A deeper dive into R Markdown and LaTeX

Tad Dallas

Reading for this week:

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.

R markdown

I have used R markdown syntax to create these lecture notes, and you may have had experience using R markdown when completing the course assignments. I would like to take this week to go into further detail about the use of R markdown and about typesetting in general. R markdown is a pretty handy way of combining data, code, and text to create a single document of analyses and interpretation. This is incredibly useful and can ensure that your analyses are transparent, and can be easily edited if the raw data change (e.g., a dynamic or living analysis, which can be updated with a single command).

R markdown files have the `.Rmd` file extension, but are simply text files, containing text and `R` code weaved together. One nice thing about R markdown files is that they can be compiled to html, pdf, or even Microsoft Word (booooo). One thing that many people do not realize, but is kind of neat. R markdown files can also run code chunks in other languages, in the same document. So your Python code, bash scripts, and `R` code can all be combined into a single document, which can be compiled to produce a beautifully typeset blend of text and code.

What can R markdown do

R markdown documents can be compiled to create documents, analyses, research articles, books, theses/dissertations, presentation slides, websites, CV/resumes, etc. etc.

How does R markdown do these things?

By standing on top of the shoulders of LaTeX and pandoc for the most part.

What is LaTeX?

A typesetting language to give users the ability to use code to produce exactly the document they want in terms of formatting. Much like `R`, it benefits from a diverse set of add-on packages to expand the functionality of the core language. I will not focus on LaTeX formatting, as it is a bit beyond the scope of this course, and it is honestly not for everyone. Personally, I am a huge fan of LaTeX and will likely continue to write all manuscripts, presentations, and professional documents as tex files. However, I recognize that many researchers simply do not see the benefit in using LaTeX when it requires time/energy, and their collaborators may not use LaTeX. Many of the benefits of using R markdown are really simply benefits of using LaTeX, as

R markdown with pdf output actually is R markdown to LaTeX to pdf through pandoc and a latex engine. So even when you are not using LaTeX, you are kind of using LaTeX.

This interface between R Markdown and LaTeX is important

Writing a research article in R Markdown

Header

```
---
title: "Your title"
author: "Name"
date: "July 28, 1942"
output: html_document
---
```

Preamble

R Markdown makes a lot of decisions for you, using a basic template which sets spacing, font, and formatting defaults. You will likely want to change these defaults when you are writing your article, as they might not fit your specifications. To do this, we can use a preamble. This is where that LaTeX integration will show itself, if you are compiling to pdf.

There are at least two spots where you can alter base formatting. First, in the yaml header of the R markdown document itself.

```
---
output:
  pdf_document:
    citation_package: natbib
    keep_tex: true
    fig_caption: true
    latex_engine: xelatex
    template: svm-latex-ms.tex
title: "Title"
author:
- name: your name
  affiliation: your uni
- name: another name
  affiliation: another uni
date: "`r format(Sys.time(), '%B %d, %Y')`"
geometry: margin=1in
fontfamily: mathpazo
fontsize: 11pt
spacing: double
bibliography: yourBib.bib
biblio-style: apa
---
```

Second, by writing a separate `.tex` file containing LaTeX code specifying document preamble structure. This is useful when you want to call specific packages (e.g., `lineno` to add line numbers).

```
output:
  pdf_document:
```

```
    includes:
      in_header: "preamble.tex"
```

where the `preamble.tex` file may look something like

```
\usepackage{geometry}
\geometry{a4paper}
\usepackage[numbers]{natbib}
\usepackage{graphicx}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage{textcomp}
\usepackage{gensymb}
\usepackage{amsmath}
\usepackage{amssymb}
\usepackage[running]{lineno}
\usepackage{setspace}

\doublespacing
```

It may be difficult to really think about all the formatting before writing the actual document, and using typesetting languages does require a bit of time to get used to... but it is definitely worth it.

