Transportation Research Part C 119 (2020) 102747

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Estimating multi-class dynamic origin-destination demand through a forward-backward algorithm on computational graphs☆

Wei Ma [a], Xidong Pi [b], Sean Qian [b,c,*]

[a] Department of Civil and Environmental Engineering, The Hong Kong Polytechnic University, Hong Kong
[b] Department of Civil and Environmental Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, United States
[c] H. John Heinz III Heinz College, Carnegie Mellon University, Pittsburgh, PA 15213, United States

ARTICLE INFO

Keywords:
O-D estimation
Machine learning
Neural network
Dynamic networks
Multi-source data

ABSTRACT

Transportation networks are unprecedentedly complex with heterogeneous vehicular flow. Conventionally, vehicles are classified by size, the number of axles or engine types, e.g., standard passenger cars versus trucks. However, vehicle flow heterogeneity stems from many other aspects in general, e.g., ride-sourcing vehicles versus personal vehicles, human driven vehicles versus connected and automated vehicles. Provided with some observations of vehicular flow for each class in a large-scale transportation network, how to estimate the multi-class spatio-temporal vehicular flow, in terms of time-varying Origin-Destination (OD) demand and path/link flow, remains a big challenge. This paper presents a solution framework for multi-class dynamic OD demand estimation (MCDODE) in large-scale networks that work for any vehicular data in general. The proposed framework cast the standard OD estimation methods into a computational graph with tensor representations of spatio-temporal flow and all intermediate features involved in the MCDODE formulation. A forward-backward algorithm is proposed to efficiently solve the MCDODE formulation on computational graphs. In addition, we propose a novel concept of tree-based cumulative curves to compute the exact multi-class Dynamic Assignment Ratio (DAR) matrix. A Growing Tree algorithm is developed to construct tree-based cumulative curves. The proposed framework is examined on a small network, a mid-size network as well as a real-world large-scale network. The experiment results indicate that the proposed framework is compelling, satisfactory and computationally plausible.

1. Introduction

Transportation networks are unprecedentedly complex with heterogeneous vehicular flow. Conventionally, vehicle heterogeneity are considered in terms of vehicle classifications (such as standard passenger cars and trucks). However, vehicle flow heterogeneity stems from many other aspects in general, e.g., ride-sourcing vehicles versus personal vehicles, human driven vehicles versus connected and automated vehicles. How to effectively estimate and manage the multi-class vehicles in a complex transportation system so as to improve the network efficiency presents a big challenge. As an indispensable component of dynamic transportation network models with heterogeneous traffic, the multi-class dynamic origin-destination (OD) demand plays a key role in transportation planning

☆ This article belongs to the Virtual Special Issue on Machine learning.

* Corresponding author at: Department of Civil and Environmental Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, United States.
E-mail addresses: wei.w.ma@polyu.edu.hk (W. Ma), xpi@alumni.cmu.edu (X. Pi), seanqian@cmu.edu (S. Qian).

https://doi.org/10.1016/j.trc.2020.102747
Received 10 June 2020; Accepted 5 August 2020
Available online 27 August 2020
0968-090X/© 2020 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

and management to understand spatio-temporal vehicular flow and its travel behavior. To our best knowledge, there is a lack of studies to understand and estimate the dynamic OD demand of multiple vehicle classes using sparse and partial flow observations. In view of this, this paper presents a data-driven framework for multi-class dynamic OD demand estimation (MCDODE) in large-scale networks. The MCDODE formulation is represented on a computational graph, and a novel forward-backward algorithm is proposed to estimate the OD demand efficiently and effectively. The proposed MCDODE framework is examined in a small networks, a mid-size network as well as a real-world large-scale network to demonstrate the estimation accuracy and the computational efficiency.

The multi-class dynamic OD demand (MCDOD) represents the number of vehicles in each of general vehicle classes (*e.g.* personal cars, trucks, ride-sourcing vehicles, connected vehicles, etc.) departing from an origin and heading to a destination in a particular time interval. The definition of "classes" is very general, by vehicle sizes, specifications, and nature of trips. The MCDOD reveals the fine-grained traffic demand information for different vehicle classes and the overall spatio-temporal mobility patterns can be inferred from the OD demand and its resultant path/link flows. Policymakers can understand the departure/arrival patterns of multi-class vehicles through the MCDOD. The MCDOD also helps the policymakers understand the impact of each vehicle class on the roads, and hence each class of vehicles can be managed separately. In addition, most of Advanced Traveler Information Systems/Advanced Traffic Management Systems (ATIS/ATMS) would require accurate MCDOD as the model input (Huang and Li, 2007).

The dynamic OD estimation (DODE) has been extensively studied over the past few decades. A generalized least square (GLS) formulation is proposed for estimating dynamic OD demand with exogenous route choice model (Cascetta et al., 1993). The GLS formulation is further extended to a bi-level optimization problem, and the bi-level formulation solves for the DODE with endogenous route choice model (Tavana, 2001). Advantages and disadvantages of the bi-level formulation are discussed in Nguyen (1977), LeBlanc and Farhangian (1982), Fisk (1989), Yang et al. (1992), Florian and Chen (1995), Jha et al. (2004). Zhang et al. (2017), Osorio (2019) proposed an instantaneous approximation for the dynamic traffic assignment models, and the DODE problem is cast into a convex optimization which can be solved efficiently. The bi-level formulation can also be relaxed to a single-level problem and solved by updating the OD demand using gradient-based methods (Nie and Zhang, 2008; Lu et al., 2013).

The DODE problem can be viewed as a statistical estimation problem. For example, a statistical inference framework using Markov Chain Monte Carlo algorithm was proposed to estimate the probabilistic OD demand (Hazelton, 2008). Zhou et al. (2003) proposed a DODE framework with multi-day data and established a hypothesis testing framework to identify the demand evolution within a week. Flötteröd et al. (2011) developed a Bayesian framework to update the OD demand. In addition to the off-line DODE methods, the DODE formulation can also be solved with real-time data streams for ATIS/ATMS applications. Bierlaire and Crittin (2004) extended the GLS formulation on large-scale networks and solved the formulation efficiently on a real-time basis. The state-space models are also used to estimate the dynamic OD of each time interval sequentially (Zhou and Mahmassani, 2007; Ashok and Ben-Akiva, 2000). Djukic et al. (2012) estimated the real-time OD matrix using Principal Component Analysis (PCA).

The DODE methods often encapsulate the dynamic traffic assignment (DTA) models within the bi-level formulation. Various DTA models can be adopted to model the network dynamics and travelers' behaviors. For example, Dynamic Network Loading (DNL) models (Ma et al., 2008) simulate the vehicle trajectories and traffic conditions given determined the origin-destination (O-D) demand and fixed routes; Dynamic User Equilibrium (DUE) models (Mahmassani and Herman, 1984; Nie and Zhang, 2010) search for the equilibrated travelers' route choices to achieve user optimum; Dynamic System Optimal (DSO) models (Shen et al., 2007; Qian et al., 2012; Ma et al., 2014; Zhang and Qian, 2020) explore the optimal conditions under which the total network costs are minimized.

To solve the DODE problem, estimating the gradient of the objective functions with respect to the dynamic OD demand is essential. As the DNL models are complicated, obtaining the gradient analytically is challenging. Most of studies approximate the gradients of OD demand through finite differences by running DNL models multiple times. For example, the Stochastic Perturbation Simultaneous Approximation (SPSA) methods have been adopted to solve the bi-level formulation in many studies (Balakrishna et al., 2008; Cipriani et al., 2011; Lee and Ozbay, 2009; Vaze et al., 2009; Ben-Akiva et al., 2012; Cantelmo et al., 2014; Cantelmo et al., 2014; Lu et al., 2015; Tympakianaki et al., 2015; Antoniou et al., 2015; Cipriani et al., 2015; Cantelmo et al., 2018). Sensitivity analysis of the traffic assignment problem can also be adopted to evaluate the gradients of OD demand (Patriksson, 2004; Qian and Zhang, 2011; Qian et al., 2012; Zhang and Qian, 2020). In addition, heuristic iterative algorithms between lower and upper problem are widely used in traffic applications and can be viewed as a special case of sensitivity analysis (Yang, 1995). As the size of network increases dramatically in recent years, the finite differences based-approach becomes practically infeasible as the DNL model is computationally intensive. Recent studies start to focus on approximating the gradient with fewer simulation runs. Corthout et al. (2011), Frederix et al. (2011), Frederix et al. (2013) develop a Marginal Computation (MaC) approach to reduce the number of DNL runs by evaluating the local impact of the OD demand. Furthermore, there are studies estimating the gradients with one single DNL run. For example, Lu et al. (2013) estimates the gradient with respect to path flow using cumulative curves of link flows, and Osorio (2019) approximates the gradient by linearizing the dynamic traffic assignment models with a meta-model. In this paper, we develop a novel method to view and reformulate the general DODE problem through computational graphs following the previous work of Wu et al. (2018) on static networks with a single vehicle class, and we propose an alternative approach to linearly approximate the gradient of the objective functions with respect to the dynamic OD demand using the forward-backward algorithm on the computational graph.

To further highlight the differences between this study and other previous studies, we compare this study with Cascetta et al. (1993), Yang (1995), Balakrishna et al. (2008), Lu et al. (2013) separately.

- Cascetta et al. (1993) develops the concept of Dynamic Assignment Ratio (DAR) matrix. However, the DAR matrix is approximated using travel time information. In addition, it is not explicitly discussed how the GLS formulation can be solved in large-scale networks. In this study, we develop methods to evaluate the exact DAR matrix and the GLS formulation is cast into a computational graph to address the computational issues.
- Yang (1995) proposes a heuristic framework for DODE, which solves the equilibrium condition and the GLS formulation iteratively. The GLS formulation is solved with gradient methods, and the gradients are obtained from sensitivity analysis, which is computational impractical for large-scale networks. In this study, we approximate the gradients by linearizing the traffic equilibrium condition within each simulation using the DAR matrix, and the developed method can scale to large-scale networks.
- Balakrishna et al. (2008) proposes to use SPSA to approximate the gradient of the GLS formulation, while later we will show that the SPSA methods perform poorly on multi-class networks. In this study, we demonstrate that the proposed framework outperform SPSA by a significant margin.
- Lu et al. (2013) builds on top of Yang (1995) to solve for the equilibrium and GLS formulation iteratively, while it estimates the gradient more accurately than Yang (1995) on congested networks by using cumulative curves of link flows. In this study, we propose a new concept of tree-based cumulative curves for link/path flows, which allows us to record more detailed information and approximate the gradient more accurately.

The multi-class traffic assignment models and OD estimation models can be built by extending the single-class models. However, the main challenges are how to estimate MCDOD in such a way to match multi-source spatio-temporal data in a large-scale transportation network. A number of studies (Gundaliya et al., 2008; Venkatesan et al., 2008; Qian et al., 2017) investigate the multi-class traffic flows in general networks. The multi-class DTA models are studied by Dafermos (1972), Yang and Huang (2004), Huang and Li (2007) without the real-world data validation. There are studies estimating the static OD demand for multiple vehicle classes (Wong et al., 2005; Raothanachonkun et al., 2006; Noriega and Florian, 2007; Zhao et al., 2018). In particular, Shao et al. (2015) estimated the probabilistic distribution of the multi-class OD using traffic counts. The research gap lies in estimating MCDOD in large-scale networks that are consistent with real-world multi-source traffic data.

To summarize, there are two major challenges in solving the MCDODE problem: how to formulate the MCDODE problem with real-world multi-source data and how to solve the MCDODE problem in large-scale networks. In this paper, we develop a data-driven framework that estimates the multi-class dynamic OD demand using traffic counts and travel time data that are partially observed in general networks. The proposed framework formulates the MCDODE problem and represents it with a computational graph. The MCDODE is solved with a novel forward-backward algorithm on the computational graph. The MCDODE framework can be solved using multi-core CPUs or Graphics Processing Units (GPUs), and hence the proposed method can be computational efficient and applied to the large-scale networks and multi-day data. The closest work to this paper is that of Wu et al. (2018), which constructs a layered computational graph for the single-class static OD estimation problem. This paper extends the computational graph approach to the case of multi-class dynamic OD demand by tackling additional challenges on the incorporating flow dynamics and characteristics across both classes and time of day. We also build a novel framework to evaluate the exact multi-class Dynamic Assignment Ratio (DAR) matrix for simulation-based traffic assignment models. The main contributions of this paper are summarized as follows:

1) We propose a theoretical formulation for estimating multi-class dynamic OD demand. The formulation is represented on a computational graph such that the MCDODE can be solved for large-scale networks with large-scale traffic data. The proposed MCDODE formulation can handle any form of traffic data, such as flow, speed or trip cost.
2) We propose a novel forward-backward algorithm to solve for the MCDODE formulation on the constructed computational graph with simulation-based traffic assignment models. The proposed algorithm is a reformulation of the standard iterative heuristic methods for solving bi-level optimizations, while it provides a novel perspective to view the (MC) DODE problem as a machine learning task.
3) We use a tree-based cumulative curves to evaluate the exact multi-class Dynamic Assignment Ratio (DAR) matrix, and a Growing Tree algorithm is proposed to construct the tree-based cumulative curves during the traffic simulation.
4) We examine the proposed MCDODE framework on a large-scale network to demonstrate its effectiveness and computational efficiency of the solution algorithm.

The remainder of this paper is organized as follows. Section 2 presents the formulation details of MCDODE. Section 3 describes the solution pipeline and discusses practical issues for MCDODE framework. In Section 4, a small network is used to demonstrate the accuracy, efficiency and robustness of the MCDODE framework. Experiments are also conducted on a mid-size network to verify the effectiveness of the proposed framework. In addition, we demonstrate the scalability and computational efficiency of the proposed framework using real-world data in a large-scale network. At last, conclusions are drawn in Section 5.

2. Formulation

In this section, we discuss the multi-class dynamic origin-destination estimation (MCDODE) framework. The framework consists two primary parts: 1) a bi-level formulation of the MCDODE problem (Sections 2.1, 2.2, 2.3, 2.4 and 2.5); 2) a computational graph approach to solve the formulation (Section 2.6). Specifically, we first present the notations used in this paper, and then the multi-class dynamic network flow is modeled. Secondly, the MCDODE problem is formulated and represented on a computational graph. We then solve the MCDODE through a novel forward-backward algorithm.

2.1. Notations

Notations are summarized in Table 1.

2.2. Modeling multi-class dynamic network flow

In this section, we first formulate a general network model for multi-class traffic flow in discrete time. We denote the path flow $f_{rsi}^{kh_1}$ as the number of class-i vehicles departing from kth path for OD pair rs in time interval h_1 and link flow $x_{ai}^{h_2}$ as the number of class-i vehicles arriving at the tail of link a in time interval h_2. The relationship between path flow and link flow is presented in Eq. (1).

$$x_{ai}^{h_2} = \sum_{rs \in K_q} \sum_{k \in K_{rs}} \sum_{h_1 \in H} \rho_{rsi}^{ka}\left(h_1, h_2\right) f_{rsi}^{kh_1} \tag{1}$$

where K_q is the set of all OD pairs, and K_{rs} is the path set for OD pair rs. H is the set of all possible time intervals during study period. The indices h_1 and h_2 represent one interval in H. To avoid confusion, we use h_1 to denote the departure time interval of path flow or OD

Table 1
List of notations.

A	The set of all links
K_q	The set of all OD pairs
K_{rs}	The set of all paths between OD pair rs
B	The set of indices of the observed flow
E	The set of indices of the observed travel time
H	The set of all time intervals
D	The set of vehicle classes
	Variables as scalars
h_1	The index of departure time interval of path flow or OD flow
h_2	The index of arrival time interval at the tail of link
i	The index of vehicle class
$x_{ai}^{h_2}$	The flow arriving at the tail of link a in time interval h_2 for vehicle class i
$t_{ai}^{h_2}$	The link travel time of link a in time interval h_2 for vehicle class i
$q_{rsi}^{h_1}$	The flow of OD pair rs in time interval h_1 for vehicle class i
$f_{rsi}^{kh_1}$	The kth path flow for OD pair rs in time interval h_1 for vehicle class i
$p_{rsi}^{kh_1}$	The route choice portion of choosing path k in all paths between OD pair rs in time interval h_1 for vehicle class i
$c_{rsi}^{kh_1}$	The path travel time of kth path for OD pair rs in time interval h_1 for vehicle class i
$L_{ai}^{bh_2}$	The observation/link incidence for link a in time interval h_2 and observed flow b for vehicle class i
$M_{ai}^{eh_2}$	The weight of travel time for $t_{ai}^{h_2}$ in the observed travel time e
$\rho_{rsi}^{ka}(h_1, h_2)$	The portion of the kth path flow departing within time interval h_1 between OD pair rs which arrives at link a within time interval h_2 for vehicle class i (namely, an entry of the dynamic assignment ratio (DAR) matrix)
y_b	The bth observed flow, which might be the linear combination of link flow across vehicle classes, road segments and over time intervals
z_e	The eth observed travel time, which might be the linear combination of link travel time across vehicle classes, road segments and over time intervals
	Variables as tensors
$\mathbf{y}$	The vector of the observed flow
$\mathbf{z}$	The vector of the observed travel time
$\mathbf{x}_i$	The vector of link flow for vehicle class i
$\mathbf{t}_i$	The vector of link travel time for vehicle class i
$\mathbf{f}_i$	The vector of path flow for vehicle class i
$\mathbf{c}_i$	The vector of path travel time for vehicle class i
$\mathbf{L}_i$	The observation/link incidences matrix for vehicle class i
$\mathbf{M}_i$	The travel time weight matrix for vehicle class i
$\mathbf{p}_i$	The matrix of route choice portions for vehicle class i
$\boldsymbol{\rho}_i$	The dynamic assignment ratio (DAR) matrix for vehicle class i

flow, and h_2 to denote the arrival time interval at the tail of link, respectively. The departure time interval of path flow or OD flow represents the time interval in which the certain flow or OD flow departs, and the arrival time interval at the tail of link represents the time interval in which a certain flow arrives at the tail of the link.

