Journal of Global Optimization **18:** 143–163, 2000.

On Uniform Covering, Adaptive Random Search and Raspberries

ELIGIUS M.T. HENDRIX[1] and OLIVIER KLEPPER[2]
[1]*Department of Mathematics, Agricultural University Wageningen, Dreijenlaan 4, 6703 HA, Wageningen, The Netherlands (e-mail: eligius@rcl.wau.nl)*
[2]*Sterpolis Insurance, P.O. Box 9225, 6800 KG, Arnhem, The Netherlands*

(Received 28 August 1998; accepted in revised form 19 January 2000)

Abstract. The problem of generating a random sample over a level set, called Uniform Covering, is considered. A variant is discussed of an algorithm known as Pure Adaptive Search which is a global optimisation ideal with a desirable complexity. The algorithm of Uniform Covering by Probabilistic Rejection is discussed as an approach to the practical realisation of PAS. Consequences for the complexity and practical performance in comparison to other algorithms are illustrated.

Key words: Random methods; Stochastic global optimization; Adaptive random search; Uniform covering; Efficiency

1. Uniform covering

The discussions and analyses in this paper originate from the confidence region questions in parameter estimation. In parameter estimation a goodness of fit objective function describes how well a model fits data given values for a parameter vector. The global minimum of this function corresponds to values for the parameters which fit best, are most likely to be the 'real' parameter values. In parameter estimation and nonlinear regression the confidence region contains parameter values which are 'nearly as likely' as the optimal one. The confidence region coincides with a level set of the goodness of fit objective function, see e.g. Bates and Watts (1988), Ross (1990), Donaldson and Schnabel (1987) and Hendrix (1998). In general the confidence region, level set, is approximated by an ellipsoidal region based on the Jacobian in the minimum. This approximation can be poor when the underlying model is highly nonlinear, as illustrated by Klepper and Hendrix (1994) with a simple case of logistic regression and in Klepper and Bedaux (1997) with an application of a toxicological threshold model. This happens in general when the level set is banana shaped. Many researchers in parameter bounding have raised the question what is a good way to represent the level set, see e.g. Walter and Piet-Lahanier (1990) and Donaldson and Schnabel (1987). One way to do so is to generate parameter-values from the level set of the goodness of fit function, as if they are sample points from a uniform distribution. This is the question we look at. How can points from a uniform distribution, a so-called random sample, be

generated over a level set without having to evaluate the function (model run) too many times? We look into a number of stochastic global optimization methods accomplishing this.

We use the following symbols, common in literature on optimization methods. A goodness of fit function $f(x)$ is minimized over a robust, compact feasible set $X \subset \mathbb{R}^n$, usually a box region. Variable y represents the level of the function f and $S(y) = \{x \in X \mid f(x) \leqslant y\}$ a level set. $V(\cdot)$ denotes the volume of a set. Evaluating $f(x)$ implies running a large model, so one would like to do that as few times as possible for reaching the target. The target, called *Uniform Covering*, is to generate a sample of N points from a uniform distribution over a level set $S(\alpha)$ with a predefined level $\alpha > y_* = \min_X f(x)$.

Two criteria are important:

Effectiveness: The set of N points should represent a sample of a distribution ''as uniform as possible''. There are various methods to measure this uniformity.

Efficiency: One should use as few function evaluations as possible.

The research on these criteria was stimulated by the idea of Uniform Covering by Probabilistic Rejection (UCPR) of Klepper and Hendrix (1994), to represent a level set, confidence region. The idea is outlined in Figure 1. A set of N points $P = \{p_1, \ldots, p_N\} \subset S(\alpha)$ can be used to generate a new point in $S(\alpha)$ in the following way.

UCPR method to generate a random point in $S(\alpha)$ given a set P of points

1. Repeat
   generate a random point x_{trial} in X
   Until $\exists\, i \|x_{\text{trial}} - p_i\| \leqslant r$
2. Calculate $f(x_{\text{trial}})$
   if $f(x_{\text{trial}}) \leqslant \alpha$
   x_{trial} is in $S(\alpha)$
   else go to 1

The parameter r is, for reasons to become clear, usually taken as the average interpoint distance, the average nearest neighbour distance. So for every point p_i the distance to the nearest other point in P is determined and the average is taken over all these distances. The idea behind the UCPR approach is, that it is much cheaper in function evaluations than a Pure Random Search variant, where every random generated point is evaluated. The respberry set

$$R = \{x \in X \mid \exists\, p_i \in P, \|x - p_i\| \leqslant r\}$$

should:

(1) cover set $S(\alpha)$ well (preferably $S(\alpha) \subset R$) to ensure effectiveness; usually there are holes in R, $S(\alpha)$ may not be fully covered.

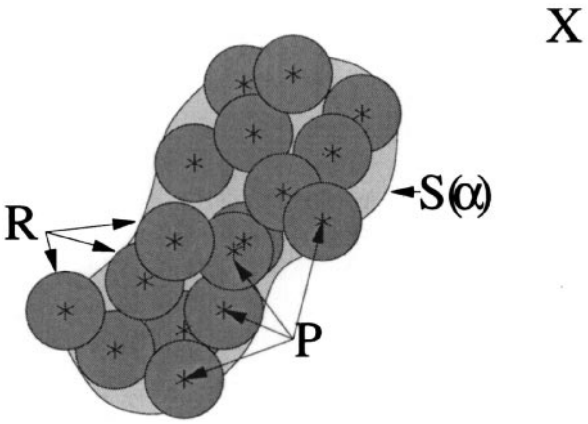

Figure 1. Uniform covering by probabilistic rejection.

(2) not stick too much out of set $S(\alpha)$ to ensure efficiency, so that function evaluations are not performed in vain.

Those aspects are analyzed in Section 3, in which complexity is discussed. We do not use the paradigm of formal proofs, but try to make results from literature plausible by using extreme cases. Moreover, we show why ideals originating from assumptions on the algorithms are improbable to be reached; a property of ideals in general. In Section 2, we look at stochastic algorithms for global optimization from the literature, which can be used. In Section 4, a numerical evaluation of the methods follows.

2. Algorithms

For the generation of the set of N points in $S(\alpha)$, several bench-mark methods in stochastic global optimization can be used.

Pure Random Search (PRS) is one extreme. It consists of generating points from a uniform distribution over X and checking whether the function value is lower than α. An algorithm for finding N such points are the following.

