Active Maintenance: A Proposal for the Long-term Computational Reproducibility of Scientific Results

Limor Peer[*] Lilla Orr[†] Alexander Coppock[†]

January 28, 2021

Abstract

Computational reproducibility, or the ability to reproduce analytic results of a scientific study on the basis of publicly available code and data, is a shared goal of many researchers, journals, and scientific communities. Researchers in many disciplines including political science have made strides toward realizing that goal. A new challenge, however, has arisen. Code too often becomes obsolete within just a few years. We document this problem with a random sample of studies posted to the ISPS Data Archive; we encountered nontrivial errors in seven of 20 studies. In line with similar proposals for the long-term maintenance of data and commercial software, we propose that researchers dedicated to computational reproducibility should have a plan in place for "active maintenance" of their analysis code. We offer concrete suggestions for how data archives, journals, and research communities could encourage and reward the active maintenance of scientific code and data.

[*]Yale University Institution for Social and Policy Studies

[†]Yale University Department of Political Science

[‡]We thank Jamie Druckman, Vicky Steeves, and Ann Green for helpful feedback on earlier drafts and Yuntian Xia and Brian Shih for excellent research assistance.

Introduction

A study is computationally reproducible if a sufficiently savvy third party could download the code and data from a public repository and successfully execute the analysis with minimal difficulty. Initiatives in political science like DA-RT (Lupia and Elman, 2014; Elman, Kapiszewski and Lupia, 2018) reflect a growing consensus that at least for research that relies on the formal analysis of quantitative or qualitative data that can be safely made public, computational reproducibility is a shared goal (Alvarez, Key and Núñez, 2018; Christensen, Freese and Miguel, 2019; Dafoe, 2014; Rohlfing et al., 2020).

While "the scientific enterprise depends on the ability of the scientific community to scrutinize scientific claims" (National Academies of Sciences Engineering and Medicine, 2019, p. 6), evaluating published scientific results can be difficult because readers must infer from written descriptions of statistical procedures what the authors must have actually done. Even when those descriptions are accurate, they are necessarily an approximation of what really occurred. The move toward computational reproducibility as a precondition of publication has been successful in revealing analysis errors and ensuring that shared materials are given persistent links (Gertler and Bullock, 2017).

We consider what happens *after* materials have been deposited in public archives. A study that was computationally reproducible on the day of deposit has already cleared a high bar. In addition to rigorous scrutiny of the manuscript, the code and data, or "replication archive" as it is commonly referred to in political science, were demonstrated by a trusted party to reproduce the numerical results reported in the paper. However, computational reproducibility is dynamic and sometimes elusive. Code that successfully executes on the day of deposit may break a year, a decade, or a generation later. Operating systems change and software packages are updated or not maintained at all. Data used for analysis may become unintelligible if the software used to create the data cannot be run. Workflows that depend on particular online resources may fail if the resources are moved or taken down. The cost of these failures is the potential for loss of knowledge and a diminished ability to make use of these materials in future settings.

In this paper, we propose "active maintenance" as a solution to this problem. In short, we suggest that replication archives should be periodically inspected to ensure that they run in contemporary computing environments according to a maintenance plan. Active maintenance builds on common practices in the data preservation (Conway, 2000) and code development communities (Fowler and Foemmel, 2006). Both recognize that supporting materials is an ongoing effort that requires resources, infrastructure, and standards.

Whether active maintenance is appropriate in a given case depends on the relative costs and benefits. The benefits of active maintenance are data and code that continue to provide scholars with tools to understand, critique, and reuse existing scientific knowledge. The costs associated with maintenance could of course be substantial. Whether the benefits exceed the costs will ultimately depend on the value of the scientific knowledge contained in the replication archive and on the difficulty of the required maintenance. We argue that studies deemed as important justify an investment in long-term reproducibility. Below, we discuss who among the authors, archives, publishers, scholarly communities, funders, universities, or other stakeholders might take responsibility for the active maintenance of digital materials.

