Computational Reproducibility via Containers in Psychology

April Clyburne-Sherin[1] and Seth Green[1]

[1] Code Ocean

Author Note

April Clyburne-Sherin is Outreach Scientist at Code Ocean. Seth Green is Developer Advocate. Correspondence concerning this article can be addressed to one or both authors: april@codeocean.com, seth@codeocean.com.

Abstract

Scientific progress relies on the replication and reuse of research. However, despite an emerging culture of sharing code and data in psychology, the research practices needed to achieve computational reproducibility —the quality of a research project entailing the provision of sufficient code, data and documentation to allow an independent researcher to re-obtain the project's results —are not widely adopted. Historically, the ability to share and reuse computationally reproducible research was technically challenging and time-consuming. One welcome development on this front is the advent of containers, a technology intended to facilitate code sharing for software development. Containers, however, remain technically demanding and imperfectly suited for research applications. This editorial argues that the use of containers *adapted for research* can help foster a culture of reproducibiliy in psychology research. We will illustrate this by introducing Code Ocean, an online computational reproducibility platform. (Disclaimer: both authors work for Code Ocean.)

Computational Reproducibility via Containers in Psychology

Introduction: the need for computational reproducibility

Stodden (2014) distinguishes between three forms of reproducibility: statistical, empirical, and computational[1]. In psychology, statistical reproducibility, broadly understood to encompass transparency about analytic choices and strategies, has received sustained attention from within and without the field (Gelman & Loken, 2014; Grange et al., 2018; Morey & Lakens, 2016; Simmons, Nelson, & Simonsohn, 2011). Likewise, empirical reproducibility —describing methods in sufficient detail to allow for independent replication —has been a high-profile issue in light of the work of the Center for Open Science (Collaboration, 2015; Nosek & Lakens, 2014).

Kitzes (2017) describes a research project as being "computationally reproducible" 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." Computational reproducibility facilitates the accumulation of knowledge by enabling researchers to assess results, particularly their robustness to alternate specifications, before embarking on conceptual or methodologically isomorphic replications. Moreover, as Donoho (2017) argues, preparing one's work for reproducibility 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). Overall, as Zelner (2016) has argued, computational reproducibility is "a necessary precondition to the new-data replication." Efforts to make psychology research computationally reproducible are therefore an essential part of broader reproducibility conversations.

Many psychology journals (Jonas & Cesario, 2015; Lindsay, 2017) address reproducibility by maintaining strong policies on sharing data, code, and materials; the

[1]Note that we use the term 'reproduction' to refer to recreating initial results using initial data, and replication to indicate furnishing 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 of this debate, see Marwick, Rokem, and Staneva (2017).

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 that researchers adhere "to SPSP's data sharing policy" (p. 3). In the same vein, the American Psychological Association's ethics policy (section 8.14) that states "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); numerous journals in the field require that their authors sign off on this policy (i.e., Cooper (2013)).

The challenge of computational reproducibility

Unfortunately, such policies alone are insufficient for computational reproducibility. Data and code that are available "upon request" may turn out to be unavailable when actually requested (Vanpaemel, Vermorgen, Deriemaecker, & Storms, 2015; Wicherts, Borsboom, Kats, & Molenaar, 2006). Even for authors willing, able, and incentivized to make their work reproducible, a number of technical hurdles must be overcome. Dependencies change over time, often in ways that break code (Bogart, Kästner, & Herbsleb, 2015), and versions are not always perfectly recordedbarba2016hard Many scientists lack the time to learn best practices for scientific programming (Sandve, Nekrutenko, Taylor, & Hovig, 2013; Wilson et al., 2017). Finally, computational resources are not evenly distributed among potential readers, meaning that memory or storage limitations can halt a reproduction effort (Deelman & Chervenak, 2008).

The result of these challenges is that even publicly available code and data are often not computationally reproducible. This holds across disciplines and programming languages. At the *Quarterly Journal of Political Science*, editors found that from "September 2012 to November 2015, for example, 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). Moreover, the editors found, "only four packages did not require any modifications. Of the remaining 20 papers, 13 had code that would not execute without errors, eight failed to

include code for results that appeared in the paper, and seven failed to include installation directions for software dependencies." When reviewing code for computer science papers, Collberg, Proebsting, and Warren (2015) find, if code does not run, it is often impossible "to determine the cause of a program crash: it could be because of internal issues, because the program was invoked with the wrong arguments, or because a file was missing, etc." (p. 7). In general, how much information suffices for reproduction becomes clear only when one is attempted.

Woodbridge (2017) recounts attempting to identify a sample of Jupyter notebooks mentioned in PubMed Central, thinking that reproduction "would simply involve 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 the notebook; troubleshooting language and tool specific issues required expertise and hindered portability; not all tools used in research are available to all researchers; and notebooks would often "assume the availability of non-Python software being available on the local system," but that 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."

A welcome development: containers

Meanwhile, software developers have created a variety of powerful tools for sharing code. Many of these tools, however. require substantial investments of time and effort to be used effectively. Chamberlain and Schommer (2014) note, for example, that virtual machines, "a safe and predictable way to share a complete computational environment ... have serious drawbacks," including 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 the advent of container technology. Silver (2017) writes that containers reduce complexity "by packaging the key elements of the computational environment needed to run the desired software, including settings and add-ons, into a lightweight, virtual box. They do not alter the resources required to run it —if a tool needs a lot of memory, then so too will its container. But they make the software much easier to use, and the results easier to reproduce."

