Reproducible Research Project

In this project, we will examine an activity data set that contains information on how many steps the wearer of a personal activity monitoring device took during 5 minute intervals on days the device was worn. The data set contains measurements on three variables: steps taken, date, and an identifier for the 5 minute increment reported. The data set has a total of 17,568 observations.

We will first begin by reading the data set into R with the read.csv function. This is performed with the code below.

```
activity = read.csv("activity.csv")
```

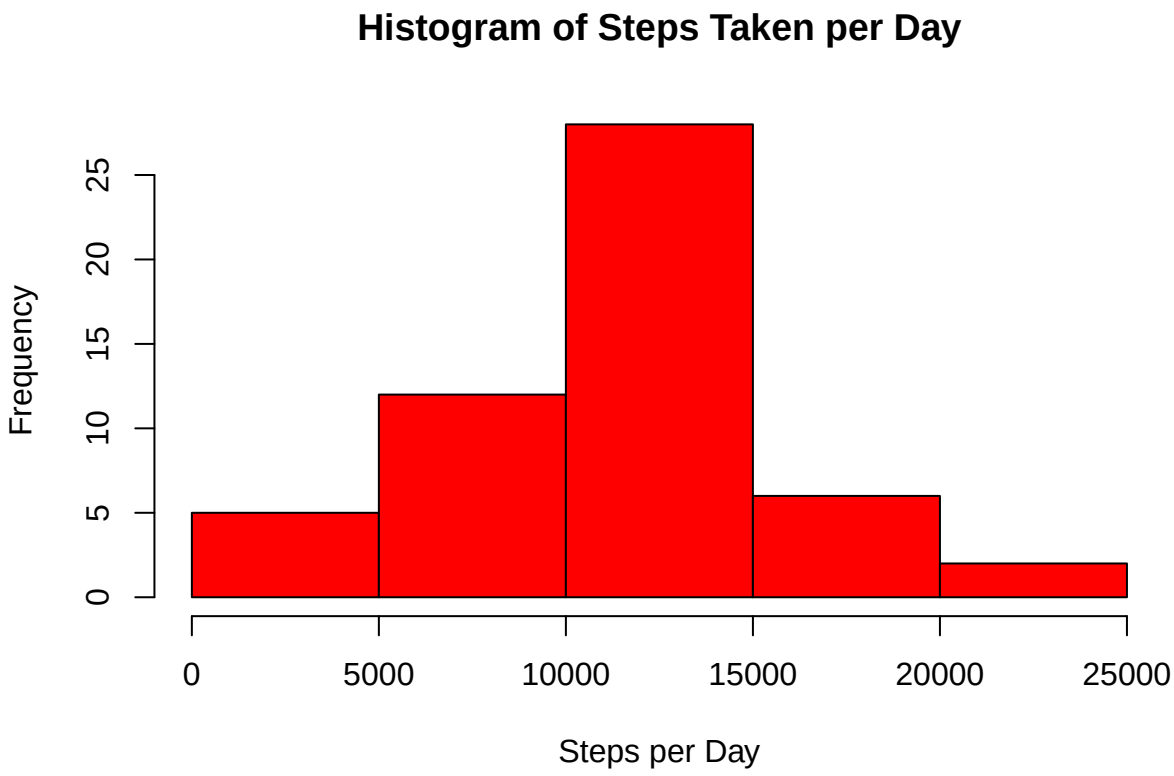
For the next portion of the project, we will use the dplyr package to handle some of the data transformation steps. Therefore we will begin by loading this package in the next code chunk. Here we will first calculate the total number of steps taken per day and then generate a histogram of steps per day.

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.2.1
```

```
##
## 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
```

```
# Remove missing values of steps since we can ignore those here.
act2 = filter(activity, steps != "NA")

# The data set will now be grouped by date so we can look at daily steps.
act3 = group_by(act2, date)

# We now summarize the data set into a set containing a variable of total steps
# taken per day.
act4 = summarize(act3, daily = sum(steps))

# We can now generate a histogram of daily steps.
hist(act4$daily, xlab = "Steps per Day",
     main = "Histogram of Steps Taken per Day", col = "Red")
```



Then we find the mean and median number of steps taken per day.

