Hindawi
Journal of Advanced Transportation
Volume 2021, Article ID 5597130, 15 pages
https://doi.org/10.1155/2021/5597130

Research Article

Calibrating Path Choices and Train Capacities for Urban Rail Transit Simulation Models Using Smart Card and Train Movement Data

Baichuan Mo,[1] **Zhenliang Ma**,[2] **Haris N. Koutsopoulos**,[3] **and Jinhua Zhao**[4]

[1]*Department of Civil and Environmental Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139, USA*
[2]*Department of Civil Engineering, Monash University, Melbourne, VIC 3800, Australia*
[3]*Department of Civil and Environmental Engineering, Northeastern University, Boston, MA 02115, USA*
[4]*Department of Urban Studies and Planning, Massachusetts Institute of Technology, Cambridge, MA 20139, USA*

Correspondence should be addressed to Zhenliang Ma; mike.ma@monash.edu

Received 10 January 2021; Revised 11 February 2021; Accepted 15 February 2021; Published 28 February 2021

Academic Editor: Erfan Hassannayebi

Transit network simulation models are often used for performance and retrospective analysis of urban rail systems, taking advantage of the availability of extensive automated fare collection (AFC) and automated vehicle location (AVL) data. Important inputs to such models, in addition to origin-destination flows, include passenger path choices and train capacity. Train capacity, which has often been overlooked in the literature, is an important input that exhibits a lot of variabilities. The paper proposes a simulation-based optimization (SBO) framework to simultaneously calibrate path choices and train capacity for urban rail systems using AFC and AVL data. The calibration is formulated as an optimization problem with a black-box objective function. Seven algorithms from four branches of SBO solving methods are evaluated. The algorithms are evaluated using an experimental design that includes five scenarios, representing different degrees of path choice randomness and crowding sensitivity. Data from the Hong Kong Mass Transit Railway (MTR) system is used as a case study. The data is used to generate synthetic observations used as "ground truth." The results show that the response surface methods (particularly constrained optimization using response surfaces) have consistently good performance under all scenarios. The proposed approach drives large-scale simulation applications for monitoring and planning.

1. Introduction

Urban rail systems are important components of the urban transportation system. Given their high reliability and large capacity, they have attracted high passenger demand. However, high demand also leads to problems such as overcrowding and disruptions, which decrease the level of service and impact passengers. To maintain service reliability and develop efficient response strategies, it is crucial for operators to better understand passenger demand and flow patterns in the network.

Transit network loading (or simulation) models for metro systems, powered by automated collected data, provide a useful instrument for network performance monitoring. They enable operators to characterize the level of service and make decisions accordingly. A typical network loading model requires origin-destination (OD) matrix, supply information, and path choice fractions as input. The supply information includes the transit network topology, actual vehicle movement data, and vehicle capacity. Thanks to the wide deployment of automated fare collection (AFC) and automated vehicle location (AVL) systems, the OD demand and train movement data can be directly obtained. However, obtaining the corresponding path choices and quantifying reasonable vehicle capacity remains a challenge. According to Liu et al. [1] and Preston et al. [2], train capacity, defined as the maximum train load when remaining passengers in the platform denied boarding, may vary

depending on the crowding levels in trains and on platforms and passenger attitudes. The calibration of path choices and train capacity can improve the accuracy of network loading models for performance monitoring. Thus, these models can provide better information to operators to adjust operating strategies, relieve congestion, and improve efficiency.

Traditionally, path choices are inferred with data from on-site surveys that are used to estimate path choice models. However, surveys are time-consuming and labor-intensive, limiting their real-world usage. To overcome these disadvantages, path choice estimation methods based on AFC data have been proposed in the literature. AFC systems provide the exact locations and times of passengers' entry and exit transactions, which can be used to extract OD demand and passengers' journey times. They provide rich information for analyzing passenger behavior [3].

In an urban rail system operated near its capacity, five critical parameters are correlated with each other: OD demand, journey time, left behind (or denied boarding), path choices, and train capacity. The relationship of these parameters can be explained in Figure 1. OD demand is the input and journey time is the output (OD exit flow is a combination of the two), which can both be observed from the AFC data. Path choices, train capacity, and left behind are not observable in the AFC data. Journey time is directly affected by path choices and left behind (left behind can increase the waiting time). Left behind is directly affected by path choices and train capacity. This figure indicates the complexity of path choice estimation using AFC data. The dependencies of different parameters (e.g., path choices vs. train capacity) should be captured.

In the context of path choice estimation, the AFC data-based methods can be categorized into two groups: path-identification methods [4–7] and parameter-inference methods [8–12]. The former studies aim to identify the exact path chosen by each user and even the train they boarded. Path attributes are used to evaluate how likely a path is chosen for a passenger's trip from their observed origins to their observed destinations. The latter studies formulate probabilistic models to describe passengers' decision-making behavior. Bayesian inference is usually used to estimate the corresponding parameters and thus derive the path choice fractions. Despite using different methods, the key components for those AFC data-based studies are similar. They all attempt to match the model-derived journey times with the observed journey times from AFC data. However, many of these studies either assume a known constant train capacity or specify a known link-impedance function. As shown in Figure 1, journey times depend on both path choices and train capacity. An unreasonable setting of train capacity may cause calibration bias of path choices. Simultaneous calibration of both parameters is more reasonable.

Train capacity is a vague concept. Normally trains may not reach their designed physical capacity for various reasons (e.g., passengers may decide not to board due to the crowding Liu et al. [1]. Therefore, assuming a fixed physical capacity or fixed link-impedance function (in many previous studies) may not be a reasonable assumption in real-world

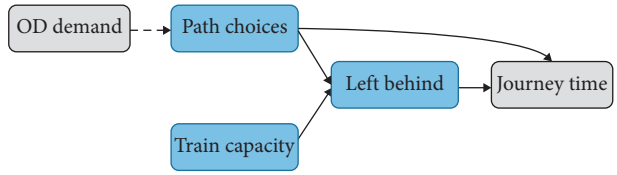


Figure 1: Relationship among critical parameters in urban rail systems.

situations, only a few studies have explored the calibration of actual train capacity in the rail system. Liu et al. [1] proposed the concept of "willingness to board" (WTB) to describe the varied capacity in a bus system and estimated passengers' WTB using a least square method. Xu and Yong proposed a passenger boarding model which revealed that the number of actually boarding passengers in a crowded train was closely related to the number of queuing passengers and train load. Mo et al. [13] proposed an effective capacity model that recognized train capacity may vary across stations depending on the corresponding number of queuing passengers and train load. The calibration of train capacity or WTB usually requires the AFC data with passengers' boarding and journey time information. However, this information may also be affected by path choices, which were neglected in previous studies.

To fill these research gaps, we propose a simulation-based optimization (SBO) model to calibrate path choices and train capacity simultaneously and also explore the efficiency of typical SBO solution algorithms. The calibration problem is formulated as an optimization problem using AFC and AVL data. The formulation can capture the interaction among these variables and their impact on journey times. Seven optimizers (solving algorithms) from four branches of SBO-solving methods are implemented for comparative analysis. They include generic algorithm (GA), simulated annealing (SA), Nelder–Mead simplex algorithm (NMSA), mesh adaptive direct search (MADS), simultaneous perturbation stochastic approximation (SPSA), Bayesian optimization (BYO), and constrained optimization using response surfaces (CORS). We compare these SBO solving algorithms within a limited computational budget, defined by the number of function evaluations. Data from the Hong Kong Mass Transit Railway (MTR) system provide the foundation for a realistic case study. The major contribution of this paper is twofold:

(i) Proposing an optimization model to simultaneously estimate path choices and train capacities using AFC and AVL data, it addresses the typical assumption of fixed and known train capacities in existing path choice estimation studies using smart card data

(ii) Validating the model using a busy urban rail network and analyzing the performance of SBO solution algorithms using systematic experiments, it represents different degrees of users' randomness in path choice and their sensitivity to crowding

The remainder of the paper is organized as follows. In Section 2, we illustrate the SBO problem formulation.

Section 3 briefly describes the various SBO methods used in this study. The proposed framework is used in a case study with data from the Hong Kong MTR network in Section 4. The results are used to compare the performance of different algorithms. Section 5 concludes the paper by summarizing the main findings and discussing future research directions.

2. Methodology

The paper aims to calibrate simultaneously the train capacity and path choices using readily available data in the closed fare payment systems (require ticket validation at both tap-in and tap-out stations). To capture the interaction among different variables in Figure 1, we use a schedule-based network loading model with capacity constraints (described in Section 2.1). It outputs a list of performance metrics given a set of inputs including OD demand, timetables/AVL, network, train capacity, and path choices. The calibration of path choice and capacity is formulated as an optimization model that attempts to minimize the error between network loading model outputs (e.g., journey time, which is a function of path choices and train capacity) and the corresponding quantities directly observed from the AFC data.

