PM 566: Homework 02

AUTHOR
Tarun Mahesh

Homework 02: Analyzing the nycflights13 package

Step 1: Installing and loading the package

```
# Installing nycflights13 (commented out to prevent error during render)
# install.packages("nycflights13")

# Assigning a variable to each required data frame for ease of access
airlines <- nycflights13::airlines
airports <- nycflights13::airports
flights <- nycflights13::flights
planes <- nycflights13::planes
weather <- nycflights13::weather

# Before we begin answering the primary question, let us explore the 5 data frames, their dimensi

summary(airlines)
```

```
  carrier             name          
Length:16          Length:16         
Class :character   Class :character  
Mode  :character   Mode  :character  
```

```
summary(airports)
```

```
    faa                name               lat             lon          
Length:1458        Length:1458        Min.   :19.72   Min.   :-176.65  
Class :character   Class :character   1st Qu.:34.26   1st Qu.:-119.19  
Mode  :character   Mode  :character   Median :40.09   Median : -94.66  
                                      Mean   :41.65   Mean   :-103.39  
```

```
                                  3rd Qu.:45.07   3rd Qu.: -82.52
                                  Max.   :72.27   Max.   : 174.11
     alt               tz             dst              tzone
Min.   : -54.00   Min.   :-10.000   Length:1458        Length:1458
1st Qu.:  70.25   1st Qu.: -8.000   Class :character   Class :character
Median : 473.00   Median : -6.000   Mode  :character   Mode  :character
Mean   :1001.42   Mean   : -6.519
3rd Qu.:1062.50   3rd Qu.: -5.000
Max.   :9078.00   Max.   :  8.000
```

```
summary(flights)
```

```
     year          month             day           dep_time    sched_dep_time
Min.   :2013   Min.   : 1.000   Min.   : 1.00   Min.   :   1   Min.   : 106
1st Qu.:2013   1st Qu.: 4.000   1st Qu.: 8.00   1st Qu.: 907   1st Qu.: 906
Median :2013   Median : 7.000   Median :16.00   Median :1401   Median :1359
Mean   :2013   Mean   : 6.549   Mean   :15.71   Mean   :1349   Mean   :1344
3rd Qu.:2013   3rd Qu.:10.000   3rd Qu.:23.00   3rd Qu.:1744   3rd Qu.:1729
Max.   :2013   Max.   :12.000   Max.   :31.00   Max.   :2400   Max.   :2359
                                                NA's   :8255
  dep_delay          arr_time    sched_arr_time   arr_delay
Min.   : -43.00   Min.   :   1   Min.   :   1   Min.   : -86.000
1st Qu.:  -5.00   1st Qu.:1104   1st Qu.:1124   1st Qu.: -17.000
Median :  -2.00   Median :1535   Median :1556   Median :  -5.000
Mean   :  12.64   Mean   :1502   Mean   :1536   Mean   :   6.895
3rd Qu.:  11.00   3rd Qu.:1940   3rd Qu.:1945   3rd Qu.:  14.000
Max.   :1301.00   Max.   :2400   Max.   :2359   Max.   :1272.000
NA's   :8255      NA's   :8713                  NA's   :9430
  carrier             flight       tailnum            origin
Length:336776      Min.   :   1   Length:336776      Length:336776
Class :character   1st Qu.: 553   Class :character   Class :character
Mode  :character   Median :1496   Mode  :character   Mode  :character
                   Mean   :1972
                   3rd Qu.:3465
                   Max.   :8500

    dest              air_time        distance         hour
Length:336776      Min.   : 20.0   Min.   :  17   Min.   : 1.00
```

```
Class :character   1st Qu.: 82.0   1st Qu.: 502   1st Qu.: 9.00
Mode  :character   Median :129.0   Median : 872   Median :13.00
                   Mean   :150.7   Mean   :1040   Mean   :13.18
                   3rd Qu.:192.0   3rd Qu.:1389   3rd Qu.:17.00
                   Max.   :695.0   Max.   :4983   Max.   :23.00
                   NA's   :9430
    minute        time_hour
Min.   : 0.00   Min.   :2013-01-01 05:00:00.00
1st Qu.: 8.00   1st Qu.:2013-04-04 13:00:00.00
Median :29.00   Median :2013-07-03 10:00:00.00
Mean   :26.23   Mean   :2013-07-03 05:22:54.64
3rd Qu.:44.00   3rd Qu.:2013-10-01 07:00:00.00
Max.   :59.00   Max.   :2013-12-31 23:00:00.00
```

```
summary(planes)
```

```
  tailnum              year         type           manufacturer
Length:3322        Min.   :1956   Length:3322        Length:3322
Class :character   1st Qu.:1997   Class :character   Class :character
Mode  :character   Median :2001   Mode  :character   Mode  :character
                   Mean   :2000
                   3rd Qu.:2005
                   Max.   :2013
                   NA's   :70
   model              engines          seats           speed
Length:3322        Min.   :1.000   Min.   :  2.0   Min.   : 90.0
Class :character   1st Qu.:2.000   1st Qu.:140.0   1st Qu.:107.5
Mode  :character   Median :2.000   Median :149.0   Median :162.0
                   Mean   :1.995   Mean   :154.3   Mean   :236.8
                   3rd Qu.:2.000   3rd Qu.:182.0   3rd Qu.:432.0
                   Max.   :4.000   Max.   :450.0   Max.   :432.0
                                                   NA's   :3299
   engine
Length:3322
Class :character
Mode  :character
```

```
summary(weather)
```

```
   origin               year          month             day          
Length:26115       Min.   :2013   Min.   : 1.000   Min.   : 1.00  
Class :character   1st Qu.:2013   1st Qu.: 4.000   1st Qu.: 8.00  
Mode  :character   Median :2013   Median : 7.000   Median :16.00  
                   Mean   :2013   Mean   : 6.504   Mean   :15.68  
                   3rd Qu.:2013   3rd Qu.: 9.000   3rd Qu.:23.00  
                   Max.   :2013   Max.   :12.000   Max.   :31.00  

      hour            temp            dewp           humid       
 Min.   : 0.00   Min.   : 10.94   Min.   :-9.94   Min.   : 12.74  
 1st Qu.: 6.00   1st Qu.: 39.92   1st Qu.:26.06   1st Qu.: 47.05  
 Median :11.00   Median : 55.40   Median :42.08   Median : 61.79  
 Mean   :11.49   Mean   : 55.26   Mean   :41.44   Mean   : 62.53  
 3rd Qu.:17.00   3rd Qu.: 69.98   3rd Qu.:57.92   3rd Qu.: 78.79  
 Max.   :23.00   Max.   :100.04   Max.   :78.08   Max.   :100.00  
                 NA's   :1        NA's   :1       NA's   :1       
    wind_dir       wind_speed       wind_gust         precip       
 Min.   :  0.0   Min.   :   0.000   Min.   :16.11   Min.   :0.000000  
 1st Qu.:120.0   1st Qu.:   6.905   1st Qu.:20.71   1st Qu.:0.000000  
 Median :220.0   Median :  10.357   Median :24.17   Median :0.000000  
 Mean   :199.8   Mean   :  10.518   Mean   :25.49   Mean   :0.004469  
 3rd Qu.:290.0   3rd Qu.:  13.809   3rd Qu.:28.77   3rd Qu.:0.000000  
 Max.   :360.0   Max.   :1048.361   Max.   :66.75   Max.   :1.210000  
 NA's   :460     NA's   :4          NA's   :20778                     
    pressure          visib          time_hour                  
 Min.   : 983.8   Min.   : 0.000   Min.   :2013-01-01 01:00:00.0  
 1st Qu.:1012.9   1st Qu.:10.000   1st Qu.:2013-04-01 21:30:00.0  
 Median :1017.6   Median :10.000   Median :2013-07-01 14:00:00.0  
 Mean   :1017.9   Mean   : 9.255   Mean   :2013-07-01 18:26:37.7  
 3rd Qu.:1023.0   3rd Qu.:10.000   3rd Qu.:2013-09-30 13:00:00.0  
```

```
 Max.   :1042.1   Max.   :10.000   Max.   :2013-12-30 18:00:00.0
 NA's   :2729
```

These summary outputs show us

1. The number of variables
2. The class/type of each variable
3. The number of NA values in each variable

Question 1:

1. Find the top 10 most popular destinations in the `flights` data. How many flights went to each?

For this question, my logic is to count the number of flights to each destination by counting the number of times a destination appears in the flights data.

```
library(dplyr)
```

```
Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
top10dest <- flights %>%
  count(dest, sort = TRUE) %>%
  slice_head(n = 10)

top10dest
```

```
# A tibble: 10 × 2
   dest      n
```

```
   <chr> <int>
 1 ORD   17283
 2 ATL   17215
 3 LAX   16174
 4 BOS   15508
 5 MCO   14082
 6 CLT   14064
 7 SFO   13331
 8 FLL   12055
 9 MIA   11728
10 DCA    9705
```

If we want to find the top 10 airports corresponding to the top 10 destinations, we can perform a left join, between "airports" and "flights", using *faa* from aiports and *dest* from flights as the key(s).

```
top10dest <- flights %>%
  count(dest, sort = TRUE) %>%
  slice_head(n = 10) %>%
  left_join(airports %>% select(faa, name),
            by = c("dest" = "faa"))

top10dest
```

```
# A tibble: 10 × 3
   dest      n name
   <chr> <int> <chr>
 1 ORD   17283 Chicago Ohare Intl
 2 ATL   17215 Hartsfield Jackson Atlanta Intl
 3 LAX   16174 Los Angeles Intl
 4 BOS   15508 General Edward Lawrence Logan Intl
 5 MCO   14082 Orlando Intl
 6 CLT   14064 Charlotte Douglas Intl
 7 SFO   13331 San Francisco Intl
 8 FLL   12055 Fort Lauderdale Hollywood Intl
 9 MIA   11728 Miami Intl
10 DCA    9705 Ronald Reagan Washington Natl
```

Now we can see the top 10 destinations, the number of flights doing to each said destination, and the name of the airport.

Question 2:

Create two new categorical variables in `flights` based on the departure time (`dep_time`), and arrival time (`arr_time`). These new variables should have four categories: “early morning” for 12am to 6am, “morning” for 6am to 12pm, “afternoon” for 12pm to 6pm, and “evening” for 6pm to 12am. Show barplots of both of these new variables. What percentage of flights were “red eye” flights? We’ll define these as flights that depart in “afternoon” or “evening” and arrive in “early morning” or “morning.”

