
Computational Optimization and Applications, 30, 297–318, 2005

Application of Deterministic Low-Discrepancy Sequences in Global Optimization

SERGEI KUCHERENKO s.kucherenko@imperial.ac.uk
Imperial College London, SW7 2AZ, UK

YURY SYTSKO
Moscow Engineering Physics Institute, 115409 Moscow, Russia

Received August 15, 2003; Revised February 27, 2004; Accepted April 2, 2004

Abstract. It has been recognized through theory and practice that uniformly distributed deterministic sequences provide more accurate results than purely random sequences. A quasi Monte Carlo (QMC) variant of a multi level single linkage (MLSL) algorithm for global optimization is compared with an original stochastic MLSL algorithm for a number of test problems of various complexities. An emphasis is made on high dimensional problems. Two different low-discrepancy sequences (LDS) are used and their efficiency is analysed. It is shown that application of LDS can significantly increase the efficiency of MLSL. The dependence of the sample size required for locating global minima on the number of variables is examined. It is found that higher confidence in the obtained solution and possibly a reduction in the computational time can be achieved by the increase of the total sample size N. N should also be increased as the dimensionality of problems grows. For high dimensional problems clustering methods become inefficient. For such problems a multistart method can be more computationally expedient.

Keywords: global optimization, stochastic optimization, low-discrepancy sequences, multi level single linkage method

1. Introduction

The motivation for this paper is to develop further efficient and robust optimization methods. Let $f(\boldsymbol{x})$: $\mathbb{R}^n \rightarrow \mathbb{R}$ be a continuous real valued objective function. A nonlinear global optimization problem is defined as follows:

$$\min f(x), \quad \boldsymbol{x} \in \mathbb{R}^n \tag{1}$$

subject to

$$\vec{g}(\boldsymbol{x}) = 0, \quad \vec{g}(\boldsymbol{x}) = \{g_i\}, \quad i = 1,\ m_e, \tag{2}$$
$$\vec{h}(\boldsymbol{x}) \geq 0, \quad \vec{h}(\boldsymbol{x}) = \{h_i\}, \quad i = m_e + 1,\ m, \tag{3}$$

where $\boldsymbol{x}$ is a vector of bounded continuous variables. No restrictions are imposed on the functional form of the objective function, $f(\boldsymbol{x})$ or the constraints $\vec{g}(\boldsymbol{x})$ and $\vec{h}(\boldsymbol{x})$.

There are two kinds of commonly used techniques for solving (1)–(3): deterministic and stochastic. Deterministic methods guarantee convergence to a global solution within a specified tolerance (a tolerance is defined as the maximum difference between the objective function value of the numerical solution and the true global optimal solution). For most deterministic methods the complexity of the problem grows exponentially as a function of the number of variables. For high dimensional problems the computational time is usually prohibitively large. Although some efficient methods have been designed for various forms of an objective function and/or the constraints, these methods are tailored to very specific problem structures and cannot be applied in the general high dimensional case. Good surveys of advances in global optimization are given in [5, 7, 9].

The present study is confined to stochastic methods and their variants based on deterministic sampling of points. A stochastic approach for global optimization in its simplest form consists only of a random search and it is called Pure Random Search (PRS). In PRS, an objective function $f(\boldsymbol{x})$ is evaluated at N randomly chosen points and the smallest value of $f(\boldsymbol{x})$ is taken as an approximation to the global minimum.

For stochastic methods the following result holds: if N points are drawn from a uniform random distribution over the n-dimensional hypercube H^n: $H^n = \{x_i \mid 0 \leq x_i \leq 1,\ i = 1, n\}$ and if $f(\boldsymbol{x})$ is a continuous function defined in the feasible domain $B = H^n$, then the sample point with lowest function value converges to the global minimum. Stochastic search methods yield an asymptotic (in a limit $N \to \infty$) guarantee of convergence. This convergence is with probability 1 (or almost surely).

The PRS approach is not very efficient because the expected number of iterations for reaching a specified tolerance grows exponentially in the dimension n of the problem. Advanced stochastic techniques use stochastic methods to search for the location of local minima and then utilize deterministic methods to solve a local minimization problem. Two phases are considered: global and local. In the global phase, the function is evaluated in a number of randomly sampled points from a uniform distribution over H^n. In the local phase the sample points are used as starting points for a local minimization search. Thus the information obtained on the global phase is refined. For continuous differentiable objective functions classical gradient-based methods are used for local minimization. For non-differentiable functions or functions whose derivatives are difficult to evaluate the local search can be obtained through further sampling in a small vicinity around a starting point. The efficiency of the multistage methods depends both on the performance of the global stochastic and the local minimization phases.

In the simplest form of the multistage approach a local search is applied to every sample point. Inevitably, some local minima would be found many times. The local search is the most computationally intensive stage and ideally it should start just once in every region of attraction. The region of attraction of a local minimum x^* is defined as the set of points starting from which a given local search procedure converges to x^*. This is the motivation behind various versions of clustering methods. An extensive review on this subject can be found in [15, 16, 27].

The objective of the global stage is to obtain as much information as possible about the underlying problem with a minimum number of sampled points. To achieve this objective, sampled points should satisfy certain criteria. First, they should be distributed as evenly

as possible. Second, on successive iterations new sampled points should fill the gaps left previously. If new points are added randomly, they do not necessarily fill the gaps between the points sampled on previous iterations. As a result, there are always empty areas and regions where the sampled points are wasted due to clustering. No information can be obtained on the behavior of the underlying problem in empty areas.

It has been recognized through theory and practice that a variety of uniformly distributed deterministic sequences provide more accurate results than purely random samples of points. Low-discrepancy sequences (LDS) are designed specifically to place sample points as uniformly as possible. Unlike random numbers, successive low discrepancy points "know" about the position of their predecessors and fill the gaps left previously.

LDS have been used instead of random numbers in evaluating multi-dimensional integrals and simulation of stochastic processes—in the areas where traditionally Monte Carlo (MC) methods were used [13, 23]. It has been found that methods based on LDS, known as quasi Monte Carlo (QMC) methods, always have performance superior to that of MC methods. Improvement in time-to-accuracy using QMC can be as large as several orders of magnitude.

LDS are a natural substitute for random numbers in stochastic optimization methods. As in other areas of applied mathematics, QMC methods provide higher accuracy with fewer evaluations of the objective function. The improvement in accuracy depends on the number of dimensions, the discrepancy of the sequence both of which are known, and the variation of the function, which is generally not known.

Central to the QMC approach is the choice of LDS. Different principles were used for constructing LDS by Holton, Faure, Sobol', Niederreiter and others (good surveys of LDS are given in [3, 12, 13]). Niederreiter's LDS have the best theoretical asymptotic properties. However, many practical studies have proven that Sobol' LDS in many aspects are superior to other LDS [14, 24]. For this reason they were used in the present study. In a classification developed by Niederreiter, the Sobol' LDS are known as (t, s) sequences in base 2 [13, 24]. The Holton LDS [8] were also used for comparison.

