Reproducibility and Open Research Software

Dr. Rachael Ainsworth
Research Software Community Manager
Software Sustainability Institute, University of Manchester

@rachaelevelyn
@rainsworth
https://doi.org/10.6084/m9.figshare.11762121

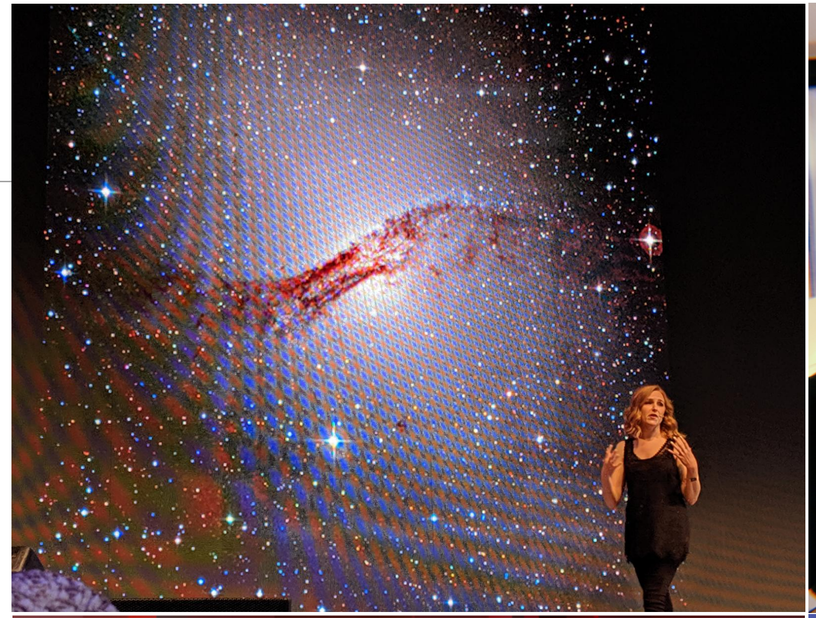
Outline

- About me and the Software Sustainability Institute
- Reproducibility and research culture
- Barriers to sharing research software
- Benefits to sharing research software
- How to share your research software and get credit
- Takeaways

About me and the Software Sustainability Institute

About me

- Community Manager for the Software Sustainability Institute at the University of Manchester
- Research background in Astrophysics
- Passionate about openness, transparency, reproducibility, wellbeing and inclusion in STEM/research
- TEDx speaker: youtu.be/c-bemNZ-IqA
- Was a cartoon in the UK's National Science and Media Museum Hello Universe exhibition
- Organise the Manchester women in data meetup group HER+Data MCR meetup.com/HER-Data-MCR



Software Sustainability Institute

- A national facility promoting the advancement of software in research by cultivating better, more sustainable, research software to enable world-class research: ***“Better software, better research”***

- Based at the Universities of Edinburgh, Manchester, Oxford and Southampton, and we have Software, Policy, Outreach, Training and Community teams to support the community developing and using research software

- Fellowship Programme to engage with and support natural ambassadors of better software practice in their research domains

- https://www.software.ac.uk/

Collaborations Workshop 2020 (CW20) #CollabW20

Software Sustainability Institute

- March 31 - April 2, 2020 at Queen's University Belfast, Northern Ireland
- Bringing together researchers, developers, innovators, managers, funders, publishers, leaders and educators to explore best practices and the future of research software
- Unconference: keynote presentations, mini-workshop/demo sessions, discussion groups, lightning talks, panel sessions, collaborative ideas and a hack day
- Themes: Open Research, Data Privacy and Software Sustainability
- http://bit.ly/ssi-cw20

Dr. Rachael Ainsworth, SSI @rachaelevelyn @OpenResLDN #ResearchSoftware
https://doi.org/10.6084/m9.figshare.11762121

Reproducibility and research culture

nature International weekly journal of science

Home | News & Comment | Research | Careers & Jobs | Current Issue | Archive | Audio & Video

Archive > Volume 533 > Issue 7604 > News Feature > Article

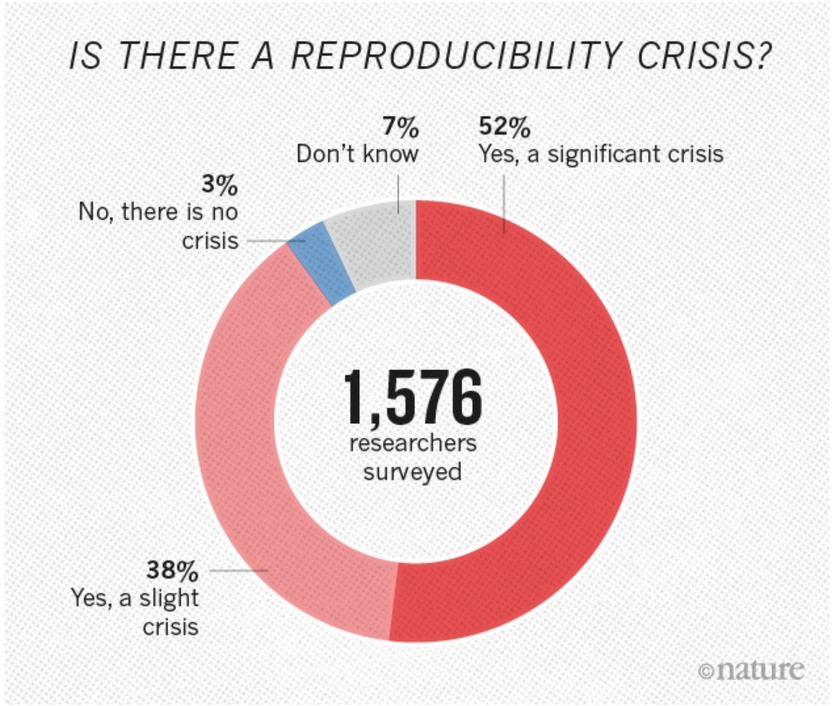
NATURE | NEWS FEATURE

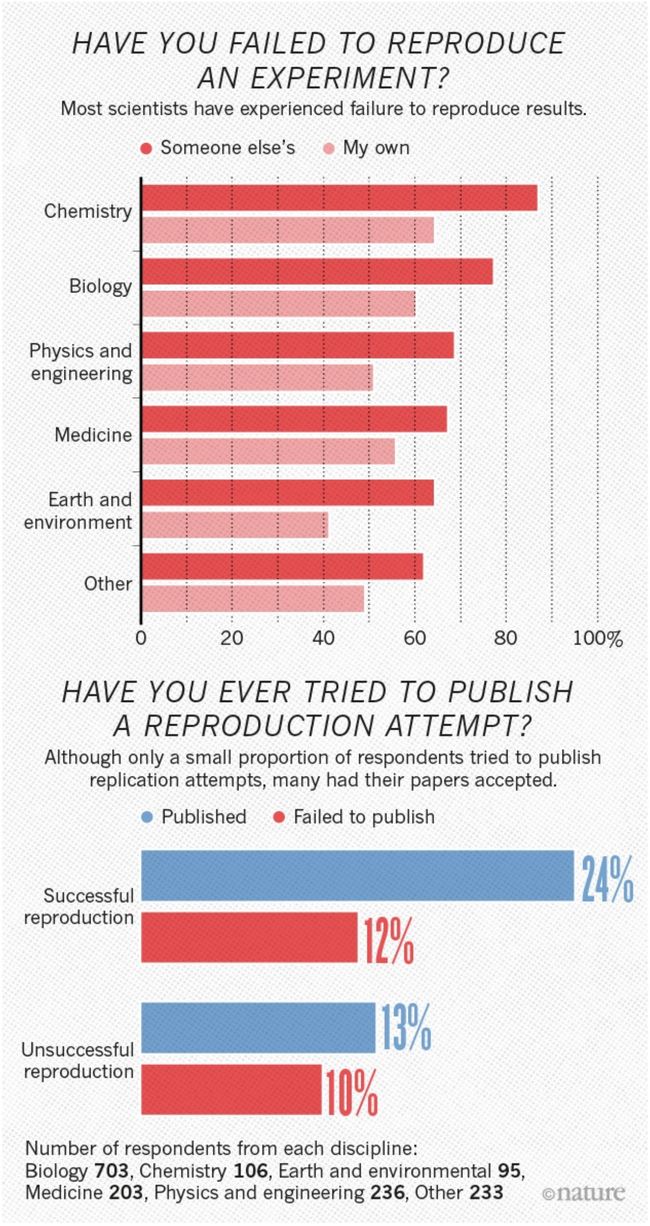
1,500 scientists lift the lid on reproducibility

