Course Project 1

Alejandro

2025-03-17

Project 1

Description of the activities can be consulted in https://github.com/rdpeng/RepData_PeerAssessment1.

Note: Since the figures are not saved automatically in the current directory, we do it manually. Here I create the directory *figure*

```
dir.create("figure")
```

```
## Warning in dir.create("figure"): 'figure' already exists
```

Loading and preprocessing the data

First, we can load the data set

```
df_orig <- read.csv("activity.csv")
```

Then, can observe how data look like

```
head(df_orig)
```

```
##   steps       date interval
## 1    NA 2012-10-01        0
## 2    NA 2012-10-01        5
## 3    NA 2012-10-01       10
## 4    NA 2012-10-01       15
## 5    NA 2012-10-01       20
## 6    NA 2012-10-01       25
```

What is mean total number of steps taken per day?

First, let's ignore the NA's

```
df <- df_orig[complete.cases(df_orig),]
```

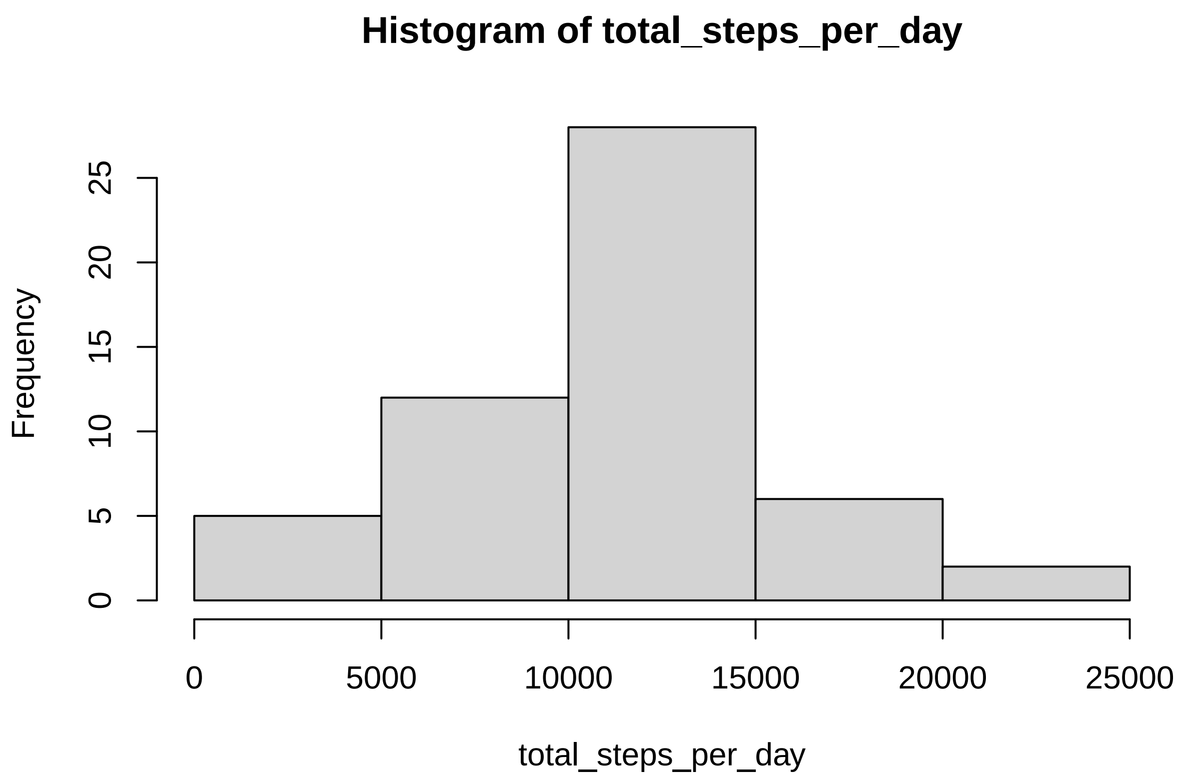
Now, we can see how the data frame looks like

```
head(df)
```

```
##     steps       date interval
## 289     0 2012-10-02        0
## 290     0 2012-10-02        5
## 291     0 2012-10-02       10
## 292     0 2012-10-02       15
## 293     0 2012-10-02       20
## 294     0 2012-10-02       25
```

Now, we proceed to calculate the total number of steps per day and make a histogram

```
by_day <- split(df$steps, df$date)
total_steps_per_day <- sapply(by_day, sum)
hist(total_steps_per_day)
```



```
dev.copy(png, "figure/hist_ignoring_NAs.png")
```

```
## png
##   3
```

```
dev.off()
```

```
## pdf
##   2
```

Then, we can see the mean and median of the total number of steps taken per day

```
mean(total_steps_per_day)
```

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

```
median(total_steps_per_day)
```

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

What is the average daily activity pattern?