The dynamic assignment ratio (DAR) $\rho_{rsi}^{ka}(h_1,h_2)$ denotes the portion of the kth path flow departing within time interval h_1 for OD pair rs which arrives at link a within time interval h_2 for vehicle class i (Ma and Qian, 2018b). For each OD pair rs, there are K_{rs} paths for travelers to choose from, and the portion of choosing path k in time interval h_1 for vehicle class i is denoted as $p_{rsi}^{kh_1}$. The OD flow and path flow for vehicle class i can be represented by Eq. (2).

$$f_{rsi}^{kh_1} = p_{rsi}^{kh_1} q_{rsi}^{h_1} \tag{2}$$

where OD demand $q_{rsi}^{h_1}$ represents the number of class-i vehicles for OD pair rs in time interval h_1. The link travel time, path travel time and DAR can be obtained from the dynamic network loading (DNL) models, as presented in Eq. (3).

$$\left\{t_{ai}^{h_2}, c_{rsi}^{kh_1}, \boldsymbol{\rho}_{rsi}^{ka}(h_1,h_2)\right\}_{r,s,i,k,a,h_1,h_2} = \Lambda\left(\left\{f_{rsi}^{kh_1}\right\}_{i,r,s,k,h_1}\right) \tag{3}$$

where $\Lambda(\cdot)$ represents the multi-class dynamic network loading models. The travel time can be generalized to any form of the disutility as long as it can be simulated by Λ. The generalized travel time can include roads tolls, left turn penalty, travelers' preferences and so on. The DNL function Λ takes the multi-class path flow as input and outputs the spatio-temporal network conditions $\{t_{ai}^{h_2}, c_{rsi}^{kh_1}\}$ and the DAR matrix $\boldsymbol{\rho}_{rsi}^{ka}(h_1,h_2)$. Though there exists analytical solutions in small networks, Λ is usually represented by simulation-based models in large-scale networks. Many existing models including, but are not limited to, DynaMIT (Ben-Akiva et al., 1998), DYNASMART (Mahmassani et al., 1992), DTALite (Zhou and Taylor, 2014) and MAC-POSTS (Ma et al., 2016; Pi et al., 2018; Pi et al., 2019), can be potentially used as function Λ as long as it can simulate the trajectory of every single vehicle. Within the DNL models, some of the link models include cell transmission model (CTM), Link Transmission Model (LTM), and Link Queue (LQ).

The route choice portion $p_{rsi}^{kh_1}$ is obtained from a generalized route choice function, as presented in Eq. (4).

$$p_{rsi}^{kh_1} = \Psi_{rsi}^{kh_1}(\mathbf{c},\mathbf{t}) \tag{4}$$

$$\mathbf{c} = \left\{c_{rsi}^{kh'} \middle| h' \in H, rs \in K_q, k \in K_{rs}, i \in D\right\} \tag{5}$$

$$\mathbf{t} = \left\{t_{ai}^{h'} \middle| h' \in H, a \in A, i \in D\right\} \tag{6}$$

where $\Psi_{rsi}^{kh_1}(\cdots)$ is a generalized route choice model that takes in the path travel time and link travel time before time interval h_1 and computes the route choice portion for travelers in kth path of OD rs for vehicle class i. We further denote $t_{ai}^{h_2}$ as the travel time of link a in time interval h_2 for vehicle class i, and $c_{rsi}^{kh_1}$ as the travel time of path flow k in OD pair rs departing in time interval h_1. Most of the state-of-art route choice models (Prashker and Bekhor, 2004; Zhou et al., 2012), including Logit and Probit models, satisfy Eq. (4). The Logit and Probit models have been adopted in many studies and achieved great success in real-world applications (Maher et al., 2001). Combining Eqs. (1) and (2), we present the relation of link flow and OD flow in Eq. (7).

$$x_{ai}^{h_2} = \sum_{rs\in K_q}\sum_{k\in K_{rs}}\sum_{h_1\in H}\rho_{rsi}^{ka}\left(h_1,h_2\right)p_{rsi}^{kh_1}q_{rsi}^{h_1} \tag{7}$$

2.3. Modeling the observed flow and observed travel time

In this section, we describe the concept of the observed flow and observed travel time. The motivation for the definition of the observed flow and travel time lies in the indirect and aggregated observations of traffic networks. For example, loop detectors measure the traffic counts in each time interval, but the vehicle classes cannot be differentiated. The Department of Transportation regularly hire individuals to count the vehicles on road segments for both directions and different vehicle classes. In this case, the observation of traffic flow does not differentiate road directions.

To accommodate different kinds of flow observations, we propose the concept of observed flow, denoted by y_b, as a linear combination of link flow across vehicle classes, road segments and time intervals. In the DODE setting, we usually do not need the definition of observed flow because the traffic count data directly indicates the traffic flow on the link. While in MCDODE, this is not the case. We might observe 100 vehicles without knowing the vehicle classes on some links, while we might be able to count the vehicles by their classes on some other links. The observed flow is hereby introduced to uniformly represent different type of data we observed. The formulation of the observed flow is presented in Eq. (8).

$$y_b = \sum_{i\in D}\sum_{a\in A}\sum_{h_2\in H}L_{ai}^{bh_2}x_{ai}^{h_2} \tag{8}$$

where $b \in B$ is the index of observed flow and B is the set of indices of observed flow. The unit of y_b is the same as $x_{ai}^{h_2}$. The weights $L_{ai}^{bh_2}$ are fixed when the observed data is known. Usually, the number of observed flow equals the number of data we have, which is the same as the number of sensors multiplied by the number of time intervals. For example, if there are two loop detectors recording the traffic counts in each interval, then we have $|B| = 2|H|$. By formulation (8), the observed flow can be the traffic count observations in various forms, including traffic counts of a single-class vehicles, aggregated traffic counts of all vehicles, aggregated traffic counts of a road for both directions, and aggregated traffic counts of multiple links. The link/observation incidence $L_{ai}^{bh_2}$ represents how the observed flow is aggregated. $L_{ai}^{bh_2}$ is 1 if link flow $x_{ai}^{h_2}$ is observed in the observed flow y_b and 0 otherwise, and a similar definition can be found in Yang et al. (2018).

Similarly, we may also observe the travel time across multiple links, vehicle classes and time intervals. For example, we may observe the total travel time of a highway which consists of multiple consecutive links, or we may observe the average travel time of car and trucks in a single link. To accommodate different kinds of travel time observations, we assume the observed travel time can be represented by a linear combination of all link travel time, as presented in Eq. (9).

$$z_e = \sum_{i\in D}\sum_{a\in A}\sum_{h_2\in H} M_{ai}^{eh_2} t_{ai}^{h_2} \tag{9}$$

where $M_{ai}^{eh_2}$ represents the weight of travel time for $t_{ai}^{h_2}$ in the observed travel time e. The unit of z_e is the same as $t_{ai}^{h_2}$. $e \in E$ is the index of observed travel time and E is the set of indices of observed travel time. The formulation (9) is also general enough to accommodate various types of travel time observations. For example, the observed travel time includes link travel time of a single-class vehicles, path travel time of single-class vehicles [1], average travel time of all vehicle classes. We note that $M_{ai}^{eh_2} \in [0,1]$ since we may observe the average travel time of multiple links, while $L_{ai}^{bh_2} \in \{0,1\}$ because traffic flow is usually observed in an aggregated manner. We note that both $L_{ai}^{bh_2}$ and $M_{ai}^{eh_2}$ are sparse in the sense that each observed data point is usually associated with a small portion of link flow/travel time. Hence only a subset of link flow and link travel time data need to be record when computing the observed flow and travel time, which makes the proposed framework more efficient. Details will be discussed in Section 3.2.

Example 1. To illustrate the formulation of the observed flow and observed travel time, we consider a two-link network presented in Fig. 1. We only consider one time interval, hence $|H| = 1$. We assume vehicle class 1 represents cars, and vehicle class 2 represents trucks.

Suppose we install a camera on link 1, and a loop detector on link 2. As the camera can detect vehicles by vehicle classes, we observe there are 50 cars passing link 1. The loop detector can only count the vehicle regardless of vehicle classes, hence suppose we observed there are 150 vehicles (trucks and cars) passing link 2. Then we have

$$\begin{aligned}
y_{b=1} &= 50\\
y_{b=2} &= 150\\
L_{a=1,i=1}^{b=1,h_2=1} &= 1\\
L_{a=1,i=2}^{b=1,h_2=1} &= 0\\
L_{a=2,i=1}^{b=2,h_2=1} &= 1\\
L_{a=2,i=2}^{b=2,h_2=1} &= 1
\end{aligned}$$

Therefore

$$\begin{aligned}
y_{b=1} &= L_{a=1,i=1}^{b=1,h_2=1} x_{a=1,i=1}^{h_2=1} + L_{a=1,i=2}^{b=1,h_2=1} x_{a=1,i=2}^{h_2=1}\\
&= x_{a=1,i=1}^{h_2=1}\\
y_{b=2} &= L_{a=2,i=1}^{b=2,h_2=1} x_{a=2,i=1}^{h_2=1} + L_{a=2,i=2}^{b=2,h_2=1} x_{a=2,i=2}^{h_2=1}\\
&= x_{a=2,i=1}^{h_2=1} + x_{a=2,i=2}^{h_2=1}
\end{aligned}$$

Similarly, the camera can compute the travel time for different car class separately, and suppose we observe the travel time for traversing link 1 is 100 for cars. Again, loop detected does not differentiate the vehicle class, and we only observe the average travel time of link 2 is 70 for cars and trucks. Then we have:

[1] The path travel time is the summation of a list of link travel times, and the matrix M needs to be updated for each simulation.

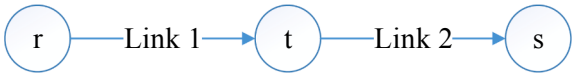

Fig. 1. A two-link network.

$$
\begin{aligned}
z_{e=1} &= 100 \\
z_{e=2} &= 70 \\
M_{a=1,i=1}^{e=1,h_2=1} &= 1 \\
M_{a=2,i=1}^{e=1,h_2=1} &= 1 \\
M_{a=2,i=1}^{e=2,h_2=1} &= 0.5 \\
M_{a=2,i=2}^{e=2,h_2=1} &= 0.5
\end{aligned}
$$

Then we have

$$
\begin{aligned}
z_{e=1} &= M_{a=1,i=1}^{e=1,h_2=1} t_{a=1,i=1}^{h_2=1} + M_{a=2,i=1}^{e=1,h_2=1} t_{a=2,i=1}^{h_2=1} \\
&= t_{a=1,i=1}^{h_2=1} + t_{a=2,i=1}^{h_2=1} \\
z_{e=2} &= M_{a=2,i=1}^{e=2,h_2=1} t_{a=2,i=1}^{h_2=1} + M_{a=2,i=2}^{e=2,h_2=1} t_{a=2,i=2}^{h_2=1} \\
&= \frac{t_{a=2,i=1}^{h_2=1} + t_{a=2,i=2}^{h_2=1}}{2}
\end{aligned}
$$

We note the "average travel time for cars and trucks" means the "average" of car travel time and truck travel time, respectively. It is also possible to consider the weighted average of travel time by car and truck flow together, which will be discussed in Section 2.6.

2.4. MCDODE formulation

The formulation for multi-class dynamic origin-destination estimation (MCDODE) is presented in Eq. (10).

$$
\begin{aligned}
\min_{\left\{q_{rsi}^{h_1}\right\}_{i,r,s,h_1}} \quad & w_1 \sum_{b\in B}\left(y_b^{'} - \sum_{i\in D}\sum_{a\in A}\sum_{h_2\in H} L_{ai}^{bh_2}\left(\sum_{rs\in K_q}\sum_{k\in K_{rs}}\sum_{h_1\in H} \rho_{rsi}^{ka}\left(h_1,h_2\right) p_{rsi}^{kh_1} q_{rsi}^{h_1}\right)\right)^2 + w_2 \sum_{e\in E}\left(z_e^{'} - \sum_{i\in D}\sum_{a\in A}\sum_{h_2\in H} M_{ai}^{eh_2} t_{ai}^{h_2}\right)^2 \\
\text{s.t.} \quad & \left(\left\{t_{ai}^{h_2}, c_{rsi}^{kh_1}, \boldsymbol{\rho}_{rsi}^{ka}\left(h_1,h_2\right)\right\}\right)_{r,s,i,k,a,h_1,h_2} = \Lambda\left(\left\{f_{rsi}^{kh_1}\right\}_{i,r,s,k,h_1}\right) \\
& f_{rsi}^{kh_1} = p_{rsi}^{kh_1} q_{rsi}^{h_1}, \forall rs \in K_q, k \in K_{rs}, i \in D, h_1 \in H \\
& p_{rsi}^{kh_1} = \Psi_{rsi}^{k}\left(\tilde{\mathbf{c}}_{rsi}^{kh_1}, \tilde{\mathbf{t}}_{ai}^{h_1}\right), \forall rs \in K_q, k \in K_{rs}, i \in D, h_1 \in H \\
& q_{rsi}^{h_1} \geqslant 0, \forall rs \in K_q, h_1 \in H, i \in D
\end{aligned}
\tag{10}
$$

where $y_b^{'}$ and $z_e^{'}$ are the observed flow and travel time data collected from real-world. The link/path travel time can be computed through tracking cumulative curves in the DNL model (Lu et al., 2013), and the computation for DAR will be discussed in Section 3.2.

Table 2
MCDODE framework variable vectorization.

Variable	Scalar	Vector	Dimension	Type	Description
OD flow	q_{rsi}^{h}	$\mathbf{q}_i$	$\mathbb{R}^{N\|K\|}$	Dense	q_{rsi}^{h} is place at entry $(h-1)\|K\|+k$
Path flow	f_{rsi}^{kh}	$\mathbf{f}_i$	$\mathbb{R}^{N\Pi}$	Dense	f_{rsi}^{kh} is placed at entry $(h-1)\Pi+k$
Link flow	x_{ai}^{h}	$\mathbf{x}_i$	$\mathbb{R}^{N\|A\|}$	Dense	x_{ai}^{h} is placed at entry $(N-1)\|A\|+k$
Link travel time	t_{ai}^{h}	$\mathbf{t}_i$	$\mathbb{R}^{N\|A\|}$	Dense	t_{ai}^{h} is placed at entry $(N-1)\|A\|+k$
Path travel time	c_{rsi}^{kh}	$\mathbf{c}_i$	$\mathbb{R}^{N\Pi}$	Dense	c_{rsi}^{kh} is placed at entry $(N-1)\Pi+k$
Observed flow	y_b	$\mathbf{y}$	$\mathbb{R}^{\|B\|}$	Dense	y_b is placed at entry b
Observed travel time	z_e	$\mathbf{z}$	$\mathbb{R}^{\|E\|}$	Dense	z_e is placed at entry e
DAR matrix	$\rho_{rsi}^{ka}(h_1,h_2)$	$\boldsymbol{\rho}_i$	$\mathbb{R}^{N\|A\|\times N\Pi}$	Sparse	$\rho_{rsi}^{ka}(h_1,h_2)$ is placed at entry $[(h_2-1)\|A\|+a,(h_1-1)\Pi+k]$
Route choice matrix	p_{rsi}^{kh}	$\mathbf{p}_i$	$\mathbb{R}^{N\Pi\times N\|K\|}$	Sparse	p_{rsi}^{kh} is placed at entry $[(h-1)\|\Pi\|+k,(h-1)\|K\|+rs]$
Observation/link incidence matrix	L_{ai}^{bh}	$\mathbf{L}_i$	$\mathbb{R}^{\|B\|\times N\|A\|}$	Sparse	L_{ai}^{bh} is placed at entry $[b,(h-1)\|A\|+a]$
Link travel time portion matrix	M_{ai}^{eh}	$\mathbf{M}_i$	$\mathbb{R}^{\|E\|\times N\|A\|}$	Sparse	M_{ai}^{eh} is placed at entry $[e,(h-1)\|A\|+a]$

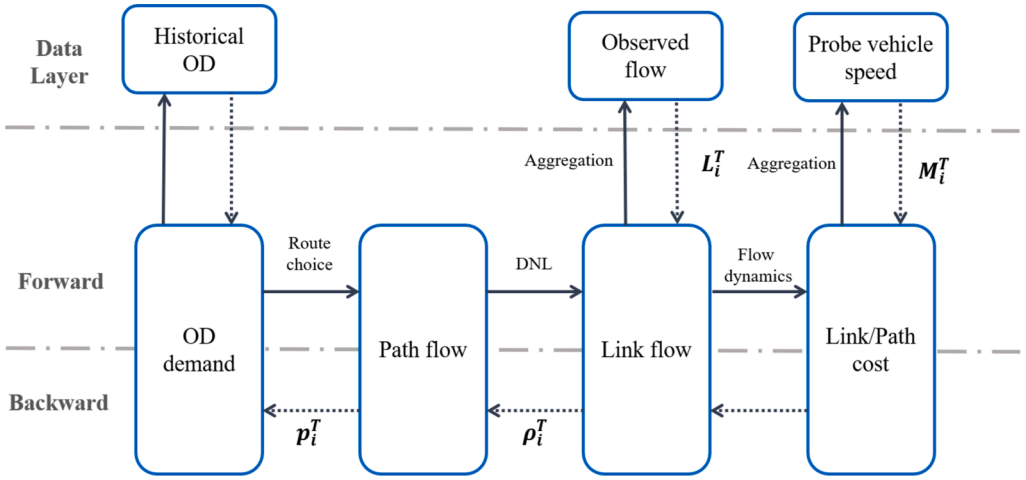

Fig. 2. An illustration of the forward-backward algorithm.

The parameters w_1 and w_2 are the weights for each data source, respectively. Higher weights represent more importance on a specific data source, and the weights can be tuned using cross-validation. The DNL model usually encapsulates a multi-class traffic simulation model, and we will discuss the simulation model in Section 3.1.

Eq. (10) is a bi-level optimization problem with upper level minimizing the ℓ^2 norm between observed and estimated flow and travel time, and the lower level solves the traffic assignment problem denoted by function $\Psi_{rsi}^k(\cdot)$ and $\Lambda(\cdot)$. When $\Psi_{rsi}^k(\cdot,\cdot)$ represents the Dynamic User Equilibrium (DUE) conditions, Eq. (10) is a Mathematical Program with Equilibrium Constraints (MPEC) problem. In contrast, when $\Psi_{rsi}^k(\cdot,\cdot)$ represents the Logit-model, Eq. (10) can be formulated as either a bi-level optimization problem or a single-level non-linear optimization problem (Davis, 1994; Ma and Qian, 2018a).

Additional data sources, such as the historical OD demand and survey data, can also be used in the demand estimation. The DODE methods with these data have been extensively studied (Zhang et al., 2008; Wu et al., 2018). This paper focuses on the computational graph approach with observed traffic flow and travel time data, while other data can be potentially incorporated into the proposed framework.

2.5. *Vectorizing the MCDODE formulation*

First of all, the variables involved in the MCDODE formulation are vectorized, and the vectorization is performed for each vehicle class separately. The vectorized variables will further be used in the computational graphs. We set $N = |H|$ and denote the total number path as $\Pi = \sum_{rs}|K_{rs}|, K = |K_q|$. The vectorized variables are presented in Table 2.