There had been a lack of a representative set of test problems for comparing global optimization methods. To remedy this a classification of essentially unconstrained global optimization problems into unimodal, easy, moderately difficult and difficult problems was proposed in [28]. The problem features giving this classification are the chance to miss the region of attraction of the global minimum, embeddedness of the global minimum, and the number of minimizers.

The purpose of this paper is the further development of optimization methods with an emphasis on comprehensive testing and a comparison of various techniques on a set of test problems of various complexity in accordance with the classification developed in [28]. In particular: a comparison was made between:

- QMC and stochastic variants of a well known multi level single linkage (MLSL) algorithm [15, 16];
- different implementations of MLSL;
- two different types of LDS;
- MLSL and a variant of a simple linkage (SL) method developed in [17].

A number of problems used for testing belong to the category of the difficult multidimensional problems.

The remainder of this paper is organized as follows. A brief analysis of a Quasirandom Search (QRS) method is given in Section 2. Descriptions of MLSL and SL methods are presented in Section 3. Results of a comparison between stochastic MLSL, LDS based MLSL and SL methods are presented in Section 4. Finally, the performance of different techniques is discussed in Section 5.

2. Analysis of quasirandom search method

A general scheme of a QRS method is similar to that of PRS: an objective function $f(\boldsymbol{x})$ is evaluated at N LDS points and then the smallest value of $f(\boldsymbol{x})$ is taken as the global minimum. Generally QRS lacks the efficiency of more advanced methods. However, in some cases QRS has the following advantages over other methods of global optimization:

1. In its most general form it does not use any assumptions about the problem structure. In particular it can be used for any class of objective function (i.e. non-differentiable functions).
2. It can explicitly account for inequality constraints. The feasible region can be non-convex and even disconnected. However, it is not possible to account explicitly for equality constraints and such an optimization problem should be transformed into an unconstrained one.
3. It belongs to the so-called nonadaptive algorithms [29], in which the numerical process depends only on the current state and not on previously calculated states. In contrast, in adaptive algorithms information is obtained sequentially. Nonadaptive algorithms are superior to adaptive ones in multi-processor parallel computations.

These advantages become more apparent as the number of variables grows. Analysis of QRS is important for understanding the advantages that the use of LDS brings to the multistage approach.

In this section it is assumed for simplicity that the problem is unconstrained and the feasible region is a closed set K, $K \subset \mathbb{R}^n$,

$$K = \left\{ x_i \,\middle|\, x_i^L \le x_i \le x_i^U,\ i = 1, \ldots, n \right\}.$$

By linear transformation of coordinates K can be mapped into the n-dimensional hypercube H^n, so that the problem is formulated as:

$$\min f(\boldsymbol{x}), \quad \boldsymbol{x} \in H^n. \tag{4}$$

Let f^* be an optimal value. Consider a sequence of points $\boldsymbol{x}_{(N)} = \{\boldsymbol{x}_j \,|\, \boldsymbol{x}_j \in H^n, j = 1, \ldots, N\}$ and an approximation f_N^* to f^*:

$$f_N^* = \min_{x_j \in x_{(N)}} f(x_j).$$

On a class of continuous functions $f(\boldsymbol{x})$ and dense sequences in H^n the following result holds:

$$f^* = \min_{N\to\infty} f_N^*.$$

For the purposes of error analysis the function $f(\boldsymbol{x})$ is assumed to have piecewise continuous partial derivatives satisfying the conditions:

$$|\partial f/\partial x_i| \le C_i, \quad i = 1, \ldots, n. \tag{5}$$

From (5) it follows that $f(\boldsymbol{x})$ satisfies a Lipschitz condition:

$$|f(\boldsymbol{x}) - f(\boldsymbol{y})| \le L\rho(\boldsymbol{x}, \boldsymbol{y}), \tag{6}$$

where L is a Lipschitz constant. The dispersion $d_N(n)$ of the sequence $\boldsymbol{x}_{(N)}$ is defined as [13]:

$$d_N(n) = \sup_{x\in H^n} \min_{1\le j\le N} \rho(\boldsymbol{x}, \boldsymbol{x}_j), \tag{7}$$

where $\rho(\boldsymbol{x}, \boldsymbol{y})$ is the Euclidian distance (metric) between points $\boldsymbol{x}$ and $\boldsymbol{y}$. Using (6) and (7) the approximation error can be written as

$$f_N^* - f^* \le Ld_N(n). \tag{8}$$

As can be seen from (8), $d_N(n)$ defines the "quality" of the sequence. Sequences with small $d_N(n)$ guarantee a small error in a function approximation. For any sequence the following error bounds hold:

$$[1/(N\omega_n)]^{1/n} \le d_N(n) \le 2\sqrt{n}(D(n, N)/N)^{1/n}, \tag{9}$$

where $\omega_n = \pi^{n/2}/\Gamma(1 + \frac{n}{2})$ is the volume of the n-dimensional unit ball and $D(n, N)$ is the discrepancy of a sequence [13]. Discrepancy is a measure of deviation from uniformity. Apparently, smaller $D(n, N)$ would provide smaller upper estimate of the dispersion $d_N(n)$. LDS are characterized by small $D(n, N)$, therefore every LDS is a low-dispersion sequence (but not conversely).

The best-constructed LDS have $D(n, N) = O(\ln^{n-1} N)$. For such LDS the resulting rate of convergence of QRS as follows from (9) is $O(N^{-1/n} \ln^{(n-1)/n} N)$. This rate is not sufficiently high when n is large. However, it is worth noting that an error bound (8) with $d_N(n)$ given by (9) was obtained in the assumption that function $f(\boldsymbol{x})$ depends equally on all variables: in other words, the constants $C_i, i = 1, \ldots, n$ in (5) were assumed to be of the same order of magnitude. This was shown to be "the worst-case scenario" [19, 20]. In practical applications, the function $f(\boldsymbol{x})$ normally strongly depends on a subset of variables:

$x_{i_1}, x_{i_2}, \ldots, x_{i_s},\ 1 \le i_1 < i_2 < \cdots < i_s, s < n$ and dependence on other variables can be weak. In this case inequality (6) becomes

$$|f(\boldsymbol{x}) - f(\boldsymbol{y})| \le L\rho(\boldsymbol{x}', \boldsymbol{y}'),$$

where $\boldsymbol{x}', \boldsymbol{y}'$ is a projection of the points $\boldsymbol{x}, \boldsymbol{y}$ on the s-dimensional face $H_{i_1,i_2,\ldots i_s}$ of H^n. One very useful property of LDS is that the projection of n-dimensional LDS on s-dimensional subspace forms s-dimensional LDS. Then (9) becomes

$$[1/(N\omega_s)]^{1/s} \le d_N(s) \le 2\sqrt{s}(D(s, N)/N)^{1/s} \tag{10}$$

and for practical applications n should be substituted by "an effective dimension number" s, which can be much less than n [20]. It can result in a much higher rate of convergence than that predicted by (9). This correction is very important for understanding the advantages of using LDS in QRS. For comparison, a cubic grid provides a better discrepancy measure than (9). At first glance such a grid search may be seen as more efficient than QRS. However, a projection of an n-dimensional cubic grid LDS on s-dimensional subspace does not form an s-dimensional cubic grid because of "the shadow effect" (projections of some points on the coordinate axis would coincide). This means that the correction similar to (10) is not applicable for the cubic grid and its discrepancy measure does not improve as s gets smaller.

