Data Analysis Report: Root Insurance Co

Issac Lee[a]

[a]Department of Statistics & Actuarial Science, 241 Schaeffer Hall, Iowa City, Iowa 52242-1409

This version was compiled on November 1, 2019

This report illustrates how I approach to the solution and make a decision for Root insurance's data analysis project. The goal of this project is to find the best-matched OBDII trip with a given trip data which is recorded by a customer's smartphone. In the first chapter of this report, we will look at the data structure and the specific pair of the data from each sensor. Using these sample data, we build the model and extend it to the whole data set in the later chapter. Lastly, the instructional manual for the written `R` function will be provided.

Data Preparation. There are two telematics data set from independent sensors: GPS in smartphone, OBDII. The two data sets are provided as two separate file. Here we assume that the two `json` files are extracted in the working directory, which means you have the following two files in the working directory:

mobile_trips.json, obd2_trips.json

Data load and structure. Since the two files are recorded as `json` format, `jsonlite` package will be used to load the data.

```
library(jsonlite)

# read mobile trip & obd2 trip
mobile_data <- fromJSON("./mobile_trips.json")
obd2_data <- fromJSON("./obd2_trips.json")
```

Smartphone data set consisists of 44 trips and OBDII data set consists of 41 trips. The colum names of the each data set and the data types are as follows:

- OBDII data
  - trip_id (`char`), timestamp (`dbl`), speed (`int`)
- Mobile data
  - trip_id (`char`), created_at (`dbl`), timestamp (`dbl`), speed (`dbl`), accuracy (`int`)

Visualization of sample trips. Fig 1 shows the speed graph of a sample trip from OBDII data. As we can see, the OBDII trip was recorded for about 1,200 seconds and we do not know the unit for the recorded speed. Fig 2 shows the speed graph of a sample trip from mobile data. Note that the scale of x-axis has been adjusted as zero. By examing the data set littel bit, it can be easily realized that the first trip from each sources corresponds to each other as in Fig. 1 and Fig 2. As we can see the whole trip data from smartphone corresponds to the trip data from OBDII around 300 seconds.

Determine the conversion factor between speed scale. Using this knowledge we found, we can figure out there is a conversion factor between the two different speed scale from each sources. If thoese sensor recorded the same trip, thier maximum speed should be the same. Thus, we can calculate the conversion factor as follows:

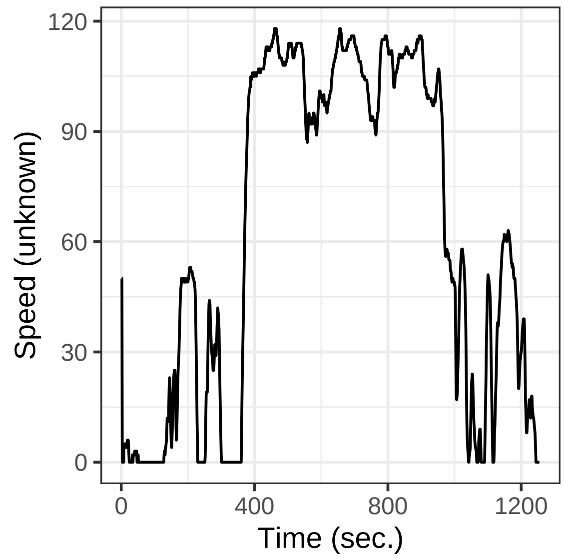

Fig. 1. A sample of speed graph of OBDII (the 1st trip).

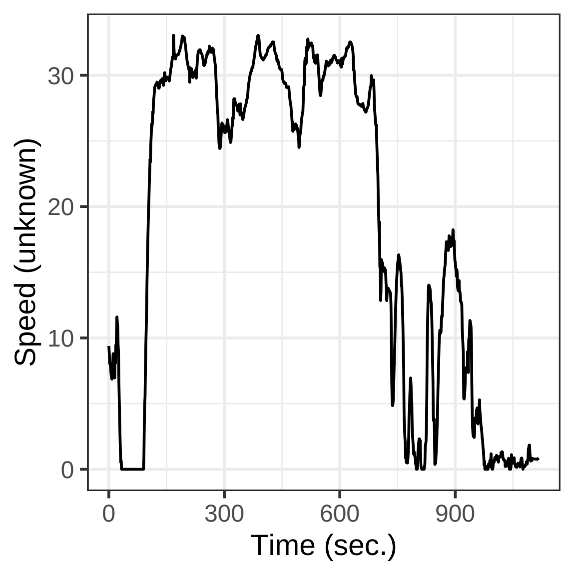

Fig. 2. A sample of speed graph of Smartphone (the 1st trip).

