Reproducible Research: Peer Assessment 1

Rajesh Ekkaladevi

Dec 10, 2018

NOTE: Please review the figures in the figure directory! as inline figures are too large to be displayed in html file. Or please check the attached PDF in the project directory containing figures inline. Thanks!

Loading and processing the data:

Load reqd. libraries

```
library(tidyverse)
library(lubridate)
library(knitr)
opts_knit$set(echo=TRUE, figure.path="figure/")
```

Load activity data.

```
activity <- read_csv("activity.zip")
```

```
## Parsed with column specification:
## cols(
##   steps = col_integer(),
##   date = col_date(format = ""),
##   interval = col_integer()
## )
```

```
glimpse(activity)
```

```
## Observations: 17,568
## Variables: 3
## $ steps    <int> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N...
## $ date     <date> 2012-10-01, 2012-10-01, 2012-10-01, 2012-10-01, 2012...
## $ interval <int> 0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 100, 10...
```

Add date_hms variable with date and time in datetime format.

```
activity <- activity %>% mutate(date_hms = ymd_hm(sprintf("%s %04d", as.character(date), interval)))
```

```
glimpse(activity)
```

```
## Observations: 17,568
## Variables: 4
## $ steps    <int> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N...
## $ date     <date> 2012-10-01, 2012-10-01, 2012-10-01, 2012-10-01, 2012...
```

```
## $ interval <int> 0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 100, 10...
## $ date_hms <dttm> 2012-10-01 00:00:00, 2012-10-01 00:05:00, 2012-10-01...
```

Histogram of the total number of steps taken each day:

Steps_per_day, mean and median number of steps taken per day after filtering NAs.

```
activity_day <- activity %>% select(date, steps) %>% filter(!is.na(steps)) %>% group_by(date) %>% summa
```

```
glimpse(activity_day)
```

```
## Observations: 53
## Variables: 2
## $ date          <date> 2012-10-02, 2012-10-03, 2012-10-04, 2012-10-05,...
## $ steps_per_day <int> 126, 11352, 12116, 13294, 15420, 11015, 12811, 9...
```

Mean and median number of steps taken each day:

Mean and median steps per day

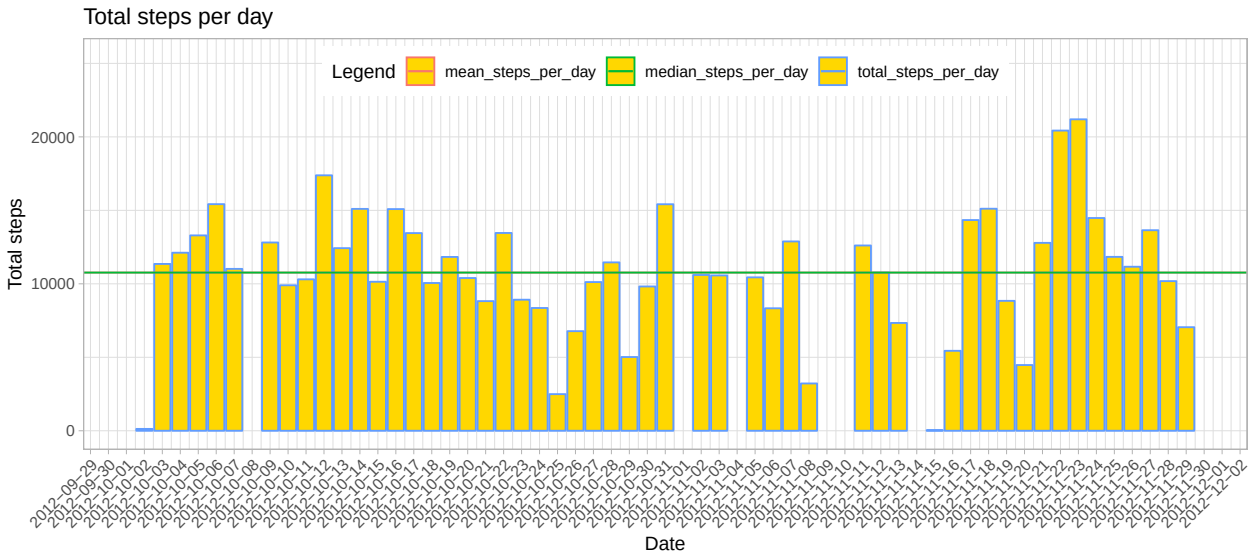
```
activity_day_mean_median <- activity_day %>% summarize(total_steps=sum(steps_per_day), mean_steps_per_da
```

```
activity_day_mean_median
```

```
## # A tibble: 1 x 3
##   total_steps mean_steps_per_day median_steps_per_day
##         <int>              <dbl>                <int>
## 1      570608             10766.                10765
```

