# Travel Time Estimation Using Floating Car Data

*Raffi Sevlian*

This project explores the use of machine learning techniques to accurately predict travel times in city streets and highways using floating car data (location information of user vehicles on a road network). The aim of this report is twofold, first we present a general architecture of solving this problem, then present and evaluate few techniques on real floating car data gathered over a month on a 5 Km highway in New Delhi.

## 1 Floating Car Data Based Traffic Estimation

Data used to estimate the travel times on road networks come in two varieties, one being fixed sensors on the side of the road such as magnetometer detectors or highway cameras [5, 6]. The second method is floating car data (FCD). Floating car data are position fixes of vehicles traversing city streets throughout the day. The most common type of FCD comes from taxi's or delivery vehicles which are on main arterial roads and highways throughout most of the day.

This second approach has many positive and negative attributes that must be dealth with to provide accurate travel time inference. First, FCD is perhaps the most inexpensive data to attain, since many taxi services, and delivery companies automatically gather this data on their vehicles for logistic purposes. Second, position fixes are generally very accurate, since GPS is used and this has high accuracy. There are however many disadvantages as well. First, FCD is usually sampled infrequenty, on the order of 2-3 minutes. The reason for this is that taxi or delivery companies do not need such fine time granularity of their vehicles position. Therefore quiet a bit of preprocessing needs to take place in order to "snap" sets of points onto the proper streets with the possibility that multiple paths might have lead to the same pair of traverse points. Another disadvantage of this method is that a high density of data is required to get meaningfull travel time predictions for a given road network. Keeping all this in mind, constructing more and more accurate travel time predictions can be a fruitful Algorithms/Machine Learning/Statistical Modelling problem with various problems to tackle.

## 2 Prediction Architecture

Building a traffic estimation, system using millions of incoming FCD streams is a computational and algorithmic challenge. Various subcomponents requiring significant research work need to be developed and integrated. Here we give a breif overview of the various processing steps required as well as the issues explored in this report. Figure (2) presents a high level view of such a system.

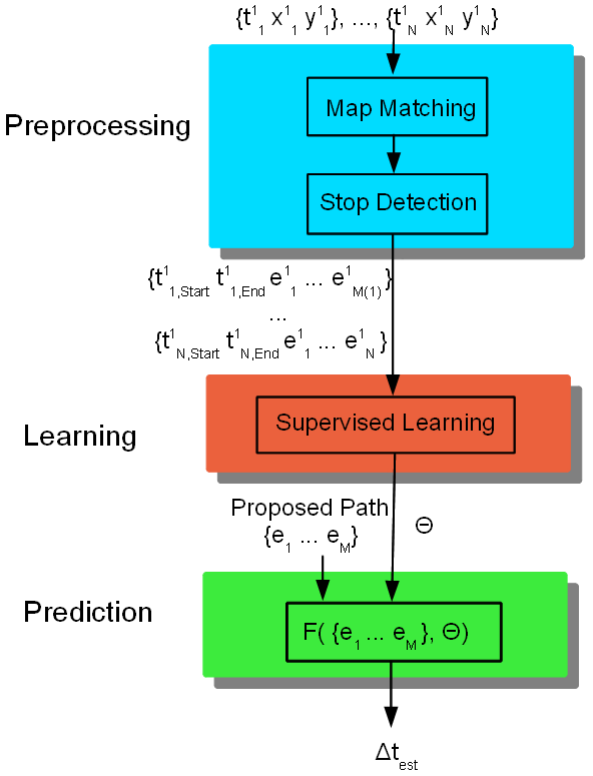



Fig. 1: Travel Time Prediction Architecture: For single user, input is sequence of $\{t_1 x_1 y_1\} \ldots \{t_N x_N y_N\}$.

# 3 Preprocessing

## 3.1 Motion Detection

Since the incoming streams of traffic data come from taxi cabs, delivery vehicles, or other comercial fleet vehicles, there will always be a certain amount of ambiguity between a slowdown in traffic and a commercial stop by the vehicle. (i.e. for a taxi customer, or a delivery vehicle dropping off packages). Therefore, any further processing must clean out all unwanted stops that are in the GPS logs. The most common and inuitive technique, and that which is used in this project is to track the the number of consequtive GPS points that are less than a specified distance $D_{max}$ from each other.

