

TDU.

INTRODUCTION TO R

Graphics and Automated Reports

Day 2



Public Health
Agency of Canada

Agence de la santé
publique du Canada

COURSE LEARNING OBJECTIVES FROM DAY 1

COURSE LEARNING OBJECTIVES

- By the end of this course, learners will be able to:

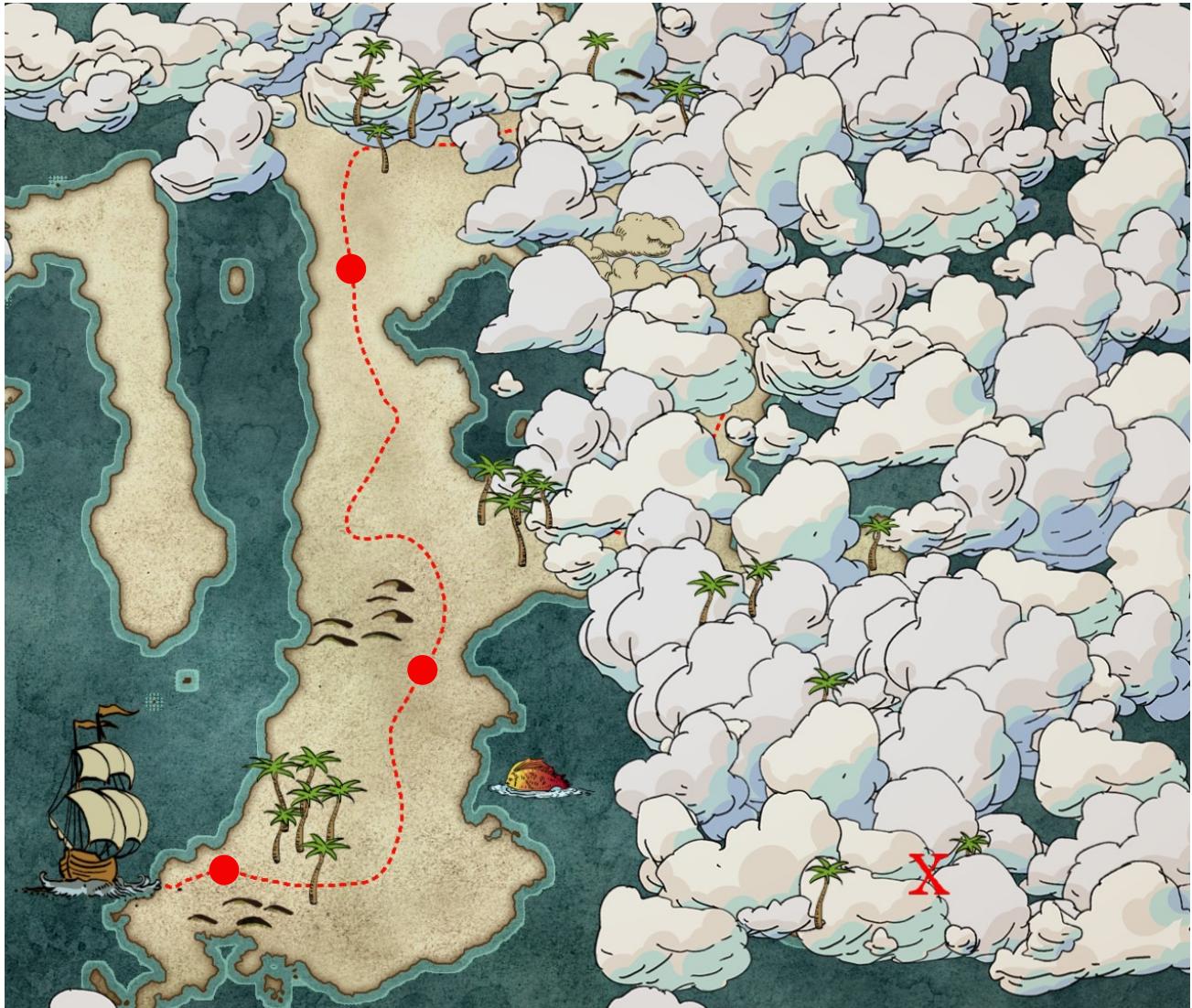


- Carry out data cleaning and processing, and descriptive epidemiological analyses (including commonly used data visualizations);
- Create automated data products (e.g., epidemiologic summaries);
- Design and carry out a data collation plan that is consistent with proposed analysis plan;
- Explain when it is most appropriate to program analyses and automates tasks using R;
- Find and appraise possible solutions to R programming challenges



THE MAP

- In Day 1, we discovered:
 - “The coast of core-functions”;
 - “Syntax Sound”;
 - “Clean-Data Caves”
- In Day 2, we will continue our quest along
 - “Markdown Beach”;
 - “Automation Bay”;
 - “Visualization Valley”;
 - “Troubleshooting Trail”



DAY 2: PRE-COURSE LEARNING

Activity	Description
Context	TB 101
Foundational Concepts	Tidygraph
Foundational Concepts	Flextable
Refresher	Pre-Course Self-Study Module

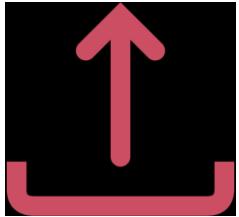
SCAVENGER HUNT

Can you spot all instances of our piRate scout in this deck?

Note the slide number on which our friend appears and what important concept is being discussed on this task, you will unlock a special seal on your R quest.

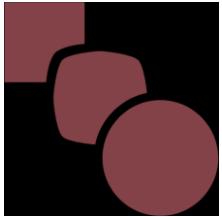


10 Considerations for Data Analysis



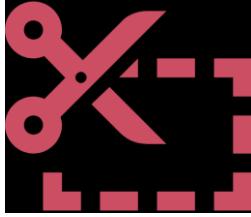
1. Read data into R

Consider (1) the type of data, (2) how they are stored, and (3) the size of the dataset.



2. Format the data

Long or wide format? Plan the analysis keeping in mind what your needs are.



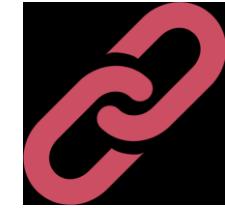
3. Select data

Decide what data you'll need in advance from the dataset: subsets vs. filtered dataset. Consider pros and cons of the approach.



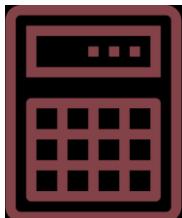
4. Modify data

Consider (1) what variables you will need, (2) calculations you will perform, and (3) algorithms needed to create the variables.



