Coursera 05 Reproducible Research

Jun Li *jun.li3@bms.com*

Contents

0.1 Code for reading in the dataset and/or processing the data

```
act <- read.csv("activity.csv", header = TRUE, sep = ",")
head(act)
```

```
##   steps       date interval
## 1    NA 2012-10-01        0
## 2    NA 2012-10-01        5
## 3    NA 2012-10-01       10
## 4    NA 2012-10-01       15
## 5    NA 2012-10-01       20
## 6    NA 2012-10-01       25
```

```
tail(act)
```

```
##       steps       date interval
## 17563    NA 2012-11-30     2330
## 17564    NA 2012-11-30     2335
## 17565    NA 2012-11-30     2340
## 17566    NA 2012-11-30     2345
## 17567    NA 2012-11-30     2350
## 17568    NA 2012-11-30     2355
```

```
dim(act)
```

```
## [1] 17568     3
```

```
act_narm <- act[!is.na(act$steps),]
head(act_narm)
```

```
##     steps       date interval
## 289     0 2012-10-02        0
## 290     0 2012-10-02        5
## 291     0 2012-10-02       10
## 292     0 2012-10-02       15
## 293     0 2012-10-02       20
## 294     0 2012-10-02       25
```

```
tail(act_narm)
```

```
##       steps       date interval
## 17275     0 2012-11-29     2330
## 17276     0 2012-11-29     2335
## 17277     0 2012-11-29     2340
## 17278     0 2012-11-29     2345
## 17279     0 2012-11-29     2350
## 17280     0 2012-11-29     2355
```

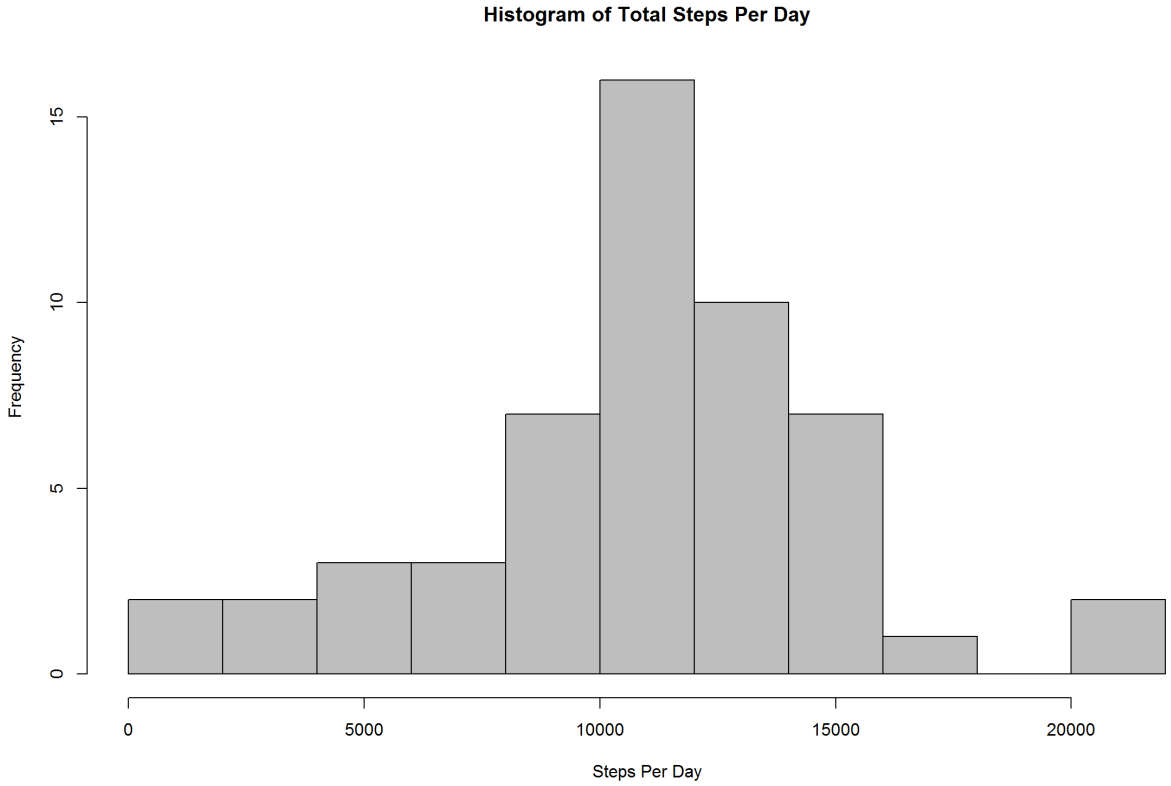
```
dim(act_narm)
```

```
## [1] 15264     3
```

