Toward Reproducible Data Science

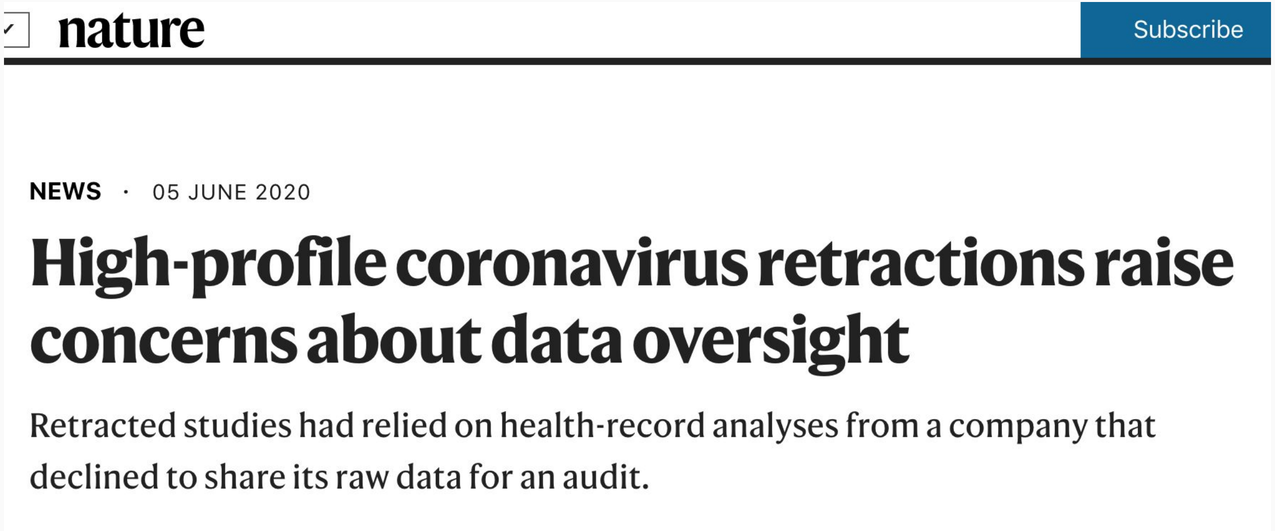
Tips and tricks of making data science projects more reproducible.

Valentina Staneva

Senior Data Scientist, eScience Institute

UNIVERSITY *of* WASHINGTON

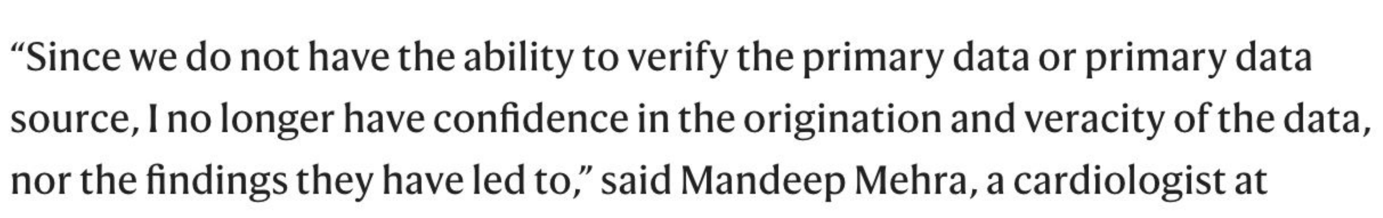
Reliable data science studies?



NEWS · 05 JUNE 2020

High-profile coronavirus retractions raise concerns about data oversight

Retracted studies had relied on health-record analyses from a company that declined to share its raw data for an audit.

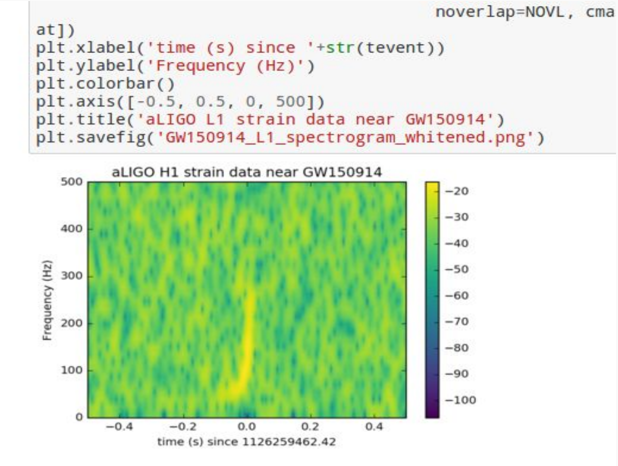


“Since we do not have the ability to verify the primary data or primary data source, I no longer have confidence in the origination and veracity of the data, nor the findings they have led to,” said Mandeep Mehra, a cardiologist at

Many more retractions!

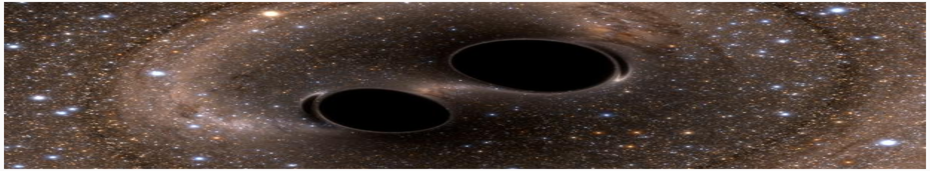
LIGO experiment

Jupyter Notebooks analyzing the data:

