



Discussion Paper

No. 2007–45

STATISTICAL TESTING OF OPTIMALITY CONDITIONS IN MULTIRESPONSE SIMULATION-BASED OPTIMIZATION

By Bert Bettonvil, Enrique del Castillo, Jack P.C. Kleijnen

June 2007

This is a revised version of CentER Discussion Paper
No. 2005-81

June 2005

ISSN 0924-7815

Statistical testing of optimality conditions in multiresponse simulation-based optimization

Bert Bettonvil 1) Enrique del Castillo 2) Jack P.C. Kleijnen 1)

1) *CentER, Department of Information Systems and Management, Tilburg University, Postbox 90153, 5000 LE Tilburg, Netherlands*

2) *Department of Industrial and Manufacturing Engineering and Department of Statistics, The Pennsylvania State University, 310 Leonhard building, University Park PA 16802, USA*

Abstract

This paper studies simulation-based optimization with multiple outputs. It assumes that the simulation model has one random objective function and must satisfy given constraints on the other random outputs. It presents a statistical procedure for testing whether a specific input combination (proposed by some optimization heuristic) satisfies the Karush-Kuhn-Tucker (KKT) first-order optimality conditions. The paper focuses on "expensive" simulations, which have small sample sizes. The paper applies the classic t test to check whether the specific input combination is feasible, and whether any constraints are binding; it applies bootstrapping (resampling) to test the estimated gradients in the KKT conditions. The new methodology is applied to three examples, which gives encouraging empirical results.

Key words: Stopping rule; metaheuristics; response surface methodology; design of experiments

JEL: C0, C1, C9, C15, C44, C61

1 Introduction

An important practical and academic type of problem is the search for the optimal input for a random simulation model. This problem area is known as *simulation optimization*. There are many methods to solve this problem; see the references in [15]. Unfortunately, all these methods are *heuristic*; i.e., they propose solutions that are not necessarily truly optimal in "small" experiments. Such small experiments are the rule in *expensive* simulations (e.g., a

user reports that a single run of a specific simulation model requires three days on a powerful PC). In this article, we give these methods "the benefit of the doubt"; i.e., our null-hypothesis is that the solution given by the heuristic, is indeed optimal. Next, we derive a statistical procedure to test this hypothesis.

Before we proceed, we give some more terminology and synonyms, because simulation optimization is studied in many disciplines—each with its own terminology. So, an input combination is also called a factor combination, input point, iterate, or scenario. Multiple simulation outputs or responses are also called multivariate. Replicates mean that a particular input combination is simulated several times, using nonoverlapping streams of pseudo-random numbers (PRN) so identically and independently distributed (IID) output observations result.

It is standard in the mathematical statistics literature on hypothesis testing that the users prespecify an upper bound for the *type-I error*, which is also known as the α error. If the simulation were not expensive, then the analysts could also constrain the *type-II* or β error by simulating enough replicates. In case of expensive simulation, however, we only desire that our procedure give a type-II error probability that decreases as the point tested moves away from the (unknown) true optimal point; i.e., we desire that the statistical power function of our test procedure increases as the alternative hypothesis deviates more from the null-hypothesis (and the power is at least $1 - \alpha$ when the null-hypothesis holds). Part of our procedure uses the classic t test, so it is obvious that this test has the desired behavior. Another part, however, uses a novel bootstrap test for the gradients, so we investigate its behavior empirically (and find that the power function has the desired shape; see Section 5).

We focus on *parametric bootstrapping* (assuming Gaussian distributions) because this bootstrap enables a minimum number of simulation replicates (e.g., only the local central point is replicated a few times, as we shall see). Nevertheless, we also briefly discuss *distribution-free bootstrapping*, which assumes that all simulated points are replicated a few times. We do not consider asymptotic tests, but refer to [2] and [32]. ([10] also uses distribution-free bootstrapping to test the optimality of a candidate solution, but that article addresses so-called "stochastic programs with recourse", which is very different from simulation.)

In practice, simulation models have *multiple outputs* (e.g., an inventory simulation may have as outputs the estimated partial cost and service rate). We focus on the following mathematical programming formulation for such simulations: minimize the expected value of one random objective (or goal) function, while satisfying prespecified constraints for the expected values of all the other random outputs. For the deterministic variant of this formulation, the *Karush-Kuhn-Tucker* (KKT) first-order optimality conditions are well-known necessary but not sufficient conditions; see [9]. For random outputs

these conditions are the focus of our article.

We examine the behavior of our procedure through three numerical examples. We focus on Example 1, which assumes second-order polynomial Input/Output (I/O) functions for each type of output. Example 2 simulates the well-known (s, S) inventory model, assuming that out-of-stock penalty costs are so hard to quantify that instead a prespecified service (or fill rate) constraint must be satisfied; see [18]. Example 3 integrates this (s, S) inventory model with a production model; see [4]. The first two examples use parametric bootstrapping for the gradients that are estimated using a local experimental design with only the center replicated. The third example uses distribution-free bootstrapping for the gradients that are estimated using a global Latin hypercube sampling (LHS) design with all points replicated between five and seven times, analyzed through a global Kriging metamodel.

We assume that the optimum occurs when at least one constraint is *binding* (or "active"); i.e., the optimum input point lies at the border of the feasible area. If the optimum occurs inside the feasible area, then the gradient of the objective function is tested to be zero; see the classic Response Surface Methodology (RSM) textbook [25] and the many RSM references in [7] and [15], and also the unconstrained problem solved via the score function method in [12].

Simulation optimization heuristics can be distinguished in various ways; e.g., *gradient based* methods (e.g., RSM) and *derivative-free* methods (e.g., the commercial software called OptQuest; see Section 5.2). We claim that factor combinations pronounced to be optimal by some heuristic, should be subjected to our KKT test. Obviously, a derivative-free method must then be augmented with a method for estimating gradients; see Section 3.

Moreover, the gradient estimation method should estimate the *distribution* of the estimated gradients, because we use this estimated density function (EDF) for bootstrapping. In general, the bootstrap can estimate the distribution of any statistic—provided that its density function is continuous; see the seminal book on bootstrapping [8] or the more recent publication [5]. More specifically, we wish to test the hypothesis that a given input combination satisfies the KKT conditions; [22] discusses bootstrap hypothesis testing, in general.

We organize the remainder of this article as follows. Section 2 formalizes our simulation optimization problem as a constrained nonlinear random optimization problem, and gives the KKT conditions. Section 3 derives the distribution of the estimated gradients that feature in the KKT conditions. Section 4 develops a procedure for testing whether the center point of a given local area satisfies the KKT conditions. Section 5 uses three examples to demonstrate the performance of the procedure. Section 7 gives conclusions and a research

agenda. The article concludes with 34 references, enabling further study.

2 Mathematical programming formulation

Before we give the general problem formulation, we introduce Example 1. This example has two deterministic inputs, z_1 and z_2 . It has three random outputs, w_0 , w_1 , and w_2 where $E(w_0)$ is the goal output and $E(w_1)$ and $E(w_2)$ must meet prespecified constraints; see Figure 1. This figure displays the I/O functions, which are the following second-order polynomials:

$$\begin{aligned} \text{minimize} \quad & E(w_0) = E[5(z_1 - 8)^2 + (z_2 + 8)^2 + e_0] \\ \text{subject to} \quad & E(w_1) = E[(z_1 - 3)^2 + z_2^2 + z_1 z_2 + e_1] \leq 4 \\ & E(w_2) = E[z_1^2 + 3(z_2 + 1.061)^2 + e_2] \leq 9 \end{aligned} \tag{1}$$

where e_0 , e_1 , and e_2 have zero means. The figure displays three "iso" goal functions, which are the set of input combinations with the same goal value—namely, 96, 76, and 66 respectively. The figure also shows the two constraints when they are "binding"; i.e., in $(1) \leq$ becomes $=$. The optimal point is A (in practice, this is the point to be found by the simulation optimization heuristic). The figure also displays three suboptimal points, namely B, C, and D. The points A through C lie on the same binding constraint; D lies on a different binding constraint. The figure also shows the gradients of the goal function and the binding constraint at these four points; these gradients are perpendicular to the local tangent lines (which are also displayed).

In practice (also see Examples 2 and 3), the simulation I/O functions are unknown, because (i) the (complicated) simulation model is an *implicit* function, and (ii) the output of the discrete-event simulation is a *random* variable, so a run of the simulation model gives only an observation on (or realization of) the multivariate output. The figure displays only points on the *border* of the feasible area. In practice, an experimental design determines the input combinations to be simulated; executing this design means running the simulation model, which gives observations on the random outputs. These observations may be used to test whether the corresponding input combinations are feasible and whether any of these feasible combinations is optimal (see below).

