Session 7: Introduction to R Markdown

R for Stata Users

DIME Analytics
The World Bank – DIME | WB Github
March 2024

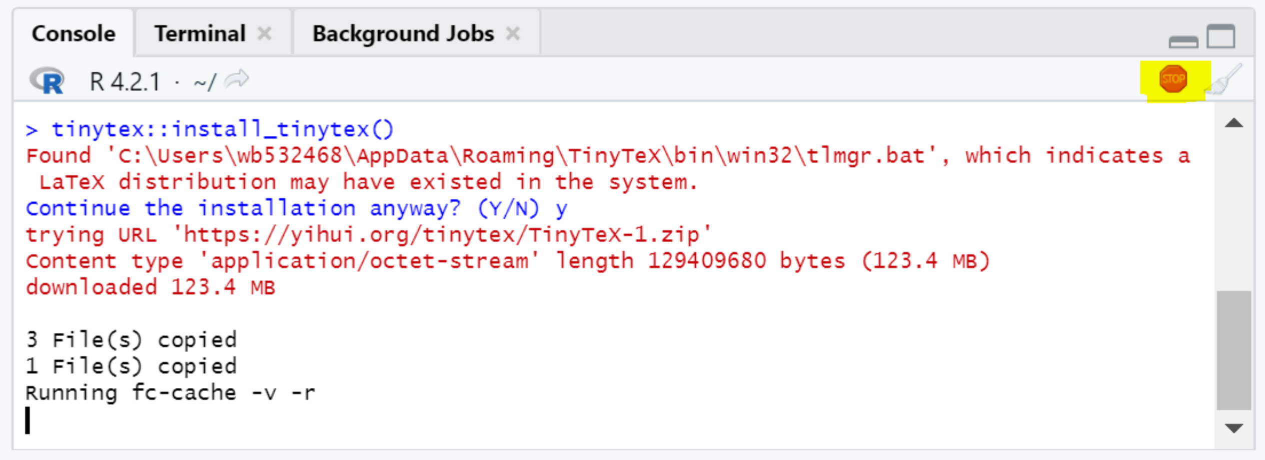
Preamble

- Make sure you have the packages `tinytex`, `stargazer`, and `huxtable` installed

```r
# Packages we used for other sessions, install only if needed
install.packages("dplyr")
install.packages("huxtable")

# New packages
install.packages("tinytex")
install.packages("stargazer")

# No need to load the packages for now
```

Preamble (🕓 5 min)

- Use `tinytex` to install LaTeX with: `tinytex::install_tinytex()`
- This will take a while. Leave it running:



```
> tinytex::install_tinytex()
Found 'C:\Users\wb532468\AppData\Roaming\TinyTeX\bin\win32\tlmgr.bat', which indicates a
 LaTeX distribution may have existed in the system.
Continue the installation anyway? (Y/N) y
trying URL 'https://yihui.org/tinytex/TinyTeX-1.zip'
Content type 'application/octet-stream' length 129409680 bytes (123.4 MB)
downloaded 123.4 MB

3 File(s) copied
1 File(s) copied
Running fc-cache -v -r
```

- LaTeX can be unpredictable in WB computers. It's possible that this didn't work
- Don't worry for now, just follow the approppriate instructions we'll specify in the exercises

Introduction

- This is an **introduction** to R Markdown
- We'll show:
  1. How to write and knit (output) R Markdown documents
  2. How to format text and R code in R Markdown documents
  3. How to include regression tables in R Markdown documents

Table of contents

1. Dynamic documents
2. Knitting
3. Markdown
4. R code
5. R plots
6. Inline code
7. Regression outputs
8. Annex

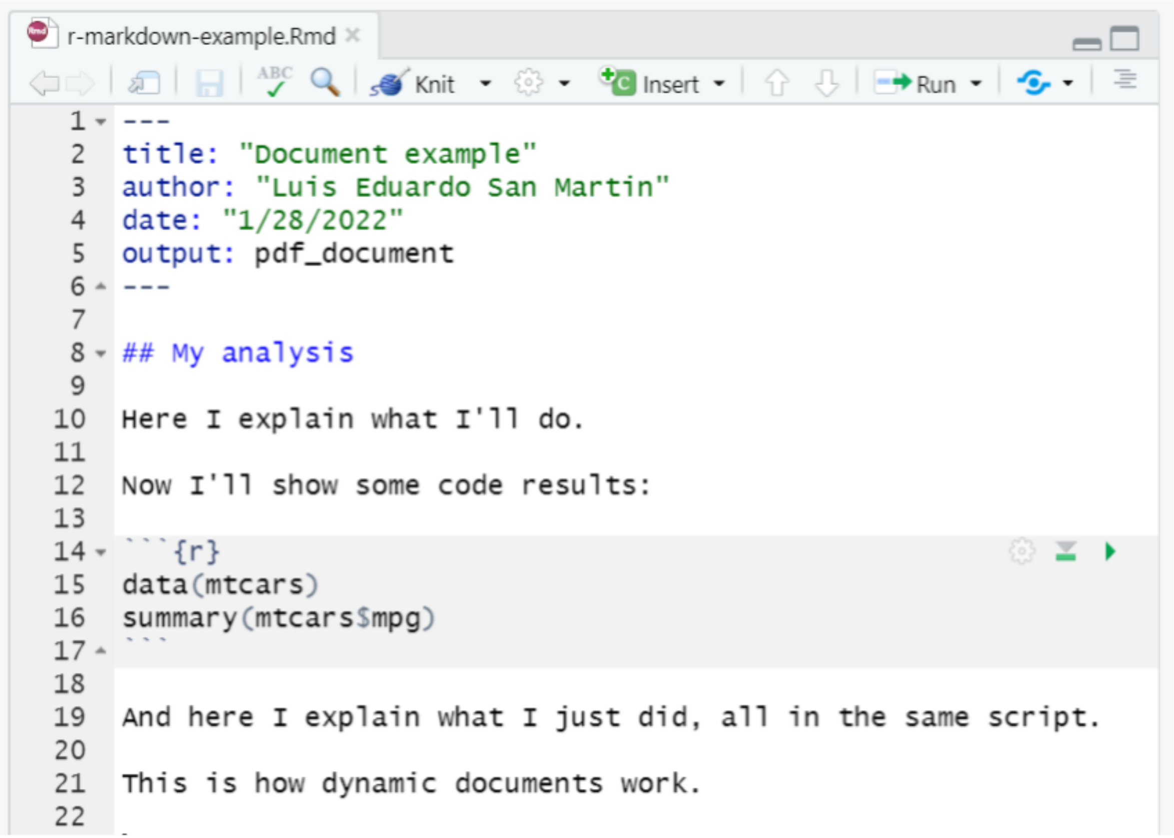
Dynamic documents

Dynamic documents and R Markdown

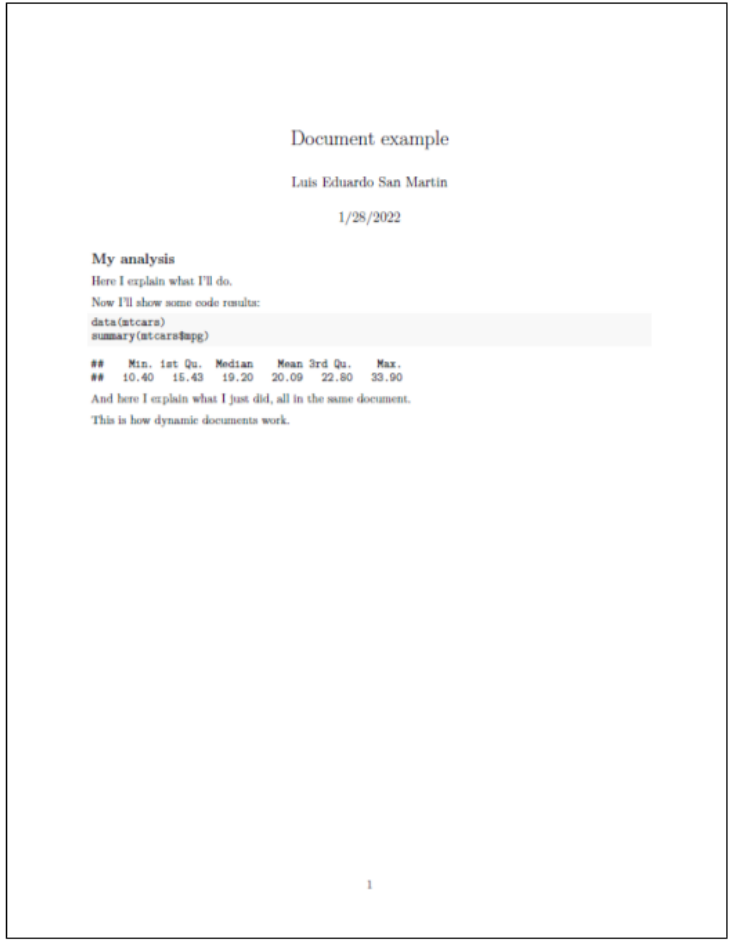
- Dynamic documents are documents that include both text and code outputs
- They are generated by a script and are updated automatically every time the script runs
- R Markdown is a type of dynamic document

Dynamic documents

- Code and documentation is produced together



~~~
---
title: "Document example"
author: "Luis Eduardo San Martin"
date: "1/28/2022"
output: pdf_document
---

## My analysis

Here I explain what I'll do.

Now I'll show some code results:

```{r}
data(mtcars)
summary(mtcars$mpg)
```

And here I explain what I just did, all in the same script.

This is how dynamic documents work.
~~~



Document example

Luis Eduardo San Martin

1/28/2022

My analysis

Here I explain what I'll do.

Now I'll show some code results:

```
data(mtcars)
summary(mtcars$mpg)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   10.40   15.43   19.20   20.09   22.80   33.90
```

And here I explain what I just did, all in the same document.

This is how dynamic documents work.

1

Why use dynamic documents?

- Increased research transparency. Documents are fully reproducible
- No more copying and pasting outputs from R to a document editor
- Nice option for simple documents that don't require a lot of formatting
- Can include code snippets

Knitting R Markdown documents

Knitting R Markdown documents

- R markdown combines text, R code, and rendered outputs
- The text follows Markdown's syntax
- The code and outputs follow R's syntax
- Knitting an R Markdown document is rendering the text and code portions into a single output
- The output can be a PDF, Word, HTML document, or others

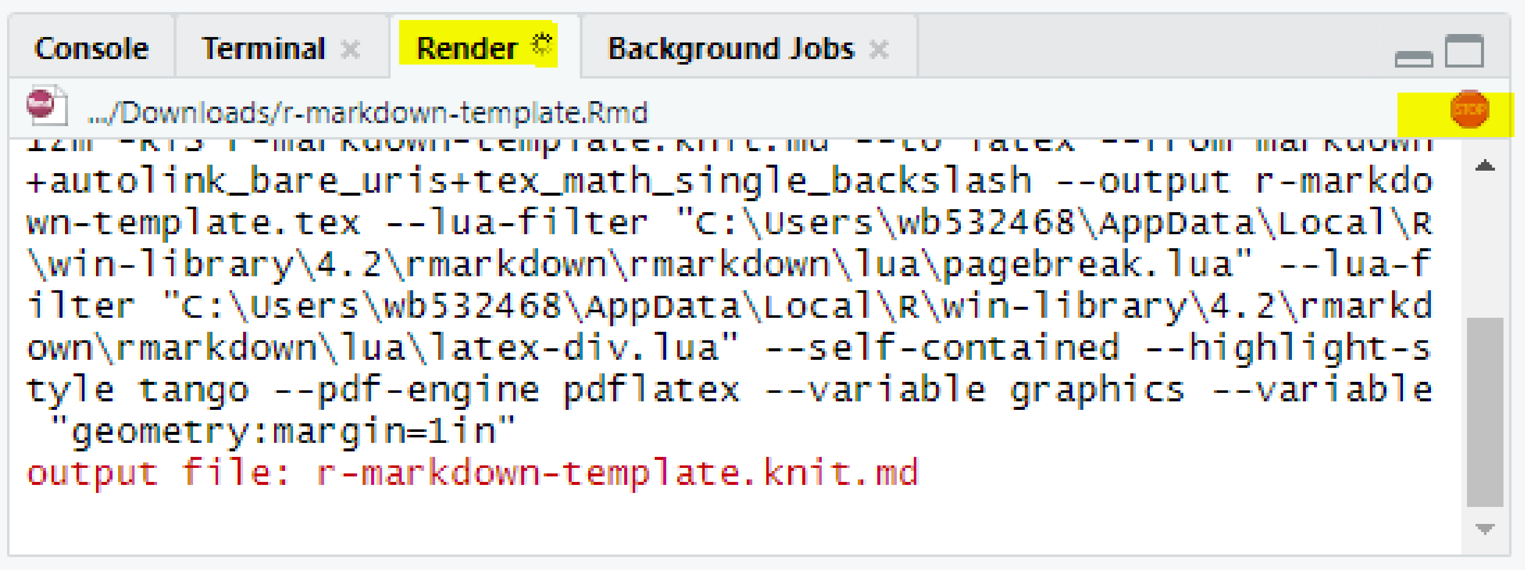
Knitting R Markdown documents

Exercise 1: Knit an R Markdown document (🕓 2 min, leave it running)