PRS algorithm for generating N points of $S(\alpha)$

$P = \varnothing$
Repeat
    generate x_{trial} from a uniform distribution over X
    if $f(x_{\text{trial}}) \leqslant \alpha$
    add x_{trial} to P
Until $|P\| = N$

Effectiveness of PRS is 100% in the sense that set P will consist of a sample from a uniform distribution over $S(\alpha)$. The efficiency is obviously not very good, as the expected number of function evaluations to obtain one point in $S(\alpha)$ is $V(X)/V(S(\alpha))$, see among others Törn and Žilinskas, 1989.

Pure Adaptive Search (PAS) is not a real implementable algorithm, but a tool for analysis on complexity and some sense an ideal, see Patel et al., 1989. The analysis in literature focuses on the question of what would happen if we could sample a point at each iteration in the improving region, the level set of the current iterate. The following scheme shows how this ability could be used to generate one point in $S(\alpha)$.

PAS algorithm for generating one point of $S(\alpha)$

0. Generate x_1 uniformly from X, $k := 1$
1. $y_k := f(x_k)$
2. If $y_k < \alpha$ STOP
3. Sample a point x_{k+1} from a uniform distribution over $S(y_k)$
4. $k := k + 1$, go to Step 1

The most important property, shown among others by Patel et al. (1988) and by Zabinsky and Smith (1992), is that in some sense the number of iterations grows less than exponential in the number of variables of the problem.* In Section 3, we will try to make this plausible to the reader and show why it is improbable that this ideal will be reached. I.e. it is unlikely that Step 3 of the algorithm can be performed in a time which grows polynomially in the dimension n of the problem. The ideal of PAS could be used to generate N points in $S(\alpha)$ by repeating the procedure N times. One could argue that it is more efficient not to restart the procedure for every point to be generated starting at Step 0, but to restart from a lower level set $S(y_k)$, with level y_k reached in earlier iterations closer to the level of α. However, we have to keep in mind that PAS is a hypothetical benchmark algorithm. In order to compare this extreme to some other algorithms, we now discuss a population variant of the algorithm.

***N* points PAS** has been introduced by the authors to analyze what happens, if we would be able to perform PAS with a population N points simultaneously.

* To be precise, the point x_{k+1} should be strictly improving in Step 1 of the algorithm for this property, i.e. $f(x_{k+1}) < y_k$.

NPAS algorithm for generating N points of $S(\alpha)$

0. Generate a set P of N random points $\{p_1, \ldots, p_N\}$ in X, $k := 1$
1. Determine $ymax_k = f(pmax_k) = \max_j f(p_j),\ p_j \in P$
2. If $ymax_k < \alpha$, STOP
3. Generate and evaluate a point in the interior of $S(ymax_k)$
   add it to the sample and remove $pmax_k$
4. $k := k + 1$; go to Step 1

We will make it plausible that the complexity is of the same magnitude as that of PAS.

The method of Price, or **Controlled Random Search** (CRS), was introduced (Price, 1979) as a simple global optimization heuristic. It was one of the first algorithms which uses a population of points. At every iteration, from the population of N points, $m + 1$ points are taken at random. In the first versions of the algorithm, m was taken as the dimension n. One can consider the choice of m as a parameter which influences the performance of the algorithm, which goes beyond the scope of our study. A new point x_{trial} is generated in a Nelder-Mead fashion by reflecting point p_{m+1} over the centroid of the remaining so-called mirroring points $p_1, \ldots, p_m$.

CRS algorithm for generating N points of $S(\alpha)$

0. Generate a set P of N points uniformly over X, $k := 1$
1. Determine $ymax_k = f(pmax_k) = \max_j f(p_j),\ p_j \in P$
2. If $ymax_k < \alpha$ STOP
3a. Take $m + 1$ points, $p_1, \ldots, p_{m+1}$, from P

3b. $$x_{\text{trial}} := \frac{2}{m} \sum_{1}^{m} p_j - p_{m+1}$$

3c. If $f(x_{\text{trial}}) < ymax_k$
   x_{trial} replaces $pmax_k$ in P
4. $k := k + 1$ and go to 1

The algorithm as given here can be seen as an approximation of NPAS. Note, however, that in contrast to the ideal, not necessarily at every iteration (function evaluation) an improving point is found.

The controlled random search algorithm, due to its simplicity, has been popular for solving practical problems, e.g. Garcia et al. (1997), Klepper and Rouse (1991). This practical use contrasts with the unpopularity of the algorithm by researchers on the theory of global optimization algorithms; there is no analysis possible on the performance of the algorithm, nor can anything be said on how good the final population resembles a sample from a uniform distribution over $S(\alpha)$. Klepper and

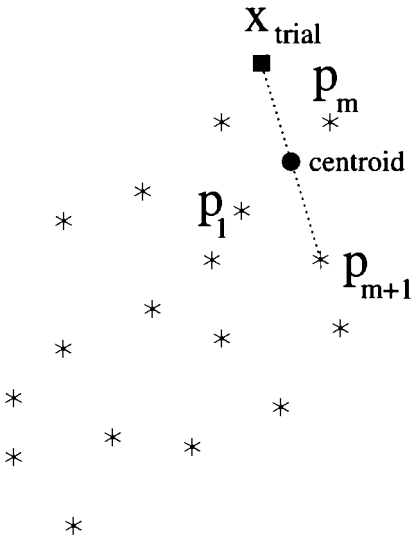

Figure 2. Generation of a new point by CRS.

Rouse (1991) have generated cases for which the resulting sample is unlikely to be from a uniform distribution over the level set (effectiveness). It can be shown that the mirroring procedure tends to bias the cloud of N points towards a radially symmetrical subset of the level set, especially for a high value of m. We will illustrate the numerical performance in Section 4.

The **UCPR** idea can be used to derive a specific algorithm which can be seen as another attempt to approximate the ideal of NPAS. Generating a point in the interior of a level set is of course the crucial practical point for PAS as well as for NPAS. By simple rejection, the complexity is that of PRS. Uniform Covering by Probabilistic Rejection (UCPR) can be seen as an attempt to come close to generating points uniformly in the level set.

