2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Physics-Informed Neural Networks for Learning the Parameters of Commercial Adaptive Cruise Control Systems

Theocharis Apostolakis and Konstantinos Ampountolas

Abstract— **This paper develops a *physics-informed neural network* (PINN) for learning the parameters of commercially implemented adaptive cruise control (ACC) systems. The constant time-headway policy (CTHP) is adopted to emulate the core functionality of stock ACC systems (proprietary control logic and its parameters) which is not publicly available. Multi-layer artificial neural networks is a class of *universal approximators*, and thus the developed PINN can serve as a surrogate approximator to capture the longitudinal dynamics of ACC-engaged vehicles and efficiently learn the unknown parameters of the CTHP. The ability of the PINN to infer the unknown ACC parameters is tested on both synthetic and empirical data of space-gap and relative velocity involved ACC-engaged vehicles in platoon formation. The results have demonstrated the superior predictive ability of the proposed PINN to learn the unknown design parameters of stock ACC systems of different vehicle makes. The set of ACC model parameters obtained from the PINN revealed that the stock ACC system of the considered vehicles in three experimental campaigns is neither $\mathcal{L}_2$ nor $\mathcal{L}_\infty$ string stable.**

I. Introduction

The growing acceptance of partially automated vehicles (up to SAE Level 2 of Driving Automation [1]) on public roads gave birth to new traffic conditions, rising concerns among the scientific community regarding their impact on traffic flow and capacity. This created the need for a deeper understanding of the fundamental principles underlying these vehicles. Adaptive cruise control (ACC) equipped vehicles serve as a typical example of such vehicles. ACC systems have long been part of automotive equipment (optional or standard). They provide additional assistance to the driver by controlling the longitudinal movement of the vehicle while monitoring the surrounding environment with several onboard sensors (radar, lidar, etc.). More specifically, ACC is an advanced driver-assistance system (ADAS) that automatically adjusts the speed of the vehicle, accelerating or decelerating it, to maintain a safe predefined distance from the vehicle in front or to reach the user-specified speed.

However, ACC's design (e.g., controller synthesis) remains publicly unknown, making ACC systems not fully understood yet. One of the key aspects behind ACC's design is the spacing policy (control law) adopted by ACC manufacturers, and its unknown design parameters. The spacing policy specifies the predefined desired distance (based on time or space) between an ACC vehicle and its vehicle in front. The constant time-headway policy (CTHP) [2] is one of the most remarkable spacing control strategies developed in the literature [2]–[4]. Despite its simplicity, it can reproduce the dynamics and driving behavior of ACC-engaged vehicles in platoons, as shown in several field trials [5]–[7].

Among others, a major issue concerning the scientific community for decades is whether ACC systems are string stable inside a platoon in the presence of unknown but bounded disturbances [8]–[12]. String stability is defined as the elimination of propagating disturbances upstream the platoon. String unstable platoons are intruding since their dynamics can lead to phantom traffic shockwaves and thus to traffic congestion and poor system throughput. In the literature, it has been repeatedly reported that these systems tend to be string unstable [5], [6], [13]–[15]. To better understand such phenomena, a broader study of ACC's principles is required.

ACC system's parameter identification is an important step towards its better understanding. Several studies have used different techniques based on synthetic or observed data, such as batch optimization, recursive least-squares, particle filtering, and unscented Kalman filtering [15]–[18]. However, the parameter identification problem of automated vehicles using empirical observations is challenging, since the underlying optimization problem is non-convex and it might be ill-conditioned under equilibrium driving conditions (i.e., where acceleration and space-gap reduce to zero), see [15]. In the latter case, the ACC system parameters cannot be uniquely identified, given input and output observations from the platoon, since the problem lacks both linear and nonlinear observability [18].

This paper introduces, for the first time in the relevant literature, physics-informed neural networks (PINNs) for the parameter learning of commercially implemented ACC systems using empirical observations of space-gap and relative velocity. The CTHP, which is speculated that is implemented in stock ACC systems of various makes, is adopted to emulate the longitudinal motion of ACC vehicles flocking in homogeneous platoons. The pursued PINN is a deep learning data-driven approach subject to the physical model of the CTHP derived from first physical principles (double integrator) and control theory (a PID-like control law).

PINNs are semi-supervised artificial neural networks of dynamical systems governed by ordinary or partial differential equations (ODEs or PDEs) and observed data [19]–[22]. A PINN consists of two ingredients: An artificial neural network representing a *physics-uninformed* surrogate

This work was supported by the Center of Research, Innovation and Excellence (CRIE) of the University of Thessaly, under the project “campaigningACC”.

Theocharis Apostolakis and Konstantinos Ampountolas are with the Automatic Control and Autonomous Systems Laboratory, Department of Mechanical Engineering, University of Thessaly, 38334 Volos, Greece (e-mail: tapostolakis@uth.gr, k.ampountolas@uth.gr).

predictor (parameterized by weights and biases); and (b) a residual network representing a *physics-and-data-informed* set of ODEs or PDEs. PINN's implementation expands into solving both forward and inverse non-linear problems [20].

The remainder of the paper is structured as follows. Section II reviews the CTHP and briefly presents the $\mathcal{L}_2$ and $\mathcal{L}_\infty$ criteria for string stability in platoons. Section III introduces the proposed PINN for parameter learning. Section IV illustrates the application of PINN to both synthetic and empirical data obtained from three real-life campaigns. It also presents the inferred parameters of stock ACC systems for various makes and discusses their string stability conditions. Section V outlooks the potential of this work.

