INFORMS Journal on Computing

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

# Quantifying Input Uncertainty via Simulation Confidence Intervals

Russell R. Barton, Barry L. Nelson, Wei Xie

To cite this article:
Russell R. Barton, Barry L. Nelson, Wei Xie (2014) Quantifying Input Uncertainty via Simulation Confidence Intervals. INFORMS Journal on Computing 26(1):74-87. http://dx.doi.org/10.1287/ijoc.2013.0548

*INFORMS Journal on Computing*
Vol. 26, No. 1, Winter 2014, pp. 74–87
ISSN 1091-9856 (print) | ISSN 1526-5528 (online)

http://dx.doi.org/10.1287/ijoc.2013.0548


# Quantifying Input Uncertainty via Simulation Confidence Intervals

Russell R. Barton
Department of Supply Chain and Information Systems, Smeal College of Business, Pennsylvania State University, University Park, Pennsylvania 16802, rbarton@psu.edu

Barry L. Nelson, Wei Xie
Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, Illinois 60202
{nelsonb@northwestern.edu, WeiXie2013@u.northwestern.edu}

We consider the problem of deriving confidence intervals for the mean response of a system that is represented by a stochastic simulation whose parametric input models have been estimated from "real-world" data. As opposed to standard simulation confidence intervals, we provide confidence intervals that account for uncertainty about the input model parameters; our method is appropriate when enough simulation effort can be expended to make simulation-estimation error relatively small. To achieve this we introduce metamodel-assisted bootstrapping that propagates input variability through to the simulation response via an equation-based model rather than by simulating. We develop a metamodel strategy and associated experiment design method that avoid the need for low-order approximation to the response and that minimizes the impact of intrinsic (simulation) error on confidence level accuracy. Asymptotic analysis and empirical tests over a wide range of simulation effort show that confidence intervals obtained via metamodel-assisted bootstrapping achieve the desired coverage.

*Key words*: input modeling; bootstrapping; confidence intervals; metamodeling; stochastic kriging
*History*: Accepted by Marvin Nakawama, Area Editor for Simulation; received January 2011; revised June 2011, January 2012, October 2012; accepted January 2013. Published online in *Articles in Advance* June 14, 2013.



## 1. Introduction

One of the most valuable aspects of simulation is its ability to characterize the behavior of arbitrarily complex stochastic systems. However, effective characterization depends on controlling simulation-estimation error, because a stochastic simulation is in every sense a statistical experiment. Years of research on measuring and controlling simulation-estimation error has yielded robust methods that are adequate for many practical simulation problems; one might even be tempted to say that the estimation-error problem has been "solved." Unfortunately, there is a hidden, and often substantial additional error in simulation experiments that is present even if best practices for modeling, experiment design, and output analysis are employed: *input-uncertainty error*. In fact, the impact of input-uncertainty error relative to simulation-estimation error can be exacerbated by best practices for controlling estimation error.

Input models are the driving stochastic processes in simulation experiments. They represent uncertainty at a basic level that resists more detailed modeling. Arrival processes in queueing simulations, demand processes in supply chain simulations, and patient occupancy times in hospital simulations are examples of inputs. Input models are often based on a finite sample of observed real-world data, and therefore are subject to error. All simulation software measures simulation-estimation error; no simulation software accounts for input-uncertainty error, and few practitioners are even aware of this problem. Yet, as we demonstrate next, input-uncertainty error can overwhelm simulation-estimation error, placing simulation users at risk of making critical and expensive decisions with unfounded confidence in (what appear to be) highly precise simulation assessments or optimizations.

### 1.1. An Illustration

The steady-state expected number of customers in an $M/M/\infty$ queue (Poisson arrival process with rate $\lambda$, exponentially distributed service times with mean $\tau$, and an infinite number of servers) is $\mu(\lambda, \tau) = \lambda\tau$. To understand how input uncertainty affects simulation, suppose we do not know this result, nor do we know $\lambda$ or $\tau$, but we want to estimate the mean number of customers in the system in steady state via simulation. Consider the following stylized experiment.

1. Start by observing $m$ real-world customer interarrival times $A_1, A_2, \ldots, A_m$ and estimate $\lambda$ by

$\hat{\lambda} = 1/\bar{A}$, where a $\bar{}$ indicates the sample average. Similarly, observe $m$ real-world service times $X_1, X_2, \ldots, X_m$ and estimate $\tau$ by $\hat{\tau} = \bar{X}$. In this illustration and throughout the paper we assume parametric input models but with unknown parameters, as is typical in input modeling. In this case the distributions (exponential) are correct, but the parameters are estimated.

2. Conditional on $\hat{\lambda}$ and $\hat{\tau}$, simulate $n$ independent replications of the queue, and on each one take a single observation of the number of customers in the system in steady state $Y_1, Y_2, \ldots, Y_n$. This is a simplified version of what actually happens in simulation where we record the number in the system for some period of time and average, not just take a single observation.

3. Finally, estimate the steady-state expected number in the queue $\mu(\lambda, \tau)$ by the sample mean $\bar{Y}$.

For this example the properties of $\bar{Y}$ can be derived:

$$\mathrm{E}[\bar{Y}] = \frac{m}{m-1}\lambda\tau, \tag{1}$$

$$\mathrm{Var}[\bar{Y}] \approx \frac{\lambda\tau}{n} + \frac{2(\lambda\tau)^2}{m}. \tag{2}$$

These results integrate over both input and simulation uncertainties. The impact of input uncertainty shows up in two ways: First, the estimator $\bar{Y}$ is biased, as shown in (1). The bias is a function of $m$, the amount of real-world data used to estimate the input models. The variance of the estimator displayed in (2) contains two terms. The first term represents the simulation-estimation error; it decreases as the number of replications $n$ increases. This is the only term that is captured by simulation-based confidence intervals (CIs), and it can be driven to 0 by increasing the number of replications. In fact, in many practical problems it is both feasible and reasonable to make simulation-estimation error negligible. The second term represents the input-uncertainty error. Notice that it may entirely dominate the first term depending upon the amount of real-world data $m$ and it is immune to more simulation effort.

This is a stylized example, but the situation is actually much worse in practical problems. A realistic simulation may have tens of input models, some estimated from huge stockpiles of data and others from very little. The simulation output is often a highly nonlinear function of these inputs, so assessing input-uncertainty error with anything like the previous mathematical analysis is impossible. Most critically, project success, corporate profitability, and even human lives may depend on decisions that are informed by overly confident characterizations of uncertainty.

### 1.2. Quantifying Input Uncertainty

The contributions of this paper are to develop a framework to construct confidence intervals for the mean simulation response that account for input uncertainty in parametric input models, and to provide a specific implementation that makes the framework efficient and effective. Our approach is to propagate input uncertainty from the input models to the simulation output using bootstrap resampling of the real-world data and a metamodel that approximates the mean simulation response as a function of the input models. As described in §2, this approach overcomes shortcomings of other methods that have been proposed. The choice of metamodel form in conjunction with an experiment design strategy tailored to this application are also contributions. The result is a framework for quantifying input uncertainty that is rigorously justified, but also practically useful.

In this paper we only consider the case of parametric input models that describe independent and identically distributed (i.i.d.) inputs and input models that are mutually independent. We assume we know the correct distribution families, but not their parameters, and that we have real-world data from which to estimate the parameters. This is a step toward solving the problem in which the families of distributions themselves are also unknown. It is a very important step, however, because the existence of a "true, correct" distribution for real processes is always fiction; any distribution is an approximation, and to account for other possible parameters that could have been chosen for the approximation is significant.

The next section describes other approaches to the input-uncertainty problem; this is followed by a formal description of the problem and development of the metamodel-assisted bootstrapping concept in §3. Implementation is considered in §4. We report results from an empirical evaluation in §5, and conclude the paper in §6.

## 2. Background

Simulation input-uncertainty has been addressed in the literature in essentially two ways: The first is to perform statistical output characterization conditional on the correctness of the input probability models and separately perform sensitivity, uncertainty, or robustness analysis of simulation output to changes in simulation-input distributions. Descriptions of this approach can be found in Kleijnen (1987, 1994) and Law and Kelton (2000). As noted by Schmeiser in his discussion in Barton et al. (2002), this method has broad applicability, and can handle qualitative uncertainty in model form as well as situations where no calibrating real-world data exist. This separate analysis approach has a significant drawback: it provides no statistical characterization of input uncertainty.

The second approach is to combine explicit characterization of the impact of input uncertainty with the measurement of output variability to provide confidence intervals for output parameter values. There have been three primary streams of research in this vein.

One stream of research has developed methods for a priori known families of parametric input distributions, using maximum likelihood estimator properties of the parameter estimates and Taylor series approximations of the expected simulation response, and assuming that the correct parametric families of the input distributions are known. Cheng (1994) and Cheng and Holland (1997) used a delta-method approximation to achieve this, which requires an additional number of simulation runs that is linear in the number of input model parameters. Cheng and Holland (1998, 2004) suggested an alternative method that requires only two runs, but yields more conservative confidence intervals. In many cases, the linear approximation of the impact of input on output that these methods require cannot be supported. Our approach, metamodel-assisted bootstrapping, directly accounts for nonlinearity.