1. Download the file `r-markdown-template.Rmd` from : https://osf.io/7g6t9/
2. Open this file in RStudio
   - If the installation of tinytex didn't work, change line 2 to: `output: html_document`
3. Click on `Knit`. If RStudio asks you to update some packages, select `Yes`

Knitting R Markdown documents

Note that this might take a while



```
12m -RTS r-markdown-template.knit.md --to latex --from markdown
+autolink_bare_uris+tex_math_single_backslash --output r-markdo
wn-template.tex --lua-filter "C:\Users\wb532468\AppData\Local\R
\win-library\4.2\rmarkdown\rmarkdown\lua\pagebreak.lua" --lua-f
ilter "C:\Users\wb532468\AppData\Local\R\win-library\4.2\rmarkd
own\rmarkdown\lua\latex-div.lua" --self-contained --highlight-s
tyle tango --pdf-engine pdflatex --variable graphics --variable
 "geometry:margin=1in"
output file: r-markdown-template.knit.md
```

We'll continue with markdown syntax while it finishes

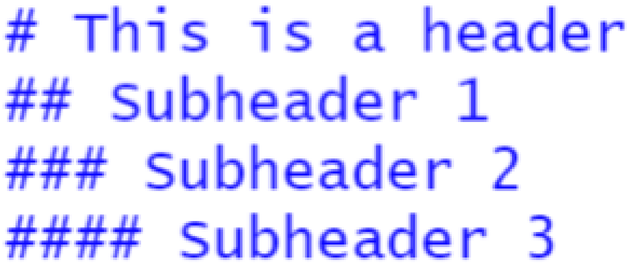
Markdown

Markdown

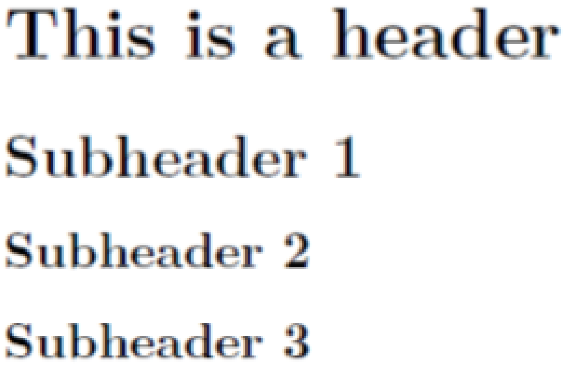
- The text part of R Markdown follows the syntax of Markdown
- Markdown is a "light" markup language. It's similar to Latex or HTML, but simpler
- Markdown was designed to be easily readable while allowing to format text and document sections

Markdown - Headers

- Headers in markdown are preceded by pound (`#`) symbols
- Additional pound symbols denote a lower level in the headers hierarchy



```
# This is a header
## Subheader 1
### Subheader 2
#### Subheader 3
```



This is a header

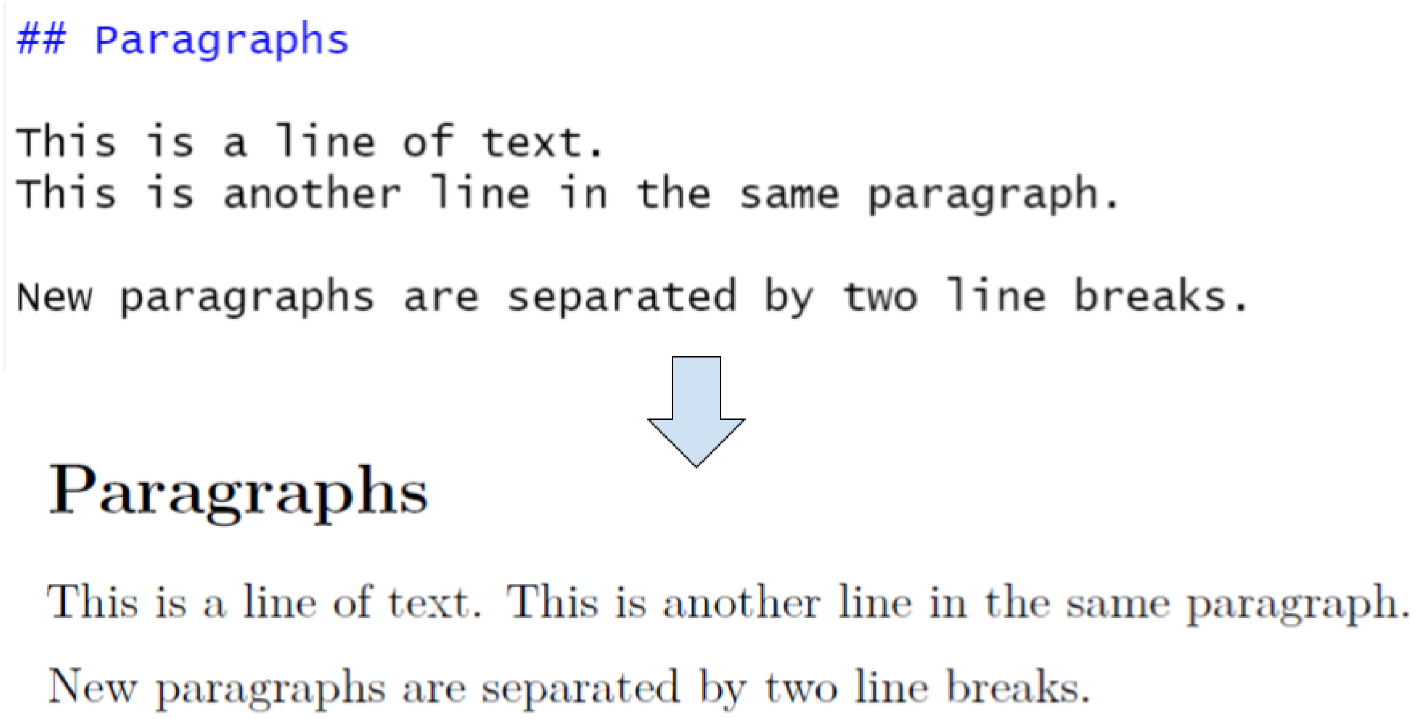
Subheader 1

Subheader 2

Subheader 3

Markdown - Paragraphs

- Text not preceded by special symbols are regular paragraphs.



```
## Paragraphs

This is a line of text.
This is another line in the same paragraph.

New paragraphs are separated by two line breaks.
```

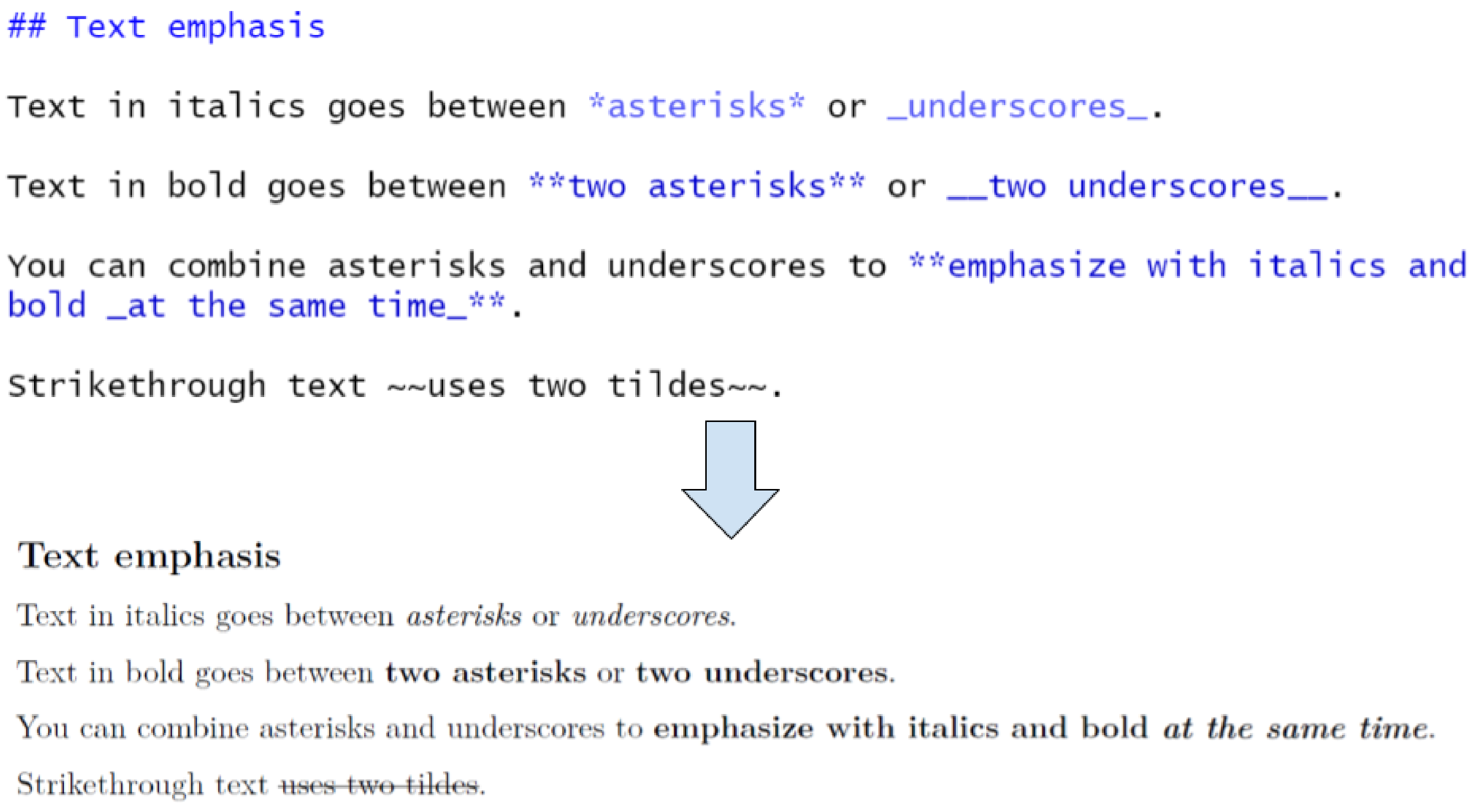
Paragraphs

This is a line of text. This is another line in the same paragraph.

New paragraphs are separated by two line breaks.

Markdown - Text emphasis

- Emphasized text is enclosed by special symbols.



```
## Text emphasis

Text in italics goes between *asterisks* or _underscores_.

Text in bold goes between **two asterisks** or __two underscores__.

You can combine asterisks and underscores to **emphasize with italics and
bold _at the same time_**.

Strikethrough text ~~uses two tildes~~.
```

Text emphasis

Text in italics goes between *asterisks* or *underscores*.

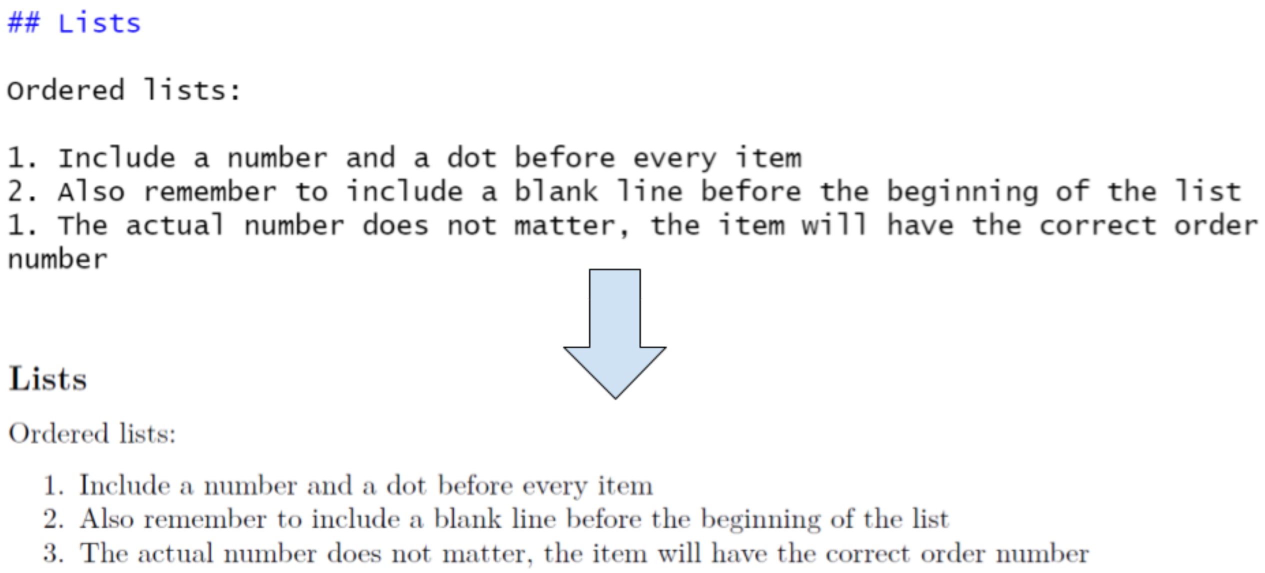
Text in bold goes between **two asterisks** or **two underscores**.

