Exploratory Data Analysis on los_angeles_video.csv

USC Machine Learning Team

11/2/2017

Data Loading

```
# load data file
if (!exists("LAV.raw")) {
    LAV.raw <- read.csv("./los_angeles_video.csv")
}
print(nrow(LAV.raw))
```

```
## [1] 4017
```

Data Cleaning & Exploratory Analysis

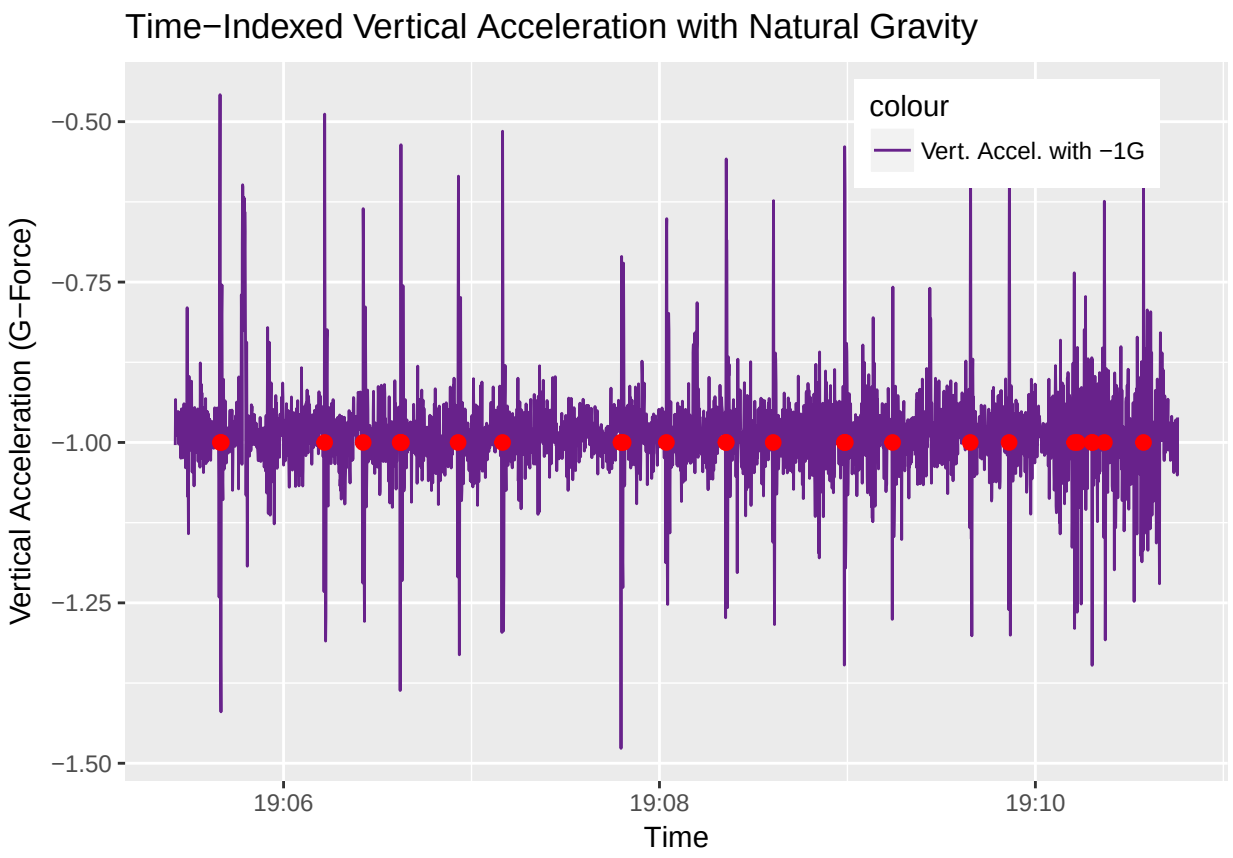
Data Cleaning & Exploratory Analysis are iterative, starting at next page.

Epoch 1: Display of Vertical Acceleration with Natural Gravity

Date: the time stamp of a data point
Speed: traveling speed of the vehicle
forw_accel: forward acceleration (front and back) of the vehicle
hori_accel: horizontal acceleration (left and right) of the vehicle
vert_accel_G: vertical acceleration (up and down) of the vehicle with natural gravity (-1G)
speedbump: whether this data point is a speedbump

```
LAV.valid = LAV.raw[, c("Date", "Latitude", "Longitude", "Speed")]
# validate DateTime format
LAV.valid$Date <- as.POSIXct(LAV.valid$Date, format="%Y-%m-%d %H:%M:%OS")
# specify orientation of accelration
LAV.valid$forw_accel = LAV.raw$X
LAV.valid$hori_accel = LAV.raw$Y
LAV.valid$vert_accel_G = LAV.raw$Z
# mark speed bumps
for (i in 1:nrow(LAV.valid)) {
    LAV.valid$speedbump[i] = "no"
}
for (i in c(28, 29, 444, 447, 777, 900, 1018, 1019, 1020, 1022, 1203, 1345, 1723, 1724, 1730, 1868, 205
    LAV.valid$speedbump[i] = "yes"
}
LAV.valid <- LAV.valid[300:3500, ]
```

```
## Loading required package: ggplot2
```