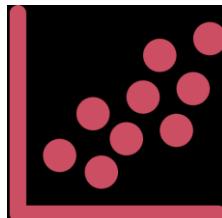
5. Link data

Assign a key variable(s) that you will link datasets on. Consider the join type you will need: inner, left, right, full, semi, or anti join.



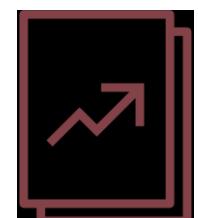
6. Analyse data

Consider grouping data meaningfully in performing calculations and obtaining descriptive statistics. Evaluate outputs and interpret summaries.



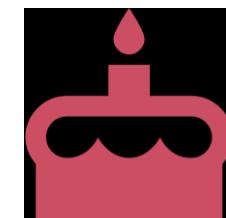
7. Visualise data

Use tables or charts, and consider the best way to display the information.



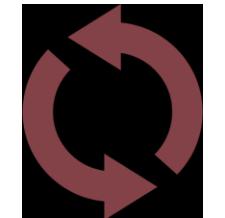
8. Report findings

Communicate findings to those who need the information created from your work. Consider the audience you are communicating with.



9. Celebrate!

Preferably with cake!



10. Review

Review, revise, and re-do as needed. Often we can go back and simplify/optimise. This is important especially if the code will be used again.

EXERCISE 1 – REFLECTIVE QUESTIONS

- Who reproduced their own code from course materials? Who wrote their own? Who approached the provided code more conceptually, as a translation exercise?
- Was there anything that you found to be particularly challenging?
- Were there any interesting discoveries you made while working on this?
- Any further questions or comments?

WHAT WE HEARD

- “Homework” versus “Exercises”
- More resources on **here()** package and R-Projects in the chat
 - If you run into issues, try running `here()` to have R tell you where your project is anchored!



ON TO THE NEXT QUEST IN OUR ADVENTURE



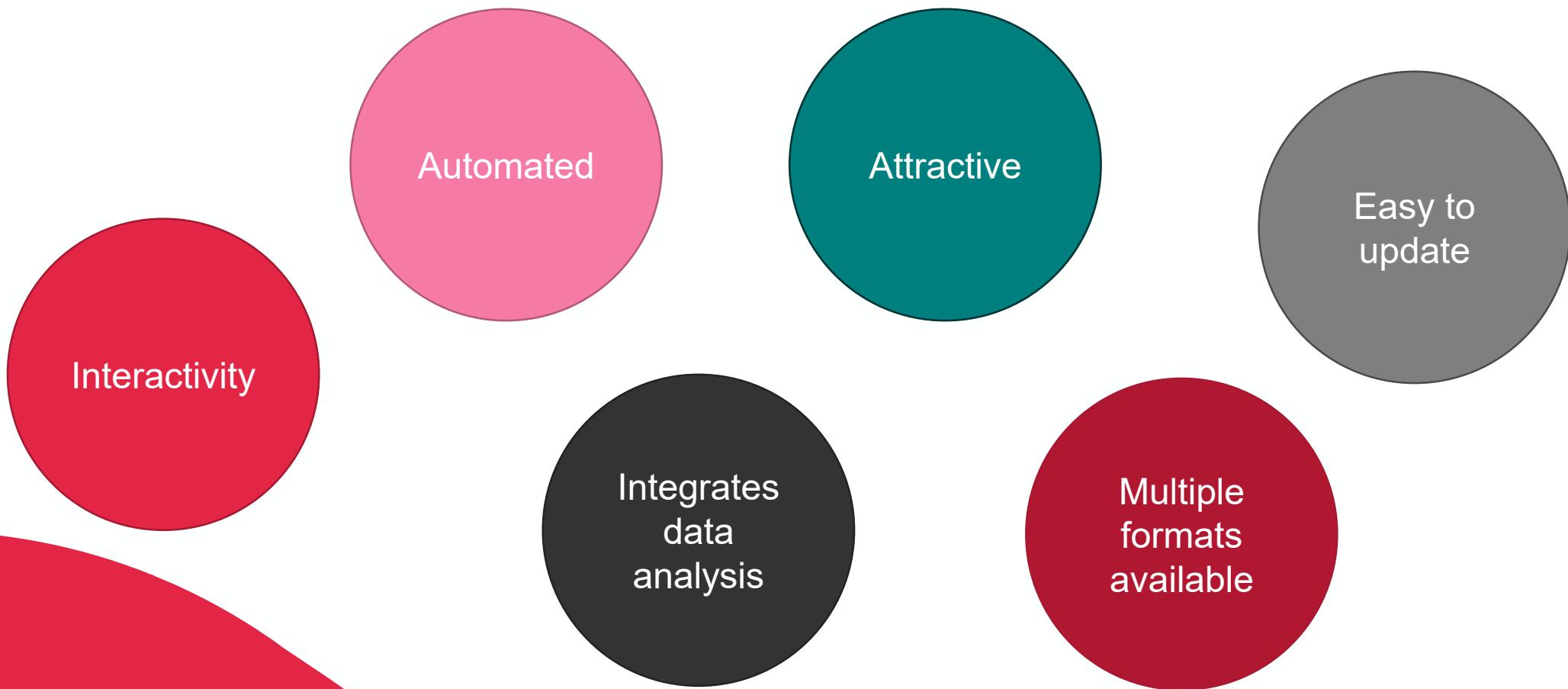
REPORTING AND VISUALS

Interpreting data to help make a decision

R MARKDOWN

REPORTING IN A ‘PERFECT-WORLD’

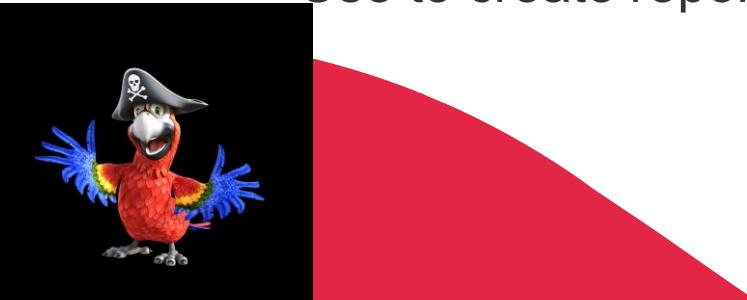
- Use the annotation tool to put a stamp on which of the following features of a reporting tool are important to you!



R MARKDOWN – WHAT IS IT?



