Transportation Research Part C 68 (2016) 311–332

Contents lists available at ScienceDirect

Transportation Research Part C

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

A global optimization algorithm for trajectory data based car-following model calibration

Li Li [a], Xiqun (Micheal) Chen [b,*], Lei Zhang [c]

[a] Department of Automation, Tsinghua University, Beijing 100084, China
[b] College of Civil Engineering and Architecture, Zhejiang University, Hangzhou 310058, China
[c] Department of Civil and Environmental Engineering, University of Maryland, College Park, MD 20742, United States

ARTICLE INFO

Article history:
Received 28 July 2015
Received in revised form 7 April 2016
Accepted 12 April 2016
Available online 29 April 2016

Keywords:
Microscopic car-following model
Calibration
Lipschitz optimization
Global optimization

ABSTRACT

How to calibrate the parameters of car-following models based on observed traffic data is a vital problem in traffic simulation. Usually, the core of calibration is cast into an optimization problem, in which the decision variables are car-following model parameters and the objective function usually characterizes the difference between empirical vehicle movements and their simulated correspondences. Since the objective function is usually nonlinear and non-convex, various greedy or stochastic algorithms had been proposed during the last two decades. However, the performance of these algorithms remains to be further examined. In this paper, we revisit this important problem with a special focus on the geometric feature of the objective function. First, we prove that, from a global perspective, most existing objective functions are Lipschitz continuous. Second, we show that, from a local perspective, many of these objective functions are relatively flat around the global optimal solution. Based on these two features, we propose a new global optimization algorithm that integrates global direct search and local gradient search to find the optimal solution in an efficient manner. We compare this new algorithm with several existing algorithms, including Nelder–Mead (NM) algorithm, sequential quadratic programming (SQP) algorithm, genetic algorithm (GA), and simultaneous perturbation stochastic approximation (SPSA) algorithm, on NGSIM trajectory datasets. Results demonstrate that the proposed algorithm has a fast convergence speed and a high probability of finding the global optimal solution. Moreover, it has only two major configuration parameters that can be easily determined in practice.

1. Introduction

Microscopic simulation plays an important role in the analysis and design of traffic facilities. It provides a flexible platform on which various traffic scenarios can be studied in a controlled maneuver without disrupting real-world traffic (Chung and Dumont, 2009; Barceló, 2010). Usually, a traffic simulation platform consists of several models which address different aspects of traffic behavior. In this paper, we will focus on longitudinal car-following models (Punzo and Simonelli, 2005; Gunay, 2007; Punzo and Tripodi, 2007; Kesting and Treiber, 2008; Ossen and Hoogendoorn, 2008; Tordeux et al., 2010; Chen et al., 2012, 2015; Ciuffo et al., 2012a,b).

* Corresponding author at: B828 Anzhong Building, College of Civil Engineering and Architecture, Zhejiang University, 866 Yuhangtang Rd, Hangzhou 310058, China.
E-mail address: chenxiqun@zju.edu.cn (X. (Micheal) Chen).

http://dx.doi.org/10.1016/j.trc.2016.04.011
0968-090X/

Most microscopic simulation tools adopt certain ordinary difference equations characterized by a few parameters to model drivers' car-following actions. In order to guarantee the selected simulation models well reproduce traffic phenomena observed in practice, different calibration methods have been proposed to appropriately determine the model parameters (Wu et al., 2003; Ahn et al., 2004; Brockfeld et al., 2005; Panwai and Dia, 2005; Punzo and Simonelli, 2005; Kesting and Treiber, 2008; Ossen and Hoogendoorn, 2008; Punzo et al., 2012).

In some microscopic car-following models, a few parameters have physical equivalents in reality, e.g., desired velocity used in Gipps' car-following model (Gipps, 1981; Wilson, 2001; Punzo and Tripodi, 2007; Ciuffo et al., 2012a,b). Other macroscopic features of traffic flow parameters, e.g., free flow velocity, critical density and jam density, are important in many models and can be straightforwardly estimated from loop detector data (Rakha et al., 2007; Rakha and Gao, 2011). However, in lots of car-following models, more parameters cannot be simply derived from macroscopic measurements (Zhang and Kim, 2005; Punzo and Simonelli, 2005; Kesting and Treiber, 2008; Ossen and Hoogendoorn, 2008; Ciuffo and Punzo, 2014).

As a result, the car-following model calibration using empirical vehicle trajectory data gained increasing interests recently, since trajectory data provided much more details of driving actions on the microscopic level (Punzo and Simonelli, 2005; Kesting and Treiber, 2008; Ossen and Hoogendoorn, 2009; Chiabaut et al., 2010; Punzo et al., 2012). There was an approach used for the first time in car-following calibration by Ossen and Hoogendoorn (2008) and successively by Punzo et al. (2012), making use of laboratory experiments to compare algorithms in a fair way, that is using the model itself to generate synthetic data on which performing calibration. This allows the analysist to know what the global optimum is. How to accurately estimate model parameters from trajectory then becomes an issue of importance; since none of existing car-following models can perfectly fit all of the empirical trajectories, due to various influence factors such as time-varying dynamics of driving actions (Wagner, 2012; Koutsopoulos and Farah, 2012), heterogeneity of different drivers (Ossen and Hoogendoorn, 2007, 2011; Wang et al., 2010), measurement noises (Kesting and Treiber, 2008; Ossen and Hoogendoorn, 2008) and asymmetric characteristics in car-following and their impacts on traffic flow evolution (Li et al., 2013a; Wei and Liu, 2013).

Recently, Punzo et al. (2012) made a deep investigation about methodological aspects in the calibration of car-following models, including a comparison of optimization algorithms. Further, different indicators were proposed to measure algorithm performance not only in terms of the objective function value but also in terms of the distance of calibrated parameters from the global ones. In that research, the optimization performance indicator was proposed to provide a measure of the accuracy of the best solution in a calibration experiment in terms of both the parameter values and the score of the objective function. In this paper, we follow the same measures of performance (time series of the follower's speed and spacing between leader and follower) and adopt one of the goodness-of-fit functions (i.e., root mean square error) used in Punzo et al. (2012), since the normalized optimization performance indicator cannot be achieved due to unknown true global optimum. The main methodological contribution of this study is to propose a combined two-stage optimization algorithm for trajectory data based car-following calibration.

In many solutions for the car-following model calibration, parameters were taken as constant and uncertainties were viewed as an additive disturbance to system dynamics. The core of calibration can thus be casted into an optimization problem, in which decision variables are model parameters and the objective function usually characterizes the difference between empirical vehicle movements and their simulated correspondences.

In the simulation, usually two consecutive vehicles are considered at one time. The state of the leading vehicle is updated according to empirical observations, and the state of the following vehicle is updated via the selected car-following model. According to simulation settings, we can further categorize the existing calibration approaches into two types (Wagner et al., 2010; Treiber and Kesting, 2013a).

The first category is called *local-fit* or direct-fit, the endogenous model variables are compared against the data, separately for each data point. Specifically, the empirical position and velocity information of both vehicles are used as input at each simulation time step. Consequently, the outputs of the car-following model at the next time steps can be calculated. The modeled accelerations are compared against the empirical ones, separately for each time step, to obtain the samples of additive disturbance. If the distribution of additive disturbances is further assumed, we can directly fit the car-following model with respect to these sampled disturbances. For example, the maximum likelihood estimation (MLE) was applied to maximize the "agreement" of model parameters with the observed data (Hoogendoorn and Ossen, 2005; Hoogendoorn and Hoogendoorn, 2010; Kim et al., 2013). Since the induced variables (usually the accelerations at different time steps) are separable in the objective function, the optimization problems of this category are usually easy to solve.

The second category is called *global-fit* or indirect-fit, a complete data trajectory is compared with a simulated trajectory. The car-following model is not directly calibrated by independently comparing the endogenous model variables with the observations. Instead, only the empirical position and velocity information of the leading vehicle are used as input at each time step. The initial state of the following vehicle is given, too. Then, the movements of the following vehicle at the rest of time steps are calculated sequentially. Finally, the simulated trajectory is compared with the empirical one. Since the induced variables (usually the velocities or positions at different time steps) are determined in a sequential manner, the objective functions of this category are usually nonlinear, non-convex and difficult to optimize.

It was demonstrated in Punzo and Simonelli (2005), Wagner et al. (2010), and Treiber and Kesting (2013a) that these two kinds of approaches might generate notably different choices of parameter values, although the same trajectory dataset was

used. In some benchmark datasets, global-fit approaches yield better performance, partly because the memory mechanism of the global-fit approaches adds additional bounded constraints to some parameters (especially ac/deceleration parameters) and prevents them to be too large or too small.

So, how to solve the optimization problem for the global-fit approaches received consistent interests (e.g., Brockfeld et al., 2003). In this paper, we will focus on this important problem.

A number of greedy search algorithms were used to find acceptable solutions. Brockfeld et al. (2005) applied the Nelder–Mead (NM) algorithm (Nelder and Mead, 1965; Lagarias et al., 1998) to solve the formulated optimization problem. They found that the NM algorithm was easy to get stuck at local minima, because of the non-convexity of the objective function. So, the authors suggested to restart the algorithm many times (at least three runs and usually more) with different initialization values of decision variables. Similarly, Ossen and Hoogendoorn (2008) repeated the whole calibration procedure several times and used different measures of performance as the objective every time. Alternatively, the sequential quadratic programming (SQP) algorithm was adopted in Wang et al. (2010). However, SQP algorithm did not guarantee to converge to global minima, either.

Several stochastic algorithms had been applied for this problem, too. For example, genetic algorithm (GA) was used to calibrate microscopic simulation models in Ma and Abdulhai (2002), Schultz and Rilett (2004), and Park and Qi (2005). Simultaneous perturbation stochastic approximation (SPSA) algorithm was adopted to find the best parameters in Ma et al. (2007) and Lee and Ozbay (2009). In the literature, a few global–local optimization studies were conducted in car-following or traffic microsimulation model calibration (Ciuffo and Punzo, 2010; Punzo et al., 2011; Ciuffo et al., 2012a,b). Punzo et al. (2012) successfully applied the OptQuest multistart algorithm that combined the seeking behavior of a gradient-based local nonlinear programming solver with the global optimization abilities of a scatter search. Since candidate starting points are generated either by a scatter search heuristic or by a randomized process, it would be difficult to select starting points and control the transition from the global stage to local stage. The computational complexity of these stochastic algorithms still needs further discussions.

This paper revisits the car-following model calibration problem from a different viewpoint. First, we prove that, from a global perspective, most existing objective functions are Lipschitz continuous. Moreover, we show that, from a local perspective, many of these objective functions are relatively flat around the global optimal solution. Based on these two features, we propose a new global optimization algorithm that integrates global direct search and local gradient search to find the optimal solution. The proposed combined algorithm is based on detailed analysis of features of the objective functions used in car-following calibration. We have solved a critical problem that exists in most combined algorithms, that is, how to determine the transition condition between global search and local search. In the first stage, the new algorithm applies the Lipschitz search algorithm (Jones et al., 1993) to determine a few candidate subspace that most likely contains the global optimal solution. Then in the second stage, based on those potentially optimal subspace, either SQP or the trust region (Fletcher et al., 2002) algorithm can be applied to search the global optimum with a fast speed. Besides, there is another important reason to propose the Lipschitz optimization algorithm with local search, i.e. easily implemented configuration parameters. Only two parameters with physical meanings for determining convergence termination need pre-selection, i.e. critical normalized size of global–local transition and number of multi-start potentially optimal subspace. The explicit feature makes the combined algorithm probably be widely used in different car-following calibration problems. Results demonstrate that the proposed algorithm has a fast convergence speed and a high probability of finding the global optimal solution.

To better present our findings, the rest of this paper is arranged as follows. Section 2 briefly reviews several existing direct search algorithms for parameter calibration, and their performance will be compared in Section 5.2. Then Section 3 presents the car-following model calibration problem and then discusses the Lipschitz continuous property of the objective functions. Section 4 presents the new algorithm that utilizes the Lipschitz continuous property to accelerate searching. Numerical tests on field trajectory sets are provided in Section 5 to compare the new algorithm with some existing algorithms and demonstrate its efficiency. Finally, Section 6 concludes the paper.

2. Several existing algorithms for parameter calibration

Among numerous algorithms proposed for car-following calibrations, we will revisit four representative ones here: Nelder–Mead (NM) algorithm, sequential quadratic programming (SQP) algorithm, genetic algorithm (GA), and simultaneous perturbation stochastic approximation (SPSA) algorithm:

(1) In Brockfeld et al. (2005), the Nelder–Mead algorithm (Nelder and Mead, 1965; Lagarias et al., 1998) was used to find an optimal solution for the calibration problem. The NM algorithm starts with an initial set of points that form a simplex (a convex hull of $(n+1)$ vertices in n dimensions with nonzero volume). In each iteration, we compare the objective function values at the vertices of this simplex and determine the worst corner point. Then, we attempt to replace the worst point by introducing a new vertex in a pre-selected way and form a new simplex. The iteration terminates if the objective function values at the vertices of the current simplex satisfy a certain form of the descent condition compared to those for the previous simplex.

The NM algorithm is a deterministic local direct search algorithm, because it transforms the worst vertex through a number of operations about the centroid of the current simplex and gradually expands the searching set. It may be trapped into a

local optimal solution. The NM simplex method is an efficient local search procedure but its convergence is extremely sensitive to the selected starting point. Moreover, the original NM algorithm is a direct search method for multidimensional unconstrained optimization but is not designed for constrained problems.

(2) The SQP algorithm is a frequently-used algorithm for constrained optimization problems (Nocedal and Wright, 2006). In each iteration of the SQP algorithm (Fletcher and Powell, 1963; Fletcher et al., 2002), it first approximates the original optimization problem as a quadratic objective subject to a linearization of the constraints. Then, we apply the Newton's method to find an extreme value point where the gradient of the objective vanishes. Sequentially apply this approximation-maximization iterations, we may finally reach a local optimal solution.

The SQP algorithm is a deterministic local direct search algorithm and can be applied to constrained problems. However, it is very sensitive to the initial vector and may be trapped into a local optimal solution, too.

(3) The GA algorithm (Goldberg, 1989) is a stochastic optimization algorithm directly inspired from the Darwinian theory of evolution of species. It mimics the process of natural selection to routinely generate new vectors. It is able to find the global optimal solution, if the running time approaches infinity. However, the performance of GA algorithm for a particular problem still needs careful examinations, especially when we preset a limit on computation time.

