Data-driven network loading

Nikolaos Tsanakas, Joakim Ekström, David Gundlegård, Johan Olstam and Clas Rydergren

The self-archived postprint version of this journal article is available at Linköping University Institutional Repository (DiVA):
http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-173416

N.B.: When citing this work, cite the original publication.

Original publication available at:
https://doi.org/10.1080/21680566.2020.1847213

Copyright: Hong Kong Society for Transportation Studies
Publisher URL Missing
Data-Driven Network Loading

N. Tsanakas\textsuperscript{a}, J. Ekström\textsuperscript{a}, D. Gundlegård\textsuperscript{a}, J. Olstam\textsuperscript{a,b} and C. Rydergren\textsuperscript{a}

\textsuperscript{a}Department of Science and Technology, Linköping University, Sweden; \textsuperscript{b}Swedish National Road and Transport Research Institute, Linköping, Sweden

**ABSTRACT**

Dynamic Network Loading (DNL) models are typically formulated as a system of differential equations where travel times, densities or any other variable that indicates congestion is endogenous. However, such endogeneities increase the complexity of the Dynamic Traffic Assignment (DTA) problem due to the interdependence of DNL, route choice and demand. In this paper, attempting to exploit the growing accessibility of traffic-related data, we suggest that congestion can be instead captured by exogenous variables, such as travel time observations. We propagate the traffic flow based on an exogenous travel time function, which has a piece-wise linear form. Given piece-wise stationary route flows, the piece-wise linear form of the travel time function allows us to use an efficient event-based modelling structure. Our Data-Driven Network Loading (DDNL) approach is developed in accordance with the theoretical DNL framework ensuring vehicle conservation and FIFO. The first simulation experiment-based results are encouraging, indicating that the DDNL can contribute to improving the efficiency of applications where the monitoring of historical network-wide flows is required.

**Abbreviations:** DDNL–Data Driven Network Loading; DNL–Dynamic Network Loading; DTA–Dynamic Traffic Assignment; ITS–Intelligent Transportation Systems; OD–Origin Destination; TTF–Travel Time Function; LTT–Linear Travel Time; DL–Demand level

**KEYWORDS**

Data-Driven Assignment; Network Loading; Network-Wide Flows

1. **Introduction**

The rapid development of information and communication technologies and extensive research on Intelligent Transportation Systems (ITS) have provided a vast amount of traffic-related data. Alongside the traditional cross-sectional traffic measurements from loop detectors or radar sensors, an alternative type of data is available from probe vehicles, road infrastructure equipped with automatic vehicle identification devices, or from cellular networks. Several studies (Bertsimas et al. 2019; Jenelius and Koutsopoulos 2013; Kong et al. 2012; Krishnakumari et al. 2019; Zheng and Van Zuylen 2013) suggest methods for utilising this cost-effective type of data and provide network-wide travel time estimations. Network-wide travel time information can be critical for various applications such as traffic management. Monitoring travel times reveals the most congested parts of a network, indicating the appropriate interventions to be made for enhanced system efficiency.

Although probe vehicle data has been widely used for generating network-wide link speeds and travel times, such type of data cannot directly provide dynamic traffic volumes or flows. This valuable information, i.e. the number of vehicles that are present...
on a link (traffic volume) or passing by a cross-section (traffic flow) over specific time periods, is an essential input to several applications. Dynamic network-wide traffic volumes can be subject to energy use or emission estimation analyses and traffic policy applications, such as pricing schemes. Several studies (Günneemann et al. 2004; Jing et al. 2016; Salanova Grau et al. 2018; Zhan et al. 2016), attempting to exploit the increased availability of the probe vehicle data, propose local, data-driven estimation approaches for obtaining network-wide traffic flows. Such estimations usually rely on speed-flow relationships derived from the fundamental diagram of traffic flow.

In contrast to the probe vehicle data, cross-sectional data from stationary detectors contain traffic flow information. However, due to maintenance and installation cost constraints, such detectors are installed, and count flows only at specific network locations. Therefore, traffic flow has again to be estimated for the parts of the network without permanent cross-sectional counts. Accordingly, several local, data-driven estimation approaches have been proposed, including regression and neural networks techniques (Basarić et al. 2014; Batterman, Cook, and Justin 2015; Coelho et al. 2014; Fu, Kelly, and Clinch 2017; Gastaldi et al. 2013; Lindhjem et al. 2012; Zhong, Lingras, and Sharma 2004).

However, using such local estimation approaches based either on cross-sectional or probe vehicle data, traffic flow is typically estimated separately for each network link. Hence, vehicle conservation cannot be guaranteed at the network nodes, and the total number of the estimated vehicles may not coincide with the demand of the network. Vehicles travelling through two consecutive network links may appear or disappear influencing the accuracy of applications such as emission modelling. Besides, we cannot acquire any route information; there is no evidence of which is the origin or the destination of the estimated number of vehicles in the link. Thus, local traffic flow estimates may be useless for applications, such as OD demand calibration, where route information or turning proportions are required. A need, then, emerges for restricting these data-based estimations by incorporating them in a methodological modelling framework where traffic flow is conserved and route choice is consistent with the link flows.

Dynamic Traffic Assignment (DTA) is a widely applied traffic modelling framework that relates the demand for a network with the corresponding infrastructure supply. The two primary components of a DTA modelling framework are the route choice and the network loading sub-models. The route choice sub-model assigns the demand, namely the desired trips from an origin to a destination, among the available routes connecting that origin-destination (OD) pair. The assignment process usually takes into consideration the generalised cost of each route, which at the same time is the output of the network loading sub-model. Dynamic Network Loading (DNL) models propagate the assigned route volume through the route links attempting to capture congestion and ensure vehicle conservation. Therefore, the DTA solution algorithms usually iterate between the route choice and the network loading until an equilibrium is achieved. In essence, such equilibrium reflects the assumed behavioural principles based on which travellers are making their choices.

Nevertheless, the iterative network loading procedure required for the attainment of equilibrium usually requires extensive computational resources for large scale networks. For this reason, several studies (Bierlaire, Chen, and Newman 2013; Bierlaire and Frejinger 2008; Jagadeesh and Srikanthan 2017; Tettamanti, Demeter, and Varga 2012; Toole et al. 2015; Yang, Lu, and Hao 2017) suggest that route costs can be instead exogenously estimated from observations, e.g. from probe vehicle data. In this way, the application of a DNL for obtaining the route costs is not needed, diminishing
the computational effort. However, we should note here that exogenous route costs may not necessarily lead to an equilibrium. The output of such data-driven route choice estimations, i.e. the number of travellers which have chosen each route can be vital for several applications of DTA. Nevertheless, for applications that require network-wide dynamic flows and volumes, a one-shot network loading still has to be applied.

DNL, given route departure rates, aims at estimating dynamic link volumes and travel times (Xu et al. 1999), and together with a route choice sub-model form a DTA model. The importance of using an efficient DNL approach has been highlighted in many studies (Nie and Zhang 2005) since DNL influences the final DTA solution both in terms of accuracy and computational burden. The DNL is commonly performed using either macroscopic analytical traffic flow models or microscopic simulation. Micro-simulation models, by their nature, are associated with a detailed description of traffic which expands the required computing resources. Hence, analytical modelling approaches seem more appealing for large scale networks. Analytical DNL models propagate the traffic flow through a network such way that i) vehicle conservation is ensured and ii) congestion effects are incorporated. Traditionally, congestion is captured by a performance function that relates the demand with the supply of a link and includes endogenous variables, such as link volumes or densities. The same variables are also included in the conservation equation yielding a system of first-order partial (for continuous-space models) or ordinary (for discrete-space model) differential equations. Such a system is typically solved employing numerical methods which may be computationally expensive for short discretisation intervals or introduce diffusion errors for long ones.

In this study, we suggest that dynamic network loading can be data-driven instead. Attempting to exploit the increasingly available network-wide traffic-related data, we propose the Data-driven Dynamic Network Loading (DDNL) approach. Our aim is to accurately model the flow propagation, solely relying on exogenous variables, such as travel time observations. To the best of our knowledge, although several data-driven network loading techniques have been lately suggested (Krishnakumari et al. 2019; Ma and Qian 2018; Yang, Lu, and Hao 2017), none of them is developed in compliance with the fundamental traffic flow theory principles. Principles such as FIFO, vehicle conservation and queue propagation are not extensively discussed, while the accuracy of such techniques under congested conditions is not evaluated. Most of them follow a route-based network loading structure and are associated with case-specific input data requirements. Thus, their applicability to a general theoretical modelling framework is limited.

Our approach is reconciled with the theoretical network loading framework ensuring the vehicle conservation and FIFO. Moreover, we follow a link-node modelling structure, which reduces the complexity compared to the route-based approaches. The DDNL does not rely on any infrastructure attribute, and therefore its only inputs are the network-wide travel time observations and route departure rates. The ease of use and the computational efficiency of this method may be crucial for applications where the monitoring of network-wide historical flows is required. We should highlight here that as any other data-driven approach, the DDNL cannot capture long-term changes in demand or supply, and therefore it is not developed for long-term planning purposes.

The paper is structured as follows: Section 2 presents the main background of this study by describing the traditional DNL approaches, while it also provides a review of some data-driven approaches suggested in the literature. The description of the DDNL is given in Section 3, providing the basic formulations and properties
of our approach. Section 4 presents and evaluates the results of simulation-based experiments. In Section 5, we discuss the main findings and the limitations of this study, while we conclude and suggest directions for future research in Section 6.

2. Analytical DNL Approaches

Analytical traffic flow models are efficient tools for propagating the flow and estimating the routes’ costs. Thus, they are frequently used in the network loading process. They typically consist of a flow propagation and a node sub-model. Flow propagation models describe how traffic flow moves along a road segment, given an inflow profile. They are usually formulated as a system of differential equations, including a vehicle conservation equation, coupled with a relation between the traffic variables in steady-state that aims at reflecting congestion effects. Node models determine the transition flows from the incoming to a node road segments to the outgoing ones.

2.1. Network Definition and Notations

Consider a transportation network represented by a directed graph $G(\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$ is a set of nodes and $\mathcal{A}$ is a set of links. Let $\mathcal{R}$ be the set of origin nodes with $\mathcal{R} \subseteq \mathcal{N}$, and $\mathcal{S}$ be the corresponding set of destinations with $\mathcal{S} \subseteq \mathcal{N}$. Given the graph, there exists a finite set of the most likely used routes, $\mathcal{K}_{r,s}$, that connect OD pair $rs$, $r \in \mathcal{R}$, $s \in \mathcal{S}$. Moreover, let the length of the study period be denoted by $T$. Let us now introduce the term continuous-time route departure rate–flow and denote it by $f_{r,s,k}(t), t \in [0, T), k \in \mathcal{K}_{r,s}$. We assume that routes flows are known, being determined by the route choice sub-model. Then, a typical DNL approach aims at propagating such route flows over the links of the route, resulting in continuous-time link flows and volumes.