2.1. Transit Network Loading Model. Transit network loading (TNL) models aim to assign passengers over a transit network given the (dynamic) OD entry demand and path choices. In this study, we adopt an event-driven schedule-based TNL model proposed by Mo et al. [13]. The model takes OD entry demand (number of tap-in passengers by time), path choices, train arrival and departure times from stations, train capacity, and infrastructure information (e.g., network topology) as inputs and outputs the passengers' tap-out times, train loads, waiting times, and other network performance indicators of interest.

Figure 2 illustrates the main functions of the TNL model [13]. Three objects are defined: train, waiting queue (on the platform), and passengers. An event is defined as a train arrival at, or departure from, a station. Events are ordered chronologically. New and transferring passengers join the waiting queue on the platform and board a train based on a first-come-first-board (FIFB) discipline. The number of successfully boarding passengers depends on the available train capacity.

The TNL model works by generating a train event list (arrivals and departures) based on the actual train movement data (AVL) and then sequentially processing the ordered events until all events are processed for the time period of interest. The processing of an individual event is based on the following rules:

(i) If the event is an arrival (Figure 2(a)), the train offloads passengers and updates its state (e.g., train load and in-vehicle passengers). Alighting passengers who need to transfer are assigned to the waiting queues on the corresponding transfer platforms (e.g., passengers transferring to platform B in Figure 2(a)). Passengers who tap out will be removed from the system. New tap-in passengers who entered the station between two events are added into the queue (e.g., new tap-in passengers in platform A in Figure 2(a)). Then, the waiting queue objects for all platforms are updated accordingly.

(ii) If the event is a departure (Figure 2(b)), passengers board trains based on a FIFB priority rule. If the on-board passengers reached the train capacity, the remaining passengers at the platform will be denied boarding and wait for the next available train. Finally, the state of the train (train load and in-vehicle passengers) and the waiting queue at the platform are updated accordingly.

More specifically, for each passenger in the simulation model, we first calculate his/her probability of choosing each available path based on the path's attributes and path choice parameters (see Section 2.2, for details). Path attributes include in-vehicle time, number of transfers, and transfer walking time. Then, each passenger is assigned with a specific path based on the choice probability. Based on the path information, the passenger walks to a specific platform, joins the waiting queue, and waits for available trains to board. The boarding and alighting behavior are as described above.

2.2. Problem Formulation. Consider a general urban rail network in a specific time period T, represented as $G = (S, A)$, where S is the set of stations and A is the set of directed links. We divide T into several time intervals with equal length τ (e.g., $\tau = 15$ min). Denote the set of all time intervals as $\mathcal{T} = \{1, 2, \ldots, T/\tau\}$. Define a *time-space (TS) node* as i_m, where $i \in S$ and $m \in \mathcal{T}$. i_m represents station i in time interval m.

For an OD pair (i, j) $(i, j \in S)$, the *OD entry flow* $(q^{i_m,j})$ represents the number of passengers entering station i during time interval m and exiting at station j. Let the set of all OD entry flows be $\mathbf{q}^{\mathbf{e}}$. The *OD exit flow* (q^{i,j_n}) represents the number of passengers who exit at station j in the time interval n with origin i. $q^{i_m,j}$ and q^{i,j_n} are inputs and outputs of the TNL model, respectively.

Let the set of all paths between (i, j) be $\mathcal{R}(i, j)$. We assume that the path choice behavior can be formulated as a C-logit model [14], which is an extension of the multinomial logit (MNL) model to correct the correlation among paths due to overlapping [15]. The path choice fraction for path $r \in \mathcal{R}(i, j)$ in time interval m $(p_r^{i_m,j})$ is formulated as follows:

$$p_r^{i_m,j} = \frac{e^{\mu\left(\beta_X \cdot X_{r,m} + \beta_{CF} \cdot CF_r\right)}}{\sum_{r' \in \mathcal{R}(i,j)} e^{\mu\left(\beta_X \cdot X_{r',m} + \beta_{CF} \cdot CF_{r'}\right)}}, \quad \forall r \in \mathcal{R}(i, j), m \in \mathcal{T}, i, j \in S, \tag{1}$$

where μ is the scale parameter of the Gumbel distribution of the error term [16], which is usually normalized to 1. Larger (smaller) μ means the choice behavior is more deterministic (random). $X_{r,m}$ is the vector of attributes for path r in time interval m (e.g., in-vehicle time, number of transfers, and transfer walking time). CF_r is the commonality factor of path r which measures the degree of similarity of path r with

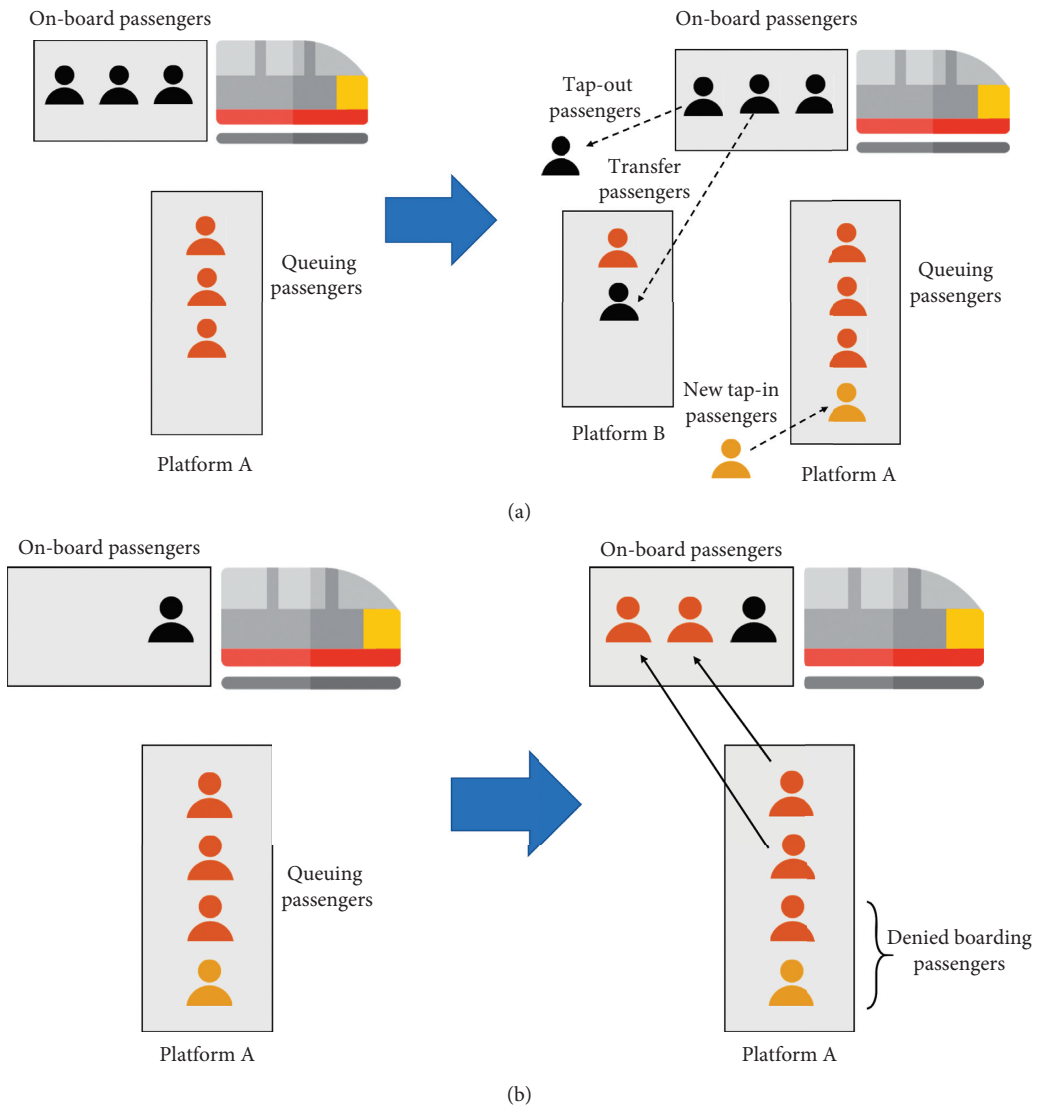


Figure 2: Main functions of the event-based transit network loading model. (a) Train arrival. (b) Train departure.

the other paths of the same OD. β_X and β_{CF} are the corresponding coefficients to be estimated. Let β be the vector that combines β_X and β_{CF} (i.e., $\beta = [\beta_X, \beta_{CF}]$).

