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

C. Voglis*and I. E. Lagaris
Department of Computer Science, University of Ioannina
P.O.Box 1186, Ioannina 45110 – GREECE

Abstract

A stochastic global optimization method based on *Multistart* is presented. In this, the local search is conditionally applied with a probability that takes in 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. Results of its application on a set of common test functions are reported, along with a performance comparison against other established methods of similar nature.

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, 3, 4], the management of mutual funds [5, 6, 7, 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:

$$\text{Find all } x_i^* \in S \subset R^n \text{ that satisfy:}$$
$$x_i^* = \arg\min_{x \in S_i} f(x), \quad S_i = S \cap \{x, ||x - x_i^*|| < \epsilon\} \tag{1}$$

S is considered to be a bounded domain of finite measure and ϵ a positive infinitesimally small number. We are adopting the stochastic class of methods. One of the most widely

*Corresponding author. Email: voglis@cs.uoi.gr

used stochastic algorithms is the so called *Multistart* [18]. It's popularity stems from it's simplicity and inherent parallelization [19, 20, 21, 22]. Many stochastic methods have been developed around it starting from the classic papers of [18, 23, 24, 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, 24, 25, 34, 20].

The *Multistart* algorithm is presented bellow:

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: $y = \mathcal{L}(x)$
If ($y \notin \{y_i, i = 1, 2, \ldots, k\}$) Then
$k = k + 1$
$y_k = y$
Endif

Iterate: Go back to the Termination Control step.

The **"region of attraction"** of a local minimum associated with a local search procedure $\mathcal{L}$ is defined as:

$$A_i \equiv \{x \in S, \mathcal{L}(x) = x_i^*\} \tag{2}$$

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

$$\cup_{i=1}^{w} A_i = S \tag{3}$$

Let $m(A)$ indicate the *Lebesgue measure* of $A \subseteq R^n$. If we assume a deterministic search $\mathcal{L}$, 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:

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

If a point in S is sampled from a uniform distribution, the apriori 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 apriori probability that at least one point is contained in A_i is given by:

$$1 - (1 - \frac{m(A_i)}{m(S)})^K = 1 - (1 - p_i)^K \qquad (5)$$

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 ($A_1 \bigcup A_2$). 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 ($x \notin \cup_{i=1}^{k} A_i$) Then
$y = \mathcal{L}(x)$
$k = k + 1$
$y_k = y$
Endif

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 $f(.)$ 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 $x \notin \cup_{i=1}^{k} A_i$
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

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 *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 *maximum attractive radius (MAR)* as:

$$R_i = \max_j \{||x_j^{(i)} - y_i||\} \tag{6}$$

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

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 $z \equiv ||y - x||/R$, then an estimate of the probability that $x \notin A(y)$ may be given by:

$$p(x \notin A(y)) = \begin{cases} 1, \text{ if } z > 1 \text{ or } \nabla f(x)^T (y - x) \geq 0 \\ \phi(z, l) * \left[1 + \frac{(y-x)^T \nabla f(x)}{||y-x|| |\nabla f(x)|}\right], \text{otherwise} \end{cases} \tag{7}$$

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

$$\begin{aligned} &\lim_{z \to 0} \phi(z, l) \to 0 \\ &\lim_{z \to 1} \phi(z, l) \to 1 \\ &\lim_{l \to \infty} \phi(z, l) \to 0 \\ &0 < \phi(z, l) < 1 \end{aligned} \tag{8}$$

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 $x \notin \cup_{i=1}^{k} A_i$ is given by the product $\prod_{i=1}^{k} p(x \notin A_i)$ and may now be approximated by the probability that $x \notin A_n$, A_n being the region of attraction of the nearest to x discovered minimizer y_n. The rationale for this approximation is that if $x \notin B(y_i, R_i)\ \forall i \neq n$, 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