2.2. Macroscopic Flow Propagation Models

Macroscopic flow propagation models are commonly based on the hydrodynamic analogy between the traffic flow and the one-dimensional flow of continuous media such as fluids or gases. The corresponding conservation law

$$\frac{\partial \rho(x,t)}{\partial t} + \frac{\partial Q(x,t)}{\partial x} = 0,$$

(1)

describes how a change in density, $\rho$, over time, $t$, relates to a change of flow, $Q$, over space, $x$. Lighthill and Whitham (1955) and Richards (1956) considered a supplementary steady-state equation where flow is a function of density, $Q_{f}$, known as the fundamental diagram. The resulting LWR model (Eq. (2)) consists of a non-linear, first order partial equation of density,

$$\frac{\partial \rho(x,t)}{\partial t} + \frac{\partial Q_{f}(\rho(x,t))}{\partial x} = 0.$$ 

(2)

In most cases, Eq. (2) needs to be solved numerically, which is typically performed using finite-differences approaches. Daganzo (1994) proposed an algorithm for solving Eq. (2), known as the Cell Transmission Model (CTM), which adopts a time and space
discretisation numerical scheme (Godunov 1959). Space is divided into homogeneous cells, with the cell size being a trade-off between accuracy and efficiency. Short cells, which also imply dense time discretisation due to Courant-Friedrichs-Lewy conditions, increase the computational effort. On the other hand, longer cells entail discretization errors known as numerical diffusion (Treiber and Kesting 2013).

**Link models**

Although continuous space-time models are more suitable for describing the spatio-temporal evolution of traffic flow, the complexity of their numerical solutions discouraged their inclusion in DTA and paved the way for the development of discrete-space models, called **link models**. The space dimension is omitted here as the change of flow over space can be approximated as $(v(t) - u(t))/L$, where $v(t)$ is the outflow and $u(t)$ the inflow rate at each time instant $t$, and $L$ the length of the space discretisation interval, which in this case corresponds to a link. Accordingly, the average density of link equals to $q(t)/L$, where $q(t)$ is the link volume. Therefore, the corresponding conservation law for the link models is

$$\frac{dq_a(t)}{dt} = u_a(t) - v_a(t), \quad \forall a \in A, t \in [0, T),$$

or

$$q_a(t) = U_a(t) - V_a(t), \quad \forall a \in A, t \in [0, T),$$

where $U_a(t)$ and $V_a(t)$ is the cumulative number of vehicles which have entered or exited the link $a$ until time $t$, respectively, with

$$U_a(t) = \int_0^t u_a(\omega) d\omega, \quad V_a(t) = \int_0^t v_a(\omega) d\omega, \quad \forall t \in [0, T).$$

In the earliest formulations of link models (Merchant and Nemhauser 1978; Wie, Friesz, and Tobin 1990), Eq. (3) is coupled with a link **performance** function which relates the exit flow to the link volume and aims at capturing congestion. The outflow is given as a continuous, strictly decreasing function of volume, $v_a(t) = W[q_a(t), \Gamma]$, where $\Gamma$ is a vector of parameters which reflect infrastructure attributes, such as free bottleneck or link storage capacities. The main drawback of such **exit link** formulations is that the FIFO rule is not explicitly guaranteed.

Alternatively, Friesz et al. (1993) are the first who expressed the DNL problem using a **link travel time** formulation. The link performance function regards the link travel time, $\tau_a(t)$, which is given as a strictly increasing function of volume, $\tau_a(t) = G[q_a(t), \Gamma]$. By definition, the link Travel Time Function (TTF), $\tau_a(t)$, denotes the required time that a user, who departs at time $t$ from the starting point of a link, needs to reach the end of the same link. An additional equation, called **propagation** equation, is employed here to relate the inflow and outflow to the link travel time,

$$V_a(t + \tau_a(t)) = U_a(t), \quad \forall a \in A, t \in [0, T),$$

Eq. (6) connotes that the cumulative number of vehicles which have entered a link at time instant $t$ equals to the cumulative number of vehicles which will exit the link $\tau(t)$ later, ensuring that no vehicle is generated or disappears along a link. By taking
the derivatives of both sides of Eq. (6), a relationship between the inflow, the outflow rate and the travel can be derived

\[ v_a(t + \tau_a(t)) = \frac{u_a(t)}{1 + \frac{d\tau_a(t)}{dt}}, \quad \forall a \in A, t \in [0, T). \quad (7) \]

Utilising Eq. (7), we can infer some important required properties regarding the TTF. Given that the inflow is greater than zero, non-negativity of the outflow is ensured if

\[ \frac{d\tau_a(t)}{dt} > -1. \quad (8) \]

Moreover, if flow moves in the right direction, FIFO rule is never violated for positive inflows and outflows (Astarita 1996).

Adamo et al. (1999); Astarita (2002, 1996); Wu, Chen, and Florian (1998); Xu et al. (1999) suggested similar link travel time formulations, forming the generic macroscopic link model (Nie and Zhang 2005),

\[ \frac{dq_a(t)}{dt} = u_a(t) - v_a(t), \quad \text{or} \quad q_a(t) = U_a(t) - V_a(t), \quad (9a) \]

\[ v_a(t + \tau_a(t)) = \frac{u_a(t)}{1 + \frac{d\tau_a(t)}{dt}}, \quad \text{or} \quad V_a(t + \tau_a(t)) = U_a(t), \quad (9b) \]

with

\[ \tau_a(t) = W[q_a(t), \Gamma], \quad \text{or} \quad v_a(t) = G[q_a(t), \Gamma], \quad \forall a \in A, t \in [0, T). \quad (9c) \]

Therefore, the output of a link model is the outflow, which explicitly or implicitly depends on the link performance, i.e. the current level of congestion. Note that Eq. (9) regards path-aggregated link flows, but DNL algorithms typically store route information for each link.

**Wave link models**

The traditional link models (Eq. (9)) treat each link as a homogeneous entity in terms of volumes and densities; the link behaviour is determined based on link-average volumes. Queues are usually vertical without a physical length, and hence, traditional link models cannot describe the propagation of traffic states along a link or spill-back effects. The later may deteriorate the performance of DTA and affect applications that require an accurate spatial estimation of congestion, such as emission modelling (Tsanakas, Ekström, and Olstam 2020).

The so-called wave link models constitute an alternative macroscopic flow propagation approach. The Link Transmission Model (Gentile et al. 2010; Raadsen and Bliemer 2019; Raadsen, Bliemer, and Bell 2016; Yperman 2007, LTM) belongs to this family where the propagation of traffic states through a link is implicitly considered. Each change of traffic state at the link boundaries triggers the formation of a shock-wave, whose propagation speed is determined according to Newels’ simplified theory of kinematic waves (Newell 1993). The LTM flow propagation model is coupled with a
node model that restricts the outflow of the incoming links based on the inflow capacity of the outgoing ones. In contrast to the typical link models, link inflows can also be restricted here when link density reaches its maximum since queues have physical length. The maximum density and the other parameters of the fundamental diagram correspond to vector \( \Gamma \), and their values are obtained from calibration. Wave link models can effectively capture the phenomena of flow metering and spill-back, by taking into account both inflow capacity and maximum density constraints, respectively. However, the traditional link models are still widely used for DNL in various DTA approaches, mainly due to their simplicity in comparison to the wave link models.

2.3. Node Models

The outflow of each incoming link to a node is subject to the node model, which determines the transition flows and the inflows to the outgoing links. Let \( A^-_n \) and \( A^+_n \) be the set of the incoming and outgoing links to node \( n, n \in \mathcal{N} \), respectively. A node model typically seeks the realised transition flows, \( \bar{q}_{a,b}(t) \), from each incoming link \( a, a \in A^-_n \), to each outgoing link \( b, b \in A^+_n \). Every incoming link \( a \) is associated with a maximum demand, \( D_a(t) \), which restricts the link outflow, \( v_a(t) \), to the realised sending flow, \( s_a(t) \). The maximum demand is exogenous to the node model and depends on the traffic conditions inside link \( a \) and its capacity. Utilising the realised transition flows we can express the realised sending flow as \( s_a(t) = \sum_{b \in A^+_n} \bar{q}_{a,b}(t) \). Therefore, the maximum demand implies that

\[
\sum_{b \in A^+_n} \bar{q}_{a,b}(t) \leq D_a(t), \quad \forall a \in A^-_n, n \in \mathcal{N}. \tag{10a}
\]

Moreover, each outgoing link \( b \), is given an exogenous maximum supply, \( S_b(t) \), that accordingly restricts the receiving flow \( r_b(t) \), with \( r_b(t) = \sum_{a \in A^-_n} \bar{q}_{a,b}(t) \). Hence,

\[
\sum_{a \in A^-_n} \bar{q}_{a,b}(t) \leq S_b(t), \quad \forall b \in A^+_n, n \in \mathcal{N}. \tag{10b}
\]

The so-called splitting ratios, \( \mu_{a,b}(t) \), relate the transition flows to the partial sending flows as

\[
\bar{q}_{a,b}(t) = \mu_{a,b}(t) \sum_{b \in A^+_n} \bar{q}_{a,b}(t), \quad \forall a \in A^-_n, n \in \mathcal{N}. \tag{10c}
\]

The route choice sub-model usually defines the splitting ratios, and thus, they are also external to the node model.

A node model typically attempts to maximise the node’s total realised sending flows, \( \sum_a \sum_b \bar{q}_{a,b}(t) \), given non-negativity, \( \bar{q}_{a,b}(t) \geq 0 \), demand (Eq. (10a)), supply (Eq. (10b)), and splitting ratio constraints (Eq. (10c)). Such formulation defines the so-called generic class of first-order node models. For a more detailed description of the generic first-order node models, we refer to Tampère et al. (2011). Once the transition flows are determined, the inflow at the outgoing links can be computed as

\[
u_b(t) = \sum_{a \in A^-_n} \bar{q}_{a,b}(t), \quad \forall b \in A^+_n, n \in \mathcal{N}. \tag{11}\]
2.4. **Exogenous Description of Congestion**

The traditional DNL models, consisting of the propagation sub-model, Eq. (9), and the node model, Eq. (10), are typically used for long-term planning purposes. For example when evaluating effects of changes in the infrastructure or effects from introducing new regulations or policies. For this reason, the link model described by Eq. (9) employs *endogenous* variables for capturing congestion. At each time instant \( t \), the link outflow is implicitly or explicitly determined from the traffic volume and its relation to the supply. At the same time, traffic volume is obtained by the difference between inflow and outflow. Hence, finding a solution to the system of Eq. (9) becomes a difficult task. Several approaches have been suggested for approximating the solution through numerical schemes, employing time discretisation. Moreover, the calibration of the infrastructure parameters reflecting the supply both in the link and the node model is often a demanding and time-consuming process which relies on data availability and reliability.

