Reproducible Research: Peer Assessment 1

Loading and preprocessing the data

```
strUrl <-"https://d396qusza40orc.cloudfront.net/repdata%2Fdata%2Factivity.zip"
strFile <-"activity.zip"
if (!file.exists(strFile))
{
     download.file(url = strUrl, destfile = "activity.zip", method = "curl", mode = "wb")
}
lstOfFiles <-unzip(zipfile = strFile)
#
df <-read.csv(file = lstOfFiles[1], colClasses = c("integer","character","integer"))
#
str(df)
```

```
## 'data.frame':    17568 obs. of  3 variables:
##  $ steps   : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ date    : chr  "2012-10-01" "2012-10-01" "2012-10-01" "2012-10-01" ...
##  $ interval: int  0 5 10 15 20 25 30 35 40 45 ...
```

```
#
df$interval.ori <-df$interval
df$interval <-sprintf(fmt = "%04d", df$interval)
#
df$date.ori <-df$date
df$date <-as.Date(df$date)
#
df$dayofweek <-weekdays(df$date)
#
df$daytype <-"weekday"
df[substr(df$dayofweek,1,1) == "S", "daytype"] <-"weekend"
df$daytype <-factor(df$daytype, levels = c("weekday","weekend"))
#
str(df)
```

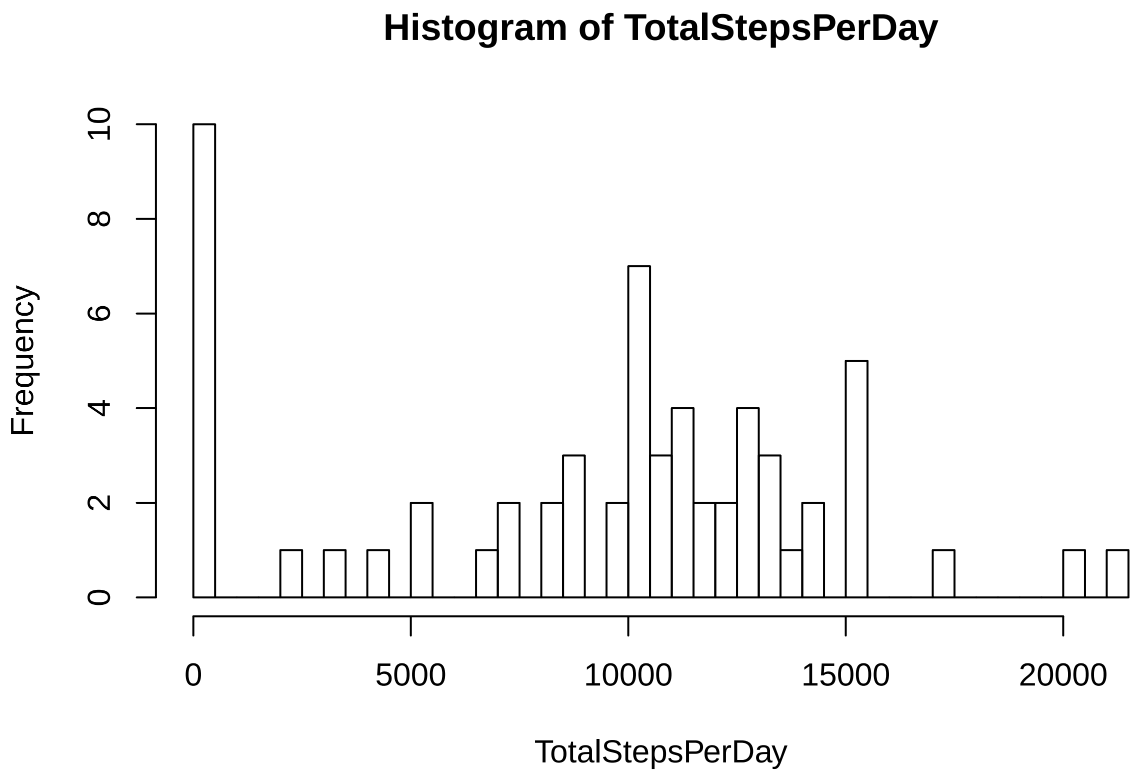
```
## 'data.frame':    17568 obs. of  7 variables:
##  $ steps       : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ date        : Date, format: "2012-10-01" "2012-10-01" ...
##  $ interval    : chr  "0000" "0005" "0010" "0015" ...
##  $ interval.ori: int  0 5 10 15 20 25 30 35 40 45 ...
##  $ date.ori    : chr  "2012-10-01" "2012-10-01" "2012-10-01" "2012-10-01" ...
##  $ dayofweek   : chr  "Monday" "Monday" "Monday" "Monday" ...
##  $ daytype     : Factor w/ 2 levels "weekday","weekend": 1 1 1 1 1 1 1 1 1 1 ...
```

