To appear in Harvard Data Science Review

Hopes and Limitations of Reproducible Statistics and Machine Learning[1]

Andrew Gelman[2]

7 Feb 2024

It is a pleasure to comment on an article by David Donoho, who throughout his career has combined mathematics and computation in order to develop methods that have made such a difference in applied statistical practice and understanding. I echo his enthusiasm for recent developments in statistics and machine learning that allow progress in many areas by going beyond old-fashioned theory. Theory can be helpful in developing our understanding of existing methods and providing tools to construct new methods and solve new problems; it also can actively get in the way of scientific progress, as we have seen most notoriously in the overuse of null hypothesis significance testing in many areas of science. Even in the low-tech areas of social science where I do most of my applied work, computational tools of statistical graphics and predictive modeling have made a bit difference in our understanding, and this is even so much more true in big-data areas of science and engineering such as computational biology, image analysis, and language processing.

Usually we think of three key contributors to modern statistics and machine learning as being new algorithms, big data, and fast parallel computing. Donoho discusses all of these while emphasizing what might be called the social or organizational context: algorithms being more portable through shared code (I'd also add the development of statistics-friendly computer languages such as S, Python, and Stan), public data repositories, and focused goals. These structural developments have facilitated progress in algorithms and statistical understanding as well as spreading ideas that would have been developed anyway.

In addition, even in the areas where we have not yet reached the frictionless future envisioned in the article under discussion, frictionless reproducibility is a useful goal. It's not all or nothing; every reduction of friction helps a bit. It can take a while between the development of a statistical idea and its implementation in a reproducible way, and that's ok. Right now I'm working with colleagues on a research project on accounting for survey weights in regression models--applying relatively high-tech methods to a problem arising from traditional and low-tech statistics--and we're doing lots of computation on our own laptops using data that happen to be immediately accessible to us. The next stage is for us to write code that others can try to break, and the next stage beyond that—if the project merits the effort—will be to write some general-use package with a smooth workflow, to make the method accessible to a wider group of users and, perhaps more importantly, because more users mean more testers, more problems for the method to be applied to, and more ways for it to fail, leading (we hope) to fixes and improvements. Setting up frictionless reproducibility in this context won't be easy and it won't come all at once, but the more my colleagues and I can do on this front, the faster our research idea should develop and become useful—or, perhaps, the faster we will realize that it's a dead end, but, yes, if that's where it will eventually go, we'd prefer to learn that sooner than later. We

[1] Discussion of "Data Science at the Singularity" by David Donoho for *Harvard Data Science Review.* We thank the U.S. Office of Naval Research for partial support of this work.
[2] Department of Statistics and Department of Political Science, Columbia University, New York.

should also recognize generational changes. I've occasionally tried to use R notebooks and similar tools to integrate writing, research, and coding, but I still find it easier and more productive to work with code and writeup in two separate files. Students, though, are just fine with notebooks, and their workflows are often much more reproducible and effective than mine. This, I assume, is the future.

It's good to aim for frictionless reproducibility. The effort it takes to make a research idea reproducible is often worth it, in that getting to reproducibility typically requires a level of care and rigor beyond what is necessary just to get a paper published. One thing I've learned from working on widely-used textbooks such as Bayesian Data Analysis and open-source software such as Stan is that much is learned in the process of developing a general tool that will be used by strangers. There's a big difference between analyzing one dataset or publishing a method and demonstrating it on one or two problems, and providing models and methods that will be used as defaults by thousands of people in the wild. Statistics has sometimes been said to be the science of defaults, which is another way of saying that we develop methods in service to science and engineering.

So I agree with Donoho regarding the past and future importance of reproducible computational science. At the same time, let's remember that reproducibility is not by itself a guarantee of quality. In the context of the replication crisis in science, we have written that honesty and transparency are not enough; that is, researchers can conduct their work with openness and good faith and still do bad science and produce results that do not persist when their studies are replicated.

From the other direction, I have heard laments from colleagues in computer science that much high-profile work published in top conference proceedings remains unreproducible. The problem here seems to be not with the technology but with incentives and expectations. If a computer science professor is supervising many students and postdocs, and each of them intends to present at multiple conferences per year, and if each paper must be written so as presenting a breakthrough (perhaps necessary to survive the competition of being accepted, let alone to receive a coveted best-paper award) . . . this is an irresistible-force-meets-immovable-object scenario. Except in rare exceptions, the lab can't possibly be making progress each year to justify the rate of claims being made. This puts pressure to get results, one way or another. And one way to get results with a machine learning algorithm and make them look good is to do problem-specific tuning and then report the results without giving all the information. Enforcement of reproducibility is a great step forward here; there will just be a challenge to do this within the current system in which more and more researchers are expected to come up with regular breakthroughs. As with the replication crisis in science, it would help if improvements in methods and structures were accompanied by reduced demands for novelty and progress on each project.

Finally, it's hard for me to believe that Donoho believes all the positions taken in his own article. For example, he refers approvingly to a vision that "data science research projects making full use of [FR-1]+[FR-2]+[FR-3] are truly empirical science; while research projects without all three will fall short in some way." Given that all these three elements ("datafication of everything," "research code sharing," and "a shared public dataset, a prescribed and quantified task performance metric, a set of enrolled competitors seeking to outperform each other on the task, and a public leaderboard") are all relatively new, that statement would imply that all research projects until very recently have not been "truly empirical science," a form of presentism that I would assume that even extreme AI proponents would not take.

Rather than drawing a line and saying that no science that came before was "truly empirical," I would prefer to say that modern science and engineering face exciting new opportunities, and reproducible workflow promises to be an important part of future progress. I like Donoho's article because he goes beyond generalities to consider specific focus on aspects of reproducibility (open data, open code, and competitions) that are particularly relevant to data science, and which remain challenging, with open data being in conflict with proprietary interests as well as privacy, open code being in conflict with commercial aims, and competitions being narrow in their scope. To recognize the challenges in fostering frictionless reproducibility is not to diminish its potential importance where it is possible, and to recognize the limitations of structural changes is not to diminish their value.