(4) The SPSA algorithm (Spall, 2000, 2003) is another stochastic optimization algorithm. It requires only two measurements of the objective and one random simultaneous perturbation to estimate the gradient, regardless of the dimension of the optimization problem. Since the SPSA algorithm introduces random perturbations when estimating the gradient, the converging speed of the SPSA algorithm often requires careful tests.

Nevertheless, the GA and SPSA algorithms have a number of configuration parameters for tuning. Selecting a set of appropriate configuration parameters are often tricky, tedious, and time consuming. The implementation of GA and SPSA may need to specify a number of algorithm coefficients. It is often difficult to find the right parameters for the algorithms themselves. In practice, some "trial and error" experimentation will be required for an effective selection of algorithm coefficients before implementation. Of course, they may have good results if selecting an appropriate set of coefficients. Constrained by the paper length limit, we will not further discuss the variations of these algorithms. Instead, the performance of these four algorithms in their standard forms will be compared in the following Section 5.

3. Global-fit model calibration

3.1. The general car-following model and calibration

Generally, a *continuous-time* car-following model can be described by a set of coupled delay-differential equations of the form

$$\begin{cases} \dot{v}_i(t|\boldsymbol{\theta}_i) = \frac{\mathrm{d}v_i(t|\boldsymbol{\theta}_i)}{\mathrm{d}t} = a_i(t|\boldsymbol{\theta}_i) = f\left(\Delta x_{i-1,i}(t-T_r), v_i(t-T_r), \Delta v_{i-1,i}(t-T_r)|\boldsymbol{\theta}_i\right) \\ \dot{x}_i(t|\boldsymbol{\theta}_i) = \frac{\mathrm{d}x_i(t|\boldsymbol{\theta}_i)}{\mathrm{d}t} = v_i(t|\boldsymbol{\theta}_i) \end{cases} \tag{1}$$

where $v_i(t|\boldsymbol{\theta}_i)$ denotes the velocity of the object vehicle i at time t; $x_i(t|\boldsymbol{\theta}_i)$ represents the position of the ith vehicle at time t; $\Delta x_{i-1,i}(t) = x_{i-1}(t) - x_i(t)$ represents the spacing between the leading $(i-1)$th vehicle and the following ith vehicle at time t. Similarly, $\Delta v_{i-1,i}(t)$ denotes the velocity difference between two consecutive vehicles. $f(\cdot)$ is a special function that describes the car-following behavior and $\boldsymbol{\theta}_i$ are a set of parameters for the driver of the ith vehicle if making an assumption of heterogeneous driving behavior. the driver's response is directly given in terms of an acceleration function or the car-following law $f(\cdot)$. $a_i(t|\boldsymbol{\theta}_i)$ is the acceleration rate generated by at time t given the car-following parameter set $\boldsymbol{\theta}_i$, the subscript i denotes the object vehicle. T_r is the reaction time that is composed of the mental processing time, the movement or action time, and the technical response time. If $T_r = 0$, Eq. (1) is a set of continuous-time differential equations; otherwise, it is delay-differential equations.

The acceleration equation in Eq. (1) cannot be integrated analytically, but it is straightforward to approximatively solve it numerically. In traffic simulation, explicit update schemes with a fixed update time interval are practical. A *discrete-time* car-following model requires less computation resources for its numerical integration.

$$\begin{cases} a_i(t|\boldsymbol{\theta}) = f\left(\Delta x_{i-1,i}(t-T_r), v_i(t-T_r), \Delta v_{i-1,i}(t-T_r)|\boldsymbol{\theta}\right) \\ v_i(t+T|\boldsymbol{\theta}) = v_i(t|\boldsymbol{\theta}) + a_i(t|\boldsymbol{\theta})\cdot T \\ x_i(t+T|\boldsymbol{\theta}) = x_i(t|\boldsymbol{\theta}) + \frac{v_i(t|\boldsymbol{\theta}) + v_i(t+T|\boldsymbol{\theta})}{2}\cdot T = x_i(t|\boldsymbol{\theta}) + v_i(t|\boldsymbol{\theta})\cdot T + \frac{1}{2}a_i(t|\boldsymbol{\theta})\cdot T^2 \end{cases} \tag{2}$$

where T is the update time interval. Assume the update rule holds for discrete time intervals $T, 2T, \ldots, t$, here t represents a multiple of T. Consequently, the integrated forms of the continuous-time car-following model Eq. (1) is approximated by Eq. (2) with a ballistic update scheme. As we can see, both update time interval and reaction time are taken into account by the general form.

The reaction time is an essential feature of human driving. It is interesting to incorporate a reaction time in the human car-following law. Even if the reaction time T_r is not a multiple of the update time interval T, one simply linear approximation of the quantity $v_i(t-T_r|\boldsymbol{\theta})$ and $x_i(t-T_r|\boldsymbol{\theta})$ can be given by

$$\begin{cases} v_i(t-T_r|\boldsymbol{\theta}) = m\cdot v_i(t-(n+1)T|\boldsymbol{\theta}) + (1-m)\cdot v_i(t-nT|\boldsymbol{\theta}) \\ x_i(t-T_r|\boldsymbol{\theta}) = m\cdot x_i(t-(n+1)T|\boldsymbol{\theta}) + (1-m)\cdot x_i(t-nT|\boldsymbol{\theta}) \end{cases} \quad n = int\left(\frac{T_r}{T}\right), \quad m = \frac{T_r}{T} - n \tag{3}$$

where $n = int(\cdot)$ and m denote the integer and fractional functions, respectively. If we apply the same linear approximation to the leading vehicle $i-1$, the spacing and speed difference at time $t-T_r$ can be estimated, i.e., $\Delta x_{i-1,i}(t-T_r|\boldsymbol{\theta})$, $\Delta v_{i-1,i}(t-T_r|\boldsymbol{\theta})$.

To implement Eqs. (2) and (3), it is necessary to temporarily save the past $n+1$ values of speed and position in the buffer, i.e., $v_i(t-(n+1)T|\boldsymbol{\theta}), v_i(t-nT|\boldsymbol{\theta}), \ldots, v_i(t|\boldsymbol{\theta})$, and $x_i(t-(n+1)T|\boldsymbol{\theta}), x_i(t-nT|\boldsymbol{\theta}), \ldots, x_i(t|\boldsymbol{\theta})$.

It is clear that considering the reaction time as a variable, the calibration problem would become a mixed continuous integer problem, which is much more difficult. It is known that most of the algorithms cannot handle this unless simplified approaches are adopted. Treiber and Kesting (2013b) stated that the reaction time depended on many factors such as the age and experience of the driver, nevertheless, nearly all models and simulators assumed a constant common value for all drivers in all situations. When modeling traffic flow by *discrete-time* iterated coupled difference equations, the reaction time is often identified with the update time (e.g. Treiber and Kesting, 2013b). Since we focus on the optimization algorithm, it is assumed that drivers' reaction time is equal to the simulation update time in this paper, too. For presentation simplicity, we set the time horizontal so that the sampled times can be denoted as $T, 2T, \ldots, NT$ in the rest of this paper, where N denotes the number of the sampled data for calibration.

Constrained by length limit, we will address two widely used car-following models in this paper. The first model is the well-known Gazis–Herman–Rothery (GHR) model (Gazis et al., 1961; Ossen and Hoogendoorn, 2005) that depicts acceleration with respect to the velocity of the leader vehicle, relative velocity and spacing between following and leading vehicles, and driver reaction time as

$$a_i((j+1)T|\boldsymbol{\theta}_i) = c v_i^m(jT|\boldsymbol{\theta}_i)\frac{\Delta v_{i-1,i}(jT|\boldsymbol{\theta}_i)}{\Delta x_{i-1,i}^l(jT|\boldsymbol{\theta}_i)} \tag{4}$$

where T is the update time interval, j denotes the sequence of simulation time interval, $j = 1, 2, \ldots, N$; c, m, and l are car-following behavioral parameters. The GHR parameters are all box-bounded and can be written in a vector form as $\boldsymbol{\theta}_i = [c, m, l]^{\mathrm{T}}$. The velocity and position update rules of Eq. (2) are applied for the GHR model.

The second model is the intelligent driver model (IDM) (Treiber et al., 2000; Treiber and Kesting, 2013a,b), given by

$$a_i(jT|\boldsymbol{\theta}_i) = a\left(1 - \left(\frac{v_i(jT|\boldsymbol{\theta}_i)}{v_0}\right)^{\delta} - \left(\frac{s^*(v_i(jT|\boldsymbol{\theta}_i),\ \Delta v_{i-1,i}(jT|\boldsymbol{\theta}_i))}{s_i(jT|\boldsymbol{\theta}_i)}\right)^2\right) \tag{5}$$

This expression consists of an acceleration strategy towards a desired velocity v_0 with the parameter of the maximum acceleration a and a braking strategy which is dominant when the net distance $s_i(jT) = \Delta x_{i-1,i}(jT) - L_i$ to the preceding vehicle becomes smaller than the desired gap s^* defined by

$$s^*(v_i(jT),\ \Delta v_{i-1,i}(jT)|\boldsymbol{\theta}_i) = s_0 + T_0 v_i(jT|\boldsymbol{\theta}_i) + \frac{v_i(jT|\boldsymbol{\theta}_i)\Delta v_{i-1,i}(jT|\boldsymbol{\theta}_i)}{2\sqrt{ab}} \tag{6}$$

where s_0 is the minimum net distance in congested traffic, T_0 is a constant desired (safety) time gap of the leading vehicle. δ denotes the free acceleration exponent, which is set as 4 for simplicity. The last term is only active in non-stationary traffic and implements an "intelligent" driving behavior. All elements of the IDM parameter vector $\boldsymbol{\theta}_i = [a, b, v_0, T_0, s_0]^{\mathrm{T}}$ are positive box-bounded. Since the IDM model formulation is not a delayed model, and the acceleration output is calculated at the same time of inputs, see Eq. (5), we will apply car-following model state update rules of Eq. (2) with zero reaction for IDM.

Let us focus on the calibration of a particular vehicle i given the known trajectory of its leading vehicle $(i-1)$. Punzo and Simonelli (2005) compared several measures of performance (e.g. velocity, inter-vehicle spacing and headway) in the objective function and discussed which measure led to the best calibration result. Hollander and Liu (2008) had summarized different choices of performance index functions (including the percent error, mean error and squares error). In Section 3.2, we mainly discuss the widely used sum of velocity difference measure

$$\min_{\boldsymbol{\theta}\in\Omega} \quad g(\boldsymbol{\theta}) = \sum_{j=1}^{N}[v_i(jT|\boldsymbol{\theta}) - \hat{v}_i(jT)]^2 \tag{7}$$

where Ω is the solution space of $\boldsymbol{\theta}$, $v_i(jT|\boldsymbol{\theta})$ and $\hat{v}_i(jT)$ denote the simulated and empirical velocities of the ith following vehicle at time $t = jT,\ j = 2, \ldots, N$, respectively. $\hat{v}_i(jT)$ is taken as known inputs in the optimization problem.

The decision variable $\boldsymbol{\theta}$ does not explicitly appear in the objective function (7), and the induced variable $v_i(jT|\boldsymbol{\theta})$ are sequentially dependent by Eq. (2). Therefore, the derivative information of this objective function is difficult to obtain. To conquer this difficulty, we need to further analyze the property of the objective function.

In addition to the objective function of the SSE of velocity defined in (7), we had also tested other performance index functions. For example, we take the SSE of location (we will call it the SSE of gap in the follows) as the performance index. In this paper, for notation simplicity, alternatively, we still use $g(\boldsymbol{\theta})$ for the gap SSE given by

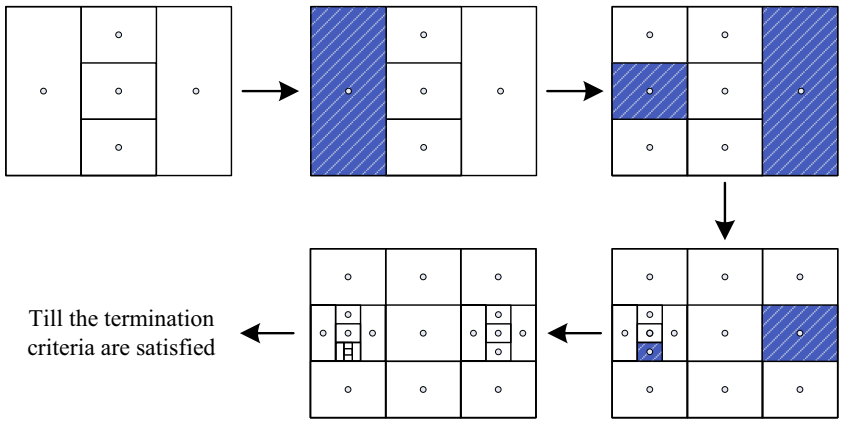


Fig. 1. An illustration of the partition and search process of the algorithm proposed in Jones et al. (1993) after 5 iterations, in which 21 objective function values are evaluated. The shadowed rectangles indicate the interested subspaces in each step.

$$\min_{\boldsymbol{\theta}\in\Omega} \quad g(\boldsymbol{\theta}) = \sum_{j=1}^{N} [x_i(jT|\boldsymbol{\theta}) - \hat{x}_i(jT)]^2 \tag{8}$$

where Ω is solution space of $\boldsymbol{\theta}$, $x_i(jT|\boldsymbol{\theta})$ and $\hat{x}_i(jT)$ denote the simulated and empirical positions of the ith following vehicle at time $t = jT,\ j = 2,\ldots,N$, respectively. $\hat{x}_i(jT)$ is taken as known inputs in the optimization problem.

3.2. The Lipschitz continuous property of the objective function

In this section, we will show that the objective functions (7) and (8) are Lipschitz continuous in the solution space $\Omega \subseteq R^n$ of $\boldsymbol{\theta}$. Constrained by the length limit, we only give the proof for the GHR model. However, the following conclusions can be similarly proven to hold for IDM and many car-following models with respect to other smooth and bounded objective functions. Indeed, the parameters of any a car-following model cannot be ill-conditioned; otherwise, the corresponding simulation results will be weird. So, we will pre-determine the appropriate parameter ranges for the car-following models. In such situations, the Lipschitz property is transferable.

