

# Reproducible Research Project 1

*Steinaka*

14/1/2019

github repo for rest of specialization: Data Science Coursera  
(<https://github.com/steinaka/ReproducibleResearchP1>)

## 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:

- Dataset: Activity monitoring data (<https://d396qusza40orc.cloudfront.net/repdata%2Fdata%2Factivity.zip>)

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 and preprocessing the data

Load the data (i.e. `read.csv()`)

Process/transform the data (if necessary) into a format suitable for your analysis

Unzip data to obtain a csv file.

```
library("data.table")  
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.5.2
```

```
fileUrl <- "https://d396qusza40orc.cloudfront.net/repdata%2Fdata%2Factivity.zip"
download.file(fileUrl, destfile = paste0(getwd(), '/repdata%2Fdata%2Factivity.zip'), method = "curl")
unzip("repdata%2Fdata%2Factivity.zip", exdir = "data")
```

## Reading csv Data into Data.Table.

```
activityDT <- data.table::fread(input = "data/activity.csv")
```

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

### 1. Calculate the total number of steps taken per day

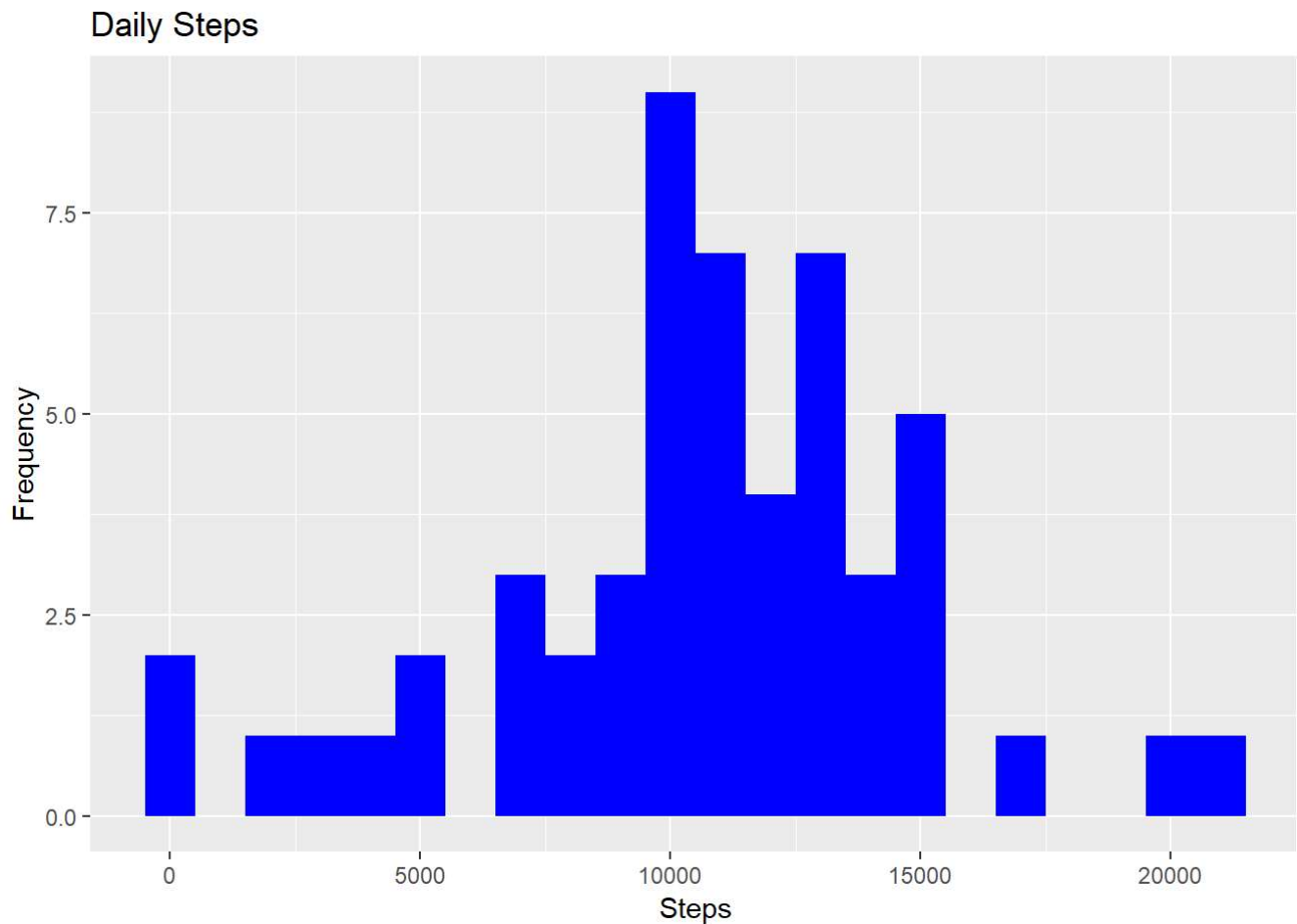
```
Total_Steps <- activityDT[, c(lapply(.SD, sum, na.rm = FALSE)), .SDcols = c("steps"), by = .(date)]
head(Total_Steps, 10)
```

