Introduction to R Markdown

Tong Zhu

27 October 2020

Abstract

In this file, Latex code, R code and R ouput are written, run/complied and presented as a whole automatically. This is known as *R Markdown* file. It is particularly useful for writing reports in data science.

Introduction

There are a large number of jobs that you could do with *R Markdown*:

- Compile a single R Markdown document to a report in different formats, such as PDF or HTML.
- Create notebooks in which you can directly run code chunks interactively.
- Make slides for presentations (HTML5, LaTeX Beamer, or PowerPoint).
- Produce dashboards with flexible, interactive, and attractive layouts.
- Build interactive applications based on Shiny.
- Write journal articles.
- Author books of multiple chapters.
- Generate websites and blogs.

In other words, this is kind of a Latex file. The difference is that it has fewer latex features, but in return, the syntax is easier to handle and you can incorporate R code into it. See the three files that I uploaded on Canvas for more information.

```
render("lab3_intro.Rmd")
```

Methods

Math in R Markdown

Usual syntax in Latex works in *R Markdown*. Inline math is marked off with a pair of dollar signs (`$`), as $\left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y}$. Mathematical displays are marked off with `\[` and `\]`, as in

$$f\left(x;\mu,\sigma^2\right)=\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{\left(x-\mu\right)^2}{2\sigma^2}\right)$$

R code

A *R Markdown* file combines your Latex file with your R script file.

```
my.mean.func = function(x){
  mean(x)
}

my.mean.func(1:10)
```

```
## [1] 5.5
```

R output

By default R code will be run. But we can adjust things to be included in the pdf file.

Notice a, b, c and d are all generated in R.

a: Both the statement and the output are shown in the pdf file.

```
a = rnorm(10)
a
```

```
##  [1] -0.08447968 -0.50981290  0.62284435  0.53771283  2.33619076 -0.25348589
##  [7] -0.39308529  0.47333874  0.59453243 -0.34320480
```

b: Only the output is shown in the pdf file, but not the statement

```
##  [1]  1.7066669 -0.7230827 -0.2020681 -1.1378950  0.5883626 -1.9767772
##  [7]  1.0552056 -0.3607013 -0.6629768 -0.8951095
```

c: Only the statement is shown in the pdf file, but not output.

```
c = rnorm(10)
c
```

d: Neither the statement nor the output is shown in the pdf file.

