Writing a reproducible paper in R Markdown[*]

Paul Bauer, Mannheim Centre for European Social Research
Camille Landesvatter, Mannheim Centre for European Social Research

First version: December 14, 2018
This version: April 05, 2023
Download: https://osf.io/q395s

Feedback? Errors? mail@paulcbauer.de
Github repository

Tired of Latex?
Try Pagedown with our new template! :-)

Abstract

The present paper provides a template for a reproducible scientific paper written in R Markdown. Below we outline some of the "tricks"/code (e.g., referencing tables, sections etc.) we had to figure out to produce this document. The underlying files which produce this document can be downloaded here (click on Code -> Download ZIP). We think we got pretty far but there is always room for improvement and more automatization, in parallel to the incredible developments in R and Rstudio (bookdown etc.). We intend to update this file when we discover more convenient code (you can follow any updates through the corresponding Github repo).

[*] Corresponding address: mail@paulcbauer.de. Acknowledgments: We are grateful to all those generous people that invest their time into open-source software.

Contents

1 Why reproducible research (in R)?

Some arguments…

- **Access**: Research is normally funded by taxpayers (researchers are also taxpayers). Hence, it should be freely accessible to everyone without any barriers, e.g., without requiring commercial software. Importantly, researchers from developing countries are even more dependent on free access to knowledge (Kirsop and Chan 2005).
- **Reproducability**: Even if you have written a study and analyzed the data yourself you will forget what you did after a few months. A fully reproducible setup will help you to trace back your own steps. Obviously, the same is true for other researchers who may want to understand your work and built on it. It may sound like a joke but why not aim for a document that can be used to reproduce your findings in 500 years.
- **Errors**: Manual steps in data analysis (e.g., manually copy/pasting values into a table etc.) may introduce errors. R Markdown allows you to **automatize** such steps and/or avoid them.
- **Revisions**: Revising a paper takes much less time if you have all the code you need in one place, i.e., one `.rmd` file. For instance, if you decide to exclude a subset of your data you simply need to insert one line of your code at the beginning and everything is rebuilt/re-estimated automatically.

2 Prerequesites

We assume that you are using R on a day-to-day basis. You may have even started to work a little in R Markdown but you don't write your complete paper in R Markdown. If you don't know what R Markdown is watch this short video. Then…

- …install R and Rstudio (most recent versions) (R Core Team 2017; RStudio Team 2015).
- …install tinytex, a lightweight version of Tex Live (Allaire et al. 2017; Xie 2018b).

```
install.packages(c('tinytex', 'rmarkdown'))
tinytex::install_tinytex()
```

- …install the packages below using the code below (Sievert et al. 2017; Xie 2014, 2015, 2016, 2017, 2018a; Zhu 2017).

```
install.packages(c("rmarkdown", "knitr", "kableExtra",
                   "stargazer", "plotly", "knitr",
                   "bookdown"))
```

- …download the 5 input files we created — `paper.rmd`, `references.bib`, `data.csv` and `american-sociological-association.csl` — from Github (click on Code -> Download ZIP). Ignore the other files.

- …learn R and read about the other underlying components namely Markdown, R Markdown and Latex.

3 Basics: Input and output files

All the files you need to produce the present PDF file are the input files…

- …a `paper.rmd` file (the underlying R Markdown file).
- …a `references.bib` file (the bibliography).
  - We use paperpile to manage my references and export the `.bib` file into the folder that contains my `.rmd` file.
- …a `data.csv` file (some raw data).
- … a `american-sociological-association.csl` file that defines the style of your bibliography.[1]

Download these files from Github (click on Code -> Download ZIP) and save them into a folder. Close R/Rstudio and directly open `paper.rmd` with RStudio. Doing so assures that the working directory is set to the folder that contains `paper.rmd` and the other files.[2]

Once you run/compile the `paper.rmd` file in Rstudio it creates mainly two output files:

- `paper.tex`
- `paper.pdf` (the one you are reading right now)

In addition, there may be files that you generate and store locally in the folder during the compilation process. This is the case for some of the Plotly graphs below.

