DataSci 306, Homework 2

Max Han, maxhan

```
knitr::opts_chunk$set(echo = TRUE)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(lubridate)
library(nycflights13)
library(babynames)
```

Question 1 (2 points)

For this challenge we will use the `babynames` dataset that is already loaded in the above code block (refer: `library(babynames)`)

```
babynames |> head(100)
```

```
## # A tibble: 100 x 5
##     year sex   name          n   prop
##    <dbl> <chr> <chr>     <int>  <dbl>
##  1  1880 F     Mary       7065 0.0724
##  2  1880 F     Anna       2604 0.0267
##  3  1880 F     Emma       2003 0.0205
##  4  1880 F     Elizabeth  1939 0.0199
##  5  1880 F     Minnie     1746 0.0179
##  6  1880 F     Margaret   1578 0.0162
##  7  1880 F     Ida        1472 0.0151
##  8  1880 F     Alice      1414 0.0145
##  9  1880 F     Bertha     1320 0.0135
## 10  1880 F     Sarah      1288 0.0132
## # i 90 more rows
```

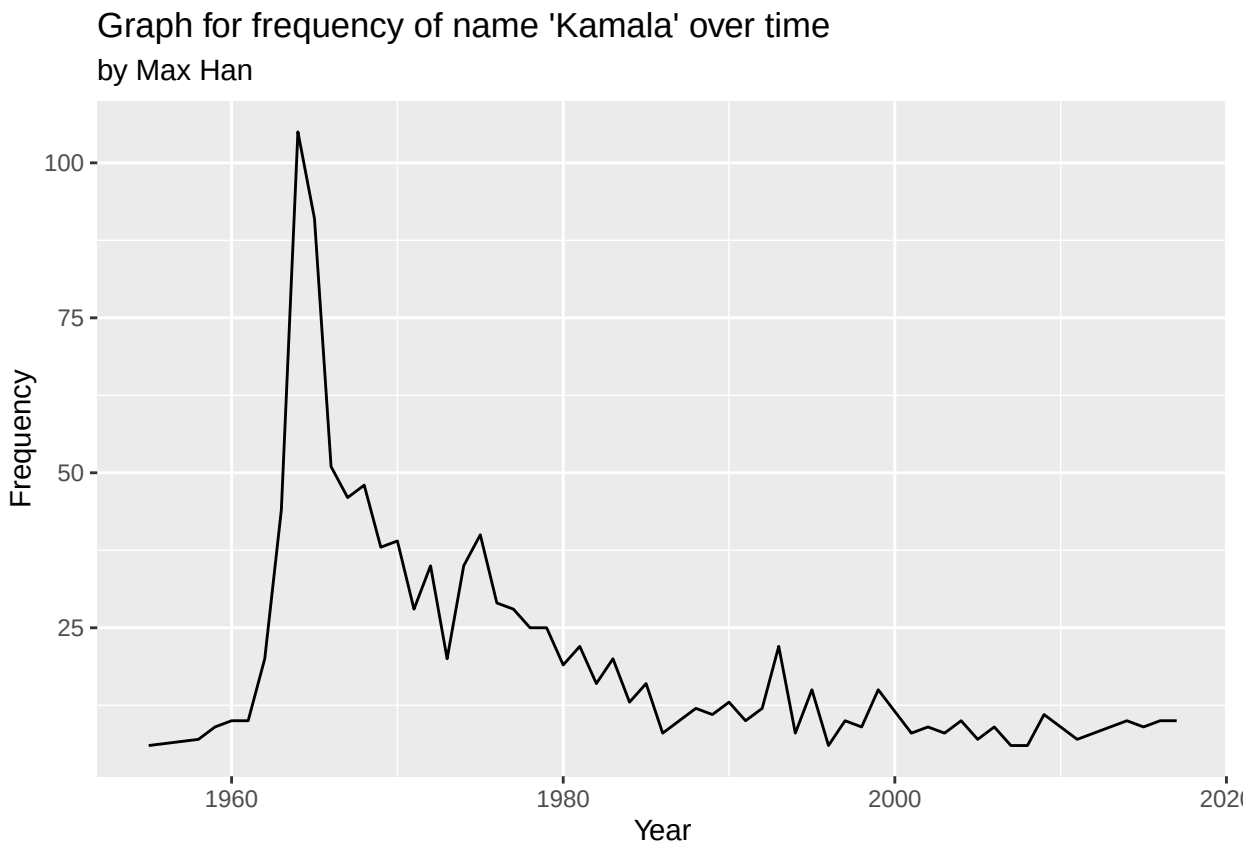
To learn more on this data, run `?babynames` from your R console

