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.

QUANTIFYING UNCERTAINTY FROM MACHINE LEARNING SURROGATE MODELS EMBEDDED IN SIMULATION MODELS

Mohammadmahdi Ghasemloo[1], David J. Eckman[1], and Yaxian Li[2]

[1]Dept. of Industrial and Systems Engineering, Texas A&M University, College Station, TX, USA
[2]Intuit AI, Mountain View, CA, USA

ABSTRACT

Modern simulation models increasingly feature complex logic intended to represent how tactical decisions are made with advanced decision support systems (DSSs). For a variety of reasons, e.g., concerns about privacy, latency, and data intensity, users might choose to replace DSSs with approximate logic within the simulation model. This paper investigates the impacts of replacing DSSs with machine learning surrogate models on the estimation of system performance metrics. We distinguish this so-called *surrogate uncertainty* from conventional input uncertainty and develop approaches for quantifying the error introduced by the use of surrogate models. Specifically, we explore bootstrapping and Bayesian model averaging methods for obtaining quantile-based confidence intervals for expected performance measures and propose using regression-tree importance scores to apportion the overall uncertainty across input and surrogate models. We illustrate our approach through a contact-center simulation experiment.

1 INTRODUCTION

Many business processes involve the use of decision support systems (DSSs) to make optimal decisions in complex, dynamic environments. Examples include production planning and inventory control in supply chains, routing contacts in contact centers, and optimizing delivery routes in parcel delivery operations. When using stochastic simulation to analyze systems featuring a DSS, a modeler must choose how to incorporate the DSS in the simulation logic. In many cases, embedding the actual DSS in the simulation model may be impracticable due to concerns about privacy, latency, and data intensity.

One approach for representing the DSS in the simulation model is to replace it with a simpler set of rules based on expert knowledge of the DSS and its role in the decision-making process. While such rule-based systems can partially encode domain knowledge, they rely on subjective expertise and generally would not incorporate real-world data. As a result, they may fail to capture the complexity of the DSS, or at least not to a sufficiently high degree of fidelity for trusting the results of the study. In light of these limitations, we consider an alternative approach of replacing the DSS with a machine learning (ML) surrogate model. Integrating ML models within simulation frameworks is becoming increasingly prevalent, driven by recent advances in the offerings of commercial simulation software such as Simio (Simio LLC 2023). Unlike rule-based logic or heuristics, ML models leverage real-world data to learn complex patterns in how decisions have historically been made and capture intricate dependencies that may be challenging to explicitly encode through hand-crafted rules. Replacing computationally expensive DSSs, such as optimization algorithms, with ML models can accelerate simulation runs and enable larger experiments (Chen et al. 2024).

This paper concerns how to quantify the uncertainty in estimating some key performance indicator (KPI) of the simulation model due to approximating DSSs with surrogate models. We seek to address two key questions: *How can we obtain confidence intervals (CIs) for the KPI that account for the added uncertainty introduced by surrogate models? And what proportion of the output variability can be attributed to uncertainty in surrogate models versus uncertainty in other input models versus intrinsic random error?* As a preliminary investigation in these directions, we study how two techniques used to quantify conventional input uncertainty—bootstrapping and Bayesian model averaging (BMA)—can be broadly applied to construct CIs that account for this so-called *surrogate uncertainty*. Additionally, we develop a regression-tree-based framework for quantifying the uncertainty attributable to different sources. These approaches offer several advantages. First, they are non-parametric and thus allow for considerable modeling flexibility without relying on strong distributional assumptions. Second, they straightforwardly accommodate cases with multiple input and surrogate models. Third, the framework can be applied across a wide range of surrogate model types without requiring model-specific assumptions.

In the simulation literature, *input uncertainty (IU)* refers to the study of the uncertainty about simulation KPIs that arises from needing to estimate input models from limited data (Nelson and Pei 2024; Barton et al. 2022; Lam and Qian 2022). Past IU approaches tend to deal with parametric input distributions, e.g., the arrival rate of a stationary Poisson process. In contrast, our approach addresses a different source of uncertainty: the approximation of embedded decision logic using ML surrogate models. This creates several important distinctions: First, rather than fitting parametric distributions to observed data, training surrogate models involves fitting (possibly parametric) functions to labeled input-output pairs, i.e., supervised learning. Second, surrogate model selection involves trade-offs between accuracy, interpretability, and computational cost, which are often irrelevant in conventional IU settings where fitting a distribution is typically straightforward and cheap. Third, whereas traditional IU concerns the input data to the simulation model, surrogate uncertainty pertains to the decision logic itself, which can alter the sequence and structure of events within a simulation run in complex ways.

In the applied mathematics community, another form of surrogate uncertainty is referred to as *uncertainty quantification (UQ)*. This area typically focuses on computer simulation experiments involving expensive deterministic simulation models, often physics based. In contrast to stochastic simulation models, output uncertainty arises from varying the model's input parameters and studying how the output is affected. The high computational costs associated with these kinds of simulation models motivates the use of surrogate models trained on data produced by a designed simulation experiment (Popp 2023; Andrés-Thió et al. 2024; Ranftl and von der Linden 2021). Our approach differs in that the surrogate models we consider are substitutes for only a part of a simulation model's logic and not for the model as a whole.

The remainder of the paper is structured as follows. In Section 2, we provide a few motivating examples of DSSs that arise in practice in systems commonly studied using stochastic simulation. Section 3 introduces a mathematical framework for surrogate uncertainty and draws comparisons to input uncertainty. We detail our approaches for uncertainty quantification in Sections 4 and 5 and apply them in numerical experiments in Section 6. We summarize our findings and future research directions in Section 7.

2 DECISION SUPPORT SYSTEMS IN SIMULATED SYSTEMS

Decision support systems play a crucial role in optimizing complex operational processes across various domains, including logistics, customer service, and energy management. These systems rely on optimization algorithms or rule-based heuristics to make informed decisions in tactical planning contexts. In this section, we present three representative examples of the kinds of DSSs that arise in systems commonly studied using stochastic simulation. The DSSs differ in terms of their application areas, their frequency of use, and the possible interdependency of the decisions they make over time.