[1] You can download various citation style files from this webpage: https://github.com/citation-style-language/styles.

[2] You can always check your working directory in R with `getwd()`.

Ideally, we can simply provide others with a zip folder that contains both our input files and our output files. Then it's possible to reproduce the process from managing/analyzing some raw data to producing the final scientific article.

Below we always display the R code in the chunks that produce the output. In your paper you will normally only present outputs (e.g., tables, figures etc.) by choosing the chunk option "Show output only" in R Studio. The chunk commands itself are not displayed but they do matter for referencing etc. So simply orient yourself at the underlying paper.rmd file.

4 Referencing within your document

To see how referencing works simply see the different examples for figures, tables and sections below. For instance in Section 7 you can find different ways of referencing tables. The code of the underlying paper.rmd will show you how we referenced Section 7 right here namely with 'Section \@ref(sec:tables)'.

5 Software versioning

Software changes and gets updated, especially with an active developer community like that of R. Luckily you can always access old versions of R and old version of R packages in the archive. In the archive you need to choose a particular package, e.g dplyr and search for the right version, e.g., dplyr_0.2.tar.gz. Then insert the path in the following function: install.packages("https://....../dplyr_0.2.tar.gz", repos=NULL, type="source"). Ideally, however, results will be simply reproducible in the most current R and package versions.

We would recommend to use the command below and simply add it to the appendix as we did here in Appendix A.1. This will make sure you always provide the package versions that you used in the last compilation of your paper. For more advanced tools see packrat.

```
cat(paste("#", capture.output(sessionInfo()), "\n", collapse =""))
  # or use message() instead of cat()
```

6 Data

6.1 Import

```
data <- read.csv("data.csv")
head(data)
```

```
##   X speed dist
## 1 1     4    2
## 2 2     4   10
## 3 3     7    4
## 4 4     7   22
## 5 5     8   16
## 6 6     9   10
```

6.2 Putting your entire data into the .rmd file

Applying the function `dput()` to an object gives you the code needed to reproduce that object. So you could paste that code into your `.rmd` file if you don't want to have extra data files. This makes sense were data files are small.

```
dput(data)
```

```
## structure(list(X = 1:50, speed = c(4L, 4L, 7L, 7L, 8L, 9L, 10L,
## 10L, 10L, 11L, 11L, 12L, 12L, 12L, 12L, 13L, 13L, 13L, 13L, 14L,
## 14L, 14L, 14L, 15L, 15L, 15L, 16L, 16L, 17L, 17L, 17L, 18L, 18L,
## 18L, 18L, 19L, 19L, 19L, 20L, 20L, 20L, 20L, 20L, 22L, 23L, 24L,
## 24L, 24L, 24L, 25L), dist = c(2L, 10L, 4L, 22L, 16L, 10L, 18L,
## 26L, 34L, 17L, 28L, 14L, 20L, 24L, 28L, 26L, 34L, 34L, 46L, 26L,
## 36L, 60L, 80L, 20L, 26L, 54L, 32L, 40L, 32L, 40L, 50L, 42L, 56L,
## 76L, 84L, 36L, 46L, 68L, 32L, 48L, 52L, 56L, 64L, 66L, 54L, 70L,
## 92L, 93L, 120L, 85L)), class = "data.frame", row.names = c(NA,
## -50L))
```

You can then insert the dput output in your `.rmd` as below.

```
data <- structure(list(X = 1:50, speed = c(4L, 4L, 7L, 7L, 8L, 9L, 10L,
10L, 10L, 11L, 11L, 12L, 12L, 12L, 12L, 13L, 13L, 13L, 13L, 14L,
14L, 14L, 14L, 15L, 15L, 15L, 16L, 16L, 17L, 17L, 17L, 18L, 18L,
18L, 18L, 19L, 19L, 19L, 20L, 20L, 20L, 20L, 20L, 22L, 23L, 24L,
24L, 24L, 24L, 25L), dist = c(2L, 10L, 4L, 22L, 16L, 10L, 18L,
26L, 34L, 17L, 28L, 14L, 20L, 24L, 28L, 26L, 34L, 34L, 46L, 26L,
36L, 60L, 80L, 20L, 26L, 54L, 32L, 40L, 32L, 40L, 50L, 42L, 56L,
76L, 84L, 36L, 46L, 68L, 32L, 48L, 52L, 56L, 64L, 66L, 54L, 70L,
92L, 93L, 120L, 85L)),
class = "data.frame", row.names = c(NA,
-50L))
```

