Online parameter estimation methods for adaptive cruise control systems

Yanbing Wang[1], George Gunter[1], Matthew Nice[2], Maria Laura Delle Monache[3] and Daniel B. Work[1]

***Abstract*—Modeling *Adaptive Cruise Control* (ACC) vehicles enables the understanding of the impact of these vehicles on traffic flow. In this work, two online methods are used to provide real time system identification of ACC enabled vehicles. The first technique is a *recursive least squares* (RLS) approach, while the second method solves a nonlinear joint state and parameter estimation problem via *particle filtering* (PF).**

We provide a parameter identifiability analysis for both methods to analytically show that the model parameters are not identifiable using equilibrium driving. The accuracy and computational runtime of the online methods are compared to a commonly used offline simulation-based optimization (i.e., batch optimization) approach. The methods are tested on synthetic data as well as on empirical data collected directly from a 2019 model year ACC vehicle using data from sensors that are part of the stock ACC system. The online methods are scalable and provide comparable accuracy to the batch method. RLS runs in real time and is two orders of magnitude faster than the batch method for modest sized (e.g., 15 min) datasets. The particle filter also runs in real-time, and is also suitable in streaming applications in which the datasets can grow arbitrarily large.

I. Introduction

A. Motivation and problem statement

The rising penetration rate of *Society of Automotive Engineers* (SAE) level one and level two automated vehicles on roadways around the world is creating new traffic flows that are a combination of human drivers and vehicle automation systems, yet the behavior of these systems is not well understood. For example, one common feature that is now available on many vehicles is *adaptive cruise control* (ACC), which enables the vehicle (instead of the human driver) to adjust velocity in response to the vehicle ahead.

The design of *string stable* ACC systems has been an important topic in the vehicle control community for decades [1]–[5]. Recently, as the vehicle systems have transitioned from research to practical deployments on commercial vehicles, the traffic modeling community is now in need of good models for how these vehicles behave in practice. Surprisingly, all commercial systems that have been tested [6]–[11] are shown to be string unstable and with varying performance characteristics. Being able to characterize the behavior of the ACC system in real time has implications for traffic management, where an emerging area of research [9], [12]–[15] aims to dampen phantom traffic jams caused by string unstable driving behavior [16].

These findings highlight the need to accurately model the implications of deploying ACC vehicles at scale, and motivate our desire for fast and accurate methods to estimate the parameters of the ACC vehicles. In our prior works investigating the string stability of commercial ACC systems [9], [17], ACC model parameters are estimated through an offline optimization procedure to best fit recorded data. While accurate and computationally efficient for small datasets, online methods are desirable for real-time applications and for computation on large (high frequency, long time duration) datasets. With the proliferation of in-vehicle (e.g., radar, lidar and GPS) sensing, it is now possible to develop fast and accurate online estimation routines which can quickly estimate the parameters of vehicles under adaptive cruise control.

This article considers the problem of estimating the parameters of adaptive cruise controlled vehicles using online algorithms that can sequentially estimate the parameters when new measurements become available. Two online methods are used based on *recursive least squares* (RLS) and *particle filtering* (PF), and both are shown to provide accurate estimates. As a proof of demonstration, we also implement the methods on data collected from a 2019 stock SUV with ACC, using only data from the vehicle's existing on-board sensors.

B. Related work

The problem of estimating parameters of automated vehicle models is closely related to the problem of calibrating models of human driving. Several works have looked at the car-following estimation problem for human drivers [18]–[20] using data from the well-known NGSIM dataset. Kesting and Treiber [21] proposed a methodology to estimate parameters for the Intelligent Driver Model by minimizing the error between simulated driving trajectories and measured ones, via the use of a genetic search algorithm in order to account for non-convexity in the search space. Punzo and Simonelli [22] estimated optimal parameters for several different car-following models on a four vehicle platoon equipped with GPS tracking units, and used a batch optimization method in which a gradient-based optimization technique was used starting from several different initial points, again in order to account for potential non-convexity. This method (along with ones in [7], [9]–[11], [17]) is in the same family of techniques used in this article as a benchmark to compare the newly presented methods. In general it should be noted that many of these techniques find parameters of non-linear models, and

[1]Department of Civil and Environmental Engineering and the Institute for Software Integrated Systems, Vanderbilt University, Nashville, TN 37240

[2]Department of Electrical Engineering and Computer Science and the Institute for Software Integrated Systems, Vanderbilt University, Nashville, TN 37240

[3]NeCS team at Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, GIPSA-Lab, 38000 Grenoble, France

as such need to employ optimization techniques that account for this difficulty.

In addition to offline batch calibration methods such as [9], [11], [17], [22], other works consider probabilistic methods that require only a single pass through recorded data to identify parameters. Van Hinsberge et al. [23] use Bayesian analysis to update prior probabilities of the parameters into posterior probabilities; Hoogendoorn & Hoogendoorn [24] use a generalized maximum likelihood estimation approach. Other methods for microscopic model calibration in the presence of automatic and connected vehicles are summarized in a recent study by Papamichail et al. [25].

An important but less investigated aspect of model calibration is *model parameter identifiability* analysis [26], [27], which characterizes the possibility of accurately recovering model parameters from experimental data. Among them, practical identifiability of car-following models is considered [19], [28]. For example, Punzo et al. [19] use variance-based sensitivity analysis to identify insensitive parameters, which are less likely to converge to the true values. Monteil & Bouroche [28] introduce procedures for robust parameter estimation including global sensitivity analysis, maximum likelihood estimation and interval estimation. Sensitivity-based identifiability analysis relies on sampling parameters and calculating the variance of all simulated trajectories, which limits its scope to offline calibration.

Online parameter estimation, which has the potential to achieve scalability and facilitate real-time applications, would also benefit from model identifiability analysis for rigorous experimental design. Convergence of car-following model parameters in estimation is significantly influenced by the sensitivity of the parameters with respect to the driving behavior captured by the dataset [29]. Specifically for online filtering methods, proper models of the system and measurement noises are important to the performance of the method [30].

C. Contributions and outline

The main contribution of this article is the use of online parameter estimation algorithms to solve the problem of recovering adaptive cruise control parameters, and a corresponding parameter identifiability analysis of the methods. We use two online estimation algorithms, RLS and PF, that are fast and scalable for real time system identification of ACC dynamics. We provide an analysis of the parameter identifiability of both methods to understand if and when the estimates can be theoretically recovered. This analysis is important but missing for the batch methods previously applied to estimate parameters of ACC systems. Finally, we provide numerical and real world examples illustrating the performance of the methods in controlled numerical simulations, as well as on a modern ACC vehicle. The real vehicle experiment uses only the onboard CAN bus data, which provides a novel experimental approach to understand the behavior of ACC vehicles.

The remainder of the paper is organized as follows. Section II reviews the ACC model and outlines the batch optimization method as a benchmark for parameter estimation. Section III-A introduces the online RLS and PF based estimators, and provides an analysis of the estimators at equilibrium driving conditions. Section IV demonstrates the performance of the estimation routines on synthetically generated (simulated) data, in order to assess the performance of the methods under controlled settings. Section V addresses the practical performance of the method using data from a real vehicle platform. It presents both the experimental protocol for collecting ACC radar and speed measurements directly from a 2019 vehicle's CAN bus, and the results of parameter estimation on that data. Finally, Section VI explores future research directions.

II. Preliminaries

In this section, we briefly review a common model assumed for ACC vehicle dynamics, and then review a standard simulation-based optimization method to estimate the model parameters used in this work as a benchmark.