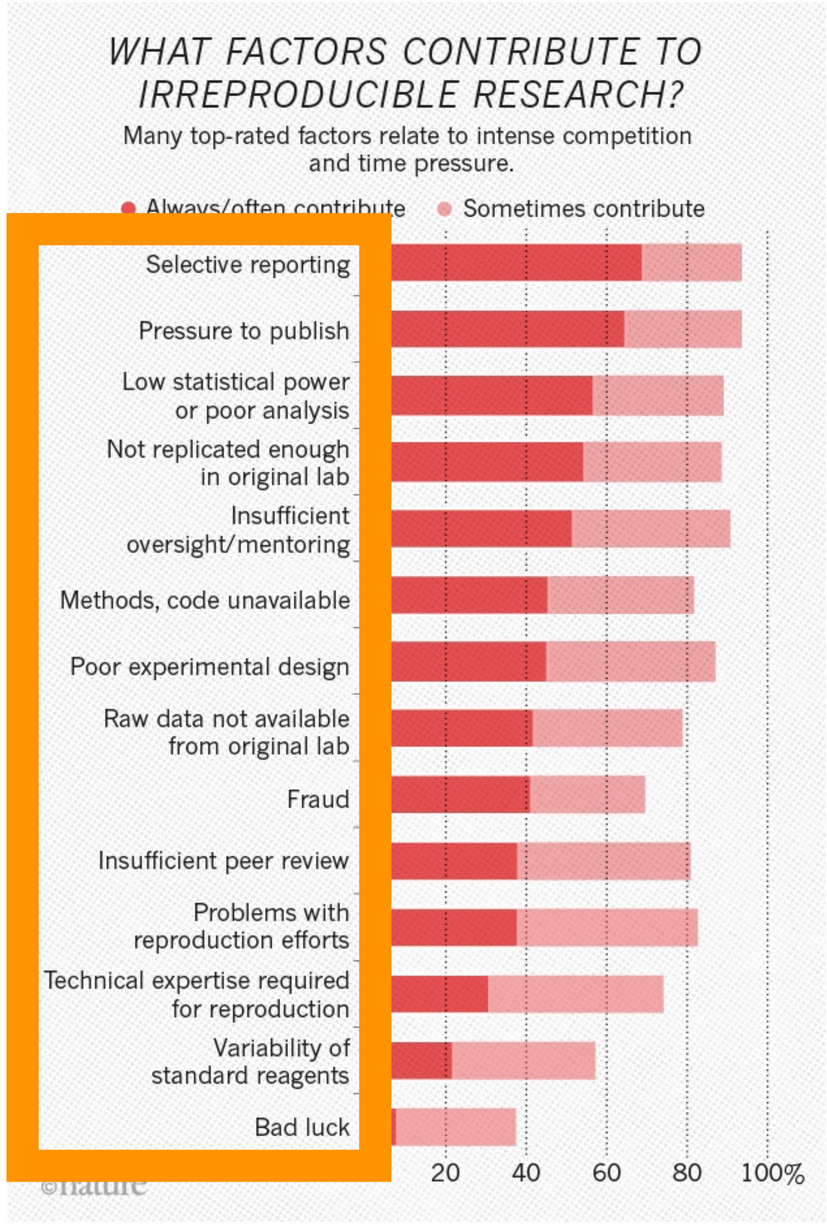
Survey sheds light on the 'crisis' rocking research.

Monya Baker

25 May 2016 | Corrected: 28 July 2016





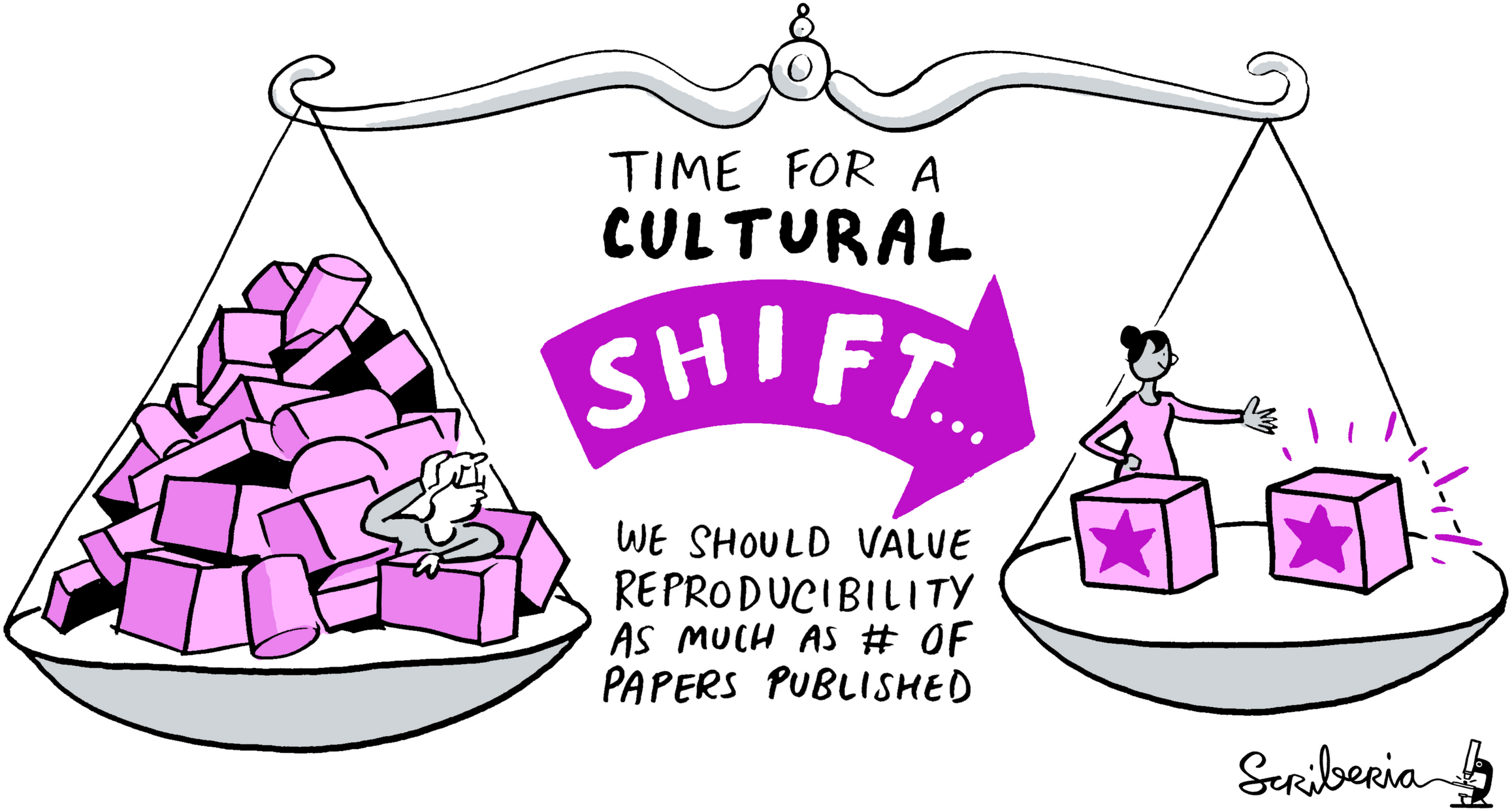


Baker (2016) https://doi.org/10.1038/533452a

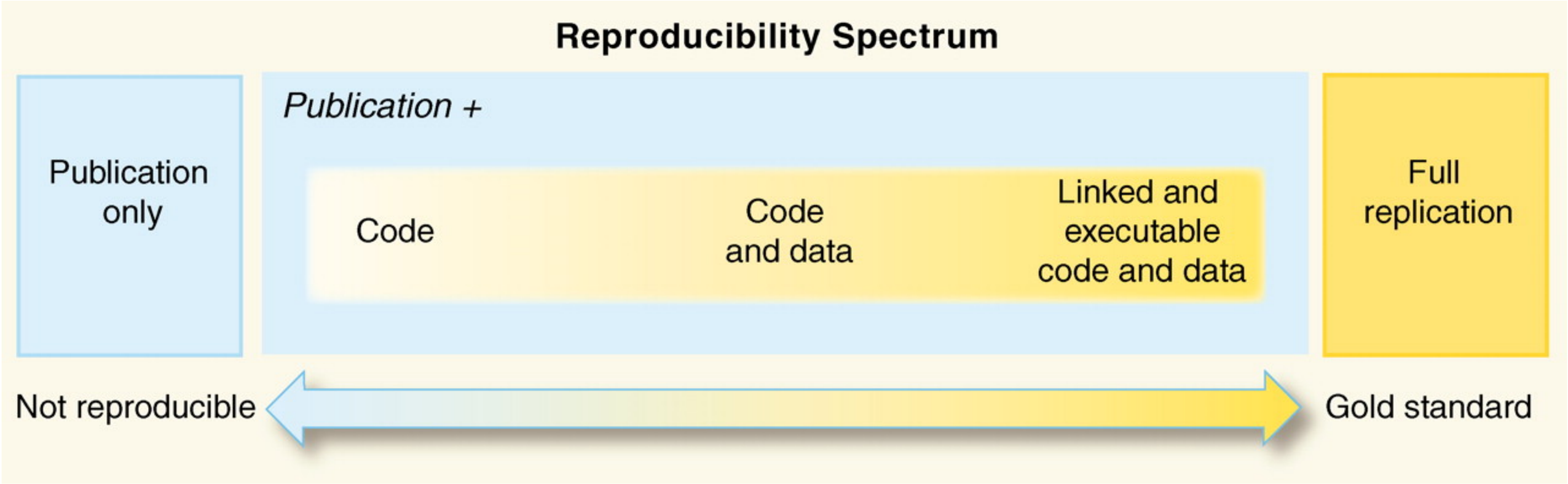
Research Culture

- Royal Society policy project on research culture: https://royalsociety.org/topics-policy/projects/research-culture/
    - There are ongoing concerns around issues such as: research integrity, career paths, permeability between sectors, recognition and reward, diversity, and support for collaboration and interdisciplinarity
- Wellcome key issue and report on research culture: https://wellcome.ac.uk/what-we-do/our-work/research-culture
    - Poor research culture is leading to unhealthy competition, bullying and harassment, and mental health issues

Words that researchers would use to describe research culture. (Wellcome, https://wellcome.ac.uk/reports/what-researchers-think-about-research-culture)



The Turing Way Community and Scriberia, http://doi.org/10.5281/zenodo.3332808



"Computational science has led to exciting new developments, but the nature of the work has exposed limitations in our ability to evaluate published findings. Reproducibility has the potential to serve as a minimum standard for judging scientific claims when full independent replication of a study is not possible."