Definition 1. A function $g(\boldsymbol{\theta}) : \Omega \to R$ is called Lipschitz continuous in Ω for any $\boldsymbol{\theta} \in \Omega$ if there exists a positive Lipschitz constant $\ell > 0$ satisfying

$$|g(\boldsymbol{\theta}) - g(\boldsymbol{\theta}')| \leqslant \ell \|\boldsymbol{\theta} - \boldsymbol{\theta}'\|_1 \quad \forall \boldsymbol{\theta},\ \boldsymbol{\theta}' \in \Omega \tag{9}$$

Proposition 1. *For the GHR model (4), the objective function (7) is Lipschitz continuous in the feasible domain Ω of parameters $\boldsymbol{\theta} \in \Omega \subseteq R^n$.*

Proof. First, let us eliminate variables $v_i((k-1)T)$ and $x_i((k-1)T)$ so that the whole optimization problem only depends on variables $a_i(kT|\boldsymbol{\theta}), k = 2,\ldots,\ j,j = 1,\ldots,N$.

Since $v_i(0)$ and $x_i(0)$ are known initial conditions of the ith vehicle, we can apply the car-following dynamic model (2) sequentially to get

$$\begin{aligned} v_i((k-1)T|\boldsymbol{\theta}) &= v_i((k-2)T|\boldsymbol{\theta}) + a_i((k-1)T|\boldsymbol{\theta}) \cdot T \\ &= v_i(0) + T\sum_{p=1}^{k-1} a_i(pT|\boldsymbol{\theta}) \end{aligned} \tag{10}$$

$$\begin{aligned} x_i((k-1)T|\boldsymbol{\theta}) &= x_i((k-2)T|\boldsymbol{\theta}) + v_i((k-2)T|\boldsymbol{\theta}) \cdot T + \frac{1}{2} a_i((k-1)T|\boldsymbol{\theta}) \cdot T^2 \\ &= x_i(0) + T\sum_{p=0}^{k-2} v_i(pT|\boldsymbol{\theta}) + \frac{T^2}{2}\sum_{p=1}^{k-1} a_i(pT|\boldsymbol{\theta}) \\ &= x_i(0) + v_i(0)(k-1)T + T^2\sum_{q=1}^{k-2}\sum_{p=1}^{q} a_i(pT|\boldsymbol{\theta}) + \frac{T^2}{2}\sum_{p=1}^{k-1} a_i(pT|\boldsymbol{\theta}) \end{aligned} \tag{11}$$

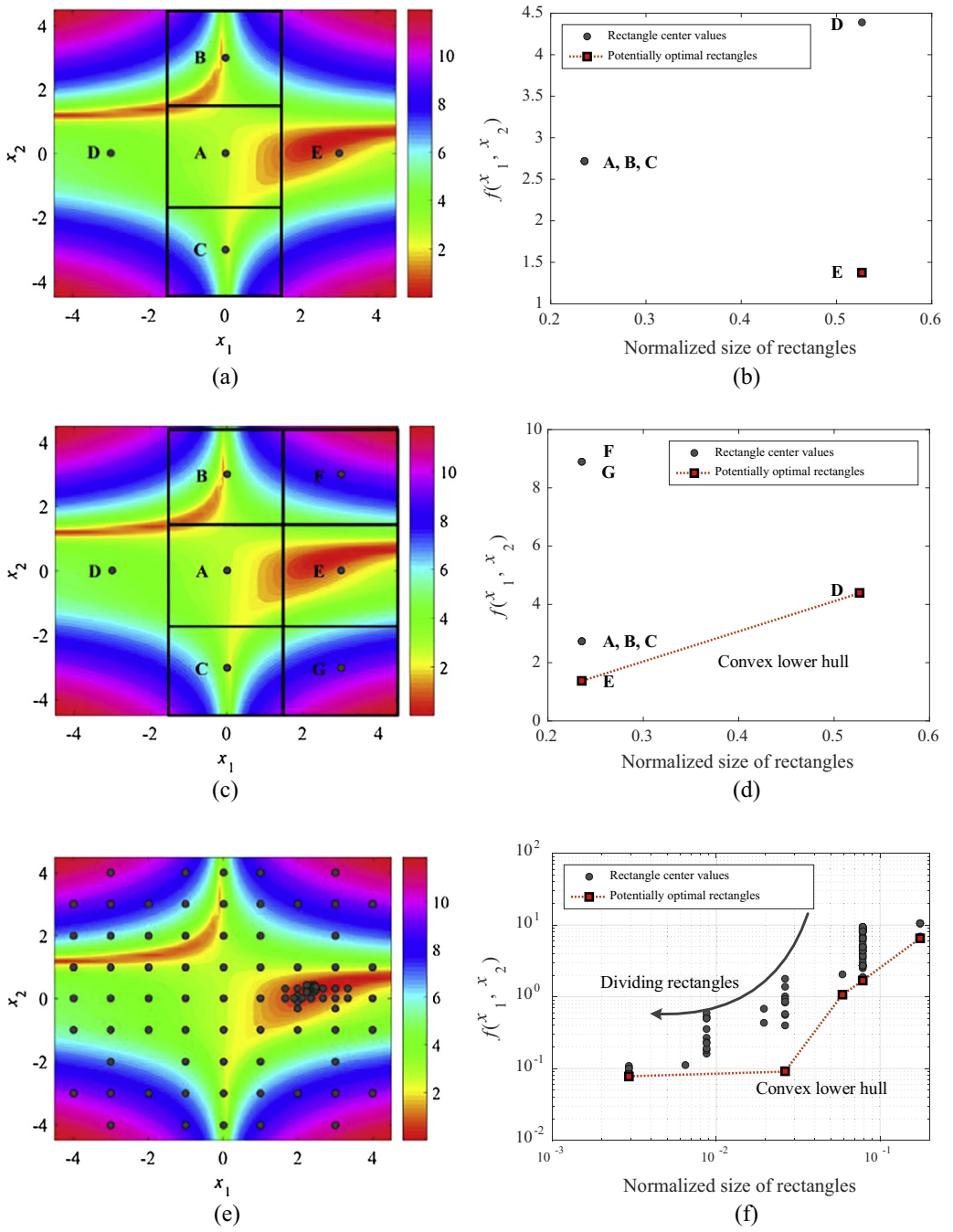


Fig. 2. The contour plot of test function and its convergence process to global optima after domain division procedures (the warm color towards to red indicates small objective function values, while the cool color towards to blue indicates large objective function values). (a–b) The initial iteration; (c–d) the 2nd iteration; (e–f) the 10th iteration. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Thus, the optimization problem can be rewritten as

$$\min_{\boldsymbol{\theta}} \quad g(\boldsymbol{\theta}) = \sum_{j=1}^{N} [v_i(jT|\boldsymbol{\theta}) - \hat{v}_i(jT)]^2 = \sum_{j=1}^{N} \left[v_i(0) + T\sum_{k=1}^{j} a_i(kT|\boldsymbol{\theta}) - \hat{v}_i(jT) \right]^2 \tag{12}$$

$$\begin{aligned}\text{s.t. } a_i(kT|\boldsymbol{\theta}) &= \frac{cv_i^m((k-1)T|\boldsymbol{\theta})[\hat{v}_{i-1}((k-1)T) - v_i((k-1)T|\boldsymbol{\theta})]}{[\hat{x}_{i-1}((k-1)T) - x_i((k-1)T|\boldsymbol{\theta})]^l} \\ &= \frac{c\left[v_i(0) + T\sum_{p=1}^{k-1} a_i(pT|\boldsymbol{\theta})\right]^m \left[\hat{v}_{i-1}((k-1)T) - v_i(0) - T\sum_{p=1}^{k-1} a_i(pT|\boldsymbol{\theta})\right]}{\left[\hat{x}_{i-1}((k-1)T) - x_i(0) - v_i(0)(k-1)T - T^2\sum_{q=1}^{k-2}\sum_{p=1}^{q} a_i(pT|\boldsymbol{\theta}) - \frac{T^2}{2}\sum_{p=1}^{k-1} a_i(pT|\boldsymbol{\theta})\right]^l}\end{aligned} \tag{13}$$

It is hard to directly prove that $g(\boldsymbol{\theta})$ is *Lipschitz-continuous* in terms of c, m, and l, because the constraints (13) are sequentially dependent. To get rid of this difficulty, we relax the constraints (13) and just assume $a_i(kT|\boldsymbol{\theta})$ are bounded (the existence of these bounds are given in Eqs. (15)–(17)). Notice that the newly formed feasible domain $\hat{\Omega}$ satisfies $\Omega \subset \hat{\Omega}$; so, if we can prove that $g(\boldsymbol{\theta})$ is Lipschitz continuous in $\hat{\Omega}, g(\boldsymbol{\theta})$ must be Lipschitz continuous in its subdomain Ω.

Based on Eq. (12), $g(\boldsymbol{\theta})$ is smooth and differentiable to each $a_i(kT|\boldsymbol{\theta})$. According to the mean value theorem, there exists a median parameter vector $\bar{\boldsymbol{\theta}} = \kappa\boldsymbol{\theta} + (1-\kappa)\boldsymbol{\theta}'$ with a positive coefficient $\kappa \in [0,1]$, such that we can write the gradient of $a_i(jT|\bar{\boldsymbol{\theta}})$ as

$$\nabla_{\boldsymbol{\theta}} a_i(jT|\bar{\boldsymbol{\theta}}) = \left(\frac{\partial a_i(jT|\bar{\boldsymbol{\theta}})}{\partial c}, \frac{\partial a_i(jT|\bar{\boldsymbol{\theta}})}{\partial m}, \frac{\partial a_i(jT|\bar{\boldsymbol{\theta}})}{\partial l}\right)^{\mathrm{T}} \tag{14}$$

whose elements can be given with respect to the GHR car-following parameters $\{c, m, l\}$ as

$$\left|\frac{\partial a_i(jT|\bar{\boldsymbol{\theta}})}{\partial c}\right| = \left|v_i^m((j-1)T|\bar{\boldsymbol{\theta}})\frac{\Delta v_{i-1,i}((j-1)T|\bar{\boldsymbol{\theta}})}{\Delta x_{i-1,i}^l((j-1)T|\bar{\boldsymbol{\theta}})}\right| \leqslant \left|\frac{v_{\max}^m \Delta v_{\max}}{\Delta x_{\min}^l}\right| = \lambda_1 \tag{15}$$

$$\begin{aligned}\left|\frac{\partial a_i(jT|\bar{\boldsymbol{\theta}})}{\partial m}\right| &= \left|cv_i^m((j-1)T|\bar{\boldsymbol{\theta}})\log v_i((j-1)T|\bar{\boldsymbol{\theta}})\frac{\Delta v_{i-1,i}((j-1)T|\bar{\boldsymbol{\theta}})}{\Delta x_{i-1,i}^l((j-1)T|\bar{\boldsymbol{\theta}})}\right| \\ &\leqslant \left|\frac{cv_{\max}^m \Delta v_{\max}}{\Delta x_{\min}^l}\log v_{\max}\right| = \lambda_2\end{aligned} \tag{16}$$

$$\begin{aligned}\left|\frac{\partial a_i(jT|\bar{\boldsymbol{\theta}})}{\partial l}\right| &= \left|cv_i^m((j-1)T|\bar{\boldsymbol{\theta}})\frac{\Delta v_{i-1,i}((j-1)T|\bar{\boldsymbol{\theta}})}{\Delta x_{i-1,i}^l((j-1)T|\bar{\boldsymbol{\theta}})}\log\frac{1}{\Delta x_{i-1,i}((j-1)T|\bar{\boldsymbol{\theta}})}\right| \\ &\leqslant \left|\frac{cv_{\max}^m \Delta v_{\max}}{\Delta x_{\min}^l}\log \Delta x_{\min}\right| = \lambda_3\end{aligned} \tag{17}$$

for $j = 1, \ldots, N$. Here, the positive scalar λ_1, λ_2 and λ_3 characterize the maximum possible values of $\left|\frac{\partial a_i(jT|\bar{\boldsymbol{\theta}})}{\partial c}\right|$, $\left|\frac{\partial a_i(jT|\bar{\boldsymbol{\theta}})}{\partial m}\right|$ and $\left|\frac{\partial a_i(jT|\bar{\boldsymbol{\theta}})}{\partial l}\right|$, respectively.

Let $\lambda_{\max} = \max\{\lambda_1, \lambda_2, \lambda_3\}$, we have

$$\begin{aligned}|g(\boldsymbol{\theta}) - g(\boldsymbol{\theta}')| &= \left|\nabla_{\boldsymbol{\theta}}^{\mathrm{T}} g(\bar{\boldsymbol{\theta}}) \cdot (\boldsymbol{\theta} - \boldsymbol{\theta}')\right| \\ &= \left|\sum_{j=1}^{N} \frac{\partial g(\bar{\boldsymbol{\theta}})}{\partial v_i(jT|\bar{\boldsymbol{\theta}})}\frac{\partial v_i(jT|\bar{\boldsymbol{\theta}})}{\partial a_i(jT|\bar{\boldsymbol{\theta}})}\nabla_{\boldsymbol{\theta}}^{\mathrm{T}} a_i(jT|\bar{\boldsymbol{\theta}}) \cdot (\boldsymbol{\theta} - \boldsymbol{\theta}')\right| \\ &= 2T\left|\sum_{j=1}^{N}\left[v_i(jT|\bar{\boldsymbol{\theta}}) - \hat{v}_i(jT)\right]\left[\nabla_{\boldsymbol{\theta}}^{\mathrm{T}} a_i(jT|\bar{\boldsymbol{\theta}}) \cdot (\boldsymbol{\theta} - \boldsymbol{\theta}')\right]\right| \\ &\leqslant 2T\left|\sum_{j=1}^{N}\left[v_i(jT|\bar{\boldsymbol{\theta}}) - \hat{v}_i(jT)\right]\right|\left|\lambda_{\max}\mathbf{1}^{\mathrm{T}} \cdot (\boldsymbol{\theta} - \boldsymbol{\theta}')\right| \\ &\leqslant 2T\lambda_{\max}\left|\sum_{j=1}^{N}\left[v_i(jT|\bar{\boldsymbol{\theta}}) - \hat{v}_i(jT)\right]\right|\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|_1\end{aligned} \tag{18}$$

where $\mathbf{1} = [1, \ldots, 1]^{\mathrm{T}}$ is the element vector.