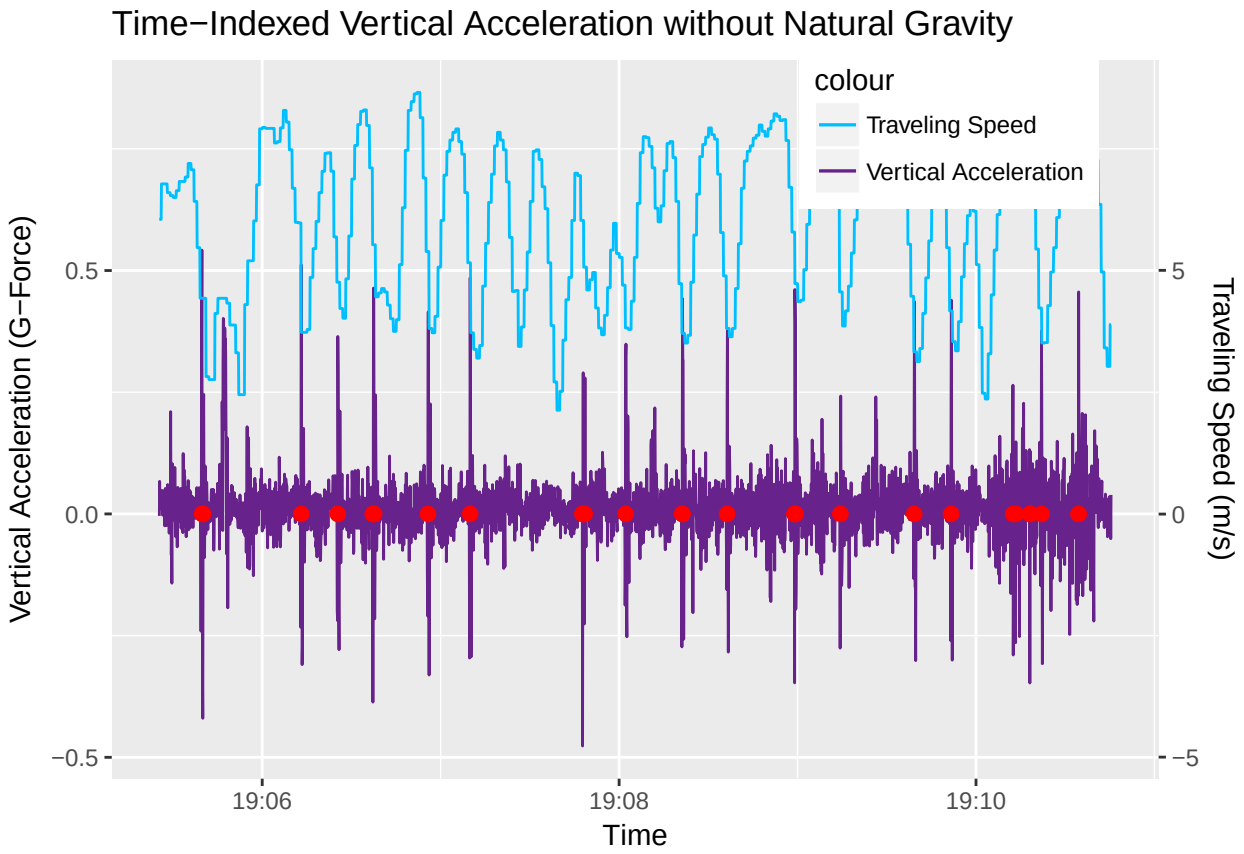


Note: Speed bumps are labeled as *red* points on the graph.

Epoch 2: Display of Vertical Acceleration without Natural Gravity

vert_accel: vertical acceleration (up and down) of the vehicle without natural gravity

```
# remove natural gravity in vertical acceleration
LAV.valid$vert_accel = LAV.valid$vert_accel_G + 1
```

