Reproducible Research Project #1

This assignment maks use of data frm a personnel activity monitoring device.

The device collects data at 5 minute intervals through out the day.

Data collected during the months of Octber and November, 2012

setwd("/Users/sstone25/datasciencecoursera/Reproducible Research")

Part 1 - Loading and preprocessing the data

===

Show any code that is needed to:

Load the data (i.e. read.csv())

Process/transform the data (if necessary) into a format sutable for your analysis

download and store the file in the Working directory

Load data with read.csv(). Used colClasses to convert date column from Factor to Character

```
activity<-read.csv("activity.csv",  header = TRUE, sep = ",", colClasses = c("numeric", "character", "integer"))
```

```
activity2 <- na.omit(activity)
```

Part #2 - What is mean total number of steps taken per day?

==

Calculate the total number of steps taken per day

```
activitySteps <- aggregate(activity2$steps, list(Date = activity2$date), FUN = "sum")$x
activitySteps
```

```
##  [1]   126 11352 12116 13294 15420 11015 12811  9900 10304 17382 12426
## [12] 15098 10139 15084 13452 10056 11829 10395  8821 13460  8918  8355
## [23]  2492  6778 10119 11458  5018  9819 15414 10600 10571 10439  8334
## [34] 12883  3219 12608 10765  7336    41  5441 14339 15110  8841  4472
## [45] 12787 20427 21194 14478 11834 11162 13646 10183  7047
```

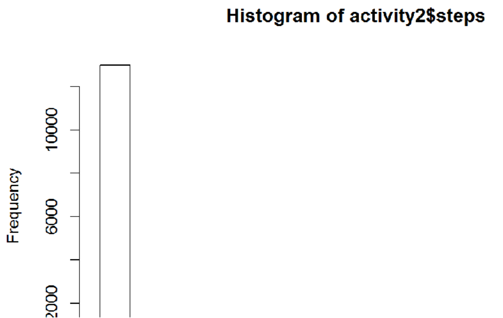
Calculate and report the mean of the total number of steps taken per day

```
stepsMean <- mean(activitySteps)
stepsMean
```

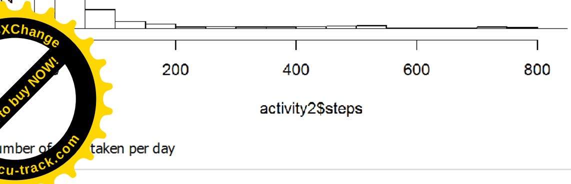
```
## [1] 10766.19
```

Make a histogram of the total number of steps taken each day

```
steps_hist <- hist(activity2$steps)
```



# Calculate and report the median of the



[illegible] number of [illegible] taken per day

```
stepsMedian <- median(activitySteps)
stepsMedian
```

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

Part #3 What is the average daily activity pattern?

==

Make a time series plot (i.e. type = “l”)ofthe 5-minute interval (x-axis) and the average . # number of steps taken, averaged across all days (y-axis)

