Proceedings of the 2024 Winter Simulation Conference
H. Lam, E. Azar, D. Batur, S. Gao, W. Xie, S. R. Hunter, and M. D. Rossetti, eds.

DATA ASSIMILATION FOR ONLINE CALIBRATION OF SIMULATION DIGITAL TWIN – A CASE STUDY WITH MULTIPLE MODEL PARAMETERS

Xiaolin Hu and Mingxi Yan

Dept. of Computer Science, Georgia State University, Atlanta, GA, USA

ABSTRACT

Calibrating a simulation digital twin using real-time observation data is a key requirement for making the simulation model aligned with a physical system under study. This paper applies a particle filter-based data assimilation framework to online calibration of simulation digital twin, with a focus on calibrating multiple model parameters. An overview of the problem formulation and the particle filter-based data assimilation is provided. The combination effect of multiple model parameters and its impact on multi-parameter calibration is discussed. Experiment results based on a simulation case study example demonstrate the effectiveness of the data assimilation for online model calibration of simulation digital twin models.

1 INTRODUCTION

Digital twin is a new technology that holds promise for supporting rapid and accurate analysis and real-time decision making for many systems. While different models have been used in different contexts, a key type of digital twin model is *simulation models* (also referred to as *simulation digital twins* in this paper) that explicitly model the dynamic behavior of physical systems. A simulation digital twin differs from other models by modeling a system's state and state transition over time. The dynamic nature of these models makes it possible to run simulations to analyze and predict future behaviors starting from some initial states. Simulation digital twins find applications in many domains, such as manufacturing process, city traffic, and wildland fire management.

To achieve full potential of simulation digital twin, it is crucial to synchronize a simulation model with the corresponding physical system using real-time observation data collected from the system (Fuller et al. 2020). The synchronization is necessary because a simulation run needs to start from an initial state, which needs to match the real-time state of a physical system in order to support real-time analysis/prediction. Furthermore, a simulation model needs to be properly parameterized based on the real-time characteristics of a physical system in order to achieve accurate simulation results. Thus, to support synchronization two tasks are needed: *dynamic state estimation* and *online model calibration*. The former allows a simulation run to be initialized to the right state, and the latter allows a simulation model to be correctly parameterized. Both dynamic state estimation and online model calibration are key activities of Dynamic Data Driven Simulation (DDDS), which refers to a new simulation paradigm where a simulation system continuously and systematically assimilates real-time data from a system in operation to support real-time prediction and analysis for the system (Hu 2023a).

Dynamical state estimation and online model calibration are challenging tasks because in many cases the state and parameters that need to be estimated/calibrated are hidden, i.e., they cannot be directly observed or computed from observation data. Thus, one need to infer the state and parameter values based on real-time observation data. An important approach for state/parameter estimation is *data assimilation*. Data assimilation is a methodology that combines observation data with a dynamic model of a system to optimally estimate the evolving state of the system. It has been used in many science fields such as meteorology and geosciences, but received less attention in the modeling and simulation community. Data assimilation in science fields typically deals with continuous models with continuous state variables. Recently, Hu (2023a) systematically introduced data assimilation as an enabling technology for DDDS that

involves discrete simulation models with discrete or hybrid states. A tutorial on Bayesian sequential data assimilation for discrete simulation models can be found in Hu (2023b).

With growing interest in digital twin technologies, applying data assimilation to simulation digital twins becomes an important research topic. In previous work (Hu and Yan 2024), we developed a particle filter-based data assimilation framework for online model calibration in discrete event simulations. This paper extends previous work by applying the framework to online calibration of simulation digital twins, with a focus on calibrating multiple model parameters. A challenge of multi-parameter calibration is associated with the combination effect of multiple parameters, i.e., there may exist multiple combinations of parameter values that produce the same or similar observation data. As a result, the true parameter values cannot be effectively distinguished from the observation data. We provide an in-depth discussion of this issue and its impact on online model calibration. To evaluate the particle filter-based data assimilation for online calibration of digital twins, a case study example of a one-way traffic control system simulation is presented, and experiment results are provided.

It is important to note that the problem of online model calibration is different from the traditional problem of calibrating simulation models in an offline fashion. Offline model calibration is considered part of the model evaluation process that includes activities such as verification, calibration, and validation (Rykiel 1996). Typical offline model calibration methods include parameter sweeps, hill climbing, simulated annealing, and genetic algorithms (Malleson 2014). Offline model calibration is usually formulated as a global optimization problem by using historical data. On the other hand, online model calibration uses real-time data to adjust a simulation model to make it match the real-time characteristics of a system. Compared to offline calibration, less work exists for online model calibration. An extended Kalman filter-based online calibration algorithm was developed to support real-time calibration of large-scale traffic simulators (Zhang 2020). Within the context of digital twin, a machine learning-based method was developed to support online autonomous calibration of digital twin models for nuclear power plants (Song et al. 2022). A particle filter-based method was used to continuously calibrate a digital twin model, and its performance was compared with static and sequential Bayesian calibration approaches (Ward et al. 2021). Titscher et al. (2023) developed a Bayesian calibration method and applied it to online model calibration using real measurement data from a lab-based demonstrator bridge.

2 ONLINE CALIBRATION OF SIMULATION DIGITAL TWIN

A simulation digital twin typically has many parameters that characterize the physical system under study. While some parameters can be determined or calibrated offline using historical data or domain knowledge, others need to be calibrated in an online fashion based on real-time observation data. This is because some characteristics of the physical system are known only after the system operates in the field, and thus the corresponding parameters can only be determined when the system works in real time. Furthermore, a system's characteristics may change or shift over time due to changing operating conditions. For these systems, the digital twin models' parameters are not static – they are dynamic parameters whose values need to be calibrated online based on real-time observation data.

Online model calibration is a challenging task due to multiple reasons. First, the parameter values often cannot be directly observed or computed from observation data. Thus, their values need to be inferred indirectly from the observation data. For example, a traffic light digital twin model may need to calibrate its parameters characterizing the durations of the red/green light. Nevertheless, there may be no direct observations about the traffic light state or light switch time. In this case, one need to infer the parameter values based on other available observation data, such as the number of cars that passed the intersection in previous time intervals. Second, a system's behavior is influenced not only by its characteristics but also by its state that dynamically change over time. This means it is often difficult to separate the task of parameter estimation from the task of state estimation because both impact the observation data. Thus, the parameters and dynamically-changing states need to be estimated at the same time. Third, often there is a need to calibrate multiple model parameters. The multiple parameters bring the combination effect of multiple parameters that will be discussed further in Section 2.2.