While endogenous congestion in DNL is crucial for long-term planning, several researchers (Ashok and Ben-Akiva 2002; Cascetta, Inaudi, and Marquis 1993; Ma and Qian 2018; Nam et al. 2017; Seo and Kusakabe 2015; Sohn and Kim 2008) suggest that congestion can instead be given *exogenous* for short-term planning or monitoring purposes. The later may concern applications where the estimation of historical traffic flows is required, such as emission estimation or OD demand calibration. Although we can consider endogenous congestion in such cases as well, exogenous congestion could be more beneficial because i) the system of Eq. (9) can be decomposed and ii) the supply parameters (and consequently their calibration) can be excluded. The exogenous congestion approaches typically rely on field data, and thus we refer to them as *data-driven* approaches. While traditional flow propagation models usually deploy a demand-supply relationship (a performance function or fundamental diagram) to determine the congestion level, data-driven propagation techniques exploit external variables (speed or travel time observations) for this purpose.

In the literature, one can find several studies which address the problem of propagating the traffic flow given some exogenous observations. Seo and Kusakabe (2015) and Nam et al. (2017) suggest probe vehicle-based techniques for estimating flow propagation, which adhere to the conservation law and are independent of any supply constraint. The probe vehicles are equipped with sensors that detect adjacent vehicles and measure the spacing. Spacing is then the main input for estimating the traffic variables. These approaches are developed in the context of a highway traffic flow estimation and are associated with a specific type of input data which limits their applicability to a network scale.

Alternatively, several data-driven network-wide flow propagation methods have been proposed from an OD demand calibration perspective. The most straightforward approach assumes that a group of vehicles behave as a single entity forming discrete packets which travel at discrete-time (Krishnakumari et al. 2019). During every discretised time interval, the vehicles of each packet simultaneously depart from their origin. Then, each packet travels through the links of the networks given a specific link travel time (or average speed) according to the available field data. The accuracy of such approaches depends on the discretisation length–size of packets.

Contrariwise, according to the continuous packet approach (Cascetta, Inaudi, and Marquis 1993), packets have a physical length in time dimension as vehicles are evenly distributed among the first (head) and the last (tail) vehicle of the packet. Hence, packets can travel at continuous-time. At the earliest formulations of the continuous
packet approaches, the length of each packet is fixed while every vehicle in the packet has the same travel time–speed. However, Ashok and Ben-Akiva (2002) propose that the assumption of a fixed packet length can be relaxed as vehicles may stretch or squeeze among the head and the tail of the packet. This approach permits the two boundary vehicles to have different link travel times. The difference in travel time is, then, uniformly distributed over the remaining vehicles of the packet, i.e. travel time linearly evolves from the first to the last vehicle. Ashok and Ben-Akiva (2002) also highlight the difficulties of using such continuous packets because information on the departure and arrival time of the head and the tail of each packet is required. Sohn and Kim (2008) follow a continuous packet approach and attempt to utilise data from mobile cellular networks to determine the travel times of a packet’s head and tail. Nevertheless, the cellular network structure of this approach impedes its applicability on general link-node networks.

The authors of a more recent study (Ma and Qian 2018) also adopt the continuous packet approach. They derive the travel times of the head and the tail from a link TTF. In contrast to the traditional analytical DNL approaches, the TTF is derived from exogenous probe vehicles-based observations. However, the method of obtaining such a link TTF, as well as its form and mathematical properties are not described in detail. Even though the propagation of the two boundary vehicles coincide with the value that the TTF returns for their departure time, an arbitrary TTF form may lead to inconsistencies for the remaining vehicles of a packets. According to the continuous packet approach, the travel time linearly evolves from the first to the last vehicle of a packet. Therefore, the link travel time should be a linear function of time throughout the departure of the packet’s vehicles.

Although at first glance, the propagation of flow given travel time observations seems a straight forward process, aspects such as the demand or time discretisation, travel time-flow consistency, and vehicle conservation should also be addressed here. Most of the data-driven flow propagation approaches neglect discussions on FIFO and spill-back; principles which are widely investigated in the traditional endogenous DNL models. Furthermore, such data-driven techniques (Ashok and Ben-Akiva 2002; Krishnakumari et al. 2019; Ma and Qian 2018; Sohn and Kim 2008) provide limited references on the required field data type and availability.

In this study, we suggest a continuous packet approach using an exogenous continuous TTF which is constrained according to the theoretical DNL framework. The TTF has a piece-wise linear form and can be estimated by utilising any dynamic link travel time information. Therefore, our approach can be integrated into any DNL methodological framework where some kind of link travel time observation is available. Furthermore, the piece-wise linear form of the TTF ensures the travel time-flow consistency. Each assigned vehicle travel time coincides with the value that the TTF returns for the departure time of that vehicle. An additional novelty of our approach is that packets are generated only at specific time instants, called events. Thus, the computational burden is reduced. The piece-wise linear TTF together with the event-based packet structure provide a modelling framework which we can embed in the existing event-based network loading algorithms (Raadsen and Bliemer 2019; Raadsen, Bliemer, and Bell 2016).
3. Data-driven Dynamic Network Loading

3.1. Formulation and Notations

Let us assume that $N_a$ link travel time observations, $y_{a,\nu}$, are available for every link $a \in A$ at arbitrary time instants $x_{a,\nu}$, with $x_{a,\nu} \in [0, T]$ and $\nu = 1, \ldots, N_a$. We can, then, simplify the traditional DNL problem (Eq. (9)) to the Data-Driven Network Loading (DDNL) link model,

$$v_a(t + \tau_a(t)) = \frac{u_a(t)}{1 + \frac{\tau_a(t)}{dt}},$$

(12a)

$$\tau_a(t) = \tilde{g}(y_a), \quad \forall a \in A,$$

(12b)

where $\tilde{g}$ is a process for estimating the TTF, $\tau_a$, given the exogenous observations, $y_a = [y_{a,\nu}]_{\nu=1}^{N_a}$. Eq. (12a) is the corresponding propagation equation, describing how flow propagates through a link. Accordingly, Eq. (12b) is the performance function. A significant difference between Eq. (12b) and the traditional formulations of the performance function (Eq. (9c)) is that travel time neither depends on any endogenous variable nor any infrastructure attribute. We assume that the TTF can adequately reflect phenomena such as flow metering and spill-back, and hence we don’t impose any capacity or jam density constraints. We can also simplify the node model of Eq. (10) to the DDNL node model,

$$q_{a,b}(t) = \mu_{a,b}(t)v_a(t), \quad a \in A_n^-, b \in A_n^+, n \in \mathbb{N}.$$  

(13)

Thus, for the calculation of the realised transition flows, $q_{a,b}$, we only need the splitting ratios, $\mu_{a,b}$, and the outflow rates, $v_a$. We assume that the splitting ratios are known, being externally determined by the route choice model. Moreover, we assume that possible node delays are incorporated into the link travel time since, according to the typical node-link network representations, nodes do not have a physical length. Note that we do not impose any supply constraint at the node model as well.

We can then compute the inflow rate for each outgoing link $b$ of node $n$, $b \in A_n^+$, as

$$u_b(t) = \left\{ \begin{array}{ll}
\sum_{a \in A_n^-} \tilde{q}_{a,b}(t), & \text{if } n \neq r \\
\sum_{a \in A_n^-} q_{a,b}(t) + \sum_{k \in K_{r,s}} \delta_{b,k} f_{r,s,k}(t), & \text{if } n = r, n \in \mathbb{N}, r \in \mathbb{R}, s \in \mathbb{S},
\end{array} \right.$$

(14)

where $\delta_{b,k}$ is a link-route indicator taking the value of 1 when link $b$ belongs to route $k$ and 0 otherwise. According to Eq. (14), the inflow rate of link $b$ equals to the sum of the realised transition flows from each incoming link plus the route flow rates from the routes having $b$ as the first link. We assume that the route flow rates are known, being also determined by the route choice model. However, a route choice model typically provides demand period-specific discrete-time route volumes. Let the analysis period be divided into $H$ equal-length periods, termed demand periods, indexed by $h$. For each demand period, the route choice model determines the number of vehicles (stationary demand) which follow route $k$ of OD pair $r,s$, denoted by $\bar{f}_{k,h}$. Assuming that the $\bar{f}_{k,h}$ vehicles uniformly depart during the length of a demand period, $T_h$, we can derive

---

1We omit here the index $r,s$ for notational convenience. Actually, from here onwards we assume that $k$ is element of the global route set, $k \in \bigcup_{r \in \mathbb{R}} \bigcup_{s \in \mathbb{S}} K_{r,s}$. 

10
the continuous-time route flows as
\[ f_k(t) = \frac{f_{k,h}}{T_h}, \quad \text{if } (h - 1)T_h \leq t < hT_h, \quad h \in \mathcal{H}, \] (15)

where \( \mathcal{H} = \{1, \ldots, H\} \). Therefore, the cumulative inflow curve for the first link of each path has a piece-wise linear form while the inflow rate is piece-wise stationary. As we show in a later section, according to Eq. (12a) and given a piece-wise linear TTF, the outflow rate piece-wise stationary as well. This means that the conditions on the boundaries of each link change only at specific times-events. Such event-based grid-free formulation implies a reduced number of computations compared to fixed time interval approaches.

### 3.2. Estimation of the Travel Time Function

We assume that each link travel time is linear during specific time periods called linear travel time (LTT) periods, indexed by \( i \) and denoted by \( \mathcal{I}_{a,i} \). The number of LTT intervals, \( I_a \), and their collection, \( \mathcal{I}_a \), is different for each link \( a \). Furthermore, for every link, the collection of LTT intervals forms a partition of the analysis period \([0, T]\). Nevertheless, for notational convenience, we will omit the link index, \( a \), from here onwards in this section. We assume that during each LTT, a packet of vehicles departs from the downstream boundary of each link. The starting time of each LTT period, \( \theta_i \), is also the departure time of the first vehicle of the packet, whose travel time is \( \tau(\theta_i) \). The last vehicle departs just before the end of the period, \( \phi_i \), with its travel time being \( \tau(\phi_i) \). Then, the travel time of the vehicles departing during each LTT period is evenly distributed between the travel time of the first and last vehicle forming a linear TTF,

\[ \tau(t) = \alpha_i t + \beta_i, \quad \text{if } t \in \mathcal{I}_i = [\theta_i, \phi_i), i = 1, \ldots, I, \] (16)

where \( \alpha_i \) and \( \beta_i \) are the parameters of the linear TTF for the \( i^{th} \) LTT period. As the collection of the LTT intervals partition the analysis period, the TTF has a piece-wise linear form in \([0, T]\), with \( \theta_i \) being the breakpoints. Our aim, here, is to estimate the number of LTT periods, the breakpoints and parameters of each linear segment based only on the travel time observations, \( y \). We attempt to develop an estimation approach that adheres to the theoretical DNL framework. Hence, we constraint the estimation in such way that the outcome meets some desired properties; for any \( t \in [0, T) \): i) the travel time of each link \( a \) is greater than or equal to the free-flow travel time of that link, \( \tau_0^a \), ii) FIFO conditions are guaranteed, iii) non-negativity and continuity of outflow are ensured.