Before we start, let us try and see what kind of data exists in the columns dep_time and arr_time.

```
summary(flights$dep_time)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
      1     907    1401    1349    1744    2400    8255 
```

```
summary(flights$arr_time)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
      1    1104    1535    1502    1940    2400    8713 
```

Question 2, Part 1:

It seems that the min and max values are integers between 1 and 2400, which can correspond to time in hh:mm. For example, the integer value 2413 is actually 24:13 AM. So, to define the “hour”, we need to divide the dep_time/arr_time variable by 100, and use the numbers before the decimal point. Instead of doing a regular division, we can use integer division (%/%), which will retain only the value before the decimal point, as in, the hour value. Since our time divisions start from midnight, we can map “24” to “0”, using ‘%% 24’ (The “modulo” operation). Using right = FALSE in the cut operation allows to include all times until 5:59 AM in “early morning”, and 6:00 AM flights get classified as “morning”.

```r
flights_hours <- nycflights13::flights %>%
  mutate(
    dep_hr = (dep_time %/% 100) %% 24,
    arr_hr = (arr_time %/% 100) %% 24,
    dep_period = cut(dep_hr, c(0, 6, 12, 18, 24), right = FALSE,
                     labels = c("early morning","morning","afternoon","evening")),
    arr_period = cut(arr_hr, c(0, 6, 12, 18, 24), right = FALSE,
                     labels = c("early morning","morning","afternoon","evening"))
  )

summary(flights_hours$dep_period)
```

```
early morning       morning     afternoon       evening          NA's
         8759        122293        120761         76708          8255
```

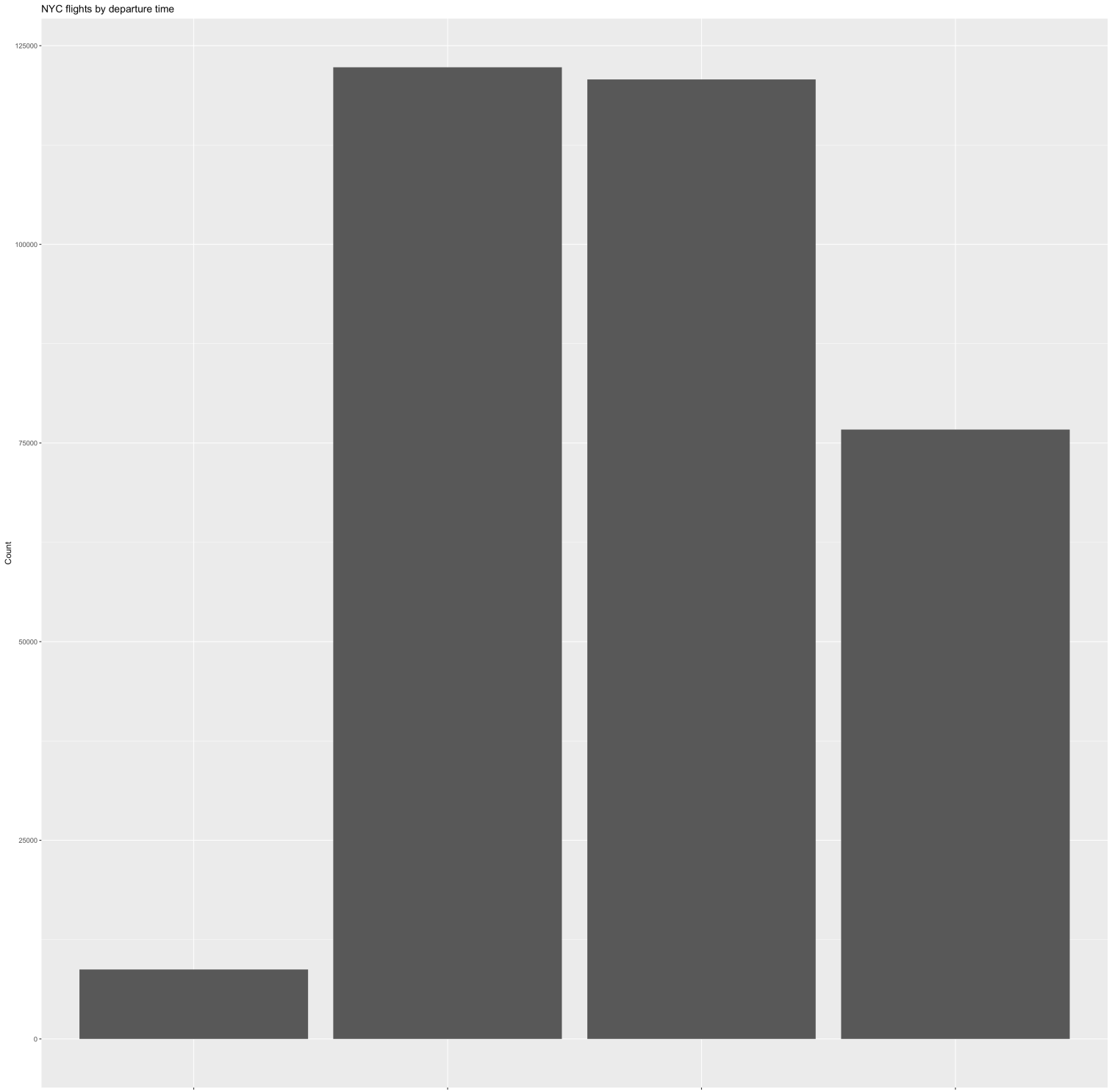
```r
summary(flights_hours$arr_period)
```

```
early morning       morning     afternoon       evening          NA's
        11928         88506        108911        118718          8713
```

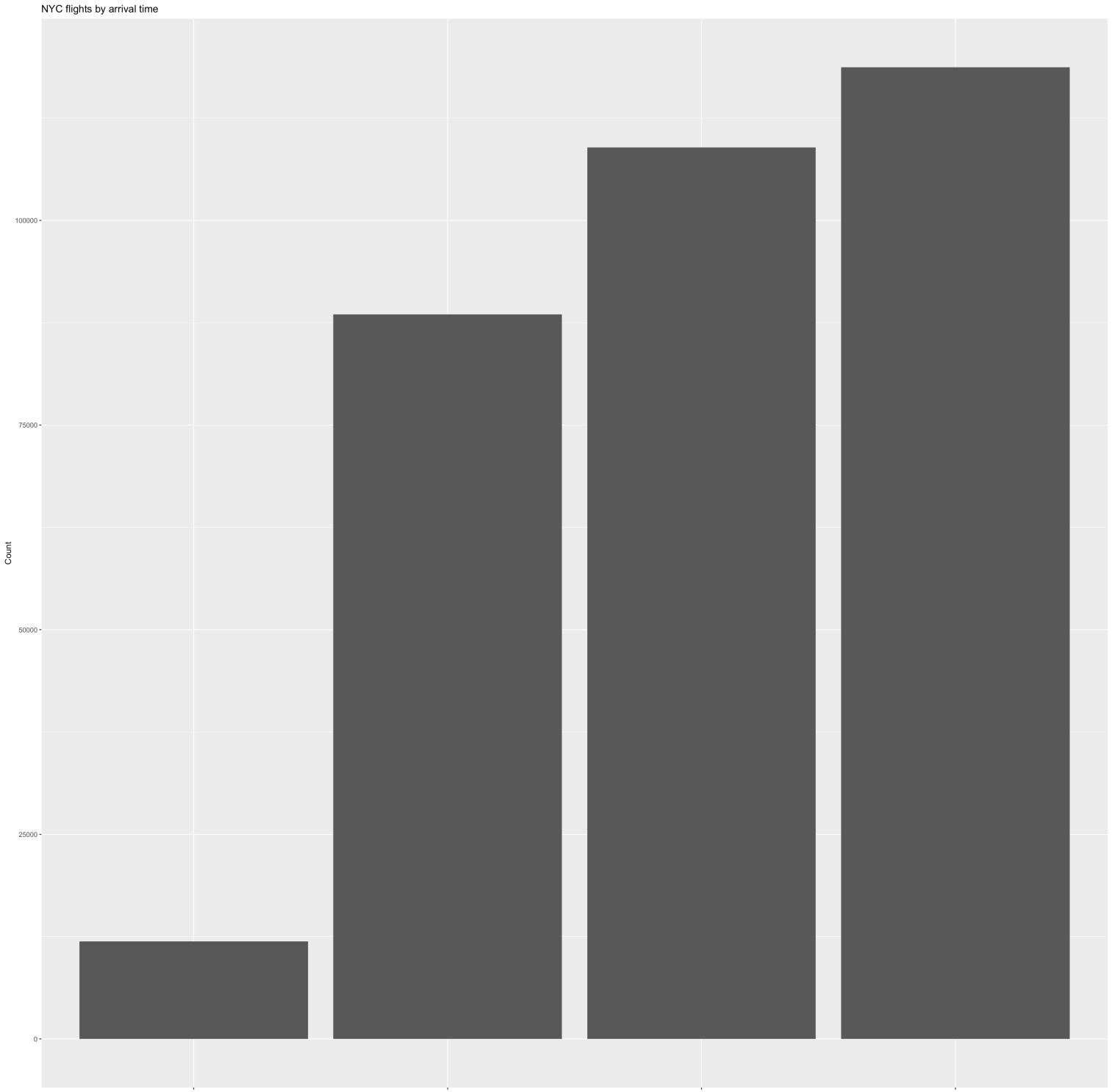
Question 2, Part 2:

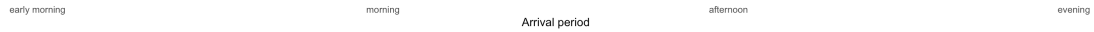
To create barplots for the new variables:

```r
library(ggplot2)

flights_hours %>% filter(!is.na(dep_period)) %>%
  ggplot(aes(dep_period)) + geom_bar(na.rm = TRUE) + labs(x = "Departure period", y = "Count", t
```


NYC flights by departure time
125000
100000
75000
50000
25000
0
Count


```
flights_hours %>% filter(!is.na(arr_period)) %>%
  ggplot(aes(arr_period)) + geom_bar(na.rm = TRUE) + labs(x = "Arrival period", y = "Count", tit
```


NYC flights by arrival time
Count
100000
75000
50000
25000
0



From these barplots, it is evident that most flights leave NYC in the morning (and closely followed by the afternoon), while most flights arrive in NYC in the evening.

Question 2, Part 3:

To find the percentage of flights that are “red-eye” (dep_period = afternoon/evening, arr_period = early morning/morning):

```
dep_noNA <- !is.na(flights_hours$dep_period)
arr_noNA <- !is.na(flights_hours$arr_period)
dep_arr_noNA <- dep_noNA & arr_noNA

dep_cleaned <- flights_hours$dep_period[dep_arr_noNA]
arr_cleaned <- flights_hours$arr_period[dep_arr_noNA]

dep_redeye <- dep_cleaned %in% c("afternoon", "evening")
arr_redeye <- arr_cleaned %in% c("early morning", "morning")
red_eye_flights <- dep_redeye & arr_redeye

redeye_percentage <- (sum(red_eye_flights)/length(red_eye_flights)) * 100
redeye_percentage
```

```
[1] 3.278029
```

3.28 % of flights arriving/departing to/from NYC are red-eye flights.

Question 3:

Were there any planes that flew for multiple airlines? If so, how many were there and which airlines did they fly for?

