Reproducible Research: Peer Assessment 1

prepared by: CY Ting

Loading and preprocessing the data

The loading process begins with loading all the required libraries:

```
library(data.table)
library(ggplot2)
library(lubridate)
library(lattice)
```

The next steps is to read the file

```
dt<-fread("activity.csv")
dt$date<-as.Date(dt$date,"%Y-%m-%d")
```

What is mean total number of steps taken per day?

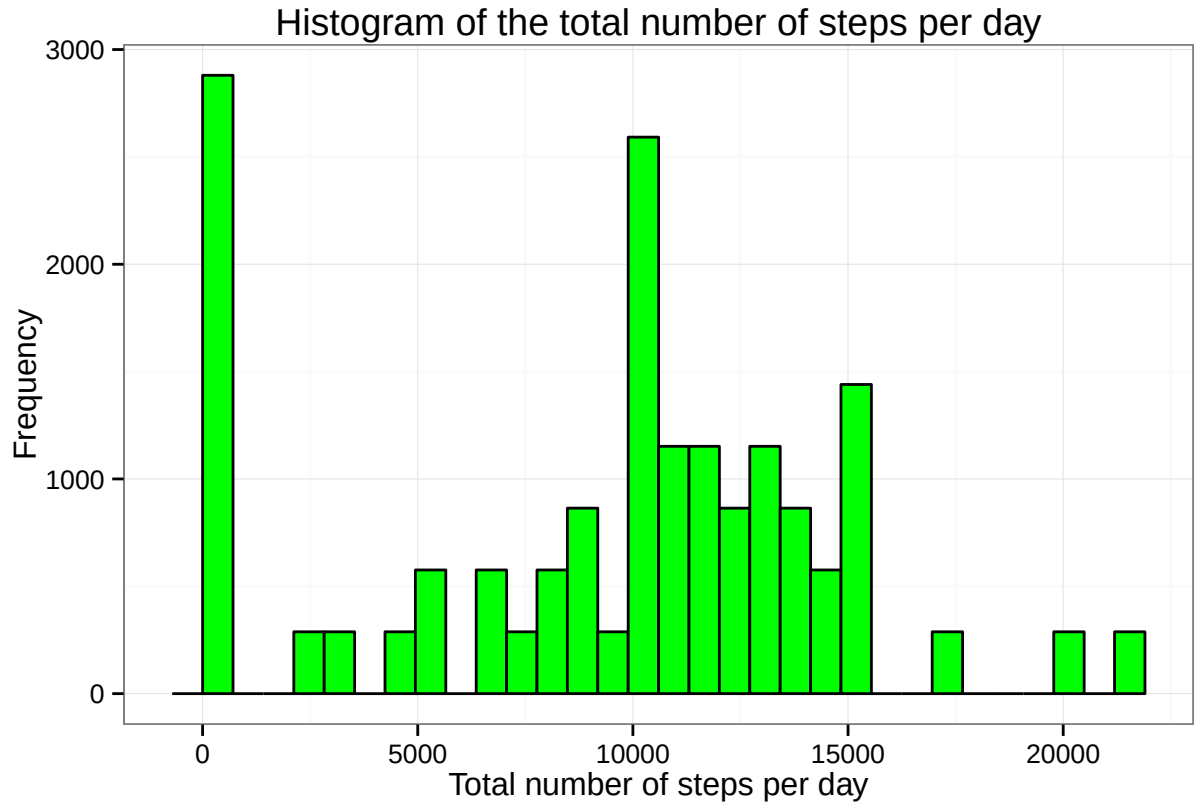
Extracting the total number of steps grouped by date

```
dt.sum.date = dt[, total_steps:= sum(steps, na.rm = T),by = date]
```

Histogram showing the total number of steps taken each day

```
ggplot(dt.sum.date, aes(x=total_steps)) +
     geom_histogram(colour="black", fill="green") +
     labs(x="Total number of steps per day",y="Frequency",
          title="Histogram of the total number of steps per day")+
     theme_bw()
```

```
## stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.
```



The mean and median of total steps per day

```
mean(dt.sum.date$total_steps)
```

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

```
median(dt.sum.date$total_steps)
```

