HW2

From HW 1:

```
#setwd("G:/My Drive/STAT 3105")

library(jsonlite)
```

```
## Warning: package 'jsonlite' was built under R version 4.0.3
```

```
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
```

```
#process data
data <- fromJSON("unemployment_cpi_unempl_2000_2020.json")
df <- as.data.frame(bind_rows(data$Results))
a <- data.frame(df$data[1])
b <- data.frame(df$data[2])

#remove unnecessary columns
a <- subset(a, select = -footnotes)
b <- subset(b, select = -footnotes)

#reassign 1st/2nd half status to each month
a$period[a$period=="M12" | a$period == "M11" | a$period == "M10" | a$period == "M09" | a$period == "M08

a$period[a$period=="M06" | a$period == "M05" | a$period == "M04" | a$period == "M03" | a$period == "M02

#change type of column from char to double
a$value = as.double(a$value)

#obtain average unemployment per period for first part
first_part <- a %>%
  group_by(year, period) %>%
  summarise(mean(value))
```

```
## `summarise()` regrouping output by 'year' (override with `.groups` argument)
```

```
names(first_part) <- c("year", "period", "unemployment")

#change type of column from char to double
b$value = as.double(b$value)

#obtain average CPI per period for 2nd part
second_part <- b %>%
  group_by(year, periodName) %>%
  summarise(mean(value))
```

```
## `summarise()` regrouping output by 'year' (override with `.groups` argument)
```

```
names(second_part) <- c("year", "period", "cpi")
```

*Add CPI and Export to CSV:

```
#add CPI to df

first_part$cpi <- second_part$cpi
df <- first_part

#export to csv
write.csv("CPI.csv")
```

```
## "","x"
## "1","CPI.csv"
```

```
#calculate inflation
df$inflation <- rep(1:20)

#convert to dataframe
df <- as.data.frame(df)

inflation <- rep(1:20)

counter = 1
for(index in df$cpi){
   if(counter == 1) {
    inflation[counter] <- NA
    counter = counter + 1
  }
  inflation[counter] <- (df$cpi[counter] - df$cpi[counter - 1])/df$cpi[counter -1] * 100
  counter = counter + 1
}

#remove last NA
inflation <- inflation[1:20]

#add column to df
df$inflation <- inflation
```

Graph:

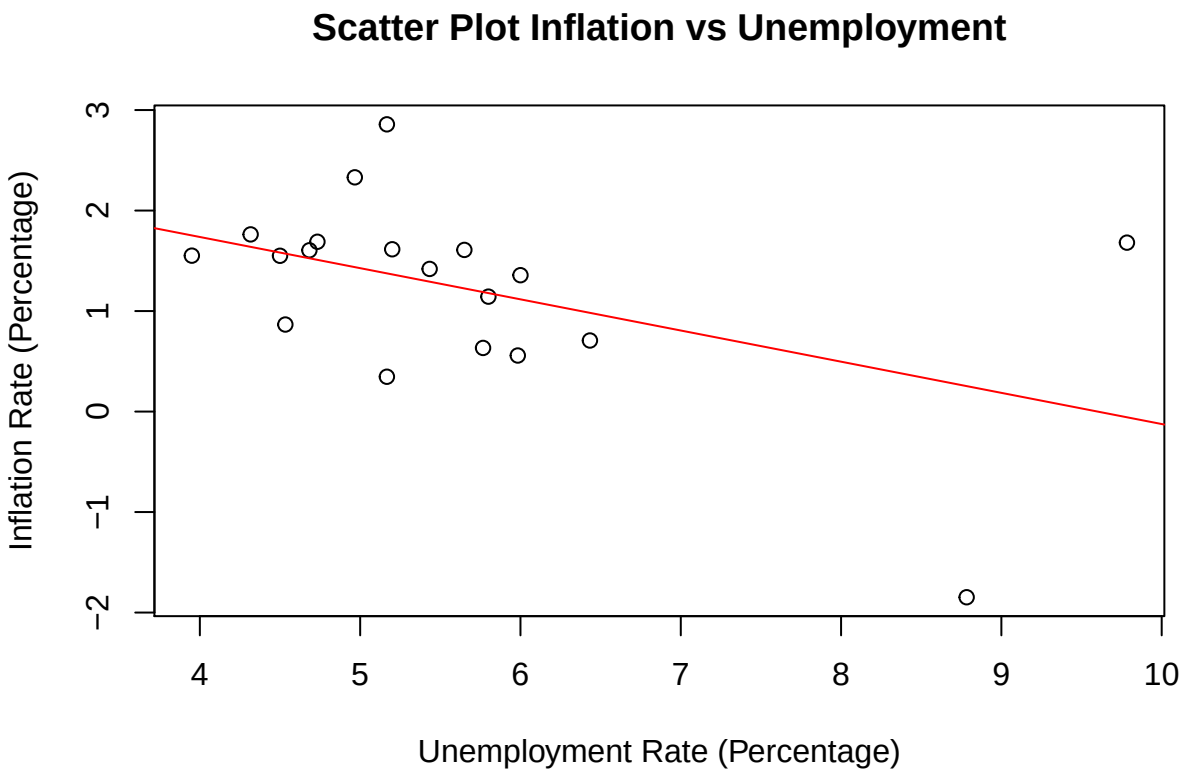
Problem 1

1a

```
mod <- lm(df$inflation ~ df$unemployment)

summary(mod)
```

```
##
## Call:
## lm(formula = df$inflation ~ df$unemployment)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.10071 -0.41553  0.07961  0.24506  1.73776
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       2.9777     0.8251   3.609  0.00217 **
## df$unemployment  -0.3102     0.1423  -2.180  0.04361 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8768 on 17 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.2185, Adjusted R-squared:  0.1725
## F-statistic: 4.752 on 1 and 17 DF,  p-value: 0.04361
```

```
#scatter plot inflation vs unemployment
plot(df$unemployment, df$inflation,
     main = "Scatter Plot Inflation vs Unemployment", xlab = "Unemployment Rate (Percentage)",
     ylab = "Inflation Rate (Percentage)")
abline(lm(df$inflation ~ df$unemployment), col = "red")
```



This model gave a slope for inflation of -0.3102 with a p value of 0.0436. The p value represents the probability that the null hypothesis that there is no relationship between the variables is true. Since the p value is very low ($0.05 > 0.0436$), we can be confident that there is likely a relationship, and that the coefficient is not zero. The slope being negative tells us that inflation decreases slightly as unemployment increases, that is for every 1 percentage increase in unemployment, inflation decreases by -0.31%.

