Meta-Psychology, 2019, vol 3, MP.2018.892
https://doi.org/10.15626/MP.2018.892
Article type: Original Article
Published under the CC-BY4.0 license

Open data and materials: N/A
Open and reproducible analysis: N/A
Open reviews and editorial process: Yes
Preregistration: N/A

Edited by: Rickard Carlsson
Reviewed by: Nick Brown, Jack Davis, Nicholas A Coles
All supplementary files can be accessed at OSF:
https://doi.org/10.17605/OSF.IO/PS5RU

Computational Reproducibility via Containers in Psychology

April Clyburne-Sherin
Independent Consultant

Xu Fei
Code Ocean

Seth Ariel Green
Code Ocean

Abstract

Scientific progress relies on the replication and reuse of research. Recent studies suggest, however, that sharing code and data does not suffice for computational reproducibility —defined as the ability of researchers to reproduce "particular analysis outcomes from the same data set using the same code and software" (Fidler and Wilcox, 2018). To date, creating long-term computationally reproducible code has been technically challenging and time-consuming. This tutorial introduces Code Ocean, a cloud-based computational reproducibility platform that attempts to solve these problems. It does this by adapting software engineering tools, such as Docker, for easier use by scientists and scientific audiences. In this article, we first outline arguments for the importance of computational reproducibility, as well as some reasons why this is a nontrivial problem for researchers. We then provide a step-by-step guide to getting started with containers in research using Code Ocean. (Disclaimer: the authors all worked for Code Ocean at the time of this article's writing.)
Keywords: computational reproduciblity, social psychology, containers, Docker, Code Ocean

Introduction: The need for computational reproducibility

Stodden (2014) distinguishes between three forms of reproducibility: statistical, empirical, and computational. In psychology, statistical reproducibility, encompassing transparency about analytic choices and strategies, has received sustained attention (Simmons, Nelson, and Simonsohn, 2011; Grange et al., 2018; Gelman and Loken, 2014; Morey and Lakens, 2016). Likewise, empirical reproducibility —providing enough information about procedures to enable high-fidelity independent replication —has been a high-profile issue in light of work by the Center for Open Science (Collaboration, 2015; Nosek and Lakens, 2014). Computational reproducibility, by contrast, has been less of a focus.

Kitzes (2017) describes a research project as being "computationally reproducible"[1] when "a second investigator (including you in the future) can recreate the final reported results of the project, including key quantitative findings, tables, and figures, given only a set of files and written instructions."[2] Computational repro-

[1]Note that we use the term 'reproduction' to refer to recreating given results using given data, and replication to refer to analyzing new data. This is broadly in line with the definitions used by, among others, Peng (2011), Claerbout (2011), and Donoho, Maleki, Rahman, Shahram, and Stodden (2008), but some disciplines use the terms differently; for an overview, see Marwick, Rokem, and Staneva (2017) and Barba (2018).

[2]Fidler and Wilcox (2018) distinguish between two senses

ducibility facilitates the accumulation of knowledge by enabling researchers to assess the analytic choices, assumptions, and implementations that led to a set of results; it also enables testing the robustness of methods to alternate specifications. Hardwicke et al. (2018) call this form of reproducibility a "minimum level of credibility" (p. 2). Moreover, as Donoho (2017) argues, preparing one's work for reproducible publication provides "benefits to authors. Working from the beginning with a plan for sharing code and data leads to higher quality work, and ensures that authors can access their own former work, and those of their co-authors, students and postdocs" (p. 760). Because computations are central to modern research in the social sciences, their reproducibility, or lack thereof, warrants ministration and attention within the broader open science movement and the scientific community.

Many psychology journals (Lindsay, 2017; Jonas and Cesario, 2015) address reproducibility through strong policies on sharing data, code, and materials. The Society for Personality and Social Psychology's 'Task Force on Publication and Research Practices" (Funder et al., 2014) advises authors to make "available research materials necessary" to reproduce statistical results, and to adhere "to SPSP's data sharing policy" (p. 3). The American Psychological Association's ethics policy (section 8.14) asks that "psychologists do not withhold the data on which their conclusions are based from other competent professionals who seek to verify the substantive claims through reanalysis" (Association, 2012). Many journals in the field require that authors sign off on this policy (e.g., Cooper, 2013).

