Reproducible Research: Peer Assessment 1

Loading and preprocessing the data

We begin by loading libraries, clearing the workspace, and unzipping the `activity.zip` dataset, which includes the comma-separated `activity.csv` file, which contains the raw dataset for this analysis.

```
library(dplyr); library(mice); library(lattice); library(moments); library(lambda.tools)
knitr::opts_chunk$set(cache=TRUE)
# Set working directory (edit this path as desired), fetch data to it
dataDirName <- "C:/Users/tom/Documents/DataScience/course 5/wk2/RepData_PeerAssessment1"
setwd(dataDirName)  # make it working directory
# skip the unzip if activity.csv already exists; otherwise, unzip it to current working director
y
if( !file.exists("./activity.csv") ) unzip("./activity.zip", overwrite = T, exdir = getwd())
activity <- read.csv("activity.csv", header = T, sep = ",")
```

The activity dataset consists of 17568 observations across 3 variables. We observe some missing values in the `steps` field, denoted as NAs in the summary below. These will be discussed later in the report. For now we will ignore the missing values.

```
str(activity)
```

```
## 'data.frame':    17568 obs. of  3 variables:
##  $ steps   : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ 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 ...
```

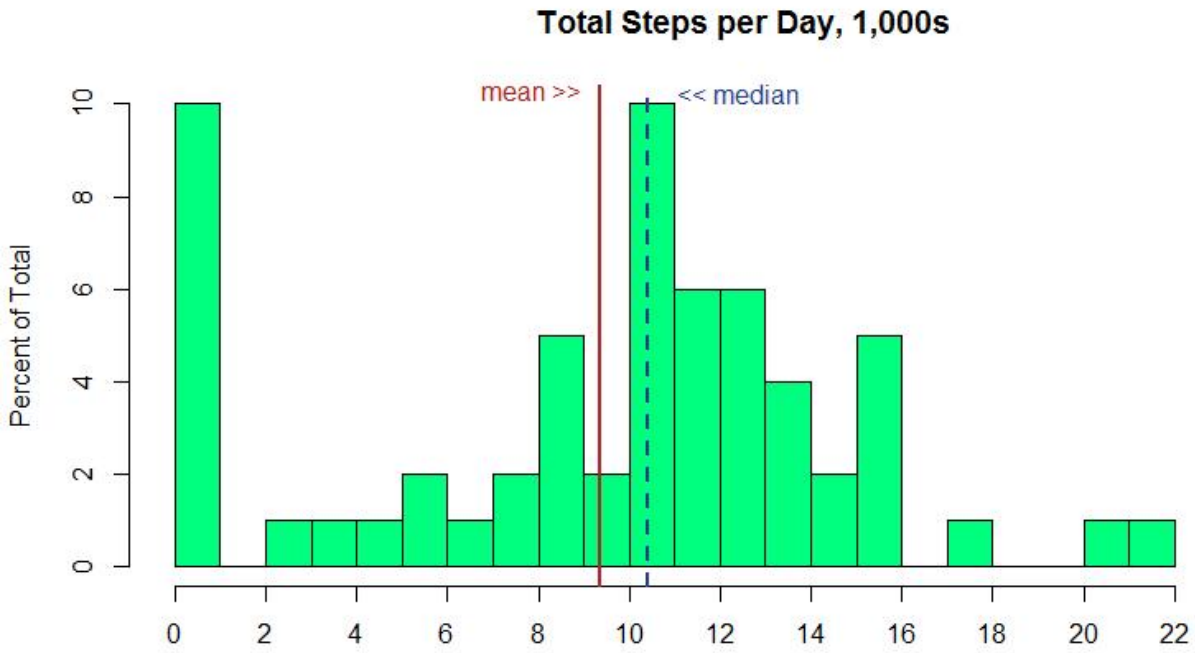
Mean and median total number of steps taken per day