Many well-known LDS were constructed mainly upon asymptotic considerations, as a result they do not perform well in real practical tests. The Sobol' LDS were constructed by following three main requirements [18]:

1. Best uniformity of distribution as N goes to infinity.
2. Good distribution for fairly small initial sets.
3. A very fast computational algorithm.

Points generated by the Sobol' LDS produce a very uniform filling of the space even for a rather small number of points N, which is a very important case in practice.

In some cases, it is convenient to employ the dispersion in the maximum (or infinite) metric $\rho'(\boldsymbol{x}, \boldsymbol{y})$ defined by

$$\rho'(\boldsymbol{x}, \boldsymbol{y}) = \max_{1 \le i \le n} |x_i - y_i|,$$

where $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ and $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)$. The dispersion $d'_N(n)$ of the sequence $\boldsymbol{x}_{(N)}$ in the maximum metric

$$d'_N(n) = \sup_{\boldsymbol{x} \in H^n} \min_{1 \le j \le N} \rho'(\boldsymbol{x}, \boldsymbol{x}_j)$$

has the following error bounds:

$$\frac{1}{2N^{1/n}} \le d'_N(n) \le \frac{\alpha(n, N)}{N^{1/n}}, \tag{11}$$

where the parameter $\alpha(n, N)$ generally is a weak function of N. For (t, s) sequences this parameter does not depend on N and an improved error bounds has the form

$$\frac{1}{2N^{1/n}} \leq d'_N(n) \leq \frac{b^{(n+t)/n}}{N^{1/n}}. \tag{12}$$

In particular for the Sobol' LDS (12) becomes

$$\frac{1}{2N^{1/n}} \leq d'_N(n) \leq \frac{2^{1+T_2(n)/n}}{N^{1/n}}, \tag{13}$$

where $T_2(n)$ is a function with an upper bound

$$T_2(n) < n(\log_2 n + \log_2 \log_2 n + 1). \tag{14}$$

These results were used in the frameworks of quasi random linkage methods presented in [10, 17].

QRS was applied to solve global optimization problems in [1] and [26] as early as 1970 (see also [21, 22]). However, as it was stated above, with the development of more advanced multistage methods the application of pure QRS is limited mainly to cases of non-differentiable objective functions and to problems in which high accuracy in finding a global solution is not required. In the framework of MLSL, QRS can be seen as a global phase of MLSL. A description of MLSL is given in the next section.

3. Single linkage and multilevel single linkage methods

In the simplest variant of a multistage methods, a small number of random points are sampled and then a deterministic local search procedure (LS) is applied to all of these points. All located stationary points are sorted and the one with the lowest value of the objective function is taken as a global minimum. The general scheme of a Multistart (MS) algorithm is as follows:

Step 1. Sample a point from a uniform distribution over H^n.
Step 2. Apply LS to the new sample point.
Step 3. If a termination criterion is not met, then return to Step 1.

One problem with the Multistart technique is that the same local minimum may be located several times. Ideally, LS should be started only once in every region of attraction. A few algorithms had been developed with such a property. Only those sample points whose function values are small enough are chosen as starting points. Points are grouped into clusters, which are initiated by a seed point. The seed point is normally a previously found local minimum $\boldsymbol{x}^*$, $\boldsymbol{x}^* \in \boldsymbol{X}^*$, where $\boldsymbol{X}^*$ is a set of all local minimum points. All sample points within a critical distance are assigned to the same cluster.

Efficiency can be improved by reducing the number of local searches, namely by discarding some of the sampled points. If $\{f_i\}$ is an ordered set: $\{f_i\} = \{f(\boldsymbol{x}_i) \mid f(\boldsymbol{x}_i) < f(\boldsymbol{x}_{i+1}), i = 1, \ldots, N\}$ and $\boldsymbol{X}$ is a corresponding ordered set of all sampled points, then the reduced sample set is taken as:

$$\boldsymbol{X}_r = \{\boldsymbol{x}_i \in \boldsymbol{X} \mid i = 1, \ldots, N_r, N_r = \alpha N\}, \tag{15}$$

where $0 < \alpha < 1$. In this case, some local minima can be discarded without affecting the global minimum search.

An important question in applying any numerical method is when to stop searching for the global minimum. Among various proposed termination criteria of the global stage, one of the most reliable was developed in [2]. It is based on Bayesian estimates for the number of real minima not yet identified and the probability that the next local search will locate a new local minimum. An optimal Bayesian stopping rule is defined as follows: if W different local minima have been found after N local searches started in uniformly distributed points, then the expectation of the number of local minima is

$$W_{\text{exp}} = W(N-1)/(N-W-2), \tag{16}$$

provided that $N > W + 2$. The searching procedure is terminated if

$$W_{\text{exp}} < W + 0.5. \tag{17}$$

The MLSL method developed by Rinnooy Kan and Timmer [15, 16] is one of the best algorithms among various clustering methods. The general scheme of the MLSL algorithm is outlined below:

Step 1. Set $W := 0, k := 0$.
Step 2. Set $k := k + 1, i := 0$.
Step 3. Sample a set $\boldsymbol{X}$ of N points from a uniform distribution over H^n.
Step 4. Evaluate an objective function on set $\boldsymbol{X}$, sort $\{f_i\}$ in order of increasing function values and select a reduced set $\boldsymbol{X}_r$ according to (15).
Step 5. Set $i := i + 1$ and take $\boldsymbol{x}_i \in \boldsymbol{X}_r$.
Step 6. Assign the sample point $\boldsymbol{x}_i$ to some cluster $\boldsymbol{C}_l$ if $\exists\, \boldsymbol{x}_j, x_j \in \boldsymbol{C}_l$ such that $\rho(\boldsymbol{x}_i, \boldsymbol{x}_j) \leq r_k$, $f(\boldsymbol{x}_j) \leq f(\boldsymbol{x}_i)$, where r_k is a critical distance given by (18). If $\boldsymbol{x}_i$ is not assigned to any cluster yet then start a local search at $\boldsymbol{x}_i$ to yield a local minimum $\boldsymbol{x}^*$. If $\boldsymbol{x}^* \notin \boldsymbol{X}^*$, then add $\boldsymbol{x}^*$ to $\boldsymbol{X}^*$, set $W := W + 1$ and initiate the W-th cluster by $\boldsymbol{x}^*$. Assign $\boldsymbol{x}_i$ to the cluster that is initiated by $\boldsymbol{x}^*$.
Step 7. If $i := N_r$ go to Step 8. Else go to Step 5.
Step 8. If $k := Iter_{\max}$, where $Iter_{\max}$ is the maximum allowed number of iterations, or the stopping rule (15), (16) is satisfied, then stop. Else go to Step 2.

