Reproducible Research: Peer Assessment 1

Loading & preprocessing the data.

```
unzip(zipfile = "repdata-data-activity.zip")
dataRead <- read.csv("activity.csv")
```

- View the loaded data frame.

```
names(dataRead)
```

```
## [1] "steps"    "date"     "interval"
```

```
str(dataRead)
```

```
## '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 ...
```

```
head(dataRead, 10)
```

```
##    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
## 7     NA 2012-10-01       30
## 8     NA 2012-10-01       35
## 9     NA 2012-10-01       40
## 10    NA 2012-10-01       45
```

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

```
# Subset data frame to values without na for next process.
data_Without_NA <- dataRead[complete.cases(dataRead),]
```

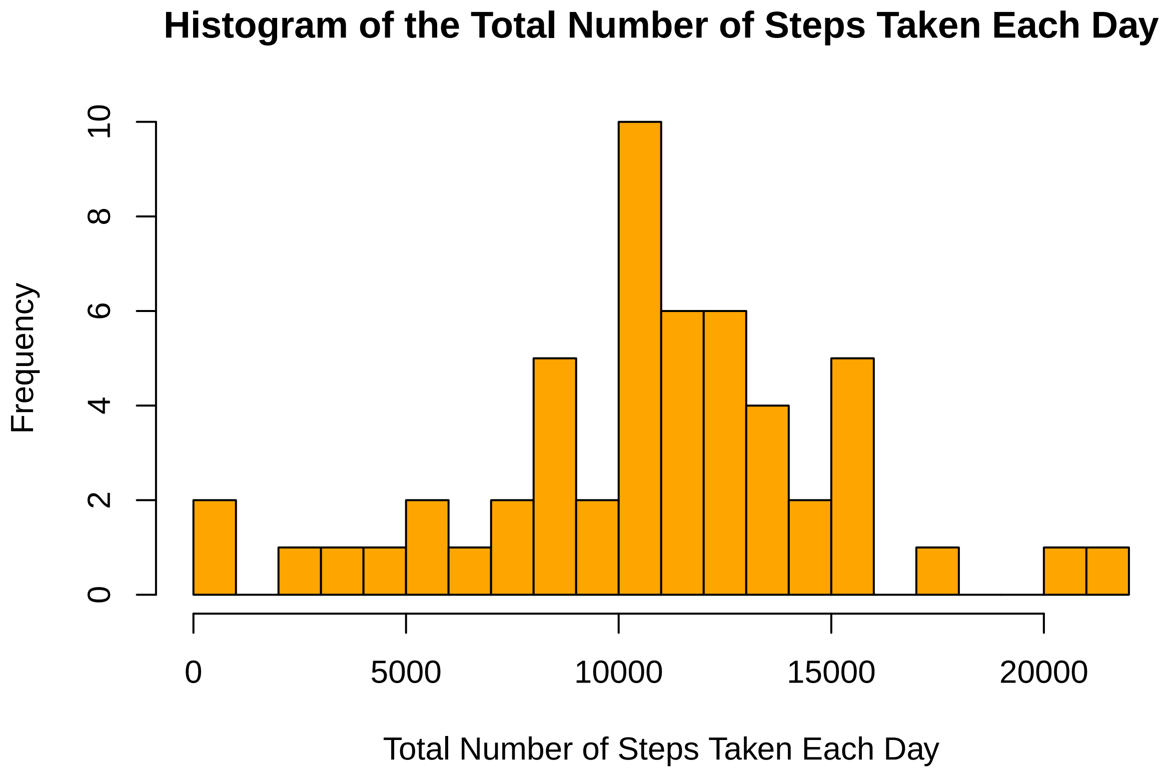
What is mean total number of steps taken per day?

