Exploring Computational Reproducibility in Jupyter Notebooks: Insights and Challenges

Sheeba Samuel

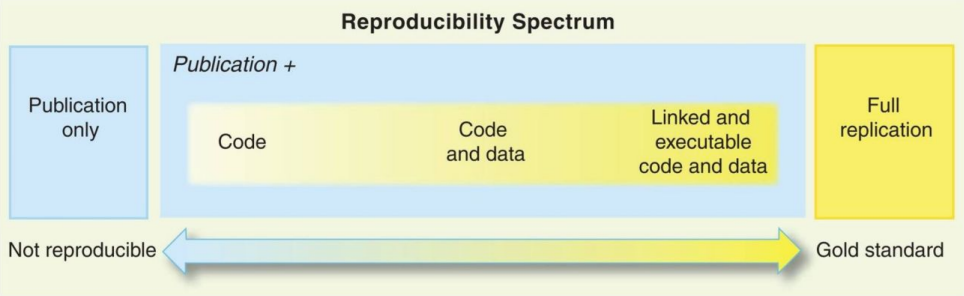
Frühjahrstreffen 2024 der Fachgruppe Datenbanken
12th March 2024

FRIEDRICH-SCHILLER-UNIVERSITÄT JENA
FUSION
MichaelStifelCenterJena for Data-Driven and Simulation Science
Carl Zeiss Stiftung

Outline

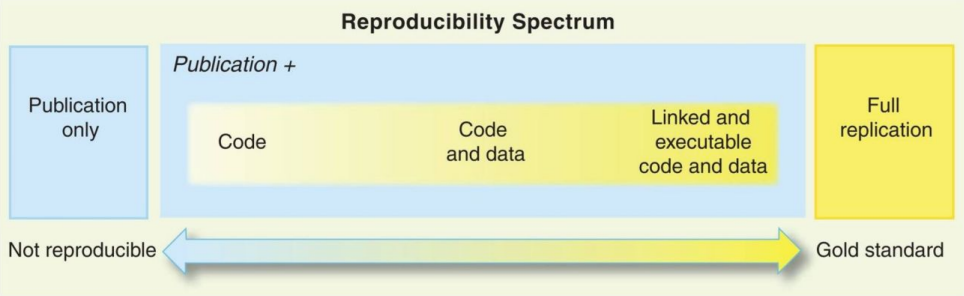
- Background
- Methodology
- Implementation
- Results
- Implications
- Recommendations

Background



[Reproducible Research in Computational Science, Roger D. Peng, 2011]

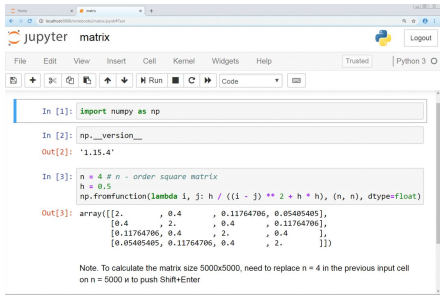
Background



[Reproducible Research in Computational Science, Roger D. Peng, 2011]

A Large-scale Study about Quality and Reproducibility of Jupyter Notebooks

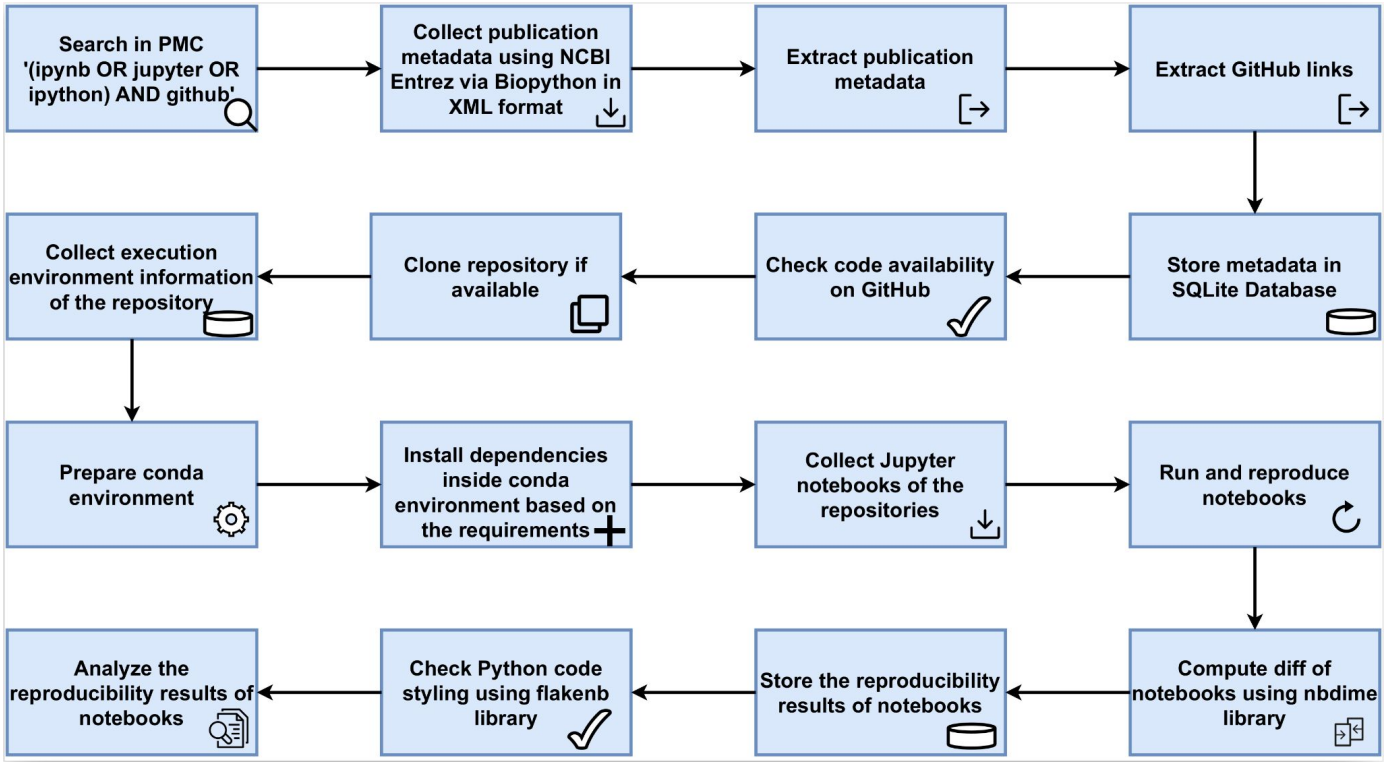
João Felipe Pimentel*, Leonardo Murta*, Vanessa Braganholo*, and Juliana Freire†



Ten simple rules for writing and sharing computational analyses in Jupyter Notebooks

Adam Rule[1], Amanda Birmingham[2], Cristal Zuniga[3], Ilkay Altintas[4], Shih-Cheng Huang[4¤], Rob Knight[3,5], Niema Moshiri[6], Mai H. Nguyen[4], Sara Brin Rosenthal[2], Fernando Pérez[7], Peter W. Rose[4]*

