Preparing code and data for computational reproducibility: a hands-on workshop

Society for the Improvement of Psychological Science, Rm: Maxima in Engels
Sunday, July 7, 2019

April Clyburne-Sherin, Director of Scientific Outreach, Code Ocean

http://bit.ly/sips-workshop

http://bit.ly/sips-workshop

Workshop POP

- **Purpose:** To introduce **skills and tools** in organization, documentation, automation, containerization, and dissemination of research.

- **Outcome:** You feel more confident applying relevant skills and tools to guide the sharing of your research code and data.

- **Process:** You adapt & apply some skills or tools we discuss today next time your share or publish your research.

Agenda

13:45 Reproducibility guidance

Organization

14:15 Exercise 1: Data collection

14:30 BREAK (10 minutes)

14:40 Exercise 2: One repository
Exercise 3: Separate code & data

Documentation

Exercise 4: Specify environment
Exercise 5: Specify dependencies
Exercise 6: Containerization
Demo: Literate programming
Demo: Create a README file + data dictionary

15:15 BREAK (30 minutes)

Automation

15:45 Exercise 7: Create a master script
Exercise 8: Create relative paths

Dissemination

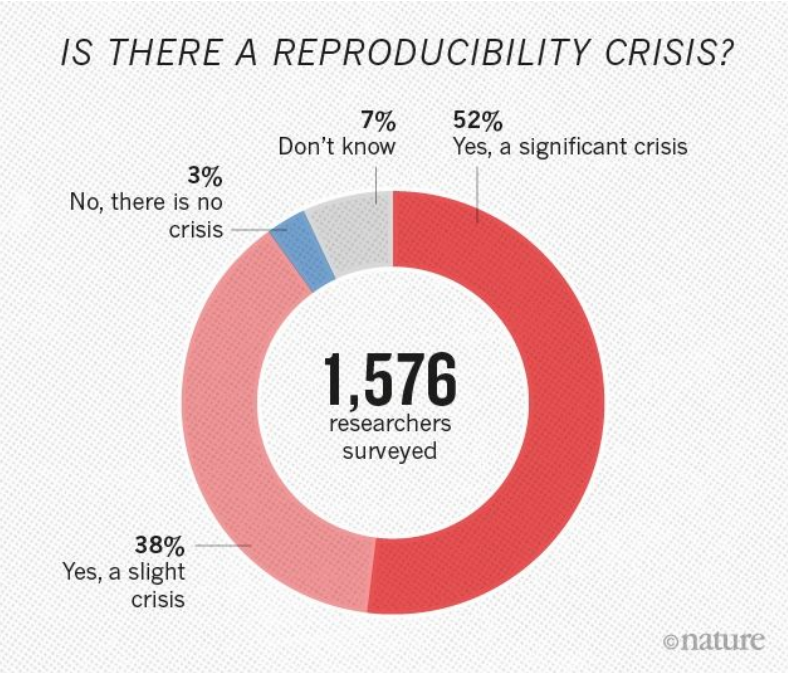
Exercise 9: Specify a license
Exercise 10: Share your code!

Icebreaker

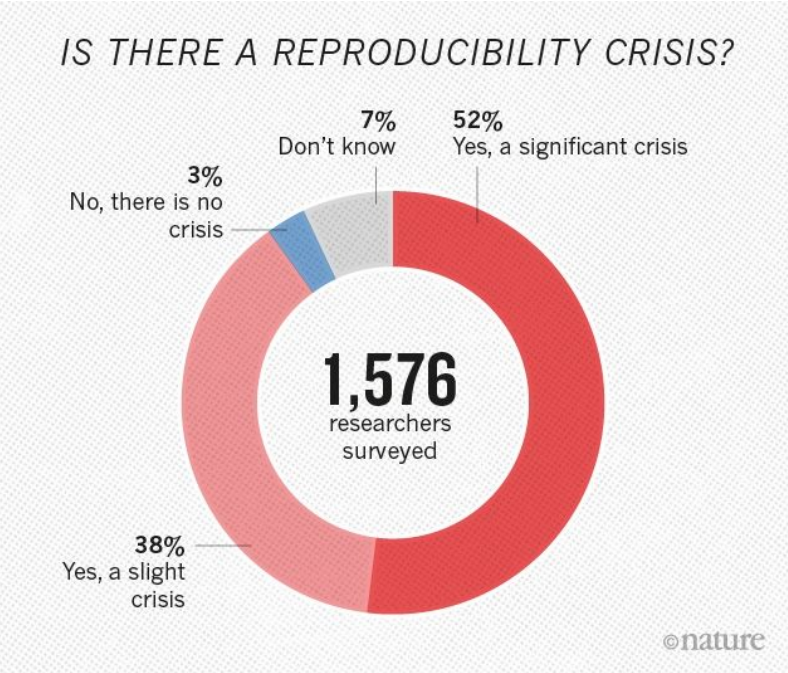
Your thoughts?

IS THERE A REPRODUCIBILITY CRISIS?

