Proceedings of the 2025 Winter Simulation Conference
E. Azar, A. Djanatliev, A. Harper, C. Kogler, V. Ramamohan, A. Anagnostou, and S. J. E. Taylor, eds.

MONTE CARLO DIGITAL TWIN WORLDVIEW: A PROPOSAL

Susan R. Hunter[1], Raghu Pasupathy[2], and Bruce W. Schmeiser[1]

[1]Edwardson School of Industrial Engineering, Purdue University, West Lafayette, IN, USA
[2]Department of Statistics, Purdue University, West Lafayette, IN, USA

ABSTRACT

We propose a worldview to facilitate the discussion of how Monte Carlo simulation experiments are conducted in a digital twin context. We use the term *Monte Carlo digital twin* to refer to a digital twin in which a Monte Carlo simulation model mimics the physical twin. The worldview we propose encompasses Nelson and Schmeiser's classical worldview for simulation experiments and adds key components of a Monte Carlo digital twin including a predictor, a decision maker, and bidirectional interaction between the Monte Carlo digital twin and the physical twin. We delineate the proposed worldview components in three examples.

1 INTRODUCTION

The use of Monte Carlo simulation models in a digital twin framework has become a popular topic at the Winter Simulation Conference, with an established dedicated track, "Simulation as Digital Twin." In the 2024 Winter Simulation Conference, Onggo and Currie (2024) discuss the idea that the evolution of simulation models from a mostly offline setting to an increasingly online setting implies that some traditionally offline simulation concepts, such as input modeling, require updates for the online setting. Building on the idea of updating traditional simulation concepts, in this paper we propose an expansion of Nelson and Schmeiser's Monte Carlo simulation worldview (discussed in Section 2) to facilitate the discussion of how Monte Carlo simulation is used in a digital twin context. While diagrams indicating the integration of simulation with digital twins have appeared before (e.g., those in Biller et al. (2022) and Zhou (2024)), to the best of our knowledge, this is the first appearance of a detailed worldview at the Winter Simulation Conference.

The concept of a digital twin does not require the use of Monte Carlo simulation per se — the concept, as defined in the report by the National Academies of Sciences, Engineering, and Medicine (NASEM) (National Academies of Sciences, Engineering, and Medicine 2023, p. 21), is broader; we refer to this definition as the *NASEM definition*:

> A digital twin is a set of virtual information constructs that mimics the structure, context, and behavior of a natural, engineered, or social system (or system-of-systems), is dynamically updated with data from its physical twin, has a predictive capability, and informs decisions that realize value. The bidirectional interaction between the virtual and the physical is central to the digital twin.

The NASEM report that contains this definition notes that the definition of a digital twin may vary by sub-discipline, and also quotes the following definition as being one of the origins of the term *digital twin*, attributed to a NASA report by Shafto et al. (2012):

> A digital twin is an integrated multiphysics, multi-scale, probabilistic simulation of a vehicle or system that uses the best available physical models, sensor updates, fleet history, etc., to

mirror the life of its flying twin. The digital twin is ultra-realistic and may consider one or more important and interdependent vehicle systems, including propulsion and energy storage, life support, avionics, thermal protection, etc.

For completeness, we also asked OpenAI ChatGPT (2025) for a definition of a digital twin and obtained the following in response (with citations corrected due to minor hallucinations in the original transcript and notation changed to avoid conflicts with later definitions):

A *digital twin* is a data-driven, model-integrated computational artifact designed to represent, synchronize with, and support decision-making about a physical system. Formally, a digital twin is a triplet $(\mathbb{M},\mathbb{D},\mathbb{C})$, where:

- $\mathbb{M}$ is a *model component*: a parametrized, simulatable representation of the physical system's state and evolution. The model may be physics-based, data-driven, or hybrid, and it must capture the system's behavior.
- $\mathbb{D}$ is a *data pipeline*, responsible for streaming observational or operational data from the physical system to the model. This enables state estimation, model calibration, and anomaly detection (Rasheed et al. 2020).
- $\mathbb{C}$ is a *coupling mechanism* that provides bi-directional interaction between physical and digital layers. It supports monitoring, control, and experimentation (Fuller et al. 2020).

Together, the triplet $(\mathbb{M},\mathbb{D},\mathbb{C})$ encapsulates the defining features of a digital twin: high-fidelity modeling, real-time synchronization, and actionable decision integration. This formalism generalizes the original vision of Grieves and Vickers (2017) and aligns with scalable architectures such as those described by Kapteyn et al. (2021).

Essential properties include fidelity, synchrony, adaptivity, and interactivity. These properties distinguish a digital twin from static or offline models and enable its use in dynamic, data-rich, operational settings.

While the descriptions above identify important components, especially the digital twin as defined by OpenAI's ChatGPT, our interest is a concise worldview with further precision about components and their relationships, so as to facilitate thought and meaningful discussion. We consider only digital twins in which "the set of virtual constructs that mimics the structure, context, and behavior of a system" per the NASEM definition, and the model component $\mathbb{M}$ per the ChatGPT definition, is a Monte Carlo simulation model. Thus, the system can be characterized by probability models (Asmussen and Glynn 2007; Nelson and Pei 2021), and its performance can be assessed using simulation experiments. We refer to such a setting as a *Monte Carlo digital twin*.

In what follows, we first recall Nelson and Schmeiser's simulation worldview in Section 2. Then, we introduce the proposed Monte Carlo digital twin worldview in Section 3. Section 4 contains examples, deliberately simplified, that describe the elements of the worldview in context. Section 5 contains concluding remarks.

2 NELSON AND SCHMEISER'S SIMULATION WORLDVIEW

Nelson and Schmeiser's simulation worldview concisely encapsulates the idea that the purpose of a simulation experiment is to analyze a system's performance by using Monte Carlo to estimate an unknown performance measure θ. This simulation worldview has conceptual origins in Barry Nelson's 1983 thesis (Nelson 1983). In particular, the top of p. 20 in Nelson (1983) discusses inputs, outputs, and estimators. The third chapter, "Simulation," develops the worldview in great detail, but without using the diagram or the word "worldview." Further discussion appears in the Winter Simulation Conference papers Nelson and Schmeiser (1983) and Schmeiser (1984).