The **mean** and **median** total number of steps taken per day are reported below. In addition, a histogram representing the frequency distribution of the total number of steps taken each day is presented. The **mean** and **median** values (dark red line, dashed blue line, respectively) are shown as vertical intercept lines on the plot. We observe that the **mean** is slightly less than the **median**, so this is a slightly left-skewed distribution.

```
sumSteps <- activity %>% group_by(date) %>% summarize(sum=sum(steps, na.rm=TRUE))
cat("The data spans", dim(sumSteps)[1], "days.",
     "\nStatistics, steps taken per day:\tMean:", (mn <- round(mean(sumSteps$sum),0)),
    "\tMedian:", (md <- round(median(sumSteps$sum),0)))
```

```
## The data spans 61 days.
## Statistics, steps taken per day: Mean: 9354  Median: 10395
```

```
hst <- function() {
     hist(sumSteps$sum/1000, breaks=20, main='Total Steps per Day, 1,000s', xlab='',
        ylab='Percent of Total', xlim=c(0,25), col='springgreen', axes=F)
     axis(2)
     axis(1, at=seq(0,ceiling(max(sumSteps$sum)/1000), by=2),
        labels=seq(0,ceiling(max(sumSteps$sum)/1000), by=2))
     abline(v=md/1000, col='royalblue4', lwd=2, lty=2)  # dashed blue line to indicate median
     abline(v=mn/1000, col='firebrick', lwd=2)  # dark red line to indicate mean
     text(mn/1000-1.5, 10.25, "mean >>", col = 'firebrick')
     text(md/1000+2, 10.25, "<< median", col = 'royalblue4')
}
hst()  # display it
```




```
cat("Skewness:", round(skewness(sumSteps$sum),2))
```

```
## Skewness: -0.39
```

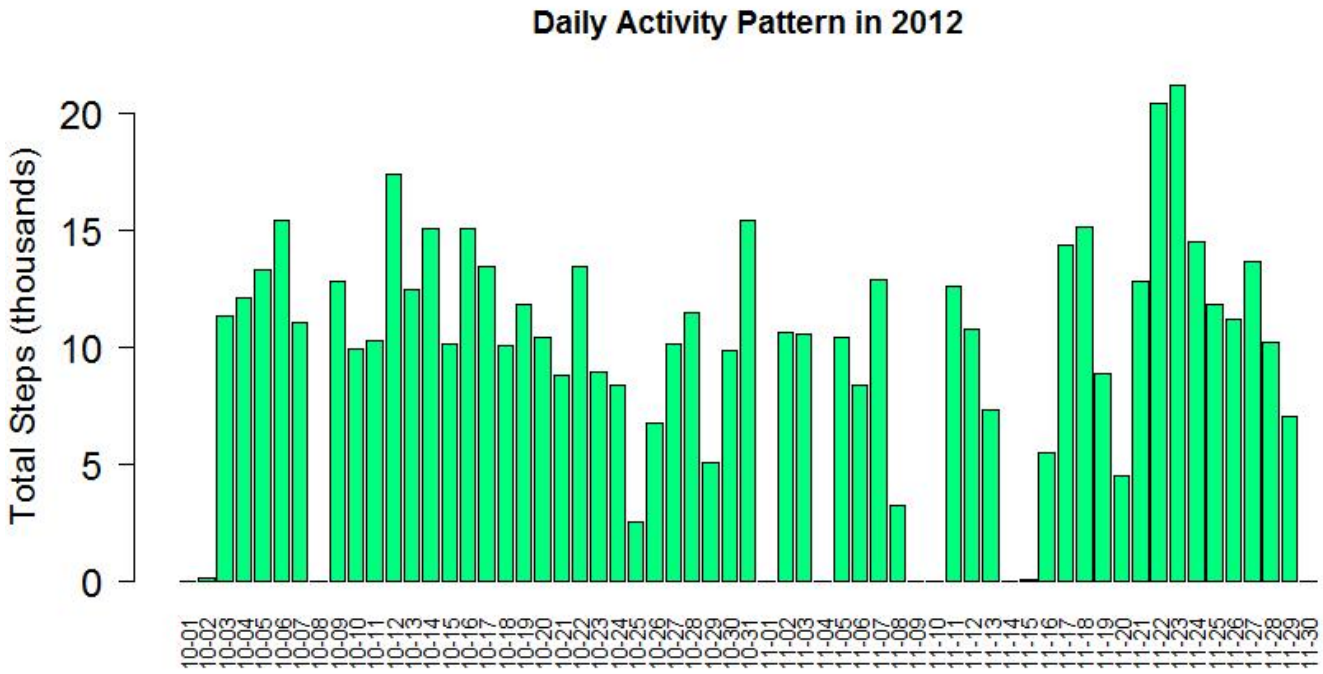
The mean and median total steps per day values for the original dataset are 9354 and 10395, respectively.