Peng (2011) https://doi.org/10.1126/science.1213847

Barriers to sharing research software

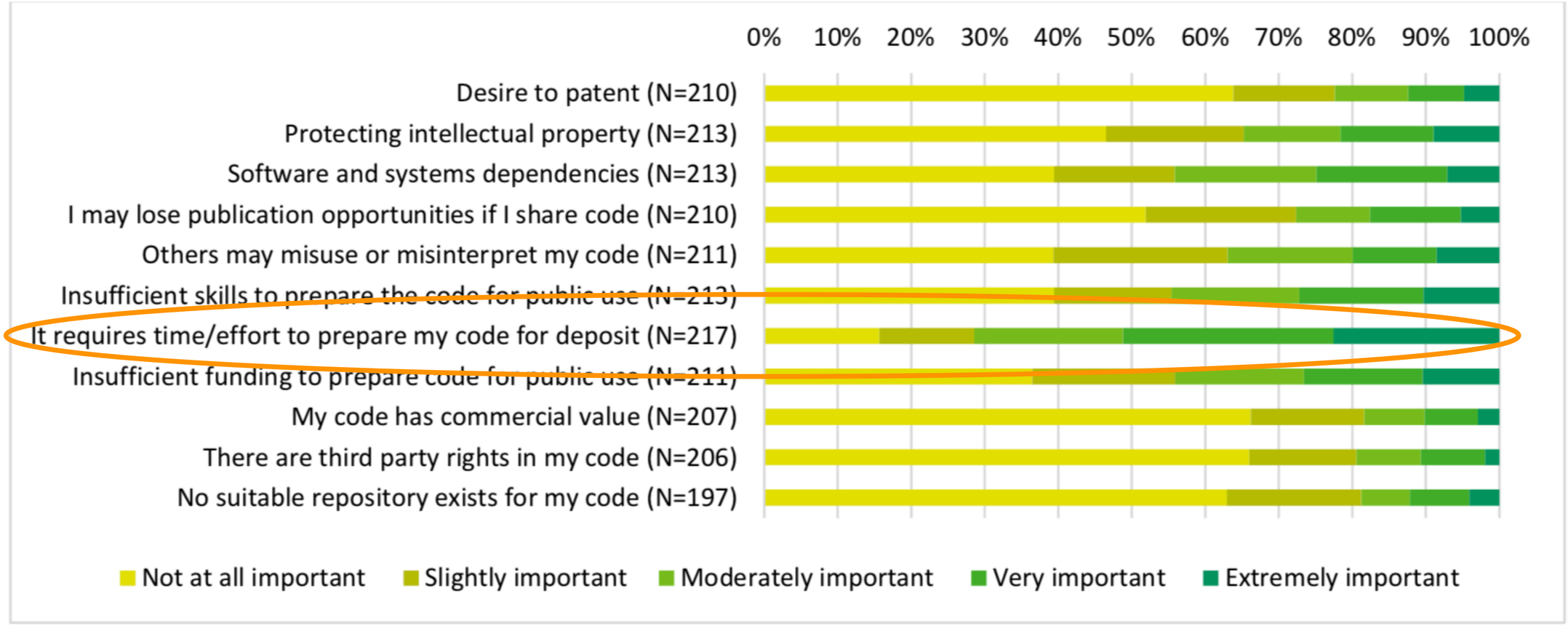
Barriers to Open Research

- Lack of awareness and training
- Cultural inertia and misinformation
- Challenging the establishment
- Follow the status quo to succeed
- Perceived lack of reward
- Not considered for promotion
- Requires additional skills
- Takes time
- Publication bias towards novel findings

Fig: https://doi.org/10.6084/m9.figshare.5558653

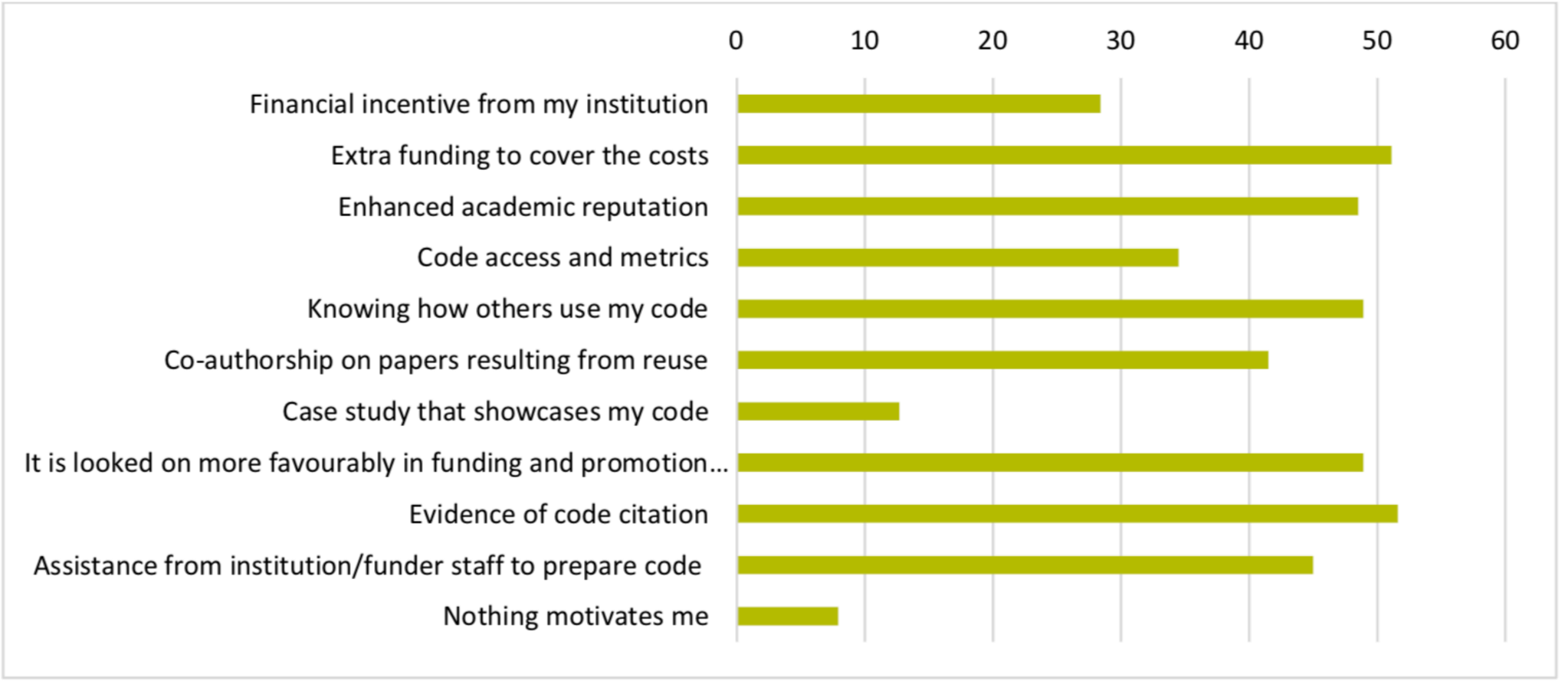
Whitaker (2018) https://doi.org/10.6084/m9.figshare.7140050.v2

FIGURE 25 RESPONDENT EVALUATION OF CODE SHARING BARRIERS



Van den Eynden, Veerle; Knight, Gareth; Vlad, Anca; Radler, Barry; Tenopir, Carol; Leon, David; et al. (2016): Survey of Wellcome researchers and their attitudes to open research. figshare. Journal contribution. https://doi.org/10.6084/m9.figshare.4055448.v1

FIGURE 26 FACTORS THAT WOULD MOTIVATE THE RESPONDENT TO MAKE MORE CODE AVAILABLE, AS PERCENTAGE OF RESPONDENTS (N=229)



Van den Eynden, Veerle; Knight, Gareth; Vlad, Anca; Radler, Barry; Tenopir, Carol; Leon, David; et al. (2016): Survey of Wellcome researchers and their attitudes to open research. figshare. Journal contribution. https://doi.org/10.6084/m9.figshare.4055448.v1

“Give some indication that sharing code is valued when funding decisions are made. Editing code from the state where it works on my computer to where it can be used by everybody takes a huge amount of time. In addition to making the code better / more robust, making it public also requires a significant amount of documentation. There is little credit given for this effort, especially when the code is supporting a specific paper (rather than code for a tool that will be widely used by the community).”

Van den Eynden, Veerle; Knight, Gareth; Vlad, Anca; Radler, Barry; Tenopir, Carol; Leon, David; et al. (2016): Survey of Wellcome researchers and their attitudes to open research. figshare. Journal contribution. https://doi.org/10.6084/m9.figshare.4055448.v1