You can combine asterisks and underscores to **emphasize with italics and bold *at the same time***.

Strikethrough text ~~uses two tildes~~.

Markdown - Lists

- Markdown allows us to use both ordered and unordered lists.



```
## Lists

Ordered lists:

1. Include a number and a dot before every item
2. Also remember to include a blank line before the beginning of the list
1. The actual number does not matter, the item will have the correct order
number
```

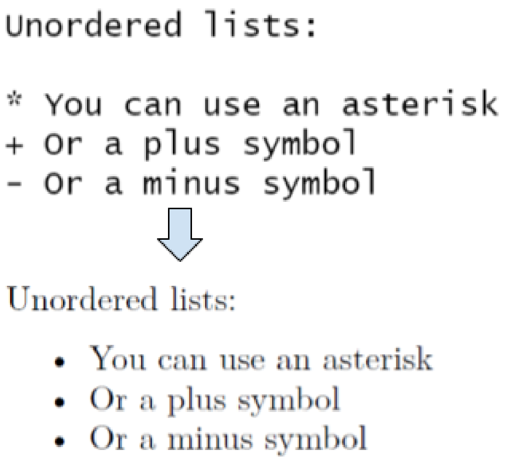
Lists

Ordered lists:

1. Include a number and a dot before every item
2. Also remember to include a blank line before the beginning of the list
3. The actual number does not matter, the item will have the correct order number

Markdown - Lists

- Markdown allows us to use both ordered and unordered lists.



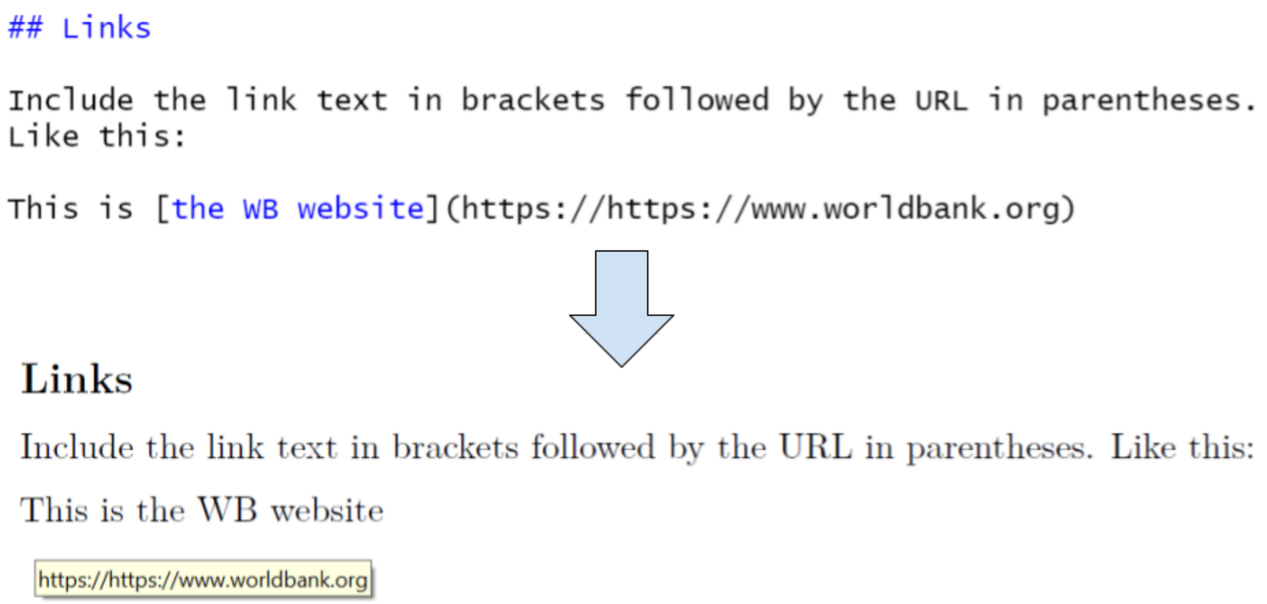
```
Unordered lists:

* You can use an asterisk
+ Or a plus symbol
- Or a minus symbol
```

Unordered lists:

- You can use an asterisk
- Or a plus symbol
- Or a minus symbol

Markdown - Links

- We can also include links as text in Markdown.



```
## Links

Include the link text in brackets followed by the URL in parentheses.
Like this:

This is [the WB website](https://https://www.worldbank.org)
```

Links

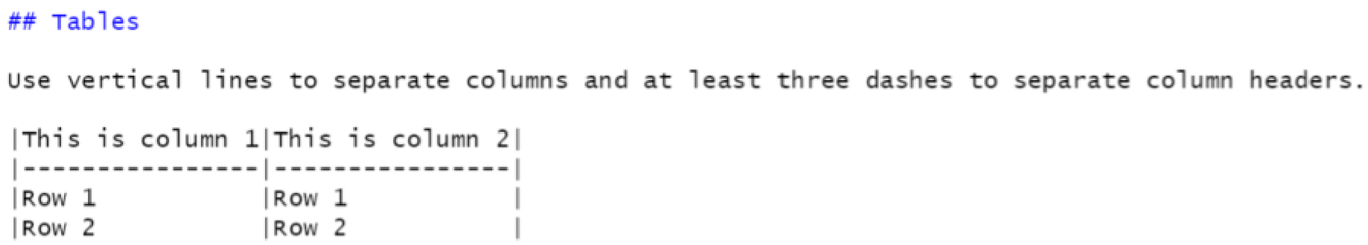
Include the link text in brackets followed by the URL in parentheses. Like this:

This is the WB website

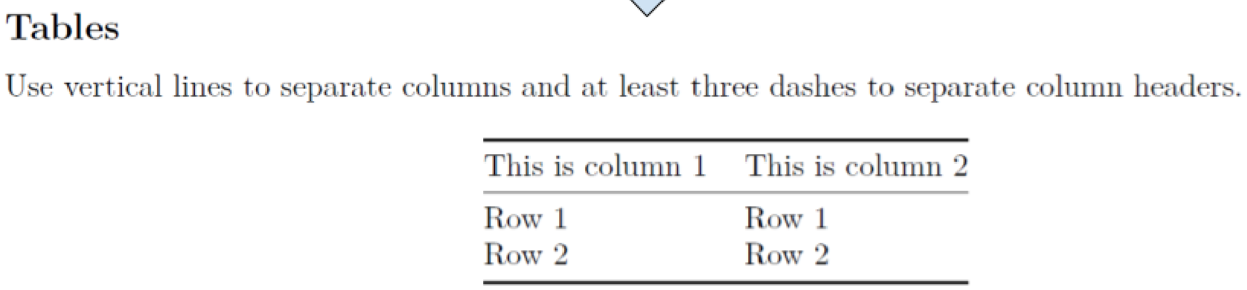
https://https://www.worldbank.org

Markdown - Tables

- Lastly, we can include tables in Markdown text.



```
## Tables

Use vertical lines to separate columns and at least three dashes to separate column headers.

|This is column 1|This is column 2|
|----------------|----------------|
|Row 1           |Row 1           |
|Row 2           |Row 2           |
```



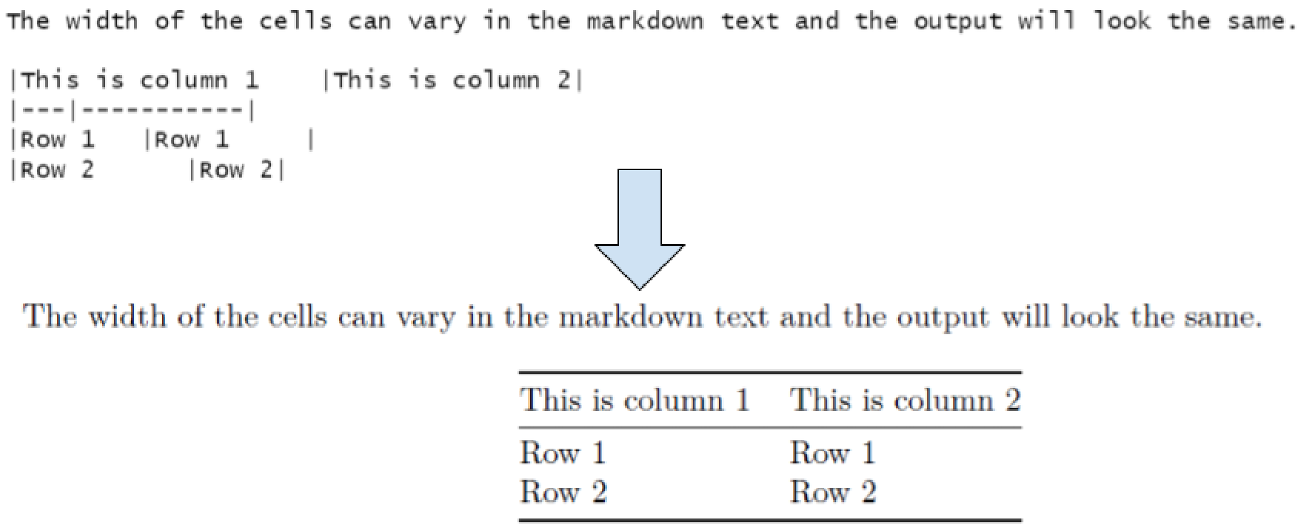
Tables

Use vertical lines to separate columns and at least three dashes to separate column headers.

This is column 1	This is column 2
Row 1	Row 1
Row 2	Row 2

Markdown - Tables

- Lastly, we can include tables in Markdown text.



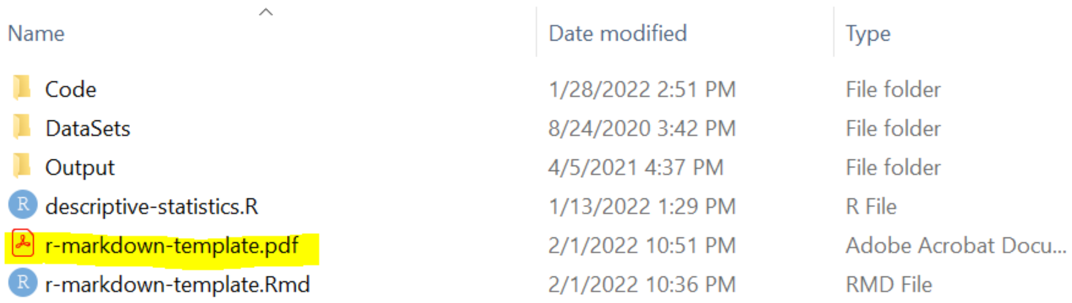
```
The width of the cells can vary in the markdown text and the output will look the same.

|This is column 1    |This is column 2|
|---|-----------|
|Row 1   |Row 1     |
|Row 2      |Row 2|
```

The width of the cells can vary in the markdown text and the output will look the same.