What is mean total number of steps taken per day?

```
# 1. Calculate the total number of steps taken per day
TotalStepsPerDay <-tapply(X = df$steps, INDEX = df$date, FUN = sum, na.rm = TRUE)
head(TotalStepsPerDay, n = 25)
```

```
## 2012-10-01 2012-10-02 2012-10-03 2012-10-04 2012-10-05 2012-10-06
##          0        126      11352      12116      13294      15420
## 2012-10-07 2012-10-08 2012-10-09 2012-10-10 2012-10-11 2012-10-12
##      11015          0      12811       9900      10304      17382
## 2012-10-13 2012-10-14 2012-10-15 2012-10-16 2012-10-17 2012-10-18
##      12426      15098      10139      15084      13452      10056
## 2012-10-19 2012-10-20 2012-10-21 2012-10-22 2012-10-23 2012-10-24
##      11829      10395       8821      13460       8918       8355
## 2012-10-25
##       2492
```

```
# 2. Make a histogram of the total number of steps taken each day.
hist(x = TotalStepsPerDay, breaks = 60)
```



```
# 3. Calculae and report the mean and the median of the total number of steps taken per day.
(meanTotalStepsPerDay <-mean(TotalStepsPerDay))
```

```
## [1] 9354.23
```

```
(medianTotalStepsPerDay <-median(TotalStepsPerDay))
```