UCPR algorithm for generating one point of $S(\alpha)$

0. Generate N random points $P = \{p_1, \ldots, p_N\}$ in X, $k := 1$
1. Determine $ymax_k = f(pmax_k) = \max_j f(p_j)$, $p_j \in P$
2. If $ymax_k < \alpha$ STOP
3a. Calculate the average interpoint distance r_k

$R_k := \{x \in X | \exists p_i \in P, \quad \|x - p_i\| \leqslant c^* r_k\}$

3b. Generate and evaluate a point x_{trial} from a uniform distribution over R_k.
3c. If $f(x_{\text{trial}}) < ymax_k$
x_{trial} replaces $pmax_k$ in P
4. $k := k + 1$; go to Step 1

Calculation of ray r_k in Step 3a for reasons to become clear, is done by determining the average nearest neighbour distance. So for every point p_i the distance to the nearest other point in P is determined and the average is taken over all these distances. The choice of the distance parameter c is discussed in 3.1 and 3.2 and explored in numerical tests. As in the NPAS algorithm the set $S_k = S(ymax_k)$ defines the current level set. This set is globally covered by the raspberry set R_k, from which new candidate points are generated. Stated in other words, in contrast to PRS no points are evaluated that are too distant from the current point set. The evaluation of points is assumed to include far more computational effort than the generation of random points. Therefore, the efficiency of an algorithm is measured as the number of function evaluations required. Notice again, that similar to the CRS algorithm, not every function evaluation in Step 3 leads to an improvement. This is caused by R_k sticking out of $S(ymax_k)$. When the generation of random points in UCPR is done in a rejection way, as suggested in Section 1, the set X can be replaced by a rectangular box around R_k. A further refinement is to rotate the box along the principal axes of the current point set. The algorithm is illustrated by Figure 3.

Hit and Run (H&R) is a way of generating points in an adaptive random search algorithm. The random process proceeds as follows. At the current iterate a direction is generated from a uniform distribution over the unit sphere. The possible step-range is determined in that direction and a step-length is generated from a uniform distribution over the possible step-range.

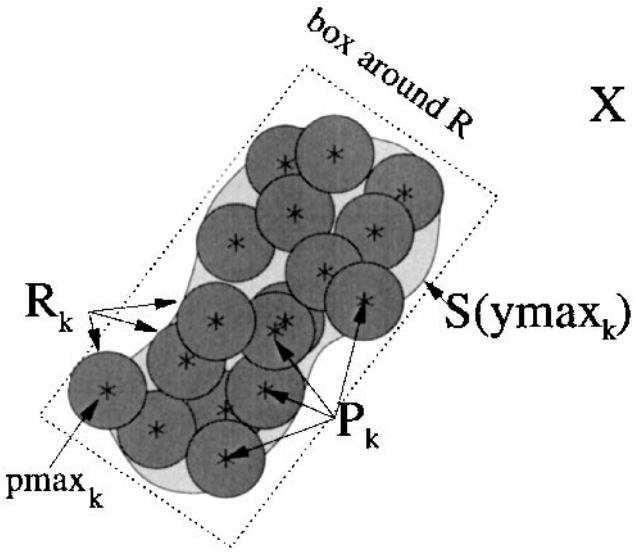

Figure 3. Elements of the UCPR algorithm.

Hit and Run process for generating points on X

0. $k := 1$, $x_1 \in X$ is starting point
1. Generate d_k from a uniform distribution over the unit sphere
3. Generate λ_k from a uniform distribution over $\{\lambda \in \mathbb{R},\ x_k + \lambda d_k \in X\}$
4. $x_{k+1} := x_k + \lambda_k d_k$
5. $k := k + 1$, go to 1

The process was studied among others by Smith (1984), who showed that for $k \to \infty$ the points x_k are uniformly distributed over X. This resembles the target we are looking for. Therefore, we will analyze the method, construct an algorithm and use this variant in the numerical illustration. In global optimization the process was studied by Romeijn (1992) and Romeijn and Smith (1994) who used it in a simulated annealing context. Zabinsky et al. (1993) studied the efficiency of the algorithm.

There are many ways to use the Hit and Run (H&R) process to generate points in a level set. We define a population variant here, where one point of the population is

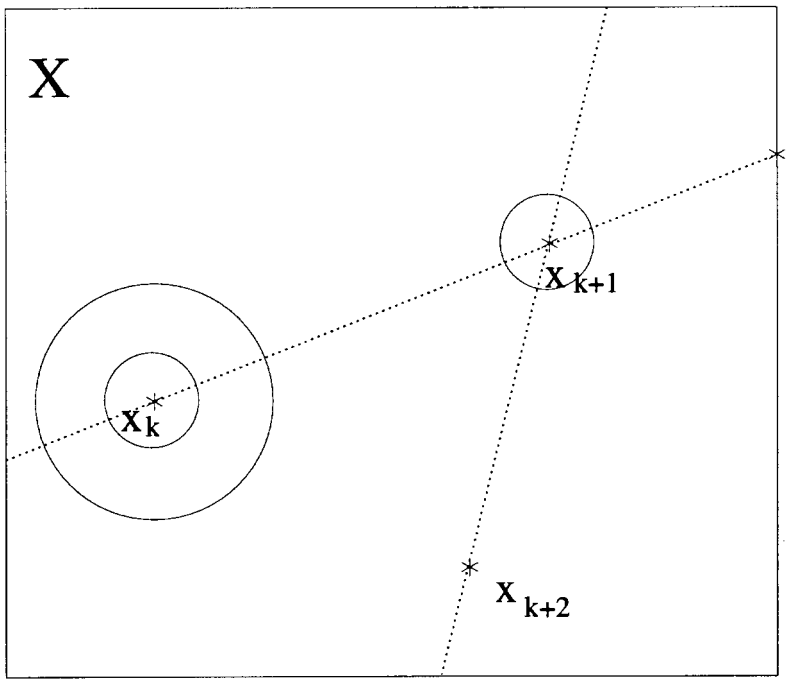

Figure 4. Hit and Run process.

used as a starting point to generate new points until it fails to find a point in the level set, a so-called hit. In the numerical examples, a constant W is defined to determine the level α defined as $\alpha = W \times y_*$. In this algorithm explicit use is made of W by defining an estimate $ymin_k$ of y_* and in this way updating the level of acceptance α whenever a better (record) point is found. Again the sequence is initialised by first drawing N points uniformly in X. The lowest function value in this set is used to estimate y_*. At random one of the points in the set which is also in $S(\alpha)$ is used to start H&R. Occasionally the estimate for y_* and the corresponding acceptance level α is adjusted. Notice that this procedure differs from the former algorithms, where the focus is on $ymax_k$. One could think of many alternatives to construct a population algorithm which uses the Hit and Run process; this is just one of them.