0.2 Histogram of the total number of steps taken each day

```
actDate <- group_by(act_narm, date)
totStepsDate <- summarise(actDate, totalSteps = sum(steps))

hist(totStepsDate$totalSteps, main = "Histogram of Total Steps Per Day",
     xlab = "Steps Per Day", breaks = 10, col = "grey")
```




```
# or use aggregate()
summary(totStepsDate)
```

```
##          date      totalSteps
##  2012-10-02: 1   Min.   :   41
##  2012-10-03: 1   1st Qu.: 8841
##  2012-10-04: 1   Median :10765
##  2012-10-05: 1   Mean   :10766
##  2012-10-06: 1   3rd Qu.:13294
##  2012-10-07: 1   Max.   :21194
##  (Other)   :47
```

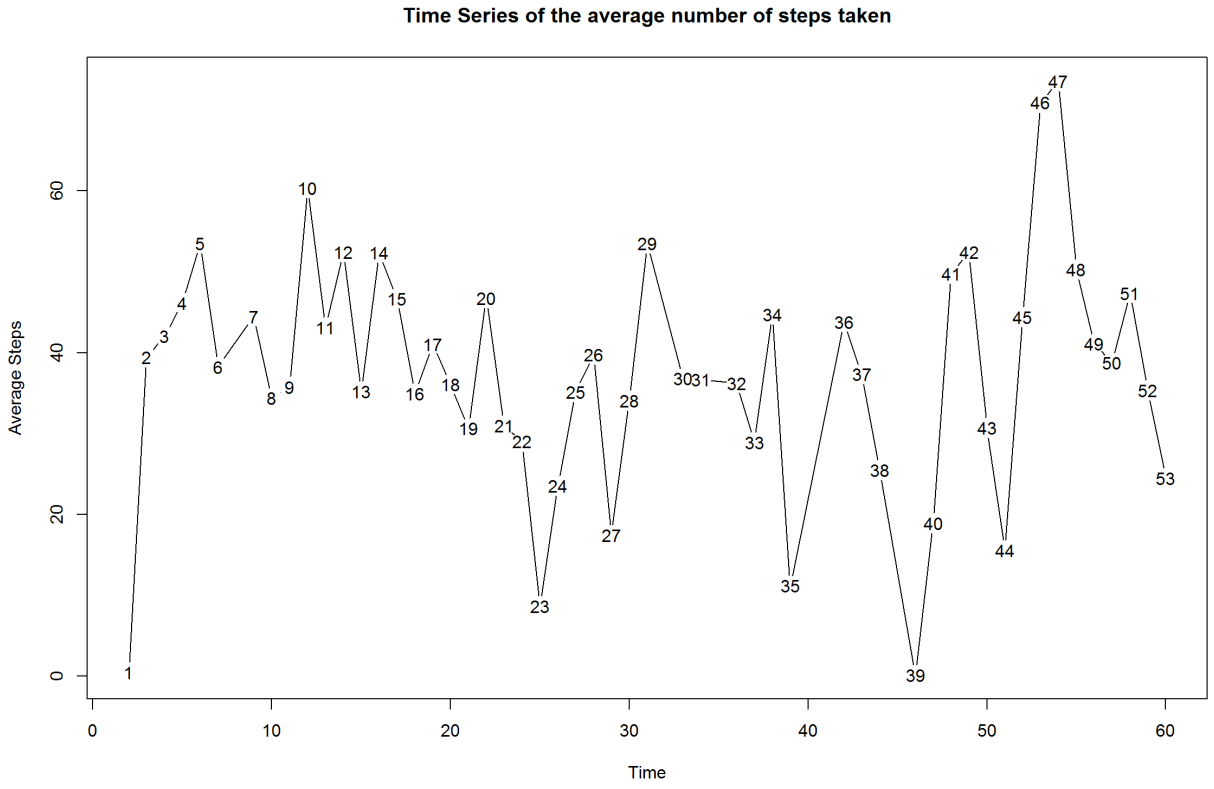
0.3 Mean and median number of steps taken each day

```
meanStepsDate = mean(totStepsDate$totalSteps)
medianStepsDate = median(totStepsDate$totalSteps)

# or try this
totStepsDate_na = summarise(group_by(act, date), totalSteps = sum(steps))
mean_na = mean(totStepsDate_na$totalSteps, na.rm = TRUE)
median_na = median(totStepsDate_na$totalSteps, na.rm = TRUE)
```