A second stream of research is based on Bayesian methods. The distribution of the expected value of the simulation output is characterized by running simulations at each of many repeated samplings from a posterior distribution for the parametric input model parameters. The posterior distributions are determined by applying Bayes' rule to a prior distribution and observations of real-world data. This Bayesian model averaging (BMA) strategy has been described and refined by Chick (1997, 1999, 2000, 2001), Chick and Ng (2002), Ng and Chick (2001), Zouaoui and Wilson (2001a), and Zouaoui and Wilson (2001b, 2003). These methods require normality and homogeneity assumptions on the impact of input uncertainty that may not be tenable. Zouaoui and Wilson (2004) used a different BMA approach to capture both parameter uncertainty and uncertainty across an a priori determined set of possible parametric families. The assumption of homogeneous variance induced by input variability remains, and the $t$-type confidence interval they describe also depends on approximate normality because of input uncertainty. Biller and Corlu (2011) extended the Bayesian framework to handle a large number of correlated inputs.

A third stream of research takes a frequentist approach, characterizing the impact of input uncertainty on the simulation output using direct sampling and bootstrap resampling methods. This work includes the papers on nonparametric bootstrap approaches by Barton and Schruben (1993, 2001) and Barton (2007), and a parametric bootstrap approach by Cheng and Holland (1997). These percentile intervals do not require normality or homogeneity because of input uncertainty. Unfortunately, because the bootstrap statistic is the output of a stochastic simulation, it is not a smooth function of the input data; this violates a requirement for asymptotic consistency of a bootstrap confidence interval, which is one of the central issues that metamodel-assisted bootstrapping addresses.

Further, the sampling methods in this third stream and the percentile method in Zouaoui and Wilson (2004) suffer from a failure to distinguish the behavior of output variability and input variability. Schmeiser in his discussion in Barton et al. (2002, p. 363) noted "Any reasonable version [of a method capturing both input-uncertainty error and simulation-estimation error] needs to reflect the fundamental difference that sampling error decreases with additional simulation sampling while additional sampling has no impact on modeling error." This means that confidence intervals based on resampling strategies will have coverage that exceeds the nominal value if the simulation effort for each resample is relatively low. Barton identified this error for nonparametric bootstrap methods in Barton et al. (2002) and proposed simulation effort guidelines in Barton (2007).

There has been related work on allocation-of-effort strategies to minimize the joint impact of input uncertainty and simulation-output uncertainty. Lee and Glynn (2003) developed a framework to estimate the distribution of the conditional expectation of a simulation-output statistic in the presence of model and parameter uncertainty, and presented methods to minimize the mean squared error of an estimator of the expected value. Steckley and Henderson (2003) further developed the framework and described a kernel approach for estimating the density of the conditional expectation depending on a single unknown parameter. Ng and Chick (2001) used the delta method in a Bayesian framework to sequentially choose which real-world data to augment with additional samples to reduce the variance of the simulation output. Freimer and Schruben (2002) described two approaches to determine whether additional real-world data should be collected, given a fixed level of simulation effort. In the first approach, upper and lower confidence limits on model parameters are used as parameter values in a factorial set of simulation experiments. If the parameter effects are statistically significant, then more real-world data must be collected. In the second approach, the real-world data are bootstrap resampled to generate random values of the parameters, and a random-effects model is used to test for significance of the parameter uncertainty.

Henderson (2003) gives a detailed review and critique of these approaches. He found no clearly superior method, and suggested the need for a method

that is transparent, statistically valid, implementable, and efficient. In this paper we describe a metamodel-assisted bootstrapping method and show its use with parametric input models. This approach addresses many of the shortcomings in the prior work: The use of a general-form metamodel can provide a higher fidelity approximation than a first-order Taylor approximation and makes the bootstrap statistic a smooth function of the input data. Further, the use of a metamodel reduces the impact of intrinsic simulation error on the accuracy of the confidence intervals that we construct.

In Barton et al. (2010) we conducted an empirical study comparing metamodel-assisted bootstrapping against the alternatives using two test cases with special structure. The results provided compelling evidence that metamodel-assisted bootstrapping is both effective and superior to competitors, motivating its extension in this paper to more general input-uncertainty problems.

The two test cases in Barton et al. (2010) were $M/M/\infty$ and $M/M/1/q$ queues. We compared metamodel-assisted bootstrapping to the standard conditional confidence interval (i.e., ignoring input uncertainty), direct bootstrapping, Bayesian bootstrapping, and a fully Bayesian analysis using noninformative priors; detailed algorithms are given in Barton et al. (2010). Factors we varied included the quantity of real-world data, simulation budget, and number of bootstrap/posterior resamples.

Not surprisingly, the coverage of the conditional CIs was far below nominal level and decreased with greater simulation budget. As has been noted in the literature, the direct bootstrap, Bayesian bootstrap, and Bayesian methods could have substantial overcoverage (for instance, we observed 100% coverage in 1,000 trials when the nominal level was 95%). Although overcoverage is certainly better than undercoverage, overcoverage is not harmless as it misleads the analyst into thinking that there is more uncertainty than there really is.

Metamodel-assisted bootstrapping was robust, providing coverage right at the nominal level across all experimental settings. However, it is important to note that because all of the input distributions in these examples were exponential and thus had a *single* parameter (the mean), metamodeling is relatively straightforward. The open questions remaining from Barton et al. (2010) are what form should the metamodel take, and what is an effective experiment design to fit the metamodel, for general multiparameter distributions? Answers to these questions, along with a proof of validity of the bootstrap approach, are the subject of the remainder of the paper. We will not include the competitors in this paper because they were already shown to be inferior in Barton et al. (2010).

## 3. A Framework for Confidence Intervals

To account for input uncertainty, we introduce the following representation of a computer simulation.

A simulation is a function $g\colon \omega \to Y$, where $\omega$ is an elementary outcome from a probability space $(\Omega, P)$; to be concrete, one might think of $\omega$ as a realization of an infinite sequence of i.i.d. uniform $(0,1)$ random numbers (although only a finite subset will be used in the simulation). The mapping $g$ is also a functional of $L$ input distributions, say $\{F_1, F_2, \ldots, F_L\}$. Thus, $g(\omega) = g(\omega \mid F_1, F_2, \ldots, F_L)$. In this paper we assume that the different input processes are independent. We represent the simulation in this way to indicate that different choices are possible for the driving input processes, and we distinguish $L$ distributions because the typical simulation is driven by i.i.d. samples from one or more distinct distributions and "input modeling" involves characterizing each of these distributions separately. For instance, in a queueing simulation, $F_1$ might be the distribution of the interarrival times of a stationary arrival process, and $F_2$ could be the distribution of service times. In this paper, we only consider parametric input models. The parameters of the $l$th input distribution are denoted by $\mathbf{x}_l$, a $p_l \times 1$ vector. The collection of all of the distribution parameters is denoted $\mathbf{x}^\top = (\mathbf{x}_1^\top, \mathbf{x}_2^\top, \ldots, \mathbf{x}_L^\top)$, facilitating the equivalent but more concise representation $Y = g(\omega \mid \mathbf{x})$. We discuss our choice of parameterization next.

In addition, the simulation output often depends on other parameters describing the structure of the model or the experiment, such as the number of servers, the "feeds and speeds" of the equipment, or the simulation stopping time, but this dependence is omitted from the notation and not a part of our framework. We assume that the objective is to characterize system behavior given a fixed set of these controllable parameters.

Our interest is in the mean response

$$\mu(\mathbf{x}) = \int g(\omega \mid \mathbf{x})\, dP,$$

and in particular, the mean response at the true, correct parameters $\mathbf{x}^c$, which are not known and must be estimated from real-world data. Assuming the existence of $\mathbf{x}^c$, our goal in this paper is to obtain random bounds $C_L$ and $C_U$ such that

$$\Pr\{\mu(\mathbf{x}^c) \in [C_L, C_U]\} \geq 1 - \alpha. \tag{3}$$

The approach we take exploits a metamodel that predicts the mean response $\mu(\mathbf{x})$ as a function of the parameters $\mathbf{x}$. Because we will use a spatial correlation metamodel, we need to define "space" so that input distributions that are "close" to each other lead

to similar simulation output. A natural way to measure the distance between two parametric distributions is as a function of the difference between their parameters. For instance, a gamma($\alpha$, $\beta$) distribution has a shape parameter $\alpha$ and scale parameter $\beta$, so the distance between two gamma distributions could be of the form $\theta_1(\alpha_i - \alpha_j)^2 + \theta_2(\beta_i - \beta_j)^2$, where $(\theta_1, \theta_2)$ reflect the importance of differences in each dimension. Clearly when this distance is small, the gamma distributions will yield probabilistically similar inputs. Nevertheless, there are a number of reasons why we do not measure the distance between input distributions as a function of the natural model parameters.

To build metamodels we will run simulation experiments that cover the relevant parameter space. But even two-parameter distributions are not always parameterized by "shape" and "scale." Therefore, if we use the natural parameters then distance between two, say, time-to-failure distributions might be measured differently than between two service-time distributions. More importantly, to fit a metamodel we need to define a relevant space to cover. The feasible parameter space for most parametric distributions is enormous (e.g., $\alpha > 0$ for the gamma) and would require far too many experiments to cover completely. So instead we want an experiment design strategy that adapts to the real-world data we have, and the possible bootstrap resamples it could generate. The natural parameters of many distributions are complex nonlinear functions of the data, making it difficult or costly to deduce the relevant range.

In this paper, we measure the distance between distributions as a function of the distance between the first few moments (or standardized moments) of the distribution. We assume that a $p_l$-parameter distribution is uniquely specified by its first $p_l$ moments, which is true for the distributions that are most often used in stochastic simulation. We are exploiting the fact that when the parametric family of distributions is known and fixed then the values of its moments completely specify it, and when the moments are close, then the distributions will be similar. Thus, the independent variable $\mathbf{x}$ is the concatenation of the first $p_l$ moments of all input distributions $l = 1, 2, \ldots, L$. The dimension of $\mathbf{x}$ is $d = \sum_{l=1}^{L} p_l$.