A negative skewness value close to zero indicates a slight left-skew; symmetric distributions have zero skew. We note the very large proportion at the far left of the plot.

What is the average daily activity pattern?

The total number of steps taken each day during the period 10-01 to 11-30 is presented in the barplot below.

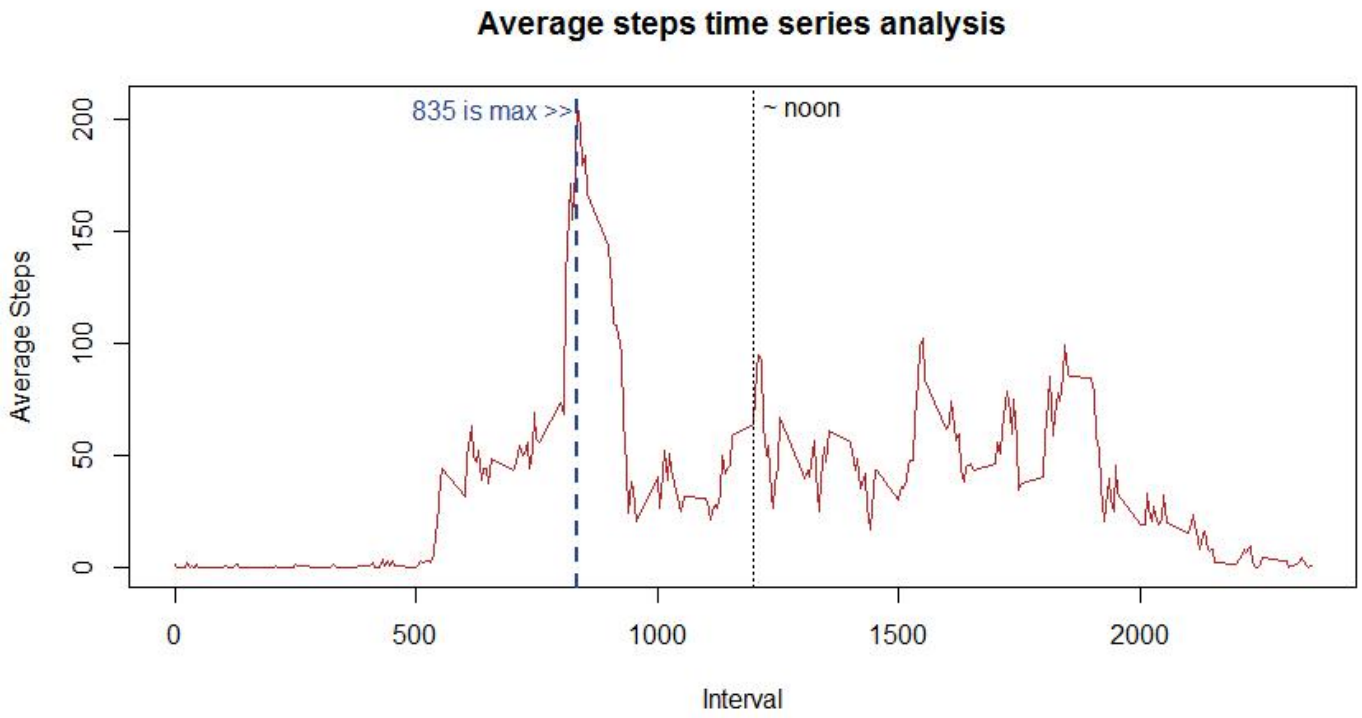
```
{par(las=2) # x-axis label orientation perpendicular
barplot(height=sumSteps$sum/1000,
        main="Daily Activity Pattern in 2012", xlab="", ylab="Total Steps (thousands)",
        names.arg = sub("2012-", "", sumSteps$date), col='springgreen', cex.names = .8,
        cex.axis = 1.5, cex.lab=1.3)
par(las=0)}
```



We will come back to the daily activity patterns later in the report, specifically to determine if there are differences between weekday and weekend-day steps activity.

