A conversation on tools for more reproducible and interactive data analysis

with Jupyter, R Markdown, and Binder demos

Dana Miller

Aims

- Discuss how researchers and collaborators in our field can take advantage of increasingly common tools to supplement and share their research and teaching materials
- Hands-on demos
- Get feedback and iterate

→ What tools or problems are *you* are most interested in?

Overview

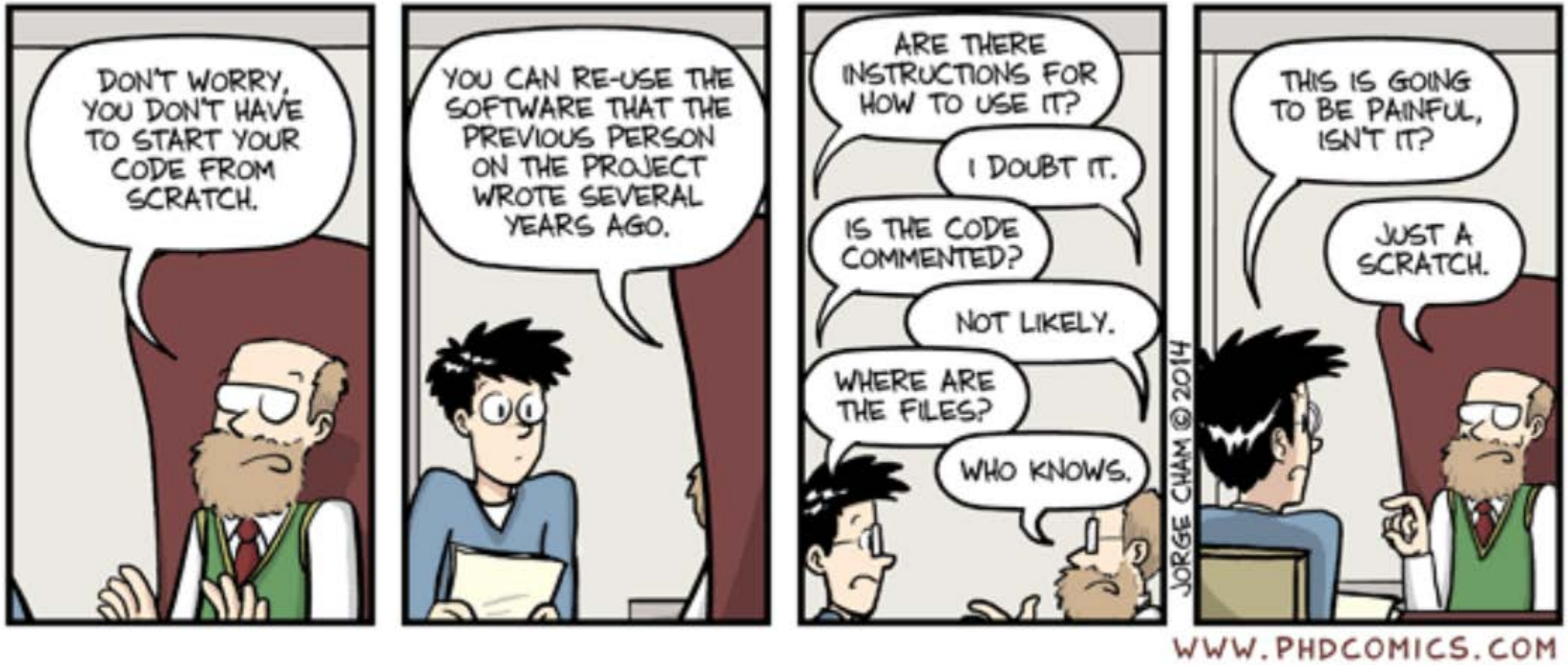
- What is reproducible research?
  - Motivating examples
  - Definitions
  - State of practice
- Literate programming tools
  - R Notebooks
    - History, key features
    - Demo
  - Jupyter Notebooks
    - History, key features
    - Demo
- Towards interactive + portable code
  - MyBinder + Docker
    - Demo
  - Cloud computer resources
- What next?

Not explicitly covered

- Version control+ collaboration (eg with git + GitHub)
  - Separate workshop
- Latest version of every tool (rapidly evolving ecosystem)
  - JupyterLab

Why reproducible research?

Motivating example 1 - productivity



http://phdcomics.com/comics/archive.php?comicid=1689

Motivating example 2 - posterity

Email from grad student to a professor in 2017:

> "…The citation on the slide is [your final project report on a state government website]. I downloaded that PDF and do not see [important statistic] anywhere in it. Table 3 seems to have [different value] , Table 15 has [related value] in California…
>
> …The closest match that I've found is [other statistic] from Table 15... Is it possible there was an error in transcription? If not, can you help me figure out what's going on here? What does [important statistic] on [slide] represent?"

Response from professor:

> "I don't remember the details. [Former student], do you? Thanks in advance."
>
> [Former student] never responded

Motivating example 3 – interactive publications!

New paper
(in this case, a preprint)

+

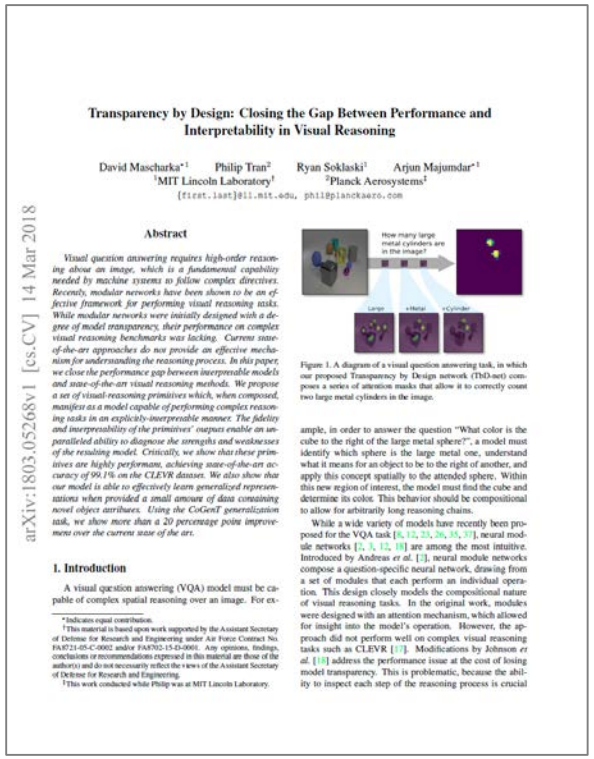
Website explaining paper

(has code to replicate experiments + plots and explicit software versions)

+

Interactive Jupyter notebook with code to apply their method to your own images

(with all required files and software ready to run in your browser with no installation required, thanks to Binder)



Transparency by Design: Closing the Gap Between Performance and Interpretability in Visual Reasoning

David Mascharka*[1] Philip Tran[2] Ryan Soklaski[1] Arjun Majumdar*[1]
[1]MIT Lincoln Laboratory[1] [2]Planck Aerosystems[2]
{first.last}@ll.mit.edu, phil@planckaero.com

Abstract

Visual question answering requires high-order reasoning about an image, which is a fundamental capability needed by machine systems to follow complex directives. Recently, modular networks have been shown to be an effective framework for performing visual reasoning tasks. While modular networks were initially designed with a degree of model transparency, their performance on complex visual reasoning benchmarks was lacking. Current state-of-the-art approaches do not provide an effective mechanism for understanding the reasoning process. In this paper, we close the performance gap between interpretable models and state-of-the-art visual reasoning methods. We propose a set of visual-reasoning primitives which, when composed, manifest as a model capable of performing complex reasoning tasks in an explicitly-interpretable manner. The fidelity and interpretability of the primitives' outputs enable an unparalleled ability to diagnose the strengths and weaknesses of the resulting model. Critically, we show that these primitives are highly performant, achieving state-of-the-art accuracy of 99.1% on the CLEVR dataset. We also show that our model is able to effectively learn generalized representations when provided a small amount of data containing novel object attributes. Using the CoGenT generalization task, we show more than a 20 percentage point improvement over the current state of the art.

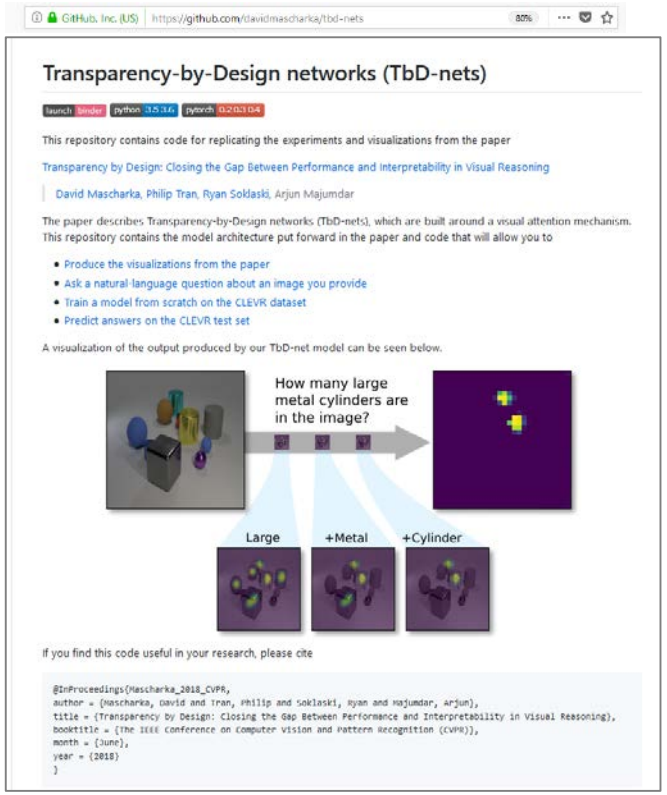
1. Introduction

A visual question answering (VQA) model must be capable of complex spatial reasoning over an image. For example, in order to answer the question "What color is the cube to the right of the large metal sphere?", a model must identify which sphere is the large metal one, understand what it means for an object to be to the right of another, and apply this concept spatially to the attended sphere. Within this new region of interest, the model must find the cube and determine its color. This behavior should be compositional to allow for arbitrarily long reasoning chains.

While a wide variety of models have recently been proposed for the VQA task [8, 12, 23, 26, 35, 37], neural module networks [2, 3, 12, 18] are among the most intuitive. Introduced by Andreas *et al.* [2], neural module networks compose a question-specific neural network, drawing from a set of modules that each perform an individual operation. This design closely models the compositional nature of visual reasoning tasks. In the original work, modules were designed with an attention mechanism, which allowed for insight into the model's operation. However, the approach did not perform well on complex visual reasoning tasks such as CLEVR [17]. Modifications by Johnson *et al.* [18] address the performance issue at the cost of losing model transparency. This is problematic, because the ability to inspect each step of the reasoning process is crucial

*Indicates equal contribution.
This material is based upon work supported by the Assistant Secretary of Defense for Research and Engineering under Air Force Contract No. FA8721-05-C-0002 and/or FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Assistant Secretary of Defense for Research and Engineering.
This work conducted while Philip was at MIT Lincoln Laboratory.

https://arxiv.org/abs/1803.05268



GitHub, Inc. (US) https://github.com/davidmascharka/tbd-nets

Transparency-by-Design networks (TbD-nets)

This repository contains code for replicating the experiments and visualizations from the paper

Transparency by Design: Closing the Gap Between Performance and Interpretability in Visual Reasoning

David Mascharka, Philip Tran, Ryan Soklaski, Arjun Majumdar

The paper describes Transparency-by-Design networks (TbD-nets), which are built around a visual attention mechanism. This repository contains the model architecture put forward in the paper and code that will allow you to

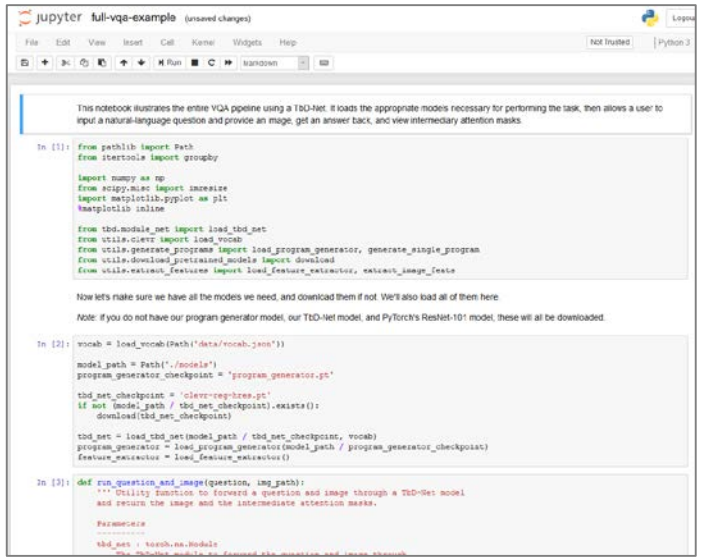
- Produce the visualizations from the paper
- Ask a natural-language question about an image you provide
- Train a model from scratch on the CLEVR dataset
- Predict answers on the CLEVR test set

A visualization of the output produced by our TbD-net model can be seen below.

If you find this code useful in your research, please cite

```
@InProceedings{Mascharka_2018_CVPR,
author = {Mascharka, David and Tran, Philip and Soklaski, Ryan and Majumdar, Arjun},
title = {Transparency by Design: Closing the Gap Between Performance and Interpretability in Visual Reasoning},
booktitle = {The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)},
month = {June},
year = {2018}
}
```

https://github.com/davidmascharka/tbd-nets



jupyter full-vqa-example (unsaved changes)

This notebook illustrates the entire VQA pipeline using a TbD-Net. It loads the appropriate models necessary for performing the task, then allows a user to input a natural-language question and provide an image, get an answer back, and view intermediary attention masks.

```
In [1]: from pathlib import Path
        from itertools import groupby

        import numpy as np
        from scipy.misc import imresize
        import matplotlib.pyplot as plt
        %matplotlib inline

        from tbd.module_net import load_tbd_net
        from utils.clevr import load_vocab
        from utils.generate_programs import load_program_generator, generate_single_program
        from utils.download_pretrained_models import download
        from utils.extract_features import load_feature_extractor, extract_image_featu
```

Now let's make sure we have all the models we need, and download them if not. We'll also load all of them here.

Note: if you do not have our program generator model, our TbD-Net model, and PyTorch's ResNet-101 model, these will all be downloaded.

```
In [2]: vocab = load_vocab(Path('data/vocab.json'))

        model_path = Path('./models')
        program_generator_checkpoint = 'program_generator.pt'

        tbd_net_checkpoint = 'clevr-reg-hres.pt'
        if not (model_path / tbd_net_checkpoint).exists():
            download(tbd_net_checkpoint)

        tbd_net = load_tbd_net(model_path / tbd_net_checkpoint, vocab)
        program_generator = load_program_generator(model_path / program_generator_checkpoint)
        feature_extractor = load_feature_extractor()
```

```
In [3]: def run_question_and_image(question, img_path):
            ''' Utility function to forward a question and image through a TbD-Net model
            and return the image and the intermediate attention masks.

            Parameters
            ----------
            tbd_net : torch.nn.Module
```

https://mybinder.org/v2/gh/davidmascharka/tbd-nets/binder?filepath=full-vqa-example.ipynb

Additional reference: Link to tweet by first author publicizing this work

Demo – reproduce figures from paper on the previous slide

1. See the paper: *https://arxiv.org/abs/1803.05268*
   - *Note the PDF it includes link to a GitHub repo*

2. Inspect the code
- *Open the associated GitHub repository containing the Jupyter notebook used to produce the figures for the paper*
  - *Repo:* https://github.com/davidmascharka/tbd-nets
  - *Notebook: https://github.com/davidmascharka/tbd-nets/blob/master/visualize-output.ipynb*

3. Reproduce the visualizations without installing anything!
   - *Click the* launch binder *button on the main page of the repo*

Interactive publications – in R too!

New paper
(open access)

ECOLOGY LETTERS

Ecology Letters, (2018) doi: 10.1111/ele.13085

REVIEW AND SYNTHESIS

From noise to knowledge: how randomness generates novel phenomena and reveals information

Carl Boettiger*

Dept of Environmental Science, Policy, and Management, University of California Berkeley, Berkeley, CA 94720-3114, USA

*Correspondence: E-mail: cboettig@berkeley.edu

Abstract

Noise, as the term itself suggests, is most often seen a nuisance to ecological insight, a inconvenient reality that must be acknowledged, a haystack that must be stripped away to reveal the processes of interest underneath. Yet despite this well-earned reputation, noise is often interesting in its own right: noise can induce novel phenomena that could not be understood from some underlying deterministic model alone. Nor is all noise the same, and close examination of differences in frequency, colour or magnitude can reveal insights that would otherwise be inaccessible. Yet with each aspect of stochasticity leading to some new or unexpected behaviour, the time is right to move beyond the familiar refrain of "everything is important" (Bjørnstad & Grenfell 2001). Stochastic phenomena can suggest new ways of inferring process from pattern, and thus spark more dialog between theory and empirical perspectives that best advances the field as a whole. I highlight a few compelling examples, while observing that the study of stochastic phenomena are only beginning to make this translation into empirical inference. There are rich opportunities at this interface in the years ahead.

Keywords

Coloured noise, demographic noise, environmental noise, quasi-cycles, stochasticity, tipping points.

Ecology Letters (2018)

INTRODUCTION: NOISE THE NUISANCE

To many, stochasticity, or more simply, noise, is just that – something which obscures patterns we are trying to infer (Knape & de Valpine 2011); and an ever richer batteries of statistical methods are developed largely in an attempt to strip away this undesirable randomness to reveal the patterns beneath (Coulson 2001). Over the past several decades, literature in stochasticity has transitioned from thinking of stochasticity in such terms; where noise is a nuisance that obscures the deterministic skeleton of the underlying mechanisms, to the recognition that stochasticity can itself be a mechanism for driving many interesting phenomena (Coulson *et al.* 2004). Yet this transition from *noise the nuisance* to *noise the creator* of ecological phenomena has had, with a few notable exceptions, relatively little impact in broader thinking about stochasticity. One of the most provocative of those exceptions has turned the classical notion of noise the nuisance on its head: recognising that noise driven phenomena can become a tool to reveal underlying processes: to become *noise the informer*. Here I argue that this third shift in perspective offers an opportunity to better bridge the divide between respective primarily theoretical and primarily empirical communities by seeing noise not as mathematical curiosity or statistical bugbear, but as a source for new opportunities for inference.

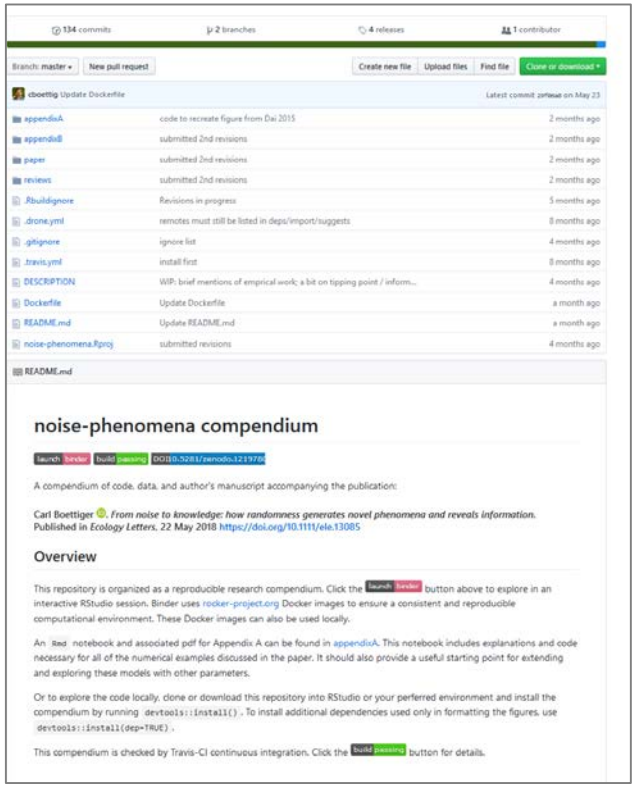
In arguing for this shift, it essential to recognise this is a call for a bigger tent, not for the rejection of previous paradigms. What I will characterize as 'noise the nuisance' reflects a predominately statistical approach, in which noise, almost by definition, represents all the processes we are not interested in that create additional variation which might obscure the pattern of interest. By contrast, an extensive literature has long explored how noise itself can create patterns and explain processes from population cycling to coexistence. These broad categories should be seen as a spectrum and not be mistaken for either a sharp dichotomy nor a reference to a strictly empirical-theoretical divide. Each paradigm expands upon rather than rejects the previous notion of noise: the recognition that noise can create novel phenomena does not mean that noise cannot also obscure the signal of some process of interest. Likewise, seeking to use noise as a novel source of information about underlying processes will be informed by both previous paradigms, as our discussion will illustrate.

Accompanying this discussion, I provide concise and commented code for simulating each of the models we will discuss as Appendix S1, and more mathematical background and derivations in Appendix S2. Numerical simulations permit poking and prodding investigation unencumbered by either experimental design or mathematical formalism. A copy of these appendices is maintained at https://github.com/cboettig/noise-phenomena.

To emphasise the underlying trend in the changing roles in which we see and understand noisy processes, I will also restrict my focus to relatively simple models primarily from population ecology context. Simplicity not only makes examples (in equations and in code) more tractable but also allows us to focus on aspects that are germane to many contexts rather than unique to particular complexities (Bartlett 1960;

https://onlinelibrary.wiley.com/doi/epdf/10.1111/ele.13085

Corresponding "compendium"
(with text, code, data)



noise-phenomena compendium

A compendium of code, data, and author's manuscript accompanying the publication:

Carl Boettiger. *From noise to knowledge: how randomness generates novel phenomena and reveals information.* Published in *Ecology Letters*, 22 May 2018 https://doi.org/10.1111/ele.13085

Overview

This repository is organized as a reproducible research compendium. Click the launch binder button above to explore in an interactive RStudio session. Binder uses rocker-project.org Docker images to ensure a consistent and reproducible computational environment. These Docker images can also be used locally.

