Reproducible Research: Peer Assessment 1

Benoit Séguin

2024-09-22

This document presents the steps necessary to analyze the activity data which shows the number of steps taken in 5 minute intervals

## Loading and preprocessing the data

To answer the question, we will first need to load the required libraries to execute the code

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(tidyr)  
library(lubridate)

##   
## Attaching package: 'lubridate'

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

Then we will load the activity.csv data using read.csv

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

Finally, we will prepare the data to ensure we can analyze it properly

activity$date = as.Date(activity$date,"%Y-%m-%d")

## What is mean total number of steps taken per day?

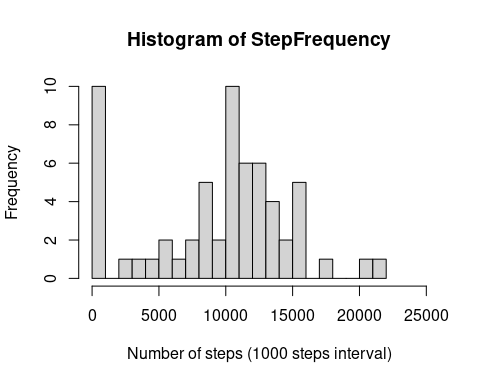
#### 1. Calculate the total number of steps taken per day, ignoring missing values as indicated.

DailySteps <- activity %>% group\_by(date) %>% summarize(Steps = sum(steps,na.rm = TRUE))

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

#### 2. Plot histogram

StepFrequency <- DailySteps$Steps  
hist(StepFrequency, breaks=25, xlim=c(0,25000), xlab = "Number of steps (1000 steps interval)")



#### 3. Calculate and report the mean and median of total number of steps

MeanDailySteps <- mean(DailySteps$Steps, na.rm = TRUE)  
MedianDailySteps <- median(DailySteps$Steps, na.rm = TRUE)

The daily steps mean is:

print(MeanDailySteps)

## [1] 9354.23

The daily steps median is:

print(MedianDailySteps)

## [1] 10395

## What is the average daily activity pattern?

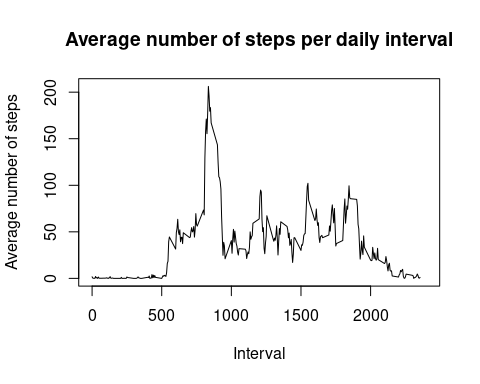
#### 1. Timeseries plot of the average number of steps per interval

Sum the total number of steps per interval, then average by interval and plot, ignoring missing values as indicated.

IntervalSteps <-   
 activity %>% group\_by(interval) %>%   
 summarize(mean = mean(steps, na.rm = TRUE))

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

plot(  
 IntervalSteps$interval,IntervalSteps$mean,  
 xlim=c(0,2400),  
 main = "Average number of steps per daily interval",  
 xlab = "Interval",  
 ylab = "Average number of steps",  
 type="l")



#### 2. Identify which 5 minute interval, on average, contains the maximum number of steps

Use the max function to identify all 5 minute intervals with the max average value.

MaxInterval <-   
 IntervalSteps[IntervalSteps$mean == max(IntervalSteps$mean), ]$interval

The interval with the max average is:

print(MaxInterval)

## [1] 835

## Imputing missing values

#### 1. Calculate and report the total number of missing values

Use the function is.na to identify missing values

sum(is.na(activity$steps))

## [1] 2304

sum(is.na(activity$date))

## [1] 0

sum(is.na(activity$interval))

## [1] 0

There are 2304 records with missing steps

#### 2. Impute interval mean for missing steps

Replicate Steps column and Impute interval mean to NA values

activity$ImputedSteps <- activity$steps  
activity <- activity %>%   
 group\_by(interval) %>%  
 mutate(  
 ImputedSteps = ifelse(is.na(ImputedSteps), mean(ImputedSteps, na.rm = TRUE),ImputedSteps))