A crisis? *(Nature 2016)*

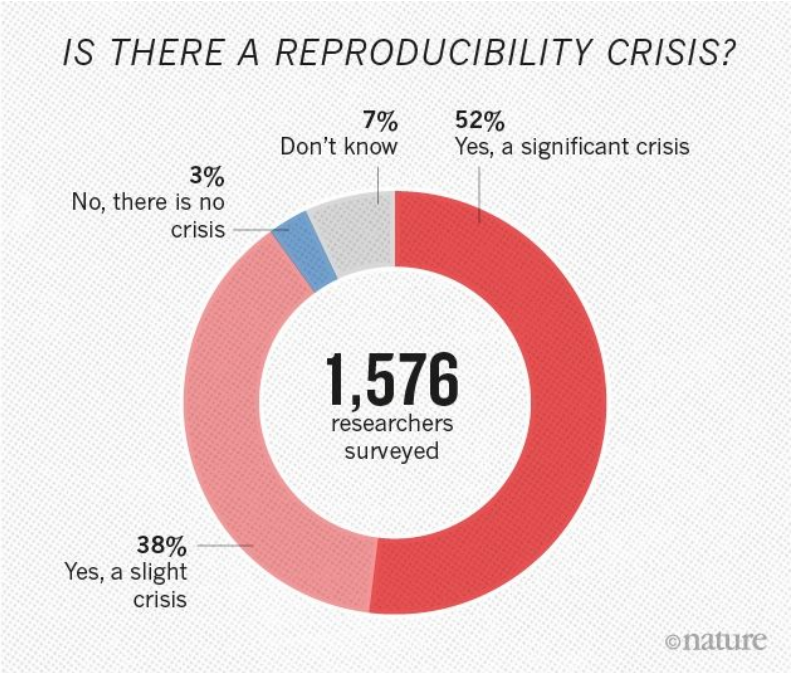


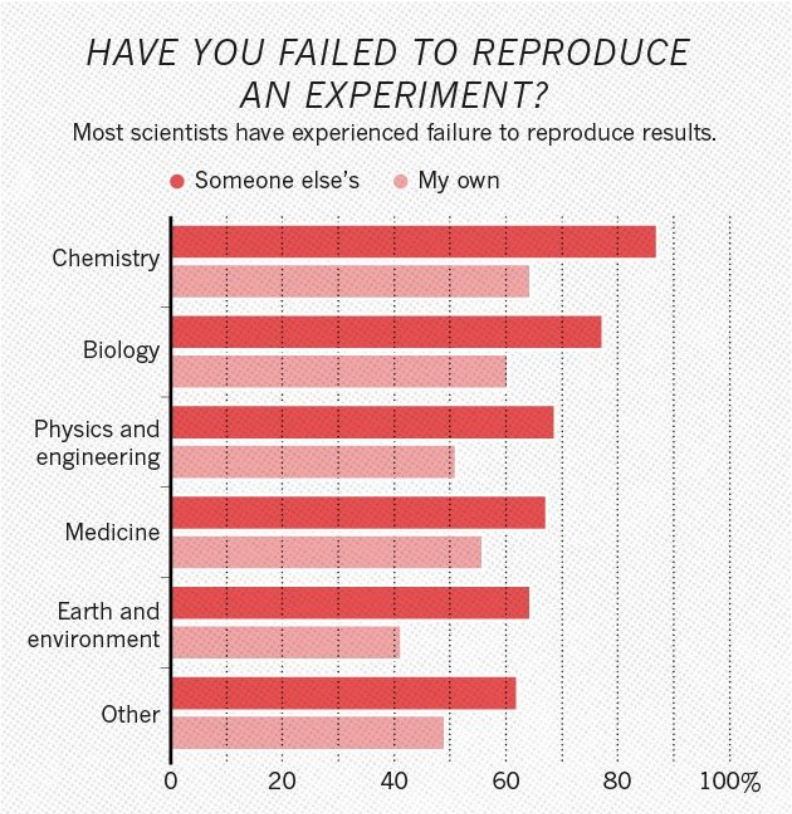
Your experience? *(Nature 2016)*



HAVE YOU FAILED TO REPRODUCE AN EXPERIMENT?

A common experience *(Nature 2016)*

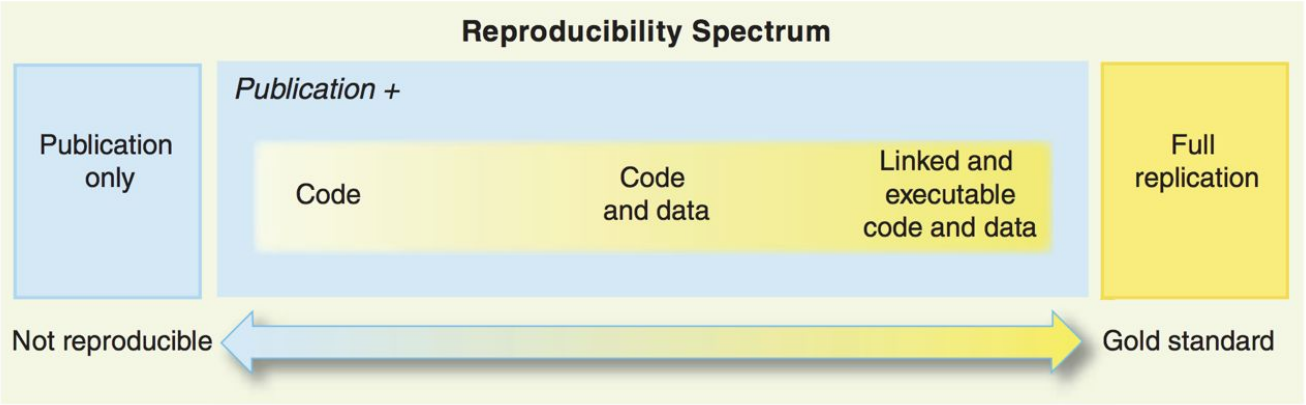




http://bit.ly/sips-workshop

An opportunity to help your future self

"It takes some effort to organize your research to be reproducible… the **principal beneficiary is generally the author herself**."- Schwab & Claerbout