A. Model description and string stability

With the increasing interest in how vehicles with automated driving systems [2], [3] will affect traffic flow patterns, several works have looked at modeling ACC vehicles using car-following models [9], [11], [17]. A common variation of these models is the *constant time headway relative velocity* (CTH-RV) model:

$$\dot{v}(t) = f(\boldsymbol{\theta}, s(t), v(t), \Delta v(t)) = \alpha(s(t) - \tau v(t)) + \beta(\Delta v(t)), \tag{1}$$

where s, v, and Δv are the space gap, velocity, and velocity difference between the ACC vehicle and a leading vehicle. The vector of model parameters $\boldsymbol{\theta} = [\alpha, \beta, \tau]^T$ control the gain on the constant time headway term and the relative velocity term respectively, while the parameter τ is the time gap at equilibrium.

We note that models considering constant time headway and relative velocity terms are regularly used both to design string stable adaptive cruise control systems, as well as to model the behavior of vehicles under ACC control [4], [11], [31]–[34]. Compared to other modeling choices, it is observed that CTH-RV model performs about as well in terms of data fitting real ACC systems compared to more complex nonlinear models [35]. However, the model is a simplification of the proprietary control logic and complex vehicle dynamics of real ACC vehicles, and the quality of fit can drop for some specialized vehicles (e.g., hybrid vehicles) [9]. To avoid the need to know the proprietary control logic, we adopt a similar strategy to what is done for human drivers, namely model the full system as an ordinary differential equation. For the remainder of this work, we adopt the CTH-RV as the assumed model of the ACC equipped vehicle.

Given (1), it is easy to check the string stability [36] of the ego (i.e., follower) vehicle by evaluating partial derivatives of the model with respect to s, v, and Δv. Following the analysis of [37], if

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

then the model is said to be $\mathcal{L}_2$ *strict string stable*, which is consistent with the string stability condition provided in [36]. Moreover, if

$$(\alpha\tau + \beta)^2 - 4\alpha \geq 0, \tag{3}$$

then the model is said to be $\mathcal{L}_\infty$ *strict string stable* [37]. All studies that have collected empirical data on commercial ACC systems have found them to be string unstable [9]–[11], [17] (in the $\mathcal{L}_2$ sense). In Section V, we illustrate that the new online methods introduced in this work find that a stock 2019 SUV is neither $\mathcal{L}_2$ nor $\mathcal{L}_\infty$ strict string stable.

Note that although string stability manifests along a platoon of vehicles, it is a property of the individual vehicle car following behavior [36], [37]. Thus identifying the car-following model parameters of the follower vehicle is sufficient to analytically prove the string stability of the follower vehicle. The interpretation of a string stable vehicle is that a homogeneous platoon consisting of these vehicles will dissipate disturbances rather than amplify them as the perturbation propagates in through platoon. Since no priori knowledge of the ACC system string stability is assumed in the experiments, we do not use string stability as constraints during parameter estimation.

B. Offline batch optimization

Here a well known batch technique for car-following parameter estimation [17], [22] is reviewed to estimate the parameters of the *ordinary differential equation* (ODE) model (1). The parameter estimation problem is posed as an optimization problem in which the ACC model appears as a constraint. It can be directly solved as a simulation-based optimization problem using standard descent-based optimization routines.

The parameter values are optimized to minimize the *root mean squared error* (RMSE) between simulated space gap data and recorded space gap data. The RMSE space gap error is used here because it was found to perform well in previous works [21], [22]. The general form of optimization problem is written as:

$$\begin{aligned} \text{minimize}: \quad & \sqrt{\tfrac{1}{T}\int_0^T (s^{\mathrm{m}}(t) - s(t))^2 dt} \\ \text{subject to}: \quad & \dot{s}(t) = u(t) - v(t) = \Delta v \\ & \dot{v}(t) = f(\boldsymbol{\theta}, s, v, \Delta v), \end{aligned} \tag{4}$$

with possible additional constraints on the initial conditions, and bounds on the parameters. In (4), $f(\boldsymbol{\theta}, s, v, \Delta v)$ corresponds to the car-following model in (1). The term $u(t)$ is the lead vehicle velocity as a function of time and is assumed to be available from measured data. Similarly, $s^{\mathrm{m}}(t)$ denotes the measured space gap, which is compared to the space gap predicted by the model in the objective function. The total time of the dataset and simulation is T.

It is important to note that the problem is nonlinear in the decision variables (the state and model parameters), and depending on the form of the car-following model, it may also be non-convex. To combat this potential problem the optimization problem can be solved many times, with each run starting from randomly selected different initial candidate parameter values, as in [22].

III. Online parameter estimation techniques

We next introduce two online methods to estimate the adaptive cruise control model parameters using velocity, space gap, and relative velocity data.

A. Recursive least-squares formulation

First we derive a RLS estimator. Unlike (4), the least-squares method proposed here does not require multiple starting points or repeatedly solving an ODE within each optimization run, substantially reducing the runtime. We briefly derive the least-squares formulation for the ACC car-following model (1).

First we rewrite the continuous time ODE (1) in discrete-time using a forward Euler step scheme:

$$v_{k+1} = v_k + \alpha(s_k - \tau v_k)\Delta T + \beta(u_k - v_k)\Delta T, \tag{5}$$

where v_k, s_k and u_k denote the velocity of the follower vehicle, the space gap, and the velocity of the leading vehicle at timestep k, respectively. The term ΔT is the timestep size, which is selected to correspond to the frequency at which the velocity, space gap, and relative velocity data is measured (e.g., on the order of 1/10 of a second for some sensor platforms including the experiments presented later in this work). The dynamics can be rewritten as:

$$v_{k+1} = \gamma_1 v_k + \gamma_2 s_k + \gamma_3 u_k, \tag{6}$$

with $\gamma_1 := (1 - (\alpha\tau + \beta)\Delta T)$, $\gamma_2 := (\alpha\Delta T)$ and $\gamma_3 := (\beta\Delta T)$. Note that instead of directly estimating the parameters α, β, τ, we can instead estimate $\gamma_1, \gamma_2, \gamma_3$. Except in the degenerate case when $\gamma_2 = 0$, we can always uniquely determine the values of α, β, τ given a set of values for $\gamma_1, \gamma_2, \gamma_3$.

We now demonstrate that one can recover $\boldsymbol{\gamma} := [\gamma_1, \gamma_2, \gamma_3]^T$ from an experimental dataset containing (v_k, s_k, u_k) for all $k \in \{1, ..., K\}$, via least-squares. We expand (6) in time by stacking the uniformly sampled measurements to obtain:

$$\begin{bmatrix} v_2 \\ v_3 \\ \vdots \\ v_K \end{bmatrix} = \begin{bmatrix} v_1 & s_1 & u_1 \\ v_2 & s_2 & u_2 \\ \vdots & \vdots & \vdots \\ v_{K-1} & s_{K-1} & u_{K-1} \end{bmatrix} \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_3 \end{bmatrix}, \tag{7}$$

or

$$Y = X\boldsymbol{\gamma}. \tag{8}$$

The term Y contains the values of v_k from timestep 2 to K. The term X contains measurements of v_k, s_k, and u_k from timestep 1 to $K-1$ in column-wise order.

Given the data matrices Y and X, $\boldsymbol{\gamma}$ has a unique solution if and only if $\text{rank}(X) = \text{rank}([X \quad Y]) = 3$. Note that this condition is not satisfied at equilibrium, where $v_i = v_j = u_k$ for all timesteps i, j and k. In other words, using only data from equilibrium driving, it is not possible to recover the model parameters. In non-equilibrium driving and when sensor noise is present, it is easy to generate an over determined system of equations, motivating the search for least squares solution to (7).

To convert the least squares problem into an online method, a recursive implementation is desired. The least squares solution to (7) has an exact recursive implementation by considering the k^{th} row of measurements $\{Y_k, X_k\}$ one row at a time. The least squares estimate of the parameter vector at time k using all data collected from timestep 1 through k, denoted $\hat{\boldsymbol{\gamma}}_k$, can be sequentially updated by:

$$\hat{\boldsymbol{\gamma}}_k = \hat{\boldsymbol{\gamma}}_{k-1} + P_k X_k (Y_k - \hat{\boldsymbol{\gamma}}_{k-1} X_k), \tag{9}$$

where $P_k^{-1} = \sum_{i=1}^{k} X_i X_i^T$ is the cumulative outer product of X_k. Solving (9) requires an initial estimate of the parameters $\boldsymbol{\gamma}_0 \sim \mathcal{N}(\hat{\boldsymbol{\gamma}}_0, P_0)$, which are specified in the numerical experiment in Section IV.