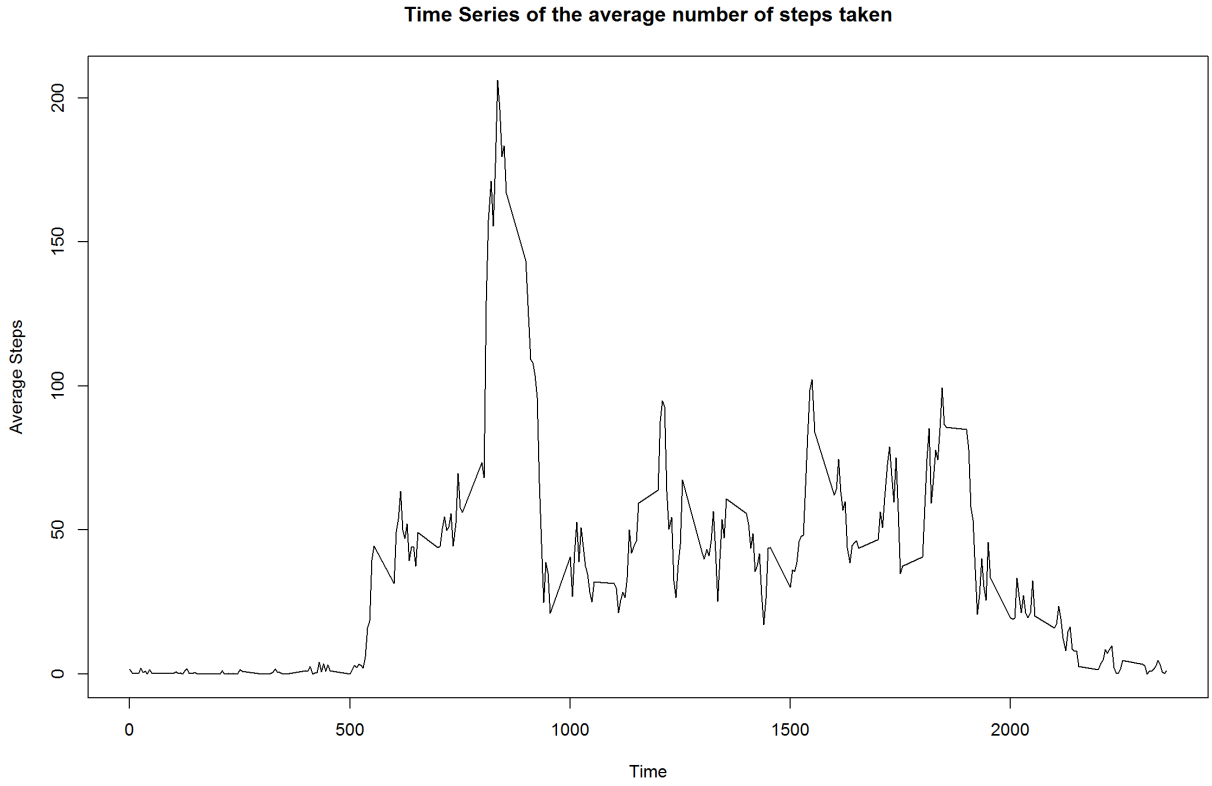
0.4 Time series plot of the average number of steps taken

```
averageSteps = summarise(group_by(act_narm, date), aveSteps = mean(steps))
par(mfrow=c(1, 1))
plot.ts(averageSteps$date, averageSteps$aveSteps, main = "Time Series of the average number of steps ta
        xlab = "Time", ylab= "Average Steps")
```



