useR! Salzburg 2024

Reproducible data science with WebAssembly and webR

George Stagg
Posit, PBC

What is “reproducible” software?

1. The ability for anyone to easily re-run and verify the result of some computational procedure.

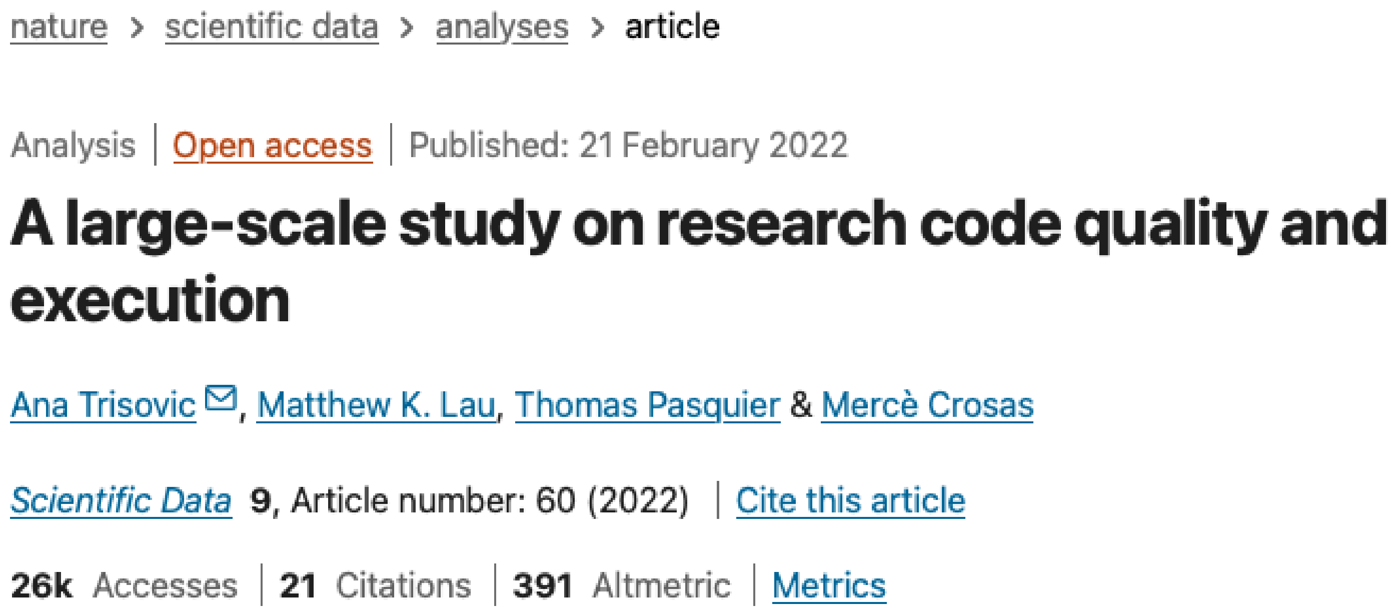
2. The ability to modify or extend the procedure (software and/or data) to gain new insights.

There is some level of subjectivity here: “easily”

There are levels of reproducibility, some are easier than others.

More info:
🔗 Building reproducible analytical pipelines with R, Bruno Rodrigues
🔗 ONS RAP Companion
🔗 Software Carpentry, Data Carpentry

Reproducibility in practice



nature > scientific data > analyses > article

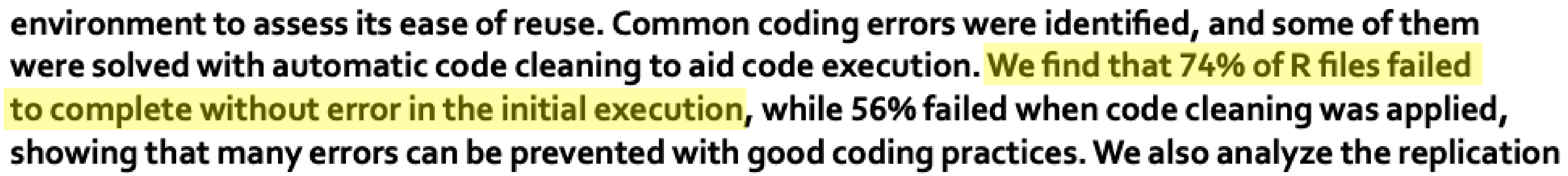
Analysis | Open access | Published: 21 February 2022