2.1 Overview of Problem Formulation and Particle Filter-based Data Assimilation

The goal of online calibration of simulation digital twin is to dynamically estimate the parameters of a simulation digital twin based on real-time observation data to make the model more accurately capture the characteristics of a physical system. As an estimation problem, it can be defined in a probabilistic way. Let $y_k := y(t_k)$ be the observation data y at time t_k; $y_{0:k} := (y(t_0), y(t_1), \dots, y(t_k))$ be the sequence of observation data up to time t_k. Let θ be the parameter vector to be calibrated. We define $\theta_k := \theta(t_k)$ be the parameter vector θ at time t_k. Then the online calibration problem can be defined as $p(\theta_k|y_{0:k})$, i.e., computing the probability distribution of θ_k conditioned on the observation data $y_{0:k}$. This is carried out in an iterative way: when new observation data become available at time t_{k+1}, a new calibration is carried out to update the parameter estimate.

To apply data assimilation to online model calibration, a common approach is to formulate it as a joint state-parameter estimation problem. In this approach, the to-be-estimated parameters are included as part of the state vector that needs to be estimated. Let x_k be the n-dimensional state vector and θ_k be the h-dimensional parameter vector that need to be estimated online at step k. Typically, the set of parameters that need to estimated online is a small subset of all the parameters of a simulation digital twin. We define an augmented state vector $\hat{x}_k$ by appending the parameter vector θ_k to the state vector x_k, i.e., $\hat{x}_k = \binom{x_k}{\theta_k}$ or $\hat{x}_k = (x_{1,k}, x_{2,k}, \cdots, x_{n,k}, \theta_{1,k}, \theta_{2,k}, \cdots, \theta_{h,k})^T$, where $\hat{x}_k$ is a $n+h$ dimensional vector, $x_{i,k}$ $(i = 1, \dots, n)$ is the ith element of the state vector, and $\theta_{j,k}$ $(j = 1, \dots, h)$ is the jth element of the parameter vector.

In data assimilation, a dynamic system is generally modeled as a dynamic *state-space model* that includes a *state transition model* and a *measurement model*. With the augmented state vector $\hat{x}_k$, the state transition model can be defined as follows.

$$\hat{x}_k = \binom{x_k}{\theta_k} = \hat{f}_k(\hat{x}_{k-1}, u_k, \gamma_k, \zeta_k) = \begin{pmatrix} f_k(x_{k-1}, \theta_{k-1}, u_k, \gamma_k) \\ \theta_{k-1} + \zeta_k \end{pmatrix}. \tag{1}$$

This model includes two parts. The first part $x_k = f_k(x_{k-1}, \theta_{k-1}, u_k, \gamma_k)$ describes how the state x_k evolves over time. For a simulation digital twin, the $f_k(\,)$ is defined by the simulation model that specifies a systems' state transition based on its previous state x_{k-1} and the model parameters θ_{k-1}. The u_k is the external input of the simulation model at step k, and γ_k is the process noise that models the uncertainty of the state transition. The second part $\theta_k = \theta_{k-1} + \zeta_k$ models how the parameters θ_k evolve over time. A common approach is to add small random perturbations to the parameter values in each step of the transition (Liu and West 2001). Typically, the random perturbations are drawn from a zero-mean Gaussian distribution, i.e., $\zeta_k \sim N(0, W_k)$, where W_k is a diagonal covariance matrix for the Gaussian noises that has variance σ_j^2 for the jth parameter. Adding random perturbations to the parameter values allows generating new parameter values in each step of the data assimilation. This supports robust estimation even when parameters' initial values are far away from the true values. Large variances of the Gaussian noises lead to large changes of the parameter values in each step. Large variances may be necessary if the estimation has not converged or if one expects the parameter values change dynamically in a fast pace. Otherwise, small variances are preferred. For example, when a parameter under estimation is expected to be a static parameter, a small variance will lead to more stable estimation results.

With the augmented state vector $\hat{x}_k$, the measurement model is defined as

$$y_k = \hat{g}_k(\hat{x}_k, \varepsilon_k) \stackrel{\text{def}}{=} g_k(x_k, \varepsilon_k), \tag{2}$$

where $\hat{g}_k()$ maps from the augmented state vector $\hat{x}_k$ to the observation data vector y_k, and ε_k is the measurement noise. The $\hat{g}_k(\hat{x}_k, \varepsilon_k)$ is equivalent to the function $g_k(x_k, \varepsilon_k)$ that defines the mapping from the state x_k to the observation data y_k. This is because the state space formulation of data assimilation

generally assumes the Markov property, which means the observation data y_k at step k is completely defined by the state x_k at that step. More explanations of the state transition model and measurement model described above can be found in Hu (2023a).

The state space formulation described above allows us to carry out data assimilation to estimate the state and model parameters at the same time. In this work, we employ particle filters to carry out data assimilation. Particle filters are a set of algorithms that use the Monte Carlo techniques to realize sequential Bayesian filtering. They are non-parametric filters that work well with simulation models with discrete or hybrid states. They also have the advantage of working with systems that have non-linear non-gaussian behaviors, which is the case for many simulation models. Specifically, we choose to use the bootstrap filter algorithm (Doucet et al. 2001; Arulampalam et al. 2002) to carry out data assimilation. The bootstrap filter algorithm uses the state transition model to evolve particles during the sampling step. For the joint state-parameter estimation problem, the state transition model is defined by Equation (1) described above. Another major advantage of the bootstrap algorithm is that it simplifies the computation of particles' importance weights, where the weight is defined by the likelihood probability that can be computed from the measurement model (Equation (2)).