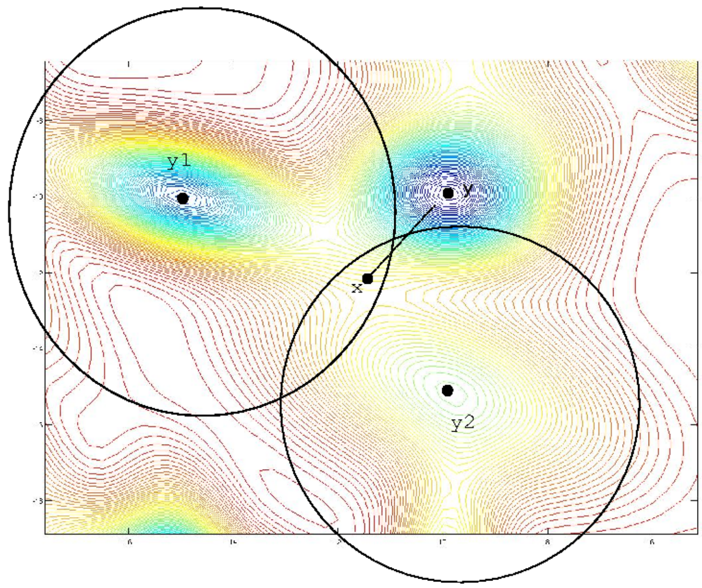


Figure 1: A point x that would lead to a new minimum y, is inside the overlap region of the spheres around two recovered minima y_1 and y_2

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 closeby 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 bellow:

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

$p_k = -\frac{1}{||\nabla f(x_k)||}B_k^{-1}\nabla f(x_k)$

End if

Step 2 (**Line search**):

$min_a(f(x_k + \alpha p_k))$, yielding a_k

Step 3 (**Next iterate**):

$x_{k+1} = x_k + \alpha_k p_k$

Step 4 (**Update approximation**):

$\gamma_k = \nabla f(x_{k+1}) - \nabla f(x_k)$

$\delta_k = x_{k+1} - x_k$

$B_{k+1} = bfgs\_update(B_k, \gamma_k, \delta_k)$

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 figs. 2(a)-2(d). The single minimum appearing in fig. 2(d) is the first minimum in fig. 2(b). Note that in fig. 2(c) the line search ends up to the nearest minimum while that of fig. 2(a) 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.

2.4 Asymptotic guaranty

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

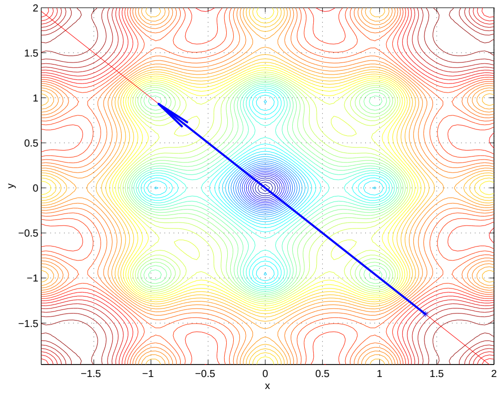
$$p_y^{(i)} = \int_{x \in A(y)} p_{LS}^{(i)}(x)\frac{dx}{|S|} \tag{9}$$

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 *MAR's* etc. The probability that after k trials y is not found is then given by:

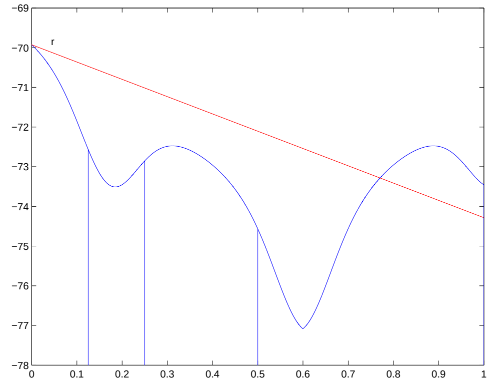
$$\pi_y^{(k)} = \prod_{i=1}^{k}(1 - p_y^{(i)}) \leq \left(1 - \min_i\{p_y^{(i)}\}\right)^k \tag{10}$$

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

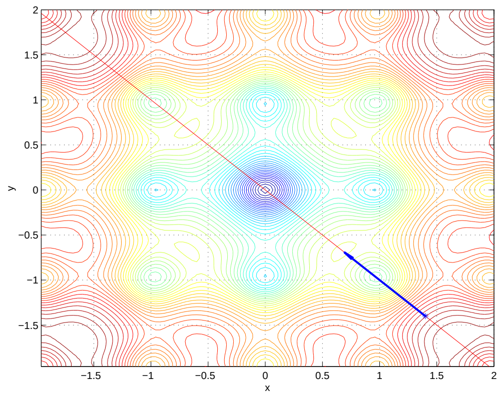
$$p_y^{(i)} = \int_{x \in A_1(y)} p_{LS}^{(i)}(x)\frac{dx}{|S|} + \int_{x \in A_2(y)} p_{LS}^{(i)}(x)\frac{dx}{|S|} \tag{11}$$



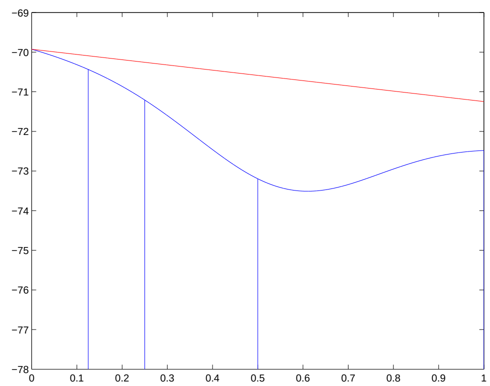
(a) Ackley's function contour plot and search step with $a_0 = 1$.