With the vectorized variables, Eqs. (7)–(9) can be rewritten in Eq. (11).

$$\begin{aligned} \mathbf{x}_i &= \boldsymbol{\rho}_i \mathbf{p}_i \mathbf{q}_i, \forall i \in D \\ \mathbf{y} &= \sum_{i\in D} \mathbf{L}_i \mathbf{x}_i \\ \mathbf{z} &= \sum_{i\in D} M_i \mathbf{t}_i \end{aligned} \tag{11}$$

Multiplications between sparse matrix and sparse matrix, sparse matrix and dense vector are very efficient, especially on multi-core CPUs and Graphics Processing Units (GPUs). Therefore, Eq. (11) can be evaluated efficiently. The MCDODE formulation in Eq. (10) can be cast into the vectorized form presented in Eq. (12).

$$\begin{aligned} \min_{\{\mathbf{q}_i\}_i} \quad & w_1\left(\left\|\mathbf{y}' - \sum_{i\in D}\mathbf{L}_i\mathbf{x}_i\right\|_2^2\right) + w_2\left(\left\|\mathbf{z}' - \sum_{i\in D}\mathbf{M}_i\mathbf{t}_i\right\|_2^2\right) \\ \text{s.t.} \quad & \{\mathbf{t}_i, \mathbf{c}_i, \boldsymbol{\rho}_i\}_i = \Lambda\left(\{\mathbf{f}_i\}_i\right) \\ & \mathbf{f}_i = \mathbf{p}_i\mathbf{q}_i \quad \forall i \in D \\ & \mathbf{x}_i = \boldsymbol{\rho}_i\mathbf{f}_i \quad \forall i \in D \\ & \mathbf{p}_i = \Psi_i(\{\mathbf{c}_i\}_i, \{\mathbf{t}_i\}_i) \quad \forall i \in D \\ & \mathbf{q}_i \geqslant 0 \quad \forall i \in D \end{aligned} \tag{12}$$

where $\Psi_i(\{\mathbf{c}_i\}_i, \{\mathbf{t}_i\}_i)$ is the vectorized route choice function for vehicle class *i*. We further substitute the path flow and link flow to the objective function, as presented in Eq. (13). To summarize, $\mathbf{y}'$ and $\mathbf{z}'$ are known as observed data, and we estimate $\{\mathbf{q}_i\}_i$ such that all

variables including $\mathbf{f}_i, \mathbf{x}_i, \mathbf{t}_i, \mathbf{c}_i, \boldsymbol{\rho}_i, \mathbf{p}_i$ can be computed from Λ and Ψ_i. In addition, $\mathbf{L}_i$ and $\mathbf{M}_i$ are determined by the network topology and data availability.

$$\begin{aligned} \min_{\{\mathbf{q}_i\}_i} \quad & w_1\left(\left\|\mathbf{y}' - \sum_{i\in D}\mathbf{L}_i\boldsymbol{\rho}_i\mathbf{p}_i\mathbf{q}_i\right\|_2^2\right) + w_2\left(\left\|\mathbf{z}' - \sum_{i\in D}\mathbf{M}_i\mathbf{t}_i\right\|_2^2\right) \\ \text{s.t.} \quad & \{\mathbf{t}_i, \mathbf{c}_i, \boldsymbol{\rho}_i\}_i = \Lambda\left(\{\mathbf{f}_i\}_i\right) \\ & \mathbf{p}_i = \Psi_i(\{\mathbf{c}_i\}_i, \{\mathbf{t}_i\}_i) \quad \forall i \in D \\ & \mathbf{q}_i \geqslant 0 \quad \forall i \in D \end{aligned} \tag{13}$$

2.6. A computational graph for MCDODE

In order to solve the MCDODE problem, our goal is to obtain the gradient of the objective function with respect to the dynamic OD demand $\frac{\partial \mathscr{L}}{\partial \mathbf{q}_i}$ for formulation (13). Throughout this paper, we use GOF to represent $\frac{\partial \mathscr{L}}{\partial \mathbf{q}_i}$. The GOF can be estimated by running DNL models through finite differences. For example, heuristic iterative algorithms between lower and upper problem are widely used in traffic applications (Yang, 1995). Sensitivity analysis of the traffic assignment problem has been adopted to evaluate the gradients of bi-level problem. The SPSA algorithm approximates the gradient by finite differences (Lu et al., 2015; Balakrishna et al., 2008). Frederix et al. (2011), Frederix et al. (2013) estimated the gradients by the Marginal Computation (MaC) approach without running the whole DNL model (Corthout et al., 2011). The finite difference-based methods can hardly be applied to large-scale networks due to the computational intensity of the DNL model. The second way to compute the GOF is to approximate the objective functions using a simplified and analytical form, and this paper adopts this method. To the best of our knowledge, Lu et al. (2013) and Osorio (2019) are closest to our study. Lu et al. (2013) estimated the GOF using the cumulative curves, and Osorio (2019) approximated the gradient by linearizing the dynamic traffic assignment models. In this paper, we propose an alternative approach to compute the instantaneous GOF within each simulation efficiently using a novel tree-based cumulative curve which can be constructed and evaluated efficiently on large-scale networks.

To this end, we propose a novel approach to obtain the GOF through the forward-backward algorithm on a computational graph. First we cast Eq. (13) into a computational graph representation, and Fig. 2 describes the structure of the computational graph for MCDODE. The forward-backward algorithm runs on the computational graph, and algorithm consists of two processes: the forward iteration and the backward iteration.

In general, the forward iteration assumes that OD demand is fixed and it solves for the network conditions, while the backward iteration assumes the network conditions are fixed and it updates the OD demand. The whole process of forward-backward algorithm resembles some heuristic methods that solve the upper level and lower level problem iteratively (Yang, 1995), while the forward-backward algorithm explores the analogy of a MCDODE problem and a machine learning task (*i.e.*, neural networks).

Forward iteration: Forward iteration takes multi-class OD demand as the input and solves the traffic assignment problem presented in Eq. (14).

$$\begin{aligned} \mathbf{f}_i &= \mathbf{p}_i\mathbf{q}_i \quad \forall i \in D \\ \{\mathbf{t}_i, \mathbf{c}_i, \boldsymbol{\rho}_i\}_i &= \Lambda\left(\{\mathbf{f}_i\}_i\right) \\ \mathbf{p}_i &= \Psi_i(\{\mathbf{c}_i\}_i, \{\mathbf{t}_i\}_i) \quad \forall i \in D \end{aligned} \tag{14}$$

The forward iteration includes the multi-class dynamic network loading models Λ and route choice models $\{\Psi_i\}_i$. The output of the forward iteration is route choice portion $\mathbf{p}_i$, DAR matrix $\boldsymbol{\rho}_i$ and link/path travel time $(\mathbf{t}_i, \mathbf{c}_i)$. We omit the solution method for traffic assignment problem, as it has been extensively studied in many literature (Peeta and Ziliaskopoulos, 2001). After solving the dynamic traffic assignment models, the forward iteration compute the objective function (loss) in formulation (13), which can be represented by a series of equations in Eqs. (15) and (16).

$$\mathscr{L} = w_1\mathscr{L}_1 + w_2\mathscr{L}_2 \tag{15}$$

$$\begin{aligned} \mathscr{L}_1 &= \|\mathbf{y}' - \mathbf{y}\|_2^2 & \mathscr{L}_2 &= \|\mathbf{z}' - \mathbf{z}\|_2^2 \\ \mathbf{y} &= \sum_{i\in D}\mathbf{L}_i\mathbf{x}_i & \mathbf{z} &= \sum_{i\in D}\mathbf{M}_i\mathbf{t}_i \\ \mathbf{x}_i &= \boldsymbol{\rho}_i\mathbf{f}_i & \{\mathbf{t}_i\}_i &= \overline{\Lambda}(\{\mathbf{x}_i\}_i) \\ \mathbf{f}_i &= \mathbf{p}_i\mathbf{q}_i & \mathbf{x}_i &= \boldsymbol{\rho}_i\mathbf{f}_i \\ & & \mathbf{f}_i &= \mathbf{p}_i\mathbf{q}_i \end{aligned} \tag{16}$$

where $\mathbf{y}'$ is the observed flow and $\mathbf{y}$ is the reproduction of the observed flow estimated from the traffic assignment model. Similarly, $\mathbf{z}'$ is the observed travel time and $\mathbf{z}$ is the reproduction of the observed travel time estimated from the traffic assignment model. We use $\overline{\Lambda}$ to represent a part of the function Λ, and $\overline{\Lambda}$ takes dynamic link flow as input and outputs the link travel time $\{\mathbf{t}_i\}_i$. Precisely, $\overline{\Lambda}$ represents the dynamic link models (Zhang et al., 2013; Jin, 2012). We assume $\{\mathbf{t}_i\}_i$ is differentiable with respect to the incoming link flow. Most existing link models such as the Cell Transmission Model (CTM), Link Transmission Model (LTM), and Link Queue (LQ) can be used as $\{\mathbf{t}_i\}_i$.

The forward iteration solves the DTA problem with considering any form of route choice model. It means we might need to run the DNL model for several time solve for some of the route choice models, such as DUE and logit model, while some other route choice models only require one DNL run, such as adaptive routing and hybrid routing (Qian and Zhang, 2013).

Backward iteration: The backward iteration searches for the GOF for formulation (17) with the route choice portion $\mathbf{p}_i$ and DAR matrix $\boldsymbol{\rho}_i$ known and fixed from the forward iteration. Precisely, the route choice portion $\mathbf{p}_i$ and DAR matrix $\boldsymbol{\rho}_i$ are independent of OD demand in the backward iteration.

$$\begin{aligned} \min_{\{\mathbf{q}_i\}_i} \quad & w_1\left(\left\|\mathbf{y}' - \sum_{i\in D}\mathbf{L}_i\boldsymbol{\rho}_i\mathbf{p}_i\mathbf{q}_i\right\|_2^2\right) + w_2\left(\left\|\mathbf{z}' - \sum_{i\in D}\mathbf{M}_i\mathbf{t}_i\right\|_2^2\right) \\ \text{s.t.} \quad & \mathbf{q}_i \geqslant 0 \quad \forall i \in D \end{aligned} \tag{17}$$

When the GOF can be computed or approximated, a projected gradient descent method can be used to solve Eq. (17). The reason we call the solution process for Eq. (17) a "backward iteration" is that, the GOF for Eq. (17) can be evaluated through the backpropagation (BP) method. Taking the derivative of the objective function step by step, we have a series of equations presented in Eq. (17)

$$\begin{aligned} \frac{\partial \mathscr{L}}{\partial \mathscr{L}_1} &= w_1 & \frac{\partial \mathscr{L}}{\partial \mathscr{L}_2} &= w_2 \\ \frac{\partial \mathscr{L}_1}{\partial \mathbf{y}} &= 2\left(\mathbf{y}' - \sum_{i'\in D}\mathbf{L}_{i'}\boldsymbol{\rho}_{i'}\mathbf{p}_{i'}\mathbf{q}_{i'}\right) & \frac{\partial \mathscr{L}_2}{\partial \mathbf{z}} &= 2\left(\mathbf{z}' - \sum_{i'\in D}\mathbf{M}_{i'}\mathbf{t}_{i'}\right) \\ \frac{\partial \mathscr{L}_1}{\partial \mathbf{x}_i} &= -\mathbf{L}_i^T\frac{\partial \mathscr{L}_1}{\partial \mathbf{y}} & \frac{\partial \mathscr{L}_2}{\partial \mathbf{t}_i} &= -\mathbf{M}_i^T\frac{\partial \mathscr{L}_2}{\partial \mathbf{z}} \\ \frac{\partial \mathscr{L}_1}{\partial \mathbf{f}_i} &= \boldsymbol{\rho}_i^T\frac{\partial \mathscr{L}_1}{\partial \mathbf{x}_i} & \frac{\partial \mathscr{L}_2}{\partial \mathbf{x}_i} &= \frac{\partial \overline{\Lambda}(\{\mathbf{x}_i\}_i)}{\partial \mathbf{x}_i}\frac{\partial \mathscr{L}_2}{\partial \mathbf{t}_i} \\ \frac{\partial \mathscr{L}_1}{\partial \mathbf{q}_i} &= \mathbf{p}_i^T\frac{\partial \mathscr{L}_1}{\partial \mathbf{f}_i} & \frac{\partial \mathscr{L}_2}{\partial \mathbf{f}_i} &= \boldsymbol{\rho}_i^T\frac{\partial \mathscr{L}_2}{\partial \mathbf{x}_i} \\ & & \frac{\partial \mathscr{L}_2}{\partial \mathbf{q}_i} &= \mathbf{p}_i^T\frac{\partial \mathscr{L}_2}{\partial \mathbf{f}_i} \end{aligned} \tag{18}$$

Combining the equations in (18), the GOF is presented in Eq. (19).

$$\begin{aligned} \frac{\partial \mathscr{L}}{\partial \mathbf{q}_i} &= \frac{\partial \mathscr{L}}{\partial \mathscr{L}_1}\frac{\partial \mathscr{L}_1}{\partial \mathbf{q}_i} + \frac{\partial \mathscr{L}}{\partial \mathscr{L}_2}\frac{\partial \mathscr{L}_2}{\partial \mathbf{q}_i} \\ &= -2w_1\mathbf{p}_i^T\boldsymbol{\rho}_i^T\mathbf{L}_i^T\left(\mathbf{y}' - \sum_{i'\in D}\mathbf{L}_{i'}\boldsymbol{\rho}_{i'}\mathbf{p}_{i'}\mathbf{q}_{i'}\right) - 2w_2\mathbf{p}_i^T\boldsymbol{\rho}_i^T\frac{\partial \overline{\Lambda}(\{\mathbf{x}_i\}_i)}{\partial \mathbf{x}_i}\mathbf{M}_i^T\left(\mathbf{z}' - \sum_{i'\in D}\mathbf{M}_{i'}\mathbf{t}\right) \end{aligned} \tag{19}$$

Note the backward iteration does not need any information from the route choice model as long as the DAR matrix ρ_i is known from the forward iteration. In addition, the derivative $\frac{\partial \overline{\Lambda}(\{\mathbf{x}_i\}_i)}{\partial \mathbf{x}_i}$ represents the sub-gradient of function $\overline{\Lambda}(\{\mathbf{x}_i\}_i)$, hence it suffices to assume

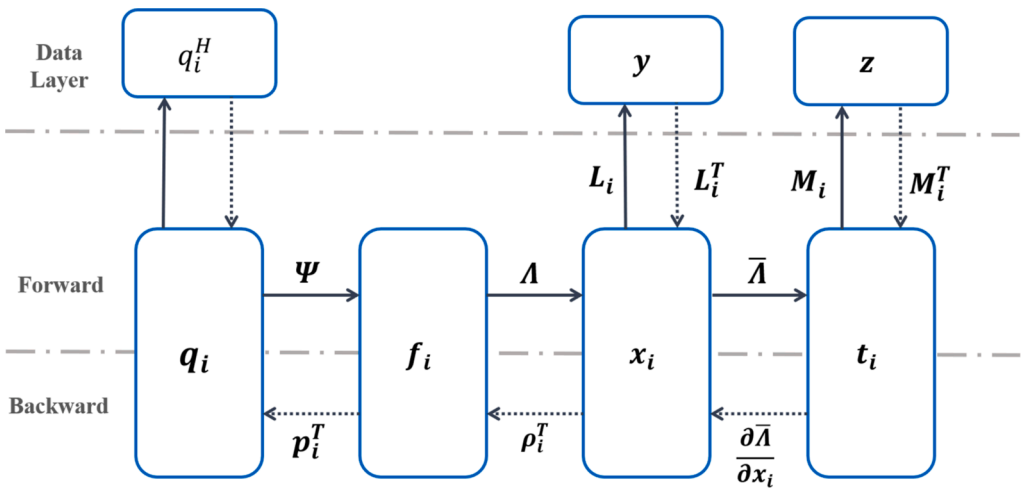

Fig. 3. The forward-backward algorithm among all variables.

$\overline{\Lambda}(\{\mathbf{x}_i\}_i)$ being continuous. Details of the sub-gradient $\frac{\partial\overline{\Lambda}(\{\mathbf{x}_i\}_i)}{\partial\mathbf{x}_i}$ will be discussed in Section 3.3. Theoretically, $\boldsymbol{\rho}_i$ and $\mathbf{p}_i$ are also functions of $\mathbf{q}_i$ (Frederix et al., 2013), while we assume both quantities are fixed in the backward iterations and their values are updated in the forward iterations. Essentially, the forward-backward algorithm is a reformulation of the standard iterative method to solve for the bi-level optimization problems (Yang, 1995), while the new formulation provides a novel perspective to view the (MC) DODE problem as a machine learning problem.

The proposed computational graph and forward-backward algorithm are illustrated in Fig. 2, and the algorithms is also described using the relationship among different variables, as presented in Fig. 3. The solid arrow represents the forward iteration and the dashed arrow represents the backward iteration. We omit the forward and backward iterations for historical OD data, while the other processes are described above. The forward and backward iterations for historical OD data are straightforward by adding $w_3\mathscr{L}_3 = w_3\sum_{i\in D}\left\|\mathbf{q}_i^H - \mathbf{q}_i\right\|_2^2$ to the MCDODE formulation (13) where $\mathbf{q}_i^H$ is the historical OD demand for vehicle class i (Zhang et al., 2008). For readers who are not familiar with computational graphs and neural networks, detailed tutorials are referred to better understand the presented computational graph (Wu, 2018; Wu and Zhou, 2018).

To solve the MCDODE formulation in Eq. (12), we run the forward-backward algorithm to compute the GOF. Then the projected gradient descent method is used to update the OD demand. There is a family of gradient-based methods that can be used, and we will discuss them in Section 3.4. Comparisons among different methods will also be conducted in the numerical experiments.

Our computational graph approach shares many similarities with the deep learning models: 1) both models contain high dimensional parameters; 2) multi-core CPU and GPU can be used to speed up the solution process (details will be discussed in Section 3.4); 3) many advanced variants of gradient-based method can be used to solve the models; 4) Backpropagation method can be used to evaluate the gradient layer by layer (Rumelhart et al., 1985; Wu et al., 2018). Potentially, all the techniques used in the training for deep learning can be used for the proposed computational graph. In this paper, we will test the advanced gradient-based method and multi-processing, while leaving other techniques, such as dropout (Gal and Ghahramani, 2016), transfer learning (Pan and Yang, 2010) and regularization (Neyshabur et al., 2014) for future research.

By reformulating the MCDODE problem to a computational graph, many existing techniques in deep learning can be used to improve and accelerate the solution methods for MCDODE. However, there are still disadvantages. Similar to other OD estimation methods, the underdeterminacy issue still exists in the computational graph. There are studies to address he underdeterminacy issue by assuming the structure of OD demand (Cascetta et al., 2013; Bauer et al., 2017; Krishnakumari et al., 2019) or using historical OD demand (Yang et al., 2018). Due to the limited observed information and complicated behavior models, the solution to the OD demand is not unique. Similar to the deep neural networks, the computation graph approach for MCDODE can also get stuck at local minima, while there is no theoretical guarantee for the convergence. As discussed above, the GOF is approximated by linearizing the DNL process, and hence the changes of $\boldsymbol{\rho}_i$ and $\mathbf{p}_i$ with respect to $\mathbf{q}_i$ are not considered when evaluating the GOF.

There are many flexible ways to incorporate the observed data in the backward-forward algorithm. For example, it is possible to view the vehicle trajectories as samples from path flow and incorporate the trajectory data into the computational graph. We can also compute the weighted average speed for cars and trucks to reproduce the "average link travel time" since the network conditions (flow, travel time) are fixed in the backward iterations. Hence the question left in Example 1 is answered.

As for the stopping criteria, we first claim Proposition 1 holds by the definition of the forward-backward algorithm.

Proposition 1. *In the proposed forward-backward algorithm, the DAR matrix, route choice portions and link/path travel time do not change if and only if* $\frac{\partial\mathscr{L}}{\partial\mathbf{q}_i} = 0, \forall i$ *during the forward and backward iterations.*

Proposition 1 indicates that the forward-backward algorithm converges when either of the following two conditions hold: 1) in the forward iteration, the DAR matrix, route choice portions and link/path travel time do not change; 2) in the backward iteration, $\frac{\partial\mathscr{L}}{\partial\mathbf{q}_i} = 0$. During the forward-backward algorithm, we can either monitor the change of DAR matrix, route choice portions and link/path or the change of the GOF. We can also see that the forward-backward algorithm converges to the local minimum for formulation (13) in Proposition 2.

