Reproducible Research: Peer Assessment 1

Loading and preprocessing the data

In order to load activity.csv, it was first necessary to extract the .csv file from the activity.zip file using unzip locally. Next, the activity.csv dataset was loaded using the code immediately below. In addition, various attributes were transformed to their corresponding, appropriate datatypes. Lastly, the structure of the data frame is presented. As you will immediately see, the step attribute has several NAs that will be addressed in a later transformation.

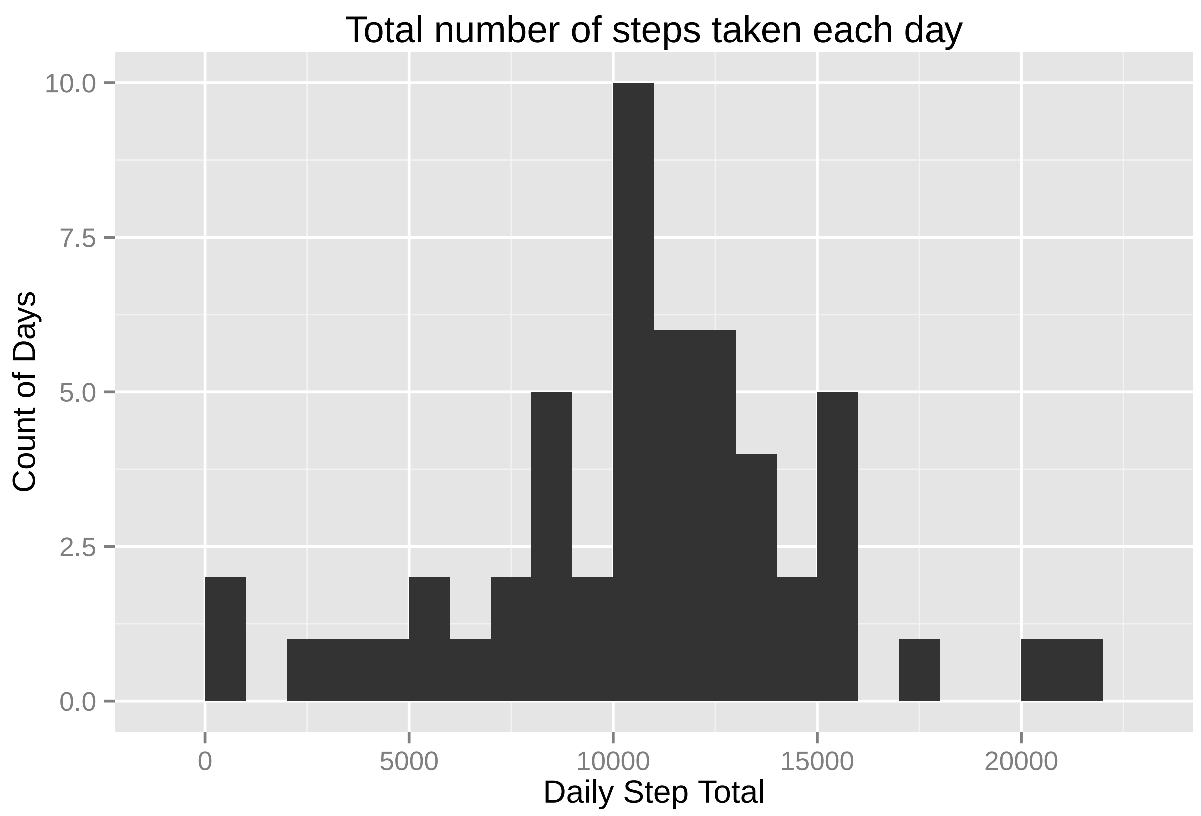
```r
rawMonitoringData <- read.csv('activity.csv')
rawMonitoringData$date <- as.Date(rawMonitoringData$date)
rawMonitoringData$steps <- as.numeric(rawMonitoringData$steps)
rawMonitoringData$interval <- as.numeric(rawMonitoringData$interval)

str(rawMonitoringData)
```

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

What is mean total number of steps taken per day?

Below, is a histogram that captures the total number of steps taken each day:

```r
library(ggplot2)

# calculate total step counts per day
dailyStepTotals <- aggregate( steps ~ date, data=rawMonitoringData,
                              FUN=sum, na.rm=TRUE)

# build histogram of total steps taken per day
qplot(dailyStepTotals$steps, main="Total number of steps taken each day",
      xlab="Daily Step Total",
      ylab="Count of Days",
      ylim=c(0,10), binwidth=1000)
```



The mean and median number of steps taken per day are reported below:

```
mean(dailyStepTotals$steps)
```