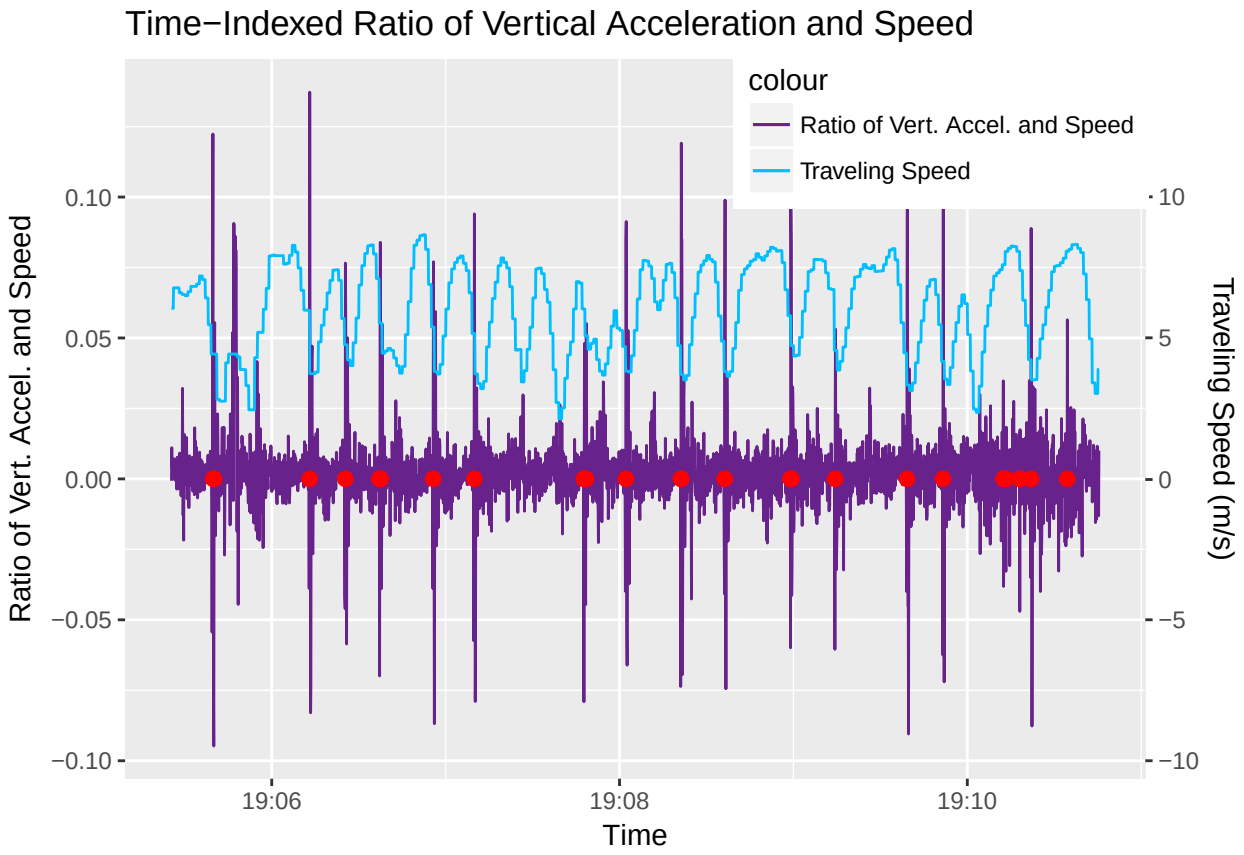


Note: Speed bumps are labeled as *red* points on the graph.

Epoch 3: Display of Ratio between Vertical Acceleration (w/o Natural G) and Speed

vert_accel_ratio_speed: the ratio between vertical acceleration (without natural gravity) and traveling speed

```
# calculate ratio between vertical acceleration and speed
for (i in 1:nrow(LAV.valid)) {
    if (LAV.valid$Speed[i] == 0) {
        LAV.valid$vert_accel_ratio_speed[i] = 0
    }
    else {
        LAV.valid$vert_accel_ratio_speed[i] = LAV.valid$vert_accel[i] / LAV.valid$Speed[i]
    }
}
```

