Data Science for Public Policy

Aaron R. Williams - Georgetown University

Reproducible Research with R Markdown

Reading

- R for Data Science – Chapters 26, 27, 29, and 30
- RStudio R Markdown introduction
- Mastering Markdown

Motivation

There are many problems worth avoiding in an analysis:

- Copying-and-pasting, transposing, and manual repetition
- Out-of-sequence documents
- Parallel documents (a script and a narrative Word doc)
- Code written for computers that is tough to parse by humans

Not convinced? Maybe we just want to make cool stuff.

Literate (Statistical) Programming

Source: Jacob Applebaum

Literature Programming and LaTeX

> Let us change our traditional attitude to the construction of programs: Instead of imagining that our main task is to instruct a computer what to do, let us concentrate rather on explaining to human beings what we want a computer to do. ~Literate Programming (1984)

Example

We used a linear model because there is reason to believe that the population model is linear. The observations are independent and the errors are independently and identically distributed with an approximately normal distribution.

```
model1 <- lm(formula = dist ~ speed, data = cars)
model1
```

```
##
## Call:
## lm(formula = dist ~ speed, data = cars)
##
## Coefficients:
## (Intercept)        speed
##     -17.579        3.932
```

An increase in travel speed of one mile per hour is associated with a 3.93 foot increase in stopping distance on average.

R Markdown

R Markdown is the most popular framework for literate statistical programming in R and it is an important tool for reproducible research. It combines narrative text with styles, code, and the output of code and can be used to create many types of documents including PDFs, html websites, slides, and more.

Sweave is a competing framework that it is out-of-date and Jupyter (**Ju**lia, **Py**thon, and **R**) is a competing framework that is popular for Python but has not caught on for R.

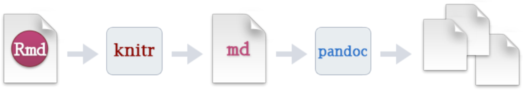
According to Wickham and Grolemund, there are three main reasons to use R Markdown:

1. "For communicating to decision makers, who want to focus on the conclusions, not the code behind the analysis."
2. "For collaborating with other data scientists (including future you!), who are interested in both your conclusions, and how you reached them (i.e. the code)."
3. "As an environment in which to do data science, as a modern day lab notebook where you can capture not only what you did, but also what you were thinking."

R Markdown uses

- plain text files ending in `.Rmd` that are similar to `.R` files.
- `library(rmarkdown)` and `library(knitr)`.
- pandoc.

`library(rmarkdown)` calls `library(knitr)` and "knits" `.Rmd` (R Markdown files) into `.md` (Markdown files), which Pandoc then converts into any specified output type. `library(rmarkdown)` and `library(knitr)` don't need to be explicitly loaded and the entire process is handled by clicking the "knit" button in RStudio.



Source: R for Data Science by Hadley Wickham and Garrett Grolemund

Clicking the "knit" button starts this process.

`library(rmarkdown)`, `library(knitr)`, and Pandoc are all installed with RStudio. The only additional software you will need is a LaTeX distribution. Follow these instructions to install `library(tinytex)` if you want to make PDF documents.

Exercise 1

1. If you already have a LaTeX distribution like `tinytext` or `MiKTeX`, then skip this exercise.
2. Follow these instructions to install `library(tinytex)`.

. . .

The "knitting" workflow has a few advantages:

1. All code is rerun in a clean environment when "knitting". This ensures that the code runs in order and is reproducible.
2. It is easier to document code than with inline comments.
3. The output types are really appealing. By creating publishable documents with code, there is no need to copy-and-paste or transpose results.
4. The process is iterable and scalable.

Exercise 2

1. Click the new script button and add a "R Markdown".
2. Give the document a name, an author, and ensure that HTML is selected.

3. Save the document as “hello-markdown.Rmd”.
4. Click “Knit”.

. . .

Three Ingredients in a `.Rmd`

1. YAML header
2. Markdown text
3. Code chunks

1. YAML header

YAML stands for “yet another markup language”. The YAML header contains meta information about the document including output type, document settings, and parameters that can be passed to the document. The YAML header starts with `---` and ends with `---`.

Here is the simplest YAML header for a PDF document:

```
---
output: pdf_document
---
```

YAML headers can contain many output specific settings. This YAML header creates an HTML document with code folding and a floating table of contents:

```
---
output:
  html_document:
    code_folding: hide
    toc: TRUE
    toc_float: TRUE
---
```

Parameters can be specified as follows

```
---
output: pdf_document
params:
  state: "Virginia"
---
```

Now state can be referred to anywhere in R code as `params$state`.

