Physics-Informed Machine Learning for Calibrating Macroscopic Traffic Flow Models

Yu Tang[a], Li Jin[b], and Kaan Ozbay[a]

[a]*C2SMARTER Center, Department of Civil & Urban engineering, Tandon School of Engineering, New York University, 6 Metrotech Ctr. 4th Floor, Brooklyn, New York, 11201, USA*
[b]*UM Joint Institute and Department of Automation, Shanghai Jiao Tong University, 800 Dongchuan Lu, Shanghai, 200240, China*

Abstract

Well-calibrated traffic flow models are fundamental to understanding traffic phenomena and designing control strategies. Traditional calibration has been developed base on optimization methods. In this paper, we propose a novel physics-informed, learning-based calibration approach that achieves performances comparable to and even better than those of optimization-based methods. To this end, we combine the classical deep autoencoder, an unsupervised machine learning model consisting of one encoder and one decoder, with traffic flow models. Our approach informs the decoder of the physical traffic flow models and thus induces the encoder to yield reasonable traffic parameters given flow and speed measurements. We also introduce the denoising autoencoder into our method so that it can handles not only with normal data but also with corrupted data with missing values. We verified our approach with a case study of I-210 E in California. It turns out that our approach can achieve comparable performance to the-state-of-the-art calibration methods given normal data, and outperform them given corrupted data with missing values.

Keywords: Physics-informed machine learning, parameter identification, traffic flow models

1 Introduction

1.1 Motivation

Macroscopic traffic flow models have been shown to be capable of reproducing congestion propagation and explaining complicated phenomena, such as capacity drops (Khoshyaran and Lebacque, 2015) and stop-and-go waves (Laval and Leclercq, 2010). They provide a solid foundation for the performance analysis of traffic systems (Huang et al., 2020; Shi and Li, 2021) and control design for freeway management (Gomes and Horowitz, 2006; Papamichail et al., 2010). However, before they are put into practice, these models should be carefully calibrated to accurately replicate real-life complications briefly mentioned above.

Extensive studies have been devoted to the calibration of traffic flow models. They mainly utilized optimization approaches to determine model parameters in specific days (Spiliopoulou et al., 2014, 2017; Mohammadian et al., 2021; Wang et al., 2022). Unfortunately, the parameters, such as capacity, reportedly vary significantly across multiple days (Dervisoglu et al., 2009), and thus their transferability could be poor (Wang et al., 2022). This indicates that daily re-calibration is necessary for quantifying parameter uncertainties and designing robust traffic control strategies.

Admittedly, one can apply optimization-based approaches to day-to-day calibration, but this can be cumbersome in practice. First, these methods bring about heavy computation costs when being adopted for long-term modeling. They typically involve non-convex and even non-smooth problems, and it is hard to solve them, let alone to repeat the calibration procedures for several months or years. The second problem arises from data quality. Traditional traffic sensors, such as induction loops, are infamously unreliable. For instance, the proportion of malfunctioning detectors in the California's freeway system vary day by day, roughly around 36% (Caltrans, 2022). Clearly, optimization methods are vulnerable to the fluctuation in data quality since they only consider day-specific data for each calibration.

In response to the above challenges, we develop a novel physics-informed, learning-based approach of identifying traffic flow parameters across days, including capacity, free flow speed, jam density and congestion wave speed. The proposed method belongs to the unsupervised machine learning category since the actual values of parameters are unknown in advance. It is inspired by the classical deep autoencoder shown in Figure 1a) (Hinton and Salakhutdinov, 2006) and the denoising autoencoder shown in Figure 1b) (Vincent et al., 2008). Both of them comprise two neural networks, one as an encoder and the other one as a decoder, but the classical deep autoencoder is usually used for dimension reduction while the denoising autoencoder, after being well-trained, is mainly applied to restore corrupted data. We build our calibration methods based on these two kinds of autoencoders by informing them of physics knowledge about traffic flows. More concretely, the classical deep autoencoder-based approach handles model parameter estimation given intact traffic measurements; the denoising autoencoder-based one resolves calibration when partial traffic data are corrupted with missing values. A unified framework is illustrated in Figure 1c). Specially, we use the neural network-based decoder to approximate the physics-based model, which simultaneously induces the encoder to yield reasonable traffic flow parameters. The major motivation of introducing the neural network-based decoder is because it is easy to differentiate neural networks. Without it, we need to take the physics-based model as the decoder. In that case, computing the gradients of model outputs with respect to model parameters could have higher computational costs, especially when the outputs are obtained by iterating the physics-based model with many steps.

1.2 Related work

Most of the previous work on calibrating macroscopic traffic flow models employed optimization methods applied over specific days. One approach is to estimate the fundamental diagram (FD), especially for first-order traffic flow models (Muñoz et al., 2004; Dervisoglu et al., 2009; Zhong et al., 2016). It requires little computation, but may suffer from accuracy losses. First, fitting the left part of FD, namely free-flow regime, is usually easy, but the same task can be hard for the

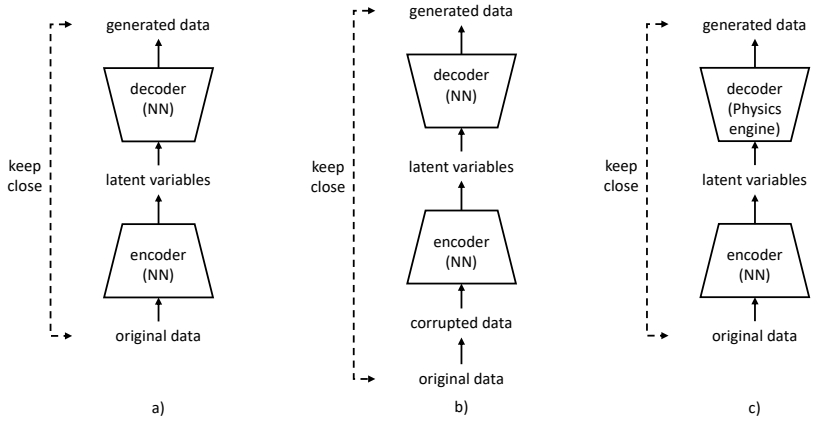


Figure 1: Current frameworks of autoencoders: a) classical deep autoencoder consisting of two neural networks (NNs), b) denoising autoencoder, c) autoencoder for parameter identification.

right part since traffic data collected during congestion periods are sparse and scattered. Second, individual calibration of FDs along a freeway corridor fails to capture flow interactions. More studies formulated the calibration problem as mathematical programming that considered flow dynamics specified by first- or high-order models (Lee and Ozbay, 2008; Spiliopoulou et al., 2014; Mudigonda and Ozbay, 2015; Mohammadian et al., 2021). In this case, one needs to solve a non-convex optimization problem with locally optimal solutions, and thus heuristic optimization/search algorithms, such as simultaneous perturbation stochastic approximation, simulated annealing, etc, can be used; please see Wang et al. (2022) for a comprehensive review.

Recently, some researchers discussed the idea of learning-based calibration in the context of traffic state estimation (TSE). That is, they integrated traffic flow models into machine learning methods to enhance TSE (Huang and Agarwal, 2020; Yuan et al., 2021a,b; Shi et al., 2021b,a; Di et al., 2023), to incorporate simultaneous parameter identification and state estimation techniques. Some work investigated uncertainty quantification for TSE as well (Mo et al., 2022a,b). It should be pointed out that these studies are different from what is proposed in this paper. First, they put emphasis on TSE that used partial observations to infer full states. Thus, they divided training and testing data by sensor locations to evaluate transferability over space. By contrast, our method takes in full observations and returns traffic parameters. We desire transferability over time periods and thereby separate training and testing data by time. Second, although the current studies can update traffic parameters, they still require relatively good initial values that are normally obtained from classical calibration approaches (Yuan et al., 2020) because poor initial traffic parameters do not guarantee convergence (Shi et al., 2021a). This kind of good initial parameter estimates, however, is not necessarily required by our approach. Third, these approaches typically assume homogeneous traffic parameters, which could be inappropriate for modeling large-scale freeway networks, while our method is more flexible and can yield inhomogeneous parameters, e.g. various capacities in different freeways segments.

Our proposed method is also related to deep autoencoder-based system identification which has emerged in recent years. Depending on identification goals, the latent variables, obtained from the autoencoder, can be recognized as either states or parameters. If one wants to fit a neural network approximating to the physical dynamic model, the encoder gives the states (Masti and

Bemporad, 2021; Beintema et al., 2021; Gedon et al., 2021). The encoder can also yield exact parameters of physical models. In that case the decoder is a physical model rather than a NN (Nagel and Huber, 2021; Yang et al., 2022); see the structure in Figure 1c). The successful application of this framework requires gradient back-propagation from the physics engine to the encoder. For linear time-invariant systems, this is easily achieved since it is possible to derive exact expression of gradients (Nagel and Huber, 2021). For non-linear systems, it could be also achieved by applying neural ODEs and automatic differentiation (Yang et al., 2022).