An Rmd notebook and associated pdf for Appendix A can be found in appendixA. This notebook includes explanations and code necessary for all of the numerical examples discussed in the paper. It should also provide a useful starting point for extending and exploring these models with other parameters.

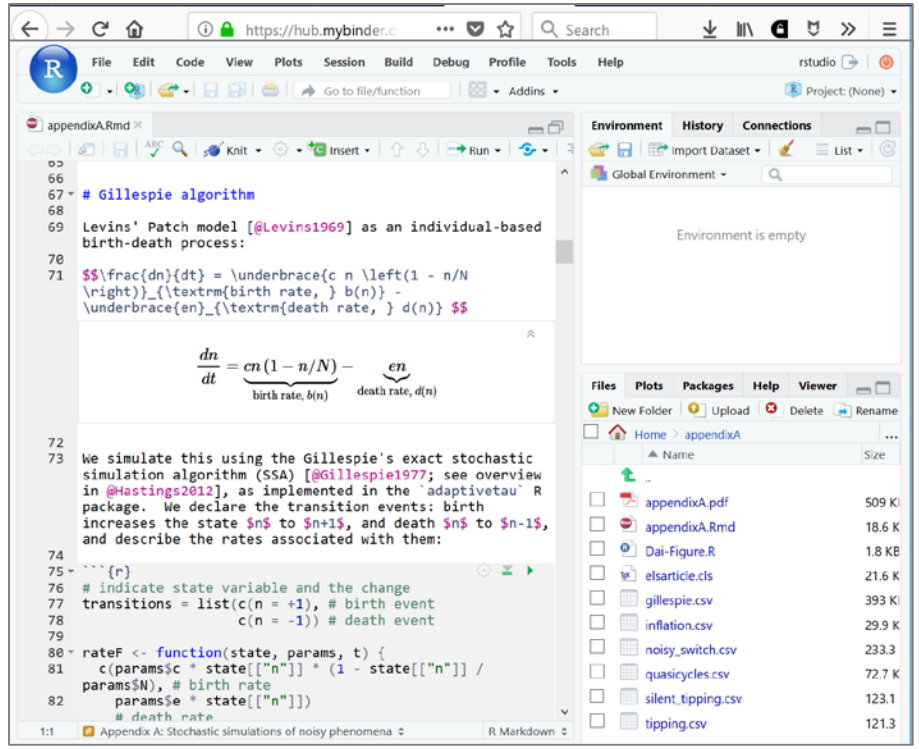
Or to explore the code locally, clone or download this repository into RStudio or your perferred environment and install the compendium by running `devtools::install()`. To install additional dependencies used only in formatting the figures, use `devtools::install(dep=TRUE)`.

This compendium is checked by Travis-CI continuous integration. Click the build passing button for details.

https://github.com/cboettig/noise-phenomena

Interactive R Markdown notebook to explore code

(with all required files and software ready to run in Rstudio in your browser with no installation required, thanks to Binder)



https://hub.mybinder.c

File Edit Code View Plots Session Build Debug Profile Tools Help

appendixA.Rmd

```
# Gillespie algorithm

Levins' Patch model [@Levins1969] as an individual-based
birth-death process:

$$\frac{dn}{dt} = \underbrace{c n \left(1 - n/N
\right)}_{\textrm{birth rate, } b(n)} -
\underbrace{en}_{\textrm{death rate, } d(n)} $$

$$\frac{dn}{dt} = \underbrace{cn(1-n/N)}_{\text{birth rate, } b(n)} - \underbrace{en}_{\text{death rate, } d(n)}$$

We simulate this using the Gillespie's exact stochastic
simulation algorithm (SSA) [@Gillespie1977; see overview
in @Hastings2012], as implemented in the `adaptivetau` R
package.  We declare the transition events: birth
increases the state $n$ to $n+1$, and death $n$ to $n-1$,
and describe the rates associated with them:

```{r}
# indicate state variable and the change
transitions = list(c(n = +1), # birth event
                   c(n = -1)) # death event

rateF <- function(state, params, t) {
  c(params$c * state[["n"]] * (1 - state[["n"]] /
params$N), # birth rate
    params$e * state[["n"]])
  # death rate
```
```

Environment History Connections

Import Dataset List

Global Environment

Environment is empty

Files Plots Packages Help Viewer

New Folder Upload Delete Rename

Home > appendixA

Name	Size
appendixA.pdf	509 KI
appendixA.Rmd	18.6 K
Dai-Figure.R	1.8 KB
elsarticle.cls	21.6 K
gillespie.csv	393 KI
inflation.csv	29.9 K
noisy_switch.csv	233.3
quasicycles.csv	72.7 K
silent_tipping.csv	123.1
tipping.csv	121.3

1:1 Appendix A: Stochastic simulations of noisy phenomena R Markdown

Seriously, why reproducible research?

As summarized by the ROpenSci Project:

- Show evidence of the correctness of our results
- Enable others to make use of our methods and results

http://ropensci.github.io/reproducibility-guide/sections/introduction/

Echoed by why Fernando Perez created Jupyter:

- **Ethical:** openness as fairness
- **Human/social:** openness fosters collaboration.
- **Epistemological:** proprietary science is an oxymoron.
- **Technical:** Python was cool :)

https://speakerdeck.com/fperez/project-jupyter-architecture-and-evolution-of-an-open-platform-for-modern-data-science

Attempts to establish a vocabulary for reproducibility

- ***Reproducibility*** *(also called repeatability or preproducibility) [6, 7]*
  - *Code, data (subject to privacy restrictions), software environment, and statistical methods are available so data analysis can be repeated and get a similar result [1, 2, 3]*
- ***Replicability*** *(also called reproducibility)*
  - *Another researcher is able to conduct the same experiment and get similar results*
- ***Correctness***
  - *Reproducible research can still be wrong! [4]*

→ There is an entire fascinating paper comparing how words used for types of reproducibility by decline [6]

1 - Peng *Science* 2011, DOI: 10.1126/science.1213847
2 - http://ropensci.github.io/reproducibility-guide/sections/introduction/
3 - https://simplystatistics.org/2014/06/06/the-real-reason-reproducible-research-is-important/
4 - Reproducible research can still be wrong, Leek, Peng PNAS 2015, DOI: 10.1073/pnas.1421412111
6 – Barba 2018 https://arxiv.org/abs/1802.03311
7 – Stark 2018 https://www.nature.com/articles/d41586-018-05256-0

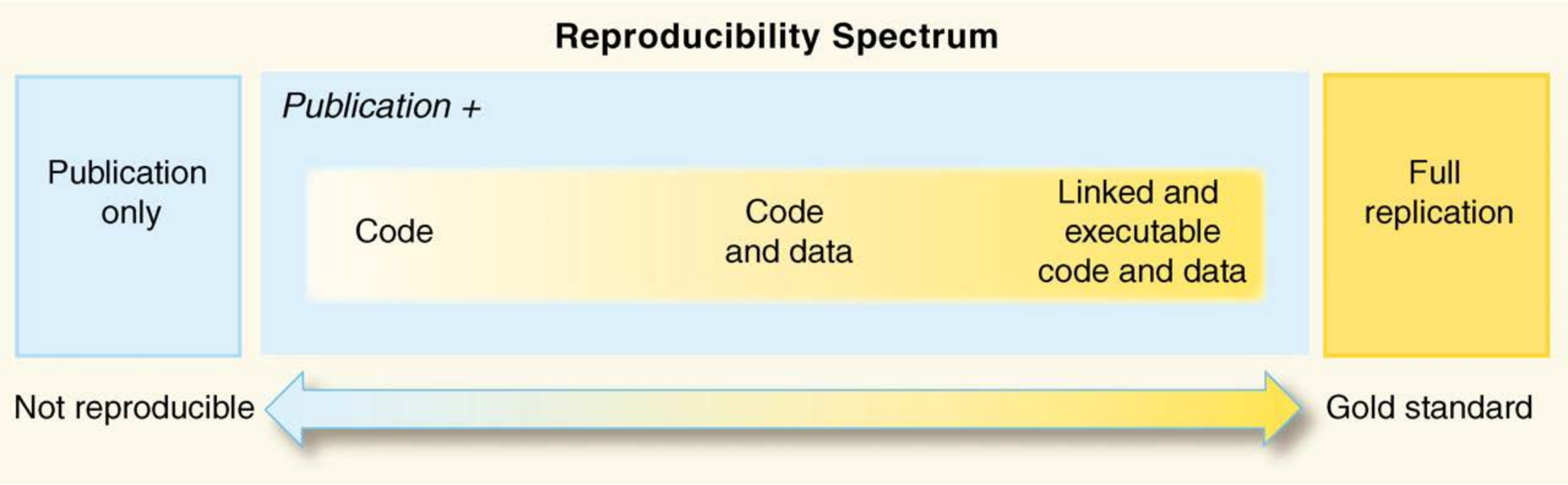
Attempts to establish a vocabulary for reproducibility within one field

Classification created by the American Society for Cell Biology [5]:

1. **analytic replication -** reproduce results by reanalysing original data
2. **direct replication** – repeat original experiment
3. **systematic replication** - repeat with different experimental conditions (eg different cell line)
4. **conceptual replication–** repeat validity of a concept (eg in different organisms)

5- https://www.nature.com/news/muddled-meanings-hamper-efforts-to-fix-reproducibility-crisis-1.20076#/b1, citing https://www.ascb.org/wp-content/uploads/2015/11/How-can-scientist-enhance-rigor.pdf citing Schmidt, Stefan. (2009). Shall We Really Do It Again? The Powerful Concept of Replication Is Neglected in the Social Sciences. Review of General Psychology. 13. 90-100. 10.1037/a0015108.

A reproducibility spectrum for computational research



Reproducible Research in Computational Science, Roger D. Peng
Science 02 Dec 2011 : 1226-1227 *http://science.sciencemag.org/content/334/6060/1226.full*

Reference – even more vocabulary for types of reproducibility

"Computational reproducibility:

- when detailed information is provided about code, software, hardware and implementation details.

Empirical reproducibility:

- when detailed information is provided about non-computational empirical scientific experiments and observations. In practise this is enabled by making data freely available, as well as details of how the data was collected.

Statistical reproducibility:

- when detailed information is provided about the choice of statistical tests, model parameters, threshold values, etc. This mostly relates to pre-registration of study design to prevent p-value hacking and other manipulations."

*Victoria Stodden, 2014, "**2014 : WHAT SCIENTIFIC IDEA IS READY FOR RETIREMENT?**" https://www.edge.org/response-detail/25340*

Reference – another reproducibility spectrum - any of these steps are helpful

Adopted from [Stodden et al 2013:]

***Reviewable Research*.**

- The descriptions of the research methods can be independently assessed and the results judged credible.

***Replicable Research*.**

- Tools are made available (might only be made available to referees or only upon request.) that would allow one to duplicate the results of the research, for example by running the authors' code to produce plots in the publication

***Confirmable Research*.**

- The main conclusions of the research can be attained independently without the use of software provided by the author. (But using the complete description of algorithms and methodology provided in the publication and any supplementary materials.)

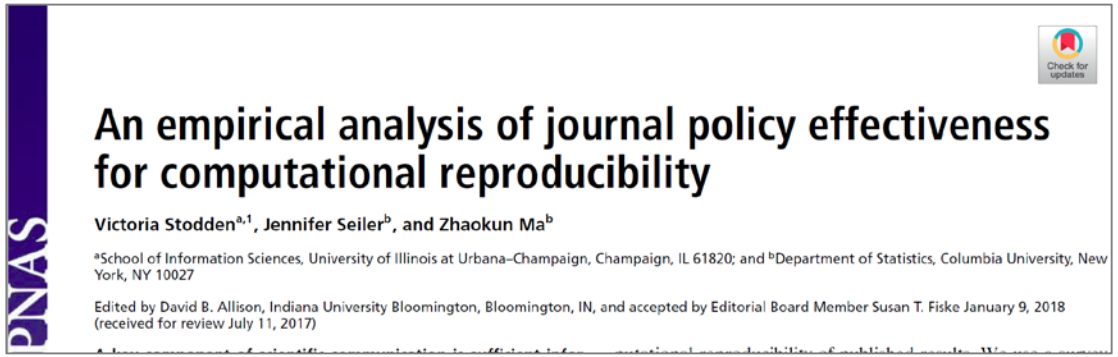
***Auditable Research*.**

- Sufficient records (including data and software) have been archived so that the research can be defended later if necessary or differences between independent confirmations resolved. The archive might be private, as with traditional laboratory notebooks.

***Open or Reproducible Research*.**

- Auditable research made openly available. This comprised well-documented and fully open code and allow one to (a) fully audit the computational procedure, (b) replicate and also independently reproduce the results of the research, and (c) extend the results or apply the method to new problems.

Reference– what is the state of practice?



PNAS

An empirical analysis of journal policy effectiveness for computational reproducibility

Victoria Stodden[a,1], Jennifer Seiler[b], and Zhaokun Ma[b]

[a]School of Information Sciences, University of Illinois at Urbana–Champaign, Champaign, IL 61820; and [b]Department of Statistics, Columbia University, New York, NY 10027

Edited by David B. Allison, Indiana University Bloomington, Bloomington, IN, and accepted by Editorial Board Member Susan T. Fiske January 9, 2018 (received for review July 11, 2017)

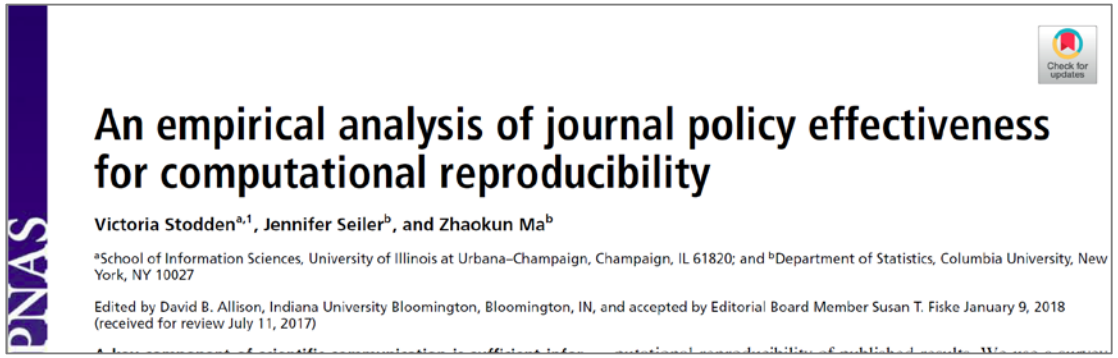
Stodden, Seiler, Ma, PNAS 2018

"This work evaluates the effectiveness of journal policy that requires the data and code necessary for reproducibility be made available postpublication by the authors upon request. We assess the effectiveness of such a policy by
(i) requesting data and code from authors and
(ii) attempting replication of the published findings.

We chose a random sample of 204 scientific papers published in the journal Science after the implementation of their policy in February 2011.

We found that we were able to obtain artifacts from 44% of our sample and were able to reproduce the findings for 26%."

Reference– what is the state of practice?



PNAS

An empirical analysis of journal policy effectiveness for computational reproducibility

Victoria Stodden[a,1], Jennifer Seiler[b], and Zhaokun Ma[b]

[a]School of Information Sciences, University of Illinois at Urbana–Champaign, Champaign, IL 61820; and [b]Department of Statistics, Columbia University, New York, NY 10027

Edited by David B. Allison, Indiana University Bloomington, Bloomington, IN, and accepted by Editorial Board Member Susan T. Fiske January 9, 2018 (received for review July 11, 2017)

[Stodden, Seiler, Ma, PNAS 2018]

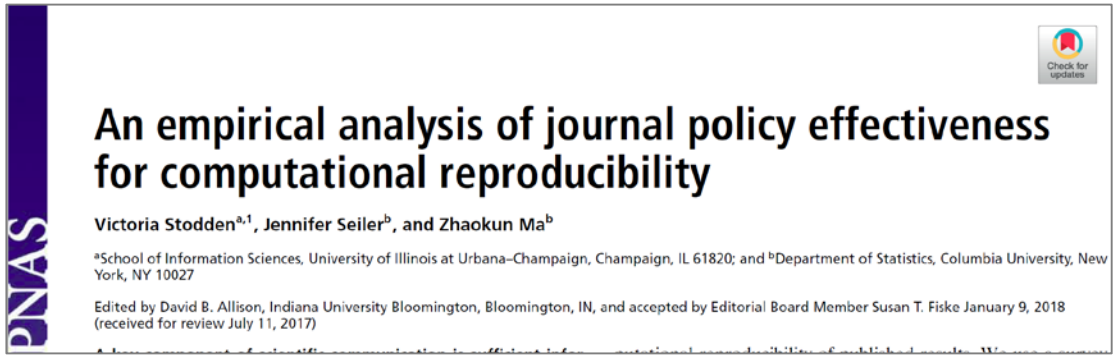
Table 5. Changes in disclosure practices

Disclosure practice	2009–2010, %	2011–2012, %
Citations to data and/or code in references	25	29
Data location given in acknowledgements	29	48
Code location given in acknowledgements	4	5

Table 6. Materials availability via inspection

Materials availability	2009–2010, %	2011–2012, %
Most or all relevant data locations given	52	75
Most or all relevant software locations given	43	54
Some data and software locations given	25	45
All major software and data locations given	15	25
Code, scripts, parameters, documentation	10	12
No supporting materials available	4	1

Reference– what is the state of practice?



PNAS

An empirical analysis of journal policy effectiveness for computational reproducibility

Victoria Stodden[a,1], Jennifer Seiler[b], and Zhaokun Ma[b]

[a]School of Information Sciences, University of Illinois at Urbana–Champaign, Champaign, IL 61820; and [b]Department of Statistics, Columbia University, New York, NY 10027

Edited by David B. Allison, Indiana University Bloomington, Bloomington, IN, and accepted by Editorial Board Member Susan T. Fiske January 9, 2018 (received for review July 11, 2017)

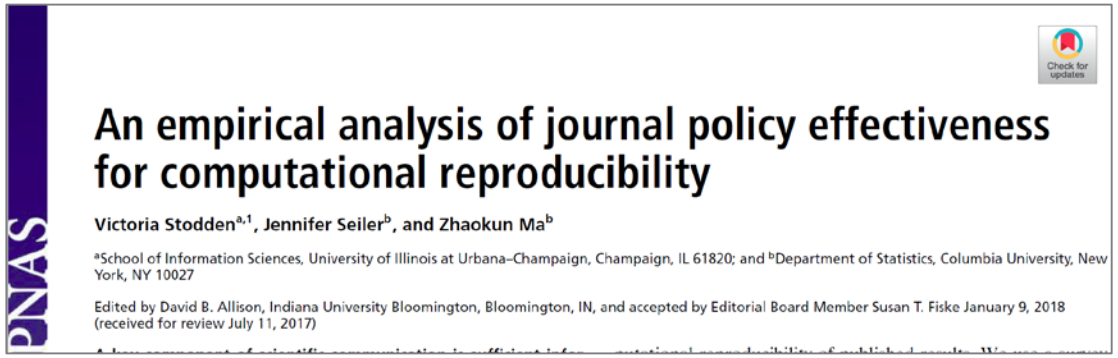
[Stodden, Seiler, Ma, PNAS 2018]

Table 1. Responses to emailed requests (n = 180)

Type of response	Count	Percent, %
Did not share data or code:		
Contact another person	20	1
Asked for reasons	20	1
Refusal to share	12	
Directed back to supplement	6	
Unfulfilled promise to follow up	5	
Impossible to share	3	2
Shared data and code	65	36
Email bounced	3	2
No response	46	26

~ 2% not possible to share
~ 35 % responded, did not share
~ 36 % shared
~ 28% unable to contact/ no response

Reference– what is the state of practice?



