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

---

Day 3: R

Feedback for Today

One thing I liked

Good breadth of information covered.
Brilliant to see practical application of R in gg plot
Everthing
Felt much better today - more applicable and the code was structured in a way that my brain works better - data, aesthetics, geom. I liked playing around with the charts too, with different themes etc. Good reference sources were offered for future help too - I've saved them all down.
Good breadth of info and good to see how can use practically to see the data.
Enjoyed the visualisations, seeing the data brought to life that way showed how useful these skills are
Today was better in that it wasn't so overwhelming; the code was easier and great to see what can be done with the data.

One thing I disliked/could be better

Nothing.
Good day- thank you!
Nothing
nothing - a much easier to digest day than day 2
Nothing x
Really helpful, nothing disliked
The only minor thing I have to add is that there was perhaps too much material to cover in the afternoon as the instructor skipped over some things in order to fit it all in; not that I considered that anything fundamental was left out. More just an observation that maybe the learning material document needs to be reduced a little.

10:00 Start

Morning: Data Wrangling

12:00-13:00 Lunch break

Afternoon: Visualisation with ggplot2

15:00 Finish

Exercise 1 - pipes
https://datacarpentry.org/r-socialsci/03-dplyr.html#exercise

Exercise 2 - mutate
https://datacarpentry.org/r-socialsci/03-dplyr.html#exercise-1

Exercise 3 - split-apply-combine
https://datacarpentry.org/r-socialsci/03-dplyr.html#exercise-2

Day 2 - afternoon code
https://pastebin.com/MjmRS9Lp

Exercise 4 - ggplotting fundamentals
https://datacarpentry.org/r-socialsci/05-ggplot2.html#exercise

Attendance, Day 3 - morning (write your name):

1) Andrzej Romaniuk (Host)
2) Edwina Emery
3) Hanna Mors
4) Zena Timmons
4) Craig Livie
5) Cristina Stewart
6) Gary Furlong
7) Jen Ward
8) Jasmine Fyfe
9) Elizabeth Thompson-MacRae
10) Taiwo Akindele
11) Brian Omondi (helper)
12) Gemma Blackledge-Foughali
13) Ritji Samson
14) Andy Monaghan

Haolan Tu (helper)

