IMT 573: Problem Set 1 - Exploring Data

Your Name Here

Due: Tuesday, October 08, 2019

Collaborators:

Instructions:

Before beginning this assignment, please ensure you have access to R and RStudio; this can be on your own personal computer or on the IMT 573 R Studio Server.

1. Download the `problemset1.Rmd` file from Canvas or save a copy to your local directory on RStudio Server. Open `problemset1.Rmd` in RStudio and supply your solutions to the assignment by editing `problemset1.Rmd`.
2. Replace the "Insert Your Name Here" text in the `author:` field with your own full name. Any collaborators must be listed on the top of your assignment.
3. Be sure to include well-documented (e.g. commented) code chucks, figures, and clearly written text chunk explanations as necessary. Any figures should be clearly labeled and appropriately referenced within the text. Be sure that each visualization adds value to your written explanation; avoid redundancy – you do no need four different visualizations of the same pattern.
4. Collaboration on problem sets is fun and useful, and we encourage it, but each student must turn in an individual write-up in their own words as well as code/work that is their own. Regardless of whether you work with others, what you turn in must be your own work; this includes code and interpretation of results. The names of all collaborators must be listed on each assignment. Do not copy-and-paste from other students' responses or code.
5. All materials and resources that you use (with the exception of lecture slides) must be appropriately referenced within your assignment.
6. Remember partial credit will be awarded for each question for which a serious attempt at finding an answer has been shown. Students are *strongly* encouraged to attempt each question and to document their reasoning process even if they cannot find the correct answer. If you would like to include R code to show this process, but it does not run withouth errors you can do so with the `eval=FALSE` option as follows:

```
a + b # these object dont' exist
# if you run this on its own it with give an error
```

7. When you have completed the assignment and have **checked** that your code both runs in the Console and knits correctly when you click `Knit PDF`, rename the knitted PDF file to `Yps1_ourLastName_YourFirstName.pdf`, and submit the PDF file on Canvas.

Setup:

In this problem set you will need, at minimum, the following R packages.

```
# Load standard libraries
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 3.6.1
```

```
## Warning: package 'dplyr' was built under R version 3.6.1
```

```
library(nycflights13)
```

```
## Warning: package 'nycflights13' was built under R version 3.6.1
```

Problem 1: Exploring the NYC Flights Data

In this problem set we will use the data on all flights that departed NYC (i.e. JFK, LGA or EWR) in 2013. You can find this data in the `nycflights13` R package.

```
# Load the nycflights13 library which includes data on all
# lights departing NYC
data(flights)
# Note the data itself is called flights, we will make it into a local df
# for readability
flights <- tbl_df(flights)
# Look at the help file for information about the data
# ?flights
flights
```

```
## # A tibble: 336,776 x 19
##     year month   day dep_time sched_dep_time dep_delay arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>
##  1  2013     1     1      517            515         2      830
##  2  2013     1     1      533            529         4      850
##  3  2013     1     1      542            540         2      923
##  4  2013     1     1      544            545        -1     1004
##  5  2013     1     1      554            600        -6      812
##  6  2013     1     1      554            558        -4      740
##  7  2013     1     1      555            600        -5      913
##  8  2013     1     1      557            600        -3      709
##  9  2013     1     1      557            600        -3      838
## 10  2013     1     1      558            600        -2      753
## # ... with 336,766 more rows, and 12 more variables: sched_arr_time <int>,
## #   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>
```

```
# summary(flights)
```