The challenge of computational reproducibility

For two reasons, however, such policies do not suffice for computational reproducibility. First, data and code that are available "upon request" may turn out to be unavailable when actually requested (Stodden, Seiler, and Ma, 2018; Wicherts, Borsboom, Kats, and Molenaar, 2006; Vanpaemel, Vermorgen, Deriemaecker, and Storms, 2015; Wood, Müller, and Brown, 2018). Second, code and data that are publicly available do not necessarily yield the results one sees in the accompanying paper. This is due to a number of technical challenges. Dependencies —the packages and libraries that a researcher's code relies on —change over time, often in ways that produce errors (Bogart, Kästner, and Herbsleb, 2015) or change outputs. Software versions are not always perfectly recorded (Barba, 2016), which makes reconstruction of the original computational environment difficult. While there are many useful guides to best practices for scientific research (Wilson et al., 2017; Sandve, Nekrutenko, Taylor, and Hovig, 2013), adopting them is an investment of scarce time and attention. More prosaically, differences between scientists' machines can be nontrivial, and memory or storage limitations can halt a reproduction effort (Deelman and Chervenak, 2008).

As a result, publicly available code and data are often not computationally reproducible. An example comes from the journal *Cognition*. Following the journal's adoption of a mandatory data sharing policy, Hardwicke et al. (2018) attempted to reproduce the results of 35 articles for which they had code and data, and were able to do so, without author assistance, for just 11 papers; a further 11 were reproducible with author assistance, and the remaining 13 were not reproducible "despite author assistance" (p. 3). While the authors are careful to note that these issues do not appear to "seriously impact" original conclusions, nevertheless, "suboptimal data curation, unclear analysis specification, and reporting errors can impede computational reproducibility" (p. 3).[3]

Rates of reproducibility appear similar in other disciplines. At the *Quarterly Journal of Political Science*, editors found that from "September 2012 to November 2015. . . 14 of the 24 empirical papers subject to in-house review were found to have discrepancies between the results generated by authors' own code and those in their written manuscripts" (Eubank, 2016, p. 273). In sociology, after working closely with authors, Liu and Salganik (2019) were able to reproduce the results of seven of 12 papers for a special issue of the journal *Socius*. In development economics, Wood et al. (2018) looked at 109 papers and found only 29 to be "push button replicable" (the authors' synonym for computationally reproducible). In general, how much information suffices for reproduction becomes clear only when it is attempted.

Literate programming, a valuable paradigm for documentation and explanation, does not necessarily address these issues. Woodbridge (2017) recounts attempting to identify a sample of Jupyter notebooks (Kluyver et al., 2016) mentioned in PubMed Central, thinking that reproduction "would simply involve

---

of the term: "direct (reproducing particular analysis outcomes from the same data set using the same code and software)...[and] conceptual (analyzing the same raw data set with alternative approaches, different models or statistical frameworks)." This article primarily concerns direct reproducibility.

[3]The authors also note that "assessments in the "reproducible" category took between 2-4 person hours, and assessments in the "reproducible with author assistance" and "not fully reproducible, despite author assistance" categories took between 5-25 person hours" (p. 28).

searching the text of each article for a notebook reference, then downloading and executing it. . . It turned out that this was hopelessly naive." Dependencies were frequently unmentioned and were not always included with notebooks; troubleshooting language and tool specific issues required expertise and hindered portability; and notebooks would often "assume the availability of non-Python software being available on the local system," but such software "may not be freely available."

In sum, as Silver (2017) notes, lab-built tools

> rarely come ready to run. . . Much of the software requires additional tools and libraries, which the user may not have installed. Even if users can get the software to work, differences in computational environments, such as the installed versions of the tools it depends on, can subtly alter performance, affecting reproducibility. (p. 173)

A welcome development: containers

Meanwhile, tools designed by engineers engineers to share code are available, but are often befuddling to non-specialists. Chamberlain and Schommer (2014) note that virtual machines "have serious drawbacks," including the difficulty of use "without a high level of systems administration knowledge" and requiring "a lot of storage space, which makes them onerous to share" (p.1).

One major advance for sharing code is container software. Containers reduce complexity, Silver (2017) writes, "by packaging the key elements of the computational environment needed to run the desired software. . . into a lightweight, virtual box. . . [T]hey make the software much easier to use, and the results easier to reproduce" (p. 174).