- A tidyverse package: rmarkdown
- An R markdown file (.Rmd) is a file format that creates documents (static or dynamic)
- Markdown is a formatting language that allows for documents to be written in plain text with formatting cues
 - “Chunks” of R code can be written directly in the markdown document or existing R scripts can be called or sourced using the source() function and specifying the file pathway
 - Narrative text can be added and formatted nicely
 - Use to create reports and other data products in various formats



R MARKDOWN

- Three main components of an R markdown file (.Rmd):
 - Metadata
 - YAML header
 - Written between “---”
 - Text
 - Markdown syntax
 - Code
 - R code chunks

File | ABC | Knit | Insert | Run |

```
---
```

```
title: "This is my R markdown document"
author: "J. Stares"
date: "2021-01-25"
output: html_document
---
```

This is the text in my R markdown document. After this section, I will include a code chunk:

```
```{r}
#this is the first code chunk in my R markdown document
library(dplyr)
print(cars)
```

...

speed <dbl>	dist <dbl>
4	2
4	10
7	4
7	22
8	16
9	10
10	18
10	26
10	34
11	17

1-10 of 50 rows

Previous 1 2 3 4 5 Next

15

---  
title: "This is my R markdown document"  
author: "J. Stares"  
date: "2021-01-25"  
output: html\_document  
---

This is the text in my R markdown document. A code chunk:

```
```{r}  
#this is the first code chunk in my R markdown  
library(dplyr)  
print(cars)  
```
```

| speed<br><dbl> | dist<br><dbl> |
|----------------|---------------|
| 4              | 2             |
| 4              | 10            |
| 7              | 4             |
| 7              | 22            |
| 8              | 16            |
| 9              | 10            |
| 10             | 18            |
| 10             | 26            |
| 10             | 34            |
| 11             | 17            |

1-10 of 50 rows

Previous 1 2 3 4 5 Next

Run Selected Line(s) Ctrl+Enter

Run Current Chunk Ctrl+Shift+F10 or selection

Run Next Chunk Ctrl+Alt+N

Run Setup Chunk

Run Setup Chunk Automatically

Run All Chunks Above Ctrl+Alt+P

Run All Chunks Below

Restart R and Run All Chunks

Restart R and Clear Output

Run All Ctrl+Alt+R

# This is my R markdown document

J. Stares

2021-01-25

This is the text in my R markdown document. After this section, I will include a code chunk:

```
#this is the first code chunk in my R markdown document
library(dplyr)
```

```
Warning: package 'dplyr' was built under R version 4.0.3
```

```

Attaching package: 'dplyr'
```

```
The following objects are masked from 'package:stats':

filter, lag
```

```
The following objects are masked from 'package:base':

intersect, setdiff, setequal, union
```

```
print(cars)
```

```
speed dist
1 4 2
2 4 10
```



# R Markdown Reference Guide

Learn more about R Markdown at [rmarkdown.rstudio.com](http://rmarkdown.rstudio.com)

Learn more about Interactive Docs at [shiny.rstudio.com/articles](http://shiny.rstudio.com/articles)

Contents:

1. **Markdown Syntax**
2. Knitr chunk options
3. Pandoc options

## Syntax

Plain text

End a line with two spaces  
to start a new paragraph.

\*italics\* and \_italics\_

\*\*bold\*\* and \_\_bold\_\_

superscript<sup>2</sup>

~~strikethrough~~

[link](www.rstudio.com)

# Header 1

## Header 2

### Header 3

#### Header 4

##### Header 5

##### Header 6

endash: --

emdash: ---

ellipsis: ...

inline equation: \$A = \pi \* r^2\$

image: 

## Becomes

Plain text

End a line with two spaces to start a new paragraph.

*italics* and *italics*

**bold** and **bold**

superscript<sup>2</sup>

strikethrough

link

## Header 1

## Header 2

## Header 3

## Header 4

## Header 5

## Header 6

endash: –

emdash: —

ellipsis: ...

inline equation:  $A = \pi * r^2$

image:

# R

# R Markdown

## Reference Guide

Learn more about R Markdown at [rmarkdown.rstudio.com](https://rmarkdown.rstudio.com)

Learn more about Interactive Docs at [shiny.rstudio.com/articles](https://shiny.rstudio.com/articles)

### Contents:

- 1. Markdown Syntax**
2. Knitr chunk options
3. Pandoc options

### Syntax

\*\*\*

> block quote

\* unordered list  
\* item 2  
  + sub-item  
  + sub-item

1. ordered list  
2. item 2  
  + sub-item  
  + sub-item

Table Header

Table Cell

Cell 3



Second Header

Cell 2

Cell 4

### Becomes

block quote

- unordered list
- item 2
  - sub-item 1
  - sub-item 2

1. ordered list
2. item 2
  - sub-item 1
  - sub-item 2

Table Header

Table Cell

Cell 3

Second Header

Cell 2

Cell 4



# R Markdown Reference Guide

Learn more about R Markdown at [rmarkdown.rstudio.com](http://rmarkdown.rstudio.com)

Learn more about Interactive Docs at [shiny.rstudio.com/articles](http://shiny.rstudio.com/articles)

## Contents:

1. Markdown Syntax
- 2. Knitr chunk options**
3. Pandoc options

## Syntax

Make a code chunk with three back ticks followed by an r in braces. End the chunk with three back ticks:

```
```{r}
paste("Hello", "World!")
```
```

Place code inline with a single back ticks. The first back tick must be followed by an R, like this `r paste("Hello", "World!")`.

Add chunk options within braces. For example, `echo=FALSE` will prevent source code from being displayed:

```
```{r eval=TRUE, echo=FALSE}
paste("Hello", "World!")
```
```

## Becomes

Make a code chunk with three back ticks followed by an r in braces. End the chunk with three back ticks:

```
paste("Hello", "World!")

[1] "Hello World!"
```

Place code inline with a single back ticks. The first back tick must be followed by an R, like this Hello World!.

Add chunk options within braces. For example, `echo=FALSE` will prevent source code from being displayed:

```
[1] "Hello World!"
```

Learn more about chunk options at <http://yihui.name/knitr/options>

## Chunk options

| option                 | default value | description                                                                                                                                                                                                                                                                     |
|------------------------|---------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Code evaluation</b> |               |                                                                                                                                                                                                                                                                                 |
| <b>child</b>           | NULL          | A character vector of filenames. Knitr will knit the files and place them into the main document.                                                                                                                                                                               |
| <b>code</b>            | NULL          | Set to R code. Knitr will replace the code in the chunk with the code in the code option.                                                                                                                                                                                       |
| <b>engine</b>          | 'R'           | Knitr will evaluate the chunk in the named language, e.g. <code>engine = 'python'</code> . Run <code>names(knitr::knit_engines\$get())</code> to see supported languages.                                                                                                       |
| <b>eval</b>            | TRUE          | If FALSE, knitr will not run the code in the code chunk.                                                                                                                                                                                                                        |
| <b>include</b>         | TRUE          | If FALSE, knitr will run the chunk but not include the chunk in the final document.                                                                                                                                                                                             |
| <b>purl</b>            | TRUE          | If FALSE, knitr will not include the chunk when running <code>purl()</code> to extract the source code.                                                                                                                                                                         |
| <b>Results</b>         |               |                                                                                                                                                                                                                                                                                 |
| <b>collapse</b>        | FALSE         | If TRUE, knitr will collapse all the source and output blocks created by the chunk into a single block.                                                                                                                                                                         |
| <b>echo</b>            | TRUE          | If FALSE, knitr will not display the code in the code chunk above it's results in the final document.                                                                                                                                                                           |
| <b>results</b>         | 'markup'      | If 'hide', knitr will not display the code's results in the final document. If 'hold', knitr will delay displaying all output pieces until the end of the chunk. If 'asis', knitr will pass through results without reformatting them (useful if results return raw HTML, etc.) |
| <b>error</b>           | TRUE          | If FALSE, knitr will not display any error messages generated by the code.                                                                                                                                                                                                      |
| <b>message</b>         | TRUE          | If FALSE, knitr will not display any messages generated by the code.                                                                                                                                                                                                            |
| <b>warning</b>         | TRUE          | If FALSE, knitr will not display any warning messages generated by the code.                                                                                                                                                                                                    |
| <b>Code Decoration</b> |               |                                                                                                                                                                                                                                                                                 |
| <b>comment</b>         | '##'          | A character string. Knitr will append the string to the start of each line of results in the final document.                                                                                                                                                                    |
| <b>highlight</b>       | TRUE          | If TRUE, knitr will highlight the source code in the final output.                                                                                                                                                                                                              |
| <b>prompt</b>          | FALSE         | If TRUE, knitr will add > to the start of each line of code displayed in the final document.                                                                                                                                                                                    |
| <b>strip.white</b>     | TRUE          | If TRUE, knitr will remove white spaces that appear at the beginning or end of a code chunk.                                                                                                                                                                                    |
| <b>tidy</b>            | FALSE         | If TRUE, knitr will tidy code chunks for display with the <code>tidy_source()</code> function in the <code>formatR</code> package.                                                                                                                                              |

# R MARKDOWN – LET'S REVIEW

- What sorts of documents can be created using R markdown?
  - Pdf, word, html, ppt, dashboard, webpages and more
- Is the # needed to add narrative text to your R markdown file?
  - No, the # isn't needed to add narrative text – it is used for formatting
  - Note # still required to comment R code within code “chunks”
- Why would you use R markdown?
  - Reproduce your work at the click of a button, export results as a report or link to dashboard, create a transparent record of your work, etc.

# R MARKDOWN – LET'S REVIEW

- What is the header used to open a chunk of R code in a markdown file?
  - ``{r}
- How do you suppress printing of R code in your rendered file?
  - In the code chunk header: echo=FALSE  
````{r, echo=FALSE}`
- How do you create an ordered list in a markdown document?
 - Just number the text:
 1. Item 1
 2. Item 2, etc.
- How do you render and export your document?
 - Knit the document:

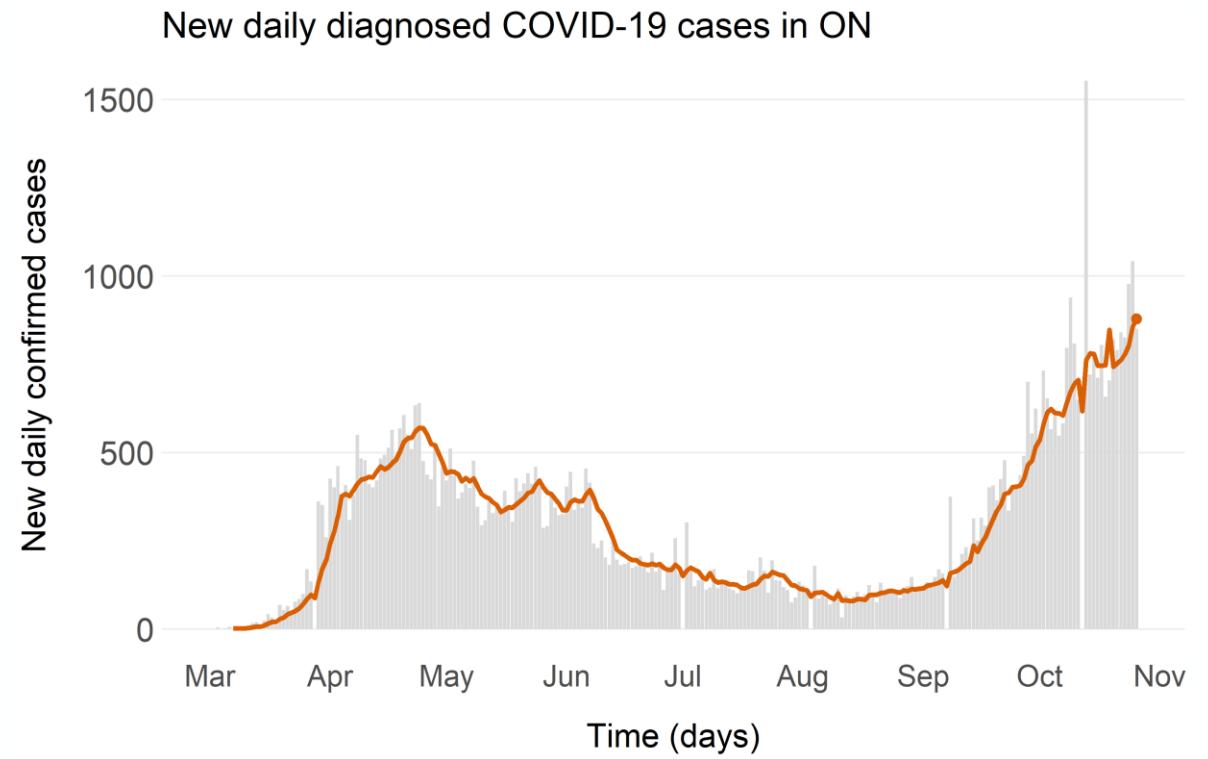


DATA VISUALIZATION

Very, very, very briefly

DATA VISUALISATION

- Recall Exercise 1 – ggplot2:
 - Bar chart: geom_bar()
 - Line graph: geom_line()
 - Point: geom_point()
 - New cases (smoothed) for the maximum date in the dataset