CF_r is defined as follows:

$$CF_r = \ln \sum_{r' \in \mathcal{R}(i,j)} \left(\frac{D_{r,r'}}{D_r D_{r'}} \right)^{\gamma}, \tag{2}$$

where $D_{r,r'}$ is the number of common stations of path r and r', D_r and $D_{r'}$ are the number of stations for path r and r', respectively, and γ is a fixed positive parameter. Let the set of all path choice fractions be $\mathbf{p}$.

The values of β can be bounded from above and below. The boundaries can be obtained from the prior knowledge and previous survey results. Denote the upper bound as U_β and lower bound as L_β ($L_\beta \le \beta \le U_\beta$), where U_β and L_β are both vectors with the same cardinality as β.

According to Mo et al. [13], the actual train capacity utilized by passengers is determined by three factors: (a) waiting passenger distribution on the platform, (b) train load and distribution across the train, and (c) passengers' willingness to board a crowded train. Thus, train capacity is not constant. Instead, it is dynamic and changes across stations and trains depending on the crowding level of the train and the platform. Mo et al. [13] model the capacity of train k at station i ($C_{k,i}$) is as

$$C_{k,i} = \begin{cases} \theta_0 n_i + \theta_1 H_{k,i} + \theta_2 Q_{k,i} & \text{if station \$i\$ is in the list of congested stations,} \\ \theta_0 n_i & \text{otherwise,} \end{cases} \quad \forall k, i, \tag{3}$$

where n_i is the number of cars of train i, $H_{k,i}$ is the load of train k when it arrives at station i, $Q_{k,i}$ is the number of passengers waiting on the platform when train k arrives at station i, and θ_0, θ_1, and θ_2 are parameters to be estimated ($\theta_0, \theta_1, \theta_2 > 0$). Specifically, θ_0 is a measure of service standard. $\theta_0 n_i$ can be seen as the base capacity, that is, the train load that represents acceptable service standards. At uncongested stations, passengers are assumed not to board when the train load is greater than $\theta_0 n_i$. At congested stations, passengers may still board a train even if it is already crowded [1], which makes the effective train capacity higher than $\theta_0 n_i$. θ_1 captures the effect that the effective capacity is higher when train load is higher. This is because passengers may worry if they did not board this crowded train, and they cannot board the following trains as well [1]. θ_2 captures the effect that more waiting passengers at the platform may push more passengers to board, leading to higher effective capacity.

In the discussion that follows, let θ be the vector of these three parameters. We assume that the values that these parameters can take is $L_\theta \le \theta \le U_\theta$, where L_θ and U_θ are the corresponding lower and upper bounds, respectively.

The goal is to calibrate θ and β vectors (used by the TNL model) based on indirect observations. Two sets of observations are used for the calibration: observed OD exit flows and observed journey time distribution (JTD). Both of them can be obtained from the AFC data.

Let the ground truth (observed) OD exit flow be $\tilde{q}^{i,j_n}$. Let $f_{i,j_t}(x)$ be the model-derived JTD of passengers with origin i who exit at station j during time interval t. Let $\tilde{f}_{i,j_t}(x)$ be the corresponding observed JTD extracted from the AFC data. Since $f_{i,j_t}(x)$ and $\tilde{f}_{i,j_t}(x)$ are estimated from passengers' journey time observations, only the OD pairs with more than E passengers exiting in a specific time interval are considered, where E is a predetermined threshold to ensure enough sample size. Denote the set of corresponding OD pairs and exit time intervals as $\mathscr{E}$, where $\mathscr{E} = \{(i, j_n): \tilde{q}^{i,j_n}, q^{i,j_n} > E, \quad \forall i, j \in S, n \in \mathscr{T}\}$.

The calibration problem is formulated as an optimization problem:

$$\min_{\beta,\theta} w_1 \sum_{i,j\in S, m\in\mathscr{T}} \left(q^{i,j_n} - \tilde{q}^{i,j_n}\right)^2 + w_2 \sum_{(i,j_n)\in \mathrm{E}} D_{\mathrm{KL}}\left(\left\| f_{i,j_n} \tilde{f}_{i,j_n}\right.\right), \tag{4a}$$

$$q^{i,j_n} = \mathrm{TNL}\left(\mathbf{p}, \mathbf{q}^{\mathbf{e}}, \theta\right) \quad \forall i, j \in S, m \in \mathscr{T}, \tag{4b}$$

$$f_{i,j_n}(x) = \mathrm{TNL}\left(\mathbf{p}, \mathbf{q}^{\mathbf{e}}, \theta 2\right) \quad \forall (i, j_n) \in \mathscr{E}, \tag{4c}$$

$$p_r^{i_m,j} = \frac{e^{\mu\left(\beta_X \cdot X_{r,m} + \beta_{CF} \cdot CF_r\right)}}{\sum_{r' \in \mathscr{R}(i,j)} e^{\mu\left(\beta_X \cdot X_{r',m} + \beta_{CF} \cdot CF_{r'}\right)}} \quad \forall p_r^{i_m,j} \in \mathbf{p}, \tag{4d}$$

$$L_\beta \le \beta \le U_\beta, \tag{4e}$$

$$L_\theta \le \theta \le U_\theta. \tag{4f}$$

The objective function (equation (4)) has two parts: the square error between model-derived OD exit flows and the corresponding observations and the difference between model-derived and observed JTD. w_1 and w_2 are weights used to balance the scale and the importance of the two parts. The difference of the two distributions is expressed using Kullback–Leibler (KL) divergence (D_{KL}):

$$D_{\mathrm{KL}}\left(\left\| f_{i,j_n} \tilde{f}_{i,j_n}\right.\right) = \int_x f_{i,j_n}(x) \cdot \log \frac{f_{i,j_n}(x)}{\tilde{f}_{i,j_n}(x)} \mathrm{d}x. \tag{5}$$

$\mathrm{TNL}(\mathbf{p}, \mathbf{q}^{\mathbf{e}}, \theta)$ is the black-box function that corresponds to the TNL model, which can output the model-derived OD exit flows and JTD for a given set of path choices and train capacity. Since the TNL model has no analytic form, equation (4) is a SBO problem with upper and lower bound constraints. In the following section, we discuss seven different algorithms appropriate for the solution of SBO problems. These algorithms belong to four general approaches of SBO solving methods.

It is worth noting that $X_{r,m}$ (i.e., the path attributes vector) is known and fixed in this study. It is assumed to represent the historical path conditions based on which passengers make their habitual choices. Different from typical transit/traffic assignment problems where path choices are estimated by assuming user equilibrium (for planning purposes), the AFC data-based estimation aims to find the actual realized path choices based on real-world observations (i.e., OD entry-exit flows). Since passengers make decisions before knowing the actual travel or waiting times, $X_{r,m}$ should reflect passengers' historical perceptions of path attributes and should not change within the model estimation process. Therefore, though $C_{k,i}$ captures the actual path crowding information, it should *not* be included in the path choice formulation as passengers make decisions before knowing the actual crowding.

3. Simulation-Based Optimization Algorithms

There are four major classes of methods for solving the SBO problems, including the heuristic methods, direct search methods, gradient-based methods, and response surface methods (Osorio and Bierlaire [17]; Amaran et al. [18]). Heuristic methods are partial search algorithms that may provide a sufficiently good solution to an optimization problem, especially with incomplete or imperfect information or limited computation capacity. Direct search methods are derivative-free methods that are based on the sequential examination of trial points generated by a certain strategy. They are attractive as they are easy to describe and implement. More importantly, they are suitable for objective functions where gradients do not exist everywhere. Gradient-based approaches (or stochastic approximation methods) attempt to optimize the objective function using estimated gradient information. These methods aim to imitate the steepest descent methods in derivative-based optimization. Finite difference schemes can be used to estimate gradients but they may involve a large number of

expensive function evaluations if the number of decision variables is large. Response surface methods are useful in the context of continuous optimization problems. They focus on learning input-output relationships to approximate the underlying simulation by a predefined functional form (also known as a metamodel or surrogate model). This functional form can then be used for optimization leveraging powerful derivative-based optimization techniques.

In this study, we use seven representative algorithms belonging to these four classes of SBO methods to address the aforementioned path choice and train capacity calibration problem. Table 1 summarizes the main characteristic of these algorithms. The summary of all algorithms is described in Table 1.

In the discussion that follows, let Θ be the combined vector of β and θ (i.e., $\Theta = [\beta, \theta]$ is the vector of all coefficients to be estimated). Let N be the dimension of Θ (i.e., $\Theta \in \mathbb{R}^N$).

3.1. Genetic Algorithm (GA). GA is a heuristic method for solving both constrained and unconstrained optimization problems, which belongs to the larger class of evolutionary algorithms inspired by natural selection, the process that drives biological evolution. The GA repeatedly modifies a population of individual solutions as an evolution process [26]. The GA can be used to solve a variety of optimization problems that are not well suited for standard optimization algorithms, such as the SBO problem where the objective function (or constraints) is nondifferentiable and highly nonlinear.

