STAT30340: Data Programming with R

Assignment 2

Anthony Doyle (19758695)

1 Task: Manipulation

1.1

```
setwd("/Users/tonydoyle/Documents/UCD/Year6/STAT30340.DataProgrammingR")

library(rio)
library(tidyverse)
library(ggplot2)
library(lubridate)
library(skimr)
library(knitr)

pedestrian_2023 <- import("pedestrian_2023.csv", setclass = "tibble") |>
  select(-ends_with("IN"), -ends_with("OUT"))
# anyNA(pedestrian_2023)
dim(pedestrian_2023)
```

```
[1] 8760   27
```

The data set 'pedestrian_2023' contains hourly data concerning pedestrian traffic in Dublin in 2023 (from January 1st 2023 at 12am to December 31st 2023 at 11pm). It comprises 8760 observations across 27 variables. All variables that ends with "IN" or OUT" are removed, and the data set is specified as a tibble for improved data handling. Note, there are missing observations in the data set.

1.2

```
# sapply(pedestrian_2023, class) # all data classes
class(pedestrian_2023$Time)
```

```
[1] "character"
```

```r
pedestrian_2023 <- pedestrian_2023 |>
  mutate(Time = dmy_hm(Time)) # dmy_hm() (day-month-year-hours-and-minutes)

pedestrian_2023[, -1] <- lapply(pedestrian_2023[, -1], as.numeric)

# sapply(pedestrian_2023, class)
class(pedestrian_2023$Time)
```

```
[1] "POSIXct" "POSIXt"
```

We examine the data classes of all variables in the data set ‘pedestrian_2023’. “Time” is stored as a “character”, while all other variables are stored as “numeric” or “integer”. The `lubridate()` package is used to convert “Time” to an appropriate class (“POSIXct” “POSIXt”), whereas all other variables are converted to “numeric”.

1.3

```r
weather_2023 <- import("weather_2023.txt", setclass = "tibble")
# colnames(weather_2023)
# anyNA(weather_2023)

# Renaming weather variables (column names)
weather_2023 <- weather_2023 |>
  rename(`Precipatation(mm)` = rain,
         `Temperature(C)` = temp,
         `Wind Speed(kt)` = wdsp,
         `Cloud Cover(0-9)` = clamt)
colnames(weather_2023)
```

```
[1] "Time"              "Precipatation(mm)" "Temperature(C)"
[4] "Wind Speed(kt)"    "Cloud Cover(0-9)"
```

```r
dim(weather_2023)
```

```
[1] 8760    5
```

The data set ‘weather_2023’ contains hourly data concerning weather conditions in Dublin in 2023 (from January 1st 2023 at 12am to December 31st 2023 at 11pm). It comprises 8760 observations across 5 variables. The data set is also specified as a tibble for improved data handling, and variables are renamed for improved understanding.

1.4

```r
weather_2023 <- weather_2023 |>
  mutate(`Cloud Cover(0-9)` = factor(`Cloud Cover(0-9)`, ordered = TRUE))

print(levels(weather_2023$`Cloud Cover(0-9)`))
```

```
[1] "0" "1" "2" "3" "4" "5" "6" "7" "8"
```

```r
is.ordered(weather_2023$`Cloud Cover(0-9)`)
```

```
[1] TRUE
```

The variable "Cloud Cover(0-9)" in the 'weather_2023' data set in converted to an ordered factor. The range of possible categorical values are cloud clover (okta) are: 0 oktas represents the complete absence of cloud; 1 okta represents a cloud amount of 1 eighth or less, but not zero, etc.; 7 oktas represents a cloud amount of 7 eighths or more, but not full cloud cover; 8 oktas represents full cloud cover with no breaks; and 9 oktas represents sky obscured by fog or other meteorological phenomena.