We can compute the average number of steps across all the days in the following way

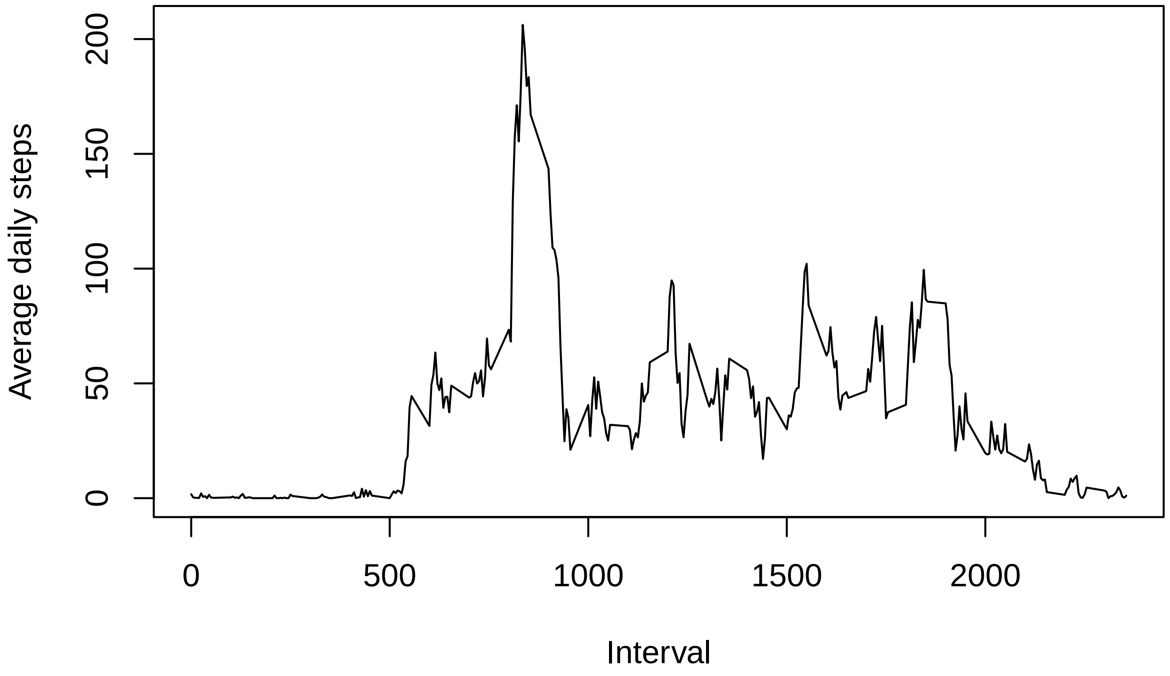
```
steps_by_interval <- split(df$steps, df$interval)
a <- sapply(steps_by_interval, mean)
```

Another option, could be the following

```r
b <- numeric()
df_steps_by_interval <- data.frame( steps_by_interval )
for (i in 1:ncol(df_steps_by_interval))
    b[i] <- mean(df_steps_by_interval[,i])
```

Then, we can make a time series like plot like the next one

```r
plot(names(steps_by_interval), a, type="l", xlab="Interval", ylab = "Average daily steps")
```



```r
dev.copy(png, "figure/Average_daily_steps.png")
```

```
## png
##   3
```

```r
dev.off()
```

```
## pdf
##   2
```

We can find out which 5-minute interval contains the maximum number of steps taken like this

```r
which.max(a)
```

```
## 835
## 104
```

where the first integer denotes the required interval (835) and the second integer (104) corresponds to the id in the list as we can see

```r
a[104]
```

```
##      835
## 206.1698
```

Imputing missing values

Let's consider the original data set again. The number of NA's in the data set is

```r
sapply(df_orig, function(x) sum(is.na(x)))
```

```
##    steps     date interval 
##     2304        0        0
```