The evolution starts from a population of randomly generated individuals and is an iterative process, with the population in each iteration called a generation. In each generation, the genetic algorithm selects individuals at random from the current population to be parents and uses them to produce the children for the next generation. Over successive generations, the population "evolves" toward an optimal solution. The genetic algorithm uses three main procedures at each step to create the next generation from the current population. (1) Selection: select the individuals, called parents, who contribute to the population of the next generation. Individuals with better objective function values are more likely to be selected. (2) Crossover: combine two parents to form children for the next generation. (3) Mutation: apply random changes to individual parents to form children.

In this study, we adopted a blend crossover and Gaussian mutation methods. The probability of crossover is set as 0.8 and the probability of mutating is set as 0.4. And, the population size is set as 6 given the limited computational budget. The algorithm is implemented by the Python deap package [19].

3.2. Simulated Annealing (SA). SA is a heuristic method for solving optimization problems [27]. The method is based on the physical process of heating a material and then slowly lowering the temperature to decrease defects, thus minimizing the system energy.

At each iteration of the SA algorithm, a new point is randomly generated. The distance of the new point from the current point, or the extent of the search, is based on a probability distribution with a scale proportional to the temperature. A distorted Cauchy–Lorentz visiting distribution is used in this study [20]. The algorithm accepts not only all new points that lower the objective function but also, with a certain probability, points that raise the objective function. By accepting points that raise the objective function, the algorithm avoids being trapped in local minima. An annealing schedule is selected to systematically decrease the temperature as the algorithm proceeds. As the temperature decreases, the algorithm reduces the extent of its search to converge to a minimum.

In this study, the SA algorithm in Python Scipy package is adopted for the implementation with all model parameters set as default [28].

3.3. Nelder–Mead Simplex Algorithm (NMSA). NMSA is a simplex method for finding a local minimum [29]. NMSA in N dimensions maintains a set of $N + 1$ test points arranged as a *simplex*. Denote the initial value of Θ as Θ^{ini}. The initial simplex set ($N + 1$ points) is generated as $\{\{\Theta\colon \Theta = \Theta^{\text{ini}} + e_i, \forall i = 1, \ldots, N\}\} \cup \{\Theta^{\text{ini}}\}$, where $e_i \in \mathbb{R}^N$ is the unit vector in the ith coordinate and σ is the step size which is set as 0.05 in this study [21].

Based on the initial simplex, the model evaluates the objective function for each test point, in order to find a new test point to replace one of the old test points. The new candidate can be generated through simplex centroid reflections, contractions, or other means depending on the function value of the test points. The process will generate a sequence of simplexes, for which the function values at the vertices get smaller and smaller. The size of the simplex is reduced, and finally, the coordinates of the minimum point are found.

Four possible operations, reflection, expansion, contraction, and shrink, are associated with the corresponding scalar parameters: α_1 (reflection), α_2 (expansion), α_3 (contraction), and α_4 (shrink). In this study, we set the value of these parameters as $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\} = \{1, 2, 0.5, 0.5\}$ as suggested in [21]. The algorithm is implemented by the Python scikit-learn package with all parameters set as default. Since NMSA is designed for unconstrained problem, we turned the bound of Θ into a big penalized term in the objective function for this algorithm. For more details regarding the NMSA, one can refer to [21].

3.4. Mesh Adaptive Direct Search (MADS). The MADS algorithm is a direct search framework for nonlinear optimization [30]. It seeks to improve the current solution by testing points in the neighborhood of the current point (the incumbent). The neighborhood points are generated by moving one step in each direction from the incumbent on an iteration-dependent mesh. Each iteration of MADS consists of a SEARCH stage and an optional POLL stage. The SEARCH stage evaluates a finite number of points proposed by the searching strategy (e.g., moving one step around from

Table 1: Algorithms' summary.

Type	Algorithm	Constraints	Stochastic	Source
Heuristic method	Genetic algorithm (GA)	Yes	Yes	Fortin et al. [19]
	Simulated annealing (SA)	Yes	Yes	Tsallis and Stariolo [20]
Direct search	Nelder–Mead simplex algorithm (NMSA)	No	No	Gao and Han [21]
	Mesh adaptive direct search (MADS)	Yes	Yes	Abramson et al. [22]
Gradient-based	Simultaneous perturbation Stochastic approximation (SPSA)	Yes	Yes	Spall et al. [23]
Response surface	Bayesian optimization (BYO)	Yes	Yes	Snoek et al. [24]
	Constrained optimization using Response surfaces (CORS)	Yes	Yes	Regis and Shoemaker [25]

the current point). Whenever the SEARCH step fails to generate an improved mesh point, the POLL step is invoked. The POLL step conducts local exploration near the current incumbent, which also intends to find an improved point on the mesh. Once an improved point is found, the algorithm updates the current point and constructs a new mesh. According to [30], the mesh size parameters approach zero as the number of iteration approaches infinity, which demonstrates the convergence of the MADS algorithm.

In this paper, we use a variant of the MADS method called ORTHO-MADS, which leverages a special orthogonal positive spanning set of polling directions. More details regarding the algorithm can be found in [22]. NOMAD 3.9.1 [31] with the Python interface is used for the MADS algorithm application. The hyperparameters are tuned based on the NOMAD user guide. The direction type is set as orthogonal, with $N + 1$ directions generated at each poll. Latin hypercube search is not applied.

3.5. Simultaneous Perturbation Stochastic Approximation (SPSA). SPSA is a descent direction method for finding local minimums. It approximates the gradient with only two measurements of the objective function, regardless of the dimension of the optimization problem. Denote the objective function in equation (4) as $Z(\Theta)$. The estimated parameters in the kth iteration is denoted as $\Theta^{(k)}$. Then, one iteration for the SPSA is performed as

$$\Theta^{(k+1)} = \Theta^{(k)} - a_k \cdot \widehat{\nabla} Z\left(\Theta^{(k)}\right), \tag{6}$$

where

$$\widehat{\nabla} Z\left(\Theta^{(k)}\right) = \frac{Z\left(\Theta^{(k)} + c_k\Delta_k\right) - Z\left(\Theta^{(k)} - c_k\Delta_k\right)}{2c_k\Delta_k}, \tag{7}$$

$$a_k = \frac{a}{(k + 1 + A)^{\alpha}}, \tag{8}$$

$$c_k = \frac{c}{(k + 1)^{\gamma}}, \tag{9}$$

where Δ_k is a random perturbation vector, whose elements are obtained from a Bernoulli distribution with the probability parameter equal to 0.5. $\{\alpha, \gamma, a, c, A\}$ are tuned as $\{0.602, 0.101, 0.001, 0.007, 0.1M\}$ in this study according to the numerical tests and guidelines from prior empirical studies of Gomez-Dans [32]. M is the maximum number of iterations.

3.6. Bayesian Optimization (BYO). BYO constructs a probabilistic model of the objective function and exploits this model to determine where to evaluate the objective function for the next step. The philosophy of BYO is to use all of the information available from previous evaluations, instead of simply relying on the local gradient and Hessian approximations. This enables BYO to find the minimum of difficult nonconvex functions with relatively few function evaluations.

BYO assumes a prior distribution for the objective function values and uses an acquisition function to determine the next point to evaluate. In this study, we use the Gaussian process as the prior distribution for the objective function due to its flexibility and tractability. For the acquisition function, we tested three common criteria: probability of improvement (POI), expected improvement (EI), and upper confidence bound (UCB) [24]. The EI criterion is used in this path choice calibration problem due to its best performance in our problem. The BYO is implemented in Python with the bayes_opt package. More details regarding the BYO can be found in [24].

3.7. Constrained Optimization Using Response Surfaces (CORS). CORS is a response surface method for global optimization. In each iteration, it updates the response surface model based on all previously probed points and selects the next point to evaluate. The principles for the next point selection are (a) finding new points that have lower objective function value and (b) improving the fitting of the response surface model by sampling feasible regions where little information exists. Hence, the next point is selected by solving the minimization problem of the current response surface function subject to constraints that the next point should be more than a certain distance away from all previous points [25].

An algorithm following the CORS framework requires two components: (a) a scheme for selecting an initial set of points for objective function evaluation and (b) a procedure for globally approximating the objective function (i.e., a response surface model). In this study, the initial sampling is conducted using the Latin hypercube methods, with the initial sampling number equal to $0.2\times$ the total number of function evaluations allowed. The radial basis function (RBS) is used as the response surface model. For the subsequent sampling, a modified version of the CORS algorithm with space re-scaling is used. Details about the algorithm can be found in [25, 33].