1.5

```r
skim_without_charts(weather_2023)
```

Table 1: Data summary

Name	weather_2023
Number of rows	8760
Number of columns	5
Column type frequency:	
factor	1
numeric	3
POSIXct	1
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
Cloud Cover(0-9)	0	1	TRUE	9	7: 3865, 6: 1338, 8: 733, 1: 727

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
Precipatation(mm)	0	1	0.11	0.50	0.0	0.0	0.0	0.0	10.6
Temperature(C)	0	1	10.74	4.85	-4.3	7.4	10.7	14.3	25.5
Wind Speed(kt)	0	1	8.99	4.21	0.0	6.0	9.0	11.0	31.0

Variable type: POSIXct

skim_variable	n_missing	complete_rate	min	max	median	n_unique
Time	0	1	2023-01-01	2023-12-31 23:00:00	2023-07-02 11:30:00	8760

The function `skim_without_charts` from the `skimr()` package provides a comprehensive summary of the data set 'weather_2023'. This includes the data set dimensions (8760 rows; 5 columns); the type of variables and their classes; and statistics (missing values, min, max, median, ordered, top counts, etc.) by variable type/class.

1.6

```
class(weather_2023$Time)
```

```
[1] "POSIXct" "POSIXt"
```

```
range(weather_2023$Time)
```

```
[1] "2023-01-01 00:00:00 UTC" "2023-12-31 23:00:00 UTC"
```

```
range(pedestrian_2023$Time)
```

```
[1] "2023-01-01 00:00:00 UTC" "2023-12-31 23:00:00 UTC"
```

The "Time" variable in 'weather_2023' is stored an appropriate class ("POSIXct" "POSIXt"), and is the same format to the "Time" variable in the data set 'pedestrian_2023'. Moreover, both data sets share the same timeline: "2023-01-01 00:00:00 UTC", "2023-12-31 23:00:00 UTC".

1.7

```
pedestrian_weather_2023 <- merge(pedestrian_2023, weather_2023)
pedestrian_weather_2023 <- as_tibble(pedestrian_weather_2023)
dim(pedestrian_weather_2023)
```

```
[1] 8760   31
```

The merged data set 'pedestrian_weather_2023' contains hourly data concerning pedestrian traffic and weather conditions in Dublin in 2023 (from January 1st 2023 at 12am to December 31st 2023 at 11pm). It comprises 8760 observations across 31 variables. The data set is specified as a tibble for improved data handling.

1.8

```
pedestrian_weather_2023$`Day` <- wday(pedestrian_weather_2023$Time,
                                      label = TRUE)
pedestrian_weather_2023$`Month` <- month(pedestrian_weather_2023$Time,
                                         label = TRUE, abbr = TRUE)

print(levels(pedestrian_weather_2023$Day))
```

```
[1] "Sun" "Mon" "Tue" "Wed" "Thu" "Fri" "Sat"
```

```
is.ordered(pedestrian_weather_2023$Day)
```

```
[1] TRUE
```

```
print(levels(pedestrian_weather_2023$Month))
```

```
 [1] "Jan" "Feb" "Mar" "Apr" "May" "Jun" "Jul" "Aug" "Sep" "Oct" "Nov" "Dec"
```

```
is.ordered(pedestrian_weather_2023$Month)
```

```
[1] TRUE
```

The functions `wday()` and `month()` from the `lubridate()` package are used to extract the day of the week and month from the variable "Time" in the data set 'pedestrian_weather_2023'. Two new variables "Day" and "Month" are created, and both variables are ordered factors.