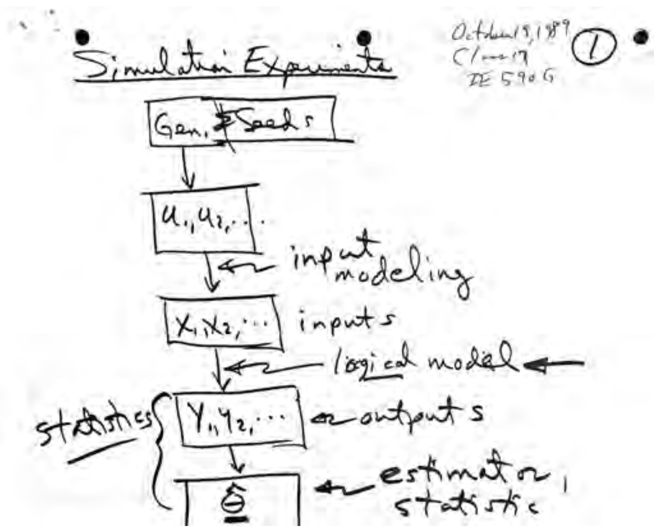

Figure 1: A scan of the first known appearance of Nelson and Schmeiser's simulation worldview in diagram form, as part of Schmeiser's IE 590G course notes at Purdue University from Fall 1989. IE 590G later obtained the permanent course number and title "IE 581 Simulation Design and Analysis."

The simulation worldview seems to have first appeared in diagram form as part of Schmeiser's class notes from October 1989; see Figure 1. A similar diagram appears in Schmeiser (1999), where the simulation experiment is written as the sequence of transformations

$$G \to U \to X \to Y \to \hat{\theta}. \tag{1}$$

Here, G represents the random number generator, U represents the random numbers, X represents the input random variables or data, Y represents the output random variables or data, and $\hat{\theta}$ represents the point estimator of the (not necessarily scalar) performance measure θ, e.g., a mean, quantile, or other properties of the distribution of the output random variables. The logic model and the probabilistic input model, which are annotated in Figure 1 but not explicitly shown in (1), are crucial for a complete description of a simulation experiment. The logic model transforms the input data X into the output data Y, and the probabilistic input model transforms the random numbers U into the input data X. We adopt the version of Nelson and Schmeiser's simulation worldview shown in Figure 2, which explicitly includes the input model, the logic model, and θ, where θ is labeled generally as an unknown "desired parameter."

Figure 2 provides a scaffolding for discussing the components of a Monte Carlo simulation model. Since its first appearance, the simulation worldview has been used by the authors to organize content and discussion in their simulation courses: first in Schmeiser's IE 581 at Purdue as Figure 1, in Pasupathy's ISE 5424 at Virginia Tech as Figure 2, and also in Hunter's IE 581 at Purdue as Figure 2. Further, an iteration on (1) appears in Larry Leemis's course notes for IE 680 at William and Mary (Leemis 2025,

$$\begin{tikzcd}
G \arrow[r] & U \arrow[r] & \begin{array}{c}\text{input}\\\text{model}\end{array} \arrow[r] & X \arrow[r] & \begin{array}{c}\text{logic}\\\text{model}\end{array} \arrow[r] & Y \arrow[r] & \hat{\theta} \arrow[r, squiggly] & \theta \\
\text{generator} & \begin{array}{c}\text{random}\\\text{numbers}\end{array} & & \begin{array}{c}\text{input}\\\text{variables}\end{array} & & \begin{array}{c}\text{output}\\\text{variables}\end{array} & \text{estimator} & \begin{array}{c}\text{desired}\\\text{parameter}\end{array}
\end{tikzcd}$$

Figure 2: Nelson and Schmeiser's worldview for discussing Monte Carlo simulation models, modified by Pasupathy to include the desired parameter θ. Every straight arrow in the diagram means "as an input to," and the squiggly arrow means "is an estimator for."

p. 3); in personal communication, Leemis also describes using the simulation worldview as a scaffolding for discussion.

3 A MONTE CARLO DIGITAL TWIN WORLDVIEW

As with Nelson and Schmeiser's simulation worldview, our aim is to create a Monte Carlo digital twin worldview that provides a scaffolding for discussion. The proposed worldview in Figure 3 expands upon Figure 2 to explicitly delineate between the processes that occur in real time in the physical world and those that occur in digital time in the virtual world. In particular, Figure 3 includes a physical twin, which can be updated by a decision maker and which produces data that is sent to the Monte Carlo digital twin. The Monte Carlo digital twin encompasses the simulation worldview and adds the *predictor*. The predictor receives data in the form of the physical twin decision variable values $\tilde{x}$ and other information $\tilde{\mathcal{D}}$, as well as the response $\tilde{Y}$ from the physical twin. The predictor also controls the Monte Carlo simulation model by initiating updates to the logic model and input model using decision variables x and data $\mathcal{D}$, which are analogous to the corresponding objects $\tilde{x}$ and $\tilde{\mathcal{D}}$ from the physical twin, and then sending seeds to the generator in service of observing the resulting estimator $\hat{\theta}$ from the Monte Carlo simulation model. The estimator $\hat{\theta}$ is constructed from output variables Y which are analogous to the response $\tilde{Y}$ in the physical twin. The predictor processes the information available from the physical twin and the information it collects from the Monte Carlo simulation model to produce an estimator $(\hat{x}^*, \hat{\theta}^*)$ for the desired prediction, which is conveyed to the decision maker. The choice of the term *predictor*, and the related terms *desired prediction* and *estimated prediction*, are consistent with the NASEM definition's requirement that the digital twin has a predictive capability. We acknowledge that our nomenclature is somewhat arbitrary; other terms