7 Tables

Producing good tables and referencing these tables within a R Markdown PDF has been a hassle but got much better. Examples that you may use are shown below. The way you reference tables is slightly different, e.g., for `stargazer` the label is contained in the function, for `kable` it's contained in the chunk name.

7.1 stargazer(): Summary and regression tables

Table 1 shows summary stats of your data.[3] We normally use `stargazer()` (Hlavac 2013) which offers extreme flexibility regarding table output (see `?stargazer`).

```
library(stargazer)
stargazer(cars,
          title = "Summary table with stargazer",
          label="tab1",
          table.placement = "H",
          header=FALSE)
```

Table 1: Summary table with stargazer

Statistic	N	Mean	St. Dev.	Min	Max
speed	50	15.400	5.288	4	25
dist	50	42.980	25.769	2	120

[3] To reference the table where you set the identifier in the stargazer function you only need to use the actual label, i.e., ´tab1´.

Table 2 shows the output for a regression table. Make sure you name all your models and explicitly refer to model names (M1, M2 etc.) in the text.

```
library(stargazer)
model1 <- lm(speed ~ dist, data = cars)
model2 <- lm(speed ~ dist, data = cars)
model3 <- lm(dist ~ speed, data = cars)
stargazer(model1, model2, model3,
          title = "Regression table with stargazer",
          label="tab2",
          table.placement = "H",
          column.labels = c("M1", "M2", "M3"),
          model.numbers = FALSE,
          header=FALSE)
```

Table 2: Regression table with stargazer

	Dependent variable:		
	speed		dist
	M1	M2	M3
dist	0.166^{***}	0.166^{***}	
	(0.017)	(0.017)	
speed			3.932^{***}
			(0.416)
Constant	8.284^{***}	8.284^{***}	-17.579^{**}
	(0.874)	(0.874)	(6.758)
Observations	50	50	50
R^2	0.651	0.651	0.651
Adjusted R^2	0.644	0.644	0.644
Residual Std. Error (df = 48)	3.156	3.156	15.380
F Statistic (df = 1; 48)	89.567^{***}	89.567^{***}	89.567^{***}
Note:			*p<0.1; **p<0.05; ***p<0.01

7.2 kable() and kable_styling()

Another great function is `kable()` (`knitr` package) in combination with `kableExtra`. Table 3 provides an example.[4] Again you can modify so many things in both the `kable()` and the `kable_styling()` function. See this overview of all the kable stylings that are possible provided by the package author himself.

Table 3: Table with kable() and kablestyling()

	speed	dist
1	4	2
2	4	10
3	7	4
4	7	22
5	8	16
6	9	10
7	10	18
8	10	26
9	10	34
10	11	17

8 Inline code & results

Reproduction reaches new heights when you work with inline code. For instance, you can automatize the display of certain coefficients within the text. An example is to include estimates, e.g., the coefficient of `dist` of the model we ran above. `` `r round(coef(model1)[2], 2)` `` will insert the coefficient as follows: 0.17. Or `` `r 3 + 7` `` will insert a 10 in the text.
Inline code/results that depend on earlier objects in your document will automatically be updated once you change those objects. For instance, imagine a reviewer asks you to omit certain observations from your sample. You can simply do so in the beginning of your code and push play subsequently.. at time you might have to set `cache = FALSE` at the beginning so that all the code chunks are rerun.
Researchers often avoid referring to results in-text etc. because you easily forget to change them when revising a manuscript. At the same it can make an article much more informative and easier to read, e.g., if you discuss a coefficient in the text you can directly show it in the section in which you discuss it. Inline code allows you to do just that. R Markdown allows you to that do so in a reproducible and automatized manner.