Our general problem formulation is as follows. The simulation model has $k \geq 1$ inputs or decision variables $\mathbf{z} = (z_1, \dots, z_k)'$. This model has $r \geq 2$ outputs w_h ($h = 0, 1, \dots, r - 1$) where w_0 denotes the goal output, which is to be minimized. This results in the following *constrained nonlinear random optimization*

problem:

$$\begin{aligned} & \text{minimize} && E(w_0|\mathbf{z}) \\ & \text{subject to} && E(w_{h'}|\mathbf{z}) \geq a_{h'} \text{ with } h' = 1, \dots, r-1. \end{aligned} \tag{2}$$

Let \mathbf{z}^0 denote a local minimizer for the deterministic variant of our problem. The *KKT conditions* for \mathbf{z}^0 are then

$$\begin{aligned} \beta_0 &= \sum_{h \in A(\mathbf{z}^0)} \lambda_h^0 \beta_h \\ \lambda_h^0 &\geq 0 \\ h &\in A(\mathbf{z}^0) \end{aligned} \tag{3}$$

where β_0 denotes the k -dimensional vector with the gradient of the goal function; $A(\mathbf{z}^0)$ is the index set with the indices of the constraints that are binding at \mathbf{z}^0 ; λ_h^0 is the Lagrange multiplier for binding constraint h ; β_h is the gradient of the output in that binding constraint (in Figure 1, there is only one binding constraint at A through D). The KKT conditions imply that the gradient of the objective can be expressed as a nonnegative linear combination of the gradients of the binding constraints, at \mathbf{z}^0 . (Moreover, there is a certain constraint qualification that is relevant when there are nonlinear constraints in the problem; see [9], 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 \mathbf{z}^0 be linearly independent.)

In Figure 1, A satisfies the KKT conditions; B has two gradients that point in different but similar directions—and so does C. However, D has two gradients that point in completely different directions.

Unfortunately, in *random* simulation the gradients must be estimated. Moreover, the slacks of the constraints must be estimated, to check which constraints are binding. This estimation changes our problem into a problem of nonlinear statistics—discussed next.

3 Estimation of gradients in random simulation

There are many methods for the estimation of the gradients in the KKT conditions (again see Section 1). Whatever method is used, a so-called optimal point should be subjected to the KKT test. Obviously, a derivative-free method must then be augmented with a method for estimating gradients.

In Section 5.3, we shall fit Kriging models, which also enable the estimation of gradients. In the present section, however, we focus on low-order polynomials per output—fitted locally through ordinary least squares (OLS)—using a so-called resolution-III (R-III) design augmented with a center point to fit first-order polynomials or a central composite design (CCD) to fit second-order polynomials. R-III and CCD will be defined below.

OLS implies that "fitting errors" occur; see e in (1). OLS assumes that these errors are white noise; i.e., e is normally (Gaussian), identically, and independently distributed (NIID) with zero mean and "constant" variance (say) σ_e^2 . This variance remains constant within each local area, but not when these areas change; e.g., the local areas centered around A through D in Figure 1 may have different variances.

By definition, a *R-III* design gives unbiased estimators of the parameters of a first-order polynomial, assuming such a polynomial is a valid approximation. This design requires only $n \approx k + 1$ input combinations; e.g., the $n = 8$ combinations of a 2^{7-4} design suffice if $4 \leq k \leq 7$. The design is D-optimal; i.e., it minimizes the determinant of $\mathbf{cov}(\hat{\boldsymbol{\beta}})$, the covariance matrix of $\hat{\boldsymbol{\beta}}$ where $\hat{\boldsymbol{\beta}} = (\hat{\beta}_0, \hat{\beta}_1, \dots, \hat{\beta}_k)'$ denotes the OLS estimators of these parameters. The estimated gradient is $\hat{\boldsymbol{\beta}}_{-0} = (\hat{\beta}_1, \dots, \hat{\beta}_k)'$.

One part of a CCD is a two-level factorial that may be a fractional factorial—provided this fractional has a resolution at least V. A *resolution-V* (R-V) design gives unbiased OLS estimators of β_0 and β_j ($j = 1, \dots, k$), and the two-factor interactions (or cross-products) $\beta_{j;j'}$ ($j < j' = 2, \dots, k$)—if all other effects are zero. The number of regression parameters (say) q is now $1 + k + k(k - 1)/2$. Obviously, the design must satisfy the condition $n \geq q$. Our three examples have only $k = 2$ factors, so part of the CCD is the following 2^2 design where row j ($j = 1, 2$) shows the value of factor j in combination i ($i = 1, \dots, 4$):

$$\mathbf{D}' = \begin{bmatrix} -1 & +1 & -1 & +1 \\ -1 & -1 & +1 & +1 \end{bmatrix}. \quad (4)$$

The other part of the *CCD* augments the first part such that the purely quadratic effects $\beta_{j;j}$ can also be estimated; e.g., (4) is augmented to

$$\mathbf{D}' = \begin{bmatrix} -1 & +1 & -1 & +1 & 0 & -c & +c & 0 & 0 \\ -1 & -1 & +1 & +1 & 0 & 0 & 0 & -c & +c \end{bmatrix} \quad (5)$$

where c is some constant. In general, a CCD adds the central point $\mathbf{0} = (0, \dots, 0)'$ and $2k$ "axial" points with the negative axial point $x_j = -c$ and all other $k - 1$ factors fixed at the center and the positive axial point with $x_j = +c$ and

$$x_{j'} = 0.$$

Obviously, the gradients estimated through first-order and second-order polynomials have different values, because they are estimated from different I/O data. In a small local region, a first-order polynomial may give an accurate approximation; our procedure will test for lack-of-fit (see below).

We assume that the multiple simulation outputs are *multivariate normal* if we use parametric bootstrapping, as we shall do for Examples 1 and 2. For Example 3, however, we do not assume normality because we shall use *distribution-free bootstrapping*. The normality assumption is necessary when simulation is so expensive that replication is to be avoided as much as possible; i.e., the parameters of the multivariate distribution can be estimated if only the local center point is replicated a few times. The normality assumption may be realistic if the output is an average; e.g., [13] discusses the *Functional Central Limit Theorem*.

Even if the multiple simulation outputs are correlated and nonnormally distributed, OLS still gives the *best linear unbiased estimator* (BLUE)—because all r outputs use the same input combinations; see [28]. So we use

$$\widehat{\beta}_h = (\mathbf{Z}'\mathbf{Z})^{-1}\mathbf{Z}'\mathbf{w}_h \text{ with } h = 0, \dots, r-1, \quad (6)$$

where the total number of simulation runs is $N = \sum_{i=1}^n m_i$ where $m_i = 1$ for all points of the R-III design or the CCD except for the center point, which has $m_i > 1$; n denotes the number of different input points; \mathbf{Z} denotes the $N \times q$ matrix of explanatory (regression) variables, and \mathbf{w}_h the N -dimensional vector with the simulation output of type h . The inversion in (6) requires $n \geq q$.

\mathbf{Z} is completely determined by $\mathbf{D} = (d_{ij})$, the design matrix for the k inputs because we use the linear transformation that is standard in design of experiments (DOE):

$$d_{ij} = \frac{z_{ij} - \bar{z}_j}{(u_j - l_j)/2} \text{ with } i = 1, \dots, n \text{ and } j = 1, \dots, k, \quad (7)$$

where \bar{z}_j denotes the average value of the original (nonstandardized) input j , and $(u_j - l_j)$ is the range of the original input j in the local experiment. The intercept β_0 implies that we include a column with n one's in \mathbf{Z} . In a second-order polynomial, the interactions and purely quadratic effects imply that we add $k(k-1)/2 + k$ columns; e.g., $k = 2$ implies the following standardized matrix of explanatory variables (say) \mathbf{X} (its columns 2 and 3 are identical to

\mathbf{D} with \mathbf{D}' defined in (5)):

$$\mathbf{X}' = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -1 & 1 & -1 & 1 & 0 & -c & 0 & c & 0 \\ -1 & -1 & 1 & 1 & 0 & 0 & -c & 0 & c \\ 1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & c^2 & 0 & c^2 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & c^2 & 0 & c^2 \end{bmatrix}. \quad (8)$$

Besides the point estimates $\widehat{\beta}_{h;-0}$, our parametric bootstrap needs the estimated covariance matrix of these estimates. Our designs (either a R-III design augmented with the center point, or a CCD) enable the computation of the classic mean squared residuals (MSR). Moreover, they enable the estimation of the covariances between different types of simulation output; this MSR we call the multivariate MSR (MMSR):

$$\widehat{MMSR}_{h;h'} = \frac{1}{N-q} (\mathbf{w}_h - \widehat{\mathbf{y}}_h)' (\mathbf{w}_{h'} - \widehat{\mathbf{y}}_{h'}) \text{ with } h, h' = 0, \dots, r-1, \quad (9)$$

where the N -dimensional vector with regression predictors for output h is $\widehat{\mathbf{y}}_h = \mathbf{X}\widehat{\beta}_h$ ($h = 0, \dots, r-1$) (replicated points have identical predictors). This MMSR is unbiased if the regression metamodel is valid and no common random numbers (CRN) are used to simulate the different (local) combinations. More precisely, if $h = h'$, then (9) gives an unbiased estimator of the locally constant variance of w_h (simulation output h); if $h \neq h'$, then (9) gives an unbiased estimator of the covariance between w_h and $w_{h'}$ (assumed to be constant within the local area).