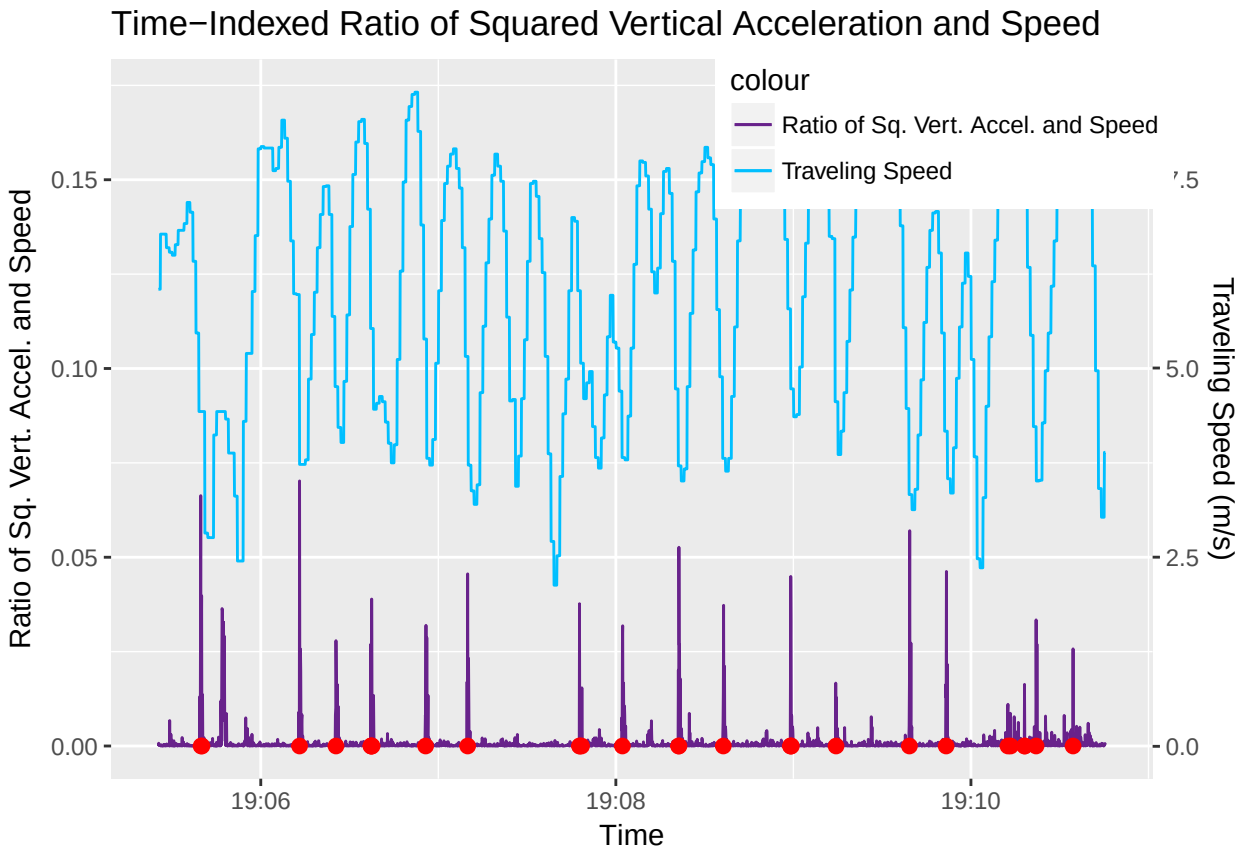


Note: Speed bumps are labeled as *red* points on the graph.

Epoch 4: Display of Ratio between Vertical Acceleration (w/o Natural G) and Speed

sq_vert_accel_ratio_speed: the ratio between squared vertical acceleration (without natual gravity) and traveling speed

```
# calculate ratio between vertical acceleration and speed
for (i in 1:nrow(LAV.valid)) {
    if (LAV.valid$Speed[i] == 0) {
        LAV.valid$sq_vert_accel_ratio_speed[i] = 0
    }
    else {
        LAV.valid$sq_vert_accel_ratio_speed[i] = (LAV.valid$vert_accel[i] * LAV.valid$vert_accel[i]) / I
    }
}
```



Note: Speed bumps are labeled as *red* points on the graph.

Epoch 5: Display of Vertical Jolt (Incremental Change of Vert. Accel.)

vert_jolt: vertical jolt of the vehicle (incremental change of vertical acceleration)

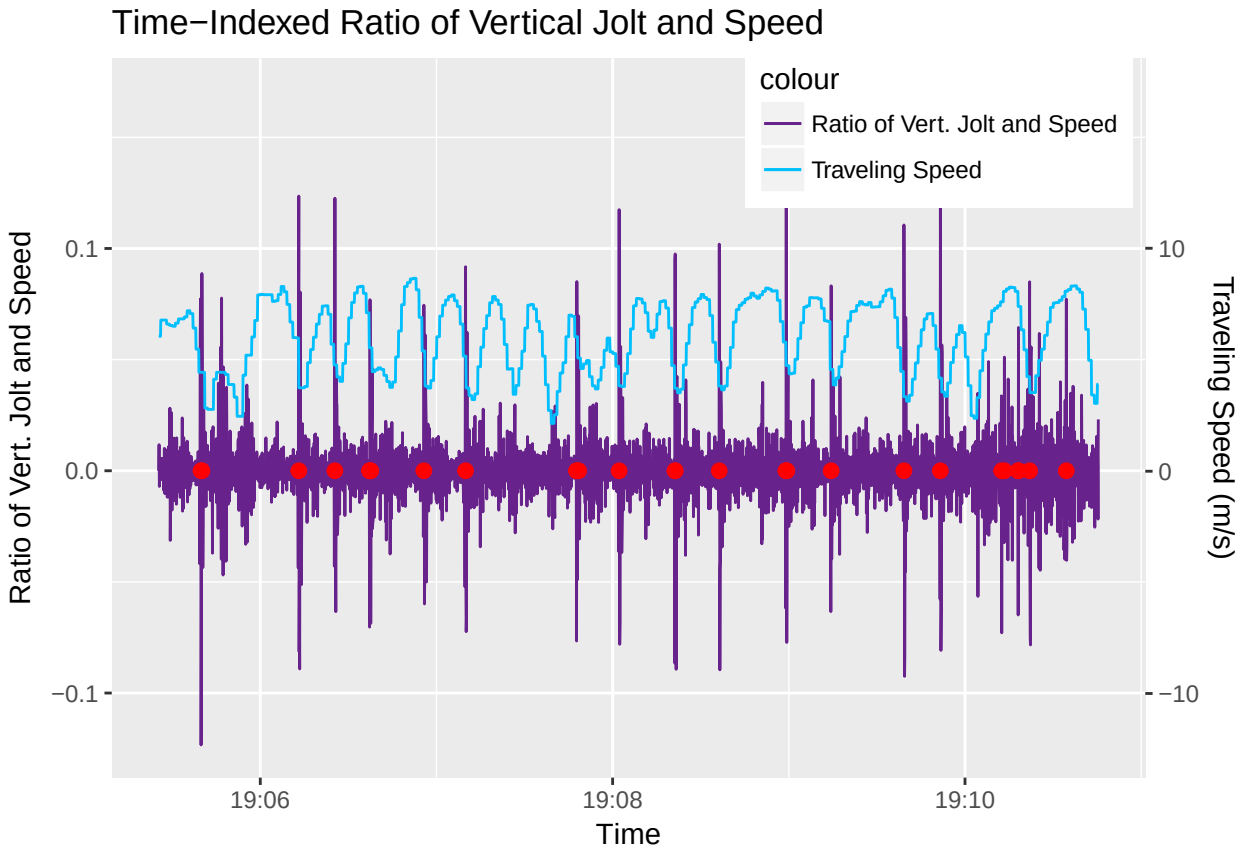
```
# calculate vertical jolt
for (i in 1:nrow(LAV.valid)) {
    if (i == 1) {
        LAV.valid$vert_jolt[i] = 0
    }
    else {
        LAV.valid$vert_jolt[i] = LAV.valid$vert_accel[i] - LAV.valid$vert_accel[i - 1]
    }
}
```