Define a Lipschitz constant $\ell = 2T\lambda_{\max}\left|\sum_{j=1}^{N}\left[v_i(jT|\bar{\boldsymbol{\theta}}) - \hat{v}_i(jT)\right]\right|$, the conclusion is naturally reached according to Definition 1. □

Proposition 2. *For the GHR model (4), the objective function (8) is Lipschitz continuous in the feasible domain Ω of parameters $\boldsymbol{\theta} \in \Omega \subseteq R^n$.*

Proof. First, similar to (10) and (11), the optimization problem can be rewritten as

$$\begin{aligned}\min_{\boldsymbol{\theta}} \quad g(\boldsymbol{\theta}) &= \sum_{j=1}^{N}\left[x_i(jT|\boldsymbol{\theta}) - \hat{x}_i(jT)\right]^2 \\ &= \sum_{j=1}^{N}\left[x_i(0) + jT\cdot v_i(0) + T^2\cdot\sum_{k=1}^{j}\left[a_i(kT|\boldsymbol{\theta})\cdot\left(j-k+\frac{1}{2}\right)\right] - \hat{x}_i(jT)\right]^2\end{aligned} \tag{19}$$

s.t. (13)

Based on Eq. (19), $g(\boldsymbol{\theta})$ is smooth and differentiable to each $a_i(kT|\boldsymbol{\theta})$, then we have

$$\begin{aligned}|g(\boldsymbol{\theta}) - g(\boldsymbol{\theta}')| &= \left|\nabla_{\boldsymbol{\theta}}^{\mathrm{T}} g(\bar{\boldsymbol{\theta}})\cdot(\boldsymbol{\theta}-\boldsymbol{\theta}')\right| \\ &= \left|\sum_{j=1}^{N}\frac{\partial g(\bar{\boldsymbol{\theta}})}{\partial x_i(jT|\bar{\boldsymbol{\theta}})}\frac{\partial x_i(jT|\bar{\boldsymbol{\theta}})}{\partial a_i(jT|\bar{\boldsymbol{\theta}})}\nabla_{\boldsymbol{\theta}}^{\mathrm{T}} a_i(jT|\bar{\boldsymbol{\theta}})\cdot(\boldsymbol{\theta}-\boldsymbol{\theta}')\right| \\ &= T^2\left|\sum_{j=1}^{N}\left[x_i(jT|\bar{\boldsymbol{\theta}}) - \hat{x}_i(jT)\right]\left[\nabla_{\boldsymbol{\theta}}^{\mathrm{T}} a_i(jT|\bar{\boldsymbol{\theta}})\cdot(\boldsymbol{\theta}-\boldsymbol{\theta}')\right]\right| \\ &\leqslant T^2\left|\sum_{j=1}^{N}\left[x_i(jT|\bar{\boldsymbol{\theta}}) - \hat{x}_i(jT)\right]\right|\left|\lambda_{\max}\mathbf{1}^{\mathrm{T}}\cdot(\boldsymbol{\theta}-\boldsymbol{\theta}')\right| \\ &\leqslant T^2\lambda_{\max}\left|\sum_{j=1}^{N}\left[x_i(jT|\bar{\boldsymbol{\theta}}) - \hat{x}_i(jT)\right]\right|\|\boldsymbol{\theta}-\boldsymbol{\theta}'\|_1\end{aligned} \tag{20}$$

Define a new Lipschitz constant $\ell' = T^2\lambda_{\max}\left|\sum_{j=1}^{N}\left[x_i(jT|\bar{\boldsymbol{\theta}}) - \hat{x}_i(jT)\right]\right|$, the conclusion is naturally reached according to Definition 1. □

The above important property indicates that we could apply the Lipschitz optimization to solve microscopic car-following model calibration problems.

4. The new optimization algorithm for model calibration

Most practical solving methods for the car-following model calibration problem belong to direct search algorithms (Kolda et al., 2003; Conn et al., 2009; Rios and Sahinidis, 2013), which usually refer to optimization techniques that do not explicitly use derivatives. They are often described as sequential examinations of new trial decision vectors generated by a certain strategy with respect to existing trial decision vectors.

In Section 4.1, we will introduce the Lipschitz optimization algorithm that is also a direct search algorithm and discuss whether it could be used for the parameter calibration. Then, we propose a new optimization algorithm in Section 4.2.

4.1. A Lipschitzian optimization algorithm without presetting the Lipschitz constant

The above four direct search algorithm do not explicitly employ the Lipschitz continuous property of the objective function. In this subsection, we will introduce a direct search algorithm dedicated to Lipschitzian continuous objectives.

Lipschitz optimization refers to a kind of deterministic algorithms that try to find the global minimum points by iteratively partitioning the feasible region into smaller sub-regions by using the Lipschitz continuous property of the objective function.

In all Lipschitzian optimization algorithms, the assumed Lipschitz constant controls the preference balance between global search and local search. If the Lipschitz constant is not known, we often assume that it is a large enough number, because we must guarantee it is equal to or larger than the actual Lipschitz constant. As a result, the convergence speed toward the global optimal solution is often slow.

To conquer this difficulty, a Lipschitzian optimization algorithm without pre-selecting the Lipschitz constant was proposed in Jones et al. (1993). This algorithm was named according to one of its primary operations: DIviding RECTangles. DIRECT is a sampling algorithm that requires no knowledge of the objective function gradient. It does not try to exploit any correlation between the function value at one point and its value at nearby points, and thus can be very useful when the objective function is implicit or based on simulation. We will abbreviate it as DIRECT algorithm in the rest of this paper.

The first step of the DIRECT algorithm is to transfer the solution space into a hyper-rectangle. Then, we pick the center-point of this hyper-rectangle, evaluate the objective function in that point, and divide the hyper-rectangle into smaller rectangles. Instead of using a pre-selected Lipschitz constant to determine which hyper-rectangle(s) to explore next, a set of potentially optimal (PO) hyper-rectangles are identified that may contain the global optimal solution. The mathematical definition and the identification method of PO hyper-rectangles are given in Appendix A. All of these PO hyper-rectangles will be further divided into even smaller ones whose center-points are evaluated; see an illustration in Fig. 1. This procedure will be repeated, until the objective function values at the vertices of all of the current rectangles satisfy some pre-selected terminating criteria, e.g. when it exceeds its budget of iterations, rectangle divisions, or objective function evaluations.

The Lipschitzian optimization algorithm is illustrated using a widely applied two-dimensional test function for better visualization. We aim to reveal the detailed search process of DIRECT using the following illustrative numerical example in this paper, which is defined as

$$f(x_1,x_2)=\log\left[(1.5-x_1+x_1x_2)^2+(2.25-x_1+x_1x_2^2)^2+(2.625-x_1+x_1x_2^3)^2+1\right] \tag{21}$$

with $-4.5\leqslant x_1\leqslant 4.5$ and $-4.5\leqslant x_2\leqslant 4.5$. The global minimum is $f_{\min}=0$ at $\mathbf{x}^*=(x_1^*,x_2^*)=(3,0.5)$.

The function value at the middle of each rectangle and the half value of its diagonal are used to determine PO rectangles. Each PO rectangle is divided in the next iteration. Fig. 2 shows the sequential procedures of the algorithm for the test function Eq. (21). In the 10th iteration, we can see the algorithm finds the basin of global optima and more dense division is made to this region.

The DIRECT algorithm carries out simultaneous searches in different subspaces using all of the possible Lipschitz constants, and therefore better balances global and local searches. As proven in Jones et al. (1993), the DIRECT algorithm guarantees to hit the global optimal solution, if it runs enough iterations. It has no configuration parameters to select. Moreover, it explicitly takes box-bounded constraints into account and are thus suitable for car-following model calibration problems.

One major shortcoming of DIRECT algorithm is its slow convergence speed around the global optimal solution, if the objective function is flat at the global minimum. This is because DIRECT algorithm can only discard very few subspace at each round of iterations in such situations.

For the car-following model calibration problem, we find that the DIRECT algorithm can quickly reach the basin of the global optimal solution after a few iterations; see Fig. 3 for example, where the objective function is the sum of square errors (SSE) of simulated velocities. Unfortunately, the converging speed of the DIRECT algorithm then becomes really slow, because the objective function (6) is often flat around the global optimal solution. So, we need to find other algorithms to accelerate the convergence speed.

4.2. Combine Lipschitz optimization algorithm with local search algorithm

In this subsection, we propose a new algorithm. It first executes the Lipschitz optimization algorithm to identify a few number of hyper-rectangles which may contain the global optimal solution(s). Then, this new algorithm switches to local search, e.g., SQP algorithm, starting at the centers of these hyper-rectangles to find the local optimal solutions within the selected hyper-rectangles. This strategy overcomes the weaknesses of random multi-start heuristics used in many previous studies (Brockfeld et al., 2005; Ossen and Hoogendoorn, 2008), since the global optimal solution may still locate far away from all of the randomly generated starting vectors.

Parameter	Notation
d_0	Critical size of transition from global to local search
$g_{\min}$	Minimal objective function value
κ	Maximal multi-start times
m	Division iteration
r	The sequence of PO hyper-rectangles
S_m	Set of PO hyper-rectangles at division iteration m
$\boldsymbol{\theta}_{\max}$	Upper bound of decision variables $\boldsymbol{\theta}\in R^n$
$\boldsymbol{\theta}_{\min}$	Lower bound of decision variables $\boldsymbol{\theta}\in R^n$
$\boldsymbol{\theta}_{mr}$	Center of the PO hyper-rectangle r at division iteration m
$\hat{\boldsymbol{\theta}}$	Optimal solution
$\hat{\boldsymbol{\theta}}_{mr}$	Local optima corresponding to PO hyper-rectangle r at division iteration m
$[\boldsymbol{\theta}]$	Normalized decision variables, $[\boldsymbol{\theta}]\in[0,1]^n$
$[\hat{\boldsymbol{\theta}}]_{mr}$	Normalized local optima corresponding to PO hyper-rectangle r at division iteration m

Algorithm 1. Combined Lipschitz Global and Local Search Algorithm

Input: set configuration values for the critical size d_0, maximal multi-start times κ, upper bound $\boldsymbol{\theta}_{\max}$ and lower bound $\boldsymbol{\theta}_{\min}$ of decision variables $\boldsymbol{\theta}\in R^n$.

Initialization: Let the division iteration $m=1$. Normalize the feasible domain to be the unit hyper-rectangle $[\boldsymbol{\theta}]=(\boldsymbol{\theta}-\boldsymbol{\theta}_{\min})/(\boldsymbol{\theta}_{\max}-\boldsymbol{\theta}_{\min})\times 100\%\in[0,1]^n$ with the normalized center $[\boldsymbol{\theta}]_{11}$ (the first subscript number denotes the division iteration, the second one is the order of PO hyper-rectangles), which corresponds to an original vector $\boldsymbol{\theta}_{11}\in[\boldsymbol{\theta}_{\min},\boldsymbol{\theta}_{\max}]$. Evaluate $g(\boldsymbol{\theta}_{11})$ and let the optimal solution $\hat{\boldsymbol{\theta}}=\boldsymbol{\theta}_{11}, g_{\min}=g(\boldsymbol{\theta}_{11})$. Divide the hyper-rectangle.

while number of PO hyper-rectangles (whose sizes $\leqslant d_0$) $\leqslant \kappa$ **do**[a]
  Identify the set S_m of all PO hyper-rectangles
  for all $r \in S_m$ **do**
    Evaluate the objective function value at the center $[\boldsymbol{\theta}]_{mr}$ (corresponds to an original vector $\boldsymbol{\theta}_{mr} \in [\boldsymbol{\theta}_{\min}, \boldsymbol{\theta}_{\max}]$) of the PO hyper-rectangle r, i.e. $g(\boldsymbol{\theta}_{mr})$. If $g(\boldsymbol{\theta}_{mr}) < g_{\min}$, then $g_{\min} = g(\boldsymbol{\theta}_{mr}),\ \hat{\boldsymbol{\theta}} = \boldsymbol{\theta}_{mr}$
    if size of r is equal to or smaller than d_0 **do**
      Switch to local optimization SQP algorithm with the starting point of $[\boldsymbol{\theta}]_{mr}$
      Converge to local optima $[\hat{\boldsymbol{\theta}}]_{mr}$, If $g(\hat{\boldsymbol{\theta}}_{mj}) < g_{\min}$, then $g_{\min} = g(\hat{\boldsymbol{\theta}}_{mr}),\ \hat{\boldsymbol{\theta}} = \hat{\boldsymbol{\theta}}_{mr}$
    else do
      Divide the PO hyper-rectangle r into smaller hyper-rectangles
      Evaluate objective function values at centers of new hyper-rectangles
      Update $g_{\min}$ and $\hat{\boldsymbol{\theta}}$ if a smaller value is found
    end if
  end for
  division iteration $m = m + 1$
end while
return $\hat{\boldsymbol{\theta}}$ and $g_{\min}$

[a] Note: If more than κ PO hyper-rectangles of an equal size are found, then arbitrary κ of them will be selected as starting hyper-rectangles for local search.

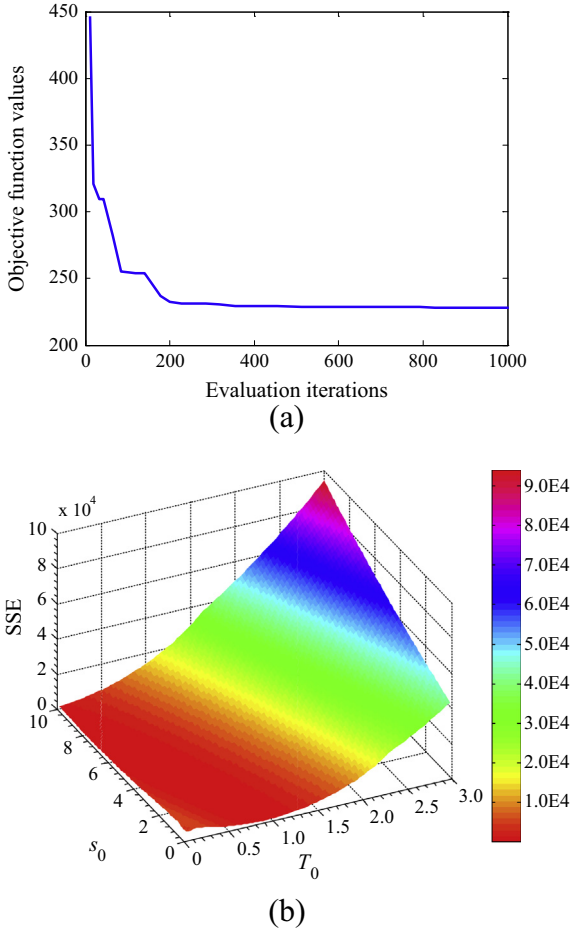


Fig. 3. (a) The convergence process of DIRECT algorithm for an IDM calibration problem with the SSE of velocity according to a particular trajectory pair in NGSIM. The termination condition has not been satisfied even after testing 1000 vectors. (b) The surface of the corresponding objective function projected in the 2D s_0 versus T_0 space.