In particle filtering, the belief distribution of the state under estimation is represented by a set of samples, each of which is called a *particle*. A particle is a concrete instantiation of the augmented state vector $\hat{x}$. Let $\{\hat{x}_k^{(1)}, \hat{x}_k^{(2)}, \cdots, \hat{x}_k^{(N)}\}$ be the set of particles representing the posterior distribution $p(\hat{x}_k|y_{1:k}, u_{1:k})$ at time step k, where N is the size of the particle set. The bootstrap algorithm has a standard algorithmic structure that includes three sub-steps in each data assimilation step: 1) *sampling*, 2) *importance weight computation*, and 3) *resampling*. The sampling step evolves each particle to a new state using the state transition model (Equation (1)). The outcome of the sampling step is a set of particles representing the prior distribution of the state estimate. Then the importance weight of each particle is computed according to its likelihood probability of observing the observation data at this step. A higher likelihood probability leads to a larger weight and a lower likelihood probability leads to a smaller weight. After all particles' weights are computed, a set of weighted particles are formed, which act as an intermediate approximation for the posterior distribution at step k. Subsequently, the resampling step selects the particles according to their importance weights. The particles with large weights are selected multiple times while the particles with small weights may be eliminated. The set of resampled particles represent the final posterior distribution of the state at this step, which also serve as the input for the next iteration of the bootstrap filter algorithm. More details about the bootstrap filter algorithm and its implementation for data assimilation can be found in Hu (2023a; 2023b) and Hu and Yan (2024).

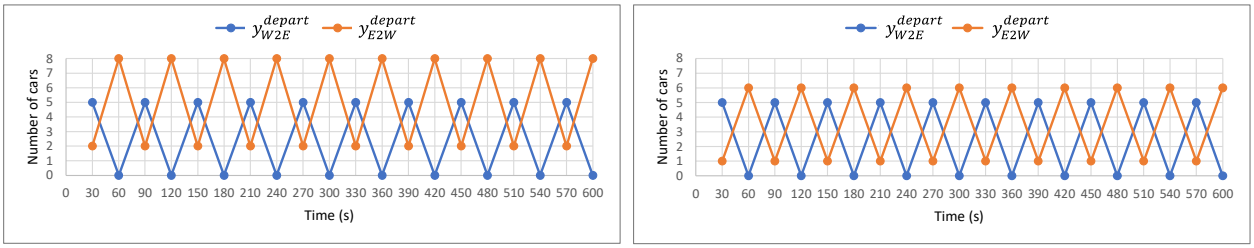
Initialization of the bootstrap filter algorithm needs to generate N initial particles following the belief of the initial state. When knowledge about the initial state is available, the initial set of particles can be generated using that knowledge. Otherwise, a common practice is to generate the initial particles randomly covering a wide state space in a uniform way. For the model parameters that need to be calibrated online, their initial values may be sampled around the baseline values based on an offline model calibration.

2.2 The Combination Effect of Multiple Model Parameters

The problem formulation and particle filter-based data assimilation described above are applicable to both single-parameter and multi-parameter calibration. Compared to calibrating a single parameter, calibrating multiple parameters works with a higher dimensional augmented state vector, which makes it more difficult for the data assimilation to converge to the true state/parameter values. More importantly, multi-parameter calibration brings another challenge: there may exist multiple combinations of parameter values that make a physical system to exhibit the same or similar observation data. In this case, the true parameter values of the physical system cannot be effectively inferred or distinguished from the observation data. This issue was also reported in an earlier work of parameter calibration for wildfire spread simulation (Bai et al. 2011).

To better illustrate this issue, we consider a simplified version of the one-way traffic control system that will be described in Section 3. This simplified one-way traffic control system includes a road segment controlled by a traffic light. The system assigns green light to the west-to-east moving direction for a duration of T_{W2E}, and then switch to the east-to-west direction for a duration of T_{E2W}. It then switches back and repeats the cycle. During a green light, cars move through the road segment one by one: a car enters into the road segment only after its front car has finished crossing the road segment. It takes a constant time (denoted as T_{car}) for all cars to move across the road segment. The observation data is the number of cars that finish crossing the road segment from west to east (denoted as y_{W2E}^{depart}) in a 30-second time interval, and these data are collected every 30 seconds starting from time 0.

In this example, the T_{W2E}, T_{E2W}, and T_{car} are parameters of the one-way traffic control system, and the y_{W2E}^{depart} is the observation data. During rush hours when there are many cars on both sides of the road segment, the number of cars that can cross the road segment in a 30-second time interval is directly related to the green light duration of the corresponding direction within the 30-second interval as well as the time for a car to cross the road segment. Figure 1 shows the y_{W2E}^{depart} observation data (blue line) for two cases that have different parameters: 1) Case 1: $T_{car} = 4.0s$, $T_{E2W} = 40s$, $T_{W2E} = 20s$, and 2) Case 2: $T_{car} = 5.0s$, $T_{E2W} = 35s$, $T_{W2E} = 25s$. As can be seen, the two cases have the exactly same y_{W2E}^{depart}: the first data point (at time=30s) is 5 and the second data point (at time=60s) is 0, and then the remaining data points repeat the 5-0-5-0…cycle. This is because in Case 1 the west-to-east green light is 20s and it takes 4.0s for a car to cross the road segment. Thus, in the first 30s only 5 cars can cross the road segment from west to east; and in the second 30s no car can cross the road segment from west to east because the traffic light is in the opposite direction. In Case 2 the west-to-east green light is 25s and it takes 5.0s for a car to cross the road segment. Again, 5 cars can cross the road segment from west to east in the first 30s, and 0 car can cross the road segment from west to east in the second 30s. This is the same as in Case 1. In both cases, a new traffic light cycle starts at time=60s that repeats the observation data from the previous cycle.



Case 1: $T_{car} = 4.0s$, T_{E2W}=40s, $T_{w2E} = 20s$ Case 2: $T_{car} = 5.0s$, T_{E2W}=35s, $T_{w2E} = 25s$

Figure 1: Observation data for two cases with different parameter values.

For this system, there exist many other combinations of parameter values that result in the same y_{W2E}^{depart} (i.e., 5-0-5-0-5-0…) as shown in Figure 1. For example, the combination of $T_{W2E} = 30s$, T_{E2W}=30s, and $T_{car} = 6.0m/s$ will also make the y_{W2E}^{depart} be 5-0-5-0…. In fact, as long as $T_{W2E} \leq 30s$, and $T_{W2E}/T_{car} = 5$, and $T_{W2E} + T_{E2W} = 60s$, then the y_{W2E}^{depart} will be the same as shown in Figure 1.