Parcel Delivery Route Optimization A courier company receives a batch of parcels each day that need to be delivered to customers, where the number of parcels and their locations are both random. An optimization algorithm determines the most efficient delivery routes for the fleet of vehicles by minimizing some combination of travel distance and operational costs. Each day's deliveries are assumed to be independent, hence the decisions made with the aid of the DSS do not depend on past decisions (Gendreau et al. 1999). If the company aims to evaluate the long-run expected cost over, say, the next year, it might turn to a simulation model that generates random orders and locations and simulates delivery operations.

Contact-Center Routing A contact center assigns contacts awaiting service to agents based on complex decision rules involving queue lengths, agent availability, and service priorities as illustrated in

Figure 1. To develop a long-term staffing and resource allocation plan for the contact center, a decision-maker must run thousands of replications of a simulator of daily contact-center operations in which a routing DSS is invoked for each contact. For such a large-scale experiment, directly embedding the real DSS into the simulation could be computationally prohibitive; a surrogate routing model trained on historical contact routing decisions might be used instead.

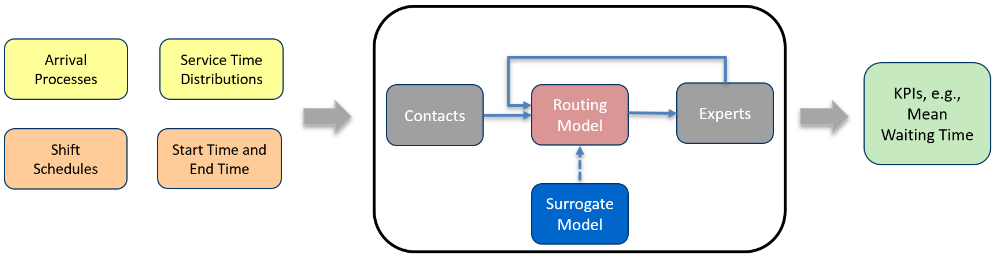


Figure 1: Schematic representation of a contact-center simulation model consisting of stochastic input model parameters (yellow), structural parameters (orange), core system components (gray), a surrogate model (blue), and outputs (green).

Power Dispatching In power systems, real-time dispatching of electricity is performed every five minutes based on forecasted demand and the solution of an optimization problem. Past dispatch decisions influence generator availability and grid conditions in subsequent periods. A stochastic simulation model can be employed to evaluate power dispatching policies under demand uncertainty. Exactly solving the optimization problem at each time step throughout a simulation replication is computationally expensive, particularly as the dimension of the demand grows. To reduce simulation run times, the optimization routine (the DSS) can be replaced with an optimization proxy, a surrogate model trained on historical optimization problem instances and their solutions (Chen et al. 2024).

In addition to those mentioned above, there are various reasons why a user may decide against embedding a DSS in a simulation model:

- **Privacy and Security**: The DSS may interact with proprietary information, algorithms, business rules, or trade secrets. These concerns are further amplified when simulations are built or run by third-party entities, such as consultants. Directly integrating the DSS into the simulation could expose confidential information or decision logic, creating potential vulnerabilities.
- **Latency**: The DSS may be too slow to be repeatedly invoked within the simulation, i.e., simulation time advances too quickly relative to the real-world time experienced when calling the DSS.
- **Data Intensity**: Having the simulation make many queries of the DSS may slow down the DSS when it is simultaneously needed for real-time decision-making, causing delays in critical business functions and possibly degrading real-world performance.

3 PROBLEM FORMULATION

We define a DSS as a mapping that takes as input system state or contextual parameters and produces an output that changes the state of the system. In terms of notation, we define a DSS as a function $D : \mathscr{S} \to \mathscr{A}$, where D maps an instance $s \in \mathscr{S}$ to an action $a = D(s) \in \mathscr{A}$ taken by the system. The input s consists of all variables needed to describe an applicable problem instance for the DSS. These variables may originate from multiple sources, including input distributions, the current system state, external non-stochastic parameters (contextual factors), and objects within the system. Input variables can be continuous, such as customer waiting times in a contact center, discrete, such as queue lengths, or categorical, such as whether an operator is idle or busy. The output (action) may represent a variety of decisions depending on the context, such as assignments, schedules of future events, or other state-updating operations.

3.1 Machine Learning Surrogate Models

We assume that the surrogate model is an ML model trained and tested on real-world data. In this statistical learning task, the features are the state and contextual variables comprising s, and the response is the action made by the DSS. In this paper, we assume that the DSS is deterministic, i.e., given a fixed instance s, it returns a fixed action a. This setup deviates from the typical supervised learning setting wherein the observed response includes random noise. Our intention behind making this assumption is to avoid dealing with the added complexity (and error) of having stochastic elements in the simulation model's logic. Nonetheless, standard ML models can still be applied to this noiseless case.

The surrogate models we envisage can be broadly divided into two categories and further analyzed in terms of interpretability, accuracy, and computational efficiency:

- **Non-Bayesian models:** Simple models such as linear regression and decision trees offer high interpretability and low computational cost, whereas more flexible models like artificial neural networks (ANNs) and gradient boosting machines can achieve higher predictive accuracy, especially with large datasets, but at the cost of reduced interpretability and increased training time (Hastie et al. 2009). Bootstrapping is a suitable approach for assessing uncertainty caused by incorporating these models.
- **Bayesian models:** Bayesian models capture epistemic uncertainty through probabilistic inference. Bayesian linear regression, for example, is highly interpretable and suitable for smaller datasets, while more complex models like Gaussian processes (GPs) and Bayesian neural networks (BNNs) offer greater flexibility and predictive accuracy. However, this comes at the expense of increased computational cost, especially when sampling from the posterior distribution or scaling to large datasets. For these models, uncertainty can be quantified through Bayesian model averaging.

Both Bayesian and non-Bayesian models span the spectrum from parametric to nonparametric forms. Parametric models, such as linear regression or BNNs, rely on a fixed number of parameters where nonparametric models like gradient boosting and GPs are data adaptive without assuming a predefined structure. Our framework accommodates this full range of models without making strong assumptions about their form.