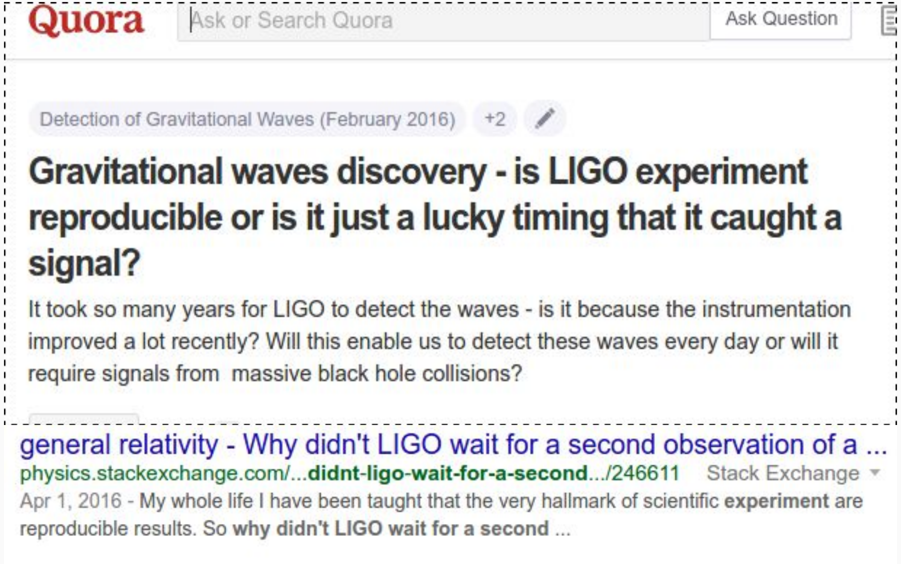


```
                                        noverlap=NOVL, cma
at])
plt.xlabel('time (s) since '+str(tevent))
plt.ylabel('Frequency (Hz)')
plt.colorbar()
plt.axis([-0.5, 0.5, 0, 500])
plt.title('aLIGO L1 strain data near GW150914')
plt.savefig('GW150914_L1_spectrogram_whitened.png')
```

https://losc.ligo.org/s/events/GW150914/GW150914_tutorial.html



Is the experiment reproducible?



Detection of Gravitational Waves (February 2016) +2

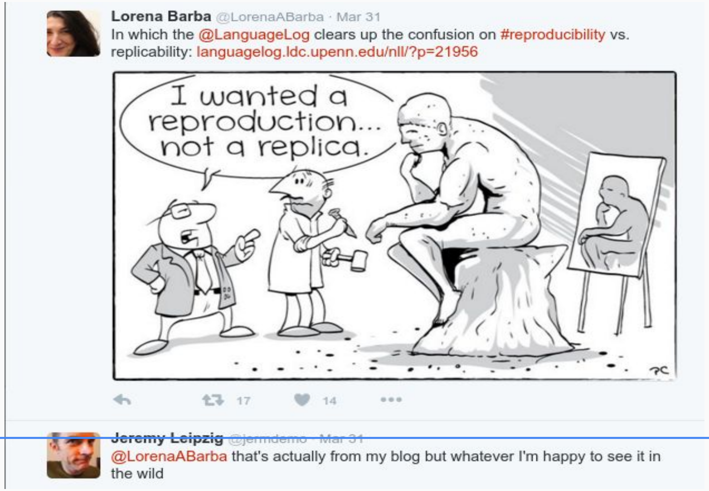
Gravitational waves discovery - is LIGO experiment reproducible or is it just a lucky timing that it caught a signal?

It took so many years for LIGO to detect the waves - is it because the instrumentation improved a lot recently? Will this enable us to detect these waves every day or will it require signals from massive black hole collisions?

general relativity - Why didn't LIGO wait for a second observation of a ...
physics.stackexchange.com/...**didnt-ligo-wait-for-a-second**.../246611 Stack Exchange
Apr 1, 2016 - My whole life I have been taught that the very hallmark of scientific **experiment** are reproducible results. So **why didn't LIGO wait for a second** ...

Second Gravitational Wave Detected!

Reproducibility vs Replicability



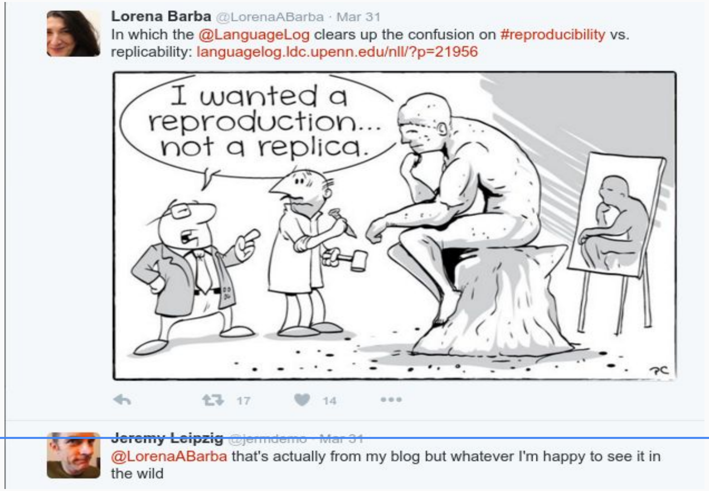
Lorena Barba @LorenaABarba · Mar 31
In which the @LanguageLog clears up the confusion on #reproducibility vs. replicability: languagelog.ldc.upenn.edu/nll/?p=21956

Jeremy Leipzig @jermdemo · Mar 31
@LorenaABarba that's actually from my blog but whatever I'm happy to see it in the wild

Two main notions:

- Results of an experiment are regenerated using the same data and methods.
- Results of an experiment are regenerated using new data or alternative methods.

