A Health Economist Walks Into a Tech Company: Principles for Reproducible Real-World Analyses

Blythe Adamson, PhD, MPH, Josh Kraut, MA, Carrie Bennette, PhD, MPH, Flatiron Health, New York, NY, USA

The need to use more reproducible techniques in health economics and outcomes research (HEOR) is growing rapidly as analyses of real-world data become more frequent, involve larger datasets and employ more complex computations. Guiding principles for reproducible code are 1) write with an audience in mind, 2) do not repeat yourself, and 3) write code that is modular and reusable.

The oncologist struggled to find the right words. The scientific publication upon which she based her most recent treatment recommendation for the patient sitting in front of her had just been retracted from a prestigious journal. She reflected on a lengthy discussion with this patient 6 months prior considering the trade-offs between treatment options. Balancing the evidence of efficacy, value of hope, and impact on quality of life was difficult enough when based on accurate and reliable research. The retracted comparative-effectiveness study that had once embodied so much promise now brought bitterness and confusion.

The cost of bad clinical research often extends beyond these intimate conversations to the broader scientific field. Scientific advances are almost universally incremental; they build upon the foundation laid by the previous generation. If that foundation turns out to be unstable, entire research areas that were built on top of it can crumble.

For centuries, the responsibility to identify mistakes in scientific research has fallen largely on the shoulders of peer reviewers. They are challenged to evaluate the integrity and accuracy of a manuscript critically. Peer reviewers can be "generous" to the authors by giving them the benefit of the doubt and assuming the black box of methods described is full of the rigorous tools we expect. However, unfortunately, manuscripts are often missing detailed methods, analysis code, and/or the raw data necessary to check computationally intensive research critically. As fields like HEOR embrace the enormous potential of "big data" and become increasingly reliant on modern scientific computing tools to answer important research questions, the gap between what is included in a written manuscript and what is needed to evaluate the research critically grows.

HOW DO WE KNOW IF THE RESULTS OF STUDIES ARE ACCURATE?

The first step is simple: reproducibility. But how do you define "reproducible"? Does it simply mean other people in your organization can run your analysis code on their machine? Or if we asked a stranger to read one of your publications and you handed them the raw data, should they find the exact same answer if they tried to recreate the analysis? Years from now, when I want to update an old analysis with new data, will I be able to dust off my old code, understand it, and run the analysis again?

There are 2 main reasons why we need to ensure research is reproducible. First, we must show evidence that methods and results are accurate (improve transparency). This reduces uncertainty for decision makers and peer reviewers. Second, we must enable others to make use of and/or build on the methods and results. This is needed to accelerate the development of new medicines.

Although reproducibility correlates with better science, it is no guarantee. Recent discussions of the book, *Rigor Mortis: How Sloppy Science Creates Worthless Cures, Crushes Hope, and Wastes Billions*, by NPR Scientific Correspondent Richard Harris created waves of realization and plans for reformation in the research community.[1] Discussions in the media and in scientific literature have recently emphasized the importance of reproducible research, including a special issue of the journal *Science*.

The need to use more-reproducible tools in HEOR is growing rapidly as analyses of real-world data become more frequent, involve larger datasets, and employ more complex computations. Data scientists now demand and support the curation of high-quality data—aligning with regulatory agencies, health technology authorities, clinicians, patients and healthcare payers around the world that demand high-quality, real-world evidence to make decisions.

THINGS SOFTWARE ENGINEERS CAN TEACH US

Transformation of messy data into meaningful evidence often needs teams of researchers from different disciplines working together with clear

>

communication, documentation, and organized code. Despite being commonplace in computer science programs, graduate training programs in health economics and epidemiology often miss the mark on the opportunity to teach students how to structure and organize code, particularly in team-based settings. Software engineers have developed mature solutions for building robust and reproducible analytic software and provide a wealth of knowledge and tools that can be leveraged by health economics and outcomes researchers.

WHAT IS "GOOD" CODE?