This is column 1	This is column 2
Row 1	Row 1
Row 2	Row 2

Exercise 1 results

- If exercise 1 worked, you'll now see this PDF file (or HTML) in the folder where you saved `r-markdown-template.Rmd`



Name	Date modified	Type
Code	1/28/2022 2:51 PM	File folder
DataSets	8/24/2020 3:42 PM	File folder
Output	4/5/2021 4:37 PM	File folder
descriptive-statistics.R	1/13/2022 1:29 PM	R File
r-markdown-template.pdf	2/1/2022 10:51 PM	Adobe Acrobat Docu...
r-markdown-template.Rmd	2/1/2022 10:36 PM	RMD File

- If it's still running, let it run until it finishes
- If it failed, try again after changing `output: html_document` in line 2

R Code

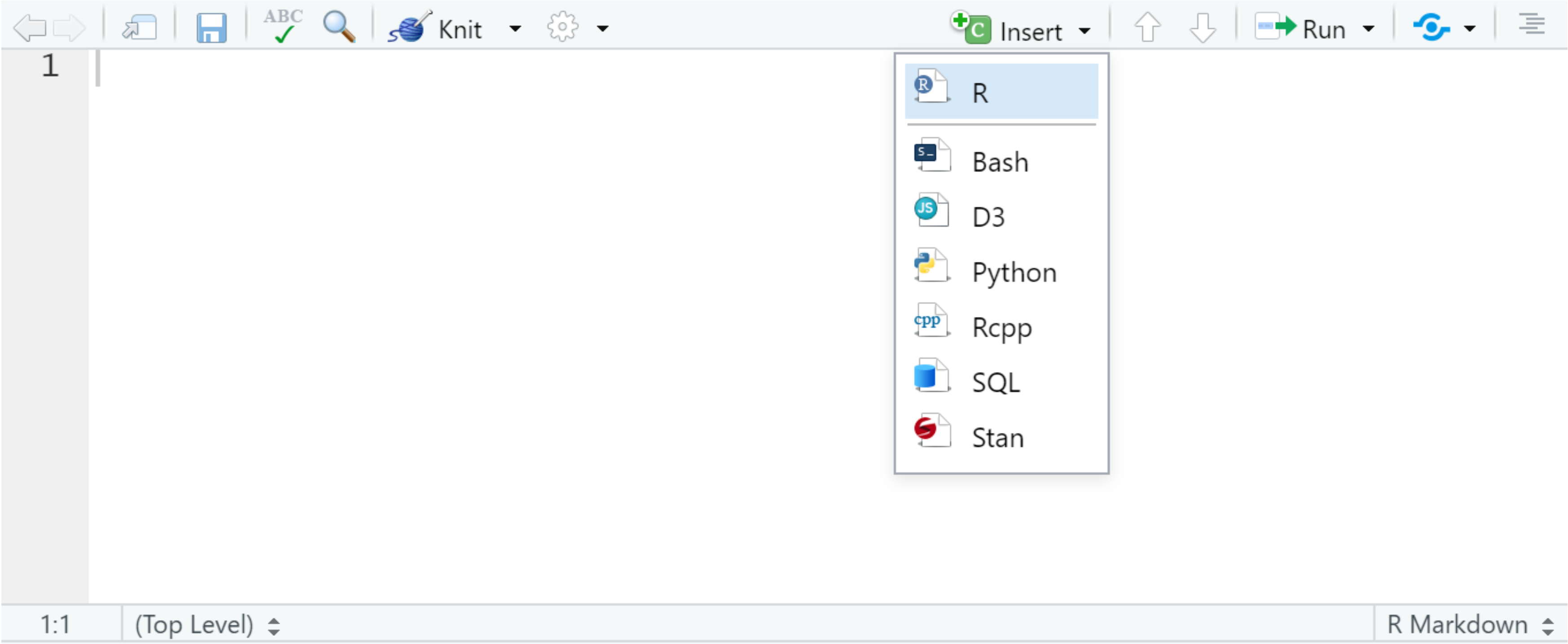
Including R code

- R code in R Markdown goes inside **fenced code blocks**, as the one below

~~~
```{r}
# Your R code goes here
```
~~~

Including R code

- To add new block, you can type the fences directly, or go to `Insert` > `R` in the script panel of RStudio, or type `CTRL` + `ALT` + `i`



Including R code

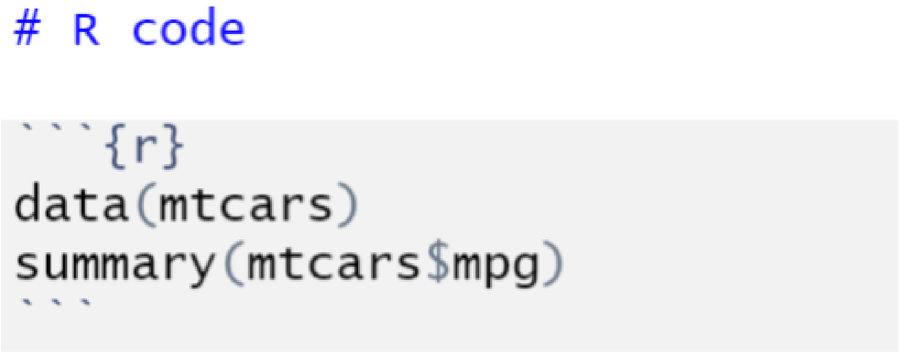
Exercise 2: Include the summary of a variable (🕑 2 min)

1. Create a header named `R Code` at the bottom of `r-markdown-template.Rmd`
2. Create a new fenced code block where you load the dataset `mtcars`
   - `mtcars` is a built-in dataset. Load it with: `data(mtcars)`
3. Inside the same block, get the summary of the variable `mpg` with `summary(mtcars$mpg)`
4. Knit. You'll have to **close the PDF document if you have it opened**

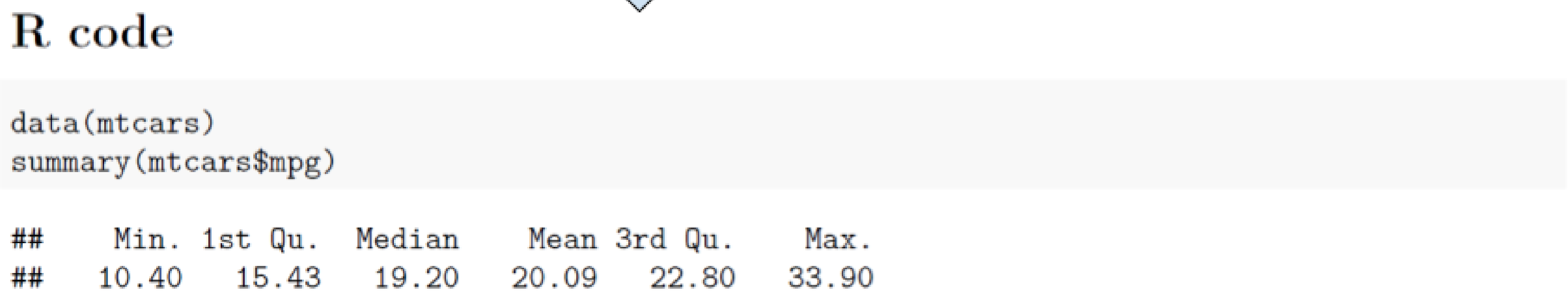
Including R code

~~~
## R Code

```{r}
data(mtcars)
summary(mtcars$mpg)
```
~~~

Including R code



~~~
# R code

```{r}
data(mtcars)
summary(mtcars$mpg)
```
~~~



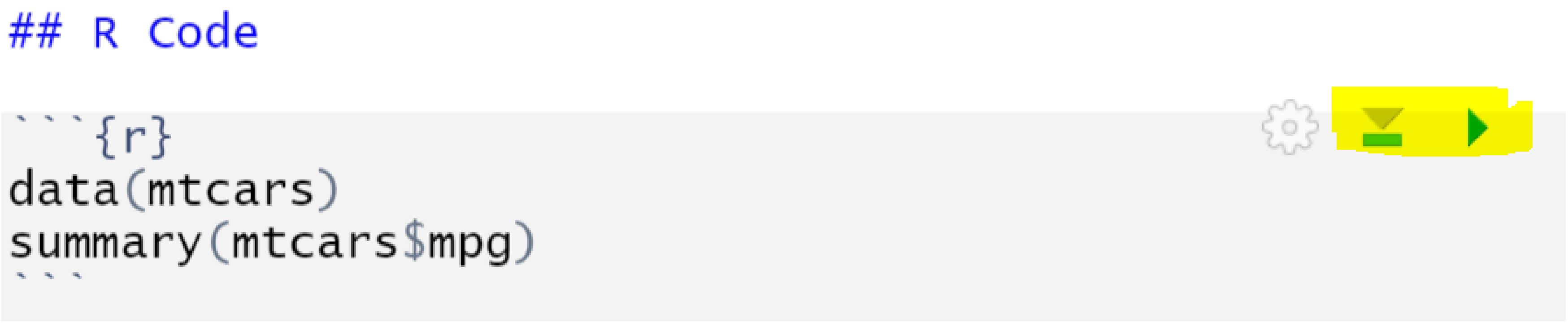
R code

```
data(mtcars)
summary(mtcars$mpg)
```

```
##     Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    10.40   15.43   19.20   20.09   22.80   33.90
```

Including R code

- What about running only the code block and not knitting the document?
- You can do that with the ▶ icon at the upper right corner of the block
- The other icon (⏯) will run all previous code blocks until this block



~~~~
## R Code

```{r}
data(mtcars)
summary(mtcars$mpg)
```
~~~~

Including R code

- Note that the output echoes both the code and the output
- What if we wanted to include the output but not the code?
- We use the argument `echo = FALSE` in the fenced code block for that
- Code block arguments are separated by commas inside the curly brackets, as in: `{r, echo = FALSE}`

Including R code

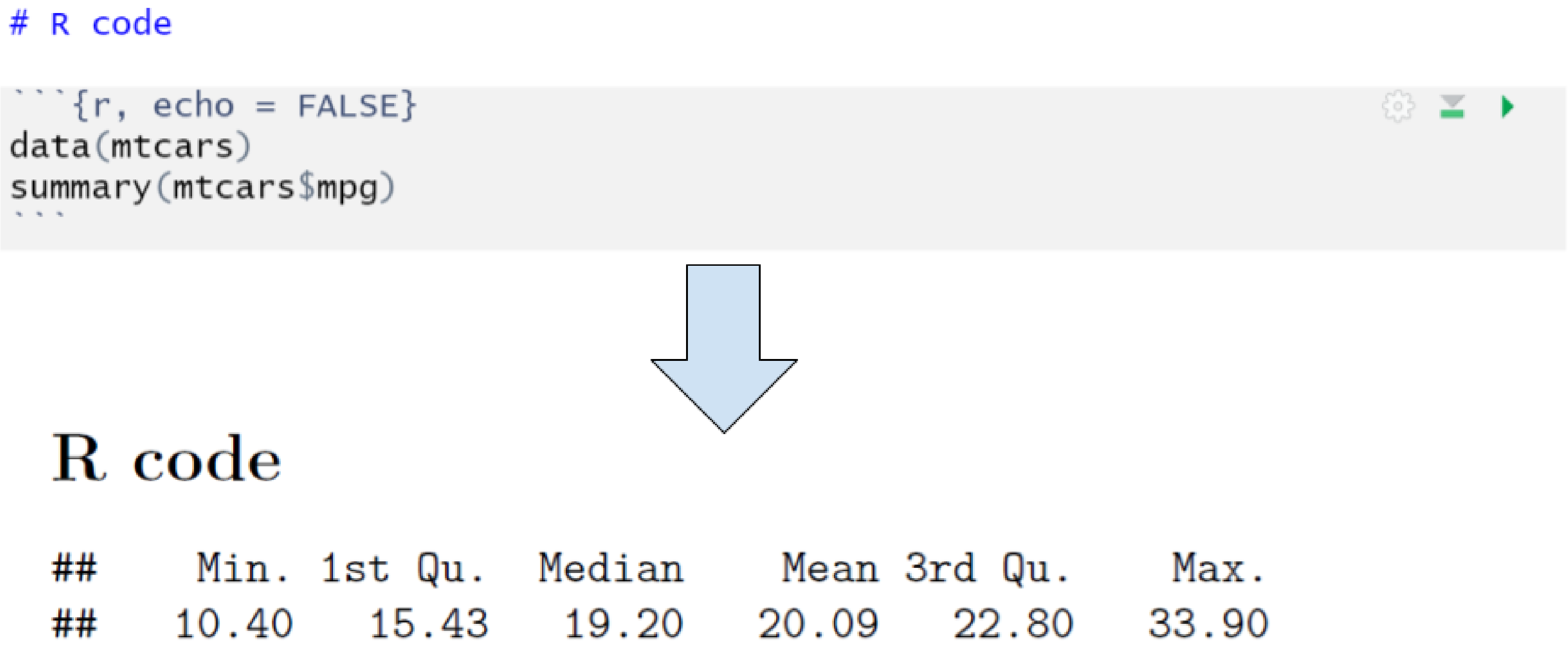
Exercise 3: Omit the code when knitting R code (🕓 1 min)

1. Add the option `echo = FALSE` to the fenced code block created in exercise 2
2. Knit the document and see how it's different now

Including R code

~~~
```{r, echo = FALSE}
data(mtcars)
summary(mtcars$mpg)
```
~~~