This dataset contains 288 intervals per day, each five minutes in duration. We are interested in the daily activity pattern, specifically the average number of steps taken in each five-minute interval, averaged across all days. This pattern is presented on the plot below.

```
# calculate average number of steps per interval across all dates
avgSteps <- activity %>% group_by(interval) %>% summarize(average=mean(steps, na.rm=TRUE))
# which interval across all dates contains highest number of steps?
mx_idx<-with(avgSteps, which(average==max(average))) # interval index with max average
mx_val<-with(avgSteps, interval[which(average==max(average))]) # max interval average value
plot(y=avgSteps$average, x=avgSteps$interval, main="Average steps time series analysis",
     xlab="Interval", ylab="Average Steps", col='firebrick', type = "l")
abline(v=mx_val, lwd=2.5, lty=2, col = 'royalblue4')
abline(v=1200, lwd=1, lty=3)
text(mx_val-175, 205, paste0(mx_val, " is max >>"), col = 'royalblue4')
text(1300, 205, "~ noon", col = 'black')
```



We observe a number of activity spikes during the day, the **largest of which at interval 835**, which likely corresponds to a activity just after 8:00AM. The maximum value (835) is highlighted on the chart as a dashed vertical line. We also observe spikes in activity mid-day (dotted line), at mid-afternoon, and at about 4PM and 6PM.

Imputing missing values

As noted earlier, there are a significant number of observations with missing values (coded as `NA`) in the `steps` field. These are a potential source of undesirable bias. The analysis so far has ignored missing values, but we will explore one method for imputing missing values.

```
cat("The steps variable includes", (s<-sum(is.na(activity$steps))), "NAs, or ",
    round(100*s/length(activity$steps),0), "% of observations")
```

```
## The steps variable includes 2304 NAs, or  13 % of observations
```

We note a high proportion of missing values in `steps`, and they cluster throughout the observations (including at the beginning and end) in the `activity` dataset, as tabulated below. We use the range_for() function to easily identify ranges for missing values.

```
NAspan <- range_for(TRUE, is.na(activity$steps))
data.frame(`lower bound`=NAspan$min, `upper bound`=NAspan$max)
```

```
##   lower.bound upper.bound
## 1           1         288
## 2        2017        2304
## 3        8929        9216
## 4        9793       10080
## 5       11233       11808
## 6       12673       12960
## 7       17281       17568
```

The author chose not to use simpler methods for imputing missing values, such as substituting column means or medians. Instead, we will examine the MICE method to impute missing values - MICE is an acronym for Multivariate Imputation by Chained Equations. The associated function is mice(), found in the `mice` library, which was loaded earlier. The author experimented with the following mice() function variables: `m` (number of multiple imputations), `method` (imputation method), and `maxit` (number of iterations), specifically the value ranges m={5,7,8}, method={fastpmm, pmm}, maxit={10,20,30,50,80,100}.

Fit was evaluated with a densityplot, the author determined that the m=5, method=pmm, maxit=100 parameter values yielded suitable results. That densityplot is included as a pdf in the /figure directory of this github listing. On the plot, magenta denotes the imputed values - we observe that they cluster approximately around original, valid points (the blue line), with notable exceptions at very low values.

The `date` and `interval` column data is unchanged by this operation - the author checks this assumption below.

```
micedActivity <- mice(data = activity, m = 5, method = "pmm", maxit = 100, seed=2222, print=F)
```

```
newActivity <- complete(micedActivity, 1)  # extract the new dataframe
# show that `date` and `interval` variables are unchanged from original dataset
{if(sum(newActivity$date!=activity$date) == 0 ) cat("post-imputation:  date variable unaltered")
if(sum(newActivity$interval!=activity$interval) == 0 ) cat("\t\tinterval variable unaltered")}
```