PNAS

An empirical analysis of journal policy effectiveness for computational reproducibility

Victoria Stodden[a,1], Jennifer Seiler[b], and Zhaokun Ma[b]

[a]School of Information Sciences, University of Illinois at Urbana–Champaign, Champaign, IL 61820; and [b]Department of Statistics, Columbia University, New York, NY 10027

Edited by David B. Allison, Indiana University Bloomington, Bloomington, IN, and accepted by Editorial Board Member Susan T. Fiske January 9, 2018 (received for review July 11, 2017)

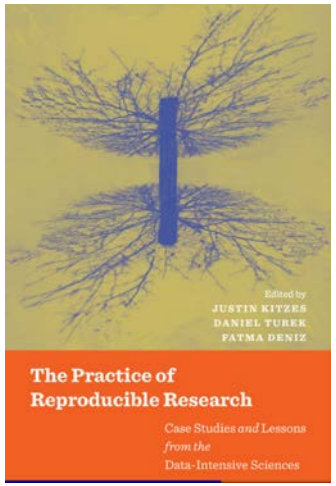
[Stodden, Seiler, Ma, PNAS 2018]

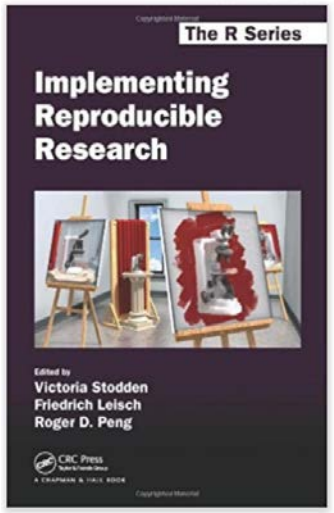
Table 4. Classification of reproducibility effort (n = 22)

Classification	Percent, %
Impossible to reproduce (missing essential code, data, or methodology)	5
Nearly impossible to reproduce (specialized hardware, intense computation requirements, sensitive data, human study, or other unavoidable reasons)	14
Difficult to reproduce because of unavoidable inherent complexity (e.g., requiring 300 million Markov chain Monte Carlo steps on each dataset, or needing months to do runs)	14
Reproducible with substantial tedious effort (e.g., individual download of a large number of datasets, hand coding of data into a new format, i.e., from an image, many archiving steps required)	5
Reproducible with substantial intellectual effort (e.g., methods well defined but required some knowledge of jargon or understanding of the field; or down the rabbit hole references to past articles required to reproduce; etc.)	5
Could reproduce with fairly substantial skill and knowledge (e.g., required GPU programing abilities to run code that wasn't given; translating complex models into MATLAB code; pseudo code with functions not detailed described in text into code; missing scripts)	23
Reproducible after tweaking (e.g., missing parameters required fiddling to find, missing modified code lines, missing arguments required for differing architecture; missing minor method step)	5
Minor difficulty in reproducing (e.g., installing a specialized library, converting to a different computational system)	18
Straightforward to reproduce with minimal effort	14

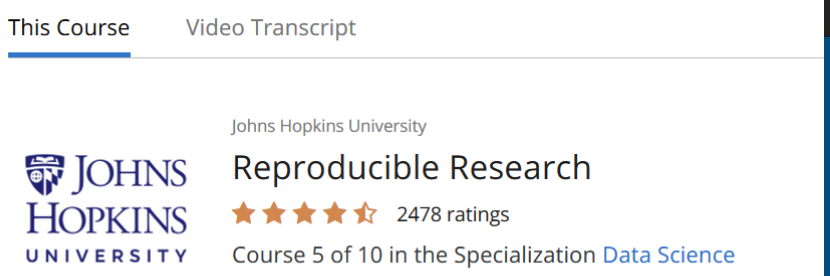
An abundance of resources + recommendations for reproducibility…

Books





Coursera



https://www.coursera.org/learn/reproducible-research/lecture/tVCoy/reproducible-research-concepts-and-ideas-part-1

Scholarly tools

https://dashboard.wholetale.org/

http://wholetale.org/runthrough.html

Free resources

eg Software Carpentry

R for Reproducible Scientific Analysis (English & Spanish)

Eg PyData videos (include R)

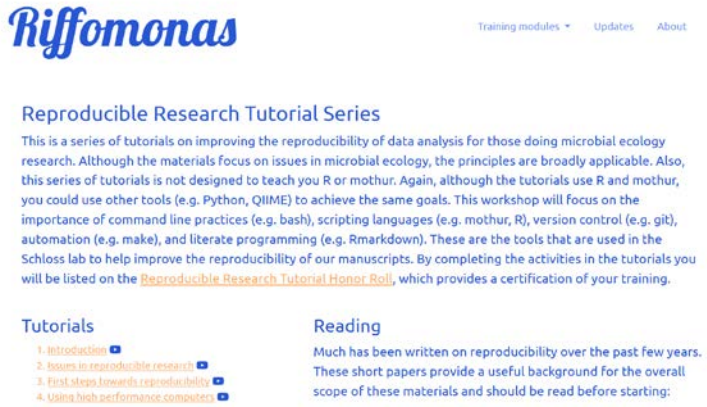
CarpentryCon West @

UC Davis June 29-July 1,

Includes Binder workshop

Video tutorials

(eg Pat Schloss, U Michigan)



https://github.com/riffomonas/reproducible_research

statements from scholarly groups, and more…

Two very common (and free) interactive notebooks:

Other interactive notebooks:
Beaker (Two Sigma)
Zeppelin (Apache)
Databricks (Databricks)

Jupyter

Sampling from the generative model

In this notebook, we will use the generative model of the HDHP (Hierarchical Dirichlet-Hawkes Process) in order to sample e
We will start with a predifined number of users, say 10, and we will attempt to model their behavior as they are posting ques
an online platform. For simplicity, our "vocabulary" will be dummy.

We start by importing all the libraries that will be required.

```
%matplotlib inline
import datetime
import string
import hdhp
import notebook_helpers
import seaborn as sns
```

Now, let us set some parameters for our model. These fall under two categories; the ones relevant to the content and then c

```
vocabulary = ['word' + str(i) for i in range(100)]  # the `words` of our documents
doc_min_length = 5
doc_length = 10
words_per_pattern = 50

alpha_0 = (2.5, 0.75)
mu_0 = (2, 0.5)
```

```
overlap = notebook_helpers.compute_pattern_overlap(process)
sns.distplot(overlap, kde=True, norm_hist=True, axlabel='Content overlap')
```

```
Average overlap: 0.338826769742
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x10de8ca50>
```

R Markdown

Unit 3: Fisheries Collapse Module

This module will focus on understanding and replicating fisheries stock assessment data and fisheries collapse.

The Database

We will use data from the RAM Legacy Stock Assessment Database

First, load in the necessary libraries. Note that this time we need a package we haven't used before `readxl`. This package is useful for reading in .xls or .xlsx files. As always if you want more info on a package run `?readxl` after loading it.

```
library("tidyverse")
library("readxl")
library("scales") # For y-axis labels not in scientific notation - is there a better way to do this since 2012?
```

Reading in the tables

```
download.file("https://depts.washington.edu/ramlegac/wordpress/databaseVersions/RLSADB_v3.0_(assessment_data_only)_excel.zip",
              "ramlegacy.zip")
path <- unzip("ramlegacy.zip")  # unzip the .xls files
sheets <- readxl::excel_sheets(path) # use the readxl package to identify sheet names

# purrr:map is the tidyverse version of lapply
ram <- lapply(sheets, readxl::read_excel, path = path)  # read the data from all 3 sheets into a list
names(ram) <- sheets # give the list of datatables their assigned sheet names

## check the names
names(ram)
```

```
##  [1] "area"                   "assessment"
##  [3] "assessmethod"           "assessor"
##  [5] "biometrics"             "bioparams"
##  [7] "bioparams_ids_views"    "bioparams_units_views"
##  [9] "bioparams_values_views" "management"
## [11] "stock"                  "taxonomy"
## [13] "timeseries"             "timeseries_ids_views"
## [15] "timeseries_units_views" "timeseries_values_views"
## [17] "tsmetrics"
```

```
## check our data
```

https://github.com/activityhistory

https://github.com/ds421/fish-template/blob/master/assignment/fish-assignment.Rmd

Similarities of Jupyter and R notebooks – compatible with best practices

Embed code in a narrative written for humans to read ('literate programming')

Script repetitive tasks so computers, not humans, repeat them

Write code in small steps with immediate feedback

Visualize data frequently

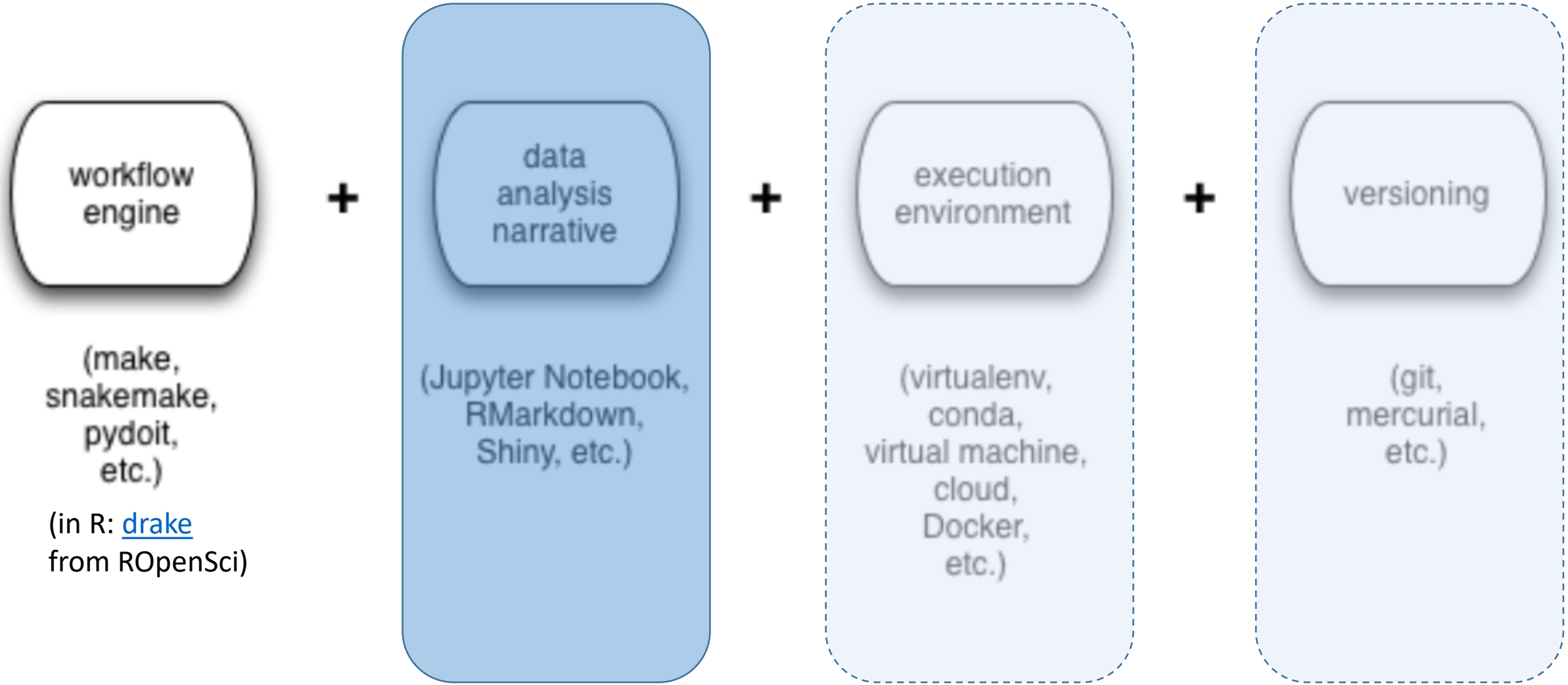
Import mature packages/libraries to build on existing methods

Use version control (good for R, Jupyter is still harder)

Easy to share with other people

All but data visualization based on Wilson G, Aruliah DA, Brown CT, Chue Hong NP, Davis M, Guy RT, et al. (2014) Best Practices for Scientific Computing. PLoS Biol 12(1): e1001745. https://doi.org/10.1371/journal.pbio.1001745

Interactive notebooks are one part of a toolset for reproducibility



Background– what is literate programming?

https://en.wikipedia.org/wiki/Donald_Knuth#/media/File:KnuthAtOpenContentAlliance.jpg

"Instead of imagining that our main task is to instruct a *computer* what to do, let us concentrate rather on explaining to *human beings* what we want a computer to do."

- Donald Knuth (1984)

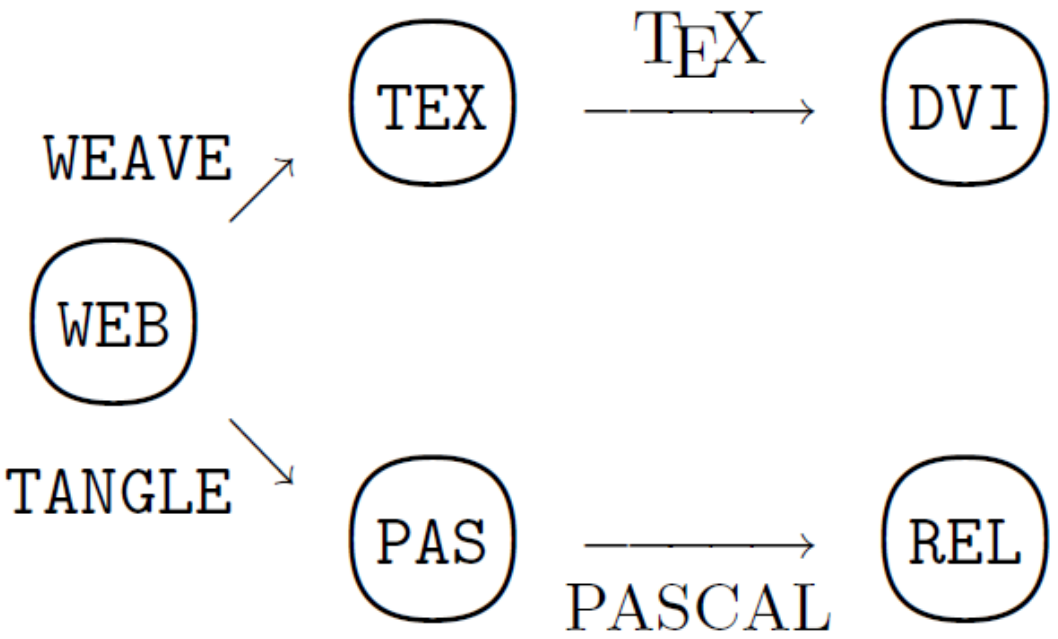


Figure 1. Dual usage of a WEB file.

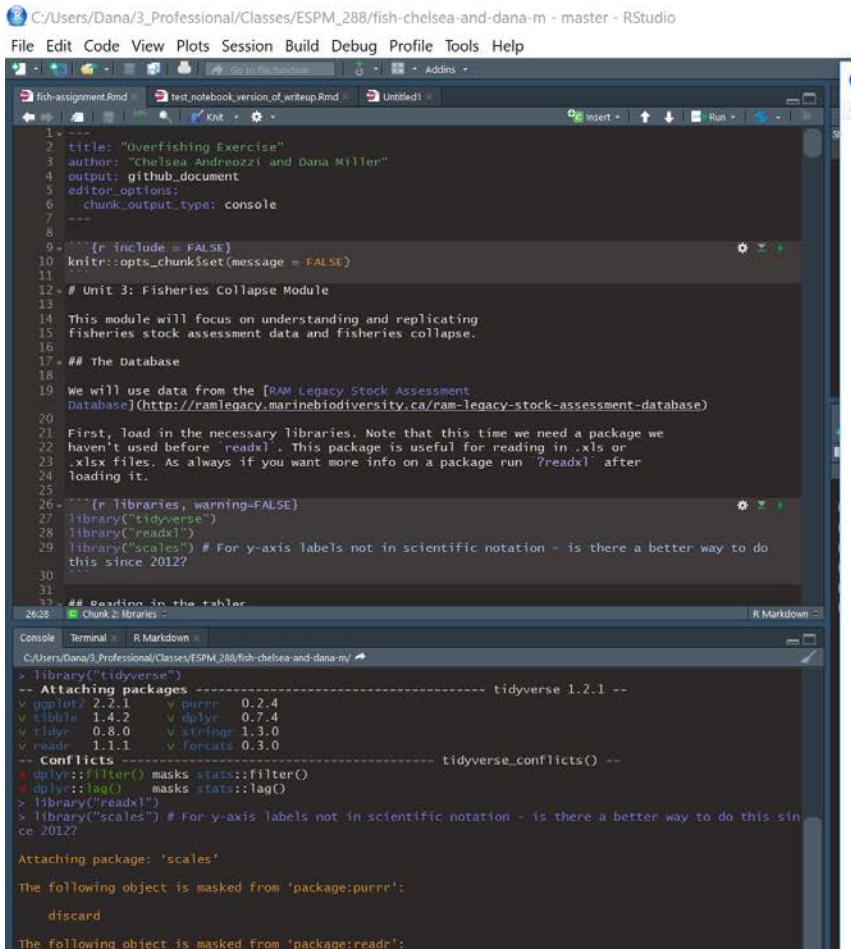
D. E. Knuth; Literate Programming, The Computer Journal, Volume 27, Issue 2, 1 January 1984, Pages 97–111, https://doi.org/10.1093/comjnl/27.2.97

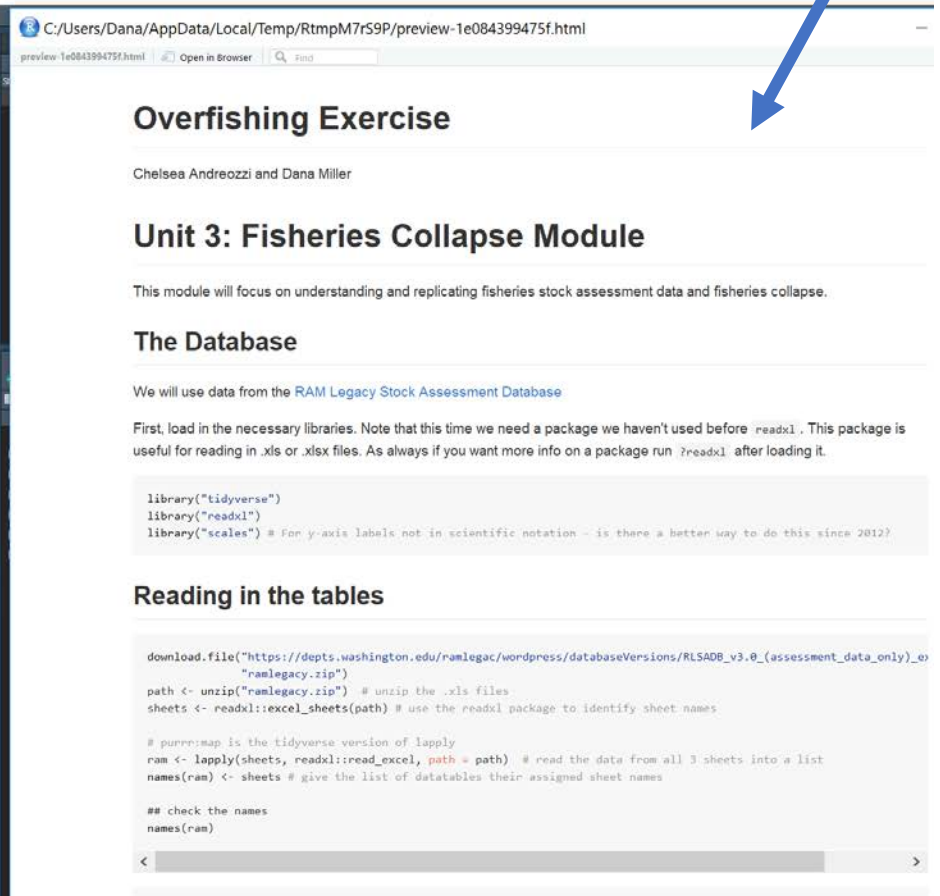
Literate programming example with R Markdown

Starting with one .Rmd file with both R code and text

This button or command:

executes all R code (output shown in console)





This button or command:

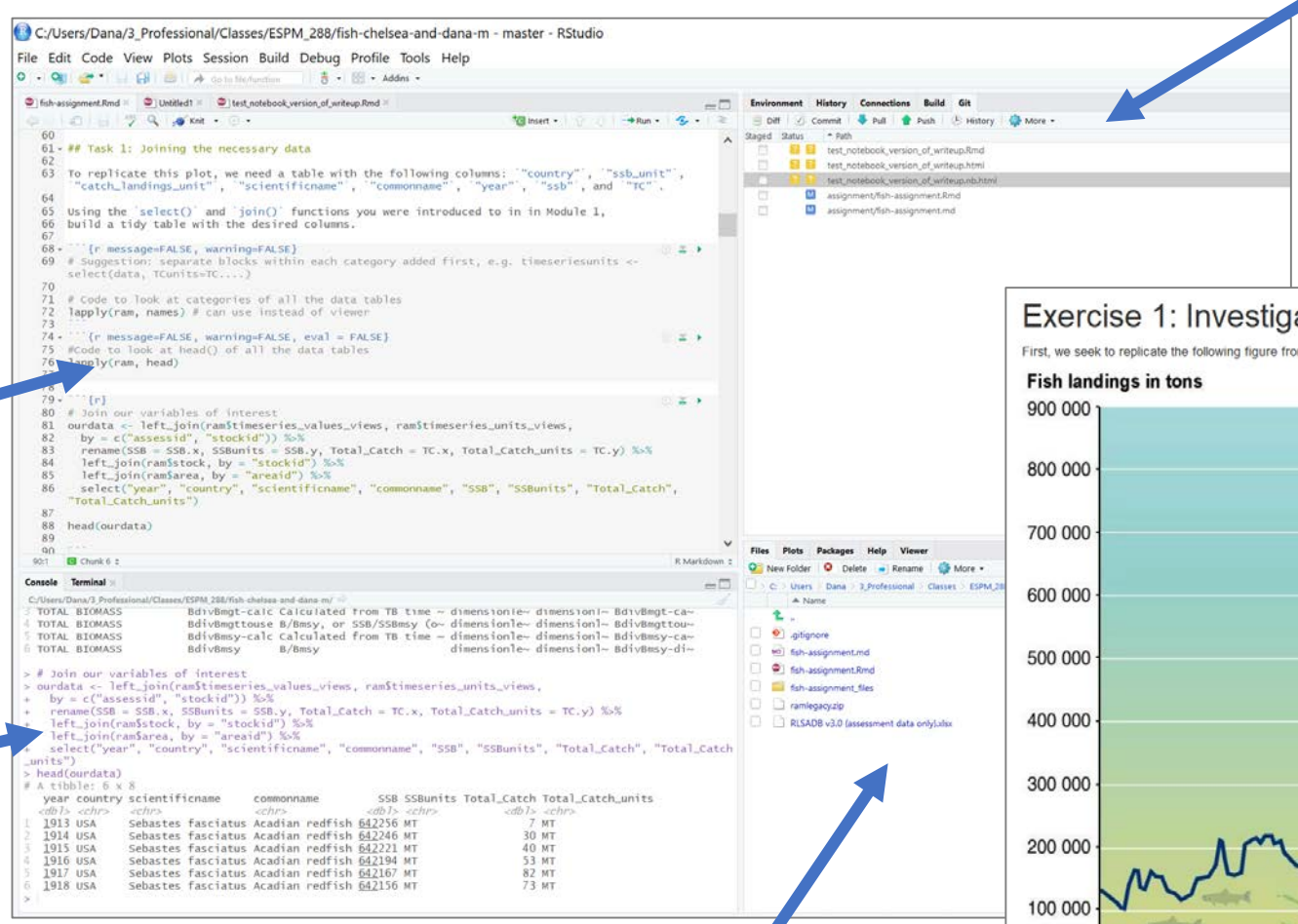
renders formatted document with executed code and text (.md, .pdf, .doc etc)

R Markdown documents – key features

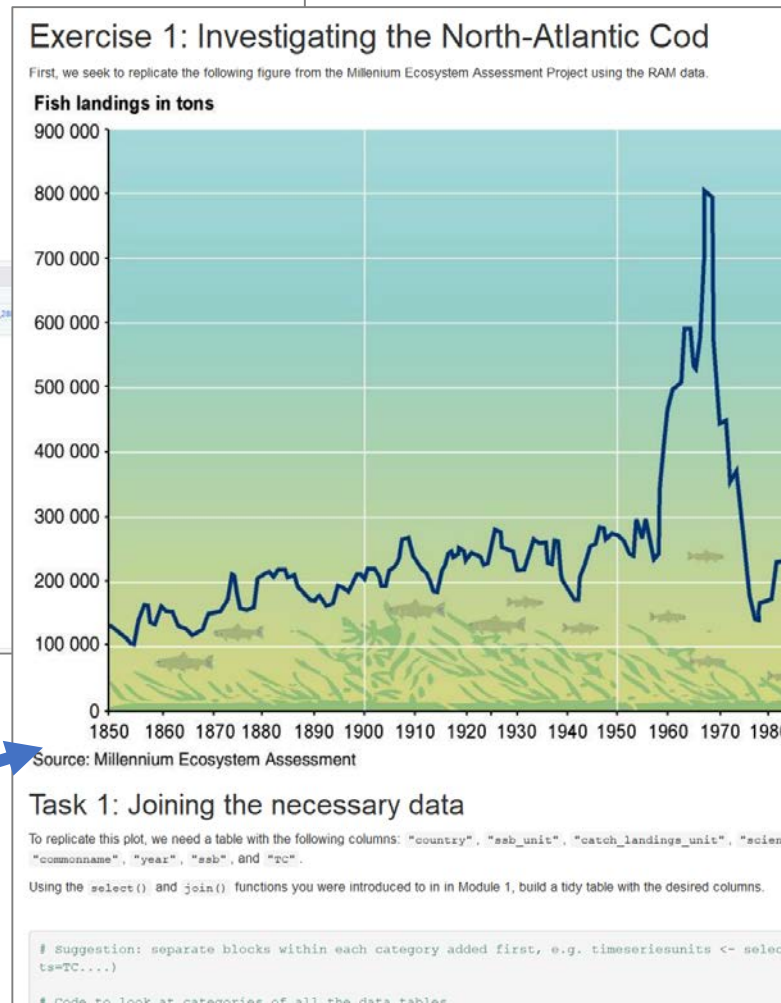
History
Rstudio ~ 2011
knitr ~ 2012
Markdown rendering ~ 2013
R Notebook ~ 2016

R Markdown (.Rmd) document mixing code and text

Code can be executed in-line (above) or in console (below)



Version control tracker
See happygitwithr



See file explorer, help documentation, plots, packages

Convert file to .md, .doc, .pdf, html, html notebook, slides, book, blog, dashboards, Shiny app, etc
See R Markdown output formats

What is markdown?