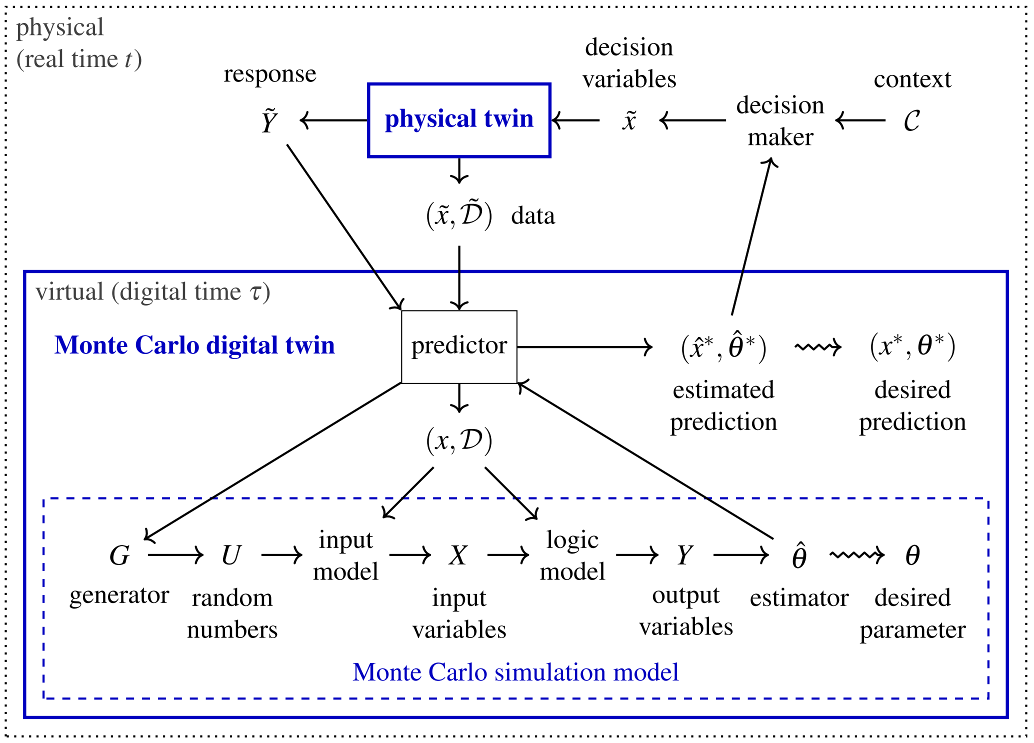

Figure 3: The proposed Monte Carlo digital twin worldview, where straight arrows mean "as an input to," and squiggly arrows mean "is an estimator for."

for the entity currently called *predictor* include *analyzer*, *controller*, *coupler* (from the ChatGPT definition), *optimizer*, *searcher*, and *seeker*. Another term for *desired prediction* would be *ideal performance*.

From Figure 3, three features explicitly or implicitly characterize Monte Carlo digital twins:

1. **Decision Variables**: Decision variables $\tilde{x}$ form the mechanism by which a *decision maker* alters the *physical twin*, based on advice from the predictor and other contextual considerations. Decision variables $\tilde{x}$ have their counterpart x in the digital twin.
2. **Physical-Virtual Feedback Loop**: The physical twin outputs the current state of the decision variables $\tilde{x}$ and data $\tilde{\mathcal{D}}$ to the predictor, as well as the response from the physical twin $\tilde{Y}$; $(\tilde{x},\tilde{\mathcal{D}})$ and $\tilde{Y}$ may be available at the same or different real times t. The predictor uses this information and its ability to control the generator to construct the estimated prediction $(\hat{x}^*,\hat{\theta}^*)$, where $\hat{x}^*$ is the estimated decision variable value which produces the estimated predicted performance for the physical twin, $\hat{\theta}^*$. The decision maker closes the loop by combining information from the predictor with contextual information $\mathcal{C}$ to update the physical twin.
3. **Predictor-Simulation Feedback Loop**: The predictor updates the input model and the logic model of the simulation using virtual decision variable and data values $(x,\mathcal{D})$. The predictor then uses the updated simulation model to perform simulation experiments by determining the number of simulation replications to conduct at the given value of x, sending the relevant seeds to the generator, and possibly also varying the values of the virtual decision variables x to perform "what-if" analyses. The ultimate goal of the predictor's simulation experiments is to construct the estimated predictions $(\hat{x}^*,\hat{\theta}^*)$, which are part of the physical-virtual feedback loop.

To keep the figure relatively simple, time is implicit in each loop of Figure 3. Specifically, there are two inherent clocks corresponding to physical time t and virtual time τ. Each element of the figure is a function of one clock, with the exception of the predictor which interacts with both clocks. There are also two implicit decision rules, which occur in physical time t: (i) How often does the physical twin send data to the predictor? and (ii) How much time is given to the predictor to simulate and explore different values of the virtual decision variables x? Although implicit in Figure 3, time arises explicitly in our examples in Section 4. Finally, Figure 3 implicitly assumes that the clock in the Monte Carlo simulation model runs (much) faster than real time.

The worldview in Figure 3 reflects only the relationships between the results of many modeling processes and choices, such as choice of random number generator, input modeling, output analysis, variance reduction, verification, validation, and recalibration. A significant amount of logic and complication can be embedded within any component or arrow in Figure 3. For example, the predictor requires a variety of coordinating and predictive capabilities which are not explicitly detailed. Similarly, each arrow from $(x,\mathcal{D})$ to the input model and the logic model requires transforming the data into a format which enables an update to the Monte Carlo simulation model. Further logic may also be required to recalibrate a Monte Carlo digital twin; see Rhodes-Leader and Nelson (2023) and He et al. (2024).