Moreover, we replicate the center point (say) $m \geq 2$ times, to obtain a second estimator that is unbiased even if the metamodel is not valid:

$$\widehat{\sigma}_{h;h'} = \frac{\sum_{s=1}^m [(w_{\mathbf{0};h;s} - \overline{w_{\mathbf{0};h}})(w_{\mathbf{0};h';s} - \overline{w_{\mathbf{0};h'}})]}{m-1} \quad (10)$$

where $w_{\mathbf{0};h;s}$ denotes output h of replication s of the center point $\mathbf{0}$, and $\overline{w_{\mathbf{0};h}}$ denotes the average of these replicates; if $h = h'$ then $\widehat{\sigma}_{h;h} = \widehat{\sigma}_h^2$.

The classic (univariate) *lack-of-fit F-statistic* is

$$F_{n-q;N-n;h} = \frac{\sum_{i=1}^n m_i (\overline{w_{i;h}} - \widehat{y}_{i;h})^2 / (n-q)}{\sum_{i=1}^n \sum_{s=1}^{m_i} (w_{i;h;s} - \overline{w_{i;h}})^2 / (N-n)}, \quad (11)$$

where $w_{i;h;s}$ denotes simulation output h in replication s of point i ; see [25]. We reject the regression metamodel if this statistic is significant—using Bon-

ferroni's inequality. This inequality implies that in each individual test we replace the classic α -value by α divided by the number of tests; e.g., if we test the fit of the regression metamodel for three outputs, then we test at $\alpha/3$ where $\alpha = 0.10$ (classic value) or $\alpha = 0.20$ (not an unusual value for multiple tests); see [24]. The use of this inequality makes the test *conservative*; i.e., the test has smaller type-I error probability than prespecified and higher type-II error probability (smaller power).

If we find significant lack-of-fit, we have two options: (i) Decrease the local area; e.g., halve each factor's range. (ii) Increase the order of the polynomial; e.g., switch from a first-order to a second-order polynomial. If we do not find significant lack of fit, then we still base our bootstrap on (10) because (9) may be inflated by undetected bias. Moreover, we use (10) to estimate the covariance matrix of the OLS regression parameters:

$$\widehat{\text{cov}}(\widehat{\beta}_h, \widehat{\beta}_{h'}) = \widehat{\Sigma} \otimes (\mathbf{Z}'\mathbf{Z})^{-1} \quad (h, h' = 0, \dots, r-1), \quad (12)$$

where $\widehat{\Sigma} = (\widehat{\sigma}_{h;h'})$ is the $r \times r$ matrix following from (10) and \otimes is the well-known "Kronecker product" operator; also see [27].

4 Testing the KKT conditions

The KKT conditions (3) for our random optimization problem (2) require statistical testing of various hypotheses.

4.1 Hypothesis 1: constraints are binding

We formulate the hypothesis that a given point (say) \mathbf{z}^0 is *feasible*, and a constraint is *binding* at that point; see (3) where $A(\mathbf{z}^0)$ denotes the set with the indices of the binding constraints at \mathbf{z}^0 :

$$H_0^{(1)} : E(w_{h'} | \mathbf{d} = \mathbf{0}) = a_{h'} \text{ with } h' = 1, \dots, r-1, \quad (13)$$

where $\mathbf{d} = \mathbf{0}$ corresponds with the center of the local area (see (7) and (8)) and $a_{h'}$ denotes the bounds in (2). We test the center point of the local area because that point is more representative than the other (extreme) points of the R-III design or the CCD; moreover, we now test a single point instead of n points. Our null-hypothesis implies zero slack for constraint h' .

To save computer time, a local experiment should start at its center point including replicates. If it turns out that either no constraint is binding or at least one constraint is violated, then the other KKT hypotheses (presented in

the next two subsections) need not be tested and the other $n - 1$ points of the local design need not be simulated.

We test the hypothesis (13) through the following t statistic:

$$t_{m-1}^{(h')} = \frac{\overline{w_{h'}(\mathbf{d} = \mathbf{0})} - a_{h'}}{\sqrt{\widehat{\sigma^2_{h'}}/m}} \text{ with } h' = 1, \dots, r-1, \quad (14)$$

where both the numerator and the denominator are based on the m replicates at the local center point; see (10).

This statistic may give the following three different results.

- 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 the optimal point is not yet found (assuming that the optimum occurs when at least one constraint is binding; see Section 1).
- The statistic is *significantly negative*; i.e., the current point is not feasible.
- The statistic is *nonsignificant*; i.e., the current point is feasible and constraint h' is binding. We then proceed to the next hypotheses —as follows.

4.2 Hypothesis 2: linear combination of gradients

We formulate the hypothesis that the expected value of the goal gradient may be expressed as the expected value of a linear combination of the estimated gradients of the binding constraints; i.e., in (3) we replace the corresponding deterministic quantities by their random estimators:

$$H_0^{(2)} : E(\widehat{\beta_{0;-0}}) = E\left(\sum_{h \in A(\mathbf{z}^0)} \widehat{\lambda_h^0} \widehat{\beta_h}\right). \quad (15)$$

We estimate this linear combination through OLS using as explanatory variables the estimated gradients of the (say) J binding constraints, so the explanatory variables are random. We collect the latter gradients in the $k \times J$ matrix $\widehat{\mathbf{B}_{J;-0}}$. The OLS estimator of the goal gradient is $\widehat{\beta_{0;-0}}$:

$$\widehat{\beta_{0;-0}} = \widehat{\mathbf{B}_{J;-0}} (\widehat{\mathbf{B}_{J;-0}}' \widehat{\mathbf{B}_{J;-0}})^{-1} \widehat{\mathbf{B}_{J;-0}}' \widehat{\beta_{0;-0}} = \widehat{\mathbf{B}_{J;-0}} \widehat{\boldsymbol{\lambda}}, \quad (16)$$

so $\widehat{\boldsymbol{\lambda}} = (\widehat{\mathbf{B}_{J;-0}}' \widehat{\mathbf{B}_{J;-0}})^{-1} \widehat{\mathbf{B}_{J;-0}}' \widehat{\beta_{0;-0}}$ is the OLS estimator of the Lagrange multipliers in the KKT conditions. (Obviously, if $\widehat{\beta_{0;-0}}$ and the vectors in $\widehat{\mathbf{B}_{J;-0}}$ point in the same direction, then all the components of $\widehat{\boldsymbol{\lambda}}$ are positive; if $\widehat{\beta_{0;-0}}$

and $\widehat{\mathbf{B}}_{J;-0}$ are perpendicular, then $\widehat{\boldsymbol{\lambda}}$ is zero; if $\widehat{\boldsymbol{\beta}}_{0;-0}$ and $\widehat{\mathbf{B}}_{J;-0}$ point in opposite directions, then $\widehat{\boldsymbol{\lambda}}$ is negative; also see Figure 1.) The expression in (16) is highly nonlinear; bootstrapping is a classic analysis method for nonlinear statistics.

There are several statistics for quantifying the accuracy of a model fitted through OLS. We use the k -dimensional vector of residuals

$$\mathbf{e}(\widehat{\widehat{\widehat{\boldsymbol{\beta}}_{0;-0}}}) = \widehat{\widehat{\widehat{\boldsymbol{\beta}}_{0;-0}}} - \widehat{\boldsymbol{\beta}}_{0;-0}. \quad (17)$$

Hypothesis (15) implies $\mathbf{e}(\widehat{\widehat{\widehat{\boldsymbol{\beta}}_{0;-0}}}) = \mathbf{0}$.

To test the latter hypothesis, we "simulate" gradient values that agree with the observed randomness, quantified through the covariance matrix in (12). Therefore, we sample from the relevant distribution; i.e., we apply parametric bootstrapping. This procedure consists of the following four steps, where we use the standard notation for bootstrapped values, namely the superscript $*$.

Step 1: Use the *Monte Carlo* method to sample $\text{vec}(\widehat{\boldsymbol{\beta}}_{0;-0}^*, \widehat{\mathbf{B}}_{J;-0}^*)$, which is a $(k+kJ)$ -dimensional vector formed by "stapling" (stacking) the k -dimensional goal gradient vector and the J k -dimensional vectors of the $k \times J$ matrix $\widehat{\mathbf{B}}_{J;-0}^*$:

$$\text{vec}(\widehat{\boldsymbol{\beta}}_{0;-0}^*, \widehat{\mathbf{B}}_{J;-0}^*) \sim N \left(\text{vec}(\widehat{\boldsymbol{\beta}}_{0;-0}, \widehat{\mathbf{B}}_{J;-0}), \mathbf{cov}[\text{vec}(\widehat{\widehat{\widehat{\boldsymbol{\beta}}_{0;-0}}}, \widehat{\widehat{\widehat{\mathbf{B}}_{J;-0}}})] \right), \quad (18)$$

where $N(a, b)$ denotes a normal distribution with mean a and variance b ; $\mathbf{cov}[\text{vec}(\widehat{\widehat{\widehat{\boldsymbol{\beta}}_{0;-0}}}, \widehat{\widehat{\widehat{\mathbf{B}}_{J;-0}}})]$ is the $(k+kJ) \times (k+kJ)$ matrix computed through (12) (the latter matrix gives the covariances for all simulation outputs, but we do not need the covariances for the non-binding outputs).