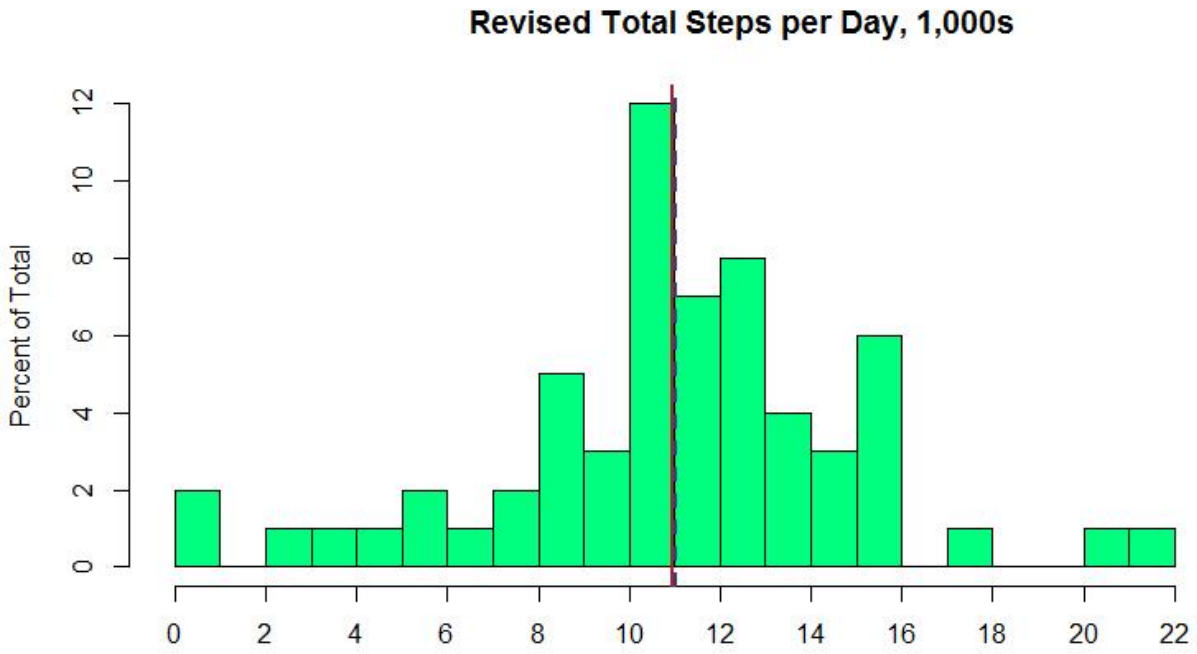
```
## post-imputation:  date variable unaltered        interval variable unaltered
```

Calculate statistics for imputed dataset

```
revSumSteps <- newActivity %>% group_by(date) %>% summarize(sum=sum(steps, na.rm=TRUE))
cat("Revised statistics, steps taken per day:\tMean:", (revMn <-
round(mean(revSumSteps$sum),0)),
    "\tMedian:", (revMd <- round(median(revSumSteps$sum),0)))
```

```
## Revised statistics, steps taken per day: Mean: 10951     Median: 11015
```

```
hst <- function() {
     hist(revSumSteps$sum/1000, breaks=20, main='Revised Total Steps per Day, 1,000s', xlab='',
        ylab='Percent of Total', xlim=c(0,25), col='springgreen', axes=F)
     axis(2)
     axis(1, at=seq(0,ceiling(max(revSumSteps$sum)/1000), by=2),
        labels=seq(0,ceiling(max(revSumSteps$sum)/1000), by=2))
     abline(v=revMd/1000, col='royalblue4', lwd=2, lty=2)  # dashed blue line to indicate median
     abline(v=revMn/1000, col='firebrick', lwd=2)  # dark red line to indicate mean
}
hst()  # display it
```