The fact that there exist two or more combinations of parameters that result in the same observation data means that it is impossible to determine the exact parameter values from observation data. In the above example, we assume the observation data has zero noise and the system has deterministic behavior (e.g., all cars use the same constant time to cross the road segment and the traffic light has a fixed schedule). To complicate things further, observation data are noisy and there exists uncertainty in system behavior. The noisy observation data and uncertain system behavior bring more challenges for precisely estimating the true parameter values from observation data. A major advantage of particle filter-based data assimilation is that it can represent multimodal distributions or other arbitrary distributions that are needed for handling

the combination effect of multiple parameters. For example, when there are two possible combinations of parameter values, particles will represent a bimodal distribution that has two local peaks. When there are multiple or infinite combinations of parameter values, the corresponding distributions can be represented by particles too. This will be demonstrated by the experiment results in Section 4.2.

The challenge associated with the combination effect of multiple parameters is closely related to the limited observability of observation data used in data assimilation. In the above example, the observation data y_{W2E}^{depart} carries limited information that is insufficient to distinguish the parameter values. When this happens, other observation data (if available) may be used to help distinguish the parameter values. For example, let us assume the observation data (denoted as y_{E2W}^{depart}) of the number of cars that finish crossing the road segment from east to west in every 30s is also available, as shown in Figure 1 (red line). One can see that the two cases have different y_{E2W}^{depart}: Case 1's y_{E2W}^{depart} are 2-8-2-8…, and Case 2's y_{E2W}^{depart} are 1-6-1-6… When assimilating both data of y_{W2E}^{depart} and y_{E2W}^{depart}, the data assimilation should be able to distinguish the two cases from one to another. Section 4.2 shows some results to demonstrate this.

3 THE CASE STUDY EXAMPLE

We consider a one-way traffic control system as illustrated in Figure 2, which is adapted from the example originally described in Hu (2022). During road construction, the one-way traffic control is managed by two persons deployed to the west and east ends of the road segment. Each person carries a STOP/SLOW hand-held traffic paddle to control the traffic, where the STOP sign means cars should stop and wait, and the SLOW sign means cars can slowly move ahead to pass the road segment. It is assumed that the two persons coordinate and always use the STOP/SLOW signs in opposite directions. In the following description, we refer to the STOP sign as the red traffic light and the SLOW sign as the green traffic light, and refer to cars' moving directions as west-to-east (also called *east-moving*) and east-to-west (also called *west-moving*). During the time when the traffic light is green on a specific direction, the arrival cars moving in the opposite direction are queued. The queues at the west side and east side of the road segment are named as the west-side queue and east-side queue, respectively. The cars at both sides of the road segment arrive randomly and independently, modeled by two Poisson distributions that have arriving rates of $\lambda_{eastMov}$ and $\lambda_{westMov}$ for the east-moving and west-moving cars, respectively.

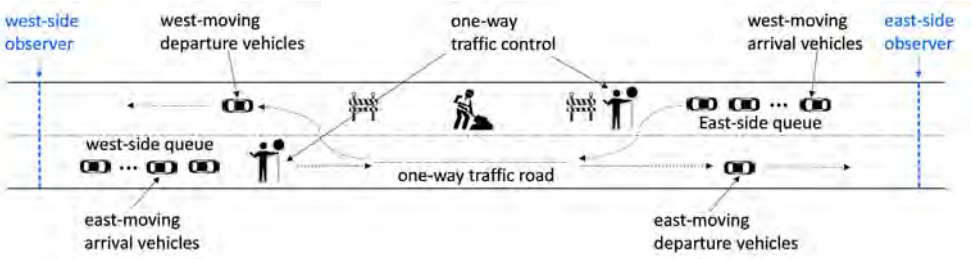


Figure 2: The one-way traffic control system. Adapted from Hu (2022).

To ensure safety, only one car is allowed to move on the road segment at any time. The time it takes for a car to cross the road segment is modeled as a random number drawn from a truncated normal distribution that has mean T_{car} (seconds), variance $\sigma^2 = 0.5^2$, and lies within the range of $[T_{car} - 1, T_{car} + 1]$. During a green light period, the traffic-control person on the corresponding side would signal a car to move ahead only after the previous car has finished crossing the road segment. The traffic control system switches the traffic lights using two rules: 1) Rule 1: switch the traffic light if the elapsed time for the current moving direction reaches a pre-defined threshold. The pre-defined thresholds for the west-to-east and east-to-west moving directions may be different, described by two parameters T_{w2E} and T_{E2W}, respectively. 2) Rule 2: if the current moving direction has no car waiting and the opposite direction has cars waiting, switch the

traffic light even if the pre-defined time threshold has not reached. We note that in both cases, the traffic light switches only after the road segment is cleared if there is already a car moving on the road.

A discrete event simulation model was developed to model this system based on the DEVS formalism (Zeigler et al. 2000). The DEVS model includes three atomic models: *eastMovCarGenr*, *westMovCarGenr*, and *oneWayTrafficRoad*. The *eastMovCarGenr* generates the east-moving traffic arriving at the west side of the road segment. The *westMovCarGenr* generates the west-moving traffic arriving at the east side of the road segment. The *oneWayTrafficRoad* models the one-way traffic road segment, including the traffic control logic as well as the time for cars to cross the road segment. This model has two input ports *eastMovArrival* and *westMovArrival* that receive cars generated from the *eastMovCarGenr* and *westMovCarGenr*, respectively. The cars finishing crossing the road segment are sent out through the *eastMovDeparture* and *westMovDeparture* output ports.

The *oneWayTrafficRoad* model use several parameters. In this study, we consider the following three parameters as candidates of online model calibration:

- T_{car}: this parameter specifies the average time for a car to cross the road segment.
- T_{W2E}: this parameter specifies the threshold for the traffic light to switch from west-to-east direction to east-to-west direction under Rule 1.
- T_{E2W}: this parameter specifies the threshold for the traffic light to switch from east-to-west direction to west-to-east direction under Rule 1.