LIVE CODING FOR TODAY

```
#loading required library
library(tidyverse)

#loading data (make sure you have correct working directory set)
interviews = read_csv("data/SAFI_clean.csv", na = "NULL")
interviews

# selecting columns
args(select)

select(interviews, village, no_membrs, months_lack_food) # dplyr

interviews[,c('village','no_membrs','months_lack_food')] # base R

# select slices
select(interviews, village:respondent_wall_type)
# select except
select(interviews, -village, -no_membrs)

# filtering rows
filter(interviews, village == 'Chirodzo')

#filtering rows
filter(interviews, village == 'Chirodzo')
interviews[interviews$village == 'Chirodzo',]


# filtering rows on multiple conditionals
filter(interviews, village == "Chirodzo", rooms > 1, no_meals > 2)   # similar to:
filter(interviews, village == "Chirodzo" & rooms > 1 & no_meals > 2) # & operator

filter(interviews, village == "Chirodzo" | village == "Ruaca") # | is or operator

#pipes
#using intermediate steps
interviews2 = filter(interviews, village == "Chirodzo")
interviews_col = select(interviews2, village:respondent_wall_type)

#using pipes
interviews %>%
%>%  filter(village == "Chirodzo")

#Exercise 1
```

```
# Using pipes, subset the interviews data to include interviews where
# respondents were members of an irrigation association (memb_assoc) and retain
# only the columns affect_conflicts, liv_count, and no_meals.

# Answer
interviews %>%
  filter(memb_assoc == "yes") %>%
  select(affect_conflicts, liv_count, no_meals)

#mutate
interviews %>%
  mutate(vilalge_lower = tolower(village)) %>%
  select(vilalge, village_lower) %>%
  filter(village_lower == 'god')

interviews %>%
  select(no_membrs, rooms) %>%
  mutate(people_per_room = no_membrs/rooms)

interviews %>%
  filter(!is.na(memb_assoc)) %>%
  mutate(people_per_room = no_membrs/rooms)

#split/apply/combine
interviews %>%
  group_by(village, respondent_wall_type) %>%
  summarise(mean_no_membrs = mean(no_membrs)) %>%
  ungroup() %>%
  arrange(desc(mean_no_membrs))

#counting
interview %>%
  count(village)    # approach 1, using count()

interviews %>%
  group_by(village) %>%
  summarise(n = n())  # approach 2, using summarise() and n()

#Necessary code
interviews_plotting <- interviews %>%
  ## pivot wider by items_owned
  separate_rows(items_owned, sep = ";") %>%
  ## if there were no items listed, changing NA to no_listed_items
  replace_na(list(items_owned = "no_listed_items")) %>%
  mutate(items_owned_logical = TRUE) %>%
  pivot_wider(names_from = items_owned,
              values_from = items_owned_logical,
              values_fill = list(items_owned_logical = FALSE)) %>%
  ## pivot wider by months_lack_food
```

```
  separate_rows(months_lack_food, sep = ";") %>%
  mutate(months_lack_food_logical = TRUE) %>%
  pivot_wider(names_from = months_lack_food,
              values_from = months_lack_food_logical,
              values_fill = list(months_lack_food_logical = FALSE)) %>%
  ## add some summary columns
  mutate(number_months_lack_food = rowSums(select(., Jan:May))) %>%
  mutate(number_items = rowSums(select(., bicycle:car)))

## ggplot fundamentals - how to visualise with it
## by + we join basic information to work with in ggplot() with
## geometry to be created based on this data
ggplot(data = interviews_plotting,
           aes(x = no_membrs,
                 y = number_items)) +
geom_point(alpha = .3, position = position_jitter(width = .1, height = 0))

ggplot(data = interviews_plotting,
           aes(x = no_membrs,
                 y = number_items)) +
                 geom_point(aes(color = village),
                 alpha = .8,
                 position = position_jitter())

## exercise
ggplot(interviews_plotting,
           aes(x = village, y = rooms)) +
           geom_point(aes(color = respondent_wall_type),
           position = position_jitter(width = 0.1))


#exporting a table into csv file
write.csv(interviews_plotting, "data_output/interviews_plotting.csv")
```

Day 2: R

Feedback for Today

One thing I liked

good support
Good pace
covered a lot of functions and features of R within a short space of time, and very patient with us when

we're asking lots of questions.
Reasonable pace
Great exercises to work through
Support was good when needed
Helpful support

Good Pace - enjoyed the functions to explore csv's in R

One thing I disliked/could be better

nothing
Nothing
I'm struggling to apply this learning to 'real-world' applications - like, I think the last feature 'unique' could maybe help me identify unique words within a survey comments open text box, but I'm not 100% sure if that's true - I don't think I'll ever have to subset a list of words in real life. I expected to be able to perhaps apply some of this to my day to day work after Thursday, and I'm struggling to think of ways I will take this learning forward, at this time. Is there any follow-up learning/practice we could do to help build confidence? Also maybe a jargon buster - like vectors, factors, subsets, functions - these are all new terms to me. I'd also like more explanation on the 'solutions' given by Data Carpenter, like why is there a minus and why 7 instead of 6 on the last exercise? Same, lots of concepts that aren't necessarily clear to someone who isn't using this already.

To add to this comment above perhaps there might be an option in the future for those attending to have a day or half a day working with some of their own data to see how the learning outcomes may be applied in the real world.
Could do with a small break in afternoon as a lot to process - yes
I've found today really difficult as so much packed in as a complete novice to this. Struggling to make connections of what we are doing and how it might be used in practice at points so I'm hoping that I'll make those connections in time. -Ditto this..and another ditto from me! :D
Understanding some of the concepts may be easier starting with a sample dataset instead of creating vectors, that ties into a comment above when thinking about how we apply these functions in real datasets
comments on more complex code from today

I hope it's okay for me to share this but this online (free) textbook is an EXCELLENT intro to R with great exercises to work through and nice examples. My first into to R was reading through this book and it was so helpful, highly recommend: https://r4ds.hadley.nz/

You can share ;) haha thanks!

