Reproducible science: Module6-2

R Markdown with details

Gbadamassi G.O. Dossa

Xishuangbanna Tropical Botanical Garden, XTBG-CAS

2021/10/1 (updated: 2021-10-21)

Part one

Part 1. Introducing R Markdown

R Markdown Document — Create from within RStudio

- Create a new R Markdown document from the RStudio menu:*

  ```
  File -> New File -> R Markdown -> OK
  ```

- Save your new document:**

  ```
  File -> Save
  ```

- Observe that
  - the document has been saved to your working directory, and
  - it has the .Rmd extension

R Markdown Document — Components

Observe also that the document has three components

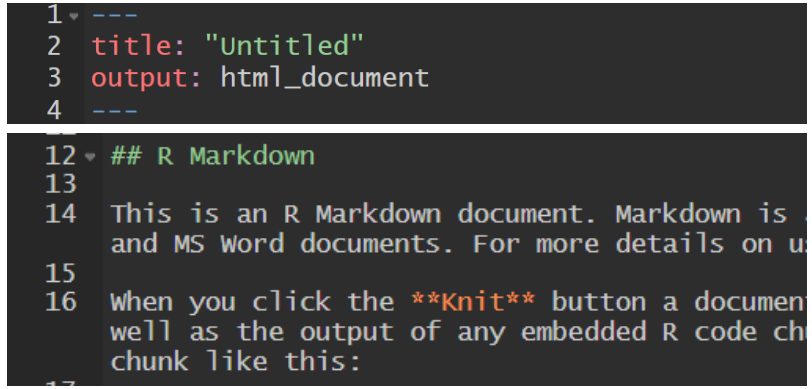
- YAML

```
---
title: "Untitled"
output: html_document
---
```

R Markdown Document — Components

Observe also that the document has three components

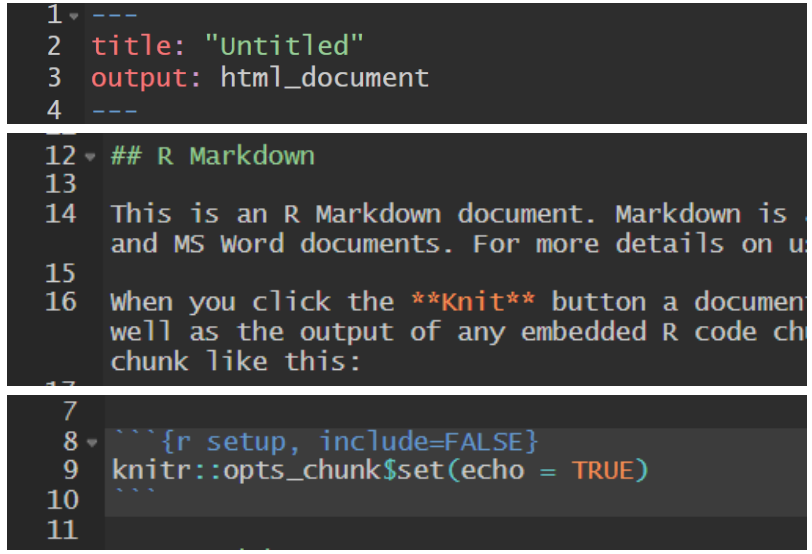
- YAML
- text



R Markdown Document — Components

Observe also that the document has three components

- YAML
- text
- code chunks



R Markdown Document — Document Toolbar

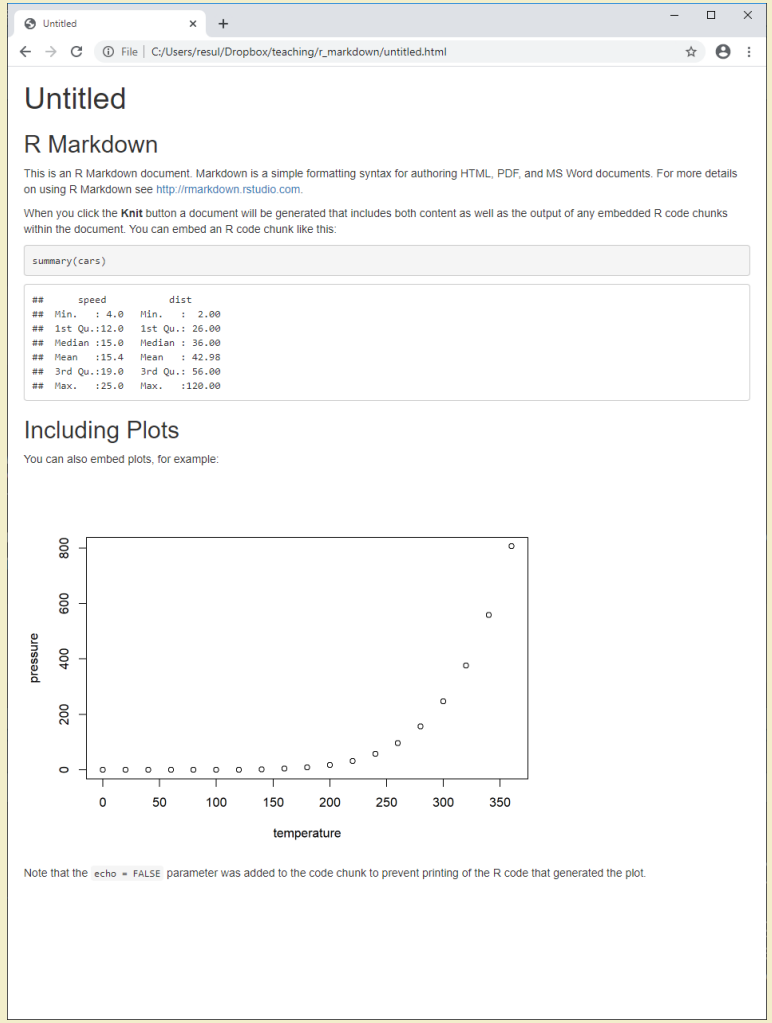
Observe also that the document toolbar offers extended tools for .Rmd documents

These include, most impotantly,

- the  button to compile .Rmd documents

R Markdown Document — Compile

- Click the `Knit` button to compile your .Rmd document, and observe that
  - the output document has the same name as your .Rmd document



R Markdown Document — Compilation Process

- When you `Knit`, the following happens:

  ```
  .Rmd --knitr--> .md --pandoc--> output
  ```

  - `knitr`* executes the code if there is any, converts the resulting document from .Rmd (R Markdown) into .md (Markdown)
  - `pandoc`** transforms the .md document into your preferred output format(s)
    - e.g., HTML, LaTeX, PDF, Word
- This process is automated by the `rmarkdown` package

* If you had not already have the `knitr` package, it would have been installed together with the `rmarkdown` package.

** RStudio comes with a copy of `pandoc` (http://pandoc.org), which is not an R package, so that you do not have to install it separately.

R Markdown Document — Notes

- Behind the scenes, each .Rmd file is compiled in its own session, and therefore
  - the code needs to stand alone, for reproducibility reasons
  - e.g., if you load a package in the Console, it will not be available to a given .Rmd file — even in the same R session
- R Markdown can produce more than documents,* including
  - presentations, again with `rmarkdown`
  - books, with `bookdown` (Xie, 2021c)
  - websites, with `blogdown` (Xie et al., 2021b)

Part 2. Setting Metadata

YAML — Overview

.Rmd documents start* with YAML

- includes the metadata variables
  - e.g., title, output format
- written between a pair of three hyphens -

```
---
title:
output:
---
```

* Technically, we can place YAML anywhere in a .Rmd document. However, it is a good practice to start with YAML so that the metadata is easily accessbile.

YAML — Variables

- `title` and `output` are the basic variables of YAML
  - variable names are typed in lower case, followed by a colon `:`
  - the list of available variables, as well as options and sub-options for these variables, depends on the output format
    - Pandoc User's Guide provides a comprehensive documentation
    - R Markdown Cheat Sheet provides a helpful list
- Typical YAML variables for an research paper are as follows:

```
---
title:
author:
date:
bibliography:
csl:
output:
---
```

YAML — Variables

Variables can take strings

```
---
title: "Journals: Random Words With Random Data" #<<
output:
---
```

YAML — Variables

Variables can take strings, options

```
---
title: "Journals: Random Words With Random Data"
output: pdf_document                                    #<<
---
```

YAML — Variables

Variables can take strings, options, sub-options

```
---
title: "Journals: Random Words With Random Data"
output:
    pdf_document:
        keep_tex: true                    #<<
---
```

YAML — Variables

Variables can take strings, options, sub-options, and code

```
---
title: "Journals: Random Words With Random Data"
output:
    pdf_document:
        keep_tex: true
date: "`r format(Sys.Date(), '%d %B %Y')`"      #<<

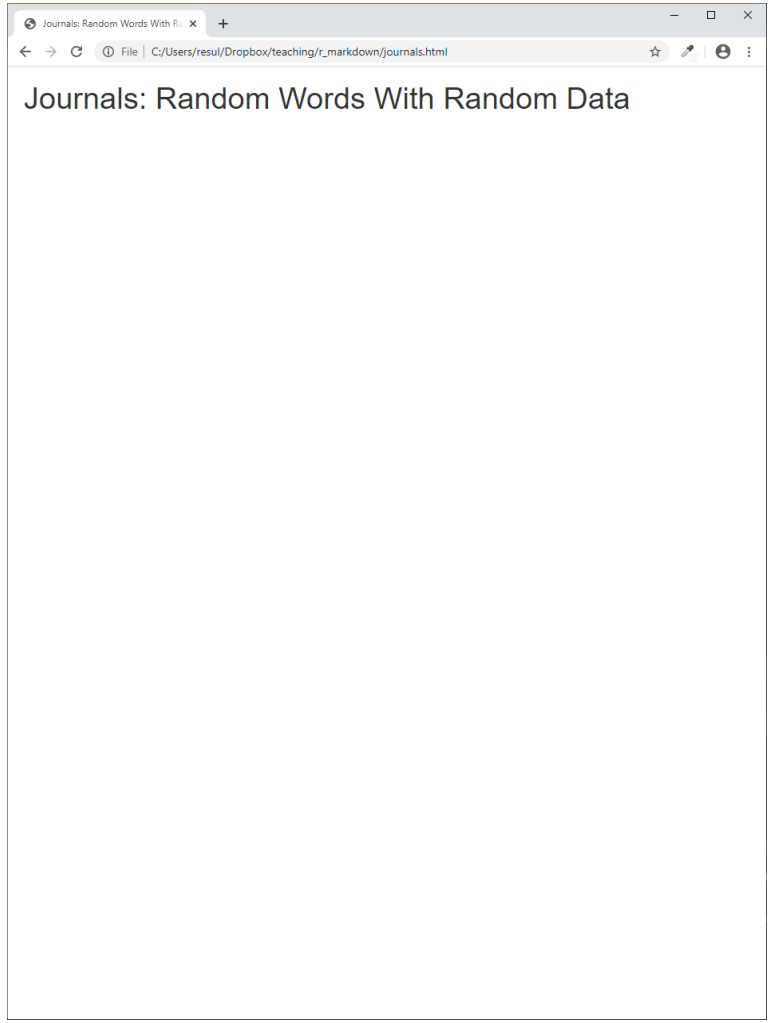
---
```

YAML — Variables — Output Formats

Documents as output formats include

- HTML

```
---
title: "Journals: Random Words With Random Data"
output: html_document #<<
---
```

