

Lecture 5

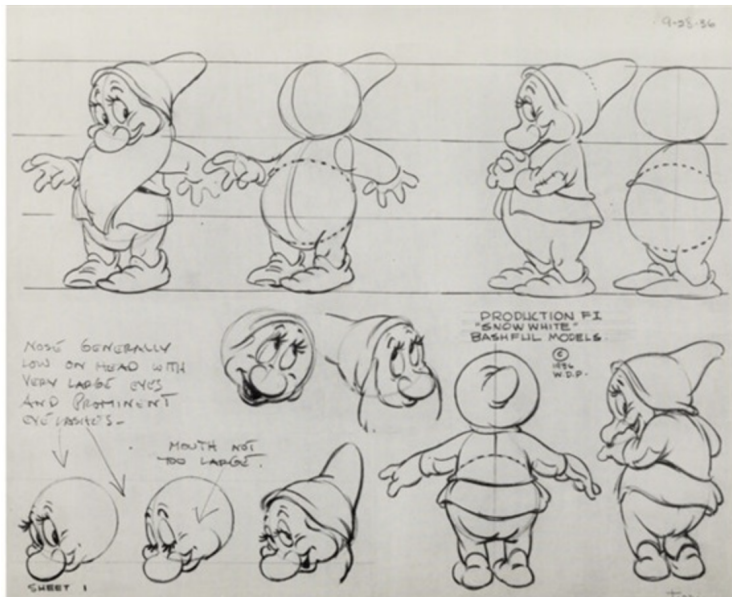
Brooke Anderson

Barriers to reproducible papers

In theory:

- ▶ Necessary to know what to include and how to share it
- ▶ Data are very large
- ▶ Data cannot ethically or legally be shared (e.g. private health data)
- ▶ Necessary to know how to share important information about the environment in which the code was run

Barriers to reproducible papers



Barriers to reproducible papers

In practice:

- ▶ I don't want to share the code / data
 - ▶ The code's too messy
 - ▶ Someone might scoop my work
 - ▶ Someone might find a bug
- ▶ The code's available, but very hard to understand
 - ▶ I'll put the code up, but it's not worth taking the time to clean it first
 - ▶ Subconsciously creating a barrier to prevent easy reproduction

What to include with a computationally reproducible paper?

Remember, to be computationally reproducible, the reader must be able to start from the data that were collected and arrive at the same results, including tables and figures.

- ▶ All data used in analysis in the paper
- ▶ All code to get from these data to the results shown in the paper
- ▶ Any environmental information (e.g., package versions) that are necessary for the code to generate the results

Sharing code and data

There are a few options for how you share the code and data for the paper, including:

- ▶ Supplemental files through the journal's website
- ▶ GitHub
- ▶ Scientific repository, like [examples, NIH]
- ▶ Personal / academic website
- ▶ R package plus script

What to include with a computationally reproducible paper?

It's also helpful to include:

- ▶ Adequate documentation for the user to recreate the results from the data using the code and environmental information
- ▶ Guidance on how the code can be reused (license)

Data to include

[Image: raw data to processed data to right before tables / figures]

Extremely large data

- ▶ Repositories built for data curation might be better suited for storing data that methods with data size limits ([example])
- ▶ Some repositories might have APIs—data can be downloaded programatically (from the code script), so the code could be connected to the data in this way
- ▶ Some repositories allow options to toggle private versus public (similar to GitHub)

AIR POLLUTION

Differential Mortality Risks Associated With PM_{2.5} Components

A Multi-Country, Multi-City Study

Masselot, Pierre^a; Sera, Francesco^{a,b}; Schneider, Rochelle^{a,c,d}; Kan, Haidong^e; Lavigne, Éric^{f,g}; Stafoggia, Massimo^h; Tobias, Aurelio^{i,j}; Chen, Hong^k; Burnett, Richard T.^k; Schwartz, Joel^l; Zanobetti, Antonella^l; Bell, Michelle L.^m; Chen, Bing-Yuⁿ; Guo, Yue-Liang Leonⁿ; Ragettli, Martina S.^o; Vicedo-Cabrera, Ana Maria^{p,q}; Åström, Christofer^r; Forsberg, Bertil^r; Íñiguez, Carmen^{s,t}; Garland, Rebecca M.^{u,v,w}; Scovronick, Noah^x; Madureira, Joana^{y,z}; Nunes, Baltazar^{aa,bb}; De la Cruz Valencia, César^{cc}; Hurtado Diaz, Magali^{cc}; Honda, Yasushi^{dd,ee}; Hashizume, Masahiro^{ff}; Ng, Chris Fook Cheng^j; Samoli, Evangelia^{gg}; Katsouyanni, Klea^{gg,hh}; Schneider, Alexandraⁱⁱ; Breitner, Susanne^{ii,jj}; Rytí, Niilo R.I.^{kk,ll}; Jaakkola, Jouni J.K.^{kk,ll,mm}; Maasikmets, Marekⁿⁿ; Orru, Hans^{oo}; Guo, Yuming^{pp}; Valdés Ortega, Nicolás^{qq}; Matus Correa, Patricia^{rr}; Tong, Shilu^{rr,ss,tt,uu}; Gasparrini, Antonio^{a,c,vv}

