Transportation Research Part B 119 (2019) 156–173

Contents lists available at ScienceDirect

Transportation Research Part B

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

Efficient calibration of microscopic car-following models for large-scale stochastic network simulators

Carolina Osorio[a,*], Vincenzo Punzo[b]

[a] Department of Civil and Environmental Engineering, MIT, USA
[b] Department of Civil, Architectural and Environmental Engineering University of Naples Federico II, Italy

ARTICLE INFO

Article history:
Received 19 November 2017
Revised 28 July 2018
Accepted 5 September 2018
Available online 5 December 2018

ABSTRACT

This paper proposes a simulation-based optimization methodology for the efficient calibration of microscopic traffic flow models (i.e., car-following models) of large-scale stochastic network simulators. The approach is a metamodel simulation-based optimization (SO) method. To improve computational efficiency of the SO algorithm, problem-specific and simulator-specific structural information is embedded into a metamodel. As a closed-form expression is sought, we propose adopting the steady-state solution of the car-following model as an approximation of its simulation-based input-output mapping. This general approach is applied for the calibration of the Gipps car-following model embedded in a microscopic traffic network simulator, on a large network. To this end, a novel formulation for the traffic stream models corresponding to the Gipps car-following law is provided.

The proposed approach identifies points with good performance within few simulation runs. Comparing its performances to that of a traditional approach, which does not take advantage of the structural information, the objective function is improved by two orders of magnitude in most experiments. Moreover, this is achieved within tight computational budgets, i.e., few simulation runs. The solutions identified improve the fit to the field measurements by one order of magnitude, on average. The structural information provided to the metamodel is shown to enable the SO algorithm to become robust to both the quality of the initial points and the simulator stochasticity.

1. Introduction

Our urban transportation systems are becoming increasingly connected, automated, real-time responsive and intricate. This has led to the design of higher resolution models that aim to describe both demand and supply in greater detail. A recent review of traffic simulation models is given in Barceló (2010). Nonetheless, as these models become more intricate, so does the challenge of properly calibrating them. The problem of model calibration consists of fitting the input parameters of the traffic model such as to replicate field-observed traffic patterns (e.g., link counts, link speeds). It has been extensively studied with seminal work such as (Cascetta et al., 1993; Cascetta and Nguyen, 1988). Reviews include Balakrishna (2006) and Zhang et al. (2017).

* Corresponding author at: MIT, 77 Massachusetts Avenue, Office 1-232, Cambridge, MA 02139, United States.
E-mail addresses: osorioc@mit.edu (C. Osorio), vinpunzo@unina.it (V. Punzo).

https://doi.org/10.1016/j.trb.2018.09.005
0191-2615/

The calibration of simulation-based network models remains a difficult optimization problem to address. It is a simulation-based optimization (SO) problem that is non-convex and often non-differentiable (Ciuffo et al., 2008). The underlying network simulator is stochastic. Hence, the performance metrics can only be estimated via simulation. Each estimation involves running multiple simulation replications, each of which is computationally costly to evaluate. Both the intricacy of the optimization problem and the cost of each single simulation run call for computationally efficient calibration algorithms that are able to yield good quality solutions within few simulations. Yet, there is a lack of efficient algorithms in the scientific literature.

Given the high computational cost of evaluating the simulator, the most common algorithmic approach for calibration is the use of general-purpose derivative-free optimization algorithms, such as the genetic algorithm or the Box-Cox algorithm (Vaze et al., 2009; Balakrishna et al., 2007; Kattan and Abdulhai, 2006; Stathopoulos and Tsekeris, 2004; Kunde, 2002). General-purpose algorithms that require first-order derivative information, yet that are designed to estimate it efficiently, have also been extensively adopted by the transportation community. For instance, the simultaneous perturbation stochastic approximation (SPSA) algorithm of Spall (2003) has been often used. Recent work has focused on tailoring it to transportation problems to improve its short-term (i.e., small sample) performance (Lu et al., 2015; Tympakianaki et al., 2015; Cipriani et al., 2011). A traditional approach to address calibration problems in a computationally efficient way has been the use of parallelization techniques (Huang, 2010), as well as the use of methods to reduce the problem dimensionality, such as global sensitivity analysis (Zhong et al., 2016; Punzo et al., 2015; Ciuffo and Azevedo, 2014; Ge et al., 2014; Ciuffo et al., 2013).

While general purpose optimization algorithms are designed to achieve asymptotic performance guarantees (e.g., asymptotic convergence properties), they are used for transportation calibration problems under tight computational budgets. Practitioners increasingly apply these models in computationally challenging settings (e.g., online applications), where good quality solutions are needed within few simulation evaluations. This paper, therefore, contributes to the challenge of designing efficient algorithms for the calibration of stochastic simulation-based traffic models.

In particular, it focuses on the calibration of microscopic traffic flow models (i.e., car-following) in large-scale network simulation models. In a network simulator, a given car-following model is specified for each vehicle class. The car-following model determines the speed of each vehicle for each simulation time step.

This calibration problem has attracted a significant amount of research efforts in recent years and is expected to become even more relevant with the increasing heterogeneity of technologies brought by the growth of vehicle automation. We view the efficient calibration of disaggregate vehicle-specific traffic models (e.g., car-following, lane-changing) as a particularly important problem for the next generation of transportation network models.

The main idea of the proposed approach is to achieve efficiency by formulating and embedding problem-specific structural information within the SO algorithm. Our approach is to formulate an analytical approximation of the input-output mapping between calibration parameters (i.e., the decision vector) and simulation outputs of interest (e.g., expected link counts, expected link speeds). In particular, this analytical mapping is derived from the microscopic vehicle-level models of the simulator. By embedding such structural information into the SO algorithm, the simulator is no longer a black-box for the algorithm itself. This differs from general-purpose algorithms that treat the simulator as a black-box. This allows the SO algorithm to identify good solutions within fewer simulation runs.

In order to embed the analytical mapping within the SO algorithm, we use a metamodel approach. A metamodel is defined as an analytical approximation of a simulation-based function. We use the derivative-free metamodel SO algorithm of Osorio and Bierlaire (2013), which defines the metamodel as the sum of a problem-specific component (known in the literature as a physical metamodel) and a general-purpose component (known in the literature as a functional metamodel). For details on the comparison of functional and physical metamodels, see Søndergaard (2003). In the metamodel literature, the most common choice of metamodels are functional metamodels such as low-order polynomial functions, radial-basis functions, Kriging functions. In the field of calibration, functional metamodels have been used in combination with sensitivity analysis (Azevedo et al., 2015; Ciuffo and Azevedo, 2014; Ge et al., 2014; Ciuffo et al., 2013).

This general idea of combining physical and functional metamodels has been successfully used to efficiently address various types of transportation optimization problems, including signal control problems (Chong and Osorio, 2018; Osorio and Nanduri, 2015), congestion pricing (Osorio and Atastoy, 2017) and demand calibration (Zhang et al., 2017). For a given optimization problem, the main challenge in developing an efficient metamodel SO algorithm is the formulation of the problem-specific (i.e., physical) model. In other words, the key to achieving efficiency is the formulation of an efficient analytical input-output mapping that approximates well the (unknown) simulation-based input-output mapping.

The idea of this paper is to formulate a steady-state expression of a car-following model. This expression analytically approximates the simulation-based (non-stationary) input-output mapping. It is used as the physical component of a metamodel.

Traffic stream models corresponding to a specific microscopic car-following law can be derived from the steady-state solution of that law (i.e., from the function that describes driver behavior at equilibrium) (see e.g., Treiber et al., 2000). Unlike classical traffic stream models (Hall, 2001), which involve only macroscopic quantities, the bivariate relationships derived from a car-following model relate the macroscopic traffic variables (i.e., flow, speed and density) to the microscopic parameters of the car-following model, such as maximum speed or minimum time headway (Treiber et al., 2000). Hence, they can be directly used to approximate the mapping between calibration parameters and expected network performance (e.g., expected link speeds).

In this paper, we formulate this idea to calibrate the Gipps car-following model (Gipps, 1981). The traffic stream models corresponding to the Gipps car-following model have been derived by Punzo and Tripodi (2007), based on the equilibrium speed-distance function obtained by Wilson (2001). We extend the models in Punzo and Tripodi (2007) such as to tailor them to a specific car-following model formulation that is used as part of a commonly used microscopic network simulator. In fact, the latter implements a car-following model that is an extension of the Gipps car-following model.

It is worth mentioning that, although the proposed approach is herein applied for a specific traffic simulator, which implements a specific car-following model, it is general. Whenever an equilibrium (or steady state) formulation of a car-following model can be derived, the proposed approach can be applied to calibrate the corresponding traffic simulator.

As the case studies of Section 4 indicate, the use of this analytical traffic model enables the SO algorithm to become computationally efficient, scalable and robust to both the simulator's stochasticity and to the quality of the initial points. The key to achieving these improvements is shown to be the proposed analytical traffic model formulation. The latter provides the SO algorithm with problem-specific and simulator-specific structural analytical information. By exploiting this structural information, the SO algorithm no longer treats the simulator as a black-box and can identify points with good performance within few simulation evaluations.

More specifically, Section 4 shows that the use of the analytical traffic model leads to a computationally efficient algorithm: it identifies, within few simulation evaluations, points that improve: (i) the objective function estimates by 2 orders of magnitude, and (ii) the fit to the field measurements (link speeds) by 1 order of magnitude, on average. The scalability of the proposed algorithm is achieved by formulating the analytical traffic model as a system of n nonlinear and n linear equations, for a network with n links. Since the traffic model is formulated as a system of nonlinear equations, it can be evaluated efficiently with a variety of traditional solvers.

This paper is structured as follows. Section 2 formulates the calibration problem of interest. The proposed methodology is formulated in Section 3. The method is compared to a benchmark method both for a small synthetic network as well as for a network of the city of Lausanne (Section 4). Conclusions are presented in Section 5.

2. Problem formulation

Different simulation softwares embed different car-following model specifications. The proposed methodology is suitable for software that embed car-following models that are based on the Gipps model, which is a popular and commonly used model. The formulation of the Gipps model is briefly summarized in Appendix A. For the case studies of this paper, we use a software that uses a car-following model which is an extension of the Gipps model. For instance, the software's car-following model accounts for local attributes for each vehicle (e.g., attributes of the link the vehicle is currently on).

