Simulation Optimization through Regression or Kriging Metamodels

Kleijnen, Jack

Document version:
Early version, also known as pre-print

Publication date:
2017

Link to publication

Citation for published version (APA):
No. 2017-026

SIMULATION OPTIMIZATION THROUGH REGRESSION OR KRIGING METAMODELS

By

Jack P.C. Kleijnen

16 May, 2017

ISSN 0924-7815
ISSN 2213-9532
Simulation Optimization through Regression or Kriging Metamodels

Jack P.C. Kleijnen
Tilburg University, Postbox 90153, Tilburg, Netherlands
kleijnen@tilburguniversity.edu

May 16, 2017

Abstract
This chapter surveys two methods for the optimization of real-world systems that are modelled through simulation. These methods use either linear regression metamodels, or Kriging (Gaussian processes). The metamodel type guides the design of the experiment; this design fixes the input combinations of the simulation model. These regression models use a sequence of local first-order and second-order polynomials—known as response surface methodology (RSM). Kriging models are global, but are re-estimated through sequential designs. "Robust" optimization may use RSM or Kriging, and accounts for uncertainty in simulation inputs.

Keywords: Cross-validation, Robust optimization, Regression analysis, Kriging, Gaussian process, Response surface methodology (RSM), Efficient global optimization (EGO), Taguchi, Bootstrap, Common random numbers (CRN), Latin hypercube sampling (LHS), Karush-Kuhn-Tucker (KKT).

JEL: C0, C1, C9, C15, C44

1 Introduction
In this chapter we survey two methods for simulation optimization (SO) where SO means optimization of real-world systems modelled through simulation; this SO may have a single objective or multiple objectives. Examples are the optimization of the number of checkout lanes at a specific supermarket, and the optimization of the order quantities for the many inventory items in such a supermarket. We may also apply SO to calibrate a simulation model; i.e., to estimate the optimal parameter values of the simulation model. For example, Liu et al. (2017) calibrates an agent-based simulation (ABS) model of an emergency department while data are scarce.

Actually, there are many SO methods, as this book illustrates. Some SO methods use metamodels—also called surrogates or emulators—which we define
as explicit and relatively simple approximations of the input/output (I/O) functions that are implicitly defined by the given simulation models. Metamodels treat these simulation models as black boxes; i.e., only the I/O (not the internal variables) of the simulation model is observed. There are many types of metamodels; e.g., artificial neural networks (ANNs), radial basis functions (RBFs), and splines; see Bartz-Beielstein and Zaefferer (2017) and Kleijnen (2015, p. 10). Moreover, these metamodel types may be combined into a so-called ensemble; see Bartz-Beielstein and Zaefferer (2017), Friese et al. (2016), and Kleijnen (2015, p.11). We, however, focus on one of the following two types: (i) first-order and second-order polynomials—which are linear regression models—that are applied in response surface methodology (RSM), and (ii) Kriging or Gaussian process (GP) models. Both types can be extended from single-objective (univariate, scalar) to multi-objective (multivariate, vector) SO. We provide a state-of-the-art survey of SO using one of these two types of metamodels. As we stated above, no metamodels are used by some other SO methods; e.g., evolutionary methods, artificial ant colonies, and simulated annealing. The advantage of using metamodels is that they result in more efficient SO methods. This efficiency is important if the simulation is expensive; i.e., the simulation requires much computer time to obtain the value of the simulation objective(s) for a specific combination of simulation inputs or scenario. A first example is the deterministic simulation of a car-crash model at Ford that required 36 to 160 hours of computer time; see Simpson et al. (2004). A second example is a random simulation of a waiting-line system that requires very many "customers" if we want an accurate estimate of the steady-state mean waiting-time for a high traffic rate. A third example is a random "rare event" simulation aimed at estimating a small probability (such as the probability of a nuclear accident), so we need an extremely long simulation run (unless we successfully apply importance sampling). Moreover, even if the simulation is cheap (computationally inexpensive), the number of input combinations may be so big that it becomes expensive or impossible to simulate all possible combinations; actually, if one or more inputs are continuous variables, then there are infinitely many combinations. Altogether, metamodels are efficient and effective, provided they are "adequate" approximations. An additional advantage of regression and Kriging metamodels is that they can quantify the uncertainty (measured through the variance) of their predictors, as we shall see in the next sections.

Note: Regression models have been developed and applied since the 1800s: Gauss used least-squares (LS) regression. Kriging originated in geostatistics or spatial data, and was named after the South-African mining engineer Krige (born 1919); see Cressie (1993). Kriging has also become popular in machine learning; see Rasmussen and Williams (2006). Kriging in deterministic simulation or computer experiments started with Sacks et al. (1989). Kriging in discrete-event simulation got established in Ankenman et al. (2010). Recent references and software for Kriging in both types of simulation are presented in Kleijnen (2015, pp. 179–239). Jalali and Van Nieuwenhuyse (2015) claims that metamodel-based optimization is "relatively common" and that RSM is the most popular metamodel-based method, while Kriging is popular in theo-
oretical publications. We note that most simulation models have many inputs, which leads to the curse of dimensionality; in such situations we should apply so-called factor screening before optimization; see Kleijnen (2015, pp. 135–178) and Kleijnen (2017). Finally, we note that Pontes et al. (2016) optimizes ANNs, applying full factorial designs and evolutionary operations (EVOP).

As the preceding text illustrates, in this chapter we discuss both deterministic simulation and random (or stochastic) simulation. There are various types of random simulation: discrete-event dynamic systems, agent-based simulation, and stochastic differential equations. Mathematically, these simulations give random outputs (responses) because these outputs are transformations of pseudorandom numbers (PRNs); these PRNs are produced by a PRN generator that uses an initial PRN or seed, which is a special type of simulation input.

SO requires experimentation with the simulation model; i.e., we simulate different "values" or "levels" for the simulation model's parameters, input values, and starting values of the inputs—we use the terminology in Zeigler et al. (2000); i.e., we must infer the value of a parameter, whereas we can directly observe the value of an input. For example, in a queueing or waiting-line simulation for a supermarket we may start with an "empty" (no waiting customers) simulated system, exponential interarrival times with a fixed arrival rate, and a fixed number of servers with a given "helping" rate and cost per server; our goal is to estimate the optimal number of servers. The statistical theory on the design of experiments (DOE) speaks of factors, which have fixed values during one "run" of the simulation experiment; by definition, DOE determines the input combinations of the experiment. When we experiment, we use a metamodel. Actually, simulation analysts may not realize that they are using a metamodel; e.g., if they change only one factor at a time, then they implicitly assume that the factors do not interact. As we shall see in this chapter, a metamodel is used to analyze the simulation I/O data; this metamodel also determines the type of design for the simulation experiment.

Note: Classic or "traditional" DOE and design and analysis of simulation experiments (DASE) have the following important differences. DOE was developed for real-world experiments in agriculture, engineering, psychology, etc. In these experiments it is impractical to investigate "many" factors; i.e., ten factors seems a maximum. Moreover, it is hard to investigate more than "a few" levels per factor; five levels seems the limit. Simulation models, however, may have thousands of factors—each with many values. Consequently, a multitude of factor combinations may be simulated (also see our comments on "expensive" and "cheap" simulations). Moreover, simulation is well-suited to "sequential" designs instead of "one shot" designs, because simulation experiments run on computers that typically produce output sequentially (apart from parallel computers; see this book and Gramacy (2015)), whereas agricultural experiments run during a single growing season. Altogether, many simulation analysts need a change of mindset. We also refer to Sanchez et al. (2012).

In robust optimization (RO) we may try to estimate the optimal combination of the decision variables of the simulation model while accounting for uncertainty in the parameters of that model. In this chapter we shall present RO approaches
based on either Taguchi or mathematical programming (MP).