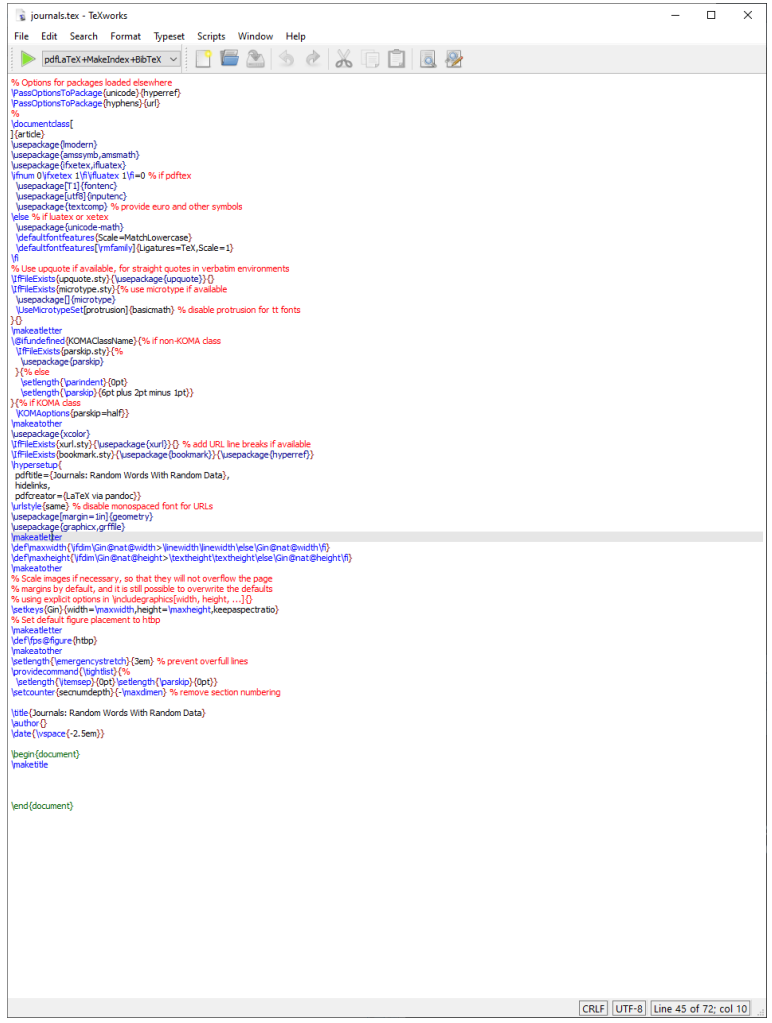


YAML — Variables — Output Formats

Documents as output formats include

- HTML
- LaTeX

```
---
title: "Journals: Random Words With Random Data"
output: latex_document #<<
---
```

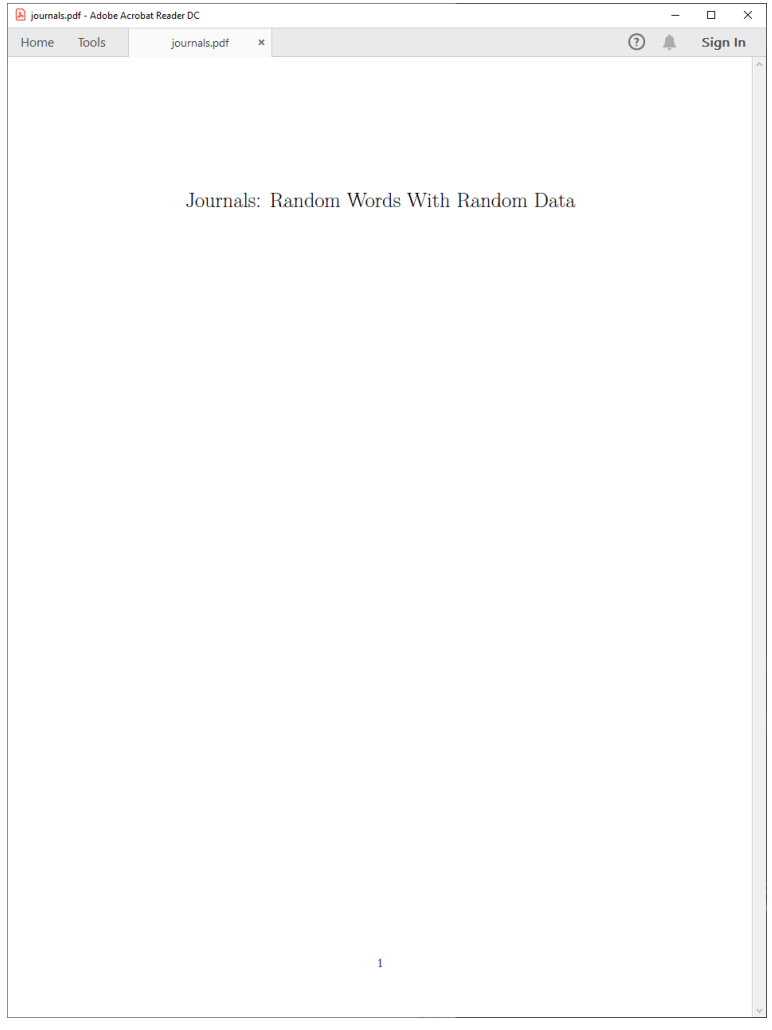


YAML — Variables — Output Formats

Documents as output formats include

- HTML
- LaTeX
- PDF

```
---
title: "Journals: Random Words With Random Da
output: pdf_document #<<
---
```

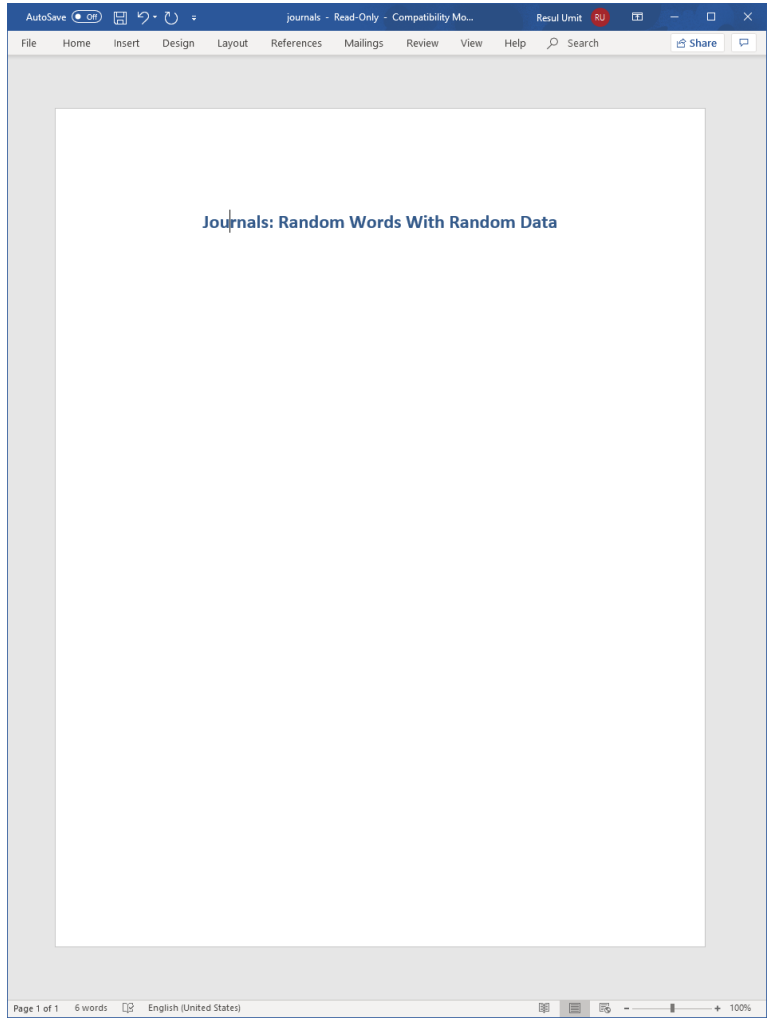


YAML — Variables — Output Formats

Documents as output formats include

- HTML
- LaTeX
- PDF
- Word

```
---
title: "Journals: Random Words With Random Data"
output: word_document #<<
---
```



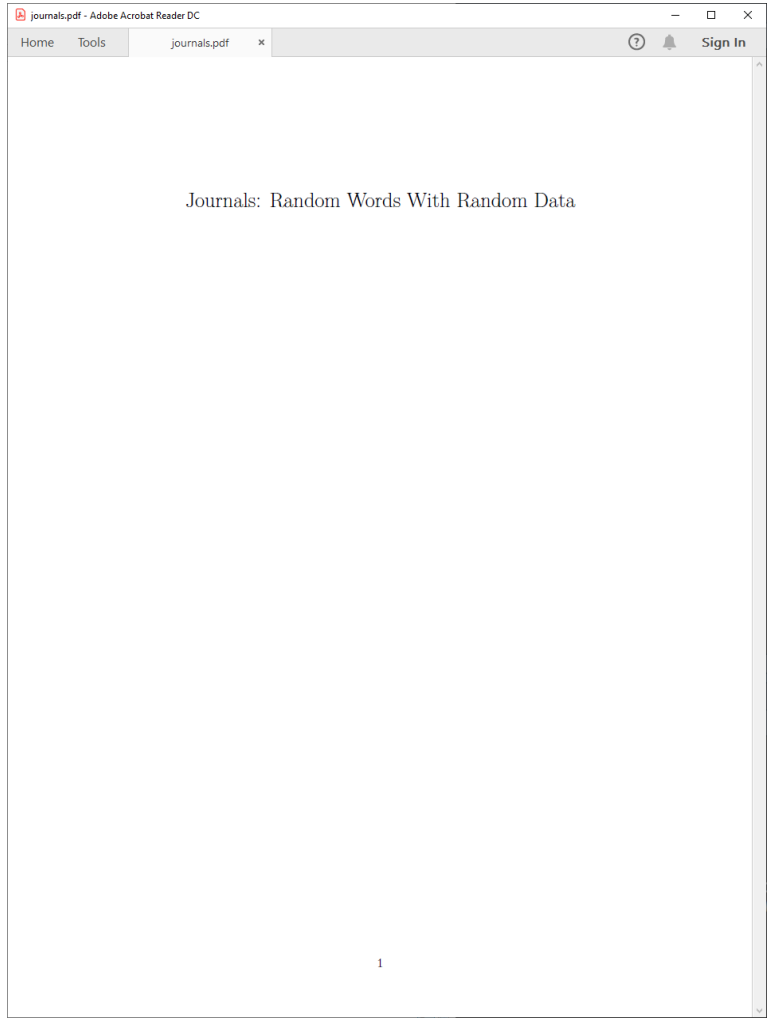
YAML — Variables — Output Formats

- Documents as output formats
  - `html_document`
  - `latex_document`
  - `pdf_document`*
  - `word_document`
  - `github_document`
  - `md_document`
  - `odt_document`
  - `rtf_document`
- Presentations as output formats
  - `beamer_presentation`
  - `iosslides_presentation`
  - `powerpoint_presentation`
  - `slidy_presentation`

YAML — Strings

Strings with special characters, such as colon, require quotation marks — single ' or double "