The proposed Monte Carlo digital twin worldview implies that errors in decision-making arise (only) due to three sources of error: (i) modeling error, due to deviations in the representation of the physical twin by the digital twin; (ii) estimation error, due to the deviation of the Monte Carlo estimator $\hat{\theta}$ from the desired parameter θ; and (iii) prediction error, due to the deviation of estimated prediction $(\hat{x}^*,\hat{\theta}^*)$ from the desired prediction (x^*,θ^*). Whereas the modeling error arises as a result of modeling approximations and coding errors, the estimation error and the prediction error arises from simulation uncertainty, e.g., not performing enough Monte Carlo replications, and numerical uncertainty, e.g., not solving an optimization problem to infinite precision, respectively.

A Monte Carlo digital twin worldview should have two properties: (i) a given real-world system could have many digital twin instantiations encompassed by the worldview, and (ii) given a digital twin instantiation, we (anyone) should be able to identify the various worldview components. We think Figure 3 attains these two properties.

4 EXAMPLES

To illustrate the proposed Monte Carlo digital twin worldview in context, we now present three examples. The first example is an M/M/$\tilde{x}$ queueing system with the number of servers $\tilde{x}$ as the decision variable. The second example is a newsvendor problem with restocking and time-dependent demand. To illustrate the worldview beyond the context of optimization, the third example considers anomaly detection in a water transmission network. In each of the examples, we describe the components of the Monte Carlo digital twin worldview depicted in Figure 3. Keep in mind the following two points:

1. Our examples intentionally increase in complexity, with the first example being trivial. All three examples are simpler than a typical messy real-world application.
2. Monte Carlo is one of three ways to "do" probability; the other two are closed-form analytics and numerical analyses. One user might find a closed-form or numerical solution while another might use Monte Carlo. Within the context of Monte Carlo, the model can range from "crude" to more-efficient, via any variance-reduction techniques.

4.1 Queue with Variable Number of Servers

Let the *physical twin* be a literal M/M/$\tilde{x}$ queuing system, with a single First-In First-Out queue, known constant arrival rate $\lambda > 0$, and known equal service rates $\mu > 0$. The number of servers, $\tilde{x} \in \{0,1,2,...,10\}$, is the single *decision variable*, which is updated in the physical system every five minutes. The *response* $\tilde{Y}$ is the random vector of physical wait times, measured as the time taken from when a customer enters the queue to starting service, and observed as $\tilde{y}$; $\tilde{\mathcal{D}}$ contains the current queue length $\tilde{Q}$ and elapsed service times for customers currently in service, $\tilde{S}_1, \tilde{S}_2, \ldots, \tilde{S}_{\tilde{x}}$. For convenience, define the elapsed service time of an idle or non-existent server to be zero.

Let W be the wait time of a random customer arriving to the queue during the next five minutes. For x servers, the performance measure is the *desired parameter*

$$\theta(x) := \Pr\{W \leq 3\,\text{minutes}\,|\,x, \tilde{Q} = \tilde{q}, \tilde{S}_1 = \tilde{s}_1, \tilde{S}_2 = \tilde{s}_2, ..., \tilde{S}_{\tilde{x}} = \tilde{s}_{\tilde{x}}\}.$$

The optimal number of servers, x^*, is the minimal value of x so that $\theta \geq 0.95$, or $x = 10$ if the threshold cannot be met. (Yes, the authors know that conditioning on the elapsed services times is unnecessary, because the service times are exponential. Our user seems not to know.)

Ignoring that a sophisticated user can easily find more efficient approaches, our user's *digital twin* is a crude Monte Carlo simulation model that perfectly reflects the physical twin. At the beginning of every physical five-minute interval, the *predictor* is awakened by the physical twin with its wait times $\tilde{y}$, its current number of servers $\tilde{x}$, its queue length $\tilde{q}$, and its current elapsed service times $\tilde{s}_1, \tilde{s}_2, \ldots, \tilde{s}_{\tilde{x}}$.

The predictor estimates the next five minutes' optimal number of servers x^* using the digital twin. For each feasible number of servers x, and for each of 1000 independent and identically distributed replications, the predictor runs the Monte Carlo model, with the initial state being the current physical queue length $\tilde{q}$ and elapsed service times $\tilde{s}_1, \tilde{s}_2, \ldots, \tilde{s}_x$; the run lengths are long enough to determine the wait time for each customer arriving during the digital five-minute interval. For each x, the estimator $\hat{\theta}(x)$ is the fraction of digital customers that waited no more than three minutes. The predictor estimates the optimal number of servers x^* and performance at the optimal number of servers $\theta^* = \theta(x^*)$ by assuming that $\hat{\theta}(x) = \theta(x)$; that is, it sets $\hat{x}^*$ to the smallest value x for which at least 95% of digital customers waited no more than three minutes (or to $\hat{x}^* = 10$ if no x values achieve this goal), and sets the estimated predicted performance to $\hat{\theta}^* = \hat{\theta}(\hat{x}^*)$.

The predictor passes $(\hat{x}^*, \hat{\theta}^*)$ to the *decision maker*. Having an idealized *context* $\mathcal{C}$, the decision maker simply sets the number of physical servers, $\tilde{x}$, to the predictor's estimated optimal number of servers; that is, $\tilde{x} = \hat{x}^*$. Contextual concerns such as timing, costs, illness, and server maintenance are left to real-world examples. Also due to this example's simplicity, notice that the predictor had no use for knowing the previous time interval's wait-times response $\tilde{y}$.

4.2 Newsvendor with Restocking and Time-Dependent Demand

Consider a *physical twin* consisting of a periodically-restocked 24-hour newsstand selling newspapers to arriving customers. That is, the newsstand is replenished with $\tilde{x}_{i-1} \geq 0$ additional newspapers at the beginning of the ith time increment $[(i-1)\delta, i\delta)$, $i \in \{1,2,\ldots\}$ for some $\delta > 0$, say every three hours starting from 5am. (While we consider only a constant restocking time interval δ below, a more complicated model could allow random or event-based restocking.) Customers arrive and purchase an available newspaper at price p dollars per unit, where the total number of customers in increment i is denoted by the demand random variable D_i. If no newspaper is available, the customer leaves the newsstand with no purchase. At the end of each m time increments, which we refer to as a news cycle, newspapers obtained at cost c dollars per unit expire and are sold at a salvage price of s dollars per unit. For example, a newsstand may be restocked by adding newspapers every three hours at 5am, 8am, 11am, and so on, but after $m = 8$ time increments, all unsold newspapers are considered expired and sold for salvage at price s; see Figure 4. Fresh newspapers are stocked for the next news cycle.