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

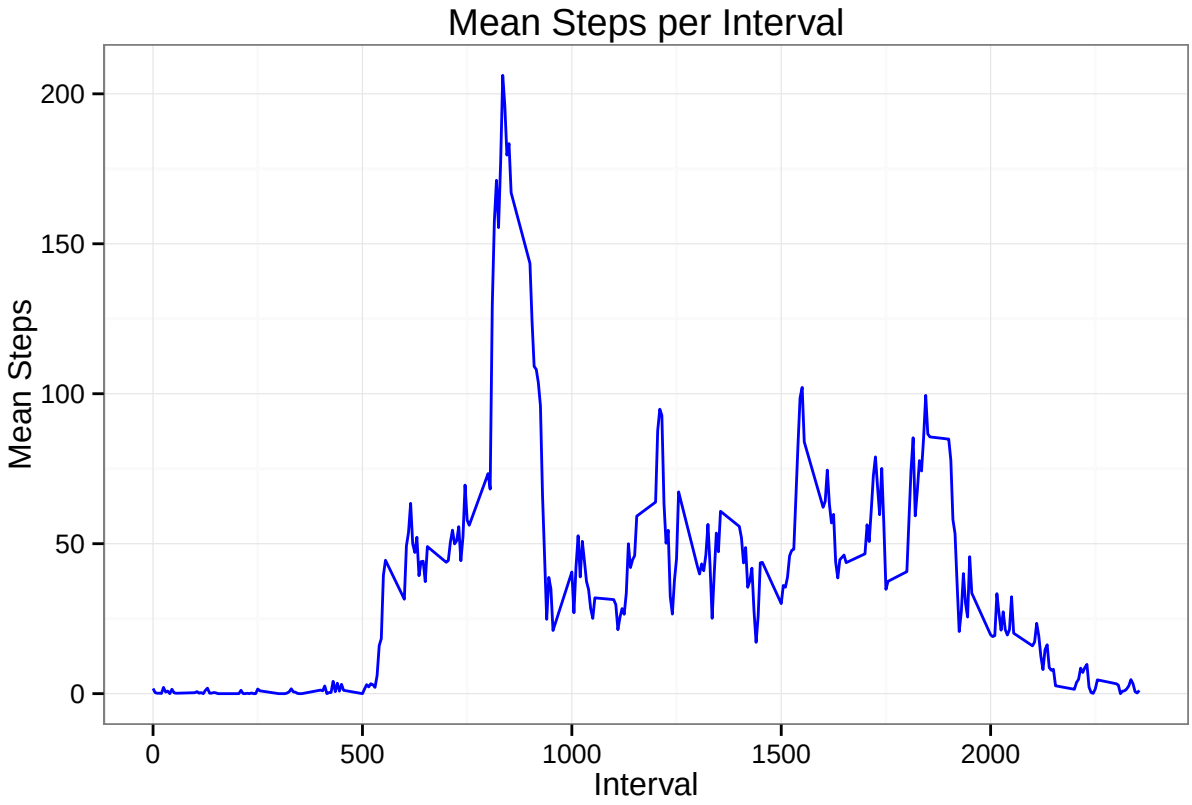
What is the average daily activity pattern?

Getting the mean across each day

```
dt.mean.interval = dt[, mean_steps:= mean(steps, na.rm = T),by = interval]
```

Line graph for average steps for each interval

```
ggplot(data=dt.mean.interval, aes(x=interval, y=mean_steps))+
     geom_line(col="blue")+
     labs(x="Interval",y="Mean Steps",title="Mean Steps per Interval")+
     theme_bw()
```



