# Data Science Course 5, Week 2 Assignment

The following integrated document is going to contain a mixture of the following things:

a. Original question text repeated from the assignment on Coursera, so that the assignment sections in this markdown file will have the same structure as in the original,

b. My own narrative and remarks to make it flow and create context

c. R code chunks

d. R code output

e. Inline plots

Preliminary note about locale

Gianfranco Campana points out that the weekdays() function will generate days of the week based on locale, and not necessarily in English. This could hurt reproducibility in case the code then tests for hardcoded strings like 'Saturday'. Excellent point. So I am going to explicitly set locale:

```
Sys.setlocale(category = "LC_ALL", locale = "C")
```

```
## [1] "C"
```

# SECTION ONE. Loading and preprocessing the data

Remark: I assume that your working directory contains activity.csv. You can download it from here (https://d396qusza40orc.cloudfront.net/repdata%2Fdata%2Factivity.zip)

Remark: Load up several of the hipstr utility libraries:

```
library("tidyr");library("stringr");library("plyr");library("dplyr");library("knitr")
```

Remark: Also load up the comparatively unhip 'lattice':

```
library(lattice)
```

## 1a. Show any code that is needed to load the data (i.e. read.csv())

## 1b. Process/transform the data (if necessary) into a format suitable for your analysis

Remark: There is not going to be any preprocessing necessary right now. Immediately after reading in the csv, I create one subset with no NA's, and the complement, which is all NA's. This is useful later on, because the rows that are NA's tells you which date-interval pairs need a value imputed. Note that I am taking some nrow() calls so that I can verify the totals at the very end after pieces have been subsetted and then rbind'ed.

```
activity <- read.csv("activity.csv")
activity_na_false <- na.omit(activity)
activity_na_true <- subset(activity,is.na(activity$steps) == TRUE)
nrow(activity)
```

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

```
nrow(activity_na_false) + nrow(activity_na_true)
```

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

# SECTION TWO. What is mean total number of steps taken per day?

## 2a. Calculate the total number of steps taken per day

Remark: The following ddply call is like saying “roll up by date, so that each aggregate contains all the intervals for that date. And for each of the ~60 aggregates, find the total number of steps per day.”