Step 2: Use the bootstrap values from Step 1, to compute the *OLS* estimate of the bootstrapped goal gradient using the bootstrapped gradients of the binding constraints as explanatory variables; i.e., we use (16) adding the superscript $*$ to all random variables, which results in $\widehat{\widehat{\widehat{\boldsymbol{\beta}}_{0;-0}^*}}$ and $\widehat{\boldsymbol{\lambda}}^*$.

Step 3: Use $\widehat{\widehat{\widehat{\boldsymbol{\beta}}_{0;-0}^*}}$ from Step 2 and $\widehat{\boldsymbol{\beta}}_{0;-0}^*$ from Step 1, to compute the *bootstrap residual* $\mathbf{e}(\widehat{\widehat{\widehat{\boldsymbol{\beta}}_{0;-0}^*}}) = \widehat{\widehat{\widehat{\boldsymbol{\beta}}_{0;-0}^*}} - \widehat{\boldsymbol{\beta}}_{0;-0}^*$, analogously to (17).

Step 4: *Repeat* the preceding three steps (say) 1000 times, to get the EDF of the bootstrapped residuals per input j ($j = 1, \dots, k$)—denoted by $\mathbf{e}(\widehat{\widehat{\widehat{\boldsymbol{\beta}}_{0;-0;j}^*}})$. Reject the hypothesis in (15) if this EDF implies a two-sided $(1 - \alpha/(2k))$

confidence interval (CI) that does not cover the value 0—for any of the k inputs (the factor k in the CI is explained by Bonferroni's inequality).

4.3 Hypothesis 3: positive Lagrange multipliers

We formulate the hypothesis that the Lagrange multipliers are non-negative:

$$H_0^{(3)} : E(\hat{\lambda}) \geq \mathbf{0}. \quad (19)$$

To test this hypothesis, we expand Step 2 of the preceding subsection and add 1 to a counter (say) c^* if any of the bootstrapped Lagrange multipliers $\hat{\lambda}^*$ is negative (we are not interested in the magnitude of a Lagrange multiplier, but only in its direction or sign). After the 1000 bootstrap samples of Step 4 in that subsection, we have the final value of this counter. We reject the hypothesis in (19) if the fraction $c^*/1000$ is significantly higher than 50% (if the true Lagrange multiplier is only "slightly" larger than zero, then "nearly" 50% of the bootstrapped values is negative). To test the latter fraction, we approximate the binomial distribution through the normal distribution with mean 0.50 and variance $(0.50 \times 0.50)/1000 = 0.00025$.

Note that the $k \times J$ matrix $\widehat{\mathbf{B}}_{J,-0}$ may be a square matrix. If $\widehat{\mathbf{B}}_{J,-0}$ is also non-singular, then $\widehat{\mathbf{e}}(\widehat{\beta}_{0,-0}^*) = \mathbf{0}$; i.e., the random KKT problem reduces to a deterministic problem! For solving such problems we refer to the literature on deterministic nonlinear programming. In practice, however, this deterministic variant of the originally random problem seems rare; e.g., [14] presents a simulation model for production planning at a Dutch steel-tube manufacturer with $k = 14$ inputs and $r = 2$ outputs so $\widehat{\mathbf{B}}_{J,-0}$ cannot be a square matrix.

5 Examples

We test our procedure extensively through Example 1, for which we (but not our procedure) know the *true* I/O functions of the "simulation" model, namely the second-order polynomials in (1). Consequently, we know the true optimum and the binding constraints; see the points A through D in Figure 1. Moreover, we ensure that the simulation outputs are multivariate normal; i.e., to the polynomials we add multivariate normal noise; see again (1).

We further study the *robustness* of our procedure through Examples 2 and 3, which are discrete-event simulation models. We do not know their I/O

functions, so we do not know the true optima or the points that make the constraints binding. We assume that the simulation outputs of Example 2 are Gaussian; for Example 3 we use distribution-free bootstrapping.

Note that any numerical experiment requires a choice of factor values and combinations. This choice should not bias the outcomes in favor of the method being evaluated; i.e., the readers must trust the experimenters, and other researchers should be able to reproduce and extend the experiment.

5.1 Example 1: second-order polynomial I/O functions

To generate data for Example 1, we must select values for $\mathbf{cov}(\mathbf{e})$, which characterizes the additive noise. This $\mathbf{cov}(\mathbf{e})$ determines the signal/noise ratio, $\beta/[var(\hat{\beta})]$ —once we have selected the range of the local area; also see (12). Two conflicting arguments apply—one mathematical and one statistical (also see [30]): (i) the smaller the range of the local area, the better the local low-order approximation (Taylor series argument); (ii) the larger this range, the higher the signal/noise ratio; i.e., the smaller the noise $var(\hat{\beta})$. We select the following standard deviations for \mathbf{e} : $\sigma_0 = 1$, $\sigma_1 = 0.15$, and $\sigma_2 = 0.4$; we select the following correlations: $\rho_{0;1} = 0.6$, $\rho_{0;2} = 0.3$, and $\rho_{1;2} = -0.1$; also see [1]. Together with our choice of the size of the local areas, our choices turn out to give reasonable signal/noise values.

Our type-I error probability is $\alpha = 0.10$ per test. To estimate the power function of our procedure, we apply the procedure to four local areas—with center points corresponding with the points A through D in Figure 1. Point A is the optimal point, so at this point our test procedure should reject the KKT conditions with a probability smaller than α (because we use the conservative Bonferroni inequality). Point B is "near" the optimum, and has the same binding constraint as point A. At point B, our procedure should reject the KKT conditions with a probability higher than α ; i.e., our procedure should show increasing power as the point tested moves away from the true optimum. Point C is "far away" from the optimum, so our procedure should now reject the KKT conditions with a probability higher than that of point B. Point D is even further away from the optimum (and with a different binding constraint).

To get an accurate estimate of the power of our procedure, we run 1,000 *macro-replicates* of our example; i.e., we take 1,000 sampled vectors of the simulation output \mathbf{w} per input combination, estimate 1,000 gradients for simulation output $h = 0, 1, 2$, obtain 999 bootstrap samples per gradient (999 can be shown to be more convenient than 1,000), etc.

Table 1 displays the fraction of these macro-replicates that reject our various null-hypotheses. We display results for the four locations (A through D),

for which we also display their coordinates (e.g., A has coordinates 2.53 and -1.99). Moreover, we experiment with the *size* of the local area; i.e., in the "large" area, the original values that correspond with the coded values 0 and 1 differ by 0.1; in the "small" area, these values change by 0.01 (for "coding" see again (7)). The lack-of-fit F -statistic enables us to test whether the area's size is acceptable—given the metamodel (e.g., a second-order polynomial). We use CRN for the two areas, which results in a perfect correlation coefficient of $+1$. And we experiment with the magnitude of the *noise*; i.e., "small" noise means that the standard deviations are only 10% of the values specified above (namely, $\sigma_0 = 1$, $\sigma_1 = 0.15$, $\sigma_2 = 0.4$).

To explain the numbers in Table 1, we start with point A's upper-left element and proceed with the other elements in the same row, etc.:

$69/1000 = 0.07$: The denominator equals our number of macro-replicates. Our procedure uses the t statistic (14) to test if at least one constraint is binding. This hypothesis is rejected for 69 macro-replicates. (Of these 69 macro-replicates, 42 replicates find the first constraint to have positive slack, and the second constraint to have negative slack; the remaining $69 - 42 = 27$ replicates find both constraints to have positive slacks; these numbers are not displayed, to save space.) So the classic t statistic combined with Bonferroni's inequality gives slightly conservative results (fractions smaller than $\alpha = 0.10$)—as we expected. The other elements in this row give similar results. The other points (B, C, D) have rows with similar results.

$79/931 = 0.08$: The number of macro-replicates that remain is $1000 - 69 = 931$. We use the lack-of-fit F -statistic in (11), to test the adequacy of the second-order polynomial fitted locally to the simulation I/O data. Because Example 1 has multiple simulation outputs, we again use Bonferroni's inequality and get 0.08, which is lower than the nominal $\alpha = 0.10$. We get similar results for the other cases: see the elements in the same row as 79/931, and the appropriate rows for points B, C, and D.

$106/852 = 0.12$: The number of remaining macro-replicates is $931 - 79 = 852$. We use bootstrapping to test whether the estimated goal gradient can be adequately expressed as a linear function of the estimated gradient of the binding constraint. Because Example 1 has $k = 2$ simulation inputs, we again apply Bonferroni's inequality. The rejected fraction is 0.12, which is slightly higher than the nominal 0.10. Two other results in this row are similar; the case of a small local area with large noise, however, gives a low fraction, namely 0.02—which is taken care of, in the next line of the table.