The critical distance r_k is found using cluster analysis on a uniformly distributed sample:

$$r_k = \left[\frac{m(B)}{\omega_n} \frac{\sigma \log(kN_r)}{kN_r} \right]^{1/n}. \tag{18}$$

Here $m(B)$ is the Lebesgue measure (if $B = H^n$ then $m(B) = 1$), k is an iteration index, σ is a known parameter. In our calculations the parameter σ was taken to be 2.0.

Sporadic clustering which is characteristic of relatively small sets of random points would result in inhibiting many LS because such clustered points could be assigned to the same clusters initiated by local minima. A comparison between (10) and (18) shows that the dispersion of LDS and critical distance r_k have a similar asymptotic behavior. It suggests that LDS are better suited for optimization problems than random sets of points. As in other cases of transition from MC to QMC algorithms, a significant improvement in efficiency can be achieved simply by substituting random points with LDS.

Schoen argued that the regularity of LDS can be further exploited [17]. He suggested using instead of (18) a critical distance

$$r_{N,\beta} = \beta N^{-1/n}, \tag{19}$$

where β is a known parameter. It was proved that within the framework of a Simple Linkage (SL) method that if the sampled points are generated according to LDS whose dispersion is limited by (11) then the total number of LS started even if the algorithm is never stopped will remain finite, provided that $\beta > \alpha(n, N)$. A SL method was developed in Locatelli and Schoen (1996) in order to circumvent some deficiencies of MLSL. A LDS based SL method was presented in [17]. The scheme of the SL method adopted for LDS sampling is the following:

Step 1. Set $N := 0$; choose $\varepsilon > 0$;
Step 2. Let $N := N + 1$;
Step 3. Generate a point $\boldsymbol{x}$ from LDS in H^n;
Step 4. Apply a local search algorithm from $\boldsymbol{x}$ except if $\exists\ \boldsymbol{x}_j$ in the sample such that: $\rho'(\boldsymbol{x}, \boldsymbol{x}_j) \leq r_{N,\beta}$ and $f(\boldsymbol{x}_j) \leq f(\boldsymbol{x}) + \varepsilon$;
Step 5. If stopping criteria is satisfied, then stop. If not, add $\boldsymbol{x}$ to the sample and go to Step 2.

It is important to note that the SL method makes use of the maximum (or infinite) metric $\rho'(\boldsymbol{x}, \boldsymbol{y})$ instead of the Euclidean one which is used in MLSL.

In the computational examples in Section 4, the performance of QMC and stochastic variants of MLSL are compared with the SL algorithms.

4. Computational experiments

As stated in [28] the choice of test problems should be systematic, so that they represent different types of problems ranging from easy to difficult to solve. Following this strategy, a C++ program called SobolOpt which employs all discussed algorithms, namely

SL, stochastic MLSL and its QMC variants with Sobol' and Holton LDS points was applied to a number of test problems of different complexity. All problems presented below are unconstrained, although the techniques used are readily applicable to constrained problems.

A local search was performed using standard nonlinear programming routines from the NAG library [11]. All computational experiments were carried out on an Athlon-800 MHz PC.

In most cases the objective was to find all the local minima that were potentially global. Four criteria for comparing the algorithms were used: (i) success in locating a global minimum; (ii) number of located local minima; (iii) number of calls of a local minimizer; (iv) average CPU time (in seconds).

The results are displayed in the tables. The following notation is used:

"N"—total number of sampled points in each iteration. For the Sobol' LDS the equidistribution property and improved discrepancy estimates hold for N equal to a power of 2. In all experiments N was taken to be equal to 2^m, where m is an integer number;
"N_r"—reduced number of sampled points on each iteration;
"$N_{\min}$"—total number of located minima;
"*Iter*"—total number of iterations on the global stage;
"$Iter_{\max}$"—maximum number of iterations on the global stage;
"LM"—number of calls of the local minimizer;
"GM"—"y" ("n")—global minimum (GM) was found (not found) in a particular run, "Y"—global minimum was found in all four runs, "N"—was not found in any of four runs;
"LDS Sobol"'—the MLSL method based upon Sobol' LDS sampling;
"LDS Holton"—the MLSL method based upon Holton LDS sampling;
"Random"—the MLSL method based upon random sampling;
"Zakovic"—based upon random sampling implementation of the MLSL algorithm developed in [30, 31].
"SL Schoen"—Single Linkage Schoen's algorithm based upon Sobol' LDS sampling with Bayesian stopping rule.
"LDS Sobol' (NS)", "LDS Holton (NS)", "Random (NS)", "SL Schoen (NS)"—versions of the above mentioned algorithm in which the Bayesian stopping rule is not used, however the maximum number of iterations is limited above by $Iter_{\max}$.
" .... (NC)"—a version of " .... (NS)" algorithm in which clustering is not used (MS method).
"SL Schoen (mod)"—modified "SL Schoen (NS)" with a different strategy of sampling (details are given in Section 4.1).

Four independent runs for each test problem were performed. For the Random MLSL method all runs were statistically independent. For the LDS Sobol' (Holton) method for each run a different part of the Sobol' (Holton) LDS was used.

SL Schoen's algorithm and its variants proved to be less efficient than MLSL methods. The results for this algorithm are presented only for test Problems 1, 2A and 2B.

4.1. Problem 1: Six-hump camel back function

$$f(x, y) = 4x^2 - 2.1x^4 + 1/3x^6 + xy - 4y^2 + 4y^4,$$
$$-3.0 \le x \le 3.0, \quad -2.0 \le y \le 2.0.$$

Global Solution:

$$f(x, y) = -1.03163,$$
$$(x, y) = (0.08984, -0.712266),$$
$$(x, y) = (-0.08984, 0.712266).$$

This is a well known test for global optimization [6]. There are 6 known solutions, two of which are global. Results for this test are presented in Tables 1.1–1.2. According to the classification of problems into the degrees of difficulty suggested in [28] this problem belongs to a class of "easy" (E1) problems.

In all four runs of the LDS Sobol' and Holton algorithms all six local minima were located with just six LM. For the Random MLSL method in one of the four runs only four local minima were found, five—in two runs and six—in one run. For this method, in almost all runs LM was larger than a number of located minima. To compare our results with those of other authors we used a program developed by Zakovic [30, 31]. This program was an implementation of the MLSL algorithm, similar to that of Dixon and Jha [4]. In all four runs Zakovic's program located all six minima but at the expense of 21 iterations and 96 calls of the local minimizer. Similar results with LM equal 92 were reported in [4]. It shows that the above mentioned implementations of the MLSL algorithm by other authors are not very efficient. The differences were mainly due to the ways in which clustering and sorting algorithms were implemented. It was not possible to make a straightforward CPU time comparison as Zakovic's program is written in Fortran and makes use of a different local minimizer routine. However, other factors being equal one can expect the CPU time to be proportional to LM.