The output from a single simulation replication can be written as $Y(\mathbf{x}, \omega) = \mu(\mathbf{x}) + \varepsilon(\mathbf{x}, \omega)$, where $\mu(\mathbf{x}) = \int g(\omega \mid \mathbf{x})\, dP$ and $\varepsilon(\mathbf{x}, \omega) = g(\omega \mid \mathbf{x}) - \mu(\mathbf{x})$. From here on we drop the dependence of these random variables on $\omega$ for notational convenience, giving the random output

$$Y(\mathbf{x}) = \mu(\mathbf{x}) + \varepsilon(\mathbf{x}). \tag{4}$$

When independent replications are obtained (from independent realizations $\omega_j$ of $\omega$) we append a subscript $j$ to indicate the $j$th replication as in $Y_j(\mathbf{x}) = \mu(\mathbf{x}) + \varepsilon_j(\mathbf{x})$.

For many simulation settings the output is an average of a large number of more basic outputs, so a version of the central limit theorem implies that the distribution of $\varepsilon(\mathbf{x})$ (and consequently $Y(\mathbf{x})$) is approximately Gaussian. Simulation outputs in this category include continuous-time-average statistics such as utilization and discrete-time-average statistics such as average waiting time. If $\mathbf{x}^c$ were known then the desired CI could be obtained from classical statistics using simulated outputs $\{Y_j(\mathbf{x}^c), j = 1, 2, \ldots, n\}$ from a set of $n$ replicated simulation runs.

Of course, $\mathbf{x}^c$ is typically unknown and must be estimated using finite real-world samples, and the number of observations from each of the $L$ input distributions might differ. Let $m_l$ be the number of i.i.d. observations from $l$th input distribution, and let the observed values be $\mathbf{Z}_{l,m_l} = \{Z_{l,1}, Z_{l,2}, \ldots, Z_{l,m_l}\}$. Let $\mathcal{Z}_{\mathbf{m}} = \{\mathbf{Z}_{l,m_l}, l = 1, 2, \ldots, L\}$ be the collection of observations from all $L$ input distributions, where $\mathbf{m} = (m_1, m_2, \ldots, m_L)$. The $p_l \times 1$ vector of moment estimators for the $l$th process is a function of this data, $\hat{\mathbf{X}}_{l,m_l} = \hat{\mathbf{X}}_l(\mathbf{Z}_{l,m_l})$. Combining moment estimators from all processes, we obtain a $d \times 1$ moment vector $\hat{\mathbf{X}}_{\mathbf{m}}^{\top} = (\hat{\mathbf{X}}_{1,m_1}^{\top}, \ldots, \hat{\mathbf{X}}_{L,m_L}^{\top})$.

Simulation-output analysis is usually based on a particular realization of $\mathcal{Z}_{\mathbf{m}}$, say $\mathbf{z}_{\mathbf{m}}^{(0)}$, giving a corresponding moment (parameter) estimate $\hat{\mathbf{x}}_{\mathbf{m}}^{(0)}$. Standard simulation-output analysis constructs a confidence interval that is conditioned on $\hat{\mathbf{x}}_{\mathbf{m}}^{(0)}$; that is, it forms random bounds $C_L$ and $C_U$ such that

$$\Pr\{\mu(\hat{\mathbf{x}}_{\mathbf{m}}^{(0)}) \in [C_L, C_U] \mid \hat{\mathbf{X}}_{\mathbf{m}}^{(0)} = \hat{\mathbf{x}}_{\mathbf{m}}^{(0)}\} \geq 1 - \alpha. \tag{5}$$

The probability in the conditional CI is with respect to $\omega$; the randomness in $\hat{\mathbf{x}}_{\mathbf{m}}^{(0)}$ is ignored. Thus, the bounds provide a probability guarantee of covering $\mu(\hat{\mathbf{x}}_{\mathbf{m}}^{(0)})$ rather than $\mu(\mathbf{x}^c)$. As has been shown in many papers, this can lead to coverage probabilities for $\mu(\mathbf{x}^c)$ that are far from $1 - \alpha$. Our goal is to obtain asymptotically correct confidence intervals for $\mu(\mathbf{x}^c)$ as the amount of real-world data increases, under the assumptions that the real-world input distributions are stable and the model logic is correct. Clearly the real-world distributions must be unchanging for the probability statement (3) to make sense.

Our focus on a CI for $\mu(\mathbf{x}^c)$ avoids the problem of correcting for the bias of $\mu(\hat{\mathbf{x}}_{\mathbf{m}}^{(0)})$ as an estimator of $\mu(\mathbf{x}^c)$, as illustrated in §1.1; instead we bound the value of $\mu(\mathbf{x}^c)$ with high probability and let this interval account for the bias.

How can we obtain a CI like (3) rather than one like (5)? Suppose, as a thought experiment, that we could obtain i.i.d. observations of size $\mathbf{m}$ of the real-world process, $\mathcal{Z}_{\mathbf{m}}^{(0)}, \mathcal{Z}_{\mathbf{m}}^{(1)}, \mathcal{Z}_{\mathbf{m}}^{(2)}, \ldots$, and let $\hat{\mathbf{X}}_{\mathbf{m}}^{(i)}$ denote the corresponding moment estimator obtained from

the $i$th set. This makes it easy to see that the distribution of the simulation-output $Y(\hat{\mathbf{X}}_{\mathbf{m}}^{(i)})$ has two sources of randomness: the simulation output variability conditional on $\hat{\mathbf{X}}_{\mathbf{m}}^{(i)}$, and the randomness from the input variability of $\mathcal{Z}_{\mathbf{m}}^{(i)}$ used in constructing $\hat{\mathbf{X}}_{\mathbf{m}}^{(i)}$. Specifically,

$$\begin{aligned} Y(\hat{\mathbf{X}}_{\mathbf{m}}^{(i)}) &= \mu(\hat{\mathbf{X}}_{\mathbf{m}}^{(i)}) + \varepsilon(\hat{\mathbf{X}}_{\mathbf{m}}^{(i)}) \\ &= \mu(\mathbf{x}^c) + [\mu(\hat{\mathbf{X}}_{\mathbf{m}}^{(i)}) - \mu(\mathbf{x}^c)] + [Y(\hat{\mathbf{X}}_{\mathbf{m}}^{(i)}) - \mu(\hat{\mathbf{X}}_{\mathbf{m}}^{(i)})] \\ &= \mu(\mathbf{x}^c) + D + V. \end{aligned} \tag{6}$$

Now suppose that the function $\mu(\cdot)$ were known, as well as the distribution of $D$, denoted $F_D$. Then if we observed only one real-world sample $\mathcal{Z}_{\mathbf{m}}^{(0)}$, the random interval $C_U = \mu(\hat{\mathbf{X}}_{\mathbf{m}}^{(0)}) - F_D^{-1}(\alpha/2)$ and $C_L = \mu(\hat{\mathbf{X}}_{\mathbf{m}}^{(0)}) - F_D^{-1}(1-\alpha/2)$ achieves objective (3) because

$$\begin{aligned} &\Pr\{C_L \le \mu(\mathbf{x}^c) \le C_U\} \\ &\quad = \Pr\{F_D^{-1}(\alpha/2) \le \mu(\hat{\mathbf{X}}_{\mathbf{m}}^{(0)}) - \mu(\mathbf{x}^c) \le F_D^{-1}(1-\alpha/2)\} \\ &\quad = 1-\alpha, \end{aligned}$$

because $\mu(\hat{\mathbf{X}}_{\mathbf{m}}^{(0)}) - \mu(\mathbf{x}^c)$ has distribution $F_D$. Of course, neither $\mu(\cdot)$ nor $F_D$ are known, but this insight suggests a path to a CI:

- Use simulation experiments to build a metamodel $\hat{\mu}(\cdot)$ to stand in for $\mu(\cdot)$.
- Let bootstrap resampling stand in for repeated observations of the real-world data. More specifically, for the $l$th input process, $l = 1, 2, \ldots, L$, draw $m_l$ random samples with replacement from $\mathbf{Z}_{l,m_l}^{(0)}$; repeat this procedure $B$ times to obtain $B$ bootstrap samples and corresponding moment estimators $\hat{\mathbf{X}}_{\mathbf{m}}^{(b)}$, $b = 1, 2, \ldots, B$. Approximate the distribution $F_D$ of $\mu(\hat{\mathbf{X}}_{\mathbf{m}}^{(i)}) - \mu(\mathbf{x}^c)$ by the empirical distribution of $\hat{\mu}(\hat{\mathbf{X}}_{\mathbf{m}}^{(b)}) - \hat{\mu}(\hat{\mathbf{X}}_{\mathbf{m}}^{(0)})$.

Building a metamodel for the mean simulation response as a function of the input moments is key to our framework, and building an accurate one is essential to making it work. By running a designed experiment in the input moment space we leverage more information about the response surface $\mu(\cdot)$, and the impact of distribution choice on output performance, than we would have from a single simulation at $\hat{\mathbf{x}}_{\mathbf{m}}^{(0)}$. Fitting a metamodel allows us to spend the simulation budget carefully across a relatively small number of well-chosen design points providing precise estimates not only at those design points, but throughout the mean response surface. As we show empirically, this effectively eliminates the impact of $V$. Using a metamodel $\hat{\mu}(\cdot)$ also provides the continuity conditions that support the asymptotic validity of bootstrapping, which is that the bootstrap statistic be a smooth function of the input data. Of course, the challenges are designing the experiment in input-model space, in the choice of metamodel, and in proving that it works. These are the primary contributions of this paper.

