Strategic Use of Data Assimilation for Dynamic Data-Driven Simulation

Cho, Yubin; Huang, Yilin; Verbraeck, Alexander

DOI
10.1007/978-3-030-50433-5_3

Publication date
2020

Document Version
Final published version

Published in
Computational Science – ICCS 2020 - 20th International Conference, Proceedings

Strategic Use of Data Assimilation for Dynamic Data-Driven Simulation

Yubin Cho[1,2], Yilin Huang[1(✉)], and Alexander Verbraeck[1]

[1] Faculty of Technology, Policy and Management, Delft University of Technology, Delft, The Netherlands
{y.huang,a.verbraeck}@tudelft.nl
[2] To70 Aviation Consultants, The Hague, The Netherlands
yubin.cho@to70.nl

Abstract. Dynamic data-driven simulation (DDDS) incorporates real-time measurement data to improve simulation models during model run-time. Data assimilation (DA) methods aim to best approximate model states with imperfect measurements, where particle Filters (PFs) are commonly used with discrete-event simulations. In this paper, we study three critical conditions of DA using PFs: (1) the time interval of iterations, (2) the number of particles and (3) the level of actual and perceived measurement errors (or noises), and provide recommendations on how to strategically use data assimilation for DDDS considering these conditions. The results show that the estimation accuracy in DA is more constrained by the choice of time intervals than the number of particles. Good accuracy can be achieved without many particles if the time interval is sufficiently short. An over estimation of the level of measurement errors has advantages over an under estimation. Moreover, a slight over estimation has better estimation accuracy and is more responsive to system changes than an accurate perceived level of measurement errors.

Keywords: Dynamic Data-Driven Simulation · Data Assimilation · Particle Filters · Discrete-event simulation · Sensitivity analysis

1 Introduction

Simulation modeling has been widely used for studying complex systems [10–12]. In a highly evolving environment, classical simulation shows limitations in situational awareness and adaptation [8,9]. Dynamic Data-Driven Application Systems (DDDAS) is a relative new paradigm [4] proposed to integrate the computational and instrumental aspects of complex application systems offering more accurate measurements and predictions in real-time. A related concept is Dynamic Data-Driven Simulation (DDDS) [6,9], where Data Assimilation (DA) [3,14] is used to combine a numerical model with real-time measurements at simulation run-time. DA aims to obtain model states that best approximate the current and future states of a system with imperfect measurements [18].

V. V. Krzhizhanovskaya et al. (Eds.): ICCS 2020, LNCS 12142, pp. 31–44, 2020.
https://doi.org/10.1007/978-3-030-50433-5_3

Owing to disciplinary traditions, DA is predominantly used with simulation of continuous systems but less with discrete systems [7]. A few examples of the latter can be found e.g. in wildfire and transport simulations [5–7,26], and in agent-based simulations that predict the behavior of residents in buildings [21,22]. For DA in discrete systems simulations, the Sequential Monte Carlo (SMC) methods, a.k.a. Particle Filters (PFs), are commonly used [6,7,23,25]. Two major reasons are mentioned in literature. First, PFs methods are more suitable to DDDS than variational methods [15] since the models can easily incorporate the real-time data that arrives sequentially [23]. Second, the classical sequential methods such as Kalman Filter and its extensions rely on requirements that are difficult to fulfil by systems that exhibit non-linear and non-Gaussian behaviors which typically do not have analytical forms [7]. SMC or PFs are sample-based methods that use Bayesian inference, stochastic sampling and importance resampling to iteratively estimate system states from measurement data [7,23,25]. The probability distributions of interest are approximated using a large set of random samples, named particles, from which the outcomes are propagated over time [7,23,25].

In this paper, we study three common and critical conditions of DA using PFs for discrete-event simulation – the time interval of iterations, the number of particles and the level of measurement errors (or noises) – to understand the effect of these conditions on the estimation accuracy of system states. A number of works studied the conditions of DA for continuous systems such as meteorology, geophysics and oceanography [13,16,17,20]. But little is known for discrete-event simulation in this regard.

The time interval of assimilating measurement data and the number of particles in PFs are two critical conditions because they directly affect computational cost and estimation accuracy in DA. One recent research studied the effects of both conditions independently [24]. Our experiments also study their mutual influences, since they are two conditions that restrict one another given that the computational time is often limited between two successive iterations in DA. The level of measurement errors is another critical condition in DA. The actual level of measurement errors is rarely known in real world situations. What is included in DA algorithms is always the perceived level (or assumptions) of measurement errors. Our experimental setup imitates the actual level of measurement errors, and allows the study of the differences between the actual and perceived measurement errors, and their effects on estimation accuracy. In the following, we present the methodology used, discuss the experimental results and provide recommendations on future research.

2 Methodology

This research uses an $M/M/1$ single server queuing system with balking for the DA experiments. The real system is imitated with a sensing process that generates measurement data where errors (or noises) are introduced. The discrete-event simulation model is a perfect representation of the real system. The DA

process uses PFs to iteratively construct probability distributions for particle weight calculation incorporating measurement data. The DA results are evaluated with regard to different time intervals Δt, the numbers of particles N and the levels of actual and perceived measurement errors ϵ and ϵ'.

