MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Joint Data-Driven Estimation of Origin-Destination Demand and Travel Latency Functions in Multi-Class Transportation Networks

Wollenstein-Betech, Salomon; Sun, Chuangchuang; Zhang, Jing; Cassandras, Christos G.; Paschalidis, Ioannis Ch.

TR2022-078 August 06, 2022

Abstract

The Traffic Assignment Problem (TAP) is a widely used formulation for designing, analyzing, and evaluating transportation networks. The inputs to this model, besides the network topology, are the Origin-Destination (OD) demand matrix and travel latency cost functions. It has been observed that small perturbations to these inputs have a large impact on the solution. However, most efforts on estimating these using data do so separately and are typically based on parametric models or surveys. In this paper, we present a kernel-based framework that jointly estimates the OD demand matrix and travel latency function in single and multi-class vehicle networks. To that end, we formulate a bilevel optimization problem and then we transform it to a Quadratic Constraint Quadratic Program (QCQP). To solve this QCQP, we propose a trust-region feasible direction algorithm that sequentially solves a quadratic program. In addition, we also provide an alternating optimization method. Our results show that the QCQP method achieves better estimates when compared with disjoint and sequential methods. We show the applicability of the method by performing case studies using data for the transportation networks of Eastern Massachusetts and New York City.

IEEE Transactions on Control of Network Systems 2022

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

Joint Data-Driven Estimation of Origin-Destination Demand and Travel Latency Functions in Multi-Class Transportation Networks

Salomón Wollenstein-Betech[1], Chuangchuang Sun[2], Jing Zhang[3], Christos G. Cassandras[1], and Ioannis Ch. Paschalidis[1].

***Abstract*—The Traffic Assignment Problem (TAP) is a widely used formulation for designing, analyzing and evaluating transportation networks. The inputs to this model, besides the network topology, are the Origin-Destination (OD) demand matrix and travel latency cost functions. It has been observed that small perturbations to these inputs have a large impact on the solution. However, most efforts on estimating these using data do so separately and are typically based on parametric models or surveys. In this paper, we present a kernel-based framework that jointly estimates the OD demand matrix and travel latency function in single and multi-class vehicle networks. To that end, we formulate a bilevel optimization problem and then we transform it to a Quadratic Constraint Quadratic Program (QCQP). To solve this QCQP, we propose a *trust-region feasible direction* algorithm that sequentially solves a quadratic program. In addition, we also provide an alternating optimization method. Our results show that the QCQP method achieves better estimates when compared with disjoint and sequential methods. We show the applicability of the method by performing case studies using data for the transportation networks of Eastern Massachusetts and New York City.**

***Index Terms*—Multi-class transportation networks; variational inequalities; inverse optimization; travel latency function; congestion function estimation; demand estimation.**

I. Introduction

THE continuous increase of traffic congestion in urban areas, together with the availability of massive amounts of speed data, has motivated researchers, companies, and policymakers to improve urban mobility through traffic control. A fundamental step when implementing new transportation policies or infrastructure projects is to create models that help us understand the outcomes of an intervention. One of the most common tools to analyze transportation networks —and the one used in this paper— is the Traffic Assignment Problem (TAP) which requires, apart from the network topology, two main inputs: (i) an *Origin Destination (OD)* demand matrix; and (ii) a link *latency cost* or *travel time* function that relates traffic flows with travel times. The *user-centric* TAP assumes that users selfishly choose the best available route instead of cooperating towards a social objective, which results in an equilibrium known as the *Wardrop equilibrium*. Modeling commuters' routing behavior under this assumption is widely-used for the purpose of analyzing transportation networks, with applications in traffic diagnosis, control, and optimization.

An important drawback that has been observed when using the user-centric TAP is that small perturbations in the inputs to the problem, namely the OD demands and travel time functions have a large impact on the equilibrium solution [1], [2]. Therefore, the problem of accurately estimating these inputs is relevant to design interventions with reliable models. This joint estimation problem, that relies on traffic counts data, can be formally written as

$$\min_{\boldsymbol{\beta},\mathbf{g}} \; F_1(\mathbf{x}(\boldsymbol{\beta},\mathbf{g}),\mathbf{x}^d),$$

where F_1 measures a non-negative "distance" between the TAP flows $\mathbf{x}(\mathbf{g},\boldsymbol{\beta})$ and the flow data $\mathbf{x}^d$; $\mathbf{g}$ represents the OD demand; $\boldsymbol{\beta}$ represents the parameters specifying a travel latency function and $\mathbf{x}(\mathbf{g},\boldsymbol{\beta})$ is the solution to the TAP.

Literature Review: The problem of estimating $\mathbf{g}$ alone (for a fixed $\boldsymbol{\beta}$) has received extensive attention. In practice, urban planners estimate OD demand patterns through surveys. This task is expensive and time consuming, which makes it impractical to perform on a regular basis. In contrast, academics have estimated $\mathbf{g}$ assuming access to flow data and using a variety of methods. An appropriate way to classify the literature on OD demand estimation is by considering the level of congestion in the network. For uncongested networks, where path enumeration connecting each OD pair is possible and flows do not impact the routing decisions of users, entropy maximization [3], [4], multi-objective optimization [5], generalized least squares [6], [7], maximum likelihood estimation [8], and Bayesian inference [9] have been employed. See [10] for a comparison between these approaches. Alternatively, for congested networks, where there exist a circular dependence between the OD estimation problem and the traffic assignment problem, the problem has been posed as a bilevel problem. The first works to formulate this bilevel problem were [11] and [12] and they tackled it by sequentially solving an entropy maximization and a user assignment problem. After that, [13] proposed a gradient-descent heuristic algorithm that calculates derivatives by assuming that routing decisions are locally

Research partially supported by the NSF under grants IIS-1914792,DMS-1664644, ECCS-1931600, and CNS-1645681, by the ONR under grant N00014-19-1-2571, by the NIH under grants R01 GM135930 and UL54 TR004130. This article solely reflects the opinions and conclusions of its authors and not NSF, TRI, or any other entity.

[1]The authors are with the Division of Systems Engineering and the Center for Information and Systems Engineering, Boston University, Boston, MA 02215, USA. {salomonw, yannisp, cgc}@bu.edu

[2]The author is with the Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA. Work performed while at Boston University. ccsun1@mit.edu

[3]The author is with Mitsubishi Electric Research Laboratories, Cambridge, MA, USA. jingzhang@merl.com

constant for small perturbations of $\mathbf{g}$ and solved the problem sequentially. Since then, many approaches have been proposed for the congested case, including coordinate descent [14], fixed point algorithms with generalized least squares [15], and extensions to a stochastic user equilibrium setting [16], [17] that incorporates uncertainty in the routing decision process. In addition to these, [18] solves the problem by writing the augmented Largrangian problem and solving it using a Frank-Wolfe algorithm. However all of these approaches assume full knowledge of $\boldsymbol{\beta}$.

Estimating *travel time functions* has received less attention than the OD demand. In general, this problem has been addressed by fitting data of a particular link using regression [19], [20], simulation [21], and more recently by incorporating stochasticity in the link's capacity [22]. In contrast to single-link methods, there has been some work trying to solve the *inverse* TAP which assumes knowledge of $\mathbf{x}$ and $\mathbf{g}$ and aims to recover $\boldsymbol{\beta}$. In this context [23] estimates a single parameter β and uses a column-generation alternating approach. A different method that stands out is [2]. Here, a kernel method is used, allowing more flexibility in the specification of the travel latency function. This last method serves as the basis of our work.

To the best of our knowledge, solving the joint (OD demand and travel time functions) estimation problem has only been studied in [24] where the authors considered the parametric BPR-type function and a heuristic algorithm to solve the joint problem. Different to [24], in this paper we allow more flexibility in the travel latency function by specifying it as any monotonically increasing polynomial. Similar to the joint problem, several studies have estimated the OD demand together with the route-choice *dispersion parameter* θ encountered in the stochastic user equilibrium (SUE). [25] used a two-stage method and showed its convergence to a local minimum for uncongested networks. For the congested case, [1] proposed a sequential quadratic procedure that requires estimating flow derivatives with respect to θ and $\mathbf{g}$. To compute the derivatives they used [26] and showed that their algorithm converges only under certain circumstances. Moreover, [27] and [28] solved the same problem using an alternating method, and [29] tackled the problem with an augmented Lagrangian method.

The contribution of this paper is threefold. First, we tackle the problem of jointly estimating (instead of separately) the OD demands and travel latency functions using exclusively network flow data. To achieve this, we consider a bilevel problem which we solve using two different methodologies. One involves an alternating method that solves sequentially the two estimation problems. The second entails including the first-order necessary conditions of the lower-level problem as constraints to the upper-level problem. This transforms the original bilevel program to a quadratically constraint quadratic program (QCQP) for which we provide a relaxation and an iterative trust-region feasible direction method that seeks to find a local minimum. Our second contribution is the extension of these methods from single- to multi-class vehicle networks which allows us to estimate $\mathbf{g}$ and $\boldsymbol{\beta}$ for different vehicle types such as self-driving vehicles, trucks, or bikes. Our third contribution is to perform experiments and case studies using real urban and suburban transportation networks and speed data. In particular, we use the Eastern Massachusetts Area (EMA) and New York City (NYC) transportation networks and we observe that our joint methods converge to a solution that yields better estimates than solving the two estimation problems separately.

Building on our preliminary analysis in [30], in this paper: (i) We extend our method from single- to multi-class transportation networks. To that end, we use the methodology introduced in [31] and discuss its implications in the joint demand-cost estimation problem. (ii) We propose an alternating (apart from the QCQP) optimization method and compare the performance of the two methods against the framework of estimating $\mathbf{g}$ while keeping $\boldsymbol{\beta}$ constant, and with a vanilla gradient-descent method. This is relevant, as it demonstrates the benefits of our approach and helps us assess its advantages/disadvantages against other methodologies. (iii) Unlike [30], we perform a series of experiments using real transportation networks. To accomplish this, we gathered the network topology together with large volumes of historical speed data. We transformed the speed data into flow data using the fundamental diagram of traffic flow (see Appendix B for details). Then, we estimated the OD demand matrix and travel latency functions for the EMA highway network and the NYC urban network. For both settings, we report the estimated OD demand matrix and the travel latency function.

The remainder of this paper is organized as follows. In Sec. II we introduce definitions and preliminaries. In Sec. III we introduce the user-centric multi-class TAP and Wardrop Equilibrium conditions together with its inverse formulation for recovering the travel latency function. In Sec. IV we formulate the joint problem written as a bilevel program, as well as its transformation to QCQP. We then introduce the trust-region feasible direction method. Finally, we validate the approach through experiments and consider case studies in Sec. V. Limitations and conclusions are in Sections VI and VII, respectively.

Notation: All vectors are column vectors and denoted by bold lowercase letters. Bold uppercase letters denote matrices. We use "prime" to denote the transpose of a matrix or vector. We denote by $\mathbf{0}$ and $\mathbf{I}$ the vector of all zeroes and the identity matrix, respectively. Unless otherwise specified, $\|\cdot\|$ denotes the ℓ_2 norm. $|\mathcal{D}|$ denotes the cardinality of a set $\mathcal{D}$, and $[\![\mathcal{D}]\!]$ the set $\{1,\dots,|\mathcal{D}|\}$.

II. Preliminaries

A. Multi-class Transportation Network Model

Let $\tilde{\mathcal{U}}$ be the set of *user* (or *vehicle*) classes and, without loss of generality, assume that all vehicle classes travel in the same transportation network. Let the *original* network be denoted with a directed graph $\tilde{\mathcal{G}} = \big(\tilde{\mathcal{V}}, \tilde{\mathcal{A}}, \tilde{\mathcal{W}}\big)$ where $\tilde{\mathcal{V}}$ is the set of nodes, $\tilde{\mathcal{A}}$ is the set of arcs, and $\tilde{\mathcal{W}}$ is the set of K OD pairs defined as $\tilde{\mathcal{W}} = \big\{\mathbf{w}_k : \mathbf{w}_k := (w_{sk}, w_{tk}),\ k = 1,\dots,K\big\}$.

