

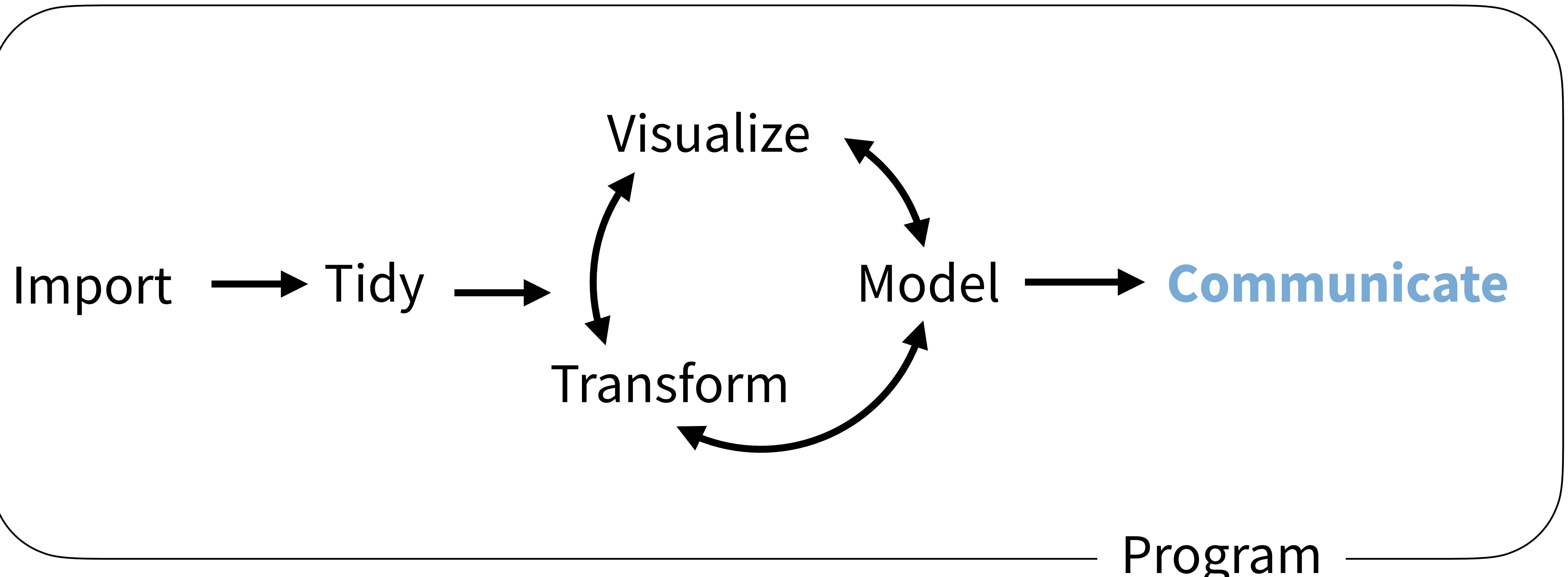
Report Reproducibly with



Navigate to the main page of the class: **<https://astamm.github.io/data-science-with-r/>**.

Download **05-Report-Exercises.qmd** and **05-Report-Parameters.qmd** from the outline table and open them.

(Applied) Data Science



Quarto



Quarto

Plain text file with 3 types of content:

The screenshot shows the Quarto interface with a plain text file open. The file contains three distinct types of content:

- A YAML header surrounded by ---**: Lines 1 through 8.
- Code chunks surrounded by '```'**: Lines 10 through 27.
- Text in markdown**: Lines 29 through 41.

```
1 ---  
2 title: "Models"  
3 format:  
4   html:  
5     eval: true  
6     code-tools:  
7       source: repo  
8 ---  
9  
10 ````{r setup}  
11 #| include: false  
12 library(tidyverse)  
13 library(modelr)  
14 library(broom)  
15  
16 # Import wages here  
17 library(readxl)  
18 wages <- read_excel("wages.xlsx", na = "NA")  
19  
20 # Fall back in case you cannot load wages  
21 # wages <- heights |>  
22 #   filter(income > 0) |>  
23 #   mutate(  
24 #     marital = as.character(marital),  
25 #     sex = as.character(sex)  
26 #   )  
27 ````  
28  
29 ## Your Turn 1  
30  
31 * Change the working directory to the folder where `wages.xlsx` is located and  
32 this file is saved.  
33 * Then import `wages.xlsx` as `wages` and *copy the code to your setup chunk*.  
34 * Be sure to set `NA` to `NA`.  
35 * Switch the `eval` option in the YAML header to `true`.  
36  
37 ## Your Turn 2  
38  
39 - Fit the model  
40  
41 $$
```

At the bottom of the interface, it says "Copilot: No completions available." and "Quarto".

A YAML header
surrounded by

Code chunks
surrounded by
'` '

Text in
markdown

How it works

A large, semi-transparent graphic consisting of two overlapping circles. The top circle contains the letter 'R' and the bottom circle contains the letter 'A', both in a bold, white, sans-serif font.

RA

knitr



pandoc

7

HTML



ioslides
slidy, Beamer



Powerpoint



Microsoft Word

quarto®

Logistics

1

Knitr runs the document in a fresh R session,
which means you need to load the libraries
that the document uses *in the document*

Logistics

1

Knitr runs the document in a fresh R session, which means you need to load the libraries that the document uses *in the document*

2

Objects made in one code chunk will be available to code in later code chunks.

KNITR IS MULTILINGUAL!

 **SAS**

 **PYTHON**

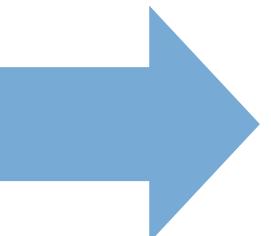
 **MORE**

engine

python

```
Some python code,  
```{python}  
x = 'hello, python
world!'
print(x)
print(x.split(' '))
```
```

To embed non R code, change the chunk label from r to the language to use.



Some python code:

```
x = 'hello, python world!'  
print(x)  
print(x.split(' '))
```

```
## hello, python world!  
## ['hello,', 'python', 'world!']
```

Reticulate

Combining R and Python with {reticulate} and Quarto

Sometimes you might need to use R. Sometimes you might need to use Python. Sometimes you need to use both at the same time. This blog post shows you how to combine R and Python code using {reticulate} and output the results using Quarto.

JANUARY 6, 2023

The R versus Python debate has been going on for as long as both languages have existed. I'm not one to takes sides - I think you need to use the best tool for the job. Sometimes R will be better. Sometimes Python will be better. But what happens if you need both languages in the same workflow? Do you need to choose? No, is the simple answer. You can use both. This blog post will show you how you can combine R and Python code in the same analysis using {reticulate} and output the results using Quarto.