10:00 Start

Morning: Introduction to R

12:00-13:00 Lunch break

Afternoon: More Introduction to R

15:00 Finish

Entire R course is here (save link for future reference!): https://datacarpentry.org/r-socialsci/

Setup:

All links here -> https://datacarpentry.org/r-socialsci/index.html
We will need:

1. RStudio and R downloaded, installed, and check everything is working (if possible!)
2. SAFI_clean.csv downloaded and placed somewhere we can find it

https://datacarpentry.org/r-socialsci/02-starting-with-data.html

exercise 1 - calculating rarea
https://datacarpentry.org/r-socialsci/01-intro-to-r.html#exercise-1

Exercise 2 - vectors
https://datacarpentry.org/r-socialsci/01-intro-to-r.html#exercise-3

Exercise 3 - subsetting

```
# conditional subsetting
hh_members[c(1,2)] # subset by index (element number)
hh_members[c(TRUE, FALSE, TRUE, TRUE, FALSE, FALSE)] # subset by conditional

# conditional operators
# >
# <
# <=
# >=
# &
# "|"
# == -> equals to
# != -> not equals to

hh_members[hh_members > 5]
hh_members[hh_members < 5]

hh_members[hh_members < 14 & hh_members > 5]

respondent_wall_type[respondent_wall_type == 'burntbricks']
respondent_wall_type[respondent_wall_type == 'burntbricks' | respondent_wall_type == 'sunbricks']
```

Exercise 4 - missing data
https://datacarpentry.org/r-socialsci/01-intro-to-r.html#exercise-4

Exercise 5 - subsetting dataframes
https://datacarpentry.org/r-socialsci/02-starting-with-data.html#exercise

Attendance, Day 2 - morning (write your name):

1) Andrzej Romaniuk (host)
2) Brian Omondi (helper)
3) Jasmine Fyfe
4) Jen Ward
5) Cristina Stewart
6) Edwina Emery
7) Hanna Mors
8) Aleksandra Chybowska (helper)
9) Zena Timmons
10) Craig Livie
11) Elizabeth Thompson-MacRae
12) Gemma Blackledge-Foughali
13) Gary Furlong
14) Dorcas Ojo
15) Taiwo Akindele
16) Andy Monaghan
17) Ritji Samson

USEFUL INFO/CODE/ETC:

SETUP

setwd("path") -> setting your working directory (default directory), either manually in code, with path into "" or via cog sign -> set as a working directory after navigating to the directory via Files in RStudio
getwd() -> function that states your current working directory

Also, you can set the working directory automatically by creating an "R project" (upper right RStudio -> New Project...) Then each time you open rproj file at the project folder you go directly to R, with all elements as defined for the project alrady loaded

```
# Once we have our working directory, we can create folders in it, e.g. by using dir.create() function
dir.create("data")
dir.create("data_output")
dir.create("fig_output")
# folders can be also set manually
```