Motivating example: "Shy Trump Voters"

We begin with a case study that illustrates over-time degradation in computational reproducibility and how active maintenance could address it. Coppock (2017) reports the results of a study that attempts to discern whether some portion of the polling misses in the 2016 U.S. Presidential election could be attributed to survey respondents who misreport their support for Donald Trump for fear of being perceived as racist or sexist.

The ISPS Data Archive reviewed the replication materials for Coppock (2017) in July of 2017 as part of its routine process (Peer and Green, 2012). Archive staff confirmed that all files necessary to replicate the reported results were available. The analysis consisted primarily of descriptive summaries and a comparison of direct and list experimental estimates with bootstrapped standard errors. Two years later, archive staff revisited these materials as part of a training exercise. Newly hired staff members encountered an error which prevented the code from running in full. Archive staff contacted the author, who traced the error to the removal of the "bootstrap" function from the `broom` package for R (Robinson and Hayes, 2019) and rewrote the bootstrap code using the `rsample` package instead (Kuhn, Chow and Wickham, 2020). ISPS staff then confirmed that results reported in the paper could be computationally reproduced and posted the updated materials.

This case is an example of unplanned active maintenance. Close working relationships between archive staff and ISPS researchers helped to facilitate a simple troubleshooting and update process. Extending the computational reproducibility of this study cost perhaps 10 emails and an hour of debugging.

Computational reproducibility is dynamic

Achieving computational reproducibility at the time of deposit is in itself a feat. Even before deposit, computational reproducibility typically requires that researchers maintain organized data, use version control, and test code, as per best practices of reproducible research (Christensen, Freese and Miguel, 2019; Dafoe, 2014). Reproducibility is challenged *after* deposit if future users cannot make use or sense of the files.

Reproducibility breaks down if datasets are damaged at the bit level or contained in proprietary file formats incompatible with current systems. After a period of time, the original software might be altogether unavailable (Peng, 2011) or restricted (e.g., closed-source) or the ability to use it constrained by a large number of components and dependencies (Hinsen, 2019; Ivie and Thain, 2018). Changes to statistical programing languages can ripple through the many add-on modules and packages that analysts rely on. For example, R updated how it generates random samples in 2019 to allow for greater uniformity of uniform random samples (Smith, 2019). As a result, analyses that depend on random number generation will yield slightly different results with later version of R, even if the replication archive included a random number seed. Further, the interpretability of study documentation and metadata is itself dynamic. Even if the documentation makes sense to a contemporary user, it may be less interpretable to users 50 years in the future – or even to the original researcher a few years down the line (Bowers, 2011). These breakdowns can prevent future researchers from computationally reproducing the original results and from using the materials in a meaningful way.

Evaluation of the long-term computational reproducibility of 20 studies

In order to assess long-term computational reproducibility in the ISPS Data Archive, we attempted to execute the analysis files associated with a sample of studies, all of which were computationally reproducible upon deposit. We compiled a list of all 97 archived studies, excluded three books, then randomly selected 20 studies. The sample included studies archived between 2009 and 2019, with four to 88 files associated with each study. These studies were archived using versions of Stata (18), R (1), and SPSS (1) available at the time of deposit.

Between December 2019 and March 2020, archive staff reviewed the program files using Stata 15, R 3.6.1 and 3.6.2, and SPSS 26. For each study, they reviewed documentation and downloaded data and analysis files. After adjusting working directories and file names, they

Table 1: Computational reproducibility in 20 studies

	Author Provided Code N = 20	Archive Added Code N = 11
No errors	13 (65%)	3 (27%)
Resolvable errors	6 (30%)	8 (74%)
Errors requiring further information	3 (15%)	0
Data preparation scripts not archived	3 (15%)	0

attempted to run each program file and recorded any errors. Errors were classified based on whether a moderately-trained replicator would be able to overcome them without substantially updating the script for the statistical analysis, or after receiving help from the study authors, as summarized in Table 1.

