Towards “Ideal Multistart”. A stochastic approach for locating the minima of a continuous function inside a bounded domain

C. Voglis *, I.E. Lagaris

*Corresponding author.
E-mail address: voglis@cs.uoi.gr (C. Voglis).

A stochastic global optimization method based on Multistart is presented. In this, the local search is conditionally applied with a probability that takes into account the topology of the objective function at the detail offered by the current status of exploration. As a result, the number of unnecessary local searches is drastically limited, yielding an efficient method.

Abstract

A stochastic global optimization method based on Multistart is presented. In this, the local search is conditionally applied with a probability that takes into account the topology of the objective function at the detail offered by the current status of exploration. As a result, the number of unnecessary local searches is drastically limited, yielding an efficient method.

1. Introduction

Global optimization (GO) has received a lot of attention in recent years [1], due to the ever emerging scientific and industrial demand. For instance the description of the stable conformations of a molecule [2–4], the management of mutual funds [5–8], location and allocation issues [9,10], engineering design and the design of drugs [11,12], to mention a few topics, are in need of efficient global optimization techniques.

There exist several categories of GO methods. We distinguish two main classes: the deterministic [13,14] and the stochastic one. For a detailed account on classification of stochastic algorithms we refer to [15]. Deterministic methods provide a theoretical guarantee of locating the global optimum. Stochastic methods offer only a probabilistic (asymptotic) guarantee: their convergence proofs usually declare that the global optimum will be identified in infinite time with probability one. Moreover, stochastic methods adapt better to black-box formulations and extremely ill-behaved functions, whereas deterministic methods are usually based on at least some theoretical assumptions such as Lipschitz continuity and heavily depends on the problem at hand. A direct comparison between these two approaches may be found in [16], where the authors conclude that the stochastic approach is to be preferred. In addition deterministic methods suffer from the problem of dimensionality. For example, the complexity of interval global optimization [17] rises exponentially with the problem’s dimension.

The problem we are interested in, may be expressed as:

\[
\text{Find all } x^*_i \in S \subset \mathbb{R}^n \text{ that satisfy: } \\
x^*_i = \arg \min_{x \in S_i} f(x), \quad S_i = S \cap \{x, \|x - x^*_i\| < \epsilon\},
\]

(1)

\(S\) is considered to be a bounded domain of finite measure and \(\epsilon\) a positive infinitesimally small number. We are adopting the stochastic class of methods. One of the most widely used stochastic algorithms is the so called Multistart [18]. Its popularity stems from its simplicity and inherent parallelization [19–22]. Many stochastic methods have been developed around it.

starting from the classic papers of [18,23–25] were the popular Single Linkage Clustering, Density Clustering and Multi-Level Single Linkage procedures were introduced. Törn and Viitanen in [26] presented a Topographical Clustering algorithm which was extended by Ali and Storey in [27] to the well known Topographical Multi-Level Single Linkage algorithm. More recently Hart in his PhD dissertation [28] proposes an adaptive method based on clustering and local searches, Locatelli [29] introduces the family of Random Linkage algorithms and Schoen [30] and Locatelli [31] give an analysis Two-phase methods. More recently, Liang et. al. [32] introduce a function’s landscape approximation, Bolton et. al. [33] provide a parallel framework based on clustering, while Tsoulos and Lagaris [34] proposed the so called typical distance clustering. Also related software may be found in [35].

In Multistart a point is sampled uniformly from the feasible region, and subsequently a local search is started from it. The weakness of this algorithm is that the same local minima may be found over and over again, wasting so computational resources. For this reason clustering methods have been developed that attempt to avoid repetitive discovery of the same minima [23–25,34,20].

The Multistart algorithm is presented below:

<table>
<thead>
<tr>
<th>Multistart Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Initialize:</strong> Set k=1</td>
</tr>
<tr>
<td>Sample $x \in S$</td>
</tr>
<tr>
<td>$y_k = \mathcal{P}(x)$</td>
</tr>
<tr>
<td><strong>Termination Control:</strong> If a stopping rule applies, STOP.</td>
</tr>
<tr>
<td><strong>Sample:</strong> Sample $x \in S$</td>
</tr>
<tr>
<td><strong>Main step:</strong> $y = \mathcal{P}(x)$</td>
</tr>
<tr>
<td>If $(y \notin {y_i, i = 1, 2, \ldots, k})$ Then</td>
</tr>
<tr>
<td>$k = k + 1$</td>
</tr>
<tr>
<td>$y_k = y$</td>
</tr>
<tr>
<td>Endif</td>
</tr>
<tr>
<td><strong>Iterate:</strong> Go back to the Termination Control step.</td>
</tr>
</tbody>
</table>

The “region of attraction” of a local minimum associated with a local search procedure $\mathcal{P}$ is defined as:

$$A_i \equiv \{x \in S, \mathcal{P}(x) = x_i^*\}$$  \hspace{1cm} (2)

where $\mathcal{P}(x)$ is the minimizer returned when the local search procedure $\mathcal{P}$ is started at point $x$. If $S$ contains a total of $w$ local minima, from the definition above follows:

$$\bigcup_{i=1}^{w} A_i = S.$$  \hspace{1cm} (3)