B. Online joint state and parameter estimation formulation

The parameter estimation problem can also be framed as an online joint parameter and state estimation problem, in which model and measurement noises are explicitly considered. Such methods, if fast enough, may also be used for real-time processing of data in order to estimate the model parameters during data collection.

1) Problem formulation: To jointly estimate the state and model parameters, we consider an augmented state formulation in which the model parameters are added to the state vector. The model of the evolution of the augmented state is completed by assuming the parameters are constant in time.

We proceed as follows. First, $\boldsymbol{\theta} = [\alpha, \beta, \tau]^T$, is concatenated to the physical system state $\boldsymbol{x}_k \in \mathbb{R}^2 = [s_k, v_k]^T$ to form an augmented state $\boldsymbol{x}_k^a \in \mathbb{R}^5$. This is written as:

$$\boldsymbol{x}_k^a = \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{\theta} \end{bmatrix}_k = \begin{bmatrix} s & v & \alpha & \beta & \tau \end{bmatrix}_k^T. \tag{10}$$

The discrete time dynamics of the augmented system using the same discretization approach as (5) can be written as:

$$\boldsymbol{x}_k^a = \mathcal{F}_d(\boldsymbol{x}_{k-1}^a, u_{k-1}), \tag{11}$$

where $\mathcal{F}_d$ refers the system dynamics in the augmented state. The nonlinearities in the dynamics appear due to the product of augmented state variables representing the ACC model parameters and the physical states.

The augmented state dynamics (11) are written as:

$$\mathcal{F}_d(\boldsymbol{x}_{k-1}^a, u_{k-1}) = \begin{bmatrix} s_{k-1} + \Delta T(u_{k-1} - v_{k-1}) \\ v_{k-1} + \Delta T\left[\alpha_{k-1}(s_{k-1} - \tau_{k-1} v_{k-1}) + \beta_{k-1}(u_{k-1} - v_{k-1})\right] \\ \alpha_{k-1} \\ \beta_{k-1} \\ \tau_{k-1} \end{bmatrix} \tag{12}$$

Additionally, a measurement equation is written to reflect the condition that the physical states (s_k, v_k) may be directly measured, but we do not measure the parameters $(\alpha_k, \beta_k, \tau_k)$:

$$\boldsymbol{y}_k = C\boldsymbol{x}_k^a = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix} \boldsymbol{x}_k^a. \tag{13}$$

In (13), $y_k \in \mathbb{R}^2$ are the measurements, and C is the measurement matrix.

2) Particle filter: Because the augmented state (of the nonlinear augmented system) is to be estimated, a nonlinear estimator must be considered. Here, we outline an approach to estimate the augmented state using a PF.

The filter takes in the discrete-time system that considers model and measurement noises. The state-space form is written as:

$$\begin{aligned} \boldsymbol{x}_k^a &= \mathcal{F}_d(\boldsymbol{x}_{k-1}^a, u_{k-1}) + \boldsymbol{w}_k \\ \boldsymbol{y}_k &= C\boldsymbol{x}_k^a + \boldsymbol{\nu}_k, \end{aligned} \tag{14}$$

where $\boldsymbol{w}_k \sim (0, Q) \in \mathbb{R}^5$ and $\boldsymbol{\nu}_k \sim (0, R) \in \mathbb{R}^2$ are independent white noise processes for the model and the measurement equations, respectively, at time k. $Q \in \mathbb{R}^{5\times5}$ and $R \in \mathbb{R}^{2\times2}$ are the known process and measurement error covariance matrices.

Recall that the Bayesian state estimation method sequentially approximates the posterior *probability density function* (PDF) of the augmented state at step k given past observations, i.e., $p(\boldsymbol{x}^a|\boldsymbol{y}_{1:k})$. The Bayesian state estimation method can be summarized into two parts:

1) **State propagation**: obtain the prior distribution at k:

$$p(\boldsymbol{x}_k^a|\boldsymbol{y}_{1:k-1}) = \int p(\boldsymbol{x}_k^a|\boldsymbol{x}_{1:k-1}^a) p(\boldsymbol{x}_{k-1}^a|\boldsymbol{y}_{1:k-1}) d\boldsymbol{x}_{k-1}^a \tag{15}$$

2) **State update**: obtain the posterior distribution at k:

$$p(\boldsymbol{x}_k^a|\boldsymbol{y}_{1:k}) = \frac{p(\boldsymbol{y}_k|\boldsymbol{x}_k^a) p(\boldsymbol{x}_k^a|\boldsymbol{y}_{1:k-1})}{p(\boldsymbol{y}_k|\boldsymbol{y}_{1:k-1})}. \tag{16}$$

The particle filter [38], [39], among other filtering techniques, is deployed to approximate the prior and the posterior distributions from equations (15) and (16) because of its flexibility in noise distribution and its relaxed assumption about the linearity of the (augmented) dynamics of the system. PF uses weighted particles (samples) to approximate the conditional state distribution given all measurements up to the current timestep using a sequential estimation approach. Therefore, the output is a probability distribution for each parameter at each time step.

Algorithm 1 Particle filter

Initialize $(k = 0)$

Draw i particles $\{\boldsymbol{x}_0^{a,(i)}\}_{i=1:N_p}$ from an initial distribution $p(\boldsymbol{x}_0^a)$. Assign equal weights $\omega_0^{(i)} = 1/N_p$, where $i = 1, \ldots, N_p$, and N_p is the number of particles.

for $k = 1 \ldots T$ **do**

  State propagation:

  $\boldsymbol{x}_k^{a,(i)} = \mathcal{F}_d(\boldsymbol{x}_{k-1}^{a,(i)}, u_{k-1}) + \boldsymbol{w}_k^{(i)}$ for all i.

  State update:

  Assign weight: $\omega_k^{(i)} := \omega_{k-1}^{(i)} p(\boldsymbol{y}_k|(\boldsymbol{x}_k^{a,(i)})$ for all i.

  Normalize weight: $\omega_k^{(i)} := \omega_k^{(i)} / \sum_{i=1}^{N_p} \omega_k^{(i)}$ for all i.

  Resample:

  Draw $\boldsymbol{x}_k^{a,(i)}$ with probability $\omega_k^{(i)}$ for all i.

end

A summary of the PF is written in Algorithm 1. During implementation, it is important to monitor the effective particle size [40] to ensure a valid estimation result. For more details on the PF implementation, readers are referred to standard references such as [41].

C. Observability analysis

In this section we provide insights on the ability to estimate the ACC model parameters via an observability analysis. An observable system indicates theoretically that its initial state can be inferred from observing the outputs. For parameter estimation in the joint state-parameter form, recovering the initial state indicates identifying the non-changing parameters. In this work we consider a special case of parameter observability if the parameters are considered as constant state variables. This assumption allows us to equate the notion of observable augmented state to uniquely identifiable parameters given measurements. Given that the augmented state dynamics (14) are nonlinear, observability must be assessed on a linearized version of the model. This can only be done at fixed values of the augmented state, and is explained as below.