However, there is a fundamental problem behind the structure illustrated in Figure 1c). When a dynamical model (no matter neural ODEs or recurrent networks) is too deep with too many iterations, the gradient computation could fail because of the vanishing-explosion problem (Pascanu et al., 2013; Choromanski et al., 2020; Nguyen et al., 2022). This problem could emerge in traffic flow models. Due to the stability requirement, the traffic flow models, such as the CTM, typically iterates with a small step size (e.g., 5 seconds Kontorinaki et al. (2017)). In practice, we only have access to aggregated measurements (e.g., 5-min traffic data). It means that for reproducing one observation, we need to iterate the CTM 60 times. Long observations, covering congestion building-up and dissipation, are necessary for calibrating traffic flow models correctly. Given observations during a period of peak hours up to several hours, the traffic flow models need to iterate more than one thousand times. In that case, gradient computation becomes unreliable due to the vanishing-explosion problem and it is unlikely to use the framework shown in Fig. 1c) to train the encoder to learn parameters of traffic flow models. Thus, we propose a framework of physics-informed autoencoder shown in Fig. 3, where we introduce a neural network-based decoder as a surrogate for the traffic flow model. Meanwhile, we keep the traffic flow model and train the decoder to learn from it. By doing so, we can train the encoder to yield reasonable physics parameters by using gradient back-propagation from the decoder rather than from the physics model.

We summarize the existing work on learning-based traffic state estimation (TSE) and parameter identification (PI) in Table 1. It should be pointed out that our calibration method has the capability of generalization over physics parameters. It means that given a series of observations or measurements, our well-trained autoencoder-based model could yield associated physics parameters. This is different from physics-informed neural networks for TSE, which need retraining once the physics models change. More importantly, by introducing a neural network-based encoder as the surrogate, our calibration method could handle long observations for nonlinear CTM, which hasn't been reported in the existing studies.

Table 1: Summary of related work on learning-based TSE and PI.

Reference	Scope	Physics model	Generalization over physics parameters	Input of neural networks	Handle long observations
Huang and Agarwal (2020)	TSE	LWR model	No	time and coordinates	-
Yuan et al. (2021a)	TSE	LWR model	No	time and coordinates	-
Yuan et al. (2021b)	TSE	LWR model	No	time and coordinates	-
Shi et al. (2021a)	TSE & PI	ARZ model	No	time and coordinates	-
Shi et al. (2021b)	TSE & PI	LWR & ARZ models	No	time and coordinates	-
Lu et al. (2023)	TSE & PI	Traffic flow & fluid queue models	No	time and coordinates	-
Jaques et al. (2020)	PI	General dynamical systems	Yes	Observation trajectory	No
Nagel and Huber (2021)	PI	Linear dynamical systems	Yes	Observation trajectory	Yes
Yang et al. (2022)	PI	General dynamical systems	Yes	Observation trajectory	No
This paper	PI	Nonlinear CTM	Yes	Observation trajectory	Yes

1.3 Our contributions

We try to address two main questions:

(i) How can we train an encoder, given perfect freeway measurements, to yield appropriate traffic parameters that are comparable to those calibrated by traditional methods?

(ii) How can we complete the task in (i), even given corrupted traffic data with missing values?

To resolve (i), we extend the deep autoencoder (Hinton and Salakhutdinov, 2006) that converts high-dimensional data into latent variables, typically with lower dimensions, by minimizing the error between encoder inputs and decoder outputs; see Figure 1a). Generally, the original low-dimensional representations do not have significant physical meanings; they cannot be recognized as parameters of traffic flow models. To address this problem, we also feed encoder outputs into traffic flow models and inform our decoder of extra discrepancies between the simulation result and its own output. Clearly, minimizing the new error encourages the decoder to learn physical laws of traffic flows, and the total error decreases only when the encoder yields appropriate parameters of traffic flow models. Besides, we introduce the concept of conditional generation. That is, the decoder relies not only on latent variables (parameters), but also on boundary conditions, such as upstream traffic volumes. This is a straightforward but indispensable extension since traffic observations, mimicked by the decoder, are determined by these conditions.

To answer (ii), we integrate denoising autoencoder (Vincent et al., 2008), a simple but robust variant of the original autoencoder, into the calibration approach. That is, we use unspoiled sensor readings to generate new data with partially missing values, which mimics the pattern of real unreliable data, and then apply this synthesized data to training. It allows to deploy the parameter identification on real data with missing values after training.

2 Problem Statement

In this section, we generally state the calibration problem and then explicitly present the considered freeway model. It consists of a dynamics model, the CTM incorporating capacity drop, and an observation model.

2.1 Learning-based calibration problem

In this paper, we consider calibrating traffic in a certain period of peak hours during which recurrent congestion builds up and dissipates. Following the definition (Dowling et al., 2004), recurrent congestion here indicates the congestion occurring on a normal weekday without incidents, bad weather or other random events. It implies that we do not consider non-recurrent congestion in this paper. Non-recurrent congestion usually has more complicated patterns than recurrent congestion. Thus we believe the first step should be aimed at calibrating recurrent congestion. We consider addressing the calibration of non-current congestion in future research.

We also assume i) that the parameters keep the same within a certain period (e.g., morning or evening peak hours) given a specific day but ii) that the parameters could vary from day to day. These assumptions are reasonable and they have been supported by much practice (Muñoz et al., 2004; Dervisoglu et al., 2009; Kontorinaki et al., 2017; Wang et al., 2022). It should be pointed out that the assumptions do not imply the same traffic parameters for an entire day; they allow different periods of peak hours (e.g., morning and evening peak hours) to have different traffic flow parameters. Besides, our calibration method is aimed at a single period of peak hours. The current practice shows it suffices to separately calibrate multiple periods of peak hours (e.g., morning and evening peak hours in (Wang et al., 2022)).

We consider a freeway corridor with K mainline *cells*, K on-ramp *buffers*, and K off-ramps, as shown in Figure 2. The kth cell is characterized by traffic density, denoted by $\rho_k(t)$. Note that the first buffer is not an actual on-ramp; instead, it represents the upstream freeway section. The kth buffer generates a time-varying *demand* $\alpha_k(t) \in \mathbb{R}_{\geq 0}$ for the kth cell. In addition, we apply *mainline ratio* $\eta_k(t) \in [0,1]$ to model off-ramp flows. This ratio denotes the fraction of traffic from cell k entering cell $k+1$; the remaining traffic flow leaves the freeway at the kth off-ramp. We also assume that the last cell K discharges outflows at a speed of $\tilde{v}_K(t)$, which can be measured as the downstream boundary condition. Finally, we denote by $f_k(t)$ the flows from cell k to the downstream cell.

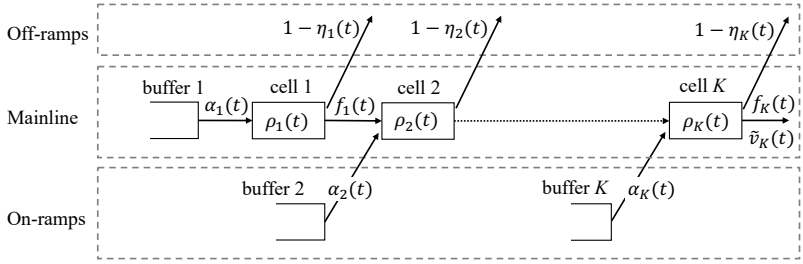


Figure 2: A freeway corridor with states $\rho(t)$, traffic demands $\alpha(t)$, mainline ratio $\eta(t)$ and downstream discharging speed $\tilde{v}_K(t)$, where $\rho(t) := [\rho_1(t), \cdots, \rho_K(t)]^{\mathrm{T}}$, $\alpha(t) := [\alpha_1(t), \cdots, \alpha_K(t)]^{\mathrm{T}}$ and $\eta(t) := [\eta_1(t), \cdots, \eta_K(t)]^{\mathrm{T}}$.

Now we suppose that the following freeway model:

$$\rho(t+1) = F(\rho(t), q(t), \alpha(t), \eta(t), \tilde{v}_K(t); \theta), \tag{1a}$$

$$y(t) = H(\rho(t), q(t)) + \varepsilon(t), \tag{1b}$$

where F in (1a) denotes the dynamical model parameterized by θ, H in (1b) denotes an observation model, $\varepsilon(t)$ represents measurement noise and $y(t)$ is observed traffic measurement. As per the convention (Schoukens and Ljung, 2019), we assume that the noise $\varepsilon(t)$ is zero-mean with finite variance.