Other experiments were performed with smaller samples of points ($N/N_r = 64/32$, $N/N_r = 128/64$). In these not all local minima were located and in some cases only one global minimum was found. We can conclude that the set of parameters $N/N_r = 256/128$ were the most efficient settings.

Table 1.1. Comparison of various realizations of MLSL for Problem 1.

Algorithm	N/N_r	*Iter*	LM	$N_{\min}$	GM	CPU
LDS Sobol'	256/128	1	6	6	Y	0.1
LDS Holton	256/128	1	6	6	Y	0.12
Random	256/128	1	6	5	Y	0.1
Zakovic	256/128	21	96	6	Y	not available

Table 1.2. Comparison of various realizations of SL for Problem 1.

Algorithm	N/N_r	*Iter*	LM	$N_{\min}$	GM	CPU
SL Schoen	1/1	1	1	1	n/n/y/y	$8\ 10^{-2}$
SL Schoen (NS)	1/1	60	4	4	y/n/n/y	0.3
SL Schoen (mod)	16/1	30	7	4	Y	0.4

Four independent tests were performed using original and modified SL Schoen algorithms. The results are given in Table 1.2. With the original Schoen implementation only one minimum was located. Better results were obtained with the SL Schoen (NS) algorithm. The maximum number of iterations $Iter_{\max}$ was set to 200. Four minima were located with 60 iterations on average. Global minima were found in two out of four runs. The modified SL Schoen algorithm was proved to be the most efficient. The following strategy of sampling was used: on each iteration 16 points were sampled as in MLSL but only the "best start" point was used to start a local minimizer. Two points of global minima plus two second best local minima were located with 30 iterations used on average. The CPU time was in 4 times higher than that of LDS Sobol' ($N/N_r = 256/128$).

It can be concluded that (i) LDS Sobol' and Holton algorithms are more efficient than other considered methods (ii) our implementation of stochastic MLSL is more efficient than that used in [4, 30, 31].

4.2. Problems 2A, B: Griewank function

$$f(\boldsymbol{x}) = 1 + \sum\nolimits_{i=1}^{n} x_i^2/d^2 - \prod_{i=1}^{n}(\cos x_i/\sqrt{i})$$

Configurations:

$$\begin{aligned} &\textit{Problem } 2A.\ n = 2,\ d = 200,\\ &\quad -100 \le x_i \le 100,\ i = 1, 2.\\ &\textit{Problem } 2B.\ n = 10,\ d = 4000,\\ &\quad -600.0 \le x_i \le 600.0,\ i = 1, \ldots, 10. \end{aligned}$$

Global Solution:

$$\begin{aligned} &f(\boldsymbol{x}) = 0.0,\\ &\quad \boldsymbol{x} = \{\mathbf{0.0}\}. \end{aligned}$$

Problem 2A. Both problems belong to the class of "moderate" (M2) [28]. The objective of the test was to evaluate the performance of each method on a problem with a large number of minima. Apart from the global minimum at the origin, this function has some 500 local minima corresponding to the points where the i-th coordinate equals a multiple of $\pi\sqrt{i}$. Because of the very large number of local minima the region of attraction of the global minimum is very small, therefore a very large number of points must be sampled to locate it. In tests with $N/N_r = 32768/128$ in all four runs all tested algorithms successfully located some 20 minima including the global one (Table 2.1A). LM was equal to the number of located minima. The slightly higher value of the CPU time for LDS Holton is explained by the slower process of generating Holton points compared with that for Sobol' or random points.

The performance of MLSL methods largely depends on the sample size. Other tests were performed with smaller samples, with N ranging from 128 to 16384. None of the methods were able to locate the global minimum in all four runs. This may explain results of similar tests reported in [16]: "for the two-dimensional problem the method never really got started. After the first sample of 100 points, only one minimum was found in all cases, after which the method terminated. Global minimum was located once, two runs ended with one of the second best minima while seven runs terminated with a minima with a function value close to one". There is a strong dependence of the algorithm efficiency on the size of samples: for this test problem samples of 100 points were not sufficient to locate the global minimum.

It is known that for problems with a very large number of local minima the Baesian termination criteria do not produce reliable results [16]. Because of this reason a standard MS method which does not use clustering and Bayesian stopping techniques was tested. Table 2.2A presents results of experiments with $N/N_r = 128/128$.

The solution was limited by a single iteration. Results clearly show the advantages of using LDS points: in all four runs of LDS Sobol' (NC) and LDS Holton (NC) algorithms the global minimum was found in contrast with only one successful run of the Random (NC)

Table 2.1A. Comparison of various realizations of MLSL for Problem 2A.

Algorithm	N/N_r	*Iter*	LM	$N_{\min}$	GM	CPU
LDS Sobol'	32768/128	2	23	23	Y	5.0
LDS Holton	32768/128	2	23	23	Y	7.0
Random	32768/128	2	22	22	Y	5.0

Table 2.2A. Comparison of various realizations of MS for Problem 2A.

Algorithm	N/N_r	*Iter*	LM	$N_{\min}$	GM	CPU
LDS Sobol'(NC)	128/128	1	128	107	Y	3.0
LDS Holton (NC)	128/128	1	128	106	Y	3.5
Random (NC)	128/128	1	128	107	n/n/n/y	3.0

Table 2.3A. Comparison of various realizations of SL for Problem 2A.

Algorithm	N/N_r	*Iter*	LM	$N_{\min}$	GM	CPU
SL Schoen	1/1	1	1	1	y/n/n/n	$8\ 10^{-2}$
SL Schoen (NS)	1/1	60	25	25	y/n/n/n	0.3
SL Schoen (mod)	1024/1	400	40	22	Y	6.0

algorithm. LM was nearly equal to the number of located minima. It confirms the high efficiency of the MS approach in test problems with a very large number of local minima. It is worth noting that the CPU time was nearly half of that for the MLSL method (with $N/N_r = 32768/128$, Table 2.1A), while the number of located minima was almost five times higher. The results for the Random (NC) algorithm agree well with the observations made for the same algorithms in [16].

Four independent tests were performed using the SL Schoen algorithm and its variants. The results are given in Table 2.3A. For the original SL Schoen algorithm only one local minimum was found. The global minimum was located only in one run. For the SL Schoen (NS) algorithm the maximum number of iterations was limited to 500. The global minimum was located only once. 60 iterations were needed to locate 25 minima on average. For the modified SL Schoen algorithm with $N/N_r = 1024/1$ the global minimum was located in all four runs. The CPU time for this method was comparable with that for LDS Sobol' algorithm and was only two times higher than that of LDS Sobol' (NC) (with $N/N_r = 128/128$). However, the LDS Sobol' algorithm located four times as many local minima.