(b) Function profile along with the back-tracking points



(c) Same as (a) but using $a_0 = \frac{1}{||\nabla f||}$



(d) Same as (b) for the normalized direction

Figure 2: Illustration of the modified line search

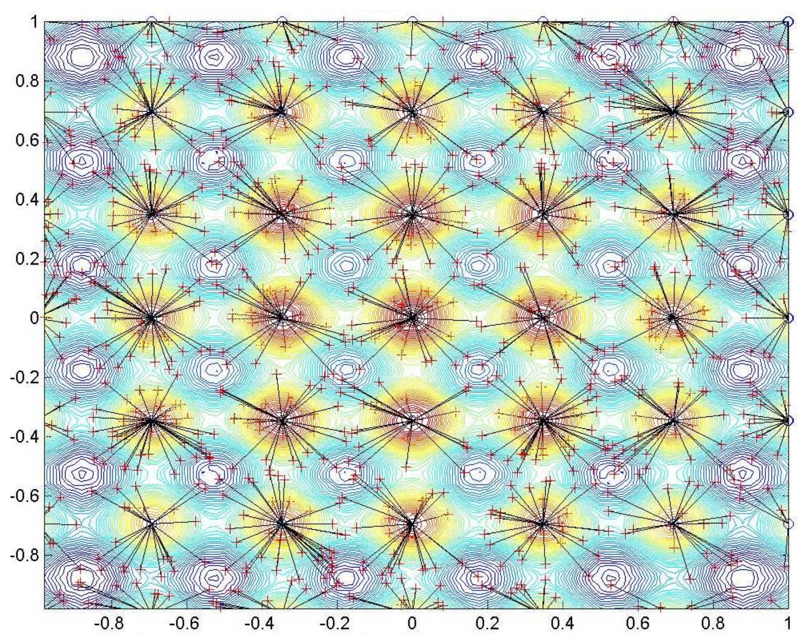


Figure 3: A suitable local search, with contiguous regions of attraction

where

$$
\begin{aligned}
A_1(y) &= \{x \in A(y); (y_c - x)^T \nabla f(x) \leq 0\} \\
A_2(y) &= \{x \in A(y); (y_c - x)^T \nabla f(x) > 0\}
\end{aligned}
$$

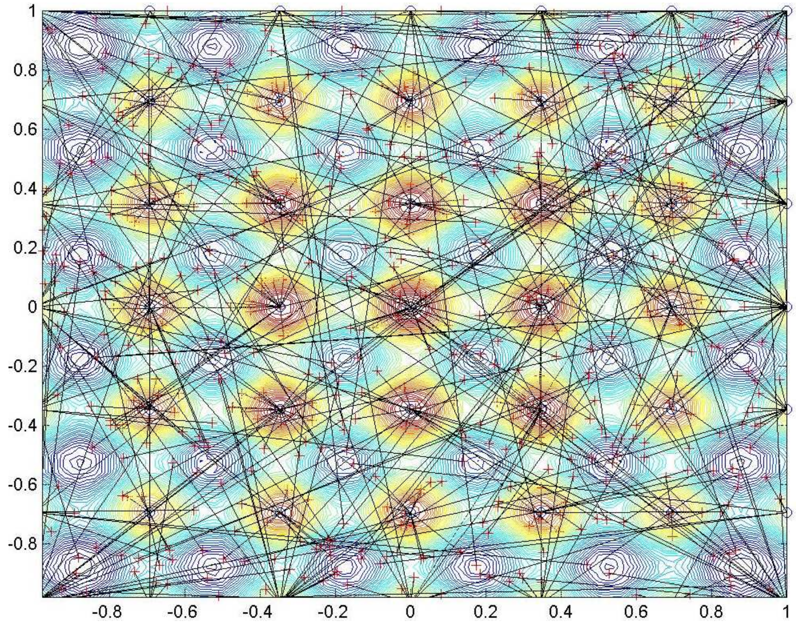


Figure 4: An improper local search, with disjoint regions of attraction

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)

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

At the limit as $k \to \infty$ we deduce from above and eq. (10) that $\pi_y^{(k)} \to 0$, 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.

$$\phi(z, l) = ze^{-l^2(z-1)^2}, \ \forall z \in (0, 1) \tag{12}$$

A graphical representation is depicted in fig. (2.5).