At each iteration a running tally $N_t$ is kept for the number of GPS points that are within $D_{max}$ of each other. If $N_t$ is greater than some threshold $N_{max}$ then the previous $N_{max}$ points are labeled as invalid. To compute the threshold parameters ($N_{max}D$), a supervised learning algorithm is can technique. In live taxi data, the taxi driver will activate a counter to track the distance crossed by the vehicle and the time spent farying the customer accross town, this is used to construct the proper labels for motion and stopping of the taxi. If this secondary informaiton is not available, another option is to visually inpect a large data set of GPS data, and manually identify the regions where a stop has taken place.

## 3.2 Map Matching

Map Matching is a widely studied problem in transportation research is perhaps the most computaionally difficult and important subcomponent. The input is a time indexed sequence of GPS coordinates $\{t_1 x_1 y_1\} ... \{t_N x_N y_N\}$, where $t_i$ is a standard unix timestamp data structure, and pairs $(x_i y_i)$ represent latitude and longitude coordinate. Map matching attempts to efficiently and accurately match consecutive points to sequence of links representing road segments in a stardard city map. Standard map matching outputs, $\left\{t_{1,Start}\, t_{1,End}\, \left\{e_n^1\right\}_{n\in E_1}\right\} \ldots \left\{t_{N,Start}\, t_{N,End}\, \left\{e_n^N\right\}_{n\in E_1}\right\}$, here$\left\{e_n^i\right\}_{n\in E_1}$ represents a set road segment links for the $i^{th}$ path inthegral. Also, $E_i$ is the set of all indices representing such path edges. Figure (4.1) shows a simple example of map matching.

Map matching is done in a variety of ways depending on a tradeoff of accuracy and efficiency. For the task of realtime high capacity map matching, several techniques involving hueristic graph search are shown in [1, 2, 3]. These techniques follow a general strategy of greedily adding road segments to a solution set as points are processed, each candidate road segment receives a score by a distance function defined on road segments and GPS probes. A standard distance function is is given by finding the shortest distance between the GPS point and point on the road segment.

Given a road segment defined by all convex combinations of two GPS coordinates, $A$ and $B$. $e_{A,B} = \{(x,y) : \alpha \in [0,1], (x,y) = \alpha A + (1-\alpha) B\}$ points inside the road segment are defined as, $e_{A,B}(\alpha) = \alpha A + (1-\alpha) B$. With the standard distance function used map matching algorithms is.

$$d(\{xy\}, e) = \begin{cases} \min\limits_{\alpha} d_{GPS}(e_{A,B}(\alpha), \{xy\}) & \measuredangle(\{xy\},A) < \frac{\pi}{2} \text{ and } \measuredangle(\{xy\},B) < \frac{\pi}{2} \\ \min\{d_{GPS}(A,\{xy\}), d_{GPS}(B,\{xy\})\} & else \end{cases}$$

Where $d_{GPS}(p_1\, p_2)$and $\measuredangle(p_1, p_2)$ represent standard geographic distance and angle between two GPS coordinates. After the candidate pool grows to a specified size, pruning via a hueristic shrinks the candidate pool.

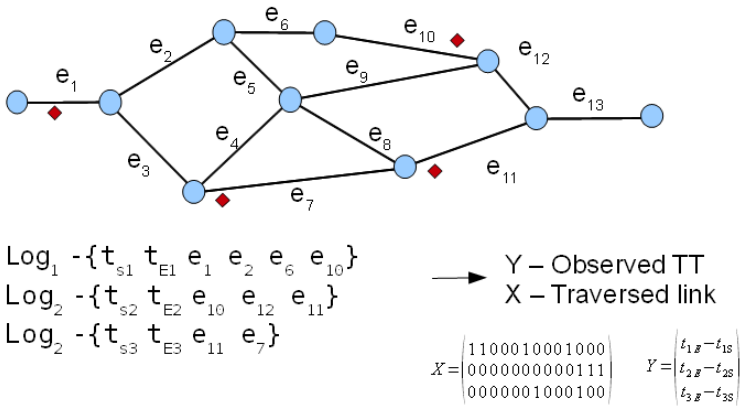



Fig. 2: Example of map matching of set of GPS points. Input is set of points with outputted path integrals.