Proposition 1 also depicts the interdependency among all variables in the traffic networks. If any variable changes in the computational graph, the rest of variables will change accordingly. The network conditions are stabilized only when the equilibrium (*e.g. DUE, SUE*) conditions are satisfied. This suggests that the computational graph has great potentials in modeling, managing and operating the whole transportation system in a comprehensive and coherent manner (Sun et al., 2019).

Proposition 2. *When the forward-backward algorithm converges, the estimated OD demand* $\{\mathbf{q}_i\}_i$ *is a local minimum for formulation (13).*

Proof. When the forward-backward algorithm converges, we know $\frac{\partial\mathscr{L}}{\partial\mathbf{q}_i} = 0$ by Proposition 1. Since formulation (17) is convex, $\frac{\partial^2\mathscr{L}}{\partial\mathbf{q}_i^2} \geq 0$. Therefore, $\{\mathbf{q}_i\}_i$ is the local minimum solution. □

3. Solution algorithms

In this section, we first discuss several practical issues to complete the MCDODE framework with the forward-backward algorithm.

We develop a multi-class traffic simulation package called MAC-POSTS for the network loading function $\Lambda(\cdot)$, and a Growing Tree algorithm is proposed to obtain the DAR matrix. Secondly, we discuss how to evaluate the derivative of link travel time in dynamic networks. Thirdly, we present how to incorporate multi-day observation data in the proposed MCDODE framework with multiprocessing. Lastly, the whole framework for MCDODE is presented.

3.1. Multi-class traffic simulation

The dynamic network loading function $\Lambda(\cdot)$ is fulfilled with the mesoscopic multi-class traffic simulation package Mobility Data Analytics Center - Prediction, Optimization, and Simulation toolkit for Transportation Systems (MAC-POSTS) developed by the Mobility Data Analytics Center at Carnegie Mellon University. To simulate heterogeneous traffic with multi-class vehicles like cars and trucks, the simulation package captures flow dynamics and outputs the traffic metrics for multi-class vehicles. The flow metrics include the traffic volumes, traffic speed and travel time. Due to the page limitation, more details about the mesoscopic multi-class traffic simulation model in MAC-POSTS is presented in Qian et al. (2017), Pi et al. (2018).

With the multi-class dynamic OD demand known, the mesoscopic multi-class traffic simulation model performs the following steps orderly in every loading interval (*e.g.* five seconds):

1. Vehicle generation: multi-class vehicles are generated at origins according to the traffic demand.
2. Routing: the route choice behaviors of all vehicles are updated, according to the route choice models.
3. Node evolution: cars and trucks are moved through intersections following the intersection flow model.
4. Link evolution: cars and trucks are moved on links following the link flow model.
5. Network flow statistics: the model records link flow counts, link speeds, travel time and other network performance statistics.

We note the loading interval is different from the interval defined in this paper, since loading interval is usually much shorter. After running the simulation, the route choice portion $\mathbf{p}_i$ and link/route travel time $(\mathbf{t}_i, \mathbf{c}_i)$ for each vehicle class can be obtained based on the simulation results. The DAR matrix $\boldsymbol{\rho}_i$ for each vehicle class can also be obtained by constructing the tree-based cumulative curves during the simulation process, and details are presented in Section 3.2.

3.2. Tree-based cumulative curve

In this section, we develop a novel method to compute the DAR matrix during the traffic simulation. Computing the DAR matrix during the simulation is more efficient than obtaining the DAR matrix after the simulation. However, computing the dynamic assignment ratio (DAR) during the traffic simulation is challenging. A naive method to obtain the DAR matrix is by recording the trajectories of all simulated vehicles in the DNL process. This method iterates across all paths and links over all time intervals, and in each iteration the method computes the number of vehicles arriving at a specific link from a specific path. Since the dimension of DAR matrix increases exponentially with respect to the size of network and the number of time intervals, the naive method is computational implausible for large-scale networks. In this section, we propose a novel method to compute the DAR matrix through the tree-based cumulative curves, and the proposed method is efficient in both computational time (time complexity) and memory (space complexity).

We define $\chi^{kh_1}_{arsi}(\cdot)$ as the tree-based cumulative curve installed at the tail of link a for class-i vehicles departing from path k in OD pair rs in time interval h_1. $\chi^{kh_1}_{arsi}(t)$ takes the time t as input and outputs the total number of vehicles departing in time interval h_1 from path rsk and arriving at link a before time t. Then the DAR can be computed by Eq. (20).

$$\rho^{ka}_{rsi}\left(h_1, h_2\right) = \frac{\chi^{kh_1}_{arsi}\left(\overline{t_2}\right) - \chi^{kh_1}_{arsi}\left(\underline{t_2}\right)}{f^{kh_1}_{rsi}} \tag{20}$$

where $\underline{t_2}$ is the beginning of time interval h_2 and $\overline{t_2}$ is the end of time interval h_2. The tree-based cumulative curves record the path-dependent and departure time-dependent vehicle arrival curves with respect to time at a certain location. That is to say, the tree-based cumulative curves contain multiple cumulative curves, and each cumulative curve records the arrival curve of vehicles following a certain path and departing in a certain time interval.

We note that $\chi^{kh_1}_{arsi}(\cdot)$ records the cumulative curves for each path flow and departing time interval separately, and hence it requires more memory and computational power than the standard link-based cumulative curve (Lu et al., 2013). However, only a very small fraction of vehicles pass a specific link a during the simulation. There are only a small fraction of paths containing a specific link, and hence the $\chi^{kh_1}_{arsi}(\cdot)$ is sparse in terms of path indices rsk and time indices h_1. Using this intuition, we develop a Growing Tree algorithm to build the tree-based cumulative curve $\chi^{kh_1}_{arsi}(\cdot)$ for each link a. Since the algorithm is tree-based, so we refer $\chi^{kh_1}_{arsi}(\cdot)$ as the tree-based cumulative curves. In addition, we only need to install the tree-based cumulative curves on those links that are observed (*i.e.* appear in either $\mathbf{M}_i$ or $\mathbf{L}_i$). This procedure can further reduce the computational time and memory consumption of the proposed framework.

The process of the Growing Tree algorithm is presented in Algorithm 1.

Algorithm 1. Growing Tree algorithm for constructing $\chi_{arsi}^{kh_1}(\cdot)$

```
GrowingTree (S, n);
Input : Traffic simulator S, number of loading intervals n
Output: Tree-based cumulative curves χ
Initialize an empty dictionary χ;
for (i = 0; i < n; ++i) do
    Run the simulator S for one loading interval;
    for a ∈ A do
        Initialize an empty dictionary χ[a];
        Extract the set of vehicles going in link a and denote it as Q;
        for v ∈ Q do
            Suppose vehicle v follows path k in OD pair rs and departs in time interval h_1 and the vehicle
              class is i;
            if i is not the key of dictionary χ[a] then
                Initialize χ[a][i] with an empty dictionary;
            end
            if h_1 is not the key of dictionary χ[a][i] then
                Initialize χ[a][i][h_1] with an empty dictionary;
            end
            if rsk is not the key of dictionary χ[a][i][h_1] then
                Initialize χ[a][i][h_1][rsk] with an empty cumulative curve;
            end
            Add record (i, 1) to the cumulative curve χ[a][i][h_1][rsk];
        end
    end
end
```

In the algorithm, a record $(i, 1)$ means one vehicle arriving at the link a at time t. $\chi_{arsi}^{kh_1}(\cdot)$ is constructed as a tree, as presented in Fig. 4. The x-axis and y-axis of the figure stored at each leaf Fig. 4 are time and cumulative counts, respectively. During the construction, when a vehicle transverses a link, a leaf containing a standard cumulative curve is either created or updated to record the location of that vehicle. The tree $\chi_{arsi}^{kh_1}(\cdot)$ is unbalanced, so a hashmap-based tree is more memory efficient. In Algorithm 1, a dictionary refers to a key-value mapping implemented by hashmap, and readers can view the dictionary as one of the following data structures: `dictionary` in Python, `HashMap` in Java, or `unordered_map` in C++.

3.3. Derivatives of link travel time

In the backward iteration described in Section 2.6, we need to compute the derivatives of link travel time $\frac{\partial\overline{\Lambda}(\{\mathbf{x}_i\}_i)}{\partial\mathbf{x}_i}$. The function $\overline{\Lambda}$ represents the link flow models which include, but is not limited to, point queue, spatial queue, cell transmission model, link transmission model and link queue model (Jin, 2012; Zhang et al., 2013). There is no closed-form for link flow models, hence the derivatives of function $\overline{\Lambda}$ can be challenging to evaluate analytically. In contrast, there exists methods to approximate the link travel time derivatives for the simulation-based models, and the basic idea of this kind of methods is to examine the extra link travel time induced by a marginal vehicle added to the link. Readers are referred to Qian et al. (2012), Lu et al. (2013), Zhang and Qian (2020) for the implementation details. In this paper, we adopt the approximation approach discussed in Lu et al. (2013) to evaluate $\frac{\partial\overline{\Lambda}(\{\mathbf{x}_i\}_i)}{\partial\mathbf{x}_i}$. To be precise, $\frac{\partial\overline{\Lambda}(\{\mathbf{x}_i\}_i)}{\partial\mathbf{x}_i}$ is a zero matrix when all the links are not congested, and $\frac{\partial\overline{\Lambda}(\{\mathbf{x}_i\}_i)}{\partial\mathbf{x}_i} = \texttt{diag}\left(\widetilde{\mathbf{X}}_i^{-1}\right)$ when all the links are congested based on Proposition 1, 2, 3 in Lu et al. (2013). The vector $\widetilde{\mathbf{X}}_i^{-1}$ is the element-wise reciprocal of $\widetilde{\mathbf{X}}_i$, which is the flow exiting from the head of each link for vehicle class i, similar to the definition of $\mathbf{X}_i$. $\texttt{diag}(\widetilde{\mathbf{X}}_i^{-1})$ outputs a square matrix with the diagonal entries being $\widetilde{\mathbf{X}}_i^{-1}$ and other entries being zero. In the actual implementation, each entry of $\frac{\partial\overline{\Lambda}(\{\mathbf{x}_i\}_i)}{\partial\mathbf{x}_i}$ is chosen from either the zero matrix or $\texttt{diag}(\widetilde{\mathbf{X}}_i^{-1})$ based on whether the corresponding link is congested or not.

3.4. Incorporating multi-day observations with multiprocessing

From formulation (13), the multi-class demand is estimated with one data sample $(\mathbf{y}', \mathbf{z}')$. In real world applications, we may observe the flow and travel time on multiple days. Suppose we collect data samples for M days and we let $(\mathbf{y}_m', \mathbf{z}_m')$ denote the observed

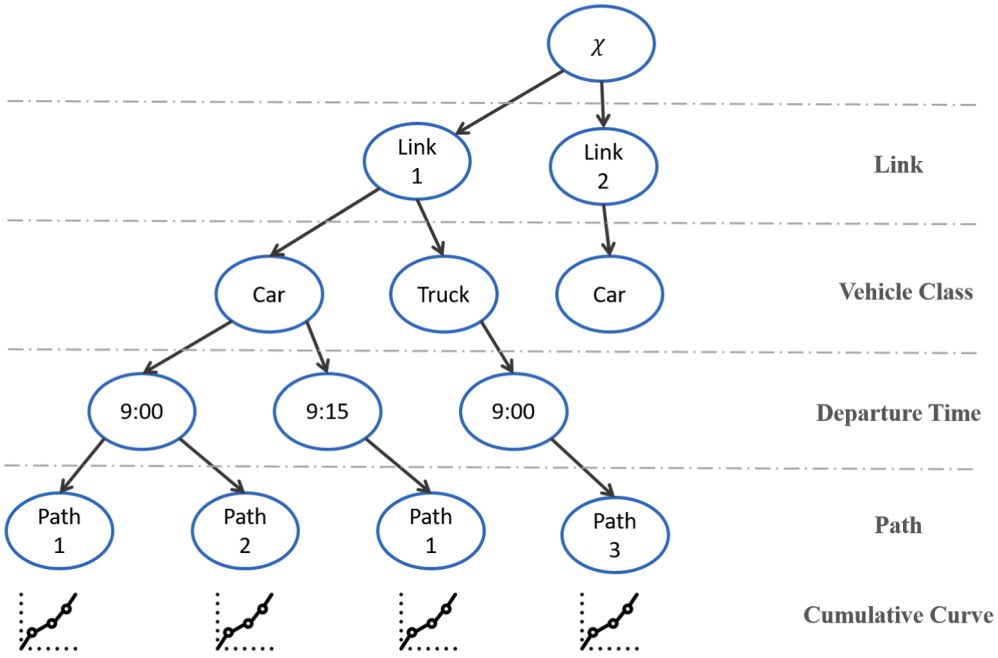

Fig. 4. An illustration of the Growing Tree algorithm.

flow and travel time on day m, then the MCDODE Problem 13 can be extended to accommodate multi-day observations in formulation (21)

$$\begin{aligned}
&\min_{\{\mathbf{q}_i\}_i} \ \frac{1}{M}\sum_{1\leqslant m\leqslant M}\left[w_1\left(\left\|\mathbf{y}'_m-\sum_{i\in D}\mathbf{L}_i\boldsymbol{\rho}_i\mathbf{p}_i\mathbf{q}_i\right\|_2^2\right)+w_2\left(\left\|\mathbf{z}'_m-\sum_{i\in D}\mathbf{M}_i\mathbf{t}_i\right\|_2^2\right)\right]\\
&\text{s.t.} \qquad \{\mathbf{t}_i,\mathbf{c}_i,\boldsymbol{\rho}_i\}_i=\Lambda\left(\{\mathbf{f}_i\}_i\right)\\
&\qquad\quad \mathbf{p}_i=\Psi_i(\{\mathbf{c}_i\}_i,\{\mathbf{t}_i\}_i) \qquad \forall i\in D\\
&\qquad\quad \mathbf{q}_i\geqslant 0 \qquad\qquad\quad \forall i\in D
\end{aligned} \tag{21}$$

Formulation (21) can be directly solved using the forward-backward algorithm. We only need to compute the gradients of OD demand for each data sample and use the average gradient over all data samples to update the OD demand during the backward iteration. This process is the same as Gradient Descent (GD) method. In addition, the stochastic gradient descent (SGD) method can also be used to solve formulation (21). In the process of SGD, we evaluate the GOF for one randomly selected data sample and then use it to update the OD demand. The comparisons between GD and SGD exist in many machine learning models, readers are referred to (Saad, 1998) for more details. Many advanced gradient descent methods can also be used to solve formulation (21). For example, Adagrad is one of the most representatives of variants of SGD with adaptive step sizes, and it is often used in the optimization of deep neural networks (Duchi et al., 2011).

We can further speed up the solution process for formulation (21) by utilizing the power of multiprocessing. The delayed stochastic gradient descent (delayed-SGD) method can evaluate the gradient of multiple data samples on a multi-core CPU at the same time (Zinkevich et al., 2009). Each core is responsible for evaluating one single data sample at one time. Comparing to the traditional SGD, the delayed-SGD makes full use of the multi-core CPU and hence it can solve the MCDODE framework more efficiently. It is also possible to extend Formulation (21) to incorporate multi-day data that are observed on different links. To do that, we can replace $\mathbf{L}_i$ and $\mathbf{M}_i$ to $\mathbf{L}_{im}$ and $\mathbf{M}_{im}$ for each day m separately.

To analyze the efficiency of the multiprocessing framework, we suppose there are $\mathscr{M}$ independent processes on either CPUs ($\mathscr{M}$ cores) or GPUs ($\mathscr{M}$ graphics cards), and one process serves as the master process and the rest processes serve as workers. We further assume the communication time between master and worker is $\mathscr{C}_{\mathrm{C}}$, and it takes $\mathscr{C}_{\mathrm{I}}$ time to finish one iteration within one process. To finish $\mathscr{N}$ iterations, it takes $\mathscr{N}\mathscr{C}_{\mathrm{C}}+\left\lceil\frac{\mathscr{N}}{\mathscr{M}-1}\right\rceil\mathscr{C}_{\mathrm{I}}$ for the multiprocessing framework, and $\mathscr{N}\mathscr{C}_{\mathrm{I}}$ for the single-process framework. Clearly $\mathscr{C}_{\mathrm{I}}\gg\mathscr{C}_{\mathrm{C}}$, and hence the multiprocessing framework can conduct more iterations within a fixed time so as to improve the efficiency for the whole MCDODE framework.

3.5. Solution framework

The solution algorithm for MCDODE is summarized in Table 3.

4. Numerical experiments

In this section, we first examine the proposed MCDODE framework in a small network. Estimation results are presented and discussed. We examine the effects of multiprocessing, compare different variants of gradient descent methods, and conduct the sensitivity analysis of the MCDODE framework. A mid-size network is also used to verify the effectiveness and efficiency of the proposed framework. In addition, the effectiveness, efficiency and scalability of the MCDODE framework are demonstrated in a large-scale network. All the experiments in this section are conducted on a desktop with Intel Core i7-6700 K CPU 4.00 GHz × 8, 2133 MHz 2 × 16 GB RAM, 500 GB SSD.

4.1. A small network

We first work with a small network with seven links, three paths and one O-D pair, as presented in Fig. 5. Two classes of vehicles are considered: cars and trucks. Link 1 and Link 7 are OD connectors, and we use the identical triangular fundamental diagram (FD) for the rest of 5 links. In the FD, length of each road segment is 0.55 mile, free flow speed is 35 miles/hour for car and 25 miles/hour for truck, flow capacity is 2,200 vehicles/hour for car and 1,200 vehicles/hour for truck, and the holding capacity is 200 vehicles/mile for car and 80 vehicles/mile for truck.

We generate the multi-class dynamic OD demand by random number generators and then treat it as the "true" OD demand. We generate the observed flow by running the multi-class network simulation model and then adding the noise. The performance of the MCDODE estimation formulation is assessed by comparing the estimated flow with the "true" flow (flow includes observed flow, link flow, path flow and OD demand) (Antoniou et al., 2015). We use R-square between the "true" flow (travel time) and estimated flow (travel time) to measure the estimation accuracy. Throughout the experiments, we use "flow" to represent OD demand flow, path flow, link flow, and observed flow, while a specific flow is represented by its full name.