(a) Importing and Inspecting Data

Load the data and describe in a short paragraph how the data was collected and what each variable represents. Perform a basic inspection of the data and discuss what you find.

```
View(flights)
colnames(flights)
```

```
##  [1] "year"           "month"          "day"            "dep_time"
##  [5] "sched_dep_time" "dep_delay"      "arr_time"       "sched_arr_time"
##  [9] "arr_delay"      "carrier"        "flight"         "tailnum"
## [13] "origin"         "dest"           "air_time"       "distance"
## [17] "hour"           "minute"         "time_hour"
```

```
head(flights)
```

```
## # A tibble: 6 x 19
##    year month   day dep_time sched_dep_time dep_delay arr_time
```

```
##   <int> <int> <int>    <int>          <int>     <dbl>    <int>
## 1  2013     1     1      517            515         2      830
## 2  2013     1     1      533            529         4      850
## 3  2013     1     1      542            540         2      923
## 4  2013     1     1      544            545        -1     1004
## 5  2013     1     1      554            600        -6      812
## 6  2013     1     1      554            558        -4      740
## # ... with 12 more variables: sched_arr_time <int>, 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>
```

```
tail(flights)
```

```
## # A tibble: 6 x 19
##    year month   day dep_time sched_dep_time dep_delay arr_time
##   <int> <int> <int>    <int>          <int>     <dbl>    <int>
## 1  2013     9    30       NA           1842        NA       NA
## 2  2013     9    30       NA           1455        NA       NA
## 3  2013     9    30       NA           2200        NA       NA
## 4  2013     9    30       NA           1210        NA       NA
## 5  2013     9    30       NA           1159        NA       NA
## 6  2013     9    30       NA            840        NA       NA
## # ... with 12 more variables: sched_arr_time <int>, 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>
```

```
stat(flights)
```

```
## # A tibble: 336,776 x 19
##     year month   day dep_time sched_dep_time dep_delay arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>
##  1  2013     1     1      517            515         2      830
##  2  2013     1     1      533            529         4      850
##  3  2013     1     1      542            540         2      923
##  4  2013     1     1      544            545        -1     1004
##  5  2013     1     1      554            600        -6      812
##  6  2013     1     1      554            558        -4      740
##  7  2013     1     1      555            600        -5      913
##  8  2013     1     1      557            600        -3      709
##  9  2013     1     1      557            600        -3      838
## 10  2013     1     1      558            600        -2      753
## # ... with 336,766 more rows, and 12 more variables: sched_arr_time <int>,
## #   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>
```

The data is a collection of all flights departing from airports in New York City and contains information regarding the date of flights, delays in arrival and/or departure alongwith place of origin and destination. It also contains information about the time of flights and scheduled arrival and departure times.

year: Integer value for year month:Integer value for month (1 = Jan, 2 = Feb) day: Integer value for date of the month dep_time: Actual time of departure (accounts for delays) arr_time: Actual time of arrival (accounts for delays) sched_dep_time: Scheduled time for departure sched_arr_time:Scheduled time for arrival dep_delay: Departure delay in mins (negative value indicates early departure) arr_delay:

Arrival delay in mins (negative value indicates early arrival) carrier: Name of the airline flight/ tail number : additional information for airplane identification airtime: Amount of time spent in the air distance: distance between airports in miles hour, minute : time of departure broken down into hours and minutes ##### (b) Formulating Questions

Consider the NYC flights data. Formulate two motivating questions you want to explore using this data. Describe why these questions are interesting and how you might go about answering them.