What is Quarto? ↗

Quarto is an open-source scientific and technical publishing system that lets you combine narrative text with code to create reproducible and elegantly formatted output. If you're familiar with R Markdown, Quarto might sound somewhat similar - you can think of Quarto as next generation of R Markdown. The great thing about Quarto is that it doesn't just support code written in R - it supports other languages, including Python!

Quarto documents have two main sections: (i) the YAML header, where we specify document-wide properties such as the output format, and (ii) the content, which can include text, images, code, and more. For the example in this blog post, I'm going to output the code and results from the analysis to revealjs slides so my YAML header looks like this:

The reticulate package lets you use Python and R together seamlessly in R code, in R Markdown documents, and in the RStudio IDE.

Python in R Markdown

(Optional) Build Python env to use.
Add knitr-knit_engines\$set_python =
reticulate::eng_python to the setup
chunk to set up the reticulate Python
engine (not required for knitr >= 1.18).
Suggest the Python environment
to use, in your setup chunk.
Begin Python chunks with `{{python}}`.
Chunk options like echo, include, etc. all
work as expected.
Use the `py` object to access objects created
in Python chunks from R chunks.
Python chunks all execute within a
single Python session so you have access
to all objects created in previous chunks.
Use the `r` object to access objects created
in R chunks from Python chunks.
Output displays below chunk,
including matplotlib plots.

Python in R code

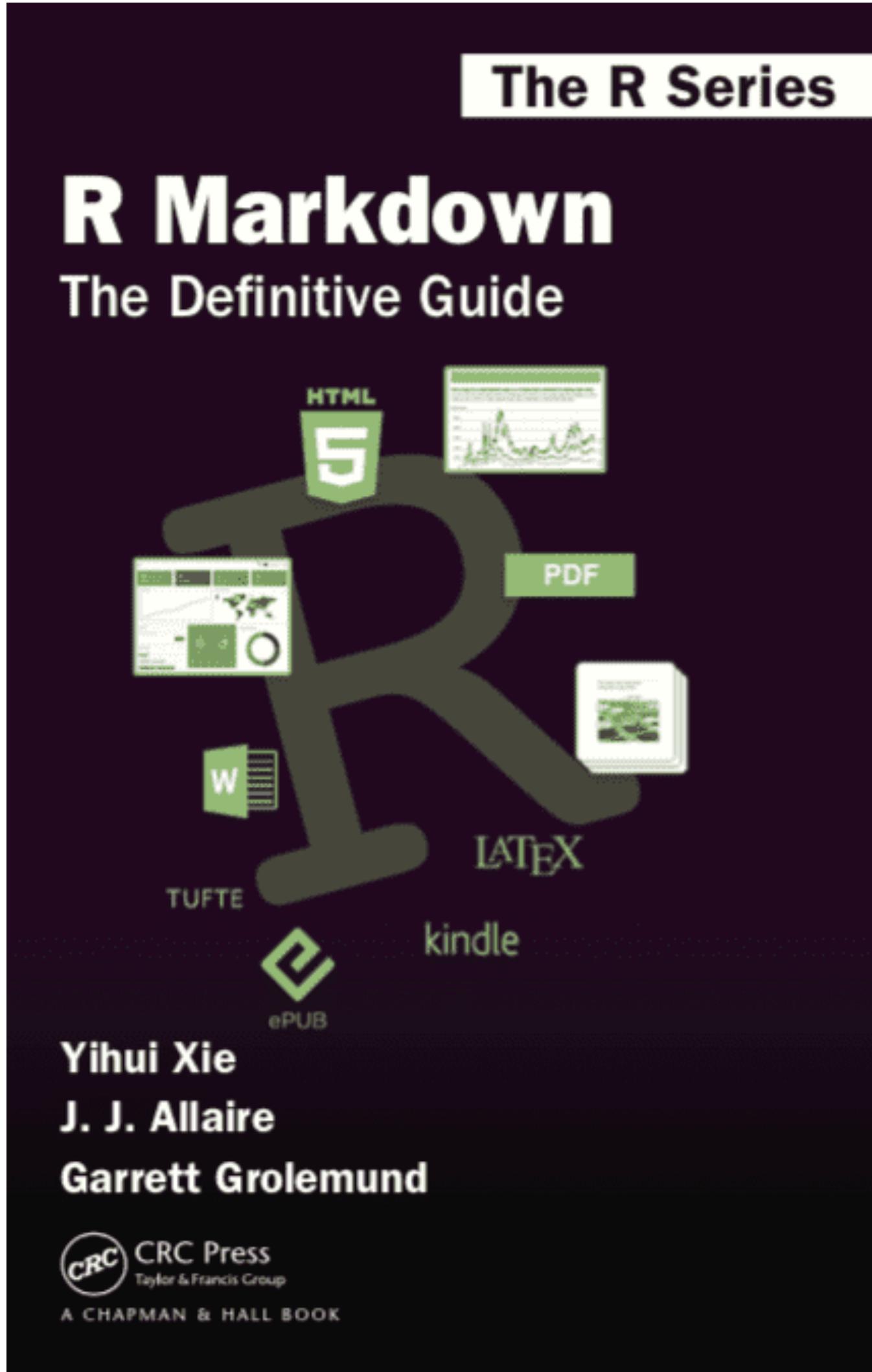
Call Python from R in three ways:
• `import`(module, as=NULL, convert =
TRUE, delay_load = FALSE) Import a
Python module. If convert = TRUE,
Python objects are converted to their
equivalent R types. Also
`import_from_path`(import_name)
• `import_main`(convert = TRUE)
Import the main module, where Python
executes code by default. `import_main()`
• `import_builtin`(convert = TRUE)
Import Python's built-in functions.
`import_builtin()`

SOURCE PYTHON FILES

Use `source_python` to source a Python script
and make the Python functions and objects it
creates available in the calling R environment.
• `source_python`(file, envr = NULL, convert =
TRUE) Run a Python script,
assigning objects to a specified R
environment. `source_python("my.py")`

RUN PYTHON CODE

Execute Python code into the `main` Python
module with `py_run_file` or `py_run_string`.
• `py_run_string`(code, local = FALSE, convert =
TRUE) Run Python code
(passed as a string) in the main
module. `py_run_string("x = 10", py$x)`
• `py_run_file`(file, local = FALSE, convert =
TRUE) Run a Python file in the
main module. `py_run_file("my.py")`
Access the results, and anything else in Python's
main module, with `py`.
• `py`(obj) Convert an R object
to a Python expression. Also `py$call`,
`py$eval`, `py$eval_code`, `py$eval_file`,
`py$eval_string`, `py$get`, `py$get_attr`,
`py$get_error`, `py$get_file`, `py$get_string`,
`py$get_value`, `py$iter`, `py$iter_attr`,
`py$iter_file`, `py$iter_string`, `py$load`,
`py$load_file`, `py$load_string`, `py$next`,
`py$read`, `py$read_file`, `py$read_string`,
`py$set`, `py$set_attr`, `py$set_error`,
`py$set_file`, `py$set_string`, `py$unwind`,
`py$write`, `py$write_file`, `py$write_string`