To collect observation data from the system, observers (sensors) are deployed at the west-side and east-side locations as marked in Figure 2. The east-side observer is able to count the number of cars moving crossing its location for the west-to-east departure cars (denoted as y_{W2E}^{depart}) and east-to-west arrival cars (denoted as $y_{E2W}^{arrival}$). Similarly, the west-side observer is able to count the number of cars moving crossing its location for the east-to-west departure cars (denoted as y_{E2W}^{depart}) and west-to-east arrival cars (denoted as $y_{W2E}^{arrival}$). Each observer reports data every 30 seconds. It does not record the specific time that a car crosses the observation location – all it reports is the total number of cars that have departed and arrived in the past time interval. The data reported by the observer is noisy, with a 10% noise added to the actual number of cars crossing the observer location.

4 EXPERIMENT RESULTS

We use the identical twin experiment (Hu 2023a; Hu and Yan 2024) to evaluate the data assimilation for online parameter calibration. In the identical twin experiment, a simulation is first run to represent the "physical system" under study. The observation data obtained from this simulation are regarded as the observation data collected from the physical system, and the state trajectory recorded and parameters used in this simulation are considered the "true" state and "true" parameters, respectively. Using the collected observation data, data assimilation is then carried out and the state/parameter estimates are checked against the "true" state/parameters. In this work, the first simulation that serves as the physical system is based on the same simulation model as the one used in data assimilation. Nevertheless, it has different model configurations including different initial condition, input trajectories of west-side and east-side car arriving, and random number seeds. More importantly, it uses model parameters that are "unknown" to the data assimilation. The goal of data assimilation for online model calibration is to estimate the unknown model parameters of the physical system based on real-time data that are collected from the physical system.

The data assimilation carries out joint state-parameter estimation. For the one-way traffic control system, the state variables that need to be estimated include the *westSideQueue* size and *eastSideQueue* size, the traffic light state (i.e., red or green), and the elapsed time in the current traffic light state. Our previous work (Hu and Yan 2024) has reported results on joint state-parameter estimation involving a single parameter for this system. We observed similar state estimation results when carrying out joint state-parameter estimation involving multiple parameters. Due to this reason, in the following experiments we skip the state estimation results and focuses only on the results for the multiple parameters.

In all the experiments except for the ones in Section 4.2, the Poisson distributions used to generate incoming cars have the arriving rates $\lambda_{eastMov} = 1/9$ (1 car per 9 seconds in average) and $\lambda_{westMov} = 1/7$ (1 car per 7 seconds in average). With these arriving rates, in the beginning there are not many cars on either side of the road segment. The small number of cars means that the traffic light can switch frequently in the beginning because once one side has no car waiting, the traffic light switches even when the elapsed time has not reached T_{W2E} or T_{E2W} (due to the Rule 2 described in Section 3). This makes it difficult for the data assimilation to infer the true T_{W2E} and T_{E2W} in the beginning, as will be observed from the experiment results. All the experiments in this section use the data assimilation step of 30s, which is the same as how often observation data are collected. We run 200 steps of data assimilation (6000s) for all the experiments. Except for the last experiment in Section 4.2, all experiments assimilate only the observation data collected from the east-side observer, i.e., y_{W2E}^{depart} and $y_{E2W}^{arrival}$. In all the experiments, the particle filter algorithm uses 5000 particles.

4.1 Online Calibration of Two and Three Parameters

Our first experiment studies online model calibration involving two parameters. In this experiment, the one-way traffic control system uses the same time threshold for the traffic light to switch from west to east and from east to west. In other words, the system uses only one parameter that is shared by T_{W2E} and T_{E2W}. For simplicity, we refer to this parameter as $T_{trafficLight}$. This parameter and the other parameter T_{car} of the physical system are unknow and need to be estimated in real time based on real-time observation data.

To set up the experiment, we make the physical system use different combinations of T_{car} and $T_{trafficLight}$ to generate observation data. Four combination cases are considered, including: Case 1: $T_{car} = 4.0s$ and $T_{trafficLight} = 90s$; Case 2: $T_{car} = 4.0s$ and $T_{trafficLight} = 150s$; Case 3: $T_{car} = 5.0s$ and $T_{trafficLight} = 90s$; and Case 4: $T_{car} = 5.0s$ and $T_{trafficLight} = 150s$. Figure 3 shows the parameter estimation results for the four combination cases, each of which includes two charts. The top shows the result for the T_{car} parameter, and the bottom shows the result for the $T_{trafficLight}$ parameter. In each chart, the horizontal axis represents the time (seconds) and the vertical axis represents the corresponding parameter value. The blue line (denoted with a "*true_*" prefix) is the true parameter value from the physical system that needs to be estimated. The red line (denoted with a "*ave_*" prefix) is the estimation result from the data assimilation. The estimation result in each step is averaged from all the particles of that step.

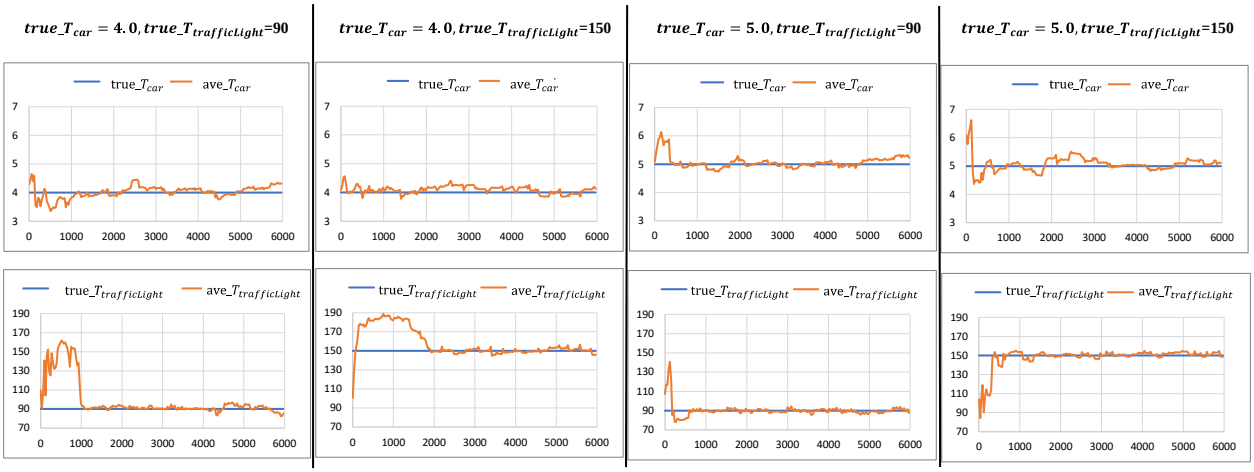


Figure 3: Online calibration of two model parameters.