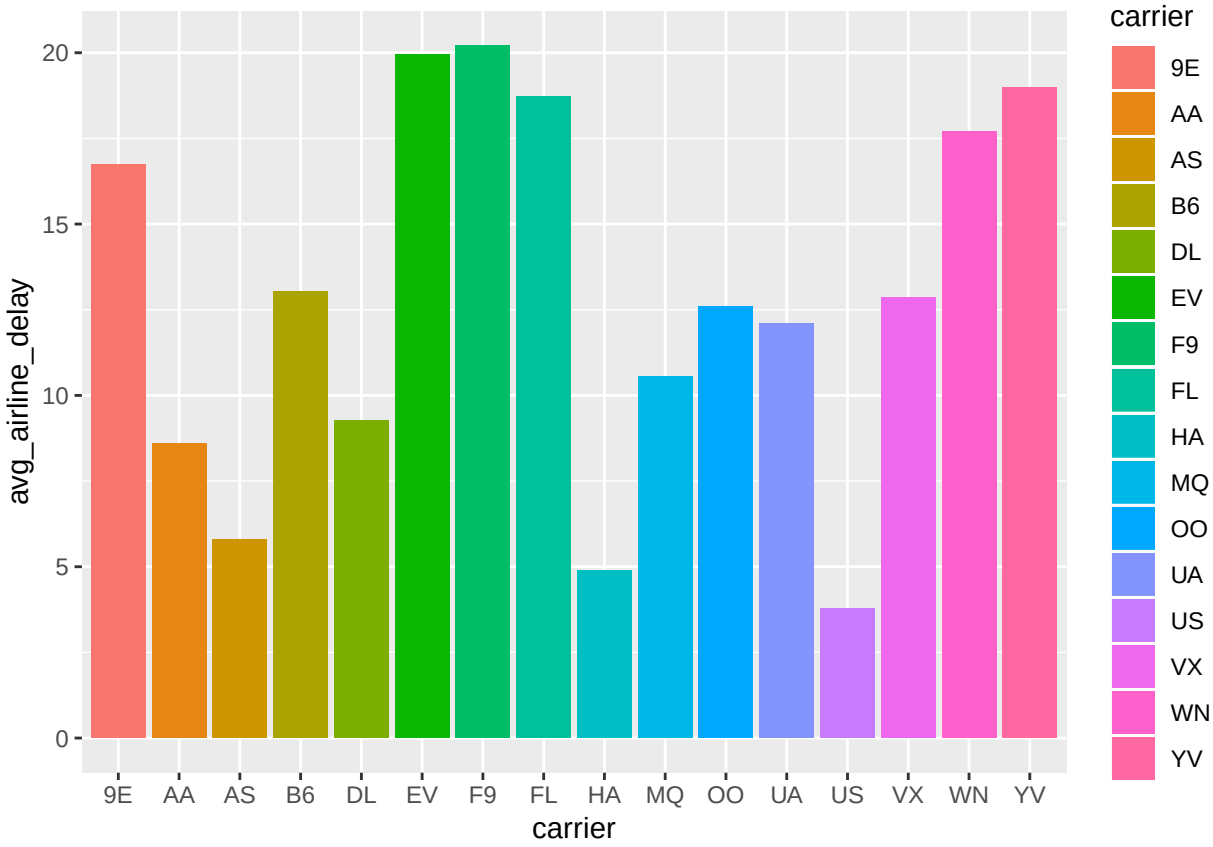
1. Which airlines have the worst possible delays? This question is important because it tells which airlines allow you to be a bit late yet make the flight or conversely tells you flights that have a tendency to always be delayed.
2. Is there a correlation between the distance and delays. This question is important as it gives information whether flight delay is a affected drastically by the air time and if adjustments can be made to reduce the delays for longer flights.

(c) Exploring Data

For each of the questions you proposed in Problem 1b, perform an exploratory data analysis designed to address the question. At a minimum, you should produce two visualizations (graphics or tables) related to each question. Be sure to describe what the visuals show and how they speak to your question of interest.