Plot showing histogram of steps_per_day across all days.

```
ggplot(activity_day, aes(date, steps_per_day)) +
geom_bar(stat="identity", aes(color="total_steps_per_day"), fill="gold", size=.5, show.legend=TRUE) +
geom_hline(aes(yintercept=activity_day_mean_median$mean_steps_per_day, color="mean_steps_per_day"), siz
geom_hline( aes(yintercept=activity_day_mean_median$median_steps_per_day, color="median_steps_per_day")
labs(x="Date", y="Total steps", title="Total steps per day", color="Legend") +
scale_x_date(date_breaks="1 day", date_labels="%Y-%m-%d") +
theme_light() +
theme(axis.text.x=element_text(angle=45, vjust=1, hjust=1), legend.direction="horizontal", legend.positi
ylim(0,max(activity_day$steps_per_day)*1.2)
```



Time series plot of the average number of steps taken by interval:

Steps_per_interval, mean and median number of steps taken per interval after filtering NAs.

```
activity_interval <- activity %>% filter(!is.na(steps)) %>% group_by(interval) %>% summarize(steps_per_
```

```
glimpse(activity_interval)
```

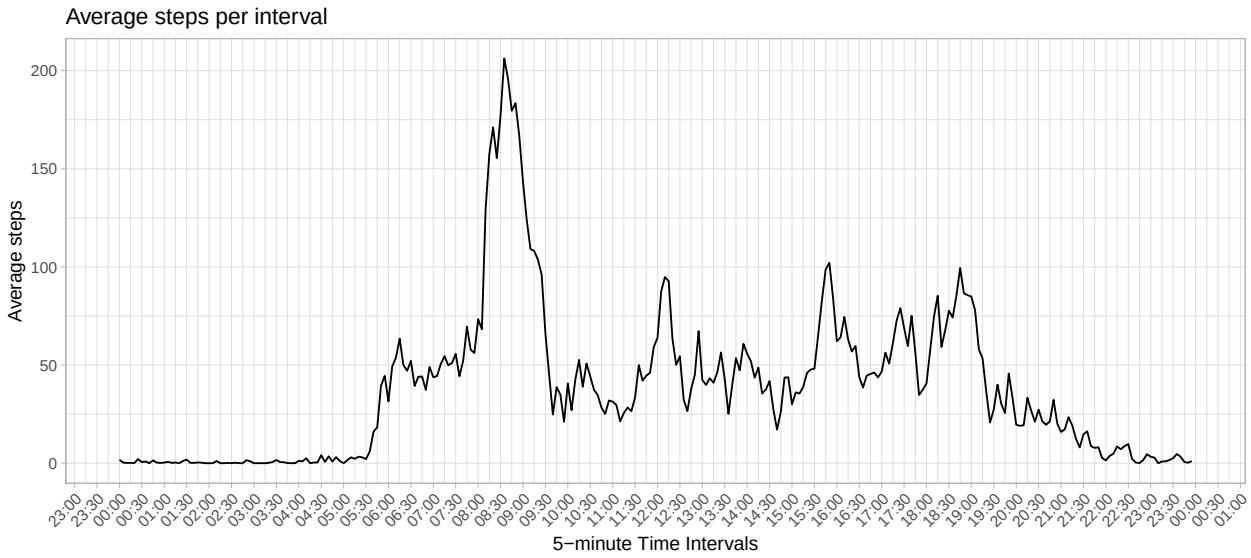
```
## Observations: 288
## Variables: 3
## $ interval               <int> 0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 5...
## $ steps_per_interval     <int> 91, 18, 7, 8, 4, 111, 28, 46, 0, 78, 16...
## $ avg_steps_per_interval <dbl> 1.72, 0.34, 0.13, 0.15, 0.08, 2.09, 0.5...
```

Add interval_time variable from interval in datetime format for plotting.

```
activity_interval_plot <- activity_interval %>% mutate(interval_time = as.POSIXct(strptime(sprintf("%04
```