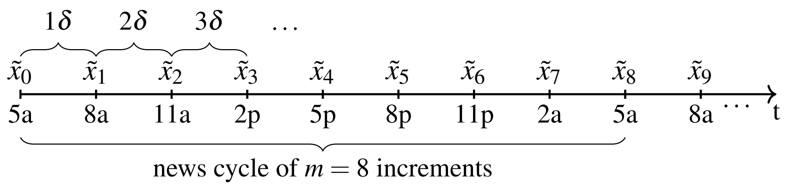

Figure 4: A timeline for the newsvendor with restocking and time-dependent demand.

The *decision maker* is the entity with the power to place orders and make changes to the newsstand (physical twin), such as a human newsstand manager or an algorithmic controller. Thus, the decision maker determines the *decision variable*, which is the number of additional newspapers to stock $\tilde{x}_{i-1}$ before the start of each time increment $i \geq 1$. The decision maker makes this decision while taking into account the system *context*, denoted $\mathcal{C}$, which may inform updates or changes to the newsstand. Such contextual information may include anticipated events that could affect demand for newspapers in the next cycle.

The physical twin produces a random profit for the newsvendor in each time increment, as follows. At time $t = 0$ and perhaps based on context $\mathcal{C}$, the decision maker stocks $\tilde{x}_0$ newspapers. The profit at the end of the first time increment, $[0, \delta)$, is

$$\tilde{Y}_1(\tilde{x}_0) = \begin{cases} p\min(D_1, \tilde{x}_0) - c\tilde{x}_0 + s\big(\tilde{x}_0 - \min(D_1, \tilde{x}_0)\big) & \text{if } m = 1, \\ p\min(D_1, \tilde{x}_0) - c\tilde{x}_0 & \text{otherwise.} \end{cases}$$

The number of unsold and unsalvaged newspapers at the end of the first time increment that are carried forward to the beginning of the next time increment $[\delta, 2\delta)$ is

$$\widetilde{X}_0^+ = \begin{cases} \tilde{x}_0 - \min(D_1, \tilde{x}_0) & \text{if } m \neq 1, \\ 0 & \text{otherwise.} \end{cases}$$

At the beginning of the next time increment $[\delta, 2\delta)$, the decision maker has the option to add $\tilde{x}_1 \geq 0$ newspapers at cost c; we assume the replenishment is instantaneous. We now write the profit expression for increments $i \geq 2$ recursively by conditioning on the number of unsold and unsalvaged newspapers carried forward from the previous time increment, $\tilde{x}_{i-2}^+$. Then the newsstand obtains a profit at the end of time increment $i \geq 2$ as

$$\tilde{Y}_i(\tilde{x}_{i-1} \,|\, \widetilde{X}_{i-2}^+ = \tilde{x}_{i-2}^+)$$
$$= \begin{cases} p\min(D_i, \tilde{x}_{i-2}^+ + x_{i-1}) - cx_{i-1} + s\big(\tilde{x}_{i-2}^+ + x_{i-1} - \min(D_i, \tilde{x}_{i-2}^+ + x_{i-1})\big) & \text{if } i \bmod m = 0, \\ p\min(D_i, \tilde{x}_{i-2}^+ + x_{i-1}) - cx_{i-1} & \text{otherwise,} \end{cases} \tag{2}$$

where the number of unsold and unsalvaged newspapers at the end of the *i*th time increment is

$$\widetilde{X}_{i-1}^{+} = \begin{cases} x_{i-2}^{+} + x_{i-1} - \min(D_i, x_{i-2}^{+} + x_{i-1}) & \text{if } i \bmod m \neq 0, \\ 0 & \text{otherwise.} \end{cases}$$

In each time increment *i*, the physical twin produces *data* containing the time of sale of each newspaper, denoted by $\tilde{\mathcal{D}}_i = \{S_{i1}, \ldots, S_{iN_i}\}$, where $N_1 = \min(D_1, x_0)$ and $N_i = \min(D_i, \tilde{x}_{i-2}^{+} + x_{i-1})$, $i \geq 2$ is the total number of newspapers sold in increment *i* (implicitly conditional on the leftover newspapers from the prior increment). Notice that unless the demand D_i is smaller than the total number of newspapers available at the beginning of increment *i*, we do not observe D_i. Instead, we observe the demand which is right-censored by the number of newspapers available.

The newsvendor would like to operate the newsstand to maximize the expected profit. To accomplish this goal, the newsvendor implements a Monte Carlo digital twin for the purpose of estimating the restocking policy which maximizes the expected profit in the next time increment, given data produced by the physical twin so far. That is, in time increment *i*, given access to data from all time increments up to and including $i-1$, $i \geq 2$, and looking forward from the start of time increment $i+1$, the newsvendor's *desired prediction* includes:

1. The optimal number of additional newspapers to stock at the beginning of the next time increment $i+1$ to maximize the expected profit over the time interval $[i\delta, (i+1)\delta)$; that is,

$$x^* = \text{argmax}_{x_i \geq 0} \, \mathrm{E}\left[\tilde{Y}_{i+1}(x_i \,|\, \widetilde{X}_{i-2}^{+} = \tilde{x}_{i-2}^{+})\right];$$

2. The expected profit under the optimal restocking amount x^*,

$$\theta^* = \mathrm{E}\left[\tilde{Y}_{i+1}(x^* \,|\, \widetilde{X}_{i-2}^{+} = \tilde{x}_{i-2}^{+})\right].$$

