# Simulation Optimization through Regression or Kriging Metamodels

Kleijnen, J.P.C.

*Publication date:*
2017

*Document Version*
Early version, also known as pre-print

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 uses 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-

retical 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 $w = f_{\text{sim}}(\mathbf{z}, r)$ where $w$ denotes the scalar simulation output (e.g., average waiting time), $f_{\text{sim}}$ the implicit I/O function of the simulation model, $\mathbf{z}$ 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), $\mathbf{z}$ is *standardized* (scaled) such that the resulting input $\mathbf{d}$ has elements either $-1 \leq d_j \leq 1$ or $0 \leq d_j \leq 1$ with $j = 1, ..., k$. We approximate $w = f_{\text{sim}}(\mathbf{z},r)$ by $y = f_{\text{meta}}(\mathbf{x}) + e$ 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, $\mathrm{E}(w)$; this $\mathrm{E}(w)$ 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 $\mathbf{y} = \mathbf{X}_N\boldsymbol{\beta} + \mathbf{e}$ where $\mathbf{y}$ 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 $N = n$), $\mathbf{X}_N$ is the $N \times q$ matrix of independent (explanatory) regression variables—obviously, $\mathbf{X}_N$ has $m_i$ identical rows, whereas $\mathbf{X}_n$ denotes the corresponding $n \times q$ matrix without any identical rows determined by the $n \times k$ design matrix $\mathbf{D}$ and the type of regression model (e.g., second-order polynomial), $\boldsymbol{\beta}$ denotes the $q$-dimensional vector with regression parameters (coefficients), and $\mathbf{e}$ denotes the $N$-dimensional vector with the residuals $\mathrm{E}(\mathbf{y}) - \mathrm{E}(\mathbf{w})$ where $\mathbf{w}$ 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 $\mathbf{x}_i = (x_{i;1}, ..., x_{i;q})$.

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) $\beta_{j;j'}$ ($j < j'$), and $k$ purely quadratic effects $\beta_{j;j}$. 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 $\mathbf{x}$, 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 3.5).

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

Given the equality $\boldsymbol{\Sigma}_{\hat{\beta}} = (\mathbf{X}'_N\mathbf{X}_N)^{-1}\sigma_w^2$, we may select $\mathbf{X}_N$ such that we "optimize" $\boldsymbol{\Sigma}_{\hat{\beta}}$. Obviously, $\mathbf{X}_N$ is determined by $\mathbf{X}_n$, $m_i$, and the type of regression model (e.g., $\mathbf{X}_N$ may include $x_j^2$). 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 $\mathbf{X}_n$ implies an orthogonal $\mathbf{X}_N$ (we may specify $\mathbf{X}_N$ as $\mathbf{X}_n$ "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, $\mathbf{X}_n$ is determined by $\mathbf{D}$. To select a specific $\mathbf{D}$ with $z_{i;j}$ standardized such that $-1 \leq d_{i;j} \leq 1$, we try to minimize $\text{Var}(\widehat{\beta}_g)$ ($g = 1, ..., q$); other criteria are discussed in Kleijnen (2015, pp. 66–67). We can prove that this minimization requires an orthogonal $\mathbf{X}_N$, which gives $\mathbf{\Sigma}_{\hat{\beta}} = (N\mathbf{I})^{-1}\sigma_w^2 = \mathbf{I}\sigma_w^2/N$. Because this $\mathbf{\Sigma}_{\hat{\beta}}$ is diagonal, the $\widehat{\beta}_j$ are statistically independent. Moreover, these $\widehat{\beta}_j$ have the same variance; namely, $\sigma_w^2/N$. So we can *rank* the explanatory variables in order of importance, using either $\widehat{\beta}_g$ or $t_{N-q}$ with $\beta_g = 0$ (so $t_{N-q} = \widehat{\beta}_g/s(\widehat{\beta}_g)$); usually, we do not hypothesize that the intercept $\beta_0$ is zero.