Given the observations data set, \( \{x_\nu, y_\nu\}_{\nu=1}^N \), we can estimate the piece-wise linear TTF of each link using segmented regression. Each piece or linear segment, \( i \), corresponds to a LTT period, \( \mathcal{I}_i \), which contains at least two data points to define a line, namely, there are at least two \( \nu \) for which \( x_\nu \in \mathcal{I}_i \). We assume that each travel time observation, \( y_\nu \), equals to

\[ y_\nu = \tau(x_\nu) + \varepsilon_\nu, \quad \nu \in \mathcal{L}, \] (17)
where \( \varepsilon_\nu \) is a noise term reflecting the measurements error and \( L = \{ 1, \ldots, N \} \). Then, we can estimate the parameters \( \alpha_i \) and \( \beta_i \) by minimising the squares of the error term for each linear segment. The latter can be achieved by solving the least square quadratic programming problem,

\[
[\alpha_i, \beta_i] = \min_{\alpha, \beta} \sum_{\nu \in M_i} [y_\nu - (\alpha x_\nu + \beta)]^2, \quad i = 1, \ldots, I,
\]

where

\[
M_i = \{ \nu \in L | x_\nu \in I_i \}.
\]

In order to meet the desired properties of the TTF, we impose some constraints to Problem (19). Firstly, for each link, the travel time should be greater than or equal to the free-flow travel time \( \tau_0 \). As the travel time linearly evolves among \( \theta_i \) and \( \phi_i \), the minimum travel time for the interval \( I_i \) corresponds either to \( t = \theta_i \), if \( \alpha_i > 0 \) or to \( t = \phi_i \), otherwise. Therefore, in any case, the inequalities

\[
\alpha_i \theta_i + \beta_i \geq \tau_0, \quad \text{and} \quad \alpha_i \phi_i + \beta_i \geq \tau_0, \quad i = 1, \ldots, I,
\]

should be valid. Regarding FIFO and non-negativity of outflow, both attributes are ensured if the derivative of travel time is greater than \(-1\), according to Eq. (8). In our case, the derivative of the TTF is equal to \( \alpha_i \) for any \( t \in (\theta_i, \phi_i) \) but is not defined for \( t = \theta_i \). Even though setting \( \alpha_i > -1 \) ensures FIFO for any \( t \in (\theta_i, \phi_i) \), FIFO cannot be guaranteed for any \( t \) where the derivative of travel time is not defined. In other words, although FIFO is valid for a packet of vehicles departing during a LTT period, it is not ensured among vehicles of different packets.

Figure 1 illustrates the estimated linear piece-wise TTF, \( \tau_i \), for three LTT periods, \( I_i, i = 1, 2, 3 \), as well as the trajectories for some representative vehicles. The thicker lines connote the trajectory of the first and the last vehicle of each packet. The first vehicle departs from the most upstream node of link, \( a_0 \), at \( t = \theta_1 \) and the last at \( t = \phi_1 \). Accordingly, the arrival time to the most downstream end of link \( a \), \( a_L \) is denoted by \( \theta'_i \) for the first vehicle and by \( \phi'_i \) for the last vehicle, respectively, with \( \theta'_i = \theta_i \alpha_i + 1 + \beta_i \) and \( \phi'_i = \phi_i \alpha_i + 1 + \beta_i \). We notice a FIFO violation here as some vehicles of the third packet arrive at \( a_L \) earlier than some vehicles of the second packet. Additionally, note the discontinuity of flow period, i.e. period of time where the outflow rate is equal to zero, between \( \phi'_1 \) and \( \theta'_2 \). Such FIFO violation and flow discontinuity periods may result to a highly fluctuated outflow rate; the outflow rate is zero for \( \phi'_1 \leq t < \theta'_2 \) and excessive for \( \theta'_2 \leq t < \phi'_2 \) when the flow of two packets is merged.

Nevertheless, if we constraint the first vehicle of packet \( i \) to arrive at \( a_L \) no earlier than the last vehicle of packet \( i-1 \),

\[
\theta'_i \geq \phi'_{i-1} \Rightarrow \alpha_i \theta_i + \beta_i \geq \alpha_{i-1} \phi_{i-1} + \beta_{i-1}, \quad i = 2, \ldots, I,
\]

FIFO is ensured for any \( t \in [0, T) \), given \( a_i > -1 \) and \( \theta_i = \phi_{i-1}, i = 2, 3, \ldots, I \).
Furthermore, if we transform inequality (22) to equality,

$$\theta'_i = \phi'_{i-1} \Rightarrow \alpha_i \theta_i + \beta_i = \alpha_{i-1} \theta_i + \beta_{i-1}, \quad i = 2, \ldots, I,$$

the TTF becomes continuous for any \( t \in [0, T) \). In this way, the zero-outflow periods are also eliminated, ensuring the continuity of the outflow.

Imposing each \( \alpha_i \) to be greater than \(-1\), not only results in positive outflows but also leads to smoother recoveries of traffic after congestion. Negative slopes of the linear TTF, usually connote transition from congested to flowing conditions. Setting a lower limit to such slopes establishes a smoother and more realistic transition. Accordingly, we also set here an upper limit equal to 1, to avoid rough transitions from flowing to congested conditions. Finally, for each LTT period, \( i \), the parameters \( \alpha_i \) and \( \beta_i \) are estimated by solving the constrained least square problem problem

$$[\alpha_i \beta_i] = \min_{\alpha \beta} \sum_{\nu \in \mathcal{M}_i} [y_{\nu} - (\alpha x_{\nu} + \beta)]^2,$$

subject to

$$|\alpha| < 1,$$

$$\alpha \theta_i + \beta > 0,$$

$$\alpha \phi_i + \beta > 0,$$

$$\alpha \theta_i + \beta = \alpha_{i-1} \theta_i + \beta_{i-1}.$$

The problem of finding the optimal number and length of the LTT periods, namely the number and location of the breakpoints, can be treated as a segmentation problem and be algorithmically solved. We choose the use of a sliding window algorithm (see Algorithm 1). Due to their simplicity and intuitiveness, sliding window algorithms have been widely applied in various fields for solving segmentation problems.
Algorithm 1 starts by anchoring the most left point of the window indexed by the anchor indicator \( \hat{l} \), namely the first point among the potential points of set \( \mathcal{M} \) that are used for the estimation of the \( i^{th} \) linear segment. Next, at each iteration, the algorithm slides the window to the right by adding the point indexed by \( \hat{r} \) in the \( \mathcal{M} \) set, estimates the linear segment and computes a selected measure of estimation error, which in this case is the average squared error. At some point, \( \hat{r} \), the estimation error becomes greater than a predefined threshold, \( \tilde{\varepsilon}^2 \). This point becomes a transition point and is the last one included for the estimation of the \( i^{th} \) segment. Furthermore, this point becomes the first point for the estimation of the \( i+1^{th} \) segment. The same procedure is repeated for the estimation of any linear segment, \( i \), until the index of the last point in the window, \( \hat{r} \), becomes equal to the total number of the input data set, \( N \).

The value of the accepted error, \( \tilde{\varepsilon} \), is a trade-off between accuracy and complexity, and its selection depends on the required accuracy level of the DDNL’s application. On the one hand, low values of \( \tilde{\varepsilon} \) result in more accurate estimations of the TTF. However, at the same time, they may lead to an increased number of linear segments, namely an increased number of LTT periods, \( I \). Augmented values of \( I \) proliferate the number of events while flow propagates through the links of a route (see Subsection 3.3), adding complexity in the DDNL. On the other hand, higher values of \( \tilde{\varepsilon} \) tend to generate a decreased number of linear segments. Such simplified representation may not adequately capture the travel time variations, diminishing the accuracy. Note that the accepted error, \( \tilde{\varepsilon} \), and the free-flow travel times, \( \tau_0 \), are two additional inputs to the estimation of the TTF.

Figure 2 illustrates the estimated TTF for an imaginary link \( a \) considering two characteristic values of the accepted error, given some hypothetical travel time estimates \( y_\nu \). Accordingly, Figure 3 presents how the accepted error, \( \tilde{\varepsilon} \), influences the number of linear segments and the mean estimation error,

\[
\tilde{\varepsilon} = \frac{\sum_{\nu \in \mathcal{L}} \sqrt{\left[ y_\nu - \tau(x_\nu) \right]^2}}{|\mathcal{L}|}.
\]  

Note that for higher values of the accepted error, the piece-wise representation of travel time is simplified and the quality of the estimation is deteriorated. Contrariwise, the estimation is improved with lower accepted error values, but the number of linear segments is also increased, which may affect the efficiency of the DDNL.

\footnote{The transition point indexed by \( \hat{r} \) is therefore included in both estimations of the first and the second linear segment. Although considering the transition point for estimating both the \( i^{th} \) and the \( i+1^{th} \) linear segment increases the estimation errors, it introduces a smoother transition between the two consecutive linear segments.}
Figure 2.: The estimated TTF for link \( a \), for an accepted error, \( \tilde{\varepsilon} \), of 5 and 30 seconds. The small circles symbolise the breakpoints of each piece-wise curve.

Figure 3.: Number of segments of the piece-wise linear TTF and mean estimation error, \( \bar{\varepsilon} \), for the same link \( a \) and for different values of the accepted error, \( \tilde{\varepsilon} \).