Methodology



Sheeba Samuel & Daniel Mietchen - DOI: 10.1093/gigascience/giad113

Implementation

- Initial Run
  - 2021
  - Preprint: https://doi.org/10.48550/arXiv.2209.04308
- Rerun
  - 2023

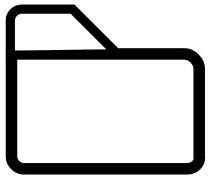
Code : https://github.com/fusion-jena/computational-reproducibility-pmc
Data : https://doi.org/10.5281/zenodo.8226725
Paper : https://doi.org/10.1093/gigascience/giad113

Results

740
Journals

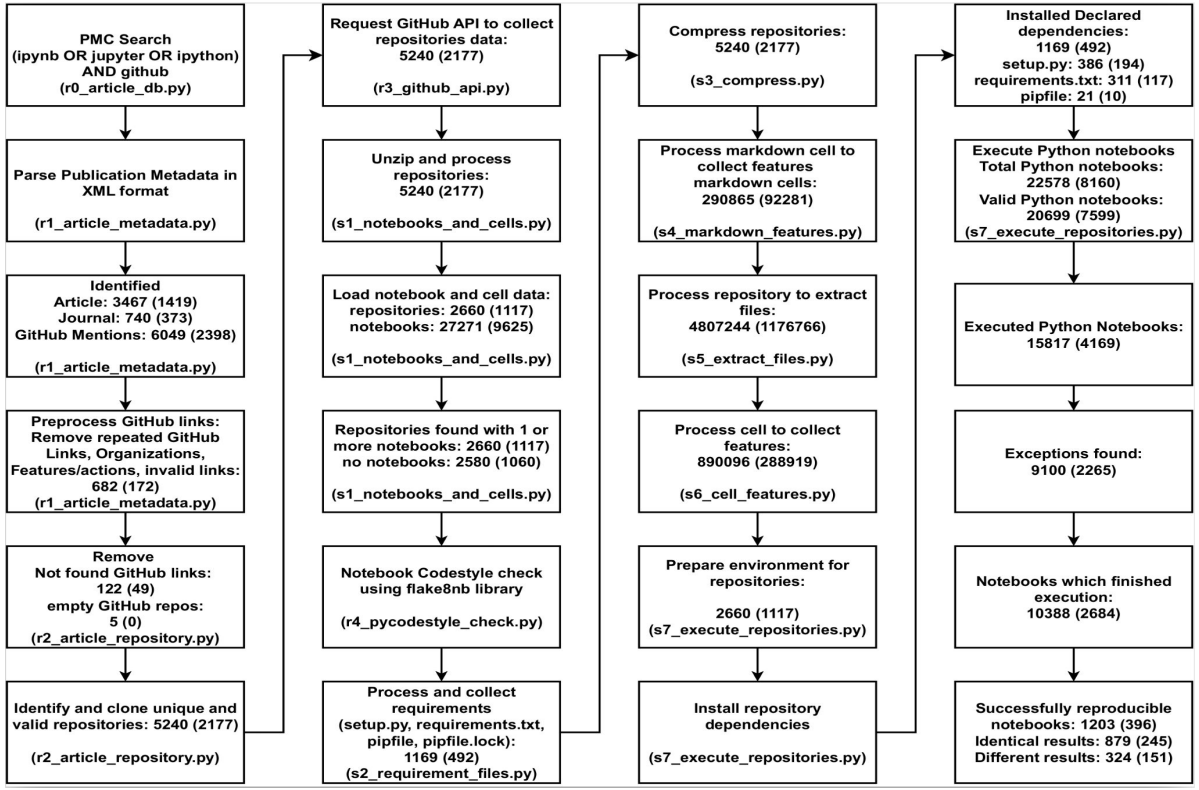
3,467
Publications

2,660
Repositories

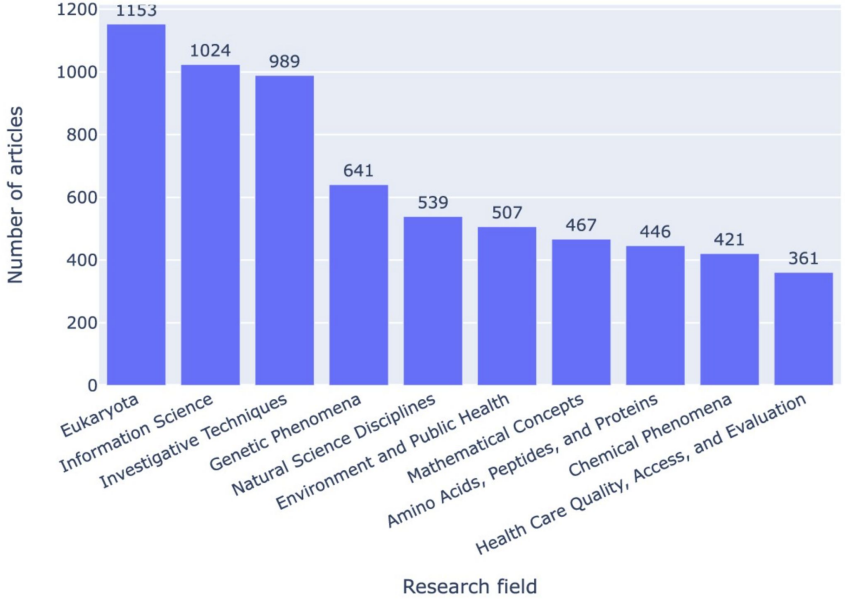


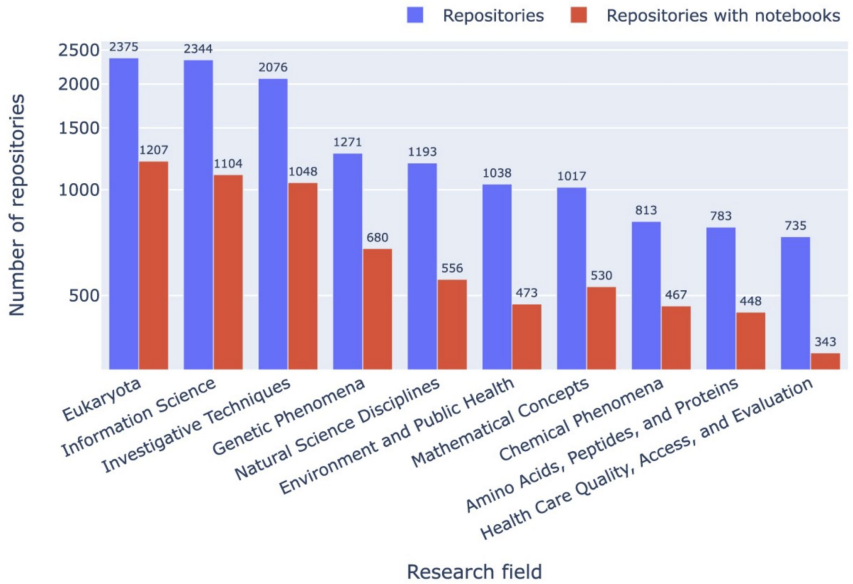
27,271
Notebooks

Results - Initial vs Rerun

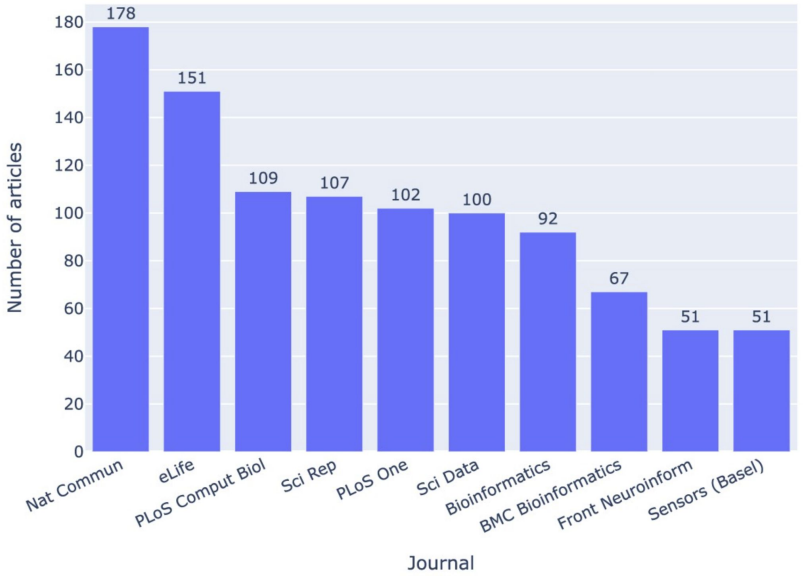


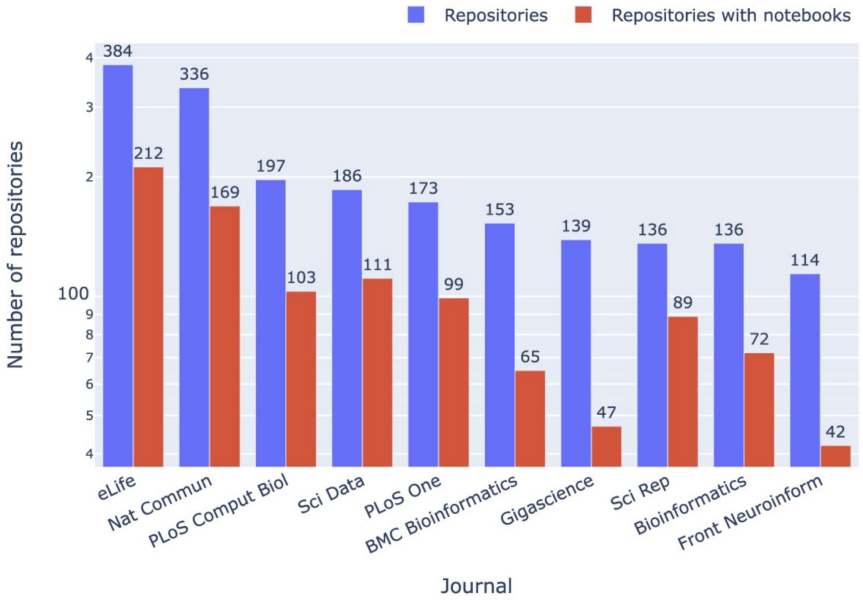
Results - Research Field



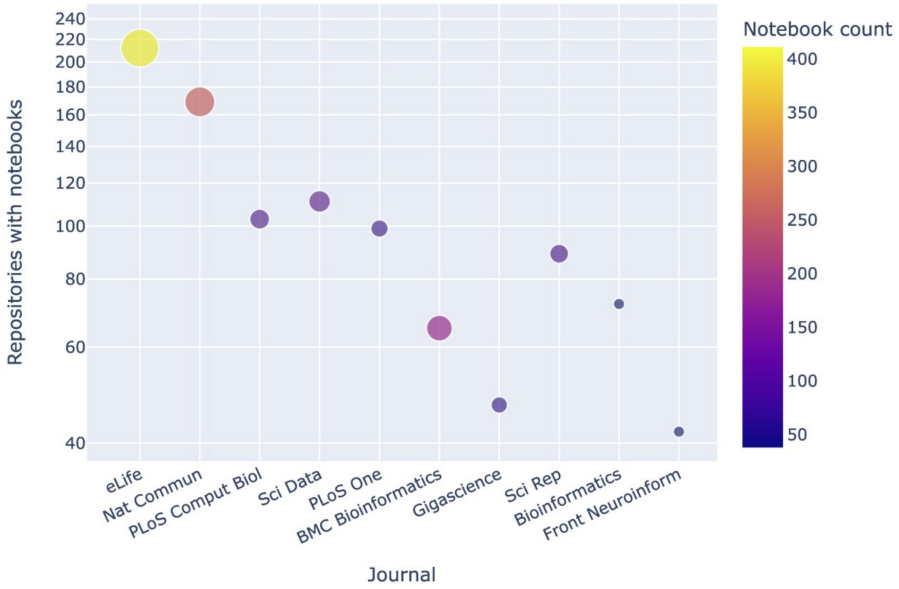


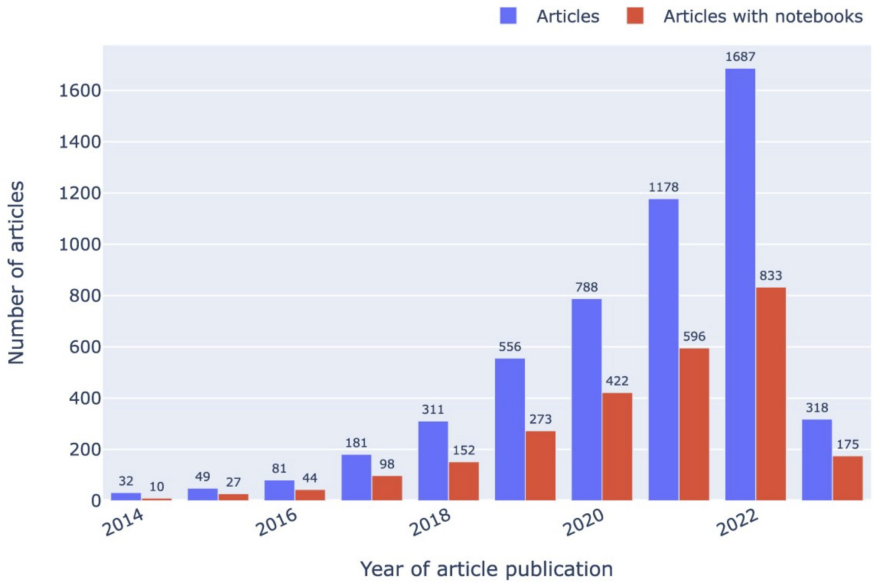
