Welcome to our Software Carpentry Workshop!

Tuesday-Wednesday (July 9-10, 2024)

Workshop Website: https://justinshaffer.github.io/2024-07-09-Software-Carpentry
This etherpad: https://pad.carpentries.org/2024-07-09-Software-Carpentry

Lessons websites:

- **The Unix Shell**: https://swcarpentry.github.io/shell-novice/
- **R and RStudio:** https://swcarpentry.github.io/r-novice-gapminder/

Welcome to **Software Carpentry** and the **Carpentries** https://carpentries.org/
Teaching fundamental coding and data science skills to researchers worldwide.

Testimonials: https://carpentries.org/testimonials/
Yor workshop is part of the many workshops around the globe https://carpentries.org/workshops/

Hands-on Training:

- This is a hands-on training :)
- It is **interactive**, which means your interaction and awareness will improve your learning
- *Questions are always welcome!!!!!*

The Carpentries Etherpad!

We will use this Etherpad to share links, snippets of code, take notes, do exercises, ask and answer questions, and whatever else comes to mind.
This pad is synchronized as you type, so that everyone viewing this page sees the same text.
The Etherpad has three major parts:

- The *left side* holds today's notes
- The *top right* side shows the names of users who are logged in
- The *bottom right* is a real time chat window for asking questions of the instructor and your fellow learners.

*****Remember with big power come big responsibilities, do not delete parts of this etherpad* ****

Friendly, respectful and active participants!
Users are expected to follow our code of conduct:
https://docs.carpentries.org/topic_folders/policies/code-of-conduct.html

Code of Conduct (the short version)

We are dedicated to providing a welcoming and supportive environment for all people, regardless of background or identity. By participating in this community, participants accept to abide by The Carpentries' Code of Conduct and accept the procedures by which any Code of Conduct incidents are resolved. Any form of behaviour to exclude, intimidate, or cause discomfort is a violation of the Code of Conduct. In order to foster a positive and professional learning environment we encourage the following kinds of behaviours in all platforms and events:

- Use welcoming and inclusive language
- Be respectful of different viewpoints and experiences
- Gracefully accept constructive criticism
- Focus on what is best for the community
- Show courtesy and respect towards other community members.

If you believe someone is violating the Code of Conduct, we ask that you report it to The Carpentries Code of Conduct Committee, who will take the appropriate action to address the situation. To report, use this form:
https://docs.google.com/forms/d/e/1FAIpQLSdi0wbplgdydl_6rkVtBIVWbb9YNOHQP_XaANDClmVNu0zs-w/viewform

Public content page

Use of this service is restricted to members of The Carpentries community; this is not for general purpose use (for that, try etherpad.wikimedia.org).

Twitter

@thecarpentries

Instructor:

- Justin Shaffer, Ph.D. (he/him/his), California State University, Fresno, shaffer@csufresno.edu

Helpers:

- Francine Arroyo, Ph.D., California State University, Fresno
- Alija Mujic, Ph.D., California State University, Fresno

Pre-workshop survey:

Make you sure you have filled the Software Carpentry workshop pre-survey so we know what is your pre-knowledge before the workshop:

Pre-workshop survey:
https://carpentries.typeform.com/to/wi32rS?slug=2024-07-09-Software-Carpentry

---

Sign-in:

- Enter your name on the top right of this page above and hit enter!
(If you don't like your pre-assigned color, you can change it by clicking on the color next to your name above the chat window. Pick the color that best reflects your mood and personality)

- Please add your name below, and tell us something about you: your favorite kind of cake or pie / favorite fictional creature
Christopher V: tres leches or tiramisu, and there be DRAGONS, because dragon heart was heart wrenching.
Abbas: My favorite cake is tiramisu.
Keyora: My favorite pie is peach cobbler.
Nicholas: My favorite cake is vanilla/ Favorite fictional creature Micheal Scott
Ulrike: my fav cake is Zwiebelkuchen, my fav facctional character is Murderbot
Varun: My favorite kind of cake is devil's cake, my favorite fictional character is Leonard from The Big Bang Theory.
Jennifer: I love carrot cake and my favorite fictional character is a Hippogriff. (Buckbeak from Harry Potter)
Ayuba: My favorite cake is Cheese Cake. I love the spiderman character.
Ana Ramirez: My favorite cake is a tres leches cake.
Danica: My favorite cake is carrot cake (or any cake with cream cheese frosting)
Eema: My favorite cake is red velvet.
Emily: My favorite cake is carrot cake.
Kate: Favorite cake is chocolate, favorite character is any fictional detective
Melissa Jauregui: My favorite cake is Ice cream cake.
Deanna.. I might be scratching my mosquito bites, hehe lol

Hiba: My favorite kind of pie is apple pie.
Geraldine: my favorite cake is marble.
AlejandroM: Favorite cake would be tres leches con frutas
Jose Gonzalez: I love choco flan
Justin Shaffer: Key lime pie - Sea Turtle Man

Matt: My favorite cake is carrot cake. Me too!
Francine: homemade double crust apple pie/dragons

Notes:
- Shell note for Windows users: use --help for getting the manuals for commands (instead of man).

