peer assessment 1

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

This assignment makes use of data from a personal activity monitoring device. This device collects data at 5 minute intervals through out the day. The data consists of two months of data from an anonymous individual collected during the months of October and November, 2012 and include the number of steps taken in 5 minute intervals each day. The variables included in this dataset are:

steps: Number of steps taking in a 5-minute interval (missing values are coded as NA)

date: The date on which the measurement was taken in YYYY-MM-DD format

interval: Identifier for the 5-minute interval in which measurement was taken

Loading and preprocessing the data

We load the data and check variable formatting, dimensions, and the first six rows with the following code:

```
activity=read.csv("activity.csv", stringsAsFactors=F)
str(activity)  ## check variable formats
```

```
## 'data.frame':    17568 obs. of  3 variables:
##  $ steps   : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ date    : chr  "10/1/2012" "10/1/2012" "10/1/2012" "10/1/2012" ...
##  $ interval: int  0 5 10 15 20 25 30 35 40 45 ...
```

```
dim(activity)  ## check dimensions
```

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

```
head(activity) ## Look at first 6 rows
```

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

Let's look at summary statistics:

```
summary(activity)
```

```
##      steps           date              interval
##  Min.   :  0.0   Length:17568       Min.   :   0
##  1st Qu.:  0.0   Class :character   1st Qu.: 589
##  Median :  0.0   Mode  :character   Median :1178
##  Mean   : 37.4                      Mean   :1178
##  3rd Qu.: 12.0                      3rd Qu.:1766
##  Max.   :806.0                      Max.   :2355
##  NA's   :2304
```

The largest interval is 2355, which is meant to be 23:55. I realized with some effort (not just the above line of code) that interval is coded as HHMM rather than just in minutes. I will recode it into minutes so that there won't be gaps. I recode as follows

```
interval.minutes = activity$interval %% 100
table(interval.minutes)
```

```
## interval.minutes
##    0    5   10   15   20   25   30   35   40   45   50   55
## 1464 1464 1464 1464 1464 1464 1464 1464 1464 1464 1464 1464
```

```
interval.hour = activity$interval %/% 100
table(interval.hour)
```

```
## interval.hour
##   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17
## 732 732 732 732 732 732 732 732 732 732 732 732 732 732 732 732 732 732
##  18  19  20  21  22  23
## 732 732 732 732 732 732
```

```
activity$interval = interval.minutes + 60*interval.hour
summary(activity$interval)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0     359     718     718    1080    1440
```

Now the largest interval is 1435, which is equal to 60*24-1, what we expect (since the first interval is 0 rather than 1).

```
60*24-1
```

```
## [1] 1439
```

We will also format the dates:

```
activity$date =
  as.POSIXct(  as.character(activity$date), format="%m/%d/%y")
str(activity)
```

```
## 'data.frame':    17568 obs. of  3 variables:
##  $ steps   : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ date    : POSIXct, format: "2020-10-01" "2020-10-01" ...
##  $ interval: num  0 5 10 15 20 25 30 35 40 45 ...
```

What is mean total number of steps taken per day?