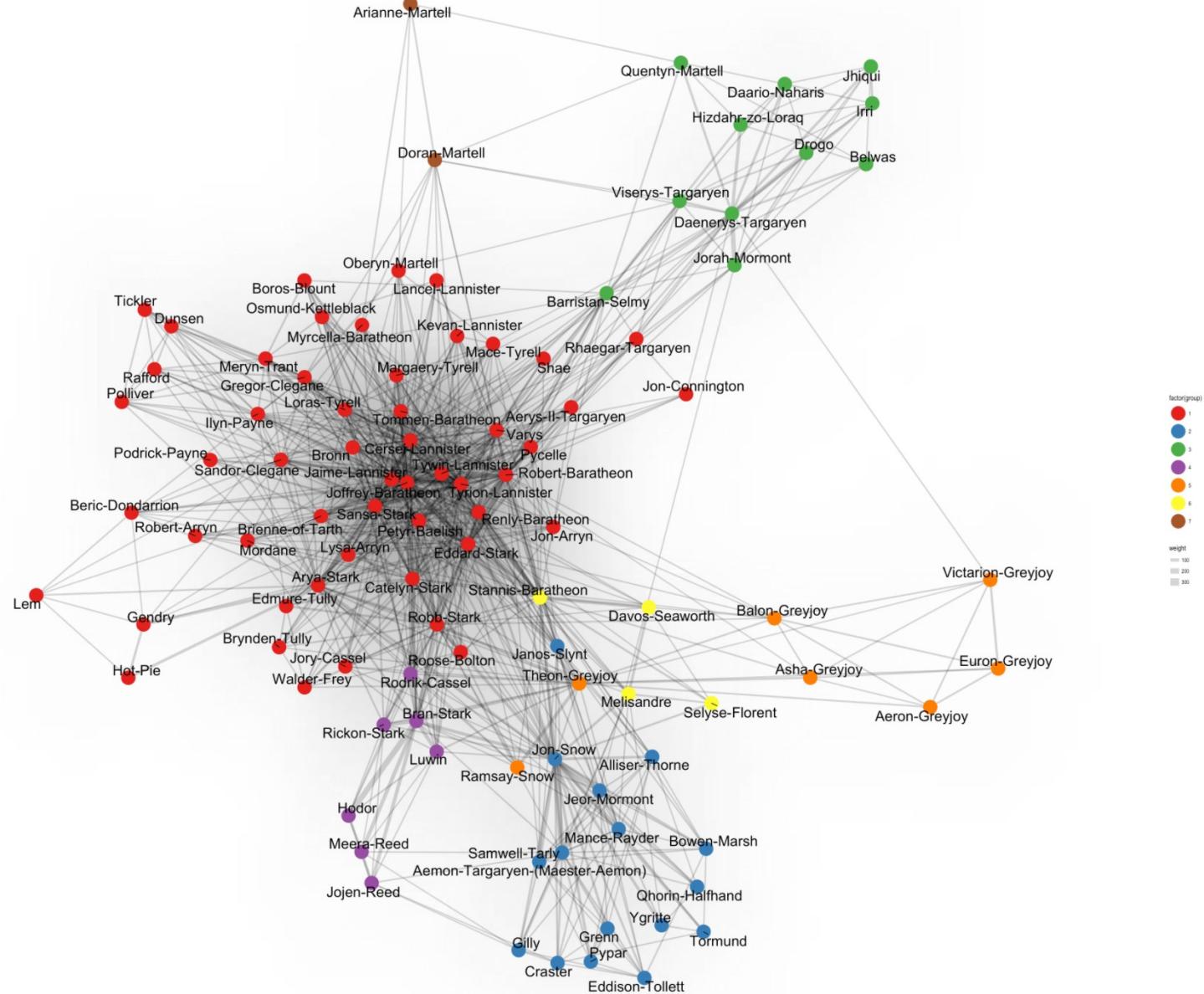




Code for all these and more: <http://r-statistics.co/Top50-Ggplot2-Visualizations-MasterList-R-Code.html#Hierarchical%20Dendrogram>

DATA VISUALISATION – SOCIAL NETWORK DIAGRAM

- Use in person-to-person outbreaks
 - Social network diagram
 - Epidemic curve
 - Other?



LET'S REVIEW SOME SNA TERMINOLOGY

TIDYGRAPH

- Package that uses the social network analytics capabilities of another package (igraph) but allows use of tidyverse language and structure
 - Can use igraph itself, as well as other packages including epicontacts
- Need two different data elements:
 - 1.
 - 2.

SOCIAL NETWORK DIAGRAM - TIDYGRAPH

- In an outbreak situation, what would the nodes represent?
 - Cases and contacts
 - As a dataframe also include demographic information and case classification
 - 1 row per person
- What would the edges represent?
 - Relationships between cases and contacts
 - As a dataframe also include characterisations of that relationship (e.g., household, workplace, etc.)
 - 1 row per person

SOCIAL NETWORK DIAGRAM - TIDYGRAPH

- After processing your case and contact data to obtain edge and node dataframes
- Using ***tidygraph***:
 - Apply the *tbl_graph()* function to combine node and edge dataframes into a tidygraph object.