Docker

A container platform called Docker is rising in popularity in some academic fields (Merkel, 2014; Boettiger, 2015). Docker's core virtues include:

1. a rich and growing ecosystem of supporting tools and environments, such as Rocker (Boettiger and Eddelbuettel, 2017), a repository of Docker images[4] specifically for R users, and BiocImageBuilder for Bioconductor-based builds (Almugbel et al., 2017);

2. ease of use, relative to other container and virtual machine technology;

3. an open-source code base, allowing for adaptation (Hung, Kristiyanto, Lee, and Yeung, 2016) and integration with existing academic software (Grüning et al., 2016; Almugbel et al., 2017);

4. relatively lightweight installation, because a Docker container "does not replicate the full operating system, only the libraries and binaries of the application being virtualized" (Chamberlain and Schommer, 2014); and

5. compatibility with any programming language that can be installed on Linux.[5]

Adoption of container technology like Docker in psychology, however, remains scant.[6] A few explanations come to mind. The first is simply lack of awareness. The second is lack of incentives, as journals increasingly require the sharing of code and data but not of a full-fledged computational environment. The third is that Docker, though easier to use than many other software engineering tools, requires familiarity with the command line and dependency management. These skills take time and effort to learn, are not part of the standard curriculum for training researchers (Boettiger, 2015), and are not self-evidently a worthwhile investment when weighing opportunity costs.

Code Ocean: customizing container technology for researchers

Code Ocean attempts to address these issues. It is a platform for creating, running, and collaborating on research code. It allows scientists to package code, data, results, metadata, and a computational environment into a single compendium —called a 'compute capsule,'[7] or simply 'capsule' for short —whose results can be reproduced by anyone who presses a 'Run' button. It does so by providing a simple-to-use interface for configuring computational environments, getting code up and running online, and publishing final results. Each published capsule is assigned a unique, persistent identifier in the form of a digital object identifier (DOI) and can be embedded either directly into the text of an article or its landing page. The platform hopes to make code accompanying research articles reproducible in perpetuity[8] by

[4] A Docker image is the executable package containing all necessary prerequisites for a software application to run.

[5] For a more thorough overview of Docker's capabilities and scientific use cases, see Boettiger (2015).

[6] A search on 23 April 2019 of http://www.apa.org, for instance for the words "Docker container" yielded zero matches.

[7] Thank you to Christopher Honey for the term.

[8] For Code Ocean's preservation plan, see https://help.codeocean.com/faq/code-oceans-preservation-plan.

containing all analyses within stable and portable computational environments.

The remainder of this article will illustrate these features by walking through a capsule called "The contact hypothesis re-evaluated: code and data", available at https://doi.org/10.24433/CO.4024382.v6 or https://codeocean.com/capsule/8235972/tree/v6. This capsule reproduces the results of a July 2018 article published in *Behavioural Public Policy* (Paluck, Green, and Green, 2018).[9] (It may help to open up the capsule in a new tab or window while reading.)

Reuse without downloading or technical setup

Code Ocean allows reuse without installing anything locally. Figure 1 shows the default view for this capsule. Code is in the top left, data are in the bottom left, and a set of published results are on the right (in the 'Reproducibility pane'). Readers can view and edit selected files in the center pane. A published capsule's code, data, and results are open-access; they can be viewed and downloaded by all, with or without a Code Ocean account.[10] The 'Reproducible Run' button reproduces all results in their entirety. This is possible by dint of two things: a run script, and a fully configured computational environment.

The 'run' script (also called the 'master script'), visible in Figure 1 as the code file with the flag icon, is a script that executes each analysis script in its proper order. Authors can designate different files as their entrypoints by selecting 'Set as File to Run'. All capsules must have a run script to be published.

Clicking on 'environment' will give the user a snapshot of the computational environment (Figure 2). This tab offers a number of common package managers, customized for each base environment, and a postInstall script wherein you can download and install software that isn't currently available through a package manager, or precisely specify an order of operations (Figure 3). Whenever possible, package versions are labeled and held static to ensure transparency and long-term stability of computations. For published capsules, environments are pre-configured by authors and do not need to be altered by readers to reproduce results.

Configured to support research workflows

Code Ocean offers support for any open-source language that can be installed on Linux, and also the proprietary languages Stata and MATLAB (Figure 4). This particular capsule runs Stata and R code in sequence (Figure 2). Each language comes with pre-configured base environments for common use cases; readers can also start from a blank slate, with no scientific programming languages installed.

Metadata and preservation

Code Ocean asks authors to provide sufficient metadata on published capsules to facilitate intelligibility. Attaching rich metadata to a capsule encourages citation and signals that published code is a first-class research object.

In addition to metadata provided by authors, published capsules are automatically provided with a DOI and citation information (Figure 5). Metadata about an associated publication establishes a compute capsule as a 'version of record' of code and data to support published findings.