LIVE CODE - TODAY

```
#You can type directly in RStudio console (by default bottom left corner)
```

```
 #e.g.
 3 + 5

#You can also write code in a "script" file, visible upper left corner when opened (file -> new -> rscript)

 #by the way, by using # we can type comments in code, omitted when running code

#mathematical and logical operatos work as intended in code
# see https://www.statmethods.net/management/operators.html
3 + 5
12 / 5
12 ** 3

 #the only difference is that = is for assigning values, not as math operator
 #(for math we use == instead)
 # Here we assign a value of 1.0 to a name area_hectares, thus creating a
 # permament object in our environment (we can later use it in later work)
area_hectares = 1.0
# alternative to = is <- , included in R due to some confusion with =
area_hectares <- 1.0
#name for an object can be anything, as long as it does not start from a number
#or include special characters
#consistency is important when naming, for both our sake, RStudio and other people #that may work on this code
# you can view an object content either by typing it in console and pressing enter
area_hectares
# ... or you can use a specific function called print(), with the object name within ()
print(area_hectares)

#As written before, we can perform further work on an object, and save it as a #different object

area_hectares = 1.0                    #creating an object
area_acres = area_hectares * 2.47    #using this object in an equation
                                          #to create a new object
area_acres                                # print acres in console

# functions, the way to do stuff without typing too much ;)
b = sqrt(9)
# functions are written to perform/automate specific tasks
# we can use ? to check a function, e.g. what it does
?round
# or args() to get info of what input arguments are required or expected
args(round)

# for round() we know there are two arguments, one, x, required, and one, digits, set
# as default but changeable if stated
round(3.1419, digits = 3)
round(3.1419, 3)            # not need to stare argument name if we put them in
                            # expected order
```

```
round(digits = 3, x = 3.1419) # if not in order, stat what is what

# vectors are a single-dimension set of values, e.g. text strings, numbers, etc
#we can create vectors by using c() and putting stuff in (), with , as separator
hh_members = c(3, 7, 10)

#We can have text vectors
respondent_wall_type = c("burntbricks","sunbricks","muddaub")

## we can use typeof() or str() to find informaiton about a vector
typeof(hh_members)
str(hh_members)

## you can manipulate, e.g. add new values at the end or start of a vector
hh_members = c(hh_members, 15)
hh_members

hh_members = c(3, hh_members)
hh_members

# data types?
numeric_vector = c(3, 55, 5, 4, 43)
character_vector = c('test','vector')
logical_vector = c(TRUE, FALSE)

# subsetting vectors
respondent_wall_type
str(respondent_wall_type)

respondent_wall_type[2] #indexing by scalar

respondent_wall_type[c(2,3)] # indexing by a vector
respondent_wall_type[c(1,2,1,1,1,3)]

# conditional subsetting (only logically true values are returned)
hh_members[c(TRUE, FALSE, TRUE, TRUE), FALSE)]

# conditional operators, like < or >, can be utilized
hh_members > 5
hh_members[hh_members > 5]
hh_members[hh_members < 14 & hh_members > 5]

respondent_wall_type[respondent_wall_type == "burntbricks"]
respondent_wall_type[respondent_wall_type == "burntbricks" | respondent_wall_type == "sunbricks"]

# missing data
rooms = c(2,1,2,NA,7)
mean(rooms)
max(rooms)
```

```
min(rooms)

# how do we handle NA values? Some functions have built-in options
# good to first check through ? option
# for mean() it is na.rm as True (or T)
?mean
mean(rooms, na.rm = T)

# another method to inverse the outcome of is.na() function by using ! with it
!is.na(rooms)
new_rooms = rooms[!is.na(rooms)]
 mean(new_rooms)

 #another approach, using na.omit()
 mean(na.omit(rooms))



 # load in libraries
 library(tidyverse)

 # load in our data file (hopefully in you working directory/data)
 interviews = read_csv('data/SAFI_clean.csv', na = 'NULL')

 interviews
 colnames(interviews)
 rownames(interviews)
 glimpse(interviews)

 #subsetting dataframes
 #first number rows, second columns (if we define both values)
interviews[5,7]
interviews[c(1,2,3,4,5),7]
interviews[1:10, 1:10] # using slices
interviews[3,]
interviews[,3]

 head(interviews)
 interviews[1:6,]

interviews[,-3]
interviews[,c(-2.-3)]

## subsetting by column name
interviews['village']
interviews[c('village','key_ID')]
interviews[,'village']
interviews[['village']]
interviews$village
```

```
# factors (format of text data, where we have specific categories stated)
fct = factor(interviews$village)
class(fct)
levels(fct)

fct = factor(interview$village, levels = c('God','Chirodzo','Ruaca'))
```

Day 1: Spreadsheets and OpenRefine

10:00-10:10 Intro

10:10 -11:00 Data organisation with Spreadsheets (Andrzej)

11:00-11:05 Coffee break 1

11:05-12:00 Data organisation with Spreadsheets (Andrzej)

12:00-13:00 Lunch break

13:00-14:00 Data cleaning with OpenRefine (Andrzej)

14:00-14:05 Coffee break 2

14:05-15:00 Data cleaning with OpenRefine (Andrzej)