Note: If all $\widehat{\beta}_g$ 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 $\widehat{\beta}_g$ 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 $\widehat{\beta}_j$ ($j = 1, ..., k$). However, these $\widehat{\beta}_j$ do not have minimum variances; a R-III design does minimize these variances because this design gives an orthogonal $\mathbf{X}_N$, as we shall see in this subsection. Furthermore, we can prove that $\text{Var}(\widehat{\beta}_j)$ 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 $2_{III}^{k-p}$ with integer $p$ such that $0 \leq p < k$ and $2^{k-p} \geq 1 + k$: we first discuss these $2_{III}^{k-p}$ 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* $\mathbf{X}_n$. 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 $2_{III}^{k-p}$ designs is given in Kleijnen (2015, pp. 53–54).

Whereas $2_{III}^{k-p}$ 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 $2_{III}^{12-8}$ implies $n = 16$). If $8 \leq k < 11$, then we do not use $n-(k+1)$ columns of $\mathbf{D}$. 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 $2^{k-p}_{III}$ designs, these PB designs are balanced and they give an orthogonal $\mathbf{X}_n$.

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

A R-V design enables unbiased $\widehat{\beta}_j$ and $\widehat{\beta}_{j;j'}$with $j < j'$. Obviously, $q$ equals $1 + k + k(k-1)/2$. The DOE literature gives tables for generating $2^{k-p}_V$ designs. Unfortunately, these designs are not saturated at all; e.g., the $2^{8-2}_V$ design implies $n = 64 \gg q = 37$. *Rechtschaffner* 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 $\widehat{\beta}_j$ and $\widehat{\beta}_{j;j'}$with $j \leq j'$. A CCD consists of three subdesigns: (i) a R-V design; (ii) the *central* combination $\mathbf{0}'_k$; (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* $\mathbf{X}_n$; 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 4). 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: $\widehat{\beta}^{(l)} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{w}^{(l)}$. Furthermore, CIs and tests for the elements in $\widehat{\beta}^{(l)}$ 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 ($i = 1, ..., n$), we obtain $\overline{w}_i = \sum_{r=1}^{m_i} w_{i;r}/m_i$ and the corresponding $\widehat{e}_i = \widehat{y}_i - \overline{w}_i$. For simplicity of presentation, we assume that $m_i$ is a constant $m$. If $w_{i;r}$ has a constant variance $\sigma_w^2$, then $\overline{w}_i$ also has a constant variance; namely, $\sigma_{\overline{w}}^2 = \sigma_w^2/m$. Unfortunately, even if $\overline{w}_i$ has a constant variance $\sigma_{\overline{w}}^2$ and is independent of $\overline{w}_{i'}$ with $i \neq i'$ (no CRN), then $\mathbf{\Sigma}_{\widehat{\mathbf{e}}} = [\mathbf{I} - \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}']\sigma_{\overline{w}}^2$ so $\widehat{\mathbf{e}}$ 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, $w_{i;1}, ..., w_{i;m_i}$ 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, $m_i \gg 1$. We apply this resampling to each combination $i$. The resulting $w_{i;1}^*, ..., w_{i;m_i}^*$ give the average $\overline{w}_i^*$, which give the $n$-dimensional vector $\overline{\mathbf{w}}^*$. For simplicity's sake, we now assume $m_i = m > 1$, so the bootstrapped OLS estimator of $\beta$

is $\widehat{\beta}^* = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\overline{\mathbf{w}}^*$. 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 $\widehat{\beta}^*_b$ with $b = 1, ..., B$. 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 $\widehat{\beta}^*_q$ obtained through sorting the $B$ observations on $\widehat{\beta}^*_{q;b}$. 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 $\widehat{\beta}^*_{q;(B\alpha/2)}$. Analogously we obtain $\widehat{\beta}^*_{q;(B[1-\alpha/2])}$. These two quantiles give a two-sided asymmetric $(1-\alpha)$ CI: $\widehat{\beta}^*_{q;(B\alpha/2)} < \beta_q < \widehat{\beta}^*_{q;(B[1-\alpha/2])}$. 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 $\mathbf{x}_i$ 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 $s^2(w_i) = \sum_{r=1}^{m_i}(w_{i;r} - \overline{w}_i)^2/(m_i - 1)$. 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 $\mathbf{\Sigma}_{\hat{\beta}}$ becomes $(\mathbf{X}'_n\mathbf{X}_n)^{-1}\mathbf{X}'_n\mathbf{\Sigma}_{\overline{\mathbf{w}}}\mathbf{X}_n(\mathbf{X}'_n\mathbf{X}_n)^{-1}$ where for simplicity's sake we assume $m_i = m$ so $\mathbf{\Sigma}_{\overline{\mathbf{w}}}$ 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 $s^2(w_i)/m_i$ $(i = 1, ..., n)$. Therefore we initially take a *pilot sample* of size $m_0 \geq 2$ for each combination, which gives (say) $s_i^2(m_0)$. Next we select a number of additional replications $\widehat{m}_i - m_0$ with

$$\widehat{m}_i = m_0 \times nint \left[\frac{s_i^2(m_0)}{min_i\ s_i^2(m_0)}\right] \tag{1}$$