Cloud Workstations

By default, pressing 'Reproducible Run' on Code Ocean runs the main script from top to bottom. Readers may also wish to run code line by line (or snippet by snippet) iteratively. Cloud Workstations support this. Following instructions provided on https://help.codeocean.com/en/articles/2366255-cloud-workstations-an-overview, Authors and readers can currently run Terminal, Jupyter, JupyterLab, R Shiny, and Rstudio workstations, with more options planned. This particular capsule has Rstudio preinstalled and ready to launch (Figure 6).

Exporting capsules for local reproduction

For any capsule readers have access to, including all public capsules, they can download code, data, metadata, and a formula for the computational environment, as well as instructions on reproducing results locally. Local reproduction will require some familiarity with Docker, as well as all applicable software licenses (Figure 7).

Share or embed a capsule

Finally, Code Ocean lets readers easily share published capsules. Capsules can be posted to social media, or as interactive widgets embedded directly into the text of articles, websites, or blogs (Figure 8).

---

[9] Note that one author (Seth Green) is the author of this capsule and a co-author of the accompanying BPP article.

[10] Running code requires an account to prevent abuse of available computational resources, which include GPUs. Authors who sign up with academic email addresses receive 10 hours of runtime per month and 20 GB of storage by default. Code Ocean's current policy is to provide authors with any and all resources they need to publish capsules on the platform. For more details, see https://codeocean.com/pricing.

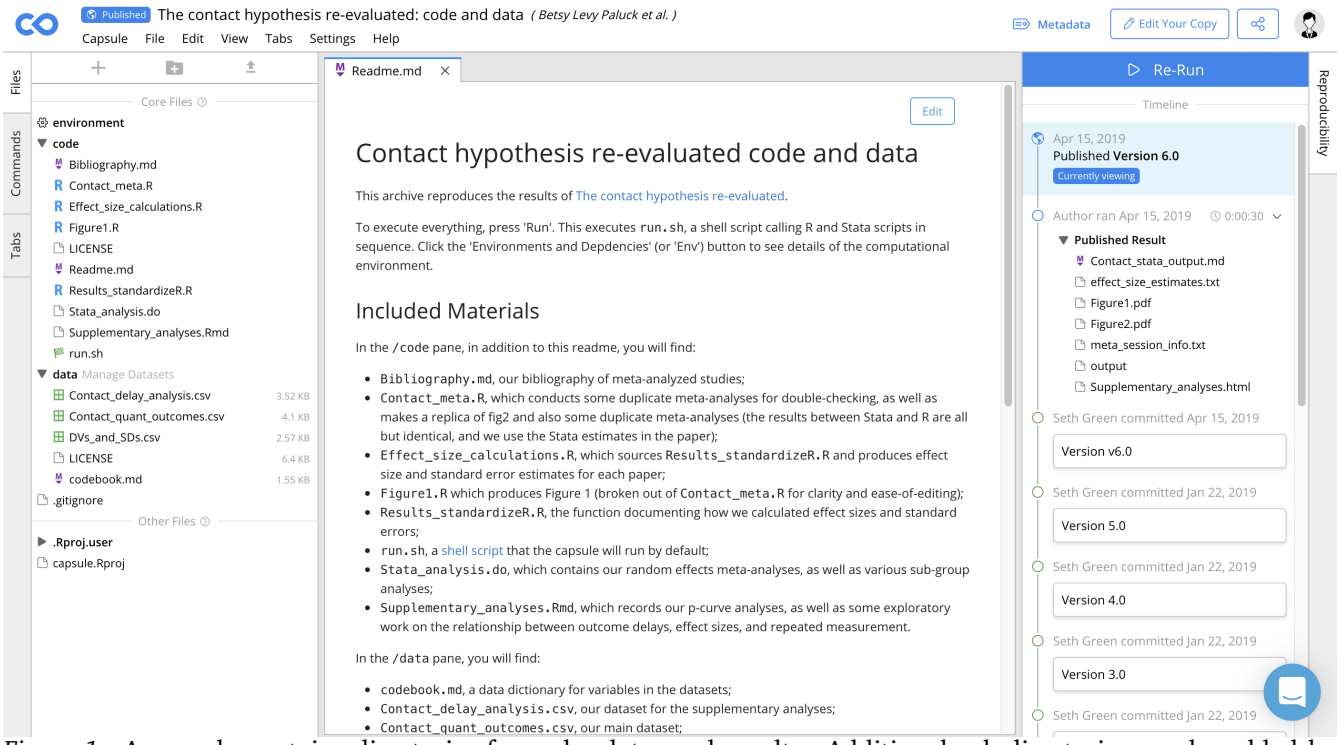

Figure 1. A capsule contains directories for code, data, and results. Additional sub-directories can be added by authors. Note: all figures are available as standalone files at https://doi.org/10.17605/OSF.IO/S8MZ4.