## 4 Supervised learning for Traffic Estimation

This report explores the use of supervised learning for traffic inference on links on a road network. Each technique relies on different modelling assumptions and produces different estimation results and has particular advantages and disadvantages that dictates use in individual situations.

The most general form of learning and prediction is that after learning takes place, some model parameterized by $\Theta$ will be available. Some prediction function, $F\left(\left\{e_n^i\right\}_{n\in E_i};\Theta\right)$ will use the learned parameters, and output an estimate of the travel times through a new set of links $\left\{e_n^i\right\}_{n\in E_i}$.

### 4.1 Regression Technique:

Regression techniques rely on an additive model for travel times. That is the travel time for a commuter traversing a set of links will be the sum of the travel times for the set of links traversed. Therefore if the current link travel times are already known, then the estimated travel time for some path through the network in figure (4.1) will the sum of the link travel times.

$F\left(\left\{e_n^i\right\}_{n\in E_i};\Theta\right)=\sum_{n\in E_i}t_\Theta\left(e_n\right)$

Under a regression model, each $t_\Theta\left(e_i\right)=\theta_i$ therefore given link travel times, any observed travel time will follow

$y^i|\sum_{n\in E_i}t_\Theta\left(e_n\right)\sim N\left(\sum_{n\in E_i}t_\Theta\left(e_n\right),\sigma^2\right)$

$Y=X\Theta+E$

Throughout this project, we make different assumptions on the observed travel times depending on whether long term averages, or short term deviations are being computed.

### 4.1.1 Long Run Historical Travel Time

A goal of traffic flow analysis is having long term historic data on flow volume through various links. To construct a regression model taking into account spatial variation of historic TT values, we can assume that adjacent links are distributed normally:

$t(e_n) - t(e_{n-1}) \sim N(0,\, \tau^2)\ \ n = \{1, \ldots, N-1\}$

This enforces a penalty on sudden changes to the historic means, as well as makes estimation of parameters tractable when the maximum likelihood estimator is underdetermined. With this assumption we construct the following maximum a posteriori estimator.

$$\begin{aligned}\Delta &= \arg\max_{t(e)} \textstyle\sum_i log\, P\left(y^i | \sum_{n\in E_i} t(e_n)\right) + \sum_m log\, P\left(t(e_m) - t(e_{m-1})\right) \\ &= \arg\min_{t(e)} \textstyle\sum_i \frac{1}{\sigma^2} \left\|y^i - \sum_{n\in E_i} t(e_n)\right\|_2^2 + \frac{1}{\tau^2} \left\|\sum_m t(e_m) - t(e_{m-1})\right\|_2 \\ &= \arg\min_{\Theta} \|Y - X\Theta\| + \lambda_1 \|D\Theta\|_2\end{aligned}$$

Which is simply a Ridge Regression or L2 Regression with smoothness penalty on the variation of historic mean along different links [6].

### 4.1.2 Short Term Incidence Detection on Live Data

Given a short time window (on order of 1 hour) we assume the distribution on any given day for link $t(e_i) = \theta_i + \Delta_i$where $\theta_i$is a historical average and $\Delta_i$is random variable representing the deviation that day from the historical average. The important assumption here is that this deviation is distributed with a heavy tail. (i.e. no deviation with high probability and large deviation with low probability). To model this, we use a laplacian prior distrbution: $p(\Delta_i; \sigma) = e^{|\frac{-\Delta}{\sigma}|}$.

Given a set of historic data, and a small number of measured daily measured paths, we can use this model to computer short term deviations from the historic values.

$$\begin{aligned}\hat{\Delta} &= \arg\max_{t_\Theta(e)} \textstyle\sum_i log\, P\left(y^i | \sum_{n\in E_i} t_\Theta(e_n)\right) + \sum_m log\, P\left(t_\Theta(e_m)\right) \\ &= \arg\min_{t_\Theta(e)} \textstyle\sum_i \frac{1}{\sigma^2} \left\|y^i - \sum_{n\in E_i} t_\Theta(e_n)\right\|_2^2 + \frac{1}{\tau^2} \sum_m \left\|t_\Theta(e_m)\right\|_2^2 \\ &= \arg\min_{\Delta} \|Y - X(\Theta + \Delta)\|_2^2 + \lambda_2 \|\Theta + \Delta\|_1\end{aligned}$$

Therefore in every small time window that live probe data arrives, an L1 regression is performed estimate deviations from the historic travel times. Since probe volume is small compared to the number of links, this technique makes intuitive sense. When performing live prediction we assume that the parameters $\{\Theta, \lambda_1, \lambda_2\}$ are already computed. Therefore in each time window used for regression, the parameters $\{\Delta\}_i$ are computed and predictions are given by $\sum_{n\in E_i} (\theta_n + \Delta_n)$.