$0/746 = 0.00$: The number of remaining macro-replicates is $852 - 106 = 746$. We now test whether the linear combination has positive Lagrange multipliers. None of the remaining replicates gives a negative multiplier—except for the

case of a small local area with large noise, which gives a high fraction, namely 0.66.

From Table 1 we derive the following conclusions:

- Both the classic t test for the identification of the binding constraints, and the classic lack-of-fit F -test perform well—independent of the distance from the optimum, the size of the local area, and the magnitude of the noise.
- The farther away a point is from the optimum, the higher is the probability of rejecting the OLS model that expresses the estimated goal gradient as a linear function of the estimated gradient of the binding constraint. The type-I error probability at the optimum itself is acceptable. However, in case of a small signal and large noise the test often does not reject that linear model (such behavior is usual in any statistical test); fortunately, in such a case the KKT conditions are rejected because of the wrong (negative) signs of the estimated Lagrange multipliers.

We also investigate the fitting of first-order (instead of second-order) polynomials to the simulation's I/O data. Such polynomials have only $q = k + 1$ parameters, so a R-III design instead of a CCD suffices. Example 1 has only $k = 2$ simulation inputs, so our procedure simulates the combinations of a 2^2 design plus the center point, which is replicated $m = 4$ times. We experiment with the same cases as in Table 1. We obtain results that are similar to the results in that table, except for the case of a large local area with small noise. In the latter case, the lack-of-fit F -test rejects the polynomial approximation more often; e.g., in point A that test reject 200 of 927 macro-replicates so the fraction is 0.22, whereas Table 1 rejects only 77 of 908 macro-replicates so the fraction is 0.08 (we know that second-order approximations are perfect for this example, so first-order polynomials are only approximations). To save space we do not present further details.

5.2 Example 2: an (s, S) inventory model with a service level constraint

Inventory simulation is an important topic in the simulation literature; see the classic simulation textbook [20]. A well-known building block for such simulation is the (s, S) inventory model. Moreover, this model is often used to illustrate simulation optimization; see [3], [11], and [26]. By definition, the (s, S) inventory model (with $s < S$) replenishes the inventory (say) I whenever it decreases to a value smaller than or equal to the reorder level s ; the order quantity Q is such that the inventory is raised to the order-up-to level S :

$$Q = \begin{cases} S - I & \text{if } I \leq s \\ 0 & \text{if } I > s \end{cases} \quad (20)$$

where I does not denote the physical inventory but the inventory including orders that have already been placed.

In practice, out-of-stock costs are hard to quantify so a *service or fill rate constraint* is specified instead; e.g., the expected fraction of total demand satisfied from stock on hand should be at least 90%. The following variant is studied in [3]. The inventory systems has random demand and random lead time. This random lead time implies that orders may cross in time; i.e., orders are not necessarily received in the order in which they are placed—which complicates the mathematical analysis so simulation is used. Estimating the *optimal* control limits (say) s_0 and S_0 turns out to be very difficult, as the vast literature on inventory management shows.

Estimating these s_0 and S_0 may use *brute force*; i.e., a grid with all the combinations of the integers s and S within a given experimental area is used to search for the optimum. This method is indeed used in [1], [3], and [34]. Unfortunately, these three publications report different (s, S) combinations as being optimal! Moreover, brute force cannot be applied to realistic problems; e.g., practical inventory systems control thousands of Stock Keeping Units (SKUs), so "the curse of dimensionality" arises. Furthermore, the inventory system may be only a subsystem of a production-inventory system (the SKU is not purchased but is manufactured by the same company; see Example 3). Therefore Example 2 only illustrates how in practice optimization of realistic inventory systems may be done.

The following three simulation-optimization heuristics are compared in [18]: (i) OptQuest, which combines the metaheuristics of Tabu Search, Neural Networks, and Scatter Search into a single search heuristic (see

<http://spot.colorado.edu/~glover/>); (ii) Generalized RSM or GRSM (developed in [1]), and (iii) perturbation analysis (PA) combined with the feasible directions (FD) method (used in [3]). Furthermore, [18] uses our procedure to check whether the three optimal points estimated through these heuristics indeed satisfy the KKT conditions. (Examples 1 and 2 use the same Matlab code for testing the KKT conditions.) Obviously, the first two heuristics treat the simulation model as a black box, whereas the third heuristic uses a white-box approach.

OptQuest enables the users to control the search as follows. OptQuest allows different *precision criteria* for both the objective and the constrained simulation outputs; i.e., Example 2 restricts the number of replicates to $10 \leq m \leq 100$. Moreover, OptQuest allows the users to select a *relative precision*; i.e., Example 2 selects m such that the halfwidth of the 95% confidence interval for the average output is within 5% of the true mean. Finally, OptQuest allows different *stopping or termination criteria*; i.e., Example 2 stops the search af-

ter either 5 hours of PC time or 500 "nonimproving solutions". We propose to add our procedure to OptQuest, and use it as a better stopping rule.

In Example 2, the following two gradients are relevant where c denotes the expected value of the total relevant costs, and f the expected disservice rate (the required—not the expected—disservice rate is denoted by γ):

$$\beta_{0,-0} = \nabla(c) = \left(\frac{\partial c}{\partial s}, \frac{\partial c}{\partial S} \right)' \quad (21)$$

and

$$\beta_{1,-0} = \nabla(f) = \left(\frac{\partial f}{\partial s}, \frac{\partial f}{\partial S} \right)' \quad (22)$$

To estimate these two gradients, OptQuest may be augmented with a local experiment at the point that OptQuest estimates to be optimum. GRSM implies that the two gradients are estimated from a local simulation experiment. PA estimates the gradients from a single simulation run, using the explicit functions (e.g. (20)) inside the simulation model.

The null-hypothesis (13), which states that the current slack is zero, now becomes

$$H_0^{(1)} : E(\hat{f}) = 0.10 \quad (23)$$

where \hat{f} is an unbiased estimator of f , which was defined before (21).

To estimate the goal gradient, [18] fits the following second-order polynomial locally:

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 s + \hat{\beta}_2 S + \hat{\beta}_3 sS + \hat{\beta}_4 s^2 + \hat{\beta}_5 S^2 \quad (24)$$

so the gradient is estimated through

$$\widehat{\nabla(c)} = (\hat{\beta}_1 + \hat{\beta}_3 S + 2\hat{\beta}_4 s, \hat{\beta}_2 + \hat{\beta}_3 s + 2\hat{\beta}_5 S)'. \quad (25)$$

Another second-order polynomial is fitted locally for the fill rate, to get $\widehat{\nabla(f)}$ analogous to (25).

The assumptions originally used in [3] are also used in [18]; i.e., demands are exponentially distributed with mean 100; lead times are Poisson distributed with mean 6 ; the maximum disservice level γ is 0.10; holding cost is 1, variable ordering cost is 1, and fixed ordering cost is 36.

Table 2 (reproduced from [18]) summarizes the results of the brute-force grid searches and the search heuristics used by three research teams. Obviously, Angün et al.'s solution differs much from the solutions reported by the other two teams.

To estimate the performance of our KKT test procedure, [18] investigate the following three *local area sizes* for the integer combinations of s and S : (i) a "small" local area of 4×4 units; (ii) an "intermediate" local area of 10×10 ; (iii) a "large" local area of 20×20 . Moreover, that article investigates two *noise levels*: (i) relatively small noise resulting from simulation runs with a length of 30,000 periods; (ii) relatively big noise resulting from simulation runs of only 3,000 periods.

Table 3 displays results only for the small local area and the small noise level (1 of the $3 \times 2 = 6$ experimental conditions). Its points denoted by A through D are specified in Table 2; its point E is (985, 1188) and is obviously not optimal (E was found during the OptQuest search for the true optimum). The nominal type-I error probability per test is again $\alpha = 0.10$. The number of macro-replicates is now 500. The CCD replicates the center point $m = 3$ times. The rest of the notation is analogous to Table 1.

The results for points A, B, and C are quite similar. Point D, however, gives significant slacks for all 500 macro-replicates. Point E results in 18% ($= 68/388$) of the Lagrange multipliers being negative; i.e., the two estimated gradients point in different directions so this point is suboptimal.

Results for all six combinations of local area size and noise level are presented in [34]. These results resemble Table 3. Increasing m (number of replicates at center of local area) from 3 to 20 increases the power of the KKT tests. (Too small an m value must be avoided, because too much noise never gives significant results—for any procedure.)

5.3 *Example 3: an integrated (s , S) inventory-production model with a service level constraint*

Example 3 is reported in [4]. This example concerns the optimization of the two control parameters s and S of an integrated production-inventory simulation model, including our KKT procedure. Example 3 uses a global Kriging metamodel (instead of a series of low-order local polynomials, as Examples 1 and 2 do).

