Masters introduction

Abstract

Statisticians and analysts have been using R for a long time. R programming environment has reproducible document standards embedded, such as R markdown. While R markdown requires the programmer to manually construct the structure of reproducible file, R list-down Package provides a programmatic solution to generate the files. On top of the already available R list down package, this paper demonstrates implanting graphs and interactive plots when users generate any reproducible documents by using trelliscopejs package. The concept is followed by a demonstration using gapminder dataset.

Introduction

The R markdown (Baumer, Cetinkaya-Rundel, Bray, Loi, and Horton, 2014) demonstrates the possibility of constructing reproducible documents using R language. The format allows author to integrate R codes, written work, data tables, visualization plots and much more information into one directly structured document. Amongst scientific writings and analytical reports knitted using R markdown, the majority are made up of codes and narrative writings. The codes are usually but not limited to common computing languages, such as R, Python, SQL and others. The writings which contextualize the codes are usually inserted before ot follow the code chunks spatially. Sometimes writings are also placed between the codes using a hash. In this paper, the chunk of computing codes and the narrative writings will be referred to computational components and narrative components respectively (Kane,Jiang and Urbanek, 2020).

R markdown provides several output formats when users knit the file into output documents.

R markdown

Statistical analysis tends to be more reachable and interpretable to public audience accompanied by the rapid rise of computing abilities. While the statistical computing threshold lowers with the invention of R language, the needs to integrate computationally derived objects with narrative explanations arise. The usage of R markdown centralizes different data types with a specific format, further process with the technical work in a computationally organized way.

The first thing is data cleaning and tidying. However, this processes usually requires other environments and configuration to manipulate. Such steps have different needs in computational purpose than report or presentations.

Before a highly informative presentation, multiple explanatory analyses are often carried out. These explanatory analyses contain numerous amounts of table, plots and graphs, most of them have comments and notes. R markdown can save the robust components and generate documents without spending time on layouts and formats.

The advantage of R markdown’s narrative feature can be demonstrated in multiple prospects. Firstly, R markdown is as its names implies, a Markdown mark-up language of R. By using combinations of codes and embedding symbols to control the formatting and layout of a R object file, users gets the desired final output document. All of these are easy to achieve without extensive skills in coding and consumes less time and effort to learn. Packages such as “bookdown” (Xie, 2016) demonstrates the easiness for users indenting to edit long narrative components in contrast of using another commonly-used markup language – Latex. Latex

whereas the syntax is relatively more complicated.

On the other hand, statistical reports and presentations often contains numerous computational and narrative components. R markdown supports multiple file formats once a R object is completed and read for publishing. The function in R markdown for this objective is called “knit”. The process includes running all computational components, then formats the outputs along with the narrative components.

Each computational component starts with ```{}, with the language name between the curly brackets and ends with ```. During the knitting process, computation components along with their results are laid out precedingly. Each narrative component, which is so called ordinary text with out code is combined within the computational components, resulting in a desired file.

The final result is also customizable not only as in pdf or HTML format, but also includes editable formats such as Microsoft word documents. This conforms with the intention for a typical statistical report or presentation, that is, to make audience understand statistics with less to no statistical knowledge (Baumer, Cetinkaya-Rundel, Bray, Loi, and Horton, 2014). R markdown fulfils this concept by offering modifiable documents for collaborators and other users to develop narrative components based on the statistical analysis results generated beforehand.

While PDF and Word formats are commonly used by researchers in the fields for their formatting specification, they do not provide a pragmatical solution for interactive plots and graphics. Compared to static graphics, interactive graphics are extremely powerful for explanatory analysis, and complements the visualization prospect of statistical visualizations (Theus & Urbanek, 2008). HTML, another format producible by R markdown, is often underestimated in its ability for interactive visualizations. R has numerous packages that supports interactive widgets which can by knitted and shown on HTML webpages. This paper will demonstrate how the interactive graphics can be implemented into HTML using package Listdown.

R Listdown

Nonetheless, the data cleaning and other processes iteratively repeats each time generating documents for presentation whenever knitr is used in a R markdown file. A R markdown file usually lacks semantic structure. When all computational components and narrative components are stored in one single file manually paragraphed by the author, extracting and editing components partially in a R markdown file often leads to increase of workloads after the changes are committed.

In addition, R markdown does not hold any data dedicated for the file itself. In order for a computational component in R markdown that reads in the data to work, the data set has to be stored or set to a pathway specifying the location of the data, depending on either it is saved locally or on a server.

To overcome the above mentioned aspects, package listdown (Kane,Jiang and Urbanek, 2020) was introduced providing functions to programmatically create R markdown files from named lists. By using functions from the package, the components can be turned into a single named list, organized in a hierarchical structure. The contents of each list denoted, including the name and type of R object can be viewed in dendrograms. On top of the lists, decorators and other customizable functions can be added to assist the problem of visualizing. This is particular useful when large datasets are added to its corresponding computational component list and the author intends to present them. Large data sets requires a substantial amount of space to be fully shown.

During a statistical analysis, the analytical process and results can be seen as two parts which can be stored separately. If the results contain graphs and plots, they could be further stored in single named lists. As everything is organized in named lists, the package has several advantages compared to normal R markdown files containing all computational and narrative components. The first is it allows multiple pathways working in parallel from the same data. When the experiment and objective is different based on the same data, computational components are expected to be different. This will affect the narrative components such as conclusions and discussions, but the data and other process remains constant. Since the different “pathways” can be stored into different lists, listdown package allows users to selectively pick the reproducible lists along with the narrative components. Normal R markdown files shows the experiment in a serial way if different experiments are stored in the same file, or users will have to open two R markdown files with the same computational component for processing in both R markdown files.