Let $m(A)$ indicate the Lebesgue measure of $A \subseteq \mathbb{R}^n$. If we assume a deterministic search $\mathcal{P}$, then the regions of attraction do not overlap, i.e. $A_i \cap A_j = \emptyset$ for $i \neq j$, and from Eq. (3) one obtains:

$$m(S) = \sum_{i=1}^{w} m(A_i).$$  \hspace{1cm} (4)

If a point in $S$ is sampled from a uniform distribution, the a priori probability $p_i$ that it is contained in $A_i$ is given by $p_i = \frac{m(A_i)}{m(S)}$. If $K$ points are sampled from $S$, the a priori probability that at least one point is contained in $A_i$ is given by:

$$1 - \left(1 - \frac{m(A_i)}{m(S)}\right)^K = 1 - (1 - p_i)^K.$$  \hspace{1cm} (5)

From the above we infer that for large enough $K$, this probability tends to one, i.e. it becomes “asymptotically certain” that at least one sampled point will be found to belong to $A_i$. This holds $\forall A_i$, with $m(A_i) \neq 0$.

In this article we first define the “Ideal Multistart”, a variation of Multistart in which every local minimum is found only once. This ideal version assumes that the region of attraction of a minimizer is determined as soon as the minimizer is located. Since this is a false hypothesis this version is of no practical value. It offers however a framework and a goal to work towards.

In section (2), we lay-out the new ideas involved and we present the corresponding algorithm, while in section (3), we give a description of the numerical experiments that were performed along with the results. Finally in section (5), our conclusions are summarized and we give a recommendation for future research.

### 2. Description of the method

"Ideal Multistart" starts by sampling a point from $S$ and applying a local search leading to the first minimum $y_1$, with region of attraction $A_1$. Sampling points from $S$ is continued until a point is found that does not belong to $A_1$. Once such a point is encountered, a local search is performed that leads to the second minimum $y_2$, having a region of attraction $A_2$. The next
sample point from which a local search will start, is a point that belongs neither to \( A_1 \) nor to \( A_2 \), i.e. it does not belong to their union \( (A_1 \cup A_2) \). This procedure goes on, until a stopping rule instructs termination. The detailed algorithm is laid out in the following paragraph.

2.1. Ideal Multistart

Ideal Multistart Algorithm

**Initialize:** Set \( k=1 \)
- Sample \( x \in S \)
- \( y_k = \mathcal{L}(x) \)

**Termination Control:** If a stopping rule applies, STOP.

**Sample:** Sample \( x \in S \)

**Main step:** If \( (x \notin \bigcup_{i=1}^{k} A_i) \) Then
- \( y = \mathcal{L}(x) \)
- \( k = k + 1 \)
- \( y_k = y \)

**Iterate:** Go back to the Termination Control step.

This algorithm invokes the local search procedure only \( w \) times, \( w \) being the number of existing minima of \( f(\cdot) \) in \( S \). The main step is deterministic and requires the regions of attraction \( A_i \) of the already located minima to be known, which is not the case in practice. Hence we apply a stochastic modification to the main step, by allowing the local search to be performed with a probability, namely:

**Main step (Stochastic):**
- Calculate the probability \( p \), that \( x \notin \bigcup_{i=1}^{k} A_i \)
- Draw a random number \( \xi \in (0,1) \) from a uniform distribution
- If \( (\xi < p) \) Then
  - \( y = \mathcal{L}(x) \)
  - If \( (y \notin \{y_i, i = 1, 2, \ldots, k\}) \) Then
    - \( k = k + 1 \)
    - \( y_k = y \)
  - Endif
- Endif

**Fig. 1.** A point \( x \) that would lead to a new minimum \( y \), is inside the overlap region of the spheres around two recovered minima \( y_1 \) and \( y_2 \).
This step requires the probability that a point does not belong to the region of attraction of any of the minima collected so far. This requirement is easier to fulfill, since even with a low accuracy estimate for the probability, the algorithm will succeed. Notice that an overestimated probability \((p \rightarrow 1)\) will transform the algorithm into the usual \textit{Multistart}. On the other hand underestimation \((p \rightarrow 0)\) is not of considerable cost, since no local search is performed. Performance however will be optimized if reasonably accurate estimates for the probability can be calculated. Several ways may be designed to accomplish this goal. We suggest one in the following paragraph.

2.2. Estimating the local search probability

The required probability may depend on several factors, such as the distance from existing minimizers, the direction of the gradient, the number of times each minimizer has been discovered, etc. We consider how each factor influences the probability and combine them together to get the required estimate.

Let us define the \textit{maximum attractive radius} (MAR) as:

\[
R_i = \max_j \left\{ \|x_j^{(i)} - y_i\| \right\},
\]

where \(x_j^{(i)}\) are the sampled points which led the subsequent local search to the \(i^{th}\) minimizer \(y_i\).

\[\text{Fig. 2. Illustration of the modified line search.}\]
Given a sampled point $x$, let $y$ be anyone of the recovered minimizers, with MAR denoted by $R$. If $\|y - x\| < R$, then $x$ is likely to be inside the region of attraction of $y$. If however $\nabla f(x)^T(y - x) \geq 0$, i.e. the direction from $x$ to $y$ is ascent, then $x$ is likely to be outside $y$'s region of attraction. Letting $z = \|y - x\| / R$, then an estimate of the probability that $x \notin A(y)$ may be given by:

$$
p(x \notin A(y)) = \begin{cases} 
1, & \text{if } z > 1 \text{ or } \nabla f(x)^T(y - x) \geq 0, \\
\phi(z, l) \cdot \left[1 + \frac{(y - x)^T \nabla f(x)}{\|y - x\| \|\nabla f(x)\|}\right], & \text{otherwise},
\end{cases}
$$