1b

#From lecture 5 slides

We assume that there is a large enough sample size, and that the errors in our data are normally distributed.

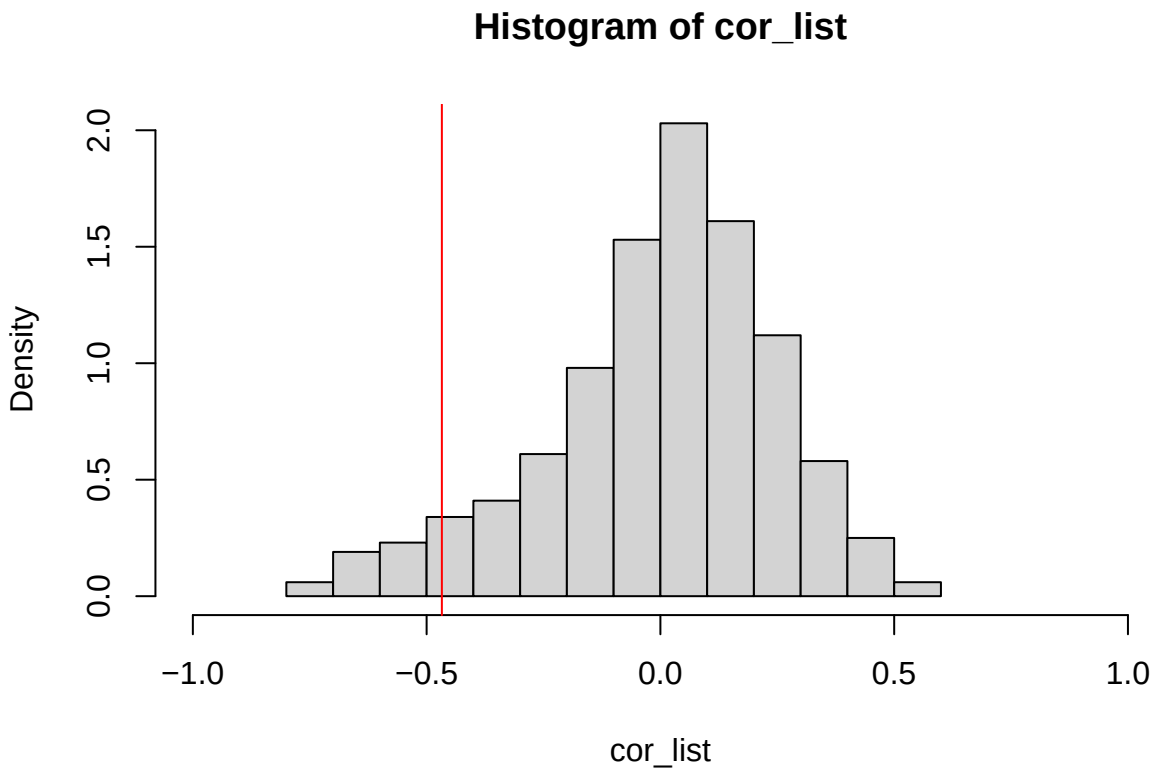
1c

```
#remove first row so that vectors are equal lengths
cor_df <- df[-1,]

#calulate correlation between 2 variables
cor <- cor(cor_df$unemployment, cor_df$inflation)

cor
```

```
## [1] -0.4674133
```

```
## Permutation Test
M <- 1000
cor_list <- c()
```

```r
for(i in 1:M){
  idx_permute <- sample(1:length(cor_df$unemployment), size = length(cor_df$unemployment), replace = F)
  x_permute <- cor_df$unemployment[idx_permute]
  cor_permute <- cor(x_permute, cor_df$inflation)
  cor_list <- c(cor_list, cor_permute)
}
hist(cor_list, probability = T, xlim = c(-1, 1))
abline(v = cor(cor_df$unemployment, cor_df$inflation), col = "red")
```



The correlation between the two variables was -0.4674133.

If there variables were independent, the correlation would be close to 0.0 on the histogram (indicating no correlation when the variables were shuffled). This did not appear to be the case, as we had correlations from -0.9 to 0.5. This is in line with our correlation of -0.46, and means that there is likely a relationship between the variables.

2a:

```r
library(dplyr)
library(jsonlite)

#read JSON
data <- read_json("nytimes_2020_articles_with_comments.json", simplifyVector = TRUE)
```

This data is of comments on 2020 NYT articles. Each list represents a different article, with the each of rows in each of the 1406 lists representing a different comment. We are given a userID number, the user's username,

the content of their comments, the date when the comment was last updated, the date it was approved, how many times it was recommended, how many times it was replied to, a binary variable indicating weather the comment was featured in the editor section and finally another binary variable indicating weather the comment was posted anonymously.

2b:

```
#create variables to represent maximum length and URL, for foor loop later
max_length <- 0
name <- ""


#iterate through JSON, find the article with the highest number of replies
for(i in 1:length(data)){
  url <- as.character(names(data[i]))
  length <- length(data[[i]][[2]])

  if(length > max_length){
    max_length <- length
    name <- url
    }
}

print(paste("Article: ",name, " Length: ", max_length, sep = ""))
```

```
## [1] "Article: nyt://article/a6546112-4515-5f3d-8b38-77c3f65d526d Length: 4400"
```

The article with the most comments is found at the link: nyt://article/a6546112-4515-5f3d-8b38-77c3f65d526d . It had 4400 replies.