Author Information 

Epidemiology 33(2):p 167-175, March 2022. | DOI: 10.1097/EDE.0000000000001455

https://journals.lww.com/epidem/Abstract/2022/03000/Differential_Mortality_Risks_Associated_With_PM2_5.3.aspx

Sensitive or protected data

Differential mortality risks associated with PM_{2.5} components: a multi-country, multi-city study

Masselot P, Sera F, Schneider R, Kan H, Lavigne E, Stafoggia M, Tobias A, Chen H, Burnett RT, Schwartz J, Zanobetti A, Bell ML, Chen BY, Leon Guo YL, Ragettli MS, Vicedo-Cabrera AM, Astrom C, Forsberg B, Iniguez C, Garland RM, Scovronick N, Madureira J, Nunes B, De la Cruz Valencia C, Hurtado Diaz M, Honda Y, Hashizume M, Fook Cheng Ng C, Samoli E, Katsouyanni K, Schneider A, Breitner S, Rytli NRI, Jaakkola JJK, Maasikmets M, Orru H, Guo Y, Valdes Ortega N, Matus Correa P, Tong S, **Gasparrini A**
Epidemiology. 2022;**33**(2):167-175. DOI: 10.1097/EDE.0000000000001455. PMID: 34907973



Article
[Download File](#)



Supplemental material
[Download File](#)

Updated R code

Available at https://github.com/PierreMasselot/Paper--2022--Epidemiology--PM2.5_components

Read the README at the bottom of the webpage and download the files using the *Clone* or *Download* button

http://www.ag-myresearch.com/2022_masselot_epidemiol.html

Sensitive or protected data

Data and Results

Data are not available currently due to restricted data sharing agreement between the collaborators of this study. Therefore, the code is not fully reproducible.

Data are normally included in a subfolder *Data*. It should contain:

- Mortality and pollution data, stored in a list of city-specific `data.frames`. Also contains a descriptive `data.frame` with one line for each city.
- PM2.5 components, stored in one csv files per year of data. Each csv file contains one line per city.
- City-specific characteristics, stored as a `data.frame` with one line per city.

From the GitHub repository

https://github.com/PierreMasselot/Paper--2022--Epidemiology--PM2.5_components

Masselot P, et al. Differential mortality risks associated to PM2.5 components: a multi-country multi-city study. *Epidemiology*. 2022.

Sensitive or protected data

- ▶ Can include mock dataset (same format, different numbers) to use to test and demonstrate the function of the code
- ▶ Can use `.gitignore` to include the data in a git repo on a local computer (or a secure server you're working on) but not push it to remote versions of that repo, like one on GitHub
- ▶ When working on a server for security reasons, it can be helpful to have RStudio Server installed, as this provides a friendlier interface for working with git version control on the server (avoids having to do much from the terminal)

Sensitive or protected data

If you look through the repository for Masselot P, et al. 2022, you'll see that they listed all data in the ".gitignore" file. This would allow them to have the data on their local computer but not share it publicly with the rest of the repo.

```
# Session Data files
```

```
.RData
```

```
**/*.RData
```

```
# RStudio files
```

```
.Rproj.user/
```

```
# produced vignettes
```

```
vignettes/*.html
```

```
vignettes/*.pdf
```

```
# Other
```

```
Data/
```

Processed data versus raw data

A lot of health-related data can require extensive pre-processing to extract the information you need to answer a scientific question.

It is helpful to include the original, raw data and the code used to get to the processed data.

Saving intermediate data

It can be useful to save intermediate versions of the data, as the code moves from raw data to the final figures, tables, and other results. Intermediate data can include:

- ▶ Processed data (e.g., if raw data are from flow cytometry, the results after gating and counting the data)
- ▶ Data points that are plotted in figures (i.e., data at the point immediately before it is graphed)

Note that this is in addition to the raw data.