2.1 Experimental Setup

The experimental setup consists of four components (cf. [7,24]): (1) Real System, (2) Measurement Model, (3) Simulation Model, and (4) Data Assimilation. The real system and the simulation model are implemented with Salabim[1]. The whole experimental setup is implemented in python[2].

Real System. The real system is represented by an ESP32 microcontroller, which (1) imitates the real $M/M/1$ queuing system with balking, and (2) generates the "sensor data" in real-time.

The queuing process has exponential inter-arrival times of jobs (or customers) with mean arrival rate λ, and exponential service times with mean service (or processing) rate μ. The queue has a limit of length L for balking [1]: when the queue reaches length L, no new job is appended to the queue. The state of the queuing system S_{real} at time t is denoted as

$$S_{t,real} := \{arrRate_{t,real}, procRate_{t,real}, queLen_{t,real}\}$$

where $arrRate_{t,real}$ is the mean arrival rate λ, i.e. the inter-arrival time $T_{arr,real} \sim Exp(arrRate_{t,real})$; $procRate_{t,real}$ is the mean processing rate μ, i.e. the processing time $T_{proc,real} \sim Exp(procRate_{t,real})$; and $queLen_{t,real} \in [0, L]$ is the queue length.

To imitate second order dynamics [8] in the queuing system, every 15 s the values of $arrRate_{t,real}$ and $procRate_{t,real}$ are updated stochastically from a uniform distribution as

$$arrRate_{t,real} \sim U(0, 20)$$
$$procRate_{t,real} \sim U(0, 20)$$

These are the two internal values (i.e. non observables) the data assimilation component needs to estimate for the simulation model. Two "observables" are measured from the real system:

$$\{numArr_{real}, numDep_{real}\}$$

the number of arrival $numArr_{real}$ at the queue, and the number of departure $numDep_{real}$ from the queue during a measurement period. These two variables are added with noises and then used for DA.

[1] https://www.salabim.org.
[2] https://github.com/yuvenious/ddds_queuing.

Measurement Model. The "real system" sends sensor data (a set of two values each time) $\{numArr_{real}, numDep_{real}\}$ through serial communications, and generates measurement data:

$$\{numArr_{measure}, numDep_{measure}\}$$

The measurement data at time t is denoted as

$$numArr_{t,measure} = numArr_{t,real} + error_{t,arr}$$
$$numDep_{t,measure} = numDep_{t,real} + error_{t,dep}$$

where $error_{t,arr}$ and $error_{t,dep}$ are the imitated measurement errors (or noises), sampled from Gaussian distributions $N \sim (0, \sigma^2)$ at time t. The variance σ can take one of the four values denoted by $\epsilon \cdot \Delta t^2$, where ϵ is the level of measurement errors during the sensing process: $\epsilon \in [0, 3]$ represents the error levels from zero (0) to low (1), medium (2) till high (3). Δt is the time interval of DA. For example, if $\Delta t = 5$ then σ is set to be $[0, 5, 10, 15]$ in the experiments depending on the corresponding error levels. In addition, σ_{arr} and σ_{dep} are independent to each other in the experiments. As such, the joint probability can be obtained by the product of the two probabilities.

Note that in our experiments, the data assimilation process uses the perceived level of measurement errors ϵ' to represent the difference between the assumption of the level of measurement errors and their actual level. To our knowledge, these two are deemed as the same, i.e. $\epsilon = \epsilon'$, in previous works.

Simulation Model. The simulation model of the single server queuing system with balking has state $S_{t,sim}$ at time t denoted as

$$S_{t,sim} := \{arrRate_{t,sim},\ procRate_{t,sim},\ queLen_{t,sim}\}$$

where $arrRate_{t,sim}$ is the mean arrival rate; $procRate_{t,sim}$ is the mean processing rate; and $queLen_{t,real}$ is the queue length. In the simulation, the inter-arrival times and processing times also have exponential distributions, and the queue has maximum length L as in the "real system".

Each simulation replication is a particle in the DA. The state transition of a replication i (i.e. particle i) from time t to $t + \Delta t$ is denoted as

$$S^i_{t,sim} \longmapsto S^i_{t+\Delta t,sim}\ (i = 1, 2, \cdots, N)$$
$$S^i_{t,sim} := \{arrRate^i_{t,sim},\ procRate^i_{t,sim},\ queLen^i_{t,sim}\}$$

where N is the total number of particles. The simulation time is repeatedly advanced by time interval Δt, each time after the measurement data becomes available and when the calculations in the DA are completed. The measurement data is "compared with" the corresponding predicted values by the simulation model:

$$\{numArr_{sim}, numDep_{sim}\}$$

which are the number of arrival and the number of departure in the simulation.