2c

```
#turn object into a DF
df <- as.data.frame(bind_rows(data))


#add a URL column to the DF to represent which article a given article was pulled from

#create new list
articles <- c()

#iterate through the JSON object. Get the URL, figure out how many times that article appears (length),
for (i in 1:length(data)){
  url <- as.character(names(data[i]))
  length <- length(data[[i]][[2]])
  article <- rep(url, length)
  articles <- append(articles, article)
}

df$url <- articles


#create a new column for word count and unique words
df$WordCount <- NA
```

```
df$Num.Unique.Words <- NA


#iterate through DF, split string and turn the result into a list (each word is an index). Add number o

for(i in 1:length(df$commentBody)){

  c <- unlist(strsplit(df$commentBody[i], split = "\\s+"))

  df$WordCount[i] <- length(c)
  df$Num.Unique.Words[i] <- length(unique(c))
}

#rank updates

#convert the UpdateDate column to a DateTime object
library(lubridate)
```

```
## Warning: package 'lubridate' was built under R version 4.0.5
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
df$Date <- df$updateDate %>% as.integer() %>% as_datetime()

#create rankings for the dates
df <- df %>%
  group_by(url) %>%
  mutate(time_rank = rank(Date, ties.method = "first"))

#display first few rows of DF
head(df)
```

```
## # A tibble: 6 x 14
## # Groups:   url [1]
##   userID userDisplayName commentBody updateDate approveDate recommendations
##    <int> <chr>           <chr>       <chr>      <chr>                 <int>
## 1 2.40e7 Timit           "Sanders i~ 1583304108 1583149287                2
## 2 6.65e7 Ed              "Democrats~ 1583180043 1583145168                1
## 3 6.90e7 Victor          "I want a ~ 1583304169 1583114186                6
## 4 5.21e7 Sally Peabody   "Joe Biden~ 1583110435 1583110435                0
## 5 6.65e7 Kai             "The key q~ 1583180213 1583108077                1
## 6 6.10e7 Murphy          "Let Sande~ 1583114204 1583102959                2
## # ... with 8 more variables: replyCount <int>, editorsSelection <lgl>,
## #   isAnonymous <lgl>, url <chr>, WordCount <int>, Num.Unique.Words <int>,
## #   Date <dttm>, time_rank <int>
```

I used the head function above to display the first few rows of the data frame for your review.

2d

```r
#Filter DF to only include March Articles
march <- df %>%
  filter(Date > as.Date("2020-03-01 00:00:00") & Date < as.Date("2020-04-01 00:00:00"))

#Get number of March Articles
length(unique(march$url))
```

```
## [1] 1222
```

```r
#dimensions of df
dim(march)
```

```
## [1] 346290     14
```

There are about 1222 unique articles for March 2020 in this dataset. Using the dims function, we can see that there are 346290 comments that pertain to March 2020 in this dataset. This dataset likely contains a representative sample of comments on articles on March 2020 NYT articles, as 1222 articles is a decent sample size.

According to the piazza, this dataset actually represents the entire population of NYT articles from this time period. This makes it an actual perfect representation of the population of interest.

Project Question Series 1

1

I would like to have a data set, perhaps provided by US air traffic control, that lists flight numbers, and their scheduled arrival and departure times, as well as their actual arrival and departure times. For me, the usual cause of flight delays is weather, so I would like to combine this dataset with a dataset that gives information on historical weather. Maybe I could code a categorical variable for the weather to represent how unfavorable the weather conditions were in the hour prior to takeoff/landing (mild, bad, really bad, hurricane).

The Bureau of Transportation Statistics publishes this data on the internet (https://transtats.bts.gov/ONTIME/)

2

One potential issue that I might face is that during the COVID-19 pandemic, many governments decided to spontaneously cancel international travel to prevent the spread of the virus. Thus, flights from February-April 2020 might be more likely to be cancelled, which will affect the results, and make it look like flights are more likely to be cancelled than they otherwise would be, as there were a significant amount of flight cancellations (which would obviously result on the flights not arriving on time). I would thus exclude data from these months.1

Dataset

How many columns are in the dataset? What does each column stand for?

```r
flights <- read.csv("pnwflights14.csv")

dim(flights)
```

```
## [1] 148236     16
```

```
colnames(flights)
```

```
##  [1] "year"      "month"     "day"       "dep_time"  "dep_delay" "arr_time"
##  [7] "arr_delay" "carrier"   "tailnum"   "flight"    "origin"    "dest"
## [13] "air_time"  "distance"  "hour"      "minute"
```

There are 16 columns in the data. The columns give information including when the flights departed and arrived, when they were scheduled to depart/arrive, how long any delays were, flight number, origin/destination airport, air time and flight distance.

How many airlines are in the dataset? Please aggregate the number of flights per month and visualize this with clear labels

```
length(unique(flights$carrier))
```

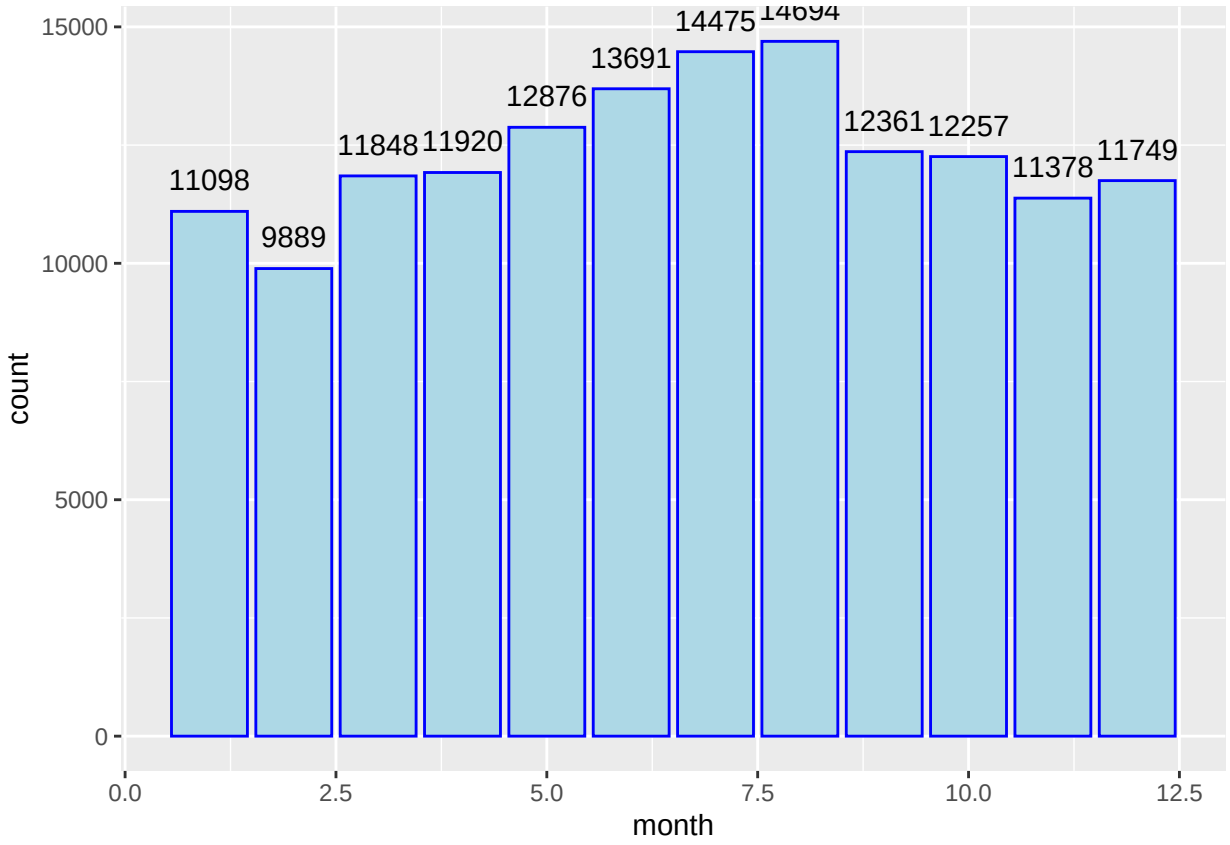
```
## [1] 11
```

```
#number of flights per month
flights %>% group_by(month) %>%
  summarise(n = n())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 12 x 2
##    month     n
##    <int> <int>
##  1     1 11098
##  2     2  9889
##  3     3 11848
##  4     4 11920
##  5     5 12876
##  6     6 13691
##  7     7 14475
##  8     8 14694
##  9     9 12361
## 10    10 12257
## 11    11 11378
## 12    12 11749
```

```
library(ggplot2)

#Bar Graph of # of flights per month
ggplot(data = flights, aes(x=month)) +
  geom_bar(color = "blue", fill = "light blue") +
  geom_text(stat='count', aes(label=..count..), vjust=-1)
```