Notation: The field of real numbers is denoted by $\mathbb{R}$. The space of Lebesgue measurable functions $f : \mathbb{R} \to \mathbb{R}$ such that $t \to |f(t)|^{\mathsf{p}}$ is integrable over $\mathbb{R}$ is denoted by $\mathcal{L}_{\mathsf{p}}$, here $\mathsf{p} = 2, \infty$ is used to discuss string stability. For $\mathsf{p} = \infty$ no integration is used, and instead, the norm on $\mathcal{L}_\infty$ is given by the essential supremum.

II. Modeling Adaptive Cruise Control Systems

In the sequel, the CTHP, embedded in a simplified dynamic vehicle model, is considered to imitate the longitudinal motion of ACC-engaged vehicles flocking in homogeneous platoons. The vehicle dynamics of the ACC vehicles are described by a double integrator, while acceleration is governed by the CTHP. Despite its simplicity, this model is able to reproduce the dynamics and driving behavior of ACC-engaged vehicles in platoons as shown in field trials [5]–[7].

A. ACC Platooning with Constant Time-Headway Policy

Consider a platoon of M homogeneous ACC-engaged ego vehicles, where vehicle i (follower) follows vehicle $i-1$ (leader). The leader of the platoon is indexed by $i = 0$ and might be affiliated to a human-driven vehicle (HDV). The longitudinal dynamics of the ACC equipped vehicles can be described by the following system of ODEs [2], [9]:

$$\dot{p}_i(t) = \Delta v_i(t), \quad i = 1, 2, \dots, M, \tag{1}$$

$$\dot{v}_i(t) = \alpha\big[p_i(t) - \tau v_i(t)\big] + \beta \Delta v_i, \tag{2}$$

where $p_i(t)$ [m] is the space-gap between two vehicles i and $i-1$, i.e., the distance between the follower's front bumper and the leader's rear bumper; and $\Delta v = v_{i-1}(t) - v_i(t)$ is the relative velocity between the vehicle i and the vehicle $i-1$ in a platoon. In the control law (2), τ [s] represents the constant time-headway the ACC-engaged vehicle i strives to maintain with its leader $i-1$, which in turn indicates the desired space-gap, $\tau v_i(t)$, between the two vehicles. The two non-negative gains, α [1/s^2] and β [1/s], control the trade-off between the space-gap difference, $p_i(t) - \tau v_i(t)$, and the relative velocity Δv_i. Finally, parameter τ [s] can also be considered as the time-gap under equilibrium driving conditions:

$$v_{i-1}(t) - v_i(t) = 0 \ \text{ and } \ p_i(t) - \tau v_i(t) = 0, \ \forall\, i. \tag{3}$$

The CTHP parameters α, β and τ that characterize the ACC equipped vehicles are constant but unknown. Thus, the model (1)–(2) can be re-written in compact form as:

$$\dot{\boldsymbol{\xi}}_i(t) = \mathbf{f}[\boldsymbol{\xi}_i(t), \boldsymbol{\omega}], \quad i = 1, 2, \dots, M, \tag{4}$$

where $\boldsymbol{\xi}_i(t) = [p_i(t) \ v_i(t)]^{\mathsf{T}}$ is the state vector, $\boldsymbol{\omega} = [\alpha \ \beta \ \tau]^{\mathsf{T}}$ is the parameters vector (to be learned), and $\mathbf{f} = [f_1 \ f_2]^{\mathsf{T}}$ is a vector function that reflects the right-hand side of (1)–(2). Note that one set of parameters $\boldsymbol{\omega}$ is considered here for all vehicles within the platoon due to the homogeneity assumption above.

B. Parameter Identification and String Stability

For the CTHP, the following conditions for string stability in the sense of $\mathcal{L}_2$ and $\mathcal{L}_\infty$ norms are well-established in the relevant literature [23], [24]. The CTHP model parameters α, β and τ must satisfy,

1) $\mathcal{L}_2$ Strict String Stability:

$$\alpha^2\tau^2 + 2\alpha\beta\tau - 2\alpha \geq 0. \tag{5}$$

As can be seen, as τ approaches ∞ the system is $\mathcal{L}_2$ strict string stable for all non-negative gains α and β of the CTHP, while as τ approaches zero the system becomes unstable.

2) $\mathcal{L}_\infty$ Strict String Stability:

$$(\alpha\tau + \beta)^2 - 4\alpha \geq 0. \tag{6}$$

Subtracting (5) from (6) yields [24]:

$$\beta^2 \geq 2\alpha \ \Rightarrow \ (\mathfrak{L}_\infty \text{ stability} \Leftrightarrow \mathfrak{L}_2 \text{ stability}), \tag{7}$$

suggesting that $\mathcal{L}_2$ stability is stronger than the $\mathcal{L}_\infty$ stability. This is realistic since even if the $\mathcal{L}_2$ energy of a signal is small, it may occasionally contain large peaks, provided the peaks (i.e., the $\mathcal{L}_\infty$ norm) are not too frequent and do not contain too much energy.

III. Physics-Informed Neural Networks

A. Architecture of Multi-layer Neural Networks

Consider a fully-connected feed-forward artificial neural network (ANN or NN), $\mathcal{N}^L(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^m$ of L (or $L-1$ hidden) layers, with N_ℓ denoting the number of artificial neurons at layer $\ell = 1, 2, \dots, L-1$, while for the input and output layers $N_0 = n$ and $N_L = m$ holds, respectively. Thus, the input layer has the dimension of the raw training data $\mathbf{x} \in \mathbb{R}^n$, while the dimension of the output layer is defined by the context. Each neuron at layer ℓ is equipped with a (user-defined) nonlinear activation function, $\sigma(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^n$, to transform the weighted linear sum input of various neurons at layer $\ell - 1$ into an output that is passed (neuron fires) on to the next (hidden or output) layer $\ell + 1$. The weights and bias of the weighted sum input at layer $\ell = 1, 2, \dots, L$ are organized in the matrices $\mathbf{A}^\ell \in \mathbb{R}^{N_\ell \times N_{\ell-1}}$ and vectors $\mathbf{b}^\ell \in \mathbb{R}^{N_\ell}$, respectively; or in a single parameters vector $\boldsymbol{\theta} = \{\mathbf{A}^\ell, \mathbf{b}^\ell\}_{1 \leq \ell \leq L}$. The architecture of the ANN can then be summarized as the following compositional function:

$$\begin{aligned} \textit{Input layer:} \quad & \mathcal{N}^0(\mathbf{x}) = \mathbf{x} \in \mathbb{R}^n, \\ \textit{Hidden layers:} \quad & \mathcal{N}^\ell(\mathbf{x}) = \sigma(\mathbf{A}^\ell \mathcal{N}^{\ell-1}(\mathbf{x}) + \mathbf{b}^\ell) \in \mathbb{R}^{N^\ell}, \\ \textit{Output layer:} \quad & \mathcal{N}^L(\mathbf{x}) = \mathbf{A}^L \mathcal{N}^{L-1}(\mathbf{x}) + \mathbf{b}^L \in \mathbb{R}^m, \end{aligned}$$

where $\ell = 1, 2, \ldots, L-1$ for the hidden layers.

The goal is to optimize the weighting matrices $\mathbf{A}^\ell$ and bias vectors $\mathbf{b}^\ell$ connecting the NN from the ℓ-th to $(\ell+1)$-th layer using labeled pairs of input (see *Input layer*) and output (see *Output layer*) data. This procedure, which is known as *training* in the NN nomenclature, involves *automatic differentiation* [25] over the compositional function $\mathcal{N}^L(\mathbf{x})$ and nonlinear optimization, and thus is computationally expensive. *Stochastic gradient descent* (SGD) [26], [27] and *back-propagation* [28] are two important algorithms that can be used for the efficient training of such multi-layer ANN.

B. ACC Parameter Learning via PINNs

Consider the parameterized ODEs (1)–(2) that characterize the CTHP of ACC equipped vehicles. Assume that a solution, $\boldsymbol{\xi}_i(t)$ on the time domain $t \in [t_0, t_e] = \Omega$, of the ODEs exists, given some boundary conditions $\mathcal{B}_i(\boldsymbol{\xi}_i, t) = \mathbf{0}$ on $\partial\Omega$:

$$\mathcal{F}_i(\boldsymbol{\xi}_i, \dot{\boldsymbol{\xi}}_i, \boldsymbol{\omega}, t) = \dot{\boldsymbol{\xi}}_i(t) - \mathbf{f}[\boldsymbol{\xi}_i(t), \boldsymbol{\omega}] = \mathbf{0}, i = 1, \ldots, M, \quad (8)$$

where $\mathcal{F}_i \in \mathbb{R}^2$ is a nonlinear operator (residual of (4)) parameterized by $\boldsymbol{\xi}_i$, $\dot{\boldsymbol{\xi}}_i$, and $\boldsymbol{\omega}$ (to be learned).

Multi-layer feed-forward ANN are a class of *universal approximators* [29], thus a neural network, $\hat{\boldsymbol{\xi}}_i(t; \boldsymbol{\theta})$, can be developed as a surrogate of the solution $\boldsymbol{\xi}_i(t)$ for all $i = 1, 2, \ldots, M$. Provided the availability of empirical data of space-gap and relative velocity, $\mathcal{D} = \{\boldsymbol{\xi}_i(t)\}_{t \in \Omega}$, the (output of the) neural network $\hat{\boldsymbol{\xi}}_i(t; \boldsymbol{\theta})$ (predictor) can be constrained to satisfy the physical model (8) of the CTHP and its boundary conditions $\mathcal{B}_i(\boldsymbol{\xi}_i, t) = \mathbf{0}$ on $\partial\Omega$. Additional internal conditions, $\mathcal{I}_i(\boldsymbol{\xi}_i, t) = \mathbf{0}$ on some $t \subset \Omega$, can be also incorporated for readily solving the *inverse parameter optimization problem.* In the inverse problem, the vector of CTHP parameters is to be learned using training data, $\mathcal{D}$, such that (8) and boundary/internal conditions are satisfied.

Concluding, a PINN for each ACC vehicle i on a platoon consists of (see Fig. 1): (a) the *physics-uninformed* surrogate predictor $\hat{\boldsymbol{\xi}}_i(t; \boldsymbol{\theta})$; (b) the *physics-informed* residual constraints $\mathcal{F}_i(\boldsymbol{\xi}_i, \dot{\boldsymbol{\xi}}_i, \boldsymbol{\omega}, t)$ together with the boundary and internal conditions $\mathcal{R}_i(\boldsymbol{\xi}_i, t) = \{\mathcal{B}_i(\boldsymbol{\xi}_i, t) \cup \mathcal{I}_i(\boldsymbol{\xi}_i, t)\}$. Upon training, the PINN is calibrated to predict the entire solution of the system of ODEs (8), as well as the unknown CTHP parameters that define the underlying dynamics in a platoon.

C. Training of the PINN

Training of the PINN requires discretization or sampling of the continuous time domain on a sufficient small time interval, resulting to the set of pseudo-points (or induced points) $\mathcal{T} = \{t_1, t_2, \ldots, t_{|\mathcal{T}|}\} \in \Omega$, where $|\mathcal{T}|$ is the cardinality of set $\mathcal{T}$. The set $\mathcal{T} = \{\mathcal{T}_f, \mathcal{T}_b\}$ includes all points in the time domain $\mathcal{T}_f \subset \Omega$ and its boundary $\mathcal{T}_b \subset \partial\Omega$ where evaluation of the residual (8) and boundary and internal conditions $\mathcal{R}_i$ is necessary, given predictions of the surrogate model $\hat{\boldsymbol{\xi}}_i(t; \boldsymbol{\theta})$ and empirical data of space-gap and relative-velocity, $\mathcal{D} = \{\boldsymbol{\xi}_i(t)\}_{t \in \mathcal{T}}$. Thus, the NN takes as input the set $\mathcal{T}$ and provides predictions of $\hat{\boldsymbol{\xi}}_i = [\hat{p}_i \ \hat{v}_i]^\mathsf{T}$ (see the input and output layers, respectively, in Fig. 1).

