Creating Reproducible and Interactive Analyses with JupyterLab and Binder

Elena Auer
Richard Landers
University of Minnesota, Twin Cities

SIOP 2019

UNIVERSITY OF MINNESOTA
Driven to Discover®

Today's Tutorial

Purpose: create **interactive**, **literate** code documents, enabling others to replicate analyses with one click on the web

Today's Tutorial

- Intended audience:
  - Anyone using R or Python
  - Anyone interested in improving the reproducibility of their analyses
  - Anyone interested in sharing their work

Today's topics

Conceptual (~25 min)

- Overview of Reproducibility
- Intro to JupyterLab & Jupyter Notebook
- Intro to Binder & Docker Containers
- Some advance techniques (if there's time)

Demo (~50 min)

- Intro to JupyterLab, Jupyter Notebooks, and Binder
- Converting R script to an interactive, literate code document

Today's Topics

What we won't cover:

- R or Python
- Advanced uses of JupyterLab, Binder & Docker
- All possible tools for computational reproducibility (there are a lot!)

Following along

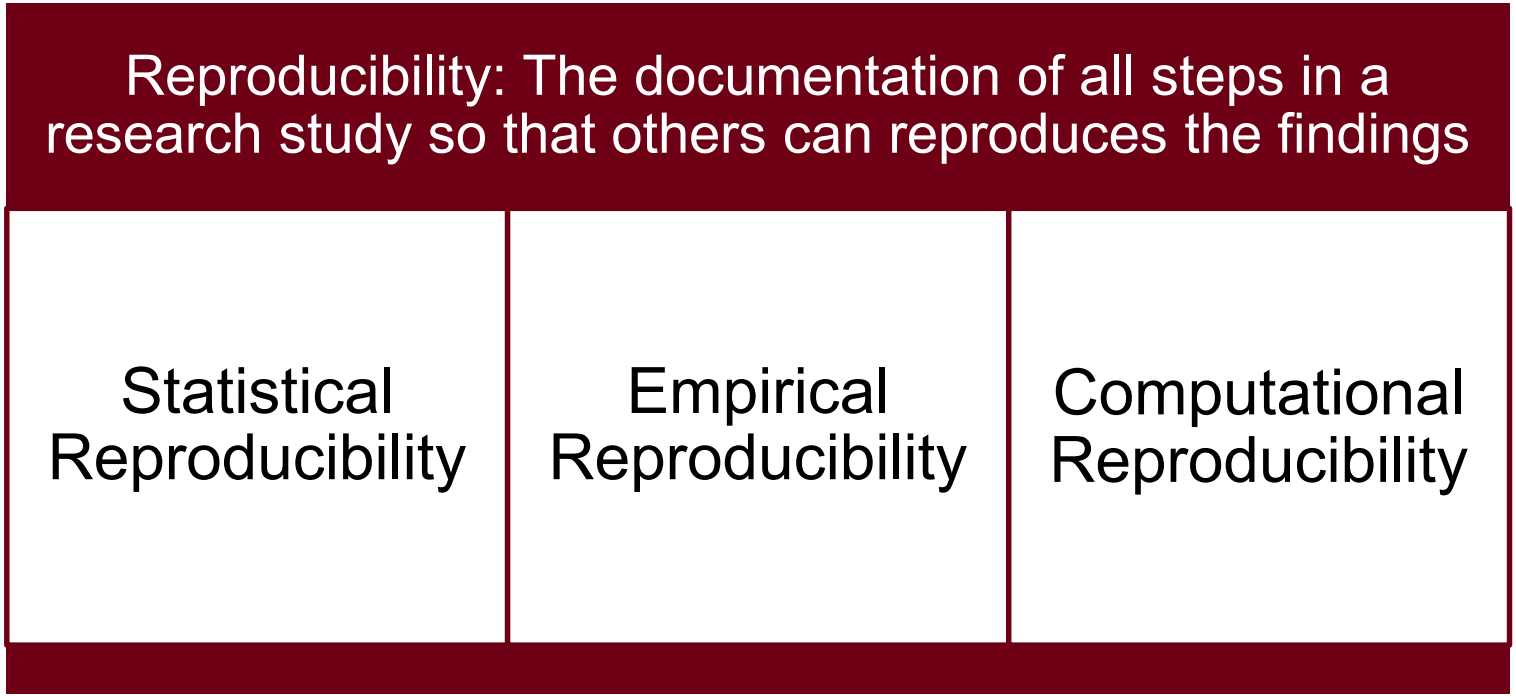
Open up our Jupyter Notebooks using Binder