There are 11 airlines in the dataset.

Above I have included a graph with the number of flights per month. All of the data is from 2014, there was nothing in the dataset in any other year.

What data quality issues have you noticed? What results will be impacted by these issues?

Departure time has negative values, which shouldn't be possible. The values should be between 00:00 and 24:00. There must have been some error in the data collection.

Distance and air time also have negative values.

Additionally, some columns have certain values as NA or with no data.

Do certain airlines departure/arrive late their flights more often than others?

```
#convert the timing into a date time object
flights$date <- paste(flights$year, "-", flights$month, "-", flights$day, " ", flights$hour, ":", flight

flights$dep_date <- strptime(flights$date, format="%Y-%m-%d %H:%M")

flights2 <- flights


flights2$arr_time2 <- substr(as.POSIXct(sprintf("%04.0f", flights$arr_time), format='%H%M'), 12, 16)

flights2$arr_date <- paste(flights$year, "-", flights$month, "-", flights$day, " ", flights2$arr_time2,

flights2$arr_date <- strptime(flights2$arr_date, format="%Y-%m-%d %H:%M")
```

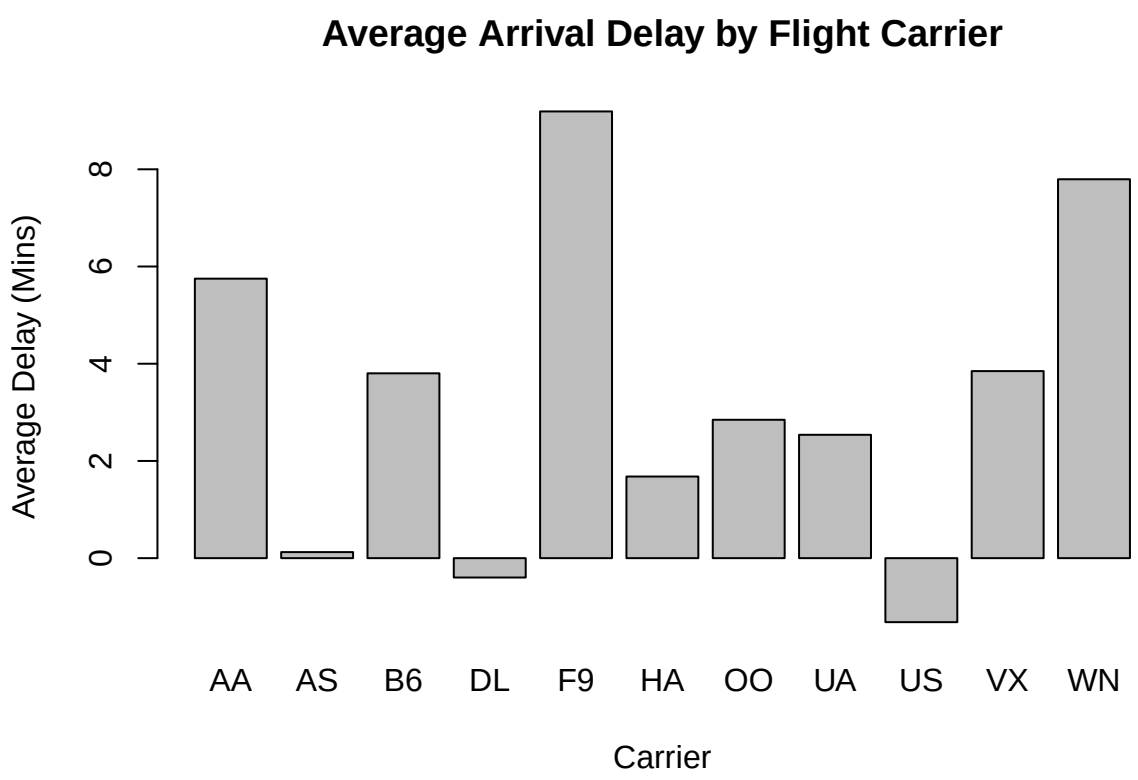
```r
flights$arr_date <- flights2$arr_date
```

```r
#number of flights per month

avg_delay <- flights %>% group_by(carrier) %>%
  summarise(avg_delay = mean(arr_delay, na.rm = TRUE), avg_dep_delay = mean(dep_delay, na.rm = TRUE), av
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
carriers <- unique(flights$carrier)



barplot(height = avg_delay$avg_arr_delay, names.arg = avg_delay$carrier, data = avg_delay, main = "Avera
```

```
## Warning in plot.window(xlim, ylim, log = log, ...): "data" is not a graphical
## parameter

## Warning in axis(if (horiz) 2 else 1, at = at.l, labels = names.arg, lty =
## axis.lty, : "data" is not a graphical parameter

## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...): "data"
## is not a graphical parameter

## Warning in axis(if (horiz) 1 else 2, cex.axis = cex.axis, ...): "data" is not a
## graphical parameter
```



