Quality Technology &
Quantitative Management
Vol. 6, No. 3, pp. 325-352, 2009

QTQM
© ICAQM 2009

Automated Response Surface Methodology for Simulation Optimization Models with Unknown Variance

Robin Nicolai and Rommert Dekker

Econometric Institute, Erasmus University Rotterdam, The Netherlands

(*Received* September 2006, *accepted* September 2007)

Abstract: Response Surface Methodology (RSM) is a collection of mathematical and statistical techniques useful for developing, improving, and optimizing processes. Applications of RSM can be found in *e.g.* chemical, engineering and clinical sciences. Still, there does not seem to be an established code of practice for the automated application of RSM in the field of simulation optimization. In this paper our aim is to find the best settings for an automated RSM procedure when there is very little information about the objective function. We present a framework of the RSM procedures for finding optimal solutions and emphasize the use of both stopping rules and restart procedures. Various versions of the RSM algorithms are compared on a number of test functions, including a simulation model for cancer screening. The results show that considerable improvement is possible over the proposed settings in the existing literature. Accordingly, we give general recommendations on the application of automated RSM algorithms in simulation optimization.

Keywords: Code of practice, estimation, Response Surface Methodology, simulation via optimization.

1. Introduction

Response Surface Methodology (RSM) is a tool that was introduced in the early 50´s by Box and Wilson [5]. RSM is a collection of mathematical and statistical techniques that is useful for the approximation and optimization of stochastic models. The objective function associated with such models is subject to random noise and is referred to as *noisy* or *stochastic* objective function. RSM is based on approximations of the objective function by a low order polynomial on a small sub-region of the domain. Using regression analysis based on a number of 'noisy' measurements of the objective function, the best local solution is determined together with a search direction for possible improvement. To this end, the objective is evaluated in a specific arrangement of points referred to as an experimental design. Many applications of the RSM procedure are performed in a manual setting, for example in physical, engineering, biological, medical and food sciences [17].

In a manual setting the user can interfere in the optimization process according to his/her personal intuition and likings. In an automated RSM optimization algorithm the settings of the algorithm have to be fixed in a systematic manner. Ideally, such an algorithm does not stop to ask for input from the user during an optimization run; instead it reads the input, performs a systematic search for an optimum and reports the optimum back to the user. The OPTQUEST [9] simulation optimization procedure operates in a similar way, yet it is primarily oriented at optimization of discrete decision variables and it uses other techniques.

Our interest is in finding the best settings for an automated RSM algorithm when there is very little information about the objective function. Here best means that the algorithm finds near-optimal solutions under all circumstances and has a demonstrated performance. Under consideration are noisy objective functions with unknown variance that are somewhat time-consuming to evaluate for each solution. This situation especially occurs when one optimizes a simulation model, that is, when one tries to find the model parameters that optimize specific stochastic output statistics of the simulation model. For this type of simulation models other simulation optimization methods are less practical, because they require many simulation runs (see Kleijnen [15]). RSM is a typical example of a meta heuristic. It considers the simulation model as a black box and only observes its input and output. The advantage of such a procedure is that the original simulation model can be left intact, while procedures like infinitesimal perturbation require software changes that are not desirable in case of very complex models. Since simulation models become more and more complex and have many input parameters, there is certainly need for fast automated RSM algorithms in simulation optimization (see Safizadeh and Thornton [23]).

For an automated RSM to be called successful it should be reliable, precise and fast. The procedure should recognize when no further progress is being made and the differences in the subsequent iterations can only be attributed to the noise in the objective function. Moreover, the procedure should also be able to distinguish optimal solutions from random fluctuations.

In this paper we present a framework of RSM procedures for finding optimal solutions in the presence of noise. It includes feedback iterations, precision checks (stopping rules) and a restart procedure. It iterates between first-order and second-order approximation models in order to continue the search for optima beyond the first-order approximation. Furthermore we study which settings and choices result in the best automated RSM procedure, both with regard to computing time and precision. In the literature on RSM, which is discussed extensively in Section 2, we found rather confusing and non-systematic recommendations for the settings of such a procedure. That is, there is no established code of practice for (automated) RSM algorithms. Such a code is especially important in medical applications (see Tan *et al.* [27]). In order to standardize the algorithm we fix some of the possible choices in the process based on the existing literature. Other choices are defined in a number of test algorithms that are compared using randomized deterministic test functions and a simulation model well-known in the medical literature.

The set-up of this paper is as follows. In Section 2 we present our framework and we extensively discuss the choices to be made in the RSM procedure. We fix a number of the settings based on pre-tests and previous literature. Other choices are subject to experiments and the design of the experiments is discussed in Section 3. The test functions and the simulation model are described in Section 4 and the results of our experiments are given in Section 5. Finally, in Section 6 we discuss the test results and give our recommendations.

2. Response Surface Methodology

Let us first give a description of the steps in RSM; emphasis is on the choices that need to be defined for an automated RSM algorithm. Without loss of generality, we discuss RSM for a minimization problem. Secondly, we assume that the objective function is subject to stochastic noise, and that we want to optimize the expected value of the stochastic output. Mathematically, the problem can be defined as

$$\min_{\xi} f(\xi) : D \to \Re,\ D \subseteq \Re^k. \tag{1}$$

Here $f(\xi) = E(F(\xi))$ is an unknown function of the decision variables $\xi = (\xi_1,..., \xi_k)$, with $F(\xi)$ being the stochastic output for given input ξ and E denoting expectation. In RSM $F(\xi)$ is referred to as response and the relation between the response and the *true* response function f is given by $F(\xi) = f(\xi) + \varepsilon$, where ε is a normal distributed error term having mean zero and unknown variance σ^2. The latter especially applies to stochastic simulation models, where the objective function can be seen as a black box that returns a randomized output value for a given input. A simulation optimization exercise aims to find the input parameters that result into the minimum output value, such as finding the order-up to level that minimizes total costs in an inventory model. That is, the stochastic output $F(\xi)$ is utilized in order to find the optimal value of the vector ξ (see *e.g.* Spall [26]).

We need to make some comments about above model and its assumptions. Firstly, note that the decision variables in problem (1) are continuous. Secondly, we consider an unconstrained optimization problem. Although we focus on this kind of problems, RSM can also be applied to constrained optimization problems. The interested reader is referred to Smith [24], Myers and Montgomery [18] and Angün [1]. Thirdly, since the simulation model is treated as a black box, variance reduction is only possible by applying more simulation runs.

The goals of RSM are approximating and mapping the response function over a particular region of the domain and determining the values of the decision variables, possibly under restrictions. The RSM procedure comprises two phases. In the first phase the objective function is locally approximated by a first-order polynomial and in the second phase by a second-order polynomial. For this purpose in both phases a region of interest (ROI) is defined, which is a sub-region of the domain. In order to approximate the objective function it is evaluated / measured in the points of an experimental design. These points are usually located on the borders of the region of interest. When the first-order model is adequate a steepest descent procedure is applied to find a new region of interest. Otherwise the RSM algorithm moves to the second phase. When a second-order model approximates the objective adequately, a stationary point is determined and classified, and then an appropriate action has to be taken. Usually the algorithm is terminated and the stationary point is returned. However, we will discuss why it is profitable to continue the algorithm beyond the second phase especially for a stochastic function with unknown error variance.

In particular we will define an extension where the algorithm returns to a first-order approximation in some cases. Stopping rules are based on the quality of the current solution rather than on a certain phase in the optimization process. Furthermore, a restart procedure is applied after the algorithm is ended for the first time. Figure 1 shows the optimization process on a global scale and it displays when a stopping rule is checked. The dotted arrows show the proposed extensions. The stopping rules applied are the same after first and second phase. In this perspective we define an iteration of the RSM algorithm as the run between two checks of the stopping criteria of the algorithm.

There are a number of settings that need to be implemented in an automated RSM procedure using a consistent decision rule. These settings can be divided into what we call building blocks, strategic choices, stopping rules and a restart mechanism. In order to arrive at an automated RSM procedure we therefore follow to a large part the steps of the framework proposed by Neddermeijer *et al.* [19]. We define each step in this framework either as a building block or as a strategic choice. The building blocks of the algorithm consist of well-defined procedures that can be used to determine the next move of the algorithm. Strategic choices of the algorithm determine the action taken when a building block returns a result. In the remainder of this section we formulate the new automated

RSM algorithm. In Sections 2.2 and 2.3 we present the building blocks and the strategic choices. Stopping rules and a restart mechanism are defined in Sections 2.4 and 2.5. In Section 2.6 a number of parameter settings in the algorithm is discussed. Ultimately the description in the next sections leads to the framework shown in Figure 2. The steps in the RSM algorithm shown in Figures 1 and 2 are now first illustrated by means of an example. The thick arrows give the corresponding route through the framework.

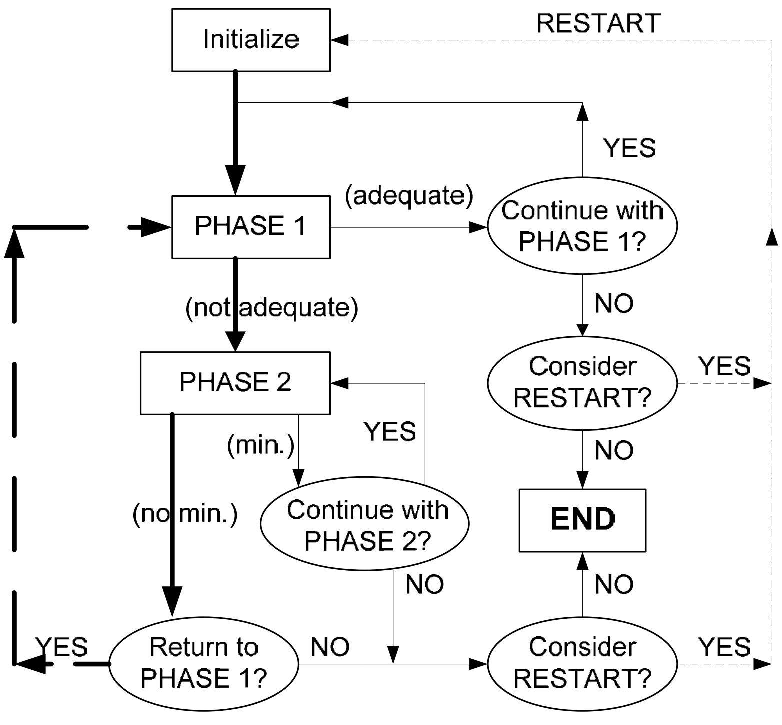


Figure 1. Schematic overview of the algorithm. Dotted arrows denote extensions. Thick arrows denote the route discussed in Section 2.1. Note: min. = minimum.