Reproducibility vs Replicability



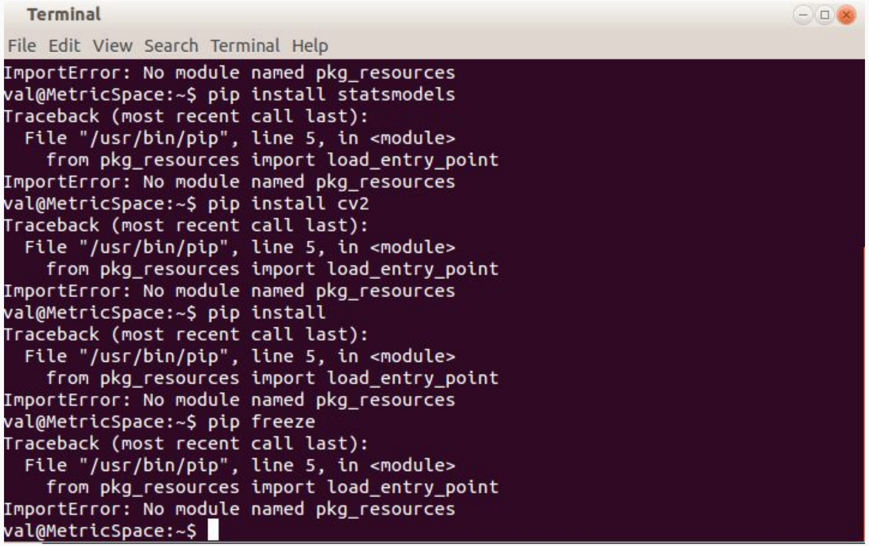
Lorena Barba @LorenaABarba · Mar 31
In which the @LanguageLog clears up the confusion on #reproducibility vs. replicability: languagelog.ldc.upenn.edu/nll/?p=21956

Jeremy Leipzig @jermdemo · Mar 31
@LorenaABarba that's actually from my blog but whatever I'm happy to see it in the wild

Two main notions:

- Results of an experiment are regenerated using the same data and methods.
- Results of an experiment are regenerated using new data or alternative methods.

It is hard...

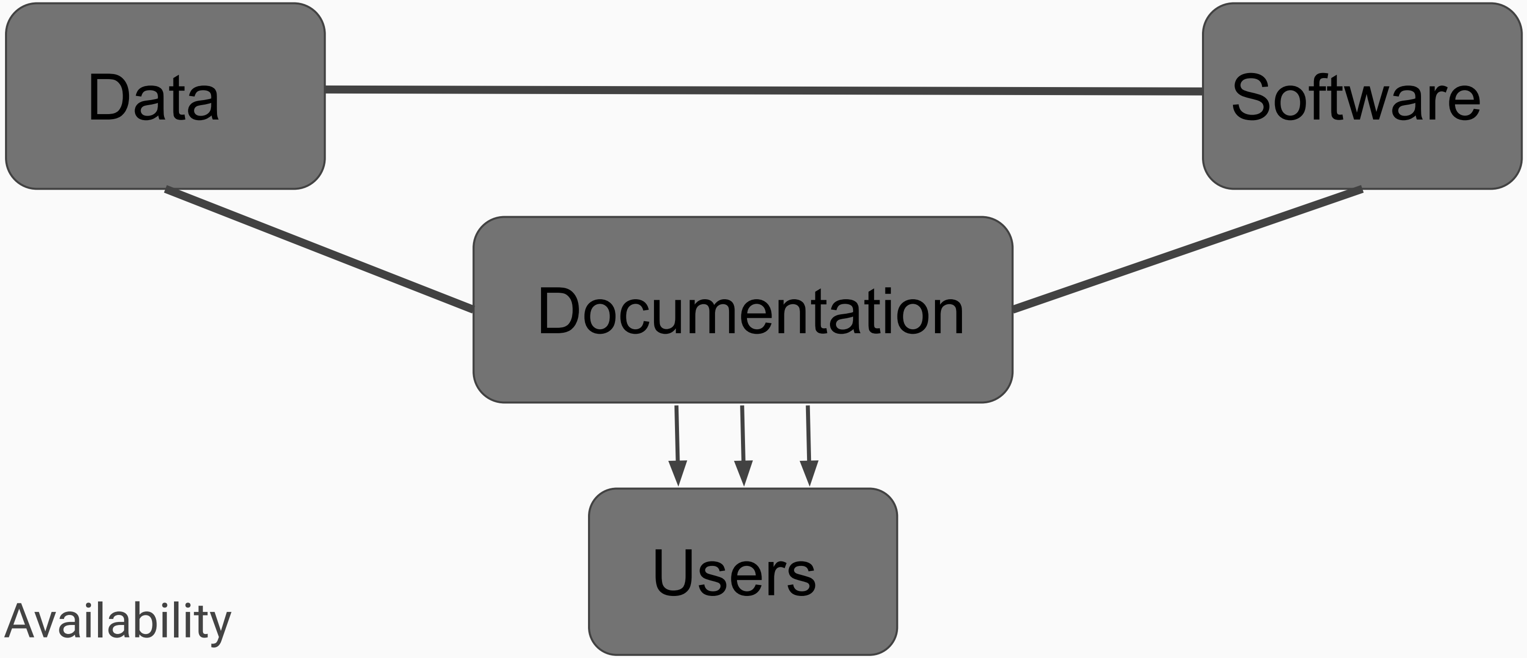


```
ImportError: No module named pkg_resources
val@MetricSpace:~$ pip install statsmodels
Traceback (most recent call last):
  File "/usr/bin/pip", line 5, in <module>
    from pkg_resources import load_entry_point
ImportError: No module named pkg_resources
val@MetricSpace:~$ pip install cv2
Traceback (most recent call last):
  File "/usr/bin/pip", line 5, in <module>
    from pkg_resources import load_entry_point
ImportError: No module named pkg_resources
val@MetricSpace:~$ pip install
Traceback (most recent call last):
  File "/usr/bin/pip", line 5, in <module>
    from pkg_resources import load_entry_point
ImportError: No module named pkg_resources
val@MetricSpace:~$ pip freeze
Traceback (most recent call last):
  File "/usr/bin/pip", line 5, in <module>
    from pkg_resources import load_entry_point
ImportError: No module named pkg_resources
val@MetricSpace:~$
```

It is not about reproducible or not reproducible.