In order to include multiple vehicle classes, we use the idea in [31] of making $|\tilde{\mathcal{U}}|$ copies of $\tilde{\mathcal{G}}$, each corresponding to a vehicle class. We use these graphs to create an *enlarged* network, denoted with $\mathcal{G} = (\mathcal{V}, \mathcal{A}, \mathcal{W})$, where the sets of arcs,

nodes and OD pairs are constructed as the collection of all $|\tilde{\mathcal{U}}|$ graphs. Formally:

$$\mathcal{V} = \big\{v(i,u) : i \in [\![\tilde{\mathcal{V}}]\!], u \in [\![\tilde{\mathcal{U}}]\!]\big\}, \tag{1a}$$

$$\mathcal{A} = \big\{a(i,u) : i \in [\![\tilde{\mathcal{A}}]\!], u \in [\![\tilde{\mathcal{U}}]\!]\big\}, \tag{1b}$$

$$\mathcal{W} = \big\{\mathbf{w}(k,u) : \mathbf{w}(k,u) := (w_s(k,u), w_t(k,u)), \; k \in [\![\tilde{\mathcal{W}}]\!], u \in [\![\tilde{\mathcal{U}}]\!]\big\}. \tag{1c}$$

Let the node-link incidence matrix of $\mathcal{G}$ be $\mathbf{N} \in \{0,1,-1\}^{|\mathcal{V}|\times|\mathcal{A}|}$ and let $\mathbf{e}_{iu}$ denote the $|\mathcal{A}|$-dimensional vector where all entries are zero except the entry corresponding to link $a(i,u)$ which is set to one. To account for users' demand, for every OD $k \in [\![\tilde{\mathcal{W}}]\!]$ and class $u \in [\![\tilde{\mathcal{U}}]\!]$ let $d^{\mathbf{w}(k,u)}$ be its flow demand rate from node $w_s(k,u)$ to node $w_t(k,u)$. Moreover, let the vector $\mathbf{d}^{\mathbf{w}(k,u)} \in \mathbb{R}^{|\mathcal{V}|}$ be composed of all zeros except for the coordinates of nodes $w_s(k,u)$ and $w_t(k,u)$ which take values $-d^{\mathbf{w}(k,u)}$ and $d^{\mathbf{w}(k,u)}$, respectively. Let $\mathcal{F}$ be the set of feasible flow vectors defined as

$$\mathcal{F} = \Big\{\mathbf{x} \in \mathbb{R}^{|\mathcal{A}|}_{\geq 0} : \mathbf{x} = \sum_{\mathbf{w}\in\mathcal{W}} \mathbf{x}^{\mathbf{w}}, \, \mathbf{N}\mathbf{x}^{\mathbf{w}} = \mathbf{d}^{\mathbf{w}}, \, \forall \mathbf{w} \in \mathcal{W}\Big\},$$

where $\mathbf{x} = \big(x_{iu};\, i \in [\![\tilde{\mathcal{A}}]\!], u \in [\![\tilde{\mathcal{U}}]\!]\big)$, and x_{iu} denotes the flow of class u on link $a(i,u)$. Let $\mathbf{x}_u = \big(x_{iu};\, i \in [\![\tilde{\mathcal{A}}]\!]\big)$ be the flow vector for class u and $\mathbf{x}_{a_i} = \big(x_{iu};\, u \in [\![\tilde{\mathcal{U}}]\!]\big)$ the flow vector of all classes corresponding to the ith *physical* arc.

It has been shown that the single-class formulation is a special case of the multi-class model, and more interestingly, that we can treat the multi-class model as an enlarged single-class model [32]. Thus, we only need to consider general multi-class models. However, due to coupling of flows in the latency function from different classes of vehicles, some of the properties of the single-class might differ to the multi-class case. These are further discussed in the following subsection and in Remark 1. For convenience, we vectorize the demand for vehicle class u with $\mathbf{g}_u = \big(d^{\mathbf{w}(k,u)};\, \in [\![\tilde{\mathcal{W}}]\!]\big)$ and denote with $\mathbf{g} = \big(\mathbf{g}_u; u \in [\![\tilde{\mathcal{U}}]\!]\big)$ the full demand vector.

B. BPR-type Cost Functions

We consider the vector of travel latency functions to be defined by:

$$\mathbf{t}(\mathbf{x}) = \big(t_{iu}(\mathbf{x}_{a_i});\, i \in [\![\tilde{\mathcal{A}}]\!], u \in [\![\tilde{\mathcal{U}}]\!]\big), \tag{2}$$

where the cost on a *physical* link does not depend on the flows elsewhere, but on the flows present in the link from all classes. To simplify the analysis, we will restrict our travel latency functions to have the form of a *generalized* Bureau of Public Roads (BPR) form. This is, for each link $i \in [\![\tilde{\mathcal{A}}]\!]$ and user type $u \in [\![\tilde{\mathcal{U}}]\!]$ we have

$$t_{iu}(\mathbf{x}) = t^0_{iu} f(\boldsymbol{\theta}'\mathbf{x}_{a_i}/m_i) \tag{3}$$

where t^0_{iu} is the *free-flow travel time* for vehicle class u on link i; $f(\cdot)$ is a travel latency function which satisfies $f(0) = 1$, is strictly increasing and is continuously differentiable on $\mathbb{R}_+$; m_i is the *flow capacity* of link i, and $\boldsymbol{\theta} = \big(\theta_u;\, u \in [\![\tilde{\mathcal{U}}]\!]\big)$ is a weight vector such that $\theta_u \geq 1$, $\forall u \in [\![\tilde{\mathcal{U}}]\!]$. As a special case, the single-class network corresponds to $|\tilde{\mathcal{U}}| = 1$ and $\theta_1 = 1$. Finally, we make the following standard assumption in the context of the TAP for our analysis of the models.

Assumption A. *(i) $\mathcal{G}$ is strongly connected (there is at least one route connecting any origin with its destination). (ii) Demands $\mathbf{d}^{\mathbf{w}}$ are non-negative. (iii) Travel time functions are positive and continuous for every link.*

III. Models

In this section we present the models employed in this work to later construct the joint problem. We review the variational inequality of the TAP, as well as, an *inverse* TAP model. We begin by defining the notion of a Wardrop Equilibrium.

Definition 1. *A feasible flow $\mathbf{x}^* \in \mathcal{F}$ is a* Wardrop Equilibrium *if for every OD pair $\mathbf{w} \in \mathcal{W}$, and any route r_{iu} connecting $(w_s(i,u), w_t(i,u))$ with positive flow $h_{r_{iu}}$, the cost of traveling along that route is no greater than the cost of traveling along any other route.*

For an OD pair $\mathbf{w}(k,u)$ let r_{ku} be a path connecting its origin to its destination and $\mathcal{R}^{\mathbf{w}(k,u)}$ the set of all paths. Furthermore, let $h_{r_{ku}}$ be the flow assigned to path r_{ku}. Then a Wardrop Equilibrium exists if

$$h_{r_{ku}} > 0 \implies c_{r_{ku}} = \pi_{\mathbf{w}(k,u)}, \quad \forall r_{ku} \in \mathcal{R}^{\mathbf{w}(k,u)} \tag{4a}$$

$$h_{r_{ku}} = 0 \implies c_{r_{ku}} \geq \pi_{\mathbf{w}(k,u)}, \quad \forall r_{ku} \in \mathcal{R}^{\mathbf{w}(k,u)} \tag{4b}$$

where $c_{r_{ku}}$ is the travel time in route r_{ku} and $\pi_{\mathbf{w}(k,u)}$ is the travel time on the fastest route of $\mathbf{w}(k,u)$.

A. Variational Inequality

One way of modeling and solving, the TAP is via a *Variational Inequality* (VI) formulation [33]. In this context, let $\mathbf{h}$ be a feasible route flow vector. Utilizing Assumption A and the conditions described in (4), $\mathbf{h}^*$ is a Wardrop Equilibrium flow vector if and only if

$$\mathbf{c}(\mathbf{h}^*)'(\mathbf{h} - \mathbf{h}^*) \geq 0, \quad \forall \mathbf{h} \in \mathcal{H}, \tag{5}$$

where $\mathcal{H}$ is a convex feasible set for the route-based problem. If we also assume that the route costs are additive for a set of links, we can rewrite these conditions in the link-based form:

$$\mathbf{t}(\mathbf{x}^*)'(\mathbf{x} - \mathbf{x}^*) \geq 0, \quad \forall \mathbf{x} \in \mathcal{F}. \tag{6}$$

The existence and uniqueness of the solution to (6) for the single- and multi-class networks is shown in Theorems 2.4 and 2.5 of [32] as long as $\mathbf{t}(\cdot)$ is strongly monotone over $\mathcal{F}$, $\mathbf{t}(\cdot)$ is continuously differentiable, and $\mathcal{F}$ contains an interior point (Slater's condition).

Unfortunately, as stated in [34], it is not easy to verify that $\mathbf{t}(\cdot)$ is strongly monotone over $\mathcal{F}$ for general multi-class transportation networks. This happens because the cost $\mathbf{t}(\cdot)$ of a particular link depends on more than one vehicle class. This, in turn, creates an asymmetric Jacobian matrix of $\mathbf{t}(\cdot)$ for which first-order methods may not be sufficient to find the solution to (6). We therefore cannot always guarantee obtaining *unique* link flows for each vehicle class. However, we still hope to accurately estimate the cost functions by using a weighted sum of link flows (cf. (3)) for different user

types. Still, for both single- and multi-class cases there are many standard algorithms to solve (6) such as the Method of Successive Averages (MSA) [32] and the Frank-Wolfe algorithm [35].

B. User-Centric Inverse Model

The objective of the inverse model is to estimate the latency cost function $\mathbf{t}(\cdot)$ (specifically $f(\cdot)$ in (3)) using data. The key idea is to use observable equilibrium flow data and *known* OD demands to estimate $f(\cdot)$ by formulating an inverse optimization model [2].

To do so, we assume that we are given $|\mathcal{K}|$ samples of the link flow vector $\mathbf{x}$. One can think of these as flow samples which are produced by the same $\mathbf{t}(\cdot)$. The inverse formulation seeks to learn $f(\cdot)$ so that the flow observations are as close as possible to an equilibrium. Given that this formulation relies on measured data, we expect some measurement noise. Hence, the notion of an approximate solution to (6) is needed. Therefore, for a given $\varepsilon > 0$, we define:

Definition 2 (ε-approximate VI)**.** *Given $\varepsilon > 0$, $\hat{\mathbf{x}} \in \mathcal{F}$ is called an ε-approximate solution to* (6) *if*

$$\mathbf{t}(\hat{\mathbf{x}})'(\mathbf{x} - \hat{\mathbf{x}}) \geq -\varepsilon, \quad \forall \mathbf{x} \in \mathcal{F}. \tag{7}$$

Assume now we are given $|\mathcal{K}|$ link flow observations. For each $\kappa \in [\![\mathcal{K}]\!]$ we have $\mathbf{x}^{(\kappa)} = \big(x_{iu}^{(\kappa)};\, u \in [\![\tilde{\mathcal{U}}]\!]\big);\, i \in [\![\tilde{\mathcal{A}}]\!]\big)$. The inverse problem is then finding a function $\mathbf{t}(\cdot)$ such that $\mathbf{x}^{(k)}$ is an ε_κ-approximate solution of (6). Letting $\boldsymbol{\varepsilon} := (\varepsilon_\kappa; \kappa \in [\![\mathcal{K}]\!])$, we formulate the inverse VI problem as

$$\min_{\mathbf{t},\boldsymbol{\varepsilon}} \ \|\boldsymbol{\varepsilon}\| \tag{8a}$$

$$\text{s.t. } \mathbf{t}(\mathbf{x}^{(\kappa)})'(\mathbf{x} - \mathbf{x}^{(\kappa)}) \geq -\varepsilon_\kappa, \quad \forall \mathbf{x} \in \mathcal{F}^{(\kappa)}, \kappa \in [\![\mathcal{K}]\!], \tag{8b}$$

$$\varepsilon_\kappa > 0, \quad \forall \kappa \in [\![\mathcal{K}]\!], \tag{8c}$$

where the optimization is over $\boldsymbol{\varepsilon}$, and the selection of function $\mathbf{t}(\cdot)$. Note that (8) is not solvable yet as we have not specified $\mathbf{t}(\cdot)$. Also observe that the set of constraints restricts the travel time function to be within ε_k units of the Wardrop equilibrium flows for each sample. In this sense, if we solve the problem using a large $[\![\mathcal{K}]\!]$ corresponding to multiple observed networks, we will find a more "stable" travel latency function as we expect the variance of the estimated parameters to decrease.