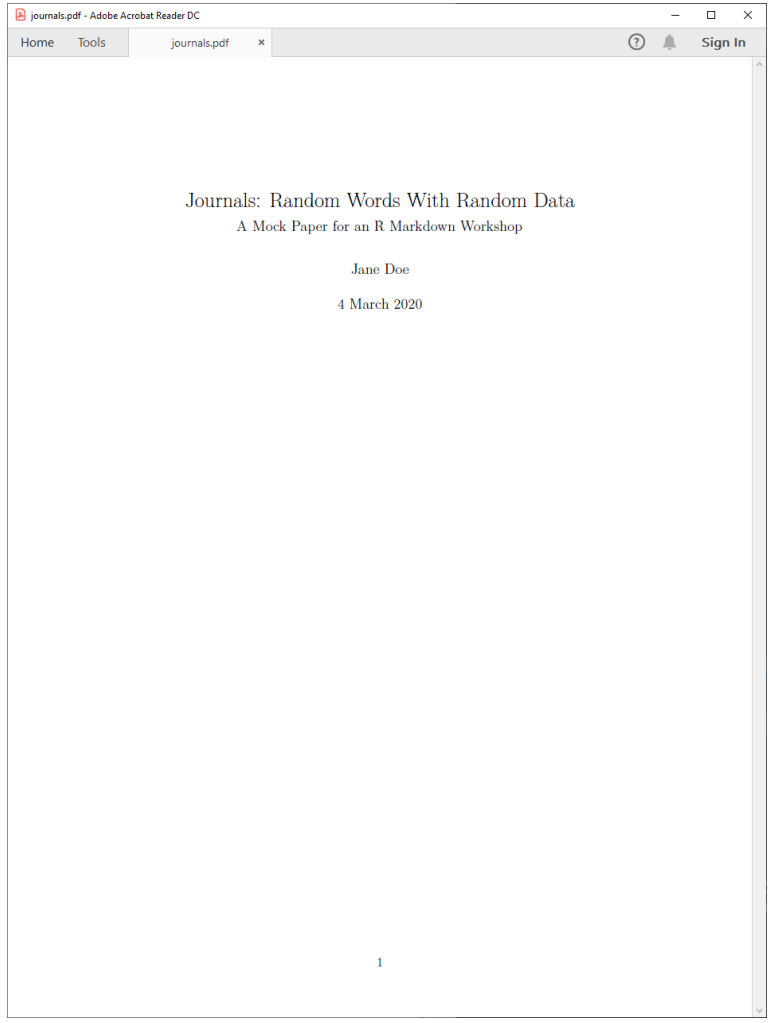
```
---
title: "Journals: Random Words With Random Data" #<<
output: pdf_document
---
```



YAML — Strings

Quotation marks are optional for strings without special characters

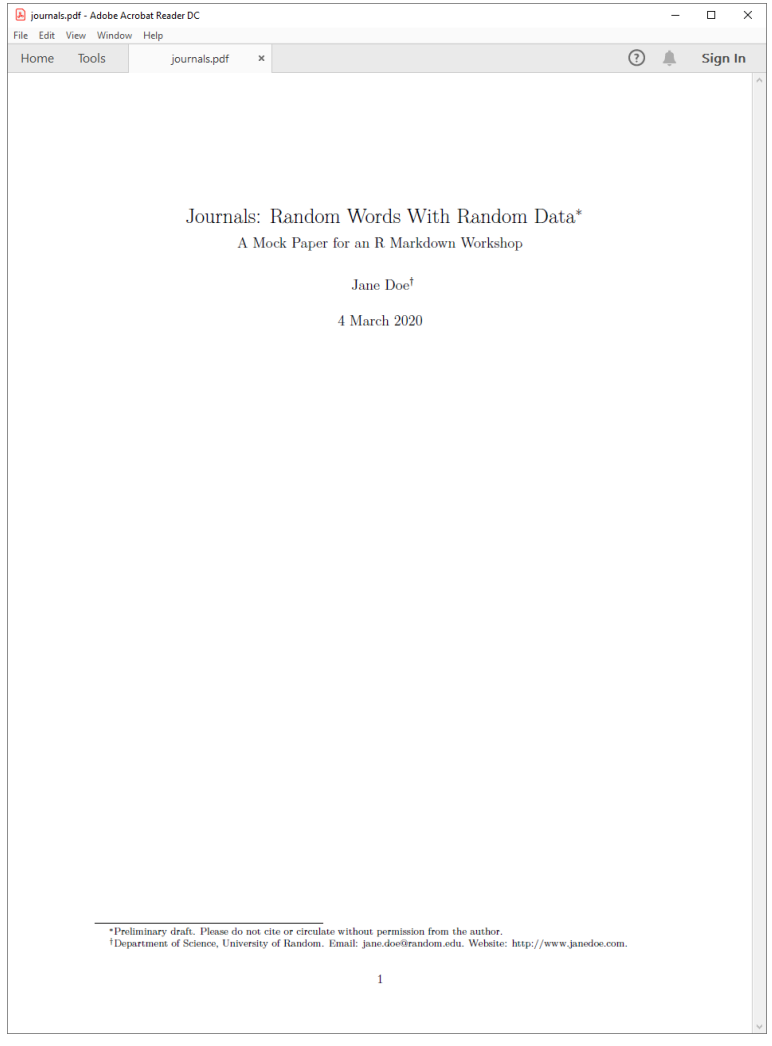
```
---
title: "Journals: Random Words With Random Data"
subtitle: A Mock Paper for an R Markdown Workshop #<<
author: Jane Doe                                  #<<
date: 4 March 2020                                #<<
output: pdf_document
---
```



YAML — Strings — Footnotes

The syntax `^[footnotes_go_here]` adds footnotes to strings

```
:e: "Journals: Random Words With Random Data^[Preliminary
:itle: A Mock Paper for an R Markdown Workshop
:or: "Jane Doe^[Department of Science, University of Rand
:: 4 March 2020
:ut: pdf_document
```



YAML — Strings — External Files

The `bibliography` and `csl` variables take strings as well

```
---
title: "Journals: Random Words With Random Data^[Preliminary draft. Please do not cite or circula
subtitle: A Mock Paper for an R Markdown Workshop
author: "Jane Doe^[Department of Science, University of Random. Email: jane.doe@random.edu. Webs
date: 4 March 2020
bibliography: references.bib   #<<
csl: apa_7th.csl               #<<
output: pdf_document
---
```

YAML — Strings — External Files

The strings for external files indicate (a) where the files are located and (b) how they are named

```
---
...
bibliography: references/ref_library.bib
csl: "../../styles/chicago_manual_17.csl"
...
---
```

YAML — Strings — External Files

The strings for external files indicate (a) where the files are located and (b) how they are named

```
---
...
bibliography: references/ref_library.bib
csl: "../../styles/chicago_manual_17.csl"
...
---
```

Notice that

- the locations above are specified as relative to the working directory
  - the former (references) is a sub-directory, or folder, one level down while the latter (styles) is two levels up
- for reproducibility reasons, hard-coded strings should be avoided
  - e.g., `"C:/Users/resulumit/Dropbox/styles/chicago_manual_17.csl"` called absolute path.

YAML — Strings — External Files

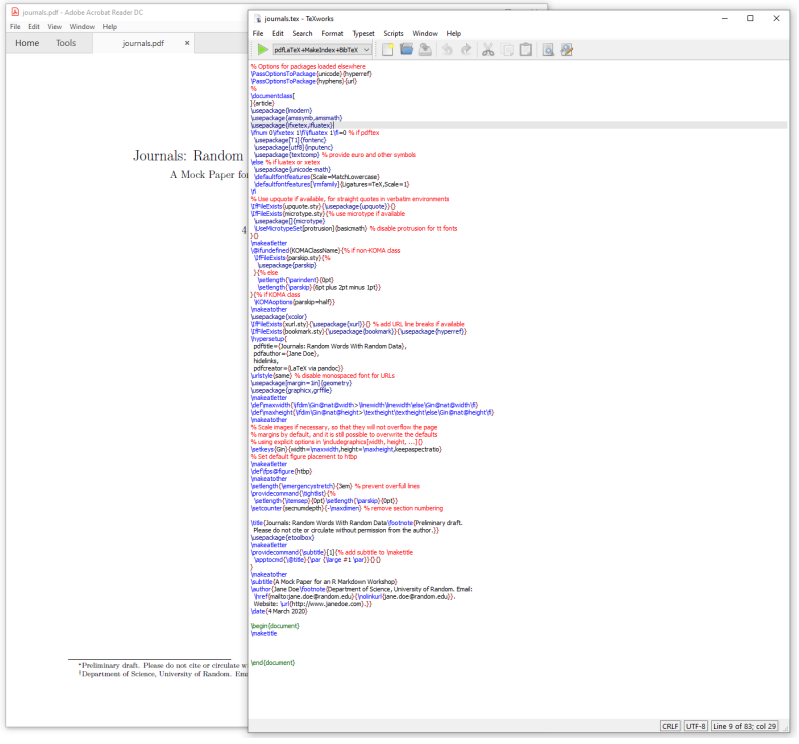
The strings indicate (a) where the files are located and (b) how they are named

```
---
...
bibliography: references/ref_library.bib
csl: "../../styles/chicago_manual_17.csl"
...
---
```

YAML — Options and Sub-Options

Options can have sub-options

```
---
title: "Journals: Random Words With Random Data^[Prelim
subtitle: A Mock Paper for an R Markdown Workshop
author: "Jane Doe^[Department of Science, University of
date: 4 March 2020
bibliography: references.bib
csl: apa_7th.csl
output:                         #<<
    pdf_document:               #<<
        keep_tex: true          #<<
---
```



YAML — Options and Sub-Options

Options can have sub-options

```
---
title: "Journals: Random Words With Random Da
subtitle: A Mock Paper for an R Markdown Work
author: "Jane Doe^[Department of Science, Uni
date: 4 March 2020
bibliography: references.bib
csl: apa_7th.csl
output:                    #<<
    pdf_document:          #<<
        keep_tex: true     #<<
---
```

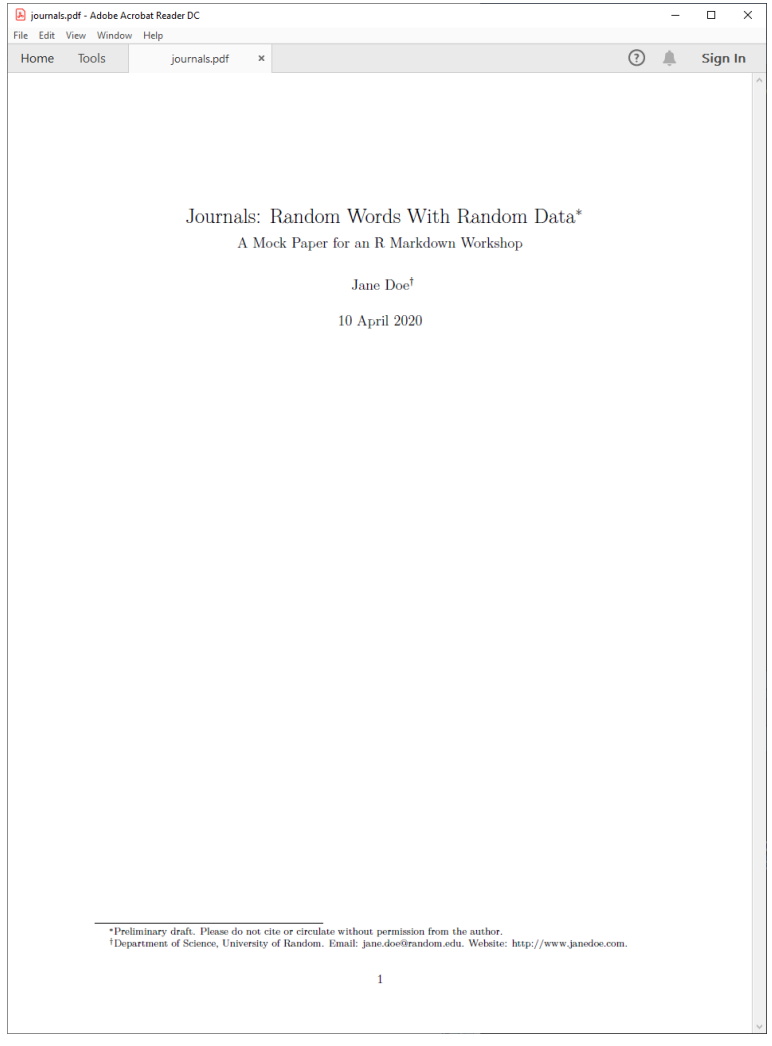
Notice that

- this specific setting, highlighted, will create multiple outputs
    - a LaTeX and a PDF document
- all but the last option (i.e., `true`) takes a colon
- options and sub-options (except the last option, again) are stepwise indented
    - exactly with four spaces
    - the alignment between the colons for `pdf_document` and `keep_tex` is coincidental