### 4.1.3 Computing Model Parameters

In modelling travel time distributions, we explicitley separate long term and short term behaviour to easily estimate the model parameters. With this, parameter estimation is performed in two stages. First, a subset of the training data is aggregated and crossvalidation is performed to determine $\Theta$ and appropriate $\lambda_1$. In the second stage, the short term model parameter $\lambda_2$ is computed with similar cross validation with training data (See note (1)). This data processing model is shown in figure (4.1.3). This is a visual interpretation of how data processing takes place. Each block is a tuple of paths and travel times (i.e. X, and Y from figure (4.1)).

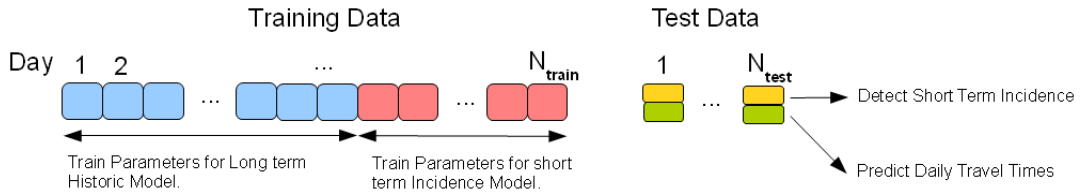



Fig. 3: Data Processing model for estimating historic means for link travel times, as well as L1 regression parameters. Test data used daily to compute short term deviations from historic average.

Tab. 1: Input Data Statistics: 4/5/2008 - 4/26/2008 (22 days, 8-9 AM)

| min | max | mean | std | |
|---|---|---|---|---|
| 34 | 110 | 66.3 | 27.5 | Raw Data in Sector |
| 11 | 65 | 26.8 | 13.6 | Processed Path Integrals |

[1]

## 4.2 Travel Time Median Backprojection:

The second general technique used in travel time estimation is to merely backproject the travel time distances across each traversed link weighted proportionally to the lenght of the links. After many paths are processed, each link will map to a list of backprojected travel time values. The travel time estimate of the link is simply the median over the distribution. We use this algorithm as a naive baseline to compare the more sophisticated model.

# 5 Experimental Results

## 5.1 Preprocessing Results

Components of the system architecture were implemented according to figure (2) to process floating car data from approximately 40 Taxi's over the course of about one month in urban New Delhi. The total dataset of GPS logs span 22 days: dates 4/5/2008 to 4/26/2008. The data was sampled infrequently at about 2-3 minutes. The reason for this is that some GPS devices sampled every 2 minutes while some sampled every 3 minutes. Since the focus of this report is the core inference techniques, a single 5 Km highway (figure (5.1)) was chosen for analysis instead of a region of the city, or the entire city. instead of an entire city. Another reason why a long highway was chosen is that the fundemental assumption that link travel times can be added to estimate the combined travel time is a more accurate model for highways than for city arterial roads [4].

[1] Recall that this separation between long term historic data and short term fluctuations is an assumption we explicitly make. A more thorough model would merely include daily laplacian deviations into the first model. However comstructing an estimator for this latent variable model, requires an EM algorithm. This is a definately a future extension of our work.

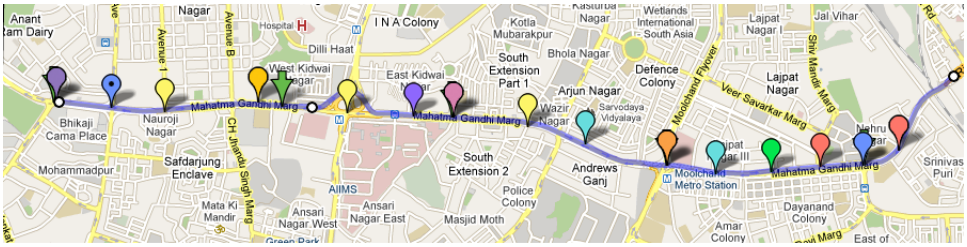



Fig. 4: Single road learning experiment.