To solve (8), we require to give more structure to $\mathbf{t}(\cdot)$, or more specifically $f(\cdot)$. Aiming to recover a function with good data reconciling and generalization properties, we apply an approach which expresses the function $f(\cdot)$ in a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$ as in [2]. Hence, we introduce the set of dual variables $\mathbf{y} = \big(\mathbf{y}^{\mathbf{w}} \in \mathbb{R}^{|\mathcal{V}|};\, \mathbf{w} \in \mathcal{W}^{(\kappa)}, \kappa \in [\![\mathcal{K}]\!]\big)$ which can be interpreted as the "cost" of transporting $\mathbf{d}^{\mathbf{w}}$ units of flow trough the network. Then, the inverse problem is to

$$\min_{f,\boldsymbol{\varepsilon},\mathbf{y}} \ \|\boldsymbol{\varepsilon}\| + \gamma \|f\|_{\mathcal{H}}^2 \tag{9a}$$

$$\text{s.t. } \mathbf{e}_{iu}'\mathbf{N}_\kappa'\mathbf{y}^{\mathbf{w}} \leq t_{iu}^0 f(\boldsymbol{\theta}'\mathbf{x}_{a_i}^{(\kappa)}/m_i), \tag{9b}$$

$$\forall i \in [\![\tilde{\mathcal{A}}]\!],\ u \in [\![\tilde{\mathcal{U}}]\!],\ \mathbf{w} \in \mathcal{W}^{(\kappa)},\ \kappa \in [\![\mathcal{K}]\!],$$

$$\sum_{i=1}^{|\tilde{\mathcal{A}}|} \Big(\sum_{u=1}^{|\tilde{\mathcal{U}}|} t_{iu}^0 x_{a_i u}^{(\kappa)} \ \ f(\boldsymbol{\theta}'\mathbf{x}_{a_i}^{(\kappa)}/m_i) \Big) - \sum_{\mathbf{w} \in \mathcal{W}_\kappa} (\mathbf{d}^{\mathbf{w}})'\mathbf{y}^{\mathbf{w}} \leq \varepsilon_\kappa, \quad \forall \kappa \in [\![\mathcal{K}]\!], \tag{9c}$$

$$f(\boldsymbol{\theta}'\mathbf{x}_{a_i}^{(\kappa)}/m_i) < f(\boldsymbol{\theta}'\mathbf{x}_{a_{\tilde{i}}}^{(\kappa)}/m_{\tilde{i}}), \tag{9d}$$

$$\forall i, \tilde{i} \in [\![\tilde{\mathcal{A}}]\!] \ \text{ s.t.} \frac{\boldsymbol{\theta}'\mathbf{x}_{a_i}^{(\kappa)}}{m_i} < \frac{\boldsymbol{\theta}'\mathbf{x}_{a_{\tilde{i}}}^{(\kappa)}}{m_{\tilde{i}}}; \forall \kappa \in [\![\mathcal{K}]\!],$$

$$f(0) = 1, \tag{9e}$$

$$\boldsymbol{\varepsilon} \geq \mathbf{0}, \quad f \in \mathcal{H},$$

where $\mathbf{N}_\kappa$ is the node-link incidence matrix of the κ-th network and where we have replaced constraints (8b) with (9b)-(9e). In the objective, γ is a regularization parameter and $\|f\|_{\mathcal{H}}^2$ denotes the squared norm of $f(\cdot)$ in $\mathcal{H}$ (note that to solve (9), $f(\cdot)$ still need to be specified). The first constraint (9b) corresponds to dual feasibility, (9c) states that the solution to the κ-th network is within ε_κ distance, in other words, it is the suboptimality constraint (primal-dual gap). (9d) enforces $f(\cdot)$ to be strictly increasing, and (9e) is for normalization purposes (see (3)). Note that a larger γ implies giving more weight to estimating a "better" $f(\cdot)$ that generalizes better out-of-sample, rather than fitting the data better. In contrast, a smaller γ would recover a "tighter" $f(\cdot)$ in terms of data reconciliation but might not provide good generalization.

One question that may arise is: How does $\|\boldsymbol{\varepsilon}\|$ behaves as we increase the number of data samples? To answer this question, we refer to Theorems 6 and 7 of [2, Sec. 6] where the convergence of (9) is discussed in detail.

So far, solving (9) is ambiguous since it optimizes over undefined functions $f(\cdot)$. To make the estimation problem solvable, we specify $\mathcal{H}$ (and thus the class of $f(\cdot)$) by choosing its reproducing kernel [36] as a polynomial $\phi(x, y) = (c + xy)^n$ for some choice of $c \in \mathbb{R}_{\geq 0}$ and $n \in \mathbb{N}$. We believe this is a good choice since it matches our intuition on how congestion affects the latency cost of links (cf. (3)). The polynomial kernel function is

$$\phi(x, y) = (c + xy)^n = \sum_{i=0}^{n} \binom{n}{i} c^{n-1} x^i y^i.$$

Using the representer theorem for kernel functions, we modify the travel latency function of (9) to a quadratic function parameterized by $\boldsymbol{\beta} = \{\beta_j : j = [\![n]\!]\}$ where n is the degree of the polynomial. This renders to the following tractable Quadratic Programming (QP) problem (see (3.2), (3.2), and (3.6) in [36] for details)

$$\min_{\boldsymbol{\beta},\mathbf{y},\boldsymbol{\varepsilon}} \ \|\boldsymbol{\varepsilon}\| + \gamma \sum_{j=0}^{n} \frac{\beta_j^2}{\binom{n}{j} c^{n-j}} \tag{10}$$

$$\text{s.t.} \quad (9\text{b}) - (9\text{d}), \quad \beta_0 = 1.$$

The output of (10) contains $\boldsymbol{\beta}^*$, and therefore the $f(\cdot)$ estimator is

$$\hat{f}(x) := \sum_{i=0}^{n} \beta_i^* x^i = 1 + \sum_{i=1}^{n} \beta_i^* x^i,$$

where we set $\beta_0 = 1$ to have $f(0) = 1$. Note that the well-recognized vanilla BPR function (i.e., $f(x) = 1 + 0.15x^4$) is a special case of the proposed polynomial $\hat{f}(x)$ when $\boldsymbol{\beta} = [1, 0, 0, 0, 0.15]$.

Remark 1. In the above QP formulation, we have assumed that the parameter vector $\boldsymbol{\theta}$ and the set of user classes $\tilde{\mathcal{U}}$ are the same for all $|\mathcal{K}|$ networks. Since (3) is a weighed sum of link flows of different classes of vehicles. Hence, as noted previously, we aim to accurately recover the travel latency function from such weighted sum of link flows. In Sec. V we will illustrate this by conducting a numerical experiment.

To facilitate the analysis, let us write (10) using the following compact notation:

$$\min_{\boldsymbol{\beta},\mathbf{y},\boldsymbol{\varepsilon}} \quad \boldsymbol{\varepsilon}'\mathbf{I}\boldsymbol{\varepsilon} + \boldsymbol{\beta}'\mathbf{H}\boldsymbol{\beta} \tag{11a}$$

$$\text{s.t.} \quad \mathbf{A}(\mathbf{g})\mathbf{y} + \mathbf{B}\boldsymbol{\beta} + \mathbf{C}\boldsymbol{\varepsilon} + \mathbf{h} \leq \mathbf{0}, \tag{11b}$$

where $\mathbf{H}$ is a positive definite matrix and $\mathbf{A}(\mathbf{g})$ depends on the demand vector $\mathbf{g}$ through constraint (9c). Therefore, we have properly defined an optimization problem to estimate $f(\cdot)$ via $\boldsymbol{\beta}$. This formulation was proposed for single-class networks in [2] and employed in [37] to estimate the *Price of Anarchy* (ratio between the solutions of the user-centric and system-optimal TAP). We have extended this model for the multi-class case and in the next section we present our main contribution, the joint estimation of $f(\cdot)$ and $\mathbf{g}$ for the multi-class TAP.

IV. The Joint Problem

A. Bilevel Formulation

Unlike most previous work in this field, we would like to recover the *travel time* function $f(\cdot)$ (i.e., $\boldsymbol{\beta} = (\beta_0, \ldots, \beta_n)$) and the OD demand vector $\mathbf{g}$ jointly. To achieve this, we define the joint problem as a bilevel formulation and we propose an algorithm to find its solution. To ease notation, for any $\boldsymbol{\beta}$ and $\mathbf{g}$ we let $\mathbf{x}(\boldsymbol{\beta}, \mathbf{g}) = (\mathbf{x}_{a_i}(\boldsymbol{\beta}, \mathbf{g});\ \forall i \in [\![\tilde{\mathcal{A}}]\!])$ be the optimal solution to (6) (i.e. the TAP).

Assume that we observe $|\mathcal{K}|$ different equilibrium link flow vectors $\mathbf{x}^{(\kappa)}$, and let $\mathbf{x}^d := (\mathbf{x}^{(\kappa)}, \forall \kappa \in [\![\mathcal{K}]\!])$ be the vectorized version of all the observed data. Using these definitions we write the bilevel optimization problem as

$$\min_{\boldsymbol{\beta},\mathbf{g}} \quad \hat{F}(\boldsymbol{\beta}, \mathbf{g}) := \sum_{\kappa=1}^{|\mathcal{K}|} \sum_{u=1}^{|\tilde{\mathcal{U}}|} \sum_{i=1}^{|\tilde{\mathcal{A}}|} ([\mathbf{x}(\boldsymbol{\beta}, \mathbf{g})]_{iu} - x_{iu}^{(\kappa)})^2 \tag{12a}$$

$$\text{s.t. } [\mathbf{x}(\boldsymbol{\beta}, \mathbf{g})]_u = \text{TAP}(\boldsymbol{\beta}, \mathbf{g}), \quad \forall u \in [\![\tilde{U}]\!]$$

$$(\boldsymbol{\beta}, \mathbf{y}, \boldsymbol{\varepsilon}) = \arg\min_{\boldsymbol{\beta},\mathbf{y},\boldsymbol{\varepsilon}} \big\{ \boldsymbol{\varepsilon}'\mathbf{I}\boldsymbol{\varepsilon} + \boldsymbol{\beta}'\mathbf{H}\boldsymbol{\beta}, \tag{12b}$$

$$\text{s.t. } \mathbf{A}(\mathbf{g})\mathbf{y} + \mathbf{B}(\mathbf{x}^d)\boldsymbol{\beta} + \mathbf{C}\boldsymbol{\varepsilon} + \mathbf{h} \leq \mathbf{0} \big\}.$$

Remark 2. Typically, most related work aiming to solve this problem include a term in (12a) which penalize the deviation of the estimated demand $\mathbf{g}$ from a *known* demand vector $\mathbf{g}^0$. To include this penalty one requires *prior knowledge* (or a good estimate) of the true demand through $\mathbf{g}^0$. We believe this term has been historically included due to the fact that the optimization problem is non-convex in terms of $\mathbf{x}$. Hence, this prior knowledge aims to steer the solution towards a local optimum close to the initial demand estimate $\mathbf{g}^0$. In contrast, we avoid imposing such connection with an initial demand estimate. Still, our methodology allows, if desired, to include such term in the objective function.

