6
Simulation Optimization

This chapter is organized as follows. Section 6.1 introduces the optimization of real systems that are modeled through either deterministic or random simulation; this optimization we call *simulation optimization* or briefly *optimization*. There are many methods for this optimization, but we focus on methods that use specific metamodels of the underlying simulation models; these metamodels were detailed in the preceding chapters, and use either linear regression or Kriging. Section 6.2 discusses the use of linear regression metamodels for optimization. Section 6.2.1 summarizes basic *response surface methodology* (RSM), which uses linear regression; RSM was developed for experiments with real systems. Section 6.2.2 adapts this RSM to the needs of random simulation. Section 6.2.3 presents the *adapted steepest descent* (ASD) search direction. Section 6.2.4 summarizes *generalized RSM* (GRSM) for simulation with multiple responses. Section 6.2.5 summarizes a procedure for testing whether an estimated optimum is truly optimal—using the *Karush-Kuhn-Tucker* (KKT) conditions. Section 6.3 discusses the use of Kriging metamodels for optimization. Section 6.3.1 presents *efficient global optimization* (EGO), which uses Kriging. Section 6.3.2 presents *Kriging and integer mathematical programming* (KrIMP) for the solution of problems with constrained outputs. Section 6.4 discusses *robust optimization* (RO), which accounts for uncertainties in some inputs. Section 6.4.1 discusses RO using RSM, Sect. 6.4.2 discusses RO using Kriging, and Sect. 6.4.3 summarizes the *Ben-Tal* et al. approach to RO. Section 6.5

J.P.C. Kleijnen, *Design and Analysis of Simulation Experiments*,
International Series in Operations Research & Management
Science 230, DOI 10.1007/978-3-319-18087-8_6

summarizes the major conclusions of this chapter, and suggests topics for future research. The chapter ends with Solutions of exercises, and a long list of references.

6.1 Introduction

In *practice*, the optimization of engineered systems (man-made artifacts) is important, as is emphasized by Oden (2006)'s "National Science Foundation (NSF) Blue Ribbon Panel" report on simulation-based engineering. That report also emphasizes the crucial role of *uncertainty* in the input data for simulation models; we find that this uncertainty implies that robust optimization is important.

In *academic research*, the importance of optimization is demonstrated by the many sessions on this topic at the yearly Winter Simulation Conferences on discrete-event simulation; see
http://www.wintersim.org/.

The simplest type of optimization problems has no constraints for the input or the output, has no uncertain inputs, and concerns the expected value of a single (univariate) output; see the many test functions in Regis (2014). Obviously, in deterministic simulation the expected value is identical to the observed output of the simulation model for a given input combination. In random simulation, the expected value may also represent the probability of a binary variable having the value one, so $P(w = 1) = p$ and $P(w = 0) = 1 - p$ so $E(w) = p$. The expected value, however, excludes quantiles (e.g., the median and the 95 % quantile or percentile) and the mode of the output distribution. Furthermore, the simplest type of optimization assumes that the inputs are continuous (not discrete or nominal; see the various scales discussed in Sect. 1.3). The assumption of continuous inputs implies that there is an infinite number of systems, so we cannot apply so-called *ranking and selection* (R&S) and *multiple comparison* procedures (there are many publications on these procedures; see the next paragraph). We also refer to
http://simopt.org/index.php,
which is a testbed of optimization problems in discrete-event simulation. There are so many optimization methods that we do not try to summarize these methods. Neither do we refer to references that do summarize these methods—except for some very recent comprehensive references on simulation optimization that we list in the following note.

Note: Ajdari and Mahlooji (2014), Alrabghi and Tiwari (2015), Chau et al. (2014), Dellino and Meloni (2015), Figueira and Almada-Lobo (2014), Fu et al. (2014), Gosavi (2015), Homem-de-Mello and Bayraksan (2014), Hong et al. (2015), Jalali and Van Nieuwenhuyse (2015), Lee et al. (2013), Lee and Nelson (2014), Qu et al. (2015), Pasupathy and Ghosh (2014), Tenne and Goh (2010), Van der Herten et al. (2015) with its 800 pages, Xu et al. (2015) and Zhou et al. (2014).

In this chapter we focus on optimization that uses metamodels (approximations, emulators, surrogates); metamodels were introduced in Sect. 1.2. Moreover, we focus on metamodels that use either linear regression or Kriging; these two types of metamodels are detailed in the preceding four chapters. 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 theoretical publications. Like we did in the preceding chapters, we consider both deterministic and random simulation models in the present chapter. We define random simulation (including discrete event simulation) as simulation that uses pseudorandom numbers (PRN).

Note: Outside the discrete-event simulation area, some authors speak of RSM but they mean what we call the what-if regression-metamodeling approach, not the sequential (iterative) optimization approach. Other authors speak of RSM, but use global Kriging instead of local low-order polynomials. Many authors use the term "response surface" instead of "metamodel"; an example is Rikards and Auzins (2002).

Like in the preceding chapters, we focus on *expensive* simulation, in which it takes relatively much computer time for a single simulation run (such a run is a single realization of the time path of the simulated system). For example, 36 to 160 h of computer time were needed to simulate a crash model at Ford Motor Company; see the panel discussion reported in Simpson et al. (2004). This panel also reports the example of a (so-called "cooling") problem with 12 inputs, 10 constraints, and 1 objective function. For such expensive simulations, many simulation optimization methods are unpractical. An example is the popular software called *OptQuest* (which combines so-called tabu search, neural networks, and scatter search; it is an add-on to discrete-event simulation software such as Arena, CrystallBall, MicroSaint, ProModel, and Simul8); see

`http://www.opttek.com/OptQuest`.

OptQuest requires relatively many simulation replications and input combinations; see the inventory example in Kleijnen and Wan (2007). Fortunately, the mathematical and statistical computations required by optimization based on RSM or Kriging are negligible—compared with the computer time required by the "expensive" simulation runs.

In many OR applications, a single simulation run is computationally inexpensive, but there are extremely many input combinations; e.g., an M/M/1 model may have one input—namely, the traffic rate—that is continuous, so we can distinguish infinitely many input values but we can simulate only a fraction of these values in finite time. Actually, most simulation models have multiple inputs (say) k, so there is the "curse of dimensionality"; e.g., if we have $k = 7$ inputs (also see Miller 1956) and we experiment with only 10 values per input, then we still have 10^7 (10 million) combinations. Moreover, a single run may be expensive if we wish to estimate the steady-state performance of a queueing system with a high

traffic rate; e.g. we might need to simulate one million customers. Finally, if we wish to estimate the failure probability of a *highly reliable* system, then we need to simulate extremely many customers—unless we apply importance sampling.

Note: This chapter is based on Kleijnen (2014).

6.2 Linear Regression for Optimization

Linear regression models are used in RSM. We shall discuss RSM in several subsections; namely Sect. 6.2.1 on basic RSM, Sect. 6.2.2 on RSM in random simulation, Sect. 6.2.3 on adapted steepest descent (ASD), Sect. 6.2.4 on generalized RSM (GRSM) for multiple responses, and Sect. 6.2.5 on testing the KKT conditions of an optimum estimated through GRSM. We shall return to RSM in the section on robust optimization; see especially Sect. 6.4.1.

6.2.1 Response Surface Methodology (RSM): Basics

Originally, RSM was developed for the optimization of *real* (physical) systems.

Note: The classic article is Box and Wilson (1951). The origin of RSM is nicely discussed in Box (1999), an overview of RSM publications during the period 1966–1988 is Myers et al. (1989) and a recent overview is Khuri and Mukhopadhyay (2010), a popular handbook is Myers et al. (2009), and recent RSM software can be found on the Web; e.g., the Design-Expert software and Minitab's "Response Optimizer" are found on

`www.statease.com`
`http://www.minitab.com/`.

RSM in *simulation* was first detailed in the monograph Kleijnen (1975). Unfortunately, RSM (unlike search heuristics such as OptQuest) has not yet been implemented as an add-on to *commercial off the shelf* (COTS) simulation software.

Note: One of the first case-studies on RSM in random simulation is Van den Bogaard and Kleijnen (1977), reporting on a computer center with two servers and three priority classes—with small, medium, and large jobs—estimating the 90 % quantiles of the waiting times per class for different class limits, and applying RSM to find the optimal class limits. RSM in random simulation is also discussed in Alaeddini et al. (2013), Barton and Meckesheimer (2006), Huerta and Elizondo (2014), Law (2015), and Rosen et al. (2008). Google gave more than two million results for the term "Response Surface Methodology", on 4 February 2014.

RSM treats the real system or its simulation model—either a deterministic or a random model—as a *black box*; i.e., RSM observes the input/output

(I/O) of the simulation model—but not the internal variables and specific functions implied by the simulation's computer modules. RSM is a *sequential* heuristic; i.e., it uses a sequence of local experiments that is meant to lead to the optimum input combination. Note that an input combination is also called a point or a scenario. RSM uses design of experiments (DOE) and the concomitant linear regression analysis. Though RSM is only a heuristic, it has gained a good track record, as we shall see in the next subsections.

Regarding this track record, we add that practitioners may not be interested in convergence proofs, because realistic experiments may be so expensive that large sample sizes are impossible; e.g., the computer budget may be so limited that only a small sample is possible (see the literature on *optimal computer budget allocation* or OCBA). Practitioners may be more interested in finding better solutions than the current one. Actually, we may claim that "the best is the enemy of the better" (this claim is inspired by Voltaire's expression "le mieux est l'ennemi du bien" or "perfect is the enemy of good"). Herbert Simon (1956) claims that humans strive for a "satisficing" solution instead of the optimal solution. Samuelson (2010) also emphasizes that it may be impractical to search for the very best. Furthermore, the website

http://simopt.org/index.php

states "We are particularly interested in increasing attention on the finite time performance of algorithms, rather than the asymptotic results that one often finds in related literature". Finally, we quote an anonymous source: "Unfortunately, these theoretical convergence results mean little in practice where it is more important to find high quality solutions within a reasonable length of time than to guarantee convergence to the optimum in an infinite number of steps."

We assume that RSM is applied, only after the important inputs and their experimental area have been identified; i.e., before RSM starts, we may need to use *screening* to identify the really important inputs among the many conceivably important inputs. Case studies illustrating screening followed by RSM are Morales-Enciso and Branke (2015) and Shi et al. (2014). In Chap. 4 we detailed various screening methods, focusing on sequential bifurcation. Chang et al. (2014) combines RSM with screening in a single method. We point out that RSM without a preceding screening phase may imply the simulation of extremely many combinations of simulation inputs, as we shall see in this section.

RSM starts with a sequence of local *metamodels* that are first-order polynomials in the inputs. Once the optimum seems close, RSM augments the latest first-order polynomial to a second-order polynomial. Basic RSM tries to minimize the expected value of a single output, with continuous inputs and without any constraints:

$$\min E(w_0|\mathbf{z}) \tag{6.1}$$

where $E(w_0|\mathbf{z})$ is the goal or objective output (in Sect. 6.2.4 we shall discuss multiple outputs w_h with $h = 0, 1, \ldots, r$), which is to be minimized through the choice of the input combinations $\mathbf{z} = (z_1, \ldots, z_k)'$ where z_j $(j = 1, \ldots k)$ denotes the j^{th} "original" input; i.e., the inputs are not standardized such that they lie between -1 and 1 (sometimes, the inputs are standardized such they lie between 0 and 1). Obviously, if we wish to maximize (instead of minimize) the output $E(w_0)$, then we simply add a minus sign in front of the output in Eq. (6.1) before we minimize it. If the output is deterministic, then $E(w_0) = w_0$.

Note: In random simulation, we may write $E(w_0|\mathbf{z})$ in Eq. (6.1) as

$$E(w_0|\mathbf{z}) = \int_0^1 \cdots \int_0^1 f_{\text{sim}}(\mathbf{z}, \mathbf{r}) d\mathbf{r}$$

where $f_{\text{sim}}(\mathbf{z}, \mathbf{r})$ denotes the computer simulation program, which is a mathematical function that maps the inputs $\mathbf{z}$ and the PRN vector $\mathbf{r}$ (with elements r that have a uniform marginal distribution on $(0, 1)$) to the random simulation response (output) w_0.

RSM has the following *characteristic*s, which we shall detail below.

- RSM is an *optimization heuristic* that tries to estimate the input combination that minimizes a given goal function; see again Eq. (6.1). Because RSM is only a heuristic, it does not guarantee success.
- RSM is a *stepwise* (multi-stage) method; see the steps below.
- In each step, RSM fits a local first-order *polynomial* regression (meta) model—except for the last step, in which RSM fits a second-order polynomial.
- To fit (estimate, calibrate) these first-order polynomials, RSM uses I/O data obtained through so-called *resolution-III (R-III) designs*; for the second-order polynomial, RSM uses a *central composite design (CCD)*; we have already detailed these R-III designs and CCDs in Chap. 2.
- Each step—except the last one—selects the direction for changing the inputs through the *gradient* implied by the first-order polynomial fitted in that step. This gradient is used in the mathematical (not statistical) technique of *steepest descent*—or steepest ascent, in case the output is to be maximized.
- In the final step, RSM takes the *derivatives* of the locally fitted *second-order polynomial* to estimate the optimum input combination. RSM may also apply the mathematical technique of *canonical analysis* to this polynomial, to examine the shape of the optimal subregion; i.e., does that region have a unique minimum, a saddle point, or a ridge with stationary points?

More specifically, the RSM algorithm (for either real or simulated systems) consists of the following *steps* (also see Fig. 6.1 in Sect. 6.2.4, which gives an example with a random goal output w_0 and two constrained random outputs w_1 and w_2; these constrained outputs vanish in basic RSM).

Algorithm 6.1

1. Initialize RSM; i.e., select a *starting point.*
   Comment: This starting point may be the input combination that is currently used in practice if the system already exists; otherwise, we should use intuition and prior knowledge (as in many other heuristics).

2. In the *neighborhood* of this starting point, approximate the I/O behavior through a local first-order polynomial metamodel augmented with additive white noise e:

$$y = \beta_0 + \sum_{j=1}^{k} \beta_j z_j + e \tag{6.2}$$

   with the regression parameters $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_k)'$ where β_0 denotes the intercept and β_j denotes the first-order or "main" effect of input j with $j = 1, \ldots, k$.
   Comment: The first-order polynomial approximation may be explained by Taylor's series expansion. *White noise* (see Definition 2.3 in Chap. 2) means that e is normally, independently, and identically distributed (NIID) with zero mean and a constant variance (say) σ^2 in the local experimental area: $e \sim \text{NIID}(0, \sigma^2)$. However, when the next step moves to a new local area, RSM allows the variance to change.

   Compute the *best linear unbiased estimator* (BLUE) of $\boldsymbol{\beta}$; namely, the *least squares* (LS) estimator

$$\widehat{\boldsymbol{\beta}} = (\mathbf{Z}'\mathbf{Z})^{-1}\mathbf{Z}'\mathbf{w} \tag{6.3}$$

   where $\mathbf{Z}$ denotes the $N \times (k+1)$ matrix determined by the R-III design and the m_i replications of combination i ($i = 1, \ldots, n$) with $n \geq k+1$ and $\mathbf{w} = (w_1, \ldots w_N)'$ denotes the vector with the N outputs with $N = \sum_{i=1}^{n} m_i$ where $m_i \geq 1$ denotes the number of replications of combination i.
   Comment: $\mathbf{Z}$ has m_i identical rows where each row has as first element the value 1 which corresponds with the intercept β_0. Obviously, deterministic simulation implies $m_i = 1$ so $N = n$. Unfortunately, there are no general guidelines for determining the appropriate *size* of the local area in a step of RSM; again, intuition and prior knowledge are important. However, Chang et al. (2013) decides on the size of the local area, using a so-called trust region; we shall give some details

in Sect. 6.2.2. Furthermore, so-called "finite differencing" replaces the R-III design by a less efficient one-factor-at-a-time design (see again Sect. 2.3.2) and also faces the problem of selecting an appropriate size for the local area; the optimal size depends on the unknown variance and second-order derivatives; see Brekelmans et al. (2005), Safizadeh (2002), Saltelli et al. (2005), and Zazanis and Suri (1993).

3. Select the next subarea, following the *steepest descent* direction.
   Comment: For example, if the estimated local first-order polynomial is $\widehat{y} = \widehat{\beta}_0 + \widehat{\beta}_1 z_1 + \widehat{\beta}_2 z_2$, then a corresponding contour line is $\widehat{y} = a$ where a denotes some constant (if the goal output w_0 denotes costs, then the contour is also called the iso-costs line). The steepest descent path is *perpendicular* to the local contour lines. This path implies that if $\widehat{\beta}_1 \gg \widehat{\beta}_2$, then z_1 is decreased much more than z_2. Unfortunately, the steepest-descent method is *scale dependent*; i.e., linear transformations of the inputs affect the search direction; see Myers et al. (2009, pp. 193–195). We shall present a scale-independent variant in Sect. 6.2.3, which may interest both practitioners and researchers.

4. Take a step in the direction of steepest descent (estimated in step 3), experimenting with some intuitively selected values for the step size.
   Comment: If the intuitively selected step size yields an output that is significantly higher instead of lower, then we reduce the step size. Otherwise, we take one more step in the current steepest descent direction. A more sophisticated mathematical procedure for selecting the step size will follow in Sect. 6.2.4.

5. If the observed output w increases, then generate n outputs for a new local area centered around the best point found so far.
   Comment: After a number of steps in the steepest descent direction, the output will increase instead of decrease because the first-order polynomial in Eq. (6.2) is only a local approximation of the true I/O function. When such deterioration occurs, we simulate the $n > k$ combinations specified by a R-III design centered around the best point found so far; i.e., we use the same design as in step 2 (see Table 2.3 for an example), but we translate the standardized inputs x_j into different values for the original inputs z_j. One of the corner points of this R-III design may be the best combination found so far; see again Fig. 6.1 below.

6. Estimate the first-order effects in the new local polynomial approximation, using Eq. (6.3).

7. Return to step 3, if the latest locally fitted first-order polynomial is found to be adequate; else proceed to the next step.
   Comment: To test the *adequacy* of the fitted first-order polynomial,

we may apply one or more methods that we have already discussed for estimated linear regression metamodels in general; namely, the lack-of-fit F-statistic for testing whether all estimated first-order effects and hence the gradient are zero (see Sect. 2.2.2), and the coefficient of determination R^2 and cross-validation (see Sect. 3.6).

8. Fit the *second-order polynomial*

$$y = \beta_0 + \sum_{j=1}^{k} \beta_j z_j + \sum_{j=1}^{k} \sum_{j' \geq k}^{k} \beta_{j;j'} z_j z_{j'} + e, \tag{6.4}$$

where β_0 denotes the intercept, β_j $(j = 1, \ldots, k)$ the first-order effect of input j, $\beta_{j;j}$ the purely quadratic effect of input j, and $\beta_{j;j'}$ $(j < j')$ the interaction between inputs j and j'; estimate these $q = 1+2k+k(k-1)/2$ effects through a CCD with $n \geq q$ combinations
Comment: It is intuitively clear that the *plane* implied by the most recently estimated local first-order polynomial cannot adequately represent a *hill top* when searching to maximize the output or—equivalently—minimize the output as in Eq. (6.1). So in the neighborhood of the optimum, a first-order polynomial is not adequate. We therefore fit the second-order polynomial defined in Eq. (6.4); RSM uses a CCD to generate the I/O data.

9. Use this fitted second-order polynomial, to estimate the optimal values of the inputs by straightforward *differentiation* or by more sophisticated *canonical analysis*; see Myers et al. (2009, pp. 224–242).

10. If time permits, then try to escape from a possible local minimum and *restart* the search; i.e., return to step 1 with a different initial local area.

    Comment: We shall discuss a *global* search method (namely, efficient global optimization, EGO) in Sect. 6.3.1.