Plot of timeseries of averaging steps per day across all days.

```
ggplot(activity_interval_plot, aes(interval_time, avg_steps_per_interval)) +
geom_line() +
labs(x="5-minute Time Intervals", y="Average steps", title="Average steps per interval", color="Legend"
scale_x_datetime(date_breaks="30 mins", date_labels="%H:%M") +
theme_light() +
theme(axis.text.x=element_text(angle=45, vjust=1, hjust=1), legend.direction="horizontal", legend.positi
```



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

```
activity_interval %>% select(interval, avg_steps_per_interval) %>% filter(avg_steps_per_interval == max
```

```
## # A tibble: 1 x 2
##   interval avg_steps_per_interval
##      <int>                  <dbl>
## 1      835                   206.
```

From the above max average steps per interval is noticed on interval 835.

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

Find the number of NAs and its proportion in the data.

```
activity %>% select(steps) %>% mutate(NAs = ifelse(is.na(steps), 'yes', 'no')) %>% group_by(NAs) %>% su
```

```
## # A tibble: 2 x 3
##   NAs   count percent
##   <chr> <int>   <dbl>
## 1 no    15264   0.869
## 2 yes    2304   0.131
```

Find the date's on which NAs are there.

```
activity %>% filter(is.na(steps)) %>% group_by(date) %>% summarize(n = n()) %>% mutate(cumulative_NAs =
```

```
## # A tibble: 8 x 3
##   date           n cumulative_NAs
##   <date>     <int>          <int>
## 1 2012-10-01   288            288
## 2 2012-10-08   288            576
## 3 2012-11-01   288            864
```

```
## 4 2012-11-04   288           1152
## 5 2012-11-09   288           1440
## 6 2012-11-10   288           1728
## 7 2012-11-14   288           2016
## 8 2012-11-30   288           2304
```

The number of NAs from the above two snippets match to **2304** indicating on the days where the data is missing it is missing for the full day.

Join original data set activity with activity_interval to get avg_steps_per_interval in the dataset.

```
activity_joined <- left_join(activity, activity_interval)
```

```
## Joining, by = "interval"
```

```
head(activity_joined)
```

```
## # A tibble: 6 x 6
##   steps date       interval date_hms            steps_per_inter~
##   <int> <date>        <int> <dttm>                         <int>
## 1    NA 2012-10-01        0 2012-10-01 00:00:00              91
## 2    NA 2012-10-01        5 2012-10-01 00:05:00              18
## 3    NA 2012-10-01       10 2012-10-01 00:10:00               7
## 4    NA 2012-10-01       15 2012-10-01 00:15:00               8
## 5    NA 2012-10-01       20 2012-10-01 00:20:00               4
## 6    NA 2012-10-01       25 2012-10-01 00:25:00             111
## # ... with 1 more variable: avg_steps_per_interval <dbl>
```

Impute all intervals where steps is NA.

```
activity_imputed <- activity_joined %>% mutate(steps = ifelse(is.na(steps), avg_steps_per_interval, step
head(activity_imputed)
```

```
## # A tibble: 6 x 6
##   steps date       interval date_hms            steps_per_inter~
##   <dbl> <date>        <int> <dttm>                         <int>
## 1  1.72 2012-10-01        0 2012-10-01 00:00:00              91
## 2  0.34 2012-10-01        5 2012-10-01 00:05:00              18
## 3  0.13 2012-10-01       10 2012-10-01 00:10:00               7
## 4  0.15 2012-10-01       15 2012-10-01 00:15:00               8
## 5  0.08 2012-10-01       20 2012-10-01 00:20:00               4
## 6  2.09 2012-10-01       25 2012-10-01 00:25:00             111
## # ... with 1 more variable: avg_steps_per_interval <dbl>
```

Check if all dates with step values as NA are updated.

```
activity_imputed %>% filter(is.na(steps)) %>% group_by(date) %>% summarize(n = n()) %>% mutate(cumulati
```

```
## # A tibble: 0 x 3
## # ... with 3 variables: date <date>, n <int>, cumulative_NAs <int>
```

From the above all NAs are update hence no more NAs.