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

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

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

### 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 $w_{i;r}$ ($i = 1, ..., n$; $r = 1, ..., m$) into a matrix $\mathbf{W} = (\mathbf{w}_1, ..., \mathbf{w}_m)$ with $\mathbf{w}_r = (w_{1;r}, ..., w_{n;r})'$. Obviously, CRN create correlation between $w_{i;r}$ and $w_{i';r}$. Moreover, different replications use nonoverlapping PRN streams so $w_{i;r}$ and $w_{i;r'}$ with $r \neq r'$—or the $n$-dimensional vectors $\mathbf{w}_r$ and $\mathbf{w}_{r'}$—are independent. CRN are meant to reduce $\text{Var}(\widehat{\beta}_g)$ and $\text{Var}(\widehat{y})$; 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 $\hat{\beta}$, we do not use $\mathbf{W}$, but the vector $\mathbf{w}$ with $N = \sum_{i=1}^{n} m_i$ elements. To compute $\widehat{\mathbf{\Sigma}}_{\hat{\beta}}$ in case of CRN, we use the non-diagonal matrix $\widehat{\mathbf{\Sigma}}_{\mathbf{w}}$. Unfortunately, this $\widehat{\mathbf{\Sigma}}_{\mathbf{w}}$ is *singular* if $m \leq n$; if $m > n$, then we may compute CIs for $\widehat{\beta}_j$ from $t_{m-1}$. An alternative method requires only $m > 1$, and computes

$$\hat{\beta}_r = (\mathbf{X}_n' \mathbf{X}_n)^{-1} \mathbf{X}_n' \mathbf{w}_r (r = 1, ..., m). \tag{3}$$

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

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 $\overline{w}_i$ has the same—absolute or relative—width of the CI around $\overline{w}_i$; 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 $R^2 = \sum_{i=1}^{n} (\widehat{y}_i - \overline{\overline{w}})^2 / \sum_{i=1}^{n} (\overline{w}_i - \overline{\overline{w}})^2 = 1 - \sum_{i=1}^{n} (\widehat{y}_i - \overline{w}_i)^2 / \sum_{i=1}^{n} (\overline{w}_i - \overline{\overline{w}})^2$ where $\overline{\overline{w}} = \sum_{i=1}^{n} \overline{w_i}/n$ and $m_i \geq 1$. If $n = q$ (saturated design), then $R^2 = 1$—even if $\text{E}(\widehat{e}_i) \neq 0$. If $n > q$ and $q$ increases, then $R^2$ increases—whatever the size of $|\text{E}(\widehat{e}_i)|$ is; because of possible *overfitting*, we may therefore use the *adjusted* $R^2$: $R^2_{\text{adj}} = 1 - (1 - R^2)(n-1)/(n-q)$. Unfortunately, we do not know *critical values* for $R^2$ or $R^2_{\text{adj}}$. 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 $R^2_{\text{adj}}$. Suppose again that $m_i = m \geq 1$, so we replace $\mathbf{w}$ by $\overline{\mathbf{w}}$ in OLS. In cross-validation, we delete I/O combination $i$ to obtain $(\mathbf{X}_{-i}, \overline{\mathbf{w}}_{-i})$ where we suppress the subscript $n$ of $\mathbf{X}$. Next we compute $\widehat{\beta}_{-i} = (\mathbf{X}_{-i}' \mathbf{X}_{-i})^{-1} \mathbf{X}_{-i}' \overline{\mathbf{w}}_{-i}$ ($i = 1, ..., n$). This gives $\widehat{y}_{-i} = \mathbf{x}_i' \widehat{\beta}_{-i}$. We may "eyeball" the *scatterplot* with $(\overline{w}_i, \widehat{y}_{-i})$, 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*