We recommend not to eliminate inputs that have *nonsignificant* effects in a first-order polynomial fitted within the current local experimental area: these inputs may have significant effects in a next experimental area. The selection of the *number of replications* m_i is a moot issue in metamodeling, as we have already discussed for experimental designs in case of linear regression with heterogeneous variances (see Sect. 3.4.5) and for the selection of the number of replications through the sequential probability ratio test (SPRT) for sequential bifurcation (see Sect. 4.5), and for Kriging (see Sect. 5.6.2). For the time being, we recommend estimating the true mean response for a given input combination such that a relative precision of (say) 10 % has a (say) 90 % probability, using the method detailed in Law (2015).

The Taylor series argument suggests that a higher-order polynomial is more accurate than a lower-order polynomial. A statistical counterargument, however, is that *overfitting* gives less accurate estimators of the polynomial coefficients. Consequently, the higher-order polynomial may give a predictor $\widehat{y}$ with lower bias but higher variance such that its mean squared error (MSE) increases. Moreover, a higher-order polynomial requires the simulation of more input combinations.

In Sect. 3.4 we have already mentioned that a *deterministic simulation* model gives a fixed value for a given input combination, so we might assume white noise for the residuals e of the metamodel and apply basic RSM. In *random simulation*, however, we prefer the RSM variant detailed in the next section.

6.2.2 RSM in Random Simulation

We consider the following two characteristics of random simulation that violate the assumption of *white noise* within a given local area:

1. The constant variance assumption does not hold.
2. The independence assumption does not hold if common random numbers (CRN) are applied.

Sub 1: Many simulation models represent queueing systems; e.g., supply chains and telecommunication networks. The simplest queueing model is the so-called M/M/1 model (see Definition 1.4) for which we know that as its traffic rate increases, its mean steady-state waiting time increases and the variance increases even more; consequently, the assumption of a constant variance does not hold.

Sub 2: CRN are often applied in experiments with random simulation models, because CRN are the default option in many simulation software packages (e.g., Arena); moreover, CRN are a simple and intuitive variance reduction technique that gives more accurate estimators of the first-order or second-order polynomial metamodel in Eqs. (6.2) and (6.4). Obviously, the outputs of all input combinations that use CRN are statistically dependent; actually, we expect these outputs to be positively correlated.

Note: CRN are related to *blocking* in real-life experiments. In simulation experiments, we may use blocking when combining CRN and antithetic random numbers through the so-called Schruben-Margolin strategy; this strategy is recently discussed in Chih (2013).

Sub 1 and 2: The preceding two characteristics imply that ordinary LS (OLS) does not give the BLUE. As we have already discussed in Sect. 3.5, generalized LS (GLS) gives the BLUE, but assumes known response variances and covariances. We therefore recommend the following simple estimator, which we have already detailed in Sect. 3.5.

We assume a constant number of replications $m_i = m$ $(i = 1, \ldots, n)$, which is a realistic assumption if CRN are applied. We then compute the OLS estimator per replication replacing $\mathbf{w}$ in Eq. (6.3) by $\mathbf{w}_r$ to get the estimator $\widehat{\boldsymbol{\beta}}_r$ $(r = 1, \ldots, m)$. 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 the estimated accuracy to be 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 the number of replications.

Actually, if we have $m_i > 1$ $(i = 1, \ldots, n)$ replications, then we can further explore the statistical properties of the OLS estimator of $\boldsymbol{\beta}$ through *distribution-free bootstrapping*, as we have already discussed in Sect. 3.3.5. We can also use the bootstrapped estimator $\widehat{\boldsymbol{\beta}}^*$ to derive confidence intervals (CIs) for the corresponding estimated steepest ascent direction and optimum.

Instead of distribution-free bootstrapping we can apply *parametric bootstrapping*, which assumes a specific type of distribution; e.g., a Gaussian distribution (also see the testing of the KKT conditions in Sect. 6.2.5 below). Parametric bootstrapping may be attractive if m_i is small and no CRN are used; e.g., the n expected values $E(w_i)$ and n variances σ_i^2 can be estimated if the weak condition $m_i > 1$ holds. If CRN are used, then the $n \times n$ covariance matrix $\boldsymbol{\Sigma}_{\mathbf{w}} = (\text{cov}(w_i, w_{i'}))$ with $i, i' = 1, \ldots, n$ needs to be estimated; this estimation requires $m > n$, as proven in Dykstra (1970). So parametric bootstrapping may require fewer replications, but the assumed distribution may not hold for the simulated outputs.

Chang et al. (2013) presents *the stochastic trust-region response-surface method* (STRONG), which is a completely automated variant of RSM combined with so-called trust regions. STRONG is proven to converge to the true optimum (but see again our discussion of convergence, in Sect. 6.2.1). Originally, trust regions were developed in Conn et al. (2000) for deterministic nonlinear optimization. By definition, a *trust region* is a subregion in which the objective function is approximated such that if an adequate approximation is found within the trust region, then the region is expanded; else the region is contracted. STRONG uses these trust regions instead of the "local" regions of basic RSM, detailed in the preceding section. STRONG includes statistical tests to decide whether trust regions should be expanded or shrunken in the various steps, and to decide how much these areas should change. If necessary, the trust region is small and a second-order polynomial is used. Next, Chang et al. (2014) combines STRONG with *screening*, and calls the resulting procedure STRONG-S where S denotes screening. This method is applied to several test functions with multiple local minima. Contrary to the Taylor-series argument, STRONG may

have a relatively large trust region that does not require a second-order polynomial metamodel but only a first-order polynomial metamodel. Chang and Lin (2015) applies STRONG—including some adaptation—to a renewable energy system. RSM in random simulation is also discussed in Law (2015, pp. 656–679). Ye and You (2015) uses trust regions, not applied to low-order polynomial metamodels but to deterministic Kriging metamodels of the underlying random simulation model.

Note: I/O data in RSM may contain outliers, which should be detected; for this detection, Huang and Hsieh (2014) presents so-called *influence analysis.*

Exercise 6.1 *Apply RSM to the following problem that is a simple Monte Carlo model of a random simulation:*

$$min\ E[5(z_1 - 1)^2 + (z_2 - 5)^2 + 4z_1z_2 + e]$$

where $\mathbf{z} = (z_1, z_2)'$ *and* $e \sim NIID(0, 1)$. *RSM treats this example as a black box; i.e., you select the input combination* $\mathbf{z}$, *sample* e *from* $NIID(0, 1)$, *and use these input data to compute the output (say)* w. *You (not RSM) may use the explicit function to derive the true optimum solution,* $\mathbf{z}_o$.

6.2.3 Adapted Steepest Descent (ASD) for RSM

Kleijnen et al. (2004) derives the so-called *adapted steepest descent* (ASD) direction that accounts for the covariances between the k components of the estimated gradient $\widehat{\boldsymbol{\beta}}_{-0} = (\widehat{\beta}_1, \ldots, \widehat{\beta}_k)'$ where the subscript -0 means that the intercept $\widehat{\beta}_0$ of the estimated first-order polynomial vanishes in the estimated gradient; i.e., $\widehat{\boldsymbol{\beta}} = (\widehat{\beta}_0, \widehat{\boldsymbol{\beta}}_{-0})'$ with $\widehat{\boldsymbol{\beta}}$ defined in Eq. (6.3). Obviously, *white noise* implies

$$\boldsymbol{\Sigma}_{\widehat{\boldsymbol{\beta}}} = \sigma_w^2(\mathbf{Z}'\mathbf{Z})^{-1} = \sigma_w^2 \begin{pmatrix} a & \mathbf{b}' \\ \mathbf{b} & \mathbf{C} \end{pmatrix} \tag{6.5}$$

where σ_w^2 denotes the variance of the output w; $\mathbf{Z}$ is the $N \times (1 + k)$ matrix of explanatory regression variables including the column with N one's; $N = \sum_{i=1}^{n} m_i$ where n is the number of different observed input combinations; m_i is the number of IID replications for combination i; a is a scalar; $\mathbf{b}$ is a k-dimensional vector; and $\mathbf{C}$ is a $k \times k$ matrix such that $\boldsymbol{\Sigma}_{\widehat{\boldsymbol{\beta}}_{-0}} = \sigma_w^2\mathbf{C}$.

We notice that $\mathbf{Z}$'s first column corresponds with the intercept β_0. Furthermore, $\mathbf{Z}$ is determined by the R-III design, transformed into the original values of the inputs in the local area. To save computer time, we may replicate only the center of the local area; this center is not part of the R-III design.

The variance σ_w^2 in Eq. (6.5) is estimated through the *mean squared residuals* (MSR):

$$\widehat{\sigma}_w^2 = \frac{\sum_{i=1}^{n} \sum_{r=1}^{m_i} (w_{i;r} - \widehat{y}_i)^2}{N - (k+1)} \tag{6.6}$$

where $\widehat{y}_i = \mathbf{z}_i' \widehat{\boldsymbol{\beta}}$; also see Eq. (2.26).

It can be proven that the predictor variance $\text{Var}(\widehat{y}|\mathbf{z})$ increases as $\mathbf{z}$—the point to be predicted—moves away from the local area where the gradient is estimated. The point with the minimum predictor variance is $-\mathbf{C}^{-1}\mathbf{b}$, where $\mathbf{C}$ and $\mathbf{b}$ were defined below Eq. (6.5). ASD means that the new point to be simulated is

$$\mathbf{d} = -\mathbf{C}^{-1}\mathbf{b} - \lambda \mathbf{C}^{-1} \widehat{\boldsymbol{\beta}}_{-0} \tag{6.7}$$

where $-\mathbf{C}^{-1}\mathbf{b}$ is the point where the local search starts (namely, the point with the minimum local variance), λ is the step size, $\widehat{\boldsymbol{\beta}}_{-0}$ is the steepest descent direction, and $\mathbf{C}^{-1}\widehat{\boldsymbol{\beta}}_{-0}$ is the steepest descent direction adapted for $\boldsymbol{\Sigma}_{\widehat{\boldsymbol{\beta}}_{-0}}$. It is easy to see that if $\mathbf{C}$ is a diagonal matrix, then the higher the variance of an estimated input effect is, the less the search moves into the direction of that input.

Exercise 6.2 *Prove that the search direction in Eq. (6.7) does not change the steepest descent direction if the design matrix is orthogonal (so* $\mathbf{Z}'\mathbf{Z} = N\mathbf{I}$*).*

It can be proven that ASD, which accounts for $\boldsymbol{\Sigma}_{\widehat{\boldsymbol{\beta}}_{-0}}$, gives a *scale-independent* search direction. Experimental results are presented in Kleijnen et al. (2004, 2006). These results imply that ASD performs "better" than steepest descent; i.e., the angle between the search direction based on the true $\boldsymbol{\beta}_{-0}$ and the search direction estimated in ASD is smaller. In one example this angle reduces from 89.87 for steepest descent to 1.83 for ASD.

Note: Fan and Huang (2011) derives another alternative for steepest ascent, using *conjugate gradients* (which were originally developed for unconstrained optimization in mathematical programming). Joshi et al. (1998) derives one more alternative, using *gradient deflection* methods. Safizadeh (2002) examines how to balance the variance and the bias via the MSE of the estimated gradient for different sizes of the local experimental area, assuming random simulation with CRN.

6.2.4 Multiple Responses: Generalized RSM (GRSM)

In practice, simulation models have *multiple* responses types (multivariate output); e.g., a realistic inventory simulation model may estimate (i) the sum of all inventory costs excluding the (hard-to-quantify) out-of-stock costs and (ii) the service rate (fill rate), and the goal of this simulation is to minimize this sum of inventory costs such that the service rate is not lower than (say) 90 %. Simulation software facilitates the collection of multiple outputs. There are several approaches to solve the resulting issues;

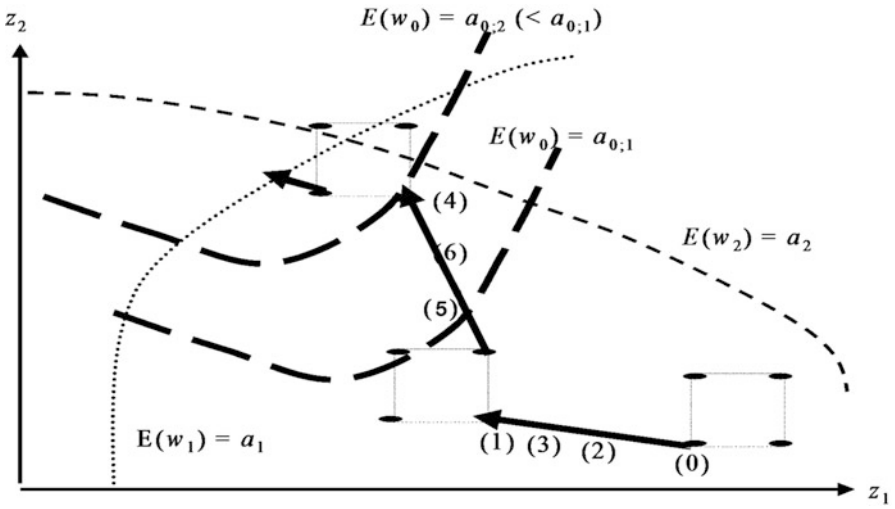

FIGURE 6.1. GRSM example with two inputs, two contour plots for the goal output, two constraints for the other outputs, three local areas, three search directions, and six steps in these directions

see the survey in Rosen et al. (2008). The RSM literature also offers several approaches for such situations, but we shall focus on GRSM.

Note: For RSM with multiple responses we refer to the surveys in Angün (2004), Khuri and Mukhopadhyay (2010), and Ng et al. (2007) and the recent case study in Shi et al. (2014) combining two output types into a single criterion. We shall discuss Kriging for simulation with multiple outputs, in Sect. 6.3.

GRSM is explained in Angün et al. (2009). Informally, we may say that GRSM is RSM for problems with multiple random outputs such that one goal output is minimized while the other outputs satisfy prespecified constraints (so GRSM does not use multi-objective optimization); moreover, the deterministic input variables may also be subjected to constraints. GRSM combines RSM and mathematical programming; i.e., GRSM generalizes the steepest descent direction of RSM through the *affine scaling search direction*, borrowing ideas from *interior point* methods (a variation on Karmarkar's algorithm) as explained in Barnes (1986). As Fig. 6.1 illustrates, the GRSM search 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 than steepest descent. Moreover, this search tries to stay inside the feasible area, so the simulation program does not crash. We shall discuss Fig. 6.1 in detail, at the end of this subsection. We point out that Angün et al. (2009) proves that the GRSM search direction is scale independent. Though we focus on *random* simulations, we might easily adapt GRSM for *deterministic* simulations and *real* systems.

Because GRSM is rather complicated, readers may wish to skip the rest of this subsection and also skip the next subsection (Sect. 6.2.5)—on testing an estimated optimum in GRSM through testing the Karush-Kuhn-Tucker conditions—without lessening their understanding of the rest of this book.

Formally, GRSM extends the basic RSM problem in Eq. (6.1) to the following *constrained nonlinear random optimization problem*:

$$\min E(w_0|\mathbf{z}) \tag{6.8}$$

such that the other $(r-1)$ random outputs satisfy the constraints

$$E(w_{h'}|\mathbf{z}) \geq a_{h'} \quad \text{with } h' = 1, \ldots, r-1, \tag{6.9}$$

and the k deterministic inputs z_j satisfy the *box constraints*

$$l_j \leq z_j \leq u_j \quad \text{with } j = 1, \ldots, k. \tag{6.10}$$

An example is an inventory simulation, in which the sum of the expected inventory carrying costs and ordering costs should be minimized while the expected service percentage should be at least 90 % so $a_1 = 0.9$ in Eq. (6.9); both the reorder quantity $z_1 = Q$ and the reorder level $z_2 = s$ should be non-negative so $z_1 \geq 0$ and $z_2 \geq 0$ in Eq. (6.10). A stricter input constraint may be that z_2 should at least cover the expected demand during the expected order lead time; obviously, these expectations are known inputs of the simulation. More complicated input constraints than Eq. (6.10)—namely, linear budget constraints—feature in a call-center simulation in Kelton et al. (2007).

Note: Optimization of simulated call-centers—but not using GRSM—is also studied in Atlason et al. (2008). Aleatory and epistemic uncertainties—discussed in Sect. 5.9 on risk analysis—in call-center queueing models are studied in Bassamboo et al. (2010). Geometry constraints are discussed in Stinstra and Den Hertog (2008). Input constraints resulting from output constraints are discussed in Ng et al. (2007).

Analogously to RSM's first steps using Eq. (6.2), GRSM locally approximates the multivariate I/O function by r univariate first-order polynomials augmented with white noise:

$$\mathbf{y}_h = \mathbf{Z}\boldsymbol{\beta}_h + e_h \quad \text{with } h = 0, \ldots r-1. \tag{6.11}$$

Analogously to RSM, GRSM assumes that locally the white noise assumption holds for Eq. (6.11), so the BLUEs are the following OLS estimators:

$$\widehat{\boldsymbol{\beta}}_h = (\mathbf{Z}'\mathbf{Z})^{-1}\mathbf{Z}'\mathbf{w}_h \quad \text{with } h = 0, \ldots r-1. \tag{6.12}$$

The vector $\widehat{\boldsymbol{\beta}}_0$ (OLS estimator of first-order polynomial approximation of goal function) and the goal function in Eq. (6.8) result in

$$\min \widehat{\boldsymbol{\beta}}_{0;-0}\mathbf{z} \tag{6.13}$$

where $\widehat{\boldsymbol{\beta}}_{0;-0} = (\widehat{\beta}_{0;1}, \ldots, \widehat{\beta}_{0,k})'$ is the OLS estimator of the local gradient of the goal function. Combining Eq. (6.12) and the original output constraints in Eq. (6.9) gives

$$\widehat{\boldsymbol{\beta}}'_{h';-0}\mathbf{z} \geq c_{h'} \quad \text{with } h' = 1, \ldots, r-1 \tag{6.14}$$

where $\widehat{\boldsymbol{\beta}}_{h';-0} = (\widehat{\beta}_{h';1}, \ldots, \widehat{\beta}_{h',k})'$ is the estimator of the local gradient of constraint function h', and $c_{h'} = a_{h'} - \widehat{\beta}_{h';0}$ is the modified right-hand side of this constraint function. The box constraints in Eq. (6.10) remain unchanged.

Now we collect the k-dimensional vectors $\widehat{\boldsymbol{\beta}}_{h';-0}$ $(h' = 1, \ldots, r-1)$ in Eq. (6.14) in the $(r-1) \times k$ matrix denoted by (say) $\mathbf{B}$. Likewise, we collect the $(r-1)$ elements $c_{h'}$ in the vector $\mathbf{c}$. Furthermore, we define $\mathbf{l}$ as the vector with the k elements l_j, and $\mathbf{u}$ as the vector with the k elements u_j. Finally, we introduce the k-dimensional vectors with the non-negative *slack variables* $\mathbf{s}$, $\mathbf{r}$, and $\mathbf{v}$, to get the following problem formulation that is the equivalent of the problem formulated in Eq. (6.8) through Eq. (6.10):

$$\begin{array}{ll} \text{minimize} & \widehat{\boldsymbol{\beta}}'_{0;-0}\mathbf{z} \\ \text{subject to} & \mathbf{B}\mathbf{z} - \mathbf{s} = \mathbf{c} \\ & \mathbf{z} + \mathbf{r} = \mathbf{u} \\ & \mathbf{z} - \mathbf{v} = \mathbf{l}. \end{array} \tag{6.15}$$

Obviously, the constrained optimization problem in Eq. (6.15) is linear in the inputs $\mathbf{z}$ (the OLS estimates $\widehat{\boldsymbol{\beta}}_{0;-0}$ and $\widehat{\boldsymbol{\beta}}_{h';-0}$ in $\mathbf{B}$ use the property that this problem is also linear in the regression parameters). Angün et al. (2009) uses this problem formulation to derive the following *GRSM search direction*:

$$\mathbf{d} = -(\mathbf{B}'\mathbf{S}^{-2}\mathbf{B} + \mathbf{R}^{-2} + \mathbf{V}^{-2})^{-1}\widehat{\boldsymbol{\beta}}_{0;-0} \tag{6.16}$$

where $\mathbf{S}$, $\mathbf{R}$, and $\mathbf{V}$ are diagonal matrixes with as main-diagonal elements the current estimated slack vectors $\mathbf{s}$, $\mathbf{r}$, and $\mathbf{v}$ in Eq. (6.15). Note that $\widehat{\boldsymbol{\beta}}_{0;-0}$ in Eq. (6.16) is the estimated steepest ascent direction in basic RSM. As the value of a slack variable in Eq. (6.16) decreases—so the corresponding constraint gets tighter—the GRSM search direction deviates more from the steepest descent direction. Possible singularity of the various matrices in Eq. (6.16) is discussed in Angün (2004).