A large-scale study on research code quality and execution

Ana Trisovic, Matthew K. Lau, Thomas Pasquier & Mercè Crosas

Scientific Data **9**, Article number: 60 (2022) | Cite this article

26k Accesses | **21** Citations | **391** Altmetric | Metrics



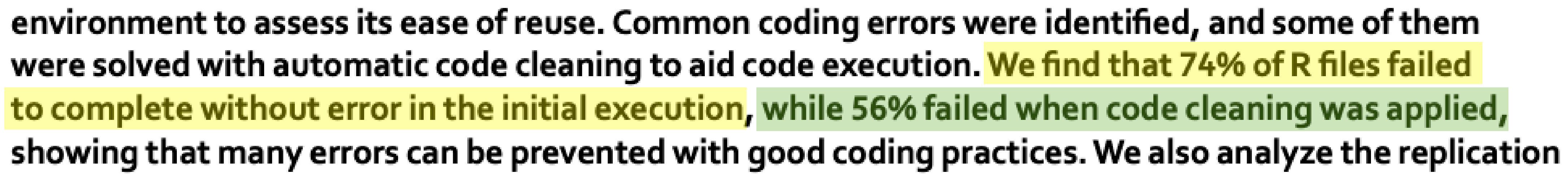
environment to assess its ease of reuse. Common coding errors were identified, and some of them were solved with automatic code cleaning to aid code execution. We find that 74% of R files failed to complete without error in the initial execution, while 56% failed when code cleaning was applied, showing that many errors can be prevented with good coding practices. We also analyze the replication

At the language level

“It works on my machine ¯\_(ツ)_/¯”

- Hard-coded paths, `setwd()`, project organisation.
- Source code management, `git`, GitHub.
- Defensive programming, handling error conditions.
- Organising software into modules or packages.
- Documentation and tests.

A small effort here, even using automated tools, makes a big difference.



environment to assess its ease of reuse. Common coding errors were identified, and some of them were solved with automatic code cleaning to aid code execution. We find that 74% of R files failed to complete without error in the initial execution, while 56% failed when code cleaning was applied, showing that many errors can be prevented with good coding practices. We also analyze the replication

Computing environments

Many different computational environments exist, all with the potential to affect your analysis.

Language and package management

- Versions of interpreter software:
  R 3.6.3, 4.1.3, 4.4.1
- Versions of packages

Tools: rig, p3m, renv

System and library management

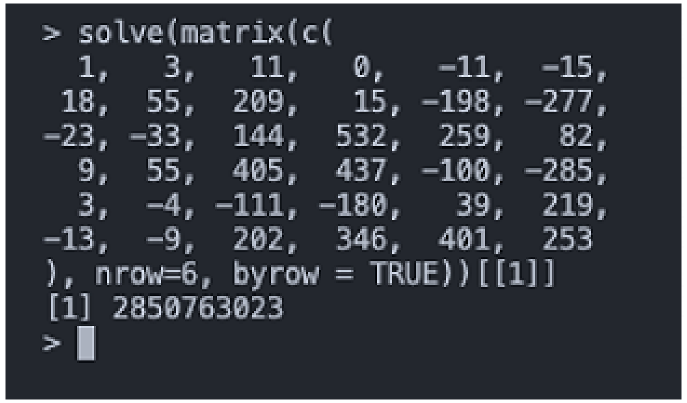
- System libraries: GSL, NLopt, BLAS/LAPACK
- Operating system: Windows, macOS, Linux

Tools: Virtual Machines, Docker, Nix/Rix.

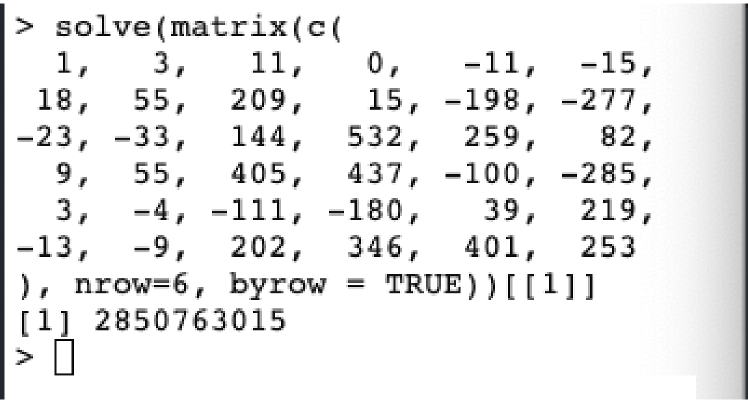
- **Very slow** to reproduce environment in full, especially without caching!
- Difficult to use, very steep learning curve.