[4] To reference the table produced by the chunk you need to add ´tab:´ to the chunk name, i.e., ´tab:tab3´.

9 Figures

9.1 R base graphs

Inserting figures can be slightly more complicated. Ideally, we would produce and insert them directly in the `.rmd` file. It's relatively simple to insert R base graphs as you can see in Figure 1.

```
plot(cars$speed, cars$dist)
```

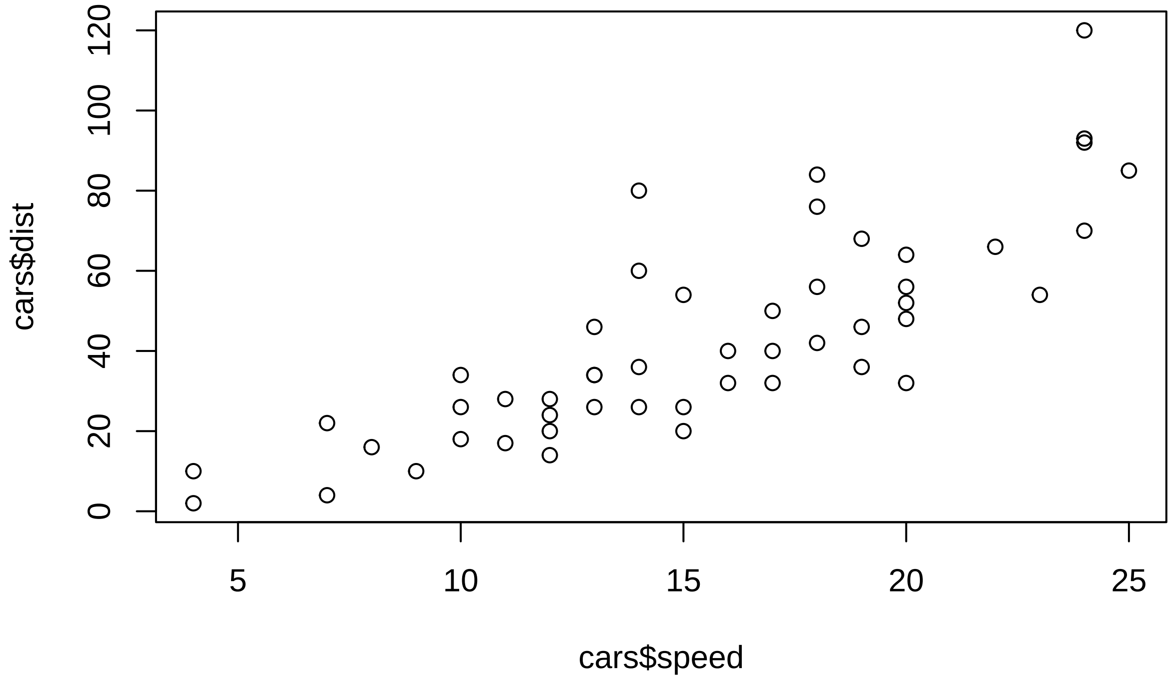

Figure 1: Scatterplot of Speed and Distance

But it turns out that it doesn't always work so well.

9.2 ggplot2 graphs

Same is true for ggplot2 as you can see in Figure 2.

```
mtcars$cyl <- as.factor(mtcars$cyl) # Convert cyl to factor
library(ggplot2)
ggplot(mtcars, aes(x=wt, y=mpg, shape=cyl)) + geom_point() +
  labs(x="Weight (lb/1000)", y = "Miles/(US) gallon",
       shape="Number of \n Cylinders") + theme_classic()
```