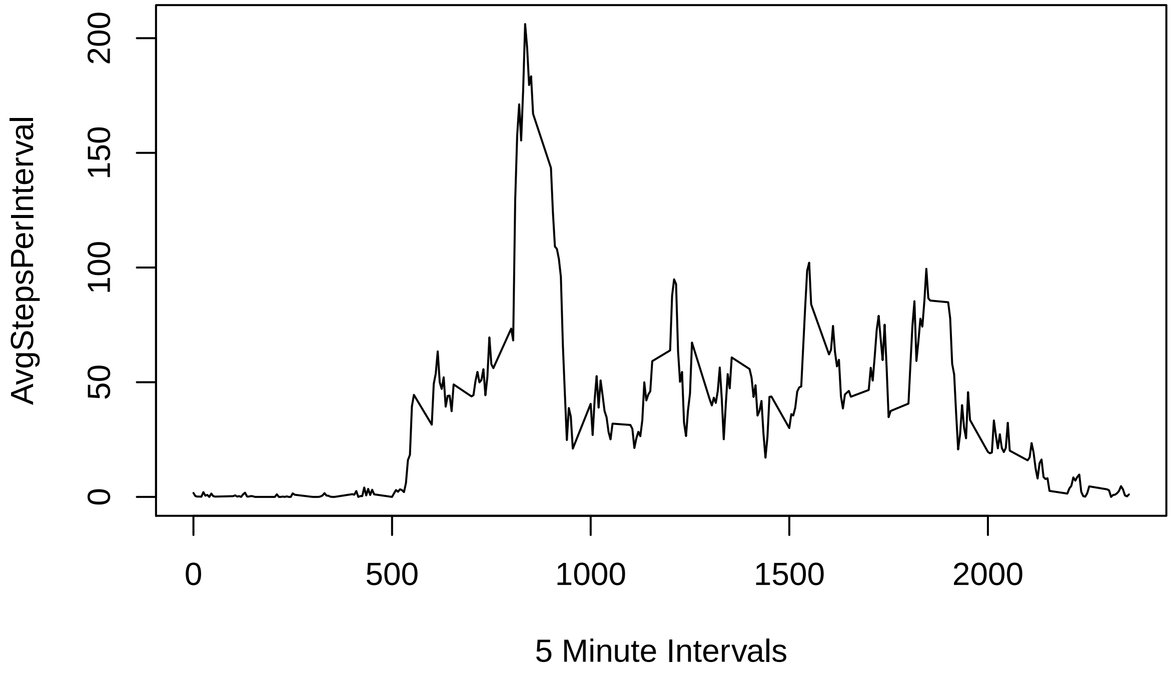
```
## [1] 10395
```

What is the average daily activity pattern?

```
AvgStepsPerInterval <-tapply(X = df$steps, INDEX = df$interval, FUN = mean, na.rm = TRUE)
head(AvgStepsPerInterval, n = 25)
```

```
##      0000      0005      0010      0015      0020      0025      0030
## 1.7169811 0.3396226 0.1320755 0.1509434 0.0754717 2.0943396 0.5283019
##      0035      0040      0045      0050      0055      0100      0105
## 0.8679245 0.0000000 1.4716981 0.3018868 0.1320755 0.3207547 0.6792453
##      0110      0115      0120      0125      0130      0135      0140
## 0.1509434 0.3396226 0.0000000 1.1132075 1.8301887 0.1698113 0.1698113
##      0145      0150      0155      0200
## 0.3773585 0.2641509 0.0000000 0.0000000
```

```
# 1. Make a time series plot (i.e. type = "l") of the 5-minute interval (x-axis) and the average number
plot(names(AvgStepsPerInterval), AvgStepsPerInterval,type = "l", xlab = "5 Minute Intervals")
```



```
# 2. Which 5-minute interval, on average across all the days in the dataset, contains the maximum numbe
(maxAvgStepsPerInterval <-AvgStepsPerInterval[AvgStepsPerInterval == max(AvgStepsPerInterval)])
```

```
##     0835
## 206.1698
```

Imputing missing values

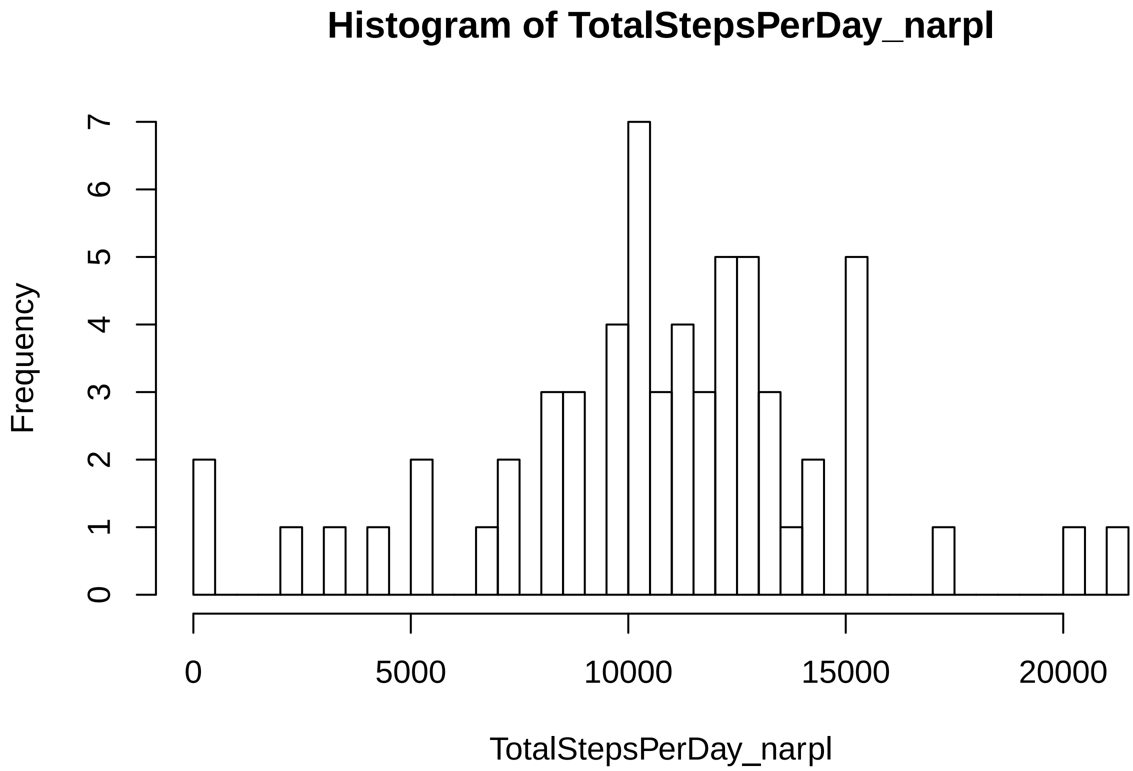
```
# 1. Calculate and report the total number of missing values in the dataset (i.e. the total number of r
sum(!complete.cases(df))
```

