Module 5: Reproducible Research - Course Project 1

Roddy Mendoza Marriott

2025-06-09

Introduction

It is now possible to collect a large amount of data about personal movement using activity monitoring devices such as a Fitbit, Nike Fuelband, or Jawbone Up. These type of devices are part of the "quantified self" movement – a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks. But these data remain under-utilized both because the raw data are hard to obtain and there is a lack of statistical methods and software for processing and interpreting the data.

This assignment makes use of data from a personal activity monitoring device. This device collects data at 5 minute intervals through out the day. The data consists of two months of data from an anonymous individual collected during the months of October and November, 2012 and include the number of steps taken in 5 minute intervals each day.

The data for this assignment can be downloaded from the course web site:

The dataset

- Dataset: Activity monitoring data [52K]

The variables included in this dataset are:

- **steps**: Number of steps taking in a 5-minute interval (missing values are coded as *NA*)
- **date**: The date on which the measurement was taken in YYYY-MM-DD format
- **interval**: Identifier for the 5-minute interval in which measurement was taken The dataset is stored in a comma-separated-value (CSV) file and there are a total of 17,568 observations in this dataset.

Loading the required packages

```
library(data.table)
```

```
## Warning: package 'data.table' was built under R version 4.3.3
```

```
library(ggplot2)
library(scales)
```

```
## Warning: package 'scales' was built under R version 4.3.3
```

```
library(lubridate)
```

```
## Warning: package 'lubridate' was built under R version 4.3.3
```

Loading in Data

Downloading and unzipping data to obtain the csv file.

```
url <- "https://d396qusza40orc.cloudfront.net/repdata%2Fdata%2Factivity.zip"
download.file(url, destfile = "Factivity.zip")
unzip("Factivity.zip")
```

Reading csv into a data.table.

```
activity_data <-fread("activity.csv")
```

What is mean total number of steps taken per day?

1. Creating a new variable called total_steps

```
total_steps <- aggregate(steps ~ date, data = activity_data, sum)
```

2. Calculate the total number of steps taken per day

```
mean(total_steps$steps, na.rm = T)
```

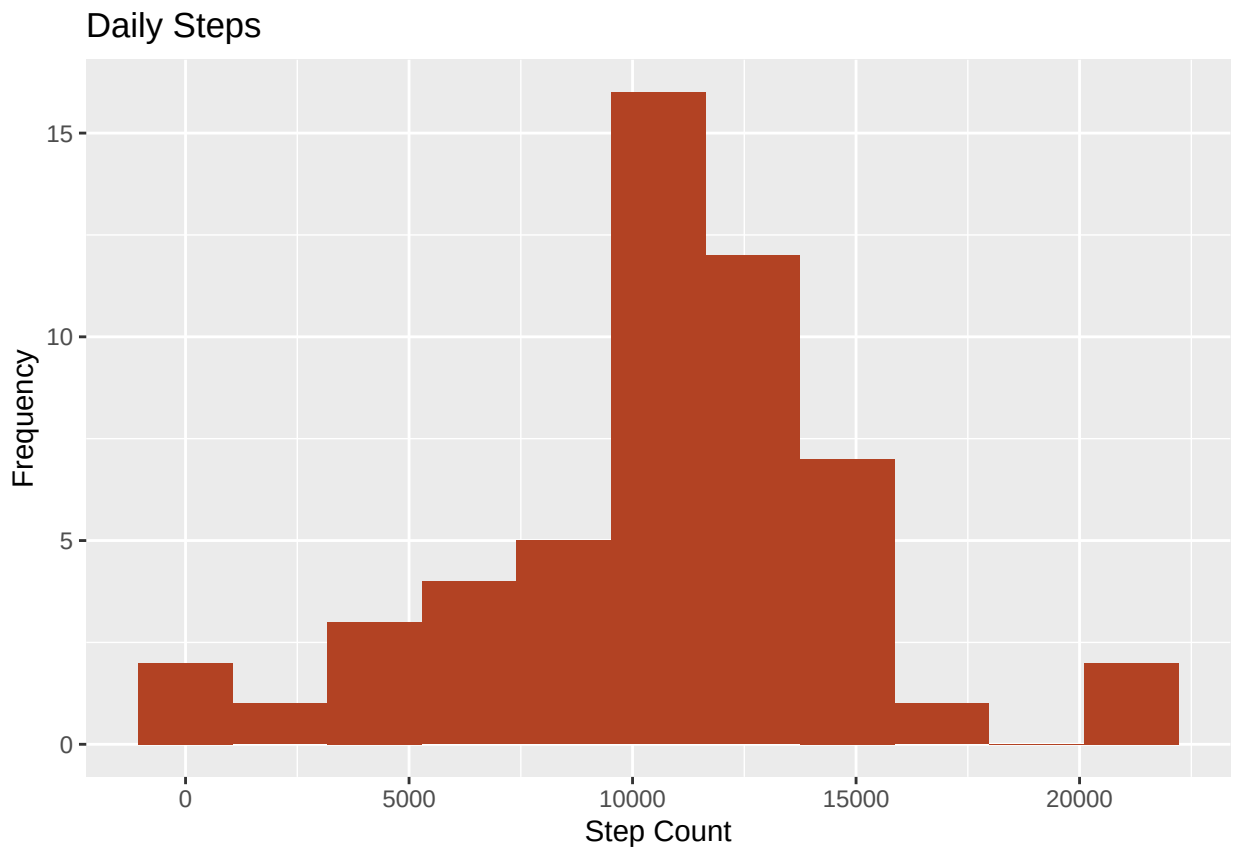
```
## [1] 10766.19
```

```
median(total_steps$steps, na.rm = T)
```

```
## [1] 10765
```