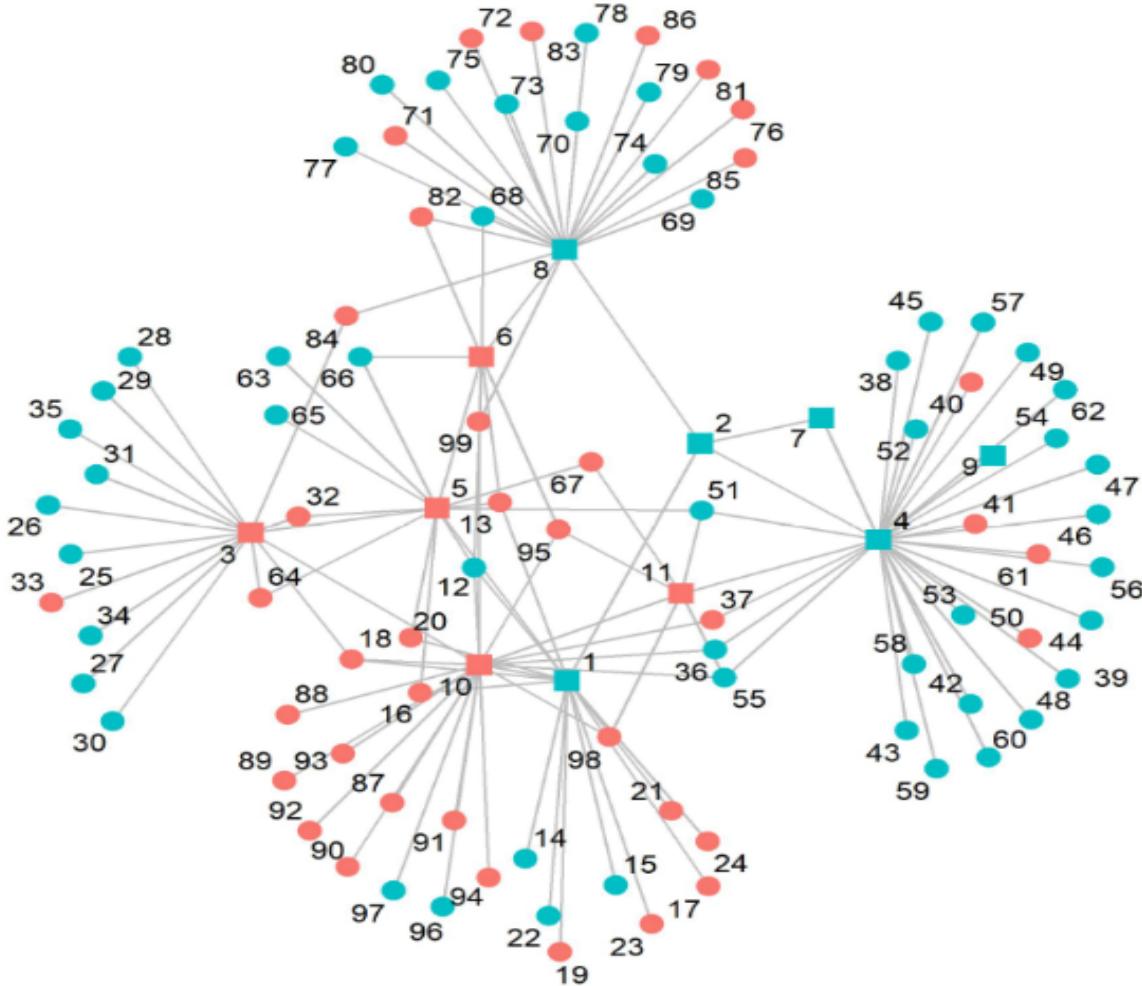
```
network_full <- tbl_graph(nodes = nodes_full,  
                           edges = edges_full, directed = FALSE)
```

SOCIAL NETWORK DIAGRAM - TIDYGRAPH

- After combining edge and node dataframes
- Using *tidygraph*:
 - Create network plot object using *ggraph*
 - Note the similarity in format / structure to *ggplot2*.
 - Format node & edge shape, colour, size according to various attributes.
 - Apply themes that have already been created.

```
network_full %>%
  ggraph() +
  geom_edge_link0(color = "#bdbdbd") +
  geom_node_point(aes(colour = Location, shape = classification),
                  alpha = 1,
                  size = 4) +
  scale_fill_brewer(type = "qual",
                    palette = 5) +
  geom_node_text(label = label_full,
                 repel = TRUE,
                 point.padding = 0.1,
                 segment.color = NA) +
  theme(panel.background = element_blank(),
        plot.title = element_text(size = 20),
        legend.position = "bottom") +
  scale_shape_manual(values = c(15,16))
```

SOCIAL NETWORK DIAGRAM - TIDYGRAPH



BREAK-TRIVIA: KNOW YOUR KNOTS!

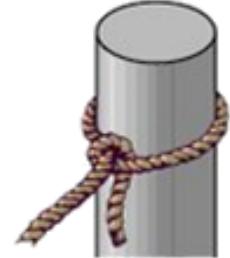
A



B



C



D



E



F



1. Sheepshank

2. Cat's Paw

3. Overhand

4. Half-hitch

5. Slipknot

6. Fisherman's Bend

TABLES, SOURCING CODE, AND TROUBLESHOOTING

Ways to catch wind in your sails so that you reach your destination quickly and safely

TABLES FOR DESCRIPTIVE EPIDEMIOLOGY



- Descriptive epidemiology: tables
- The `flextable` package provides a way to easily create nicely formatted tables of report and publication quality
- Embed tables in R markdown documents or render as graphic files
- Workhorse function: `flextable()`
 - Use dataframe containing the data as the object

FLEXTABLE

- Formatting the table



- Default

- Prebuilt themes

- e.g., `theme_vanilla()` or `theme_vader()`

- Custom design of themes

| measures | | | | | | time |
|----------|---------|------|------|-------|-----|------|
| Ozone | Solar.R | Wind | Temp | Month | Day | |
| 41 | 190 | 7.4 | 67 | 5 | 1 | |
| 36 | 118 | 8.0 | 72 | 5 | 2 | |
| 12 | 149 | 12.6 | 74 | 5 | 3 | |
| 18 | 313 | 11.5 | 62 | 5 | 4 | |
| | | 14.3 | 56 | 5 | 5 | |
| 28 | | 14.9 | 66 | 5 | 6 | |

| measures | | | | | | time |
|----------|---------|------|------|-------|-----|------|
| Ozone | Solar.R | Wind | Temp | Month | Day | |
| 41 | 190 | 7.4 | 67 | 5 | 1 | |
| 36 | 118 | 8.0 | 72 | 5 | 2 | |
| 12 | 149 | 12.6 | 74 | 5 | 3 | |
| 18 | 313 | 11.5 | 62 | 5 | 4 | |
| | | 14.3 | 56 | 5 | 5 | |
| 28 | | 14.9 | 66 | 5 | 6 | |

FLEXTABLE

- Other useful functions:
 - `font()` → specify font
 - `fontsize()` → specify font size
 - `bold()` → bold face text
 - `italic()` → italic face text
 - `bg()` → background colour of table/table section
 - `colour()` → font colour
 - `align()` → text alignment (e.g., centre)
 - `autofit()` → automatically adjusts table size to fit the size of table contents
 - `rotate()` → rotate text (i.e., display vertically)
 - `set_header_labels()` → Specify labels in table header (default is column name)
 - `add_header_row()` and `add_footer_row()` → add a single row of labels to display among one or more columns
 - `fix_border_issues()` → fixes rendered borders when cells are merged
 - `set_caption()` → specify a title for your table
 - `footnote()` → add footnote to table

FLEXTABLE - EXAMPLE

```
tab_case_site_infection <- flextable(case_site_infection) %>%  
  bg(bg = "#E6E6E6", part = "header") %>%  
  bold(part = "header") %>%  
  fontsize(size = 10, part = "all") %>%  
  font(part = "all", fontname = "Arial") %>%  
  align(align = "center", part = "all") %>%  
  set_header_labels(site_infection = "Site of infection" ) %>%  
  align(j=c("Site_infection"), align = "left") %>%  
  bold( i = 6, bold = TRUE, part = "body") %>%  
  width(j = 1, width = 1.5) %>%  
  set_caption(" Site of TB infection", style = "Table Caption")  
  
tab_case_site_infection
```



| Site of TB infection | |
|----------------------|-------|
| Site of infection | Count |
| Abdominal | 1 |
| Meningeal | 1 |
| Miliary | 1 |
| Pleural | 4 |
| Pulmonary TB | 4 |
| Total | 11 |