It is about more reproducible.

Improving Reproducibility



- Availability
- Automation
- Sustainability

Tips for more reproducible data science.

So far you have learnt:

- Some programming/data analysis
- Code style and documentation
- Lots of version control!!
- Machine Learning
- Team building

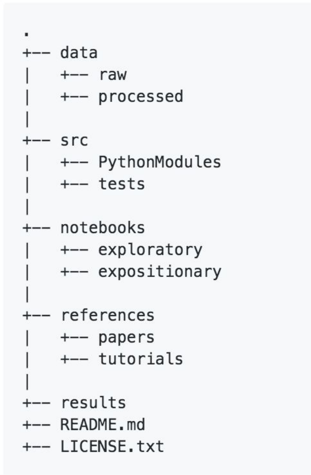
What we will discuss today:

- Project Organization
- Modular Programming
- Literate Programming
- Virtualization
- Testing
- Software Licensing
- Data Sharing

Project Repository Organization

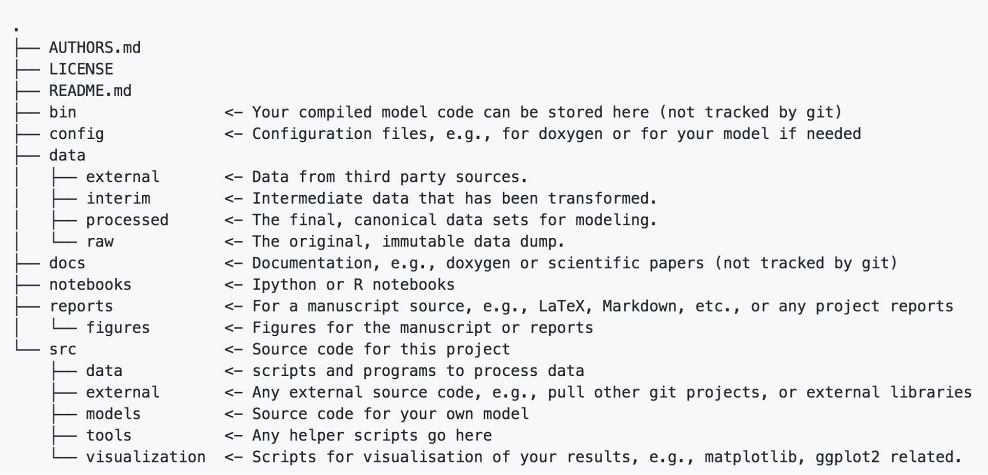
- R Project Structure: https://nicercode.github.io/blog/2013-04-05-projects/
- R Project Template: http://projecttemplate.net/getting_started.html
- Data Science Project Structure: Cookiecutter
- Python Module Template: Shablona

Start simple:



```
.
+-- data
|    +-- raw
|    +-- processed
|
+-- src
|    +-- PythonModules
|    +-- tests
|
+-- notebooks
|    +-- exploratory
|    +-- expositionary
|
+-- references
|    +-- papers
|    +-- tutorials
|
+-- results
+-- README.md
+-- LICENSE.txt
```

Expand as needed:



```
.
├── AUTHORS.md
├── LICENSE
├── README.md
├── bin                <- Your compiled model code can be stored here (not tracked by git)
├── config             <- Configuration files, e.g., for doxygen or for your model if needed
├── data
│   ├── external       <- Data from third party sources.
│   ├── interim        <- Intermediate data that has been transformed.
│   ├── processed      <- The final, canonical data sets for modeling.
│   └── raw            <- The original, immutable data dump.
├── docs               <- Documentation, e.g., doxygen or scientific papers (not tracked by git)
├── notebooks          <- Ipython or R notebooks
├── reports            <- For a manuscript source, e.g., LaTeX, Markdown, etc., or any project reports
│   └── figures        <- Figures for the manuscript or reports
└── src                <- Source code for this project
    ├── data           <- scripts and programs to process data
    ├── external       <- Any external source code, e.g., pull other git projects, or external libraries
    ├── models         <- Source code for your own model
    ├── tools          <- Any helper scripts go here
    └── visualization  <- Scripts for visualisation of your results, e.g., matplotlib, ggplot2 related.
```

Pick and adjust for your project!

Software License Selection

Code without a license is protected by the author's copyright law.

Choose a license: http://choosealicense.com/

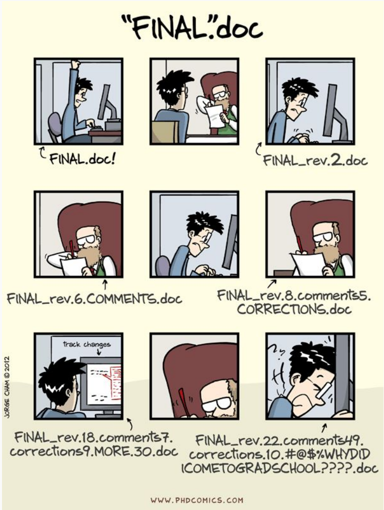
I need to work in a community.

I want it simple and permissive.

I care about sharing improvements.

