Simulation-optimization via Kriging and bootstrapping: a survey

Jack P.C. Kleijnen
Department of Information Management / CentER,
Tilburg University, Postbox 90153, 5000 LE Tilburg,
Netherlands,phone +31-13-466-2029; fax +31-13-466-3069;
kleijnen@tilburguniversity.edu

Abstract

This survey considers the optimization of simulated systems. The simulation may be either deterministic or random. The survey reflects the author's extensive experience with simulation-optimization through Kriging (or Gaussian process) metamodels using a frequentist (non-Bayesian) approach. The analysis of Kriging metamodels may use bootstrapping. The survey discusses both parametric bootstrapping for deterministic simulation and distribution-free bootstrapping for random simulation. The survey uses only basic mathematics and statistics; its 51 references enable further study. More specifically, this article reviews the following recent topics: (1) A popular simulation-optimization heuristic is Efficient Global Optimization (EGO) using Expected Improvement (EI); parametric bootstrapping can estimate the variance of the Kriging predictor, accounting for the randomness resulting from estimating the Kriging parameters. (2) Optimization with constraints for random simulation outputs and deterministic inputs may use mathematical programming applied to Kriging metamodels; validation of these metamodels may use distribution-free bootstrapping. (3) Taguchian robust optimization accounts for an uncertain environment; this optimization may use mathematical programming applied to Kriging metamodels, while distribution-free bootstrapping may estimate the variability of the Kriging metamodels and the resulting robust solution. (4) The bootstrap may improve the convexity or monotonicity of the Kriging metamodel, if the input/output function of the underlying simulation model is assumed to have such a characteristic.

Keywords: Simulation; Global optimization; Gaussian Process; Resampling; Sensitivity analysis; Robustness

Version: November 27, 2012

1 Introduction

In this paper we consider the problem of optimizing complex real-life systems that are represented through simulation models. These simulation models may be either deterministic or random. *Deterministic* models often represent real-life systems that are governed by laws of physics; many examples are found in computer aided engineering; see [15] and the references in [24], p. 3. *Random* or stochastic simulation models—including discrete-event simulation—often represent social systems in which humans create noise; examples are queueing systems in telecommunications and logistics with random customer arrival and service times; see [32]. Finding the optimal input combinations for these simulation models may use a large variety of methods, as is illustrated by the articles in this Special Issue; also see [18] and [24]. Notice that "input combinations" are also called "points" (in the search space) or "scenarios".

In this survey we limit our review to our own research performed together with various co-authors, on simulation-optimization through Kriging metamodels. In the simulation literature, a *metamodel* is an explicit model of an underlying simulation model; metamodels are also called response surfaces, surrogates, emulators, etc. The most popular type of metamodel is a first-order or second-order polynomial in the simulation inputs, but we focus on *Kriging* metamodels. The latter models are also called Gaussian process (GP) models (the term GP will become clear in our technical discussion in Section 2.1). For the analysis of Kriging metamodels we use *bootstrapping*; in general, bootstrapping is a versatile method for analyzing nonlinear statistics; e.g., a nonlinear statistic is the ratio of two random variables (say) x/y, for which it is well-known that $E(x/y) \neq E(x)/E(y)$. A more interesting example of a nonlinear statistic is the variance of the predictor given by a Kriging metamodel with estimated parameters; see Section 3. We shall discuss both the parametric bootstrap for deterministic simulation and the nonparametric or distribution-free bootstrap for random simulation. The bootstrap method avoids complicated asymptotic methods; i.e., bootstrapping is a simple small-sample method—and small samples are common in so-called expensive simulation, which requires much computer time.

We shall also mention software for simulation-optimization, Kriging, and bootstrapping. Obviously, such software stimulates the application of methods in practice. We give a survey of these various methods, using only basic mathematics and statistics; our 51 references enable readers to study the technical details of these methods (we avoid giving a long list of references on simulation-optimization in general). Notice that we

use a frequentist approach, not a Bayesian approach; the latter is also popular in Kriging and simulation-optimization, but we have no personal experience with Bayesian methods.

After a summary of the basics of Kriging and bootstrapping in Section 2, we present a survey of the following related topics in simulation-optimization through Kriging and bootstrapping.

1. Estimating the variance of the nonlinear Kriging predictor that uses estimated Kriging parameters, applying parametric bootstrapping: see Section 3.

2. Using this bootstrapped variance estimator in the popular simulation-optimization heuristic called Efficient Global Optimization (EGO) with its Expected Improvement (EI) criterion: Section 4.

3. Constrained optimization via integer nonlinear programming (INLP) and Kriging metamodels validated through distribution-free bootstrapping: Section 5.

4. "Robust" simulation-optimization—in the sense of Taguchi (see [48]), accounting for an uncertain environment—applying distribution-free bootstrapping to quantify the variability of Kriging metamodels: Section 6.

5. Convexity-improving and monotonicity-preserving Kriging through distribution-free bootstrapping: Section 7.

Note: So-called "random" simulation may be interpreted in three different ways: (i) The simulation model is deterministic, but has numerical noise caused by numerical approximations; see [15], p. 141. (ii) The simulation model is deterministic, but its inputs have uncertain values so these values are sampled from a prior input distribution; this procedure is called "uncertainty propagation". This uncertainty is called "epistemic", "subjective", or "the analysts' uncertainty"; see [20]. (iii) The simulation uses pseudo-random numbers (PRNs); examples are queueing simulations, which are "discrete event" simulation models. This uncertainty is called "aleatory", "objective", or "the system's inherent" uncertainty. Only a few publications consider discrete-event simulations with uncertain parameters, combining epistemic and aleatory uncertainties; for references see [24], p. 124.

To situate our own research within the general context of simulation-optimization, we give several references at the start of the various sections. Other references enable the readers to learn the details of our own methods. Future research per topic is also briefly mentioned in the

various sections; general problems concern the effects of a large number of inputs, and applications in real-life situations.

2 Basics of Kriging and bootstrapping

We assume that the readers are familiar with the basics of Kriging and bootstrapping, so we mainly define symbols and terminology in subsection 2.1 summarizing Kriging basics and in subsection 2.2 summarizing the basics of bootstrapping.

2.1 Kriging: basics

Originally, Kriging was developed by Daniel Krige—a South African mining engineer—for the interpolation of geostatistical (spatial) sampling data; see the classic 1993 textbook [8]. Later on, Kriging was applied to obtain a global (not local) metamodel for the I/O data of "computer experiments" with deterministic simulation models; see the classic 1989 article [44] and also the popular 2003 textbook [45]) and the 2008 textbook [15]. A recent survey of Kriging in simulation-based metamodeling is [25]. The literature on Kriging is vast and covers diverse disciplines, such as mechanical engineering, operations research, and statistics. The following website consists of 33 printed pages emphasizing machine learning:

http://www.gaussianprocess.org/

This site also gives alternative Kriging books such as [41] and [47].

There is much *software* for Kriging; see the preceding textbooks and website. In all our own experiments, however, we have used only DACE, which is a free-of-charge Matlab-toolbox well documented in [35]. Alternative free software is mentioned in [16] and [24], p. 146; also see the toolboxes called Surrogates and SUMO on

http://sites.google.com/site/felipeacviana/surrogatestoolbox

and

http://www.sumo.intec.ugent.be/?q=sumo_toolbox.

The statistical R community has also developed much software; see, e.g., [9]'s mlegp and [43]'s DiceKriging. Some publications focus on problems caused by large I/O data sets (so the matrix inversions in the equations 3 and 4 below become problematic); also see the topic called "approximations" on the website mentioned earlier; namely,

http://www.gaussianprocess.org/.

Kriging may give a valid metamodel (an adequate explicit approximation) of the implicit I/O function implied by the underlying simulation model, even when the simulation experiment covers a "big" input area so the simulation experiment is *global* (not local). For example, Kriging can approximate the I/O function of a simulation model for a traffic rate

ranging all the way between (say) 0.1 and 0.9; see Figure 3 discussed in Section 7.

"Ordinary Kriging"—simply called "Kriging" in the remainder of this paper—assumes that the I/O function being approximated is a realization of the *Gaussian Process*

$$Y(\mathbf{x}) = \mu + Z(\mathbf{x}) \tag{1}$$

where $\mathbf{x}$ is a point in a d-dimensional input space with d a given positive integer denoting the number of simulation input variables, μ is its constant mean, and $Z(\mathbf{x})$ is a stationary Gaussian stochastic process with mean zero, variance σ^2, and some correlation function. In simulation, the most popular correlation function is a product of the d individual correlation functions:

$$corr[Y(\mathbf{x}_i),Y(\mathbf{x}_j)] = \prod_{k=1}^{d} \exp(-\theta_k \left|x_{ik} - x_{jk}\right|^{p_k}) \text{ with } \theta_k > 0, p_k \geq 1. \tag{2}$$

The correlation function (2) implies that the outputs $Y(\mathbf{x}_i)$ and $Y(\mathbf{x}_j)$ are more correlated as their input locations $\mathbf{x}_i$ and $\mathbf{x}_j$ are "closer"; i.e., they have smaller Euclidean distance in the k^{th} dimension of the input combinations $\mathbf{x}_i$ and $\mathbf{x}_j$. The correlation parameter θ_k denotes the importance of input k; i.e., the higher θ_k is, the faster the correlation function decreases with the distance in this dimension. The parameter p_k determines the smoothness of the correlation function; e.g., $p_k = 2$ gives the so-called Gaussian correlation function, which gives smooth, continuous functions. Altogether, the Kriging *(hyper)parameters* are $\psi = (\mu, \sigma^2, \theta')'$with $\theta = (\theta_1, \ldots \theta_d)'$.