First we write the linearized state-space model of the nonlinear discrete-time system (11):

$$\begin{aligned} \boldsymbol{x}_k^a &= A_{k-1}\boldsymbol{x}_{k-1}^a + B_{k-1}u_{k-1} \\ \boldsymbol{y}_k &= C\boldsymbol{x}_k^a, \end{aligned} \tag{17}$$

where A_k is the Jacobian of $\mathcal{F}_d$ defined in (11) with respect to the augmented state variables at time k, B_k is the Jacobian with respect to the control inputs at time k, and C is the time-invariant measurement matrix as defined above in (13). Further, A_k can be written as

$$\begin{aligned} A_k &= \left.\frac{\partial \mathcal{F}_d}{\partial \boldsymbol{x}^a}\right|_{\boldsymbol{x}_k^{a*},u_k^*} \\ &= \begin{bmatrix} 1 & -\Delta T & 0 \\ \alpha\Delta T & 1-\alpha\tau\Delta T-\beta\Delta T & (s-\tau v)\Delta T \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \\ &\quad \begin{bmatrix} 0 & 0 \\ (u-v)\Delta T & -\alpha v\Delta T \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}_{\boldsymbol{x}_k^{a*},u_k^*}, \end{aligned} \tag{18}$$

where $\boldsymbol{x}_k^{a*}, u_k^*$ are the state and input points about which the system is linearized and B_k is defined similarly.

We choose to analyze the system observability by computing the above partial derivatives evaluated at an equilibrium point. The condition for equilibrium reduces to zero acceleration and space gap change, i.e.:

$$\begin{aligned} u_k - v_k &= 0 \\ s_k - \tau_k v_k &= 0\ . \end{aligned} \tag{19}$$

In addition, the system (11) reduces to a linear time invariant system, and A_k from (18) at equilibrium simplifies to:

$$A = \begin{bmatrix} 1 & -\Delta T & 0 & 0 & 0 \\ \alpha\Delta T & 1-\alpha\tau\Delta T-\beta\Delta T & 0 & 0 & -\alpha v\Delta T \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

From here the observability matrix can be calculated using A and C as follows:

$$\mathcal{O} = \begin{bmatrix} C,\ CA,\ CA^2,\ CA^3,\ CA^4 \end{bmatrix}^T,$$

where $\mathcal{O}$ is the observability matrix, the rank of which is used to assess observability.

When analyzed at any equilibrium point, the resulting observability matrix has rank($\mathcal{O}$) $= 3 \neq 5$, corresponding to a non-observable system. The corresponding null space of the observability matrix is:

$$\text{null}(\mathcal{O}) = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}^T, \tag{20}$$

indicating two unidentifiable parameters, α and β at the equilibrium points. This analysis shows that for the augmented-state estimation problem there is no guarantee for exact recovery of α and β at equilibrium when using filtering techniques. The observability matrix derivation in this section is based on the linearized system around an equilibrium point. Therefore, it does not provide insight on parameter estimation for non-equilibrium trajectories, which we will instead explore numerically in the computational experiments in Section IV using the PF described next.

IV. Estimation On Synthetic Data

In this section, each parameter estimation routine described above is run on synthetically generated data. This is done to understand the potential to recover the true model parameters under controlled settings. We show that all methods produce good estimates under non-equilibrium driving but have limited ability to recover true parameters under equilibrium driving conditions, consistent with the discussion from Section III.

A. Setup of synthetic experiments

The general setup is as follows. First, synthetic data is created by selecting a set of model parameters and a predefined lead driver velocity profile. A time-series of velocity and space gap data is then created via a forward simulation of (5) under the selected parameters and input signal. The simulated data is then fed into each estimation method, with each returning a set of estimated parameter values. The accuracy of the recovered parameters and the resulting state error of the system trajectory under the recovered parameters is then compared. The runtime for each method is also reported.

1) Equilibrium driving: In order to create a set of synthetic measurement data for driving under equilibrium, we begin by setting the true parameters as: $\boldsymbol{\theta}_{true} = [0.08, 0.12, 1.5]^T$. These values are representative of parameter values that have been reported for commercial ACC systems [9]. Additionally, in order to generate a synthetic dataset, the velocity profile of a lead vehicle is needed, along with an initial space gap, and the initial velocity of the following vehicle.

We generate equilibrium data by setting the velocity of the lead vehicle at a constant $u_k = v_{lead} = 24$ m/s while the initial space gap and the initial velocity of the follower satisfy $s_0 = \tau_{true} v_0$. A total of 900 seconds of velocity and space gap data is generated at a measurement frequency of 10

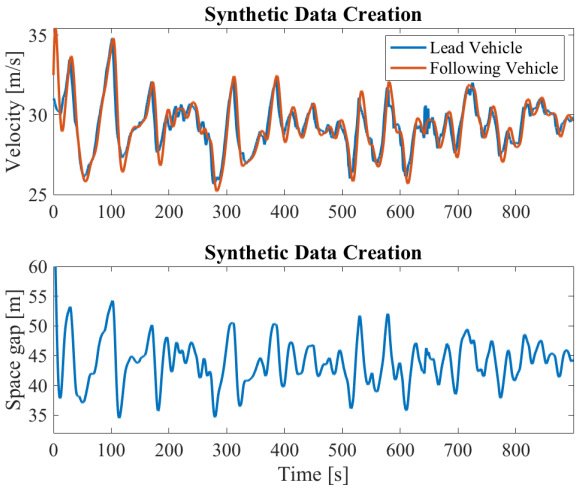

Fig. 1: Synthetic space gap and following vehicle data, generated from an empirical lead vehicle profile.

Hz. Because the data is generated such that the equilibrium condition (19) is satisfied, the true parameters α and β do not influence the space gap or velocity of the follower vehicle.

2) Non-equilibrium driving: We also consider a non-equilibrium driving scenario where the lead vehicle velocity (the system input) is empirically generated from a human driven vehicle in real highway driving. Using the empirical input, the ACC velocity and space gap data are still generated via forward simulation of the ACC model under the known true parameters. About 900 seconds of lead vehicle data is collected in which the driver of the vehicle follows the traffic rules but has variations in speed due interactions with other vehicles on the roadway. The simulation is initialized at a follower velocity and space gap of 32.5 m/s and 37.8 m, and the data is again generated at 10 Hz. Figure 1 displays the lead and follower velocity profiles, and the space gap data.

The true parameters used to generate the synthetic dataset are again $\boldsymbol{\theta}_{true} = [0.08, 0.12, 1.5]^T$, which is neither $\mathcal{L}_2$ (2) nor $\mathcal{L}_\infty$ (3) strict string stable. This means the ACC may amplify lead vehicle disturbances. As can be seen in the simulation data (Figure 1), in several occasions the follower vehicle slows down more than the leader.

B. Parameter estimation results on synthetic data

Using the synthetic data created above, we now turn to the results of each parameter estimation routine that attempt to recover the true parameters using only the measurement data. We use the *mean absolute error* (MAE) in space gap and velocity to compare the accuracy of each estimation method. Both RLS and the PF require several algorithm parameters to be set, which are summarized in Table I. For the batch method, we sample 100 starting points for the parameters from uniform distributions described in Table I. For RLS, we set the initial coefficient vector $\boldsymbol{\gamma}_0$ and its corresponding covariance matrix P_0. For the PF, we set the number of particles used in the estimator N_p, the initial distribution of the augmented state vector (assumed to follow a normal distribution with mean

Method	Parameters	Values
Batch optimization	$\boldsymbol{\theta}_0$	$[\mathcal{U}(0,1)\ \mathcal{U}(0,1)\ \mathcal{U}(1,3)]^T$
	# initial points	100
Least squares	$\boldsymbol{\gamma}_0$	$[0.976, 0.01, 0.01]^T$
	P_0	$0.1\boldsymbol{I}_3^{\ddagger}$
Particle filter	N_p	500 particles
	$\boldsymbol{\mu}_{x_0^a}$	$[37.8\ 32.5\ 0.1\ 0.1\ 1.4]^T$
	Q_0	$\text{diag}[0.5\ 0.5\ 0.2\ 0.2\ 0.3]^2$
	Q	$\text{diag}[0.2\ 0.1\ 0.01\ 0.01\ 0.01]^2$
	R	$\text{diag}[0.2\ 0.1]^2$

TABLE I: Parameters and initialization for all estimation routines. $^{\ddagger}\boldsymbol{I}_3$ is the identity matrix with size 3×3.