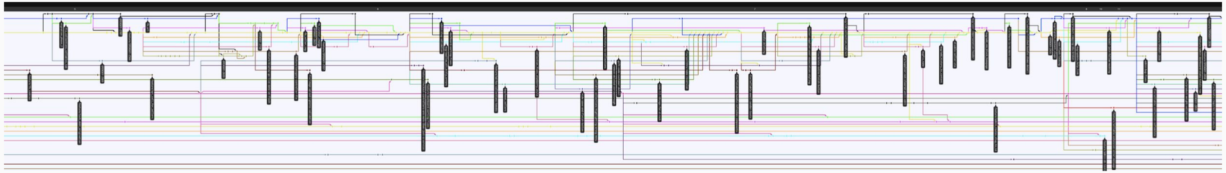
- Permissible licenses: MIT, BDS, Apache
  - With/without attribution, with/without explicit modification explanation
- Copyleft licenses: GPL
  - Forces all derivatives to have the same license
  - Viral licensing: for example GPL code may be hard to integrate with MIT code.
- Creative Commons (good for documents or educational materials)
  - with/without attribution, with/without derivatives, with/without commercial use
  - https://chooser-beta.creativecommons.org/

Version Control Workflow



- Version control for code: git & Github
  - Software Carpentry Tutorials: https://swcarpentry.github.io/git-novice/
  - Atlassian Tutorials: https://www.atlassian.com/git/tutorials/what-is-version-control
  - Cheetsheets: https://services.github.com/on-demand/downloads/github-git-cheat-sheet.pdf
- Version control for data:
  - Git Large File Storage
  - Quilt: https://quiltdata.com/
  - Data Version Control: https://dvc.org/ (for Machine Learning Projects)

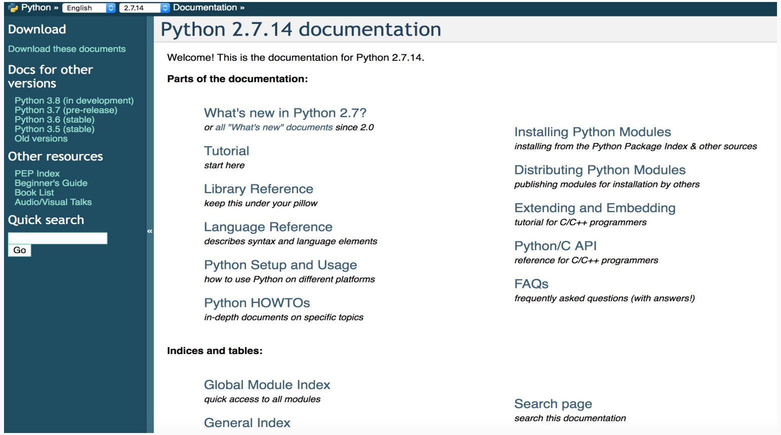
Decide on a strategy with your team!



http://justinhileman.info/article/changing-history/opensky-branches.png

Documentation

Python - Sphinx, Read the Docs



Python 2.7.14 documentation

Welcome! This is the documentation for Python 2.7.14.

Parts of the documentation:

What's new in Python 2.7?
or all "What's new" documents since 2.0

Tutorial
start here

Library Reference
keep this under your pillow

Language Reference
describes syntax and language elements

Python Setup and Usage
how to use Python on different platforms

Python HOWTOs
in-depth documents on specific topics

Installing Python Modules
installing from the Python Package Index & other sources

Distributing Python Modules
publishing modules for installation by others

Extending and Embedding
tutorial for C/C++ programmers

Python/C API
reference for C/C++ programmers

FAQs
frequently asked questions (with answers!)

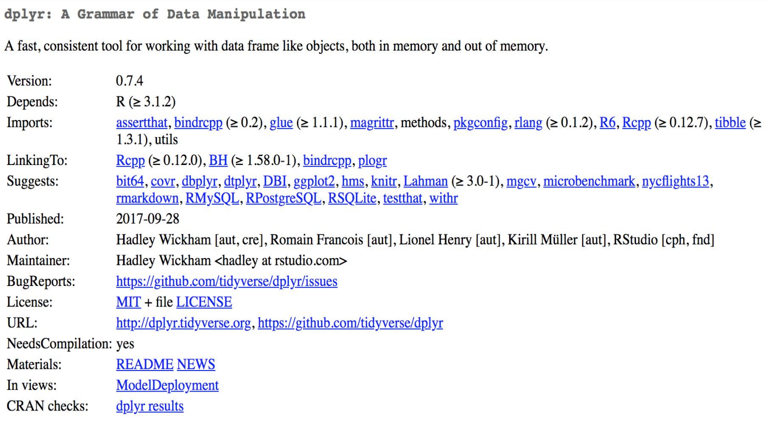
Indices and tables:

Global Module Index
quick access to all modules

General Index

Search page
search this documentation

R - Vignettes



dplyr: A Grammar of Data Manipulation

A fast, consistent tool for working with data frame like objects, both in memory and out of memory.