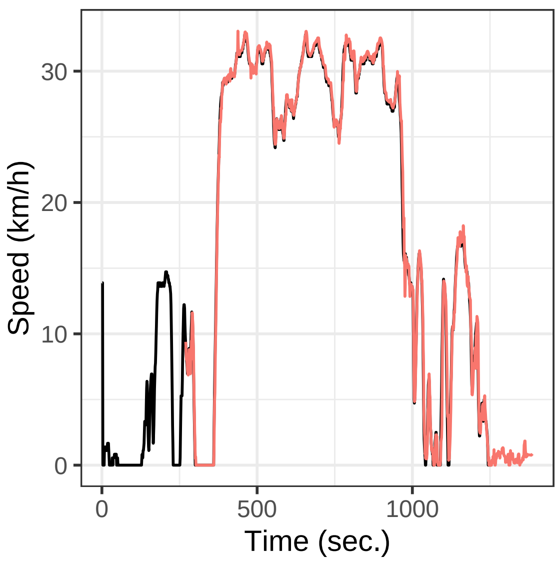

Fig. 3. A sample output of matched speed graph.

```
# Conversion factor
max(obd2_trip_sample1$speed) /
  max(mobile_trip_sample1$speed)
```

```
#  [1] 3.570348
```

We can see that it is around 3.6. There are many units for speed such as miles per hour (mph), kilometers per hour (km/h), etc. Using the cue that the smartphone sensor should use either `mph` or `km/h`, we can easily guess that the OBDII uses `m/s` since the relationship between `km/h` and `m/s` is as follows:

$$1\ m/s = 3.6\ km/h$$

Thus, using conversion factor and the lagging time (300 sec.), we can guess our final output should be similar to Fig 3. From now on, we will use `km/h` as a speed scale since the result plot re-confirms that the unit of each sensors.

Lagging time detection algorithm. The next step is to automatically determine the lag time given that we have two matched trips. We will use the same sample trips used in the previous section. In Fig. 3, the exact lag time for smartphone trip data was 270 seconds. To detect the lagging time, we need to consider every possible combination of these two trip by using sliding window algorithm. A nice visualization of this concept can be found at here. We consider the speed graph from OBDII as a fixed function and the speed graph from smartphone as a floating function in the algorithm.

Let $x \in \mathbb{R}^+$ be a real positive vector of size n_x whose elements represent the speeds or a trip from smartphone, while $y \in \mathbb{R}^+$ and n_y represents the speed and the size of the speed vector from OBDII repectively.

$$x = (x_1, x_2, ..., x_{n_x})^T$$
$$y = (y_1, y_2, ..., y_{n_y})^T$$

To implement the sliding window algorithm, we uses a dummy variable `k` from 1 to $n_x + n_y$ to search all the possible overlapped combination of the two graph. For example, when $k = 1$, we consider the situation where x_{n_x} and y_1 are overlapped each other. When $k = 2$, (x_{n_x-1}, x_{n_x}) and (y_1, y_2) are considered. Thus, for any $k \leq n_x + n_y$ where $k \in \mathbb{N}$, the overlapped vectors, x^* and y^* can be written as follows;

$$\begin{aligned} x^* &= (max(n_x - k, 1), ..., n_x - max(0, k - n_y))^T \\ y^* &= (max(k - n_x, 1), ..., min(n_y, k))^T \end{aligned} \qquad [1]$$

Equation 1 shows the compact expression for the three cases:

- Case 1: $k < n_x$ and $k < n_y$
  - $x^* = (n_x - k, ..., n_x)$
  - $y^* = (1, ..., k)$
- Case 2: $k > n_x$ and $k < n_y$
  - $x^* = (1, ..., n_x)$
  - $y^* = (k - n_x, ..., k)$
- Case 3: $k > n_x$ and $k > n_y$
  - $x^* = (1, ..., n_y - (k - n_x))$
  - $y^* = ((k - n_x), ..., n_y)$

Measure for the similarity. There are many measures for the similarity of the two functions whose domains are the same: area under the difference of the two functions, maximum difference of the two functions values, etc. Among these, the following measure is used for detecting the similarity between the two speed vectors:

$$f(x, y) = \frac{\sqrt{\left(\sum_{i \in A} (x_i - y_i)^2\right)}}{|A|} + \frac{\lambda}{|A|} \qquad [2]$$

where the vector x and y are the speed vector from smartphone and OBDII, and the set A is the collection of the pair of coordinates of OBDII and smartphone speed vectors overlapped each other for fixed k. Note that the function $|.|$ indicates the caldinality of a set. The reason of the division in Equation 2 is to calculate the average of the errors.