Data Assimilation. At initialization ($t = 0$), N sets of mean arrival rates and mean processing rates are sampled from uniform distribution $U(0, 20)$ for the N particles in the simulation, and each particle has equal weight:

$$\{arrRate^i_{0,sim}, procRate^i_{0,sim}\}$$

The simulation time t of each particle i then advances by Δt denoted as

$$S^i_{0,sim} \longmapsto S^i_{0+\Delta t,sim}$$

Iteratively, the simulation time t advances by Δt, and each simulation (replication, i.e. particle i) $S^i_{t,sim} \longmapsto S^i_{t+\Delta t,sim}$ is interpreted as the predictive distribution $p(x^i_{t+\Delta t}|x^i_t)$ of state variable $x \in S_{sim}$.

The importance weight w^i of each particle i is calculated by comparing the measurement data with the simulation (prediction). Each particle i is equally weighted at initialization: $w^i_0 = 1/N$. For the subsequent iteration steps, weights are calculated as:

$$\begin{aligned} w^i_{t+\Delta t} &= p(measure_{t+\Delta t} \,|\, predict^i_{t+\Delta t}) \cdot w^i_t \quad \text{where} \\ measure_{t+\Delta t} &= \{numArr_{t+\Delta t,measure},\ numDep_{t+\Delta t,measure}\} \\ predict^i_{t+\Delta t} &= \{numArr^i_{t+\Delta t,sim},\ numDep^i_{t+\Delta t,sim}\} \end{aligned}$$

As mentioned earlier, the level of measurement errors ϵ is used to imitate the measurement noises, $error_{arr}$ and $error_{dep}$, that are added into the measurement data. A different value (i.e. the level of perceived measurement errors ϵ') is used for the weight calculation of each particle, comparing the measurement data, $measure_{t+\Delta t}$ (or $measure_t$), with the prediction by the simulation, $predict^i_{t+\Delta t}$ (or $predict^i_t$). The conditional probability of $measure_t$ given $predict^i_t$, is interpreted as the conditional probability of the difference between the two, $measure_t - predict^i_t$, given the level of perceived measurement errors ϵ' (cf. [23] p.47):

$$\begin{aligned} p(measure_t \,|\, predict^i_t) &= p(measure_t - predict^i_t \,|\, \epsilon') \\ &= \frac{1}{\sigma'\sqrt{2\pi}} \cdot e^{-\frac{\left(measure_t - predict^i_t\right)^2}{2\sigma'^2}} \\ \text{where } \sigma' &= \epsilon' \cdot \Delta t^2 \end{aligned}$$

In each iteration, $arrRate^i_{sim}$ and $depRate^i_{sim}$ of every particle i are resampled according to its weight w^i. This means a higher probability of resampling is given to a particle with a higher weight. As a result, the resampled particles are located nearby the highly weighted particles in the previous iteration.

For example, if the evaluated weight of particle i is $w^i_{t+\Delta t} = 0.6$ and $N = 1000$, then 600 new particles ($j = 1, 2, \cdots, 600$) are subjected to resampling derived from particle i. In principle, $S^i_{t+\Delta t,sim}$ is assigned to $S^j_{t+\Delta t,sim}$ as

$$S^j_{t+\Delta t,sim} \longleftarrow \{arrRate^i_{t+\Delta t,sim},\ procRate^i_{t+\Delta t,sim},\ queLen^i_{t+\Delta t,sim}\}$$

But since all these resampled particles contain the identical state, different random seeds shall be used to prevent identical simulation runs. We also use Gaussian distributions to scatter the values of $arrRate^i_{t,sim}$ and $depRate^i_{t,sim}$. This additional treatment guarantees that the resampled particle j is close but different to the previous particle i to represent the dynamic change of the system.

$$arrRate^j_{t+\Delta t,sim} \sim N(arrRate^i_{t,sim}, arrRate^i_{t,sim}/10)$$
$$depRate^j_{t+\Delta t,sim} \sim N(depRate^i_{t,sim}, depRate^i_{t,sim}/10)$$

Thereafter, all resampled particles are evenly weighted: $w^j_{t+\Delta t} = 1/N$. These resampled particles are used for the next iteration ($t \leftarrow t + \Delta t$).

The (aggregated) system state at time t can be estimated by the state of each particle and their corresponding weights as

$$S_{t,sim} = \frac{1}{N} \sum_{i}^{N} (S^i_{t,sim} \cdot w^i_t)$$

2.2 Sensitivity Analysis

In the experiments, three critical conditions in DA are investigated to study their effects on the estimation accuracy: (1) the time interval Δt, (2) the number of particles N, and (3) the level of measurement errors ϵ and the level of perceived measurement errors ϵ'. The time interval Δt determines the frequency of the DA steps, i.e. how often the measurement data is assimilated to the simulation which triggers the calculation of the subsequent predictive distributions. The number of particles N is the number of simulation replications used for the DA algorithm. It determines the "number of samples" used for the predictive distribution. The level of measurement errors ϵ is used to introduce noises in the measurement data, and the level of perceived measurement errors ϵ' is used in importance weight calculation. The experiments make combinations of the levels of actual and perceived measurement errors to study the effect.