2.1. Illustrative Example

Let us consider the unconstrained minimization problem $\min_{\xi_1,\xi_2} E(F(\xi_1,\xi_2))$, where $F(\xi_1,\xi_2)=f(\xi_1,\xi_2)+\varepsilon$, $\varepsilon \sim N(0,1)$ and $E(\varepsilon)=0$. The true objective function is given by $f(\xi_1,\xi_2)=-10\exp\{-[(100-\xi_1)^2+(100-\xi_2)^2]/15000\}$ and takes its minimum in $(100,100)$ with corresponding value -10. In our problem setting the true objective function is unknown and we have to utilize the stochastic output $F(\xi_1,\xi_2)$ to find the optimal values of the decision variables ξ_1 and ξ_2. Let $(10,10)$ be the centre of the first region of interest (ROI) and let the half width of this square region be equal to 40. The vertices of the ROI are thus given by $(-30,-30)$, $(-30,50)$, $(50,-30)$ and $(50,50)$. The vertices and the centre point together form the points of a full factorial design. In the first phase of RSM the objective function is approximated by a linear model: $f(\xi_1,\xi_2)\cong\alpha_0+\alpha_1\xi_1+\alpha_2\xi_2$. The decision variables or *factors* ξ_i are coded by $x_i=(\xi_i-10)/40$, i=1, 2. The coded centre point is $(0,0)$ and the coded vertices are $(-1,-1)$, $(1,-1)$, $(-1,1)$ and $(1,1)$. The corresponding first-order approximation reads $f(x_1,x_2)\cong\beta_0+\beta_1x_1+\beta_2x_2$. In order to estimate the coefficients β_i, $i=1,2$, the objective function is evaluated according to a design once in each of the four vertices and five times in the centre point. Suppose $y^T=(-2.16,-1.82,$ $-1.72,-6.91,-3.61,-2.96,-2.13,-4.38,-3.68)$ is the vector of observed responses, where the last five values represent the five 'replicate runs' in the centre point, then the estimates of the coefficients of the coded variables are $(\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_2) = (-3.26, -1.21, -1.16)$. The replicate runs enable us to test for lack-of-fit. The test statistic equals 5.38 and since this does not exceed the critical value at 5% significance (6.94), we cannot reject the hypothesis that the linear model describes the response adequately. On the other hand the test for significance of regression indicates that we cannot reject the hypothesis that all coefficients equal zero (test statistic equals 3.14). In Section 2.3.2 we argue why it is wise to move to phase 2 now.

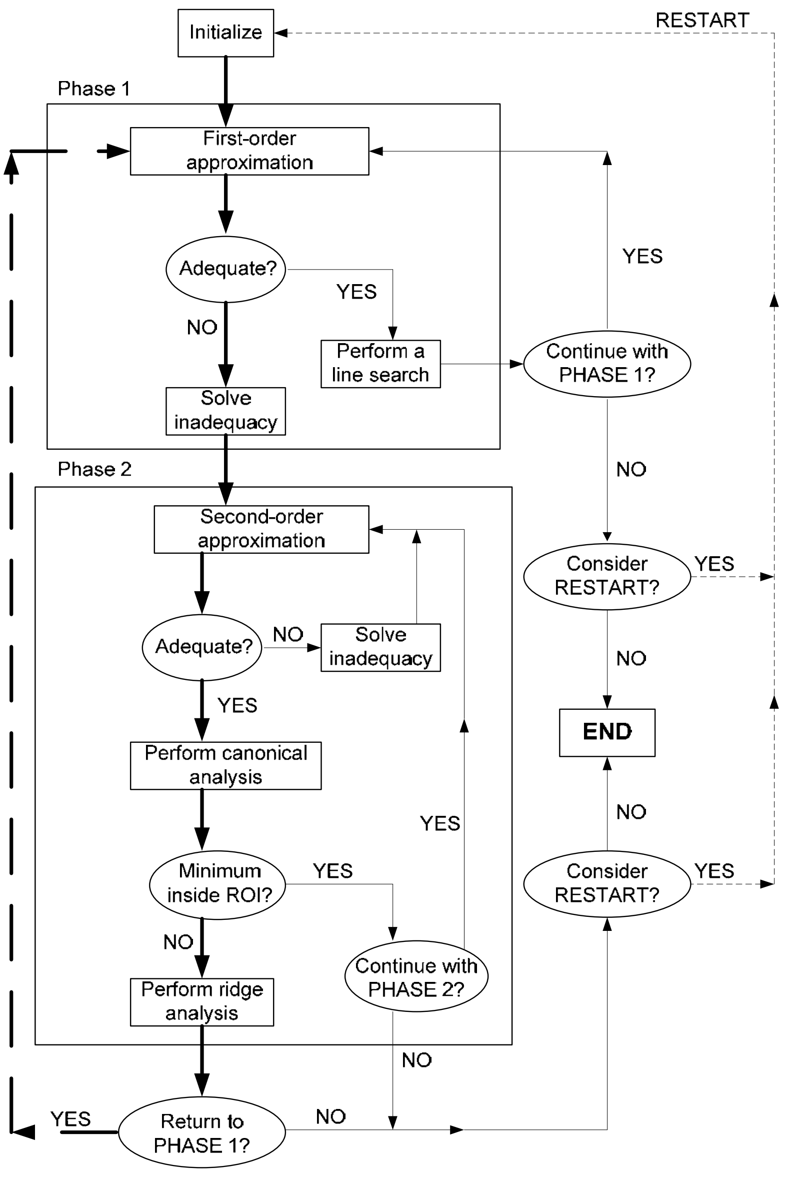


Figure 2. Framework for automated RSM algorithms with building blocks, strategic choices, stopping rules, and a restart mechanism. The building blocks are given by rectangles, the strategic choices are given by ellipses. The decision to continue or to restart the algorithm is also a strategic choice. Dotted lines denote extensions. Thick arrows denote the route discussed in Section 2.1.

In the second phase of the RSM algorithm the factorial design is augmented with four axial points: $(-\sqrt{2},0)$, $(\sqrt{2},0)$, $(0,-\sqrt{2})$ and $(0,\sqrt{2})$. Let $(-1.80,-5.44,-1.06,-5.82)$ be the response values in these points. Note that by adding the axial points the ROI becomes spherical; it is actually a circle with radius $R=\sqrt{2}$. The estimates of the coefficients of the second-order approximation $f(x_1,x_2)\cong\beta_0+\beta_1x_1+\beta_2x_2+\beta_{12}x_1x_2+\beta_{11}x_1^2$ $+\beta_2x_2^2$ now appear to be equal to $(\hat{\beta}_0,\hat{\beta}_1,\hat{\beta}_2,\hat{\beta}_{12},\hat{\beta}_{11},\hat{\beta}_{22})=(-3.35,-1.25,-1.42,-1.38,$ $-0.0394,0.0518)$. The test statistic of the lack-of-fit test equals 0.42 and does not exceed the critical value of 6.59 at 5% significance. The hypothesis that the model is adequate cannot be rejected (it also appears that the parameters are not all equal to zero) and we proceed with the canonical analysis. The stationary point $(-33.7,23.6)$ with predicted response -2.07 is a saddle point and we perform the ridge analysis to find a point of minimum response in the current ROI. Such a stationary point $\mathbf{s}_r$ should satisfy $\mathbf{s}_r'\mathbf{s}_r=R^2$ $=2$ and should minimize the estimated response in the ROI. Lagrangean analysis yields $\mathbf{s}_r=(1,1)$, which corresponds with the point $(50,50)$. Note that the predicted response value for this point is -7.41, whereas the actual function value is -7.17. The point $(50,50)$ is the centre point of the ROI in a possible next step.

Now, it is first checked whether the stopping criteria are met. Suppose that one of the stopping criteria (to be discussed in Section 2.4) is met, then the algorithm is either stopped or restarted (the topic of Section 2.5). We will point out in Section 2.3.3 that if none of the criteria is met and the stationary point is not a minimum of the second-order model, then the algorithm would better return to phase 1. The half width of the region of interest in phase 1 is then again 40 and the other 'natural' design points are $(10,10)$, $(10,90)$, $(90,10)$ and $(90,90)$. The procedure is iterated until a stopping criterion (see Section 2.4) is met.

2.2. Building Blocks

Figure 2 shows the building blocks of RSM in rectangles. Below we will discuss them in detail.

2.2.1. Initialization

At the start of the algorithm an initial starting point and the initial size of the region of interest (ROI) should be given. A restart may ask for a different initialization: the starting point may be chosen randomly, be equal to the starting point used in the previous optimization run or equal to the best point in the previous optimization run.

2.2.2. Approximate the Objective Function by a First-Order Model

In order to approximate the objective function it is evaluated in the points of an experimental design. There are many designs available, like fractional or full factorial, and two-level or three-level designs [18]. All designs can be augmented by the centre point of the region of interest. In non-automated optimization the user tries to fit a first-order approximation with different designs, apply coding of the variables to obtain better parameter estimates or recalculate the objective values in the design points. For instance, replicating the evaluation of the objective function in the centre point provides protection against curvature [18].

For an automated RSM procedure we follow the literature and evaluate the objective function once in the 2^k points of a two-level full factorial design and 5 times in the centre point of the current region of interest [11, 18]. This design is orthogonal and does not require as many points as higher level full factorial designs. Furthermore, full factorial designs can quite easily be augmented to derive a second-order design. When the number of parameters k is large (say $k > 10$) it is more efficient to use resolution III designs in the first phase [15, 18]. Eventually, the coefficients of the first-order model are determined by applying least squares on the observed function values, while using coded variables to reduce the covariances of the parameter estimates.

2.2.3. Test the First-Order Model for Adequacy

Usually, a test for lack of fit [29] and a test for significance of regression are performed [18]. Box and Draper [4] showed that the test for lack of fit is a joint test for interaction between factors as well as curvature. In non-automated optimization one can decide to use other tests and one can vary the significance levels based on the results from these tests. In automated optimization the levels should be fixed. This is subject to tests.

2.2.4. Perform a Line Search in the Steepest Descent Direction

If the first-order model is adequate a line search is performed from the centre point of the current region of interest in the steepest descent direction to find a point of improved response. Numerous implementations of the line search have been proposed (see [4, 11, 12, 14, 15, 18]). We use increments $\Delta_1, \ldots, \Delta_k$ along the path of steepest descent equal to the distance from the centre point to the point of intersection of the direction of steepest descent and the sphere given by $\sum_{i=1}^{k} \Delta_i^2 = 1$. In a manual RSM algorithm one can observethe results of a line search and use personal likings to stop the search. However, in automated optimization, the algorithm needs a stopping rule that recognizes the lack of improvement in response during the line search.