In this work, we consider a single vehicle class (i.e., the speeds of all vehicles are governed by the same car-following model). We consider a single time interval, i.e., we do not use time-dependent field measurements nor time-dependent calibration parameters. Hereafter, the term speed refers to the expected space-mean speed. To formulate the calibration problem of interest, we introduce the following notation.

x	decision vector (i.e., calibration parameter vector);
$f(x)$	(unknown) simulation-based objective function;
u_1	endogenous simulation variables;
u_2	exogenous simulation parameters;
y_i	field measurement of average (space mean) speed of link i;
$E[S_i(x, u_1; u_2)]$	expected (space mean) speed of link i;
x_L	lower bound vector;
x_U	upper bound vector;
$\mathcal{I}$	set of links with sensors.

The calibration problem is formulated as follows:

$$\min_x f(x) = \sum_{i \in \mathcal{I}} (y_i - E[S_i(x, u_1; u_2)])^2 \tag{1}$$

subject to

$$x_L \leq x \leq x_U. \tag{2}$$

The objective function f Eq. (1) id defined as the sum of the squared distance between the speed measurement for link i, y_i, and the simulator's expected speed for link i, $E[S_i(x, u_1; u_2)]$, which is hereafter denoted $E[S_i]$. The latter expectation depends on the decision vector x (i.e., the calibration parameter vector), on the vector of endogenous simulation variables (e.g., link queue-lengths, route choice probabilities) and on the vector of exogenous simulation parameters (e.g., network topology, traffic management strategies). This expectation is an unknown function, which needs to be estimated via simulation. Lower and upper bound constraints are assumed Eq. (2). These bounds are analytical constraints. Hence the problem is a continuous SO problem with simulation-based objective function and analytical deterministic constraints.

In this paper, the decision vector x is a three dimensional vector that represents: (i) $\tilde{\tau}$: the sum of a reaction time parameter and an additional "comfort" lag (using the notation of the Gipps model in Appendix A: $\tilde{\tau} = \tau + \theta$), (ii) $v_{\max}$: the

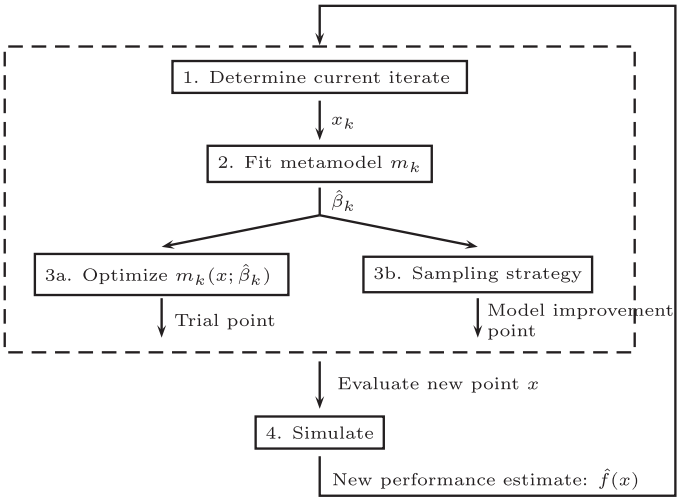

Fig. 1. Metamodel simulation-based optimization framework. Figure from Chong and Osorio (2018).

vehicles desired speed, (iii) s: the sum of the vehicle length and the minimum between-vehicle safety distance (using the notation of the Gipps model in Appendix A: $s = \tilde{S}_n \ \forall n$). In other words $x = [\tilde{\tau}, v_{\max}, s]$.

3. Methodology

To address Problem (1)–(2), we consider a metamodel SO approach. The main idea of metamodel SO approaches is to approximate the unknown and intricate (often non-convex) simulation-based objective function with an analytical function, which is known as the metamodel. The metamodel is a parametric function, the parameters of which are fitted, at every iteration of the algorithm, such as to minimize a distance function between metamodel predictions and simulated observations. At every iteration of the algorithm, the metamodel is optimized. Thus, at every iteration of the SO algorithm, an analytical optimization problem is solved. This allows for a variety of efficient optimization solvers for analytical problems to be used. This optimization of analytical subproblems contributes to the overall efficiency of the SO approach.

As previously discussed, our past work has proposed metamodel formulations for various classes of transportation optimization problems. The main goal of our SO work is to derive computationally efficient algorithms. That is, algorithms that can derive good quality solutions by simulating few points. This reflects how transportation practitioners, and even researchers, use SO algorithms: they terminate the algorithm when a finite, and typically small, set of points has been simulated. Our general approach to this challenge of developing efficient SO algorithms is to formulate metamodels that provide structural problem-specific information to the general-purpose SO algorithm. This differs from traditional SO algorithms that are suitable for a broader class of problems (e.g., non-transportation problems) and are designed such as to achieve asymptotic performance guarantees.

The main steps of a metamodel SO algorithm are depicted in Fig. 1. A detailed description of the SO calibration algorithm is given in Appendix B. At a given iteration k, consider the set of points (i.e. of decision vectors x) that have been simulated until iteration k. Let us denote this set as the current sample. Step 1 selects the point that is considered to have the best performance among the current sample. This point, denoted x_k, is known as the current iterate. Step 2 considers a metamodel function, $m_k(x;\ \beta_k)$, with parameter vector β_k. Step 2 uses the current sample to fit the parameter vector β_k; this yields $\hat{\beta}_k$. The goal of step 3a is to identify points that perform well according to the just estimated metamodel. Hence, Step 3a solves the (analytical) metamodel optimization problem for the decision vector x, whose solution is known as a trial point. It may also be of interest to simulate points that achieve other goals. For instance, one might want to simulate points that improve the fit of the metamodel, or points that improve geometric properties of the sampled space. These types of points are known as model improvement points, and are derived in step 3b. Step 4 considers any points obtained through steps 3a and 3b and simulates them to obtain estimates of their objective function, $\hat{f}(x)$. These steps are carried out at each SO iteration until the algorithm terminates. In our work, since we consider computationally inefficient simulators, we evaluate the performance of the algorithms for small sample sizes (i.e., few points are simulated). More specifically, we determine a priori a small maximum number of points to simulate, known as the computational budget, and we terminate the algorithm once this budget has been depleted. Hence, the solutions proposed by the method are not assumed to be optimal. Instead, the goal is to derive solutions with improved quality (compared to initial points, or to points used by practitioners) within few simulations.

In this paper, we use the SO algorithm of Osorio and Bierlaire (2013). The focus of this paper is on the formulation of a metamodel that enables the algorithm to address the calibration problem of interest in a computationally efficient way (i.e., with a small sample size). In other words, the focus of this paper is on the formulation of the function $m_k(x;\ \beta_k)$, such

that, with small samples, step 3a can yield solutions that perform well according to the simulator (i.e., they have low $\hat{f}(x)$ values).

To formulate the optimization problem solved at Step 3a, we use the following notation. The index k refers to a given iteration of the SO algorithm. The index i refers to a given link.

$m_k(x;\beta_k)$	metamodel function;
$\beta_{k,j}$	element j of the metamodel parameter vector β_k;
$\phi(x;\beta_k)$	functional metamodel component (polynomial function);
	Endogenous variables of the analytical traffic model:
x	decision vector;
$\tilde{\tau}$	sum of the reaction time and the "comfort" time lag;
$v_{\max}$	vehicle's desired (maximum) speed;
s	sum of the vehicle length and the minimum between-vehicle safety distance;
v_i	average (space-mean) speed of link i derived by the analytical traffic model;
k_i	average density of link i derived by the analytical traffic model;
	Exogenous parameters of the analytical traffic model:
$v_i^{\text{max-link}}$	maximum link speed;
b	deceleration parameter;
c_1, c_2, d_1, d_2	exogenous scalar parameters;
	Exogenous calibration problem parameters:
y_i	field measurement of average (space mean) speed of link i;
x_L	lower bound vector;
x_U	upper bound vector;
$\mathcal{I}$	set of links with sensors.

The proposed formulation for the metamodel optimization problem of Step 3a is given by:

$$\min_x m_k(x;\beta_k) = \beta_{k,0}\left(\sum_{i\in\mathcal{I}}(y_i - v_i)^2\right) + \phi(x;\beta_k) \tag{3}$$

$$x_L \le x \le x_U \tag{4}$$

$$v_i = \min\left\{v_{\max}, v_i^{\text{max-link}}, 3.6\frac{c_1\tilde{\tau}}{b}\left(-1+\sqrt{1+\frac{2(1000/k_i - s)b}{(c_1\tilde{\tau})^2}}\right)\right\} \tag{5}$$

$$k_i = c_2\frac{s-d_1}{d_2-d_1}. \tag{6}$$

Recall that the decision vector is the three-dimensional vector: $x = [\tilde{\tau}, v_{\max}, s]$. Problem (3)–(6) differs from Problem (1)–(2) in the following two ways. First, the (unknown) SO objective function (f of Eq. (1)) is replaced by an analytical function (m_k). Hence, Problem (3)–(6) is a fully analytical problem that can be addressed with traditional solvers. Second, there are a new set of constraints (5)–(6). It is this set of constraints that we refer to as the analytical traffic model. They provide an analytical approximation of how the calibration parameters (i.e. the decision vector) relate to (average) link speeds and densities. Constraints (4) are lower and upper bound constraints for the decision vector. These constraints are the same as those of the SO problem Eq. (2).

The objective function of this problem Eq. (3) is the metamodel. The first term of the metamodel is the sum of squares distance between the speed field measurements and the speeds derived by the analytical traffic model. This summation corresponds to the physical or problem-specific component of the metamodel. This summation represents the approximation of the objective function (f of Eq. (1)) provided by the analytical traffic model. This simple analytical approximation is corrected for parametrically in two ways: with a scaling parameter ($\beta_{k,0}$) and with an additive error term (ϕ). The latter corresponds to the functional or general-purpose metamodel component. It is defined as a quadratic polynomial with diagonal second-order derivative matrix:

$$\phi(x;\beta_k) = \beta_{k,1} + \sum_{j=1}^{D} x_j\beta_{k,j+1} + \sum_{j=1}^{D} x_j^2\beta_{k,j+D+1}, \tag{7}$$

where D denotes the dimension of the decision vector.

The key element of the metamodel (m_k of Eq. (3)) is the term v_i, which is an analytical problem-specific approximation of the simulation-based expected speed of link i ($E[S_i]$ of Eq. (1)). It is defined by Eq. (5), which itself is based on Eq. (9) of Punzo and Tripodi (2007). The latter equation is the macroscopic speed-density relationship arising from the Gipps car-following model (Gipps, 1981) at equilibrium (i.e., when vehicles follow each other at stationary speed).

