Welcome to The Carpentries Etherpad! This pad is synchronized as you type, so that everyone viewing this page sees the same text. This allows you to collaborate seamlessly on documents. Use of this service is restricted to members of The Carpentries community; this is not for general purpose use (for that, try https://etherpad.wikimedia.org). Users are expected to follow our code of conduct: https://docs.carpentries.org/topic_folders/policies/code-of-conduct.html

---

Welcome to our Data Carpentry Workshop! This Etherpad is where we take notes, ask questions and do some practical exercises.

The Carpentries:
https://carpentries.org/

What is Data Carpentry:
https://datacarpentry.org/about/

Workshop Website:
https://nwu-eresearch.github.io/2021-09-27-NWU-Online/

Zoom link:
https://carpentries.zoom.us/my/carpentriesroom1

Pre- worskhop survey:
https://carpentries.typeform.com/to/wi32rS?slug=2021-09-27-NWU-Online

Link to the Etherpad
https://pad.carpentries.org/2021-09-27-NWU-Online

Daily Feedback sticky notes
https://jamboard.google.com/d/1_IKXhH2HYz_yHVV6NPht-32KjBMNAJHfUjyyqKDJD7A/viewer?f=0

Lesson Material
https://datacarpentry.org/R-ecology-lesson/index.html
https://community.rstudio.com/

Please sign in below ->
Day 1 Sign (Name & Surname,Email address, Affiliation): Twitter
Sebastian Mosidi, Sebastian.Mosidi@nwu.ac.za, North West University, @NWU_eResearch) Instructor
Heather Hodgson, University of Cape Town -
Annajiat Alim Rasel, annajiat@gmail.com, Brac University, @annajiat, Instructor
Jacinto Mathe, jacintomathe@gmail.com,University of Oxford,@jacinto, DPhil, student
Rispah N. Ng'ang'a, rispah.nganga@gmail.com, University of East Anglia, PhD student
Pheza A. Otieno, phezaotieno@gmail.com, Statistician and Monitoring and Evaluation Officer
Yolande Coetzee, yc.caly@gmail.com, North West University, M student
Please remember to rename yourself in the zoom meeting to add Windows/Mac/Linux at the end of your name.

Data File: The Portal Project Teaching DataSet
https://ndownloader.figshare.com/files/2252083

Exercise 1

We're going to take a messy version of the survey data and describe how we would clean it up.

1. Download the data by clicking here to get it from FigShare.
2. Open up the data in a spreadsheet program.
3. You can see that there are two tabs. Two field assistants conducted the surveys, one in 2013 and one in 2014, and they both kept track of the data in their own way in tabs 2013 and 2014 of the dataset, respectively. Now you're the person in charge of this project and you want to be able to start analyzing the data.
4. With the person next to you, identify what is wrong with this spreadsheet. Also discuss the steps you would need to take to clean up the 2013 and 2014 tabs, and to put them all together in one spreadsheet.

Important Do not forget our first piece of advice: to create a new file (or tab) for the cleaned data, never modify your original (raw) data.

Answers:
In the first tab, the unit was combined with the measurement of the weight, also I think that there is many table for each sheet.
in the 2014 sheet, there is also many tables and abreviation of the species name without any metadata
Tthe date sheet, there is abreviation of name of the species also without metadata and the date is recorded in the confused format, we woul not know which number means date and which means month or year.

The spreadsheet contains many tables, which is problematic to analyze.
No data was captured for some rows
Highlighted cells in the 2014 tab
Use of multiple tabs (i.e. data for 2013 & 2014 were recorded in separate tabs) hence, difficult to compare the two

Exercise

Challenge: pulling month, day and year out of dates

- Let's create a tab called dates in our data spreadsheet and copy the 'plot 3' table from the 2014 tab (that contains the problematic dates).
- Let's extract month, day and year from the dates in the Date collected column into new columns. For this we can use the following built-in Excel functions:

YEAR()
MONTH()
DAY()
(Make sure

Exercise 4

Challenge: pulling hour, minute and second out of the current time
Current time and date are best retrieved using the functions NOW(), which returns the current date and time, and TODAY(), which returns the current date. The results will be formatted according to your computer's settings.
1) Extract the year, month and day from the current date and time string returned by the NOW() function.
2) Calculate the current time using NOW()-TODAY().
3) Extract the hour, minute and second from the current time using functions HOUR(), MINUTE() and SECOND().
4) Press F9 to force the spreadsheet to recalculate the NOW() function, and check that it has been updated.