The most straightforward rule ends the line search when an observed value of the objective function is higher than the preceding observation [7]. We will not use this stopping rule, known as the 1-in-a-row rule, because it is very sensitive to the noise in the response function. In a similar way, the *n*-in-a-row stopping rule ends the line search when n observed values of the objective function are higher than the preceding observation. According to the Myers & Khuri stopping rule [16], the line search is ended when the mean response in a line search point, is significantly (statistically) higher than the mean response in a preceding line search point. This rule requires evaluating the response function in a line search point more than once, because the variance of the response is not known at the start of the algorithm. In our algorithms we use the small sample t test in order to compare the mean responses in different points. This statistical test is robust with respect to both non-normality and unequal variances (Wackerley *et al.* [28]). We test our version of the Myers & Khuri rule against the 3-in-a-row rule for our setting, where the variance in the objective function is not known a priori (contrary to the set-up by Del Castillo and Myers and Khuri).

2.2.5. Approximate the Objective Function by a Second-Order Model

The coefficients of the second-order model are again determined by regression analysis, applied to observations performed in an experimental design. A popular second-order design is the central composite design (CCD) [18]. The CCD arises when the full factorial design is augmented by adding $2k$ axial points [5]. We make this design spherical by choosing the axial points such that all points are equidistant from the centre point of the current ROI [18]. This design is near-rotatable [18]. A fully rotatable design ensures equal variance of the estimate of the mean response at points equidistant from the centre point. However, for a fully rotatable design the distance of the axial points to the centre point would be large as compared to the distance of the existing points to the centre point.

2.2.6. Test the Second-Order Model for Adequacy

This module checks if the second-order model describes adequately the behaviour of the objective function in the current region of interest. Similar to the first-order model a lack of fit test is performed. The null hypothesis of this test is that the true regression model is quadratic. In manual optimization one can use different significance levels or decide to overrule the outcomes of the lack of fit test. In automated optimization one has to determine these levels beforehand. The exact setting of the significance level is subject to tests.

2.2.7. Perform Canonical Analysis and, if Necessary, Ridge Analysis

In the canonical analysis the stationary point of the second-order model is located and classified. Ridge analysis is done when the stationary point is a maximum, a saddle point or a minimum outside the current region of interest. In the ridge analysis we look for a point of minimum response inside the current region of interest since it is not correct to extrapolate the second-order model outside the current region of interest [18].

2.3. Strategic Choices

In Figure 2 the strategic choices are given in ellipses. Although the choice whether to continue, to restart or to stop the algorithm is also a strategic choice, we discuss stopping rules and a restart mechanism separately in Sections 2.4 and 2.5.

2.3.1. What to Do When the First-Order Model is Adequate?

If the first-order approximation is adequate, then the steepest descent procedure defined in Section 2.2.4 will be applied. The output of this procedure is a new point, which is then used as the centre point of the next region of interest. On this new region, the objective function is approximated again by a first-order model [18].

2.3.2. How to Solve First-Order Model Inadequacy?

If the first-order model is not accepted, then there is some evidence of curvature or interaction between factors on the current ROI, or the regression coefficients are all equal to zero. Most references suggest approximating the response function by a second-order model (see *e.g.* Fu [8]; Myers and Montgomery [18]). Alternatively the number of simulation runs could be increased, resulting in more precise estimates of the objective. However this alternative is time-consuming and does not guarantee that the inadequacy is solved. Secondly, the size of the region of interest could be decreased, but pre-tests have shown that this is not beneficial in this stage of the procedure as it then gets stuck in non-optimal points. Our algorithm moves to the second phase if the first-order model is inadequate.

2.3.3. What to Do When the Second-Order Model is Adequate / How to Proceed after the Canonical Analysis?

If the second-order approximation is adequate, then the appropriate action depends on the location and the nature of the stationary point. It is shown [10] that for many functions a first-order model is inappropriate over a large percentage of the domain, so the algorithm can turn to the second phase quite early. The first stationary point found by a second-order approximation is therefore not likely to be the best point in the domain. If we stop the algorithm at this point [8, 13] the optimum could still be located far away from the current ROI. We consider two alternatives. (i) If a minimum is found inside the region of interest, then this point will be used as the centre point of a new design and a new second-order approximation will be performed. We suggest to reduce the size of the ROI, to what extent is subject to tests, and to continue with phase 2 since we assume that we are close to the minimum of the objective function. (ii) If the stationary point of the second-order polynomial is not a minimum inside the region of interest, then we perform ridge analysis to find a new stationary point (see Section 2.2.7). We conclude that we are not close to the optimum and return to phase 1. By considering these alternatives we now need stopping rules to decide when we are satisfied with the current solution. In Section 2.4 we discuss a number of stopping criteria and our tests will show the best stopping rules for specific functional forms.

2.3.4. How to Solve Second-Order Model Inadequacy?

If the second-order model is inadequate, the region of interest is too large or the stochastic nature of the function disturbs the approximation process. Stopping the algorithm at this point is only a good idea if there is an indication that the current centre point is close to the optimum. For example, if the observed function values have not improved significantly in the last few iterations and the best function value significantly differs from the function value in the starting point, then the algorithm might be ended. We come back to stopping the procedure in Section 2.4. If there is no such indication, the inadequacy is solved and the second phase done again. One way of solving the inadequacy of the approximation is increasing the precision used in evaluating a design point, i.e. variance reduction of the estimated response. This way the second-order polynomial will fit the objective function better. We could also reduce the size of the current region of interest like Joshi *et al.* [11] propose. Our algorithm either reduces the size of the current region of interest or increases the precision used in evaluating the points of the second-order design. This is subject to tests.

2.4. Stopping Rules

In most studies on RSM the use of stopping rules is not mentioned. Of course, in automated optimization the RSM algorithm needs to be ended by consistent stopping rules that do not end the algorithm before a good solution is found and also do not unnecessarily prolong the algorithm. In Section 2.3 we referred to the RSM literature where the optimization is ended after estimating only one second-order model (see Fu [8]; Kleijnen [13]). We recommend ending the optimization if either the estimated response value does not improve sufficiently anymore, or, in case there are budget constraints, if a fixed maximum number of (function) evaluations has been performed. In this section we motivate why these criteria are consistent and discuss in detail how they are incorporated in the automated algorithm. In our experiments we also consider the following stopping criterion: the algorithm is ended if the input values do not change sufficiently anymore, *i.e.* if consecutive centre points are close to each other.

2.4.1. The Estimated Response Does not Improve Sufficiently Anymore (IMPROVE)

Algorithms for finding the optimum of a deterministic function can simply be ended when the function value does not improve sufficiently in consecutive iterations. When optimizing stochastic objective functions though, one has to take noise into account. Because the variance of the response is not known at the start of the algorithm it is estimated by evaluating the response in the new centre point of the region of interest more than once. A statistical test has to decide whether the (stochastic) function values observed in the last few iterations have improved significantly. That is, can one attribute the difference between the mean responses in two centre points to noise? Notice that if the mean response does not decrease significantly in consecutive centre points one could still make progress. For instance the mean response may decrease from value 10.2 to 8.1 in 5 iterations, while in each separate iteration the change is not significant. We therefore implement the following criterion. Stop the algorithm if the mean response in the previous centre point does not differ significantly from the mean response in the penultimate centre point in n consecutive iterations. It is important to note that the penultimate centre point is only changed in case the mean response differs significantly from the mean response in the previous centre point. The number of iterations is subject to tests. Notice that this stopping rule is based on the stopping rules for the steepest descent line search. It makes use of elements of both the Myers & Khuri rule as well as the *n*-in-a-row rule. In terms of our example we would thus employ the small sample t-test to find out whether the mean response in $(10,10)$ differs from that in $(50,50)$.

2.4.2. Convergence of Input Values (CONVERGE)

Algorithms that are used to find the optimum of a deterministic function are usually ended if the input parameters of the function do not change anymore. Therefore, we propose to end the algorithm if the Euclidean distance between two consecutive centre points is small, *i.e.* less than $\varepsilon\sqrt{k}$, where ε is a small number and k is the number of parameters of the objective function. In this way, the precision of each estimated parameter will be approximately equal to ε. In our example the Euclidean distance between the last two candidate solutions equals

$$\sqrt{(50-10)^2+(50-10)^2} \approx 56.6\,.$$

2.4.3. Fixed Maximum Number of Function Evaluations (MAXEVAL)

Our interest in the RSM is especially intended for stochastic models where the evaluation of the corresponding stochastic objective function is expensive or time-consuming. Therefore, ending the algorithm after a maximum number of function evaluations is appropriate when there are budget constraints. Notice that this stopping criterion does not consider the noise in the objective function. In our example $9+4=13$ evaluations are performed.

The stopping rules discussed in this section are applied after both phases. It is then decided whether the algorithm is continued, restarted or really terminated.

2.5. A Restart Mechanism

Because RSM is a local search method there is no guarantee for finding a global optimum. The first centre point used in the RSM procedure is either selected by the user or randomly chosen and can influence the outcome of the procedure. Neddermeijer *et al.* [19] consider multiple starting points and/or multiple searches from the same starting point, when optimizing a stochastic objective function. In this study, we will use an adjusted mechanism that is based on these suggestions.

In order to escape from a non-optimal region the algorithm is restarted as soon as the algorithm is ended by one of the stopping criteria. The starting point of the restart is the best centre point of the 'normal run'. Because the algorithm cannot escape from a non-optimal region when the size of the region of interest is too small, the size of the region is reset to its initial value. The restart can now either confirm the quality of the solution already found or admit that there exists a better solution.

In general one should apply randomly chosen starting points when optimizing stochastic objective functions. If one has information about the unimodality of a function, it may not be necessary, especially if one has some idea about a good starting point. In the test problems we consider in this paper such was the case and we restart always in the same point. In this study we will run algorithms more than once starting from the same point. This can be seen as another restart mechanism and will give insight in the consistency of algorithms. We will come back to this in Section 5.

In all optimization runs the best solution, *i.e.* the parameter values for which the mean response of the stochastic objective function is best, will be remembered. So our restart procedure cannot deteriorate the solution found in the 'normal' optimization run. Note that since the response function is stochastic, it is possible that the mean response measured in a non-optimal point is better than the mean response measured in the real optimal point.

2.6. Parameter Settings

A number of parameter settings are subject to pre-tests. For instance, the lack of fit test can be performed at different significance levels. The size of the increment in the precision of the function evaluation in the design points is also subject to tests. Furthermore, it is tested by how much the size of the region of interest is decreased in order to solve the inadequacy of the second-order model.