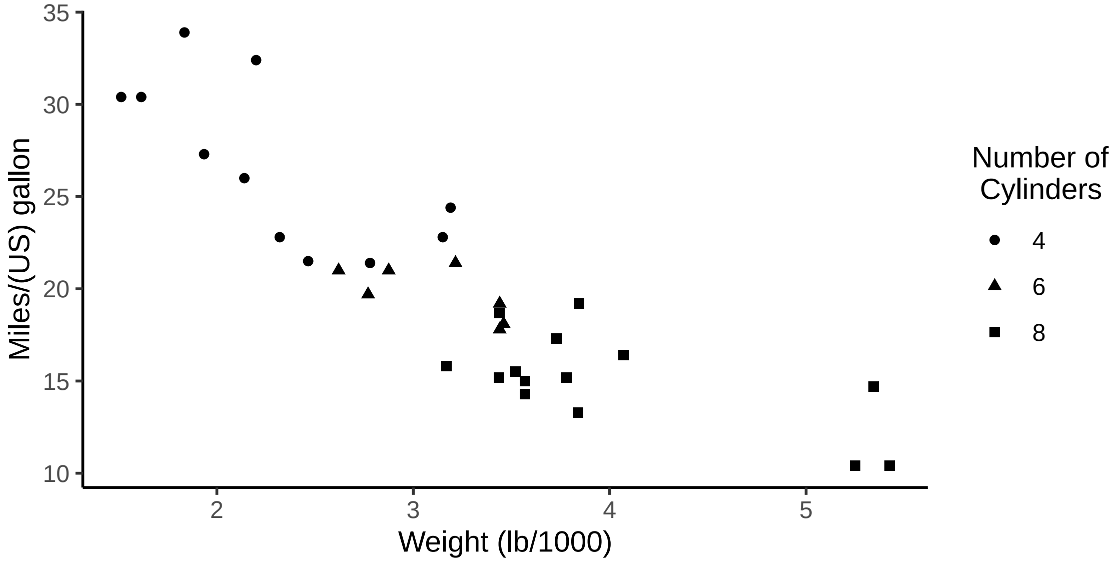

Figure 2: Miles per gallon according to the weight

10 Good practices

Every researcher has his own optimized setup. Currently we would recommend the following:

- Keep all files of your project (that matter for producing the PDF) in one folder without subfolders. You can zip and directly upload that folder to the Harvard dataverse.
- Make sure that filenames have a logic to them.
  - Main file with text/code: “paper.rmd”, “report.rmd”
  - Data files: “data_xxxxxx.*”
  - Image files: “fig_xxxxxx.*”
  - Tables files: “table_xxxx.*”
  - etc.
  - Ideally, your filenames will correspond to the names in the paper. For instance, Figure 1 in the paper may have a corresponding file called `fig_1_xxxxx.pdf`.
- Use the document outline in R studio (Ctrl + Shift + O) when you work with R Markdown.

- Name rchunks according to what they do or produce:
  - "`fig-...`" for chunks producing figures
  - "`table-...`" for chunks producing tables
  - "`model-...`" for chunks producing model estimates
  - "`import-...`" for chunks importing data
  - "`recoding-...`" for chunks in which data is recoded
- Ideally give chunks the labels of the figures/tables in the final publication, e.g., "`fig-1`" is the name of the chunk that produces Figure 1 in the final paper.
- Use "really" informative variable names:
  - Q: What do you think does the variable *trstep* measure? It actually measures trust in the European parliament.
    - How could we call this variable instead? Yes, `trust.european.parliament` which is longer but will probably be understood by another researcher.
  - If your setup is truly reproducible you will probably re-use the variable names that you generate as variable names in the tables you produce. Hence, there is an incentive to use good names.
- Use unique identifiers in the final document:
  - e.g., name the models you estimate "M1", "M2" etc.
  - These unique names should also appear in the published paper.
  - Think of someone who wants to produce Figure 1/Model 1 in your paper but doesn't find it in your code...

11 Additional tricks for publishing

- Make your script anonymous
  - Simply put a `<!-- ... -->` around any identifying information, e.g., author names, title footnote etc.
- Counting words
  - Use adobe acrobat (commerical software) to convert your file to a word file. Then open in word and delete all the parts that shouldn't go into the word count. The word count is displayed in the lower right.
  - Use an one of the online services to count your words (search for "pdf word count")
- Appendix: You can change the numbering format for the appendix in the rmd file
  - What is still not possible in this document is to automatically have separate reference sections for paper and appendix.