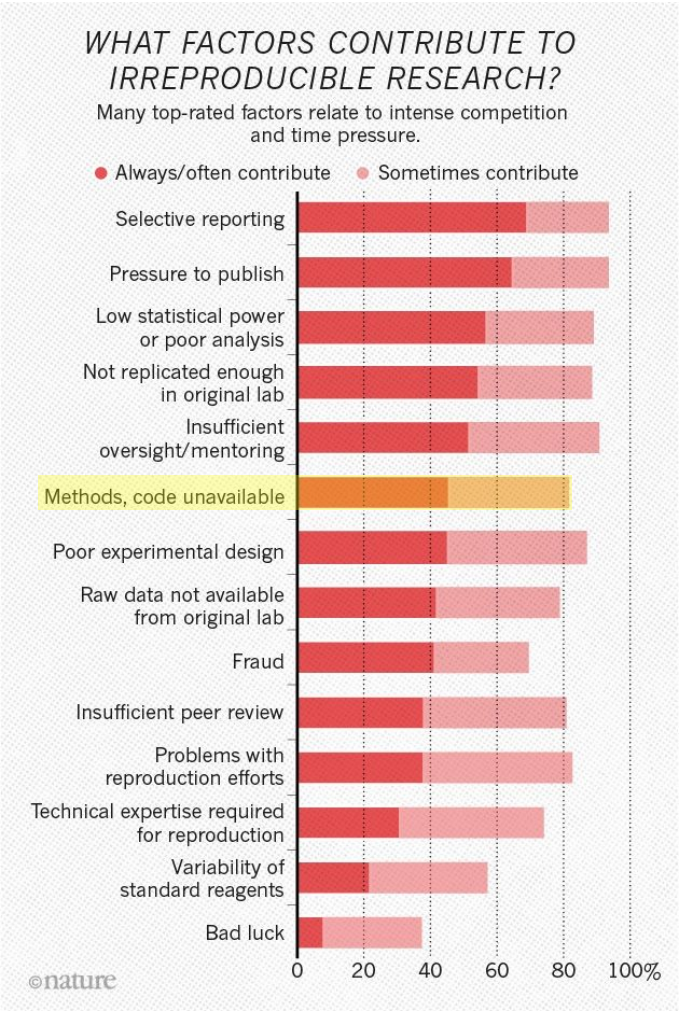


Peng, R.D. (2011) Science

Computational reproducibility

"An article about computational science in a scientific publication is not the scholarship itself, it is merely advertising of the scholarship. The actual scholarship is the **complete software development environment and the complete set of instructions** which generated the figure."

- Buckheit and Donoho (1995)'s distillation of Claerbout and Karrenbach (1992)



Nature 2016

http://bit.ly/sips-workshop

Communication during exercises:

1. Post a **pink sticky note** on your laptop at the start of the exercise.

2. Switch to a **green sticky note** when you finish and have no questions.

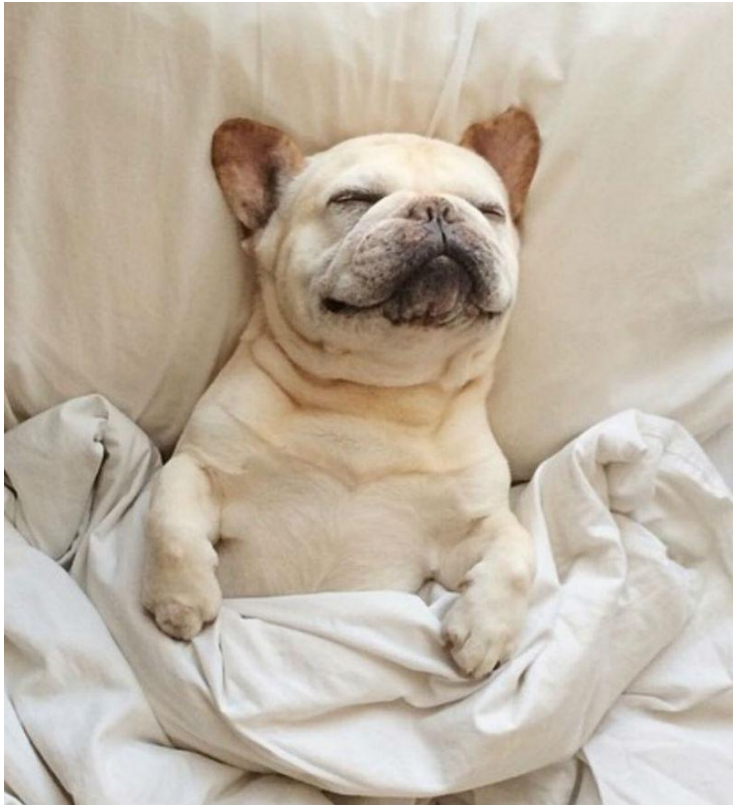
3. If you finish early, find someone with a **pink sticky note** and see if you can help!