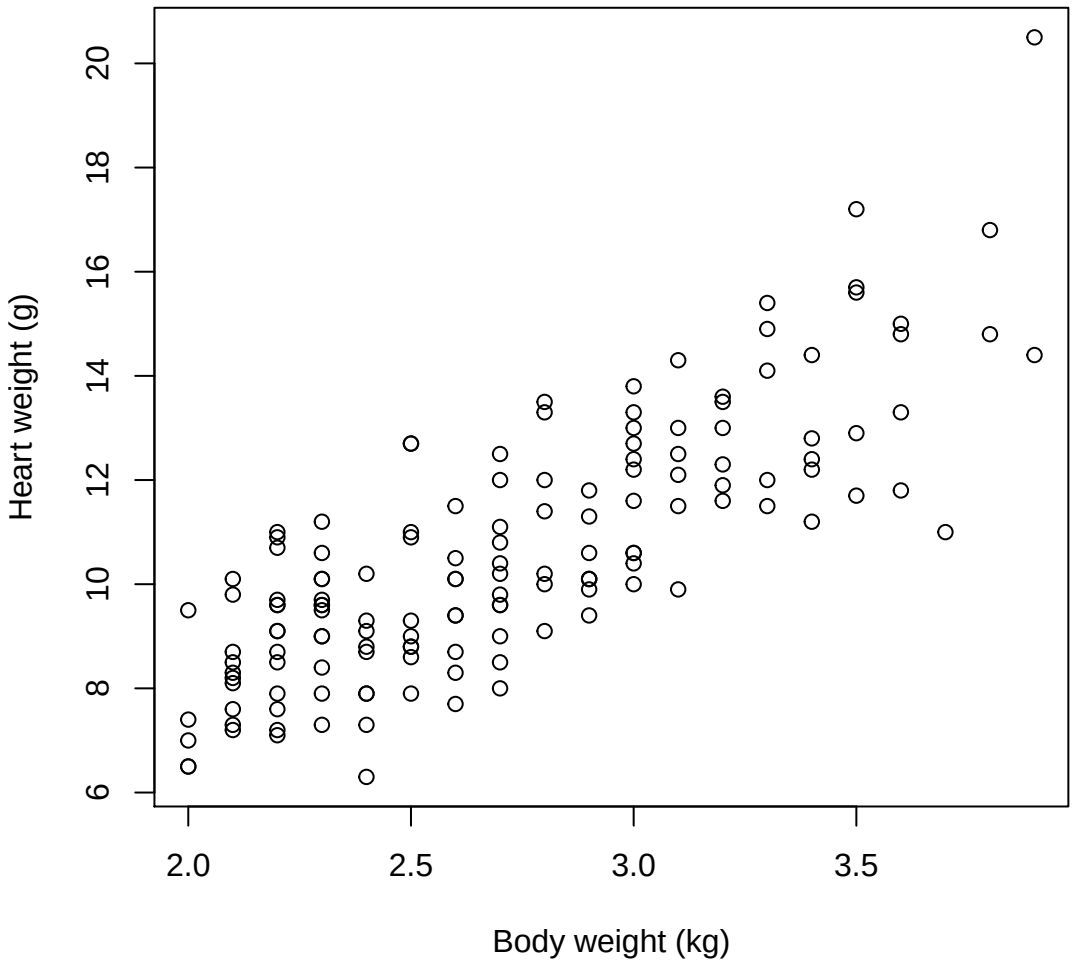
Inline R Code

Of course, sometime you need a number from your data analysis. This can be easily obtained and updated in *R Markdown* file, e.g. There are 10 elements in a, and the mean of b is -0.2608375.

Plot

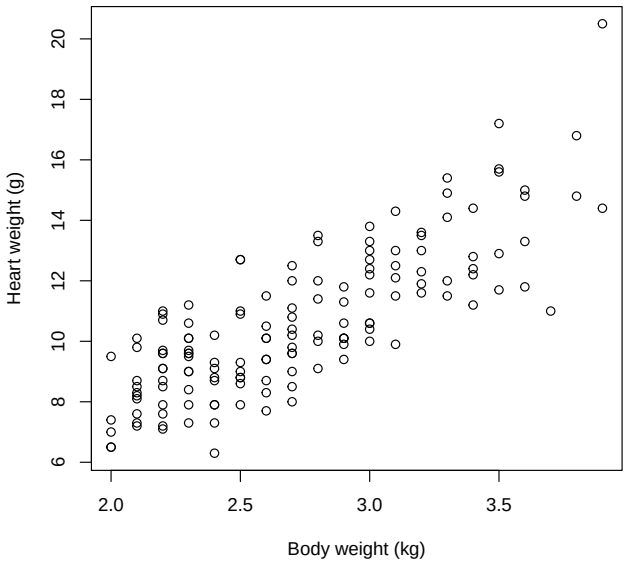
One of the best features of R markdown is graphs are automatically generated and included in the pdf file.

```
library(MASS)
data(cats)
plot(Hwt ~ Bwt, data = cats, xlab = "Body weight (kg)", ylab = "Heart weight (g)")
```



There are a bunch of options for adjusting the placement of the plot which R produces.

```
plot(Hwt ~ Bwt, data = cats, xlab = "Body weight (kg)", ylab = "Heart weight (g)")
```



- The option `fig.align` controls the horizontal **alignment** (left, right, or center).

- The options `out.height` and `out.width` let you specify the desired height or width of the figure.

Tables

The default summary output in R is too ugly to be included in a formal report. Compare the followings:

```
coefficients(summary(lm(Hwt ~ Bwt, data=cats)))
```

```
##               Estimate Std. Error    t value     Pr(>|t|)
## (Intercept) -0.3566624  0.6922770 -0.5152019 6.072131e-01
## Bwt          4.0340627  0.2502615 16.1193908 6.969045e-34
```

```
library(knitr) # Only need this the first time!
kable(coefficients(summary(lm(Hwt ~ Bwt, data=cats))))
```

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	-0.3566624	0.6922770	-0.5152019	0.6072131
Bwt	4.0340627	0.2502615	16.1193908	0.0000000

Computationally intense task

By default, R Markdown will re-run all of your code every time you render/knit your document. If some of your code is slow, this can add up to a lot of time. You can, however, ask R Markdown to keep track of whether a chunk of code has changed, and only re-run it if it has. This is called **caching** the chunk.

```
data.sim.vec = rnorm(1e6)
x.vec = sort(data.sim.vec)
y = max(x.vec)
```

Results, conclusions and recommendations

In addition to executing R code chunks, it can also execute chunks in a variety of other languages that are widely used in data science. Some of the available language engines include:

- Python
- SQL
- Bash
- Rcpp
- Stan
- JavaScript
- CSS

See the book "*R Markdown: The Definitive Guide*" if you are interested.

Conclusion

Use it!