Each DA experiment run lasts 50 s, during which $arrRate_{real}$ and $procRate_{real}$ change every 15 s in the "real system". The values of $numArr$ and $numDep$ are assimilated to the simulation model in the experiment using different time interval Δt which ranges from 1 to 5 s. The number of particles N for the DA varies from 10 to 2000. The measurement errors and perceived measurement errors are set to be different as will be further explained in the next section.

To compare the estimation accuracy of different DA experiment settings, distance correlation [2,19] is used to measure the association between the state variables of the "real system" and the simulated values:

$$0 \leq dCor(S_{real}, S_{sim}) = \frac{dCov(S_{real}, S_{sim})}{\sqrt{dVar(S_{real})dVar(S_{sim})}} \leq 1$$

$dCor$ is measured for each state variable. The overall distance correlation of the estimation is the mean of the individual distance correlations.

3 Experimental Results and Discussions

This section first presents the results regarding time interval and number of particles, as they produce related effects on computational cost and estimation accuracy. Since computational cost is often limited in practice, experiments are also made to show the trade-offs of the two. The second part of this section compares the effect of measurement errors with perceived measurement errors.

3.1 Time Interval and Number of Particles

The time interval Δt of iternation in DA is experimented ranging from 1 to 5 s. The number of particles N is set to be 1000 in those experiments ($\epsilon = 1$ and $\epsilon' = 1$). As shown in Fig. 1, when Δt decreases, the estimation accuracy $dCor$ increases significantly with narrower variances.

The number of particles N is experimented ranging from 10 to 2000 with different steps, as shown in Fig. 2, where $\Delta t = 1$, $\epsilon = 1$ and $\epsilon' = 1$. The estimation accuracy $dCor$ increases with narrower variances as more particles are used in the DA. However, when N exceeds 100, the increment in accuracy becomes slower. The Tuckey test (CI = 95%) is performed to compare the difference of $dCor$ between $N = 100$ and higher numbers of particles. The result shows that the increase in the number of particles above 400 in these experiments is no more effective in improving estimation accuracy.

Trade-Off Between Time Interval and Number of Particles. To understand the relation between the time interval Δt and number of particles N with regard to the estimation accuracy $dCor$, an extensive number of DA experiments are performed. The results are displayed in Fig. 3, where the X-axis shows the total number of simulation runs over one DA experiment. For example, if $\Delta t = 2$ s and $N = 1000$ in a DA experiment, then the number of total simulation runs within that experiment is $50/2 \cdot 1000 = 25000$. The Y-axis is the resulting $dCor$ of that experiment. Each dot in Fig. 3 hence represents one DA experiment, where the size of the dot (small to large) denotes the number of particles $N \in \{500, 1000, 1500, 2000\}$, and the color of the dot (blue to red) indicates the time interval $\Delta t \in \{1, 2, 3, 4, 5\}$ used in that DA experiment.

The result shows that when N increases (large dots) and Δt decreases (blue dots), thereby more simulation replications and iterations executed, the estimation accuracy improves and $dCor$ approaches to 1. Notably, there is hardly any red dots close to $dCor = 1$, and many large red dots (i.e. experiments with high numbers of particles and long time intervals) are located at where $dCor \leq 0.8$. This means, if Δt is too long, using a large number of particles increases computational cost *without* improvement in estimation accuracy. On the other hand, there are small blue dots (i.e. experiments with low numbers of particles and short time intervals) that are located close to $dCor = 1$. This indicates, if Δt is sufficiently short, good estimation accuracy can be achieved even though not many particles are used.

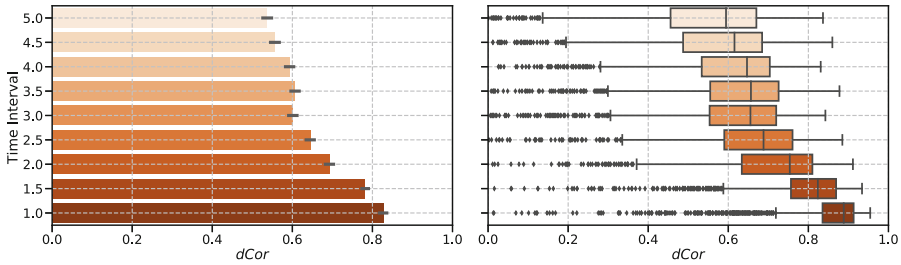


Fig. 1. Time interval Δt and estimation accuracy $dCor$ (N=1000)