Francine: Something new I learned: Pipe in R (tidyverse) is %>% and the shortcut on Macs is CMD+Shift+M

To reformat a line of code on Mac, the shortcut is: Shift+CMD+A

Francine: ggplot cheat sheet! https://www.maths.usyd.edu.au/u/UG/SM/STAT3022/r/current/Misc/data-visualization-2.1.pdf

This is a textbook that teaches you how to further customize your graphs in ggplot2: https://r-graphics.org/

Code from Wednesday 7/10/24

```
installed.packages()
install.packages("vegan")
library(vegan)

?write.table()
?write.csv()

vignette()
vignette(package = "vegan")
vignette("decision-vegan")

??set

sessionInfo()

?dput

cats <- data.frame(coat = c("calico", "black", "tabby"),
                   weight = c(2.1, 5.0, 3.2),
                   likes_string = c(1, 0, 1))
cats

write.csv(x = cats, file = "data/feline-data.csv", row.names = FALSE)
```

```
rm(cats)

cats <- read.csv(file = "data/feline-data.csv")
cats

cats$weight
cats$coat

cats$weight - 2

paste ("My cat is", cats$coat)

cats$weight + cats$coat

typeof(cats$weight)
typeof(cats$coat)
typeof(cats$likes_string)
typeof(cats$`likes string`) # if it had spaces

typeof(3.14)
typeof(1)
typeof(1L)
typeof(TRUE)
typeof('bananaa')

str(cats)

combine_vector <- c(2,6,3)
combine_vector

coercion <- c('a', TRUE)
coercion

another_coercion <- c(0, TRUE)
another_coercion

typeof(combine_vector)
typeof(coercion)
typeof(another_coercion)

# logical -> integer -> double (numeric) -> complex character

char_vector <- c('0', '2', '4')
typeof(char_vector)

char_coerce_double <- as.double(char_vector)
typeof(char_coerce_double)

double_coerce_logical <- as.logical(char_coerce_double)
```

```
double_coerce_logical
typeof(double_coerce_logical)

c()

ab_vector <- c('a', 'b')
ab_vector

combine_example <- c(ab_vector, 'SWC')
combine_example

my_series <- 1:10

seq(10)
seq(1, 10, by = 0.1)

sequence_example <- 20:25

head(sequence_example)

tail(sequence_example)

length(sequence_example)

typeof(sequence_example)

first_element <- sequence_example[1]
first_element

sequence_example[1] <- 30
sequence_example

list_example <- list(1, 'a', TRUE, 1 + 4i)

str(list_example)

second_item_in_list <- list_example[[2]]
second_item_in_list

another_list <- list(title = "Numbers", numbers = 1:10, data = TRUE)
another_list

another_list$title

another_list["title"]

names(another_list)[1] <- "start"
another_list
```

```
typeof(cats)

str(cats)

class(cats)

cats$coat
cats[,1]
typeof(cats[,1])
str(cats[,1])

cats[1,]

names(cats)[2] <- "weight_kg"

matrix_example <- matrix(0, ncol = 6, nrow = 3)
matrix_example

dim(matrix_example)
typeof(matrix_example)
str(matrix_example)

nrow(matrix_example)
ncol(matrix_example)

matrix_example[7]
matrix_example[,3]

set.seed(1)
m <- matrix(rnorm(6*4), ncol = 4, nrow = 6, byrow=TRUE)
m
m[3:4, c(3, 1)]

age <- c(2, 3, 5)
cats
cbind(cats, age)

new_row <- list("tortoiseshell", 3.3, TRUE, 9)
cats_updated <- rbind(cats, new_row)
cats_updated

cats_updated_again <- cats_updated[c(-4,-1,-3),]
cats_updated_again

gapminder <- read.csv("data/gapminder_data.csv")
gapminder
str(gapminder)
summary(gapminder)
typeof(gapminder)
```

```
download.file("https://raw.githubusercontent.com/swcarpentry/r-novice-gapminder/main
              /episodes/data/gapminder_data.csv",
              destfile = "data/gapminder_data.csv")
gapminder <- read.csv("data/gapminder_data.csv")

install.packages("tidyverse")
library(ggplot2)

ggplot(data = gapminder)


ggplot(data = gapminder, mapping = aes(x = gdpPercap, y = lifeExp))

min(gapminder$lifeExp)
max(gapminder$lifeExp)

min(gapminder$gdpPercap)
max(gapminder$gdpPercap)

ggplot(data = gapminder, mapping = aes(x = gdpPercap, y = lifeExp)) +
  geom_point()

ggplot(data = gapminder, mapping = aes(x = year, y = lifeExp, color = continent)) +
  geom_line()

ggplot(data = gapminder,
       mapping = aes(x = year,
                     y = lifeExp,
                     group = country)) +
  geom_line(mapping = aes(color = continent)) +
  geom_point()

ggplot(data = gapminder, mapping = aes(x = gdpPercap, y = lifeExp)) +
  geom_point(alpha = 0.5) +
  scale_x_log10()

gapminder$gdpPercap_ln <- log(gapminder$gdpPercap)

ggplot(data = gapminder, mapping = aes(x = gdpPercap_ln, y = lifeExp)) +
  geom_point(alpha = 0.5)

ggplot(data = gapminder, mapping = aes(x = gdpPercap, y = lifeExp)) +
  geom_point(alpha = 0.5) +
  scale_x_log10() +
  geom_smooth(method = "lm",
              linewidth = 0.5,
              color = "orange")
```

```
americas <- gapminder[gapminder$continent == "Americas",]

ggplot(data = americas,
       mapping = aes(x = year,
                     y = lifeExp)) +
  geom_line() +
  facet_wrap(~country) +
  labs(x = "Year",
       y = "Life expectancy",
       title = "Figure 1") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

my_awesome_plot <- ggplot(data = americas,
                          mapping = aes(x = year,
                                        y = lifeExp)) +
  geom_line() +
  facet_wrap(~country) +
  labs(x = "Year",
       y = "Life expectancy",
       title = "Figure 1") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        plot.title = element_text(hjust = 0.5))

ggsave(filename = "results/lifeExp.png",
       plot = my_awesome_plot,
       width = 24,
       height = 20,
       dpi = 300,
       units = "cm")

aust_subset <- gapminder[gapminder$country == "Australia",]

write.table(aust_subset,
            file = "data/gapminder_aus.csv",
            sep = ",",
            quote = FALSE)

write.table(aust_subset,
            file = "data/gapminder_aus.txt",
            sep = "\t",
            quote = FALSE)

ggplot(data = gapminder[gapminder$continent == "Europe",], mapping = aes(x = country, y =
gdpPercap)) +
  geom_violin(outlier.alpha = 0) +
  geom_jitter(size = 0.5) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