Benefits to sharing research software

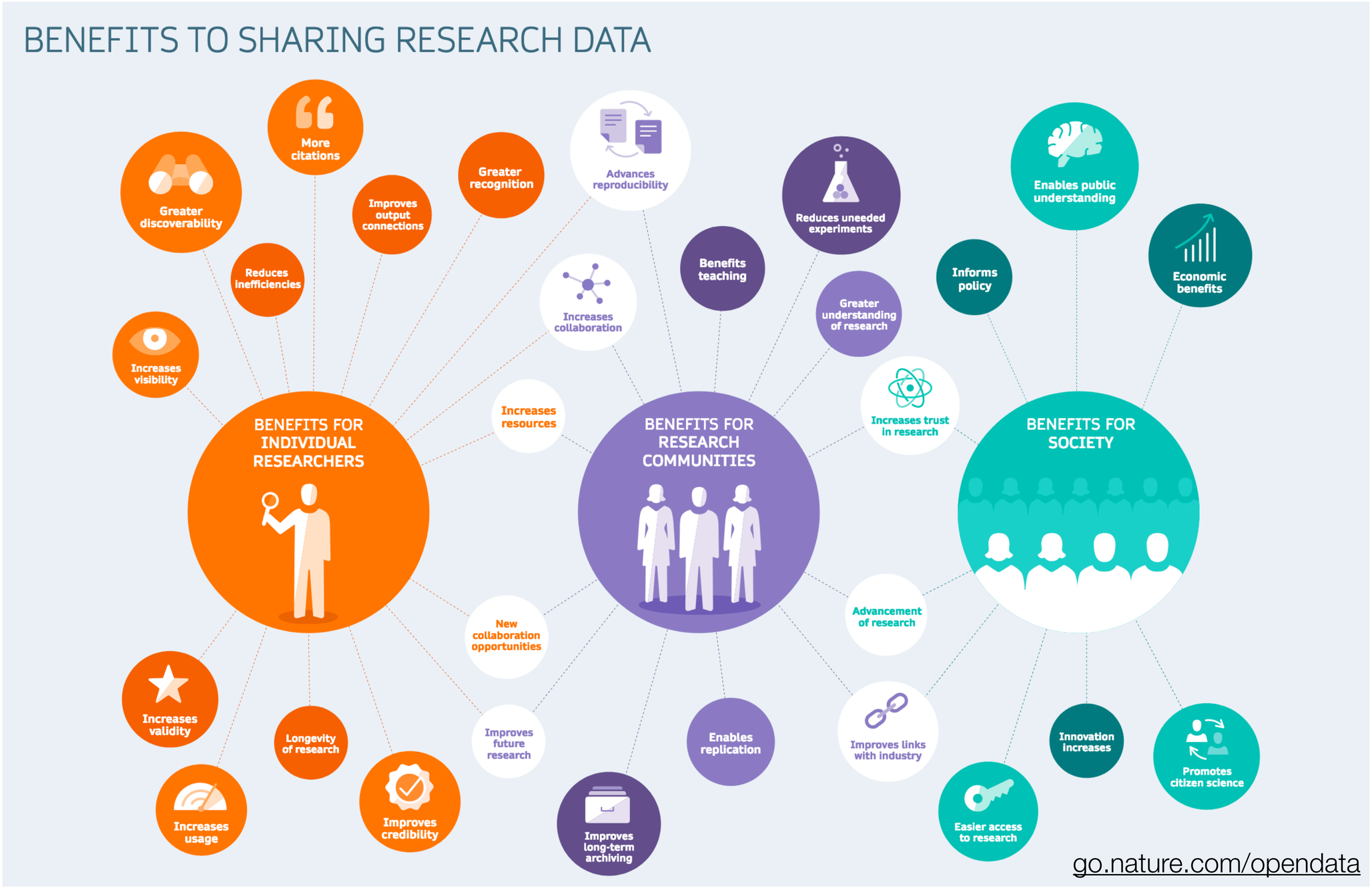
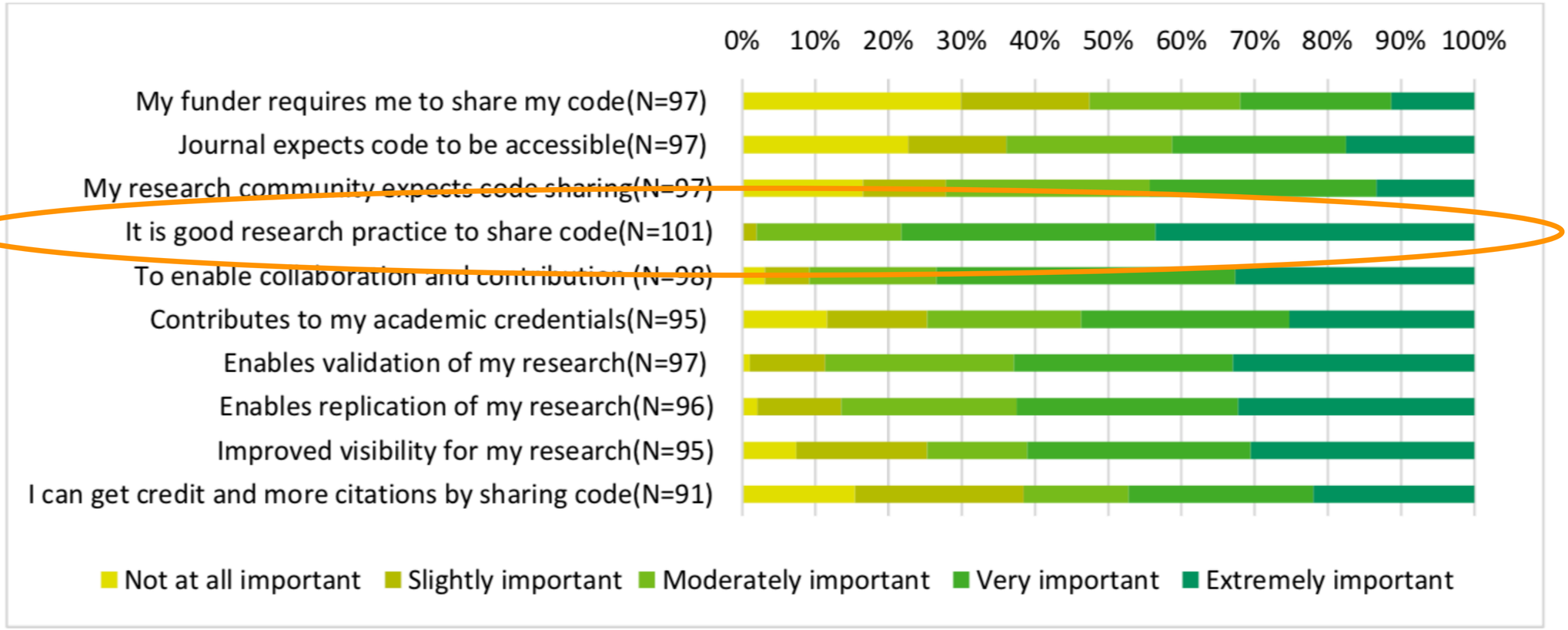
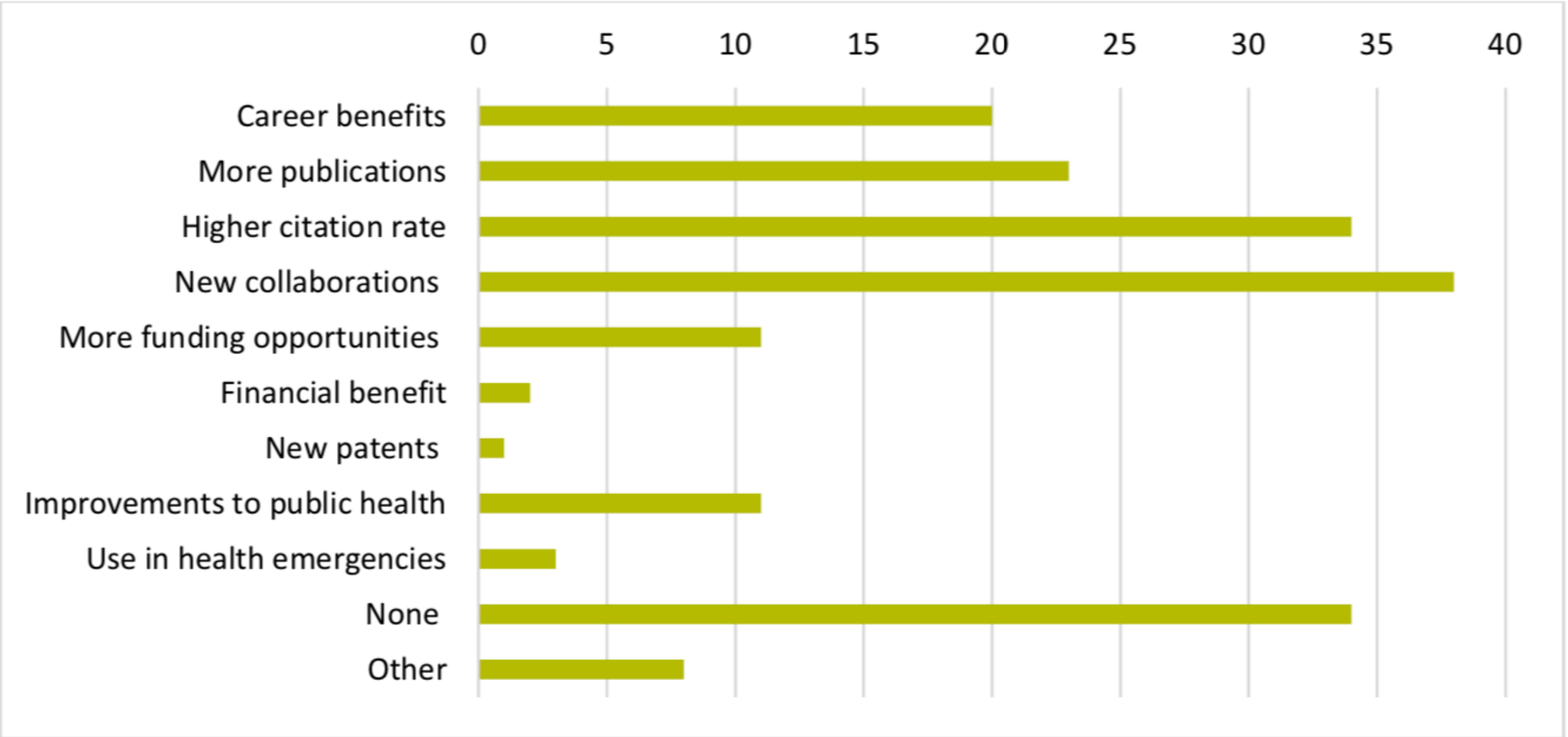

BENEFITS TO SHARING RESEARCH DATA
BENEFITS FOR INDIVIDUAL RESEARCHERS
BENEFITS FOR RESEARCH COMMUNITIES
BENEFITS FOR SOCIETY
More citations
Greater discoverability
Improves output connections
Greater recognition
Reduces inefficiencies
Increases visibility
Increases validity
Increases usage
Longevity of research
Improves credibility
Advances reproducibility
Increases collaboration
Benefits teaching
Reduces uneeded experiments
Greater understanding of research
Increases resources
New collaboration opportunities
Improves future research
Improves long-term archiving
Enables replication
Improves links with industry
Advancement of research
Increases trust in research
Enables public understanding
Informs policy
Economic benefits
Innovation increases
Promotes citizen science
Easier access to research
go.nature.com/opendata

FIGURE 23. REASONS FOR MAKING CODE AVAILABLE IN A REPOSITORY OR OTHER ONLINE FORM



Van den Eynden, Veerle; Knight, Gareth; Vlad, Anca; Radler, Barry; Tenopir, Carol; Leon, David; et al. (2016): Survey of Wellcome researchers and their attitudes to open research. figshare. Journal contribution. https://doi.org/10.6084/m9.figshare.4055448.v1

FIGURE 24. PERCENTAGE OF RESPONDENTS THAT HAVE GAINED PERSONAL BENEFITS BY CODE SHARING (N=100)



Van den Eynden, Veerle; Knight, Gareth; Vlad, Anca; Radler, Barry; Tenopir, Carol; Leon, David; et al. (2016): Survey of Wellcome researchers and their attitudes to open research. figshare. Journal contribution. https://doi.org/10.6084/m9.figshare.4055448.v1

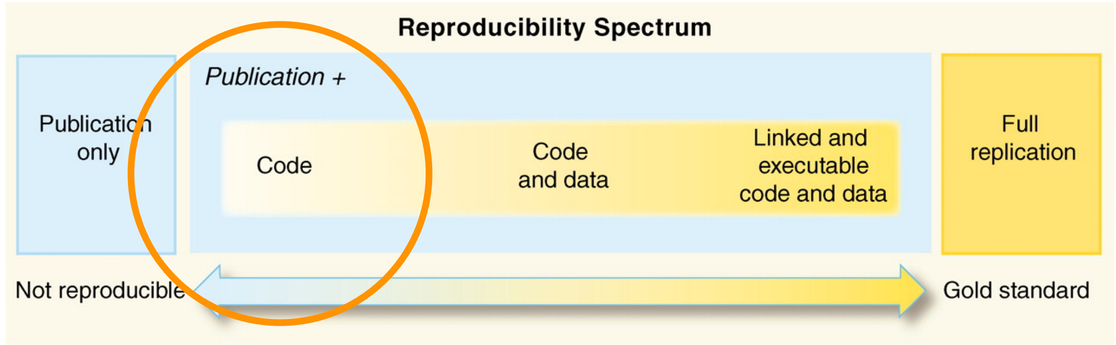
“It is about mind-sets and culture: An unsung part of open software are its communities that promote and enable a more inclusive, kinder culture.”