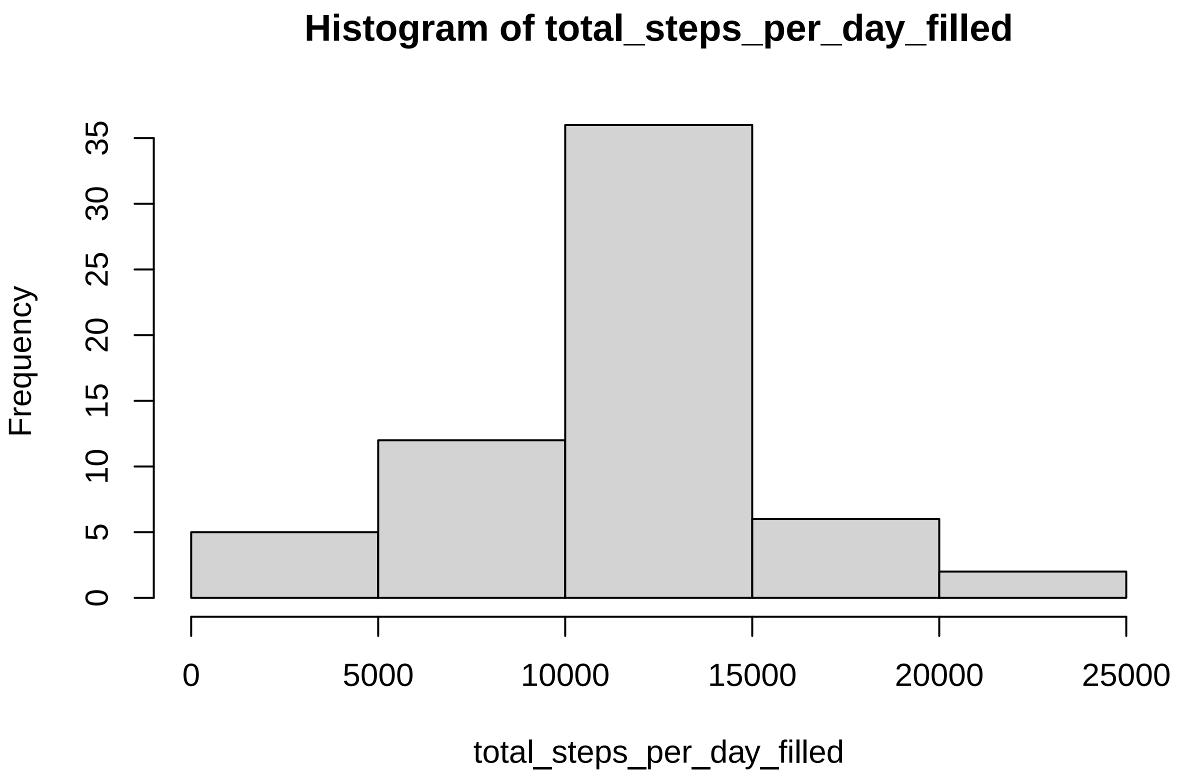
Let's fill the NA values with the **median** for that 5-minute interval

```r
df_filled <- df_orig
for (i in 1:nrow(df_filled))
    if ( is.na(df_filled$steps[i]) )
    {
        id <- which(df_filled$interval == df_filled$interval[i])[[1]]
        df_filled$steps[i] <- b[id]
    }

head(df_filled)
```

```
##       steps       date interval
## 1 1.7169811 2012-10-01        0
## 2 0.3396226 2012-10-01        5
## 3 0.1320755 2012-10-01       10
## 4 0.1509434 2012-10-01       15
## 5 0.0754717 2012-10-01       20
## 6 2.0943396 2012-10-01       25
```

Now, let's repeat the histogram analysis we did before but this time using the data frame with the imputing missing data version of the data set

```r
by_day_filled <- split(df_filled$steps, df_filled$date)
total_steps_per_day_filled <- sapply(by_day_filled, sum)
hist(total_steps_per_day_filled)
```



```r
dev.copy(png, "figure/hist_filling_NAs.png")
```

```
## png
##   3
```

```r
dev.off()
```

```
## pdf
##   2
```

```r
mean(total_steps_per_day_filled)
```

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

```r
median(total_steps_per_day_filled)
```

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

So, **Do these values differ from the estimates from the first part of the assignment?** The answer is yes. We can see in the y-axis that now the scale has risen up to above 35 (in the first histogram the y-axis scale maximum value was 25).

Hence, **What is the impact of imputing missing data on the estimates of the total daily number of steps?** Well, aside what we have just pointed out, we can see that the median gets closer to the mean. This is due to the fact that we are artificially adding more values which are identical to the mean. In the limit case, the mean and median will be the same.

Are there differences in activity patterns between weekdays and weekends?