1.9

```r
pedestrian_weather_2023<- pedestrian_weather_2023 |>
  relocate(Month, Day, .after = Time)
print(colnames(pedestrian_weather_2023))
```

```
 [1] "Time"
 [2] "Month"
 [3] "Day"
 [4] "Aston Quay/Fitzgeralds"
 [5] "Baggot st lower/Wilton tce inbound"
 [6] "Baggot st upper/Mespil rd/Bank"
 [7] "Capel st/Mary street"
 [8] "College Green/Bank Of Ireland"
 [9] "College Green/Church Lane"
[10] "College st/Westmoreland st"
[11] "D'olier st/Burgh Quay"
[12] "Dame Street/Londis"
[13] "Grafton st/Monsoon"
[14] "Grafton Street / Nassau Street / Suffolk Street"
[15] "Grafton Street/CompuB"
[16] "Grand Canal st upp/Clanwilliam place"
[17] "Grand Canal st upp/Clanwilliam place/Google"
[18] "Henry Street/Coles Lane/Dunnes"
[19] "Mary st/Jervis st"
[20] "North Wall Quay/Samuel Beckett bridge East"
[21] "North Wall Quay/Samuel Beckett bridge West"
[22] "O'Connell St/Parnell St/AIB"
[23] "O'Connell st/Princes st North"
[24] "Phibsborough Rd/Enniskerry Road"
[25] "Richmond st south/Portabello Harbour inbound"
[26] "Richmond st south/Portabello Harbour outbound"
[27] "Talbot st/Guineys"
[28] "Westmoreland Street East/Fleet street"
[29] "Westmoreland Street West/Carrolls"
[30] "Precipatation(mm)"
[31] "Temperature(C)"
[32] "Wind Speed(kt)"
[33] "Cloud Cover(0-9)"
```

The function `relocate()` from the `dplyr()` package is used to move the newly created variables, “Month” and “Day”, to appear after the “Time” variable in the ‘pedestrian_weather_2023’ data set.

2 Task: Analysis

2.1

```
pedestrian_weather_2023$`Pedestrian Traffic` <- rowSums(pedestrian_weather_2023[, 4:29],
                                                        na.rm = TRUE)

pedestrian_traffic_2023 <- aggregate(pedestrian_weather_2023$`Pedestrian Traffic`,
                                     list(Month = pedestrian_weather_2023$`Month`),
                                     sum, na.rm = TRUE)

kable(pedestrian_traffic_2023,
      caption = "Total Pedestrian Traffic by Month in Dublin 2023",
      col.names = c("Month", "Pedestrian Traffic"))
```

Table 5: Total Pedestrian Traffic by Month in Dublin 2023

Month	Pedestrian Traffic
Jan	23393404
Feb	22663558
Mar	25002430
Apr	24133550
May	19151872
Jun	15128010
Jul	15348281
Aug	16181211
Sep	20912351
Oct	21664557
Nov	22315247
Dec	19829327

```
print(pedestrian_traffic_2023[which.max(pedestrian_traffic_2023$x), ])
```

```
  Month        x
3   Mar 25002430
```

```
print(pedestrian_traffic_2023[which.min(pedestrian_traffic_2023$x), ])
```

```
  Month        x
6   Jun 15128010
```

The function `rowSums()` is used to calculate the sum of each row of the columns 4-29 (pedestrian footfall figures by street name). (Note "NA" values are ignored.) Then, using `aggregate()` function, the data is grouped by "Month" and a total pedestrian traffic count is computed for each month using the sum argument. Missing values (na.rm = TRUE) are excluded in the summation, which is a requirement of the 'aggregate()' function. Table 5 displays the results. March (25002430) had the highest total pedestrian traffic count, while June (15128010) had the lowest.

2.2

```
# colSums(is.na(pedestrian_weather_2023))

daily_pedestrian_2023 <- pedestrian_weather_2023 |>
  mutate(Date = as.Date(Time)) |>
  group_by(Date) |>
  summarise(
    `O'Connell st/Princes st North` = sum(`O'Connell st/Princes st North`,
                                          na.rm = TRUE),
    `Grafton st/Monsoon` = sum(`Grafton st/Monsoon`,
                               na.rm = TRUE),
    `Mary st/Jervis st` = sum(`Mary st/Jervis st`,
                              na.rm = TRUE)
  )

dp_long <- daily_pedestrian_2023 |>
  pivot_longer(cols = c(`O'Connell st/Princes st North`,
                        `Grafton st/Monsoon`,
                        `Mary st/Jervis st`),
               names_to = "Location",
               values_to = "DailyPed")
```