```
library(dplyr)

mean(gapminder$gdpPercap[gapminder$continent == "Africa"])
mean(gapminder$gdpPercap[gapminder$continent == "Asia"])
mean(gapminder$gdpPercap[gapminder$continent == "Americas"])

select()
filter()
group_by()
summarize()
mutate()

year_country_gdp <- select(gapminder,
                           year,
                           country,
                           gdpPercap)

smaller_gapminder_data <- select(gapminder,
                                 -continent)

year_country_gdp <- gapminder %>% select(year, country, gdpPercap)

tidy_gdp <- year_country_gdp %>% rename(gdp_per_capita = gdpPercap)
head(tidy_gdp)

year_country_gdp_europe <- gapminder %>%
  filter(continent == "Europe") %>%
  select(year, country, gdpPercap) %>%
  rename(gdp_per_capita = gdpPercap)

head(year_country_gdp_europe)

str(gapminder)

str(gapminder %>% group_by(continent))

year_country_gdp_europe <- gapminder %>%
  group_by(continent) %>%
  select(year, country, gdpPercap) %>%
  rename(gdp_per_capita = gdpPercap)

gdp_bycontinents <- gapminder %>%
  group_by(continent) %>%
  summarize(mean_gdpPercap = mean(gdpPercap))

gdp_bycontinentsyear <- gapminder %>%
  group_by(continent, year) %>%
  summarize(mean_gdpPercap = mean(gdpPercap),
            sd_gdpPercap = sd(gdpPercap),
```

