistent style and formatting. Good ones are:
2. Google’s R Style Guide – <https://google.github.io/styleguide/Rguide.xml>
3. Hadley Wickham’s Style Guide – <http://adv-r.had.co.nz/Style.html>
4. Literate programming – For important pieces of code, it can be useful to go beyond comments in the source file, and display code in presentation text or in an appendix. “Discussion with code.”
5. Explicitly tie your files together
6. Make the links between your files explicit
7. If everything is just a text file, the project is comprised of individual text files with relationships to each other.
8. Tie commands are most dynamic way to link files. See Gandrud p. 28
9. Have a plan to organize, store, and make your files available
10. Need to be organized so that independent researchers can figure it out
11. Organized your files to limit the amount of content in any one file
12. Making discrete parts will make it easier to find errors
13. Cloud services (Dropbox and GitHub) can help organize and make files available.