For convenience of notation, we let θ_p, $\mathbf{M}_p := (y(0), \cdots, y(T))_p$, $\mathbf{I}_p := (\rho(0))_p$ and $\mathbf{B}_p := (\alpha(0), \cdots, \alpha(T), \eta(0), \cdots, \eta(T), \tilde{v}_K(0), \cdots, \tilde{v}_K(T))_p$ denote the parameter, real measurements, initial conditions and boundary conditions, respectively. The calibration problem is formulated as follows. Suppose $\mathscr{P}^{\text{train}}$ is a set of periods, each of them with $T+1$ time steps. Given observations $\{\mathbf{M}_p\}_{p \in \mathscr{P}^{\text{train}}}$, boundary conditions $\{\mathbf{B}_p\}_{p \in \mathscr{P}^{\text{train}}}$ and initial conditions $\{\mathbf{I}_p\}_{p \in \mathscr{P}^{\text{train}}}$, we aim

at training a machine learning model so that it can yield suitable model parameters $\{\theta_p\}_{p\in\mathscr{P}^{\text{test}}}$ over the testing dataset $\mathscr{P}^{\text{test}}$, $\{\mathbf{M}_p\}_{p\in\mathscr{P}^{\text{test}}}$, $\{\mathbf{I}_p\}_{p\in\mathscr{P}^{\text{test}}}$ and $\{\mathbf{B}_p\}_{p\in\mathscr{P}^{\text{test}}}$.

2.2 Freeway model

In the following, we first use the cell transmission model (CTM) to specify the dynamics functions F and G, and the parameter θ in (1a) and then build the observation model H in (1b) that relies on induction loops.

2.2.1 Dynamics model

The CTM is favored due to its simplicity and wide use, and it is not a necessary requirement of our approach. Assuredly, high-order models (Payne, 1971; Whitham, 1974; Messner and Papageorgiou, 1990; Aw and Rascle, 2000; Zhang, 2002) can reproduce traffic phenomena with higher accuracy, but they have more parameters that could complicate the training process. To the best of our knowledge, this is the first attempt to estimate traffic flow parameters using unsupervised training. It is thus worthwhile starting with a simple model. We leave identification of high-order models as a future research task. Specially, it will be interesting to investigate whether we can accelerate it via calibrating first-order models since first- and high-order models have common traffic flow parameters.

By the CTM, the flows between cells, f_k for $k = 1,2,\cdots,K$, are given by

$$f_k(t) = \eta_k(t)\min\left\{v_k\rho_k(t), Q_k(t), w_{k+1}(\bar{\rho}_{k+1}-\rho_{k+1}(t)) - \alpha_{k+1}(t)\right\},\ 1\le k\le K-1 \tag{2a}$$

$$f_K(t) = \tilde{v}_K(t)\rho_K(t) \tag{2b}$$

where δ_t denotes time step size, v_k denotes free-flow speed of cell k, $Q_k(t)$ denotes capacity of cell k. The flow functions (2a)-(2b) indicate higher merging priority of on-ramp flows and the first-in-first-out rule for off-ramp flows (Ferrara et al., 2018). Besides, note that we consider time-varying capacity $Q_k(t)$ which allows to model capacity drop as follows:

$$Q_k(t) = \begin{cases} \bar{Q}_k, & \rho_k(t)\le\rho_k^{\text{c}}, \\ (1-\zeta_k)\bar{Q}_k, & \rho_k(t)>\rho_k^{\text{c}}, \end{cases} \tag{3}$$

where $\bar{Q}_k$ is the capacity of cell k, $\zeta_k\in[0,1]$ represents capacity drop ratio, and $\rho_k^{\text{c}} := \bar{Q}_k/v_k$ denotes critical density of cell k; see more discussions and modeling of capacity drop in (Kontorinaki et al., 2017).

Then, by the conservation law of flows, the traffic dynamics is given by

$$q_k(t+1) = q_k(t) + \delta_t(\alpha_k(t) - r_k(t)),\ 1\le k\le K,\ t = 0,1,\cdots, \tag{4a}$$

$$\rho_1(t+1) = \rho_1(t) + \frac{\delta_t}{\ell_1}(r_1(t) - \frac{f_1(t)}{1-\eta_1(t)}),\ t = 0,1,\cdots, \tag{4b}$$

$$\rho_k(t+1) = \rho_k(t) + \frac{\delta_t}{\ell_k}(r_k(t) + f_{k-1}(t) - \frac{f_k(t)}{1-\eta_k(t)}),\ 2\le k\le K,\ t = 0,1,\cdots, \tag{4c}$$

where ℓ_k denotes the length of cell k.

Then the parameters to be calibrated are presented below:

$$\theta = (\{v_k\}_{k=1}^{K-1}, \{\bar{Q}_k\}_{k=1}^{K-1}, \{\zeta_k\}_{k=1}^{K-1} \{\bar{\rho}_k\}_{k=2}^{K}, \{w_k\}_{k=2}^{K}). \tag{5}$$

2.2.2 Observation model

Now we consider widely-used induction loops for the observation function H and the sensor output $y(t)$ in (1b). In practice, induction loops update measurements of flow rates and speed at a certain frequency Δ_t that is larger than the time step size δ_t of the traffic model. We suppose $\Delta_t = m\delta_t$ with a multiple $m \in \mathbb{Z}_{>0}$. We also denote by $\mathscr{K}$ the set of cells where sensors are deployed. Then the sensor outputs are given by

$$\bar{f}_k(t) = \sum_{i=t-m+1}^{t} f_k(i)/m + \varepsilon_f(t),\ k \in \mathscr{K},\ t = m, 2m, \ldots, \tag{6a}$$

$$\bar{v}_k(t) = \sum_{i=t-m+1}^{t} f_k(i) v_k(i) / \sum_{i=t-m+1}^{t} f_k(i) + \varepsilon_v(t),\ k \in \mathscr{K},\ t = m, 2m, \ldots, \tag{6b}$$

where $v_k(t) := f_k(t)/\rho_k(t)$ denotes traffic speed of cell k at time t, $\varepsilon_f(t)$ (resp. $\varepsilon_v(t)$) represents zero-mean noise in flow (resp. speed) measurements. Clearly, (6a)-(6b) yield

$$y(t) = \{\bar{f}_k(t), \bar{v}_k(t)\}_{k \in \mathscr{K}}, t \in \mathscr{T} = \{m, 2m, \cdots\}. \tag{7}$$

3 Proposed Method

In this section, we first introduce the framework of physics-informed autoencoder for parameter identification. Then we present more details about inside structures of the autoencoder.

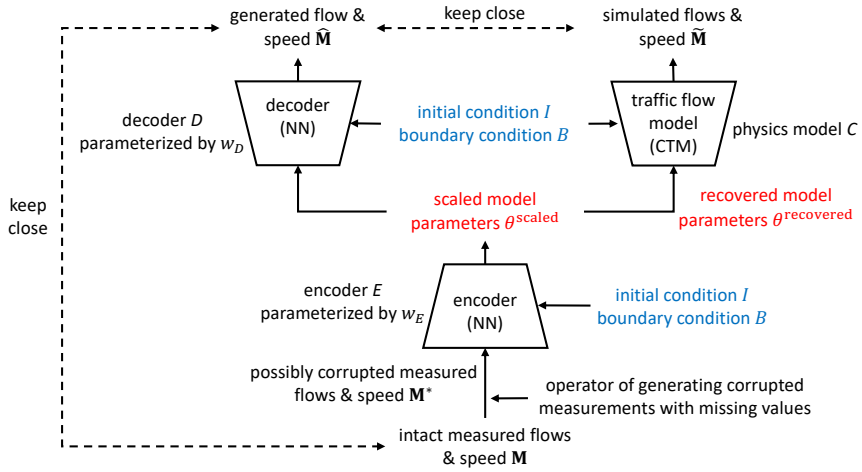


Figure 3: A novel architecture of physics-informed autoencoders.

3.1 Physics-informed autoencoder for parameter identification

Figure 3 illustrates the novel architecture of our proposed physics-informed autoencoder. The encoder first feeds traffic measurements, boundary conditions and initial conditions and outputs model parameters scaled between zero and one. The decoder takes the concatenation of the encoder output, boundary conditions and initial conditions as its input; it attempts to generate flow and speed measurements. Meanwhile, the model parameters, boundary conditions and initial conditions are passed into the CTM to obtain simulated flows and speed. Note that if we need to train the autoencoder for calibration over corrupted measurements with missing values, we feed synthesized data $\mathbf{M}^*$, with the same missing pattern, to the encoder, as shown in Figure 3. Clearly, if there is no such a need, we just let $\mathbf{M}^* = \mathbf{M}$.