Results - Journals



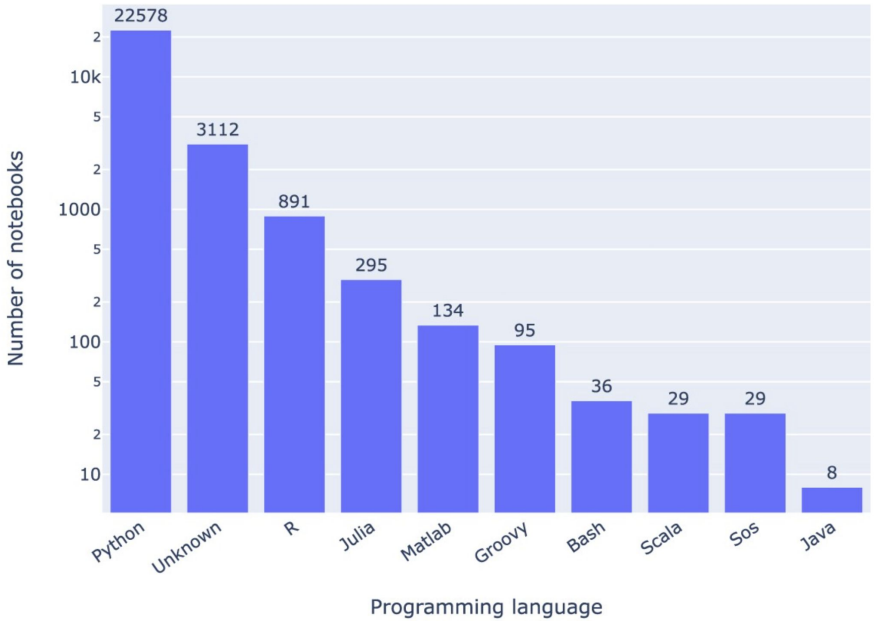


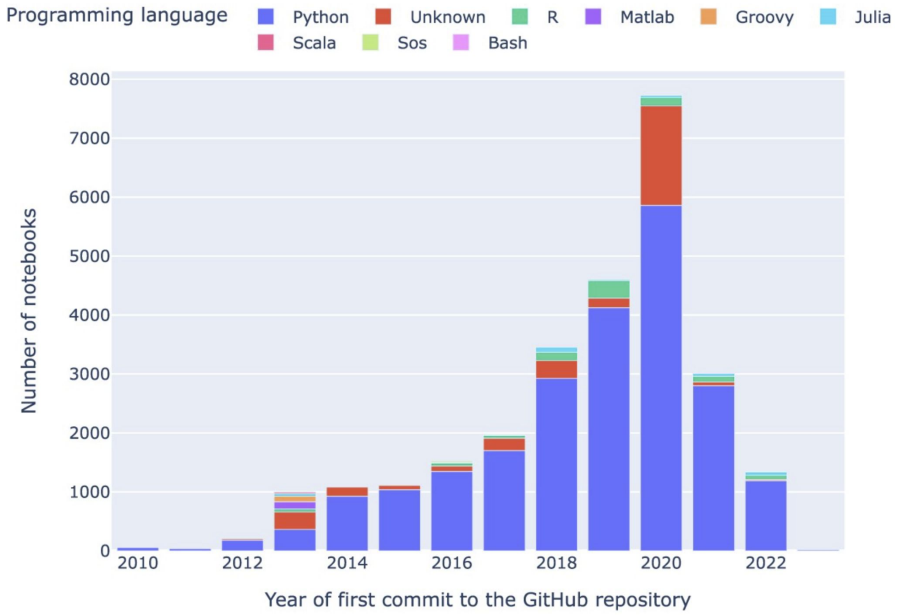
Results - Journals





Results - Programming Languages

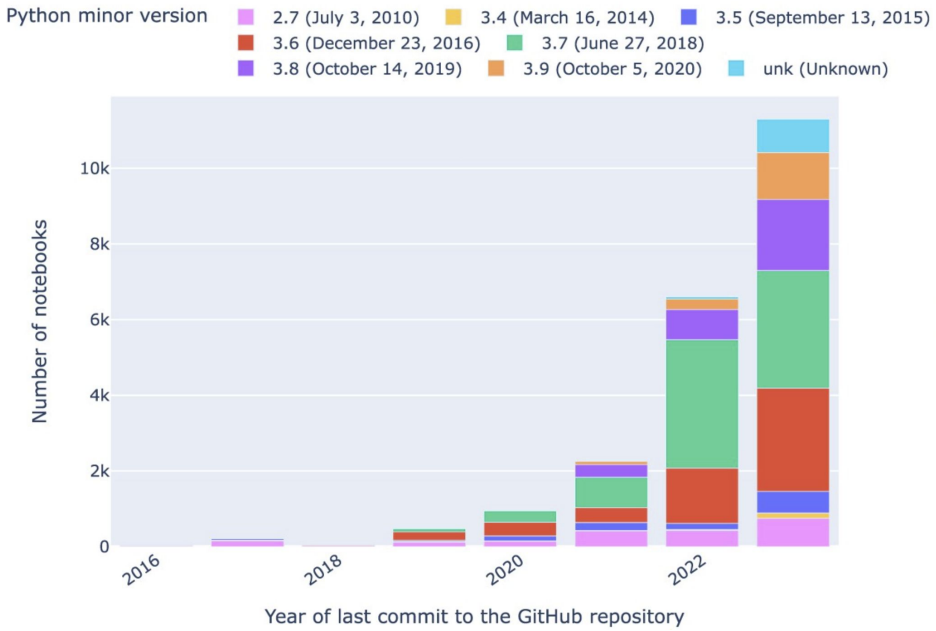


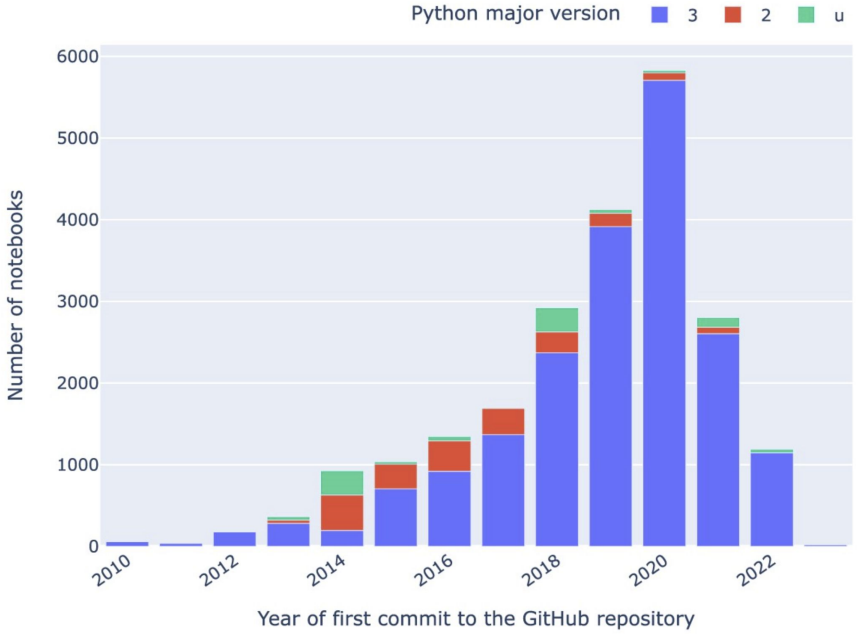


Results - Programming Languages

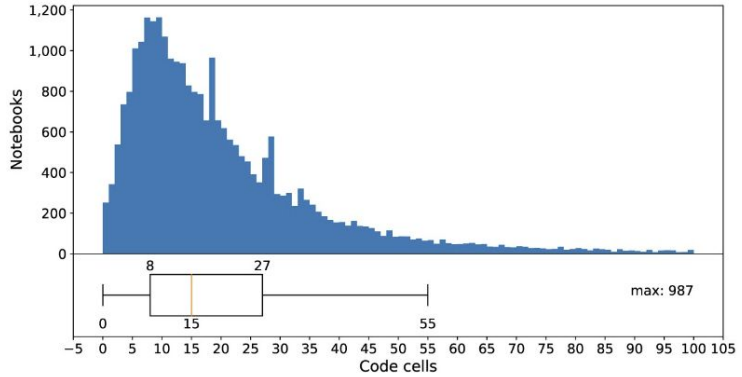
Notebook language	Rerun	Initial run	Ratio rerun/initial
Matlab	134	9	14.9
Julia	295	59	5.0
