---

title: "R Markdown Demo"

author: "Ines Arana"

date: "2022-23"

output:

html\_document:

number\_section: true

---

```{r setup, include=FALSE}

knitr::opts\_chunk$set(echo = TRUE)

```

# R Markdown Resources

1. There are excellent [RMarkdown resources](https://rmarkdown.rstudio.com/) including a [tutorial](https://rmarkdown.rstudio.com/lesson-1.html).

1. The cheatsheet is available from RStudio by going to

\*\*Help -> Cheat Sheets -> R Markdown Cheat Sheet\*\*. You can also get it from [this link](https://raw.githubusercontent.com/rstudio/cheatsheets/master/rstudio-ide.pdf).

# R Markdown document elements

1. A \*\*YAML\*\* header which includes title, aughor, etc.

This header appears at the top of the document between \*\*---\*\*.

For an example, check the first few lines of this document.

1. \*\*Code chunks\*\*. These are delimited between

\* ```{r}

\* ```

These code chunks is where the R code goes.

1. \*\*Text area\*\*. This is any area except the YAML header and code chunks.

# Code chunk options

A code chunk can have a number of different options, depending on whether:

\* the code and/or the results are wanted in the final document.

\* the code is to be run or not.

\* messages are to appear or not.

\* warnings are included in the final document or not.

\* errors in the code stop the production of the document or whether these are displayed and the processing of the rest of the file continued.

The default display includes code and output.

The options are included in the line which indicates the start of an R block, after the r.

## include

The include option indicates whether the code chunk and results are to be included in the final document. The code is run.

For example, the code below can be used to clear everything and set the working directory to the folder where this file (Demo.Rmd) is located. if we do not wish this code and results to appear, we add ",include = F, i.e. our start of code chunk becomes

\* ```{r include = F}

```{r include = F}

x <- 10

y <- 20

z <- 10 \* 20

z

```

Note that you will not see the above chunk in the final document.

By default include is set to TRUE, so the code below (which does not have the include option set) will be included as well as any output.

```{r}

p <- 1000

y <- p / 55

```

## The \*echo\* option

The example below prints code and results.

```{r}

x = 4

y = x \* 6

y

```

Use echo = FALSE to exclude the R code from the document, displaying only the results of running the code.

```{r echo=FALSE}

x = 4

y = x \* 6

y

```

## The \*message\* option

This option can be used suppress messages from appearing when set to FALSE.

For example, below we try to multiply 3 vectors of different lengths. Note that the length of the 1st vector (the longest) is not a multiple of the length of the 2nd vector. Thus, multiplying these 2 vectors will give a warning.

```{r}

a <- c(10,20,30)

b <- c(7,8)

a \* b

```

But if we set message to false, the warning will not appear.

```{r message = FALSE}

a <- c(10,20,30)

b <- c(7,8)

a \* b

```

## The \*eval\* option

Use eval = FALSE to display the code, but not run it.

```{r eval=FALSE}

z = x \* y

z

```

## The \*warning\* option

There are times when the code runs, but issues warnings which you do not wish to include in your final document. The warning option can be used to exclude warnings.

For example, the following will issue a warning.

```{r}

p <- c(1:10)

q <- c(25:27)

r <- p\*q

r

```

Use warning = FALSE to suppress warnings.

Compare the output of this code with the one for the previous block

```{r warning=FALSE}

p <- c(1:10)

q <- c(25:27)

r <- p\*q

r

```

## The \*error\* option

To "ignore" errors and allow the code to run with errors use error = TRUE setting, this tries to run the code, fails when an error occurs, displays the error in the markdown output file but then carries on to process the remainder of the markdown file:

```{r error = TRUE}

a <- "this is text"

b <- 10

c <- a / b # this produces an error as varible a contains text.

```

Any later code that relies on something that has failed to run will also have errors.

# Text options

## Sections

One or more has characters at the beginning of a line indicate that the text following them is a section heading. The more hashes, the lower the level of the section. For example, one hash indicates section, two hashes indicates subsection, three hashes indicates subsubsection, etc.

For an example of a section, see how the R Markdown resources were indicated towards the beginning of this file.

Sections must be preceded by a blank line (or an R block).

## This is a sub-section

So the actual heading is smaller than for a section.

### This is a sub-sub-section

So the heading is even smaller.

You can use even more hashes for sub-sub-...sections.

###### This is not effective, but see how it is numbered.

If the \*number\_section\* in the YAML area is set to true, your sections (and subsections, etc) will be numbered. If the number\_section is not set or it is set to false, sections will not be numbered. With the \*number\_section\* set to true, you can ensure specific sections are not numbered by adding {-} after their section name (title).

For example

### This is an unumbered subsubsection when others are numbered {-}

The hashes used to denote a section must be followed by a blank space.

###This is not a subsubsection

The above is not a subsubsection because there is no space between the three hashes and the section title.

## Basic text formatting

\* Text between single stars \*is in italics\*

\* Text between double stars \*\*is in bold\*\*

\* Text between triple-stars \*\*\*is in bold italics\*\*\*

\* Superscripts are indicated between ^ symbols. For example

x square is x^2^ and y to the power of 12 is y^12^.

\* Subscripts are indicated between ~ symbols. For example, results subscript test1 is created using results~test1~.

\* Strikethrough text is indicated between double ~ symbols. For example ~~crossed out rubish text~~.

## Bullets, numbers and indentations

### Bullets

\* A single start preceding text produces a bulletted paragraph.

### Numbers

1. A number ending with a full stop at the beginning of a line indicates a numbered paragraph. Number 1 can be used repeatedly and the numbering will be "put right".

1. So this section which was started with "1." will be numbered 2.

8. This starts with an 8, but will be numbered 3.

### Indentations

Indentations are indicated by exactly 4 blank spaces preceding the paragraph. This can also be used to indent numbering.

1. this is a numbered paragragh

1. this is a numbered paragraph inside a numbered paragraph.

1. this is a second numbered paragraph

## Horizontal lines

Three starts on a new line gives a horizontal line.

\*\*\*

## Tables

You can display a basic table (from a data frame), e.g.

```{r}

data(iris)

iris[1:4,] # first 4 rows, all columns

iris[1:3, 2:5] # first 3 rows, columns 2 to 5

```

The knitr library has a kable function that can generate nicer tables.

```{r}

library(knitr)

kable(iris [1: 6,], caption = "Table 1: The first 6 instances of the iris dataset")

```

## Links

Links are indicated with the text inside[] immediately followed by the link inside ().

For example, earlier on we saw the R markdown resources

1. There are excellent [RMarkdown resources](https://rmarkdown.rstudio.com/) including a [tutorial](https://rmarkdown.rstudio.com/lesson-1.html).

1. The cheatsheet is available from RStudio by going to

\*\*Help -> CheatSheets -> R Markdown Cheat Sheet\*\*. You can also get it from [this link](https://raw.githubusercontent.com/rstudio/cheatsheets/master/rstudio-ide.pdf).