(a) Plot the frequency (the `n` variable in the data, denotes the frequency) of the name `Kamala` over the years. (1 point)

```
babynames |> filter(name == "Kamala") |>
  ggplot(aes(x = year, y =n)) +
```

```
  geom_line(color = "black", fill = "white") +
  labs(title = "Graph for frequency of name 'Kamala' over time",
       subtitle = "by Max Han",
       x = "Year",
       y = "Frequency")
```

```
## Warning in geom_line(color = "black", fill = "white"): Ignoring unknown
## parameters: `fill`
```



(b) Find the three most popular baby names in the year 1964 (1 point)

```
babynames |> filter(year == 1964) |> slice_max(n, n = 3)
```

```
## # A tibble: 3 x 5
##    year sex   name        n   prop
##   <dbl> <chr> <chr>   <int>  <dbl>
## 1  1964 M     Michael 82653 0.0408
## 2  1964 M     John    82529 0.0407
## 3  1964 M     David   75056 0.0370
```

Question 2 (5 points)

Manipulating `flights` data

In this problem, we will explore `nycflights` data. This data is loaded for you in the above code block (refer: `library(nycflights13)`)

```
flights |> head()
```

```
## # A tibble: 6 x 19
##    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
## 1  2013     1     1      517            515         2      830            819
## 2  2013     1     1      533            529         4      850            830
## 3  2013     1     1      542            540         2      923            850
## 4  2013     1     1      544            545        -1     1004           1022
## 5  2013     1     1      554            600        -6      812            837
## 6  2013     1     1      554            558        -4      740            728
## # i 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

Learn more on this data by invoking `?flights` on your console. Then answer the below questions

(a) How many flights had dep_delay (i.e., departure delay) value of more than 2 hours? (0.5 point)

```
flights |> filter(dep_delay > 120) |> count()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1  9723
```

(b) Mutating a new column (1.5 pt)

Add a new column called `status` that marks the flight as 'Delayed' if the dep_delay value is more than 10. If the dep_delay is < 0, then mark it is as 'Early'. Rest of them mark it as 'On-time'

Then select only `dep_delay` and the new column `status` to display the first 30 records

Hint: Will you use `if_else` or `ifelse`? There are other ways to solve it as well (you could solve it differently too) but we have so far understood the use of if-else statements.

```
flights_1 <- mutate(flights,
                    status = if_else(dep_delay > 10, "Delayed",
                                     if_else(dep_delay < 0, "Early", "On-time")))
select(flights_1, dep_delay, status) |> head(30)
```