Unknown	3,112	720	4.3
Python	22,578	8,160	2.8
R	891	461	1.9
Bash	36	24	1.5
Sos	29	24	1.2
Groovy	95	95	1.0
Scala	29	29	1.0
Java	8	8	1.0

Results - Python Version

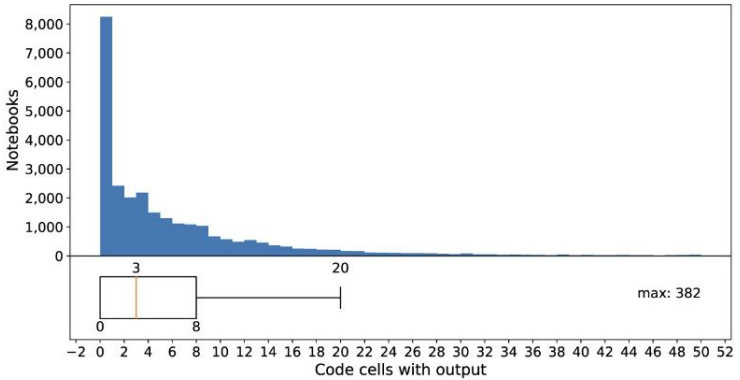




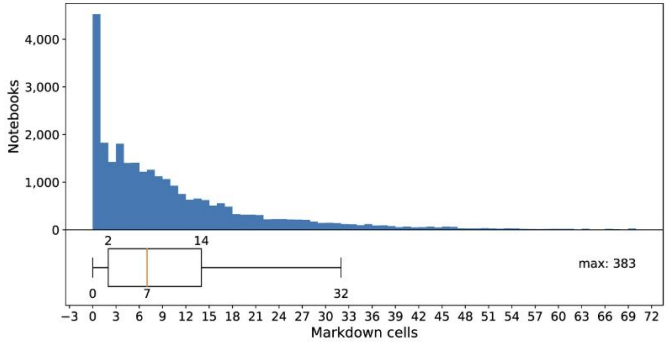
Results - Notebook Structure



A Distribution of the number of code cells.

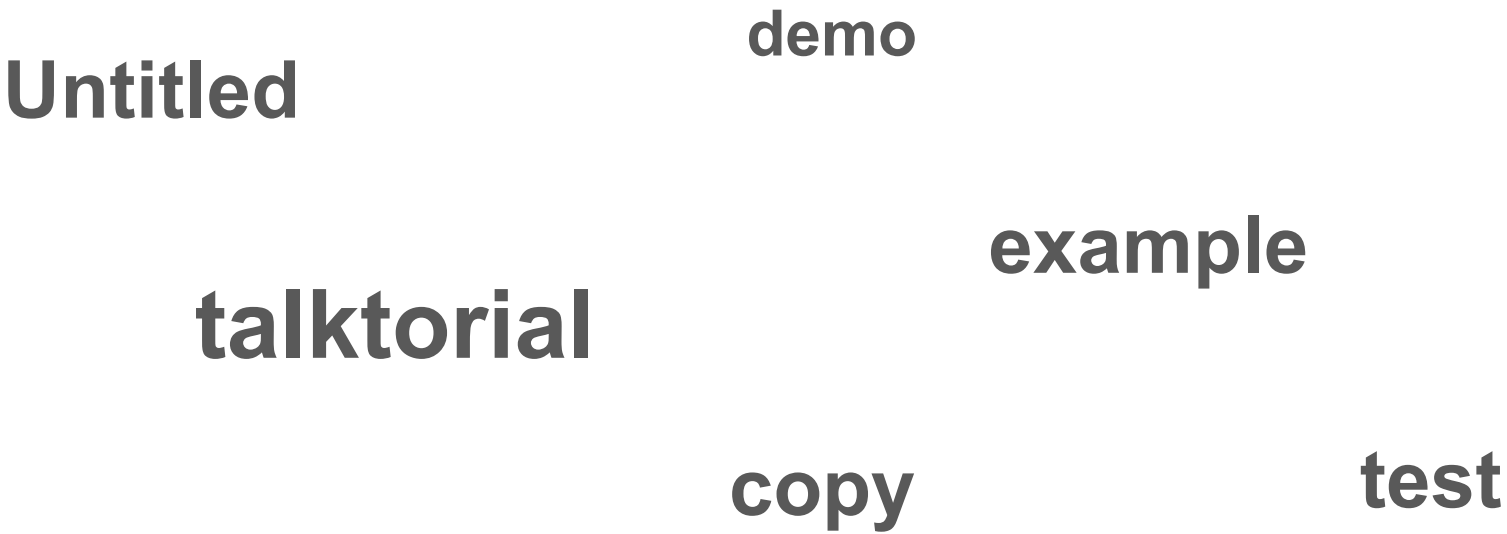


B Distribution of the number of code cells with outputs.

