Paper CM10

Custom Monitoring Solutions in R: How to create powerful dashboards using R Markdown

Kelly O'Briant, RStudio PBC, USA

ABSTRACT

R Markdown provides an authoring framework for doing reproducible work in data science using open source programming tools and package libraries. R Markdown documents are created by interweaving executable code chunks with plain text. These computational documents can be used to produce static or interactive output formats. R Markdown can be leveraged to create dashboards for monitoring batch or real-time data flows. This paper will cover an introduction to the flexdashboard R package library, which provides a quick and easy dashboard output format for R Markdown.

INTRODUCTION

At RStudio, our thesis is that data science, as a scientific process, requires writing code. We build free and open source software that serves the day to day needs of data scientists who do their work in code. Data science is all about finding meaningful opportunities in the systems that exist around us. Scientific data discovery is inherently a complex and messy process. Without process and rigor, reproducibility is at risk, which can have serious implications ranging from financial, to a crisis of core credibility.

Great data science work should be reproducible. Being able to repeat experiments is the foundation of all science. Reproducing work is also critical for business applications: scheduled reporting, team collaboration, project validation (RStudio 2020b).

At the 2019 *RStudio Conference* in Austin Texas, Dr. Garrett Grolemund, Data Scientist and Lead Educator at RStudio, presented a talk titled, *"R Markdown: The bigger picture"*, which details a compelling argument about why it is critical for data science to be reproducible and how the R statistical programming language leverages an open source package ecosystem to support those reproducibility ideals:

Statistics has made science resemble math, so much so that we've begun to conflate p-values with mathematical proofs. We need to return to evaluating a scientific discovery by its reproducibility, which will require a change in how we report scientific results. This change will be a windfall to commercial data scientists because reproducible means repeatable, automatable, parameterizable, and schedulable. (Grolemund 2019b)

R for the Development of Reproducible Monitoring Frameworks

The vibrant R community that exists in clinical data science and health care industries has been leading the way when it comes to thinking about reproducibility. The *R in Pharma* conference held annually in Boston Massachusetts, brings together industry leaders who share how R and open source tooling is used at the forefront of innovation within their organizations. At the most recent meeting, in August 2019, several of the invited speakers presented on the use of R as applied to monitoring systems.

Rebecca Krouse, Senior Biostatistician at Rho Inc, presented a talk titled, *Building Open Source Tools for Safety Monitoring.* This presentation featured an introduction to an open source R Package called safetyGraphics which aims to be a framework for evaluation of clinical trial safety. She shared the American Statistical Association (ASA) Biopharm Safety Working Group Mission, defined as: *Empower the biostatistics community and interdisciplinary safety management partnerships to better enable qualitative and quantitative safety evaluation throughout drug development life cycle* ("ASA Biopharm Safety Working Group" 2011). This mission charter has inspired the establishment of an Interactive Safety Graphics (ISG) task force at Rho, which aims to satisfy a number of guiding principles: highly collaborative, open source, interactive, rooted in medical practice, easy to use and customize, compliant with data standards, agile, and engaging (Krouse 2019). Projects like development of

the safetyGraphics package represent the work of a combined group of stakeholders from across the pharmaceutical industry, including the FDA, and represents a keen interest in open source languages like R.

Jeremy Wildfire, Senior Data Scientist at Rho Inc, presented on a similar body of work at the 2019 RStudio Conference in Austin Texas. He described the problems facing clinical data scientists as follows:

Clinical trial research is highly regulated and notoriously slow moving. As a result, modern data analysis tools and techniques that have become commonplace in other industries have struggled to achieve broad or consistent adoption. This flexible technical framework, combined with an open source development workflow in GitHub, may provide a useful road map for creating similar tools in other change-resistant environments. (Wildfire 2019)

The actual implementation of these ideas and frameworks also requires defining infrastructure goals. Krouse also described how infrastructure plays a role in the success of getting R adopted at Rho through the Interactive Safety Graphics Task Force. Infrastructure goals were presented in three buckets: accuracy, reproducibility, and extensibility. Accuracy components included such things as unit testing, continuous integration, peer code review, regression testing, community pilot testing, and an R validation effort. The reproducibility components included the ability to export reports containing the code to recreate charts in R and application pipeline recreation as individual R functions. Finally, extensibility included goals around platform generalization (as opposed to bespoke, specialized application development) as well as extending support for many different chart types (Krouse 2019).

R Markdown for Reporting

As the previous section illustrates, open source languages like R and the associated tooling appear to be gaining traction in the clinical data science space. Practitioners claim that open source adoption encourages more collaboration and innovation, and that community collaborations address a common need where widespread support results in a highly vetted toolset (Krouse 2019).