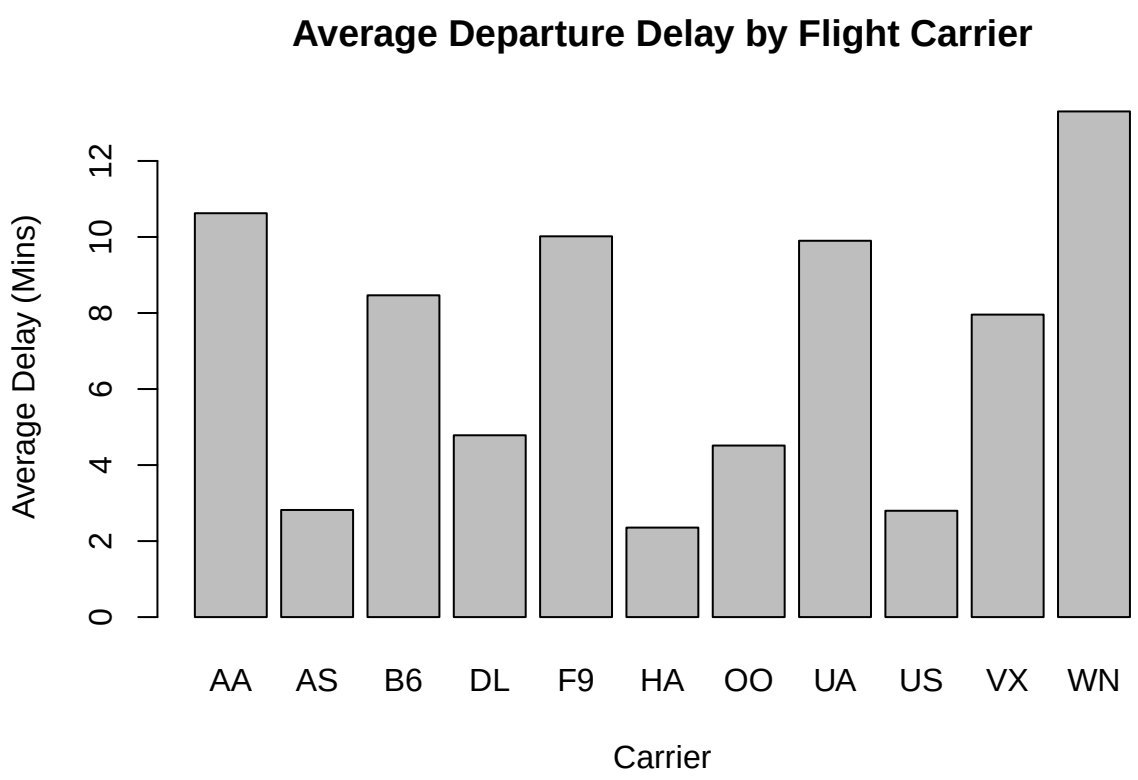

```
barplot(height = avg_delay$avg_dep_delay, names.arg = avg_delay$carrier, data = avg_delay, main = "Avera
```

```
## Warning in plot.window(xlim, ylim, log = log, ...): "data" is not a graphical
## parameter
```

```
## Warning in axis(if (horiz) 2 else 1, at = at.l, labels = names.arg, lty =
## axis.lty, : "data" is not a graphical parameter
```

```
## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...): "data"
## is not a graphical parameter
```

```
## Warning in axis(if (horiz) 1 else 2, cex.axis = cex.axis, ...): "data" is not a
## graphical parameter
```



As shown by the bar plot, certain airlines do tend to have longer cartain departure delays than others. In particular, F9 has the longest arrival delays (9 min), while US appears to leave 2 minutes early on average. For departure delays, WN has the highest average delay (12 mins) while HA has the lowest (3)

