

EC workshop: R Markdown

Annie Schiffer

2025-01-27

Outline:

- Overview of R Markdown
- Document output and knitting
 - practice: create an R Markdown document
- Basic syntax
 - practice: create an outline for an analysis
- Adding code and math chunks
 - practice: write model in mathematical notation and in code
- Embedding images and videos

OVERVIEW OF R MARKDOWN

R Markdown:

- file type that combines code and text
- output is a report or document
- default is text, have to add chunks of code

Why use R Markdown?

- relatively straightforward analyses
- manuscript writing
- lecture materials that involve coding
- need a mix of math and writing

The complete R Markdown guide is here: <https://bookdown.org/yihui/rmarkdown/>

DOCUMENT OUTPUT AND KNITTING

Documents

R Markdown is actually two files: your R Markdown file (.Rmd) and your output file (.pdf, .html, .doc). Upon creating a new R Markdown file, you can choose the output file. Pdf is most common, html good if you'd prefer to view it in a browser, and honestly I wouldn't recommend Word, the formatting gets very weird.

Let's see what those look like...

Your YAML heading up top controls the output document. For example, in a pdf, you can change...

```
output:
pdf_document:
  toc: true (create a table of contents)
  fontsize: 10pt (change the font size, either 10, 11, or 12)
  geometry: margin=1in (change the margins)
  highlight: tango (controls the color scheme of your code chunks)
  number_sections: true (add numbers to outline automatically)
```

Or in an html file, you can change...

```
output:
html_document:
  toc: true (create a table of contents)
  toc_float: true (make it a floating table of contents)
  theme: united (change the appearance)
```

For more information on how to change document elements, see the R Markdown documentation:

<https://bookdown.org/yihui/rmarkdown/documents.html>

Knitting

Hit “Knit” at the top. By default, output file is whatever you selected, but you can knit it to other document types still.

This shows up when you open every R Markdown document. Means by default, code chunks are shown in output, unless specified otherwise.

Your turn!

Create a new R Markdown document called “Data analysis”, where the output file is an html. Leave the default text in the document, but add a floating table of contents.

BASIC SYNTAX

R Markdown syntax is based on Latex syntax. Here are few examples of the most common things you’ll use in document preparation.

Heading 1

Heading 2

Heading 3

When you use these headings, you’re creating an outline. You can quickly jump between sections with the orange hashtag sign below.

`code font`

italics

bold

italics and bold

- solid bullet point
- open bullet point

R Markdown doesn't register tab and spaces, so you have to indent with:

no indent

indent

Insert a page break...

Change text colors

color test

One more note: R Markdown is weird about line breaks and spacing. Add 2 line breaks if you want something in a new line. For example,

same line

different

lines

Your turn!

Delete the default text in your R Markdown document, then create an outline using headers for a project analysis. Include sections for Introduction, Description of Model, and Analysis. Create page breaks in between each section. Then knit it.

ADDING CODE AND MATH CHUNKS

Code chunks

chunk that appears like code, but can't be run

good for code that you want to use as an example but don't want to run

```
# this is regular r code, as if you were in an R script
```

```
x <- 1:5
```

```
y <- 8:10
```

```
z <- c(x,y)
```

```
z
```

```
## [1] 1 2 3 4 5 8 9 10
```

```
# load packages, data, everything as usual in this script
```

```
library(tidyverse)
```

```
## Warning: package 'ggplot2' was built under R version 4.3.1
```

```
## -- Attaching core tidyverse packages ----- tidyverse 2.0.0 --
```

```
## v dplyr      1.1.3      v readr      2.1.4
```

```
## v forcats   1.0.0      v stringr   1.5.0
```

```
## v ggplot2    3.5.0      v tibble    3.2.1
```

```
## v lubridate  1.9.2      v tidyr     1.3.0
```

```
## v purrr      1.0.1
```

```
## -- Conflicts ----- tidyverse_conflicts() --
```

```
## x dplyr::filter() masks stats::filter()
```

```
## x dplyr::lag()     masks stats::lag()
```

```
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

Hit the green arrow to run the chunk. The output will appear in your console AND in a separate window underneath the code chunk. This separate window will appear in your output document, unless you tell it not to.