To efficiently train the surrogate predictor $\hat{\boldsymbol{\xi}}_i(t; \boldsymbol{\theta})$ and simultaneously satisfy the residual and boundary constraints, the following cost criterion of two terms is considered: (a) a semi-unsupervised cost criterion, $\Psi_{\mathrm{ODE|U}}$, for the residual (8) and its boundary and internal conditions; (b) a supervised (data-driven) cost criterion, $\Psi_{\mathrm{D|S}}$, governed by the measurements $\boldsymbol{\xi}_i$ and the predictions $\hat{\boldsymbol{\xi}}_i$. For the inverse optimization problem the cost criterion reads:

$$\begin{aligned}\Psi(\boldsymbol{\theta}, \boldsymbol{\omega}) &= \Psi_{\mathrm{ODE|U}} + \Psi_{\mathrm{D|S}} \\ &= \underbrace{\Psi_{\mathcal{F}}(\boldsymbol{\theta}, \boldsymbol{\omega}) + \Psi_{\mathcal{R}}(\boldsymbol{\theta}, \boldsymbol{\omega})}_{\Psi_{\mathrm{ODE|U}}} + \underbrace{\Psi_{\mathcal{D}}(\boldsymbol{\theta})}_{\Psi_{\mathrm{D|S}}}, \end{aligned} \quad (9)$$

where,

$$\Psi_{\mathcal{F}}(\boldsymbol{\theta}, \boldsymbol{\omega}) = \frac{1}{|\mathcal{T}_f| \times M} \sum_{i=1}^{M} \sum_{t \in \mathcal{T}_f} \|\mathcal{F}_i(\hat{\boldsymbol{\xi}}_i, \dot{\hat{\boldsymbol{\xi}}}_i, \boldsymbol{\omega}, t)\|^2_{\mathbf{Q}}, \quad (10)$$

$$\Psi_{\mathcal{R}}(\boldsymbol{\theta}, \boldsymbol{\omega}) = \frac{1}{|\mathcal{T}_b| \times M} \sum_{i=1}^{M} \sum_{t \in \mathcal{T}_b} \|\mathcal{R}_i(\hat{\boldsymbol{\xi}}_i, \boldsymbol{\omega}, t)\|^2_{\mathbf{R}}, \quad (11)$$

$$\Psi_{\mathcal{D}}(\boldsymbol{\theta}) = \frac{1}{|\mathcal{T}| \times M} \sum_{i=1}^{M} \sum_{t \in \mathcal{T}} \|\psi_i(\hat{\boldsymbol{\xi}}_i, \boldsymbol{\xi}_i, t)\|^2_{\mathbf{S}}, \quad (12)$$

and ψ is a function that penalizes deviations of the NN surrogate approximation from the empirical training data set $\mathcal{D} = \{\boldsymbol{\xi}_i(t)\}_{t \in \mathcal{T}}$, e.g., for the mean square error (MSE), $\psi_i(\hat{\boldsymbol{\xi}}_i, \boldsymbol{\xi}_i, t) = \hat{\boldsymbol{\xi}}_i - \boldsymbol{\xi}_i$. The positive-definite weighting matrices $\mathbf{Q}$, $\mathbf{R}$, $\mathbf{S}$ are penalty terms for the unsupervised and supervised cost functions, respectively. These matrices can be selected via a trial-and-error procedure to speed up convergence for a particular data set and application.

The optimal vectors of the inverse problem, NN weights $\boldsymbol{\theta}$ and CTHP parameters $\boldsymbol{\omega}$, can be obtained by solving the following optimization problem during PINN training:

$$[\boldsymbol{\theta}^*, \boldsymbol{\omega}^*] = \arg \min_{\boldsymbol{\theta}, \boldsymbol{\omega}} \Psi(\boldsymbol{\theta}, \boldsymbol{\omega}). \quad (13)$$

This is a highly nonlinear optimization problem, and thus is computationally expensive; but can be solved efficiently using (batch) SGD or quasi-Newton methods using Hessian information (e.g., Adam [30] and L-BFGS [31]).

IV. Application and Results

To demonstrate the effectiveness of the proposed data-driven approach to learn the parameters of the CTHP of ACC systems, a PINN is developed and tested to both synthetic and empirical data of space-gap and relative velocity.

A. Empirical Data Description

The empirical data are taken from three car-following experimental campaigns that took place in Autostrada A26 motorway in Ispra-Vicolungo (fleet of five vehicles in platoon formation) and Ispra-Casale (car-following scenario with two vehicles) routes in 2019 and 2020, respectively, Italy, and AstaZero test track (five premium ACC equipped vehicles in platoon formation) in mid 2019, Sweden. In all campaigns, data acquisition was performed using high-accuracy on-board equipment (e.g., U-blox GNSS receivers) with a sampling