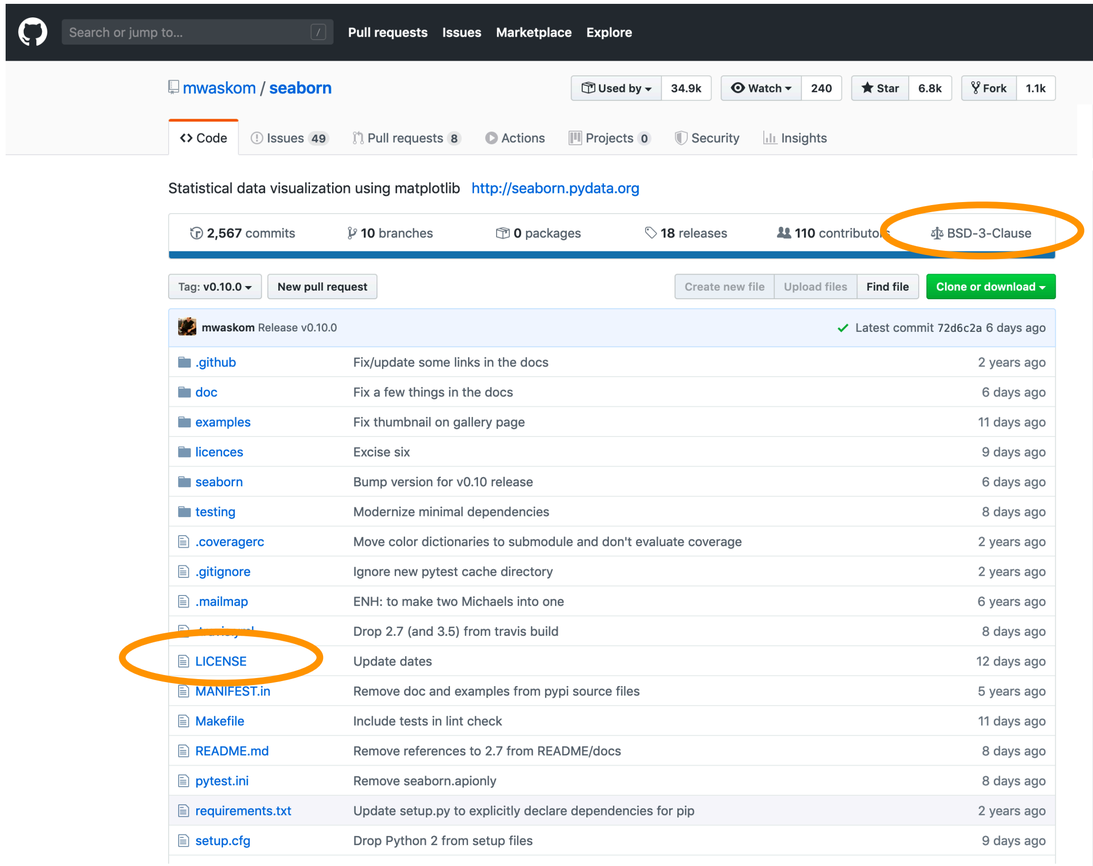
– Julia Stewart Lowndes, Open Software Means Kinder Science
https://blogs.scientificamerican.com/observations/open-software-means-kinder-science

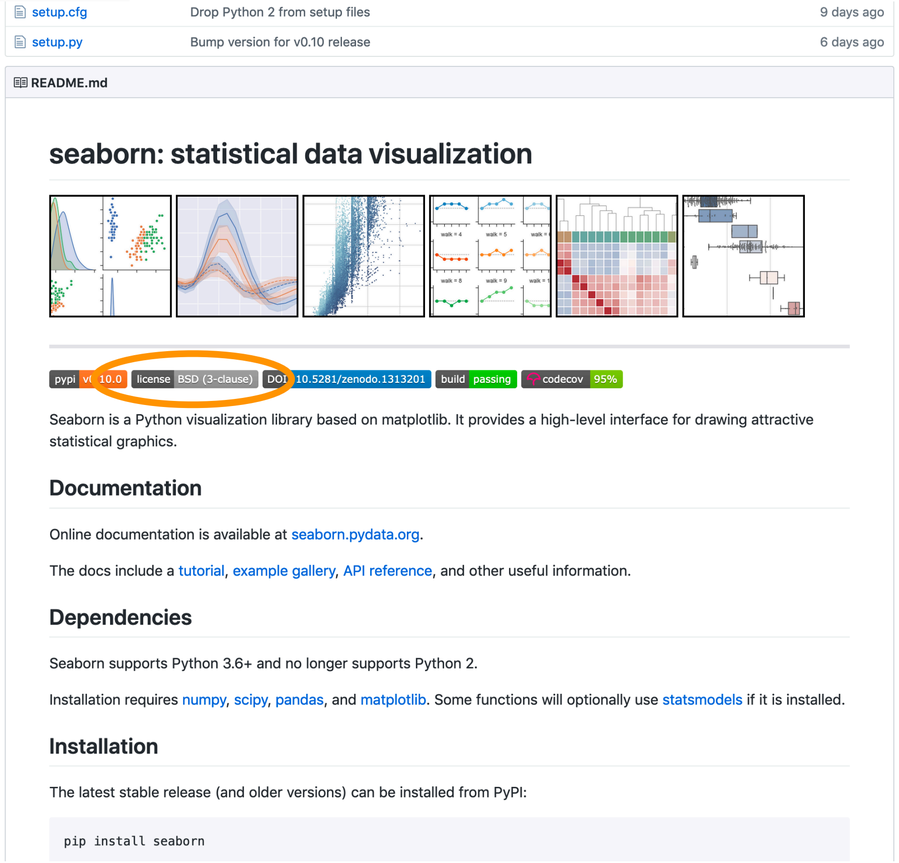
How to share your research software and get credit

Share software in repositories such as GitHub, Bitbucket & GitLab

- Version control
- Facilitates open, collaborative & reproducible science/code/ research
- Online portfolio & webpage for your work



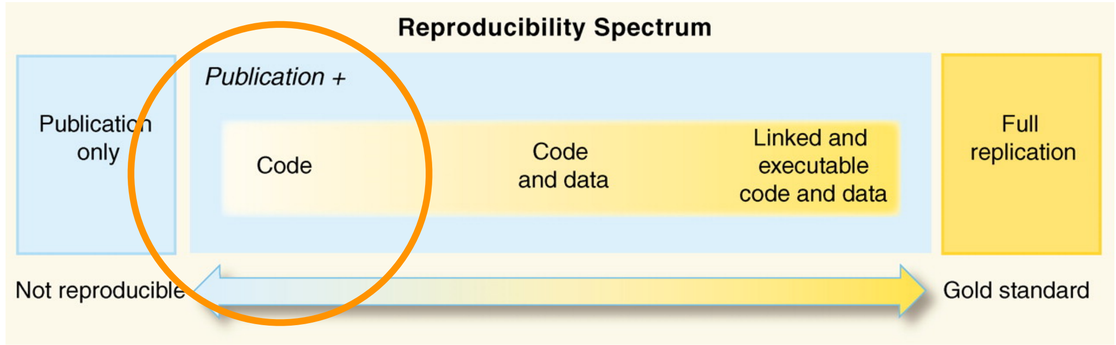




https://github.com/mwaskom/seaborn/tree/v0.10.0

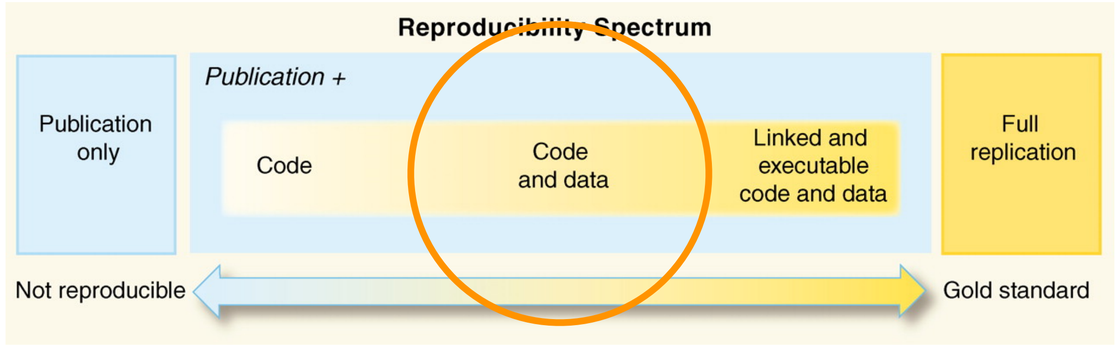
Choose an open source license to allow adoption & reuse

- GitHub guide on choosing an open source license: https://choosealicense.com/
- SSI Guide on Choosing an open-source licence: https://www.software.ac.uk/resources/guides/adopting-open-source-licence
- A Quick Guide to Software Licensing for the Scientist-Programmer: https://doi.org/10.1371/journal.pcbi.1002598
- tl;drLegal summarises software licenses in plain English and has developed a tool to help you manage your open source licenses: https://tldrlegal.com/



https://choosealicense.com/

Make software citable by archiving in repositories such as Zenodo & Figshare



Assigns a distinct persistent identifier: Digital Object Identifier (DOI)

zenodo Search Upload Communities Log in Sign up

January 24, 2020 Software Open Access

mwaskom/seaborn: v0.10.0 (January 2020)

Michael Waskom; Olga Botvinnik; Joel Ostblom; Saulius Lukauskas; Paul Hobson; MaozGelbart; David C Gemperline; Tom Augspurger; Yaroslav Halchenko; John B. Cole; Jordi Warmenhoven; Julian de Ruiter; Cameron Pye; Stephan Hoyer; Jake Vanderplas; Santi Villalba; Gero Kunter; Eric Quintero; Pete Bachant; Marcel Martin; Kyle Meyer; Corban Swain; Alistair Miles; Thomas Brunner; Drew O'Kane; Tal Yarkoni; Mike Lee Williams; Constantine Evans

This is a major update that is being released simultaneously with version 0.9.1. It has all of the same features (and bugs!) as 0.9.1, but there are important changes to the dependencies.