Formally, we denote by E the encoder parameterized by w_E, by D the decoder parameterized by w_D and by C the physics model CTM. The workflow is given by

$$\theta_p^{\text{scaled}} = E(\mathbf{M}_p^*, \mathbf{B}_p, \mathbf{I}_p; w_E), \tag{8a}$$

$$\theta_p^{\text{recovered}} = (\theta^{\max} - \theta^{\min}) \odot \theta_p^{\text{scaled}} + \theta^{\min} \tag{8b}$$

$$\hat{\mathbf{M}}_p = D(\theta_p^{\text{scaled}}, \mathbf{B}_p, \mathbf{I}_p; w_D), \tag{8c}$$

$$\tilde{\mathbf{M}}_p = C(\theta_p^{\text{recovered}}, \mathbf{B}_p, \mathbf{I}_p), \tag{8d}$$

where we use the subscript p to indicate the dependence on period p, and $\odot$ denotes element-wise product. θ_p^{scaled} is a vector whose elements are scaled between zero and one. This can be achieved by applying the sigmoid function to the last layer of the encoder. $\theta^{\min}$ (resp. $\theta^{\max}$) is a predetermined lower bound (resp. upper bound) of traffic parameters. By (8b), the recovered parameter $\theta_p^{\text{recovered}}$ could have suitable physical meanings and thus can be admitted into the traffic flow model. Once our machine leaning model is well trained, we recognize $\theta_p^{\text{recovered}}$ as calibrated traffic model parameters for period p.

As for training, we consider three kinds of loss functions below:

$$L_1 = \sum_{p \in \mathscr{P}^{\text{train}}} ||\hat{\mathbf{M}}_p - \tilde{\mathbf{M}}_p||_2^2, \tag{9a}$$

$$L_2 = \sum_{p \in \mathscr{P}^{\text{train}}} ||\mathbf{M}_p - \hat{\mathbf{M}}_p||_2^2, \tag{9b}$$

$$\begin{aligned} L_3 = \sum_{p \in \mathscr{P}^{\text{train}}} \Big(& \sum_{k=1}^{K-2} (v_{k+1,p}^{\text{scaled}} - v_{k,p}^{\text{scaled}})^2 + \sum_{k=1}^{K-2} (\bar{Q}_{k+1,p}^{\text{scaled}} - \bar{Q}_{k,p}^{\text{scaled}})^2 + \sum_{k=1}^{K-2} (\zeta_{k+1,p}^{\text{scaled}} - \zeta_{k,p}^{\text{scaled}})^2 \\ & + \sum_{k=2}^{K-1} (\bar{\rho}_{k+1,p}^{\text{scaled}} - \bar{\rho}_{k,p}^{\text{scaled}})^2 + \sum_{k=2}^{K-1} (w_{k+1,p}^{\text{scaled}} - w_{k,p}^{\text{scaled}})^2 \Big), \end{aligned} \tag{9c}$$

where $v_{k,p}^{\text{scaled}}$, $\bar{Q}_{k,p}^{\text{scaled}}$, $\zeta_{k,p}^{\text{scaled}}$, $\bar{\rho}_{k,p}^{\text{scaled}}$ and $w_{k,p}^{\text{scaled}}$ are components of θ_p^{scaled} by recalling (5). The first loss function quantifies error between the decoder outputs and simulation results. Minimizing it ensures that the decoder learns the physical laws of the CTM. The second one gives error between the decoder outputs and original inputs. Minimizing it together with L_1 induces the encoder to yield appropriate traffic flow parameters. The last loss function measures discrepancies of traffic model

parameters of two consecutive cells. It usually acts as a regularization term against overfitting (Engl et al., 1996). The final loss function is given by

$$\min_{w_E, w_D} L = \min_{w_E, w_D} L_1 + \beta L_2 + \gamma L_3 \tag{10}$$

where β and γ denote weights of the loss functions L_2 and L_3. Clearly, if β is too small or too large, the encoder fails to yield appropriate traffic model parameters. Besides, large γ reduces inhomogeneity of traffic model parameters.

3.2 Encoder and decoder structures

The following introduces the structures inside the encoder and the decoder. It should be pointed out that our major novelty in this paper lies in the proposed architecture. To demonstrate its effectiveness, we prefer not to consider advanced structures of the encoder or decoder. Clearly, the structures inside the encoder and decoder are not unique. We consider the classical convolutional neural network-based (CNN-based) structure (Masci et al., 2011). Indeed, other advanced models, such as neural operator with the advantage of discretization- invariance (Kovachki et al., 2021), could be used to refine the structures considered in this paper and even to extend our calibration method for PDE-based traffic flow models. We leave it in the future research.

The encoder and decoder structures are illustrated in Figure 4. As for the encoder, we first apply convolution operators to the measurements, initial conditions and the boundary conditions. Then we flatten and concatenate the results. After that, the concatenation is passed into dense layers to yield physics parameters. As for the decoder, we first flatten boundary conditions and concatenate them with physics parameters and initial condition. Then the results are fed into dense layers. Finally, we exploit deconvolution operators to generate measurements. It should be pointed out that the boundary conditions include temporal information.

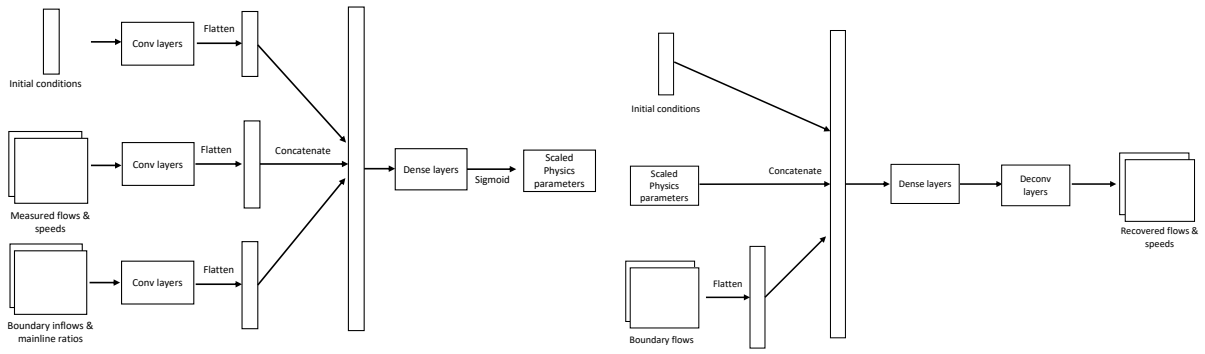
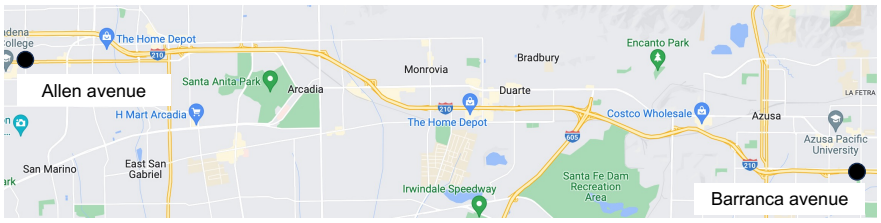


Figure 4: The structure of (a) encoder and (b) decoder.

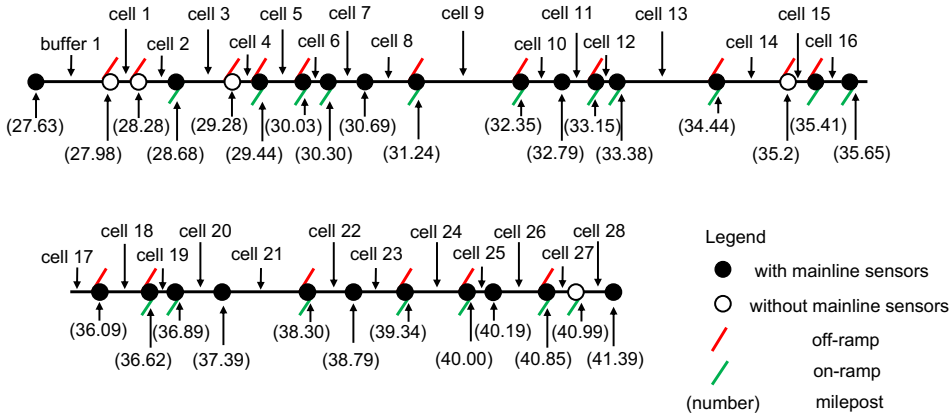
4 Case Study

We test our method on a segment of Interstate 210 Eastbound (I210 E) from Allen Avenue to Barranca Avenue, up to 22.15 kilometers, shown in Figure 5a). This segment is considered since it encompasses a full process of congestion build-up and dissipation; see Figure 5a). It has 18 on-ramps, 17 off-ramps, all of them equipped with induction loops expect for the off-ramps of cells 14 and 18. There are also 25 mainline induction loops. We divide it into 28 cells based on locations of on-ramps, off-ramps and mainline sensors; see Figure 5b). Clearly, we have all of the cells equipped with mainline sensors, except for cells 1, 3, 14 and 27, namely

$$\mathscr{K} = \{1, 2, \cdots, 28\} \setminus \{1, 3, 14, 27\}. \tag{11}$$



(a) A segment of I210 E from Allen Avenue to Barranca Avenue.