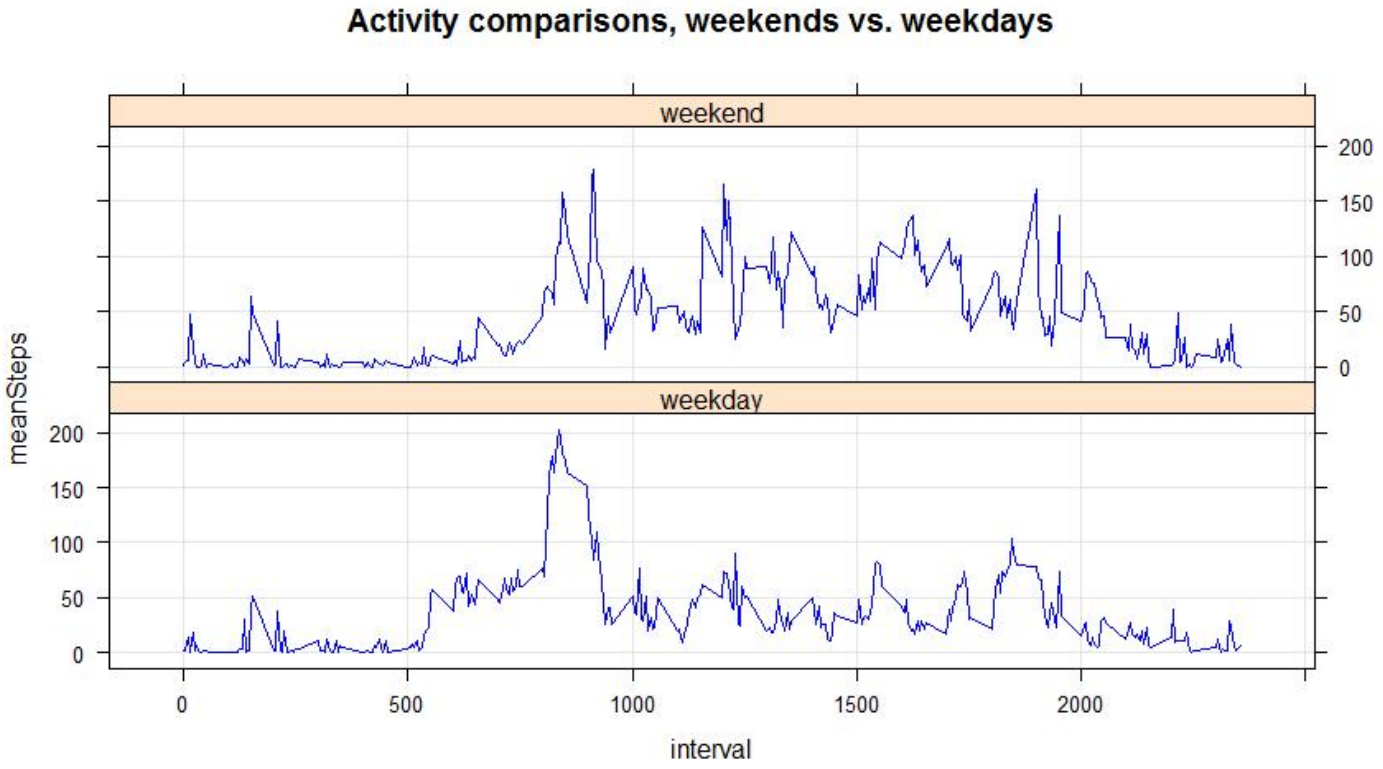
Imputing missing `step` values has had an interesting impact on the distribution of the total steps taken per day, as shown above. **The revised mean and median values are 10951 and 11015, respectively.** As before, the **revised mean** and **revised median** values are shown as vertical intercept lines on the histogram (dark red line, dashed blue line, respectively). We observe that **revised mean** and **revised median** values have both shift slightly to larger values after imputation, and they are nearly equal, which indicates a more symmetrical distribution as a result of imputation.

Are there differences in activity patterns between weekdays and weekends?

Now we will explore any differences in activity patterns between weekdays and weekends as measured in the `step` variable. We will add a new factor variable to the `activity` dataset to differentiate between weekend and weekday observations.

A panel plot containing a time series plot of the 5-minute interval and the average number of steps taken, averaged across all weekday days or weekend days, is presented below.

```
library(lattice);library(ggplot2)
# create new factor variable `dayType`
dayType <- weekdays(as.Date(newActivity$date))
for(i in 1:length(dayType)) {
    if(dayType[i] == "Saturday" | dayType[i] == "Sunday") {
        dayType[i] <- "weekend"
    } else { dayType[i] <- "weekday"
    }
}
newActivity$dayType <- as.factor(dayType)
# calculate means by interval by dayType
dayTypeNewActivity <-
     newActivity %>% group_by(interval, dayType) %>% summarize(meanSteps=round(mean(steps),2))
(g1 <- xyplot(meanSteps ~ interval | dayType, data = dayTypeNewActivity,
       type = "l", col="blue", grid=T, layout = c(1, 2),
       main="Activity comparisons, weekends vs. weekdays"))
```




```
g2 <- ggplot(data=dayTypeNewActivity, aes(x=meanSteps, fill=dayType)) +
     geom_histogram(color = "black", binwidth = 2) + ylim(0,30) + xlim(0,180) +
     facet_grid(. ~ dayType) + theme(legend.position="bottom") +
     ggtitle("Activity comparisons, weekdays vs. weekends")
```

The weekday activity is represented by the plot above. There is more sustained early-morning weekday activity for this individual as compared to the weekend case at top. The weekend activity appears to increase during mid-day hours. Overall, there appears to be more mid-day activity on weekends. The author has also provided a ggplot() output as a pdf file.