Text using Markdown syntax

```
Heading
=======

## Sub-heading

Paragraphs are separated
by a blank line.

Two spaces at the end of a line
produces a line break.

Text attributes _italic_,
**bold**, `monospace`.

Horizontal rule:

---

Bullet list:

  * apples
  * oranges
  * pears
```

Corresponding HTML produced by a Markdown processor

```
<h1>Heading</h1>

<h2>Sub-heading</h2>

<p>Paragraphs are separated
by a blank line.</p>

<p>Two spaces at the end of a line<br />
produces a line break.</p>

<p>Text attributes <em>italic</em>,
<strong>bold</strong>,
<code>monospace</code>.</p>

<p>Horizontal rule:</p>

<hr />

<p>Bullet list:</p>

<ul>
<li>apples</li>
<li>oranges</li>
```

Text viewed in a browser

Heading [edit]

Sub-heading [edit]

Paragraphs are separated by a blank line.

Two spaces at the end of a line
produces a line break.

Text attributes *italic*, **bold**, `monospace`.

Horizontal rule:

Bullet list:

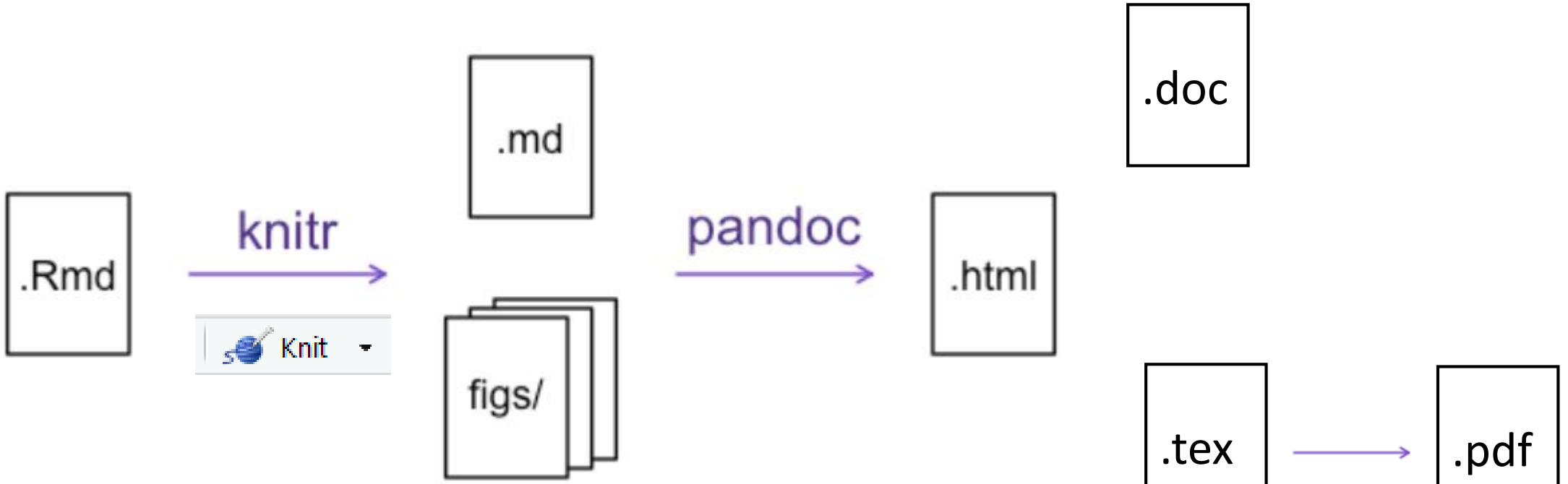
- apples
- oranges
- pears

Numbered list:

1. wash
2. rinse
3. repeat

https://en.wikipedia.org/wiki/Markdown

Whoa, how does knitting work?



https://usesthis.com/interviews/yihui.xie/

Yihui Xie
RStudio
(Iowa State Statistics PhD)

https://philosophy.berkeley.edu/people/detail/1

John Macfarlane
Philosophy, UC Berkeley

[Try pandoc]

https://swcarpentry.github.io/r-novice-gapminder/15-knitr-markdown/

Demo: R Markdown

Learning objectives:

1.

- How to create a new R Markdown document in Rstudio
- Make edits
- Knit it
- Share it somewhere
- Have someone else download and run it

2.

- Example of assignments written in R Markdown

Reference to set up git w/ R: *http://happygitwithr.com*

Limitations of R + RMarkdown

- Not a "general purpose programming language"
- By default, loads entire data into memory (not w/ sparklyr)
- Less widely used in industry than python
- Possible to run chunks out of order (but can "run all")

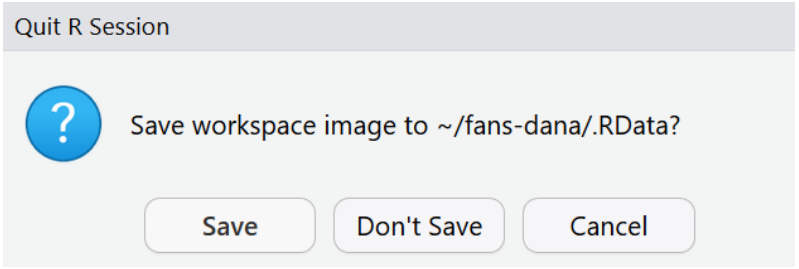
Three R tips learners can start using today:

- here::here (use with R Project)
  - Freedom from hardcoded file paths!

```
library(here)
db <- read_csv(here("subfolder_name", "file_name.csv"))
```

- read_csv
  - tidyverse version of read.csv
  - won't coerce strings to factors, outputs a dataframe (tibble)

- "Save workspace image?"
  - Don't save!



More R resources

R for Data Science book:
https://r4ds.had.co.nz

ROpenSci tools:
https://ropensci.org/

Reference for R with git:
http://happygitwithr.com

Stat 545:
http://stat545.com/topics.html

Software Carpentry reproducible research lesson:
https://swcarpentry.github.io/r-novice-gapminder/15-knitr-markdown/

Packrat (package manager)

Thesis writing:
https://eddjberry.netlify.com/post/writing-your-thesis-with-bookdown/

https://rosannavanhespenresearch.wordpress.com/2016/02/03/writing-your-thesis-with-r-markdown-1-getting-started/

Website w/ blogdown:
https://twitter.com/dsquintana/status/993410504570888192

R notebooks online: https://rpubs.com/

Dplyr cheatsheet
http://stat545.com/bit001_dplyr-cheatsheet.html

Journal R templates:
https://github.com/rstudio/rticles

Talk on teaching R:
http://michaellevy.name/blog/useR-talk-on-teaching-R/

Bonus:
One of many grad stats textbooks in R
http://xcelab.net/rm/statistical-rethinking/

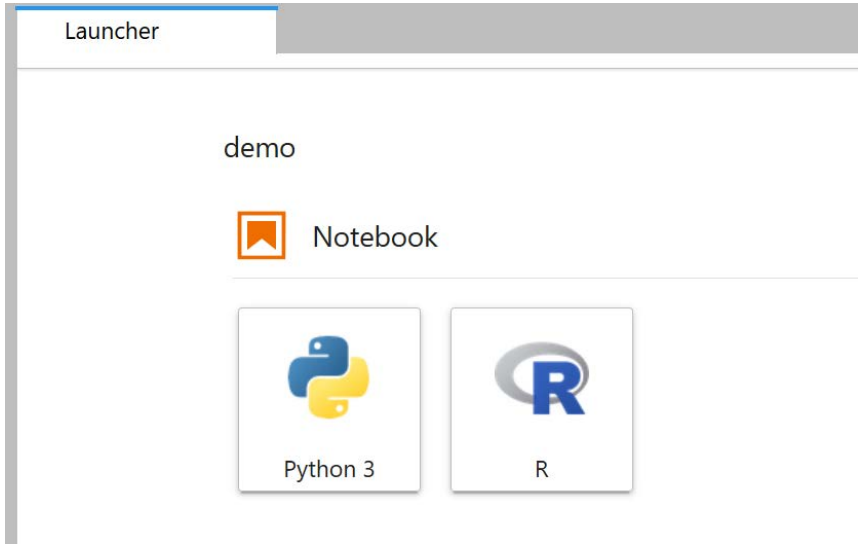
Workflow manager (like make) for R:
https://github.com/ropensci/drake

What about R and Python? They can be friends

Interoperability – though mastery of either takes plenty of time

Run R in Jupyter

- Run R inside python with rpy2 (formally rmagics)
- Or run R kernel (supported in Jupyter and JupyterLab)
  - R in Jupyter demo



Run Python in R

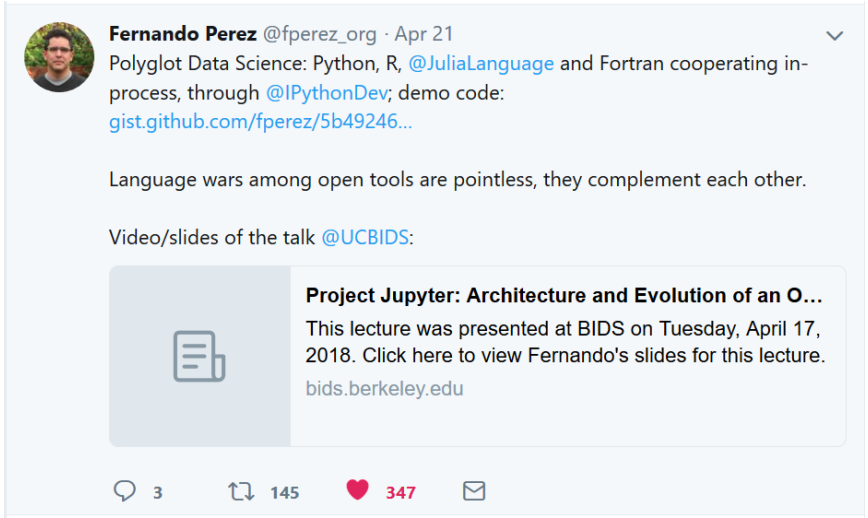
- Reticulate

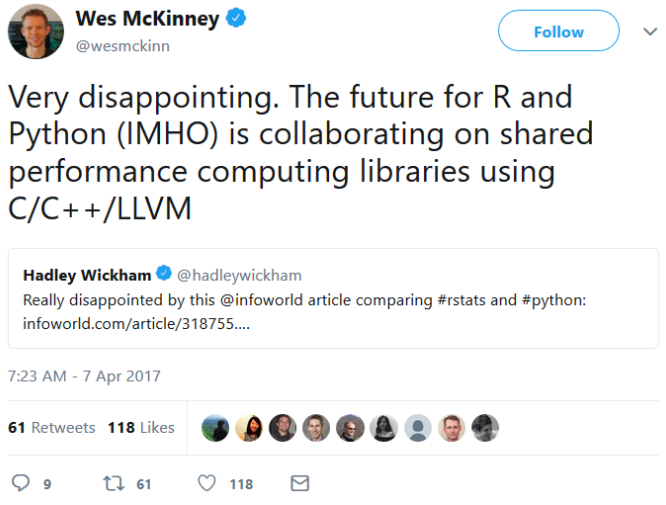
```
13
14 ```{python}
15 import pandas
16 flights = pandas.read_csv("flights.csv")
17 flights = flights[flights['dest'] == "ORD"]
18 flights = flights[['carrier', 'dep_delay', 'arr_delay']]
19 flights = flights.dropna()
20 ```
21
22 ```{r, fig.width=7, fig.height=3}
23 library(ggplot2)
24 ggplot(py$flights, aes(carrier, arr_delay)) + geom_point() + geom_jitter()
25 ```
26
```

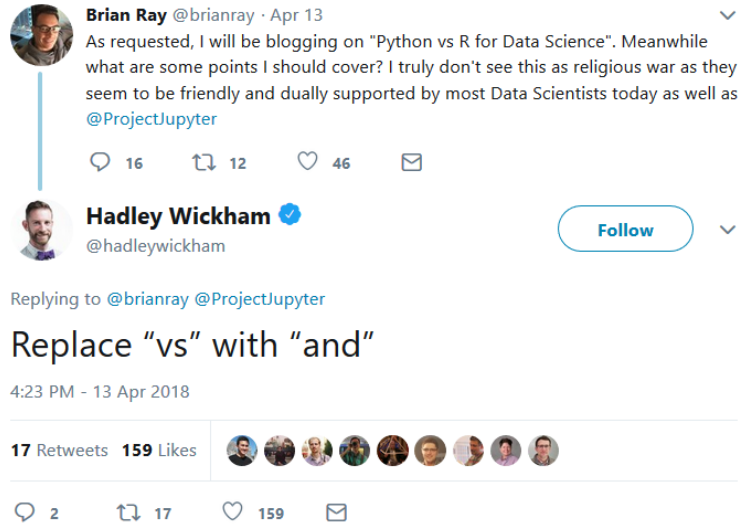
https://rstudio.github.io/reticulate/

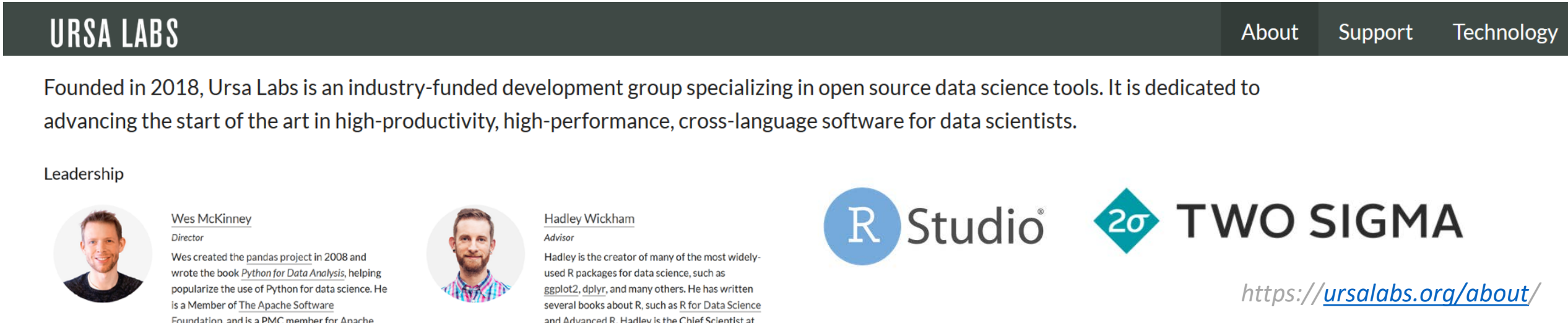
R and Python can be friends

Thankfully, key RStudio, python, and Jupyter developers have no patience for "language wars"









https://ursalabs.org/about/

Jupyter notebooks – origins

Watch the talk:

https://bids.berkeley.edu/events/project-jupyter-architecture-and-evolution-open-platform-modern-data-science

Slides:

https://speakerdeck.com/fperez/project-jupyter-architecture-and-evolution-of-an-open-platform-for-modern-data-science

Jupyter notebooks – key features

Narrative Text

Notebook title and introduction

Description of model parameters

Description of need to profile data

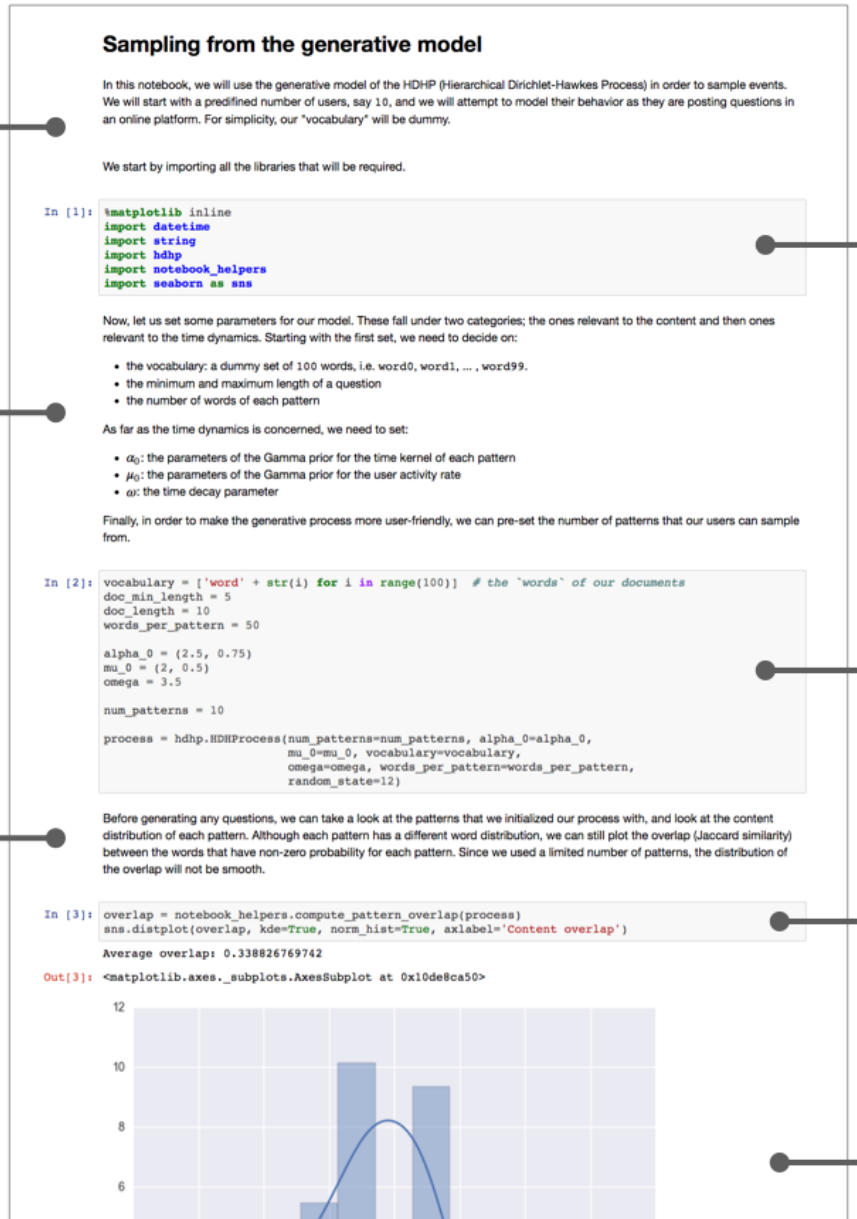
Code and Visualizations

Importing external packages

Implementation of parameters

Profile plotting code

Inline plot



Sampling from the generative model

In this notebook, we will use the generative model of the HDHP (Hierarchical Dirichlet-Hawkes Process) in order to sample events. We will start with a predifined number of users, say 10, and we will attempt to model their behavior as they are posting questions in an online platform. For simplicity, our "vocabulary" will be dummy.

We start by importing all the libraries that will be required.

In [1]:
```
%matplotlib inline
import datetime
import string
import hdhp
import notebook_helpers
import seaborn as sns
```

Now, let us set some parameters for our model. These fall under two categories; the ones relevant to the content and then ones relevant to the time dynamics. Starting with the first set, we need to decide on:

- the vocabulary: a dummy set of 100 words, i.e. word0, word1, ... , word99.
- the minimum and maximum length of a question
- the number of words of each pattern

As far as the time dynamics is concerned, we need to set:

- α_0: the parameters of the Gamma prior for the time kernel of each pattern
- μ_0: the parameters of the Gamma prior for the user activity rate
- ω: the time decay parameter

Finally, in order to make the generative process more user-friendly, we can pre-set the number of patterns that our users can sample from.