Algorithm for generating N points of $S(\alpha)$ based on Hit and Run process

0. Generate a set P of N points uniformly over X, $k := 1$
1a. Determine $ymin_k = \min_j f(p_j)$, $p_j \in P$
1b. $\alpha = W \times ymin_k$
1c. Remove p_j from P with $f(p_j) > \alpha$
2. If $|P| = \mathrm{N}$ STOP
3a. Select at random a point, p from P as starting point for Hit and Run
3b. Repeat generate x_{trial} with one Hit and Run step, $k := k + 1$
    If $f(x_{\mathrm{trial}}) < \alpha$ add x_{trial} to P and use it as starting point
    else go to Step 3a
    Until $|P| = \mathrm{N}$ or $f(x_{\mathrm{trial}}) < ymin_k$
4. go to 1

Theoretically the algorithm may stop too early when $W \times ymin_k$ exceeds the predefined level (not estimated) of α. However, in the experiments we use in Section 4, it never did.

3. Analysis on the complexity of the algorithms

In this section some complexity results, presented in literature, are discussed and the relation to the algorithms in Section 2 is given. We will not follow the paradigm of formal proofs, but instead try to make results plausible. As a tool, some extreme problems are used.

First problem $Q1$ is introduced:

$$\begin{aligned} &\min f(x) = \|x\| \\ &\text{on } X = \{x \in \mathbb{R}^n \mid \|x\| \leqslant 1\}. \end{aligned} \tag{$Q1$}$$

For PRS, the probability of obtaining a point in $S(\alpha)$ is exponentially related to n; for problem $Q1$ exactly:

$$V(S(\alpha))/V(X) = \alpha^n .$$

This probability goes to zero exponentially when the dimension n increases. The expected number of function evaluations to obtain a point in $S(\alpha)$ therefore also increases exponentially in the dimension n. This efficiency gives a kind of upper bound on the expected number of function evaluations necessary to obtain a random sample on $S(\alpha)$.

As stated before, PAS gives a kind of ideal. It was shown by Zabinsky and Smith (1992) that for problems satisfying the Lipschitz condition, the expected number of iterations grows linearly in the dimension. The expected number of iterations is shown to be bounded by

$$1 + n \times \ln(L \times d/\alpha) ,$$

in which L is the Lipschitz constant and d the diameter of the feasible area X. Before going further on the impact of this result for complexity issues and raising the question ''what happens with L and d in the formula when the dimension n grows'', we try to make the result plausible by exercising with problem $Q1$. Problem $Q1$ makes the analysis as simple as possible, as both Lipschitz constant L as well as diameter d are constant for increasing dimension.

For PAS we focus on the improvement per iteration in terms of the volume reduction of the level set $V(S_{k+1})/V(S_k)$. At a certain iteration a level y_k with corresponding level set $S_k = S(y_k)$ has been reached. Let $\boldsymbol{x}$ be a random variable uniformly distributed over S_k and $\boldsymbol{y} = f(\boldsymbol{x})$ the corresponding random function value. For problem $Q1$, random variable $\boldsymbol{y}$ has the c.d.f. $F_k(y) = y^n/y_k^n$. In every iteration of PAS, the volume $V(S_k)$ of S_k is reduced to $V(S_{k+1})$, which is also a random variable. The expectation of the reduction is

$$E\{V(S_{k+1})/(S_k)\} = E[\boldsymbol{y}^n/y_k^n] = \frac{1}{y_k^n}\int_0^{y_k} y^n dF_k(y) = 1/2 .$$

On average in every iteration half of the volume is thrown away. The same result based on a completely different analysis was found by Baritompa and Steel (1996). Because the reductions are independent and identically distributed, the expected reduction after k iterations is a multiplication $(1/2)^k$ of the expected reduction of the volume of $1/2$.

If the volume reduction would in every iteration be the expected value, the necessary number of iterations to obtain one point in $S(\alpha)$ (with relative volume α^n) is at least $n \times \ln(\alpha)/\ln(1/2)$. Ignoring the variation in the reduction leads to linearity of the number of iterations in the dimension. Generating N points by running PAS N times, if realisable, would require a number of iterations which is linear in N as well as n.

Now we focus on the impact of this linearity property. The theorem says that if

we are able to perform sampling in the improving region (Step 3 in the PAS algorithm) in polynomial time, we are able to solve some types of problems in polynomial time. To realise the consequence, we construct problem $Q2$.

Let g be an arbitrary Lipschitz continuous function defined on the unit cube $X = [0, 1]^n$ with a given Lipschitz constant ε ($\varepsilon < 1$). So for all binary solutions ν_j, ν_k $(j, k = 1, \ldots, 2^n)|g(\nu_j) - g(\nu_k)| \leqslant \varepsilon \times \sqrt{n}$. To find the minimum of g over all binary solutions ν_j, it is necessary to check all 2^n feasible vertices of the unit cube, so that the optimum cannot be found in polynomial time. Now we construct $Q2$ as a Global Optimization equivalent of this binary problem:

$$\begin{aligned} &\min f(x) = g(x) + \sum x_i(1 - x_i) \\ &X = \{0 \leqslant x_i \leqslant 1,\ i = 1, \ldots, n\} \end{aligned} \qquad (Q2)$$

Problem $Q2$ has been constructed such that in every vertex ν_j, $g(\nu_j)$ equals $f(\nu_j)$ and that all of the 2^n vertices of X are local optima of $Q2$. To realise this one should see that a direction into the feasible area from an arbitrary vertex is an ascent direction. Given any direction pointing into the feasible area, the directional derivative of the penalty $\Sigma\, x_i(1 - x_i)$ namely dominates ($\geqslant 1$) the directional derivative of g ($> -\varepsilon$). Therefore every vertex is a (strict) local minimum.

So $Q2$ is equivalent to a binary programming problem. To solve $Q2$, a calculation time is necessary which grows exponentially in the dimension n. Considering the PAS result for $Q2$, we see that $L = \varepsilon$ is constant and the diameter d of X grows with $\sqrt{n}$. If we were able to get a point in the improving region $S(y_k)$ in a polynomial calculation time (Step 3), the expected number of function evaluations is bounded by a polynomial to solve complex problem $Q2$. We would be able to solve an NP hard problem in an expected number of iterations which grows polynomially in the dimension of the problem. It is improbable that we will be able to do so.

The great 20th century philosopher Karl Popper (1902–1994) taught us how science proceeds and how it concerns problems; ‘‘we start with a problem’’. By further analyzing one problem, we get better insight and arrive at the next problem. In this context we consider the PAS result as a progress in the Popperian way of thinking. We started with the problem

– Can we solve global optimization problems in polynomial time?

By logic steps (mathematics) the next question was derived:

– Can we generate points from a uniform distribution in a level set (given one point at the boundary) in polynomial time?

The latter problem remains worthwhile to look into.