It can be concluded that a reliable detection of the global minimum can be achieved with the MLSL method using large samples or alternatively, with the MS method using small samples of points. The SL Schoen algorithm and its variants were less efficient on this problem.

Problem 2B. Problem 2B has an extremely high number of local minima. However, in comparison with the two-dimensional problem 2A it turned out to be much easier to locate the global minimum. This is in line with the results of Törn, Ali and Vjitanen [28]. In tests with the same sample sizes as in various realizations of MLSL for Problem 2A (Table 2.2A) in all four runs all tested algorithms successfully located some 15 minima including the global one (Table 2.1B). Average LM and $N_{\min}$ were similar for all methods. The LDS Holton algorithm was the slowest one.

Table 2.1B. Comparison of various realizations of MLSL for Problem 2B.

Algorithm	N/N_r	*Iter*	LM	$N_{\min}$	GM	CPU
LDS Sobol'	32768/128	2	18	14	Y	1.2
LDS Holton	32768/128	2	20	16	Y	5.1
Random	32768/128	2	19	14	Y	1.2

Table 2.2B. Comparison of various realizations of LDS Sobol' for Problem 2B.

Algorithm	N/N_r	*Iter*	LM	$N_{\min}$	GM	CPU
LDS Sobol'	4096/128	18	43	33	Y	3.1
LDS Sobol'	2048/128	46	79	59	Y	12.3
LDS Sobol'	1024/128	78	107	81	Y	25.2
LDS Sobol'	512/128	98	150	102	Y	26.4

Table 2.2B illustrates the dependence of the number of LM from the ratio $\gamma = N/N_r$ for the Sobol' algorithm. Reduction in sample size results in increasing LM and the corresponding CPU time. In all four runs the global minimum was found. As sample size decreases CPU time increases super linearly with the number of LM. It is interesting to note that although the number of located $N_{\min}$ increased in comparison to previous tests with $N/N_r = 32768/128$, very few second best minima were found. Thus, it can be concluded that the strategy with large samples is more efficient if the objective is to locate only the global minimum.

As in the above case of lower dimension ($n = 2$) the MS method performs much better in terms of locating high a number of local minima then the MLSL method. The results of testing with $N/N_r = 512/128$ are presented in Table 2.3B. Calculations were limited to 10 iterations. In addition to locating the global minimum and a large number of local minima all second best minima were located as well. A comparison between MLSL LDS Sobol' $N/N_r = 32768/128$ (Table 2.1B) and MS LDS Sobol' methods (Table 2.3B) shows that the number of located minima increased almost 60 times while the CPU time increased only 30 times. Since in most cases the objective is to locate only the global minimum, in the case of the MS the sample size and maximum number of iterations $Iter_{\max}$ can be reduced even further. Other tests showed that a reliable detection of the global minimum can be achieved with N/N_r as small as 32/16 and $Iter_{\max} = 1$. The corresponding CPU time for such a case can be reduced to 0.4 s.

The SL Schoen algorithm and its variants are proved to be not very efficient for this problem. Results of testing are given in Table 2.4B. In all cases only one local minimum was located and in two runs out of four the global minimum was not found. The low efficiency of the SL Schoen algorithm and its variants for the case of high dimensional problems is caused by the clustering technique upon which the algorithms are based. The maximum metric $\rho'(\boldsymbol{x}, \boldsymbol{y})$ is limited above by 1 (we recall that the variables to be optimized are scaled to a unit hypercube). The critical threshold $r_{N,\beta}$ given by (19) is always greater than 1 in the first iteration ($\beta > 1$). Thus new points can be assigned to the first cluster found

Table 2.3B. Comparison of various realizations of MS for Problem 2B.

Algorithm	N/N_r	$Iter_{\max}$	LM	$N_{\min}$	GM	CPU
LDS Sobol' (NC)	512/128	10	1280	796	Y	36.0
LDS Holton (NC)	512/128	10	1280	782	Y	61.0
Random (NC)	512/128	10	1280	776	Y	36.0

Table 2.4B. Comparison of various realizations of SL for Problem 2B.

Algorithm	N/N_r	*Iter*	LM	$N_{\min}$	GM	CPU
SL Schoen	1/1	1	1	1	y/n/n/y	$6\ 10^{-2}$
SL Schoen (NS)	1/1	1	1	1	y/n/y/n	6.0
SL Schoen (mod)	32768/1	1	1	1	y/y/n/n	15.0

and further local search procedures can be prohibited. This drawback of the algorithm was discussed by Shoen [17]. It was recommended to limit $r_{N,\beta}$ artificially so that local search would not be inhibited. However, no practical advice was given with regard to the $r_{N,\beta}$ correction. On each successive iteration the sample size grows and the critical distance $r_{N,\beta}$ decreases. For low dimensional problems, after a few iterations $r_{N,\beta}$ becomes less than 1. In this case the algorithm is capable of distinguishing between different clusters and can locate more than one minimum. This explains the limited success of the SL Shoen (mod) algorithms in solving test Problems 1 and 2A.

The MLSL algorithm makes use of the Euclidean metric $\rho(x, y)$, which is limited from above by $\sqrt{n}$. The critical distance r_k given by (18) is likely to be greater than 1 for high dimensional problems especially on the initial iterations. However, $\rho(\boldsymbol{x}, \boldsymbol{y})$ can also be greater than r_k in which case a new local search would not be prohibited. It is also important to note that r_k decreases more rapidly with the increase in sample size and/or the number of iterations than $r_{N,\beta}$. It results in more efficient clustering of the MLSL algorithm in comparison with that of Shoen.

It can be concluded that the reliable detection of the global minimum can be achieved with the MLSL method and rather large samples. The value of the $\gamma = N/N_r$ has a significant impact on the efficiency of the method: increasing γ can result in a dramatic decrease of the CPU time (Table 2.2B).

For problems with a high number of local minima the MS method can be a good alternative to the MLSL method. Even runs with small sample size can produce a large value of $N_{\min}$ (Table 2.2A). A quasi Monte Carlo variant of the MS method is much more efficient than the stochastic one. SL Schoen's algorithm and its variants proved to be less efficient for such problems.

4.3. Problems 3A, B: Shubert function

Problem 3A. $n = 3$

$$
\begin{aligned}
f(\boldsymbol{x}) = {} & (\pi/n)\left\{k_1 \sin^2(\pi y_1) + \sum_{i=1}^{n-1} (y_i - k_2)^2[1 + k_1 \sin^2(\pi y_{i+1})] + (y_n - k_2)^2\right\} \\
& + \sum_{i=1}^{n} u(x_i, 10, 100, 4), \\
y_i = {} & 1 + 0.25(x_i + 1), k_1 = 10 \quad \text{and} \quad k_2 = 1, \\
& -10 \le x_i \le 10, i = 1, 2, 3.
\end{aligned}
$$

$u(x_i, a, k, m)$ is a penalty function defined by

$$u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a, \\ 0, & -a \le x_i \le a, \\ k(-x_i - a)^m, & x_i < -a, \end{cases}$$

Global Solution

$$f(\boldsymbol{x}) = 0.0,$$
$$\boldsymbol{x} = (-1.0, -1.0, -1.0).$$

Problems 3A and 3B belong to the class of "easy" (E2) problems [28]. Problem 3A has approximately 5^3 local minima. The objective of this test was to test the performance of the LDS Sobol', LDS Holton and Random methods on problems with a large number of minima. In tests with $N/N_r = 32/16$ and $Iter_{\max} = 10$ in all four runs all algorithms successfully located approximately the same number of local minima including the global one (Table 3.1A). LDS Sobol'(NS) showed slightly better performance.