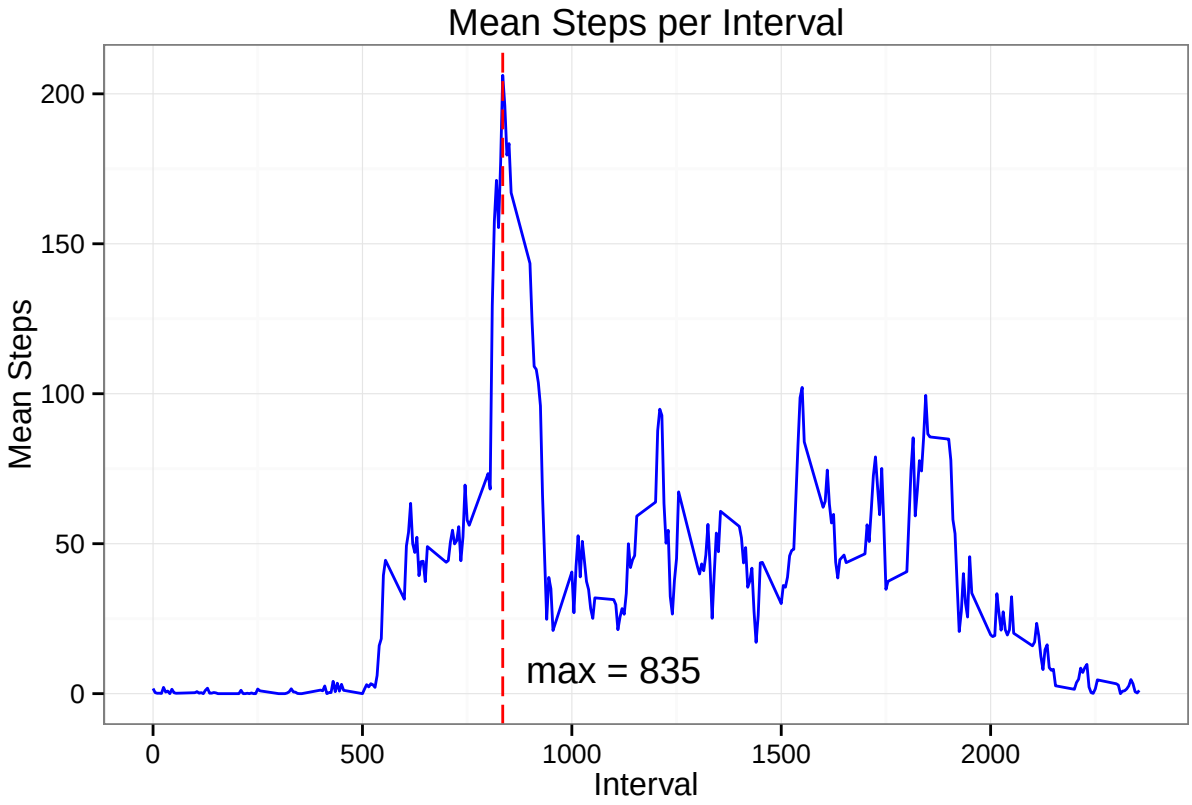
Identifying the interval with maximum number of steps

```
dt.mean.interval[which.max(dt.mean.interval$mean_steps)]
```

```
##    steps       date interval total_steps mean_steps
## 1:    NA 2012-10-01      835           0   206.1698
```

A vertical line at max value

```
ggplot(data=dt.mean.interval, aes(x=interval, y=mean_steps))+
     geom_line(col="blue")+
     labs(x="Interval",y="Mean Steps",title="Mean Steps per Interval")+
     geom_vline(xintercept = 835, colour="red", linetype = "longdash")+
     annotate("text", x = 1100, y = 8, label = "max = 835")+
     theme_bw()
```



Imputing missing values

The strategy used in this assignment to fill up missing values is done through replacing all "NA" with mean for the month.

The first step is to identify the number of NA in the Activity dataset.

```
cat(nrow(dt[is.na(dt$steps),]))
```

```
2304
```

Second, calculate the mean step for each month.

```
dt.meansteps.month<-dt[,mean(steps,na.rm=TRUE),by=month(date)]
```

Assigning NA with randomly assigned value between 30 to 45. The reason is that the mean for the months is centering around 37. Random sampling would be a better representation for the missing values.

```
dt.no.missing<-copy(dt)

set.seed(1)
for(i in 1:nrow(dt)){
     if(is.na(dt.no.missing$steps[i])){
          dt.no.missing$steps[i]<-sample(30:45,1)
```

```
    }
}
```

The sum of steps per day - after removing missing values

```
dt.sum.date.no.missing<-dt.no.missing[, total_steps:= sum(steps), by=date]
```