Baseline setting: in the small network, we randomly sample the "true" car and truck OD demand from uniform distributions $\texttt{Unif}(0, 300)$ and $\texttt{Unif}(0, 60)$ for each time interval, respectively. We consider 10 time intervals, and each time interval represents fifteen minutes. We randomly generate the route choice portions and treat them as unknown, then we run the MAC-POSTS Λ to obtain the "true" network conditions. We construct the observed flow as follows: firstly we randomly generate the matrix $\{\mathbf{L}_i\}_i$ by a Bernoulli distribution with $p = 0.5$ for link $3, 4, 5, 6$ and leave the flow of link 2 hidden from all observations; secondly we aggregate the "true" link flow to obtain the observed flow by $\{\mathbf{L}_i\}_i$; thirdly we multiply $1+\varepsilon$ to the "true" observed flow to get the observed flow with noise, where $\varepsilon \sim \texttt{Unif}(-\xi, \xi)$ and $\xi \in [0, 1)$ represents the noise level. We set $|B| = 10$, and 6 observations are from car flow and 4 observations are from truck flow. Assuming we directly observe the travel time for link $3, 4, 5, 6$ for cars and trucks separately, $\{\mathbf{M}_i\}_i$ and E can be constructed accordingly. We set $w_1 = 1, w_2 = 0.01$. We also add noise to the observed link travel time using the same method as observed flow. We observe 8 data samples $M = 8$ and the noise level $\xi = 0.1$. We use single-process Adagrad with step size 1 as the solution method, and the initial OD demand is generated from $\texttt{Unif}(0, 15)$ and $\texttt{Unif}(0, 3)$ for car and trucks, respectively. No historical OD information is used. The fixed routing is used in the experiment, and it means the route choice portions are predetermined and unknown. The above setting is called the baseline setting.

To measure the congestion level of the network under the baseline setting, we use the average/minimum/maximum Volume-to-Capacity Ratio (R-C ratio) to represent the congestion level of the studied networks, as shown in Fig. 22.

$$\bar{\nu} = \frac{1}{|H||A|}\sum_{a\in A}\sum_{h_2\in H}\frac{\sum_i \chi_i x_{ai}^{h_2}}{\mathscr{C}_a} \tag{22}$$

$$\nu_{min} = \frac{1}{|A|}\sum_{a\in A}\min_{h_2\in H}\left(\frac{\sum_i \chi_i x_{ai}^{h_2}}{\mathscr{C}_a}\right) \tag{23}$$

Table 3
MCDODE solution framework.

Algorithm	[*MCDODE-FRAMEWORK*]
Step 0	*Initialization.* Initialize the OD demand vector $\{\mathbf{q}_i\}_i$ for each vehicle class.
Step 1	*Forward iteration.* Solve the traffic assignment model presented in Eq. (14) with OD demand $\{\mathbf{q}_i\}_i$, and construct the tree-based cumulative curve χ through Growing Tree algorithm presented in Algorithm 1.
Step 2	*Variable retrieval.* Extract the link/path travel time from the simulation model, compute the route choice matrix from route choice model by Eq. (4), and obtain the DAR matrix from the tree-base cumulative curves by Eq. (20).
Step 3	*Backward iteration.* Compute the GOF using the backward iteration presented in Eqs. (18) and (19).
Step 4	*Update OD demand.* Update the OD demand with the gradient-based projection method discussed in Section 3.4.
Step 5	*Convergence check.* Stop when the change of OD demand $\{\mathbf{q}_i\}_i$ is less than tolerance. Otherwise, go to *Step 1*.

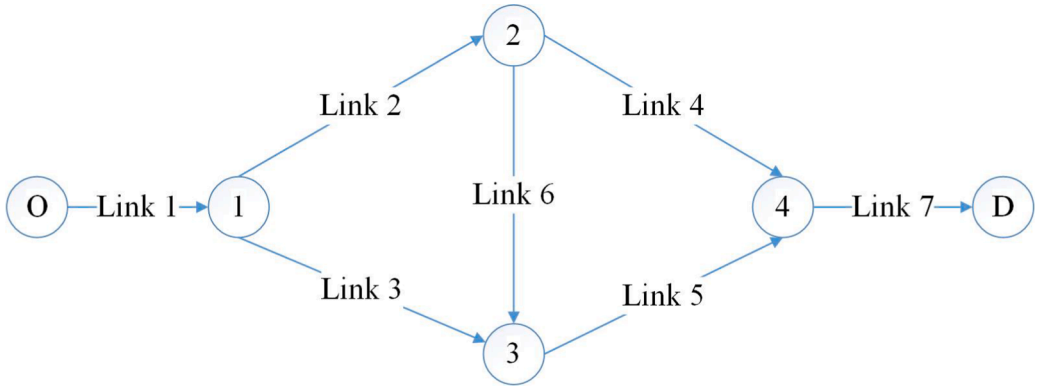

Fig. 5. A small network.

$$\nu_{max} = \frac{1}{|A|}\sum_{a\in A}\max_{h_2\in H}\left(\frac{\sum_i \chi_i x_{ai}^{h_2}}{\mathscr{C}_a}\right) \tag{24}$$

where ξ_i is the conversion factor for vehicle class i, and $\mathscr{C}_a$ is the maximum free-flow capacity of link a. For the baseline setting, $\bar{\nu} = 0.72, \nu_{min} = 0.25, \nu_{max} = 1.17$, the total car demand is 2,432, and the total truck demand is 301.

4.1.1. Basic estimation results

In this section, we examine the basic estimation results of the proposed MCDODE framework for the baseline setting. We run the MCDODE framework presented in Section 3.5 until convergence. The change of loss $\mathscr{L}$ against the number of iterations using Adagrad is presented in Fig. 6.

To analyze the convergence of the observed flow and travel time separately, we decompose the loss $\mathscr{L}$ into four components: car flow, car travel time, truck flow and truck travel time. We plot the loss for the four components separately, and the results are presented in Fig. 7. Note we normalize the loss such that it is between 0 and 1, and the travel cost represents the travel time.

The comparisons between the "true" and estimated values for observed flow, link flow, link travel time and OD demand are presented in Fig. 8–11, respectively.

The R-squares between the "true" and estimated flow (travel time) are presented in Table 4[2] The proposed MCDODE framework yields accurate estimation of the multi-class dynamic OD demand in the small network. The average R-squares for car flow and truck flow are above 0.98. The estimation accuracy for truck is lower than car, which is probably attributed to the low truck demand. Since the multi-class traffic loading model is discretized and stochastic (Qian et al., 2017), low demand may incur a large variance in the simulation results. Therefore, the truck GOF becomes noisy when the demand is low.

The R-square for the travel time is lower than that for flows, since the derivative of travel time $\frac{\partial\bar{\Lambda}(\{\mathbf{x}_i\}_i)}{\partial\mathbf{x}_i}$ is approximated by the simulation rather than evaluated in a closed-form. Again, due to the discretization and stochasticity of the simulation model, the approximations of $\frac{\partial\bar{\Lambda}(\{\mathbf{x}_i\}_i)}{\partial\mathbf{x}_i}$ can be noisy.

4.1.2. Comparing with the SPSA method

To further demonstrate efficiency of the proposed MCDODE framework, we compare with the SPSA method, as presented in Fig. 12. One can see the MCDODE outperforms the SPSA by a significant margin. The convergence curve of SPSA is unstable, due to the stochastic nature of the SPSA algorithm. SPSA usually performs well in single-class small networks, however, it cannot estimate the multi-class dynamic OD demand in small networks, which indicates that MCDODE is more challenging to solve than the DODE problem.

4.1.3. Comparing different gradient-based methods

In this section, we examine the performance of three gradient-based methods: gradient descent (GD), stochastic gradient descent (SGD) method and Adagrad. We solve the MCDODE problem for the baseline setting three times using different gradient-based methods, and we plot the convergence curves for the three methods in Fig. 13.

For all three method, it takes less than 2 min to complete the 100 iterations. One can clearly see that SGD outperforms the GD throughout the 100 iterations. Though GD and SGD converge faster in the first 20 iterations, the Adagrad outperforms both methods in terms of convergence rate and final loss after 100 iterations. The reason for the best performance of Adagrad is probably because Adagrad can select the step size adaptively during the solution, hence it maintains a good convergence rate throughout the 100

[2] The R-square is computed by $1-\frac{\sum\widehat{Y}_i^2}{\sum Y_i^2}$ where $\widehat{Y}_i$ is the ith estimated value and Y_i is the ith true value.

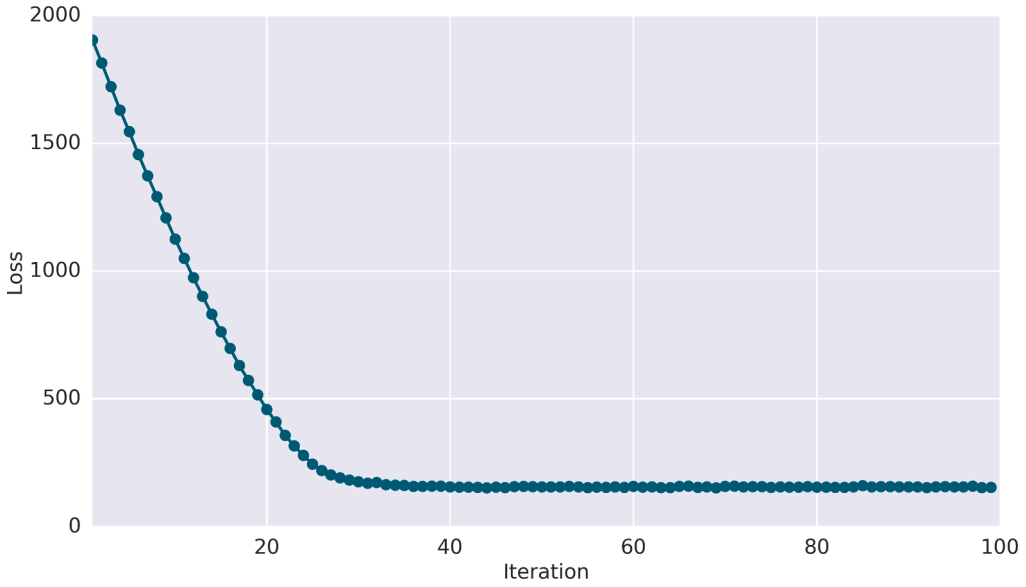

Fig. 6. Convergence curve for the loss $\mathscr{L}$.

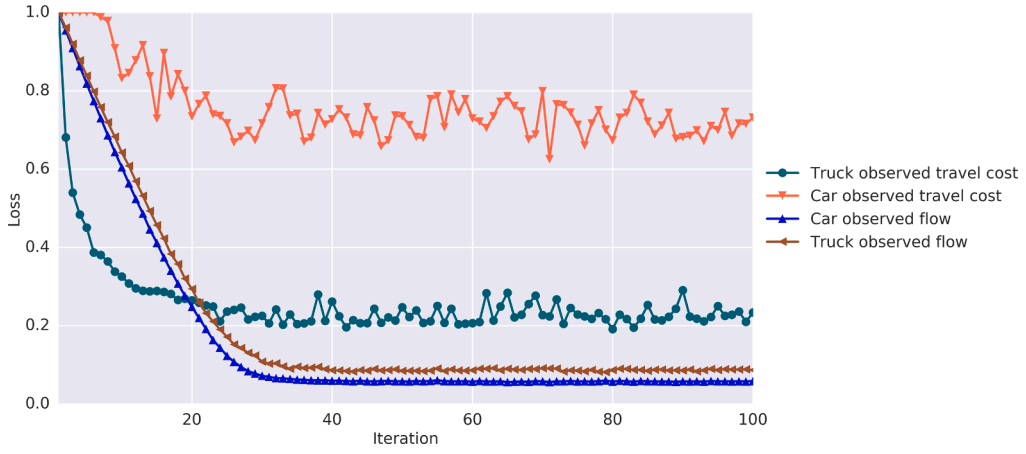

Fig. 7. Decomposed convergence curve for observed flow and travel time (normalized).

iterations. Adagrad is adopted as the standard method to solve the MCDODE formulation in the baseline setting.

4.1.4. *Multiprocessing*

In this section, we demonstrate the power of multiprocessing in solving the MCDODE framework. We solve the baseline setting four times using different number of processes. We use the delayed version of Adagrad to enable the multiprocessing (Zinkevich et al., 2009). We examine the convergence curves for 1, 2, 4, and 8 processes and plot the results in Fig. 14. We note that different from previous figures, the x-axis in Fig. 14 is the time rather than iterations.

All four methods converges to the same optimal solution. One can see that the 1-process method converges in 50s, 2-process method converges in 30s, and the 4-process method converges in $20 \sim 25$ s. By using the multiprocessing, the solving time for the MCDODE framework can be reduced by at least a half. The marginal effects of adding processes decreases when the number of processes increases, as a result of the increasing communication costs among different processes. Since we conduct the experiments on an eight-core CPU, the eight-process method makes use of all computing resources of the CPU and hence it achieves the best convergence rate.

4.1.5. *Impact of data quantity*

In this section, we analyze the impact of number of data samples on the MCDODE framework. We solve for the baseline setting, while the number of data samples varies from 1 to 256. We keep track of the convergence curves for different number of data samples, the results are plotted in Fig. 15.

As can be seen from Fig. 15, the convergence rate increases when the number of data sample increases. The solution method with

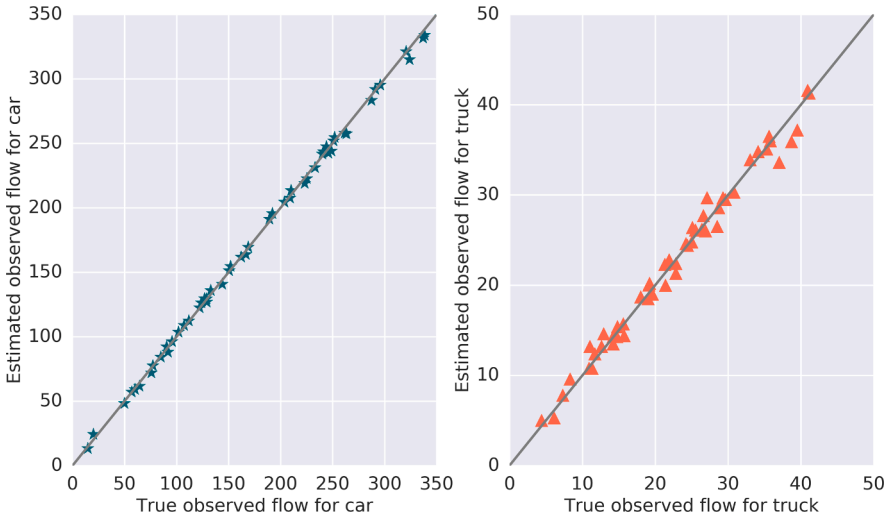

Fig. 8. Estimated and "true" observed flow for cars and trucks (unit:vehicle/15mins).

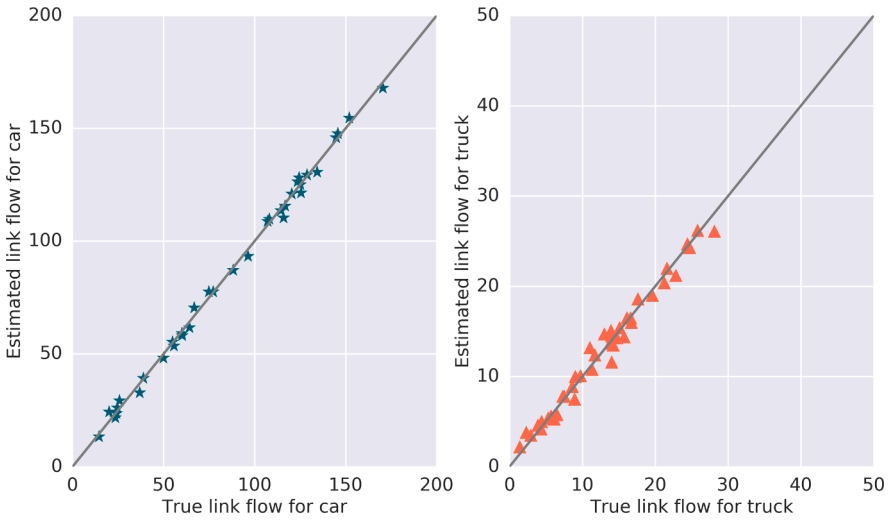

Fig. 9. Estimated and "true" link flow for cars and trucks (unit:vehicle/15mins).

one data sample does not converge in 100 iterations while the solution method with 256 data samples converges within 10 iterations. In the large-scale networks, limited number of iterations can be conducted due to the lack of computational resources and time constraints, hence more data samples are usually required to ensure estimation accuracy.

4.1.6. Impact of noise level

We further demonstrate the impact of noise level on the proposed solution algorithm. We solve the MCDODE framework for the baseline setting, while we change the noise level from 0 to 0.9. The convergence curves under different noise levels are presented in Fig. 16.

One can see from Fig. 16 that the noise level has a significant impact on the estimation accuracy of the multi-class OD demand. When there is no noise in the observed data, the estimation method can converges to nearly zero loss in 30 iterations. This is because the observation noise can be averaged out by the central limit theorem. When the noise level is high, the estimation method does not converge to the optimal solution. Especially when noise level is 0.9, the loss can only be reduced by half and the R-square for the estimated OD is around 0.7.

4.1.7. Sensitivity analysis

In this section, we conduct the sensitivity analysis of the proposed MCDODE framework in terms of initial OD demand, "true" OD demand and step sizes.

Firstly, we perform the sensitivity analysis on the initial OD demand. We solve the MCDODE framework for the baseline setting for

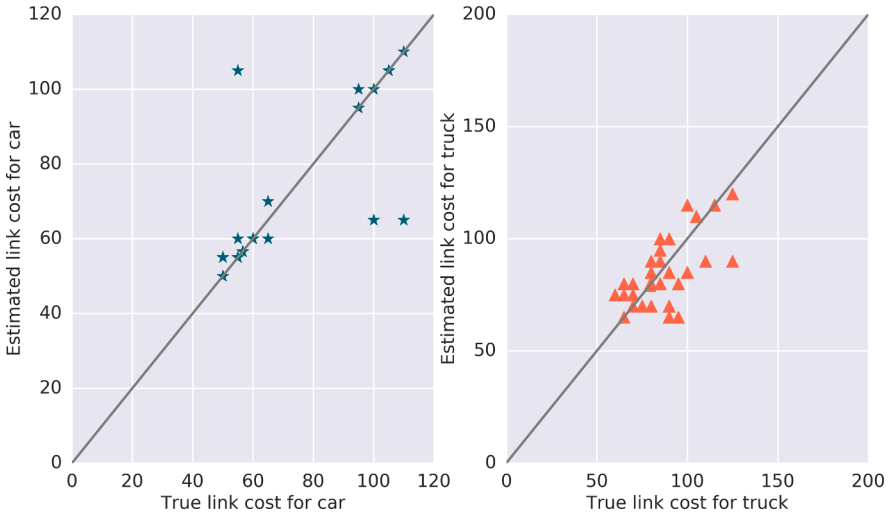

Fig. 10. Estimated and "true" link travel time for cars and trucks (unit:seconds).