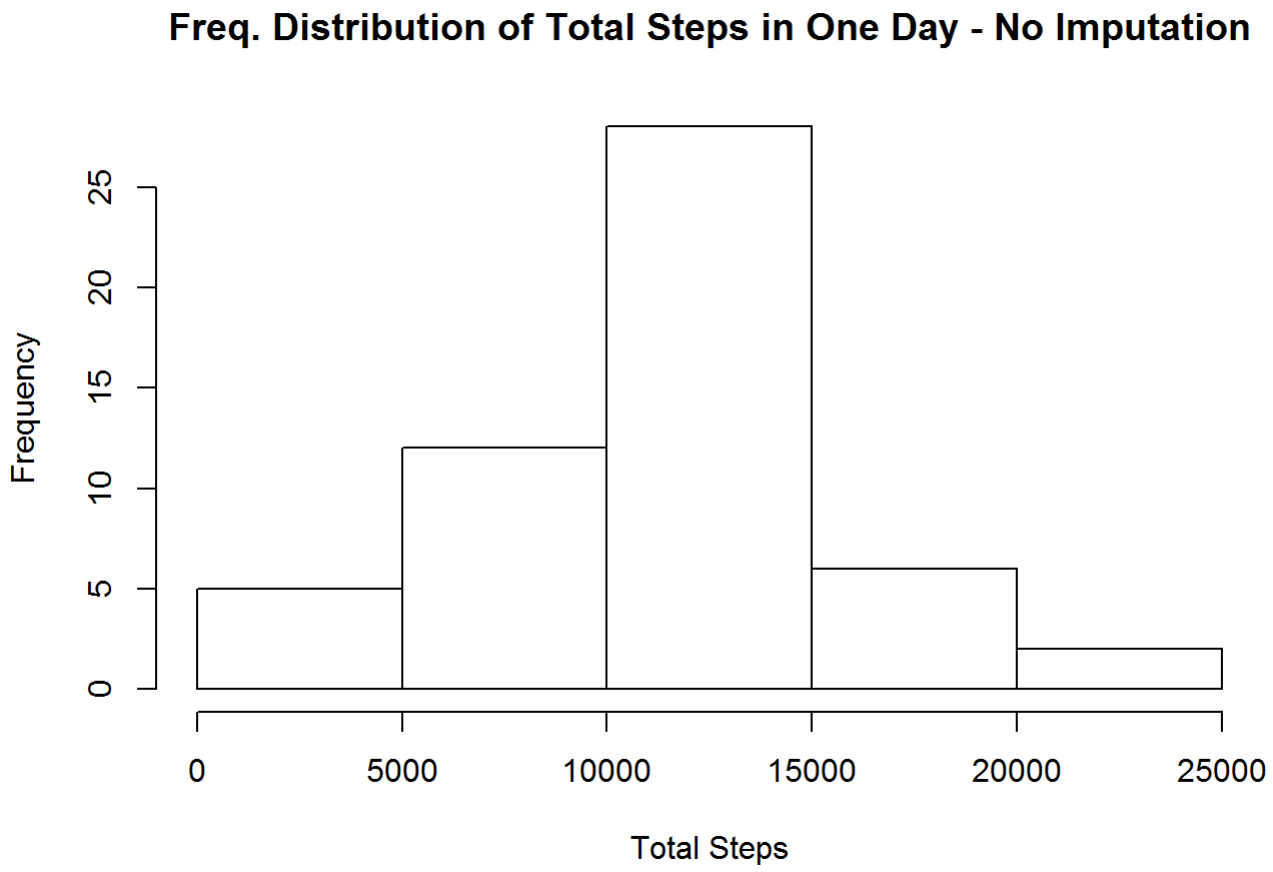
```
activity_na_false_rollup1 <- ddply(activity_na_false,.(date),summarize,sum_steps=sum(
steps))
print(activity_na_false_rollup1)
```

```
##          date sum_steps
## 1  2012-10-02       126
## 2  2012-10-03     11352
## 3  2012-10-04     12116
## 4  2012-10-05     13294
## 5  2012-10-06     15420
## 6  2012-10-07     11015
## 7  2012-10-09     12811
## 8  2012-10-10      9900
## 9  2012-10-11     10304
## 10 2012-10-12     17382
## 11 2012-10-13     12426
## 12 2012-10-14     15098
## 13 2012-10-15     10139
## 14 2012-10-16     15084
## 15 2012-10-17     13452
## 16 2012-10-18     10056
## 17 2012-10-19     11829
## 18 2012-10-20     10395
## 19 2012-10-21      8821
## 20 2012-10-22     13460
## 21 2012-10-23      8918
## 22 2012-10-24      8355
## 23 2012-10-25      2492
## 24 2012-10-26      6778
## 25 2012-10-27     10119
## 26 2012-10-28     11458
## 27 2012-10-29      5018
## 28 2012-10-30      9819
## 29 2012-10-31     15414
## 30 2012-11-02     10600
## 31 2012-11-03     10571
## 32 2012-11-05     10439
## 33 2012-11-06      8334
## 34 2012-11-07     12883
## 35 2012-11-08      3219
## 36 2012-11-11     12608
## 37 2012-11-12     10765
## 38 2012-11-13      7336
## 39 2012-11-15        41
## 40 2012-11-16      5441
## 41 2012-11-17     14339
## 42 2012-11-18     15110
## 43 2012-11-19      8841
## 44 2012-11-20      4472
## 45 2012-11-21     12787
## 46 2012-11-22     20427
## 47 2012-11-23     21194
## 48 2012-11-24     14478
## 49 2012-11-25     11834
```

```
## 50 2012-11-26     11162
## 51 2012-11-27     13646
## 52 2012-11-28     10183
## 53 2012-11-29      7047
```

## 2b. Make a histogram of the total number of steps taken each day

```
hist(activity_na_false_rollup1$sum_steps,xlab="Total Steps",main="Freq. Distribution
of Total Steps in One Day - No Imputation")
```




## 2c. Calculate and report the mean and median of the total number of steps taken per day

```
mean(activity_na_false_rollup1$sum_steps)
```