Before you start (Spreadsheets, https://datacarpentry.org/spreadsheets-socialsci/):

*Make sure you have a spreadsheet program installed before the workshop (Excel, Apple Numbers etc.). A free
spreadsheet editor (part of the Libre Office open-source project) can be downloaded here:
https://www.libreoffice.org/download/download/
For installation instructions on different operating software, see this page:
https://datacarpentry.org/spreadsheets-socialsci/setup.html

*Download all necessary files:
SAFI_clean.csv - https://ndownloader.figshare.com/files/11492171
SAFI_messy.xlsx - https://ndownloader.figshare.com/files/11502824
SAFI_dates.xlsx - https://ndownloader.figshare.com/files/11502827

*Make sure you have all the files in the same, accesible folder

Presentation used today can be downloaded here:
https://docs.google.com/presentation/d/1nmuJmVmUQqfwc5npyAySpQjOp5ZTkDAs/edit?usp=share_link&ouid=104347110123739783728&rtpof=true&sd=true

Before you start (OpenRefine, https://datacarpentry.org/openrefine-socialsci/):

*Download OpenRefine package from: https://openrefine.org/download.html
On Windows, you simply unpack the zipped folder. See installation instructions here:
https://datacarpentry.org/openrefine-socialsci/setup.html

* Download the file we will work on here: https://ndownloader.figshare.com/files/11502815

*Best is to have both OpenRefine and the data file in the same folder

Most common issues:

The most common issue encountered is a lack of administrative privileges. If you plan to access the workshop
from your study/workplace computer, firstly make sure you have the relevant software preinstalled or can install it on your own. If not, you will have to ask a relevant person in charge (IT/system administrator) to install
and prepare software for you. It should be simpler for home computers, with OpenRefine sometimes requiring to
open the executable file as an administrator (on Windows, can be done by right click -> run as administrator, for
one time only, or by navigating to properties -> comparability -> run as admin -> apply, to set it permanently).
For OpenRefine, it is generally advised to install version with Java files included.

Webiste: https://edcarp.github.io/2024-04-09-dusc-dc-r-online/

Attendance, Day 1 - morning (write our name):

1. Andrzej Romaniuk (Instructor)
2. Mario Antonioletti (host)
3. Aleksandra Chybowska (helper)
4. Haolan Tu (helper)
5. Zena Timmons
6. Edwina Emery

7. Hanna Mors
8. Elizabeth Thompson-MacRae
9. Gary Furlong
10. Jasmine Fyfe
11. Andy Monaghan
12. Jen Ward
13. Taiwo Akindele
13. Craig Livie
14. Gemma Blackledge-Foughali
15. Ritji Samson

Attendance, Day 1 - afternoon (write your name):

1) Andrzej Romaniuk (Instructor)
2) Aleksandra Chybowska (helper)
3) Haolan Tu (helper)
4) Zena Timmons
5) Jasmine Fyfe
6) Elizabeth Thompson-MacRae
7) Andy Monaghan
8) Hanna Mors
9) Jen Ward
10) Mario Antonioletti (host)
11) Taiwo Akindele
12) Gary Furlong
13) Edwina Emery
14. Craig Livie
15. Gemma Blackledge-Foughali
16. Ritji Samson

