Reproducible Research Peer Assessment 1

Sunday, February 15, 2015

Loading and preprocessing the data

The code below sets the global echo to true, so you can see all the code:

```
library(knitr)
opts_chunk$set(echo=TRUE, eval=TRUE)
```

Now, we'll load the data

```
fitData <- read.csv("activity.csv")
```

Data description: The data for this project is two months of data from a personal activity monitoring device, October through November of 2012. Includes the number of steps taken in five minute intervals each day. There are three variables in the raw data:

1. **steps**: Number of steps taken in a 5-minute interval.
2. **date**: The date of measurement, format: YYYY-MM-DD.
3. **interval**: Identifies the 5 minute interval for the day.

What is mean total number of steps taken per day?

For this part of the assignment, we were instructed to ignore missing values in the data.

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

I made a dataframe for the total steps by date.

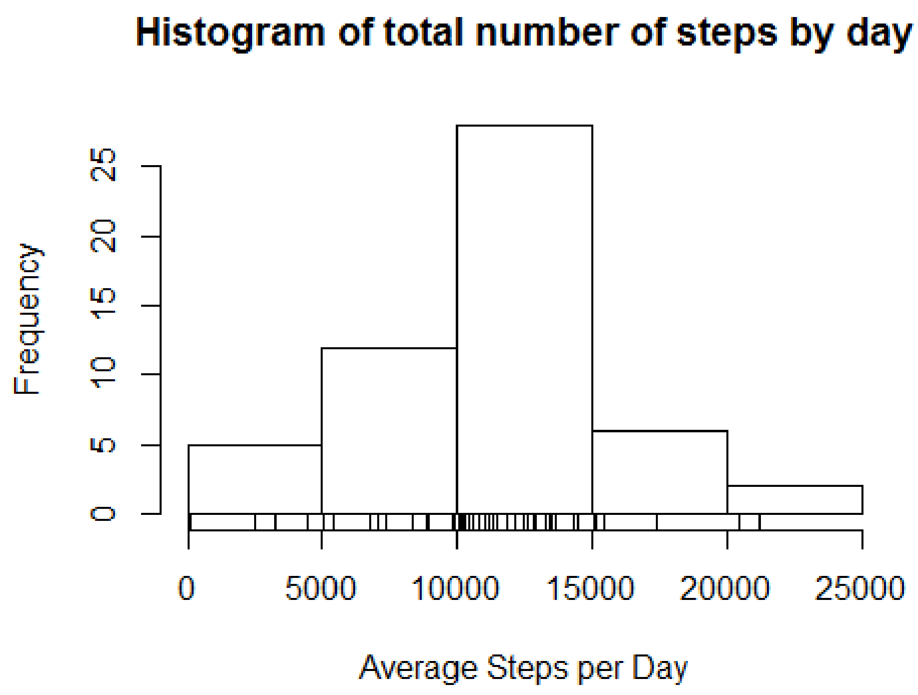
```
steps.by.date <- aggregate(. ~ date, data=fitData, FUN=sum)
## Just keep steps and date
steps.by.date <- steps.by.date[,1:2]
str(steps.by.date)

## 'data.frame':    53 obs. of  2 variables:
##  $ date : Factor w/ 61 levels "2012-10-01","2012-10-02",..: 2 3 4 5 6 7 9
10 11 12 ...
##  $ steps: int  126 11352 12116 13294 15420 11015 12811 9900 10304 17382
...
```

The first thing to note here is that there are 53 rather than 61 rows. This is the first hint that there are 8 days for which data was not collected.

2. **Make a histogram of the total number of steps taken each day**

This is the typical histogram, with a rug.