4. Case Study

The proposed modeling framework is tested using data from the Hong Kong MTR network. MTR is the operator of the Hong Kong urban rail network, which provides services for the urbanized areas of Hong Kong Island, Kowloon, and the New Territories. The system currently consists of 11 lines with 218.2 km (135.6 miles) of rail, serving 159 stations including 91 heavy rail stations and 68 light rail stops. It serves over 5 million trips on an average weekday. Most of the passengers use a smart card fare payment system named Octopus. For the urban heavy rail lines, trip transactions are recorded when passengers enter and exit the system, providing information about the tap-in and tap-out stations and corresponding timestamps.

4.1. Experimental Design. We use AFC data on a typical weekday afternoon peak period (18 : 00–19 : 00) in March 2017 for the model application. Li [34] conducted a revealed-preference (RP) path choice survey of more than 20,000 passengers in the MTR system and used them to estimate a path choice model. The estimation results are shown in A. The following attributes were used in the specification of the model: (a) total in-vehicle time, (b) the number of transfer times, (c) relative walking time (total walking time divided by total path distance), and (d) the commonality factor (equation (2)). Future research may consider more path choice attributes such as perceived crowding levels and estimated waiting times.

As the real-world path choice information and train capacity are usually unavailable, we validate the models with synthetic data. To generate the synthetic data, we first extract the OD entry flow ($q^{i_m,j}$) from the real-world AFC records. We assume a synthetic Θ as the "true" path choice and train capacity parameters. The TNL model with the true OD entry flow, train timetable, and synthetic Θ as inputs is used to simulate the travel of passengers in the system and record people's tap-in and tap-out time. The input timetable is treated as the *synthetic AVL data*. The resulting passengers' tap-in and tap-out times are treated as the *synthetic AFC data*. The synthetic data, including "true" passenger path choices and train capacity, are used to evaluate the performance of the model under the various solution algorithms. All OD pairs of the whole network are considered in the experiments.

To compare the different SBO solving algorithms, we design five test scenarios summarized in Table 2. Each scenario has a different synthetic Θ. The selection of synthetic Θ can represent different assumptions about passengers' choice behavior and sensitivity to crowding. For the reference scenario, we use the path choice parameters in Table 3 as the synthetic β and use the estimated train capacity parameters in [13] as the synthetic θ.

Passengers' actual path choice behavior is assumed to be random (each path is equally likely to be selected) or deterministic. For the random path choice scenario, we set all synthetic choice parameters as 0, which means all available paths are equally likely to be chosen. For the deterministic (the word "deterministic" here just represents the degree of randomness is low. The "truly" deterministic corresponds to all parameters go to $\longrightarrow -\infty$) path choice scenario, we set all synthetic choice parameters as the lower bounds (i.e., the maximum absolute value possible). Under this scenario, a slight difference in attributes between two paths can lead to a high difference in choice probability (i.e., this is close to passengers following the shortest path). As for the train capacity, the synthetic θ for these two scenarios is the same as the reference scenario.

Passengers' sensitivity to crowding may also vary. If all passengers are not sensitive to the crowding, train capacity can be modeled as a fixed value. However, if passengers become more sensitive to the crowding, the actual train capacity may largely depend on the crowding level in the train and on the platform. Therefore, passengers' sensitivity to crowding can be reflected by the scale of θ_1 and θ_2 [13]. For the crowding-sensitive scenario, we set the synthetic train capacity parameters as $\theta_0 = 225$, $\theta_1 = 0.2$, and $\theta_2 = 0.2$. Compared to the reference scenario, θ_1 and θ_2 are higher to represent higher sensitivity. And, θ_0 is decreased to offset the capacity increase caused by the increase of θ_1 and θ_2. As for the crowding-insensitive scenario, we set the synthetic train capacity parameters as $\theta_0 = 235$, $\theta_1 = 0$, and $\theta_2 = 0$, which can be seen as a fixed-capacity model.

4.2. Case Study Settings. The lower and upper bounds of all parameters ($L_\beta, U_\beta, L_\theta, U_\theta$) are shown in Table 2. Θ^{ini} is set as $(L_\Theta + U_\Theta)/2$ for all scenarios. To compare different algorithms, a fixed computational budget, 100 function evaluations, is applied to all algorithms. All algorithms except for NMSA (deterministic algorithm) are replicated for 5 times (with different random seeds) to decrease the impact of randomness.

4.3. Reference Scenario Results. The convergence results of the reference scenario are depicted in Figure 3. Each point represents the average value over all replications. We found that the performance of different algorithms varied. Given the limited number of function evaluations, CORS, BYO, and SPSA converge to a relatively small objective function. GA, MADS, and SA have relatively large objective function values upon termination. In terms of convergence speed, the response surface methods (BYO and CORS) have the fastest convergence speed. They also reach the lowest objective function value. This is consistent with conclusions regarding the performance of the SBO algorithms when used in the transportation domains [17, 35–37].

Figure 3 also summarizes the behavior of the algorithm stability. The vertical line indicates the 1/4× standard deviations over the five replications. NMSA is a deterministic algorithm and not affected by randomness. BYO and CORS show high randomness in the first half iterations. However, as the number of function evaluations increases, the standard deviation of the objective function decreases, and the results become stable. GA, SA, and MADS are unstable compared to other algorithms. This means that the heuristic algorithms (GA and SA) are not suitable for the calibration problem studied in this paper. The instability of

Table 2: Scenario design.

Parameter category	Synthetic Θ	Scenarios					
		Reference	Path choice		Train capacity		Bound
			Random	Deterministic	Crowding-sensitive	Crowding-insensitive	
Path choice	In-vehicle time	−0.147	0	−2.0	−0.147	−0.147	[−2, 0]
	Relative walking time	−1.271	0	−5.0	−1.271	−1.271	[−5, 0]
	Number of transfers	−0.573	0	−3.0	−0.573	−0.573	[−3, 0]
	Commonality factor	−3.679	0	−10.0	−3.679	−3.679	[−10, 0]
Train capacity	θ_0	232	232	232	225	235	[220, 260]
	θ_1	0.0732	0.0732	0.0732	0.2	0	[0, 0.2]
	θ_2	0.0607	0.0607	0.0607	0.2	0	[0, 0.2]

Table 3: Path choice model estimation results.

	Estimate	Std. error	t-value	
In-vehicle time	−0.147	0.011	−13.64	***
Relative walking time	−1.271	0.278	−4.56	***
Number of transfers	−0.573	0.084	−6.18	***
Commonality factor	−3.679	1.273	−2.89	**
$\rho^2 = 0.54$				

***: $p < 0.01$; **: $p < 0.05$.

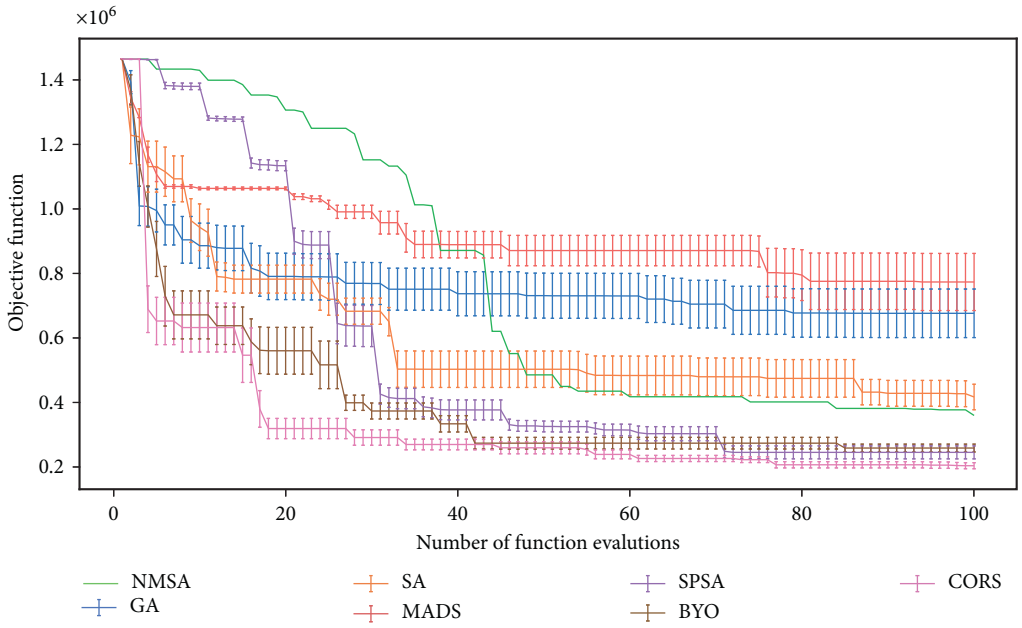


Figure 3: Convergence results of reference scenario. The error bar indicates 1/4× standard deviation. NMSA has no error bar because it is a deterministic algorithm.