- Journals may require you to use their tex style: Sometimes you can simply use their template in your rmarkdown file. See here for a PLOS one example.

12 Citation styles

If your study needs to follow a particular citation style, you can set the corresponding style in the header of your `.rmd` document. To do so you have to download the corresponding `.csl` file.

In the present document we use the style of the American Sociological Association and set it in the preamble with `csl: american-sociological-association.csl`. However, you also need to download the respective `.csl` file from the following Github page: https://github.com/citation-style-language/styles and copy it into your working directory for it to work.

The Github directory contains a wide variety of citation style files depending on what discipline you work in.

13 References

Allaire, JJ, Jeffrey Horner, Vicent Marti, and Natacha Porte. 2017. *Markdown: 'Markdown' Rendering for r.*

Hlavac, Marek. 2013. "Stargazer: LaTeX Code and ASCII Text for Well-Formatted Regression and Summary Statistics Tables." *URL: Http://CRAN. R-Project. Org/Package= Stargazer.*

Kirsop, Barbara and Leslie Chan. 2005. "Transforming Access to Research Literature for Developing Countries." *Serials Review* 31(4):246–55.

R Core Team. 2017. *R: A Language and Environment for Statistical Computing.* Vienna, Austria: R Foundation for Statistical Computing.

RStudio Team. 2015. *RStudio: Integrated Development Environment for r.* Boston, MA: RStudio, Inc.

Sievert, Carson, Chris Parmer, Toby Hocking, Scott Chamberlain, Karthik Ram, Marianne Corvellec, and Pedro Despouy. 2017. *Plotly: Create Interactive Web Graphics via 'Plotly.js'.*

Xie, Yihui. 2014. "Knitr: A Comprehensive Tool for Reproducible Research in R." in *Implementing reproducible computational research*, edited by V. Stodden, F. Leisch, and R. D. Peng. Chapman; Hall/CRC.

Xie, Yihui. 2015. *Dynamic Documents with R and Knitr.* 2nd ed. Boca Raton, Florida: Chapman; Hall/CRC.

Xie, Yihui. 2016. *Bookdown: Authoring Books and Technical Documents with R Markdown*. Boca Raton, Florida: Chapman; Hall/CRC.

Xie, Yihui. 2017. *Bookdown: Authoring Books and Technical Documents with r Markdown*.

Xie, Yihui. 2018a. *Knitr: A General-Purpose Package for Dynamic Report Generation in r*.

Xie, Yihui. 2018b. *Tinytex: Helper Functions to Install and Maintain 'TeX Live', and Compile 'LaTeX' Documents*.

Zhu, Hao. 2017. *kableExtra: Construct Complex Table with 'Kable' and Pipe Syntax*.

Online Appendix

A Online appendix

A.1 Attach R session info in appendix

Since R and R packages are constantly evolving you might want to add the R session info that contains information on the R version as well as the packages that are loaded.

```
## R version 4.2.2 (2022-10-31)
## Platform: aarch64-apple-darwin20 (64-bit)
## Running under: macOS Ventura 13.0.1
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.2-arm64/Resources/lib/libRblas.0.d
## LAPACK: /Library/Frameworks/R.framework/Versions/4.2-arm64/Resources/lib/libRlapack.d
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] ggplot2_3.4.2    kableExtra_1.3.4 knitr_1.42       stargazer_5.2.3
##
## loaded via a namespace (and not attached):
##  [1] compiler_4.2.2    pillar_1.9.0      tools_4.2.2       digest_0.6.31
##  [5] evaluate_0.20     lifecycle_1.0.3   tibble_3.2.1      gtable_0.3.3
##  [9] viridisLite_0.4.1 pkgconfig_2.0.3   rlang_1.1.0       cli_3.6.1
## [13] rstudioapi_0.14   yaml_2.3.7        xfun_0.38         fastmap_1.1.1
## [17] withr_2.5.0       dplyr_1.1.1       httr_1.4.5        stringr_1.5.0
## [21] xml2_1.3.3        generics_0.1.3    vctrs_0.6.1       systemfonts_1.0.4
## [25] tidyselect_1.2.0  webshot_0.5.4     grid_4.2.2        svglite_2.1.1
## [29] glue_1.6.2        R6_2.5.1          fansi_1.0.4       rmarkdown_2.21
## [33] bookdown_0.33     farver_2.1.1      magrittr_2.0.3    scales_1.2.1
## [37] htmltools_0.5.5   rvest_1.0.3       colorspace_2.1-0  labeling_0.4.2
## [41] utf8_1.2.3        stringi_1.7.12    munsell_0.5.0
```