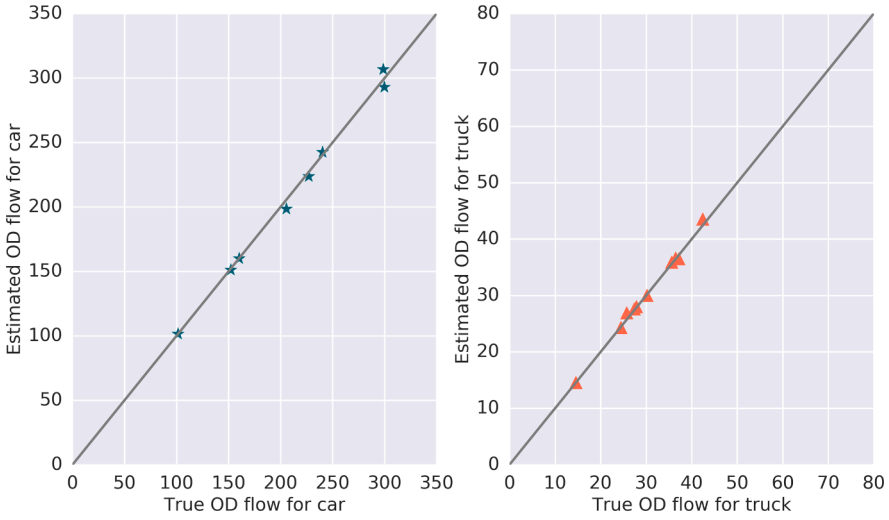

Fig. 11. Estimated and "true" OD demand for cars and trucks (unit:vehicle/15mins).

Table 4
R-square between the "true" and estimated flow and travel time for cars and trucks.

	Car	Truck
Observed flow	0.9992	0.9858
Link flow	0.9982	0.9808
Link travel time	0.9309	0.9586
OD demand	0.9965	0.9940

100 times. In each time, we change the initial OD demand by random generators $\texttt{Unif}(0, 15)$ for car demands and $\texttt{Unif}(0, 3)$ for truck demands. We compute the R-squares of the estimated OD demand for all 100 estimations and the boxplot for the R-squares is presented in Fig. 17.

As can be seen from Fig. 17, the R-squares for car flow are above 0.98 and the variance is low for different initial OD demand, while the variance of R-squares for truck flow is high. Especially for link flow, the R-square between the "true" and estimated link flow for truck can be as low as 0.95. However, all the R-squares for truck flow are still above 0.9. The results imply that the proposed method is generally robust to the initial OD demand, but estimating the truck demand is more challenging than estimating the car demand. To ensure a satisfactory estimation result, it is desirable to run the MCDODE framework multiple times and chose the best one as the final OD demand. In contrast, the R-square of link travel time for car is lower than that for trucks, which is probably because car speeds can

Fig. 12. Comparison between the proposed MCDODE with SPSA.

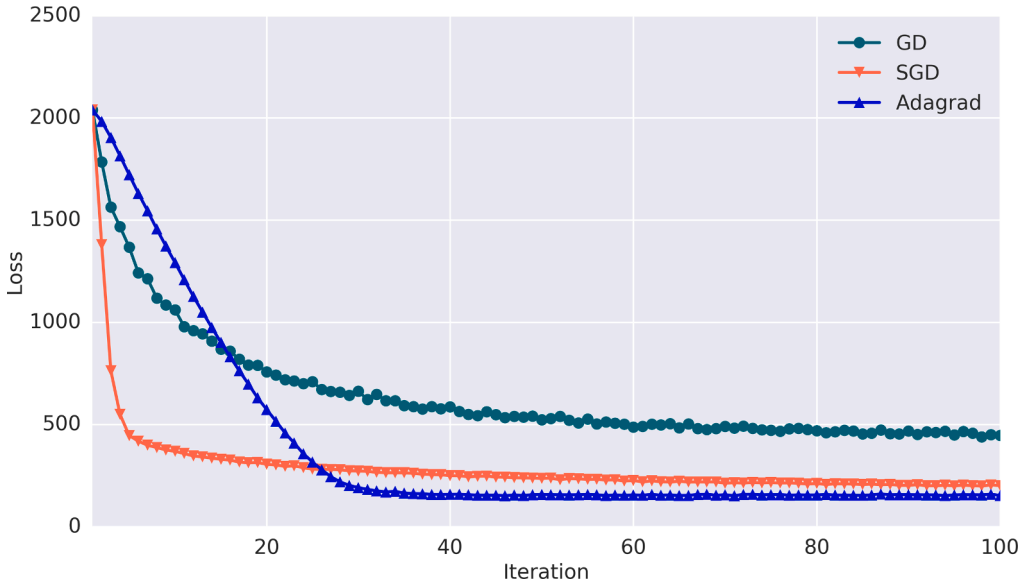

Fig. 13. Comparison among GD, SGD and Adagrad.

vary in a wide range but truck speed is relatively stable.

Secondly, we fix the initial OD demand and solve for the baseline setting for 100 times. In each run, we sample the "true" OD demand from $\texttt{Unif}(0, 300)$ and $\texttt{Unif}(0, 60)$ for car and truck, respectively. We compute the R-squares of estimated OD demand for the 100 runs and the boxplot for the R-squares is presented in Fig. 18.

The MCDODE framework achieves satisfactory accuracy for different "true" OD demands, and most of the R-squares for both car and truck flow are over 0.9. The R-square for truck flow is lower than that for car flow, and the variance of R-squares for truck flow is also higher. As we discussed above, estimating the truck OD demand is more challenging, as a result of the discretized and stochastic behaviors of trucks in the traffic simulation model. In addition, the R-squares of link travel time for cars are lower than that for trucks, which is similar to previous study for the initial OD demand.

Thirdly, we examine the Adagrad method with different step sizes. We solve the MCDODE framework for 7 times under the baseline setting. In each run, we vary the step size from 0.01 to 100, and the convergence curves are presented in Fig. 19.

One can see that the Adagrad is robust to the step size, and any step size between 0.5 and 2 can guarantee the method to converge to the optimal solution within 60 iterations. When the step size is too small, the method converges slowly; when the step is too large, the

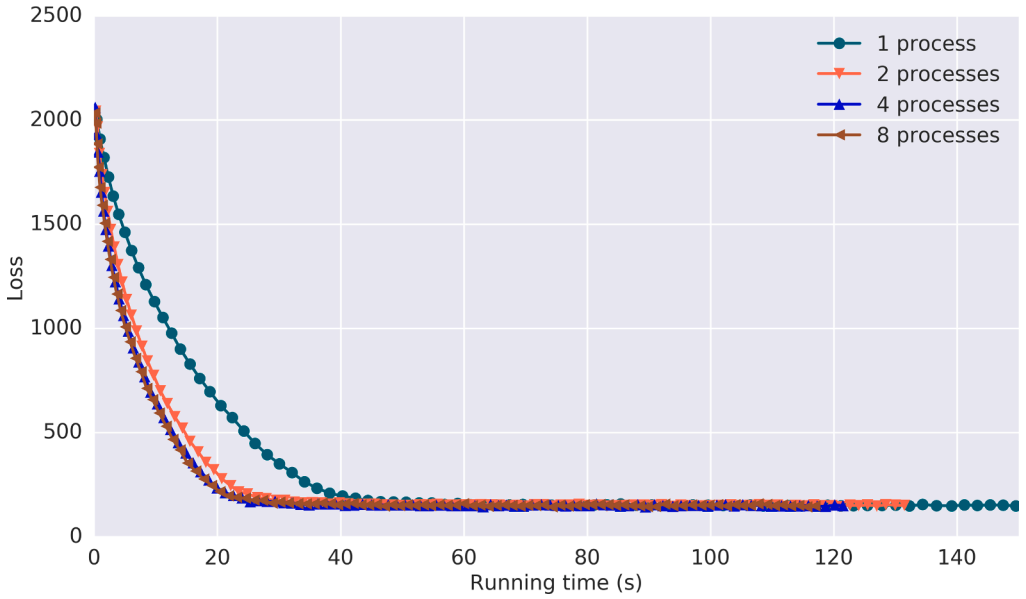

Fig. 14. Convergence curves using 1, 2, 4, and 8 processes.

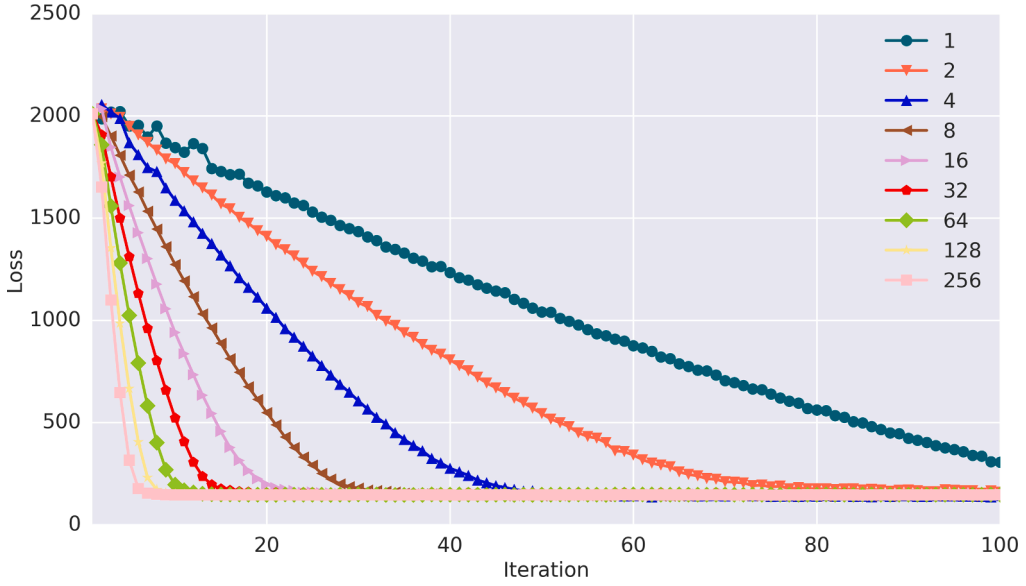

Fig. 15. Convergence curves with different number of data samples.

convergence becomes unstable and fluctuating.

4.2. A mid-size network: SR-41 corridor

In this section, we evaluate the MCDODE framework on the SR-41 corridor. The SR-41 corridor network is located in the City of Fresno, California. An overview of the network network is presented in Fig. 20, and it contains 2,413 links and 7,110 O-D pairs. The study period contains 10 intervals and each interval represents 15 min.

We randomly generate the "true" dynamic path flow from $\texttt{Unif}(0, 0.5)$ and $\texttt{Unif}(0, 0.1)$ for car and truck, respectively, and the "true" dynamic OD demand can be computed accordingly. The initial OD is generated from $\texttt{Unif}(0, 0.01)$ and $\texttt{Unif}(0, 0.001)$ for car and truck, respectively. We assume 500 links are installed with cameras and the traffic count for each vehicle class can be observed separately, and only traffic count data (no speed data) is used in this experiment. The number of data samples, observation noise level, route choice model and configurations of the solution algorithms are the same as the baseline setting. In the SR-41 setting, $\bar{v} = 0.86$,

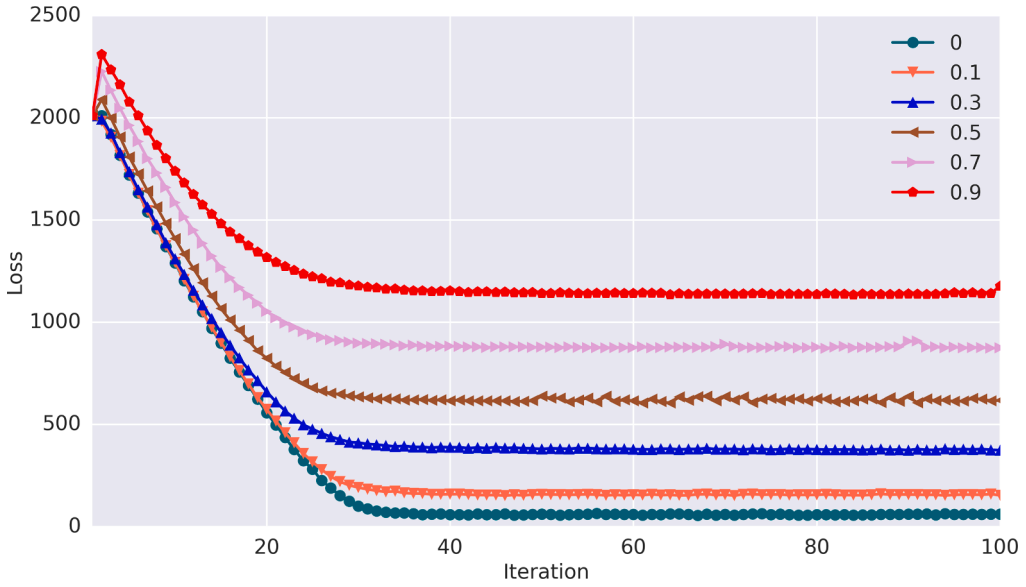

Fig. 16. Convergence curves under different noise levels.

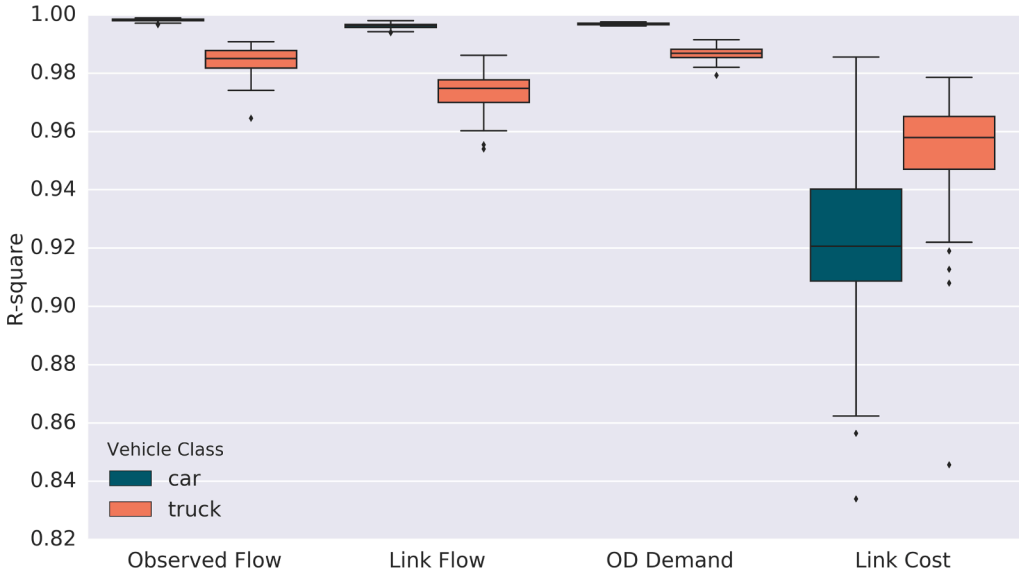

Fig. 17. Boxplot for R-squares with different initial OD demand.

$\nu_{min} = 0.65, \nu_{max} = 1.05$, the total car demand is 69,380, and the total truck demand is 11,877.

We run the MCDODE framework with 100 iterations, and the convergence curve for both car and truck is presented in Fig. 21. The comparison between the "true" and estimated values for observed flow is presented in Fig. 22. We omit the results for the other comparisons as the results are similar to the small network.

One can read the proposed method converges quickly on SR-41 corridor network. The R-square for car and truck flow is 0.975 and 0.912, respectively. The results indicate that the proposed framework can obtain the dynamic MCDOD accurately and efficiently.

4.2.1. Comparing with the SPSA method

The proposed MCDODE framework is also compared with SPSA method in SR-41 corridor network, and the convergence curve is presented in Fig. 23. Similar to the results in Section 4.1.2, the SPSA converges quickly while the loss is still high, and the proposed MCDODE framework outperform the SPSA method by a significant margin.

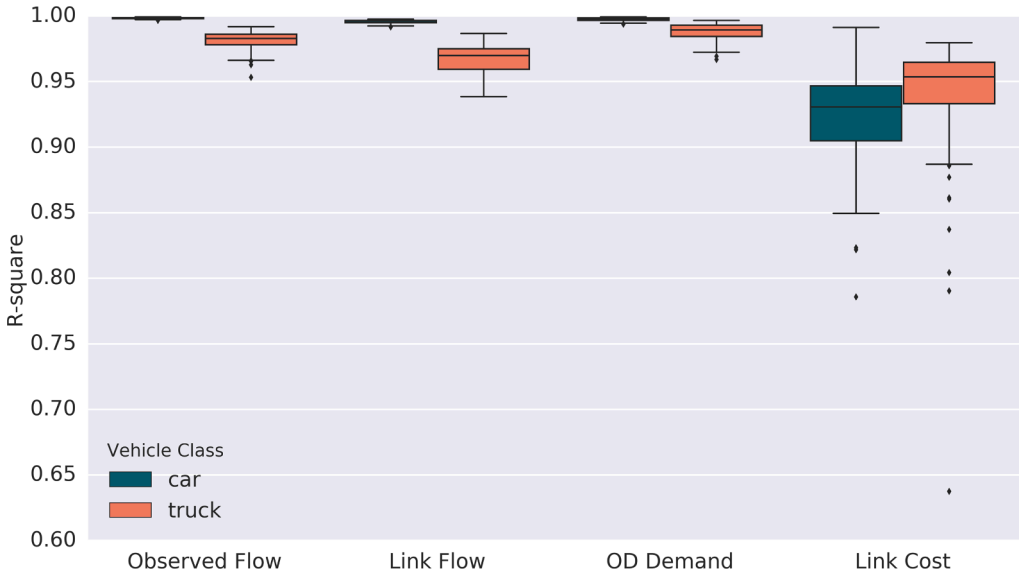

Fig. 18. Boxplot for R-squares with different "true" OD demand.

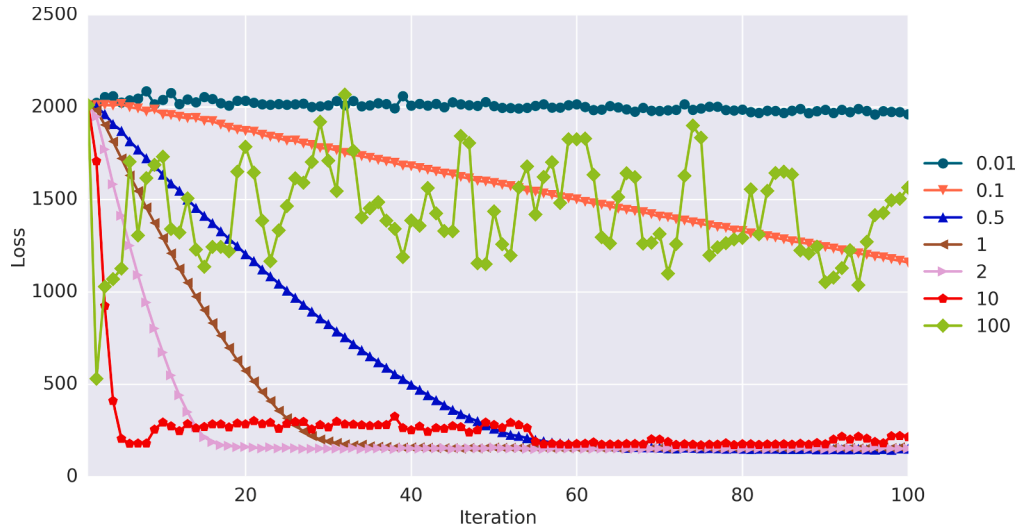

Fig. 19. Convergence curves with different step sizes.