CODE TO APPLY FOR TODAY

```
[
  {
    "op": "core/mass-edit",
    "engineConfig": {
      "facets": [],
      "mode": "row-based"
    },
    "columnName": "village",
    "expression": "value",
    "edits": [
      {
        "from": [
          "Ruaca",
          "Ruca"
        ],
        "fromBlank": false,
```

```
      "fromError": false,
      "to": "Ruaca"
    },
    {
      "from": [
        "Ruaca-Nhamuenda",
        "Ruaca - Nhamuenda"
      ],
      "fromBlank": false,
      "fromError": false,
      "to": "Ruaca"
    }
  ],
  "description": "Masowa edycja komórek w kolumnie village"
},
{
  "op": "core/mass-edit",
  "engineConfig": {
    "facets": [],
    "mode": "row-based"
  },
  "columnName": "village",
  "expression": "value",
  "edits": [
    {
      "from": [
        "Chirdozo"
      ],
      "fromBlank": false,
      "fromError": false,
      "to": "Chirodzo"
    }
  ],
  "description": "Masowa edycja komórek w kolumnie village"
},
{
  "op": "core/text-transform",
  "engineConfig": {
    "facets": [],
    "mode": "row-based"
  },
  "columnName": "interview_date",
  "expression": "value.toDate()",
  "onError": "keep-original",
  "repeat": false,
  "repeatCount": 10,
  "description": "Text transform on cells in column interview_date using expression value.toDate()"
},
{
```

```
  "op": "core/mass-edit",
  "engineConfig": {
    "facets": [],
    "mode": "row-based"
  },
  "columnName": "village",
  "expression": "value",
  "edits": [
    {
      "from": [
        "49"
      ],
      "fromBlank": false,
      "fromError": false,
      "to": "Chirodzo"
    }
  ],
  "description": "Masowa edycja komórek w kolumnie village"
},
{
  "op": "core/text-transform",
  "engineConfig": {
    "facets": [],
    "mode": "row-based"
  },
  "columnName": "years_farm",
  "expression": "value.toNumber()",
  "onError": "keep-original",
  "repeat": false,
  "repeatCount": 10,
  "description": "Text transform on cells in column years_farm using expression value.toNumber()"
},
{
  "op": "core/text-transform",
  "engineConfig": {
    "facets": [],
    "mode": "row-based"
  },
  "columnName": "years_farm",
  "expression": "value.toNumber()",
  "onError": "keep-original",
  "repeat": false,
  "repeatCount": 10,
  "description": "Text transform on cells in column years_farm using expression value.toNumber()"
},
{
  "op": "core/text-transform",
  "engineConfig": {
    "facets": [],
```

```
      "mode": "row-based"
    },
    "columnName": "items_owned",
    "expression": "grel:value.replaceChars(\"[] '\",\"\")",
    "onError": "keep-original",
    "repeat": false,
    "repeatCount": 10,
    "description": "Text transform on cells in column items_owned using expression
grel:value.replaceChars(\"[] '\",\"\")"
  },
  {
    "op": "core/text-transform",
    "engineConfig": {
      "facets": [],
      "mode": "row-based"
    },
    "columnName": "months_lack_food",
    "expression": "grel:value.replaceChars(\"[] '\",\"\")",
    "onError": "keep-original",
    "repeat": false,
    "repeatCount": 10,
    "description": "Text transform on cells in column months_lack_food using expression
grel:value.replaceChars(\"[] '\",\"\")"
  },
  {
    "op": "core/text-transform",
    "engineConfig": {
      "facets": [],
      "mode": "row-based"
    },
    "columnName": "no_food_mitigation",
    "expression": "grel:value.replaceChars(\"[] '\",\"\")",
    "onError": "keep-original",
    "repeat": false,
    "repeatCount": 10,
    "description": "Text transform on cells in column no_food_mitigation using expression
grel:value.replaceChars(\"[] '\",\"\")"
  },
  {
    "op": "core/column-addition",
    "engineConfig": {
      "facets": [],
      "mode": "row-based"
    },
    "baseColumnName": "items_owned",
    "expression": "grel:value.contains(\"bicycle\").replace(\"true\",\"yes\").replace(\"false\",\"no\")",
    "onError": "set-to-blank",
    "newColumnName": "bicycle_owned",
    "columnInsertIndex": 53,
```

"description": "Create column bicycle_owned at index 53 based on column items_owned using expression grel:value.contains(\"bicycle\").replace(\"true\",\"yes\").replace(\"false\",\"no\")"
  }
]

Feedback for Today

One thing I liked

Learning about OpenRefine - it will be very useful in the future when working with survey comments.
Support on hand if needed help
Clear instructions, helpful environment
Good instruction- and learning about OpenRefine, sorry two points :)

One thing I disliked/could be better

Nothing
More thinking time
Agree with more thinking time, not all delegates work at the same pace
nothing
nothing