Exercise 5

What happens to the dates in the dates tab of our workbook if we save this sheet in Excel (in csv format) and then open the file in a plain text editor (like TextEdit or Notepad)? What happens to the dates if we then open the csv file in Excel?

Exercise 6

Create a variable (Gender) that only allows a user to insert the letter M for male and F for female

Exercise 7

Link to data file:
https://github.com/datacarpentry/spreadsheet-ecology-lesson/blob/gh-pages/data/survey_sorting_exercise.xlsx?raw=true

We've combined all of the tables from the messy data into a single table in a single tab. Download this semi-cleaned data file to your computer: survey_sorting_exercise
Once downloaded, sort the Weight_grams column in your spreadsheet program from Largest to Smallest. What do you notice?

Exercise 8

1. Make sure the Weight_grams column is highlighted.
2. In the main Excel menu bar, click Home > Conditional Formatting... choose a formatting rule.
3. Apply any 2-Color Scale formatting rule.
4. Now we can scan through and different colors will stand out. Do you notice any strange values?

Please sign in below ->
Day 2 Sign (Name & Surname,Email address, Affiliation): Twitter
Sebastian Mosidi, Sebastian.Mosidi@nwu.ac.za, North West University, @NWU_eResearch) Instructor
Naima Mungai, naima@thecreativeabikus.com, The AWJP @theawjp @TheAbikus Program Manger
Annajiat Alim Rasel, annajiat@gmail.com, Brac University, @annajiat , Instructor
Caroline F Ajilogba, carolfadeke@gmail.com, ARC-ISCW
Yolande Coetzee, yc.caly@gmail.com, North West University, Master's Student

https://github.com/OpenRefine/OpenRefine/wiki/Clustering-In-Depth

https://www.go-fair.org/fair-principles/

1. Marwick et al. (2017) Computational Reproducibility in Archaeological Research: Basic Principles and a Case Study of Their Implementation
2. Stewart Lowndes et al. (2017) Our path to better science in less time using open data science tools

https://codeblog.jonskeet.uk/2010/08/29/writing-the-perfect-question/

https://docs.openrefine.org/

Press CTRL + C to close the black screen of openrefine

Link For R Windows:
https://cran.r-project.org/bin/windows/base/release.htm

Link For R Studio Windows:
https://www.rstudio.com/products/rstudio/download/#download

Link For R Mac:
https://cran.r-project.org/bin/macosx/

R Studio for Mac:
https://www.rstudio.com/products/rstudio/download/#download

If some of us do not have access to computer at the moment or would like to run R codes online, we may wish to trycreating a FREE account at https://rstudio.cloud/
or
https://rdrr.io/snippets/

Here are some more options with varying degree of features:
https://mybinder.org/v2/gh/binder-examples/r/master?filepath=index.ipynb
https://replit.com/languages/rlang
https://www.w3schools.com/r/tryr.asp?filename=demo_compiler
https://paiza.io/en/projects/new?language=r
https://ideone.com/l/r
https://www.tutorialspoint.com/execute_r_online.php

https://www.mycompiler.io/online-r-compiler
https://www.jdoodle.com/execute-r-online/

Please sign in below ->

Day 3 Sign (Name & Surname,Email address, Affiliation): Twitter

- Caroline F Ajilogba, carolfadeke@gmail.com, ARC-ISCW
- Yolande Coetzee, yc.caly@gmail.com, North West University, Master's student
- Naima Mungai, naima@thecreativeabikus.com, The AWJP @theawjp @TheAbikus Program Manger
- Joseph Oguegbulu, ebukahjospeh@gmail.com, Bingham University, Karu, Nig.
- Jacinto Mathe, jacintomathe@gmail.com,University of Oxford,@jacinto, DPhil, student

Sebastian Mosidi, Sebastian.Mosidi@nwu.ac.za, North West University, @NWU_eResearch) Instructor

Exercise 1
1. We've seen that atomic vectors can be of type character, numeric (or double), integer, and logical. But what happens if we try to mix these types in a single vector?
2. What will happen in each of these examples? (hint: use class() to check the data type of your objects):

