Transparency and Reproducibility in Social Science Research

Jeremy Freese
Stanford University
jfreese@stanford.edu

"Credibility crisis" in scientific research

With brief episodes from:

- Epidemiology
- Political science
- Economics
- Sociology
- Psychology

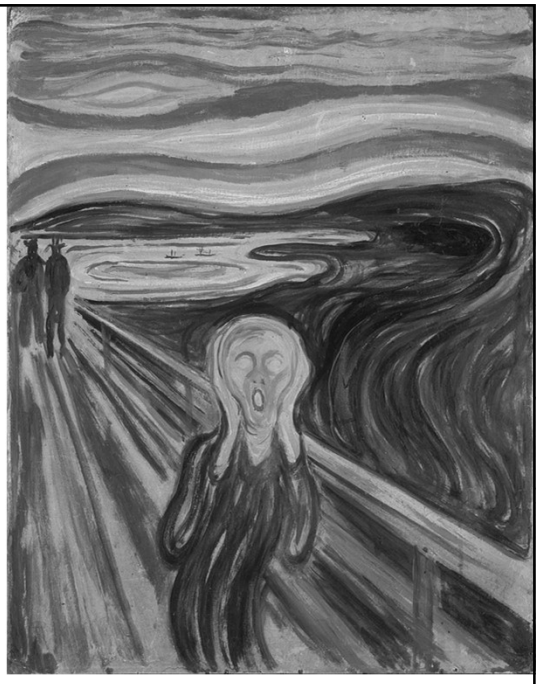


Image: Wikimedia commons [as reproduction of public domain work of art]

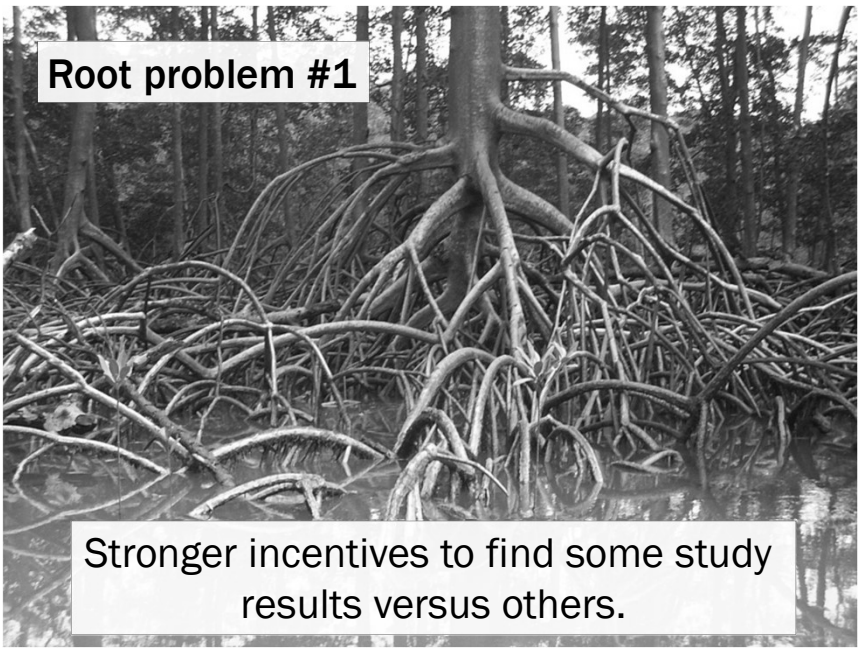


Image: Wikimedia Commons [as unrestricted]

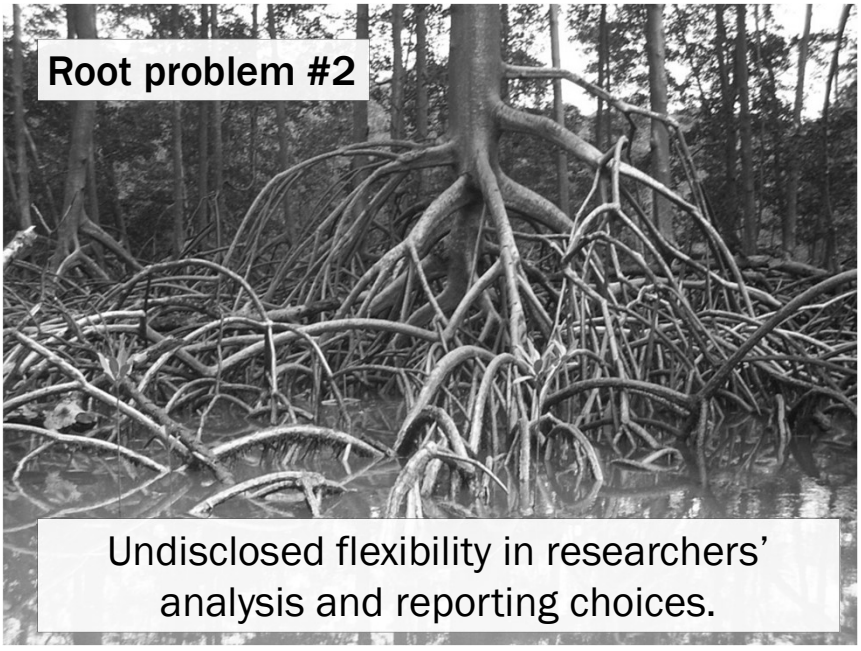


Image: Wikimedia Commons [as unrestricted]