Most notably, all support for Python 2 has now been dropped. Support for Python 3.5 has also been dropped. Seaborn is now strictly compatible with Python 3.6+.

Minimally supported versions of the dependent PyData libraries have also been increased, in some cases substantially. While seaborn has tended to be very conservative about maintaining compatibility with older dependencies, this was causing increasing pain during development. At the same time, these libraries are now much easier to install. Going forward, seaborn will likely stay close to the Numpy community guidelines for version support.

This release also removes a few previously-deprecated features:

- The `tsplot` function and `seaborn.timeseries` module have been removed. Recall that `tsplot` was replaced with `lineplot`.
- The `seaborn.apionly` entry-point has been removed.
- The `seaborn.linearmodels` module (previously renamed to `seaborn.regression`) has been removed.

Looking forward

Now that seaborn is a Python 3 library, it can take advantage of keyword-only arguments. It is likely that future versions will introduce this syntax, potentially in a breaking way. For guidance, most seaborn functions have a signature that looks like

```
func(x, y, ..., data=None, **kwargs)
```

where the `**kwargs` are specified in the function. Going forward it will likely be necessary to specify `data` and all subsequent arguments with an explicit `key=value` mapping. This style has long been used throughout the documentation, and the formal requirement will not be introduced until at least the next major release. Adding this feature will make it possible to enhance some older functions with more modern capabilities (e.g., adding a native `hue` semantic within functions like `jointplot` and `regplot`).

Preview

seaborn-v0.10.0.zip

mwaskom-seaborn-e071450

13,737 views 406 downloads

See more details...

Available in GitHub

Indexed in OpenAIRE

Publication date: January 24, 2020

DOI: DOI 10.5281/zenodo.3629446

Related identifiers: Supplement to https://github.com/mwaskom/seaborn/tree/v0.10.0

Communities: Zenodo

License (for files): Other (Open)

will make it possible to enhance some older functions with more modern capabilities (e.g., adding a native `hue` semantic within functions like `jointplot` and `regplot`).

Preview

seaborn-v0.10.0.zip

mwaskom-seaborn-e071450	
.coveragerc	142 Bytes
.github	
CONTRIBUTING.md	1.3 kB
.gitignore	109 Bytes
.mailmap	145 Bytes
.travis.yml	1.4 kB
LICENSE	1.5 kB
MANIFEST.in	87 Bytes
Makefile	343 Bytes
README.md	3.7 kB
doc	
.gitignore	256 Bytes
Makefile	6.0 kB
_static	
copybutton.js	2.7 kB
favicon.ico	270.4 kB
stvle.css	1.9 kB

Files (334.5 kB)

Name	Size	
mwaskom/seaborn-v0.10.0.zip	334.5 kB	Preview Download

md5:c0871c123b926edd32e2a91bafdd9f37

Beta Citations 0

Show only: Literature (0) Dataset (0) Software (0) Unknown (0) Citations to this version

Search

No citations.

/tree/v0.10.0

Communities: Zenodo

License (for files): Other (Open)

Versions

Version v0.10.0 10.5281/zenodo.3629446	Jan 24, 2020
Version v0.9.1 10.5281/zenodo.3629445	Jan 24, 2020
Version v0.9.0 10.5281/zenodo.1313201	Jul 16, 2018
Version v0.8.1 10.5281/zenodo.883859	Sep 3, 2017
Version v0.8.0 10.5281/zenodo.824567	Jul 8, 2017

View all 10 versions

Cite all versions? You can cite all versions by using the DOI 10.5281/zenodo.592845. This DOI represents all versions, and will always resolve to the latest one. Read more.

Share

Cite as

Michael Waskom, Olga Botvinnik, Joel Ostblom, Saulius Lukauskas, Paul Hobson, MaozGelbart, ... Constantine Evans. (2020, January 24). mwaskom/seaborn: v0.10.0 (January 2020) (Version v0.10.0). Zenodo. http://doi.org/10.5281/zenodo.3629446

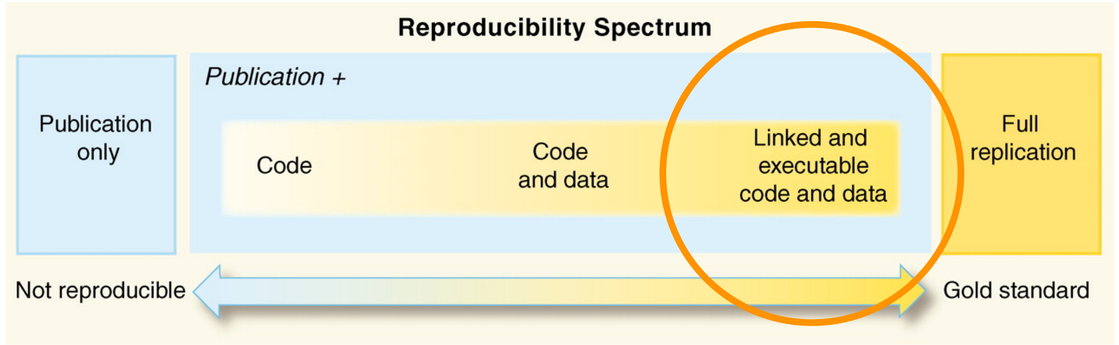
Start typing a citation style...

https://zenodo.org/record/3629446
http://doi.org/10.5281/zenodo.3629446

Dr. Rachael Ainsworth, SSI @rachaelevelyn @OpenResLDN #ResearchSoftware

https://doi.org/10.6084/m9.figshare.11762121

Make it as easy as possible for others to use your software



- Open Notebooks display code that can be executed independently and interactively, facilitating transparency in the analysis of data, reproducibility, and documentation of the entire workflow
  - Jupyter and RStudio
- Containers can be used to package entire scientific workflows, software and libraries, and even data, eliminating the pain of having to install missing dependencies
  - Docker and Singularity
- Binder launches a Git repository containing open notebooks in an executable environment, making your code immediately reproducible by anyone, anywhere

Publish your Research Software

- Full article in peer reviewed domain specific journals (example: Astropy article in *Astronomy & Astrophysics*, https://doi.org/10.1051/0004-6361/201322068)
- *Journal of Open Research Software* (JORS) features peer reviewed Software Metapapers describing research software with high reuse potential
- *Journal of Open Source Software* (JOSS) is a developer friendly academic journal with a formal peer review process that is designed to improve the quality of the software submitted


A&A 558, A33 (2013)
DOI: 10.1051/0004-6361/201322068
© ESO 2013

Astronomy & Astrophysics

Astropy: A community Python package for astronomy

The Astropy Collaboration, Thomas P. Robitaille[1], Erik J. Tollerud[2,3], Perry Greenfield[4], Michael Droettboom[4], Erik Bray[4], Tom Aldcroft[5], Matt Davis[4], Adam Ginsburg[6], Adrian M. Price-Whelan[7], Wolfgang E. Kerzendorf[8], Alexander Conley[8], Neil Crighton[1], Kyle Barbary[9], Demitri Muna[10], Henry Ferguson[4], Frédéric Grollier[12], Madhura M. Parikh[11], Prasanth H. Nair[12], Hans M. Günther[5], Christoph Deil[13], Julien Woillez[14], Simon Conseil[15], Roban Kramer[16], James E. H. Turner[17], Leo Singer[18], Ryan Fox[12], Benjamin A. Weaver[19], Victor Zabalza[13], Zachary I. Edwards[20], K. Azalee Bostroem[4], D. J. Burke[5], Andrew R. Casey[21], Steven M. Crawford[22], Nadia Dencheva[4], Justin Ely[4], Tim Jenness[23,24], Kathleen Labrie[25], Pey Lian Lim[4], Francesco Pierfederici[4], Andrew Pontzen[26,27], Andy Ptak[28], Brian Refsdal[5], Mathieu Servillat[29,5], and Ole Streicher[30]