An alternative to creating a metamodel is running separate simulation experiments at each of the bootstrap input moments $\hat{\mathbf{X}}_{\mathbf{m}}^{(b)}$, $b = 1, 2, \ldots, B$. Because, for accuracy of the bootstrap CI, $B$ is recommended to be at least 1,000, this spreads the simulation budget across a relatively large number of bootstrap experiments, leading to imprecise estimates of the mean response at those points. Metamodeling leverages all of the data from a designed experiment to provide better predictions of the response at each bootstrap point.

## 4. Metamodel-Assisted Bootstrapping

Next we give an overview of our approach, hinting at the particular (and important) implementation details we describe in the sections that follow. Within this framework other implementation choices are possible, although we will make arguments in favor of our selections.

1. Input: A set of real-world data $\mathbf{z}_{\mathbf{m}}^{(0)}$.
2. Construct metamodel $\hat{\mu}(\mathbf{x})$: Select an experiment design, run simulations at the design points, and fit a metamodel $\hat{\mu}(\mathbf{x})$ as an approximation to the true mean response $\mu(\mathbf{x})$ as a function of the input moments $\mathbf{x}$. This requires two choices:
   a. *Metamodel form*. We choose stochastic kriging, which is reviewed in §4.1, because it does not require strong assumptions about the response surface $\mu(\mathbf{x})$, and it is specifically intended to account for output variability in stochastic simulation.
   b. *Experiment design*. The design points $\mathbf{x}_i$, $i = 1, 2, \ldots, k$ fill the space of most likely bootstrap resample moments. To locate this region, we generate a preliminary bootstrap resample from the real-world data $\mathbf{z}_{\mathbf{m}}^{(0)}$ whose sole purpose is to map the relevant design space (see §4.2).
3. Construct confidence interval: Generate independent bootstrap resamples $\mathcal{Z}_{\mathbf{m}}^{(b)} \sim \mathbf{z}_{\mathbf{m}}^{(0)}$ and corresponding sample moments $\hat{\mathbf{X}}_{\mathbf{m}}^{(b)}$, $b = 1, 2, \ldots, B$, and let the confidence interval be the $\alpha/2$ and $1-\alpha/2$ sample quantiles of

$$\{\hat{\mu}(\hat{\mathbf{X}}_{\mathbf{m}}^{(1)}), \hat{\mu}(\hat{\mathbf{X}}_{\mathbf{m}}^{(2)}), \ldots, \hat{\mu}(\hat{\mathbf{X}}_{\mathbf{m}}^{(B)})\}.$$

That is, we use a bootstrap percentile interval. The validity of the approach is discussed in §4.3.

### 4.1. Stochastic Kriging

At the heart of our approach is a metamodel that predicts the mean simulation response as a function of the moments of the input models that drive the simulation. Consider the examples used in Barton et al. (2010): Because the input models are a Poisson arrival process and exponentially distributed service times, the arrival rate $\lambda$ and the mean service time $\tau$ provide complete characterizations. For the $M/M/\infty$

queue the steady-state expected number in queue is $\mu(\lambda, \tau) = \lambda\tau$, and for the $M/M/1/q$ queue it is

$$\mu(\lambda, \tau) = \frac{\lambda\tau}{1-\lambda\tau} - \frac{(q+1)(\lambda\tau)^{q+1}}{1-(\lambda\tau)^{q+1}}.$$

Notice that even restricting ourselves to Markovian queueing models, these two response functions are quite different. To be general, we need a metamodeling framework that neither makes strong assumptions about the response surface (such as being low-order polynomial), nor is tied to particular input distributions (such as exponential). Our proposal is to use stochastic kriging for the metamodel, and input-distribution moments as the independent variables.

Kriging is an interpolation-based method that has been widely used for the design and analysis of deterministic computer experiments (Santner et al. 2003). Differing from computer experiments with deterministic response, the outputs from stochastic simulation include intrinsic output variability, and this variability often changes significantly across the design space. We use the term "intrinsic" uncertainty or variability to refer to the variability inherent in the sampling that generates stochastic simulation output, as opposed to "extrinsic" uncertainty (introduced later) that refers to our lack of knowledge about the mean response surface. To meet the requirements of stochastic simulation, stochastic kriging (SK) was introduced by Ankenman et al. (2010). Whereas kriging metamodels assume that there is no error at the design points, SK appropriately weights the contributions of the design points depending on their precision. In this section we provide enough background on SK to see why we chose it and how we use it.

In SK, Ankenman et al. (2010) represent the simulation output on replication $j$ at design point $\mathbf{x}$ as

$$Y_j(\mathbf{x}) = \mathbf{f}(\mathbf{x})^T\boldsymbol{\beta} + W(\mathbf{x}) + \varepsilon_j(\mathbf{x}). \tag{7}$$

The independent variable $\mathbf{x} = (x_1, x_2, \ldots, x_d)$ is interpreted as a location in space and the variation in the simulation response is divided into three uncorrelated parts: a trend model $\mathbf{f}(\mathbf{x})^T\boldsymbol{\beta}$, extrinsic (response-surface) uncertainty $W(\mathbf{x})$, and intrinsic (simulation-output) uncertainty $\varepsilon_j(\mathbf{x})$.

When we have no process physics to justify a parametric trend model—which is the situation we assume—best practice in kriging and stochastic kriging is to use only a constant trend $\mathbf{f}(\mathbf{x})^T\boldsymbol{\beta} = \beta_0$. Including a trend model, if appropriate, would not change our approach.

Uncertainty about the response surface is modeled by a mean 0, second-order stationary Gaussian random field $W$, which accounts for the spatial dependence. Loosely, a Gaussian random field is a function from $\Re^d$ to $\Re$ such that any finite collection of $W(\mathbf{x}_1), W(\mathbf{x}_2), \ldots, W(\mathbf{x}_k)$ has a multivariate normal distribution.

A parametric form for the spatial covariance of $W(\mathbf{x})$ is typically assumed:

$$\boldsymbol{\Sigma}(\mathbf{x}, \mathbf{x}') = \mathrm{Cov}[W(\mathbf{x}), W(\mathbf{x}')] = \tau^2 r(\mathbf{x}-\mathbf{x}' \mid \boldsymbol{\theta}),$$

where $r$ is a correlation function that depends on some unknown parameters $\boldsymbol{\theta}$ and $\tau^2$ is the variance. Stationarity implies that the response-surface correlation depends only on $\mathbf{x}-\mathbf{x}'$ and not the actual location of the design points.

Based on the assumed smoothness of the response, different correlation functions can be chosen. The product-form Gaussian correlation function is the most commonly used in practice, as well as the one we use in this paper:

$$r(\mathbf{x}-\mathbf{x}' \mid \boldsymbol{\theta}) = \exp\left\{\sum_{j=1}^{d} \theta_j (x_j - x_j')^2\right\}. \tag{8}$$

To generate a global predictor, we choose $k$ design points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k$, and then run $n_i$ replications at the $i$th point. The sample mean of the simulation outputs at $\mathbf{x}_i$ is $\bar{Y}(\mathbf{x}_i) = \sum_{j=1}^{n_i} Y_j(\mathbf{x}_i)/n_i$, and the vector of sample means at all design points is denoted by $\bar{\mathbf{Y}} = (\bar{Y}(\mathbf{x}_1), \bar{Y}(\mathbf{x}_2), \ldots, \bar{Y}(\mathbf{x}_k))^T$. The variances of the simulation-output uncertainty $\varepsilon$ at different $\mathbf{x}$'s are typically not equal. Let $\sigma^2(\mathbf{x}) = \mathrm{Var}[\varepsilon(\mathbf{x})]$. Because the outputs at different $\mathbf{x}$'s are independent,[1] the diagonal matrix $\mathbf{C}$ of intrinsic output variances can be written as

$$\mathbf{C} = \begin{bmatrix} \sigma^2(\mathbf{x}_1)/n_1 & & \\ & \ddots & \\ & & \sigma^2(\mathbf{x}_k)/n_k \end{bmatrix}.$$

Let $\boldsymbol{\Sigma}$ represent the $k \times k$ covariance matrix of $\boldsymbol{\Sigma}(\mathbf{x}_i, \mathbf{x}_j)$ across all design points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k$, and let $\boldsymbol{\Sigma}(\mathbf{x}, \cdot)$ be the $k \times 1$ covariance vector between prediction point $\mathbf{x}$ and all design points. Then the minimum mean squared error (MSE) linear predictor (Ankenman et al. 2010) is

$$\hat{\mu}(\mathbf{x}) = \beta_0 + \boldsymbol{\Sigma}(\mathbf{x}, \cdot)^\top [\boldsymbol{\Sigma} + \mathbf{C}]^{-1} (\bar{\mathbf{Y}} - \beta_0 \cdot \mathbf{1}_{k\times 1}). \tag{9}$$

Notice that the prediction at $\mathbf{x}$ is the weighted average of the observations, with the design points closer to the prediction point and with smaller output variance having larger weight.