To find the planes that flew for multiple airlines, we require two datasets- planes and flights. The column “tailnum” maps planes to fllights, and we can then use the “carrier” column from flights to map tailnum to the names of different airlines.

```r
library(dplyr)

flights_airlines <- flights %>% filter(!is.na(tailnum)) %>%
  distinct(tailnum, carrier) %>%
  inner_join(planes %>% select(tailnum), by = "tailnum") %>%
  left_join(airlines, by = "carrier")

multiple_airlines <- flights_airlines %>%
  group_by(tailnum) %>%
  summarise(no_of_carriers = n(), airlines = paste(sort(unique(na.omit(name))), collapse = " and
  filter(no_of_carriers > 1) %>%
  arrange(desc(no_of_carriers), tailnum)

paste("The number of planes that flew for multiple airlines: ", nrow(multiple_airlines))
```

```
[1] "The number of planes that flew for multiple airlines:  17"
```

```r
multiple_airlines
```

```
# A tibble: 17 × 3
   tailnum no_of_carriers airlines
   <chr>            <int> <chr>
 1 N146PQ               2 Endeavor Air Inc. and ExpressJet Airlines Inc.
 2 N153PQ               2 Endeavor Air Inc. and ExpressJet Airlines Inc.
 3 N176PQ               2 Endeavor Air Inc. and ExpressJet Airlines Inc.
 4 N181PQ               2 Endeavor Air Inc. and ExpressJet Airlines Inc.
 5 N197PQ               2 Endeavor Air Inc. and ExpressJet Airlines Inc.
 6 N200PQ               2 Endeavor Air Inc. and ExpressJet Airlines Inc.
 7 N228PQ               2 Endeavor Air Inc. and ExpressJet Airlines Inc.
 8 N232PQ               2 Endeavor Air Inc. and ExpressJet Airlines Inc.
 9 N933AT               2 AirTran Airways Corporation and Delta Air Lines Inc.
10 N935AT               2 AirTran Airways Corporation and Delta Air Lines Inc.
11 N977AT               2 AirTran Airways Corporation and Delta Air Lines Inc.
12 N978AT               2 AirTran Airways Corporation and Delta Air Lines Inc.
13 N979AT               2 AirTran Airways Corporation and Delta Air Lines Inc.
14 N981AT               2 AirTran Airways Corporation and Delta Air Lines Inc.
15 N989AT               2 AirTran Airways Corporation and Delta Air Lines Inc.
```

```
16 N990AT               2 AirTran Airways Corporation and Delta Air Lines Inc.
17 N994AT               2 AirTran Airways Corporation and Delta Air Lines Inc.
```

Question 4:

In the figure above, there is a missing relationship between `weather` and `airports`. What is the relationship and how should it appear in the diagram?

For this, lets take a look at both dataframes first

```
# Columns in weather
colnames(weather)
```

```
 [1] "origin"     "year"       "month"      "day"        "hour"
 [6] "temp"       "dewp"       "humid"      "wind_dir"   "wind_speed"
[11] "wind_gust"  "precip"     "pressure"   "visib"      "time_hour"
```

```
# Columns in airports
colnames(airports)
```

```
[1] "faa"   "name"  "lat"   "lon"   "alt"   "tz"    "dst"   "tzone"
```

It seems like origin in the weather dataframe could map to faa in the airports dataset.

```
table(weather$origin)
```

```
 EWR  JFK  LGA
8703 8706 8706
```

These are faa codes, and we can quickly reconfirm this

```
airport_faa <- table(airports$faa)
head(airport_faa)
```

```
04G 06A 06C 06N 09J 0A9
  1   1   1   1   1   1
```

So, the missing relationship is between “faa” in airports, and “origin” in weather. (They are connected indirectly via origin in the “flights” dataset)

Primary Question: Which weather phenomena have the most impact on flight delays?

Question 5:

Question 5 Part 1: In order to address our primary question, we want to prepare a tidy dataset (a single data frame). This will consist of information from the `flights` and `weather` data. The `year`, `month`, `day`, `hour`, and `origin` variables from `weather` provide *almost* enough information to give each observation a unique value. Create a new variable in the weather dataset by pasting together those 5 variables, then determine how many duplicated values there are and explain why.

To do this, we can paste together the values for each of these variables and specify a separator (like ‘-’). The “duplicated” function returns the number of duplicated values for a specific variable (here, the varibale we are creating) in the data frame.

```
library(dplyr)

weather_2 <- weather %>%
  mutate(pasted_var = paste(year, month, day, hour, origin, sep = "-"))

dupes <- sum(duplicated(weather_2$pasted_var))
dupes
```

```
[1] 3
```

```
weather_2 %>% count(pasted_var, sort = TRUE) %>%
  filter(n > 1) %>%
```

```
  slice_head(n = 5)
```

```
# A tibble: 3 × 2
  pasted_var          n
  <chr>           <int>
1 2013-11-3-1-EWR     2
2 2013-11-3-1-JFK     2
3 2013-11-3-1-LGA     2
```

3 observations are repeated, and this can be because we only consider the hourly data for each origin (and record other time values ranked higher than "hour" (day, month and year)).

Within one hour, some locations may have multiple observations (maybe a sudden change in the weather?), which may have been recorded at different minutes, but since we include only the hour, the values are recorded as duplicates.

Question 5 Part 2: Merge the `flights` data and the `weather` data so that each flight contains information about the weather at its **departure airport** at the time it was **scheduled** to take off. You'll want to use the variables `time_hour` and `origin` in order to do this.

This can be achieved using a simple left_join function.

```
flights_weather <- flights %>%
  left_join(weather, by = c("origin", "time_hour"))
```

Question 6:

On this merged dataset, perform steps 2-5 of the EDA checklist presented in class. Remember the context provided by our primary question above.

Steps 2-5 of the EDA checklist are:

- 2: Check the size of the data
- 3: Examine the variables and their types
- 4: Look at the top and bottom of the data

- 5: Visualize the distribution of the key variables

2. Size of the data

```
dim(flights_weather)
```

```
[1] 336776     32
```

3. Variables and their types

```
str(flights_weather)
```

```
tibble [336,776 × 32] (S3: tbl_df/tbl/data.frame)
 $ year.x        : int [1:336776] 2013 2013 2013 2013 2013 2013 2013 2013 2013 2013 ...
 $ month.x       : int [1:336776] 1 1 1 1 1 1 1 1 1 1 ...
 $ day.x         : int [1:336776] 1 1 1 1 1 1 1 1 1 1 ...
 $ dep_time      : int [1:336776] 517 533 542 544 554 554 555 557 557 558 ...
 $ sched_dep_time: int [1:336776] 515 529 540 545 600 558 600 600 600 600 ...
 $ dep_delay     : num [1:336776] 2 4 2 -1 -6 -4 -5 -3 -3 -2 ...
 $ arr_time      : int [1:336776] 830 850 923 1004 812 740 913 709 838 753 ...
 $ sched_arr_time: int [1:336776] 819 830 850 1022 837 728 854 723 846 745 ...
 $ arr_delay     : num [1:336776] 11 20 33 -18 -25 12 19 -14 -8 8 ...
 $ carrier       : chr [1:336776] "UA" "UA" "AA" "B6" ...
 $ flight        : int [1:336776] 1545 1714 1141 725 461 1696 507 5708 79 301 ...
 $ tailnum       : chr [1:336776] "N14228" "N24211" "N619AA" "N804JB" ...
 $ origin        : chr [1:336776] "EWR" "LGA" "JFK" "JFK" ...
 $ dest          : chr [1:336776] "IAH" "IAH" "MIA" "BQN" ...
 $ air_time      : num [1:336776] 227 227 160 183 116 150 158 53 140 138 ...
 $ distance      : num [1:336776] 1400 1416 1089 1576 762 ...
 $ hour.x        : num [1:336776] 5 5 5 5 6 5 6 6 6 6 ...
 $ minute        : num [1:336776] 15 29 40 45 0 58 0 0 0 0 ...
 $ time_hour     : POSIXct[1:336776], format: "2013-01-01 05:00:00" "2013-01-01 05:00:00" ...
 $ year.y        : int [1:336776] 2013 2013 2013 2013 2013 2013 2013 2013 2013 2013 ...
 $ month.y       : int [1:336776] 1 1 1 1 1 1 1 1 1 1 ...
 $ day.y         : int [1:336776] 1 1 1 1 1 1 1 1 1 1 ...
 $ hour.y        : int [1:336776] 5 5 5 5 6 5 6 6 6 6 ...
 $ temp          : num [1:336776] 39 39.9 39 39 39.9 ...
```

```
$ dewp          : num [1:336776] 28 25 27 27 25 ...
$ humid         : num [1:336776] 64.4 54.8 61.6 61.6 54.8 ...
$ wind_dir      : num [1:336776] 260 250 260 260 260 260 240 260 260 260 ...
$ wind_speed    : num [1:336776] 12.7 15 15 15 16.1 ...
$ wind_gust     : num [1:336776] NA 21.9 NA NA 23 ...
$ precip        : num [1:336776] 0 0 0 0 0 0 0 0 0 0 ...
$ pressure      : num [1:336776] 1012 1011 1012 1012 1012 ...
$ visib         : num [1:336776] 10 10 10 10 10 10 10 10 10 10 ...
```

```
summary(flights_weather)
```

```
    year.x        month.x           day.x          dep_time    sched_dep_time
Min.   :2013   Min.   : 1.000   Min.   : 1.00   Min.   :   1   Min.   : 106
1st Qu.:2013   1st Qu.: 4.000   1st Qu.: 8.00   1st Qu.: 907   1st Qu.: 906
Median :2013   Median : 7.000   Median :16.00   Median :1401   Median :1359
Mean   :2013   Mean   : 6.549   Mean   :15.71   Mean   :1349   Mean   :1344
3rd Qu.:2013   3rd Qu.:10.000   3rd Qu.:23.00   3rd Qu.:1744   3rd Qu.:1729
Max.   :2013   Max.   :12.000   Max.   :31.00   Max.   :2400   Max.   :2359
                                                NA's   :8255
  dep_delay          arr_time    sched_arr_time   arr_delay
Min.   : -43.00   Min.   :   1   Min.   :   1   Min.   : -86.000
1st Qu.:  -5.00   1st Qu.:1104   1st Qu.:1124   1st Qu.: -17.000
Median :  -2.00   Median :1535   Median :1556   Median :  -5.000
Mean   :  12.64   Mean   :1502   Mean   :1536   Mean   :   6.895
3rd Qu.:  11.00   3rd Qu.:1940   3rd Qu.:1945   3rd Qu.:  14.000
Max.   :1301.00   Max.   :2400   Max.   :2359   Max.   :1272.000
NA's   :8255      NA's   :8713                  NA's   :9430
  carrier             flight       tailnum             origin
Length:336776      Min.   :   1   Length:336776      Length:336776
Class :character   1st Qu.: 553   Class :character   Class :character
Mode  :character   Median :1496   Mode  :character   Mode  :character
                   Mean   :1972
                   3rd Qu.:3465
                   Max.   :8500

    dest              air_time        distance        hour.x
Length:336776      Min.   : 20.0   Min.   :  17   Min.   : 1.00
Class :character   1st Qu.: 82.0   1st Qu.: 502   1st Qu.: 9.00
```

```
Mode  :character   Median :129.0   Median : 872   Median :13.00
                   Mean   :150.7   Mean   :1040   Mean   :13.18
                   3rd Qu.:192.0   3rd Qu.:1389   3rd Qu.:17.00
                   Max.   :695.0   Max.   :4983   Max.   :23.00
                   NA's   :9430
    minute        time_hour                       year.y
Min.   : 0.00   Min.   :2013-01-01 05:00:00.00   Min.   :2013
1st Qu.: 8.00   1st Qu.:2013-04-04 13:00:00.00   1st Qu.:2013
Median :29.00   Median :2013-07-03 10:00:00.00   Median :2013
Mean   :26.23   Mean   :2013-07-03 05:22:54.64   Mean   :2013
3rd Qu.:44.00   3rd Qu.:2013-10-01 07:00:00.00   3rd Qu.:2013
Max.   :59.00   Max.   :2013-12-31 23:00:00.00   Max.   :2013
                                                 NA's   :1556
   month.y           day.y           hour.y           temp
Min.   : 1.000   Min.   : 1.00   Min.   : 1.00   Min.   : 10.94
1st Qu.: 4.000   1st Qu.: 8.00   1st Qu.: 9.00   1st Qu.: 42.08
Median : 7.000   Median :16.00   Median :13.00   Median : 57.20
Mean   : 6.531   Mean   :15.67   Mean   :13.17   Mean   : 57.00
3rd Qu.: 9.000   3rd Qu.:23.00   3rd Qu.:17.00   3rd Qu.: 71.96
Max.   :12.000   Max.   :31.00   Max.   :23.00   Max.   :100.04
NA's   :1556     NA's   :1556    NA's   :1556    NA's   :1573
     dewp           humid           wind_dir       wind_speed
Min.   :-9.94   Min.   : 12.74   Min.   :  0.0   Min.   : 0.000
1st Qu.:26.06   1st Qu.: 43.99   1st Qu.:130.0   1st Qu.: 6.905
Median :42.80   Median : 57.73   Median :220.0   Median :10.357
Mean   :41.63   Mean   : 59.56   Mean   :201.5   Mean   :11.114
3rd Qu.:57.92   3rd Qu.: 75.33   3rd Qu.:290.0   3rd Qu.:14.960
Max.   :78.08   Max.   :100.00   Max.   :360.0   Max.   :42.579
NA's   :1573    NA's   :1573     NA's   :9796    NA's   :1634
  wind_gust          precip          pressure          visib
Min.   :16.11    Min.   :0.0000   Min.   : 983.8   Min.   : 0.000
1st Qu.:20.71    1st Qu.:0.0000   1st Qu.:1012.7   1st Qu.:10.000
Median :24.17    Median :0.0000   Median :1017.5   Median :10.000
Mean   :25.25    Mean   :0.0046   Mean   :1017.8   Mean   : 9.256
3rd Qu.:28.77    3rd Qu.:0.0000   3rd Qu.:1022.8   3rd Qu.:10.000
Max.   :66.75    Max.   :1.2100   Max.   :1042.1   Max.   :10.000
NA's   :256391   NA's   :1556     NA's   :38788    NA's   :1556
```