(b) Cell division.

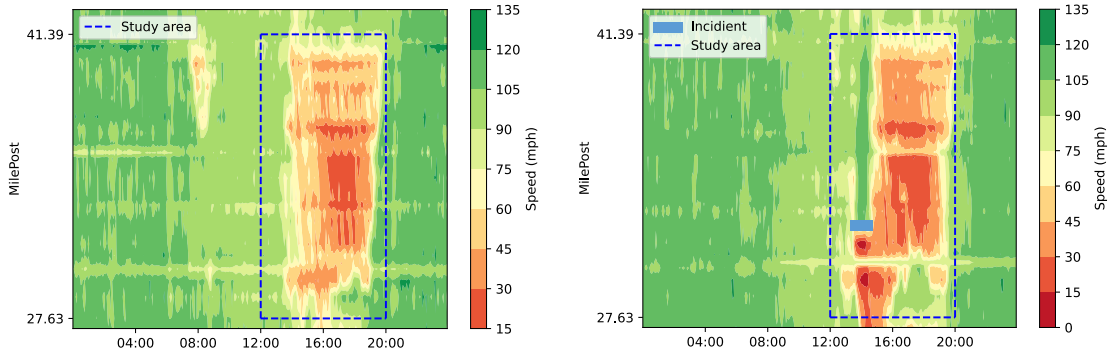
Figure 5: Modeling I210 E.

4.1 Experiment design

4.1.1 Data preparation

We collected three types of data in 2019 from the PeMS (Caltrans, 2022), namely i) sensor measurements, ii) sensor quality data and iii) incident records. The first data provide 5-min flows through the mainline, on-ramps and off-ramps and also 5-min speed at the mainline. Note that we inferred the off-ramp flow of cells 14 and 18 based on the conservation law since there are none of sensors deployed at these two off-ramps. The second data indicate whether each 5-min

measurement is from a normal or malfunctioning sensor. The last data give location and duration of incidents.



(a) (b)

Figure 6: Congestion a) without significant incidents on 2019/04/16, b) with significant incidents on 2019/10/28. We exclude the latter from our training and testing dataset.

Preliminary analysis showed there is recurrent congestion between 12:00 and 21:00 in the selected freeway segment. Based on the three types of data above, we classified the days in 2019 into four types: i) congested without significant incidents, ii) congested with significant incidents, iii) uncongested (typically on weekends) and iv) unknown (due to breakdown of all sensors). Figures 5a) and 5b) illustrate congestion with and without significant incidents, respectively. Clearly, incidents can temporarily change traffic parameters, such as capacity, and induce new bottlenecks.

As discussed in Section 2.1, we only consider the first type of days in this paper. It turns out there are 182 such days in 2019. Among them, only 73 days have high-quality traffic measurements for all sensors; in the remaining days, the sensors at milepost 28.68, 30.03, 32.79, 34.44, 35.65, 37.39 and 40.85 malfunctioned. We randomly divided these 73 days, around 70% for training dataset denoted by $\{\mathbf{M}_p\}_{p\in\mathscr{P}^{\text{train}}}$, 15% for testing dataset denoted by $\{\mathbf{M}_p\}_{p\in\mathscr{P}^{\text{test}}}$ and 15% for validation dataset denoted by $\{\mathbf{M}_p\}_{p\in\mathscr{P}^{\text{val}}}$.

We also synthesized training (resp. testing) data from real-world training (resp. testing) data. The procedure is elaborate below. We first use the fundamental diagram-based approach to obtain rough estimation of traffic model parameters. Then we add noises into these parameters, boundary conditions and initial conditions. Finally, we use the CTM to generate the simulation data which is recognized as traffic measurements. Clearly, we known the ground truths of the traffic parameters behind these measurements. We denote by $\{\mathbf{M}^s_p\}_{p\in\mathscr{P}^{\text{train}}}$, $\{\mathbf{M}^s_p\}_{p\in\mathscr{P}^{\text{test}}}$ and $\{\mathbf{M}^s_p\}_{p\in\mathscr{P}^{\text{val}}}$ synthesized training, testing and validation dataset, respectively. Data synthesis is considered for two reasons. First, we know the ground truths of traffic parameters behind the synthesized training and testing dataset. Thus, we can better evaluate the calibration results. Second, data synthesis provides substantial data for pretraining our machine learning model before we deploying it on real-world data.

4.1.2 Settings of our proposed method

Here we introduce basic settings of our autoencoder-based calibration method. In the encoder, each convolution module in Figure 3 consists of two convolution layers, with 4×4 and 2×2 filters respectively. Besides, there are two dense layers with 512 and 256 hidden neurons. In the decoder, there are also two dense layers, with 256 and 512 hidden neurons respectively, and two deconvolution layers, with 2×2 and 4×4 filters respectively. This architecture is inspired by the classical neural network LeNet (LeCun et al., 1998).

We also specify the maximum and minimum values of traffic parameters for $\theta^{\max}$ and $\theta^{\min}$ in (8b); see Table 2. These values are set based on existing calibration results (Dervisoglu et al., 2009).

Table 2: Minimum and maximum values of traffic parameters.

Parameters	Min value	Max value
Free-flow speed v (km/h)	100	120
Capacity $\bar{Q}$ (veh/(hour·lane))	1400	2200
Capacity drop ratio ζ	0	0.15
Jam density $\bar{\rho}$ (veh/(km·lane))	67	167
Congestion wave speed w (km/h)	10	32

4.1.3 Benchmarks and metrics

We consider another three calibration approaches as benchmarks:

- the first is to calibrate the fundamental diagram (Dervisoglu et al., 2009). Note that not every cell can be calibrated by this approach due to lack of mainline sensors. For these cells, we computed traffic model parameter by interpolating upstream and downstream parameters (Muñoz et al., 2004);
- the second is to formulate nonlinear programming and to solve it with the Nelder-Mead algorithm (Kontorinaki et al., 2017);

As for performance metrics, we consider mean absolute percentage error (MAPE) of flows, speeds and traffic model parameters:

$$e_{\text{flow}} = \frac{1}{|\mathscr{P}||\mathscr{T}||\mathscr{K}|} \sum_{p\in\mathscr{P}} \sum_{t\in\mathscr{T}} \sum_{k\in\mathscr{K}} \frac{|\bar{f}_{k,p}(t) - \tilde{f}_{k,p}(t)|}{\bar{f}_{k,p}(t)}, \tag{12a}$$

$$e_{\text{speed}} = \frac{1}{|\mathscr{P}||\mathscr{T}||\mathscr{K}|} \sum_{p\in\mathscr{P}} \sum_{t\in\mathscr{T}} \sum_{k\in\mathscr{K}} \frac{|\bar{v}_{k,p}(t) - \tilde{v}_{k,p}(t)|}{\bar{v}_{k,p}(t)}, \tag{12b}$$

$$e_{\text{param}} = \frac{1}{5K|\mathscr{P}|} \sum_{p\in\mathscr{P}} \Big(\sum_{k=1}^{K-1} \frac{|v_{k,p} - \hat{v}_{k,p}|}{v_{k,p}} + \sum_{k=1}^{K-1} \frac{|\bar{Q}_{k,p} - \hat{Q}_{k,p}|}{\bar{Q}_{k,p}} + \sum_{k=1}^{K-1} \frac{|\zeta_{k,p} - \hat{\zeta}_{k,p}|}{\zeta_{k,p}}$$

$$+\sum_{k=2}^{K}\frac{|\bar{\rho}_{k,p}-\hat{\rho}_{k,p}|}{\bar{\rho}_{k,p}}+\sum_{k=2}^{K}\frac{|w_{k,p}-\hat{w}_{k,p}|}{w_{k,p}}\Big) \tag{12c}$$