4. If you are colorblind, the **pink sticky note** has a "**p**" written on it.

Exercise 1: Data Collection - Candy Trade

- **Pre-trade (Trade 0):** Review your selection of candy. Rate how happy you are with your selection on a scale from 1 (unhappy) to 10 (very happy).
  - In **this google form**, record your first name, your candy happiness rating, and select trade number "0".
- **Trade 1:** Find one trading partner. Trade the candy you don't like for candy you do like with that partner only. Rate how happy you are with your selection on a scale from 1 (unhappy) to 10 (very happy).
  - In **this google form**, record your first name, your candy happiness rating, and select trade number "1".
- **Trade 2:** Now trade with everyone in the room. Trade candy you don't like for candy you do like. Rate how happy you are with your selection on a scale from 1 (unhappy) to 10 (very happy).
  - In **this google form**, record your first name, your candy happiness rating, and select trade number "2".

10 MINUTE BREAK

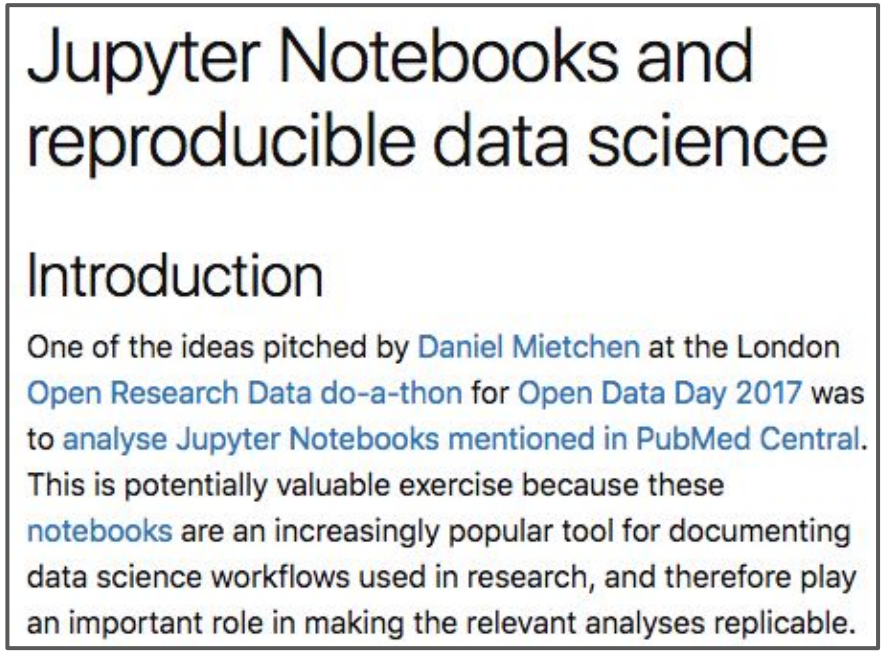


Tools we will use:

- OSF https://osf.io/
  - or Github https://github.com/
- Code Ocean https://codeocean.com/
- Binder (does not need account)

Lessons learned: testing computational reproducibility

- PMC “jupyter OR ipynb” -> 107 papers
- “My initial thought was that analysing the validity of the notebooks would simply involve searching the text of each article for a notebook reference, then downloading and executing it … **It turned out that this was hopelessly naive**…”



Jupyter Notebooks and reproducible data science

Introduction

One of the ideas pitched by Daniel Mietchen at the London Open Research Data do-a-thon for Open Data Day 2017 was to analyse Jupyter Notebooks mentioned in PubMed Central. This is potentially valuable exercise because these notebooks are an increasingly popular tool for documenting data science workflows used in research, and therefore play an important role in making the relevant analyses replicable.

Mark Woodbridge, Daniel Sanz, Daniel Mietchen, & Ross Mounce (2017). Jupyter Notebooks and reproducible data science, https://markwoodbridge.com/2017/03/05/jupyter-reproducible-science.html.

What *Woodbridge et al.* found:

- Files, data, dependencies needed to execute analyses **were often missing**.

We can **organize for reproducibility**:

- **Bundle dependencies** and include them in your repository rather than retrieve on demand.
- **Link to repositories**, not just files.
- **Archive the exact versions** of materials used and include them in your repository.

Exercise 2:

- **Create one repository that holds all related research files:**
  - Data
  - Code
  - Notebooks
  - Documentation
  - etc.

Join our Candy Swap OSF project

R: https://osf.io/yswhv/
Python: https://osf.io/jh8fc/

Or fork the github repo

R: https://github.com/aprilcs/sips-workshop
Python: https://github.com/aprilcs/sips-workshop-py

Exercise 3:

- **Organize your research to separate code from data.**


```
.
|-- CITATION
|-- README
|-- LICENSE
|-- requirements.txt
|-- data
|   -- birds_count_table.csv
|-- doc
|   -- notebook.md
|   -- manuscript.md
|   -- changelog.txt
|-- results
|   -- summarized_results.csv
|-- src
|   -- sightings_analysis.py
|   -- runall.py
```

Resource on reproducible organization:

- Karl Broman: http://kbroman.org/steps2rr/pages/organize.html

Checklist

```
/ [root]
├── code
│   ├── my_algorithm.py
│   ├── README.md
│   ├── run.sh
│   └── ...
├── data
│   ├── my_data.csv
│   ├── my_sample_image.png
│   └── ...
└── results
    └── [your future results]
```

- Create one repository or directory that holds all related research files.
- Organize your research to separate data, code, and results.
- Save results explicitly.
- Identify a strategy for sensitive data.

