ReachNow Seattle Temporal Analysis

CP

December 22, 2017

R Markdown

```
library(readxl)
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
```

```
library(maps)
library(mapproj)

# Read the trip data from the excel file
# Total ReachNow i3 data for USA
RN_US_data <- read_excel("../i3_TRIP_DATA_REACHNOW_USA.xlsx", col_types = NULL)

# Read the map boundaries for Seattle
maps_data <-
  read.csv("../ReachNow_boundaries_Seattle_total.txt",
           header = TRUE,
           sep = "\t")

# Find the Seattle lat, long extremes
min_lat <- min(maps_data['latitude'])
min_long <- min(maps_data['longitude'])
max_lat = max(maps_data['latitude'])
max_long = max(maps_data['longitude'])

# Extract Seattle trips from USA trips
# Remove NAs
RN_US_data_nonNull <-
  subset(RN_US_data, !RN_US_data$StartLocationLat == 'NA')
# Remove illegal trips and trips from other regions  !! Longitudes are working in reverse (?)
RN_Sea_data <-
  subset(
    RN_US_data_nonNull,
    StartLocationLat >= min_lat &
      EndLocationLat <= max_lat &
      StartLocationLng <= min_long & EndLocationLng >= max_long
  )
```

```r
#Get Member trips
RN_Sea_data_member <- filter(RN_Sea_data, Type == 'M')

# Filter trips with non-zero miles travelled
RN_Sea_data_member_miles <-
  filter(RN_Sea_data_member, MileageDriven > 0)

# Get start and end points
startPoints <-
  data.frame(lat = RN_Sea_data_member_miles$StartLocationLat,
             lon = RN_Sea_data_member_miles$StartLocationLng)
endPoints <-
  data.frame(lat = RN_Sea_data_member_miles$EndLocationLat,
             lon = RN_Sea_data_member_miles$EndLocationLng)
# Assimilate the trips and save to a file
RN_Sea_trips <-
  data.frame(startPoints$lat,
             startPoints$lon,
             endPoints$lat,
             endPoints$lon)
write.table(
  RN_Sea_trips,
  "RN_Sea_trips.txt",
  sep = "\t",
  row.names = FALSE,
  quote = FALSE
)

# Temporal Analysis
# Find the variation in days
library(lubridate)
```

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

## The following object is masked from 'package:base':
##
##     date
```

```r
attach(RN_Sea_data_member_miles, warn.conflicts = FALSE)
# Parse the ReservationStartTime as datetime after sorting
df_startTime <-
  data.frame(parse_date_time(sort(ReservationStartTime),
                  c("Ymd HMS", "Ymd HM"),
                  tz = "America/Los_Angeles"))

ResStartTime = parse_date_time(sort(ReservationStartTime),
                               c("Ymd HMS", "Ymd HM"),
                               tz = "America/Los_Angeles")
i <- 1
date_counter <-
  data.frame(ReservationDate = as.Date(character()), Count = integer())
day_counter <- 1
day_frequency <- 1
total_rows <- nrow(RN_Sea_data_member_miles)
```

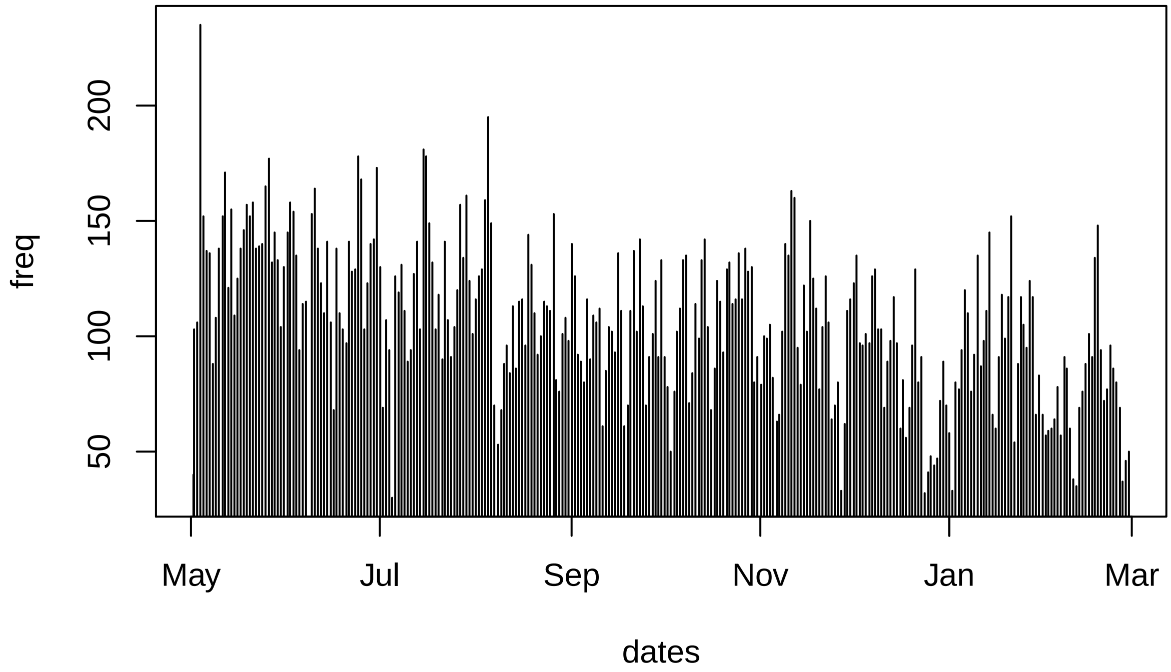
```
dates = structure(integer(), class = "POSIXct")
freq = c()

  while (i < total_rows) {
    day_frequency <- 1
    nexti <- i + 1
    while (year(ResStartTime[i]) == year(ResStartTime[nexti]) &&
           month(ResStartTime[i]) == month(ResStartTime[nexti]) &&
           day(ResStartTime[i]) == day(ResStartTime[nexti])) {
      # print("date at i: ")
      # print(ResStartTime[i] )
      # print("date at nexti: ")
      # print(ResStartTime[nexti] )
      day_frequency <- day_frequency + 1


      if (nexti == total_rows) {
        break
      }
      nexti <- nexti + 1

    }
    dates = c(dates, ResStartTime[i])
    freq = c(freq, day_frequency)
    i <- nexti
    day_counter <- day_counter + 1
  }

date_counter <- data.frame(dates, freq)
plot(date_counter, type="h")
```


freq
50
100
150
200
May
Jul
Sep
Nov
Jan
Mar
dates