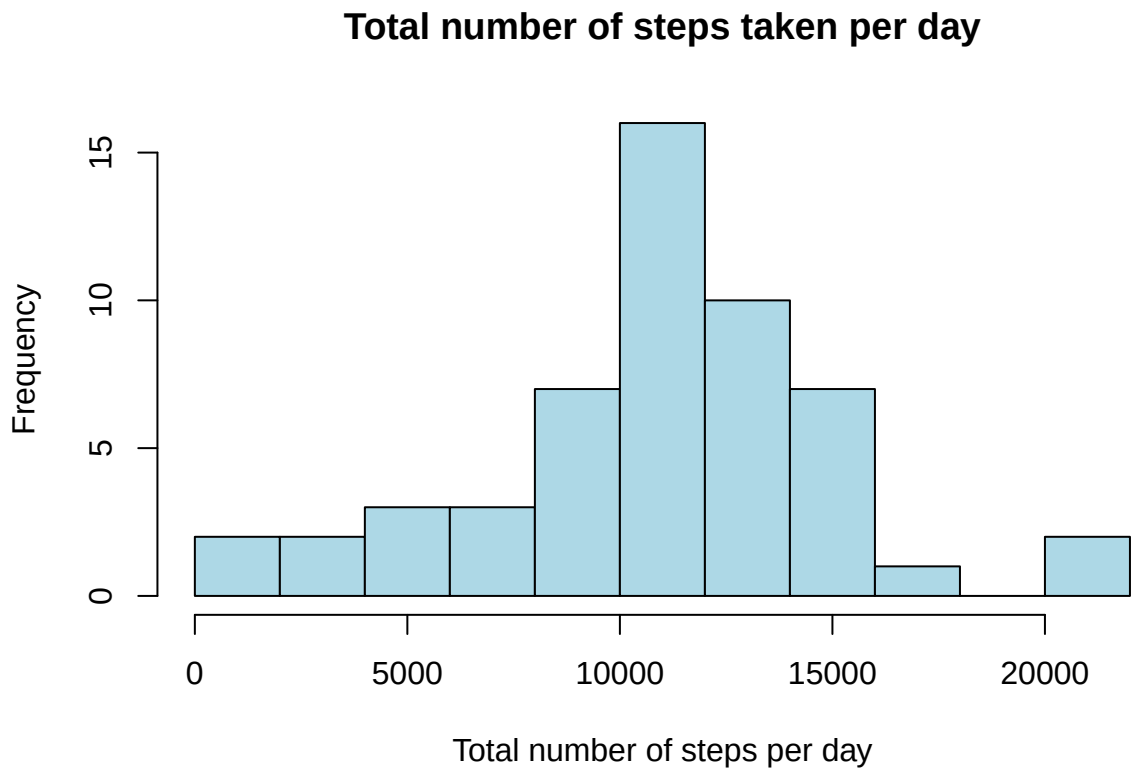
The next task is to make a histogram of total number of steps taken per day, and compute the mean and median total number of steps per day. To compute the total number of steps per day, we type:

```
total.steps = tapply(activity$steps, activity$date, sum)
```

In this calculation, if number of steps was missing for one interval during a certain day, then the total will be missing for that day.

We create the histogram with the following code:

```
hist(total.steps, main="Total number of steps taken per day",
     col="lightblue",
     breaks=12,
     xlab="Total number of steps per day")
```



We compute the mean and median, ignoring missing values, as follows:

```
mean(total.steps, na.rm=T)
```

```
## [1] 10766
```

```
median(total.steps, na.rm=T)
```

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

What is the average daily activity pattern?

We want to compute the average number of steps taken (averaging across all days) for each 5-minute interval.

We compute the average number of steps taken in each interval as follows:

```
ave.steps.per.5min = tapply(activity$steps, activity$interval, mean, na.rm=T)
```

Are any intervals missing from the above vector? Let's check:

```
length(names(ave.steps.per.5min))
```

```
## [1] 288
```

```
60*24/5
```