(7)

$l$ is the number of times $y$ has been recovered so far, while $\phi(z, l)$ is a model with the following properties:

\[
\begin{align*}
\lim_{z \to 0} \phi(z, l) & \to 0, \\
\lim_{z \to 1} \phi(z, l) & \to 1, \\
\lim_{l \to \infty} \phi(z, l) & \to 0, \\
0 & < \phi(z, l) < 1.
\end{align*}
\]

(8)

Notice that the factor inside the square brackets in Eq. (7), varies from zero to one, as the gradient from anti-parallel becomes perpendicular to $y - x$.

The probability that $x \notin \bigcup_{i=1}^{L} A_i$ is given by the product $\prod_{i=1}^{L} p(x \notin A_i)$ and may now be approximated by the probability that $x \notin A_y, A_n$, being the region of attraction of the nearest to $x$ discovered minimizer $y_n$. The rationale for this approximation is that if $x \notin B(y, R) \forall i \neq n$, where $B(y, R)$ is a sphere of radius $R$ centered at $y$, then the above approximation is exact since all other probabilities as following from Eq. (7) equal 1. If on the other hand $x$ is inside the intersection of two or more overlapping spheres, the product of small terms may result to too small a probability for a point that could lead to a new minimum (see in Fig. 1, an example). The spheres are expected to overlap, due to the manner their radii are chosen by Eq. (6). Hence the approximation is prudent, and essentially in most cases does not overestimate the local search probability. One may employ alternative approximations, by considering for example the first two (or more) nearest minimizers. This is an issue that needs further consideration and is outside the scope of the present article.

2.3. Local search properties

The probability model is based on distances from the discovered minima. It is implicitly assumed that the closer to a minimum a point is, the greater the probability that falls inside its region of attraction. This implies that the regions of attraction are contiguous and surround the minima. This is not true for all local search procedures and hence this assumption influences the local search choice. For example widely used methods such as Newton or quasi Newton, employing either a line
search or a trust region strategy, create disjoint regions of attraction. Hence these methods have to be modified so that their regions of attraction are contiguous, resembling those of a descent method with an infinitesimal step. In Fig. 3 we connect start-points (marked by +) to the minimum they arrive via a local search. This is a desirable local search since its regions of attraction are contiguous. Start points are attracted towards the close by minima.

In this work we apply the BFGS method with a modified line search. This modification creates contiguous regions of attraction ensuring a strictly descent path [23].

We present the associated algorithm below:

**Modified Local Search Algorithm**

Input:
\[ k = 0, B_k = I, \epsilon > 0 \]

**Step 1 (Calculate descent direction):**
\[
p_k = -B_k^{-1}\nabla f(x_k)
\]
If \( \| \nabla f(x) \| > \epsilon \) Then
\[
p_k = -\nabla f(x_k)
\]
End if

**Step 2 (Line search):**
\[
\min_a (f(x_k + ap_k)), \text{ yielding } a_k
\]

**Step 3 (Next iterate):**
\[
x_{k+1} = x_k + a_k p_k
\]

**Step 4 (Update approximation):**
\[
\gamma_k = \nabla f(x_{k+1}) - \nabla f(x_k)
\]
\[
\delta_k = x_{k+1} - x_k
\]
\[
B_{k+1} = \text{bfgs_update}(B_k, \gamma_k, \delta_k)
\]

**Step 5 (Termination Control):**
If termination conditions are met stop, Else set \( k \leftarrow k + 1 \) and repeat from Step 1.

To illustrate the behavior of this normalization at Step 1 of the line search we provide Fig. 2a–d. The single minimum appearing in Fig. 2d is the first minimum in Fig. 2b. Note that in Fig. 2c the line search ends up to the nearest minimum while that of Fig. 2a in a different minimum further apart.

In Fig. 4 we connect start-points (marked by +) to the minimum the arrive via a different local search. This illustrates an undesirable local search since its regions of attraction are disjoint. Start points are attracted towards distant minima.

![Fig. 4. An improper local search, with disjoint regions of attraction.](image-url)
2.4. Asymptotic guaranty

The probability that minimizer $y$ is found with one trial is given by:

$$p_y^{(i)} = \int_{\mathbb{R}} p_{LS}^{(i)}(x) \frac{dx}{|S|},$$  \hspace{1cm} (9)

where $1/|S|$ is the pdf of the uniform distribution and $p_{LS}^{(i)}(x)$ is the local search probability at $x$. The superscript $i$ denotes the state of the process, i.e. the number of minima discovered so far, the number of times each minimizer is found, the MAP's, etc. The probability that after $k$ trials $y$ is not found is then given by:

$$p_y^{(k)} = \prod_{i=1}^{k} (1 - p_y^{(i)}) \leq \left(1 - \min_{i} \{p_y^{(i)}\}\right)^k.$$  \hspace{1cm} (10)

From the definition of $p_y^{(i)}$ in Eq. (9), we have:

$$p_y^{(i)} = \int_{x \in A_y} p_{LS}^{(i)}(x) \frac{dx}{|S|} + \int_{x \in A_2(y)} p_{LS}^{(i)}(x) \frac{dx}{|S|},$$  \hspace{1cm} (11)

where

$$A_1(y) = \{x \in A(y) : (y_c - x)^T f(x) \leq 0\},$$
$$A_2(y) = \{x \in A(y) : (y_c - x)^T f(x) > 0\},$$

and $y_c = y_c(x)$ is the closest to $x$ discovered minimizer.