Guide

- Authoring
- Computations
- Tools
- Documents
- Presentations
- Dashboards
- Websites
- Books
- Manuscripts
- Interactivity
- Publishing
- Projects
- Advanced

Guide

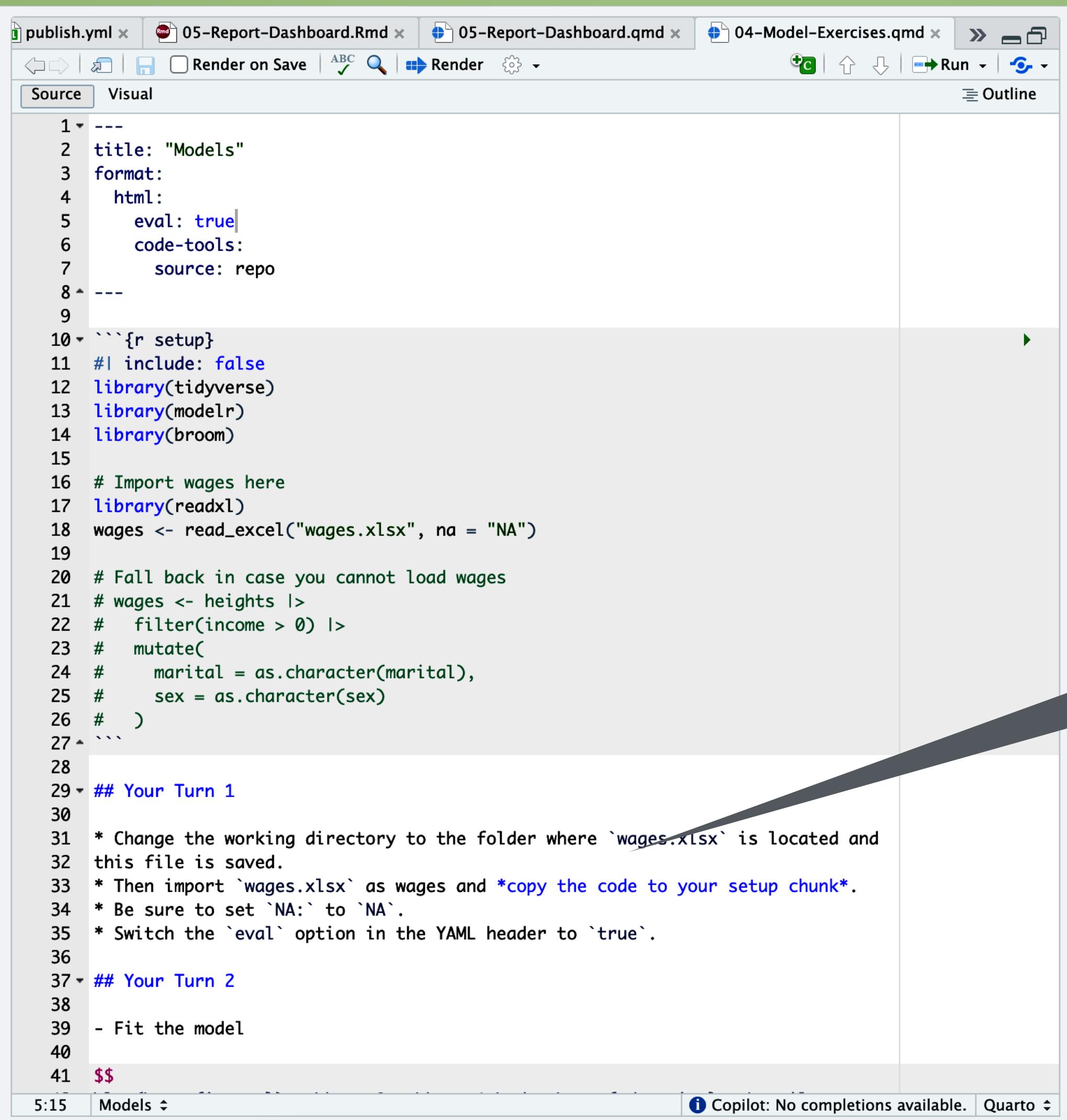
Comprehensive guide to using Quarto. If you are just starting out, you may want to explore the [tutorials](#) to learn the basics.

| Authoring | Computations | Tools | Documents |
|------------------------------------|-------------------------------------|-------------------------------------|---------------------------------|
| Create content with markdown | Execute code and display its output | Use your favorite tools with Quarto | Generate output in many formats |
| Markdown Basics | Using Python | JupyterLab | HTML |
| Figures | Using R | RStudio IDE | PDF |
| Tables | Using Julia | VS Code | MS Word |
| Diagrams | Using Observable | Neovim | Typst |
| Citations | Execution Options | Text Editors | Markdown |
| Cross References | Parameters | Visual Editor | All Formats |
| Article Layout | | | |
| Shortcodes | | | |
| Presentations | Dashboards | Websites | Books |
| Present code and technical content | Publish data with dashboards | Create websites and blogs | Create books and manuscripts |
| Presentation Basics | Dashboard Basics | Creating a Website | Creating a Book |
| Revealjs (HTML) | Layout | Website Navigation | Book Structure |
| PowerPoint (Office) | Data Display | Creating a Blog | Book Crossrefs |
| Beamer (PDF) | Interactivity | Website Search | Customizing Output |
| | Deployment | Website Listings | |
| Manuscripts | Interactivity | Publishing | Projects |
| Write and publish | Engage readers with | Publishing documents | Scale up your work with |

bookdown.org/yihui/rmarkdown/

ONLINE, FREE

Markdown



The screenshot shows a Quarto editor interface with several open files in the top bar: `publish.yml`, `05-Report-Dashboard.Rmd`, `05-Report-Dashboard.qmd`, and `04-Model-Exercises.qmd`. The main area displays R code:

```
1 ---  
2 title: "Models"  
3 format:  
4   html:  
5     eval: true  
6   code-tools:  
7     source: repo  
8 ---  
9  
10 ````{r setup}  
11 #| include: false  
12 library(tidyverse)  
13 library(modelr)  
14 library(broom)  
15  
16 # Import wages here  
17 library(readxl)  
18 wages <- read_excel("wages.xlsx", na = "NA")  
19  
20 # Fall back in case you cannot load wages  
21 # wages <- heights |>  
22 #   filter(income > 0) |>  
23 #   mutate(  
24 #     marital = as.character(marital),  
25 #     sex = as.character(sex)  
26 #   )  
27 ````  
28  
29 ## Your Turn 1  
30  
31 * Change the working directory to the folder where `wages.xlsx` is located and  
32 this file is saved.  
33 * Then import `wages.xlsx` as wages and *copy the code to your setup chunk*.  
34 * Be sure to set NA: ` to NA`.  
35 * Switch the eval option in the YAML header to true.  
36  
37 ## Your Turn 2  
38  
39 - Fit the model  
40  
41 $$
```

The code includes a YAML header with `eval: true` and a `setup` chunk. A callout bubble points from the `eval: true` line to the text **Text in markdown**.