Note: Taguchi has been popular in mechanical engineering, since several decades. In MP, RO is a recent important topic. Taguchi inspired RO through RSM in Dellino et al. (2010), while Taguchi inspired RO through Kriging in Dellino et al. (2012). An example of RO in MP is Bertsimas and Mišić (2017). An example of RO combining Taguchi and MP is Yanikoglu et al. (2017).

We base this survey on Kleijnen (2017), which is a tutorial based on Kleijnen (2015). Actually, Kleijnen (2017) surveys more topics, besides SO; in this chapter we also survey deterministic simulation, besides random simulation. Kleijnen (2015) includes hundreds of additional references, and many website addresses for software.

We organize this chapter as follows. Section 2 summarizes classic linear regression and DOE. Section 3 presents solutions for DASE if the classic statistical assumptions are violated in practice. Section 4 presents RSM. Section 5 summarizes Kriging and its designs. Section 6 presents SO using Kriging. Section 7 extends RSM and Kriging to RO. We try to use mathematical symbols consistently, but sometimes we ran out of symbols so the context should eliminate confusion; e.g., $r$ has multiple meanings.

2 Classic linear regression and designs

Because we assume that the readers are familiar with classic linear regression, we limit our discussion of this regression to definitions of our mathematical symbols and terminology. We define $w = f_{\text{sim}}(z, r)$ where $w$ denotes the scalar simulation output (e.g., average waiting time), $f_{\text{sim}}$ the implicit I/O function of the simulation model, $z$ the $k$-dimensional vector with the values of the $k$ simulation inputs, and $r$ the seed of the PRNs used by the random simulation model; in deterministic simulation, this special input $r$ vanishes. Usually (but not in RSM), $z$ is standardized (scaled) such that the resulting input $d$ has elements either $-1 \leq d_j \leq 1$ or $0 \leq d_j \leq 1$ with $j = 1, ..., k$. We approximate $w = f_{\text{sim}}(z, r)$ by $y = f_{\text{meta}}(x) + e$ where $y$ is the metamodel output, which may deviate from $w$ so the approximation error $e$ may have $\mu_e \neq 0$; if $\mu_e = 0$, then we call $f_{\text{meta}}$ adequate or valid.

The simplest optimization problem has no input constraints besides simple box constraints such as $0 \leq d_j \leq 1$; it has continuous inputs, which have no uncertainty. Moreover, it concerns the expected value of a single output, $E(w)$; this $E(w)$ may represent the probability of a binary variable such as failure or success, but excludes quantiles and the mode of the output distribution.

2.1 Classic linear regression

We define the linear regression (meta)model $y = X_N \beta + e$ where $y$ denotes the $N$-dimensional vector with the observations on the dependent (explained) variable, $N = \sum_{i=1}^{n} m_i$ with $n$ the number of simulated input combinations, $m_i$ the number of replications for combination $i$ (obviously, deterministic simulation
implies $m_i = 1$ so $N = n$), $X_N$ is the $N \times q$ matrix of independent (explanatory) regression variables—obviously, $X_N$ has $m_i$ identical rows, whereas $X_n$ denotes the corresponding $n \times q$ matrix without any identical rows determined by the $n \times k$ design matrix $D$ and the type of regression model (e.g., second-order polynomial), $\beta$ denotes the $q$-dimensional vector with regression parameters (coefficients), and $e$ denotes the $N$-dimensional vector with the residuals $E(y) - E(w)$ where $w$ denotes the $N$-dimensional vector with independent simulation outputs; this independence requires that random simulation does not use common random numbers (CRN). We also define the $q$-dimensional row vector $x_i = (x_{i1}, ..., x_{iq})$.

We focus on a special type of linear regression; namely, a second-order polynomial with $k$ simulation inputs, which has an intercept $\beta_0$, $k$ first-order effects $\beta_j (j = 1, ..., k)$, $k(k-1)/2$ two-factor interactions (cross-products) $\beta_{jj'} (j < j')$, and $k$ purely quadratic effects $\beta_{jj}$. These interactions mean that the effect of an input depends on the values of one or more other inputs. A purely quadratic effect means that the marginal effect of the input is not constant, but either diminishes or increases. This polynomial is nonlinear in $x$, and linear in $\beta$, so this polynomial is a linear regression model. We assume that interactions among three or more inputs are unimportant, because such interactions are hard to interpret and in practice are often unimportant indeed. Of course, we should check this assumption; i.e., we should "validate" the estimated metamodel (see Section 3.3).

The ordinary least squares (OLS) estimator of $\beta$ is $\hat{\beta} = (X'_N X_N)^{-1}X'_N w$, assuming the inverse of $X'_N X_N$ exists; e.g., $\hat{\beta}$ exists if $X_N$ is orthogonal. If $m_i$ is a positive integer constant (say) $m$, then we may replace $w$ by $\tilde{w}$ with the $n$ elements $\tilde{w}_i = \sum_{r=1}^m w_{ir}/m$ and replace $X_N$ by $X_n$. Moreover, $\hat{\beta}$ is the maximum likelihood estimator (MLE) if $e$ is white noise, so $e$ is normally, independently, and identically distributed (NIID) with zero mean and constant variance $\sigma^2_e$; i.e., $e \sim N_N(0_N, \sigma^2_e I_{N \times N})$ where $N_N$ stands for $N$-variate (see subscript) normally (symbol: $N$) distributed, $0_N$ for the $N$-dimensional vector with zeroes, and $I_{N \times N}$ for the $N \times N$ identity matrix. If the metamodel is valid, then $\sigma^2_e = \sigma^2_w$. White noise implies that $\hat{\beta}$ has the $q \times q$ covariance matrix $\Sigma_\beta = (X'_N X_N)^{-1} \sigma^2_w$. Because $\sigma^2_e$ is unknown, we estimate $\sigma^2_\epsilon = \sigma^2_w$ through the mean squared residuals (MSR) $(\tilde{y} - \tilde{w})(\tilde{y} - \tilde{w})/(N - q)$ with predictor $\tilde{y} = X_N \hat{\beta}'$ and degrees of freedom (DOF) $N - q > 0$; this inequality is satisfied, even if $n = q$ but $m_i > 1$ for at least one value of $i$. This MSR gives the estimator $\hat{\Sigma}_\beta$. This $\hat{\Sigma}_\beta$ has a main diagonal with the elements $s^2(\tilde{\beta}_g) (g = 1, ..., q)$, which give the square roots $s(\tilde{\beta}_g)$. Confidence intervals (CIs) and tests for the individual $\tilde{\beta}_g$ follow from the Student $t$-statistic with $N - q$ DOF: $t_{N-q} = (\tilde{\beta}_g - \beta_g)/s(\tilde{\beta}_g)$. Finally, $\tilde{y} = X_N \hat{\beta}'$ implies $s^2(\tilde{y}|x_i) = x'_i \hat{\Sigma}_\beta x_i$. This $s^2(\tilde{y}|x_i)$ is minimal at the center of the experimental area.

Given the equality $\Sigma_\beta = (X'_N X_N)^{-1} \sigma^2_w$, we may select $X_N$ such that we "optimize" $\Sigma_\beta$. Obviously, $X_N$ is determined by $X_n$, $m_i$, and the type of regression model (e.g., $X_N$ may include $x^2_j$). DOE does not say much about
the selection of $m_i$; typically, DOE assumes $m_i = 1$. If $m_i = m \geq 1$, then an orthogonal $X_n$ implies an orthogonal $X_N$ (we may specify $X_N$ as $X_n$ "stapled" or "stacked" $m$ times). We shall further discuss $m_i$ in Section 3.3. In the next subsections we shall discuss first-order and second-order polynomials. The order of these polynomials require designs of a specific resolution (abbreviated to R). Obviously, $X_n$ is determined by $D$. To select a specific $D$ with $z_{i,j}$ standardized such that $-1 \leq d_{i,j} \leq 1$, we try to minimize $\text{Var}(\beta_g)$ ($g = 1, \ldots, q$); other criteria are discussed in Kleijnen (2015, pp. 66–67). We can prove that this minimization requires an orthogonal $X_N$, which gives $\Sigma_\beta = (NI)^{-1} \sigma_w^2 = I\sigma_w^2/N$. Because this $\Sigma_\beta$ is diagonal, the $\beta_j$ are statistically independent. Moreover, these $\beta_j$ have the same variance; namely, $\sigma_w^2/N$. So we can rank the explanatory variables in order of importance, using either $\beta_g$ or $t_{N-q}$ with $\beta_g = 0$ (so $t_{N-q} = \beta_g/s(\beta_g)$); usually, we do not hypothesize that the intercept $\beta_0$ is zero.