```
## # A tibble: 30 x 2
##    dep_delay status
##        <dbl> <chr>
##  1         2 On-time
##  2         4 On-time
##  3         2 On-time
##  4        -1 Early
##  5        -6 Early
##  6        -4 Early
##  7        -5 Early
##  8        -3 Early
##  9        -3 Early
## 10        -2 Early
## # i 20 more rows
```

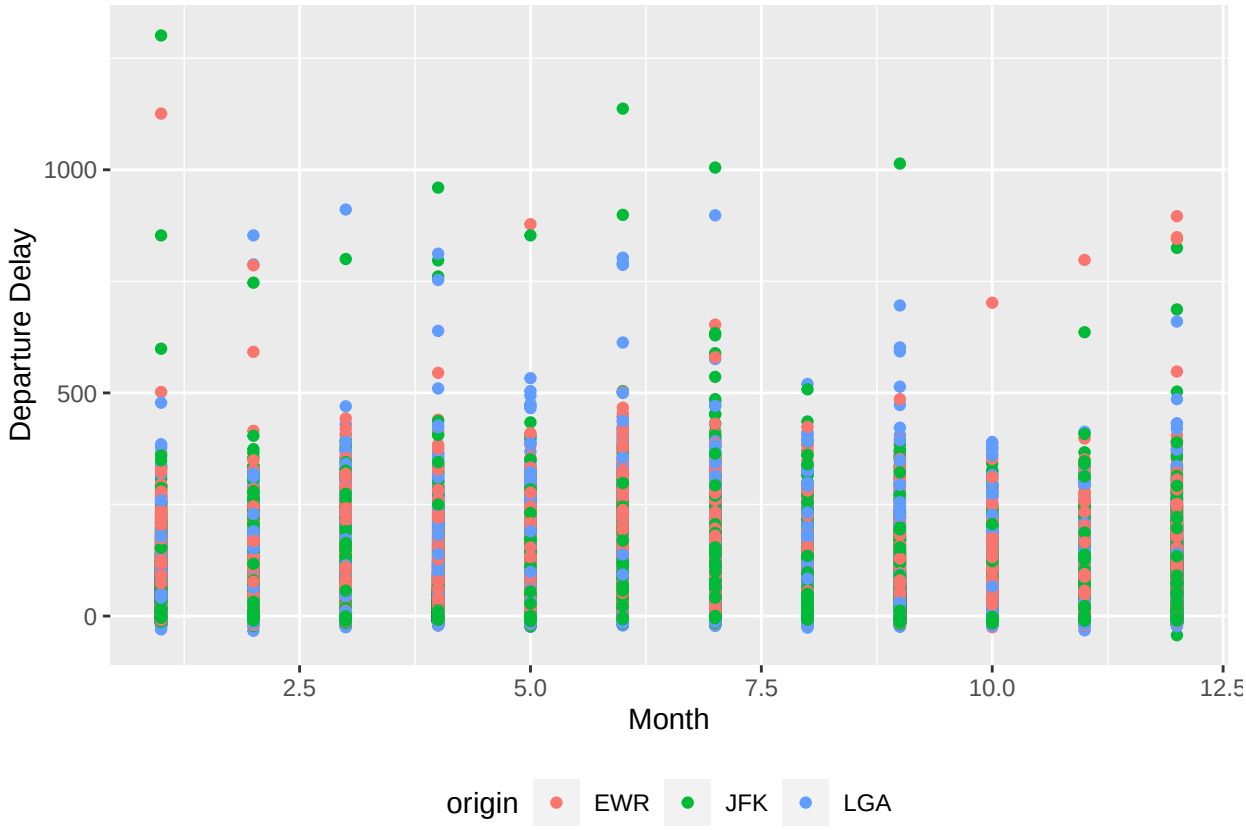
(c) Delays by origin (2 points)

Here is a plot showing the departure delays over time, attempting to breakdown the data by origin using color

```
flights |>
  ggplot(aes(x = month, y = dep_delay, color = origin)) +
```

```
  geom_point() +
  theme(legend.position="bottom") +
  labs(y = "Departure Delay",
       x = "Month")
```