$$t_{m-1}^{(i)} = \frac{\overline{w}_i - \widehat{y}_{-i}}{\sqrt{s^2(\overline{w}_i) + s^2(\widehat{y}_{-i})}} \tag{4}$$

where $s^2(\overline{w}_i) = s^2(w_i)/m$ and $s^2(\widehat{y}_{-i}) = \mathbf{x}_i' \widehat{\mathbf{\Sigma}}_{\widehat{\beta}_{-i}} \mathbf{x}_i$ with $\widehat{\mathbf{\Sigma}}_{\widehat{\beta}_{-i}} = s^2(\overline{w}_i)(\mathbf{X}_{-i}' \mathbf{X}_{-i})^{-1}$. We reject the metamodel if $max_i \; |t_{m-1}^{(i)}| > t_{m-1;1-[\alpha/(2n)]}$ 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 $\widehat{y}_{-i}$, but also $\widehat{\beta}_{-i}$ (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 $\widehat{\beta}_{-i}$ remain stable?

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

If the validation suggests am 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 $k$ *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 $\mathrm{E}(w|\mathbf{z})$ where $w$ denotes the simulation output and $\mathbf{z}$ 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 $\mathrm{Var}(w)$ 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 $\mathbf{z}$ 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 $\nabla(\widehat{y})$, which denotes the *gradient* implied by this estimated first-order polynomial; so, $\nabla(\widehat{y}) = \widehat{\gamma}_{-0}$ where $-0$ means that the intercept $\widehat{\gamma}_0$ is removed from the vector with the estimates $\widehat{\gamma}$ of $\gamma$. This $\nabla(\widehat{y})$ 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 $\mathbf{\Sigma}_{\widehat{\gamma}}$, as follows. We write

$$\mathbf{\Sigma}_{\widehat{\gamma}} = (\mathbf{Z}_N'\mathbf{Z}_N)^{-1}\widehat{\sigma}_w^2 = \begin{bmatrix} a & \mathbf{b}' \\ \mathbf{b} & \mathbf{C} \end{bmatrix} \widehat{\sigma}_w^2$$

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

$$\mathbf{z} = -\mathbf{C}^{-1}\mathbf{b} - l\mathbf{C}_{\widehat{\gamma}_{-0}}^{-1}\widehat{\gamma}_{-0}.$$

where $\mathbf{C}^{-1}\mathbf{b}$ is the point where the local search starts, $l$ is the step size, $\widehat{\gamma}_{-0}$ is the classic steepest descent direction, and $\mathbf{C}^{-1}_{\widehat{\gamma}_{-0}}\widehat{\gamma}_{-0}$ is the adapted direction. If $\mathbf{C}$ 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*:

$$\begin{aligned} &\min_{\mathbf{z}} \mathrm{E}(w^{(1)}|\mathbf{z}) \\ &\mathrm{E}(w^{(l')}|\mathbf{z}) \geq c_{l'} \ (l' = 2, \ldots, v) \\ &L_j \leq z_j \leq H_j \text{ with } j = 1, \ldots, k. \end{aligned} \tag{5}$$

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 $\widehat{\gamma}_r$; 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 $\widehat{\gamma}$ 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 $y(\mathbf{x}) = \mu + M(\mathbf{x})$ where $\mu$ is the constant mean $\mathrm{E}[y(\mathbf{x})]$ and $M(\mathbf{x})$ is a zero-mean *Gaussian stationary process*, which has covariances that depend only on the distance between the input combinations $\mathbf{x}$ and $\mathbf{x}'$. We call $M(\mathbf{x})$ the *extrinsic noise* (to be distinguished from "intrinsic" noise in stochastic simulation; see Section 5.3). Let $\mathbf{X}$ denote the $n \times k$ matrix with the $n$ *old* combinations $\mathbf{x}_i$ ($i = 1, ..., n$) of the $k$ simulation inputs, where the original inputs $\mathbf{z}_i$ are standardized to obtain $\mathbf{x}_i$ (unlike DOE, Kriging does not use the symbol $\mathbf{D}$ for the design matrix). The *best linear unbiased predictor* (BLUP*)* for the *new* combination (say) $\mathbf{x}_0$ is the weighted average of the $n$ old outputs $\widehat{y}(\mathbf{x}_0) = \sum_{i=1}^{n} \lambda_i w_i = \boldsymbol{\lambda}'\mathbf{w}$. Because this $\widehat{y}(\mathbf{x}_0)$ is unbiased, $\mathbf{x}_0 = \mathbf{x}_i$ implies that the predictor is an *exact interpolator*: $\widehat{y}(\mathbf{x}_i) = w(\mathbf{x}_i)$. The "best" $\widehat{y}(\mathbf{x}_0)$ minimizes the *mean squared error* (MSE), which equals $\mathrm{Var}[\widehat{y}(\mathbf{x}_0)]$; see (8) below. Altogether, the *optimal* $\boldsymbol{\lambda}$ 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 $\boldsymbol{\Sigma}_M = (\sigma_{i;i'}) = (\mathrm{Cov}(y_i, y_{i'}))$ $(i, i' = 1, ..., n)$, the $n$-dimensional vector with the covariances between the metamodel's new output $y_0$ and $y_i$ is $\sigma_M(\mathbf{x}_0) = (\sigma_{0;i}) = (\mathrm{Cov}(y_0, y_i))$, and $\mathbf{1}_n$ is the $n$-dimensional vector with ones:

$$\lambda_o' = [\sigma_M(\mathbf{x}_0) + \mathbf{1}_n \frac{1 - \mathbf{1}_n' \boldsymbol{\Sigma}_M^{-1} \sigma(\mathbf{x}_0)}{\mathbf{1}_n' \boldsymbol{\Sigma}_M^{-1} \mathbf{1}_n}]' \boldsymbol{\Sigma}_M^{-1}. \tag{6}$$

The weight $\lambda_{i;0}$ in $\lambda_o$ decreases with the *distance* between $\mathbf{x}_0$ and $\mathbf{x}_i$ (so $\lambda$ is not a constant vector, whereas $\beta$ in linear regression is). Substitution of $\lambda_o$ into $\widehat{y}(\mathbf{x}_0) = \boldsymbol{\lambda}'\mathbf{w}$ gives the BLUP; namely,

$$\widehat{y}(\mathbf{x}_0) = \mu + \sigma_M(\mathbf{x}_0)' \boldsymbol{\Sigma}_M^{-1} (\mathbf{w} - \mu \mathbf{1}_n). \tag{7}$$

Obviously, $\widehat{y}(\mathbf{x}_0)$ varies with $\sigma_M(\mathbf{x}_0)$, whereas $\mu$, $\boldsymbol{\Sigma}_M$, and $\mathbf{w}$ remain fixed.

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

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

$$\mathrm{Var}[\widehat{y}(\mathbf{x}_0)] = \tau^2 - \sigma_M(\mathbf{x}_0)' \boldsymbol{\Sigma}_M^{-1} \sigma_M(\mathbf{x}_0) + \frac{[1 - \mathbf{1}_n' \boldsymbol{\Sigma}_M^{-1} \sigma_M(\mathbf{x}_0)]^2}{\mathbf{1}_n' \boldsymbol{\Sigma}_M^{-1} \mathbf{1}_n}. \tag{8}$$

This equation implies $\text{Var}[\widehat{y}(\mathbf{x}_0)] = 0$ if $\mathbf{x}_0 = \mathbf{x}_i$. Experimental results suggest that $\text{Var}[\widehat{y}(\mathbf{x}_0)]$ has *local maxima* at $\mathbf{x}_0$ approximately halfway between old input combinations (also see Section 6). Kriging gives bad extrapolations compared with interpolations (linear regression also gives minimal $\text{Var}[\widehat{y}(\mathbf{x}_0)]$ at the center of the experimental area; see Section 4.2).

Obviously, the correlation matrix $\mathbf{R} = (\rho_{i;i'})$ equals $\tau^{-2}\mathbf{\Sigma}_M$; furthermore, $\rho(\mathbf{x}_0) = \tau^{-2}\sigma_M(\mathbf{x}_0)$. There are several types of correlation functions; see Kleijnen (2015, pp.185–186). Most popular is the *Gaussian correlation function*:

$$\rho(\mathbf{h}) = \prod_{j=1}^{k} \exp\ \left(-\theta_j h_j^2\right) = \exp\ \left(-\sum_{j=1}^{k} \theta_j h_j^2\right) \tag{9}$$

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

Obviously, we need to *estimate* the *(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) $\widehat{\psi}$ is challenging, so different $\widehat{\psi}$ may result from different software packages or from different starting values for the same package; see Erickson et al. (2017).