2.6 The ADAPT Algorithm

The proposed algorithm, in summary, is presented below:

ADAPT Algorithm
Input:
The input function $f : R^n \longrightarrow R$ The search domain $S \subseteq R^n$ A local search procedure $\mathcal{L}(x)$ having the properties described in Section 2.3.

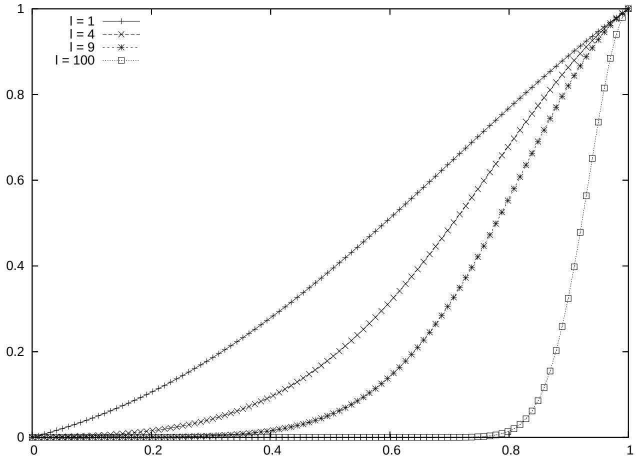


Figure 5: Model plots for several l values

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: Sample $x \in S$

Main step: i = $\underset{j=1,\dots,k}{argmin}||x - y_j||$
$d = ||x - y_i||$
If $(d < r_i)$ Then
  If $(\nabla f(x)^T (y_i - x) < 0)$ Then
    $z = \frac{||y_i - x||}{r_i}$
    $p = \phi(z, n_i)\left[1 + \frac{(y_i - x)^T \nabla f(x)}{||(y_i - x)^T \nabla f(x)||}\right]$
  Else
    $p = 1.0$
  Endif
Else
  $p = 1.0$
Endif
Let ξ be a uniform random in $[0, 1]$
If $(\xi < p)$ Then
  $y = \mathcal{L}(x)$
  If (y 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 M.J.D. 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 M0, 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).

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* [36] 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).

Function	TML				Multistart				MinFinder				Adapt	
	Min.	FC	GC	LS	Min.	FC	GC	L.S	Min.	FC	GC	L.S	Min.	FC	GC
Ackley	121	10259	14457	1207	121	23281	36543	2054	121	7510	11926	208	121	7340	4600
Bird	158.5	84798	103889	2507	141.5	212196	150529	3737	172.8	122639	145460	1832	171.7	56008	55296
Bohachefsky	25	139190	125684	2369	25	241501	187175	2547	18.7	25907	32243	538	24.3	18332	23112
Giunta	196	104812	16606	719	196	45688	67311	1212	196	18753	20972	791	196	10211	17821
Grienwank	527.2	1883423	1461617	39090	526.4	1912452	1892111	38727	529	1133908	1284982	30577	528.5	231123	27811
Guillin Hills	25	81153	69847	2451	24.9	87411	76563	2617	24.8	22901	23570	820	24.7	17751	31811
Holder	85	28749	23346	622	85	69038	34468	988	85	8289	8977	261	85	16788	16461
Langermann	257	129521	124360	3566	260	185669	111478	4169	270	503470	500675	19123	270	80578	80386
Levy No3	527	170541	171643	6999	527	494578	277868	8909	527	59830	91479	2320	527	146574	179502
Levy No5	508	173026	183092	5011	508	365258	175718	6783	508	81037	160683	2733	508	84152	83462
Liang	224.6	90506	51538	2464	233.1	180419	79899	3161	236	637784	676941	22607	235.8	73215	50569
Piccionni	43	58042	42536	1475	42.9	74125	72918	2090	43	33333	36238	1222	43	48123	45647
Rastrigin	49	11340	14812	741	49	22233	17063	1705	49	1730	2833	85	49	17810	7481
Voglis	60.8	16938	1888	944	60.5	35408	2505	2304	61	21684	23126	694	61	16932	1267
Schaffer	93.7	48401	23295	865	94.8	56722	73922	1811	94.5	22370	24682	876	94.7	18922	16779
Shubert	399.6	890899	2594	2297	398.2	1062260	10732	10475	400	16551	36065	665	400	193211	10780
M0	65.1	53817	35311	1654	64.5	85266	87221	2741	64	14667	16005	799	65.7	16659	17033
M3	25.8	30601	20295	1507	25.8	47188	33872	2184	23.7	8914	11285	790	25.6	8752	17168
Borne	595.6	1090974	322968	11324	593.4	3821280	3343620	15922	598.2	1916295	2138766	68865	598.2	1880314	2626185
Rast(N=5)[a]	243	131646	36084	662	243	399909	111467	2011	243	59298	64349	1022	243	30350	37634
Griew(N=5)[b]	160.1	1859878	1619266	27717	159.6	2154055	2023821	32101	170	2074573	2193769	32710	169.8	1833628	2052681
Griew(N=10)[c]	11.1	86815	78843	1277	11.4	145552	125187	2141	10.4	85454	85209	1270	12.7	76221	121176
Shub(N=5)[d]	32	12221	15622	508	32	17881	19822	811	32	6136	6520	158	32	7022	8112
Shub(N=10)[e]	1021.2	1779357	220435	3977	1002.1	1866619	1973621	33230	1024	563927	565624	10302	1024	406503	526229