Following the GRSM direction defined by Eq. (6.16), we must decide on the *step size* (say) λ along this path. Angün et al. (2009) selects

$$\lambda = 0.8 \min \left[\frac{c_{h'} - \widehat{\boldsymbol{\beta}}'_{h';-0}\mathbf{z}_c}{\widehat{\boldsymbol{\beta}}'_{h';-0}\mathbf{d}} \right] \tag{6.17}$$

where the factor 0.8 decreases the probability that the *local* metamodel in Eq. (6.14) is misleading when applied *globally*; $\mathbf{z}_c$ denotes the current (see the subscript c) input combination.

Combining the search direction in Eq. (6.16) and the step size in Eq. (6.17) gives the new combination $\mathbf{z}_c + \lambda\mathbf{d}$. The box constraints in Eq. (6.10) for the deterministic inputs hold globally, so it is easy to check whether this new combination $\mathbf{z}_c + \lambda\mathbf{d}$ satisfies these constraints.

Analogously to basic RSM, GRSM proceeds *stepwise*. After each step along the search path, GRSM tests the following two null-hypotheses $H_0^{(1)}$ and $H_0^{(2)}$:

1. Pessimistic null-hypothesis: $w_0(\mathbf{z}_c+\lambda\mathbf{d})$ (output of new combination) is *no improvement* over $w_0(\mathbf{z}_c)$ (output of old combination):

$$H_0^{(1)} : E[w_0(\mathbf{z}_c + \lambda\mathbf{d})] \geq E[w_0(\mathbf{z}_c)]. \tag{6.18}$$

2. Optimistic null-hypothesis: this step is *feasible*; i.e., $w_{h'}(\mathbf{z}_c + \lambda\mathbf{d})$ satisfies the $(r-1)$ constraints in Eq. (6.9):

$$H_0^{(2)} : E[w_{h'}(\mathbf{z}_c + \lambda\mathbf{d})] \geq a_{h'} \quad \text{with } h' = 1, \ldots, r-1. \tag{6.19}$$

To test these two hypotheses, we may apply the following simple statistical procedures; more complicated parametric bootstrapping is used in Angün (2004), permitting *non*normality and testing the *relative* improvement $w_0(\mathbf{z}_c + \lambda\mathbf{d})/w_0(\mathbf{z}_c)$ and slacks $s_{h'}(\mathbf{z}_c + \lambda\mathbf{d})/s_{h'}(\mathbf{z}_c)$.

Exercise 6.3 *Which statistical problem arises when testing the ratio of the slack at the new solution and the slack at the old solution,* $s_{h'}(\mathbf{z}_c + \lambda\mathbf{d})/s_{h'}(\mathbf{z}_c)$*?*

To test $H_0^{(1)}$ defined in Eq. (6.18), we apply the paired Student statistic t_{m-1}; we use the "paired" statistic because we assume that CRN are used. We reject the hypothesis if significant improvement is observed. To test $H_0^{(2)}$ in Eq. (6.19), we again apply a t_{m-1} -statistic; because we test multiple hypotheses, we apply Bonferroni's inequality so we divide the classic α value by $(r-1)$ (number of tests).

Actually, a better solution may lie somewhere between $\mathbf{z}_c$ (old combination) and $\mathbf{z}_c+\lambda\mathbf{d}$ (new combination). Therefore GRSM uses *binary search*; i.e., GRSM simulates a combination that lies halfway between these two combinations—and is still on the search path. This halving of the step size may be applied several times; also see Fig. 6.1.

Next, GRSM proceeds analogously to basic RSM; i.e., around the best combination found so far, GRSM selects a new local area. Again a R-III design specifies the new simulation input combinations, and r first-order polynomials are fitted, which gives a *new* search direction, etc. Note that we might use the m replications $\widehat{\boldsymbol{\beta}}_r$ to estimate the accuracy of the search direction; to test the accuracy of the estimated optimum, we shall present a test in the next subsection.

Now we discuss Fig. 6.1 in more detail. This plot illustrates GRSM for a problem with simple known test functions (in practice, we use simulation to estimate the true outputs of the various implicit I/O functions of the simulation model). This plot shows two inputs, corresponding to the two axes labeled z_1 and z_2. Because the goal function is to be minimized, the plot shows two *contour plots* or *iso-costs functions* defined by $E(w_0) = a_{0;1}$ and $E(w_0) = a_{0;2}$ with $a_{0;2} < a_{0;1}$. The plot also shows two constraints; namely, $E(w_1) = a_1$ and $E(w_2) = a_2$. The search starts in the lower-right local area of the plot, using a 2^2 design; see the four elongated points. Together with the replications that are not shown, the I/O data give the search direction that is shown by the arrow leaving from point (0). The maximum step-size along this path takes the search from point (0) to point (1). The binary search takes the search back to point (2), and next to point (3). Because the best point so far turns out to be point (1), the 2^2 design is again used to select four points in this new local area; point (1) is selected as one of these four points. Simulation of the four points of this 2^2 design gives a new search direction, which indeed avoids the boundary. The maximum step-size now takes the search to point (4). The binary search takes the search back to point (5), and next to point (6). Because the best point so far turns out to be point (4), the 2^2 design is simulated in a new local area with point (4) as one of its points. A new search direction is estimated, etc.

Angün (2004) gives details on two examples, illustrating and evaluating GRSM. One example is an inventory simulation with a service-level constraint specified in Bashyam and Fu (1998); no analytical solution is known. The other example is a test function with a known solution. The results for these examples are encouraging, as GRSM finds solutions that are both feasible and give low values for the goal functions. Leijen (2011) applies GRSM to a bottle-packaging line at Heineken with nine inputs and one stochastic output constraint besides several deterministic input constraints; the analysis of the solutions generated by GRSM indicates that GRSM can find good estimates of the optimum. Mahdavi et al. (2010) applies GRSM to a job-shop manufacturing system. We shall briefly return to GRSM when discussing Eq. (6.35).

Exercise 6.4 *Apply GRSM to the following artificial example reproduced from Angün et al. (2009):*

$$\begin{array}{ll} \textit{Minimize} & E[5(z_1-1)^2+(z_2-5)^2+4z_1z_2+e_0] \\ \textit{subject to} & E[(z_1-3)^2+z_2^2+z_1z_2+e_1] \le 4 \\ & E[z_1^2+3\,(z_2+1.061)^2+e_2] \le 9 \\ & 0 \le z_1 \le 3,\ -2 \le z_2 \le 1 \end{array} \tag{6.20}$$

where e_0, e_1, and e_2 are the components of a multivariate normal variate with mean $\mathbf{0}$*, variances $\sigma_{0;0} = 1$ (so $\sigma_0 = 1$), $\sigma_{1;1} = 0.0225$ (so $\sigma_1 = 0.15$), and $\sigma_{2;2} = 0.16$ (so $\sigma_2 = 0.4$), and correlations $\rho_{0;1} = 0.6$, $\rho_{0;2} = 0.3$, $\rho_{1;2} = -0.1$.*

6.2.5 Testing a GRSM Optimum: Karush-Kuhn-Tucker (KKT) conditions

Obviously, it is uncertain whether the optimum estimated by the GRSM heuristic is close enough to the true optimum. In *deterministic* nonlinear mathematical programming, the first-order necessary optimality-conditions are known as the KKT conditions; see Gill et al. (2000). First we present the basic idea behind these conditions; next, we explain how to test these conditions in random simulation.

To explain the basic idea of the KKT conditions, we use Fig. 6.2 that illustrates the same type of problem as the one in Fig. 6.1. Figure 6.2 shows a goal function $E(w_0)$ with three contour plots that correspond with the threshold values 66, 76, and 96; also see Eq. (6.8). Furthermore, there are two constrained simulation outputs; namely, $E(w_1) \geq 4$ and $E(w_2) \geq 9$; also see Eq. (6.9). So, the plot shows the boundaries of the feasible area that is determined by the equalities $E(w_1) = 4$ and $E(w_2) = 9$. Obviously, the optimum combination is point A. The two points B and C lie on the same boundary; namely, the boundary $E(w_2) = 9$. Point D lies on the other boundary; namely, the boundary $E(w_1) = 4$. Obviously, the optimal point A and the point D lie far away from each other. The plot also displays the local gradients at the four points A through D for the goal function and for the *binding constraint*, which is the constraint with a zero slack value in Eq. (6.9). These gradients are *perpendicular* to the local tangent lines; those lines are shown only for the binding constraint—not for the goal function. These tangent lines are first-order polynomials; see Eq. (6.11). (Obviously, the estimated gradient is biased if second-order effects are important and yet a first-order polynomial is fitted.)

Note: There is a certain constraint qualification that is relevant when there are nonlinear constraints in the problem; see Gill et al. (2000, p. 81). There are several types of constraint qualification, but many are only of theoretical interest; a practical constraint qualification for nonlinear constraints is that the $r - 1$ constraint gradients at the locally optimal combination be linearly independent.

Now we present the statistical procedure for testing the KKT conditions in random simulation that was derived in Bettonvil et al. (2009). Before we shall discuss the technical details of this procedure, we point out that the empirical results for this procedure are encouraging; i.e., the classic t-test for zero slacks performs as expected and the new bootstrap tests give observed type-I error rates close to the prespecified (nominal) rates, while the type-II error rate decreases as the tested input combination is farther away from the true optimum; see the points A through D in Fig. 6.2.

Note: We add that Kasaie et al. (2009) also applies this procedure to an agent-based simulation model of epidemics; this model is also discussed in Kasaie and Kelton (2013). Furthermore, Wan and Li (2008) applies the asymptotic variant of this procedure to the (s, S) inventory problem formulated in Bashyam and Fu (1998) with good results.

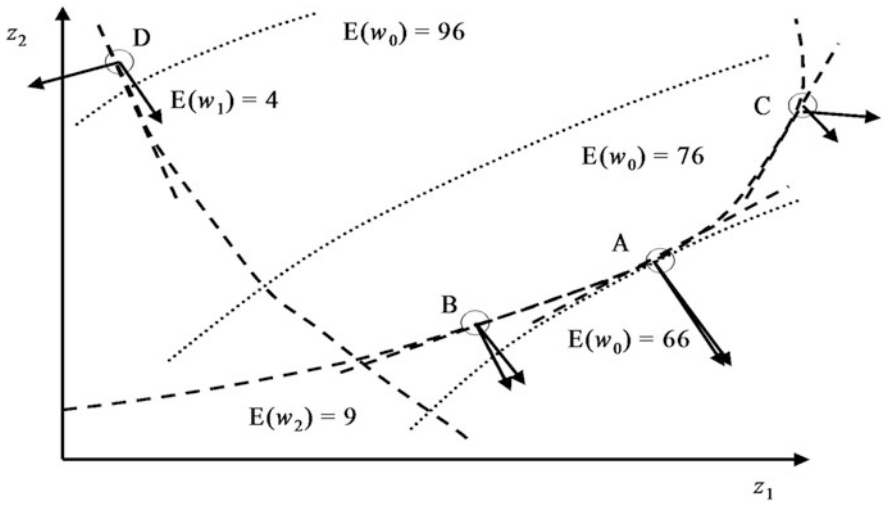

FIGURE 6.2. A constrained nonlinear random optimization problem: three contour plots with goal values 66, 76, and 96; two other outputs with lower bounds 4 and 9; optimal point A; points B and C on bound 9; point D on bound 4; local gradients at A through D for goal function and binding constraint, perpendicular to local tangent lines for binding constraint

Let $\mathbf{z}_o$ denote the input combination that gives a local minimum (or optimum; see the subscript o) for the deterministic variant of the problem defined in Eq. (6.8) through Eq. (6.10). The KKT conditions for $\mathbf{z}_o$ are then(besides some regularity conditions)

$$\begin{aligned} &\boldsymbol{\beta}_{0;-0} = \sum_{h \in A(\mathbf{z}_o)} \lambda_h \boldsymbol{\beta}_{h;-0} \\ &\lambda_h \geq 0 \\ &h \in A(\mathbf{z}_o) \end{aligned} \tag{6.21}$$

where $\boldsymbol{\beta}_{0;-0}$ denotes the k-dimensional vector with the gradient of the goal function, as we have already seen in Eq. (6.13); $A(\mathbf{z}_o)$ is the index set with the indices of those constraints that are binding at $\mathbf{z}_o$; λ_h is the Lagrangian multiplier for binding constraint h; $\boldsymbol{\beta}_{h;-0}$ is the gradient of the output in that binding constraint. Now we give two examples illustrating that Eq. (6.21) implies that the gradient of the objective is a nonnegative linear combination of the gradients of the binding constraints, at $\mathbf{z}_o$.

Example 6.1 *Figure 6.2 has only one binding constraint at the point A, so Eq. (6.21) then stipulates that the goal gradient $\boldsymbol{\beta}_{0;-0}$ and the gradient of the output with a binding constraint (namely, output $h = 2$) are two*

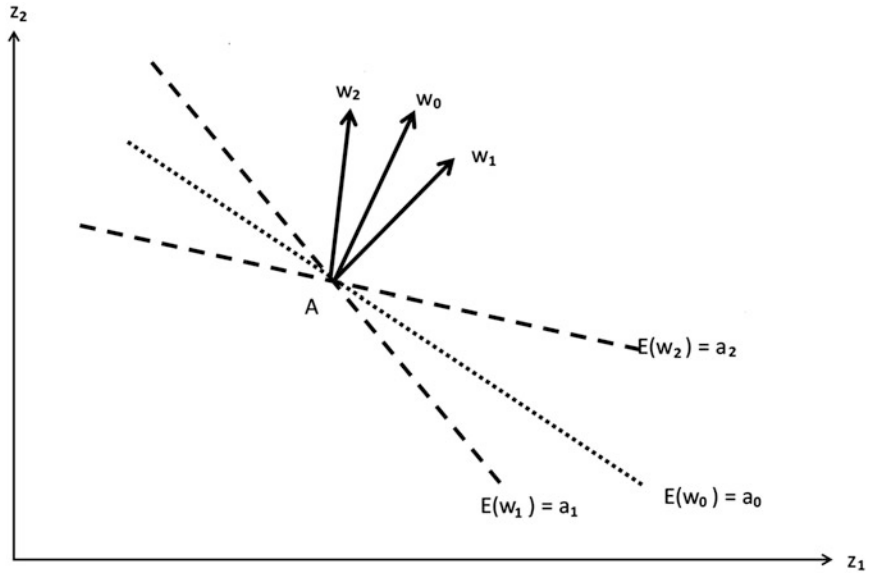

FIGURE 6.3. A LP problem: one contour line with goal value $w_0 = a_0$; two other outputs with upper bounds a_1 and a_2; optimal point A; local gradients at A for goal function and two binding constraints

vectors that point in the same direction. Indeed, point B has two gradients that point in different but similar directions—and so does C—whereas D has two gradients that point in completely different directions.

Example 6.2 *Figure 6.3 is actually a linear programming (LP) problem. One contour line for the goal output w_0 shows the input combination (z_1, z_2) that result in $w_0(z_1, z_2) = a_0$; the two other outputs are w_1 and w_2, which should satisfy the constraints $w_1 \leq a_1$ and $w_2 \leq a_2$; point A is the optimal input combination $\mathbf{z}_o$; the local gradients at point A are displayed for the goal function and the two binding constraints. Obviously, the goal gradient is a linear combination with positive coefficients of the two other gradients.*

Note: If the optimum occurs *inside* the feasible area, then there are no binding constraints so the KKT conditions reduce to the condition that the goal gradient be zero. Basic RSM includes tests for a zero gradient estimated from a second-order polynomial; see again Sect. 6.2.1.

In random simulation we must *estimate* the gradients; moreover, to check which constraints are binding, we must estimate the slacks of the constraints. This estimation changes the KKT conditions into a problem of nonlinear statistics. An asymptotic test is presented in Angün (2004), using the so-called *Delta method* and a generalized form of the so-called *Wald statistic*. A small-sample bootstrap test is presented in Bettonvil

et al. (2009), which we now present because it is simpler and it suits expensive simulation. Nevertheless, this bootstrap test is still rather complicated, so readers may skip to the next section (Sect. 6.3, on Kriging for optimization)—without lessening their understanding of the rest of this book.

As in basic RSM, we assume *locally constant variances and covariances* for each of the r simulation outputs w_h $(h = 0, 1, \ldots, r-1)$. OLS per univariate simulation output gives $\hat{\boldsymbol{\beta}}_h$ defined in Eq. (6.12). These estimators have the following estimated covariance matrix:

$$\widehat{\boldsymbol{\Sigma}}_{\hat{\boldsymbol{\beta}}_h, \hat{\boldsymbol{\beta}}_{h'}} = \widehat{\boldsymbol{\Sigma}}_{w_h, w_{h'}} \otimes (\mathbf{Z}'\mathbf{Z})^{-1} \quad (h, h' = 0, \ldots, r-1) \tag{6.22}$$

where $\otimes$ denotes the *Kronecker product* and $\widehat{\boldsymbol{\Sigma}}_{w_h, w_{h'}}$ is the $r \times r$ matrix with the classic estimators of the (co)variances based on the m replications at the local center so the replication number l runs from 1 through m (we use the symbol l instead of our usual symbol r, because r now stands for the number of output types); so $\widehat{\boldsymbol{\Sigma}}_{w_h, w_{h'}}$ is defined by

$$\widehat{\boldsymbol{\Sigma}}_{w_h, w_{h'}} = (\widehat{\sigma}_{h;h'}) = \left(\frac{\sum_{l=1}^{m} w_{h;l} - \overline{w}_h)(w_{h';l} - \overline{w}_{h'})}{m-1}\right). \tag{6.23}$$

The Kronecker product implies that $\widehat{\boldsymbol{\Sigma}}_{\hat{\boldsymbol{\beta}}_h, \hat{\boldsymbol{\beta}}_{h'}}$ is an $rq \times rq$ matrix where q denotes the number of regression parameters (so $q = 1 + k$ in a first-order polynomial); this matrix is formed from the $r \times r$ matrix $\widehat{\boldsymbol{\Sigma}}_{w_h, w_{h'}}$ by multiplying each of its elements by the entire $q \times q$ matrix $(\mathbf{Z}'\mathbf{Z})^{-1}$ (e.g., $\mathbf{Z}$ is an $N \times (1 + k)$ matrix in Eq. (6.5)). The matrix $\widehat{\boldsymbol{\Sigma}}_{w_h, w_{h'}}$ is singular if $m \leq r$; e.g., the case study in Kleijnen (1993) has $r = 2$ output types and $k = 14$ inputs so $m \geq 3$ replications of the center point are required. Of course, the higher m is, the higher is the power of the tests that use these replications. Bettonvil et al. (2009) does not consider cases with all n local points replicated or with CRN; these cases require further research.

Basic RSM (explained in Sect. 6.2.1) assumes that the output is Gaussian, and now in GRSM we assume that the r-variate simulation output is *multivariate Gaussian*. We use the *center* point to test whether a constraint is binding in the current local area, because this point is more representative of the local behavior than the extreme points of the R-III design applied in this area. To save simulation runs, we should start a local experiment at its center point including replications; if it turns out that either no constraint is binding or at least one constraint is violated in Eq. (6.24) below, then we do not need to test the other two hypotheses given in Eq. (6.25) and Eq. (6.26) and we do not need to simulate the remainder of the local design.