```
            mean_pop = mean(pop),
            sd_pop = sd(pop))

count()
n()

gapminder %>%
  group_by(continent) %>%
  summarize(se_le = sd(lifeExp)/sqrt(n()))

gapminder %>%
  filter(year == 2002) %>%
  count(continent, sort = TRUE)

gapminder %>%
  group_by(continent) %>%
  summarize(mean_le = mean(lifeExp),
            min_le = min(lifeExp),
            max_le = max(lifeExp),
            se_le = sd(lifeExp)/sqrt(n()))

gdp_pop_bycontinents_byyear <- gapminder %>%
  mutate(gdp_billion = gdpPercap * pop / 10^9) %>%
  group_by(continent, year) %>%
  summarize(mean_gdpPercap = mean(gdpPercap),
            sd_gdpPercap = sd(gdpPercap),
            min_gdpPercap = min(gdpPercap),
            max_gdpPercap = max(gdpPercap),
            se_gdpPercap = sd(gdpPercap)/sqrt(n()))
gdp_pop_bycontinents_byyear

gdp_pop_bycontinents_byyear_above25 <- gapminder %>%
  mutate(gdp_billion = ifelse(lifeExp > 25, gdpPercap * pop / 10^9, NA)) %>%
  group_by(continent, year) %>%
  summarize(mean_gdpPercap = mean(gdpPercap),
            sd_gdpPercap = sd(gdpPercap),
            mean_pop = mean(pop),
            sd_pop = sd(pop))

gdp_pop_bycontinents_byyear_above25 <- gapminder %>%
  mutate(gdp_billion = ifelse(lifeExp > 25, gdpPercap * pop / 10^9, gdpPercap)) %>%
  group_by(continent, year) %>%
  summarize(mean_gdpPercap = mean(gdpPercap),
            sd_gdpPercap = sd(gdpPercap),
            mean_pop = mean(pop),
            sd_pop = sd(pop))


# ggplot with dplyr
```

```
# dplyr way to subset
gapminder %>%
  filter(continent == "Americas") %>%
  ggplot(mapping = aes(x = year, y = lifeExp)) +
  geom_line() +
  facet_wrap(~country) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

# base R way to subset
ggplot(data = gapminder[gapminder$continent == "Europe",], mapping = aes(x = country, y =
gdpPercap)) +
  geom_violin(outlier.alpha = 0) +
  geom_jitter(size = 0.5) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

# creating maps containing sample locations as points

# Set working directory
setwd("data/")

# Install and load packages
install.packages("tmap")
install.packages("ggspatial")
install.packages("sf")
install.packages("rnaturalearth")
install.packages("rnaturalearthdata")

library(tmap)
library(ggspatial)
library(sf)
library(rnaturalearth)
library(rnaturalearthdata)

# Read in metadata
md <- read_tsv("emp500_metadata_basic.txt", na = c("", "NA", "not applicable"))



# Add in columns for colors based on EMPO categories
md_noControls$colors_empo_4 <- as.factor(md_noControls$empo_4)
levels(md_noControls$colors_empo_4)
levels(md_noControls$colors_empo_4) <- c("purple4",
                                         "purple4",
                                         "darkmagenta",
                                         "darkmagenta",
                                         "plum3",
                                         "plum3",
                                         "pink1",
                                         "orange",
```

```
                                "green4",
                                "green3",
                                "green3",
                                "wheat3",
                                "wheat3",
                                "chocolate4",
                                "chocolate4",
                                "gray75",
                                "gray45",
                                "mediumblue",
                                "mediumblue")

# Create column to represent shape based on salinity
md_noControls$shape_empo_salinity <- as.factor(md_noControls$empo_salinity)
levels(md_noControls$shape_empo_salinity) <- c("21", "22")
md_noControls$shape_empo_salinity <- as.numeric(levels(md_noControls$shape_empo_salinity))
[md_noControls$shape_empo_salinity]

# Create map
world <- ne_countries(scale = "medium", returnclass = "sf")
class(world)

md_noControls_sf <- st_as_sf(as.data.frame(md_noControls),
                             coords = c("longitude", "latitude"),
                             agr = "constant",
                             crs = 4326)

ggplot(data = world) +
  theme_bw() +
  geom_sf() +
  xlab('Longitude') +
  ylab('Latitude') +
  geom_sf(data = md_noControls_sf,
          size = 3,
          shape = md_noControls$shape_empo_salinity,
          fill = md_noControls$colors_empo_4,
          color = "black",
          alpha = 0.5) +
  annotation_north_arrow(location = "bl", which_north = "true",
                         pad_x = unit(0.1, "in"), pad_y = unit(0.3, "in"),
                         style = north_arrow_fancy_orienteering) +
  coord_sf(expand = F)
```