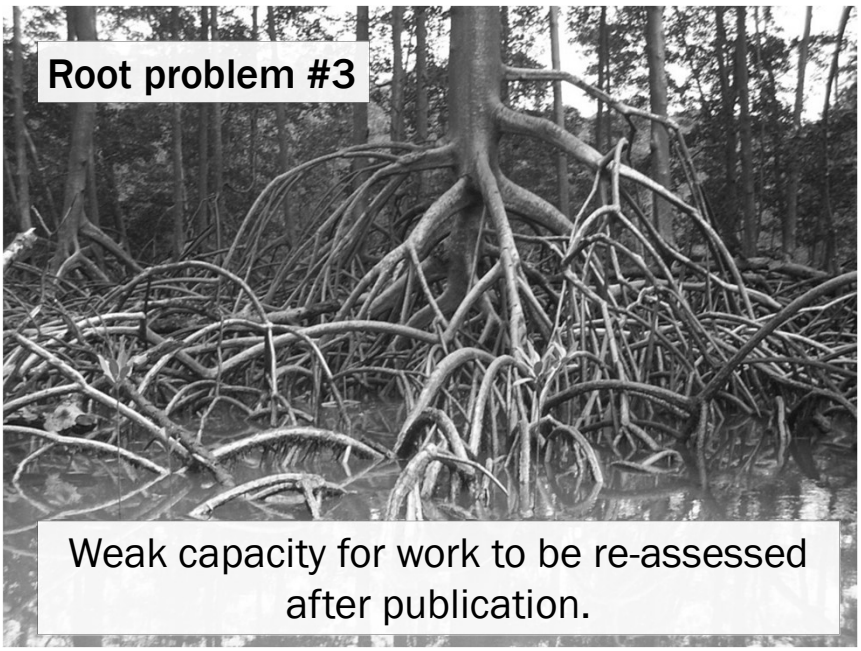
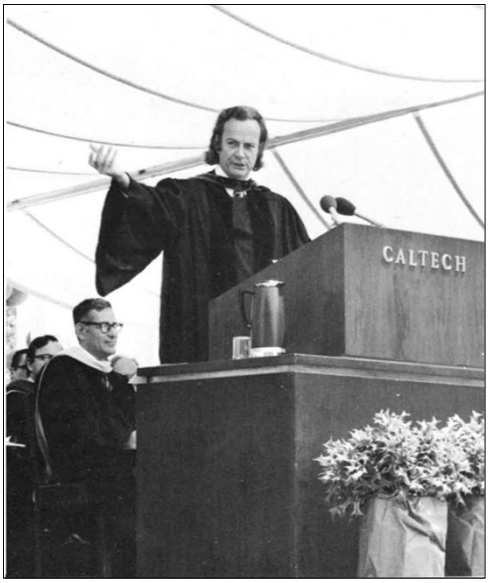


Image: Wikimedia Commons [as unrestricted]

"The first principle is that you must not fool yourself—and you are the easiest person to fool."



Richard Feynman (1974 Caltech commencement)

Image: Wikimedia commons [as public domain]

Collective benefits of efforts to ensure explicit reproducibility of research

1. Increasing credibility of research by showing its availability to be verified (even if verification in practice is rare)

2. Increasing usefulness of work for others

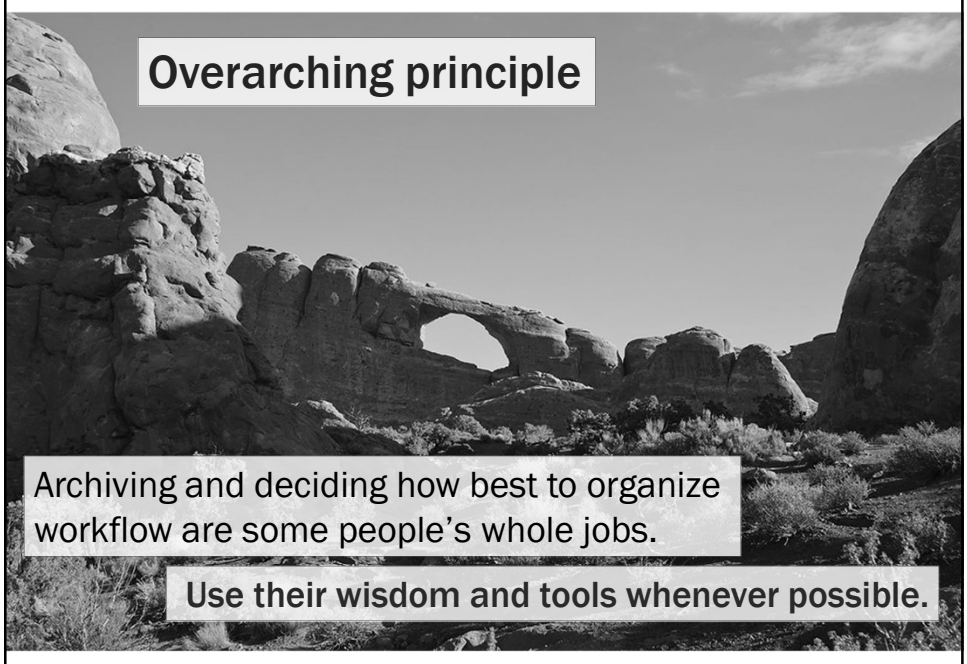


Image: Creative Commons Images [CC-BY 2.0 to BFS Man]