Of these 20 studies, 13 could be reproduced with present day hardware and software. A variety of challenges to computational reproducibility emerged in the remaining seven studies. In six, program files produced errors that could be resolved by seasoned analysts. Necessary adjustments, such as updating Stata's `outreg` command to `outreg2` are trivial for advanced users. In three of the studies, program files produced errors which could not be resolved by present day users without further information or documentation. For example, variables which now appear to be missing would most easily be supplied or explained by study authors. In three of the 20 studies, raw data and associated cleaning files were not deposited. Since it is often necessary to exclude raw data from replication files, we did not classify the resulting errors as indications of failed computational replication.

For 11 studies, archive staff added R files at the time of deposit as part of an effort to convert statistical code to open source formats. Of the eight instances where the R programs did not run, several included simple typos. However, emerging challenges to computational reproducibility were generally more difficult to overcome. For example, three relied on packages that had been removed from CRAN (e.g., `Design`), or updated in ways that made it impossible to run the original syntax (e.g., `Zelig`).

In sum, we find that even in the relatively short period of 10 years, replication materials that originally allowed for computational reproducibility broke down. Most of the challenges were encountered in the older studies. Sophisticated users would have little difficulty resolving many of the errors which appeared, but even some of the user-resolvable errors would have required substantial troubleshooting and re-coding.

Technology-based approaches

A valuable, if not wholly satisfying, solution to the over-time degradation of computational reproducibility is containerization. The replication archive and the computing environment as it existed at the moment of deposit (a "snapshot") are placed in a virtual container guaranteeing that, even if run on different machines, the same results will be reproduced. Emulation – running legacy software on modern computers – is a procedurally distinct approach that similarly allows the user to execute code on a host system. Both approaches have the principal virtue of dramatically increasing the probability that old code will run in the future.

Nevertheless, these approaches face inherent limitations when the goal is for future users to make use or make sense of the files. As Katz (2017) observes, containers "provide bitwise reproducibility, but aren't scientifically useful, because as black boxes, you can't really remix the contents." For example, consider Brady and Ansolabehere's 1989 analysis of voter preferences, which was conducted in FORTRAN. Containerization was not available at the time, but if it had been, we would in principle be able to reproduce the analysis within the confines of the time capsule from 1989. However, extending the analysis to modern data or incorporating later methodological advances would not be possible.

Technology-based approaches also face a coordination problem. For example, containerization and packaging tools (e.g., Research Compendium, Reprozip, WholeTale) are proliferating, but importantly, any such tool will be most reliable when it is popular. If a technological solution falls out of favor in the scientific community, or fails to gain traction in the first place, it is not likely to be maintained. With no training or support networks for users in the somewhat distant future, re-opening containers or accessing any technological tool may become increasingly difficult over time.

Active Maintenance

Our proposed solution to the problem of over-time degradation in computational reproducibility is active maintenance. Unlike static images, these strategies afford more robust options for future reuse. Active maintenance requires monitoring materials and deploying a combination of strategies drawn from two fields: data preservation and code development. Recent recommendations on how to sustain content over time parallel our active maintenance suggestion (Daigle et al., 2018).

Our thinking about active maintenance is influenced by digital archivists' approaches to preserving the usability of digital materials. "Active preservation" is a curatorial responsibility and relies on strategies including copying and migration. Copying entails saving a copy of the data in a system-independent, nonproprietary format that is likely to be machine readable in future decades. This procedure has some advantages, but some information may be lost in translation. Migration goes futher and "includes refreshing the media but also addresses the internal structure of the files so that the information within can be read on subsequent computer platforms, operating systems, and software" (Green et al., 1999). With code being instrumental for computational reproducibility, more archives and repositories are applying these strategies to software (Chassanoff et al., 2018).

Software developers are keenly aware of how changes to the computational environment affect the performance of source code. Software testing techniques such as continuous integration involve recompiling software packages regularly to verify the software runs without errors. Key principles include frequent automatic testing, documenting specifications and dependencies, and saving all components in one location. Adopting code testing principles makes science more robust (Krafczyk et al., 2019). Committing to automatically re-run computational analysis whenever relevant changes are made can increase the probability that the code remains available and usable (Beaulieu-Jones and Greene, 2017).