Version:	0.7.4
Depends:	R (≥ 3.1.2)
Imports:	assertthat, bindrcpp (≥ 0.2), glue (≥ 1.1.1), magrittr, methods, pkgconfig, rlang (≥ 0.1.2), R6, Rcpp (≥ 0.12.7), tibble (≥ 1.3.1), utils
LinkingTo:	Rcpp (≥ 0.12.0), BH (≥ 1.58.0-1), bindrcpp, plogr
Suggests:	bit64, covr, dbplyr, dtplyr, DBI, ggplot2, hms, knitr, Lahman (≥ 3.0-1), mgcv, microbenchmark, nycflights13, rmarkdown, RMySQL, RPostgreSQL, RSQLite, testthat, withr
Published:	2017-09-28
Author:	Hadley Wickham [aut, cre], Romain Francois [aut], Lionel Henry [aut], Kirill Müller [aut], RStudio [cph, fnd]
Maintainer:	Hadley Wickham <hadley at rstudio.com>
BugReports:	https://github.com/tidyverse/dplyr/issues
License:	MIT + file LICENSE
URL:	http://dplyr.tidyverse.org, https://github.com/tidyverse/dplyr
NeedsCompilation:	yes
Materials:	README NEWS
In views:	ModelDeployment
CRAN checks:	dplyr results

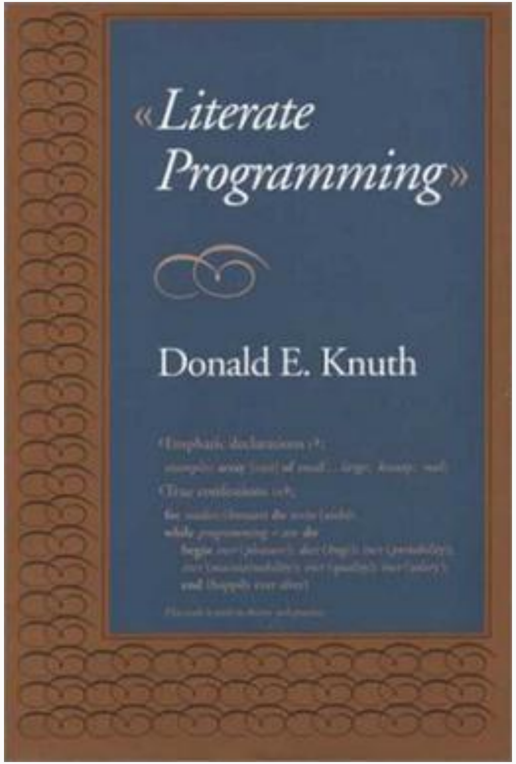
- Journal of Open Source Software
- Journal of Statistical Software

Literate Programming

Combining documentation and code in a single program.

“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.”

R Reporting and sharing: [Knitr](), [RPubs]()
Notebooks - [Jupyter](), [R Notebooks](), [Zeppelin](), [Sage](), [Beaker]()
Notebook Environments: [Binder](), [Colaboratory](), [Kaggle](), [Azure](), [AWS Sagemaker Notebooks]()



[Image by Wikipedia]()

Free Notebook Environments

Azure Notebooks (Microsoft)

- 4GB RAM
- 1GB disk space
- Great Integration with Github
- R and Python

Cons: limited resources

Dask Tutorial Example

Colaboratory Notebooks (Google)

- 13GB RAM
- 33 GB disk
- GPU support
- Notebooks and data on Google Drive
- Integration with Github
- Simultaneous Editing
- Python only so far

Cons: not real filesystem

Kaggle Kernels (Google)

- 16GB RAM
- 5GB disk
- GPU support
- Upload/Edit/Download Notebooks
- Kaggle Datasets: public and private(20GB)
- Version Control Support
- R and Python

Cons: no Github integration

Non-free:

Azure and Colab Notebook can be connected to cloud services for more power.

AWS Sagemaker for ML

Label
Build
Train & Tune
Deploy & Manage

Combining Notebooks

- Binder (mybinder.org)
  - Binds and demos notebooks on a github repo: xarray example
- GitBook (https://docs.gitbook.com/)
  - Combines notebooks into a book
- Papermill (https://papermill.readthedocs.io/en/latest/)
  - Executes notebooks, generates reports

Modular Programming

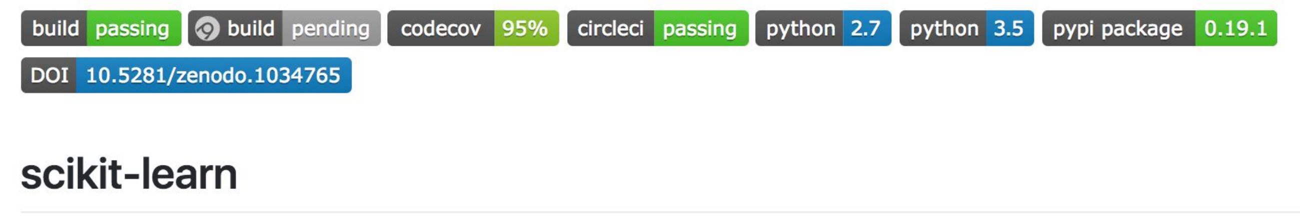
Commands -> Functions -> Modules/Libraries

Convert Notebooks/Scripts to libraries

Virtualization

- Virtual Environments (Conda) - package dependencies
    - supports both for Python and R
    - Make your virtual environment now!
    - Store your dependencies in a requirements.txt file
    - Document each installation while doing it not later!
- Docker Containers - Linux environment, works on all OS
    - Dockerfile: scriptable setup
    - DockerHub: ready-to-go images
        - E.g. postgres database
    - Resolve installation mess
    - Deploy, Scale
- Vagrant - virtual machine manager, can run both Docker containers and full VMs
- Virtual Machines - VirtualBox, VMWare
- Cloud Images - AWS AMIs

Testing



build passing | build pending | codecov 95% | circleci passing | python 2.7 | python 3.5 | pypi package 0.19.1 | DOI 10.5281/zenodo.1034765

scikit-learn

We are already writing tests, need to save them.