Major aspects of reproducibility initiatives

APSA [political science] Code of Ethics revision (2012): "researchers have an ethical obligation to facilitate evaluation of their evidence based knowledge claims"

1. **Data access:** "provide access to [the data used] or explain why they cannot"
2. **Production transparency:** "full account of the procedures" for collecting or generating data
3. **Analytic transparency:** "clearly explicate the links connecting data to conclusions"

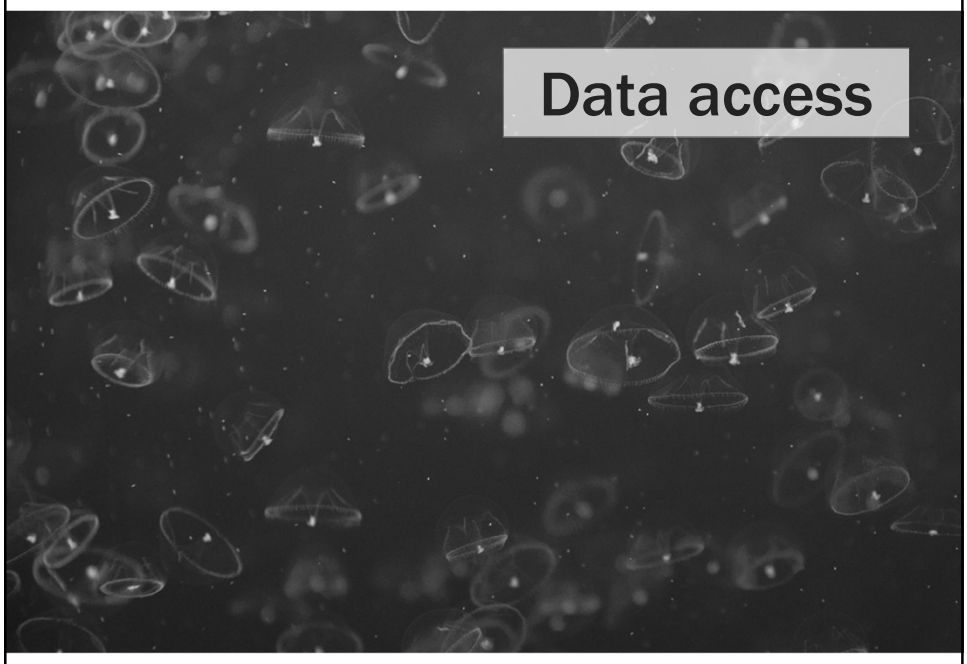
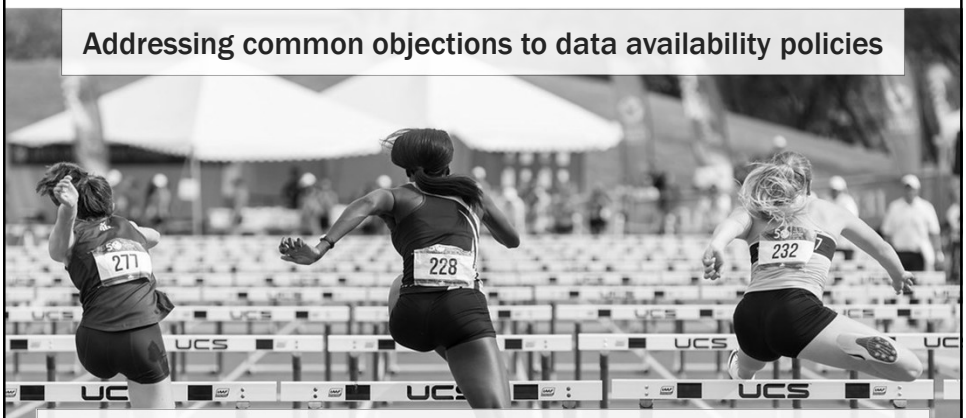


