Dates and Times with `lubridate`

Contents

This lab will cover select functions in the `lubridate` package that are useful in working with dates and times:

```
# install.packages("lubridate")
library(lubridate)
```

Practice Problems

Question 1: On January 27th 1967 at 6:31 PM, the Apollo 1 spacecraft, planned to be the first manned mission of the Apollo space program, experienced a cabin fire on the landing pad in Cape Kennedy Air Force Station, Florida during a launch simulation, killing all three crew members on board. Nearly 19 years later, on January 28, 1986 at 11:39 AM, the Challenger Shuttle exploded just off the coast of Cape Canaveral, Florida. Rounding each date to the nearest day, determine how many days passed between these two events.

```
apollo <- "1986 Jan 27th at 6:31:19 PM UTC"
challenger <- "28 January 1967, 1139am"
```

```
t1 <- ymd_hms(apollo)
t2 <- dmy_hm(challenger)

round_date(t1, unit = "day") - round_date(t2, unit = "day")
```

```
## Time difference of 6940 days
```

Question 2: Create a date/time object for 9:15pm in Los Angeles on February 14, 2020. Then, round this date to the nearest day, then determine which day of the week that day was.

```
date_q2 = as.POSIXct("02/14/2020 9:15", format = "%m/%d/%Y %H:%M",
                     tz = "America/Los_Angeles")
round_date(date_q2, unit = "day") %>% wday(label=TRUE)
```

```
## [1] Fri
## Levels: Sun < Mon < Tue < Wed < Thu < Fri < Sat
```

Question 3: The 2015 Boston Marathon took place on April 20th, 2015. It was the 119th running of one of the world's most well-known races. The data below contain information, results, and splits for each finisher of the marathon:

```
#marathon <- read.csv("https://collinn.github.io/data/BostonMarathon2015.csv")
marathon <- read.csv("~/gitsite/data/BostonMarathon2015.csv")
```

- **Part A**: A marathon is approximately 26.2 miles, making the first half 13.1 miles. Calculate the per mile pace (in seconds) for each participant in the first half of the race. Be sure to store your results.
- **Part B**: Now calculate the pace per mile in the second half of the race. Be sure to store your results.

- **Part C**: Now create a scatter plot displaying the relationship between pace per mile in the first half of the race vs. pace per mile in the second half of the race by age and sex. To do this, you should assemble your results from Parts A and B into a data frame, and you should also include the "Age" and "M.F" columns from the original data when you create this data frame. A target graphic is given below. *Note*: `scale_x_time()` and `scale_y_time()` can be used to display your first half and second half paces on a time scale. The graph shown below uses the argument `alpha = 0.2` to reduce the impact of over-plotting, and a 45-degree line is added using `geom_abline()`.

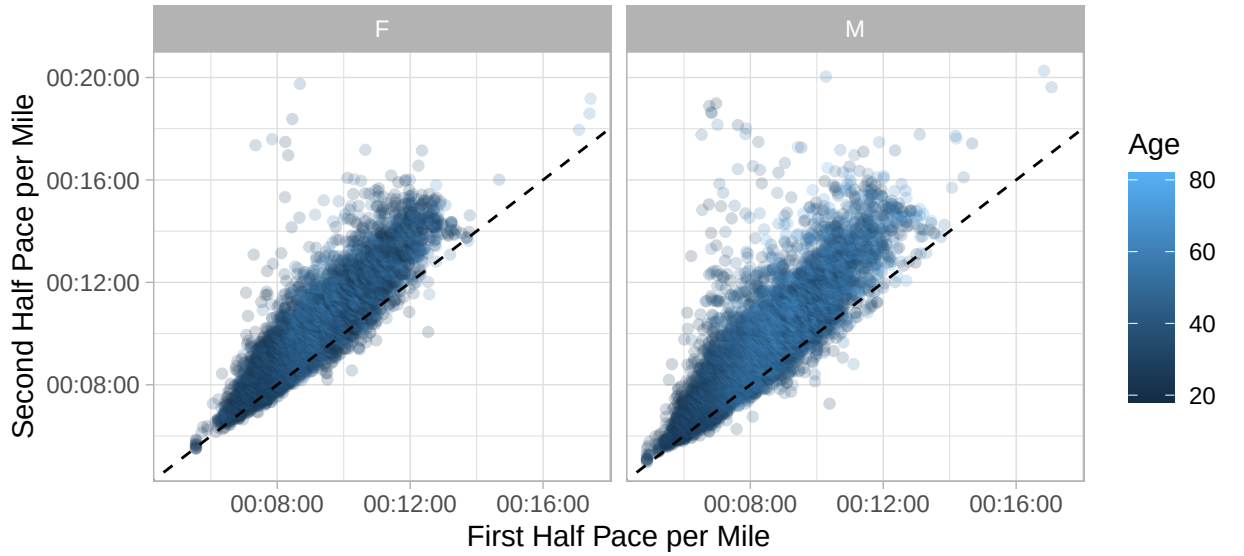
Note: If you get an error that `hms` is not available, you need to install it with `install.packages("hms")`. However, do not load the entire package with `library(hms)`, as this will cause problems with the `lubridate` package. Instead, to use a function from a package without loading the entire package, we use the double colon `::` with the form `packagename::function`. In this case, we are wanting to use the function `scale_x_time()` from the package `hms`, hence `hms::scale_x_time()`. If you do not get this error, you can ignore this paragraph.

```
library(dplyr)
library(ggplot2)
## PART A
first_pace <- lubridate::hms(marathon$Half) %>% seconds() / 13.1

## PART B
second_pace <- (lubridate::hms(marathon$Official.Time) -
                  lubridate::hms(marathon$Half)) %>% seconds() / 13.1

## Part C
plot_df <- data.frame(Age = marathon$Age, Sex = marathon$M.F,
                      first_pace = first_pace, second_pace = second_pace)

ggplot(plot_df, aes(x = first_pace, y = second_pace, color = Age)) +
  geom_point(alpha = 0.2) + facet_wrap(~Sex) +
  scale_x_time() + scale_y_time() +
  geom_abline(intercept = 0, slope = 1, linetype = "dashed") +
  theme_light() +
  labs(x = "First Half Pace per Mile", y = "Second Half Pace per Mile")
```