Including R code



~~~
# R code

```{r, echo = FALSE}
data(mtcars)
summary(mtcars$mpg)
```
~~~

R code

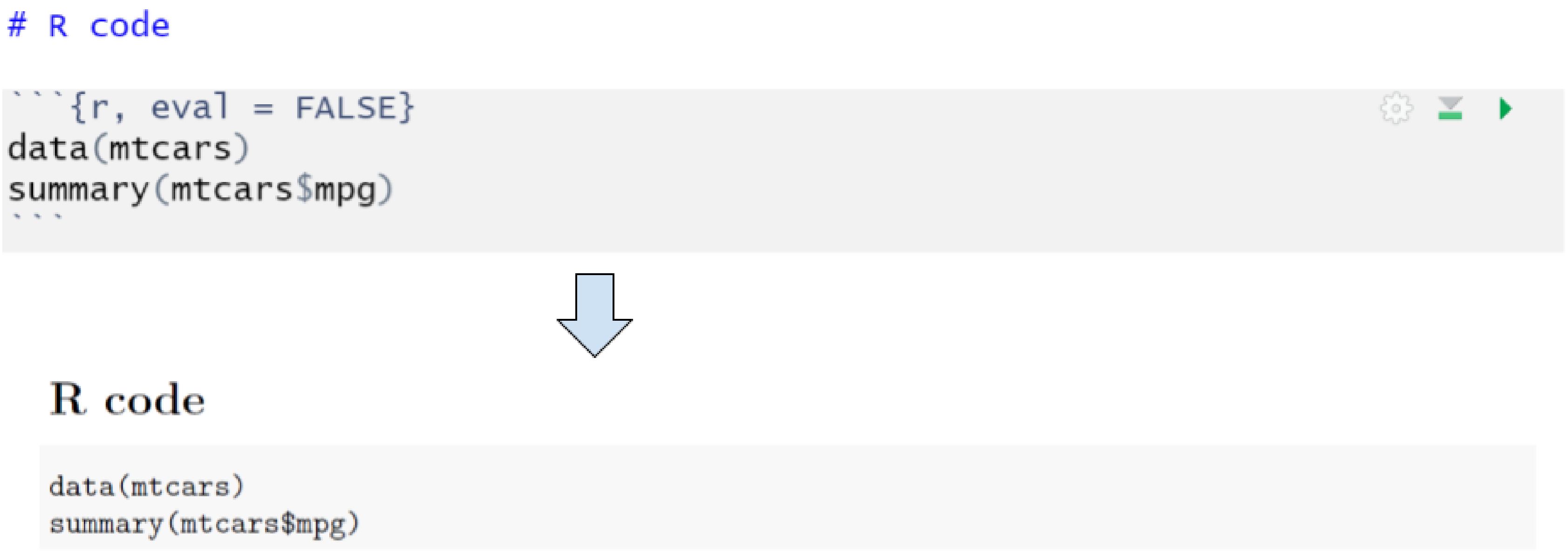
```
##     Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    10.40   15.43   19.20   20.09   22.80   33.90
```

Including R code

- To include only R code but not the output, we use the option `eval = FALSE`

~~~
```{r, eval = FALSE}
data(mtcars)
summary(mtcars$mpg)
```
~~~

Including R code



~~~
# R code

```{r, eval = FALSE}
data(mtcars)
summary(mtcars$mpg)
```
~~~

R code

```
data(mtcars)
summary(mtcars$mpg)
```

R Plots

Including R plots

- Adding R plots is similar to adding R code
- Include the code producing the plot in a fenced block
- The block option `echo = FALSE` is useful when we only want to include the plot but not the code producing it

Including R plots

Exercise 4: Include an R plot in your document (🕓 2 min)

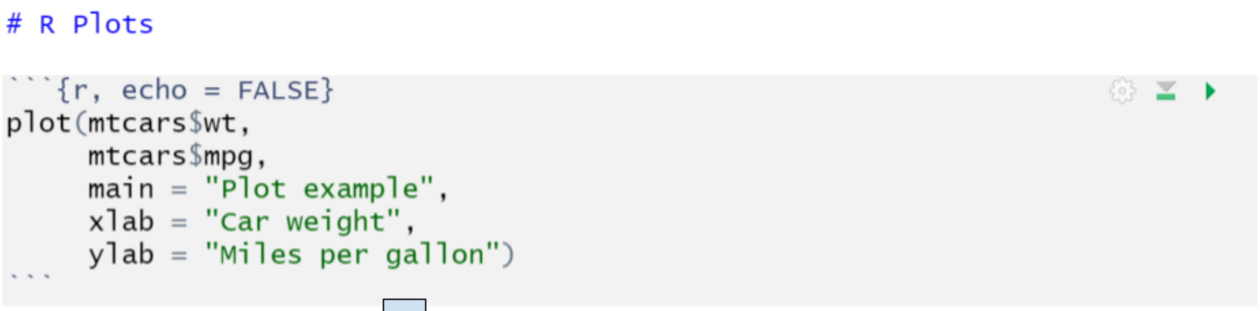
1. Create a header named `R Plots`
2. Create a new fenced code block with the option `echo = FALSE`
3. Add the following code inside the new block:

```r
plot(mtcars$wt,
     mtcars$mpg,
     main = "Plot example",
     xlab = "Car weight",
     ylab = "Miles per gallon")
```

Including R plots

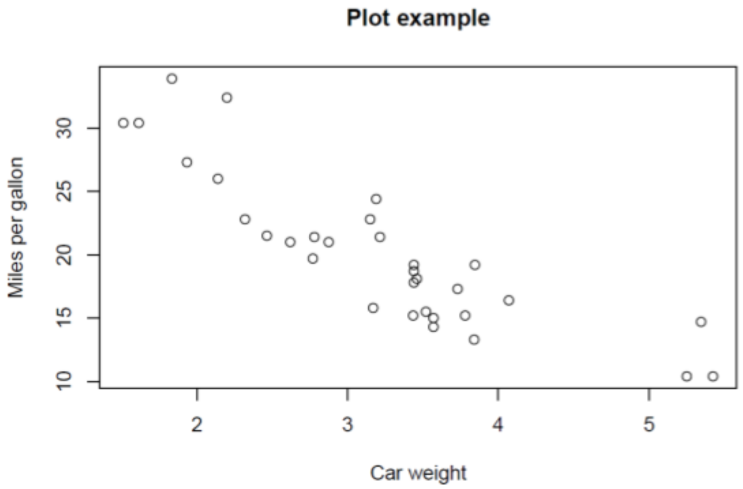
~~~
# R plots

```{r, echo = FALSE}
plot(mtcars$wt,
     mtcars$mpg,
     main = "Plot example",
     xlab = "Car weight",
     ylab = "Miles per gallon")
```
~~~

Including R plots



~~~
# R Plots

```{r, echo = FALSE}
plot(mtcars$wt,
     mtcars$mpg,
     main = "Plot example",
     xlab = "Car weight",
     ylab = "Miles per gallon")
```
~~~

R Plots



Inline code

Including code inline

- Inline code is enclosed by backtick followed by an r (`` `r ``) and a single backtick
- For example:

```
The mean of mpg is `r mean(mtcars$mpg)`.
```

- Will be rendered as:

```
The mean of mpg is 20.090625.
```

- Note that inline code doesn't go enclosed in code blocks, it's just regular Markdown text

Including code inline

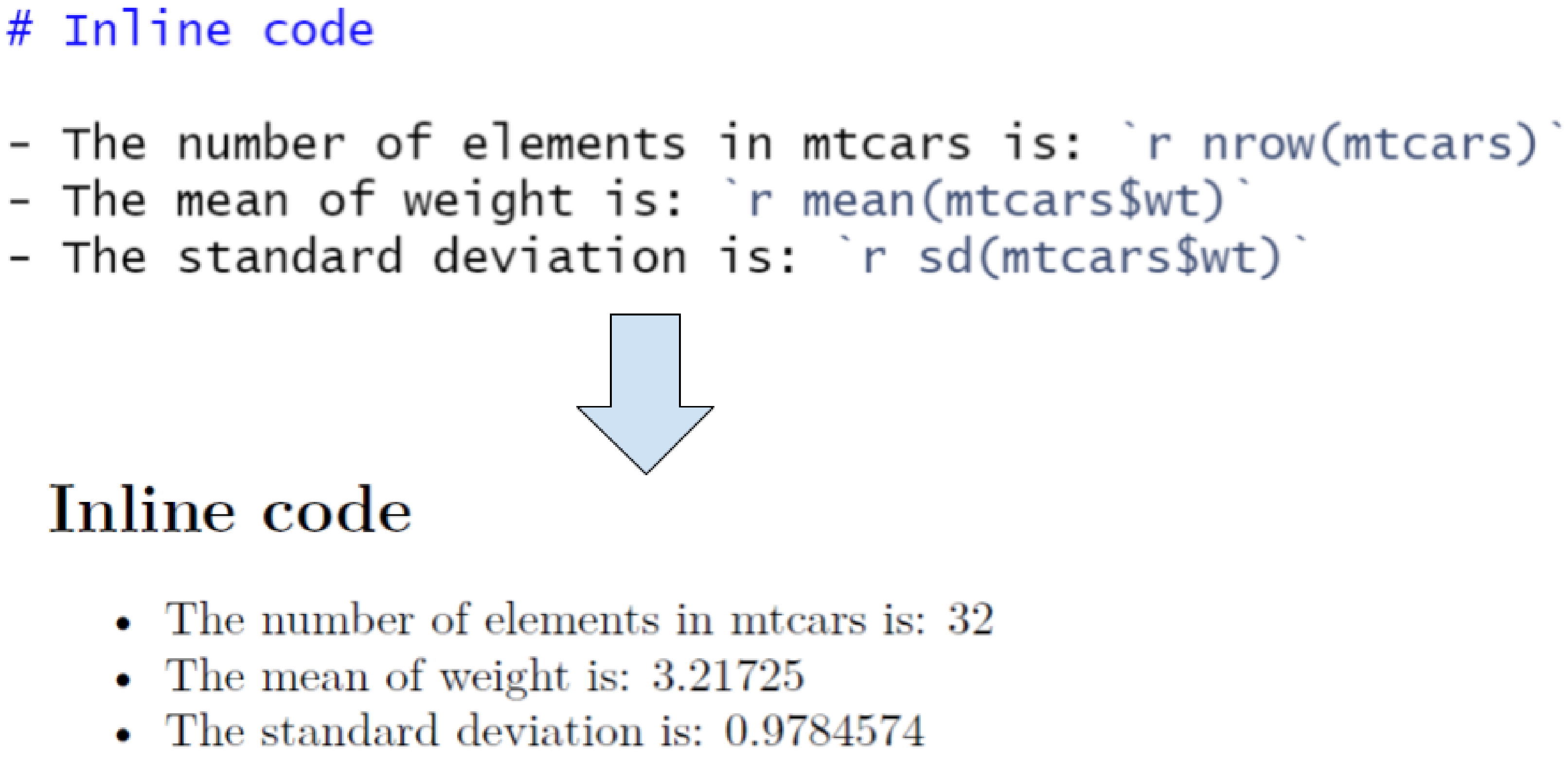
Exercise 5 (🕒 2 min)

1. Create a new header named `Inline code` in `markdown-template.Rmd`
2. Add an unordered list with the following text and include inline R code to render the corresponding numbers in each case
   - The number of elements in mtcars is: (use function `nrow(mtcars)`)
   - The mean of weight is: (use function `mean(mtcars$wt)`)
   - The standard deviation is: (use function `sd(mtcars$wt)`)

Including code inline

```
# Inline code

- The number of elements in mtcars is `r nrow(mtcars)`
- The mean of weight is `r mean(mtcars$wt)`
- The standard deviation is `r sd(mtcars$wt)`
```

Including code inline



```
# Inline code

- The number of elements in mtcars is: `r nrow(mtcars)`
- The mean of weight is: `r mean(mtcars$wt)`
- The standard deviation is: `r sd(mtcars$wt)`
```

Inline code

- The number of elements in mtcars is: 32
- The mean of weight is: 3.21725
- The standard deviation is: 0.9784574

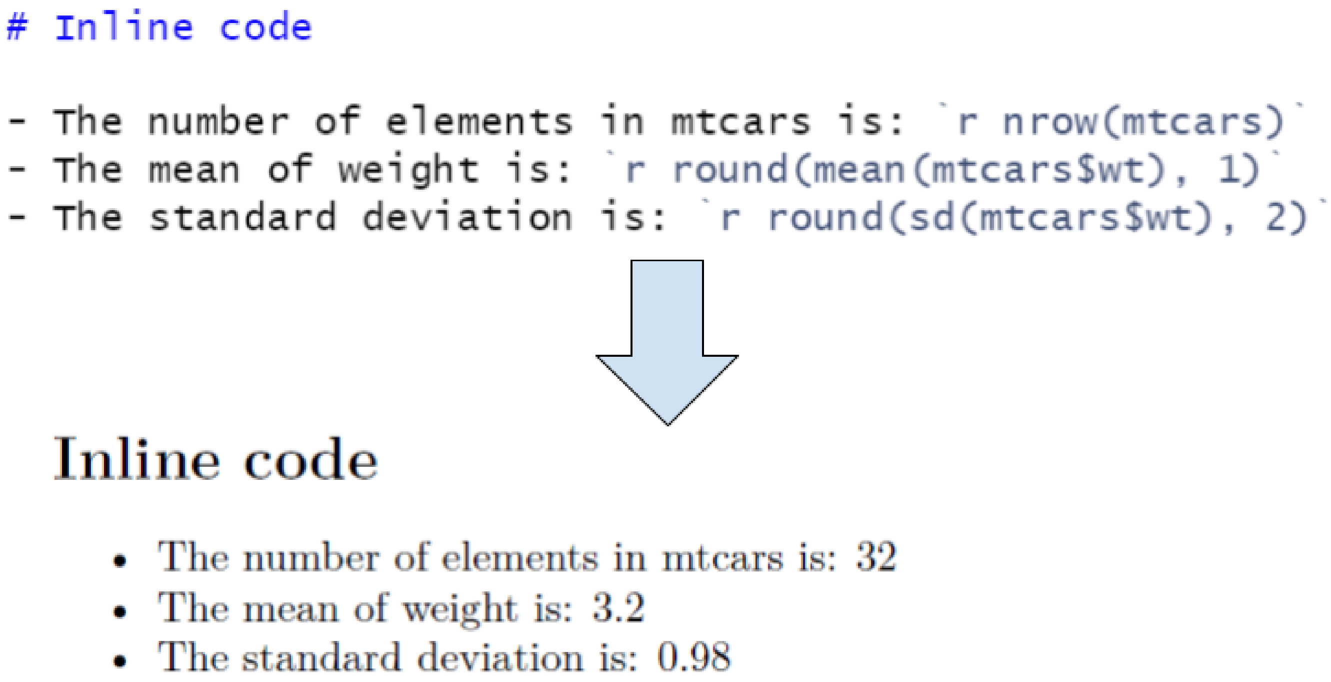
Including code inline

You can use the function `round()` to control the number of decimals displayed.

```
# Inline code

- The number of elements in mtcars is `r nrow(mtcars)`
- The mean of weight is `r round(mean(mtcars$wt), 1)`
- The standard deviation is `r round(sd(mtcars$wt), 2)`
```