0.5 The 5-minute interval that, on average, contains the maximum number of steps

```
averageSteps_int = summarise(group_by(act_narm, interval), aveSteps = mean(steps))
plot(averageSteps_int, type = "l", main = "Time Series of the average number of steps taken",
     xlab = "Time", ylab= "Average Steps")
```




```
summary(averageSteps_int$aveSteps) #orto find out at exact interval
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   2.486  34.110  37.380  52.830 206.200
```

```
max = averageSteps_int[which.max(averageSteps_int$aveSteps), ]
```

0.6 Code to describe and show a strategy for imputing missing data

```
for(cell in names(act)) {
    missing <- sum(is.na(act[,cell]))
    if (missing > 0) {
        print(c(cell,missing))
    }
}
```

```
## [1] "steps" "2304"
```

```
#simple way
total_NA = sum(is.na(act$steps))

# Devise a strategy for filling in all of the missing values in the dataset. Use the mean for that
# 5-minute interval

# 1) make a copy of the original data.frame "act"
# 2) find the index of the missing "step"
# 3) find the corresponding "interval" value, subsetting
# 4) assign the average interval value  "aveSteps" to the missing step in the new table

act_new <- act
for (i in 1:nrow(act_new)) {
  if (is.na(act_new$steps[i])) {
    #interval_value <- act_new$interval[i]
    steps_value <- averageSteps_int[averageSteps_int$interval == act_new$interval[i],]
    act_new$steps[i] <- steps_value$aveSteps
  }
}
```