3. Test Design

In a pre-testing phase we determine the parameter values that give the best performance with respect to running time and quality of the solutions returned. In our experiments we consider only choices that we feel have the most impact on the efficiency of the RSM procedure. We benchmark our set-up against the set-up by Fu [8] who stops after doing the second phase only once.

In particular we consider the following alternatives. We test the 3-in-a-row stopping rule against our version of the Myers & Khuri stopping rule for the steepest descent line search. Myers and Khuri [17] conclude that their stopping rule dominates the 3-in-a-row rule for stochastic functions with known variance. When the variance is not known it is not sure whether the Myers & Khuri rule is still as successful. One can expect that algorithms using this rule are less efficient since every point in the line search is evaluated 5 times.

When the second-order model is adequate and a minimum is found within the region of interest we assume that we are close to an optimal solution. We then shrink the region of interest with either 50% or 10%. These values are already set by pre-testing, but the difference between these shrink-percentages is important. When the size of the region of interest is decreased by 10% it will take more time to focus on a certain region that is suspected to contain the global optimum and the algorithm could be prolonged unnecessarily. However if we shrink the region of interest too soon, we are at higher risk of returning a non-optimal solution. The success of the alternatives will closely interact with the stopping rules and the restart procedure. For instance, we expect that the 50% shrink procedure (ensures fast convergence) in connection with a strict stopping rule with respect to noise (which avoids unnecessary prolongation and ensures fast convergence) and a restart procedure that returns to the original size of the region of interest (recognize quality of solutions) will perform well.

When the second-order approximation is rejected either the noise is dominant or the region of interest is too large. In order to solve this problem we either reduce the noise, *e.g.* by performing more simulation runs, or we reduce the size of the region of interest. We think that algorithms using noise reduction will give more accurate results since the quality of the estimates of the response function is increased. We expect the running time to be higher for these algorithms.

As a result we will test eight algorithms, using all proposed alternatives in combination with the others. For all these eight algorithms we will study the effect of the stopping rules and restart procedure explained in Sections 2.4 and 2.5.

The set-up of the experiments is now as follows. First we apply the algorithms for a large number of iterations and for each algorithm we record the relevant values for the application of the stopping criteria. Especially, at each iteration we record (the mean response in) the current (best) centre point, the number of function evaluations and the Euclidean distance between the current and the last centre point. We then decide on the exact setting of each stopping rule such that the average performance of each algorithm for a test set defined in the next section is "best". Each algorithm can have a different setting of the stopping rules for which it performs best.

In the first place, the criteria for the preferred performance of the stopping rules are based on the precision of the solutions compared to the deterministic values of the test functions. Precision is measured in two ways. On the one hand we look at the error of an optimization run, which is defined as the absolute difference between the true objective value in the true optimal point and the true objective value in the observed best point of the run. On the other hand we consider the distance of an optimization run, which is defined as the Euclidean distance between the true optimal point and the observed best point of the run. Notice that we have full knowledge of the optima of the test functions and corresponding solution values. We can therefore observe the parameters of each stopping rule of the RSM procedure for every test function and determine when the procedure is no longer effectively improving the solutions.

Secondly, the preferred performance of the stopping rule is also based on the running times of the algorithms needed to fulfill the different settings of the stopping criteria. We measure the running time by the number of function evaluations. Note that the standard deviation of the error and the distance of optimization runs can be used as an indicator of the consistency of an algorithm.

The settings of the algorithms are now taken such that the average performance, i.e. quality and running time, over all test functions is best. We do this because there is no a priori knowledge of the objective function. Note that each algorithm is applied on each test function 100 times to find the average behaviour of the algorithm.

In a second phase of experiments we run the algorithms again for the new settings of the stopping rules and we apply a restart procedure when each algorithm is stopped. The restart procedure is applied once and the result of the algorithm after the restart is returned as the output of the algorithm. We apply each algorithm 100 times on each test function. Note that the replication of an optimization run is in itself a restart procedure and therefore allows us to compare two different restart strategies. The performance of the eight algorithms is compared to the performance of four algorithms using the set-up of Fu [8], applied to the same set of test functions. Note that we deal with four algorithms since these algorithms are ended when the objective is approximated adequately by a second-order polynomial once. So, for these algorithms the third column of Table 1 ('Shrink region of interest') is of no importance. We call these algorithms the Fu versions of the algorithms in Table 1. These algorithms can be seen as examples of applying response surface techniques manually.

Table 1. The test design.

Algorithm	Stopping rule steepest descent	Shrink region of interest	Solve second-order model inadequacy
1	3-in-a-row	50%	Noise reduction
2	3-in-a-row	50%	Shrink design
3	3-in-a-row	10%	Noise reduction
4	3-in-a-row	10%	Shrink design
5	Myers & Khuri	50%	Noise reduction
6	Myers & Khuri	50%	Shrink design
7	Myers & Khuri	10%	Noise reduction
8	Myers & Khuri	10%	Shrink design

4. The Test Problems

We test the optimization algorithms on a set of seven test functions, which consist of a deterministic and a stochastic term. Their optima are known. These functions have also been used in comparing different versions of the Nelder and Mead simplex method (NMSM) and in comparing RSM algorithms with NMSM algorithms [20, 21]. A simulation version of an existing cancer-screening model is also considered. This model has three parameters that need to be estimated from an observed data set using minimization of a goodness-of-fit statistic [21]. Although for this particular model the optimal parameters can also be determined analytically, we are interested in the performance of the RSM procedure for this simulation model, as for more complicated medical simulation models one needs a standardized application of RSM.

4.1. One-Stage-One Test Breast Cancer Model

The simulation model is a simulation implementation of the breast cancer-screening model developed by Day and Walter [6]. In this model only one disease stage, the detectable pre-clinical phase (DPCP) is modelled. The DPCP has incidence rate J and it is assumed that the duration of the DPCP is exponentially distributed with parameter λ. At the end of the DPCP a cancer is clinically detected, whereas during the DPCP a cancer can be detected by breast cancer screening.

A screening program of four annual screening rounds is simulated. The sensitivity of the screening test is denoted by φ. In each run 50,000 individual life histories, including the disease processes and the impact of screening, are simulated. The simulation model generates detection rates at each of the screening rounds and incidence rates of clinical disease in the period following a negative screening test, for each of the screening rounds and for different intervals since the screening test.

The model is applied to data from the first randomized trial for breast cancer screening, viz. the HIP study [6, 22, 25]. In the HIP study approximately 62,000 women, who were aged between 40 and 64 at entry, were randomly allocated to either a study group or a control group. Only the study group was offered annual breast cancer screening for four years. About 65 percent of the study group (20,166 women) agreed to take part and were screened at least once (these women all attended the first screening). We will use follow-up data until 5 years after the last screening. The results from the HIP screening trial that will be used are described by Day and Walter [6], and consist of 4 detection rates and 14 incidence rates of interval cancers occurring after a previous negative test result.

The parameters J, λ, and φ are estimated from the observed data set through minimization of a chi-square goodness-of-fit test statistic. The simulation objective function is given by

$$F(J,\lambda,\varphi)=\sum_{i=1}^{18}(O_i(J,\lambda,\varphi)-E_i(J,\lambda,\varphi))^2 / E_i(J,\lambda,\varphi)\,, \qquad (2)$$

where O_i is the observed number of cancers during screening round or interval i and E_i is the number of simulated cancers during screening round or interval i, $i=1,\ldots,18$.

The true optimal parameters of the model for the HIP data were derived using the objective function

$$f(J,\lambda,\varphi)=\sum_{i=1}^{18}(O_i(J,\lambda,\varphi)-A_i(J,\lambda,\varphi))^2 / A_i(J,\lambda,\varphi)\,, \qquad (3)$$

where A_i is the number of cancers during screening round or interval i, $i=1,\ldots,18$, as predicted by the analytical implementation of the breast cancer screening model [6]. The optimal parameters $(J^*,\lambda^*,\varphi^*)$ of the model, applied to the HIP data, were determined by extensive enumeration (using step sizes 10^{-5} for J, 10^{-3} for λ, and 10^{-3} for φ) of $f(J,\lambda,\varphi)$: $f(J^*,\lambda^*,\varphi^*)=f(0.00213,0.614,0.871)\approx 13.343$.

The fact that we know the optimal parameter makes it attractive to apply our algorithms to this stochastic objective function. Assuming that the noise in the objective function due to simulation is approximately normally distributed, RSM is a suitable optimization method. Moreover, since many simulation runs are necessary to evaluate the stochastic objective function, it is worthwhile to apply an optimization method that does not need many function evaluations.

Table 2. The seven deterministic test functions.

Name	Functional form	Minimum
Rosenbrock	$f(\xi_1,\xi_2)=100(\xi_2-\xi_1^2)^2+(1-\xi_1)^2$	$f(1,1)=0$
Powell	$f(\xi_1,\xi_2,\xi_3,\xi_4)=(\xi_1+10\xi_2)^2+5(\xi_3-\xi_4)^2$ $+(\xi_2-2\xi_3)^4+10(\xi_1-\xi_4)^4$	$f(0,0,0,0)=0$
Parabolic		$f(0,0,0,0,0)=0$
Gaussian	$f(\xi_1,\xi_2)=-10e^{\{-[(100-\xi_1)^2+(100-\xi_2)^2]/15000\}}$	$f(100,100)=-10$
Asymmetric	$f(\xi_1,..,\xi_8)=\sum_{i=1}^{8}[2^{\xi_i-4}+(6-\xi_i)]$	$=23.311$
Beale	$f(\xi_1,\xi_2)=[1.5-\xi_1(1-\xi_2)]^2+[2.25-\xi_1(1-\xi_2^2)]^2$ $+[2.625-\xi_1(1-\xi_2^3)]^2$	$f(3,\ 0.5)=0$
Wood	$f(\xi_1,\xi_2,\xi_3,\xi_4)=100(\xi_2-\xi_1^2)^2+(1-\xi_1)^2$ $+90(\xi_4-\xi_3^2)^2+(1-\xi_3)^2$ $+10.1\{(1-\xi_2^2)^2+(1-\xi_4)^2\}$ $+19.8(1-\xi_2)(1-\xi_4)$	

4.2. The Test Functions

In addition to the HIP screening simulation model we test randomized versions of seven deterministic unconstrained nonlinear minimization problems. These functions have been extensively used in testing optimization algorithms. All problems have a unique global optimum. The test functions show a characteristic behaviour that may occur in stochastic objective functions resulting from simulation models. The deterministic test functions are randomized by adding a normal distributed error term with zero mean and variance $\sigma^2 = 1$. For each optimization run we use independent random number streams. The test functions are given in Table 2.