Including code inline

You can use the function `round()` to control the number of decimals displayed.



```
# Inline code

- The number of elements in mtcars is: `r nrow(mtcars)`
- The mean of weight is: `r round(mean(mtcars$wt), 1)`
- The standard deviation is: `r round(sd(mtcars$wt), 2)`
```

Inline code

- The number of elements in mtcars is: 32
- The mean of weight is: 3.2
- The standard deviation is: 0.98

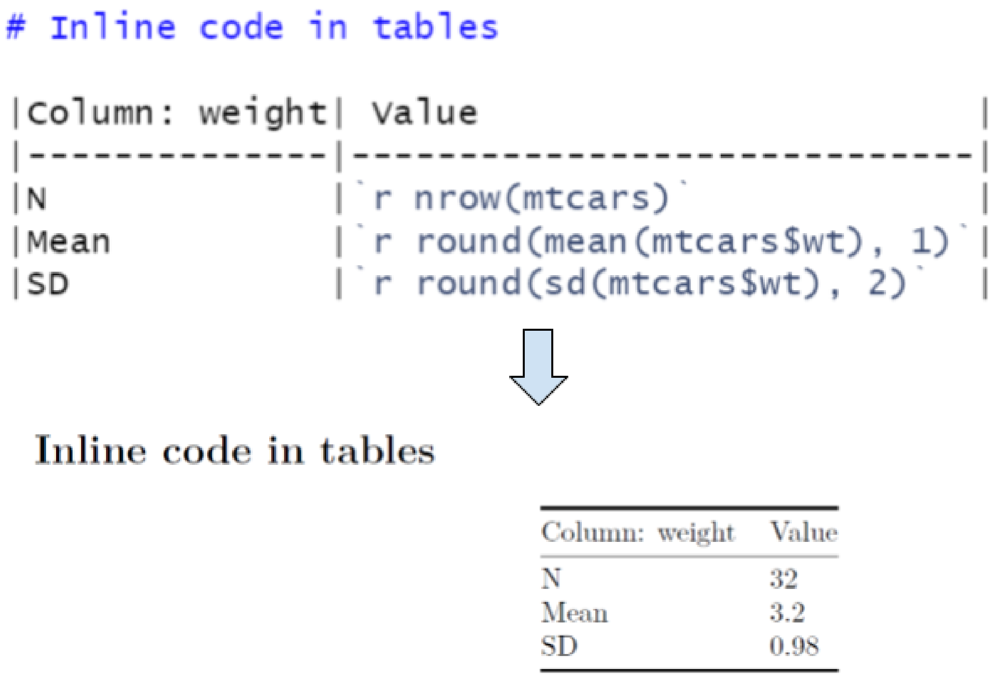
Including code inline

You can also combine R inline code with the markdown syntax for tables to produce statistics tables.

```
# Inline code in tables

|Column: weight| Value                       |
|--------------|-----------------------------|
|N             |`r nrow(mtcars)`             |
|Mean          |`r round(mean(mtcars$wt), 1)`|
|SD            |`r round(sd(mtcars$wt), 2)`  |
```

Including code inline

You can also combine R inline code with the markdown syntax for tables to produce statistics tables.



```
# Inline code in tables

|Column: weight| Value                          |
|--------------|--------------------------------|
|N             | `r nrow(mtcars)`               |
|Mean          | `r round(mean(mtcars$wt), 1)`  |
|SD            | `r round(sd(mtcars$wt), 2)`    |
```

Inline code in tables

Column: weight	Value
N	32
Mean	3.2
SD	0.98

Including regression outputs

Including regression outputs

- In a previous session, we saw that we can produce regression tables in LaTeX
- We can use code producing LaTeX outputs along with the code block option `results = "asis"` to display them in the knitted document

Including regression outputs - Stargazer

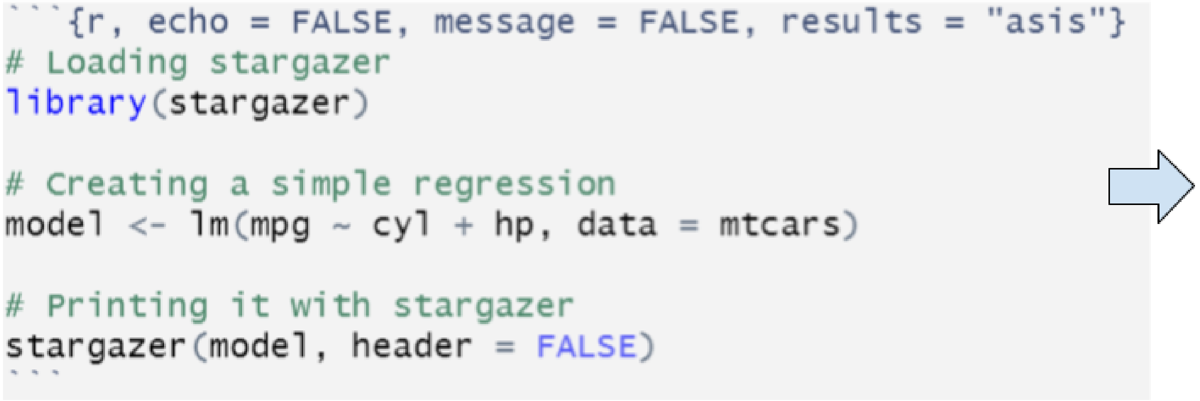
- First, we'll start with the function `stargazer()` from the package `stargazer`
- The first argument of `stargazer()` is a regression result
- We also include the arguments `echo = FALSE` and `message = FALSE` in the code block to omit printing the code and messages that appear when loading stargazer
- In `stargazer()` we include `header = FALSE` to omit printing stargazer metadata

Important: When using external packages in RMarkdown, you need to have them loaded in a code block regardless of if they're already loaded in your current session. Libraries have to load again for each knit.

Including regression outputs - Stargazer

~~~
```{r, echo = FALSE, message = FALSE, results = "asis"}
# Loading stargazer
library(stargazer)

# Creating a simple regression
model <- lm(mpg ~ cyl + hp, data = mtcars)

# Printing it with stargazer
stargazer(model, header = FALSE) # add: type = "html" if knitting to HTML
```
~~~

Including regression outputs - Stargazer



~~~
```{r, echo = FALSE, message = FALSE, results = "asis"}
# Loading stargazer
library(stargazer)

# Creating a simple regression
model <- lm(mpg ~ cyl + hp, data = mtcars)

# Printing it with stargazer
stargazer(model, header = FALSE)
```
~~~

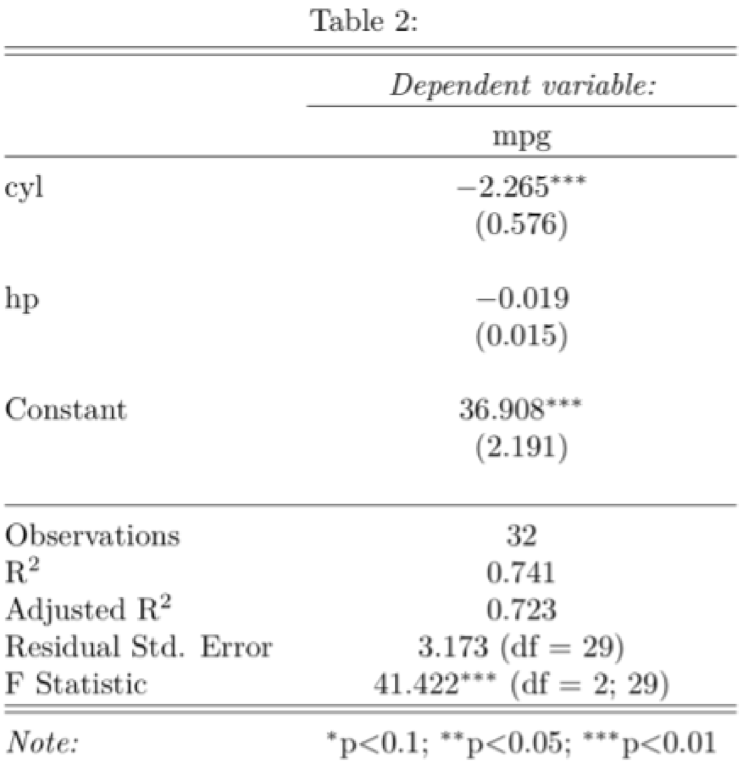


Table 2:

	Dependent variable:
	mpg
cyl	−2.265***
	(0.576)
hp	−0.019
	(0.015)
Constant	36.908***
	(2.191)
Observations	32
R^2	0.741
Adjusted R^2	0.723
Residual Std. Error	3.173 (df = 29)
F Statistic	41.422*** (df = 2; 29)
Note:	*p<0.1; **p<0.05; ***p<0.01

Including regression outputs - Stargazer

Exercise 6 (🕒 3 min)

1. Create a new header named `Regressions - Stargazer` in `r-markdown-template.Rmd`
2. Add a new code block with the arguments `echo = FALSE` and `results = "asis"`
3. Load stargazer in the code block
4. Add a regression of the variable `mpg` on `wt` and `hp`
5. Use stargazer's arguments `header = FALSE`, `title = "your_title"` and `omit = c("Constant")` to customize your table
   - If your output is HTML instead of PDF, include the argument `type = "html"` in `stargazer()`

Including regression outputs - Stargazer

~~~
# Regressions - Stargazer

```{r, echo = FALSE, message = FALSE, results = "asis"}
library(stargazer)
model <- lm(mpg ~ wt + hp, data = mtcars)
stargazer(model,
          header = FALSE,
          title = "Best table ever",
          omit = c("Constant"))
```
~~~