As can be seen, in all the cases, the estimates converge to the true parameter values within some error bounds after some time. The time of convergence is different. For the case of $T_{car} = 4.0s$ and

$T_{trafficLight} = 90s$, it takes about 1000s for the $T_{trafficLight}$ to converge to the true state. For the case of $T_{car} = 4.0s$ and $T_{trafficLight} = 150s$, it takes about 2000s to converge. The longer convergence time is due to the fact that when $T_{trafficLight} = 150s$, it is easier for all the waiting cars on one side to finish crossing the road segment, and thus make the traffic light switch earlier. The early switch of traffic light makes it more difficult to infer that the $T_{trafficLight}$ is actually150s. A similar pattern can be observed when $T_{car} = 5.0s$. Comparing the cases of $T_{car} = 4.0s$ and $T_{car} = 5.0s$, we can see that the $T_{car} = 5.0s$ cases converge earlier. This is because cars move faster when $T_{car} = 4.0s$, which makes it more likely for all the cars on one side to finish crossing the road segment, and thus make the traffic light switch earlier.

The results shown in Figure 3 are based on four specific data assimilation runs. To quantitatively show the effectiveness of the parameter estimation, for each case we carry out 20 independent data assimilation runs and compute the Root Mean Square Error (RMSE) of the data assimilation results. The RMSE is calculated based on the differences between the estimated parameter and the true parameter, averaged from all 20 runs and from all the particles. The RMSE of the two parameters for the fours combination cases described above are: Case 1: RMSE_T_{car} = 0.6, RMSE_$T_{trafficLight}$= 3.6; Case 2: RMSE_T_{car} = 0.6, RMSE_$T_{trafficLight}$= 4.2; Case 3: RMSE_T_{car} = 0.6, RMSE_$T_{trafficLight}$= 2.5; Case 4: RMSE_T_{car} = 0.6, RMSE_$T_{trafficLight}$= 2.9. As can be seen, they are all relatively small, indicating the effectiveness of the data assimilation for the online parameter calibration.

Our next experiment studies online model calibration involving three parameters. To set up the experiment, we make the physical system use different combinations of T_{car}, T_{E2W} and T_{W2E} to generate observation data. The T_{car}, T_{E2W} and T_{W2E} are unknown to the data assimilation and need to be estimated based on real-time observation data. Figure 4 shows a specific simulation run for the case when the true $T_{car} = 4.0s$, $T_{E2W} = 120s$, and $T_{W2E} = 80s$. Similar as in Figure 3, the blue line is the true parameter value and the red line is the estimation result from the data assimilation. The estimation result in each step is averaged from all the particles of that step.

Figure 4: Online calibration of three parameters. The true $T_{car} = 4.0s$, $T_{E2W} = 120s$, $T_{W2E} = 80s$.

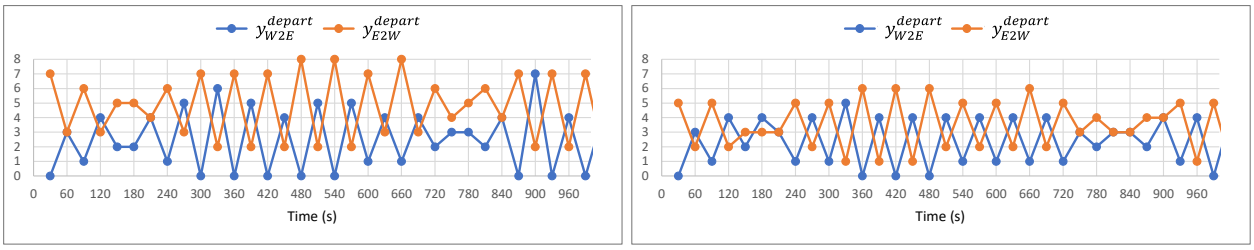
One can see that the estimated T_{car}, T_{E2W} and T_{W2E} all converge to the corresponding true values after some time and stayed close to the true values for the rest of the data assimilation. Specifically, it took less than 200s for the estimated T_{car} to converge to the true value, and it took a bit more than 1000s for the estimated T_{E2W} and T_{W2E} to converge the true values. We note that the time for the T_{E2W} and T_{W2E} to converge is significantly influenced by the number of cars arriving at the west or east side of the road segment. When there is less cars arriving, the traffic light can switch early frequently, making it more difficult for the T_{E2W} and T_{W2E} to converge (due to the same reason as explained for the two-parameter cases). Again, to quantitatively show the parameter estimation results, we carry out 20 independent data assimilation runs for this case of $T_{car} = 4.0s$, $T_{E2W} = 120s$, and $T_{W2E} = 80s$ and compute the RMSE values. The results are: RMSE_T_{car} = 0.6, RMSE_T_{E2W}= 4.3, and RMSE_T_{W2E}=4.3.

4.2 Studying the Combination Effect of Multiple Parameters

This experiment evaluates and demonstrates how the particle filter-based data assimilation works when there exists the combination effect of multiple parameters as described in Section 2.2. To set up this

experiment, we make the physical system have the same parameters as the ones described in Section 2.2. Specifically, we consider two cases of the physical system that have the following parameters: 1) Case 1: $T_{car} = 4.0s$, $T_{E2W} = 40s$, $T_{W2E} = 20s$, and 2) Case 2: $T_{car} = 5.0s$, $T_{E2W} = 35s$, $T_{W2E} = 25s$. For the illustrative example described in Section 2.2, these two sets of parameters make the system generate exactly the same y_{W2E}^{depart}. The one-way traffic control system considered in this experiment is more complex than the illustrative example from Section 2.2, due to the following factors: 1) observation data is noisy; 2) cars arrive randomly; 3) traffic light can switch early if the current direction has no car waiting; 4) the time to cross the road segment is a random number; and 5) traffic light switches only after the road segment is cleared if there is a car already on the road. This last factor means in most cases the actual green time is longer than T_{W2E} (or T_{E2W}) because when T_{W2E} (or T_{E2W}) is reached the traffic light still needs to stay in green until the car (if any) on the road segment finishes crossing it. For better demonstration, in this experiment both the west side and east side have high car arriving rates: $\lambda_{eastMov} = 1/2$ and $\lambda_{westMov} = 1/2$. With these high arriving rate, both sides of the road segment can quickly accumulate a large number of cars that need to cross the road segment. As a result, the complexity associated with the factors 2) and 3) described above is minimized.