5. Results

Preliminary tests give us the best settings of the parameters we described in Section 2.6. Table 3 shows these settings.

Table 3. Parameter settings.

Parameter	Setting
Significance level of all statistical tests	5%
Shrink design for solving second-order inadequacy	50%
Increase precision for solving second-order inadequacy	0.95 / 25%

Notice that the last row of Table 3 contains two entries: 0.95 and 25%. In order to solve second-order model inadequacy, the variance of the error term is multiplied by 0.95 for the seven randomized deterministic test problems; for the simulation model the number of simulated life histories is increased by 25%. We expect that the latter has almost the same effect on the variance of the objective function of the simulation model.

5.1. Phase 1

In phase 1 of our numerical experiments we determine the exact settings of the stopping criteria. To this end we have recorded the results of the eight specified algorithms for a number of settings of the stopping criteria. These settings are shown in Table 4. Note that the settings of the stopping rules may depend on the number of parameters (*k*) of a test problem.

Table 4. Possible settings of stopping criteria, with k the number of parameters.

Stop Criterion	Setting	Unit / Quantity
IMPROVE	5, 10, 15, 20, 25	Number of consecutive iterations with non-signifcant change of function
CONVERGE	5E-04, 1E-03, 2E-03, 5E-03, 1E-02	Euclidean distance
MAXEVAL	$50\ 2^k$, $100\ 2^k$, $150\ 2^k$, $200\ 2^k$, $250\ 2^k$	Number of evaluations

In Tables 5(a)-(c) the results of the application of algorithm 1 on the Powell, Beale and Wood test functions are shown. The results for this algorithm and these test problems are representative for all experiments in phase 1 and show that the restart mechanism and good stopping rules are necessary.

Tables 5(a)-(c) show that the more iterations an algorithm runs the lower the error is. However, it appears that the Euclidean distance between the estimated parameters and the actual location of the minimum is not always smaller when the algorithm runs longer. This especially holds for the optimization of the Powell and the Beale function. This is probably due to the fact that these functions are very steep. We expect that the restart mechanism solves this problem as it enlarges the ROI.

It is striking that the standard deviations of the errors of the algorithms are roughly of the same order of magnitude as the errors themselves. This points out that the algorithms do not consistently perform well; a small part of the solutions found worsens the mean error. However, there is a high probability that a good solution is found when we an algorithm is applied more than once to the same problem.

Table 5(a). Results of applying algorithm 1 on the Powell function.

Stop Criterion	**SETTING**	**ERROR**	**DISTANCE**	**nEVAL**
IMPROVE	5	0.746 (0.60)	0.288 (0.086)	1628 (238)
	10	0.472 (0.41)	0.263 (0.075)	2113 (525)
	15	0.370 (0.37)	0.259 (0.080)	2735 (1253)
	20	0.302 (0.31)	0.260 (0.090)	3601 (1976)
	25	0.261 (0.27)	0.259 (0.091)	4562 (2686)
CONVERGE	1.0E-03	0.276 (0.35)	0.260 (0.092)	5542 (3061)
	2.0E-03	0.208 (0.27)	0.265 (0.091)	8864 (3707)
	4.0E-03	0.172 (0.15)	0.262 (0.078)	11634 (3807)
	1.0E-02	0.187 (0.16)	0.264 (0.078)	12898 (3258)
	2.0E-02	0.185 (0.14)	0.267 (0.095)	13598 (2341)
MAXEVAL	800	3731 (642)	3.94 (0.33)	800 (0)
	1600	0.665 (0.60)	0.280 (0.085)	1600 (0)
	2400	0.369 (0.38)	0.255 (0.081)	2400 (0)
	3200	0.278 (0.27)	0.254 (0.079)	3200 (0)
	4000	0.260 (0.27)	0.256 (0.081)	4000 (0)
Max. iterations	500	0.169 (0.17)	0.278 (0.11)	18103 (236)

Note: The first column in Tables 5a-5c contains the name of the stopping criteria and in the second column the setting of each criterion is given. The third column (ERROR) shows the mean (standard deviation) of the absolute differences between the optimal value and the true objective value in the solution found. In the fourth column (DISTANCE) the mean (standard deviation) of the Euclidean distances between the actual minimum and the solution found is given. The fifth column (nEVAL) shows the mean (standard deviation) of the number of evaluations needed to fulfill the different stopping criteria. The last row contains the results of ending the algorithm after the maximum number of iterations, in this case 150.

As it was stated before, the performance of an algorithm does not only depend on the precision of the solutions found, but also on the computing time. When optimizing stochastic objective functions the number of function evaluations needed to obtain a certain solution is an important indicator for the computing time. The values in the last column in Tables5a-c show that the difference in the number of function evaluations between the least restrictive setting and the most restrictive setting of stopping criteria CONVERGE and IMPROVE can be quite large. The extra running time needed to increase the precision does not always balance the extra computing time. For example, for the Beale function we find that the second setting of CONVERGE requires 569 evaluations on average. The precision of the solution found however, is not much worse than the solution resulting from ending the algorithm by the fourth setting of CONVERGE, requiring twice as many evaluations. Moreover, when the algorithm is ended after 150 iterations, requiring more than 3000 evaluations, the precision of the solution found is not much better. Thus, it only seems useful to prolong the optimization run for the Powell and the Wood functions.

The mean error, the mean Euclidean distance and the mean number of evaluations do not tell the whole story. Therefore we have also compared the 100 observed errors, distances and number of evaluations separately for all algorithms by using nonparametric statistical tests. This way we determined whether there is any stochastic difference between the different settings of the stopping criteria CONVERGE, IMPROVE and MAXEVAL. The specific nonparametric test we applied is the Kruskal-Wallis test (see *e.g.* Wackerley *et al.* [28]).

Table 5(b). Results of applying algorithm 1 on the Beale function.

Stop Criterion	**SETTING**	**ERROR**	**DISTANCE**	**nEVAL**
IMPROVE	5	0.329 (0.31)	1.88 (0.029)	281 (118)
	10	0.283 (0.19)	1.87 (0.035)	517 (269)
	15	0.266 (0.18)	1.88 (0.036)	805 (405)
	20	0.255 (0.17)	1.88 (0.045)	1038 (520)
	25	0.254 (0.17)	1.88 (0.044)	1218 (613)
CONVERGE	7.071E-04	0.319 (0.30)	1.88 (0.038)	354 (193)
	1.414E-03	0.264 (0.19)	1.88 (0.046)	569 (305)
	2.828E-03	0.249 (0.17)	1.88 (0.052)	863 (418)
	7.071E-03	0.241 (0.16)	1.88 (0.054)	1076 (458)
	1.414E-02	0.243 (0.16)	1.88 (0.054)	1266 (542)
MAXEVAL	200	0.347 (0.32)	1.88 (0.024)	200 (0)
	400	0.275 (0.20)	1.87 (0.035)	400 (0)
	600	0.263 (0.18)	1.87 (0.038)	600 (0)
	800	0.262 (0.18)	1.88 (0.042)	800 (0)
	1000	0.241 (0.16)	1.88 (0.056)	1000 (0)
Max. iterations	150	0.228 (0.14)	1.88 (0.074)	3272 (133)

The nonparametric tests do not give identical answers for the different test problems. It appears that for most of the test functions less restrictive settings of the three stopping criteria do not result in statistically lower errors and distances. That is, running the algorithms longer does not improve the precision of the solutions found. However, the errors and distances for the Wood and the Powell function and to a less extent the Beale and the Rosenbrock function are significantly lower when the settings are less restrictive. Of course the computing time, measured as the number of function evaluations, increases significantly when the settings of the stopping criteria are less restrictive. The results of the nonparametric tests are uniform over the different algorithms.

Table 5(c). Results of applying algorithm 1 on the Wood function.

Stop Criterion	**SETTING**	**ERROR**	**DISTANCE**	**nEVAL**
IMPROVE	5	4.96 (4.0)	0.533 (0.19)	1720 (402)
	10	2.76 (3.7)	0.396 (0.20)	2577 (735)
	15	1.99 (2.6)	0.348 (0.17)	3515 (1247)
	20	1.62 (2.1)	0.325 (0.16)	4588 (1941)
	25	1.50 (2.1)	0.315 (0.16)	5775 (2598)
CONVERGE	1.0E-03	16.7 (14)	0.930 (0.36)	1128 (586)
	2.0E-03	2.42 (4.0)	0.360 (0.22)	3507 (1724)
	4.0E-03	1.26 (2.0)	0.294 (0.16)	9183 (3780)
	1.0E-02	1.38 (2.0)	0.310 (0.16)	11109 (4137)
	2.0E-02	1.57 (2.2)	0.323 (0.17)	11366 (3945)
MAXEVAL	800	31.8 (25)	1.27 (0.45)	800 (0)
	1600	5.62 (4.0)	0.572 (0.18)	1600 (0)
	2400	2.64 (3.3)	0.395 (0.18)	2400 (0)
	3200	2.04 (2.9)	0.351 (0.18)	3200 (0)
	4000	1.67 (2.5)	0.327 (0.17)	4000 (0)
Max. iterations	500	1.06 (1.8)	0.277 (0.16)	18085 (252)

Since we want to find algorithms that optimize all test functions very precisely we cannot choose the settings of the stopping rules too restrictive (in phase 2). In Table 6 the settings of the rules IMPROVE and CONVERGE are given. These settings (see Table 4 for their definition) are used to end the algorithms in our second phase of experiments. We decided not to use MAXEVAL since this rule ends the algorithms on almost the same moment as IMPROVE. We prefer IMPROVE since it takes noise into account whereas MAXEVAL is a more rigid rule. As a consequence, we allow both IMPROVE and CONVERGE to terminate the algorithms in the second phase of experiments.

Table 6. Settings of the different stopping criteria for all test functions in phase 2.

Algorithm / **Stopping rule**	1	2	3	4	5	6	7	8
IMPROVE	5	4	5	4	5	4	4	4
CONVERGE	2	2	3	3	3	3	3	3
MAXEVAL	---	---	---	---	---	---	---	---

5.2. Phase 2

In phase 2 of the experiments we study both the eight algorithms with the settings as given in Table 6 and we test versions of these algorithms that are ended quite early. We expect that a comparison of both groups of algorithms gives us some understanding of the performance of our algorithms and in particular of the quality of our stopping rules. Moreover we test whether restarting the algorithm improves the precision of the solution.

In Table 7 the mean errors and the mean number of evaluations of the four Fu algorithms are shown. Note that we use the numbering of the algorithms as given in Table 1. This is done because the four Fu versions resemble the algorithms in Table 1, but they are ended when an adequate second-order model is estimated. In these algorithms there is no action taken when a minimum is found.