In general, *Kriging* models (also called spatial correlation models) are typically fitted to data that are obtained for larger experimental areas than the areas used in low-order polynomial metamodels; i.e., Kriging models are global rather than local. These models are used for prediction; the final goals are sensitivity analysis and optimization.

Kriging was originally developed in *geostatistics* (or spatial statistics) by the South-African mining engineer Krige. A classic geostatistics textbook is [6].

Later on, Kriging models were also applied to the I/O data of *deterministic simulation* models; see [31]. Recently, Kriging has been applied to *random simulation* models; see [33]. We also refer to the recent review on Kriging in [16].

We focus on the simplest type of Kriging, called *Ordinary Kriging*—henceforth briefly called Kriging. This Kriging assumes

$$w(\mathbf{d}) = \mu + \delta(\mathbf{d}), \quad (26)$$

where $\mathbf{d} = (d)$ denotes the vector of simulation inputs, μ the simulation output averaged over the experimental area, and $\delta(\mathbf{d})$ the additive noise that forms a "stationary covariance process" with zero mean. By definition, a time series w_t is a *stationary covariance process* if it has a constant mean $E(w_t) = \mu$, a constant variance $\text{var}(w_t) = \sigma^2$, and covariances depending only on the lag $|t - t'|$ so $\text{cov}(w_t, w_{t'}) = \sigma_{|t-t'|}$.

Kriging uses the following *linear* predictor:

$$y(\mathbf{d}) = \boldsymbol{\lambda}(\mathbf{d}, \mathbf{D})' \overline{\mathbf{w}}(\mathbf{D}) = \boldsymbol{\lambda}' \overline{\mathbf{w}}, \quad (27)$$

where the weights $\boldsymbol{\lambda}(\mathbf{d}, \mathbf{D})$ —abbreviated to $\boldsymbol{\lambda}$ —are not constants (whereas the regression parameters $\boldsymbol{\beta}$ are) but decrease with the *distance* between the "new" input (say) \mathbf{d}_0 to be predicted and the "old" inputs \mathbf{D} (see Section 3); the vector with simulation outputs averaged over the m_i replicates is $\overline{\mathbf{w}}(\mathbf{D})$ —abbreviated to $\overline{\mathbf{w}}$. (How these weights decrease exactly, we shall explain below.)

To select the optimal weights, Kriging uses the criterion of the *best linear unbiased predictor (BLUP)*:

$$\min_{\boldsymbol{\lambda}} \text{MSE}[y(\mathbf{d})] = \min_{\boldsymbol{\lambda}} [E\{y(\mathbf{d}) - w(\mathbf{d})\}^2], \quad (28)$$

where the predictor must satisfy the *unbiasedness* constraint

$$E\{y(\mathbf{d})\} = E\{w(\mathbf{d})\}. \quad (29)$$

It can be proven that *optimal* weights must satisfy the condition

$$\sum_{i=1}^n \lambda_i = 1 \quad (30)$$

or (in matrix notation) $\mathbf{1}'\boldsymbol{\lambda} = 1$. Note that some weights may be negative! It can be derived that the optimal weights are

$$\boldsymbol{\lambda}_o = \boldsymbol{\Gamma}^{-1}[\boldsymbol{\gamma} + \mathbf{1} \frac{1 - \mathbf{1}'\boldsymbol{\Gamma}^{-1}\boldsymbol{\gamma}}{\mathbf{1}'\boldsymbol{\Gamma}^{-1}\mathbf{1}}] \quad (31)$$

where $\mathbf{\Gamma} = (\text{cov}(w_i, w_{i'}))$ with $i, i' = 1, \dots, n$ is the $n \times n$ symmetric and positive semi-definite matrix with the covariances between the old outputs; $\boldsymbol{\gamma} = (\text{cov}(w_i, w_0))$ is the n -dimensional vector with the covariances between the n old outputs and the new output.

These weights imply that—for an old input—the predictor equals the observed average simulation output at that input:

$$y(\mathbf{d}) = \overline{w(\mathbf{d})} \text{ if } \mathbf{d} \in \mathbf{D}, \quad (32)$$

so all weights are zero except the weight of the observed output, which is one. This property implies that the Kriging predictor is an exact *interpolator* (the regression predictor is not—unless $n = q$).

Finally, it can be proven that

$$y(\mathbf{d}_0) = \hat{\mu} + \boldsymbol{\gamma}(\mathbf{d}_0)' \mathbf{\Gamma}^{-1} (\bar{\mathbf{w}} - \hat{\mu} \mathbf{1}) \quad (33)$$

with

$$\hat{\mu} = (\mathbf{1}' \mathbf{\Gamma}^{-1} \mathbf{1})^{-1} \mathbf{1}' \mathbf{\Gamma}^{-1} \mathbf{w}. \quad (34)$$

Obviously, the optimal weights in (31) depend on the covariances—or equivalently the correlations—between the simulation outputs. The usual assumption is that the *correlation function* for a k -dimensional input vector is the *product* of the k one-dimensional functions. A stationary covariance process implies that these correlations depend only on the distance

$$h_j = |d_{ij} - d_{gj}| \quad (j = 1, \dots, k) \quad (i = 1, \dots, n)(g = 0, 1, \dots, n). \quad (35)$$

Example 3 uses the popular assumption of a *Gaussian correlation function* (say) ρ :

$$\rho = \exp\left[-\sum_{j=1}^k h_j^2 \theta_j\right] = \prod_{j=1}^k \exp(-h_j^2 \theta_j), \quad (36)$$

where θ_j measures the importance of input j .

We point out that $\nabla(y) = (\partial y / \partial d_1, \dots, \partial y / \partial d_k)'$ —the *gradient* of the Kriging predictor—follows from (33) and (34), where $\boldsymbol{\gamma}$ is a function of the input \mathbf{d} for the output w_0 . For example, assuming a single input and a Gaussian correlation function, [15] derives

$$\partial y / \partial d_1 = \left(-2\theta(d_0 - d_1)e^{-\theta(d_0 - d_1)^2}, \dots, -2\theta(d_0 - d_n)e^{-\theta(d_0 - d_n)^2}\right) \cdot \mathbf{\Gamma}^{-1}(\mathbf{w} - \hat{\mu} \mathbf{1}).$$

A major problem is that the optimal weights depend on the correlation function of the underlying simulation model—but *this correlation function is unknown*. Therefore the parameter values θ_j in (36) must be estimated. Most

Kriging literature uses maximum likelihood estimators (MLEs). Replacing the weights by their estimates $\widehat{\lambda}_0$ makes the Kriging predictor *nonlinear*.

For the estimation of the correlation functions, the optimal weights, and the gradients, Example 3 uses the Matlab Kriging toolbox DACE—which is well documented and free of charge; see [21]. Though Example 3 has multiple (namely $r = 2$) simulation outputs (analogous to Example 2), a Kriging model is fitted per simulation output \mathbf{w}_h (with $h = 1, 2$). Ignoring the multivariate character of the output is usual when Kriging in simulation.

Unfortunately, the DACE software ignores both the multivariate character of the simulation output and the random character (DACE was developed for deterministic simulation). Example 3 applies the DACE software such that the Kriging predictors equal the *average* outputs at the inputs already observed; see (32).

The stationary covariance process implies a constant variance. In random simulation, however, the output variances are not constant. Fortunately, [17] demonstrates that the Kriging model is not very sensitive to this variance heterogeneity.

To obtain the necessary simulation I/O data, Example 3 uses *Latin hypercube sampling* (LHS). Originally, LHS was developed in [23] (albeit not for Kriging). References to popular *software* for LHS are given in [15]. The simplest LHS proceeds as follows: (i) LHS divides the range of each standardized input d_j ($j = 1, \dots, k$) into n (number of input combinations) mutually exclusive and exhaustive intervals of equal length. (ii) LHS randomly selects one value for the first input d_1 from each interval, without replacement. (iii) LHS pairs these n values with the n values of the second input, d_2 , randomly without replacement. (iv) LHS combines these n pairs with the n values of the third input, d_3 , randomly without replacement to form n triplets. (v) And so on, until a set of n k -tuples is formed.

A crucial characteristic of LHS is that there is no strict mathematical relationship between n and k , whereas a R-III design has $n = 2^{k-p}$ with $0 \leq p < k$. Obviously, simulating more input combinations does not hurt. Rules of thumb can be found in the literature; Example 3 uses $n = 10k$.

There are several LHS variants; Example 3 uses *maximin LHS*, which maximizes the minimum distance between the n points in the LHS design; see <http://www.spacefillingdesigns.nl/>.

Random simulation requires a number of replicates m_i ($i = 1, \dots, n$) to obtain accurate output data. Example 3 (like Example 2) uses a *relative precision* criterion; i.e., m_i is such that the halfwidth of the 95% confidence interval for the average output of point i is within 5% of the true mean. This criterion

turns out to require between five and seven replicates.