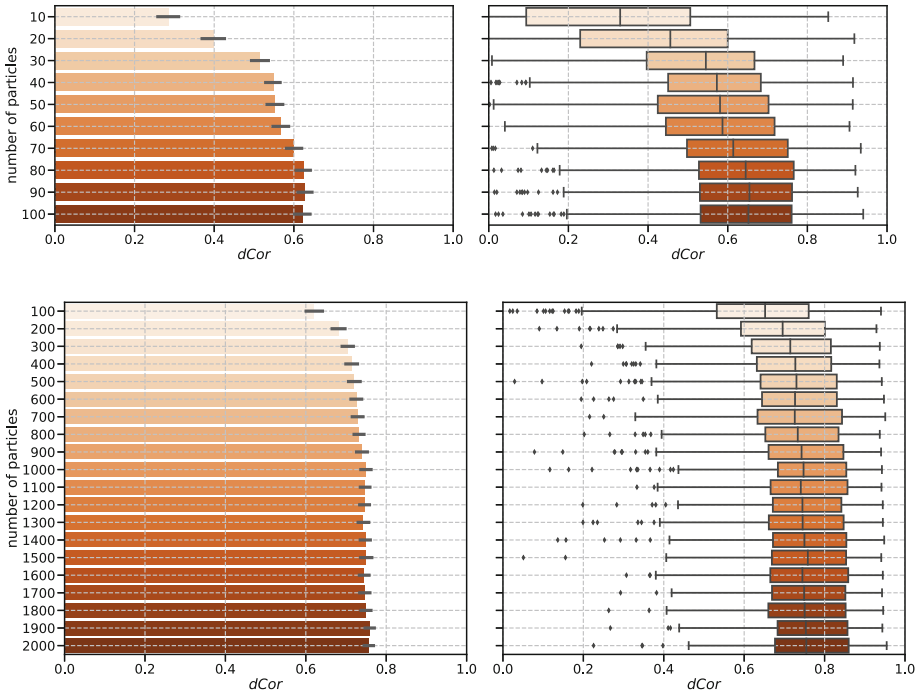
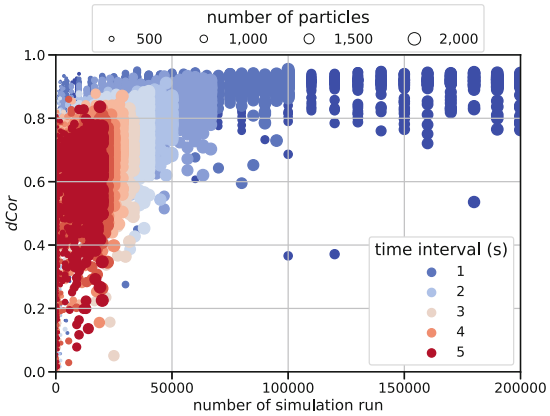
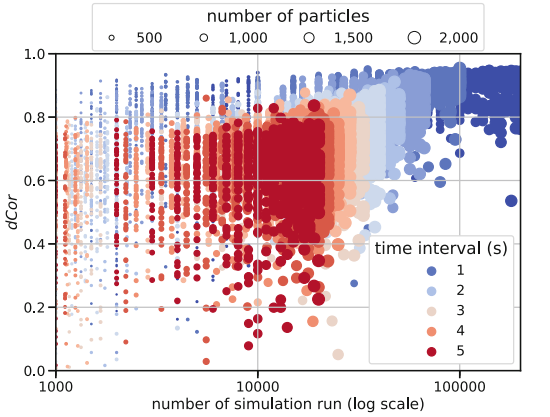


Fig. 2. Number of particles N and estimation accuracy $dCor$ (Δt=1)

To summarize the findings: while the number of particles is positively correlated and the time interval is negatively correlated to estimation accuracy in DA, the estimation accuracy is more constrained by the choice of time interval than the number of particles in the experiments. This implies that, given limited computational resources in DA applications, once the number of particles is sufficiently large, more computational resources can be allocated to shorten the time interval of iteration in DA to improve the estimation accuracy.



(a) Linear scale



(b) Log scale

Fig. 3. Trade-off between the time interval Δt and number of particles N (Color figure online)

3.2 Measurement Errors and Perceived Measurement Errors

In the experiments, the levels of measurement errors $\epsilon \in [0, 3]$ are from zero (0) to low (1), medium (2) till high (3). The levels of perceived measurement errors ϵ' are represented in a similar manner. Different levels of measurement errors $\epsilon \in [0, 3]$ are experimented first with perceived measurement errors $\epsilon' = 1$, $\Delta t = 1$ and $N = 400$. As shown in Fig. 4, when ϵ increases from zero to high, the estimation accuracy $dCor$ decreases with increasing variances.

The levels of perceived measurement errors $\epsilon' \in [1, 4]$ are experimented with $\epsilon = 1$, $\Delta t = 1$ and $N = 400$. Figure 5 shows that a higher level of perceived measurement errors in DA does not seem to generate a clear pattern in relation with $dCor$. The variances of $dCor$ have slight reduction, however.