Tools

OSF

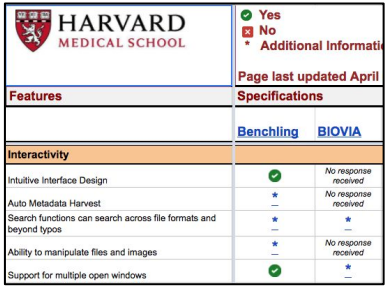
GitHub

CODE OCEAN BETA

SCINOTE

- Open Science Framework: collaborative project organization tool
- GitHub: collaborative coding, and project management
- eLNs: free or paid, lab organization
- Code Ocean: built in best practices

Resources



HARVARD MEDICAL SCHOOL

Yes
No
* Additional Informati

Page last updated April

Features	Specifications	
	Benchling	BIOVIA
Interactivity		
Intuitive Interface Design	✓	No response received
Auto Metadata Harvest	*	No response received
Search functions can search across file formats and beyond typos	*	*
Ability to manipulate files and images	*	No response received
Support for multiple open windows	✓	*

- Strategies for sensitive data sharing: Code Ocean Summary
- Harvard eLN Features Matrix: https://docs.google.com/spreadsheets/d/1ar8fqwaqOh30E31EAPL-Gorwn_q6XNf81q3VDQnQ_I8/edit?usp=sharing

What *Woodbridge et al.* found:

- There is no way to **directly express dependencies** of published code.

We can **publish using containers**:

- Use container technology to **directly express dependencies.**
- **Configure an image** for your analyses with Docker, binder, WholeTale, or Code Ocean.

The terms:

- **Dockerfile**: Readable instructions for how to build an image.
- **Image**: Everything your application needs to run, all bundled together (includes Dockerfile, libraries, and code).
- **Layer**: A Dockerfile directs Docker to build the initial image layer from a base image, and then other layers are built on top.
- **Container**: Started and created from an image.
- **Registry**: Images are stored and retrieved from registries.

Hale, Jeff. *Learn Enough Docker to be Useful*. https://towardsdatascience.com/learn-enough-docker-to-be-useful-b7ba70caeb4b

The metaphor: PIZZA!

- **Dockerfile**: The recipe.
- **Image**: The recipe and the ingredients combined as an all-in-one pizza-making-kit.
- **Layer**: The ingredients are the layers. You've got crust, sauce, and cheese for this pizza.
- **Container**: Cooked pizza. Cooked by Docker (the oven).
- **Registry**: All-in-one pizza-making-kit factories?

Hale, Jeff. *Learn Enough Docker to be Useful*. https://towardsdatascience.com/learn-enough-docker-to-be-useful-b7ba70caeb4b

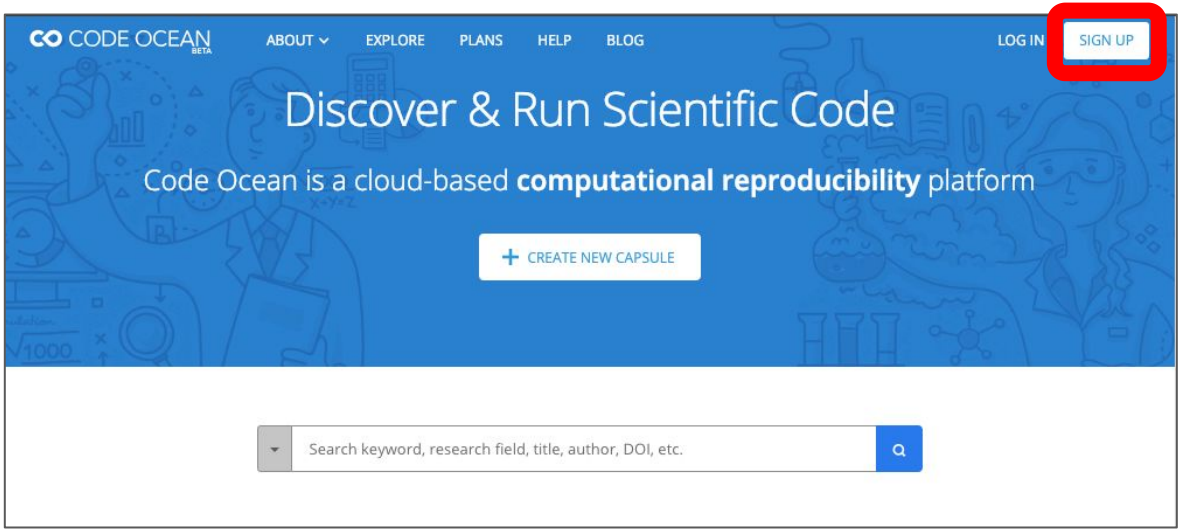
Containers solve:

- Dependency Hell - install, error, google, install, error...
  - Provides other researchers with a binary image in which all the software has already been installed, configured, and tested.
- Imprecise documentation - missing installation info.
  - Dockerfile provides a human readable summary of the necessary software dependencies needed to execute the code. Dependencies are automatically documented as they are installed.
- Code rot - dependencies change, the code breaks
  - Reduced risk with by archiving images

Boettiger, Carl. *An introduction to Docker for reproducible research.* 10.1145/2723872.2723882

Create a Code Ocean account

Pink sticky up!