The kernel of this two-step search strategy lies in when to start the local search. Given a hyper-rectangle that may contain the global optimal solution, we use its size to determine whether we should further divide it into smaller hyper-rectangles or start the local search within this hyper-rectangle. If the size of this PO hyper-rectangle is larger than a pre-selected threshold, we execute the DIRECT algorithm to further divide it; otherwise, we will switch to a local search starting from the current midpoint of this hyper-rectangle. In this paper, we choose the size to determine the switch condition; since the smaller the PO hyper-rectangle is, the higher probability this PO hyper-rectangle contains the global optimal solution.

So, the first configuration parameter in this new optimization algorithm is the pre-selected configuration value d_0 of the smallest PO hyper-rectangle.

In this paper, we normalize the parameter vector $\boldsymbol{\theta}$ into $[0,1]$ by transforming the box-bounded solution space into $(\boldsymbol{\theta}-\boldsymbol{\theta}_{\min})/(\boldsymbol{\theta}_{\max}-\boldsymbol{\theta}_{\min})\times 100\%$. Moreover, we define the size of hyper-rectangle as the half of its diagonal, or equivalently, the distance from the midpoint to an arbitrary vertex. So, the normalized hyper-rectangle becomes a n-dimensional unit hypercube with size $\sqrt{n}/2$.

As shown in Fig. 4, in each iteration, the DIRECT algorithm will determine a number of PO hyper-rectangles. We will check whether the size of any of these PO hyper-rectangles is smaller than a pre-selected critical size d_0. If the switching condition had been satisfied in any a PO hyper-rectangle, we will stop applying the DIRECT algorithm and begin to apply the SQP algorithm in the smallest few PO hyper-rectangles for local searches. Otherwise, we will begin a new iteration, continue to divide all the PO hyper-rectangles and determine a new set of PO hyper-rectangles.

So, the second configuration parameter in this new optimization algorithm is κ, the number of the PO hyper-rectangles in which we apply the SQP algorithm for local searches. If we have more than κ PO hyper-rectangles whose sizes are equally the smallest, we will choose arbitrary κ of them to start local searches. Besides, we always start the local searches at the center-points of all PO hyper-rectangles.

5. Numerical tests

5.1. Test data, algorithm settings and performance criteria

In this section, the performance of the proposed algorithm in Section 3 is tested on the open-access trajectory dataset provided by the Next Generation SIMulation (NGSIM) program. We use the data collected on a segment of U.S. Highway 101 (Hollywood Freeway) in Los Angeles, California between 7:50 AM and 8:35 AM on June 15, 2005 (NGSIM, 2006). The sample frequency of vehicle trajectories is 10 Hz (with a resolution of 0.1 s). This dataset includes overall 6101 vehicles monitored on the 640-m freeway segment. Detailed data filtering procedures for the NGSIM Highway 101 dataset can be found in Thiemann et al. (2008), Chen et al. (2010, 2013), and Li et al. (2013a,b).

As shown in Fig. 5, since the length of the data collection site is only less than 700 m, to keep more data samples of paired vehicles for the calibration purpose, we randomly chose 100 leader–follower trajectory pairs whose travel times were longer than 60 s along the study highway and used these six algorithms to solve the calibration problem, respectively. Those trajectories were impacted by at least one shockwave, so we could use them to well describe the drivers' acceleration and deceleration maneuvers. Moreover, the simulation time step was set as $T = 0.5$ s.

In this paper, we adopt the package contributed by Finkel (2003) to implement the DIRECT algorithm. The NM algorithm is implemented according to Lagarias et al. (1998). The SQP algorithm is implemented according to Fletcher and Powell

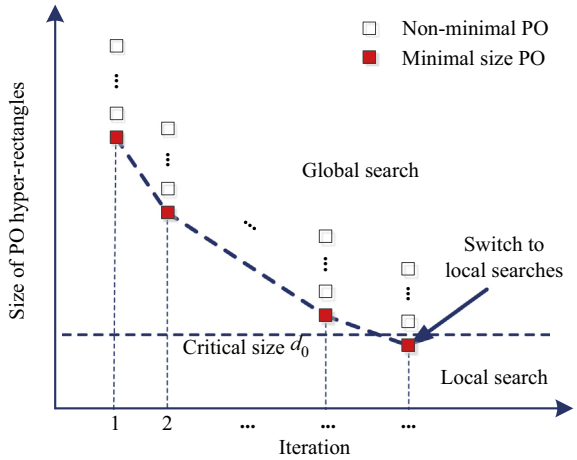


Fig. 4. An illustration of the switching condition if the minimal potentially optimal points identification.

(1963) and Fletcher et al. (2002). The GA algorithm is implemented according to Goldberg (1989). The SPSA algorithm is implemented according to Spall (2000, 2003). The configuration parameters of each algorithm are set as follows:

- NM. Maximum function evaluation is 10^4, maximum iteration is 10^3, both termination tolerances on function value and decision variables are 10^{-12}, and uniform random start vector is within $[\boldsymbol{\theta}_{\min}, \boldsymbol{\theta}_{\max}]$.
- SQP. Maximum function evaluation is 10^4, uniform random start vector is within $[\boldsymbol{\theta}_{\min}, \boldsymbol{\theta}_{\max}]$, and gradient sup-norm tolerance is 10^{-5}.
- GA. Population size is 50, maximum number of generations is 200 (each generation includes 50 individuals), termination tolerance on function value is 10^{-12}, fraction of population created by the crossover function is 0.8, initial value of penalty parameter is 10, migration fraction is 0.2, migration interval is 20, and fraction of individuals to keep on the first Pareto front while the solver selects individuals from higher fronts is 0.35.
- SPSA. Maximum iteration is 10^4, two sided simultaneous perturbation to unknown gradient, generate random perturbation vector with Bernoulli distribution, and uniform random start vector is within $[\boldsymbol{\theta}_{\min}, \boldsymbol{\theta}_{\max}]$.
- DIRECT. Maximum function evaluation is 10^4, maximum hyper-rectangle division is 10^4, maximum iteration is 10^3.
- DIRECT + SQP. Global search: Critical size of normalized PO hyper-rectangles is $d_0 = 10^{-2}$ for the IDM and $d_0 = 10^{-4}$ for the GHR model in this test (transform the original hyper-rectangle into a unit hypercube using the normalization of $(\boldsymbol{\theta} - \boldsymbol{\theta}_{\min})/(\boldsymbol{\theta}_{\max} - \boldsymbol{\theta}_{\min}) \times 100\%$, then the size of normalized hyper-rectangles lies between 0 and $\sqrt{n}/2$), maximum function evaluation is 10^4, maximum hyper-rectangle division is 10^4, and maximum iteration is 10^3; Local search: Maximum function evaluation is 10^4, a gradient sup-norm tolerance is 10^{-5}.

The bounds of the IDM parameters are $\boldsymbol{\theta}_{\text{IDM, min}} \leqslant \boldsymbol{\theta}_{\text{IDM}} \leqslant \boldsymbol{\theta}_{\text{IDM, max}}$, with $\boldsymbol{\theta}_{\text{IDM, min}} = [1\ \text{m/s}^2,\ 1\ \text{m/s}^2,\ 10\ \text{m/s},\ 0\ \text{s},\ 1\ \text{m}]^{\text{T}}$ and $\boldsymbol{\theta}_{\text{IDM, max}} = (3\ \text{m/s}^2,\ 4\ \text{m/s}^2,\ 30\ \text{m/s},\ 3\ \text{s},\ 10\ \text{m})$. The bound of the GHR model parameters is set as $\boldsymbol{\theta}_{\text{GHR, min}} \leqslant \boldsymbol{\theta}_{\text{GHR}} \leqslant \boldsymbol{\theta}_{\text{GHR, max}}$, with $\boldsymbol{\theta}_{\text{GHR, min}} = [0, 0,\ 0]^{\text{T}}$ and $\boldsymbol{\theta}_{\text{GHR, max}} = [500,\ 1,\ 5]^{\text{T}}$. It should note that the parameters of the GHR model don't have specific units or physical meanings as the IDM's.

In particular, we use abbreviation DIRECT + SQP (1) to refer the algorithm that apply the SQP algorithm in only the smallest PO hyper-rectangle. We then use abbreviation DIRECT + SQP (3) to refer the algorithm that apply the SQP algorithm in the smallest three PO hyper-rectangles.

To compare the performance of different algorithms, we set up two criteria:

(1) *The probability of finding the global optimal solution.*

Because the calibration problem is highly nonlinear and non-convex, no existing algorithm can guarantee to find the global optimal solution within a limited time period. Thus, an algorithm is the best, if it has higher chance to find the global optimal solution within a pre-selected time period than all its competitors.

(2) *The number of function evaluations that have been conducted before an algorithm reaches the global optimal solution (vector in the solution space) or satisfies the termination condition.*

For each candidate vector of car-following parameters, a calibration problem requires to simulate the whole trajectory before comparing it with the empirical one. Since each simulation procedure consumes a certain time cost, an algorithm is the best, if it searches fewer vectors on average to find the global optimal solution than all of its competitors.

Here, we run the DIRECT algorithm for a very large number of iterations (10^7 iterations) and assume the finally obtained vector is the global optimal solution. If the difference between the optimized objective function and the global optimal value is no larger than 10^{-4} $[\text{m/s}]^2$ for the SSE of velocity (it minimizes the sum of square errors between the observed velocities in the dataset and the corresponding simulated velocities by a car-following model), we assume it has reached the global optimal solution.

We do not further consider how to balance these two criteria in the following tests, since the DIRECT + SQP algorithm simultaneously has both the fastest convergence speed and the highest probability of finding the global optimal solution.

5.2. Test results

To provide an intuitive illustration, Fig. 6 plots the convergence processes of six algorithms aforementioned when they are applied to one randomly selected pair of trajectories. For both IDM and GHR car-following models, the objectives are minimizing the sum of velocity difference between the observation and the simulation in (7). A convergence curve is ended, after the corresponding algorithm reaches a local optimal solution or the termination condition is satisfied.

In the tests, we randomly generate initial vectors within the box bounds for the NM, SQP, GA and SPSA algorithms as the starting points. We generate only one starting points for these algorithms so as to study their performances when they are run for just once.

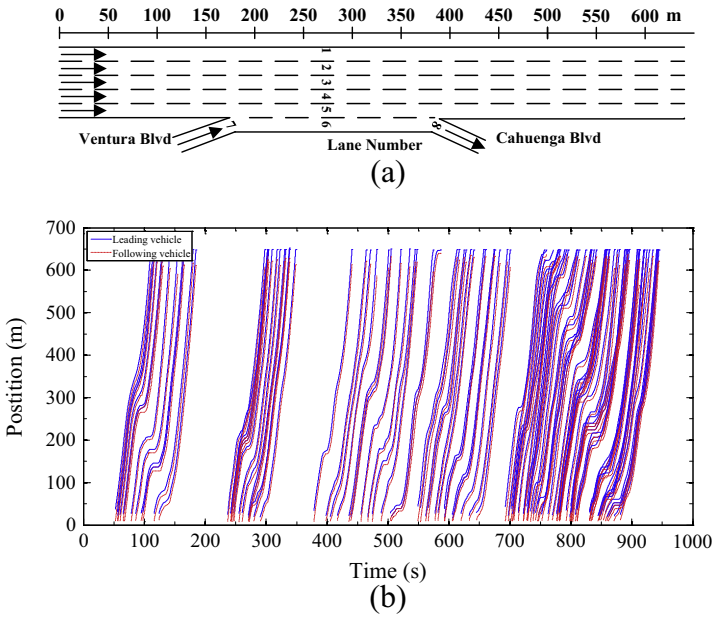


Fig. 5. (a) Data collection scheme; (b) car-following vehicle pairs for testing algorithm.

Fig. 6(a) compares the convergence processes of the six algorithms for the IDM calibration with the objective function Eq. (6). Since the original NM algorithm does not consider the constraints of the decision vector, it will soon converge to a solution that is outside the bound $[\boldsymbol{\theta}_{\min}, \boldsymbol{\theta}_{\max}]$ (although the found objective value is similar to the global optimal solution). The SQP algorithm is quickly trapped at a local minimum $g_{SQP} = 496$ [m/s]^2, which is beyond the vertical axis limit so that we cannot plot it in Fig. 5(a). The convergence speeds of GA and SPSA algorithms are both slow. Neither of them hit the global optimal solution after testing 1000 candidate vectors. The best objective function values of SQP and SPSA are larger than 45 [m/s]^2 (the best objective function value given by SPSA is $g_{SPSA} = 726$ [m/s]^2), which is also beyond the vertical axis limit. So, we cannot plot them in Fig. 6(a). The DIRECT algorithm reaches the basin of the global optimal solution after testing about 300 candidate vectors. However, the convergence speed of DIRECT algorithm slows down after that and prevents it from hitting the global minimum even after testing 400 candidate vectors. The DIRECT + SQP (1) algorithm hits the global minimum after testing about 120 candidate vectors, which gives the fastest convergence speed in the six algorithms. The computation time spent by the six algorithms are 17.49 s, 0.09 s, 5.27 s, 3.26 s, 2.09 s, 0.12 s, respectively.