YAML — R Code

Variables can take code as well

```
---
title: "Journals: Random Words With Random Data^[Prelim
subtitle: A Mock Paper for an R Markdown Workshop
author: "Jane Doe^[Department of Science, University of
date: "`r format(Sys.Date(), '%d %B %Y')`"  #<<
bibliography: references.bib
csl: apa_7th.csl
output: pdf_document
---
```



YAML — R Code

Variables can take code as well

```
---
title: "Journals: Random Words With Random Da
subtitle: A Mock Paper for an R Markdown Work
author: "Jane Doe^[Department of Science, Uni
date: "`r format(Sys.Date(), '%d %B %Y')`"  #
bibliography: references.bib
csl: apa_7th.csl
output: pdf_document
---
```

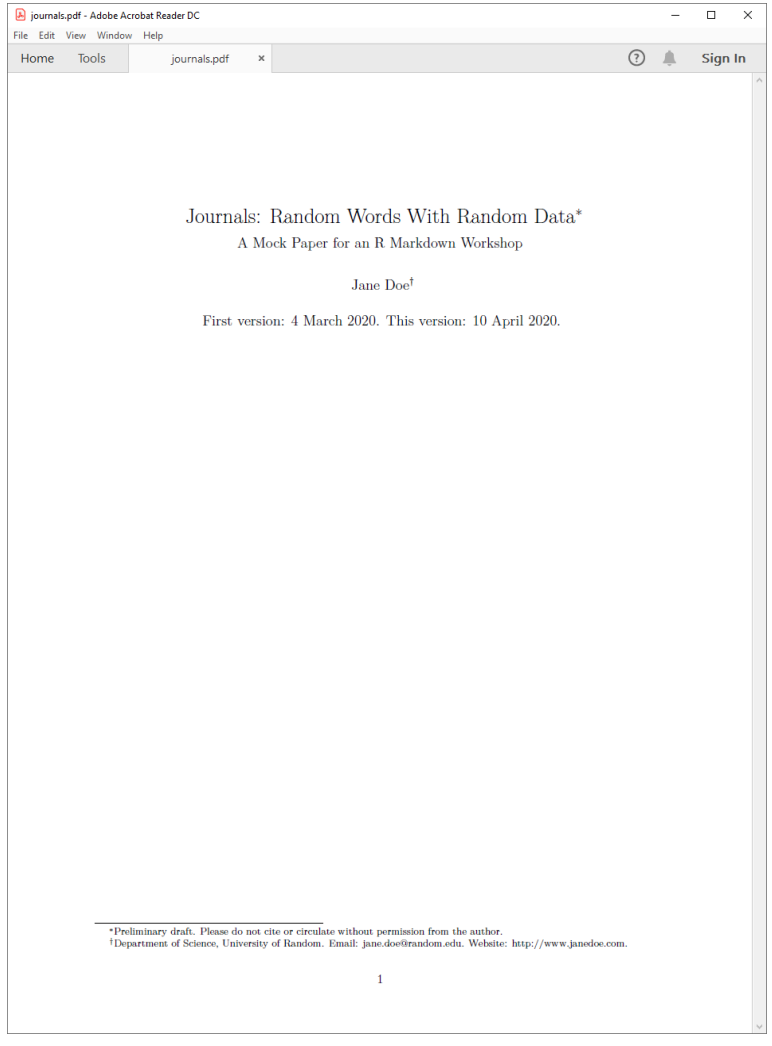
Notice that

- such codes can be particularly useful for variables
  - that need frequent updates
  - and that can be automatically updated
    - e.g., `date`
- there are quotation marks around the code
- we will cover codes in .Rmd documents later on in the workshop

YAML — R Code

Code and text can be combined in a string

```
---
title: "Journals: Random Words With Random Data^[Prelim
subtitle: A Mock Paper for an R Markdown Workshop
author: "Jane Doe^[Department of Science, University of
date: "First version: 4 March 2020. This version: `r for
bibliography: references.bib
csl: apa_7th.csl
output: pdf_document
---
```



YAML — Some Further Settings for PDF Outputs

- `fontsize`
  - the default is `10pt`
  - the other options are `11pt` and `12pt`
- `linkcolor`, `urlcolor`, `citecolor`
  - the default is the colour of the text
  - the other options are white, red, green, blue, cyan, magenta, yellow
- `link-citations`
  - the default is `no`
  - the other option is `yes` — a click on an citation will take the screen to the relevant entry in the list of references

Part 3. Writing Text

Syntax — Overview

- There are not one, but several different versions of Markdown
  - e.g., Pandoc, MultiMarkdown, CommonMark
  - each might implement the same things (e.g., citations) slightly differently, and each might offer unique functionalities

- R Markdown follows the syntax in Pandoc's Markdown
  - for the complete rules of the syntax, see Pandoc User's Guide
  - for a useful summary of the syntax, see the R Markdown Cheat Sheet

Syntax — Lines

Multiple spaces on a given line are reduced to one

```
This is a sentence followed by four spaces.    This is another sentence on the same line.
```

This is a sentence followed by four spaces. This is another sentence on the same line.

Line endings with fewer than two spaces are ignored

```
This is a sentence followed by one space.
This is another sentence on a new line.
```

This is a sentence followed by one space. This is another sentence on a new line.

Syntax — Hard Breaks

Two or more spaces at the end of lines introduce hard breaks, forcing a new line

```
This is a sentence followed by two spaces.
This is another sentence on a new line.
```

This is a sentence followed by two spaces.
This is another sentence on a new line.

Syntax — Line Blocks

Spaces on lines that start with a vertical line | are kept

```
| a one-space indent
|     a five-space indent
|          a ten-space indent
```

a one-space indent
   a five-space indent
      a ten-space indent

Syntax — Block Quotes

Lines starting with the greater-than sign > introduce block quotes*

```
> In God, we trust. All others must bring data.
>
> --- Anonymous
```

> In God, we trust. All others must bring data.
>
> — Anonymous

* Notice that three hyphens grouped together introduce an em-dash. Dashes are covered later on in the workshop.

Syntax — Paragraphs

One or more* blank lines introduce a new paragraph

```
This is the first sentence of a paragraph as it is preceded by a blank line. This is the second sentence of that paragraph, which is followed by a blank line.

This is the first sentence of a *new paragraph* as it is preceded by a blank line. This is the second sentence of that paragraph, which is followed by a blank line.
```

This is the first sentence of a paragraph as it is preceded by a blank line. This is the second sentence of that paragraph, which is followed by a blank line.

This is the first sentence of a *new paragraph* as it is preceded by a blank line. This is the second sentence of that paragraph, which is followed by a blank line.

* Multiple blank lines between paragraphs reduce to one.

Syntax — Comments

Text with the syntax `<!-- comments -->` is omitted from output

```
<!-- This paragraph needs re-writing -->
This is the first sentence of a paragraph as it is preceded by a blank line. This is the second
sentence of that paragraph, which is followed by a blank line.

This is the first sentence of a new paragraph <!-- I've removed italics --> as it is preceded
by a blank line. This is the second sentence of that paragraph, which is followed by a blank
line.
```

This is the first sentence of a paragraph as it is preceded by a blank line. This is the second sentence of that paragraph, which is followed by a blank line.

This is the first sentence of a new paragraph as it is preceded by a blank line. This is the second sentence of that paragraph, which is followed by a blank line.

Syntax — Headers

The number sign # introduces headers; lower levels are created with additional signs — up to total five levels

`# Introduction` becomes

Introduction

`## 1. Introduction` becomes

1. Introduction

`### 3.1 Introduction` becomes

3.1 Introduction

`#### Introduction` becomes

Introduction

`##### Introduction` becomes

Introduction

Syntax — Emphases

A pair of single asterisk * or underscores _ introduces italics

`*italics*` becomes *italics*

`_italics_` becomes *italics* as well

A pair of double asterisk or underscores introduces bold

`**bold**` becomes **bold**

`__bold__` becomes **bold** as well

These two rules can be combined

`**_bolditalics_**` becomes ***bolditalics***

`_**bolditalics**_` becomes ***bolditalics*** as well

Syntax — Strikethrough

A pair of double tildes ~ introduces strikethrough

`~~strikethrough~~` becomes ~~strikethrough~~

Strikethrough can be combined with italics or bold

`**~~strikebold~~**` or `__~~strikebold~~__`, they both become ~~**strikebold**~~

`~~**strikebold**~~` or `~~__strikebold__~~`, they both become ~~**strikebold**~~ as well

`*~~strikeitalitcs~~*` or `_~~strikeitalitcs~~_`, they both become ~~*strikeitalitcs*~~

`~~*strikeitalitcs*~~` or `~~_strikeitalitcs_~~`, they both become ~~*strikeitalitcs*~~ as well

Syntax — Links — Internal*

You can link text to section headers in the same document

`[Conclusion](#conclusion)` becomes Conclusion, and a click takes the screen to that section

Multi-word headers need hyphenation

`[Literature Review](#literature-review)` becomes Literature Review, and it works only if the second part is hyphenated

Syntax — Links — External

You can link text to URLs

`[visit my website](https://resulumit.com/)` becomes visit xtbg website

`[https://resulumit.com](https://resulumit.com/)` becomes https://resulumit.com

`<https://resulumit.com>` becomes https://resulumit.com as well

You can also link text to an email address

`[email me](mailto:resuluy@uio.no)`* becomes email me

`<resuluy@uio.no>` becomes resuluy@uio.no

* Notice the prefix mailto: in the syntax.

Syntax — Equations

Inline equations go between a pair of single dollar signs `$` — with no space between the signs and the equation itself

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

Block equations go in between a pair of double dollar signs — with or without spaces, it works

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

$$E = mc^{2}$$

`$$E = mc_{2} $$` becomes

$$E = mc_{2}$$

Syntax — Footnotes — Inline Notes

For inline footnotes, use the `^[footnote]` syntax

`Jane Doe^[Corresponding author.]` becomes Jane Doe[1]

Notice that

- the caret sign ^ comes before the left square bracket [
- this syntax works in YAML as well as in text
  - footnotes in YAML get symbols, in text they get numbers

[1] Corresponding author.

Syntax — Footnotes — Notes with Identifiers

An alternative is to use the `[^identifier]` syntax, with identifiers defined elsewhere in the same document

```
Dr Doe holds a PhD in rock science.[^defence_date]

[^defence_date]: She defended her thesis in 2017.
```

Dr Doe holds a PhD in rock science.[1]

Notice that

- the caret sign comes after the left square bracket
- this syntax works in text, but not in YAML

[1] She defended her thesis in 2017.