<b>date</b> <chr>	<b>steps</b> <int>
2012-10-01	NA
2012-10-02	126
2012-10-03	11352
2012-10-04	12116
2012-10-05	13294
2012-10-06	15420
2012-10-07	11015
2012-10-08	NA
2012-10-09	12811
2012-10-10	9900
1-10 of 10 rows	

### 2. If you do not understand the difference between a histogram and a barplot, research the difference between them. Make a histogram of the total number of steps taken each day.

```
ggplot(Total_Steps, aes(x = steps)) +
  geom_histogram(fill = "blue", binwidth = 1000) +
  labs(title = "Daily Steps", x = "Steps", y = "Frequency")
```

```
## Warning: Removed 8 rows containing non-finite values (stat_bin).
```



3. Calculate and report the mean and median of the total number of steps taken per day

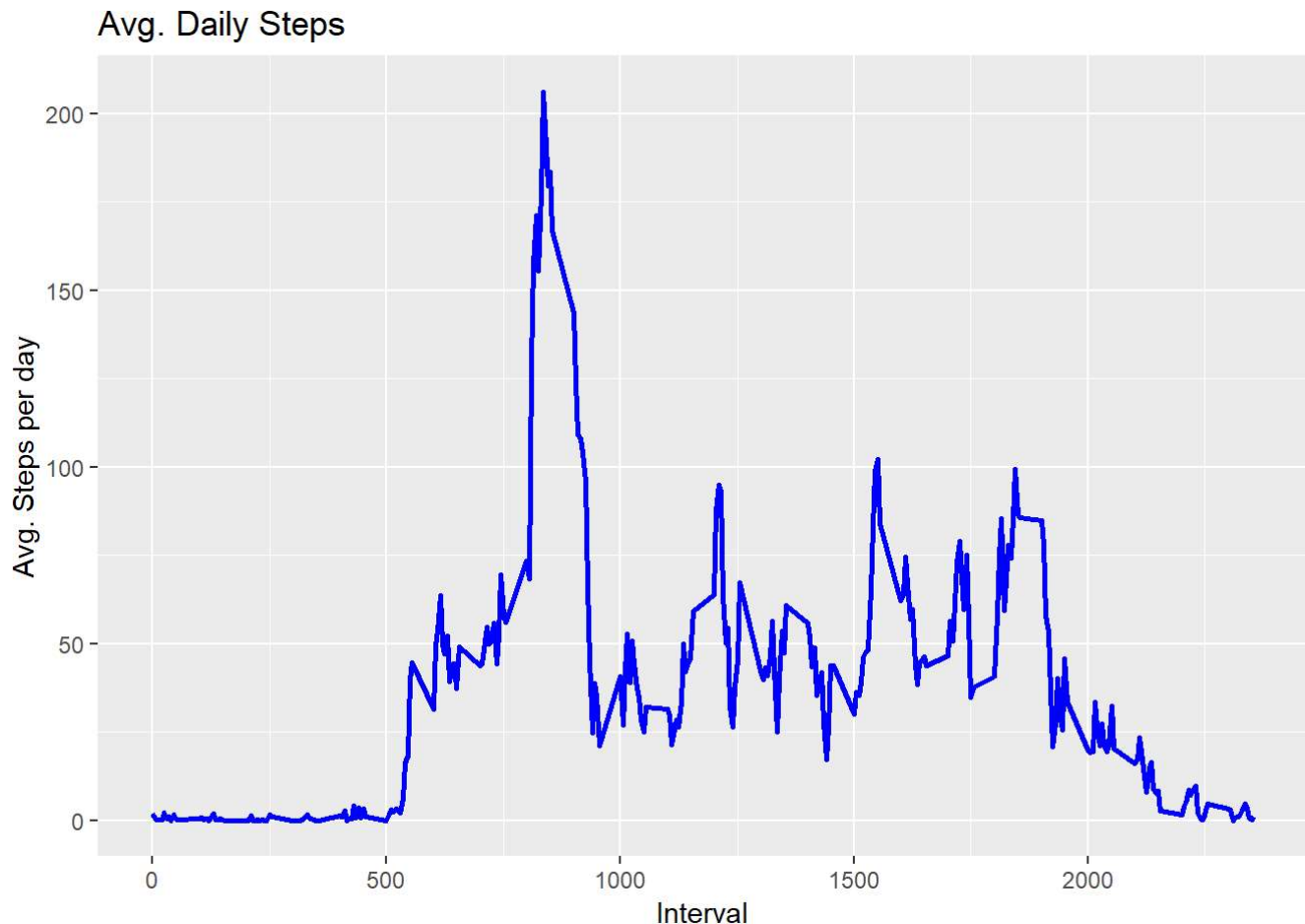
```
Total_Steps[, .(Mean_Steps = mean(steps, na.rm = TRUE), Median_Steps = median(steps, na.rm = TRUE))]
```