Because the constant $\beta_0$ and spatial correlation parameters $\tau^2$ and $\boldsymbol{\theta}$ are unknown, maximum likelihood estimates are typically used for prediction. SK

[1] Chen et al. (2012) showed that the use of common random numbers inflates the mean squared error of prediction for stochastic kriging, so we use independent simulations.

employs an estimate of $\mathbf{C}$ using sample variances $S^2(\mathbf{x}_i)$. To facilitate experiment design, SK also fits a variance model $\sigma^2(\mathbf{x}) = \sigma^2 + V(\mathbf{x})$, where $V(\mathbf{x})$ is an independent Gaussian random field. See Ankenman et al. (2010) for details.

### 4.2. Experiment Design

In this section, we describe an experiment design strategy for fitting a stochastic kriging metamodel $\hat{\mu}(\cdot)$ so that it can accurately estimate the mean simulation response at bootstrap resample moments.

The process of bootstrapping involves three steps: (i) resampling with replacement from the original sample data, (ii) computing the statistic under study from the $b$th set of bootstrap resampled data, and (iii) characterizing the empirical distribution of the $B$ values of the bootstrapped statistic to yield confidence intervals. In our case the resampled data are themselves moments, and the computed statistic is the simulation output. The design space for fitting the metamodel is well defined by considering where the values computed under step (i) fall, and so these bootstrap resample moments themselves could be used for the experiment design. Unfortunately, this approach has two drawbacks. First, if the number of bootstrap resamples ($B$) is not very large, then the randomness of the selection process will not likely result in uniform coverage of the design space. On the other hand, if $B$ is very large, then running the simulation an extremely large number of times will be computationally intractable. One might imagine selecting a small subset of a large number of candidate bootstrap resamples to maximize some uniformity property, such as a maxi–min design, but this strategy is also computationally intractable when $B$ is large. This has been a motivating factor for computationally efficient design construction strategies such as Latin hypercubes and other orthogonal arrays, and uniform designs. Our strategy is to modify one of these efficient design generation schemes to generate points falling in some regular region where bootstrap resample moments can be expected to fall.

The most common regular region used to define a design space is a hyperbox. For ease of exposition, we first suppose that there is a single input from which we have a sample of real-world data $z_1^{(0)}, z_2^{(0)}, \ldots, z_m^{(0)}$ and a two-parameter input distribution (such as the gamma, lognormal, or Weibull). A bootstrap resample is a sample of size $m$ with replacement from $z_1^{(0)}, z_2^{(0)}, \ldots, z_m^{(0)}$. Let $Z_1^{(b)}, Z_2^{(b)}, \ldots, Z_m^{(b)}$ denote the $b$th bootstrap resample, with corresponding standardized sample moments (mean and standard deviation) of

$$\bar{Z}_m^{(b)} = \frac{1}{m}\sum_{j=1}^{m} Z_j^{(b)}$$

$$S_m^{(b)} = \sqrt{\frac{1}{m-1}\sum_{j=1}^{m}(Z_j^{(b)} - \bar{Z}_m^{(b)})^2}.$$

The metamodel will be used to predict the mean simulation response given any bootstrap resample moments $(\bar{Z}_m^{(b)}, S_m^{(b)})$. Thus, we need the metamodel to be accurate over the space of possible $(\bar{Z}_m^{(b)}, S_m^{(b)})$ pairs that we could obtain from bootstrap resampling the real-world data.

There is no way to specify a tight experiment design—one that covers only the relevant moment space—prior to obtaining the real-world data. Therefore, we need to use the real-world data to guide the experiment design. Given $z_1^{(0)}, z_2^{(0)}, \ldots, z_m^{(0)}$, the extreme moments can be computed: the minimum and maximum sample means are $\min z_j^{(0)}$ and $\max z_j^{(0)}$, respectively, and the minimum standard deviation is 0 and the maximum (if $m$ is even) is

$$\sqrt{\frac{m}{m-1}\left(\frac{\max z_j^{(0)} - \min z_j^{(0)}}{2}\right)^2}.$$

However, the rectangle defined by these extremes is a poor choice for the design space because the extremes are very unlikely, or impossible in the case of the maximum standard deviation paired with either the minimum or maximum mean. Further, $(\bar{Z}_m^{(b)}, S_m^{(b)})$ will tend to be correlated so that the entire rectangle is not equally relevant. These deficiencies become even more severe when there are $L > 1$ input processes if we let the experimental region be the hyperrectangle defined by the extreme moments.

The tilted cloud shape common in scatter plots suggests that an ellipse (or ellipsoid when $d > 2$) will tend to provide a more suitable regular region than a hyperbox. When $B$ is not very large, the enclosing ellipsoid combined with a transformed Latin hypercube design that falls within the ellipsoid is easy to compute and helps ensure uniform coverage. This leads to the following algorithm.

**Fit Ellipse.**

1. Generate $B_0$ bootstrap resamples from $\mathbf{z}_\mathbf{m}^{(0)}$ and compute the corresponding sample moments $\hat{\mathbf{X}}_\mathbf{m}^{(b)}$, $b = 1, 2, \ldots, B_0$.

2. Fit an ellipsoid to the sample moment data in such a way that it contains a large fraction of the pairs, say $q$. Specifically, find the smallest $a^2$ such that

$$\#\{b\colon (\hat{\mathbf{X}}_\mathbf{m}^{(b)} - \mathbf{M})^\top \mathbf{V}^{-1}(\hat{\mathbf{X}}_\mathbf{m}^{(b)} - \mathbf{M}) \le a^2\}/B_0 \ge q \qquad (10)$$

where

$$\mathbf{M} = \sum_{b=1}^{B_0} \hat{\mathbf{X}}_\mathbf{m}^{(b)}/B_0 \quad \text{and}$$

$$\mathbf{V} = \sum_{b=1}^{B_0} (\hat{\mathbf{X}}_\mathbf{m}^{(b)} - \mathbf{M})^\top (\hat{\mathbf{X}}_\mathbf{m}^{(b)} - \mathbf{M})/(B_0 - 1).$$

3. Generate $B_1$ independent bootstrap resamples from $\mathbf{z}_{\mathbf{m}}^{(0)}$ and compute $\hat{\mathbf{X}}_{\mathbf{m}}^{(b)}$, $b = B_0 + 1, B_0 + 2, \ldots, B_0 + B_1$. If more than $c$ of these $B_1$ additional pairs are contained in the ellipsoid (10), then accept this ellipsoid as the design space. Otherwise, add these bootstrap resamples to the data set, let $B_0 \leftarrow B_0 + B_1$, and go to step 2.

Here, $B_1$ and $c$ are chosen to obtain a desired type I error and power for a hypothesis test based on the binomial distribution that the probability a bootstrap resample moment is contained in the ellipsoid is $\geq q$. In our experiments we set $q = 0.99$, type I error to 0.005, and power to 0.95 when the true probability is $q = 0.97$ or less.

The output of this algorithm is an ellipsoid defined by $\mathbf{M}$, $\mathbf{V}$, and $a$

$$\{\mathbf{x} \in \Re^d\colon (\mathbf{x} - \mathbf{M})^\top \mathbf{V}^{-1}(\mathbf{x} - \mathbf{M}) \leq a^2\}. \tag{11}$$

To place $k$ design points into this space, we employ an algorithm due to Sun and Farooq (2002), §3.2.1, for generating points uniformly distributed in the desired ellipsoid. The algorithm first generates the polar coordinates of a point uniformly distributed in a hypersphere, then transforms it to Cartesian coordinates, and finally transforms it again to a point uniformly distributed in an ellipsoid. The advantage of this approach is that each element of the initial polar coordinates are independently distributed, allowing them to be generated coordinate by coordinate via their inverse cumulative distribution function. Therefore, if we start with points $\{\mathbf{u}_j, j = 1, 2, \ldots, k\} \in [0, 1]^d$ that are *space filling instead of random*, then their space filling property will tend to be preserved in the ellipsoid. We obtain $\{\mathbf{u}_j, j = 1, 2, \ldots, k\}$ by generating many Latin hypercube samples and picking the one that maximizes the minimum distance between points. The output is then an experiment design $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k$ with all the $\mathbf{x}_i$ falling in the ellipsoid (11). Figure 1 illustrates how this works when there is $L = 1$ input distribution with two parameters.

Note that no simulation experiments are required to generate the experiment design. Of course the bootstrap resamples that are generated to create the ellipsoid could be evaluated directly via simulation runs rather than fitting a metamodel at all. This strategy is inferior to what we propose for three reasons. First, the simulation response is stochastic: two replications with the same input parameters produce different results, so the bootstrap statistic is not a smooth function of the data. Second, when the number of input parameters is not too large, the experiment design that we generate to fit a metamodel will require significantly fewer simulation runs (e.g., tens) than would be needed for direct bootstrapping (e.g., thousands). Third, the intrinsic error effect on the correctness of bootstrap intervals as discussed in Barton (2007) is reduced significantly when it is propagated through the fitted metamodel.

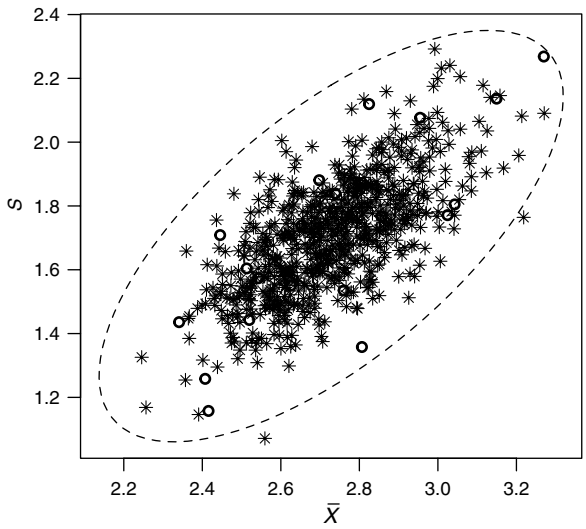


**Figure 1 Experiment Design in Moment Space, Where the Horizontal Axis Is the Mean and the Vertical Axis Is the Standard Deviation**

*Note.* A $*$ indicates a bootstrap resample, the dashed line is an ellipse that covers 99% of the data, and a $\circ$ indicates a design point.