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

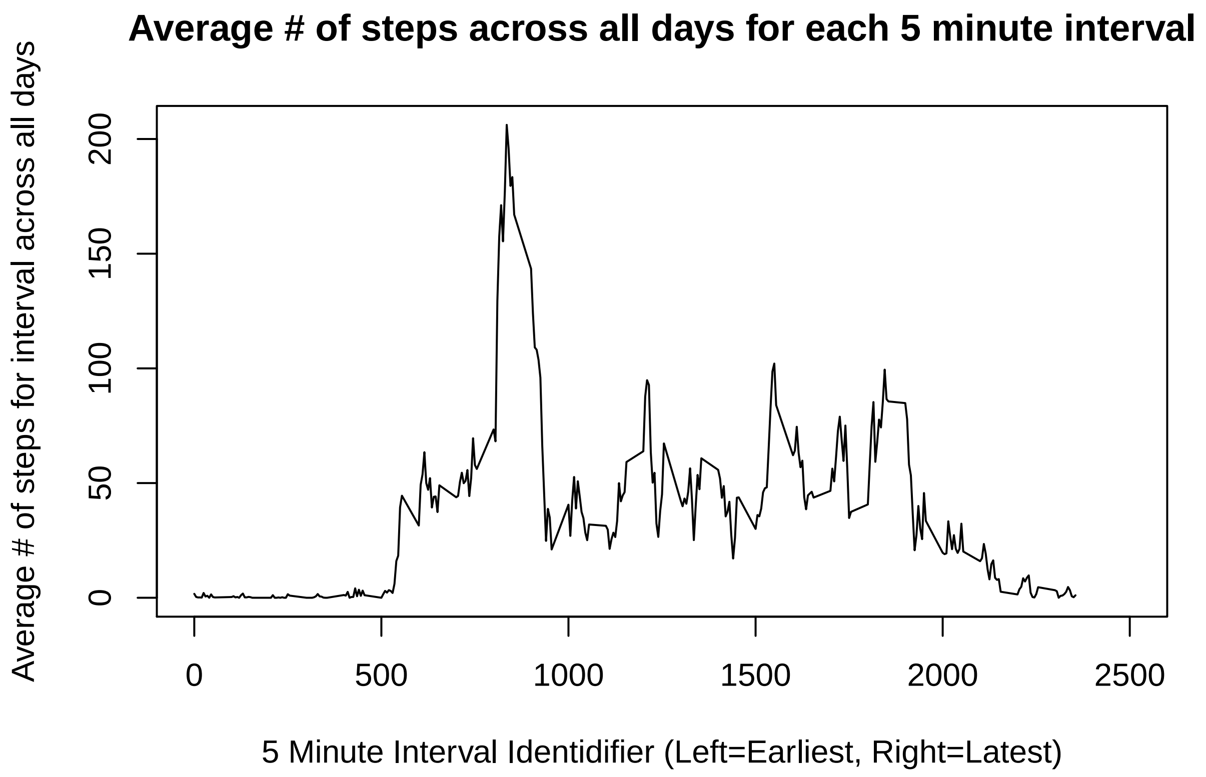
```
median(dailyStepTotals$steps)
```

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

What is the average daily activity pattern?

Below is a chart that captures the average number of steps across all days for each 5 minute measured interval. Note that the 5-minute interval with the largest number of steps is '835' which corresponds with 8:35AM. Also note that the number of steps associated with this interval is '206.1698' which corresponds with the highest peak in the chart.

```
AverageStepsByInterval <- aggregate( steps ~ interval, data=rawMonitoringData,
                                     FUN=mean, na.rm=TRUE)

plot( steps ~ interval, data=AverageStepsByInterval, type="l",
      main="Average # of steps across all days for each 5 minute interval",
      ylab="Average # of steps for interval across all days",
      xlab="5 Minute Interval Identidifier (Left=Earliest, Right=Latest)",
      xlim=c(0,2500)
      )
```



```r
intervalWithMax <- AverageStepsByInterval[which.max(AverageStepsByInterval$steps),]$interval
intervalWithMax
```

```
## [1] 835
```

```r
subset(AverageStepsByInterval, interval==intervalWithMax)$steps
```

```
## [1] 206.1698
```

Imputing missing values

As indicated earlier, the activity data set has several records with missing step counts (denoted by NA). Below, you will see that out of a total of 15,264 records in the activity.csv dataset, 2,304 records contain at least one missing value.

```r
sum(complete.cases(rawMonitoringData))
```

```
## [1] 15264
```

```r
nrow(rawMonitoringData) - sum(complete.cases(rawMonitoringData))
```

