A real time forecasting tool for dynamic travel time from clustered time series

Andres Ladino Lopez, Alain Kibangou, Carlos Canudas de Wit, Hassen Fourati

A real time forecasting tool for dynamic travel time from clustered time series

A. Ladino[a], A. Y. Kibangou[a], C. Canudas de Wit[a], H. Fourati[a]

[a]*Univ. Grenoble Alpes, CNRS, Inria, Gipsa-Lab, F-38000 Grenoble*

Abstract

This paper addresses the problem of dynamic travel time (DTT) forecasting within highway traffic networks using speed measurements. Definitions, computational details and properties in the construction of DTT are provided. DTT is dynamically clustered using a K-means algorithm and then information on the level and the trend of the centroid of the clusters is used to devise a predictor computationally simple to be implemented. To take into account the lack of information in the cluster assignment for the new predicted values, a weighted average fusion based on a similarity measurement is proposed to combine the predictions of each model. The algorithm is deployed in a real time application and the performance is evaluated using real traffic data from the South Ring of the Grenoble city in France.

Keywords: Traffic forecasting, travel time forecasting, sensor fusion, clustering, Kalman filter

1. Introduction

Short-term travel time forecasting is one of the most important tools for traffic management and it has received a lot of attention during the last decade. This kind of tool has been widely requested by traffic management operators and drivers using traffic infrastructures. In the actual era of data deluge, measurements collected by multiple sensors are important sources of information that require analysis, classification and processing in order to detect patterns and behaviours that can be exploited in traffic prediction. Technological and analytic solutions have been the focus of several research papers along the last years. For the sake of completeness, [1] presented a collection of methods for traffic forecasting, and a more recent updated study is summarized in [2]. For data driven approaches, the collection of information can be classified in order to facilitate the data analysis. The separation of the information into multiple groups can be achieved through unsupervised learning techniques such as K-means, where each cluster is created in an automatic way from traffic data patterns, those can characterize in some cases typical regimes such as congestion. This information can be obtained from different variables such as travel time, queue length, density, delay, among others. These indicators qualify and provide the status of the network and multiple tools have emerged to extend this processed information to the users.

Email addresses: andres.ladino-lopez@gipsa-lab.fr (A. Ladino), alain.kibangou@gipsa-lab.fr (A. Kibangou), carlos.canudas-de-wit@gipsa-lab.fr (C. Canudas de Wit), hassen.fourati@gipsa-lab.fr (H. Fourati)

This work is motivated by the forecasting problem originally proposed in [3] based on the existence of traffic regimes identified from a clustering process. The main idea to reconstruct the forecast relies on the existence of travel time patterns identified along the history as described in [4] and [5] when large historical data sets are observed. The problem has been already studied in the literature from different approaches, here we raise the more relevant recent works related to this one. [6] proposes a travel time forecast based on an Adaptive Kalman Filter (AKF) strategy in which observations are built from historical data set of speed and flow. [3] considers a similar strategy in flow prediction and presents the problem of multi-step ahead forecasting based on clustered time series by applying several predictors such as Gaussian maximum likelihood (GML) and AKF. [7] presents an approach for short-term flow forecasting using multiple ARMAX based predictors obtained from clustered data. The ARMAX model is adapted independently to different groups of flow time series and a single prediction is selected based on one criteria that considers minimum error estimation for the predicted signals. [8] uses Link Node Cell Transmission Model calibrated via Monte Carlo methods in order to generate a prediction using the expectation maximization algorithm. Recently [9] has proposed data driven strategies based on KNN ($K-$nearest neighbors) selection in order to produce the forecast. All the aforementioned methods consider the selection of single forecast among multiple candidates in order to provide an outcome for the prediction phase. Although different criteria can be established for the selection, regimes described by clustered data are not totally separable and studies like [10] have shown the improvement of performance with combined forecasts. New methods have been emerging to combine information from these models inspired from data fusion algorithms such as [11]. For instance [12] proposes a forecast strategy based on interactive multiple models by combining different individual forecasting methods. [13] proposes an adaptive fusion method combining historical information and current day data. Lately [14] proposes a fusion of the $K-$nearest neighbours information where the terms are weighted according to a spatio-temporal correlation and [15] proposes the combination of multiple forecasts through a weighted least squares optimization problem in which the weights are determined by the covariance error matrix.

In this study we consider the dynamic travel time (DTT), this indicator provides useful information for multiple users and it is common in actual traffic simulators and applications such as [16], [17]. The approach in this work constructs observations for the DTT as explained in [18], [19] (also referred as the experienced travel time for a driver) for historical data and current day data. Generally, forecast algorithms in the literature are designed to satisfy a set of constraints given by the forecast problem. Most of the algorithms take into account availability of a full set of measurements for all possible locations and time instants within the traffic network and they overcome the problems of missing data, low penetration ratio or unbalanced spatial coverage by introducing additional steps such as imputation algorithms. First, we explain the data collection and aggregation process, then, we consider a simple imputation algorithm for a real time application in order to address this issue and we present results about the performance of the applied data imputation. From the imputed data, a clustering approach is applied defining then different clusters characterized by a centroid containing the mean of the data and a given dispersion around it. The evolution of the centroid can be used as future observation that can feed a Kalman filter. Therefore the prediction problem can be viewed as a filtering one. Nevertheless the assignment of the observation during the current day to a specific cluster remains an open issue since we don't know its future. To overcome this issue, we run a Kalman filter for each cluster and then we make the fusion of the obtained forecasts. The main contribution of this paper is to apply a fusion method based on a similarity distance between the known information of current day and clusters in the history. The full mechanism is deployed in a real time application and the behaviour of this strategy over the DTT is explored. The performance of the proposed method is evaluated using a real traffic data from the Grenoble Traffic Lab (GTL) [16].

The remainder of this work is structured as follows. Next section will present the data workflow and the main definitions for dynamic and instantaneous travel times, considering real data scenarios. In section 3 we present the clustering method used to identify and separate traffic regimes. Section 4 describes the forecasting algorithm and the interpretation of data. Finally we present the main results of the methodology over the real time application [16]. For the sake of completeness, we have included an appendix dedicated to the data imputation problem and its effect on the computation of the DTT.

1.1. Notations

n : Total number of links.

$v(x,t)$: Velocity space/time continuous field.

$v(x_i, s_j)$: Discrete velocity space/time field.

$DTT(x_0, x_n, t_0)$: Dynamic travel time (DTT) from x_0 to x_n at t_0.

$ITT(x_0, x_n, t_0)$: Instantaneous travel time (ITT) from x_0 to x_n at t_0.

$\tilde{v}_l(x_i, t_\tau)$: Imputed velocity from temporal coherence.

$\tilde{v}_s(x_i, t_\tau)$: Imputed velocity from spatial coherence.

$\tilde{v}_d(x_i, t_\tau)$: Imputed velocity from historical coherence.

$\mathcal{K}$: Number of clusters.

$D_{\mathcal{K}}$: Total distortion for a clustered set with $\mathcal{K}$ clusters.

$\mu_q(k)$: Cluster's centroid q at time t_k.

$\Delta\mu_q(k)$: Cluster derivative's centroid q at time t_k.

$y_h(k)$: DTT for day h at time t_k.

$w_{d,q}(k)$: DTT noise for day d with respect to cluster q

$R_q(k)$: Sample based variance associated to cluster q at time t_k.

$v_{d,q}(k)$: DTT noise derivative for day d with respect to cluster q

$V_q(k)$: Sample based variance associated to derivative of cluster q at time t_k.

$\bar{y}_{d,q}(k)$: Level measurement of the DTT of day d with respect to cluster q at time t_k.

$P_{d,q}(k)$: Variance of DTT model of day d with respect to cluster q at time t_k.

$K_q(k)$: Kalman gain for the q-th filter.

$\hat{y}_{d,q}(k)$: Prediction for DTT of day d with respect to cluster q at time t_k.

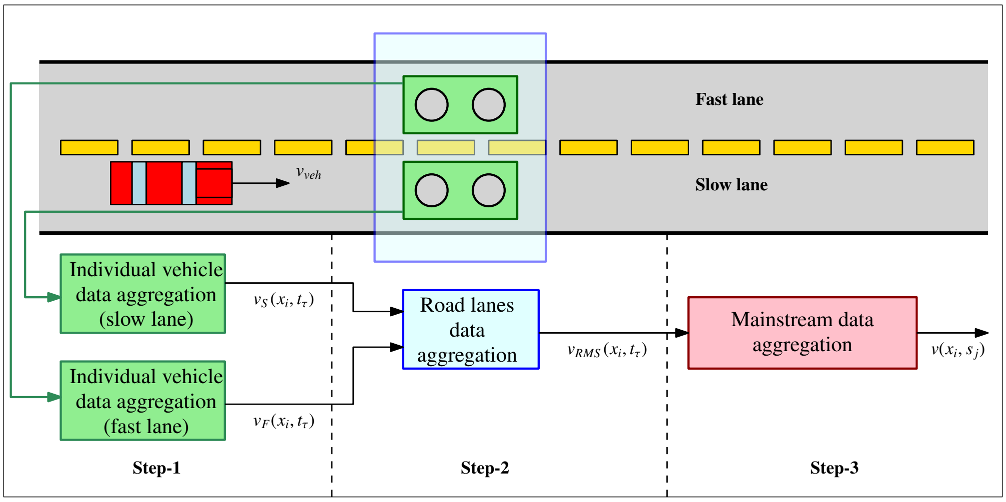


Figure 1: Schematic representation of the 3-levels data aggregation from the sensor samples to the final velocity used for travel time prediction.

$S_{d,q}(k)$: Temporal similarity measurement between day d and cluster q.

$U(k)$: Cluster weight similarity matrix .

λ_q: Weight assigned to the q-th cluster.

$\hat{y}_d(k)$: Prediction for DTT of day d at time t_k.

$AE(x_i, t_\tau)$: Absolute error in location x_i at time t_τ.

$APE(k_0 + h, d)$: Absolute percentage error for the prediction at departure time $k_0 + h$ for day d.

2. Speed collection and travel time models

In this section, we describe first the whole data aggregation process from each individual sensor measurement to the aggregated velocity used for the travel time computation. Second, some definitions of *instantaneous* and *dynamic* travel times for a given origin-destination path and the associated computations are introduced using the aggregated velocity.

2.1. Data aggregation process

Fig. 1 shows the sensor layout used in the GTL, an experimental platform for real time collection of traffic data coming from a dense wireless sensor network installed in the south ring of Grenoble, see [16]. Sensor technology, provided by SENSYS [20], consists in magnetometers deployed by pairs and separated about 5m at each lane. They provide aggregated data every 15sec: velocity flows, occupancies, but also histograms of vehicle lengths and speeds for each lane. The whole data aggregation consists in three main steps:

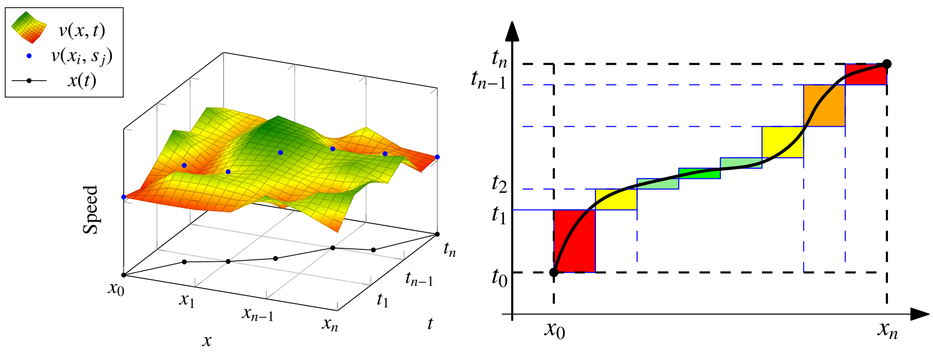
- *Step 1. Individual vehicle data aggregation.* Each sensor pair measures per lane: flow, speed and occupancy every time-interval $\delta = 15$sec at each location x_i. The aggregated lane speed, $v_{lane}(x_i, t_\tau)$, results from averaging all individual vehicle velocities $v_{veh}(x_i, t)$, collected at each lane during the time-interval $I_\tau = [t_\tau - \delta, t_\tau]$, $\tau \in \mathbb{Z}^+$. When no vehicle is detected, the system prevents such event by sending a specific code and then v_{lane} is replaced (See Appendix A for details).