http://bit.ly/SIOPJupyterBinderTutorial

- Click on one of the binder badges to follow along with demonstrations

Reproducible Research

Reproducible Research



(Fomel & Claerbout, 2009; Stodden et al., 2014)

The puzzle of deriving insight from data

- A LOT of decisions can go into cleaning and analyzing data
  - Removing incomplete/test cases
  - Merging/restructuring datasets
  - Creating composite variables
  - Removing outliers
  - Specifying interactions

- Bigger data & more complex modeling approaches exponentially increases these decisions

Reproducible Puzzles (Analyses)

Reproducibility exists to varying degrees

Written description of puzzle

Box of puzzle pieces

Puzzle put together in a puzzle mat

Written description of data analyses

Code and data sharing

Interactive & Literate Code

Low Reproducibility

High Reproducibility

Barriers to Computational Reproducibility

1. Dependency hell –installation and versioning issues
2. Imprecise documentation – can be overly complex, incorrect, or not up-to-date
3. Code rot – dependencies are updated, changing results
4. Barriers to adoption and reuse – knowledge and time required to put together the pieces

Boettiger (2015)

Reproducibility Matters

- Provides evidence for the ***correctness of findings*** & limits questionable research practice opportunity
- Preserves all steps *for* ***future use and extension*** by others (and yourself!)

Jupyter Notebooks & JupyterLab

Designing and Creating Jupyter Notebooks using JupyterLab

Project Jupyter: organization focused on developing open-source software and services for interactive computing for multiple programming languages

- **Jupyter Notebook (previously IPython notebook):** open-source web application for creating and sharing documents with code, visualizations, and text
- **JupyterLab:** “next generation” web-based user interface for working with Jupyter Notebooks (will eventually replace Jupyter Notebook)

Jupyter Notebooks

Rich Text

This is a Jupyter Notebook

[72]:
```
bfi_data <- cbind(bfi_data,as.data.frame(scores$scores))
head(bfi_data)
```

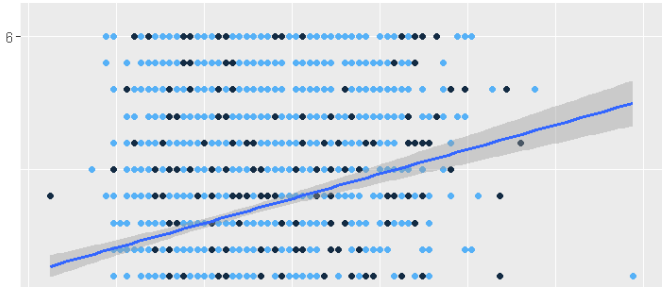
	A1	A2	A3	A4	A5	C1	C2	C3	C4	C5	...	O4	O5	gender	education	age	agree	conscientious	extraversion	neuroticism	openness
61617	2	4	3	4	4	2	3	3	4	4	...	4	3	1	NA	16	4.0	2.8	3.8	2.8	3.0
61618	2	4	5	2	5	5	4	4	3	4	...	3	3	2	NA	18	4.2	4.0	5.0	3.8	4.0
61620	5	4	5	4	4	4	5	4	2	5	...	5	2	2	NA	17	3.8	4.0	4.2	3.6	4.8
61621	4	4	6	5	5	4	4	3	5	5	...	3	5	2	NA	17	4.6	3.0	3.6	2.8	3.2
61622	2	3	3	4	5	4	4	5	3	2	...	3	3	1	NA	17	4.0	4.4	4.8	3.2	3.6
61623	6	6	5	6	5	6	6	6	1	3	...	6	1	2	3	21	4.6	5.6	5.6	3.0	5.0