- num_char <- **c**(1, 2, 3, "a")
- num_logical <- **c**(1, 2, 3, TRUE)
- char_logical <- **c**("a", "b", "c", TRUE)
- tricky <- **c**(1, 2, 3, "4")

3. Why do you think it happens?
4. How many values in combined_logical are "TRUE" (as a character) in the following example (reusing the 2 ..._logicals from above):

- combined_logical <- **c**(num_logical, char_logical)

Exercise2
Challenge

1. Using this vector of heights in inches, create a new vector, heights_no_na, with the NAs removed.

heights <- **c**(63, 69, 60, 65, NA, 68, 61, 70, 61, 59, 64, 69, 63, 63, NA, 72, 65, 64, 70, 63, 65)
heights <- c(63, 69, 60, 65, NA, 68, 61, 70, 61, 59, 64, 69, 63, NA, 72, 65, 64, 70, 63, 55)
heights[! is.na(heights)]
heights_no_na(heights[! is.na(heights)]) *error
heights_no_na <-(heights[! is.na(heights)])
heights_no_na

2. Use the function median() to calculate the median of the heights vector.
median(heights_no_na)

3. Use R to figure out how many people in the set are taller than 67 inches.

Exercise 3

1. Create a data.frame (surveys_200) containing only the data in row 200 of the surveys dataset.
2. Notice how nrow() gave you the number of rows in a data.frame?
    - Use that number to pull out just that last row in the data frame.
    - Compare that with what you see as the last row using tail() to make sure it's meeting expectations.
    - Pull out that last row using nrow() instead of the row number.
    - Create a new data frame (surveys_last) from that last row.
3. Use nrow() to extract the row that is in the middle of the data frame. Store the content of this row in an object named surveys_middle.
4. Combine nrow() with the - notation above to reproduce the behavior of head(surveys), keeping just the first through 6th rows of the surveys dataset.

```
#Exercise 2_challenge
surveys_200=head(surveys,200)
nrow(surveys_200)
surveys_200[200,]
tail(surveys_200)
surveys_200[nrow(surveys_200),]
new_dataframe=surveys_200[nrow(surveys_200),]



dir.create("data_raw")
dir.create("fig_output")
dir.create("fig")
#Comment
#1. Functions and their Arguments====
b<- sqrt(a)
sqrt(9)
round(3.14159)
round(3.14159, digits = 2)
round(x=3.14159, digits = 2)
round(3.14159, 2)
round(2, 3.14159)
round(digits = 2, x=3.14159)
#2. Vectors and data types####
weight_kg <- 55
#c=concatenate
weight_kg <- c(50, 55, 60, 65, 82)
weight_kg
```

```
#weight_kg <- c(55, 60, 63)-to show lack of c
animals <- c("mouse", "rat", "dog")
animals
#3. functions to help inspect vectors====
length(weight_kg)
length(animals)
str(weight_kg)
str(animals)
#use c to add more data
weight_kg <- c(weight_kg, 90)
weight_kg
weight_kg <- c(30, weight_kg)
weight_kg
#data types====
#character
#numeric
#logical
#integer
#complex
#raw
typeof(weight_kg)
typeof(animals)
class(animals)
class(weight_kg)

num_char <- c(1, 2, 3, "a")

class(num_char)

num_logical <- c(1, 2, 3, TRUE)

class(num_logical)

char_logical <- c("a", "b", "c", TRUE)

class(char_logical)

tricky <- c(1, 2, 3, "4")
tricky
num_logical <- c(1, 2, 3, TRUE)
char_logical <- c("a", "b", "c", TRUE)
combined_logical <- c(num_logical, char_logical)
combined_logical
class(combined_logical)
#4. Subsetting Vectors####
#use of square bracket
animals <- c("mouse", "rat", "dog")
animals[2]
animals[c(3, 2)]
```

```
more_animals <- animals[c(1, 2, 3, 2, 1, 3)]
more_animals
#conditional subsetting
weight_kg
weight_kg <- c(50, 55, 60, 65, 82)

weight_kg[c(TRUE, FALSE, FALSE, TRUE, TRUE)]
weight_kg>50
weight_kg[weight_kg>50]
#other symbols for subseting-&-and, |-or,