In [2]:
```
vocabulary = ['word' + str(i) for i in range(100)]  # the `words` of our documents
doc_min_length = 5
doc_length = 10
words_per_pattern = 50

alpha_0 = (2.5, 0.75)
mu_0 = (2, 0.5)
omega = 3.5

num_patterns = 10

process = hdhp.HDHProcess(num_patterns=num_patterns, alpha_0=alpha_0,
                          mu_0=mu_0, vocabulary=vocabulary,
                          omega=omega, words_per_pattern=words_per_pattern,
                          random_state=12)
```

Before generating any questions, we can take a look at the patterns that we initialized our process with, and look at the content distribution of each pattern. Although each pattern has a different word distribution, we can still plot the overlap (Jaccard similarity) between the words that have non-zero probability for each pattern. Since we used a limited number of patterns, the distribution of the overlap will not be smooth.

In [3]:
```
overlap = notebook_helpers.compute_pattern_overlap(process)
sns.distplot(overlap, kde=True, norm_hist=True, axlabel='Content overlap')
```

```
Average overlap: 0.338826769742
```

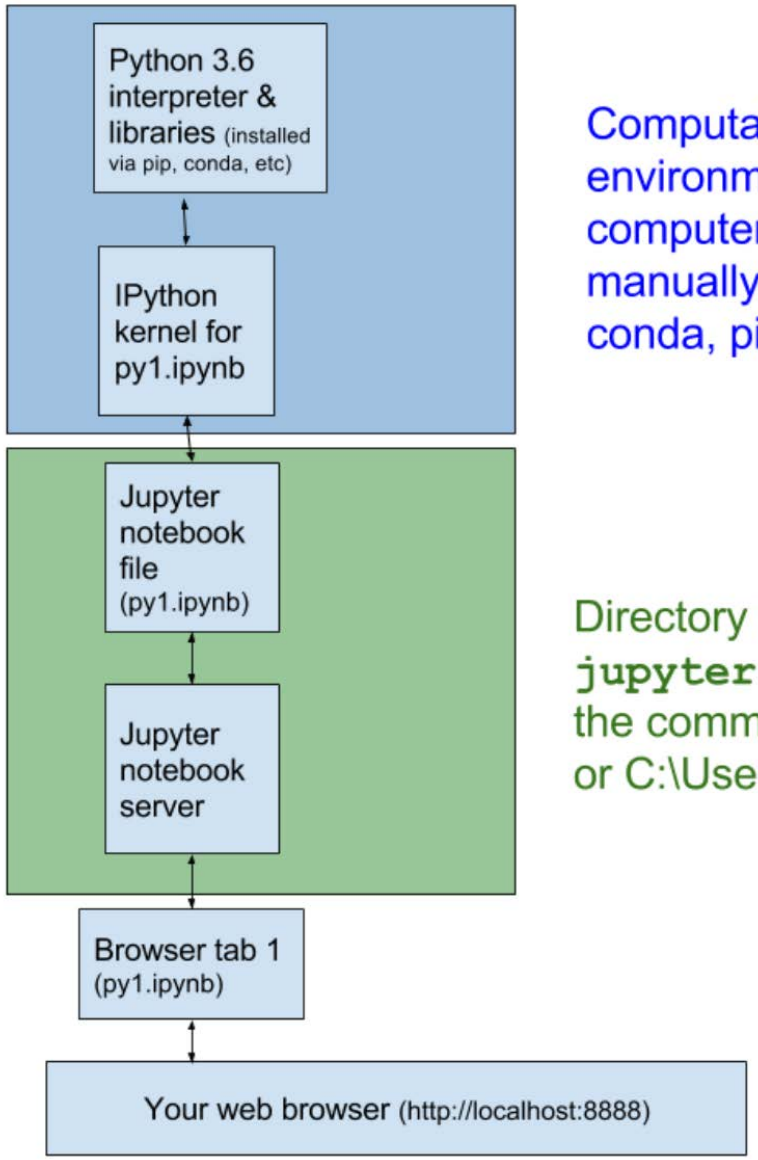
Out[3]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x10de8ca50>
```


C San Diego & Université Lyon 1

ttps://github.com/activityhistory

allery of interesting notebooks

How do notebooks work?



Computational environment on your computer (installed manually or managed by conda, pip, apt-get, etc)

Directory where you run `jupyter notebook` from the command line (/home/stu/ or C:\Users\Stu)

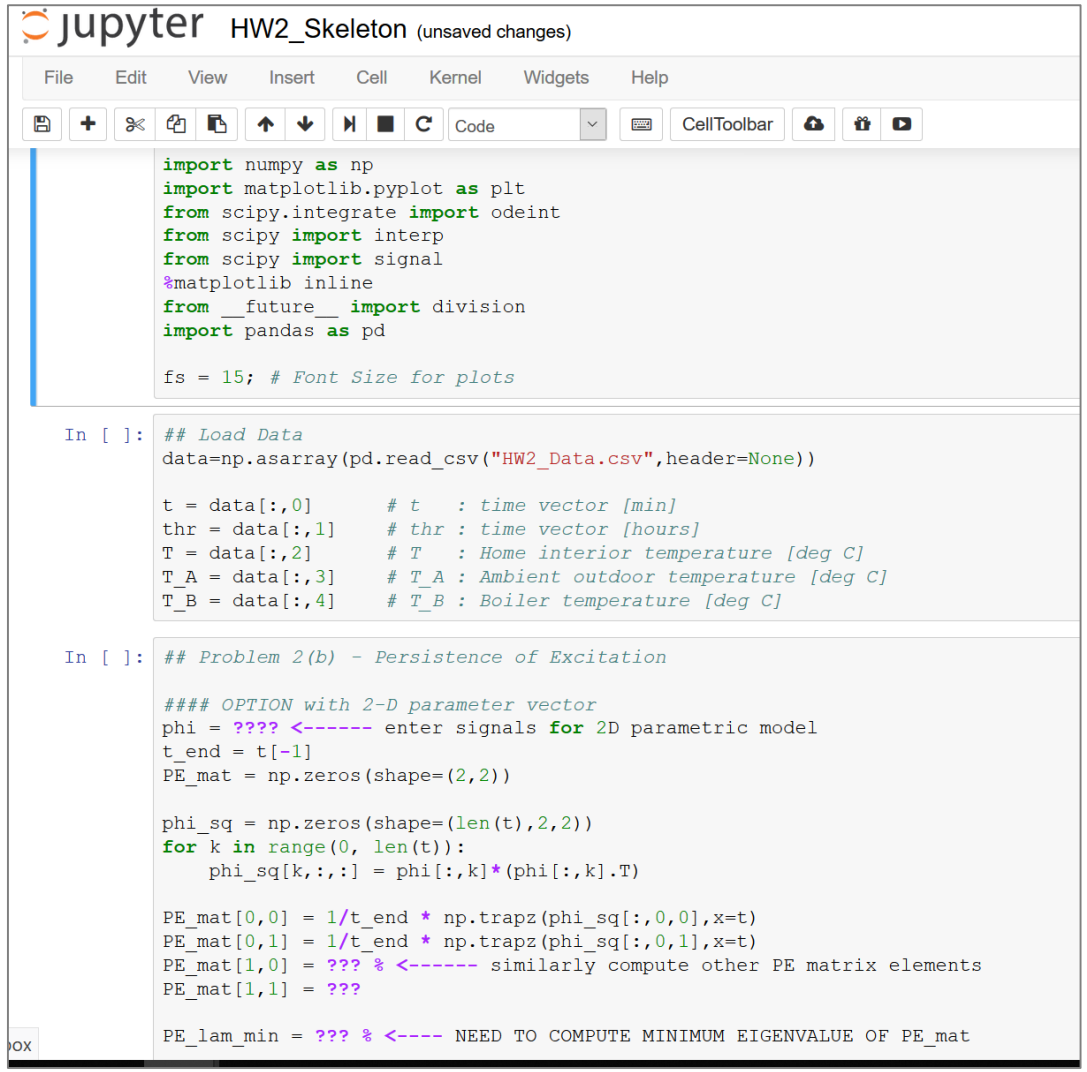
Ways to share Jupyter notebooks:

- Static
  - nbviewer
  - GitHub
- Interactive
  - Download and run
  - Share with Binder

[Stuart Geiger, http://www.thehackerwithin.org/berkeley/posts/2017-10-11-jupyter-f17.html](http://www.thehackerwithin.org/berkeley/posts/2017-10-11-jupyter-f17.html)

Course content in Jupyter

eg Homework in Jupyter notebooks



Numerous entire classes shared online:

An introductory notebook on uncertainty quantification and sensitivity analysis (for cardiovascular modelling)

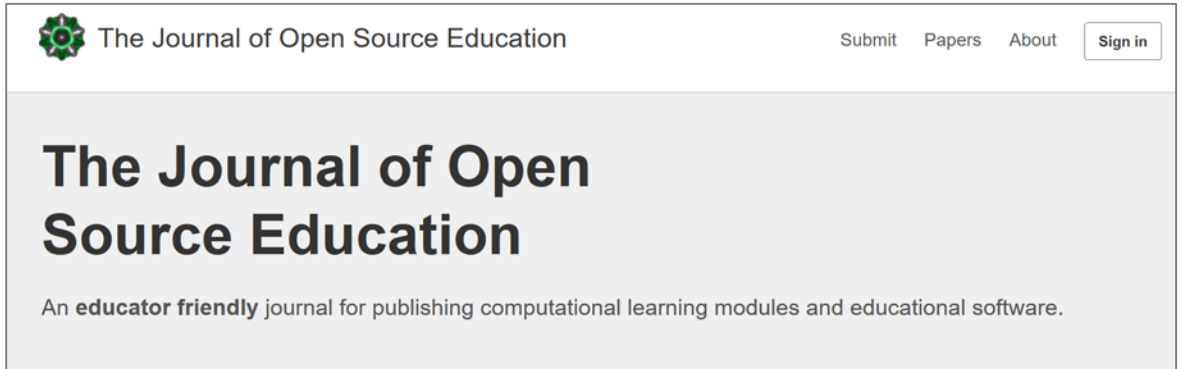
CFD Python: 12 steps to Navier-Stokes

Pytherm – applied thermodynamics

UCB Data 8 - Foundations of Data Science

More tools for educators + researchers:

nbgrader – assigning + grading Jupyter notebooks



Demo: Jupyter notebooks

Learning objectives:

1. Distinguish between static and interactive formats of sharing and viewing Jupyter notebooks
    - **Static:** nbviwer or preview on GitHub
        - Example publication with data + Jupyter notebook
        - Example notebook accompanying publication
    - **Interactive:** in Binder, or run locally

2. Run a Jupyter notebook locally
    - *Note: requires having Python installed*

3. Write some code, save + share it

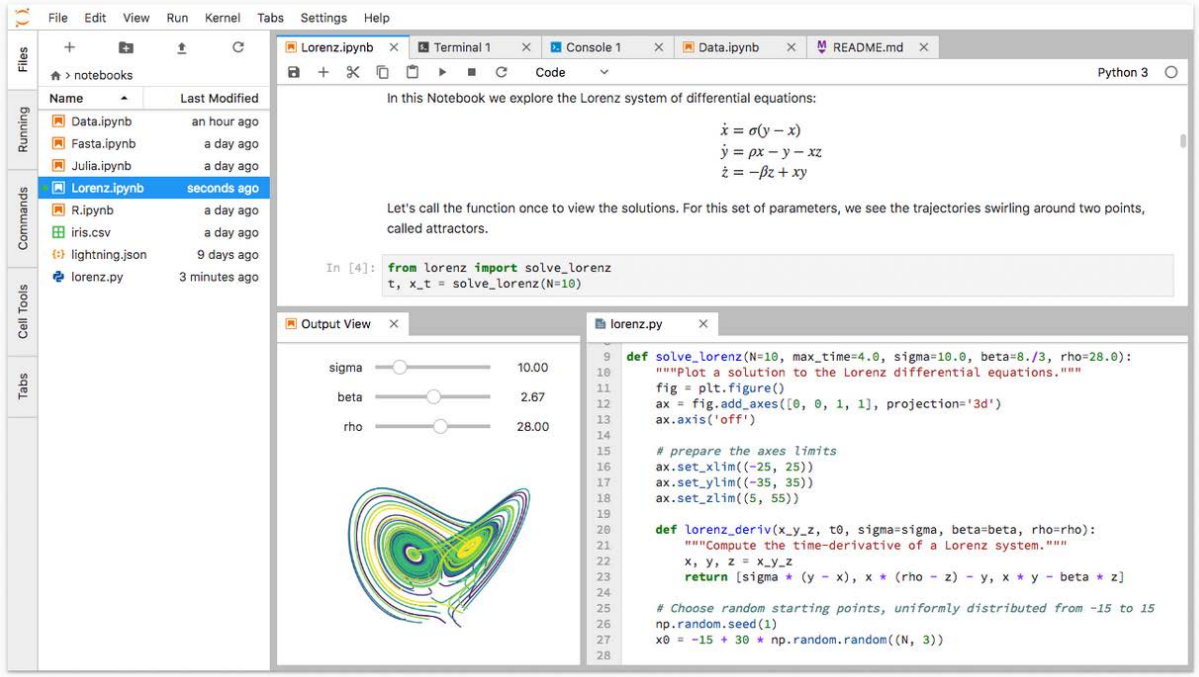
Current citation for Jupyter (2016)

Limitations of Jupyter Notebooks

- Challenging to version control
- No linting (autocorrect)
- Linear narrative structure
- Possible to run cells out of order
- Tricky to manage add-ons (plugins)
- Not an environment for software engineering

The future of Jupyter

Jupyter Lab (notebook will eventually retire)

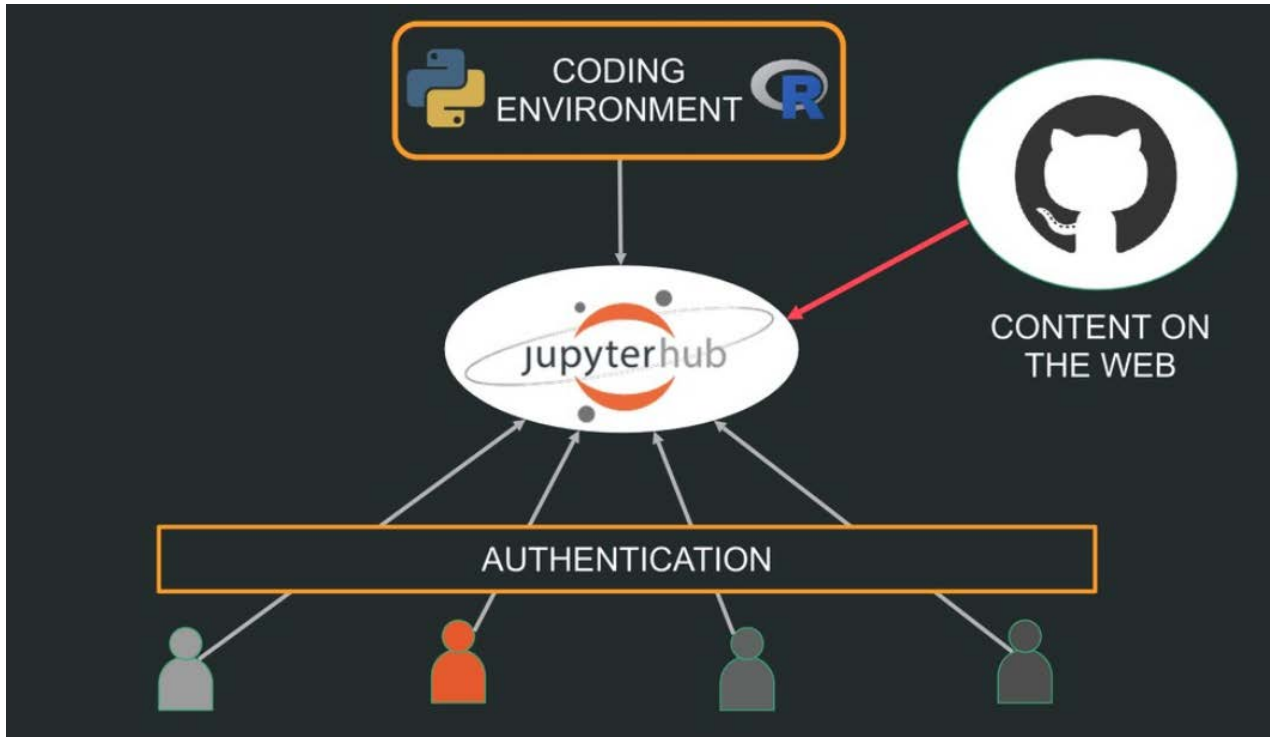


[Launched Feb 2018](#) – includes demo in Docker

JupyterCon 2017, "JupyterLab: The Next-Generation Jupyter Frontend"
https://www.youtube.com/watch?v=w7jq4XgwLJQ

Jupyter Hub

(no local install required, used in Data8)



Slide from Fernando Perez's talk at BIDS:
https://speakerdeck.com/fperez/project-jupyter-architecture-and-evolution-of-an-open-platform-for-modern-data-science

Gallery of JupyterHub deployments

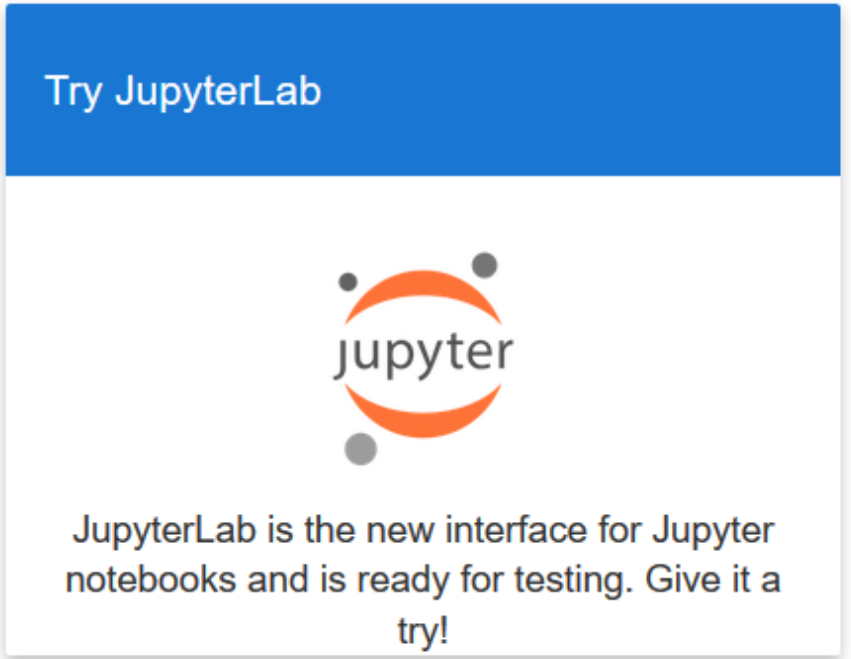
Demo: Jupyter Lab

1. Go to https://jupyter.org/try
   - Open a Jupyter Lab demo with the button shown below

   - Leaning objectives
     - Open a notebook

     - Try rearranging panes
       (can be similar to Rstudio layout)

     - Appreciate scrolling smoothly through demo/big.csv, a file with too many rows (>1.1 Million) to open in Excel



A rough comparison:

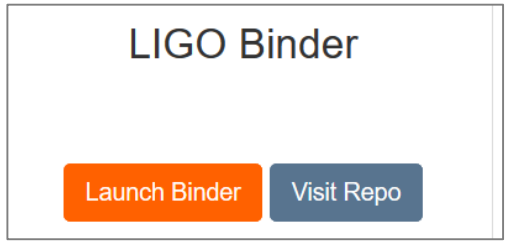
	Jupyter Notebook (.ipynb)	R Markdown document (.Rmd)
Interactive execution	In browser (after download, or hosted remotely)	In RStudio (after download, or hosted remotely)
Local install	Managing python install more complicated (use conda)	Usually straightforward
Format	JSON	Markdown
Version control	Improving (line-by-line diffs less nice, nbdime released)	Seamlessly integrated (line-by-line diffs, GUI in RStudio)
Environment	Python and packages on computer (local or remote), or per install with virtualenv	R and packages on computer (local or remote), or per project with packrat
Convert to output	Can download as pdf, markdown, HTML, etc or use nbconvert (requires installs)	Conversion to more types built in to RStudio Configure rendering with YAML header
Become a website	Yes, eg nbviwer, GitHub	Yes, “can publish to any static web host service”, eg RPubs, GitHub
Other languages	Over 100 kernels, including R, Or run R in python with r2py	R, python, bash, C++, SQL, can call functions from Julia, Fortran…
IDE	Notebook – no, Lab – yes, based on RStudio	Beautiful, Includes linting, integrated help
Render on GitHub	.ipynb – code and output	.Rmd – code only, not output .md + .png files – code, output, and images
Progress bar	Not by default (available as plugin)	Yes – shows progress when executing code

Demo: Notebooks in containers on cloud compute resources with Binder

1. Open one of these links to data from a famous gravity wave experiment:

   https://notebooks.gesis.org/ , alternate link

2. Click “Launch Binder”



3. Open index.ipynb
   - *Ctrl + enter to run cells of code*

https://mybinder.org/

already ~50,000 users/week (source)

Try it: https://mybinder.org/

More examples:
http://ivory.idyll.org/blog/tag/reproducibility.html

Demo: RStudio running on cloud compute resources (eg AWS, Jetstream)

RStudio in Binder demo:

- Link from recent publication

NSF-supported on-demand cloud computing environments
Indiana University / UT Austin

Demo above uses Rocker project :

- "provides a widely-used suite of Docker images with customized R environments for particular tasks"
- Over 3 Million installs!

image	description	size	metrics	build status
r-ver	Specify R version in docker tag. Builds on `debian:stable`	221.8MB 8 layers	docker pulls 23k	docker build automated
rstudio	Adds rstudio	0B 17 layers	docker pulls 1M	docker build automated

https://www.rocker-project.org/images/

An Introduction to Rocker: Docker Containers for R
Carl Boettiger, Dirk Eddelbuettel, 2017, arXiv:1710.03675

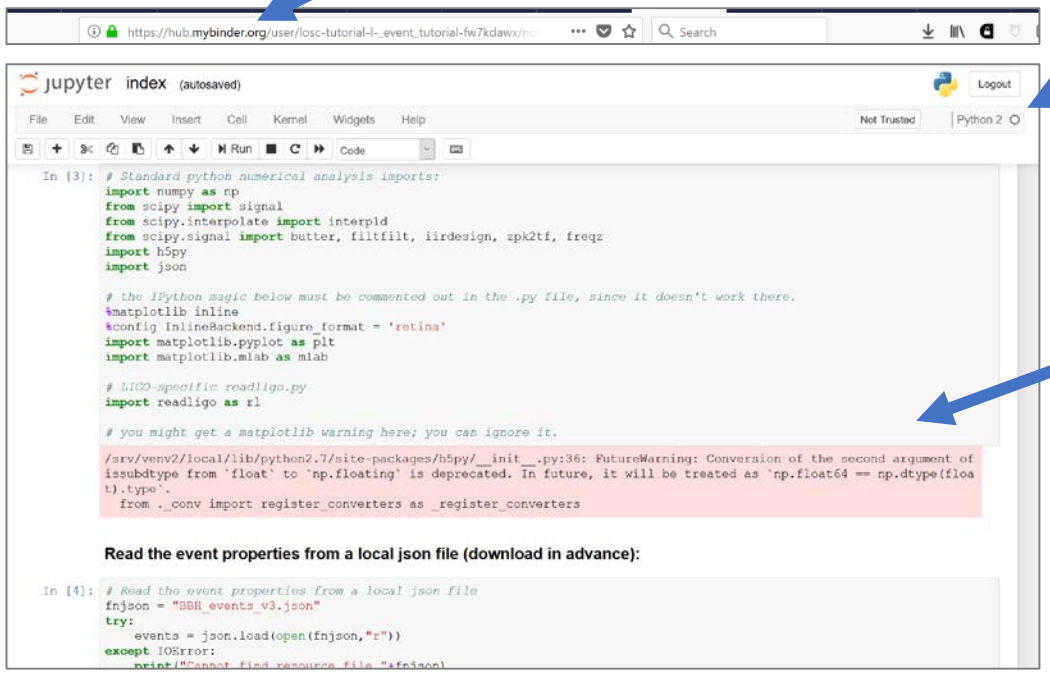
How does Binder work?