B. Optimality conditions of (12b)

To write (12) as a computationally solvable joint optimization problem, we replace the lower-level problem (12b) by its optimality conditions and write (12) as a single-level problem. Note that both (12) and (11) have convex quadratic objectives as the only non-zero entries in their objective are in the diagonal of the $\mathbf{Q}$ matrix in their canonical QP standard form. We begin our analysis by writing the Lagrangian function:

$$\mathcal{L}(\boldsymbol{\beta}, \mathbf{y}, \boldsymbol{\varepsilon}; \boldsymbol{\nu}) = \boldsymbol{\varepsilon}'\mathbf{I}\boldsymbol{\varepsilon} + \boldsymbol{\beta}'\mathbf{H}\boldsymbol{\beta} + \boldsymbol{\nu}'(\mathbf{A}\mathbf{y} + \mathbf{B}\boldsymbol{\beta} + \mathbf{C}\boldsymbol{\varepsilon} + \mathbf{h}), \tag{13}$$

where $\boldsymbol{\nu}$ denotes the dual variables corresponding to constraints (11b). To simplify the notation, we dropped the dependence of $\mathbf{A}$ and $\mathbf{B}$ on $\mathbf{g}$ and $\mathbf{x}^d$, respectively. Furthermore, the first-order conditions of (11) are

$$\partial\mathcal{L}/\partial\boldsymbol{\varepsilon} = 2\mathbf{I}\boldsymbol{\varepsilon} + \mathbf{C}'\boldsymbol{\nu} = \mathbf{0} \Rightarrow \boldsymbol{\varepsilon} = -(1/2)\mathbf{I}^{-1}\mathbf{C}'\boldsymbol{\nu}, \tag{14a}$$

$$\partial\mathcal{L}/\partial\boldsymbol{\beta} = 2\mathbf{H}\boldsymbol{\beta} + \mathbf{B}'\boldsymbol{\nu} = \mathbf{0} \Rightarrow \boldsymbol{\beta} = -(1/2)\mathbf{H}^{-1}\mathbf{B}'\boldsymbol{\nu}, \tag{14b}$$

$$\partial\mathcal{L}/\partial\mathbf{y} = \mathbf{A}'\boldsymbol{\nu} = \mathbf{0}, \tag{14c}$$

which, by substituting $\boldsymbol{\varepsilon}$ and $\boldsymbol{\beta}$ in (13) using (14), yields to the dual function

$$D(\boldsymbol{\nu}) = -(1/4)\boldsymbol{\nu}'(\mathbf{C}\mathbf{I}\mathbf{C}' + \mathbf{B}\mathbf{H}^{-1}\mathbf{B})\boldsymbol{\nu} + \mathbf{h}'\boldsymbol{\nu}.$$

It follows that for each primal-dual pair $(\boldsymbol{\beta}, \mathbf{y}, \boldsymbol{\varepsilon};\ \boldsymbol{\nu})$ in (12b), the necessary and sufficient conditions are

$$\mathbf{A}\mathbf{y} + \mathbf{B}\boldsymbol{\beta} + \mathbf{C}\boldsymbol{\varepsilon} + \mathbf{h} \leq \mathbf{0}, \tag{15a}$$

$$\mathbf{A}'\boldsymbol{\nu} = \mathbf{0}, \tag{15b}$$

$$\boldsymbol{\nu} \geq \mathbf{0}, \tag{15c}$$

$$\boldsymbol{\varepsilon}'\mathbf{I}\boldsymbol{\varepsilon} + \boldsymbol{\beta}'\mathbf{H}\boldsymbol{\beta} = -\frac{1}{4}(\boldsymbol{\nu}'\mathbf{C}\mathbf{I}\mathbf{C}' + \boldsymbol{\nu}'\mathbf{B}\mathbf{H}^{-1}\mathbf{B}' + \mathbf{h}')\boldsymbol{\nu}, \tag{15d}$$

where (15a)-(15c) correspond to primal and dual feasibility and (15d) equates the primal and dual objectives to ensure strong duality.

C. Relaxation and Trust-Region Feasible-Direction Method

At this point we are ready to convert the bilevel problem to a single-level joint problem. We do this by replacing the lower-level problem (12b) by its KKT conditions (15). In other words, we minimize (12a) subject to (15). However, note that (15d), corresponding to strong duality, is a non-convex quadratic equality constraint. To address this issue, we relax this constraint by requiring that the gap is upper bounded by some $\xi \in \mathbb{R}_{\geq 0}$. We penalize this gap in the objective using some λ. Then, the new joint formulation becomes

$$\min_{\boldsymbol{\beta},\mathbf{g},\mathbf{y},\boldsymbol{\nu},\boldsymbol{\varepsilon},\xi} \quad F(\boldsymbol{\beta}, \mathbf{g}, \xi) := \hat{F}(\boldsymbol{\beta}, \mathbf{g}) + \lambda\xi \tag{16a}$$

$$\text{s.t. (15a), (15b),}$$

$$\boldsymbol{\varepsilon}'\mathbf{I}\boldsymbol{\varepsilon} + \boldsymbol{\beta}'\mathbf{H}\boldsymbol{\beta} + \frac{1}{4}\boldsymbol{\nu}'(\mathbf{C}\mathbf{I}\mathbf{C}' + \mathbf{B}\mathbf{H}^{-1}\mathbf{B}')\boldsymbol{\nu} - \mathbf{h}'\boldsymbol{\nu} \leq \xi, \tag{16b}$$

$$\boldsymbol{\nu}, \boldsymbol{g}, \boldsymbol{\beta}, \xi \geq \mathbf{0}, \tag{16c}$$

where $\boldsymbol{\beta}$ and $\mathbf{g}$ are the quantities of interest; $\mathbf{y}$ and $\boldsymbol{\nu}$ are the dual variables defined in (9) and (13), respectively; and $\boldsymbol{\varepsilon}$ and ξ are the primal-dual gap relaxations.

Unfortunately, both the objective and the constraints, through $\mathbf{A}$, are nonlinear functions of $\mathbf{g}$, $\mathbf{y}$ and $\boldsymbol{\nu}$. Therefore, to solve (16), we propose an iterative *trust-region feasible direction* method. To do so, let $\mathbf{z} = (\boldsymbol{\beta}, \mathbf{g}, \xi)$ and denote with j the iteration count. With this sequential approach, we evaluate the gradient of $F(\cdot)$ at the previous iteration and seek the steepest feasible direction of descent by solving the following optimization problem:

$$\min_{\mathbf{z}^j, \mathbf{y}, \boldsymbol{\nu}, \boldsymbol{\varepsilon}} \nabla F(\mathbf{z}^{j-1})'(\mathbf{z}^{j-1} - \mathbf{z}^j) \tag{17a}$$

$$\text{s.t. (15a), (15b), (16b)}$$

$$\mathbf{g}^{j-1} - c_{1j}\mathbf{e} \;\leq\; \mathbf{g}^j \;\leq\; \mathbf{g}^{j-1} + c_{2j}\mathbf{e}, \tag{17b}$$

$$\boldsymbol{\beta}^{j-1} - d_{1j}\mathbf{e} \;\leq\; \boldsymbol{\beta}^j \;\leq\; \boldsymbol{\beta}^{j-1} + d_{2j}\mathbf{e}, \tag{17c}$$

$$\boldsymbol{\nu}, \mathbf{z}^j \geq \mathbf{0}, \tag{17d}$$

where $\mathbf{e}$ denote the vector of all ones and c_{1j}, c_{2j}, d_{1j}, d_{2j} are used as step-size parameters. The matrix $\mathbf{A}$ in the constraint (15a), and (15b) is a function of $\mathbf{g}$ and we approximate it by using $\mathbf{g}^{j-1}$. Then, the gradient in (17a) is expressed by

$$\nabla F(\mathbf{z}^j)' = \Bigg[\sum_{u=1}^{|\tilde{\mathcal{U}}|} \sum_{i=1}^{|\tilde{\mathcal{A}}|} 2(x_{iu}(\mathbf{z}^j) - x^*_{iu}) \frac{\partial x_{iu}(\boldsymbol{\beta}^j, \mathbf{g}^j)}{\partial \beta_l},\ l = [\![n]\!];$$
$$\sum_{i=1}^{|\tilde{\mathcal{A}}|} 2(x_{iu}(\mathbf{z}^j) - x^*_{iu}) \frac{\partial x_{iu}(\boldsymbol{\beta}^j, \mathbf{g}^j)}{\partial g_{ku}},\ k = [\![\tilde{\mathcal{W}}]\!], u = [\![\tilde{\mathcal{U}}]\!]; \lambda \Bigg]. \tag{18}$$

As a result, problem (17) has a linear objective (provided that we can evaluate the partial derivatives) and constraints that are linear and convex quadratic, making it a tractable problem. In fact, this problem can be written as a quadratic program by replacing ξ^j in the objective (17a) by the left-hand side of constraint (16b). Given these "constant" approximations of the constraints at the prior iterate, the role of c_{1j}, c_{2j}, d_{1j}, d_{2j} is to ensure that the optimization takes place in a relatively small "trust" region for $(\boldsymbol{\beta}^j, \mathbf{g}^j)$ that is not too far from the prior iterate $(\boldsymbol{\beta}^{j-1}, \mathbf{g}^{j-1})$. Note, however, that to solve (17), we still need to estimate the partial derivatives of the flows with respect to $\mathbf{g}$ and $\boldsymbol{\beta}$.

D. Derivatives

It has been observed that it is hard to derive analytical expressions for the partial derivatives of $\mathbf{x}(\boldsymbol{\beta}, \mathbf{g})$ with respect to $\boldsymbol{\beta}$ and $\mathbf{g}$. To overcome this, we use classical approximation techniques. We refer the interested reader to a comprehensive discussion carried out in [38].

1) Directional flow derivatives w.r.t OD demand: We derive an approximation to the gradient of $\mathbf{x}(\boldsymbol{\beta}, \mathbf{g})$ with respect to $\mathbf{g}$. To that end, fix $i \in [\![\tilde{\mathcal{A}}]\!], u \in [\![\tilde{\mathcal{U}}]\!]$ and add the flows over the OD pairs demands

$$\begin{aligned} x_{iu}(\boldsymbol{\beta}, \mathbf{g}) &= \sum_{k \in [\![\tilde{\mathcal{W}}]\!]} \sum_{r \in \mathcal{R}_u^k} \delta_r^{a(i,u)} p_u^{kr} g_{ku} \\ &= \sum_{k \in [\![\tilde{\mathcal{W}}]\!]} g_{ku} \sum_{r \in \mathcal{R}_u^k} \delta_r^{a(i,u)} p_u^{kr}, \end{aligned} \tag{19}$$

where $\mathcal{R}_u^k$ denotes the set of feasible routes for vehicle class u in OD pair k; p_u^{wr} stands for the percentage of class u vehicles using route $r \in \mathcal{R}_u^k$, and

$$\delta_r^{a(i,u)} := \begin{cases} 1, & \text{if route } r \text{ uses link } a(i,u), \\ 0, & \text{otherwise.} \end{cases} \tag{20}$$

As pointed out by [39], [13], for all $k \in [\![\tilde{\mathcal{W}}]\!]$ and all $u \in [\![\tilde{\mathcal{U}}]\!]$, and assuming that the route probabilities are locally constant, equation (19) implies

$$\frac{\partial x_{iu}(\boldsymbol{\beta}, \mathbf{g})}{\partial g_{ku}} = \begin{cases} \sum_{r \in \mathcal{R}_u^k} \delta_r^{a(i,u)} p_u^{kr}, \\ 0, \ \text{otherwise.} \end{cases} \tag{21}$$

If we only consider the shortest route $r_{ku}(\boldsymbol{\beta}, \mathbf{g})$ based on the travel latency cost, we have that

$$\frac{\partial x_{iu}(\boldsymbol{\beta}, \mathbf{g})}{\partial g_{ku}} \approx \delta^{a(i,u)}_{r_{ku}(\boldsymbol{\beta}, \mathbf{g})} = \begin{cases} 1, & \text{if } a(i,u) \in r_{ku}(\boldsymbol{\beta}, \mathbf{g}), \\ 0, & \text{otherwise,} \end{cases} \tag{22}$$

where $a(i,u) \in r_{ku}(\boldsymbol{\beta}, \mathbf{g})$ indicates that the shortest route $r_{ku}(\boldsymbol{\beta}, \mathbf{g})$ uses link $a(i,u)$. By (22) we obtain an approximation to the Jacobian matrix

$$\left[\frac{\partial x_{iu}(\boldsymbol{\beta}, \mathbf{g})}{\partial g_{ku}};\ i \in [\![\tilde{\mathcal{A}}]\!],\ u \in [\![\tilde{\mathcal{U}}]\!],\ k \in [\![\tilde{\mathcal{W}}]\!] \right]. \tag{23}$$

The reasons to consider only the shortest routes for the purpose of calculating these gradients are: (i) GPS routing services are widely-used by vehicle drivers so they tend to always select the fastest routes, (ii) considering only the fastest routes significantly simplifies the calculation of the route-choice probabilities, and (iii) extensive numerical experiments and algorithms show that such an approximation of the gradients performs well, e.g., TAPAS, or MSA.

