Reproducible Research: Peer Assessment 1 week 2

Nipun Jain

4 April 2018

# 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.

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.

# DATA

The dataset contains 3 variables and 17568 observations over a period of 61 days, October and November 2012.

Variables:

* 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, there are 61 days in the dataset.
* interval: Identifier for the 5-minute interval in which measurement was taken, There are 288 intervals per day (24 hours \* 12 intervals of 5 minutes per hour), numbered from 0 to 2355. Interval 115 will be the 5 minutes interval starting at 01.15am. For each day, the first interval is 0 (00.00pm), the last interval is 2355 (11.55pm)

The elements below follow the structure proposed on the [assignment page](https://www.coursera.org/learn/reproducible-research/peer/gYyPt/course-project-1)

## Loading and preprocessing the data

The data was downloaded on April 4th, 2018, from [this link](https://d396qusza40orc.cloudfront.net/repdata%2Fdata%2Factivity.zip).

We start by removing all objects from the workspace and load the data:

#remove all objects  
rm(list=ls())  
#set working directory:   
  
#we check if the source file is loaded onto the wd of the reader, if not, we download it and unzip the file:  
if(!file.exists("activity.csv")) {  
 tempfile <- tempfile()  
 download.file("http://d396qusza40orc.cloudfront.net/repdata%2Fdata%2Factivity.zip",destfile = tempfile)  
 unzip(tempfile)  
 unlink(tempfile)  
}  
#load data  
activity <- read.csv("activity.csv")

We can then inspect the dataset:

summary(activity)

## steps date interval   
## Min. : 0.00 2012-10-01: 288 Min. : 0.0   
## 1st Qu.: 0.00 2012-10-02: 288 1st Qu.: 588.8   
## Median : 0.00 2012-10-03: 288 Median :1177.5   
## Mean : 37.38 2012-10-04: 288 Mean :1177.5   
## 3rd Qu.: 12.00 2012-10-05: 288 3rd Qu.:1766.2   
## Max. :806.00 2012-10-06: 288 Max. :2355.0   
## NA's :2304 (Other) :15840

str(activity)

## 'data.frame': 17568 obs. of 3 variables:  
## $ steps : int NA NA NA NA NA NA NA NA NA NA ...  
## $ date : Factor w/ 61 levels "2012-10-01","2012-10-02",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ interval: int 0 5 10 15 20 25 30 35 40 45 ...

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

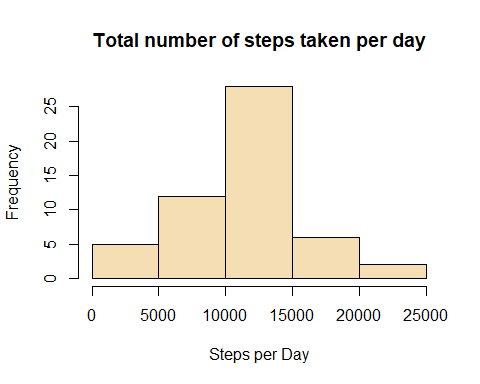
### 1. Total number of steps taken per day

We use the aggregate function, removing NAs, and draw the histogram with base plotting:

activity\_steps\_day <- aggregate(steps ~ date, data = activity, FUN = sum, na.rm = TRUE)

### 2. Histogram of the total number of steps taken each day.

hist(activity\_steps\_day$steps, xlab = "Steps per Day", main = "Total number of steps taken per day", col = "wheat")



Note: we could set the number of breaks (bins) to values higher than 5, but we chose not to, the insights seem sufficient at this stage.

### 3. Mean and median of the total number of steps taken per day

mean\_steps <- mean(activity\_steps\_day$steps)  
median\_steps <- median(activity\_steps\_day$steps)  
#we set a normal number format to display the results  
mean\_steps <- format(mean\_steps,digits=1)  
median\_steps <- format(median\_steps,digits=1)

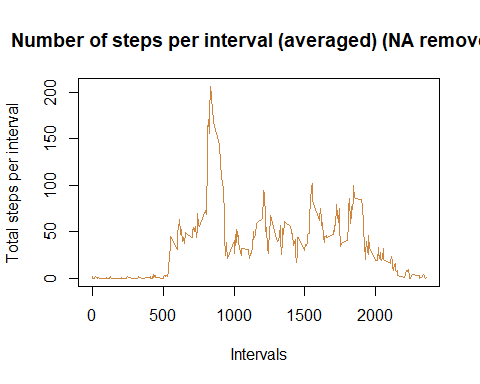
Mean steps per day: **10766**  
Median steps per day: **10765**