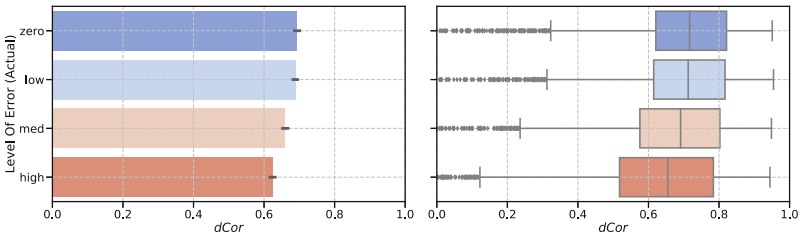


Fig. 4. Measurement errors ϵ and estimation accuracy $dCor$

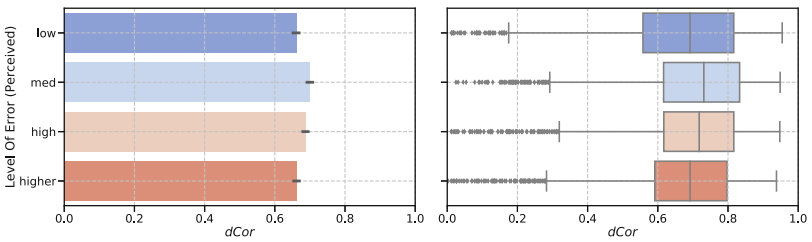


Fig. 5. Perceived measurement errors ϵ' and estimation accuracy $dCor$

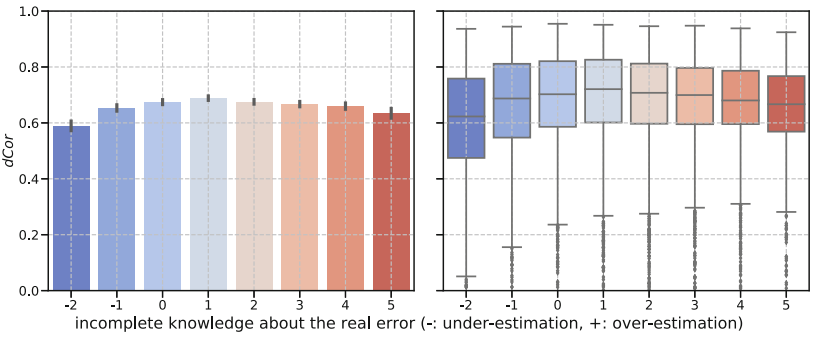
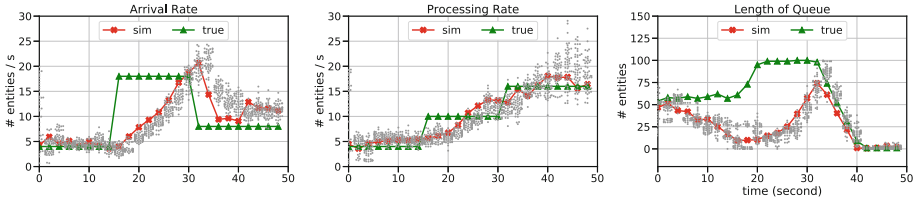


Fig. 6. Difference between perceived measurement errors and actual measurement errors $\epsilon' - \epsilon$ and estimation accuracy $dCor$

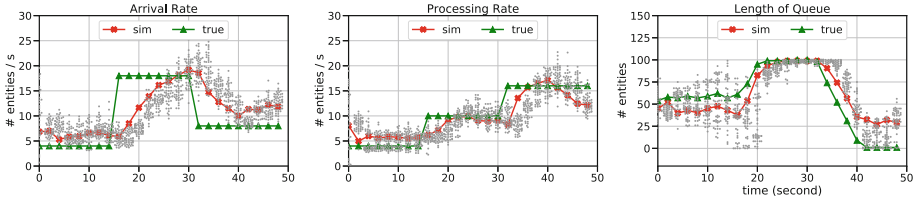
How does the difference between ϵ and ϵ' affect the estimation accuracy in DA? We further experiment this by sweeping $\epsilon \in \{0, 1, 2, 3\}$ and $\epsilon' \in \{1, 2, 3, 4, 5\}$ where $\Delta t = 1$ and $N = 400$. The results are shown in Fig. 6, where the X-axis shows the difference of perceived measurement errors and actual measurement errors by subtracting the value of the latter from the former, i.e. $x = \epsilon' - \epsilon$. For example, when the levels of measurement errors $\epsilon = 0$ and the levels of perceived measurement errors $\epsilon' \in \{1, 2, 3, 4, 5\}$, the results are plotted along $x \in \{1, 2, 3, 4, 5\}$; when $\epsilon = 3$ then the results are along $x \in \{-2, -1, 0, 1, 2\}$. This means, a negative x value indicates under estimation and a positive x indicates over estimation of the measurement errors.

The experimental results show that under estimation of the measurement errors ($x < 0$) leads to lower estimation accuracy $dCor$ in average, and over estimation ($x > 0$) often has higher $dCor$ than under estimation ($x < 0$). Perfect knowledge about measurement errors ($x = 0$) does not necessarily result in better $dCor$, while slight over estimation ($x = 1$) has better $dCor$ than perfect knowledge. In the cases when $x > 1$, $dCor$ gradually decreases again (see the slight right skew of the bars in Fig. 6) but it is no worse than the same levels of under estimation. In addition, $dCor$ has lower variances when over estimating the errors than under estimation, which is often a desired feature in DA.