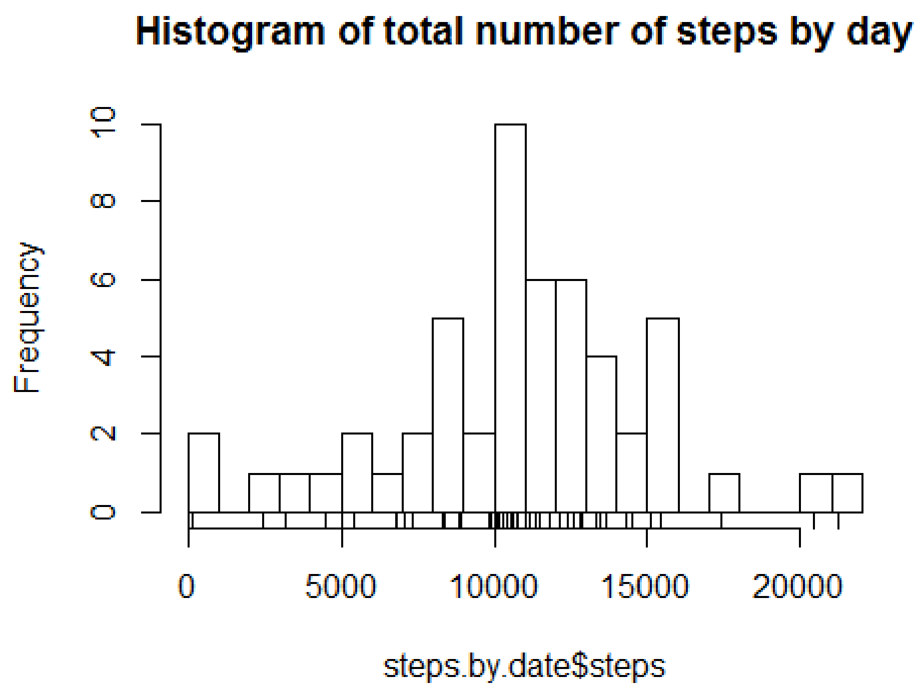
```r
hist(steps.by.date$steps, main="Histogram of total number of steps by day",
xlab="Average Steps per Day")
rug(steps.by.date$steps)
```



The histogram tells us that there are a whole lot of observations around the 10000+ mark. If you have one of these devices, you know that the default goal is 10000 steps per day.

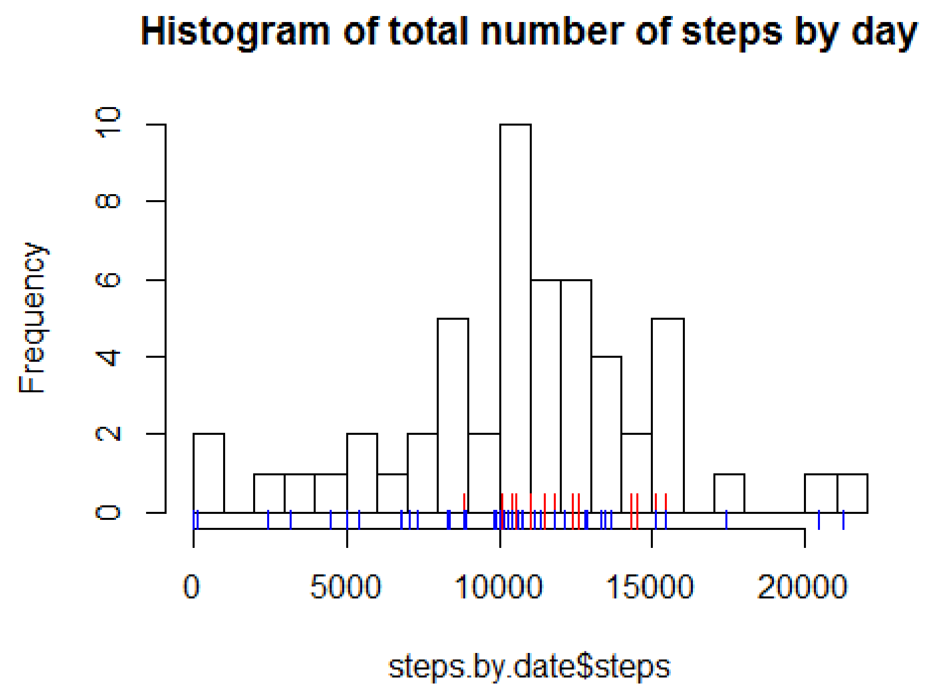
So, what if we play with the bin size:

```r
hist(steps.by.date$steps, breaks=20, main="Histogram of total number of steps
by day")
rug(steps.by.date$steps)
```



I like this better because it shows how clumped around the 10000 mark the data is. But, what if we check days of week.

```
steps.by.date$day <- weekdays(as.Date(steps.by.date$date))
hist(steps.by.date$steps, breaks=20, main="Histogram of total number of steps
by day")

steps.by.date$weekend <- steps.by.date$day %in% c("Saturday", "Sunday")
rug(steps.by.date$steps[which(steps.by.date$weekend)], col="red",
ticksize=0.08, lwd=1.5)
rug(steps.by.date$steps[which(!steps.by.date$weekend)], col="blue",
ticksize=0.04, lwd=1.5)
```