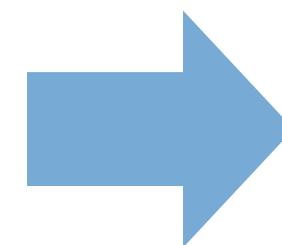
Text in
markdown

Headers

Use # to create headers.

Multiple #'s create lower level headers.

```
# Header 1  
## Header 2  
### Header 3  
#### Header 4  
##### Header 5  
##### Header 6
```



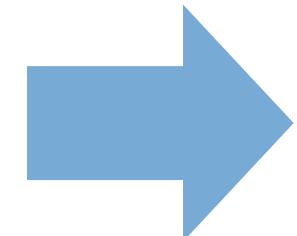
Header 1
Header 2
Header 3
Header 4
Header 5
Header 6

Text

Add two spaces at
the end of a line to
start a new line

Text is rendered as plain text. Surround
text with `_`, `**`, or ``` to format it.

Text
`_italics_`
`**bold**`
``code``



Text
italics
bold
`code`

Lists

Use asterisks to make bullet points.

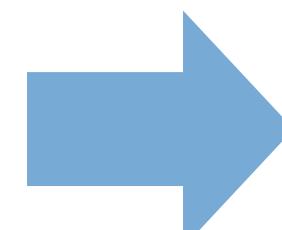
Use numbers to make numbered lists.

Bullets

- * bullet 1
- * bullet 2

Numbered list

1. item 1
2. item 2



Bullets

- bullet 1
- bullet 2

Numbered list

1. item 1
2. item 2

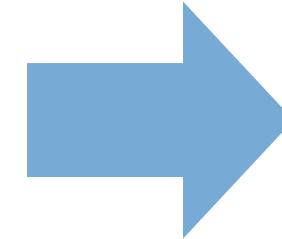
Hyperlinks

Use brackets to denote a link.

Place the URL in parentheses.

This is a
[link](www.git.com).

This is a link.

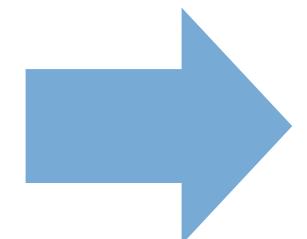


Images

Use a link preceded by an ! to insert an image.

The link text should be a URL (if the image is hosted online), or a file path (if the image is saved as a file)

The RStudio logo.

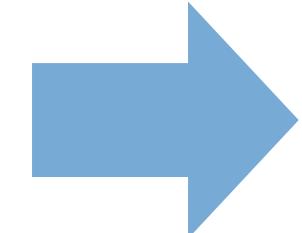


The RStudio logo.

Equations

Write equations with latex math commands and surround them in \$'s.

According to Einstein,
 $E=mc^2$



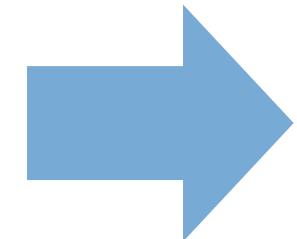
According to Einstein, $E = mc^2$

Equation blocks

Use two \$'s to make
centered equation blocks.

According to
Einstein,

`$$E=mc^{\{2\}}$$`



According to
Einstein,

$$E = mc^2$$

Markdown

Pandoc's Markdown
Write with syntax on the left to create effect on right (after render)

```

Plain text
End a line with two spaces
to start a new paragraph.
*italics* and **bold**
`verbatim code`
sub/superscript22
~~strikethrough~~
escaped: `\\` 
endash: -, emdash: --
equation: $A = \pi r^2$ 
equation block:

$SE = mc^2$$

> block quote

# Header1 [#anchor]

## Header 2 {css_id}

### Header 3 {css_class}

#### Header 4

##### Header 5

##### Header 6

<--Text comment-->

\textbf{Text ignored in HTML}
<em>HTML ignored in pdfs</em>

<a href="http://www.rstudio.com">
  [link]([www.rstudio.com])
  Jump to [Header 1](#anchor)
  image:
  !{Caption}(smallorb.png)
  * unordered list
    + sub-item 1
    + sub-item 2
    - sub-sub-item 1
  * item 2
    Continued (indent 4 spaces)
  1. ordered list
  2. item 2
    i. sub-item 1
    A. sub-sub-item 1
  (@) A list whose numbering
  continues after
  2. an interruption
  (@) an interruption
  Term 1
  Definition 1
  Right Left Default Center
  12 12 12 12
  123 123 123 123
  1 1 1 1
  - slide bullet 1
  - slide bullet 2
  (>- to have bullets appear on click)
  horizontal rule/slide break:
  ***
  A footnote [^1]
  [^1]: Here is the footnote.
  1. Here is the footnote.

```

Pandoc's Markdown
Write with syntax on the left to create effect on right (after render)

```

Plain text
End a line with two spaces
to start a new paragraph.
*italics* and **bold**
`verbatim code`
sub/superscript22
~~strikethrough~~
escaped: `\\` 
endash: -, emdash: --
equation: $A = \pi r^2$ 
equation block:

$SE = mc^2$$

> block quote

# Header1 [#anchor]

## Header 2 {css_id}

### Header 3 {css_class}

#### Header 4

##### Header 5

##### Header 6

<--Text comment-->

\textbf{Text ignored in HTML}
<em>HTML ignored in pdfs</em>

<a href="http://www.rstudio.com">
  [link]([www.rstudio.com])
  Jump to [Header 1](#anchor)
  image:
  !{Caption}(smallorb.png)
  * unordered list
    + sub-item 1
    + sub-item 2
    - sub-sub-item 1
  * item 2
    Continued (indent 4 spaces)
  1. ordered list
  2. item 2
    i. sub-item 1
    A. sub-sub-item 1
  (@) A list whose numbering
  continues after
  2. an interruption
  (@) an interruption
  Term 1
  Definition 1
  Right Left Default Center
  12 12 12 12
  123 123 123 123
  1 1 1 1
  - slide bullet 1
  - slide bullet 2
  (>- to have bullets appear on click)
  horizontal rule/slide break:
  ***
  A footnote [^1]
  [^1]: Here is the footnote.
  1. Here is the footnote.

```

Dictionary of formatting cues.