0.7 Histogram of the total number of steps taken each day after missing values are imputed

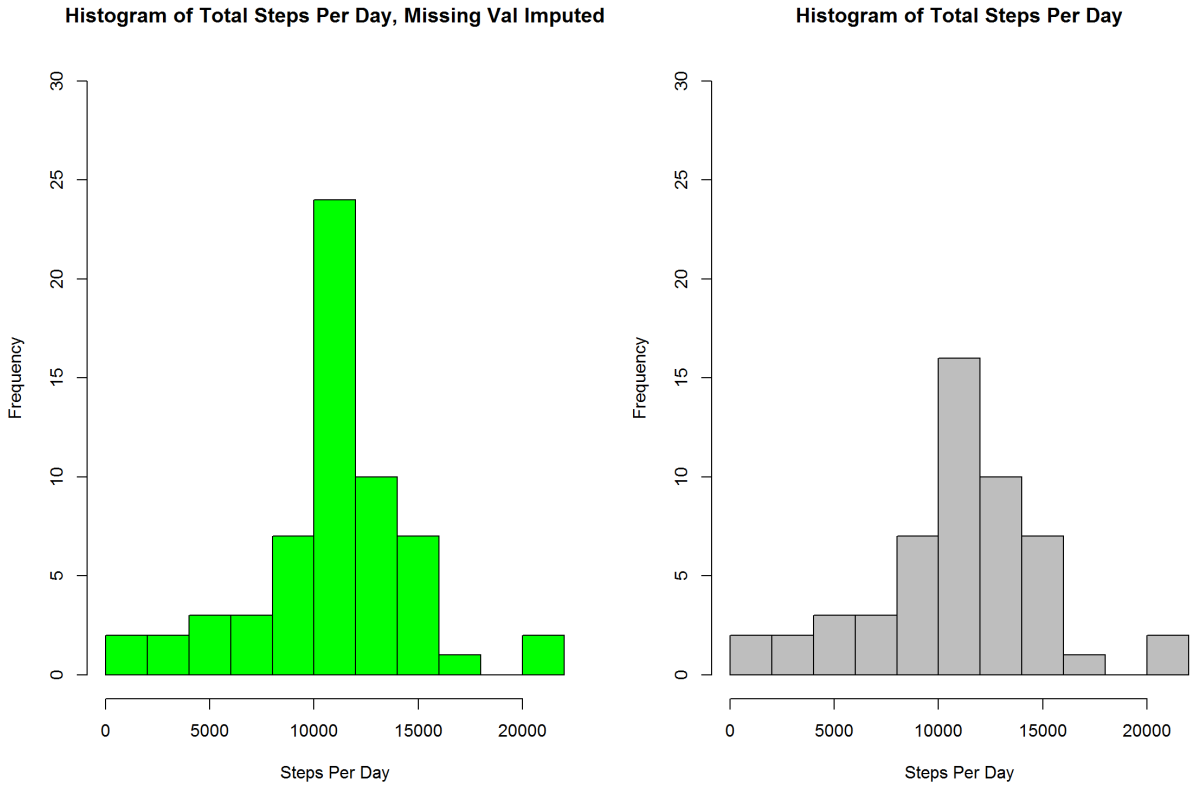
Make a histogram of the total number of steps taken each day and Calculate and report the mean and median total number of steps taken per day. Do these values differ from the estimates from the first part of the assignment? What is the impact of imputing missing data on the estimates of the total daily number of steps?

```
actDate_new <- group_by(act_new, date)
totStepsDate_new <- summarise(actDate_new, totalSteps_new = sum(steps))

par(mfrow=c(1, 2))

hist(totStepsDate_new$totalSteps_new, main = "Histogram of Total Steps Per Day, Missing Val Imputed", xl

hist(totStepsDate$totalSteps, main = "Histogram of Total Steps Per Day", xlab = "Steps Per Day", breaks
```




```
summary(totStepsDate_new$totalSteps_new)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      41    9819   10770   10770   12810   21190
```

```
mean(totStepsDate_new$totalSteps_new)
```

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

```
median(totStepsDate_new$totalSteps_new)
```

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

```
# With missing value imputed, the histogram of the total number of steps taken each day is differ
# from the estimates from the first part of the assignment

# The impact of imputing missing data on the estimates of the total daily number of steps: mean and med
# of total number of steps per day are the same
```