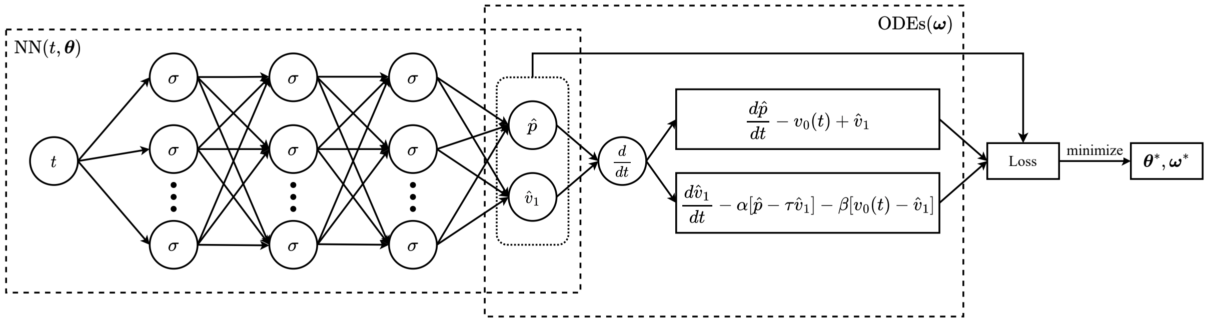

NN($t, \boldsymbol{\theta}$)
ODEs($\boldsymbol{\omega}$)
t
σ
$\hat{p}$
$\hat{v}_1$
$\frac{d}{dt}$
$\frac{d\hat{p}}{dt} - v_0(t) + \hat{v}_1$
$\frac{d\hat{v}_1}{dt} - \alpha[\hat{p} - \tau\hat{v}_1] - \beta[v_0(t) - \hat{v}_1]$
Loss
minimize
$\boldsymbol{\theta}^*, \boldsymbol{\omega}^*$

Fig. 1. Architecture of the proposed Physics-Informed Neural Network (PINN) for CTHP parameter learning. The NN on the left represents the *physics-uninformed* surrogate predictor $\hat{\boldsymbol{\xi}}(t;\boldsymbol{\theta})$, while the right network depicts the *physics-and-data-informed* residual and boundary constraints of the CTHP.

frequency at 10 Hz (0.1 s). Data can be freely accessed through the OpenACC database [13]. Table II provides information (make and model) on the vehicles involved.

B. PINN Setup

Fig. 1 depicts the developed PINN for learning the unknown design parameters of the CTHP, $\boldsymbol{\omega}$, for a particular ego vehicle i in a car-following scenario (thus the index i is omitted in the sequel), $\hat{\boldsymbol{\xi}}(t;\boldsymbol{\theta}) \equiv \text{NN}(t,\boldsymbol{\theta})$, consisting of one (1) input layer with one (1) neuron t (the discretized time domain), three (3) hidden-layers with sixty (60) neurons each, and one (1) output layer with two (2) neurons, the predictor $\hat{\boldsymbol{\xi}} = [\hat{p}\ \ \hat{v}]^\mathsf{T}$. Thus, $L = 4$, $N_0 = 1$, $N_\ell = 60$ for $\ell = 1,2,3$, and $N_4 = 2$. The weights vector of the NN reads $\boldsymbol{\theta} = \{\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \mathbf{A}^{(3)}, \mathbf{A}^{(4)}, \mathbf{b}^{(1)}, \mathbf{b}^{(2)}, \mathbf{b}^{(3)}, \mathbf{b}^{(4)}\} \in \mathbb{R}^{7562}$. Each neuron of the NN is equipped with a non-linear hyperbolic tangent activation function, $\sigma(x) = \tanh(x)$.

For the inverse optimization problem for a car-following scenario with two vehicles, $M = 1$ (the extension to $M > 1$ is straightforward), the cost criterion (9) reads:

$$\Psi_{\mathcal{F}}(\boldsymbol{\theta},\boldsymbol{\omega}) = \frac{1}{|\mathcal{T}_f|}\sum_{t\in\mathcal{T}_f}\Big\{\big[\hat{\dot{p}}_1(t) - v_0(t) + \hat{v}_1(t)\big]^2 + \big[\hat{\dot{v}}_1(t) - \alpha[\hat{p}_1(t) - \tau\hat{v}_1(t)] - \beta[v_0(t) - \hat{v}_1(t)]\big]^2\Big\},$$

$$\Psi_{\mathcal{D}}(\boldsymbol{\theta}) = \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}}\Big\{[p_1(t) - \hat{p}_1(t)]^2 + [v_1(t) - \hat{v}_1(t)]^2\Big\},$$

where v_0 and v_1 are the velocities of the leader and the ACC follower, respectively; and p_1 is the space-gap between the two vehicles. Note that the term $\Psi_{\mathcal{R}}$ is absent above since $|\mathcal{T}_b| = |\mathcal{T}|$ and, thus, the internal and boundary conditions $\mathcal{R}$ are included in $\Psi_{\mathcal{F}}$ and $\Psi_{\mathcal{D}}$. To train the NN the learning rate is set to 0.001 and the maximum number of iterations is set to 60,000. In each iteration, the input of the NN is fed with a set of $|\mathcal{T}| = |\mathcal{T}_f| = |\mathcal{T}_b| = 3000$ collocation training pseudo-points sampled inside the time domain of $t = [0,300]$ s at a frequency of 10 Hz (0.1 s), i.e. $\mathcal{T} = \{t_1, t_2, \ldots, t_{|\mathcal{T}|=3000}\}$.

Training of the PINN takes about 7 CPU-minutes on an 11th Generation Intel(R) Core(TM) i7-11700K @ 3.60GHz with 8 cores on Windows 10 Pro 64-bit. Best parameter convergence is achieved in around 20,000 iterations for both tests on synthetic and empirical data.