3.2 Surrogate Uncertainty

Let $\mathbf{F}^c = (F_1^c, F_2^c, \ldots, F_I^c)$ represent a collection of I true (correct) input models that drive the inherent stochasticity in the simulated system. Throughout this paper, we assume that the true input models $F_1^c, F_2^c, \ldots, F_I^c$ are mutually independent, meaning that the random variables generated from them in the course of a replication are independent of each other. Similarly, let $\mathbf{D}^c = (D_1^c, D_2^c, \ldots, D_L^c)$ denote the set of true DSSs whose logic is to be captured in the simulation model. For simplicity, we adopt the standard setup in the IU literature of studying a single expected simulation output. The simulation output on a given replication $j = 1, 2, \ldots$ for a simulation model endowed with $\mathbf{F}^c$ and $\mathbf{D}^c$ can be expressed as

$$Y_j(\mathbf{F}^c, \mathbf{D}^c) = \mu(\mathbf{F}^c, \mathbf{D}^c) + \varepsilon_j(\mathbf{F}^c, \mathbf{D}^c),$$

where $\mu(\mathbf{F}^c, \mathbf{D}^c) \equiv \mathbb{E}[Y_j(\mathbf{F}^c, \mathbf{D}^c)]$ represents the expected simulation output when hypothetically running the simulation model with the true input models and true DSSs, and $\varepsilon_j(\mathbf{F}^c, \mathbf{D}^c)$ is a mean-zero random variable capturing the stochastic noise produced from the use of stochastic input models.

Because the true input models are invariably unknown and the true DSSs might be impractical to embed in the simulation model, a standard approach to estimating $\mu(\mathbf{F}^c, \mathbf{D}^c)$ is to obtain estimated input models $\hat{\mathbf{F}}$ and estimated DSSs $\hat{\mathbf{D}}$ from real-world data, run n replications using $\hat{\mathbf{F}}$ and $\hat{\mathbf{D}}$, and compute

$$\bar{Y}(\hat{\mathbf{F}}, \hat{\mathbf{D}}) = \frac{1}{n} \sum_{j=1}^{n} Y_j(\hat{\mathbf{F}}, \hat{\mathbf{D}}).$$

A confidence interval for $\mu(\hat{\mathbf{F}},\hat{\mathbf{D}})$ can then be constructed as $\bar{Y}(\hat{\mathbf{F}},\hat{\mathbf{D}}) \pm t_{\alpha/2,n-1}\cdot s/\sqrt{n}$, where s^2 is the sample variance and $t_{\alpha/2,n-1}$ is the $1-\alpha/2$ quantile of a Student's t-distribution with $n-1$ degrees of freedom. This method, however, does not yield a satisfactory confidence interval for $\mu(\mathbf{F}^c,\mathbf{D}^c)$, as it overlooks the inherent uncertainty in estimating $\mathbf{F}^c$ and $\mathbf{D}^c$ using $\hat{\mathbf{F}^c}$ and $\hat{\mathbf{D}}$. Prior research on input uncertainty has largely focused on the error in estimating $\mu(\mathbf{F}^c,\mathbf{D}^c)$ introduced by using $\hat{\mathbf{F}}$ in place of $\mathbf{F}^c$, whereas our work emphasizes the error introduced by running the simulation with $\hat{\mathbf{D}}$ instead of $\mathbf{D}^c$, which we call *surrogate uncertainty*. We also aim to consider the joint effect of having estimated both $\mathbf{F}^c$ and $\mathbf{D}^c$ on the overall uncertainty.

To explain how conventional IU is handled, we temporarily suppose that the true DSSs $\mathbf{D}^c$ can be embedded in the simulation model. In this setting, the estimated input models $\hat{\mathbf{F}}$ are treated as random, reflecting the epistemic uncertainty due to limited input data. Under these assumptions, the total variance of the estimator $\bar{Y}(\hat{\mathbf{F}},\mathbf{D}^c)$ can be decomposed using the law of total variance:

$$\mathrm{Var}(\bar{Y}(\hat{\mathbf{F}},\mathbf{D}^c)) = \mathrm{Var}(\mathbb{E}[\bar{Y}(\hat{\mathbf{F}},\mathbf{D}^c) \mid \hat{\mathbf{F}}]) + \mathbb{E}[\mathrm{Var}(\bar{Y}(\hat{\mathbf{F}},\mathbf{D}^c) \mid \hat{\mathbf{F}}] = \sigma^2_{\mathrm{IU}} + \sigma^2_{\mathrm{MC}}.$$

Here, σ^2_{IU} captures the contribution of input uncertainty—that is, how much the expected simulation output varies due to uncertainty in $\hat{\mathbf{F}}$—while σ^2_{MC} captures the (Monte Carlo) simulation noise conditional on fixed inputs and DSSs.

In this paper, we wish to additionally study the effect of replacing $\mathbf{D}^c$ in the simulation model with surrogate models $\hat{\mathbf{D}}$. As before, the standard confidence interval produced from running n i.i.d. replications with $\hat{\mathbf{F}}$ and $\hat{\mathbf{D}}$ will fail to account for the uncertainty about $\mathbf{D}^c$ and may fail to cover $\mu(\mathbf{F}^c,\mathbf{D}^c)$ with the prescribed confidence. This is due to the fact that replacing $\mathbf{D}^c$ with $\hat{\mathbf{D}}$ will affect both the variance and bias of the estimator $\bar{Y}(\hat{\mathbf{F}},\hat{\mathbf{D}})$. Applying the law of total variance slightly differently, the variance of the estimator $\bar{Y}(\hat{\mathbf{F}},\hat{\mathbf{D}})$—where now both $\hat{\mathbf{F}}$ and $\hat{\mathbf{D}}$ are treated as random quantities—can be decomposed as

$$\mathrm{Var}(\bar{Y}(\hat{\mathbf{F}},\hat{\mathbf{D}})) = \mathrm{Var}(\mathbb{E}[\bar{Y}(\hat{\mathbf{F}},\hat{\mathbf{D}}) \mid \hat{\mathbf{F}},\hat{\mathbf{D}}]) + \mathbb{E}[\mathrm{Var}(\bar{Y}(\hat{\mathbf{F}},\hat{\mathbf{D}}) \mid \hat{\mathbf{F}},\hat{\mathbf{D}})] = \sigma^2_{\mathrm{IS}} + \sigma^2_{\mathrm{MC}}.$$

This decomposition separates the simulation error, σ^2_{MC}—which subtly differs from the one previously mentioned—and attributes the remaining uncertainty to the input and surrogate models, σ^2_{IS}. We aim to further decompose σ^2_{IS} across each source of uncertainty in Section 5. Estimating the variance of our estimator $\bar{Y}(\hat{\mathbf{F}},\hat{\mathbf{D}})$ is a vital part of building an asymptotically valid confidence interval for $\mu(\mathbf{F}^c,\mathbf{D}^c)$.