Binary-level differences

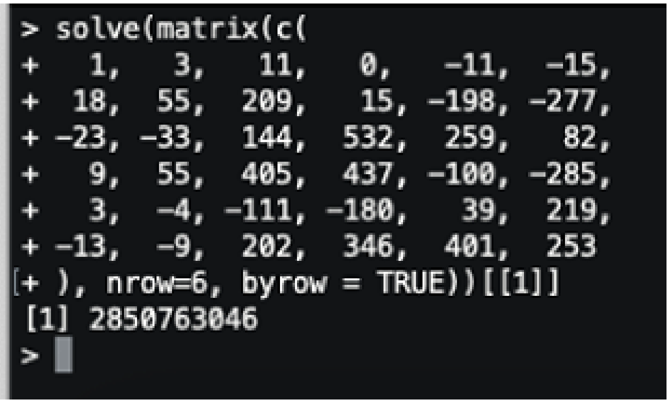
The same software can give different binary output depending on the type of hardware (e.g. ARM vs x86_64 vs RISC)



```
> solve(matrix(c(
   1,   3,   11,    0,   -11,  -15,
  18,  55,  209,   15, -198, -277,
 -23, -33,  144,  532,  259,   82,
   9,  55,  405,  437, -100, -285,
   3,  -4, -111, -180,   39,  219,
 -13,  -9,  202,  346,  401,  253
), nrow=6, byrow = TRUE))[[1]]
[1] 2850763023
>
```



```
> solve(matrix(c(
   1,   3,   11,    0,   -11,  -15,
  18,  55,  209,   15, -198, -277,
 -23, -33,  144,  532,  259,   82,
   9,  55,  405,  437, -100, -285,
   3,  -4, -111, -180,   39,  219,
 -13,  -9,  202,  346,  401,  253
), nrow=6, byrow = TRUE))[[1]]
[1] 2850763015
>
```



```
> solve(matrix(c(
+    1,   3,   11,    0,   -11,  -15,
+   18,  55,  209,   15, -198, -277,
+  -23, -33,  144,  532,  259,   82,
+    9,  55,  405,  437, -100, -285,
+    3,  -4, -111, -180,   39,  219,
+  -13,  -9,  202,  346,  401,  253
+ ), nrow=6, byrow = TRUE))[[1]]
[1] 2850763046
>
```

More info:

🔗 https://github.com/numpy/numpy/issues/9187
🔗 Floating point determinism
🔗 What every computer scientist should know about floating point arithmetic

WebAssembly

- A portable binary code format
- Enables high-performance applications on web pages
- Near-native execution speed
- Supported by most modern browsers
- Interactive through JavaScript integration

Also provides benefits for security in the form of containerisation and sandboxing.

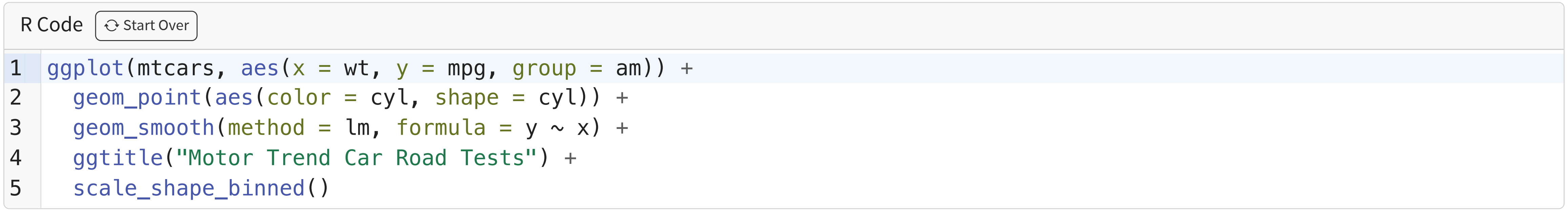
R for WebAssembly: webR

The webR project is a version of the R interpreter built for WebAssembly.

