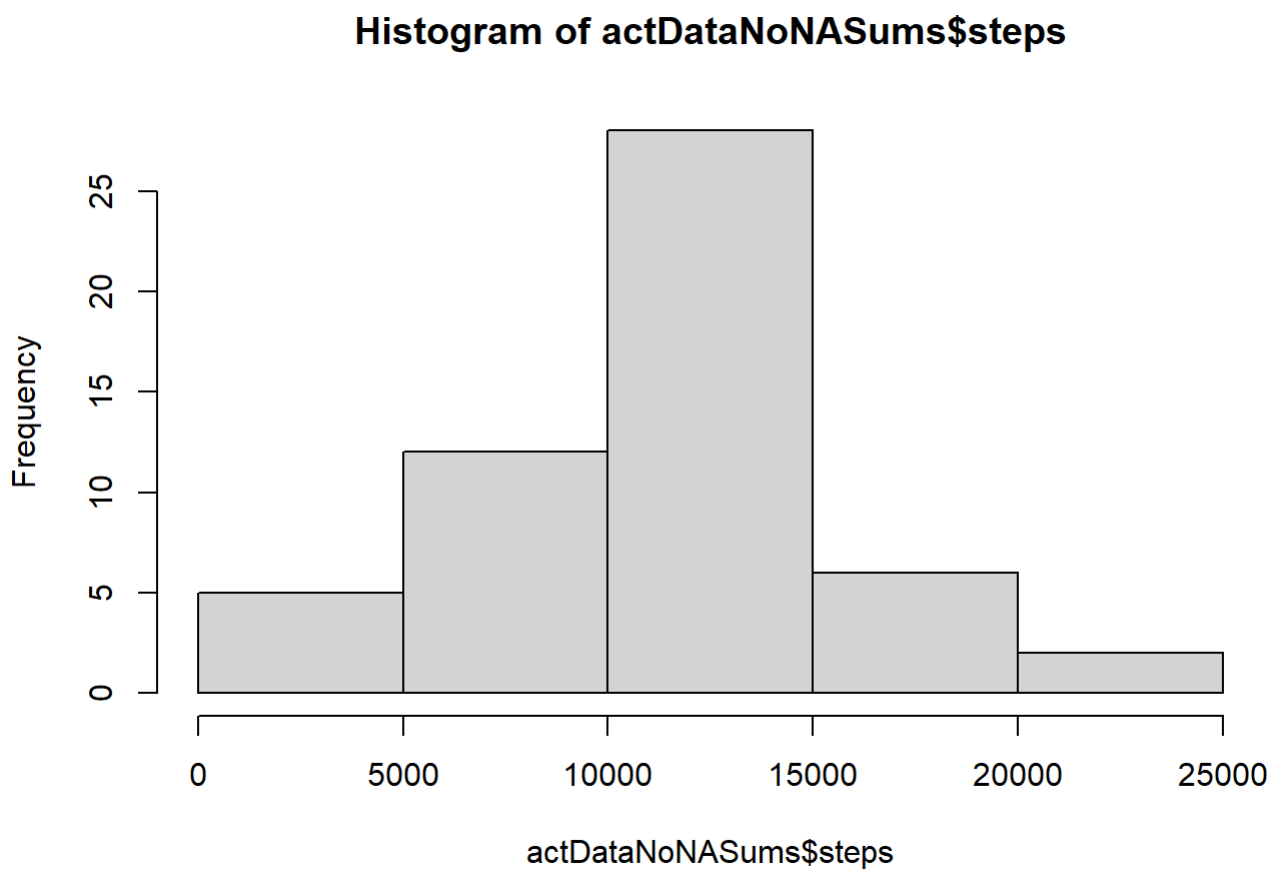
Reproducible Research: Peer Assessment 1