3. **Calculate and report the mean and median of the total number of steps**

```
summary(steps.by.date)
```

```
##          date        steps           day              weekend       
##  2012-10-02: 1   Min.   :   41   Length:53          Mode :logical  
##  2012-10-03: 1   1st Qu.: 8841   Class :character   FALSE:39       
##  2012-10-04: 1   Median :10765   Mode  :character   TRUE :14       
##  2012-10-05: 1   Mean   :10766                      NA's :0        
##  2012-10-06: 1   3rd Qu.:13294                                     
##  2012-10-07: 1   Max.   :21194                                     
##  (Other)   :47
```

```
mean(steps.by.date$steps)
```

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

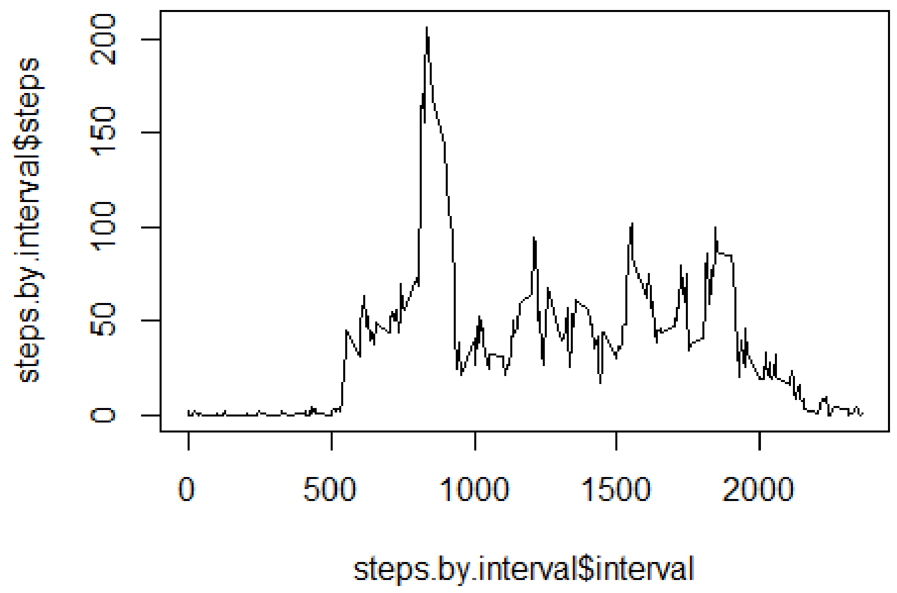
```
median(steps.by.date$steps)
```

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

What is the average daily activity pattern?

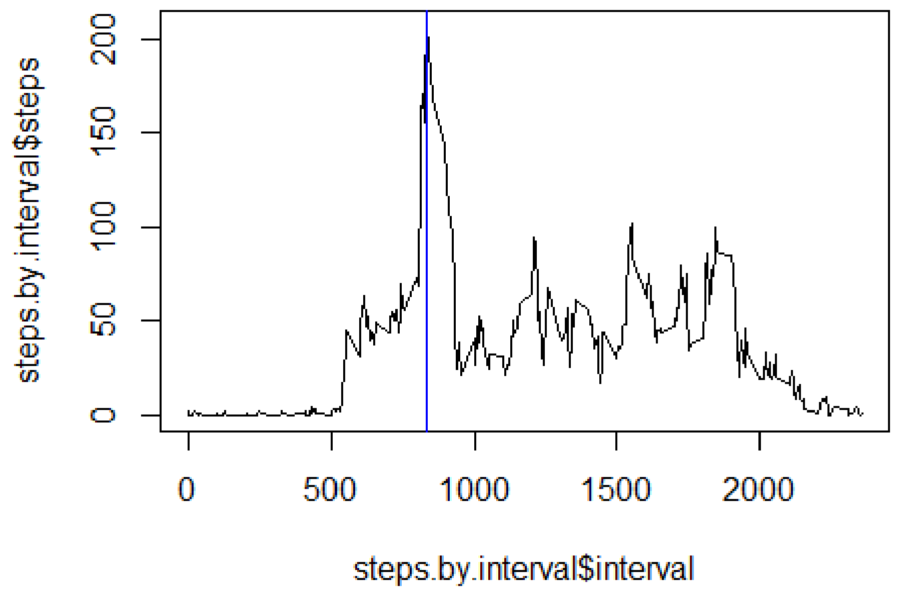
1. **Make a time series plot of the 5-minute interval (x-axis) and the average number of steps taken averaged across all days (y-axis)**

```
steps.by.interval <- aggregate(. ~ interval, data=fitData, FUN=mean)
 plot(steps.by.interval$interval, steps.by.interval$steps,type="l" )
```