Like we do in basic RSM, we should test the *validity* of the local metamodel. GRSM assumes multiple outputs, so we may apply *Bonferroni's inequality*. If we reject a metamodel, then we have two options:

- Decrease the local area; e.g., halve the range of each input.

- Increase the order of the polynomial; e.g., switch from a first-order to a second-order polynomial.

We do not explore these options any further, but refer back to Sect. 6.2.2.

To test the KKT conditions, we test the following three null-hypotheses denoted by the superscripts (1) through (3):

1. The current solution is feasible and at least one constraint is binding; see Eq. (6.9):

$$H_0^{(1)} : E(w_{h'}|\mathbf{x} = \mathbf{0}) = a_{h'} \quad \text{with } h' = 1, \ldots, r-1 \tag{6.24}$$

   where $\mathbf{x} = \mathbf{0}$ corresponds with the center of the local area expressed in the standardized inputs.

2. The expected value of the estimated goal gradient may be expressed as the expected value of a linear combination of the estimated gradients of the simulation outputs in the binding constraints; i.e., in Eq. (6.21) we replace the deterministic quantities by their estimators:

$$H_0^{(2)} : E(\widehat{\boldsymbol{\beta}}_{0;-0}) = E(\sum_{h \in A(\mathbf{z}_o)} \widehat{\lambda}_h \widehat{\boldsymbol{\beta}}_h). \tag{6.25}$$

3. The Lagrangian multipliers in Eq. (6.25) are nonnegative:

$$H_0^{(3)} : E(\widehat{\boldsymbol{\lambda}}) \geq \mathbf{0}. \tag{6.26}$$

Each of these three hypotheses requires multiple tests, so we apply Bonferroni's inequality. Moreover, we test these three hypotheses sequentially, so it is hard to control the final type-I and type-II error probabilities (basic RSM has the same type of problem, but that RSM has nevertheless acquired a track record in practice).

Sub 1: To test $H_0^{(1)}$ in Eq. (6.24), we use the classic t-statistic:

$$t_{m-1}^{(h')} = \frac{\overline{w}_{h'}(\mathbf{x} = \mathbf{0}) - a_{h'}}{\sqrt{\widehat{\sigma}_{h';h'}/m}} \quad \text{with } h' = 1, \ldots, r-1 \tag{6.27}$$

where both the numerator and the denominator use the m replications at the local center point; see Eq. (6.23). This t-statistic may give the following three results:

(i) The statistic is *significantly positive*; i.e., the constraint for output h' is not binding. If none of the $(r-1)$ constraints is binding, then we have not yet found the optimal solution—assuming that at the optimum at least one constraint is binding; otherwise, we apply basic RSM. The search for better solutions continues; see again Sect. 6.2.4.

(ii) The statistic is *significantly negative*; i.e., the current local area does not give feasible solutions so we have not yet found the optimal solution. The search should back-up into the feasible area.

(iii) The statistic is *nonsignificant*; i.e., the current local area gives feasible solutions and the constraint for output h' is binding. We should then include the index of this gradient in $A(\mathbf{z}_o)$; see Eq. (6.25). And the KKT test proceeds as follows.

Sub 2 and 3: To estimate the *linear* combination in Eq. (6.25), we apply OLS with as explanatory variables the estimated gradients of the (say) J binding constraints; obviously, these explanatory variables are random. We collect these J estimated gradients in the $k \times J$ matrix $\widehat{\mathbf{B}}_{J;-0}$. These explanatory variables have linear weights $\boldsymbol{\lambda}$ that equal the parameters that are estimated through OLS, denoted by $\widehat{\boldsymbol{\lambda}}$. Let $\widehat{\widehat{\boldsymbol{\beta}}}_{0;-0}$ denote the OLS estimator of the goal gradient, so

$$\widehat{\widehat{\boldsymbol{\beta}}}_{0;-0} = \widehat{\mathbf{B}}_{J;-0}(\widehat{\mathbf{B}}'_{J;-0}\widehat{\mathbf{B}}_{J;-0})^{-1}\widehat{\mathbf{B}}'_{J;-0}\widehat{\boldsymbol{\beta}}_{0;-0} = \widehat{\mathbf{B}}_{J;-0}\widehat{\boldsymbol{\lambda}} \tag{6.28}$$

with $\widehat{\boldsymbol{\lambda}} = (\widehat{\mathbf{B}}'_{J;-0}\widehat{\mathbf{B}}_{J;-0})^{-1}\widehat{\mathbf{B}}'_{J;-0}\widehat{\boldsymbol{\beta}}_{0;-0}$; also see the general formula for OLS in Eq. (2.13). To quantify the *validity* of this linear approximation, we use the k-dimensional vector with the residuals

$$\widehat{\mathbf{e}}(\widehat{\widehat{\boldsymbol{\beta}}}_{0;-0}) = \widehat{\widehat{\boldsymbol{\beta}}}_{0;-0} - \widehat{\boldsymbol{\beta}}_{0;-0}. \tag{6.29}$$

$H_0^{(2)}$ in Eq. (6.25) implies that $\widehat{\mathbf{e}}(\widehat{\widehat{\boldsymbol{\beta}}}_{0;-0})$ in Eq. (6.29) should satisfy $E[\widehat{\mathbf{e}}(\widehat{\widehat{\boldsymbol{\beta}}}_{0;-0})] = \mathbf{0}$. Furthermore, $H_0^{(2)}$ involves a product of multivariates, so standard tests do not apply; therefor we use *bootstrapping*. We do not apply distribution-free bootstrapping, because in expensive simulation only the center point is replicated a few times. Instead, we apply *parametric bootstrapping*; i.e., we assume a Gaussian distribution (like we do in basic RSM), and we estimate its parameters from the simulation's I/O data. The resulting bootstrap algorithm consists of the following four steps, where the superscript $*$ is the usual symbol for a bootstrapped value.

Algorithm 6.2

1. Use the Monte Carlo method to sample

$$vec(\widehat{\boldsymbol{\beta}}^*_{0;-0}, \widehat{\mathbf{B}}^*_{J;-0}) \sim N(vec(\widehat{\boldsymbol{\beta}}_{0;-0}, \widehat{\mathbf{B}}_{J;-0}), \widehat{\boldsymbol{\Sigma}}_{vec(\widehat{\boldsymbol{\beta}}_{0;-0}, \widehat{\mathbf{B}}_{J;-0})}) \tag{6.30}$$

where $\text{vec}(\widehat{\boldsymbol{\beta}}^*_{0;-0}, \widehat{\mathbf{B}}^*_{J;-0})$ is a $(k + kJ)$-dimensional vector formed by stapling (stacking) the estimated k-dimensional goal gradient vector and the J k-dimensional vectors of the $k \times J$ matrix $\widehat{\mathbf{B}}^*_{J;-0}$; $\text{vec}(\widehat{\boldsymbol{\beta}}_{0;-0}, \widehat{\mathbf{B}}_{J;-0})$ is defined analogously to $\text{vec}(\widehat{\boldsymbol{\beta}}^*_{0;-0}, \widehat{\mathbf{B}}^*_{J;-0})$ but uses

Eq. (6.12), and $\widehat{\boldsymbol{\Sigma}}_{\text{vec}(\widehat{\boldsymbol{\beta}}_{0;-0},\widehat{\mathbf{B}}_{J;-0})}$ is the $(k+kJ)\times(k+kJ)$ matrix computed through Eq. (6.22).

2. Use the bootstrap values sampled in step 1 to compute the OLS estimate of the bootstrapped goal gradient where this OLS uses the bootstrapped gradients of the binding constraints as explanatory variables; i.e., use Eq. (6.28) adding the superscript $*$ to all random variables resulting in $\widehat{\widehat{\boldsymbol{\beta}}}^{*}_{0;-0}$ and $\widehat{\boldsymbol{\lambda}}^{*}$.

3. Use $\widehat{\widehat{\boldsymbol{\beta}}}^{*}_{0;-0}$ from step 2 and $\widehat{\boldsymbol{\beta}}^{*}_{0;-0}$ from step 1 to compute the bootstrap residual $\widehat{\mathbf{e}}(\widehat{\widehat{\boldsymbol{\beta}}}^{*}_{0;-0}) = \widehat{\widehat{\boldsymbol{\beta}}}^{*}_{0;-0}$ - $\widehat{\boldsymbol{\beta}}^{*}_{0;-0}$, analogously to Eq. (6.29); if any of the bootstrapped Lagrangian multipliers $\widehat{\boldsymbol{\lambda}}^{*}$ found in step 2 is negative, then increase the counter (say) c^{*} with the value 1.

4. Repeat the preceding three steps (say) 1,000 times, to obtain the estimated density function (EDF) of $\widehat{\mathbf{e}}(\widehat{\widehat{\boldsymbol{\beta}}}^{*}_{0;-0;j})$—which denotes the bootstrapped residuals per input j ($j = 1, \ldots, k$)—and the final value of the counter c^{*}. Reject $H_0^{(2)}$ in Eq. (6.25) if this EDF implies a two-sided $(1 - \alpha/(2k))$ CI that does not cover the value 0, where the factor k is explained by Bonferroni's inequality. Reject $H_0^{(3)}$ in Eq. (6.26) if the fraction $c^{*}/1{,}000$ is significantly higher than 50 %. To test the fraction $c^{*}/1{,}000$, approximate the binomial distribution through the normal distribution with mean 0.50 and variance $(0.50 \times 0.50)/1{,}000 = 0.00025$.
Comment: If the true Lagrangian multiplier is only "slightly" larger than zero, then "nearly" 50 % of the bootstrapped values is negative.

Altogether, this KKT test-procedure uses the following three models:

1. The *simulation* model, which is treated as a black box in GRSM.

2. The *regression* metamodel, which uses the simulation I/O data $(\mathbf{Z}, \mathbf{w})$ as input and gives the estimates of the gradients for the goal response $(\widehat{\boldsymbol{\beta}}_{0;-0})$ and the constrained responses with binding constraints $(\widehat{\mathbf{B}}_{J;-0})$. The regression analysis also gives the estimator $\widehat{\boldsymbol{\Sigma}}_{\text{vec}(\widehat{\boldsymbol{\beta}}_{0;-0},\widehat{\mathbf{B}}_{J;-0})}$ (estimated covariance matrix of estimated gradients).

3. The *bootstrap* model, which uses the regression output $(\widehat{\boldsymbol{\beta}}_{0;-0}, \widehat{\mathbf{B}}_{J;-0}, \widehat{\boldsymbol{\Sigma}}_{\text{vec}\widehat{\boldsymbol{\beta}}_{(0;-0},\widehat{\mathbf{B}}_{J;-0})}$ as parameters of the multivariate normal distribution of its output $\widehat{\boldsymbol{\beta}}^{*}_{0;-0}$ and $\widehat{\mathbf{B}}^{*}_{J;-0}$.

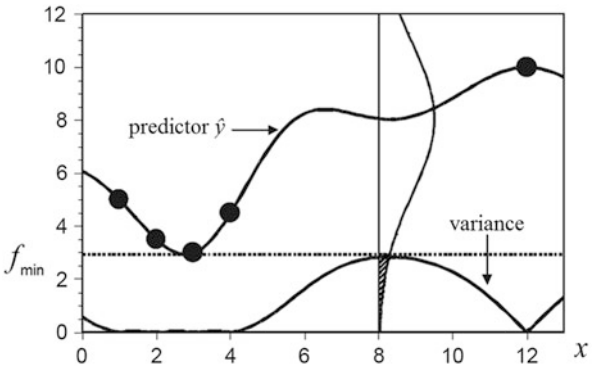

FIGURE 6.4. Expected improvement (EI) at $x = 8$: see *shaded area*; five observations on $f(x)$: see *dots*; Kriging predictor $\widehat{y}$ and variance of $\widehat{y}$

6.3 Kriging Metamodels for Optimization

In Sect. 6.2 we discussed optimization through RSM, which uses linear regression metamodels; namely, first-order and second-order polynomials fitted locally. Now we discuss optimization through Kriging metamodels, which are fitted globally. In Sect. 6.3.1 we shall discuss so-called *efficient global optimization* (EGO), which was originally developed for the minimization of the unconstrained output of a deterministic simulation model. In Sect. 6.3.2 we shall discuss constrained optimization in random simulation, using a combination of Kriging and *integer mathematical programming* (IMP) called KrIMP. We shall use the symbol x (not z) to denote the input (ignoring standardization), as the Kriging literature usually does.

6.3.1 Efficient Global Optimization (EGO)

EGO is a well-known *sequential* method; i.e., EGO selects the next input combination or "point" as experimental I/O results become available.Typically, EGO balances *local* and *global* search; i.e., EGO combines *exploitation* and *exploration*. More precisely, when selecting a new point, EGO estimates the maximum of the *expected improvement* (EI) comparing this new point and the best point that was found so far. EI uses the global Kriging metamodel to predict the output of a new point, while accounting for the predictor variance; this variance increases as a new point does not lie in a local subarea formed by some old points; also see Fig. 6.4. Obviously, EI is large if either the predicted value $\widehat{y}$ is much smaller than the minimum found so far denoted by $f_{\min} = \min w(\mathbf{x}_i)$, or the estimated predictor variance $\widehat{\sigma}(\mathbf{x})$ is large so the prediction shows much uncertainty. We shall further explain and formalize EGO in Algorithm 6.3 below.

The classic reference for EGO is Jones et al. (1998), which includes references to older publications that inspired EGO. In practice, EGO has shown to perform well when optimizing the unconstrained output of a deterministic simulation model; its theoretical convergence properties are analyzed in Bull (2011) and Vazquez and Bect (2010). EGO has also been implemented in software; see

http://cran.r-project.org/web/packages/DiceOptim/index.html.

We present only the *basic* EGO algorithm. There are many *variants* of EGO for deterministic and random simulations, constrained optimization, multi-objective optimization including Pareto frontiers, the "admissible set" or "excursion set", robust optimization, estimation of a quantile (instead of the mean), and Bayesian approaches.

Note: For these variants we list only the most recent publications plus some classic publications: Binois et al. (2015), Chevalier et al. (2014), Davis and Ierapetritou (2009), Feng et al. (2015), Forrester and Jones (2008), Forrester and Keane (2009), Forrester et al. (2008, pp. 90–101, 125–131, 141–153), Frazier (2010), Frazier et al. (2009), Gano et al. (2006), Gorissen (2010), Gramacy et al. (2015), Gramacy and Lee (2010), Huang et al. (2006), Jala et al. (2014), Jalali and van Nieuwenhuyse (2014), Janusevskis and Le Riche (2013), Kleijnen et al. (2012), Koch et al. (2015), Marzat et al. (2013), Mehdad and Kleijnen (2015), Morales-Enciso and Branke (2015), Müller and Shoemaker (2014), Nakayama et al. (2009), Picheny et al. (2013a), Picheny et al. (2013b), Preuss et al. (2012), Quan et al. (2013), Razavi et al. (2012), Regis (2014), Roustant et al. (2012), Salemi et al. (2014), Sasena et al. (2002), Scott et al. (2011), Scott et al. (2010), Sun et al. (2014), Svenson and Santner (2010), Tajbakhsh et al. (2013), Tong et al. (2015), Ur Rehman et al. (2014), Villemonteix et al. (2009a), Villemonteix et al. (2009b), Wagner (2013), Wiebenga (2014), and Williams et al. (2010).

We present a basic EGO algorithm for minimizing w, which denotes the output of a given deterministic simulation model. Our algorithm consists of the following five steps.

Algorithm 6.3

1. Fit a Kriging metamodel $y(\mathbf{x})$ to the old I/O simulation data $(\mathbf{X}, \mathbf{w})$. Comment: In Sect. 5.2 we presented details on Kriging metamodels for deterministic simulation, where $\mathbf{X}$ denoted the $n \times k$ matrix with the n combinations of the k simulation inputs, $\mathbf{w}$ denoted the n-dimensional vector with simulation outputs, and we speak of n "old" I/O data and a "new" input combination that is yet to be simulated.
2. Find the minimum output simulated so far: $f_{\min} = \min_{1 \le i \le n} w(\mathbf{x}_i)$.
3. Defining EI at a point $\mathbf{x}$ as

$$\mathrm{EI}(\mathbf{x}) = E\left[\max\left(f_{\min} - y(\mathbf{x}), 0\right)\right], \tag{6.31}$$

Jones et al. (1998) derives the following closed-form expression for its estimate:

$$\widehat{\mathrm{EI}}(\mathbf{x}) = (f_{\min} - \widehat{y}(\mathbf{x}))\,\Phi\left(\frac{f_{\min} - \widehat{y}(\mathbf{x})}{\widehat{\sigma}(\mathbf{x})}\right) + \widehat{\sigma}(\mathbf{x})\phi\left(\frac{f_{\min} - \widehat{y}(\mathbf{x})}{\widehat{\sigma}(\mathbf{x})}\right) \tag{6.32}$$

where $\widehat{y}(\mathbf{x})$ is the Kriging predictor with plugged-in estimates defined in Eq. (5.19); $\widehat{y}(\mathbf{x})$ is assumed to be normally distributed with mean $\widehat{y}(\mathbf{x})$ and standard deviation $\widehat{\sigma}(\mathbf{x})$ which is the square root of $\widehat{\sigma}^2(\mathbf{x})$; Φ and ϕ are the usual symbols for the cumulative distribution function and probability density function of the "standard" normal variable, which has zero mean and unit variance. Using Eq. (6.32), find $\widehat{\mathbf{x}}_o$, which denotes the estimate of $\mathbf{x}$ that maximizes $\widehat{\mathrm{EI}}(\mathbf{x})$.
Comment: To find the *maximizer* of Eq. (6.32), we may apply a *global optimizer* such as the *genetic algorithm* (GA) in Forrester et al. (2008, p. 78), the branch-and-bound algorithm in Jones et al. (1998), the genetic optimization using derivatives in Picheny et al. (2013b), or the evolutionary algorithm in Viana et al. (2013). Obviously, a *local* optimizer is undesirable, because $\mathrm{EI}(\mathbf{x})$ has many local optima; e.g., if $\mathbf{x} = \mathbf{x}_i$, then $\widehat{\sigma}^2(\mathbf{x}) = 0$ so $\mathrm{EI}(\mathbf{x}) = 0$. Instead of a global optimizer, we may use a set of *candidate points* selected through Latin hypercube sampling (LHS), and select the candidate point that maximizes $\widehat{\mathrm{EI}}(\mathbf{x})$; see Boukouvalas et al. (2014), Echard et al. (2011), Kleijnen and Mehdad (2013), Scott et al. (2012), and Taddy et al. (2009). Obviously, we may use *parallel* computer hardware to compute $\mathrm{EI}(\mathbf{x})$ for different candidate points $\mathbf{x}$, if we have such hardware available; also see Ginsbourger et al. (2010).

4. Run the simulation model with the input $\widehat{\mathbf{x}}_o$ found in step 3, to find the corresponding output $w(\widehat{\mathbf{x}}_o)$.

5. Fit a new Kriging metamodel to the old I/O data of step 1 and the new I/O of step 4. Update n and return to step 2 if the stopping criterion is not yet satisfied.
Comment: Sun et al. (2014) presents a fast approximation for re-estimation of the Kriging metamodel in exploitation versus exploration in discrete optimization via random simulation. Kamiński (2015) also presents several methods for avoiding re-estimation of the Kriging parameters. A stopping criterion may be max $\widehat{\mathrm{EI}}(\mathbf{x})$ is "close" to zero. Different stopping criteria are discussed in Razavi et al. (2012), Sun et al. (2014).

DiceOptim, which is an R package, implements EGO and enables the evaluation of multiple new points instead of a single new point. For details on DiceOptim we refer to Roustant et al. (2012).