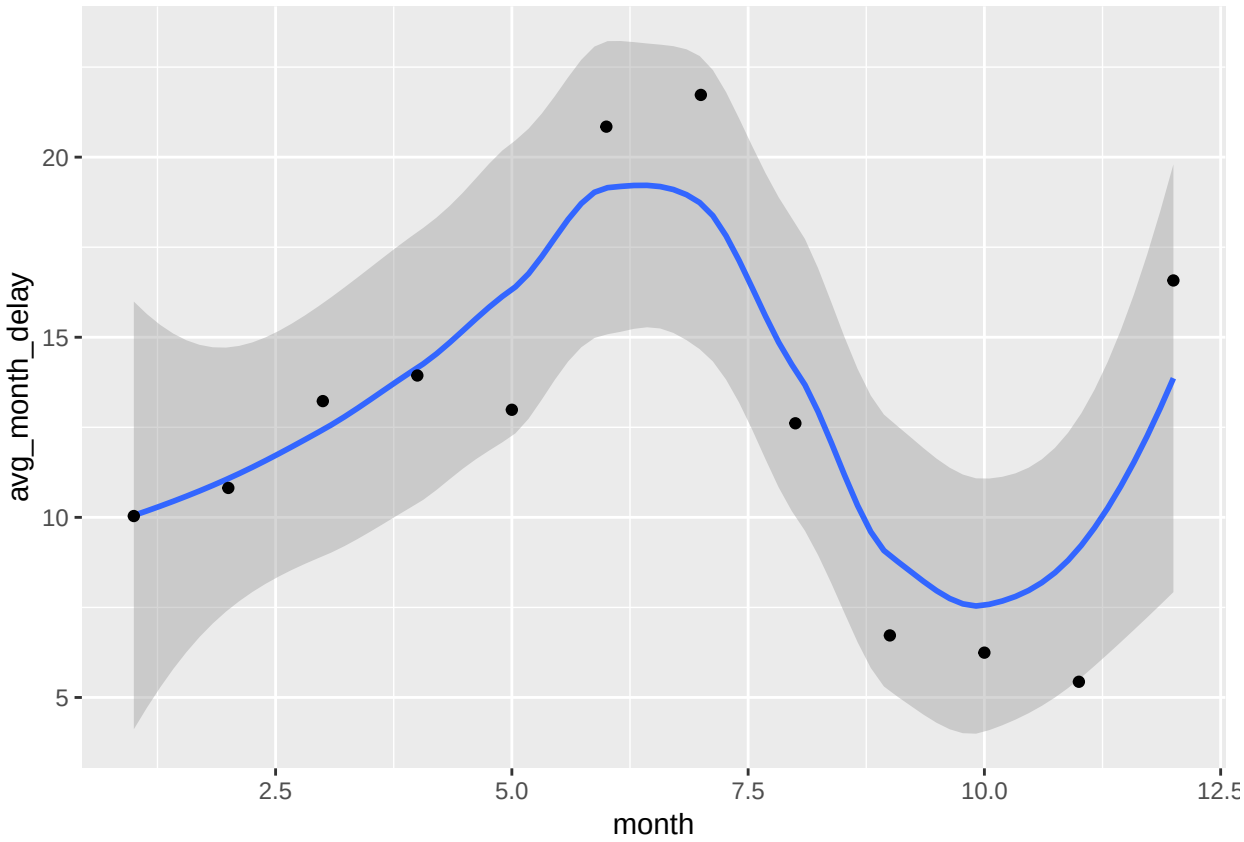
Q1:

```
q1 <- flights %>% group_by(carrier) %>% summarise(avg_airline_delay = mean(dep_delay, na.rm = TRUE))

ggplot(q1,mapping = aes(x = carrier, y = avg_airline_delay))+geom_bar(stat = "identity", aes(fill = car
```



```
test <- flights %>% group_by(month) %>% summarise(
  avg_month_delay = mean(dep_delay, na.rm = TRUE)
)

ggplot(test, mapping = aes(x = month, y = avg_month_delay))+ geom_smooth() + geom_point()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