## What is the average daily activity pattern?

### 1. Time series plot

Time series plot of the 5-minute interval (x-axis) and the average number of steps taken, averaged across all days (y-axis)

#Aggregate function for mean over all days, for each interval  
activity\_steps\_mean <- aggregate(steps ~ interval, data = activity, FUN = mean, na.rm = TRUE)  
#Plot  
plot(activity\_steps\_mean$interval, activity\_steps\_mean$steps, type = "l", col = "tan3", xlab = "Intervals", ylab = "Total steps per interval", main = "Number of steps per interval (averaged) (NA removed)")



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

#what is the highest steps value? (maximum of steps on one given interval)  
max\_steps <-max(activity\_steps\_mean$steps)  
#for which interval are the numbers of steps per interval at the highest?  
max\_interval <- activity\_steps\_mean$interval[which(activity\_steps\_mean$steps == max\_steps)]  
max\_steps <- round(max\_steps, digits = 2)

The highest number of steps for a 5 minutes interval is **206.17**,  
which corresponds to interval **835**

## Imputing missing values

### 1. Calculate total number of missing values in the dataset

sum(is.na(activity))

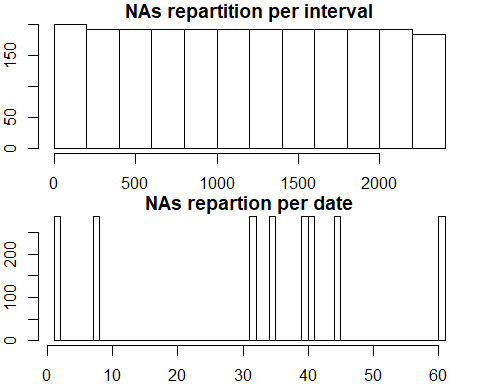
## [1] 2304

This corresponds to the summary results found at the top of document, all NA values found in ‘steps’ variable only.

### 2. Devise a strategy for filling in all of the missing values in the dataset

We run a couple of unsophisticated charts to decide which unsophisticated strategy we’ll adopt

#subset general dataset with missing values only  
missing\_values <- subset(activity, is.na(steps))  
#plot repartition, by date or by intervals  
par(mfrow = c(2,1), mar = c(2, 2, 1, 1))  
hist(missing\_values$interval, main="NAs repartition per interval")  
hist(as.numeric(missing\_values$date), main = "NAs repartion per date", breaks = 61)



We see that NAs run equally over all intervals. On the other hand, checking with dates, we see all NA’s are spread between 8 specific days only. To reduce that effect, best will be to take the mean for missing interval across all the days in the dataset.

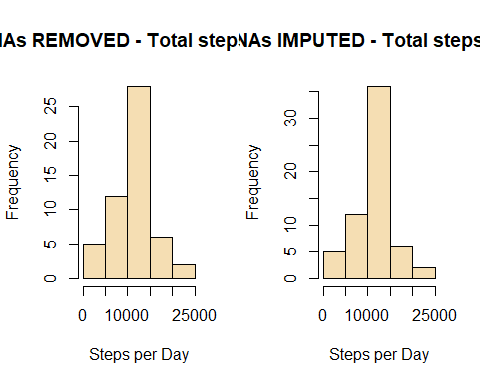
### 3. Create new dataset with the missing data filled in

We’ll follow these steps to replace the missing data:  
- calculate average steps per interval, across all the days  
- cut the ‘activity’ dataset in two parts: activity\_NAs, activity\_non\_NAs  
- in ‘activity\_NAs’, we convert, we convert the ‘steps’ variable into a factor, then replace levels with new computed values  
- a bit of formatting to convert the ‘steps’ vector variable into an integer vector  
- merge/rbind the two datasets into a new dataset, with missing data filled in

# calculate mean of steps per interval, we end up with a mean for all 288 intervals  
MeanStepsPerInterval <- tapply(activity$steps, activity$interval, mean, na.rm = TRUE)  
# cut the 'activity' dataset in 2 parts (with and without NAs)  
activity\_NAs <- activity[is.na(activity$steps),]  
activity\_non\_NAs <- activity[!is.na(activity$steps),]  
#replace missing values in activity\_NAs  
activity\_NAs$steps <- as.factor(activity\_NAs$interval)  
levels(activity\_NAs$steps) <- MeanStepsPerInterval  
#change the vector back as integer   
levels(activity\_NAs$steps) <- round(as.numeric(levels(activity\_NAs$steps)))  
activity\_NAs$steps <- as.integer(as.vector(activity\_NAs$steps))  
#merge/rbind the two datasets together  
imputed\_activity <- rbind(activity\_NAs, activity\_non\_NAs)