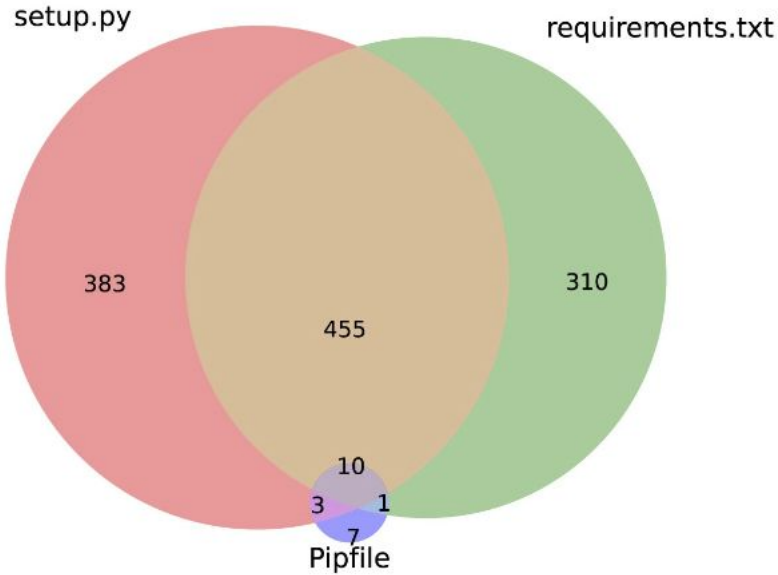


C Distribution of the number of Markdown cells.

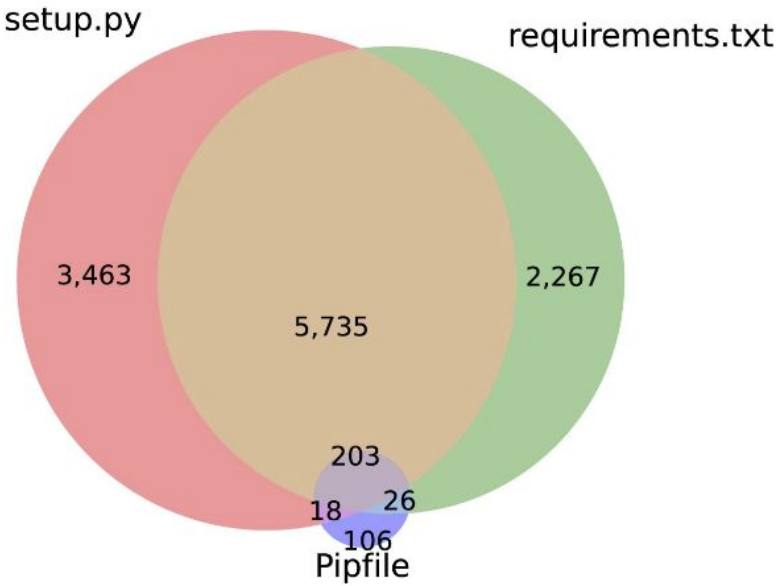
Results - Notebook Naming



Results- Notebook Dependencies

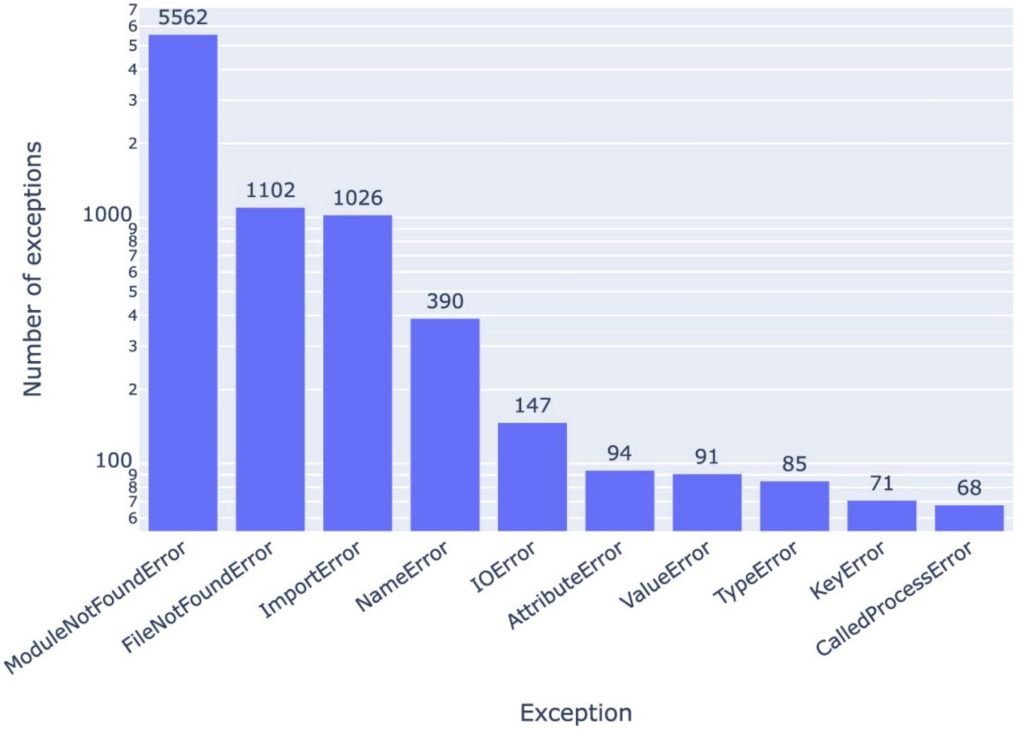


Repositories with dependencies.

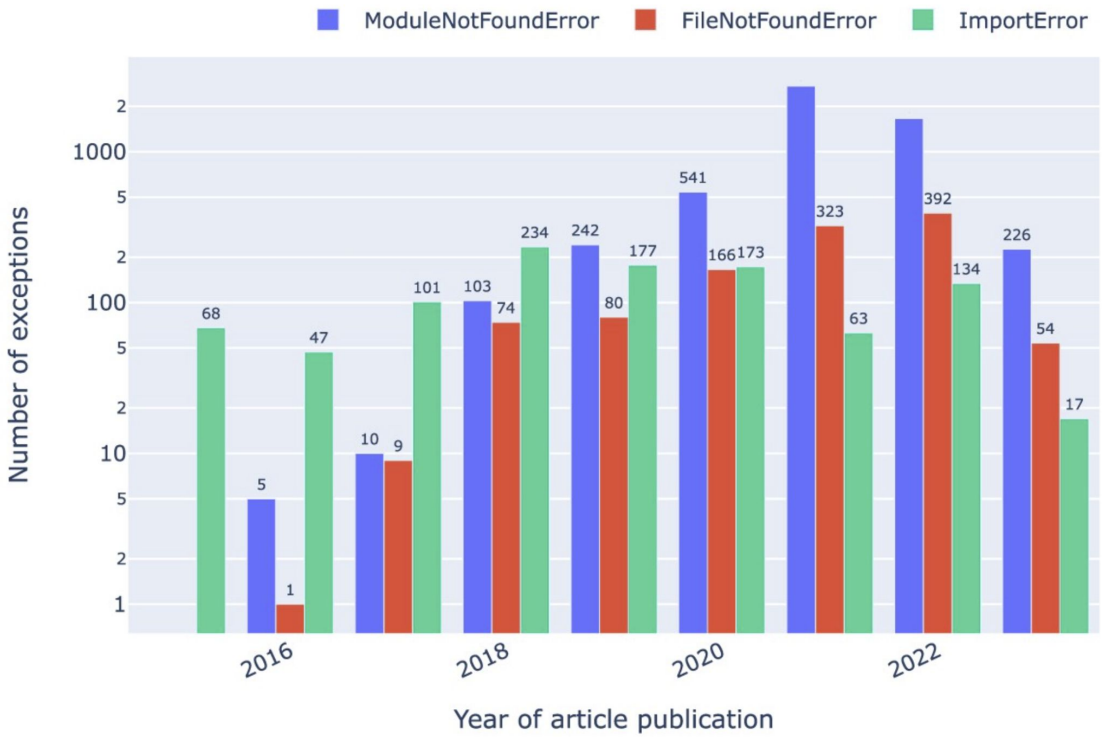


Notebooks with dependencies.