- https://codeocean.com/
- You can **delete it and opt out of any communications** if you wish! For completing the exercises only. :)
- You will need to verify your email address

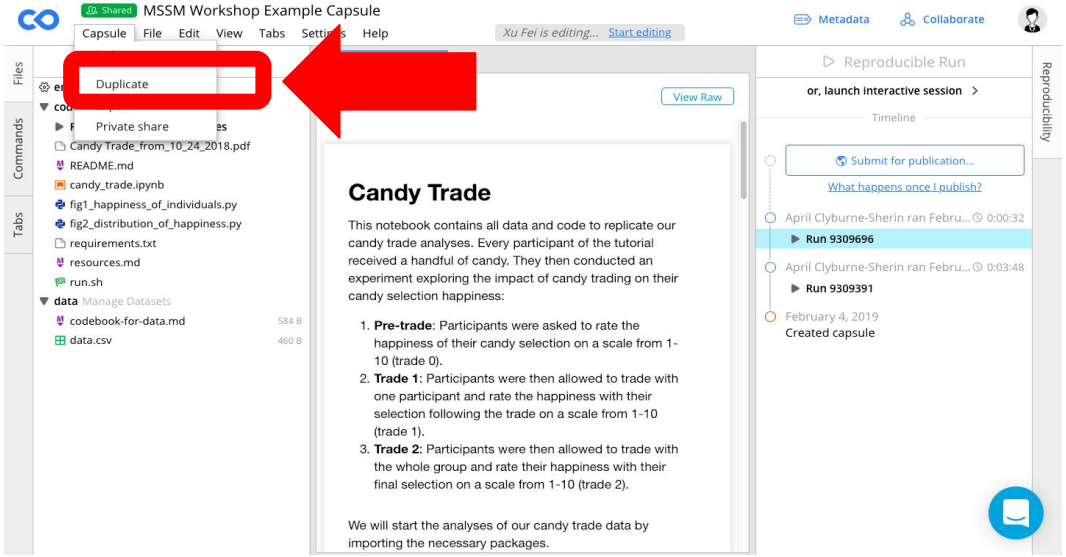
Duplicate this capsule:

R: http://bit.ly/r-example

Python: http://bit.ly/py-example

Pink sticky up!

- Click “Capsule”
- Select “Duplicate”



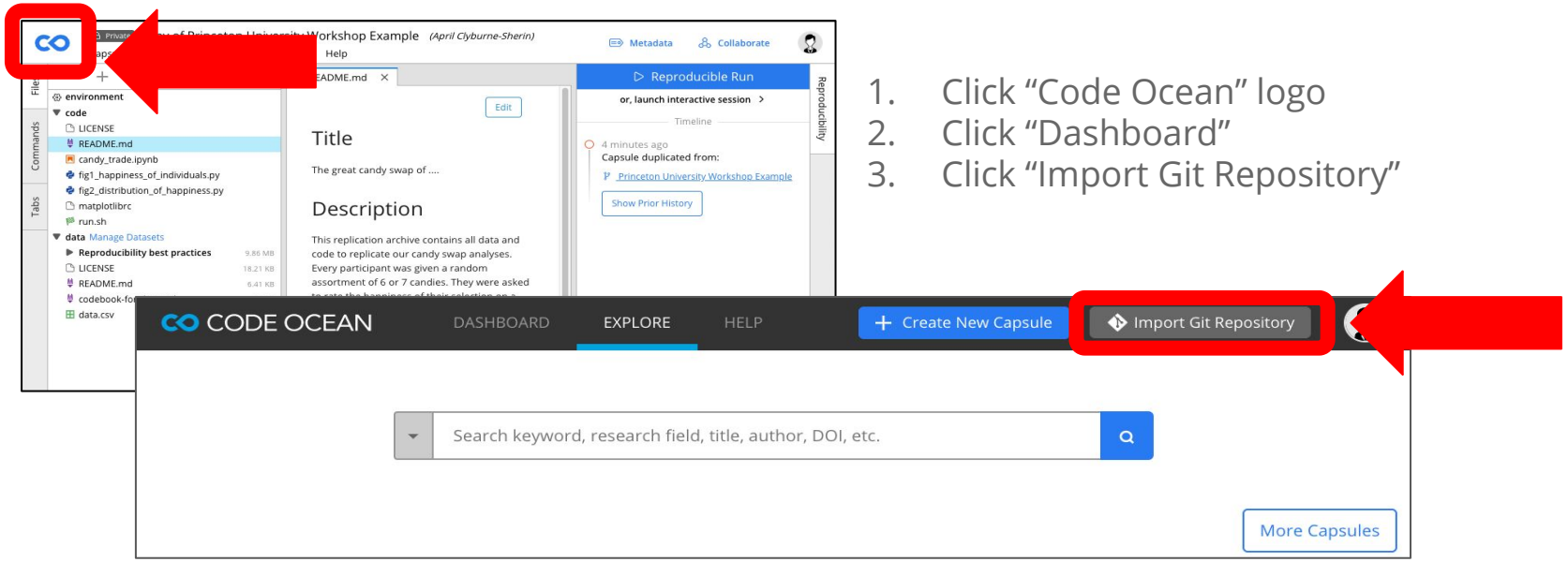
Create a new compute capsule

Import Git Repository:

R: https://github.com/aprilcs/sips-workshop

Python: https://github.com/aprilcs/sips-workshop-py

Pink sticky up!



1. Click “Code Ocean” logo
2. Click “Dashboard”
3. Click “Import Git Repository”

Pink sticky up!