Gerard M Dion

9/13/2020

```
## Loading and preprocessing the data
##
## 1. Load the data (i.e. read.csv())
##
## 2. Process/transform the data (if necessary) into a format
## suitable for your analysis

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

# transform the intervals to a number between 0 and 1440, the minutes in one day
# correpsonding to raw values of 0, 5,...,55, 100,   where 100 means
# 1:00 and 205 means 2:05 and so on
actData[nchar(actData$interval)==3,]$interval <- as.integer((60 * as.integer(substr(actData[ncha
r(actData$interval)==3,]$interval,1,1)) + as.integer(substr(actData[nchar(actData$interval)==3,]
$interval,2,3))))

actData[nchar(actData$interval)==4,]$interval <- as.integer((60 * as.integer(substr(actData[ncha
r(actData$interval)==4,]$interval,1,2)) + as.integer(substr(actData[nchar(actData$interval)==4,]
$interval,3,4))))

actDataNoNA <- na.omit(actData)
actDataNoNA$date <- as.POSIXct(actDataNoNA$date, format="%m/%d/%Y")
actDataNoNASums <- aggregate(steps ~ date, data=actDataNoNA, FUN=sum)
actDataNoNAMeans <- aggregate(steps ~ interval, data=actDataNoNA, FUN=mean)
```

```
## What is mean total number of steps taken per day?
##
## For this part of the assignment, you can ignore the missing
## values in the dataset.
##
## 1. Make a histogram of the total number of steps taken each
## day
##
## 2. Calculate and report the mean and median total number of
## steps taken per day

hist(actDataNoNASums$steps, breaks=4)
```




```
summary(actDataNoNASums$steps)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      41    8841   10765   10766   13294   21194
```

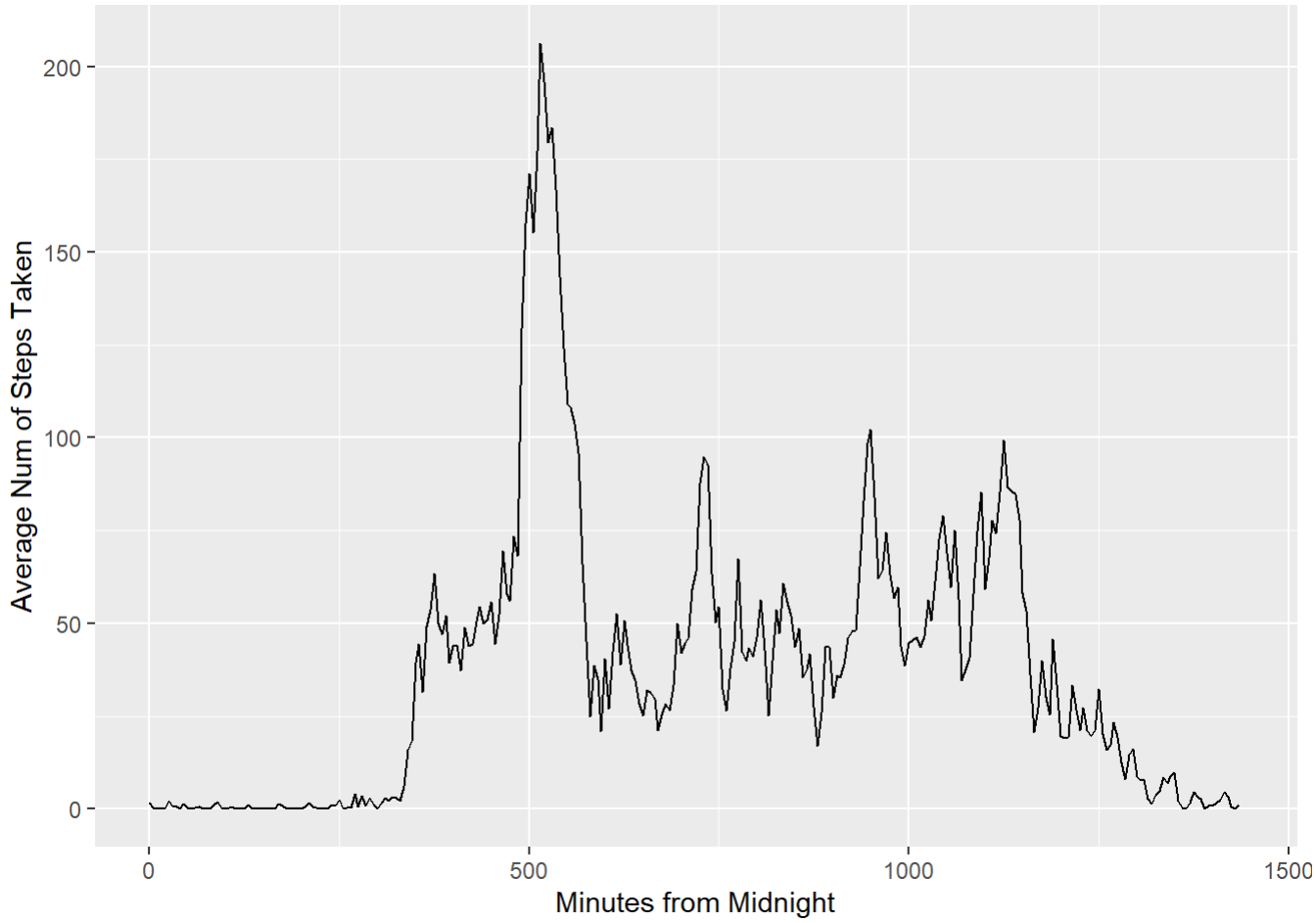
```
numDaysWithData <- nrow(actDataNoNASums)
meanSteps <- mean(actDataNoNASums$steps)
medianSteps <- median(actDataNoNASums$steps)
```