Note that in this work, we aim to calibrate the car-following model of a network simulator, while in the work of Punzo and Tripodi (2007) the goal is to calibrate an isolated Gipps model (i.e., the car-following model is considered a stand-alone model, it is not embedded within a broader traffic network simulation tool). In particular, for the simulation software used in the case studies of this paper (Section 4), the standard (and often default) car-following model of the software is an extension of the Gipps model. The simulator's car-following model differs from the Gipps model in several ways. For instance, for every vehicle it uses local information (such as the maximum speed of the link that the vehicle is currently on) to specify the desired speed of each vehicle. To account for these differences, we have extended the expression in Punzo and Tripodi (2007) in the following two simple ways: (i) Eq. (5) uses the parameter c_1, which is not used in Punzo and Tripodi (2007), and (ii) the maximum link speed $v_i^{\text{max-link}}$ is accounted for.

In Eq. (6), we formulate an analytical (approximate) function that maps the spacing calibration parameter, s, and the average density of link i, k_i. Underlying this formulation is a linear proportionality assumption between the spacing parameter s and the density k_i. The exogenous scalar parameters of the analytical traffic model (c_1, c_2, d_1, d_2) allow to account for the differences between the Gipps car-following model and the Gipps extension deployed in a specific traffic network simulation software. They are estimated prior to optimization based on preliminary simulation results from small synthetic networks.

For a network of n links, the analytical traffic model (Eq. (5)–(6)) is defined as a system of n linear and n non-linear constraints. This makes it a scalable model suitable for the calibration of large-scale network models.

The deceleration parameter b in Eq. (5) is equivalent to the term denoted $1/b_n - 1/\hat{b}_{n-1}$ of the Gipps model in Appendix A. In this paper, we assume it is fixed. For the simulation software used for the case studies of this paper, the implementation of this deceleration parameter (TSS, 2010, Sections 13.6.1–13.6.2) differs from that formulated in the original Gipps model (Gipps, 1981). Hence, in this formulation we do not calibrate it.

The metamodel of Eq. (3) combines a functional metamodel term (ϕ) with a physical metamodel term (term within parenthesis of Eq. (3)). The latter has a functional form that is problem-specific. This paper contributes by formulating a problem-specific (i.e., physical metamodel) function that approximates the mapping between the microscopic car-following model calibration parameters and aggregate traffic metrics (e.g., speeds). More generally, the role of this physical component is to provide problem-specific analytical structural information to the SO algorithm. Recall that this analytical information is an approximation of the car-following model embedded in the simulator. Providing this information within the SO algorithm enables the algorithm to no longer treat the simulator as black-box. As is illustrated in the case studies of Section 4, this problem-specific information enhances both the computational efficiency of the SO algorithm and its robustness to the quality of the initial points.

In particular, the goal of the metamodel is not to replace the simulator. Instead, its goal is to provide an analytical approximate mapping of the simulator's input-output mapping that captures sufficient structural information such that it can guide the calibration algorithm to identify points with good quality performance within small computational budgets. In other words, its role is to accelerate the optimization process.

The formulation of a metamodel defined as the sum of a physical component and a functional component, as in Eq. (3), was first proposed in Osorio and Bierlaire (2013). In our past metamodel work, all metamodels have been based on this general specification. Linear or quadratic polynomials have been used as functional components (term ϕ of Eq. (3)). For a specific family of SO problems, the key to designing a computationally efficient algorithm lies in the formulation of the physical component. The latter should provide a good approximation of the SO objective function such that points with good (simulation-based) performance can be identified within few simulation evaluations. In other words, the physical component should capture sufficient problem-specific structural information to enable the acceleration of the optimization process. Nonetheless, since the metamodel optimization problem (Problem (3)–(6)) is solved at every iteration of the SO algorithm, it needs to be solved efficiently (otherwise, one is better off allocating the computational resources to evaluating the simulation-based objective function). Hence, the physical model should also be efficient. Therefore, the main challenge in formulating an efficient metamodel approach lies in the formulation of a physical component that provides a suitable balance between accuracy and efficiency.

Past work has formulated physical metamodel components for the calibration of route choice models (Zhang et al., 2017) and for the calibration of OD matrices (Zhang and Osorio, 2017; Osorio, 2018, 2017). This past work did not require a detailed description of car-following behavior. Hence, the analytical traffic models used relied on traditional fundamental diagrams formulated based on the link's supply parameters (e.g., maximum speed, jam density, flow capacity). Since the proposed work aims to calibrate car-following model parameters, a different and novel formulation for the analytical traffic model is proposed. The latter relates the parameters of the car-following model to the link's traffic conditions (e.g., average speeds and densities).

The proposed analytical traffic model assumes exogenous route choice, while those of Zhang et al. (2017) and Osorio (2018, 2017) allow for endogenous route choice. The proposed model can be extended to account for endogenous route choice following the ideas in Osorio (2018, 2017). The proposed model is similar to that of Zhang et al. (2017) in that it uses a single metamodel, rather than one metamodel per link with a measurement sensor.

4. Case studies

In this section, we evaluate the performance of the proposed method with experiments on two networks: (i) a small synthetic network, (ii) the Lausanne city network. We compare the performance of our proposed method to that of an SO

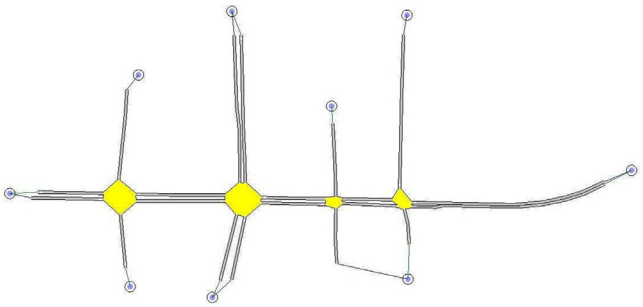

Fig. 2. Road network.

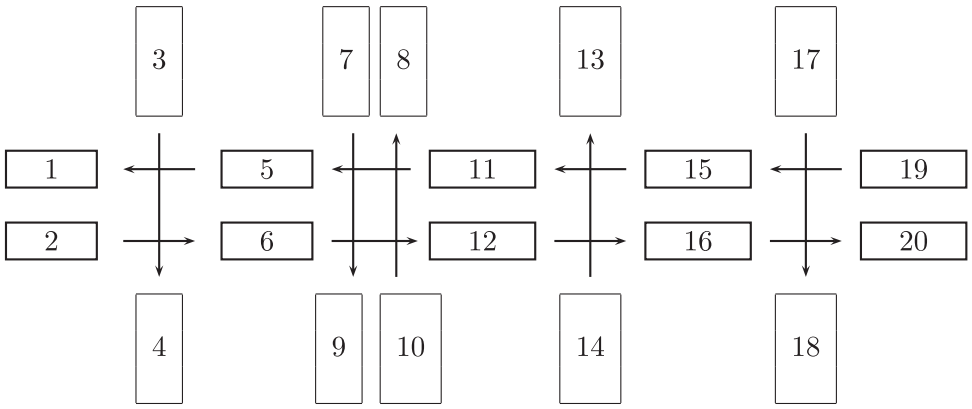

Fig. 3. Road network: permitted turning movements.

Table 1
Demand in vehicles per hour for the synthetic network.

Origin → destination	19 → 1	2 → 20	3 → 4	7 → 9	10 → 8	14 → 13	17 → 18
Demand level	700	700	100	600	600	100	100

method that differs only in that it does not include the structural equations Eq. (5)–(6). In other words, the benchmark method uses a metamodel that consists only of the ϕ term in Eq. (3). Both methods use the same metamodel SO algorithm and differ only in the metamodel choice. This comparison serves to evaluate the added value of the problem-specific component of the metamodel (i.e., the term in parenthesis of Eq. (3)). The benchmark method is referred to as the traditional method, since the use of a polynomial metamodel is a traditional and common choice in the metamodel literature.

For both networks, we use synthetic speed data. This means that the term y_i of Eqs. (1) and (3) is obtained by considering a given decision vector and assuming it to be the "true" optimal solution. We then use it to create synthetic "true" speed observations by running 50 simulation replications. For each link with a sensor, we take the sample average (over the 50 simulation replications) speed and consider it to be the "true" speed.

4.1. Synthetic network

The synthetic road network is that of Osorio and Yamani (2017). It is displayed in Fig. 2. It consists of 20 single-lane roads and 4 signalized intersections. Drivers travel along a single direction (i.e., they do not turn within the network). External arrivals and departures to the network occur at the boundaries of the network (represented by the circles in Fig. 2). The network is also represented in Fig. 3, where each road is denoted with a rectangle. Using the link labeling of Fig. 3, we consider the demand scenario with origin-destination demand defined in Table 1. The indices in the first row of Table 1 correspond to the link indices of Fig. 3. We assume an initially empty network and allow for a 15 minute warm-up period followed by a 1 h simulation. We assume that all 20 links have sensors. We use the Aimsun simulator (TSS, 2010).

We consider 6 random initial points, which are uniformly drawn from the feasible region Eq. (2). For each initial point, we run each method (i.e., algorithm) 5 times. Running a method multiple times for the same scenario allows us to evaluate the impact of stochasticity on the method's performance. We consider a computational budget of 50 points. In other words,