Note: If all $\beta_g$ are independent, then the full regression model with $q$ effects and the reduced model with nonsignificant effects eliminated have identical values for those estimated effects that occur in both models. If not all $\beta_g$ are independent, then so-called backwards elimination of nonsignificant effects changes the values of the remaining estimates.

### 2.2 R-III designs for first-order polynomials

If a first-order polynomial is an adequate metamodel, then changing one factor at a time gives unbiased $\beta_j$ ($j = 1, \ldots, k$). However, these $\beta_j$ do not have minimum variances; a R-III design does minimize these variances because this design gives an orthogonal $X_N$, as we shall see in this subsection. Furthermore, we can prove that $\text{Var}(\beta_j)$ is minimal if we simulate only two levels per factor, as far apart as the experimental area allows (in either a one-factor-at-a-time or a R-III design); see Kleijnen (2015, pp. 44–49).

R-III designs are also known as Plackett-Burman (PB) designs. A subclass are fractional-factorial two-level R-III designs, denoted by $2^{k-p}_{III}$ with integer $p$ such that $0 \leq p < k$ and $2^{k-p} \geq 1 + k$: we first discuss these $2^{k-p}_{III}$ designs. Any $2^{k-p}$ design is balanced; i.e., each input is simulated $n/2$ times at its lower value (say) $L_j$ and at its higher value $H_j$. Furthermore, such a design gives an orthogonal $X_n$. This design may be saturated: $N = q$ (with $N = \sum_{i=1}^{n} m_i$, $n = 2^{k-p}$, and $q = 1 + k$). A saturated design implies that the MSR is undefined (because $N - q = 0$). To solve this problem, we may obtain replications for one or more combinations of this design; e.g., the combination at the center of the experiment where $d_j = 0$ if $d_j$ is quantitative and $d_j$ is randomly selected as $-1$ or $1$ if $d_j$ is qualitative with two levels. A simple algorithm for constructing $2^{k-p}_{III}$ designs is given in Kleijnen (2015, pp. 53–54).

Whereas $2^{k-p}_{III}$ designs have $n$ equal to a power of 2, there are also PB designs with $n$ a multiple of 4; e.g., $8 \leq k \leq 11$ implies $n = 12$ (whereas $2^{12-8}_{III}$ implies $n = 16$). If $8 \leq k < 11$, then we do not use $n-(k+1)$ columns of $D$. Actually, there
are PB designs for $12 \leq n \leq 96$; for $12 \leq n \leq 36$ these designs are tabulated in Montgomery (2009, p. 326) and Myers et al. (2009, pp. 165). Like $2^k\!-\!p$ designs, these PB designs are balanced and they give an orthogonal $X_n$.

### 2.3 R-V designs for two-factor interactions

A R-V design enables unbiased $\hat{\beta}_j$ and $\hat{\beta}_{j,j'}$ with $j < j'$. Obviously, $q$ equals $1 + k + k(k - 1)/2$. The DOE literature gives tables for generating $2^k\!-\!p$ designs. Unfortunately, these designs are not saturated at all; e.g., the $2_8^\!-2$ design implies $n = 64 \gg q = 37$. Rechtschaffen designs, however, do include saturated R-V designs; see Kleijnen (2015, pp. 62–63). We shall use R-V designs in the next subsection.

### 2.4 CCDs for second-order polynomials

A CCD or central composite design enables unbiased $\hat{\beta}_j$ and $\hat{\beta}_{j,j'}$ with $j \leq j'$. A CCD consists of three subdesigns: (i) a R-V design; (ii) the central combination $0_k'$; (iii) the $2k$ axial combinations—which form a star design—with $d_j = c$ and $d_{j'} = 0$ where $j' \neq j$, and $d_j = -c$ and $d_{j'} = 0$. Obviously, $c \neq 1$ implies five values per input, whereas $c = 1$ implies three values per input. The usual choice of $c$ is not 1. The "optimal" choice of $c$ assumes white noise, which does not hold in practice so we do not detail this choice. Finally, if $c \leq 1$, then $-1 \leq d_{i,j} \leq 1$; else $-c \leq d_{i,j} \leq c$.

A CCD gives a non-orthogonal $X_n$; e.g., any two columns corresponding with $\beta_0$, $\beta_{j,j}$, and $\beta_{j,j'}$ are not orthogonal. A CCD is rather inefficient (i.e., $n \gg q$); yet, CCDs are popular in DOE, especially in RSM (see Section 1). For further discussion of CCDs and other types of designs for second-degree polynomials we refer to Kleijnen (2015, pp. 64–66), and Myers et al. (2009, pp. 296–317). A more efficient modified CCD is derived in Kleijnen and Shi (2017).

### 3 Classic assumptions vs. simulation practice

The classic assumptions stipulate a single type of output (univariate output) and white noise; see Section 2. In simulation practice, however, the simulation model often has a multivariate output and no white noise—as we discuss now.

#### 3.1 Multivariate simulation output

We assume that for $v$-variate simulation output with $v \geq 1$ we use $v$ univariate polynomials of the same order (e.g., second-order), so $\mathbf{y}^{(l)} = \mathbf{X}_N \beta^{(l)} + \mathbf{e}^{(l)}$ with $l = 1, \ldots, v$ where $\mathbf{y}^{(l)}$ corresponds with output type $l$; $\mathbf{X}_N$ is the $N \times q$ matrix for metamodel $l$; $\beta^{(l)}$ is the vector with the $q$ regression parameters for metamodel $l$; and $\mathbf{e}^{(l)}$ is the $N$-dimensional vector with the residuals of metamodel $l$. Obviously, $\mathbf{e}^{(l)}$ has variances that may vary with $l$ (e.g., the variances differ for simulated inventory costs and service percentages), and $e_i^{(l)}$ and $e_i^{(l')}$ are not
independent (they are different transformations of the same PRNs). Nevertheless, it can be proven that the *best linear unbiased estimator* (BLUE) of $\beta^{(l)}$ is the OLS estimator computed per output: $\hat{\beta}^{(l)} = (X'X)^{-1}X'w^{(l)}$. Furthermore, CIs and tests for the elements in $\hat{\beta}^{(l)}$ use the classic formulas in the preceding section. We are not aware of any general designs for multivariate output. For further discussion of multivariate output we refer to Kleijnen (2015, pp. 85–88).

### 3.2 Nonnormal simulation output

The normality assumption often holds *asymptotically*; i.e., if the simulation run is long, then the sample average of the autocorrelated observations is nearly normal. Estimated quantiles, however, may be very nonnormal, especially in case of an "extreme" (e.g., 99%) quantile. The $t$-statistic (used in the CIs) is quite insensitive to nonnormality. Whether the actual simulation run is long enough to make the normality assumption hold, is always hard to know. Therefore it seems good practice to test whether the simulation output has a Gaussian probability density function (PDF). For these tests we may use various residual plots and *goodness-of-fit statistics*; e.g., the chi-square statistic. These tests assume that the outputs are IID. We may therefore obtain "many" (say, 100) replications for a specific input combination (e.g., the base scenario). However, if the simulation is expensive, then these plots are too rough and these tests have no power.