[1] Max-Planck-Institut für Astronomie, Königstuhl 17, 69117 Heidelberg, Germany
e-mail: robitaille@mpia.de
[2] Department of Astronomy, Yale University, PO Box 208101, New Haven, CT 065
[3] Hubble Fellow
[4] Space Telescope Science Institute, 3700 San Martin Drive, Baltimore, MD 21218,
[5] Harvard-Smithsonian Center for Astrophysics, 60 Garden Street, Cambridge, MA
[6] Center for Astrophysics and Space Astronomy, University of Colorado, Boulder, C
[7] Department of Astronomy, Columbia University, Pupin Hall, 550W 120th St., New
[8] Department of Astronomy and Astrophysics, University of Toronto, 50 Saint Geor
[9] Argonne National Laboratory, High Energy Physics Division, 9700 South Cass Av
[10] Department of Astronomy, Ohio State University, Columbus, OH 43210, USA
[11] S.V.National Institute of Technology, 395007 Surat., India
[12] Independent developer
[13] Max-Planck-Institut für Kernphysik, PO Box 103980, 69029 Heidelberg, Germany
[14] European Southern Observatory, Karl-Schwarzschild-Str. 2, 85748 Garching bei M
[15] Laboratoire d'Astrophysique de Marseille, OAMP, Université Aix-Marseille et CN
[16] ETH Zürich, Institute for Astronomy, Wolfgang-Pauli-Strasse 27, Building HIT, FI
[17] Gemini Observatory, Casilla 603, La Serena, Chile
[18] LIGO Laboratory, California Institute of Technology, 1200 E. California Blvd., Pas
[19] Center for Cosmology and Particle Physics, New York University, New York, NY
[20] Department of Physics and Astronomy, Louisiana State University, Nicholson Hall
[21] Research School of Astronomy and Astrophysics, Australian National University, Weston Creek ACT 2611, Australia
[22] SAAO, PO Box 9, Observatory 7935, 7925 Cape Town, South Africa
[23] Joint Astronomy Centre, 660 N. A'ohōkū Place, Hilo, HI 96720, USA
[24] Department of Astronomy, Cornell University, Ithaca, NY 14853, USA
[25] Gemini Observatory, 670 N. A'ohōkū Place, Hilo, HI 96720, USA
[26] Oxford Astrophysics, Denys Wilkinson Building, Keble Road, Oxford OX1 3RH,
[27] Department of Physics and Astronomy, University College London, London WC1
[28] NASA Goddard Space Flight Center, X-ray Astrophysics Lab Code 662, Greenbel
[29] Laboratoire AIM, CEA Saclay, Bât. 709, 91191 Gif-sur-Yvette, France
[30] Leibniz-Institut für Astrophysik Potsdam (AIP), An der Sternwarte 16, 14482 Pots

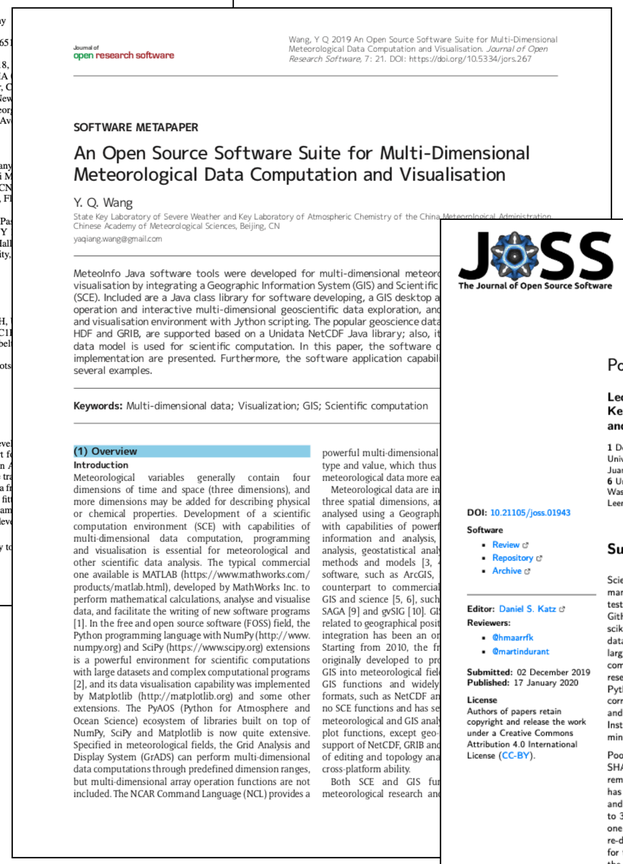
Received 12 June 2013 / Accepted 23 July 2013

ABSTRACT

We present the first public version (v0.2) of the open-source and community-deve
provides core astronomy-related functionality to the community, including support f
image transport system (FITS) files, Virtual Observatory (VO) tables, and common A
conversions, physical constants specific to astronomy, celestial coordinate and time tr
support, generalized containers for representing gridded as well as tabular data, and a fr
conversions. Significant functionality is under active development, such as a model fit
aperture and point spread function (PSF) photometry tools. The core development team
to the current code base, and we encourage anyone interested to participate in the dev

Key words. methods: data analysis – methods: miscellaneous – virtual observatory t

Article published by EDP Sciences


Journal of open research software

Wang, Y Q 2019 An Open Source Software Suite for Multi-Dimensional Meteorological Data Computation and Visualisation. *Journal of Open Research Software*, 7: 21. DOI: https://doi.org/10.5334/jors.267

SOFTWARE METAPAPER

An Open Source Software Suite for Multi-Dimensional Meteorological Data Computation and Visualisation

Y. Q. Wang
State Key Laboratory of Severe Weather and Key Laboratory of Atmospheric Chemistry of the China Meteorological Administration, Chinese Academy of Meteorological Sciences, Beijing, CN
yaqiang.wang@gmail.com

MeteoInfo Java software tools were developed for multi-dimensional meteoro
visualisation by integrating a Geographic Information System (GIS) and Scientific
(SCE). Included are a Java class library for software developing, a GIS desktop a
operation and interactive multi-dimensional geoscientific data exploration, and
and visualisation environment with Jython scripting. The popular geoscience data
HDF and GRIB, are supported based on a Unidata NetCDF Java library; also, it
data model is used for scientific computation. In this paper, the software c
implementation are presented. Furthermore, the software application capabil
several examples.

Keywords: Multi-dimensional data; Visualization; GIS; Scientific computation

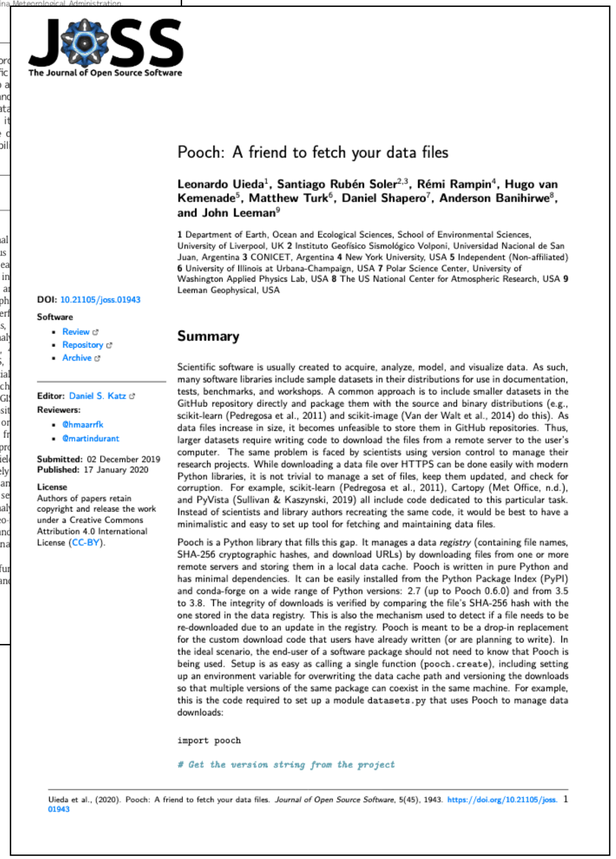
(1) Overview

Introduction

Meteorological variables generally contain four dimensions of time and space (three dimensions), and more dimensions may be added for describing physical or chemical properties. Development of a scientific computation environment (SCE) with capabilities of multi-dimensional data computation, programming and visualisation is essential for meteorological and other scientific data analysis. The typical commercial one available is MATLAB (https://www.mathworks.com/products/matlab.html), developed by MathWorks Inc. to perform mathematical calculations, analyse and visualise data, and facilitate the writing of new software programs [1]. In the free and open source software (FOSS) field, the Python programming language with NumPy (http://www.numpy.org) and SciPy (https://www.scipy.org) extensions is a powerful environment for scientific computations with large datasets and complex computational programs [2], and its data visualisation capability was implemented by Matplotlib (http://matplotlib.org) and some other extensions. The PyAOS (Python for Atmosphere and Ocean Science) ecosystem of libraries built on top of NumPy, SciPy and Matplotlib is now quite extensive. Specified in meteorological fields, the Grid Analysis and Display System (GrADS) can perform multi-dimensional data computations through predefined dimension ranges, but multi-dimensional array operation functions are not included. The NCAR Command Language (NCL) provides a powerful multi-dimensional type and value, which thus meteorological data more ea

Meteorological data are in three spatial dimensions, a analysed using a Geograph with capabilities of powerf information and analysis, analysis, geostatistical anal methods and models [3, software, such as ArcGIS, counterpart to commercial GIS and science [5, 6], such SAGA [9] and gvSIG [10]. GI related to geographical posit integration has been an or Starting from 2010, the fr originally developed to pro GIS into meteorological fiel GIS functions and widely formats, such as NetCDF ar no SCE functions and has se meteorological and GIS anal plot functions, except geo- support of NetCDF, GRIB an of editing and topology ana cross-platform ability.

Both SCE and GIS fu meteorological research an


JOSS The Journal of Open Source Software

Pooch: A friend to fetch your data files

Leonardo Uieda[1], Santiago Rubén Soler[2,3], Rémi Rampin[4], Hugo van Kemenade[5], Matthew Turk[6], Daniel Shapero[7], Anderson Banihirwe[8], and John Leeman[9]

1 Department of Earth, Ocean and Ecological Sciences, School of Environmental Sciences, University of Liverpool, UK **2** Instituto Geofísico Sismológico Volponi, Universidad Nacional de San Juan, Argentina **3** CONICET, Argentina **4** New York University, USA **5** Independent (Non-affiliated) **6** University of Illinois at Urbana-Champaign, USA **7** Polar Science Center, University of Washington Applied Physics Lab, USA **8** The US National Center for Atmospheric Research, USA **9** Leeman Geophysical, USA

DOI: 10.21105/joss.01943

Software
- Review
- Repository
- Archive

Editor: Daniel S. Katz
Reviewers:
- @hmaarrfk
- @martindurant

Submitted: 02 December 2019
Published: 17 January 2020

License
Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License (CC-BY).