First, we examine which three locations to select. Streets with high levels of missing values are discarded for this task. We decide to analyse: O'Connell st/Princes st North; Grafton st/Monsoon; and Mary st/Jervis st. A modified data set 'daily_pedestrian_2023' is created, which mutates the "Time" variable to a date format, groups the data by "Date", and calculates/aggregates a daily tally for pedestrian traffic in the three selected locations. Then, the data set is converted from a wide to a long format (for ease of analysis and plotting), which comprises three variables: "Date"; "Location" and "Pedestrian Traffic".

```
figure1 <- ggplot(dp_long, aes(x = Date, y = DailyPed, color = Location)) +
  geom_line(size = .75) +
  labs(x = NULL,
       y = "Pedestrian Traffic",
       color = "Location") +
  geom_vline(xintercept = as.Date(c("2023-03-17", "2023-12-24")),
             color = "black", size = 0.5) +
  theme_bw() +
  scale_color_manual(values = c(`O'Connell st/Princes st North` = "#377EB8",
                                `Grafton st/Monsoon` = "#E41A1C",
                                `Mary st/Jervis st` = "#4DAF4A")) +
  geom_text(data = data.frame(Date = as.Date(c("2023-03-17", "2023-12-24")),
```

```r
                               label = c("St Patrick's day", "Christmas day")),
            aes(x = Date, y = -Inf, label = label), vjust = -0.5,
            color = "black", size = 2, family = "serif") +
  theme(
    legend.title = element_text(size = 9),
    legend.text = element_text(size = 8),
    legend.key.size = unit(0.75, "lines")
  ) +
  theme(axis.title = element_text(family = "serif"),
        axis.text = element_text(family = "serif"),
        legend.text = element_text(family = "serif"),
        legend.title = element_text(family = "serif"),
        plot.title = element_text(family = "serif"))
figure1
```