Remark. The presence of surrogate models and input distributions also affects the bias of the estimator:

$$\mathrm{Bias} = \mathbb{E}[Y(\hat{\mathbf{F}},\hat{\mathbf{D}})] - \mu(\mathbf{F}^c,\mathbf{D}^c),$$

where the expectation in the first term is taken over $\hat{\mathbf{F}}$, $\hat{\mathbf{D}}$, and replications of the simulation model. Although bias estimation is not the primary focus of this paper, it nonetheless plays an important role in how well the resulting CIs cover $\mu(\mathbf{F}^c,\mathbf{D}^c)$. Bias is often overlooked or disregarded in the IU literature—partly because it is difficult to estimate and is expected to vanish asymptotically faster than the variance, as the size of the input dataset increases, assuming the classes of estimated input models match those of the true input models, or are increasingly flexible enough to closely approximate them (Morgan et al. 2019). Our experiments reveal that this reasoning may not so easily extend to the surrogate uncertainty setting. In particular, when the ML technique used to fit a surrogate model has high bias, increasing the size of the training data may not necessarily eliminate the bias term, and this persistent bias can adversely affect CI coverage. We acknowledge the significance of this issue and leave a more detailed investigation of bias estimation and its implications for future work.

4 (RE)SAMPLING INPUT AND SURROGATE MODELS

Before describing ways to estimate the variance of $\bar{Y}(\hat{\mathbf{F}},\hat{\mathbf{D}})$, we first discuss two ways of sampling from the space of input and surrogate models: bootstrapping, which entails resampling input or surrogate training data and refitting models, and Bayesian model averaging, which naturally captures the epistemic uncertainty arising from limited input or training data.

4.1 Bootstrapping

Bootstrapping (Efron and Tibshirani 1994) is a non-parametric resampling technique that involves resampling the observed data *with replacement* to create multiple bootstrapped datasets and is commonly applied to study input uncertainty (Barton et al. 2022). A straightforward way to leverage bootstrapping for studying surrogate uncertainty is to resample the input-output pairs in the training dataset with replacement (Tang and Westling 2024). While bootstrapping does not impose strong distributional assumptions on the data, it can be challenging to apply in certain contexts, e.g., time series data, which exhibits temporal dependencies, meaning that randomly resampling observations can disrupt inherent correlations and patterns. This situation could arise, for example, if the DSS were a forecasting tool. Specialized techniques such as moving block bootstrap (Lahiri 2003) or stationary bootstrap (Politis and Romano 1994) have been developed to address this case, but they can be more complicated to implement. Bootstrapping also tends to multiply the required computational effort. In conventional input uncertainty settings, obtaining $\hat{\mathbf{F}}$s via bootstrapping is relatively inexpensive because it entails fitting input models, which is typically fast. In contrast, when bootstrapping is used to quantify surrogate model uncertainty, the computational burden is expected to be higher because a surrogate model needs to be trained for each bootstrap sample. Complex ML models, such as deep neural networks, often require substantial training time, especially when the data are high-dimensional or the model architecture is deep. When using bootstrapping for IU, the dominant cost is often the simulation runs themselves, whereas in our setting of surrogate uncertainty, the cost of repeatedly training surrogate models may become comparable or even dominant.

4.2 Bayesian Model Averaging

Bayesian model averaging offers a principled framework for accounting for uncertainty in the form of a posterior distribution—obtained via Bayes' rule—which inherently captures the remaining uncertainty about the model after observing the data (Chick 2001). This approach treats the parameters of input or surrogate models as random variables, which are updated using Bayes' rule when training the model. Bayesian surrogate models, e.g., GPs or BNNs, are equipped with posterior distributions on their predictions as functions of inputs for GPs (Rasmussen and Williams 2006) or parameters weights for BNNs (Neal 2012). Instead of resampling and retraining the surrogate model multiple times, as required in bootstrapping, many Bayesian methods simply require sampling from the posterior distribution of the model parameters. For example, when working with BNNs, generating new surrogate models requires only sampling from the posterior distribution rather than performing a full retraining process. On the other hand, sampling from GPs becomes increasingly expensive as the number of evaluation points grows, which in our setting corresponds to the number of times the surrogate model is invoked within a simulation replication.

5 UNCERTAINTY QUANTIFICATION

Let $\{\hat{F}_i^{*(b)}\colon b = 1,2,\ldots,B \text{ and } i = 1,2,\ldots,I\}$ and $\{\hat{D}_l^{*(b)}\colon b = 1,2,\ldots,B \text{ and } l = 1,2,\ldots,L\}$ represent B instances of each source of input uncertainty and surrogate uncertainty, respectively, generated via either bootstrapping or posterior sampling. It is likely prohibitively expensive to simulate all B^{I+L} possible combinations (henceforth called configurations) of these instances, even for moderate values of B, I, and L. To address this challenge, we turn to design of experiments (DOE) to choose a much smaller design of configurations that reasonably covers the space of input and surrogate model instances. In this paper, we employ Latin Hypercube (LH) designs (Santner et al. 2018), which ensure that every level of every factor (in our context, every instance of every source of uncertainty) appears exactly once in a design. More specifically, a LH design is constructed by sampling *without replacement* from the set of B instances of each input and surrogate model to form configurations until all instances have been exhausted. This process can be repeated to create additional LH designs, that can then be stacked to form a larger design. The total number of configurations in such a stacked design is $B' = BS$, where S is the number of stacks. The simulation model is then run n times at each configuration, resulting in a dataset as depicted in Table1. The

B' configurations do not represent B' unique input or surrogate models, but rather are formed by selecting combinations from base sets of B bootstrapped instances of each input and surrogate model using stacked LH designs.