Docker

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

1. a rich and growing ecosystem of supporting tools and environments, such as Rocker (Boettiger & Eddelbuettel, 2017), Docker images[2] specifically for R users; (edited)
2. being comparatively easy-to-use;
3. an open-source code base, allowing for adaptation (Hung, Kristiyanto, Lee, & Yeung, 2016) and integration with existing academic software (Grüning et al., 2016);
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 & Schommer, 2014)
5. compatibility with any programming language that can be installed on Linux.[3]

Adoption of container technology like Docker in psychology, however, remains scant,[4] for which 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. And the third may be that

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

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

[4] A search on 16 January 2018 of http://www.apa.org for the words "Docker container" yielded zero matches.

building, sharing, and running containers generally requires significant technical expertise above and beyond that needed to conduct a typical research project in psychology. The barriers to learning these skills are significant, not part of the standard curriculum for training researchers (Boettiger, 2015), nor an obviously effective use of researchers' time. To address these issues, we advocate for the development and use of container technology adapted specifically to the needs of researchers.

Code Ocean is an online computational reproducibility platform that aims to meet this need. Code Ocean provides infrastructure for storing and running Docker containers on cloud servers. This allows scientists to package code, data, results, metadata, and computational environment into a single compendium —which we call a 'compute capsule,'[5] or simply 'capsule' for short —whose results any researcher can reproduce by clicking 'run'. Each published capsule is assigned a DOI and can be embedded either directly into the text of an article or onto its landing page. Creating a tool that is fit-for-purpose for researchers means that more researchers can achieve computational reproducibility, easily reuse the analyses of others, and ensure that their code and data will run in perpetuity.

To illustrate how this works, the next section of this article will illustrate these features by walking through one compute capsule (Paluck, Green, & Green, 2017), available at `https://codeocean.com/2017/11/09/contact-hypothesis-revisited`.[6]

Code Ocean: customizing container technology for researchers

This section will outline some of the steps Code Ocean has taken to make the process of using Docker containers easier for researchers and those interested in research.

Reuse without downloading or technical setup

First, a capsule's results can be reproduced at the click of a button.[7] More specifically, doing so will reproduce the results shown in the 'Published Result' field. This

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

[6]This capsule is also available at `https://doi.org/10.24433/CO.f152260c-bebb-4157-a640-44579452b4e4.v2`.

[7]The manuscript is available as a preprint at `https://osf.io/preprints/socarxiv/w2jkf/`, and forthcoming in *Behavioural Public Policy*.

is accomplished by executing one main file —in this case, 'run.sh'[8] —that runs the other scripts in sequence, all on the cloud. Previous runs will display the length of time it took to produce those results, and no command line knowledge is presumed; nothing needs to be downloaded.

Configured to support research workflows

Second, Code Ocean offers support for any open-source language that can be installed on Linux, and also the proprietary languages Stata and MATLAB. This particular capsule runs with Stata 15 and R 3.4, illustrating that multiple languages can be run in sequence and that language versions are recorded. Each language comes with a number of pre-configured Docker images for common use cases; users cam also opt to start from a blank slate, with no scientific programming languages installed.

Dependencies made visible and accessible

Third, our package management system provides a transparent record of the packages necessary for reproducing results as well as the specific versions used.[9] The setup script allows for further customization if available package managers are not sufficient.[10]

Built-in reuse of common research datasets

Fourth, Code Ocean offers a number of built-in datasets. This makes the process of working with very common datasets a bit easier, and allows for easy loading of large datasets without undue demand on users' hard drives.

[8] A master script that runs all analysis script in sequence is an under-appreciated step that researchers can take to make their work accessible to others. Other important steps include using relative rather than absolute paths, specifying dependency versions, and saving results as files rather than presuming they will appear in a user's environment.

[9] When possible —Stata's SSC repository does not currently archive versions of user-written packages.

[10] In this particular case, the setup script is used to download some Stata packages from GitHub and a very old package for Stata that is not available through SSC.

Share or embed a capsule

Finally, Code Ocean lets readers easily share published capsules. Capsules can be posted to different social media, and an interactive widget may be embedded directly into the text of an article.[11]

Conclusion: Answering the call to make reproducibility tools simpler

In context of discussing Docker, (Boettiger, 2015, pp. 4-5) writes:

> 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 . . . To gain more widespread adoption, reproducible research technologies must make it easier, not harder, for a researcher to perform the tasks they are already doing . . . 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.

We believe that container technology is an important advance in this direction, and that Code Ocean, by building on this technology and adapting it specifically to the needs of researchers, gets a bit closer to the ideal of a fully reproducible workflow that is "easy to use and adapt" to existing research habits.

[11] For an example, see Gilad and Mizrahi-Man (2015).

References

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

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.

Collberg, C., Proebsting, T., & Warren, A. M. (2015). Repeatability and benefaction in computer systems research. *University of Arizona TR 14*, *4*.

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).

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.

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., ... others (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.

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.

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

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, B. L., Green, S. A., & Green, D. P. (2017, November). *Contact hypothesis revisited.* `https://www.codeocean.com/`. Retrieved from `https://codeocean.com/2017/11/09/contact-hypothesis-revisited/`

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*.

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.

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

Zelner, J. (2016). *Reproducibility starts at home.* Retrieved from `http://www.jonzelner.net/statistics/make/docker/reproducibility/2016/05/31/reproducibility-pt-1/`