Set render options with YAML

When you render R Markdown

1. runs the R code, embeds results and text into .md file with knitr
2. then converts the .md file into the finished format with pandoc

Set a document's default output format in the YAML header:

```

output: html_document
---
```

sub-option **description**

| sub-option | description |
|-----------------------|--|
| citation_package | The LaTeX package to process citations, natbib, biblatex or none |
| code_folding | Let readers to toggle the display of R code, "none", "hide", or "show" |
| colortheme | Beamer color theme to use |
| css | CSS file to use to style document |
| dev | Graphics device to use for figure output (e.g. "png") |
| duration | Add a countdown timer (in minutes) to footer of slides |
| fig_caption | Should figures be rendered with captions? |
| fig_height, fig_width | Default figure height and width (in inches) for document |
| highlight | Syntax highlighting: "tango", "pygments", "kate", "zenburn", "textmate" |
| includes | File of content to place in document (in _header, before_body, after_body) |
| incremental | Should bullets appear one at a time (on presenter mouse clicks)? |
| keep_md | Save a copy of .md file that contains knitr output |
| keep_tex | Save a copy of .tex file that contains knitr output |
| latex_engine | Engine to render latex, "pdflatex", "xelatex", or "luatex" |
| lib_dir | Directory of dependency files to use (Bootstrap, MathJax, etc.) |
| mathjax | Set to local or a URL to a local/URL version of MathJax to render equations |
| md_extensions | Markdown extensions to add to default definition of R Markdown |
| number_sections | Add section numbering to headers |
| pandoc_args | Additional arguments to pass to Pandoc |
| preserve_yaml | Preserve YAML front matter in final document? |
| reference_docx | docx file whose styles should be copied when producing docx output |
| self_contained | Embed dependencies into the doc |
| slide_level | The lowest heading level that defines individual slides |
| smaller | Use the smaller font size in the presentation? |
| smart | Convert straight quotes to curly, dashes to em-dashes, ... to ellipses, etc. |
| template | Pandoc template to use when rendering file quarterly_report.html). |
| theme | Beamer or Bootstrap theme to use for page |
| toc | Add a table of contents at start of document |
| toc_depth | The lowest level of headings to add to table of contents |
| toc_float | Float the table of contents to the left of the main content |

Customize output with sub-options (listed to the right):

```

... output: html_document:
  code_folding: hide
  toc_float: TRUE
  ...
  # Body

```

html tabs

Use tabset css class to place sub-headers into tabs

```

# Tabset .tabset.tabset-fade.tabset-pills{
  # Tab 1
  text 1
  # Tab 2
  text 2
  ...
  # End tabset
  text 1
  End tabset

```

Create a Reusable Template

1. Create a new package with a inst/markdown/templates directory
2. In the directory, Place a folder that contains: template.yaml (see below) Template files (contents of the template) any supporting files
3. Install the package
4. Access template in wizard at File ► New File ► R Markdown template.yaml

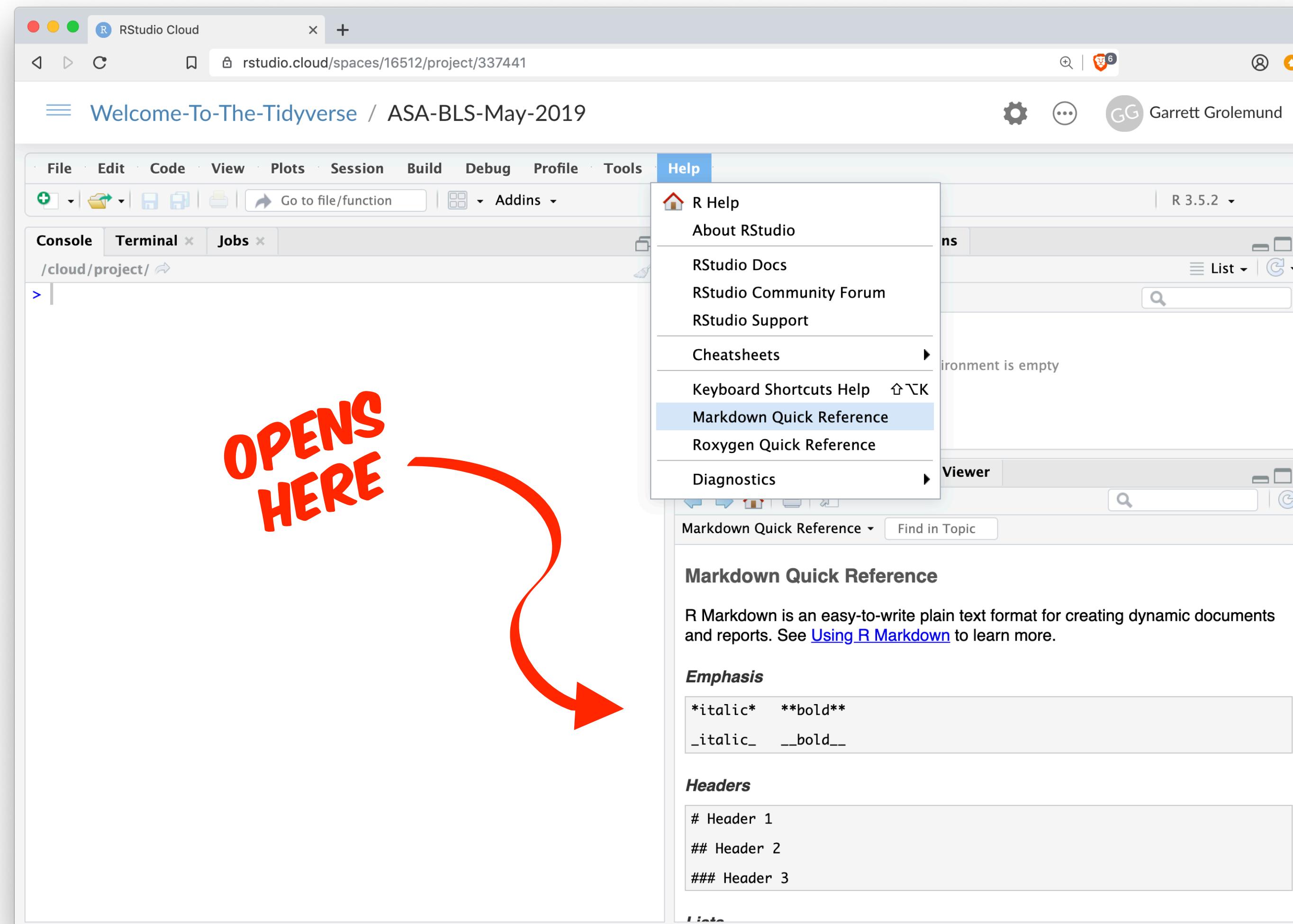
RStudio® is a trademark of RStudio, Inc. • CC BY SA RStudio • info@rstudio.com • 844-448-1212 • rstudio.com • Learn more at rmarkdown.rstudio.com • rmarkdown 1.6 • Updated: 2016-02