Obviously, deterministic simulation does not give a normally distributed $w$. Actually, the white-noise assumption concerns $e$ in the metamodel, not $w$ in the deterministic or random simulation model. Given $m_i \geq 1$ replications ($i = 1, \ldots, n$), we obtain $\bar{w}_i = \sum_{r=1}^{m_i} w_{i,r}/m_i$ and the corresponding $\bar{y}_i = \bar{y}_i - \bar{w}_i$. For simplicity of presentation, we assume that $m_i$ is a constant $m$. If $w_{i,r}$ has a constant variance $\sigma^2_w$, then $\bar{w}_i$ also has a constant variance; namely, $\sigma^2_{\bar{w}} = \sigma^2_w / m$. Unfortunately, even if $\bar{w}_i$ has a constant variance $\sigma^2_{\bar{w}}$ and is independent of $\bar{w}_{i'}$ with $i \neq i'$ (no CRN), then $\Sigma_{\bar{w}} = [I - X(X'X)^{-1}X']\sigma^2_{\bar{w}}$ so $\bar{w}$ does not have IID components; so, the interpretation of the popular plot with estimated residuals is not straightforward.

We may apply *normalizing transformations*; e.g., $\log(w)$ may be more normally distributed than $w$. Unfortunately, the metamodel now explains the behavior of the transformed output—not the original output; also see Kleijnen (2015, p. 93).

A statistical method that allows nonnormal random simulation output $w$ is *distribution-free bootstrapping* or *nonparametric bootstrapping*. We denote the original observations by $w$, and the bootstrapped observations by $w^*$. We assume that these $w$ are IID; indeed, $w_{i,1}, \ldots, w_{i,m_i}$ are IID because the $m_i$ replications use nonoverlapping PRN streams. We *resample—with replacement*—these $m_i$ observations such that the original sample size $m_i$ remains unchanged; obviously, $m_i \geq 1$. We apply this resampling to each combination $i$. The resulting $w_{i,1}^*, \ldots, w_{i,m_i}^*$ give the average $\bar{w}_i^*$, which give the $n$-dimensional vector $\bar{w}^*$. For simplicity's sake, we now assume $m_i = m > 1$, so the bootstrapped OLS estimator of $\beta$
is $\hat{\beta}^* = (X'X)^{-1}X'w^*$. To reduce sampling error, we select a bootstrap sample size (say) $B$, and repeat this resampling $B$ times; e.g., $B$ is 100 or 1,000. This $B$ gives $\hat{\beta}^*_b$ with $b = 1, \ldots, B$. For simplicity’s sake, we focus on $\beta_q$ (last element of $\beta$). To compute a two-sided $(1 - \alpha)$ CI, the percentile method computes the $\alpha/2$ quantile (or percentile) of the empirical density function (EDF) of $\hat{\beta}^*_q$ obtained through sorting the $B$ observations on $\hat{\beta}^*_q$. This sorting gives the order statistics, denoted by the subscript $(\cdot)$ where—for notational simplicity—we assume that $B\alpha/2$ is integer so the estimated $\alpha/2$ quantile is $\hat{\beta}^*_{q(B[1 - \alpha/2])}$. Analogously we obtain $\hat{\beta}^*_{q(B[1 - \alpha/2])} < \beta_q < \hat{\beta}^*_{q(B[1 - \alpha/2])}$. We shall mention more bootstrap examples, in later sections.

### 3.3 Heterogeneous variances of simulation outputs

In practical random simulations, $\text{Var}(w_i)$ changes as $x_i$ changes ($i = 1, \ldots, n$). In some applications, however, we may hope that this variance heterogeneity is negligible. Unfortunately, $\text{Var}(w_i)$ is unknown so we must estimate it. The classic unbiased estimator is $s^2(w_i) = \frac{\sum_{r=1}^{m_i} (w_{ir} - \overline{w}_i)^2}{m_i - 1}$. This $s^2(w_i)$ itself has a high variance. To compare the $n$ estimators $s^2(w_i)$, we can apply many tests; see Kleijnen (2015, p. 101).

If we either assume or find variance heterogeneity, then we may still use OLS. Actually, $\hat{\beta}$ is still unbiased, but $\Sigma_{\beta}$ becomes $(X_n'X_n)^{-1}X_n'\Sigma_{w}X_n(X_n'X_n)^{-1}$ where for simplicity’s sake we assume $m_i = m$ so $\Sigma_{w}$ is the $n \times n$ diagonal matrix with the main-diagonal elements $\text{Var}(w_i)/m$.

The DOE literature ignores designs for heterogeneous output variances. We propose classic designs with $m_i$ such that we obtain approximately constant $s^2(w_i)/m_i (i = 1, \ldots, n)$. Therefore we initially take a pilot sample of size $m_0 \geq 2$ for each combination, which gives (say) $s^2_1(m_0)$. Next we select a number of additional replications $m_i - m_0$ with

$$m_i = m_0 \times \text{nint} \left[ \frac{s^2_1(m_0)}{\min \{ s^2_i(m_0) \} } \right]$$

where $\text{nint}[x]$ denotes the integer closest to $x$. Combining the $m_i$ replications of the two stages gives $\overline{w}_i$ and $s^2(w_i)$. This $\overline{w}_i$ gives $\hat{\beta}$, while $s^2(w_i)$ gives the diagonal matrix $\hat{\Sigma}_w$ with main-diagonal elements $s^2_i(\hat{m}_i)/\hat{m}_i$. This $\hat{\Sigma}_w$ gives $\hat{\Sigma}_{\beta}$, which—together with $t_{m_0-1}$—gives a CI for $\beta_j$.

Actually, (1) gives the relative number of replications $\hat{m}_i/\hat{m}_i^*$. To select absolute numbers, we recommend the rule in Law (2015, p. 505) with a relative estimation error (say) $r_{ee}$:

$$\hat{m} = \min \left[ r \geq m : \frac{t_{r-1;1-\alpha/2}\sqrt{s^2_l(m)}r}{\overline{w}(m)} \leq \frac{r_{ee}}{1 + r_{ee}} \right].$$

$$9$$
3.4 Common random numbers

Random simulation often uses CRN; actually, CRN are the default in software for discrete-event simulation. If \( m_i = m, \) then we can arrange the simulation output \( w_{ir} (i = 1, \ldots, n; r = 1, \ldots, m) \) into a matrix \( \mathbf{W} = (w_1, \ldots, w_m) \) with \( \mathbf{w}_r = (w_{1r}, \ldots, w_{nr})' \). Obviously, CRN create correlation between \( w_{ir} \) and \( w_{ir'} \).

Moreover, different replications use nonoverlapping PRN streams so \( w_{ir} \) and \( w_{ir'} \) with \( r \neq r' \) are independent. CRN are meant to reduce \( \text{Var} (\hat{\beta}_g) \) and \( \text{Var}(\hat{\gamma}) \); unfortunately, CRN increase the variance of the estimated intercept. For details on the effective usage of CRN we refer to Law (2015, pp. 592-604).

To compute \( \hat{\beta}_r \), we do not use \( \mathbf{W} \), but the vector \( \mathbf{w} \) with \( N = \sum_{i=1}^n m_i \) elements. To compute \( \hat{\Sigma}_\beta \) in case of CRN, we use the non-diagonal matrix \( \hat{\Sigma}_w \).