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

```
median(activity_na_false_rollup1$sum_steps)
```

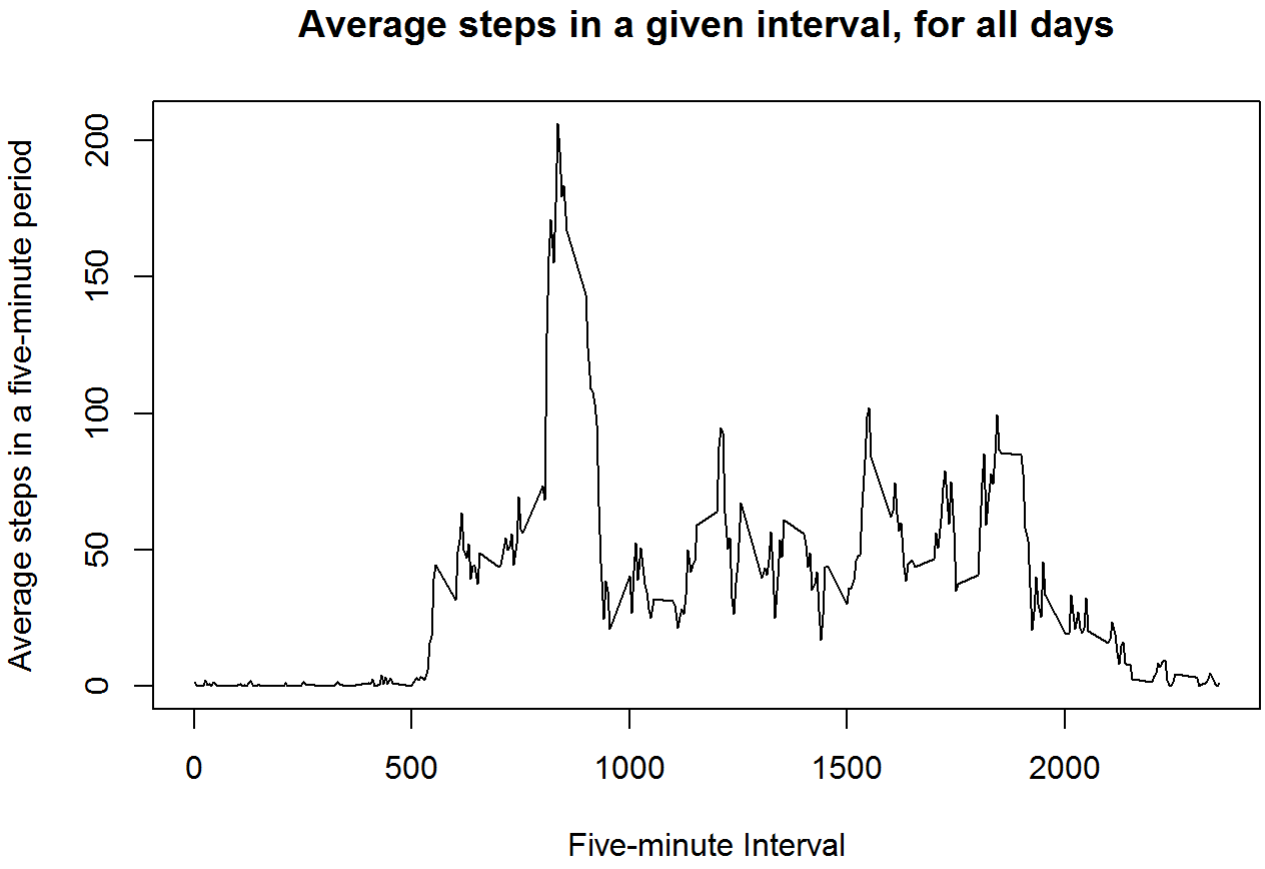
```
## [1] 10765
```

# SECTION THREE. What is the average daily activity pattern?

## 3a. Make 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 days (y-axis)

Remark: The following ddply call is in a way the “opposite” of the ddply call above. It is like saying “roll up by interval, so that each of the ~2880 rolled-up intervals will represent an aggregated operation, in this case the average steps value for all the dates for that interval.”

```
activity_na_false_rollup2 <-
ddply(activity_na_false,.(interval),summarize,mean_steps=mean(steps))
plot(activity_na_false_rollup2$interval,activity_na_false_rollup2$mean_steps,type="l"
,xlab="Five-minute Interval",main="Average steps in a given interval, for all days",y
lab="Average steps in a five-minute period")
```