Discussion

Computational reproducibility is a special kind of knowledge that offers future analysts a way to scrutinize and understand scientific studies more intimately and tangibly than the journal article can provide. But computational reproducibility is time-varying: we showed in the case of the ISPS Archive how reproducibility degraded over time. As political science and other fields develop shared values and norms around computational reproducibility, we must recognize that our commitment to this knowledge should not end on the day we deposit the study in the archive.

We propose active maintenance as a way to enhance long-term computational reproducibility. Active maintenance requires effort, raising two important questions: is the effort worth it and who should be making the effort?

We think that the labor required to open up replication archives every so often to see if they still run will differ from study to study. As a general rule, "clean" well documented code with proper metadata will be easier to maintain and should be a matter of course. Whether the substantial effort required to update complicated replication archives is worth it will depend on the importance of the knowledge contained in the replication archive as well as various communities' ideas about how the materials ought to be used in the future. In our view, important replication archives should be actively maintained, but we refrain from passing judgment on which studies those are.

Who should be making these efforts depends on collective values, priorities, and incentives. Research teams could take the responsibility of enhancing reproducibility. Under this model, the onus is on authors to adhere to "best practice" guidelines and to keep up with advances in contemporary statistical computing. Currently, authors do not face incentives that encourage this kind of sustained investment, though their incentives are at least in part a function of disciplinary norms that could change. Under an alternative model, active maintenance could be crowd-sourced to the wider academic community. Much like user contributions to software made on GitHub, one could imagine the intended audience of the replication archives – future scholars – contributing up-to-date analysis scripts. Many courses have replication project assignments. Updating analysis code and uploading it to the repository could be a fitting capstone to such an assignment. This model faces the obvious difficulty that it relies on the good will of the unspecified "crowd" to contribute to knowledge, not to mention the cooperation of authors and archives to review their revisions. Under a third model, independent third parties such as the Odum Institute or CASCAD could also provide active maintenance services on behalf of journals, universities, or scholarly associations. Currently, these services certify computational reproducibility at the time of deposit, though one could imagine expanding their range of responsibilities. Finally, the responsibility for active maintenance could fall to archive or repository staff. Data archives are well-positioned to provide services verifying computational reproducibility as part of standard curation practice (Peer, Green and Stephenson, 2014). Archives can conduct an initial review, incorporate tools, and enforce policies conducive to active maintenance. For example, requiring particular file formats or specific information in a readme file at the time of deposit. Many archives already conduct automated bit-level checks and could potentially expand to perform routine reproducibility checks.

In light of the dynamic nature of computational reproducibility, we urge the community to recognize that supporting data and code is an ongoing effort that requires resources, infrastructure, standards, and policies. Studies that were computationally reproducible on the day they were archived often cannot be reproduced on modern software and hardware only a few years later – for some such studies, we think computational reproducibility is worth actively maintaining.

References

Alvarez, R. Michael, Ellen M. Key and Lucas Núñez. 2018. "Research Replication: Practical Considerations." *PS: Political Science & Politics* 51(2):422–426.

Beaulieu-Jones, Brett K. and Casey S. Greene. 2017. "Reproducibility of Computational Workflows is Automated using Continuous Analysis." *Nature Biotechnology* 35(4):342–346.

Bowers, Jake. 2011. "Six steps to a better relationship with your future self.".
URL: *https://cpb-us-e1.wpmucdn.com/blogs.rice.edu/dist/d/2418/files/2013/09/tpm_v18_n2.pdf*

Brady, Henry E. and Stephen Ansolabehere. 1989. "The Nature of Utility Functions in Mass Publics." *American Political Science Review* 83(1):143–163.

Chassanoff, Alexandra, John Borghi, Yasmin AlNoamany and Katherine Thornton. 2018. "Software Curation in Research Libraries: Practice and Promise." *Journal of Librarianship and Scholarly Communication* 1(eP2239).

Christensen, Garret, Jeremy Freese and Edward Miguel. 2019. *Transparent and Reproducible Social Science Research: How to do Open Science.* University of California Press.