We can start to visualize some relationships amongst the variables

Code

[74]:
```
ggplot(aes(x=age, y=agree),data=bfi_data) +
geom_point(aes(color=gender))+  # scatter plot
 geom_smooth(method="lm")
```

Output



Designing and Creating Jupyter Notebooks using JupyterLab

Some key terms & jargon:

Term	Description
Kernels	Computational engine that executes code (i.e., Python, R)
Cells	Markdown, Code, Raw
Markdown	markup language with plain text formatting syntax
Extensions	Adding functionality to jupyterlab environment (Github, table of contents, inspecting variables)
Filename Extension	.ipynb

Designing and Creating Jupyter Notebooks using JupyterLab

Downloading and accessing Jupyterlab

Recommended: Anaconda

https://www.anaconda.com/distribution/

- GUI (select *64-Bit Graphical Installer*)
- Command line (select *64-Bit Command Line Installer*)
- Python 2 or 3-recommended
- Available for Mac, Windows, or Linux

Designing and Creating Jupyter Notebooks using JupyterLab

- Jupyter Notebooks enable **literate** code documents that:
  - Tell a comprehensive story in plain words
  - Include code chunks for explicit documentation of steps and analyses
  - Render output and visualizations for lasting documentation
- But is this enough for reproducibility?

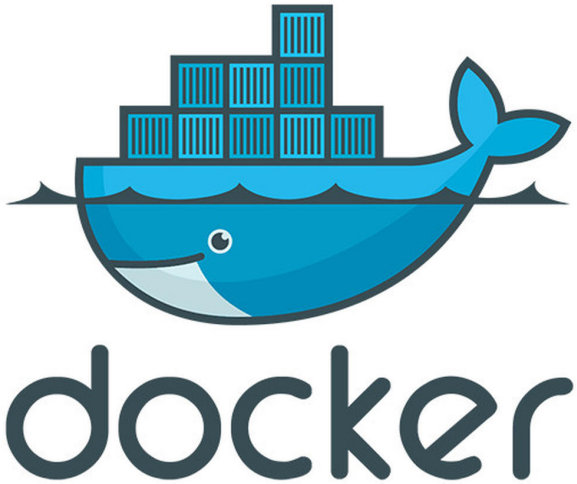

binder

Designing and Creating a Binder

- Enables **Interactivity**
- Binder primarily does three things:
  1. Captures a code repository and its technical environment using a Docker container
  2. Generates a user session using that technical environment using BinderHub
  3. Provides a link for users to share and interact with the environment using mybinder.org

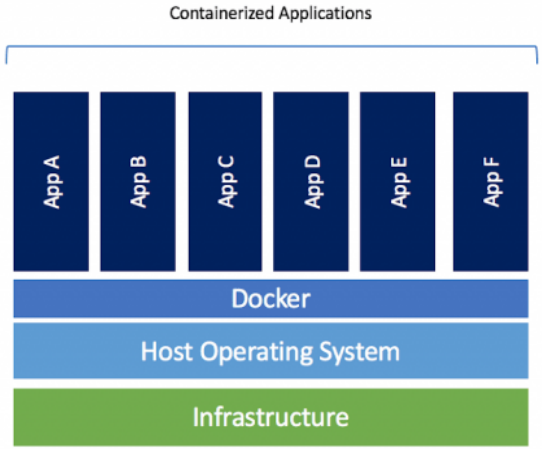
What is a Docker Container?