Types of testing: unit, integration, system, regression

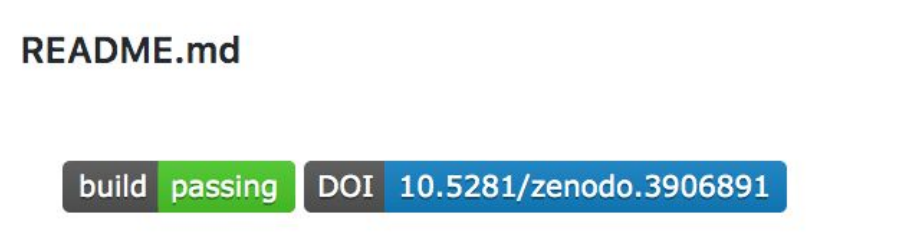
- Locally
  - Python - nose, pytest, tox
  - R - testthat

- Remotely - Continuous Integration
  - Travis, CircleCI, AppVeyor, Github Actions

Start by testing the environment.

Digital Object Identifiers

- Articles :
  - Arxiv, preprint server
  - journals create it for you
- Slides, Posters:
  - F1000 Research
- Software:
  - For any github repository using Zenodo
- Datasets:
  - Persistent repositories provide DOI



README.md

build passing DOI 10.5281/zenodo.3906891

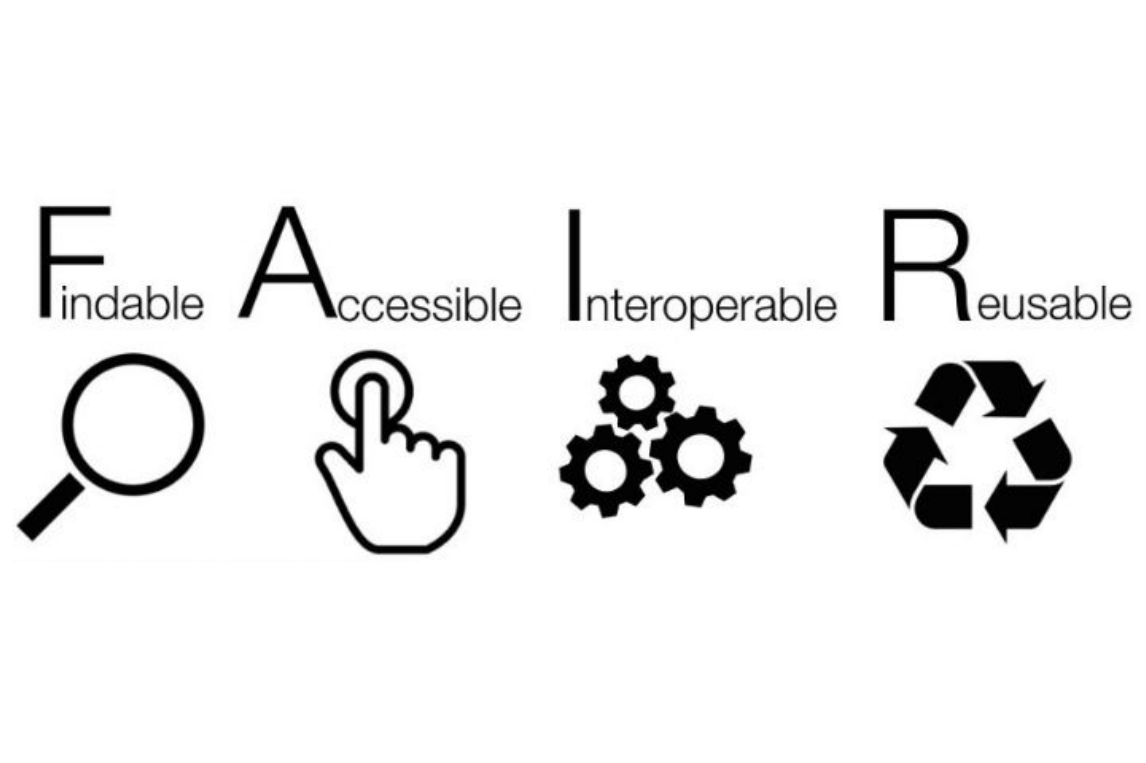
Data Repositories

zenodo	DRYAD	figshare	IEEEDataPort™
Up to 50GB free Not-for-profit - EU funded (contact if more)	Publishing Fee - $120 Excess fees after 20GB Associated with articles Not-for-profit	100GB free per manuscript Institutional plans For-profit	Up to 2TB Subscription Based Free Promo Codes

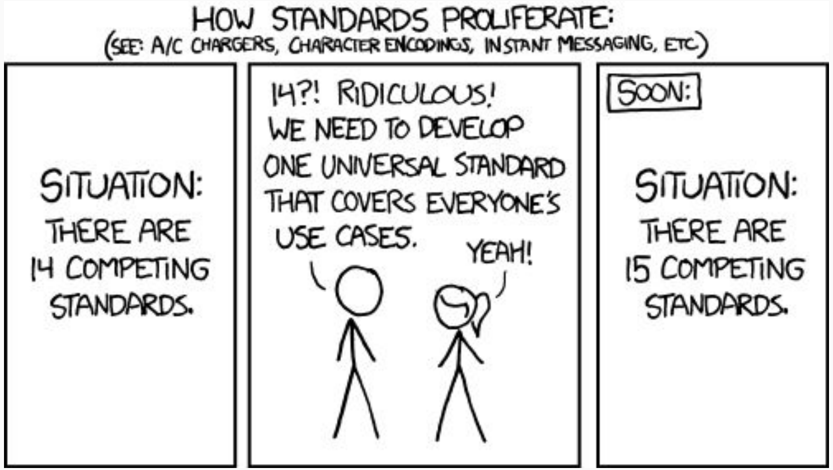
- Datasets receive Digital Object Identifier (DOI)
- Cloud Storage: free to upload, fees for storage, higher fees to download
  - Some public datasets can be stored for free
- Nature Journal Scientific Data: https://www.nature.com/sdata/

Data and Metadata

[*‘FAIR Guiding Principles for scientific data management and stewardship’*, Wilkinson et.al, Nature Scientific Data, 2016]



Standardization


HOW STANDARDS PROLIFERATE:
(SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC)
SITUATION: THERE ARE 14 COMPETING STANDARDS.
14?! RIDICULOUS! WE NEED TO DEVELOP ONE UNIVERSAL STANDARD THAT COVERS EVERYONE'S USE CASES.
YEAH!
SOON:
SITUATION: THERE ARE 15 COMPETING STANDARDS.