ON BACK OF
RMARKDOWN
CHEAT SHEET

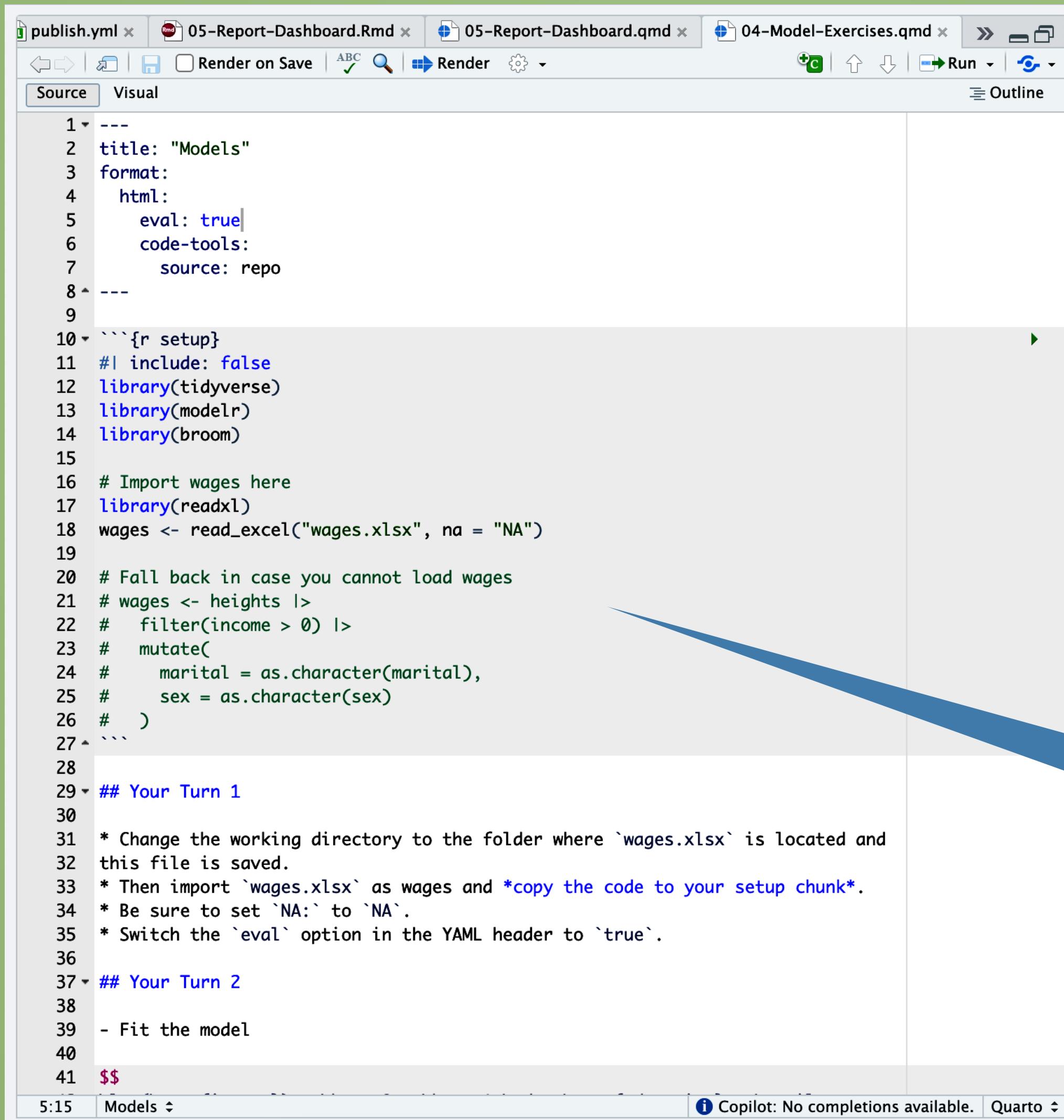


IDE Reference

Go to Help > Markdown Quick Reference



Code



The screenshot shows a Quarto editor interface with multiple files open in tabs at the top: publish.yml, 05-Report-Dashboard.Rmd, 05-Report-Dashboard.qmd, and 04-Model-Exercises.qmd. The main area displays an R script. A blue arrow points from the text "Code chunks surrounded by" in the callout box below to the opening brace of the first code chunk on line 10.

```
1 ---  
2 title: "Models"  
3 format:  
4   html:  
5     eval: true  
6     code-tools:  
7       source: repo  
8 ---  
9  
10 ```{r setup}  
11 #| include: false  
12 library(tidyverse)  
13 library(modelr)  
14 library(broom)  
15  
16 # Import wages here  
17 library(readxl)  
18 wages <- read_excel("wages.xlsx", na = "NA")  
19  
20 # Fall back in case you cannot load wages  
21 # wages <- heights |>  
22 #   filter(income > 0) |>  
23 #   mutate(  
24 #     marital = as.character(marital),  
25 #     sex = as.character(sex)  
26 #   )  
27 ```  
28  
29 ## Your Turn 1  
30  
31 * Change the working directory to the folder where `wages.xlsx` is located and  
32 this file is saved.  
33 * Then import `wages.xlsx` as wages and *copy the code to your setup chunk*.  
34 * Be sure to set `NA:` to `NA`.  
35 * Switch the `eval` option in the YAML header to `true`.  
36  
37 ## Your Turn 2  
38  
39 - Fit the model  
40  
41 $$
```

Code chunks
surrounded by
` `` `

Code chunks

Insert a chunk of R code with

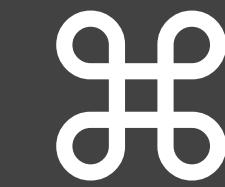
```
```{r}
some code
```
```

When you render the report, R Markdown will run the code and include its results. R Markdown will also remove the ```{r} and ```.

Code chunks

Insert a chunk of R code with

```
```{r}  
some code
```
```



+

Opt

+

A white lowercase letter 'i' inside a dark rounded rectangle.

(Mac)

A white lowercase letter 'c' inside a dark rounded rectangle.

+

A white lowercase letter 'a' inside a dark rounded rectangle.

+

A white lowercase letter 'i' inside a dark rounded rectangle.