Note: Mehdad and Kleijnen (2015) considers EGO with the predictor variance estimated through either *bootstrapped* Kriging (BK) or *conditional simulation* (CS); these two methods were discussed in Sect. 5.3. Several experiments suggest that BK and CS give predicted variances that do not differ significantly from each other, but that may be significantly bigger than the classic estimate (nevertheless, BK and CS do not give CIs that are significantly better than classic Kriging). Experiments with EGO using these alternative predictor variances suggest that EGO with BK or CS may or may not perform better than classic Kriging (CK). So, EGO may not be a good heuristic if the problem becomes complicated; also see Yarotsky (2013). More precisely, EGO with a specific correlation function and the classic estimator of the Kriging predictor variance replaced by the BK or CS estimators may be a refinement that does not improve EGO drastically. We might therefore stick to CK if we accept some possible inefficiency and prefer the simple analytical computations in Eq. (6.32).

6.3.2 *Kriging and Integer Mathematical Programming (KrIMP)*

Kleijnen et al. (2010) derives a heuristic that is not guided by EGO, but is more related to classic operations research (OR); this heuristic is called "Kriging and integer mathematical programming (KrIMP)". The heuristic addresses constrained optimization in random simulation, but may be easily adjusted (simplified) for deterministic simulation. Applications include an (s, S) inventory system with random lead times and a service level constraint that was originally investigated in Bashyam and Fu (1998), and a complicated call-center simulation in Kelton et al. (2007), which also minimizes costs while satisfying a service constraint; moreover, the call-center simulation must satisfy a budget constraint for the deterministic inputs (namely, resources such as personnel with specific skills) and these inputs must be nonnegative integers.

These two applications are examples of the *constrained nonlinear random optimization problem* that we have already presented in Eq. (6.8) through Eq. (6.10), but that we now augment with constraints for the deterministic inputs $\mathbf{z}$ that must satisfy s constraints f_g (e.g., budget constraints), and must belong to the set of non-negative integers $\mathbf{N}$:

$$\begin{aligned} &\min_{\mathbf{x}} E(w_0|\mathbf{x}) \\ &E(w_{h'}|\mathbf{x}) \geq c_h \ (h' = 1, \ldots, r-1) \\ f_g(\mathbf{x}) &\geq c_g \ (g = 1, \ldots, s) \\ x_j &\in \mathbf{N} \ (j = 1, \ldots, d). \end{aligned} \tag{6.33}$$

To solve this problem, KrIMP combines the following three methodologies:

1. *sequentialized* DOE to specify the next simulation combination (EGO also uses a sequential design);

2. *Kriging* to analyze the simulation I/O data that result from methodology #1 (like EGO does), and obtain explicit functions for $E(w_h|\mathbf{x})$ $(h = 0, 1, \ldots, r-1)$ instead of the implicit (black box) functions of simulation;

3. *integer nonlinear programming* (INLP) to estimate the optimal solution from the explicit Kriging metamodels that result from methodology #2; obviously INLP is a part of integer mathematical programming (IMP).

KrIMP comprises modules that use free off-the-shelf software. We may replace these modules, as we learn more about DOE, Kriging, and INLP. For example, we may replace Kriging by intrinsic Kriging (IK); we mentioned IK in Sect. 5.4. If our application has continuous inputs, then we may replace INLP by a solver that uses the gradients; these gradients are estimated by Kriging "for free", as we discussed in Sect. 5.2 (after Exercise 5.2). In future research we may adapt KrIMP for deterministic simulations with constrained multiple outputs and inputs.

Kleijnen et al. (2010) compares the results of KrIMP with those of OptQuest, which is the popular commercial heuristic embedded in discrete-event simulation software such as Arena; see Kelton et al. (2007). In the two applications mentioned above, KrIMP turns out to require fewer simulated input combinations and to give better estimated optima than OptQuest does.

Now we discuss some salient characteristics of KrIMP that are summarized in Fig. 6.5; readers may wish to skip to the next section (Sect. 6.4, on robust optimization). KrIMP simulates a new input combination and uses the augmented I/O data either to improve the Kriging metamodel or to find the optimum—similar to "exploration" and "exploitation" in EGO. The r global Kriging metamodels should be accurate enough to enable INLP to identify either infeasible points (which violate the constraints on the $r-1$ random outputs $E(w_{h'})$) or suboptimal points (which give a too high goal output $E(w_0)$ when trying to minimize $E(w_0)$). KrIMP may add a new point throughout the entire input-feasible area, which implies exploration. The global Kriging metamodel for output w_h $(h = 0, 1, \ldots, r-1)$ uses all observations for this output, obtained so far. To guide the INLP search, KrIMP simulates each point with a given relative precision so KrIMP is reasonably certain of the objective values and the possible violation of the constraints; i.e., KrIMP selects the number of replications m_i such that the halfwidth of the 90 % CI for the average simulation output is within

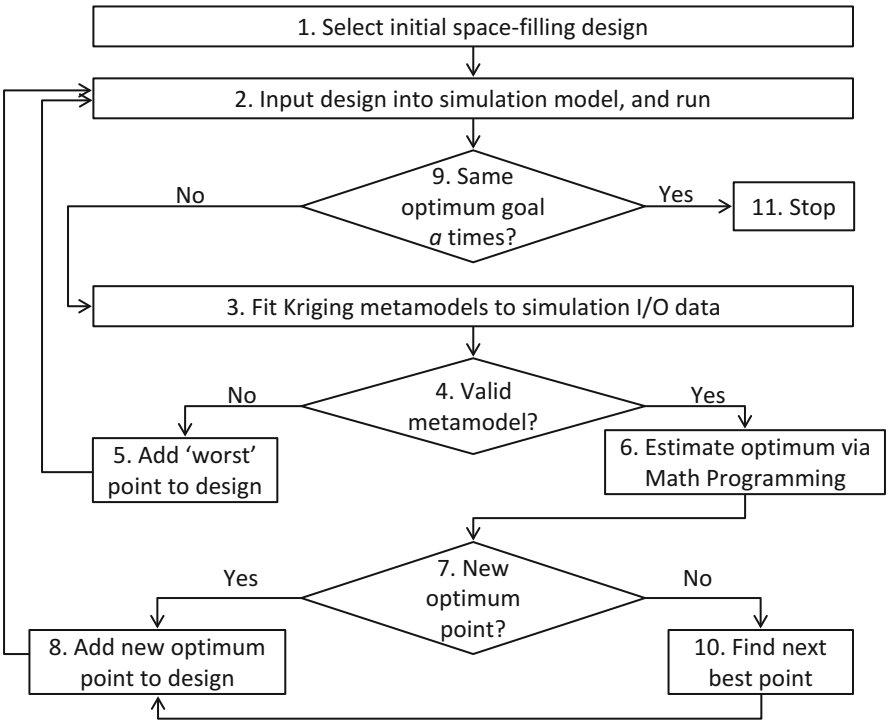

FIGURE 6.5. Overview of the KrIMP heuristic, combining Kriging and integer mathematical programming (IMP)

15 % of the true mean for all r outputs; also see our discussion on designs for linear regression metamodels with heterogeneous response variances, in Sect. 3.4.5. Furthermore, KrIMP uses CRN to improve the estimate of the optimum solution. KrIMP applies Kriging to the average output per simulated input combination, and does so for each of the r types of output; i.e., KrIMP does not use stochastic Kriging (SK) discussed in Sect. 5.6 and does not apply multivariate Kriging discussed in Sect. 5.10. KrIMP also uses distribution-free bootstrapping**,** combined with cross-validation. This bootstrapping gives an estimate of the predictor variance for output h at the deleted combination x_i, denoted by $\widehat{\sigma}^2(\widehat{y}_h^*(\mathbf{x}_i))$. Actually, the bootstrap in KrIMP accounts for multivariate (namely, r-variate) output created through CRN and for nonconstant replication numbers m_i This bootstrap and cross-validation give the following *Studentized* prediction errors for output h of deleted combination i with $i = 1, \ldots, n_{\text{cv}}$ where n_{cv} denotes the number of cross-validated combinations ($n_{\text{cv}} < n$ because

KrIMP avoids extrapolation in its Kriging):

$$t_{m_i-1}^{h,i} = \frac{\overline{w}_h(\mathbf{x}_i) - \widehat{y}_h(-\mathbf{x}_i)}{\{\widehat{\sigma}^2[\overline{w}_h(\mathbf{x}_i)] + \widehat{\sigma}^2[\widehat{y}_h^*(\mathbf{x}_i)]\}^{1/2}} \quad (h = 0, \ldots, r-1)\ (i = 1, \ldots, n_{\text{cv}}) \tag{6.34}$$

where

$$\widehat{\sigma}^2[\overline{w}_h(\mathbf{x}_i)] = \frac{\widehat{\sigma}^2[w_h(\mathbf{x}_i)]}{m_i}$$

with

$$\widehat{\sigma}^2[w_h(\mathbf{x}_i)] = \frac{\sum_{r=1}^{m_i} [w_{h;r}(\mathbf{x}_i) - \overline{w}_h(\mathbf{x}_i)]^2}{m_i - 1}.$$

The highest absolute value of the $t_{m_i-1}^{h,i}$ in Eq. (6.34) over all r outputs and all n_{cv} cross-validated combinations is denoted by $\max |t_{m_i-1}^{h,i}|$. Bonferroni's inequality implies that KrIMP divides the traditional type-I error rate α by $r \times n_{\text{cv}}$. If $\max |t_{m_i-1}^{h,i}|$ is significant, then KrIMP rejects all r Kriging metamodels; else, KrIMP uses the metamodels in its INLP, to estimate the constrained optimum.

Actually, we think that it is not good enough that KrIMP simulates each point with a given relative precision; i.e., we think that KrIMP should treat the $r-1$ constraints $E(w_{h'}|\mathbf{x}) \geq c_h$ in Eq. (6.33)—or Eq. (6.9) in case of GRSM—more rigorously such that

$$P[\forall h' : E(w_{h'}|\mathbf{x}) \geq c_h] \leq p \tag{6.35}$$

where p is a given small number; e.g., $p = 0.05$ is the probability that all $r-1$ constraints are satisfied. Obviously, this *chance-constrained* formulation concerns the $1-p$ quantile of the output $w_{h'}$ given the input combination $\mathbf{x}$: $P[(w_{h'}|\mathbf{x}) < c_h] = 1-p$. Similar quantiles are used in Feyzioğlu et al. (2005) applying second-order polynomials (instead of Kriging) to solve a multi-objective optimization problem (instead of a constrained optimization problem such as Eq. (6.33)); Kleijnen et al. (2011) also uses quantiles in a similar approach. Hong et al. (2015) also considers chance- constrained optimization in case of a given limited number of alternative simulated systems. Furthermore—inspired by EGO—we may adapt KrIMP such that it does not minimize the expected value $E(w_0|\mathbf{x})$ in Eq. (6.33), but it minimizes a preselected quantile—namely, the q-quantile—of the goal output: $\min_{\mathbf{x}}\ (w_{0;q}|\mathbf{x})$ where $P[(w_0|\mathbf{x}) < w_{0;q}] = q$. Obviously, if $q = 0.50$ and w_0 has a symmetric distribution (as the Gaussian assumption in Kriging implies), then $w_{0;q} = E(w_0)$. Various choices of q are discussed in Picheny et al. (2013a). Finally, to predict the joint probability in Eq. (6.35), KrIMP may use SK defined in Sect. 5.6.

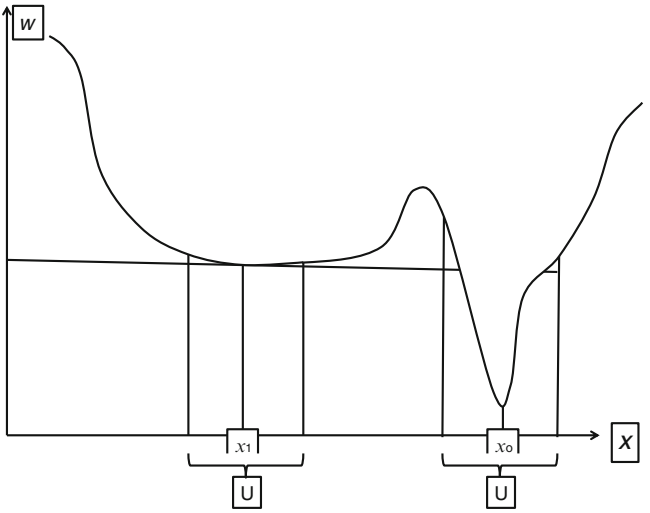

FIGURE 6.6. Robust solution x_1 in case of implementation error within range U, and nominally optimal solution x_o for simulation output $w = f_{\text{sim}}(\text{x})$

6.4 Robust Optimization

We start with a simple artificial example; see Fig. 6.6. In this example we assume that an implementation error (say) e occurs when a recommended solution is realized in the system being simulated; the possible values of this error fall within a range denoted by U, so $e \in U$ where the symbol U stands for the *uncertainty set* in the mathematical programming approach to robust optimization (see the next paragraph). The "nominally" optimal solution ignores this implementation error, so in the plot the global optimum is x_o. A better solution accounting for this implementation error is x_1, which is the best worst-case or min-max solution. In Taguchian robust optimization (also introduced in the next paragraph) we assume a probability density function (PDF) for e; e.g., we assume a Gaussian PDF with a mean $E(e) = 0$ and a variance such that $x + e$—the realized value of the implemented solution—has a 99 % probability of falling within the range U around the recommended solution x. Obviously, this PDF together with the curvature of the simulation's I/O function $w = f_{\text{sim}}(x)$ implies that in this example the simulation output w has $\text{Var}(w|x_o) > \text{Var}(w|x_1)$. A Taguchian solution tries to balance the mean and the variance of the output w through a robust solution for the decision variable x.

In general, the practical importance of robust optimization is emphasized by the panel reported in Simpson et al. (2004). Indeed, we think that robustness is crucial, given today's increased uncertainty in organizations and their environment; e.g., robust optimization may guide strate-

gic decisions on supply chains that are meant to be "agile" or "resilient". More specifically, the optimum solution for the decision variables—that we may estimate through local linear regression metamodels or global Kriging metamodels, as we explained in the preceding sections—may turn out to be inferior when ignoring uncertainties in the noncontrollable environmental variables; i.e., these uncertainties create a *risk*. Taguchi (1987) discusses "robust optimization" for the design of products. Ben-Tal and Nemirovski (1998) discusses robust optimization in mathematical programming models with uncertain coefficients.

Note: Taguchi (1987) is updated in Myers et al. (2009) and Wu and Hamada (2009). Furthermore, Ben-Tal and Nemirovski (1998) is updated in Ben-Tal and Nemirovski (2008), Gabrel et al. (2014), Wiesemann et al. (2014), and Yanikoğlu et al. (2015). Finally, robust optimization in simulation is also discussed in Hamarat et al. (2014) and Jalali and Van Nieuwenhuyse (2015). Robust decision-making is discussed in Grubler et al. (2015).

Taguchi (1987) emphasizes that in practice some inputs of a manufactured product are under complete control of the engineers, whereas other inputs are not; e.g., the design of a car engine is completely controlled by the engineers, but the driving style is not. Consequently, an engineering design—in this chapter we should distinguish between an engineering design and a statistical design—that allows some flexibility in its use is "better"; e.g., a car optimized only for the race circuit does not perform well in the city streets. Likewise, in simulation—either deterministic or random—our estimated optimum solution may be completely wrong when we ignore uncertainties in some inputs; e.g., the nominally optimal decision on the inventory control limits s (reorder level) and S (order-up-to level) may be completely wrong if we ignore the uncertainty in the parameters that we assumed for the random demand and delivery time distributions. Taguchi (1987) therefore distinguishes between two types of inputs:

- *decision variables*, which we now denote by d_j $(j = 1, \ldots, k)$ so $\mathbf{d} = (d_1, \ldots, d_k)'$, and
- *environmental inputs* or *noise factors* e_g $(g = 1, \ldots, c)$ so $\mathbf{e} = (e_1, \ldots, e_c)'$.

Note: Stinstra and Den Hertog (2008) points out that a source of uncertainty may be *implementation error*, which occurs whenever recommended values of decision variables are to be realized in practice; e.g., continuous values are hard to realize in practice, because of limited accuracy (see again Fig. 6.6). Besides implementation errors, there are validation errors of the simulation model (compared with the real system) and the metamodel (compared with the simulation model); also see the discussion on the validation of metamodels in simulation, in Kleijnen and Sargent (2000).

We perceive the following major differences between Taguchi's and Ben-Tal et al.'s approaches. Originally, Ben-Tal et al. assumed static deterministic linear problems solved by LP, whereas we assume dynamic nonlinear problems solved by either deterministic or random simulation. Ben -Tal et al. assume that uncertainty implies that the coefficients of the LP problem lie in a mathematical set called the *uncertainty set*; see the example in Fig. 6.6. We, however, assume that in deterministic or random simulation some inputs have a given statistical distribution; also see Sect. 5.9, in which we discussed risk analysis, uncertainty propagation, epistemic uncertainty, etc. Currently, Ben-Tal et al. also consider multi-stage nonlinear problems and uncertainty sets based on historical data. Another essential characteristic of simulation is that the objective and constrained functions are not known explicitly; actually, these functions are defined implicitly by the simulation model (we may replace these implicit functions by explicit metamodels, which are linear in the inputs if we use first-order polynomials or nonlinear if we use either higher-order polynomials or Kriging; metamodels treat the simulation model as a black box, as we explained in Sect. 2.1). Moreover, a random simulation model gives random outputs, which only estimate the true outputs (these outputs may be expected values or specific quantiles).

The goal of robust optimization is the design of robust products or systems, whereas the goal of *risk analysis* is to quantify the risk of a given engineering design; that design may turn out to be not robust at all. For example, Kleijnen and Gaury (2003) presents a random simulation of production-management (through methods such as Kanban, Conwip, and related methods), using RSM to estimate an optimal solution assuming a specific—namely the most likely—combination of environmental input values. Next, the robustness of this solution is estimated when the environment changes; technically, these environments are generated through LHS. In robust optimization, however, we wish to find a solution that—from the start of the analysis—accounts for all possible environments, including their likelihood; i.e., whereas Kleijnen and Gaury (2003) performs an *ex post* robustness analysis, we wish to perform an *ex ante* analysis.

Note: Whereas optimization is a "hot" topic in simulation (either deterministic or random), robust optimization is investigated in only a few publications; see the older references in Kleijnen (2008, pp. 131–132) and also Bates et al. (2006), Dengiz (2009), Kenett and Steinberg (2006), Meloni and Dellino (2015), Wiebenga (2014), and the references in the next subsections.

Next we shall discuss Taguchi's approach, using RSM in Sect. 6.4.1 and Kriging in Sect. 6.4.2; we shall discuss Ben-Tal et al.'s approach in Sect. 6.4.3.

6.4.1 Taguchian Robust Optimization Through RSM

Taguchi (1987) assumes a single output—which we denote by w—focusing on its mean μ_w and its variance caused by the noise factors $\mathbf{e}$ so $\sigma^2(w|\mathbf{d}) > 0$. These two outputs are combined in a *scalar loss function* such as the *signal-to-noise* or *mean-to-variance* ratio μ_w/σ_w^2; also see the discussion of these functions in Myers et al. (2009, pp. 486–488). Instead of this scalar function, we use both μ_w and σ_w^2 separately and formulate the following mathematical problem:

$$\min E(w|\mathbf{d}) \quad \text{such that } \sigma(w|\mathbf{d}) \leq T \tag{6.36}$$

where $E(w|\mathbf{d})$ is the mean of the simulation output w determined by the distribution function of the environmental variables $\mathbf{e}$ and controlled through the decision factors $\mathbf{d}$; the constraint concerns $\sigma(w|\mathbf{d})$, which is the standard deviation of the goal output w, and has a given upper threshold T. We also refer to Myers et al. (2009, pp. 488–495) and the surveys on robust optimization in Beyer and Sendhoff (2007) and Park et al. (2006).

Note: An alternative for the standard deviation $\sigma(w|\mathbf{d})$ in Eq. (6.36) may be the variance $\sigma^2(w|\mathbf{d})$, but the standard deviation uses the same measurement unit as the mean $(w|\mathbf{d})$. Kleijnen and Gaury (2003) uses the probability of a specific disastrous event happening; e.g., $P(w > c|\mathbf{d})$.