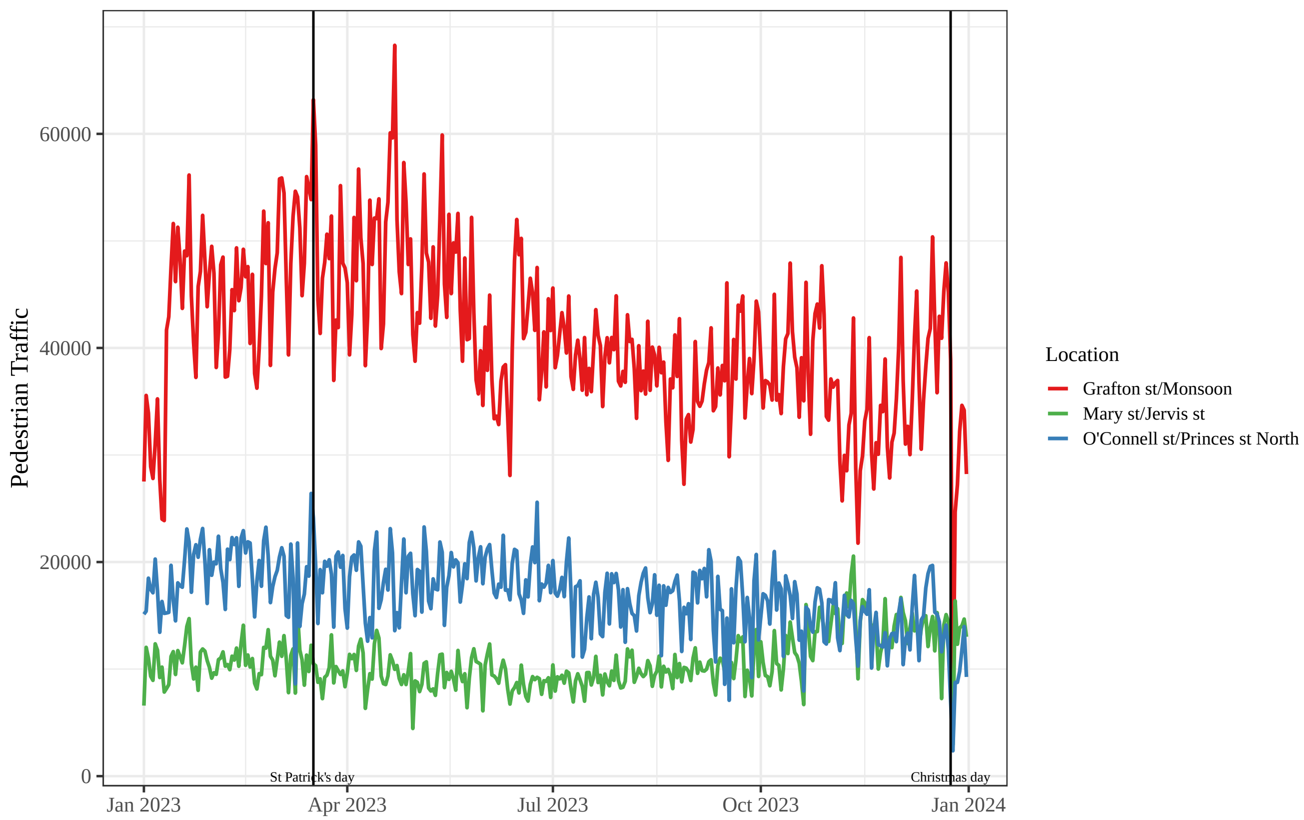

Figure 1: Daily pedestrian traffic on O'Connell st/Princes st North, Grafton st/Monsoon and Mary st/Jervis st in 2023

Figure 1 displays a time series plot of the daily footfall on O'Connell st/Princes st North, Grafton st/Monsoon and Mary st/Jervis st in 2023. Two vertical bars notably indicate the days: St. Patrick's day (2023-03-17) and Christmas day (2023-12-24). On St Patrick's day, daily pedestrian traffic, in the three locations, was above-trend, whereas on Christmas Day it was below trend.

2.3

```r
#|label: 2.3-weather_2023-table

weather_2023$Time <- as.POSIXct(weather_2023$Time)
weather_2023$Month <- as.numeric(format(weather_2023$Time, "%m"))

# Define seasons
weather_2023$Season <- case_when(
  weather_2023$Month %in% c(12, 1, 2) ~ "Winter",
  weather_2023$Month %in% c(3, 4, 5) ~ "Spring",
  weather_2023$Month %in% c(6, 7, 8) ~ "Summer",
  weather_2023$Month %in% c(9, 10, 11) ~ "Autumn"
)

weather_table <- weather_2023 |>
  group_by(Season) |>
  summarise(
    `Min Temperature (C)` = min(`Temperature(C)`, na.rm = TRUE),
    `Max Temperature (C)` = max(`Temperature(C)`, na.rm = TRUE),
    `Mean Precipitation (mm)` = mean(`Precipatation(mm)`, na.rm = TRUE),
    `Mean Wind Speed (kt)` = mean(`Wind Speed(kt)`, na.rm = TRUE)
  )

kable(weather_table,
      caption = "Summary Statistics for seasonal weather in Dublin 2023",
      digits = 1)
```

Table 6: Summary Statistics for seasonal weather in Dublin 2023

Season	Min Temperature (C)	Max Temperature (C)	Mean Precipitation (mm)	Mean Wind Speed (kt)
Autumn	-2.0	25.5	0.1	8.3
Spring	-4.3	19.4	0.1	8.6
Summer	3.8	24.0	0.1	8.5
Winter	-4.3	13.8	0.1	10.5

To prepare the 'weather_2023' data set for the summary statistics table, we first convert the "Time" variable to a date format using the `as.POSIXct()` from the `lubridate()` package. A new variable, "Month", is created by extracting the month from the "Time" variable, and reclassifying it as numeric data (e.g. "2023-01-02 19:00:00" = 1 (January)). We then use the `case_when` function from the `dplyr()` package to assign the "Month" observations to their newly created, respective "Season" variable. Then, the `aggregate()` function is used to calculate the relevant summary statistics.