## 3b. Which 5-minute interval, on average across all the days in the dataset, contains the maximum number of steps?

Remark: Do this with subset() and max().

```
max_interval <- subset(activity_na_false_rollup2, activity_na_false_rollup2$mean_step
s == max(activity_na_false_rollup2$mean_steps))
print(max_interval)
```

```
##     interval mean_steps
## 104      835   206.1698
```

# SECTION FOUR. Imputing missing values

## 4a. Calculate and report the total number of missing values in the dataset (i.e. the total number of rows with NAs)

Remark: the variable activity_na_true was subsetted above, so here to find the total number of missing values, we can simply nrow this.

```
nrow(activity_na_true)
```

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

## 4b. What is my strategy for filling in missing values?

Remark: Here's the strategy. I am not going to use impute(). I have a subset found above with just the rows that have NA. So the date-interval pairs that need a figure are already isolated. The trick will be to bring in activity_na_false_rollup2 from Part 3 above, which amounts to a handy 'lookup table' of the average steps values per interval! So I'm going to use merge and populate just the rows-with-NA with a figure from the lookup table for that interval. At this point, the complementary no-NA's and all-NA's portions of the original activity data can be joined back together using rbind(), (with a couple of nominal steps to make the columns align.)

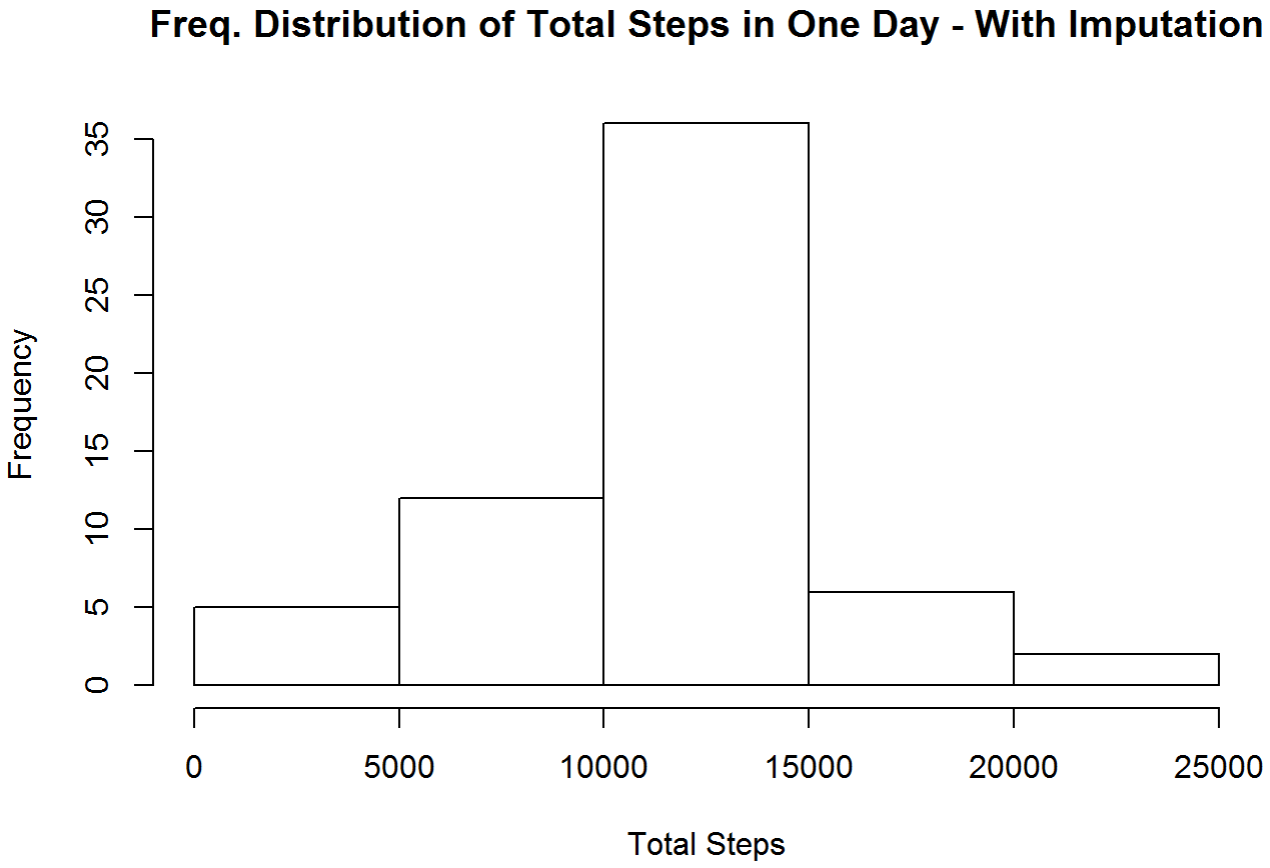
```
activity_na_derived <- merge(activity_na_true, activity_na_false_rollup2, by.x="inter
val",by.y="interval")
activity_na_derived_for_rbind <- activity_na_derived[-2]
activity_na_derived_for_rbind = rename(activity_na_derived_for_rbind,steps=mean_steps
)
```