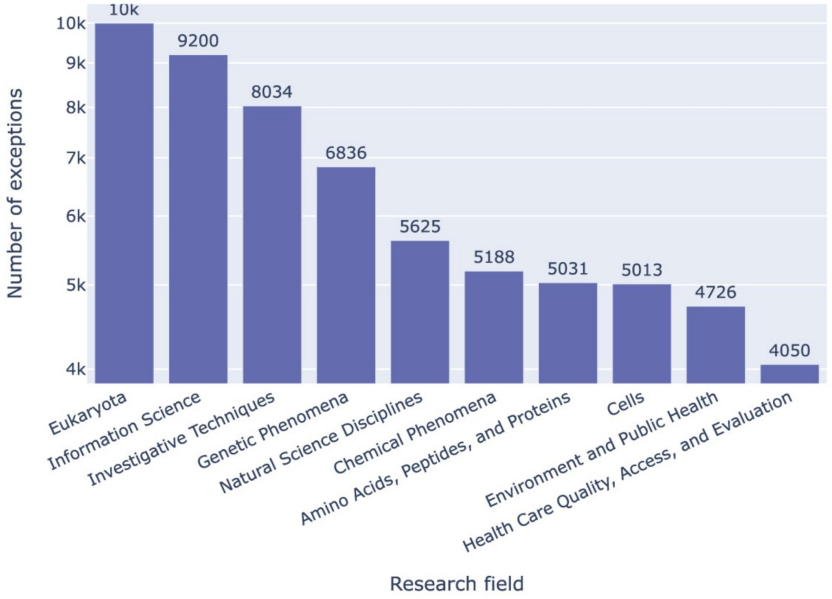
Results - Notebook Reproducibility

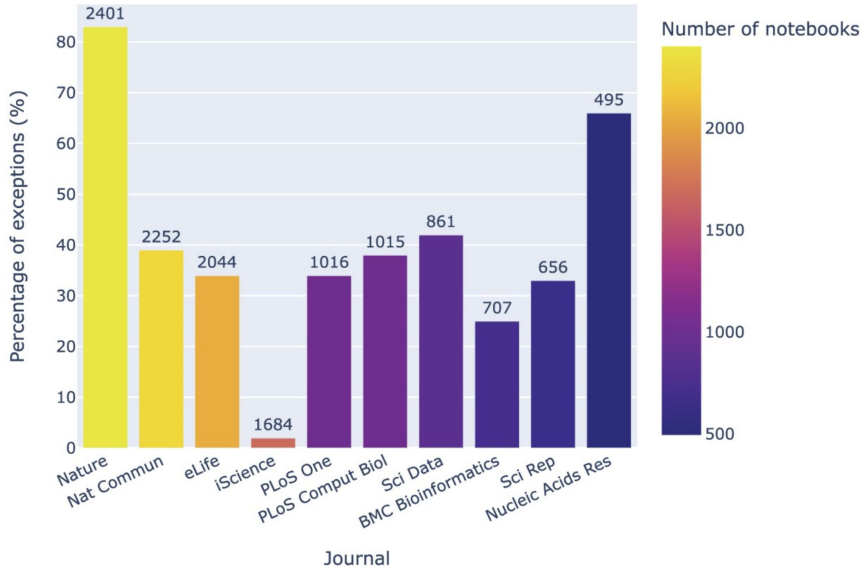


Results - Notebook Reproducibility

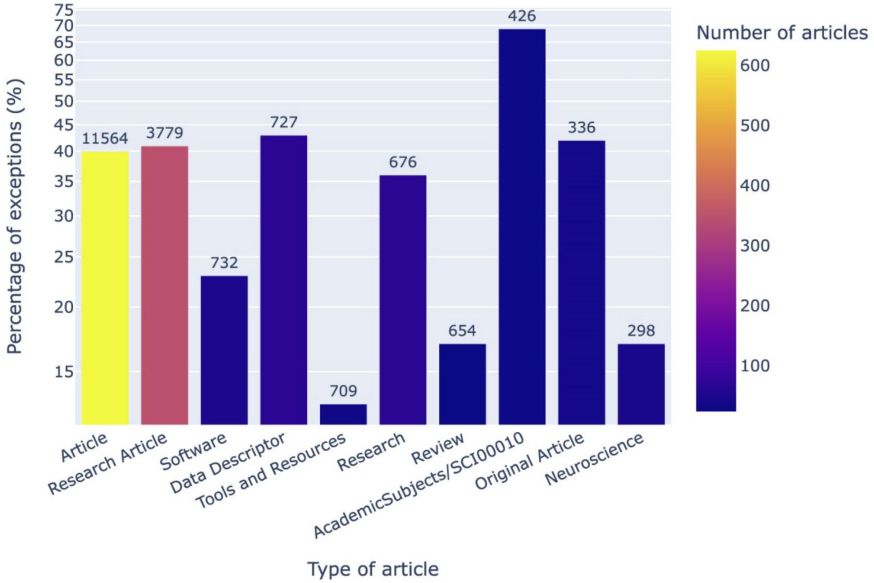


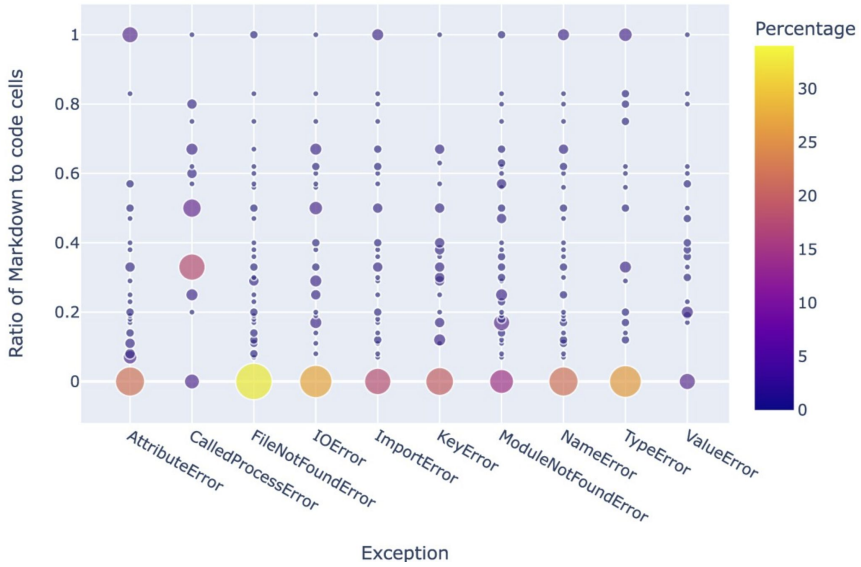
Results - Notebook Reproducibility





Results - Notebook Reproducibility





Results - Successful reproductions

Features	Notebooks with `different` results	Notebooks with `identical` results
Number of notebooks	324 (⊠ 151)	879 (⊠ 245)
setup.py	0 (⊠ 0)	344 (⊠ 98)
requirement.txt	1 (⊠ 0)	353 (⊠ 107)
pipfile	0 (⊠ 0)	0 (⊠ 0)
Total cells	23 (⊠ 17.9)	19.6 (⊠ 17.1)
Code cells	15.4 (⊠ 12.3)	11.3 (⊠ 9.8)
Markdown cells	7.6 (⊠ 5.6)	8.3 (⊠ 6.7)
Ratio of Markdown vs. code cells	0.49 (⊠ 0.46)	0.73 (⊠ 0.68)
Empty cells	0.9 (⊠ 0.7)	0.7 (⊠ 0.7)
Differences	6.3 (⊠ 5.3)	0 (⊠ 0)
Execution time (s)	18.3 (⊠ 22.1)	57.6 (⊠ 16.4)
Execution time per code cell (s)	1.88 (⊠ 1.80)	5.09 (⊠ 1.67)

Results - Notebook Styling

Error code	Description	Count (%)
E231	Missing whitespace after commas, semicolons, or colons	686,382 (25.2%) ⊠ 102,218 (27.3%)
E225	Missing whitespace around operator	187,528 (6.9%) ⊠ 25,979 (6.9%)
E265	Block comment should start with “#”	110,972 (4.1%) ⊠ 10,769 (2.9%)
E402	Module level import not at top of file	108,067 (4.0%) ⊠ 10,478 (2.8%)
E262	Inline comment should start with “#”	32,704 (1.2%) ⊠ 8,369 (2.2%)
E703	Statement ends with a semicolon	13,944 (0.5%) ⊠ 2,023 (0.5%)
E127	Continuation line overindented for visual indent	15,486 (0.6%) ⊠ 1,290 (0.3%)
E701	Multiple statements on 1 line	5,147 (0.2%) ⊠ 500 (0.1%)
E741	Do not use variables named “l,” “O,” or “I”	2,398 (0.1%) ⊠ 432 (0.1%)
E401	Multiple imports on 1 line	1,293 (0.0%) ⊠ 95 (0.0%)
E101	Indentation contains mixed spaces and tabs	2,881 (0.1%) ⊠ 32 (0.0%)
F405	*Name* may be undefined, or defined from star imports: *module*	46,825 (1.7%) ⊠ 4,840 (1.3%)
F401	*Module* imported but unused	46,988 (1.7%) ⊠ 3,938 (1.1%)
F821	Uundefined name “X”	20,987 (0.8%) ⊠ 2,071 (0.6%)
F403	“From module import*” used; unable to detect undefined names	4,371 (0.2%) ⊠ 263 (0.1%)
F841	Local variable “X” is assigned to but never used	3,195 (0.1%) ⊠ 225 (0.1%)
F404	Future import(s) name after other statements	309 (0.0%) ⊠ 44 (0.0%)
F402	Import “X” from line Y shadowed by loop variable	79 (0.0%) ⊠ 10 (0.0%)
F633	Use of >> is invalid with print function	6 (0.0%) ⊠ 6 (0.0%)

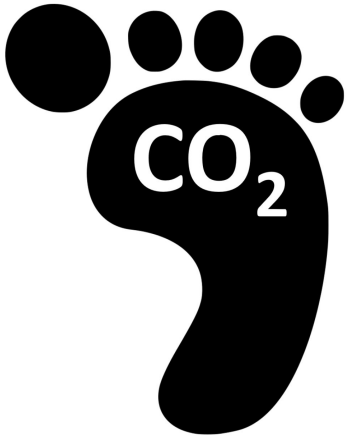
Environmental Footprint

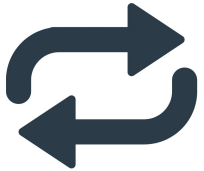
Simplified calculation: carbon footprint = energy needed * carbon intensity

(http://calculator.green-algorithms.org/)

47.38 kWh

16.05 kg CO2e





373.78 kWh

126.58 kg CO2e

Implications

