Data Jujutsu I – The name game[*]

Stefano Allesina & Graham Smith *University of Chicago*

Description of the data

The Social Security Administration releases every year data on first names as reported on Social Security cards. Basically, all names that were given to 5+ people in a given year are reported (details here). The data goes back to 1880, and the latest release covers all years until 2018. The data are organized by year, and contained in a zip file that you can access at `https://www.ssa.gov/oact/babynames/names.zip` (about 7Mb). For each year, a file (e.g. `yob2012.txt`) contains information on all the names, organized in three columns (**no header**):

```
Sophia,F,22313
Emma,F,20945
Isabella,F,19099
Olivia,F,17316
Ava,F,15533
...
```

The first column is the name (with spaces and hypens removed: `Mary-Jane` → `Maryjane`), the second column the recorded sex (`F` or `M`) and the third column the number of babies for a given sex that were given that name.

The challenge

1. Write code to download, parse and organize the data, building a tibble with columns `year`, `name`, `sex`, `prop`, where `prop` is the proportion of babies with a given name for a year and sex combination.

```
source("solution_name_game.R") # this is the code **you** have to write
all_names # store all information here
```

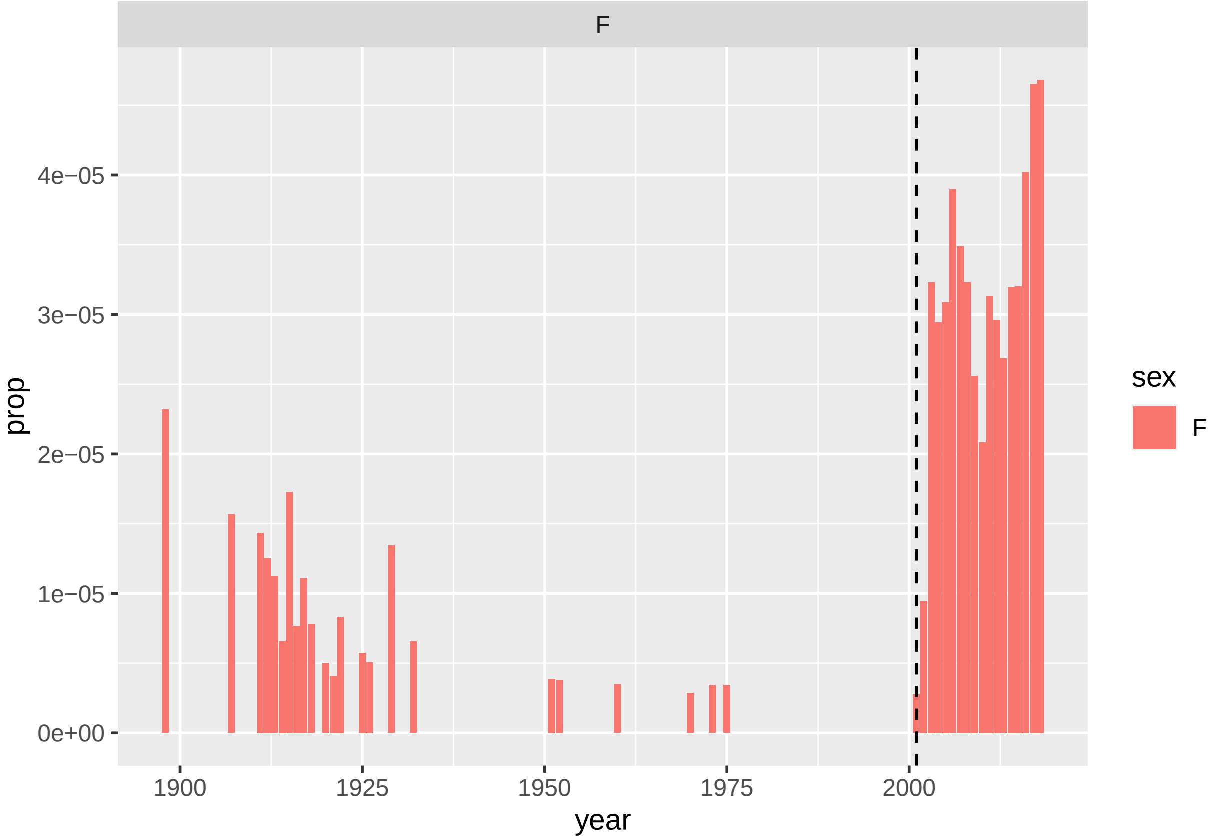
```
## # A tibble: 1,957,046 x 4
##     year name      sex     prop
##    <dbl> <chr>     <chr>  <dbl>
##  1  1880 Mary      F     0.0776
##  2  1880 Anna      F     0.0286
##  3  1880 Emma      F     0.0220
##  4  1880 Elizabeth F     0.0213
##  5  1880 Minnie    F     0.0192
##  6  1880 Margaret  F     0.0173
##  7  1880 Ida       F     0.0162
```

[*] This document is included as part of the Advanced Computing I packet for the U Chicago BSD qBio6 boot camp 2020. **Current version**: July 06, 2020; **Corresponding author**: sallesina@uchicago.edu.

```
##  8  1880 Alice     F     0.0155
##  9  1880 Bertha    F     0.0145
## 10  1880 Sarah     F     0.0142
## # ... with 1,957,036 more rows
```