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

```
steps.by.interval$interval[steps.by.interval$steps ==
max(steps.by.interval$steps)]

## [1] 835

 plot(steps.by.interval$interval, steps.by.interval$steps,type="l" )
abline(v=steps.by.interval$interval[steps.by.interval$steps ==
max(steps.by.interval$steps)], col="blue")
```



Imputing missing values

Note that there are a number of days/intervals where there are missing values. 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 the dataset.**

```
nrow(fitData[is.na(fitData$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 the 5-minute interval, etc.**

To answer this question, I'd like to know more about the nature of the missing data.

```
fitData$day <- weekdays(as.Date(fitData$date))
naData <- fitData[is.na(fitData$steps),]
unique(naData$day)

## [1] "Monday"    "Thursday"  "Sunday"    "Friday"    "Saturday" 
"Wednesday"
```

So, there's no consistent day of the week that's missing.

```
unique(naData$date)
```

```
## [1] 2012-10-01 2012-10-08 2012-11-01 2012-11-04 2012-11-09 2012-11-10
## [7] 2012-11-14 2012-11-30
## 61 Levels: 2012-10-01 2012-10-02 2012-10-03 2012-10-04 ... 2012-11-30
```

I've decided to fill NAs with the average of the interval for that time period and day of the week.

```
theNAs <- which(is.na(fitData$steps))
length(theNAs)

## [1] 2304

New.data <- fitData
for (i in 1:length(theNAs)) {
    New.data$steps[theNAs[i]] <-
mean(New.data[New.data$interval==New.data$interval[theNAs[i]] &
New.data$day==New.data$day[theNAs[i]],1], na.rm=TRUE)
}
New.data$steps[is.na(New.data$steps)]

## numeric(0)

## I want to replace NAs with the average for the interval for that day of
the week.
## So you should make a matrix: 288 rows, 7 days
```

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

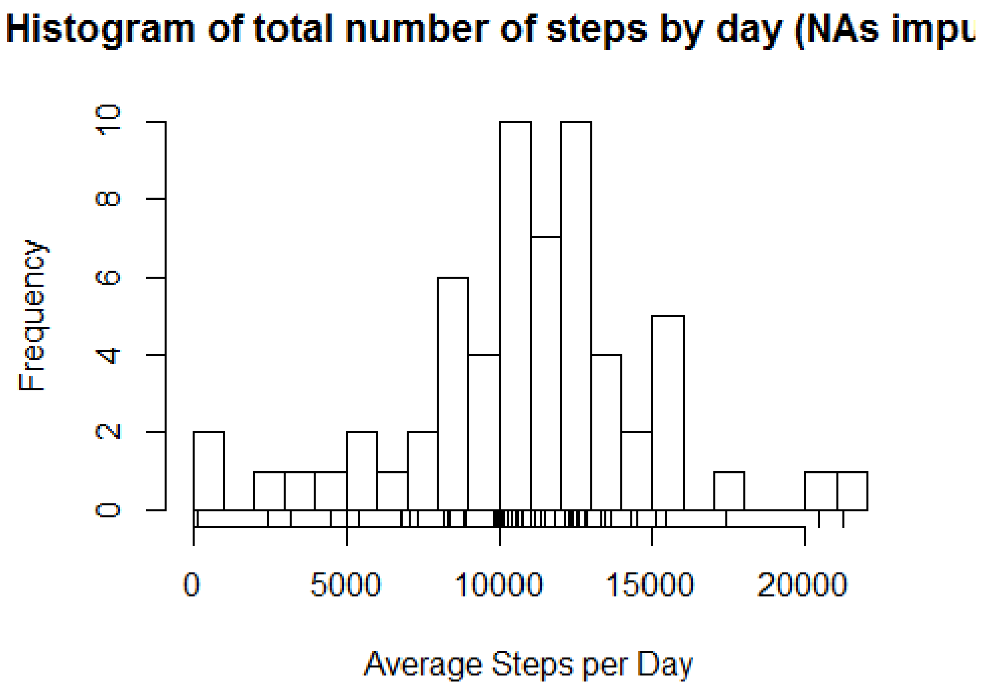
See above. I did and named it New.data.

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?**

```
str(New.data)

## 'data.frame':    17568 obs. of  4 variables:
##  $ steps   : num  1.43 0 0 0 0 ...
##  $ 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 ...
##  $ day     : chr  "Monday" "Monday" "Monday" "Monday" ...