Are the arrival delays independent of the destination airport? How would you test this idea?

```
simple_delay_mod <- lm(arr_delay~as.factor(dest),data = flights)

flights_useful <- subset(flights, select = c(origin, dest, distance, air_time, arr_delay))

complex_mod <- lm(arr_delay ~. , data = flights_useful)

summary(simple_delay_mod)
```

```
##
## Call:
## lm(formula = arr_delay ~ as.factor(dest), data = flights)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
##  -60.71  -14.28   -5.86    4.77 1535.37
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)            3.55181    1.07344   3.309 0.000937 ***
## as.factor(dest)ANC    -6.58627    1.13964  -5.779 7.52e-09 ***
```

```
## as.factor(dest)ATL  -5.83037    1.18793  -4.908 9.21e-07 ***
## as.factor(dest)AUS  -6.03119    1.90190  -3.171 0.001519 **
## as.factor(dest)BLI   4.25052    3.50331   1.213 0.225022
## as.factor(dest)BNA  -6.64740    2.86087  -2.324 0.020151 *
## as.factor(dest)BOI  -8.88876    3.39822  -2.616 0.008905 **
## as.factor(dest)BOS  -3.15817    1.28488  -2.458 0.013975 *
## as.factor(dest)BUR  -5.17307    1.28288  -4.032 5.52e-05 ***
## as.factor(dest)BWI   2.50541    1.93862   1.292 0.196232
## as.factor(dest)BZN -14.55181   21.89406  -0.665 0.506278
## as.factor(dest)CLE  24.34008    5.19623   4.684 2.81e-06 ***
## as.factor(dest)CLT  -4.79423    1.42496  -3.364 0.000767 ***
## as.factor(dest)COS  -0.87313    2.00608  -0.435 0.663387
## as.factor(dest)CVG  -9.12223    2.80846  -3.248 0.001162 **
## as.factor(dest)DCA  -7.10826    1.45478  -4.886 1.03e-06 ***
## as.factor(dest)DEN   1.07409    1.12367   0.956 0.339139
## as.factor(dest)DFW   0.08162    1.15679   0.071 0.943748
## as.factor(dest)DTW  -4.77332    1.30703  -3.652 0.000260 ***
## as.factor(dest)EUG  -2.34970    1.68319  -1.396 0.162722
## as.factor(dest)EWR  -2.20509    1.27756  -1.726 0.084346 .
## as.factor(dest)FAI  -7.30093    1.41215  -5.170 2.34e-07 ***
## as.factor(dest)FAT  -0.72041    1.59449  -0.452 0.651404
## as.factor(dest)FLL  -3.40766    2.00608  -1.699 0.089383 .
## as.factor(dest)GEG  -0.65426    1.32416  -0.494 0.621242
## as.factor(dest)HDN   3.61486    5.74736   0.629 0.529376
## as.factor(dest)HNL  -5.14188    1.27197  -4.042 5.29e-05 ***
## as.factor(dest)HOU   6.86954    2.55446   2.689 0.007163 **
## as.factor(dest)IAD  -3.61831    1.38851  -2.606 0.009164 **
## as.factor(dest)IAH  -4.27511    1.21179  -3.528 0.000419 ***
## as.factor(dest)JAC  -1.01335    8.64414  -0.117 0.906679
## as.factor(dest)JFK   2.76719    1.22628   2.257 0.024036 *
## as.factor(dest)JNU  -0.75526    1.40597  -0.537 0.591147
## as.factor(dest)KOA  -8.57112    1.60417  -5.343 9.16e-08 ***
## as.factor(dest)KTN   0.17883    1.40547   0.127 0.898752
## as.factor(dest)LAS  -0.71420    1.13133  -0.631 0.527848
## as.factor(dest)LAX  -1.69137    1.11932  -1.511 0.130771
## as.factor(dest)LGB  -5.32555    1.24740  -4.269 1.96e-05 ***
## as.factor(dest)LIH  -9.84275    1.67883  -5.863 4.56e-09 ***
## as.factor(dest)LMT  -7.73226    2.88846  -2.677 0.007431 **
## as.factor(dest)MCI  -3.80181    1.55478  -2.445 0.014477 *
## as.factor(dest)MCO  -6.91674    1.84895  -3.741 0.000183 ***
## as.factor(dest)MDW   0.11421    1.33272   0.086 0.931709
## as.factor(dest)MIA  -0.95664    2.01040  -0.476 0.634184
## as.factor(dest)MKE  -2.73499    2.00608  -1.363 0.172773
## as.factor(dest)MSP  -5.08260    1.18537  -4.288 1.81e-05 ***
## as.factor(dest)MSY -21.01417    2.50882  -8.376  < 2e-16 ***
## as.factor(dest)OAK   6.50238    1.15790   5.616 1.96e-08 ***
## as.factor(dest)OGG  -5.90748    1.34404  -4.395 1.11e-05 ***
## as.factor(dest)OMA  -5.87121    2.00180  -2.933 0.003358 **
## as.factor(dest)ONT  -3.91958    1.30790  -2.997 0.002728 **
## as.factor(dest)ORD   1.23428    1.14156   1.081 0.279600
## as.factor(dest)PDX  -0.42595    1.27233  -0.335 0.737789
## as.factor(dest)PHL   1.42994    1.39498   1.025 0.305338
## as.factor(dest)PHX  -2.85692    1.12873  -2.531 0.011371 *
## as.factor(dest)PSP  -4.99830    1.50095  -3.330 0.000868 ***
```

```
## as.factor(dest)RDM  -1.32139    1.61017  -0.821 0.411846
## as.factor(dest)RNO   2.35295    2.09889   1.121 0.262269
## as.factor(dest)SAN  -3.57475    1.17119  -3.052 0.002272 **
## as.factor(dest)SAT -12.63240    2.00180  -6.311 2.79e-10 ***
## as.factor(dest)SBA  -2.72700    1.62329  -1.680 0.092974 .
## as.factor(dest)SEA   0.40039    1.27233   0.315 0.752998
## as.factor(dest)SFO   5.37882    1.11135   4.840 1.30e-06 ***
## as.factor(dest)SIT  -1.65293    3.44939  -0.479 0.631801
## as.factor(dest)SJC   1.28173    1.14855   1.116 0.264443
## as.factor(dest)SLC  -3.02799    1.15247  -2.627 0.008605 **
## as.factor(dest)SMF   3.81582    1.16944   3.263 0.001103 **
## as.factor(dest)SNA  -5.14999    1.20491  -4.274 1.92e-05 ***
## as.factor(dest)STL  -3.85403    1.81042  -2.129 0.033272 *
## as.factor(dest)TPA -25.68102    2.55446 -10.053  < 2e-16 ***
## as.factor(dest)TUS  -8.66928    1.61017  -5.384 7.29e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 30.93 on 146995 degrees of freedom
##   (1170 observations deleted due to missingness)
## Multiple R-squared:  0.01623,    Adjusted R-squared:  0.01576
## F-statistic: 34.65 on 70 and 146995 DF,  p-value: < 2.2e-16
```