3. Making a histogram of the total number of steps taken per day

```
activity_data$date <- as.Date(activity_data$date)
total_steps <- aggregate(steps ~ date, data = activity_data, sum)
plot1 <- ggplot(total_steps, aes(x=steps)) +
  geom_histogram(fill = "#B24223", bins = 11) +
  labs(title = "Daily Steps", x = "Step Count", y = "Frequency")
print(plot1)
```



4. Calculating the mean and median of the total number of steps taken per day

```
mean(total_steps$steps, na.rm = T)
```

```
## [1] 10766.19
```

```
median(total_steps$steps, na.rm = T)
```

```
## [1] 10765
```

What is the average daily activity pattern?

1. Making a time series

```
interval_steps <- aggregate(steps ~ interval, data = activity_data, mean)
```

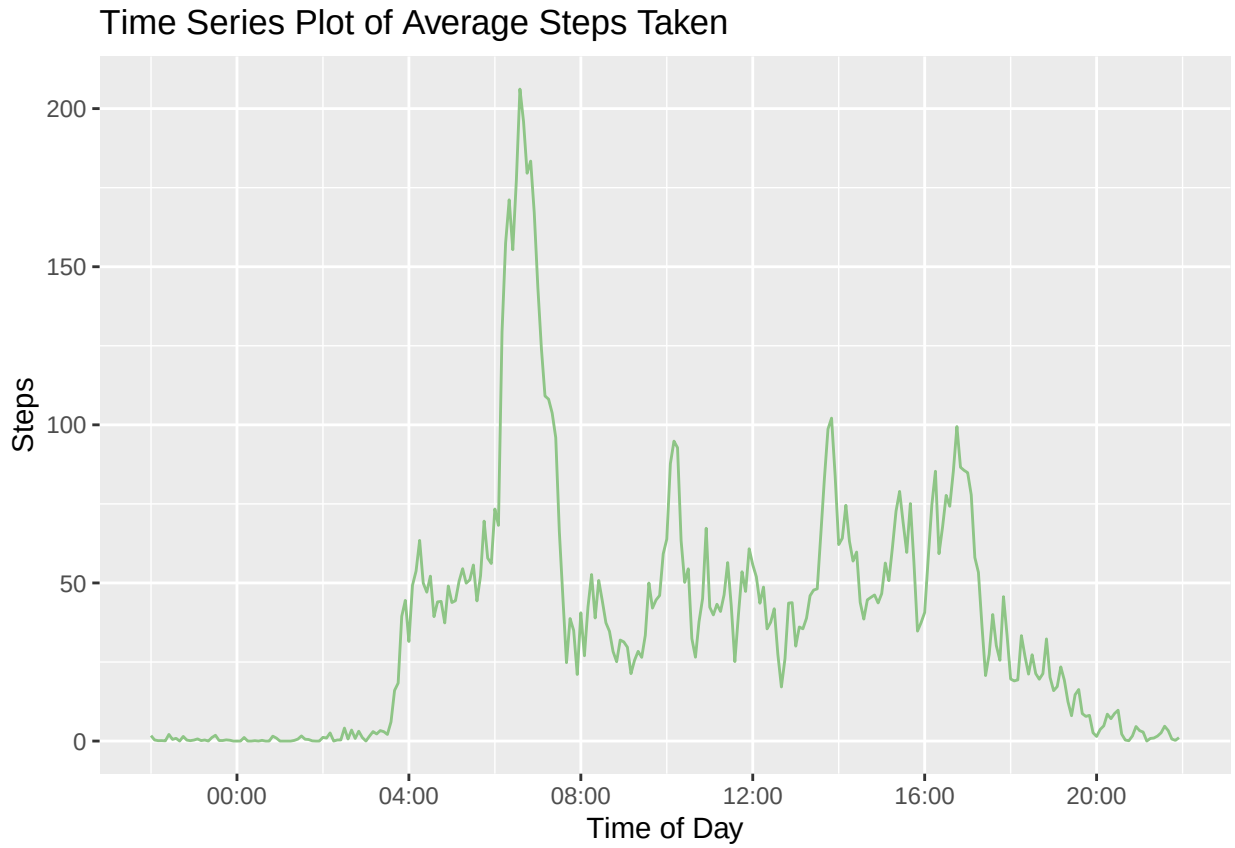
2. Converting 'interval' column data in a valid time data format

```
interval_steps$time <- as.character(interval_steps$interval)
for (i in 1:2){
  interval_steps$time[i] <- as.character(paste0("0",interval_steps$time[i]))
}
```

```
for (i in 1:12){
  interval_steps$time[i] <- as.character(paste0("00",interval_steps$time[i]))
}
for (i in 13:120){
  interval_steps$time[i] <- as.character(paste0("0",interval_steps$time[i]))
}
interval_steps$time <- as.POSIXct(interval_steps$time, format = "%H%M")
```