The mean and median of total steps per day

```
mean(dt.sum.date.no.missing$total_steps)
```

```
## [1] 10766.34
```

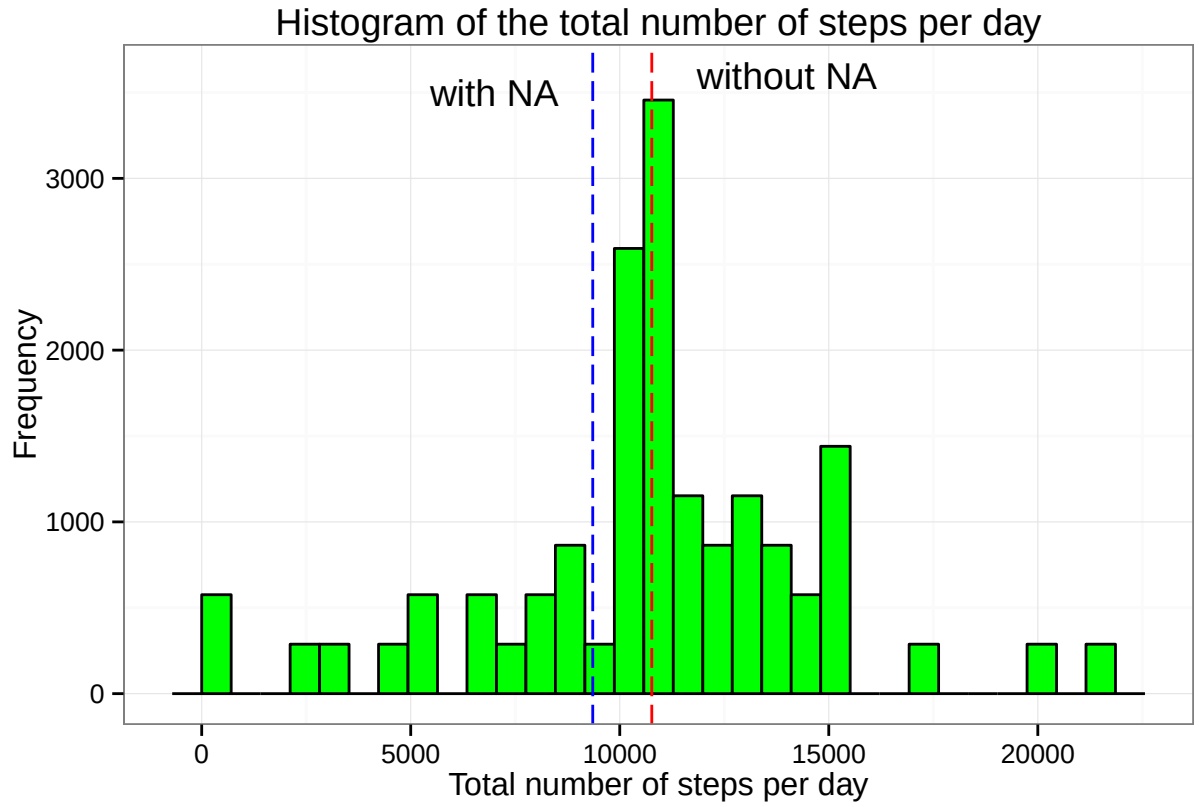
```
median(dt.sum.date.no.missing$total_steps)
```

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

Histogram for total steps for each date after removing the missing values

```
ggplot(dt.sum.date.no.missing, aes(x=total_steps)) +
    geom_histogram(colour="black", fill="green") +
    labs(x="Total number of steps per day",y="Frequency",
         title="Histogram of the total number of steps per day")+
    geom_vline(xintercept = mean(dt.sum.date$total_steps), colour="blue",
               linetype = "longdash")+
    annotate("text", x = 7000, y = 3500, label = "with NA")+
    geom_vline(xintercept = mean(dt.sum.date.no.missing$total_steps),
               colour="red", linetype = "longdash")+
    annotate("text", x = 14000, y = 3600, label = "without NA")+
    theme_bw()
```

```
## stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.
```



Are there differences in activity patterns between weekdays and weekends?