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

Compute avg_steps_per_day on the imputed data.

```
activity_imputed_day <- activity_imputed %>% select(date, steps) %>% group_by(date) %>% summarize(steps_
```

Mean and median number of steps taken each day after data impute.

```
activity_imputed_day_mean_median <- activity_imputed_day %>% summarize(total_steps=sum(steps_per_day), m
```

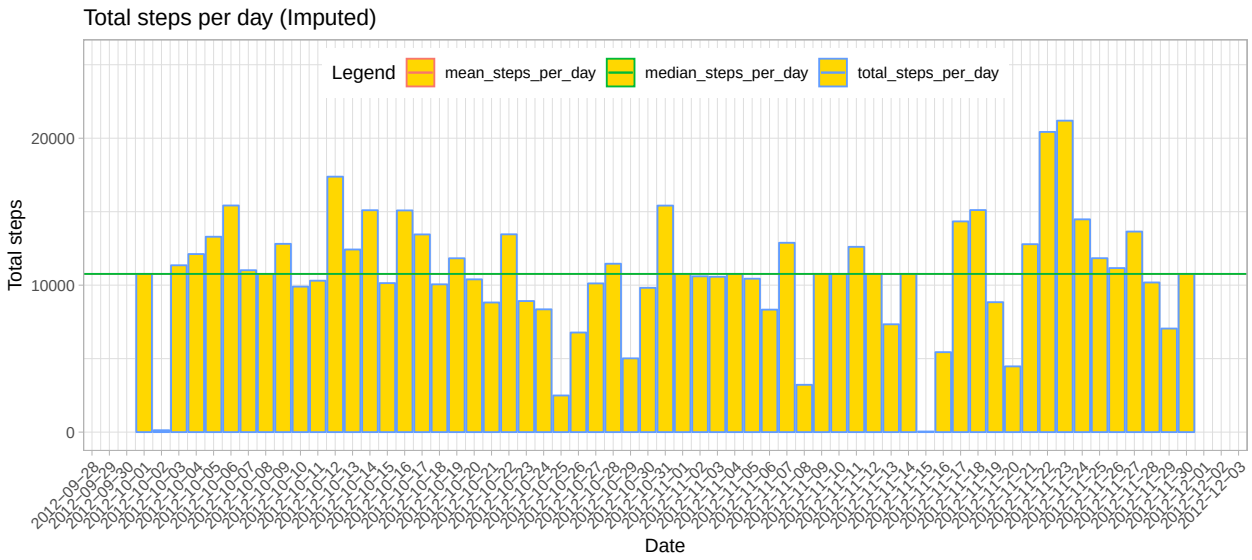
```
activity_imputed_day_mean_median
```

```
## # A tibble: 1 x 3
##   total_steps mean_steps_per_day median_steps_per_day
##         <dbl>              <dbl>                <dbl>
## 1     656737.             10766.               10766.
```

The inputed mean and medians have changed from activity_day_mean_median (before impute).

Plot showing histogram of steps_per_day across all days after impute.

```
ggplot(activity_imputed_day, aes(date, steps_per_day)) +
geom_bar(stat="identity", aes(color="total_steps_per_day"), fill="gold", size=.5, show.legend=TRUE) +
geom_hline(aes(yintercept=activity_imputed_day_mean_median$mean_steps_per_day, color="mean_steps_per_day
geom_hline( aes(yintercept=activity_imputed_day_mean_median$median_steps_per_day, color="median_steps_pe
labs(x="Date", y="Total steps", title="Total steps per day (Imputed)", color="Legend") +
scale_x_date(date_breaks="1 day", date_labels="%Y-%m-%d") +
theme_light() +
theme(axis.text.x=element_text(angle=45, vjust=1, hjust=1), legend.direction="horizontal", legend.positi
ylim(0,max(activity_day$steps_per_day)*1.2)
```



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

Add weekday to the data using lubridate function.

```
activity_imputed_week <- activity_imputed %>% mutate(weekday = as.character(wday(date, label=TRUE)), is
str(activity_imputed_week)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    17568 obs. of  8 variables:
##  $ steps                 : num  1.72 0.34 0.13 0.15 0.08 2.09 0.53 0.87 0 1.47 ...
##  $ date                  : Date, format: "2012-10-01" "2012-10-01" ...
##  $ interval              : int  0 5 10 15 20 25 30 35 40 45 ...
##  $ date_hms              : POSIXct, format: "2012-10-01 00:00:00" "2012-10-01 00:05:00" ...
##  $ steps_per_interval    : int  91 18 7 8 4 111 28 46 0 78 ...
##  $ avg_steps_per_interval: num  1.72 0.34 0.13 0.15 0.08 2.09 0.53 0.87 0 1.47 ...
##  $ weekday               : chr  "Mon" "Mon" "Mon" "Mon" ...
##  $ is_weekday            : chr  "Weekday" "Weekday" "Weekday" "Weekday" ...
```

