week5

Hakan Mehmetcik

Communicate

So far, you've learned the tools to get your data into R, tidy it into a form convenient for analysis, and then understand your data through transformation, and visualization. However, it doesn't matter how great your analysis is unless you can explain it to others: you need to **communicate** your results.

Quarto

Introduction

Quarto provides a unified authoring framework for data science, combining your code, its results, and your prose. Quarto documents are fully reproducible and support dozens of output formats, like PDFs, Word files, presentations, and more.

Quarto files are designed to be used in three ways:

1. For communicating to decision-makers, who want to focus on the conclusions, not the code behind the analysis.
2. For collaborating with other data scientists (including future you!), who are interested in both your conclusions, and how you reached them (i.e. the code).
3. As an environment in which to *do* data science, as a modern-day lab notebook where you can capture not only what you did, but also what you were thinking.

> **Note**
>
> Please refer to the Quarto documentation for further details.

Any Quarto documents contain three important types of content:

1. An (optional) **YAML header** surrounded by `---`s.
2. **Chunks** of R code surrounded by ```.
3. Text mixed with simple text formatting like `# heading` and `_italics_`.

Markdown options

Text formatting

italic **bold** ~~strikeout~~ ‘code‘

superscript^2^ subscript~2~

[underline]{.underline} [small caps]{.smallcaps}

Headings

1st Level Header

2nd Level Header

3rd Level Header

Lists

- Bulleted list item 1
- Item 2
- Item 2a
- Item 2b

1. Numbered list item 1
2. Item 2. The numbers are incremented automatically in the output.

Links and images

<http://example.com>

[linked phrase](http://example.com)

![optional caption text](quarto.png){fig-alt=“Quarto logo and the word quarto spelled in small case letters”}

Tables

| First Header | Second Header | |————-|————-| | Content Cell | Content Cell | | Content Cell | Content Cell |

Code Chunks

To run code inside a Quarto document, you need to insert a chunk. There are three ways to do so:

1. The keyboard shortcut Cmd + Option + I / Ctrl + Alt + I.
2. The "Insert" button icon in the editor toolbar.
3. By manually typing the chunk delimiters ```` ```{r} ```` and ```` ``` ````.

I'd recommend you learn the keyboard shortcut. It will save you a lot of time in the long run!

Chunk options

Chunk output can be customized with **options**, fields supplied to chunk header. Knitr provides almost 60 options that you can use to customize your code chunks. Here we'll cover the most important chunk options that you'll use frequently. You can see the full list at https://yihui.org/knitr/options.

The most important set of options controls if your code block is executed and what results are inserted in the finished report:

- `eval: false` prevents code from being evaluated. (And obviously if the code is not run, no results will be generated). This is useful for displaying example code, or for disabling a large block of code without commenting each line.
- `include: false` runs the code, but doesn't show the code or results in the final document. Use this for setup code that you don't want cluttering your report.
- `echo: false` prevents code, but not the results from appearing in the finished file. Use this when writing reports aimed at people who don't want to see the underlying R code.
- `message: false` or `warning: false` prevents messages or warnings from appearing in the finished file.
- `results: hide` hides printed output; `fig-show: hide` hides plots.

- `error: true` causes the render to continue even if code returns an error. This is rarely something you'll want to include in the final version of your report, but can be very useful if you need to debug exactly what is going on inside your `.qmd`. It's also useful if you're teaching R and want to deliberately include an error. The default, `error: false` causes rendering to fail if there is a single error in the document.

Each of these chunk options get added to the header of the chunk, following `#|`, e.g., in the following chunk the result is not printed since `eval` is set to false.

```
2 * 2
```

Global options

As you work more with knitr, you will discover that some of the default chunk options don't fit your needs and you want to change them.

You can do this by adding the preferred options in the document YAML, under `execute`. For example, if you are preparing a report for an audience who does not need to see your code but only your results and narrative, you might set `echo: false` at the document level. That will hide the code by default, so only showing the chunks you deliberately choose to show (with `echo: true`). You might consider setting `message: false` and `warning: false`, but that would make it harder to debug problems because you wouldn't see any messages in the final document.

> **Note**
>
> title: "My report"
> execute:
> echo: false

Inline Code

There is one other way to embed R code into a Quarto document: directly into the text, with: `` `r ` ``. This can be very useful if you mention properties of your data in the text.

Figures

The figures in a Quarto document can be embedded (e.g., a PNG or JPEG file) or generated as a result of a code chunk.

Figures Sizing

The biggest challenge of graphics in Quarto is getting your figures the right size and shape. There are five main options that control figure sizing: `fig-width`, `fig-height`, `fig-asp`, `out-width` and `out-height`. Image sizing is challenging because there are two sizes (the size of the figure created by R and the size at which it is inserted in the output document), and multiple ways of specifying the size (i.e. height, width, and aspect ratio: pick two of three).

We recommend three of the five options:

- Plots tend to be more aesthetically pleasing if they have consistent width. To enforce this, set `fig-width: 6` (6") and `fig-asp: 0.618` (the golden ratio) in the defaults. Then in individual chunks, only adjust `fig-asp`.
- Control the output size with `out-width` and set it to a percentage of the body width of the output document. We suggest to `out-width: "70%"` and `fig-align: center`.

  That gives plots room to breathe, without taking up too much space.
- To put multiple plots in a single row, set the `layout-ncol` to 2 for two plots, 3 for three plots, etc. This effectively sets `out-width` to "50%" for each of your plots if `layout-ncol` is 2, "33%" if `layout-ncol` is 3, etc. Depending on what you're trying to illustrate (e.g., show data or show plot variations), you might also tweak `fig-width`.

> **Note**
>
> Figure sizing and scaling is an art and science and getting things right can require an iterative trial-and-error approach. You can learn more about figure sizing in the taking control of plot scaling blog post.

Tables

Similar to figures, you can include two types of tables in a Quarto document. They can be markdown tables that you create directly in your Quarto document (using the Insert Table menu) or they can be tables generated as a result of a code chunk.

```
mtcars[1:5, ]
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2

```
knitr::kable(mtcars[1:5, ], )
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2

> **Note**
>
> Read the documentation for `?knitr::kable` to see the other ways in which you can customize the table. For even deeper customization, consider the **gt**, **huxtable**, **reactable**, **kableExtra**, **xtable**, **stargazer**, **pander**, **tables**, and **ascii** packages. Each provides a set of tools for returning formatted tables from R code.

Caching

Normally, each render of a document starts from a completely clean slate. This is great for reproducibility, because it ensures that you've captured every important computation in code. However, it can be painful if you have some computations that take a long time. The solution is `cache: true`.

Bibliographies and Citations

Quarto can automatically generate citations and a bibliography in a number of styles. The most straightforward way of adding citations and bibliographies to a Quarto document is using the visual editor in RStudio.

To add a citation using the visual editor, go to Insert > Citation. Citations can be inserted from a variety of sources:

1. DOI (Document Object Identifier) references.

2. Zotero personal or group libraries.
3. Searches of Crossref, DataCite, or PubMed.
4. Your document bibliography (a `.bib` file in the directory of your document)

Under the hood, the visual mode uses the standard Pandoc markdown representation for citations (e.g., `[@citation]`).

If you add a citation using one of the first three methods, the visual editor will automatically create a `bibliography.bib` file for you and add the reference to it. It will also add a `bibliography` field to the document YAML. As you add more references, this file will get populated with their citations. You can also directly edit this file using many common bibliography formats including BibLaTeX, BibTeX, EndNote, Medline.

> **Note**
>
> bibliography: rmarkdown.bib
> csl: apa.csl