### 4.3. Metamodel-Assisted Bootstrap CI

In this section we present the metamodel-assisted bootstrapping algorithm and provide some theoretical support for its use.

**Metamodel-Assisted Bootstrap**

1. Given real-world data $\mathbf{z}_{\mathbf{m}}^{(0)}$, obtain experiment design $\mathbf{x}_i$, $i = 1, 2, \ldots, k$ as described in §4.2.

2. For design points $i = 1, 2, \ldots, k$, simulate i.i.d. replications $Y_j(\mathbf{x}_i)$ for $j = 1, 2 \ldots, n_i$, and compute the sample average $\bar{Y}(\mathbf{x}_i)$ and sample variance $S^2(\mathbf{x}_i)$ of the simulation outputs.

3. Fit a stochastic kriging metamodel $\hat{\mu}(\mathbf{x})$ using $(\bar{Y}(\mathbf{x}_i), S^2(\mathbf{x}_i), \mathbf{x}_i)$, $i = 1, 2, \ldots, k$ as described in §4.1 (see Equation (9)).

4. For $b = 1$ to $B$,

(a) generate bootstrap resample $\mathcal{Z}_{\mathbf{m}}^{(b)} \sim \mathbf{z}_{\mathbf{m}}^{(0)}$ and compute sample moments $\hat{\mathbf{X}}_{\mathbf{m}}^{(b)}$;

(b) set $\hat{\mu}_b = \hat{\mu}(\hat{\mathbf{X}}_{\mathbf{m}}^{(b)})$.

Next $b$

5. Report bootstrap percentile confidence interval $[\hat{\mu}_{(\lceil B\alpha/2 \rceil)}, \hat{\mu}_{(\lceil B(1-\alpha/2) \rceil)}]$, where $\hat{\mu}_{(1)} \leq \hat{\mu}_{(2)} \leq \cdots \leq \hat{\mu}_{(B)}$ are the sorted values.

Consider the algorithm above with $\mu(\cdot)$, the true response function, replacing the simulation-based estimator $\hat{\mu}(\mathbf{x})$ everywhere in the algorithm. We establish the asymptotic coverage of the bootstrap percentile CI in this case, then discuss the necessity for estimating $\mu(\cdot)$ following the proof.

THEOREM 1. *Suppose that the following conditions hold*:

1. *The input distribution $F_l$ is uniquely determined by its first $p_l$ moments, and it has finite first $2p_l$ moments, for $l = 1, 2, \ldots, L$.*

2. *We have i.i.d. observations $Z_{l,1}^{(0)}, Z_{l,2}^{(0)}, \ldots, Z_{l,m_l}^{(0)}$ from $F_l$, for $l = 1, 2, \ldots, L$. As $m \to \infty$ we have $m/m_l \to 1$, $l = 1, 2, \ldots, L$, where $m$ drives the number of observations to $\infty$ from all input sources.*

3. *The response surface $\mu(\mathbf{x})$ is finite and continuously differentiable with nonzero gradient in a neighborhood of $\mathbf{x}^c$.*

4. *If $\mathbf{x}'$ is a moment vector such that $\mu(\mathbf{x}')$ does not exist, then the definition of $\mu(\cdot)$ is extended so that $\mu(\mathbf{x}') = \infty$ or $-\infty$.*[2]

*Then the bootstrap CI is consistent, meaning*

$$\lim_{m\to\infty}\lim_{B\to\infty}\Pr\{\mu(\mathbf{x}^c)\in[\mu_{(\lceil B\alpha/2\rceil)},\mu_{(\lceil B(1-\alpha/2)\rceil)}]\}=1-\alpha. \quad (12)$$

PROOF. We show that the CI satisfies the conditions of Theorem 4.1 of Shao and Tu (1995) for consistency of a bootstrap CI.

The distribution of $\sqrt{m}[\mu(\hat{\mathbf{X}}_{\mathbf{m}}^{(b)}) - \mu(\hat{\mathbf{X}}_{\mathbf{m}}^{(0)})]$ is strongly consistent for the distribution of $\sqrt{m}[\mu(\hat{\mathbf{X}}_{\mathbf{m}}^{(0)}) - \mu(\mathbf{x}^c)]$ as $m \to \infty$ by Theorem 3.1 of Shao and Tu (1995). The distribution of $\sqrt{m}[\mu(\hat{\mathbf{X}}_{\mathbf{m}}^{(0)}) - \mu(\mathbf{x}^c)]$ is asymptotically normal with mean 0 as $m \to \infty$ using standard delta-method arguments. Together, these results establish that the percentile interval $[\mu_{(\lceil B\alpha/2\rceil)}, \mu_{(\lceil B(1-\alpha/2)\rceil)}]$ satisfies the conditions of Theorem 4.1 of Shao and Tu (1995) and is therefore asymptotically consistent as $m, B \to \infty$; that is, (12) holds. □

The CI in Theorem 1 is $[\mu_{(\lceil B\alpha/2\rceil)}, \mu_{(\lceil B(1-\alpha/2)\rceil)}]$, and the CI from metamodel-assisted bootstrapping is $[\hat{\mu}_{(\lceil B\alpha/2\rceil)}, \hat{\mu}_{(\lceil B(1-\alpha/2)\rceil)}]$. Stated differently, the theorem assumes that $\hat{\mu}(\mathbf{x}) = \mu(\mathbf{x})$ and there is no metamodeling error. As a practical matter, what our proof establishes is that metamodel-assisted bootstrapping will be effective when we have a (nearly) global metamodel with a good fit. We chose stochastic kriging because it explicitly accounts for simulation variability through the intrinsic variance matrix $\mathbf{C}$ and can provide a good fit without making strong assumptions about the form of the response surface, such as being linear as assumed in a first-order Taylor series approximation. Notice, also, that stochastic kriging with the Gaussian correlation function (8) always satisfies the smoothness property of condition 3. What we have found to date (see the results in §5 and in Barton et al. 2010) is that when the input uncertainty is substantially more significant than the output variability, then the metamodel-assisted bootstrap CI attains the correct coverage. This shows that in practice the metamodel need not be exact, but only sufficiently good so that it can be treated as exact. Unfortunately, the interaction between the amount of real-world data available and the simulation effort required to obtain a metamodel that is "sufficiently good" is not simple: When $\mathbf{m}$ is very large, the metamodel may only need to be accurate over a very small region near the true values of the input-model parameters; thus, the entire experiment design is concentrated within a small ellipsoid. When $\mathbf{m}$ is small, the variation in the likely parameter values could be substantial, leading to an ellipsoid that covers a large space of possible parameter values and a complex response surface. For these reasons we introduce an empirical test for sufficiently good next.

For stochastic kriging, having an adequate number of design points $k$ seems to be more important than having a substantial number of replications per design point $n_i$. The standard recommendation from the kriging literature of $k \geq 10d$, where $d$ is the dimension of the independent variable $\mathbf{x}$, works well and is what we recommend. In the absence of any prior information about the variances of the simulation outputs at the design points $(\sigma^2(\mathbf{x}_i), i = 1, 2, \ldots, k)$, an equal allocation of $n_i = n_0$ replications per design point is sensible; we recommend $n_0 \geq 10$ simply to insure a stable estimate of the intrinsic variance matrix $\mathbf{C} = \text{diag}[\sigma^2(\mathbf{x}_1), \sigma^2(\mathbf{x}_2), \ldots, \sigma^2(\mathbf{x}_k)]/n_0$.

Following these simple guidelines will frequently be sufficient to achieve the desired CI coverage. However, it is possible that if the intrinsic variance is very large then there might be undercoverage; therefore, we provide the following easy-to-apply empirical test and remedial action if the test fails.

Our CI is based on the metamodel $\hat{\mu}(\cdot)$ formed from $k$ design points with (say) $n_0$ replications at each. Let $\hat{\mathbf{C}}$ be the estimate of $\mathbf{C}$. Notice that, given a set of parameters, $\hat{\mathbf{C}}$ is the only term in the stochastic kriging metamodel (9) that is affected by the number of replications. If we increased the number of replications per design point to $n_0 + \Delta n$, then we would expect the new estimate of $\mathbf{C}$ to be approximately $\hat{\mathbf{C}}^+ = (n_0/(n_0 + \Delta n))\hat{\mathbf{C}}$. As a test for whether the CI in step 5 is sensitive to additional replications, we add the following steps to the metamodel-assisted bootstrap algorithm:

6. Form an adjusted metamodel $\hat{\mu}^+(\cdot)$ by only replacing $\hat{\mathbf{C}}$ with $\hat{\mathbf{C}}^+$ and leaving everything else the same (no new simulation-output data are generated).

7. If

$$\frac{\#\{\hat{\mu}^+(\mathbf{X}_{\mathbf{m}}^{(b)})\in[\hat{\mu}_{(\lceil B\alpha/2\rceil)},\hat{\mu}_{(\lceil B(1-\alpha/2)\rceil)}]\}}{B}\approx 1-\alpha, \quad (13)$$

then accept the CI. Otherwise, actually increase the number of replications per design point to $n_0 \leftarrow n_0 + \Delta n$ and go to step 3.