```
## Warning: Removed 8255 rows containing missing values (`geom_point()`).
```

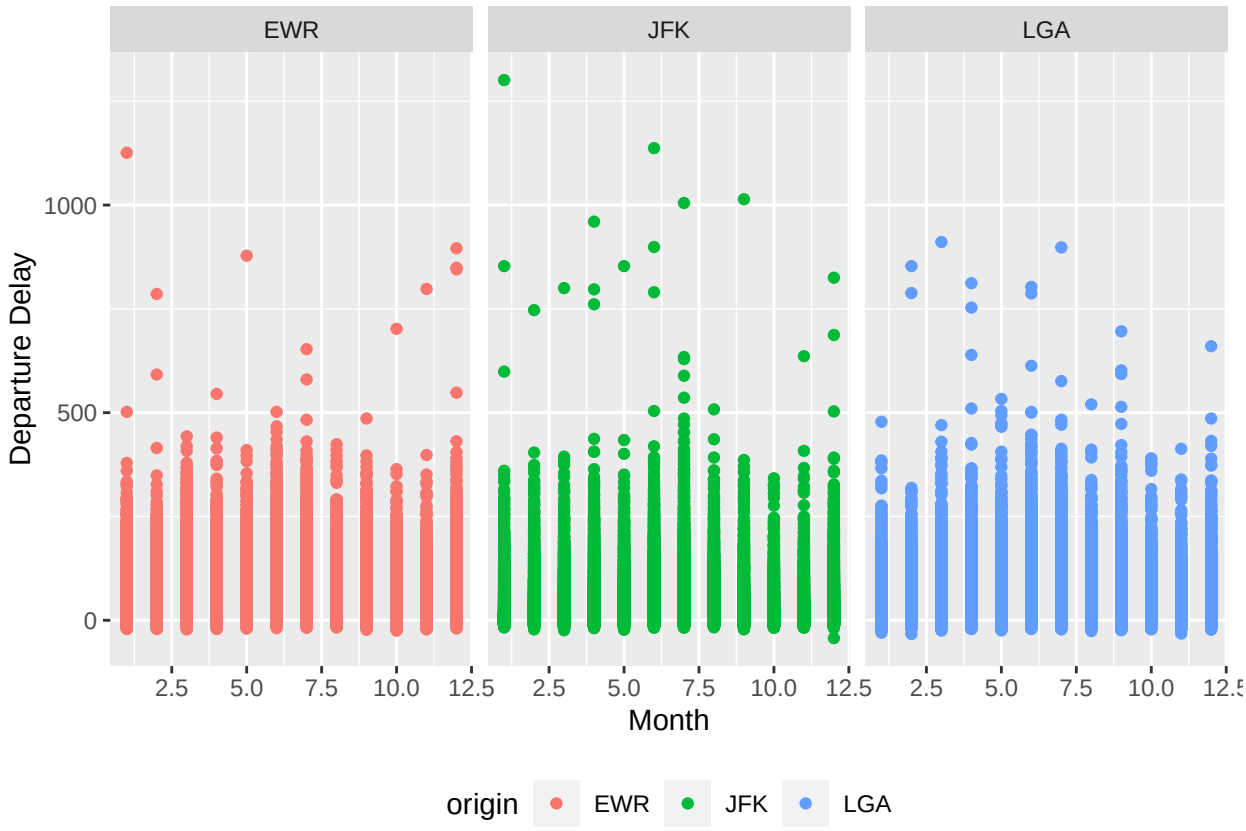


What would be a better way to display this data? Implement a better plot and explain why it is better. Also answer, which airport shows the biggest issues with departure delays?

Hint: Explore how `facet` chart can help. Also for over-plotting situation, which geometry would work better? What about the color? Can you set a different variable for color to make it more readable?