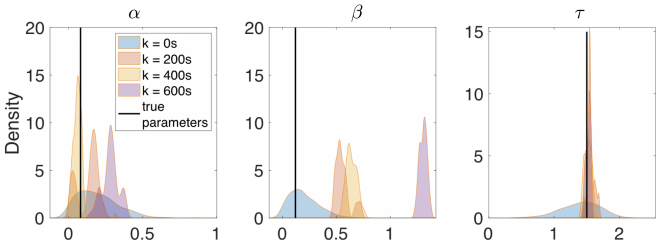

Fig. 2: Posterior parameter PDFs for equilibrium driving dataset from PF estimates. The plot shows that parameter α and β are not identified correctly, i.e., the distributions drift away from the true values (black vertical lines) over time. Only the distribution of τ converges to the true value.

$\boldsymbol{\mu}_{x_0^a}$ and covariance Q_0), and the model and measurement covariance matrices Q and R.

1) Equilibrium driving results: The performance of both online parameter estimators and the batch estimator are summarized in Table II. The summary includes the estimated parameters, the corresponding MAE for space gap and velocity, and the run time.

Compared to the true parameters $\boldsymbol{\theta}_{true} = [0.08\ 0.12\ 1.5]^T$, all methods estimate the true τ accurately, but have larger errors on α and β. In the least squares method, this is due to the fact that the matrix X from (7) has rank 1 (i.e., the columns in X (8) are linearly dependent, since $v_i = v_j = u_i = u_j$ for $i, j \in \{1, ..., K\}$ at equilibrium), and consequently $\boldsymbol{\gamma}$ does not have a unique solution. In the PF, the lack of observability of the system (17) leads to the the non-convergence of the PDFs in the PF (see Figure 2). Only τ converges to the true value, while the distributions of α and β drift away from the true values over time. Unlike the RLS estimator and the PF, the batch method does not benefit from additional information about the true parameters via the prior given at time 0. As a consequence, the errors on α and β are largest for the batch method.

The experiments illustrate that for all methods, the parameters are not identified at equilibrium. This is consistent with the lack of persistent excitation in the input signal [42].

Even though the true α and β are not recovered correctly, all methods produce models that have negligible MAE for speed and space gap. This is again due to the fact that α and β do not influence the trajectory of the ACC vehicle at equilibrium.

Finally, we compare the runtime of each method. The

Criteria	Batch optimization	RLS	PF
Estimated parameter values	$\alpha = 8.34$ $\beta = 7.30$ $\tau = 1.50$	$\alpha = 0.0965$ $\beta = 0.0976$ $\tau = 1.50$	$\alpha = 0.065$ $\beta = 0.604$ $\tau = 1.50$
Algorithm	Offline	Online	Online
Running time (s)	12.44	0.06	8.45
MAE space gap (m)	0.00	0.00	0.14
MAE velocity (m/s)	0.00	0.00	0.00

TABLE II: Performance on synthetic equilibrium data. True parameters are: $\alpha_{true} = 0.08, \beta_{true} = 0.12, \tau_{true} = 1.5$.

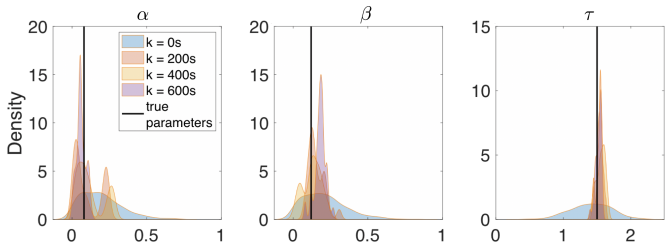

Fig. 3: Posterior parameter PDFs for non-equilibrium driving dataset from PF estimates.

recursive least-squares approach recovers the parameters in 0.06 seconds, which is the fastest runtime of the three methods by more than 2 orders of magnitude. The PF runs in 8 seconds, which is a factor of 100 faster than real time (recall the dataset is 900 seconds long). Although the total time to execute the batch method on a 900-second data is only 12 seconds, this is an offline method and cannot be run until all the data has been collected. This is in contrast to the online methods, which produce new estimates every time a new measurement is available. All the experiments are performed on the same MacBook Pro with 2.7 GHz CPU, such that the running time is comparable.

2) Non-equilibrium driving results: We now turn to the performance of the methods on the synthetic dataset generated from the ACC model when fed empirically collected non-equilibrium lead vehicle driving data. The results of each method are summarized in Table III.

Both the batch method and the RLS estimator recover the true model parameters used to generate the data. The PF performs worse but the estimates improve over time (Figure 3). When simulating with the mean values of the PF parameter estimates, the calibrated model has an MAE of 0.32 m/s in velocity and 2.54 m in the space gap. Given that the PF assumes both a model and measurement noise, it is not surprising that it does not perfectly recover the true parameters (since the model and measurement in fact have zero error in this synthetic example). We note the runtime for the two online methods again outperforms the batch method, and are significantly faster than real time.

Table III also shows the string stability estimate based on the recovered parameters using each method. Because the value of the parameters are used to determine the model string stability, it is important to know if errors in the estimates are large enough to change the string stability estimate. In this experiment, we find that all models under the estimated parameters are string unstable, as is the case with the true model parameters.

Additional numerical experiments can be found in [43], which explores the RLS performance in the presence of measurement noise, looking both at the real noise expected from the stock sensors as well as the sensitivity of the estimator to a range of noises. We find that the method can tolerate the noise levels present on the commercial vehicle platforms.

Criteria	Batch optimization	RLS	PF
Estimated parameter values	$\alpha = 0.08$ $\beta = 0.12$ $\tau = 1.5$	$\alpha = 0.08$ $\beta = 0.12$ $\tau = 1.5$	$\alpha = 0.04$ $\beta = 0.21$ $\tau = 1.41$
Algorithm	Offline	Online	Online
Running time (s)	11.27	0.06	8.43
MAE space gap (m)	0.00	0.00	2.54
MAE velocity (m/s)	0.00	0.00	0.32
$\mathcal{L}_2$ strict string stable	No	No	No
$\mathcal{L}_\infty$ strict string stable	No	No	No

TABLE III: Performance on synthetic nonequilibrium data: True parameters are: $\alpha_{true} = 0.08, \beta_{true} = 0.12, \tau_{true} = 1.5$. The reported parameter values are the maximum a posteriori (MAP) estimates of the last timestep (at 900 s), which can be slightly different from the estimates of the earlier timesteps.

V. Case Study on a 2019 ACC Equipped Vehicle

We now present a case study in which all methods are used to estimate the model parameters using data collected from a 2019 ACC equipped vehicle.

A. Experimental details

We first describe the velocity and space gap data collection using measurements from the stock radar system of an ACC-equipped vehicle. We compare the measurements collected from the ACC vehicle CAN bus to measurements collected using those from the GPS devices mounted on the ACC vehicle and its leader during the experimental data collection.

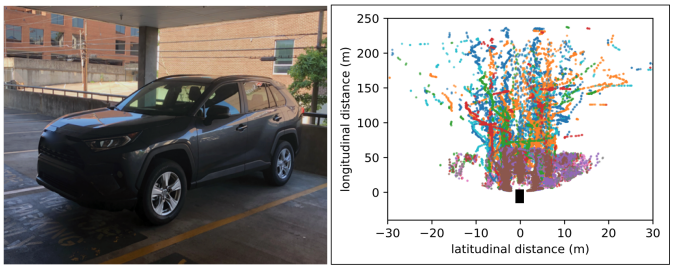

Fig. 4: *Left*: The 2019 ACC equipped stock SUV used in the experiment. *Right*: relative position data collected from the CAN bus of the vehicle within a duration of 15 min drive. Each point corresponds to the latitudinal and longitudinal distance to an object detected by the stock radar sensor and reported on the CAN bus. Colors correspond to distinct objects.