C. CTHP Parameter Learning on Synthetic Data

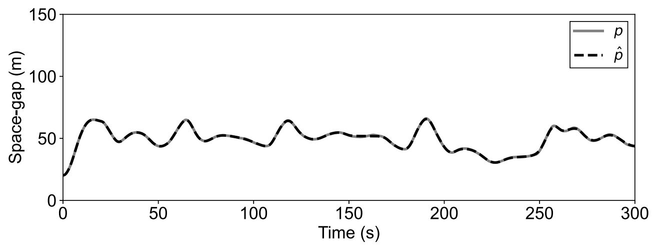
Initially, synthetic data with known CTHP parameters, $\boldsymbol{\omega}$, were generated and used to validate the proposed parameter learning approach. To this end, consider a car-following-scenario with two vehicles, a leader (HDV) and a follower (ACC ego vehicle), a known set of design parameters $\boldsymbol{\omega}^* = [\alpha^*\ \beta^*\ \tau^*]^\mathsf{T} = [0.08\ \ 0.12\ \ 1.5]^\mathsf{T}$, and a predefined leader's velocity profile taken from the Ispra-Casale campaign (solid blue line in Fig. 2b). Then, synthetic data of $p(t)$ and $v(t)$, for $t > 0$, were generated from the system of ODEs (1)–(2) (with $M = 1$), using the a priori known $\boldsymbol{\omega}^*$ and the initial condition $\boldsymbol{\xi}(0) = [p(0)\ v(0)]^\mathsf{T} = [20.3\ \ 21.3]^\mathsf{T}$ (in m and m/s, respectively). The data was generated for 300 s with a frequency of 10 Hz (0.1 s).

Figs 2a–2b display the obtained trajectories of space-gap $p(t)$ and velocity $v(t)$ (both in solid lines). As can be seen, both equilibrium (3) and non-equilibrium driving conditions are considered in the synthetic data set. Note that the known set of design parameters, $\boldsymbol{\omega}^*$, corresponds to a string unstable system in terms of both the $\mathcal{L}_2$ and $\mathcal{L}_\infty$ norms, check (5) and (6), respectively. Thus the ACC-engaged vehicle is expected to amplify any random perturbations of the HDV leader.

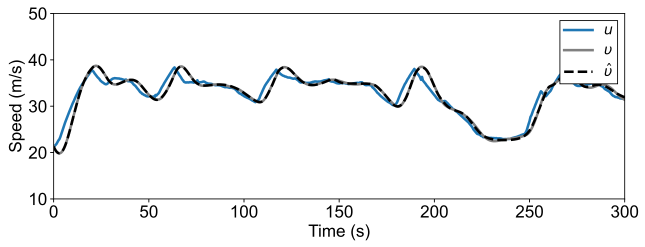
The synthetic data were used to train the PINN described in Section IV-B towards learning the a priori known vector of CTHP parameters, $\boldsymbol{\omega}^*$, of the ACC-engaged vehicle. Fig. 3a depicts the obtained results on the learning trajectories of $\hat{\alpha}$, $\hat{\beta}$, and $\hat{\tau}$ converging successfully to $\hat{\boldsymbol{\omega}} = [0.0784\ \ 0.12\ \ 1.5]^\mathsf{T}$ after 20,000 iterations. The PINN is also calibrated to predict the entire trajectories of $p(t)$ and $v(t)$ corresponding to the true parameter values. Fig. 2 presents the estimated values of $p(t)$ and $v(t)$ in each point inside the entire time domain, with mean absolute errors (MAEs) of 0.0939 m and 0.1509 m/s, respectively. This experiment underlines the ability of the developed framework to successfully learn the true values of the CTHP parameters and reconstruct the full range of dynamics in the space-gap and velocity profiles.

D. CTHP Parameter Learning on Empirical Data

This section demonstrates the superior predictive ability of the proposed PINN to learn the unknown design parameters of stock ACC systems of different makes (see Table II), using empirical data of space-gap and relative velocity from

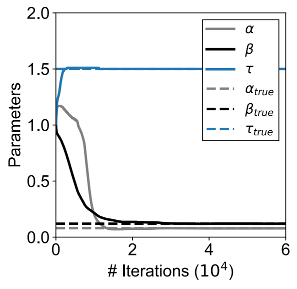


(a) Space-gap: Empirical data, solid line; PINN prediction, dashed line.

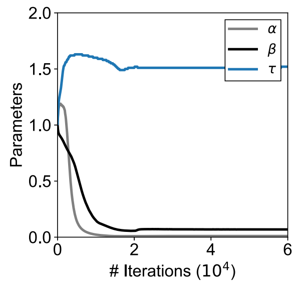


(b) Velocity: Empirical data, solid lines; PINN prediction, dashed line.

Fig. 2. Space-gap and velocity trajectories for the synthetic dataset.



(a) Synthetic data



(b) Ispra-Casale (Exp. #1)

Fig. 3. CTHP parameter learning and convergence.

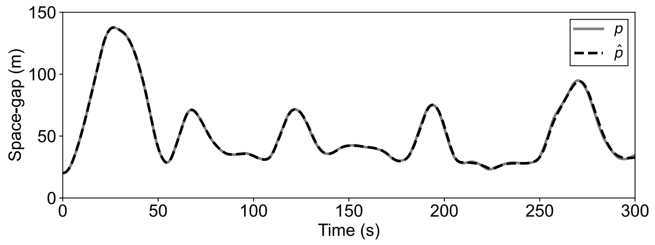
three experimental campaigns, namely Ispra-Casale, Ispra-Vicolungo, and AstaZero. It also aims to examine whether the stock ACC systems of various makes are strict string stable inside platoons using the obtained ACC parameters and the string stability criteria (5) and (6) from Section II.

Parameter Learning for Ispra-Casale: To investigate the sensitivity of parameter learning on different initial points $\hat{\boldsymbol{\omega}}(0)$, five different training trials of the PINN were carried out. Fig. 4 displays the empirical data of space-gap and relative velocity (solid lines) used for training. As can be seen, the leader is engaged in various acceleration perturbations, while the follower seems to be string unstable. Table I shows the obtained results from PINN's training on five different trials with α, β and τ starting from different initial values. As we can see, α and β differ to the third decimal place, while τ converges to the same value in each experiment. Inserting these values in the string stability criteria, it turns out that the ACC vehicle is neither $\mathcal{L}_2$ nor $\mathcal{L}_\infty$ strict string stable. Fig. 3b shows the parameters' convergence from the first experiment in Table I, while Fig. 4 displays the estimated trajectories of space-gap and speed of the ACC vehicle, with their MAEs being relatively low (see Table I).