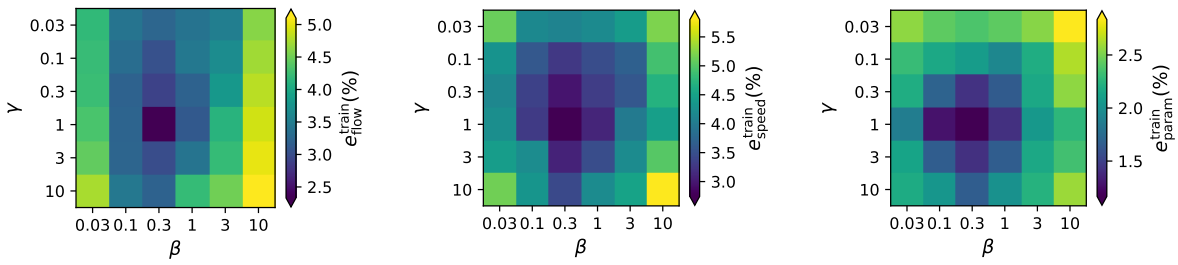
where $\mathscr{T}$ is given by (7), $\mathscr{K}$ is given by (11), $\mathscr{P}$ is a set of days, $\bar{f}_{k,p}(t)$ and $\bar{v}_{k,p}(t)$ are traffic measurements of cell k at time t on day p, $\hat{f}_{k,p}(t)$ and $\hat{v}_{k,p}(t)$ are simulation results from the calibrated models, $(v_{k,p},\bar{Q}_{k,p},\zeta_{k,p},\bar{\rho}_{k,p},w_{k,p})$ are ground truths of traffic parameters of synthesized data, and $(\hat{v}_{k,p},\hat{Q}_{k,p},\hat{\zeta}_{k,p},\hat{\rho}_{k,p},\hat{w}_{k,p})$ denote calibrated traffic flow parameters. Clearly, e_{flow} and e_{speed} can be used to evaluate calibration both on synthesized and real-world dataset, while e_{param} can be applied to assess calibration on synthesized dataset.

4.2 Calibration on data without missing values

This section focuses on evaluation the calibration over dataset without missing values. We first compare the proposed method with the two benchmarks introduced in the previous section and then conduct an ablation study on the neural network-based decoder.

4.2.1 Method comparison

To apply the proposed method, we first conduct sensitivity analysis of the impacts of the weight β and γ on our calibration approach; see Figure 7. It should be pointed out we should select the best combination (β,γ) based on only $e^{\text{train}}_{\text{flow}}$ and $e^{\text{train}}_{\text{speed}}$ since in general we do not know the ground truths of traffic parameters in practice. But as shown in Figure 7, we can minimize $e^{\text{train}}_{\text{param}}$ by selecting appropriate β and γ that minimizes $e^{\text{train}}_{\text{flow}}$ and $e^{\text{train}}_{\text{speed}}$. The result is reasonable. Too large β forces the decoder to fit the original traffic measurements and to ignore the physics from the CTM, while too small β makes the decoder learn the dynamics without awareness of the original traffic measurements. Besides, too large γ restricts the inhomogeneity of traffic parameters and too small γ may lead to overfitting.



(a) (b) (c)

Figure 7: Sensitivity analysis of weights β and γ on a) $e^{\text{train}}_{\text{flow}}$, b) $e^{\text{train}}_{\text{speed}}$, c) $e^{\text{train}}_{\text{param}}$.

Based on Figure 7, we chose the weight $\beta=0.3$ and $\gamma=1$. Then we compared our approach

with the benchmarks. The results on the synthesized data are summarized in Table 5. Clearly, our approach outperforms the fundamental diagram-based (FD-based) calibration, and achieves performance comparable to the nonlinear programming-based (NP-based). It should be pointed out our encoder and decoder directly takes the testing data and yields calibration, while the NP-based approach still requires to solve the optimization problem over the testing dataset. It indicates that our approach could be favored given massive calibration requests.

Table 3: Performances of calibration over synthesized data without missing values.

	$e^{\text{train}}_{\text{flow}}$	$e^{\text{test}}_{\text{flow}}$	$e^{\text{val}}_{\text{flow}}$	$e^{\text{train}}_{\text{speed}}$	$e^{\text{test}}_{\text{speed}}$	$e^{\text{val}}_{\text{speed}}$	$e^{\text{train}}_{\text{param}}$	$e^{\text{test}}_{\text{param}}$	$e^{\text{val}}_{\text{param}}$
FD-based	2.48%	2.44%	2.53%	2.80%	2.90%	3.05%	2.27%	2.05%	2.18%
NP-based	2.28%	2.28%	2.25%	2.18%	2.42%	2.43%	1.00%	1.01%	1.05%
Proposed	2.26%	2.30%	2.30%	2.34%	2.44%	2.50%	1.09%	1.00%	1.06%

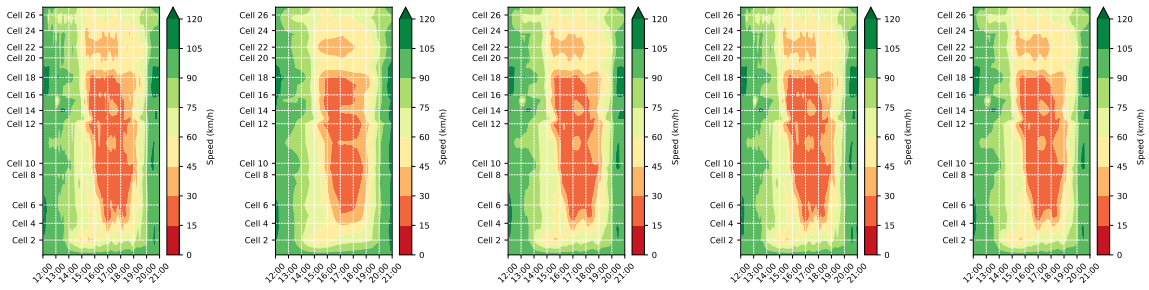
The performances of calibration over real-word dataset are summarized in Table 4. In this case, we do not know the ground truths of traffic parameters, and thus we mainly compare calibration approaches in terms of e_{flow} and e_{speed}. It is not surprising to see that all of the calibration approaches achieves worse performance on the real-world dataset that on the synthesized data. However, the findings from the synthesized data still hold. We can conclude that our approach can outperform the FD-based approach, achieve performance comparable to the NP-based approach. It should be pointed out that each call of the NP-based approach takes around 15 minutes. Thus the calibration of all days need 18 hours if we apply the optimization-based approach. Although our calibration method requires two-hour training, the computational costs of calibration after training are marginal. From this point of view, our calibration is more efficient.

Table 4: Performances of calibration over real-world data without missing values.

	$e^{\text{train}}_{\text{flow}}$	$e^{\text{test}}_{\text{flow}}$	$e^{\text{val}}_{\text{flow}}$	$e^{\text{train}}_{\text{speed}}$	$e^{\text{test}}_{\text{speed}}$	$e^{\text{val}}_{\text{speed}}$
FD-based	4.89%	4.62%	4.80%	11.31%	10.82%	11.08%
NP-based	4.28%	4.14%	4.14%	4.13%	4.01%	4.08%
Proposed	4.27%	4.28%	4.32%	4.09%	4.13%	4.13%

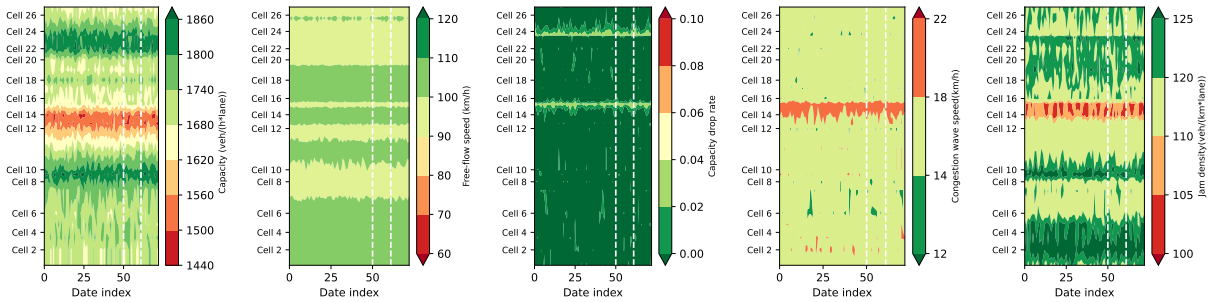
We provide a typical speed heatmap in the latest manuscript, as shown in Figure 8 below. It shows spatial-temporal speed distribution on 2019-12-03 from the testing data. Due to strong fitting capability, the decoder can yield outputs similar to the true values; see Figs. 8a) and d). Besides, given well-calibrated parameters, the physical model can also reproduce the congestion occurrences and clearances; see Figs. 8c) and e).