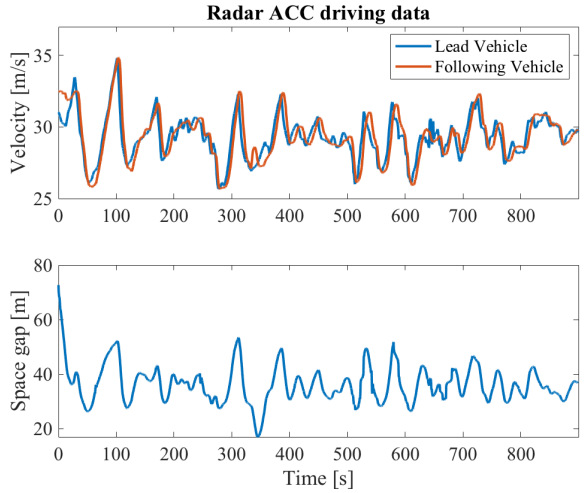

Fig. 5: CAN bus measurement of an ACC-enabled vehicle following a human-driven vehicle.

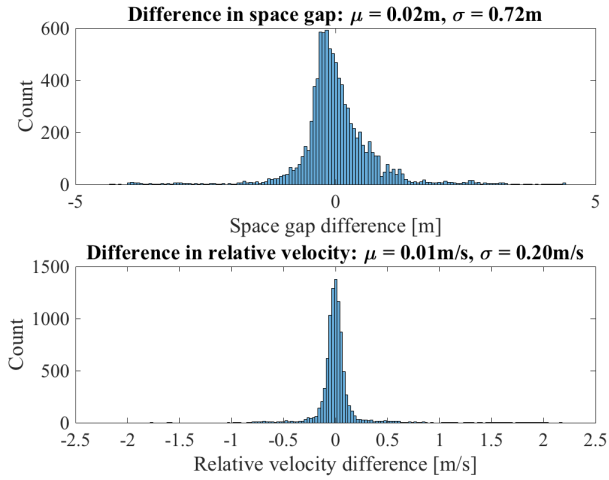

Fig. 6: Histogram of the difference between GPS measurements and CAN bus measurements for space gap and relative velocity measurements.

To set up the experiment, we create a two-vehicle system in which an ACC-equipped vehicle follows an instrumented lead vehicle driven by a human in real freeway driving conditions. The vehicle used in this experiment is a commercially available 2019 SUV with a full velocity range adaptive cruise control system (Fig. 4, left). A total of 15 minutes (900 seconds) of data are recorded at 10 Hz in which the ACC vehicle follows the lead vehicle through traffic on a freeway in Nashville, TN.

The driver of the lead vehicle is instructed to drive as they would normally in traffic, while the ACC equipped vehicle follows with ACC engaged. The entire experiment is conducted without any ACC de-activations or overrides, and no cars cut in between the leader and the ACC follower.

Velocity, lead vehicle velocity, and space gap data is collected by recording measurements from the CAN bus on the vehicle. The radar unit reports the latitudinal and longitudinal distance to objects in front of the vehicle (Figure 4, right), from which the space gap between the two vehicles can be computed. It also reports the relative velocity of objects in the field of view of the sensor. Since the ACC vehicle velocity is also published to the CAN bus, the lead vehicle velocity can be determined from the radar data. With straightforward processing of the radar data, it is possible to convert into velocity data of the leader and follower, and space gap, as shown in Figure 5.

B. CAN bus velocity and space gap data validation

In order to assess the accuracy of the stock vehicle radar unit, both vehicles are additionally equipped with sub-meter accurate GPS units which track global position and velocity. The devices are the same units used for primary data collection in our previous work [9], [17], [44]. The time-series of space gap and velocity are recorded from both GPS devices and the radar unit on the ACC vehicle and compared.

A histogram of the differences between the two measurement techniques is displayed in Figure 6. The distribution of differences between radar space gap measurements and GPS space gap measurements is approximately zero-centered, as are the relative velocity differences. The sensors do not appear to be biased. The standard deviations of the differences between the measurement devices is 0.72 m and 0.20 m/s for the space gap and relative velocity respectively, suggesting the sensor noises are also low.

As a further check, we briefly note that all estimation methods described next were run on data collected from the GPS devices as well as on the data from the radar unit as logged on the CAN bus. In all cases the estimated parameters are similar across the two sensor platforms. We conclude that the on-board radar measurements reported on the CAN bus are comparable to the GPS devices.

The fact that the space gap, velocity, and relative velocity data can be collected directly from the CAN bus significantly simplifies experimental data collection. Compared to our earlier work that required instrumenting two vehicles, the approach here can be applied using only a single vehicle (the ACC equipped vehicle). Eventually this may allow improved data collection from ACC vehicles in increasingly realistic settings, such as under cut-ins and lane changing.

C. Parameter estimation results on a 2019 ACC vehicle

With the CAN bus data validated, we now turn to the parameter estimation problem applied to the vehicle. The data contains non-equilibrium driving data, which is used to estimate the parameters using each method. We follow the same setup as the synthetic data experiments, with the notable exception that the true parameters of the ACC vehicle are unknown. The MAE between the measured space gap and the space gap under the estimated parameters is reported. Similarly, the MAE of the velocity is used to assess the quality of the estimated parameters. The string stability of the calibrated model under the parameters estimated by each method is also determined.

Criteria	Batch optimization	RLS	PF
Estimated parameter values	$\alpha = 0.0227$ $\beta = 0.194$ $\tau = 1.227$	$\alpha = 0.0174$ $\beta = 0.164$ $\tau = 1.127$	$\alpha = 0.0431$ $\beta = 0.164$ $\tau = 1.221$
Algorithm	Offline	Online	Online
Running time (s)	11.98	0.06	8.70
MAE space gap (m)	2.02	2.24	2.60
MAE velocity (m/s)	0.24	0.26	0.35
$\mathcal{L}_2$ strict string stable	No	No	No
$\mathcal{L}_\infty$ strict string stable	No	No	No

TABLE IV: Performance summary of all estimation methods on ACC data.

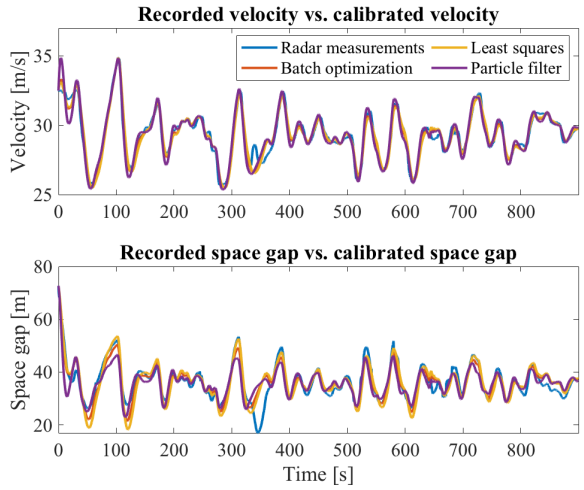

Fig. 7: Comparison between recorded vehicle velocity and space gap vs simulated for each model found.

These results are summarized in Table IV. All methods produce parameters that fit the data well, with some differences in the actual parameter values. A simulation using the estimated parameters from each method is shown in Figure 7. All methods have nearly identical velocity profiles, with slight differences in the space gap profiles. The batch optimization achieves both the lowest MAE velocity and space gap errors at 0.24 m/s and 2.02 m. This represents errors of 0.8% in velocity and 5.0% in space gap. The least-squares method has a comparable performance, with MAE values of 0.26 m/s and 2.24 m (0.87% in velocity and 5.6% in space gap). Finally, the PF estimated parameters produce slightly higher MAEs of 0.35 m/s and 2.60 m, which correspond to percent errors of 1.2% in velocity and 6.5% in space gap. Overall, the MAEs are comparable and low both in absolute values and in percent. Moreover, the models are similar in scale to those found in other works [9], [11].