Syntax — Lists

Lines starting with asterisk `*` as well as plus `+` or minus `–` signs introduce lists

```
- books
- articles
- reports
```

- books
- articles
- reports

Syntax — Lists — Nesting

Lists can be nested within each other, with indentation

```
+ books
+ articles
    - published
    - under review
        + revised and resubmitted
    - work in progress
```

- books
- articles
  - published
  - under review
    - revised and resubmitted
  - work in progress

Syntax — Lists — Numbering

List items can be numbered

```
1. books
2. articles
    - published
    - under review
        + revised and resubmitted
    - work in progress
```

1. books
2. articles
    - published
    - under review
        - revised and resubmitted
    - work in progress

Syntax — Dashes

Two hyphens grouped together introduce an en-dash

`--` becomes –

Three hyphens grouped together introduce an em-dash

`---` becomes —

Syntax — Subscript and Superscript

A pair of tildes introduces subscript

`CO~2~` becomes CO_2

A pair of carets introduces superscript

`R^2^` becomes R^2

Syntax — Sub- and Super-scripts

A pair of tildes introduces subscript

`CO~2~` becomes CO_2

A pair of carets introduces subscript

`R^2^` becomes R^2

Notice that

- the syntax here (Markdown-based) is different than the one for equations (LaTeX-based)
  - e.g., `R^2^` versus `mc^{2}`

Part 4. Managing References

References — Bibliography Database

- References are defined in .bib files
  - they follow the BibTeX format
- `pandoc` looks for a .bib file, and for the definitions therein, to process citations
  - .bib files are specified with the `bibliography` variable in YAML
- `pandoc` can process a citation only if there is a linked entry in the .bib file
  - but not all entries have to be cited

References — Bibliography Database — Entries

- A BibTeX entry consists of three elements
  - a type
    - e.g., `@article`
  - a citation-key
    - e.g., `bennett2015`
  - a number of tags
    - e.g., `title`, `author`

```
@article{bennett2015,
  title={Peanut butter and jelly},
  author={Bennett, Stephanie},
  journal={Journal of Bone},
  year={2015}
  volume={1},
  number={12},
  pages={3--35},

}
```

- Different tags are available for different reference types
  - some tags are required, others are optional

References — Bibliography Database — Entries

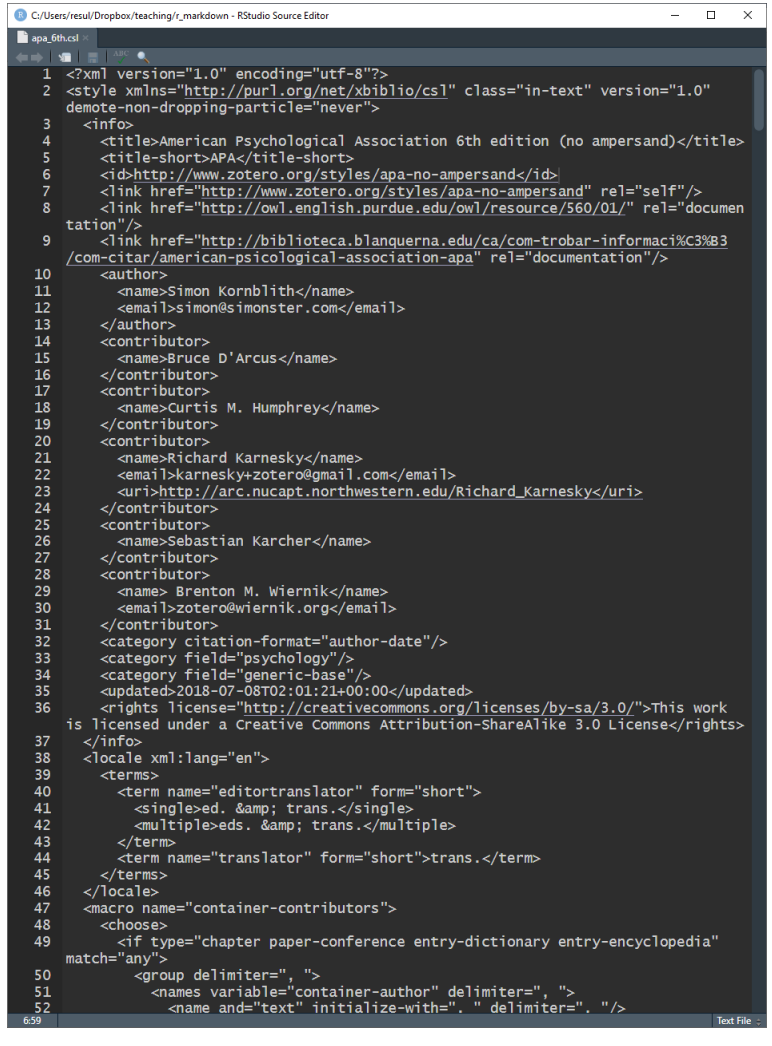
- One could create entries by hand
  - requires knowing the BibTeX format, entry types, tags, and related information about references to be cited
  - neither efficient nor necessary

- A good alternative is to use `Google Scholar`, which provides BibTeX entries
  - follow `cite -> BibTex` and copy
  - paste into .bib, edit if necessary, and save

- Some publishers and journals provide BibTeX entries on their website as well

References — Style

- Reference styles are defined in .csl files
  - files for different styles (e.g., APA) are available at https://www.zotero.org/styles
- `pandoc` looks for a .csl file, and for the styles therein, to style citations and references
  - .csl files are specified with the `csl` variable in YAML
  - if unspecified, it uses a Chicago author-date format
- .csl files affect the style only in outputs
  - no matter which the style is used, the citation syntax in .Rmd documents remains the same



References — In-text Citation Syntax — Author-Date Styles*

All citations keys take the 'at' sign @ while square brackets and/or minus signs introduce variation

`[@bennett2015]` becomes (Bennett, 2015)

`@bennett2015` becomes Bennett (2015)

`[-@bennett2015]` becomes (2015)

`-@bennett2015` becomes 2015

`[@bennett2015 35]` becomes (Bennett, 2015, p. 35)

`[@bennett2015 33-35]` becomes (Bennett, 2015, pp. 33–35)

`[@bennett2015, ch. 1]` becomes (Bennett, 2015, ch. 1)

`[@bennett2015; @gilbert2019]` becomes (Bennett, 2015; Gilbert, 2019)

`[see @bennett2015, for details]` becomes (see Bennett, 2015, for details)

`@bennett2015 [33-35]` becomes Bennett (2015, pp. 33–35)

* Specifically, the outputs on this slide are formatted according to the APA 7th edition.

References — In-text Citation Syntax — Numerical Styles

All citations keys take the 'at' sign @

`A clever sentence.[@bennett2015]` becomes A clever sentence.[1] in certain numerical sytles

`A clever sentence.[@bennett2015; @gilbert2019]` becomes A clever sentence.[1,2]

Individual styles may or may not use additional information, such as page numbers

`A clever sentence.[@bennett2015 35]` might become A clever sentence.[1] as well

Individual styles may or may not be sensitive to variation, such as square brackets

`A clever sentence. @bennett2015` might become A clever sentence.[1] as well

Citations — Reference List

The list of references appears after the last line of the output document, with no section header

- so that you can choose the header yourself, by ending .Rmd documents with a header of your choice

```
This is the last sentence of an APA style manuscript.

## References
```

This is the last sentence of an APA style manuscript.

References

Bennett, S. (2015). Peanut butter and jelly. *Journal of Bone, 1*(12), 3–35.

Gilbert, T. (2019). Turning wine into water. In M. Albert (Ed.), *The book of ground* (pp. 124–142). Antman.

References — Internal Links

For internal links from in-text citations to the reference list, set `link-citations: yes` in YAML

- a click on these links takes the screen to the relevant entry in the list
- the `linkcolor` variable make these links explicit
  - setting this is not necessary for the links to work — the default is black

```
---
...
bibliography: references.bib
csl: apa_7th.csl
link-citations: yes    #<<
linkcolor: blue        #<<
...
---
```

Part 5. Adding Code, Figures, and Tables

Code, in and outside Chunks

Code — Overview

Most codes go inside code chunks

- e.g., code that imports and cleans data, and/or produces tables and/or figures

````
```{r}

df <- read.csv("rmd_workshop_files/images_data/journals.csv") %>%
      mutate(age = 2020 - since,
             english = factor(english),
             subfield = factor(subfield))

```
````

Codes can also go in line with text

- e.g., code that results in a single statistic

```
The average H5 Index for the journals in the dataset is `r mean(df$h5_index)`.
```

Code Chunks — Overview

- Code chunks are delimited spaces between a pair of three backticks `
  - placed on their own lines in .Rmd documents, separate from text
  - their output, if there is any, appear in the output document
    - at about the same place as the chunk
    - might float around text to avoid breaking across pages

```
```
```
```

Code Chunks — Overview

- Code chunks are delimited spaces between a pair of three backticks `
  - placed on their own lines in .Rmd documents, separate from text
  - their output, if there is any, appear in the output document
    - at about the same place as the chunk
    - might float around text to avoid breaking across pages

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

- On the same line with the first delimiter, and in curly brackets {, code chunks take
  - a languge engine

Code Chunks — Overview

- Code chunks are delimited spaces between a pair of three backticks `
  - placed on their own lines in .Rmd documents, separate from text
  - their output, if there is any, appear in the output document
    - at about the same place as the chunk
    - might float around text to avoid breaking across pages

```
```{r, setup}
```
```

- On the same line with the first delimiter, and in curly brackets {, code chunks take
  - a language engine
  - a label

Code Chunks — Overview

- Code chunks are delimited spaces between a pair of three backticks `
  - placed on their own lines in .Rmd documents, separate from text
  - their output, if there is any, appear in the output document
    - at about the same place as the chunk
    - might float around text to avoid breaking across pages

- On the same line with the first delimiter, and in curly brackets {, code chunks take
  - a language engine
  - a label
  - one or more options

````
```{r, setup, echo=FALSE}
```
````

Code Chunks — Lenguage Engines

The first item in code chunks indicates the engine to run the code

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

Note that

- indicating an engine for each chunk is a must
  - otherwise, any code* in these chunks cannot be executed
- `r` is the specified engine, indicating that the code in the chunk above should be run by R
  - it could have been `python`, which we will not cover in this workshop

* The above chunk has no code — it is for demonstration only.

Code Chunks — Labels

It is recommended, but optional, to label the code chunks

````
```{r, data_import}
df <- read_csv("data/journals.csv")
```
````

Note that

- labels are written after the language engine, separated by a comma
  - in the example above, the chunk is labelled as `data_import`
- chunks without labels are otherwise automatically numbered
  - specifying informative labels can be helpful for, e.g., navigating through error messages
- duplicate labels lead to errors during compilation

Code Chunks — Options

Code chunks can take further options

```
```{r, setup, include=FALSE}
```
```

Note that

- in the example above, the `include` option is set to `FALSE`
  - with this option and value, nothing from this chunk will be included in the output document
- The complete list of options is available at https://yihui.org/knitr/options
  - R Markdown Cheat Sheet provides a helpful list as well
- leaving spaces around the equal sign `=`, between option tags and values, should be avoided
  - such spaces might lead to errors

Code Chunks — Options — Alternative Syntax

Options can be specified inside code chunks as well, after a number sign and a vertical line `#|`

- therefore the following chunks have the same function

````
```{r, echo=FALSE, eval=TRUE}

```
````

````
```{r}

#| echo = FALSE, eval = TRUE
```
````

````
```{r}

#| echo = FALSE
#| eval = TRUE
```
````

Code Chunks — Options — Defaults

Options have default values

- e.g., for `echo`, the default is `TRUE`
  - `echo`: should the source code printed in the output?
  - `TRUE`: yes it should

- therefore the following two chunks have the same function

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

````
```{r, echo=TRUE}
```
````

Code Chunks — Options — Defaults

This chunk prints two things in the output document — (a) the code and (b) the head of the data frame

````
```{r}
head(df)
```
````

```
head(df)
```

```
##                name   origin   branch h5_index h5_median english subfield
## 1  Journal of Bears Americas Physical       73        97       1        1
## 2   Journal of Moon     Asia   Social       72       106       1        0
## 3 Journal of Lumber Americas Physical       72       100       1        1
## 4 Journal of Houses   Europe   Social       72       102       1        0
## 5  Journal of Water   Europe   Social       70       100       1        0
## 6  Journal of Jeans Americas Physical       69       101       1        1
##   issues age
## 1      7  61
```

Code Chunks — Options — Examples

Setting `echo=FALSE` prevents the code from being displayed in the output document

````
```{r ... echo=FALSE}
head(df)
```
````

This chunk therefore prints one thing in the output document — the head of the data frame

```
##                name   origin   branch h5_index h5_median english subfield
## 1  Journal of Bears Americas Physical       73        97       1        1
## 2   Journal of Moon     Asia   Social       72       106       1        0
## 3 Journal of Lumber Americas Physical       72       100       1        1
## 4 Journal of Houses   Europe   Social       72       102       1        0
## 5  Journal of Water   Europe   Social       70       100       1        0
## 6  Journal of Jeans Americas Physical       69       101       1        1
##   issues age
## 1      7  61
## 2      6  64
## 3      8  30
## 4      8  38
```

Code Chunks — Options — Examples

Prevent the result(s) of the source code from being displayed in the output document

````
```{r ... results="hide"}
head(df)
```
````

This chunk therefore prints one thing in the output document — the source code

```
head(df)
```

Setting `results="asis"` passes the results as they are produced by the code — `pandoc` does not transform these. In creating tables for PDF output with the `stargazer` package, this option is a must.