Table 1: Sampling framework for input and surrogate model uncertainty.

Input Models	**Surrogate Models**	**Output**
$\hat{\mathbf{F}}^{\langle 1\rangle}$	$\hat{\mathbf{D}}^{\langle 1\rangle}$	$Y_1^{\langle 1\rangle} = Y_1(\hat{\mathbf{F}}^{\langle 1\rangle}, \hat{\mathbf{D}}^{\langle 1\rangle})$ $Y_2^{\langle 1\rangle} = Y_2(\hat{\mathbf{F}}^{\langle 1\rangle}, \hat{\mathbf{D}}^{\langle 1\rangle})$ $\vdots$ $Y_n^{\langle 1\rangle} = Y_n(\hat{\mathbf{F}}^{\langle 1\rangle}, \hat{\mathbf{D}}^{\langle 1\rangle})$
$\hat{\mathbf{F}}^{\langle 2\rangle}$	$\hat{\mathbf{D}}^{\langle 2\rangle}$	$Y_1^{\langle 2\rangle} = Y_1(\hat{\mathbf{F}}^{\langle 2\rangle}, \hat{\mathbf{D}}^{\langle 2\rangle})$ $Y_2^{\langle 2\rangle} = Y_2(\hat{\mathbf{F}}^{\langle 2\rangle}, \hat{\mathbf{D}}^{\langle 2\rangle})$ $\vdots$ $Y_n^{\langle 2\rangle} = Y_n(\hat{\mathbf{F}}^{\langle 2\rangle}, \hat{\mathbf{D}}^{\langle 2\rangle})$
$\vdots$	$\vdots$	$\vdots$
$\hat{\mathbf{F}}^{\langle B'\rangle}$	$\hat{\mathbf{D}}^{\langle B'\rangle}$	$Y_1^{\langle B'\rangle} = Y_1(\hat{\mathbf{F}}^{\langle B'\rangle}, \hat{\mathbf{D}}^{\langle B'\rangle})$ $Y_2^{\langle B'\rangle} = Y_2(\hat{\mathbf{F}}^{\langle B'\rangle}, \hat{\mathbf{D}}^{\langle B'\rangle})$ $\vdots$ $Y_n^{\langle B'\rangle} = Y_n(\hat{\mathbf{F}}^{\langle B'\rangle}, \hat{\mathbf{D}}^{\langle B'\rangle})$

The choices of B, S, and n affect the computational cost of the experiment and the coverage probability of the CIs. Increasing B enhances the diversity of the sampled input and surrogate model instances, increasing S improves the coverage of the cross-product of those instances, and increasing n reduces the impact of simulation noise by averaging over more replications. Note that the number of input and surrogate models that must be fitted or retrained depends on B, not S. A user can conduct a preliminary experiment with a small design to assess the contributions of each source of uncertainty and then choose B, S, and n accordingly for a refined design.

We next describe how one can analyze the data shown in Table 1 to construct CIs for $\mu(\mathbf{F}^c, \mathbf{D}^c)$ and use regression trees to decompose and attribute overall uncertainty to individual sources.

5.1 Confidence Intervals

We construct an approximate $1-\alpha$ quantile-based confidence interval for $\mu(\mathbf{F}^c, \mathbf{D}^c)$ as

$$\left[Q_{\alpha/2}(\bar{Y}), Q_{1-\alpha/2}(\bar{Y})\right],$$

where $Q_p(\bar{Y})$ denotes the empirical p-quantile of the sample means $\{\bar{Y}^{\langle 1\rangle}, \bar{Y}^{\langle 2\rangle}, \ldots, \bar{Y}^{\langle B'\rangle}\}$ where

$$\bar{Y}^{\langle b'\rangle} = \frac{1}{n}\sum_{j=1}^{n} Y_j(\hat{\mathbf{F}}^{\langle b'\rangle}, \hat{\mathbf{D}}^{\langle b'\rangle}) \quad \text{for } b' = 1, 2, \ldots, B'.$$

This confidence interval provides a nonparametric estimate of the overall uncertainty incorporating both input model and surrogate model uncertainty. The confidence interval is asymptotically valid as the size of the input and training datasets grows large, assuming that the true input and surrogate models lie within the classes used to generate the sampled instances. This mirrors established results in the input uncertainty literature for both bootstrapping and Bayesian model averaging. However, coverage can still be adversely affected by bias—particularly for small sample sizes—underscoring the importance of selecting flexible models and acquiring sufficient training data.

5.2 Using Regression Trees to Attribute Uncertainty

The confidence interval presented above quantifies overall uncertainty, but does not distinguish the individual contributions from each input model and surrogate model. Consequently, decision-makers looking to prioritize efforts in reducing uncertainty—such as increasing the number of simulation replications, collecting more real-world data to improve input models, or enhancing the surrogate model through hyperparameter tuning or switching to a more flexible architecture—could benefit from techniques that attribute uncertainty to specific sources. Such a decomposition could allow practitioners to target their computational or data collection efforts toward the most influential contributors to uncertainty.

We introduce a method for quantifying the contribution of different sources to overall uncertainty that is compatible with both parametric and non-parametric input and surrogate models. Our approach leverages *variable importance* metrics from regression trees (James et al. 2021), which assign a value to each feature based on the total reduction in the total sum of squares (TSS) resulting from splits on that feature. Our approach strongly resembles Analysis of Variance (ANOVA); however, whereas ANOVA separately accounts for interaction effects, tree-based importance scores absorb interactions into the main effects.

In our context, we consider a regression tree trained on the dataset illustrated in Table 1, where the I input models $\hat{\mathbf{F}}^{\langle\cdot\rangle}$ and L surrogate models $\hat{\mathbf{D}}^{\langle\cdot\rangle}$ are treated as categorical features with levels corresponding to the sampled model instances. The tree is grown using recursive binary splitting, iteratively minimizing the residual sum of squares (RSS), until each configuration is its own terminal node, at which point the RSS corresponds to the portion of the TSS that is not explained by the input and surrogate models.