We explore the errors in more detail. The largest error between the measured data and the ACC model run with estimated parameters occurs between roughly 325 seconds and and 375 seconds, in which the real ACC vehicle engages in an acceleration that is not captured by any of the calibrated models. This underscores that while each calibrated model produces a good overall reconstruction of the ACC vehicle,

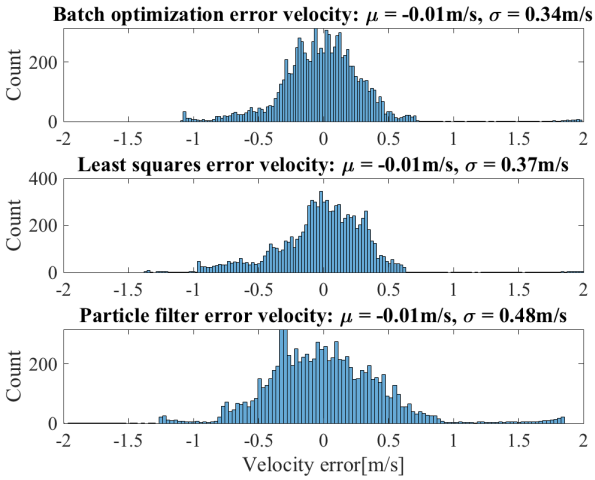

Fig. 8: velocity error distribution for each calibrated model.

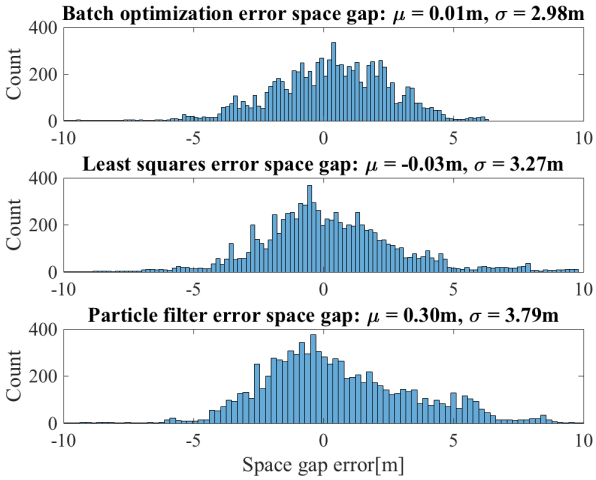

Fig. 9: space gap error distribution for each calibrated model.

none are able to perfectly describe the complex nonlinear vehicle dynamics controlled by a proprietary ACC system.

The histograms of the errors are shown in Figure 8 and 9. The average velocity and space gap error of both online methods is relatively similar to that of the batch method. All three methods return a model that has an average velocity error of within 0.01 m/s and similar standard deviations. Both the online methods produce the average space gap errors that are slightly more biased than the batch method, and standard deviations within 4 m. These MAE values are of the same order as is reported in other works [9], [11], [17]. Given that both the online methods have similar average velocity and space gap errors compared to the batch methods and the estimated models fit the recorded data relatively well, this suggests that both the RLS and the PF are viable online methods for learning ACC model parameters.

With the estimated parameters from each of the methods, we check the string stability of the vehicle under ACC control. Like all previous studies considering commercial ACC systems [9]–[11], [17], we find that the calibrated model of

the vehicle tested in this work is neither $\mathcal{L}_2$ or $\mathcal{L}_\infty$ strict string stable. The results are consistent across the different calibration methods, as summarized in Table IV.

With respect to the runtime, the recursive least-squares method is again the fastest, with a total computation time of 0.06 seconds to process the 15 minute dataset. The PF executes in 8.7 seconds, while the batch optimization method runs in 11.98 seconds. The runtime of the batch method is sensitive to the initial guess and the number of parameters to be estimated. The online methods have a distinct advantage in real-time applications, since they produce new estimates of the parameters as new data becomes incrementally available, and they can scale to arbitrarily long datasets.

VI. Conclusion

This work uses two online methods to estimate parameters of vehicles under control of a stock ACC system, and provides a corresponding parameter identifiability analysis for the estimators. The online methods used here are scalable and suitable for real time implementations, and produce comparable results to an offline batch optimization method. All methods are tested on a 2019 vehicle with ACC using sensor data from the stock vehicle platform as reported on the CAN bus. All methods indicate the vehicle ACC system is string unstable, adding to the findings of eight other ACC systems as reported in [9], [17]. We further intend to exploit the experimental platform used in this work, which allows critical velocity and space gap data to be recorded directly from the CAN bus of the vehicle. Such data could be valuable for a variety of experimental settings in which humans and automation systems interact in mixed traffic. We envision in our own work to generalize the application of the online system identification methods to study human driving behavior, using vehicles equipped with stock sensors similar to the ACC vehicle used in this work. As humans or automated vehicles may change driving behaviors depending on traffic conditions or environmental factors, it is important to apply online system identification algorithms to capture such behavioral changes characterized by the model parameters.

VII. Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant Nos. CMMI-1853913 (Wang) and CNS-1837652 (Gunter). This research was supported by the Inria associated team "ModEling autonoMous vEhicles iN Traffic flOw" (MEMENTO).

References

[1] R. Rajamani, *Vehicle dynamics and control*. Springer Science & Business Media, 2011.

[2] P. Ioannou, Z. Xu, S. Eckert, D. Clemons, and T. Sieja, "Intelligent cruise control: theory and experiment," in *Proceedings of 32nd IEEE Conference on Decision and Control*, Dec 1993, pp. 1885–1890 vol.2.

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

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

[5] D. Swaroop and J. Hedrick, "String stability of interconnected systems," *IEEE Transactions on Automatic Control*, vol. 41, no. 3, pp. 349–357, 1996.

[6] M. Makridis, K. Mattas, B. Ciuffo, F. Re, A. Kriston, F. Minarini, and G. Rognelund, "Empirical study on the properties of adaptive cruise control systems and their impact on traffic flow and string stability," *Transportation Research Record*, vol. 2674, no. 4, pp. 471–484, 2020.

[7] 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," 2020.

[8] R. E. Stern, S. Cui, M. L. D. Monache, R. Bhadani, M. Bunting, M. Churchill, N. Hamilton, R. Haulcy, H. Pohlmann, F. Wu, B. Piccoli, B. Seibold, J. Sprinkle, and D. B. Work, "Dissipation of stop-and-go waves via control of autonomous vehicles: Field experiments," *Transportation Research Part C: Emerging Technologies*, vol. 89, pp. 205 – 221, 2018.

[9] 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?" *arXiv:1905.02108*, 2019.

[10] V. L. Knoop, M. Wang, I. Wilmink, D. M. Hoedemaeker, M. Maaskant, and E.-J. Van der Meer, "Platoon of SAE level-2 automated vehicles on public roads: Setup, traffic interactions, and stability," *Transportation Research Record*, p. 0361198119845885, 2019.

[11] V. Milanés and S. E. Shladover, "Modeling cooperative and autonomous adaptive cruise control dynamic responses using experimental data," *Transportation Research Part C: Emerging Technologies*, vol. 48, pp. 285–300, 2014.

[12] C. Wu, A. M. Bayen, and A. Mehta, "Stabilizing traffic with autonomous vehicles," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 6012–6018.

[13] M. Čičić and K. H. Johansson, "Stop-and-go wave dissipation using accumulated controlled moving bottlenecks in multi-class ctm framework," in *2019 IEEE 58th Conference on Decision and Control (CDC)*, 2019, pp. 3146–3151.

[14] J. Wang, Y. Zheng, Q. Xu, J. Wang, and K. Li, "Controllability analysis and optimal controller synthesis of mixed traffic systems," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, 2019, pp. 1041–1047.