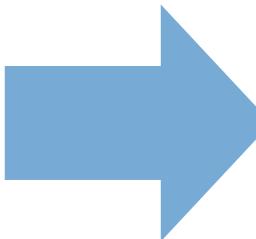
(PC)

chunk options

By default, R Markdown includes both the code and its results

Here's some code

```
```{r}  
dim(iris)
```
```



Here's some code

```
dim(iris)
```

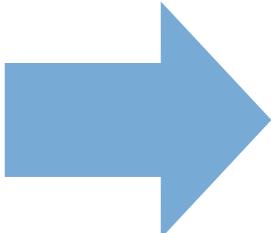
```
## [1] 150 5
```

echo

Add options in the chunk by starting the line with `#|`.

echo: false hides the code.

```
Here's some code  
```{r}  
#| echo: false
dim(iris)
```
```



```
Here's some code  
## [1] 150 5
```

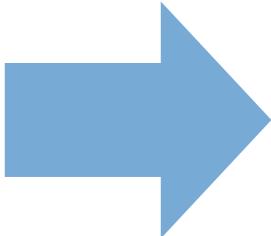
Very useful
for plots



eval

eval: false prevents the code from being run. As a result, no results will be displayed with the code.

```
Here's some code  
```{r}  
#| eval: false
dim(iris)
```
```

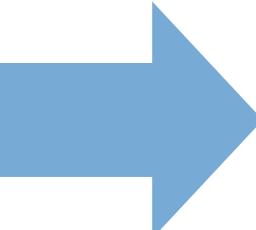


```
Here's some code  
dim(iris)
```

include

include: false runs the code, but prevents both the code and the results from appearing (e.g. to setup).

```
Here's some code  
```{r}  
#| include: false
dim(iris)
```
```

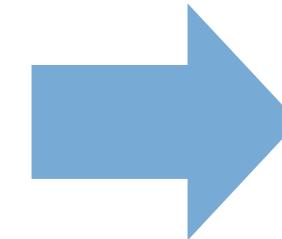


```
Here's some code
```

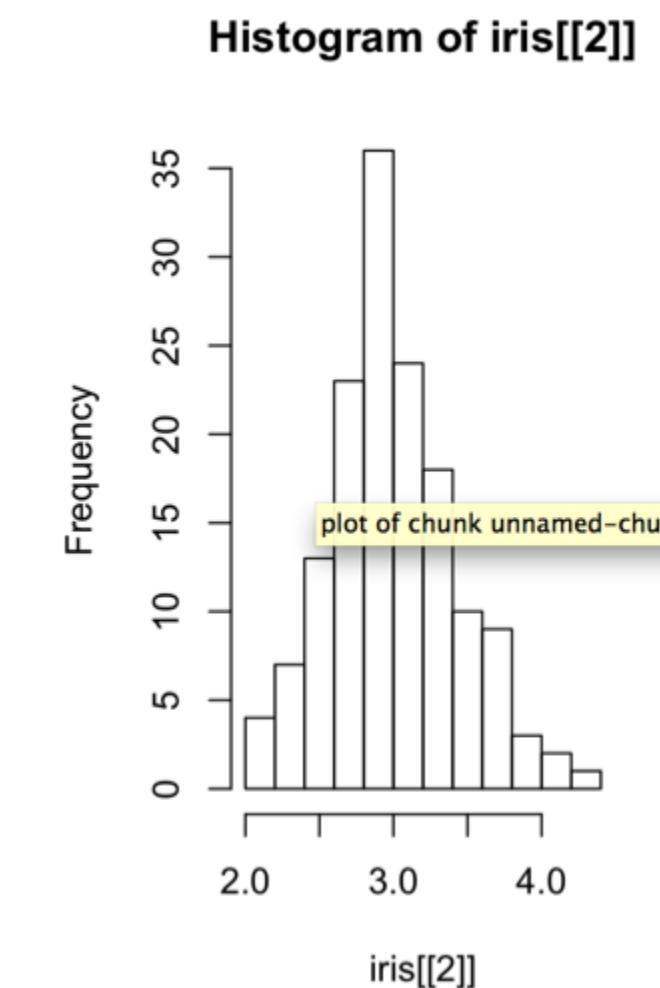
fig-height, fig-width

Specify the dimension of plots (in inches) with `fig-width` and `fig-height`. Separate multiple arguments with commas.

```
Here's a plot  
```{r}  
#| echo: false
#| fig-width: 3
#| fig-height: 5
hist(iris[[2]])
```
```



Here's a plot



Pop Quiz

Do you notice the TODOs in
05-RMarkdown-Exercises.qmd?

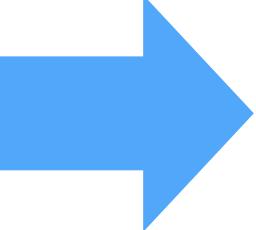
Do you notice the long setup chunk?

Inline code

Place code in a sentence with `r <code>`.

Quarto will replace the code with its results.

Today is
`r Sys.Date()`.



Today is 2015-04-16.

Inline code

Code whose results are inserted into the text.

Today is `r Sys.Date()`.

Surround
with `r`

Code to run. Only the
result will be included.