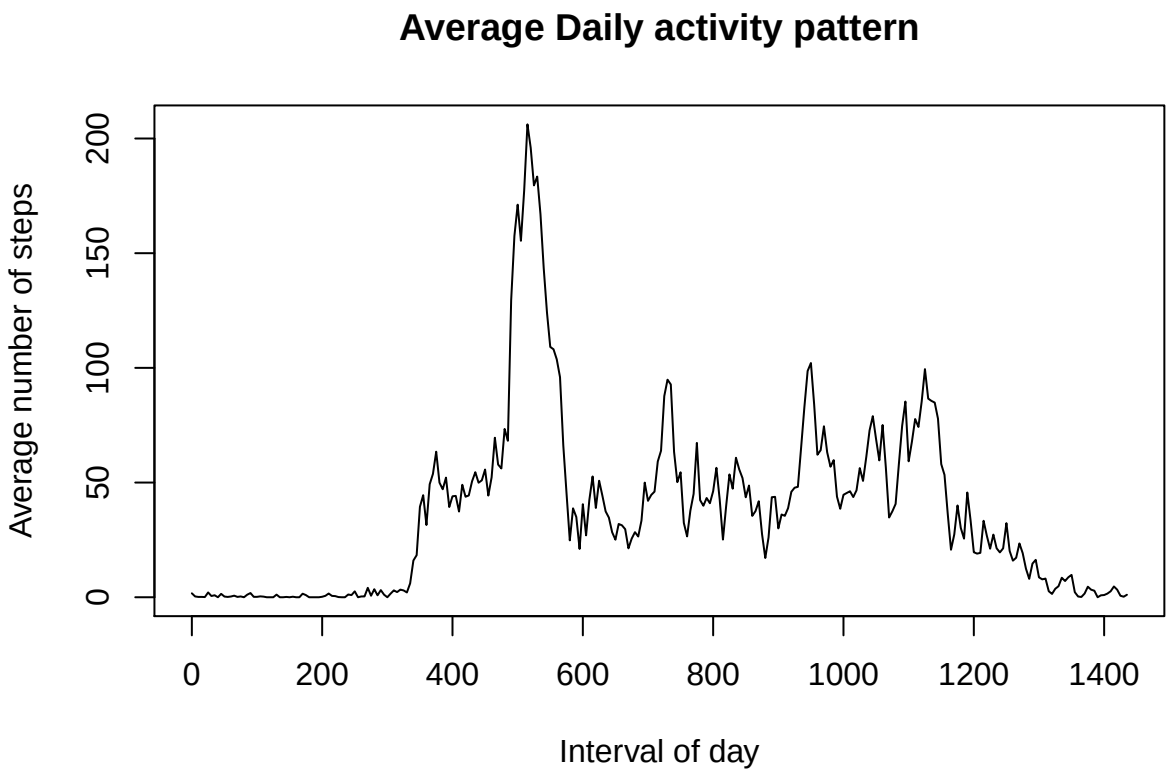
```
## [1] 288
```

No - we have an value in our vector for every 5-min interval. The intervals are:

```
intervals = as.numeric(names(ave.steps.per.5min))
```

We'll make a time-trend plot. The x-axis is the 5-minute interval of the day; the y-axis is the average number of daily steps in that interval.

```
plot(ave.steps.per.5min ~ intervals, type="l",
     ylab="Average number of steps", xlab="Interval of day", main=
       "Average Daily activity pattern")
```



At what time during the day does this person take the most steps per minute?

```
time.of.max = intervals[ave.steps.per.5min==max(ave.steps.per.5min)]
```

This is the time in minutes. The time in hours (army time, so 4 am = 4:00 and 4 pm =16:00) is:

```
paste(time.of.max %/% 60, ":",
time.of.max %% 60, sep="")
```

```
## [1] "8:35"
```

Imputing Missing Values

How many rows in the data set have missing values?

```
summary(activity)
```

```
##      steps            date                          interval
##  Min.   :  0.0   Min.   :2020-10-01 00:00:00   Min.   :   0
##  1st Qu.:  0.0   1st Qu.:2020-10-16 00:00:00   1st Qu.: 359
##  Median :  0.0   Median :2020-10-31 00:00:00   Median : 718
##  Mean   : 37.4   Mean   :2020-10-31 00:28:31   Mean   : 718
##  3rd Qu.: 12.0   3rd Qu.:2020-11-15 00:00:00   3rd Qu.:1076
##  Max.   :806.0   Max.   :2020-11-30 00:00:00   Max.   :1435
##  NA's   :2304
```

From above, only steps has missing values - no missing dates or intervals.

I impute missing number of steps for each interval to be the mean number of steps in that interval as follows:

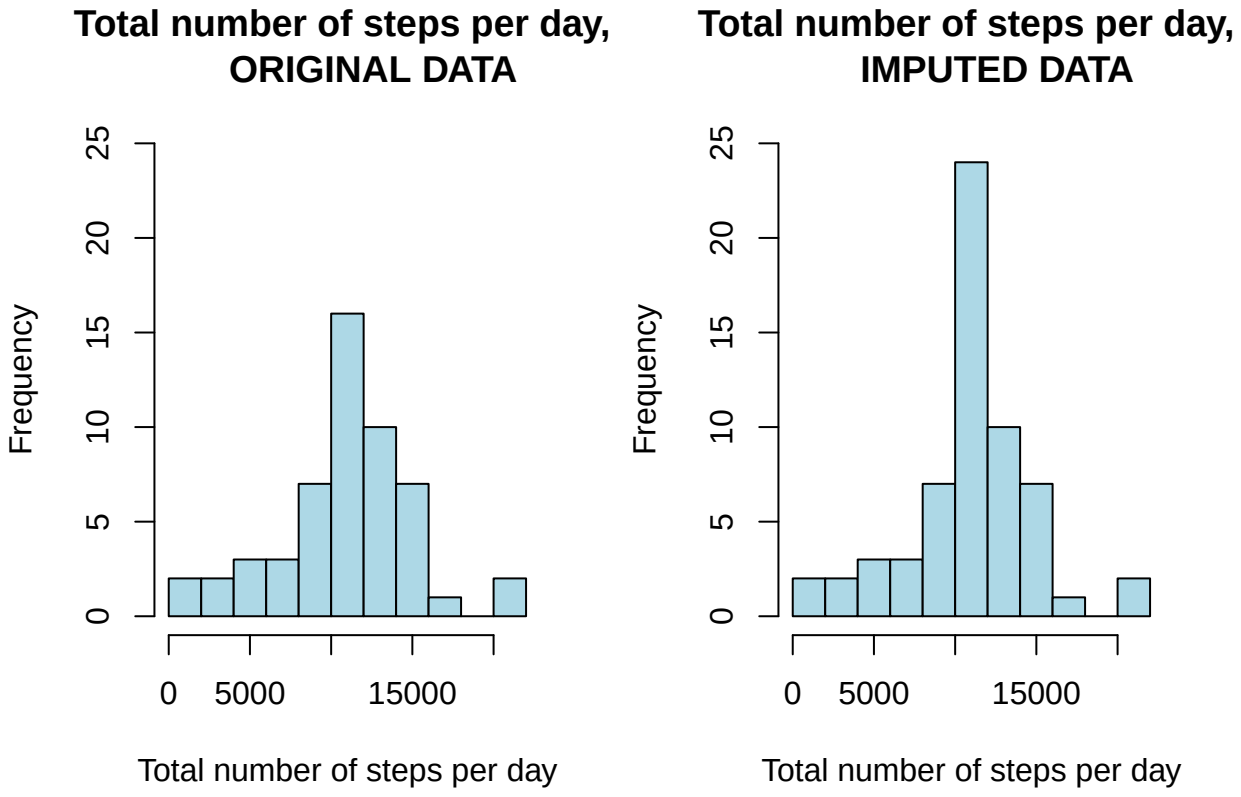
```
activity2=activity
missing.indices = which(is.na(activity$steps))
for (i in missing.indices){
   activity2$steps[i] =
     mean(activity$steps[activity$interval==activity$interval[i]], na.rm=T)
   }
```

I'll compare the distribution of total number of steps per day with this new data set to the original data set. Let's compute the new total number of steps per day:

```
total.steps2 = tapply(activity2$steps, activity2$date, sum)
```

We create side-by-histograms with the following code:

```
par(mfrow=c(1,2))
hist(total.steps, main="Total number of steps per day,
     ORIGINAL DATA", ylim=c(0,25),
     col="lightblue",
     breaks=12,
     xlab="Total number of steps per day")
hist(total.steps2, main="Total number of steps per day,
     IMPUTED DATA", ylim=c(0,25),
     col="lightblue",
     breaks=12,
     xlab="Total number of steps per day")
```



We compare the means and medians as follows:

```
mean(total.steps, na.rm=T) ## Original mean
```

```
## [1] 10766
```

```
mean(total.steps, na.rm=T) ## Imputed mean
```

```
## [1] 10766
```

```
median(total.steps, na.rm=T) ## Original median
```

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

```
median(total.steps2, na.rm=T) ## Imputed median
```

```
## [1] 10766
```

The means and medians have barely changed. The distribution has more values at or near the mode but is otherwise similar.

Are there differences in activity patterns between weekdays and weekends?