[2] The canonical example of this is an open queueing system where the arrival rate is greater than the service rate. This extension insures that bootstrap samples leading to infeasible moments imply responses in an extreme tail.

In words, (13) tests whether the empirical coverage of the bootstrap CI would change if we increased the number of replications per design point. If not, then the current CI is stable with respect to the simulation-output variance. The algorithm is guaranteed to terminate because $\mathbf{C} \to \mathbf{0}$ with probability 1 as $n \to \infty$. We illustrate the contribution of these additional steps in the next section.

## 5. Empirical Evaluation

To evaluate metamodel-assisted bootstrapping, we consider estimating the mean number of customers in a $GI/G/1/50$ queue where the interarrival times are gamma$(\alpha_A, \beta_A)$ and the service times are gamma$(\alpha_S, \beta_S)$. The real-world parameters are $\alpha_A = 2$, $\beta_A = 5/3$, $\alpha_S = 3$, $\beta_S = 1$, implying a traffic intensity of 0.9. Based on a very long simulation, the true mean number of customers in the system in steady state is 4.147. In the experiments we assume that the input distribution families are known to be gamma, but the parameters $\alpha_A$, $\beta_A$, $\alpha_S$, $\beta_S$ are unknown. To imitate obtaining real-world data, we generate samples from the correct gamma distributions.

When we do metamodel-assisted bootstrapping, the metamodel is a function of the means and standard deviations of both interarrival and service times; that is, $p_A = p_S = 2$ and the metamodel is four-dimensional. We systematically examined the impact of the quantity of real-world data, the total simulation budget, and allocation of the budget between design points and replications by varying them as follows: we chose real-world sample sizes $m = 10, 50, 100, 500$ (both interarrival and service times have a common sample size); two levels of total computational budget $n_c = 600, 1{,}200$ replications; and number of design points $k = 20, 40, 60$ (see Figure 2). At each design point, the run length was 1,000 finished customers with warm-up period of 300 customers. The number of replications at all design points is $n_c/k$, so we did not take advantage of the ability of SK to differentially allocate replications based on differences in output variance.

We also report results for the conditional CI, which means fitting the gamma distributions to the real-world data and allocating all $n_c$ replications to simulating the resulting system. The competitors to metamodel-assisted bootstrapping were evaluated in Barton et al. (2010).

Table 1 displays the coverage, width, and standard deviation of the width of metamodel-assisted bootstrapping CIs and conditional CIs based on 1,000 macroreplications of the entire experiment; each macroreplication obtains an independent sample of real-world data, and the nominal confidence level is 95%. The small width of the conditional CIs suggests that we have estimated the true mean very precisely. However, the coverage of the conditional CI is far from the nominal 95% level, even when the quantity of real-world data is a relatively large $m = 500$ observations. The much wider CI obtained with metamodel-assisted bootstrapping shows that, even with 500 real-world samples, input uncertainty dominates the simulation uncertainty. In other words, the simulation has no practical value if we really want to know what the long-run average number of customers would be in the real world: the CI width is three to 11 times as big as the mean value. Without some evaluation of input uncertainty the analyst would have no clue how little they actually knew.

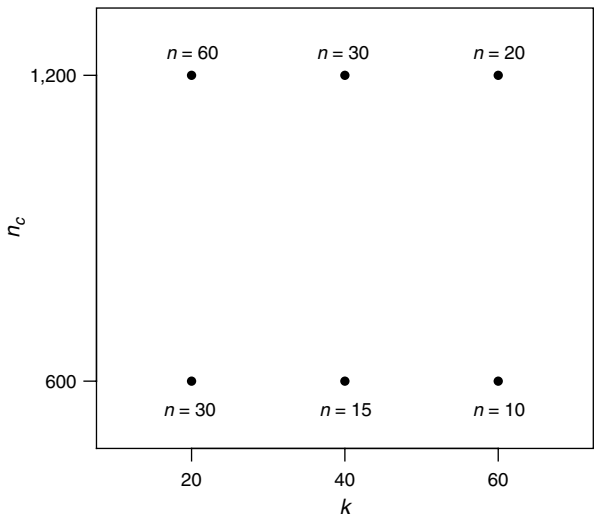


**Figure 2** **Experiments for $GI/G/1/50$ Queue**

Notice that the coverage of the metamodel-assisted bootstrapping CI is very close to the nominal value when $m$ is 50 or greater, and the width of the CI decreases as roughly $\sqrt{m}$. The coverage is closer to 90% when we have only 10 real-world observations, but this is a very small sample of data to estimate the parameters of any distribution. In these experiments the coverage of the metamodel-assisted CI is robust to the experiment design; that is, it works well with a smaller design each receiving many replications, or a larger design with fewer replications per design point.

These results, as well as those in Barton et al. (2010), demonstrated robust coverage provided that there is at least a modest amount of real-world data; and even when there is little real-world data ($m = 10$ in our experiments) the coverage is much closer to the nominal level than the conditional CI.

The amount of real-world data is often not under our control, but the amount of simulation effort is. Therefore, we created an example in which $n_c$ is initially too small to obtain the desired coverage to show that our test (steps 6–7) can both detect the lack of coverage and compensate for it. To do this, we cut

**Table 1 Coverage, Width, and Standard Deviation (Std. Dev.) of Width of Metamodel-Assisted Bootstrapping and Conditional CIs**

| | $n_c = 600$ | | | $n_c = 1{,}200$ | | |
|---|---|---|---|---|---|---|
| | $k = 20$ | $k = 40$ | $k = 60$ | $k = 20$ | $k = 40$ | $k = 60$ |
| $m = 10$ | | | | | | |
| M-A bootstrap coverage (%) | 88.8 | 90.9 | 86.2 | 89.0 | 90.2 | 83.3 |
| M-A bootstrap width | 48.97 | 47.90 | 44.09 | 49.39 | 48.45 | 43.64 |
| Std. dev. bootstrap width | 16.39 | 15.43 | 18.20 | 15.82 | 15.86 | 18.38 |
| Conditional CI coverage (%) | | 1 | | 0.3 | | |
| Conditional CI width | | 0.27 | | 0.19 | | |
| Std. dev. conditional width | | 0.20 | | 0.10 | | |
| $m = 50$ | | | | | | |
| M-A bootstrap coverage (%) | 94.6 | 95.3 | 95.0 | 94.9 | 96.0 | 95.4 |
| M-A bootstrap width | 40.40 | 39.02 | 37.85 | 40.36 | 39.21 | 38.00 |
| Std. dev. bootstrap width | 15.00 | 13.72 | 13.67 | 14.91 | 13.87 | 13.64 |
| Conditional CI coverage (%) | | 2 | | 2 | | |
| Conditional CI width | | 0.49 | | 0.32 | | |
| Std. dev. conditional width | | 0.28 | | 0.13 | | |
| $m = 100$ | | | | | | |
| M-A bootstrap coverage (%) | 95.4 | 94.7 | 94.8 | 96.5 | 95.6 | 95.5 |
| M-A bootstrap width | 33.14 | 32.12 | 31.30 | 33.22 | 32.04 | 32.09 |
| Std. dev. bootstrap width | 14.76 | 14.54 | 14.54 | 14.92 | 14.28 | 14.20 |
| Conditional CI coverage (%) | | 3 | | 1 | | |
| Conditional CI width | | 0.45 | | 0.34 | | |
| Std. dev. conditional width | | 0.24 | | 0.13 | | |
| $m = 500$ | | | | | | |
| M-A bootstrap coverage (%) | 96.1 | 96.1 | 93.8 | 94.8 | 95.6 | 96.2 |
| M-A bootstrap width | 12.27 | 11.71 | 11.46 | 12.62 | 12.29 | 11.64 |
| Std. dev. bootstrap width | 8.49 | 8.00 | 8.39 | 8.43 | 8.32 | 8.04 |
| Conditional CI coverage (%) | | 6 | | 4 | | |
| Conditional CI width | | 0.32 | | 0.23 | | |
| Std. dev. conditional width | | 0.08 | | 0.03 | | |

the run length of each replication from 1,000 completed customers to 50 completed customers, applied the rule of thumb of $k = 10d = 40$ design points, initially made $n_0 = 10$ replications at each design point, and used the increment $\Delta n = 10$ if the test in step 6 indicated that more replications were needed (estimated coverage that deviated by no more than 0.005 from 0.95 was required to pass the test). We did this for two levels of real-world data, $m = 100, 500$. This gave coverages after the initial $n_0$ replications of only 0.921 and 0.924, respectively. When $m = 100$, implementation of steps 6 and 7 increased the coverage to 0.953 with an average of 14.9 replications per design point at termination. When $m = 500$ the coverage was 0.955 with an average of 18.4 replications per design point at termination. As before, coverage was estimated using 1,000 macroreplications.

## 6. Conclusions

The empirical results in this paper, and those in Barton et al. (2010), illustrate good performance for metamodel-assisted bootstrapping when the simulation budget is sufficient to give a short conditional CI; in other words, when input uncertainty dominates point-estimator variability. We also provided an empirical test to verify that this is the case. Because simulation replications are typically cheap relative to real-world data, metamodel-assisted bootstrapping addresses the situation most frequently encountered in practice. We do not recommend this method when the simulation budget is so tight that even the conditional CI is relatively wide.