MADS may be because it may converge to nonstationary points [38].

Table 4 compares the parameters estimated by different algorithms with the synthetic ones. Although some algorithms can reach similar objective function values, they result in different estimated parameters. For example, CORS and SPSA have similar objective function values. However, SPSA performs better in path choice estimation, while CORS performs better in train capacity estimation. We also observe that the train capacity parameters are relatively harder to estimate. This may be because most of the stations in the rail system are not congested and all passengers can board the trains. Thus, the objective function is not very sensitive to train capacity parameters.

4.4. Sensitivity Analysis

4.4.1. Impact of Randomness in Path Choice Behavior. Figure 4 shows the estimation results for the two path choice-related scenarios: random and deterministic. The estimated parameters are shown in Tables 5 and 6. For the

Table 4: Estimation results of the reference scenario.

Category	Variable name	"True"	Estimated parameters						
			GA	SA	NMSA	MADS	SPSA	BYO	CORS
Path choice	In-vehicle time	−0.147	−0.392	−0.327	−0.342	−0.454	−0.170	−0.207	−0.229
	Relative walking time	−1.271	−2.205	−3.010	−3.020	−0.302	−2.257	−2.493	−2.486
	Number of transfers	−0.573	−1.143	−0.787	−0.389	−1.248	−0.598	−0.776	−0.756
	Commonality factor	−3.679	−6.482	−6.851	−7.250	−7.834	−4.419	−5.434	−5.716
Train capacity	θ_0	232	239	243	259	252	241	234	243
	θ_1	0.073	0.117	0.118	0.146	0.040	0.162	0.110	0.069
	θ_2	0.061	0.069	0.110	0.080	0.080	0.163	0.100	0.086
Objective function		—	676,392	416,923	359,663	773,526	245,269	258,688	203,885

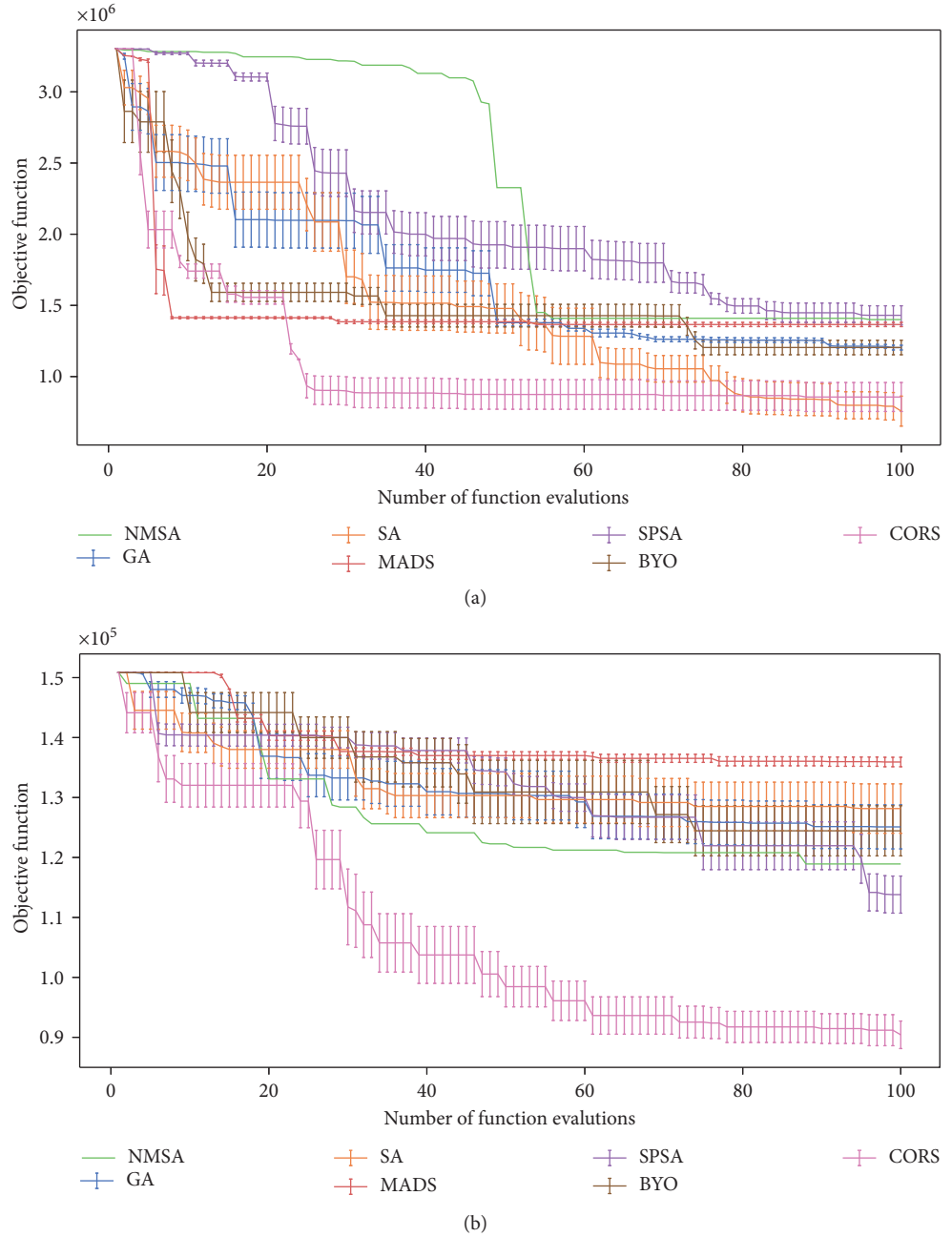


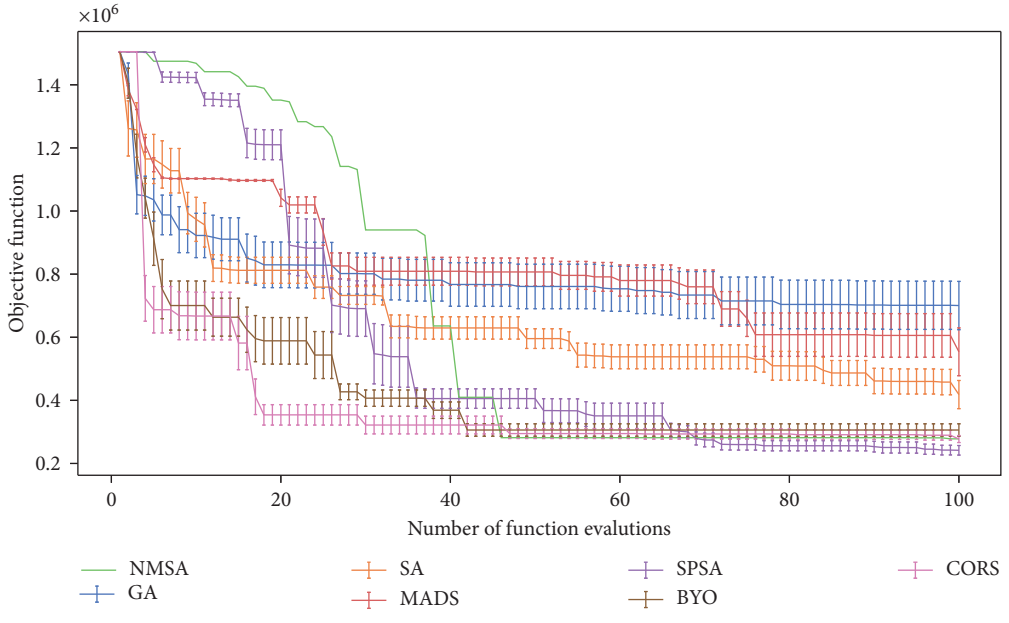
Figure 4: Algorithm performance in the two path choice scenarios. (a) Random. (b) Deterministic.

Table 5: Estimation results of the random path choice scenario.