Table 7 shows that the restart mechanism improves the precision of the solution found after the 'normal' run considerably for most test functions and also contributes to a more consistent performance of the algorithm. For most test functions the quality of the solutions found in the normal run is quite bad. This is mainly due to the fact that the total number of evaluations is low in comparison with the most restrictive settings of our stopping rules, cf. Tables 5(a)-5(c). The Fu versions of the algorithms just do not perform enough function evaluations to find an accurate solution.

In columns 2-4 of Table 8 the mean errors resulting from the application of the eight different algorithms on the test functions are shown. Algorithm 3 serves as a benchmark and in general algorithm 5 finds the smallest errors and standard deviations. The latter algorithm ends the line search using the Myers & Khuri rule, shrinks the region of interest by 50% when a minimum is found and solves the second-order model by reducing the noise in the objective values.

Table 7. Results of applying the four Fu algorithms on all eight test functions.

	NO RESTART		**RESTART**	
Test Function	**ERROR**	**nEVAL**	**ERROR**	**nEVAL – add.**
Rosenbrock function	9.92 (86) [1]	107 (10)	1.11 (0.18) [6]	49.7 (102)
Powell singular function	19334 (0.0) [1]	34.6 (4.1)	18287 (0.0) [1]	36.9 (8.7)
Gaussian function	0.96 (1.16) [1]	37.7 (13)	0.24 (0.25) [1]	23.1 (8.5)
Parabolic function	735 (2924) [5]	1187 (282)	0.23 (0.17) [6]	160 (296)
Asymmetric function	2.40 (10.8) [6]	1688 (502)	0.33 (0.36) [1]	853 (605)
Beale function	1134 (746) [2]	42.9 (36)	620 (732) [5]	120 (155)
Wood function	4026 (3219) [2]	207 (264)	2184 (2965) [2]	230 (251)
Simulation model	0.094 (0.11) [5]	27.5 (9.6)	0.085 (0.108) [5]	25.0 (4.2)

Note 1: nEVAL – add. = additional number of evaluations.
Note 2: The first column of Table 7 shows the eight test functions. The second column shows the smallest mean error among the four Fu algorithms (numbered 1, 2, 5 and 6); error has been defined as the absolute difference between the true objective value in the true optimal point and the true objective value in the observed best point of the run. Superscripts denote the version of the Fu algorithm for which the minimum is attained. The third column shows the mean (standard deviation) of the number of evaluations performed by the algorithm without restart that yields the smallest mean error. The fourth and fifth column give the same performance measures for the algorithms with restart, but note that the fifth column shows the mean of the additional number of evaluations performed in the restart.

A comparison with Table 7 learns that the precision and consistency of the eight algorithms is higher than it is for the Fu versions. This can be proved by nonparametric statistical tests. On the other hand the Fu versions need significantly less function evaluations to find solutions.

Although the eight algorithms perform better than the Fu algorithms, these algorithms do not consistently perform well. Actually, it turns out that the algorithms sometimes get stuck in non-optimal points and in a number of optimization runs the first-and second-order models are repeatedly inadequate in the first few iterations. As a consequence the region of interest is decreased repeatedly and the algorithm is ended by one of the stopping rules. Of course, one or two of these bad solutions can really influence the mean error. We have to note here that in the point $(0, 0)$ one of the partial derivatives of the Rosenbrock function is equal to 0. The objective value in this point is 1. The algorithms are usually stuck in this point, but this is not caused by inadequate approximations of the functional form of the objective.

Table 8. Results of the eight algorithms using the new stopping criteria specified in Table 6.

	Mean error			**Mean of best 10 errors**			**Mean # of evaluations**		
Test function	**algo 3**	**best**	**avrg**	**algo 3**	**best**	**avrg**	**algo 3**	**min**	**avrg**
Rosenbrock	1.16	1.03 ^{5}	1.11	0.826	0.758 ^{5}	1.07	405	318 ^{4}	658
function	(0.27)	(0.16)		(0.070)	(0.157)		(157)	(123)	
Powell sing.	0.25	0.24 ^{5}	0.65	0.019	0.020 ^{1}	0.37	4239	2391 ^{4}	3833
function	(0.24)	(0.23)		(0.013)	(0.012)		(1951)	(820)	
Gaussian	0.17	0.16 ^{5}	0.17	0.003	0.003 ^{3}	0.12	662	568 ^{4}	957
function	(0.20)	(0.15)		(0.002)	(0.002)		(275)	(167)	
Parabolic	0.22	0.20 ^{5}	0.23	0.037	0.037 ^{3}	0.18	4968	2784 ^{4}	4694
function	(0.17)	(0.13))		(0.016)	(0.016)		(4035)	(1559)	
Asymmetric	0.20	0.18 ^{7}	0.27	0.032	0.030 ^{7}	0.18	31121	16269^{8}	2326
function	(0.16)	(0.15)		(0.012)	(0.011)		(21364)	(9588)	2
Beale	0.31	0.20 ^{5}	0.32	0.129	0.123 ^{5}	0.18	398	382 ^{4}	664
function	(0.25)	(0.10)		(0.002)	(0.005)		(165)	(145)	
Wood	4.77	0.81 ^{5}	14.5	0.083	0.077 ^{5}	7.60	2701	2138 ^{4}	3916
function	(7.8)	(0.87)		(0.039)	(0.050)		(1116)	(861)	
Simulation	0.061	0.061 ^{3}	0.081	0.004	0.003 ^{4}	0.05	202	184 ^{7}	389
model	(0.074)	(0.074)		(0.002)	(0.001)		(113)	(113)	

Note: The first column shows the eight test functions. Columns 2-3 contain the mean error (and the standard deviation of the error) of algorithm 3 (algo 3), which serves as a benchmark and the algorithm that yields the smallest mean error (best). Column 4 (avrg) shows the average of the mean error over the eight algorithms. Columns 5-7 contain similar performance measures as columns 2-4, but now with respect to the mean of the best 10 errors (out of 100). Columns 8-9 show the mean number of evaluations for algorithm 3 (algo 3) and for the algorithm that needs the smallest number of evaluations (min). Column 10 (avrg) contains the average of the mean number of evaluations over the eight algorithms. Superscripts denote the algorithm for which the smallest mean error or number of observations is obtained.

Columns 5-7 in Table 8 show the mean of the ten smallest errors resulting from the application of the eight algorithms on the test functions. Both the mean and standard deviation of the ten smallest errors are small compared to the mean and standard deviation of all errors. This indicates that the algorithms perform very well in at least 10% of the optimization runs. In general algorithms 3 and 5 find the most precise solutions. Note that if the objective function is unknown the only way to select the best solutions is to run many simulations for the solutions found. Although it may be somewhat time-consuming to perform a number of optimization runs and then select the best solutions, it can be really advantageous. This is a discussion point to which we return later.

In columns 8-10 of Table 8 the mean number of evaluations resulting from the application of the eight different algorithms on the test functions is shown. It can be seen that algorithm 5 needs the highest number of evaluations, whereas its natural counterpart, algorithm 4, needs the least number of evaluations. This is not surprising since algorithm 5 uses the Myers & Khuri rule to end the line search and this rule requires evaluating the objective function more than once in each point on the line. The individual results (not shown here) show that algorithms using this rule in general need more evaluations. This also holds for algorithms using noise reduction to solve second-order model inadequacy also in comparison with algorithms that shrink the region of interest. Moreover, algorithms that shrink the region of interest by 50% when a minimum is found need more evaluations than algorithms that shrink it by 10%. It is difficult to explain the latter results; we expected the counterpart since shrinking the region of interest by 50% a number of times results in a very small region.

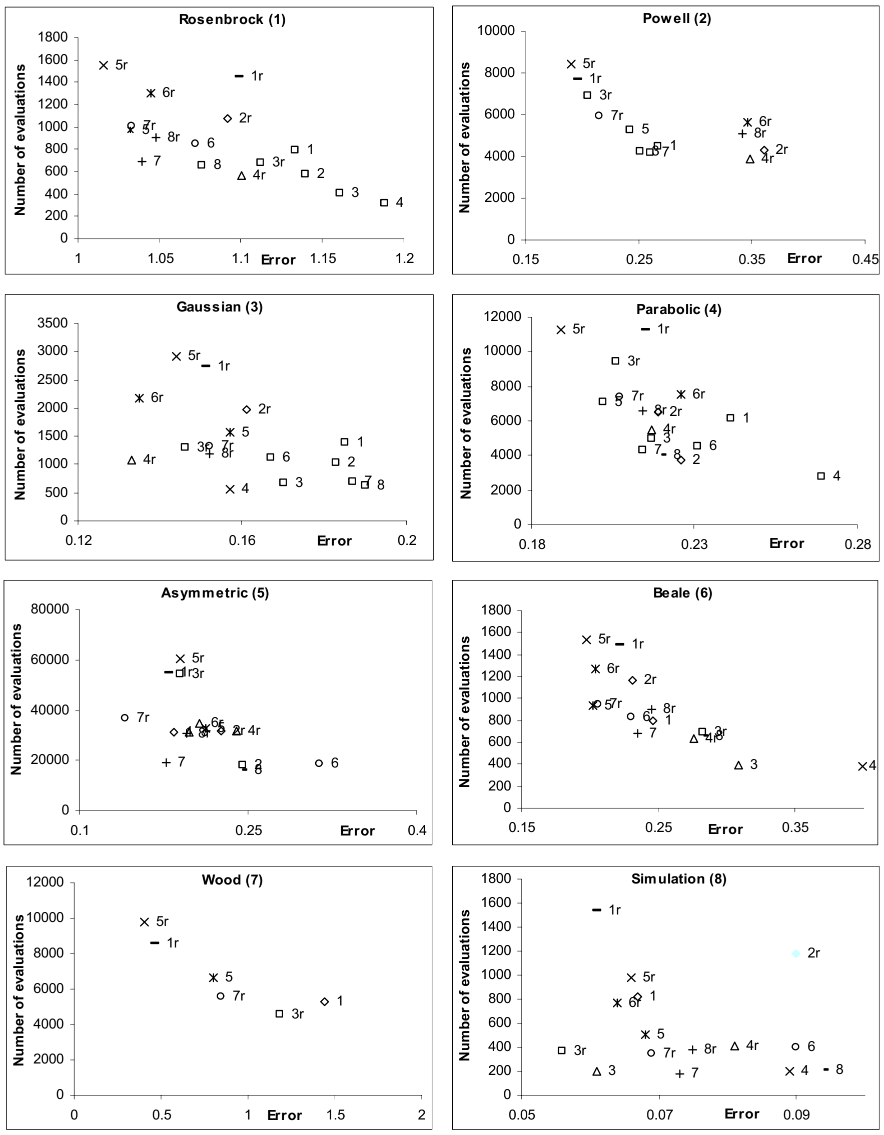
Table 9. Results of the eight algorithms using the restart mechanism and the new stopping criteria specified in Table 6.