3. Making a time series plot for 24- hour time period

```
plot2 <- ggplot(interval_steps, aes(x = time, y = steps)) +
  geom_line(col = "#8EC586") +
  labs(title = "Time Series Plot of Average Steps Taken", x = "Time of Day", y = "Steps") +
  scale_x_datetime(labels = date_format("%H:%M", tz = "MST"), date_breaks = "4 hours")
print(plot2)
```



4. Which 5-minute interval, on average across all the days in the dataset, contains the maximum number of steps?

```
interval_steps[which.max(interval_steps$steps),1:2]
```

```
##     interval    steps
## 104      835 206.1698
```

Imputing missing values

1. Calculating the total numbers of missing values in the dataset

```
nas <- is.na(activity_data$steps)
sum(nas)
```

```
## [1] 2304
```

```
mean(nas)
```

```
## [1] 0.1311475
```

2. Replacing the missing values

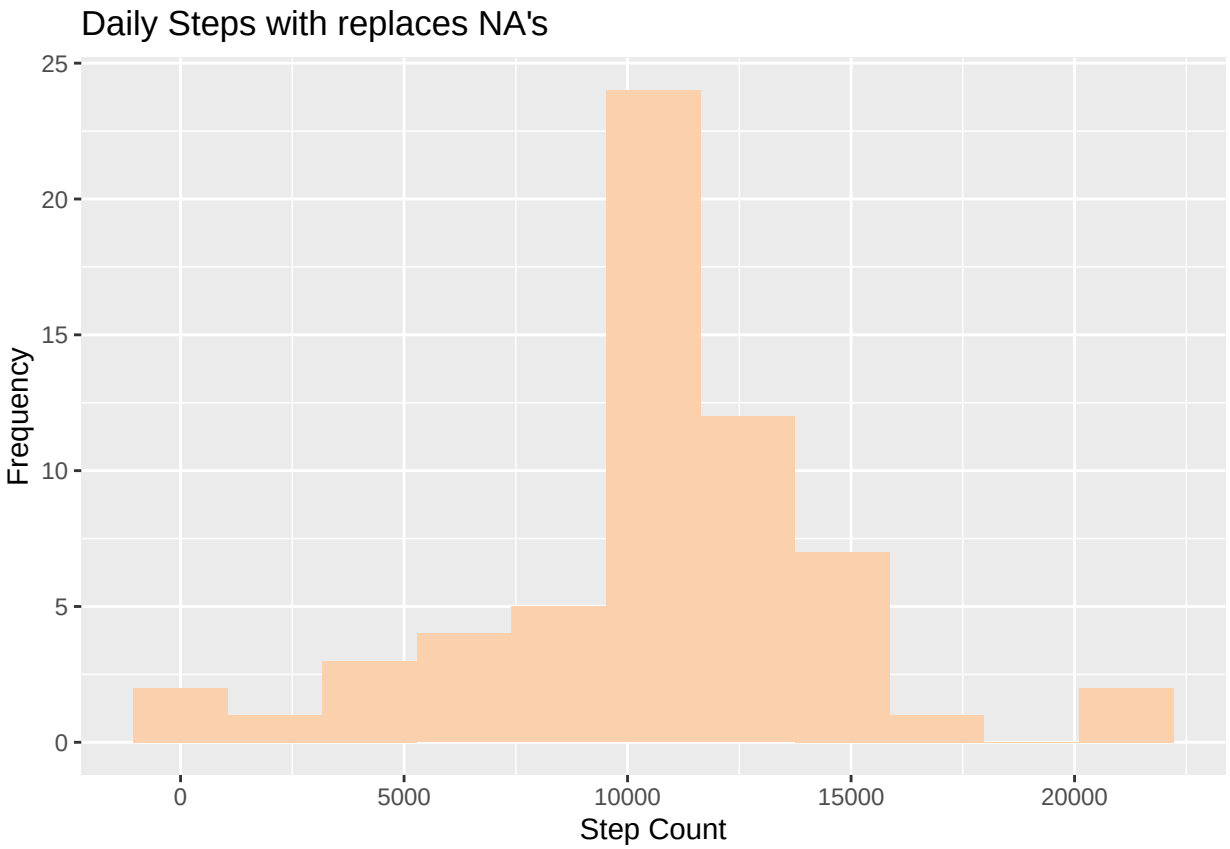
```
replaceNas= function(steps, interval) {
  replace = NA
  if (!is.na(steps)) {
    replace = steps }
  else {
    replace = interval_steps[interval_steps$interval == interval, "steps"]}
  return(replace) }
```