- Computational reproducibility is a key element of scientific reproducibility
- Learning/ teaching through errors and fixing them
- Mechanisms to communicate successful reproduction to the scientific community

Error type	Underlying problem	Some potential fixes
ModuleNotFoundError	Module cannot be located	Check that module is present in repo or installed in the environment
FileNotFoundError	File cannot be located at designated path	Check path and that file exists at path
ImportError	Attribute, function, class, or variable cannot be imported from a module as specified	Check documentation of what is to be imported, including the module's dependencies
NameError	Variable or function is used but not defined	Check documentation about where it ought to be defined (e.g., another cell or module); execute that code before using that name
IOError	Trying to read from or write to a destination that does not exist or for which user does not have pertinent permissions	Check existence, path, and permissions of that destination
AttributeError	Trying to access an attribute or method that does not exist as specified	Check documentation and ensure the access is handled as required
ValueError	A function is called with an argument of the correct type but with a wrong value	Check that argument meets the requirements of the function
TypeError	A function is called with an argument of the correct type but with a wrong value	Check that argument number and argument types meet the requirements
KeyError	Trying to access a dictionary key that does not exist	Check that the key exists in the target dictionary; consider setting default values for cases when key does not exist
CalledProcessError	A subprocess was called but returned a nonzero exit status	Check that the subprocess is being called as required and that the called code actually works as intended

Recommendations

- Existing recommendations are good but need to be
  - integrated into actual workflows
    - e.g. see Julynter
  - adapted to the role(s) of different stakeholders
  - expanded to take into account different types of dependencies
- Make notebooks citable
  - run them before citing
- Routinely re-run notebooks
  - both your own and those of others
- Monitor the environmental footprint

Thank you

- Carl Zeiss Foundation for the project “A Virtual Werkstatt for Digitization in the Sciences (K3)”
- Alfred P. Sloan Foundation under grant number G-2021-17106.
- Ara Cluster of Friedrich Schiller University Jena under DFG grants INST 275/334-1 FUGG and INST 275/363-1 FUGG.

- **Contact**: sheeba.samuel@uni-jena.de & daniel.mietchen@ronininstitute.org

FRIEDRICH-SCHILLER-
UNIVERSITÄT
JENA

FUSION

MichaelStifelCenterJena
for Data-Driven and Simulation Science

Carl Zeiss
Stiftung