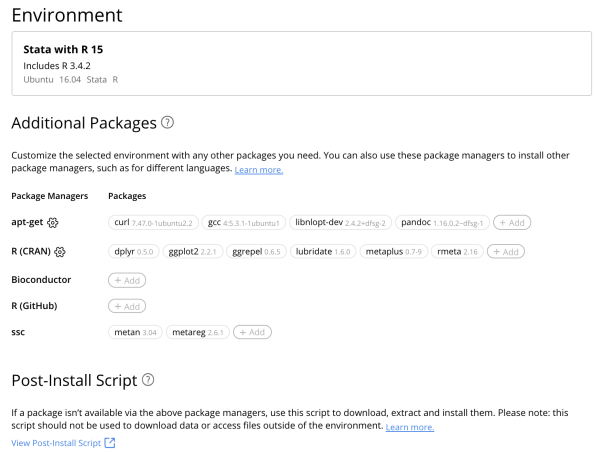

Figure 2. This capsule requires packages from apt-get, CRAN, and SSC.

Figure 3. Software can also be added via a custom post-install script.

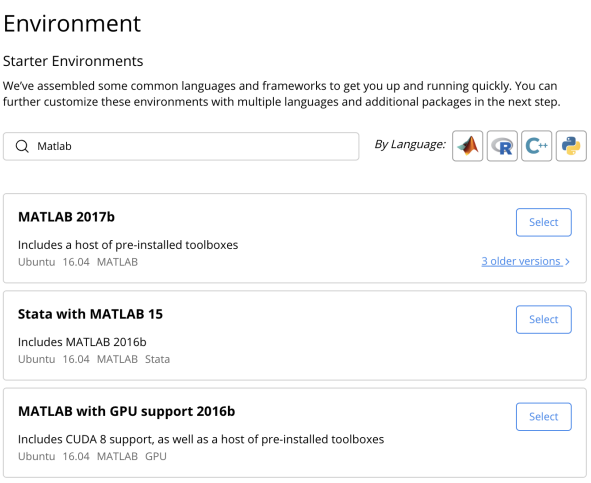

Figure 4. When creating a new compute capsule, an author can select environments with pre-installed languages and language-specific installers, or start from a blank slate ('Ubuntu Linux'). This figure displays available MATLAB environments.

Conclusion: Answering the call to make reproducibility tools simpler

In the context of discussing Docker, Boettiger (2015) writes that:

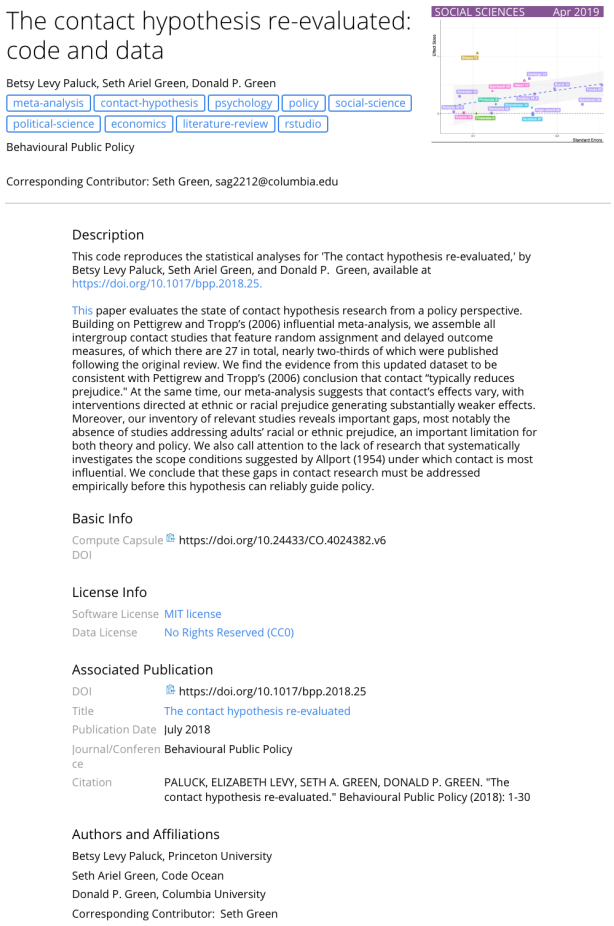

Figure 5. An excerpt from a capsule's metadata. A DOI and citation data are automatically added to any published capsule.