Category	Variable Name	"True"	Estimated Parameters						
			GA	SA	NMSA	MADS	SPSA	BYO	CORS
Path choice	In-vehicle time	0	0	−0.072	−0.050	0	−0.108	−0.037	0
	Relative walking time	0	−2.151	−1.139	−1.807	−1.000	−1.719	−3.725	−0.702
	Number of transfers	0	−0.348	−0.185	−0.435	−1.334	−0.631	−0.207	0
	Commonality factor	0	−5.997	−1.945	−9.991	−5.432	−5.127	−4.155	−8.000
Train capacity	θ_0	232	243	224	254	232	241	248	223
	θ_1	0.073	0.067	0.050	0.124	0.048	0.106	0.079	0.016
	θ_2	0.061	0.037	0.072	0.136	0.134	0.112	0.159	0.072
Objective function		—	1,202,761	756,321	1,399,836	1,365,291	1,429,942	1,203,696	855,627

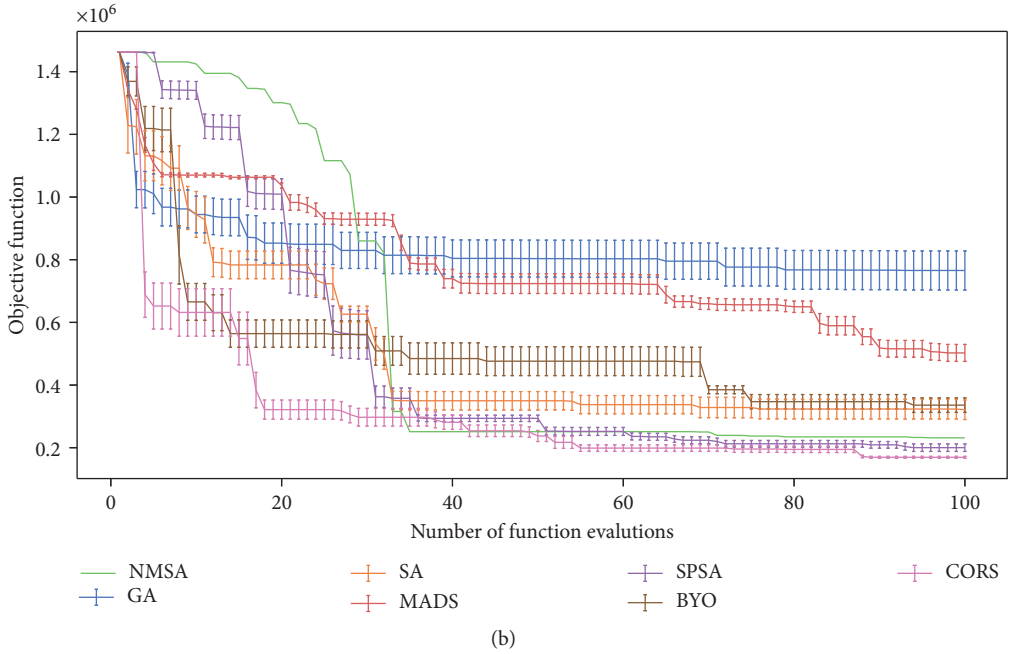
Table 6: Estimation results of the deterministic path choice scenario.

Category	Variable Name	"True"	Estimated Parameters						
			GA	SA	NMSA	MADS	SPSA	BYO	CORS
Path choice	In-vehicle time	−2	−1.240	−1.243	−1.205	−1.160	−1.544	−1.537	−1.830
	Relative walking time	−5	−3.180	−3.358	−2.819	−2.480	−3.728	−3.807	−4.492
	Number of transfers	−3	−1.575	−1.551	−1.419	−1.524	−1.786	−1.761	−2.661
	Commonality factor	−10	−5.307	−5.251	−4.735	−4.920	−6.346	−6.379	−8.819
Train capacity	θ_0	232	237	232	228	237	239	232	237
	θ_1	0.073	0.095	0.076	0.095	0.180	0.097	0.110	0.123
	θ_2	0.061	0.101	0.069	0.091	0.062	0.110	0.106	0.093
Objective function		—	125,100	128,157	118,915	135,922	113,805	124,448	63,220



(a)

Figure 5: Continued.



(b)

Figure 5: Algorithm performance in the two train capacity scenarios. (a) Crowding-insensitive. (b) Crowding-sensitive.

Table 7: Estimation results of the crowding-insensitive train capacity scenario.

Category	Variable Name	"True"	Estimated Parameters						
			GA	SA	NMSA	MADS	SPSA	BYO	CORS
Path choice	In-vehicle time	−0.147	−0.392	−0.181	−0.254	−0.460	−0.191	−0.197	−0.177
	Relative walking time	−1.271	−2.153	−2.044	−2.636	−2.294	−2.284	−2.469	−2.025
	Number of transfers	−0.573	−1.127	−1.614	−1.279	−0.490	−0.760	−0.908	−1.011
	Commonality factor	−3.679	−6.489	−6.500	−7.492	−7.750	−5.299	−5.474	−5.130
Train capacity	θ_0	235	239	245	249	230	241	238	236
	θ_1	0	0.088	0.109	0.096	0.050	0.096	0.093	0.084
	θ_2	0	0.05	0.058	0.108	0.050	0.150	0.078	0.063
Objective function		—	700,441	418,196	277,835	553,765	241,533	305,846	277,212

Table 8: Estimation results of the crowding-sensitive train capacity scenario.

Category	Variable Name	"True"	Estimated Parameters						
			GA	SA	NMSA	MADS	SPSA	BYO	CORS
Path choice	In-vehicle time	−0.147	−0.472	−0.217	−0.228	−0.332	−0.177	−0.195	−0.196
	Relative walking time	−1.271	−2.533	−1.575	−2.735	−1.568	−2.118	−1.763	−2.534
	Number of transfers	−0.573	−0.759	−1.169	−1.323	−0.816	−0.495	−0.892	−0.734
	Commonality factor	−3.679	−6.489	−6.324	−7.040	−7.834	−4.361	−6.046	−5.021
Train capacity	θ_0	225	238	244	238	245	244	237	237
	θ_1	0.2	0.166	0.149	0.121	0.112	0.140	0.085	0.099
	θ_2	0.2	0.080	0.114	0.125	0.144	0.129	0.123	0.110
Objective function		—	765,621	320,745	231,228	502,341	199,753	335,558	169,057

random scenario, all "true" (synthetic) path choice parameters are set as zero, which means all paths are equally likely to be chosen. We observe that, in this scenario (Figure 4(a)), CORS and SA algorithms perform the best with the lowest objective function. Compared to the reference scenario in Section 4.3, the decreased performance of BYO and SPSA may be due to the "true" β is close to the upper bound ($U_\beta = 0$). The Gaussian posterior distribution in BYO and gradient estimation in SPSA can suffer from instability in the boundary. From Table 5, we observe the

parameters of in-vehicle time and number of transfers are better estimated than those of relative walking time and commonality factors.

Figure 4(b) shows the results of the deterministic scenario. The initial objective function is relatively small (1.5×10^5) compared to the reference scenario (1.5×10^6). All algorithms only reduce the objective function by around 1/3 except for the CORS algorithm. The good performance of CORS may come from global searching with the Latin hypercube method. It is better suited to explore the points near boundaries. Although the objective function does not decrease too much, the estimated parameters are still acceptable (see Table 6).

4.4.2. Impact of Crowding Sensitivity. Figure 5 shows the estimation results of the two scenarios related to train capacity (i.e., crowding-sensitive and crowding-insensitive). In the crowding-insensitive scenario (Figure 5(a)), the conclusions are similar to the reference scenario. CORS, BYO, NMSA, and SPSA converge to low objective function values and outperform other algorithms. The performance of NMSA and MADS is improved compared to the reference scenario. In the crowding-sensitive scenario, we still observe a good performance by the CORS, NMSA, and SPSA algorithms. The performance of BYO is slightly reduced. The results shown in Tables 7 and 8 indicate that θ_0 (base capacity) is hard to estimate. This may be because trains at most stations do not reach the capacity. Therefore, for many OD pairs, the OD exit flows (directly related to the objective function) are not sensitive to the base capacity parameter.

5. Conclusion

In this paper, we propose an SBO framework to calibrate train capacity and path choice model parameters simultaneously for metro systems using AFC and AVL data. The advantage of the proposed framework lies in capturing the collective effect of both path choices and train capacity on passenger journey times. Seven representative algorithms from four main branches of SBO methods are applied and compared with respect to their solution accuracy, convergence speed, and stability. We applied the proposed framework using data from the Hong Kong MTR network and compared the performance of the different algorithms. Overall, the results show that some algorithms result in a reasonable estimation of the parameters of interest. These results also support the effectiveness of the proposed SBO framework for calibrating these key parameters using AFC and AVL data. Especially, the response surface methods (particularly CORS) exhibit consistently good performance. The SBO framework is flexible to accommodate a wide range of path choice and train capacity models in transit simulation models.

This paper has some limitations. First, we validate the framework and evaluate the algorithmic performance only using synthetic AFC and AVL data. Therefore, the complexities of noise and uncertainties in actual data do not play any role. This is caused by the absence of real-world path choice and train capacity information. Future research can collect real-world path choice and train capacity data to conduct more realistic model validation. Second, we assumed that the path choice behavior is similar for the whole network (same β values). Given the real-world path choice behavior is possibly more diverse and heterogeneous, future research can explore clustering different OD pairs with different β values based on individual mobility characteristics [39].