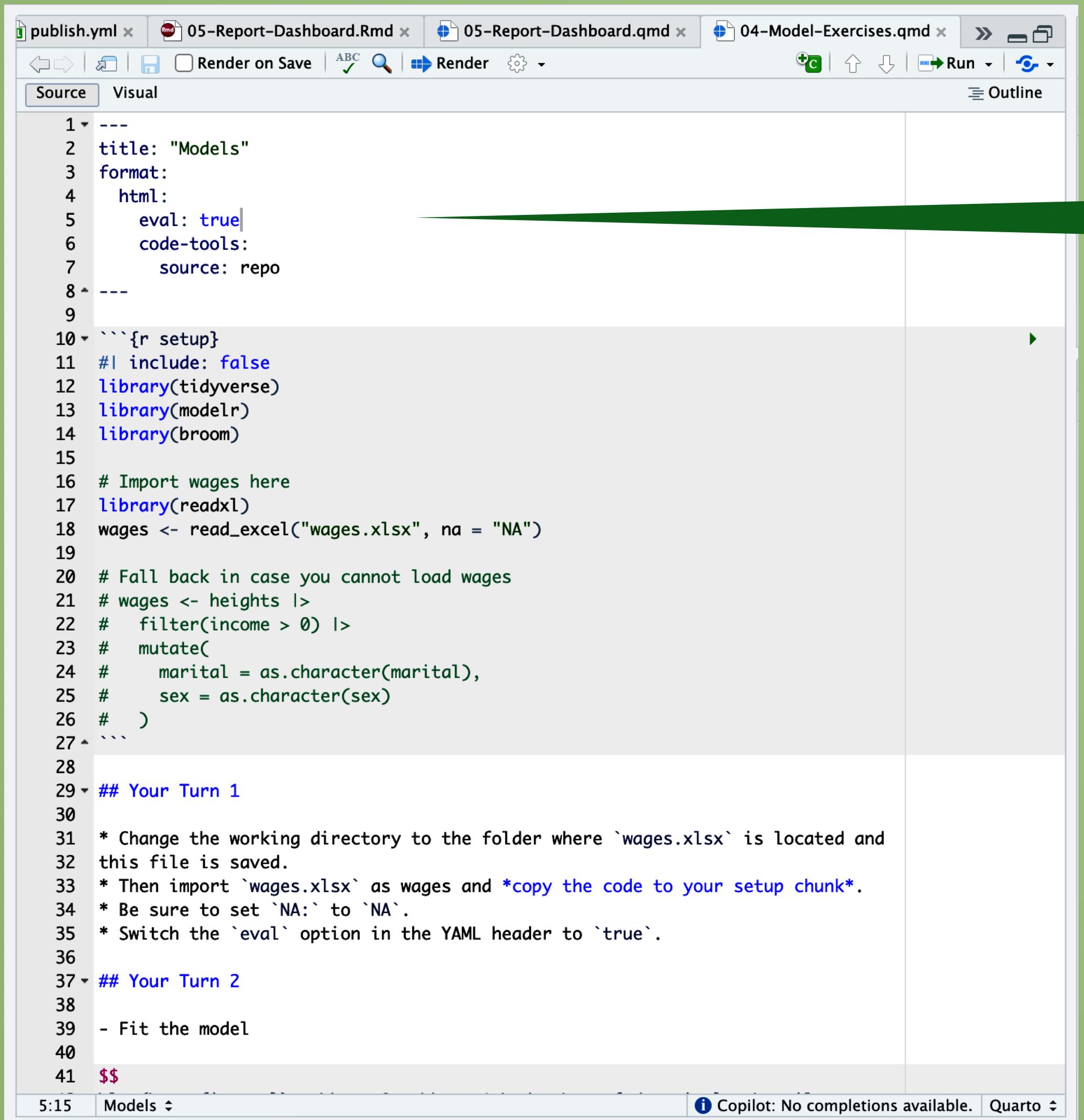
Your Turn 1

In 05-RMarkdown-Exercises.qmd:

1. Replace every Garrett with your name
2. Replace every TODO with inline R code
3. Check that the setup chunk is not included with the output
4. Ensure that only the output of the plot chunk is shown (not the code)
5. Knit the document



YAML



A screenshot of a Quarto editor interface. The main window shows a code block with a YAML header at the top. The YAML header consists of three dashes followed by the key-value pairs: title: "Models", format: html, eval: true, code-tools: source: repo, and another three dashes. Below the header is a setup chunk containing R code to import tidyverse, modelr, and broom packages, and to read a wages.xlsx file. The code block also includes two sections labeled ## Your Turn 1 and ## Your Turn 2, each with instructions. The status bar at the bottom indicates it's page 5:15, the document is titled Models, and there are no completions available.

```
1 ---  
2 title: "Models"  
3 format:  
4   html:  
5     eval: true  
6     code-tools:  
7       source: repo  
8 ---  
9  
10 ````{r setup}  
11 #| include: false  
12 library(tidyverse)  
13 library(modelr)  
14 library(broom)  
15  
16 # Import wages here  
17 library(readxl)  
18 wages <- read_excel("wages.xlsx", na = "NA")  
19  
20 # Fall back in case you cannot load wages  
21 # wages <- heights |>  
22 #   filter(income > 0) |>  
23 #   mutate(  
24 #     marital = as.character(marital),  
25 #     sex = as.character(sex)  
26 #   )  
27 ````  
28  
29 ## Your Turn 1  
30  
31 * Change the working directory to the folder where `wages.xlsx` is located and  
32 this file is saved.  
33 * Then import `wages.xlsx` as wages and *copy the code to your setup chunk*.  
34 * Be sure to set `NA:` to `NA`.  
35 * Switch the `eval` option in the YAML header to `true`.  
36  
37 ## Your Turn 2  
38  
39 - Fit the model  
40  
41 $$
```

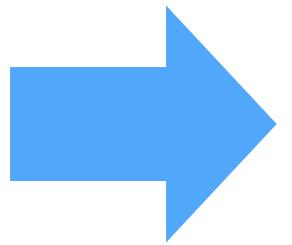
A YAML header
surrounded by
— — —

YAML

A section of key:value pairs
separated by dashed lines ----

```
---  
title: "Untitled"  
author: "RStudio"  
date: "February 4, 2015"  
format: html  
---
```

Text of document



Untitled

RStudio

February 4, 2015

Text of document

format

The format: field sets the format of the final report

The screenshot shows the Quarto website's navigation bar at the top, featuring the Quarto logo, a search bar, and social media links. A prominent banner at the top of the main content area announces "Quarto 1.6 released! Download | Read More". The main content is titled "All Formats" and includes an "Overview" section. It discusses Pandoc's support for various output formats and provides examples of YAML and command-line usage. On the right side, there are "On this page" navigation links and buttons for "Edit this page" and "Report an issue". At the bottom, there's a link to a list of all output formats.

Overview > Documents > All Formats

All Formats

Overview

Pandoc supports a huge array of output formats, all of which can be used with Quarto. To use any Pandoc format just use the `format` option or the `--to` command line option.

For example, here's some YAML that specifies the use of the `html` format as well as a couple of format options:

```
---
```

```
title: "My Document"
format:
  html:
    toc: true
    code-fold: true
---
```

Alternatively you can specify the use of a format on the command line:

```
Terminal
```

```
quarto render document.qmd --to html
```

See below for a list of all output formats by type along with links to their reference documentation.

Parameters

A faint watermark of the R logo is visible in the bottom right corner, consisting of a circular arrow and the letters "R".

R

Your Turn 2

Open 05-RMarkdown-Parameters.qmd.

Render the document.

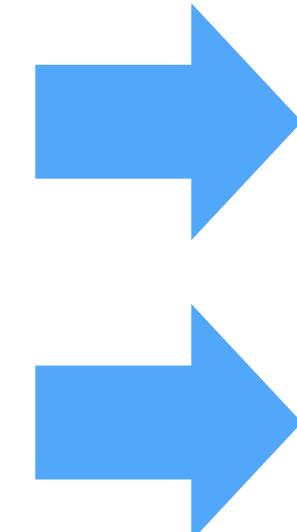
Change the name property in the YAML header and re-render document. What happened?



Parameters

A list of values that you can call in R code chunks

params list
**elements and
values**



```
---
```

```
title: "Untitled"
```

```
format: html
```

```
params:
```

```
  filename: "data.csv"
```

```
  symbol: "FB"
```

```
---
```

colon

New line.
Indented two
spaces

Using Parameters

Call parameter values as elements of the params list, **params\$num**

```
---
```

```
params:
```

```
  num: 42
```

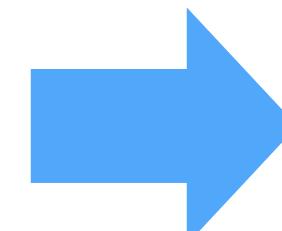
```
---
```

```
The value of the parameter is `r params$num`, e.g.
```

```
```{r}
```

```
params$num
```

```
```
```



The value of the parameter is 42, e.g.

```
params$num
```

```
## [1] 42
```

quarto::quarto_render() + for

```
names <- c("Alice", "Bob", "Cathy")

for (name in names) {
  quarto::quarto_render("05-RMarkdown-
Parameters.Rmd",
  execute_params = list(data = name))
}
```



Demo

Report Reproducibly with