Figure 5 shows the observation data y_{W2E}^{depart} and y_{E2W}^{depart} for a typical run of the two cases of the physical system. As can be seen, the y_{W2E}^{depart} data between the two cases are different, which are also different from what was shown in Figure 1. These differences are due to the noisy observation data and the complexity and uncertainty associated with the system behavior as explained above. Despite the differences, one can see that there is significant similarity between the two cases' y_{W2E}^{depart}. This similarity of observation data will make it difficult for the data assimilation to distinguish the two sets of parameters during online model calibration.



Case 1: $T_{car} = 4.0s$, T_{E2W}=40s, $T_{w2E} = 20s$ Case 2: $T_{car} = 5.0s$, T_{E2W}=35s, $T_{w2E} = 25s$

Figure 5: Observation data for two cases of the physical system that have different parameters.

We carry out particle filter-based data assimilation to estimate the three parameters for the two cases. We first assimilate only the observation data y_{W2E}^{depart} and $y_{E2W}^{arrival}$. Figure 6 shows the estimation results for the three parameters T_{W2E}, T_{E2W}, T_{car} for both cases. Similar as before, the blue line is the true parameter value and the red line is the estimation result, which is averaged from all the particles in each step. As can be seen, in both cases, the T_{car} and T_{E2W} have relatively large errors. Specifically, the estimated T_{car} is larger than the true T_{car} for both cases, and the estimated T_{E2W} is smaller than the true T_{E2W} for both cases. Compared to T_{car} and T_{E2W}, the estimated T_{W2E} has relatively small error but still does not converge to the true value – it is larger than the true value for both cases.

The results from Figure 6 show that the particles did not converge to the true parameter values. This is due to the combination effect of the multiple parameters as described in Section 2.2. Since there exist multiple combinations of parameter values that can produce similar y_{W2E}^{depart} as the ones sown in Figure 5, all these combinations of parameter values are represented by the particles during data assimilation. When this happens, one should not rely on the average from all the particles as the data assimilation results, because the posterior distribution represented by the particles is not a typical normal distribution. To further

check this, Figure 7(a) shows the histogram of all the particles' estimates for T_{car} at step 100 (time=3000s) for the Case 2 when assimilating only the east-side observation data y_{W2E}^{depart} and $y_{E2W}^{arrival}$. As can be seen, the posterior distribution represented by the particles spans a wide range between 4.1 to 7.6. In particular, it forms a plateau shape from 5.3 to 6.6, indicating that there is high likelihood for the T_{car} to be any value in this range. We note that the true $T_{car} = 5.0$ does not have the highest representation by the particles but still has a significant representation.

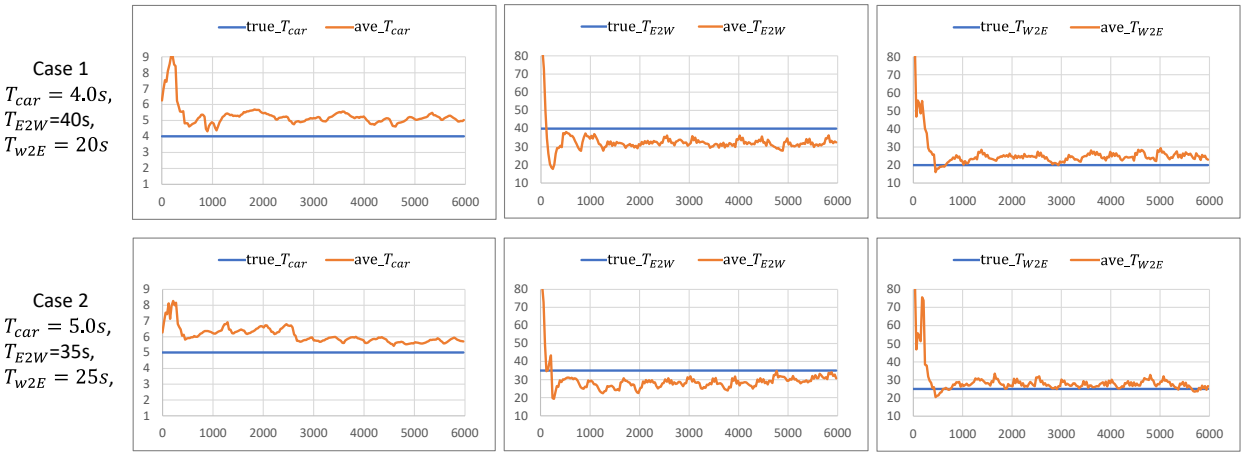


Figure 6: Parameter estimation when there exists combination effect of multiple parameters.

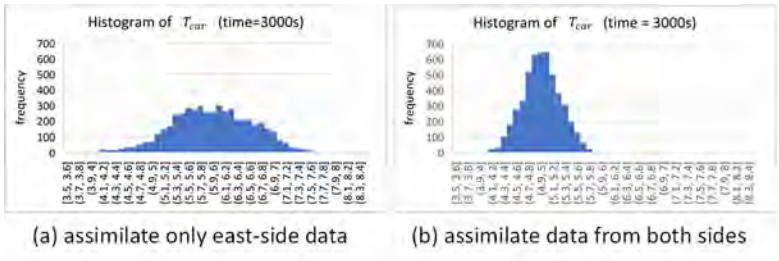


Figure 7: Histogram of particles' estimates of T_{car}.