Also, the second term can be thought as a penalty function that prevents the case where the length of the overlapped is too short, so the dissimilarity is too small. We can put the weights on the panalty using λ, and we uses 10 for λ value for this case. Since the value of $f(x, y)$ decreases when the two vector x and y are similar to each other, the interpretation of the measure should be dissimilarity of the two vector. Fig. 4 shows the dissimilarity with respect to the k from 1 to 2354, which is the summation of the length of speed vector from OBDII and smartphone. The index which makes the dissimilarity to be the smallest value is $k = 1374$. According to Equation 1, this corresponds to the 255th time stamp of the OBDII trip.

Find the best matched trips. In the previous section, we have discussed how to find the lagging time using dissimilarity measure. To find a best matched trip among the collection of the OBDII trips for a given smartphone trip, we can find the OBDII trip whose minimum of the dissmilarity with the given smartphone trip is the lowest among the whole collection of the OBDII data set. However, since we cannot guarantee that the lowest minimum of the dissmilarity implies that the two trip from each sources actually matched each other, we set 0.1 as a threshold. Thus, if the lowest minimum dissimilarity is less than 0.1, we decide that the pair of OBDII trip and smartphone trip are matched each other. Fig. 5 represents the minimum of dissimilarity for each trips in OBDII

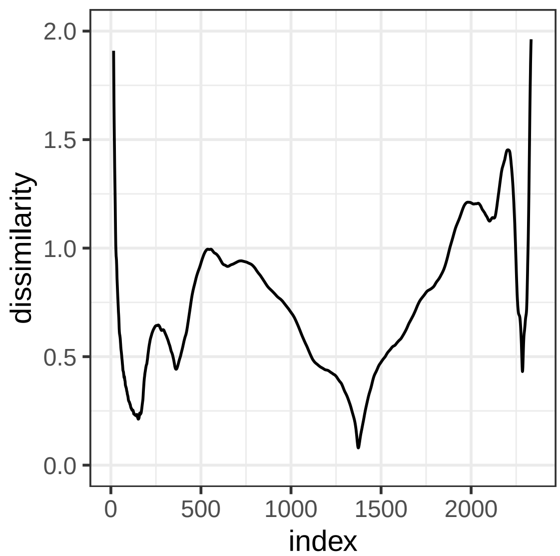

Fig. 4. Dissimilarity measure for w.r.t. k

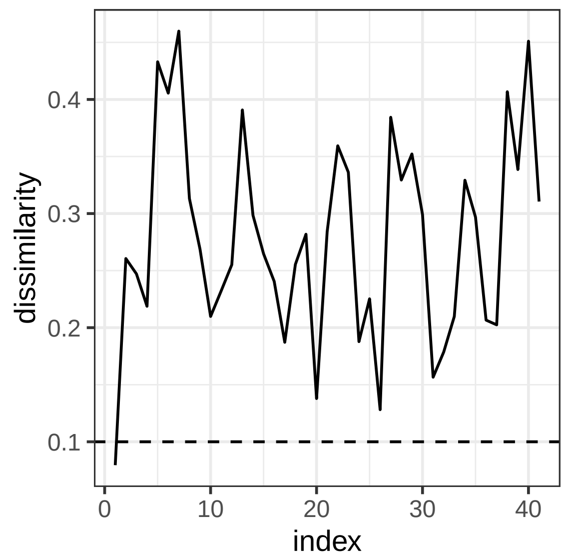

Fig. 5. Minimum values of dissimilarity for each trips in OBDII data set

data with the first trip in mobile data set. The dotted line, in Fig. 5, indicates the threshold for matched trip. Since the minimum of the dissimilarity of the first trip is less than the 0.1, we choose the first trip in OBDII data as the best matched trip with the first trip in mobile data.

User instruction.

Load data. You can load the JSON file into R using `jsonlite` package as at the beginning of this document.

```
library(jsonlite)

# read mobile trip & obd2 trip
mobile_data <- fromJSON("./mobile_trips.json")
obd2_data <- fromJSON("./obd2_trips.json")
```

Find the best OBDII trip for given smartphone trip. The following code will find the best OBDII trip which matches with `given_trip` mobile trip, and save the information into `match_info`.

```
# Select the second trip in the mobile data set
given_trip <- mobile_data[[2]]

# Find the best trip from OBDII data set
match_info <- FindBestTrip(given_trip, obd2_data)
```

The `match_info` variable has the matched result information. We can see that there are seven information as follows:

```
summary(match_info)
```

```
#                  Length Class  Mode
#   start_index_ref 1      -none- numeric
#   start_index_tar 1      -none- numeric
#   overlap_length  1      -none- numeric
#   dissimilarity   1      -none- numeric
#   macthed_trip    1      -none- numeric
```

```
# Visualization
VisTrip(given_trip, obd2_data, match_info)
```