**Algorithm 1:** Sliding window algorithm for the estimation of the travel time function

**Input:** \( \{x_\nu, y_\nu\}_{\nu=1}^N \), \( \tilde{\varepsilon}, \tau_0, T \)

**Output:** \( I, \alpha_i, \beta_i, \theta_i, \phi_i, i = 1, \ldots, I \)

**Initialization:** \( \theta_1 = 0, \hat{l} := 1, \hat{r} := 2, i := 1, \mathcal{M} := \{1\} \)

**while** \( \hat{r} \leq N \) **do**

**if** \( i = 1 \) **then**

\[
[\alpha^*, \beta^*] := \arg \min_{\alpha, \beta} \sum_{\nu \in \mathcal{M}} [y_\nu - (\alpha x_\nu + \beta)]^2,
\]

subject to

\[
|\alpha| < 1, \quad \alpha \theta_1 + \beta > \tau_0, \quad \alpha x_{\hat{r}} + \beta > \tau_0
\]

**else**

\[
[\alpha^*, \beta^*] := \arg \min_{\alpha, \beta} \sum_{\nu \in \mathcal{M}} [y_\nu - (\alpha x_\nu + \beta)]^2,
\]

subject to

\[
|\alpha| < 1, \quad \alpha \theta_1 + \beta > \tau_0, \quad \alpha x_{\hat{r}} + \beta > \tau_0, \quad \alpha \theta_i + \beta = \alpha_{i-1} \theta_i + \beta_{i-1}
\]

**end**

**if** \( \sum_{\nu \in \mathcal{M}} [y_\nu - (\alpha^* x_\nu + \beta^*)]^2 / |\mathcal{M}| > \tilde{\varepsilon}^2 \) **then**

\[
\alpha_i = \alpha^*, \quad \beta_i = \beta^*, \quad \phi_i = \frac{x_{\hat{r}-1} + x_{\hat{r}}}{2}, \quad \theta_{i+1} = \phi_i,
\]

\( \hat{l} := \hat{r}, \quad \mathcal{M} := \{\hat{l}\}, \quad i := i + 1 \)

**end**

\( \hat{r} := \hat{r} + 1 \)

**end**

\( \alpha_i = \alpha^*, \quad \beta_i = \beta^*, \quad \phi_i = T, \quad I = i \)
3.3. Flow propagation

We consider stationary demand during each demand period that yields piece-wise stationary inflow rates for the first link of each route. Moreover, the piece-wise linear travel time leads to piece-wise stationary outflow rates, which accordingly result in piece-wise stationary inflow rates for the succeeding links. Hence, the conditions on the boundaries of each link are constant between two discrete events. Let $t_j, \bar{t}_z$ be the time instants when a change occurs on the upstream and the downstream boundary of a link, respectively. We can compute the continuous flow rates and cumulatives on the link boundaries as,

\[
\begin{align*}
u(t) &= u_j \quad \text{and} \quad U(t) = U(t_j) + u_j \cdot (t - t_j), \quad \text{if} \quad t_j \leq t < t_{j+1}, \quad j = 1, \ldots, J, \quad (26) \\
v(t) &= v_z \quad \text{and} \quad V(t) = V(\bar{t}_z) + v_z \cdot (t - \bar{t}_z), \quad \text{if} \quad \bar{t}_z \leq t < \bar{t}_{z+1}, \quad z = 1, \ldots, Z, \quad (27)
\end{align*}
\]

where $u_j$ and $v_z$ is the $j^{th}$ piece-wise stationary inflow and the $z^{th}$ piece-wise stationary outflow, respectively. Therefore, finding the discrete flow rates and event times is enough for the flow propagation in continuous time. Assume that $b$ is an outgoing link of node $n$, $b \in A_n^+, n \in N$. The inflow rate of link $b$ changes from $u_{b,j}$ to $u_{b,j+1}$,

- when a transition in demand periods occurs, namely at every $t = (h - 1)T_h, h \in H$, if $n$ is an origin, $n \in R$, or,
- when the outflow rate of an incoming link $a$ to node $n$, $a \in A_n^-$, changes, i.e. at every $t = \bar{t}_{n,z}, z = 1, \ldots, Z_a$.

Given that the TTF has the piece-wise linear form of Eq. (16), the continuous-time outflow rate equals \(^{34}\)

\[
v(t + \tau_i(t)) = \frac{u_j}{\alpha_i + 1}, \quad \text{if} \quad t_j \leq t < t_{j+1}, \quad \text{and} \quad \theta_i \leq t < \theta_{i+1}, \quad (28)
\]

or

\[
v(t) = \frac{u_j}{\alpha_i + 1}, \quad \text{if} \quad t_j + \tau_i(t_j) \leq t < t_{j+1} + \tau_i(t_{j+1}), \\
\quad \text{and} \quad \theta_i + \tau_i(\theta_i) \leq t < \theta_{i+1} + \tau_i(\theta_{i+1}), \quad (29)
\]

where $t_i(t) = \alpha_i t + \beta_i$. Hence, the outflow rate is stationary for specific time intervals. Accordingly, we have a transition from $v_z$ to $v_{z+1}$,

- $\tau_i(t_j)$ time units after the occurrence of a change at the link inflow at $t_j, j = 1, \ldots, J$, or,
- $\tau_i(\theta_i)$ time units after a shift in the slope of the TTF at $\theta_i, i = 1, \ldots, I$.

Event-Based Algorithm

We suggest the use of an event-based algorithm (see Algorithm 2) to determine the sequence of downstream and upstream events in a generalised network, $G(N, A)$. Given that any demand or LTT period transition triggers a state change on a link boundary, we employ Algorithm 2 to propagate such state perturbation further downstream in the network. Raadsen and Bliemer (2019) provide an extensive discussion on the DNL event-based algorithms and their efficiency at the continuous-time flow propagation.

\(^{3}\)We skip the link index also here.

\(^{4}\)Since $\phi_i = \theta_{i+1}$, from here onwards we use $\theta_{i+1}$ to denote the end of a LTT period.
Let us highlight here the main differences between Algorithm 2 and the event-based solution algorithms in the literature that consider endogenous congestion (Raadsen and Bliemer 2019; Raadsen, Bliemer, and Bell 2016). In both approaches (endogenous and exogenous congestion), the event-based algorithms predict a set of time instants at which a state change is expected to occur on either the downstream or upstream boundary of each link. However, in the endogenous congestion approach, such a set of pending events requires validation. A state change travels either downstream (positive shock-wave speed) or upstream (negative shock-wave speed), while two different shock-waves may collide. Any interaction between two shock-waves yields a new shock-wave, causing the invalidation of the corresponding pending events.

Contrariwise, the exogenous congestion’s nature description combined with the FIFO restriction allows us to skip the event validation step, enhancing the efficiency of the algorithm. As we do not consider any supply constraints, every state change propagates downstream, while FIFO prevents any shock-wave interaction. Therefore, each pending event is already positively validated. Moreover, the node model occupies a remarkable portion of the total computational time of the endogenous congestion solution algorithms (Raadsen and Bliemer 2019). Our simplified and unconstrained way of determining the transition flows at the nodes further reduces the computational burden.

The inputs to Algorithm 2 are the network structure, the parameters of the piece-wise linear TTF for each link and the demand period-specific route volumes. For each link, the algorithm stores the combination of the time and inflow, \((t_j, u_j)\), for the \(j^{th}\) upstream event, and the combination of the time and outflow, \((\bar{t}_z, v_z)\), for the \(z^{th}\) downstream event, accordingly. We distinguish three possible types of events: event due to (i) transition of demand period, called demand event, (ii) change in the slope of the TTF, named travel time event, and (ii) arrival of any upstream flow perturbation on the downstream link boundary, referred to as downstream event.

At each iteration of Algorithm 2, we determine a candidate event time for every event type. While the time \(t\) is less than the analysis period, \(T\), the algorithm seeks for the earliest candidate event and the time is updated accordingly. A demand event triggers a state transition on the upstream boundary of the first link of each route, which in turn adds a pending event on the downstream boundary of the same links. Before the release of a pending downstream event, more upstream events may occur, triggering additional pending events. We store the pending downstream events in a FIFO data structure, with \(\zeta_a\) indicating the number of pending downstream events for each link \(a \in \mathcal{A}\). A travel time event enqueues a pending downstream event due to the slope change of the TTF. A downstream event occurs when the first of the pending downstream events is released, leading to a transition on the downstream boundary state. The node model then transmits this transition to the upstream boundary of the succeeding links. Such state changes on the upstream boundaries yield the corresponding pending downstream events.

Typically, the number of events grows exponentially as we propagate the flow through the network. We can omit the events carrying insignificant state changes by establishing a flow rate difference threshold, \(\Delta\), as suggested by Raadsen, Bliemer, and Bell (2016). An event is, then, verified only if it carries a flow difference higher than \(\Delta\). Even though the establishment of a flow rate difference threshold reduces the number of computations and consequently, the required running time, it might violate the flow conservation. Thus, we should use a small value for \(\Delta\) (less than \(10^{-3}\) veh/s) to expect a sufficiently low loss of vehicle hours. Nevertheless, this value might vary depending on the DNL application and the size and level of congestion of the
Algorithm Description

We initialise the algorithm at $t = 0$, by setting the indices of upstream and downstream events, $j_a$ and $z_a$, respectively, to zero for every link $a \in \mathcal{A}$. The index of the demand period, $h$, is also set to zero. Accordingly, we set the index of the travel time linear segment, $i_a$, to one for each link $a \in \mathcal{A}$ (we consider the first linear segment at $t = 0$).

The candidate time for a demand event is set to $h \cdot T_h$. If it is the minimum among the three candidate event times, a demand period transition occurs, and the demand period counter is increased by one. A demand period transition causes an upstream event, $j$, at time $t_j$, and on the upstream boundary of the first link of each route. We can compute the new inflow rate of every such link, $b$, by recalling Eq. 14 and Eq. 15:

$$u_{b,j} = \begin{cases} 
\sum_{a \in \mathcal{A}_a^-} \bar{q}_{a,b}(t_j) + \sum_k \delta_{b,k} \bar{f}_{k,h} T_h, & \text{if } n \in \mathcal{R}, \\
\sum_{a \in \mathcal{A}_a^-} \bar{q}_{a,b}(t_j), & \text{otherwise}, \quad b \in \mathcal{A}_a^+,
\end{cases}$$

(31)

where $\bar{q}_{a,b}(t_j)$ are the realised transition flows from link $a$ to link $b$ at the moment of the event. We will return to the description of their determination later in this section. Furthermore, such change on the upstream boundary triggers the $\xi$th pending downstream event that carries a pending outflow rate $v'_{b,\xi}$, with

$$v'_{b,\xi} = \frac{u_{b,j}}{\alpha_{b,i} + 1}.$$  

(32)

Moreover, the pending downstream event time, $\bar{t}_{b,\xi}'$, is set to

$$\bar{t}_{b,\xi}' = (\alpha_{b,i} + 1)t_{b,j} + \beta_{b,i}.$$  

(33)

The time instants, $\theta_{b,i+1}$, indicate the time when the next transition of LTT will occur for every individual link $b$, $b \in \mathcal{A}$. The earliest awaiting LTT transition indicates the candidate time for a travel time event. Let us assume that this is also the earliest candidate event time (between the three event types) and that it is associated with link $b$. Then, a travel time event occurs, and we refer to link $b$ as event link. Next, we increase the LTT index, $i_b$, by one, while we enqueue a pending downstream event. The carrying pending outflow and time can be found via Eq. (32) and Eq. (33), respectively.