- *Step 2. Road lanes data aggregation.* The lane velocities v_{lane} resulting from *Step 1* are now aggregated at the level of a *road measurement section* v_{RMS}. For this, we use a weighted average to better represent the speed center of gravity as a function of the number of vehicles per lane. The weights N_{lanes} correspond to the number of vehicles counted during I_τ.

- *Step 3. Main stream data aggregation.* Finally the road velocities v_{RMS} which are obtained every time-interval δ are now averaged over the database time-interval $\delta_s = N_m\delta$, $N_m \in \mathbb{Z}^+$ (in our case $N_m = 4, \delta_s = 1$min) which will be used to store information in the database. The velocity-heat map describing the velocity quantification in the discrete $v(x_i, s_j)$-map.

2.2. Travel time: definition and computation

Consider the case of single origin-destination path, and let $v(x,t)$ be the continuous space/time distribution of the velocity field along the considered single road segment as seen in Fig. 2.



(a) Space/time speed surface

(b) Speed updates for DTT computation

Figure 2: Space/time speed evolution for DTT computation

We define two approximations for the travel time given by:

Definition 1. *The dynamic travel time (DTT) from position x_0 to x_n starting at time $t = t_0$ and ending at time $t = t_n$ is defined as:*

$$DTT(x_0, x_n, t_0) = t_n - t_0 = \sum_{i=0}^{n-1} \frac{x_{i+1} - x_i}{v(x_i, DTT(x_0, x_i, t_0))}, \tag{1}$$

where $x_i, i = 1, \ldots, n-1$ stand for intermediate locations between x_0 and x_n.

Definition 2. *The instantaneous traveling time (ITT) from position x_0 to x_n starting at time $t = t_0$ and ending at time $t = t_n$ is defined as:*

$$ITT(x_0, x_n, t_0) = t_n - t_0 = \sum_{i=0}^{n-1} \frac{x_{i+1} - x_i}{v(x_i, t_0)}. \tag{2}$$

The DTT can be considered as the "true" travel time of a vehicle along the velocity flow whereas the ITT is a crude approximation of the true one. Definition 1 implies a recursive computation of the travel time and the computation select points as seen in Fig. 2a from discrete velocity field $v(x_i, s_j)$ as stored in the database. The DTT according to (1) for two consecutive locations is given by

$$DTT(x_i, x_{i+1}, t_k) = t_{k+1} - t_k \approx \frac{x_{i+1} - x_i}{v(x_i, t_k)}, \quad t_k := DTT(x_0, x_i, t_0).$$

Note however that the time argument, t_k, in $v(x_i, t_k)$ is a continuous variable (k is considered here as an event), and that velocity data is stored in a discrete-time database as $v(x_i, s_j)$. Therefore, index j needs to be defined from t_k to match $s_j = \lfloor t_k \rfloor$, that is $j(t_k) = \frac{\lfloor t_k \rfloor}{\delta_s}$. Then, DTT can be written in a compact form as

$$DTT(x_0, x_n, t_0) = \sum_{k=0}^{n-1} (x_{k+1} - x_k) v(x_k, s_{j(t_k)})^{-1}. \tag{3}$$

Similarly, the following formula is obtained for ITT

$$ITT(x_0, x_n, t_0) = \sum_{k=0}^{n-1} (x_{k+1} - x_k) v(x_k, s_{j(t_0)})^{-1}. \tag{4}$$

The error introduced in the ITT computation (we assume that the DTT is the true travel time) crossing homogeneous sections of length δ_x can be estimated as:

$$ITT_{error} = DTT - ITT = \delta_x \sum_{k=1}^{n-1} (v(x_k, s_{j(t_k)})^{-1} - v(x_k, s_{j(t_0)})^{-1}). \tag{5}$$

This error will grow as the traffic conditions experience important changes in the near future. In presence of traffic shock waves the term $(v(x_k, s_{j(t_k)})^{-1} - v(x_k, s_{j(t_0)})^{-1})$ will induce large differences. This case is illustrated in Fig. 3. Except for free flow situations, when the DTT and the ITT become close, important errors in the ITT are also expected when considering large and complex networks with heterogeneous traffic conditions along trajectory paths. Furthermore, ITT can be computed by the sum of the individual ITT values of each road section (region between two sensor points) while the DTT is computed by the accumulation of several terms, this effect adds computational complexity for multiple origin destinations to the algorithm and prediction strategies as the one developed in [15] cannot be applied directly.

Finally, note that the ITT can be straightforwardly computed using information at the current time t_0. However, to compute the DTT it is necessary to build predictions ahead in time for $v(x_k, t_0)$, namely to devise prediction algorithms for $\hat{v}(x_k, s_j), \forall s_j \in [t_0, t_0 + \Delta_f]$, where Δ_f is the future time-horizon necessary for such a prediction. The predicted DTT is then computed as

$$\widehat{DTT}(x_0, x_n, t_0) = \sum_{k=0}^{n-1} (x_{k+1} - x_k) \hat{v}(x_k, s_{j(t_k)})^{-1}, \tag{6}$$

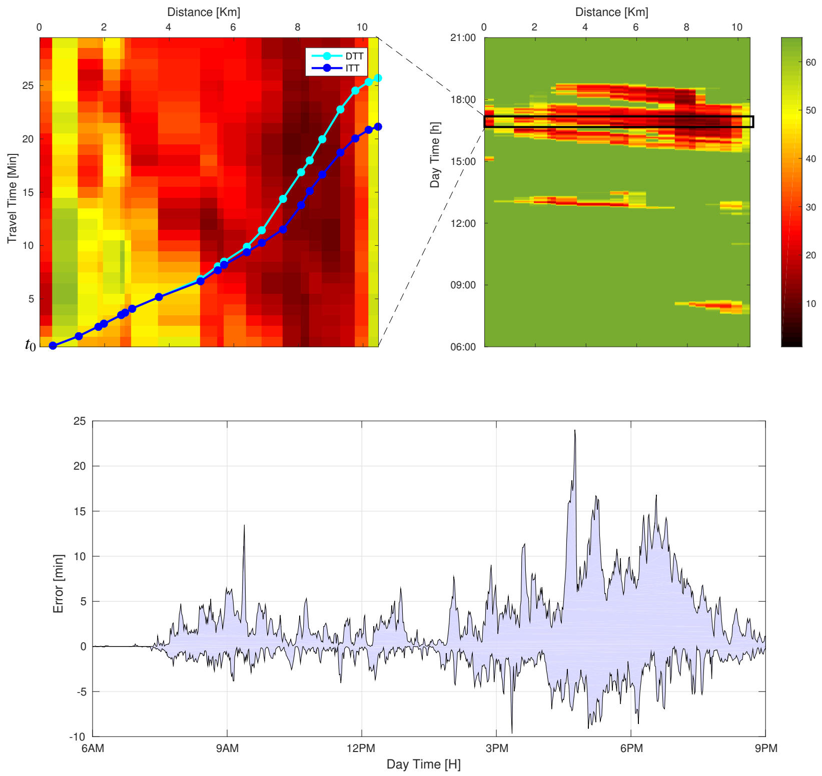


Figure 3: Top-Figure. Illustration of the time/space evolution of the DTT vs. the ITT computed with data from the GTL [16] for September 1^{st}, 2015. The heat map in the figure displays the measured speed profile in Km/h for the day. For this particular case the ITT under estimates the DTT and the main differences start to appear in the middle of the trajectory when the congestion wave reaches the vehicle trajectory, i.e at $x = 5$Km, $t = t_0 + 7$min. The free flow travel time is here 7min. Bottom-Figure. Error bounds incurred by the ITT computation from May to December 2015. Maximum error reaches up to 3 times the free-flow travel time.

with $\hat{v}(x_k, s_{j(t_0)}) = v(x_k, t_0)$, and $j(t_k) = \frac{\lfloor t_k \rfloor}{\delta_s}$.
The previous description on DTT computation, requires speed measurements to be complete. Nevertheless, data collection process previously described is affected by failures in acquisition, transmission, and storage. The main sources of failure are:

- *Acquisition errors*: When a vehicle, within the time interval I_τ, crosses just one of the two sensors in a specific location, the pair of sensors cannot measure the time window this vehicle took to cross the space between them (See Fig. 1). Therefore measuring its speed is not possible. In this case, an arbitrary value is set as speed measurement; the value -1 is used for Sensys network [20].

- *Transmission errors*: When data is sent by the sensor but the communication link is broken at some point due to networking infrastructure problems, the samples are not received and as a consequence holes appear in the raw database. The value -2 is used in this case for Sensys network.

- *Technical problems*: Sensor deployment due to maintenance in the road pavement, power supply malfunctioning, bad configuration constitute other sources of missing samples, battery replacements among others. The value -2 is also used in these cases.

In order to compensate the errors introduced by missing data we rely in the imputation algorithm described in details in Appendix A. The algorithm recovers speeds in order to reconstruct the travel time as explained in section 2.2. The DTT obtained with the imputed speed will be used later to constitute the historical database from which the prediction algorithm will be devised. In the next section, we describe how to exploit such historical knowledge through clustering and the rest of the paper is devoted to describe the prediction mechanism of the travel time $D\hat{T}T(x_0, x_n, s_j), \forall s_j \in [t_0, t_0 + \Delta_f]$.

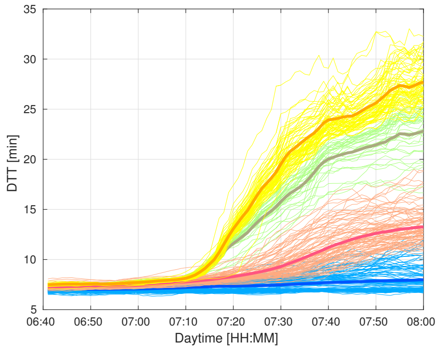
3. Clustering of dynamic travel time data

The aim of clustering is to divide a set of elements, time series of DTT in our case, into subsets or partitions denominated clusters. Elements belonging to the same cluster share natural properties explained by a similarity measurement. The tool has been widely used in applications for pattern recognition, image segmentation and also in traffic [21] even in DTT computation as in [19]. In this case, we consider data clustering as a tool for identification of multiple regimes in traffic.

