Solutions to Promote Reproducibility

Jürgen Schneider*

2025-07-22

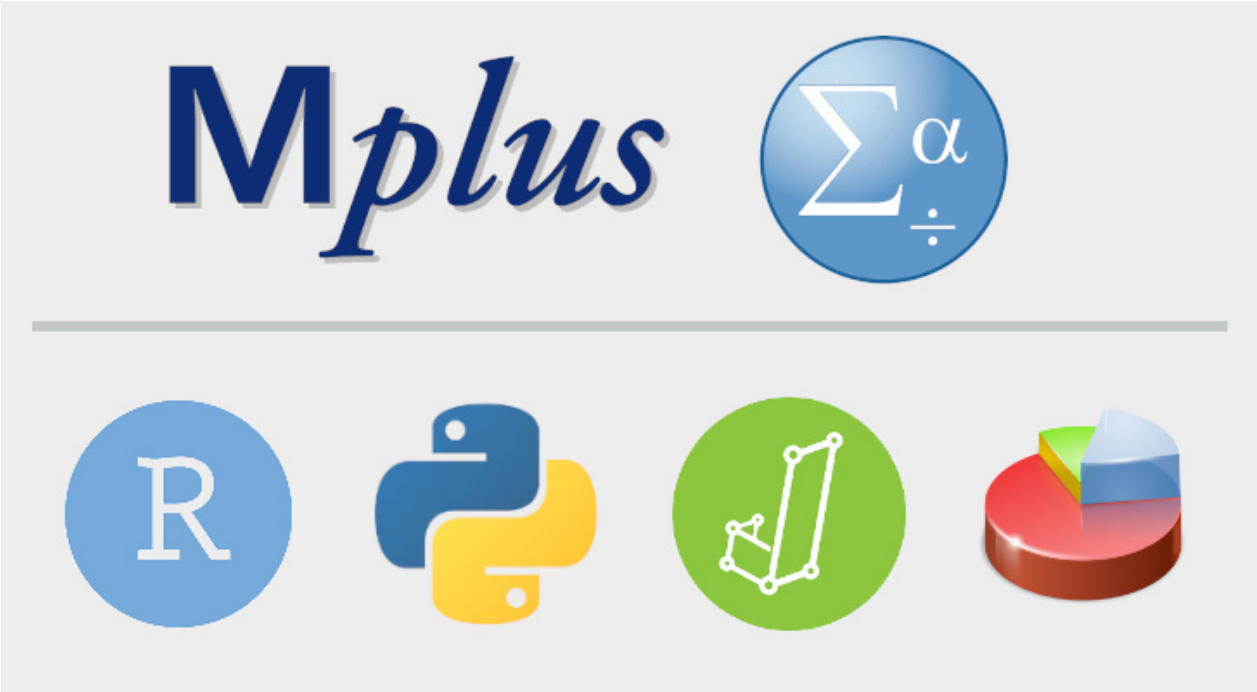
Table of contents

Solutions to Promote Reproducibility

Below you'll find information on how open file formats and software, Input-Output-Documents ("Notebooks"), and version management and containerization promote computational reproducibility in data analysis.

A basic prerequisite for reproducibility is, of course, that both the data and the analysis code are shared in the first place — "as open as possible, as closed as necessary". Depending on applicable restrictions, this can be achieved in different ways — for example, by using repositories such as Zenodo or the Open Science Framework (OSF), which allow for configurable access settings being fully open or restricted.