Similarly, Fig. 6(b) shows that the performance of the six algorithms for the GHR model calibration using the same trajectory pair. The objective function is still Eq. (7). In this case, the NM algorithm again violates the box constraint of the decision vector. The SPSA algorithm is trapped into a local solution whose best objective function value is $g_{SQP} = 2832$ [m/s]^2. Since this value is beyond the vertical axis limit, we cannot plot it in Fig. 6(b). The GA algorithm cannot find the minimum after even testing 1000 candidate vectors. The DIRECT algorithm hits the global minimum after testing about 350 candidate vectors; while the DIRECT + SQP (1) algorithm hits the global minimum after testing about 260 candidate vectors. This indicates that the DRECT + SQP algorithm outperforms other algorithms in convergence speed. The computation time spent by the six algorithms are 11.07 s, 0.48 s, 4.76 s, 4.70 s, 2.71 s, 0.16 s, respectively.

To explore the reliability (percentage of search processes that reach the global optimal solution finally) of each algorithm, we apply the six algorithms to the randomly selected 100 trajectories. Table 1 shows the results for all six algorithms that minimize the SSE of velocity for the IDM, where, NM, SQP, GA, and SPSA algorithms use one randomly generated starting point and we call it single-start.

The first column of Table 1 shows how many calibration problems have been correctly solved by each algorithm. The DIRECT + SQP (1) algorithm hits the global optimal solution for 98 problems and obviously has the largest probability to find the global optimal solution.

The second column of Table 1 shows how fast each algorithm reaches the basin of the global optimal solution. Here, we assume the basin is reached if the relative error of the current objective value and the global optimal value is below 1%. As shown in Table 1, the DIRECT algorithm has a poor convergence speed when it approaches the global optimal solution. This shortcoming can be well fixed when we switch to the SQP algorithm for a much faster local search, after we incorporate the transition condition when the minimal size of PO hyper-rectangles is less than the critical size, i.e. $d_0 = 10^{-2}$ for the IDM and $d_0 = 10^{-4}$ for the GHR model in this test. We will further examine the sensitivity of d_0 at the end of this section.

The third and fourth columns of Table 1 shows how fast each algorithm solves the calibration problem, if it correctly stops at the global optimal solution. In this paper, tests are run with a 2.60 GHz-quad CPU and 4 GB-Ram computer. It should be

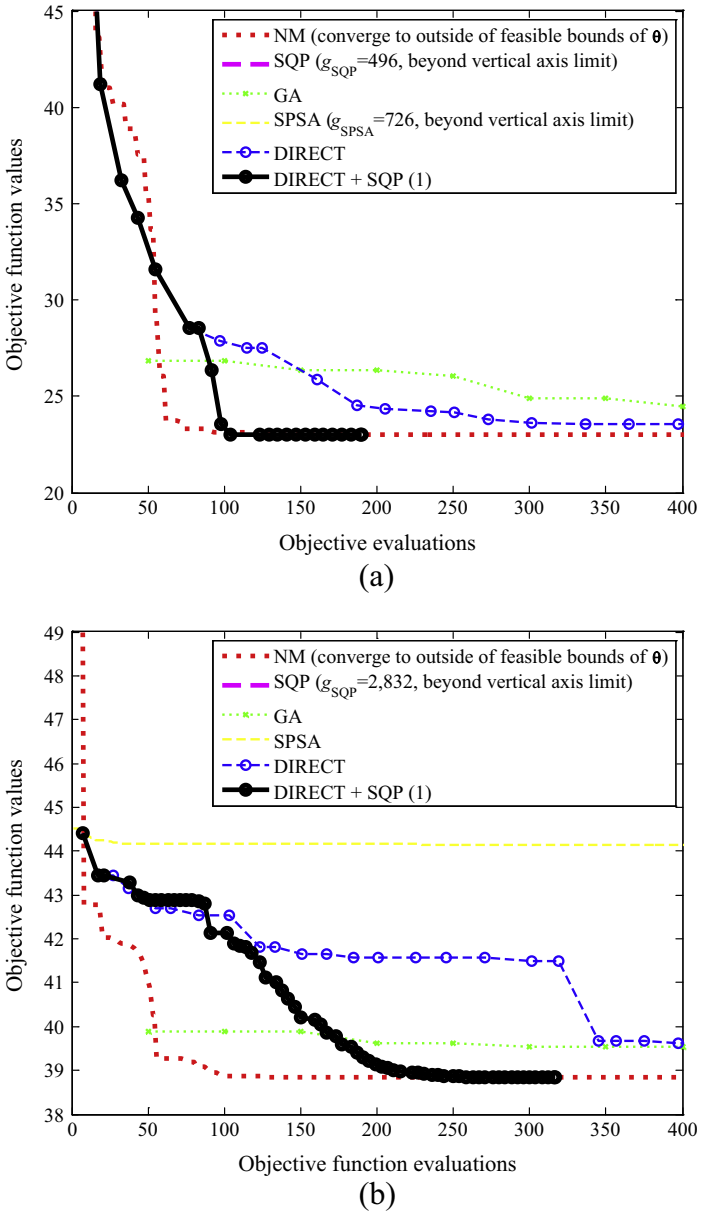


Fig. 6. Comparison of the speeds of convergence for different calibration algorithms: (a) IDM and (b) GHR model, with the SSE of velocity.

pointed out that SQP algorithm consumes more time when applying the Newton's method to solve a quadratic objective subject to a linearization of the constraints. So, the total calibration time does not grow proportionally with the tested objective function evaluations. Consequently, the average calibration time spent on one objective evaluation of the DIRECT step is much shorter than the SQP step (due to additional computation efforts made on the quadratic programming). Despite the DIRECT + SQP algorithm requires more objective evaluations, its calibration time doesn't increase dramatically. Clearly, the DIRECT + SQP algorithm achieves the largest probability to reach the global optimal solution and consumes the least computation cost.

Table 2 shows the comparison results for the GHR model in terms of the SSE of velocity. We can see that the above conclusions also hold for the GHR model. So we can conclude that the proposed DIRECT + SQP algorithm performs better than DIRECT in terms of both global search accuracy and efficiency. It outperforms SQP in global search accuracy with marginally increased calibration time.

With respect to the SSE of gap in IDM and GHR, Tables 3 and 4 show the analogous comparison results with Tables 1 and 2. We assume, if the difference between the optimized objective function and the global optimal value is no larger than

Table 1
Comparison of different algorithms minimizing square velocity errors of the IDM with respect to 100 NGSIM trajectories.

Algorithms	How many search processes had reached the global optimal solution finally (%)	How many objective function evaluations had been tested (on average), if the relative difference between the current best solution and the global optimal solution is less than 1%	How many objective function evaluations had been tested to reach the best solution (on average), if it reaches the global optimal solution finally	Total calibration time of each trajectory (on average), if it reaches the global optimal solution finally (s)
NM	6	197	837	58.9
SQP	92	68	146	1.7
GA	53	815	9330	22.0
SPSA	0	N/A[a]	N/A	N/A
DIRECT	33	377	>10,000	13.1
DIRECT + SQP (1)	98	311	680	2.3

[a] N/A: not applicable.

Table 2
Comparison of different algorithms minimizing square velocity errors of the GHR model with respect to 100 NGSIM trajectories.

Algorithms	How many search processes it had reached the global optimal solution finally (%)	How many objective function evaluations had been tested (on average), if the relative difference between the current best solution and the global optimal solution is less than 1%	How many objective function evaluations had been tested to reach the best solution (on average), if it reaches the global optimal solution finally	Total calibration time of each trajectory (on average), if it reaches the global optimal solution finally (s)
NM	47	227	677	16.3
SQP	42	143	246	2.2
GA	32	275	7691	18.5
SPSA	1	85	>10,000	34.7
DIRECT	6	1246	>10,000	12.9
DIRECT + SQP (1)	92	373	1458	2.3

Table 3
Comparison of different algorithms minimizing square gap errors of the IDM with respect to 100 NGSIM trajectories.

Algorithms	How many search processes had reached the global optimal solution finally (%)	How many objective function evaluations had been tested (on average), if the relative difference between the current best solution and the global optimal solution is less than 1%	How many objective function evaluations had been tested to reach the best solution (on average), if it reaches the global optimal solution finally	Total calibration time of each trajectory (on average), if it reaches the global optimal solution finally (s)
NM	6	786	1134	43.5
SQP	73	111	175	2.1
GA	1	2052	>10,000	23.2
SPSA	0	N/A	N/A	N/A
DIRECT	0	N/A	N/A	N/A
DIRECT + SQP (1)	98	151	209	2.8

10^{-4} [m]^2 for the SSE of gap, it has reached the global optimal solution. These results again verify the proposed new algorithm outperforms other methods both in reliability and efficiency.

5.3. The selection of two configuration parameters

There are two configuration parameters in the proposed algorithm: d_0, the critical size of the PO hyper-rectangles, and κ, the number of PO hyper-rectangles in which we apply the SQP algorithm for local search. Let us first discuss the later one.

Test results indicate that carrying out local search in more PO hyper-rectangles greatly improves the performance of the proposed algorithm. This idea is somewhat similar to the multi-start mechanism used for NM, SQP, GA, and SPSA algorithms, but their effects can be quite different. For existing algorithms, the randomly generated initial starting points may still locate far away from the global optimal solution. In the new algorithm, the starting points locate in the PO hyper-rectangles which have been carefully selected by the DIRECT algorithm. As a result, these points have a much higher probability to locate close to the global optimal solution. So, carrying out local search in only a few PO hyper-rectangles may notably increase the probability of finding the global optimal solution.

Table 4
Comparison of different algorithms minimizing square gap errors of the GHR model with respect to 100 NGSIM trajectories.

Algorithms	How many search processes had reached the global optimal solution finally (%)	How many objective function evaluations had been tested (on average), if the relative difference between the current best solution and the global optimal solution is less than 1%	How many objective function evaluations had been tested to reach the best solution (on average), if it reaches the global optimal solution finally	Total calibration time of each trajectory (on average), if it reaches the global optimal solution finally (s)
NM	25	278	686	22.1
SQP	52	149	202	2.3
GA	19	543	8668	18.2
SPSA	0	N/A	N/A	N/A
DIRECT	6	2172	>10,000	12.1
DIRECT + SQP (1)	86	232	1309	2.0

Table 5
Comparison of the probability of reaching the global optimal solution and corresponding average calibration time of different algorithms.

Algorithms	SSE of velocity				SSE of gap			
	IDM		GHR		IDM		GHR	
	Probability of reaching global optimal solution (%)	Average calibration time of each trajectory (s)	Probability of reaching global optimal solution (%)	Average calibration time of each trajectory (s)	Probability of reaching global optimal solution (%)	Average calibration time of each trajectory (s)	Probability of reaching global optimal solution (%)	Average calibration time of each trajectory (s)
NM[b]	9	188.0	50	49.3	6	136.1	36	77.4
SQP[b]	98	5.0	65	6.5	86	6.4	70	7.1
GA	53	22.0	32	18.5	1	23.2	19	18.2
SPSA[b]	0	N/A[c]	1	104.1	0	N/A	0	N/A
DIRECT[a]	33	13.1	6	12.9	0	N/A	6	12.1
DIRECT +QP (1)[a]	98	2.3	92	2.3	98	2.8	86	2.0
DIRECT +QP (3)[b]	100	2.9	97	2.9	99	4.4	96	2.6

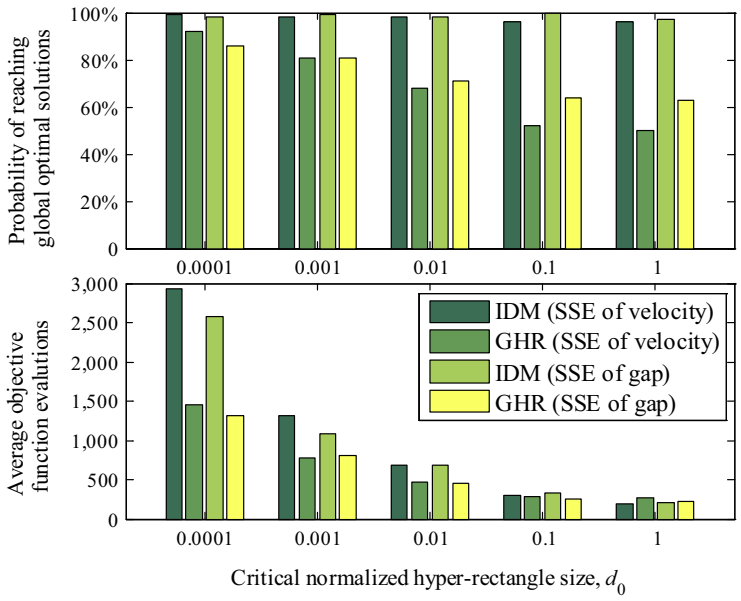
[a] Single-start mechanism (three random initial points).
[b] Multi-start mechanism.
[c] Not applicable.

For a demonstration purpose, we randomly generate three different starting points for NM, SQP, GA, and SPSA algorithms, and choose the best final solution from three candidates. Table 5 summarizes the corresponding calibration results where the DIRECT + SQP (3) algorithm is applied. We can see that multi-start mechanism increases the reliability of NM, SQP, GA, and SPSA, but also increase time costs. The DIRECT + SQP (1) algorithm still gives a higher reliability than NM, SQP, GA, and SPSA algorithms in all of the four scenarios and consumes considerably less time. The DIRECT + SQP (3) algorithm gives even higher reliability and consumes a bit more time than the DIRECT + SQP (1) algorithm. So, if time allowed, we suggest to apply SQP algorithm in more than one PO hyper-rectangles for local search. The detailed value of d_0 can be determined from the calibration results of sample trajectory pairs.

The other configuration parameter, the critical size of normalized hyper-rectangle, can also be easily determined in practice. To explore the sensitivity of the critical hyper-rectangle size d_0 in the proposed DIRECT + SQP algorithm with respect to different objective functions, we let decrease from 1 to 10^{-4} and estimate the algorithm's reliability using the aforementioned 100 pairs of NGSIM trajectories. Fig. 7(a) shows that the probabilities of DIRECT + SQP (1) for finding the global optimal solution increases when d_0 decreases, while the average objective function evaluations to reach global optimal solutions increase quickly. We can see that the reliability of IDM is closed to 100% for both error functions (7) and (8), and the reliability of GHR is over 80% for both error functions when $d_0 < 10^{-3}$. Fig. 7(b) shows the results of DIRECT + SQP (3), it can be seen that the probabilities of reaching global solutions increase given the same d_0 after the three-start mechanism is applied, while the increase computation time is no significant.