Execute R code directly in a web browser, without a supporting R server. Alternatively, run an R process server-side using Node.js

Available on GitHub and NPM as a JavaScript & TypeScript library.

Live code execution is reproducible by default



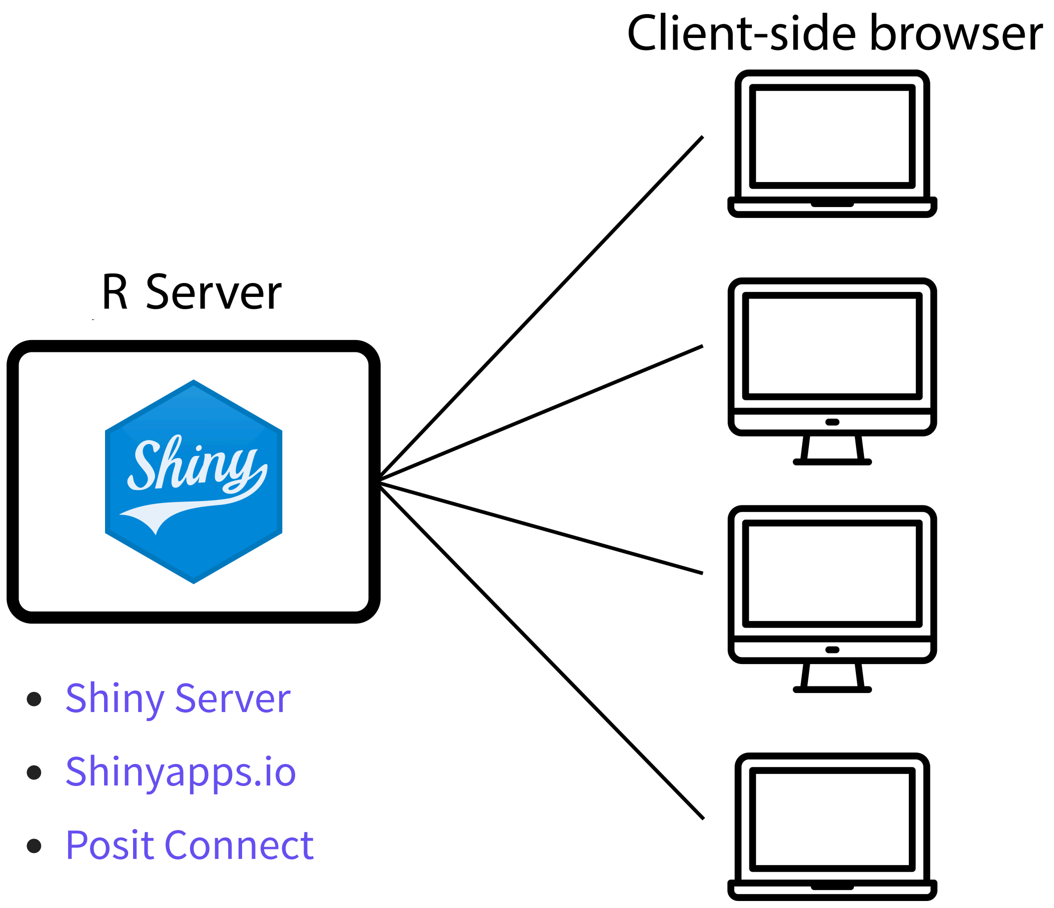
R Code
Start Over

```r
ggplot(mtcars, aes(x = wt, y = mpg, group = am)) +
  geom_point(aes(color = cyl, shape = cyl)) +
  geom_smooth(method = lm, formula = y ~ x) +
  ggtitle("Motor Trend Car Road Tests") +
  scale_shape_binned()
```

Shinylive

Run Shiny applications entirely in a web browser, without the need for a computational server.