Note: Speed bumps are labeled as *red* points on the graph.

Epoch 6: Display of Ratio between Vertical Jolt and Speed

vert_jolt_ratio_speed: the ratio between vertical jolt and traveling speed

```
# calculate ratio between vertical jolt and speed
for (i in 1:nrow(LAV.valid)) {
    if (LAV.valid$Speed[i] == 0) {
        LAV.valid$vert_jolt_ratio_speed[i] = 0
    }
    else {
        LAV.valid$vert_jolt_ratio_speed[i] = LAV.valid$vert_jolt[i] / LAV.valid$Speed[i]
    }
}
```

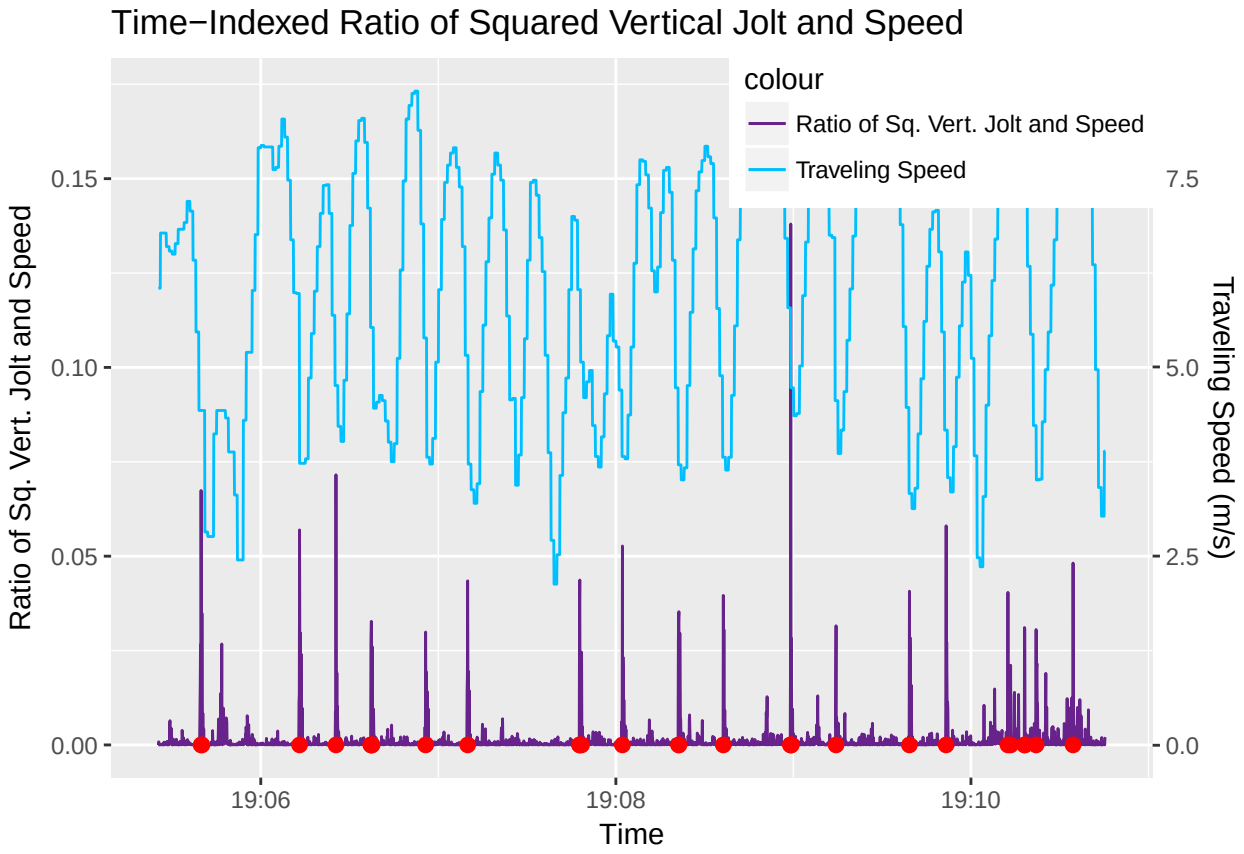


Note: Speed bumps are labeled as *red* points on the graph.