Code Chunks — Options — Examples

Cache results for future compilations

```
```{r ... cache=TRUE}
```
```

Note that caching

- is useful especially for chunks that take a long time to execute
  - it can speed up the compilation process
- avoids executing the chunks at every compilation
  - unless the chunk is newly created or edited since the last cached compilation
- creates a new folder in your working directory
  - an alternative location can be specified with the `cache.path` option

Code Chunks — Options — Examples

Prevent R from running the code in the chunk altogether

```
```{r ... eval=FALSE}
```
```

Prevent messages and/or warnings from being displayed in the output

```
```{r ... error=FALSE, message=FALSE, warning=FALSE}
```
```

Code Chunks — Options — Examples

Define the actual dimensions of figures, in inches

```
```{r ... fig.height=6, fig.width=9}
```
```

Define the size of figures as they appear in the output document, with `out.width` and/or `out.height`

```
```{r ... out.width="50%"}
```
```

Define the alignment of figures — `left`, `right`, or `center`

```
```{r ... fig.align="center"}
```
```

Code Chunks — Options — Examples

Define captions for figures

```
```{r ... fig.caption="A Scatter Plot"}
```
```

Set the resolution for figures

```
```{r ... dpi=300}
```
```

Set extra options, such as angle, that output format would accept for figures

```
```{r ... out.extra="angle=45"}
```
```

Code Chunks — The Setup Chunk

It is recommended to use the first code chunk for general setup, where you can

- define your own defaults for chunk options, with `knitr::opts_chunk$set()`
  - avoids repeating chunk options
- load the necessary packages
- import raw data

````
```{r, setup, include=FALSE}

# chunk option defaults
knitr::opts_chunk$set(echo=FALSE, message=FALSE)

# packages
library(dplyr)
library(ggplot2)
library(stargazer)

# data
df_raw <- read.csv("journals.csv")

```
````

Code Chunks — The Data Chunk

I recommend using the second chunk for the main operations* on raw data

- e.g., for data cleaning and other transformations
- some minor transformations could be left to lower chunks
  - e.g., capitalizing variable names for figures

````
```{r, data, echo=FALSE ...}

df <- df_raw %>%
      mutate(subfield = as.factor(subfield),
             english = as.factor(english),
             age = 2020 - since) %>%
      select(-since)

```
````

Inline Code — Overview

Code can also be incorporated in text, with the `` `r ` `` syntax

- unlike chunks, these do not take options
- the output document will display the result of the code
  - in the exact place of the source code
- the result of the code will have the same formatting with the text

Inline Code — Examples

```
If we multiply _pi_ by 5, we get `r pi * 5`.
```

If we multiply *pi* by 5, we get 15.7079633.

```
The average H5 Index for the journals in the dataset is `r mean(df$h5_index)`, which would
round to `r round(mean(df$h5_index), digits = 1)`.
```

The average H5 Index for the journals in the dataset is 26.3611366, which would round to 26.4.

```
__Only `r nrow(subset(df, english == 0))` journals__ in the dataset are published in a language
other than English.
```

Only 113 journals in the dataset are published in a language other than English.

Figures

Figures — Images — Markdown Syntax

The syntax `![Figure Caption](figure.extension)` embeds images, and/or figures produced elsewhere,* into .Rmd documents

- similar to the link syntax, only this time it is preceded by an exclamation mark !
- goes outside code chunks, on a new line
- simple, but not very customisable

* Ideally, reproducible papers should produce their own images with data and code. However, there might be situations where this is not possible.

Figures — Images — Markdown Syntax

```
![A screenshot of the Google Scholar homepage](../image/google_scholar.png)
```

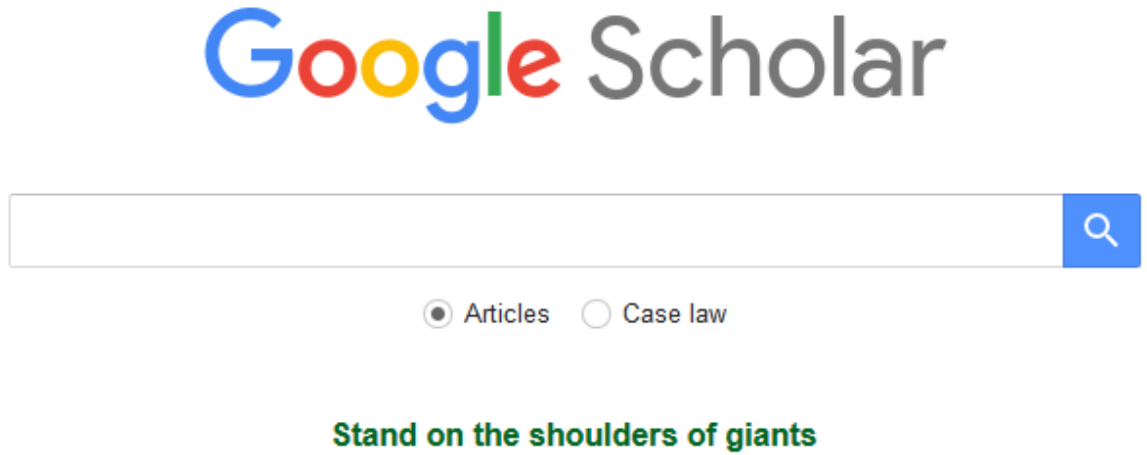


Figure 1: A screenshot of the Google Scholar homepage.

Figures — Images — Markdown Syntax

Figures are numbered automatically

```
![A screenshot of the Google Scholar homepage](../image/google_scholar.png)
```

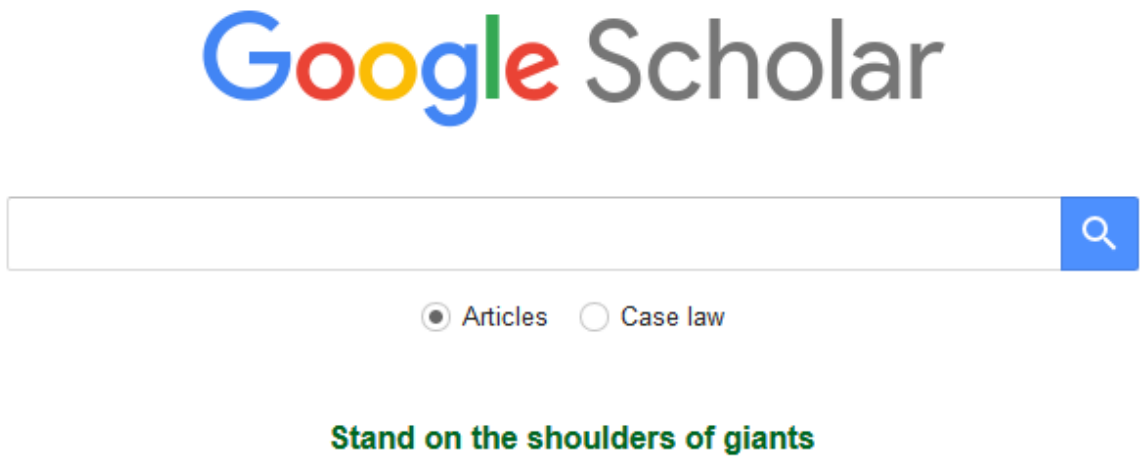


Figure 1: A screenshot of the Google Scholar homepage.

Figures — Images — Markdown Syntax

The syntax can accept `width` or `height` attributes as follows

```
![A screenshot of the Google Scholar homepage](../image/google_scholar.png){ width=40% }
```

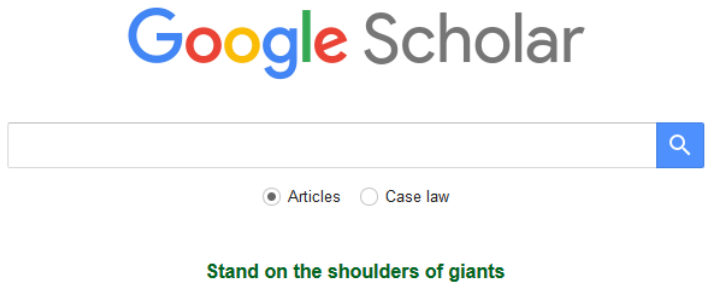


Figure 1: A screenshot of the Google Scholar homepage.

Figures — Images — `knitr`

The `knitr` package offers a capable alternative with the `include_graphics()` function

- this goes inside code chunks
  - use the function with the double-colon operator ::
    - e.g., `knitr::include_graphics("figure.extension")`

- this is more customisable, through the use of code chunks
  - size is defined with the `out.width` or `out.hight` options
    - rather than `fig.height` and/or `fig.width`

Figures — Images — `knitr`

The `knitr` package offers a capable alternative with the `include_graphics()` function

````
```{r, screenshot, echo=FALSE, fig.cap="A screenshot of the Google Scholar homepage."}
knitr::include_graphics("../image/google_scholar.png")
```
````

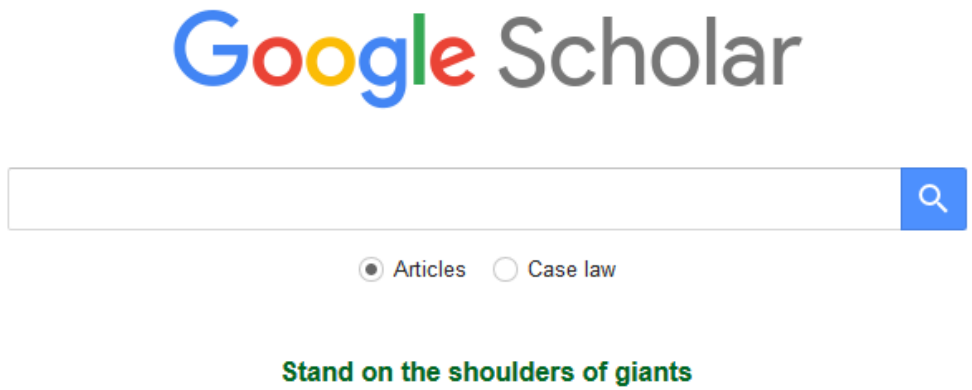


Figure 1: A screenshot of the Google Scholar homepage.

Figures — Images — `knitr`

Size is defined with the chunk options `out.width` or `out.hight`

```
```{r ... out.width="40%"}
knitr::include_graphics("../image/google_scholar.png")
```
```

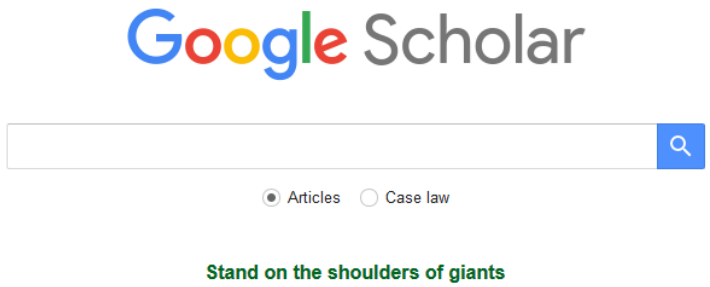


Figure 1: A screenshot of the Google Scholar homepage.