"Binder allows you to create custom computing environments that can be shared and used by many remote users"

https://mybinder.readthedocs.io/en/latest/

Visible in your browser,
but running on a remote machine

Your browser

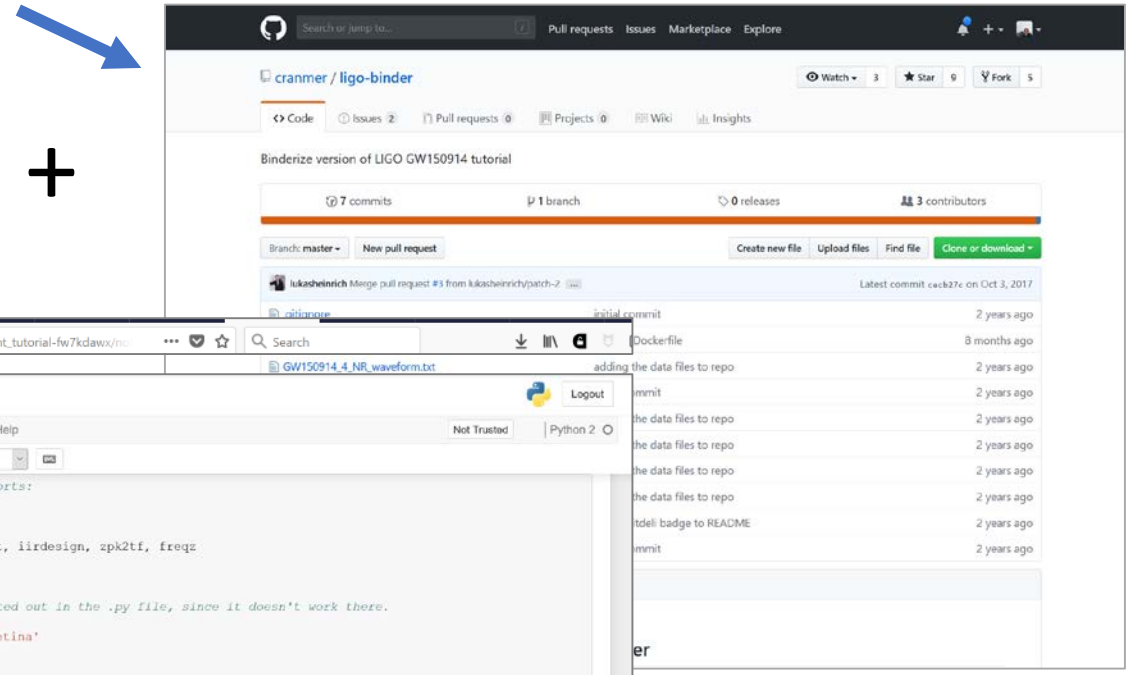


Specified version
of Python

You see the code in
the .ipynb file

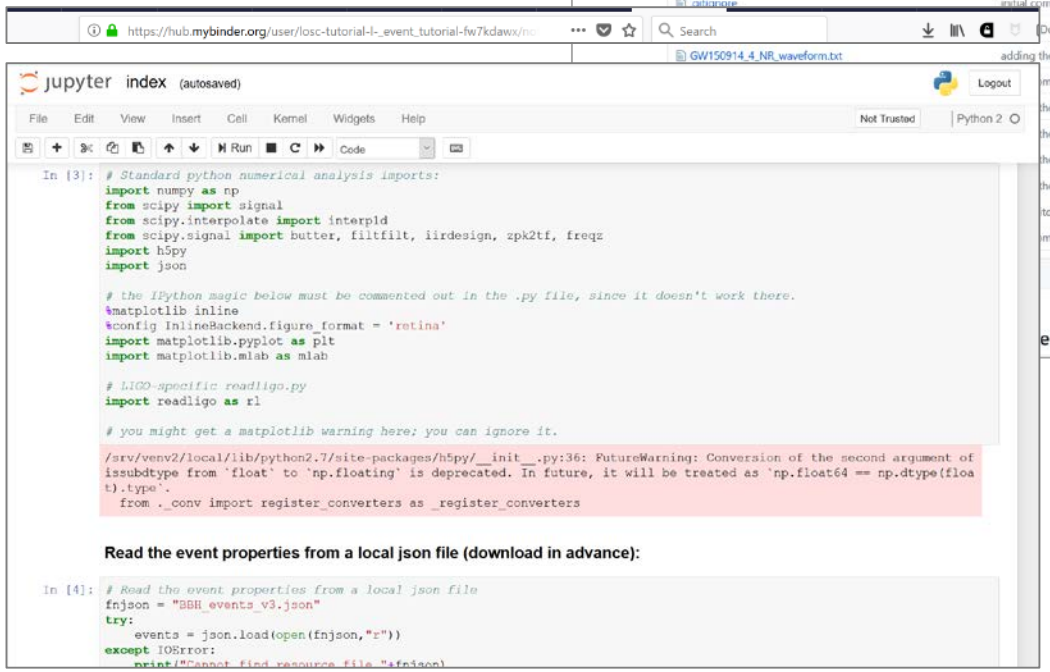
How does Binder work?

Code for notebook and list of dependencies and files available on GitHub (right now Binder only works with public repositories)



+

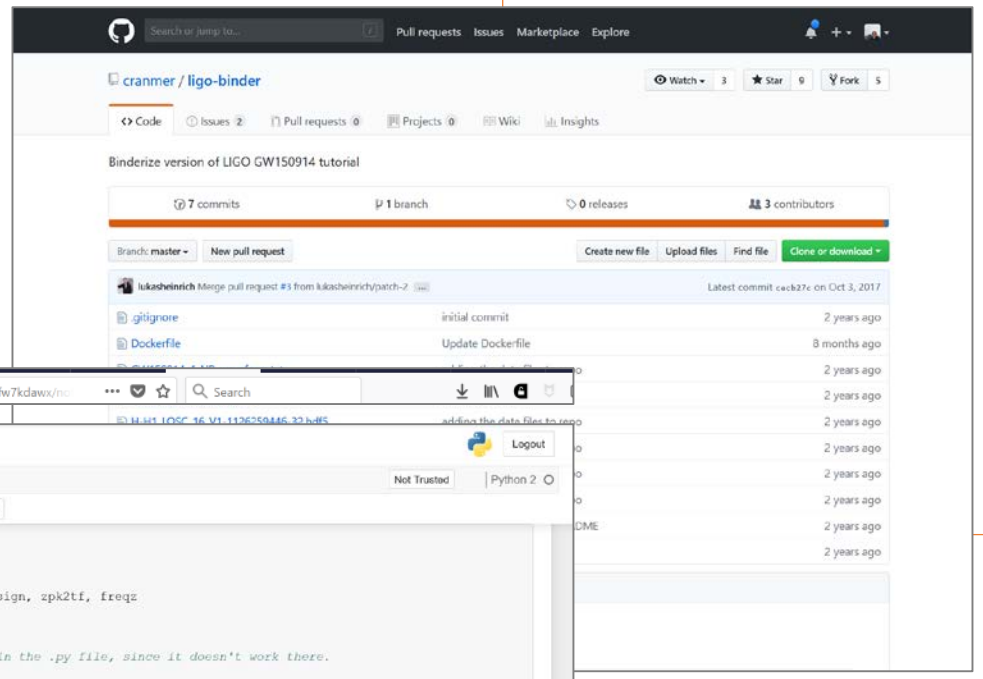
Your browser



How does Binder work?

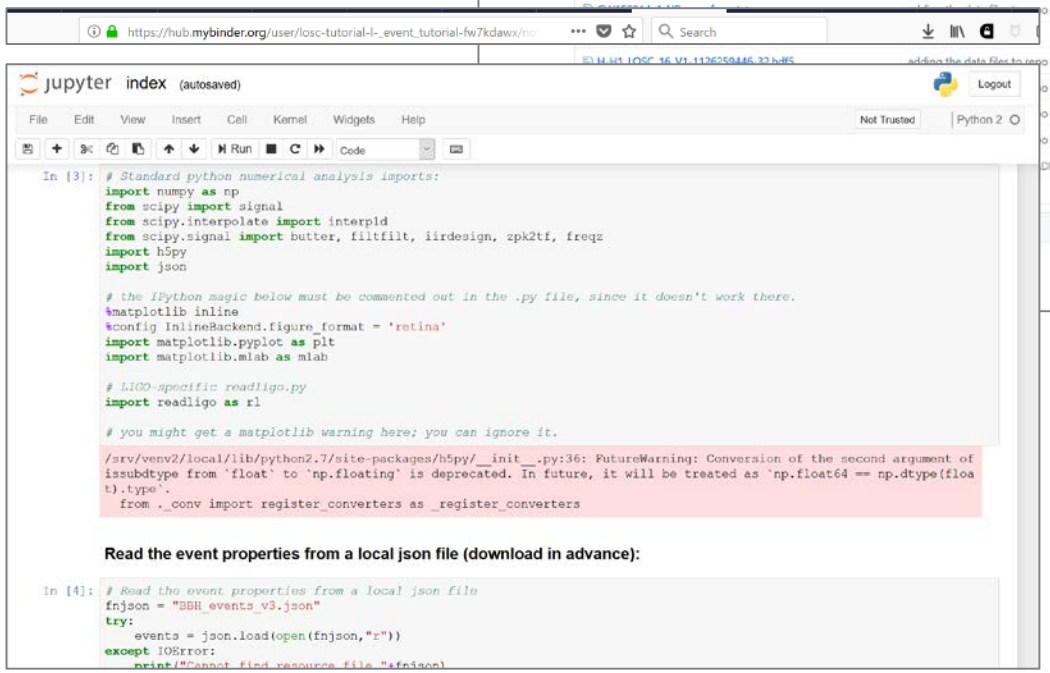
binder (beta) builds Docker image based on repo and generates URL for public access

Code and dependencies in public repository



+

Your browser



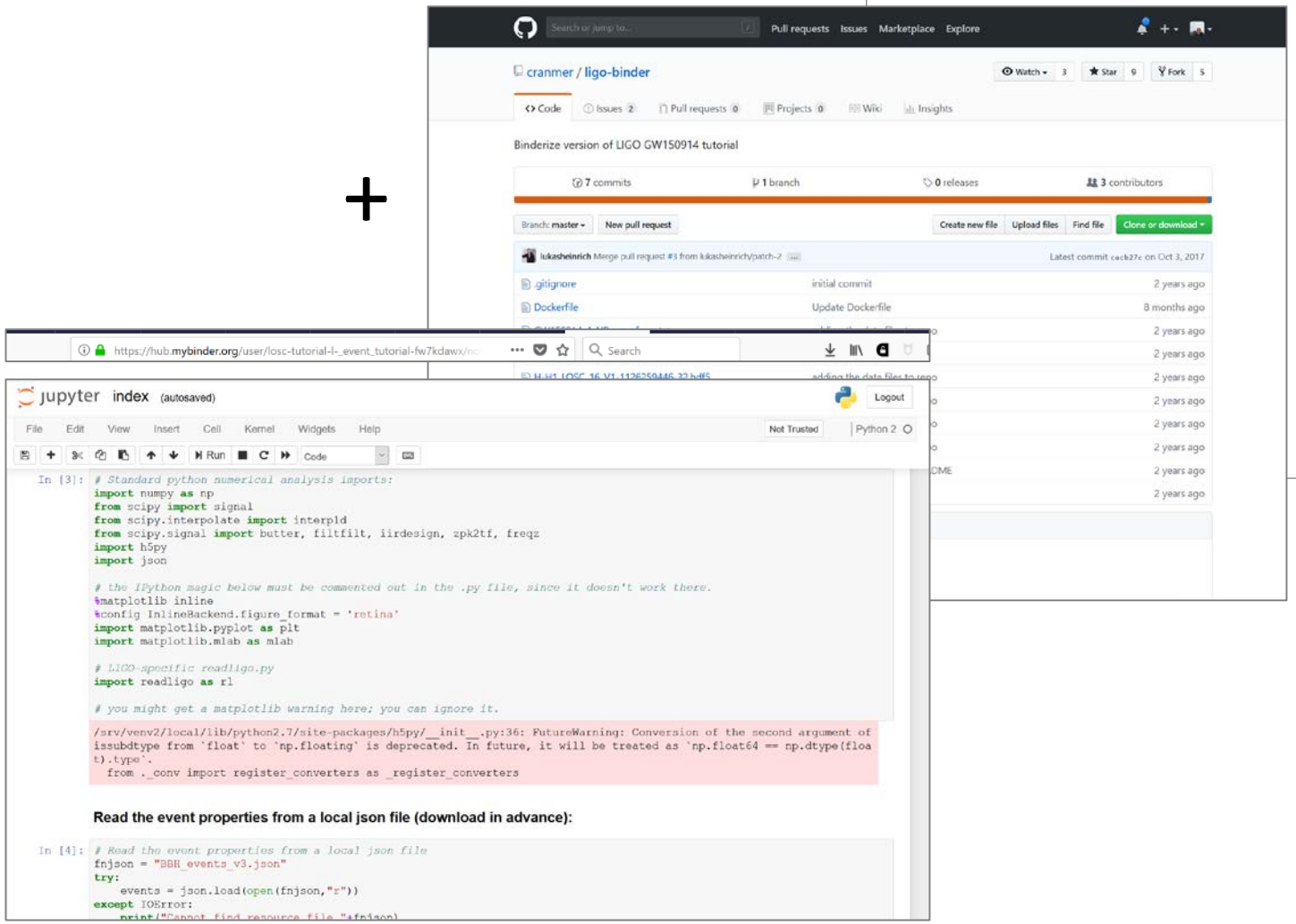
required software + packages

included data

How does Binder work?

+

Generates URL for temporary instance



+

+

required software + packages

included data

JupyterHub

+

Kubernetes

Generated binders are hosted by a hub that deploys, scales, and manages compute resources using Kubernetes

(can run on any cloud platform provider)
More details: BinderHub

Background: What are (software) containers?

- Commonly used metaphor of global shipping industry + standardized shipping containers

- **A container contains "code, configurations, and dependencies"** [3],
- like 'putting down a cloth over your table (Operating System) before installing anything'

- Enables multiple isolated sets of code, each of which can only access the files and resources allocated to it, all able to run on a shared operating system kernel

1- https://en.wikipedia.org/wiki/Operating-system-level_virtualization
2 - https://cloud.google.com/containers/
3 - https://aws.amazon.com/what-are-containers/

A few more resources…

- "Our path to better science in less time using open data science tools", 2017, https://www.nature.com/articles/s41559-017-0160

- "Developing a modern data workflow for living data" https://www.biorxiv.org/content/early/2018/06/12/344804

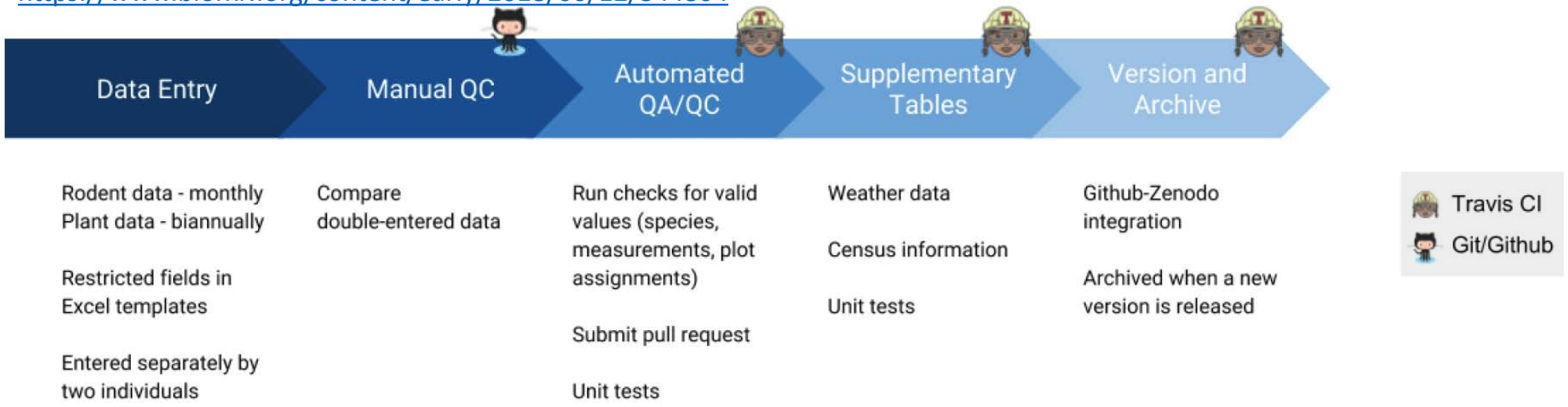


Figure 1: Our data workflow

- "Good enough practices in scientific computing" 2017, http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1005510
- "Ten simple rules for making research software more robust", 2017, http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1005412
- "Ten quick tips for teaching programming" 2018 http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1006023
- UO Psychology data science tutorials gallery https://robchavez.github.io/datascience_gallery/

Sidebar – who maintains or pays for all this software?

Key Python libraries for science + data analysis:

Screenshot from Prof. Kelle Cruz's (Hunter College/CUNY) Simons Foundations Lecture (link to slides)

https://www.numfocus.org/
https://www.simonsfoundation.org/event/collaborations-in-the-extreme-the-rise-of-open-code-development-in-the-scientific-community/

One model of support: an umbrella non-profit (mostly python)

NUMFOCUS

OPEN CODE = BETTER SCIENCE

- Funding
- Stability
- Independence
- Logistical support
  - Fiscal + Legal
  - Community

“From Netflix to NASA, researchers use our open source tools to solve the most challenging problems.”

https://www.numfocus.org/

Another model: RStudio employs >25 software engineers/developers

Shiny, Rpubs

Joe Cheng • 3rd
CTO at RStudio, Inc.
Greater Seattle Area

Tidyverse, R for Data Science

Hadley Wickham • 3rd
Chief Scientist at RStudio, Inc.
Houston, Texas Area

knitr, bookdown, blogdown

Yihui Xie • 3rd
Software Engineer at RStudio, Inc.
Greater Omaha Area

R Graphics Cookbook

Winston Chang • 3rd
Software Developer at RStudio
Greater Minneapolis-St. Paul Area

RMarkdown, R + Tensorflow, reticulate (python in R)

JJ Allaire • 3rd
CEO at RStudio
Greater Boston Area

Jenny Bryan

STAT545 resources, readxl

R for Data Science co-author

Garrett Grolemund •
Data Scientist and Mast
Greater Nashville Area, TN

StationaRy

Richard Iannone • 2nd
Software Engineer at RStudio, Inc.

Sparklyr (Spark in R)

Javier Luraschi
javierluraschi

Source: LinkedIn as of May 2018

jupyter has been supported by grants, employers, and donations

Jupyter steering council, 2018

Damian Avila

Anaconda, Inc.
@damianavila on GitHub

Matthias Bussonnier

UC Berkeley
@carreau on GitHub

Sylvain Corlay

QuantStack
@sylvaincorlay on GitHub

Brian Granger

Cal Poly, San Luis Obispo
@ellisonbg on GitHub

Jason Grout

Bloomberg
@jasongrout on GitHub

Jessica Hamrick

DeepMind
@jhamrick on GitHub

Paul Ivanov

Bloomberg
@ivanov on GitHub

Kyle Kelley

Netflix
@rgbkrk on GitHub

Thomas Kluyver

University of Southampton
@takluyver on GitHub

Peter Parente

Valassis Digital
@parente on GitHub

Fernando Perez

UC Berkeley
@fperez on GitHub

Min Ragan-Kelley

Simula Research Lab
@minrk on GitHub

Ana Ruvalcaba

Cal Poly, San Luis Obispo
@ruv7 on GitHub

Steven Silvester

JPMorgan Chase
@blink1073 on GitHub

Carol Willing

Cal Poly
@willingc on GitHub

Sponsors

Project Jupyter receives direct funding from the following sources:

THE LEONA M. AND HARRY B. HELMSLEY CHARITABLE TRUST

ALFRED P. SLOAN FOUNDATION

GORDON AND BETTY MOORE FOUNDATION

Google

rackspace the #1 managed cloud company

fastly

Horizon 2020 European Union funding for Research & Innovation

Microsoft

Institutional Partners

Institutional Partners are organizations that support the project by employing Jupyter Steering Council members. Current Institutional Partners include:

ANACONDA

Bloomberg

NETFLIX

CAL POLY SAN LUIS OBISPO

Berkeley UNIVERSITY OF CALIFORNIA

QuantStack Scientific Computing

JPMORGAN CHASE & CO.

Donations

Jupyter will always be 100% open source software, free for all to use and released under the liberal terms of the modified BSD license. If you have found Project Jupyter to be useful in your work, research or company, please consider making a donation to the project commensurate with your resources.

All donations will be used strictly to fund the development of Project Jupyter's open source software, documentation and community. Our donations are managed by the NumFOCUS Foundation, which is the legal and fiscal umbrella for the project. NumFOCUS is a 501(c)3 non-profit foundation; if you are subject to US Tax law, your contributions will be tax-deductible.

NUMFOCUS
OPEN CODE = BETTER SCIENCE

Support Project Jupyter

Where do we go from here?

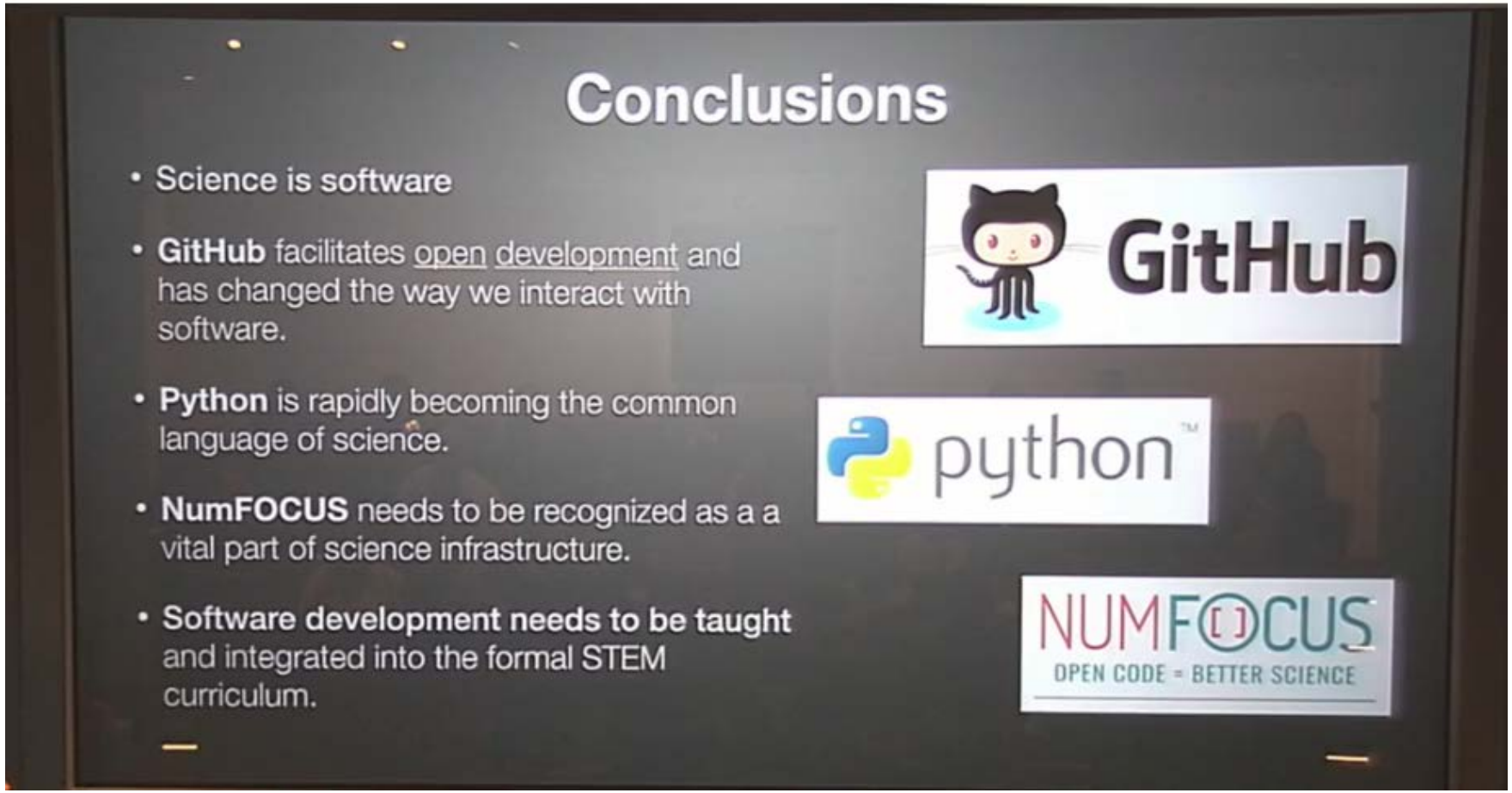
ROpenSci's perspective:

- **Train students** by putting homework, assignments & dissertations on the reproducible research spectrum
- **Publish examples** of reproducible research in our field
- **Request code & data** when reviewing