To obtain an estimator for the desired prediction (x^*, θ^*), the newsvendor implements a Monte Carlo digital twin of the newsstand which can simulate the newsstand operations over the time horizon $[i\delta, (i+1)\delta)$, $i \geq 1$. The desired prediction implies that the *desired parameter* θ for the embedded Monte Carlo simulation model is the expected profit in the next time increment for a given restocking decision *x*. In this simulation model, the *input variables* are the customer arrival times during the time horizon. To generate the customer arrival times, we require a *generator G* which is capable of generating *random numbers U*, such as `mrg32k3a` (L'Ecuyer et al. 2002), and an input model for the customer arrival process. Since the Monte Carlo digital twin should predict the future as a function of the past, we assume a parametric form for the customer arrival process which explicitly accounts for a long-term trend and cyclic behavior,

$$\lambda(\tau) = \exp\{\textstyle\sum_{k=0}^{p} \beta_k \tau^k + \eta \sin(\omega\tau + \phi)\} \text{ for all } \tau \geq 0,$$

where $\beta_1, \ldots, \beta_p, \eta, \omega, \phi$ are the parameters. The long-term trend is modeled through $\sum_{i=0}^{p} \beta_i \tau^i$ and the cyclic behavior is modeled through $\eta \sin(\omega\tau + \phi)$ (Nelson and Pei 2021, p. 172). Then at the start of time increment $i \geq 2$, the predictor sends the collected data $\mathcal{D}_i = \{S_{j1}, S_{j2}, \ldots, S_{jN_i}\}$ from the real-world system to the input model. The arrow implies a function which updates the estimated arrival process model parameters, denoted $(\hat{\beta}_1, \ldots, \hat{\beta}_p, \hat{\eta}, \hat{\omega}, \hat{\phi})$, perhaps using a combination of data from past increments with the current increment, thus yielding the updated *input model*

$$\hat{\lambda}_i(\tau) = \exp\{\textstyle\sum_{k=0}^{p} \hat{\beta}_k \tau^k + \hat{\eta} \sin(\hat{\omega}\tau + \hat{\phi})\} \text{ for all } \tau \geq 0;$$

see Lee et al. (1991). The input model yields the *input variables*, which consist of the virtual customer arrival times. Assuming customers arriving according to the input model are served instantaneously, the *logic model* appends each customer arrival time with a 1 if a newspaper is purchased and a 0 if no newspapers are available, where, for simplicity, we model each newspaper purchase as belonging to a separate customer. The logic model has policy parameter $x \geq 0$ which specifies the number of newspapers added at the beginning of the simulated time interval, and the system state variable maintained by the simulation model includes the current number of newspapers available for purchase. Then, the *output variable* for the simulation Y_{i+1} corresponds to the profit in the time increment $i+1$ under the restocking policy x. Given several replications, that is, several simulated customer arrival processes over the time horizon $\tau \in [i\delta, (i+1)\delta]$, the output random variables are used to calculate the *estimator* $\hat{\theta}$ for the *desired parameter* θ. Here, the desired parameter θ is a vector including the expected profit and other desirable quantities for decision-making, such as the probability of a stock-out.

Finally, the predictor belongs to both the physical-virtual feedback loop and the predictor-simulation feedback loop. In this sense, it serves as the coupling mechanism $\mathbb{C}$ from the ChatGPT definition of a digital twin. Its roles in each loop are as follows: The predictor controls the simulation as part of the predictor-simulation feedback loop. Therefore, the predictor controls when the input model receives updated data $\mathcal{D}$. It also determines which value of x to simulate next and the required precision for the estimator $\hat{\theta}$; thus, a simulation optimization algorithm is also embedded in the predictor. As part of the physical-virtual feedback loop, the predictor interfaces with the data from the physical twin and makes the estimated predictions available to the decision maker. Thus, in this setting, the predictor can declare that a good-enough estimator for x^* has been found using a stopping rule, or it can declare that there is not enough real time to find a better estimator, and, thus, recommend an update to the real-world system based on the available information.

4.3 Anomaly Detection in a Water Transmission Network

Consider a vendor who uses a large network of transmission lines to distribute water to storage facilities or consumers. The digital twin can be used to detect impending anomalies in the operation of such a network. In what follows, we provide an overview of the important components in such a digital twin, but at a broader level of detail than in Subsection 4.2.

The *physical twin* in this example consists of transmission lines equipped with pressure sensors installed at ℓ network locations, capable of recording and transmitting pressure data every $\delta > 0$ seconds. The *data* $\tilde{\mathcal{D}}$ consists of realizations from a vector time series $\tilde{V}(t\delta), t = 0,1,2,\ldots$, $\tilde{V}(t\delta) = (\tilde{V}_1(t\delta), \tilde{V}_2(t\delta), \ldots, \tilde{V}_\ell(t\delta))$ representing pressure observations at the ℓ fixed locations. The time series $\tilde{V}(t\delta), t = 0,1,2,\ldots$ can exhibit significant serial correlation and seasonal variability in response to variations in demand for water.

The *decision maker*, with safety and efficiency as overarching goals, seeks to distinguish variations in water pressure due to seasonal demand from those due to anomalous behavior, as might result from such conditions as the build-up of line debris or leakage. The *decision variable* $\tilde{x}$ may be particular aspects of the network that can be changed in response to an imminent anomaly, or to ensure a more efficient functioning of the physical twin. Specifically, $\tilde{x}$ might represent the configuration of valves ("on" or "off") within the network, or the capacity of individual links in the network. For concrete discussion, suppose the decision variable $\tilde{x}$ represents the configuration of states ("on" or "off") of the various valves in the transmission network, and suppose x corresponds to a "what-if" scenario, that is, an altered configuration being considered by the decision maker. Toward facilitating such a "what-if" analysis, the predictor first updates the input model using available data, and then seeks to use the Monte Carlo simulation model to report the network performance under different network configurations specified by changing the values of x. Using the notation in Figure 3, this corresponds to setting $\tilde{Y} := \{\tilde{V}(t\delta), t = 0,1,2,\ldots\}$ and $\tilde{\mathcal{D}} := \emptyset$.