The TSS is defined as

$$\text{TSS} = \sum_{b'=1}^{B'} \sum_{j=1}^{n} \left(Y_j^{\langle b'\rangle} - \bar{\bar{Y}}\right)^2, \quad \text{where} \quad \bar{\bar{Y}} = \frac{1}{nB'} \sum_{b'=1}^{B'} \sum_{j=1}^{n} Y_j^{\langle b'\rangle},$$

and the RSS is given by:

$$\text{RSS} = \sum_{b'=1}^{B'} \sum_{j=1}^{n} \left(Y_j^{\langle b'\rangle} - \hat{Y}_j^{\langle b'\rangle}\right)^2,$$

where $\hat{Y}_j^{\langle b'\rangle}$ is the predicted value of the KPI for the configurations b' from the fully grown regression tree, which in our case is $\bar{Y}^{\langle b'\rangle}$.

To further understand the role of model complexity in this framework, we consider the concept of *degrees of freedom* (DoF), which represents the effective number of parameters used by a model to fit data. In regression settings, DoF is often interpreted as a measure of a model's flexibility or complexity and plays a critical role in error decomposition (Ye 1998). For regression trees, a common approximation for DoF is the number of terminal (leaf) nodes (Hastie et al. 2009). Furthermore, the contribution of individual features to model complexity is the number of times each feature is used for splitting, where each split contributes approximately one additional DoF (Mentch and Zhou 2020). To attribute uncertainty to each source, we compute the reduction in TSS attributable to each source and normalize it by the number of times the corresponding source appears in the tree splits. Finally, we divide this value by the DoF of the RSS, $nB' - B' + 1$, to obtain the simulation error's contribution to the overall variance.

6 A NUMERICAL EXAMPLE

To demonstrate the measurable effects of substituting a surrogate model for a DSS in a simulation model, we consider a contact-center simulation. The model includes two contact groups, where contacts from each group are assumed to arrive according to stationary Poisson processes with arrival rates $\lambda_1 = 80$ and $\lambda_2 = 90$ contacts per hour and have exponential service time distributions with means of $\theta_1 = 4$ and $\theta_2 = 5$ minutes per contact. We are interested in estimating the expected average waiting time for the first contact group. For simplicity, we assume that the patience times of the contacts are sufficiently large so that no contacts abandon the queue. There are three expert groups having 4, 4, and 5 experts, respectively, and each expert group is capable of handling every type of contact. So the only difference in experts comes from routing.

The contact center employs a routing mechanism that dynamically assigns contacts to expert groups based on system conditions. The routing mechanism is triggered when one of two scenarios occurs: If a contact arrives to find their contact group queue is empty and one or more experts are available, the contact is immediately assigned to one of the available experts. Alternatively, if an expert becomes available when queues are present, a waiting contact is selected to be serviced by that expert. Because of these cases, we consider two distinct DSSs: a contact-triggered routing model, denoted by D^c_C, and an expert-triggered routing model, denoted by D^c_E. Both D^c_C and D^c_E are rule-based functions with non-trivial decision rules. Specifically, D^c_C takes as input the number of idle experts in each expert group and D^c_E takes as input the longest waiting times of each contact group and their queue lengths. We run the simulation for five days using the true input models and the true DSSs to generate datasets of routing decisions, which are used to train the surrogate models, and datasets of arrival and service times, which are used to estimate the input model parameters. We further simulate 100 replications (days) under this setting and estimate the expected mean waiting time of the first contact group to be 1.07 ± 0.031 minutes. We test two modeling paradigms in our experiments: a frequentist approach that uses maximum likelihood estimation (MLE) for estimating the parameters of the interarrival time and service time distributions and decision trees for surrogate modeling, and a Bayesian approach that uses Bayesian parameter estimation for the input model parameters—which reduces to MLE under non-informative priors—and BNNs for surrogate modeling.

In the first experiment, we fit decision tree classifiers as surrogates for the two routing models. The contact-side routing model achieves an accuracy of 98%, while the expert-side routing model achieves an accuracy of 97%. Three cases are examined: In the first case, simulations are conducted without incorporating any input or surrogate uncertainty, i.e., we run 1250 replications using the estimated input parameters and the trained surrogate models and calculate a point estimate and 90% CI for the expected average waiting time of the first contact group. In the second case, only input uncertainty is considered. The trained surrogate models are used, while the input parameters are bootstrapped 5 times to obtain new estimates for each parameter, defining a design point. At each design point, the simulation model is executed 10 times using $\hat{D}_C$ and $\hat{D}_E$, and the sample mean is computed. A quantile-based method is then used to construct a 90% confidence interval. The third case extends the analysis by accounting for both input and model uncertainty. Samples are drawn from both the routing models and the input parameters before running the simulations. Bootstrapping on training datasets is performed to generate $\hat{D}^{*(b)}_C$ and $\hat{D}^{*(b)}_E$, $b = 1, 2, \ldots, 5$. Additionally, $B = 5$ input parameter instances are obtained for each of $\lambda_1, \lambda_2, \theta_1, \theta_2$ via bootstrapping. The simulation model is then executed using sampled surrogate models $\hat{D}^{*(b)}_C$ and $\hat{D}^{*(b)}_E$ and input parameters $\lambda^{*(b)}_1, \lambda^{*(b)}_2, \theta^{*(b)}_1$, and $\theta^{*(b)}_2$. We used $S = 25$ stacks of non-overlapping Latin-hypercube designs with $n = 10$ replications taken at each design point.

In the second experiment, we perform the same three-case experiment, but this time we use Bayesian parameter estimation and train two BNNs for the surrogate models, where each BNN consists of one hidden layer with 128 nodes and the ReLU activation function and a softmax activation function at the output layer. All parameters in the neural network have prior distributions that are normal with mean 0 and variance of 0.1. We use 10 epochs to train each BNN and then use the posterior means of the parameters to specify the surrogate models. The contact-side and expert-side routing surrogate models achieved classification accuracies of 93% and 92%, respectively. For the input models, we used an exponential-Gamma conjugate prior to obtain posterior distributions of the arrival and service rate parameters.