```
# This code produces the mean of daily steps.
mean(act4$daily)
```

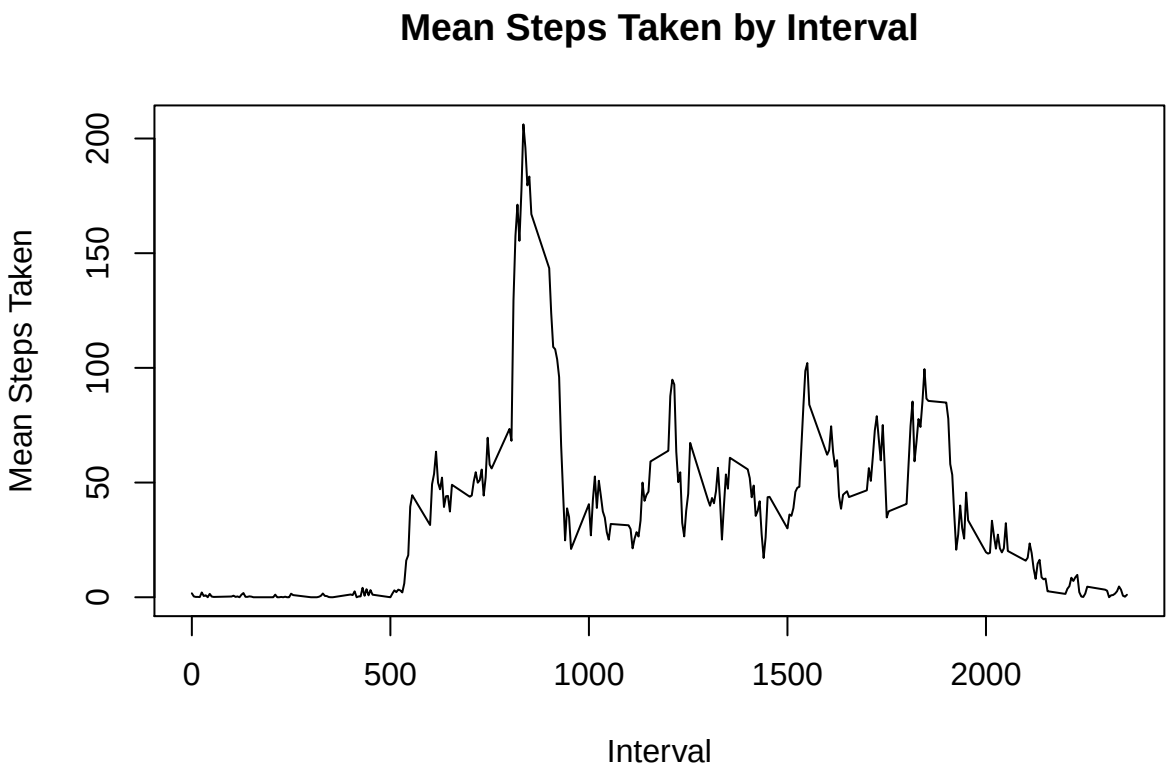
```
## [1] 10766.19
```

```
# This code produces the median of daily steps.
median(act4$daily)
```

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

We will now investigate the average daily activity pattern across 5 minute intervals. We will do this by producing a time series plot of average steps taken during each 5 minute increment averaged over all days. We again ignore missing values. The code to produce this plot is provided below.

```
# The data will first be grouped by the 5 minute intervals.
act5 = group_by(act2, interval)

# We will now calculate mean steps within each 5 minute increment.
act6 = summarize(act5, mnstep = mean(steps))

# The code below now produces the time series plot with no error bars.
plot(act6$interval, act6$mnstep, type = "l",
     xlab = "Interval", ylab = "Mean Steps Taken",
     main = "Mean Steps Taken by Interval")
```



We will now determine which interval, on average across all days observed, contains the highest average number of steps.