The decision maker seeks a network configuration that results in a non-anomalous network pressure behavior. To define this idea more precisely, suppose μ_t denotes the distribution of the water pressure $\tilde{V}(t)$ at a future real time t, and μ_0 a reference distribution serving as the nominal distribution under regular network operation. (Both $\tilde{V}(t)$ and it's distribution μ_t depend on the configurations $\tilde{x}$, although we have suppressed such dependence for notational convenience.) An *anomaly* is said to happen at real time t if

$$\theta_t = \begin{cases} 1 & \text{if } |\psi(\mu_t) - \psi(\mu_0)| \geq \varepsilon, \\ 0 & \text{otherwise,} \end{cases}$$

where $\psi\colon \mathcal{P}([0,\infty)) \to \mathbb{R}$ is some chosen statistical functional, e.g., the total variation of a probability measure, $\mathcal{P}([0,\infty))$ is the set of probability measures supported on $[0,\infty)$, and $\varepsilon > 0$ is a chosen tolerance. Then, the *desired parameter* is an indicator variable of whether an anomaly occurs,

$$\theta := \max\{\theta_{t_k^*}\colon k = 1,2,\ldots,s\},$$

where $t_1^* < t_2^* < \ldots < t_s^*$ denote a given set of future times of interest where anomalous behavior will be checked. These time increments may also be constructed in the service of finding the earliest *time of occurrence* of the anomaly: $t^* := \inf\{t \in \{t_1^*, t_2^*, \ldots, t_s^*\}\colon |\psi(\mu_t) - \psi(\mu_0)| \geq \varepsilon\}$. (Let $t^* = \infty$ if $|\psi(\mu_t) - \psi(\mu_0)| < \varepsilon$ for all $t \in \{t_1^*, t_2^*, \ldots, t_s^*\}$.) The parameter θ is unknown because μ_t is unknown for all future t; it is also clearly dependent on $\tilde{x}$, even though we have suppressed such dependence for convenience. The decision maker implicitly seeks a configuration $\tilde{x}$ that ensures that $\theta = 0$ (or $t^* = \infty$), that is, an $\tilde{x}$ such that $|\psi(\mu_t) - \psi(\mu_0)| < \varepsilon$ for all $t \in \{t_1^*, t_2^*, \ldots, t_s^*\}$. If there is no such configuration, the decision maker may settle for a configuration $\tilde{x}$ that minimizes the deviation $|\psi(\mu_t) - \psi(\mu_0)|$, a scenario we do not consider.

To understand how one might construct the estimator $\hat{\theta}_t$ of θ_t for any $t \geq 0$ (that is, an estimator $\hat{\theta}_\tau$ evaluated at $\tau = t$), suppose we simulate n initial segments $V_i(\tau\delta), \tau \leq t_s^*; i = 1,2,\ldots,n$ of an underlying time series $V = \{V(\tau\delta), \tau \geq 0\}$ obtained from the Monte Carlo simulation of network water pressures at the configuration x. (Again, for convenience of notation, we have suppressed the dependence of V on x.) The simulated data can then be used to construct estimator $\hat{\mu}_{t_k^*,n}, k = 1,2,\ldots,s$ of the true distributions $\mu_{t_k^*}, k = 1,2,\ldots,s$, which in turn can be used to construct the estimators

$$\hat{\theta} := \max\left\{\hat{\theta}_{t_k^*,n}\colon k = 1,2,\ldots,s\right\} \text{ where } \hat{\theta}_{t_k^*,n} := \mathbb{I}(|\psi(\hat{\mu}_{t_k^*,n}) - \psi(\mu_0)| > \varepsilon);$$

a standard error estimator $\widehat{\text{se}}(\hat{\theta})$ can be constructed using batching methods (Su et al. 2024). While various choices exist, a simple estimated prediction $(\hat{x}^*, \hat{\theta}^*)$ corresponds to any configuration $x = \hat{x}^*$ that minimizes $\hat{\theta}$, along with $\hat{\theta}^* := \hat{\theta}(\hat{x}^*)$. Finally, the estimated prediction $(\hat{x}^*, \hat{\theta}^*)$ is communicated back to the decision maker, who might then use any available *contextual information* $\mathcal{C}$ to decide which of the altered configurations in $\hat{x}^*$ should be chosen for incorporation in the physical twin as $\tilde{x}$.

5 CONCLUDING REMARKS

The proposed Monte Carlo digital twin worldview is designed to provide a concise and precise description of the elements that constitute a Monte Carlo digital twin and how they interact. The three examples of Monte Carlo digital twins elucidate how the various components of the proposed worldview arise in different settings. We make no attempt to advocate for the use of digital twins; rather we hope to facilitate discussion of this popular topic.

ACKNOWLEDGMENTS

The authors thank the reviewers for insightful comments which improved the paper. Hunter would also like to thank Purdue students Zach Bernstein, Kaitlyn Borcherding, Kristine Ehlinger, and Sathwik Palwai for interesting discussions about the Monte Carlo digital twin worldview in IE 581 during the Spring 2025 semester; David Eckman, Shane Henderson, and Eunhye Song for interesting discussions; and the Air Force Office of Scientific Research for support under award number FA9550-23-1-0488.

REFERENCES

Asmussen, S., and P. W. Glynn. 2007. *Stochastic Simulation: Algorithms and Analysis*, Volume 57 of *Stochastic Modeling and Applied Probability*. New York: Springer-Verlag https://doi.org/10.1007/978-0-387-69033-9.

Biller, B., X. Jiang, J. Yi, P. Venditti, and S. Biller. 2022. "Simulation: The Critical Technology in Digital Twin Development". In *Proceedings of the 2022 Winter Simulation Conference*, 1340–1355 https://doi.org/10.1109/WSC57314.2022.10015246.

Fuller, A., Z. Fan, C. Day, and C. Barlow. 2020. "Digital Twin: Enabling Technologies, Challenges and Open Research". *IEEE Access* 8:108952–108971 https://doi.org/10.1109/ACCESS.2020.2998358.