- Plot a histogram of the total number of steps taken each day.

```r
# Find out the total steps taken per day.
totalSteps <- aggregate(steps ~ date, data_Without_NA, sum)

# Put the descriptive variable names in data frame.
names(totalSteps)[2] <- "sum_steps"

# View new created data frame.
head(totalSteps, 10)
```

```
##          date sum_steps
## 1  2012-10-02       126
## 2  2012-10-03     11352
## 3  2012-10-04     12116
## 4  2012-10-05     13294
## 5  2012-10-06     15420
## 6  2012-10-07     11015
## 7  2012-10-09     12811
## 8  2012-10-10      9900
## 9  2012-10-11     10304
## 10 2012-10-12     17382
```

```r
# Plot histogram of the total steps taken per day.
hist(
        totalSteps$sum_steps,
        col = "orange",
        main = "Histogram of the Total Number of Steps Taken Each Day",
        xlab = "Total Number of Steps Taken Each Day",
        breaks = 30
)
```



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

```r
mean(totalSteps$sum_steps)
```

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

```r
median(totalSteps$sum_steps)
```

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

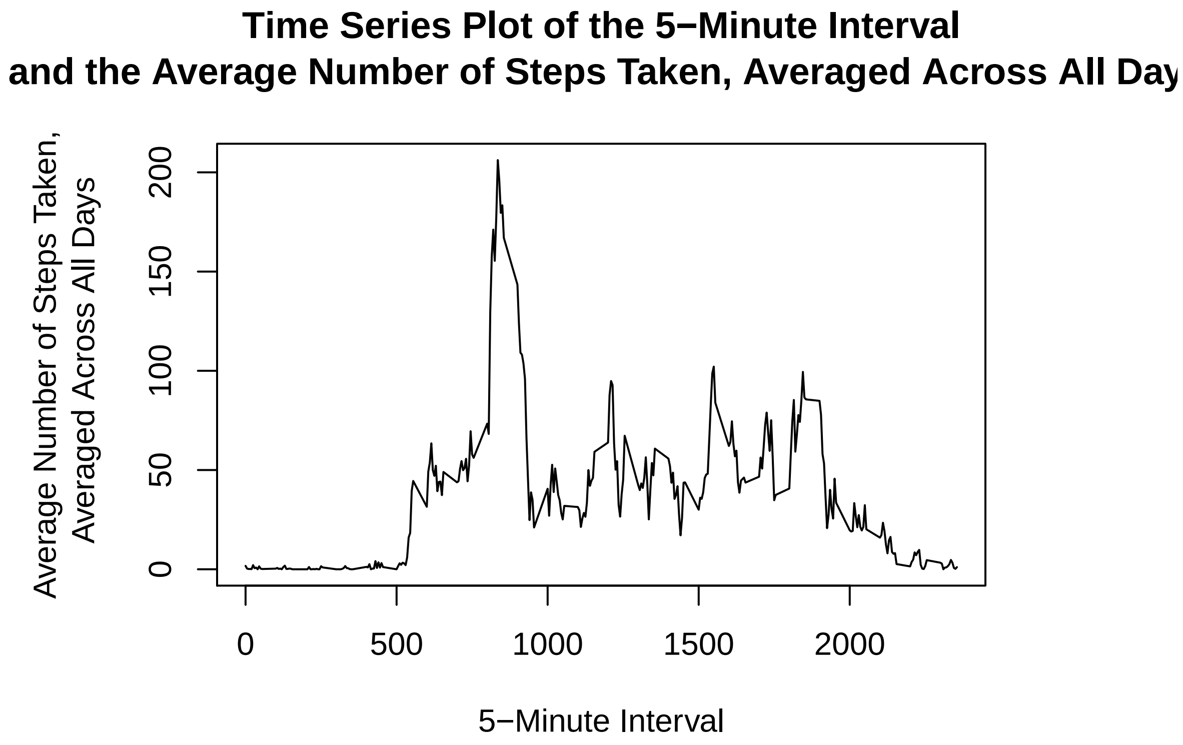
What is the average daily activity pattern?

- 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)

```r
# the average number of steps taken, averaged across all days for each 5-minute
# Find the "interval" dataframe.
interval_dataFrame <- aggregate(steps ~ interval, data_Without_NA, mean)

# Put the descriptive variable names in data frame.
names(interval_dataFrame)[2] <- "mean_steps"

# View new created data frame.
head(interval_dataFrame, 10)
```

```
##    interval mean_steps
## 1         0  1.7169811
## 2         5  0.3396226
## 3        10  0.1320755
## 4        15  0.1509434
## 5        20  0.0754717
## 6        25  2.0943396
## 7        30  0.5283019
## 8        35  0.8679245
## 9        40  0.0000000
## 10       45  1.4716981
```

```
# Format plot margins (bottom, left, top, right) for long text labels.
par( mai = c(1.2, 1.5, 1,1) )

# Plot time series.
plot(
        x = interval_dataFrame$interval,
        y = interval_dataFrame$mean_steps,
        type = "l",
        main = "Time Series Plot of the 5-Minute Interval\n and the Average Number of Steps Taken, Avera
        xlab = "5-Minute Interval",
        ylab = "Average Number of Steps Taken,\n Averaged Across All Days"
)
```