```
summary(complex_mod)
```

```
##
## Call:
## lm(formula = arr_delay ~ ., data = flights_useful)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
##  -59.71  -13.97   -6.07    4.26 1537.44
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.604e+01  1.294e+00 -12.397  < 2e-16 ***
## originSEA    6.082e-01  1.861e-01   3.268 0.001082 **
## destANC     -1.358e+01  1.156e+00 -11.746  < 2e-16 ***
## destATL     -1.964e+01  1.333e+00 -14.734  < 2e-16 ***
## destAUS     -1.500e+01  1.926e+00  -7.791 6.70e-15 ***
## destBLI      1.968e+01  3.548e+00   5.547 2.91e-08 ***
## destBNA     -1.862e+01  2.889e+00  -6.446 1.15e-10 ***
## destBOI      3.977e+00  3.415e+00   1.165 0.244178
## destBOS     -2.265e+01  1.524e+00 -14.866  < 2e-16 ***
## destBUR     -1.614e+00  1.285e+00  -1.256 0.209184
## destBWI     -1.541e+01  2.067e+00  -7.456 8.96e-14 ***
## destBZN     -4.870e+00  2.177e+01  -0.224 0.823034
## destCLE      1.166e+01  5.194e+00   2.244 0.024816 *
## destCLT     -2.093e+01  1.576e+00 -13.280  < 2e-16 ***
## destCOS      2.653e-01  1.997e+00   0.133 0.894289
## destCVG     -2.039e+01  2.835e+00  -7.192 6.40e-13 ***
## destDCA     -2.408e+01  1.620e+00 -14.859  < 2e-16 ***
## destDEN      3.364e+00  1.121e+00   3.001 0.002690 **
## destDFW     -6.898e+00  1.190e+00  -5.799 6.68e-09 ***
```

```
## destDTW     -1.521e+01  1.379e+00 -11.032  < 2e-16 ***
## destEUG      1.340e+01  1.789e+00   7.490 6.91e-14 ***
## destEWR     -2.052e+01  1.486e+00 -13.803  < 2e-16 ***
## destFAI     -1.512e+01  1.429e+00 -10.576  < 2e-16 ***
## destFAT      4.826e+00  1.608e+00   3.002 0.002681 **
## destFLL     -2.716e+01  2.212e+00 -12.276  < 2e-16 ***
## destGEG      1.319e+01  1.446e+00   9.121  < 2e-16 ***
## destHDN      7.737e+00  5.718e+00   1.353 0.176060
## destHNL     -3.190e+01  1.618e+00 -19.722  < 2e-16 ***
## destHOU     -4.668e+00  2.583e+00  -1.807 0.070768 .
## destIAD     -1.983e+01  1.550e+00 -12.793  < 2e-16 ***
## destIAH     -1.446e+01  1.281e+00 -11.291  < 2e-16 ***
## destJAC      6.305e+00  8.602e+00   0.733 0.463586
## destJFK     -1.588e+01  1.454e+00 -10.928  < 2e-16 ***
## destJNU      9.371e-01  1.406e+00   0.666 0.505156
## destKOA     -3.659e+01  1.901e+00 -19.249  < 2e-16 ***
## destKTN      5.554e+00  1.428e+00   3.890 0.000100 ***
## destLAS      2.996e+00  1.140e+00   2.629 0.008558 **
## destLAX      6.949e-01  1.121e+00   0.620 0.535160
## destLGB     -3.076e+00  1.247e+00  -2.468 0.013599 *
## destLIH     -3.803e+01  1.965e+00 -19.351  < 2e-16 ***
## destLMT      4.641e+00  2.920e+00   1.589 0.112006
## destMCI     -8.173e+00  1.560e+00  -5.238 1.62e-07 ***
## destMCO     -2.754e+01  2.027e+00 -13.586  < 2e-16 ***
## destMDW     -9.395e+00  1.379e+00  -6.811 9.71e-12 ***
## destMIA     -2.325e+01  2.216e+00 -10.491  < 2e-16 ***
## destMKE     -1.036e+01  2.021e+00  -5.127 2.95e-07 ***
## destMSP     -8.156e+00  1.188e+00  -6.864 6.72e-12 ***
## destMSY     -3.492e+01  2.560e+00 -13.638  < 2e-16 ***
## destOAK      1.298e+01  1.191e+00  10.896  < 2e-16 ***
## destOGG     -3.225e+01  1.665e+00 -19.367  < 2e-16 ***
## destOMA     -8.291e+00  1.994e+00  -4.157 3.23e-05 ***
## destONT     -7.527e-01  1.308e+00  -0.575 0.565119
## destORD     -6.756e+00  1.189e+00  -5.681 1.34e-08 ***
## destPDX      1.404e+01  1.419e+00   9.892  < 2e-16 ***
## destPHL     -1.697e+01  1.583e+00 -10.716  < 2e-16 ***
## destPHX     -2.014e+00  1.123e+00  -1.793 0.072999 .
## destPSP     -3.077e+00  1.496e+00  -2.056 0.039796 *
## destRDM      1.418e+01  1.718e+00   8.253  < 2e-16 ***
## destRNO      1.242e+01  2.126e+00   5.844 5.12e-09 ***
## destSAN     -2.359e+00  1.167e+00  -2.021 0.043250 *
## destSAT     -2.178e+01  2.026e+00 -10.753  < 2e-16 ***
## destSBA      1.441e+00  1.624e+00   0.887 0.374838
## destSEA      1.508e+01  1.403e+00  10.754  < 2e-16 ***
## destSFO      1.160e+01  1.145e+00  10.138  < 2e-16 ***
## destSIT      1.294e+00  3.435e+00   0.377 0.706471
## destSJC      7.317e+00  1.178e+00   6.213 5.22e-10 ***
## destSLC      4.520e+00  1.185e+00   3.813 0.000137 ***
## destSMF      1.256e+01  1.219e+00  10.308  < 2e-16 ***
## destSNA     -3.635e+00  1.203e+00  -3.021 0.002521 **
## destSTL     -1.137e+01  1.831e+00  -6.210 5.30e-10 ***
## destTPA     -4.625e+01  2.678e+00 -17.269  < 2e-16 ***
## destTUS     -9.031e+00  1.602e+00  -5.639 1.72e-08 ***
## distance    -1.252e-03  4.868e-04  -2.572 0.010114 *
```

```
## air_time     1.458e-01  3.658e-03  39.849  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 30.75 on 146992 degrees of freedom
##   (1170 observations deleted due to missingness)
## Multiple R-squared:  0.02729,    Adjusted R-squared:  0.02681
## F-statistic: 56.49 on 73 and 146992 DF,  p-value: < 2.2e-16
```