There were 53 days of data out of the total 2 months

The Mean Number of steps per day over the 53 days with values: 10766

The Median Number of steps per day over the 53 days with values: 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)
##
## 2. Which 5-minute interval, on average across all the days in
## the dataset, contains the maximum number of steps?

library(ggplot2)
stepsTimeSeriesPlot <- ggplot(actDataNoNAMeans, aes(interval, steps)) + geom_line() + xlab("Minu
tes from Midnight") + ylab("Average Num of Steps Taken") + xlim(0,1440)
print(stepsTimeSeriesPlot)
```




```
maxMeanSteps <- max(actDataNoNAMeans$steps)
maxMeanStepsInterval <- actDataNoNAMeans[actDataNoNAMeans$steps==max(actDataNoNAMeans$steps),]$i
nterval
```

The interval with the maximum mean/average number of steps is 515 with 206 steps

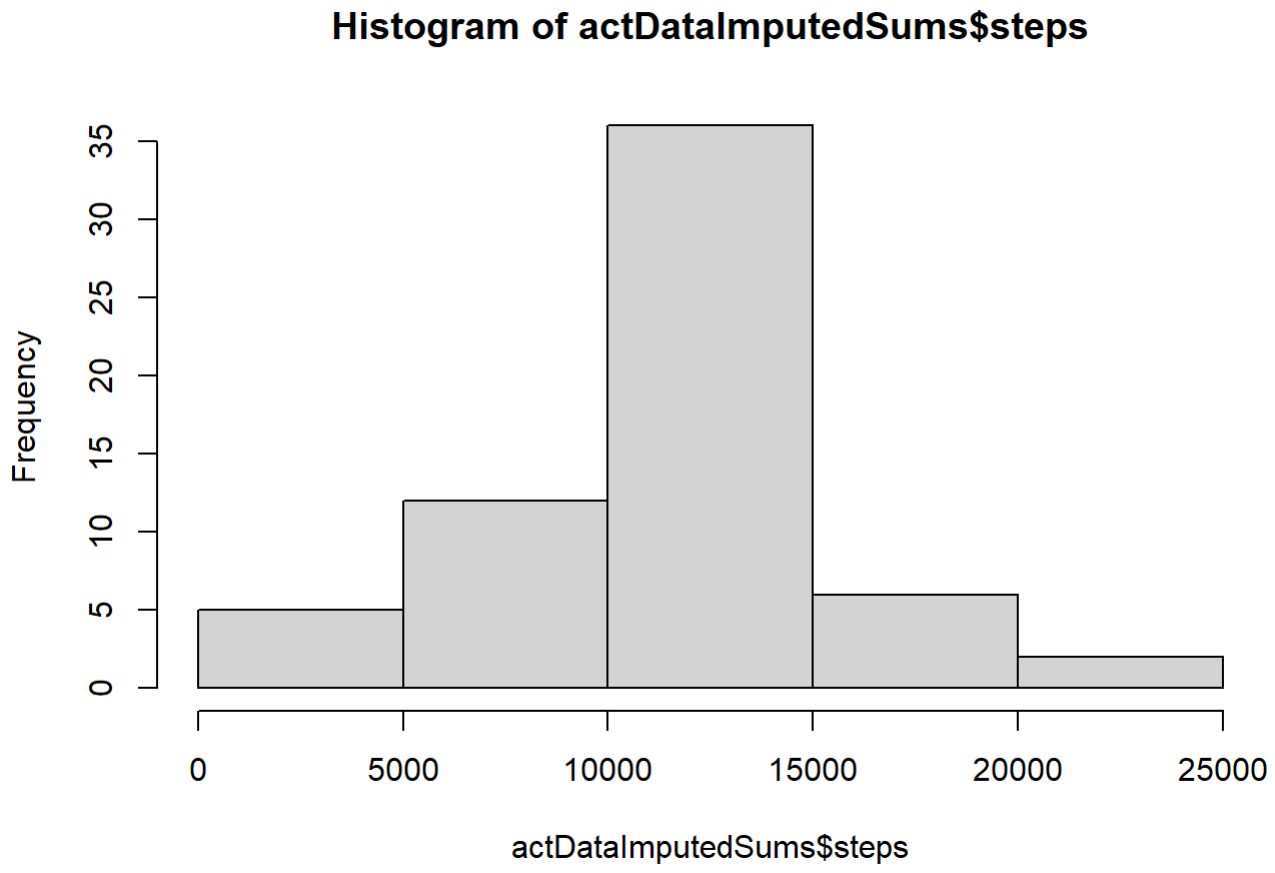
```
## Imputing missing values
##
## Note that there are a number of days/intervals where there are
## missing values (coded as NA). The presence of missing days may
## introduce bias into some calculations or summaries of the
## data.