The Kriging parameters are selected using the best linear unbiased predictor (BLUP) criterion which minimizes the mean squared error (MSE) of the predictor. Given a set of n "old" observations (or training points) $\mathbf{y} = (y_1, \ldots, y_n)'$, it can be proven that this criterion gives the following *linear* predictor for a point $\mathbf{x}_{n+1}$ (sometimes denoted by $\mathbf{x}_0$), which may be either a new or an old point:

$$\widehat{y}(\mathbf{x}_{n+1}) = \mu + \mathbf{r}'\mathbf{R}^{-1}(\mathbf{y} - \mathbf{1}\mu) \tag{3}$$

where $\mathbf{r} = \{corr[Y(\mathbf{x}_{n+1}), Y(\mathbf{x}_1)], \ldots, corr[Y(\mathbf{x}_{n+1}), Y(\mathbf{x}_n)]\}'$ is the vector of correlations between the outputs at the point $\mathbf{x}_{n+1}$ and the n old points $\mathbf{x}_i$, $\mathbf{R}$ is the $n \times n$ matrix whose $(i,j)^{th}$ entry is given by (2), and $\mathbf{1}$ denotes the n-dimensional vector with ones. The correlation vector $\mathbf{r}$ and matrix $\mathbf{R}$ may be replaced by the corresponding covariance vector $\mathbf{\Sigma}_{n+1} = \sigma^2\mathbf{r}$ and matrix $\mathbf{\Sigma} = \sigma^2\mathbf{R}$, because σ^2 then cancels out in (3). If the new input combination $\mathbf{x}_{n+1}$ coincides with an old point $\mathbf{x}_i$, then

the predictor $\widehat{y}(\mathbf{x}_i)$ equals the observed value $y(\mathbf{x}_i)$); i.e., the Kriging predictor (3) is an *exact interpolator*. Notice that in Kriging we avoid extrapolation,; see [2], p. 9.

A major problem in Kriging is that its parameters ψ are unknown. In most simulation studies the analysts assume a Gaussian correlation function so $p_k = 2$ in (2). To estimate the parameters ψ, the standard literature and software use maximum likelihood estimators (MLEs), denoted by hats; so the MLE of ψ is $\widehat{\psi} = (\widehat{\mu}, \widehat{\sigma^2}, \widehat{\theta'})'$. Computing $\widehat{\psi}$ requires constrained maximization, which is a hard problem because matrix inversion is necessary, the likelihood function may have multiple local maxima and a ridge, etc.; see [37]. The MLE estimator of the mean μ turns out to be the generalized least squares (GLS) estimator $\widehat{\mu} = (\mathbf{1}^T\widehat{\mathbf{R}}^{-1}\mathbf{1})^{-1}\mathbf{1}^T\widehat{\mathbf{R}}^{-1}\mathbf{y}$ where $\widehat{\mathbf{R}}$ denotes the MLE of $\mathbf{R}$, which is determined by θ. It can be proven (see [15], p. 84) that the MSE of the BLUP or the *predictor variance* is

$$\sigma^2(\mathbf{x}) = \sigma^2(1 - \mathbf{r}'\mathbf{R}^{-1}\mathbf{r} + \frac{(1 - \mathbf{1}'\mathbf{R}^{-1}\mathbf{r})^2}{\mathbf{1}'\mathbf{R}^{-1}\mathbf{1}}) \tag{4}$$

where $\sigma^2(\mathbf{x})$ denotes the variance of $\widehat{y}(\mathbf{x})$ with $\widehat{y}(\mathbf{x})$ denoting the Kriging predictor at point $\mathbf{x}$; see (3).

The classic Kriging literature, software, and practice simply replace the unknown $\mathbf{R}$ and $\mathbf{r}$ in (3) and (4) by their estimators. Unfortunately, this substitution into (3) changes the linear predictor $\widehat{y}(\mathbf{x})$ into the *nonlinear* predictor, which we denote by $\widehat{\widehat{y}}(\mathbf{x}_{n+1})$ (with double hats). The classic literature ignores this complication, and simply plugs the estimates $\widehat{\sigma^2}$, $\widehat{\mathbf{r}}$, and $\widehat{\mathbf{R}}$ into the right-hand side of (4) to obtain (say) $s^2(\mathbf{x})$, the *estimated predictor variance* of $\widehat{\widehat{y}}(\mathbf{x})$. There is abundant Kriging software for the computation of the resulting Kriging predictor and predictor variance. Notice that the estimated predictor variance $s^2(\mathbf{x})$ is zero at the n old input locations, and tends to increase as the new location $\mathbf{x}_{n+1}$ lies farther away from old locations; we shall return to this behavior in Sections 3 and 4.

The interpolation property of Kriging is not desirable in *random* simulation, because the observed average output per scenario is noisy. Therefore the Kriging metamodel may be changed such that it includes *intrinsic* noise; see (in historical order) [45], pp. 215-249, [15], p. 143, [51], [1], and [6]. The resulting "stochastic Kriging" does not interpolate the n outputs averaged over the (say) m_i replicates for input combination i ($i = 1, \ldots, n$). Notice that [6] also accounts for common random numbers (CRN), which are used to simulate outputs for different input combinations; we shall return to CRN, in several sections. This stochastic Kriging may avoid overfitting; overfitting may result in a wiggling

(erratic) Kriging metamodel (also see Section 7).

More precisely, stochastic Kriging augments (1) with a *white noise* random variable (say) e:

$$Y(\mathbf{x}) = \mu + Z(\mathbf{x}) + e \tag{5}$$

where e is normally, independently, and identically distributed (NIID) with zero mean and (constant) variance σ_e^2. This e is generalized in [1] such that e has a variance that depends on the input combination $\mathbf{x}_i$; this is called "variance heterogeneity". And [6] accounts for CRN so the covariance matrix of e (say) $\mathbf{\Sigma_e}$ does no longer equal $\sigma_e^2\mathbf{I}$ (white noise) but becomes a covariance matrix with heterogeneous variances on the main diagonal (also see [51]) and positive covariances off this diagonal. The Kriging predictor (3) then becomes

$$\widehat{y}(\mathbf{x}_{n+1}) = \mu + \mathbf{\Sigma}'_{n+1}(\mathbf{\Sigma} + \mathbf{\Sigma}_{\overline{e}})^{-1}(\overline{\mathbf{y}} - \mathbf{1}\mu) \tag{6}$$

where $\mathbf{\Sigma}_{\overline{e}}$ is the covariance matrix of $\overline{e} = \sum_{j=1}^{m_i} e_{i;j}/m_i$ and $\overline{\mathbf{y}}$ is the n-dimensional vector with the output averages $\overline{y_i} = \sum_{j=1}^{m_i} y_{i;j}/m_i$ computed from the m_i replicates at point i ($i = 1, \ldots, n$). Notice that $\mathbf{\Sigma}_e = c\mathbf{I}$ is used to solve numerical problems in the computation of $\mathbf{R}^{-1}$; see [36], p. 12.

As far as software for stochastic simulation with white noise is concerned, [19] provides Matlab code and [45], pp. 215-249 provides C code. Matlab code for CRN is provided on

http://www.stochastickriging.net/.

2.2 Bootstrapping: basics

In general, the bootstrap is a simple method for quantifying the behavior of nonlinear statistics (such as $\widehat{\widehat{y}}(\mathbf{x})$); see the classic textbook on bootstrapping [14]. Its statistical properties such as asymptotic consistency are discussed in [7] and in the many references given in [24]. The bootstrap is a *data driven* method, so we suppose that a data set is given (say) $y_1, \ldots y_n$ and we assume that the n elements y_i ($i = 1, \ldots, n$) are IID, but not necessarily NIID. We consider the following two simple examples:

1. The y_i are *exponentially* distributed with parameter λ: $y_i \sim Exp(\lambda)$.

2. The statistic of interest is *nonlinear*; namely, the estimated skewness $\sum_{i=1}^{n}(y_i - \overline{y})^3/[(n-1)s^3]$ with sample average $\overline{y}$ and sample standard deviation s.

Suppose that in Example 1 we are interested in the distribution of the sample average $\overline{y}$. If the y_i were NIID with mean μ and standard deviation σ—denoted by $y_i \sim NIID(\mu, \sigma)$— then it is well known that the average would have a normal distribution $N(\mu, \sigma/\sqrt{n})$. In this example, however, we assume $y_i \sim Exp(\lambda)$. We may then estimate the parameter λ from y_i; e.g., $\widehat{\lambda} = 1/\overline{y}$. Next we can sample n new observations (say) y_i^* from $Exp(\widehat{\lambda})$: this is called *parametric bootstrapping*, which is Monte Carlo sampling with the parameter λ estimated from the data y_i; the superscript $*$ is the usual symbol denoting bootstrapped observations. From these bootstrapped observations y_i^* we compute the statistic of interest, $\overline{y^*} = \sum_{i=1}^{n} y_i^*/n$. To estimate the empirical density function (EDF) of $\overline{y^*}$, we repeat this resampling (say) B times, where B is called the "bootstrap sample size"; a typical value is $B = 100$. If we wish to estimate a (say) 90% confidence interval (CI) for the population mean $E(y)$, then the simplest method uses the "order statistics" $\overline{y^*_{(b)}}$ with $b = 1, ..., B$ so $\overline{y^*_{(1)}} < \overline{y^*_{(2)}} < ... < \overline{y^*_{(n-1)}} < \overline{y^*_{(n)}}$; i.e., this CI is $(\overline{y^*_{(0.05B)}}, \overline{y^*_{(0.95B)}})$ assuming $0.05B$ and $0.95B$ are integers (otherwise rounding is necessary); see [14], pp. 170-174.