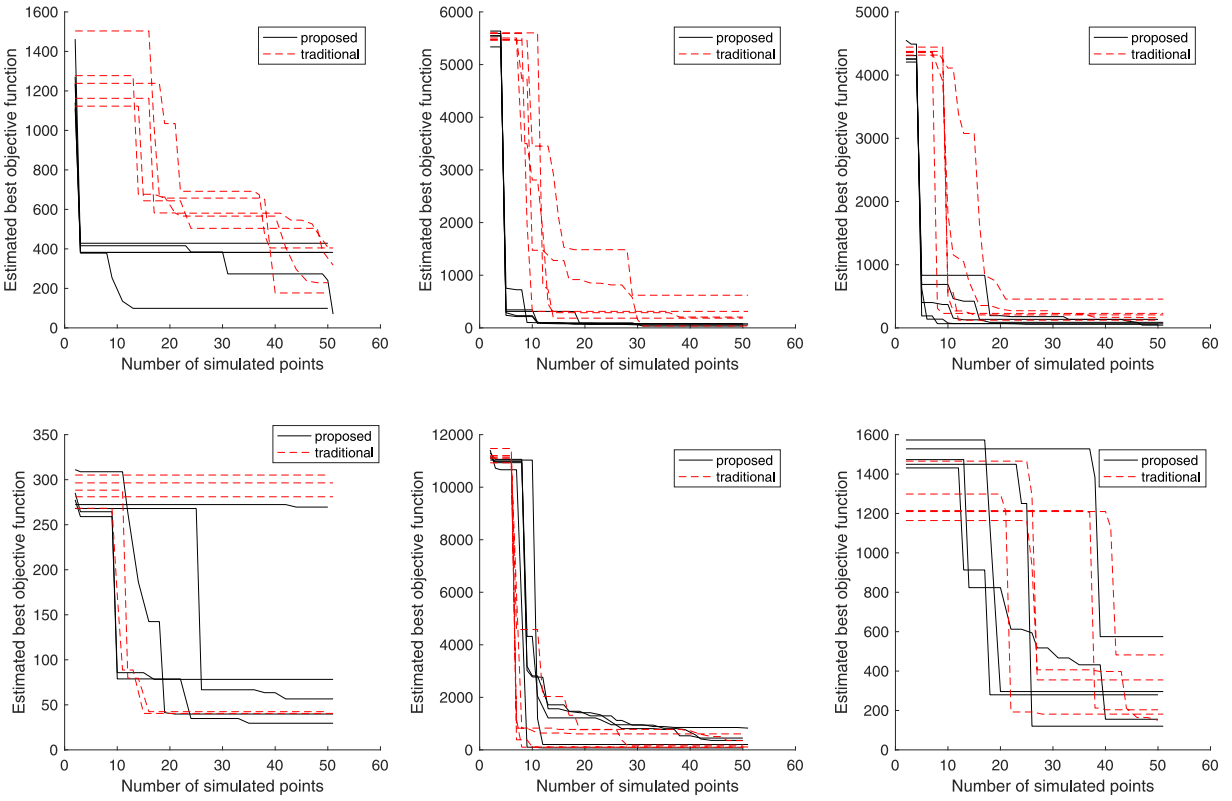

Fig. 4. Estimated performance of the current iterate (i.e., best point found so far) as a function of the number of simulated points. Each plot represents one initial point and displays the performance of the proposed and of the traditional methods.

an algorithm is terminated once it has simulated 50 points. Each point is evaluated by running 5 replications and using the sample average to estimate the objective function.

Fig. 4 isplays six plots. Each plot considers a given initial point. Each plot contains 5 black solid (respectively, red dashed) curves that represent the 5 runs of the proposed (resp. traditional) method. The x-axis considers the number of simulated points, the y-axis considers the objective function estimate of the best point found so far (i.e., the current iterate). Note that the different plots have different y-axis limits. For 3 of the 6 points (i.e., the top plots), the proposed method tends to identify points with lower objective functions faster than the traditional method. For the remaining 3 points, both methods have similar performance. The lower left plot considers an initial point that has good performance. In this case, 3 of the 5 runs of the traditional method do not identify any points with better performance than that of the initial point. These are displayed as straight horizontal lines. For the proposed method, this occurs for 1 of the 5 runs.

Fig. 5 considers the points proposed by each algorithmic run (i.e., the current iterate upon termination of the algorithm). Let us refer to these points as the "best" points. We run 50 simulation replications of these best points to obtain 50 realizations of their objective function. We do this for each of the 5 algorithmic runs of each method. Hence, for a given initial point and a given method, we obtain 50*5=250 simulation observations of the objective function. Fig. 5 compares the cumulative distribution function (cdf) of these 250 points. Each plot of Fig. 5 considers an initial point. Each plot contains 2 curves: a black solid curve, which represents the proposed method, and a red dashed curve, which represents the traditional method. The curve is the cdf of the 250 simulation observations. The x-axis of the plots is the objective function estimate, and the y-axis is the proportion of observations (out of the 250) that have an objective function smaller or equal to x. Thus, the more a curve is shifted to the left, the better the performance of the corresponding method. As an example of how to interpret these curves, consider the top left plot. For $x = 200$, the y value of the red dashed curve is 0.1. This means that 10% of the 250 observations of the best solutions of the traditional method have objective functions smaller or equal to 200. The corresponding y value of the black solid curve is approximately 0.4, i.e., 40% of the 250 observations of the best solutions of the traditional method have objective functions smaller or equal to 200. For 4 out of the 6 initial points (i.e., initial points 1–4), the proposed method outperforms the traditional method. For initial point 5, the traditional method outperforms the proposed method. For initial point 6, the performance of both methods appears similar. However, the traditional method has a tail of points with objective function values that are sensibly higher than the worst one in the proposed method (see upper part of the diagram).

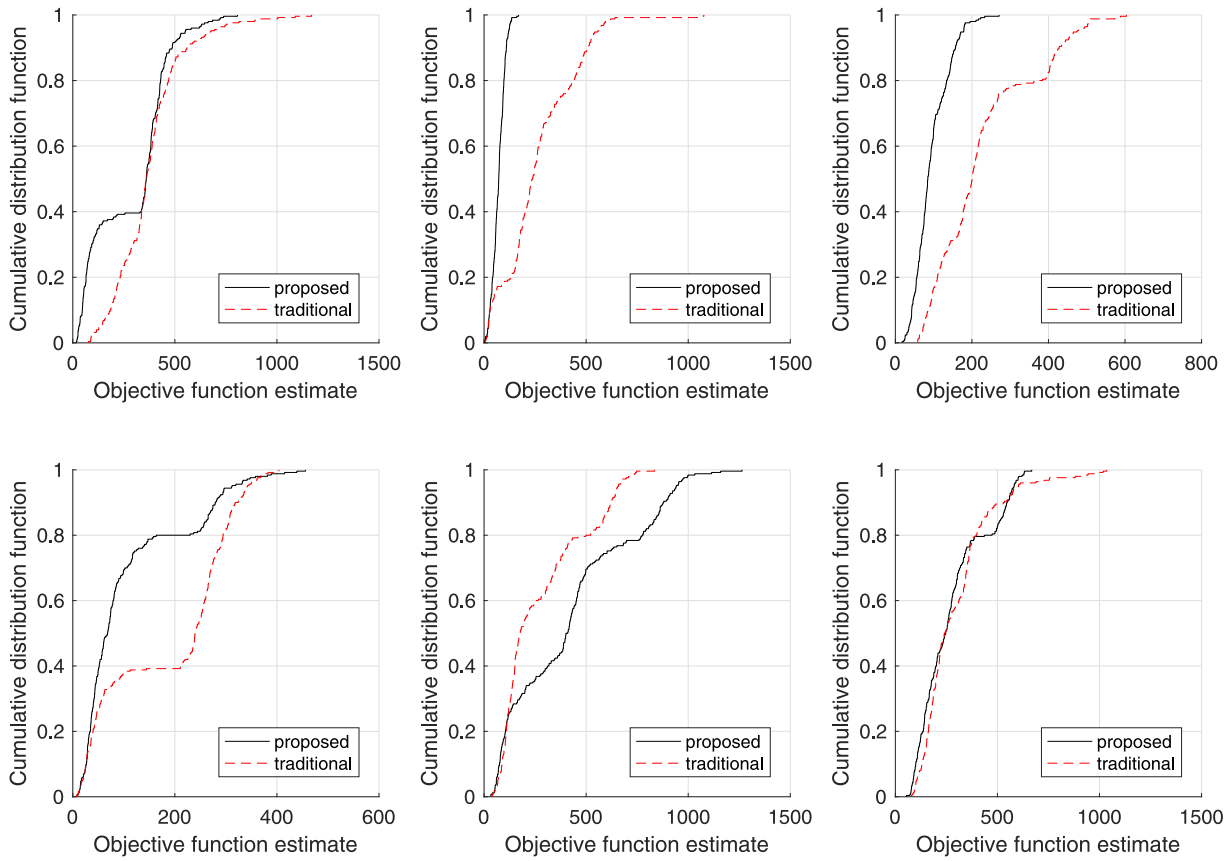

Fig. 5. Aggregate estimated performance of the final solution proposed by each method. Each plot represents one initial point and displays the performance of the proposed and of the traditional methods.

4.2. Lausanne network

We now evaluate the performance of the two methods for a larger network case study of the city of Lausanne. The city map is displayed in Fig. 7, the considered area is delimited in white. We use a microscopic traffic simulation model of the Lausanne city developed by Dumont and Bert (2006). It is implemented with the Aimsun simulator (TSS, 2010), and is calibrated for evening peak period demand. The corresponding network model consists of 603 links and 231 intersections. It is displayed in Fig. 8. We consider an hour of the evening peak-period 5pm-6pm. During this period, congestion gradually increases. The 3 boxplots of Fig. 6 display, respectively, the average link density, the average link delay and the average link queue length. This figure indicates that the links of the network have various levels of congestion, ranging from free flowing to congested. We consider 5 links to have sensors.

We proceed similarly as for the synthetic network. Given that the Lausanne simulation model is computationally more time-consuming to run than that of the synthetic network, we consider a smaller set of 3 random initial points uniformly drawn from the feasible region. For each initial point, we run each method 5 times.

Fig. 9 displays three plots. Each plot considers a given initial point and contains 5 black solid (respectively, red dashed) curves that represent the 5 runs of the proposed (resp. traditional) method. As before, these plots display the objective function estimate of the current iterate as a function of the number of simulated points. For each initial point (i.e., for each plot), all 5 runs of the proposed method outperform all 5 runs of the traditional method. The right-most plot shows a case where the quality of the initial point is relatively good. In this case, for all runs, the traditional method proposes solutions with similar performance than those of the initial point. On the other hand, for 3 of the 5 runs, the proposed method identifies points with significantly better performance.

Table 2 displays for each method and each initial point, the average objective function estimate of the final solution (i.e., current iterate of the last iteration). The average is computed over the 5 algorithm runs. The standard deviation is displayed in parenthesis beside its corresponding average. Initial points 1, and 3 correspond, respectively, to the left-most, center and right-most plots of Fig. 9. For initial points 1 and 2, the proposed method improves, on average, the objective function by 2

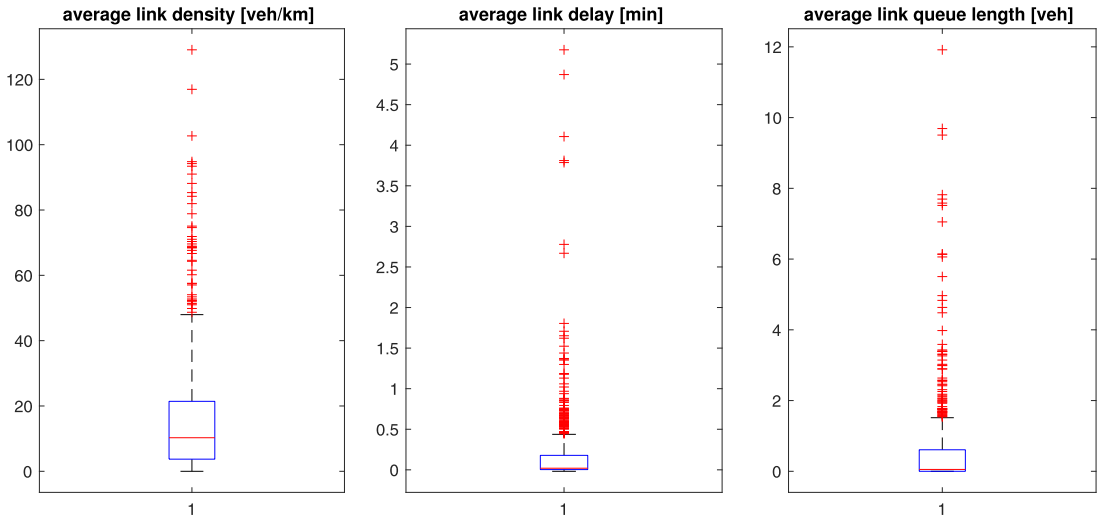

Fig. 6. Link congestion metrics.