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

```
# Find out the maximum steps.
interval_dataFrame[ interval_dataFrame$mean_steps == max(interval_dataFrame$mean_steps), ]
```

```
##     interval mean_steps
## 104      835   206.1698
```

Imputing missing values

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

```
missing <- is.na(dataRead$steps)

# How many missing
table(missing)
```

```
## missing
## FALSE  TRUE
## 15264  2304
```

- 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.

```
# Use the mean for the 5-minute interval to simulate NA values for a given internval.
```

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

```
# First, merge the original activity data frame with interval data frame.
NA_filled <- merge(dataRead, interval_dataFrame, by = 'interval', all.y = F)

# Then, merge NA values with averages rounding up for integers.
NA_filled$steps[is.na(NA_filled$steps)] <- as.integer(round(NA_filled$mean_steps[is.na(NA_filled$steps)]

# Drop and reorder columns to match original activity data frame.
keeps <- names(dataRead)
NA_filled <- NA_filled[keeps]
head(NA_filled, 10)
```

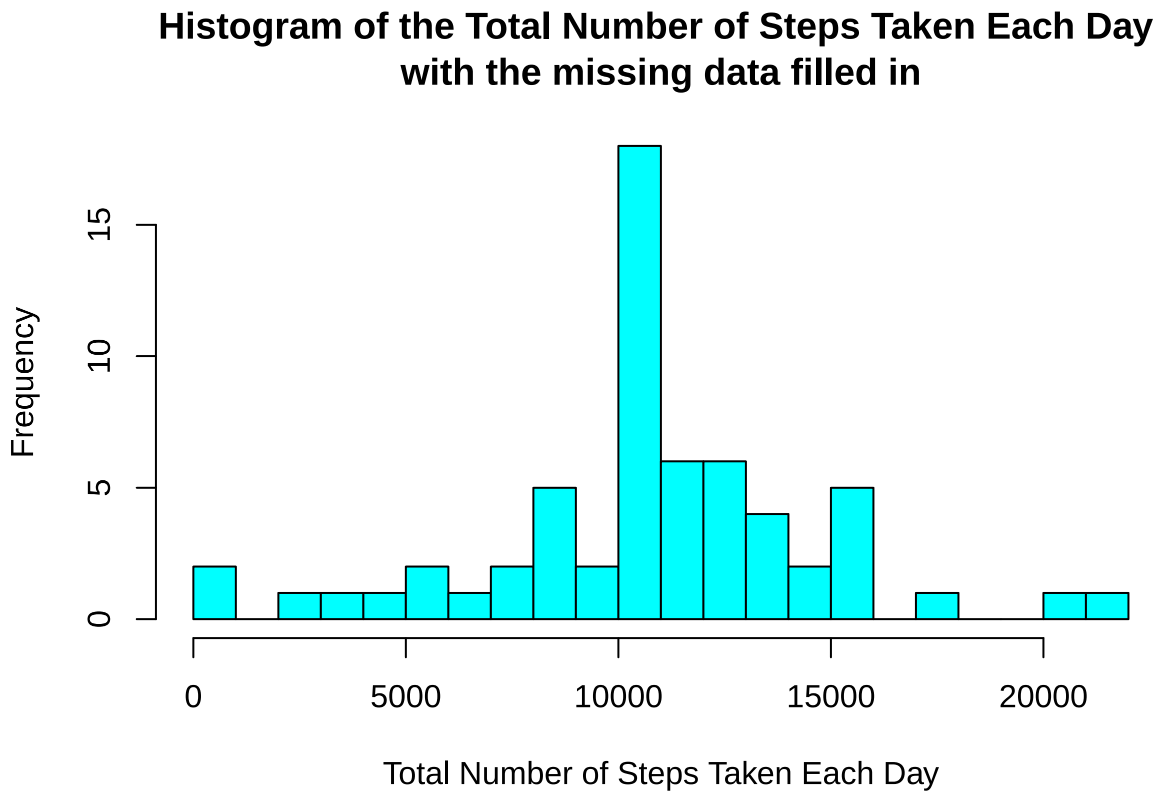
```
##    steps       date interval
## 1      2 2012-10-01        0
## 2      0 2012-11-23        0
## 3      0 2012-10-28        0
## 4      0 2012-11-06        0
## 5      0 2012-11-24        0
## 6      0 2012-11-15        0
## 7      0 2012-10-20        0
## 8      0 2012-11-16        0
## 9      0 2012-11-07        0
## 10     0 2012-11-25        0
```