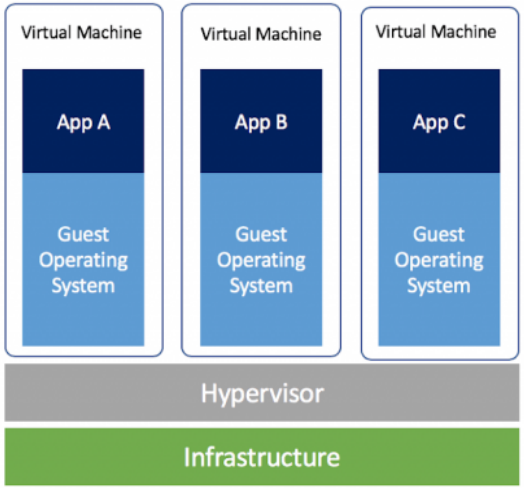
- Remember that puzzle mat?
- Packages software into standardized units for development, shipment, and deployment
  - Lightweight
  - Standalone (Can be run on any OS)
  - Executable
  - Isolated
- Prevents the “works on my machine” issue



What is a Docker Container?

Similar to a virtual machine but containers virtualize the operating system instead of the hardware – increases portability and efficiency





What is repo2docker?

1. Fetches a git repository (e.g., one found on github)
2. Builds a docker container image based on the configuration files found in the repository
3. Docker container can then be run via a Jupyter server (JupyterHub)

Documentation:
https://repo2docker.readthedocs.io/en/latest/

What is JupyterHub?

A Multi-user Hub created to spawn, manage, and proxy multiple instances of the single-user Jupyter notebook server

- Gives users access to computational environments and resources
- Useful for sharing notebooks with collaborators or class

Documentation:
https://jupyterhub.readthedocs.io/en/stable/

What is BinderHub?

- Integrates JupyterHub and repo2docker
  - Creates and launches a custom computing environment for multiple users at a URL address
  - Can be deployed on most cloud providers
- Custom BinderHub enables more computational resources for interested users

Documentation:
https://binderhub.readthedocs.io/en/latest/

What is mybinder.org?

A free, public deployment of BinderHub

- Run by Project Jupyter & Binder team
- Grant funded (Moore Foundation and Google Cloud Platform)
- Public & Free to use but limited computational resources
  - 1-2 GB of RAM
  - Ephemeral (shuts down after 10 min of inactivity)

Demo

Demo: Using JupyterLab & Binder

- Want to follow along?
  http://bit.ly/SIOPJupyterBinderTutorial
- Key topics covered:
  - Creating a Jupyter Notebook in JupyterLab
  - Building & sharing a Binder
  - Full workflow: converting plain R script into a fully interactive & reproducible document

Reproducible Research

The final product:

Interactive & **literate** code documents

JupyterLab: Advanced Topics

- Widgets
  - Building interactive documents (slider, text box, interacting with dataframes)
- Magic Commands
  - Shortcuts (passing variables between notebooks, listing variables in environment, using multiple kernels)
- Hosting notebooks on server with multiple users (Jupyterhub)
- Jupyter Notebooks in HPC or cloud environment

Binder: Advanced Topics

- Range of simplicity for creating docker containers
  - mybinder.org
  - BinderHub on your server (https://binderhub.readthedocs.io/en/latest/#)
  - Docker container (https://www.docker.com/get-started)
- Choosing an approach
  - Benefits and drawbacks (simplicity, privacy, data storage)

Other Use Cases

- Classes and Tutorials (like today's)
- Collaboration
- Dashboards
- Scalable computing: moving code to an HPC/Cloud computing environment