Grieves, M., and J. Vickers. 2017. "Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems". In *Transdisciplinary Perspectives on Complex Systems*, 85–113. Switzerland: Springer https://doi.org/10.1007/978-3-319-38756-7_4.

He, L., L. Rhodes-Leader, and E. Song. 2024. "Digital Twin Validation with Multi-Epoch, Multi-Variate Output Data". In *Proceedings of the 2024 Winter Simulation Conference*, 347–358 https://doi.org/10.1109/WSC63780.2024.10838742.

Kapteyn, M. G., J. V. R. Pretorius, and K. E. Willcox. 2021. "A Probabilistic Graphical Model Foundation for Enabling Predictive Digital Twins at Scale". *Nature Computational Science* 1:337–347 https://doi.org/10.1038/s43588-021-00069-0.

L'Ecuyer, P., R. Simard, E. J. Chen, and W. D. Kelton. 2002. "An Object-Oriented Random-Number Package with Many Long Streams and Substreams". *Operations Research* 50(6):1073–1075 https://doi.org/10.1287/opre.50.6.1073.358.

Lee, S., J. R. Wilson, and M. M. Crawford. 1991. "Modeling and Simulation of a Nonhomogeneous Poisson Process Having Cyclic Behavior". *Communications in Statistics – Simulation and Computation* 20(2-3):777–809 https://doi.org/10.1080/03610919108812984.

Leemis, L. 2025. "Course Outline". IE 680 at William and Mary, https://www.math.wm.edu/~leemis/advsim.pdf, accessed April 28, 2025.

National Academies of Sciences, Engineering, and Medicine 2023. *Foundational Research Gaps and Future Directions for Digital Twins*. Washington, DC: The National Academies Press https://doi.org/10.17226/26894.

Nelson, B. L. 1983. *Variance Reduction in Simulation Experiments: A Mathematical - Statistical Framework*. Ph. D. thesis, Purdue University.

Nelson, B. L., and L. Pei. 2021. *Foundations and Methods of Stochastic Simulation: A First Course*. 2 ed. New York: Springer https://doi.org/10.1007/978-3-030-86194-0.

Nelson, B. L., and B. W. Schmeiser. 1983. "Variance Reduction: Basic Transformations". In *Proceedings of the 1983 Winter Simulation Conference*, 255–258. https://dl.acm.org/doi/pdf/10.5555/800043.801503.

Onggo, B. S., and C. S. Currie. 2024. "Extending Simulation Modeling Methodology for Digital Twin Applications". In *Proceedings of the 2024 Winter Simulation Conference*, 3058–3069 https://doi.org/10.1109/WSC63780.2024.10838633.

OpenAI ChatGPT 2025. "Definition and Formalization of Digital Twins". *OpenAI ChatGPT Response to User Query*. Available upon request or from session transcript dated May 3, 2025, https://chat.openai.com/.

Rasheed, A., O. San, and T. Kvamsdal. 2020. "Digital Twin: Values, Challenges and Enablers from a Modeling Perspective". *IEEE Access* 8:21980–22012 https://doi.org/10.1109/ACCESS.2020.2970143.

Rhodes-Leader, L. A., and B. L. Nelson. 2023. "Tracking and Detecting Systematic Errors in Digital Twins". In *Proceedings of the 2023 Winter Simulation Conference*, 492–503 https://doi.org/10.1109/WSC60868.2023.10408052.

Schmeiser, B. 1999. "Advanced Input Modeling for Simulation Experimentation". In *Proceedings of the 1999 Winter Simulation Conference*, 110–115 https://doi.org/10.1145/324138.324173.

Schmeiser, B. W. 1984. "Analytic Representations of Simulation". In *Proceedings of the 1984 Winter Simulation Conference*, 155–159. https://dl.acm.org/doi/pdf/10.5555/800013.809453.

Shafto, M., M. Conroy, R. Doyle, E. Glaessgen, C. Kemp, J. LeMoigne *et al*. 2012. "Modeling, Simulation, Information Technology and Processing Roadmap". Technical Report 32, National Aeronautics and Space Administration.

Su, Z., R. Pasupathy, Y. Yeh, and P. W. Glynn. 2024. "Overlapping Batch Confidence Intervals on Statistical Functions Constructed from Time Series: Application to Quantiles, Optimization, and Estimation". *ACM Transactions on Modeling and Computer Simulation* 34(2):10:1–10:43 https://doi.org/10.1145/3649437.

Zhou, E. 2024. "Data-Driven Simulation Optimization in the Age of Digital Twins: Challenges and Developments". In *Proceedings of the 2024 Winter Simulation Conference*, 31–45 https://doi.org/10.1109/WSC63780.2024.10838871.

AUTHOR BIOGRAPHIES

SUSAN R. HUNTER is an associate professor in the Edwardson School of Industrial Engineering at Purdue University. Her research interests include theoretical and algorithmic aspects of stochastic optimization in the presence of multiple performance measures. She currently serves as Vice President / President Elect of the INFORMS Simulation Society and previously served as the Program Chair for the 2024 Winter Simulation Conference. Her email address is susanhunter@purdue.edu and her website is https://web.ics.purdue.edu/~hunter63/.

RAGHU PASUPATHY is a professor in the Department of Statistics at Purdue University. His current research interests lie broadly in stochastic optimization, uncertainty quantification, and simulation methodology. He has been actively involved with the Winter Simulation Conference for the past 20 years. His e-mail address is pasupath@purdue.edu and his website https://web.ics.purdue.edu/~pasupath/ contains links to papers, software codes, and other material..

BRUCE W. SCHMEISER is an emeritus professor in the Edwardson School of Industrial Engineering at Purdue University. His research interests center on developing methods for better simulation experiments. He is a fellow of INFORMS and IISE, as well as recipient of the INFORMS Simulation Society's 2014 Lifetime Professional Achievement Award. A long-time participant in the Winter Simulation Conference, he served as the 1983 Program Chair and the 1988–1990 President of the Board of Directors. His email address is bruceschmeiser@gmail.com.