2) Directional flow derivatives w.r.t coefficients of the travel latency function: To the best of our knowledge there are two main techniques to calculate directional derivatives of the link flows with respect to a perturbation ρ on the cost coefficients $\boldsymbol{\beta}$. In [26], [38], sensitivity analysis is performed with respect to the routes and require solving a linear system that it is hard to solve for large-scale networks. To overcome this issue, [40] develops a QP formulation to calculate such derivatives. However, this QP has a similar complexity as the TAP. Therefore, although we are able to use any of these methods to calculate $\partial x_{iu}(\boldsymbol{\beta}, \mathbf{g})/\partial \beta_l$, we prefer to use a finite-difference approximation. This is because, (i) the complexity of solving the TAP is similar to that of the QP proposed in [40]; (ii) there are fast algorithms such as MSA, Frank-Wolfe, TAPAS to solve the TAP efficiently; and (iii) the TAP allows to include all routes connecting any OD pair of any class rather than defining a route set as in [40]. Using $\text{TAP}_{iu}(\cdot)$ to denote the solution of the TAP for link i and class u, for some small enough ρ we compute

$$\frac{\partial x_{iu}(\boldsymbol{\beta}^j, \mathbf{g}^j)}{\partial \beta_l} \approx \frac{\text{TAP}_{iu}(\boldsymbol{\beta}^j + \rho \mathbf{e}_l, \mathbf{g}^j) - \text{TAP}_{iu}(\boldsymbol{\beta}^j, \mathbf{g}^j)}{\rho},$$

where $\mathbf{e}_l$ is the lth unit vector.

Using these two approximation to the partial derivatives we have developed a complete method for solving the joint problem. We summarize our approach in Algorithm 1.

Algorithm 1 Joint cost-demand estimation.

Input: $\mathcal{G}$; λ; ρ; $(\mathbf{g}^0, \boldsymbol{\beta}^0)$: initial OD demand and travel latency function; step-size rule; (η, T) : precision parameters.

Initialize: Set $j = 0$, calculate $F(\boldsymbol{\beta}^0, \mathbf{g}^0)$, **if:** $F(\boldsymbol{\beta}^0, \mathbf{g}^0) = 0$ **then:** terminate and output $(\boldsymbol{\beta}^0, \mathbf{g}^0)$ **else:**

1: Compute $\mathbf{x}_u^j = \text{TAP}_u(\mathbf{g}^j, \boldsymbol{\beta}^j)$ for every $u \in [\![\tilde{\mathcal{U}}]\!]$.
2: Obtain derivatives $\partial \mathbf{x}(\boldsymbol{\beta}^j, \mathbf{g}^j)/\partial \boldsymbol{\beta}^j$ and $\partial \mathbf{x}(\boldsymbol{\beta}^j, \mathbf{g}^j)/\partial \mathbf{g}^j$ using (23) and (24), respectively.
3: Approximate the descent direction $\nabla F(\boldsymbol{\beta}^j, \mathbf{g}^j)$ using (18).
4: Choose a trust region/step size.
5: Solve for a steepest descent direction using (17), and obtain $\boldsymbol{\beta}^{j+1}$ and $\mathbf{g}^{j+1}$.
6: (Termination criterion)
- **if** $\frac{\|F(\boldsymbol{\beta}^j, \mathbf{g}^j) - F(\boldsymbol{\beta}^{j+1}, \mathbf{g}^{j+1})\|_2}{F(\boldsymbol{\beta}^0, \mathbf{g}^0)} \geq \eta$ **and** $j < T$ **then** let $j = j + 1$ and go to Step 1,
- **else** Terminate and output: $(\boldsymbol{\beta}^{j+1}, \mathbf{g}^{j+1})$.

Algorithm 2 Alternating cost-demand estimation.

Input: $\mathcal{G}$; λ; ρ; $(\mathbf{g}^0, \boldsymbol{\beta}^0)$: initial OD demand and travel latency function; step-size rule; (η, T) : precision parameters.

Initialize: Set $j = 0$, calculate $F(\boldsymbol{\beta}^0, \mathbf{g}^0)$, **if:** $F(\boldsymbol{\beta}^0, \mathbf{g}^0) = 0$ **then:** terminate and output $(\boldsymbol{\beta}^0, \mathbf{g}^0)$ **else:**

1: Compute $\mathbf{x}_u^j = \text{TAP}_u(\mathbf{g}^j, \boldsymbol{\beta}^j)$ for every $u \in [\![\tilde{\mathcal{U}}]\!]$.
2: Obtain derivative $\partial \mathbf{x}(\boldsymbol{\beta}^j, \mathbf{g}^j)/\partial \boldsymbol{\beta}^j$ using (23).
3: Find best step-size by doing line-search: $\alpha = \arg\min_{c_1 \leq \alpha \leq c2} F(\boldsymbol{\beta}^j, \mathbf{g} - \alpha \nabla_{\mathbf{g}} F(\boldsymbol{\beta}^j, \mathbf{g}^j))$
4: Take a gradient-step: $\mathbf{g}^{j+1} = \mathbf{g}^j - \alpha \nabla_{\mathbf{g}} F(\boldsymbol{\beta}^j, \mathbf{g}^{j+1})$
5: Obtain $\boldsymbol{\beta}^{j+1}$ by solving (9) for a fixed $\mathbf{g}^{j+1}$ and constraint (17c).
6: If $F(\mathbf{g}^{j+1}, \boldsymbol{\beta}^{j+1}) > F(\mathbf{g}^{j+1}, \boldsymbol{\beta}^j)$ let $\boldsymbol{\beta}^{j+1} = \boldsymbol{\beta}^j$.
7: (Termination criterion): Same as in Alg. 1

Our algorithm proceeds as a trust-region feasible direction method that seeks to solve the joint cost-demand estimation problem (defined in (12)). The key idea of the algorithm is to jointly select, at every iteration j, the vectors $\boldsymbol{\beta}^j$ and $\mathbf{g}^j$ that maximize the reduction in the objective function of (12) by solving (17). To achieve this, we evaluate the gradient of the function at the previous iteration of $\boldsymbol{\beta}^j$ and $\mathbf{g}^j$ and restrict our new iterates to be within a trust region. By recursively performing this procedure, we expect (and observe empirically) the algorithm to converge to a local minimum.

Note that Alg. 1 is a method that solves a sequence of quadratic problems. This approach has some similarities to the one in [1] where, in general, convergence cannot be guaranteed. This is due to the dependence between $\boldsymbol{\beta}$ and $\mathbf{g}$. In addition, a convergence analysis would need to account for the fact that $\mathbf{A}(\mathbf{g}^j)$ is approximated by $\mathbf{A}(\mathbf{g}^{j-1})$, which is not straightforward. A potential approach to address this issue is by selecting a small step-size and by using the minimization rule. To that end, we need to perform a line search at every iteration (this is standard in alternating methods, see [41], [42]). However computing $\mathbf{x}(\boldsymbol{\beta}, \mathbf{g})$ is computationally demanding and line search would be too expensive.

Moreover, we point out that solving (17) recursively is computationally intensive as it involves solving a large optimization problem (e.g., the dimensions of $\boldsymbol{\nu}$ and $\mathbf{y}$ are $\mathcal{O}(|\mathcal{K}| \times |\mathcal{A}|)$ and $\mathcal{O}(|\mathcal{K}| \times (|\mathcal{W}| \times |\mathcal{V}|))$, respectively) with a linear objective and quadratic constraints. We leave the development of techniques to reduce the computational burden, such as the one described in [2], to future work.

Both the computational burden and the hardness of proving convergence motivate us to propose an *alternating method* described in Alg 2. This algorithm proceeds by adjusting $\mathbf{g}$ using gradient descent, and then estimating the *restricted* travel latency function using (9) with constraint (17c). The algorithm possesses two advantages over Alg. 1. First, it decomposes the problem making it more computationally tractable. Second, we can guarantee its convergence. This follows by observing that $F(\boldsymbol{\beta}, \mathbf{g})$ is lower-bounded by 0 and by observing that at every iteration the objective is either reduced or stays at its current value. To show this, we argue that if $c_1 < 0 < c_2$ then Step 3 certifies $F(\boldsymbol{\beta}^j, \mathbf{g}^{j+1}) \leq F(\boldsymbol{\beta}^j, \mathbf{g}^j)$ since we can always select $\alpha = 0$. Moreover, by imposing Step 6 we ensure a similar argument for $\boldsymbol{\beta}$. Thus, by using the monotone convergence theorem [43] we guarantee its convergence. Its main drawback is that the steepest directions of both $\mathbf{g}$ and $\boldsymbol{\beta}$ are done independently rather than jointly as in Alg 1 and it may not converge to a local minimum of the joint problem.

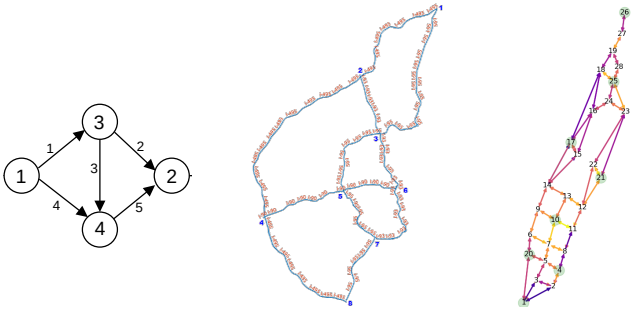
V. Model Validation and Case Studies

We report numerical experiments conducted on a benchmark network and subnetworks from the Eastern Massachusetts Area (EMA) and New York City (NYC). We begin by validating our methods on single-class and multi-class variants of the well-known Braess Network. Then, we perform additional validation experiments using real networks: first on the EMA network focusing on an interstate highway subnetwork and then on an urban network in NYC.

A. Model Validation

To perform these experiments, we generate "ground truth" data by selecting specific OD demands and cost functions $(\boldsymbol{\beta}^*, \mathbf{g}^*)$. Then, we solve the TAP problem using these "true" inputs to obtain $\mathbf{x}^*$, the "true" flow (note that, normally, we would obtain $\mathbf{x}^*$ from data). To test the performance, we take the generated flows, $\mathbf{x}^*$ as input to Alg. 1 and compare the resulting $(\boldsymbol{\beta}^j, \mathbf{g}^j)$ to the ground truth. In addition to reporting the performance of the methods developed herein, we compare them with simpler algorithms. In particular: (i) the approach of just estimating OD demands by assuming a fixed travel latency cost function ($f(\cdot) = \text{BPR}$), and (ii) a gradient descent method (GD) using the estimated derivatives in (23), (24).

1) Single-Class Braess Network: As our first experiment, we use the Braess network (Fig. 1a). We generate ground truth data by considering a single OD pair which transports $4,000$ vehicles per hour from node 1 to 2 and with a true travel latency function $f(x) = 1 + 0.45x^4$. The resulting "true" flows when solving the TAP for this network are: $(2095, 1904, 2095, 0, 1904, 1)$ for links $(1, 2, 3, 4, 5)$ respectively. Then, we initiate Alg. 1 with $\mathbf{g}_0 = \mathbf{0}$ and $\boldsymbol{\beta}_0 = (1, 0, 0, 0, 0.15, 0)$. We set $c = 30$, $\lambda = 0.1$, $\rho = 0.1$, $n = 5$, and adaptive step-sizes $c_{1j} = c_{2j} = \frac{200}{\sqrt{j}}$, $d_{1j} = d_{2j} = \frac{0.02}{j^{3/4}}$.



(a) Braess' (b) EMA (c) NYC

Fig. 1. Network topologies.

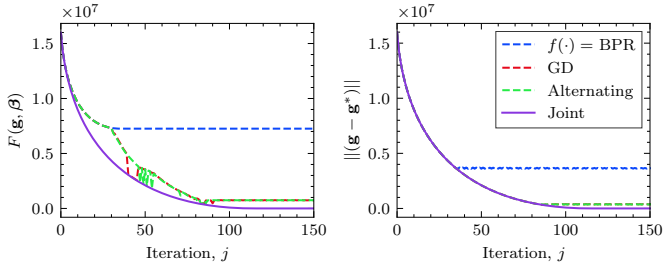


Fig. 2. Model validation on Braess Network. $f(\cdot)$ solves the problem using a static BPR function; *GD* uses a gradient decent; *Alternating* solves the two problems sequentially; *Joint* solves the problem using Alg. 1

In the left plot of Fig. 2 we observe the objective function of the joint problem (16a) converging to zero when solving the problem jointly. In contrast, when only adjusting OD Demands with a fixed $f(\cdot)$, the method converges to a higher value. In addition, the right hand side plot of Fig. 2 shows the norm between the true demand and the estimated one. We see that this distance converges to a lower value when solving the joint problem in contrast to all other methods. Highlighting the benefits of solving the joint formulation over other approaches. It is imperative to point out that the algorithm is sensitive to the selected parameters. In particular, the selection of step sizes is relevant as it may cause the algorithm to diverge. We believe this happens because the matrix $\mathbf{A}$ is evaluated at the previous iterations and therefore, the selection of $(c_{1j}, c_{2j}, d_{1j}, d_{2j})$ has a direct impact on both the closeness to the previous iteration and the algorithm's convergence rate.