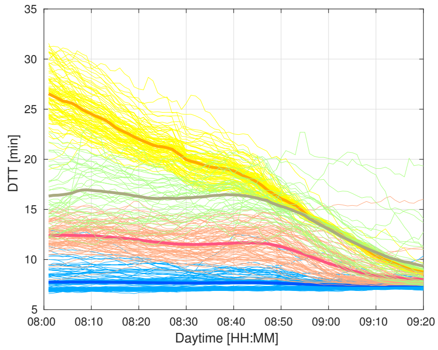
To illustrate the interest in clustering, let consider a set of DTT data computed between September 2015 and May 2016, from $07:00$AM to $07:00$PM everyday. In order to highlight similarities between these data, we show the full set in short intervals of time along several times in the day (See Fig. 4). In addition we plot in bold color the effect of 4 identified centroids for this data. As it is seen the data might be very variable during the day, nevertheless it contains historical similarities, that, in this particular case, distinguish a congested and free flow traffic regimes. In Fig. 4, color represents elements whose euclidean distance is the smallest to the corresponding centroid. We illustrate several moments of the day and its corresponding traffic behaviours. In order to identify these partitions, clustering is indeed a suitable tool [22]. Several aspects must be taken into account for the clustering algorithm, such as: time series to be clustered, type of algorithm, number of partitions. The identification and assessment of these aspects are presented in the following subsections.

3.1. Dynamic Clustering

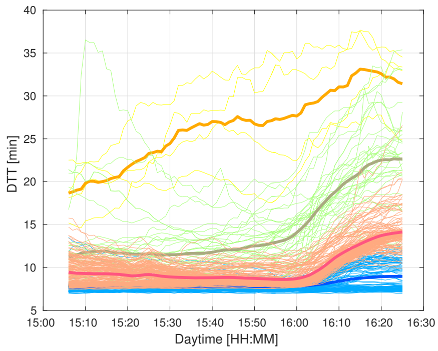
Traffic data exhibits regular patterns day after day, and the time window in which these patterns are identified constitute an important aspect within the clustering technique. There is a clear distinction between working days and weekends for instance. One can think clustering per day is enough. Before providing details on the clustering algorithm used in this approach, let us consider Fig. 5 where full days time series have been used. A similarity index is given by the smallest Euclidean distance between the day in solid line and each one of the days in dashed lines. In the case of Fig. 5, it establishes high similarity (minimum distance) between the day in solid blue and the day in dashed line marked as μ_1. We can indeed note a strong similarity almost all the day except for the time window $03:45$PM to $05:00$PM. By reducing the analysis window to this time period, the day of interest is rather assigned to the dashed day marked as μ_3. Since the knowledge gained from the clusters will be used for prediction purpose, we promote dynamic clustering where clustering is achieved using a moving time window.



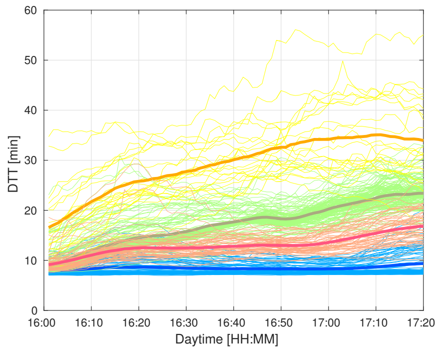
(a) Morning appearing congestion



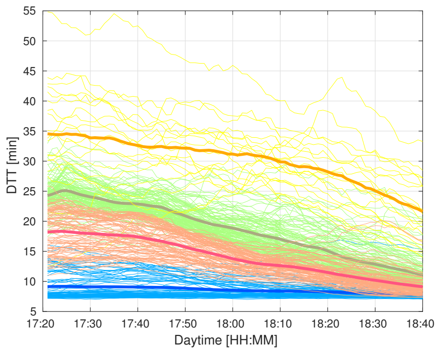
(b) Morning vanishing congestion



(c) Mid-afternoon appearing congestion



(d) Afternoon congestion



(e) Evening vanishing congestion

Figure 4: DTT time series for an historical data set. In color, time signals representing centroids for particular groups. Along the day 4 fixed clusters are identified describing regimes of the traffic behaviour such as free-flow conditions, middle level congestions and strong congestions

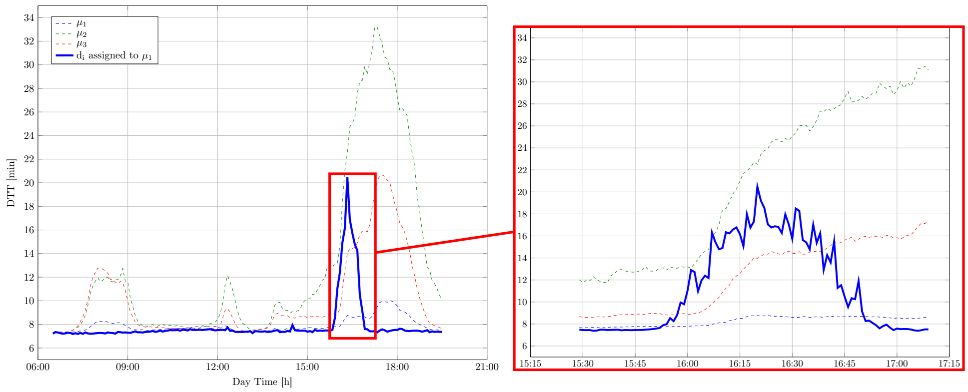


Figure 5: Full days clustering performance vs dynamic clustering.

3.2. Clustering techniques

Several techniques have been developed in order to extract features from data, in general all of them identify the underlying structure in data, perform natural classification and in addition serve as a tool to compress information [23]. In our case we aim to identify determined traffic regimes from time series by compressing multiple historical realizations of variable in partitions. Clustering time series seeks similarity between fluctuations of a variable in time, 4 main categories are commonly applied in time series classification: hard partitioning, connectivity based, distribution based and density model based [24].

Partition based methods are oriented to identify disjoint partitions of the elements. $K-$means [25] and $K-$medioids [26] are the most representative methods. Connectivity methods seek to establish relationships of each member with a subset of elements, i.e $K-$ nearest neighbor finds homogeneous partitions of size K for a given data set [27]. Hierarchical clustering is another technique in this group that creates agglomerations with hierarchy of elements that share some similarity [28].

Among the multiples alternatives we seek specific features for the clustering. Since the aim is to identify traffic regimes we are particularly interested in cluster algorithms that are able to provide strict partitioning, it means an element must belong just to a single cluster, this requirement makes obsolete the use of fuzzy or overlapping clustering techniques. It is also desired to support independence between the partitions since later on we aim to extract statistical properties of each group. Strategies like hierarchical clustering may perform better in terms of separation but the existence of parental relationships between sub groups generates dependencies we try to avoid.

Besides, due to the nature of big data collected historically it is also commendable to represent with minimum information the available partitions if possible a certain unique characteristic. Finally, we aim to apply the clustering to time series by preserving the identity of each time series, since it is desirable to assign a single element to a specific partition. The aforementioned conditions lead to a reduced group of techniques among them: Gaussian Mixture Expectation Maximization (GMEM), $K-$ means/medioids or more sophisticated techniques as explained in [24]. In this work we focus on the K-means algorithm since it is a technique with low time complexity while being highly scalable with large data sets [23, 29].

3.3. K-means algorithm

K-means algorithm is a partition type clustering algorithm that creates $\mathcal{K}$ clusters, $\mathcal{K}$ being a positive input parameter. In our case, given a set of time series with travel times $\mathcal{S} = \{DTT_1, \ldots, DTT_n\}$, the objective is to assign each DTT_i to a cluster C_j such that $C_i \cap C_j = \emptyset$, $C_i \neq \emptyset$ and $\cup_{j=1}^{\mathcal{K}} C_j = \mathcal{S}$. The partitions are obtained through a sequential algorithm that creates centroids which minimize the following criterion [25],

$$\underset{j}{argmin} \sum_{j=1}^{\mathcal{K}} \sum_{DTT_i \in j} ||DTT_i - \mu_j||_2^2. \tag{7}$$

The notion of distance between elements of the set $\mathcal{S}$ is characterized in this case with the euclidean norm. Here μ_j represents the centroid of each one of the clusters that are being updated sequentially, this process is illustrated in Fig. 6. In general the problem of clustering is an

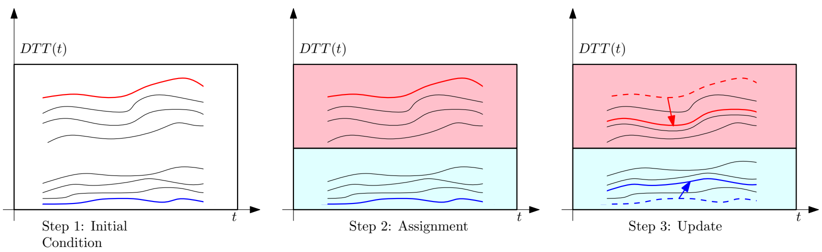


Figure 6: Separation stages in K-means: (a)- Given a number of clusters, the corresponding centroids are selected randomly among the elements of $\mathcal{S}$. (b)- then the remaining elements are assigned to the clusters according to their distance to the centroids. (c)- the process is iterated until a stable partitioning is obtained.

NP-hard problem and the algorithm may produce local minimum solutions [30]. Therefore, the convergence of the algorithm is finished either by a finite number of iterations or when the centroid achieves a minimization for (7).

The *initial condition* as well as the *number of clusters* represent important parameters, so we explain the selection of each one:

1. *Definition of number of clusters*: There exists multiple techniques to select $\mathcal{K}$, we consider the one based on the work developed in [31]. In this case the optimal number of clusters for a data set is given by:

$$\mathcal{K}^* = \underset{\mathcal{K} \in \{2,3,\ldots,\mathcal{K}_{max}\}}{argmin} \quad f(\mathcal{K}). \tag{8}$$

The aim of the problem is to minimize a determined function f which will depend on the *total distortion* for a fixed number of partitions. The idea behind is that if the measured distortion decreases for $\mathcal{K}$ clusters the optimal value is reached. For a fixed value $\mathcal{K}$ the *total distortion* is defined as

$$D_{\mathcal{K}} = \sum_{j=1}^{\mathcal{K}} \sum_{DTT_i \in C_j} ||DTT_i - \mu_j||_2. \tag{9}$$

(9) represents the dispersion within the cluster by adding errors between the elements and the centroids. In this case, the euclidean norm is applied since distortion is aimed to be

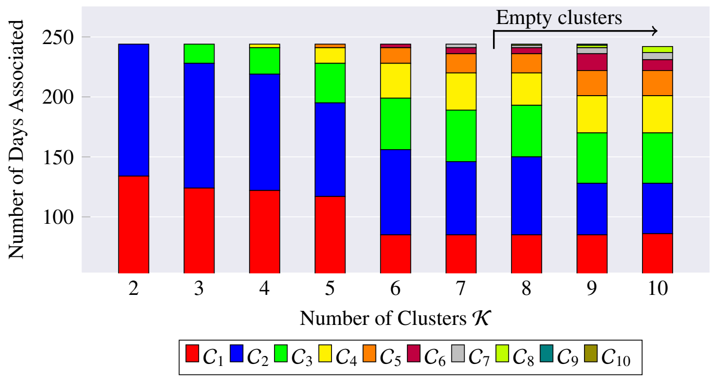


Figure 7: Number of days assigned to cluster C_i at 17h00.