[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 $[-1, 1]^1 0$

Table 1: Method comparison using the Double-Box rule [36]

Function	Boender and Kan				Observables				Expected Minimizers			
	Min.	FC	GC	LS	Min.	FC	GC	L.S	Min.	FC	GC	L.S
Ackley	121	11245	6539	759	121	4167	3630	367	121	4265	2891	326
Bird	169	84260	84659	2021	170	35551	36867	985	171	33165	31874	866
Bohachefsky	22	26213	34576	318	24	12815	16532	150	24	11615	14865	137
Giunta	193	14077	27661	1077	194	7156	11693	521	195	7829	10953	462
Grienwank	527	273304	36644	21407	528	121838	15231	10498	529	136513	22405	9179
Guillin Hills	25	23790	45749	969	24	12235	19661	469	23	10713	18995	415
Holder	85	22165	23301	504	85	11022	11525	240	81	10014	10368	216
Langermann	263	120996	109319	3397	268	54378	52878	1660	269	45700	47968	1456
Levy No3	521	215190	204894	7478	524	97859	121728	3661	524	90826	108808	3206
Levy No5	508	115365	118926	3622	508	56203	53068	1770	508	46168	49922	1552
Liang	234	99512	60608	3206	235	53217	35288	1568	236	45557	29356	1375
Piccioni	43	65782	61847	1367	42	31060	32534	666	42	29770	27678	587
Rastrigin	49	28537	12238	214	49	12873	5345	97	49	11901	4920	91
Voglis	59	26614	1136	621	60	12833	850	298	61	10620	510	265
Schaffer	95	23812	24878	982	95	13096	9704	476	95	12128	9537	420
Shubert	400	270985	17194	1982	400	132697	9041	966	400	115116	5153	851
M0	66	26782	23763	1417	66	10261	11503	689	66	10160	10899	608
M5	19	9573	24757	992	23	7286	10728	480	24	4660	9226	425
Borne	598	2685857	3312841	95370	597	1152416	1678605	46790	598	1033720	1576821	40893
Rast(N=5)	238	44294	49774	1676	240	19100	27478	816	240	14214	24037	720
Griew(N=5)	166	2654250	2776645	47468	169	1182435	1329089	23286	170	1026242	1170489	20355
Griew(N=10)	11	101772	164941	2449	13	52465	82684	1194	13	42192	73400	1050
Shub(N=5)	29	9288	12639	304	31	5379	5627	143	29	4573	5526	131
Shub(N=10)	1021	557748	754471	17493	1023	279352	372875	8577	1024	254876	310294	7501

Table 2: Adapt results using different stopping rules

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.

---

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 loosing 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.

References

[1] Pardalos Panos M., Romeijn Edwin H., Tuy Hoang (2000), Recent developments and trends in global optimization, Journal of Computational and Applied Mathematics **124**, 209–228

[2] Maranas, C. D., and Floudas, C.A. (1994), Global minimum potential energy conformations of small molecules, Journal of Global Optimization, **4**, 135-170.