Image: by permission Rebecca McDonald

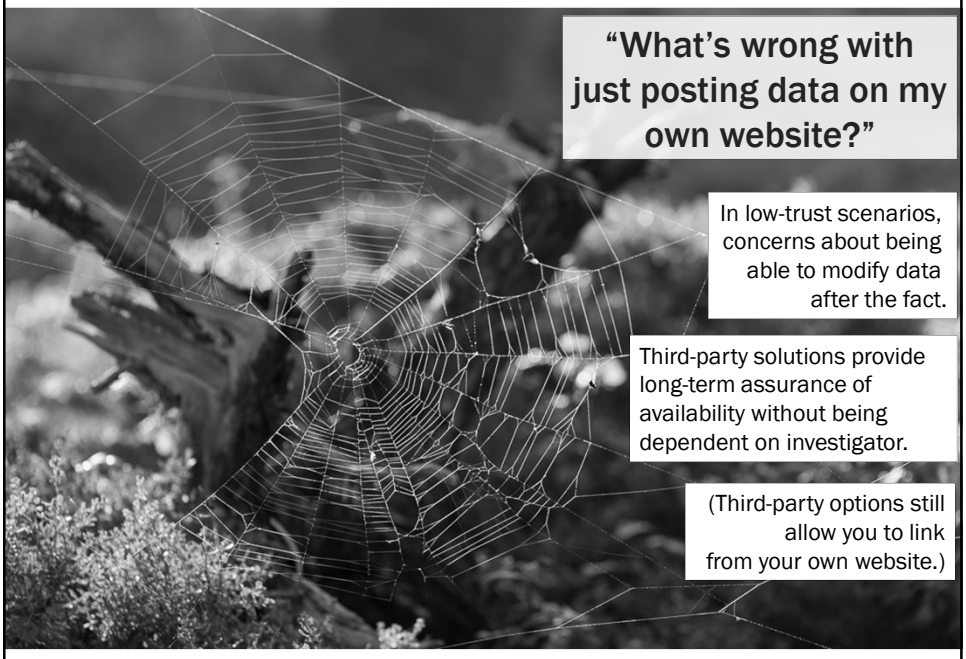
Addressing common objections to data availability policies



1. Any data availability policy in mainstream social science will have the possibility of exemption
2. Possibilities of embargoes or providing narrow replication packages for researchers concerned about "research parasites"
3. Emphasis on appropriate crediting for data collection

[image: CC-BY 2.0 by Matt Duboff via Creative Commons Images]

"What's wrong with just posting data on my own website?"



In low-trust scenarios, concerns about being able to modify data after the fact.

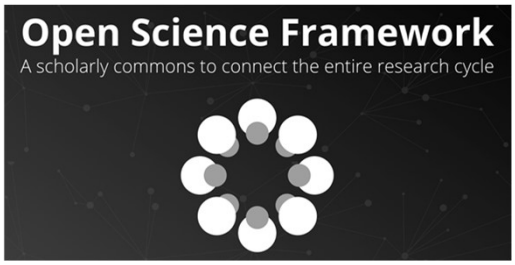
Third-party solutions provide long-term assurance of availability without being dependent on investigator.

(Third-party options still allow you to link from your own website.)

[Image: CC-BY to Andrea Goh via Flickr]

Major social science repository options

(Most closely connected to "professional" data archiving.)



(Most designed around supporting whole project process, but can be used just to deposit data.)

The Dataverse Project

(Most focused on providing archive for storing replication packages associated with particular papers.)

figshare

(Most unfamiliar to your webinar leader.)

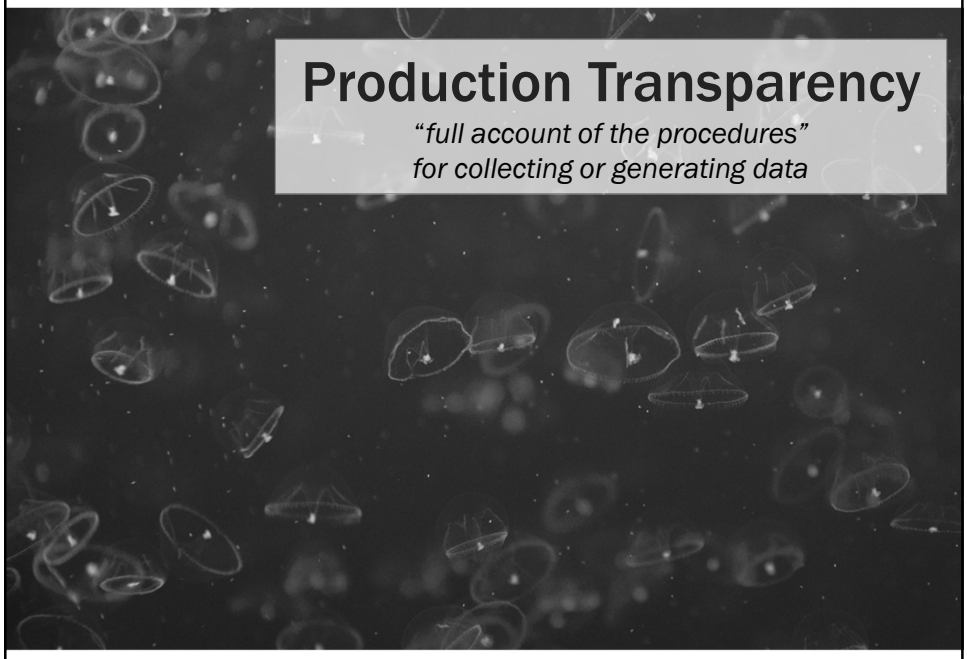


Image: by permission Rebecca McDonald

AAPOR Disclosure Standards

Survey Disclosure Checklist

In accordance with minimum disclosure requirements of the **AAPOR Code of Professional Ethics and Practice**, every survey researcher should disclose each of the following elements in any report that is for public release, or be prepared to disclose this information promptly:

- Name of the survey sponsor
- Name of the organization that conducted the survey
- The exact wording of the questions being released
- A definition of the population under study. What population is the survey designed to represent?
- A description of the sampling frame used to represent this population
- An explanation of how the respondents to the survey were selected
- The total sample size
- The method or mode of data collection
- The dates and location of data collection
- Estimates of sampling error, if appropriate
- A description of how the data were weighted (or a statement that they were not weighted), and any estimating procedures used to produce the final results
- If the survey reports findings based on parts of the sample rather than the total sample, then the size of the subgroups reported should be disclosed

https://www.aapor.org/Standards-Ethics/

AAPOR Reporting Standards

Standard Definitions

Download the full **Standard Definitions Report** (9th edition, 2016)

Download the **Methods of Calculating Eligibility Rates** (August, 2009)

Download **Response Rate Calculator V4.0** (Excel spreadsheet - May, 2016)

Download the **Mail Surveys of Unnamed Persons addendum** (April, 2016)

Background: For a long time, survey researchers have needed more comprehensive and reliable diagnostic tools to understand the components of total survey error. Some of those components, such as margin of sampling error, are relatively easily calculated and familiar to many who use survey research. Other components, such as the influence of question wording on responses, are more difficult to ascertain. Groves (1989) catalogues error into three other major potential areas in which it can occur in sample surveys. One is coverage, where error can result if some members of the population under study do not have a known nonzero chance of being included in the sample. Another is measurement effect, such as when the instrument or items on the instrument are constructed in such a way to produce unreliable or invalid data. The third is nonresponse effect, where nonrespondents in the sample that researchers originally drew differ from respondents in ways that are germane to the objectives of the survey.

Defining final disposition codes and calculating survey outcome rates is the topic for the **Standard Definitions report**. Often it is assumed — correctly or not — that the lower the response rate, the more question there is about the validity of the sample. Although response rate information alone is not sufficient for determining how much nonresponse error exists in a survey, or even whether it exists, calculating the rates is a critical first step to understanding the presence of this component of potential survey error. By knowing the disposition of every element drawn in a survey sample, researchers can assess whether their sample might contain nonresponse error and the potential reasons for that error.

https://www.aapor.org/Standards-Ethics/

Flow diagrams for documenting differences between potential participants and analytic sample

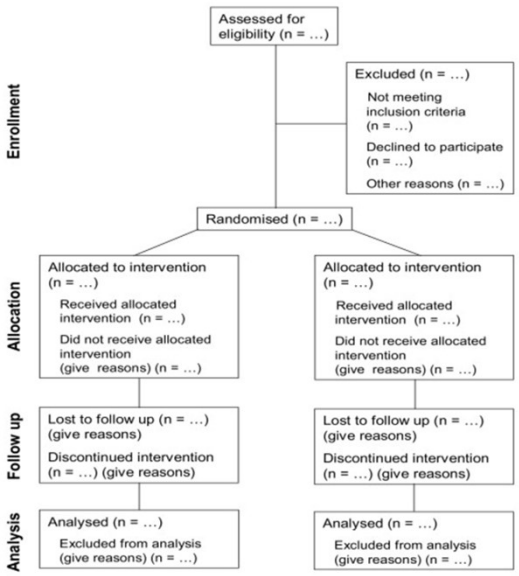
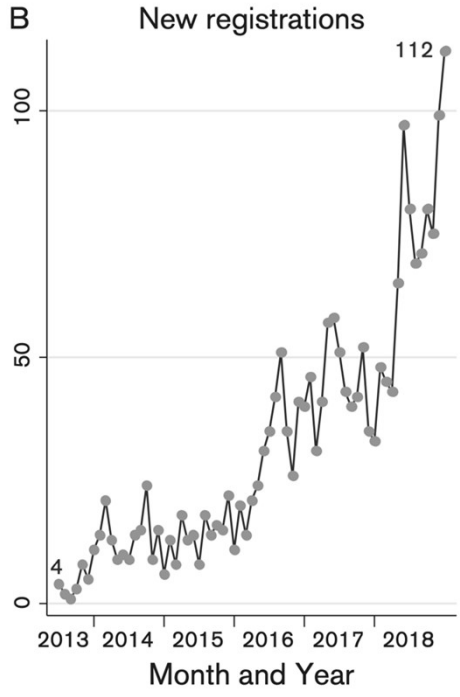


Image: Hopewell et al. 2011 Trials. doi: 10.1186/1745-6215-12-253 [CC:BY]



Study registration ("pre-registration") as a way of verifiably distinguishing confirmatory analyses from *post hoc* explanations