The pending time, $\bar{t}_{a,1}'$, indicates the release time of the first pending downstream event in the queue of each link $a \in \mathcal{A}$. As the candidate downstream event time we consider the minimum among the pending event times $\bar{t}_{a,1}'$. Let us now assume that an upstream event occurs at $t = \bar{t}_{a,1}'$ and $a$ is the event link. The outflow index $z_a$ is increased by one, we change the outflow, $v_{a,z} = v'_{a,1}$, while the downstream event time is set to $\bar{t}_{a,z} = \bar{t}_{a,1}'$. Furthermore, we dequeue the first pending event, and we reduce the number of pending events by one. Moreover, such a change of state on the downstream boundary prompts a change on the upstream boundary of each succeeding link $b$, $b \in \mathcal{A}_a^+$. Accordingly, for each $b \in \mathcal{A}_a^+$, we increase the downstream event index, $j_b$, by one, while the new inflows $u_{b,j}$ are computed via Eq. (31). We

$^5$Note that the indices $i, j, \zeta$ are also link specific, i.e. $i_b, j_b, \zeta_b$, but we omit the link sub-index to avoid confusion.
do not consider any node delay and hence, the change on the upstream boundary of succeeding links, \( b \), occurs at the same time with the change on the downstream boundary of the proceeding link \( a \). Therefore, \( t_{b,j} = t_{a,z} \).

**Multi-commodity Approach**

The transition flows in Eq. (31) are given as

\[
\bar{q}_{a,b}(t_{b,j}) = \bar{q}_{a,b}(\bar{t}_{a,z}) = \mu_{a,b}(\bar{t}_{a,z})v_{a,z}, \quad a \in A_n^-, b \in A_n^+, n \in N,
\]

where \( \mu_{a,b}(\bar{t}_{a,z}) \) are the splitting rates at the time of the downstream event \( z \). One could use fixed splitting rates per demand period (Gentile et al. 2010; Raadsen, Bliemer, and Bell 2016), namely

\[
\mu_{a,b}(\bar{t}_{a,z}) = \mu'_{a,b,h}, \quad \text{if } (h - 1)T_h \leq \bar{t}_{a,z} < h'T_h, \quad a \in A_n^-, b \in A_n^+, n \in N,
\]

where \( \mu'_{a,b,h} \) are the demand period-specific splitting rates with

\[
\mu'_{a,b,h} = \frac{\sum_k \delta_{a,k} \delta_{b,k} \bar{f}_{k,h}}{\sum_k \delta_{a,k} \bar{f}_{k,h}}, \quad a \in A_n^-, b \in A_n^+, n \in N, h' \in H.
\]

However, in this paper, we adopt the multi-commodity approach suggested by (Raadsen and Bliemer 2019). Although fixed transition ratios constitute an alternative that requires lower computing resources, it may lead to route flow inconsistencies. Part of the route volume that has departed during the \( h^{th} \) demand period may be still present in the network at the \( h + 1 \)st (or at a later one) demand period. Hence, fixed transition ratios may direct such vehicles to different than their initial destinations. According to the multi-commodity approach, the information on the mixture distribution - which in this case regards demand period departure mixture - is not fixed but propagates in parallel with the flow.

Let \( \lambda_{b,j,h} \) be the portion of the inflow of link \( b \) at the \( j^{th} \) upstream event that belongs to demand period \( h \). Note that the sum \( \sum_h \lambda_{b,j,h} \) should be equal to one. This mixture distribution will arrive to the downstream boundary of the same link \( \tau_b(t_{b,j}) \) time units later, alongside the traffic state perturbation occurred at event \( j \). Therefore, we use the same FIFO data structure for storing the pending mixture distributions. The \( \zeta^{th} \) pending downstream mixture distribution is denoted by \( \bar{\lambda}_{b,z} \) and equals the upstream distribution \( \lambda_{b,j} \). Then, such pending mixture is released at the \( \zeta^{th} \) pending event time (Eq. (33)). Similar to the flow propagation, once an upstream event is released, we change the downstream mixture propagation to \( \bar{\lambda}_{b,z} = \bar{\lambda}_{b,1} \). Then, we utilise such mixture distribution to calculate the realised splitting ratios at the \( z^{th} \) upstream event as

\[
\mu_{a,b,z} = \sum_{h' \in H} \bar{\lambda}_{b,z,h'} \mu'_{a,b,h'}, \quad a \in A_n^-, b \in A_n^+, n \in N.
\]

Accordingly, the mixture distribution on the upstream boundary of the succeeding links, \( b \), occurs at the same time with the change on the downstream boundary of the proceeding link, \( a \). Therefore,

\[
\bar{t}_{b,j} = \bar{t}_{a,z}.
\]

---

6Note that \( \lambda \) could also carry some more mixture distribution information, e.g. route-specific information.

7We omit sometimes the demand period index, the mixture distributions contain information for every \( h' \in H \).
links, $b \in \mathcal{A}_u^+$, is given by

$$
\Delta_{b,j,h'} = \begin{cases} 
\left( \frac{\sum_{a \in \mathcal{A}_n^-} \bar{q}_{a,b,h'} + \sum_{k} \delta_{b,k} \frac{f_{b,h'}}{T_k}}{u_{b,j}}, \right) / u_{b,j}, & \text{if } n \in \mathcal{R}, \ h' \in \mathcal{H}, \\
\left( \frac{\sum_{a \in \mathcal{A}_n^-} \bar{q}_{a,b,h'}}{u_{b,j}}, \right) / u_{b,j}, & \text{otherwise, } h' \in \mathcal{H}, 
\end{cases} \tag{38}
$$

where $\bar{q}_{a,b,h'}$ are the demand period-specific transition flows,

$$
\bar{q}_{a,b,h'} = \bar{\lambda}_{h,z,k} \mu_{a,b,h'} v_{a,j}, \quad a \in \mathcal{A}_n^- , b \in \mathcal{A}_n^+, n \in \mathcal{N}, h' \in \mathcal{H}. \tag{39}
$$
Algorithm 2: Event-based network loading

Input: $G(V, A), \tau_a, f_a, T, \Delta$, $a \in A, h' \in H$

Output: $u_{a, t}, \ell_{a, t}, \eta_{a, t} \forall a \in A$

Initialization: $t = 0, h = 0, \eta = hT_b, B = \emptyset, \tilde{q}_{a, b} = 0, a \in A, b \in A^+, n \in N$, $i_a = 1, j_a = 0, z_a = 0, \zeta_a = 0, v_a = 0, \lambda_{a, h'} = 0, a \in A, h' \in H$

while $t < T$
do

\[ h := h + 1; \] // update the demand period counter
for $b \in A^+_n, n \in R$, if $\sum \bar{I}_{k, h} \delta_{b, k} - \sum \bar{I}_{k, h-1} \delta_{b, h} \geq \Delta$ do (check $\Delta$ for each event link $b$)

Add an upstream event as:
\[ j_b := j_b + 1; \] // update the upstream event counter
\[ \bar{W}_{h, j_b} = t; \] // upstream event time
find $q'_{a, b} v_a \in A^+_n$ via Eq. (39); // demand period-specific sending flows
find $u_{a, j_b}$ and $\tilde{y}_a$ via Eq. (31) and Eq. (38); // upstream flow and mixture
Enqueue a pending downstream event as:
\[ \zeta_b := \zeta_b + 1; \] // update the pending event counter
\[ \bar{p}_{b, \zeta_b} := (\bar{a}_{b, \zeta_b} + 1)\bar{W}_{h, j_b} + \delta_{b, i_b}; \] // pending downstream event time
\[ v'_{a, \zeta_b} := u_{a, j_b} / (\bar{a}_{b, \zeta_b} + 1), \tilde{x}_{b, \zeta_b} := \tilde{y}_b; \] // pending downstream flow and mixture
\[ B := B \cup b; \] // update the set of links with pending events

if $t = \eta$ then Demand event
\[ \text{for } a \in \{ a \in A | \theta_{a, i_a + 1} = \eta \} \text{ do (for each event link $b$)} \]
\[ i_a := i_a + 1; \] // update the LTT period index
Enqueue a pending downstream event;

else if $t = \tilde{\eta}$ then Downstream event
\[ \text{for } a \in \{ a' \in B | \theta_{a', i_a + 1} = \tilde{\eta} \} \text{ do (for each event link $a$)} \]
Add a downstream event as:
\[ z_a := z_a + 1; \] // update the downstream event counter counters
\[ \bar{L}_{a, i_a} = t; \] // downstream event time
\[ v_a, z_a = v'_{a, \zeta_b, \zeta_a} = \tilde{x}_{a, \zeta_b, \zeta_a}; \] // downstream flow and mixture
Dequeue a pending downstream event as:
\[ \zeta_b := \zeta_b + 1; \] // update the pending event counter
\[ v'_{a, \zeta_b} := \bar{L}_{a, \zeta_b} + 1, \tilde{x}_{a, \zeta_b} := \tilde{x}_{a, \zeta_b}; \] // dequeue the event
\[ \text{else} \]
\[ B := B \setminus a; \] // update the set of links with pending events

Node model:
find $n$ such that $a \in A^-_n$; // find the downstream node of $a$
find $\mu_{a, b} \forall b \in A^+_n$ via Eq. (37); // splitting rates
\[ \tilde{q}_{a, b} := \mu_{a, b}, \bar{q}_{a, b}; \] \[ b \in A^+_n; \] // sending flows
for $b \in A^+_n$ if $\tilde{q}_{a, b} \geq \Delta$ do (check $\Delta$ for each succeeding link $b$)
Add an upstream event;
Enqueue a pending downstream event;

end

$\text{end}$

$\text{end}$

4. Results

In this section, we test the performance of the DDNL based on simulation-based experiments. We employ a microscopic simulator to generate specific traffic patterns, which we assume that represent the reality. We want to examine if the DDNL is able to reproduce such patterns relying only on the travel time of a sub-set of vehicles and
Table 1.: Simulation experiment 1 - Corridor network

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulation time, $T$</td>
<td>9000 s</td>
</tr>
<tr>
<td>Demand period duration, $T_h$</td>
<td>900 s</td>
</tr>
<tr>
<td>Aggregation period of the output, $T'_n$</td>
<td>300 s</td>
</tr>
<tr>
<td>Total demand, $\sum_h \sum_k f_{k,h}$</td>
<td>3476 veh</td>
</tr>
</tbody>
</table>

without considering any jam-density or capacity constraint. We evaluate the accuracy of our approach by comparing the results of DDNL against the results of a microscopic simulator, which we assume as the ground-truth. Our main aim is to investigate how DDNL behaves under congested conditions. Furthermore, considering that DDNL is free of any supply constraint, we want to examine if, and under which circumstances jam-density and capacity are violated.