For these intermediate data sets, it's helpful if you save them in plain text formats (e.g., “.csv” or “.txt” file).

(Sandve et al.)

Types of shared documents

There are several formats that you can use to share the files that are needed to make an article reproducible. These include:


- ▶ A set of unstructured files
- ▶ A compressed directory, providing both the files and their organization into a directory. This can include an online repo maintained through a version control platform.
- ▶ A container, including everything provided with a compressed directory as well as the code for the software needed to run the code
- ▶ An interactive platform, where the user can re-run code, as well as adapt and explore the code and data, through a web-based platform without needing to download the data


Compressed directory


Full directory can be shared in one compressed file.


Includes not only all the files, but also maintains the organization of those files into a directory.

Compressed directory

 **2019_vicedo-cabrera_Epidem_Rcodedata** Public Watch 3

 master ▾



 1 branch















 0 tags

[Go to file](#)

[Add file ▾](#)

[Code ▾](#)

 **gasparrini** Fix issue with dates in R 4.3.0 update e341f8b on Apr 24  10 commits

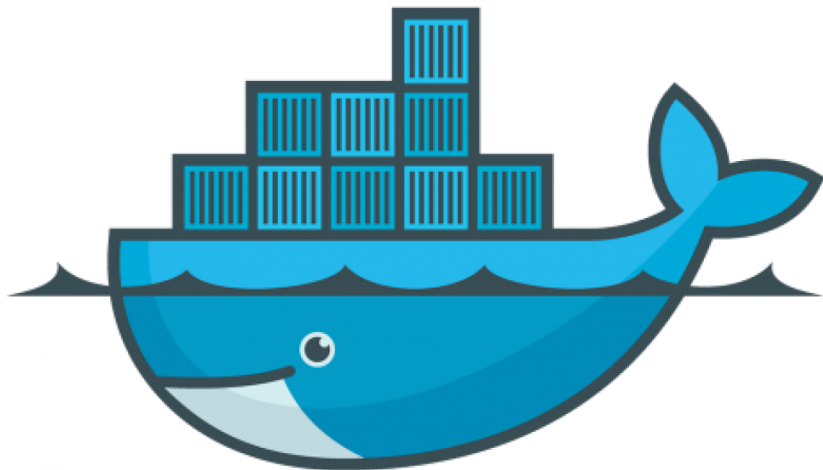
 .gitignore	Initial commit	4 years ago
 01EstimationERassociation.R	Fix issue with dates in R 4.3.0 update	3 months ago
 02ProjExpHOutSeries.R	Fix issue with dates in R 4.3.0 update	3 months ago
 03DownscalingCalibration.R	Fix issue with dates in R 4.3.0 update	3 months ago
 04_05_06ExtCurveProjUncert.R	Fix issue with dates in R 4.3.0 update	3 months ago
 07DemChangesAdapt.R	Fix issue with dates in R 4.3.0 update	3 months ago
 2019_vicedo-cabrera_Epidem_Rco...	Initial commit	4 years ago
 Codebook.pdf	Add the codebook	4 years ago
 README.html	Update data and fix code	3 years ago
 README.md	Update reference in README	4 years ago
 fhempel.r	Fix issue with dates in R 4.3.0 update	3 months ago
 Indn_obs.csv	Update data and fix code	3 years ago
 Indn_rcp4p5.csv	Fix bug in hempel	4 years ago
 Indn_rcp8p5.csv	Fix bug in hempel	4 years ago

Compressed directory

There are advantages to solutions that let you share everything as a directory:

- ▶ Relative pathnames work in scripts (because directory structure is preserved)
- ▶ If you set up your directory as an R Project, it's designed to be self-contained (can interact with other things in the directory but doesn't rely on the computer's file directory outside of that)

Container



Container

Includes all files and directory organization. Also includes all software needed to run code (including operating system?).

Downsides:

- ▶ Can be very, very large
- ▶ Can be somewhat of a black box, in terms of dissecting what's in the container
- ▶ Can be hard to adapt parts for new tasks (often relies on code that might be out of date)

Considering containers

Containers can get very large—stores not just info on the versions of each piece of software used, but full source code (?)