*Plugging* $\widehat{\psi}$ into (7) gives $\widehat{y}(\mathbf{x}_0, \widehat{\psi})$. Obviously, $\widehat{y}(\mathbf{x}_0, \widehat{\psi})$ is a *nonlinear* predictor. In practice, we *plug* $\widehat{\psi}$ into (8) to obtain $s^2[\widehat{y}(\mathbf{x}_0, \widehat{\psi})]$. To obtain a symmetric $(1 - \alpha)$ *CI* for $w(\mathbf{x}_0)$, we use $z_{\alpha/2}$ (standard symbol for the $\alpha/2$ quantile of $\text{N}(0, 1)$) and get $\widehat{y}(\mathbf{x}_0, \widehat{\psi}) \pm z_{\alpha/2} s[\widehat{y}(\mathbf{x}_0, \widehat{\psi})]$. 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 $x_j \sim \text{U}(0, 1)$. LHS divides the range of $x_j$ into $n$ mutually exclusive and exhaustive intervals (or classes) of equal probability; if $x_j \sim \text{U}(0, 1)$, 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 $\varepsilon_r(\mathbf{x}_i)$ for replication $r$ ($r$= 1, ..., $m_i$) at combination $\mathbf{x}_i$ ($i = 1, ..., n$). After averaging over these $m_i$ replications, SK uses the formulas for OK but replaces $\mathbf{w}$ by $\overline{\mathbf{w}}$ and $M(\mathbf{x}_i)$ by $M(\mathbf{x}_i) + \overline{\varepsilon}(\mathbf{x}_i)$ where $\overline{\varepsilon}(\mathbf{x}_i) \sim N(0,\text{Var}[\varepsilon_r(\mathbf{x}_i)]/m_i)$ and $\overline{\varepsilon}(\mathbf{x}_i)$ is assumed to be independent of $M(\mathbf{x})$. Obviously, $\mathbf{\Sigma}_{\overline{\varepsilon}}$ is diagonal if no CRN are used (CRN and $m_i = m$ would give $\mathbf{\Sigma}_{\overline{\varepsilon}} = \mathbf{\Sigma}_{\varepsilon}/m$; however, we assume no CRN in this subsection). To estimate $\text{Var}[\varepsilon(\mathbf{x}_i)]$, SK may use either $s^2(w_i)$ or another Kriging model for $\text{Var}[\varepsilon(\mathbf{x}_i)]$—besides the Kriging model for the mean $\text{E}[y_r(\mathbf{x}_i)]$; see Kleijnen (2015, p.208). We use the symbol $\psi_{+\varepsilon}$ to denote $\psi$ augmented with $\text{Var}[\varepsilon_r(\mathbf{x}_i)$.

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 $\widehat{\psi}_{+\varepsilon}$ 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 $\mathbf{x}_0$, we consider both $\widehat{y}(\mathbf{x}_0)$ and $s^2[\widehat{y}(\mathbf{x})]$ (we suppress $\widehat{\psi}$); e.g., if $\mathbf{x}_0$ and $\mathbf{x}_0'$ have $\widehat{y}(\mathbf{x}_0) = \widehat{y}(\mathbf{x}_0')$ and $s^2[\widehat{y}(\mathbf{x}_0)] > s^2[\widehat{y}(\mathbf{x}_0')]$, then we explore $\mathbf{x}_0$ because $\mathbf{x}_0$ has a higher probability of improvement (lower $w$). We know that $s^2[\widehat{y}(\mathbf{x}_0)]$ increases as $\mathbf{x}_0$ lies farther away from $\mathbf{x}_i$; see (8). Actually, we estimate the maximum of the *expected improvement* (EI), which is reached if either $\widehat{y}(\mathbf{x}_0)$ is much smaller than $f_{\min}$ or $s^2[\widehat{y}(\mathbf{x}_0)]$ is relatively large