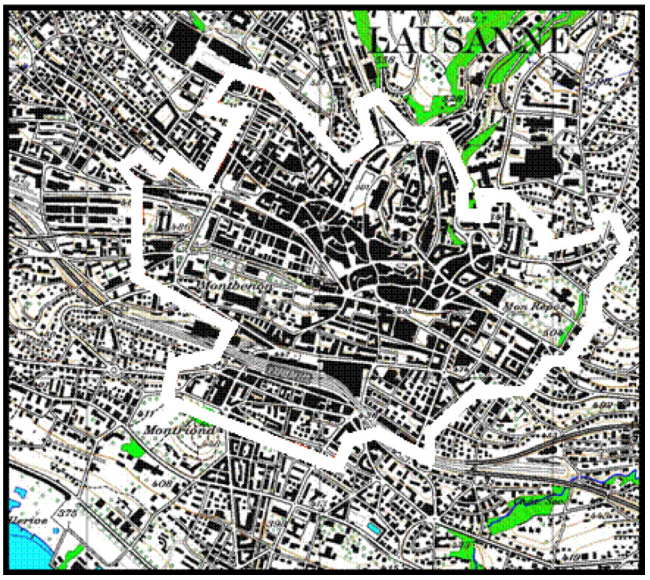

Fig. 7. Lausanne road network (adapted from Dumont and Bert, 2006).

Table 2
Average (over the 5 optimization runs) and standard deviation of the objective function of the final solution .

	Traditional method	Proposed method
Initial point 1	539.5 (292.5)	6.4 (3.4)
Initial point 2	570.5 (100.8)	5.4 (0.4)
Initial point 3	658.1 (41.6)	249.2 (318.3)

orders of magnitude compared to the traditional method. For initial point 3, the improvement in average performance is of 62%.

Similar to the previous figure, Fig. 10 also displays the objective function estimate as a function of the number of simulated points. It displays all the curves of the previous figure in the same plot and focuses on the performance of the last 10 simulated points (i.e., points indexed 40 to 50 in the previous figure). This figure shows that 14 of the 15 runs of the traditional method, yield solutions with objective estimates higher than 400. This is the case for 2 of the 15 runs of the proposed

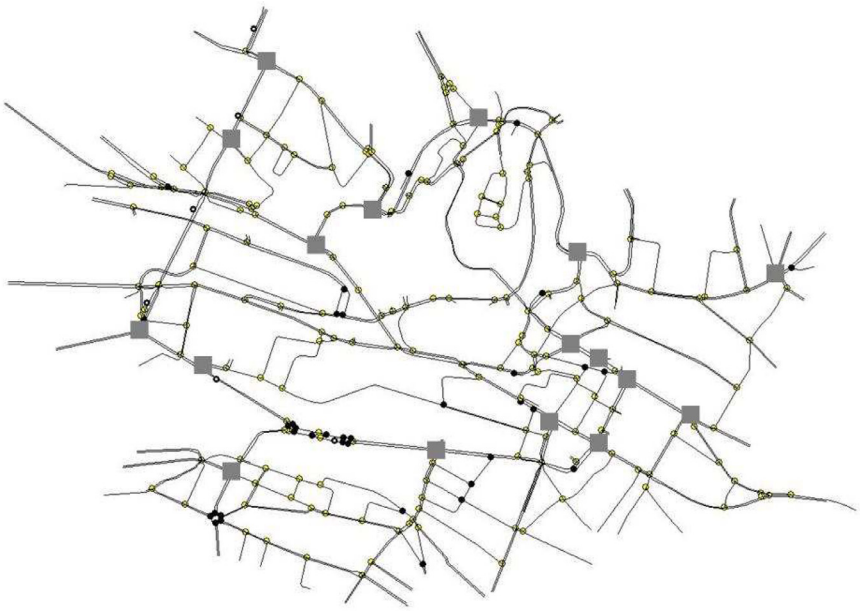

Fig. 8. Lausanne network model.

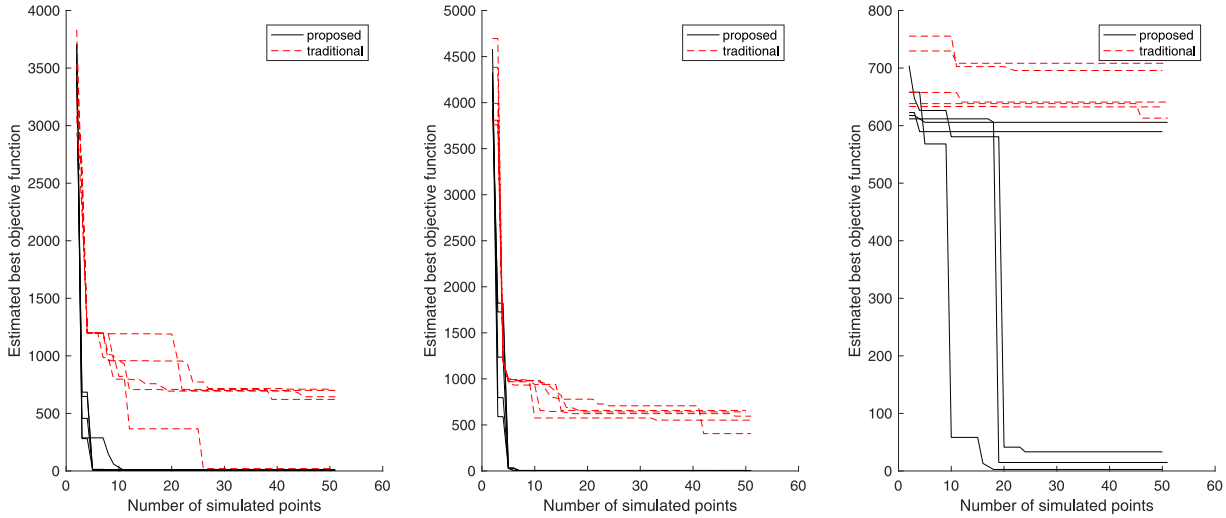

Fig. 9. Estimated performance of the current iterate as a function of the number of simulated points. Each plot represents one initial point and displays the performance of the proposed and of the traditional methods.

method, while the remaining 13 runs yield objective function estimates smaller than 15. This illustrates how, regardless of the initial point, the proposed method tends to outperform the traditional method.

Fig. 11 is similar to Fig. 5. For each initial point and each method, it considers the best points proposed by each of the 5 algorithmic runs, it runs 50 simulation replications for each best point, leading to 250 simulation observations. Each plot displays the cdf of the objective function estimates. The curves of the proposed method are now in blue (rather than black) so that they can be distinguished from the *y*-axis. All plots have the same y-axis range. To improve the legibility of the figure, the axis values along the *y*-axis are not displayed in the middle nor the right-most plots. For the left two plots, the cdf curves of the proposed method are approximately straight vertical lines that overlap with the *y*-axis. This overlap indicates that all solutions derived by the proposed method yield low (i.e., close to zero) objective function values. The verticality of the lines indicates that, for a given initial point, the proposed method yields solutions with similar quality. In other words, its performance is not affected by the stochasticity of the simulator. For the right most plot, the cdf of the proposed method has higher variability, and hence the quality of its best points does vary across runs. For all 3 plots of this figure, the proposed method outperforms the traditional method.

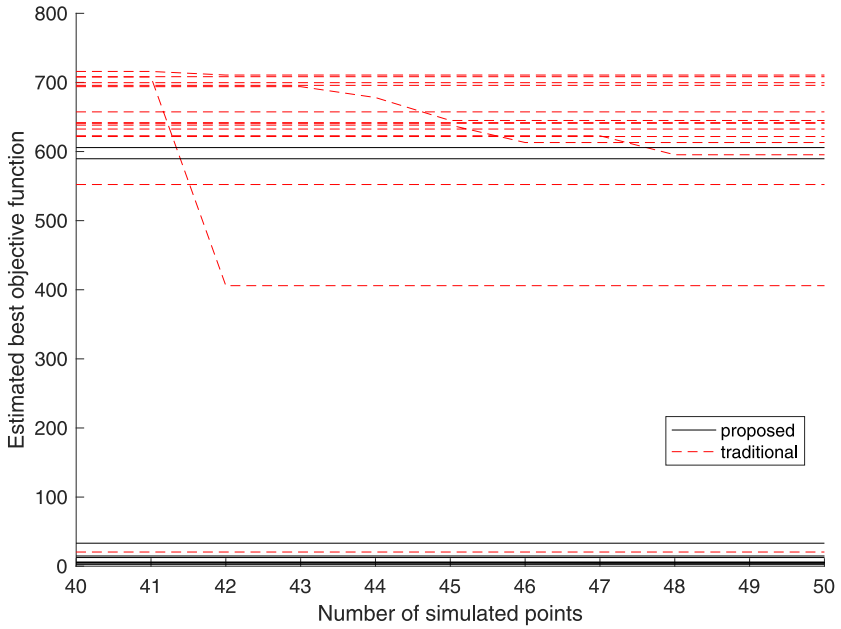

Fig. 10. Estimated performance of the current iterate as a function of the number of simulated points. Each plot represents one initial point and displays the performance of the proposed and of the traditional methods, for the last 10 simulated points.