Now suppose we do not know which type of distribution y_i has. Furthermore, n is too small for a reliable estimate of the distribution type, such as an exponential distribution type. In this case, we can apply *distribution-free* or *nonparametric bootstrapping*, as follows. Using PRN, we resample the n "original" observations y_i with replacement; e.g., we might sample y_1 zero times, or only once, or even n times (if we sample the same value n times, then obviously none of the other values is resampled). Notice that we shall detail this sampling in Step 1 in Section 7. From these resampled or bootstrapped observations y_i^* we compute the statistic of interest, which in this example is $\overline{y^*} = \sum_{i=1}^{n} y_i^*/n$. Like in parametric bootstrapping, we can compute the EDF of $\overline{y^*}$ through repeating this resampling B times; this gives a CI.

Obviously, we can apply bootstrapping to estimate the EDF of more complicated statistics than the average; e.g., the skewness in Example 2—or the Kriging predictor with estimated parameters (see the next section).

Bootstrapping has become popular since powerful and cheap computers have become widely available. *Software* for bootstrapping is available in many statistical software packages, including the BOOT macro in SAS and the "bootstrap" command in S-Plus; see [39]. Bootstrapping is also easily implemented in Matlab—which is what we do in all our applications.

3 Bootstrapped variance of Kriging predictor

We expect the true variance to be underestimated by $s^2(\mathbf{x})$, which is the estimated predictor variance in deterministic Kriging with estimated Kriging parameters defined in Section 2.1. Therefore [13] derives a bootstrapped estimator (an alternative is the Bayesian approach derived in [51]). This estimator uses *parametric* bootstrapping assuming the deterministic simulation outputs Y are realizations of the Gaussian Process defined in (1). This bootstrapping first computes (say) $\widehat{\psi} = (\widehat{\mu}, \widehat{\sigma^2}, \widehat{\theta'})'$, which denotes the MLEs of the Kriging parameters computed from the "original" old I/O data $(\mathbf{X}, \mathbf{y})$ where $\mathbf{X}$ is the $n \times d$ input matrix with rows $\mathbf{x}_i' = (x_{i1}, ..., x_{id})$ and $\mathbf{y} = (y_1, ..., y_n)'$ is the corresponding output vector. The DACE software is used by [13] to compute these MLEs (different software may give different estimates because of the difficult constrained maximization required by MLE). These MLEs specify the distribution from which to sample *bootstrapped* observations.

Note: The bootstrap algorithm that [13] calls "adding new points one at a time" considers many prediction points, but a single point is added one-at-a-time to the n old points. Unfortunately, this algorithm turns out to give bumpy plots for the bootstrapped Kriging variance as a function of a one-dimensional input; see Figure 3 in [13].

To estimate the MSE of the Kriging predictor at the new point $\mathbf{x}_{n+1}$, [13] samples *both* the n bootstrap outputs $\mathbf{y}^* = (y_1^*, ..., y_n^*)'$ at the old input combinations $\mathbf{X}$ *and* y_{n+1}^* at the new point $\mathbf{x}_{n+1}$. These $n+1$ outputs—collected in $\mathbf{y}_{n+1}^{*\prime} = (\mathbf{y}^{*\prime}, y_{n+1}^*)$—are correlated; more precisely,

$$\mathbf{y}_{n+1}^* \sim N_{n+1}(\widehat{\mu}_{n+1}, \widehat{\mathbf{\Sigma}}_{(n+1)\times(n+1)}) \tag{7}$$

where the mean vector $\widehat{\mu}_{n+1}$ has all its $(n+1)$ elements equal to $\widehat{\mu}$ and the (symmetric positive semi-definite, PSD) $(n+1) \times (n+1)$ covariance matrix equals

$$\begin{bmatrix} \widehat{\mathbf{\Sigma}} & \widehat{\mathbf{\Sigma}_{n+1}} \\ \widehat{\mathbf{\Sigma}_{n+1}}' & \widehat{\sigma^2} \end{bmatrix}$$

with symbols defined below (3). The bootstrapped data for the points $(\mathbf{X}, \mathbf{y}^*)$ resulting from (7) give the bootstrapped MLE $\widehat{\psi}^* = (\widehat{\mu}^*, \widehat{\sigma^2}^*, \widehat{\theta}^{*\prime})'$; [13] starts the search for this $\widehat{\psi}^*$ from $\widehat{\psi}$ (the MLE based on the original data $(\mathbf{X}, \mathbf{y})$). This $\widehat{\psi}^*$ gives the bootstrapped Kriging predictor for the new point, $\widehat{\widehat{y}}_{n+1}^*$.

The squared errors (SEs) at these old points are zero, because classic Kriging is an exact interpolator; however, the squared error at the new point is

$$SE_{n+1} = (\widehat{\widehat{y}}_{n+1}^* - y_{n+1}^*)^2 \tag{8}$$

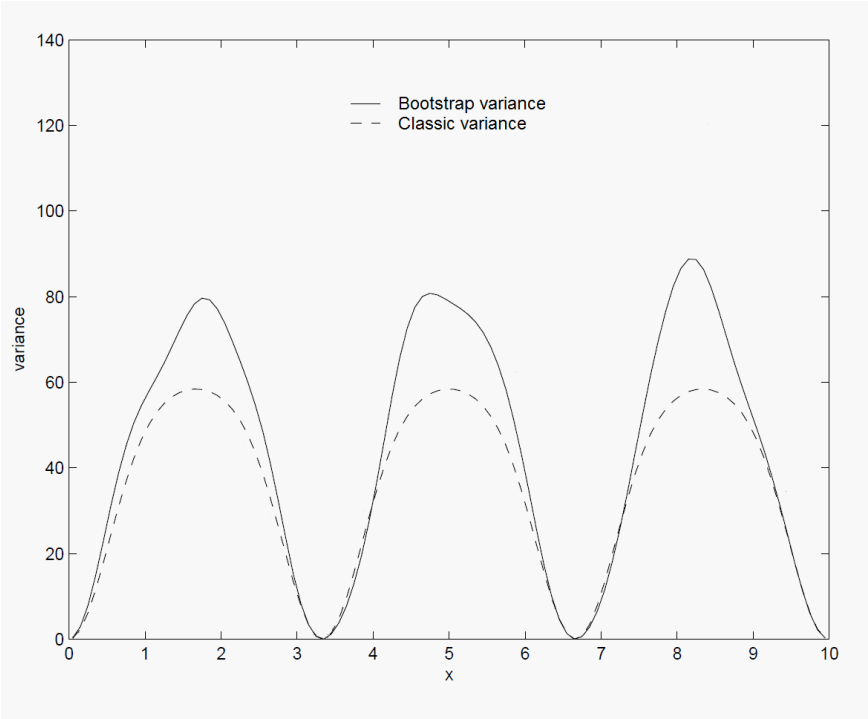

Figure 1: Classic versus bootstrap estimates of Kriging predictor variance at input x

where y^*_{n+1} results from (7).

To reduce sampling error, bootstrapping is repeated B times (e.g., $B = 100$), which gives $\widehat{y}^*_{n+1;b}$ and $y^*_{n+1;b}$ with $b = 1, \ldots, B$. This gives *the* bootstrap estimator of the Kriging predictor's variance:

$$s^2(\widehat{y}^*_{n+1}) = \frac{\sum_{b=1}^{B} (\widehat{y}^*_{n+1;b} - y^*_{n+1;b})^2}{B}, \tag{9}$$

which uses the classic bootstrap denominator B instead of $B - 1$.

Several examples are given in [13]; namely, four mathematical functions in one or two dimensions, and one circuit-simulator taken from [44] with $n = 32$ observations and $d = 6$ dimensions. Figure 1 is reproduced from [13], and gives a typical result; namely, (i) the true variance seems underestimated by the classic estimator, and (ii) the classic and the bootstrapped estimates do not reach their maximum at the same input value. This second characteristic may make the bootstrapped variance estimator (9) useful in EGO, as we explain next.

4 EGO with bootstrapped variance

EGO is the topic of many recent publications; see the forthcoming publication [40] in *Technometrics* including invited comments such as [26]; also see [15], pp. 125-131, 141-153, [17], and [49]. Those publications generalize EGO to random simulation and constrained optimization, whereas classic EGO assumes deterministic simulation aimed at finding the unconstrained *global* minimum of the objective function. We limit our discussion to this classic EGO, which uses the Kriging predictor $\widehat{\widehat{y}}$ and its classic estimated predictor variance $s^2(\mathbf{x})$ defined in Section 2.1. EGO is meant to balance local and global search, also called *exploitation* and *exploration*. The classic reference for EGO (including predecessors of EGO) is the 1998 article [21]; a recent and in-depth discussion of classic EGO is [15], pp. 90-101.

EGO uses the following steps:

1. Find among the n *old* simulation outputs y_i $(i = 1, \ldots, n)$ the *minimum*, denoted by $\min_i y_i$.

2. Estimate the input combination $\mathbf{x}$ that maximizes $\widehat{EI(\mathbf{x})}$, the estimated *expected improvement* (EI) compared with $\min_i y_i$ found in Step 1:

$$\max_{\mathbf{x}} \widehat{EI(\mathbf{x})} = \int_{-\infty}^{\min_i y_i} [\min_i y_i - y(\mathbf{x})] f[y(\mathbf{x})] dy(\mathbf{x}) \qquad (10)$$

   where $f[y(\mathbf{x})]$ denotes the distribution of $\widehat{\widehat{y}}(\mathbf{x})$ (Kriging predictor with MLE $\widehat{\psi}$, for input combination $\mathbf{x}$). EGO assumes that this distribution is Gaussian with estimated mean $\widehat{\widehat{y}}(\mathbf{x})$ and the classic variance $s^2(\mathbf{x})$. To find the *maximizer* of (10), EGO may use either a space-filling design with *candidate* points or a *global optimizer* (GA) such as the genetic algorithm in [15], p. 78. (A *local* optimizer is undesirable, because $\widehat{EI(\mathbf{x})}$ may have many local optima; i.e., all old input combinations show $s^2(\mathbf{x}) = 0$ so $\widehat{EI(\mathbf{x})} = 0$; see again Figure 1.)