	Mean_Steps <dbl>	Median_Steps <int>
1 row	10766.19	10765

What is the average daily activity pattern?

1. Make a time series plot (i.e. `type = "l"`) of the 5-minute interval (x-axis) and the average number of steps taken, averaged across all days (y-axis)

```
IntervalDT <- activityDT[, c(lapply(.SD, mean, na.rm = TRUE)), .SDcols = c("steps"), by = .(interval)]
ggplot(IntervalDT, aes(x = interval , y = steps)) + geom_line(color="blue", size=1) + labs(title = "Avg. Daily Steps", x = "Interval", y = "Avg. Steps per day")
```



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

```
IntervalDT[steps == max(steps), .(max_interval = interval)]
```

**max\_interval**  
<int>

835

1 row

## Imputing missing values

1. Calculate and report the total number of missing values in the dataset (i.e. the total number of rows with NAs)

```
activityDT[is.na(steps), .N ]
```

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

```
# alternative solution
nrow(activityDT[is.na(steps),])
```

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

2. Devise a strategy for filling in all of the missing values in the dataset. The strategy does not need to be sophisticated. For example, you could use the mean/median for that day, or the mean for that 5-minute interval, etc.

```
# Filling in missing values with median of dataset.
activityDT[is.na(steps), "steps"] <- activityDT[, c(lapply(.SD, median, na.rm = TRUE)), .SDcols
= c("steps")]
```

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

```
data.table::fwrite(x = activityDT, file = "data/tidyData.csv", quote = FALSE)
```

4. Make a histogram of the total number of steps taken each day and calculate and report the mean and median total number of steps taken per day. Do these values differ from the estimates from the first part of the assignment? What is the impact of imputing missing data on the estimates of the total daily number of steps?

```
# total number of steps taken per day
Total_Steps <- activityDT[, c(lapply(.SD, sum)), .SDcols = c("steps"), by = .(date)]
# mean and median total number of steps taken per day
Total_Steps[, .(Mean_Steps = mean(steps), Median_Steps = median(steps))]
```