Initially, we test the DDNL on a corridor-network where there is only one route available between each OD pair. We select a corridor-network to more illustratively show how DDNL behaves in congestion and examine its ability to describe dynamic traffic flow phenomena such as spill-back. Next, we consider a small toy-network, where there is more than one route available for each OD pair and vehicles from different OD pairs merge in a link.

4.1. **Simulation Experiment 1: Corridor-network**

The morning peak traffic (from 8:00 to 10:30 divided into ten demand periods) is simulated for a 10 km long road stretch using the AIMSUN microscopic traffic simulator (Barceló and Casas 2005). The road stretch consists of 14 equal-length homogeneous links (see Figure 5(a)). Traffic flows at one direction for each link, while the number of lanes drops from two to one for links 4, 8 and 13, reducing their capacity to the half of its initial value. By adjusting the route volumes, we set the inflow at the tail of link 1 to be higher than the one-lane capacity, and hence link 4 becomes an activated bottleneck. Moreover, we adjust the additional traffic volume that comes from the two on-ramps located at the tail of links 6 and 11, such way that it triggers the activation of the other two bottlenecks (link 8 and 13). The route volumes (demand) are not constant but varied among the ten demand periods (see Figure 4). Table 1 presents the main parameters of this first simulation experiment.

Given such parameters, we ran one random replication in AIMSUN micro-simulator, assuming that the values of the micro-simulation stochastic variables represent the morning peak traffic of a random day. Figure 4 illustrates how the demand is distributed among the demand periods and the simulated density profile for a typical link of the network. Then, we took the average simulated inflows and densities for each 5-minutes period indexed by $n$, and each link, $a$, to construct our ground-truth data-set, \{$(u_{a,n}^{(sim)}, \rho_{a,n}^{(sim)})$\}. Figure 5 (b) illustrates how the ground-truth density evolves over time for each link. As the demand is higher than the bottlenecks’ capacity, the remaining vehicles are stacked in queues raising the density on the links upstream the bottlenecks. At some point in time, such densities reach their maximum value, and the queues spill-back to further upstream links.
Figure 4.: Demand distribution and density profile.

Figure 5.: (a) The simulated corridor-network, (b) 5-minutes average density based on micro-simulation, (c) 5-minutes average density based on DDNL.
Setting the Experiment Parameters

For the same road stretch and given the same route volumes, we perform now the loading using the DDNL instead of the micro-simulator. We assume that a portion of the simulated vehicles is equipped with a sensing device able to provide link travel times, \( y^{(\text{obs})} \). Let \( N' \) be the total number of simulated vehicles. Given a penetration rate of the probe vehicles, \( \tilde{r} \), we randomly select \( \tilde{r} N' \) vehicles and extract their simulated link travel times. For each time instant, \( x_{a,i} \), when vehicle \( i \) enters link \( a \), we store its simulated link travel time. Then, we assume that the travel time observations from the sensing devices, \( y^{(\text{obs})}_{a,i} \), correspond to the simulated link travel times plus a random error term representing the observation noise. The error term, which is added to reflect the measurements noise, is assumed to be Gaussian distributed having zero mean and a variance of \( 3^2 \) seconds. Next, by taking the mean of all the travel times observed during one minute, we construct the input to Algorithm 1 data-set, \( \{x_{a,v}, y_{a,v}\}_{v \in \mathcal{L}} \), for each link \( a \in \mathcal{A} \).

We consider five different levels of probe vehicles penetration rate \( \tilde{r} \): 2, 5, 10, 20 and 90 per cent. The sampling of the \( \tilde{r} N \) vehicles is a random process and may influence the results, especially for low penetration levels. Hence, for each penetration level, we run 20 replications, indexed by \( s \). Moreover, to investigate the effects of the accepted error parameter, \( \tilde{\varepsilon} \), on the efficiency and accuracy of the DDNL, we consider four different values: 2, 5, 10 and 20 seconds. Therefore, we estimate the link travel function 20 times for every combination of \( \tilde{r} \) and \( \tilde{\varepsilon} \). Finally, for each aggregation period, \( n \) (\( T'_n = 5 \) minutes), we estimate the average link densities, \( \rho^{(\text{DDNL})}_{a,n,\tilde{r},\tilde{\varepsilon},s} \) and inflows, \( u^{(\text{DDNL})}_{a,n,\tilde{r},\tilde{\varepsilon},s} \). During all of our experiments, the flow rate difference threshold, \( \Delta \), is set to \( 10^{-4} \) veh/s.

We use a Root Mean Square Error (RMSE) as a measure of the differences between the simulated and the DDNL densities. We choose density as the reference variable because its estimation implicitly uses the link inflows and outflows. Thus, for each aggregation level, accepted error value, penetration rate level and replication, the RMSE is computed as

\[
RMSE_{\tilde{\varepsilon},\tilde{r},s} = \sqrt{\frac{\sum_a \sum_n (\rho^{(\text{DDNL})}_{a,n,\tilde{r},\tilde{\varepsilon},s} - \rho^{(\text{sim})}_{a,n})^2}{|\mathcal{A}| \cdot T_n/T_n'}}. 
\]  

(40)

Numerical Results

Figure 5 (c) illustrates the DDNL densities for a 5 per cent penetration rate and a accepted error of 5 seconds. Note that although density fluctuates more compared to the ground-truth at the links upstream the bottlenecks (links 3, 7 and 12), it remains close to jam-density. Moreover, the spill-back effect is sufficiently described as the low values of density are propagated to the upstream links. However, there are still some discrepancies between the ground-truth and DDNL, which also seem to propagate downstream.

The box-plot illustrated in Figure 6 displays how the RMSEs are distributed among the 20 replications. The boxes contain the data points that span the Interquartile Range (IQR), i.e. the points between the 25th (Q1) and 75th (Q3) per centiles of the sample data. The wide horizontal lines illustrate the median, while the dots represent

---

8The densities are calculated by dividing the volume, \( q_a(t) = U_a(t) - V_a(t) \), of a link \( a \) by its length.
the outliers (the data points that lie above Q3 + 1.5IQR or below Q1 − 1.5IQR). As we expected, RMSE generally gets higher while the value of the accepted error, \( \tilde{\varepsilon} \), increases. Regarding the penetration rates, RMSE is higher and more widespread for the low penetration rate levels, especially for \( \tilde{r} = 2\% \). The sparse travel time observations coming from the 2 per cent of the vehicles may not provide sufficient information for the estimation of the TTF. The penetration rate of 2 per cent also shows a high variance among the replications, indicating the importance of the random vehicle sampling. However, we observe a noticeably improved performance when 5 per cent of the vehicles contribute to the TTF estimation (which we believe is a realistic scenario). Finally, the influence of the penetration rate to the RMSE seems to decline for penetration rates greater than or equal to 10 per cent (possible future scenarios).

One more aim of this section is to investigate if the DDNL results in severe violations of capacity and jam density. Figure 7 and 8 illustrate how each ground-truth inflow and density value (x-axis) relates to the corresponding one modelled by the DDNL (y-axis). They depict the intensity of the points \( \{u_{a,n}^{(\text{sim})}, u_{a,n,\tilde{\varepsilon},\tilde{r},s}^{(\text{DDNL})}\} \) and \( \{\rho_{a,n}^{(\text{sim})}, \rho_{a,n,\tilde{\varepsilon},\tilde{r},s}^{(\text{DDNL})}\} \) for every link \( a \) and replication \( s \), for an accepted error of 5 seconds, and three indicative penetration rate levels: 2, 10 and 90 per cent. While Figure 6 presents average differences, Figures 7 and 8 illustrate how the discrepancies between DDNL and ground-truth are distributed for different levels of inflows and densities. Even though DDNL shows a good performance for low inflows and densities, the divergences are higher for values of inflow close to capacity and density close to jam density. Furthermore, some points lay above the extreme values of flow or density, especially for the penetration rate of 2 per cent. The number and the amplitude of the capacity or jam density violations are significantly decreased when 10 per cent of the vehicles are employed to provide travel time information. At the same time, there are not so pronounced differences between the 10 per cent and 90 per cent penetration level.

\[9\text{We select the accepted error to be 5 seconds since this value shows the best trade-off between accuracy and efficiency for this network.}\]
Figure 7.: Ground-truth vs DDNL inflow in veh/(5 min × lane) for a penetration rate of (a) 2%, (b) 10%, (c) 90%.

Figure 8.: Ground-truth vs DDNL density in veh/(km × lane) for a penetration rate of (a) 2%, (b) 10%, (c) 90%.
4.2. Simulation Experiment 2: Small Toy-network

We include an experiment considering the toy-network in Figure 9 to investigate how DDNL behaves when volumes from several OD pairs can merge in the same link. The network consists of 96 links, 33 nodes and 10 zone centroids (the rectangles from A to J) that attract and/or generate trips. There are 89 OD pairs in the network, which are connected by 199 routes in total.

The morning peak traffic (from 8:00 to 10:00 divided to 8 demand periods) for this toy-network is also simulated using AIMSUN. We consider three different demand levels (DLs) to investigate the robustness of our method and the effect of congestion on the accuracy and efficiency of the DDNL (see Table 2). The first demand level (DL1) leads to moderate congestion (short queues at individual links), DL2 adds some more congestion (the density reaches its maximum value for some links), while congestion becomes more severe due to DL3 (the spill-back effect is activated). Furthermore, we add some variations among the demand periods to investigate how the DDNL behaves under congestion variations within the analysis period. The demand portions of each period, which are the same for the three DLs, are illustrated by Figure 10. Figure 10 also shows the density profile for a typical link due to the three different DLs. The route volumes, $\bar{f}_{k,h}$, are determined by the micro-simulator utilising the one-shot simulation stochastic route choice module of AIMSUN and constitute an input to the DDNL.

We use the same experiment settings as in experiment 1. Regarding the evaluation criteria, we employ the same RMSE (see Eq. (40)), considering, though, only the 20 most congested links. We select the 20 links with the higher average simulated density to contribute to the RMSE because some of the network’s links are uncongested. This way, we can avoid any bias caused by the uncongested links and more illustratively show the differences between the three different DLs. The set of the most congested links is the same for every DL.