4.3. A large-scale network: southwestern Pennsylvania region

In this section, we perform the MCDODE on a large-scale network for the southwestern Pennsylvania region (Fig. 24). The network covers ten counties of southwestern Pennsylvania region, with the Pittsburgh city located in the center. The approximate range of the 10 counties are marked by the black quadrangle in Fig. 24. There are around 2.57 million population and 7,112 square miles area in the network. All parameters for the Pittsburgh network are listed in the Table 5.

We run the MCDODE framework with traffic flow data and travel time (traffic speed) data. The traffic flow data are obtained from the Pennsylvania Department of Transportation (PennDOT). The data are collected annually for some selected locations on the Pennsylvania state-owned roads, for each hour of the day and for one day of the year. The car traffic volume counts and truck traffic volume counts are collected separately, where car traffic volume counts are measured for all passenger cars and truck traffic volume counts includes all kinds of trucks at the measured location. The traffic count can be either one-directional or bi-directional. Since all count data are measured in hours, in data pre-processing we smooth the hourly count data to 15-min interval. There are 608 locations in total that has valid car and truck volume counts for MCDODE. Traffic speed data are obtained Federal Highway Administration (FHWA) for the year 2016. The speed data are observed every 5-min of the day for highway segments, the data are also classified to cars and trucks. We aggregate the travel time data to 15-min interval. In total, there are 945 locations with valid car and truck travel time

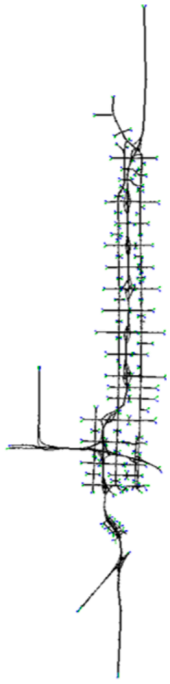

Fig. 20. The California SR-41 corridor network.

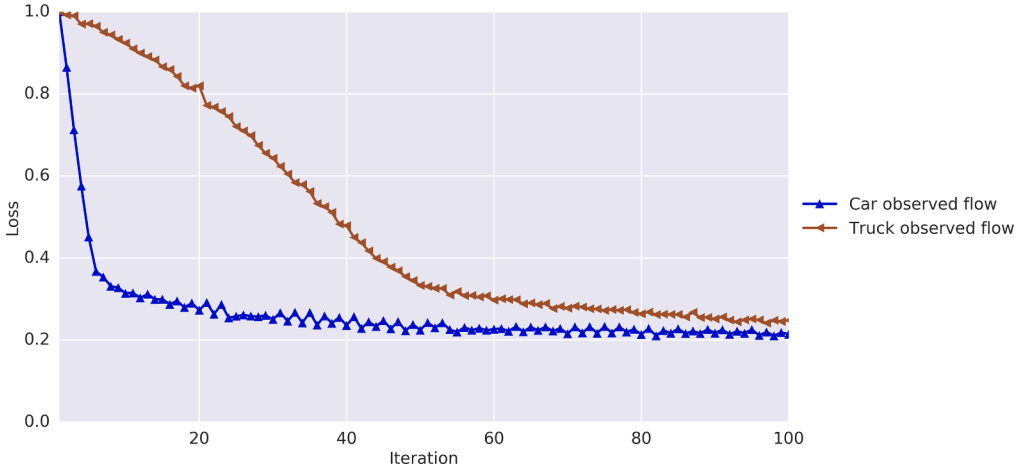

Fig. 21. Decomposed convergence curve for observed flow for car and truck (normalized).

observations.

The initial OD demands for cars and trucks are randomly generated from a uniform distribution $\texttt{Unif}(0, 0.01)$ and $\texttt{Unif}(0,0.001)$, respectively. We aggregate all the traffic flow and travel time observations to a single data sample and use the single-process Adagrad method to solve MCDODE. The step size is e^{-5}. We set $w_1 = 1, w_2 = 0.01$. We use the hybrid dynamic traffic assignment model as the route choice model (Qian and Zhang, 2013), the adaptive ratio is 0.2 for cars and 0 for trucks. The MCDODE framework runs for 55 iterations. In first 35 iterations, both OD demand for car and trucks are updated simultaneously; while for the last 20 iterations, we only update the truck demand since its more challenging to estimate.

The convergence of the proposed MCDODE framework is presented in Fig. 25. Overall, the solution algorithm performs well and the objective function converges fast. It takes around $25 \times 55 = 1375$ minutes (around 23 h) to complete the 55 iterations. For each iteration, the traffic simulation takes around 7 min, and the other 18 min are used for the demand estimation. Constructing DAR matrix is identified as the bottleneck in the demand estimation.

The comparisons for the observed flow are presented in Fig. 26, and R-squares are 0.66 and 0.59 for car and truck, respectively. One can see that the truck flow is roughly one tenth of the car flow, and the estimation accuracy for car flow is higher than truck flow.

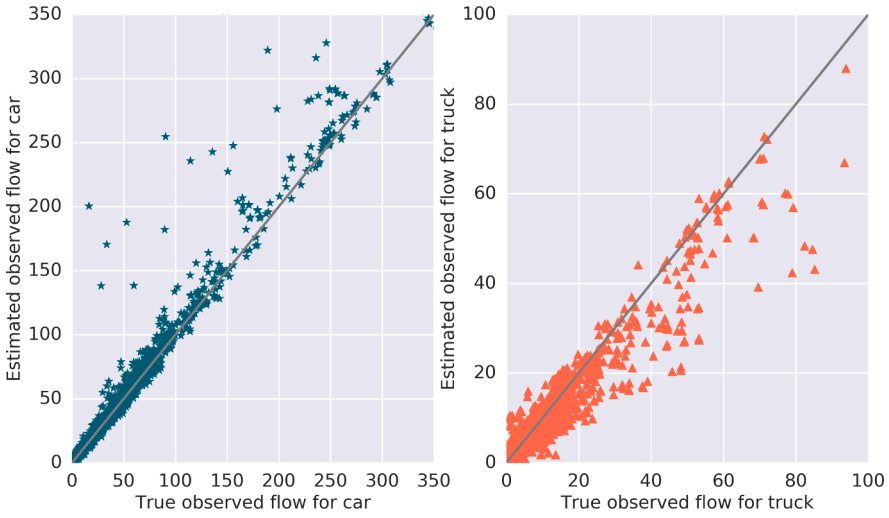

Fig. 22. Estimated and "true" observed flow for cars and trucks (unit:vehicle/15mins).

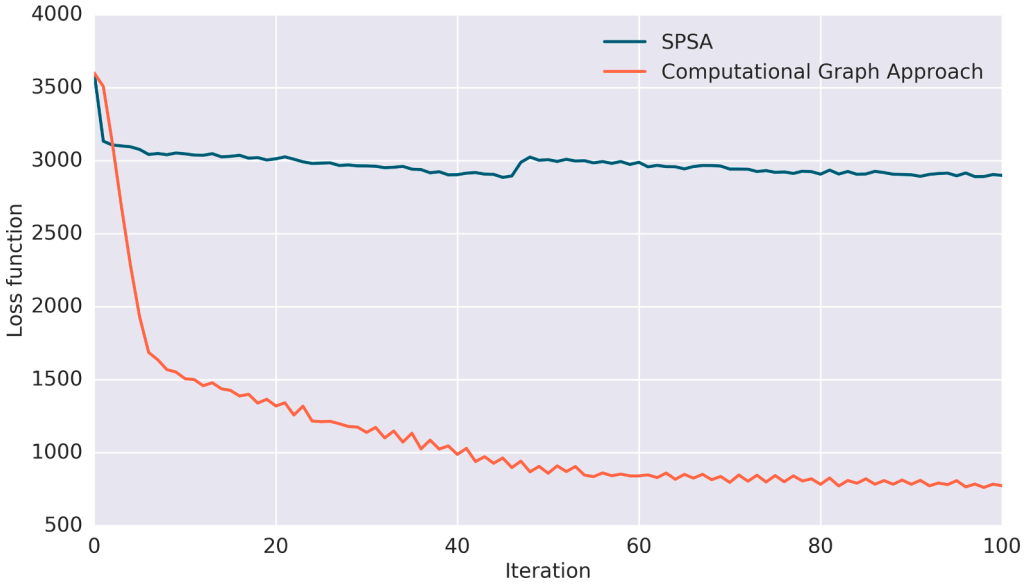

Fig. 23. Comparison between the proposed MCDODE with SPSA in SR-41.

Overall, the results of the MCDODE framework is compelling and satisfactory for such a large network. Based on the estimated traffic in SPC network, $\bar{\nu} = 0.42, \nu_{min} = 0.18, \nu_{max} = 0.77$, the total car demand is 252,586, and the total truck demand is 20,625.

We also plot the comparisons between the observed and estimated link speed in Fig. 27. The reason we plot the link speed instead of link travel time is that the link travel time can vary in a wide range for large scale networks, but link speed usually varies between 20 to 80 miles/hour. Hence, visualizing the link speed is more straightforward and legible.

One can read from Fig. 27 that the R-squares between the observed and estimated link speed are 0.45 and 0.51 for car and truck, respectively. The estimation accuracy of traffic speed is not as good as the traffic flow, which is probably attributed to unsatisfactory approximations of the link travel time derivatives $\frac{\partial \overline{\Lambda}(\{\mathbf{x}_i\}_i)}{\partial \mathbf{x}_i}$ in dynamic networks. Improving the approximation quality for the link travel time will be left for future research. In addition, there is more stochasticity in the bi-class traffic simulation models, and the speed is very sensitive to the stochasticity.

In addition, the estimation uses the randomly generated OD demand as the initial point for the MCDODE framework. If prior knowledge of the OD demand can be obtained from traditional planning models, the estimation accuracy might further be improved. The estimated multi-class OD demand is further used to study the impact of a development project in Pittsburgh, and details are presented in Pi et al. (2018).

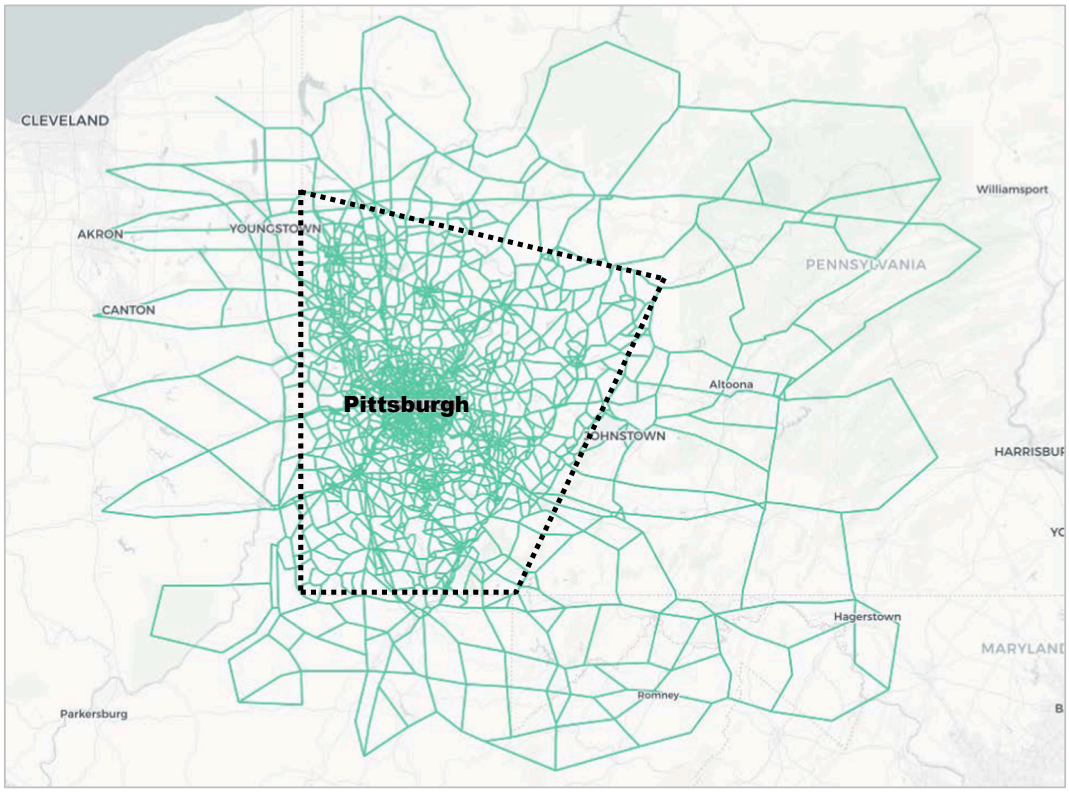

Fig. 24. An overview of the network for the southwestern Pennsylvania region.

Table 5
Network parameters.

Name	Value
studying period (weekday)	6:00 AM - 11:00 AM
simulation unit time	5 s
Length of time interval	15 min
Number of time intervals	30
number of links	16,110
number of nodes	6,297
number of origins (destinations)	283
number of origin-destination pairs	80,089

5. Conclusion

This paper presents a data-driven framework for multi-class dynamic origin-destination demand estimation (MCDODE) using observed traffic flow and travel time data. The traffic data can be any linear combinations of flow characteristics (e.g. counts, time or travel time) across vehicle classes, road segments and time intervals. All the characteristics involved in the MCDODE formulation are vectorized, and the proposed framework is represented on a computational graph. The computational graph can be solved efficiently through a forward-backward algorithm for large-scale MCDODE problems. In the forward iteration, the dynamic traffic assignment problem is solved, and the loss (objective function) is computed through a series of equations. In the backward iteration, the OD demand is updated by the backpropagation method with the route choice matrix, DAR matrix and route travel time known from the forward iteration. The MCDODE formulation is solved when the forward-backward algorithm converges. The proposed algorithm resembles the standard iterative heuristic method to solve for the bi-level optimization problems, while the new formulation provides a novel perspective to view the MCDODE problem as a machine learning problem.

Practical issues related to MCDODE framework are discussed. We adopt a mesoscopic multi-class traffic simulation package MAC-POSTS to solve for the spatio-temporal path/link flow. The DAR matrix is highly sparse, and thus we propose novel tree-based cumulative curves from MAC-POSTS to construct the sparse DAR matrix. We incorporate multi-day observation data to the MCDODE framework, and different variants of gradient-based solution algorithms are discussed and compared.

The proposed MCDODE framework is examined on a small network, a mid-size network as well as a real-world large-scale network.

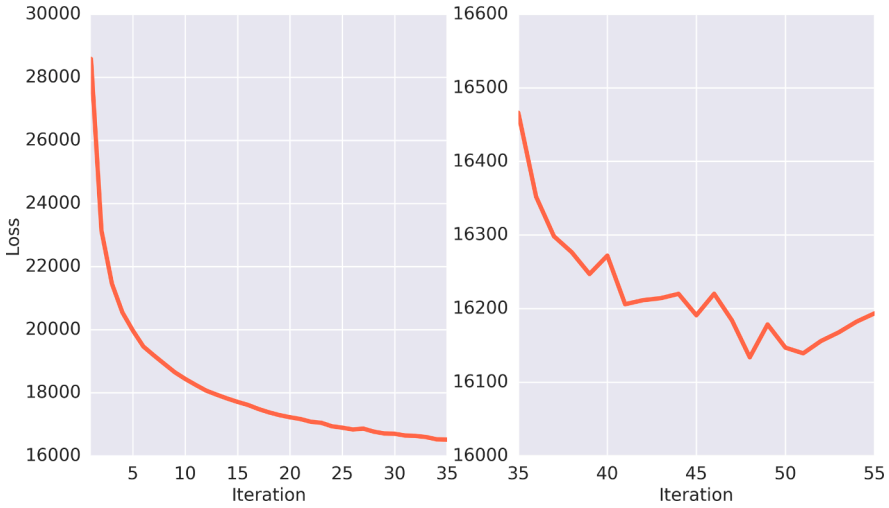

Fig. 25. Convergence of the objective function for 55 iterations (OD demands for cars and trucks are updated simultaneously in first 35 iterations, while only truck OD demand is updated in last 20 iterations.).

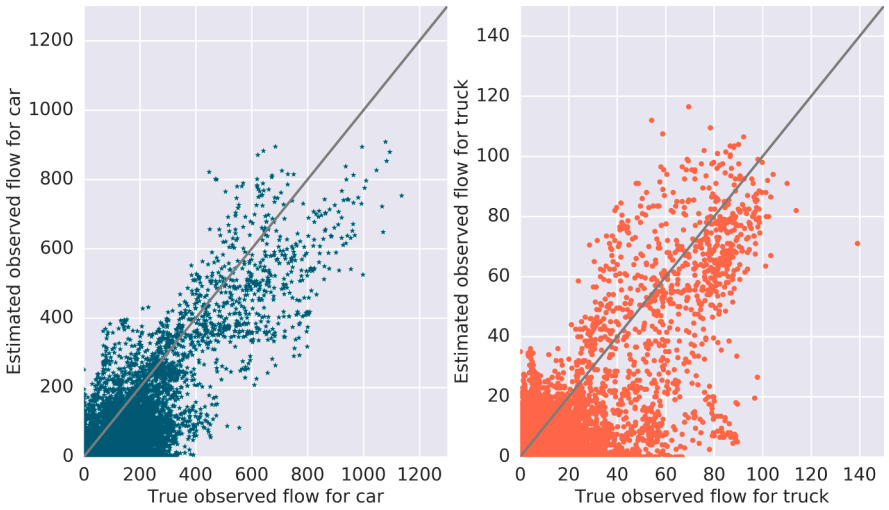

Fig. 26. Estimated and observed flow for cars and trucks (unit:vehicle/15mins).

The objective function converges quickly with the Adagrad method. We also conduct the sensitivity analysis of the estimation accuracy with respect to initial OD demand, "true" OD demand, and step sizes. Overall, the estimation results are compelling, satisfactory and robust, and the forward-backward algorithm is computationally plausible for large-scale networks. The estimated multi-class dynamic OD demand can help policymakers to better understand the dynamics of OD demand and the spatio-temporal distribution of vehicles in terms of different classes.

In the near future, we plan to improve the estimation accuracy of the MCDODE framework in the following two directions: 1) some prior knowledge about the dynamic OD demand can be used as the initial point for the solution methods. For example, we can construct the DAR matrix from the speed data directly and estimate the dynamic OD without running the simulation. The estimated OD demand can be used as the initial point for the proposed MCDODE framework (Ma and Qian, 2018b); 2) the method of approximating the derivatives of link travel time under multi-class flow can be further improved by implementing more accurate approximation methods, e.g. (Qian et al., 2012; Zhang and Qian, 2020). In addition, we plan to extend this research to calibrate the parameters in route choice models as well as the fundamental diagrams, thanks to the versatile computational graph framework. We also plan to extend this research to estimate the probabilistic distribution of multi-class dynamic OD demand and explore the spatio-temporal characteristics of dynamic OD demand.