- **Submit to & review for journals** that support reproducible research
- **Critically review & audit** data management plans in grant proposals
- Consider reproducibility wherever possible in **hiring, promotion & reference letters**.

http://ropensci.github.io/reproducibility-guide/sections/introduction/ - Leveque et al

Where do we go from here?

Prof. Kelle Cruz's perspective

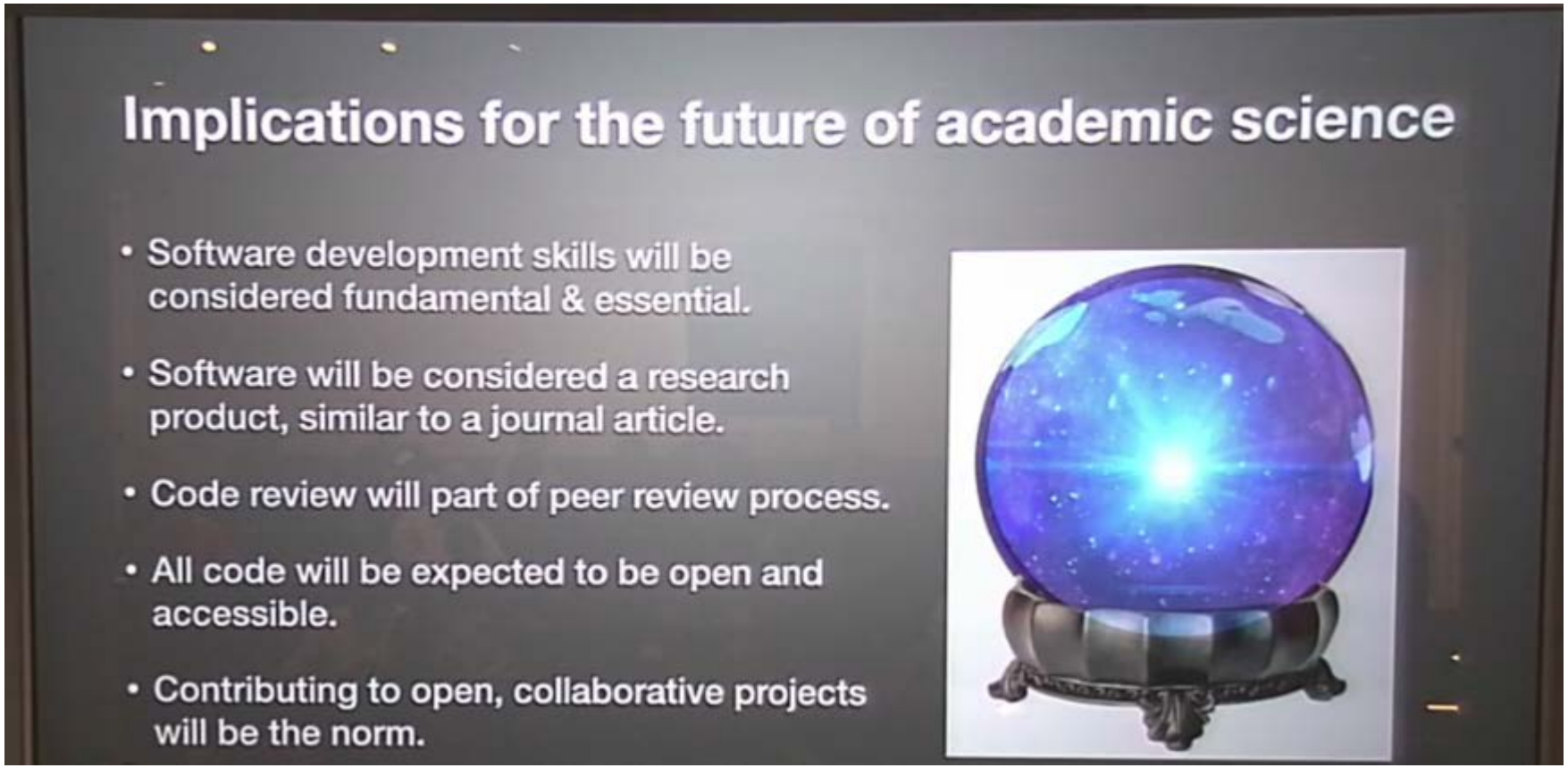


Screenshot from Prof. Kelle Cruz's (Hunter College/CUNY) Simons Foundations Lecture (link to slides)

https://www.numfocus.org/
https://www.simonsfoundation.org/event/collaborations-in-the-extreme-the-rise-of-open-code-development-in-the-scientific-community/

Where do we go from here?

Prof. Kelle Cruz's perspective



Screenshot from Prof. Kelle Cruz's (Hunter College/CUNY) Simons Foundations Lecture (link to slides)

https://www.numfocus.org/
https://www.simonsfoundation.org/event/collaborations-in-the-extreme-the-rise-of-open-code-development-in-the-scientific-community/

Discuss: Where do we go from here?

Bonus slides!

(for reference)

Preprint: “Terminologies for reproducible research”, Barba 2018

https://arxiv.org/abs/1802.03311

“…authors either,

A—make no distinction between the words reproduce and replicate, or
B—use them distinctly.

If B, then they are commonly divided in two camps.

In a spectrum of concerns that starts at a minimum standard of
“same data + same methods=same results,” to
“new data and/or new methods in an independent study=same findings,”

group 1 calls the minimum standard reproduce, while group 2 calls it replicate…”

[spacing added for clarity]

Preprint: “Terminologies for reproducible research”, Barba 2018

https://arxiv.org/abs/1802.03311

Conclusions

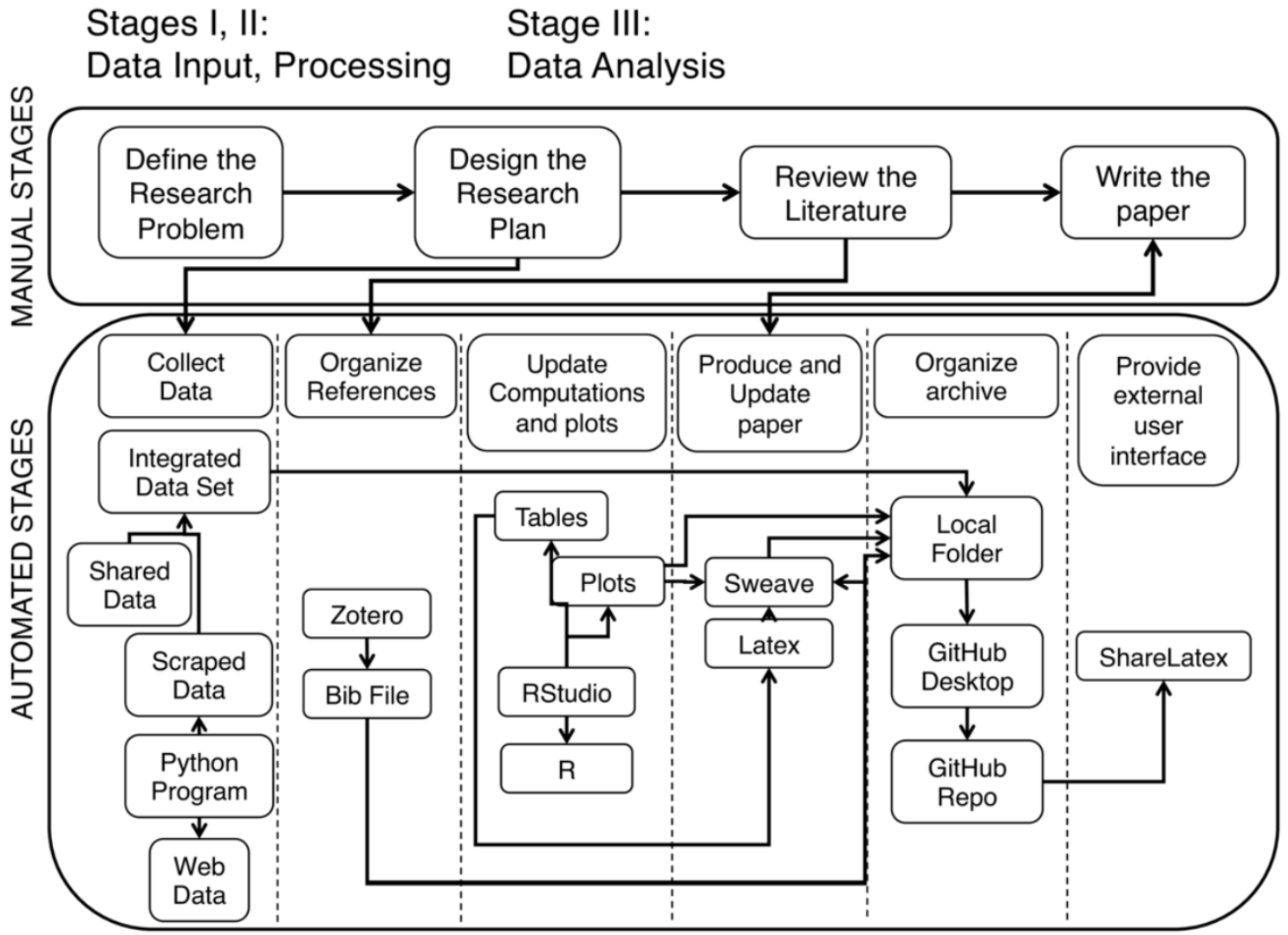
The Claerbout/Donoho/Peng terminology is broadly disseminated across disciplines (see Table 2). But the recent adoption of an opposing terminology by two large professional groups—ACM and FASEB—make standardization awkward. The ACM publicizes its rationale for adoption as based on the International Vocabulary of Metrology, but a close reading of the sources makes this justification tenuous. The source of the FASEB adoption is unclear, but there's a chance that Casadevall and Fang (2010) had an influence there. They, in turn, based their definitions on the emphatic but essentially flawed work of Drummond (2009).

Table 2: Grouping of terminologies, as in Table 1, but by discipline.

A	*B*1	*B*2
political science	signal processing	microbiology, immunology (FASEB)
economics	scientific computing	computer science (ACM)
	econometry	
	epidemiology	
	clinical studies	
	internal medicine	
	physiology (neuro)	
	computational biology	
	biomedical research	
	statistics	

Book: “The Practice of Reproducible Research”

https://www.practicereproducibleresearch.org/core-chapters/1-intro.html



From one case study by José Manuel Magallanes

https://www.practicereproducibleresearch.org/case-studies/jmMagallanes.html

“Identifying and Overcoming Threats to Reproducibility, Replicability, Robustness, and Generalizability in Microbiome Research”, Schloss 2018

Exercise 2. Imagine that a graduate student is really excited about an analysis that you performed in your most recent paper and would like to replicate the analysis with their own data. But first, they want to make sure that they reproduce your results. What steps are likely to cause the student problems?

If it is not clear to you what problems they might face, find your favorite figure from a paper by a different research group than your own. Can you reproduce the figure? What is standing in your way?

Exercise 5. Many of the threats to reproducibility and replicability are a product of scientific culture: methods sections are terse or vague, original data are not available, analyses rely on expensive and proprietary software, analysis scripts are available “upon request from the authors,” and papers are published behind paywalls. Some might give into despair thinking that one person or research group can have only a minor impact.

Have a discussion within your group about why things are this way, whether your group’s practices should change, and what would be the easiest and most impactful thing to change.

http://mbio.asm.org/content/9/3/e00525-18.full?sid=29656753-e64c-40bc-8590-f031dfcdd05a

TABLE 2 An aspirational rubric for evaluating the practices that host-associated microbiome researchers might use to increase the reproducibility and replicability of their work[a]

Practice	Good	Better	Best
Handling of confounding variables	Prior to generating data, did we identify a list of possible confounding variables—biological and technical—that may obscure the interpretation of our results?	Do we indicate the level of randomization and experimental blocking that we performed to minimize the effect of the confounding variables?	Does the interpretation of our results limit itself to only those variables that are not obviously confounded?
Sex/gender as confounding variables	Do we indicate the sex/gender of research animals/participants?	Do we provide a justification for the lack of even representation?	Is there equitable representation of sexes/genders? Do we account for them as a variable?
Experimental design considerations	Do we have an active collaboration with a statistician who helps with experimental design and analysis?	Do we indicate the number of hypothesis tests that we performed and have we corrected any *P* values for multiple comparisons?	For our primary research questions, have we run a power analysis to determine the necessary sample size?
Data analysis plan	Before starting an analysis, have we articulated a set of primary and secondary research questions?	Has someone else reviewed our data analysis plan prior to analyzing the data?	Have we registered our data analysis plan with a third party before starting the project?
Provenance of reagents	Is there a table of reagents such as cell lines, strains, and primer sequences that were used?	Where possible, have we obtained reagents from certified entities like the American Type Culture Collection (ATCC)?	Is there a statement indicating how we know the provenance and purity of each cell line and strain?
Controlling for initial microbiota	Are mice obtained from a breeding facility that allows us to track their pedigree?	Where possible, are mice from different treatment groups cohoused to control for differences in initial microbiota?	Are comparisons between mice with different genotypes made using mice that are the result of matings between animals that are heterozygous for that genotype?