2) Multi-Class Braess Network: We introduce two types of vehicles: *cars* and *commercial trucks* ($|\tilde{\mathcal{U}}| = 2$). For each vehicle class we generate ground truth demands $\mathbf{g}^*_{\text{car}} = [4000, 0]$ and , $\mathbf{g}^*_{\text{truck}} = [400, 0]$ for OD pairs $(1, 2)$ and $(2, 1)$, respectively. Recall that now we are interested on estimating the OD demand for each vehicle type, i.e., $\mathbf{g}_{\text{car}}$ and $\mathbf{g}_{\text{truck}}$, and the travel latency function $f(\cdot)$. We select $T = 500$, $\lambda = 0.1$, $\rho = 0.1$, $n = 5$, $c_{1j} = c_{2j} = \frac{400}{j}$, $d_{1j} = d_{2j} = 0.05/j$ and run Alg. 1 with parameters as in the single-class case (i.e., $f(x) = 1 + 0.15x^4$, $\mathbf{g}^0 = \mathbf{0}$). To consider the relatively lower speed and larger physical dimensions of trucks compared to cars, we assume the flow weight vector to be $\boldsymbol{\theta} = (1, 2)$ for cars and trucks, respectively. We assume $t^0_{i,\text{car}} = t^0_i$ and $t^0_{i,\text{truck}} = 1.1t^0_i$, where t^0_i is the *free-flow travel time* for the *physical* link indexed by i. The details regarding the road characteristics used for the free flow speeds and capacities, as well as the algorithms employed in this paper are available

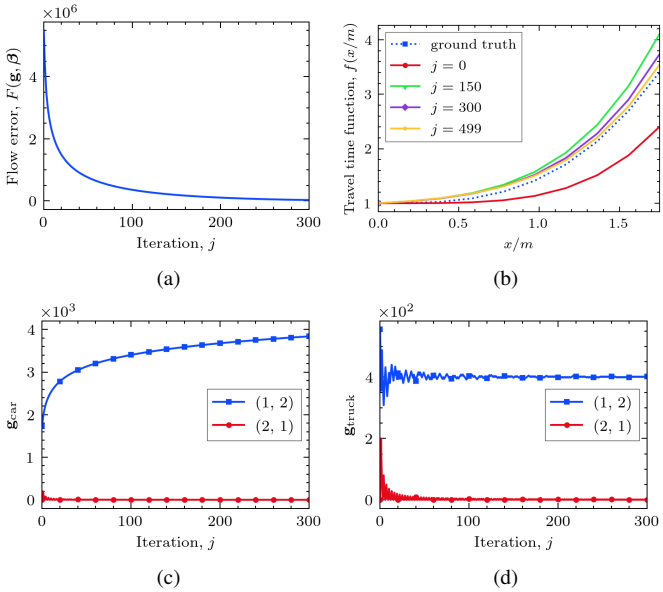


Fig. 3. Validation Results for the Multi-Class Braess Network. (a) Objective Function, (b) $f(\cdot)$ evolution, (c) Car demand estimator, (d) Truck demand estimator.

in our online public repository[1]. Figures 3c and 3d show how Alg. 1 recovers the true OD demands for both vehicle classes. In Fig. 3b we see how $\hat{f}(\mathbf{x})$ (our estimate of $f(x)$) is getting closer to the true travel latency function. Finally, in Fig. 3a we show how the objective function converges to a value close to zero. In this experiment we observe that we are able to recover the truth demand and travel latency function regardless the limitation of the multi-class model as pointed in Sec III-A.

3) EMA: Seeking for more realistic networks, we perform a validation experiment using the road network of EMA. We select this network since it helps to emulate the conditions of a interstate highway road network. The graph (Fig. 1b) consists of 8 nodes, 24 links and 56 OD pairs.

We run Algs. 1 and 2 with $T = 60$, $\lambda = 0.1$, $\rho = 0.1$, $n = 5$, $c_{1j} = c_{2j} = \frac{250}{j}$, $d_{1j} = d_{2j} = \frac{0.02}{\sqrt{j}}$ and we initialize it with $\boldsymbol{\beta}^0 = (1, 0, 0, 0, 0, 15, 0)$ and $\mathbf{g}^0 = \mathbf{0}$. In the left plot of Fig. 4, we observe that jointly adjusting the cost function while estimating demands is beneficial for approximating the flows. We believe this is the case since Alg. 1 takes the joint steepest descent direction compared with taking these steps separately as in the Alternating method. In the right hand side plot of Fig. 4, we see the progress of $\|(\mathbf{g}^j - \mathbf{g}^*)\|$ (which is not minimized explicitly). We observe the joint method reaches a lower level than all of the other approaches, especially against the BPR approach. However, after iteration 9, this quantity begins to increase. We believe this happens due to the fact that the optimization process advances by adjusting both $\hat{f}$ and $\mathbf{g}$ and might deviate the estimated demand while tuning the travel latency function.

4) NYC: To test our method in congested urban areas, we created a NYC validation network (Fig. 1c) consisting of 28 nodes, 90 edges and 8 Zones (green dots). To build this network we used two data sources: OpenStreetMaps, from which we retrieve the network topology and road characteristics, and

[1]https://github.com/salomonw/tnet

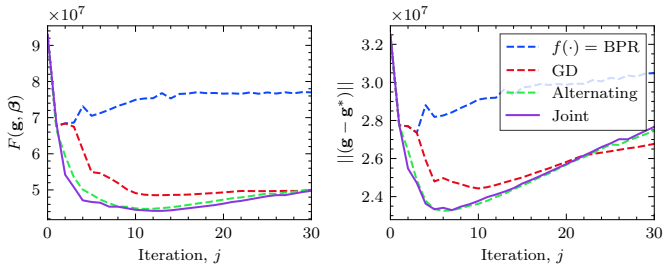


Fig. 4. Validation Results for the EMA Network. Left and right plots show the progress over the iterations of the objective function and the OD demand deviation from a specified "truth" demand $\mathbf{g}^*$, respectively.

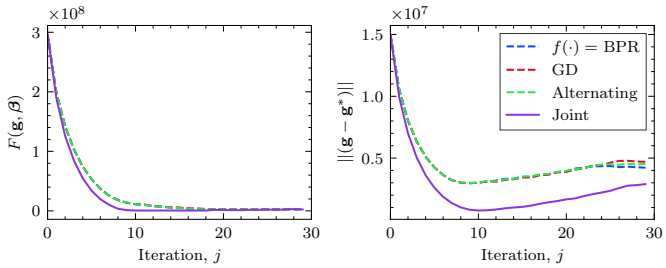


Fig. 5. Validation Results for the NYC Network. Left and right plots show the progress over the iterations of the objective function and the OD demand deviation from a specified "truth" demand $\mathbf{g}^*$, respectively

Uber Movement Speed Data set, which we use for assigning speed data to each link in the network. With this in hand, we estimated travel times, speeds, densities and flows. See Appendix A for details.

We run the algorithms with $T = 30$, $\lambda = 0.1$, $\rho = 0.1$, $n = 5$, $c_{1j} = c_{2j} = \frac{100}{\sqrt{j}}$, $d_{1j} = d_{2j} = \frac{0.02}{j^{(3/4)}}$, $\boldsymbol{\beta}^0 = (1, 0, 0, 0, 0, 15, 0)$, and $g_i^0 = 1000$ for all $i \in [\![\mathcal{W}]\!]$. Consistent with the other examples we observe in Fig. 5 similar results as in the Braess and the EMA networks where using the joint methodology is beneficial.

B. Case Studies

1) EMA: We use speed data from April, 20th, 2012 from 8 to 9 a.m. provided by the Central Transportation Planning Staff (CTPS) of the Boston Metropolitan Planning Organization (MPO). We converted this speed to flow data using the procedure described in Appendix B. With this, we run the algorithms using $T = 60$, $\lambda = 1e-3$, $\rho = 0.1$, $n = 5$, $c_{1j} = c_{2j} = \frac{100}{j}$, $d_{1j} = d_{2j} = \frac{0.02}{\sqrt{j}}$, $\boldsymbol{\beta}^0 = (1, 0, 0, 0, 0, 15, 0)$, and $\mathbf{g}^0 = \mathbf{0}$. And report the estimated OD matrix in Table I.

From field observations we understand that the traffic patterns for EMA are similar to most urban areas. In the morning, commuters travel to work towards the city center and on the afternoon, they travel to residential areas. Table I matches this intuition by estimating higher demand for OD pairs with destination in the city center. In particular for nodes 4 (Worcester) and 6 (Downtown Boston). Likewise, the origin for which most flow is generated is 8 (Taunton), corresponding to the southern most node which accounts for all the southern flow that commutes towards the Boston Metropolitan Area. In addition, our results also suggests that travelers are more sensitive to lower flows than the ones proposed by the BPR function (see that the estimated Joint $f(\cdot)$ is above the BPR in the left plot of Fig. 6). This is relevant for designing and planning transportation networks.

TABLE I
ESTIMATED OD (VEH/H) DEMANDS FOR THE EMA NETWORK.

O/D	1	2	3	4	5	6	7	8	sum
1	0.0	1.1	0.0	515.1	298.1	803.4	36.3	310.0	1964.1
2	0.0	0.0	0.0	590.5	205.3	645.3	130.5	325.8	1897.3
3	0.0	1.1	0.0	598.6	173.6	662.4	130.3	257.0	1822.9
4	0.0	1.1	0.0	0.0	349.6	720.4	20.2	331.7	1423.1
5	0.0	1.1	0.0	563.6	0.0	645.1	87.4	320.9	1618.1
6	0.0	1.1	0.0	483.5	184.3	0.0	1.7	212.8	883.3
7	0.0	1.1	0.0	619.2	283.4	807.9	0.0	197.9	1909.5
8	1706.3	0.0	0.0	1355.2	65.8	898.0	1.7	0.0	**4027.0**
sum	1706.3	6.6	0.0	**4725.6**	1560.1	**5182.5**	408.1	1956.1	15545.3

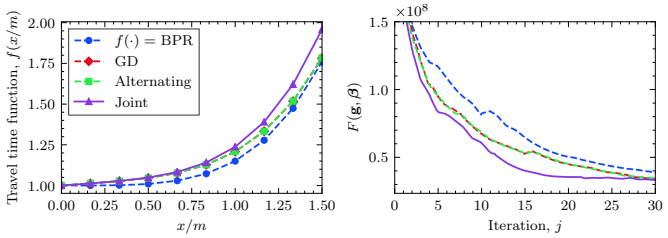


Fig. 6. Case Study results for EMA

2) New York City Network: We perform a similar case study using the NYC subnetwork. We run our algorithms for traffic data on Feb 13th, 2017 at 9 am. The right hand plot in Fig 7 show our results of the joint approach yielding to lower flow error than the other estimation methods. Similar to the EMA case, these results confirm our understanding of the morning traffic pattern in NYC. Table II shows that node 10 is a favorite origin and destination. This node represents Midtown, one of the busiest neighborhoods in Manhattan due to high density of offices and businesses. Moreover, we see that nodes 1 and 20 are desired destinations. These nodes represent Wall Street and Lower Manhattan areas, respectively. This follows our intuition as the south of Manhattan (also known as the Financial District) it is densely populated with offices.

In summary, the numerical results reported in this Section suggest that Alg. 1 works well in terms of reducing the objective function value of (12) while improving the estimation accuracy for the cost function and demands.

VI. LIMITATIONS

We acknowledge some limitations of our method that we believe can serve as a basis for future work, in particular we identify: (i) We cannot guarantee the adjusted OD demands to be close to the ground truth demand (since we are estimating $\mathbf{g}$ by closing the gap between $\mathbf{x}$ and $\mathbf{x}^*$) and the problem is non-convex; (ii) In practice, the output of our algorithm would heavily depend on the initial demand data (since the joint problem is non-convex), as well as, on the accuracy of the flow observations. Hence, good initial estimates are important for the success of recovering the true parameters; (iii) The selection of $(c_{1j}, c_{2j}, d_{1j}, d_{2j})$ has a direct impact on both the closeness to the previous iterate and the algorithm's convergence rate which trades-off speed and accuracy. To overcome the parameter selection issue, we suggest that practitioners use cross-validation techniques or armijo-type rules when possible; (iv) At each iteration we are solving a QP which can be solved in polynomial time. However, the requirement for memory increases exponentially with the size of the network and the number of OD pairs. Therefore, decomposition techniques

TABLE II
ESTIMATED OD DEMANDS (VEH/H) FOR THE NYC NETWORK.