Conway, Paul. 2000. *Overview: Rationale for Digitization and Preservation.* Northeast Document Conservation Center.

Coppock, Alexander. 2017. "Did shy Trump supporters bias the 2016 polls? Evidence from a nationally-representative list experiment." *Statistics, Politics and Policy* 8(1):29–40.

Dafoe, Allan. 2014. "Science Deserves Better: The Imperative to Share Complete Replication Files." *PS: Political Science & Politics* 47(1):60–66.

Daigle, Bradley, Karen Cariani, Carol Kussmann, Nathan Tallman and Lauren Work. 2018. "Levels of Digital Preservation.".
URL: *https://ndsa.org/publications/levels-of-digital-preservation/*

Elman, Colin, Diana Kapiszewski and Arthur Lupia. 2018. "Transparent Social Inquiry: Implications for Political Science." *Annual Review of Political Science* 21(1):29–47.
URL: *https://doi.org/10.1146/annurev-polisci-091515-025429*

Fowler, Martin and Matthew Foemmel. 2006. "Continuous Integration.".
URL: *https://martinfowler.com/articles/continuousIntegration.html*

Gertler, Aaron L. and John G. Bullock. 2017. "Reference Rot: An Emerging Threat to Transparency in Political Science." *PS, Political Science & Politics* 50(1):166.

Green, A G, J A Dionne, M Dennis, M J Dennis and Digital Library Federation. 1999. *Preserving the Whole: A Two-track Approach to Rescuing Social Science Data and Metadata.* Digital Library Federation.

Hinsen, Konrad. 2019. "Dealing with Software Collapse." *Computing in Science & Engineering* 21(3):104–108.

Ivie, Peter and Douglas Thain. 2018. "Reproducibility in Scientific Computing." *ACM Computing Surveys (CSUR)* 51(3):1–36.

Katz, Daniel. 2017. "Is Software Reproducibility Possible and Practical?".
URL: *https://danielskatzblog.wordpress.com/2017/02/07/is-software-reproducibility-possible-and-practical/*

Krafczyk, Matthew, August Shi, Adhithya Bhaskar, Darko Marinov and Victoria Stodden. 2019. Scientific Tests and Continuous Integration Strategies to Enhance Reproducibility in the Scientific Software Context. In *Proceedings of the 2nd International Workshop on Practical Reproducible Evaluation of Computer Systems.* P-RECS '19 New York, NY, USA: Association for Computing Machinery pp. 23–28.

Kuhn, Max, Fanny Chow and Hadley Wickham. 2020. *rsample: General Resampling Infrastructure.*
URL: *https://cran.r-project.org/package=rsample*

Lupia, Arthur and Colin Elman. 2014. "Openness in Political Science: Data Access and Research Transparency: Introduction." *PS: Political Science & Politics* 47(1):19–42.

National Academies of Sciences Engineering and Medicine. 2019. *Reproducibility and Replicability in Science.* Washington, DC: The National Academies Press.

Peer, Limor and Ann Green. 2012. "Building an Open Data Repository for a Specialized Research Community: Process, Challenges and Lessons." *International Journal of Digital Curation* 7(1):151–162.

Peer, Limor, Ann Green and Elizabeth Stephenson. 2014. "Committing to Data Quality Review." *International Journal of Digital Curation* 9(1):263–291.

Peng, Roger D. 2011. "Reproducible Research in Computational Science." *Science* 334(6060):1226–1227.

Robinson, David and Alex Hayes. 2019. *broom: Convert Statistical Analysis Objects into Tidy Tibbles.*
URL: *https://cran.r-project.org/package=broom*

Rohlfing, Ingo, Lea Königshofen, Susanne Krenzer, Jan Schwalbach and Ayjeren Bekmuratovna R. 2020. "A Reproduction Analysis of 106 Articles Using Qualitative Comparative Analysis, 2016–2018." *PS: Political Science & Politics* pp. 1–5.

Smith, David. 2019. "What's new in R 3.6.0.".
URL: *https://blog.revolutionanalytics.com/2019/05/whats-new-in-r-360.html*