A following achievement in the line of the PAS result, which stimulated our analysis, is the appearance of a paper by Baritompa et al. (1995) called ‘Towards Pure Adaptive Search’. In this paper it was shown that PAS can be relaxed by requiring that there is a fixed probability ρ that the next iteration point is uniform in the improving region, called ρ-adaptive search. The same complexity bound holds as that of PAS. Again by reasoning reversely, we can conclude that it is unlikely that

such a fixed probability exists. The idea of ρ-adaptive search was called by the authors of the paper Somewhat Adaptive Search. Considering UCPR as an approximation of PAS, we were tempted to call a paper on UCPR 'Towards Somewhat Adaptive Search'. Namely, if sampling in R_k leads to uniform points in S_k and the failure rate, frequency with which one samples outside S_k, would be bounded by a fixed proportion not depending on the dimension, the UCPR algorithm would be a realisation of ρ-adaptive search and with that an algorithm with an expected polynomial solution time. As we hope to have made clear, this is unlikely to happen. A more recent elaboration on adaptive search can be found in Bulger and Wood (1998) under the terminology of hesitant adaptive search. We will elaborate on uniform covering and failure rate of the UCPR algorithm after first considering the complexity of the N-points variant of PAS.

In the NPAS algorithm the current level set S_k is defined by the maximum $ymax_k$ of the function values of N uniformly distributed points over S_{k-1}. In the next iteration a new point is generated and evaluated within S_k. The new level set S_{k+1} is determined by the new maximum of N points in S_k; $\boldsymbol{z} := \max(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_N)$.

Let again $F_k(y)$ be the c.d.f. of the function value of a point randomly chosen (uniformly) in level set S_k. The c.d.f. of extreme order value $\boldsymbol{z}$ is given by $\Phi(z) = [F_k(z)]^N$. Specifically for problem $Q1$ this implies that

$$\Phi(z) = (z/ymax_k)^{nN} .$$

The expectation of the reduction in volume of the level set for NPAS is

$$E\{V[S(z)]/V[S(ymax_k)]\} = \frac{1}{y_k^n}\int_0^{y_k} z^n \, \mathrm{d}\Phi(z) = \frac{N}{N+1} .$$

So at every iteration the volume of the level set is reduced by a factor (expected value) $N/(N+1)$. For $N = 1$ this gives the PAS result. Neglecting the variation in this stochastic reduction leads to the conclusion that NPAS requires $n \times \ln(\alpha)/\ln(N/(N+1))$ iterations to obtain N points in $S(\alpha)$.

In practice there exists no realisation of PAS nor NPAS. We only have the approximation of NPAS by UCPR. What is the consequence of deviating from the ideal in terms of effectiveness and efficiency? Two observations were given in Section 1.

3.1. EFFECTIVENESS OF UCPR

R_k does not completely cover S_k due to the holes in the set. How much worse are we doing in terms of uniformity of the next sample point? What is the probability that a point in S_k is not covered by R_k? Let us define $S'_k = R_k \cap S_k$. Furthermore, the volume of a ball (berry) $B_i = \{x \,|\, \|x - p_i\| \leq cr_k\}$ around a point p_i in the current sample is $V(B_i) = q_n c^n r_k^n$, with the appropriate constant $q_n = \pi^{n/2}/\Gamma(1 + n/2)$ ($q_2 = \pi$; $q_3 = 4/3\pi$; $q_4 = 1/2\pi^2$, etc.). If we assume that B_i is completely in set S_k for all

i, then the probability that a uniformly randomly generated point in S_k is not in R_k (and in S'_k) is

$$\left(1 - \frac{q_n c^n r_k^n}{V(S_k)}\right)^N .$$

The quantity r_k is chosen to be the average nearest neighbour distance of the sample points, as it is known from literature on spatial statistics (see Cliff and Ord, 1981; Ripley, 1981) that $q_n r_k^n$ estimates the inverse of the density of points in a set S_k: $V(S_k)/N$. This idea implies that the ratio $V(S'_k)/V(S_k)$ can be approximated by

$$1 - \left(1 - \frac{c^n}{N}\right)^N .$$

By increasing c, this ratio can become arbitrarily close to unity. So by enlarging parameter c, the approximation of the failure rate can be pushed to zero. A larger value for c also causes a kind of repair effect, as R_k can walk in the direction of points of S_k which did not have any sample points in it. However, not only the assumption that B_i is a subset of S_k becomes less valid with increasing c, but there is also the drawback on the efficiency; more points are drawn and evaluated which are outside of S_k.

3.2. EFFICIENCY OF UCPR

We introduce the failure rate as $V(R_k \backslash S'_k)/V(R_k)$, the ratio of points that are generated outside the interior of S_k. Baritompa et al. (1995) taught us that if there is a fixed probability ρ at every iteration that a point is generated from a uniform distribution over the interior of S_k, the same linear complexity result as that of PAS holds. A necessary condition to be a ρ-adaptive algorithm, is that the failure rate is bounded above by a constant $1 - \rho$, which does not depend on the dimension.

We use problem $Q1$ again to get an idea of a bound of the part of R_k that sticks out of S_k, the failure rate. A part of a ball B_i around point p_i can only stick out of S_k if p_i is closer than cr_k to the boundary of S_k. On purpose we use the spherical problem $Q1$, as this is an extreme case in the sense that the surface/volume ratio is at a minimum. Therefore, the spherical problem gives a kind of lower bound on the failure rate, as for all other level sets the surface/volume ratio is at least as big. This means the probability of being close to the boundary is in general bigger. For problem $Q1$, the probability that p_i is closer than cr_k to the boundary of S_k is

$$1 - \left(1 - \frac{cr_k}{ymax_k}\right)^n .$$

So the fraction of points in the sample close to the surface of the boundary approaches unity exponentially in n. This means that for higher dimensions all volume or 'probability mass' tends to the boundary. This effect was noticed by many researchers in stochastic global optimization, who tried to get rid of boundary effects. Notice again that spherical level sets give a minimum of the surface/volume

ratio in the sense that a sphere has the highest volume for a given surface of the boundary. For level sets with another shape the fraction is even worse.

Is there a way to approximate ρ-adaptive better with UCPR? For $Q1$, spatial statistics tells us that the average interpoint distance $r_k \approx ymax_k \times N^{-1/n}$. In higher dimensions the distance tends to the maximum distance of 1, the world gets very empty, the nearest neighbour is desolately far away. This means a more than exponential increase of N with n is required to give the ρ-adaptive characteristic. This would again affect the convergence speed $N/(N+1)$. This analysis supports the hypothesis, that it is improbable that ρ-adaptive search will be realised in practice. The tendencies are illustrated in Section 4 for some test examples.