Taguchi's worldview has been very successful in production engineering, but statisticians have seriously criticized his statistical techniques; see the panel report in Nair (1992). To this report we add that in simulation we can experiment with many more inputs, levels (values), and combinations than we can in real-life experiments; Taguchians and many statisticians focus on real-life experiments. Myers et al. (2009, pp. 502–506) combines Taguchi's worldview with the statisticians' RSM. Whereas Myers et al. (2009) assumes that the multivariate noise $\mathbf{e}$ has the covariance matrix $\mathbf{\Omega_e} = \sigma_e^2\mathbf{I}$—and the mean $\mu_\mathbf{e}$— we assume a general $\mathbf{\Omega_e}$. Whereas Myers et al. (2009) superimposes contour plots for the mean and variance of the output to find a robust solution, we use more general and flexible mathematical programming. This mathematical programming, however, requires specification of threshold values such as T in Eq. (6.36). Unfortunately, managers may find it hard to select specific values such as T, so we may try different values and estimate the corresponding Pareto-optimal efficiency frontier. Decreasing T in Eq. (6.36) increases $E(w|\mathbf{d})$ if the constraint with the old T was binding. So, changing T gives an estimate of the Pareto-optimal efficiency frontier; i.e., $E(w|\mathbf{d})$ and $\sigma(w|\mathbf{d})$ are criteria requiring a trade-off. To estimate the variability of this frontier resulting from the various estimators, we may use bootstrapping. For details on our adaptation of the approach in Myers et al. (2009) we also refer to Dellino et al. (2010).

More precisely, Myers et al. (2009) fits a *second-order polynomial* for the decision variables $\mathbf{d}$ that are to be optimized. Possible effects of the environmental variables $\mathbf{e}$ are modelled through a first-order polynomial

	e combination			
d combination	1	2	...	n_e
1				
2				
...				
n_d				

TABLE 6.1. A crossed design combining a design for the decision variables d and a design for the environmental inputs e

in these variables **e**. *Control-by-noise* two-factor interactions (between **d** and **e**) are also considered. Altogether, the following "incomplete" second-order polynomial is fitted:

$$\begin{aligned} y &= \beta_0 + \sum_{j=1}^{k} \beta_j d_j + \sum_{j=1}^{k} \sum_{j' \geq j}^{k} \beta_{j;j'} d_j d_{j'} + \sum_{g=1}^{c} \gamma_g e_g + \sum_{j=1}^{k} \sum_{g=1}^{c} \delta_{j;g} d_j e_g + \epsilon \\ &= \beta_0 + \boldsymbol{\beta}' \mathbf{d} + \mathbf{d}' \mathbf{B} \mathbf{d} + \boldsymbol{\gamma}' \mathbf{e} + \mathbf{d}' \boldsymbol{\Delta} \mathbf{e} + \epsilon \end{aligned} \tag{6.37}$$

where we now denote the regression residual through the symbol ϵ (instead of e); we denote the first-order effects by $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_k)'$ for **d** and $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_c)'$ for **e**; we let **B** denote the $k \times k$ symmetric matrix with on the main diagonal the purely quadratic effects $\beta_{j;j}$ of **d** and off the diagonal half the interactions $\beta_{j;j'}/2$ of **d**; and we let $\boldsymbol{\Delta}$ denote the $k \times c$ matrix with the interactions $\delta_{j;g}$ between decision variable d_j and environmental variable e_g.

If $E(\epsilon) = 0$, then Eq. (6.37) implies the following regression predictor for μ_w (true mean of output w):

$$\mu_y = \beta_0 + \boldsymbol{\beta}' \mathbf{d} + \mathbf{d}' \mathbf{B} \mathbf{d} + \boldsymbol{\gamma}' \boldsymbol{\mu}_{\mathbf{e}} + \mathbf{d}' \boldsymbol{\Delta} \boldsymbol{\mu}_{\mathbf{e}}. \tag{6.38}$$

Because the covariance matrix of the noise variables **e** is $\boldsymbol{\Omega}_{\mathbf{e}}$, the regression predictor for σ_w^2 (true variance of w) is

$$\sigma_y^2 = (\boldsymbol{\gamma}' + \mathbf{d}' \boldsymbol{\Delta}) \boldsymbol{\Omega}_{\mathbf{e}} (\boldsymbol{\gamma} + \boldsymbol{\Delta}' \mathbf{d}) + \sigma_\epsilon^2 = \mathbf{l}' \boldsymbol{\Omega}_{\mathbf{e}} \mathbf{l} + \sigma_\epsilon^2 \tag{6.39}$$

where $\mathbf{l} = (\boldsymbol{\gamma} + \boldsymbol{\Delta}' \mathbf{d}) = (\partial y / \partial e_1, \ldots, \partial y / \partial e_c)'$ so **l** is the *gradient* with respect to **e**. Consequently, the larger the gradient's elements are, the larger σ_y^2 is—which stands to reason. Furthermore, if there are no control-by-noise interactions so $\boldsymbol{\Delta} = \mathbf{0}$, then we cannot control σ_y^2 through **d**.

To enable estimation of the regression parameters in Eq. (6.37), we follow the usual Taguchian approach and use a *crossed design*; i.e., we combine the design or *inner array* for **d** with n_d combinations and the design or *outer array* for **e** with n_e combinations such that the crossed design has $n_d \times n_e$ combinations as in Table 6.1. To estimate the optimal **d** through the second-order polynomial in Eq. (6.37), we use a CCD; also see again our

discussion below Eq. (6.4). For the first-order polynomial in $\mathbf{e}$, we use a R-III design; see the discussion below Eq. (6.3). Obviously, the combination of these two designs enables the estimation of the two-factor interactions $\delta_{j;g}$.

Note: Designs that are more efficient than crossed designs are discussed in Dehlendorff et al. (2011), Dellino et al. (2010), Khuri and Mukhopadhyay (2010), Kolaiti and Koukouvinos (2006), and Myers et al. (2009).

To use *linear regression analysis* for the estimation of the parameters in Eq. (6.37), we reformulate that equation as

$$y = \boldsymbol{\zeta}'\mathbf{x} + \epsilon \tag{6.40}$$

with the q-dimensional vector $\boldsymbol{\zeta} = (\beta_0, \ldots, \delta_{k;c})'$ and $\mathbf{x}$ defined in the obvious way; e.g., the element corresponding with $\beta_{1;2}$ (interaction between d_1 and d_2) is $d_1 d_2$. Obviously, Eq. (6.40) is linear in $\boldsymbol{\zeta}$, but not in $\mathbf{d}$.

The OLS estimator $\widehat{\boldsymbol{\zeta}}$ of $\boldsymbol{\zeta}$ in Eq. (6.40) is

$$\widehat{\boldsymbol{\zeta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{w} \tag{6.41}$$

where $\mathbf{X}$ is the $N \times q$ matrix of explanatory variables with $N = \sum_{i=1}^{n} m_i$ where n denotes the number of different combinations of $\mathbf{d}$ and $\mathbf{e}$, and m_i denotes the number of replications in combination i (obviously, $m_i = 1$ in deterministic simulation); $\mathbf{w}$ is the vector with the N "stapled" (or "stacked") outputs $w_{i;r}$ where $r = 1, \ldots, m_i$.

The covariance matrix of the OLS estimator $\widehat{\boldsymbol{\zeta}}$ defined in Eq. (6.41) is

$$\boldsymbol{\Sigma}_{\widehat{\boldsymbol{\zeta}}} = (\mathbf{X}'\mathbf{X})^{-1}\sigma_w^2 \tag{6.42}$$

where σ_w^2 equals σ_ϵ^2 because we assume the metamodel in Eq. (6.37) to be valid and ϵ to be white nose so $\epsilon \sim \text{NIID}(0, \sigma_\epsilon^2)$. This variance is estimated by the *mean squared residuals* (MSR), which we have already defined in Eq. (2.20) and we repeat here for convenience:

$$\text{MSR} = \frac{(\widehat{\mathbf{y}} - \mathbf{w})'(\widehat{\mathbf{y}} - \mathbf{w})}{N - q} \tag{6.43}$$

where $\widehat{\mathbf{y}} = \widehat{\boldsymbol{\zeta}}'\mathbf{x}$; also see Eq. (6.6).

Note: Santos and Santos (2011) allows σ_w^2 to be nonconstant, and estimates a metamodel for σ_w—besides a metamodel for μ_w. Shin et al. (2011) also estimates one metamodel for the mean and one for the variance.

To estimate the predictor mean μ_y in the left-hand side of Eq. (6.38), we simply plug $\widehat{\boldsymbol{\zeta}}$ defined in Eq. (6.41) into the right-hand side of Eq. (6.38), which also contains the known $\mathbf{d}$ and $\boldsymbol{\mu}_\mathbf{e}$. We also estimate the predictor variance σ_y^2 by plugging $\widehat{\boldsymbol{\zeta}}$ into Eq. (6.39), where $\boldsymbol{\Omega}_\mathbf{e}$ is known. We point out that Eq. (6.39) involves products of unknown parameters, so it implies a *nonlinear* estimator $\widehat{\sigma}_y^2$; plugged-in estimators certainly create bias, but we ignore this bias.

Note: Apley and Kim (2011) follows a Bayesian approach—called "cautious robust design"—which does account for the uncertainty of the parameter estimator $\widehat{\boldsymbol{\zeta}}$, and gives an analytical (instead of a simulation) solution.

Our final goal is to solve Eq. (6.36). We solve this constrained minimization problem through a mathematical programming solver; e.g., Matlab's "fmincon"—but a different solver might be used; see Gill et al. (2000). This solution estimates the robust optimal solution for the decision variables and the resulting mean and variance.

Dellino et al. (2010) presents an example; namely, the *economic order quantity* (EOQ) for an environment with a demand rate that is uncertain—but this rate has a known distribution (implying "uncertainty propagation" of "epistemic" uncertainty; see again Sects. 1.1 and 5.9). This example demonstrates that if management prefers low variability of inventory costs, then they must pay a price; i.e., the expected costs increases. Furthermore, different values are indeed found for the robust EOQ and the classic EOQ; this classic EOQ assumes a known fixed demand rate. More examples are referenced in Yanikoğlu et al. (2015).

Note: The solution estimated through robust optimization is a nonlinear function of the simulation output so there are no standard CIs for this solution. We may therefore evaluate the reliability of the estimated solution through bootstrapping. The final decision on the preferred solution is up to management; they should select a compromise combination of the decision variables depending on their risk attitude. Shang et al. (2004) uses plots to decide on a compromise solution; also see Fig. 6.7 where the horizontal double-pointed arrows denote the (bootstrap) CIs for the optimal solutions for the mean and variance, respectively, which do not overlap in this example. However, we leave this bootstrapping for future research. We also refer to Apley and Kim (2011), discussed in the immediately preceding Note.

Note: Future research may also address the following issues. Instead of minimizing the mean under a standard-deviation constraint as in Eq. (6.36), we may minimize a specific quantile of the simulation output distribution or minimize the *conditional value at risk* (CVaR); CVaR considers only one-sided deviations from the mean (whereas the standard deviation and the variance consider deviations on both sides of the mean). Indeed, Angün (2011) replaces the standard deviation by the CVaR and considers random simulation of the (s, S) inventory system in Bashyam and Fu (1998) and the call center in Kelton et al. (2007); in case the problem is found to be convex, this problem can be solved very efficiently. Instead of Eq. (6.36), Broadie et al. (2011) estimates the probability of a large loss in financial risk management, for various "scenarios"—these scenarios correspond with the combinations of environmental variables $\mathbf{e}$ in our approach—and examines the sequential allocation of the computer budget to estimate this loss, allowing for variance heterogeneity; we also refer to Sun et al. (2011), which we shall briefly discuss in Sect. 6.4.2 (last Note). Other risk measures are the *expected shortfall*, which is popular in the actuarial literature;

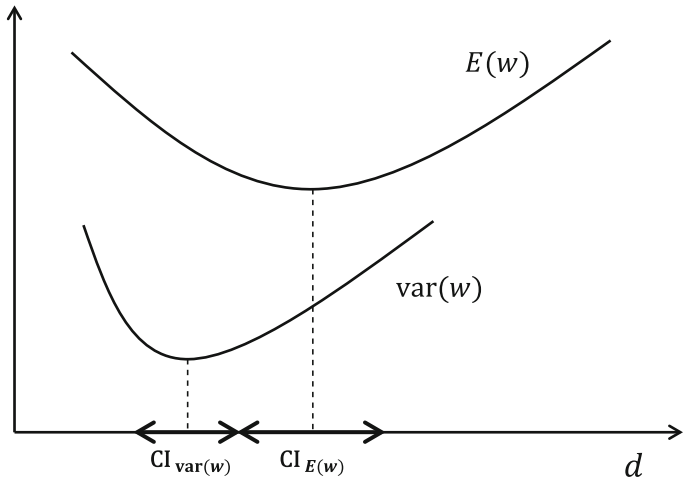

FIGURE 6.7. Example of robust optimization of a simulation model with output w, a single controllable input d; and one or more uncontrollable inputs $\mathbf{e}$ so $\text{Var}(w|\mathbf{d}) > 0$

see again Angün (2011) and also Gordy and Juneja (2010) and Lan et al. (2010). Furthermore, multi-objective optimization and genetic algorithms for estimating Pareto frontiers are discussed in Koziel et al. (2014) and Shahraki and Noorossana (2014). Another methodology for estimating the Pareto frontier is developed in Shin et al. (2011), solving a bi-objective robust design problem considering two quality characteristics. Rashid et al. (2013) also presents a method for the estimation of the efficiency frontier. Ardakani and Wulff (2013) gives an extensive overview of various optimization formulations in case of multiple outputs, using a multi-objective decision-making perspective; these formulations include our Eq. (6.36), the Pareto frontier, so-called *desirability functions*, etc.; an application of this desirability function—combining two outputs into a single criterion—is presented in Yalçinkaya and Bayhan (2009).

6.4.2 Taguchian Robust Optimization Through Kriging

Dellino et al. (2012) combines the world view of Taguchi (1987) and Kriging metamodels, for robust optimization in deterministic simulation. This approach is illustrated through the EOQ example with uncertain demand rate that was also used in Dellino et al. (2010) (discussed in the preceding subsection, Sect. 6.4.1).

More precisely, Taguchi's low-order polynomial metamodels are replaced by ordinary Kriging (OK) metamodels. Moreover, bootstrapping is applied to quantify the variability in the estimated Kriging metamodels. Instead

of Taguchi's signal-noise criterion μ_w/σ_w^2, now Kriging is combined with nonlinear programming (NLP) (NLP is also discussed in the subsection on KrIMP, Sect. 6.3.2). Changing the threshold values in the NLP model—that will be defined in Eq. (6.44)—enables the estimation of the Pareto frontier. The EOQ example shows that robust optimization may require an order quantity that differs from the classic EOQ (such a difference is also found through the RSM approach in Sect. 6.4.1).

Specifically, Dellino et al. (2012) uses the following NLP model:

$$\min E(w|\mathbf{d}) \text{ such that } \sigma(w|\mathbf{d}) \leq T \tag{6.44}$$

where $E(w|\mathbf{d})$ is the mean of the simulation output w determined by the distribution function of the environmental variables $\mathbf{e}$ and controlled through the decision factors $\mathbf{d}$; the constraint concerns $\sigma(w|\mathbf{d})$, which is the standard deviation of the goal output w, and has a given upper threshold T. The same problem was defined in Eq. (6.36).

Next, $E(w|\mathbf{d})$ and $\sigma(w|\mathbf{d})$ are replaced by their Kriging metamodels. Obviously, the constrained minimization problem in Eq. (6.44)—combined with the explicit Kriging approximations—is nonlinear in the decision variables $\mathbf{d}$.

We point out that we are *not* interested in the functional relationship between the output w and the environmental inputs $\mathbf{e}$; in the RSM approach—in Eq. (6.37)—we do estimate a low-order polynomial in $\mathbf{e}$ and $\mathbf{d}$. Following Taguchi (1987), we consider the inputs $\mathbf{e}$ as noise. Unlike Taguchi, we now use LHS to sample (say) n_e combinations of the environmental inputs $\mathbf{e}$. For the decision variables $\mathbf{d}$ we do not use a CCD, whereas we did use a CCD in the RSM approach in Sect. 6.4.1 (between Eqs. (6.39) and (6.40)). LHS does not impose a relationship between n_e (number of combinations of $\mathbf{e}$) and c (number of environmental inputs), as we explained in our discussion of LHS in Sect. 5.5.1. If we do not have prior information about the likelihood of specific values for $\mathbf{e}$, then we might use independent uniform distributions per environmental input e_g $(g = 1, \ldots, c)$ (also see our brief discussion of Bayesian prior distributions at the end of Sect. 5.9 on risk analysis). Whereas classic optimization assumes a single "scenario" (e.g., the most likely combination of environmental inputs), we now estimate the parameters in the Kriging metamodel for the decision variables $\mathbf{d}$ from the simulation outputs averaged over all simulated combinations of $\mathbf{e}$; these combinations are sampled through LHS accounting for the distribution of $\mathbf{e}$. We now explain this Kriging approach to Taguchian optimization, in more detail.

In general, if we wish to fit a Kriging metamodel to obtain an explicit approximation for the I/O function of a simulation model, then we often use LHS to obtain the I/O simulation data—as we have already discussed in detail in Sect. 5.5. Dellino et al. (2012) also uses LHS, as part of the following two approaches, especially developed for robust optimization:

1. Analogously to Dellino et al. (2010), fit two Kriging metamodels; namely, one model for $E(w|\mathbf{d})$ and one for $\sigma(w|\mathbf{d})$—both estimated from the *simulation* I/O data.

2. Analogously to Lee and Park (2006), fit a single Kriging metamodel to a relatively small number (say) n of combinations of $\mathbf{d}$ and $\mathbf{e}$; next use this metamodel to compute the *Kriging predictions* for the simulation output w for $N \gg n$ combinations of $\mathbf{d}$ and $\mathbf{e}$ accounting for the distribution of $\mathbf{e}$.

First we summarize approach 1, then approach 2, and finally the two approaches together.

Sub 1: We start with selecting the input combinations for the simulation model through a *crossed* design for $\mathbf{d}$ and $\mathbf{e}$; see again Table 6.1. Such crossed designs are traditional in Taguchian design (as we discussed between Eqs. (6.39) and (6.40)). To facilitate the fitting of a Kriging metamodel in $\mathbf{d}$, we select the n_d combinations of $\mathbf{d}$ *space-filling*; e.g., we use a maximin LHS, as we discussed in Sect. 5.5.1. The n_e combinations of $\mathbf{e}$, however, we *sample* from the distribution of $\mathbf{e}$; we may use LHS for this (stratified) sampling. The resulting I/O data form an $n_d \times n_e$ matrix. Such a crossed design enables the following estimators of the n_d conditional means and variances where $i = 1, \ldots, n_d$:

$$\overline{w}_i = \frac{\sum_{j=1}^{n_e} w_{i;j}}{n_e} \quad \text{and} \quad s_i^2(w) = \frac{\sum_{j=1}^{n_e} (w_{i;j} - \overline{w}_i)^2}{n_e - 1}. \tag{6.45}$$

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

Sub 2: We start with a relatively small number (say) n of combinations of the $k + c$ inputs $\mathbf{d}$ and $\mathbf{e}$; we select these combinations through a space-filling design (so we not yet sample $\mathbf{e}$ from its distribution). Next, we use this $n \times (k+c)$ matrix with the simulation input data and the n-dimensional vector with the corresponding simulation outputs w, to fit a Kriging metamodel that approximates w as a function of $\mathbf{d}$ and $\mathbf{e}$. Finally, we use a design with $N \gg n$ combinations, crossing a space-filling design with N_d combinations of $\mathbf{d}$ and LHS with N_e combinations of $\mathbf{e}$ accounting for the distribution of $\mathbf{e}$. We use this Kriging metamodel to compute the predictors $\widehat{y}$ of the N outputs. We then derive the N_d conditional means and standard deviations using Eq. (6.45) replacing n_d and n_e by N_d and N_e and replacing the simulation output w by the Kriging predictor $\widehat{y}$. We use these predictions to fit two Kriging metamodels; namely, one Kriging model for the mean output and one for the standard deviation of the output.