## 1. Calculate and report the total number of missing values in
## dataset (i.e. the total number of rows with NAs)
numNArows <- nrow(actData) - nrow(actDataNoNA)

## 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 five-minute interval, etc.
#
## The strategy is to assign to the NA values of steps the mean from all
## the other days for that interval.  This is done for each of the 288
## calculated means for the intervals.
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
```

```
mergedOnInterval <- merge(actData,actDataNoNAMeans,by="interval")
# for the NA entries, replace the steps with that from the means.
mergedOnInterval[is.na(mergedOnInterval$steps.x),]$steps.x <- mergedOnInterval[is.na(mergedOnInt
erval$steps.x),]$steps.y
# the merge changed the ordering.  Re-order back to same as for original
mergedOnInterval <- arrange(mergedOnInterval,date,interval)



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

# replace the steps column with the new merged and NA replaced values
actDataImputed <- actData
actDataImputed$steps <- mergedOnInterval$steps.x

## 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?

actDataImputedSums <- aggregate(steps ~ date, data=actDataImputed, FUN=sum)

hist(actDataImputedSums$steps, breaks=4)
```



```
summary(actDataImputedSums$steps)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      41    9819   10766   10766   12811   21194
```

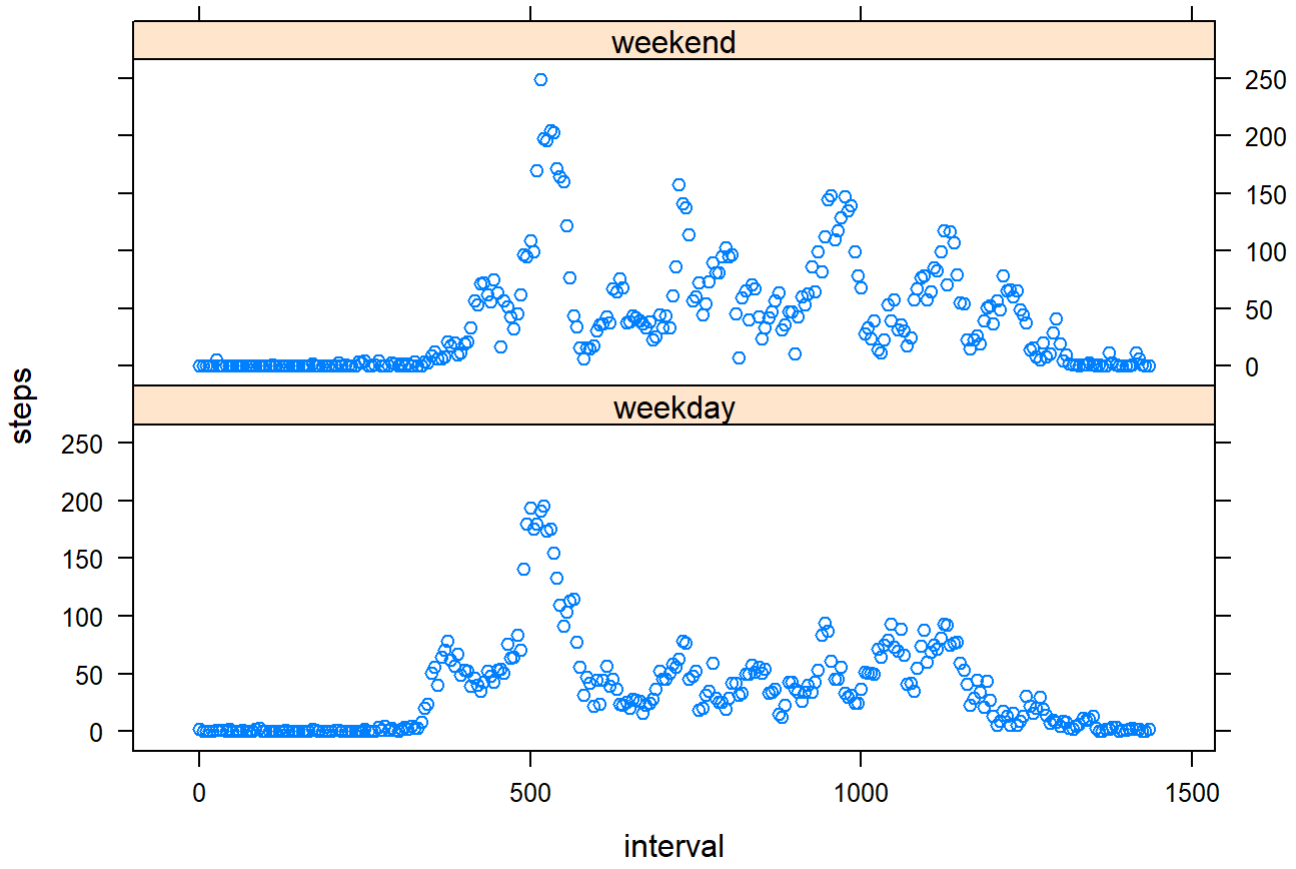
```
numDaysWithDataImputed <- nrow(actDataImputedSums)
meanStepsImputed <- mean(actDataImputedSums$steps)
medianStepsImputed <- median(actDataImputedSums$steps)
```

There were 61 days of data out of the total 2 months

The Mean Number of steps per day over the 61 days with values: 10766

The Median Number of steps per day over the 61 days with values: 10766

```