[3] Wales, J. D. and Scheraga, H. A. (1999), Global Optimization of Clusters, Crystals, and Biomolecules, Science, **285**, 1368–1372

[4] Saunders, M (1987), Stochastic exploration of molecular mechanics energy surfaces. Hunting for the global minimum, Journal of the American Chemical Society, **109**, 3150–3152.

[5] Ziemba, W. T. and Vickson, R. G. (2006), *Stochastic Optimization Models in Finance*, World Scientific.

[6] Rockafellar R. T and Uryasev S. (2000), Optimization of Conditional value-at-risk, Journal of Risk, **2**, 21–41.

[7] Bertsimas, D., Darnell, C. and Soucy, R. (1998), Portfolio construction through mixed integer programming, Sloan Working Papers, Sloan School of Management, Massachusetts Institute of Technology.

[8] Iglehart, D. L., Voessner, S. (1998), Optimization of a trading system using global search techniques and local optimization, Journal of Computational Intelligence in Finance.

[9] Floudas, C.A. and Pardalos, P.M. (1990), *A Collection of Test Problems for Constrained Global Optimization Algorithms*, Springer-Verlag New York, Inc.

[10] Himmelblau, D.M. (1972), *Applied Nonlinear Programming*, McGraw-Hill.

[11] Floudas, C.A and Pardalos, P.M. (2000), *Optimization in Computational Chemistry and Molecular Biology: Local and Global Approaches*, Kluwer Academic Publishers.

[12] Waterbeemd, H.D., Smith, D.A., Beaumont K., and Walker D.K. (2001), Property-based design: optimization of drug absorption and pharmacokinetics, Journal of Medical Chemistry, **44**, 1313–1333.

[13] Floudas, C.A. (1999), *Deterministic Global Optimization: Theory, Methods and Applications*, Springer.

[14] Horst, R. and Tuy, H. (1993), *Global Optimization: Deterministic Approaches*, Springer.

[15] Boender C.G.E. and Romeijn Edwin H. (1995), Stochastic Methods, in *Handbook of Global Optimization* (Horst, R. and Pardalos, P. M. eds.), Kluwer, Dordrecht, 829–871

[16] Liberti, Leo. and Kucherenko, S. (2005), Comparison of deterministic and stochastic approaches to global optimization, International Transactions In Operational Research, **12**, 263–285.

[17] Hansen, E. R. and Walster, G. W. (2004), *Global Optimization Using Interval Analysis*, CRC Press.

[18] Boender, C.G.E. and Rinnooy Kan, A.H.G and Timmer, G.T. and Stougie, L.(1982) , A stochastic method for global optimization, Mathematical Programming, **22**, 125–140.

[19] Ugray, Z., Lasdon, L., Plummer, John., Kelly, J. and Marti R. (2006), Scatter Search and Local NLP Solvers: A Multistart Framework for Global Optimization, McCombs Research Paper Series No. IROM-07-06 Available at SSRN: http://ssrn.com/abstract=886559.

[20] Tu, W. and Mayne, R. W. (2002), An approach to multi-start clustering for global optimization with non-linear constraints, International journal for numerical methods in engineering, **53**, 2253–2269.

[21] Maltz, J.S., Polak, E and Budinger, T.F. (1998), Multistart optimisation algorithm for joint spatial and kinetic parameter estimation in dynamic ECT, Nuclear Science Symposium, 1998. Conference Record. 1998 IEEE, **3**, 1567–1573.

[22] Dixon, L. C. W. and Jha, M.(1993), Parallel algorithms for global optimization, Journal of Optimization Theory and Applications, **79**, 385–395.

[23] Rinnooy Kan, A.H.G and Timmer, G.T.(1987), Stochastic global optimization methods. Part I: Clustering methods, Mathematical Programming, **39**, 27–56.

[24] Rinnooy Kan, A.H.G and Timmer, G.T.(1987), Stochastic global optimization methods. Part II: Multi level methods, Mathematical Programming, **39**, 57–78.

[25] Törn, A. A.(1978) A search clustering approach to global optimization, in Dixon, L.C.W and Szegö, G.P. (eds.), *Towards Global Optimization 2*, North-Holland, Amsterdam.

[26] Törn, A. and Viitanen, S. (1994) Topographical Global Optimization Using Pre–Sampled Points, Journal of Global Optimization **5**, 267–276.

[27] Ali, M.M. and Storey, C.(1994), Topographical Multilevel Single Linkage, Journal of Global Optimization **5**, 349–358.

[28] Hart, W.E. (1994), *Adaptive Global Optimization with Local Search*, Ph.D Dissertation, University of California, San Diego.