Clarity of software descriptions	Are all methods, databases, and software tools cited? Do we follow the relevant licensing requirements of each tool?	Do we indicate dates and version numbers of websites that were used to obtain data, code, and other third-party resources?	Are detailed methods registered on a website like protocols.io or GitHub?
DNA contamination	Did we quantify the background DNA concentration in our reagents? Did we sequence an extraction control?	Are we taking steps to minimize reagent contamination?	What methods do we take to confirm a result that a sequencing result may be clouded by contaminating DNA?
Availability of data products	Are all of the raw data publicly available?	Are intermediate and final data files publicly available?	Are tools like Amazon Machine Images (AMIs) used to make a snapshot of our working directory?
Availability of metadata	Are all of the metadata necessary to repeat any analyses that we performed publicly available?	Have we adhered to standards in releasing the minimum amount of metadata about our samples?	Did we go beyond the minimum to incorporate other pieces of metadata that will inform future studies?
Data analysis organization	Are all data, code, results, and documentation housed within a monophyletic folder structure on our computer?	Is this project contained within a single directory on our computer, and does it separate our raw and processed data, code, documentation, and results?	Is this folder structure under version control? Is the project's repository publicly available? Are there assurances that this repository will remain accessible?

Schloss 2018 - http://mbio.asm.org/content/9/3/e00525-18/T2.expansion.html

TABLE 2 An aspirational rubric for evaluating the practices that host-associated microbiome researchers might use to increase the reproducibility and replicability of their work[a]

TABLE 2 (Continued)

Practice	Good	Better	Best
Use of random number generator	Do we know whether any of the steps in our data analysis workflow depend on the use of a random number generator?	For analyses that utilize a random number generator, have we noted the underlying random seed?	Have we repeated our analysis with multiple seeds to show that the results are insensitive to the choice of the seed?
Defensive data analysis	Is our data analysis pipeline flexible enough to add new data?	Does our code include tests to confirm that it does what we think it does?	Did we make use of automated tests and continuous integration tools to ensure internal reproducibility?
Ensuring short- and long-term reproducibility	Did we release the underlying code and new data at the time of submitting a paper with their DOIs and accession numbers?	Did we include a reproducibility statement or declaration at the end of the manuscript? Are ORCID identifiers provided for all authors?	What mechanisms are in place to ensure that our analysis remains accessible and reproducible in 5 years?
Open science to foster reproducibility	Have we released any embargoes on our code repository and raw data prior to submitting the manuscript?	Did we post a preprint version of our manuscript prior to submission?	Have we published under a Creative Commons license? Is a permissive reuse license posted with our code?
Transparency of data analysis	Is it clear where one would go to find the data and processing steps behind any of our figures?	Are electronic notebooks publicly accessible, and do they accompany the manuscript?	Were literate programming tools used to generate summary statistics, tables, and figures?

[a]Although many of the questions can be thought of as having a yes-or-no answer, a better approach would be to see the questions as being open ended with the real question being "What can we do to improve the status of our project on this point?" With this in mind, a researcher is unlikely to have a project that satisfies the "Best" column for each line of the table. Researchers are encouraged to adapt and modify the categories to suit their own needs.

http://mbio.asm.org/content/9/3/e00525-18.full?sid=29656753-e64c-40bc-8590-f031dfcdd05a

Background: R Notebook vs R Markdown documents

A notebook, `html_notebook`, is a variation on a `html_document`. The rendered outputs are very similar, but the purpose is different. A `html_document` is focussed on communicating with decision makers, while a notebook is focussed on collaborating with other data scientists. These different purposes lead to using the HTML output in different ways. Both HTML outputs will contain the fully rendered output, but the notebook also contains the full source code. That means you can use the `.nb.html` generated by the notebook in two ways:

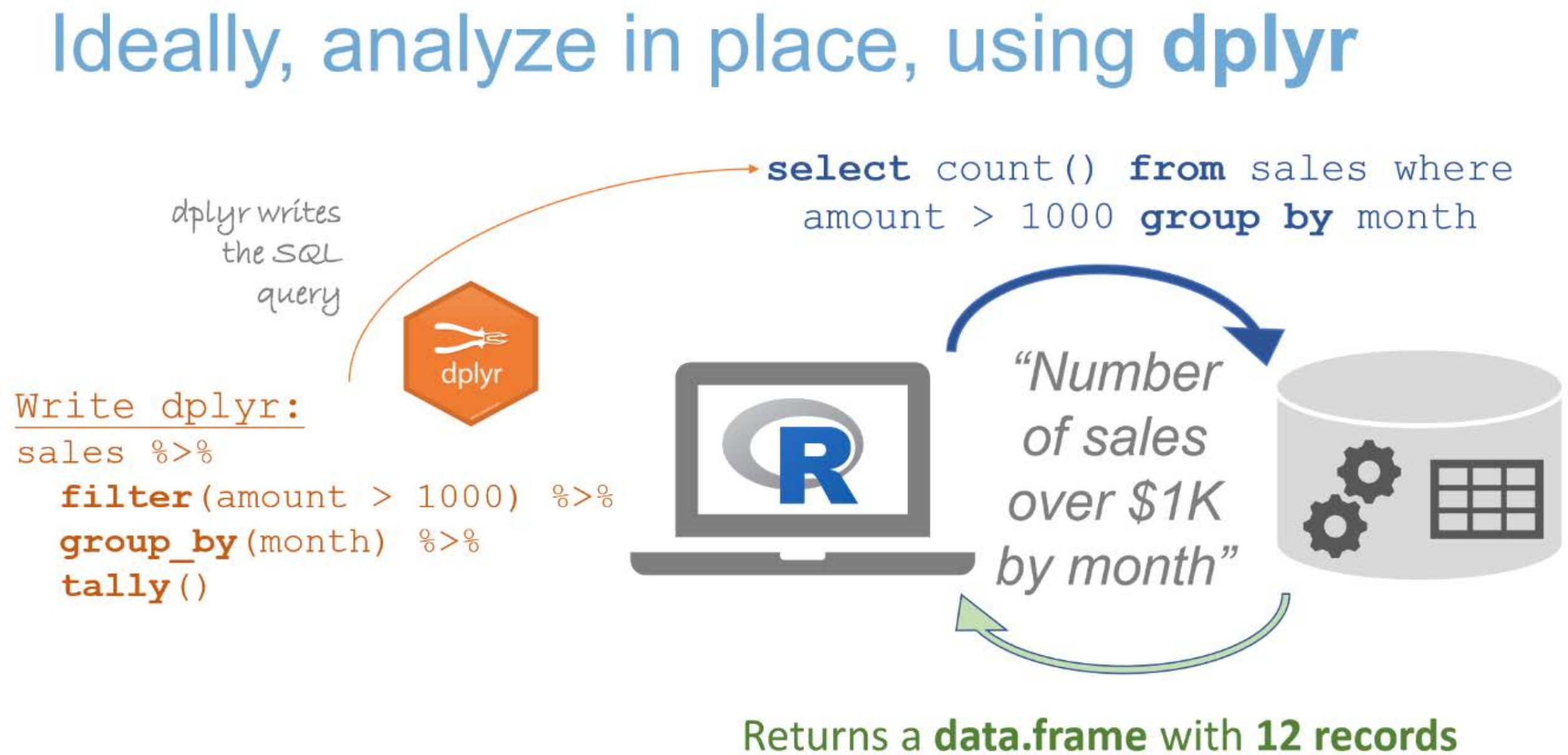
1. You can view it in a web browser, and see the rendered output. Unlike `html_document`, this rendering always includes an embedded copy of the source code that generated it.

2. You can edit it in RStudio. When you open an `.nb.html` file, RStudio will automatically recreate `.Rmd` file that generated it. In the future, you will also be able include supporting files (e.g. `.csv` data files), which will be automatically extracted when needed.

Emailing `.nb.html` files is a simple way to share analyses with your colleagues. But things will get painful as soon as they want to make changes. If this starts to happen, it's a good time to learn Git and GitHub. Learning Git and GitHub is definitely painful at first, but the collaboration payoff is huge. As mentioned earlier, Git and GitHub are outside the scope of the book, but there's one tip that's useful if you're already using them: use both `html_notebook` and `github_document` outputs:

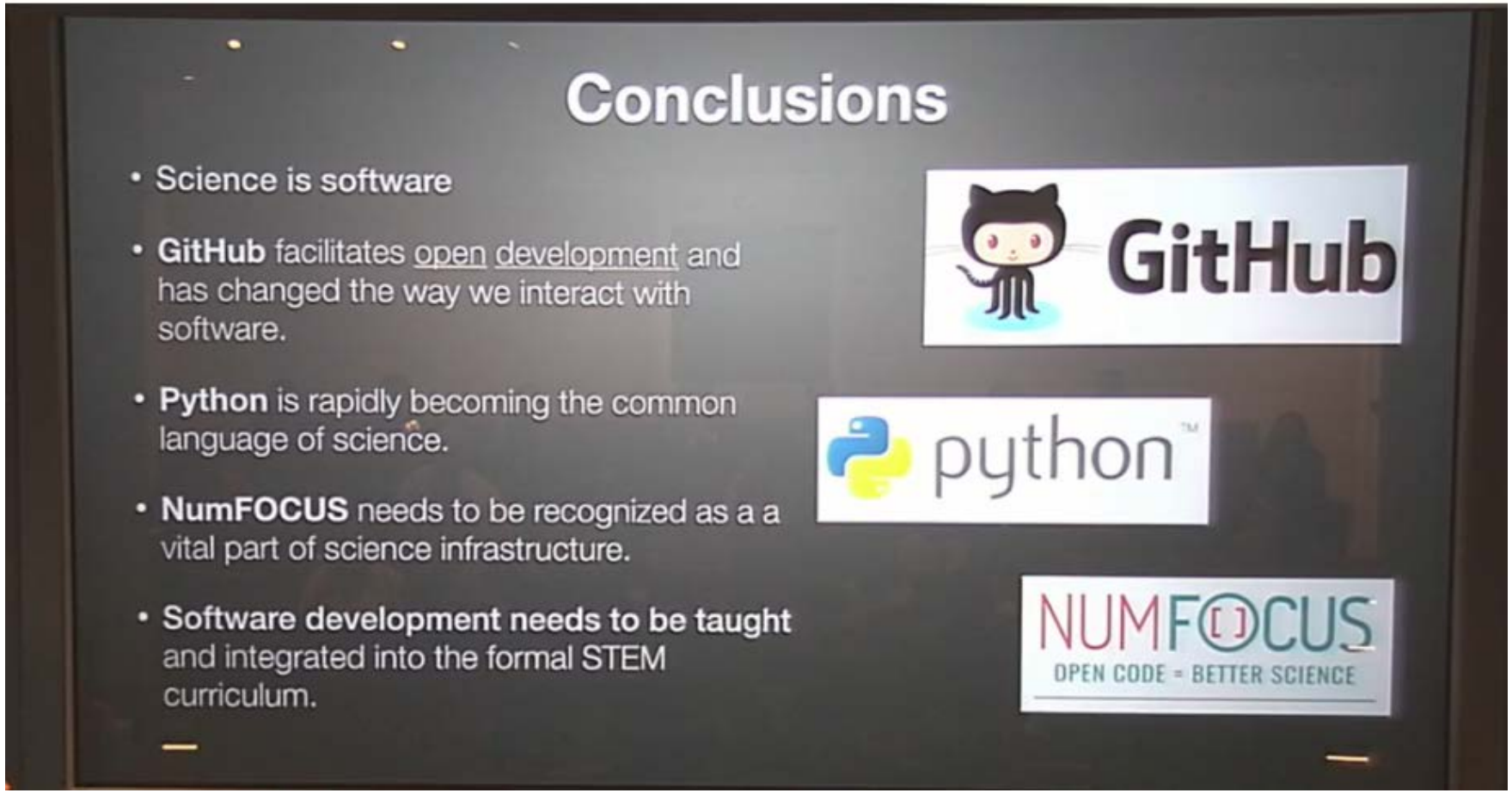
http://r4ds.had.co.nz/r-markdown-formats.html

Background: Spark in R



Where do we go from here?

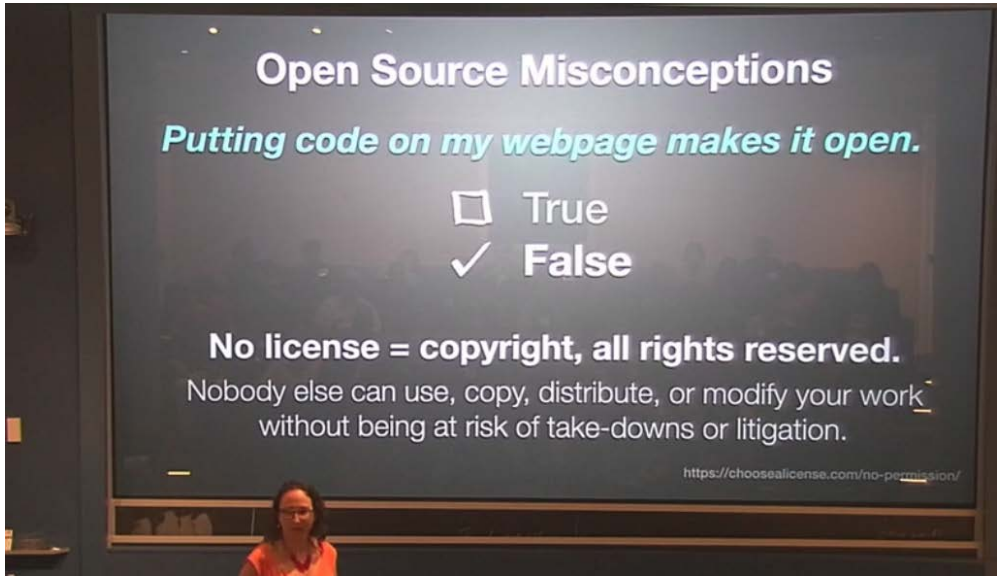
Prof. Kelle Cruz's perspective

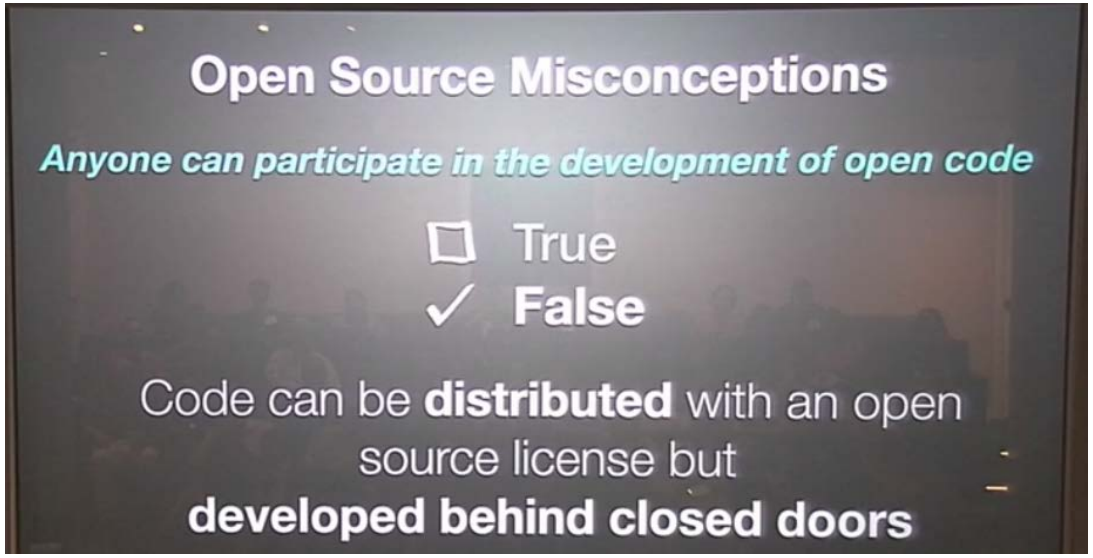


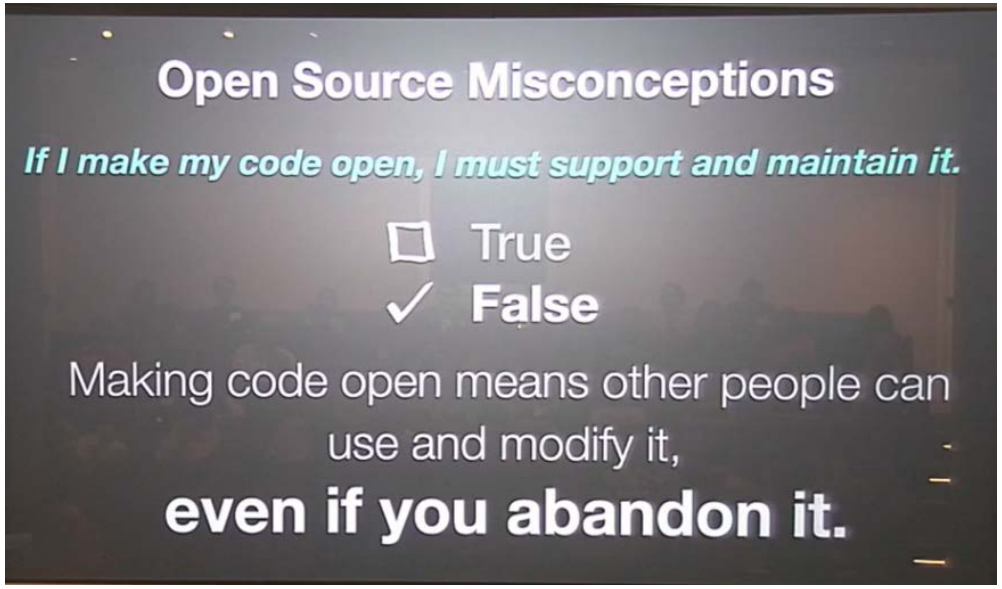
Screenshot from Prof. Kelle Cruz's (Hunter College/CUNY) Simons Foundations Lecture (link to slides)

https://www.numfocus.org/
https://www.simonsfoundation.org/event/collaborations-in-the-extreme-the-rise-of-open-code-development-in-the-scientific-community/

Open source misconceptions





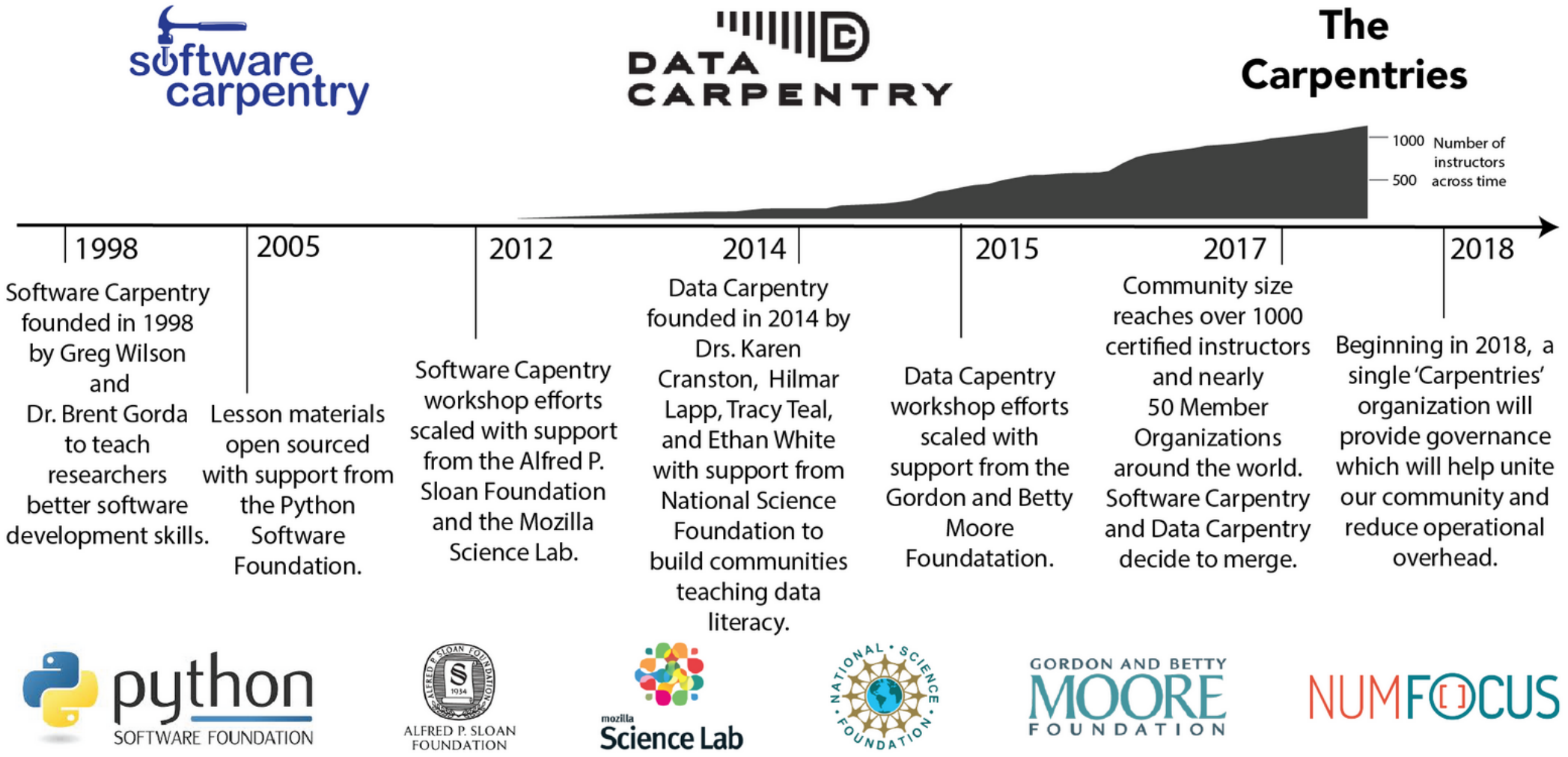


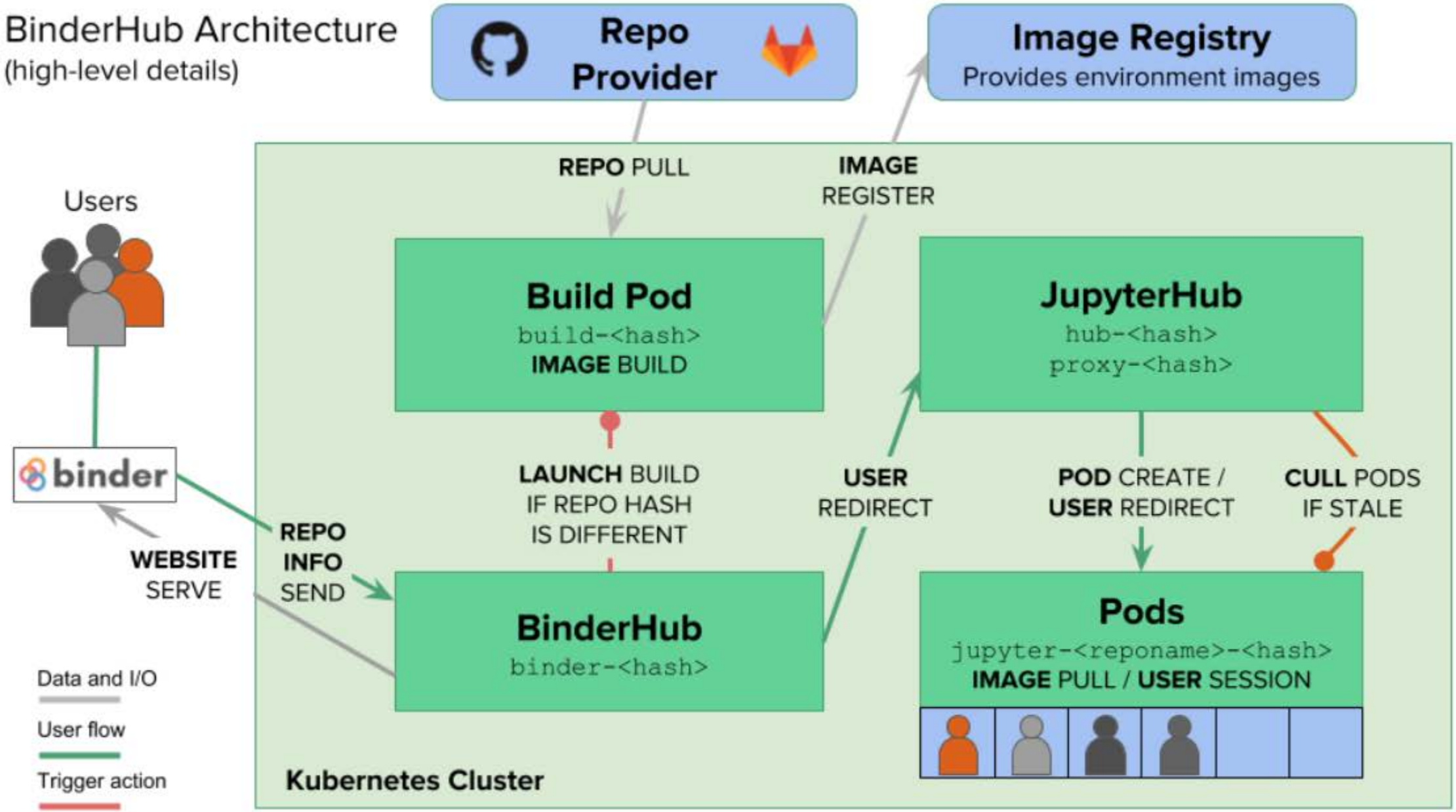
Screenshots from
Prof. Kelle Cruz's (Hunter College/CUNY)
Simons Foundations Lecture
(link to slides)

https://www.simonsfoundation.org/event/collaborations-in-the-extreme-the-rise-of-open-code-development-in-the-scientific-com

History of Software Carpentry (now: The Carpentries)

Check out upcoming workshops: https://carpentries.org





http://binderhub.readthedocs.io/en/latest/overview.html