- Make a histogram of the total number of steps taken each day

```r
# Find out the total number of steps taken per day with filled NA value.
newTotal <- aggregate(steps ~ date, NA_filled, sum)

# Put in the descriptive variable names in the newTotal data frame.
names(newTotal)[2] <- "sum_steps"

# Take a glance on this new data frame.
head(newTotal, 10)
```

```
##          date sum_steps
## 1  2012-10-01     10762
## 2  2012-10-02       126
## 3  2012-10-03     11352
## 4  2012-10-04     12116
## 5  2012-10-05     13294
## 6  2012-10-06     15420
## 7  2012-10-07     11015
## 8  2012-10-08     10762
## 9  2012-10-09     12811
## 10 2012-10-10      9900
```

```r
# Plot the histogram based on new data frame.
hist(
        newTotal$sum_steps,
        col = "cyan",
        main = "Histogram of the Total Number of Steps Taken Each Day \nwith the missing data filled in
        xlab = "Total Number of Steps Taken Each Day",
        breaks = 30
)
```



```
# Mean of this new filled data frame.
mean(newTotal$sum_steps)
```

```
## [1] 10765.64
```

```
# Median of this new filled data frame.
median(newTotal$sum_steps)
```

```
## [1] 10762
```

- Do these values differ from the estimates from the first part of the assignment?

```
# It is a subtle difference in Mean calculation between two parts of the assignment;
# Mean = 10766.19 (original data frame) and 10765.64 (data frame filled with NA),

# But quite apparently difference in Median calculation between these two parts of the assignment.
# Median = 10765 (original data frame) and 10762 (data frame filled with NA),
```

- What is the impact of imputing missing data on the estimates of the total daily number of steps?

```
# The impact is depend on the imputing level of the missing data.
# As shown in the experimental value, there was practically no much difference
# when using the average for a given interval as the averages is basically
# pulled towards to the inserted average value.
```

Are there differences in activity patterns between weekdays and weekends?

- 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.

```
# Create a new data frame.
newDataFrame <- NA_filled

# Prepare for up logical/test vector.
weekend <- weekdays( as.Date(newDataFrame$date)) %in% c("Saturday", "Sunday" )

# Fill in weekday column.
newDataFrame$daytype <- "weekday"

# Subsitute "weekday" with "weekend" where day == Sat/Sun.
newDataFrame$daytype[weekend == TRUE] <- "weekend"

# Convert new character column to factor.
newDataFrame$daytype <- as.factor(newDataFrame$daytype)

# Display the new data frame.
str(newDataFrame)
```

```
## 'data.frame':    17568 obs. of  4 variables:
##  $ steps   : int  2 0 0 0 0 0 0 0 0 0 ...
##  $ date    : Factor w/ 61 levels "2012-10-01","2012-10-02",..: 1 54 28 37 55 46 20 47 38 56 ...
##  $ interval: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ daytype : Factor w/ 2 levels "weekday","weekend": 1 1 2 1 2 1 2 1 1 2 ...
```

```
head(newDataFrame, 10)
```

```
##    steps       date interval daytype
## 1      2 2012-10-01        0 weekday
## 2      0 2012-11-23        0 weekday
## 3      0 2012-10-28        0 weekend
## 4      0 2012-11-06        0 weekday
## 5      0 2012-11-24        0 weekend
## 6      0 2012-11-15        0 weekday
## 7      0 2012-10-20        0 weekend
## 8      0 2012-11-16        0 weekday
## 9      0 2012-11-07        0 weekday
## 10     0 2012-11-25        0 weekend
```

```
# Verify the outcome.
weekdays( as.Date(newDataFrame$date[3]) )
```

```
## [1] "Sunday"
```