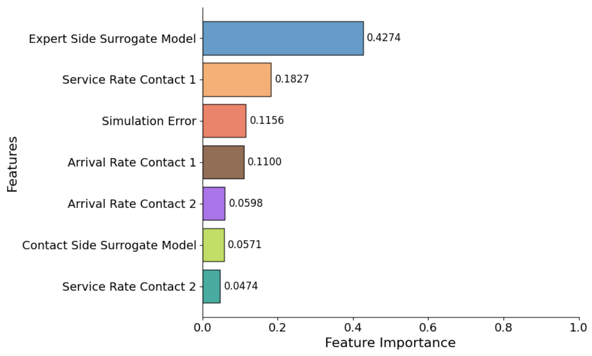
Table 2: 90% CIs for expected average waiting time under frequentist and Bayesian approaches.

Setting	**Approach**	**Mean**	**90% CI**	**Replications**
Frequentist	Conventional	1.11	[1.096, 1.124]	1250
	Input Uncertainty	1.087	[1.005, 1.175]	$S = 25$, $B = 5$, $n = 10$
	Surrogate + Input Uncertainty	1.037	[0.942, 1.134]	$S = 25$, $B = 5$, $n = 10$
Bayesian	Conventional	4.015	[3.671, 4.359]	1250
	Input Uncertainty	3.604	[1.043, 7.873]	$S = 25$, $B = 5$, $n = 10$
	Surrogate + Input Uncertainty	2.691	[0.716, 8.230]	$S = 25$, $B = 5$, $n = 10$

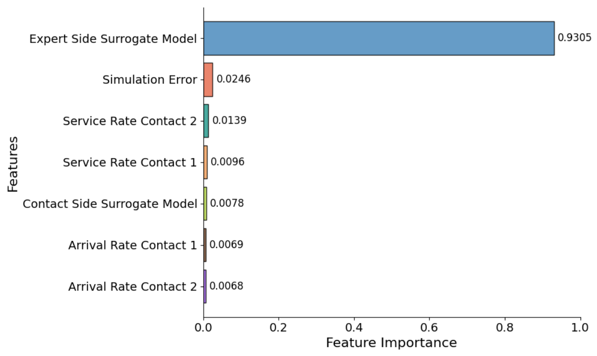
Table 2 summarizes the results of the two experiments. The results highlight several noteworthy patterns regarding the impact of uncertainty quantification on confidence interval construction. First, the conventional approach in both the frequentist and Bayesian settings yields relatively narrow CIs that fail to contain the estimated "true" expected average waiting time of 1.07 minutes. This indicates that the conventional method underestimates the variability in the simulation output which suggests that decisions based solely on the conventional approach may be overly optimistic and potentially misleading. The second and third approaches address this limitation by incorporating input uncertainty and both input and surrogate uncertainty, respectively. In both the frequentist and Bayesian settings, these expanded uncertainty quantification strategies lead to wider intervals, with those from the third approach being the widest. In the Bayesian case, this effect is even more pronounced and the interval from the third approach is not only the widest but also the most conservative. Although wider intervals can sometimes be seen as less informative, in this context they alert the decision-maker to the substantial uncertainty introduced by input and surrogate modeling.

To further analyze the contribution of each source of uncertainty, we train a decision tree regressor on the datasets generated from the third cases. We then compute and plot the feature importance scores for both frequentist and Bayesian approaches in Figure 2. The analysis reveals that the most significant source of uncertainty comes from the expert-triggered routing model. This indicates that our estimation of the expected average waiting time is heavily influenced by the model used to approximate expert-side routing. In contrast, the contact-triggered routing model contributes much less than the expert-triggered routing model. Based on these results, one could consider collecting additional data for training the expert-side surrogate model or selecting a more accurate surrogate model to enhance accuracy and reduce uncertainty in the estimation of the expected average waiting time. Moreover, the importance plots suggest that increasing the number of simulation replications or improving the accuracy of input model estimation would not be enough to effectively shrink the confidence interval.

We conclude this section with a brief discussion on how one might choose the number of stacks, S. In the second and third cases of the first experiment, for example, we generated a total of 125 design points and ran 10 replications at each point, resulting in BSn observations. Figure 3 shows a plot where the number of stacks, S, is plotted on the x-axis and the estimated importance scores of each source of uncertainty are plotted on the y-axis. This visualization allows us to assess the stability of the variable importance scores as S increases. If the plot exhibits large oscillations—especially in the ranking of the most influential sources—it suggests that more stacks are needed to obtain stable estimates. We observe that after approximately five stacks, the variability in the importance scores diminishes substantially, and the dominant sources of uncertainty become clearly identifiable. This indicates that a modest number of stacks may suffice for effective uncertainty attribution in this setting.



(a) Using bootstrapping.



(b) Using Bayesian model averaging.

Figure 2: Feature importance plots from fitting decision trees for the frequentist and Bayesian settings.

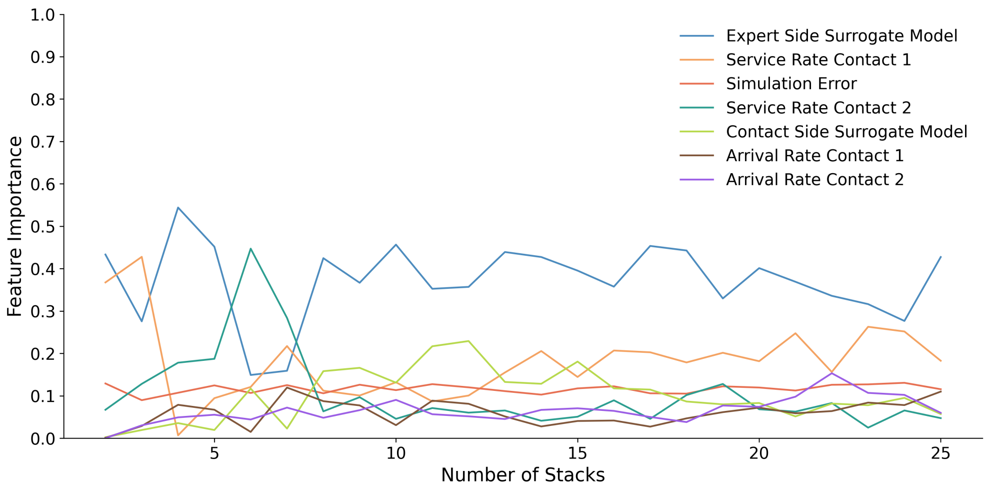


Figure 3: Importance scores of each source of uncertainty with respect to the number of stacks, S.