so $\widehat{y}(\mathbf{x}_0)$ is relatively uncertain. More precisely, we start with a pilot sample—typically selected through LHS—which results in $(\mathbf{X}, \mathbf{w})$, Next we find $f_{\min} = \min_{1 \leq i \leq n} w(\mathbf{x}_i)$. We also fit a Kriging metamodel $\widehat{y}(\mathbf{x})$. Together, this gives $\text{EI}(\mathbf{x}) = \text{E}[\max(f_{\min} - \widehat{y}(\mathbf{x}), 0)]$. Jones et al. (1998) derives the EI estimator

$$\widehat{\text{EI}}(\mathbf{x}) = (f_{\min} - \widehat{y}(\mathbf{x}))\, \Phi\left(\frac{f_{\min} - \widehat{y}(\mathbf{x})}{s[\widehat{y}(\mathbf{x}_0)]}\right) + s[\widehat{y}(\mathbf{x}_0)]\phi\left(\frac{f_{\min} - \widehat{y}(\mathbf{x})}{s[\widehat{y}(\mathbf{x}_0)]}\right) \qquad (10)$$

where $\Phi$ and $\phi$ denote the cumulative distribution function (CDF) and the PDF of $\text{N}(0, 1)$. Using (10), we estimate $\mathbf{x}_{opt}$, which denotes the $\mathbf{x}$ that maximizes $\widehat{\text{EI}}(\mathbf{x})$. To find this estimate $\widehat{\mathbf{x}}_{opt}$, we may use a (relatively large) set of candidate points selected through LHS (say) $\mathbf{X}_{cand}$; we do not simulate these candidates, but we find the candidate with the highest $\widehat{\text{EI}}(\mathbf{x})$ with $\mathbf{x} \in \mathbf{X}_{cand}$. Next we use this candidate $\widehat{\mathbf{x}}_{opt}$ as the simulation input combination, and obtain $w(\widehat{\mathbf{x}}_{opt})$. 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., $\widehat{\text{EI}}(\widehat{\mathbf{x}}_{opt})$ 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 $\mathbf{z}$ (e.g., budget constraints) and the constraint that $z$ must belong to the set of non-negative integers $\mathbf{N}$. 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 $\text{E}(w^{(l)}|z)$ 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 $k_{\text{C}}$ inputs are controllable, and the next $k_{\text{NC}}$ inputs are noncontrollable. We let $\mathbf{z}_{\text{C}}$ and $\mathbf{z}_{\text{NC}}$ denote the vector with the $k_{\text{C}}$ controllable and the $k_{\text{NC}}$ noncontrollable original (nonstandardized) inputs $\mathbf{z}$.

Taguchi assumes a single output (say) $w$, focusing on its mean $\text{E}(w)$ and its variance; obviously, this variance is caused by $\mathbf{z}_{\text{NC}}$ so $\sigma^2(w|\mathbf{z}_{\text{C}}) > 0$. For brevity's sake we denote $\sigma^2(w|\mathbf{z}_{\text{C}})$ 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

$\mathrm{E}(w)/\sigma_w^2$; see Myers et al. (2009, pp. 486–488). We, however, prefer to use $\mathrm{E}(w)$ and $\sigma_w^2$ *separately*; obviously, $\sigma_w^2$ has the same scale as $\mathrm{E}(w)$ has. We can then use *constrained optimization*; i.e., given an upper threshold $c_\sigma$ for $\sigma_w^2$, we try to solve $\min_{\mathbf{z}_\mathrm{C}} \mathrm{E}(w|\mathbf{z}_\mathrm{C})$ such that $\sigma_w^2 \leq c_\sigma$. 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 $\mathbf{z}_\mathrm{NC}$ has $\mathbf{\Sigma}_\mathrm{NC} = \sigma_w^2 I$, we assume a general $\mathbf{\Sigma}_\mathrm{NC}$. Whereas Myers et al. (2009) superimposes contour plots for $\mathrm{E}(w|\mathbf{z}_\mathrm{C})$ and $\sigma(w|\mathbf{z}_\mathrm{C})$ to estimate the optimal $\mathbf{z}_\mathrm{C}$, 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 $\mathrm{E}(w|\mathbf{z}_\mathrm{C})$ and $\sigma(w|\mathbf{z}_\mathrm{C})$, 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* $U$. Yanikoğlu et al. (2016) derives a specific $U$ for the unknown PDF of $\mathbf{z}_\mathrm{NC}$ that is compatible with the given historical data on $\mathbf{z}_\mathrm{NC}$. 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**:


Ankenman, B., B. Nelson, and J. Staum (2010), Stochastic Kriging for simulation metamodeling. *Operations Research*, 58, no. 2, pp. 371–382

Bartz-Beielstein, T. and M. Zaefferer (2017), Model-based methods for continuous and discrete global optimization. Applied Soft Computing, 55, pp. 154–167

Bertsimas, D. and V.V. Mišić (2017) Robust Product Line Design. *Operations Research*, 65, no. 1, pp. 19–37

Binois, M., R.B. Gramacy and M. Ludkovskiz (2016), Practical heteroskedastic Gaussian process modeling for large simulation experiments. *ArXiv*, 17 Nov 2016

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

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

Taguchi and response surface methodology. Int J Prod Econ 125(1):52–59

Dellino G, Kleijnen JPC, Meloni C (2012) Robust optimization in simulation: Taguchi and Krige combined. INFORMS J Comput 24(3):471–484

Erickson, C.B., S.M. Sanchez, and B.E. Ankenman (2016), Comparison of Gaussian process modeling software. *WSC 2016 Proceedings*, pp. 3692–3693

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)

Havinga J., A. H. van den Boogaard, and G. Klaseboer (2017), Sequential improvement for robust optimization using an uncertainty measure for radial basis functions. *Structural and Multidisciplinary Optimization*, 55, pp. 1345–1363

Jalali H. and I. Van Nieuwenhuyse (2015), Simulation optimization in inventory replenishment: a classification. *IIE Transactions*, 47, no. 11, pp. 1217–1235

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

Kleijnen, J. P. C. (2015), *Design and analysis of simulation experiments, second edition*. Springer

Kleijnen, J. P. C. (2017), Design and analysis of simulation experiments: tutorial. Invited chapter for Advances in Modeling and Simulation: Seminal Research from 50 Years of Winter Simulation Conferences, edited by A. Tolk, J. Fowler, G. Shao, and E. Yucesan, Springer (preprint: CentER Discussion Paper Series No. 2017-018)

Kleijnen, J. P. C. and W. Shi (2017), Sequential probability ratio tests: conservative and robust. CentER Discussion Paper; Vol. 2017-001, Tilburg: CentER, Center for Economic Research

Law, A.M. (2015), *Simulation modeling and analysis, 5th edition*. McGraw-Hill, Boston

Liu, Z., D. Rexachs, F. Epelde, and E. Luque (2017), A simulation and optimization based method for calibrating agent-based emergency department models under data scarcity. *Computers & Industrial Engineering*, 103, pp. 300–309

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

Moghaddam„S. and H. Mahlooji (2017), A new metamodel-based method for solving semi-expensive simulation optimization problems, *Communications in Statistics - Simulation and Computation*, in press

Montgomery, D. C. (2009). *Design and Analysis of Experiments; 7th Edition*, Wiley, Hoboken, NJ

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

*Surface Methodology: Process and Product Optimization Using Designed Experiments; Third Edition.* Wiley, New York

Pontes, F.J., G.F. Amorim, P.P. Balestrassi, A.P. Paiva, and J.R. Ferreira (2016), Design of experiments and focused grid search for neural network parameter optimization. Neurocomputing, 186, pp. 22-34

Rasmussen, C.E. and C. Williams (2006), *Gaussian processes for machine learning.* MIT, Cambridge

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, 1989, pp. 409–435

Sanchez SM, Lucas TW, Sanchez PJ, Nannini CJ, Wan H (2012) Chapter 12: designs for large-scale simulation experiments, with applications to defense and homeland security. In: Hinkelmann K (ed) Design and analysis of experiments, volume 3, special designs and applications. Wiley, New York, pp 413–442

Simpson T.W., A.J. Booker, D. Ghosh, A.A. Giunta, P.N. Koch, R-J. Yang (2004), Approximation methods in multidisciplinary analysis and optimization: a panel discussion. *Structural and Multidisciplinary Optimization*, 27, no, 5, pp. 302–313

Yanikoğlu, I., den Hertog, D. & Kleijnen, J. P. C. (2016), Robust dual-response optimization. *IIE Transactions: Industrial Engineering Research and Development*, 48, no. 3, pp. 298–312

Zeigler B.P., H. Praehofer, T.G. Kim (2000), *Theory of Modeling and Simulation, 2nd Edition.* Academic, San Diego