**Numerical Results**

Figure 11, 12 and 13 illustrate how the RMSE is distributing among the 20 replications for this toy-network and the three different DLs. We can observe a similar pattern in Figures 11 and 12; DL2 shows a slightly higher RMSE due to the increased congestion.
Table 2.: Simulation experiment 2 - Small Toy-network

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulation time, $T$</td>
<td>7200 s</td>
</tr>
<tr>
<td>Demand period duration, $T_h$</td>
<td>900 s</td>
</tr>
<tr>
<td>Aggregation period of the output, $T_n'$</td>
<td>300 s</td>
</tr>
<tr>
<td>Total demand, $\sum_k \sum_h \hat{f}_{k,h}$</td>
<td>DL 1 8352 veh</td>
</tr>
<tr>
<td></td>
<td>DL 2 8858 veh</td>
</tr>
<tr>
<td></td>
<td>DL 3 9707 veh</td>
</tr>
</tbody>
</table>

Figure 10.: Demand distribution and density profiles.

However, the RMSE is generally low, while the impact of the penetration rate and the accepted error is weakened. DL1 and DL2 yield moderate variations of the traffic conditions, which can sufficiently be described using having less information.

The effect of the penetration rate as well as the accepted error is more profound for the DL3. Figure 13 shows that the information provided by the 2 per cent of the vehicles is insufficient for accurately propagating the flow. The penetration of 2 per cent leads to RMSEs higher than the marginal error threshold of 15 veh/km, for every value of the accepted error. The accuracy is noticeably improved for the 5 per cent penetration rate, but the RMSE reaches its corresponding values of DL1 and DL2 when more than 10 per cent of the vehicles are employed. Our findings also suggest that for such over-congested networks, the accepted error should take the lower possible value.

As we mention in Subsection 3.2, the value of the accepted error in Algorithm 1

Figure 11.: Distribution of the density RMSE in veh/(km × lane) among the 20 replications and for the DL1.
Figure 12.: Distribution of the density RMSE in veh/(km × lane) among the 20 replications and for the DL2.

Figure 13.: Distribution of the density RMSE in veh/(km × lane) among the 20 replications and for the DL3.
Figure 14.: Number of events and average RMSE for the three different DLs and for $\tilde{r} = 10\%$. Note the non-linearity of x-axis values

(a) (b) (c)

Figure 15.: Ground-truth vs DDNL inflow in veh/(5 min × lane) for (a) DL1, (b) DL2, (c) DL3.

affects the accuracy but also influences the efficiency of the DDNL. Low values of $\tilde{\varepsilon}$ tend to generate a high number of linear segments in the piece-wise linear TTF. This correspondingly leads to an increased number of events while flow propagates through the network. Figure 14 illustrates the effect of accepted error on the number of events and the average RMSE for a 10 per cent penetration rate. Note that the number of events is analogous to the total running time of Algorithm (2). The average time for performing the computations required for one event is $1.96 \cdot 10^{-6}$ seconds.

When congestion is moderate (DL1 and DL2), low values of $\tilde{\varepsilon}$ deteriorate the efficiency without significantly improving the accuracy. We can partly observe this phenomenon in DL3, but only for accepted error values lower than 2 seconds. Nevertheless, what Figure 14 evidently shows is that the number of events depends on the congestion level.

Figure 15 shows that the DDNL inflows follow the simulated ones for almost every DL and flow level. Note the negligible effect of the DL in the inflows’ distribution. We notice a different pattern for the densities (Figure 16), where the influence of the DL is stronger. The values of density are more sensitive being obtained directly from the cumulative number of vehicles, while flows denote the cumulative vehicles’ rate of change. The results between different replications tend to diverge for severe congestion, and this becomes more pronounced when we look at the densities. Figure 16(c) also shows that although DL3 results into some violations of jam density, most of the DDNL density points coincide with the corresponding ground-truth points.

\[^{10}\text{64-bit Matlab 2018b running on a desktop: 16 GB RAM, Intel Core 3.60 GHz, Windows10.}\]
5. Discussion

Let us now discuss our main findings and highlight some limitations of this study. The results presented in this paper indicate that the DDNL can accurately capture the spatio-temporal variations of congestion, and thus they provide a proof of the concept. Nevertheless, our experiments presented are performed in the fully controllable environment of micro-simulation. We assume that a normally distributed, zero-mean measurement noise vector (to which we give an arbitrary variance value) is incorporated in the travel time observations. The availability of travel time observations is also controlled by adjusting the penetration rate, assuming that such availability is constant for the entire analysis period. However, the reliability and availability of the travel time observations in reality are highly questionable.

Although the DDNL shows a satisfying performance for high penetration rates, we can still observe some errors even when the travel time estimation is based on observations from 90 per cent of the vehicles. Therefore, even if almost flawless travel time information is available, the DDNL may result in errors due to discretisation in space and travel time aggregations. We assume that vehicles have a constant speed throughout their travelling time on a single link, a simplification which may deteriorate the accuracy. Furthermore, in our experiments, we aggregate the probe vehicle travel times over one minute to reduce the running time of Algorithm 1. Alternatively, one could consider the travel time of each single probe vehicle as an input point to Algorithm 1, increasing, however, its complexity.

We neglect an analysis regarding the running time of Algorithm 1 in Section 4 because it has a negligible effect on the total loading time in our experiments. The networks considered in both simulation experiments consist of a limited number of links. However, Algorithm 1 may require a considerably higher computational effort for more extensive networks. Applying Algorithm 1 means that regardless of the congestion level, Problem (24) has to be solved $N - 1$ times for each link. Possibly, more efficient algorithms than Algorithm 1 can be developed by growing the reference window, based for instance, on the variance of the travel time estimates. We assume here that the efficiency and accuracy of Algorithm 1 are adequate for the aims of this paper. The development and comparison of various segmented regression approaches are beyond the scope of this paper. Any segmentation algorithm that leads to a piece-wise linear representation of link travel time can be integrated into the DDNL.

The computational effort required for running the DDNL is almost entirely occupied
by Algorithm 2. We use a rather low flow rate difference threshold, \( \Delta \), to eliminate the information loss, generating, though, a large number of events. Still, the total loading time is quite low ranging from 0.09 to 0.27 seconds for the toy network of Experiment 2. Apparently, a higher \( \Delta \) value may be required for larger networks. Raadsen and Bliemer (2019) provide an analysis where they relate the value of flow difference threshold with the computational efficiency and the information loss. The flow difference threshold is a trade-off parameter to be adjusted by the user, depending on the available computing resources and the application’s requirements of exactness. Our approach allows the user to reduce the computational effort further, and compromise on the accuracy by adjusting the accepted error, \( \tilde{\varepsilon} \).

We do not include any comparison with the traditional network loading models, such as the CTM or LTM, because our method works in a different context. It is a data-driven approach that requires field-data as input instead of demand-supply relationships. The comparison against any other data-driven network-wide approach would make more sense, and this is something we consider for future research. Regarding the efficiency evaluation of the DDNL, the advantage of the event-based approaches over the grid-based methods with fixed-time intervals (CTM, LTM) has already been demonstrated by Raadsen, Bliemer, and Bell (2016). Our data-driven event-based approach can be further alleviated in terms of computational times due to the simplified nature of the link and node model.

Let us finally remark that the DDNL is developed for monitoring historical traffic flows. Our main aim is to reproduce a traffic pattern based on partially observed field-data. The DDNL can be integrated into DTA modelling frameworks used for applications that require network-wide historical flows, such as emission estimations, or short-term planning (if we assume that for a shot-time period the variations of the demand or supply are not sufficient to cause a travel time change). Similar to any data-driven approach, the DDNL cannot capture long-term changes in the demand or supply of a network.

One more possible application of the DDNL may be the calibration of dynamic OD demand matrices. The OD calibration problem is typically formulated as an optimisation problem, which by adjusting the demand matrices seeks to minimise the difference between the modelled (from DTA) and the observed flows on a subset of the network’s links. If travel time observations are also available for the same analysis period, the network loading sub-model could be data-driven, establishing some important linearities between the demand and the link flows. We should highlight here that OD calibration requires route-disaggregated link flows. Therefore, the mixture distribution should also carry route-specific information, which, however, increase the number of computations.

In the case where real-time traffic data are available (e.g., link speeds), the DDNL may also contribute to applications such as the on-line OD demand calibration. Sliding window algorithms can also be applied on-line. The same holds for Algorithm 1, which can provide the parameters of the linear LTT as well as the breakpoints in real-time. Then, the event-based Algorithm 2 can give on-line traffic flow estimations or short-term predictions. At each time instant, \( x_{a,\nu} \), when the \( \nu^{th} \) travel time observation is available for link \( a \), we can solve the constrained least square Problem (24) in real-time. If the estimation error is greater than the predefined error threshold, \( \tilde{\varepsilon} \), a slope change in the TTF occurs, which accordingly triggers a travel time event. Converting the DDNL to an on-line approach is a consideration for future research.
6. Conclusions and Future Research Directions

In this paper, we present and evaluate a data-driven approach for dynamic network loading. Our DDNL estimates network-wide dynamic link flows and volumes having as input dynamic route volumes and travel time observations. In contrast to the typical DNL approaches, congestion is an exogenous parameter in the flow propagation model. In this way, the DTA problem is decomposed, establishing some important linearities between the demand and the link flows. Our approach is continuous in time and does not depend on any infrastructure attribute, while FIFO, vehicle conservation and non-negativity of flow are always ensured. Furthermore, our modelling structure yields an event-based grid-free formulation, reducing the required number of computations.

The first simulation experiment-based results are quite encouraging. By having dynamic link travel time information from a random subset of vehicles, we were able to satisfactory capture congestion and dynamic traffic flow phenomena such as spill-back. Future work may include field experiments to evaluate the accuracy of the DDNL under real traffic conditions and for larger networks. The development of a data-driven route choice estimation approach to be coupled with the DDNL is also necessary to apply the approach to real applications. Then, the resulting data-driven network assignment has the potential of being the core of a data-driven OD calibration process.

Acknowledgement

The authors are grateful for the constructive comments from three anonymous referees.

Disclosure Statement

The authors declare that there are no conflicts of interest.

Funding

This project was mainly funded by the Swedish Energy Agency, project grant number 46963-1.

References


Astarita, Vittorio. 1996. “A continuous time link model for dynamic network loading based on travel time function.” In Transportation and traffic theory. Proceedings of the 13th


Jing, Boyu, Lin Wu, Hongjun Mao, Sinning Gong, Jianjun He, Chao Zou, Guanghua Song, Xiaoyu Li, and Zhong Wu. 2016. “Development of a vehicle emission inventory with high temporal–spatial resolution based on NRT traffic data and its impact on air pollution in


Toole, Jameson L, Serdar Colak, Bradley Sturt, Lauren P Alexander, Alexandre Evsukoff, and Marta C González. 2015. “The path most traveled: Travel demand estimation using big