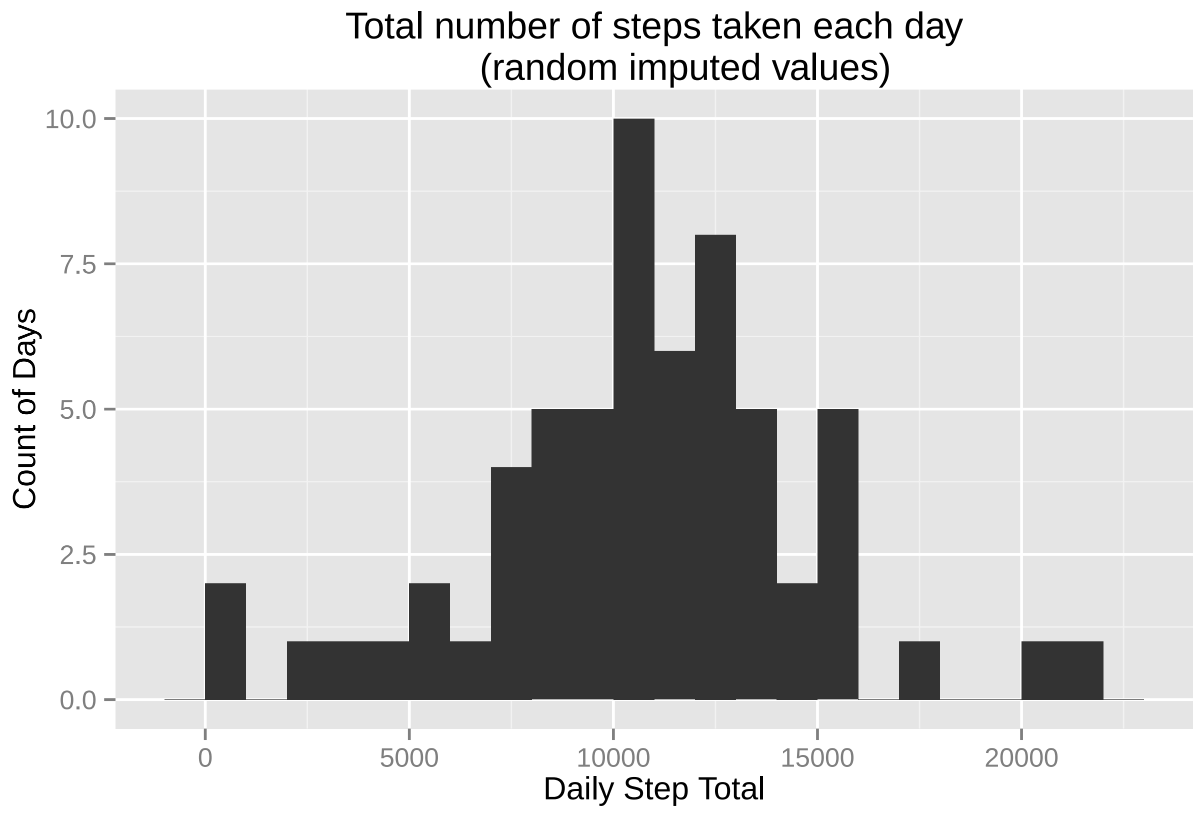
```
## [1] 2304
```

Now that these missing values have been highlighted and the relative scope has been determined, a strategy needs to be devised to handle these NA values. For the sake of simplicity, I simply used the "random draw" capability of the impute() function within the Hmisc library. This approach will simply randomly draw from the pool of existing, non-NA values in the activity dataset.

```r
library(Hmisc)
library(ggplot2)

# copy the dataframe
imputedMonitoringData <- rawMonitoringData

# use the random draw approach to impute missing step values
imputedMonitoringData$steps <- with( imputedMonitoringData, impute(steps,"random") )

# recalculate daily step totals using the new data frame with imputed values.
imputeDailyStepTotals <- aggregate( steps ~ date, data=imputedMonitoringData,
                                    FUN=sum, na.rm=FALSE)
```

Below, is an updated version of the daily step total histogram that was presented earlier. This updated histogram refelects the newly calculated imputed values; whereas, the original histogram simply ignored NA values. When comparing the mean and median values between the imputed and non-imputed data frames, we can see that both sets have the same median value of 10,765 steps; however, the mean number of steps is slightly higher with the imputed data frame (10,775.05 versus 10,766.19). The imputation of step values appears to have reduced the number of days associated with the highest daily step total estimate of ~10K. This spike was lowered from 10 to 9 days, and, generally speaking, the estimates are not as tightly clustered around the mean with the imputed values incorporated.

```r
qplot(imputeDailyStepTotals$steps, main="Total number of steps taken each day
      (random imputed values)",
      xlab="Daily Step Total", ylab="Count of Days", ylim=c(0,10),
      binwidth=1000)
```



```r
mean(imputeDailyStepTotals$steps)
```

```
## [1] 10700.08
```

```r
median(imputeDailyStepTotals$steps)
```

```
## [1] 10600
```