```
# don't show warnings from packages
```

- `warning = FALSE` : don't show warnings from code chunk

- `message = FALSE` : don't show any messages or warnings
- `eval=FALSE` : show the code chunk but don't run it (same as example above)
- `echo=FALSE` : don't show the code but run the code and show the output (good for figures)
- `include=FALSE` : run the code, but don't show the code or the results from the code (good for packages)

R Markdown chunks can also run python.

```
x = [1, 2, 3]
```

Math

R Markdown is similar to Latex – really useful if you're writing a manuscript with lots of equations. You can insert Latex-style equations with:

equation

You can indicate a variable with *variable*. Mathematical symbols are written with a `&` before the symbol abbreviation. For example, Δ and $\sum_{-\infty}^{\infty} n$. Some other examples of math notation:

$$\frac{1}{x}$$

$$y_i \sim \text{Normal}(\mu, \sigma)$$

Your turn!

In your “Data Analysis” R Markdown document, add the following math in Latex format (this is an example of a generalized linear model you might run if you have count data):

$y \sim \text{Poisson}(\lambda)$

$\log(\lambda) = mx + b$

Then add a code chunk and assign a vector of numbers to an object “x”.

Embedding links and images

Links

In R Markdown, you can embed links with pretty simple syntax:

my GitHub page

Or just include the link like normal:

<https://github.com/annieschiffer>

Images from your computer

In an R Markdown, you can run code to make plots or embed images saved to your computer. The link to the image comes from your files, but it has to be a file path relative to your working directory. To embed images from your files, the syntax is



Figure 1: this is a tree

Images from code

Or you can add an image that you make with code.

```
plot(cars)
```

Some other useful image tricks to add to the `{}` in the R code chunk:

```
fig.height = 3 fig.width = 6 fig.align = "center" fig.asp = "0.6" out.width = 40%
```

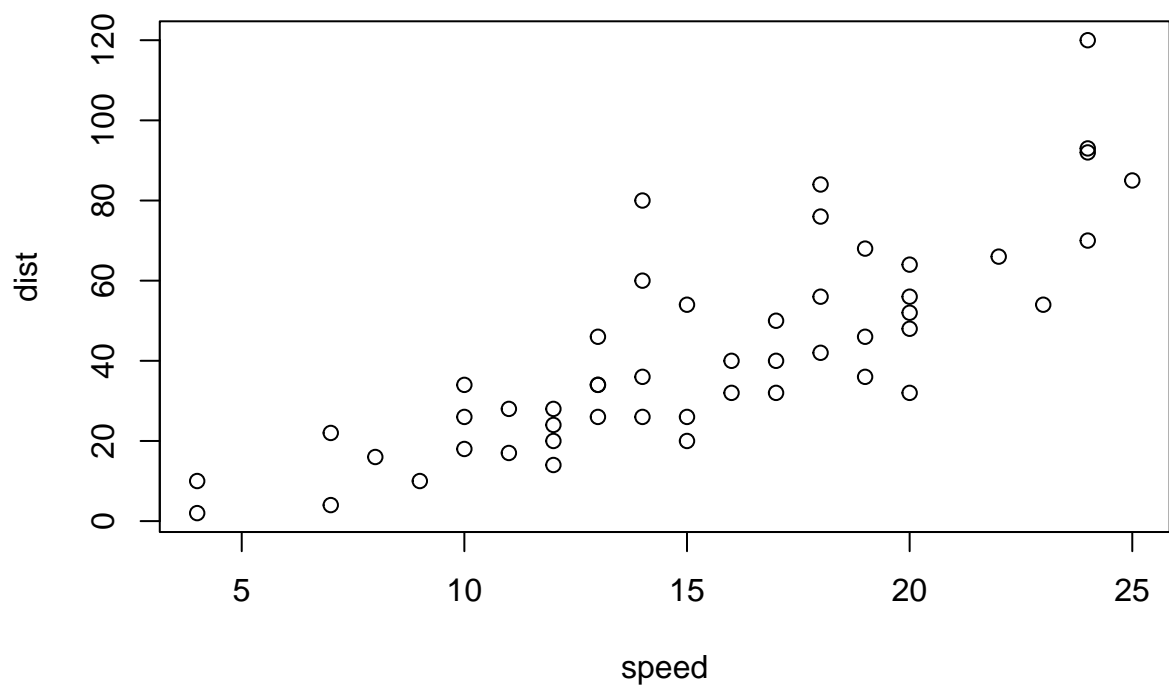


Figure 2: An amazing plot