The first visualiation portrays all carriers and the average delay associated with them. The bars have been color coded by carrier to help distinguish between airlines. We can observe EV, F9 and FL have the highest number of delays.

The second visualization shows the trend that average delays follow throughout the year. We can observe an increase in delays from January until June/July, when it peaks. This can be used to examine and associate the delays with factors like weather and the how busy the airport is during such times.

Q2

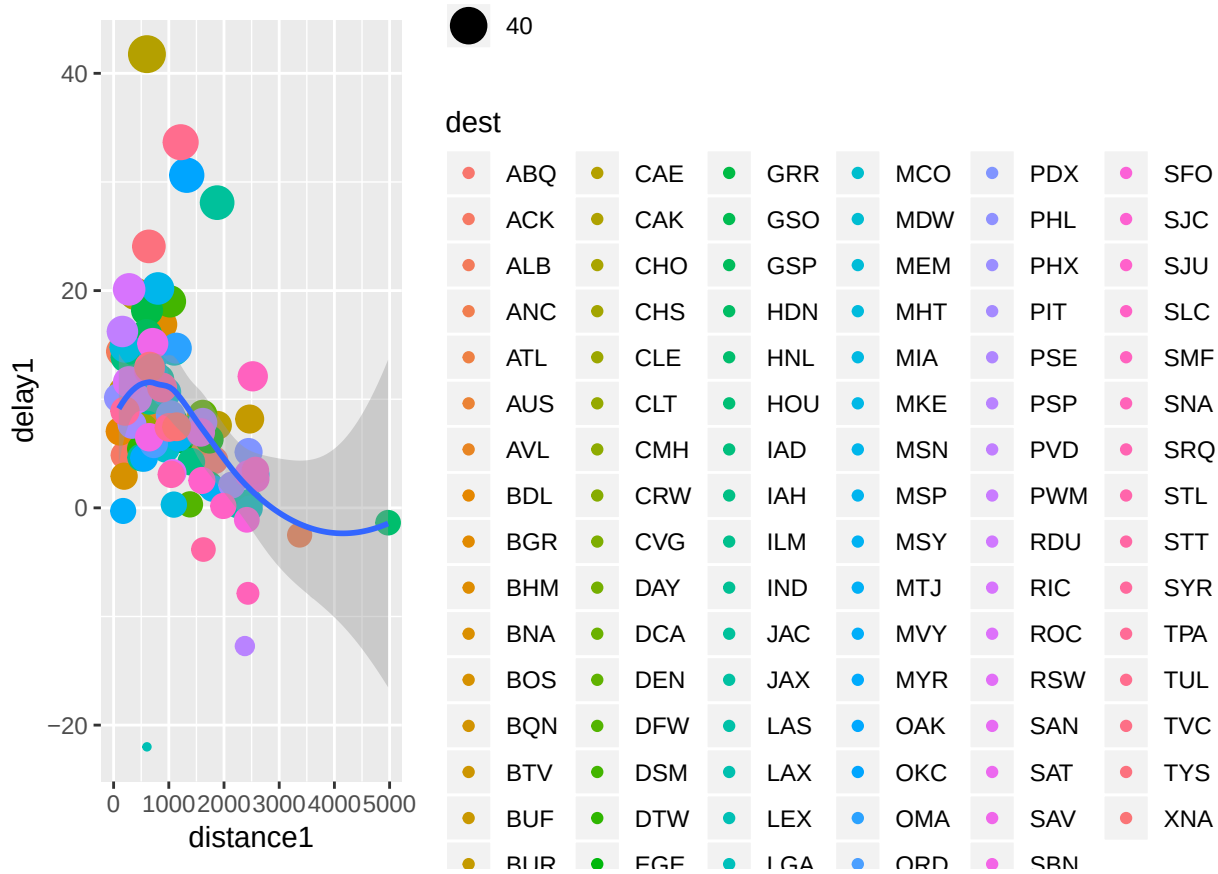
```
delays <- flights %>%   group_by(dest) %>%   summarise(count = n(),distance1 =mean(distance, na.rm = TRU