4. The top of the data

```
head(flights_weather)
```

```
# A tibble: 6 × 32
  year.x month.x day.x 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 24 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
#   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
#   hour.x <dbl>, minute <dbl>, time_hour <dttm>, year.y <int>, month.y <int>,
#   day.y <int>, hour.y <int>, temp <dbl>, dewp <dbl>, humid <dbl>,
#   wind_dir <dbl>, wind_speed <dbl>, wind_gust <dbl>, precip <dbl>,
#   pressure <dbl>, visib <dbl>
```

4. The bottom of the data

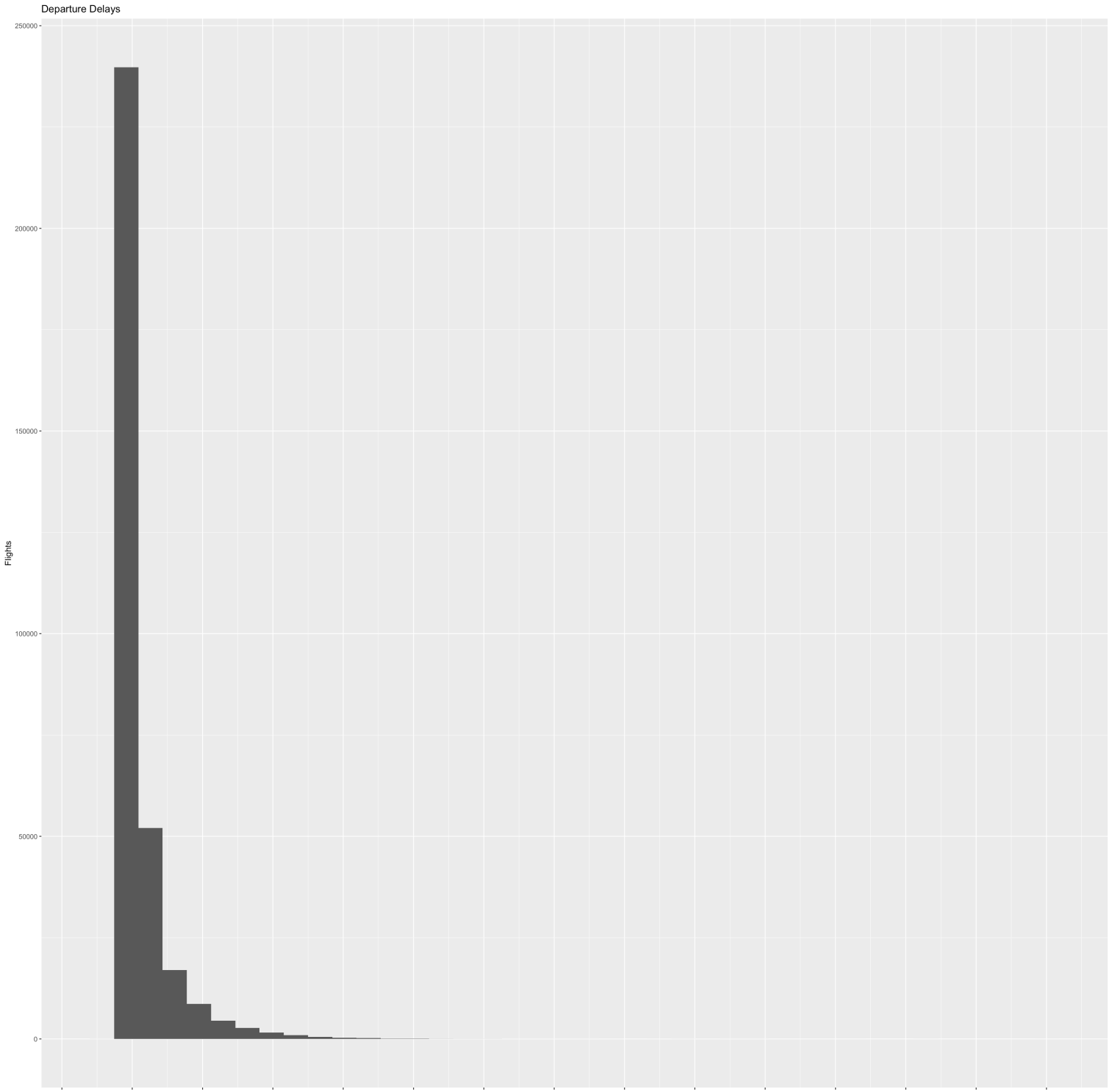
```
tail(flights_weather)
```

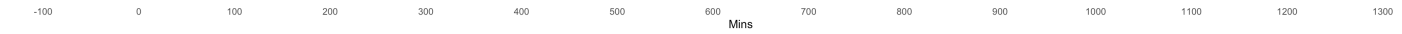
```
# A tibble: 6 × 32
  year.x month.x day.x dep_time sched_dep_time dep_delay arr_time sched_arr_time
   <int>   <int> <int>    <int>          <int>     <dbl>    <int>          <int>
1   2013       9    30       NA           1842        NA       NA           2019
2   2013       9    30       NA           1455        NA       NA           1634
3   2013       9    30       NA           2200        NA       NA           2312
4   2013       9    30       NA           1210        NA       NA           1330
5   2013       9    30       NA           1159        NA       NA           1344
6   2013       9    30       NA            840        NA       NA           1020
# ℹ 24 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
#   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
#   hour.x <dbl>, minute <dbl>, time_hour <dttm>, year.y <int>, month.y <int>,
#   day.y <int>, hour.y <int>, temp <dbl>, dewp <dbl>, humid <dbl>,
#   wind_dir <dbl>, wind_speed <dbl>, wind_gust <dbl>, precip <dbl>,
#   pressure <dbl>, visib <dbl>
```

5. Visualizing the distribution of the key variable(s)

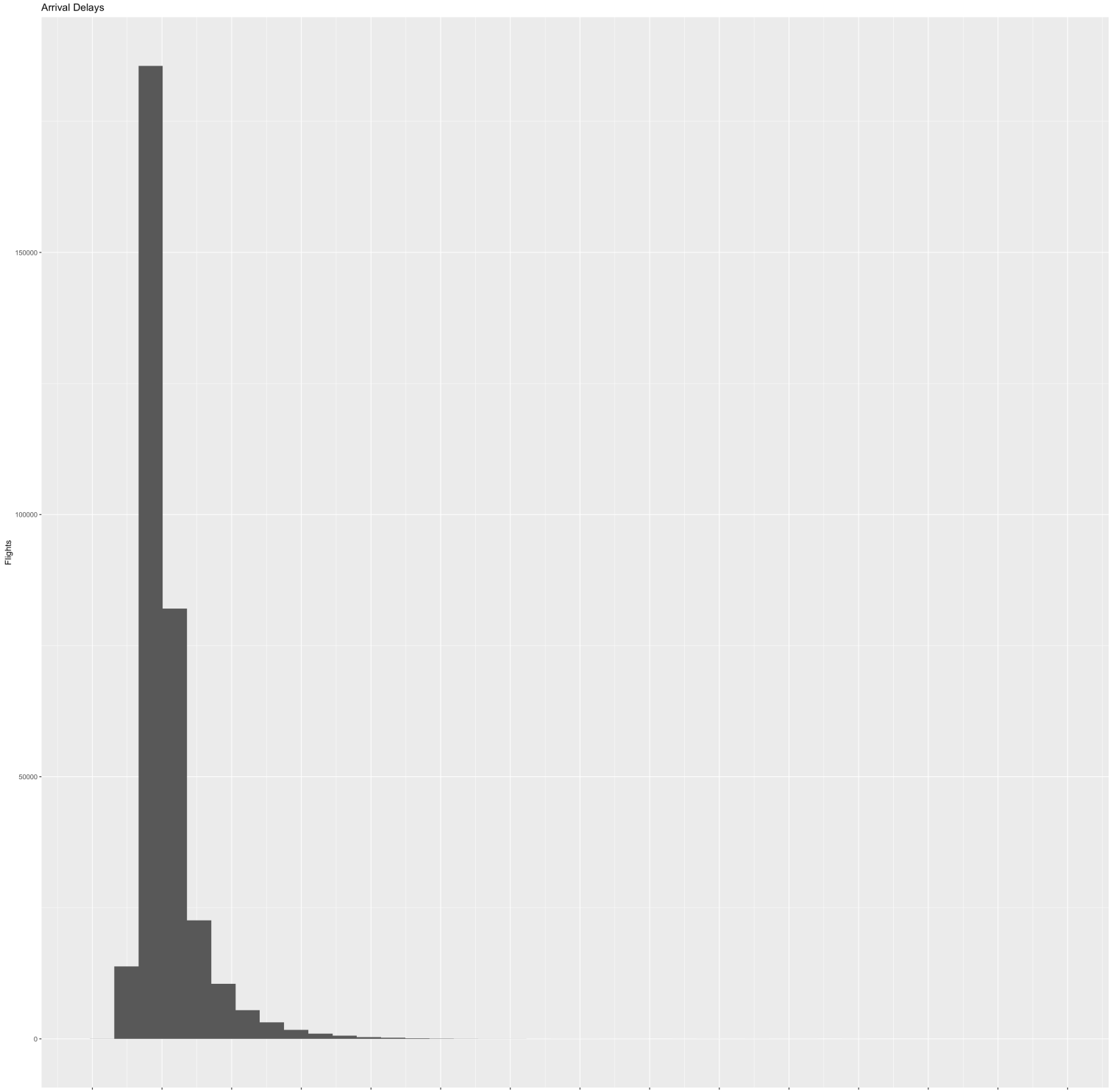
Considering the primary question, which is regarding the impact of different weather phenomena on flight delays, I think the key variables to look at are dep_delay and arr_delay. Afterwards, when we try to answer which weather phenomena affects delays most significantly, we can incorporate some of the weather variables like precipitation and wind speed.

```
library(ggplot2)

ggplot(flights_weather, aes(dep_delay)) +
  geom_histogram(bins = 40, na.rm = TRUE) +
  scale_x_continuous(breaks = seq(-100, 2000, by = 100)) +
  labs(title = "Departure Delays", x = "Mins", y = "Flights")
```