A.2 All the code in the paper

To simply attach all the code you used in the PDF file in the appendix see the R chunk in the underlying `.rmd` file:

```
knitr::opts_chunk$set(cache = FALSE)
# Use chache = TRUE if you want to speed up compilation

# A function to allow for showing some of the inline code
```

~~~r
rinline <- function(code){
  html <- '<code  class="r">
``` `r CODE` ```</code>'
  sub("CODE", code, html)
}
install.packages(c('tinytex', 'rmarkdown'))
tinytex::install_tinytex()
install.packages(c("rmarkdown", "knitr", "kableExtra",
                   "stargazer", "plotly", "knitr",
                   "bookdown"))
cat(paste("#", capture.output(sessionInfo()), "\n", collapse =""))
  # or use message() instead of cat()
data <- read.csv("data.csv")
head(data)
dput(data)
data <- structure(list(X = 1:50, speed = c(4L, 4L, 7L, 7L, 8L, 9L, 10L,
10L, 10L, 11L, 11L, 12L, 12L, 12L, 12L, 13L, 13L, 13L, 13L, 14L,
14L, 14L, 14L, 15L, 15L, 15L, 16L, 16L, 17L, 17L, 17L, 18L, 18L,
18L, 18L, 19L, 19L, 19L, 20L, 20L, 20L, 20L, 20L, 22L, 23L, 24L,
24L, 24L, 24L, 25L), dist = c(2L, 10L, 4L, 22L, 16L, 10L, 18L,
26L, 34L, 17L, 28L, 14L, 20L, 24L, 28L, 26L, 34L, 34L, 46L, 26L,
36L, 60L, 80L, 20L, 26L, 54L, 32L, 40L, 32L, 40L, 50L, 42L, 56L,
76L, 84L, 36L, 46L, 68L, 32L, 48L, 52L, 56L, 64L, 66L, 54L, 70L,
92L, 93L, 120L, 85L)),
class = "data.frame", row.names = c(NA,
-50L))
library(stargazer)
stargazer(cars,
          title = "Summary table with stargazer",
          label="tab1",
          table.placement = "H",
          header=FALSE)
library(stargazer)
model1 <- lm(speed ~ dist, data = cars)
model2 <- lm(speed ~ dist, data = cars)
model3 <- lm(dist ~ speed, data = cars)
stargazer(model1, model2, model3,
          title = "Regression table with stargazer",
          label="tab2",
          table.placement = "H",
          column.labels = c("M1", "M2", "M3"),
          model.numbers = FALSE,
          header=FALSE)
library(knitr)
library(kableExtra)
~~~

```
kable(cars[1:10,], row.names = TRUE,
      caption = 'Table with kable() and kablestyling()',
      format = "latex", booktabs = T) %>%
        kable_styling(full_width = T,
                      latex_options = c("striped",
                                        "scale_down",
                                        "HOLD_position"),
                      font_size = 10)
plot(cars$speed, cars$dist)
mtcars$cyl <- as.factor(mtcars$cyl) # Convert cyl to factor
library(ggplot2)
ggplot(mtcars, aes(x=wt, y=mpg, shape=cyl)) + geom_point() +
  labs(x="Weight (lb/1000)", y = "Miles/(US) gallon",
       shape="Number of \n Cylinders") + theme_classic()
print(sessionInfo(), local = FALSE)
```