Including regression outputs - Stargazer

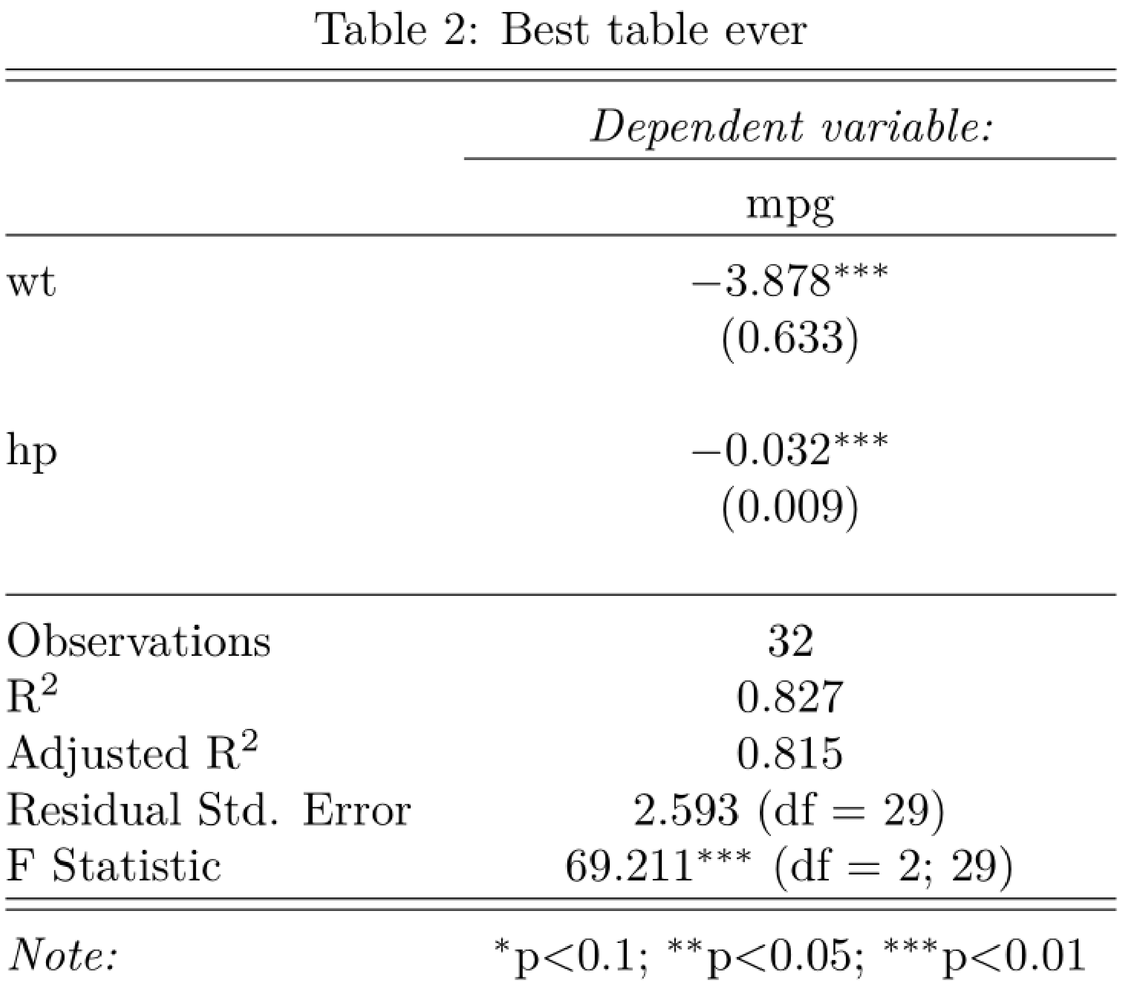


Table 2: Best table ever

	Dependent variable:
	mpg
wt	-3.878^{***}
	(0.633)
hp	-0.032^{***}
	(0.009)
Observations	32
R^2	0.827
Adjusted R^2	0.815
Residual Std. Error	2.593 (df = 29)
F Statistic	69.211^{***} (df = 2; 29)
Note:	*p<0.1; **p<0.05; ***p<0.01

Including regression outputs - Huxtable

- Remember `huxtable`? we can also use it to include regression tables in R Markdown
- The advantage of using `huxtable` compared to `stargazer` is that we don't have to define the type of output we're generating with R Markdown. `huxtable` automatically detects it and will transform the output as needed in the resulting document
- `huxtable` has an important disadvantage, though: it requires to install external libraries in your local LaTeX installation

Including regression outputs - Huxtable

- Conveniently, the library `huxtable` has a function that handles that installation for us (needed only if you're knitting to PDF)

```
# Only if you're knitting to PDF:
huxtable::install_latex_dependencies()
```

- Once this finishes, we can use `huxtable` with R Markdown

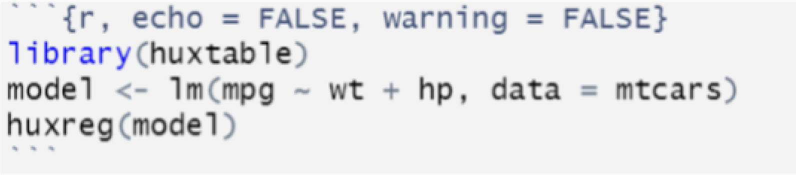
Including regression outputs - Huxtable

- For regressions, we use the function `huxreg()` as in the example below
- Note that the option `results = "asis"` is not used with `huxtable`

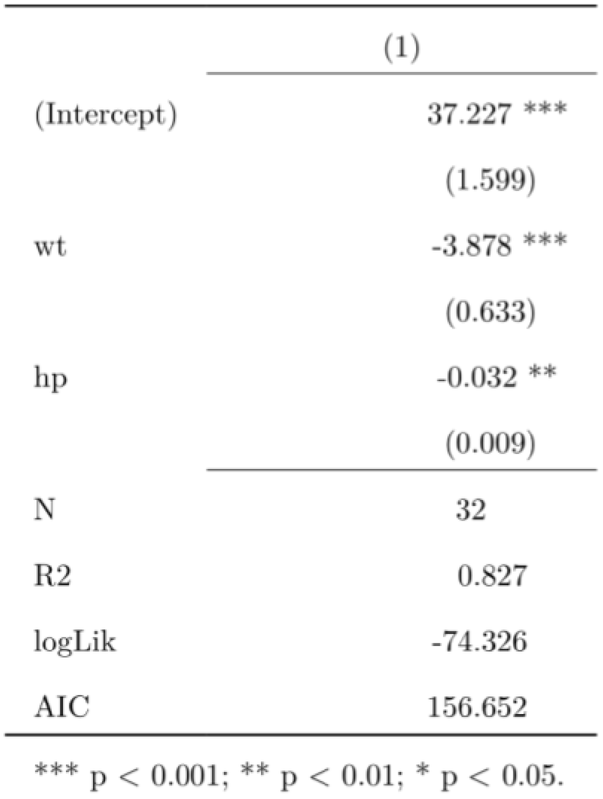
~~~
```{r, echo = FALSE, warning = FALSE}
library(huxtable)
model <- lm(mpg ~ wt + hp, data = mtcars)
huxreg(model)
```
~~~

Including regression outputs - Huxtable

- For regressions, we use the function `huxreg()` as in the example below
- Note that the option `results = "asis"` is not used with `huxtable`



~~~
```{r, echo = FALSE, warning = FALSE}
library(huxtable)
model <- lm(mpg ~ wt + hp, data = mtcars)
huxreg(model)
```
~~~



	(1)
(Intercept)	37.227 ***
	(1.599)
wt	-3.878 ***
	(0.633)
hp	-0.032 **
	(0.009)
N	32
R2	0.827
logLik	-74.326
AIC	156.652

*** p < 0.001; ** p < 0.01; * p < 0.05.

Including regression outputs - Huxtable

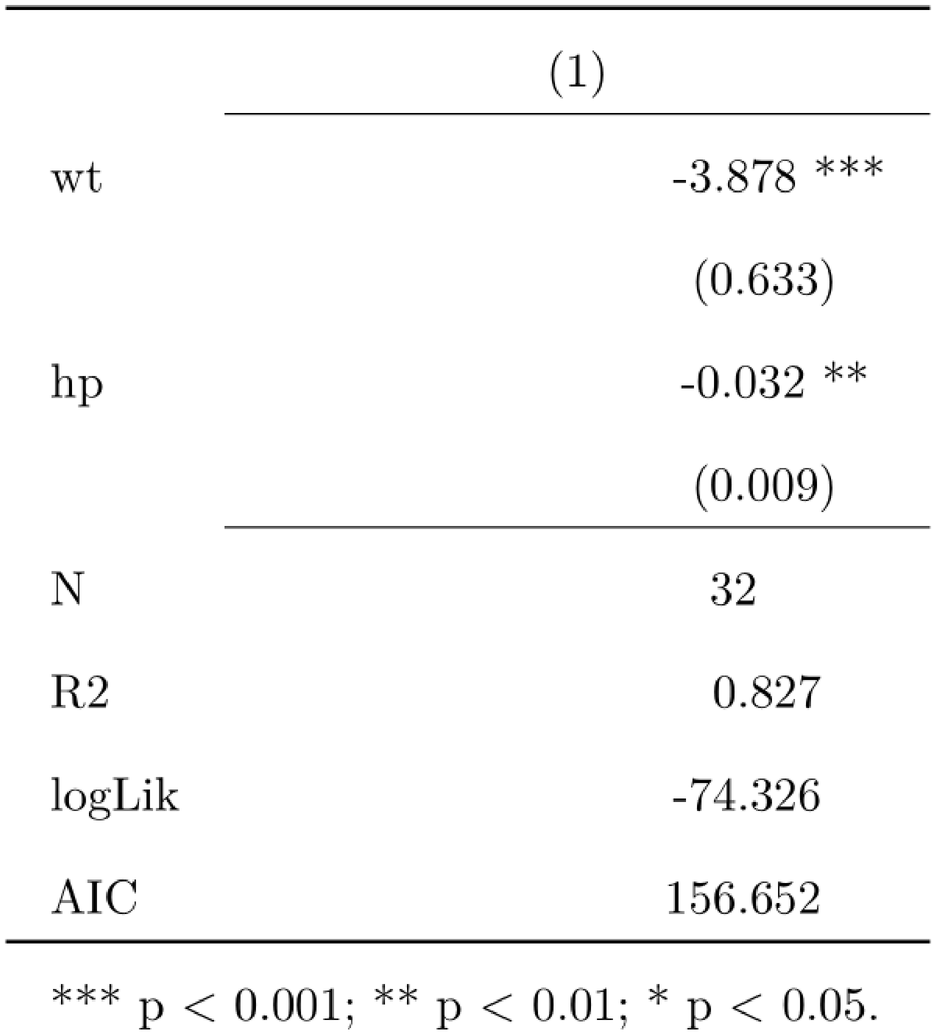
Exercise 7: Now with Huxtable (🕓 2 min)

1. Create a new header named `Regressions - Huxtable` in `r-markdown-template.Rmd`
2. Add a new code block with the argument `echo = FALSE`
3. Load huxtable in the code block
4. Add a regression table of the variable `mpg` on `wt` and `hp` using `huxreg()`
5. Use huxreg's argument `omit_coefs = c("(Intercept)")` to customize your table

Including regression outputs - Huxtable

~~~
# Regressions - Huxtable

```{r, echo = FALSE, warning = FALSE}
library(huxtable)
model <- lm(mpg ~ wt + hp, data = mtcars)
huxreg(model,
       omit_coefs = c("(Intercept)"))
```
~~~

Including regression outputs - Huxtable



	(1)
wt	-3.878 ***
	(0.633)
hp	-0.032 **
	(0.009)
N	32
R2	0.827
logLik	-74.326
AIC	156.652

*** p < 0.001; ** p < 0.01; * p < 0.05.

Including regression outputs - Huxtable

- If you want to include a title in your regression, use the command `set_caption()` with the result of `huxreg()` as argument

~~~
```{r, echo = FALSE, warning = FALSE}
library(huxtable)
library(dplyr)
model <- lm(mpg ~ wt + hp, data = mtcars)
table <- huxreg(model,
      omit_coefs = c("(Intercept)"))
table %>% set_caption("Another nice table")
```
~~~

Including regression outputs - Huxtable

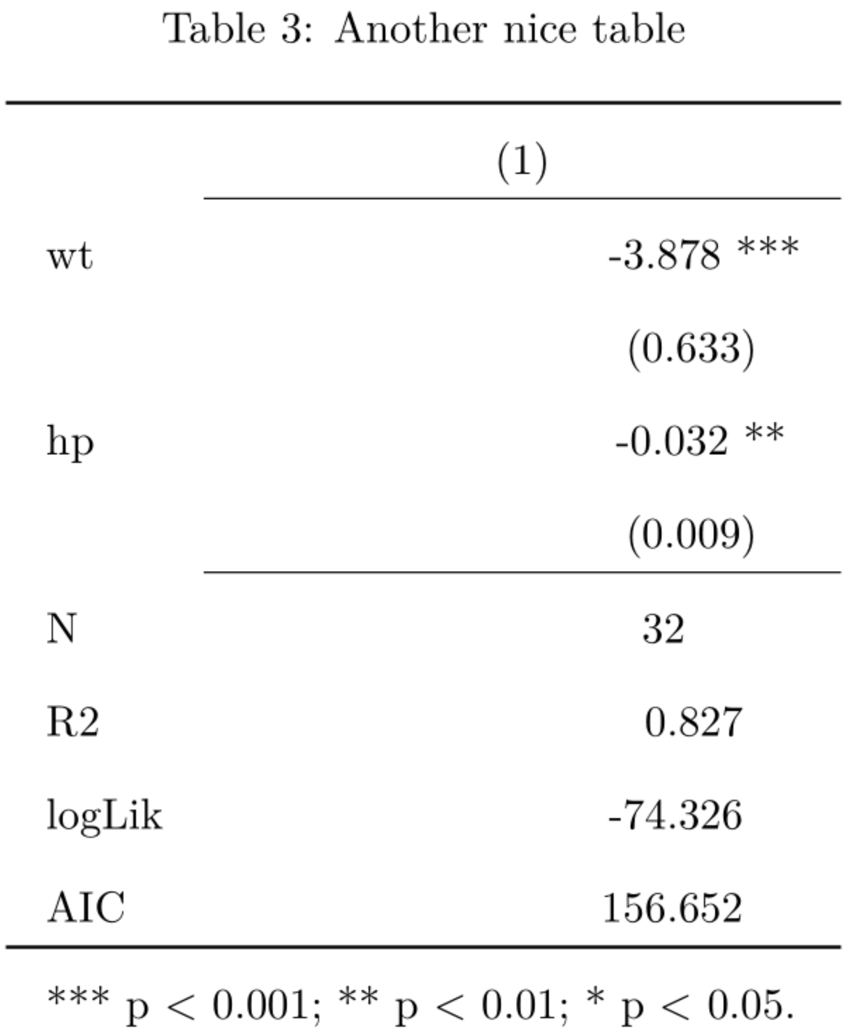


Table 3: Another nice table

	(1)
wt	-3.878 ***
	(0.633)
hp	-0.032 **
	(0.009)
N	32
R2	0.827
logLik	-74.326
AIC	156.652

*** p < 0.001; ** p < 0.01; * p < 0.05.

Thank you!