Departure Delays
250000
200000
150000
100000
50000
0
Flights



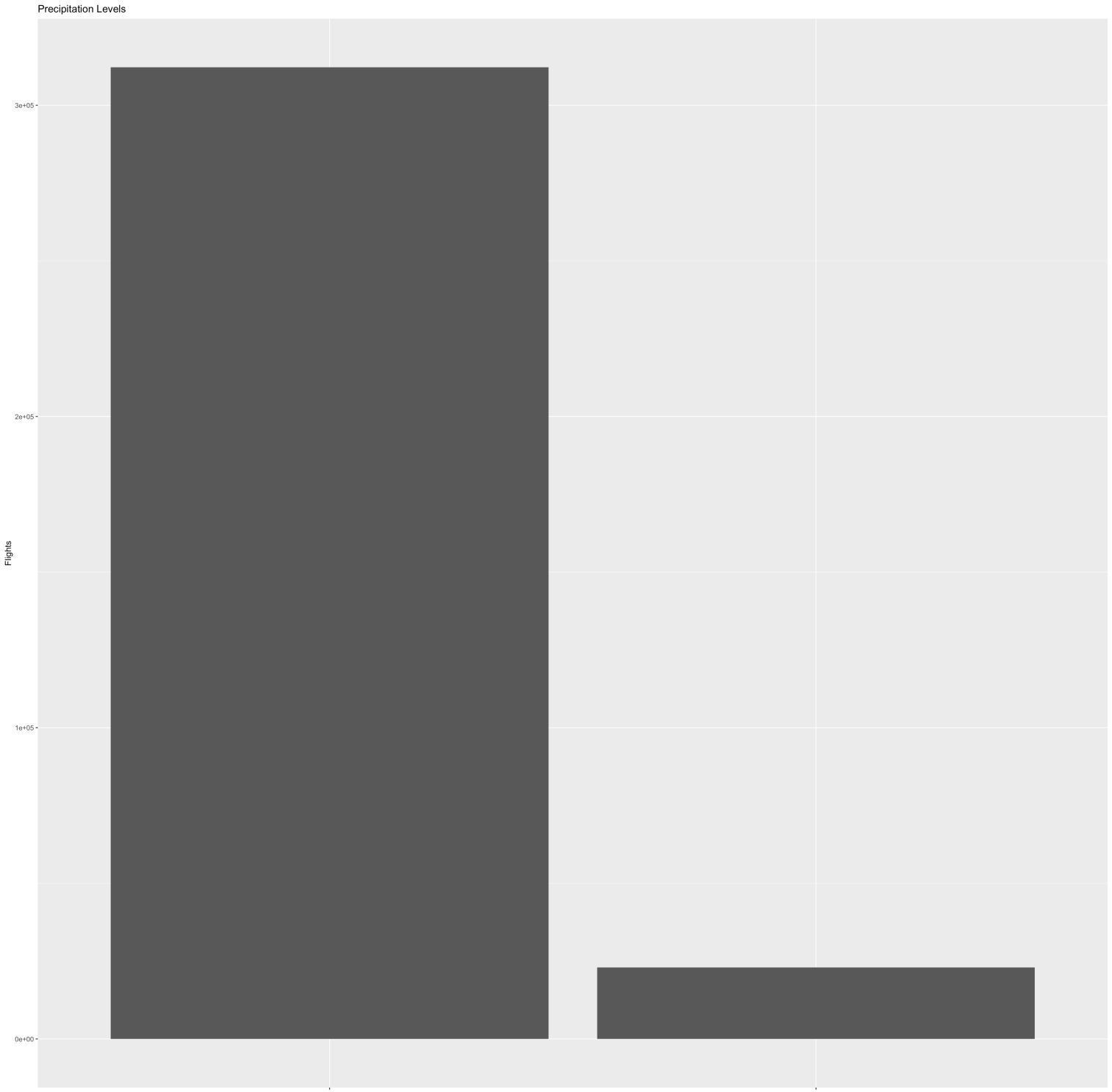

```
ggplot(flights_weather, aes(arr_delay)) +
  geom_histogram(bins = 40, na.rm = TRUE) +
  scale_x_continuous(breaks = seq(-100, 2000, by = 100)) +
  labs(title = "Arrival Delays", x = "Mins", y = "Flights")
```

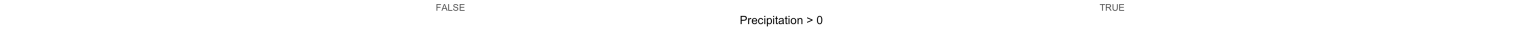

Arrival Delays
Flights
150000
100000
50000
0

It seems like most flights depart on time, or are delayed by 50 minutes. Most flights arrive earlier than expected.

Let's also create some simple visualizations of the weather variables to get an idea about how that data correlates to departure and arrival delays.

```
library(ggplot2)

ggplot(flights_weather %>% filter(!is.na(precip)), aes(precip > 0)) +
  geom_bar() +
  labs(title = "Precipitation Levels", x = "Precipitation > 0", y = "Flights")
```


Precipitation Levels
Flights
3e+05
2e+05
1e+05
0e+00



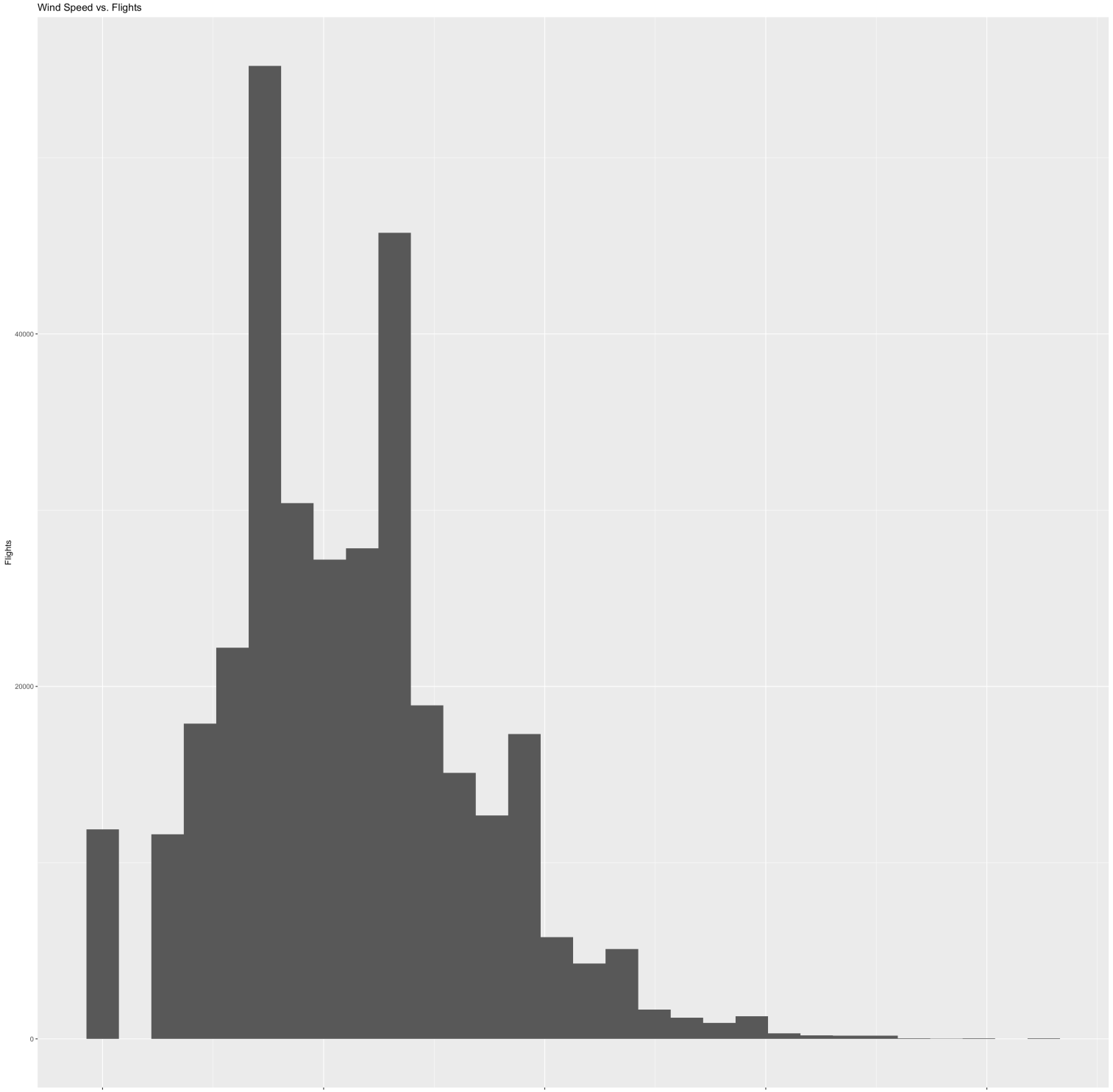

```
ggplot(flights_weather %>% filter(!is.na(precip)), aes(precip)) +
  geom_histogram(bins = 30, na.rm = TRUE) +
  labs(title = "Precipitation", x = "Inches", y = "Flights")
```


Precipitation
Flights
3e+05
2e+05
1e+05
0e+00

Precipitation was 0 for most flights, which leads me to believe that precipitation is not a significant driver of delays/arrivals.