Figures — Images — `knitr`

Most other chunk options are common with figures plotted within R Markdown, such as `fig.align`

```
```{r ... fig.align="center"}
knitr::include_graphics("../image/google_scholar.png")
```
```

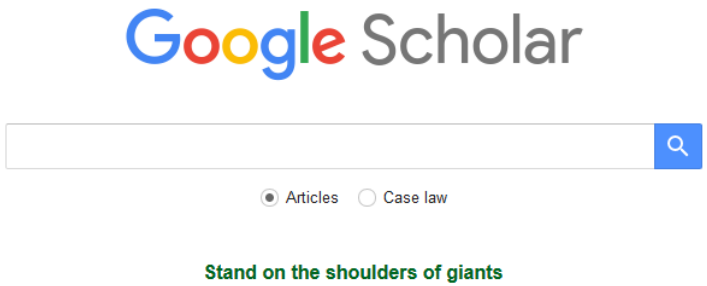


Figure 1: A screenshot of the Google Scholar homepage.

Figures — `ggplot2` — Overview

- A powerful package for visualising data
- Used widely, not only by academics, but also by large corporations such as the New York Times
- A huge amount is written on this package. See, for example,
  - the package documentation
  - this book by its creator Hadley Wickham
  - this reference page
  - this webinar by one of its authors, Thomas Lin Pedersen
  - these extensions, maintained by the `ggplot2` community
- Among its alternatives are the `base` and `plotly` packages

Figures — `ggplot2` — Basics

1) The `ggplot` function and the `data` argument

- specify a data frame in the main `ggplot` function

```
ggplot(data = df)
```

2) The mapping aesthetics, or aes; most importantly, the variable(s) that we want to plot

- specify as an additional argument in the same `ggplot` function

```
ggplot(data = df, mapping = aes(x = h5_median, y = h5_index, color = subfield))
```

3) The geometric objects, or geom; the visual representations

- specify, after a plus sign `+`, as an additional function

```
ggplot(data = df, mapping = aes(x = h5_median, y = h5_index, color = subfield)) +
       geom_point() #<<
```

Figures — ggplot2

Put the code in a chunk, and give it a caption

```
```{r, scatterplot, fig.cap = "A scatterplot of journal

ggplot(data = df, mapping = aes(x = h5_median, y = h5_ir
       geom_point()

```
```

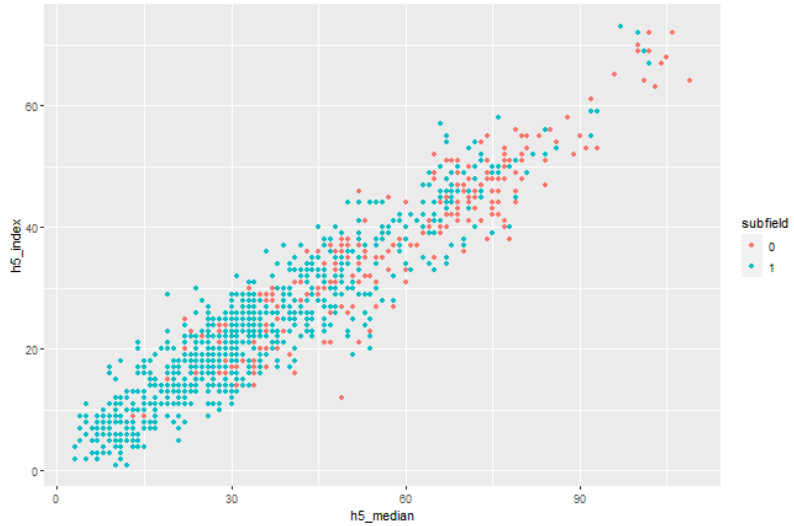


Figure 1. A scatterplot of journal metrics.

Figures — ggplot2

Add facets for subgroups, e.g., branch

```
```{r, scatterplot, fig.cap = "A scatterplot of journal

ggplot(data = df, mapping = aes(x = h5_median, y = h5_ir
       geom_point() +
       facet_wrap(. ~ branch)

```
```

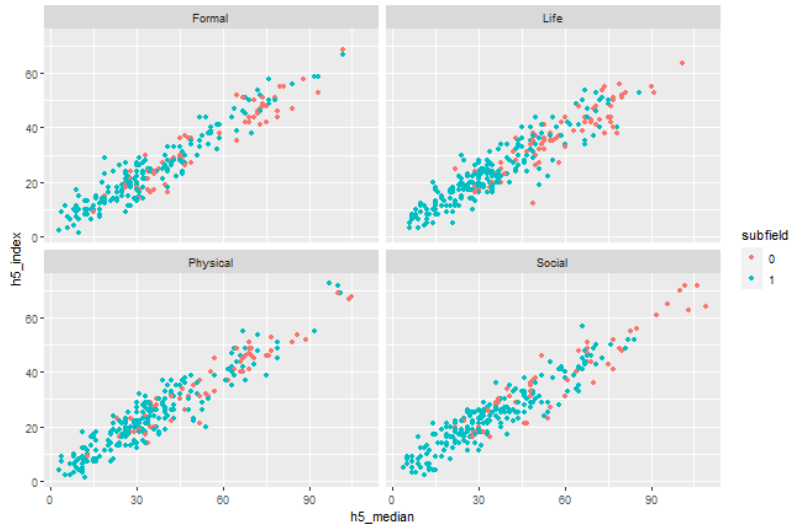


Figure 1. A scatterplot of journal metrics.

Figures — ggplot2

Scale the colour to improve the legend

```
```{r, scatterplot, fig.cap = "A scatterplot of journal

ggplot(data = df, mapping = aes(x = h5_median, y = h5_ir
       geom_point() +
       facet_wrap(. ~ branch) +
       scale_colour_discrete(name = "Journal Type", brea

```
```

Figure 1. A scatterplot of journal metrics.

Figures — ggplot2

Change the theme

```
```{r, scatterplot, fig.cap = "A scatterplot of journal

ggplot(data = df, mapping = aes(x = h5_median, y = h5_ir
       geom_point() +
       facet_wrap(. ~ branch) +
       scale_colour_discrete(name = "Journal Type", brea
       theme_bw()


```
```

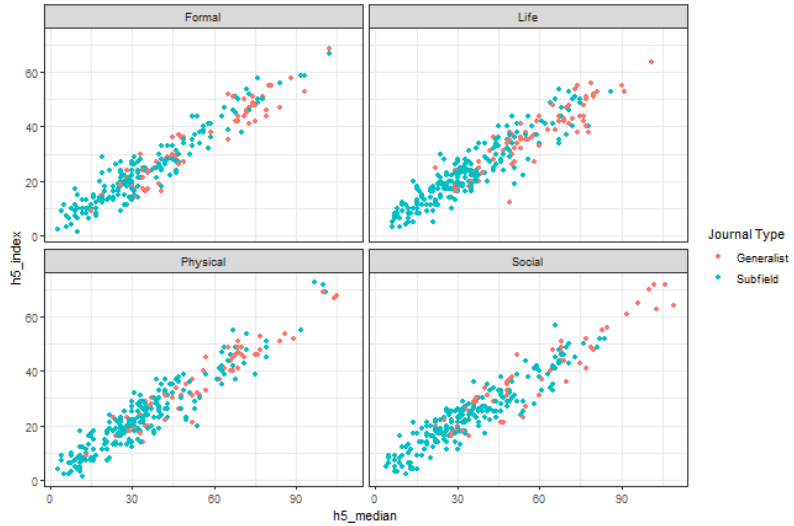


Figure 1. A scatterplot of journal metrics.

Figures — ggplot2

Improve the axis labels, e.g., with capital first letters

```
```{r, scatterplot, fig.cap = "A scatterplot of journal

ggplot(data = df, mapping = aes(x = h5_median, y = h5_ir
       geom_point() +
       facet_wrap(. ~ branch) +
       scale_colour_discrete(name = "Journal Type", brea
       theme_bw() +
       labs(x = "H5 Median", y = "H5 Index")

```
```

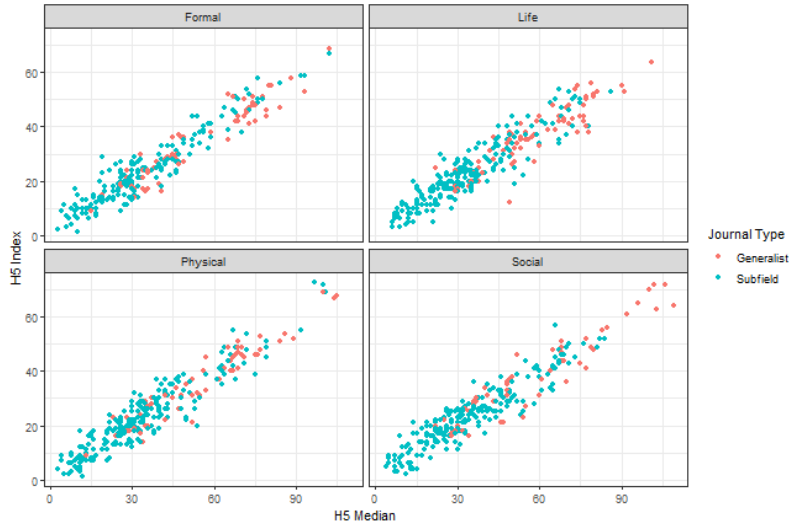


Figure 1. A scatterplot of journal metrics.

Figures — `ggplot2` — Notes

`geom_point` is one of many geoms avilable

- see this https://ggplot2.tidyverse.org/reference for other options, including
  - `geom_bar` for bar charts
  - `geom_boxplot` for box and whiskers plots

Tables

Tables — Markdown Syntax

The following syntax, outside code chunks, introduces tables that pandoc can recognise

```
First Column  Second Column
------------  -------------
First cell    First cell
Second cell   Second cell
Third cell    Third cell
```

First Column	Second Column
First cell	First cell
Second cell	Second cell
Third cell	Third cell

Tables — Markdown Syntax

The position of headers, relative to their line underneath, defines column alignments

```
Left-Aligned          Centered
---------------  ----------------
First cell        First cell
Second cell       Second cell
Third cell        Third cell
```

Left-Aligned	Centered
First cell	First cell
Second cell	Second cell
Third cell	Third cell

Tables — Markdown Syntax

A line starting with a colon, placed before or after tables, introduces captions

```
    Centered         Right-Aligned
---------------  ----------------
First cell       First cell
Second cell      Second cell
Third cell       Third cell

: A hand-made table with R Markdown
```

Table 1: A hand-made table with R Markdown

Centered	Right-Aligned
First cell	First cell
Second cell	Second cell
Third cell	Third cell

Tables — Markdown Syntax

The caption line itself needs to be surrounded by empty lines

```
    Centered         Right-Aligned
----------------  ----------------
First cell        First cell
Second cell       Second cell
Third cell        Third cell

: A hand-made table with R Markdown
```

Table 1: A hand-made table with R Markdown

Centered	Right-Aligned
First cell	First cell
Second cell	Second cell
Third cell	Third cell

Tables — Markdown Syntax

Tables are numbered automatically

```
: A hand-made table with R Markdown

    Centered         Right-Aligned
---------------  ----------------
First cell       First cell
Second cell      Second cell
Third cell       Third cell
```

Table 1: A hand-made table with R Markdown

Centered	Right-Aligned
First cell	First cell
Second cell	Second cell
Third cell	Third cell

Tables — Markdown Syntax

Grid tables, with the following syntax, can handle complex cells with multiple lines and/or lists

```
+--------------------+--------------------+
| First Column       | Second Column      |
+====================+====================+
| - First item       | First cell         |
| - Second item      |                    |
| - Third item       |                    |
+--------------------+--------------------+
|Second cell         | Second cell with a |
|                    | long text          |
+--------------------+--------------------+
| Third cell         | Third cell         |
|                    |                    |
+--------------------+--------------------+

: A grid table with multi-line cells
```

Table 1: A grid table with multi-line cells

First Column	Second Column
- First item - Second item - Third item	First cell
Second cell	Second cell with a long text
Third cell	Third cell

Tables — Markdown Syntax

Grid tables can be aligned as well, with colons at the boundaries of the header separator*

```
+--------------------+--------------------+
| Left-Aligned       | Centered           |
+:===================+:==================:+ #
| - First item       | First cell         |
| - Second item      |                    |
| - Third item       |                    |
+--------------------+--------------------+
|Second cell         | Second cell with a |
|                    | long text          |
+--------------------+--------------------+
| Third cell         | Third cell         |
|                    |                    |
+--------------------+--------------------+

: A grid table with multi-line cells
```

Table 1: A grid table with multi-line cells

Left-Aligned	Centered
- First item - Second item - Third item	First cell
Second cell	Second cell with a long text
Third cell	Third cell

* Use := for left-aligned, :=: for centered, =: for right-aligned columns.