To further illustrate the difference, we present and discuss another experiment that compares two cases: (a) perfect knowledge about measurement errors ($x = 0$); (b) slight over estimation of measurement errors ($x = 1$). The result is shown in Fig. 7. In both cases, the level of the actual measurement errors is Low ($\epsilon = 1, \Delta t = 2$ and $N = 1300$). The first case (a) has perceived measurement errors at level Low ($\epsilon' = 1$) while the second case (b) over estimates the measurement errors at level $Medium$ ($\epsilon' = 2$). These two cases perform distinctly in estimating the queue length $queLen_{sim}$ in the simulation responding to the sudden change of the arrival rate $arrRate_{real}$ and processing rate $procRate_{real}$ at time $t = 15$ in the "real system". In case (a), the simulation can not well follow the trajectory of $queLen$ already in the first 15 s ($t : 0 \rightarrow 15$). Once the sudden change occurs at $t = 15$, $queLen$ diverges more and can catch up the system state again after 10 iterations in DA. In case (b), the simulation can follow the sudden change more responsively.



(a) Accurate Estimation of Measurement Errors



(b) Over Estimation of Measurement Errors

Fig. 7. Accurate estimation (a) vs over estimation (b) of measurement errors

The difference in response time in the two cases can be explained by the spread of particles, which are depicted as gray dots in Fig. 7. Note that the

vertical spread of particles in case (a) is narrower than that in case (b). In case (a), only a few particles having a small deviation from the measurement can "survive" throughout the experiment. Particles are discarded when they are located far apart. Consequently, sudden and large changes in the system are not detected rapidly because of the restricted spread of particles. In case (b), as the particles spread wider, the aggregated result can quickly converge to the true value under sudden changes. Thus widespread particles are more tolerating and show more responsive estimation in detecting capricious system changes.

Given these observations in the experiments, we conclude that a pessimistic view on measurement errors has advantages over an optimistic view on measurement errors with respect to the resulting estimation accuracy in DA. In addition, a slight pessimistic view on measurement errors results in better estimation accuracy than an accurate view on measurement errors in the experiments. (This is rarely an intuitive choice in DA experimental setup.)

4 Conclusions and Future Work

The experiments presented in this paper study the effect of experimental conditions – namely the time interval of iterations, the number of particles and the level of measurement errors (or noises) – of data assimilation (DA) on estimation accuracy using an $M/M/1$ queuing system (which is implemented in discrete event simulation). The simulation model is constructed with perfect knowledge about the internal process of the system. The choice of a simple target system and its model have the advantages that thorough experiments can be performed with a high number of iterations and particles, and the states of the real system and the simulated system can be easily compared. In addition, the experimental results of the difference in estimation accuracy (or inaccuracy) are direct consequences of the experimental conditions but not (partly) due to model noises since the model is "perfect". The results of the experiments can thus be interpreted in relative terms contrasting different experimental setups. The main findings in the experiments are as follows.

The time interval, i.e. the inverse of the frequency of iterations, in DA has a negative correlation with the estimation accuracy of system states. More frequent assimilation of real-time measurement data is effective to improve the estimation accuracy and the confidence level of the estimation. Although the number of particles has in general a positive correlation with the estimation accuracy, increasing the number of particles is ineffective in improving estimation accuracy beyond a certain level. Notably, good estimation accuracy can be achieved even though not many particles are used if the time interval is short. Since both decreasing the time interval and increasing the particles require more computation, the former can be more cost effective when the number of particles is sufficiently large. With regard to measurement errors, an over estimation of the level of measurement errors leads to higher estimation accuracy than an under estimation in our experiments. A slight over estimation has better estimation accuracy and more responsive model adaptation to system states than

an accurate estimation of measurement errors. An overly pessimistic view on measurement errors, however, deteriorates the estimation accuracy.

In this paper, the assimilation of real-time data to the simulation model is performed with fixed time intervals during an experiment run. An event based data assimilation approach and its effects can be an interesting future research direction. The experimental setups could also be dynamically configured during DA in real-time to achieve good estimation results.