So, for point i there are m_i simulation outputs $(c_{i;s}, f_{i;s})$ where (as in Example 2) c denotes the relevant costs and f the fill rate ($s = 1, \dots, m_i$). Obviously, these two outputs are IID bivariate. To improve the signal/noise ratio, Example 3 uses *CRN*; i.e., replicate s uses the same PRN seed for all n points—with $s = 1, \dots, \min_i(m_i)$. (With CRN the simulation outputs at different points are correlated so the Gaussian correlation function decreases slower; i.e., $\hat{\theta}$ in (36) decreases.)

Our procedure first checks whether the constraint for the fill rate is binding. The original (not the bootstrap) data are used in the t test defined in (14).

Once the Kriging metamodels per simulation output have been estimated from the simulation I/O data, Matlab’s function called *fmincon*—meant for constrained nonlinear optimization—is used to estimate the optimum input combination and the resulting values for the goal output and constrained output. This estimated solution needs further refinement, because *fmincon* ignores the integer constraints on the two inputs s and S , and it is based on the Kriging metamodel (which only approximates the underlying simulation model).

Because of $5 \leq m_i \leq 7$, *distribution-free bootstrapping* seems a reasonable method. This bootstrap gives $(c_{i;s}^*, f_{i;s}^*)$ with $i = 1, \dots, n$ and $s = 1, \dots, m_i$. Kriging of these bootstrap I/O data gives r new metamodels, which give new gradients, $\widehat{\nabla}(c_o^*)$ and $\widehat{\nabla}(f_o^*)$. These two gradients are then expressed as a linear combination, using OLS which gives estimated bootstrapped residuals \widehat{e}^* and the Lagrange multiplier $\widehat{\lambda}^*$. The bootstrap sample with $B = 50$ observation gives the EDF of \widehat{e}^* . This EDF gives a confidence interval; if this interval does not cover the value zero, then the linear combination is rejected. The $\widehat{\lambda}^*$ are used to update a variable that counts the number of negative values for $\widehat{\lambda}^*$; this counter is used to test the hypothesis that the Lagrange multiplier is nonnegative.

It turns out that our procedure does not reject the null-hypothesis that the integer solution found via *fmincon* is indeed optimal: (i) the t test does not reject the zero slack hypothesis; (ii) the bootstrap does not reject the hypothesis of zero residuals in the OLS estimate expressing the goal gradient in the fill rate gradient; (iii) the same bootstrap does not reject the sign of the Lagrange multiplier.

Actually, Experiment 3 evaluates three LHS designs, which differ in size ($n = 5$ or $n = 20$) or in the actual values of the input combinations (LHS implies sampling). Each experiment gives integer input combinations that are not rejected by our KKT procedure. Experiment 3 also tests the KKT conditions

for a point that is clearly not optimal but does have zero slack. This point requires a maximum of 26 replications. Now 3 out of 50 bootstrapped Lagrange multipliers are negative so the counter is not significant at $\alpha = 0.10$. However, the residual for input 2 (namely, S) differs significantly from zero so this point is rejected! Considering the various cases in Experiments 3, we conclude that our KKT procedure does show an acceptable power function in this example.

6 Conclusions and further research

The literature on optimization of random simulation models offers many heuristic search methods. In this article, we derived a sequential statistical procedure that may be used as a stopping rule for such methods; i.e., our procedure tests the null-hypothesis that the KKT conditions hold so the alleged optimum is a true optimum.

Our procedure starts with the simulation of the input combination specified by the center of the local area that is hypothesized to be optimal. Replication of this combination enables the use of a t statistic to test whether any constraint is binding.

If a binding constraint is indeed found, then our procedure continues and obtains the EDF of the estimated gradients. Obtaining this EDF varies with the search methods; e.g., RSM uses local first-order or second-order polynomials, which may lead to a multinormal EDF.

Parametric bootstrapping uses this EDF to obtain the gradients of the goal function and the binding constraints. Distribution-free bootstrapping is an alternative method for obtaining gradients, provided all input combinations are replicated. Bootstrapping enables testing whether the goal gradient can be adequately approximated by a linear function—estimated through OLS—of the binding constraint gradients. At the same time, this bootstrapping enables testing whether the Lagrange multipliers—estimated through OLS—are non-negative.

Our three examples empirically show that our test procedure has a power function that increases as the point tested moves away from the true optimum point; at the true optimum the type-I error probability is close to the specified value. Of course, "steeper" power functions are desirable, so future research may try to improve our procedure.

Future research may also try to come up with a theoretical (instead of empirical) analysis of the properties of our *sequential* statistical procedure. Unfortunately, sequential procedures are notoriously hard to analyze (also see the

RSM procedure).

Moreover, we recommend further analysis because our procedure uses *simultaneous* testing of multiple hypotheses.

Furthermore, when using *local* approximations (as is the case in RSM), how small should the experimental area be? Various statistics (e.g., the lack-of-fit F -statistic, cross-validation statistics such as PRESS) make it possible to test whether a given metamodel is a valid approximation; see [7] and [15]. (In a global approach such as Kriging, this question vanishes; in mathematical programming, the so-called "trust" area is also rather subjective.)

Application of our procedure to more examples (besides our three examples) may show how and when our methodology is reliable and robust. Instead of assuming Gaussian distributions, we may assume a t distribution; such a distribution is used (for other problems than simulation optimization) in [19].

CRN is an interesting research topic: CRN increases the accuracy of the simulation results, but it also complicates the analysis. Moreover, CRN requires many more replicates to estimate all the correlations that it creates.

We use the residuals themselves—combined with Bonferroni’s inequality—to validate the linear approximation that expresses the goal gradient in the gradients of the binding constraints. Other statistics (e.g., the sum of squared residuals) may be alternatives. We also refer to the discussion on "pivotal" statistics in [22].

Bonferroni’s inequality provides simple but conservative tests, so future research might use the notion of "data depth"—discussed in [29].

Finally, we may test second-order optimality conditions besides the first-order KKT conditions.

Acknowledgements

We thank the three anonymous referees for their comments on the first version of this article, which lead to a major revision. Enrique del Castillo visited Tilburg University as a Fulbright Fellow during four months in 2005. During part of the research for this article, Jack Kleijnen was also employed (for 50%) by the Wageningen University and Research Centre’s Operations Research & Logistics Group/Mansholt Graduate School of Social Sciences.

References

- [1] Angün, E., D. den Hertog, G. Gürkan, and J.P.C. Kleijnen (2006), Response surface methodology with stochastic constraints for expensive simulation. Working Paper, Tilburg University, Tilburg, Netherlands
- [2] Angün, E. and J.P.C. Kleijnen (2006), An asymptotic test of optimality conditions in multiresponse simulation-based optimization. Working Paper, Tilburg University, Tilburg, Netherlands
- [3] Bashyam, S. and M. C. Fu (1998), Optimization of (s, S) inventory systems with random lead times and a service level constraint. *Management Science*, 44, pp. 243-256
- [4] Biles, W.E., J. P. C. Kleijnen, W. C. M. van Beers, and I. van Nieuwenhuyse (2007), Kriging metamodeling in constrained simulation optimization: an explorative study. *Proceedings of the 2007 Winter Simulation Conference*, edited by S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton (submitted)
- [5] Cheng, R.C.H. (2006), Resampling methods. *Handbooks in Operations Research and Management Science, Volume 13*, edited by S.G. Henderson and B.L. Nelson, pp. 415-453
- [6] Cressie, N.A.C. (1993), *Statistics for spatial data: revised edition*. Wiley, New York
- [7] Del Castillo, E. (2007), *Process optimization: a statistical approach*. Springer, New York
- [8] Efron, B. and R.J. Tibshirani (1993), *An introduction to the bootstrap*. Chapman & Hall, New York
- [9] Gill, P. E., W. Murray, and M. H. Wright (2000). *Practical optimization, 12th edition*. Academic Press, London
- [10] Higle, J.L. and S. Sen (1991), Statistical verification of optimality conditions for stochastic programs with recourse. *Annals of Operations Research*, 30, pp. 215-240
- [11] Kao, C. and S.-P. Chen (2006), A stochastic quasi-Newton method for simulation response optimization. *European Journal of Operational Research*, 173, no. 1, pp. 30-46
- [12] Karaesman, I and G. van Ryzin (2004), Overbooking with substitutable inventory classes, *Operations Research*, 52, no. 1, pp. 83-104
- [13] Kim, S., C. Alexopoulos, K. Tsui, and J.R. Wilson (2007), Distribution-free tabular CUSUM chart for autocorrelated data. *IIE Transactions*, 39, pp. 317-330