**Mean\_Steps**  
<dbl>

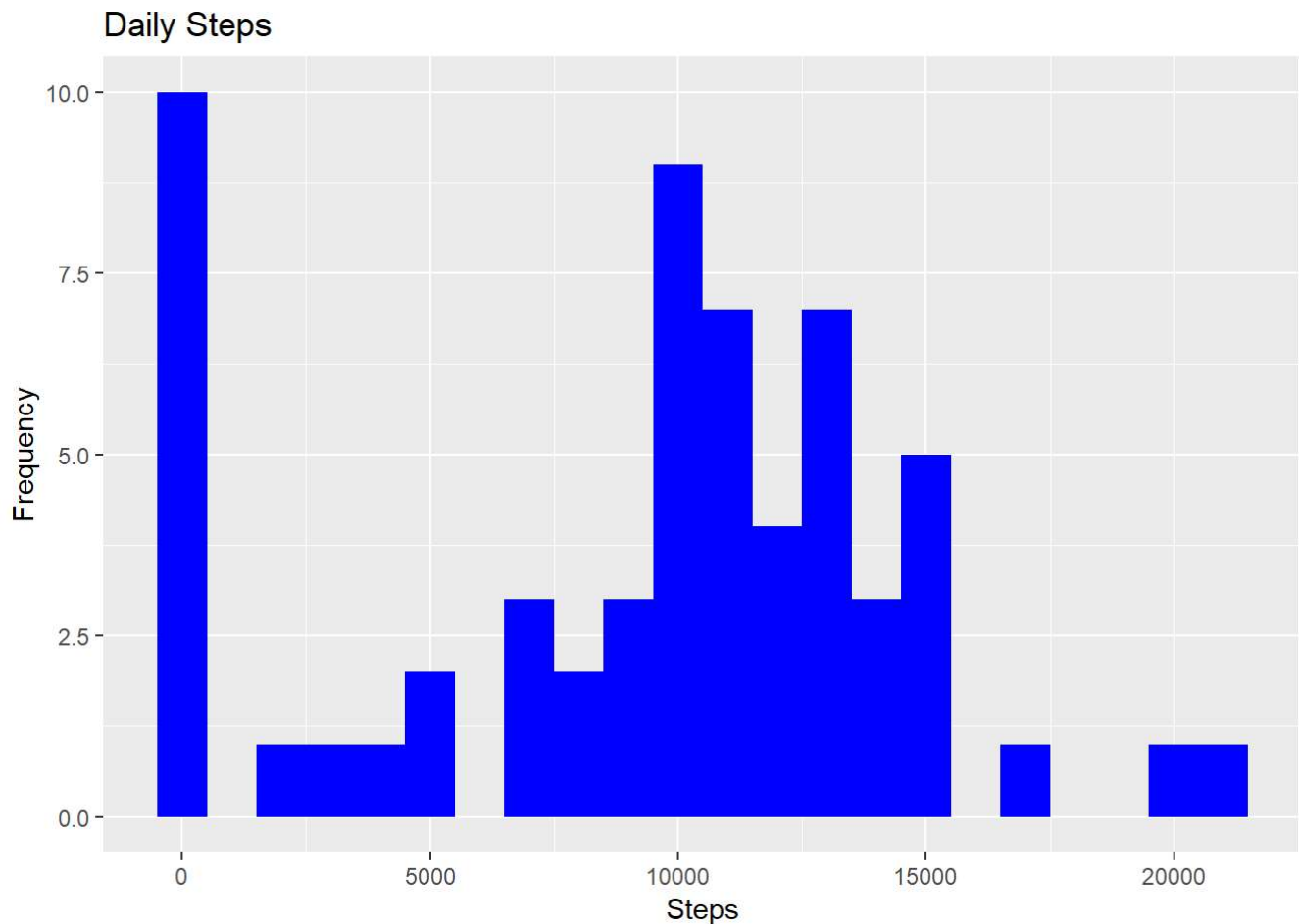
**Median\_Steps**  
<int>

9354.23

10395

1 row

```
ggplot(Total_Steps, aes(x = steps)) + geom_histogram(fill = "blue", binwidth = 1000) + labs(title = "Daily Steps", x = "Steps", y = "Frequency")
```



Type of Estimate	Mean_Steps	Median_Steps
First Part (with na)	10765	10765
Second Part (fillin in na with median)	9354.23	10395

## 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.

```
# Just recreating activityDT from scratch then making the new factor variable. (No need to, just want to be clear on what the entire process is.)
activityDT <- data.table::fread(input = "data/activity.csv")
activityDT[, date := as.POSIXct(date, format = "%Y-%m-%d")]
activityDT[, `Day of Week` := weekdays(x = date)]
activityDT[grepl(pattern = "Monday|Tuesday|Wednesday|Thursday|Friday", x = `Day of Week`), "weekday or weekend"] <- "weekday"
activityDT[grepl(pattern = "Saturday|Sunday", x = `Day of Week`), "weekday or weekend"] <- "weekend"
activityDT[, `weekday or weekend` := as.factor(`weekday or weekend`)]
head(activityDT, 10)
```

steps <int>	date <S3: POSIXct>	interval <int>	Day of Week <chr>	weekday or weekend <fctr>
NA	2012-10-01	0	Monday	weekday
NA	2012-10-01	5	Monday	weekday
NA	2012-10-01	10	Monday	weekday
NA	2012-10-01	15	Monday	weekday
NA	2012-10-01	20	Monday	weekday
NA	2012-10-01	25	Monday	weekday
NA	2012-10-01	30	Monday	weekday
NA	2012-10-01	35	Monday	weekday
NA	2012-10-01	40	Monday	weekday
NA	2012-10-01	45	Monday	weekday

1-10 of 10 rows

2. Make a panel plot containing a time series plot (i.e. type = "l") of the 5-minute interval (x-axis) and the average number of steps taken, averaged across all weekday days or weekend days (y-axis). See the README file in the GitHub repository to see an example of what this plot should look like using simulated data.

```
activityDT[is.na(steps), "steps"] <- activityDT[, c(lapply(.SD, median, na.rm = TRUE)), .SDcols = c("steps")]
IntervalDT <- activityDT[, c(lapply(.SD, mean, na.rm = TRUE)), .SDcols = c("steps"), by = .(interval, `weekday or weekend`)]
ggplot(IntervalDT, aes(x = interval, y = steps, color = `weekday or weekend`)) + geom_line() + labs(title = "Avg. Daily Steps by Weektype", x = "Interval", y = "No. of Steps") + facet_wrap(~`weekday or weekend`, ncol = 1, nrow = 2)
```

Avg. Daily Steps by Weektype