3 Task: Creativity

3.1

```
#|label: 3.1-average-pedestrian-traffic-by-month

daily_pedestrian_2023 <- daily_pedestrian_2023 |>
  mutate(
    Date = as.POSIXct(Date),
    Month = month(Date, label = TRUE, abbr = FALSE)
  )

monthly_pedestrian_2023 <- daily_pedestrian_2023 |>
  group_by(Month) |>
  summarise(
    `O'Connell st/Princes st North` = sum(`O'Connell st/Princes st North`,
                                          na.rm = TRUE),
    `Grafton st/Monsoon` = sum(`Grafton st/Monsoon`,
                               na.rm = TRUE),
    `Mary st/Jervis st` = sum(`Mary st/Jervis st`,
                              na.rm = TRUE)
  )

kable(monthly_pedestrian_2023,
      caption = "Total Pedestrian Traffic by Month in 2023")
```

Table 7: Total Pedestrian Traffic by Month in 2023

Month	O'Connell st/Princes st North	Grafton st/Monsoon	Mary st/Jervis st
January	567279	1279120	325759
February	558074	1238423	301877
March	578200	1544483	321393
April	541939	1492868	301357
May	588593	1408979	289575
June	565108	1219839	269676
July	514147	1230153	280212
August	502275	1145409	303109
September	483269	1142621	312219
October	485983	1206301	362421
November	421455	970758	434042
December	413017	1138977	414543

We use the `lubridate()` function to extract the month name from each date in the data set 'daily_pedestrian_2023'. In a similar process to the calculation of the summary statistics for figure Table 5, we now calculate the monthly mean average pedestrian traffic for our three selected streets of interest (O'Connell st/Princes st North, Grafton st/Monsoon and Mary st/Jervis st).

Table 6 displays that the total pedestrian traffic was highest in May for O'Connell st/Princes st North, in March for Grafton st/Monsoon, and November for Mary st/Jervis st. Conversely, total pedestrian traffic was lowest in November for O'Connell st/Princes st North and Grafton st/Monsoon, and June for Mary st/Jervis st.

3.2

```
#|label: 3.2-precipatation-pedestrian-traffic-relationship

pedestrian_precipitation_2023 <- pedestrian_weather_2023 |>
  mutate(Date = as.Date(Time)) |>
  group_by(Date) |>
  summarise(
    `O'Connell st/Princes st North` = sum(`O'Connell st/Princes st North`,
                                          na.rm = TRUE),
    `Grafton st/Monsoon` = sum(`Grafton st/Monsoon`,
                               na.rm = TRUE),
    `Mary st/Jervis st` = sum(`Mary st/Jervis st`,
                              na.rm = TRUE),
    `Precipitation(mm)` = sum(`Precipatation(mm)`,
                              na.rm = TRUE)
  )
```

Lastly, we want to examine the relationship between precipitation (mm) and pedestrian traffic in our three locations of interest. The `summarise()` function is used to calculate total rainfall and pedestrian traffic for each day in the respective locations. We then reshape the data from wide to long format, which makes it easier to create the scatter plot.

```
pp_long <- pedestrian_precipitation_2023 |>
  pivot_longer(
    cols = c("O'Connell st/Princes st North",
             "Grafton st/Monsoon",
             "Mary st/Jervis st"),
    names_to = "Street",
    values_to = "Traffic_C"
  )

ggplot(pp_long, aes(x = `Precipitation(mm)`, y = Traffic_C, color = Street)) +
  geom_point() +
  labs(x = "Precipitation (mm)",
       y = "Pedestrian Traffic Count"
  ) +
    geom_smooth(method = "lm", se = FALSE) +
  theme_bw() +
  theme(
    legend.title = element_text(size = 9),
```

```
    legend.text = element_text(size = 8),
    legend.key.size = unit(0.75, "lines")
  ) +
  theme(
    axis.title = element_text(family = "serif"),
    axis.text = element_text(family = "serif"),
    legend.text = element_text(family = "serif"),
    legend.title = element_text(family = "serif"),
    plot.title = element_text(family = "serif")
  )
```