Sub 1 and 2: Next we use the two Kriging metamodels—namely, one model for the mean and one model for the standard deviation of the simulation output—as input for the NLP model in Eq. (6.44) to estimate the robust optimal I/O combination. Finally, we vary the threshold T to estimate the Pareto frontier. We call this frontier the "original" frontier, to be distinguished from the bootstrapped frontier (discussed in the next Note).

Note: The original frontier is built on estimates of the mean and standard deviation of the simulation output. To quantify the variability in the estimated mean and standard deviation, we apply *distribution-free bootstrapping*. Moreover, bootstrapping assumes that the original observations are IID; however, the crossed design for $\mathbf{d}$ and $\mathbf{e}$ (see again Table 6.1) implies that the n_d observations on the output for a given combination of the c environmental factors $\mathbf{e}$ are not independent; we might compare this dependence with the dependence created by CRN. Therefore, we sample the n_d-dimensional vectors $\mathbf{w}_j$ $(j = 1, \ldots, n_e)$ n_e times with replacement. This resampling gives the n_e bootstrapped observations $\mathbf{w}_j^*$. This gives the bootstrapped conditional means $\overline{w}_i^*$ and standard deviations s_i^*. To these $\overline{w}_i^*$ and s_i^*, we apply Kriging. These two Kriging metamodels together with the NLP model in Eq. (6.44) give the predicted optimal bootstrapped mean and standard deviation. Repeating this bootstrap sampling (say) B times gives CIs. More research is needed to discover how exactly to use these CIs to account for management's risk attitude; also see Zhang and Ma (2015). Furthermore, Simar and Wilson (1998) studies bootstrapping for estimating the variability of a frontier; namely, the *efficiency frontier* in *data envelop analysis* (DEA), estimated through a LP model. We also refer to Dellino and Meloni (2013) for quantifying the variability of a fitted metamodel, using bootstrapping and cross-validation.

To compare (validate) the robust solution and the classic (nominally optimal) solution, we may sample new combinations of the environmental inputs; i.e., we replace the old LHS combinations by new combinations, because the old combinations favor the robust solution which uses estimates based on these old combinations.

Note: Using a Bayesian approach to the analysis of the I/O data from simulation, Tan (2014a) first fits a Kriging model to the I/O data, then approximates this Kriging model through a so-called *orthonormal polynomial* (which is more complicated than the polynomial models that we discussed in Sect. 2.1), and finally uses this polynomial for "functional analysis of variance" or FANOVA (we discussed FANOVA in Sect. 5.8). This FANOVA can decompose $\sigma^2(w|\mathbf{d})$ (the response variance at a given combination of the decision variables $\mathbf{d}$) into a sum of variances due to the main effects and interactions among the environmental variables $\mathbf{e}$; several sensitivity indexes within the context of robust optimization can be defined. We also refer to Tan (2014b).

Note: EGO (with its EI criterion and Kriging metamodeling, explained in Sect. 6.3.1) may also be used for robust optimization. Actually, Marzat et al. (2013) refers to several publications that extend EGO accounting for a probability distribution of $\mathbf{e}$ such that it minimizes a weighted average of the response w over a discrete set of values for these $\mathbf{e}$. Marzat et al. (2013) combines EGO with algorithms for solving the following *minimax* problem: estimate the combination of $\mathbf{d}$ that minimizes the maximum response when the worst combination of $\mathbf{e}$ occurs; several test functions are investigated. Furthermore, Ur Rehman et al. (2014) extends EGO accounting for im-

plementation errors within an "uncertainty set" (see Sect. 6.4.3 below) and estimating the "best worst-case" or "min-max" solution. Janusevskis and Le Riche (2013) also applies Kriging and EGO for robust optimization.

Note: In Sect. 5.6 on stochastic Kriging (SK) we have already mentioned that the simulation response may be a quantile, which may be relevant in chance-constrained (probabilistically constrained) optimization. Simulation optimization with probabilistic constraints—namely, min $E(w_0)$ such that $P(w_1 \leq c) \geq p$—is discussed in Andrieu et al. (2011) and Sakallı and Baykoç (2011); we also refer back to the references on EGO adapted for chance-constrained optimization in Sect. 6.3.1, and Eq. (6.35) in Sect. 6.3.2 on KrIMP. Stochastically constrained optimization in a R&S context is discussed in Hong et al. (2015). We also refer back to the "expected shortfall", discussed in Sect. 6.4.1 (last Note in that subsection) including references to Broadie et al. (2011) and Sun et al. (2011); those references and also Chen and Kim (2014) and Gan and Lin (2015) use *nested simulation*, which should be distinguished from the crossed designs—as we briefly discusses in the Note after Eq. (6.39). Furthermore, NLP may be replaced by some other optimizer; e.g., an evolutionary algorithm. Finally, we may also apply Dellino et al. (2012)'s methodology to random simulation models, replacing ordinary Kriging (OK) by stochastic Kriging (SK) or stochastic intrinsic kriging (SIK); see the discussions on SK and SIK in Chap. 5. Yin et al. (2015) use simulation of finite element models with uncertain environmental inputs. This simulation is followed by univariate Kriging metamodels. These metamodels are the inputs for a multicriteria optimization problem that combines the means and standard deviations of the multiple simulation outputs. This problem is solved through particle-swarm heuristics.

6.4.3 Ben-Tal et al.'s Robust Optimization

If the mathematical programming (MP) solution ignores the uncertainty in the coefficients of the MP model, then the so-called *nominal solution* may easily violate the constraints in the given model. The *robust solution* may result in a slightly worse value for the goal variable, but it increases the probability of satisfying the constraints; i.e., a robust solution is "immune" to variations of the variables within the *uncertainty set*. Given historical data on the environmental variables $\mathbf{e}$, Yanikoğlu et al. (2015) derives a specific uncertainty set for p where p denotes the unknown density function of $\mathbf{e}$ that is compatible with the historical data on $\mathbf{e}$ (more precisely, p belongs to this set with confidence $1-\alpha$ if we select some phi-divergence measure such as the well-known chi-square distance). The mathematical challenge in robust optimization of MP models is to develop a computationally tractable so-called *robust counterpart* of the original problem. In this section we do not present the mathematical details of the derivation of tractable robust counterparts, but refer to the references that we gave above.

Note: Taguchians assume a specific distribution for the environmental variables $\mathbf{e}$, which—in case of a multivariate Gaussian distribution—implies a mean vector $\boldsymbol{\mu}_{\mathbf{e}}$ and a covariance matrix $\boldsymbol{\Omega}_{\mathbf{e}}$; see Eqs. (6.38) and (6.39). We may estimate this distribution from historical data. However, Yanikoğlu et al. (2015) develops an approach that uses only the original observed data on $\mathbf{e}$; several numerical examples demonstrate the effectiveness of this novel combination of the two approaches originated by Taguchi and Ben-Tal et al. The uncertainty (or "ambiguity") of the estimated mean vector $\boldsymbol{\mu}_{\mathbf{e}}$ and covariance matrix $\boldsymbol{\Omega}_{\mathbf{e}}$ is also considered in Hu et al. (2012), assuming a multivariate normal distribution for the parameters $\mathbf{e}$ of the underlying simulation model and ambiguity sets for $\boldsymbol{\mu}_{\mathbf{e}}$ and $\boldsymbol{\Omega}_{\mathbf{e}}$ with the corresponding worst-case performance.

The examples in Yanikoğlu et al. (2015) include a deterministic simulation of the television example in Myers et al. (2009, p. 512) and a random simulation of a distribution-center example in Shi (2011); details on the latter example are also given in Shi et al. (2014). The latter example has as response the total throughput, and has five decision variables (e.g., number of forklifts) and two environmental variables (e.g., delay probabilities of suppliers); the incomplete second-order polynomial of Eq. (6.37) is fitted. Yanikoğlu et al. (2015) replaces Eq. (6.36) by the following related problem:

$$\min \sigma_w^2 \quad \text{such that } \mu_w \leq T \tag{6.46}$$

where the statistical parameters μ_w and σ_w^2 are based on the historical data (using the phi-divergence criterion). These two examples demonstrate that robust solutions may have better worst-case performance and also better average performance than the nominal solutions have.

6.5 Conclusions

In this chapter we started with basic RSM, which minimizes the expected value of a single response variable in real-life experiments or deterministic simulation. Next we considered RSM in random simulation. We then presented the ASD search direction, which improves the classic steepest descent direction. We also summarized GRSM for simulation with multivariate responses, assuming that one response is to be minimized while all the other responses and deterministic inputs should satisfy given constraints. Furthermore, we discussed the KKT conditions in constrained minimization, and presented a parametric bootstrap procedure for testing these conditions in random simulation. Next we discussed Kriging for optimization. We detailed EGO for unconstrained optimization in deterministic simulation, and KriMP for constrained optimization in random simulation. Finally, we considered robust optimization, using either the linear regression metamodels of RSM or Kriging metamodels; we also briefly discussed Ben-Tal et al.'s approach to robust optimization.

Future research may study the selection of the required number of replications, and the use of replications to estimate the accuracy of the resulting estimated search direction or optimum. Bootstrapping might solve this problem, but more research is needed. Numerical evaluation of the adapted steepest descent method would benefit from more applications in practice. We also see a need for more research on the KKT testing procedure when all local points (not only the center) are replicated and CRN are used; more practical applications are also needed. Various EGO variants and KriMP need more research. In Taguchian robust optimization we may vary the threshold values, to estimate the Pareto frontier; bootstrapping this frontier might enable management to make the final compromise decision—but more research and applications are needed.

Solutions of Exercises

Solution 6.1 $\mathbf{z}_o = (-5,\ 15)$*; also see Angün et al. (2009).*

Solution 6.2 *If* $\mathbf{Z}'\mathbf{Z} = N\mathbf{I}$*, then Eq. (6.5) implies* $\mathbf{C} = \mathbf{I}/N$*. Hence, Eq. (6.7) does not change the steepest descent direction.*

Solution 6.3 *The ratio of two normal variables has a Cauchy distribution so its expected value does not exist; its median does.*

Solution 6.4 $(z_{o1},\ z_{o2}) = (1.24,\ 0.52)$*; also see Angün et al. (2009).*

References

Ajdari A, Mahlooji H (2014) An adaptive hybrid algorithm for constructing an efficient sequential experimental design in simulation optimization. Commun Stat Simul Comput 43:947–968

Alaeddini A, Yang K, Mao H, Murat A, Ankenman B (2013) An adaptive sequential experimentation methodology for expensive response surface optimization—case study in traumatic brain injury modeling. Qual Reliab Eng Int 30(6): 767–793

Alrabghi A, Tiwari A (2015) State of the art in simulation-based optimisation for maintenance systems. Comput Ind Eng (in press)

Andrieu L, Cohen G, Vázquez-Abad FJ (2011) Gradient-based simulation optimization under probability constraints. Eur J Oper Res 212:345–351

Angün ME (2004) Black box simulation optimization: generalized response surface methodology. CentER dissertation series, Tilburg University, Tilburg, Netherlands (also published by VDM Verlag Dr. Müller, Saarbrücken, Germany, 2011)

Angün E (2011) A risk-averse approach to simulation optimization with multiple responses. Simul Model Pract Theory 19:911–923

Angün E, den Hertog D, Gürkan G, Kleijnen JPC (2009) Response surface methodology with stochastic constraints for expensive simulation. J Oper Res Soc 60(6):735–746

Apley DW, Kim J (2011) A cautious approach to robust design with model parameter uncertainty. IIE Trans 43(7):471–482

Ardakani MK, Wulff SS (2013) An overview of optimization formulations for multiresponse surface problems. Qual Reliab Eng Int 29:3–16

Atlason J, Epelman MA, Henderson SG (2008) Optimizing call center staffing using simulation and analytic center cutting-plane methods. Manag Sci 54(2):295–309

Barnes ER (1986) A variation on Karmarkar's algorithm for solving linear programming problems. Math Program 36:174–182

Barton RR, Meckesheimer M (2006) Metamodel-based simulation optimization. In: Simulation. Handbooks in operations research and management science, vol 13. Elsevier/North Holland, Amsterdam, pp 535–574

Bashyam S, Fu MC (1998) Optimization of (s, S) inventory systems with random lead times and a service level constraint. Manag Sci 44:243–256

Bassamboo A, Randhawa RS, Zeevi A (2010) Capacity sizing under parameter uncertainty: safety staffing principles revisited. Manag Sci 56(10):1668–1686

Bates RA, Kenett RS, Steinberg DM, Wynn HP (2006) Achieving robust design from computer simulations. Qual Technol Quant Manag 3(2):161–177

Ben-Tal A, Nemirovski A (1998) Robust convex optimization. Math Oper Res 23(4):769–805

Ben-Tal A, Nemirovski A (2008) Selected topics in robust convex optimization. Math Program 112(1):125–158

Bettonvil BWM, del Castillo E, Kleijnen JPC (2009) Statistical testing of optimality conditions in multiresponse simulation-based optimization. Eur J Oper Res 199(2):448–458

Beyer H, Sendhoff B (2007) Robust optimization—a comprehensive survey. Comput Methods Appl Mech Eng 196(33–34):3190–3218

Binois M, Ginsbourger D, Roustant O (2015) Quantifying uncertainty on Pareto fronts with Gaussian process conditional simulations. Eur J Oper Res 243: 386–394

Boukouvalas A, Cornford D, Stehlík M (2014) Optimal design for correlated processes with input-dependent noise. Comput Stat Data Anal 71:1088–1102

Box GEP (1999) Statistics as a catalyst to learning by scientific method, part II—a discussion. J Qual Technol 31(1):16–29

Box GEP, Wilson KB (1951) On the experimental attainment of optimum conditions. J R Stat Soc Ser B 13(1):1–38

Brekelmans R, Driessen L, Hamers H, den Hertog D (2005) Gradient estimation schemes for noisy functions. J Optim Theory Appl 126(3): 529–551

Broadie M, Du Y, Moallemi CC (2011) Efficient risk estimation via nested sequential simulation. Manag Sci 57:1172–1194

Bull AD (2011) Convergence rates of efficient global optimization algorithms. J Mach Learn Res 12:2879–2904

Chang K-H, Hong J, Wan H (2013) Stochastic trust-region response-surface method (STRONG)—a new response-surface framework for simulation optimization. INFORMS J Comput 25(2):230–243

Chang K-H, Li M-K, Wan H (2014) Combining STRONG with screening designs for large-scale simulation optimization. IIE Trans 46(4):357–373

Chang K-H, Lin G (2015) Optimal design of hybrid renewable energy systems using simulation optimization. Simul Model Pract Theory 52:40–51

Chau M, Fu MC, Qu H, Ryzhov I (2014) Simulation optimization: a tutorial overview and recent developments in gradient-based and sequential allocation methods. In: Tolk A, Diallo SY, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the 2014 winter simulation conference, Savannah, pp 21–35

Chen X, Kim K-K (2014) Stochastic kriging with biased sample estimates. ACM Trans Model Comput Simul 24(2):8:1–8:23

Chevalier C, Ginsbourger D, Bect J, Vazquez E, Picheny V, Richet Y (2014) Fast parallel Kriging-based stepwise uncertainty reduction with application to the identification of an excursion set. Technometrics 56(4): 455–465

Chih M (2013) A more accurate second-order polynomial metamodel using a pseudo-random number assignment strategy. J Oper Res Soc 64: 198–207

Conn AR, Gould NLM, Toint PL (2000) Trust-region methods. SIAM, Philadelphia

Davis E, Ierapetritou M (2009) A kriging based method for the solution of mixed-integer nonlinear programs containing black-box functions. J Glob Optim 43:191–205

Dehlendorff C, Kulahci M, Andersen K (2011) Designing simulation experiments with controllable and uncontrollable factors for applications in health care. J R Stat Soc Ser C (Appl Stat) 60:31–49

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

Dellino G, Meloni C (2013) Quantitative methods to analyze simulation metamodels variability. In: Spitaleri RM (ed) Proceedings of the 11th meeting on applied scientific computing and tools. IMACS series in computational and applied mathematics, vol 17, pp 91–100

Dellino G, Meloni C (eds) (2015) Uncertainty management in simulation-optimization of complex systems. Algorithms and applications. Springer, New York

Dengiz B (2009) Redesign of PCB production line with simulation and Taguchi design. In: Rossetti MD, Hill RR, Johansson B, Dunkin A, Ingalls RG (eds) Proceedings of the 2009 winter simulation conference, Austin, pp 2197–2204

Dykstra RL (1970) Establishing the positive definiteness of the sample covariance matrix. Ann Math Stat 41(6):2153–2154

Echard B, Gayton N, Lemaire M (2011) Ak-mcs: an active learning reliability method combining Kriging and Monte Carlo simulation. Struct Saf 33(2):145–154

Fan S-KS, Huang K-N (2011) A new search procedure of steepest ascent in response surface exploration. J Stat Comput Simul 81(6):661–678

Feyzioğlu O, Pierreval H, Deflandre D (2005) A simulation-based optimization approach to size manufacturing systems. Int J Prod Res 43(2): 247–266

Feng Z, Zhang Q, Tang Q, Yang T, Ma Y (2015) A multiobjective optimization based framework to balance the global exploration and local exploitation in expensive optimization. J Glob Optimi, 61(4):677–694

Figueira G, Almada-Lobo B (2014) Hybrid simulation-optimization methods: a taxonomy and discussion. Simul Model Pract Theory 46:118–134

Forrester AIJ, Jones DR (2008) Global optimization of deceptive functions with sparse sampling. In: 12th AIAA/ISSMO multidisciplinary analysis and optimization conference, Victoria, pp 10–12

Forrester AIJ, Keane AJ (2009) Recent advances in surrogate-based optimization. Prog Aerosp Sci 45(1–3):50–79

Forrester AIJ, Sóbester A, Keane AJ (2008) Engineering design via surrogate modelling; a practical guide. Wiley, Chichester, pp 79–102

Frazier PI (2010) Learning with dynamic programming. In: Cochran JJ, Cox LA, Keskinocak P, Kharoufeh JP, Smith JC (eds) Wiley encyclopedia of operations research and management science. Wiley, New York

Frazier P, Powell W, Dayanik S (2009) The knowledge-gradient policy for correlated normal beliefs. INFORMS J Comput 21:599–613

Fu MC, Bayraksan G, Henderson SG, Nelson BL, Powell WB, Ryzhov IO, Thengvall B (2014) Simulation optimization: a panel on the state of the art in research and practice. In: Tolk A, Diallo SY, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the 2014 winter simulation conference, Savannah, pp 3696–3706

Gabrel V, Murat C, Thiele A (2014) Recent advances in robust optimization: an overview. Eur J Oper Res 235(3):471–483

Gano SE, Renaud JE, Martin JD, Simpson TW (2006) Update strategies for Kriging models for using in variable fidelity optimization. Struct Multidiscip Optim 32(4):287–298

Gan G, Lin XS (2015) Valuation of large variable annuity portfolios under nested simulation: a functional data approach. Insurance: Math Econ 62:138–150

Gill PE, Murray W, Wright MH (2000) Practical optimization, 12th edn. Academic, London

Ginsbourger D, Le Riche R, Carraro L (2010) Kriging is well-suited to parallelize optimization. In: Tenne Y, Goh C-K (eds) Computational intelligence in expensive optimization problems. Springer, Berlin, pp 131–162

Gordy MB, Juneja S (2010) Nested simulation in portfolio risk measurement. Manag Sci 56(11):1833–1848

Gorissen D (2010) Grid-enabled adaptive surrogate modeling for computer aided engineering. Ph. D. dissertation Ghent University, Ghent, Belgium