It is now common to see the success of these pipelines or frameworks, as evidenced by the excellent body of work presented annually at the R in Pharma conference. These success stories typically include the production of an R package or an interactive Shiny application. What often gets looked over as a valid and potentially powerful component of a monitoring framework, is the use of computational documents.

We tend to see people learn Shiny, and then use it for everything. It's very hard to walk back from Shiny once you've adopted a shiny-based workflow. In other words, once you've committed to building tools with the shiny R package, it's difficult to step back and evaluate whether you would be better served delivering your desired user experience in another format.

Computational Documents

In September of 2019, I worked with Garrett Grolemund and Thomas Mock, my colleagues at RStudio, to produce a three-part webinar series all about computational documents and their use in production systems:

Computational documents offer limitless opportunities for your business. With them, your consumers can rerun your report with new parameters, apply your analysis to new data, or schedule future, automatic updates to your work—all with the click of a button (Grolemund 2019a).
Computational documents can be generated with many open source tools. In R, they are typically created with the rmarkdown package, referred to as *R Markdown*. In earlier works, Grolemund has made the case for R Markdown workflows because of the similarity to a classic lab notebook, commonly kept by lab scientists. Doing analysis in R Markdown is powerful because it allows for a shared space to both develop code and record prose or supply written commentary. The thinking is that you are more likely to come up with a strong analysis if you record your thoughts as you go, and continue to reflect on them. This also saves time when you eventually write up your analysis to share with others (Wickham and Grolemund 2017).

R Markdown for Dashboards

Another powerful use for R Markdown documents is to turn them into dashboards. This can be easily achieved with the help of the flexdashboard package, a solution for creating basic interactive dashboards in R, both with and without the use of a Shiny runtime.

The flexdashboard package can be used to publish a group of related data visualizations as a dashboard, and it supports a wide variety of components including htmlwidgets; base, lattice, and grid graphics; tabular data; gauges and value boxes; and text annotations (RStudio 2020a).

To lay out these components, you can choose between a row or column orientation. Output components delineated using level 3 markdown headers (###). The width of charts in flexdashboard is ultimately determined by the width of the browser. If your layout has a single column then charts will occupy the full width of the browser. If your layout has multiple columns then the columns will split the available width evenly.

If you have more than a handful of charts you'd like to include in a dashboard, you may want to consider dividing the dashboard into multiple pages. To define a page just use a level 1 markdown header (==========). Each page you define will have its own top-level navigation tab.

A variety of themes are available to modify the base appearance of flexdashboard. To set a theme, you must specify it in the YAML header with the theme parameter:

```
---
title: "Themes"
output:
  flexdashboard::flex_dashboard:
    theme: bootstrap
---
```

Numerous example layouts are available on the documentation website (RStudio 2020a)