Confirm no NA

sum(is.na(activity$ImputedSteps))

## [1] 0

#### 3. Create a new dataset that is equal to the original dataset but with the missing data filled in.

Create new dataset

activityNew <- activity[,c("ImputedSteps","date","interval")]  
names(activityNew)[1] <- "steps"  
head(activityNew)

## # A tibble: 6 x 3  
## # Groups: interval [6]  
## steps date interval  
## <dbl> <date> <int>  
## 1 1.72 2012-10-01 0  
## 2 0.340 2012-10-01 5  
## 3 0.132 2012-10-01 10  
## 4 0.151 2012-10-01 15  
## 5 0.0755 2012-10-01 20  
## 6 2.09 2012-10-01 25

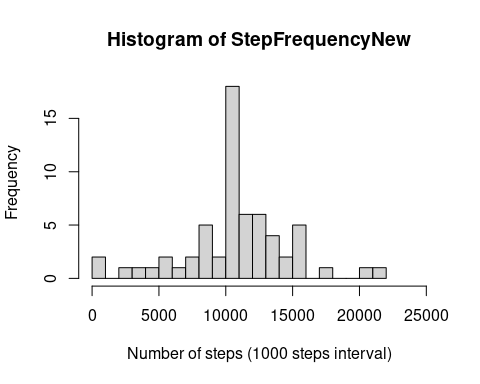
#### 3. Recreate Q1 with activityNew. Are there differences in Mean and Median total number of steps taken per day

Repurpose code from Q1 with activityNew to create new histogram

DailyStepsNew <- activityNew %>% group\_by(date) %>% summarize(Steps = sum(steps,na.rm = TRUE))

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

StepFrequencyNew <- DailyStepsNew$Steps  
hist(StepFrequencyNew, breaks=25, xlim=c(0,25000), xlab = "Number of steps (1000 steps interval)")



Calculate and report mean and median total number of steps

The new daily steps mean and median are:

MeanDailyStepsNew <- mean(DailyStepsNew$Steps, na.rm = TRUE)  
MedianDailyStepsNew <- median(DailyStepsNew$Steps, na.rm = TRUE)  
print(MeanDailyStepsNew)

## [1] 10766.19

print(MedianDailyStepsNew)

## [1] 10766.19

The impact of imputing missing data on the estimates of total daily has increased both the mean and median

## Are there differences in activity patterns between weekdays and weekends?

#### 1. Add weekday to the activityNew dataset

activityNew$Day <- weekdays(activityNew$date)

Create factor variable to identify if weekday or weekend

activityNew$Weekday <- factor(ifelse(wday(activityNew$date) %in% c(1,7),"Weekend","Weekday"))

#### 2. Timeseries panel plot

Time series panel plot for the 5 minute intervals and average number of steps taken, averaged accross all weekday or weekend days

WeekdaySteps <-   
 activityNew %>% group\_by(Weekday,interval) %>%   
 summarize(mean = mean(steps, na.rm = TRUE))

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

par(mfrow = c(2, 1), mar=c(2,4,2,0.2))  
plot(  
 WeekdaySteps$interval[WeekdaySteps$Weekday == "Weekday"],  
 WeekdaySteps$mean[WeekdaySteps$Weekday == "Weekday"],  
 xlim=c(0,2400),  
 ylim=c(0,250),  
 main = "Weekday average number of steps per daily interval",  
 xlab = "Interval",  
 ylab = "Average number of steps",  
 type="l")  
plot(  
 WeekdaySteps$interval[WeekdaySteps$Weekday == "Weekend"],  
 WeekdaySteps$mean[WeekdaySteps$Weekday == "Weekend"],  
 xlim=c(0,2400),  
 ylim=c(0,250),  
 main = "Weekend average number of steps per daily interval",  
 xlab = "Interval",  
 ylab = "Average number of steps",  
 type="l")