If $y$ is not found yet (and hence $y_c \neq y$), then $A_2(y) \neq \emptyset$ and hence $|A_2(y)| \neq 0$. Note that

$$\forall x \in A_2(y), \quad p_{LS}^{(i)}(x) = 1$$

and hence from Eq. (11)

$$p_y^{(i)} \geq \frac{|A_2(y)|}{|S|} > 0, \quad \forall i = 1, 2, \ldots, k.$$

At the limit as $k \to \infty$ we deduce from above and Eq. (10) that $p_y^{(k)} \to 0$, i.e. asymptotically all minimizers will be found.

2.5. A model for $\phi(z,l)$

Many models may be constructed with the desired properties described in (8). We propose one that is simple to visualize and easy to implement.
\[ \phi(z, l) = ze^{-l^2(z-1)^2}, \quad \forall z \in (0, 1). \] (12)

A graphical representation is depicted in (Fig. 5).

2.6. The ADAPT Algorithm

The proposed algorithm, in summary, is presented below:

<table>
<thead>
<tr>
<th>ADAPT Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong></td>
</tr>
<tr>
<td>The input function ( f : \mathbb{R}^n \rightarrow \mathbb{R} )</td>
</tr>
<tr>
<td>The search domain ( S \subseteq \mathbb{R}^n )</td>
</tr>
<tr>
<td>A local search procedure ( \mathcal{L}(x) ) having the properties described in Section 2.3.</td>
</tr>
</tbody>
</table>

**Initialize:** Set \( k = 1 \)

**Sample:** \( x \in S \)

\( y_k = \mathcal{L}(x) \)

\( r_k = \|x - y_k\|, n_k = 1 \)

**Termination Control:** If a stopping rule applies, STOP.

**Sample:** \( x \in S \)

**Main step:** \( i = \text{argmin}_{j=1,...,k} \|x - y_j\| \)

\( d = \|x - y_i\| \)

If \( (d < r_i) \) Then

If \( (\nabla f(x)^T(y_i - x) < 0) \) Then

\( z = \frac{y_i - x}{r_i} \)

\( p = \phi(z, n_i) \left[ 1 + \frac{(y_i - x)^T \nabla f(x)}{\|y_i - x\|^2 \nabla f(x)} \right] \)

Else

\( p = 1.0 \)

Endif

Else

\( p = 1.0 \)

Endif

Let \( \xi \) be a uniform random in \([0, 1]\)

If \( (\xi < p) \) Then

\( y = \mathcal{L}(x) \)

If \( (y \text{ is new minimum}) \) Then

\( k = k + 1, r_k = \|x - y_k\|, n_k = 1 \)

Else

(We discovered the \( l \)-th local minimum)

\( r_l = \max(r_l, \|x - y_l\|), n_l = n_l + 1 \)

Endif

Else

(Assuming that \( x \) belongs in the region of attraction of the \( i \)-th minimum)

\( r_i = \max(r_i, \|x - y_i\|), n_i = n_i + 1 \)

Endif

**Iterate:** Go back to the Termination Control step.

3. Experiments and comparison

The method has been tested on a number of test problems that are listed in Appendix A. These test functions have been used in the past by many authors and hence they constitute a convenient platform for comparison. We count for every problem the number of local searches, the number of function and gradient evaluations and we report averages on thirty experiments performed with different random number sequences. We also count the number of minima found. All experiments used the “**Double-Box**” stopping rule [36], with the suggested compromise factor (0.5). The local search used by ADAPT is a modification of BFGS so that the resulting regions of attraction have the properties described in Section 2.3. A comparison is made with the standard “**Multistart**” with the “**Topological Multilevel Single Linkage**” (TML) method [35] and with MinFinder [34].

All of the above methods use as a local minimizer subroutine **TOLMIN** due to Powell [54]. We coded Multistart, while the codes for TML and MinFinder were obtained from the corresponding authors and were run with the default parameters. Observing the results listed in Table 1 we note that the performance of the new method (ADAPT) is overall superior. MinFinder has similar performance on functions \( M_0, Borne, Shubert(N = 5, 10) \) while it has an edge with functions having a periodicity in their contour plots like Holder, Levy No3, Rastrigin(\( N = 2 \)), and Shubert(\( N = 2 \)).
Table 1

Method comparison using the Double-Box rule [36].