Exercise 3

1. Switch the output type to PDF and knit the document.
2. Switch the output type back to HTML.

. . .

2. Markdown text

Markdown is a shortcut for HyperText Markup Language (HTML). Essentially, simple meta characters corresponding to formatting are added to plain text.

```
Titles and subtitltes
------------------------------------------------------------

# Title 1

## Title 2

### Title 3


Text formatting
------------------------------------------------------------

*italic*

**bold**

`code`

Lists
------------------------------------------------------------

* Bulleted list item 1
* Item 2
  * Item 2a
  * Item 2b

1. Item 1
2. Item 2

Links and images
------------------------------------------------------------
```

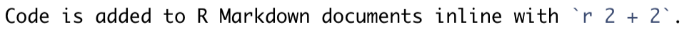
`[text](http://link.com)`

Exercise 4

1. Add text with formatting like headers and bold to your R Markdown document.
2. Knit!

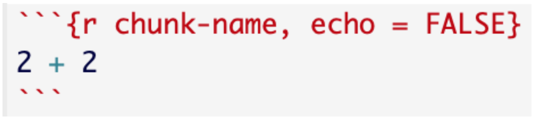
. . .

3. Code chunks



```
Code is added to R Markdown documents inline with `r 2 + 2`.
```

More frequently, code is added in code chunks:



~~~
```{r chunk-name, echo = FALSE}
2 + 2
```
~~~

The first argument inline or in a code chunk is the language engine. Most commonly, this will just be a lower case `r`. `knitr` allows for many different language engines:

- R
- Python
- SQL
- Bash
- Rcpp
- Stan
- Javascript
- CSS

The second argument inside brackets in code chunks is the chunk name. Always name code chunks. Other chunk-specific settings can be added inside the brackets. Here are the most important options:

Option	Effect
echo = FALSE	Hides code in output
eval = FALSE	Turns off evaluation of chunk
fig.height = 8in	Changes figure width
fig.width = 8in	Changes figure height

Default settings for the entire document can be changed with R code (in an R code chunk) such as the following:

```
knitr::opts_chunk$set(echo = FALSE)
```

The table added above was typed as Markdown code. But sometimes it is easier to use a code chunk to create and print a table. Pipe any data frame into `knitr::kable()` to create a table that will be formatted in the output of a knitted R Markdown document.

Exercise 5

1. Add a code chunk.
2. Load the storms data set.
3. Filter the data to only include hurricanes.
4. Make a data visualization with ggplot2 using the data from 3.
5. Include an option to hide the R code.
6. Knit!

. . .

Applications

PDF documents

```
---
output: pdf_document
---
```

- These notes!

html documents

```
---
output: html_document
---
```

- Regression in R notes
- R at the Urban Institute website

GitHub README

```
---
output: github_document
---
```

- urbnthemes

Bookdown

Bookdown is an R package by Yihui Xie for authoring books in R Markdown. Many books, including R for Data Science (GitHub) by Hadley Wickham and Garrett Grolemund, have been written in R Markdown.

Blogdown

Blogdown is an R package by Yihui Xie for creating and managing a blog in R Markdown. Up & Running with blogdown in 2021 by Alison Hill is a great tutorial for getting started with Blogdown.

Microsoft Word and Microsoft PowerPoint

It is possible to write to Word and PowerPoint. In general, I've found the functionality to be limited and it is difficult to match institutional branding standards.

Slides

```
---
output:
  revealjs::revealjs_presentation:
    css: styles.css
    incremental: true
    reveal_options:
      slideNumber: true
      previewLinks: true
---
```

Fact sheets

An alternative to knitting an R Markdown document with the knit button is to use the `rmarkdown::render()` function. This allows for rendering documents to be iterated. By passing different parameters to each rendering, it's possible to create documents for different geographies, organizations, people, or periods of time.

At the Urban Institute, we regularly iterate fact sheets at the state and county level.

- Expanding the EITC for Workers without Resident Children
- Data@Urban includes an outline.

Fact pages

It's also possible to iterate websites with `rmarkdown::render()`.

- The Urban Institute State and Local Finance Initiative creates State Fiscal Briefs by iterating R Markdown documents.
- Data@Urban

Suggestions

- Knit early, and knit often.
- Select the gear to the right of "Knit" and select "Chunk Output in Console"
- Learn math mode. Also, `library(equatiomatic)` (CRAN, GitHub) is amazing.

Resources

- R4DS R Markdown chapter
- RStudio R Markdown intro
- RStudio R Markdown gallery
- Happy Git R Markdown tutorial
- RMarkdown cheat sheet