ETC1010: Data Modelling and Computing

Week of introduction: Rmarkdown

Professor Di Cook & Dr. Nicholas Tierney

EBS, Monash U.

2019-08-02

What is this song?

Recap

Traffic Light System

Traffic Light System

Red Post it

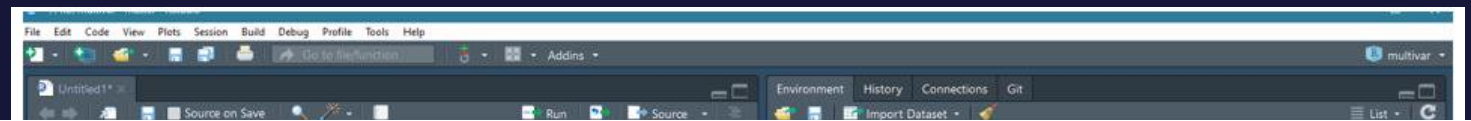
--

- I need a hand
- Slow down

Green Post it

--

- I am up to speed
- I have completed the thing


File Edit Code View Plots Session Build Debug Profile Tools Help
Addins
multivar
Untitled1*
Source on Save
Run
Source
Environment
History
Connections
Git
Import Dataset
List

R essentials: A short list (for now)

- Functions are (most often) verbs, followed by what they will be applied to in parentheses:

```
do_this(to_this)
do_that(to_this, to_that, with_those)
```

- Columns (variables) in data frames are accessed with `$`:

```
dataframe$var_name
```

- Packages are installed with the `install.packages` function and loaded with the `library` function, once per session:

```
install.packages("package_name")
library(package_name)
```

Today: Outline

- Why we care about Reproducibility
- R + markdown = Rmarkdown
- Controling output and input of rmarkdown
- Exercises on creating rmarkdown reports on the humble platypus
- Form up assignment groups
- Quiz
- Release assignment (later today)

We are in a tight spot with reproducibility

> Only 6 out of 53 landmark results could be reproduced
>
> -- Amgen, 2014*

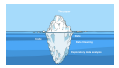
* Heard via Garret Grolemund's great talk

> An estimated 75% - 90% of preclinical results cannot be reproduced
>
> -- Begley, 2015*

* Heard via Garret Grolemund's great talk

> Estimated **annual** cost of irreproducibility for biomedical industry = 28 Billion USD
>
> -- Freedman, 2015*

* Heard via Garret Grolemund's great talk


Visible
Subconscious
Unconscious

So what can we do about it?

Reproducibility checklist

Near-term goals:

- Are the tables and figures reproducible from the code and data?
- Does the code actually do what you think it does?
- In addition to what was done, is it clear **why** it was done? (e.g., how were parameter settings chosen?)

Reproducibility checklist

Long-term goals:

- Can the code be used for other data?
- Can you extend the code to do other things?

Literate programming is a partial solution

- Literate programming shines some light on this dark area of science.
- An idea from Donald Knuth where you combine your text with your code output to create a document.
- A *blend* of your literature (**text**), and your programming (**code**), to create something you can read from top to bottom.

So

Imagine a report:

Introduction, methods, results, discussion, and conclusion,

and all the bits of code that make each section.

With rmarkdown, you can see all the pieces of your data analysis

Markdown as a new player to legibility

In 2004, John Gruber, of daring fireball created markdown, a simple way to create text that rendered into a HTML webpage.

```
- bullet list
- bullet list
- bullet list
```

- bullet list
- bullet list
- bullet list

```
1. numbered list
2. numbered list
3. numbered list

__bold__, **bold**, _italic_, *italic*

> quote of something profound
```

1. numbered list
2. numbered list
3. numbered list

bold, **bold**, *italic*, *italic*

> quote of something profound

With very little marking up, we can create rich text, that **actually resembles** the text that we want to see.

Learn to use markdown In your small groups, spend five minutes working through markdowntutorial.com

05:00

Rmarkdown helps complete the solution to the reproducibility problem

- Q: How do we take `markdown` + `R code` = "literate programming environment"
- A: `Rmarkdown`

Rmarkdown...

Provides an environment where you can write your complete analysis, and marries your text, and code together into a rich document.

You write your code as code chunks, put your text around that, and then hey presto, you have a document you can reproduce.

Reminder: You've already used rmarkdown!

How will we use R Markdown?

- Every assignment + project / is an R Markdown document
- You'll always have a template R Markdown document to start with
- The amount of scaffolding in the template will decrease over the semester
- These lecture notes are created using R Markdown (!)

The anatomy of an rmarkdown document

There are three parts to an rmarkdown document.

- Metadata (YAML)
- Text (markdown formatting)
- Code (code formatting)

DEMO

Metadata: YAML (YAML Ain't Markup Language)

- The metadata of the document tells you how it is formed - what the **title** is, what **date** to put, and other control information.
- If you're familiar with $LaTeX$, this is ksimilar to how you specify document type, styles, fonts, options, etc in the front matter / preamble.