Traditional Shiny App



- Shiny Server
- Shinyapps.io
- Posit Connect

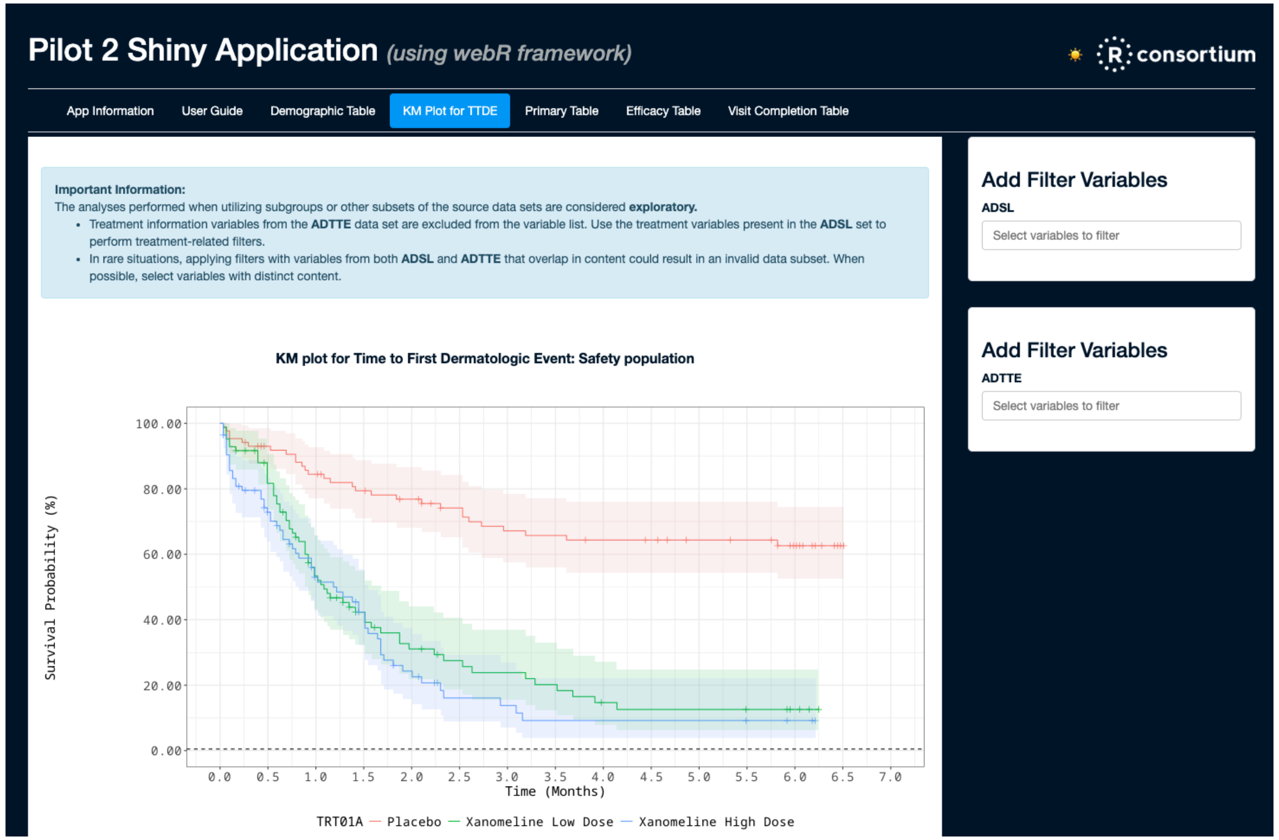
Bundle and distribute app source?

- Transfer source and data to some other machine.
- Run the app with a local Shiny server.
- Useful method for e.g. archival, app submission and review.

Requires a reproducible workflow and software installation:

- R/Python interpreter, of the correct version.
- Software and environment control: Docker, rix, renv, venv, etc.
- Software development tools: RStudio, VS Code.
- Knowledge and experience: Shiny `runApp()`, debugging.

R Consortium Submission Working Group


Pilot 2 Shiny Application *(using webR framework)*

Important Information:
The analyses performed when utilizing subgroups or other subsets of the source data sets are considered **exploratory.**