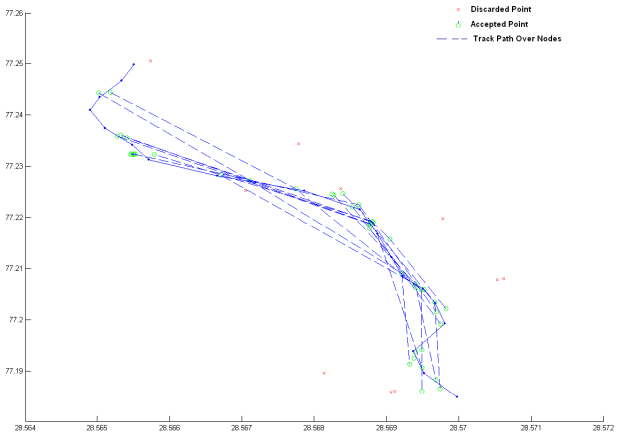
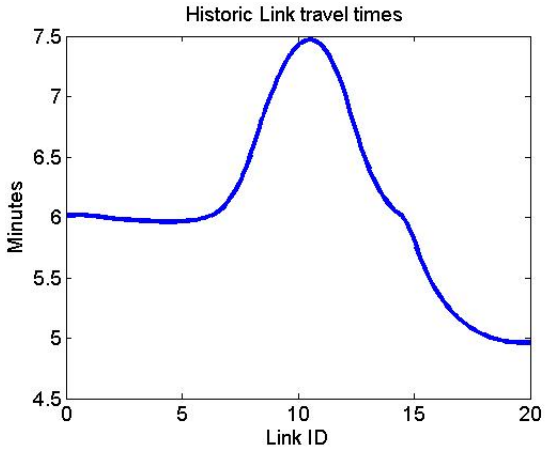
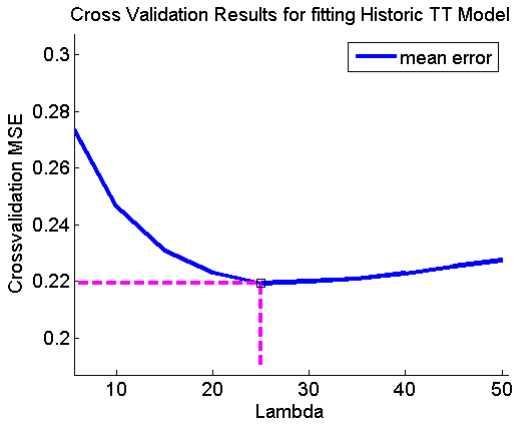



Fig. 5: Track Creation process - Primary reprocessing step.

The data included all FCD logs that were contained in the geospatial coordinates shown in figure (5.1). Therefore much of the data belonged to taxi's that might have traversed nearby streets. To isolate the data required for inference of the highlighted street, full map matching on the data was implement and used to seperate FCD data traversing nearby streets. Additionally, stop detection as described earlier was used to further remove artifacts that would cause false positives in incident detection. Statistics on the raw input data and final post processed data are given in table (5.1). For this project, additional processing was done to remove any artifacts caused by missed detections of taxi stops. From table (5.1) it is apparent that a significant proportion of the data contained stops or belonged to paths on nearby streets.

Figure (5.1) shows a sample of the processing done. The each green point was determined an acceptable path integral end point, while red points were rejected. The dashed line indicates the start and end of each path integral. All logs for this example were sampled uniformly. By visual inspection, one can infere that the upper left quadrant will have much lower travel times, than the lower right portion of the street.