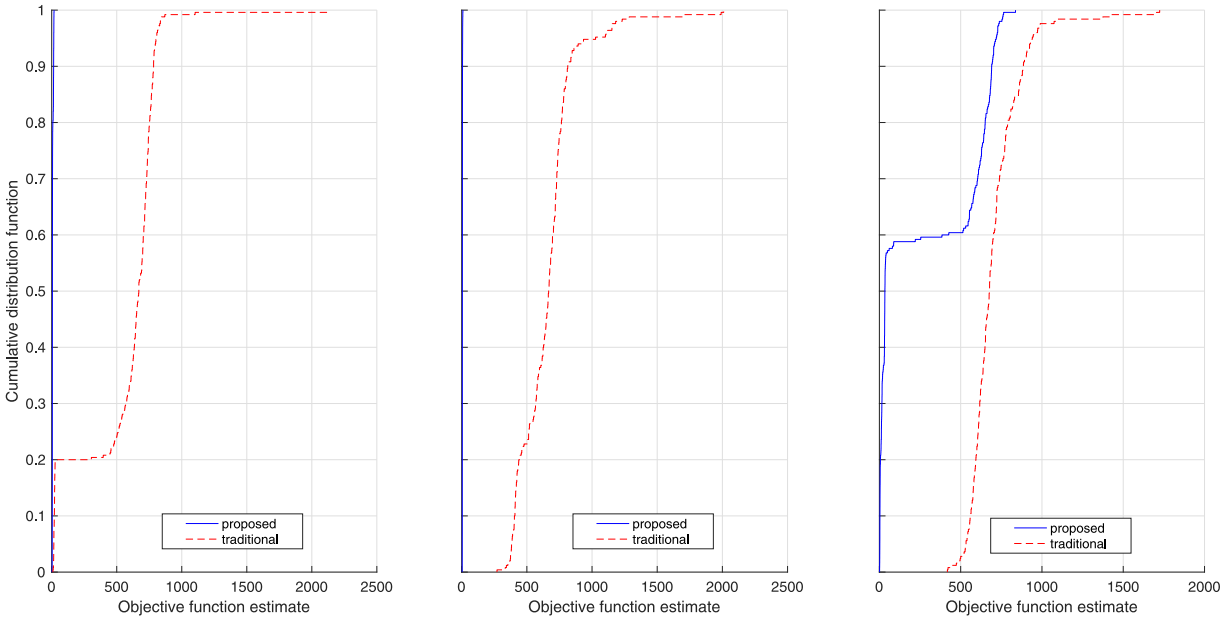

Fig. 11. Aggregate estimated performance of the final solution proposed by each method. Each plot represents one initial point and displays the performance of the proposed and of the traditional methods.

We now compare the performance of the methods in terms of the simulated speed estimates. For each initial point and for each method, we select the point with best performance. The latter is defined as the point with smallest objective function average (the average is obtained over the 50 replications). Fig. 12 displays six plots. The top (resp. bottom) plot considers the proposed (resp. benchmark) method. Each column of plots considers a given initial point. A given plot compares the *true* synthetic speeds (along the x-axis) to the simulated speeds of the best solution (along the y-axis). For a given link with a sensor (i.e., a given x value), we display 50 points along the y-axis, these correspond to the 50 speed observations obtained from the 50 simulation replications. The blue dashed line is the diagonal line. Points closer to this line have better fit. All 6 plots have the same axis limits, so they are directly comparable. To improve the legibility of the figure, the axis values along the y-axis are not displayed in the middle nor the right-most plots. For the proposed method (top plots), the

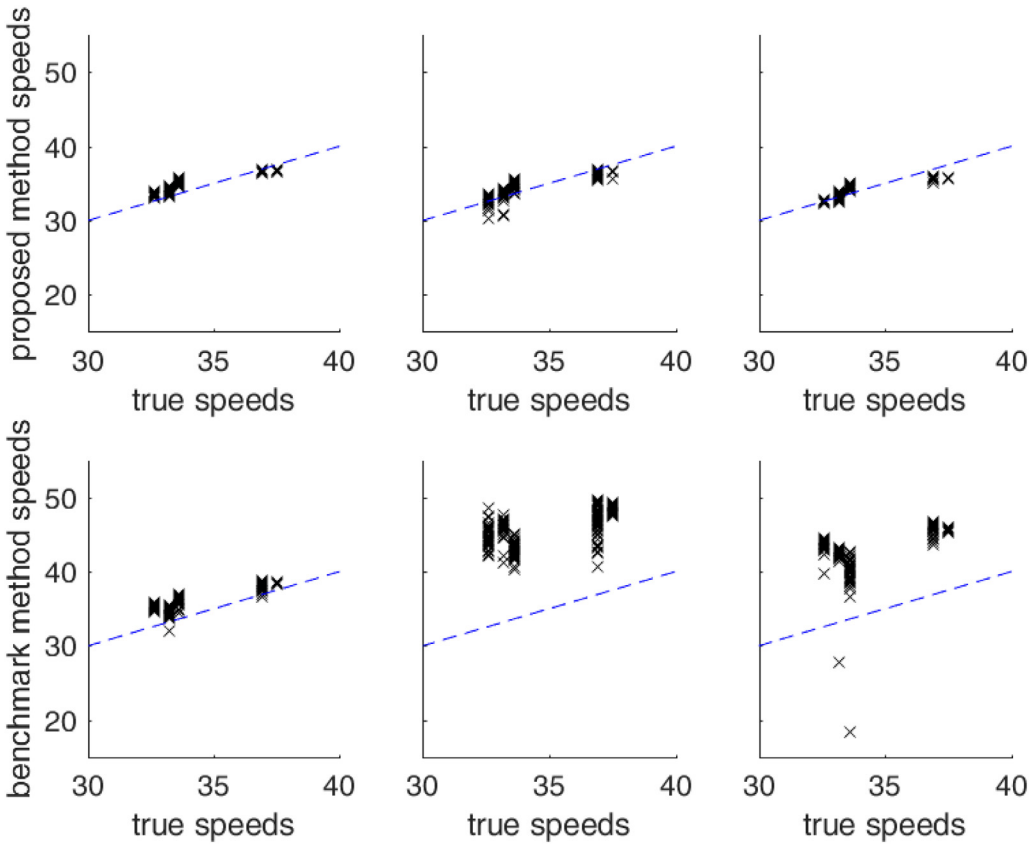

Fig. 12. Comparison of synthetic speeds with speeds of the best solutions. Each column of plots correspond to a given initial point. The top (resp. bottom) row of plots correspond to the proposed (resp. traditional) method.

Table 3
Average and standard deviation, over 50 simulation replications, of the root mean square normalized error (RMSN) statistics for the final solutions proposed by each method .

	Traditional method	Proposed method
Initial point 1	0.06 (0.006)	0.03 (0.004)
Initial point 2	0.32 (0.015)	0.02 (0.006)
Initial point 3	0.26 (0.028)	0.03 (0.002)

best solution leads to speeds that replicate well the synthetic speeds and this for all 3 initial points (i.e., all 3 top plots). For the traditional method, this is the case for 1 of the 3 initial points (the left most plot of the bottom row). For the first initial point (first column of plots), the proposed method and the benchmark method have similar performance. For initial points 2 and 3 (i.e., columns 2 and 3), the proposed method outperforms the benchmark method in terms of speed estimates. This is the case for all 5 links with sensors.

To quantify the quality of the fit of each method, we use the root mean square normalized (RMSN) error of the link speeds. Table 3 displays for each method and each initial point, the average (over the 50 replications) and the standard deviation (over the 50 replications) of the RMSN. The standard deviation appears in parenthesis beside its corresponding average. For initial point 1, the proposed method improves the RMSN by 50%. For initial points 2 and 3, the RMSN is improved by one order of magnitude. On average, over the 3 initial points, the proposed method yields an RMSN of 0.028 compared to an RMSN of 0.21 for the traditional method. It improves, on average, the RMSN by one order of magnitude.

Recall that the main goal of the analytical traffic model formulated in this paper is to accelerate the identification of points with good performance. Fig. 13 evaluates this. It compares the performance of the proposed method (solid black lines) with that of the traditional method (dashed red lines). For each method, there are 15 lines that correspond to the 5 SO runs for each of the 3 initial points. The x-axis displays the SO algorithm iteration. The y-axis displays the Euclidean norm distance between the current iterate (point with best performance) and the true (optimal) solution. Since the experiments use synthetic simulated speed data, we know the value of the calibration parameters that were used to generate these

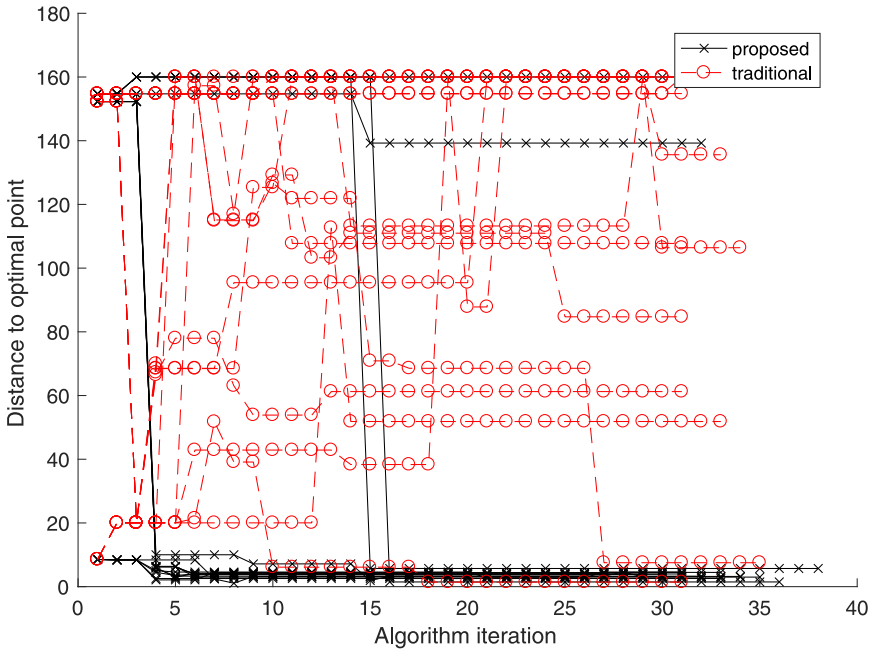

Fig. 13. Distance, for each SO run, of the current iterate to the optimal point across iterations.