explained with the same similarity measurement as in the clustering algorithm. The role of $f(\mathcal{K})$ is to explain the total distortion $D_{\mathcal{K}}$ of different values of $\mathcal{K}$. In this sense, from [31]:

$$f(\mathcal{K}) = \begin{cases} 1 & \text{if} \quad \mathcal{K} = 1 \\ \frac{D_{\mathcal{K}}}{\alpha_{\mathcal{K}} D_{\mathcal{K}-1}} & \text{if} \quad D_{\mathcal{K}-1} \neq 0 \quad \forall \quad \mathcal{K} > 1 \\ 1 & \text{if} \quad D_{\mathcal{K}-1} = 0 \quad \forall \quad \mathcal{K} > 1 \end{cases} \tag{10a}$$

$$\text{where} \quad \alpha_{\mathcal{K}} = \begin{cases} 1 - \frac{3}{4N_d} & \text{if} \quad \mathcal{K} = 2, N_d > 1 \\ \alpha_{\mathcal{K}-1} + (1 - \alpha_{\mathcal{K}-1})/6 & \text{if} \quad \mathcal{K} > 2, N_d > 1. \end{cases} \tag{10b}$$

In (10a) the term $\alpha_{\mathcal{K}} D_{\mathcal{K}-1}$ is an estimation of $D_{\mathcal{K}}$, so once $D_{\mathcal{K}}$ decreases with respect to an estimated value it means the cluster reached less total distortion. The term $\alpha_{\mathcal{K}}$ is derived under the basis that the set of time series of length N_d is uniformly distributed. (See. [32]). Now let us find the optimal number of partitions by executing a test of clustering for a given historical data set. For this purpose, we consider a set of 244 days of data, containing the DTT from Meylan to Rondeau [16]. A sliding window from 07 : 00AM to 07 : 00PM selects the interval of time in historical data to be clustered, the length of the sliding window is 90min. We proceed in steps of 1min. Each time data is clustered up to 10 clusters as depicted in Fig. 7. We can note that empty clusters start to appear from $\mathcal{K} > 7$. Therefore to find the optimal number $\mathcal{K}^*$ we solve (8) by setting $\mathcal{K}_{max} = 7$. Now, let check the correlation between the optimal number of clusters and the traffic regime given by the DTT. Fig. 8 depicts the distribution (median value and dispersion) of optimal number of clusters $\mathcal{K}$ with respect to the average DTT. . We can highlight the increasing number of partitions for congested periods, however no predominant values can be preselected without incurring in errors. For the rest of the analysis, $\mathcal{K}^*$ varies according to the solution of (8) for an specific time of the day. It is important to remark that in the real time tool [16] the clustering is also performed in a dynamic way each time a prediction is launched. (See section 3.1)

2. *Initialization of centroids*: The nature of the $\mathcal{K}$-means algorithm demands a random initial condition. In order to avoid empty clusters and to achieve repetition on the clustering process, we follow the approach given in [29]. The initial seed for the first cluster is

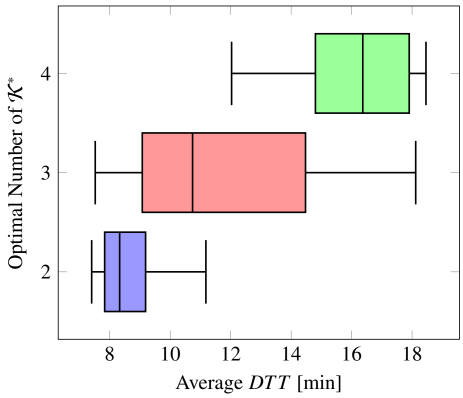


Figure 8: Box plot between the optimal number of clusters $\mathcal{K}^*$ and the historical average DTT (Meylan - Rondeau). The distribution of average DTT shows a direct correlation between the DTT and $\mathcal{K}^*$. However, it is important to remark a high variance of the average DTT in relation with a particular value of $\mathcal{K}^*$. Due to this effect, the value $\mathcal{K}^*$ is recovered from the solution of (8)

chosen randomly according to a uniform distribution, the following ones are chosen with a probability proportional to the distance between the new point and the previous selected centroid. This modified algorithm of $\mathcal{K}$-means is shown to have faster convergence. In order to achieve similarity in the results, the clustering process is replicated a fixed amount of times. The iteration taken into account is the one that minimizes the total distortion (9). Although the nature of the algorithm provides local mínima, experimentally it was found that 10 replicates of the clustering are good enough to find same cluster distributions.

In order to visualize one example of this dynamic behaviour Fig. 9 depicts the clustering obtained for the time window 6:15AM to 7:45AM. The three centroids are depicted in color lines and the area around them corresponds to the subspace occupied by the elements associated to the corresponding cluster.

4. Prediction Algorithm

Let $y_h(k)$ be the DTT at time k and day h. We aim to predict the DTT $\hat{y}_d(k)$ for the current day d and $k > k_0$, k_0 standing for the last time for which DTT can be computed using the available measurements, given $y_d(k_0)$ and $y_h(k)$, $h \neq d$, $k \in I = [k_0 - \Delta_p + 1, k_0 + \Delta_f]$. Here, Δ_p and Δ_f denote the past and the future horizons, respectively.

Two situations may be considered in the prediction scenario. The first one is given the advantage of the current information and potential future information obtained from each cluster an individual prediction can be obtained based on single cluster information like the trend of the centroid, and the value of DTT for this particular cluster in the future. The effect of this strategy has been previously studied in [6]. However, the hard selection of one single prediction is a risky choice due to the the limited information of DTT up to the current time and the limited knowledge of its the future. A second situation emerges when considering the fusion of multiple individual cluster based predictions, this possibility opens the idea to reduce the error for predictions where the cluster assignment in the interval $I_{\Delta_p} = [k_0 - \Delta_p, k_0]$ provides close similarity to multiple clusters. Therefore it is needed to define a proper similarity in I_{Δ_p} and to use this information for

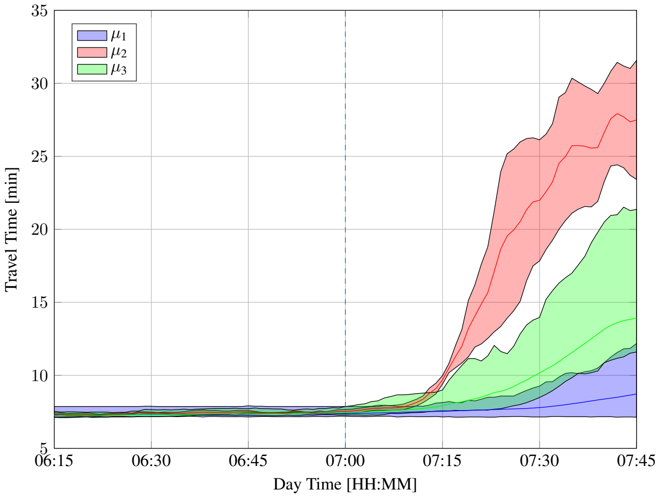


Figure 9: Example of clustered historical DTT data for a time window of 90min centered around $k_0 = 07:00$AM.

mixing mono-cluster based predictions. We explore in the following subsections the derivation of individual predictions and then the strategy used to obtain the final prediction.

4.1. Mono-cluster based prediction

As stated in Section 3, historical data are organized in clusters. Each cluster C_q is defined by its centroid $\mu_q(k)$ and the corresponding forward finite difference $\Delta\mu_q(k) = \mu_q(k+1) - \mu_q(k)$. In other words, a cluster is assumed to exhibit a similarity on both the level and the trend. Therefore, claiming that DTT for the current day belongs to cluster C_q yields:

$$y_d(k) = \mu_q(k) + w_{d,q}(k) \tag{11}$$
$$\Delta y_d(k) = \Delta\mu_q(k) + v_{d,q}(k) \tag{12}$$

for any $k \in I$.
Assumptions:

- $E[w_{d,q}(k)] = 0$ and $E[w^2_{d,q}(k)] = R_q(k)$
- $E[v_{d,q}(k)] = 0$ and $E[v^2_{d,q}(k)] = V_q(k)$

The variances $V_q(k)$ and $R_q(k)$ can be computed from the cluster as follows:

$$R_q(k) = \frac{1}{|C_q| - 1} \sum_{h \in C_q} (y_h(k) - \mu_q(k))^2,$$

$$V_q(k) = \frac{1}{|C_q| - 1} \sum_{h \in C_q} (\Delta y_h(k) - \Delta\mu_q(k))^2,$$

where $|C_q|$ stands for the cardinality of the cluster.

From similarity on the trend in (12), given a prediction $\hat{y}_d(k|k)$ with an error variance $\hat{P}_d(k|k)$, the one step-ahead prediction is given by

$$y_{d,q}(k+1|k) = \hat{y}_d(k|k) + \Delta\mu_q(k). \tag{13}$$

with the error variance:

$$P_{d,q}(k+1|k) = \hat{P}_d(k|k) + V_q(k). \tag{14}$$

On the other hand, similarity the level gives the following predictor:

$$\bar{y}_{d,q}(k+1) = \mu_q(k+1) \tag{15}$$

with error variance $R_q(k+1)$.
The trend-based predictor is a mixture of historical data, by means of the cluster centroid's trend, and current data (initial value of the predictor) while the level-based predictor is only based on the historical level. The objective is then to find a linear combination of these two predictors in order to minimize the variance of the prediction error:

$$\begin{aligned} &\min_{K_q(k+1)} E[(y_d(k+1) - \hat{y}_{d,q}(k+1|k+1))^2] \qquad (16)\\ \text{s.t.} \quad &\hat{y}_{d,q}(k+1|k+1) = (1 - K_q(k+1))y_{d,q}(k+1|k) + K_q(k+1)\bar{y}_{d,q}(k+1) \end{aligned}$$

The solution of this optimization problem is given by [33]:

$$K_q(k+1) = \frac{1}{1 + \frac{R_q(k+1)}{\hat{P}_{d,q}(k|k)+V_q(k)}} \tag{17}$$

One can note that if the dispersion around the centroid is small enough ($R_q(k+1)$ very small) the level-based predictor is preponderant. The reverse is true if the trend coherence is stronger (smaller value of $V_q(k)$. In addition, the predictor

$$\hat{y}_{d,q}(k+1|k+1) = (1 - K_q(k+1))y_{d,q}(k+1|k) + K_q(k+1)\bar{y}_{d,q}(k+1) \tag{18}$$

with a gain given by (17) has error variance

$$\hat{P}_{d,q}(k+1|k+1) = \frac{R_q(k+1)P_{d,q}(k+1|k)}{R_q(k+1) + P_{d,q}(k+1|k)}. \tag{19}$$

Equations (13), (14),(15), (17), (18) and (19) constitute the so-called Clustered Kalman filter.