We also presented the traffic parameters calibrated by our method, as shown in Figure 9. Note that the dash lines divide the dates according to the training, validation and testing date. The results are reasonable. For example, we find the capacity around cell 26 is smaller than that of the upstream cells, which indicates the existence of a bottleneck. Besides, the capacity around cell 14 is relatively small. This is because cell 14 locates at an interchange of two freeway corridors where the weaving flows reduce the capacity. Overall, the capacity drop is not significant in our case study. It only happens around the two bottlenecks aforementioned. We also find that around cell 14, congestion wave speed is relatively large and the jam density is relatively small. This indicates that the congestion happening at around cell 14 will propagates to the upstream more easily.



(a) (b) (c) (d) (e)

Figure 8: Speed heatmaps on 2019-12-03: a) true values, b) calibrated by the fundamental diagram, c) calibrated by nonlinear programming optimization, d) decoder output, e) calibrated by proposed method.



(a) (b) (c) (d) (e)

Figure 9: Traffic parameters calibrated by the proposed method: a) capacity per lane, b) free-flow speed, c) capacity-drop rate, d) congestion wave speed, e) jam density.

4.2.2 Ablation study

Now we conduct the ablation study on the neural network-based decoder. Without the decoder, the proposed architecture, illustrated in Figure 3, is reduced to the one shown in Figure 1c). In that case, we consider two approaches of training the encoder. The first one is to estimate the gradients numerically. For example, suppose $\hat{\theta}_p = [\hat{\theta}_p^1, \hat{\theta}_p^2, \cdots, \hat{\theta}_p^n]$ and we can have the gradient with respect to $\hat{\theta}_p^1$ by

$$\frac{\partial}{\partial \hat{\theta}_p^1} ||\mathbf{M}_p - \tilde{\mathbf{M}}_p||_2^2 \approx \frac{1}{2\delta}(||\mathbf{M}_p - C(\hat{\theta}_p', \mathbf{B}_p, \mathbf{I}_p)||_2^2 - ||\mathbf{M}_p - C(\hat{\theta}_p'', \mathbf{B}_p, \mathbf{I}_p)||_2^2), \tag{13}$$

where $\hat{\theta}_p' = [\hat{\theta}_p^1 + \delta, \hat{\theta}_p^2, \cdots, \hat{\theta}_p^n]$, $\hat{\theta}_p'' = [\hat{\theta}_p^1 - \delta, \hat{\theta}_p^2, \cdots, \hat{\theta}_p^n]$, and the function C is given by (8d). The second one is to implement the CTM by neural ODEs (Yang et al., 2022) and to compute the gradient directly.

We tested these two methods over the real-word dataset. The training losses are presented in Figure 10. Note that in our case study the CTM iterates with a time step size of five seconds and that the simulation considers 9-hour traffic operation. This implies that the CTM iterates 6480 times in each simulation. Clearly, given so many iterations, the neural ODE-based method hardly trained the encoder.

Besides, the gradient estimation also requires huge computation costs. As indicated by (13), each gradient estimation needs to call the simulation twice and the estimation number depends on the number of parameters to be calibrated. In our case study, updating neural network one time typically takes fewer than 0.2 seconds, however running the simulation one time requires around 2 seconds. In terms of efficiency, it is necessary to reduce the times of running the CTM. From this point of view, our calibration method is more efficient. This is because for each training, the times of simulation required by our method only depends on the batch size and has nothing to with the number of parameters to be calibrated.

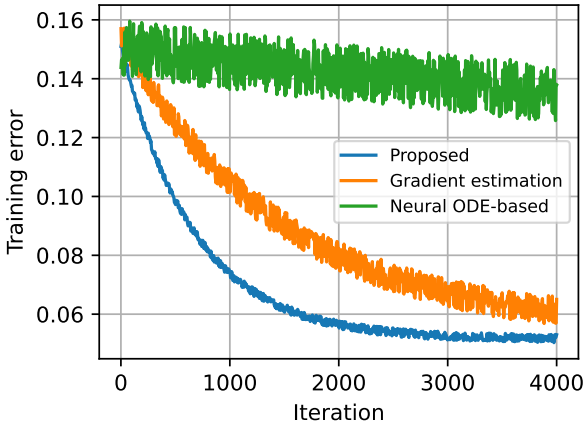


Figure 10: Training curves.

4.3 Calibration on data with missing values

Recall that our data analysis indicates that sensors at milepost 28.68, 30.03, 32.79, 34.44, 35.65, 37.39 and 40.85 malfunctioned could breakdown. Thus we generate the corrupted dataset by wiping out the traffic measurements at those locations.

Again, we considered the weight $\beta = 0.3$ and $\gamma = 1$. Then we compared our approach with the two benchmarks with the results are summarized in Table 5. We mainly focus on the comparison over the testing dataset since our learning-based approach does not have access this data during the training. Clearly, our approach achieves the best performance and even outperform the NP-based approach. The result is understandable. The NP-based approach lacks of the capability of generalization and thus they cannot use the data from other days to improve the calibration. By contrast, our learning-based approach can achieve stable calibration performance even given corrupted data with missing values.

Table 5: Performances of calibration on synthesized data with missing values.

	$e_{\text{flow}}^{\text{train}}$	$e_{\text{flow}}^{\text{test}}$	$e_{\text{flow}}^{\text{val}}$	$e_{\text{speed}}^{\text{train}}$	$e_{\text{speed}}^{\text{test}}$	$e_{\text{speed}}^{\text{val}}$	$e_{\text{param}}^{\text{train}}$	$e_{\text{param}}^{\text{test}}$	$e_{\text{param}}^{\text{val}}$
FD-based	3.86%	3.69%	3.81%	6.78%	6.47%	6.58%	6.16%	6.00%	6.12%
NP-based	3.26%	3.10%	3.20%	4.63%	4.48%	4.55%	4.30%	4.22%	4.23%
Proposed	3.29%	3.37%	3.42%	3.20%	3.29%	3.34%	1.93%	1.94%	1.99%

Finally, we verified our approach over the real-world data with missing values. We still chose the weight $\beta = 0.3$ and $\gamma = 1$. The performances of various calibration methods are summarized in Table 6. Comparing Table 6 with Table 4, we find that the performances of the three benchmark approaches degrade significantly. By contrast, our approach can still achieve stable error and thus perform better than the three benchmarks.

Table 6: Performances of calibration over real-world data with missing values.

	$e_{\text{flow}}^{\text{train}}$	$e_{\text{flow}}^{\text{test}}$	$e_{\text{flow}}^{\text{val}}$	$e_{\text{speed}}^{\text{train}}$	$e_{\text{speed}}^{\text{test}}$	$e_{\text{speed}}^{\text{val}}$
FD-based	5.08%	4.95%	4.98%	13.26%	13.63%	13.68%
NP-based	4.59%	4.70%	4.87%	7.86%	7.78%	7.96%
Proposed	4.46%	4.54%	4.68%	4.58%	4.55%	4.74%

5 Conclusion

In this paper, we propose a physics-informed, learning-based calibration approach, inspired by autoencoders. We consider calibrating the CTM, a widely-used traffic flow model. In our approach, the encoder takes as input traffic measurements and boundary conditions, and yields parameters required by CTM; the decoder recovers the measurements from the encoder output and the boundary conditions. Specially, we feed the decoder input to CTM and inform the autoencoder of a novel error between the decoder output and the simulation results besides the conventional error between the traffic measurements and the decoder output. This encourages the encoder to produce reasonable parameters so that the new error is minimized. We also introduce the denoising autoencoder

into our calibration method so that it can handles with corrupted data. Our case study of I210 E demonstrated that our approach can achieve comparable performance to the current optimization-based calibration approaches given normal traffic measurements and outperform them given corrupted traffic measurements. Possible future research includes calibrating high-order traffic models and online calibration.

Acknowledgement

This study was partially supported by US NSF Award CMMI-1949710, the C2SMART research center, a Tier 1 University Transportation Center, and Tandon School of Engineering of New York University. The contents of this paper only reflect views of the authors who are responsible for the facts and do not represent any official views of any sponsoring organizations or agencies.

References

Aw, A. and Rascle, M. (2000). Resurrection of" second order" models of traffic flow. *SIAM journal on applied mathematics*, 60(3):916–938.

Beintema, G., Toth, R., and Schoukens, M. (2021). Nonlinear state-space identification using deep encoder networks. In *Learning for Dynamics and Control*, pages 241–250. PMLR.

Caltrans (2022). *Caltrans Performance Measurement System*. `https://pems.dot.ca.gov/?logout=1`.

Choromanski, K. M., Davis, J. Q., Likhosherstov, V., Song, X., Slotine, J.-J., Varley, J., Lee, H., Weller, A., and Sindhwani, V. (2020). Ode to an ode. *Advances in Neural Information Processing Systems*, 33:3338–3350.