## 4c. Create a new dataset that is equal to the original dataset but with the missing data filled in.

```
activity2 <- rbind(activity_na_false,activity_na_derived_for_rbind)
```

## 4d. Make a histogram of the total number of steps taken each day

```
activity2_rollup1 <- ddply(activity2,.(date),summarize,sum_steps=sum(steps))
hist(activity2_rollup1$sum_steps,xlab="Total Steps",main="Freq. Distribution of Total
Steps in One Day - With Imputation")
```




## 4e. Calculate and report the mean and median total number of steps taken per day.

```
mean(activity2_rollup1$sum_steps)
```

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

```
median(activity2_rollup1$sum_steps)
```

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

## 4f. Do these values differ from the estimates from the first part of the assignment?

Remark: The mean didn't change and the median increased by about 1 step - a very small change relative to the scales that we are looking at.

## 4g. What is the impact of imputing missing data on the estimates of the total daily number of steps?

Remark: Based on the very small change in the median and no change in the mean, there was essentially no change in the distribution from the earlier section. All that happened was that the raw 'n' increased by roughly 2000 datapoints. So the height of the bars increased, but there is no change in the position of the bars with respect to each other.

# SECTION FIVE. Are there differences in activity patterns between weekdays and weekends?

## 5a. 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.

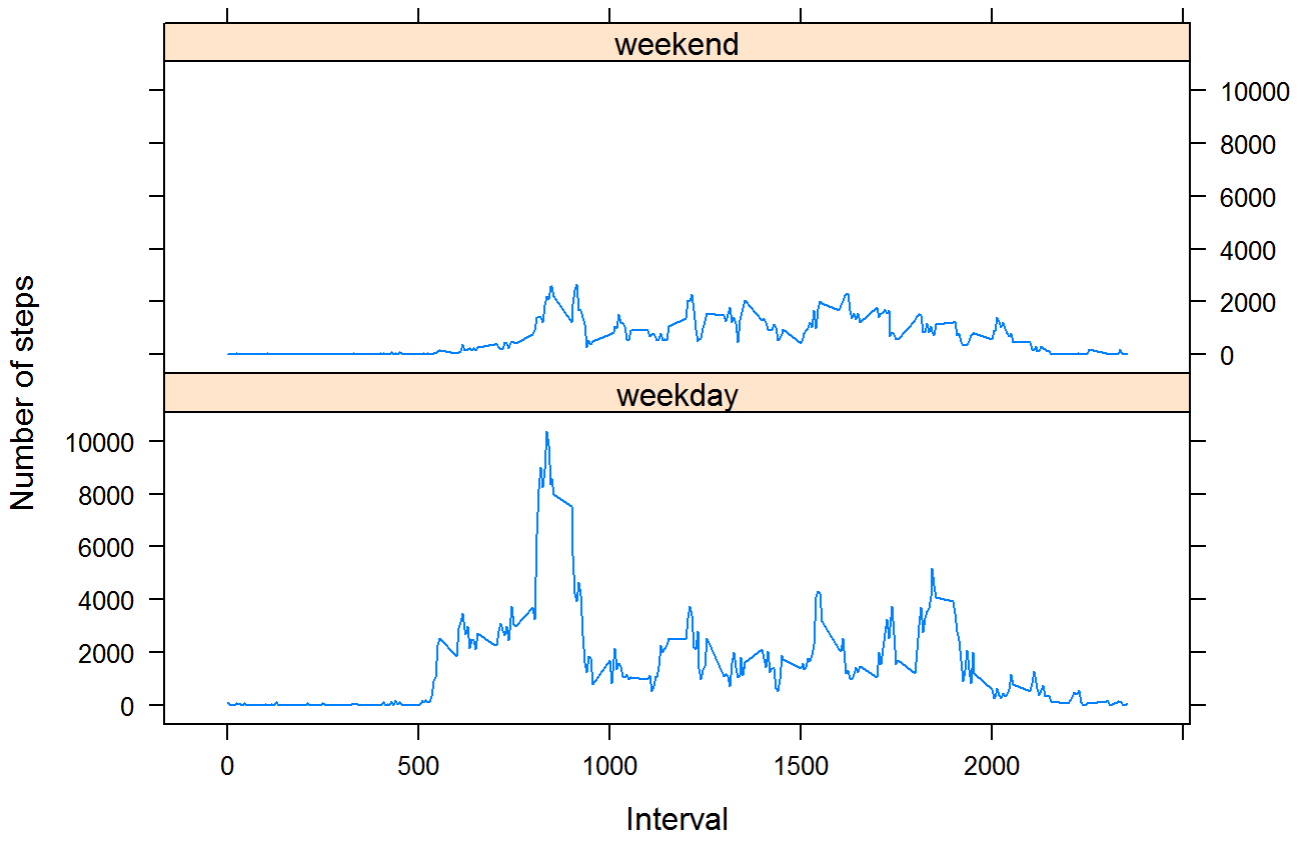
Remark: I am calling nrow() at the end of this operation as a crude checksum to verify that the dataset is the same after being carved up into complementary pieces and then rbind’ed back together. Granted, simply verifying the total rows will not pick up every possible issue.