O/D	1	4	20	10	17	21	25	26	sum
1	0.0	20.6	243.2	210.1	0.0	0.0	0.0	0.0	473.9
4	0.0	0.0	312.8	199.3	0.0	0.0	0.0	0.0	512.2
20	0.0	156.2	0.0	0.0	0.0	239.4	162.8	0.0	558.4
10	713.8	0.0	0.0	0.0	270.1	0.0	0.0	19.7	**1003.6**
17	0.0	86.4	0.0	0.0	0.0	185.5	140.5	0.0	412.4
21	0.0	86.4	0.0	0.0	0.0	0.0	140.5	0.0	226.9
25	0.0	0.0	0.0	52.2	0.0	0.0	0.0	41.1	93.3
26	0.0	0.0	0.0	52.2	0.0	0.0	0.0	0.0	52.2
sum	**713.8**	349.6	**556.0**	**513.9**	270.1	424.8	443.8	60.8	3332.8

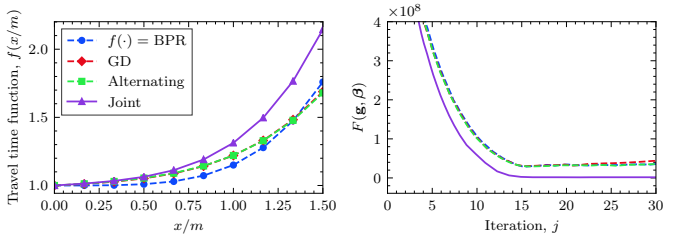


Fig. 7. Case Study results for the NYC

for the QP should be explored to increase the computational efficiency of the method.

VII. CONCLUSION AND FUTURE WORK

In this paper, we address the problem of estimating OD demands and cost functions jointly in a multi-class transportation network. We tackle this problem by rewriting the bilevel optimization problem (12) using the lower-level optimality conditions (16). To solve the resulting model, we propose an iterative approach (17) by relaxing some constraints and allowing a penalized gap to exist between the primal and dual objectives. To solve the joint problem we proposed a *trust-region feasible direction* and an *alternating* method described in Algs. 1 and 2, respectively.

Our results show that we can always reduce the objective function value of (12) to some extent, sometimes significantly, thanks to the the construction of the algorithms. However, the precision of the output highly depends on the inputs due to the non-convexity nature of the bilevel problem. We show empirically that there is value in jointly solving this problem as it reaches better estimates of the flows $\mathbf{x}$, cost function $f(\cdot)$, and OD demands $\mathbf{g}$. To do it, we compare our methods with the one of exclusively adjusting $\mathbf{g}$ and with a vanilla gradient decent method, which our approaches outperformed. Nevertheless, we believe the GD and alternating methods are good alternatives when dealing with large networks. This is because GD and Alternating typically converge close to the Joint solution (see Fig. 4 and Fig. 5) and rely exclusively on the solution of the TAP [44], [45], [46]. We hope this work nudges transportation experts to include cost function adjustments when estimating OD demands from data. An additional advantage of our methods is that it solves the problem using gradient descent. This helps to overcome the burden of inverting large sparse matrices required when estimating the OD demands as in other methods [7]. This implies that our method is easier to implement, and allow for decomposition schemes as network size and OD pairs increase.

We identify potential future directions: First, to perform sensitivity analysis of link congestion metrics with respect to key quantities, such as link capacity and free-flow speed trough the estimation of $f(\cdot)$ to enable the transportation agencies to prioritize congestion-reducing interventions [47]. Second, to integrate our algorithms to a dynamic OD demand estimation problem setting e.g., [48]. The potential outcome would be to provide a more trustworthy method to predict the "Estimated Time of Arrival" more accurately. Third, it is possible to improve the running time of our algorithm by (i) utilizing previous iterations as starting feasible solutions for subsequent iterates; (ii) exploring better stopping criteria; (iii) implementing coordinate descent schemes and accelerated methods; (iv) employing more efficient data structures, and (v) aggregating flows at the link- or bush-level.

VIII. ACKNOWLEDGMENTS

We would like to thank the Boston Region MPO, and Scott Peterson in particular, for supplying the data and providing us invaluable clarifications throughout our work. The data for the NYC network was retrieved from Uber Movement, (c) 2019 Uber Technologies, Inc., https://movement.uber.com.

REFERENCES

[1] H. Yang, Q. Meng, and M. G. H. Bell, "Simultaneous estimation of the origin-destination matrices and travel-cost coefficient for congested networks in a stochastic user equilibrium," *Transportation Science*, vol. 35, no. 2, pp. 107–123, 2001.

[2] D. Bertsimas, V. Gupta, and I. C. Paschalidis, "Data-driven estimation in equilibrium using inverse optimization," *Mathematical Programming*, vol. 153, no. 2, pp. 595–633, 2015.

[3] H. J. Van Zuylen and L. G. Willumsen, "The most likely trip matrix estimated from traffic counts," *Transportation Research Part B: Methodological*, vol. 14, no. 3, pp. 281–293, 1980.

[4] M. G. Bell, "The estimation of an origin-destination matrix from traffic counts," *Transportation Science*, vol. 17, no. 2, pp. 198–217, 1983.

[5] M. Brenninger-Göthe, K. O. Jörnsten, and J. T. Lundgren, "Estimation of origin-destination matrices from traffic counts using multiobjective programming formulations," *Transportation Research Part B: Methodological*, vol. 23, no. 4, pp. 257–269, 1989.

[6] E. Cascetta, "Estimation of trip matrices from traffic counts and survey data: a generalized least squares estimator," *Transportation Research Part B: Methodological*, vol. 18, no. 4-5, pp. 289–299, 1984.

[7] M. L. Hazelton, "Estimation of origin-destination matrices from link flows on uncongested networks," *Transportation Research Part B: Methodological*, vol. 34, no. 7, pp. 549–566, 2000.

[8] H. Spiess, "A maximum likelihood model for estimating origin-destination matrices," *Transportation Research Part B: Methodological*, vol. 21, no. 5, pp. 395–412, 1987.

[9] M. J. Maher, "Inferences on trip matrices from observations on link volumes: a bayesian statistical approach," *Transportation Research Part B: Methodological*, vol. 17, no. 6, pp. 435–447, 1983.

[10] E. Cascetta and S. Nguyen, "A unified framework for estimating or updating origin/destination matrices from traffic counts," *Transportation Research Part B: Methodological*, vol. 22, no. 6, pp. 437–455, 1988.

[11] C. Fisk, "On combining maximum entropy trip matrix estimation with user optimal assignment," *Transportation Research Part B: Methodological*, vol. 22, no. 1, pp. 69–73, 1988.

[12] ——, "Trip matrix estimation from link traffic counts: the congested network case," *Transportation Research Part B: Methodological*, vol. 23, no. 5, pp. 331–336, 1989.

[13] H. Spiess, "A gradient approach for the od matrix adjustment problem," *Publication CRT-693, Centre de recherche sur les transports, Universite de Montreal, Montreal, Quebec, Canada*, vol. 1, p. 2, 1990.

[14] M. Florian and Y. Chen, "A coordinate descent method for the bi-level od matrix adjustment problem," *International Transactions in Operational Research*, vol. 2, no. 2, pp. 165–179, 1995.

[15] E. Cascetta and M. N. Postorino, "Fixed point approaches to the estimation of o/d matrices using traffic counts on congested networks," *Transportation Science*, vol. 35, no. 2, pp. 134–147, 2001.

[16] H. Yang, T. Sasaki, Y. Iida, and Y. Asakura, "Estimation of origin-destination matrices from link traffic counts on congested networks," *Transportation Research Part B: Methodological*, vol. 26, no. 6, pp. 417–434, 1992.

[17] M. J. Maher, X. Zhang, and D. Van Vliet, "A bi-level programming approach for trip matrix estimation and traffic control problems with stochastic user equilibrium link flows," *Transportation Research Part B: Methodological*, vol. 35, no. 1, pp. 23–40, 2001.

[18] J. Doblas and F. G. Benitez, "An approach to estimating and updating origin–destination matrices based upon traffic counts preserving the prior structure of a survey matrix," *Transportation Research Part B: Methodological*, vol. 39, no. 7, pp. 565–591, 2005.

[19] A. Skabardonis and R. Dowling, "Improved speed-flow relationships for planning applications," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1572, no. 1, pp. 18–23, 1997.

[20] E. T. Mtoi and R. Moses, "Calibration and evaluation of link congestion functions," *Journal of Transportation Technologies*, vol. 4, no. 2, 2014.

[21] Z. Lu, Q. Meng, and G. Gomes, "Estimating link travel time functions for heterogeneous traffic flows on freeways," *Journal of Advanced Transportation* , vol. 50, no. 8, pp. 1683–1698, 2016.

[22] R. Neuhold and M. Fellendorf, "Volume delay functions based on stochastic capacity," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2421, no. 1, pp. 93–102, 2014.

[23] R. García-Ródenas and D. Verastegui-Rayo, "Adjustment of the link travel-time functions in traffic equilibrium assignment models," *Transportmetrica A: Transport Science*, vol. 9, no. 9, pp. 798–824, 2013.

[24] F. Russo and A. Vitetta, "Reverse assignment: calibrating link cost functions and updating demand from traffic counts and time measurements," *Inverse Problems in Science and Engineering*, vol. 19, no. 7, pp. 921–950, 2011.

[25] S. Liu and J. D. Fricker, "Estimation of a trip table and the θ parameter in a stochastic network," *Transportation Research Part A: Policy and Practice*, vol. 30, no. 4, pp. 287–305, 1996.

[26] R. L. Tobin and T. L. Friesz, "Sensitivity analysis for equilibrium network flow," *Transportation Science*, vol. 22, no. 4, pp. 242–250, 1 1988.

[27] H.-P. Lo and C.-P. Chan, "Simultaneous estimation of an origin–destination matrix and link choice proportions using traffic counts," *Transportation Research Part A: Policy and Practice*, vol. 37, no. 9, pp. 771–788, 2003.

[28] Y. Wang, X. Ma, Y. Liu, K. Gong, K. C. Henricakson, M. Xu, and Y. Wang, "A two-stage algorithm for origin-destination matrices estimation considering dynamic dispersion parameter for route choice," *PLoS ONE*, vol. 11, no. 1, p. e0146850, 2016.

[29] Q. Meng, D.-H. Lee, and R. L. Cheu, "A bilevel programming approach to simultaneously estimate od matrix and calibrate link travel time functions from observed link flows," in *Applications of Advanced Technologies in Transportation Engineering (2004)*, 2004, pp. 56–60.

[30] S. Wollenstein-Betech, C. Sun, J. Zhang, and I. C. Paschalidis, "Joint Estimation of OD Demands and Cost Functions in Transportation Networks from Data," in *Proc. IEEE Conf. on Decision and Control*, 2019.

[31] S. C. Dafermos, "The traffic assignment problem for multiclass-user transportation networks," *Transportation Science*, vol. 6, no. 1, pp. 73–87, 1972.

[32] M. Patriksson, *The Traffic Assignment Problem: Models and Methods*. Dover Publications, 1994, vol. 54, no. 2.

[33] M. J. Smith, "The existence, uniqueness and stability of traffic equilibria," *Transportation Research Part B: Methodological*, vol. 13, no. 4, pp. 295–304, 1979.

[34] P. Marcotte and L. Wynter, "A new look at the multiclass network equilibrium problem," *Transportation Science*, vol. 38, no. 3, pp. 282–292, 2004.

[35] M. Frank and P. Wolfe, "An algorithm for quadratic programming," *Naval research logistics quarterly*, vol. 3, no. 1-2, pp. 95–110, 1956.

[36] T. Evgeniou, M. Pontil, and T. Poggio, "Regularization networks and support vector machines," *Advances in Computational Mathematics*, vol. 13, no. 1, p. 1, 2000.

[37] J. Zhang, S. Pourazarm, C. G. Cassandras, and I. C. Paschalidis, "The price of anarchy in transportation networks: Data-driven evaluation and reduction strategies," *Proc. of the IEEE*, vol. 106, no. 4, pp. 538–553, April 2018.