```
flights |>
  ggplot(aes(x = month, y = dep_delay, color = origin)) +
  geom_point() +
  facet_wrap(~ origin) +
  theme(legend.position="bottom") +
  labs(y = "Departure Delay",
       x = "Month")
```

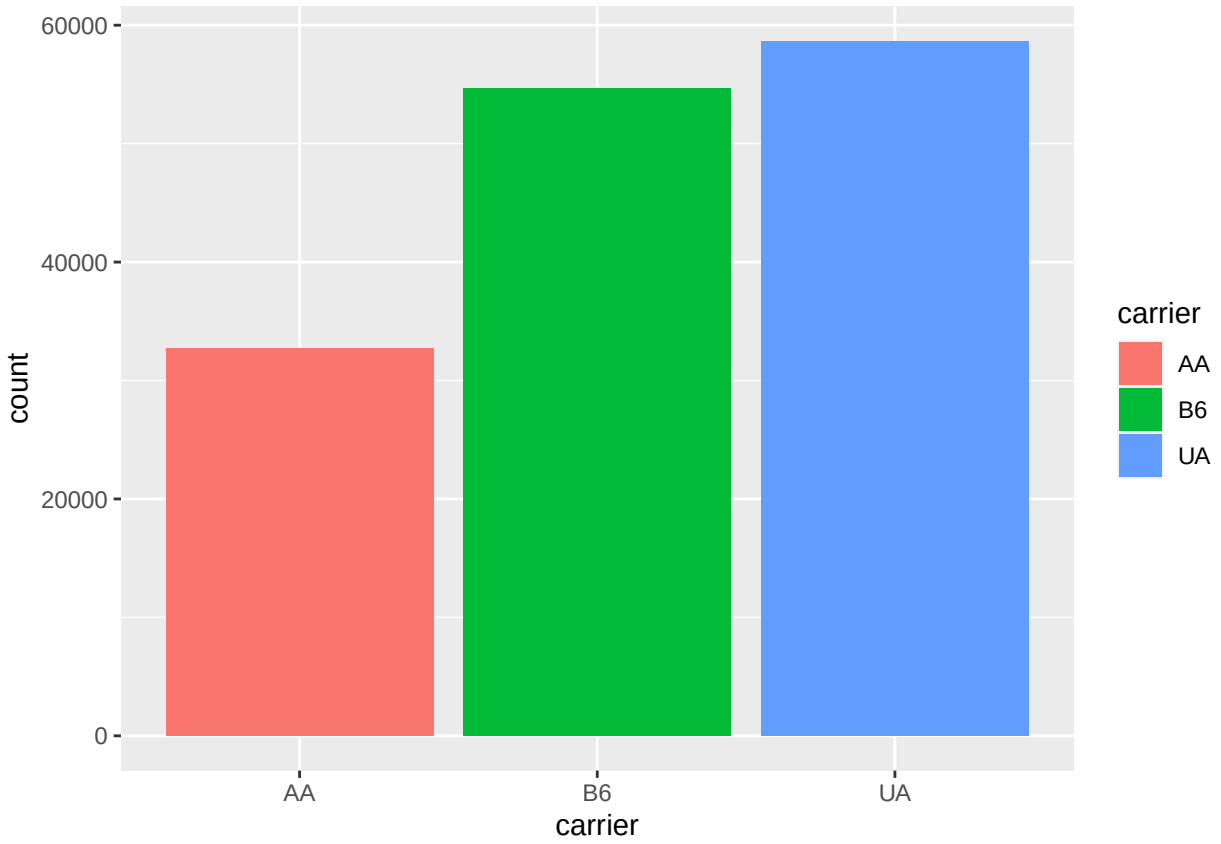
```
## Warning: Removed 8255 rows containing missing values (`geom_point()`).
```



(d) More filtering (1 point)

Select only those rows that belong to carriers 'UA', 'AA' and 'B6' and show the distribution of flights for these three carriers using a bar chart

```
flights |> filter(carrier %in% c("UA", "AA", "B6")) |>
  ggplot(aes (x = carrier, fill = carrier)) +
  geom_bar()
```



Question 3 - Challenge problem (3 points)

Spooky flights

Define a flight to be spooky if it was in transit at 13:13 h (i.e., 1:13 pm) on Friday the 13th of any month. You can assume the flight is in transit between its `dep_time` and `arr_time`.

Hint: You need to extract the weekday. Explore functions from `lubridate` package to extract the weekday.

```
# If year, month, and day are given, checks if the date is Friday.
is_friday <- function(year, month, day) {
  date <- make_date(year, month, day)
  return (wday(date, label = TRUE) == "Fri")
}

# Checks if the range between dep_time and arr_time contains 1313.
is_spooky_time <- function(dep_time, arr_time) {
  return (dep_time < 1313 & arr_time > 1313)
}

# Filters the data according to the given criteria.
spooky_flights <- flights |> filter(day == 13 &
                                    is_friday(year, month, day) &
                                    is_spooky_time(dep_time, arr_time))

spooky_flights
```

```
## # A tibble: 233 x 19
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
```

```
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013    12    13      900            900         0     1359           1350
##  2  2013    12    13      929            930        -1     1527           1535
##  3  2013    12    13      937            930         7     1514           1527
##  4  2013    12    13      947            947         0     1427           1430
##  5  2013    12    13      958           1000        -2     1320           1321
##  6  2013    12    13     1011           1015        -4     1344           1342
##  7  2013    12    13     1012           1015        -3     1337           1337
##  8  2013    12    13     1014           1019        -5     1347           1350
##  9  2013    12    13     1020           1025        -5     1338           1340
## 10  2013    12    13     1023           1030        -7     1359           1410
## # i 223 more rows
## # i 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```