Gosavi A (2015) Simulation-based optimization: parametric optimization techniques and reinforcement learning, 2nd edn. Springer, Boston

Gramacy RB, Gray GA, Le Digabel S, Lee HKH, Ranjan P, Wells G, Wild SM (2015) Modeling an augmented Lagrangian for blackbox constrained optimization. Technometrics (in press)

Gramacy RB, Lee HKH (2010) Optimization under unknown constraints. Bayesian Stat 9:1–18

Grubler A, Ermoliev Y, and Kryazhimskiy A (2015) Coping with uncertainties examples of modeling approaches at IIASA. Technological Forecasting and Social Change (in press)

Hamarat C, Kwakkel JH, Pruyt E, Loonen ET (2014) An exploratory approach for adaptive policymaking by using multi-objective robust optimization. Simul Model Pract Theory 46:25–39

Homem-de-Mello T, Bayraksan G (2014) Monte Carlo sampling-based methods for stochastic optimization. Surv Oper Res Manag Sci 19(1): 56–85

Hong LJ, Luo J, Nelson BL (2015) Chance constrained selection of the best. INFORMS J Comput 27(2):317–334

Hu Z, Cao J, Hong LJ (2012) Robust simulation of global warming policies using the DICE model. Manag Sci 58(12):2190–2206

Huang D, Allen TT, Notz W, Zheng N (2006) Global optimization of stochastic black-box systems via sequential Kriging meta-models. J Glob Optim 34:441–466

Huang Y, Hsieh C-Y (2014) Influence analysis in response surface methodology. J Stat Plan Inference 147:188–203

Huerta A, Elizondo M (2014) Analysis of scientific collaboration patterns in co-authorship network of simulation-optimization of supply chains. Simul Model Pract Theory 46:135–148

Jala M, Lévy-Leduc C, Moulines É, Conil E, Wiart J (2014) Sequential design of computer experiments for the assessment of fetus exposure to electromagnetic fields. Technometrics (in press)

Jalali H, van Nieuwenhuyse I (2014) Evaluation of Kriging-based methods for simulation optimization with homogeneous noise. In: Tolk A, Diallo SY, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the 2014 winter simulation conference, pp 4057–4058

Jalali H, Van Nieuwenhuyse I (2015, accepted) Simulation optimization in inventory replenishment: a classification. IIE Trans

Janusevskis J, Le Riche R (2013) Simultaneous kriging-based estimation and optimization of mean response. J Glob Optim 55(2):313–336

Jones DR, Schonlau M, Welch WJ (1998) Efficient global optimization of expensive black-box functions. J Glob Optim 13:455–492

Joshi S, Sherali HD, Tew JD (1998) An enhanced response surface methodology (RSM) algorithm using gradient deflection and second-order search strategies. Comput Oper Res 25(7/8):531–541

Kamiński B (2015) A method for updating of stochastic Kriging metamodels. Eur J Oper Res (accepted)

Kasaie P, Kelton WD (2013) Simulation optimization for allocation of epidemic-control resources. IIE Trans Healthc Syst Eng 3(2):78–93

Kasaie P, Vaghefi A, Naieni G (2009) Optimal resource allocation for control of epidemics: an agent based simulation approach. Working Paper, Dept. of Industrial Engineering, Iran University of Science & Technology, Tehran, 16844, Iran

Kelton WD, Sadowski RP, Sturrock DT (2007) Simulation with Arena, 4th edn. McGraw-Hill, Boston

Kenett R, Steinberg D (2006) New frontiers in design of experiments. Qual Progress 61–65

Khuri AI, Mukhopadhyay S (2010) Response surface methodology. Wiley Interdiscip Rev Comput Stat 2:128–149

Kleijnen JPC (1975) Statistical techniques in simulation, part II. Dekker, New York

Kleijnen JPC (1993) Simulation and optimization in production planning: a case study. Decis Support Syst 9:269–280

Kleijnen JPC (2008) Design and analysis of simulation experiments. Springer, New York

Kleijnen JPC (2014) Response surface methodology. In: Fu MC (ed) Handbook of simulation optimization. Springer, New York

Kleijnen JPC, Den Hertog D, Angün E (2004) Response surface methodology's steepest ascent and step size revisited. Eur J Oper Res 159:121–131

Kleijnen JPC, Den Hertog D, Angün E (2006) Response surface methodology's steepest ascent and step size revisited: correction. Eur J Oper Res 170:664–666

Kleijnen JPC, Gaury EGA (2003) Short-term robustness of production-management systems: a case study. Eur J Oper Res 148(2):452–465

Kleijnen JPC, Mehdad E (2013) Conditional simulation for efficient global optimization. In: Proceedings of the 2013 winter simulation conference, Washington, pp 969–979

Kleijnen JPC, Pierreval H, Zhang J (2011) Methodology for determining the acceptability of system designs in uncertain environments. Eur J Oper Res 209(2):176–183

Kleijnen JPC, Sargent RG (2000) A methodology for the fitting and validation of metamodels in simulation. Eur J Oper Res 120(1):14–29

Kleijnen JPC, Van Beers WCM, van Nieuwenhuyse I (2010) Constrained optimization in simulation: a novel approach. Eur J Oper Res 202: 164–174

Kleijnen JPC, Van Beers W, Van Nieuwenhuyse I (2012) Expected improvement in efficient global optimization through bootstrapped Kriging. J Glob Optim 54:59–73

Kleijnen JPC, Wan J (2007) Optimization of simulated systems: OptQuest and alternatives. Simul Model Pract Theory 15:354–362

Koch P, Wagner T, Emmerich MTM, Bäck T, Konen W (2015) Efficient multi-criteria optimization on noisy machine learning problems. Appl Soft Comput (in press)

Kolaiti E, Koukouvinos C (2006) On the use of three level orthogonal arrays in robust parameter design. Stat Probab Lett 76(3):266–273

Koziel S, Bekasiewicz A, Couckuyt I, Dhaene T (2014) Efficient multi-objective simulation-driven antenna design using co-Kriging. IEEE Trans Antennas Propag 62(11):5901–5915

Lan H, Nelson BL, Staum J (2010) A confidence interval procedure for expected shortfall risk measurement via two-level simulation. Oper Res 58(5):1481–1490

Law AM (2015) Simulation modeling and analysis, 5th edn. McGraw-Hill, Boston

Lee KH, Park GJ (2006) A global robust optimization using Kriging based approximation model. J Jpn Soc Mech Eng 49:779–788

Lee LH, Chew EP, Frazier PI, Jia Q-S, Chen C-H (2013) Foreword: advances in simulation optimization and its applications. IIE Trans 45(7):683–684

Lee S, Nelson BL (2014) Bootstrap ranking & selection revisited. In: Tolk A, Diallo SY, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the 2014 winter simulation conference, Savannah, pp 3857–3868

Leijen MCF (2011) Response surface methodology for simulation optimization of a packaging line. Master's thesis, Eindhoven University of Technology, Department of Mechanical Engineering, Systems Engineering Group, Eindhoven

Mahdavi I, Shirazi B, Solimanpur M (2010) Development of a simulation-based decision support system for controlling stochastic flexible job shop manufacturing systems. Simul Model Pract Theory 18:768–786

Marzat J, Walter E, Piet-Lahanie H (2013) Worst-case global optimization of black-box functions through Kriging and relaxation. J Glob Optim 55:707–727

Mehdad E, Kleijnen JPC (2015) Classic Kriging versus Kriging with bootstrapping or conditional simulation: classic Kriging's robust confidence intervals and optimization. J Oper Res Soc (in press)

Mehdad E, Kleijnen JPC (2014) Global optimization for black-box simulation through sequential intrinsic Kriging. CentER Discussion Paper 2014-063, Tilburg University, Tilburg, Netherlands

Meloni C, Dellino G (eds) (2015) Uncertainty management in simulation-optimization of complex systems; algorithms and applications. Springer

Miller GA (1956) The magical number seven plus or minus two: some limits on our capacity for processing information. Psychol Rev 63:81–97

Montevechi JAB, de Almeida Filho RG, Paiva AP, Costa RFS, and A.L. Medeiros (2010) Sensitivity analysis in discrete-event simulation using fractional factorial designs. J Simul 4(2):128–142

Morales-Enciso S and Branke J (2015) Tracking global optima in dynamic environments with efficient global optimization. Eur J Oper Res 242(3):744–755

Müller J, Shoemaker CA (2014) Influence of ensemble surrogate models and sampling strategy on the solution quality of algorithms for computationally expensive black-box global optimization problems. J Glob Optim 60(2):123–144

Myers RH, Khuri AI, Carter WH (1989) Response surface methodology: 1966–1988. Technometrics 31(2):137–157

Myers RH, Montgomery DC, Anderson-Cook CM (2009) Response surface methodology: process and product optimization using designed experiments, 3rd edn. Wiley, New York

Nair VN (ed) (1992) Taguchi's parameter design: a panel discussion. Technometrics 34(2):127–161

Nakayama H, Yun Y, Yoon M (2009) Sequential approximate multiobjective optimization using computational intelligence. Springer, Berlin, pp 133–141

Ng SH, Xu K, Wong WK (2007) Optimization of multiple response surfaces with secondary constraints for improving a radiography inspection process. Qual Eng 19(1):53–65

Oden JT (2006) Revolutionizing engineering science through simulation. National Science Foundation (NSF), Blue Ribbon Panel on Simulation-Based Engineering Science

Park G-J, Lee T-H, Lee KH, Hwang K-H (2006) Robust design: an overview. AIAA J 44(1):181–191

Pasupathy R, Ghosh S (2014) Simulation optimization: a concise overview and implementation guide. INFORMS Tutorials in Operations Research, pp 122–150. http://pubsonline.informs.org/doi/book/10.1287/educ.2014#Chapters

Picheny V, Ginsbourger D, Richet Y, Caplin G (2013a) Quantile-based optimization of noisy computer experiments with tunable precision (including comments and rejoinder). Technometrics 55(1):1–36

Picheny V, Wagner T, Ginsbourger D (2013b) A benchmark of kriging-based infill criteria for noisy optimization. Struct Multidiscip Optim 48:607–626

Preuss M, Wagner T, Ginsbourger D (2012) High-dimensional model-based optimization based on noisy evaluations of computer games. In: Hamadi Y, Schoenauer M (eds) Learning and intelligent optimization: 6th international conference (LION 6), Paris. Springer, Berlin, pp 145–159

Quan N, Yin J, Ng SH, Lee LH (2013), Simulation optimization via kriging: a sequential search using expected improvement with computing budget constraints. IIE Trans 45:763–780

Qu H, Ryzhov IO, Fu MC, Ding Z (2015) Sequential selection with unknown correlation structures. Oper Res 63(4):931–948

Rashid K, Bailey, WJ Couet B, and Wilkinson D (2013) An efficient procedure for expensive reservoir-simulation optimization under uncertainty. SPE Econ Manage 5(4):21–33

Razavi S, Tolson BA, Burn DH (2012) Review of surrogate modeling in water resources. Water Resour Res 48, W07401:1–322

Regis RG (2014) Locally-biased efficient global optimization using Kriging metamodels Working paper, Department of Mathematics, Saint Joseph's University, Philadelphia

Rikards R, Auzins J (2002) Response surface method for solution of structural identification problems. In: Fourth international conference on inverse problems in engineering, Rio de Janeiro

Rosen SC, Harmonosky CM, Traband MT (2008) Optimization of systems with multiple performance measures via simulation: survey and recommendations. Comput Ind Eng 54(2):327–339

Roustant O, Ginsbourger D, Deville Y (2012) DiceKriging, DiceOptim: two R packages for the analysis of computer experiments by Kriging-based metamodeling and optimization. J Stat Softw 51(1):1–55

Safizadeh MH (2002) Minimizing the bias and variance of the gradient estimate in RSM simulation studies. Eur J Oper Res 136(1):121–135

SakallıÜS, Baykoç ÖF (2011) An optimization approach for brass casting blending problem under aletory and epistemic uncertainties. Int J Prod Econ 133(2):708–718

Salemi P, Nelson BL, Staum J (2014) Discrete optimization via simulation using Gaussian Markov random fields. In: Tolk A, Diallo SY, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the 2014 winter simulation conference, Savannah, pp 3809–3820

Saltelli A, Ratto M, Tarantola S, Campolongo F (2005) Sensitivity analysis of chemical models. Chem Rev 105(7):2811–2827

Samuelson D (2010) When close is better than optimal: combining simulation and stochastic optimization for better risk management. OR/MS Today 37(6):38–41

Santos MI, Santos PM (2011) Construction and validation of distribution-based regression simulation metamodels. J Oper Res Soc 62:1376–1384

Sasena MJ, Papalambros P, Goovaerts P (2002) Exploration of metamodeling sampling criteria for constrained global optimization. Eng Optim 34(3):263–278

Scott W, Frazier P, Powell W (2011) The correlated knowledge gradient for simulation optimization of continuous parameters using Gaussian process regression. SIAM J Optim 21(3):996–1026

Scott WR, Powell WB, Simao HP (2010) Calibrating simulation models using the knowledge gradient with continuous parameters. In: Proceedings of the 2010 winter simulation conference, Baltimore, pp 1099–1109

Shang JS, Li S, Tadikamalla P (2004) Operational design of a supply chain system using the Taguchi method, response surface methodology, simulation, and optimization. Int J Prod Res 42(18):3823–3849

Shahraki AF, Noorossana R (2014) Reliability-based robust design optimization: a general methodology using genetic algorithm. Comput Ind Eng 74:199–207

Shi W (2011) Design of pre-enhanced cross-docking distribution center under supply uncertainty: RSM robust optimization method. Working Paper, Huazhong University of Science & Technology, China

Shi W, Shang J, Liu Z, Zuo X (2014) Optimal design of the auto parts supply chain for JIT operations: sequential bifurcation factor screening and multi-response surface methodology. Eur J Oper Res 236(2):664–676

Shin S, Samanlioglu F, Cho BR, Wiecek MM (2011) Computing trade-offs in robust design: perspectives of the mean squared error. Comput Ind Eng 60(2):248–255

Simar L, Wilson PW (1998) Sensitivity analysis of efficiency scores: how to bootstrap in nonparametric frontier models. Manag Sci 44(1):49–61

Simon HA (1956) Rational choice and the structure of the environment. Psychol Rev 63(2):129–138

Simpson TW, Booker AJ, Ghosh D, Giunta AA, Koch PN, Yang R-J (2004) Approximation methods in multidisciplinary analysis and optimization: a panel discussion. Struct Multidiscip Optim 27(5):302–313

Stinstra E, den Hertog D (2008) Robust optimization using computer experiments. Eur J Oper Res 191(3):816–837

Sun L, Hong LJ, Hu Z (2014) Balancing exploitation and exploration in discrete optimization via simulation through a Gaussian process-based search. Oper Res 62(6):1416–1438

Sun Y, Apley DW, Staum J (2011) Efficient nested simulation for estimating the variance of a conditional expectation. Oper Res 59(4):998–1007

Svenson JD, Santner TJ (2010) Multiobjective optimization of expensive black-box functions via expected maximin improvement. The Ohio State University, Columbus, Ohio

Taddy MA, Lee HKH, Gray GA, Griffin JD (2009) Bayesian guided pattern search for robust local optimization. Technometrics 5(4):389–401

Taguchi G (1987) System of experimental designs, vols 1 and 2. UNIPUB/ Krauss International, White Plains

Tajbakhsh S, del Castillo E, Rosenberger JL (2013) A fully Bayesian approach to the efficient global optimization algorithm. Working Paper, Pennsylvania State University

Tan MHY (2014a) Robust parameter design with computer experiments using orthonormal polynomials. Technometrics (in press)

Tan MHY (2014b) Stochastic polynomial interpolation for uncertainty quantification with computer experiments. Technometrics (in press)

Tenne Y, Goh C-K (eds) (2010) Computational intelligence in expensive optimization problems. Springer, Berlin

Tong C, Sun Z, Zhao Q, Wang Q, Wang S (2015) A hybrid algorithm for reliability analysis combining Kriging and subset simulation importance sampling. J Mech Sci Technol 29(8):3183–3193

Ur Rehman S, Langelaar M, van Keulen F (2014) Efficient Kriging-based robust optimization of unconstrained problems. J Comput Sci (in press)

Van den Bogaard W, Kleijnen JPC (1977) Minimizing waiting times using priority classes: a case study in response surface methodology. Discussion Paper FEW 77.056. http://arno.uvt.nl/show.cgi?fid=105001. Accessed 12 Mar 2014

Van der Herten J, Couckuyt I, Deschrijver D, Dhaene T (2015) A fuzzy hybrid sequential design strategy for global surrogate modeling of high-dimensional computer experiments. SIAM J Sci Comput 37(2):A1020–A1039

Vazquez E, Bect J (2010) Convergence properties of the expected improvement algorithm with fixed mean and covariance functions. J Stat Plan Inference 140(11):3088–3095

Viana FAC, Haftka RT, Watson LT (2013) Efficient global optimization algorithm assisted by multiple surrogate techniques. J Glob Optim 56(2):669–689

Villemonteix J, Vazquez E, Sidorkiewicz M, Walter E (2009a) Global optimization of expensive-to-evaluate functions: an empirical comparison of two sampling criteria. J Glob Optim 43:373–389

Villemonteix J, Vazquez E, Walter E (2009b) An informational approach to the global optimization of expensive-to-evaluate functions. J Glob Optim 44(4):509–534

Wagner T (2013) Planning and multi-objective optimization of manufacturing processes by means of empirical surrogate models. Doctoral dissertation, Technische Universität Dortmund, Dortmund, Germany

Wan J, Li L (2008) Simulation for constrained optimization of inventory system by using Arena and OptQuest. In: 2008 international conference on computer science and software engineering (CSSE 2008). IEEE, Wakefield, MA, pp 202–205

Wiebenga JH (2014) Robust design and optimization of forming processes. Ph.D. thesis, University of Twente, Enschede, Netherlands

Wiesemann W, Kuhn D, Sim M (2014) Distributionally robust convex optimization. Oper Res (in press)

Williams BJ, Santner TJ, Notz WI, Lehman JS (2010) Sequential design of computer experiments for constrained optimization. In: Kneib T, Tutz G (eds) Festschrift for Ludwig Fahrmeir, Springer, Berlin, pp 449–471

Wu CFJ, Hamada M (2009) Experiments; planning, analysis, and parameter design optimization, 2nd edn. Wiley, New York

Xu J, Huang E, Chen C-H, Lee LH (2015) Simulation optimization: a review and exploration in the new era of cloud computing and big data. Asia Pacific J Oper Res (in press)

Yalçinkaya, Ö, Bayhan GM (2009) Modelling and optimization of average travel time for a metro line by simulation and response surface methodology. Eur J Oper Res 196(1):225–233

Yanikoğlu İ, den Hertog D, and Kleijnen JPC (2015), Robust dual response optimization. IIE Trans (in press)

Yarotsky D (2013) Examples of inconsistency in optimization by expected improvement. J Glob Optim 56, pp. 1773–1790

Yin H, Fang H, Xiao Y, Wen G, Qing Q (2015) Multi-objective robust optimization of foam-filled tapered multi-cell thin-walled structures. Struct Multidiscip Optim (in press)

Ye W, You F (2015) A fast simulation-based optimization method for inventory control of general supply chain networks under uncertainty American control conference, Palmer House Hilton, Chicago, 1–3 July 2015, pp 2001–2006

Zazanis MA, Suri R (1993) Convergence rates of finite-difference sensitivity estimates for stochastic systems. Oper Res 41(4):694–703

Zhang J, Ma Y (2015) Stochastic Kriging-assisted multi-objective simulation optimization and uncertainty analysis. Simul: Trans Soc Model Simul Int (in press)

Zhou E, Bhatnagar S, Chen X (2014) Simulation optimization via gradient-based stochastic search. In: Tolk A, Diallo SY, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the 2014 winter simulation conference, Savannah, pp 3869–3879