[29] Locatteli, M. (1998) Random linkage: a family of acceptance /rejection algorithms for global optimisation, Mathematical Programming **85** 2, 379–396.

[30] Schoen, F. (2002) Two-phase methods for global optimization, in *Handbook of Global Optimization* Kluwer Academic Publishers, 151–178.

[31] Locatteli, M. (2005), On the Multilevel Structure of Global Optimization Problems, Computational Optimization and Applications, **30** 1, 5–22.

[32] Liang, K.-H., Yao, X. and Newton, C. (1999), Combining Landscape Approximation and Local Search in Global Optimization, Proc. of the 1999 Congress on Evolutionary Computation, 1514–1520.

[33] Bolton, H. P. J., Schutte, J. F. and Groenwold A. A. (2000), Multiple Parallel Local Searches in Global Optimization, LNCS 1908, 88–95.

[34] Tsoulos, I. G. and Lagaris, I. E. (2006) MinFinder: Locating all the local minima of a functions, Computer Physics Communications, **174**, 166–179.

[35] Theos, F. V., Lagaris, I. E. and Papageorgiou, D. G.(2004), PANMIN: Sequential and parallel global optimization procedures with a variety of options for the local search strategy, Comput. Phys. Commun. **159**, 63–69.

[36] Lagaris, I.E. and Tsoulos, I. (2008), Stopping rules for box-constrained stochastic global optimization, Applied Mathematics and Computation **197**, 622-632.

[37] Boender C.G.E., Kan Rinnooy A.H.G. (1987), Bayesian stopping rules for multistart global optimization methods, Mathematical Programming **37**, 59–80

[38] Goldberg, D. E. (1989), *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Longman Publishing Co.

[39] Ali, M. M, Khompatraporn, C. and Zabinsky, Z. B. (2005), A Numerical Evaluation of Several Stochastic Algorithms on Selected Continuous Global Optimization Test, Journal of Global Optimization **31** 635–672.

[40] Ackley, D. H. (1987), *A connectionist machine for genetic hillclimbing*, Kluwer Academic Publishers.

[41] Mishra, S. K. (2006), Some new test functions for global optimization and performance of repulsive particle swarm method, Munich Personal RePEc Archive, Paper No. 2718.

[42] Bohachevsky, M.E., Johnson, M.E. and Stein, M.L. (1986), Generalized simulated annealing for function optimization. Technometrics **28**, 209–217.

[43] Chichinadze, V. (1969), The Ψ-transform for solving linear and nonlinear programming problems, Automata **5**, 347–355

[44] Giunta, (1997), A. A. Aircraft Multidisciplinary Design Optimization using Design of Experiments Theory and Response Surface Modeling Methods, MAD Center Report 97-05-01, Virginia Polytechnic Institute & State Univ. Blacksburg.

[45] Griewank, A. O. (1981), Generalized descent for global optimization, J. Optim. Theory Appl., 34, 11–39.

[46] GEATbx (2006), GEATbx: The Genetic and Evolutionary Algorithm Toolbox for Matlab, `http://www.geatbx.com/index.html`.

[47] Michalewicz, Z. (1992), *Genetic Algorithms + Data Structures = Evolution Programs* Berlin, Heidelberg, New York: Springer-Verlag.

[48] Levy, A.V. and Montalvo, A. (1985) The tunneling algorithm for the global minimization of functions. SIAM Journal on Scientific and Statistical Computing, **6**(1) 15–29.

[49] Liang, J.J. and Suganthan, P.N. (2005) Dynamic Multi-Swarm Particle Swarm Optimizer, International Swarm Intelligence Symposium IEEE, 124–129.

[50] Lucidi, S. and Piccioni, M. (1989), Random tunneling by means of acceptance-rejection sampling for global optimization , Journal of Optimization Theory and Applications, **62** 2, 255–277.

[51] Rastrigin, L. A. (1968), *Statistical Search Methods*, Nauka, Moscow.

[52] Shubert, B.O. (1972), A Sequential Method Seeking the Global Maximum of a Function, SIAM J. on Numer. Analysis, **9**, 379–388.

[53] `http://www.siam.org/siamnews/01-02/challenge.pdf`

[54] Powell M.J.D., (1989) TOLMIN: A Fortran package for linearly constrained C optimization calculations, DAMTP 1989/NA2, Dept. of Applied Mathematics and Theoretical Physics, Univ. of Cambridge.