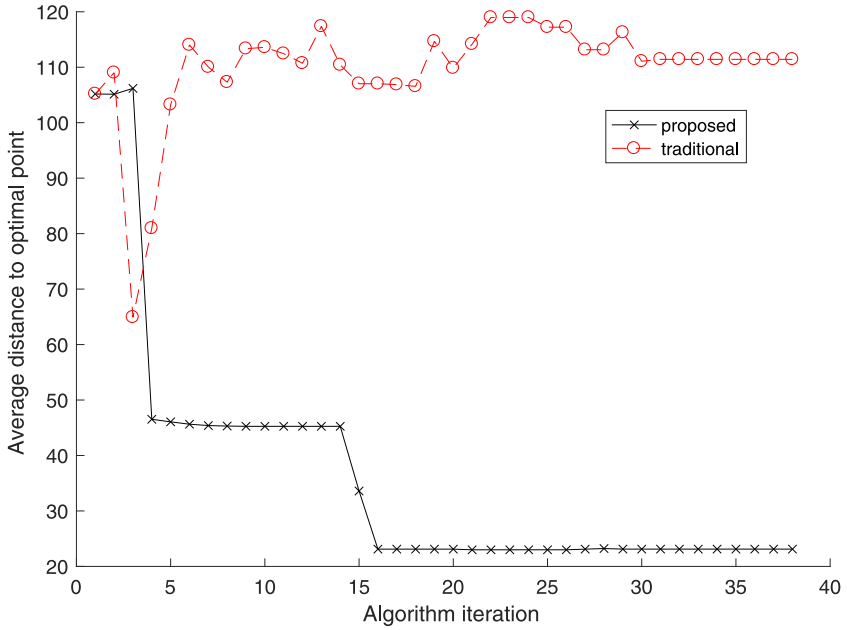

Fig. 14. Average distance of the current iterate to the optimal point across iterations.

synthetic speed measurements. These values are considered to be the true (optimal) solutions to the calibration problem. This figure indicates that for 13 out of the 15 SO runs, the proposed method yields a final solution with a distance of less than 8, while the traditional method achieves this for 2 final solutions. The average (over the 15 SO runs) distance of the final solution is 23.1 for the proposed method and 111.4 for the traditional method. Fig. 14 displays, for each method, the average (over the 15 SO runs) distance of the current iterate to the optimal solution. Figs. 13 and 14 illustrate how the structural information of the analytical traffic model accelerates the ability of the SO algorithm to find solutions close to the optimal solution.

5. Conclusions

This paper considers the calibration of microscopic car-following models embedded within stochastic simulation-based network models. Large-scale microscopic simulators are computationally inefficient to evaluate. Hence, the main contribution of this paper is to propose a computationally efficient calibration technique for large-scale microscopic network models. We propose a metamodel simulation-based optimization (SO) approach. A novel analytical traffic model is formulated, which provides an analytical, scalable and computationally efficient mapping between the calibration vector (i.e., vehicle-specific parameters of the car-following model) and the simulation-based performance measures (expected link speed). The analytical traffic model is embedded within the SO algorithm, as a component of a metamodel.

We carry out case studies on both a synthetic small network and a large-scale real network. We benchmark the approach versus an approach that differs only in that it does not use information from the analytical mapping. The case studies indicate that the problem-specific and simulator-specific structural information that the analytical mapping provides to the SO algorithm enables the algorithm to become both computationally efficient (i.e., solutions with good performance are identified within few simulation evaluations) as well as robust to the quality of the initial points and to the stochasticity of the simulator. Additionally, the proposed approach remains computationally efficient for large-scale networks. Compared to the traditional benchmark approach, the proposed approach improves both the objective function by 2 orders of magnitude and the fit to the field measurements by one order of magnitude, on average.

The joint calibration of supply and demand parameters of the traffic network simulator is of interest. We have recently proposed efficient metamodel approaches for the calibration of origin-destination (OD) matrices (Zhang and Osorio, 2017) and for behavioral parameters of travel demand models (e.g., route choice) (Zhang et al., 2017). These approaches can be readily integrated to propose the joint calibration of these parameters. Moreover, the computational efficiency of the proposed approach makes it a suitable building block for the design of online calibration techniques. The latter are of increasing importance, given the growing interest by both public and private transportation stakeholders in the design and use of real-time traffic models.

Problem formulations that exploit more information of the data are also of interest. For instance, the objective function could account for the temporal variation of the field measurements. This can be done by defining an objective function that minimizes the distance to measurements for shorter time intervals. From a methodological perspective, this can lead to interesting metamodel formulations that capture traffic dynamics. Work along these lines has been recently carried out for dynamic OD calibration problems (Osorio, 2018).

The car-following calibration problem is an intricate optimization problem. A comprehensive discussion is given in Ciuffo et al. (2008) and Punzo et al. (2012). As the intricacy of the problem increases, so does the potential for analytical structural information, such as that from analytical traffic models, to enhance the computational efficiency of calibration algorithms.

Acknowledgments

The work of C. Osorio is partially supported by the U.S. National Science Foundation under Grant No. 1351512. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Appendix A. Gipps Model

Let us briefly recall the formulation of the Gipps model (Gipps, 1981). Let vehicle n denote the vehicle of interest. It is referred to as the follower vehicle. Let vehicle $n-1$ denote the vehicle in front of vehicle n. It is referred to as the leader. The speed of vehicle n is governed by the following equations.

$$v_n(t+\tau) = min\{v_{a,n}(t+\tau), v_{b,n}(t+\tau)\} \tag{8}$$

$$v_{a,n}(t+\tau) = v_n(t) + 2.5a_n\tau\left(1 - \frac{v_n(t)}{v_{\max}}\right)\sqrt{0.025 + \frac{v_n(t)}{v_{\max}}} \tag{9}$$

$$v_{b,n}(t+\tau) = -b_n(\tau/2+\theta) + \sqrt{b_n^2(\tau/2+\theta)^2 + b_n\left(2(h_n(t) - \tilde{S}_n) - \tau v_n(t) + v_{n-1}(t)^2/\hat{b}_{n-1}\right)}, \tag{10}$$

with the following notation:

$v_{a,n}(t+\tau)$	follower's speed at time $(t+\tau)$ in case of free-flowing conditions;
$v_{b,n}(t+\tau)$	follower's speed at time $(t+\tau)$ in case of car-following conditions;
$v_n(t)$	speed of the follower vehicle at time t;
$v_{n-1}(t)$	speed of the leader vehicle at time t;
$h_n(t)$	space headway between the follower and the leader at time t (i.e., space distance between the vehicles' front bumpers);
a_n	follower's maximum desired acceleration;
$v_{\max}$	follower's desired (maximum) speed;
τ	"apparent" reaction time;
θ	"comfort" time lag;
$\tilde{S}_n$	minimum distance to the leader vehicle (i.e., length of vehicle n plus minimum between-vehicle safety distance);
b_n	maximum deceleration of the follower;
$\hat{b}_{n-1}$	follower's estimate of leader's desired deceleration.

In the Gipps model, the speed of a vehicle is limited by one of two factors. The first is the minimal (i.e., safe) distance to the leader. This is represented by Eq. (10) and is referred to as the car-following conditions. The second are the limitations set by the follower vehicle itself. This is represented by Eq. (9) and is referred to as the free-flowing conditions. In a network simulation model, at every simulation time step, the velocity of each vehicle is revised based on a car-following idea along the lines of Eqs. (8)–(10).

Appendix B. SO algorithm for car-following model calibration

The general metamodel SO framework and algorithm is that of Osorio and Bierlaire (2013). The specification below is adapted for its use to calibrate a car-following model. We use the following notation of Osorio and Bierlaire (2013).

k	SO iteration index;
x^k	current iterate at iteration k;
Δ_k	trust region radius at iteration k;
s_k	step size at iteration k;
n_k	total number of simulation runs carried out up until and including iteration k;
μ_k	total number of successive trial points rejected at iteration k;
n_{max}	total number of points to simulate (i.e., computational budget);
$\bar{r}$	number of simulation replications to evaluate per point;

The scalar constants η_1, $\bar{\gamma}$, γ_{inc}, $\bar{\tau}$, $\bar{d}$, $\bar{\mu}$, $\Delta_{\max}$ are defined such that: $0<\eta_1<1$, $0<\bar{\gamma}<1<\gamma_{inc}$, $0<\bar{\tau}<1$, $0<\bar{d}<\Delta_{\max}$, $\bar{\mu}\in\mathbb{N}^*$. Let $f_A(x)$ denote the term within parenthesis of Eq (3). Algorithm 1 specifies the algorithm.

Algorithm 1 relates to the SO framework of Fig. 1 as follows. Step 1 of Fig. 1 refers to accepting or rejecting the current iterate, this step is labeled as lines 10–16 in Algorithm 1. Step 2 of Fig. 1 fits the metamodel by solving Problem (11). Step 3a of Fig. 1 solves the metamodel optimization problem, which corresponds to lines 8–9 of Algorithm 1. Step 3b of Fig. 1 samples points other than the trial point (i.e., the solution to the metamodel optimization problem), it corresponds to lines 18–19 of Algorithm 1. Step 4 of Fig. 1 corresponds to parts of Algorithm 1 that require evaluating the simulator, (i.e., whenever we estimate $E[S_i(\tilde{x})]$ or when we estimate $f(x)$ for a given x).

The traditional method differs from the proposed method as follows: (i) the step "Analytical-only calibration" (lines labeled 3 and 4 of Algorithm 1) is not carried out; (ii) the metamodel does not contain a physical component, i.e., the term $\beta_{k,0}$ of Eq. (3) is set to zero throughout the algorithm; (iii) no computation of f_A is required.

At each iteration k of the SO algorithm, the parameters of the metamodel, β_k, are estimated using simulated observations obtained both at the current iteration as well as at all previous iterations. More specifically, they are determined by solving the following least squares problem.

$$\min_{\beta_k} \sum_{x\in\mathcal{S}_k} \left\{ w_k(x)\left(\hat{E}[S_i(x)] - m_k(x;\beta_k,q)\right)\right\}^2 + w_0^2\left((\beta_{k,0}-1)^2 + \sum_{j=1}^{2D+1}\beta_{k,j}^2\right), \tag{11}$$

where $\hat{E}[S_i(x)]$ denotes the simulation-based estimate of the expected speed on link i for point x, $w_k(x)$ denotes the weight for point x, w_0 denotes an exogenous (fixed) scalar weight coefficient and $\mathcal{S}_k$ denotes the set of points simulated up until iteration k.

The first term of Eq. (11) represents the weighted distance between the speeds predicted by the metamodel and those estimated by the simulator. The weights are defined as in Osorio and Bierlaire (2013):

$$w_k(x) = \frac{1}{1+\left\|x-x^k\right\|_2}, \tag{12}$$

where x^k denotes the current iterate.

Algorithm 1 SO algorithm for the calibration of the Gipps car-following model.