New.data$date <- as.Date(New.data$date)
New.data$day <- as.factor(New.data$day)

imputed.steps.by.date <- aggregate(steps ~ date, data=New.data, FUN=sum)
str(imputed.steps.by.date)
```

```
## 'data.frame':    61 obs. of  2 variables:
##  $ date : Date, format: "2012-10-01" "2012-10-02" ...
##  $ steps: num  9975 126 11352 12116 13294 ...
```

```
hist(imputed.steps.by.date[,2], breaks=20, main="Histogram of total number of
steps by day (NAs imputed)", xlab="Average Steps per Day")
rug(imputed.steps.by.date$steps)
```



The original mean total number of steps per day was:

```
mean(steps.by.date$steps)

## [1] 10766.19

median(steps.by.date$steps)

## [1] 10765
```

The new mean total number of steps per day is:

```
mean(imputed.steps.by.date$steps)

## [1] 10821.21

median(imputed.steps.by.date$steps)

## [1] 11015
```

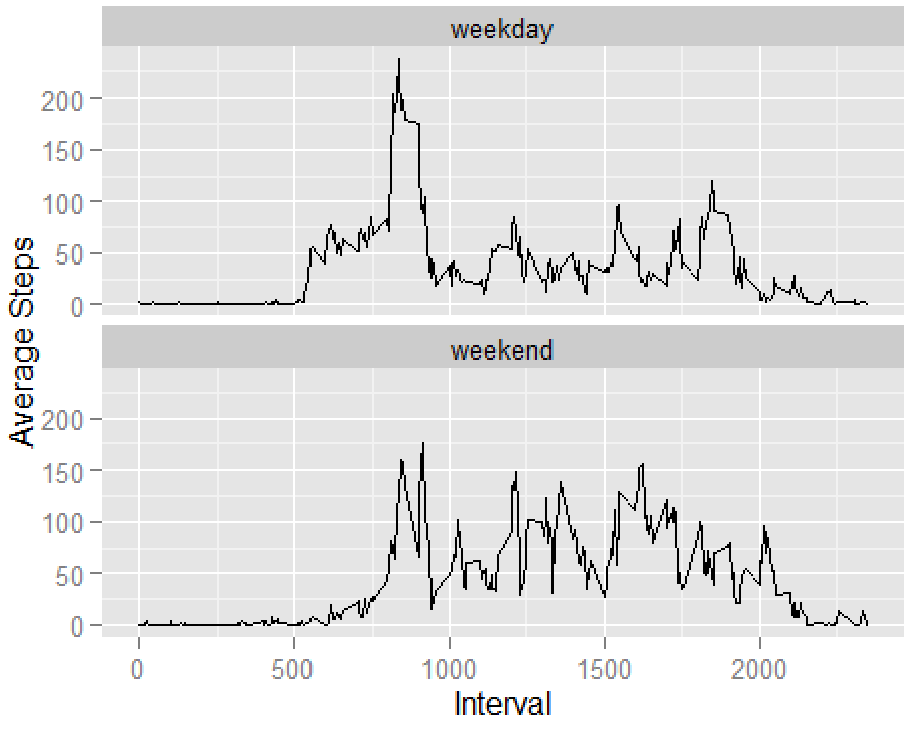
Are there differences in activity patterns between weekdays and weekends?

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.

```
New.data$weekend <- factor(New.data$day %in% c("Saturday", "Sunday"),
levels=c("FALSE","TRUE"), labels=c("weekday","weekend"))
```

2. Make a panel plot containing a time series plot of the 5-minute interval and average number of steps taken averaged across all weekdays and weekend days.

```
library(ggplot2)

steps.by.interval <- aggregate(steps ~ interval + weekend, data=New.data,
FUN=mean)

qplot(interval, steps, data=steps.by.interval, type="l", geom=c("line"),
xlab="Interval", ylab="Average Steps") + facet_wrap(~ weekend, ncol=1)
```


weekday
weekend
Average Steps
Interval