### 4. Make a histogram of the total number of steps taken each day

#Plotting parameters to place previous histogram and new one next to each other  
par(mfrow = c(1,2))  
#Plot again the histogram from the first part of the assignment  
activity\_steps\_day <- aggregate(steps ~ date, data = activity, FUN = sum, na.rm = TRUE)  
hist(activity\_steps\_day$steps, xlab = "Steps per Day", main = "NAs REMOVED - Total steps/day", col = "wheat")  
#Plot new histogram, with imputed missing values  
imp\_activity\_steps\_day <- aggregate(steps ~ date, data = imputed\_activity, FUN = sum, na.rm = TRUE)  
hist(imp\_activity\_steps\_day$steps, xlab = "Steps per Day", main = "NAs IMPUTED - Total steps/day", col = "wheat")



We calculate like previously the mean and median values, and store the new and old results in a data frame for easier comparison:

imp\_mean\_steps <- mean(imp\_activity\_steps\_day$steps)  
imp\_median\_steps <- median(imp\_activity\_steps\_day$steps)  
#we set a normal number format to display the results  
imp\_mean\_steps <- format(imp\_mean\_steps,digits=1)  
imp\_median\_steps <- format(imp\_median\_steps,digits=1)  
#store the results in a dataframe  
results\_mean\_median <- data.frame(c(mean\_steps, median\_steps), c(imp\_mean\_steps, imp\_median\_steps))  
colnames(results\_mean\_median) <- c("NA removed", "Imputed NA values")  
rownames(results\_mean\_median) <- c("mean", "median")

We use the xtable package to print the table with all values:

library(xtable)

## Warning: package 'xtable' was built under R version 3.4.4

xt <- xtable(results\_mean\_median)  
print(xt, type = "html")

NA removed

Imputed NA values

mean

10766

10766

median

10765

10762

CONCLUSIONS:  
Imputing missing values didn’t change the mean value whatsoever, the median value is reduced only by 0.027% (3/10765\*100), which is as good as nothing.  
Both histograms show the same structure, with imputed NAs we notice however higher frequencies.

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

### 1. Create a new factor variable

#elseif function to categorize Saturday and Sunday as factor level "weekend", all the rest as "weekday"  
imputed\_activity$dayType <- ifelse(weekdays(as.Date(imputed\_activity$date)) == "Samstag" | weekdays(as.Date(imputed\_activity$date)) == "Sonntag", "weekend", "weekday")  
#transform dayType variable into factor  
imputed\_activity$dayType <- factor(imputed\_activity$dayType)

### 2. Panel plot containing time series plot

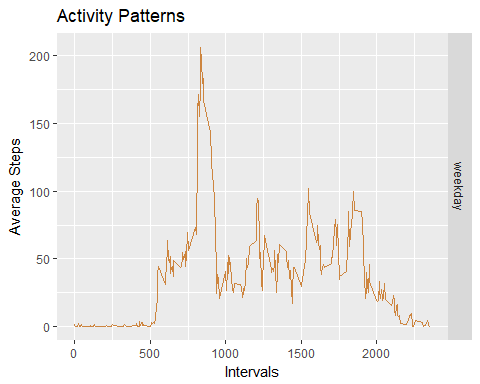
#Aggregate a table showing mean steps for all intervals, acrlss week days and weekend days  
steps\_interval\_dayType <- aggregate(steps ~ interval + dayType, data = imputed\_activity, FUN = mean)  
#verify new dataframe   
head(steps\_interval\_dayType)

## interval dayType steps  
## 1 0 weekday 1.75409836  
## 2 5 weekday 0.29508197  
## 3 10 weekday 0.11475410  
## 4 15 weekday 0.13114754  
## 5 20 weekday 0.06557377  
## 6 25 weekday 2.08196721

#add descriptive variables  
names(steps\_interval\_dayType) <- c("interval", "day\_type", "mean\_steps")  
#plot with ggplot2  
library(ggplot2)

## Warning: package 'ggplot2' was built under R version 3.4.4

plot <- ggplot(steps\_interval\_dayType, aes(interval, mean\_steps))  
plot + geom\_line(color = "tan3") + facet\_grid(day\_type~.) + labs(x = "Intervals", y = "Average Steps", title = "Activity Patterns")



CONCLUSION: it seems the tested subjects have an earlier start in the week days with a peak between 8am and 9am. On weekends, the activity seems more spread between 8am and 8pm.