Unfortunately, this \( \hat{\Sigma}_w \) is singular if \( m \leq n \); if \( m > n \), then we may compute CIs for \( \beta_j \) from \( t_{m-1} \). An alternative method requires only \( m > 1 \), and computes

\[
\hat{\beta}_r = (\mathbf{X}_n' \mathbf{X}_n)^{-1} \mathbf{X}_n' \mathbf{w}_r (r = 1, \ldots, m).
\]

(3)

We again focus on a single element of this \( \hat{\beta}_r \); namely, element \( g \) (\( g = 1, \ldots, q \)). Obviously, \( \hat{\beta}_{gr} \) and \( \hat{\beta}_{gr'} \) with \( r \neq r' \) and \( r' = 1, \ldots, m \) are IID with variance \( \text{Var}(\hat{\beta}_g) \). The \( m \) replications give \( \hat{\beta}_g = \sum_{r=1}^m \hat{\beta}_{gr}/m \) and \( s^2(\hat{\beta}_g) = \sum_{r=1}^m (\hat{\beta}_{gr} - \hat{\beta}_g)^2/[m(m-1)] \); together they give \( t_{m-1} = (\hat{\beta}_g - \beta_g)/s(\hat{\beta}_g) \).

Unfortunately, we cannot apply this alternative when estimating a quantile instead of a mean. We then recommend distribution-free bootstrapping; see Kleijnen (2015, p. 99, 110). Furthermore, \( m_i \) is not a constant if we select \( m_i \) such that \( \mathbf{w}_i \) has the same—absolute or relative—width of the CI around \( \bar{w}_i \); see again (2). We must then adjust the analysis; see Kleijnen (2015, p. 112).

3.5 Validation of metamodels

We discuss various validation methods (which we may also use to compare first-order against second-order polynomials, or linear regression against Kriging metamodels). One method uses \( R^2 = \sum_{i=1}^n (\hat{y}_i - \bar{y})^2/\sum_{i=1}^n (y_i - \bar{y})^2 = 1 - \sum_{i=1}^n (\hat{y}_i - \bar{y})^2/\sum_{i=1}^n (y_i - \bar{y})^2 \) where \( \bar{y} = \sum_{i=1}^n y_i/n \) and \( m_i \geq 1 \). If \( n = q \) (saturated design), then \( R^2 = 1 \)—even if \( \text{E}(\hat{e}_i) \neq 0 \). If \( n > q \) and \( q \) increases, then \( R^2 \) increases—whatever the size of \( \text{E}(\hat{e}_i) \) is; because of possible overfitting, we may therefore use the adjusted \( R^2 \): \( R^2_{adj} = 1 - (1-R^2)(n-1)/(n-q) \). Unfortunately, we do not know critical values for \( R^2 \) or \( R^2_{adj} \). We might either use subjective lower thresholds, or estimate the distributions of these two statistics through distribution-free bootstrapping; see Kleijnen (2015, p. 114).

Actually, we prefer cross-validation over \( R^2 \) or \( R^2_{adj} \). Suppose again that \( m_i = m \geq 1 \), so we replace \( \mathbf{w} \) by \( \mathbf{w} \) in OLS. In cross-validation, we delete I/O combination \( i \) to obtain \( (\mathbf{X}_{-i}, \mathbf{w}_{-i}) \) where we suppress the subscript \( n \) of \( \mathbf{X} \). Next we compute \( \hat{\beta}_{-i} = (\mathbf{X}_{-i}' \mathbf{X}_{-i})^{-1} \mathbf{X}_{-i}' \mathbf{w}_{-i} \) (\( i = 1, \ldots, n \)). This gives \( \bar{y}_{-i} = \mathbf{x}_{i}' \hat{\beta}_{-i} \). We may "eyeball" the scatterplot with \( (\mathbf{w}_i, \bar{y}_{-i}) \), and decide
whether the metamodel is valid. Regression software may use a shortcut to avoid the \( n \) recomputations in cross-validation.

In *random* simulation we have an alternative for this scatterplot; namely, the *Studentized prediction error*

\[
t^{(i)}_{m-1} = \frac{\bar{w}_i - \tilde{y}_i}{\sqrt{s^2(\bar{w}_i) + s^2(\bar{y}_i)}}
\]

where \( s^2(\bar{w}_i) = s^2(w_i)/m \) and \( s^2(\bar{y}_i) = x_i' \Sigma_{\hat{\beta}_{-i}} x_i \) with \( \Sigma_{\hat{\beta}_{-i}} = s^2(\bar{w}_i)(X'_{-i}X_{-i})^{-1} \).

We reject the metamodel if \( \max_i |t^{(i)}_{m-1}| > t_{m-1;1-(\alpha/2n)} \) where we use the *Bonferroni inequality*; i.e., we replace \( \alpha/2 \) by \( \alpha/(2n) \). so we control the *experimentwise* or *familywise* type-I error rate \( \alpha \).

Cross-validation affects not only \( \tilde{y}_{-i} \), but also \( \hat{\beta}_{-i} \) (see above). We may be interested not only in the predictive performance of the metamodel, but also in its *explanatory* performance; i.e., do the \( n \) estimates \( \hat{\beta}_{-i} \) remain stable?

Related to cross-validation are *diagnostic* statistics; e.g., the *prediction sum of squares* (PRESS): \( \sum_{i=1}^{n} (\tilde{y}_{-i} - w_i)^2 / n \). We may apply bootstrapping to estimate the distribution of the various validation statistics; see Kleijnen (2015, p. 120).

If the validation suggests an unacceptable fitting error \( e \), then we may consider various *transformations*. For example, we may replace \( y \) and \( x_j \) by \( \log(y) \) and \( \log(x_j) \) (\( j = 1, ..., k \)) so that the first-order polynomial approximates relative changes through *elasticity coefficients*. If we assume that \( f_{\text{sim}} \) is monotonic, then we may replace \( w \) and \( x_j \) by their ranks: *rank regression*. In the preceding subsections, we also considered transformations that make \( w \) better satisfy the assumptions of normality and variance homogeneity; unfortunately, different objectives of a transformation may conflict with each other.

In Section 2 we discussed designs for low-order polynomials. If such a design does not give a valid metamodel, then we do not recommend routinely adding higher-order terms: these terms are hard to interpret. However, if the goal is better prediction, then we may add higher-order terms; e.g., a \( 2^k \) design enables the estimation of the interactions among three or more inputs. However, adding more terms may lead to overfitting; see our comment on \( R^2_{\text{adj}} \). Adding more explanatory variables is called *stepwise regression*, whereas eliminating nonsignificant variables is called *backwards elimination*, which we briefly discussed in the Note in Section 2.1.

### 4 Response surface methodology

RSM designs and analyzes a sequence of local experiments, and has gained a good track record; see Kleijnen (2015, p. 244), Law (2015, pp. 656–679), and Myers et al. (2009). We assume that before we apply RSM, we have identified the important inputs and their experimental area (RSM and screening may be combined; see Kleijnen (2015, p. 245)).
4.1 Classic RSM

The objective of RSM is to minimize \( E(w|z) \) where \( w \) denotes the simulation output and \( z \) denotes the \( k \)-dimensional vector with the original (nonstandardized) inputs. We start RSM with a given input combination or "point" in the \( k \)-dimensional search space; e.g., the combination currently used in practice. In the neighborhood of this point we fit a first-order polynomial, assuming white noise; however, RSM allows \( \text{Var}(w) \) to change in a next step. Unfortunately, there are no general guidelines for determining the appropriate size of the local area in each step. To estimate the local first-order polynomial in \( z \) with first-order effects \( \gamma \) (the standardized effects would be \( \beta \)), we use a R-III design (see Section 2.2). To quantify the adequacy of this estimated polynomial, classic RSM computes \( R^2 \) (see Section 3.5). In the next steps we use \( \nabla(\hat{y}) \), which denotes the gradient implied by this estimated first-order polynomial; so, \( \nabla(\hat{y}) = \hat{\gamma}_{-0} \) where \( \hat{\gamma}_{-0} = 0 \) means that the intercept \( \hat{\gamma}_0 \) is removed from the vector with the estimates \( \hat{\gamma} \) of \( \gamma \). This \( \nabla(\hat{y}) \) implies the steepest descent direction. We take a step in that direction, trying intuitively selected values for the step size. After a number of such steps, \( w \) will increase (instead of decrease) because the local first-order polynomial becomes inadequate. When such deterioration occurs, we simulate the \( n > k \) combinations of the R-III design—but now centered around the best combination found so far. We re-estimate the polynomial and in the resulting new steepest-descent direction we again take several steps. Obviously, a plane (implied by a first-order polynomial) cannot adequately represent a hill top when searching to maximize \( w \) or—equivalently—minimize \( w \). So, in the neighborhood of the latest estimated optimum we now fit a second-order polynomial, using a CCD (see Section 2.4). Next we use the derivatives of this polynomial to estimate the optimum; we may apply canonical analysis to examine the shape of the estimated optimal subregion: does this subregion give a unique minimum, a saddle point, or a ridge with stationary points? To escape from a possible local optimum, we restart the search from a different initial local area—if time permits. While applying RSM, we should not eliminate inputs with nonsignificant effects in a local first-order polynomial: these inputs may become significant in a next local area.