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

```
sum(is.na(df$steps))
```

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

```
# 2. Devise a strategy for filling in all of the missing values in the dataset.
# create a matrix of avg steps per interval (row name) by day of week (column name)
mxAvgStepsPerIntervalByDow <-tapply(X = df$steps, INDEX = list(df$interval,df$dayofweek), mean, na.rm =
# 3. Create a new dataset that is equal to the original dataset but with missing data filled in.
df_narpl <-df
# make copy of original steps variable
df_narpl$steps.ori <-df_narpl$steps
# loop through data frame "df_narpl" and replace NA steps with avg steps from replacement matix
for (i in 1:nrow(df_narpl))
{
     isteps <-df_narpl[i,"steps"]
     iinterval <-df_narpl[i,"interval"]
     idayofweek <-df_narpl[i,"dayofweek"]
     # if steps is missing, replace with avg from dfcmatrix
     if(is.na(isteps))
     {
          df_narpl[i,"steps"] <-mxAvgStepsPerIntervalByDow[iinterval,idayofweek]
     }
}
# 4. Make a histogram of the total number of steps taken each day and calculate the mean and median tot
TotalStepsPerDay_narpl <-tapply(X = df_narpl$steps, INDEX = df_narpl$date, FUN = sum, na.rm = TRUE)
# histogram
hist(x = TotalStepsPerDay_narpl, breaks = 60)
```



```
# mean and median
meanTotalStepsPerDay_narpl <-mean(TotalStepsPerDay_narpl)
medianTotalStepsPerDay_narpl <-median(TotalStepsPerDay_narpl)
# compare with mean where incomplete records (NAs) were removed
(meanDiff <-meanTotalStepsPerDay_narpl - meanTotalStepsPerDay)
```

```
## [1] 1466.98
```

```
# compare with median where incomplete records (NAs) were removed
(medianDiff <-medianTotalStepsPerDay_narpl - medianTotalStepsPerDay)
```

```
## [1] 620
```

Are there differences in activity patterns between weekdays and weekends?

```
df_narpl$dayofweek <-weekdays(df_narpl$date)
# 1. Create a new factor variable in the dataset with two levels - "weekday" and "weekend" indicating wh
df_narpl$daytype <-"weekday"
df_narpl[substr(df_narpl$dayofweek,1,1) == "S", "daytype"] <-"weekend"
df_narpl$daytype <-factor(df_narpl$daytype)
# 2. Make a panel plot containg a time serie plot (i.e. type = "l") of the 5-minute interval (x-axis) a
# create a matrix of average number of steps taken per interval per daytype (weekday & weekend)
mxAvgStepsPerIntervalByDayType <-tapply(X = df_narpl$steps, INDEX = list(df_narpl$interval, df_narpl$day
str(mxAvgStepsPerIntervalByDayType)
```

```
##  num [1:288, 1:2] 2.3107 0.45 0.175 0.2 0.0889 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:288] "0000" "0005" "0010" "0015" ...
##   ..$ : chr [1:2] "weekday" "weekend"
```

```
# reshape the matrix to a data frame with 3 columns : interval, day type and avg steps per interval
require(reshape2)
```

```
## Loading required package: reshape2
```