Summary

Scientific software is usually created to acquire, analyze, model, and visualize data. As such, many software libraries include sample datasets in their distributions for use in documentation, tests, benchmarks, and workshops. A common approach is to include smaller datasets in the GitHub repository directly and package them with the source and binary distributions (e.g., scikit-learn (Pedregosa et al., 2011) and scikit-image (Van der Walt et al., 2014) do this). As data files increase in size, it becomes unfeasible to store them in GitHub repositories. Thus, larger datasets require writing code to download the files from a remote server to the user's computer. The same problem is faced by scientists using version control to manage their research projects. While downloading a data file over HTTPS can be done easily with modern Python libraries, it is not trivial to manage a set of files, keep them updated, and check for corruption. For example, scikit-learn (Pedregosa et al., 2011), Cartopy (Met Office, n.d.), and PyVista (Sullivan & Kaszynski, 2019) all include code dedicated to this particular task. Instead of scientists and library authors recreating the same code, it would be best to have a minimalistic and easy to set up tool for fetching and maintaining data files.

Pooch is a Python library that fills this gap. It manages a data *registry* (containing file names, SHA-256 cryptographic hashes, and download URLs) by downloading files from one or more remote servers and storing them in a local data cache. Pooch is written in pure Python and has minimal dependencies. It can be easily installed from the Python Package Index (PyPI) and conda-forge on a wide range of Python versions: 2.7 (up to Pooch 0.6.0) and from 3.5 to 3.8. The integrity of downloads is verified by comparing the file's SHA-256 hash with the one stored in the data registry. This is also the mechanism used to detect if a file needs to be re-downloaded due to an update in the registry. Pooch is meant to be a drop-in replacement for the custom download code that users have already written (or are planning to write). In the ideal scenario, the end-user of a software package should not need to know that Pooch is being used. Setup is as easy as calling a single function (`pooch.create`), including setting up an environment variable for overwriting the data cache path and versioning the downloads so that multiple versions of the same package can coexist in the same machine. For example, this is the code required to set up a module `datasets.py` that uses Pooch to manage data downloads:

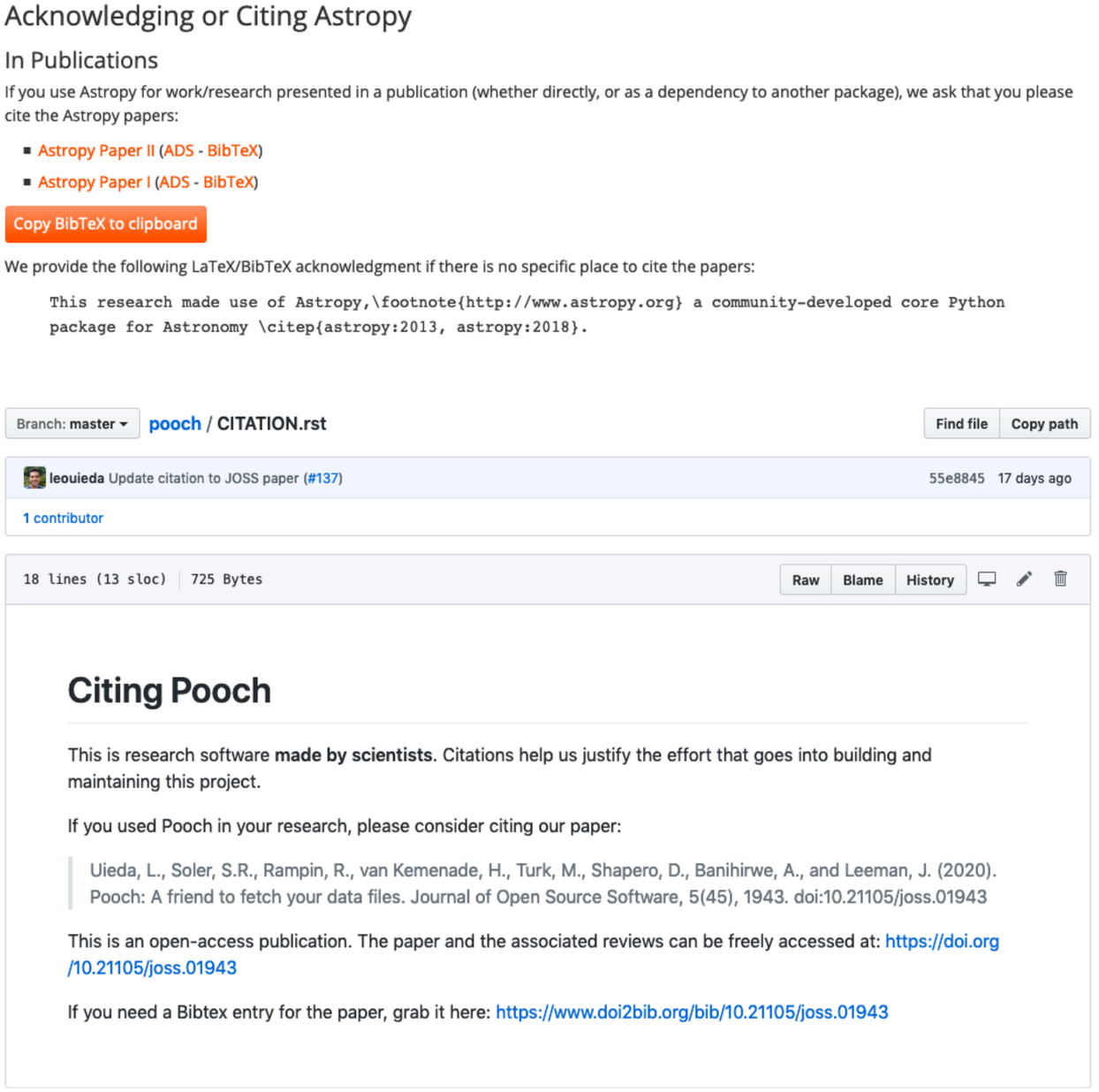
```
import pooch

# Get the version string from the project
```

Uieda et al., (2020). Pooch: A friend to fetch your data files. *Journal of Open Source Software*, 5(45), 1943. https://doi.org/10.21105/joss.01943 1

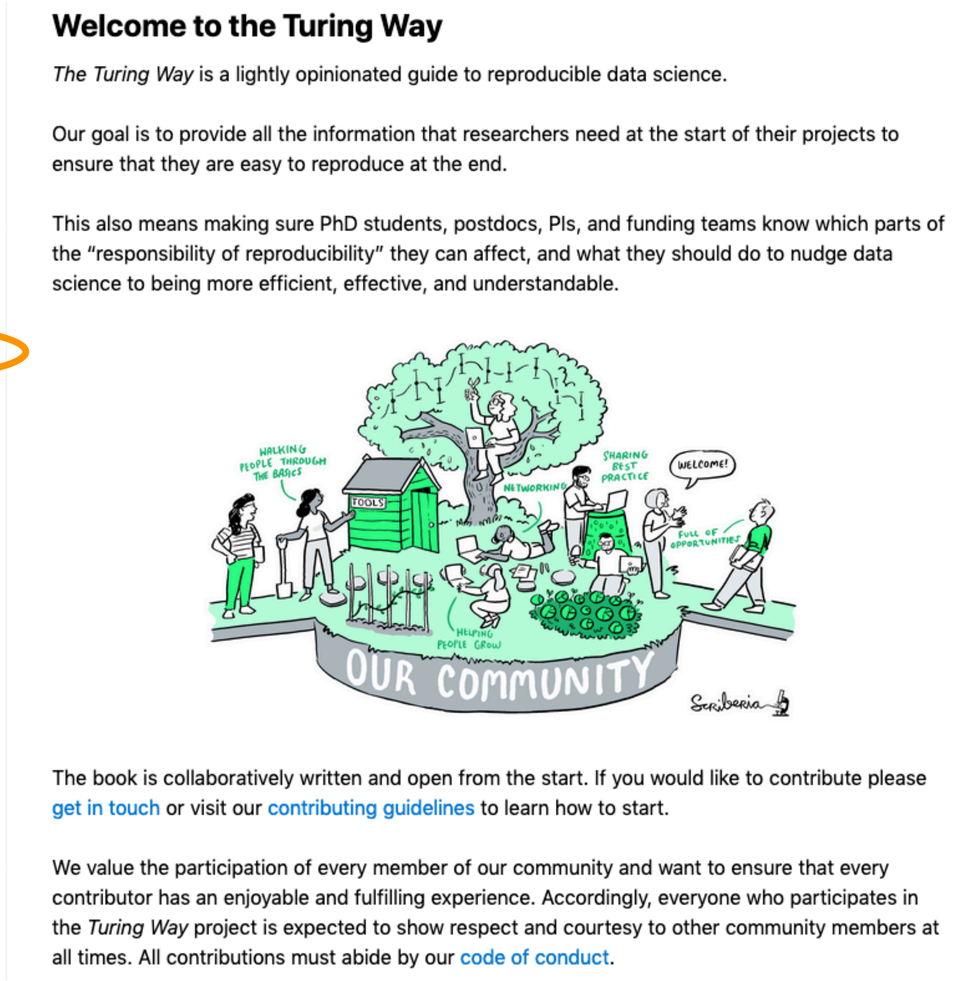
Make it easy for people to cite you

Include instructions in your software documentation (README or CITATION file), whether that includes citing a published article about your software or an archived version of your software in a digital repository like Zenodo or Figshare



The Turing Way Project

- Project led by Kirstie Whitaker at The Alan Turing Institute to make reproducible research "too easy not to do"
- In short: *The Turing Way* encompasses a handbook, community, collaboration, workshops and training
- Team of researchers, research software engineers, librarians and YOU!
- Demonstrates open and transparent project management and communication with future users, as it is openly developed at our GitHub repository: https://github.com/alan-turing-institute/the-turing-way



Handbook at: https://the-turing-way.netlify.com

Takeaways

1. Make your software available in a stable, version-controlled repository
2. Choose an open source license to allow adoption and reuse
3. Assign a distinct persistent identifier (DOI) for each version release
4. Include instructions and examples of how to cite your software in its documentation

- Check out *The Turing Way* - a handbook on reproducible research/data science openly developed at https://github.com/alan-turing-institute/the-turing-way/
- TEDx talk: Research Culture is Broken; Open Science can [help] fix it https://youtu.be/c-bemNZ-IqA
- More of my slide decks on Open Research can be found on Figshare: https://figshare.com/authors/Rachael_Ainsworth/4824354