## Are there differences in activity patterns between weekdays and weekends?
##
## For this part the weekdays() function may be of some help here. Use the dataset
## with the filled-in missing values for this part.
##
## 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.


calculate_day_type <- function(x) {

        MTWRF <- c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday")
        SS <- c("Saturday", "Sunday")

        if (weekdays(as.Date(x,format="%m/%d/%Y")) %in% MTWRF) {
                return("weekday")
        } else if (weekdays(as.Date(x,format="%m/%d/%Y")) %in% SS) {
                return("weekend")
        }
        return(NA)
}

actDataImputed$dayType <- apply(actDataImputed, 1, function(x) calculate_day_type(x['date']))
actDataImputed$date <- as.Date(actDataImputed$date,format="%m/%d/%Y")
actDataImputed <- arrange(actDataImputed,date)

weekdayOnly <- aggregate(steps ~ interval, data=filter(actDataImputed,dayType=="weekday"), FUN=m
ean)
weekendOnly <- aggregate(steps ~ interval, data=filter(actDataImputed,dayType=="weekend"), FUN=m
ean)

weekdayOnly$dayType <- rep(c("weekday"),nrow(weekdayOnly))
weekendOnly$dayType <- rep(c("weekend"),nrow(weekendOnly))

plotData <- rbind(weekdayOnly,weekendOnly)
plotData$dayType <- as.factor(plotData$dayType)



##
## 2. Make a panel plot containing a time series plot (i.e. type = "l") of the
## five-minute interval (x-axis) and the average number of steps taken, averaged
## across all weekday days or weekend days (y-axis). The plot should look
## something like the one from the branch forked from RPeng.
## Your plot will look different from the one of RPeng because you will be using
## the activity monitor data. Note that the plot of RPeng was made using the lattice
## system but you can make the same version of the plot using any plotting system
## you choose.

library(lattice)
xyplot(steps ~ interval | dayType, data=plotData, layout = c(1,2))
```



There are only slight differences between weekday and weekend step counts The general activity is very similar, with approximately 4 to 5 peaks