First, assign names of weekday to a variable named "weekdays"

```
weekdays <- c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday")
```

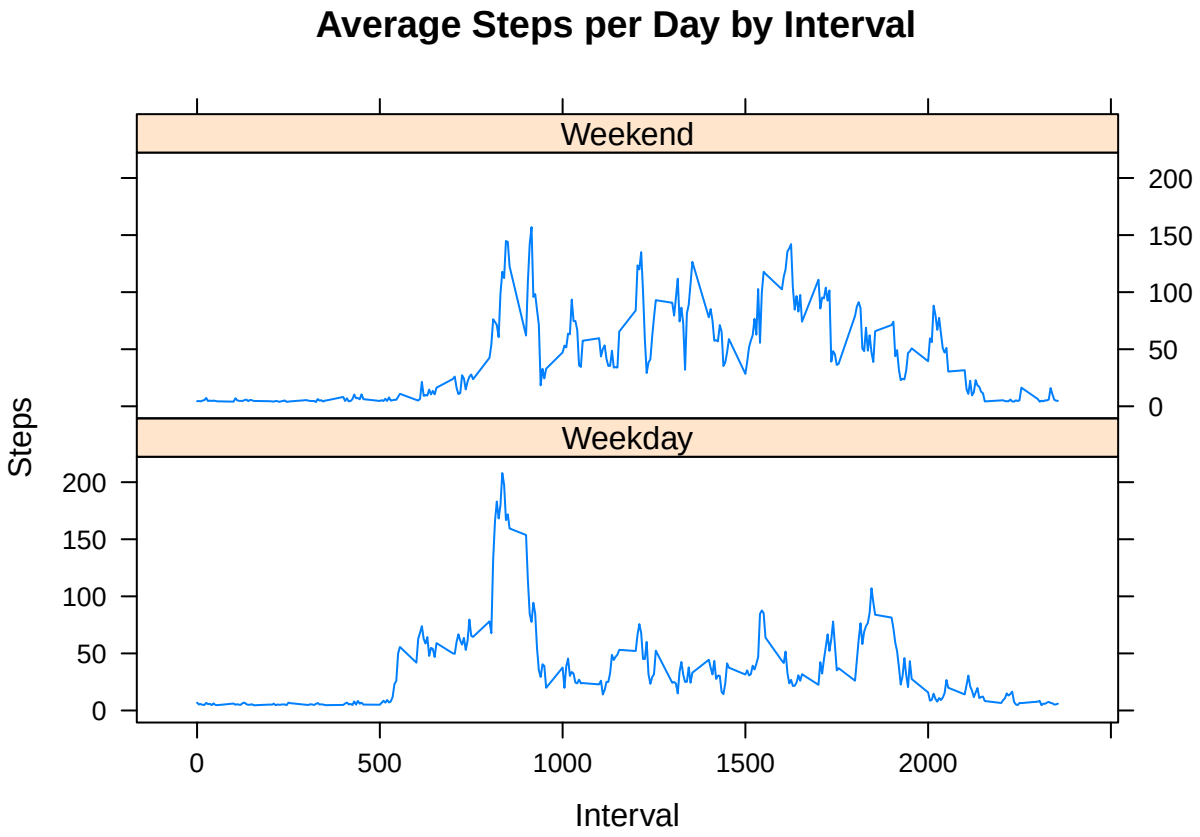
Check if a particular date falls under weekdays

```
dt.no.missing$dayType = as.factor(ifelse(is.element(weekdays(as.Date(dt.no.missing$date)),
                                                    weekdays),
                                         "Weekday", "Weekend"))
```

plotting the xyplot for comparing weekdays and weekends

```
steps.interval <- aggregate(steps ~ interval + dayType, dt.no.missing, mean)

xyplot(steps.interval$steps ~ steps.interval$interval|steps.interval$dayType,
       main="Average Steps per Day by Interval",
       xlab="Interval",
       ylab="Steps",layout=c(1,2), type="l")
```


Average Steps per Day by Interval
Weekend
Weekday
Steps
Interval
0
50
100
150
200
0
500
1000
1500
2000