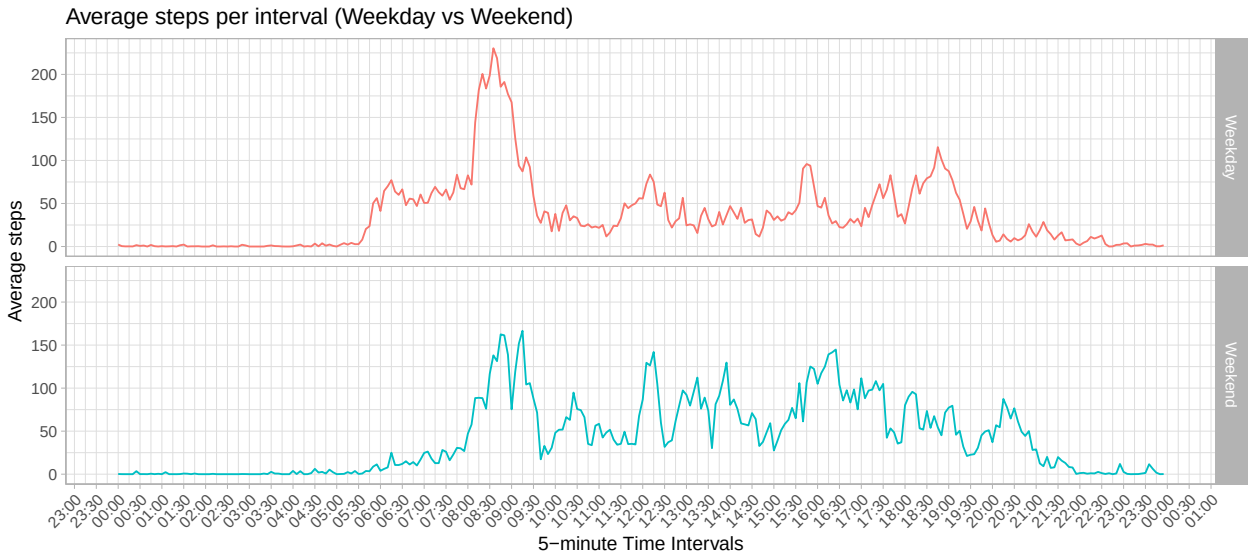
```
activity_imputed_week_interval <- activity_imputed_week %>% group_by(interval, is_weekday) %>% mutate(av
head(activity_imputed_week_interval)
```

```
## # A tibble: 6 x 8
## # Groups:   interval, is_weekday [6]
##   steps date       interval date_hms            steps_per_inter~
##   <dbl> <date>        <int> <dttm>                         <int>
## 1  1.72 2012-10-01        0 2012-10-01 00:00:00               91
## 2  0.34 2012-10-01        5 2012-10-01 00:05:00               18
## 3  0.13 2012-10-01       10 2012-10-01 00:10:00                7
## 4  0.15 2012-10-01       15 2012-10-01 00:15:00                8
## 5  0.08 2012-10-01       20 2012-10-01 00:20:00                4
## 6  2.09 2012-10-01       25 2012-10-01 00:25:00              111
## # ... with 3 more variables: avg_steps_per_interval <dbl>, weekday <chr>,
## #   is_weekday <chr>
```

Add interval_time variable from interval in datetime format for plotting.

```
activity_imputed_week_interval_plot <- activity_imputed_week_interval %>% mutate(interval_time = as.POSI
```

```
ggplot(activity_imputed_week_interval_plot, aes(interval_time, avg_steps_per_interval, group=is_weekday
geom_line() +
labs(x="5-minute Time Intervals", y="Average steps", title="Average steps per interval (Weekday vs Week
scale_x_datetime(date_breaks="30 mins", date_labels="%H:%M") +
theme_light() +
theme(axis.text.x=element_text(angle=45, vjust=1, hjust=1), legend.position="none") +
facet_grid(is_weekday ~ .)
```



From the above the max number of steps are noticed on weekday.