[15] M. L. Delle Monache, T. Liard, A. Rat, R. Stern, R. Bhadani, B. Seibold, J. Sprinkle, D. B. Work, and B. Piccoli, *Feedback Control Algorithms for the Dissipation of Traffic Waves with Autonomous Vehicles*. Cham: Springer International Publishing, 2019, pp. 275–299.

[16] Y. Sugiyama, M. Fukui, M. Kikuchi, K. Hasebe, A. Nakayama, K. Nishinari, S. ichi Tadaki, and S. Yukawa, "Traffic jams without bottlenecks—experimental evidence for the physical mechanism of the formation of a jam," *New Journal of Physics*, vol. 10, no. 3, p. 033001, mar 2008.

[17] G. Gunter, C. Janssen, W. Barbour, R. Stern, and D. Work, "Model based string stability of adaptive cruise control systems using field data," *IEEE Transactions on Intelligent Vehicles*, pp. 1–1, 2019.

[18] A. Duret, C. Buisson, and N. Chiabaut, "Estimating individual speed-spacing relationship and assessing ability of Newell's car-following model to reproduce trajectories," *Transportation Research Record*, vol. 2088, no. 1, pp. 188–197, 2008.

[19] V. Punzo, M. Montanino, and B. Ciuffo, "Do we really need to calibrate all the parameters? variance-based sensitivity analysis to simplify microscopic traffic flow models," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 1, pp. 184–193, Feb 2015.

[20] J. Monteil, R. Billot, J. Sau, C. Buisson, and N.-E. El Faouzi, "Calibration, estimation and sampling issues of car following model," in *Proceedings of the 92nd annual meeting of the Transportation Research Board*, 2014.

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

[22] V. Punzo and F. Simonelli, "Analysis and comparison of microscopic traffic flow models with real traffic microscopic data," *Transportation Research Record*, vol. 1934, no. 1, pp. 53–63, 2005.

[23] C. P. van Hinsbergen, H. W. van Lint, S. P. Hoogendoorn, and H. J. van Zuylen, "Bayesian calibration of car-following models," *IFAC Proceedings Volumes*, vol. 42, no. 15, pp. 91 – 97, 2009, 12th IFAC Symposium on Control in Transportation Systems.

[24] S. Hoogendoorn and R. Hoogendoorn, "Calibration of microscopic traffic-flow models using multiple data sources," *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences*, vol. 368, pp. 4497–517, 10 2010.

[25] I. Papamichail, N. Bekiaris-Liberis, A. I. Delis, D. Manolis, K.-S. Mountakis, I. K. Nikolos, C. Roncoli, and M. Papageorgiou, "Motorway traffic flow modelling, estimation and control with vehicle automation and communication systems," *Annual Reviews in Control*, vol. 48, pp. 325 – 346, 2019.

[26] L. Ljung, *System Identification: Theory for the User*, ser. Prentice Hall information and system sciences series. Prentice Hall PTR, 1999.

[27] H. Miao, X. Xia, A. S. Perelson, and H. Wu, "On identifiability of nonlinear ode models and applications in viral dynamics," *SIAM Review*, vol. 53, no. 1, pp. 3–39, 2011.

[28] J. Monteil and M. Bouroche, "Robust parameter estimation of car-following models considering practical non-identifiability," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, Nov 2016, pp. 581–588.

[29] J. Monteil, N. O'Hara, V. Cahill, and M. Bouroche, "Real-time estimation of drivers' behaviour," *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, pp. 2046–2052, 2015.

[30] D. Beckmann, M. H. Riva, M. Dagen, and T. Ortmaier, "Comparison of online-parameter estimation methods applied to a linear belt drive system," in *2016 European Control Conference (ECC)*, June 2016, pp. 364–369.

[31] V. Milanés, S. E. Shladover, J. Spring, C. Nowakowski, H. Kawazoe, and M. Nakamura, "Cooperative adaptive cruise control in real traffic situations," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 1, pp. 296–305, 2014.

[32] L. Xiao and F. Gao, "Practical string stability of platoon of adaptive cruise control vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1184–1194, 2011.

[33] L. Xiao, S. Darbha, and F. Gao, "Stability of string of adaptive cruise control vehicles with parasitic delays and lags," in *2008 11th International IEEE Conference on Intelligent Transportation Systems*, 2008, pp. 1101–1106.

[34] Z. Bareket, P. S. Fancher, Huei Peng, Kangwon Lee, and C. A. Assaf, "Methodology for assessing adaptive cruise control behavior," *IEEE Transactions on Intelligent Transportation Systems*, vol. 4, no. 3, pp. 123–131, 2003.

[35] G. Gunter, R. Stern, and D. B. Work, "Modeling adaptive cruise control vehicles from experimental data: model comparison," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, 2019, pp. 3049–3054.

[36] R. Wilson and J. Ward, "Car-following models: fifty years of linear stability analysis – a mathematical perspective," *Transportation Planning and Technology*, vol. 34, no. 1, pp. 3–18, 2011.

[37] 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 Transactions on Control Systems Technology*, vol. 27, no. 3, pp. 934–949, 2018.

[38] A. Doucet and A. Johansen, "A tutorial on particle filtering and smoothing: Fifteen years later," *Handbook of Nonlinear Filtering*, vol. 12, Jan 2009.

[39] Z. Chen, "Bayesian filtering: From kalman filters to particle filters, and beyond," *Statistics*, vol. 182, Jan 2003.

[40] S. C. Surace, A. Kutschireiter, and J.-P. Pfister, "How to avoid the curse of dimensionality: Scalability of particle filters with and without importance weights," *SIAM Review*, vol. 61, no. 1, pp. 79–91, 2019.

[41] D. Simon, *Optimal State Estimation*. John Wiley & Sons, Inc., 2006.

[42] S. Boyd and S. Sastry, "Necessary and sufficient conditions for parameter convergence in adaptive control," *Automatica*, vol. 22, no. 6, pp. 629 – 639, 1986.

[43] Y. Wang, G. Gunter, and D. B. Work, "Online parameter estimation of adaptive cruise control models with delays and lags," in *2020 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2020, to appear.

[44] R. Stern, G. Gunter, and D. B. Work, "Modeling and assessing adaptive cruise control stability: experimental insights," in *2019 6th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*, 2019, pp. 1–8.

Yanbing Wang is a Ph.D. student in Civil and Environmental Engineering and the Institute for Software Integrated Systems at Vanderbilt University. She earned her B.S. degree (2018) from the University of Illinois at Urbana-Champaign. Yanbing is an recipient of the Eisenhower Graduate Fellowship (2018 and 2019). Her research interests include traffic estimation and smart cities.

George Gunter is a Ph.D. student in Civil and Environmental Engineering and the Institute for Software Integrated Systems at Vanderbilt University. He earned his B.S. degree (2019) from the University of Illinois at Urbana-Champaign. His research interests include transportation cyber-physical systems and autonomous vehicles.

Matthew Nice is a M.Eng. student in Cyber Physical Systems and the Institute for Software Integrated Systems at Vanderbilt University. He earned his B.S.E. degree (2018) from Tulane University. His research interests include transportation cyber-physical systems and safety-critical systems.

Maria Laura Delle Monache is a research scientist in the Networked Controlled Systems team at Inria and in GIPSA-Lab (Department of Control) in Grenoble. Her research interest is mainly related to the mathematical and engineering aspects of traffic flow. In particular, she is interested in mathematical modeling, analysis, numerical approximation and control of traffic flow applications. Prior to Inria, she was a Postdoctoral researcher at Rutgers University Camden.

Daniel B. Work is an associate professor in Civil and Environmental Engineering and Institute for Software Integrated Systems at Vanderbilt University. Prof. Work earned his B.S. degree (2006) from the Ohio State University, and an M.S. (2007) and Ph.D. (2010) from the University of California, Berkeley, each in civil engineering. His research interests include transportation cyber physical systems. He is a recipient of the CAREER award from the National Science Foundation (2014) the Gilbreth Lectureship from the National Academy of Engineering (2018).