weight_kg <- c(30, 50, 55, 60, 65, 82, 90)
weight_kg

weight_kg[weight_kg>30 & weight_kg<55]
weight_kg[weight_kg<=30 | weight_kg == 55]
weight_kg[weight_kg<=30 & weight_kg == 55]

animals
animals <- c("mouse", "rat", "dog", "cat")
animals
animals[animals == cat & animals== rat]
animals[animals == "cat" | animals == "rat"]
#5. Missing data====
heights <- c(2, 4, 4, NA, 6)
mean(heights)
max(heights)
#na.rm
mean(heights, na.rm = TRUE)
max(heights, na.rm = TRUE)
#is.na, na.omit, complete.cases
heights[is.na(heights)]
#!, negate, or is not
heights[!is.na(heights)]

#!, negate, or is not
heights[!is.na(heights)]
na.omit(heights)
heights[complete.cases(heights)]
#6. Starting with Data====
download.file(url = "https://ndownloader.figshare.com/files/2292169",
              destfile = "data_raw/portal_data_joined.csv")
install.packages("tidyverse")
library(tidyverse)
read_csv("data_raw/portal_data_joined.csv")
surveys <- read_csv("data_raw/portal_data_joined.csv")
head(surveys)
tail(surveys)
view(surveys)
head(surveys, 100)
```

```
surveys_sample <- head(surveys, 100)
str(surveys)
#Size
dim(surveys)
nrow(surveys)
ncol(surveys)
#Content
head(surveys)
tail(surveys)
#names of col and row
names(surveys)
rownames(surveys)
#Summary
str(surveys)
summary(surveys)
#7. Indexing and subsetting data frame====
surveys[1, 6]
surveys[ , 1]
surveys[1, ]
#:, 1:6
surveys[c(1,2,3), c(5, 6)]
surveys[1:3, 5:6]
#- minus
surveys[ , -1]
nrow(surveys)#nrow gives the last row or number of rows
surveys[-(7:nrow(surveys)), ]
#subsetting using col names
surveys["species_id"]
surveys[  , "species_id"]
surveys[["species_id"]]
#$ means to extract col
surveys$species_id
#Exercise 3
#SurveyExercise 3
#Create a data.frame (surveys_200) containing only the data
#in row 200 of the surveys dataset.
#Notice how nrow() gave you the number of rows in a data.frame?
#  Use that number to pull out just that last row in the data frame.
#Compare that with what you see as the last row using tail()
#to make sure it’s meeting expectations.
#Pull out that last row using nrow() instead of the row number.
#Create a new data frame (surveys_last) from that last row.
#Use nrow() to extract the row that is in the middle of the data frame.
#Store the content of this row in an object named surveys_middle.
#Combine nrow() with the - notation above to reproduce the
#behavior of head(surveys), keeping just the first through 6th rows
#of the surveys dataset.

#Exercise 2_challenge
```

```
surveys_200 <- head(surveys,200)# for row 1 to 200
survey_200 <- surveys[1:200, ]# for row 1-200
Surveys_200 <- surveys[200, ]# for only row 200, which is correct

nrow(surveys_200)
n_row <- nrow(surveys)
surveys_last <- surveys[n_row, ]

surveys_middle <- surveys[n_row/2, ]
surveys_middle

surveys_head <- surveys[-(7:n_row), ]
surveys_head
```

Challenge

```
#Exercises
heights <- c(63, 69, 60, 65, NA, 68, 61, 70, 61, 59, 64, 69, 63, 63, NA, 72, 65, 64, 70, 63, 65)
heights=heights[complete.cases(heights)]
median(heights)
length(heights[heights>67])
```

Please sign in below ->

Day 4 Sign (Name & Surname,Email address, Affiliation): Twitter

- Sebastian Mosidi, Sebastian.Mosidi@nwu.ac.za, North West University, @NWU_eResearch) Instructor
- Jacinto Mathe, jacintomathe@gmail.com,University of Oxford,@jacinto, DPhil, student
- Naima Mungai, naima@thecreativeabikus.com, The AWJP @theawjp @TheAbikus Program Manger
- Yolande Coetzee, yc.caly@gmail.com, North West University, Master's student
- Martin Dreyer,martin.dreyer@nwu.ac.za, North West University