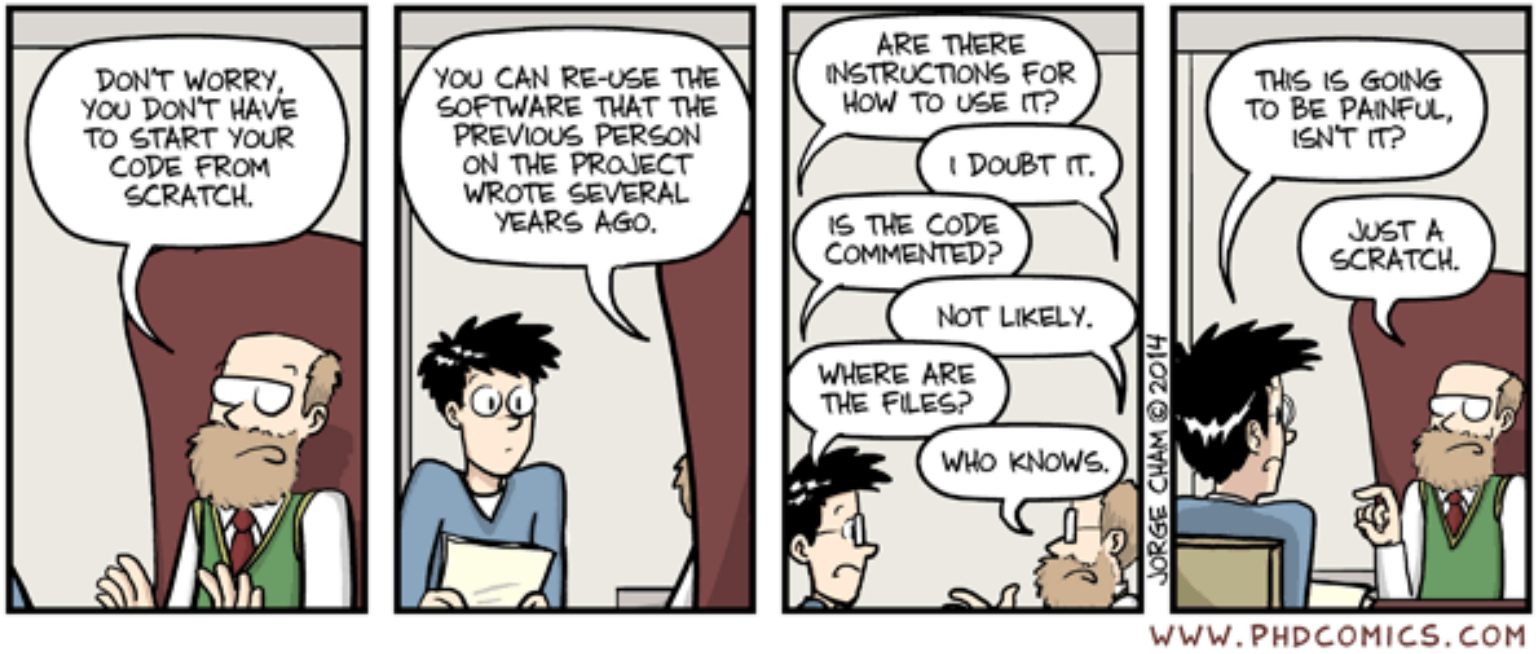
Wrap-up

- IOs are interested in computationally intensive analyses (big data, machine learning, etc.)
  - Learning data science techniques for dealing with the complexity of this code is a natural next step

- Some domain-science fields have started to integrate these practices (e.g., genomics and bioinformatics)
  - IOs have the potential to become leaders in computationally intensive, reproducible science

Resources

- All of today's materials can be found on GitHub: **http://bit.ly/SIOPJupyterBinderTutorial**
- Project Jupyter has extensive documentation for all of their software: https://jupyter.org/
  - *Note: a lot of documentation assumes some knowledge/use of command line (substitute with google)*
- Advance use cases for both Jupyter Notebooks and Binder/Docker will require some command line knowledge
  - Software Carpentry has free & helpful tutorials: http://swcarpentry.github.io/shell-novice/



Contact information:

Elena Auer (auer0027@umn.edu)

References:

Boettiger, C. (2015). An introduction to Docker for reproducible research. *ACM SIGOPS Operating Systems Review*, *49*(1), 71-79

Fomel, S., & Claerbout, J. F. (2009). Reproducible research. *Computing in Science & Engineering*, *11*(1), 5.

Jupyter et al., "Binder 2.0 - Reproducible, Interactive, Sharable Environments for Science at Scale." Proceedings of the 17th Python in Science Conference. 2018. 10.25080/Majora-4af1f417-011

Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B. E., Bussonnier, M., Frederic, J., ... & Ivanov, P. (2016, May). Jupyter Notebooks-a publishing format for reproducible computational workflows. In *ELPUB* (pp. 87-90).

Stodden, V., Leisch, F., & Peng, R. D. (Eds.). (2014). *Implementing reproducible research*. CRC Press.