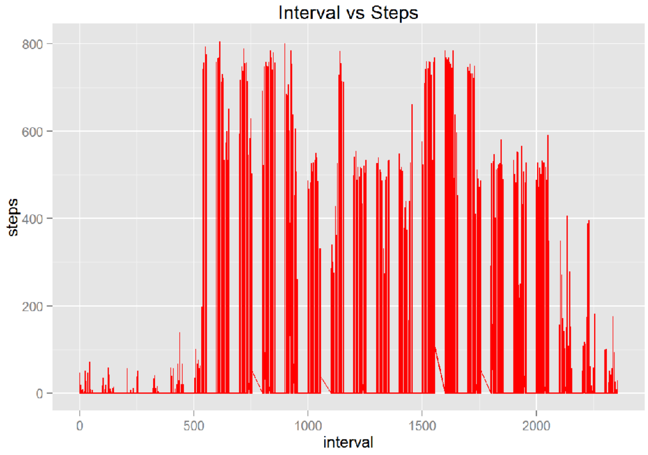
```
activitySteps2<- aggregate(activity2$steps, list(interval = activity2$interval), FUN = "sum")$x
activitySteps2
```

```
##   [1]    91    18     7     8     4   111    28    46     0    78    16
##  [12]     7    17    36     8    18     0    59    97     9     9    20
##  [23]    14     0     0     0    60     0     0     7     0    12     0
##  [34]     0    82    50     0     0     0     0    11    33    86    31
##  [45]    26     4     0     0    63    50   136     0    18    19   218
##  [56]    35   185    44   165    59     0    83   159   119   176   157
##  [67]   111   321   849   972  2091  2358  1669  2611  2850  3363  2648
##  [78]  2495  2764  2085  2333  2341  1980  2599  2322  2352  2677  2889
##  [89]  2646  2702  2951  2349  2770  3686  3066  2976  3889  3615  6860
## [100]  8349  9071  8236  9397 10927 10384  9517  9720  8852  7603  6574
## [111]  5783  5730  5497  5086  3509  2397  1314  2054  1854  1116  2150
## [122]  1430  2248  2791  2063  2692  2347  1983  1839  1502  1330  1693
## [133]  1662  1573  1130  1354  1504  1403  1772  2649  2228  2364  2440
## [144]  3137  3385  4648  5027  4917  3360  2659  2887  1718  1406  2000
## [155]  2388  3566  2244  2114  2293  2172  2451  2991  2266  1332  2118
## [166]  2838  2508  3223  2955  2754  2310  2581  1880  1990  2218  1458
## [177]   907  1382  2312  2320  1591  1912  1881  2059  2436  2531  2551
## [188]  3462  4394  5229  5412  4450  3293  3399  3951  3348  3016  3168
## [199]  2325  2044  2367  2409  2449  2315  2471  2984  2688  3245  3854
## [210]  4184  3654  3162  3980  2995  1843  1985  2156  3075  3959  4522
## [221]  3141  3592  4118  3935  4523  5271  4589  4537  4498  4125  3076
## [232]  2828  1925  1098  1452  2121  1601  1354  2420  1777  1040  1008
## [243]  1025  1767  1421  1122  1447  1131  1036  1130  1712  1068   845
## [254]   913  1243  1020   660   425   777   864   460   413   431   139
## [265]    77   195   255   451   375   461   517   117    17     6    85
## [276]   244   175   151     0    44    51    84   138   249   175    34
## [287]    12    57
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.1.3
```

```
ggplot(activity2, aes(x = interval, y = steps)) + geom_line(color = "red") + labs(title = "Interval vs Steps")
```



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

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.1.3
```

```
## 
## Attaching package: 'dplyr'
## 
## The following object is masked from 'package:stats':
## 
```

```
##     filter
##
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
max_steps2 <- arrange(activity2, desc(steps))
head(max_steps2, n = 1L)
```

```
##   steps       date interval
## 1   806 2012-11-27      615
```

Part #4 Imputing missing values

Note that there are a number of days/intervals where there are missing values (coded as NA). # The presence of missing days may introduce bias into calculations or summaries of the data.

Calculate and report the total number of missing values in the dataset (i.e. the total # # # number of rows with NAs)

```
na_dat <- sum(is.na(activity))
head(na_dat, n = 1L)
```

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

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 that 5-minute interval, etc.

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

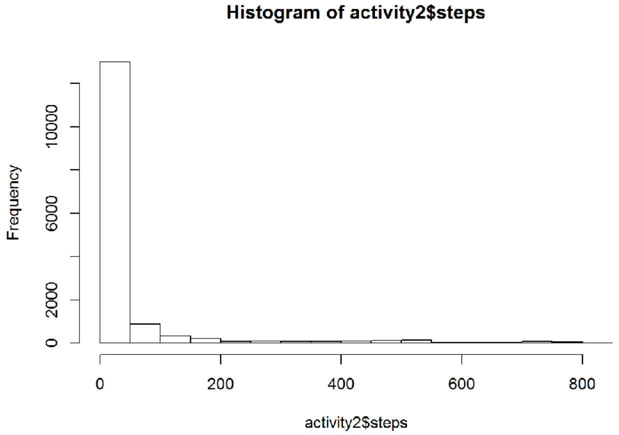
```
gnrlMean <- mean(activity2$steps)
gnrlMean
```

```
## [1] 37.3826
```

```
new_activity <- activity
for (i in 1:nrow(new_activity)) {
    if (is.na(new_activity$steps[i])) {
        new_activity$steps[i] <- 37.3
    }
}
View(new_activity$steps)
```

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

```
steps_hist2 <- hist(activity2$steps)
```



Calculate and report the mean 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?