Appendix

(A). Passenger Path Choice Model for MTR System

These results are from [34]. The C-logit model formulation is the same as equations (1) and (2). A total number of 31,640 passengers completed the questionnaire. After filtering duplicate responses, 26,996 responses were available. The model results are shown in Table 3. The main explanatory variables are the total in-vehicle time, relative transfer walking time, and number of transfers. All variables are statistically significant with the expected signs. Paths with high in-vehicle time, walking time, and number of transfers are less likely to be chosen by passengers.

Data Availability

The AFC and AVL data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

B. Mo, Z. Ma, H.N. Koutsopoulos, and J. Zhao conceptualized and designed the study; B. Mo and Z. Ma collected the data; B. Mo, Z. Ma, and H.N. Koutsopoulos analysed and interpreted the results; B. Mo and H.N. Koutsopoulos prepared the draft. All authors reviewed the results and approved the final version of the manuscript.

Acknowledgments

The authors would like to thank the Hong Kong Mass Transit Railway (MTR) for their support and data availability for this research. Also, the authors acknowledge MIT Libraries for providing funding for the open-access publication of the paper.

References

[1] Z. Liu, S. Wang, W. Chen, and Y. Zheng, "Willingness to board: a novel concept for modeling queuing up passengers," *Transportation Research Part B: Methodological*, vol. 90, pp. 70–82, 2016.

[2] J. Preston, J. Pritchard, and B. Waterson, "Train overcrowding," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2649, no. 1, pp. 1–8, 2017.

[3] H. N. Koutsopoulos, Z. Ma, P. Noursalehi, and Y. Zhu, "Transit data analytics for planning, monitoring, control, and information," in *Mobility Patterns, Big Data and Transport Analytics*, C. Antoniou, L. Dimitriou, and F. Pereira, Eds., pp. 229–261, Elsevier, Amsterdam, Netherlands, 2019.

[4] T. Kusakabe, T. Iryo, and Y. Asakura, "Estimation method for railway passengers' train choice behavior with smart card transaction data," *Transportation*, vol. 37, no. 5, pp. 731–749, 2010.

[5] F. Zhou and R.-H. Xu, "Model of passenger flow assignment for urban rail transit based on entry and exit time constraints," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2284, no. 1, pp. 57–61, 2012.

[6] P. Kumar, A. Khani, and Q. He, "A robust method for estimating transit passenger trajectories using automated data," *Transportation Research Part C: Emerging Technologies*, vol. 95, pp. 731–747, 2018.

[7] Y. Zhu, H. N. Koutsopoulos, and N. H. Wilson, "Passenger itinerary inference model for congested urban rail networks," *Transportation Research*, vol. 123, no. 7, 2020.

[8] Y. Sun and R. Xu, "Rail transit travel time reliability and estimation of passenger route choice behavior," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2275, no. 1, pp. 58–67, 2012.

[9] L. Sun, Y. Lu, J. G. Jin, D.-H. Lee, and K. W. Axhausen, "An integrated bayesian approach for passenger flow assignment in metro networks," *Transportation Research Part C: Emerging Technologies*, vol. 52, pp. 116–131, 2015.

[10] J. Zhao, F. Zhang, L. Tu et al., "Estimation of passenger route choice pattern using smart card data for complex metro systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 4, pp. 790–801, 2017.

[11] X. Xu, L. Xie, H. Li, and L. Qin, "Learning the route choice behavior of subway passengers from afc data," *Expert Systems with Applications*, vol. 95, pp. 324–332, 2018.

[12] B. Mo, Z. Ma, H. Koutsopoulos, and J. Zhao, "Assignment-based path choice estimation for metro system using smart card data," in *Proceedings of the 24th International Symposium on Transportation & Traffic Theory (ISTTT)*, Beijing, China, July 2020.

[13] B. Mo, Z. Ma, H. N. Koutsopoulos, and J. Zhao, "Capacity-constrained network performance model for urban rail systems," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2674, no. 5, pp. 59–69, 2020.

[14] E. Cascetta, A. Nuzzolo, F. Russo, and A. Vitetta, "A modified logit route choice model overcoming path overlapping problems. specification and some calibration results for interurban networks," in *Proceedings of the 13th International Symposium on Transportation and Traffic Theory*, Lyon, France, July 1996.

[15] C. G. Prato, "Route choice modeling: past, present and future research directions," *Journal of Choice Modelling*, vol. 2, no. 1, pp. 65–100, 2009.

[16] M. E. Ben-Akiva and S. R. Lerman, *Discrete Choice Analysis: Theory and Application to Travel Demand*, MIT press, Cambridge, MA, USA, 1985.

[17] C. Osorio and M. Bierlaire, "A simulation-based optimization framework for urban transportation problems," *Operations Research*, vol. 61, no. 6, pp. 1333–1345, 2013.

[18] S. Amaran, N. V. Sahinidis, B. Sharda, and S. J. Bury, "Simulation optimization: a review of algorithms and applications," *Annals of Operations Research*, vol. 240, no. 1, pp. 351–380, 2016.

[19] F. A. Fortin, F. M. De Rainville, M. A. Gardner, M. Parizeau, and C. Gagné, "DEAP: evolutionary algorithms made easy," *Journal of Machine Learning Research*, vol. 13, pp. 2171–2175, 2012.

[20] C. Tsallis and D. A. Stariolo, "Generalized simulated annealing," *Physica A: Statistical Mechanics and Its Applications*, vol. 233, no. 1-2, pp. 395–406, 1996.

[21] F. Gao and L. Han, "Implementing the nelder-mead simplex algorithm with adaptive parameters," *Computational Optimization and Applications*, vol. 51, no. 1, pp. 259–277, 2012.

[22] M. A. Abramson, C. Audet, J. E. Dennis, and S. L. Digabel, "Orthomads: a deterministic mads instance with orthogonal directions," *SIAM Journal on Optimization*, vol. 20, no. 2, pp. 948–966, 2009.

[23] J. C. Spall, "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation," *IEEE Transactions on Automatic Control*, vol. 37, no. 3, pp. 332–341, 1992.

[24] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," in *Proceedings of the 25th International Conference on Neural Information Processing Systems*, pp. 2951–2959, Krong Siem Reap, Cambodia, December 2012.

[25] R. G. Regis and C. A. Shoemaker, "Constrained global optimization of expensive black box functions using radial basis functions," *Journal of Global Optimization*, vol. 31, no. 1, pp. 153–171, 2005.

[26] D. Whitley, "A genetic algorithm tutorial," *Statistics and Computing*, vol. 4, pp. 65–85, 1994.

[27] P. J. Van Laarhoven and E. H. Aarts, "Simulated annealing," in *Simulated Annealing: Theory and Applications*Springer, Berlin, Germany, 1987.

[28] Scipy, "Scipy dual annealing algorithm," 2019, https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.dual_annealing.htmlURL:.

[29] J. A. Nelder and R. Mead, "A simplex method for function minimization," *The Computer Journal*, vol. 7, no. 4, pp. 308–313, 1965.

[30] C. Audet and J. E. Dennis, "Mesh adaptive direct search algorithms for constrained optimization," *SIAM Journal on Optimization*, vol. 17, no. 1, pp. 188–217, 2006.

[31] C. Audet, S. Le Digabel, and C. Tribes, "Nomad user guide Rapport technique," 2009.

[32] J. Gomez-Dans, "A simultaneous perturbation stochastic approximation optimisation code in python," 2012.

[33] P. Knysh and Y. Korkolis, "Blackbox: a procedure for parallel optimization of expensive black-box functions," 2016, https://arxiv.org/abs/1605.00998.

[34] W. Li, "Route and transfer station choice modeling in the mtr system," Working paper, 2014.

[35] Q. Cheng, S. Wang, Z. Liu, and Y. Yuan, "Surrogate-based simulation optimization approach for day-to-day dynamics model calibration with real data," *Transportation Research Part C: Emerging Technologies*, vol. 105, pp. 422–438, 2019.

[36] B. Mo, Z. Ma, H. Koutsopoulos, and J. Zhao, "Calibrating route choice for urban rail system: a comparative analysis using simulation-based optimization methods," in *Proceedings of the Transportation Research Board 99th Annual Meeting*, Washington, D.C, USA, January 2020.

[37] B. Mo, "Network performance model for urban rail systems," Master's Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2020.
[38] M. A. Abramson and C. Audet, "Convergence of mesh adaptive direct search to second-order stationary points," *SIAM Journal on Optimization*, vol. 17, no. 2, pp. 606–619, 2006.
[39] B. Mo, Z. Zhao, H. N. Koutsopoulos, and J. Zhao, "Individual mobility prediction: an interpretable activity-based hidden markov approach," 2021, https://arxiv.org/abs/2101.03996.