In practice, users could adaptively balance the tradeoff between the algorithm reliability and computational costs based on the time budget and the accuracy requirement. As indicated in Table 5, we could usually choose an as small as possible d_0; because even we set $d_0 = 10^{-4}$, the calibration for one trajectory pair could still be finished in less than 10 s.

We have also tested several other calibration objective functions and other car-following models. All of the test results verify that the proposed new algorithm gives better performance than existing algorithms in both models. Constrained by the length limit, we do not present them in this paper.



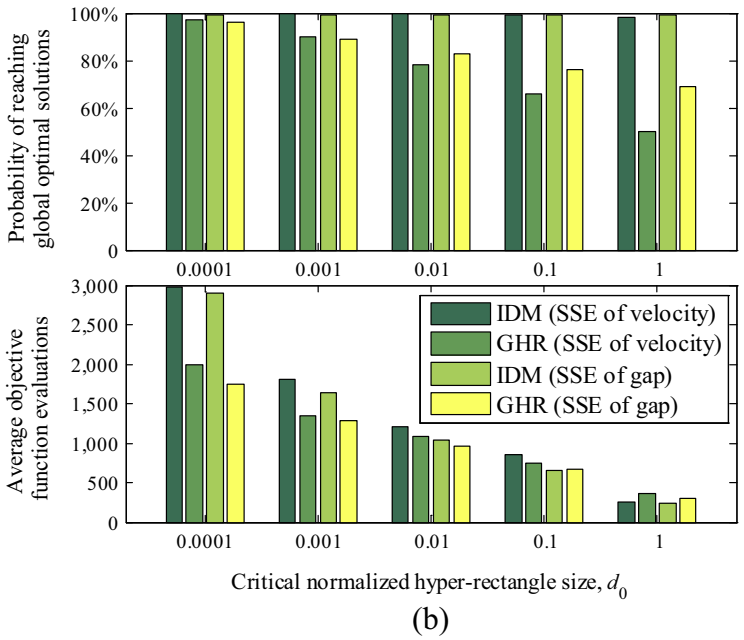


Fig. 7. Sensitivity of d_0 with respect to different car-following models and the number of objective function evaluations had been tested to reach the best solution: (a) DIRECT + SQP (1); (b) DIRECT + SQP (3).

6. Conclusions

Car-following models calibration based on trajectory data has been widely applied in traffic simulation practice. However, there is still a need of thorough discussions on the corresponding optimization algorithms. Many existing algorithms, e.g. Nelder–Mead algorithm, sequential quadratic programming algorithm, genetic algorithm, and simultaneous perturbation stochastic approximation algorithm, require to start from randomly picked-up solutions for a number of times and cannot guarantee to converge to the global optimal solution. Besides, the converging speeds of some algorithms are really slow.

In this paper, we first show that the objective function of the formulated optimization problem is usually Lipschitz continuous on the interested solution space of parameter sets. Using this feature, we can use Lipschitz optimization algorithm to

Table B.1
Comparison of the failed DIRECT + SQP (1) solutions with the global optimal solutions.

Optimization problems	Failed calibrated trajectory pairs	Best solution of DIRECT + SQP (1)	Global optimal solution	Normalized distance of two solutions	Objective function value of the best solution	Objective function value of the global optimal solution	Relative error of objective function values (%)
IDM, 98% (SSE of velocity)	1	[1.68,1.58,21.19,0.55,3.79]	[1.65,1.62,21.28,0.57,3.55]	3.35E−02	50.87	50.86	0.00
	2	[2.59,4.00,21.55,1.27,3.01]	[1.00,1.00,19.92,1.27,1.63]	1.29E+00	44.15	43.86	0.65
GHR, 92% (SSE of velocity)	1	[0.48,0.01,0.02]	[0.45,0.01,0.00]	4.29E−03	106.39	106.36	0.03
	2	[0.53,0.69,0.48]	[0.14,0.55,0.00]	1.72E−01	140.93	138.39	1.83
	3	[452.69,0.27,2.31]	[431.05,0.27,2.30]	4.34E−02	102.07	102.07	0.00
	4	[0.61,0.83,0.55]	[0.58,0.82,0.54]	5.05E−03	206.70	206.70	0.00
	5	[16.70,0.49,1.52]	[5.71,0.22,1.04]	2.81E−01	113.40	111.99	1.26
	6	[0.22,0.83,0.26]	[0.23,0.91,0.33]	8.03E−02	170.75	170.30	0.26
	7	[30.96,0.59,1.71]	[29.24,0.58,1.69]	1.27E−02	58.85	58.84	0.00
	8	[0.60,0.28,0.36]	[0.58,0.27,0.34]	7.97E−03	112.63	112.62	0.01
IDM,98% (SSE of gap)	1	[2.79,1.00,15.36,2.02,1.00]	[2.79,1.00,15.36,2.02,1.00]	3.67E−05	813.72	813.72	0.00
	2	[1.64,4.00,16.99,1.97,1.00]	[1.06,1.00,16.56,1.99,1.00]	1.04E+00	1861.00	1844.27	0.91
GHR,86% (SSE of gap)	1	[157.41,0.95,1.94]	[155.83,0.95,1.94]	5.44E−03	1152.97	1152.81	0.01
	2	[77.66,0.94,2.20]	[74.89,0.94,2.19]	6.60E−03	426.09	426.07	0.00
	3	[2.57,0.00,0.61]	[2.55,0.00,0.61]	4.55E−04	1922.74	1922.74	0.00
	4	[0.36,1.00,0.29]	[0.24,1.00,0.13]	3.07E−02	775.97	775.17	0.23
	5	[35.23,0.63,1.85]	[5.21,0.37,1.16]	2.99E−01	2253.19	1492.80	50.94
	6	[103.25,1.00,2.42]	[2.14,1.00,1.12]	3.30E−01	1776.74	1444.79	22.99
	7	[3.96,0.49,1.02]	[4.16,0.49,1.04]	3.63E−03	803.99	803.61	0.05
	8	[181.58,0.00,2.07]	[317.42,0.00,2.29]	2.75E−01	540.17	515.79	4.73
	9	[0.11,0.64,0.00]	[0.11,0.64,0.00]	1.13E−03	228.64	228.64	0.00
	10	[1.07,0.00,0.15]	[0.70,0.00,0.00]	3.04E−02	2159.51	1989.35	8.55
	11	[35.86,1.00,2.01]	[35.68,1.00,2.01]	4.92E−04	326.10	326.10	0.00
	12	[500.00,1.00,2.86]	[500.00,1.00,2.75]	2.13E−02	1769.09	1707.83	3.59
	13	[57.32,0.85,2.12]	[53.84,0.85,2.10]	7.89E−03	602.72	601.40	0.22
	14	[9.88,0.66,1.44]	[7.33,0.57,1.28]	8.71E−02	221.41	219.99	0.65

quickly determine a few sub-spaces whose sizes are much smaller than the size of the whole solution space. Since the objective function is often flat around the global optimal solution, the final converging speed of the Lipschitz optimization algorithm is very slow. So, we then switch to SQP algorithm to find the global optimal solution within these sub-spaces. The switching condition is determined by the size of PO hyper-rectangles.

Test results show that this new algorithm has both the fastest convergence speed and the highest probability to find the global optimal solution, if being compared with the existing algorithms. Moreover, it has only two major configuration parameters whose values can be easily chosen in practice. We believe this useful algorithm will be widely used for microscopic car-following problems, and can be extensively applied to solve other optimization problems in transportation research.

In the future research, the comparison of the proposed algorithm with other combined global–local algorithms can be conducted.

Acknowledgements

This work was supported in part by National Natural Science Foundation of China 51508505, 51338008, National Science and Technology Support Program 2013BAG18B00 and a Project supported by Tsinghua University under Grant 20131089307.

Appendix A. Potentially optimal hyper-rectangles of DIRECT

DIRECT begins the optimization by transforming the domain of the problem into the unit hyper-rectangle. In each rectangular division, the objective function is evaluated at the points $[\boldsymbol{\theta}]_{11} \pm \mathbf{e}_i/3,\ i = 1,\ldots,n$, where one-third the side-length of the hyper-rectangle is used, and $\mathbf{e}_i$ is a unit vector with a one in the ith position and zeros elsewhere. The algorithm begins its loop of identifying PO hyper-rectangles, which meet the criteria of Definition A.1.

Definition A.1. Let $\varepsilon > 0$ be a positive constant and let $g_{\min}$ be the current best function value. A hyper-rectangle j is said to be potentially optimal if there exists some $\hat{K} > 0$ such that

Table B.2
Comparison of the DIRECT + SQP (3) solutions with the global optimal solutions for the same failed trajectory pairs in Table B.1.

Optimization problems	Failed calibrated trajectory pairs	Best solution of DIRECT + SQP (3)	Global optimal solution	Normalized distance of two solutions	Objective function value of the best solution	Objective function value of the global optimal solution	Relative error of objective function values of the DIRECT + SQP (3) solutions with the global optimal (%)
IDM, 100% (SSE of velocity)	1	[1.65, 1.62, 21.28, 0.57, 3.55]	[1.65, 1.62, 21.28, 0.57, 3.55]	4.17E−07	50.86	50.86	0.00
	2	[1.00, 1.00, 19.92, 1.27, 1.63]	[1.00, 1.00, 19.92, 1.27, 1.63]	3.88E−07	43.48	43.86	0.00
GHR, 97% (SSE of velocity)	1	[0.45, 0.01, 0.00]	[0.45, 0.01, 0.00]	2.51E−03	106.36	106.36	0.00
	2	[0.51, 0.65, 0.42]	[0.14, 0.55, 0.00]	1.33E−01	140.68	138.39	1.66
	3	[431.05, 0.27, 2.30]	[431.05, 0.27, 2.30]	2.50E−07	102.07	102.07	0.00
	4	[0.58, 0.82, 0.54]	[0.58, 0.82, 0.54]	1.33E−06	206.70	206.70	0.00
	5	[5.71, 0.22, 1.04]	[5.71, 0.22, 1.04]	0.00E+00	111.99	111.99	0.00
	6	[0.21, 0.98, 0.37]	[0.23, 0.91, 0.33]	7.64E−02	170.64	170.30	0.20
	7	[29.24, 0.58, 1.69]	[29.24, 0.58, 1.69]	4.06E−05	58.84	58.84	0.00
	8	[0.59, 0.26, 0.36]	[0.58, 0.27, 0.34]	1.43E−02	112.63	112.62	0.00
IDM, 99% (SSE of gap)	1	[2.79, 1.00, 15.36, 2.02, 1.00]	[2.79, 1.00, 15.36, 2.02, 1.00]	1.53E−05	813.72	813.72	0.00
	2	[1.06, 1.00, 16.56, 1.99, 1.00]	[1.06, 1.00, 16.56, 1.99, 1.00]	8.72E−07	1844.27	1844.27	0.91
GHR, 96% (SSE of gap)	1	[157.83, 0.95, 1.94]	[155.83, 0.95, 1.94]	0.00E+00	1152.81	1152.81	0.00
	2	[73.23, 0.94, 2.18]	[74.89, 0.94, 2.19]	3.84E−03	426.07	426.07	0.00
	3	[2.55, 0.00, 0.61]	[2.55, 0.00, 0.61]	9.64E−05	1922.74	1922.74	0.00
	4	[0.24, 1.00, 0.13]	[0.24, 1.00, 0.13]	0.00E+00	775.17	775.17	0.00
	5	[5.21, 0.37, 1.16]	[5.21, 0.37, 1.16]	0.00E+00	1492.80	1492.80	0.00
	6	[2.14, 1.00, 1.12]	[2.14, 1.00, 1.12]	0.00E+00	1444.79	1444.79	0.00
	7	[3.96, 0.49, 1.02]	[4.16, 0.49, 1.04]	3.63E−02	803.99	803.61	0.05
	8	[2.95, 0.00, 2.27]	[317.42, 0.00, 2.29]	2.92E−02	516.38	515.79	0.11
	9	[0.11, 0.64, 0.00]	[0.11, 0.64, 0.00]	1.13E−03	228.64	228.64	0.00
	10	[0.70, 0.00, 0.00]	[0.70, 0.00, 0.00]	8.01E−14	1989.35	1989.35	0.00
	11	[35.66, 1.00, 2.01]	[35.68, 1.00, 2.01]	4.77E−05	326.10	326.10	0.00
	12	[500.00, 1.00, 2.75]	[500.00, 1.00, 2.75]	1.71E−07	1707.83	1707.83	0.00
	13	[57.14, 0.85, 2.12]	[53.84, 0.85, 2.10]	7.48E−03	602.28	601.40	0.15
	14	[8.60, 0.62, 1.36]	[7.33, 0.57, 1.28]	4.74E−02	220.91	219.99	0.42

$$g(\boldsymbol{\theta}_j) - \hat{K}d_j \leqslant g(\boldsymbol{\theta}_i) - \hat{K}d_i, \quad \forall i, \quad \text{and} \tag{A.1}$$

$$g(\boldsymbol{\theta}_j) - \hat{K}d_j \leqslant g_{\min} - \varepsilon|g_{\min}| \tag{A.2}$$

where $\boldsymbol{\theta}_j$ is the center of hyper-rectangle j, d_j is the distance from $\boldsymbol{\theta}_j$ to its vertices.

If hyper-rectangle j is potentially optimal, then $g(\boldsymbol{\theta}_j) \leqslant g(\boldsymbol{\theta}_i)$ for all hyper-rectangles that are of the same size as j (i.e. $d_j = d_i$).

If $d_j \geqslant d_k$, for all k hyper-rectangles, and $g(\boldsymbol{\theta}_j) \leqslant g(\boldsymbol{\theta}_i)$ for all hyper-rectangles such that $d_j = d_i$, then hyper-rectangle j is potentially optimal.

If $d_j \leqslant d_k$ for all k hyper-rectangles, and j is potentially optimal, then $g(\boldsymbol{\theta}_j) = g_{\min}$.