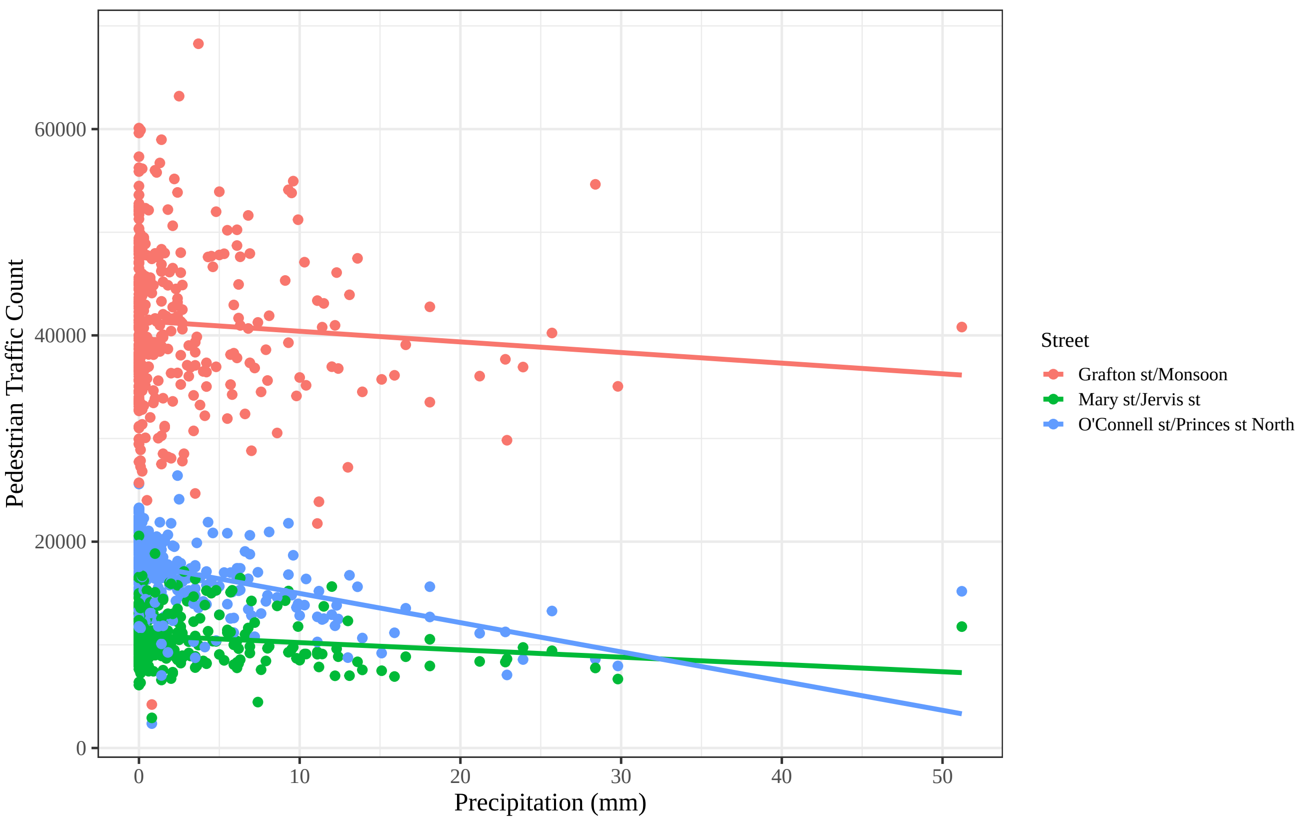

Figure 2: The relationship between total daily rainfall and pedestrian traffic

Figure 5 displays the relationship between total daily rainfall and pedestrian traffic. The scattered data is heavily clustered at lower levels of rainfall. As a result, it is harder to observe whether total daily rainfall has a significant effect on daily pedestrian traffic in the three selected locations of interest. (This is indicated by the relatively flat, downward sloping linear regression lines). Moreover, in the case where total daily rainfall was exceptionally high (i.e., above 50mm - see outliers), there was not a notably rapid decrease in pedestrian traffic for that day. (Maybe the Irish have just become used to the rain!)