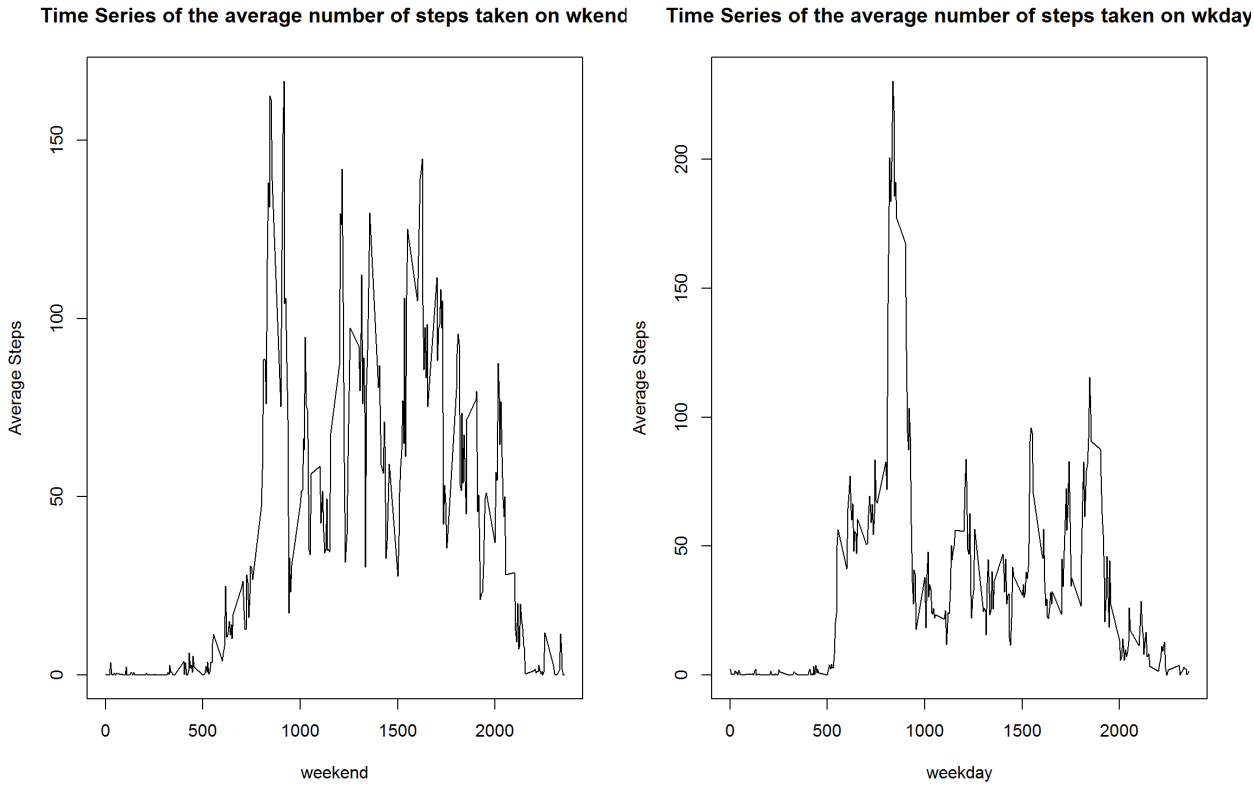
0.8 Panel plot comparing the average number of steps taken per 5-minute interval across weekdays and weekends

```
act_new$weekend = chron::is.weekend(act_new$date)
act_new_wkend = act_new[act_new$weekend == "TRUE",]
act_new_wkday = act_new[act_new$weekend == "FALSE",]
total = sum(count(act_new_wkday), count(act_new_wkend))

aveStepsWkend = summarise(group_by(act_new_wkend, interval), aveStepsWkend = mean(steps))
aveStepsWkday = summarise(group_by(act_new_wkday, interval), aveStepsWkday = mean(steps))

par(mfrow=c(1, 2))

plot(aveStepsWkend, type = "l", main = "Time Series of the average number of steps taken on wkend",
     xlab = "weekend", ylab= "Average Steps")
plot(aveStepsWkday, type = "l", main = "Time Series of the average number of steps taken on wkday",
     xlab = "weekday", ylab= "Average Steps")
```



0.9 System Information

Time required to process this report: *1.131064 secs*

R session information:

```
## R version 3.2.3 (2015-12-10)
## Platform: i386-w64-mingw32/i386 (32-bit)
## Running under: Windows 7 (build 7601) Service Pack 1
```

```
##
## locale:
## [1] LC_COLLATE=English_United States.1252
## [2] LC_CTYPE=English_United States.1252
## [3] LC_MONETARY=English_United States.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.1252
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] chron_2.3-47  ggplot2_2.0.0 dplyr_0.4.3   knitr_1.12.3
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.3      digest_0.6.9     assertthat_0.1   plyr_1.8.3
##  [5] grid_3.2.3       R6_2.1.2         gtable_0.1.2     DBI_0.3.1
##  [9] formatR_1.2.1    magrittr_1.5     scales_0.3.0     evaluate_0.8
## [13] stringi_1.0-1    lazyeval_0.1.10  rmarkdown_0.9.2  tools_3.2.3
## [17] stringr_1.0.0    munsell_0.4.2    yaml_2.1.13      parallel_3.2.3
## [21] colorspace_1.2-6 htmltools_0.3
```

‘‘‘