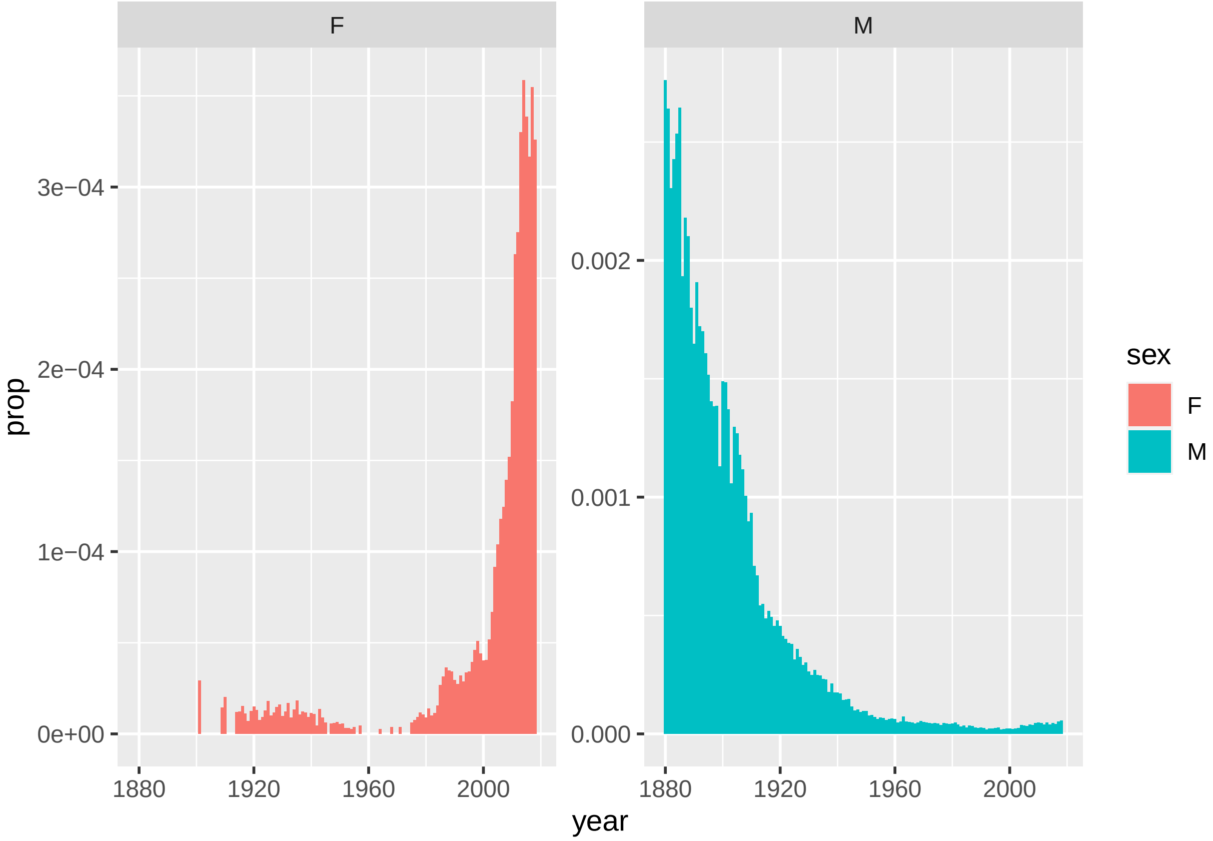
2. Write a function that plots the frequency in time, and accepts an extra parameter to highlight a particular year (a vertical dashed line marks the highlighted year). For example:

```
plot_name_in_time(data = all_names, my_name = "Hermione", highlight = 2001)
```



3. Try to find name/year combinations where the name frequency changes significantly due to people/event in popular culture (e.g., "Harry Potter and the Sorcerer's stone" movie came out in 2001; other examples are Ariel/1989, Alanis/1995, Beyonce/1998, Neo/1999, Osama/2001, Barack/2008, Ivanka/2016, etc.). Be prepared to share the most interesting combinations with the class.
4. (Optional) Certain names went from being given predominantly to boys to more frequently to girls, or vice versa. Write code to discover the most impressive transitions. For example, "Charley" was predominantly assigned to boys back in the day, while today is mostly assigned to girls:

```
plot_name_in_time(data = all_names, my_name = "Charley", highlight = NULL)
```



Hints & Nifty tricks

- If you don't want to store the downloaded zip file, use a temporary file (it will be deleted by R automatically once you call `unlink()`):

```
temp <- tempfile()
download.file("https://www.ssa.gov/oact/babynames/names.zip",temp)
```

- Similarly, you don't need to extract all the files and store them on your disk: you can use `unz` to extract a file from a zip file, and use a connection to read it directly from the memory. For example:

```
con <- unz(temp, "yob2012.txt")
dt <- read.table(con, header = FALSE, sep = ",", stringsAsFactors = FALSE)
```

- Don't read the names as `factors` (the default in `read.table`) as binding together long lists of factors takes much time.
- Save your elaborated data, to save the time it takes to download and parse the data when you re-run your code. For extra brownie points, write your code such that it checks if the data have been parsed already before starting the download. You can use `file.exists()` to check whether a file is present or not.