A technical solution, no matter how elegant, will be of little practical use for reproducible research unless it is both easy to use and adapt to the existing workflow patterns of practicing domain researchers . . . Another researcher may be less likely to build on existing work if it can only be done by using a particular workflow system or monolithic software platform with which they are unfamiliar. Likewise, a user is more likely to make their own computational environment available for reuse if it does not involve a significant added effort in packaging and documenting. Perhaps the most important feature of a reproducible research tool is that it be easy to learn and fit relatively seamlessly into existing workflow patterns of domain researcher. (pp. 4-5)

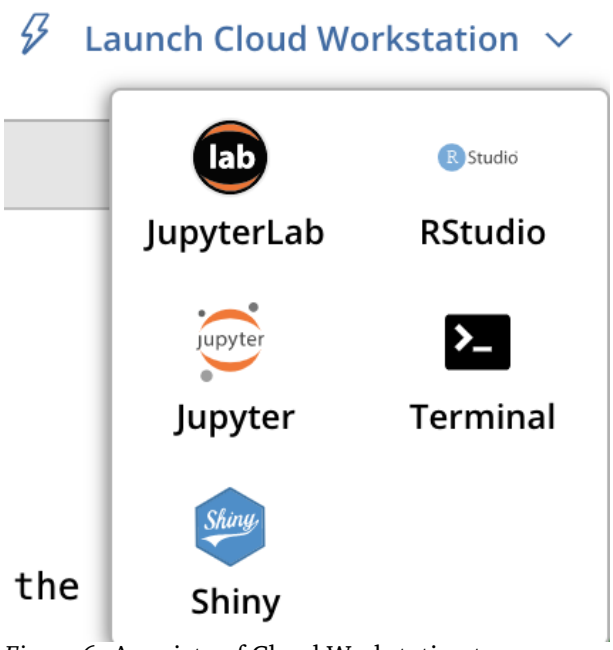

Figure 6. A variety of Cloud Workstation types are available.

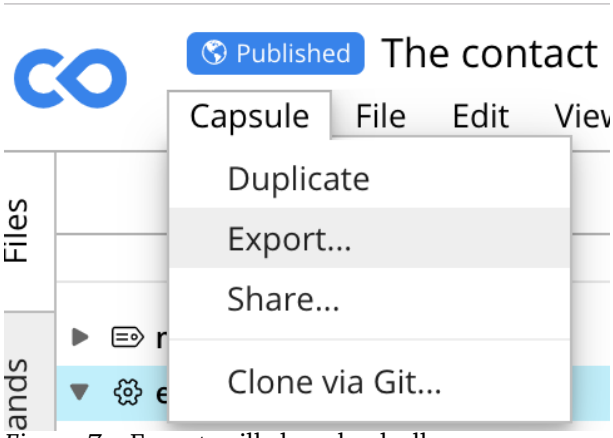

Figure 7. Export will download all necessary components for reproducing results locally.

We believe that containers are an important advance in this direction, and hope that Code Ocean, by building on this technology and adapting it specifically to the needs of researchers, helps enable a fully reproducible workflow that is "easy to use and adapt" to existing research habits.

Open Science Practices

Because this article is a tutorial, there are no relevant data, materials, analyses or preregistration(s) to be shared.

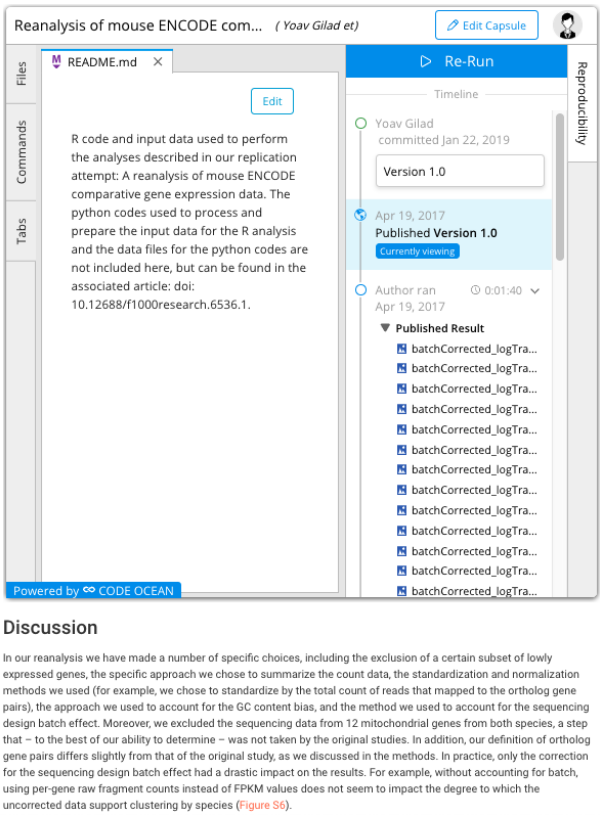

Figure 8. Compute capsules can be embedded into the text of articles so that analyses can be reviewed and assessed in context. This capsule appears within the text of Gilad and Mizrahi-Man (2015).