FLEXTABLE - EXAMPLE

```
tab_case_site_infection <- flextable(case_site_infection) %>%  
  bg(bg = "#E6E6E6", part = "header") %>%  
  bold(part = "header") %>%  
  fontsize(size = 10, part = "all") %>%  
  font(part = "all", fontname = "Arial") %>%  
  align(align = "center", part = "all") %>%  
  set_header_labels(site_infection = "Site of infection" ) %>%  
  align(j=c("Site of infection"), align = "left") %>%  
  bold( i = 6, bold = TRUE, part = "body") %>%  
  width(j = 1, width = 1.5) %>%  
  set_caption(" Site of TB infection", style = "Table Caption")  
  
tab_case_site_infection
```

| Site of TB infection | |
|----------------------|-------|
| Site of infection | Count |
| Abdominal | 1 |
| Meningeal | 1 |
| Miliary | 1 |
| Pleural | 4 |
| Pulmonary TB | 4 |
| Total | 11 |

FLEXTABLE - EXAMPLE

```
tab_case_site_infection <- flextable(case_site_infection) %>%  
  bg(bg = "#E6E6E6", part = "header") %>%  
  bold(part = "header") %>%  
  fontsize(size = 10, part = "all") %>%  
  font(part = "all", fontname = "Arial") %>%  
  align(align = "center", part = "all") %>%  
  set_header_labels(site_infection = "Site of infection" ) %>%  
  align(j=c("Site_infection"), align = "left") %>%  
  bold(i = 6, bold = TRUE, part = "body") %>%  
  width(j = 1, width = 1.5) %>%  
  set_caption(" Site of TB infection", style = "Table Caption")  
  
tab_case_site_infection
```

| Site of TB infection | |
|----------------------|-------|
| Site of infection | Count |
| Abdominal | 1 |
| Meningeal | 1 |
| Miliary | 1 |
| Pleural | 4 |
| Pulmonary TB | 4 |
| Total | 11 |

SOURCING CODE

Programming for larger projects

SOURCING CODE

- The `source()` function: calls and runs code from another script
- Useful for setting up complex projects
- Aides in debugging when re-running s
- Final version of scripts



SOURCING CODE

```
source(here("scripts", "01_define_paths.R"))
source(here("scripts", "02_load_libraries.R"))
source(here("scripts", "03_import_data.R"))
source(here("scripts", "04_clean_data.R"))
source(here("scripts", "05_summarise_data.R"))
source(here("scripts", "06_visualize_data.R"))
source(here("scripts", "07_eat_cake.R"))
```

SOURCING CODE: GROUP DISCUSSION

ADVANTAGES

- Can you think of any advantages to setting up a project by creating several smaller scripts and then calling them/sourcing them in another?

DISADVANTAGES

- Can you think of any disadvantages to setting up a project by creating several smaller scripts and then calling them/sourcing them in another?

Advantages

Disadvantages

SOURCING CODE

ADVANTAGES

- Organised workflow for complex projects
- Aides in debugging
 - i.e., easy to see what section is broken
- Aides in replication of work

DISADVANTAGES

- Requires time to go back and optimise your code
- Extra time to ensure that code is aligned and runs smoothly when compiled
- Requires extra effort in ensuring that details regarding your system are communicated in an accessible way

BREAK



- Take a few minutes for yourself
 - Need a mindfulness activity suggestion?
 - Build yourself a flotilla of paper ships!

TROUBLESHOOTING

Common errors, tips, and tricks

TROUBLESHOOTING - ERRORS

- RStudio helps to identify basic problems in your code:



34



35

```
=install.packages("readr")  
install.packages("readxl")
```

36

TROUBLESHOOTING - ERRORS

1. Error in library(X) : there is no package called 'X'
 2. Error in X() : could not find function "X"
 3. Error in function(X) : object 'X' not found
 4. Error : unexpected 'X' in [line of code]
 5. Error : 'X' does not exist in current working directory [pathway]
- A. X is an error in a line of code - revisit that line to see what the issue is
 - B. X is a package that has not been installed – install the package or check for typos
 - C. X is a file that is not located where your line of code said that it would be – ensure that the file pathway is correct and check for typos
 - D. X is an object (e.g., a dataframe) that doesn't exist – check for typos or order in which you've run your code (e.g., did you clear your workspace?)
 - E. X is a function – check to make sure the right packages have been installed and loaded for this session and check for typos

TROUBLESHOOTING - ERRORS

- Error in [line of code] : undefined columns selected
 - Confirm that your reference to a particular variable in your code is correct, whether you are indexing or calling by name
- Error in [line of code] : replacement has [x] rows, data has [y]
 - Depends on what you are doing: confirm that the steps of what you are doing are reflected accurately in the code (E.g., did you create the variable you are trying to assign values to)
- Error in [line of code] : all arguments must have the same length
 - Depends on what you're doing: try addressing NA's specifically in function options

TROUBLESHOOTING - APPROACH

- Start writing code in short pieces – one line does one thing
- Make note of changes to the data as you process it to ensure that the code is doing what it is expected to do (e.g., counts of records and variables, cross tabulations)
 - `length()`, `count()`, `table()`, `str()`, etc.
- Once your simple code works, consider:
 - Chaining statements together `%>%`
 - Recycling for similar tasks
 - Advanced: building functions, loops

TROUBLESHOOTING - TIPS

- Check spelling
 - Note: R is case sensitive
- Check correct referencing of variables and datasets
- Confirm that variables (and other objects) are being treated according to their type
 - Did your numeric data load as character for some reason?
 - Are you working with functions for dataframes, but your data are currently stored as a list?
 - Are there trailing whitespaces or non printable characters that came along with it?