Works somewhat like a black box—harder for users to explore, change, adapt

Can be useful, though, if using a large collection of different open-source software (command line tools, R, Python, etc., all in the same pipeline)

Interactive platform

Not only shares the data and code, but allows it to be explored (e.g., change and test code) on the platform.

- ▶ NSF Disaster data platform
- ▶ Galaxy?
- ▶ Immunoport?

GitHub as a method for reproducible papers

There are advantages and disadvantages to each method. However, if you are new to making papers computationally reproducible, sharing through GitHub or a similar platform might be a good place to start.

GitHub as a method for reproducible papers

- ▶ Allows you to set up the online copy of the project directory while you're working on the project
- ▶ This can facilitate collaborative work on the project, as well as provide a form of backup as you work
- ▶ You can toggle the GitHub repo from private while you work on your project to public once you submit it for publication
- ▶ This system is widely used, so there are many resources available to help you navigate it

Leveraging a sharing system before you publish

Some of these methods are platforms that you can use before you publish, and they can help you work on code-heavy projects within your team. GitHub is one example, but some agency-funded platforms now combine repositories with workspaces for coding.

For GitHub (and some of the agency-funded platforms), you can toggle between having your data and code be public or private. This allows you to work privately before you publish, but then switch the repository to public when you publish the paper.

This can be much easier than trying to figure out how to post everything after publication. It also provides some helpful tools as you work on the paper.

GitHub as a method for reproducible papers

Resources for git and GitHub with R and RStudio:

- ▶ [Jenny Bryan's webpage]
- ▶ [something in R for data science?]
- ▶ [something on RStudio site]

Versions of software

R base code changes from one version to another.

The code within packages can change even more.

Seems to be a particular source of difficulty in reproducing for work that involves Bioconductor packages.

Versions used in original analysis

You can record the information on all versions of the R software you used in your analysis (base R and packages).

`session_info` from `devtools` (or `sessionInfo` from base R)

Include this as a line in an RMarkdown file (often at the end), and it will print out the information on versions.

Versions used in original analysis

Help user recreate the “environment” that code ran in originally

If you have version numbers, you could do it by hand (all old versions of CRAN packages, for example, are archived and available for you to install, although it takes a bit more work than installing the current version of the package).

[Packrat alternatives—renv]

Deprecated functions

Future-proof your code (or at least make it robust for longer)

`spread {tidyr}`

[R Documentation](#)

Spread a key-value pair across multiple columns

Description

lifecycle **superseded**

Deprecated functions

`.drop` For `count ()`: if `FALSE` will include counts for empty groups (i.e. for levels of factors that don't exist in the data).

lifecycle deprecated For `add_count ()`: deprecated since it can't actually affect the output.

Absolute versus relative pathnames

When you want to tell R about a file, you can either give it the file's full address (from the root directory of your computer) or you can give it directions from the working directory. The first is an **absolute** filepath and the second is a **relative** filepath.

```
read_csv("/Users/ganders/my_proj/data/my_data.csv")  
read_csv("data/my_data.csv")
```

Which will be robust when someone else tries to run the code on their own computer?

Reproducing randomness

[Dice, roulette wheel]