<table>
<thead>
<tr>
<th>Function</th>
<th>TML Min.</th>
<th>TML GC</th>
<th>TML LS</th>
<th>Multistart Min.</th>
<th>Multistart GC</th>
<th>Multistart LS</th>
<th>MinFinder Min.</th>
<th>MinFinder GC</th>
<th>MinFinder LS</th>
<th>Adapt Min.</th>
<th>Adapt GC</th>
<th>Adapt LS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ackley</td>
<td>121</td>
<td>10,259</td>
<td>14,457</td>
<td>1207</td>
<td>121</td>
<td>23,281</td>
<td>36,543</td>
<td>3054</td>
<td>121</td>
<td>11,926</td>
<td>208</td>
<td></td>
</tr>
<tr>
<td>Bird</td>
<td>158.5</td>
<td>84,798</td>
<td>103,889</td>
<td>2507</td>
<td>141.5</td>
<td>212,196</td>
<td>150,529</td>
<td>3537</td>
<td>172.8</td>
<td>122,639</td>
<td>145,460</td>
<td></td>
</tr>
<tr>
<td>Bohachefsky</td>
<td>25</td>
<td>139,190</td>
<td>125,684</td>
<td>2369</td>
<td>25</td>
<td>241,501</td>
<td>187,175</td>
<td>2547</td>
<td>18.7</td>
<td>25,907</td>
<td>32,243</td>
<td></td>
</tr>
<tr>
<td>Giunta</td>
<td>196</td>
<td>104,812</td>
<td>16,606</td>
<td>719</td>
<td>196</td>
<td>45,688</td>
<td>67,311</td>
<td>1212</td>
<td>196</td>
<td>18,753</td>
<td>20,972</td>
<td></td>
</tr>
<tr>
<td>Griewank</td>
<td>527.2</td>
<td>1,883,423</td>
<td>1,461,617</td>
<td>39,090</td>
<td>526.4</td>
<td>1,912,452</td>
<td>1,892,111</td>
<td>38,727</td>
<td>529</td>
<td>1,133,908</td>
<td>1,284,982</td>
<td></td>
</tr>
<tr>
<td>Guilli Hills</td>
<td>25</td>
<td>81,153</td>
<td>69,487</td>
<td>2451</td>
<td>24.9</td>
<td>87,411</td>
<td>76,563</td>
<td>2617</td>
<td>24.8</td>
<td>22,901</td>
<td>23,570</td>
<td></td>
</tr>
<tr>
<td>Holder</td>
<td>85</td>
<td>28,749</td>
<td>23,346</td>
<td>622</td>
<td>85</td>
<td>69,098</td>
<td>34,768</td>
<td>988</td>
<td>85</td>
<td>8289</td>
<td>8977</td>
<td></td>
</tr>
<tr>
<td>Langermann</td>
<td>257</td>
<td>129,521</td>
<td>124,360</td>
<td>3566</td>
<td>260</td>
<td>185,669</td>
<td>111,478</td>
<td>4169</td>
<td>270</td>
<td>503,470</td>
<td>500,675</td>
<td></td>
</tr>
<tr>
<td>Levy No3</td>
<td>527</td>
<td>170,541</td>
<td>171,643</td>
<td>6999</td>
<td>527</td>
<td>494,578</td>
<td>277,868</td>
<td>8909</td>
<td>527</td>
<td>59,830</td>
<td>91,479</td>
<td></td>
</tr>
<tr>
<td>Levy No5</td>
<td>508</td>
<td>173,026</td>
<td>183,092</td>
<td>5011</td>
<td>508</td>
<td>365,258</td>
<td>175,718</td>
<td>6783</td>
<td>508</td>
<td>81,037</td>
<td>160,683</td>
<td></td>
</tr>
<tr>
<td>Liang</td>
<td>224.6</td>
<td>90,506</td>
<td>51,538</td>
<td>2464</td>
<td>233.1</td>
<td>180,419</td>
<td>79,899</td>
<td>3161</td>
<td>236</td>
<td>637,784</td>
<td>676,941</td>
<td></td>
</tr>
<tr>
<td>Picrocioni</td>
<td>43</td>
<td>58,042</td>
<td>42,536</td>
<td>1475</td>
<td>42.9</td>
<td>74,125</td>
<td>72,918</td>
<td>2090</td>
<td>43</td>
<td>33,333</td>
<td>36,238</td>
<td></td>
</tr>
<tr>
<td>Rastrigin</td>
<td>49</td>
<td>11,340</td>
<td>14,812</td>
<td>741</td>
<td>49</td>
<td>22,233</td>
<td>17,063</td>
<td>1705</td>
<td>49</td>
<td>1730</td>
<td>2833</td>
<td></td>
</tr>
<tr>
<td>Voglis</td>
<td>60.8</td>
<td>16,938</td>
<td>1888</td>
<td>944</td>
<td>60.5</td>
<td>35,408</td>
<td>2505</td>
<td>2304</td>
<td>61</td>
<td>21,684</td>
<td>23,126</td>
<td></td>
</tr>
<tr>
<td>Schaffer</td>
<td>93.7</td>
<td>48,401</td>
<td>23,295</td>
<td>655</td>
<td>94.8</td>
<td>56,722</td>
<td>73,922</td>
<td>1811</td>
<td>94.5</td>
<td>22,370</td>
<td>24,682</td>
<td></td>
</tr>
<tr>
<td>Shubert</td>
<td>399.6</td>
<td>890,899</td>
<td>2594</td>
<td>2297</td>
<td>398.2</td>
<td>1,062,260</td>
<td>10,732</td>
<td>10,475</td>
<td>400</td>
<td>16,551</td>
<td>36,065</td>
<td></td>
</tr>
<tr>
<td>MD</td>
<td>65.1</td>
<td>53,817</td>
<td>35,311</td>
<td>1654</td>
<td>64.5</td>
<td>85,266</td>
<td>87,221</td>
<td>2741</td>
<td>64</td>
<td>14,667</td>
<td>15,005</td>
<td></td>
</tr>
<tr>
<td>M3</td>
<td>25.8</td>
<td>30,601</td>
<td>20,295</td>
<td>1507</td>
<td>25.8</td>
<td>47,188</td>
<td>33,872</td>
<td>2184</td>
<td>23.7</td>
<td>8914</td>
<td>11,285</td>
<td></td>
</tr>
<tr>
<td>Borne</td>
<td>595.6</td>
<td>1,090,974</td>
<td>322,968</td>
<td>11,324</td>
<td>593.4</td>
<td>3,821,280</td>
<td>3,343,620</td>
<td>15,922</td>
<td>598.2</td>
<td>1,916,295</td>
<td>2,138,766</td>
<td></td>
</tr>
<tr>
<td>Rast(N = 5)</td>
<td>243</td>
<td>131,646</td>
<td>36,084</td>
<td>662</td>
<td>243</td>
<td>399,909</td>
<td>111,467</td>
<td>2011</td>
<td>243</td>
<td>59,298</td>
<td>64,349</td>
<td></td>
</tr>
<tr>
<td>Griew(N = 5)</td>
<td>160.1</td>
<td>1,859,878</td>
<td>1,619,266</td>
<td>27,717</td>
<td>159.6</td>
<td>2,154,055</td>
<td>2,023,821</td>
<td>32,101</td>
<td>170</td>
<td>2,074,573</td>
<td>2,193,769</td>
<td></td>
</tr>
<tr>
<td>Griew(N = 10)</td>
<td>11.1</td>
<td>86,815</td>
<td>78,843</td>
<td>1277</td>
<td>11.4</td>
<td>145,552</td>
<td>125,187</td>
<td>2141</td>
<td>10.4</td>
<td>85,454</td>
<td>85,209</td>
<td></td>
</tr>
<tr>
<td>Shub(N = 5)</td>
<td>32</td>
<td>12,221</td>
<td>15,622</td>
<td>508</td>
<td>32</td>
<td>17,881</td>
<td>19,822</td>
<td>811</td>
<td>32</td>
<td>6136</td>
<td>65,20</td>
<td></td>
</tr>
<tr>
<td>Shub(N = 10)</td>
<td>1021.2</td>
<td>1,779,357</td>
<td>220,435</td>
<td>3977</td>
<td>1002.1</td>
<td>1,866,619</td>
<td>1,973,621</td>
<td>33,230</td>
<td>1024</td>
<td>563,927</td>
<td>565,624</td>
<td></td>
</tr>
</tbody>
</table>