4.2. Multi-cluster based prediction

Sometimes the performance of the clustering algorithms leads to create partitions that may not characterize a forecast in the future (See Fig. 9 where overlapping of clusters can be noticed before 07 : 15AM). The main objective by introducing a fusion algorithm is to combine forecasts of various clusters which might improve its performance as it is discussed in [10], particularly during peak times. We consider the *temporal similarity measurement (TSM)* between the cluster q and the current day d as,

$$S_{d,q}(k) = \sum_{j=k-\Delta_p+1}^{k} (y_d(j) - \mu_q(j))^2 e^{-\lambda(k-j)} + \gamma(\Delta y_d(j) - \Delta\mu_q(j))^2 e^{-\lambda(k-j)} \tag{20}$$

where the exponential term introduces a forgetting factor λ that gives more importance to recent observed measurements. The objective of (20) is to provide a measurement of closeness between the current day d and the particular cluster. Furthermore, we aim to find weights at the prediction launch time k_0 that correlate in a proper way the current day data and the clustered historical data. For that we introduce, the *cluster weight similarity* given by

$$U(k) = \begin{bmatrix} e^{-\zeta S_1(k)} & \dots & 0 \\ 0 & e^{-\zeta S_2(k)} & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & e^{-\zeta S_q(k)} \end{bmatrix}.$$

Information given by $U(k)$ allow us to fuse data based on observed measurements. For this purpose, let us define by $\hat{y}_d(k|k)$ the combined prediction from $\hat{y}_{d,q}(k|k)$, $q = 1, 2, \cdots, \mathcal{K}^*$. Our aim is to compute a weighted linear combination of the predictions provided by each cluster. In this case the full mechanism of fusion is depicted in Fig. 10. For the computation of the combined prediction we get:

$$\hat{y}_d(k|k) = \sum_{q=1}^{\mathcal{K}^*} \lambda_q \hat{y}_{d,q}(k|k), \quad \lambda_q = \frac{e^{-\zeta S_q(k_0)}}{\sum_{q=1}^{\mathcal{K}} e^{-\zeta S_q(k_0)}}. \tag{21}$$

The overall process is depicted in Fig. 10. It is a two-steps process:

- All of the cluster-based predictors are fed locally at time $k + j$ in order to perform cluster-based predictions according to the subsection 4.1.
- The cluster-based predictions are fused according to (21).

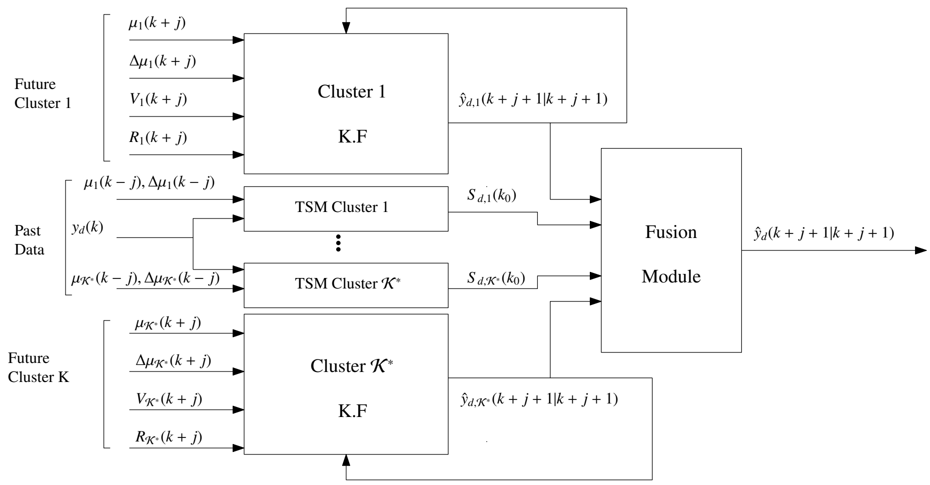


Figure 10: Block diagram of the prediction scheme.

5. Experimental evaluation

In the previous sections, we have described the data acquisition and processing procedures allowing to constitute a reliable database that will be used for prediction purpose. In this section, we evaluate the prediction scheme proposed in the previous section.

The experimental data used for the test is obtained from the Grenoble Traffic Lab (GTL) (See [16]). The network consists of 135 magnetometer sensors distributed along 10.5 Km connecting the highway A31 (north-west) to A480 (south) in Grenoble France. The network offers 68 possible locations for collecting speeds in fast and slow lanes. In this paper, we consider the scenario of a car entering the highway at Meylan and leaving at Rondeau [1]. Predictions are performed between 07 : 00AM and 07 : 00PM time in which it is found peaks of congestion. The system collects information from sensors in real time each time step k, after proceeding with the data imputation and aggregation steps as indicated in 2.1 and Appendix A.1 . The aim is to construct the predicted travel time as defined in (3). Based on the real measured speeds at each collection point in the interval, the DTT is constructed during the past days and for the current day according to the available measurements. DTT varies from 7min in free flow conditions to 40min in congested cases.

For the tests, we set the parameters λ , γ and ζ to some fixed values. The parameter λ is the amount of past information to be considered for the launched prediction, we fix $\lambda = 0.5$ in order to correlate at most the last 15 minutes of traffic regime which have shown to be highly correlated with current information [3]. γ is chosen such that the following relation is preserved:

$$\frac{\sum_{k-\Delta_p+1}^{t}\left(y_d(j)-\mu_q(j)\right)^2}{\sum_{k-\Delta_p+1}^{t} y_d(j)^2} = \gamma \frac{\sum_{k-\Delta_p+1}^{t}\left(\Delta y_d(j)-\Delta\mu_q(j)\right)^2}{\sum_{k-\Delta_p+1}^{t} \Delta y_d(j)^2}$$

The idea with the term is to normalize the amount of energy of the error between the day and the cluster with the same amount of energy given by its corresponding derivatives, this aims to consider with same importance the information given by the trend and the one given by the level. Finally we set the value of $\zeta = 0.5$. This value explains the degree of selection of the term $S_{d,q}(k_0)$, we measured the variance of multiple clusters and by examining the dispersion we fix the value such that the term $e^{-\zeta S_q(k_0)} \approx 0$ for variances larger than 4min around the centroid of the corresponding cluster.

5.1. Individual Predictions Assessment

First, we assess individual predictions for a fixed amount of clusters $\mathcal{K}^*$ as shown in Fig. 11. Here we aim to show behaviours at specific moments of the day to verify the performance of the prediction algorithm. Each one of the figures 11a,11c,11e depicts the prediction given by our approach, the data collected during the corresponding day and the centroids of the clusters considered for each case. The performance of the method is visually verified in Fig. 11a where there is a clear distinction between the several regimes and hence the prediction provides good results, an proof of this fact is also checked in the weight assigned in Fig. 11b. As a second case, Fig. 11c shows a particular initial morning congestion that regularly starts around 07 : 20AM. As seen in Fig. 11c the increasing behaviour of the congestion is captured by the TSM and hence by the forecast algorithm. The weights λ_q in this case select the three closest centroids indexed

[1]Both places correspond to the beginning and the end of the 10.5 Km of installed sensors

as $(2,4,6)$ as seen in Fig. 11d. Finally 11e shows a particular mid afternoon congestion. It is important to remark that in this case the similarity term that provides the good selection of the cluster is strongly supported by the trend of the cluster.

Even though the amount of error is high in this last case the behaviour of this prediction can be improved along the day. Consider for instance the same prediction as in Fig. 11e launched at different times (See Figures 12a to 12f). In this case, as the prediction is computed at different hours it is noticeable the reduction of error due to a higher similarity with respect to the purple cluster as time advances during the day. A more precise comparison is given in terms of the Absolute Percentage Error (22) in Fig. 12f.

5.2. Crossvalidation test

For a quantitative test, we constrain the set of historical data to 244 days collected from September 2015 to May 2016 [34], [35]. We will evaluate the prediction scheme for different departure times in two periods: morning from 07 : 00AM to 10 : 00AM and afternoon from 04 : 00PM to 07 : 00PM. In what follows, we consider k_0 as the current time and $k_0 + \Delta T$ as the desired departure time in future with $\Delta T \in \{5, 10, 15, 20, 25\}$.

According to the results of Section 3, for all the tests, the number of cluster is adapted according to (8). In order to assess the performance of the proposed method, we resort to the APE as performance index. For a particular departure at some desired future horizon $1 \leq \Delta_f$ the APE is given by:

$$APE(k_0 + h, d) = 100 \frac{|\widehat{y}_d(k_0 + h) - y_d(k_0 + h)|}{y_d(k_0 + h)}, \tag{22}$$

In order to study the behaviour of the strategy, the fixed amount of historical data will be selected, then a leave one out cross validation test is performed over the data. The element out of the set is the candidate day d for performing the prediction. We perform the prediction at some specific time k_0 of the day, the APE index in (22) is computed for each one of the forecasts, then the process is repeated along the day by advancing in steps of 1min. The results given hereafter are averaged over k_0 and d.

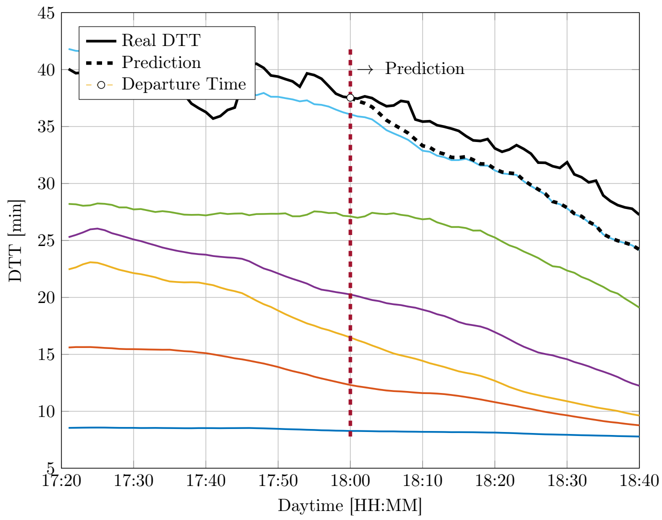
5.3. Evaluation of the mono-cluster prediction scheme

In this subsection, we first evaluate the prediction scheme derived when the cluster to which the time series of interest belongs is well known.

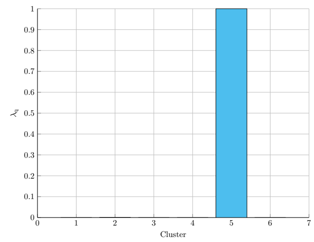
In Fig. 13, the empirical cumulative distribution function (ECDF) of the APE is depicted for different horizons and different periods of time during the day. We compare the performance with that of methods considering historical mean as prediction. For historical mean, whatever the horizons the ECDF is similar. For the proposed method, we can note that the performance decreases when the horizon of prediction increases. The gap of performance observed between horizon 5min and horizon 25min is slightly bigger in the afternoon than in the morning. Whatever the horizon, the proposed scheme outperforms the historical mean.