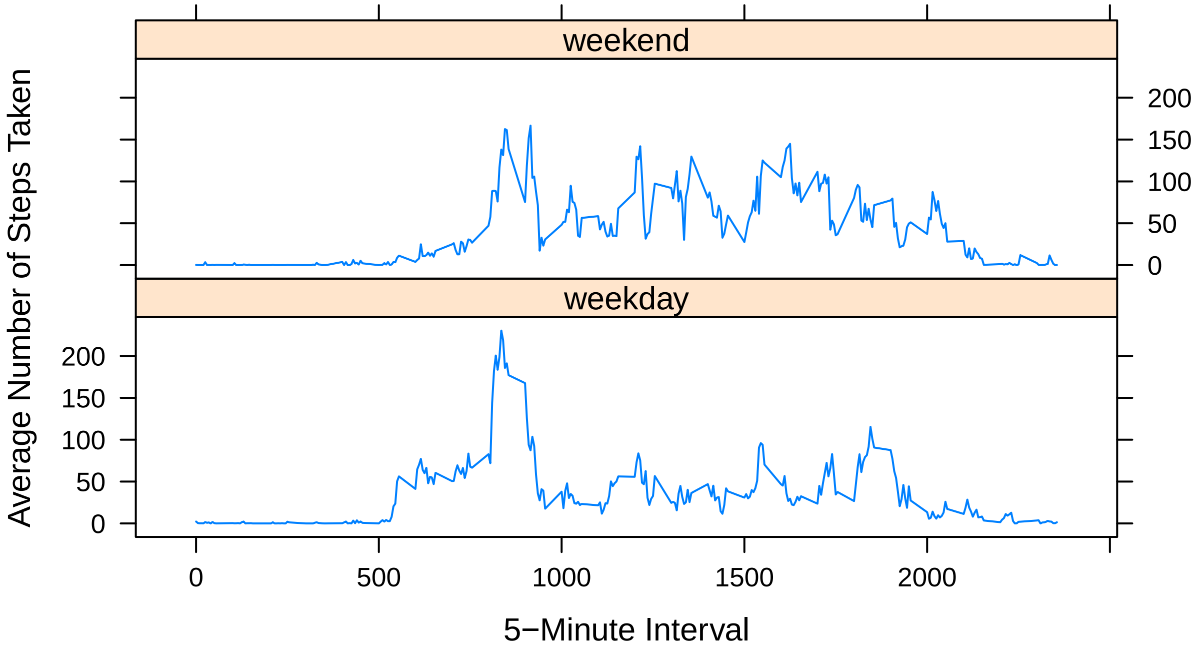
Make a panel plot containing a time series plot (i.e. type = "l") of the 5-minute interval (x-axis) and the average number of steps taken, averaged across all weekday days or weekend days (y-axis).

```
# the average number of steps taken, averaged across all days for each 5-minute interval.
new_Interval <- aggregate(steps ~ interval + daytype, newDataFrame, mean)

## Add descriptive variable names for easily understanding.
names(new_Interval)[3] <- "mean_steps"

# Display the new data frame.
head(new_Interval, 10)
```

```
##    interval daytype mean_steps
## 1         0 weekday 2.28888889
## 2         5 weekday 0.40000000
## 3        10 weekday 0.15555556
## 4        15 weekday 0.17777778
## 5        20 weekday 0.08888889
## 6        25 weekday 1.57777778
## 7        30 weekday 0.75555556
## 8        35 weekday 1.15555556
## 9        40 weekday 0.00000000
## 10       45 weekday 1.73333333
```

```
# Plot time series based on the new data frame.
library(lattice)
xyplot(
        mean_steps ~ interval | daytype,
        new_Interval,
        type = "l",
        layout = c(1,2),
        main = "Time Series Plot of the 5-Minute Interval\nand the Average Number of Steps Taken,\nAvera
        xlab = "5-Minute Interval",
        ylab = "Average Number of Steps Taken"
)
```

**Time Series Plot of the 5−Minute Interval
and the Average Number of Steps Taken,
Averaged Across All Weekday Days or Weekend Days**


weekend
weekday
Average Number of Steps Taken
5−Minute Interval