a 243 minima in [-0.5, 0.5]5.
b 171 minima in [-5, 5]5.
c 13 minima in [-3, 3]5.
d 32 minima in [-1, 1]5.
e 1024 minima in [-1.1]5.
Lagaris and Tsoulos [36] report a comparison among five stopping rules. From their results it can be seen that Multistart favors the Expected minimizers [36] rule with the Observables and Double-box criteria following closely. We conducted experiments using the criteria of Boender and Kan [37], the Observables and the Expected minimizers. Similar behavior is observed as it can be deduced by inspecting the results displayed in Table 2. Although the Double-Box is not the best performer, it is fairly easy to implement and has negligible computational overhead.

4. A parallel scheme

A sample Master–Slave parallel implementation is displayed below. The Master CPU creates candidate start points. The Slave CPUs perform local searches. Note, that since our method uses one point per iteration, each search is independent, enabling so maximum utilization of the Slave CPUs. On the other hand, most clustering methods use a collection of points, as for example in [23,24,34], that in turn create dependencies in the application of the local searches, a fact that makes the parallelization less profitable.

**Definitions.**

- **M-list:** A list that holds the minimizers (managed by the Master CPU);
- **S-list:** A list of possible starting points (managed by the Master CPU);
- **L-list:** There is one such list for every Slave CPU. Each contains the minimizers discovered by the corresponding CPU.

**Master CPU:**

1. Check if a stopping rule applies. If so terminate.
2. Take in account the updated minimizers list (M-list).
3. Create candidate start points and add them to the starting list (S-list) and assign to each one a zero flag.

**Slave CPUs:**

1. If no zero flag start-points exist in the S-list, wait.
2. Pick from the S-list a start-point with zero flag, change its flag to one, and apply a local search.
3. Add the minimizer to a temporary local minimizer list (L-list).

**Updater CPU:**

1. Pick a minimizer from the L-list and check if it is a new minimizer and remove it from the list.
2. If so, add it to the M-list.