80% of time, the proposed method allows to get predictions error less than 7% for horizon 5min and less than 10% for horizon 25min in the morning case. As for the afternoon case 80% of the times the error is less than 11% for horizon 5min and less than 18% for horizon 25min. (See. table 1). In terms of time, this means for horizon 5min the error is guaranteed to be less than 42sec in free flow [2] and less than 4min24sec during congestion. For horizon 25min, we get

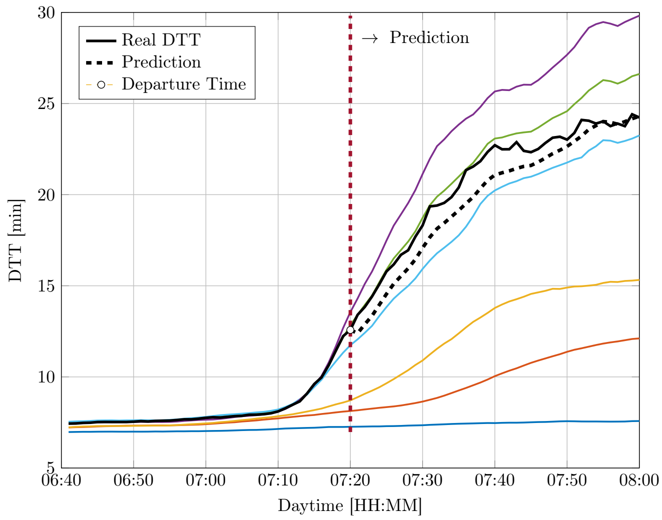
[2] Travel time is considered here as 7min for a free flow condition and 40min for a congested case



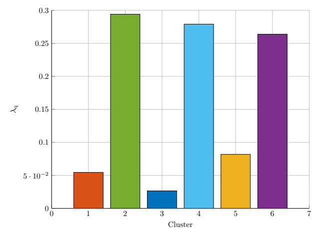
(a) Prediction weights - 22nd April,2016



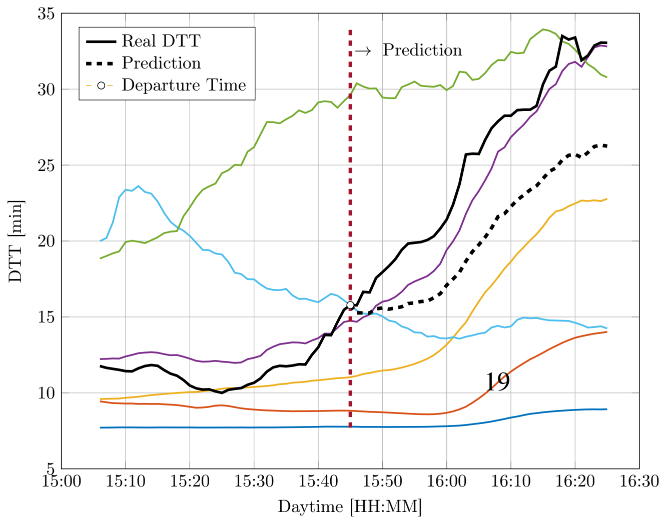
(b) λ_q - 22nd April,2016



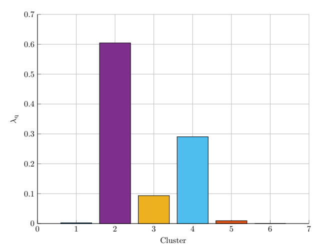
(c) Prediction Day - 17th February,2016



(d) λ_q - 17th February,2016

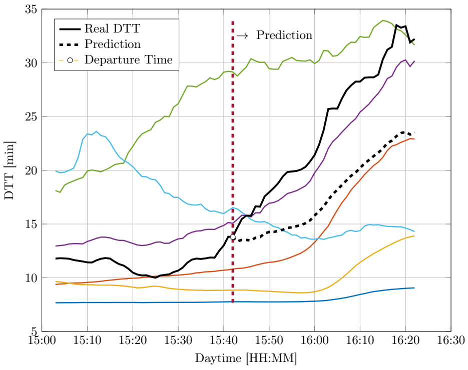


(e) Prediction Day - 10th October,2015

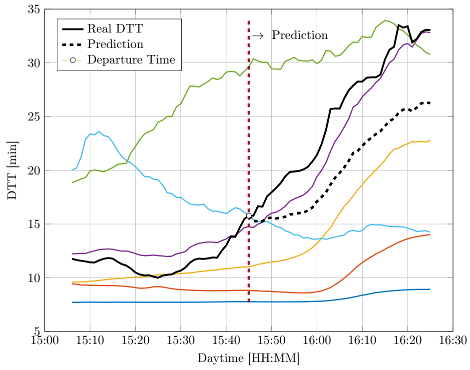


(f) λ_q - 10th October,2015

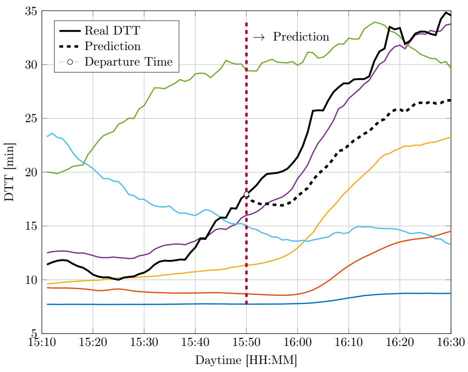
Figure 11: Individual predictions in several days at multiple times of the day. In color we visualize the centroids of the cluster. The assigned weights corresponds in color to the cluster color depicted in the left side figures



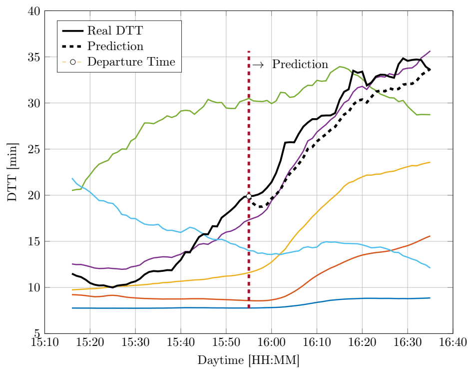
(a) Launch Time: 15h41



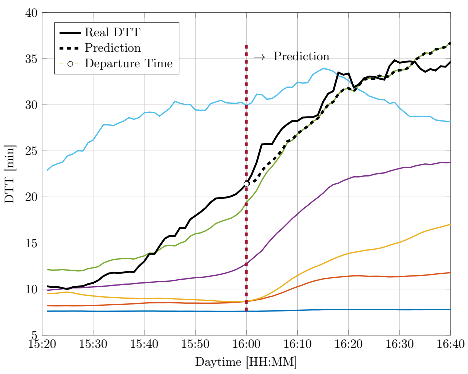
(b) Launch Time: 15h45



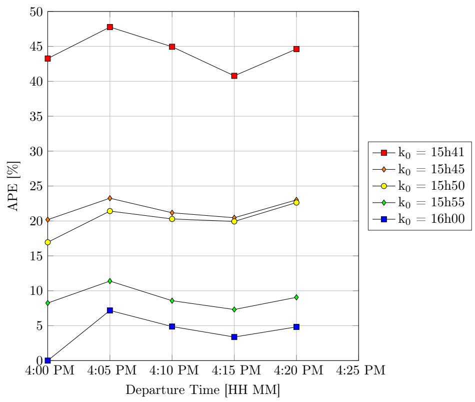
(c) Launch Time: 15h50



(d) Launch Time: 15h55

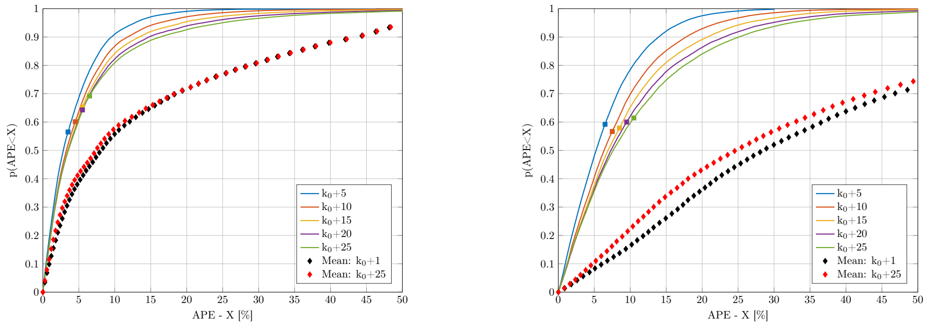


(e) Launch Time: 16h00



(f) Error Evolution

Figure 12: Analysis of same day predictions at different times during the day



(a) APE examined between 7AM and 10AM (b) APE examined between 4PM and 7PM

Figure 13: ECDF plot of the prediction performance for the mono-cluster case, while assuming that the cluster is well known

	$p < X$	$\Delta = 5$	$\Delta = 10$	$\Delta = 15$	$\Delta = 20$	$\Delta = 25$
Morning	90%	9.74%	11.73%	13.45%	14.74%	16.19%
	80%	6.84%	7.96%	8.62%	9.159%	9.62%
Afternoon	90%	13.84%	17.70%	20.54%	22.94%	24.98%
	80%	10.24%	12.86%	14.54%	15.98%	17.48%

Table 1: APE guaranteed error based for multiple horizons. Mono-cluster case

an error less than 1min15s in free flow and less than 7min in congestion. We can notice than the proposed method, despite its simplicity, is efficient enough. However, we have assumed the cluster is perfectly known. Such assumption is hard to fulfill. So in next section, we evaluate the multi-cluster strategy with fusion policy derived in previous section.

5.4. *Multi-cluster fusion strategy*

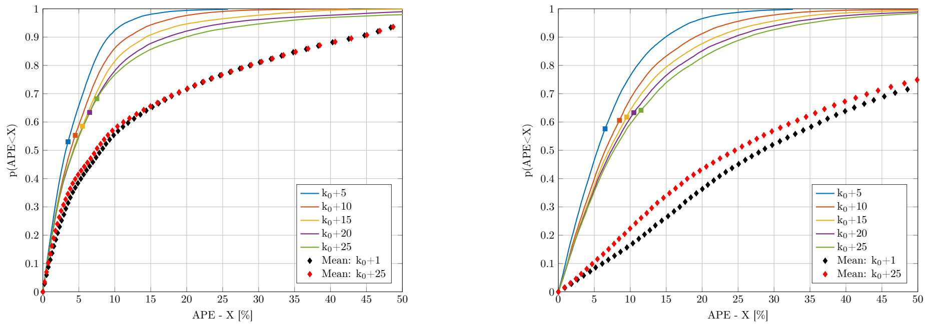
Fig. 14 depicts the ECDF for the multi-cluster fusion strategy. In this case, we have no *a priori* knowledge on the cluster to which the time series belongs. Therefore we are far from the ideal case. However, we can note that the method still perform better that the historical mean.

In the morning case, for the horizon of 5min the guaranteed error is 6.93% (meaning 29sec in free flow and 2min46sec during congestion) while for horizon of 25min we get 11.42% (meaning 47sec in free flow and 4min34sec during congestion). As for the afternoon case the error augments to 10.93% for horizon of 5min (meaning 46sec in free flow and 4min22sec during congestion) and 18.20% for horizon of 25min. For short term predictions the performance is still

	$p < X$	$\Delta = 5$	$\Delta = 10$	$\Delta = 15$	$\Delta = 20$	$\Delta = 25$
Morning	90%	9.04%	11.82%	14.19%	17.26%	19.59%
	80%	6.93%	8.35%	9.57%	10.62%	11.42%
Afternoon	90%	14.86%	18.97%	21.89%	24.35%	26.24%
	80%	10.93%	13.41%	15.27%	16.79%	18.20%

Table 2: APE guaranteed error based for multiple horizons. Multi-cluster case

good while for longer term predictions there is a significant loss of performance compared to the ideal case of the previous section. (See table 2)



(a) APE examined between 7AM and 10AM

(b) APE examined between 4PM and 7PM

Figure 14: ECDF plot of the prediction performance for the multi-cluster fusion strategy

5.5. Software development

All the steps of data acquisition, processing, clustering and prediction have been implemented in a web platform that provides to users current status of the network and prediction of the DTT. Fig. 15 depicts the User Interface of the platform. The user can provide a single origin among 8 possible entries in the Rocade sud of Grenoble city and 10 possible exits and launch a forecast 45min ahead, when the query is performed in past days the predicted information is compared with real information. At each time step the real time application executes the prediction for

Figure 15: Prediction module interface in the Grenoble Traffic Lab.

the DTT in steps of 1min. The results of the individual predictions can be seen directly at http://gtl.inrialpes.fr/predictions. An example of a single forecast is provided in Fig. 15 and Fig. 16, the blue curve corresponds to the DTT computed from measured speeds and the red one corresponds to predicted DTT. In addition to the forecast and measured signals (Fig. 15) a recommendation for optimal departure time is also provided (Fig. 16).