3. *Simulate* the maximizing combination found in Step 2, *refit* the Kriging model to the old and new I/O data, and *return* to Step 1—unless the global minimum seems reached close enough because $\max_{\mathbf{x}} \widehat{EI(\mathbf{x})}$ is "close" to zero.

Recently, [31] uses the bootstrap estimator $s^2(\widehat{\widehat{y}}^*_{n+1})$ defined in (9) to compute the EI in (10), replacing the general distribution $f[\widehat{\widehat{y}}(\mathbf{x})]$

by $N[\widehat{\widehat{y}}_{n+1}, s^2(\widehat{\widehat{y}}^*_{n+1})]$. The procedure uses candidate points, not GA. For all candidate points the bootstrapped predictions use the same bootstrapped MLE $\widehat{\psi}^*$, computed from the I/O data $(\mathbf{x}, \mathbf{y}^*)$.

Note: To speed-up the computations of $s^2(\widehat{\widehat{y}}^*_{n+1})$ for the many candidate points, [31] uses the property that the multivariate normal distribution (7) implies that its conditional output is also normal. So, let $\mathbf{y}^*$ still denote the bootstrap outputs of the n old input combinations, and y^*_{n+1} the bootstrap output of a candidate combination. Then (7) implies that the distribution of this y^*_{n+1}—given (or "conditional on") $\mathbf{y}^*$—is (also see equation 19 in [13])

$$N(\widehat{\mu} + \widehat{\boldsymbol{\Sigma}_{n+1}}' \widehat{\boldsymbol{\Sigma}}^{-1}(\mathbf{y}^* - \widehat{\mu}), \widehat{\sigma^2} - \widehat{\boldsymbol{\Sigma}_{n+1}}' \widehat{\boldsymbol{\Sigma}}^{-1} \widehat{\boldsymbol{\Sigma}_{n+1}}). \tag{11}$$

This formula may be interpreted as follows. If (say) all n elements of $\mathbf{y}^* - \widehat{\mu}$ (see the first term, representing the mean) happen to be positive, then it may be expected that y^*_{n+1} is also "relatively" high ($\widehat{\boldsymbol{\Sigma}_{n+1}}$ has positive elements only); i.e., higher than its unconditional mean $\widehat{\mu}$; similarly, if all elements of $\mathbf{y}^* - \widehat{\mu}$ happen to be negative, then it may be expected that y^*_{n+1} is lower than its unconditional mean $\widehat{\mu}$. The second term implies that y^*_{n+1} has a lower variance than its unconditional variance $\widehat{\sigma^2}$ if $\mathbf{y}$ and y_{n+1} show high positive correlations; see $\widehat{\boldsymbol{\Sigma}_{n+1}}$.

Moreover, [31] estimates the effects of the sample size n on the difference between the classic and the bootstrap estimates of the predictor variance. The empirical results suggest that the smaller n is, the more the classic estimator underestimates the true variance. Unfortunately, a "small" n—given the number of dimensions d and the unknown shape of the I/O function—increases the likelihood of an inadequate Kriging metamodel so the Kriging (point) predictor $\widehat{\widehat{y}}(\mathbf{x})$ may be misleading; i.e., this wrong predictor combined with a correct predictor variance may give a wrong EI leading to the—possibly expensive—simulation of the wrong next point.

Finally, [31] empirically compares classic and bootstrapped EGO. Four test functions are used, with d equal to 1, 2, 3, and 6. The bootstrapped EGO performs better for three of the four test functions; the one remaining test function gives a tie. Nevertheless, the analysts might wish to stick to classic EGO, in case they accept some possible inefficiency—compared with bootstrapped EGO—and prefer the simple analytical computations of classic EGO—compared with the sampling required by bootstrapped EGO. So we might conclude that classic EGO gives a quite *robust* heuristic—possibly because the bias of the classic variance estimator decreases as the sample size increases so this estimator approaches the bootstrap variance estimator; both approaches use

the same point predictor, $\widehat{\widehat{y}}(\mathbf{x})$.

5 Constrained optimization in random simulation

The following approach is not guided by EGO, but is more related to classic operations research (OR). More precisely, [30] derives a heuristic for constrained optimization in random simulation. This heuristic is applied to the academic (s, S) inventory system in [3] and a complicated call-center simulation in [23]. These two applications minimize one output (namely, costs), while satisfying a constraint for another output (service percentage, also called fill rate); moreover, the call-center simulation must satisfy a budget constraint for the deterministic inputs (namely, resources) which must be non-negative integers.

These two applications are examples of the general problem formulated in (12) below, which shows that the output $E(y_0)$ is the objective to be minimized, while the other $(r-1)$ outputs must satisfy prespecified threshold values c_h $(h = 1, \ldots, r-1)$, and the d deterministic simulation inputs x_j $(j = 1, \ldots, d)$ must satisfy s linear or nonlinear constraints f_g (e.g., budget constraints), and x_j must belong to the set of non-negative integers $\mathbf{N}$:

$$\begin{aligned} Min_{\mathbf{x}} E(y_0) \\ E(y_h) \geq c_h \ (h = 1, \ldots, r-1) \\ f_g(x_1, \ldots, x_d) \geq c_g \ (g = 1, \ldots, s) \\ x_j \in \mathbf{N} \ (j = 1, \ldots, d). \end{aligned} \tag{12}$$

To solve this type of problems, [30] develops a heuristic that combines

- sequentialized design of experiments (DOE) to specify the next simulation input combination (the preceding section shows that EGO also uses such DOE);
- Kriging to analyze the simulation I/O data (like EGO does);
- INLP to estimate the optimal solution from the explicit Kriging metamodels for the outputs $E(y_0)$ and $E(y_h)$.

The heuristic is composed of modules that use free off-the-shelf software. These components may be replaced as the knowledge in DOE, Kriging, and INLP evolves. Kriging may be replaced by other types of metamodels; e.g., radial basis functions as in [42]. Applications may have continuous inputs, so INLP may be replaced by a solver that uses the gradients, for which Kriging gives estimates "for free"; see [35] and

also Section 7. Future research may adapt the heuristic for deterministic simulations with constrained multiple outputs and inputs.

Finally, [30] compares the results with those of the popular commercial heuristic OptQuest—based on tabu search—embedded in the Arena discrete-event simulation software (see [23]); the new heuristic outperforms OptQuest in terms of the number of simulated input combinations and the quality of the estimated optimum.

The various steps of the heuristic are summarized in Figure 2, reproduced from [30]; block 10 in this figure uses the parameter $a = 30$. The figure shows that the heuristic sequentially updates the initial design; i.e., it adds additional points in steps 5 and 9 respectively. Points in step 5 should improve the metamodel, whereas points in step 9 should find the optimum—similar to "exploration" and "exploitation" in EGO and in several other discrete-event simulation optimization heuristics surveyed in [18]. The global Kriging metamodels should be accurate enough to enable INLP to identify clearly infeasible points—which violate the constraints on the random simulation outputs y_h—and suboptimal points—which generate a goal output y_0 that is too high. The heuristic may add points throughout the entire input-feasible area: exploration. The global Kriging metamodel for output h uses all observations for this output, obtained so far. To guide the INLP search, the heuristic simulates each point with required precision, to be reasonably certain of the objective values and the possible violation of the constraints; i.e., the heuristic selects the number of replicates m_i such that the halfwidth of the 90% CI for the average simulation output is within 15% of the true mean for all r outputs; also see [32], pp. 500-503. The heuristic uses CRN to improve the estimate of the optimum solution. The heuristic applies Kriging to the average output per simulated input combination, and does so for each of the r types of output.

We detail Step 4, because this step uses *bootstrapping*. This step applies the following six substeps.

1. From the set of simulation I/O data, delete one input combination at a time—but avoid extrapolation, so (say) n_{cv} are deleted successively; the subscript cv stands for cross-validation.

2. Based on the remaining I/O data, compute $y_h(-\mathbf{x}_i)$, which denotes the Kriging predictor for output h ($h = 0, \ldots, r-1$) of the deleted (see the minus sign) input combination i; we stick to the symbols in [30], so we use y instead of $\widehat{\widehat{y}}$ used in (8). Do not re-estimate the correlation functions, because the current estimates based on all observations are more reliable; also see [21] and [22].

3. Use *distribution-free bootstrapping* to obtain $\widehat{var(y_h^*(\mathbf{x}_i))}$, which denotes the bootstrapped estimator of the predictor variance for output h at the deleted combination $\mathbf{x}_i$. This bootstrapping accounts for the following three complications: (i) every replicate of a given point $\mathbf{x}_i$ gives a *multi*variate output vector (namely, r-variate output); (ii) *CRN* make the output vectors of the same replicate of the simulated input combinations positively correlated (also see [27]; (iii) m_i (number of replicates at input combination i) may *vary* with i because of the relative precision requirement; details are given in [30].

4. Use $\widehat{var(y_h^*(\mathbf{x}_i))}$ computed in substep 3, to compute the *Studentized* prediction errors for every output h of the i^{th} deleted combination:

$$t_{m_i-1}^{h,i} = \frac{\overline{y_h(\mathbf{x}_i)} - y_h(-\mathbf{x}_i)}{\sqrt{\widehat{var(\overline{y_h(\mathbf{x}_i)})} + \widehat{var(y_h^*(\mathbf{x}_i))}}} \; (h = 0, \ldots, r-1) \; (i = 1, \ldots, n_{cv}) \tag{13}$$

where $\widehat{var(\overline{y_h(\mathbf{x}_i)})} = \sum_{r=1}^{m_i} [y_{i;h;r} - \overline{y_h(\mathbf{x}_i)}]^2/[(m_i - 1)m_i]$, which is the classic variance estimator based on m_i replicates.