Reproducing randomness

A MILLION Random Digits

WITH
100,000 Normal Deviates

BY THE

RAND CORPORATION

TABLE OF RANDOM DIGITS

1

00000	10997	32533	76520	13586	24673	54876	80959	09117	39292	74945
00001	37542	04805	64894	74296	24805	24037	20436	10402	00822	91665
00002	08422	08893	19645	08203	22309	02560	13593	34764	33590	33996
00003	99019	02529	03076	70715	28311	31185	88676	74397	04436	27659
00004	12807	99970	80157	36147	64032	36653	98951	16877	12171	76833
00005	66065	74717	34072	76850	36697	36170	65813	39885	11199	29170
00006	31060	10805	45571	82408	35203	43614	86799	07439	23403	09732
00007	85269	77622	02051	05692	88665	74818	73053	85347	18632	88579
00008	63573	32135	05335	47048	90553	57548	28468	28709	83491	26524
00009	73796	45753	03529	64778	35808	34282	60935	20344	35272	88435
00010	98520	17767	14905	68607	22109	40558	69970	83433	50500	73998
00011	11805	05431	39808	27732	50723	68248	29405	24201	52775	67851
00012	83452	99634	06288	98083	12746	70078	18475	40610	68711	77817
00013	88685	40200	86507	28401	28768	67951	90364	78493	29609	11062
00014	99594	67348	87517	64669	91826	68928	93785	61365	24378	24113
00015	65481	17674	17468	50950	58047	76974	73029	57186	40218	16544
00016	80124	35635	17777	08015	45318	22374	21115	78253	14388	53783
00017	74350	99817	77402	77214	43238	00210	45521	64237	96286	02655
00018	69916	26803	66252	29148	36936	87203	76621	13990	94400	96418
00019	09893	20505	14225	68514	46427	56788	96297	78822	54382	14598
00020	91499	14523	68479	27686	46162	83554	94750	89923	37089	20048
00021	80336	94598	26940	26854	70297	34135	53140	33340	42000	82341
00022	44104	81049	85157	47954	32970	26575	57600	40881	22223	06413
00023	12350	73742	11100	02040	12860	74697	96644	89439	28707	25815
00024	83606	49329	16050	34684	40219	32563	43651	77082	07207	21790
00025	61196	90446	26457	47774	51924	33729	65394	59593	42582	60527
00026	15474	43266	95270	79953	59267	83848	82396	10118	33211	59466
00027	94557	28573	67897	54287	54622	44431	91190	42592	92927	45973
00028	42481	16213	97244	08721	18648	48767	00701	12059	25701	46670
00029	23523	78317	72208	89837	68935	91416	28252	29663	05522	82562
00030	04493	52494	75246	33824	45862	51025	61962	79335	65337	12472
00031	00549	97054	64051	88159	96119	63896	54692	82391	23287	28529
00032	33963	15307	26898	05254	33351	35462	77974	50024	90103	28233
00033	59608	08391	45427	26842	83609	49700	13021	24892	78503	20106
00034	46058	85286	01390	92286	77281	44077	92910	82847	70617	42941
00035	32179	00597	87379	25241	05567	07007	86743	71517	85394	14638
00036	69234	61406	20117	45204	15066	60000	18743	92423	87118	96338
00037	18045	41430	01758	75379	40419	21585	66674	34860	64962	85207
00038	45155	14938	19476	07246	43667	94543	59047	90033	20826	69541
00039	94864	31994	36168	10631	24888	81553	01540	35456	05014	51176
00040	98086	24826	45240	28404	44999	08896	38094	73407	35441	31880
00041	33185	16232	41941	50949	89435	48581	88695	41994	37548	73042
00042	90951	00406	96382	70774	29151	23387	25016	25288	94634	61171
00043	79752	49140	71961	28296	69861	02591	74852	20339	00387	59579
00044	18633	32537	06145	06571	31010	24674	04555	61427	77938	91836
00045	74029	43902	77557	32270	97790	17119	52527	58021	80814	51748
00046	54178	45611	80993	37143	05235	12969	58127	19255	36040	80324
00047	11664	49883	52079	84827	58281	71339	09973	33440	88481	23356
00048	48324	77928	31249	44710	02285	26870	32027	37546	13029	98994
00049	69074	94128	87637	91978	35584	04401	10518	21615	01848	76938

Reproducing randomness



Reproducing randomness

pseudorandom number generator

Reproducing randomness

You might be using random numbers if you:

- ▶ are sampling
- ▶ are using the Monte Carlo method / simulations
- ▶ are doing Bayesian statistics

Reproducing randomness

Setting seeds when code includes random number generation

Seed can be any integer

Pseudorandom number generator—depends on an initial value (the seed)

Reproducing randomness

If you don't set a seed, you will get different results when you run code that involves random number generation, because of the randomness involved.

```
sample(1:5)
```

```
## [1] 1 5 2 3 4
```

```
sample(1:5)
```

```
## [1] 2 1 4 5 3
```

Reproducing randomness

If you set the same seed each time before you run that code, you will get the same “random” results:

```
set.seed(100)  
sample(1:5)
```

```
## [1] 2 3 5 4 1
```

```
set.seed(100)  
sample(1:5)
```

```
## [1] 2 3 5 4 1
```

Reproducing randomness

```
set.seed(100)  
sample(1:5)
```

```
## [1] 2 3 5 4 1
```

```
sample(1:5)
```

```
## [1] 1 2 4 3 5
```

RMarkdown for journal articles

articles templates

How to collaborate when using RMarkdown

Guidelines:

- ▶ Be sure even treats the output (pdf, HTML, Word) as read-only. Only make edits to the RMarkdown file
- ▶ If using git and GitHub may help to exclude the output file from tracking (e.g., include in .gitignore)

Platforms:

- ▶ GitHub
- ▶ Overleaf?