Image: from data in public domain http://dx.doi .org/10.7910/DVN/FUO7FC

Ingredients of a pre-analysis plan

1. Study design
2. Study sample
3. Outcome measures
4. Planned moderators or mediators
5. Families of mean effects
6. Any multiple comparisons adjustments?
7. Planned subgroup analyses?
8. Direction of effect (one-sided tests)
9. Model specifications
10. Time stamp

[Image: CC-BY to Andrea Goh via Flickr]

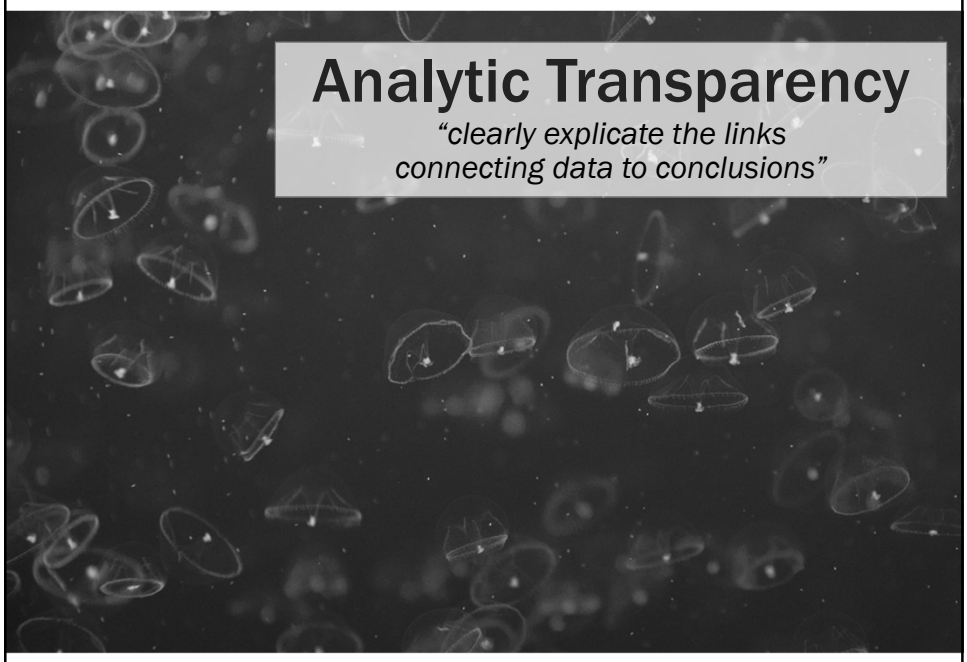


Image: by permission Rebecca McDonald

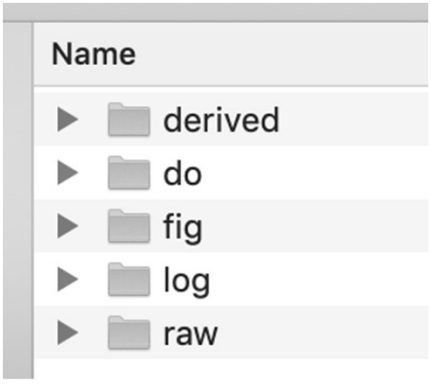
Projects should have a **chain of code** that takes one all the way from raw data to every result provided in a paper.

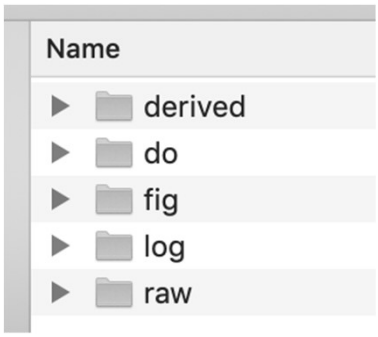
[Image: CC-BY 2.0 by Marc Falardeau]

One project, one folder

Key principles for reproducible workflow #1

Use a clear folder structure for a project





Keep raw data separate from datasets you make using the raw data

Do not change the raw data file
(except as updated by others)

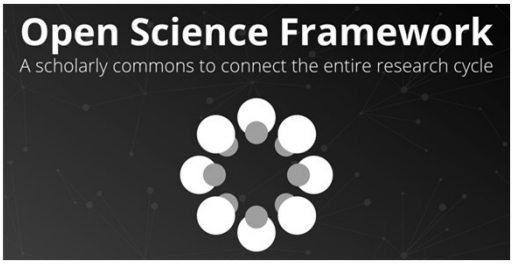
Document data versions used

Use version control

Key principles for reproducible workflow #2

A system for keeping track of files that you may want to revert to.

Also to ensure that collaborators are working with newest version of files and not inadvertently forking each other's work.



Formal version control solutions offer big advantages, but realistically may people prefer "DIY" solutions

Key concepts of formal version control solutions

1. Single folder ("repository") for project
2. You can "check out" parts of a project that you want to revise
3. You "commit" your changes when you are done, so they are checked back in
4. A log is maintained of what changes are made and by whom
5. You can revert to any past commit

Effective DIY version control

Unless absolutely necessary, find a solution in which all collaborators can work out of same folder.

When not necessary, have sancrosanct system by which somebody making changes in their folder announces that they are doing so and provides changes when they are done.