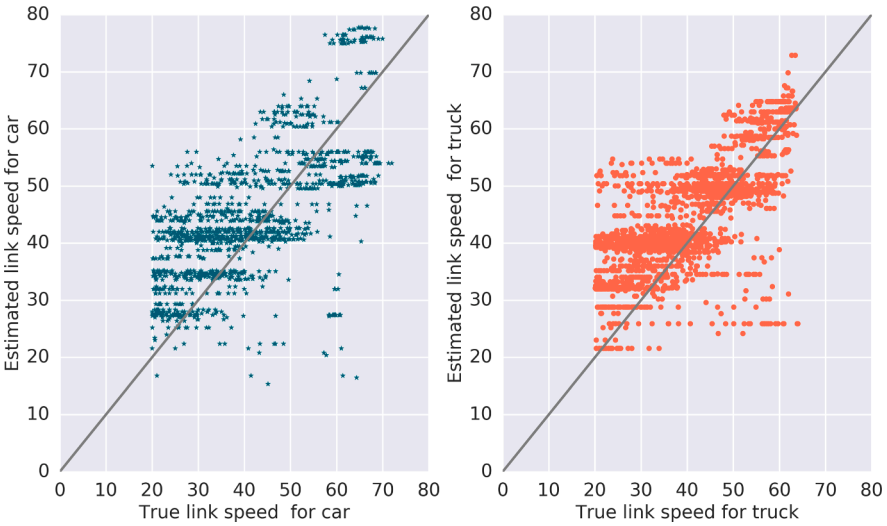

Fig. 27. Estimated and observed flow for cars and trucks (unit:miles/hour).

Supplementary materials

The mesoscopic multi-class traffic simulation package MAC-POSTS [3] and the MCDODE framework [4] are implemented and open-sourced on Github.

Acknowledgments

We wish to thank the anonymous reviewers for the valuable comments, suggestions, and discussions. This research is supported by U.S. NSF Award CMMI-1751448, and by U.S. Department of Energy Vehicle Technology Office Grant DE-0008466.

References

Antoniou, C., Azevedo, C.L., Lu, L., Pereira, F., Ben-Akiva, M., 2015. W-spsa in practice: Approximation of weight matrices and calibration of traffic simulation models. Transport. Res. Part C: Emerg. Technol. 59, 129–146.

Antoniou, C., Barceló, J., Breen, M., Bullejos, M., Casas, J., Cipriani, E., Ciuffo, B., Djukic, T., Hoogendoorn, S., Marzano, V. et al., 2015, Towards a generic benchmarking platform for origin-destination flows estimation/updating algorithms: Design, demonstration and validation. Transport. Res. Part C: Emerg. Technol.

Ashok, K., Ben-Akiva, M.E., 2000. Alternative approaches for real-time estimation and prediction of time-dependent origin-destination flows. Transport. Sci. 34 (1), 21–36.

Balakrishna, R., Ben-Akiva, M., Koutsopoulos, H., 2008. Time-dependent origin-destination estimation without assignment matrices. In: Second International Symposium of Transport Simulation (ISTS06). Lausanne, Switzerland. 4-6 September 2006. EPFL Press.

Bauer, D., Richter, G., Asamer, J., Heilmann, B., Lenz, G., Kölbl, R., 2017. Quasi-dynamic estimation of od flows from traffic counts without prior od matrix. IEEE Trans. Intell. Transp. Syst. 19 (6), 2025–2034.

Ben-Akiva, M., Bierlaire, M., Koutsopoulos, H., Mishalani, R., 1998. Dynamit: a simulation-based system for traffic prediction. In: DACCORD Short Term Forecasting Workshop, pp. 1–12.

Ben-Akiva, M.E., Gao, S., Wei, Z., Wen, Y., 2012. A dynamic traffic assignment model for highly congested urban networks. Transport. Res. Part C: Emerg. Technol. 24, 62–82.

Bierlaire, M., Crittin, F., 2004. An efficient algorithm for real-time estimation and prediction of dynamic od tables. Oper. Res. 52 (1), 116–127.

Cantelmo, G., Cipriani, E., Gemma, A., Nigro, M., 2014. An adaptive bi-level gradient procedure for the estimation of dynamic traffic demand. IEEE Trans. Intell. Transp. Syst. 15 (3), 1348–1361.

Cantelmo, G., Viti, F., Cipriani, E., Nigro, M., 2018. A utility-based dynamic demand estimation model that explicitly accounts for activity scheduling and duration. Transport. Res. Part A: Policy Practice 114, 303–320.

Cantelmo, G., Viti, F., Tampère, C.M., Cipriani, E., Nigro, M., 2014. Two-step approach for correction of seed matrix in dynamic demand estimation. Transp. Res. Rec. 2466 (1), 125–133.

Cascetta, E., Inaudi, D., Marquis, G., 1993. Dynamic estimators of origin-destination matrices using traffic counts. Transport. Sci. 27 (4), 363–373.

Cascetta, E., Papola, A., Marzano, V., Simonelli, F., Vitiello, I., 2013. Quasi-dynamic estimation of o–d flows from traffic counts: Formulation, statistical validation and performance analysis on real data. Transport. Res. Part B: Methodol. 55, 171–187.

Cipriani, E., Del Giudice, A., Marialisa, N., Viti, F., Cantelmo, G., 2015. The impact of route choice modeling on dynamic od estimation. In: 2015 IEEE 18th International Conference on Intelligent Transportation Systems. IEEE, pp. 1483–1488.

Cipriani, E., Florian, M., Mahut, M., Nigro, M., 2011. A gradient approximation approach for adjusting temporal origin-destination matrices. Transport. Res. Part C: Emerg. Technol. 19 (2), 270–282.

[3] https://github.com/Lemma1/MAC-POSTS.

[4] https://github.com/Lemma1/MAC-POSTS/tree/master/src/examples/mcDODE.

Corthout, R., Tampère, C., Frederix, R., Immers, L., 2011. Marginal dynamic network loading for large-scale simulation-based applications. In: Proceedings of the 90th Annual Meeting of the Transportation Research Board.
Dafermos, S.C., 1972. The traffic assignment problem for multiclass-user transportation networks. Transport. Sci. 6 (1), 73–87.
Davis, G.A., 1994. Exact local solution of the continuous network design problem via stochastic user equilibrium assignment. Transport. Res. Part B: Methodol. 28 (1), 61–75.
Djukic, T., Flötteröd, G., Van Lint, H., Hoogendoorn, S., 2012. Efficient real time od matrix estimation based on principal component analysis. In: 2012 15th International IEEE Conference on Intelligent Transportation Systems. IEEE, pp. 115–121.
Duchi, J., Hazan, E., Singer, Y., 2011. Adaptive subgradient methods for online learning and stochastic optimization. J. Machine Learn. Res. 12 (Jul), 2121–2159.
Fisk, C., 1989. Trip matrix estimation from link traffic counts: the congested network case. Transport. Res. Part B: Methodol. 23 (5), 331–336.
Florian, M., Chen, Y., 1995. A coordinate descent method for the bi-level o–d matrix adjustment problem. Int. Trans. Oper. Res. 2 (2), 165–179.
Flötteröd, G., Bierlaire, M., Nagel, K., 2011. Bayesian demand calibration for dynamic traffic simulations. Transport. Sci. 45 (4), 541–561.
Frederix, R., Viti, F., Corthout, R., Tampère, C., 2011. New gradient approximation method for dynamic origin-destination matrix estimation on congested networks. Transport. Res. Rec.: J. Transport. Res. Board 2263, 19–25.
Frederix, R., Viti, F., Tampère, C.M., 2013. Dynamic origin-destination estimation in congested networks: theoretical findings and implications in practice. Transportmetrica A: Transport Sci. 9 (6), 494–513.
Gal, Y., Ghahramani, Z., 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In: International Conference on Machine Learning, pp. 1050–1059.
Gundaliya, P., Mathew, T.V., Dhingra, S.L., 2008. Heterogeneous traffic flow modelling for an arterial using grid based approach. J. Adv. Transport. 42 (4), 467–491.
Hazelton, M.L., 2008. Statistical inference for time varying origin-destination matrices. Transport. Res. Part B: Methodol. 42 (6), 542–552.
Huang, H.-J., Li, Z.-C., 2007. A multiclass, multicriteria logit-based traffic equilibrium assignment model under atis. Eur. J. Oper. Res. 176 (3), 1464–1477.
Jha, M., Gopalan, G., Garms, A., Mahanti, B., Toledo, T., Ben-Akiva, M., 2004. Development and calibration of a large-scale microscopic traffic simulation model. Transport. Res. Rec.: J. Transport. Res. Board (1876), 121–131.
Jin, W.-L., 2012. A link queue model of network traffic flow. arXiv preprint arXiv:1209.2361.
Krishnakumari, P., van Lint, H., Djukic, T., Cats, O., 2019. A data driven method for od matrix estimation. Transport. Res. Part C: Emerg. Technol..
LeBlanc, L.J., Farhangian, K., 1982. Selection of a trip table which reproduces observed link flows. Transport. Res. Part B: Methodol. 16 (2), 83–88.
Lee, J.-B., Ozbay, K., 2009. New calibration methodology for microscopic traffic simulation using enhanced simultaneous perturbation stochastic approximation approach. Transport. Res. Rec.: J. Transport. Res. Board 2124, 233–240.
Lu, C.-C., Zhou, X., Zhang, K., 2013. Dynamic origin-destination demand flow estimation under congested traffic conditions. Transport. Res. Part C: Emerg. Technol. 34, 16–37.
Lu, L., Xu, Y., Antoniou, C., Ben-Akiva, M., 2015. An enhanced spsa algorithm for the calibration of dynamic traffic assignment models. Transport. Res. Part C: Emerg. Technol. 51, 149–166.
Ma, J., Zhang, H.M., et al., 2008. A polymorphic dynamic network loading model. Comput.-Aided Civil Infrastruct. Eng. 23 (2), 86–103.
Ma, R., Ban, X.J., Pang, J.-S., 2014. Continuous-time dynamic system optimum for single-destination traffic networks with queue spillbacks. Transport. Res. Part B: Methodol. 68, 98–122.
Ma, W., Pinchao, Z., Qian, Z., 2016. Dynamic network analysis & real-time traffic management for philadelphia metropolitan area. Technical Report for a Sponsor Research with PennDOT.
Ma, W., Qian, Z., 2018a. A generalized single-level formulation for origin-destination estimation under stochastic user equilibrium. Transp. Res. Rec. 2672 (48), 58–68.
Ma, W., Qian, Z.S., 2018b. Estimating multi-year 24/7 origin-destination demand using high-granular multi-source traffic data. Transport. Res. Part C: Emerg. Technol. 96, 96–121.
Maher, M.J., Zhang, X., Van Vliet, D., 2001. A bi-level programming approach for trip matrix estimation and traffic control problems with stochastic user equilibrium link flows. Transport. Research Part B: Methodological 35 (1), 23–40.
Mahmassani, H., Herman, R., 1984. Dynamic user equilibrium departure time and route choice on idealized traffic arterials. Transport. Sci. 18 (4), 362–384.
Mahmassani, H., Hu, T., Jayakrishnan, R., 1992. Dynamic traffic assignment and simulation for advanced network informatics (dynasmart). In: Proceedings of the 2nd international CAPRI seminar on Urban Traffic Networks, Capri, Italy.
Neyshabur, B., Tomioka, R., Srebro, N., 2014. In search of the real inductive bias: On the role of implicit regularization in deep learning. arXiv preprint arXiv:1412.6614.
Nguyen, S., 1977. Estimating and OD Matrix from Network Data: A Network Equilibrium Approach. Université de Montréal, Centre de recherche sur les transports, Montréal.
Nie, Y.M., Zhang, H.M., 2008. A variational inequality formulation for inferring dynamic origin-destination travel demands. Transport. Res. Part B: Methodol. 42 (7), 635–662.
Nie, Y.M., Zhang, H.M., 2010. Solving the dynamic user optimal assignment problem considering queue spillback. Networks Spatial Econ. 10 (1), 49–71.
Noriega, Y., Florian, M., 2007. Multi-class demand matrix adjustment.
Osorio, C., 2019. Dynamic origin-destination matrix calibration for large-scale network simulators. Transport. Res. Part C: Emerg. Technol. 98, 186–206.
Pan, S.J., Yang, Q., 2010. A survey on transfer learning. IEEE Trans. Knowledge Data Eng. 22 (10), 1345–1359.
Patriksson, M., 2004. Sensitivity analysis of traffic equilibria. Transport. Sci. 38 (3), 258–281.
Peeta, S., Ziliaskopoulos, A.K., 2001. Foundations of dynamic traffic assignment: The past, the present and the future. Networks Spatial Econ. 1 (3–4), 233–265.
Pi, X., Ma, W., Qian, S., 2018. Regional traffic planning and operation using mesoscopic car-truck traffic simulation. Technical report for Traffic21 Institute.
Pi, X., Ma, W., Qian, Z.S., 2019. A general formulation for multi-modal dynamic traffic assignment considering multi-class vehicles, public transit and parking. Transport. Res. Part C: Emerg. Technol. 104, 369–389.
Prashker, J.N., Bekhor, S., 2004. Route choice models used in the stochastic user equilibrium problem: a review. Transport Rev. 24 (4), 437–463.
Qian, Z.S., Li, J., Li, X., Zhang, M., Wang, H., 2017. Modeling heterogeneous traffic flow: A pragmatic approach. Transport. Res. Part B: Methodol. 99, 183–204.
Qian, Z.S., Shen, W., Zhang, H., 2012. System-optimal dynamic traffic assignment with and without queue spillback: Its path-based formulation and solution via approximate path marginal cost. Transport. Res. Part B: Methodol. 46 (7), 874–893.
Qian, Z.S., Zhang, H.M., 2013. A hybrid route choice model for dynamic traffic assignment. Networks Spatial Econ. 13 (2), 183–203.
Qian, Z., Zhang, H.M., 2011. Computing individual path marginal cost in networks with queue spillbacks. Transport. Res. Rec. 2263 (1), 9–18.
Raothanachonkun, P., Sano, K., Matsumoto, S., 2006. Estimation of multiclass origin-destination matrices using genetic algorithm. Infrastruct. Plann. Rev. 23, 423–432.
Rumelhart, D.E., Hinton, G.E., Williams, R.J., 1985. Learning internal representations by error propagation, Technical report. California Univ San Diego La Jolla Inst for Cognitive Science.
Saad, D., 1998. 'Online algorithms and stochastic approximations', Online. Learning 5.
Shao, H., Lam, W.H., Sumalee, A., Hazelton, M.L., 2015. Estimation of mean and covariance of stochastic multi-class od demands from classified traffic counts. Transport. Res. Part C: Emerg. Technol.
Shen, W., Nie, Y., Zhang, H.M., 2007. On path marginal cost analysis and its relation to dynamic system-optimal traffic assignment. In: Transportation and Traffic Theory 2007. Papers Selected for Presentation at ISTTT17.
Sun, J., Guo, J., Wu, X., Zhu, Q., Wu, D., Xian, K., Zhou, X., 2019. Analyzing the impact of traffic congestion mitigation: From an explainable neural network learning framework to marginal effect analyses. Sensors 19 (10), 2254.
Tavana, H., 2001. Internally-consistent estimation of dynamic network origin-destination flows from intelligent transportation systems data using bi-level optimization.

Tympakianaki, A., Koutsopoulos, H.N., Jenelius, E., 2015. c-spsa: Cluster-wise simultaneous perturbation stochastic approximation algorithm and its application to dynamic origin-destination matrix estimation. Transport. Res. Part C: Emerg. Technol. 55, 231–245.
Vaze, V., Antoniou, C., Wen, Y., Ben-Akiva, M., 2009. Calibration of dynamic traffic assignment models with point-to-point traffic surveillance. Transport. Res. Rec.: J. Transport. Res. Board 2090, 1–9.
Venkatesan, K., Gowri, A., Sivanandan, R., 2008. Development of microscopic simulation model for heterogeneous traffic using object oriented approach. Transportmetrica 4 (3), 227–247.
Wong, S., Tong, C., Wong, K., Lam, W.K., Lo, H., Yang, H., Lo, H., 2005. Estimation of multiclass origin-destination matrices from traffic counts. J. Urban Plann. Develop. 131 (1), 19–29.
Wu, X., 2018. Data flow programming model to express traffic demand estimation problem based on tensorflow.
Wu, X., Guo, J., Xian, K., Zhou, X., 2018. Hierarchical travel demand estimation using multiple data sources: A forward and backward propagation algorithmic framework on a layered computational graph. Transport. Res. Part C: Emerg. Technol. 96, 321–346.
Wu, X., Zhou, X.S., 2018. A short tutorial of deep learning and computational graph frameworks in transportation modeling.
Yang, H., 1995. Heuristic algorithms for the bilevel origin-destination matrix estimation problem. Transport. Res. Part B: Methodol. 29 (4), 231–242.
Yang, H., Huang, H.-J., 2004. The multi-class, multi-criteria traffic network equilibrium and systems optimum problem. Transport. Res. Part B: Methodol. 38 (1), 1–15.
Yang, H., Sasaki, T., Iida, Y., Asakura, Y., 1992. Estimation of origin-destination matrices from link traffic counts on congested networks. Transport. Res. Part B: Methodol. 26 (6), 417–434.
Yang, Y., Fan, Y., Wets, R.J., 2018. Stochastic travel demand estimation: Improving network identifiability using multi-day observation sets. Transport. Res. Part B: Methodol. 107, 192–211.
Zhang, C., Osorio, C., Flötteröd, G., 2017. Efficient calibration techniques for large-scale traffic simulators. Transport. Res. Part B: Methodol. 97, 214–239.
Zhang, H., Nie, Y., Qian, Z., 2008. Estimating time-dependent freeway origin-destination demands with different data coverage: Sensitivity analysis. Transport. Res. Rec.: J. Transport. Res. Board (2047), 91–99.
Zhang, H., Nie, Y., Qian, Z., 2013. Modelling network flow with and without link interactions: the cases of point queue, spatial queue and cell transmission model. Transportmetrica B: Transport Dyn. 1 (1), 33–51.
Zhang, P., Qian, S., 2020. Path-based system optimal dynamic traffic assignment: A subgradient approach. Transport. Res. Part B: Methodol. 134, 41–63.
Zhao, Q., Turnquist, M.A., Dong, Z., He, X., 2018. Multiclass probit-based origin-destination estimation using multiple data types. J. Transport. Eng., Part A: Syst. 144 (6), 04018018.
Zhou, X., Mahmassani, H.S., 2007. A structural state space model for real-time traffic origin-destination demand estimation and prediction in a day-to-day learning framework. Transport. Res. Part B: Methodol. 41 (8), 823–840.
Zhou, X., Qin, X., Mahmassani, H., 2003. Dynamic origin-destination demand estimation with multiday link traffic counts for planning applications. Transport. Res. Rec.: J. Transport. Res. Board 1831, 30–38.
Zhou, X., Taylor, J., 2014. Dtalite: A queue-based mesoscopic traffic simulator for fast model evaluation and calibration. Cogent Eng. 1 (1), 961345.
Zhou, Z., Chen, A., Bekhor, S., 2012. C-logit stochastic user equilibrium model: formulations and solution algorithm. Transportmetrica 8 (1), 17–41.
Zinkevich, M., Langford, J., Smola, A.J., 2009. Slow learners are fast. In: Advances in Neural Information Processing Systems, pp. 2331–2339.