```
act7 = filter(act6, mnstep == max(mnstep))
act7
```

```
## Source: local data frame [1 x 2]
##
##   interval   mnstep
## 1      835 206.1698
```

From the above printed data set, we conclude that interval 835 has the highest average number of steps.

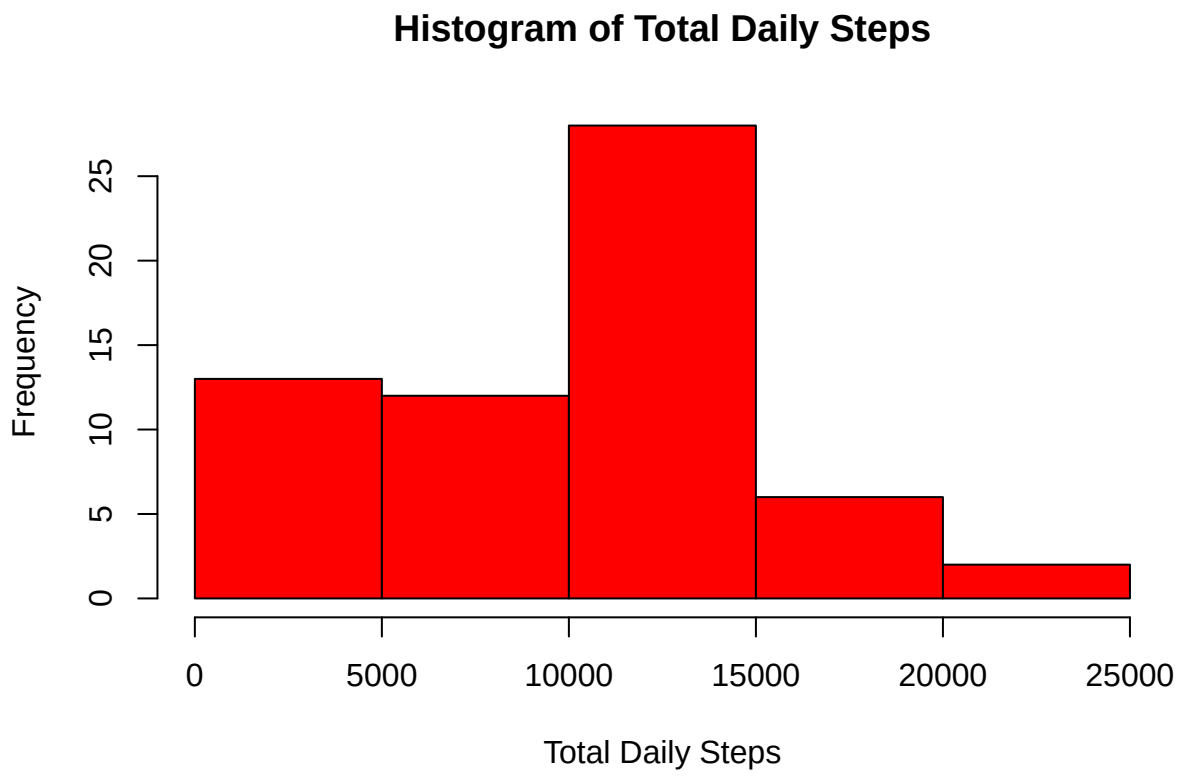
We will now begin working with the missing values in the original data set. We will first count the number of NAs in the steps variable.

```
sum(is.na(activity$steps))
```

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

From the above output, there are 2304 missing values of the steps variable. For our imputation, we will impute the median value of steps for the interval of the missing value, since the step variable is an integer. We will create a new data set with the imputed values replacing the missing values below and generate a new histogram of the daily number of steps.

```r
# Make a new variable of the median values of the steps variable.
act8 = summarize(act5, medstep = median(steps))

# Add the median variable to a joint data set.
act9 = inner_join(activity, act8, by = c("interval"))

# Impute the median variable into the original steps variable where there are NA
# values. This portion was verified after running to ensure that all NA values
# were replaced correctly.
for(i in 1:length(act9$steps)){
    if(is.na(act9$steps[i])){
        act9$steps[i] = act9$medstep[i]
    }
}

# The following code will be used to generate a histogram of total number of steps
# taken each day.
act10 = group_by(act9, date)

act11 = summarize(act10, daily = sum(steps))

hist(act11$daily, main = "Histogram of Total Daily Steps",
     xlab = "Total Daily Steps", col = "Red")
```



Comparing to the first histogram, we see significantly more zeroes in the histogram with the imputed data. This makes sense since most of the median values at each interval are equal to zero. This may indicate that

the median is not the best value to use for imputation, and that a method such as multiple imputation based on the empirical distribution may work better. Below we will investigate the new mean and median values of total daily steps with the imputed data included.