Tables — `stargazer` — Overview

- A capable package for creating at least three kinds of tables
  - raw data, in columns and rows
  - descriptive/summary statistics
  - regression models
- Used widely by academics, even tough it has not been updated since 2018
- Creates LaTeX code, HTML/CSS code, and ASCII text to be knitted
- A lot is written on this package. See, for example,
  - the package documentation
  - this vignette by its author Marek Hlavac
  - this tutorial by Jake Russ
- Among its alternatives are the `knitr`, `kableExtra`, and `huxtable` packages

Tables — `stargazer` — Notes

- The `stargazer` package requires specific settings
  - in the chunk options
  - and, in the `type` argument of the `stargazer()` function

- These settings depend on the desired output format,* as shown below

Output	Chunk Option	Type Argument
LaTex / PDF	results="asis"	latex
HTML	results="asis"	html
Word	comment=""	text

* The following slides use the setting for LaTex and PDF outputs.

Tables — `stargazer` — Notes

- `stargazer` tables look slightly different in different output formats
  - on the following slides, they will have the HTML look
  - even if the slides display the setting for LaTex and PDF outputs

- In fact, it is currently not quite possible to `knit` `stargazer` code into tables in Word documents
  - though it can `knit` ASCII text, looking like a table
  - some popular workarounds:
    - `knit` to HTML as well as Word, copy the tables from HTML to Word
    - `knit` to PDF, open the PDF in Word
    - use a different package to create tables, such as `huxtable`

Tables — `stargazer` — Basics

- The `stargazer()` function
  - this is probably the only fuction you will ever use from this package
    - but it accepts many, many arguments to customise tables

- The `data` argument of that function, with two main options
  1. a data frame for data or summary statistics tables
     - e.g., `df`, here coming from `df <- read_csv(journals.csv)`
  2. one or more regression models for regression tables
     - e.g., `lm1`, here coming from `lm1 <- lm(h5_index ~ issues, data = df)`

Tables — `stargazer` — Data Tables

Table the first four rows of the dataset

````
```{r, data_table, echo=FALSE, results="asis"}
stargazer(data = head(df, n = 4), type = "latex", summary = FALSE)
```
````

Notice the options of the chunk and the arguments of the function

- with `echo=FALSE`, the code will not be displayed in the output document
- with `results="asis"`, `knitr` will pass through results without reformatting them
  - these results are produced in LaTeX, due to `type = "latex"`
  - they should remain LaTeX because our outcome document is PDF, converted from LaTeX
- with `summary = FALSE`, the table will present the data, not its descriptive statistics

Tables — `stargazer` — Data Tables

Table the first four rows of the dataset

```
```{r, data_table, echo=FALSE, results="asis"}

stargazer(data = head(df, n = 4), type = "latex", summary = FALSE)

```
```

% Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harvard.edu
% Date and time: Fri, Apr 10, 2020 - 12:31:21

Table 1:

	name	origin	branch	h5_index	h5_median	english	subfield	issues	age
1	Journal of Bears	Americas	Physical	73	97	1	1	7	61
2	Journal of Moon	Asia	Social	72	106	1	0	6	64
3	Journal of Lumber	Americas	Physical	72	100	1	1	8	30
4	Journal of Houses	Europe	Social	72	102	1	0	8	38

Tables — `stargazer` — Data Tables

Set `header = FALSE` to remove the note preceding tables

````
```{r, data_table, echo=FALSE, results="asis"}
stargazer(data = head(df, n = 4), type = "latex", summary = FALSE, header = FALSE)
```
````

Table 1:

	name	origin	branch	h5_index	h5_median	english	subfield	issues	age
1	Journal of Bears	Americas	Physical	73	97	1	1	7	61
2	Journal of Moon	Asia	Social	72	106	1	0	6	64
3	Journal of Lumber	Americas	Physical	72	100	1	1	8	30
4	Journal of Houses	Europe	Social	72	102	1	0	8	38

Tables — `stargazer` — Data Tables

Define a caption with the `title` argument

```
```{r, data_table, echo=FALSE, results="asis"}

stargazer(data = head(df, n = 4), type = "latex", summary = FALSE, header = FALSE,
          title = "First four rows of the dataset")

```
```

Table 1: First four rows of the dataset

	name	origin	branch	h5_index	h5_median	english	subfield	issues	age
1	Journal of Bears	Americas	Physical	73	97	1	1	7	61
2	Journal of Moon	Asia	Social	72	106	1	0	6	64
3	Journal of Lumber	Americas	Physical	72	100	1	1	8	30
4	Journal of Houses	Europe	Social	72	102	1	0	8	38

Tables — `stargazer` — Summary Statistics Tables

Create a table of summary statistics instead, for the complete dataset

```
```{r, summary_table, echo=FALSE, results="asis"}

stargazer(data = df, type = "latex", summary = TRUE, header = FALSE,
          title = "Descriptive statistics")

```
```

Table 1: Descriptive statistics

Statistic	N	Mean	St. Dev.	Min	Pctl(25)	Pctl(75)	Max
h5_index	1,091	26.361	13.814	1	17	35	73
h5_median	1,091	39.400	21.272	3	25	52	109
issues	1,091	4.676	1.786	1	3	6	12
age	1,091	42.902	26.370	1	23	56	158

Tables — `stargazer` — Summary Statistics Tables

Keep only a selection of statistics

```
```{r, summary_table, echo=FALSE, results="asis"}

stargazer(data = df, type = "latex", summary = TRUE, header = FALSE,
          title = "Descriptive statistics", summary.stat = c("n", "mean", "sd", "min", "max"))

```
```

Table 1: Descriptive statistics

Statistic	N	Mean	St. Dev.	Min	Max
h5_index	1,091	26.361	13.814	1	73
h5_median	1,091	39.400	21.272	3	109
issues	1,091	4.676	1.786	1	12
age	1,091	42.902	26.370	1	158

Tables — `stargazer` — Summary Statistics Tables

Omit a selection of statistics for the same effect

```
```{r, summary_table, echo=FALSE, results="asis"}

stargazer(data = df, type = "latex", summary = TRUE, header = FALSE,
          title = "Descriptive statistics", omit.summary.stat = c("p25", "p75"))

```
```

Table 1: Descriptive statistics

Statistic	N	Mean	St. Dev.	Min	Max
h5_index	1,091	26.361	13.814	1	73
h5_median	1,091	39.400	21.272	3	109
issues	1,091	4.676	1.786	1	12
age	1,091	42.902	26.370	1	158

Tables — `stargazer` — Summary Statistics Tables

Flip the table

```
```{r, summary_table, echo=FALSE, results="asis"}

stargazer(data = df, type = "latex", summary = TRUE, header = FALSE, flip = TRUE,
          title = "Descriptive statistics", omit.summary.stat = c("p25", "p75"))

```
```

Table 1: Descriptive statistics

Statistic	h5_index	h5_median	issues	age
N	1,091	1,091	1,091	1,091
Mean	26.361	39.400	4.676	42.902
St. Dev.	13.814	21.272	1.786	26.370
Min	1	3	1	1
Max	73	109	12	158

Tables — `stargazer` — Regression Tables

Models can also be estimated outside the function first

```
```{r, regression_table, echo=FALSE, results="asis"}

 lm1 <- lm(h5_index ~ subfield, data = df)

 stargazer(data = lm1, type = "latex", header = FALSE,
           title = "Regression Results")

```
```

Table 1: Regression Results

	Dependent variable:
	h5_index
issues	1.913***
	(0.227)
Constant	17.415***
	(1.137)
Observations	1,091
R^2	0.061
Adjusted R^2	0.060
Residual Std. Error	13.391 (df = 1089)
F Statistic	70.959*** (df = 1; 1089)
Note:	*p<0.1; **p<0.05; ***p<0.01

Tables — `stargazer` — Regression Tables

Keep only a selection of statistics

```
```{r, regression_table, echo=FALSE, results="asis"}

stargazer(data = lm1, type = "latex", header = FALSE,
          title = "Regression Results",
          keep.stat = c("n", "rsq"))

```
```

Table 1: Regression Results

	Dependent variable:
	h5_index
issues	1.913^{***}
	(0.227)
Constant	17.415^{***}
	(1.137)
Observations	1,091
R^2	0.061
Note:	*p<0.1; **p<0.05; ***p<0.01

Tables — `stargazer` — Regression Tables

Display multiple models in the same table

```
```{r, regression_table, echo=FALSE, results="asis"}

stargazer(data = list(lm1, lm2), type = "latex",
          header = FALSE, title = "Regression Results",
          keep.stat = c("n", "rsq"))

```
```

Table 1: Regression Results

	Dependent variable:	
	h5_index	
	(1)	(2)
issues	1.913***	1.424***
	(0.227)	(0.212)
english1		17.262***
		(1.244)
Constant	17.415***	4.226***
	(1.137)	(1.415)
Observations	1,091	1,091
R^2	0.061	0.202
Note:	*p<0.1; **p<0.05; ***p<0.01	

Tables — `stargazer` — Regression Tables

Change variable labels

```
```{r, regression_table, echo=FALSE, results="asis"}

stargazer(data = list(lm1, lm2), type = "latex",
          header = FALSE, title = "Regression Results",
          keep.stat = c("n", "rsq"),
          dep.var.labels = "H5 Index",
          covariate.labels = c("Issues", "English"))

```
```

Table 1: Regression Results

	Dependent variable:	
	H5 Index	
	(1)	(2)
Issues	1.913***	1.424***
	(0.227)	(0.212)
English		17.262***
		(1.244)
Constant	17.415***	4.226***
	(1.137)	(1.415)
Observations	1,091	1,091
R^2	0.061	0.202
Note:	*p<0.1; **p<0.05; ***p<0.01	

Tables — `stargazer` — Regression Tables

Change significance levels

```
```{r, regression_table, echo=FALSE, results="asis"}

stargazer(data = list(lm1, lm2), type = "latex",
          header = FALSE, title = "Regression Results",
          keep.stat = c("n", "rsq"),
          dep.var.labels = "H5 Index",
          covariate.labels = c("Issues", "English"),
          star.cutoffs = c(0.05, 0.01, 0.001))

```
```

Table 1: Regression Results

	Dependent variable:	
	H5 Index	
	(1)	(2)
Issues	1.913^{***}	1.424^{***}
	(0.227)	(0.212)
English		17.262^{***}
		(1.244)
Constant	17.415^{***}	4.226^{**}
	(1.137)	(1.415)
Observations	1,091	1,091
R^2	0.061	0.202
Note:	*p<0.05; **p<0.01; ***p<0.001	

Thank you for listening!

Any questions now or email me at **dossa@xtbg.org.cn**

Slides created via the R package **xaringan**.

The chakra comes from remark.js, **knitr**, and R Markdown.