4.2 RSM with adapted steepest descent

Assuming white noise, the adapted steepest descent (ASD) direction accounts for \( \Sigma_{\hat{\gamma}} \), as follows. We write

\[
\Sigma_{\hat{\gamma}} = (Z'_N Z_N)^{-1}\hat{\sigma}_w^2 = \begin{bmatrix} a & b' \\ b & C \end{bmatrix} \hat{\sigma}_w^2
\]

where \( \hat{\sigma}_w^2 \) is the MSR, \( a \) a scalar, \( b \) a \( k \)-dimensional vector, and \( C \) a \( k \times k \) matrix such that \( \Sigma_{\hat{\gamma}_{-0}} = C\hat{\sigma}_w^2 \). The predictor variance \( \text{Var}(\hat{y}|z) \) increases as \( z \) moves away from the local area where \( \nabla(\hat{y}) \) is estimated; actually, \( \text{Var}(\hat{y}|z) \) is minimal at \( z = -C^{-1}b \). ASD means that the new simulated combination is

\[
z = -C^{-1}b - lC_{\hat{\gamma}_{-0}}^{-1}\hat{\gamma}_{-0}.
\]
where \( C^{-1}b \) is the point where the local search starts, \( l \) is the step size, \( \gamma_{0} \) is the classic steepest descent direction, and \( C_{\tilde{\gamma}}^{-1} \tilde{\gamma}_{0} \) is the adapted direction. If \( C \) is diagonal, then the higher the variance of an estimated input effect is, the less the search moves into the direction of that input. It can be proven that ASD is scale-independent. Experimental results imply that ASD performs “better” than steepest descent.

### 4.3 RSM for simulation with multiple outputs

In practice, simulation models have multiple responses types (see Section 3.1). For such situations the RSM literature offers several approaches, but we focus on generalized RSM (GRSM), which solves the following constrained nonlinear random optimization problem:

\[
\min_{z} E(w^{(l)} | z) \\
E(w^{(l')} | z) \geq c_{l'} \quad (l' = 2, \ldots, v) \\
L_{j} \leq z_{j} \leq H_{j} \quad \text{with} \quad j = 1, \ldots, k.
\]

GRSM combines classic RSM and interior point methods developed in MP. GRSM avoids creeping along the boundary of the feasible area that is determined by the constraints on the random outputs and the deterministic inputs, so GRSM moves faster to the optimum. GRSM is scale independent. For details we refer to Kleijnen (2015, pp. 253–258).

Because GRSM may miss the true optimum, we can test the first-order necessary optimality or Karush - Kuhn - Tucker (KKT) conditions. To test these conditions, we may use parametric bootstrapping that samples \( w^{*} \) from the assumed distribution; namely, a multivariate normal distribution with parameters estimated from the original \( w \). Details are given in Kleijnen (2015, pp. 259–266).

### 4.4 RSM for practical random simulations

In practice, random simulations have outputs with variances that change with the input combinations, and with positive correlations if CRN are applied (also see Section 3). Consequently, OLS does not give the BLUE. We assume a constant number of replications so \( m_{i} = m \) \((i = 1, \ldots, n)\), which is realistic if CRN are applied. Using replication \( r \), we then compute \( \gamma_{r} \); see (3). So, replication \( r \) gives an estimator of the steepest descent direction—if a first-order polynomial is used—or the optimum input combination—if a second-order polynomial is used. Together, the \( m \) replications give an estimator of the accuracy of this estimated direction or optimum. If we find this accuracy too low, then we may simulate additional replications so \( m \) increases. Unfortunately, we have not yet any experience with this simple sequential approach for selecting \( m \).

If \( m_{i} \gg 1 \), then we can apply distribution-free bootstrapping to examine the statistical properties of \( \gamma_{r} \) and the resulting steepest descent direction and optimum (see Section 3.2). Kleijnen (2015, pp. 251–252) further discusses RSM for random simulation, including trust regions.
5 Kriging metamodels and their designs

Kriging assumes a global experimental area, which is larger than the local areas in RSM with its sequence of low-order polynomial metamodels (see Section 4). Because we assume that many readers are not familiar with the basics of Kriging, we detail various types of Kriging. We use the same symbols as above, unless Kriging traditionally uses different symbols.

5.1 Ordinary Kriging in deterministic simulation

Ordinary Kriging (OK) is popular and successful in practical deterministic simulation. OK assumes \( y(x) = \mu + M(x) \) where \( \mu \) is the constant mean \( \mathbb{E}[y(x)] \) and \( M(x) \) is a zero-mean Gaussian stationary process, which has covariances that depend only on the distance between the input combinations \( x \) and \( x' \). We call \( M(x) \) the extrinsic noise (to be distinguished from "intrinsic" noise in stochastic simulation; see Section 5.3). Let \( M \) and \( \tilde{M} \) denote the \( n \times k \) matrix with the \( n \) old combinations \( x_i \) (\( i = 1, \ldots, n \)) of the \( k \) simulation inputs, where the original inputs \( z_i \) are standardized to obtain \( x_i \) (unlike DOE, Kriging does not use the symbol \( D \) for the design matrix). The best linear unbiased predictor (BLUP) for the new combination (say) \( x_0 \) is the weighted average of the \( n \) old outputs \( \tilde{y}(x_0) = \sum_{i=1}^n \lambda_i w_i = \mathbf{X} \mathbf{w} \). Because this \( \tilde{y}(x_0) \) is unbiased, \( x_0 = x_i \) implies that the predictor is an exact interpolator: \( \tilde{y}(x_i) = w(x_i) \). The "best," \( \tilde{y}(x_0) \) minimizes the mean squared error (MSE), which equals \( \text{Var}[\tilde{y}(x_0)] \); see Eq. (5) below. Altogether, the optimal \( \mathbf{X} \) is given by the next equation where the \( n \times n \) matrix with the covariances between the metamodel’s old outputs \( y_i \) is denoted by \( \Sigma_M = (\sigma_{i'j'}) = (\text{Cov}(y_i, y_{i'})) \) (\( i, i' = 1, \ldots, n \)), the \( n \)-dimensional vector with the covariances between the metamodel’s new output \( y_0 \) and \( y_i \) is \( \sigma_M(x_0) = (\sigma_{0,i}) = (\text{Cov}(y_0, y_i)) \), and \( \mathbf{1}_n \) is the \( n \)-dimensional vector with ones:

\[
\lambda_o' = \sigma_M(x_0) + \mathbf{1}_n - \frac{1}{\mathbf{1}_{n}' \Sigma_M^{-1}} \mathbf{1}_n \sigma_M(x_0) \Sigma_M^{-1}. \tag{6}
\]

The weight \( \lambda_{i,0} \) in \( \lambda_o \) decreases with the distance between \( x_0 \) and \( x_i \) (so \( \lambda \) is not a constant vector, whereas \( \beta \) in linear regression is). Substitution of \( \lambda_o \) into \( \tilde{y}(x_0) = \mathbf{X} \mathbf{w} \) gives the BLUP; namely,

\[
\tilde{y}(x_0) = \mu + \sigma_M(x_0) \Sigma_M^{-1} (\mathbf{w} - \mathbf{1}_n). \tag{7}
\]

Obviously, \( \tilde{y}(x_0) \) varies with \( \sigma_M(x_0) \), whereas \( \mu \), \( \Sigma_M \), and \( \mathbf{w} \) remain fixed.