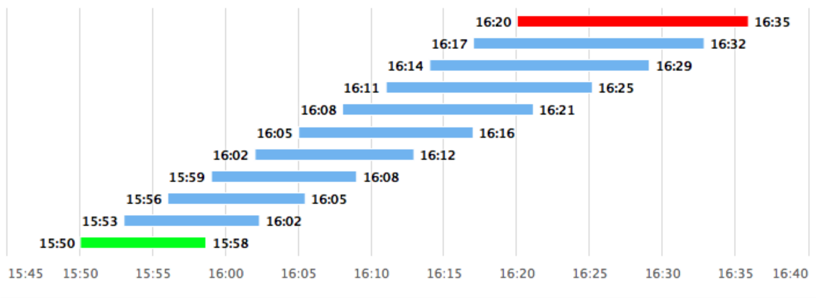


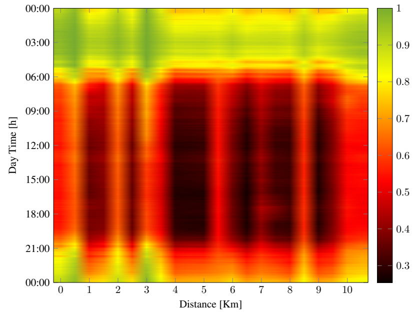
Figure 16: Output bar recommendation from the prediction system. Once the prediction is selected the user receives optimal recommendations for departure time based on the minimum predicted travel time.

6. Conclusions

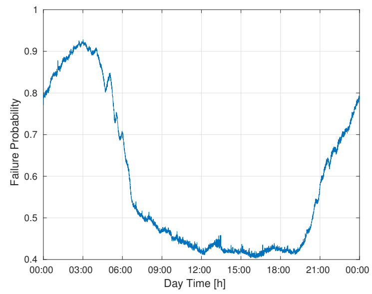
We have addressed the problem of dynamic travel time (DTT) forecasting by making use of speed measurements provided by sensors located at the node levels of the Grenoble's highway. The proposed methodology performs the prediction based on the fusion of individual forecasts obtained from clustered time series. The study is focused on a real time application where new measurements of data can be received in short periods of time ([15sec $\sim$ 1 min] in the presented case), therefore the main benefit of this work is to perform low computational workload to obtain online DTT forecasts under real time traffic conditions. For the sake of understanding, each one of the steps in the process is explained in details and performance indexes are shown. The efficacy of the prediction methodology is studied in terms of the prediction error along the future step horizon under a cross validation schema. It is found a growing behaviour of the error with the prediction horizon. The proposed scheme outperforms the historical mean and provides interesting results when *a priori* knowledge on the cluster to which belongs the time series to be predicted is available. Future directions for this work may include the analysis of fusion mechanisms in which the weights are controlled by alternative and more possibly decided parameters, the study of statistical merge between model based approaches and the presented strategy and further studies on the effect of the horizon in relationship with the clustered time series and the forecasting strategy.

Acknowledgement

Authors would like to thank to Remi Piotaix who developed visualization tools for the website interface. This work has been funded by the EU FP7 project SPEEDD (619435). This work has also received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement N° 694209).



(a) Space-time distribution of missing samples - Relative frequency of missing samples with respect to the full historical data set



(b) Network missing samples - Relative frequency of missing samples with respect to the full space data expected at each sample time

Figure A.17: Missing data evaluation of the GTL from September 1^{st}, 2015 to March 31^{st}, 2016. The figure on the left panel illustrates the distribution of missing samples analyzed from an historical view point. At each sample time and specific location the amount of missing samples is counted within the historical data set. The right panel displays the distribution of missing samples along the day, as a remark the peaks of missing data are concentrated out of the peak hours which benefits the algorithm in this study.

Appendix A. Imputation of missing values in traffic speed data

Missing data is a natural phenomenon in traffic data collection. We analyze in Fig. A.17 the relative frequency of missing samples in space and time for a historical data set of speeds recorded between September 1^{st}, 2015 and March 31^{st}, 2016. From these results, we can note high values of missing data relative frequency in non peak hours 12 : 00AM-06 : 00AM and 09 : 00PM-12 : 00AM which reflects the fact that few vehicles are detected by the sensors and no speeds are captured during this intervals of time (acquisition errors). During peak hours, the probability of missing samples varies between 40% and 60%. It is therefore necessary to resort to data imputation in order to replace missing data by their approximations which must be good enough to not impair the travel time prediction.

Appendix A.1. Mechanism of imputation

As stated above, missing velocity samples are assigned to some negative values, let us introduce an indicator function $\mathcal{I}(x_i, t_\tau)$ to mark the valid measured samples, thus $\mathcal{I}(x_i, t_\tau) = 1$ if and only if $v(x_i, t_\tau) > 0$. Each time a missing sample is encountered in the network, i.e. $\mathcal{I}(x_i, t_\tau) = 0$, an efficient strategy is set up in order to replace it by the best approximation according to some type of consistency. Three types of consistency mechanisms can be envisioned: temporal, spatial, and historical.

1. *Temporal consistency*: Based on the assumption that $v(x_i, t_\tau) \approx v(x_i, t_\tau - \delta)$ for some value of $\delta > 0$ the aim is to substitute the missing value at time t_τ by the average value within the time window $[t_{\tau-r}, t_\tau]$
$$\tilde{v}_l(x_i, t_\tau) = \frac{\sum_{l=1}^{r} v(x_i, t_{\tau-l}) \mathcal{I}(x_i, t_{\tau-l})}{\sum_{l=1}^{r} \mathcal{I}(x_i, t_{\tau-l})}. \tag{A.1}$$

2. *Spatial consistency*: Considering the assumption $v(x_i, t_\tau) \approx v(x_j, t_\tau)$ for x_j within a neighborhood $\mathcal{N}_i$ the value is replaced as

$$\tilde{v}_s(x_i, t_\tau) = \frac{\sum_{x_j \in \mathcal{N}_i} v(x_j, t_\tau)\mathcal{I}(x_j, t_\tau)}{\sum_{x_j \in \mathcal{N}_i} \mathcal{I}(x_j, t_\tau)}, \tag{A.2}$$

   where $\mathcal{N}_i$ denotes the neighbours of the location x_i defined as $\mathcal{N}_i = \{x_{i-1}, x_{i+1}\}$ representing the sensor ahead in the following location and the sensor behind in the previous location in the space axis, respectively.

3. *Historical consistency*: We denote $v^{(h)}(x_i, t_\tau)$ as the speed in space-time plane for some day h in the set of past days and $\mathcal{I}^{(h)}(x_i, t_\tau)$ its corresponding valid data indicator. Under the assumption of historical similarities $v^{(h)}(x_i, t_\tau) \approx v^{(d)}(x_i, t_\tau)$ for two different days h and d, the value is imputed as an average of historical values of preselected days, hence

$$\tilde{v}_h(x_i, t_\tau) = \frac{\sum_{h \in \mathcal{H}_d} v^{(h)}(x_i, t_\tau)\mathcal{I}^{(h)}(x_i, t_\tau)}{\sum_{h \in \mathcal{H}_d} \mathcal{I}^{(h)}(x_i, t_\tau)}, \tag{A.3}$$

   where $\mathcal{H}_d$ are groups of historical days, each one corresponding to a day d of the week.

In cases in which the expressions (A.1),(A.2),(A.3) cannot be computed due to zero values in the denominator term the value `NaN` is assigned to the corresponding variable.

Appendix A.1.1. Evaluation of the imputation schemes

In order to assess the strategies we examine a day (October 16^{th}, 2015) with no missing data from 07 : 00AM to 07 : 00PM. To emulate data loss, missing samples are generated according to a particular scenario determined by a specific pattern of lost samples in a day, then each imputation mechanism is evaluated according to the following absolute error AE:

$$AE(x_i, t_\tau) = 100\left|\frac{\tilde{v}(x_i, t_\tau) - v(x_i, t_\tau)}{v(x_i, t_\tau)}\right|(1 - \mathcal{I}(x_i, \tau_t)). \tag{A.4}$$

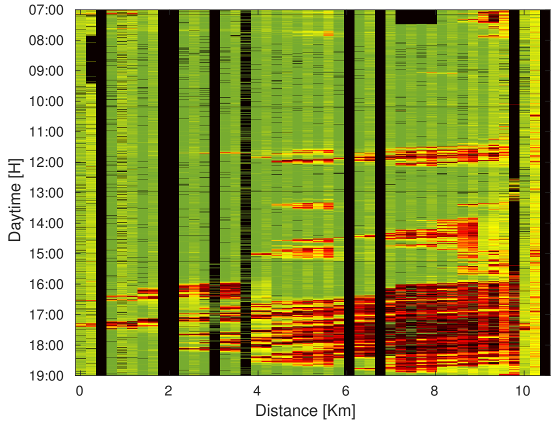
We also evaluate the quality of imputation by introducing the Percentage of Recovered Samples (PRS), that corresponds to the number of samples recovered after applying the strategy with respect to the total missing samples in the set. In this case, a pattern of missing samples with the same distribution as in the day October 7^{th}, 2015 is emulated over the day October 16^{th} in order to evaluate the method in more realistic conditions. The absence of data can be seen in Fig. A.18 and the results for the PRS and AE are detailed in Table A.3. As a main result it can be seen that the method with better performance is the historical due to the amount of information available for the recovery, however it is computationally expensive since each replaced sample should be obtained from the average of historical sets, the quality of the recovery may also vary on the amount of historical days. Experimentally small values of r showed to provide a small AE, however in order to preserve a PRS over 10% and to consider the assumption for the temporal strategy we have selected $r = 4min$.

Temporal or spatial consistency is cheap from a computational point of view since only the current day data is used for imputation in contrast to the historical one. On the other hand, the amount of error in the case of spatial and historical remain quite close. Finally, we adopt the following imputation sequence (detailed in Algorithm 1 in Fig. A.19) which seeks to complete gradually the day by first recovering information from smaller components in the network (sensors) and then up scaling to bigger aggregations (neighbours of sensors, full network):

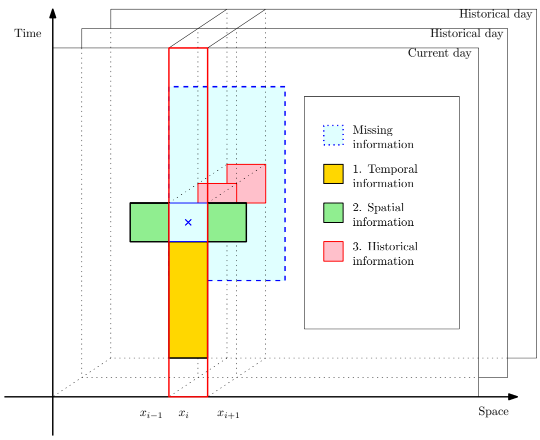
Method	PRS (%)	AE (%)
Temporal	10.6	29.4
Spatial	83.1	25.7
Historical	88.2	22.0

Table A.3: Percentage of Recovered Samples (PRS) and Absolute Error (AE) for each method applied in the day October 16^{th}, 2015. $r = 4$ is the parameter *temporal* strategy

.