Dervisoglu, G., Gomes, G., Kwon, J., Horowitz, R., and Varaiya, P. (2009). Automatic calibration of the fundamental diagram and empirical observations on capacity. In *Transportation Research Board 88th Annual Meeting*, volume 15, pages 31–59. Citeseer.

Di, X., Shi, R., Mo, Z., and Fu, Y. (2023). Physics-informed deep learning for traffic state estimation: A survey and the outlook. *Algorithms*, 16(6):305.

Dowling, R., Skabardonis, A., Carroll, M., and Wang, Z. (2004). Methodology for measuring recurrent and nonrecurrent traffic congestion. *Transportation Research Record*, 1867(1):60–68.

Engl, H. W., Hanke, M., and Neubauer, A. (1996). *Regularization of inverse problems*, volume 375. Springer Science & Business Media.

Ferrara, A., Sacone, S., and Siri, S. (2018). *Freeway traffic modelling and control*. Springer.

Gedon, D., Wahlström, N., Schön, T. B., and Ljung, L. (2021). Deep state space models for nonlinear system identification. *IFAC-PapersOnLine*, 54(7):481–486.

Gomes, G. and Horowitz, R. (2006). Optimal freeway ramp metering using the asymmetric cell transmission model. *Transportation Research Part C: Emerging Technologies*, 14(4):244–262.

Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507.

Huang, J. and Agarwal, S. (2020). Physics informed deep learning for traffic state estimation. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6. IEEE.

Huang, K., Di, X., Du, Q., and Chen, X. (2020). Scalable traffic stability analysis in mixed-autonomy using continuum models. *Transportation research part C: emerging technologies*, 111:616–630.

Jaques, M., Burke, M., and Hospedales, T. (2020). Physics-as-inverse-graphics: Unsupervised physical parameter estimation from video. In *Eighth International Conference on Learning Representations*, pages 1–16.

Khoshyaran, M. M. and Lebacque, J. P. (2015). Capacity drop and traffic hysteresis as a consequence of bounded acceleration. *IFAC-PapersOnLine*, 48(1):766–771.

Kontorinaki, M., Spiliopoulou, A., Roncoli, C., and Papageorgiou, M. (2017). First-order traffic flow models incorporating capacity drop: Overview and real-data validation. *Transportation Research Part B: Methodological*, 106:52–75.

Kovachki, N., Li, Z., Liu, B., Azizzadenesheli, K., Bhattacharya, K., Stuart, A., and Anandkumar, A. (2021). Neural operator: Learning maps between function spaces. *arXiv preprint arXiv:2108.08481*.

Laval, J. A. and Leclercq, L. (2010). A mechanism to describe the formation and propagation of stop-and-go waves in congested freeway traffic. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 368(1928):4519–4541.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lee, J.-B. and Ozbay, K. (2008). Calibration of a macroscopic traffic simulation model using enhanced simultaneous perturbation stochastic approximation methodology. Technical report.

Lu, J., Li, C., Wu, X. B., and Zhou, X. S. (2023). Physics-informed neural networks for integrated traffic state and queue profile estimation: A differentiable programming approach on layered computational graphs. *Transportation Research Part C: Emerging Technologies*, 153:104224.

Masci, J., Meier, U., Cireşan, D., and Schmidhuber, J. (2011). Stacked convolutional auto-encoders for hierarchical feature extraction. In *Artificial Neural Networks and Machine Learning–ICANN 2011: 21st International Conference on Artificial Neural Networks, Espoo, Finland, June 14-17, 2011, Proceedings, Part I 21*, pages 52–59. Springer.

Masti, D. and Bemporad, A. (2021). Learning nonlinear state–space models using autoencoders. *Automatica*, 129:109666.

Messner, A. and Papageorgiou, M. (1990). Metanet: A macroscopic simulation program for motorway networks. *Traffic engineering & control*, 31(8-9):466–470.

Mo, Z., Fu, Y., and Di, X. (2022a). Quantifying uncertainty in traffic state estimation using generative adversarial networks. In *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, pages 2769–2774. IEEE.

Mo, Z., Fu, Y., Xu, D., and Di, X. (2022b). Trafficflowgan: Physics-informed flow based generative adversarial network for uncertainty quantification. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 323–339. Springer.

Mohammadian, S., Zheng, Z., Haque, M. M., and Bhaskar, A. (2021). Performance of continuum models for realworld traffic flows: Comprehensive benchmarking. *Transportation Research Part B: Methodological*, 147:132–167.

Mudigonda, S. and Ozbay, K. (2015). Robust calibration of macroscopic traffic simulation models using stochastic collocation. *Transportation Research Part C: Emerging Technologies*, 59:358–374.

Muñoz, L., Sun, X., Sun, D., Gomes, G., and Horowitz, R. (2004). Methodological calibration of the cell transmission model. In *Proceedings of the 2004 American Control Conference*, volume 1, pages 798–803. IEEE.

Nagel, T. and Huber, M. F. (2021). Autoencoder-inspired identification of lti systems. In *2021 European Control Conference (ECC)*, pages 2352–2357. IEEE.

Nguyen, H. H. N., Nguyen, T., Vo, H., Osher, S., and Vo, T. (2022). Improving neural ordinary differential equations with nesterov's accelerated gradient method. *Advances in Neural Information Processing Systems*, 35:7712–7726.

Papamichail, I., Kotsialos, A., Margonis, I., and Papageorgiou, M. (2010). Coordinated ramp metering for freeway networks–a model-predictive hierarchical control approach. *Transportation Research Part C: Emerging Technologies*, 18(3):311–331.

Pascanu, R., Mikolov, T., and Bengio, Y. (2013). On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318. Pmlr.

Payne, H. (1971). Models of freeway traffic and control. mathematical models of public systems. simulation councils. *Inc., Vista, CA, USA*.

Schoukens, J. and Ljung, L. (2019). Nonlinear system identification: A user-oriented road map. *IEEE Control Systems Magazine*, 39(6):28–99.

Shi, R., Mo, Z., and Di, X. (2021a). Physics-informed deep learning for traffic state estimation: A hybrid paradigm informed by second-order traffic models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 540–547.

Shi, R., Mo, Z., Huang, K., Di, X., and Du, Q. (2021b). A physics-informed deep learning paradigm for traffic state and fundamental diagram estimation. *IEEE Transactions on Intelligent Transportation Systems*.

Shi, X. and Li, X. (2021). Constructing a fundamental diagram for traffic flow with automated vehicles: Methodology and demonstration. *Transportation Research Part B: Methodological*, 150:279–292.

Spiliopoulou, A., Kontorinaki, M., Papageorgiou, M., and Kopelias, P. (2014). Macroscopic traffic flow model validation at congested freeway off-ramp areas. *Transportation Research Part C: Emerging Technologies*, 41:18–29.

Spiliopoulou, A., Papamichail, I., Papageorgiou, M., Tyrinopoulos, Y., and Chrysoulakis, J. (2017). Macroscopic traffic flow model calibration using different optimization algorithms. *Operational Research*, 17(1):145–164.

Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. (2008). Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103.

Wang, Y., Yu, X., Guo, J., Papamichail, I., Papageorgiou, M., Zhang, L., Hu, S., Li, Y., and Sun, J. (2022). Macroscopic traffic flow modelling of large-scale freeway networks with field data verification: State-of-the-art review, benchmarking framework, and case studies using metanet. *Transportation Research Part C: Emerging Technologies*, 145:103904.

Whitham, G. B. (1974). Linear and nonlinear waves(book). *New York, Wiley-Interscience, 1974. 651 p.*

Yang, T.-Y., Rosca, J., Narasimhan, K., and Ramadge, P. J. (2022). Learning physics constrained dynamics using autoencoders. *Advances in Neural Information Processing Systems*, 35:17157–17172.

Yuan, Y., Wang, Q., and Yang, X. T. (2021a). Traffic flow modeling with gradual physics regularized learning. *IEEE Transactions on Intelligent Transportation Systems*.

Yuan, Y., Zhang, Z., and Yang, X. T. (2020). Highway traffic state estimation using physics regularized gaussian process: Discretized formulation. *arXiv preprint arXiv:2007.07762*.

Yuan, Y., Zhang, Z., Yang, X. T., and Zhe, S. (2021b). Macroscopic traffic flow modeling with physics regularized gaussian process: A new insight into machine learning applications in transportation. *Transportation Research Part B: Methodological*, 146:88–110.

Zhang, H. M. (2002). A non-equilibrium traffic model devoid of gas-like behavior. *Transportation Research Part B: Methodological*, 36(3):275–290.

Zhong, R., Chen, C., Chow, A. H., Pan, T., Yuan, F., and He, Z. (2016). Automatic calibration of fundamental diagram for first-order macroscopic freeway traffic models. *Journal of Advanced Transportation*, 50(3):363–385.