Problem 3B. $n = 5$.

$$f(\boldsymbol{x}) = k_3 \left\{ \sin^2(\pi k_4 x_1) + \sum_{i=1}^{n-1} (x_i - k_5)^2 [1 + k_6 \sin^2(\pi k_4 x_{i+1})] + \left(x_n - k_5\right)^2 [1 + k_6 \sin^2(\pi k_7 x_n)] \right\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4),$$
$$k_3 = 0.1, k_4 = 3, k_5 = 3, k_6 = 1, k_7 = 2.$$
$$-5 \le x_i \le 5, i = 1, \ldots, 5.$$

Global Solution

$$f(\boldsymbol{x}) = 0.0,$$
$$\boldsymbol{x} = (1.0, 1.0, 1.0, 1.0, 1.0).$$

Table 3.1.A. Comparison of various realizations of MLSL for Problem 3A.

Algorithm	N/N_r	$Iter_{\max}$	LM	$N_{\min}$	GM	CPU
LDS Sobol'(NS)	32/16	10	13	12	Y	$2.6\ 10^{-2}$
LDS Holton(NS)	32/16	10	13	11	Y	$4.6\ 10^{-2}$
Random(NS)	32/16	10	12	10	Y	$2.6\ 10^{-2}$

Table 3.1B. Comparison of various realizations of MLSL for Problem 3B.

Algorithm	N/N_r	$Iter_{\max}$	LM	$N_{\min}$	GM	CPU
LDS Sobol' (NS)	1024/512	3	68	63	Y	2.1
LDS Holton (NS)	1024/512	3	53	51	y/n/y/y	2.4
Random (NS)	1024/512	3	61	58	Y	1.9

This problem has approximately 15^5 local minima. The objective of this test was to test the performance of the LDS Sobol', LDS Holton and Random methods on problems with a very large number of minima. In tests with $N/N_r = 1024/512$ and $Iter_{\max} = 3$ in all four runs LDS Sobol'(NS) and Random(NS) algorithms successfully located some 60 local minima including the global one (Table 3.1B). There were only three successful runs of the LDS Holton (NS) algorithm. The number of minima located by the LDS Holton (NS) algorithm was also lower than that for other algorithms. This can be explained by the inferior uniformity properties of the Holton LDS even at moderate dimensions.

4.4. Problems 4A, B, C: Schaffler function

$$f(\boldsymbol{x}) = 1 + 590 \sum_{i=2}^{n} (x_i - x_{i-1})^2 + 6x_1^2 - \cos(12x_1),$$
$$-1.05 \le x_i \le 2.95, i = 1, \ldots, n.$$

Configurations:

Problem 4*A*. $n = 30$.
Problem 4*B*. $n = 40$.
Problem 4*C*. $n = 50$.

Global Solution

$$f(\boldsymbol{x}) = 0.0,$$
$$\boldsymbol{x} = \{\mathbf{0.0}\}.$$

All three problems belong to the class of "moderate" (M2) [28]. The objective of this test with 5 known solutions was to test the performance of the LDS Sobol', LDS Holton and Random methods on high-dimensional problems. LDS have better uniformity properties than pseudorandom grids. However, this advantage diminishes as the dimensionality n increases. As explained in Section 2, for high-dimensional problems the usage of LDS still can be more efficient than pseudorandom sampling if an objective function $f(\boldsymbol{x})$ strongly depends only on a subset of variables. For such problems an effective dimension number s

Table 4.1A. Comparison of various realizations of MLSL for Problem 4A.

Algorithm	N/N_r	*Iter*	LM	$N_{\min}$	GM	CPU (s)
LDS Sobol'	8192/256	1	220	3	Y	30.5
LDS Holton	8192/256	1	98	2	y/n/y/n	18.5
Random	8192/256	1	228	3	Y	31.2

can be much smaller than n. However, this is not the case for the test Problem 4. Apart from variable x_1, all other variables are equally important. This explains why LDS Sobol' and Random methods showed almost the same efficiency (Tables 4.1A, B, and C). Uniformity properties of Holton LDS rapidly degrade as n grows. Thus for high-dimensional problems the MLSL method based upon Holton LDS sampling becomes less efficient than a stochastic variant of MLSL. Values N and N_r given in Tables 4.1A, B, and C are the smallest sample sizes for which a global minimum was found in all four runs for LDS Sobol' and Random methods.

Problem 4A. $n = 30$

For LDS Holton the global minimum was found in two out of four runs for Problem 4A and Problem 4B, while for Problem 4C this algorithm failed to locate it. A comparison between Problems 4A and Problem B for LDS Sobol' and Random shows that for successful location of the global minimum in all four runs it was necessary to increase N in two times and N_r—in 16 times. It resulted in a 30 fold increase of the CPU time.

Problem 4B. $n = 40$

Increasing the dimensionality from $n = 40$ to $n = 50$ resulted in N increasing 64 fold. At the same time N_r increased only 4 fold and the CPU time increased approximately 8 fold.

Problem 4C. $n = 50$

For Problem 4C N_r was nearly equal to LM. This is because clustering becomes less efficient as dimensionality grows. Choosing larger σ in (18) may increase the cluster size and thus decrease LM. However, for consistency with other tests experiments σ was kept equal to 2.0.

Table 4.1B. Comparison of various realizations of MLSL for Problem 4B.

Algorithm	N/N_r	*Iter*	LM	$N_{\min}$	GM	CPU (s)
LDS Sobol'	16384/4096	1	4028	4	Y	$1.1 \cdot 10^3$
LDS Holton	16384/4096	1	1142	2	n/n/y/y	348.1
Random	16384/4096	1	4031	3	Y	$1.1 \cdot 10^3$

Table 4.1C. Comparison of various realizations of MLSL for Problem 4C.