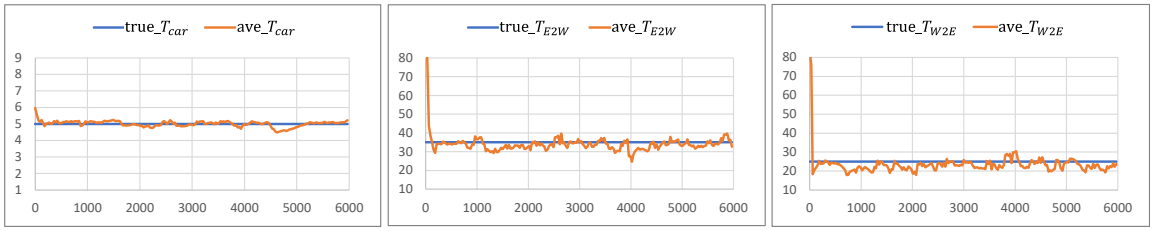


Figure 8: Parameter estimation by assimilating both the east- and west-side observation data.

As mentioned before, one way to address the combination effect of multiple parameters is to assimilate more observation data. To demonstrate this, we assimilate the observation data from both the east- and west-side observers. Figure 8 shows the estimation results for Case 2 (results for Case 1 is omitted to save space). Compare these results with the Case 2 results in Figure 6, one can see that the results have significantly improved. All the three parameters were able to quickly converge to the true values. Figure 7(b) shows the histogram of all the particles' estimates for T_{car} at step 100 when assimilating observation data from both sides of the observers. Compared to the histogram of Figure 7(a), the posterior distribution represented by the particles in Figure 7(b) is roughly a normal distribution with a much narrow span

(between 4.1 and 5.8). Furthermore, the mean of this normal distribution correctly lies at the true value of 5.0. This means the particles have correctly converged to the true parameter value.

5 CONCLUSION

This paper applies a particle filter-based data assimilation framework to online calibration of simulation digital twin models. Experiment results show that the data assimilation can effectively support online calibration of multiple model parameters using real-time observation data. The results also demonstrate the challenge associated with the combination effect of multiple parameters and its impact on data assimilation. This asks for rigorous approaches to handle the challenge of the combination effect of multiple parameters in future work. Other future works include more in-depth analysis of multi-parameter calibration under various situations, and development and evaluation of dynamic parameter calibration.

REFERENCES

Arulampalam, M.S., S. Maskell, N. Gordon, and T. Clapp. 2002. "A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking." *IEEE Transactions on Signal Processing* 50(2): 174–188.

Bai, F., S. Guo, and X. Hu. 2011. "Towards Parameter Estimation in Wildfire Spread Simulation Based on Sequential Monte Carlo Methods." In *Proceedings of the 44th Annual Simulation Symposium*, 159-166.

Doucet, A., N. Freitas, and N. Gordon. 2001. "An Introduction to Sequential Monte Carlo Methods." In *Sequential Monte Carlo Methods in Practice*, edited by A. Doucet, N. Freitas, and N. Gordon, 3–14. New York: Springer.

Fuller, A., Z. Fan, C. Day, and C. Barlow. 2020. "Digital Twin: Enabling Technologies, Challenges and Open Research." *IEEE Access* 8: 108952–108971.

Hu, X. 2022. "Data Assimilation for Simulation-Based Real-Time Prediction/Analysis." In *2022 Annual Modeling and Simulation Conference (ANNSIM)*, 404-415. IEEE.

Hu, X. 2023a. "A Tutorial on Bayesian Sequential Data Assimilation for Dynamic Data Driven Simulation." In *Proceedings of the 2023 Annual Modeling and Simulation Conference (ANNSIM)*, 680-695. IEEE.

Hu, X. 2023b. *Dynamic Data-Driven Simulation: Real-Time Data for Dynamic System Analysis and Prediction*. Singapore: World Scientific Publishing Co. Pte. Ltd.

Hu, X. and Yan, M. 2024. "Data Assimilation for Online Model Calibration in Discrete Event Simulation." *SIMULATION* 100(6): 529–544

Liang, S., X. Li, and X. Xie. 2013. *Land Surface Observation, Modeling and Data Assimilation*. New Jersey: World Scientific.

Liu, J. and M. West. 2001. "Combined Parameter and State Estimation in Simulation-Based Filtering." In *Sequential Monte Carlo Methods in Practice*, edited by A. Doucet, N. Freitas, and N. Gordon, 197–223. New York: Springer.

Malleson, N. 2014. "Calibration of Simulation Models." *Encyclopedia of Criminology & Criminal Justice* 40: 115-8.

Rykiel, E.J. 1996. "Testing Ecological Models: The Meaning of Validation." *Ecological Modelling* 90: 229–244.

Song, H., M. Song, and X. Liu. 2022. "Online Autonomous Calibration of Digital Twins Using Machine Learning with Application to Nuclear Power Plants." *Applied Energy* 326: 119995.

Titscher, T., T. van Dijk, D. Kadoke, A. Robens-Radermacher, R. Herrmann, and J. F. Unger. 2023. "Bayesian Model Calibration and Damage Detection for a Digital Twin of a Bridge Demonstrator." *Engineering Reports* 5(11): 1-27.

Ward, R., R. Choudhary, A. Gregory, M. Jans-Singh, and M. Girolami. 2021. "Continuous Calibration of a Digital Twin: Comparison of Particle Filter and Bayesian Calibration Approaches." *Data-Centric Engineering* 2: e15. doi:10.1017/dce.2021.12

Zeigler, B.P., T.G. Kim, and H. Prähofer. 2000. *Theory of Modeling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems*. 2nd ed. Amsterdam: Academic Press.

Zhang, K. 2020. "Real-Time Calibration of Large-Scale Traffic Simulators: Achieving Efficiency Through the Use of Analytical Mode." Ph.D. dissertation, Massachusetts Institute of Technology, Operations Research Center; Sloan School of Management.

AUTHOR BIOGRAPHIES

XIAOLIN HU is a full professor of the Computer Science Department at Georgia State University. He received his Ph.D. degree from the University of Arizona. His research interests include modeling and simulation theory and application, data assimilation, and dynamic data driven simulation. His email address is xhu@gsu.edu.

MINGXI YAN is a PhD candidate of the Computer Science Department at Georgia State University. Her research interests include modeling and simulation, and digital twin technology. Her email address is myan6@student.gsu.edu.