Consider the data set where the NA's have been removed. First, let's add the column corresponding to the day-type (weekday/weekend)

```r
days <- weekdays(as.Date(df$date))
day <- character()

for (i in 1:length(days))
{
    if ( days[i] == "Saturday" | days[i] == "Sunday")  day[i] <- "Weekend"
    if ( days[i] != "Saturday" & days[i] != "Sunday")  day[i] <- "Weekday"
}

df$day <- day
```

We can see this new column in the data frame

```r
head(df)
```

```
##     steps       date interval     day
## 289     0 2012-10-02        0 Weekday
## 290     0 2012-10-02        5 Weekday
## 291     0 2012-10-02       10 Weekday
## 292     0 2012-10-02       15 Weekday
## 293     0 2012-10-02       20 Weekday
## 294     0 2012-10-02       25 Weekday
```

```r
table(df$day)
```

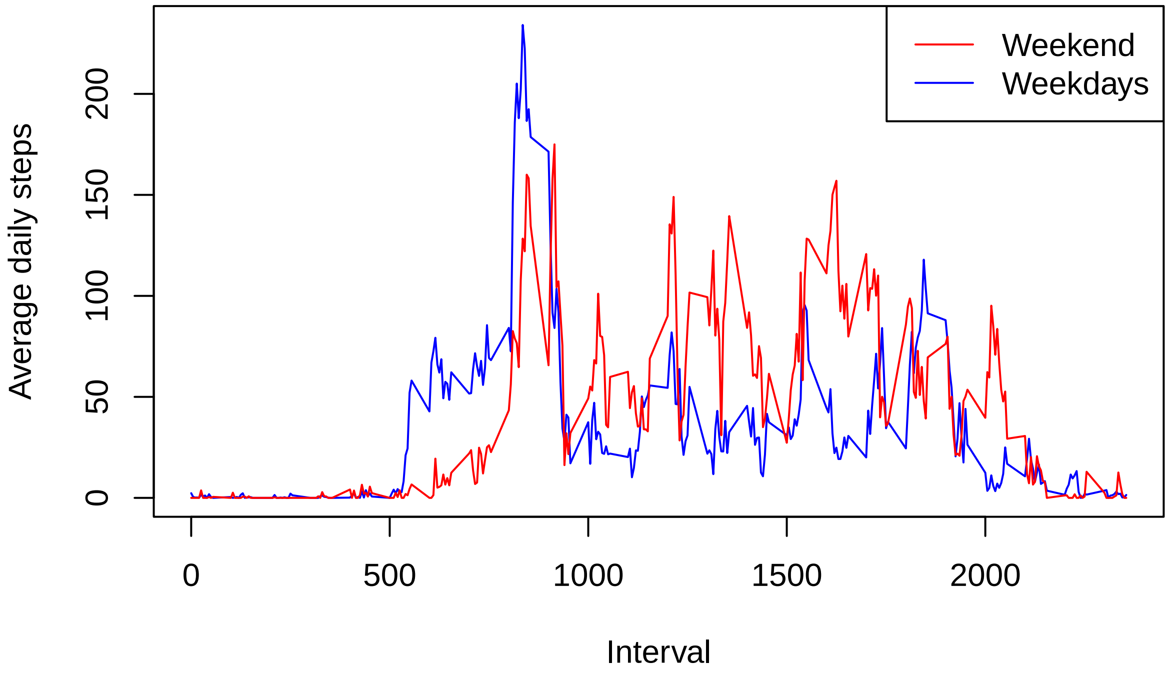
```
## 
## Weekday Weekend 
##   11232    4032
```

Then, we can compute the step average per interval constrained to weekdays or weekends

```
df_weekends <- data.frame(matrix(ncol = 3, nrow = 0))
df_weekdays <- data.frame(matrix(ncol = 3, nrow = 0))

colnames(df_weekends) <- names(df)[1:3]
colnames(df_weekdays) <- names(df)[1:3]

for (i in 1:nrow(df))
{
    if ( df$day[i] == "Weekday" ) df_weekdays[nrow(df_weekdays)+1,] <- df[i,1:3]
    if ( df$day[i] == "Weekend" ) df_weekends[nrow(df_weekends)+1,] <- df[i,1:3]
}

steps_by_interval_weekdays <- split(df_weekdays$steps, df_weekdays$interval)
a_weekdays <- sapply(steps_by_interval_weekdays, mean)

steps_by_interval_weekends <- split(df_weekends$steps, df_weekends$interval)
a_weekends <- sapply(steps_by_interval_weekends, mean)
```

Finally, we can visualize the averages

```
plot(names(steps_by_interval_weekdays), a_weekdays, type="l", col = "blue", xlab="Interval", ylab = "Ave
lines(names(steps_by_interval_weekends), a_weekends, type="l", col = "red")
legend("topright", legend=c("Weekend", "Weekdays"), lty=c(1,1), col=c("red", "blue"))
```



```
dev.copy(png, "figure/Average daily steps by day.png")
```

```
## png
##   3
```

```
dev.off()
```

```
## pdf
##   2
```