	Mean error			**Mean of best 10 errors**			**Mean # of add. evaluations**		
Test function	**algo 3**	**best**	**avrg**	**algo 3**	**best**	**avrg**	**algo 3**	**min.**	**avrg**
Rosenbrock function	1.11 (0.21)	1.02 ^{5} (0.15)	1.07	0.805 (0.070)	0.731 ^{5} (0.146)	0.86	275 (151)	250 ^{4} (121)	410
Powell sing. function	0.21 (0.19)	0.19 ^{5} (0.18)	0.28	0.016 (0.012)	0.018 ^{5} (0.009)	0.04	2688 (1988)	1475 ^{4} (886)	2150
Gaussian function	0.15 (0.14)	0.13 ^{4} (0.15)	0.15	0.003 (0.002)	0.003 ^{3} (0.002)	0.01	642 (286)	517 ^{4} (229)	879
Parabolic function	0.21 (0.17)	0.19 ^{5} (0.12)	0.21	0.037 (0.016)	0.037 ^{3} (0.016)	0.05	4446 (4039)	2728 ^{4} (1612)	3494
Asymmetric function	0.19 (0.16)	0.14 ^{7} (0.11)	0.20	0.030 (0.011)	0.030 ^{7} (0.010)	0.04	23214 (19591)	14311 ^{8} (9591)	18581
Beale function	0.28 (0.24)	0.20 ^{6} (0.11)	0.23	0.128 (0.002)	0.122 ^{5} (0.004)	0.13	300 (185)	251 ^{4} (140)	415
Wood function	1.18 (2.67)	0.41 ^{5} (0.43)	2.91	0.057 (0.024)	0.029 ^{5} (0.019)	0.48	1856 (1357)	1584 ^{4} (903)	2154
Simulation model	0.056 (0.055)	0.056 ^{3} (0.055)	0.07	0.004 (0.002)	0.003 ^{4} (0.001)	0.0046	166 (113)	163 ^{7} (91)	357

Note: The first column shows the eight test functions. Columns 2-3 contain the mean error (and the standard deviation of the error) of algorithm 3 (algo 3), which serves as a benchmark and the algorithm that yields the smallest mean error (best). Column 4 (avrg) shows the average of the mean error over the eight algorithms. Columns 5-7 contain similar performance measures as columns 2-4, but now with respect to the mean of the best 10 errors (out of 100). Columns 8-9 show the mean number of evaluations for algorithm 3 and for the algorithm that needs the smallest number of evaluations. Column 10 (avrg) contains the average of the mean number of evaluations performed in the restart over the eight algorithms. Superscripts denote the algorithm for which the smallest mean error or number of observations is obtained.

In Table 9 some results of the application of the eight different algorithms using the restart mechanism on the test functions are shown. The Powell and Wood function benefit the most from the restart. Both the mean as well as the standard deviation of the errors is lower for these functions after applying the restart. Also, algorithm 5 finds the smallest errors and standard deviations. However, the standard deviations of the errors of this algorithm are most of the times of the same order of magnitude as the errors themselves. So, it also finds bad solutions.

Columns 5-7 in Table 9 show that the ten best solutions found are much better than the ten best solutions found in the original run of the algorithms. The mean errors of algorithms 3 and 5 are smallest. Columns 8-10 of Table 9 show that algorithm 4 needs the least number of evaluations to find a solution in the restart.



Figures 3(1)-(8): Mean number of evaluations versus mean error of algorithms.

Note: 1r denotes algorithm 1 using the restart mechanism.

In Figures 3(1)-(8) we have plotted the mean number of evaluations against the mean error of every algorithm for each test function. This way we can distinguish efficient algorithms, *i.e.* algorithms that find precise solutions in reasonable time. We call algorithms efficient when they are closer to the origin than other algorithms. Note that we did not plot all test functions in each Figure; if the error of an algorithm is too big the algorithm is not in the Figure. It appears that algorithm 5r (*i.e.* algorithm 5 using the restart mechanism) finds precise solutions for almost every test function, whereas algorithm 4 is the fastest algorithm. However, as we observed earlier, there is no particular algorithm that is really efficient, *i.e.* both fast and precise. It seems that algorithm 7 using the restart is most efficient among the algorithms we have studied concerning the average precision over 100 optimization runs. After all, this algorithm appears to be closest to the origin in most of the Figures. We also have to mention algorithm 5 since it is almost equally efficient for a number of test functions. Note that the only difference between algorithms 5 and 7 is the 'shrinking percentage'. Also note that the reader is allowed to determine for himself which algorithm is most efficient. The efficiency utility curve expressing how much value one assigns to a combination of precision and computing time, may differ among people. Some prefer fast algorithms with relatively low precision to relatively slow algorithms with high precision. Others may think precision and computing time are equally important.

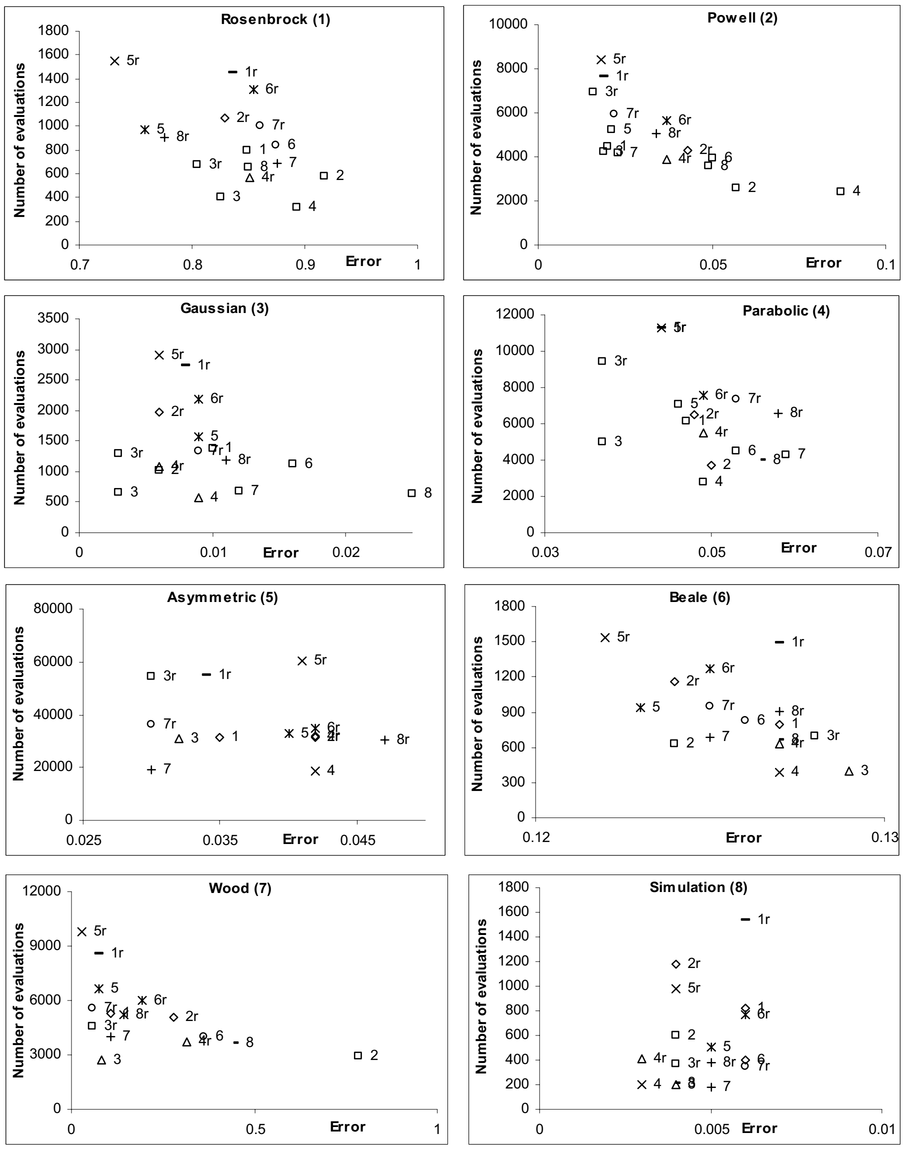
Figures 4(1)-(8) show plots of the mean number of evaluations against the mean error of the best ten solutions of every algorithm for each test function. Here we use the mean number of evaluations over all 100 optimization runs, since we need to do all runs before we can determine the best ten solutions. We mentioned earlier that the best ten solutions of algorithm 3 (without restart) are relatively precise. It appears that this algorithm does not need much more evaluations than algorithm 4, which is the fastest algorithm on average. So we may call algorithm 3 efficient among the other algorithms, in the sense that selecting the ten best solutions of this algorithm results in a high precision in reasonable time. Algorithm 3 uses the 3-in-a-row rule and noise reduction. The former setting yields a lower computing time, the latter a higher precision. Apparently this combination is efficient.

6. Discussion

In this section we discuss a number of important issues when using an optimization technique like RSM. First of all, the algorithms we have studied can serve as a benchmark algorithm in other studies. Here we aim at algorithm 5 in combination with the restart mechanism, since it is the most precise algorithm, but also at algorithm 3, which seems to be the most efficient algorithm concerning the ten best solutions. Moreover, algorithm 7 can be used as benchmark for testing the efficiency concerning all solutions.

Secondly, although the restart mechanism should not be regarded as a procedure that always recovers an algorithm when it is stuck in a bad point (solution), we have seen that the restart mechanism increases the precision of the algorithm for most test functions. The main lesson is that regularly enlarging the region of interest during the optimization run, may lead to better solutions.

An important question that is not fully answered yet is how we can employ time best to search the domain of the objective function. One possible answer to this question is by utilizing strict stopping criteria, which avoid unnecessary prolongation, and multiple restarts. Currently, the algorithm is restarted after it stops for the first time. In an alternative version the decision to restart may be based on the quality of the solution. Unfortunately, the only way to find out whether the current best point is 'good' is by comparing it with the starting point or points found in previous iterations, both with respect to the location and the function value of the points. As long as the function value improves significantly in subsequent runs, we may want to apply the restart mechanism. Combining strict stopping rules and multiple restarts may lead to better results than combining long optimization runs and few restarts. Especially when one does not have prior knowledge of the location of the optimum, the domain should first be explored. We have seen that lengthy optimization runs narrow the search down to a certain region too early. The exact number of restarts depends on the available time.