<table>
<thead>
<tr>
<th>Function</th>
<th>Boender and Kan</th>
<th>Observables</th>
<th>Expected Minimizers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min.</td>
<td>FC</td>
<td>GC</td>
<td>LS</td>
</tr>
<tr>
<td>Ackley</td>
<td>121</td>
<td>11,245</td>
<td>6539</td>
</tr>
<tr>
<td>Bird</td>
<td>169</td>
<td>84,260</td>
<td>84,659</td>
</tr>
<tr>
<td>Bohachefsky</td>
<td>22</td>
<td>26,213</td>
<td>34,576</td>
</tr>
<tr>
<td>Giunta</td>
<td>193</td>
<td>14,077</td>
<td>27,661</td>
</tr>
<tr>
<td>Griewank</td>
<td>527</td>
<td>273,304</td>
<td>36,644</td>
</tr>
<tr>
<td>Guillin Hills</td>
<td>25</td>
<td>23,790</td>
<td>45,749</td>
</tr>
<tr>
<td>Holder</td>
<td>85</td>
<td>22,165</td>
<td>23,301</td>
</tr>
<tr>
<td>Langermann</td>
<td>263</td>
<td>120,996</td>
<td>109,319</td>
</tr>
<tr>
<td>Levy No3</td>
<td>521</td>
<td>215,190</td>
<td>204,894</td>
</tr>
<tr>
<td>Levy No5</td>
<td>508</td>
<td>115,365</td>
<td>118,926</td>
</tr>
<tr>
<td>Liang</td>
<td>234</td>
<td>99,512</td>
<td>60,608</td>
</tr>
<tr>
<td>Piccioni</td>
<td>43</td>
<td>65,782</td>
<td>61,847</td>
</tr>
<tr>
<td>Rastrigin</td>
<td>49</td>
<td>28,537</td>
<td>12,238</td>
</tr>
<tr>
<td>Voglis</td>
<td>59</td>
<td>26,614</td>
<td>1136</td>
</tr>
<tr>
<td>Schaffer</td>
<td>95</td>
<td>23,812</td>
<td>24,878</td>
</tr>
<tr>
<td>Shubert</td>
<td>400</td>
<td>270,985</td>
<td>17,194</td>
</tr>
<tr>
<td>M0</td>
<td>66</td>
<td>26,782</td>
<td>23,763</td>
</tr>
<tr>
<td>M5</td>
<td>19</td>
<td>9573</td>
<td>24,757</td>
</tr>
<tr>
<td>Borne</td>
<td>598</td>
<td>2,685,857</td>
<td>3,312,841</td>
</tr>
<tr>
<td>Rastr(N = 5)</td>
<td>238</td>
<td>44,294</td>
<td>49,774</td>
</tr>
<tr>
<td>Griew(N = 5)</td>
<td>166</td>
<td>2,654,250</td>
<td>2,776,645</td>
</tr>
<tr>
<td>Griew(N = 10)</td>
<td>11</td>
<td>101,772</td>
<td>164,941</td>
</tr>
<tr>
<td>Shub(N = 5)</td>
<td>29</td>
<td>9288</td>
<td>12,639</td>
</tr>
<tr>
<td>Shub(N = 10)</td>
<td>1021</td>
<td>557,748</td>
<td>754,471</td>
</tr>
</tbody>
</table>
5. Conclusions and further work

The adaptive character of the method enables a reasonably accurate estimate of the probability that a point belongs to a region of attraction. This in turn, on one hand saves a large fraction of local search applications, and on the other hand prevents the systematic overlook of regions of attraction, reducing therefore the risk of losing minima. The method is robust and efficient as has been deduced from the results of the computational experiments. Most of the stochastic global optimization approaches use a population of points to proceed and thus the population size is an additional parameter that affects the performance of the method. The present work in contrast, uses a single point per iteration without any adjustable parameters. This feature adds another (obvious) advantage in the case where the parallel implementation is of interest.

A parallel algorithm that would benefit from a cluster of tightly coupled processors or from a parallel shared memory system would be significant development. Such systems are nowadays widely available and offer the possibility of solving harder problems. Work in this direction is underway.

Other models for the probability, such as adaptively grown Gaussian mixtures may be considered and some early preliminary results are promising.

Appendix A. Test functions

A.1. Ackley’s test function [40]

The number of existing minima in $[-5,5]^2$ is 121

$$f(x) = -20 e^{-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2}} - e^{\frac{1}{n} \sum_{i=1}^{n} \cos(\pi x_i)} + 20 + e.$$  

A.2. Bird’s test function [41]

This function has 173 minima in $[-50,50]^2$

$$f(x_1, x_2) = \sin(x_1) \ e^{(1-\cos(x_2))^2} + \cos(x_2) \ e^{(1-\sin(x_1))^2} + (x(1) - x(2))^2.$$  

A.3. Bohachevsky’s test function [42]

This function has 25 minima in $[-10,10]^2$

$$f(x_1, x_2) = x_1^2 + 2x_2^2 - 0.3 \ \cos(3\pi x_1) - 0.4 \ \cos(4\pi x_2) + 0.7.$$  

A.4. Giunta’s test function [44]

This test function has 196 minima inside $[-20,20]^2$

$$f(x_1, x_2) = 0.6 + \sin y_1 + \sin^2 y_1 + \frac{1}{50} \ \sin 4y_1 + \sin y_2 + \sin^2 y_2 + \frac{1}{50} \ \sin 4y_2,$$

where $y_1 = \frac{10}{x_1} x_1 - 1$ and $y_2 = \frac{10}{x_2} x_2 - 1$. where $y_1 = \frac{10}{x_1} x_1 - 1$ and $y_2 = \frac{10}{x_2} x_2 - 1$.