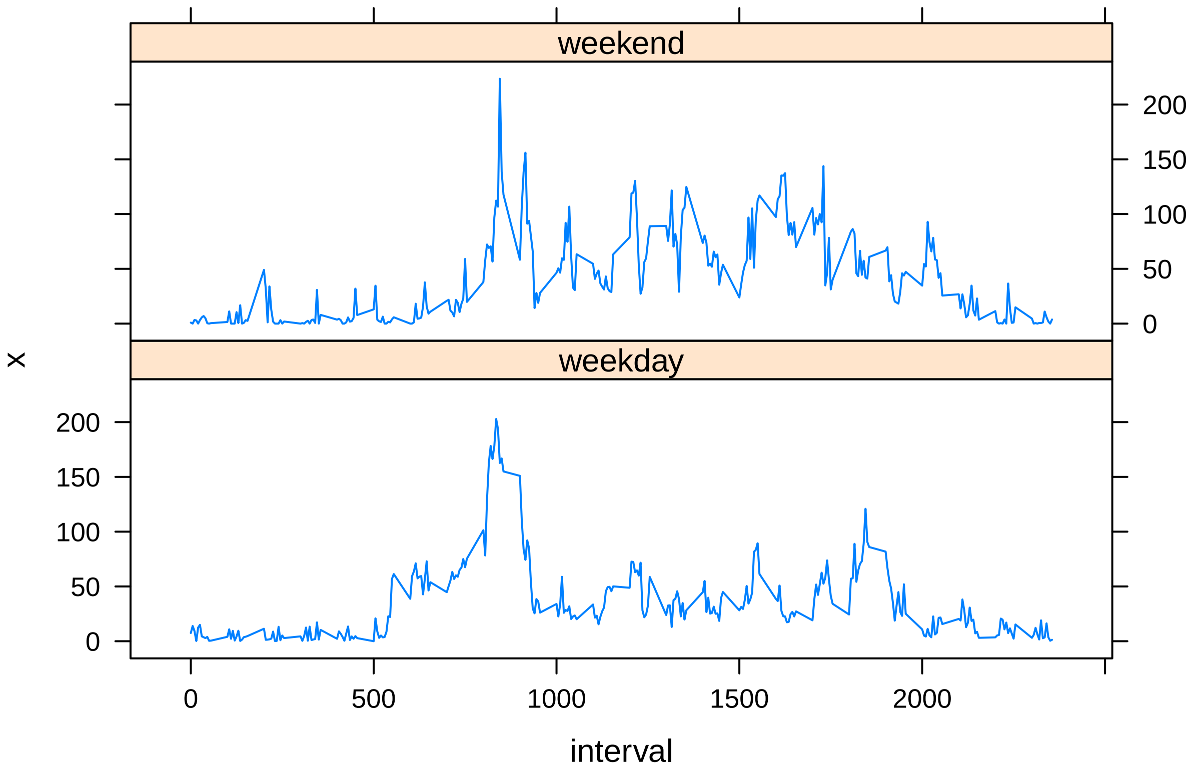
Are there differences in activity patterns between weekdays and weekends

Given that the original data set does not contain an attribute to indicate whether a given measurement date was a weekday or weekend day, it was necessary to add a new attribute called "day_type", then, populate this attribute for all records using the "weekdays function". Once the new "day_type" attribute was added and populated, it was necessary to recalculate average daily step counts per interval in order to determine diffferences between weekday and weekend activities.

```r
#Assign dates to either weekend or weekday category
imputedMonitoringData$day_type[weekdays(imputedMonitoringData$date)=="Saturday"] <- "weekend"
imputedMonitoringData$day_type[weekdays(imputedMonitoringData$date)=="Sunday"] <- "weekend"
imputedMonitoringData$day_type[is.na(imputedMonitoringData$day_type)] <- "weekday"
imputedMonitoringData$day_type <- as.factor(imputedMonitoringData$day_type)

#Aggregate imputed step data by interval and day type (e.g. weekday, weekend)
AverageImputedStepsByInterval <- aggregate( imputedMonitoringData$steps,
                                 list(interval=imputedMonitoringData$interval,
                                      day_type=imputedMonitoringData$day_type),
                                data=imputedMonitoringData,
                                FUN=mean, na.rm=TRUE)
```

From the chart below, we can see that although the highest average interval occurred during the weekday, generally speaking, there is a higher level of activity occurring during the weekend days. During the week, we see a high level of early morning activity, and, on average, this activity remains at a low level (rarely realize 100+ steps per interval). On the other hand, during the weekend, there are a large number of intervals throughout the day that exceed the 100+ step per interval mark.

```r
#Build lattice plot to hightlight weekday and weekend differences
xyplot( x ~ interval | day_type,
        data=AverageImputedStepsByInterval,
        type="l",
        layout=c(1,2))
```


weekend
weekday
x
interval
0
50
100
150
200
0
500
1000
1500
2000