delays
```

```
## # A tibble: 105 x 4
##    dest  count distance1 delay1
##    <chr> <int>     <dbl>  <dbl>
##  1 ABQ     254      1826    4.38
##  2 ACK     265       199    4.85
##  3 ALB     439       143   14.4
```

```
##  4 ANC       8     3370   -2.5
##  5 ATL   17215      757.  11.3
##  6 AUS    2439     1514.   6.02
##  7 AVL     275      584.   8.00
##  8 BDL     443      116    7.05
##  9 BGR     375      378    8.03
## 10 BHM     297      866.  16.9
## # ... with 95 more rows
```

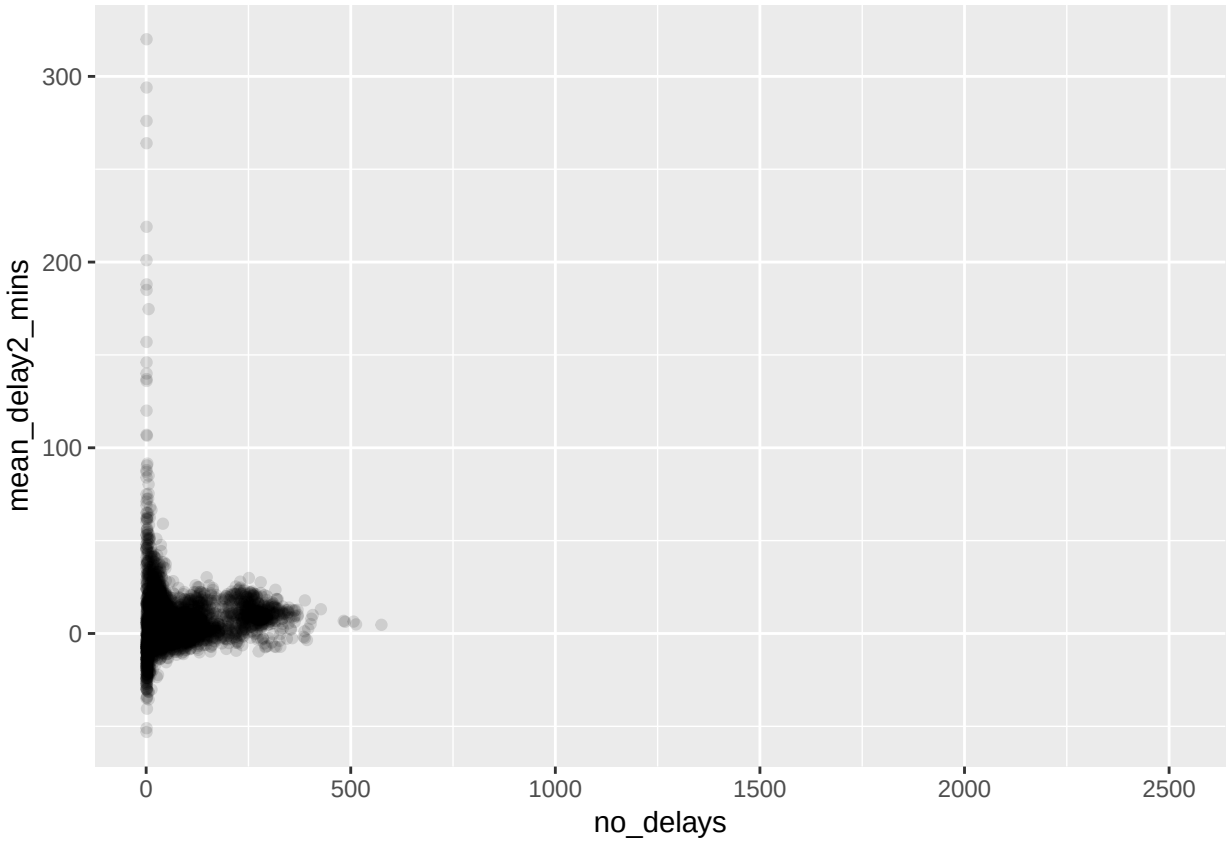
```
ggplot(delays, mapping = aes(x = distance1, y = delay1)) + geom_point(mapping = aes(size = delay1, colo
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## Warning: Removed 1 rows containing non-finite values (stat_smooth).
## Warning: Removed 1 rows containing missing values (geom_point).
```




```
delays2 <- flights %>%  group_by(tailnum) %>% summarise(mean_delay2_mins = mean(arr_delay, na.rm = TRUE)

ggplot(data = delays2, mapping = aes(x = no_delays,y = mean_delay2_mins)) + geom_point(alpha = 1/8)
```