Note: The gradient \( \nabla(\tilde{y}) \) follows from Eq. (7); see Lophaven et al. (2002, Eq. 2.18). Sometimes we can also compute \( \nabla(w) \) and estimate a better OK model; see Kleijnen (2015, pp. 183–184).

Instead of the symbol \( \text{Var}(y_i) = \sigma_{i,i} = \sigma_i^2 = \sigma^2 \) we use the classic Kriging symbol \( \tau^2 \) in

\[
\text{Var}[\tilde{y}(x_0)] = \tau^2 - \sigma_M(x_0) \Sigma_M^{-1} \sigma_M(x_0) + \frac{1}{\mathbf{1}_{n}' \Sigma_M^{-1}} \sigma_M(x_0)^2. \tag{8}
\]
This equation implies \( \text{Var}[\hat{y}(x_0)] = 0 \) if \( x_0 = x_j \). Experimental results suggest that \( \text{Var}[\hat{y}(x_0)] \) has local maxima at \( x_0 \) approximately halfway between old input combinations (also see Section 4.2). Kriging gives bad extrapolations compared with interpolations (linear regression also gives minimal \( \text{Var}[\hat{y}(x_0)] \) at the center of the experimental area; see Section 4.2.

Obviously, the correlation matrix \( \mathbf{R} = (\rho_{j,j'}) \) equals \( \tau^{-2} \mathbf{\Sigma}_M \); furthermore, \( \rho(x_0) = \tau^{-2} \mathbf{\sigma}_M(x_0) \). There are several types of correlation functions; see Kleijnen (2015, pp.185–186). Most popular is the Gaussian correlation function:

\[
\rho(h) = \prod_{j=1}^{k} \exp(-\theta_j h_{j}) = \exp(-\sum_{j=1}^{k} \theta_j h_{j}) \tag{9}
\]

with distance vector \( h = (h_j) \) where \( h_j = |x_{g,j} - x_{g',j}| \) and \( g, g' = 0, 1, ..., n \).

Obviously, we need to estimate the \( \text{hyper} \)parameters \( \psi = (\mu, \tau^2, \theta')' \) with \( \theta = (\theta_j) \). The most popular criterion is maximum likelihood (ML) (but OLS and cross-validation are also used). The computation of the ML estimator (MLE) \( \hat{\psi} \) is challenging, so different \( \hat{\psi} \) may result from different software packages or from different starting values for the same package; see Erickson et al. (2017).

Plugging \( \hat{\psi} \) into (7) gives \( \hat{y}(x_0, \hat{\psi}) \). Obviously, \( \hat{y}(x_0, \hat{\psi}) \) is a nonlinear predictor. In practice, we plug \( \hat{\psi} \) into (8) to obtain \( s^2[\hat{y}(x_0, \hat{\psi})] \). To obtain a symmetric \( (1 - \alpha) \) CI for \( w(x_0) \), we use \( z_{\alpha/2} \) (standard symbol for the \( \alpha/2 \) quantile of \( N(0, 1) \)) and get \( \hat{y}(x_0, \hat{\psi}) \pm z_{\alpha/2} s[\hat{y}(x_0, \hat{\psi})] \). There is much software for Kriging; see the many publications and websites in Kleijnen (2015, p. 190).

### 5.2 Designs for Kriging in deterministic simulation

There is an abundant literature on various design types for Kriging in deterministic simulation. Examples of these designs are orthogonal array, uniform, maximum entropy, minimax, maximin, integrated mean squared prediction error, and "optimal" designs; see Kleijnen (2015, p. 198). However, the most popular space filling design uses Latin hypercube sampling (LHS). LHS assumes that the metamodel is more complicated than a low-order polynomial, but LHS does not assume a specific type of metamodel (e.g., OK, detailed in Section 5.1).

LHS standardizes input \( x_j \) so \( 0 \leq x_j \leq 1 \) (\( j = 1, ..., k \)). If \( x_j \) is uncertain (as in RO), then LHS assumes that \( x_j \) has a given PDF; else, LHS assumes that \( x_j \) has a uniform PDF so \( x_j \sim U(0, 1) \). LHS divides the range of \( x_j \) into \( n \) mutually exclusive and exhaustive intervals (or classes) of equal probability; if \( x_j \sim U(0, 1) \), then the length of these intervals is \( 1/n \); if \( x_j \) has a PDF with a mode (e.g., a triangular PDF), then the length is smaller near this mode. LHS samples one value in each interval without replacement, so \( x_j \) has \( n \) different values. LHS may be further refined, leading to maximin LHS, nearly-orthogonal LHS, etc.; see Kleijnen (2015, p. 202).

Whereas DOE makes \( n \) increase with \( k \) (e.g., \( n = 2^{k-p} \)), LHS does not impose such a relationship. Nevertheless, if \( n \) is "small" and \( k \) is "large", then LHS covers the input space so sparsely that the fitted Kriging model may be
inadequate. A well-known rule-of-thumb for LHS in sensitivity analysis through Kriging is $n = 10k$. For SO, however, we replace one-shot designs by sequential designs that are customized for the given simulation model; i.e., we learn about $f_{\text{sim}}$ as we collect I/O data; see Section 6.

5.3 Kriging in random simulation

Ankenman et al. (2010) develops stochastic Kriging (SK), adding the intrinsic noise term $\varepsilon_r(x_i)$ for replication $r$ ($r = 1, \ldots, m_i$) at combination $x_i$ ($i = 1, \ldots, n$). After averaging over these $m_i$ replications, SK uses the formulas for OK but replaces $w$ by $w$ and $M(x_i)$ by $M(x_i) + \varepsilon(x_i)$ where $\varepsilon(x_i) \sim N(0, \text{Var}[\varepsilon_r(x_i)]/m_i)$ and $\varepsilon(x_i)$ is assumed to be independent of $M(x)$. Obviously, $\Sigma_r$ is diagonal if no CRN are used (CRN and $m_i = m$ would give $\Sigma_r = \Sigma_r/m$; however, we assume no CRN in this subsection). To estimate $\text{Var}[\varepsilon(x_i)]$, SK may use either $s^2(w_i)$ or another Kriging model for $\text{Var}[\varepsilon(x_i)]$—besides the Kriging model for the mean $E[y_r(x_i)]$; see Kleijnen (2015, p.208). We use the symbol $\psi_{+\varepsilon}$ to denote $\psi$ augmented with $\text{Var}[\varepsilon_r(x_i)]$.

An alternative for SK is hetGP developed in Binois et al. (2016. This alternative assumes $m_i \geq 1$, whereas SK assumes $m_i \gg 1$. Whereas SK gives a biased $\hat{\psi}_{+\varepsilon}$ because SK fits Kriging models for the mean and the intrinsic variances independently, hetGP couples these models through a joint likelihood for $\psi_{+\varepsilon}$ that is optimized in one shot. This alternative requires computational time of the same order as SK does.

6 Kriging for optimization

Kriging is used by efficient global optimization (EGO), which is a popular sequential method that balances local and global search; i.e., EGO balances exploitation and exploration. We present only the basic EGO-variant for deterministic simulation; also see the classic EGO reference, Jones et al. (1998). There are many more variants, for deterministic and random simulations, constrained optimization, multi-objective optimization including Pareto frontiers, RO, the "excursion set" or "admissible set", estimation of a quantile, and Bayesian approaches; see Kleijnen (2015, p. 267–269).