*ORCID: 0000-0002-3772-4198



1 Open File Formats and Software

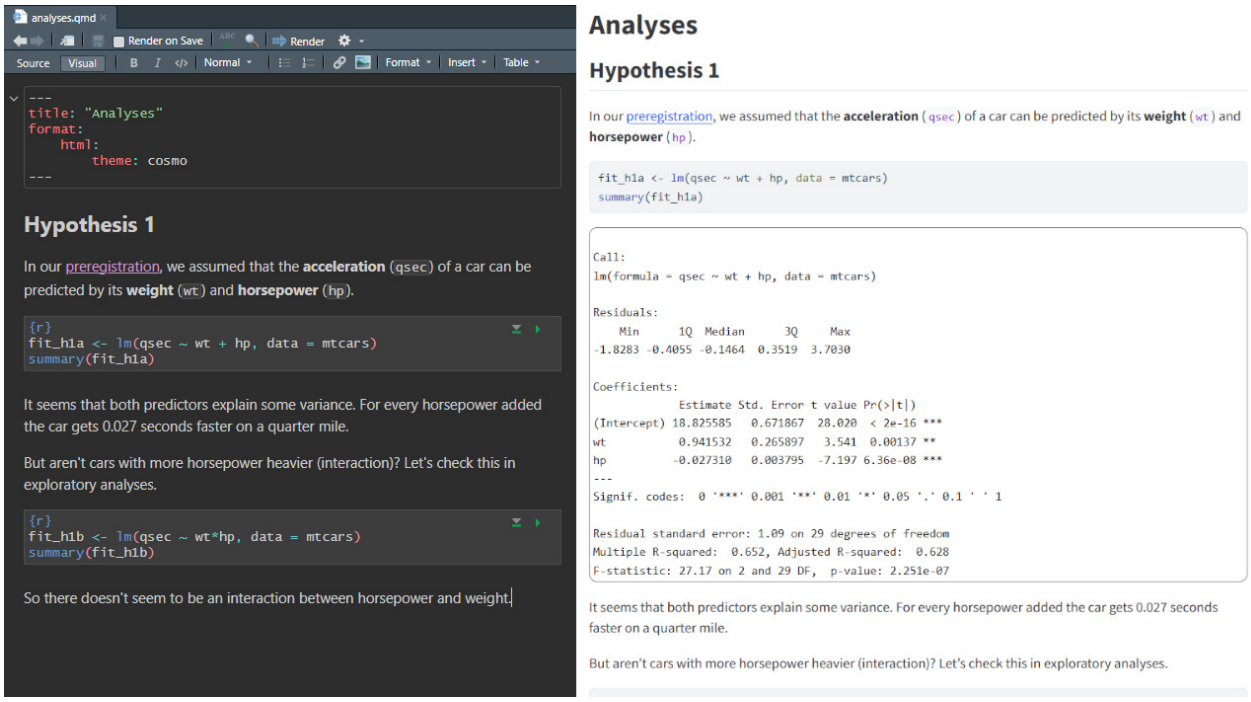
Reproducibility in research relies on sharing both the steps used to analyze data and the data itself. Using open file formats and free, open-source software helps ensure that others have access to the tools needed to open your files and – a basic prerequisite for inspecting and rerunning your analysis. With software such as R and Python, researchers can document their data analysis procedures (in so-called scripts) that others can reuse or adapt. For those less familiar with programming, point-and-click tools like JASP, jamovi, and PSPP offer user-friendly alternatives. These free options remove financial barriers and make it easier for others to verify and build on your work. In contrast, proprietary software such as SPSS or MPlus restricts accessibility, limiting reproducibility to users who can afford expensive licenses.

Advantages:

- Fundamentally enables reproducibility, as others can open and run your analyses.
- Open-source software lets researchers see how the software processes data "under the hood" when using its functions.
- Promotes equity in access to research.

Resources:

- Intro to Python (edX)
- Intro to R (edX)
- Intro to JASP (YouTube)
- Intro to jamovi (YouTube)



2 Input-Output Documents ("Notebooks")

Input-output documents integrate data analysis code (input) with corresponding results (output) in a unified format. Tools such as RMarkdown, Quarto Markdown, and Jupyter Notebooks (compatible with R or Python) enable the combination of code, results, and explanatory text—such as interpretations—into a single document exportable as HTML or PDF. Similarly, JASP and jamovi provide built-in functionality to achieve this integration within their platforms.

Advantages:

- Provides provenance of results: Directly links your transparent inputs to outputs (e.g., results).
- Ensures error-free output: HTML/PDF documents are only rendered if all your code, from data import to variable manipulation and analysis, runs without errors
- Enhances understandability: Enables you to provide detailed explanations of analytical approaches and result interpretations.

Resources:

- Get started with Quarto (YouTube)
- Share & annotate JASP
- Share jamovi
- Annotate jamovi

3 Containerization and Version Management

R and Python rely on additional packages to extend their core functionality. However, differences in package versions can lead to compatibility issues—even when using the same programming language. Tools such as `renv` and `groundhog` for R help address this by ensuring consistent package versions, making analyses more robust and sustainable over time.

Beyond package management, maintaining a consistent system environment (e.g., version of R itself, dependencies) may be relevant for full reproducibility in the long run. Containerization tools such as Docker or the R package `holepunch` bundle scripts together with the system they depend on. This encapsulation ensures that analysis scripts run reliably across different computers and over time. JASP and jamovi offer limited control over versioning, and their future backwards compatibility is uncertain.

For managing both system and package versions together, the R package `rix` offers a combined solution to streamline reproducibility workflows.

Advantages:

- Ensures sustainability of your analyses over time.
- Creates equal system environments for all collaborators.
- Ensures equal package versions across machines.

Resources:

- Tutorial: holepunch
- Tutorial: renv
- Tutorial: groundhog
- Intro to rix (YouTube)