Epoch 7: Display of Ratio between Squared Vertical Jolt and Speed

sq_vert_jolt_ratio_speed: the ratio between squared vertical jolt and traveling speed

```
# calculate ratio between vertical jolt and speed
for (i in 1:nrow(LAV.valid)) {
    if (LAV.valid$Speed[i] == 0) {
        LAV.valid$sq_vert_jolt_ratio_speed[i] = 0
    }
    else {
        LAV.valid$sq_vert_jolt_ratio_speed[i] = (LAV.valid$vert_jolt[i] * LAV.valid$vert_jolt[i]) / LAV
    }
}
```



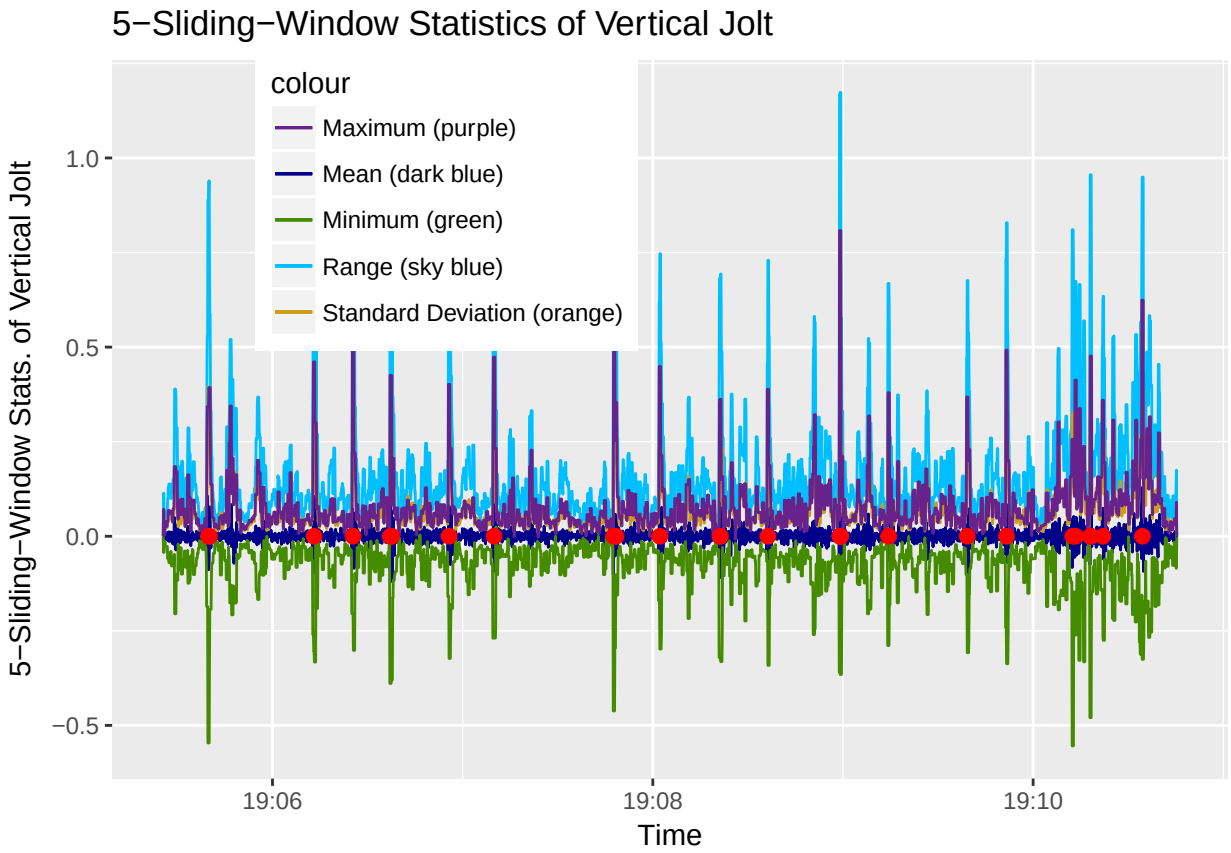
Note: Speed bumps are labeled as *red* points on the graph.