Exercise 4:

- **Specify the run environment for your analyses.**

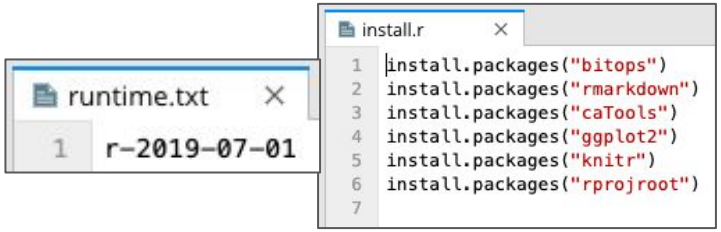
```
> sessionInfo()
R version 3.6.0 (2019-04-26)
Platform: x86_64-apple-darwin15.6.0 (64-bit)
Running under: macOS Mojave 10.14.5
```

Example: **Base Environment: R (3.5.3) or Python (3.7.0)**

Pink sticky up!

Exercise 5:

- **Specify your packages and dependencies with versions.**
  - Python: pip freeze > /requirements.txt
  - R: install.r and runtime.txt



R packages: **apt-get** pandoc; **CRAN** bitops, markdown, caTools, ggplot2, knitr, rprojroot

Python packages: **conda** matplotlib, pandas, numpy, jupyter

Resource on documenting dependencies:

- Binder: https://mybinder.readthedocs.io/en/latest/config_files.html

Exercise 6:

- **Use container technology to create an image of your complete computational environment.**
  - Code Ocean
  - Binder

Export your capsule to see how an image and Dockerfile were created through your specifications.

Inspect the Dockerfile.

We will demonstrate building a container with repo2docker using mybinder and github.

Demo:

- **Consider using literate programming to document the analysis narrative with the code.**
  - Jupyter Notebooks
  - RMarkdown

Explore Jupyter notebooks in this example capsule: http://bit.ly/uiuc-example

Explore RMarkdown in this example capsule: http://bit.ly/rmarkdown-example

- **Create a README file and data dictionary.**

Documenting your file overview and dependencies in your README:

- AJPS Replication Package: https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/EZSJ1S

Documenting your data in a codebook or data dictionary:

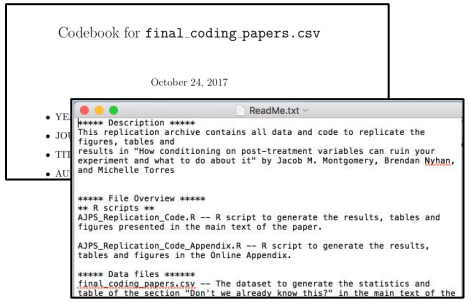
- DataONE: https://www.dataone.org/best-practices/create-data-dictionary

Resource on using markdown:

- GitHub: https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet

Reference Slide

Checklist



- Consider literate programming.
- Document each element or variable in your dataset with a data dictionary / codebook.
- Create a README file.

Tools

GitHub

jupyter

CODE OCEAN BETA

- Version control: git and GitHub tracks changes to documents and metadata
- Literate programming: knits documentation with code (Jupyter)
- Document & share metadata: Code Ocean renders documentation, notebooks, and records metadata

Resources

DCC because good research needs good data

Popular Licenses

The following OSI-approved licenses are popular, widely used,

- Apache License 2.0
- BSD 3-Clause "New" or "Revised" license
- BSD 2-Clause "Simplified" or "FreeBSD" license
- GNU General Public License (GPL)
- GNU Library or "Lesser" General Public License (LGPL)
- MIT license

- DataONE: https://www.dataone.org/best-practices/create-data-dictionary
- Cornell: https://data.research.cornell.edu/content/readme
- Digital Curation Center: http://www.dcc.ac.uk/resources/how-guides/license-research-data
- OSI: https://opensource.org/licenses

Reference Slide

Checklist

```
> sessionInfo()
R version 3.4.3 (2017-11-30)
Platform: x86_64-apple-darwin15.6.0 (64-bit)
Running under: macOS High Sierra 10.13.3

Matrix products: default
BLAS: /System/Library/Frameworks/Accelerate.framework/Versions/A/Frameworks/vecLib.framework/Version
LAPACK: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRlapack.dylib

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] multiwayvcov_1.2.3 lmtest_0.9-35      zoo_1.8-1          dummies_1.5.6      stargazer_5.2.1
[6] foreign_0.8-69

loaded via a namespace (and not attached):
 [1] Rcpp_0.12.16     lattice_0.20-35 grid_3.4.3      magrittr_1.5    pillar_1.2.1    rlang_0.2.0
 [7] boot_1.3-20      sandwich_2.4-0  forcats_0.3.0   tools_3.4.3     parallel_3.4.3  compiler_3.4.3
[13] haven_1.1.1      tibble_1.4.2
```

- Specify your computational environment and package versions.
- Configure a container to make your analysis portable and reusable.

Tools

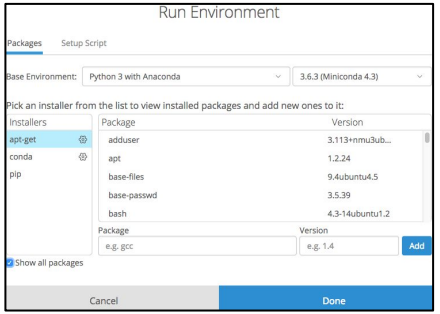
CODE OCEAN

docker

binder (beta)

- Container technology: packages data, code, metadata, & computational environment for portable analyses
- Docker: container technology for devs
- Code Ocean: easy configuring, preservation, & reuse of containers for researchers
- Binder: configure & share containers

Resources



- Documenting dependencies: http://mybinder.readthedocs.io/en/latest/using.html#preparing-a-repository-for-binder
- Specifying environments: https://help.codeocean.com/getting-started/the-computational-environment/selecting-a-base-environment

What *Woodbridge et al.* found:

- **Manual manipulation or setup** was needed to reproduce results, often without documentation of how the results were produced.