Another advantage for storing objects in a list is its capability to avoid repetitive work when data analysis updates. This is partially useful when data analysis process is updated frequently while the data source remains in the same format and standard. Once the computational components for data cleaning and process are constructed and stored into the list, the analysis may change the outputs, such as results and plots. Further changing the narrative components. However, updating analysis methods does not mean deprecating the previous methods, listdown package allows different methods to be stored and reproducible at anytime with a single knitr, this vastly improves efficiency and drops tedious repetitive works while maintaining the objectives desired.

Some useful areas of statistical analysis benefiting from the listdown package are mainly but not limited to it’s usage in clinical trial data (Kane,Jiang and Urbanek, 2020).

The list-down package

The basic function of the listdown package consists of 6 distinctive methods of which can be implanted into piping commands by various packages once the analysing and technical works are completed. The methods are (Kane,Jiang and Urbanek, 2020) :

• as ld yml() - turn a computational component list into YAML with class information

• ld cc dendro() - create a dendrogram from a list of computational components

• ld chunk opts() - apply chunk options to a presentation object

• ld ioslides header() - create an ioslides presentation header

• ld make chunks() - write a listdown object to a string

• ld rmarkdown header() - create an R Markdown header

• ld workowr header() - create a workflowr header

• listdown() - create a listdown object to create an R Markdown document

Trelliscopejs

Visualization using trelliscopejs package in list-down

2.

2.1Basic explanation of the list-down package

2.2workflow

As previously mentioned, the workflow for a typical scenario involving using list down is after the analytical and technical works are finished. Suppose when we finished a serial of work including codes, narrative explanation and other contents. All the results have been collected and all plots have been plotted, in our desirable order for publish or presentation, a document including multiple computational components is formed. The computational components are likely to be stored in different locations or even on different servers. By organizing them into a list format eases the processing and returns a less nested structure.

Once a list is created, the document containing one or multiple lists can be saved as RDS file format ready by the output R markdown document. Then we can use listdown() to specify how the computational component will be loaded, along with library needed, the decorators and other attributes presented once the M markdown document is knitted.

In the example, an explicit function was written named rld. The function shows a general syntax for rendering a listdown file into a html document. The ld\_make\_chunks() allows authors to add in extra component such as a header to the list of computational components.

3. Rendering and formatting

4. Adding plots to the listdown object

4.1 Trelliscopejs example

5. plotly example

6.conclusion

Baumer, B., Cetinkaya-Rundel, M., Bray, A., Loi, L., & Horton, N. J. (2014). R Markdown: Integrating a reproducible analysis tool into introductory statistics. *arXiv preprint arXiv:1402.1894*.

Kane, M. J., & Urbanek, S. (2020). On the Programmatic Generation of Reproducible Documents. *arXiv preprint arXiv:2007.12631*.

Baumer, B., & Udwin, D. (2015). R markdown. *Wiley Interdisciplinary Reviews: Computational Statistics*, *7*(3), 167-177.

(good reproducibility)

Xie, Y. (2016). *Bookdown: authoring books and technical documents with R markdown*. CRC Press.

(easier than latex)

Theus, M., & Urbanek, S. (2008). *Interactive graphics for data analysis: principles and examples*. CRC Press.

(interactive graphics)

Healy, K. (2018). *Data visualization: a practical introduction*. Princeton University Press.

Wilke, C. O. (2019). *Fundamentals of data visualization: a primer on making informative and compelling figures*. O'Reilly Media.

Cook, D., Swayne, D. F., & Buja, A. (2007). *Interactive and dynamic graphics for data analysis: with R and GGobi*. Springer Science & Business Media.

Sievert C (2020). Interactive Web-Based Data Visualization with R, plotly, and shiny. Chapman and Hall/CRC. ISBN 9781138331457, [https://plotly-r.com](https://plotly-r.com/)

*Jennifer Bryan (NA). gapminder: Data from Gapminder.*

*#> https://github.com/jennybc/gapminder,*

*#> http://www.gapminder.org/data/,*

*#> https://doi.org/10.5281/zenodo.594018.*

Ali, S. M., Gupta, N., Nayak, G. K., & Lenka, R. K. (2016, December). Big data visualization: Tools and challenges. In *2016 2nd International Conference on Contemporary Computing and Informatics (IC3I)* (pp. 656-660). IEEE.

Weissgerber, T. L., Garovic, V. D., Savic, M., Winham, S. J., & Milic, N. M. (2016). From static to interactive: transforming data visualization to improve transparency. *PLoS biology*, *14*(6), e1002484.

Hafen, R., Gosink, L., McDermott, J., Rodland, K., Kleese-Van Dam, K., & Cleveland, W. S. (2013, October). Trelliscope: A system for detailed visualization in the deep analysis of large complex data. In *2013 IEEE Symposium on Large-Scale Data Analysis and Visualization (LDAV)* (pp. 105-112). IEEE.

Wickham, H. (2011). ggplot2. *Wiley Interdisciplinary Reviews: Computational Statistics*, *3*(2), 180-185.