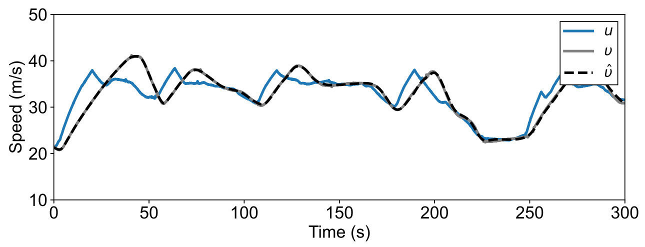
TABLE I

PARAMETER ESTIMATION FOR ISPRA-CASALE CAMPAIGN.

Experiment	#1	#2	#3	#4	#5
α (1/s^2)	0.0104	0.0104	0.0104	0.0102	0.0103
β (1/s)	0.0718	0.0712	0.0723	0.0709	0.0724
τ (s)	1.52	1.52	1.52	1.52	1.52
MAE space-gap (m)	0.4721	0.2653	0.4667	0.4519	0.3590
MAE speed (m/s)	0.1419	0.0871	0.1507	0.2183	0.1361
$\mathcal{L}_2$ strict string stability	NO	NO	NO	NO	NO
$\mathcal{L}_\infty$ strict string stability	NO	NO	NO	NO	NO



(a) Space-gap: Empirical data, solid line; PINN prediction, dashed line.



(b) Velocity: Empirical data, solid lines; PINN prediction, dashed line.

Fig. 4. Space-gap and velocity trajectories for Ispra-Casale (Exp. #1).

Parameter Learning for AstaZero and Ispra-Vicolungo: In both campaigns, data concerns the speed of each vehicle in the platoon and the space-gaps between them for a duration of 300 s. We aim to apply the suggested PINN to learn the ACC design parameters of each follower and assess the string stability of the platoons, examining whether the disturbances entering the platoons are dissipated upstream or not.

Table II summarizes the obtained CTHP parameter estimates from the application of the PINN in both experimental campaigns. As can be seen, the ACC design parameters α, β, and τ have relatively close values across the different vehicle makes, with time-headways τ being close enough to the true mean time-gaps (as estimated from real data). Time-gap is specified as the fraction of the space-gap between two vehicles to the speed of the following vehicle. The MAEs between the estimated trajectories and the real ones are also presented. The obtained MAEs are consistent to previous works on the parameter identification of commercial ACC systems [6], [15]. Finally, as shown in Table II, the ACC followers are neither $\mathcal{L}_2$ nor $\mathcal{L}_\infty$ strict string stable, except the last vehicle in the AstaZero platoon which is only $\mathcal{L}_\infty$ string stable. This is possible since $\mathcal{L}_2$ stability is stronger than the $\mathcal{L}_\infty$ stability, see (7). Also, the last follower in Ispra-Vicolungo was driving manual, and hence, there are no findings available.

TABLE II
AstaZero and Ispra-Vicolungo CTHP Parameter Learning Results.

Campaign	Vehicle i	α (1/s^2)	β (1/s)	τ (s)	Mean time-gap (s)	MAE space-gap (m)	MAE speed (m/s)	$\mathcal{L}_2$	$\mathcal{L}_\infty$	Make	Model
AstaZero	1	0.0618	0.1120	1.19	1.25	0.1320	0.0852	NO	NO	Audi	A6
	2	0.1000	0.1510	1.17	1.23	0.1102	0.0979	NO	NO	BMW	X5
	3	0.0755	0.2220	1.16	1.20	0.2998	0.1772	NO	NO	Mercedes	A-Class
	4	0.0480	0.3870	1.18	1.30	0.2806	0.1815	NO	YES	Tesla	Model 3
Ispra-Vicolungo	1	0.0767	0.1590	1.01	1.03	0.4140	0.2143	NO	NO	Ford	S-Max
	2	0.1930	0.3290	1.00	1.01	0.1873	0.1512	NO	NO	Peugeot	5008
	3	0.0712	0.1890	1.13	1.18	0.0892	0.0897	NO	NO	Kia	Niro
	4	-	-	-	1.52	-	-	-	-	Mini	Cooper

V. Conclusions

The parameter learning of commercially implemented ACC systems is challenging since their core functionality is not publicly available. This work unveiled that PINNs can be used as a surrogate approximator to capture the ACC vehicle longitudinal dynamics and efficiently infer the unknown parameters of the CTHP, often implemented in stock ACC systems of various makes. The findings of this paper demonstrate the ease in which PINNs perform in learning the unknown ACC design parameters. PINNs may retrieve successfully the true ACC parameters based on synthetic data, as well as to deliver estimates of the unknown design parameters of stock ACC systems based on empirical observations of space-gap and relative velocity. This is confirmed by the similar ACC parameter values found among the different ACC controllers in the three experimental campaigns. Applying the string stability criteria to the obtained results showed that ACC systems tend to be string unstable inside the platoon.

References

[1] SAE International, "Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles," SAE Int., Revised J3016_202104, 2021.

[2] P. A. Ioannou and C. C. Chien, "Autonomous intelligent cruise control," *IEEE Trans. Veh. Technol.*, vol. 42, no. 4, pp. 657–672, 1993.

[3] D. Swaroop and J. K. Hedrick, "Constant spacing strategies for platooning in automated highway systems," *J. Dyn. Syst. Meas. Control*, vol. 121, no. 3, pp. 462–470, 1999.