5. Repeat the preceding four substeps, until all n_{cv} combinations have been deleted one-at-a-time.

6. Find $\max \left| t_{m_i-1}^{h,i} \right|$, which denotes the highest absolute value of the $t_{m_i-1}^{h,i}$ computed in (13) over all r outputs and all n_{cv} cross-validated input combinations. Determine if this value is statistically significant using Bonferroni's inequality; this inequality implies that the traditional type-I error rate α is divided by $r \times n_{cv}$. If $\max \left| t_{m_i-1}^{h,i} \right|$ is significant, then all r Kriging models are rejected; else, the metamodels are considered to be valid, and will be used by INLP.

Note: Step 5 in Figure 2 augments the design with a new combination to improve the Kriging models, if the Kriging models are rejected. The significant $\max \left| t_{m_i-1}^{h,i} \right|$ is given by the so-called "worst point", so in the neighborhood of that point the heuristic requires extra information about the r metamodels. However, Kriging assumes that input combinations near each other have outputs with high positive correlations, so little new information would result from simulating a point close to the worst point—or close to any other point in the current design. The heuristic therefore selects the point halfway between the worst point and

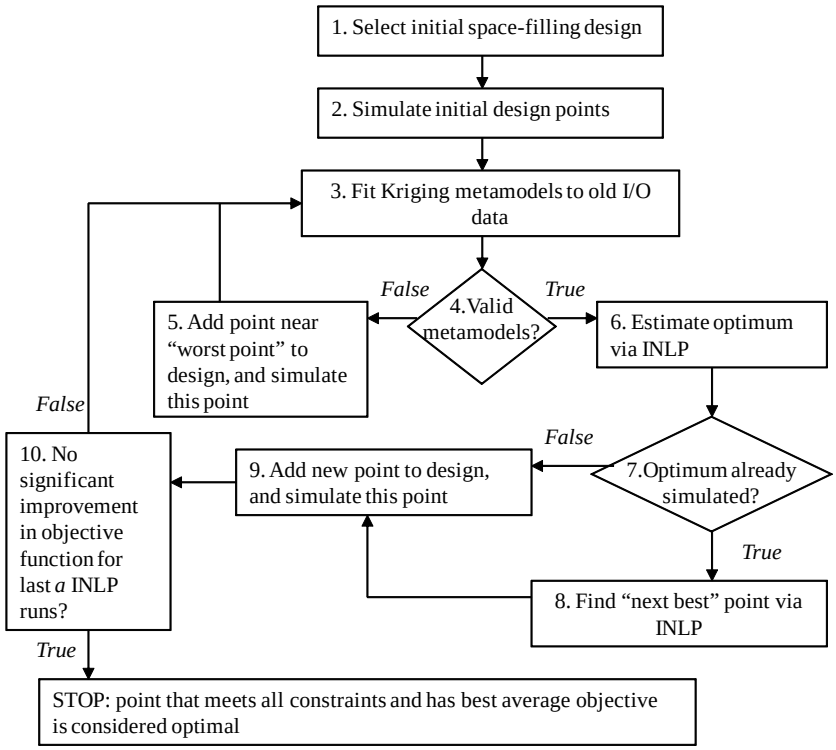

Figure 2: Overview of Kleijnen, Van Beers, and Van Nieuwenhuyse (2010)'s heuristic

its nearest neighbor in the current design; selecting a point "halfway" resembles EGO. Step 6 in Figure 2 uses a free Matlab branch-and-bound INLP solver called *bnb20.m*. A disadvantage of this solver is that it guarantees only local optimality, so it needs multiple starting points; the heuristic uses three starting points. INLP may give a previously simulated point as the optimum. In that case, Step 8 reruns INLP with the additional constraint that it should return a point that is not yet part of the design; this point is called the "next best point".

6 Taguchian robust optimization in simulation

In practice, at least some inputs of a given simulation model are *uncertain* so it may be wrong to use the optimum solution that is derived ignoring these uncertainties. Decision-making in such an uncertain world may use Taguchi's approach (see [48]), originally developed to help Toyota design *robust* cars; i.e., cars that perform reasonably well in many different circumstances in real life. Notice that the Taguchian approach

differs from robust optimization in mathematical programming, which was initiated by Ben-Tal (see [4]); the latter approach is discussed in a simulation context by [50].

Taguchian robust simulation-optimization is studied in [12], replacing Taguchi's low-order polynomial metamodels by Kriging metamodels; moreover, bootstrapping is applied to quantify the variability in the estimated Kriging metamodels. Instead of Taguchi's signal/noise criterion—the ratio of the mean and the variance of the output— [12] combines Kriging with nonlinear programming (NLP), which is also used in our Section 5. Changing specific threshold values in the NLP model (14) below, the Pareto frontier can be estimated. An illustration of the resulting methodology is a deterministic economic order quantity (EOQ) inventory simulation with an uncertain input; namely, an uncertain demand rate. For this example, it turns out that robust optimization requires an order quantity that differs from the classic EOQ.

More precisely, Taguchi distinguishes between two types of factors (inputs, parameters, variables) $\mathbf{x}$: (i) *decision* (or control) factors $\mathbf{d} = (d_1, \ldots, d_k)$, which managers can control; e.g., in inventory management, the order quantity is controllable; and (ii) *environmental* (or noise) factors $\mathbf{e} = (e_1, \ldots, e_c)$, which are beyond management's control; an example is the demand rate in inventory management. Notice that we now use symbols that are not exactly the same as the symbols in the preceding sections.

Taguchi's *statistical* methods are criticized by many statisticians; see the panel discussion in [38]. Therefore [12] uses Kriging including Latin hypercube sampling (LHS). Kriging is better in computer simulation experiments because the experimental area may be much larger than in Taguchi's real-life experiments, so in simulation a low-order polynomial may be a non-valid metamodel. LHS gives space-filling designs; references and websites for various space-filling designs are given in [24], pp. 127-130.

Whereas Taguchi focuses on the signal/noise ratio, [12] uses the following NLP model:

$$Min_{\mathbf{d}} E(w|\mathbf{d}) \text{ such that } s_w \leq T \tag{14}$$

where $E(w|\mathbf{d})$ is the mean of the simulation output w defined by the distribution function of the environmental variables $\mathbf{e}$; this mean is controlled through the decision factors $\mathbf{d}$; s_w is the standard deviation of the goal output w and must meet a given constraint value T. Unlike the variance, the standard deviation has the same scale as the mean. Next, $E(w|\mathbf{d})$ and s_w is replaced by their Kriging approximations. Notice that the constrained minimization problem (14) is nonlinear in the decision

variables $\mathbf{d}$. Changing the threshold value T gives an estimate of the Pareto-optimal efficiency frontier; i.e., $E(w|\mathbf{d})$ and s_w are the criteria requiring a trade-off.

In general, simulation analysts often use LHS to obtain the I/O simulation data to which Kriging models are fitted. Actually, [12] uses the following two approaches (which we shall detail below):

1. Similar to [11], fit two Kriging metamodels; namely, one model for the mean and one for the standard deviation—both estimated from the *simulation* I/O data.

2. Similar to [33], fit a single Kriging metamodel to a relatively small number (say) n of combinations of $\mathbf{d}$ and $\mathbf{e}$; next use this metamodel to compute the *Kriging predictions* for the simulation output w for $N >> n$ combinations of $\mathbf{d}$ and $\mathbf{e}$ accounting for the distribution of $\mathbf{e}$.

Sub 1: We detail approach 1 as follows. Start with selecting the input combinations for the simulation model through a *crossed* (combined) design for $\mathbf{d}$ and $\mathbf{e}$—as is also traditional in Taguchian design; i.e., combine the (say) n_d combinations of $\mathbf{d}$ with the n_e combinations of $\mathbf{e}$ (an alternative would be the split-plot design in [10]). These n_d combinations are space-filling, to avoid extrapolation. The n_e combinations are *sampled* from the distribution of $\mathbf{e}$, using LHS for this (stratified) sampling. The resulting I/O data form an $n_d \times n_e$ table or matrix, and enables the following estimators of the n_d conditional means and variances:

$$\overline{w_i} = \frac{\sum_{j=1}^{n_e} w_{ij}}{n_e} \ (i = 1, \ldots, n_d) \tag{15}$$

and

$$s_i^2(w) = \frac{\sum_{j=1}^{n_e} (w_{ij} - \overline{w_i})^2}{n_e - 1} \ (i = 1, \ldots, n_d). \tag{16}$$

These two estimators are unbiased, because they do not use any metamodels.

Sub 2: Start with selecting a relatively small n (number of input combinations) using a space-filling design for the $k + c$ factors $\mathbf{d}$ and $\mathbf{e}$; i.e., $\mathbf{e}$ is not yet sampled from its distribution. Next, use these $n \times (k + c)$ simulation input data and their corresponding n outputs w to fit a Kriging metamodel for the output w. Finally, for a much larger design with N combinations, use a space-filling design for $\mathbf{d}$ but use LHS for $\mathbf{e}$ accounting for the distribution of $\mathbf{e}$. Compute the Kriging predictors $\widehat{y}$ (or $\widehat{\widehat{y}}$ in the symbols of Section 2.1) for the N outputs. Then derive the conditional means and standard deviations using (15) and (16) replacing

n_e and n_d by N_e and N_d and replacing the simulation output w by the Kriging predictor $\widehat{y}$. Use these predictions to fit two Kriging metamodels; namely, one Kriging model for the mean output and one for the standard deviation of the output.