Factors

```
surveys$sex<- factor(surveys$sex)
summary(surveys$sex)

sex<-factor(c("male","female","female","male"))
levels(sex)
nlevels(sex)
sex
```

```
sex<-factor(sex,levels = c("male","female"))

#converting factors
as.character(sex)

year_fct <- factor(c(1990,1983,1977,1998,1990))
as.numeric(year_fct)
as.numeric(as.character(year_fct))
as.numeric(levels(year_fct))[year_fct]

surveys$sex <-factor(surveys$sex)

plot(surveys$sex)
sex<- surveys$sex
sex<- addNA(sex)
levels(sex)
levels(sex)[3]<-"undetermined"

plot(sex)

levels(sex)[1:2]<- c("female","male")
levels(sex)
sex <- factor(sex,levels = c("undetermined","female","male"))
levels(sex)
plot(sex)
############################################################
##Fromatting Dates

str(surveys)
library(tidyverse)
library(lubridate)

my_date<- ymd("2015-01-01")
str(my_date)

my_date <- ymd(paste("2015","1","1", sep ="-"))
str(my_date)

paste(surveys$year,surveys$month,surveys$day,sep = "-")
ymd(paste(surveys$year,surveys$month,surveys$day,sep = "-"))

surveys$date <-ymd(paste(surveys$year,surveys$month,surveys$day,sep = "-"))
str(surveys)
summary(surveys$date)

missing_dates<-surveys[is.na(surveys$date), c("year","month","day")]
head(missing_dates)

##Selecting columns and filtering rows
```

```
select(surveys,plot_id,species_id, weight)
select(surveys, -record_id, -species_id)

filter(surveys, year == 1995)
surveys2<-filter(surveys, weight<5)
surveys_sml<-select(surveys2, species_id, sex, weight)
view(surveys_sml)

surveys_sml<-select(filter(surveys,weight<5),species_id,sex,weight)

surveys %>%
  filter(weight<5) %>%
  select(species_id,sex,weight)

surveys_sml<- surveys %>%
  filter(weight<5) %>%
  select(species_id,sex,weight)

#Exercise
#Using pipes, subset the surveys data to include animals
#collected before 1995 and retain only the columns year, sex, and weight.
surveys_exe<-surveys %>%
  filter(year<1995) %>%
  select(year,sex,weight)

#Mutate
surveys %>%
  mutate(weight_kg = weight/1000) %>%
  head()

surveys %>%
  mutate(weight_kg = weight/1000,
         weight_lb = weight_kg *2.2)
view(surveys)

surveys %>%
  filter(!is.na(weight)) %>%
  mutate(weight_kg = weight/1000) %>%
  head()

#Create a new data frame from the surveys data that meets the following criteria:
#contains only the species_id column and a new column called hindfoot_cm containing
#the hindfoot_length values (currently in mm) converted to centimeters.
#In this hindfoot_cm column, there are no NAs and all values are less than 3.

surveys_hindfoot_cm<- surveys %>%
  filter(!is.na(hindfoot_length)) %>%
  mutate(hindfoot_cm = hindfoot_length/10) %>%
```

```
  filter(hindfoot_cm<3) %>%
  select(species_id, hindfoot_cm)
view(surveys_hindfoot_cm)
```