[4] D. Yanakiev and I. Kanellakopoulos, "Nonlinear spacing policies for automated heavy-duty vehicles," *IEEE Trans. Veh. Technol.*, vol. 47, no. 4, pp. 1365–1377, 1998.

[5] V. Milanés and S. Shladover, "Modeling cooperative and autonomous adaptive cruise control dynamic responses using experimental data," *Transp. Res. Part C Emerg. Technol*, vol. 48, pp. 285–300, 2014.

[6] G. Gunter, D. Gloudemans, R. E. Stern, S. McQuade, R. Bhadani, M. Bunting, M. L. Delle Monache, R. Lysecky, B. Seibold, J. Sprinkle, B. Piccoli, and D. B. Work, "Are commercially implemented adaptive cruise control systems string stable?" *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 11, pp. 6992–7003, 2021.

[7] V. Milanés, S. E. Shladover, J. Spring, C. Nowakowski, H. Kawazoe, and M. Nakamura, "Cooperative adaptive cruise control in real traffic situations," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 1, pp. 296–305, 2014.

[8] D. Swaroop and K. Hedrick, "String stability of interconnected systems," *IEEE Trans. Autom. Control*, vol. 41, no. 3, pp. 349–357, 1996.

[9] C.-Y. Liang and H. Peng, "Optimal adaptive cruise control with guaranteed string stability," *Veh. Syst. Dyn.*, vol. 32, no. 4-5, pp. 313–330, 1999.

[10] J. Eyre, D. Yanakiev, and I. Kanellopoulos, "A simplified framework for string stability analysis of automated vehicles," *Veh. Syst. Dyn.*, vol. 30, no. 5, pp. 375–405, 1998.

[11] B. Besselink and K. H. Johansson, "String stability and a delay-based spacing policy for vehicle platoons subject to disturbances," *IEEE Trans. Autom. Control*, vol. 62, no. 9, pp. 4376–4391, 2017.

[12] J. Ploeg, N. van de Wouw, and H. Nijmeijer, "$\mathcal{L}_p$ string stability of cascaded systems: Application to vehicle platooning," *IEEE Trans. Control Syst. Technol.*, vol. 22, no. 2, pp. 786–793, 2014.

[13] M. Makridis, K. Mattas, A. Anesiadou, and B. Ciuffo, "OpenACC: An open database of car-following experiments to study the properties of commercial acc systems," *Transp. Res. Part C Emerg. Technol*, vol. 125, p. 103047, 2021.

[14] G. Gunter, C. Janssen, W. Barbour, R. E. Stern, and D. B. Work, "Model-based string stability of adaptive cruise control systems using field data," *IEEE Trans. Intell. Veh.*, vol. 5, no. 1, pp. 90–99, 2020.

[15] Y. Wang, G. Gunter, M. Nice, M. L. D. Monache, and D. B. Work, "Online parameter estimation methods for adaptive cruise control systems," *IEEE Trans. Intell. Veh.*, vol. 6, no. 2, pp. 288–298, 2021.

[16] V. Punzo and F. Simonelli, "Analysis and comparison of microscopic traffic flow models with real traffic microscopic data," *Transp. Res. Rec.*, vol. 1934, no. 1, pp. 53–63, 2005.

[17] A. Kesting and M. Treiber, "Calibrating car-following models by using trajectory data: Methodological study," *Transp. Res. Rec.*, vol. 2088, no. 1, pp. 148–156, 2008.

[18] K. Ampountolas, "The unscented kalman filter for nonlinear parameter identification of adaptive cruise control systems," *IEEE Trans. Intell. Veh.*, 2023, accepted, DOI: 10.1109/TIV.2023.3272660.

[19] I. E. Lagaris, A. Likas, and D. I. Fotiadis, "Artificial neural networks for solving ordinary and partial differential equations," *IEEE Trans. Neural Netw.*, vol. 9, no. 5, pp. 987–1000, 1998.

[20] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational Physics*, vol. 378, pp. 686–707, 2019.

[21] L. Lu, X. Meng, Z. Mao, and G. Karniadakis, "Deepxde: A deep learning library for solving differential equations," *SIAM Rev.*, vol. 63, pp. 208–228, 2021.

[22] G. Karniadakis, Y. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," *Nat. Rev. Phys.*, vol. 3, pp. 422–440, 2021.

[23] R. E. Wilson and J. A. Ward, "Car-following models: fifty years of linear stability analysis – a mathematical perspective," *Transp. Plan. Technol.*, vol. 34, no. 1, pp. 3–18, 2011.

[24] J. Monteil, M. Bouroche, and D. J. Leith, "$\mathcal{L}_2$ and $\mathcal{L}_\infty$ stability analysis of heterogeneous traffic with application to parameter optimization for the control of automated vehicles," *IEEE Trans. Control Syst. Technol.*, vol. 27, no. 3, pp. 934–949, 2019.

[25] A. Baydin, B. Pearlmutter, A. Radul, and J. Siskind, "Automatic differentiation in machine learning: A survey," *J. Mach. Learn. Res.*, vol. 18, pp. 1–43, 2018.

[26] H. Robbins and S. Monro, "A stochastic approximation method," *Ann. Math. Statist.*, vol. 22, no. 3, pp. 400–407, 1951.

[27] B. Polyak and A. Juditsky, "Acceleration of stochastic approximation by averaging," *SIAM J. Control Optim.*, vol. 30, no. 4, pp. 838–855, 1992.

[28] D. Rumelhart, G. Hinton, and R. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.

[29] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[30] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. of the 3rd Int. Conf. on Learning Represent. (ICLR)*, 2014.

[31] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, "A limited memory algorithm for bound constrained optimization," *SIAM J. Sci. Comput.*, vol. 16, no. 5, pp. 1190–1208, 1995.