Metadata: YAML

- Rmarkdown documents use YAML to provide the metadata. It looks like this:

```
---
title: "An example document"
author: "Nicholas Tierney"
output: html_document
---
```

It starts an ends with three dashes `---`, and has fields like the

Text

Is markdown, as we discussed in the earlier section,

It provides a simple way to mark up text

```
1. bullet list
2. bullet list
3. bullet list
```

1. bullet list
2. bullet list
3. bullet list

Code

We refer to code in an rmarkdown document in two ways:

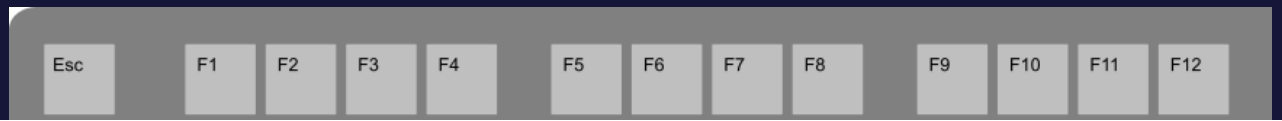
1. Code chunks, and
2. Inline code.

Code: Code chunks

`Code chunks` are marked by three backticks and curly braces with `r` inside them:

````
```{r chunk-name}
# a code chunk
```
````

a backtick is a special character you might not have seen before, it is typically located under the tilde key (~). On USA / Australia keyboards, is under the escape key:



Code: Inline code

Sometimes you want to run the code inside a sentence. This is called running the code "inline".

You might want to run the code inline to name the number of variables or rows in a dataset in a sentence like:

> There are XXX observations in the airquality dataset, and XXX variables.

Code: Inline code

You can call code "inline" like so:

```
There are `r nrow(airquality) ` observations in the airquality dataset,
and `r ncol(airquality) ` variables.
```

Which gives you the following sentence

There are 153 observations in the airquality dataset, and 6

Code: Inline code

What's great about this is that if your data changes upstream, then you don't need to work out where you mentioned your data, you just update the document.

Your Turn: Put it together

Go to `rstudio.cloud` and

- open the document "01-oz-atlas.Rmd"
- knit the document
- Change the data section at the top to be from a different state instead of "New South Wales"
- knit the document again

- How do the text and figures in the document change?

05:00

break

Code: Chunk names

Straight after the ```{r you can use a text string to name the chunk:

```{r read-crime-data}

````
```{r read-crime-data}
crime <- read_csv("data/crime-data.csv")
```
````
````

Code: Chunk Names

Naming code chunks has three advantages:

1. Navigate to specific chunks using the drop-down code navigator in the bottom-left of the script editor.
2. Graphics produced by chunks now have useful names.
3. You can set up networks of cached chunks to avoid re-performing expensive computations on every run.

Code: Chunk names

- Every chunk should ideally have a name.
- Naming things is hard, but follow these rules and you'll be fine:

1. One word that describes the action (e.g., "read")
2. One word that describes the thing inside the code (e.g, "gapminder")
3. Separate words with "-" (e.g., `read-gapminder`)

Code: Chunk options

You can control how the code is output by changing the code chunk options which follow the title.

````
```{r read-gapminder, eval = FALSE, echo = TRUE}
gap <- read_csv("gapminder.csv")
```
````

What do you think this does?

00:30

Code: Chunk options

The code chunk options you need to know about right now are:

- `cache`: TRUE / FALSE. Do you want to save the output of the chunk so it doesn't have to run next time?
- `eval`: TRUE / FALSE Do you want to evaluate the code?
- `echo`: TRUE / FALSE Do you want to print the code?
- `include`: TRUE / FALSE Do you want to include code output in the final output document? Setting to `FALSE` means nothing is put into the output document, but the code is still run.

You can read more about the options at the official documentation: https://yihui.name/knitr/options/#code-evaluation

Your turn

- go to `rstudio.cloud`, open document `01-oz-atlas.Rmd` and change the document so that the code output is hidden, but the graphics are shown. (Hint: Google "rstudio rmarkdown cheatsheet" for some tips!)
- Re-Knit the document.
- Take a look at the R Markdown Gallery.

05:00

Global options: Set and forget

You can set the default chunk behaviour once at the top of the `.Rmd` file using a chunk like:

```
knitr::opts_chunk$set(
  echo = FALSE,
  cache = TRUE
)
```

then you will only need to add chunk options when you have the

Your turn

- Go to your `01-oz-atlas.Rmd` document on `rstudio.cloud` and change the global settings at the top of the rmarkdown document to `echo = FALSE`, and `cache = TRUE`
- Update the other code chunks by removing the code chunk options.

03:00

DEMO

The many different outputs of rmarkdown

Your turn: Different types of documents

1. Change the output of your current R Markdown file to produce a **Word document**. Now try to produce pdf - this may not work! That's OK, we do'nt need it right now.
2. Create a new document that will produce a slide show `File > New R Markdown > Presentation`
3. Create a flexdashboard document - see this option in the `File > New R Markdown > From template` list.

Your Turn: Making the groups

We are going to set up the groups for doing assignment work.

1. Choose a quote from the bag.
2. Find the other people in the class with the same quote as you
3. Grab your gear and claim a table to work together at.

Your Turn: Ask your team mates these questions:

1. What is one food you'd never want to taste again?
2. If you were a comic strip character, who would you be and why?

LASTLY, come up with a name for your team and tell this to a tutor,

05:00

Your Turn

- Go to rstudio.cloud to `oz-atlas-final.Rmd`
- Read through the document and add text where prompted to learn more about the Australian native platypus!

Recap:

- There is a Reproducibility Crisis
- rmarkdown = YAML + text + code
- rmarkdown has many different output types
- Platypus are interesting!
- Assignment will be announced later today

Learning more:

- R Markdown cheat sheet and Markdown Quick Reference (Help -> Markdown Quick Reference) handy, we'll refer to it often as the course progresses

Lab quiz

Take the quiz for today from ED.

Share and share alike