Code chunks and chunk options

Code chunks live within your R markdown document. Each argument to the code chunk can be used to modify the output. For instance, if you do not want to show the `R` code, set `echo=FALSE`. Code chunk options could also be used to change figure dimensions/resolution, code output, etc. For more code chunk options, see `knitr` documentation (https://yihui.org/knitr/options/).

```
opts_chunk$set(fig.width=8, fig.height=5,
    echo=TRUE, warning=FALSE,
    message=FALSE, cache=TRUE)
```

Citations

Citations in R markdown use a LaTeX/bibtex backbone. This requires a supplemental file in the same directory as the `.Rmd` file, with a `.bib` file which contains the formatted references in a specific format.

```
Citations in R markdown are fairly straightforward [@dallas2020]
```

This specific format can be output from Google Scholar (the small quote symbol below each article, followed by clicking the "BibTeX" hyperlink). Each formatted citation in the `.bib` file has the same structure. For example, The citation above (@dallas2020) refers to the `.bib` entry below.

```
@article{dallas2020,
  title={Spatial synchrony is related to environmental change in Finnish moth communities},
  author={Dallas, Tad A and Ant{\~a}o, Laura H and P{\"o}yry,
        Juha and Leinonen, Reima and Ovaskainen, Otso},
  journal={Proceedings of the Royal Society B},
  volume={287},
  number={1927},
  pages={20200684},
  year={2020},
```

```
  publisher={The Royal Society}
}
```

Tables

```
tab <- data.frame(
    Variable=c('temperature', 'precipitation', 'ice cream'),
    t=c(0.01, 2.05, -1.49),
    p = c(0.01, 0.25, 0.86))
knitr::kable(tab)
```

Variable	t	p
temperature	0.01	0.01
precipitation	2.05	0.25
ice cream	-1.49	0.86

Variable	t	p
temperature	0.01	0.01
precipitation	2.05	0.25
ice cream	-1.49	0.86

It is also possible to use `xtable` for formatting tables in LaTeX format instead of markdown syntax.

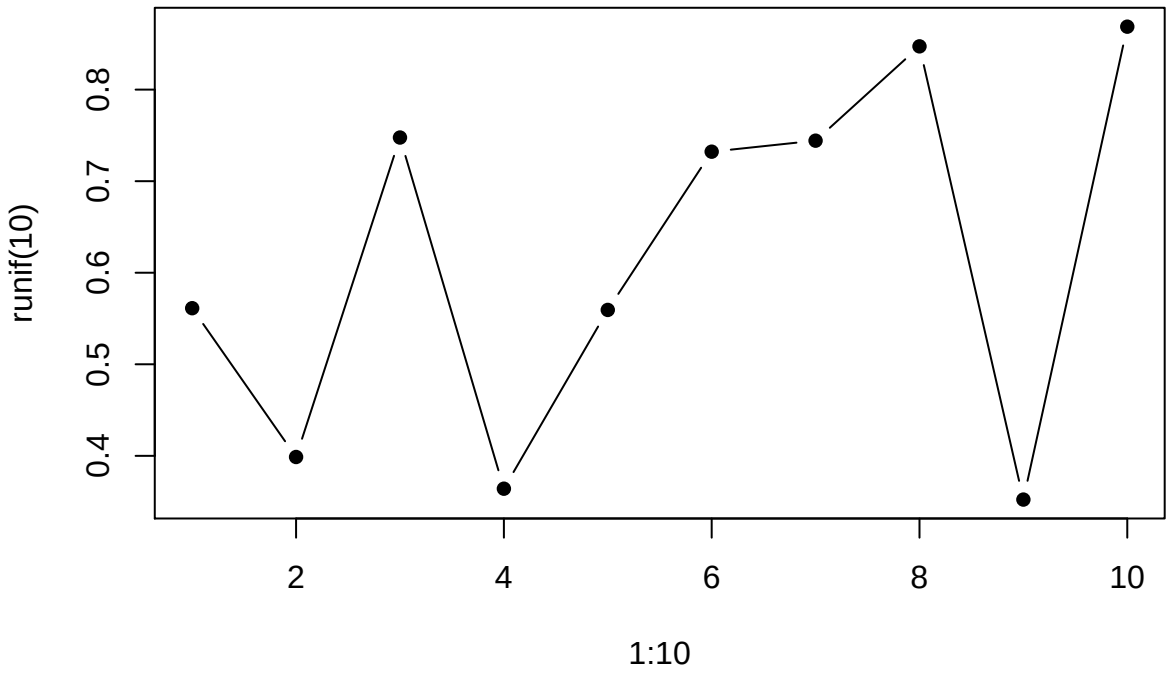
```
xtable::xtable(tab)
```

```
## % latex table generated in R 3.6.3 by xtable 1.8-4 package
## % Tue Jul 14 15:25:39 2020
## \begin{table}[ht]
## \centering
## \begin{tabular}{rlrr}
##   \hline
##  & Variable & t & p \\
##   \hline
## 1 & temperature & 0.01 & 0.01 \\
##   2 & precipitation & 2.05 & 0.25 \\
##   3 & ice cream & -1.49 & 0.86 \\
##    \hline
## \end{tabular}
## \end{table}
```