Sub 1 and 2: Combining the Kriging metamodels with the NLP model (14) and varying the threshold T gives the estimated Pareto frontier. This frontier, however, is built on estimates of the mean and standard deviation of the simulation output. To quantify the variability in the estimated mean and standard deviation, apply *distribution-free bootstrapping*. Moreover, bootstrapping assumes that the original observations are IID; however, the crossed design for $\mathbf{d}$ and $\mathbf{e}$ implies that the n_d observations on the output for a given combination of the c environmental factors $\mathbf{e}$ are not independent (this dependence may be compared with the dependence created by CRN).

Technically, the n_d-dimensional vectors $\mathbf{w}_j$ $(j = 1, \ldots, n_e)$ are resampled n_e times with replacement. This resampling gives the n_e bootstrapped observations $\mathbf{w}_j^*$. This gives the bootstrapped conditional means $\overline{w_i}^*$ and standard deviations s_i^*. To these $\overline{w_i}^*$ and s_i^* Kriging is applied. These Kriging metamodels together with NLP give the predicted optimal bootstrapped mean and standard deviation. Repeating this bootstrap sampling B times gives CIs. More research seems necessary to discover how exactly to use these CIs to account for management's risk attitude.

Future research may also address the following issues. Instead of minimizing the mean under a standard-deviation constraint as in (14), we may minimize a specific quantile of the simulation output distribution or minimize the "conditional value at risk" (CVaR). Other risk measures are the "expected shortfall at level p", which is popular in the actuarial literature. Kriging may be replaced by "generalized linear models" (GLM) and NLP by evolutionary algorithms (EAs). The methodology may also accommodate random simulation models, which imply aleatory uncertainty besides epistemic uncertainty (see again Section 1).

7 Convex and monotonic bootstrapped Kriging

As we discussed in the preceding sections, simulation-optimization may concern either a single or multiple outputs. In case of a single output, the analysts often assume a *convex* I/O function; see the classic textbook on convex optimization [5]. An example is the newsvendor problem discussed in [46]. In case of multiple outputs, the analysts may minimize one output while satisfying constraints on the other outputs. An example is the call center in Section 5, in which the costs are to be minimized while the service percentage should be at least 90% . It is realistic to assume that the mean service percentage is a *monotonically* increasing

function of the (costly) resources.

A major problem is that simulation models do not have *explicit* I/O functions, so all the analysts can do is run the simulation model for various input combinations and observe the output. Next they may fit a Kriging metamodel to these observed I/O combinations. This Kriging metamodel provides an explicit I/O function that is assumed to be an adequate approximation of the implicit I/O function of the underlying simulation model.

In this section we present both monotonicity-preserving bootstrapped Kriging metamodels and convexity-improving bootstrapped Kriging metamodels. Actually, [29] applies monotonicity-preserving bootstrapped metamodels to single-server simulation models, to improve sensitivity analysis rather than optimization. And [28] applies convexity-improving bootstrapped Kriging to various inventory simulation models to find optimal solutions. In practice, simulation analysts may indeed know that the I/O function is monotonic; e.g., as the traffic rate increases, so does the mean waiting time; as the order quantity increases, so does the mean service percentage. We assume random simulation with replicates, so distribution-free bootstrapping can be applied; in deterministic simulation we could apply parametric bootstrapping, but we do not discuss the latter situation any further.

In the next subsection we discuss monotonicity, even though the example concerns sensitivity analysis instead of simulation. Nevertheless, this subsection gives details including a procedure that will be used in the subsection on convexity, which is a crucial concept in optimization.

7.1 Monotonicity

To obtain a monotonic Kriging metamodel, [29] resamples the m_i replicated outputs $w_{i;r}$ ($r = 1, ..., m_i$) for factor combination i ($i = 1, ..., n$), and fits a Kriging metamodel to the resulting n bootstrapped averages $\overline{w_i^*}$. Notice that this procedure allows variance heterogeneity of the simulation outputs. The fitted Kriging metamodel is *accepted* only if it is monotonically increasing for all n old combinations and for a set of (say) n_c candidate combinations; the latter are selected through LHS. Monotonicity implies that the gradients at all these combinations are positive; the DACE software proves estimates of all these gradients. Notice that this bootstrapped Kriging metamodel does not interpolate the original average output $\overline{w_i}$ (it does interpolate $\overline{w_i^*}$). This bootstrapping is repeated B times, giving $\overline{w_{i;b}^*}$ with $b = 1, ..., B$ and the corresponding Kriging metamodels, etc.

To illustrate and evaluate this method, [29] uses a popular *single-server* simulation model; namely, a model with exponential interarrival

and service times resulting in a Markovian model called M/M/1. The output is either the mean or the 90% quantile of the waiting time distribution. Empirical results demonstrate that—compared with classic Kriging—monotonic bootstrapped Kriging gives a higher probability of covering the true outputs, without lengthening the CIs.

Note: This Kriging implies sensitivity analysis that is better understood and accepted by the clients of the simulation analysts so the decision-makers trust the simulation as a decision support tool. Furthermore, estimated gradients with correct signs may improve simulation optimization, but this issue is not explored in [29].

Figure 3 gives an M/M/1 example with $m_i = 5$ replicates per traffic rate; the Kriging metamodel uses the Gaussian correlation function (2). This figure assumes that if the analysts require monotonicity for the simulation model's I/O function, then they should obtain so many replicates that the n average simulation outputs $\overline{w_i}$ also show this property; see the figure. Technically, however, monotonic bootstrap Kriging has a weaker requirement than $\overline{w_i} < \overline{w_{i+1}}$; namely, $\min_i w_i < \max_i w_{i+1}$.

The *procedure* for this bootstrap consist of the following major steps, assuming no CRN but allowing different numbers of replicates (because of variance heterogeneity):

1. Resample—with replacement—a replicate number r^* from the uniform distribution defined on the integers $1, \ldots, m_i$; i.e., the uniform density function is $p(r^*) = 1/m_i$ with $r^* = 1, \ldots, m_i$.

2. Replace the r^{th} "original" output $w_{i;r}$ by the bootstrap output $w^*_{i;r} = w_{i;r*}$.

3. Compute the Kriging predictor y^* from $(\mathbf{X},\overline{\mathbf{w}}^*)$ where $\mathbf{X}$ denotes the $n \times d$ matrix with the n old combinations of the d simulation inputs and $\overline{\mathbf{w}}^*$ denotes the n-dimensional vector with the bootstrap averages $\overline{w_i}^* = \sum_{r=1}^{m_i} w^*_{i;r}/m_i$ and $i = 1, \ldots, n$. This predictor uses the MLE $\widehat{\theta}^*$ computed from this $(\mathbf{X},\overline{\mathbf{w}}^*_i)$. Notice that we use [29]'s symbol y rather than our symbol $\widehat{\widehat{y}}$ defined in Section 2.1.

4. Accept only the monotonically increasing Kriging predictor y^*; i.e., all d components of the gradients at the $n + n_c$ old and new (candidate) points are positive:

$$\nabla y^*_i > \mathbf{0} \ (i = 1, \ldots, n + n_c). \tag{17}$$

This procedure is repeated B times, but it keeps only the (say) $A \leq B$ predictors that satisfy (17) so $\nabla y^*_{i;a} > \mathbf{0}$ $(a = 1, ..., A)$. For the new input

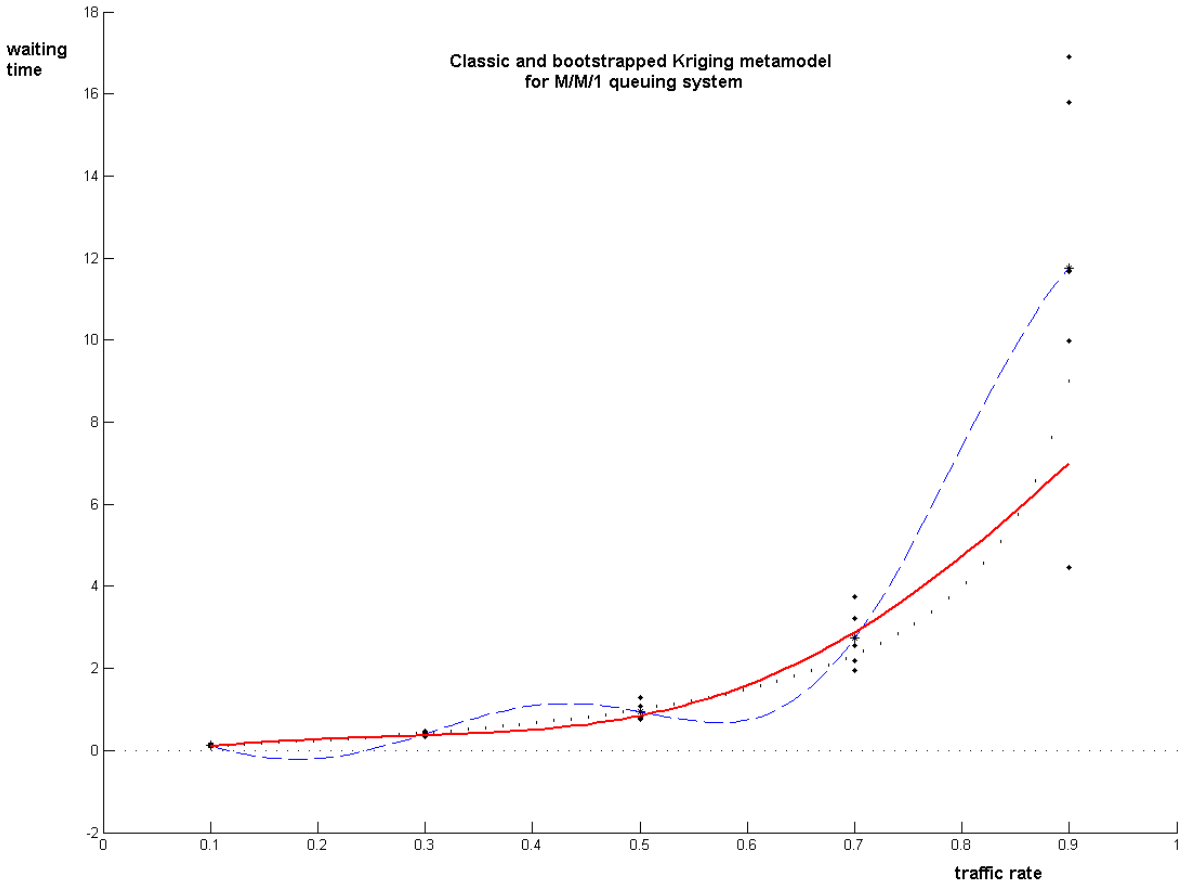

Figure 3: The classic Kriging metamodel and one monotonicity-preserving bootstrapped Kriging metamodel, and true I/O function for M/M/1 with $n = 5$ traffic rates and $m = 5$ replicates