An explicit way of implementing Definition A.1 can be found in Gablonsky (2001). Let I be the set of all indices of all hyper-rectangles existing. Let $I_1 = \{i \in I : d_i < d_j\}$, $I_2 = \{i \in I : d_i > d_j\}$, $I_3 = \{i \in I : d_i = d_j\}$. Hyper-rectangle $j \in J$ is potentially optimal if

$$g(\boldsymbol{\theta}_j) \leqslant g(\boldsymbol{\theta}_i), \quad \forall i \in I_3 \tag{A.3}$$

there exists $\hat{K} > 0$ such that

$$\max_{i \in I_1} \frac{g(\boldsymbol{\theta}_j) - g(\boldsymbol{\theta}_i)}{d_j - d_i} \leqslant \hat{K} \leqslant \max_{i \in I_2} \frac{g(\boldsymbol{\theta}_i) - g(\boldsymbol{\theta}_j)}{d_i - d_j} \tag{A.4}$$

and

$$\varepsilon \leqslant \frac{g_{\min} - g(\boldsymbol{\theta}_j)}{|g_{\min}|} + \frac{d_j}{|g_{\min}|} \min_{i \in I_2} \frac{g(\boldsymbol{\theta}_i) - g(\boldsymbol{\theta}_j)}{d_i - d_j}, \quad g_{\min} \neq 0 \tag{A.5}$$

or

$$g(\boldsymbol{\theta}_j) \leqslant d_j \min_{i \in I_2} \frac{g(\boldsymbol{\theta}_i) - g(\boldsymbol{\theta}_j)}{d_i - d_j}, \quad g_{\min} = 0 \tag{A.6}$$

Once a hyper-rectangle has been identified as potentially optimal, DIRECT divides this hyper-rectangle along its longest dimension(s) to ensure shrinking on every dimension. If the hyper-rectangle is a hyper-cube, then the divisions will be done along all sides; or dividing the potentially optimal hyper-rectangle along all dimensions of its maximal length, i.e. one-third the length of the maximum side of the potentially optimal hyper-rectangle. We refer interested reader to Finkel (2003) for more detailed explanations and some illustrative numerical examples solved by DIRECT.

Appendix B. Zoom in failed calibrated trajectories

Table B.1 lists the calibration results for those trajectory pairs that the DIRECT + SQP (1) algorithm failed to find the global optimal solution. Here, we use the normalized Euclidean distance to measure the difference between the best solution given by the DIRECT + SQP (1) algorithm and the global optimal solution. Results show that only 6 normalized distances out of the 26 failed calibrations are larger than 0.1; while the other 20 normalized distances are smaller than 0.1. This indicates that even if the DIRECT + SQP (1) algorithm cannot find the best solution in a given time, it will usually stop at a solution that is close to the global optimal solution. The failure of final convergence is mainly caused by the approximation errors of the SQP algorithm, when it tries to approximate the original optimization objective as a quadratic one.

Table B.2 shows the DIRECT + SQP (3) solutions for the same failed trajectory pairs in Table B.1. Results show that the only one normalized distances out of the 26 failed calibrations is larger than 0.1. Moreover, the distances between the found solutions and the global optimal solutions are significantly reduced, too.

Appendix C. Supplementary material

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.trc.2016.04.011.

References

Ahn, S., Cassidy, M.J., Laval, J., 2004. Verification of a simplified car-following theory. Transp. Res. Part B: Methodol. 38 (5), 431–440.

Barceló, J. (Ed.), 2010. Fundamentals of Traffic Simulation. Springer.

Brockfeld, E., Kühne, R.D., Skabardonis, A., Wagner, P., 2003. Toward benchmarking of microscopic traffic flow models. Transp. Res. Rec. 1852, 124–129.

Brockfeld, E., Kühne, R.D., Wagner, P., 2005. Calibration and validation of microscopic models of traffic flow. Transp. Res. Rec. 1934, 179–187.

Chen, D., Laval, J., Zheng, Z., Ahn, S., 2012. A behavioral car-following model that captures traffic oscillations. Transp. Res. Part B: Methodol. 46 (6), 744–761.

Chen, X., Li, L., Zhang, Y., 2010. A Markov model for headway/spacing distribution of road traffic. IEEE Trans. Intell. Transp. Syst. 11 (4), 773–785.

Chen, X., Li, Z., Li, L., Shi, Q., 2013. Characterising scattering features in flow-density plots using a stochastic platoon model. Transportmet. A: Transp. Sci. http://dx.doi.org/10.1080/23249935.2013.822941.

Chen, X., Li, L., Shi, Q., 2015. Stochastic Evolutions of Dynamic Traffic Flow: Modeling and Applications. Springer, Heidelberg.

Chiabaut, N., Leclercq, L., Buisson, C., 2010. From heterogeneous drivers to macroscopic patterns in congestion. Transp. Res. Part B: Methodol. 44 (2), 299–308.

Chung, E., Dumont, A.-G. (Eds.), 2009. Transportation Simulation: Beyond Traditional Approaches. CRC Press.

Ciuffo, B., Punzo, V., 2014. No Free Lunch theorems applied to the calibration of traffic simulation models. IEEE Trans. Intell. Transp. Syst. 15 (2), 553–562.

Ciuffo, B., Punzo, V., 2010. Verification of traffic microsimulation model calibration procedures: analysis of goodness-of-fit measures. Presented at 89th Annual Meeting of the Transportation Research Board, Washington, D.C.

Ciuffo, B., Punzo, V., Montanino, M., 2012a. The Calibration of Traffic Simulation Models: Report on the Assessment of Different Goodness of Fit Measures and Optimization Algorithms. MULTITUDE Project: COST Action TU0903. JRC68403. European Union. <http://publications.jrc.ec.europa.eu/repository>.

Ciuffo, B., Punzo, V., Montanino, M., 2012b. Thirty years of Gipps' car-following model. Transp. Res. Rec. 2315, 89–99.

Conn, A.R., Scheinberg, K., Vicente, L.N., 2009. Introduction to Derivative-free Optimization. SIAM, Philadelphia.

Finkel, D.E., 2003. DIRECT Optimization Algorithm user Guide. Center for Research in Scientific Computation, North Carolina State University.

Fletcher, R., Gould, N.I., Leyffer, S., Toint, P.L., Wächter, A., 2002. Global convergence of a trust-region SQP-filter algorithm for general nonlinear programming. SIAM J. Optim. 13 (3), 635–659.

Fletcher, R., Powell, M.J.D., 1963. A rapidly convergent descent method for minimization. Comput. J. 6, 163–168.

Gablonsky, J.M., 2001. Modifications of the Direct Algorithm (Ph.D. thesis). North Carolina State University, Raleigh, North Carolina.

Gazis, D.C., Herman, R., Rothery, R.W., 1961. Nonlinear follow-the-leader models of traffic flow. Oper. Res. 9, 545–567.

Gipps, P.G., 1981. A behavioural car following model for computer simulation. Transp. Res. B: Methodol. 15, 105–111.

Goldberg, D., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Professional, Reading, MA.

Gunay, B., 2007. Car following theory with lateral discomfort. Transp. Res. Part B: Methodol. 41 (7), 722–735.

Hollander, Y., Liu, R., 2008. The principles of calibrating traffic microsimulation models. Transportation 35, 347–362.

Hoogendoorn, S.P., Hoogendoorn, R., 2010. Generic calibration framework for joint estimation of car-following models by using microscopic data. Transp. Res. Rec. 2188, 37–45.

Hoogendoorn, S.P., Ossen, S., 2005. Parameter estimation and analysis of car-following models. In: Mahmassani, H. (Ed.), Proceedings of the 16th International Symposium on Traffic and Transportation Theory. Emerald Group Publishing Limited.

Jones, D.R., Perttunen, C.D., Stuckman, B.E., 1993. Lipschitzian optimization without the Lipschitz constant. J. Optim. Theory Appl. 79, 157–181.

Kesting, A., Treiber, M., 2008. Calibrating car-following models by using trajectory data methodological study. Transp. Res. Rec. 2088, 148–156.

Kim, I., Kim, T., Sohn, K., 2013. Identifying driver heterogeneity in car-following based on a random coefficient model. Transp. Res. Part C: Emerg. Technol. 36, 35–44.

Kolda, T.G., Lewis, R.M., Torczon, V., 2003. Optimization by direct search: new perspectives on some classical and modern methods. SIAM Rev. 45 (3), 385–482.

Koutsopoulos, H.N., Farah, H., 2012. Latent class model for car following behavior. Transp. Res. Part B: Methodol. 49, 563–578.

Lagarias, J.C., Reeds, J.A., Wright, M.H., Wright, P.E., 1998. Convergence properties of the Nelder–Mead simplex method in low dimensions. SIAM J. Optim. 9 (1), 112–147.
Lee, J.-B., Ozbay, K., 2009. New calibration methodology for microscopic traffic simulation using enhanced simultaneous perturbation stochastic approximation approach. Transp. Res. Rec. 2124, 233–240.
Li, L., Chen, X., Li, Z., 2013a. Asymmetric stochastic Tau Theory in car-following. Transp. Res. Part F: Traffic Psychol. Behav. 18, 21–33.
Li, L., Chen, X., Li, Z., Zhang, L., 2013b. Freeway travel time estimation using temporal–spatial queueing model. IEEE Trans. Intell. Transp. Syst. 14 (3), 1536–1541.
Ma, T., Abdulhai, B., 2002. Genetic algorithm-based optimization approach and generic tool for calibrating traffic microscopic simulation parameters. Transp. Res. Rec. 1800, 6–15.
Ma, J., Dong, H., Zhang, H.M., 2007. Calibration of microsimulation with heuristic optimization methods. Transp. Res. Rec. 1999, 208–217.
Nelder, J.A., Mead, R., 1965. A simplex method for function minimization. Comput. J. 7, 308–313.
Nocedal, J., Wright, S.J., 2006. Numerical Optimization, third ed. Springer.
Ossen, S., Hoogendoorn, S.P., 2005. Car-following behavior analysis from microscopic trajectory data. Transp. Res. Rec. 1934, 13–21.
Ossen, S., Hoogendoorn, S.P., 2007. Driver heterogeneity in car following and its impact on modeling traffic dynamics. Transp. Res. Rec. 1999, 95–103.
Ossen, S., Hoogendoorn, S.P., 2008. Validity of trajectory-based calibration approach of car-following models in presence of measurement errors. Transp. Res. Rec. 2088, 117–125.
Ossen, S., Hoogendoorn, S.P., 2009. Reliability of parameter values estimated using trajectory observations. Transp. Res. Rec. 2124, 36–44.
Ossen, S., Hoogendoorn, S.P., 2011. Heterogeneity in car-following behavior: theory and empirics. Transp. Res. Part C: Emerg. Technol. 19 (2), 182–195.
Panwai, S., Dia, H., 2005. Comparative evaluation of microscopic car-following behavior. IEEE Trans. Intell. Transp. Syst. 6 (3), 314–325.
Park, B., Qi, H., 2005. Development and evaluation of a procedure for the calibration of simulation models. Transp. Res. Rec. 1934, 208–217.
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., Ciuffo, B., Quaglietta, E., 2011. Kriging meta-modeling to verify traffic microsimulation calibration methods. In: Presented at 90th Annual Meeting of the Transportation Research Board, Washington, D.C.
Punzo, V., Simonelli, F., 2005. Analysis and comparison of microscopic traffic flow models with real traffic microscopic data. Transp. Res. Rec. 1934, 53–63.
Punzo, V., Tripodi, A., 2007. Steady-state solutions and multiclass calibration of Gipps microscopic traffic flow model. Transp. Res. Rec. 1999, 104–114.
Rakha, H., Gao, Y., 2011. Calibration of steady-state car-following models using macroscopic loop detector data. In: Proceedings of 75 Years of the Fundamental Diagram for Traffic Flow Theory: Greenshields Symposium, pp. 178–198.
Rakha, H., Pecker, C.C., Cybis, H.B.B., 2007. Calibration procedure for Gipps car-following model. Transp. Res. Rec. 1999, 115–127.
Rios, J.M., Sahinidis, N.V., 2013. Derivative-free optimization: a review of algorithms and comparison of software implementations. J. Global Optim. 56, 1247–1293.
Schultz, G.G., Rilett, L.R., 2004. Analysis of distribution and calibration of car-following sensitivity parameters in microscopic traffic simulation models. Transp. Res. Rec. 1876, 41–51.
Spall, J.C., 2000. Adaptive stochastic approximation by the simultaneous perturbation method. IEEE Trans. Autom. Control 45 (10), 1839–1853.
Spall, J.C., 2003. Introduction to Stochastic Search and Optimization: Estimation, Simulation and Control. Wiley, Hoboken, NJ, USA.
Thiemann, C., Treiber, M., Kesting, N., 2008. Estimating acceleration and lane-changing dynamics from next generation simulation trajectory data. Transp. Res. Rec. 2088, 90–101.
Tordeux, A., Lassarre, S., Roussignol, M., 2010. An adaptive time gap car-following model. Transp. Res. Part B: Methodol. 44 (8), 1115–1131.
Treiber, M., Hennecke, A., Helbing, D., 2000. Congested traffic states in empirical observations and microscopic simulations. Phys. Rev. E 62 (2), 1805–1824.
Treiber, M., Kesting, A., 2013a. Microscopic calibration and validation of car-following models – a systematic approach. Procedia – Soc. Behav. Sci. 80, 922–939.
Treiber, M., Kesting, A., 2013b. Traffic Flow Dynamics: Data, Models and Simulation. Springer-Verlag, Berlin Heidelberg.
Wagner, P., 2012. Analyzing fluctuation in car-following. Transp. Res. Part B: Methodol. 46, 1384–1392.
Wagner, P., Nippold, R., Toledo, T., 2010. Calibration by acceleration or by trajectory (Run it!). In: TRB Traffic Flow Theory Committee (TFTC) Committee Summer Meeting and Conference.
Wang, H., Wang, W., Chen, J., Jing, M., 2010. Using trajectory data to analyze intradriver heterogeneity in car-following. Transp. Res. Rec. 2188, 85–95.
Wei, D., Liu, H., 2013. Analysis of asymmetric driving behavior using a self-learning approach. Transp. Res. Part B: Methodol. 47, 1–14.
Wilson, E.R., 2001. An analysis of Gipps' car-following model of highway traffic. J. Appl. Math. 66, 509–537.
Wu, J., Brackstone, M., McDonald, M., 2003. The validation of a microscopic simulation model: a methodological case study. Transp. Res. Part C: Emerg. Technol. 11 (6), 463–479.
Zhang, H.M., Kim, T., 2005. A car-following theory for multiphase vehicular traffic flow. Transp. Res. Part B: Methodol. 39 (5), 385–399.