Figures

Figures can either be created within the R markdown document using `R` code, or can be imported from pdf or other format using the `knitr` backend.

```
plot(1:10, runif(10), pch=16, type='b')
```




```
knitr::include_graphics("path/to/file")
```

Equations

```
$$\sum_{i=1}^n X_i$$
```

$$\sum_{i=1}^{n} X_i$$

table/figure/equation referencing

Referencing of specific bits (tables, figures, equations) can be done by referencing code chunks by name. Naming code chunks can be done by entering text immediately after the language declaration (e.g., ““‘{r codeChunkName}”).

```
plot(1:10, runif(10), pch=16,
    type='b', col=rainbow(10))
```

I have made a figure (Figure @ref(fig:fig1)).

Customizing R markdown format

You may encounter a situation where the desired document format is not quite what is present in the default R markdown templating. This will especially be the case for dissertations, and academic journals are also pretty terrible at requiring journal-specific formatting.

```
file.copy(system.file("rmd/latex/default-1.17.0.2.tex",
          package = "rmarkdown"), "template.tex")
```

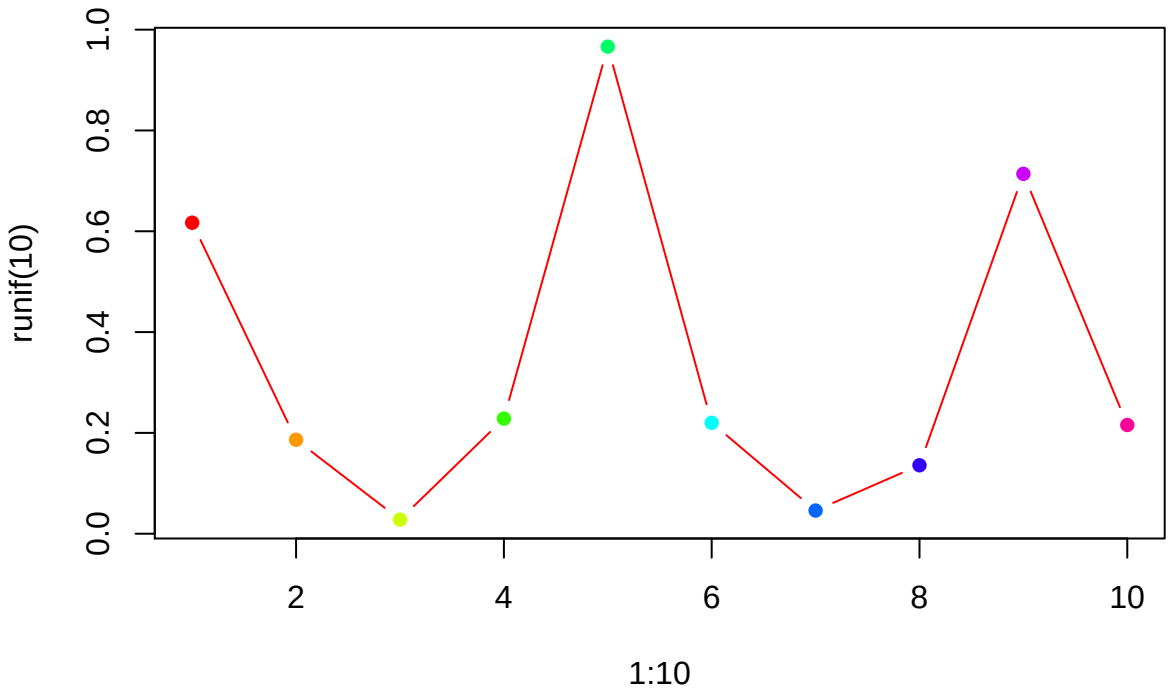


Figure 1: Figure caption

```
---
title: "Title"
author: "Name"
date: "`r format(Sys.time(), '%Y %B %d')`"
output:
   pdf_document:
      template: template.tex
      keep_tex: true
geometry: margin=1in
latex_engine: xelatex
---
```