Initialization
Set values for n_{max} and for $\bar{r}$. Set $k = 0$, $n_0 = 0$, $\mu_0 = 0$. Determine the initial point x^0 and the initial trust region radius Δ_0 ($\Delta_0 \in (0, \Delta_{max}]$). Compute $f_A(x^0)$ by solving the System of Equations (5)-(6). Estimate $E[S_i(x^0)]$ (of Eq. (1)) for each link $i \in \mathcal{I}$. Estimate $f(x^0)$ (Eq.(1)). Include the new simulation observation in the set of sampled points, i.e., set $n_0 = n_0 + \bar{r}$.
Analytical-only calibration
Solve Problem (3)-(6)using only the analytical traffic model and without using any simulation information. This is done by setting: $\beta_{k,0}$ of Eq.(3) to 1 and by setting $\beta_{k,j} = 0 \ \forall j > 0$ (also of Eq.(3)) to 0. Let $\tilde{x}$ denote the solution to this problem. Estimate $E[S_i(\tilde{x})]$ for each link $i \in \mathcal{I}$ and estimate $f(\tilde{x})$ to yield $\hat{f}(\tilde{x})$. Set $n_0 = n_0 + \bar{r}$. Set the initial current iterate to this solution, i.e., set $x^0 = \tilde{x}$.
Initial metamodel fitting
Solve Problem (11) to fit the metamodel m_0.
while $n_k < n_{max}$ **do**
  Step calculation
  Solve Problem (3)-(6) subject to a trust-region constraint (i.e., $\|s_k\| \leq \Delta_k$), let $x^k + s_k$ denote the solution, which is referred to as the trial point.
  Acceptance or rejection of the trial point
  Compute $f_A(x^k + s_k)$ by solving the System of Equations (5)-(6). Estimate $E[S_i(x^k + s_k)]$ for each link $i \in \mathcal{I}$. Estimate $f(x^k + s_k)$. Set $n_k = n_k + \bar{r}$. Compute:

$$\rho_k = \frac{\hat{f}(x^k) - \hat{f}(x^k + s_k)}{m_k(x^k) - m_k(x^k + s_k)}.$$

  if $\rho_k \geq \eta_1$ and $\hat{f}(x^k) - \hat{f}(x^k + s_k) > 0$ **then**
    Accept the trial point: $x^{k+1} = x^k + s_k$, $\mu_k = 0$;
  else
    Reject the trial point: $x^{k+1} = x^k$, $\mu_k = \mu_k + 1$.
  end if
  Solve Problem (11) to fit the metamodel m_{k+1}.
  Model improvement
  Compute $\tau_{k+1} = \frac{\|\beta_{k+1} - \beta_k\|}{\|\beta_k\|}$. If $\tau_{k+1} < \bar{\tau}$, then sample a new point x, which is uniformly and randomly drawn from the feasible region defined by Eq.(2). Evaluate $f_A(x)$ by solving the System of Equations (5)-(6). Estimate $E[S_i(x)]$ and $f(x)$. Set $n_k = n_k + \bar{r}$. Solve Problem (11) to update the fit of the metamodel m_{k+1}.
  Trust region radius update

$$\Delta_{k+1} = \begin{cases} \min\{\gamma_{inc}\Delta_k, \Delta_{max}\}, & \text{if } \rho_k > \eta_1 \\ \max\{\bar{\gamma}\Delta_k, \bar{d}\}, & \text{if } \rho_k \leq \eta_1 \text{ and } \mu_k \geq \bar{\mu} \\ \Delta_k, & \text{otherwise.} \end{cases}$$

  if $\rho \leq \eta_1$ and $\mu_k \geq \bar{\mu}$ **then**
    Set $\mu_k = 0$. Set $n_{k+1} = n_k$, $\mu_{k+1} = \mu_k$, $k = k + 1$.
  end if
end while

References

Azevedo, C.L., Ciuffo, B., Cardoso, J.L., Ben-Akiva, M.E., 2015. Dealing with uncertainty in detailed calibration of traffic simulation models for safety assessment. Transp. Res. Part C 58, 395–412.

Balakrishna, R., 2006. Off-Line Calibration of Dynamic Traffic Assignment Models. Massachusetts Institute of Technology.

Balakrishna, R., Ben-Akiva, M.E., Koutsopoulos, H.N., 2007. Offline calibration of dynamic traffic assignment: simultaneous demand-and-supply estimation. Transp. Res. Rec. 2003, 50–58.

Barceló, J., 2010. Fundamentals of Traffic Simulation. International Series in Operations Research and Management Science, vol. 145. Springer, New York, USA.

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

Cascetta, E., Nguyen, S., 1988. A unified framework for estimating or updating origin/destination matrices from traffic counts. Transp. Res. Part C Part B 22 (6), 437–455.

Chong, L., Osorio, C., 2018. A simulation-based optimization algorithm for dynamic large-scale urban transportation problems. Transp. Sci 52 (3), 637–656.

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

Ciuffo, B., Azevedo, C.L., 2014. A sensitivity-analysis-based approach for the calibration of traffic simulation models. IEEE Trans. Intell. Transp. Syst. 15 (3), 1298–1309.

Ciuffo, B., Casas, J., Montanino, M., Perarnau, J., Punzo, V., 2013. Gaussian process metamodels for sensitivity analysis of traffic simulation models. Transp. Res. Rec. 2390, 87–98.

Ciuffo, B., Punzo, V., Torrieri, V., 2008. Comparison between simulation-based and model-based calibrations of traffic flow micro-simulation models. Transp. Res. Rec. 2088, 36–44.
Dumont, A.G., Bert, E., 2006. Simulation De L'agglomération Lausannoise SIMLO. Technical Report. Laboratoire des voies de circulation, ENAC, Ecole Polytechnique Fédérale de Lausanne.
Ge, Q., Ciuffo, B., Menendez, M., 2014. An exploratory study of two efficient approaches for the sensitivity analysis of computationally expensive traffic simulation models. IEEE Trans. Intell. Transp. Syst. 15 (3), 1288–1297.
Gipps, P.G., 1981. A behavioural car-following model for computer simulation. Transp. Res. 15B (2), 105–111.
Hall, F., 2001. Traffic stream characteristics. In: Garthner, N., Messer, C., Rathi, A. (Eds.), Revised Traffic Flow Theory Monograph. Transportation Research Board.
Huang, E., 2010. Algorithmic and Implementation Aspects of On-Line Calibration of Dynamic Traffic Assignment. Massachusetts Institute of Technology.
Kattan, L., Abdulhai, B., 2006. Noniterative approach to dynamic traffic origin-destination estimation with parallel evolutionary algorithms. Transp. Res. Rec. 1964, 201–210.
Kunde, K.K., 2002. Calibration of Mesoscopic Traffic Simulation Models for Dynamic Traffic Assignment. Massachusetts Institute of Technology, Cambridge, MA, USA.
Lu, L., Xu, Y., Antoniou, C., Ben-Akiva, M., 2015. An enhanced SPSA algorithm for the calibration of dynamic traffic assignment models. Transp. Res. Part C 51, 149–166.
Osorio, C., 2017. High-Dimensional Offline OD Calibration for Stochastic Traffic Simulators of Large-Scale Urban Networks. Technical Report. Massachusetts Institute of Technology. Under review. Available at: http://web.mit.edu/osorioc/www/papers/osoODCalib.pdf.
Osorio, C., 2018. Dynamic OD Calibration for Large-Scale Network Simulators. Transp. Res. Part C forthcoming.
Osorio, C., Atastoy, B., 2017. Efficient Simulation-Based Toll Optimization for Large-Scale Networks. Technical Report. Massachusetts Institute of Technology. Under review. Available at: http://web.mit.edu/osorioc/www/papers/osoAtaTollSO.pdf.
Osorio, C., Bierlaire, M., 2013. A simulation-based optimization framework for urban transportation problems. Oper. Res. 61 (6), 1333–1345.
Osorio, C., Nanduri, K., 2015. Urban transportation emissions mitigation: coupling high-resolution vehicular emissions and traffic models for traffic signal optimization. Transp. Res. Part C Part B 81, 520–538.
Osorio, C., Yamani, J., 2017. Analytical and scalable analysis of transient tandem Markovian finite capacity queueing networks. Transp. Sci. 51 (3), 823–840.
Punzo, V., Ciuffo, B., Montanino, M., 2012. Can results of car-following model calibration based on trajectory data be trusted? Transp. Res. Rec. 2315, 11–24.
Punzo, V., Montanino, M., Ciuffo, B., 2015. Do we really need to calibrate all the parameters? Variance-based sensitivity analysis to simplify microscopic traffic flow models. IEEE Trans. Intell. Transp. Syst. 16 (1), 184–193.
Punzo, V., Tripodi, A., 2007. Steady-state solutions and multiclass calibration of Gipps microscopic traffic flow model. Transp. Res. Rec. 1999, 104–114.
Søndergaard, J., 2003. Optimization Using Surrogate Models - by the Space Mapping technique. Technical University of Denmark.
Spall, J.C., 2003. Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control. John Wiley & Sons, New Jersey, USA.
Stathopoulos, A., Tsekeris, T., 2004. Hybrid meta-heuristic algorithm for the simultaneous optimization of the O-D trip matrix estimation. Comput.-Aided Civ. Infrastruct. Eng. 19 (6), 421–435.
Treiber, M., Hennecke, A., Helbing, D., 2000. Congested traffic states in empirical observations and microscopic simulations. Phys. Rev. E 62 (2), 1805–1824.
TSS, 2010. Microsimulator and Mesosimulator. AIMSUN 6.1 Users Manual. Transport Simulation Systems.
Tympakianaki, A., Koutsopoulos, H., Jenelius, E., 2015. C-SPSA: cluster-wise simultaneous perturbation stochastic approximation algorithm and its application to dynamic origin-destination matrix estimation. Transp. Res. Part C 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. Transp. Res. Rec. 2090, 1–9.
Wilson, R.E., 2001. An analysis of Gipps car-following model of highway traffic. IMA J. Appl. Math. 66, 509–537.
Zhang, C., Osorio, C., 2017. Efficient Offline Calibration of Origin-Destination (Demand) for Large-Scale Stochastic Traffic Models. Technical Report. Massachusetts Institute of Technology. Under review. Available at: http://web.mit.edu/osorioc/www/papers/zhaOsoODcalib.pdf.
Zhang, C., Osorio, C., Flötteröd, G., 2017. Efficient calibration techniques for large-scale traffic simulators. Transp. Res. Part C Part B 97, 214–239.
Zhong, R.X., Fu, K.Y., Sumalee, A., Ngoduy, D., Lam, W.H.K., 2016. A cross-entropy method and probabilistic sensitivity analysis framework for calibrating microscopic traffic models. Transp. Res. Part C 63, 147–169.