Now looking at wind speed:

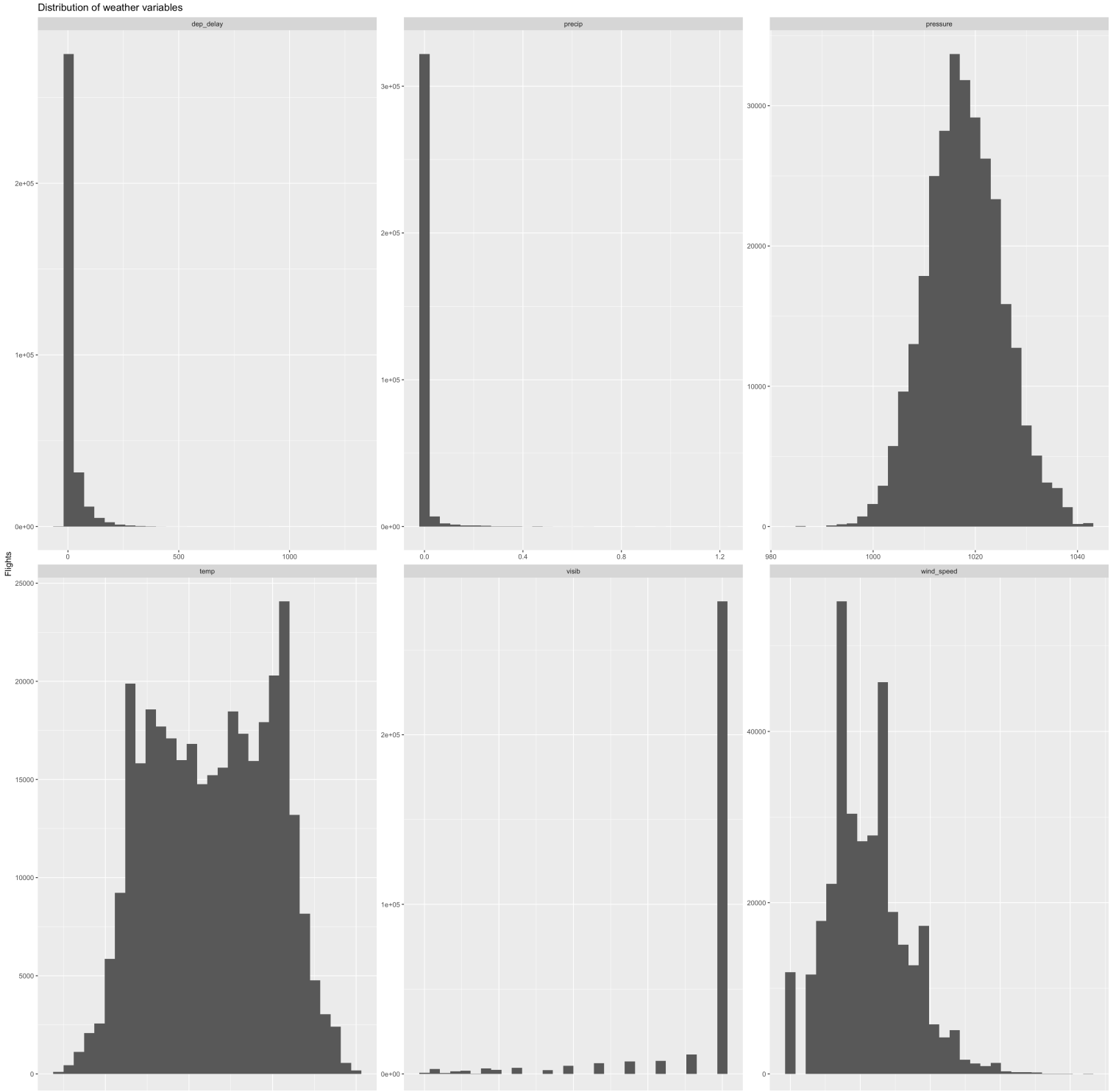
```
library(ggplot2)

ggplot(flights_weather %>% filter(!is.na(flights_weather$wind_speed)), aes(wind_speed)) + geom_hi
```

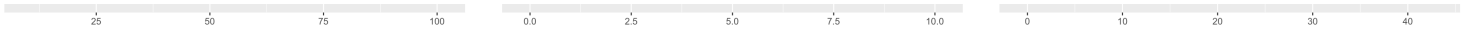

Wind Speed vs. Flights
Flights
40000
20000
0

It seems like most flights recorded wind-speeds between 5-15 mph.

To quickly look at the distribution of multiple (potential) weather predictors:

```
library(dplyr)
library(tidyr)

flights_weather %>%
  select(dep_delay, precip, visib, wind_speed, temp, pressure) %>%
  pivot_longer(everything(), names_to = "var", values_to = "value") %>%
  ggplot(aes(value)) +
  geom_histogram(bins = 30, na.rm = TRUE) +
  facet_wrap(~ var, scales = "free") +
  labs(title = "Distribution of weather variables", x = NULL, y = "Flights")
```


Distribution of weather variables
dep_delay
precip
pressure
temp
visib
wind_speed
Flights
0e+00
1e+05
2e+05
3e+05
0
10000
20000
30000
40000
5000
15000
25000
0
500
1000
0.0
0.4
0.8
1.2
980
1000
1020
1040



From here, excluding the observations I've already made for wind speed and precipitation, I see that most flights experience a pressure of ~1020 mb, and the pressure-flights data is normally distributed.

Most flights seem to have excellent visibility, and the temperatures see massive fluctuations between 25-80F, which makes sense considering the nycflights13 package has data for all flights in and out of NYC in 2013, which covers the entire range of seasons.

Question 7:

Calculate the average departure delay for each **day**. Which day had the worst average length of delay for departures? Now calculate the averages by **day** and **origin**. Which airport had the worst single day for delays and when was it? Now calculate the averages by **hour** and **origin**. Which airport had the worst single hour for delays (and when was it)?

For the first part of this question (Calculate the average departure delay for each **day**. Which day had the worst average length of delay for departures?), I am using the flights data, which contains the dep_delay per flight. I could group by just the day, but that is not useful since the "day" is meaningless without the "month" value (As there will then be only 31 values, computing the average delay picking one day from each month where that date exists). I could also then group by year, but since this data is only for 2013, it is not necessary.

```
library(dplyr)

daily_data <- flights %>%
  filter(!is.na(dep_delay)) %>%
  group_by(month, day) %>%
  summarise(avg_dep_delay = mean(dep_delay), n = n(), .groups = "drop")

worst_day_delay <- daily_data %>% filter(avg_dep_delay == max(avg_dep_delay, na.rm = TRUE))

daily_data
```

```
# A tibble: 365 × 4
   month   day avg_dep_delay     n
```

```
   <int> <int>         <dbl> <int>
 1     1     1         11.5    838
 2     1     2         13.9    935
 3     1     3         11.0    904
 4     1     4          8.95   909
 5     1     5          5.73   717
 6     1     6          7.15   831
 7     1     7          5.42   930
 8     1     8          2.55   895
 9     1     9          2.28   897
10     1    10          2.84   929
# ℹ 355 more rows
```

```
worst_day_delay
```

```
# A tibble: 1 × 4
  month   day avg_dep_delay     n
  <int> <int>         <dbl> <int>
1     3     8          83.5   799
```

The worst day for delays was the 8th of March, The average delay was 83.5 minutes. A quick google search says that there was significant snowstorm that day, which corroborates our observation.

For the next part of this question (Now calculate the averages by **day** and **origin**. Which airport had the worst single day for delays and when was it?), it is as simple as grouping by origin along with month and day.

```
library(dplyr)

daily_origin_data <- flights %>%
  filter(!is.na(dep_delay)) %>%
  group_by(month, day, origin) %>%
  summarise(avg_dep_delay = mean(dep_delay), n = n(), .groups = "drop")

worst_day_origin_delay <- daily_origin_data %>% filter(avg_dep_delay == max(avg_dep_delay, na.rm

daily_origin_data
```

```
# A tibble: 1,095 × 5
   month   day origin avg_dep_delay     n
   <int> <int> <chr>          <dbl> <int>
 1     1     1 EWR            17.5    304
 2     1     1 JFK            12.2    296
 3     1     1 LGA             3.13   238
 4     1     2 EWR            25.3    344
 5     1     2 JFK             8.14   320
 6     1     2 LGA             6.06   271
 7     1     3 EWR             8.45   333
 8     1     3 JFK            13.8    318
 9     1     3 LGA            10.8    253
10     1     4 EWR            12.1    337
# ℹ 1,085 more rows
```

```
worst_day_origin_delay
```

```
# A tibble: 1 × 5
  month   day origin avg_dep_delay     n
  <int> <int> <chr>          <dbl> <int>
1     3     8 LGA             106.   229
```

LGA airport had the worst single day for delays and it was on 8th March, 2013. An interesting thing to note is that the average delay at Laguardia on that day is worse than the overall delay for the 8th of March, 2013.

For the final part of this question (Now calculate the averages by hour and origin. Which airport had the worst single hour for delays (and when was it)?), I will use the “time_hour” variable from the flights dataset, which represents the scheduled date and hour for each flight’s departure.

```
library(dplyr)

hr_origin_delays <- flights %>%
  filter(!is.na(dep_delay)) %>%
  group_by(origin, time_hour) %>%
  summarise(average_delay = mean(dep_delay), n = n(), .groups = "drop")

worst_hr <- hr_origin_delays %>%
  filter(average_delay == max(average_delay, na.rm = TRUE))
```

```
hr_origin_delays
```

```
# A tibble: 19,434 × 4
   origin time_hour           average_delay     n
   <chr>  <dttm>                      <dbl> <int>
 1 EWR    2013-01-01 05:00:00         -1        2
 2 EWR    2013-01-01 06:00:00          3.06    18
 3 EWR    2013-01-01 07:00:00         14.2     12
 4 EWR    2013-01-01 08:00:00          0.75    20
 5 EWR    2013-01-01 09:00:00          9.05    19
 6 EWR    2013-01-01 10:00:00          2.06    18
 7 EWR    2013-01-01 11:00:00          0       11
 8 EWR    2013-01-01 12:00:00          6.73    22
 9 EWR    2013-01-01 13:00:00         28.7     28
10 EWR    2013-01-01 14:00:00         33.7     18
# ℹ 19,424 more rows
```

```
worst_hr
```

```
# A tibble: 1 × 4
  origin time_hour           average_delay     n
  <chr>  <dttm>                     <dbl> <int>
1 LGA    2013-07-28 21:00:00         280.     3
```

Interestingly, the worst single hour for delays is still at LGA airport, but the delay was not on 8th March, 2013, instead, it is on 07/28/2013, and the worst hour was 9 PM.

Question 8:

Compute the average delay by destination, then add that information to the `airports` data frame. Make a map showing the spatial distribution of delays. Use the color of the points to display the average delay for each airport.

For this question, I used arr_delay and not dep_delay. The flights table contains only flights that depart from NYC, which means that for non-NYC airports (the destinations), we have the arrival data into them but no departure data from them. So we cant really use dep_delay for the destination, or combine arrival and departure delays because (in

my opinion), the arr_delay already shows us the net outcome (for example the airplane could've left on time and arrived late, or left late and make up time to arrive on time and so on). So if we added/averaged the delays, we will be double counting the same flight's delay.