Question 4: The two files below contain results from a real experiment on cannabis impaired driving conducted in an advanced driving simulator. The file "startdose.csv" is a list of participant IDs and the time they started a 10-min *ad libitum* dose of inhaled cannabis, and the file "high_effects.csv" records each participant's self-reported feelings of "high" (on a 0-100 scale) at various points in the experiment.

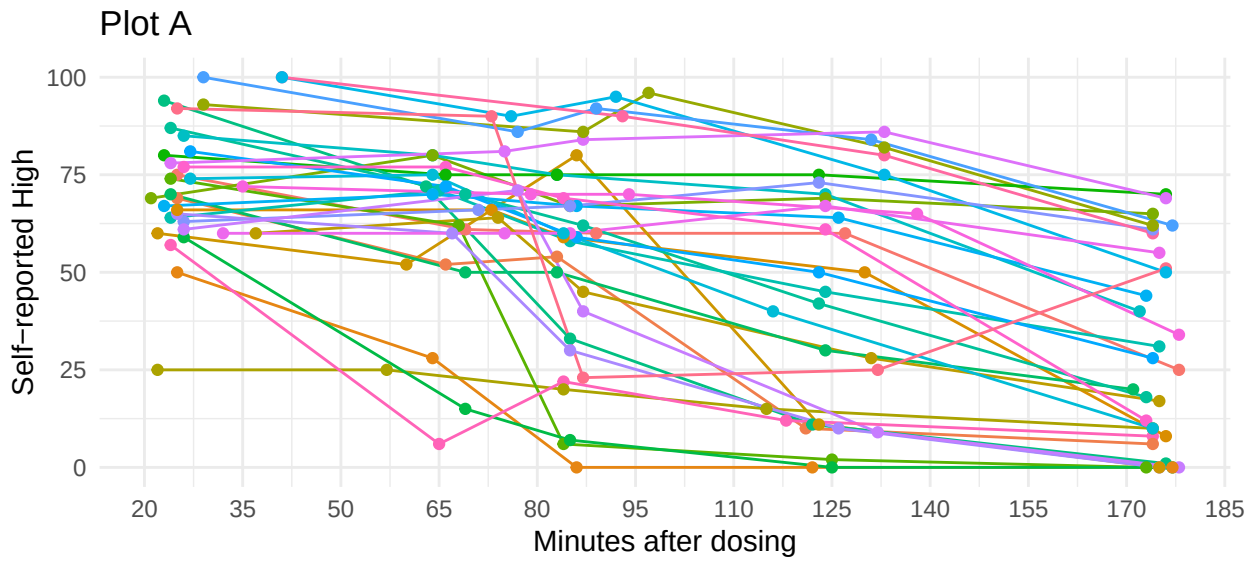
The two files below contain results from a real experiment on cannabis impaired driving in an advanced driving simulator. The first CSV, `startdose.csv`, records a list of participant ideas and the time they initially inhaled cannabis. The second CSV, `high_effects.csv`, records each participants' self-reported feelings of being "high" and on 0-100 scale at various points in the experiment

```
dose <- read.csv("https://remiller1450.github.io/data/startdose.csv")
high <- read.csv("https://remiller1450.github.io/data/high_effects.csv")
```

- **Part A:** Begin by preparing the data to be used in the visualization below. You should start by joining the datasets according to the correct key, noting that there may be participants recorded in one file that are not in the other; we should use complete entries only. Then, you should construct a new variable `MAD`, or "minutes after dosing", to create a dataset that contains the users self-reported high in minutes from onset. You should filter out all values where `Time == 1`. Once finished, reproduce the plot from `Plot 1` as closely as you can.
- **Part B:** Next, we want to reduce this sample to only individuals whose self-reported level of high at the end of the experiment was 25% or less than it was at the beginning of the experiment (e.g., a person with an initial highness of 80 should end the experiment with a level of 20 or less). See `?first` for a collection of useful functions. Once you have this, reproduce as closely as possible the visualization in `Plot B`

```
library(dplyr)
library(tidyr)
library(ggplot2)
dose$Subject <- as.character(dose$SubjectID)
res <- inner_join(high, dose, by = "Subject") %>%
  mutate(diff = mdy_hm(Time.1, tz = "America/Chicago") -
           mdy_hm(Start, tz = "America/Chicago") ) %>%
  filter(Time != 1)

ggplot(res, aes(x = diff, y = High, color = Subject, group = Subject)) +
  geom_line() + geom_point() + guides(color = "none") +
  labs(x = "Minutes after dosing", y = "Self-reported High", title = "Plot A") +
  scale_x_continuous(breaks = seq(20,190, by = 15)) + theme_minimal()
```




```
res <- group_by(res, Subject) %>%
  mutate(high_change = last(High) / first(High)) %>%
  filter(high_change < 0.25)

ggplot(res, aes(x = diff, y = High, color = Subject, group = Subject)) +
  geom_line() + geom_point() + guides(color = "none") +
  labs(x = "Minutes after dosing", y = "Self-reported High", title = "Plot B") +
  scale_x_continuous(breaks = seq(20,190, by = 15)) + theme_minimal()
```