Effective DIY version control

Problem with using names like:
`MakeIncomeVariables-20190815jf.do`
for new versions of files.

If you must do this, keep the simple name (`MakeIncomeVariables.do`) for the new versions of files and use long names (`MakeIncomeVariables-OLD-20190815.do`) for past versions.

Put old files in a subfolder called “archive” or “old.”

You must have script (batch, command, .do) files for everything.

Key principles for reproducible workflow #3

If you use point-and-click menus sometimes (I do), still make sure to paste the syntax that those menus generate so that you can run it as a script file.

Have a single master script [.do] file that calls the other script [.do] files for a project

Key principles for reproducible workflow #4

```
/*
This do file starts with the twin/sibling data and
puts them into a common dataset

creates datasets:
    _temp_TwinSib_extract
    _temp_TwinSib_extract_reshaped (data in long form--row for each grade per kid)

*/

do TwinSibDataAppend

/*
This do file uses _temp_TwinSib_extract_reshaped to compute the
ICC and between/within for boy-boy/girl-girl/opp-sex for twins and siblings

creates:
dataset of results
*/

do TwinSibICC

/*
This do file estimates the models and saves the results as a dataset
*/
```

Script files for complicated projects

1. Use separate .do files for distinct tasks
2. Use separate file(s) for data cleaning and analysis.
3. Save intermediate datasets of different points of variable construction and an ultimate analysis dataset.
   - Save in a separate folder from raw data. I use a folder called “derived” for this purpose.
4. Results should be based exclusively on analysis datasets.

Put “Mission statement” at the top of each script file that documents what it file does

```
/*

This file constructs most individual-level variables in the dataset.
It does not contstruct variables related to an individuals' publications,
which are handled in other do files.

*/

use SociologyPhDsMerged.dta, clear
```

Each table and graph should have its own script file (called from master .do file)

```
* make figures
do FIGURE-BivariatePhDQualityAndJobRating-REVISED.do
do FIGURE-SociologyJobGiniCurve.do
* make tables
do TABLE-BivariatePhDPrestigeAndASRAJS.do
do TABLE-BivariatePhDPrestigeAndAwards.do
do TABLE-BivariatePhDPrestigeAndPlacement.do
do TABLE-BivariatePhDPrestigeAndPrimarySecondaryAuthorship.do
do TABLE-BivariatePhDPrestigeAndPublications.do
```

Use **relative paths** (and working directories) whenever possible.

Key principles for reproducible workflow #5

DO THIS:

```
use SociologyPhDsMerged.dta, clear
```

```
use ../Data/WLS_Marriage_MergedRawData, clear
```

NOT THIS:

```
use “C:\Users\Jeremy\Dropbox\PhD Study\SociologyPhDsMerged.dta”, clear
```

Be mindful of possibility of version changes for any add-on packages (if plausible that they could update and change results).

Key principles for reproducible workflow #6

1. Record version information
2. Consider including add-ons in replication package

ALWAYS set the seed value for the random number generator if routines using random numbers are used.

Key principles for reproducible workflow #7

```
set seed 8675309
```

Minimize ever re-typing or simple copy-pasting of results from statistical packages into a paper

Key principles for reproducible workflow #8

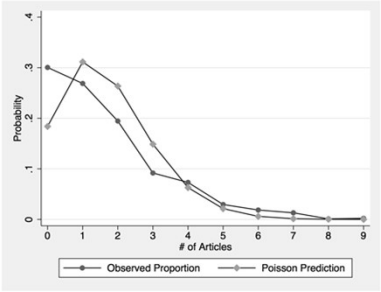
1. Automated solutions for producing regression tables (e.g. estout in Stata)
2. You can program producing your own output in easily copied ways for tables the software package does not do automatically (in Stata, not hard once you understand the display and return/ereturn commands)

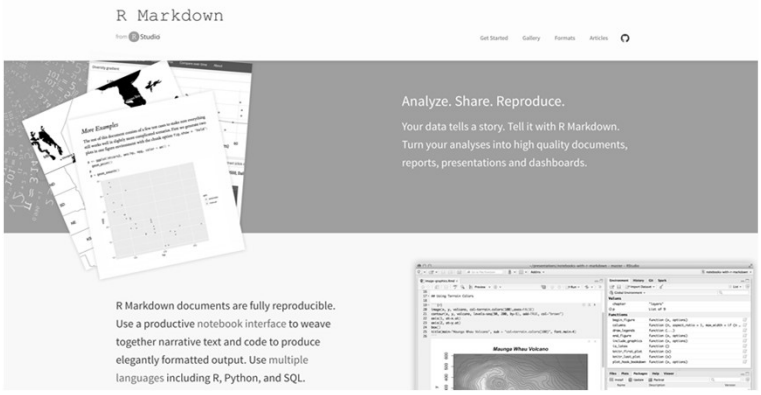
Platforms like LaTeX allow you to integrate results and graphs in ways that automatically update text

```
\noindent The listed values are the observed and predicted probabilities for
observing scientists with 0{\endash}9 publications. These can be plotted
with \texttt{graph twoway}:

\begin{stlog}
\input{stlog/c08_poi_drawplot_art.log.tex}\nullskip
\end{stlog}

\begin{center}
\includegraphics{08psnpred.pdf}
\end{center}
```

The listed values are the observed and predicted probabilities for observing scientists with 0–9 publications. These can be plotted with `graph twoway`:

```
. graph twoway connected psnobeq psnpreq psnval, ///
>     ytitle("Probability") ylabel(0(.1).4) ///
>     xlabel(0(1)9)
```





Dynamic document generation (RMarkdown, features in Stata 15+) allow you to produce a formatted document from text and code contained in a single command file.