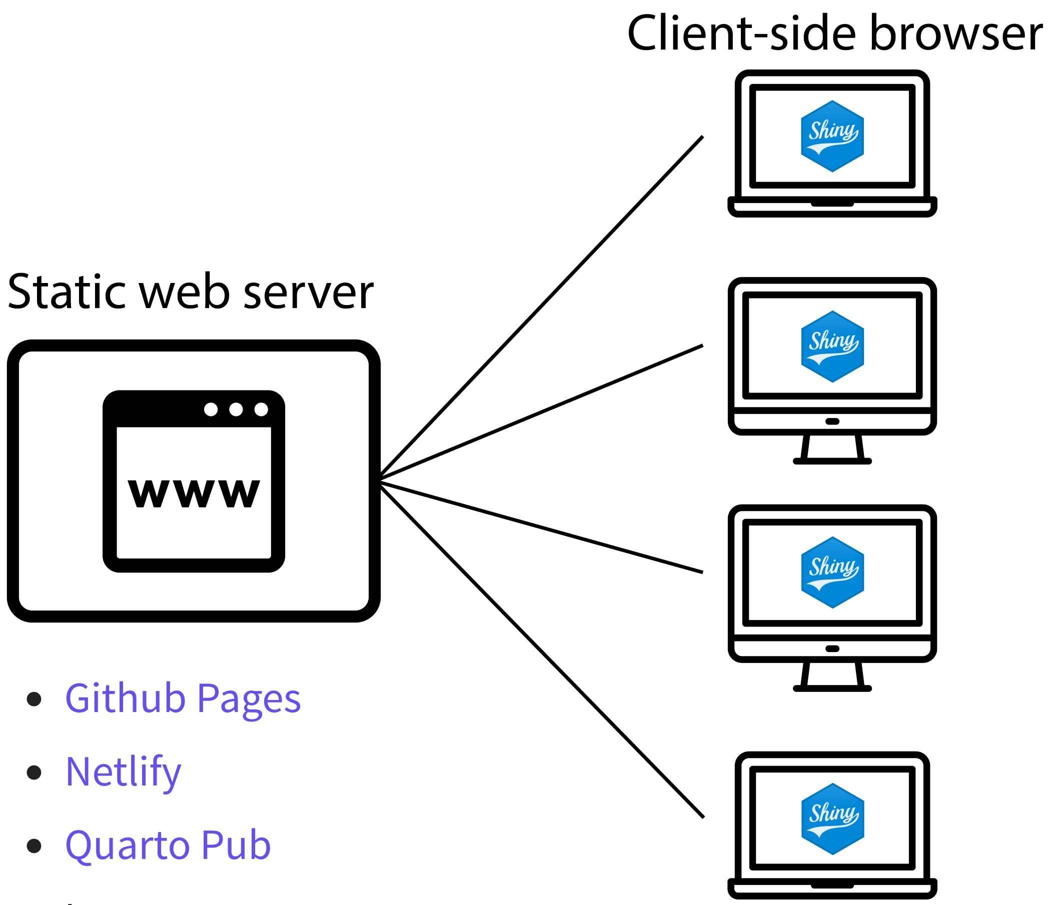
- Treatment information variables from the **ADTTE** data set are excluded from the variable list. Use the treatment variables present in the **ADSL** set to perform treatment-related filters.
- In rare situations, applying filters with variables from both **ADSL** and **ADTTE** that overlap in content could result in an invalid data subset. When possible, select variables with distinct content.

Add Filter Variables
ADSL
Select variables to filter

Add Filter Variables
ADTTE
Select variables to filter

🔗 Testing Containers and WebAssembly in Submissions to the FDA - pharmaverse.github.io