TROUBLESHOOTING - TIPS

- Become comfortable translating code to your preferred spoken language through practice - what the functions do, what precisely is happening in relation to objects and data
 - Then you will be in a position to adapt code that is shared with you or that you find
- Start your search statement with the software name
 - Incl. version, package, function, etc. as applicable
- Follow-up with question, function, or error message
 - In relation to error messages, they are sometimes long and in such cases it may be most useful to take the more generic portions of the returned message



TROUBLESHOOTING - RESOURCES

- <https://stackoverflow.com/>
- <https://www.rdocumentation.org>
- <https://www.tidyverse.org/>
- <https://www.dummies.com/programming/r/r-for-dummies-cheat-sheet/>
- <http://www.cookbook-r.com/>
- <https://www.reconlearn.org/>

TROUBLESHOOTING - TIPS

- Identify and make note of sources/websites that keep popping up that have useful information
- Don't hesitate to ask for help if you cannot find or think of a solution to your problem
- Practice, be patient with yourself as you learn, and be open to experimentation - troubleshooting skills are acquired over time



R Markdown

Refer
Learn more about R Markdown at [rmarkdown](#)
Learn more about Interactive Docs at [shiny.rstudio.com](#)

Syntax

Plain text

End a line with two `---`

Basics

Geoms

Use a geom function to represent data points, use the geom's aesthetic properties to represent variables.
Each function returns a layer.

Data Transformation with dplyr :: CHEAT SHEET

dplyr functions work with pipes and expect **tidy data**. In tidy data:



Each **variable** is in its own **column**



Each **observation**, or **case**, is in its own **row**



`x %>% f(y)` becomes `f(x, y)`

Summarise Cases

These apply **summary functions** to columns to create a new table of summary statistics. Summary functions take vectors as input and return one value (see back).

- **summary function**
- **summarise(.data, ...)**
Compute table of summaries.
`summarise(mtcars, avg = mean(mpg))`
- **count(x, ..., wt = NULL, sort = FALSE)**
Count number of rows in each group defined by the variables in ... Also **tally()**.
`count(iris, Species)`

VARIATIONS

summarise_all() - Apply funs to every column.
summarise_at() - Apply funs to specific columns.
summarise_if() - Apply funs to all cols of one type.

Group Cases

R Markdown :: CHEAT SHEET

What is R Markdown?



.rmd Structure

YAML Header
Optional section of render (e.g. pandoc)
Additional sections or documents via YAML



Syntax

Plain text

End a line with two `---`

Data Visualization with ggplot2 :: CHEAT SHEET

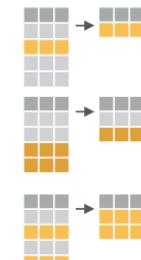


Data Transformation with dplyr :: CHEAT SHEET

Manipulate Cases

EXTRACT CASES

Row functions return a subset of rows as a new table.



`filter(.data, ...)` Extract rows that meet logical criteria. `filter(iris, Sepal.Length > 7)`



`distinct(.data, ..., .keep_all = FALSE)` Remove rows with duplicate values.
`distinct(iris, Species)`

`sample_frac(tbl, size = 1, replace = FALSE, weight = NULL, .env = parent.frame())` Randomly select fraction of rows.
`sample_frac(iris, 0.5, replace = TRUE)`

`sample_n(tbl, size, replace = FALSE, weight = NULL, .env = parent.frame())` Randomly select size rows.
`sample_n(iris, 10, replace = TRUE)`

`slice(.data, ...)` Select rows by position.
`slice(iris, 10:15)`

`top_n(x, n, wt)` Select and order top n entries (by group if grouped data).
`top_n(iris, 5, Sepal.Width)`

Logical and boolean operators to use with filter()

< <= is.na() %in% | xor()
> >= !is.na() ! &

See [?base::Logic](#) and [?Comparison](#) for help.



Manipulate Variables

EXTRACT VARIABLES

Column functions return a set of columns as a new vector or table.



`pull(.data, var = -1)` Extract column values as a vector. Choose by name or index.
`pull(iris, Sepal.Length)`



`select(.data, ...)` Extract columns as a table. Also **select_if()**.
`select(iris, Sepal.Length, Species)`

Use these helpers with **select()**, e.g. `select(iris, starts_with("Sepal"))`

`contains(match)` `num_range(prefix, range)` : e.g. mpg:cyl
`ends_with(match)` `one_of(...)` : e.g. -Species
`matches(match)` `starts_with(match)`

MAKE NEW VARIABLES

These apply **vectorized functions** to columns. Vectorized funs take vectors as input and return vectors of the same length as output (see back).

vectorized function

→ **mutate(.data, ...)**
Compute new column(s).
`mutate(mtcars, gpm = 1/mpg)`

→ **transmute(.data, ...)**
Compute new column(s), drop others.

continuous bivariate distribution

`h <- ggplot(diamonds, aes(carat, price))`

`h + geom_bin2d(binwidth = c(0.25, 500))`

`h + geom_density2d()`

`h + geom_hex()`

continuous function

`i <- ggplot(economics, aes(date, unemploy))`

`i + geom_area()`

`i + geom_line()`

`i + geom_step(direction = "hv")`

visualizing error

`df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)`

`j <- ggplot(df, aes(grp, fit, ymin = fit-se, ymax = fit+se))`

`j + geom_crossbar(fatten = 2)`

`j + geom_errorbar()`

`j + geom_errorbar(l, x, ymax, ymin, alpha, color, group, linetype, size, width (also`

DECODE THE MESSAGE

DECODE THE MESSAGE

What is R markdown?

What does the mutate() function do?

What is the function used following ggplot() to specify a bar chart?

Can you run RStudio on your computer without installing R?

Are packages developed by the R Epidemics Consortium (RECON) part of the tidyverse?

BREAK: BEACH PARTY!

Using the annotation tool – draw something you'd bring with you to a deserted island beach!



WRAP UP

DAY 2 - EXERCISE

CONCEPTUAL

- You will get some data on TB cases and contacts, and clean and explore it.
- You can source/adapt existing code from day 1, write brand new code, or a combination, to create descriptive epi figures and tables relevant to the investigation.
- You will create a social network diagram
- Build an automated report in R markdown (or part of one) containing all the figures you created and any text notes or interpretation you make, and export the report document for your site.

TECHNICAL

- You will try out ggplot2 themes to change the look/format of figures.
- Create nice-looking tables flextable for reports.
- Details on structuring data for SNA with tidygraph package.
- You will source your code in an R markdown file to output a document based on your script.

PRE-LEARNING FOR DAY 3

- Details can be found in the Participant Guide
- Webpage: Canadian Chronic Disease Indicators
- From pre-course self-study module:
 - Video: Data Manipulation Tools and Dplyr (20 mins)
 - Video: R Markdown (7 mins)



Questions?

NEED HELP OR CLARIFICATION?

Remember: Slack channel for this training session

- Phacrusers.slack.com -> intro_r_nov2022cohort
- **Use the Slack channel to:**
 - Connect with your peers
 - Find materials shared throughout the course
 - Ask questions about the materials/exercises to your peers, facilitators and subject matter experts
 - Help others who have questions if you know the answer!

EXERCISE DEMO

Getting Started