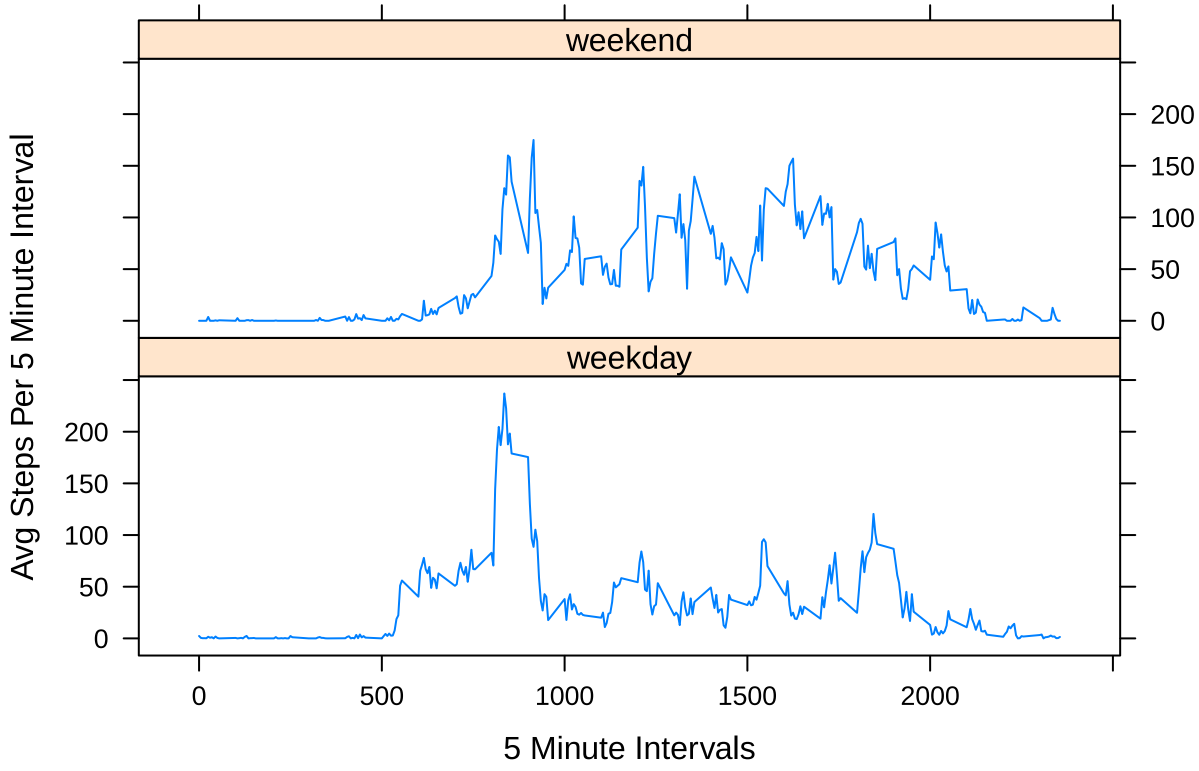
```
df_mlt <-melt(data = mxAvgStepsPerIntervalByDayType, measure.vars = c("weekday","weekend"), id.vars = c
# update column/variable names
names(df_mlt) <-c("timeInterval","dayType","avgSteps")
str(df_mlt)
```

```
## 'data.frame':    576 obs. of  3 variables:
##  $ timeInterval: int  0 5 10 15 20 25 30 35 40 45 ...
##  $ dayType     : Factor w/ 2 levels "weekday","weekend": 1 1 1 1 1 1 1 1 1 1 ...
##  $ avgSteps    : num  2.3107 0.45 0.175 0.2 0.0889 ...
```

```
# create panel plots of avg steps by interal for each day type (weekday & weekend)
require("lattice")
```

```
## Loading required package: lattice
```

```
xyplot(avgSteps ~ timeInterval | dayType, data = df_mlt, type = "l", layout = c(1,2), xlab = "5 Minute 
```


weekend
weekday
Avg Steps Per 5 Minute Interval
5 Minute Intervals