Epoch 8: Display of Sliding-Window Statistics of Vertical Jolt

vert_jolt_mean: 5-sliding-window mean of vertical jolt
vert_jolt_sd: 5-sliding-window standard deviation of vertical jolt
vert_jolt_min: 5-sliding-window minimum of vertical jolt
vert_jolt_max: 5-sliding-window maximum of vertical jolt
vert_jolt_range: 5-sliding-window range of vertical jolt

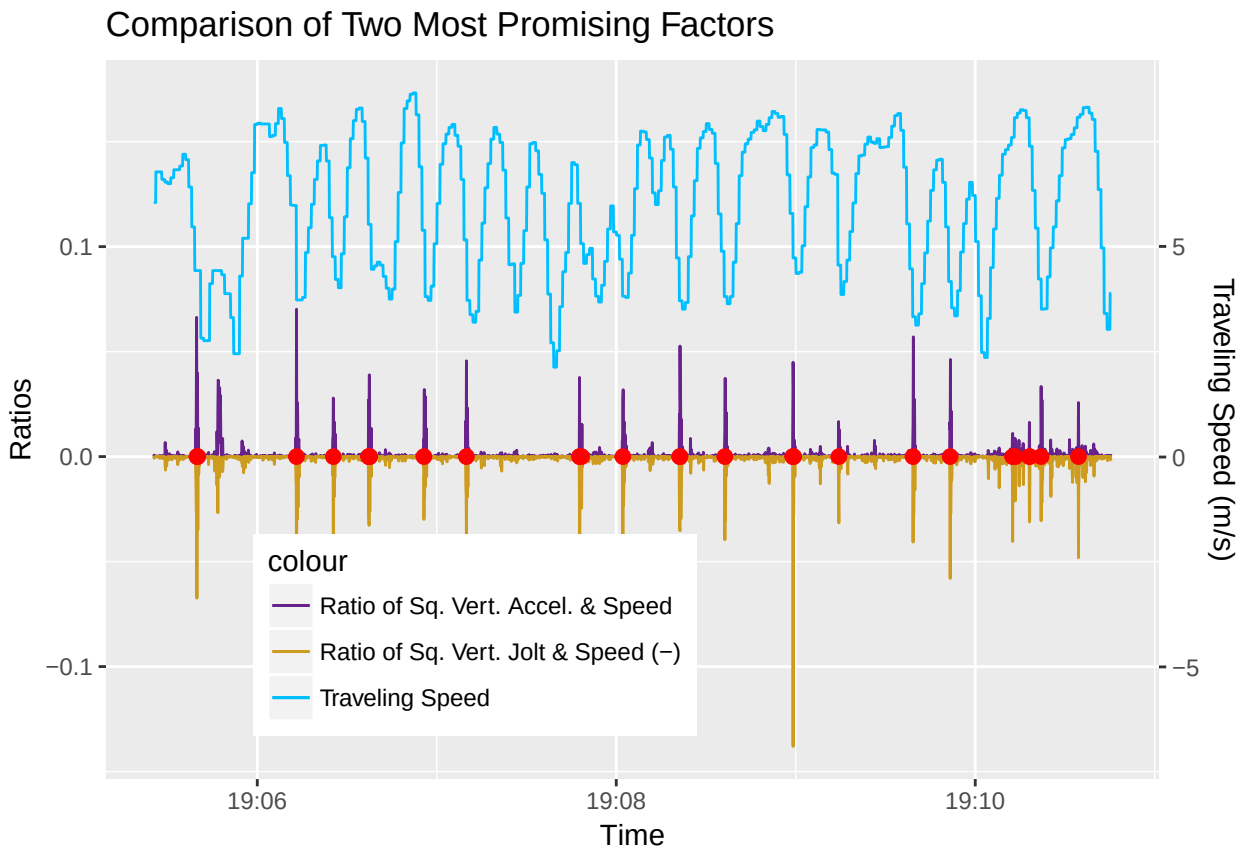
```
# calculate 5-sliding-window mean of vertical jolt
for (i in 3:nrow(LAV.valid)-2) {
    LAV.valid$vert_jolt_mean[i] = mean(c(LAV.valid$vert_jolt[i-2],
                                         LAV.valid$vert_jolt[i-1],
                                         LAV.valid$vert_jolt[i],
                                         LAV.valid$vert_jolt[i+1],
                                         LAV.valid$vert_jolt[i+2]))
}
LAV.valid$vert_jolt_mean[1] = 0
LAV.valid$vert_jolt_mean[2] = 0
LAV.valid$vert_jolt_mean[nrow(LAV.valid) - 1] = 0
LAV.valid$vert_jolt_mean[nrow(LAV.valid)] = 0
# calculate 5-sliding-window standard deviation of vertical jolt
for (i in 3:nrow(LAV.valid)-2) {
    LAV.valid$vert_jolt_sd[i] = sd(c(LAV.valid$vert_jolt[i-2],
                                     LAV.valid$vert_jolt[i-1],
                                     LAV.valid$vert_jolt[i],
                                     LAV.valid$vert_jolt[i+1],
                                     LAV.valid$vert_jolt[i+2]))
}
LAV.valid$vert_jolt_sd[1] = 0
LAV.valid$vert_jolt_sd[2] = 0
LAV.valid$vert_jolt_sd[nrow(LAV.valid) - 1] = 0
LAV.valid$vert_jolt_sd[nrow(LAV.valid)] = 0
# calculate 5-sliding-window minimum of vertical jolt
for (i in 3:nrow(LAV.valid)-2) {
    LAV.valid$vert_jolt_min[i] = min(c(LAV.valid$vert_jolt[i-2],
                                       LAV.valid$vert_jolt[i-1],
                                       LAV.valid$vert_jolt[i],
                                       LAV.valid$vert_jolt[i+1],
                                       LAV.valid$vert_jolt[i+2]))
}
LAV.valid$vert_jolt_min[1] = 0
LAV.valid$vert_jolt_min[2] = 0
LAV.valid$vert_jolt_min[nrow(LAV.valid) - 1] = 0
LAV.valid$vert_jolt_min[nrow(LAV.valid)] = 0
# calculate 5-sliding-window maxium of vertical jolt
for (i in 3:nrow(LAV.valid)-2) {
    LAV.valid$vert_jolt_max[i] = max(c(LAV.valid$vert_jolt[i-2],
                                       LAV.valid$vert_jolt[i-1],
                                       LAV.valid$vert_jolt[i],
                                       LAV.valid$vert_jolt[i+1],
                                       LAV.valid$vert_jolt[i+2]))
}
LAV.valid$vert_jolt_max[1] = 0
```

```
LAV.valid$vert_jolt_max[2] = 0
LAV.valid$vert_jolt_max[nrow(LAV.valid) - 1] = 0
LAV.valid$vert_jolt_max[nrow(LAV.valid)] = 0
# calculate 5-sliding-window range of vertical jolt
LAV.valid$vert_jolt_range = LAV.valid$vert_jolt_max - LAV.valid$vert_jolt_min
```