Tab. 2: Main Results comparing 3 algorithms.

| Technique | Error Rate (% of True) | Std Dev. | N | 95 % Confidence Interval |
|---|---|---|---|---|
| Historical Average | 26.82 % | .10 | 234 | [25.53, 28.1] |
| Historical Average And Incidence Detection | 22.1 % | .12 | 234 | [20.56, 23.6] |
| Median Backproject | 32.4 % | .14 | 234 | [30.60, 34.19] |

## 5.2 Inference Results

Supervised learning of post processed data follows the model presented in figure (4.1.3). Here, $N^1_{train}$ blocks (each representing one day's paths) were chosen to learn the historical travel time model. From the entire $N^1_{train}$ 5 fold cross validation was used with varying model parameter $\lambda_1$ to estimate the optimal historical travel time $\Theta$. That is the $\lambda_1$ yielding the lowest cross validation error for the data was chosen as the optimal parameter. Figure (5.2) shows the results from cross validation; the historical travel times, optimal $\lambda_1$and minimum training error. Cross validation error for training historic travel times yielded an optimal $\lambda_1 = 24.45$ and $MSE_{min} = 22\%$.

Fitting the parameters for the daily incidence model follows an identical methodology as for the long term historic mean estimation. $N^2_{train}$ days of data are used, wher for each day, leave one out cross validation (since the number of path integrals is on the order of 10 - 15) is performed where parameter $\Delta$ is estimated with the training data and used to predict the travel time of the one out path integral/TT. As before, $\lambda_2$ is varied and an optimal value can be determined by looking at the lowest test error.

## 5.3 Test Prediction

Finally with the model parameters computed, the incident detection technique is tested as follows: All data not used in the first two phases ($N_{test}$) is tested for each individual day, whereby all but one of the path integrals is assumed to be the day's probe data (i.e. leave one out model validation) . This data is used to to find the sparse deviations from the historic mean $\{\Delta\}$ by solving the LASSO problem presented earlier. The deviations are then incorporated with the historic mean to provide the predicted travel time.

In this report we tested 3 separate algorithms to compare their results. Multiple random assignments were made between the 22 files and the subsets used for training and testing ($N^1_{train}, N^2_{train}, N_{test}$). First we merely use the historic means as a default estimate, without updating the link travel times by any live data. Second we apply the incidence detection technique introduced in the paper. Finally we apply the naive median backproject technique. Table (5.3) summarizes the final computed test errors for the different techniques discussed. The results show an improvement of incidence detection over merely using the historical average via regularized least square. The worst technique was the median backproject. The error values are in percentage of true link travel time. A simple statistical analysis of the incidence detection results shows that the improvement is statistically significant. Under 95% confidence interval there is a statistically significant difference in each algorithms results.

## 6 Conclusions and Future Directions

This report presents a unified architecture for Floating Car Data based travel time inference as well as proposes an incidence based inference technique that is evaluated and shown to be quiet effective at estimating travel times on various links in transportation networks. The largest impediment in achieving more accuracy appears to be the amount of data required. The road network was chosen since from the data available, it had the highest density of FCD. Regardless, at the rush hour time of 8-9 AM an average of 26 or so path integrals were constructed. Therefore future tests require a higher density of FCD. Also, a unified EM algorithm (mentioned in the footnotes) for training the historical data, while incorporating the sparse deviations model should be investigated.

## References

[1] Dongdong Wu, Tongyu Zhu, Weifeng Lv, Xin Gao; , " A Heuristic Map-Matching Algorithm by Using Vector-Based Recognition," Computing in the Global Information Technology, 2007. ICCGI 2007. International Multi-Conference on , vol., no., pp.18-18, 4-9 March 2007

[2] Dongdong Wu, Tongyu Zhu, Weifeng Lv, Xin Gao; , "A Quick Map-Matching Algorithm by Using Grid Based Selecting," Computing in the Global Information Technology, 2007. ICCGI 2007. International Multi-Conference on , vol., no., pp.18-18, 4-9 March 2007

[3] Marchal F., J.K. Hackney, K.W. Axhausen," Efficient Map-Matching of Large GPS Data Sets-Tests on a speed monitoring experiment in Zurich ", STRC 2005

[4] Rajagopal R. " Large Monitoring Systems: Data Analysis, Design and Deployment ". 2009.

[5] Kwong, R. Kavaler, R. Rajagopal, and P. Varaiya. "Arterial travel time estimation based on vehicle re-identification using wireless magnetic sensors". Transportation Research Part C. 2009

[6] Hastie, T. Tibshirani R, and Friedman J. The Elements of Statistical Learning: Data Mining, Inference, and Prediction , 2008