```
activitySteps2 <- aggregate(new_activity$steps, list(Date = activity$date), FUN = "sum")$x
stepsMean2 <- mean(activitySteps2)
stepsMean2
```

```
## [illegible]07
```

Calculate and report the median total number of steps taken per day.

```
steps2Median <- median(activitySteps2)
steps2Median
```

```
## [1] 10742.4
```

Do these values differ from the estimates from the first part of the assignment?

Mean

Part #1 = 10766.19

Part #2 = 1981.278

Median

Part #1 = 10765

Part #2 = 1808

What is the impact of imputing missing data on estimates of total daily number of steps?

The results for the part #1 of the excercise are much higher and the part #2.

# Part #5 - Are there differences in activity patterns between weekdays and weekends?

For this part the weekdays() function may be of some help here. Use the dataset with the # # filled-in missing values for this part.

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_activity[, "new_weekdays"] <- new_weekdays
str(new_weekdays)
```

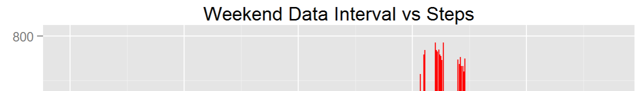
```
##  chr [1:17568] "Monday" "Monday" "Monday" "Monday" "Monday" ...
```

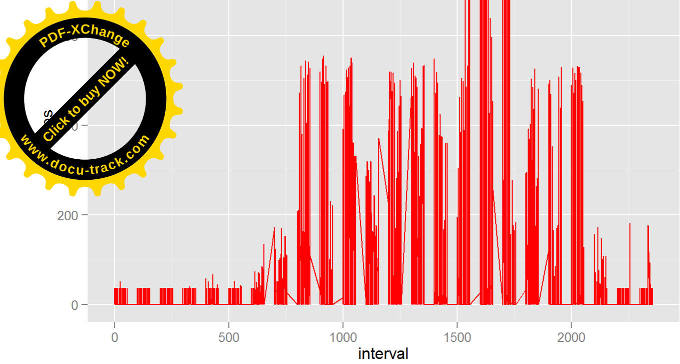
Make a panel plot containing a time series plot (i.e. type = “l”) of the 5-minute interval # (x-axis) and the average number of steps taken, averaged across all weekday days or weekend # days (y-axis). See the README file in the GitHub repository to see an example of what this # plot should look like using simulated data.

```
week_weekend <- new_activity[new_activity$new_weekdays == "Saturday" | new_activity$new_weekdays == "Sunday" | new_activity$new_wee
kdays == "Monday" | new_activity$new_weekdays == "Tuesday" | new_activity$new_weekdays == "Wednesday" | new_activity$new_weekdays =
= "Thursday" | new_activity$new_weekdays == "Friday", ]
str(week_weekend)
```

```
## 'data.frame':    17568 obs. of  4 variables:
##  $ steps       : num  37.3 37.3 37.3 37.3 37.3 37.3 37.3 37.3 37.3 37.3 ...
##  $ date        : chr  "2012-10-01" "2012-10-01" "2012-10-01" "2012-10-01" ...
##  $ interval    : int  0 5 10 15 20 25 30 35 40 45 ...
##  $ new_weekdays: chr  "Monday" "Monday" "Monday" "Monday" ...
```

```
week_weekend$newDate <- NULL
newData <- filter(week_weekend, new_weekdays == "Saturday" | new_weekdays == "Sunday")
ggplot(newData, aes(x = interval, y = steps)) + geom_line(color = "red") + labs(title = "Weekend Data Interval vs Steps")
```




s
200
0
0
500
1000
1500
2000
interval