3. Applying a function for replacing missing values: a new dataset

```
filled_activity_data = activity_data
filled_activity_data$steps = mapply(replaceNas, filled_activity_data$steps, filled_activity_data$interva
```

4. Making a new histogram of the total number of steps taken each day

```
total_steps_filled <- aggregate(steps ~ date, data = filled_activity_data, sum)
plot3 <- ggplot(total_steps_filled, aes(x = steps)) +
  geom_histogram(fill = "#FBD1AD", bins = 11) +
  labs(title = "Daily Steps with replaces NA's", x = "Step Count", y = "Frequency")
print(plot3)
```



Are there differences in activity patterns between weekdays and weekends?

1. Create a new factor variable in the dataset with two levels – "weekday" and "weekend" indicating whether a given date is a weekday or weekend day.

```
filled_activity_data$day <- weekdays(filled_activity_data$date)
weekday <- c("Monday","Tuesday","Wednesday","Thursday","Friday")
weekDayOp <- function(dayofweek) {
  fill = ""
  if (dayofweek %in% weekday) {
    fill = "Weekday" }
  else {
    fill = "Weekend" }
  return(fill) }

filled_activity_data$weekday <- mapply(weekDayOp,filled_activity_data$day)
```

Calculating the average number of steps

```
filled_totals_day <- aggregate(steps ~ interval + weekday, data = filled_activity_data, mean)
summary(filled_totals_day)
```

```
##     interval        weekday              steps
##  Min.   :   0.0   Length:288         Min.   :  0.000
```

```
##  1st Qu.: 588.8   Class :character   1st Qu.:  2.486
##  Median :1177.5   Mode  :character   Median : 34.113
##  Mean   :1177.5                      Mean   : 37.383
##  3rd Qu.:1766.2                      3rd Qu.: 52.835
##  Max.   :2355.0                      Max.   :206.170
```

Coverting 'interval' column data to a vailid date-time format

```
filled_totals_day$time <- as.character(filled_totals_day$interval)
for (i in 1:2){
  filled_totals_day$time[i] <- as.character(paste0("0",filled_totals_day$time[i]))
}
for (i in 1:12){
  filled_totals_day$time[i] <- as.character(paste0("0",filled_totals_day$time[i]))
}
for (i in 13:120){
  filled_totals_day$time[i] <- as.character(paste0("0",filled_totals_day$time[i]))
}
for (i in 121:288){
  filled_totals_day$time[i] <- as.character(paste0("0",filled_totals_day$time[i]))
}
filled_totals_day$time <- as.POSIXct(filled_totals_day$time, format = "%H%M")
head(filled_totals_day,10)
```

```
##    interval weekday     steps                time
## 1         0 Weekend 1.7169811 2025-06-11 00:00:00
## 2         5 Weekend 0.3396226 2025-06-11 00:05:00
## 3        10 Weekend 0.1320755 2025-06-11 01:00:00
## 4        15 Weekend 0.1509434 2025-06-11 01:05:00
## 5        20 Weekend 0.0754717 2025-06-11 02:00:00
## 6        25 Weekend 2.0943396 2025-06-11 02:05:00
## 7        30 Weekend 0.5283019 2025-06-11 03:00:00
## 8        35 Weekend 0.8679245 2025-06-11 03:05:00
## 9        40 Weekend 0.0000000 2025-06-11 04:00:00
## 10       45 Weekend 1.4716981 2025-06-11 04:05:00
```

Creating time series plot

```
plot4 <- ggplot(filled_totals_day, aes(time, steps, col = factor(weekday))) +
  facet_grid(.~factor(weekday)) +
  geom_line(show.legend = F) +
  labs(x = "Time of Day") +
  labs(y = "Steps") +
  labs(title = "Time Series Plot Comparison of Steps") +
  scale_x_datetime(labels = date_format("%H:%M", tz = "MST"), date_breaks = "6 hours")
print(plot4)
```

```
## Warning: Removed 48 rows containing missing values or values outside the scale range
## (`geom_line()`).
```


Time Series Plot Comparison of Steps
Weekend
200
150
100
50
0
Steps
03:00
Time of Day