Creating the weekend/weekday factor variable:

```
day.of.week = weekdays(activity$date)
activity2$wkd.or.wkday =
factor(day.of.week=="Saturday" | day.of.week=="Sunday")
levels(activity2$wkd.or.wkday) = c("weekday", "weekend")
```

Computing average number of steps per interval forweekdays and weekends separately:

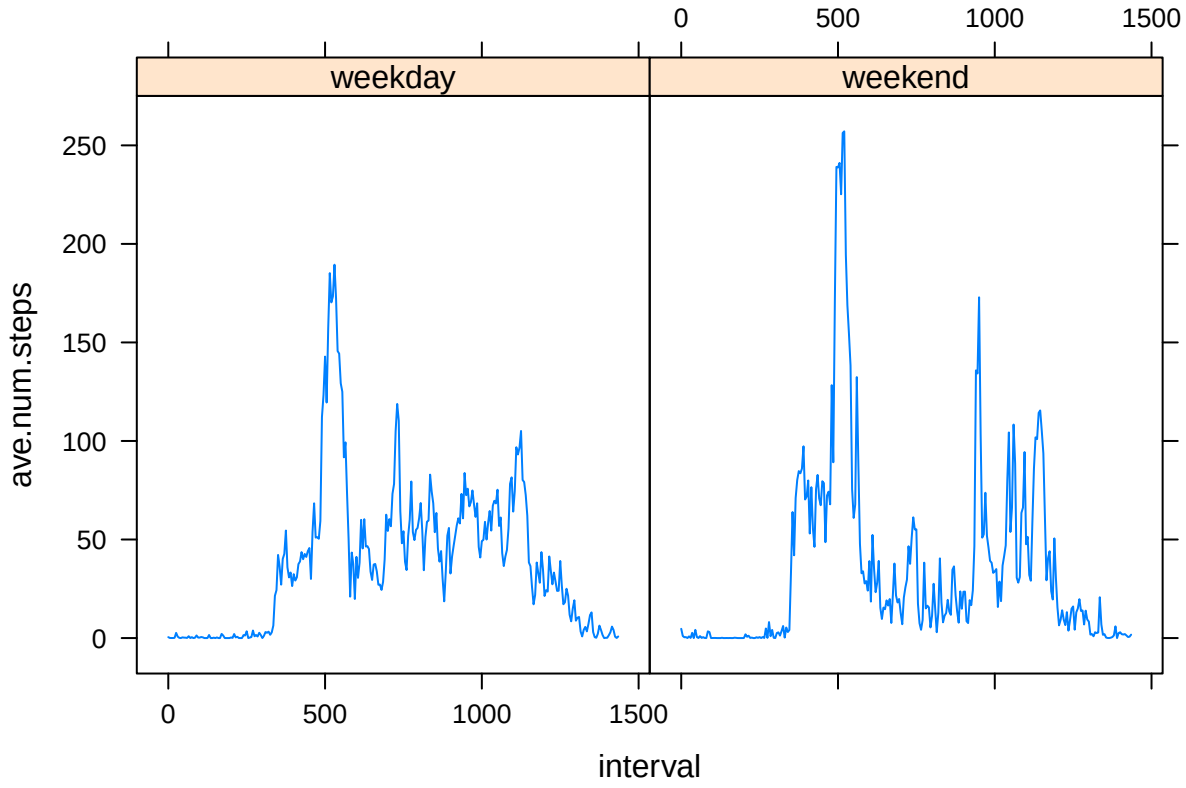
```
wkday.dat = activity2[activity2$wkd.or.wkday=="weekday",]
ave.num.steps.wkday = tapply(wkday.dat$steps, wkday.dat$interval, mean)

wkd.dat = activity2[activity2$wkd.or.wkday=="weekend",]
ave.num.steps.wkd = tapply(wkd.dat$steps, wkd.dat$interval, mean)
```

I combine these into a data frame that has one row for each interval and weekday/wkd combinations. Since there are 1440 intervals in a day, there are 2*1440=2880 rows in the data frame:

```
dat = data.frame(interval = rep(seq(0,1435, by=5),2),
  ave.num.steps = c(as.vector(ave.num.steps.wkday), as.vector(ave.num.steps.wkd)),
  wkd.or.wkday = as.factor(rep(c("weekday", "weekend"), c(288,288))))
```

Creating the plot with the lattice library:

```
library(lattice)
xyplot(ave.num.steps~ interval | wkd.or.wkday, type="l", data=dat)
```



Weekdays show more midday walking spread out over a period of several hours. On weekends, the morning spike is higher (more steps in the morning walk), mid-day numbers of steps decrease, and evening is more spikey and higher.