Although our one-stage experiment design worked well in the examples, there will be simulations with more complex response surfaces for which a sequential design will be needed to obtain an adequate metamodel. Rather than evenly distributing the simulation effort throughout the design space, we should sequentially and adaptively add design points and replications to balance the global and local fitting uncertainties. Specifically, we need an accurate fit in subareas of the moment space with responses close to the quantiles that form the bootstrap CI. Sequential design is a strength of stochastic kriging.

An advantage of metamodel-assisted bootstrapping relative to direct bootstrapping is effective use of the simulation budget: Metamodel-assisted bootstrapping spends the simulation effort on a designed experiment that covers the input space and leverages information from all design points; this makes bootstrapping not only fast but also robust to simulation-estimation error. Direct bootstrapping spreads the simulation budget across 1,000 or more bootstrap resamples, giving noisy, isolated estimates of the simulation response at each bootstrapped input setting.

However, there will be limits to the number of input models we can handle via a metamodel: 50 input models averaging two parameters each implies a 100-dimensional input space. Adding a screening

step that first determines the most influential input models, and then accounting for only their input uncertainty via the metamodel, is one way to extend the methodology to such situations.

When will our method fail or not be appropriate? Clearly, input distributions without enough finite moments (e.g., Cauchy) are outside of the scope of this approach. Also, continuity and differentiability of $\mu(\cdot)$ in a neighborhood of the true parameters $\mathbf{x}^c$ matter. Consider, for instance, a multiclass queueing network where one input model is the distribution of customer type. A response surface, such as the mean time in the system, can have nondifferentiable points when a small change in customer mix distribution causes the bottleneck queue to shift. Of course, having the true input parameters correspond exactly to such a point is unlikely.

A further limitation of our approach is the use of parametric input models. Of course, there is no true, correct input model for any real-world process, so even techniques that average over a number of possible model families suffer this problem (although perhaps less so). Our proposal is to extend the metamodel-assisted bootstrapping approach to simulations driven by empirical distributions (that is, the real-world data itself).

## Acknowledgments

Portions of this paper were previously published in Barton et al. (2010). The authors acknowledge the helpful comments of the associate editor and referees in refining the presentation. This paper is based upon work supported by the National Science Foundation [Grants CMMI-0900354 and CMMI-1068473].

## References

Ankenman B, Nelson BL, Staum J (2010) Stochastic kriging for simulation metamodeling. *Oper. Res.* 58:371–382.

Barton RR (2007) Presenting a more complete characterization of uncertainty: Can it be done? *Proc. 2007 INFORMS Simulation Soc. Res. Workshop* (INFORMS Simulation Society, Fontainebleau), http://www.informs-sim.org/2007informs-csworkshop/13.pdf.

Barton RR, Schruben LW (1993) Uniform and bootstrap resampling of input distributions. Evans GW, Mollaghasemi M, Biles WE, Russell EC, eds. *Proc. 1993 Winter Simulation Conf.* (Institute of Electrical and Electronics Engineers, Piscataway, NJ), 503–508.

Barton RR, Schruben LW (2001) Resampling methods for input modeling. Peters BA, Smith JS, Medeiros DJ, Rohrer MW, eds. *Proc. 2001 Winter Simulation Conf.* (Institute of Electrical and Electronics Engineers, Piscataway, NJ), 372–378.

Barton RR, Nelson BL, Xie W (2010) A framework for input uncertainty analysis. Johansson B, Jain S, Montoya-Torres J, Hugan J, Yücesan E, eds. *Proc. 2010 Winter Simulation Conf.* (Institute of Electrical and Engineers, Inc., Piscataway, NJ), 1189–1198.

Barton RR, Cheng RCH, Chick SE, Henderson SG, Law AM, Leemis LM, Schmeiser BW, Schruben LW, Wilson JR (2002) Panel on current issues in simulation input modeling. Yücesan E, Chen C-H, Snowdon JL, Charnes JM, eds. *Proc. 2002 Winter Simulation Conf.* (Institute of Electrical and Electronics Engineers, Piscataway, NJ), 353–369.

Biller B, Corlu CG (2011) Accounting for parameter uncertainty in large-scale stochastic simulations with correlated inputs. *Oper. Res.* 59:661–673.

Chen X, Ankenman BE, Nelson BL (2012) The effects of common random numbers on stochastic kriging metamodels. *ACM Trans. Modeling Comput. Simulation* 22(2):Article 7.

Cheng RCH (1994) Selecting input models. Tew JD, Manivannan S, Sadowski DA, Seila AF, eds. *Proc. 1994 Winter Simulation Conf.* (Institute of Electrical and Electronics Engineers, Piscataway, NJ), 184–191.

Cheng RCH, Holland W (1997) Sensitivity of computer simulation experiments to errors in input data. *J. Statist. Comput. Simulation* 57:219–241.

Cheng RCH, Holland W (1998) Two-point methods for assessing variability in simulation output. *J. Statist. Comput. Simulation* 60:183–205.

Cheng RCH, Holland W (2004) Calculation of confidence intervals for simulation output. *ACM Trans. Modeling Comput. Simulation* 14:344–362.

Chick SE (1997) Bayesian analysis for simulation input and output. Andradottir S, Healy KJ, Withers DH, Nelson BL, eds. *Proc. 1997 Winter Simulation Conf.* (Institute of Electrical and Electronics Engineers, Piscataway, NJ), 253–260.

Chick SE (1999) Steps to implement Bayesian input distribution selection. Farrington PA, Nembhard HB, Sturrock DT, Evans GW, eds. *Proc. 1999 Winter Simulation Conf.* (Institute of Electrical and Electronics Engineers, Piscataway, NJ), 317–324.

Chick SE (2000) Bayesian methods for simulation. Joines JA, Barton RR, Kang K, Fishwick PA, eds. *Proc. 2000 Winter Simulation Conf.* (Institute of Electrical and Electronics Engineers, Piscataway, NJ), 109–118.

Chick SE (2001) Input distribution selection for simulation experiments: Accounting for input uncertainty. *Oper. Res.* 49:744–758.

Chick SE, Ng SH (2002) Joint criterion for factor identification and parameter estimation. Yücesan E, Chen C-H, Snowdon JL, Charnes JM, eds. *Proc. 2002 Winter Simulation Conf.* (Institute of Electrical and Electronics Engineers, Piscataway, NJ), 400–406.

Freimer M, Schruben LW (2002) Collecting data and estimating parameters for input distributions. Yücesan E, Chen C-H, Snowdon JL, Charnes JM, eds. *Proc. 2002 Winter Simulation Conf.* (Institute of Electrical and Electronics Engineers, Piscataway, NJ), 392–399.

Henderson SG (2003) Input model uncertainty: Why do we care and what should we do about it? Chick SE, Sanchez PJ, Ferrin D, Morrice DJ, eds. *Proc. 2003 Winter Simulation Conf.* (Institute of Electrical and Electronics Engineers, Piscataway, NJ), 90–100.

Kleijnen JPC (1987) *Statistical Tools for Simulation Practitioners* (Marcel Dekker Inc., New York).

Kleijnen JPC (1994) Sensitivity analysis versus uncertainty analysis: When to use what? Grasman J, van Straten G, eds. *Predictability and Nonlinear Modelling in Natural Sciences and Economics* (Kluwer, Dordrecht, the Netherlands), 322–333.

Law AM, Kelton WD (2000) *Simulation Modelling and Analysis*, 3rd ed. (McGraw Hill, New York).

Lee S-H, Glynn PW (2003) Computing the distribution function of a conditional expectation via Monte Carlo: Discrete conditioning spaces. *ACM Trans. Modeling Comput. Simulation* 13:238–258.

Ng SH, Chick SE (2001) Reducing input parameter uncertainty for simulations. Peters BA, Smith JS, Medeiros DJ, Rohrer MW, eds. *Proc. 2001 Winter Simulation Conf.* (Institute of Electrical and Electronics Engineers, Piscataway, NJ), 364–371.

Santner TJ, Williams BJ, Notz WI (2003) *The Design and Analysis of Computer Experiments* (Springer, New York).

Shao J, Tu D (1995) *The Jackknife and Bootstrap* (Springer, New York).

Steckley SG, Henderson SG (2003) A kernel approach to estimating the density of a conditional expectation. Chick SE, Sanchez PJ, Ferrin D, Morrice DJ, eds. *Proc. 2003 Winter Simulation Conf.* (Institute of Electrical and Electronics Engineers, Piscataway, NJ), 383–391.

Sun H, Farooq M (2002) Note on the generation of random points uniformly distributed in hyper-ellipsoids. *Proc. Fifth Internat. Conf. Inform. Fusion*, 489–496.

Zouaoui F, Wilson JR (2001a) Accounting for input model and parameter uncertainty in simulation. Peters BA, Smith JS, Medeiros DJ, Rohrer MW, eds. *Proc. 2001 Winter Simulation Conf.* (Institute of Electrical and Electronics Engineers, Piscataway, NJ), 290–299.

Zouaoui F, Wilson JR (2001b) Accounting for parameter uncertainty in simulation input modeling. Peters BA, Smith JS, Medeiros DJ, Rohrer MW, eds. *Proc. 2001 Winter Simulation Conf.* (Institute of Electrical and Electronics Engineers, Piscataway, NJ), 354–363.

Zouaoui F, Wilson JR (2003) Accounting for parameter uncertainty in simulation input modeling. *IIE Trans.* 35:781–792.

Zouaoui F, Wilson JR (2004) Accounting for input-model and input-parameter uncertainties in simulation. *IIE Trans.* 36:1135–1151.