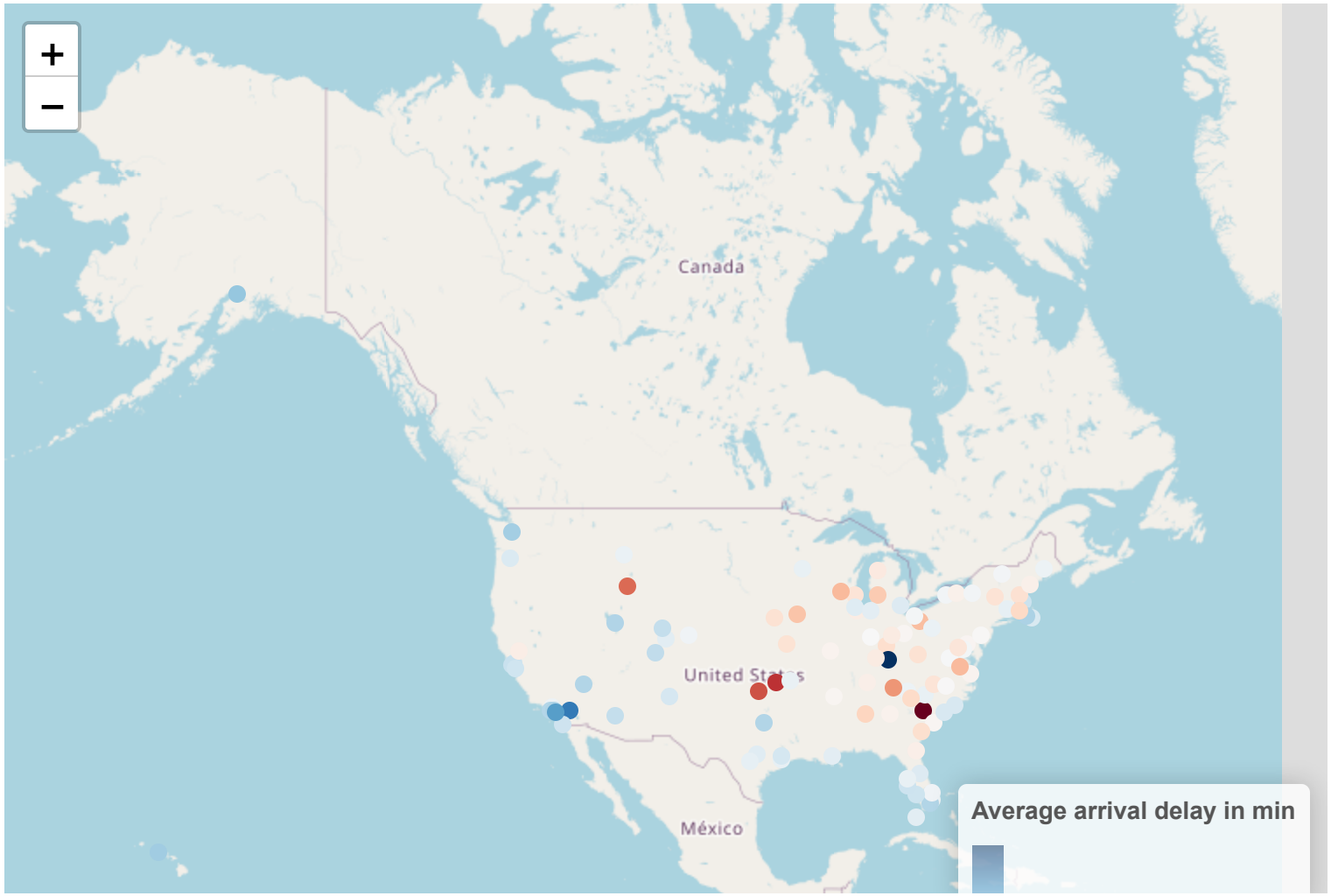
```
library(dplyr)
library(leaflet)

avg_dest_delays <- flights %>%
  group_by(dest) %>%
  summarise(avg_arr_delay = mean(arr_delay, na.rm = TRUE), n = n())

airports_delay <- airports %>%
  select(faa, name, lat, lon) %>%
  right_join(avg_dest_delays, by = c("faa" = "dest")) %>%
  filter(!is.na(lat), !is.na(lon), !is.na(avg_arr_delay))

color_pal <- colorNumeric("RdBu", airports_delay$avg_arr_delay, reverse = TRUE)

# I reversed the palette here because the usual convention is to go from blue to red, as values g

avg_dest_delays
```

```
# A tibble: 105 × 3
   dest  avg_arr_delay     n
   <chr>         <dbl> <int>
 1 ABQ            4.38   254
 2 ACK            4.85   265
 3 ALB           14.4    439
 4 ANC           -2.5      8
 5 ATL           11.3  17215
 6 AUS            6.02  2439
 7 AVL            8.00   275
 8 BDL            7.05   443
 9 BGR            8.03   375
10 BHM           16.9    297
# ℹ 95 more rows
```

```r
summary(airports_delay$avg_arr_delay)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
-22.000   4.548   8.369   8.934  12.672  41.764 
```

```r
leaflet(airports_delay) |>
  addTiles() |>
  addCircleMarkers(~lon, ~lat, color = ~color_pal(avg_arr_delay), fillOpacity = 1, stroke = FALSE
  addLegend("bottomright", pal = color_pal, values = ~avg_arr_delay,
            title = "Average arrival delay in min")
```



Question 9:

In the flights_weather dataset, there are 9 variables: precip, visib, wind_speed, wind_gust, temp, dewp, humid, wind_dir and pressure.

In order to answer which weather phenomena had the most impact on flight delays, I would ideally like to look at the impact of each of these phenomena on delay (dep_delay and not arr_delay, because the weather data is for NYC and not for the arrival region).

To do this, I would put my weather variables into buckets, based on some quick googling (and in some cases, help from ChatGPT), as well as the reported min/max values in flights_weather, as seen from our output of summary(flights_weather).

A quick run-down of the buckets I've chosen to use:

1. precip: 0 (no precipitation), 0-0.05, 0.05-0.2, >0.2 inches
2. visib: <=2, 2-5, 5-10 and >10
3. wind_speed: 0-5, 5-10, 10-15, 15-20, >20
4. wind_gust = 0, 0-20, 20-40, >40
5. temp: <= 32 (freezing point), 32-50, 50-70, >70
6. dewp: <= 20, 20-40, 40-60, >60
7. humid: <= 40, 40-60, 60-80, >80
8. pressure: <= 1000, 1000-1015, 1015-1030, >1030
9. wind_dir: I would create 8 groups based on the cardinal directions.

Defining my buckets:

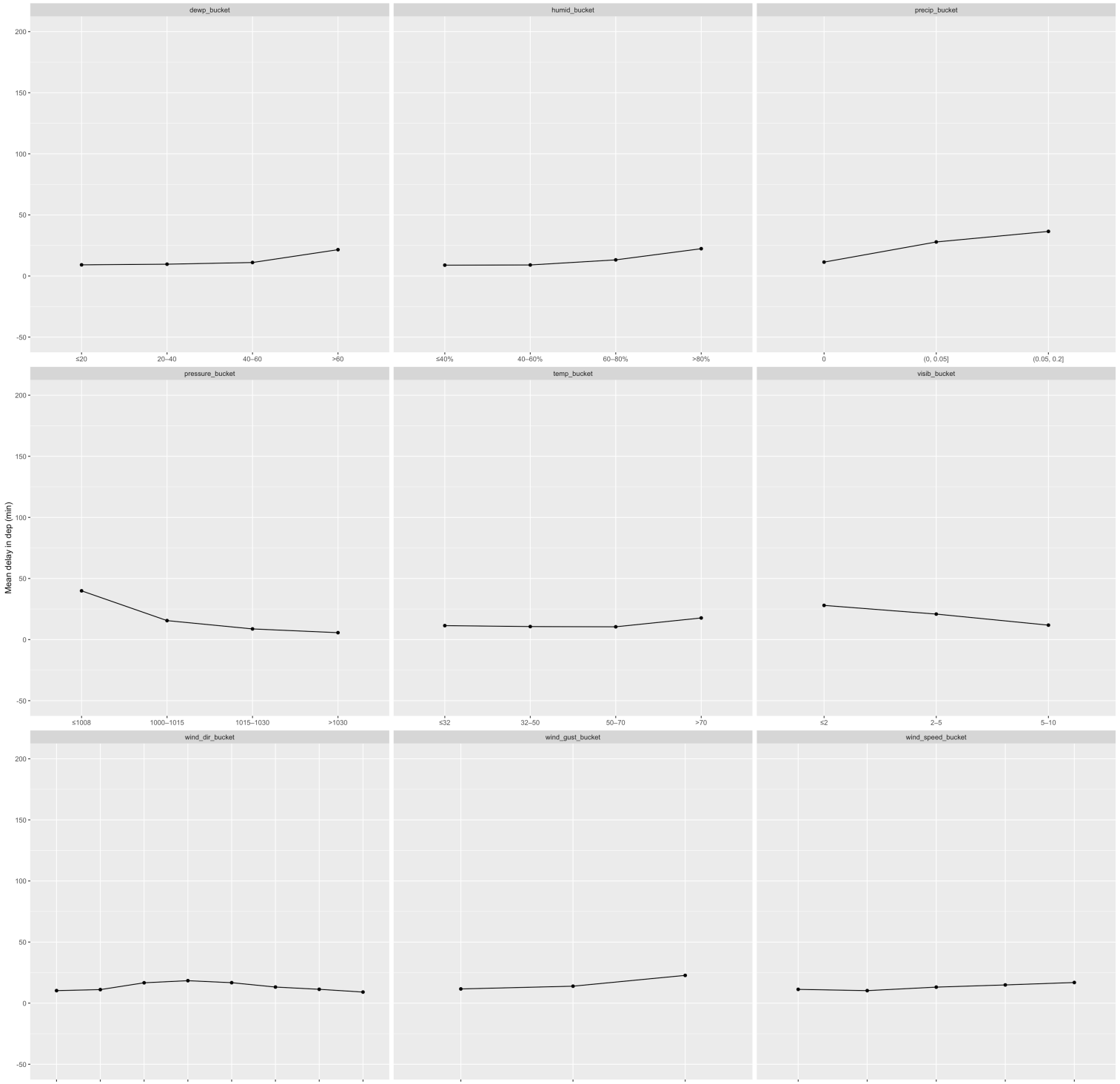
```r
library(dplyr)
library(tidyr)
library(ggplot2)

new_table <- flights_weather %>%
  filter(!is.na(dep_delay)) %>%
  select(dep_delay, arr_delay, origin, temp, dewp, humid, wind_dir, wind_speed, wind_gust, precip

precip_bucket <- ifelse(is.na(new_table$precip), NA_character_, ifelse(new_table$precip == 0, "0"

visib_bucket <- cut(new_table$visib, breaks = c(-Inf, 2, 5, 10, Inf), labels = c("≤2", "2–5", "5-

wind_speed_bucket <- cut(new_table$wind_speed, breaks = c(-Inf, 5, 10, 15, 20, Inf), labels = c('

wind_gust_bucket <- cut(new_table$wind_gust, breaks = c(-Inf, 0, 20, 40, Inf), labels = c("0", "0

temp_bucket <- cut(new_table$temp, breaks = c(-Inf, 32, 50, 70, Inf), labels = c("≤32", "32–50",

dewp_bucket <- cut(new_table$dewp, breaks = c(-Inf, 20, 40, 60, Inf), labels = c("≤20", "20–40",

humid_bucket <- cut(new_table$humid, breaks = c(-Inf, 40, 60, 80, Inf), labels = c("≤40%", "40–60

pressure_bucket <- cut(new_table$pressure, breaks = c(-Inf, 1000, 1015, 1030, Inf), labels = c("≤

wind_dir_new <- (new_table$wind_dir %% 360)
wind_dir_bucket <- cut(wind_dir_new, breaks = c(-0.1, 22.5, 67.5, 112.5, 157.5, 202.5, 247.5, 292

# Note: I asked ChatGPT and Google for help in defining the buckets for wind_dir.

# Next I added the buckets into my data

flights_weather_binned <- new_table %>%
  mutate(precip_bucket = factor(precip_bucket, levels = c("0", "(0, 0.05]", "(0.05, 0.2]", ">0.2"
         visib_bucket = visib_bucket,
         wind_speed_bucket = wind_speed_bucket,
         wind_gust_bucket = wind_gust_bucket,
         temp_bucket = temp_bucket,
         dewp_bucket = dewp_bucket,
```

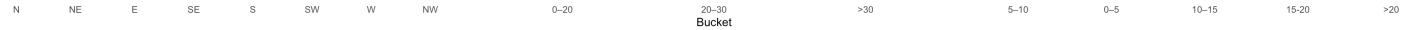
```
          humid_bucket = humid_bucket,
          pressure_bucket = pressure_bucket,
          wind_dir_bucket = wind_dir_bucket)
```