References

1. Ancker, C., Gafarian, A.: Some queuing problems with balking and reneging. i. Oper. Res. **11**(1), 88–100 (1963)
2. Bickel, P.J., Xu, Y.: Discussion of brownian distance covariance. Ann. Appl. Stat. **3**(4), 1266–1269 (2009)
3. Bouttier, F., Courtier, P.: Data assimilation concepts and methods. ECMWF (European Centre for Medium-Range Weather Forecasts) (2002)
4. Darema, F.: Dynamic data driven applications systems: a new paradigm for application simulations and measurements. In: Bubak, M., van Albada, G.D., Sloot, P.M.A., Dongarra, J. (eds.) ICCS 2004. LNCS, vol. 3038, pp. 662–669. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24688-6_86
5. Gu, F.: On-demand data assimilation of large-scale spatial temporal systems using sequential monte carlo methods. Simulation Modell. Pract. Theory **85**, 1–14 (2018)
6. Hu, X.: Dynamic data driven simulation. SCS M&S Magazine **5**, 16–22 (2011)
7. Hu, X., Wu, P.: A data assimilation framework for discrete event simulations. ACM Trans. Model. Comput. Simul. **29**(3), 171–1726 (2019). https://doi.org/10.1145/3301502
8. Huang, Y., Seck, M.D., Verbraeck, A.: Towards automated model calibration and validation in rail transit simulation. In: Sloota, P.M.A., van Albada, G.D., Dongarrab, J. (eds.) Proceedings of The 2010 International Conference on Computational Science. Procedia Computer Science, vol. 1, pp. 1253–1259. Elsevier, Amsterdam (2010)
9. Huang, Y., Verbraeck, A.: A dynamic data-driven approach for rail transport system simulation. In: Rossetti, M.D., Hill, R.R., Johansson, B., Dunkin, A., Ingalls, R.G. (eds.) Proceedings of The 2009 Winter Simulation Conference, pp. 2553–2562. IEEE, Austin (2009)
10. Huang, Y., Seck, M.D., Verbraeck, A.: Component based light-rail modeling in discrete event systems specification (DEVS). Simulation **91**(12), 1027–1051 (2015)
11. Huang, Y., Verbraeck, A., Seck, M.D.: Graph transformation based simulation model generation. J. Simul. **10**(4), 283–309 (2016)
12. Huang, Y., Warnier, M., Brazier, F., Miorandi, D.: Social networking for smart grid users - a preliminary modeling and simulation study. In: Proceedings of 2015 IEEE 12th International Conference on Networking, Sensing and Control, pp. 438–443 (2015). DOI: https://doi.org/10.1109/ICNSC.2015.7116077
13. Ma, C., et al.: Multiconstituent data assimilation with WRF-Chem/DART: Potential for adjusting anthropogenic emissions and improving air quality forecasts over eastern China. J. Geophys. Res.: Atmospheres **124**, 7393–7412 (2019). https://doi.org/10.1029/2019JD030421

14. Nichols, N.: Data assimilation: aims and basic concepts. In: Swinbank, R., Shutyaev, V., Lahoz, W.A. (eds.) Data Assimilation for the Earth System, pp. 9–20. Springer, Dordrecht (2003). https://doi.org/10.1007/978-94-010-0029-1_2
15. Petropoulos, G.P.: Remote Sensing of Surface Turbulent Energy Fluxes, chap. 3, pp. 49–84. CRC Press, Boca Raton (2008)
16. Ren, L., Nash, S., Hartnett, M.: Data assimilation with high-frequency (HF) radar surface currents at a marine renewable energy test site. C. Guedes Soares (Leiden: CRC Press/Balkema) pp. 189–193 (2015)
17. Shuwen, Z., Haorui, L., Weidong, Z., Chongjian, Q., Xin, L.: Estimating the soil moisture profile by assimilating near-surface observations with the ensemble kaiman filter (ENKF). Adv. Atmosph. Sci. **22**(6), 936–945 (2005)
18. Smith, P., Baines, M., Dance, S., Nichols, N., Scott, T.: Data assimilation for parameter estimation with application to a simple morphodynamic model. Math. Rep. **2**, 2008 (2008)
19. Székely, G.J., Rizzo, M.L., Bakirov, N.K., et al.: Measuring and testing dependence by correlation of distances. Ann. Stat. **35**(6), 2769–2794 (2007)
20. Tran, A.P., Vanclooster, M., Lambot, S.: Improving soil moisture profile reconstruction from ground-penetrating radar data: a maximum likelihood ensemble filter approach. Hydrol. Earth Syst. Sci. **17**(7), 2543–2556 (2013)
21. Wang, M., Hu, X.: Data assimilation in agent based simulation of smart environment. In: Proceedings of the 1st ACM SIGSIM Conference on Principles of Advanced Discrete Simulation, pp. 379–384. ACM (2013)
22. Wang, M., Hu, X.: Data assimilation in agent based simulation of smart environments using particle filters. Simulation Modell. Pract. Theory **56**, 36–54 (2015)
23. Xie, X.: Data assimilation in discrete event simulations. Ph.D. thesis, Delft University of Technology (2018)
24. Xie, X., van Lint, H., Verbraeck, A.: A generic data assimilation framework for vehicle trajectory reconstruction on signalized urban arterials using particle filters. Transport. Res. Part C: Emerg. Technol. **92**, 364–391 (2018)
25. Xie, X., Verbraeck, A., Gu, F.: Data assimilation in discrete event simulations: a rollback based sequential monte carlo approach. In: Proceedings of the Symposium on Theory of Modeling & Simulation, p. 11. Society for Computer Simulation International (2016)
26. Xue, H., Gu, F., Hu, X.: Data assimilation using sequential monte carlo methods in wildfire spread simulation. ACM Trans. Model. Comput. Simulation (TOMACS) **22**(4), 23 (2012)