combination $\mathbf{x}_{n+1}$, this gives the A predictions $y^*_{n+1;a}$.These $y^*_{n+1;a}$ give as the point estimate the sample median $y^*_{n+1;(\lceil 0.50A \rceil)}$. To obtain a (say) 90% CI, the A accepted predictions $y^*_{n+1;a}$ are sorted, which gives the order statistics $y^*_{(n+1;a)}$ (order statistics are usually denoted by subscripts in parentheses); these order statistics give the lower and upper bounds of the 90% CI; namely, $y^*_{n+1;(\lfloor 0.05A \rfloor)}$ and $y^*_{n+1;(\lceil 0.95A \rceil)}$ If this interval turns out to be too wide, then A is increased by increasing the bootstrap sample size B. CIs in the *classic* Kriging literature assume normality of the simulation output and use the variance estimator for the Kriging predictor that ignores the random character of the Kriging parameters; see Section 3. An additional advantage of monotonic Kriging is that its CIs obviously exclude negative values if negative values are not observed when running the simulation model. For the mean and the 90% quantile and $n = 10$ simulated traffic rates, the coverages turn out to be close to the nominal 90% for monotonic Kriging, whereas classic Kriging gives coverages far below the desired nominal value; for $n = 5$ the coverages of bootstrapped Kriging are still better than classic Kriging,but lower than the required 90%.

7.2 Convexity

A procedure to obtain convexity-improving bootstrapped Kriging metamodels is derived in [28]. This procedure follows the "monotonicity" procedure given in the preceding subsection, adjusting Step 4 including (17) as follows. A convex function has a positive semi-definite (PSD) Hessian, which is the square matrix of second-order partial derivatives of the I/O function. To estimate the Hessian, the DACE software must be augmented with some extra programming; an example of the formulas for a Hessian is (6) in [28] for an (s, S) inventory simulation. To illustrate and evaluate this procedure, [28] use the following examples:

1. An (s, S) inventory simulation with as output the mean cost, and as inputs the reorder quantity s and the order-up-to quantity S. Unfortunately, it seems impossible to prove that this simulation has a convex I/O function.

2. A second-order polynomial with coefficients suggested by the (s, S) simulation such that this polynomial is certainly convex.

3. The same polynomial, but now augmented with Gaussian noise with zero mean and specific standard deviations that vary with the input combinations.

4. A newsvendor problem for which [46] proves that it has a convex I/O function. The newsvendor must decide on the order quantity x

to satisfy random demand D. Furthermore, this example assumes a uniform distribution for D. Actually, this distribution gives an I/O function that is a second-order polynomial in the input x.

5. Another variant of the newsvendor problem which assumes a Gaussian distribution for D.

Notice that the (s, S) model is a discrete-event simulation, but the newsvendor model is not; yet, the latter is still a random simulation.

These five examples do not give truly convex classic or bootstrapped Kriging metamodels. Therefore [28] accepts only those bootstrapped Kriging metamodels that have at least as many PSD estimated Hessians at the old plus new points, as the classic Kriging metamodel has. We emphasize that bootstrapped "convexity improving" Kriging does give a CI for the optimal input combination and the corresponding output.

We discuss only the two newsvendor examples (numbered 4 and 5 in the preceding list). For the input, $n = 10$ values are selected, because of the rule-of-thumb in [34]. To select the specific n values, LHS is used. The number of replicates m_i is selected such that with 90% certainty the average simulation output per input value is within 10% of the true output. Since there is a single input—namely, the order quantity x—it is not necessary to estimate the Hessians, but it suffices to check that the first-order derivatives increase—from a negative value to a positive value. The derivatives are checked at 10 old points and at 100 equally spaced points. This check shows lack of convexity in roughly half of the old and new points. Figure 4 implies $10 \leq m_i \leq 110$ replicates per point, and displays both the sample means and the sample ranges of these replicates. A visual check suggests that the averages do not show convexity, but these averages are random—as the individual replicates illustrate.

We expect that the accepted Kriging metamodels improve simulation-optimization. There are many simulation–optimization methods, but [28] applies a simple *grid search*; i.e., in the area of interest the Kriging predictor is computed at a grid and the combination that gives the minimum predicted output is selected. So, the A accepted Kriging metamodels give the estimated optimum outputs (say) $y^*_{a;opt}$ with $a = 1, ..., A$. The resulting order statistics $y^*_{(a);opt}$ give both a CI and the median point estimate. The same grid search may also be applied to the classic Kriging metamodel. Bootstrapped Kriging with its A accepted metamodels also gives the estimated optimum input combinations. Sorting these estimates for the optimal input gives a CI and a median. Furthermore, there is an estimated optimal input for the classic Kriging metamodel. The conclusion in [28] is that bootstrapping helps find better solutions

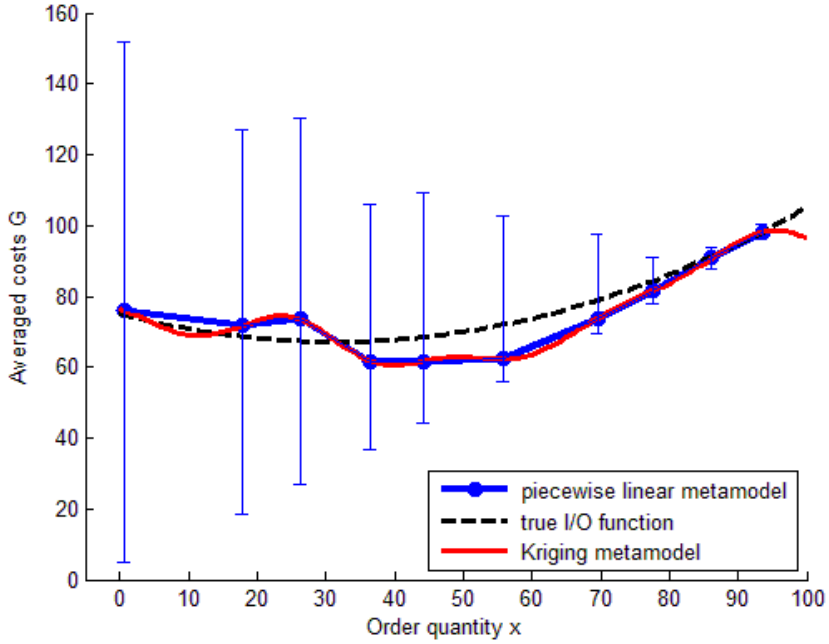

Figure 4: Cost for newsvendor model with Gaussian demand; true I/O function, range of replicated costs, average costs, fitted Kriging metamodel, and piecewise linear metamodel

than classic Kriging suggests; the CIs for the optimal inputs help select an experimental area for the simulation experiments in the next stage. We point out that classic Kriging does not provide a CI for the estimated optimal input combination; it does provide a naive CI for the estimated output that corresponds with this input combination.

8 Conclusions

In this paper we surveyed simulation-optimization via Kriging metamodels of either deterministic or random simulation models. These metamodels may be analyzed through bootstrapping. The various sections demonstrated that the bootstrap is a versatile method, but it must be tailored to the specific problem being analyzed. Distribution-free bootstrapping applies to random simulation models, which are run several times for the same scenario. Deterministic simulation models, however, are run only once for the same scenario, so parametric bootstrapping applies assuming a Gaussian process (multivariate Gaussian distribution) with parameters estimated from the simulation I/O data.

More specifically, we surveyed the following topics:

- EGO in deterministic simulation, using Kriging: We add parametric bootstrapping to obtain a better estimator of the Kriging predictor's variance that accounts for the randomness resulting from estimating the Kriging parameters.
- Constrained optimization in random simulation; We add distribution-free bootstrapping for the validation of the Kriging metamodels that are combined with mathematical programming.
- Robust optimization accounting for an uncertain environment: We combine Kriging metamodels and mathematical programming, which results in a robust solution; the effects of the randomness in the Kriging metamodels are analyzed through distribution-free bootstrapping.
- Bootstrapped Kriging either improving the convexity or preserving the monotonicity of the metamodel when the simulation model is assumed to have a convex or monotonic I/O function.