Note: Speed bumps are labeled as *red* points on the graph.

Epoch 9: Comparative Display of Two Most Promising Factors



Note: Ratio of Sq. Vert. Jolt & Speed is negatively filpped for visualization purpose.
Note: Speed bumps are labeled as *red* points on the graph.

Data Writing

Date: the time stamp of a data point
Latitude:
Longitude:
Speed: traveling speed of the vehicle
speedbump: whether this data point is a speedbump
forw_accel: forward acceleration (front and back) of the vehicle
hori_accel: horizontal acceleration (left and right) of the vehicle
vert_accel_G: vertical acceleration (up and down) of the vehicle with natural gravity (-1G)
vert_accel: vertical acceleration (up and down) of the vehicle without natural gravity
vert_accel_ratio_speed: the ratio between vertical acceleration (without natural gravity) and traveling speed
sq_vert_accel_ratio_speed: the ratio between squared vertical acceleration (without natual gravity) and traveling speed
vert_jolt: vertical jolt of the vehicle (incremental change of vertical acceleration)
vert_jolt_ratio_speed: the ratio between vertical jolt and traveling speed
sq_vert_jolt_ratio_speed: the ratio between squared vertical jolt and traveling speed
vert_jolt_mean: 5-sliding-window mean of vertical jolt
vert_jolt_sd: 5-sliding-window standard deviation of vertical jolt
vert_jolt_min: 5-sliding-window minimum of vertical jolt
vert_jolt_max: 5-sliding-window maximum of vertical jolt
vert_jolt_range: 5-sliding-window range of vertical jolt

```
jpeg("./Epoch1_LAV.vert_accel_G.plot.jpeg", width = 1200, height = 750, units = "px")
print(LAV.vert_accel_G.plot)
dev.off()
```

```
## pdf
##   2
```

```
jpeg("./Epoch2_LAV.vert_accel.plot.jpeg", width = 1200, height = 750, units = "px")
print(LAV.vert_accel.plot)
dev.off()
```

```
## pdf
##   2
```

```
jpeg("./Epoch3_LAV.vert_accel_ratio_speed.plot.jpeg", width = 1200, height = 750, units = "px")
print(LAV.vert_accel_ratio_speed.plot)
dev.off()
```

```
## pdf
##   2
```

```
jpeg("./Epoch4_LAV.sq_vert_accel_ratio_speed.plot.jpeg", width = 1200, height = 750, units = "px")
print(LAV.sq_vert_accel_ratio_speed.plot)
dev.off()
```

```
## pdf
##   2
```

```
jpeg("./Epoch5_LAV.vert_jolt.plot.jpeg", width = 1200, height = 750, units = "px")
print(LAV.vert_jolt.plot)
dev.off()
```

```
## pdf
```

```
##   2
```

```
jpeg("./Epoch6_LAV.vert_jolt_ratio_speed.plot.jpeg", width = 1200, height = 750, units = "px")
print(LAV.vert_jolt_ratio_speed.plot)
dev.off()
```

```
## pdf
##   2
```

```
jpeg("./Epoch7_LAV.sq_vert_jolt_ratio_speed.plot.jpeg", width = 1200, height = 750, units = "px")
print(LAV.sq_vert_jolt_ratio_speed.plot)
dev.off()
```

```
## pdf
##   2
```

```
jpeg("./Epoch8_LAV.vert_jolt_5.plot.jpeg", width = 1200, height = 750, units = "px")
print(LAV.vert_jolt_5.plot)
dev.off()
```

```
## pdf
##   2
```

```
jpeg("./Epoch9_LAV.comparative.plot.jpeg", width = 1200, height = 750, units = "px")
print(LAV.comparative.plot)
dev.off()
```

```
## pdf
##   2
```

```
write.csv(LAV.valid, "./los_angeles_video_labeled.csv")
```