```
activity2$DOW <- weekdays(as.POSIXct(activity2$date))
activity_weekend <- subset(activity2,DOW %in% c("Saturday","Sunday"))
activity_weekday <- subset(activity2,DOW %in% c("Monday","Tuesday","Wednesday","Thurs
day","Friday"))
activity_weekend$plotfactor <- "weekend"
activity_weekday$plotfactor <- "weekday"
activity3 <- rbind(activity_weekend,activity_weekday)
nrow(activity3)
```

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

## 5b. 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).

```
activity_weekend_rollup <- ddply(activity_weekend,.(interval),summarize,sum_steps=sum
(steps))
activity_weekday_rollup <- ddply(activity_weekday,.(interval),summarize,sum_steps=sum
(steps))
activity_weekend_rollup$day_group <- "weekend"
activity_weekday_rollup$day_group <- "weekday"
activity_for_panel <- rbind(activity_weekend_rollup,activity_weekday_rollup)
xyplot(sum_steps~interval | day_group ,data=activity_for_panel,layout=c(1,2),type="a"
,xlab="Interval",ylab="Number of steps",main="Average steps over interval, for weeken
ds versus weekdays")
```




5c. Are there differences in activity patterns between weekdays and weekends?

Remark: Yes, there is a spike in number of steps seen early in the morning on weekdays and not on weekends. This might be a reflection of people preparing for work, going to work or added activity on the job early in the day from Monday through Friday.

# SECTION SIX. Prepare literate statistical document and upload to github

Remark: My version of R told me to use render() rather than knit2html(). I did this with the clean=FALSE parameter. The image paths expected are therefore slightly different than the /figure folder, so I have adjusted these in the markdown directly for the 4 plots.