```
mean(act11$daily)
```

```
## [1] 9503.869
```

```
median(act11$daily)
```

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

The mean and median values of 9504 and 10395 are substantially lower than what we observed without imputation. The new data is also more right-skewed than the original data, most likely due to the addition of many 0 values in place of NA values.

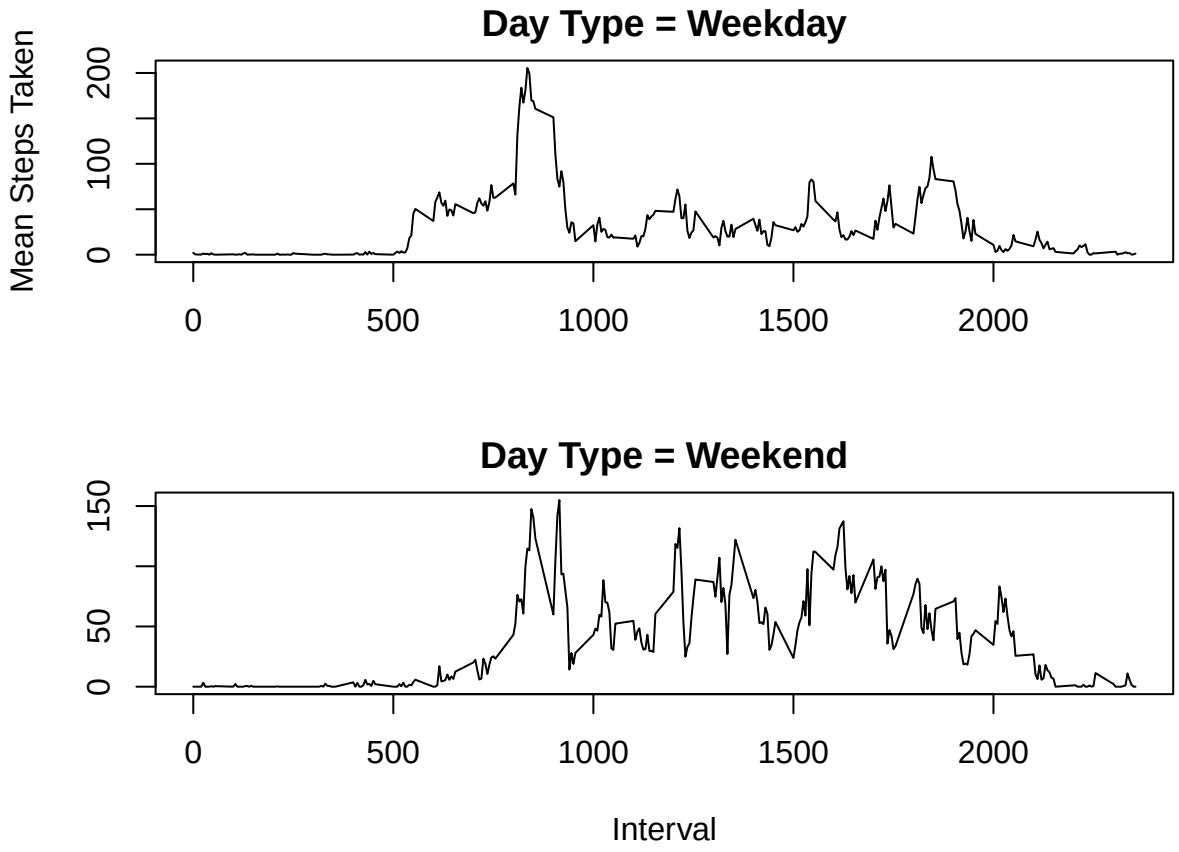
We will continue using the data set with imputed missing values for the next portion of the project. We will first add a factor variable called daytype with values weekday and weekend using the weekdays() function as an intermediate. The code is given below.

```
act10 = mutate(act9, weekday = weekdays(as.POSIXlt(date)))

daytype = rep(NA, nrow(act10))

for(i in 1:nrow(act10)){
    if(act10$weekday[i] == "Saturday" | act10$weekday[i] =="Sunday"){
        daytype[i] = "Weekend"
    }
    else{
        daytype[i] = "Weekday"
    }
}

daytype = as.factor(daytype)

act11 = cbind(act10,daytype)
```

We will now make a panel plot of the mean steps taken by the 5 minute increments split by weekdays or weekend days. The code is provided below.

```
act12 = group_by(act11, interval)

act13 = act12[which(act12$daytype == "Weekday"),]

act14 = act12[which(act12$daytype == "Weekend"),]

act15 = summarize(act13, intmean = mean(steps))

act16 = summarize(act14, intmean = mean(steps))

par(mfrow = c(2,1), mar=c(4,4,2,2))
plot(act15$interval, act15$intmean, type = "l",
```

```
     xlab = "", ylab = "Mean Steps Taken",
     main = "Day Type = Weekday")
plot(act16$interval, act16$intmean, type = "l",
     xlab = "Interval", ylab = "",
     main = "Day Type = Weekend")
```



From the above plot, we see that the activity peaks are generally higher during the weekdays, but the activity distribution is more level throughout the day during the weekends.