Annex

Annex - Opening a new R Markdown in R Studio

- Go to `File` > `New File` > `R Markdown`
- You can register the author name and the document title. This can be changed later if needed
- You can also define the default output format (HTML, PDF, Word). This can also be changed later
- Selecting `OK` will generate a template with document sections and code blocks that you can modify
- Selecting `Create Empty Document` will ignore the author, title, and output format registered and will result in a completely blank R Markdown document

Annex - Author, title, and output type

- The section enclosed in `---` at the beginning of the document can contain the author, title, and default output format
- You can add the author and document title with `author: NAME` and `title: TITLE`
- You can also change the default output format. Some options are:
  - `output: html_document`
  - `output: pdf_document`
  - `output: word_document`
  - `output: beamer_presentation`

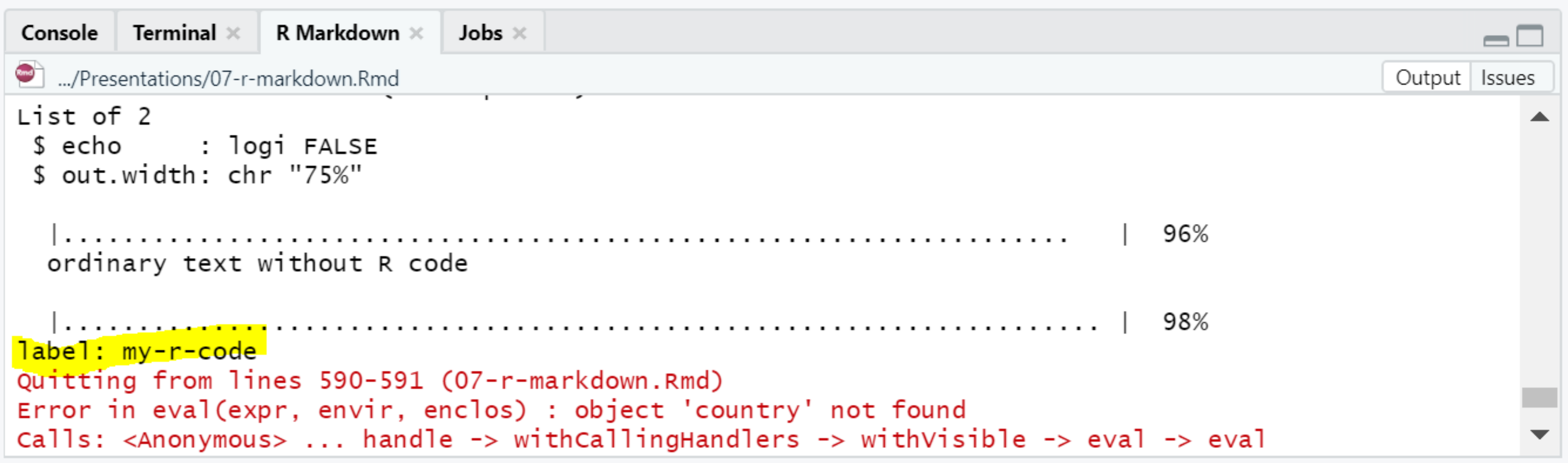
Annex - Naming R code blocks

- You can name R code blocks if you add the name after the `r` in the initial brackets
- The example below has the name `my-r-code`

~~~~
```{r my-r-code}
summary(mtcars$mpg)
```
~~~~

Annex - Naming R code blocks

- This is very convenient to debug code blocks by clicking on `Output` under the `R Markdown` tab of the console, in case your file has an error



```
List of 2
 $ echo     : logi FALSE
 $ out.width: chr "75%"

  |..................................................................    |  96%
  ordinary text without R code

  |......................................................................|  98%
label: my-r-code
Quitting from lines 590-591 (07-r-markdown.Rmd)
Error in eval(expr, envir, enclos) : object 'country' not found
Calls: <Anonymous> ... handle -> withCallingHandlers -> withVisible -> eval -> eval
```

Annex - Including images

- The Markdown syntax to include images is: `![Image name](path/to/image)`
- For example:

```
![R logo](img/r-markdown/r-logo.jpg)
```

- Renders:

Annex - Including a LaTeX preamble in a PDF doc

- If you want to further customize a PDF document in R Markdown and you're familiar with LaTeX, you can include a LaTeX preamble that will be executed when you knit your document
- To enable this feature, replace `output: pdf_document` with the following code in the section enclosed by the three dashes (`---`) at the beginning of your document

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

Annex - Complete regression table using Stargazer

~~~
```{r, echo = FALSE, message = FALSE, results = "asis"}
library(stargazer)
reg1 <- lm(mpg ~ wt + hp, data = mtcars)
reg2 <- lm(mpg ~ wt + hp + factor(gear), data = mtcars)
reg3 <- lm(qsec ~ wt + hp, data = mtcars)
reg4 <- lm(qsec ~ wt + hp + factor(gear), data = mtcars)

stargazer(reg1,
          reg2,
          reg3,
          reg4,
          title = "Best table ever",
          keep = c('wt', 'hp'),
          covariate.labels = c('Weight',
                               'Horsepower'),
          dep.var.labels = c('Miles per Gallon',
                             '1/4 Mile Time'),
          dep.var.caption = '',
          add.lines = list(c('N Gears FE', 'No', 'Yes', 'No', 'Yes')),
          keep.stat = c('n', 'adj.rsq'),
          header = FALSE,
          notes = 'Standard errors in parentheses')
```
~~~

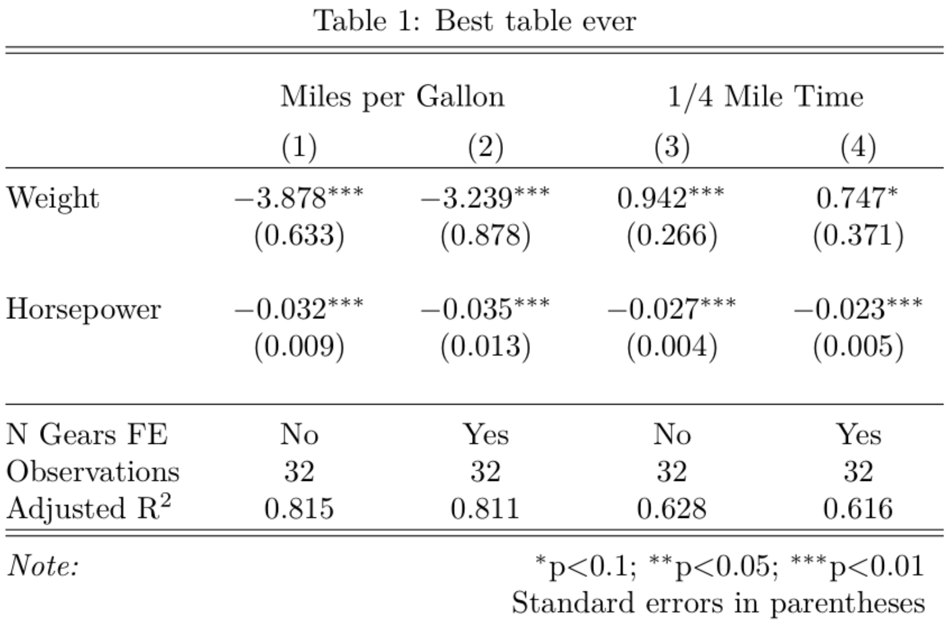


Table 1: Best table ever

	Miles per Gallon		1/4 Mile Time	
	(1)	(2)	(3)	(4)
Weight	-3.878^{***}	-3.239^{***}	0.942^{***}	0.747^{*}
	(0.633)	(0.878)	(0.266)	(0.371)
Horsepower	-0.032^{***}	-0.035^{***}	-0.027^{***}	-0.023^{***}
	(0.009)	(0.013)	(0.004)	(0.005)
N Gears FE	No	Yes	No	Yes
Observations	32	32	32	32
Adjusted R^2	0.815	0.811	0.628	0.616

Note: $^{*}p<0.1$; $^{**}p<0.05$; $^{***}p<0.01$
Standard errors in parentheses

Annex - Complete regression table using Stargazer

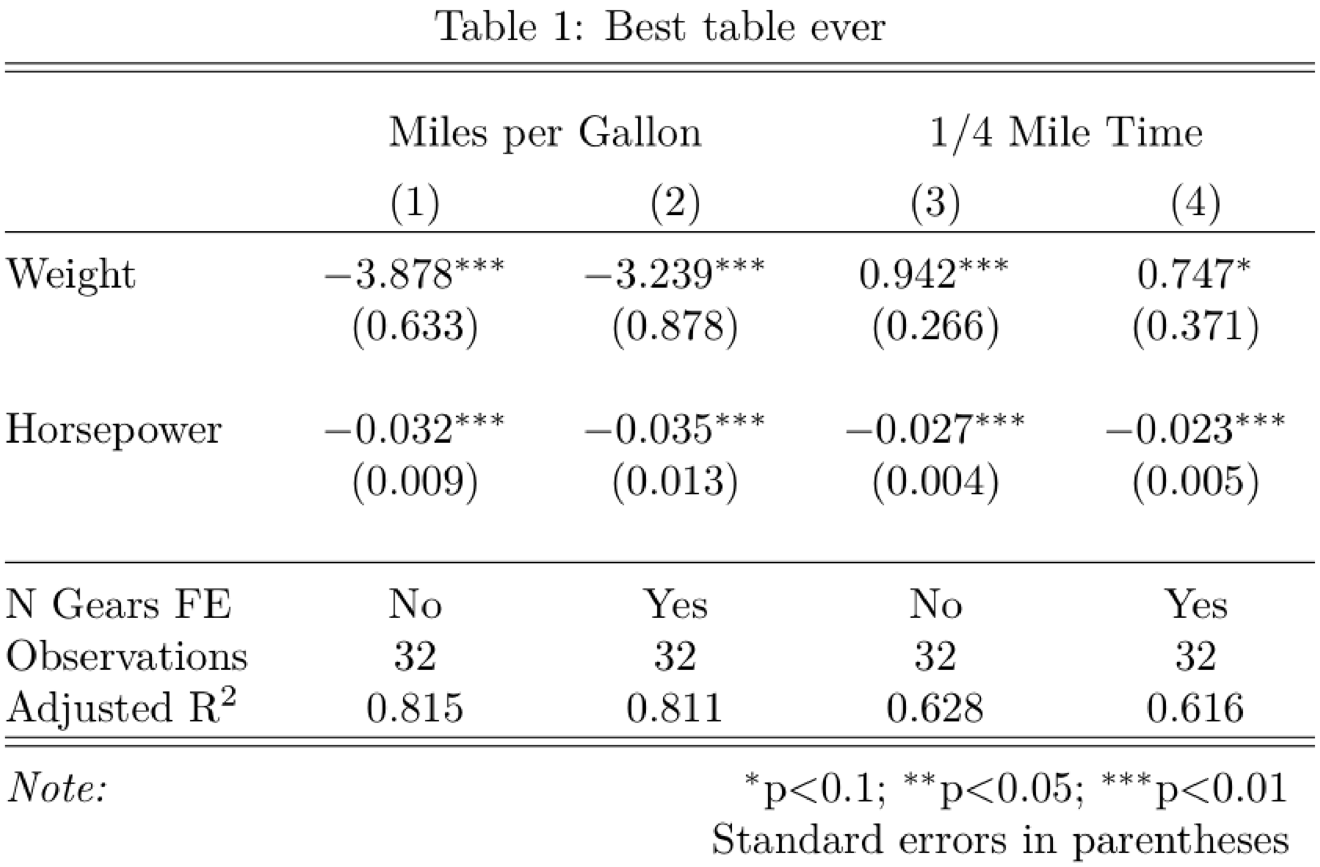


Table 1: Best table ever

	Miles per Gallon		1/4 Mile Time	
	(1)	(2)	(3)	(4)
Weight	-3.878^{***}	-3.239^{***}	0.942^{***}	0.747^{*}
	(0.633)	(0.878)	(0.266)	(0.371)
Horsepower	-0.032^{***}	-0.035^{***}	-0.027^{***}	-0.023^{***}
	(0.009)	(0.013)	(0.004)	(0.005)
N Gears FE	No	Yes	No	Yes
Observations	32	32	32	32
Adjusted R^2	0.815	0.811	0.628	0.616

Note: *p<0.1; **p<0.05; ***p<0.01
Standard errors in parentheses

Annex - Looking ahead

- Markdown guide
- R Markdown: The Definitive Guide
- An introduction to Stata Markdown
- Stargazer official manual
- Introduction to Huxtable