We follow and teach these guiding principles for reproducible code:

1. Specify your analysis plan prior to accessing your dataset
2. Write with an audience in mind
3. Do not repeat yourself
4. Code should be modular and reusable
5. Code should be version controlled

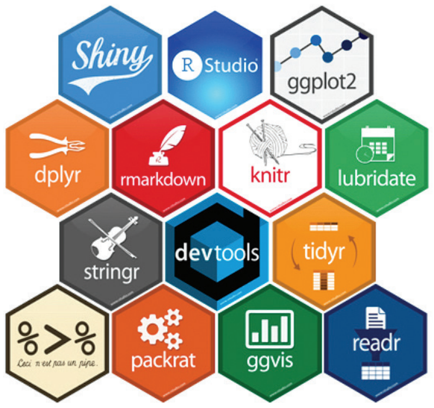
In today's digital data era, it can be very easy for scientists to simply test many different analytic approaches to their dataset and cherry-pick the results that are best suited for their research aims. To prevent this type of behavior, it is critical for scientists to define their analytic protocol prior to undertaking the analysis step and stick to the protocol. Today's software may make it easy for scientists to iterate over their analysis many times, but this opens the door for introducing a type 1 error.

Importantly, we should all strive to write human-readable code. Analysis code should be easy for anyone on your team and your future self to look at and understand what it is doing. Writing readable code reduces errors and increases efficiency during code review and when revisiting old analyses. To that end, analytic code should aim to create a narrative story that is easy for readers to follow. Even if you don't think someone else will be looking at your code, assume you are going to end up looking at it down the road and that you'll have no idea what you were thinking when you wrote it.

Writing functions is one of the building blocks to writing reusable and robust analytic code. Well- written functions help make your intent clear. They can reduce copy/paste mistakes and make updating and testing your code easier. Our guiding best practices for writing functions include: 1) keep them short, 2) do one thing and do it well, and 3) use intuitive names.

Finally, the use of formal version control systems like Git and SVN provide critical functionality for tracking changes made to code. In addition to allowing users to formally keep a working record of all changes to a project's code, version control systems allow for easy collaboration between code authors and provide built-in mechanisms that make it easier for code authors to review one another's code. These version control tools help code authors manage their analysis and ensure that specific versions of an analysis can easily be recalled later.

Figure 1. Recommended data science tools in R that are free and publicly available. Image credit: http://docs.rstudio.com/products.html



FREE TOOLS AVAILABLE TO HELP YOU

Excellent tools for publishing and sharing reproducible documents are commonplace in data science organizations at technology companies, although they are rarely utilized in academic research. We use and have had great success with R, Python, Rstudio, and Jupyter for writing scientific code. These are free, open-source, and exponentially growing in use. The utilization of Integrated Development Environments (IDEs) like Rstudio and Jupyter can make it easier for less-technical scientists to interact with computational analyses.

Using open-source programming languages and tools has many benefits. The key benefit of markdown-based notebooks (Rmarkdown, Jupyter) is the ability to keep your analysis code and output all in one place—the concept of literate statistical programming. Copying and pasting results from SAS/STATA output is no longer accepted as reproducible. Modern open-source programming languages also make it easy to communicate results with colleagues. By running a single command, R and Python file scan automatically and reproducibly write and export beautiful html web pages, Microsoft Word documents, and publication-worthy PDFs.

Packages can be built for internal use in an organization to ensure that analysts implement methods consistently between people and over time. Within the R universe, Hadley Wickham, the data scientist who pioneered the concept of "tidy data," has assembled an entire "tidyverse" of packages to help wrangle messy real-world data into tidy data.[2] Within the Python universe, Wes McKinney's "pandas" library is widely used for tabular data analysis.

NEXT-GEN OUTCOMES RESEARCH

As HEOR increasingly relies on large and complex real-world data, next-gen researchers will need to adopt more skills from the field of software engineering. Adopting these tools across the scientific research space and developing new standards and best practices for real-world data scientists are critical to ensure the next generation of research is reproducible. •

REFERENCES

1. Harris R. Rigor Mortis: *How Sloppy Science Creates Worthless Cures, Crushes Hope, and Wastes Billions*. New York, NY: Basic Books; 2017.

2. Grolemund G, Wickham H. *R for Data Science*. Sebastopol, CA: O'Reilly Media, Inc; 2017.

ADDITIONAL INFORMATION

The preceding article is based on the author's blog post, https://flatiron.com/blog/tools-for-reproducible-real-world-data-analysis/ and corresponding ISPOR Europe 2018 short course "Tools for Reproducible Real-World Data Analysis."