7 CONCLUSION

This paper introduces a framework for quantifying the uncertainty introduced by ML surrogate models embedded in simulation models. We leverage bootstrapping and Bayesian model averaging to construct quantile-based CIs that capture the variability from both input and surrogate models. We further decompose this uncertainty using regression-tree-based variable importance metrics to identify which sources contribute most to output variability. Through a contact-center simulation experiment, we demonstrate how conventional methods can significantly underestimate uncertainty and that accounting for surrogate model uncertainty can yield more reliable CIs. Our results underscore the importance of selecting flexible surrogate models and having sufficient training data and provide actionable insights for decision-makers seeking to allocate resources to reduce model risk.

ACKNOWLEDGMENTS

This work was supported by the Intuit University Collaboration Program. We also thank Aditya Sangave for his coding assistance and Shashank Rao for his support throughout the project.

REFERENCES

Andrés-Thió, N., M. A. Muñoz, and K. Smith-Miles. 2024. “Methodology and Challenges of Surrogate Modelling Methods for Multi-Fidelity Expensive Black-Box Problems”. *The ANZIAM Journal* 66(1):35–61.

Barton, R. R., H. Lam, and E. Song. 2022. “Input Uncertainty in Stochastic Simulation”. In *The Palgrave Handbook of Operations Research*, 573–620. Cham, Switzerland: Springer.

Chen, W., M. Tanneau, and P. Van Hentenryck. 2024. “Real-Time Risk Analysis with Optimization Proxies”. *Electric Power Systems Research* 235(1):110822.

Chick, S. E. 2001. “Input Distribution Selection for Simulation Experiments: Accounting for Input Uncertainty”. *Operations Research* 49(5):744–758.

Efron, B., and R. J. Tibshirani. 1994. *An Introduction to the Bootstrap*. Chapman & Hall/CRC.

Gendreau, M., A. Hertz, and G. Laporte. 1999. “A Tabu Search Heuristic for the Vehicle Routing Problem”. *Management Science* 45(9):1276–1290.

Hastie, T., R. Tibshirani, and J. Friedman. 2009. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2 ed. Springer Series in Statistics. New York: Springer.

James, G., D. Witten, T. Hastie, and R. Tibshirani. 2021. *An Introduction to Statistical Learning: With Applications in R*. 2 ed. Springer Texts in Statistics. New York: Springer.

Lahiri, S. N. 2003. *Resampling Methods for Dependent Data*. Springer Series in Statistics. New York: Springer.

Lam, H., and H. Qian. 2022. “Subsampling to Enhance Efficiency in Input Uncertainty Quantification”. *Operations Research* 70(3):1891–1913.

Mentch, L., and W. Zhou. 2020. “Randomization as Regularization: A Degrees of Freedom Explanation for Random Forests”. *Journal of Machine Learning Research* 21(202):1–39.

Morgan, L. E., B. L. Nelson, A. C. Titman, and D. J. Worthington. 2019. “Detecting Bias Due to Input Modelling in Computer Simulation”. *European Journal of Operational Research* 279(3):869–881.

Neal, R. M. 2012. *Bayesian Learning for Neural Networks*. Ph. D. thesis, University of Toronto.

Nelson, B., and L. Pei. 2024. *Foundations and Methods of Stochastic Simulation*. 2 ed, Volume 187 of *International Series in Operations Research & Management Science*. New York: Springer.

Politis, D. N., and J. P. Romano. 1994. “The Stationary Bootstrap”. *Journal of the American Statistical Association* 89(428):1303–1313.

Popp, L. 2023. “Surrogate Models for Black-Box Optimization Problems”. B.S. thesis, Technical University of Munich.

Ranftl, S., and W. von der Linden. 2021. “Bayesian Surrogate Analysis and Uncertainty Propagation”. *Physical Sciences Forum* 3.

Rasmussen, C. E., and C. K. I. Williams. 2006. *Gaussian Processes for Machine Learning*. Cambridge, MA: MIT Press.

Santner, T. J., B. J. Williams, and W. I. Notz. 2018. *The Design and Analysis of Computer Experiments*. 2 ed. New York: Springer.

Simio LLC 2023. “Simio Simulation Software”. https://www.simio.com. Version 14.230.

Tang, Z., and T. Westling. 2024. “Consistency of the Bootstrap for Asymptotically Linear Estimators Based on Machine Learning”. *arXiv preprint arXiv:2404.03064*.

Ye, J. 1998. “On Measuring and Correcting the Effects of Data Mining and Model Selection”. *Journal of the American Statistical Association* 93(441):120–131.

AUTHOR BIOGRAPHIES

MOHAMMADMAHDI GHASEMLOO is a PhD student in the Wm Michael Barnes ’64 Department of Industrial and Systems Engineering at Texas A&M University. His research interests lie at the intersection of machine learning and stochastic simulation optimization. His e-mail address is mohammad_ghasemloo@tamu.edu.

DAVID J. ECKMAN is an Assistant Professor in the Wm Michael Barnes ’64 Department of Industrial and Systems Engineering at Texas A&M University. His research interests deal with optimization and output analysis for stochastic simulation models. He is a co-creator of SimOpt, a testbed of simulation optimization problems and solvers. His e-mail address is eckman@tamu.edu.

YAXIAN LI is a Staff AI Scientist at Intuit within the A2D department. Her research interests encompass simulation, optimization, artificial intelligence and machine learning. Her e-mail address is Yaxian_Li@intuit.com.