3.3. EFFECTIVENESS OF HIT AND RUN

Some last remarks are due to the Hit and Run process. The appealing property of the process is, that for $k \to \infty$, the iterate x_k resembles a random variate from a uniform distribution over X, as shown by Smith (1984). Zabinsky et al. (1993) derived further that the efficiency in the limit situation where the algorithm is in a local search stage converging for a spherical (or elliptical) function to the unique optimum, the expected number of function evaluations grows with $n^{2.5}$. The question is what will happen in practice, where we are not in the limit situation $k \to \infty$ and in cases when there are several global optimum points with respect to the effectiveness. Let us have a look at the individual iterations. One of the observations due to Romeijn et al. (1997) is that if x_k is in a corner of X, the next iteration tends to stay in the environment of x_k. Therefore, Romeijn et al. suggested a repair mechanism by a reflection operation.

Another observation which is usually made from Figure 4 is the decreasing sampling density when a point is further away from x_k. We are typically interested in what happens when the dimension increases. Notice first that in the one-dimensional case, every iteration is obviously already uniformly distributed over X. To make the sampling density analyzable in higher dimension, we refrain from boundary effects and introduce the following two sets:

$$\text{set 'close'} \qquad C: \{x \in X \,|\, \|x - x_k\| \leqslant \varepsilon\}$$

$$\text{set 'far away'} \qquad F: \{x \in X \,|\, \varepsilon \leqslant \|x - x_k\| \leqslant 2\varepsilon\}\,.$$

In all dimensions the probability to obtain the next iterate in inner sphere C is equal to the probability to get the next point in outer sphere F. When the ratio of the volumes of F and C is inspected

$$\frac{V(F)}{V(C)} = \frac{(2\varepsilon)^n - \varepsilon^n}{\varepsilon^n} = 2^n - 1\,,$$

it can be seen that points in F have a lower probability density than points in C. The discrepancy with the uniform distribution of the Hit and Run step increases in some sense exponentially in the dimension n. Analysis of discrete and continuous variants of the Hit and Run process makes it possible to study the speed of convergence of

the associated Markov chains towards a uniform distribution in higher dimensions. This defines an interesting problem. The limit theory is valid, but intuitively we require more and more (exponentially more) points to reach a uniform distribution in higher dimensions. We will see this effect in the numerical illustrations.

4. Numerical illustration

In this section the algorithm of UCPR is compared with implementations of other algorithms with respect to effectiveness and efficiency. The algorithm is effective if its result is a sample of a uniform distribution over $S(\alpha)$. Several numerical tests measuring departure from a uniform distribution are available from spatial statistics. We are using an easy to interpret measurement. The set $S(\alpha)$ is partitioned and the number of points in each compartment or partition set is compared to the expected number of points in the set. The number of compartments depends on the number of global minimum points of the test function. To measure the efficiency, the expected number of function evaluations to obtain the sample of N points is used. This is approximated by measuring the average number of function evaluations over several runs.

To illustrate the performance we use seven nonlinear test functions defined on a hyperrectangular domain. Furthermore the complexity of the UCPR algorithm is illustrated by a variant of the spherical problem in various dimensions and with varying values for the parameter c. First the test functions and the implementation of the algorithms are discussed.

4.1. TEST FUNCTIONS

To compare the performance, the 7 test functions given in Table 1 were used. Functions f_3 and f_4 are standard test functions which can be found in literature on Global Optimization among others in Törn and Žilinskas, 1989. The target of the algorithms is to generate a prefixed number $N = 100n$ points in the level set $S(\alpha)$. The level α is defined by $\alpha = Wy_*$. For the first test function $W = 1.15$ and for the others $W = 1.2$ was used.

4.2. THE ALGORITHMS

Pure Random Search (PRS) has a straightforward implementation for the test problems. We continue until N points have been found. To implement Pure Adaptive Search (NPAS), a way should be found to generate points in the interior of S_k. In this test, pure random search was done, but function evaluations to measure the efficiency were only counted for points generated in the current level set. In the implementation of Controlled Random Search (CRS) we used $m = 2$.

4.3. EFFECTIVENESS

The uniformity of the coverage of the $N = 100n$ points generated was tested for the

Table 1. Test functions used in the comparison between methods

	Domain	Definition	α
f_1	$[12, 22]^2$	least-squares fit $\Sigma (b-f)^2$ of $f = 1/((1+(a_1-x_1)^2(1+(a_2-x_2)^2)$ to data $\{(a_1, a_2, b)\} = \{(15, 15, 0.6), (15, 20, 0.7), (20, 15, 0.7), (20, 20, 0.8)\}$ (note: x_1 and x_2 parameters)	1.41
f_2	$[-1, 3] \times [-2, 3]$	least-squares fit $\Sigma (b-f)^2$ of $f = \exp(x_1)a/(\exp(x_2)+a)$ to data $\{(a, b)\} = \{(0.9, 0.4), (0.9, 0.55), (1, 0.4), (1, 0.6)\}$	0.037
f_3	$[-2, 2]^2$	(Golden-Price), $f_3 = 0.1 g_3 h_3$ with $g_3 = 1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)$ $h_3 = 30+(2x_1-3x_2)^2(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)$	0.36
f_4	$[-5, 10] \times [0, 15]$	(Branin) $f_4 = g_4 + h_4$, $g_4 = (x_2 - 5.1(x_1/2\pi))^2 + 5x_1/\pi - 6)^2$ and $h_4 = 10(1-1/8\pi)\cos(x_1)+10$	0.48
f_5	$[0, 4.34]^2$	$3 - \sin(x_1^2) - \sin(x_2^2)$	1.2
f_6	$[-2, 2]^n$	if $(x_1+x_2+\cdots+x_n) < 0$: $1+\Sigma(1-x_i)^2$ else $1+\Sigma(x_i+1)^2$	1.2
f_7	$[-1, 1]^n$	$1+\Sigma x_i^2$	

functions f_1, f_4, f_5 and f_6 that have multiple optima by comparing the expected number NR_{expected} (with respect to uniform covering) of points to arrive in a particular compartment of the level set to the actual number NR_{sample}. Functions f_1, f_4 and f_5 have 3, 3 and 9 global minimum points respectively and the corresponding level set has as many compartments. The size of the compartments and corresponding expected number of points can be determined numerically. Function f_7 is a so-called spherical problem (spherical level sets) with one global minimum point and although the effectiveness cannot be measured in the same way, we can observe the efficiency in several dimensions. Analogously, function f_6 can be called a bispherical problem, where the level sets have two equal sized compartments and therefore allows us to measure effectiveness in the same way and the effect of a varying dimension n. For the illustration $n = 2$, 4 and 8 are used. The results of measuring $(NR_{\text{sample}} - NR_{\text{expected}})/NR_{\text{expected}}$ (in an ideal uniform situation with a value of zero) are depicted graphically in Figure 5.