[38] M. Patriksson, "Sensitivity analysis of traffic equilibria," *Transportation Science*, vol. 38, no. 3, pp. 258–281, Aug. 2004.

[39] Y. Noriega and M. Florian, *Algorithmic approaches for asymmetric multi-class network equilibrium problems with different class delay relationships*. CIRRELT, 2007.

[40] M. Josefsson and M. Patriksson, "Sensitivity analysis of separable traffic equilibrium equilibria with application to bilevel optimization in network design," *Transportation Research Part B: Methodological*, vol. 41, no. 1, pp. 4–31, 2007.

[41] J. T. Lundgren and A. Peterson, "A heuristic for the bilevel origin–destination-matrix estimation problem," *Transportation Research Part B: Methodological*, vol. 42, no. 4, pp. 339–354, 2008.

[42] R. García-Ródenas and Á. Marín, "Simultaneous estimation of the origin–destination matrices and the parameters of a nested logit model in a combined network equilibrium model," *European Journal of Operational Research*, vol. 197, no. 1, pp. 320–331, 2009.

[43] D. Bertsekas, *Nonlinear programming*, 3rd ed. Athena Scientific, 2016.

[44] L. J. LeBlanc, E. K. Morlok, and W. P. Pierskalla, "An efficient approach to solving the road network equilibrium traffic assignment problem," *Transportation Research*, vol. 9, no. 5, pp. 309–318, 1975.

[45] S. Nguyen and S. Pallottino, "Equilibrium traffic assignment for large scale transit networks," *European Journal of Operational Research*, vol. 37, no. 2, pp. 176–186, 1988.

[46] C. N. Yahia, V. Pandey, and S. D. Boyles, "Network partitioning algorithms for solving the traffic assignment problem using a decomposition approach," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2672, no. 48, pp. 116–126, 2018.

[47] S. Wollenstein-Betech, M. Salazar, A. Houshmand, M. Pavone, I. C. Paschalidis, and C. G. Cassandras, "Routing and rebalancing intermodal autonomous mobility-on-demand systems in mixed traffic," *IEEE Transactions on Intelligent Transportation Systems*, 2021, to appear.

[48] A. R. Pitombeira-Neto, C. F. G. Loureiro, and L. E. Carvalho, "A dynamic hierarchical bayesian model for the estimation of day-to-day origin-destination flows in transportation networks," *Networks and Spatial Economics*, pp. 1–29, 2020.

[49] OpenStreetMap, "Planet dump retrieved from https://planet.osm.org ," https://www.openstreetmap.org , 2017.

[50] Uber, "Uber movement," https://movement.uber.com, 2019, data retrieved from Uber Movement.

Salomón Wollenstein-Betech is a Ph.D. candidate in the Division of Systems Engineering at Boston University (BU). He received his B.S. degree in Industrial Engineering from Tecnologico de Monterrey in 2015. He is currently a member of the CODES Lab and the NOC Lab and is the recipient of the Dean's Fellowship awarded by BU. His research focuses on data-driven assessment, optimization and control of networks with a focus on Intelligent Transportation Systems

Chuangchuang Sun is an Assistant Professor at the aerospace engineering department at Mississippi State University since August 2021. Prior to that, he was a postdoctoral associate at MIT (2019-2021) and Boston University (2018-2019). He received his Ph.D. in August 2018 from the Ohio State University and a B.S. degree from the Beijing University of Aeronautics and Astronautics, China in 2013, both in Aerospace Engineering. His research interests focus on control, optimization, reinforcement learning with applications in robotics and space systems.

Jing Zhang received the B.S. degree in mathematics from Wuhan University, Wuhan, Hubei, China, in 2010, the M.S. degree in complex systems and control from the Institute of Systems Science, Chinese Academy of Sciences, Beijing, China, in 2013, and the Ph.D. degree in systems engineering from Boston University, Boston, MA, USA, in 2017. He is currently a Research Scientist at Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, USA. His research interests include anomaly detection, optimization, machine learning, and time series analysis. Dr. Zhang is a recipient of the Boston Area Research Initiative (BARI) Research Seed Grant Award (Spring 2017). He placed second within the New England states in the 2017 Net Impact & Toyota Next Generation Mobility Challenge.

Christos G. Cassandras (Life Fellow, IEEE) received the B.S. degree from Yale University, New Haven, CT, USA, in 1977, the M.S.E.E. degree from Stanford University, Stanford, CA, USA, in 1978, and the M.S. and Ph.D. degrees from Harvard University, Cambridge, MA, USA, in 1979 and 1982, respectively. He was with ITP Boston, Inc., Cambridge, from 1982 to 1984, where he was involved in the design of automated manufacturing systems. From 1984 to 1996, he was a Faculty Member with the Department of Electrical and Computer Engineering, University of Massachusetts Amherst, Amherst, MA, USA. He is currently a Distinguished Professor of engineering with Boston University, Brookline, MA, USA, the Head of the Division of Systems Engineering, and a Professor of electrical and computer engineering. He specializes in the areas of discrete event and hybrid systems, cooperative control, stochastic optimization, and computer simulation, with applications to computer and sensor networks, manufacturing systems, and transportation systems. He has authored over 450 refereed articles in these areas and six books. He is a fellow of the International Federation of Automatic Control (IFAC). He was a recipient of several awards, including the 2011 IEEE Control Systems Technology Award, the 2006 Distinguished Member Award of the IEEE Control Systems Society, the 1999 Harold Chestnut Prize (IFAC Best Control Engineering Textbook), the 2011 Prize and a 2014 Prize for the IBM/IEEE Smarter Planet Challenge competition, the 2014 Engineering Distinguished Scholar Award at Boston University, several honorary professorships, the 1991 Lilly Fellowship, and the 2012 Kern Fellowship. He was the Editor-in-Chief of the IEEE Transactions on Automatic Control from 1998 to 2009. He serves on several editorial boards and has been a guest editor for various journals. He was the President of the IEEE Control Systems Society in 2012. He is also a member of Phi Beta Kappa and Tau Beta Pi.

Ioannis Ch. Paschalidis (M'96–SM'06–F'14) received the Diploma degree in electrical and computer engineering from the National Technical University of Athens in 1991, and the M.S. and Ph.D. degrees, both in electrical engineering and computer science, from the Massachusetts Institute of Technology in 1993 and 1996, respectively. In September 1996, he joined Boston University, Boston, MA, where he has been ever since. He is a Professor and Data Science Fellow with Boston University having appointments with the Department of Electrical and Computer Engineering, the Division of Systems Engineering, the Department of Biomedical Engineering, and the Faculty of Computing and Data Sciences. He is the Director of the Center for Information and Systems Engineering. He has held visiting appointments with the MIT and Columbia University. His current research interests include the fields of systems and control, networks, machine learning, optimization, computational biology, and medical informatics. Dr. Paschalidis was the recipient of the National Science Foundation CAREER award (2000), several best paper and best algorithmic performance awards, and a 2014 IBM/IEEE Smarter Planet Challenge Award. He was an invited participant at the 2002 Frontiers of Engineering Symposium, organized by the U.S. National Academy of Engineering and the 2014 U.S. National Academies Keck Futures Initiative Conference. From 2013 to 2019, he was the Founding Editor-in-Chief for the IEEE Transactions on Control of Network Systems.

APPENDIX

A. Description of datasets

Eastern Massachusetts Area (EMA): The Boston Region Metropolitan Planning Organization (MPO) has given us access to two datasets of the EMA network: (i) A dataset reporting average speeds for more than 13,000 road segments for 2012 and 2015. For each road segment we obtained 10-min data on the average speed, and free-flow speeds, date, time, and travel time; (ii) A capacity (in *vehicles per hour*) dataset which includes data for more than 100,000 road segments in EMA.

New York City (NYC): We built the NYC transportation network through two open source datasets: (i) OpenStreetMaps (OSM) [49] from where we retrieved the network topology and road characteristics; (2) the *Uber Movement Speed Data set* [50] which contains average speeds of road segments on an hourly basis and which we matched with OSM.

B. Preprocessing

1) Selecting a sub-network: To mitigate the computational complexity while still capturing the key elements of the EMA network, we considered a representative highway sub-network (Fig. 1b) where we include 701 road segments, composing a network with 8 nodes and 24 links. We consider every node as a zone, yielding to 8×8 OD pairs. To provide an example using an urban environment, we reduced the dimensionality of the full NYC network. We first select the set of nodes from the full network to be the nodes of the NYC subnetwork (Fig. 1c). Then, we create edges between these nodes following the grid connectivity of NYC. To find the travel times of each arc, we solve a shortest path problem on the larger (original) network and use these travel times in the smaller network. With these travel times, we compute speeds on each link.

2) Calculating average speed and free-flow speed: For EMA and NYC, we construct the set $\mathcal{T}$ consisting of multiple time intervals of interest and calculate the average speed for every road segment in every time interval. Then, for each road segment we compute a proxy of the *free-flow speed* by using the 85th-percentile of the observed speeds on that segment.

3) Aggregating flows of the segments on each link: For $i \in [\![\tilde{\mathcal{A}}]\!]$, let $\{v_i^j, t_i^j, v_i^{0j}, t_i^{0j}, m_i^j;\ j = 1, \ldots, J_i\}$ denote the available observations (v_i^j, t_i^j), and parameters (v_i^{0j}, t_i^{0j}, m_i^j) of the segments composing the ith *physical* link. For each segment j, the average speed is v_i^j and the free-flow speed is v_i^{0j}, both in miles per hour. Moreover, t_i^j (resp., t_i^{0j}) is the travel time (resp., free-flow travel time) in hours, and m_i^j is the segment capacity in veh/hr). Then, using the fundamental diagram of traffic flow, we calculate the flow (in veh/hr) on segment j by $\hat{x}_i^j = \frac{4m_i^j}{v_i^{0j}} v_i^j - \frac{4m_i^j}{(v_i^{0j})^2}(v_i^j)^2$. In our analysis, we enforce $v_i^j \leq v_i^{0j}$ to make sure that the flow given by $\hat{x}_i^j$ is non-negative. In particular, if for some time instance $v_i^j > v_i^{0j}$ (this rarely happens), we set $v_i^j = v_i^{0j}$. Aggregating over all segments composed of link i we compute:

$$\hat{x}_i = \frac{\sum_{j=1}^{J_i} \hat{x}_i^j t_i^j}{\sum_{j=1}^{J_i} t_i^j};\ t_i^0 = \sum\nolimits_{j=1}^{J_i} t_i^{0j};\ m_i = \frac{\sum_{j=1}^{J_i} m_i^j t_i^{0j}}{\sum_{j=1}^{J_i} t_i^{0j}},$$

where $t_i^{0j} = v_i^j t_i^j / v_i^{0j}$, $j = 1, \ldots, J_i$.

4) Adjusting link flows to satisfy conservation: For $i \in [\![\tilde{\mathcal{A}}]\!]$, let $\hat{x}_i$ denote the original estimate of the flow on link i. Note that these may not comply with conservation of flow. Hence, we let x_i be its adjustment, and ξ_{iu} the flow percentage on link i for vehicle class $u \in [\![\tilde{\mathcal{U}}]\!]$ (note that $\xi_{iu} \geq 0$ and $\sum_{u=1}^{|\tilde{\mathcal{U}}|} \xi_{iu} = 1$). Then, $x_{iu} = \xi_{iu} x_i$ (recall that x_{iu} denotes the flow on link $a(i,u)$; i.e., x_{iu} is the flow on link i for vehicle class u). To ensure that the flows are conserved, we solve the following *Least Squares* problem:

$$\min_{\mathbf{x} \geq 0} \quad \sum_{i=1}^{|\tilde{\mathcal{A}}|} \sum_{u=1}^{|\tilde{\mathcal{U}}|} \xi_{iu}^2 \left(x_i - \hat{x}_i\right)^2 \tag{24a}$$

$$\text{s.t.} \quad \sum_{i \in \mathcal{I}(j)} \xi_{iu} x_i = \sum_{i \in \mathcal{O}(j)} \xi_{iu} x_i, \qquad \forall j \in \mathcal{V},\ u \in [\![\tilde{\mathcal{U}}]\!], \tag{24b}$$

where the first constraint enforces flow conservation for each node $j \in \tilde{\mathcal{V}}$ with $\mathcal{I}(j)$ (resp., $\mathcal{O}(j)$) denoting the set of incoming (resp., outgoing) links to (resp., from) node j.