We can **automate the execution of our analyses**:

- Create a master script to execute all analyses.
- Reproduce results automatically as a function of the data & the code; Save results explicitly.
- Use relative paths.

Exercise 7:

- **Create a master script to execute your code.**

- In R, use a run.r or main.r master script
  - Use source() to run your scripts
  - Run your install.r script
- In Python, use a main.py or run.sh master script
  - In your run.sh script, use nbconvert to execute your notebook into the results directory.
- Case study: https://www.practicereproducibleresearch.org/core-chapters/3-basic.ht

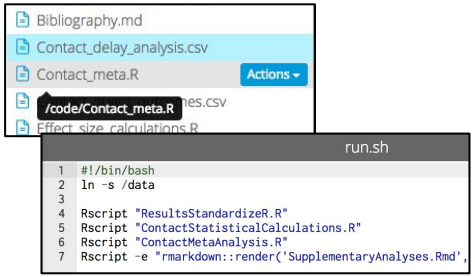
Exercise 8:

- **Change absolute paths to relative paths.**

Resource explaining paths:

- Karl Broman: http://kbroman.org/steps2rr/pages/organize.html

Checklist



- Use relative rather than absolute paths.
- Create a master script that runs your scripts in sequence.

Tools

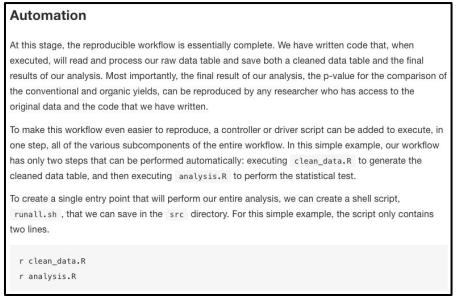
CODE OCEAN

docker

binder (beta)

- Docker: share automated code for devs
- Code Ocean: easy configuring, preservation, & reuse of automated code
- Binder: share automated code for using containers

Resources



- Karl Broman on paths: http://kbroman.org/steps2rr/pages/organize.html
- Resource on automation using a master script: https://www.practicereproducibleresearch.org/core-chapters/3-basic.html

What *Woodbridge et al.* found:

- There is no standardized way of **attaching code to published articles**.
- Therefore it is difficult to **discover and retrieve** code.

We can **embed or link code persistently**:

- **Obtain a DOI for your repository and use this link throughout your article.**
  - Example: Github -> Binder/WholeTale -> Zenodo -> DOI linked in article
  - Example: CodeOcean -> DOI in article
- **Cross link repository with published article in metadata of each.**
- **Embed executable capsule within the article.**
  - Example: https://doi.org/10.1017/bpp.2018.25

Exercise 9:

- **Specify a license for your data and your code.**

Resource on choosing a data licence:

Digital Curation Center: http://www.dcc.ac.uk/resources/how-guides/license-research-data

Resources on choosing a code licence:

- Karl Broman: http://kbroman.org/steps2rr/pages/licenses.html
- License picker: https://choosealicense.com/
- Open Source Initiative: https://opensource.org/licenses

Exercise 10:

- **Share your code!**

- Check whether your container is ready to publish by hitting "Run".

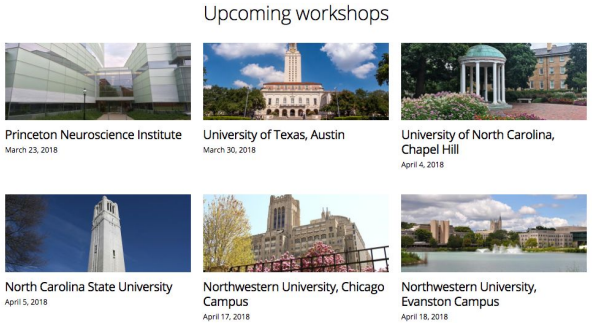
Reproducibility support

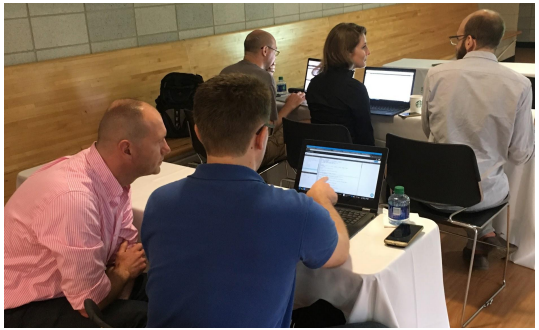
Workshops & Webinars

- Theory or hands-on
- Customized to researcher needs
- Request a workshop or webinar at https://codeocean.com/events

1:1 Computational Reproducibility Consult

- In person
  - Lab meeting
  - Office visit
- Virtual
- Request at april@codeocean.com or https://doodle.com/codeocean





http://bit.ly/sips-workshop

CODE OCEAN

Reproducibility community

Reproducibility Ambassador Program

- **Scholarships** to present your research at conferences
- **Support** for lab events, journal clubs, meetups
- **Training**, mentorship, and community forum
- **Opportunities** to share your perspective on reproducibility
- **Co-development** role to help us meet your needs and try out new features

Preprint journal club

- Build peer review **skills** including code review
- **Contribute** feedback to new research

Thank you for your time :)

Please fill out an evaluation so we can keep improving!

http://bit.ly/workshop-survey-2019

April Clyburne-Sherin
Code Ocean

april@codeocean.com