Algorithm	N/N_r	*Iter*	LM	$N_{\min}$	GM	CPU (s)
LDS Sobol'	1048576/16384	1	16206	3	Y	$7.5 \cdot 10^3$
LDS Holton	1048576/16384	1	5665	2	N	$3.3 \cdot 10^3$
Random	1048576/16384	1	16253	3	Y	$7.6 \cdot 10^3$

5. Conclusion

In this study QMC and stochastic variants of MLSL were compared. The Program SobolOpt employing the discussed techniques was applied to a number of test problems. When compared with other implementations of MLSL reported in the literature, it showed a superior performance. It was proved that application of LDS results in a significant reduction in computational time for low and moderately dimensional problems. Two different LDS were tested and their efficiency was analyzed. Uniformity properties of Holton LDS degrade as dimensionality grows and for high dimensional problems the MLSL method based on Holton LDS becomes less efficient than the stochastic MLSL method. Sobol' LDS can still be superior to pseudorandom sampling especially for problems in which an objective function strongly depends only on a subset of variables.

To increase the probability of finding the global minimum, the full sample size should be increased with the increase of the dimensionality of a problem. However, it may not be very practical if the objective function is difficult to evaluate. The ratio of the full/reduced sample size γ should be kept high to reduce the computational time. It was shown that the use of a large total number of sampled points is more efficient than that of a small one if the objective is to locate only a global minimum as opposed to locate as many local minima as possible.

The developed technique was generalized to account for mixed continuous and discrete variables (MINLP). Preliminary tests have shown a good performance of the multistage methods based on LDS sampling for constrained MINLP problems. Results will be presented in a future paper.

Acknowledgment

We are grateful to C. Pantelides for his support and interest in this work. We would like to thank C. Adjiman and S. Zakovic for interesting discussions, the latter also for providing a code for the comparison of different implementations of the MLSL method. We express our gratitude to I. Sobol' for numerous discussions and the invaluable help in improving the techniques presented.

One of the authors (SK) gratefully acknowledges the financial support of the United Kingdom's Engineering and Physical Sciences Research Council (EPSRC) under Platform Grant GR/N08636.

References

1. I.I. Artobolevskii, M.D. Genkin, V.K. Grinkevich, I.M. Sobol', and R.B. Statnikov, "Optimization in the theory of machines by an LP-search," Dokl. Akad. Nauk SSSR, vol. 200, pp. 1287–1290, 1971 (in Russian).
2. C.G.E. Boender, The Generalized Multinomial Distribution: A Bayesian Analysis and Applications, Ph.D. Dissertation, Erasmus Universiteit Rotterdam, Centrum voor Wiskunde en Informatica: Amsterdam, 1984.
3. P. Bratley, B.L. Fox, and H. Niederreiter, "Implementation and tests of low-discrepancy sequences," ACM Trans. Model, Comput. Simulation, vol. 2, pp. 195–213, 1992.
4. L.C.W. Dixon and M. Jha, "Parallel algorithms for global optimization," Journal of Optimization Theory and Applications, vol. 79, no. l, pp. 385–395, 1993.
5. C.A. Floudas and Panos M. Pardalos (Eds.), State of the Art in Global Optimization: Computational Methods and Applications, Princeton University Press, 1996.
6. C. Floudas, M. Pardalos Panos, C. Adjiman, W.R. Esposito, Z. Gumus, S.T. Harding, J.L. Klepeis, C.A. Meyer, and C.A. Schweiger, Handbook of Test Problems in Local and Global Optimization, Kluwer Academic Publishers, 1999.
7. C. Floudas and P.M. Pardalos (Eds.), Encyclopedia of Optimization, Kluwer Academic Publishers, 2001.
8. J.H. Holton, "On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals," Numer. Math, vol. 2, pp. 84–90, 1960.
9. R. Horst and P.M. Pardalos (Eds.), Handbook of Global Optimization, Kluwer Academic Publishers, 1995.
10. M. Locatelli and F. Schoen, "Simple linkage: Analysis of a threshold-accepting global optimization method," Journal of Global Optimization, vol. 9, pp. 95–111, 1996.
11. NAG Library, http://www.nag.co.uk, 2002.
12. H. Niederreiter, "Point sets and sequences with small discrepancy," Monatsh. Math., vol. 104, pp. 273–337, 1987.
13. H. Niederreiter, Random Number Generation and Quasi-Monte Carlo Methods, SIAM: Philadelphia, 1992.
14. S. Paskov and J.F. Traub, "Faster evaluation of financial derivatives," The Journal of Portfolio Management, vol. 22, no. 1, pp. 113–120, 1995.
15. A.H.G. Rinnooy Kan and G.T. Timmer, "Stochastic global optimization methods, part I, clustering methods," Mathematical Programming, vol. 39, pp. 27–56, 1987.
16. A.H.G. Rinnooy Kan and G.T. Timmer, "Stochastic global optimization methods, part II, multilevel methods," Mathematical Programming, vol. 39, pp. 57–78, 1987.
17. F. Schoen, "Random and quasi-random linkage methods in global optimization," Journal of Global Optimization, vol. 13, pp. 445–454, 1998.
18. I.M. Sobol', "On the distribution of points in a cube and the approximate evaluation of integrals," Comput. Math. Math. Phys., vol. 7, pp. 86–112, 1967.
19. I.M. Sobol', "On the systematic search in a hypercube," SIAM J. Numer. Anal., vol. 16, pp. 790–793, 1979.
20. I.M. Sobol', "On an estimate of the accuracy of a simple multidimensional search," Soviet Math. Dokl., vol. 26, pp. 398–401, 1982.
21. I.M. Sobol', "On the search for extreme values of functions of several variables satisfying a general Lipschitz condition," USSR Comput. Math. Math. Phys., vol. 28, pp. 112–118, 1988.
22. I.M. Sobol', "An efficient approach to multicriteria optimum design problems," Survey Math. Ind., vol. 1, pp. 259–281, 1992.
23. I.M. Sobol', Primer for the Monte Carlo Method, CRC Press: Florida, 1994.
24. I.M. Sobol', "On quasi-Monte Carlo integrations," Mathematics and Computers in Simulation, vol. 47, pp. 103–112, 1998.
25. I.M. Sobol' and B.V. Shukhman, "Integration with quasirandom sequences," Numerical Experience, Internat. J. Modern Phys. C, vol. 6, no. 2, pp. 263–275, 1995.
26. A.G. Sukharev, "Optimal strategies of the search for an extremum," Zh. Vychisl. Mat. i Mat. Fiz, vol. 11, pp. 910–924 1971 (in Russian).
27. A. Törn and A. Zilinskas, Global Optimization, Lecture Notes in Computer Science 350, Springer: Berlin, 1989.
28. A. Törn, M. Afi, and S. Vjitanen, "Stochastic global optimization, problem classes and solution techniques," Journal of Global Optimization, vol. 14, pp. 437–447, 1999.

29. J.F. Traub and H. Wozniakowski, A General Theory of Optimal Algorithms, Academic Press: New York, 1980.
30. S. Zakovic, Global Optimization Applied to an Inverse Light Scattering Problem, Ph.D. Thesis, University of Hertfordshire: Hartfield, 1997.
31. S. Zakovic, Z. Ulanowski, and M.C. Bartholomew-Biggs, "Application of global optimisation to particle identification using light scattering," Inverse Problems, vol. 14, pp. 1053–1067, 1998.