Shinylive App 🔗 https://shinylive.io/r/

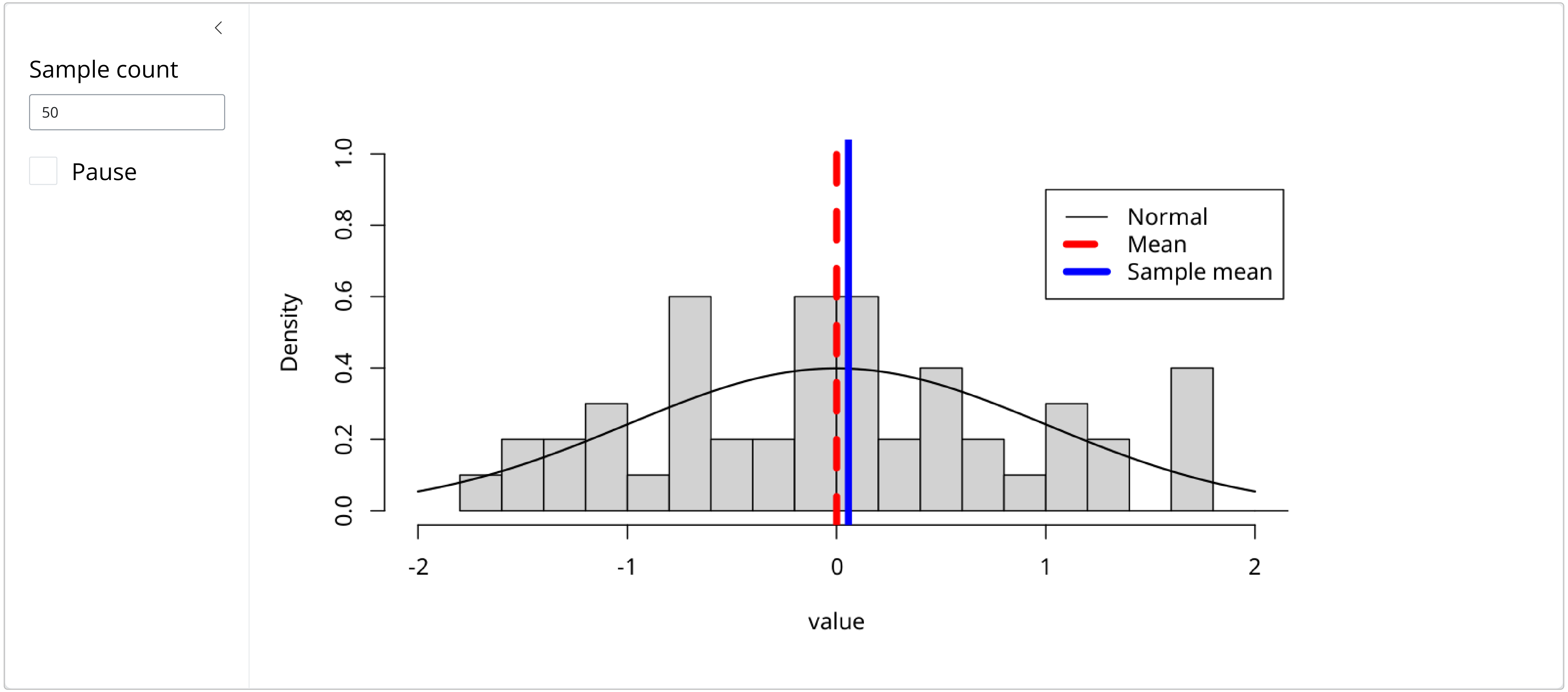


- Github Pages
- Netlify
- Quarto Pub
- Lots more…

Shinylive in Quarto

~~~
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor
incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam.

```{shinylive-r}
#| standalone: true
library(shiny)

# Create Shiny UI
ui <- [...]

# Create Shiny server function
server <- function(input, output, session) {
  [...]
}

# Build Shiny app
shinyApp(ui = ui, server = server)
```

Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu
fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in
culpa qui officia deserunt mollit laborum.
~~~

Shinylive for R


Sample count

50

Pause

Convert a Shiny app to Shinylive

Install the Shinylive R package:

```r
install.packages("shinylive")
```

Convert the app:

```r
shinylive::export("myapp", "site")
```

Binary bundle ready to transfer to another machine or host on a static web service.

```r
httpuv::runStaticServer("site")
```

WebAssembly R packages

Binary R packages for Wasm are available from a CRAN-like CDN:

- 🔗 https://repo.r-wasm.org
- Over 65% of CRAN packages available for webR
- Build custom Wasm package repositories at 🔗 https://r-universe.dev

What if R packages change?

- R packages are always updating and changing.
- Wasm R package binaries will be frozen and bundled with an app automatically.
- Apps will continue to work in the future, even as Wasm package repositories update.

Future work and current issues

- Not all R packages work under WebAssembly.
- Building custom R packages using GitHub Actions is still experimental.

- There will always be good reasons to use a traditional Shiny deployment.
- Browser security restrictions: limited networking, no raw socket access.

- 😱 There are no secrets with a Shinylive app!
- All code and data is sent to the client, deploy accordingly.

WebR demo website

https://webr.r-wasm.org/v0.4.0/

Shinylive examples

https://shinylive.io/r/

Documentation

https://docs.r-wasm.org/webr/v0.4.0/

https://github.com/posit-dev/shinylive

https://github.com/quarto-ext/shinylive