(a) Missing sample distribution- Speed spatio-temporal distribution, in black it is showed the spatio-temporal distribution of missing samples according to a specific day of the week. The distribution of missing samples is particularly found on October 7^{th}, 2015



(b) Sources of information for the imputation algorithm

Figure A.18: Real traffic scenario of missing samples. The figure describes a typical missing data distribution in which information at particular locations is lost during the full day. The sources of imputation to replace the missing data are shown in A.18b

Data: Raw: $v(x_i, t_\tau)$
Result: Imputed value $v(x_i, t_\tau)$
Define parameters: $r = 4\text{min}$
$\mathcal{N}_i = \{x_{i-1}, x_{i+1}\}$
$\mathcal{H}_d = \{h_d^{(1)}, \dots, h_d^{(H)}\}$
Compute $\tilde{v}_l, \tilde{v}_s, \tilde{v}_h$, as (A.1), (A.2) and (A.3)
while $v(x_i, t_\tau) < 0$ **do**
  $v(x_i, t_\tau) \leftarrow \texttt{NaN}$
  $v(x_i, t_\tau) \leftarrow \tilde{v}_s(x_i, t_\tau)$
  if $v(x_i, t_\tau) = \mathtt{NaN}$ **then**
    $v(x_i, t_\tau) \leftarrow \tilde{v}_l(x_i, t_\tau)$
  end
  if $v(x_i, t_\tau) = \mathtt{NaN}$ **then**
    $v(x_i, t_\tau) \leftarrow \tilde{v}_h(x_i, t_\tau)$
  end
end

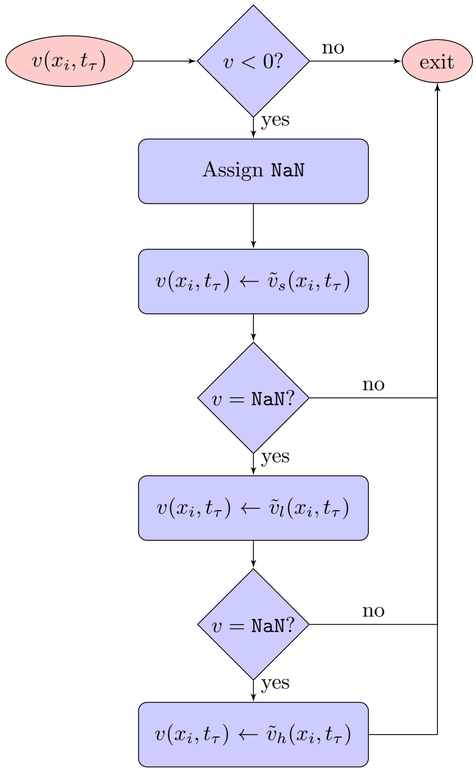


Figure A.19: Algorithm applied for imputation. Left-hand side: Algorithm sequence / Right-hand side: Flow diagram to illustrate data work flow

1. Spatial consistency (since it produces a low AE and it is cheaper than historical, also it could potentially cover a PRS similar to the historical), else
2. Temporal consistency (since it provides the local information and the cost is very cheap with respect to the historical one), else
3. Historical consistency in which the objective is to recover a reduced set of missing samples.

Appendix A.2. Imputation effects on DTT

Data imputation is a required step in order to have a complete full matrix of speed information that allows to compute the travel time. Without this step, some errors are generated in the computation of the travel time or even inconsistencies due to the nature of measurements. In this case we check the performance of Algorithm 1 in terms of reconstruction error on DTT.

After applying Algorithm 1, we obtained the results depicted in Fig. A.20. In this case the DTT is computed with the real speed profile and compared with respect to the values obtained from imputed one. Increasing the proportion of missing samples causes the error to rise by a small amount. In 90% of the cases the error in the DTT does not overpasses 5%. From the panels on scattered data the concentration of error occurs more often in congested time when travel time

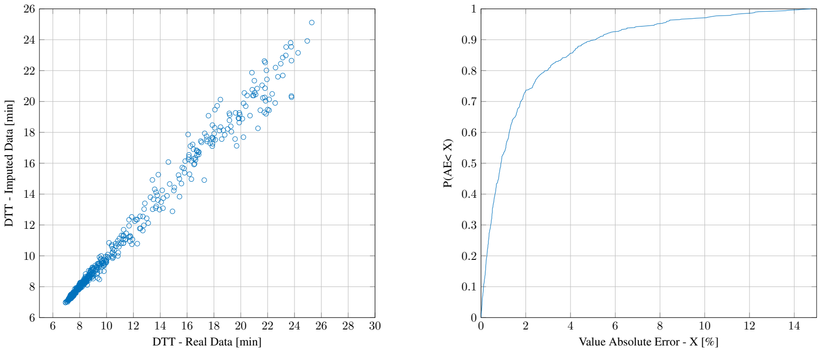


Figure A.20: Left Panel. Scatter plot of actual measured DTT vs estimated DTT after imputation. The test in this case shows the scenario 2 when a regular pattern of missing data is removed. The imputation algorithm is applied over speeds (See Algorithm. 1), and DTT is then computed as (3). Right Panel. Absolute Error Cumulative Distribution Function

is high, however the error incurred can still be considered satisfactory for the estimation of the DTT.

References

[1] E. I. Vlahogianni, J. C. Golias, M. G. Karlaftis, Short term traffic forecasting: Overview of objectives and methods, Transport Reviews 24 (2004) 533–557.

[2] E. I. Vlahogianni, M. G. Karlaftis, J. C. Golias, Short-term traffic forecasting: Where we are and where we're going, Transportation Research Part C: Emerging Technologies 43 (2014) 3–19.

[3] L. L. Ojeda, A. Y. Kibangou, C. Canudas de Wit, Adaptive Kalman Filtering for Multi-Step ahead Traffic Flow Prediction, in: American Control Conference (ACC), Washington, 2013, pp. 4731–4736.

[4] W. Weijermars, E. van Berkum, Analyzing highway flow patterns using cluster analysis, in: Proceedings. 2005 IEEE Intelligent Transportation Systems, 2005., IEEE, 2005, pp. 831–836. doi:10.1109/ITSC.2005.1520157.

[5] R. Chrobok, O. Kaumann, J. Wahle, M. Schreckenberg, Different methods of traffic forecast based on real data, European Journal of Operational Research 155 (2004) 558–568.

[6] L. Ojeda, A. Y. Kibangou, C. Canudas de Wit, Online Dynamic Travel Time Prediction using Speed and Flow Measurements, in: European Control Conference, IEEE, Zurich, 2013, pp. 4045–4050.

[7] C.-j. Wu, T. Schreiter, R. Horowitz, Multiple-clustering ARMAX-based predictor and its application to freeway traffic flow prediction, in: 2014 American Control Conference, IEEE, Portland, 2014, pp. 4397–4403.

[8] N. Wan, G. Gomes, R. Horowitz, Prediction on Travel-Time Distribution for Freeways Using Online Expectation Maximization Algorithm, in: Transportation Research Board 93rd Annual Meeting, Washington, 2014, pp. 14–3221.

[9] F. G. Habtemichael, M. Cetin, Short-term traffic flow rate forecasting based on identifying similar traffic patterns, Transportation Research Part C: Emerging Technologies 66 (2016) 61–78.

[10] Y. Zhang, Y. Liu, Analysis of peak and non-peak traffic forecasts using combined models, Journal of Advanced Transportation 45 (2011) 21–37.

[11] S.-L. Sun, Multi-sensor information fusion white noise filter weighted by scalars based on Kalman predictor, Automatica 40 (2004) 1447–1453.

[12] Y. Zhang, Hourly Traffic Forecasts Using Interacting Multiple Model (IMM) Predictor, IEEE Signal Processing Letters 18 (2011) 607–610.

[13] L. Du, S. Peeta, Y. H. Kim, An adaptive information fusion model to predict the short-term link travel time distribution in dynamic traffic networks, Transportation Research Part B: Methodological 46 (2012) 235–252.

[14] P. Cai, Y. Wang, G. Lu, P. Chen, C. Ding, J. Sun, A spatiotemporal correlative k-nearest neighbor model for short-term traffic multistep forecasting, Transportation Research Part C: Emerging Technologies 62 (2016) 21–34.

[15] A. Ladino, A. Y. Kibangou, H. Fourati, C. Canudas de Wit, Travel time forecasting from clustered time series via optimal fusion strategy, in: European Control Conference 2016, 2016.

[16] C. Canudas de Wit, F. Morbidi, L. L. Ojeda, A. Y. Kibangou, I. Bellicot, P. Bellemain, Grenoble Traffic Lab: An Experimental Platform for Advanced Traffic Monitoring and Forecasting, IEEE Control Systems 35 (2015) 23–39.

[17] M. Papageorgiou, I. Papamichail, A. Messmer, Y. Wang, Traffic Simulation with METANET, 2010, pp. 399–430.

[18] M. Yildirimoglu, N. Geroliminis, Experienced travel time prediction for congested freeways, Transportation Research Part B: Methodological 53 (2013) 45–63.

[19] M. Elhenawy, H. Chen, H. A. Rakha, Dynamic travel time prediction using data clustering and genetic programming, Transportation Research Part C: Emerging Technologies 42 (2014) 82–98.

[20] R. Bajwa, R. Rajagopal, P. Varaiya, R. Kavaler, N. Street, In-Pavement Wireless Sensor Network for Vehicle Classification, in: Proceedings of the 10th ACM/IEEE International Conference on Information Processing in Sensor Networks, Chicago, USA, 2011, pp. 85–96.

[21] M. Montazeri-Gh, A. Fotouhi, Traffic condition recognition using the k-means clustering method, Scientia Iranica 18 (2011) 930–937.

[22] A. Khaleghi, D. Ryabko, J. Mary, P. Preux, Consistent Algorithms for Clustering Time Series, Journal of Machine Learning Research 17 (2016) 1–32.

[23] A. K. Jain, Data clustering: 50 years beyond K-means, Pattern Recognition Letters 31 (2010) 651–666.

[24] D. Xu, Y. Tian, A Comprehensive Survey of Clustering Algorithms, Annals of Data Science 2 (2015) 165–193.

[25] S. Lloyd, Least squares quantization in PCM, IEEE Transactions on Information Theory 28 (1982) 129–137.

[26] H.-S. Park, C.-H. Jun, A simple and fast algorithm for K-medoids clustering, Expert Systems with Applications 36 (2009) 3336–3341.

[27] E. Fix, J. L. Hodges, Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties, International Statistical Review / Revue Internationale de Statistique 57 (1989) 238.

[28] M.-F. Balcan, Y. Liang, P. Gupta, Robust Hierarchical Clustering, Journal of Machine Learning Research 15 (2014) 4011–4051.

[29] D. Arthur, S. Vassilvitskii, k-means++: the advantages of careful seeding, in: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, Association for Computing Machinery, 2007, pp. 1027–1025.

[30] M. Mahajan, P. Nimbhorkar, K. Varadarajan, WALCOM: Algorithms and Computation, volume 5431 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[31] D. T. Pham, S. S. Dimov, C. D. Nguyen, Selection of K in K-means clustering, Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science 219 (2005) 103–119.

[32] D. T. Pham, S. S. Dimov, C. D. Nguyen, An Incremental K-means algorithm, Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science 218 (2004) 783–795.

[33] R. Kalman, A new approach to linear filtering and prediction problems, Journal of Fluids Engineering 82 (1960) 35–45.

[34] R. Piotaix, A. Ladino, Grenoble Traffic Lab - Predictions, 2016. URL: `http://necs.inrialpes.fr/pages/grenoble-traffic-lab/results.php`.

[35] A. Ladino, A. Y. Kibangou, H. Fourati, C. Canudas de Wit, Speed, counts, occupancy dataset (Grenoble Traffic Lab), 2016. URL: `http://dx.doi.org/10.17632/tf483zkcmb.3`. doi:`10.17632/tf483zkcmb.3`.