#Split-apply-combine data analysis and the summarize() function

```
surveys %>%
  group_by(sex) %>%
  summarize(mean_weight = mean(weight, na.rm = TRUE))

surveys %>%
  group_by(sex,species_id) %>%
  summarize(mean_weight= mean(weight, na.rm=TRUE)) %>%
  head()

surveys %>%
  filter(!is.na(weight)) %>%
  group_by(sex,species_id) %>%
  summarize(mean_weight = mean(weight)) %>%
  print(n = 25)

surveys %>%
  filter(!is.na(weight)) %>%
  group_by(sex,species_id) %>%
  summarize(mean_weight = mean(weight),
            min_weight = min(weight)) %>%
  arrange(sex)

#Counting

surveys %>%
  count(sex)

surveys %>%
  group_by(sex) %>%
  summarise(count=n())

surveys %>%
  count(sex, sort=TRUE)

#How many animals were caught in each plot_type surveyed?

surveys %>%
  count(plot_type)

#Use group_by() and summarize() to find the mean, min, and max hindfoot length for each species
#(using species_id). Also add the number of observations (hint: see ?n).

surveys %>%
```

```
  filter(!is.na(hindfoot_length)) %>%
  group_by(species_id) %>%
  summarize(mean_hindfoot_length = mean(hindfoot_length),
            min_hindfoot_length = min(hindfoot_length),
            max_hindfoot_lengt = max(hindfoot_length),
            n=n())
#What was the heaviest animal measured in each year?
#Return the columns year, genus, species_id, and weight

surveys %>%
  filter(!is.na(weight)) %>%
  group_by(year) %>%
  filter(weight == max(weight)) %>%
  select(year,genus,species_id, weight) %>%
  arrange(year)

#Exporting Data
#remove missing data
surveys_complete<- surveys %>%
  filter(!is.na(weight),
         !is.na(hindfoot_length),
         !is.na(sex))
#extract the most common species
species_counts<-surveys_complete %>%
  count(species_id) %>%
  filter(n >=50)
#Only keep most common species
surveys_complete <- surveys_complete %>%
  filter(species_id %in% species_counts$species_id)

#write csv into data folder
write_csv(surveys_complete,file="data/surveys_complete.csv")

#Visualizing data

surveys_complete <- read_csv("data/surveys_complete.csv")

#ggplot(data = <DATA>, mapping = aes(<MAPPINGS>)) +  <GEOM_FUNCTION>()

ggplot(data=surveys_complete,mapping=aes(x = weight, y = hindfoot_length)) +
  geom_point()
  #assign your plot to a variable
surveys_plot<- ggplot(data=surveys_complete,mapping=aes(x = weight, y = hindfoot_length))
```

Exercise 1:

- Rename “F” and “M” to “female” and “male” respectively.

- Now that we have renamed the factor level to “undetermined”, can you recreate the barplot such that “undetermined” is first (before “female”)?

Exercise 2

Exercise 2

```
surveys %>%
  filter(year<1995) %>%
  select(year,sex,weight)

surveys_tst <- surveys %>%
  filter(year<1995) %>%
  select(year,sex,weight)

view(surveys_tst)
```

Exercise 3

Create a new data frame from the surveys data that meets the following criteria: contains only the species_id column and a new column called hindfoot_cm containing the hindfoot_length values (currently in mm) converted to centimeters. In this hindfoot_cm column, there are no NAs and all values are less than 3.

Hint: think about how the commands should be ordered to produce this data frame!

Feedback for day4

Things we can improve on

- I really liked the collaboration from todays class, it made learning easier, humanising all of us. More of that please.
- Not sure how, because I know they are on the notes, but a quick tip sheet that has basic definitions for new commands. So that I can review them, especially when doing the exercises/challenges
- I second a definitions list because I don't always understand the codes and then struggle to apply them.

Things that worked for you

- Liked how the instructor took the time to explainconcepts that I was having issues with.
- The way the classes are structured is really good because I was able to build up on skills as I learnt

them.
- I like the frequent short breaks because the information can get overwhelming at times without them.

Please sign in below ->

Day 5 Sign (Name & Surname,Email address, Affiliation): Twitter

- Sebastian Mosidi, Sebastian.Mosidi@nwu.ac.za, North West University, @NWU_eResearch) Instructor
- Naima Mungai, naima@thecreativeabikus.com, The AWJP @theawjp @TheAbikus Program Manger
- Martin Dreyer, martin.dreyer@nwu.ac.za, North West University.
- Jacinto Mathe, jacintomathe@gmail.com,University of Oxford,@jacinto, DPhil, student
- Yolande Coetzee, yc.caly@gmail.com, North West University, Master's student

Exercise1

1. How many animals were caught in each plot_type surveyed?

```
surveys %>%
  group_by(plot_type) %>%
  summarise(count=n())
```

2. Use group_by() and summarize() to find the mean, min, and max hindfoot length for each species (using species_id). Also add the number of observations (hint: see ?n).

```
surveys %>%
  group_by(hindfoot_length) %>%
  summarize(mean=mean(hindfoot_length), min=min(hindfoot_length),
        max=max(hindfoot_length), count=n())
```

1. What was the heaviest animal measured in each year? Return the columns year, genus, species_id, and weight.

Exercise 2

Use what you just learned to create a scatter plot of weight over species_id with the plot types showing in different colors. Is this a good way to show this type of data?

```
ggplot(data=surveys_complete,mapping=aes(x = species_id, y = weight))+geom_point(color="green")
+theme_light()
```