Author Note

April Clyburne-Sherin is an independent consultant on open science tools, methods, training, and community stewardship; as of August 2019, she was Director of Scientific Outreach at Code Ocean. Xu Fei is Outreach Scientist at Code Ocean. Seth Green is Developer Advocate at Code Ocean. Correspondence concerning this article can be addressed to xufei at codeocean dot com and seth at codeocean dot com.

We would like to thank Shahar Zaks, Christopher Honey, Rickard Carlsson, our reviewers Nick Brown and Jack Davis for their feedback, and Nicholas A. Coles for his helpful comments on our PsyArXiv preprint.

Author Contributions

April Clyburne-Sherin contributed conceptualization, investigation, visualization, and writing (original draft, reviewing and editing). Xu Fei contributed conceptualization, visualization and writing (reviewing and editing). Seth Ariel Green contributed conceptualization, investigation, visualization, and writing (original draft, review and editing. Authorship order accords to alphabetical order of last names.

Conflict of Interest

All three authors worked at Code Ocean during the writing of this paper.

Funding

The authors did not receive any grants for writing this paper.

References

Almugbel, R., Hung, L.-H., Hu, J., Almutairy, A., Ortogero, N., Tamta, Y., & Yeung, K. Y. (2017). Reproducible bioconductor workflows using browser-based interactive notebooks and containers. *Journal of the American Medical Informatics Association*, *25*(1), 4–12.

Association, A. P. (2012). Ethics code updates to the publication manual. *Retrieved September*, *1*, 2012.

Barba, L. A. (2016). The hard road to reproducibility. *Science*, *354*(6308), 142–142.

Barba, L. A. (2018). Terminologies for reproducible research. *CoRR*, *abs/1802.03311*.

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

Boettiger, C. & Eddelbuettel, D. (2017). An introduction to rocker: docker containers for r. *arXiv preprint arXiv:1710.03675*.

Bogart, C., Kästner, C., & Herbsleb, J. (2015). When it breaks, it breaks. In *Proc. of the workshop on software support for collaborative and global software engineering (scgse)*.

Chamberlain, R. & Schommer, J. (2014). Using docker to support reproducible research. *DOI: https://doi. org/10.6084/m9. figshare*, *1101910*.

Claerbout, J. (2011). Reproducible computational research: a history of hurdles, mostly overcome. *technical report*.

Collaboration, O. S. (2015). Estimating the reproducibility of psychological science. *Science*, *349*(6251), aac4716.

Cooper, J. (2013). On fraud, deceit and ethics. *Journal of Experimental Social Psychology*, *2*(49), 314.

Deelman, E. & Chervenak, A. (2008). Data management challenges of data-intensive scientific workflows. In *Cluster computing and the grid, 2008. ccgrid'08. 8th ieee international symposium on* (pp. 687–692). IEEE.

Donoho, D. (2017). 50 years of data science. *Journal of Computational and Graphical Statistics*, *26*(4), 745–766.

Donoho, D., Maleki, A., Rahman, I., Shahram, M., & Stodden, V. (2008). 15 years of reproducible research in computational harmonic analysis. *Technical report*.

Eubank, N. (2016). Lessons from a decade of replications at the quarterly journal of political science. *PS: Political Science & Politics*, *49*(2), 273–276.

Fidler, F. & Wilcox, J. (2018). Reproducibility of scientific results. In E. N. Zalta (Ed.), *The stanford encyclopedia of philosophy* (Winter 2018). Metaphysics Research Lab, Stanford University.

Funder, D. C., Levine, J. M., Mackie, D. M., Morf, C. C., Sansone, C., Vazire, S., & West, S. G. (2014). Improving the dependability of research in personality and social psychology: recommendations for research and educational practice. *Personality and Social Psychology Review*, *18*(1), 3–12.

Gelman, A. & Loken, E. (2014). The statistical crisis in science. *American Scientist*, *102*(6), 460.

Gilad, Y. & Mizrahi-Man, O. (2015). A reanalysis of mouse encode comparative gene expression data. *F1000Research*, *4*.