- Try using standard formats whenever possible!
- If format does not fit your case discuss with the community! ISO, NIST
- Often standards are more permissive than data formats!
- Interoperability
- Persistency

Standardization: examples

Example 1: you have pulled some election data and you want to organize it so that it is easy for other researchers to analyse it

- Store it in excel sheets
- Store it in csv files
- Check how the data for other states is stored and store it the same format
- Check out if there are standards for election data
  - https://www.nist.gov/publications/election-results-common-data-format-specification-revision-20

Example 2: you want to save a Deep Learning model so you can apply to future data:

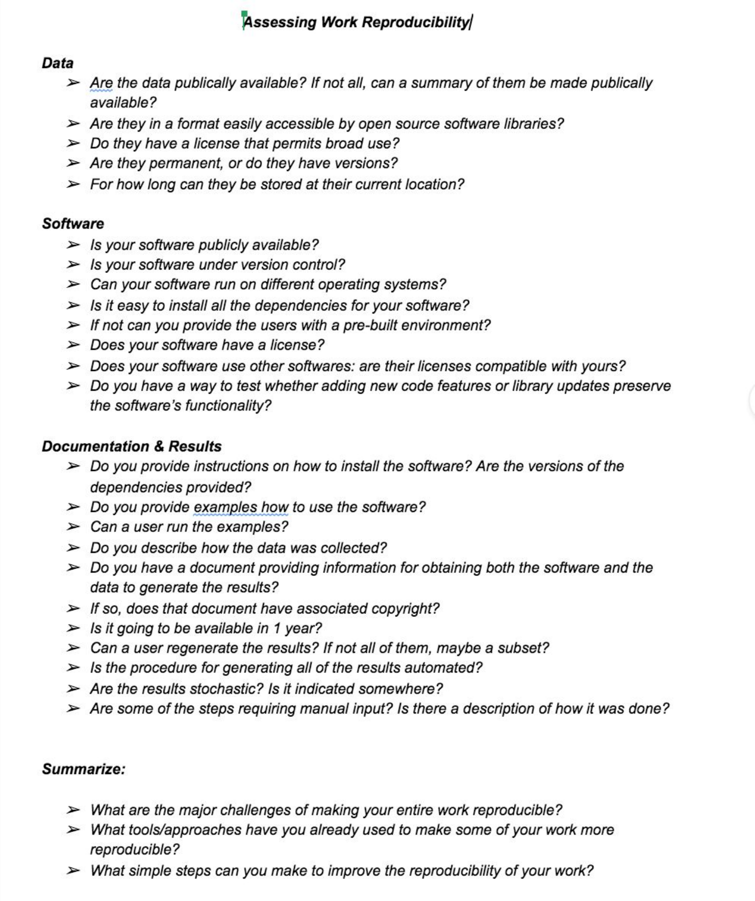
- Python pickle file (Python specific, sometimes version dependent)
- HDF format (Python and domain independent, local database: but fields are not standardized)
- Tensorflow HDF (libraries come and go)
- ONNX (Open Neural Network Exchange) format (library/language independent, stores the 'math', i.e. the computational graph operations, hardware)

Beyond exact reproducibility

- Design your study
- Registering Hypothesis
- Baseline
- Nested Analyses
- Test simple scenarios first
- Precision Recall Curves:
  - Baseline changes for different class distributions
- Cross-validation with dependence in time series and groups
  - https://scikit-learn.org/stable/modules/cross_validation.html
- Multiple Testing

What about your projects?

Reproducibility Checklist:



Assessing Work Reproducibility

Data

- *Are the data publically available? If not all, can a summary of them be made publically available?*
- *Are they in a format easily accessible by open source software libraries?*
- *Do they have a license that permits broad use?*
- *Are they permanent, or do they have versions?*
- *For how long can they be stored at their current location?*

Software

- *Is your software publicly available?*
- *Is your software under version control?*
- *Can your software run on different operating systems?*
- *Is it easy to install all the dependencies for your software?*
- *If not can you provide the users with a pre-built environment?*
- *Does your software have a license?*
- *Does your software use other softwares: are their licenses compatible with yours?*
- *Do you have a way to test whether adding new code features or library updates preserve the software's functionality?*

Documentation & Results

- *Do you provide instructions on how to install the software? Are the versions of the dependencies provided?*
- *Do you provide examples how to use the software?*
- *Can a user run the examples?*
- *Do you describe how the data was collected?*
- *Do you have a document providing information for obtaining both the software and the data to generate the results?*
- *If so, does that document have associated copyright?*
- *Is it going to be available in 1 year?*
- *Can a user regenerate the results? If not all of them, maybe a subset?*
- *Is the procedure for generating all of the results automated?*
- *Are the results stochastic? Is it indicated somewhere?*
- *Are some of the steps requiring manual input? Is there a description of how it was done?*

Summarize:

- *What are the major challenges of making your entire work reproducible?*
- *What tools/approaches have you already used to make some of your work more reproducible?*
- *What simple steps can you make to improve the reproducibility of your work?*

Survey

https://tinyurl.com/2020ReproducibleScience