Note: Moghaddam and Mahlooji (2017) replaces EGO by particle swarm optimization (PSO), still using Kriging. Havinga et al. (2017) adapts EGO, and uses RBFs (instead of Kriging) for RO.

In this basic variant we consider the best output observed (simulated) so far. To select a new combination $x_0$, we consider both $\tilde{y}(x_0)$ and $s^2(\tilde{y}(x))$ (we suppress $\psi$); e.g., if $x_0$ and $x_0'$ have $\tilde{y}(x_0) = \tilde{y}(x_0')$ and $s^2(\tilde{y}(x_0)) > s^2(\tilde{y}(x_0'))$, then we explore $x_0$ because $x_0$ has a higher probability of improvement (lower $w$). We know that $s^2(\tilde{y}(x_0))$ increases as $x_0$ lies farther away from $x_i$; see 3. Actually, we estimate the maximum of the expected improvement (EI), which is reached if either $\tilde{y}(x_0)$ is much smaller than $f_{\text{min}}$ or $s^2(\tilde{y}(x_0))$ is relatively large.
so \( \hat{g}(x_0) \) is relatively uncertain. More precisely, we start with a pilot sample—typically selected through LHS—which results in \((X, w)\). Next we find \( f_{\text{min}} = \min_{1 \leq i \leq n} w(x_i) \). We also fit a Kriging metamodel \( \hat{g}(x) \). Together, this gives

\[
E(\bar{I}(x)) = E[\max (f_{\text{min}} - \hat{g}(x), 0)].
\]

Jones et al. (1998) derives the EI estimator

\[
\tilde{EI}(x) = (f_{\text{min}} - \hat{g}(x)) \Phi \left( \frac{f_{\text{min}} - \hat{g}(x)}{s[\hat{g}(x_0)]} \right) + s[\hat{g}(x_0)] \phi \left( \frac{f_{\text{min}} - \hat{g}(x)}{s[\hat{g}(x_0)]} \right)
\]

(10)

where \( \Phi \) and \( \phi \) denote the cumulative distribution function (CDF) and the PDF of \( N(0, 1) \). Using (10), we estimate \( x_{\text{opt}} \), which denotes the \( x \) that maximizes \( \tilde{EI}(x) \). To find this estimate \( \hat{x}_{\text{opt}} \), we may use a (relatively large) set of candidate points selected through LHS (say) \( X_{\text{cand}} \); we do not simulate these candidates, but we find the candidate with the highest \( \tilde{EI}(x) \) with \( x \in X_{\text{cand}} \). Next we use this candidate \( \hat{x}_{\text{opt}} \) as the simulation input combination, and obtain \( w(\hat{x}_{\text{opt}}) \). Then we fit a new Kriging model to the augmented I/O data. We update \( n \), and return to (10)—until we satisfy a stopping criterion; e.g., \( \tilde{EI}(\hat{x}_{\text{opt}}) \) is “close” to 0 or the computer budget is exhausted.

As an alternative for the various EGO variants we now consider the constrained optimization problem in (5), augmented with constraints \( f_g \) for \( z \) (e.g., budget constraints) and the constraint that \( z \) must belong to the set of nonnegative integers \( N \). To solve this problem, we may apply Kriging and integer mathematical programming (KIMP), which combines (i) sequentialized designs to specify the next combination (like EGO does); (ii) Kriging to obtain explicit functions for \( E(w^{(l)}|z) \) with \( l = 1, ..., v \) (like EGO); (iii) integer nonlinear programming (INLP) to estimate the optimal solution from these explicit Kriging models (without using an EI variant, unlike EGO). Experiments with KIMP and OptQuest suggest that KIMP requires fewer simulated combinations and gives better estimated optima.

7 Robust optimization

The estimated optimum (see the preceding two sections) may turn out to be inferior because this optimum ignores uncertainties in some of the simulation inputs. Taguchi emphasizes that in practice some inputs of a manufactured product (e.g., a car) are under complete control of the engineers (the car’s design), whereas other inputs are not (the driver). Taguchi therefore distinguishes between (i) controllable or decision variables, and (ii) noncontrollable or environmental noisy (or random) factors. We sort the \( k \) simulation inputs such that the first \( k_C \) inputs are controllable, and the next \( k_{NC} \) inputs are noncontrollable. We let \( z_C \) and \( z_{NC} \) denote the vector with the \( k_C \) controllable and the \( k_{NC} \) noncontrollable original (nonstandardized) inputs \( z \).

Taguchi assumes a single output (say) \( w \), focusing on its mean \( E(w) \) and its variance; obviously, this variance is caused by \( z_{NC} \) so \( \sigma^2(w|z_C) > 0 \). For brevity’s sake we denote \( \sigma^2(w|z_C) \) by \( \sigma_w^2 \). Taguchi combines these two outputs into a scalar loss function such as the signal-to-noise or mean-to-variance ratio.
\( E(w)/\sigma^2_w; \) see Myers et al. (2009, pp. 486–488). We, however, prefer to use \( E(w) \) and \( \sigma^2_w \) separately; obviously, \( \sigma^2_w \) has the same scale as \( E(w) \) has. We can then use constrained optimization; i.e., given an upper threshold \( c_\sigma \) for \( \sigma^2_w \), we try to solve \( \min_{w} E(w|z_C) \) such that \( \sigma^2_w \leq c_\sigma \). Constrained optimization is also discussed in Myers et al. (2009, p. 492).

Taguchi’s worldview is successful in production engineering, but statisticians criticize his statistical techniques. Moreover—compared with real-life experiments—simulation experiments have more inputs, more input values, and more input combinations (see again Section 1). Myers et al. (2009, pp. 502–506) combines Taguchi’s worldview with the statisticians’ RSM (see Section 4). Whereas Myers et al. (2009) assumes that \( z_{NC} \) has \( \Sigma_{NC} = \sigma^2_{\Sigma} I \), we assume a general \( \Sigma_{NC} \). Whereas Myers et al. (2009) superimposes contour plots for \( E(w|z_C) \) and \( \sigma(w|z_C) \) to estimate the optimal \( z_C \), we use MP. This MP, however, requires specification of the threshold \( c_\sigma \). In practice, managers may find it hard to select a specific value for \( c_\sigma \), so we may try different \( c_\sigma \) values and estimate the corresponding Pareto-optimal efficiency frontier. To estimate the variability of this frontier that results from the estimators of \( E(w|z_C) \) and \( \sigma(w|z_C) \), we may use bootstrapping. Instead of RSM combined with MP we may apply Kriging with MP. Details are given in Kleijnen (2015, pp. 273–284).

Finally, we summarize RO in MP; see again Bertsimas and Mišić (2017). If MP ignores the uncertainty in the coefficients of the MP model, then the resulting so-called nominal solution may easily violate the constraints in the given model. RO may give a slightly worse value for the goal variable, but RO increases the probability of satisfying the constraints; i.e., a robust solution is immune to variations of the variables within the so-called uncertainty set \( U \). Yanikoglu et al. (2016) derives a specific \( U \) for the unknown PDF of \( z_{NC} \) that is compatible with the given historical data on \( z_{NC} \). RO in MP develops a computationally tractable robust counterpart of the original problem. Compared with the output of the nominal solution, RO may give better worst-case and average outputs.

References:


Dellino G, Kleijnen JPC, Meloni C (2010) Robust optimization in simulation:


Friese, M., T. Bartz-Beielstein, and M. Emmerich (2016), BUILDING ENSEMBLES OF SURROGATES BY OPTIMAL CONVEX COMBINATION. Conference paper

Gramacy, R.B. (2015), laGP: large-scale spatial modeling via local approximate Gaussian processes in R. Journal of Statistical Software (available as a vignette in the laGP package)


Myers, R.H., D.C. Montgomery, and C.M. Anderson-Cook (2009), Response