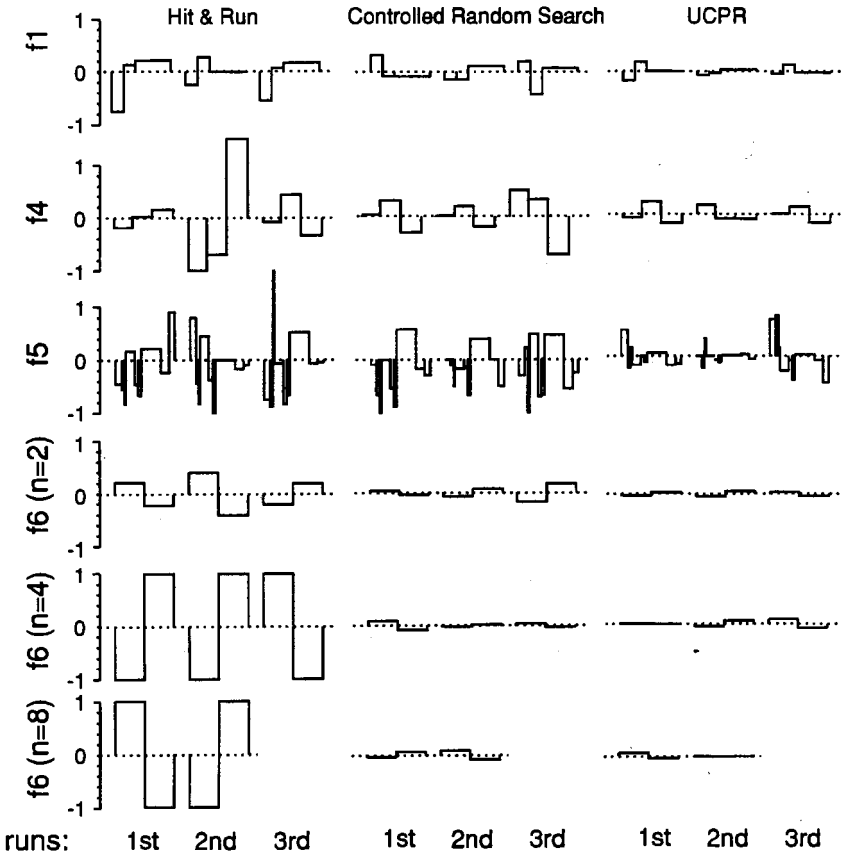

Figure 5. Measure of uniformity of the final samples (horizontal line would indicate complete uniformity), three test runs for every function and algorithm.

On the x-axis the three test runs are adjacent (one run of f_6 with $n = 8$ failed). The width of the bars is proportional to the expected number NR_{expected} of points in each compartment. The y-axis gives the relative deviation of the resulted number of points NR_{sample} in each compartment from the expected number of points: $(NR_{\text{sample}} - NR_{\text{expected}})/NR_{\text{expected}}$. A horizontal line would indicate perfect uniformity.

The results show that both H&R and CRS are significantly non-uniform: H&R in 15 out of 17 cases and CRS in 8 out of 17 cases. H&R places all points in one of the compartments for bispherical problem f_6 in higher dimensions. UCPR passes all tests with only one exception, one run for the function f_5 which has relatively many compartments. Perhaps surprisingly, CRS (which is known to be biased) is better than H&R, which should be uniform in limit. Apparently the rate at which H&R converges towards a uniform distribution is quite slow as discussed in Section 3.3.

4.4. EFFICIENCY

The efficiency has been measured as the number of function evaluations (for NPAS the records) necessary to obtain the set of N points.

For the spherical (f_6) and bispherical (f_7) test problems with variable dimension n, the efficiency of UCPR is no longer better than that of the others. In particular in the case of f_6 the performance of H&R is superior, but it has a serious drawback on uniformity as can be derived from Figure 5. All of the N points are concentrated in only one of the two compartments of $S(\alpha)$. This result illustrates that while the rate of convergence for H&R is affected only in minor extent with increasing dimension, its convergence to uniformity decreases rapidly. With increasing n the chance of accidentally arriving in the second compartment of the level set becomes vanishing small. This non-uniformity is also the explanation of the fact that H&R converges considerably faster than 'optimal' (NPAS) for $n = 4$ and $n = 8$.

4.5. COMPLEXITY

To reduce the advertisement character (this conflicts with Popperian science), which usually speaks from numerical results such as that of Table 2, let us focus on the analysis of failure rate and uniformity with the aid of a simple (extreme case) numerical experiment. The analysis of Section 3 is illustrated here by applying the UCPR algorithm for generating $N = 50$ points in $S(1.05)$ for spherical problem f_7, with varying dimension and value for the parameter c. It has been argued that parameter c gives a trade-off between effectiveness (uniformity of the final sample) and efficiency (number of function evaluations). There are many ways to measure the tendency of the sample to originate from a uniform distribution (see Ripley, 1981). We illustrate the idea of a Chi-square statistic here. The parameter space can be partitioned arbitrarily. In the former, we applied the compartments of the level set

Table 2. Numerical results being average number of function evaluations over three runs

Function	n	N	PRS	NPAS	H&R	CRS	UCPR
f_1	2	200	5688	870	1207	1115	943
f_2	2	200	9453	833	1970	1526	939
f_3	2	200	$1.05\ 10^6$	1775	$1.18\ 10^4$	5617	2221
f_4	2	200	$1.32\ 10^5$	1496	7452	2886	2132
f_5	2	200	6086	881	2210	2035	1217
f_6	2	200	2546	450	1084	786	482
	4	400	$2.59\ 10^5$	2990	2853	6909	5879
	8	800	$4.03\ 10^9$	$1.31\ 10^4$	8189	$5.32\ 10^4$	$6.33\ 10^4$
f_7	2	200	941	511	421	425	533
	4	400	3069	1217	911	750	1347
	8	800	7793	2623	1785	1877	3122

as partition sets. For f_7 (only one global minimum point) the parameter space is divided into 4 parts of equal size determined by the sign of the parameters x_1 and x_2. Measuring and adding $(NR_{\text{sample}} - NR_{\text{expected}})^2 / NR_{\text{expected}}$ for each partition set defines a statistic which (approximately) has a Chi-square distribution. Here it means that under the hypothesis of uniformity, the statistic has an expected value of 3 and the hypothesis is accepted when a value less than 7.8 is found (95% confidence limit, chi-square distribution with 3 degrees of freedom).