A.5. Griewank’s test function [45]

This function has 529 minima inside $[-100,100]^2$

$$f(x) = \frac{1}{200} \sum_{i=0}^{n} x_i^2 - \prod_{i=1}^{n} \cos \frac{x_i}{\sqrt{i}} + 1.$$  

A.6. Guillin Hills’s test function [34]

This test function possesses 25 minima inside $[0,1]^2$

$$f(x) = 3 + \sum_{i=1}^{n} \frac{c_i (x_i + 9)}{x_i + 10} \ \sin \left( \frac{\pi}{1 - x_i + \frac{1}{2k}} \right),$$

where $c_i = 2$, $i = 1, \ldots, n$ and $k = 5$.

A.7. Holder test function [41]

This function has 85 minima inside $[-20,20]^2$

$$f(x_1, x_2) = -\cos x_1 \cos x_2 e^{-\sqrt{x_1^2 + x_2^2}}.$$  

A.8. Langermann's test function [46]

This test function has 270 minima inside \([0, 7]^2\)

\[
f(x) = \sum_{k=0}^{5} c_k e^{\sigma_k} \cos \lambda_k.
\]

In current implementation \(a = (3.5, 2, 1.7)^T\), \(c = (1.2, 5, 2.3)^T\) where \(\sigma_k = \sum_{i=1}^{n} \frac{(x_i - a_k)^2}{\pi}\) and \(\lambda_k = \sum_{i=1}^{n} \pi(x_i - a_k)^2\).

A.9. Levy's 3rd test function [48]

This test function has 527 minima inside \([-10, 10]^2\)

\[
f(x_1, x_2) = \sum_{k=1}^{5} k \cos ((k-1)x_1 + k) \sum_{k=1}^{5} k \cos ((k+1)x_2 + k).
\]

A.10. Levy's 5th test function [48]

This test function has 508 minima inside \([-10, 10]^2\)

\[
f(x_1, x_2) = f_{levy3}(x_1, x_2) + (x_1 + 1.42513)^2 + (x_2 + 0.80032)^2.
\]

A.11. Liang's test function [49]

This test function has 236 local minima inside \([1, 4]^2\)

\[
f(x_1, x_2) = -(x_1 \sin(20x_2) + x_2 \sin(20x_1))^2 \cosh(\sin(10x_1)x_1) - (x_1 \cos(20x_2) - x_2 \sin(10x_1))^2 \cosh(\cos(10x_2)x_2).
\]

A.12. Piccioni's test function [50]

This test function has 28 minima inside \([-5, 5]^2\)

\[
f(x) = -10 \sin(\pi x_1)^2 - \sum_{i=1}^{n-1} (x_i - 1)^2(1 + 10 \sin(\pi x_{i+1})) - (x_n - 1)^2.
\]

A.13. Rastrigin's test function [51]

This test function has 49 minima inside \([-1, 1]^2\)

\[
f(x) = 10n + \sum_{i=1}^{n} (x_i^2 - 10 \cos(2\pi x_i)).
\]

A.14. Voglis's test function

This test function has 61 minima inside \([-25, 25]^2\)

\[
f(x) = \alpha_0 \left( \frac{1}{2} x^T Q_0 x + x^T d_0 \right) + \sum_{i=1}^{80} \alpha_i e^{-x^T Q_i x + x^T d_i}.
\]

Function dimension \(n = 2\), \(Q_j\) specific positive definite \(2 \times 2\) matrices, \(d_j\) 2-dimensional vectors and \(\alpha_j\) appropriate scaling constants.

A.15. Schaffer's test function [41]

This test function has 95 minima inside \([-3, 3]^2\)

\[
f(x_1, x_2) = 0.5 + \frac{\sin(x_1^2 + x_2^2)^2 - 0.5}{(1 + 0.001(x_1^2 + x_2^2))^2} + 0.1 \sin(10x_1) + 0.1 \sin(10x_2).
\]
A.16. Shubert’s test function [52]

This test function has 400 minima inside \([-10, 10]^2\)

\[
f(x) = -\sum_{i=1}^{n} \sum_{j=1}^{5} f(0.1x_i + j).
\]

A.17. M0 test function [52]

This test function has 66 minima inside \([-5, 1]^2\)

\[
f(x) = \sin \left(2.2\pi x_1 + \frac{\pi}{2}\right) \frac{2 - x_2}{2} - \frac{3 - x_3}{2} + \sin \left(\frac{\pi}{2} x_2 + \frac{\pi}{2}\right) \frac{2 - x_3}{2} - \frac{3 - x_1}{2}.
\]

A.18. M3 test function [52]

This test function has 26 minima inside \([-2, 2]^2\)

\[
f(x) = -(x_2^2 - 4.5x_2)x_1x_2 - 4.7 \cos(3x_1 - x_3^2(2 + x_1)) \sin(2.5\pi x_1) + (0.3 \times x_1)^2.
\]

A.19. Siam Problem 4 Function [53]

This test function has 600 minima inside \([-1, 1]^2\)

\[
f(x) = \exp(\sin(x_1)) + \sin(60 \exp(x_2)) + \sin(70 \sin(x_1)) + \sin(\sin(80x_2)) - \sin(10(x_1 + x_2)) + \frac{x_1^2 + x_2^2}{4}.
\]

References