Grange, J., Lakens, D., Adolfi, F., Albers, C., Anvari, F., Apps, M., ... Benning, S., et al. (2018). Justify your alpha. *Nature Human Behavior*.

Grüning, B., Rasche, E., Rebolledo-Jaramillo, B., Eberhart, C., Houwaart, T., Chilton, J., ... Nekrutenko, A. (2016). Enhancing pre-defined workflows with ad hoc analytics using galaxy, docker and jupyter. *bioRxiv*, 075457.

Hardwicke, T. E., Mathur, M. B., MacDonald, K., Nilsonne, G., Banks, G. C., Kidwell, M. C., ... Henry Tessler, M., et al. (2018). Data availability, reusability, and analytic reproducibility: evaluating the impact of a mandatory open data policy at the journal cognition. *Royal Society open science*, *5*(8), 180448.

Hung, L.-H., Kristiyanto, D., Lee, S. B., & Yeung, K. Y. (2016). Guidock: using docker containers with a common graphics user interface to address the reproducibility of research. *PloS one*, *11*(4), e0152686.

Jonas, K. J. & Cesario, J. (2015). Guidelines for authors. Retrieved from http://www.tandf.co.uk/journals/authors/rrsp-submission-guidelines.pdf

Kitzes, J. (2017). Introduction. In J. Kitzes, D. Turek, & F. Deniz (Eds.), *The practice of reproducible research: case studies and lessons from the data-intensive sciences*. University of California Press.

Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B. E., Bussonnier, M., Frederic, J., ... Corlay, S., et al. (2016). Jupyter notebooks-a publishing format for reproducible computational workflows. In *Elpub* (pp. 87–90).

Lindsay, D. S. (2017). Sharing data and materials in psychological science. SAGE Publications Sage CA: Los Angeles, CA.

Liu, D. & Salganik, M. (2019). Successes and struggles with computational reproducibility: lessons from the fragile families challenge. *SocArXiv*.

Marwick, B., Rokem, A., & Staneva, V. (2017). Assessing reproducibility. In J. Kitzes, D. Turek, & F. Deniz (Eds.), *The practice of reproducible research: case studies and lessons from the data-intensive sciences*. Univ of California Press.

Merkel, D. (2014). Docker: lightweight linux containers for consistent development and deployment. *Linux Journal*, *2014*(239), 2.

Morey, R. D. & Lakens, D. (2016). Why most of psychology is statistically unfalsifiable. Submitted.

Nosek, B. A. & Lakens, D. (2014). Registered reports. Hogrefe Publishing.

Paluck, E. L., Green, S. A., & Green, D. P. (2018). The contact hypothesis re-evaluated. *Behavioural Public Policy*, 1–30.

Peng, R. D. (2011). Reproducible research in computational science. *Science*, *334*(6060), 1226–1227.

Sandve, G. K., Nekrutenko, A., Taylor, J., & Hovig, E. (2013). Ten simple rules for reproducible computational research. *PLoS computational biology*, *9*(10), e1003285.

Silver, A. (2017). Software simplified. *Nature*, *546*(7656), 173–174.

Simmons, J. P., Nelson, L. D., & Simonsohn, U. (2011). False-positive psychology: undisclosed flexibility in data collection and analysis allows presenting anything as significant. *Psychological science*, *22*(11), 1359–1366.

Stodden, V. (2014). What scientific idea is ready for retirement. *Edge*.

Stodden, V., Seiler, J., & Ma, Z. (2018). An empirical analysis of journal policy effectiveness for computational reproducibility. *Proceedings of the National Academy of Sciences*, *115*(11), 2584–2589.

Vanpaemel, W., Vermorgen, M., Deriemaecker, L., & Storms, G. (2015). Are we wasting a good crisis? the availability of psychological research data after the storm. *Collabra: Psychology*, *1*(1).

Wicherts, J. M., Borsboom, D., Kats, J., & Molenaar, D. (2006). The poor availability of psychological research data for reanalysis. *American Psychologist*, *61*(7), 726.

Wilson, G., Bryan, J., Cranston, K., Kitzes, J., Nederbragt, L., & Teal, T. K. (2017). Good enough practices in scientific computing. *PLoS computational biology*, *13*(6), e1005510.

Wood, B. D., Müller, R., & Brown, A. N. (2018). Push button replication: is impact evaluation evidence for international development verifiable? *PloS one*, *13*(12), e0209416.

Woodbridge, M. (2017). Jupyter notebooks and reproducible data science. Retrieved from https://markwoodbridge.com/2017/03/05/jupyter-reproducible-science.html