Figures 4(1)-(8): Mean number of evaluations versus mean error of best 10 solutions.
Note: 1r denotes algorithm 1 using the restart mechanism.

Thirdly, in this paper we have applied all algorithms 100 times to each optimization problem. We have seen that the algorithms do not perform consistently well. On the other hand determining the best ten of the 100 solutions shows that the algorithms also find very good solutions. Therefore it seems profitable to run an algorithm more than once and then select the best solutions. Selecting good solutions is not easy though; Boesel *et al.* [3] present an advanced method to select the best solution from all solutions visited during an optimization run. We feel that running an optimization algorithm 100 times and then re-evaluating the ten solutions with the lowest estimated response is a good way to find a good solution. Since we have an estimated response based on 5 function evaluations for every solution, we do not need additional evaluations for determining the best ten. Next, the ten best solutions are re-evaluated using more simulation runs than used in the optimization procedure. The resulting estimated responses are then compared using standard techniques.

Fourthly, in this study the experiments concerned only functions with few parameters. For functions with very many parameters either more efficient designs should be used in conjunction with RSM or other optimization methods such as stochastic approximation or Nelder and Mead should be considered. The latter method appears to perform quite well for bigger problem instances, concerning both precision and computing time (see *e.g.* Barton and Ivey [2]; Neddermeijer *et al.* [20]).

7. Conclusions

In this paper we worked towards a standardized automated RSM algorithm. The basis for this algorithm is the framework of Neddermeijer *et al.* [19]. We have extended this by introducing explicit decision rules, stopping rules and a restart mechanism. Consequently, various settings for such an automated RSM procedure for simulation optimization have been compared. We studied the efficiency (precision vs. computing time) of the different algorithms for optimization of a set of test problems, including a simulation model for cancer screening. Below we summarize our findings.

- The proposed RSM algorithms are suitable for optimizing objective functions with up to eight variables quite precisely in reasonable time. Especially algorithm 3 appears to be efficient in this respect. The best ten solutions of this algorithm are of high quality; the number of evaluations needed to find these solutions is quite low.
- The test problems considered in this study have unimodal objective functions, meaning that they do not have local optima. Nevertheless it appears that the RSM algorithms sometimes get stuck in non-optimal solutions. Generally, this occurs when the objective function is very flat in comparison with the associated noise on a certain region of the domain. This problem can be tackled by using the restart mechanism, but no global solution is guaranteed. Resetting the region of interest to its initial size, *i.e.* an enlargement of this region, and starting the search all over again, gives the algorithm the opportunity to escape from regions where the objective function is comparatively flat.
- A number of refinements has been tested. It appears that some of the modified RSM procedures, *e.g.* the algorithms using noise reduction and the Myers & Khuri rule, find more precise solutions. Unfortunately, using these settings has a drawback: the computing time measured as the number of function evaluations, is higher for these algorithms.
- It appears that rerunning the algorithms can lead to very different results. The standard deviations of the error of the algorithms are sometimes, even when the restart mechanism is applied, of the same order of magnitude as the errors themselves. That is, the quality of the solutions found is not consistently high. On the other hand,

the restarting procedure gives more confidence in that a good solution is found. Thus, running the algorithm with restart ten times, already gives a very high probability of finding a good solution.

- The proposed stopping rules help to recognize when the RSM algorithm makes no further progress. In the first phase of our experiments we noticed that most of the algorithms were terminated somewhat too early, but this is primarily because the region of interest was decreased too early. Combining comparatively strict stopping rules with a restart seems to be more efficient than doing one lengthy optimization run. This is a topic for further research.
- No stopping rule seems to outperform the other two. Therefore we recommend to implement a combination of stopping rules like it was done in Section 5.2. As soon as there is a sign that there is no improvement being made the algorithm should be stopped and possibly restarted.
- Finally, the traditional RSM algorithms (Fu versions) are terminated too early and consequently their solutions are not very accurate. The proposed extended algorithms can find better and more stable solutions, but this is at the expense of additional computing time. In the end the practitioner should decide for himself whether the gain in precision is high enough. Our study may contribute to this decision.

Acknowledgments

The authors would like to thank Nanda Piersma and Gerrit van Oortmarssen who initiated this research project. We also thank Alex Koning for his useful suggestions regarding the statistical aspects of this paper. We are also indebted to the editor and two anonymous referees whose comments improved this manuscript considerably.

References

1. Angün, E. (2004). *Black Box Simulation Optimization: Generalized Response Surface Methodology*. PhD thesis, Tilburg University.
2. Barton, R. R. and Ivey, J. S. (1996). Nelder-Mead simplex modifications for simulation optimization. *Management Science*, 42(7), 954-973.
3. Boesel, J., Nelson, B. L. and Kim, S.-H. (2003). Using ranking and selection to `clean up' after simulation optimization. *Operations Research*, 51, 814-825.
4. Box, G. E. P. and Draper, N. R. (1987). *Empirical Model-Building and Response Surfaces*. John Wiley & Sons, New York.
5. Box, G. E. P. and Wilson, K. B. (1951). On the experimental attainment of optimum conditions. *Journal of the Royal Statistical Society*, Series B, 13, 1-38, discussion: 38-45.
6. Day, N. E. and Walter, S. D. (1984). Simplified models of screening for chronic disease: estimation procedures from mass screening programmes. *Biometrics*, 40, 1-14.
7. Del Castillo, E. (1997). Stopping rules for steepest ascent in experimental optimization. *Communications in Statistics. Simulation and Computation*, 26(4), 1599-1615.
8. Fu, M. C. (1994). Optimization via simulation: a review. *Annals of Operations Research* 53, 199-247.
9. Glover, F., Kelly, J. and Laguna, M. (1999). New advances for wedding optimization and simulation. *Proceedings of the 1999 Winter Simulation Conference*, 255-259. Orlando, FL, USA.
10. Greenwood, A. G., Rees, L. P. and Siochi, F. C. (1998). An investigation of the behavior of simulation response surfaces. *European Journal of Operational Research*, 110,

282-313.

11. Joshi, S., Sherali, H. D. and Tew, J. D. (1998). An enhanced response surface methodology (RSM) algorithm using gradient deflection and second-order search strategies. *Computers and Operations Research*, 25(7/8), 531-541.
12. Khuri, A. I. and Cornell, J. A. (1996). *Response Surfaces: Designs and Analyses.* Marcel Dekker, New York.
13. Kleijnen, J. P. C. (1998). Experimental design for sensitivity analysis, optimization, and validation of simulation models. *In Handbook of Simulation: Principles, Methodology, Advances, Applications and Practice* (Edi. J. Banks). John Wiley & Sons, New York, 173-223.
14. Kleijnen, J. P. C., den Hertog, D. and Angün E. (2004). Response surface methodology's steepest ascent and step size revisited. *European Journal of Operational Research*, 159, 121-131.
15. Kleijnen, J. P. C. (2006). *Generalized Response Surface Methodology*. CentER Discussion Paper 2006-77, Tilburg University, The Netherlands.
16. Myers, R. H. and Khuri, A. I. (1979). A new procedure for steepest ascent. *Communications in Statistics. Theoretical Methods*, Series A, 8(14), 1359-1376.
17. Myers, R. H., Khuri, A. I. and Carter Jr, W. H. (1989). Response surface methodology: 1966-1988. *Technometrics*, 31(2), 137-157.
18. Myers, R. H. and Montgomery, D. C. (1995). *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*, 2nd edition. John Wiley & Sons, New York.
19. Neddermeijer, H. G., van Oortmarssen, G. J., Piersma N. and Dekker R. (2000a). A framework for response surface methodology for simulation optimization. *Proceedings of the 2000 Winter Simulation Conference*, 129-136. San Diego, CA, USA.
20. Neddermeijer, H. G., van Oortmarssen, G. J., Piersma, N., Dekker R. and Habbema, J. D. F. (2000b). *Adaptive Extensions of the Nelder and Mead Simplex Method for Optimization of Stochastic Simulation Models.* Econometric Institute Report EI2000-22/A. Erasmus Universiteit Rotterdam, The Netherlands.
21. Neddermeijer, H. G., van Oortmarssen, G. J., Piersma, N., Habbema, J. D. F. and Dekker R. (1999). *Comparison of Response Surface Methodology and the Nelder and Mead Simplex Method for Optimization in Micro Simulation Models.* Econometric Institute Report EI-9924/A. Erasmus Universiteit Rotterdam, The Netherlands.
22. Oortmarssen, G. J. van, Habbema, J. D. F., Lubbe, J. Th. N. and van der Maas, P. J. (1990). A model-based analysis of the HIP project for breast cancer screening. *International Journal of Cancer*, 46, 207-213.
23. Safizadeh, M. H. and Thornton, B. M. (1984). Optimization in simulation experiments using response surface methodology. *Computers & Industrial Engineering*, 8(1), 11-28.
24. Smith, D. E. (1976). Automatic optimum-seeking program for digital simulation. *Simulation*, 27, 27-31.
25. Shapiro, S., Goldberg, J. D. and Hutchinson, G. B. (1974). Lead time in breast cancer detection and implications for periodicity of screening. *American Journal of Epidemiology*, 100, 357-366.
26. Spall, J. C. (2003). *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control.* John Wiley & Sons, Hoboken (NJ).
27. Tan, S. Y. G. L., van Oortmarssen, G. J. and Piersma, N. (2003). Estimating parameters of a microsimulation model for breast cancer screening using the score function method. *Annals of Operations Research* 119, 43-61.
28. Wackerley, D. D., Mendenhall, W. and Scheaffer, R. L. (1996). *Mathematical Statistics with Applications*, 5th Edition. Duxbury Press, Belmont (CA).
29. Weisberg, S. (1985). *Applied Linear Regression.* John Wiley & Sons, New York.

Authors' Biographies:

Robin P. Nicolai works as advisor risk analysis and safety at HKV Consultants. He obtained his M.Sc. degree in Econometrics from Erasmus University Rotterdam. In March 2008 he successfully defended his Ph.D. thesis on maintenance optimization. His research interests are deterioration modelling, discrete event systems and simulation optimization. He has published in *Reliability Engineering & System Safety* and co-authored two chapters in Complex System Maintenance Handbook: Blending Theory with Practice.

Rommert Dekker is a full-time Professor in Operations Research, Quantitative Logistics and IT at Erasmus University Rotterdam. He obtained his Ph.D. in operations research at the State University of Leiden and his M.Sc. degree in industrial engineering from Twente University of Technology. His research interests include logistics, transportation and maintenance optimization in general. He has an extensive experience with simulation and analytical methods. He has published over 80 papers in scientific journals. His web address is www.few.eur.nl/few/people/rdekker.