To test weather or not arrival delay was independent of destination airport, I regressed it onto destination airport and also conducted a hypothesis test for slope (ANOVA). This is reported in the summary above. It appears that there is a statistically significant relationship between certain destination airports and arrival time, but not all airports.

I also made another model that included origin station, distance traveled, and airtime. Origin appeared to be a significant factor with a relatively large coefficient of 6.082, and airtime appeared to have a small effect (both statistically significant). Like the previous model, this model also has certain destination airports as having statistically significant coefficients, and some that do not.

Thus, we can be confident that arrival delays are not entirely independent of airport, but it also depends which airport the aircraft is arriving to, as we cannot be sure that the coefficient is not zero for all destination airports.

```
anova(simple_delay_mod)
```

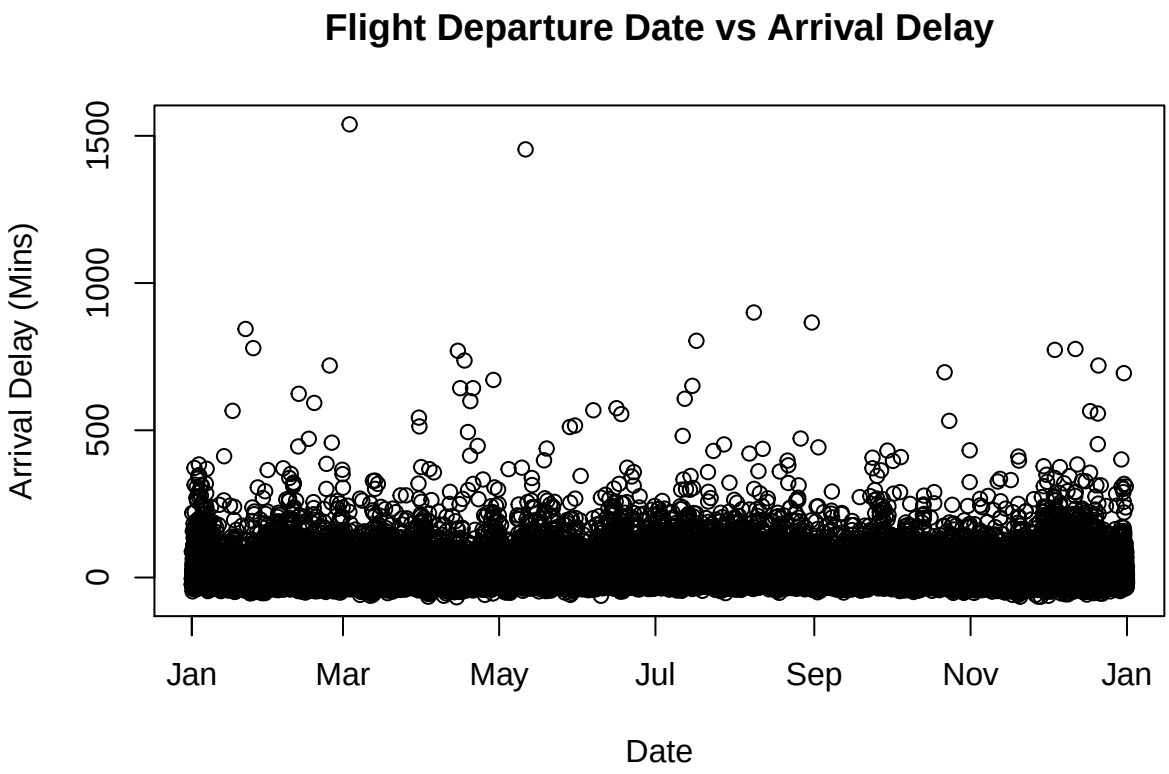
```
## Analysis of Variance Table
##
## Response: arr_delay
##                     Df    Sum Sq Mean Sq F value    Pr(>F)
## as.factor(dest)     70   2319746   33139   34.65 < 2.2e-16 ***
## Residuals       146995 140585299     956
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The p value of our anova test was less than 0.05. Thus, we can be sure that the slopes of at least some of the coefficients in our model are not zero, meaning there exists a relationship between the variables.

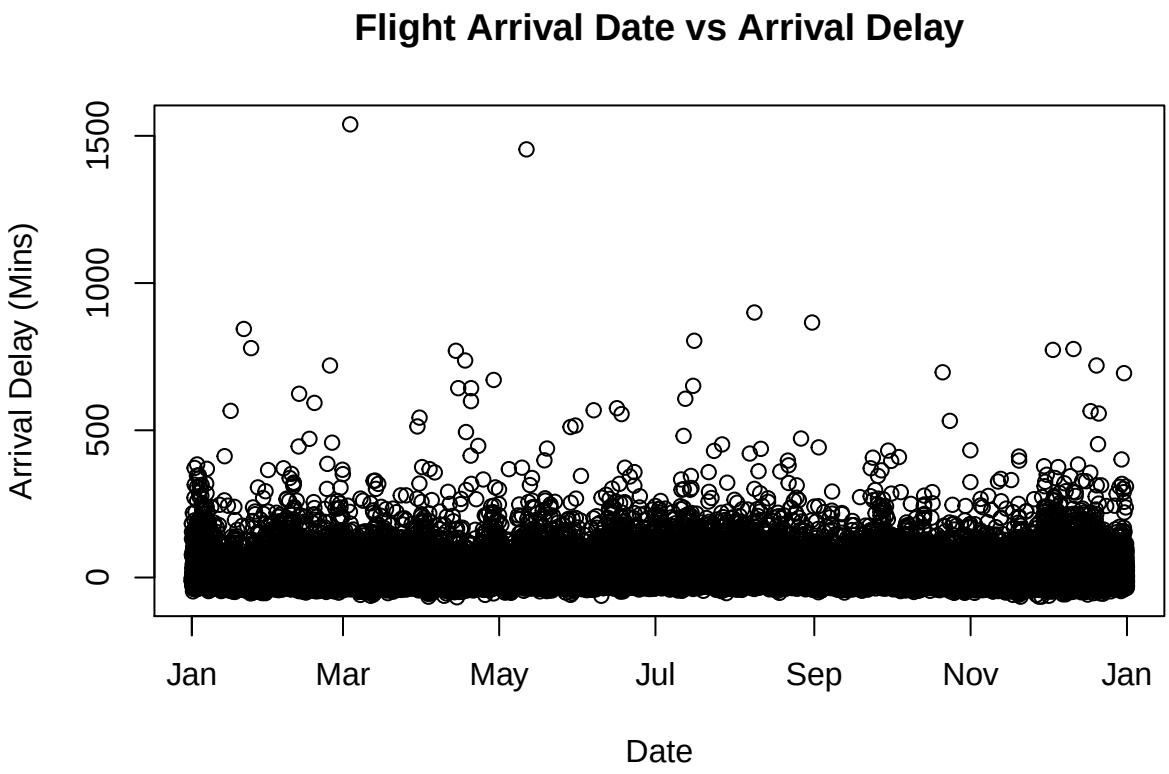
Is there a cyclical pattern to the arrival time? Please use a graph and a paragraph to justify your answer.

*Plot arrival time against arrival delay

```
plot(flights$dep_date, flights$arr_delay, xlab = "Date", ylab = "Arrival Delay (Mins)", main = "Flight
```




```
#Plot arrival delay vs arrival time
plot(flights$arr_date, flights$arr_delay, xlab = "Date", ylab = "Arrival Delay (Mins)", main = "Flight
```



The plot above shows a scatterplot of flight date vs arrival delay. Any patterns would be lose and relatively undefined, but there could be a slight increase in flight delays towards the begining of each month, with a decrease in delay as the month goes on. This explains the sort of peaks and then dips in the graph. These patterns are slight though. Mostly, it appears that arrival times are constant over time.

Are arrival delays the same across airlines (i.e. carriers)?

```
summary(aov(arr_delay ~ carrier, data = flights))
```

```
##                 Df    Sum Sq Mean Sq F value Pr(>F)
## carrier         10   1309621  130962     136 <2e-16 ***
## Residuals   147055 141595425     963
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 1170 observations deleted due to missingness
```

To determine weather there was a real difference in means between groups, I ran a one-way anova test, which provided an F statistic of 136 and a p value of almost 0. Since the p value is less than 0.05, we can confidently say with statistical significance that there is a difference in the mean arrival delay between carriers in this dataset.

2

I explained above the data quality to issues. To fix this, I will assume the negative distance, airtime and departure time values were entered in error, and should be positive. Fixing missing data and NA values is not possible:

```r
flights$distance <- abs(flights$distance)
flights$air_time <- abs(flights$air_time)
flights$dep_time <- abs(flights$air_time)
```