```
## Warning: Removed 7 rows containing missing values (geom_point).
```



The third visualization plots the ditance that a flight travels against the average delay and the scatter points are color coded by the destiation. Shorter flights generally have a longer delay period and we can summarise that long flights can make up the delay in departure by travelling at a slightly increased rate to reduce landing delays.

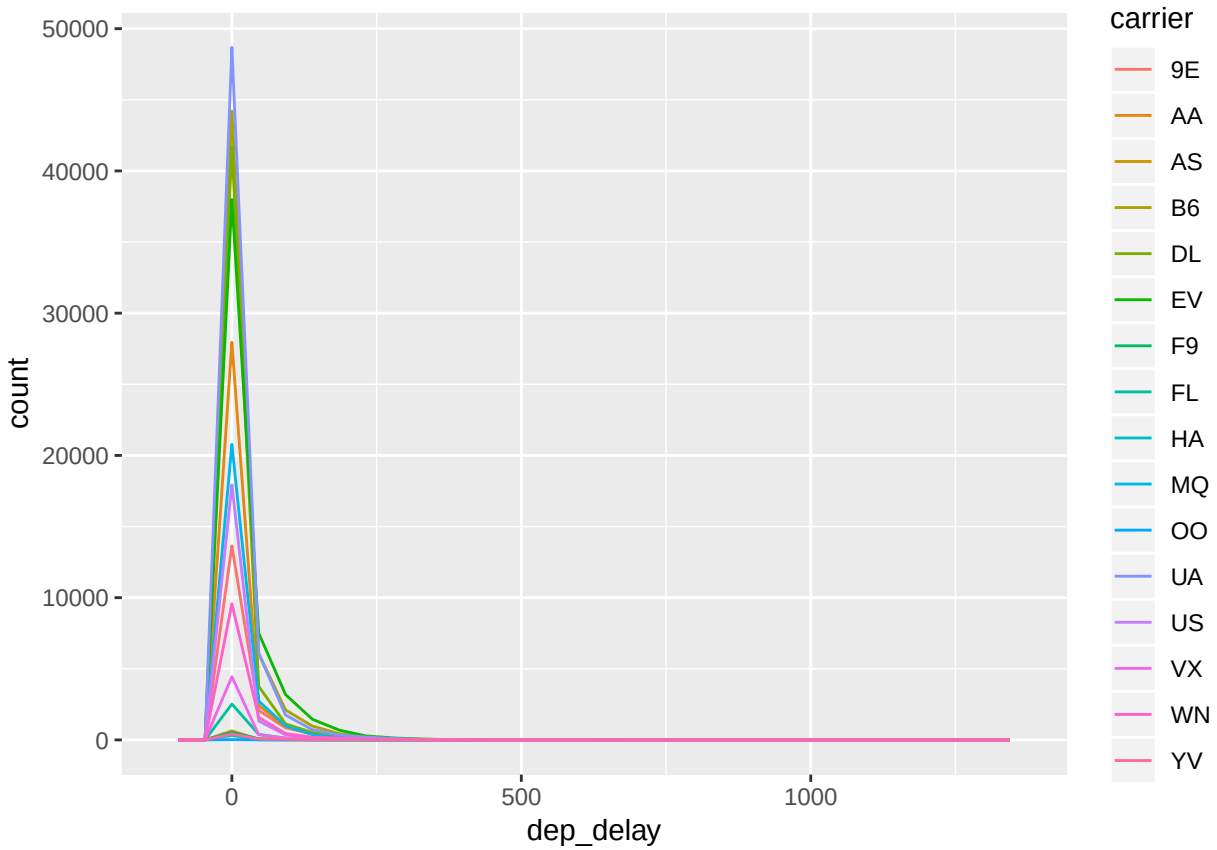
The fourth visualization plots the number of delays by each individual flight (identified uniquely by tailnumber attribute) and the average delay that the flight had in its entire operational lifespan as dictated in the data set. It shws the presence of certain outliers with extremely high flight delays. This result shows us that these outliers might skew the conclusions in a major way that we derive about flight delays. ##### (d) Challenge Your Results

After completing the exploratory analyses from Problem 1c, do you have any concerns about your findings? How well defined was your original question? Do you still believe this question can be answered using this dataset? Comment on any ethical and/or privacy concerns you have with your analysis.

```
flights %>% ggplot(aes(dep_delay, color = carrier)) + geom_freqpoly()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 8255 rows containing non-finite values (stat_bin).
```



The above visualization depicts the presence of outliers that lead to a skewed pereception as opposed to what the data is actaully about.

I am skeptical about my findings as there are some flights that were cancelled (arr_delay = NA, dep_delay = NA) and these cancelled flights actually increase the number of records and lead me to believe that mean might not be the best statistic to use for my analysis and that variance and standard deviation must be taken into account when creating conclusions.

I also feel that this data should be linked with a weather data to give us true insight into the cause of the delays. I have visualized how longer flights generally have consistent delay patterns as they can make up for lost time during transit. But the reason why shorter flights have a longer delay is still not clear to me.