An important factor in the analysis of Section 3 is the so-called failure rate; the number of function evaluations that do not give an improvement of the level set divided by the total number of function evaluations. We measured those criteria for one run of the UCPR algorithm with values for the parameter c of 2, 1.5 and 1.2 for the problem f_7 with dimension n of 2, 3 and 4. We just consider one run for each case, to be able to have a look at the final sample.

It is clear that for lower values of c the failure rate decreases, which causes the total number of evaluations that have to be done going down. The drawback is that the uniformity of the final set of points grows worse, although this is hard to conclude from the table given that only one sample has been generated. On the other hand, an extreme example could be observed for the case with $c = 1.2$ and $n = 3$: In a certain iteration all points generated have negative values for x_2. The parameter c

Table 3. Results of UCPR problem for f_7, varying dimension n and values of c

		Number of function evaluations			Failure rate			Uniformity (Chi-square stat.)		
c	n	2	3	4	2	3	4	2	3	4
2		437	1158	2809	18%	55%	77%	0.88	110.96	3.28
1.5		344	709	1437	8%	28%	55%	10.48	16.24	0.88
1.2		292	734	1155	4%	16%	25%	12.56	77.04	36.08

is too small to give that new points are generated at the other side of the axis.

5. Conclusions

The problem of generating a sample of N points over a level set $S(\alpha)$ was studied. The number of function evaluations necessary to generate N points from a uniform distribution over a level set $S(\alpha)$, is linear in the dimension for the theoretical ideal algorithm of Pure Adaptive Search (PAS). Like every good ideal, it probably is impossible to reach. This impossibility can be derived from considerations on complexity of global and integer programming. The same theoretical complexity may be reached by a modification of PAS which uses a sample of points called N-points PAS. The algorithm of Uniform Covering by Probabilistic Rejection (UCPR) is a heuristic practical approximation of N-points PAS. For higher dimensions the deviation from the ideal becomes bigger. The test results show that UCPR performs in general better than other practical alternatives such as Pure Random Search, Controlled Random Search and a variant of the Hit and Run algorithm. The development of better algorithms, i.e. closer to the ideal, is still a challenge to Popperian science.

Acknowledgements

Special thanks are due to Paul van Beek for reading and commenting on earlier versions of the manuscript, to the anonymous referees for their valuable remarks and to Fabio Schoen for reading and helping to stop the leaks in example $Q2$.

References

Baritompa, W.P. and Steel, M.A. (1996), Bounds on absorption times of directionally biased random sequences, *Random Structures and Algorithms* 9: 279–293.

Baritompa, W.P., Mladineo, R.H., Wood, G.R., Zabinsky, Z.B. and Zhang Baoping (1995), Towards pure adaptive search, *Journal of Global Optimization* 7: 73–110.

Bates, D.M. and Watts, D.G. (1988), *Nonlinear regression analysis and its applications*, Wiley, New York.

Bulger, D.W. and Wood, G.R. (1998), Hesitant adaptive search for global optimisation, *Mathematical Programming* 81: 89–102.

Cliff, A.D. and Ord, J.K. (1981), *Spatial processes, models and applications*, Pion Ltd., London.

Donaldson, J.R. and Schnabel, R.B. (1987), Computational experience with confidence regions and confidence intervals for nonlinear least squares, *Technometrics* 29: 67–82.

Garcia, I., Ortigosa, P.M., Casado, L.G., Herman, G.T. and Matej, S. (1997), Multidimensional Optimization in Image Reconstruction from Projections, in I.M. Bomze, T. Csendes, R. Horst and P.M. Pardalos (eds.), *Developments in Global Optimization*, Kluwer, pp. 289–300.

Hendrix, E.M.T. (1998), *Global Optimization at Work*, Ph.D. dissertation, Wageningen Agricultural University.

Klepper, O. and Rouse, D.I. (1991), A procedure to reduce parameter uncertainty for complex models by comparison with real system output illustrated on a potato growth model, *Agricultural Systems* 36: 375–395.

Klepper, O. and Hendrix, E.M.T. (1994), A method for robust calibration of ecological models under different types of uncertainty, *Ecological Modelling* 74: 161–182.

Klepper, O. and Bedaux, J.M. (1997), Nonlinear parameter estimation for toxicological threshold models, *Ecological Modelling* 102: 315–324.

Patel, N.R., Smith, R. and Zabinsky, Z.B. (1988), Pure adaptive search in Monte Carlo optimization, *Mathematical Programming* 43: 317–328.

Price, W.L. (1979), A controlled random search procedure for global optimization, *The Computer Journal* 20: 367–370.

Ripley, B.D. (1981), *Spatial Statistics*, Wiley, New York.

Romeijn, H.E. (1992), *Global Optimization by Random Walk Sampling Methods*, Ph.D. dissertation, Erasmus University, Rotterdam.

Romeijn, H.E. and Smith, R.L. (1994), Simulated annealing for constrained global optimization, *Journal of Global Optimization* 5: 101–126.

Romeijn, H.E., Zabinsky, Z.B., Graesser, D.L. and Neogi, S. (1999), A new reflexion generator for simulated annealing for mixed-interger/continuous global optimization, *Journal of Optimization: Theory and Applications* 101: 403–427.

Ross, G.J.S. (1990), *Nonlinear Estimation*, Springer, New York.

Smith, R.L. (1984), Efficient Monte Carlo procedures for generating points uniformly distributed over bounded regions, *Operations Research* 32: 1296–1308.

Törn, A. and Žilinskas, A. (1989), *Global Optimization*, Springer, Berlin.

Walter, E. and Piet-Lahanier, H. (1990), Estimation of parameter bounds from bounded-error data: a survey, *Mathematics and Computers in Simulation* 32: 449–468.

Zabinsky, Z.B. and Smith, R.L. (1992), Pure adaptive search in global optimization, *Mathematical Programming* 53: 323–338.

Zabinsky, Z.B., Smith, R.L., McDonald, J.F., Romeijn, H.E. and Kaufman, D.E. (1993), Improving Hit-and-Run for global optimization, *Journal of Global Optimization* 3: 171–192.