Document code thoroughly…

Key principles for reproducible workflow #9

```
*************************************************************
** HEIGHT AND WEIGHT (needed for BMI measure)
*************************************************************

** 1992 survey -- note: irregularly named grade and sib vars

    clonevar height92 = mx010rec if rtype == "g"
    replace height92 = nx024rec if rtype == "s"
    replace height92 = height92 / 100
    * recode outlier values of height -- 6'8" or above
        replace height92 = .z if height92 >= 80 & height92 < .
    label variable height92 "Height in 1992 (in, self-rep)"

    clonevar weight92 = mx009rer if rtype == "g"
    replace weight92 = nx023rer if rtype == "s"
    label variable weight92 "Weight in 1992 (lbs, self-rep)"
```

…but make code self-documenting whenever possible

Key principles for reproducible workflow #9

```
gen born_date = rabdate
format born_date %td

gen died_date = raddate
format died_date %td

gen first_intv_date = .
forvalues i = 1(1)11 {
    replace first_intv_date = r`i'iwend ///
        if first_intv_date == . & r`i'iwend < .
}
format first_intv_date %td
```

Use mnemonic variable names

Use variable names long enough to be clear (but not excessive)

Write clean code in script files

Key principles for reproducible workflow #10

1. Other people (including future you) can figure out what code is supposed to do
2. Multiple people can work on the same code in a productive way
3. Minimizes errors even for work without collaborators
4. Avoids the problem of *code-shyness* as a reason to be reluctant to share work.

Google's style guide for R

GOOD:

```
if (debug)
x[1, ]
```

BAD:

```
if ( debug )  # No spaces around debug
x[1,]  # Needs a space after the comma
```

Curly Braces

An opening curly brace should never go on its own line; a closing curly brace should always go on its own line. You may omit curly braces when a block consists of a single statement; however, you must *consistently* either use or not use curly braces for single statement blocks.

```
if (is.null(ylim)) {
  ylim <- c(0, 0.06)
}
```

xor (but not both)

```
if (is.null(ylim))
  ylim <- c(0, 0.06)
```

Always begin the body of a block on a new line.

BAD:

```
if (is.null(ylim)) ylim <- c(0, 0.06)
if (is.null(ylim)) {ylim <- c(0, 0.06)}
```

Surround else with braces

An `else` statement should always be surrounded on the same line by curly braces.

```
if (condition) {
```

Use predetermined maximum line length (e.g. 80 characters) and stick to it as much as you can

Clean code tip #1

```
sts graph, ///
    hazard noboundary tmax(35) ///
    ploto(lc(cranberry)) width(1) ///
    xlabel(0(5)35, grid) ///
    xtitle("Years since marriage") title("Implied hazard function") ///
    ytitle("Hazard rate", size(medlarge)) ///
    name(divorce_hazard, replace)
```

Use all lowercase variable names (convert dataset at the outset if necessary)

Clean code tip #2

(Real example where I didn't do this and regretted it)

```
* versions of these variables used in tables

    capture drop PhDTableCat
    recode PhDRatingCat (1=1) (2=1) (3=2) (4=3) (*=.), gen(PhDTableCat)
    label variable PhDTableCat "PhD Institution Quality"
    label define PhDTableCat 1 "< 3 or unrated" 2 "3-3.9" 3 "4+", replace
    label values PhDTableCat PhDTableCat

    capture drop GoodSocJob
    gen GoodSocJob = GoodJob
    replace GoodSocJob = 0 if RatedSocJob != 1

    capture drop EliteSocJob
    gen EliteSocJob = EliteJob
    replace EliteSocJob = 0 if RatedSocJob != 1
```

Use indentation and vertical white space to make your script file easy to follow.

Clean code tip #3

```
// PRIVATE UNIVERSITY

    capture drop private
    gen private = Private
    label variable private "Private university?"

// HIRED UP VARIABLE

    capture drop hiredup
    gen hiredup = (HiredUp == 1)

// RANDOM SUBSAMPLE VARIABLE

    capture drop randomsubsample
    gen randomsubsample = (TenPctLookup == 1)
    label variable randomsubsample "random subsample variable"
```

Use macros, loops, and programs to eliminate redundancy in code

Clean code tip #4

```
gen first_intv_date = .
forvalues i = 1(1)11 {
      replace first_intv_date = r`i'iwend ///
            if first_intv_date == . & r`i'iwend < .
}
```

Larger point: don't repeat yourself in code whenever possible because it is easy to introduce errors when re-typing or revising