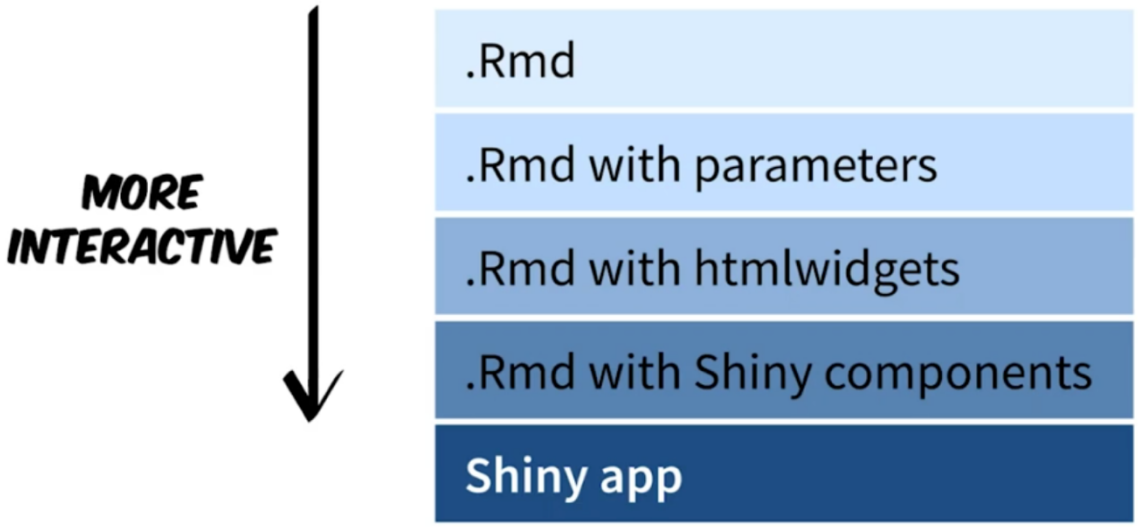
Using Shiny with flexdashboard

By adding Shiny to a flexdashboard, you can create dashboards that enable viewers to change underlying parameters and see the results immediately, or that update themselves incrementally as their underlying data changes (see reactiveFileReaderand reactivePoll). This is done by adding runtime: shiny to a standard flexdashboard and then adding one or more input controls and/or reactive expressions that dynamically drive the appearance of the components within the dashboard. Using Shiny with flexdashboard turns a static R Markdown report into an Interactive Document. It's important to note that interactive documents need to be deployed to a Shiny Server to be shared broadly (whereas static R Markdown documents are standalone web pages that can be attached to emails or served from any standard web server) (RStudio 2020a).

What we want to avoid is including a Shiny runtime, or level of interactivity, that isn't necessary to achieve your goal or desired user experience.

CONCLUSION

In the RStudio Webinar, *Interactivity in Production*, presented in October 2019, I used Garrett Grolemund's "Hierarchy of Interactivity" (Figure 1), to explore how a monitoring application, originally prototyped in Shiny, could be refactored into a less complex R Markdown dashboard, and rendered using a scheduler to produce custom report outputs and email alerts.



Hierarchy of Interactivity (Grolemund 2019a)

Interactive products take your data science to a new level, but they require new coding decisions. Maybe you started with a Shiny Application, but found out along the way, that it wasn't actually the best medium for communication. An overlooked piece of "Shiny in Production" is evaluating whether you built the right thing. Tailor the experience to your audience, and the design solution to your functional goals and infrastructure. (O'Briant 2019)

As interactivity is added, the complexity of your solution and the infrastructure needed to support it increases accordingly. Increased complexity can inhibit other goals as well. It can make reproducibility harder to achieve or justify, and therefore be an unnecessary additional barrier when you're trying to produce a minimal viable product or generate buy-in for the adoption of open source tooling.

Computational documents, like R Markdown dashboards, can often be used in place of fully reactive applications to create powerful monitoring solutions that can also be productionized without the same infrastructure complexity required for the same solution built as a Shiny Application.

REFERENCES

"ASA Biopharm Safety Working Group." 2011. ASA Biopharmaceutical Section. https://community.amstat.org/biop/workinggroups/safety/safety-home.

Grolemund, Garrett. 2019a. "Reproducibility in Production." RStudio, PBC. https://resources.rstudio.com/webinars/reproducibility-in-production.

———. 2019b. "R Markdown: The Bigger Picture." RStudio, PBC. https://resources.rstudio.com/rstudio-conf-2019/r-markdown-the-bigger-picture.

Krouse, Rebecca. 2019. "Building Open Source Tools for Safety Monitoring: Advancing Research Through Community Collaboration." Rho, Inc. https://github.com/rinpharma/rinpharma2019program.

O'Briant, Kelly. 2019. "Interactivity in Production." RStudio, PBC. https://resources.rstudio.com/webinars/interactivity-in-production.

RStudio. 2020a. "Flexdashboard Documentation." RStudio, PBC. https://rmarkdown.rstudio.com/flexdashboard/index.html.

———. 2020b. “Reproducible Environments: Manage Environments for Data Science.” RStudio, PBC. https://environments.rstudio.com/.

Wickham, Hadley, and Garrett Grolemund. 2017. *R for Data Science*. 1st ed. O’Reilly Media. https://r4ds.had.co.nz/.

Wildfire, Jeremy. 2019. “Modernizing the Clinical Trial Analysis Pipeline with R and Javascript.” Rho, Inc. https://github.com/RhoInc/RStudioConf2019-ePoster.

ACKNOWLEDGMENTS

This paper has drawn on packages, vignettes, and work produced by many individuals employed by RStudio, a Public Benefit Corporation. RStudio documentation is written and maintained by groups of individuals over time, including members of various engineering teams, including the team I am affiliated with, RStudio Solutions Engineering.

CONTACT INFORMATION

Comments and questions can be directed to the following individuals at RStudio, PBC:

Phil Bowsher, Director of Life Sciences & Healthcare
phil@rstudio.com
Web: linkedin.com/in/philip-bowsher-67151015/

Kelly O’Briant, Product Manager
kelly@rstudio.com
Web: solutions.rstudio.com