Acknowledgement 1 *This paper is based on a seminar I presented at the conference "Stochastic and noisy simulators" organized by the French Research Group on Stochastic Analysis Methods for COdes and NUMerical treatments called "GDR MASCOT-NUM" in Paris on 17 May 2011. I also presented a summary of this paper at the workshop*

"Accelerating industrial productivity via deterministic computer experiments and stochastic simulation experiments" organized at the Isaac Newton Institute for Mathematical Sciences in Cambridge on 5 - 9 September 2011. And I presented a summary at the "First Workshop On Applied Meta-Modeling", Cologne University of Applied Sciences, Gummersbach, Germany, 16 November 2012. I like to thank the following co-authors with whom I wrote the various articles that are summarized in this survey; namely, David Deflandre, Gabriella Dellino, Dick den Hertog, Ehsan Mehdad, Carlo Meloni, Alex Siem, Wim van Beers, Inneke van Nieuwenhuyse, and İhsan Yanıkoğlu.

References

[1] Ankenman, B., B. Nelson, and J. Staum (2010), Stochastic kriging for simulation metamodeling, *Operations Research*, 58, no. 2, pp. 371-382

[2] Baldi Antognini, A. and M. Zagoraiou (2010), Exact optimal designs for computer experiments via Kriging metamodeling, *Journal of Statistical Planning and Inference*, 140, pp. 2607-2617

[3] Bashyam, S. and M. C. Fu (1998), Optimization of (s, S) inventory systems with random lead times and a service level constraint. *Management Science*, 44, pp. 243–256

[4] Ben-Tal, A. and A. Nemirovski (2008), Selected topics in robust optimization, *Mathematical Programming*, 112, 125-158

[5] Boyd, S. and L. Vandenberghe (2004), *Convex optimization*. Cambridge University Press, Cambridge, U.K.

[6] Chen, X., B. Ankenman, and B.L. Nelson (2010), The effects of common random numbers on stochastic Kriging metamodels.Working Paper, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, Illinois

[7] Cheng, R.C.H. (2006), *Resampling methods. Handbooks in Operations Research and Management Science, Volume 13*, edited by S.G. Henderson and B.L. Nelson, pp. 415–453

[8] Cressie, N.A.C. (1993), *Statistics for spatial data: revised edition*. Wiley, New York

[9] Dancik, G.M. and K.S. Dorman (2008), mlegp: statistical analysis for computer models of biological systems using R. *Bioinformatics*, 24, no. 17, pp. 1966–1967

[10] Dehlendorff, C., M. Kulahci, and K. Andersen (2011), Designing simulation experiments with controllable and uncontrollable factors for applications in health care. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 60, pp. 31-49

[11] Dellino, G., J.P.C. Kleijnen, C. Meloni. (2010), Robust optimization in simulation: Taguchi and Response Surface Methodology. *International Journal of Production Economics*, 125, pp. 52-59

[12] Dellino, G. J.P.C. Kleijnen, and C. Meloni (2012), Robust optimization in simulation: Taguchi and Krige combined. *INFORMS Journal on Computing*, 24, no. 3, pp. 471–484

[13] Den Hertog, D., J.P.C. Kleijnen, and A.Y.D. Siem (2006), The correct Kriging variance estimated by bootstrapping. *Journal Operational Research Society*, 57, pp. 400–409

[14] Efron, B. and R.J. Tibshirani (1993), *An introduction to the bootstrap*. Chapman & Hall, New York

[15] Forrester, A., A. Sóbester, and A. Keane (2008), *Engineering design via surrogate modelling: a practical guide*. Wiley, Chichester, United Kingdom

[16] Frazier, P.I. (2011), Learning with Dynamic Programming. In: *Wiley Encyclopedia of Operations Research and Management Science*, Cochran, J.J. , Cox, L.A., Keskinocak, P., Kharoufeh, J.P., Smith, J.C. (eds.), Wiley, New York

[17] Frazier, P., W. Powell, and S. Dayanik (2009), The knowledge-gradient policy for correlated normal beliefs. *INFORMS Journal on Computing*, 21, pp. 599-613

[18] Fu, M.C. (2007), Are we there yet? The marriage between simulation & optimization. *OR/MS Today*, 34, pp.16–17

[19] Han, G. and T.J. Santner (2008), MATLAB parametric empirical Kriging (MPErK) user's guide. Department of Statistics, The Ohio State University, Columbus, OH 43210-1247

[20] Janusevskis, J. and R. Le Riche. (2010), Simultaneous kriging-based sampling for optimization and uncertainty propagation. HAL report: hal-00506957 (http://hal.archives-ouvertes.fr/hal-00506957_v1/)

[21] Jones, D.R., M. Schonlau, and W.J. Welch (1998), Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13, pp. 455-492

[22] Joseph, V. R., Y. Hung, and A. Sudjianto (2008), Blind Kriging: a new method for developing metamodels. *Journal of Mechanical Design*, 130, no. 3, pp. 031102-1 - 031102-8

[23] Kelton, W.D., R.P. Sadowski, D.T. Sturrock (2007), *Simulation with Arena; 4th edition*. McGraw-Hill, Boston

[24] Kleijnen, J.P.C. (2008), *Design and analysis of simulation experiments.* Springer

[25] Kleijnen, J.P.C. (2009), Kriging metamodeling in simulation: a review. *European Journal of Operational Research*, 192, no. 3, pp.

707-716

[26] Kleijnen, J.P.C. (2013), Discussion: A discrete-event simulation perspective. *Technometrics*, accepted

[27] Kleijnen, J.P.C. and D. Deflandre (2006), Validation of regression metamodels in simulation: bootstrap approach.*European Journal of Operational Research*, 170, pp. 120–131

[28] Kleijnen, J.P.C., E. Mehdad, and W. van Beers (2012), Convex and monotonic bootstrapped Kriging. *Proceedings of the 2012 Winter Simulation Conference*, edited by C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A. M. Uhrmacher

[29] Kleijnen, J.P.C. and W.C.M. van Beers (2012), Monotonicity-preserving bootstrapped Kriging metamodels for expensive simulations. *Journal of the Operational Research Society* (accepted)

[30] Kleijnen, J.P.C., W.C.M. Van Beers, and I. van Nieuwenhuyse (2010), Constrained optimization in simulation: a novel approach. *European Journal of Operational Research*, 202, pp. 164-174

[31] Kleijnen, J.P.C., W.C.M. van Beers, and I. van Nieuwenhuyse (2011), Expected improvement in efficient global optimization through bootstrapped Kriging. *Journal of Global Optimization* (accepted)

[32] Law, A.M. (2007), *Simulation modeling and analysis; fourth edition.* McGraw-Hill, Boston

[33] Lee, K.H. and G.J. Park (2006), A global robust optimization using Kriging based approximation model. *Journal of the Japanese Society of Mechanical Engineering*, 49, pp. 779-788

[34] Loeppky, J. L., Sacks, J., and Welch, W. (2009), Choosing the Sample Size of a Computer Experiment: A Practical Guide, Technometrics, 51, pp. 366-376

[35] Lophaven, S.N., H.B. Nielsen, and J. Sondergaard (2002a), DACE: a Matlab Kriging toolbox, version 2.0. IMM Technical University of Denmark, Lyngby

[36] Lophaven, S.N., H.B. Nielsen, and J. Sondergaard (2002b), Aspects of the Matlab toolbox DACE. IMM-Report 2002-13, IMM Technical University of Denmark, Lyngby

[37] Martin, J.D. and T.W. Simpson (2005), On the use of Kriging models to approximate deterministic computer models. *AIAA Journal*, 43, no. 4, pp. 853-863

[38] Nair, V.N., editor.(1992), Taguchi's parameter design: a panel discussion. *Technometrics*, 34, pp. 127-161

[39] Novikov, I. and B. Oberman (2007), Optimization of large simulations using statistical software *Computational Statistics & Data Analysis*, 51, no. 5, pp. 2747-2752

[40] Picheny, V., D. Ginsbourger, Y. Richet, and G. Caplin (2013), Quantile-based optimization of noisy computer experiments with tunable precision. *Technometrics*, accepted

[41] Rasmussen, C.E. and C. Williams (2006), *Gaussian processes for machine learning*, The MIT Press, Cambridge, Massachusetts

[42] Regis, R.G. (2011), Stochastic radial basis function algorithms for large-scale optimization involving expensive black-box objective and constraint functions. *Computers & Operations Research*, 38, pp. 837-853

[43] Roustant, O., D. Ginsbourger, and Y. Deville (2011), DiceKriging: Kriging methods for computer experiments. R package version 1.3.2

[44] Sacks, J., W.J. Welch, T.J. Mitchell and H.P. Wynn (1989), Design and analysis of computer experiments (includes Comments and Rejoinder). *Statistical Science*, 4, no. 4, pp. 409-435

[45] Santner, T.J., B.J. Williams, and W.I. Notz (2003), *The design and analysis of computer experiments*. Springer-Verlag, New York

[46] Shapiro, A. and A. Philpott (2012), A tutorial on stochastic programming. Downloaded from http://stoprog.org/index.html?SPTutorial/SPTutorial.html

[47] Stein, M.L. (1999), *Statistical interpolation of spatial data: some theory for Kriging,* Springer

[48] Taguchi, G. (1987), *System of experimental designs, volumes 1 and 2.* UNIPUB/ Krauss International, White Plains, New York

[49] Villemonteix, J., E. Vazquez, and E. Walter (2009), An informational approach to the global optimization of expensive-to-evaluate functions. Journal of Global Optimization,44, no. 4, pp. 509-534

[50] Yanıkoğlu, İ., D. den Hertog, and J.P.C. Kleijnen (2012), Adjustable robust parameter design with unknown distributions. Working Paper

[51] Yin, J., S.H. Ng, and K.M. Ng (2009), A study on the effects of parameter estimation on Kriging model's prediction error in stochastic simulations. *Proceedings of the 2009 Winter Simulation Conference*, edited by M.D. Rossini, R.R. Hill, B. Johansson, A. Dunkin, and R.G. Ingalls, pp. 674-685