- [14] Kleijnen, J.P.C. (1993), Simulation and optimization in production planning: a case study. *Decision Support Systems*, 9, pp. 269-280
- [15] Kleijnen, J.P.C. (2007), *DASE: Design and analysis of simulation experiments*. Springer, New York
- [16] Kleijnen, J.P.C. (2007), Kriging metamodeling in simulation: a review *European Journal of Operational Research* (accepted conditionally)
- [17] Kleijnen, J.P.C. and W.C.M. van Beers (2005), Robustness of Kriging when interpolating in random simulation with heterogeneous variances: some experiments. *European Journal of Operational Research*, 165, no. 3, pp. 826-834
- [18] Kleijnen, J.P.C. and J. Wan (2007), Optimization of simulated systems: OptQuest and alternatives. *Simulation Modelling Practice and Theory*, 15, pp. 354-362
- [19] Lange, K. L., R.J.A. Little, and J.M.G. Taylor. (1989), Robust statistical modeling using the t distribution, *Journal of the American Statistical Association*, 84, pp. 881-896
- [20] Law, A.M. (2007), *Simulation modeling and analysis; fourth edition*. McGraw-Hill, Boston
- [21] Lophaven, S.N., H.B. Nielsen, and J. Sondergaard (2002), DACE: a Matlab Kriging toolbox, version 2.0. IMM Technical University of Denmark, Lyngby
- [22] Martin, M.A. (2007), Bootstrap hypothesis testing for some common statistical problems: A critical evaluation of size and power properties *Computational Statistics & Data Analysis*, in press
- [23] McKay, M.D., R.J. Beckman, and W.J. Conover (1979), A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21, no. 2, pp. 239-245 (reprinted in *Technometrics*, 42, no. 1, 2000, pp. 55-61)
- [24] Miller, R.G. (1981), *Simultaneous statistical inference; second edition*. Springer-Verlag, New York
- [25] Myers, R.H. and D.C. Montgomery (2002), *Response surface methodology: process and product optimization using designed experiments; second edition*. Wiley, New York
- [26] Pitchitlamken, J. and B.L. Nelson (2003), A combined procedure for optimization via simulation. *ACM Transactions on Modeling and Computer Simulation*, 13, no. 2, pp. 155-179
- [27] Porta Nova, A.M. and J.R. Wilson (1989), Estimation of multiresponse simulation metamodels using control variates. *Management Science*, 35, no. 11, pp. 1316-1333
- [28] Rao, C.R. (1959), Some problems involving linear hypothesis in multivariate analysis. *Biometrika*, 46, pp. 49-58

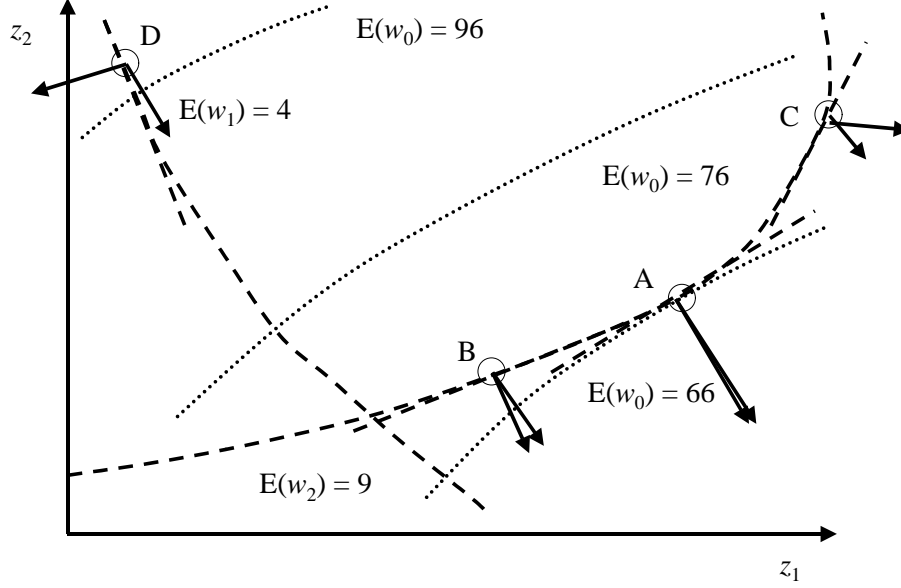


Fig. 1. Example 1: a constrained random optimization problem with second-order polynomial I/O functions

- [29] Rousseeuw, P.J. and A. Struyf (1998), Computing location depth and regression depth in higher dimensions, *Statistics and Computing*, 8, pp. 193-203
- [30] Safizadeh, M.H. (2002), Minimizing the bias and variance of the gradient estimate in RSM simulation studies. *European Journal Operational Research*, 136, no. 1, pp. 121-135
- [31] Santner, T.J., B.J. Williams, and W.I. Notz (2003), *The design and analysis of computer experiments*. Springer-Verlag, New York
- [32] Shapiro, A. (2000), Statistical inference of stochastic optimization problems. In: *Probabilistic constrained optimization: theory and applications*, edited by S.P. Uryasev, Kluwer, pp. 91-116
- [33] Van Beers, W. and J.P.C. Kleijnen (2003), Kriging for interpolation in random simulation. *Journal of the Operational Research Society*, no. 54, pp. 255-262
- [34] Wan, J. and J.P.C. Kleijnen (2006), Simulation for the optimization of (s, S) inventory system with random lead times and a service level constraint by using Arena and OptQuest. Working Paper, Hebei University of Technology

	large area large noise	large area small noise	small area large noise	small area small noise
A: (2.53, -1.99)				
constraints binding	69/1000 = 0.07	92/1000 = 0.09	69/1000 = 0.07	92/1000 = 0.09
fit of polynomial metamodel	79/931 = 0.08	77/908 = 0.08	79/931 = 0.08	77/908 = 0.08
linear combination of gradients	106/852 = 0.12	107/831 = 0.13	16/852 = 0.02	98/831 = 0.12
positive Lagrange multipliers	0/746 = 0.00	0/724 = 0.00	550/836 = 0.66	0/733 = 0.00
B: (2.00, -2.35)				
constraints binding	67/1000 = 0.07	101/1000 = 0.10	67/1000 = 0.07	101/1000 = 0.10
fit of polynomial metamodel	81/933 = 0.09	79/899 = 0.09	81/933 = 0.09	79/899 = 0.09
linear combination of gradients	232/852 = 0.27	820/820 = 1.00	17/852 = 0.02	210/20 = 0.26
positive Lagrange multipliers	0/620 = 0.00	0/0	541/852 = 0.63	0/610 = 0.00
C: (3.00, -1.10)				
constraints binding	75/1000 = 0.08	82/1000 = 0.08	75/1000 = 0.08	82/1000 = 0.08
fit of polynomial metamodel	73/925 = 0.08	77/918 = 0.08	73/925 = 0.08	77/918 = 0.08
linear combination of gradients	548/852 = 0.64	841/841 = 1.00	17/852 = 0.02	538/841 = 0.64
positive Lagrange multipliers	4/304 = 0.01	0/0	598/835 = 0.72	2/303 = 0.01
D: (1.00, -1.00)				
constraints binding	67/1000 = 0.07	67/1000 = 0.07	67/1000 = 0.07	67/1000 = 0.07
fit of polynomial metamodel	81/933 = 0.09	81/933 = 0.09	81/933 = 0.09	81/933 = 0.09
linear combination of gradients	847/852 = 0.99	852/852 = 1.00	33/852 = 0.04	847/852 = 0.99
positive Lagrange multipliers	5/5 = 1.00	0/0	735/819 = 0.90	5/5 = 1.00

Table 1

Example 1: fraction of rejected macro-replicates in the local areas centered around points A, B, C, and D

Team	Method	Symbol	s_o, S_o	costs (SE)	disservice (SE)
Kleijnen & Wan	Brute force	A	1020, 1075	624 (2.4).	0.10 (0.004)
	OptQuest	B	1021, 1077	625 (3.8)	0.10 (0.005)
Bashyam & Fu	Brute force		N/A	703 (N/A)	0.11 (N/A)
	PA & FD	C	1040,1065	708 (N/A)	0.11 (N/A)
Anglin et al.	Brute force	D	1160, 1212	647 (8.6)	0.11 (0.010)
	GRSM		1185, 1231	671 (N/A)	N/A (N/A)

Table 2

Example 2: optima estimated through three different research teams (SE: standard error; N/A: not available)

A	
constraints binding	$37/500 = 0.07$
fit of polynomial metamodel	$33/463 = 0.07$
linear combination of gradients	$11/430 = 0.03$
positive Lagrange multipliers	$44/430 = 0.10$
B	
constraints binding	$40/500 = 0.08$
fit of polynomial metamodel	$29/460 = 0.06$
linear combination of gradients	$6/431 = 0.01$
positive Lagrange multipliers	$47/431 = 0.11$
C	
constraints binding	$33/500 = 0.07$
fit of polynomial metamodel	$35/457 = 0.08$
linear combination of gradients	$21/432 = 0.05$
positive Lagrange multipliers	$48/432 = 0.11$
D	
constraints binding	$500/500 = 1.00$
fit of polynomial metamodel	N/A
linear combination of gradients	N/A
positive Lagrange multipliers	N/A
E	
constraints binding	$92/500 = 0.18$
fit of polynomial metamodel	$20/408 = 0.05$
linear combination of gradients	$3/388 = 0.01$
positive Lagrange multipliers	$68/388 = 0.18$

Table 3

Example 2: fraction of rejected macro-replicates for small noise and small local areas centered around points A through E