Now that I have made my buckets and added it into my table, I want to plot panels of mean delay vs the phenomena, as well as a scatter plot of delays and quantify the correlation.

```
dep_delay_weather <- flights_weather_binned %>%
  select(dep_delay, precip_bucket, visib_bucket, wind_speed_bucket, wind_gust_bucket, temp_bucket
  pivot_longer(cols = c(precip_bucket, visib_bucket, wind_speed_bucket, wind_gust_bucket, temp_bu
  filter(!is.na(bucket)) %>%
  group_by(var, bucket) %>%
  summarise(mean_dep_delay = mean(dep_delay), n = n(), .groups = "drop")

# Above: I averaged the dep_delay for each bucket
# The plot:
ggplot(dep_delay_weather, aes(x = bucket, y = mean_dep_delay, group = 1)) +
  geom_line() + geom_point() +
  facet_wrap(~ var, scales = "free_x") +
  coord_cartesian(ylim = c(-50, 200)) +
  labs(title = "Mean departure delay vs bucketed weather", x = "Bucket", y = "Mean delay in dep
```

Mean departure delay vs bucketed weather





This is good, because we now see the effect of each of our buckets on the delay in departure. But if I wanted to quickly see the numerical, raw value effect of each of the weather data, I could do this:

```
weather_effect <- dep_delay_weather %>%
  group_by(var) %>%
  summarise(change_in_mean = max(mean_dep_delay) - min(mean_dep_delay), .groups = "drop") %>%
  arrange(desc(change_in_mean))

weather_effect
```

```
# A tibble: 9 × 2
  var               change_in_mean
  <chr>                      <dbl>
1 pressure_bucket            34.3
2 precip_bucket              25.1
3 visib_bucket               16.2
4 humid_bucket               13.5
5 dewp_bucket                12.4
6 wind_gust_bucket           11.1
7 wind_dir_bucket             9.30
8 temp_bucket                 7.24
9 wind_speed_bucket           6.65
```

Explanation: I found the “delta”, essentially the difference between the maximum and minimum delay for each weather variable, and the pressure seems to have the most drastic change, leading me to believe that air pressure is a serious determinant of flight delays.

For the next set of plots, I want to calculate the correlation between weather phenomena and the delay.

```
scatter_plot_data <- new_table %>%
  pivot_longer(cols = c(temp, dewp, humid, wind_dir, wind_speed, wind_gust, precip, pressure, vis

# I'm removing wind direction because theyre cartesian diresctions and not numerical values

correlation <- scatter_plot_data %>% filter(var != "wind_dir") %>% group_by(var) %>% summarise(co
```

```
correlation
```

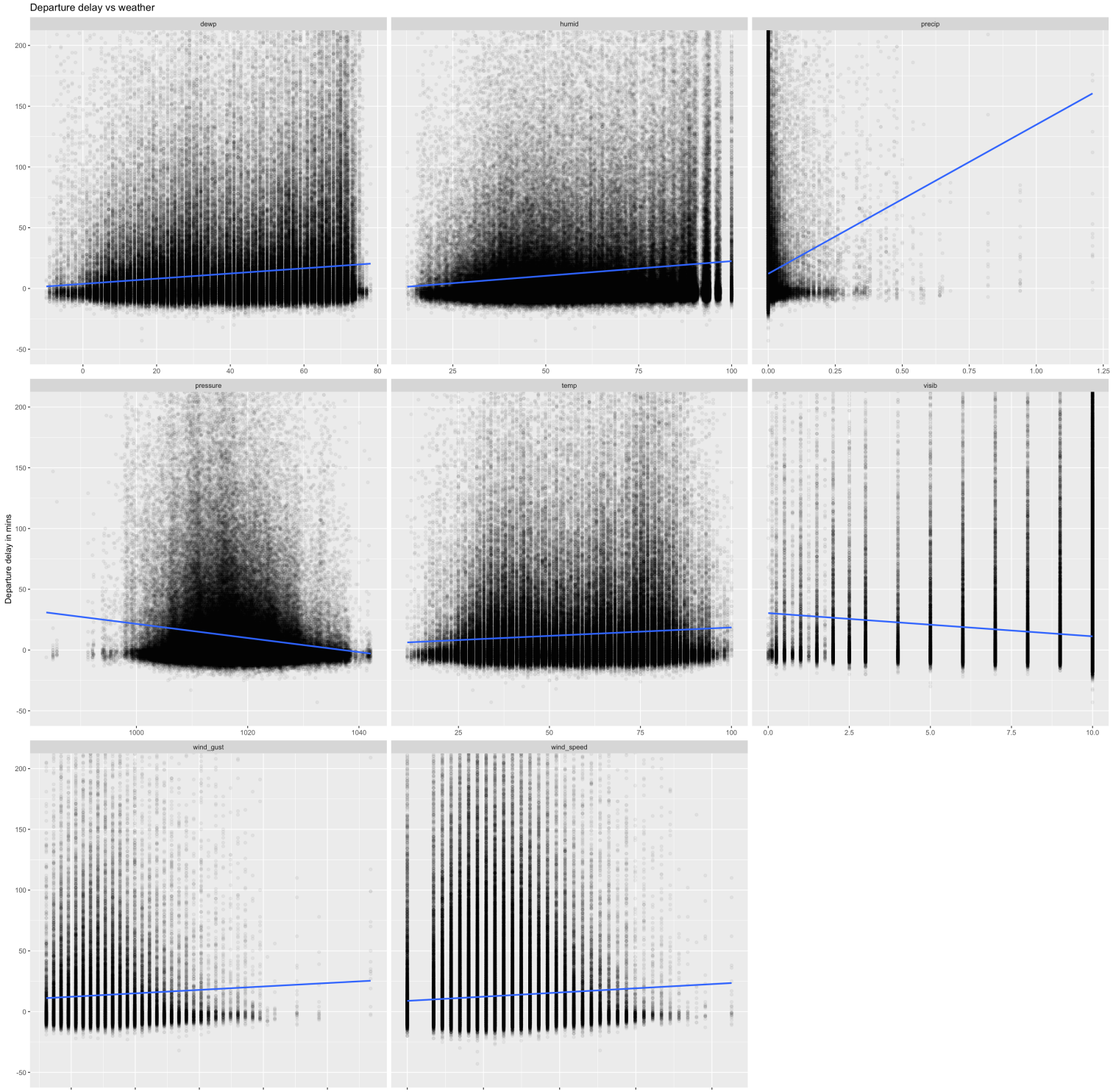
```
# A tibble: 8 × 2
  var        correlation
  <chr>            <dbl>
1 humid           0.117
2 pressure       -0.114
3 dewp            0.102
4 visib          -0.0941
5 precip          0.0904
6 temp            0.0615
7 wind_speed      0.0474
8 wind_gust       0.0413
```

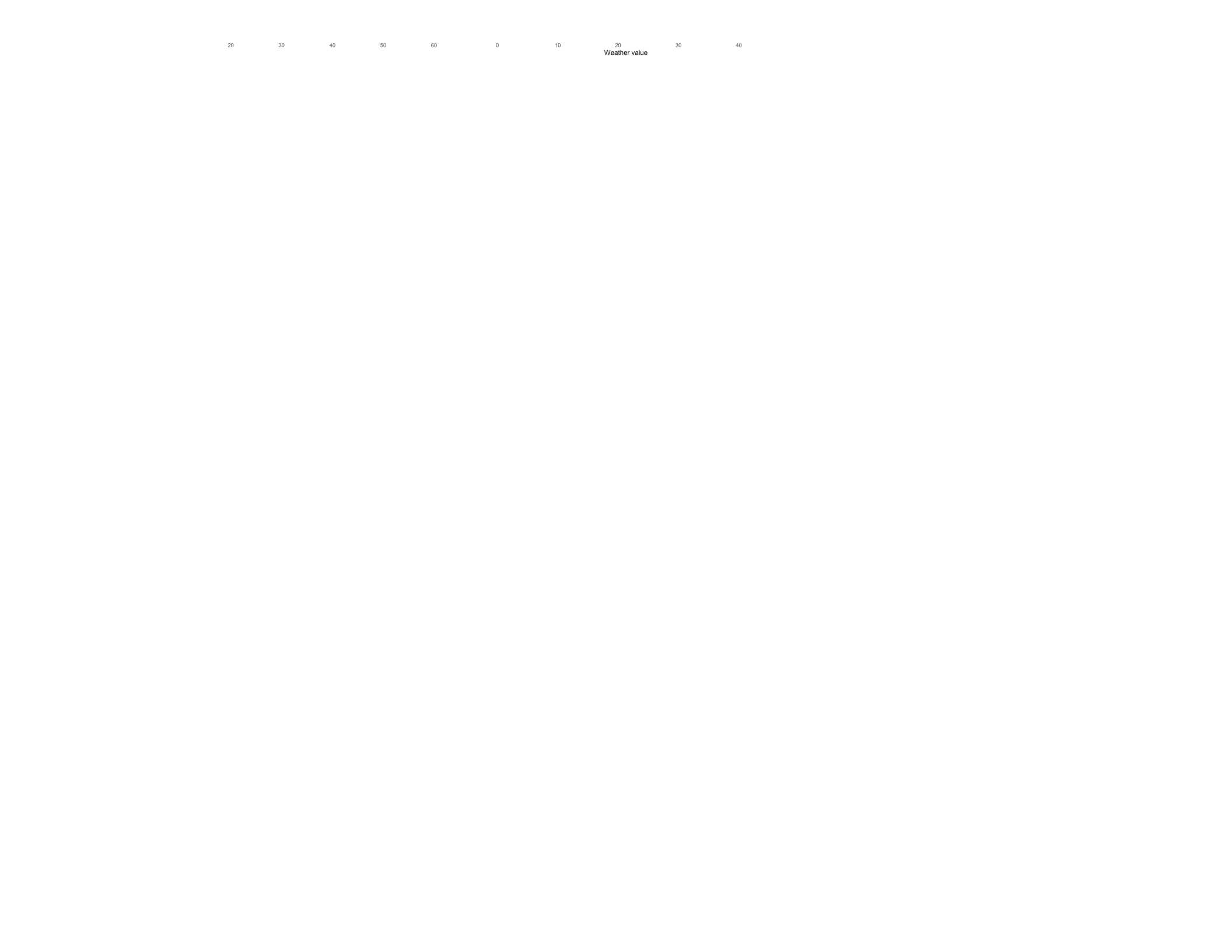
So it seems like pressure really is the main determinant of delays, followed by precipitation.

Now for the plots:

```
ggplot(scatter_plot_data %>% filter(var != "wind_dir"), aes(x = value, y = dep_delay)) +
  geom_point(alpha = 0.05, na.rm = TRUE) +
  geom_smooth(method = "lm", se = FALSE) +
  facet_wrap(~ var, scales = "free_x") +
  coord_cartesian(ylim = c(-50, 200)) +
  labs(title = "Departure delay vs weather",
       x = "Weather value", y = "Departure delay in mins")
```

```
`geom_smooth()` using formula = 'y ~ x'

Warning: Removed 296403 rows containing non-finite outside the scale range
(`stat_smooth()`).
```


Departure delay vs weather
dewp
humid
precip
pressure
temp
visib
wind_gust
wind_speed
Departure delay in mins
200
150
100
50
0
-50
0
20
40
60
80
25
50
75
100
0.00
0.25
0.50
0.75
1.00
1.25
1000
1020
1040
0.0
2.5
5.0
7.5
10.0


20
30
40
50
60
0
10
20
30
40
Weather value