6 Appendix - Test Functions

6.1 Ackley's test function ([40])

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

$$f(x) = -\alpha e^{-b\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}} - e^{\frac{1}{n}\sum_{i=1}^{n} cos(cx_i)} - \alpha e^1$$

6.2 Bird's test function ([41])

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

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

6.3 Bohachevsky 's test function ([42])

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

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

6.4 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 $y_1 = \frac{16}{15}x_1 - 1$ and $y_2 = \frac{16}{15}x_2 - 1$. where $y_1 = \frac{16}{15}x_1 - 1$ and $y_2 = \frac{16}{15}x_2 - 1$.

6.5 Griewank's test function ([45])

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

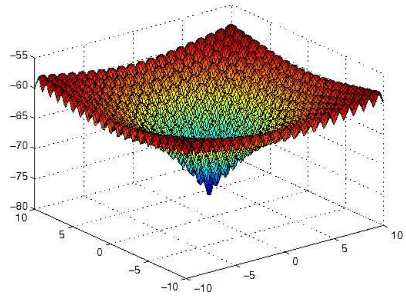
$$f(x) = \frac{1}{200}\sum_{i=0}^{n} x_i^2 - \prod_{i=1}^{n}\cos\frac{x_i}{\sqrt{i}} + 1$$

6.6 Guillin Hills's test function ([34])

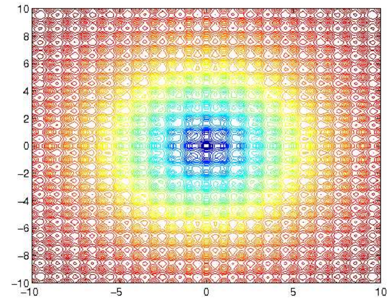
This test function possesses 25 minima inside $[0,1]^2$.

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

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

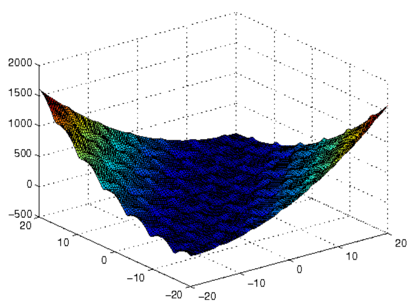


(a) Surface plot

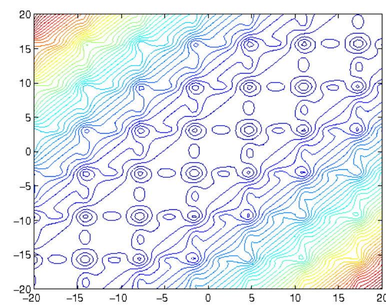


(b) Contour plot

Figure 6: Ackley's test function

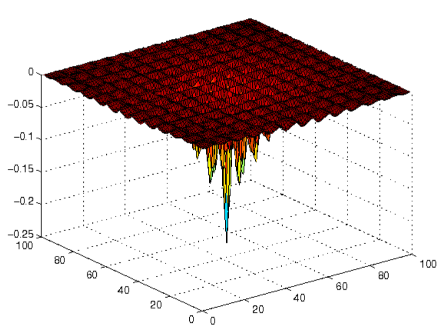


(a) Surface plot

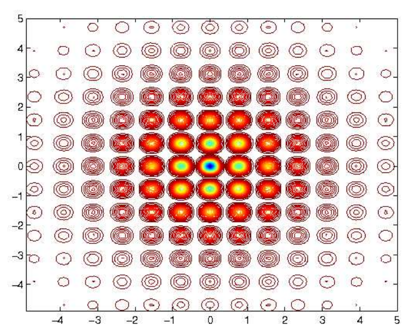


(b) Contour plot

Figure 7: Birds's test function

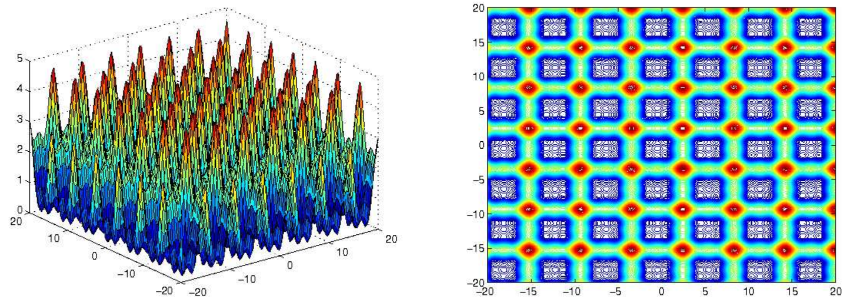


(a) Surface plot



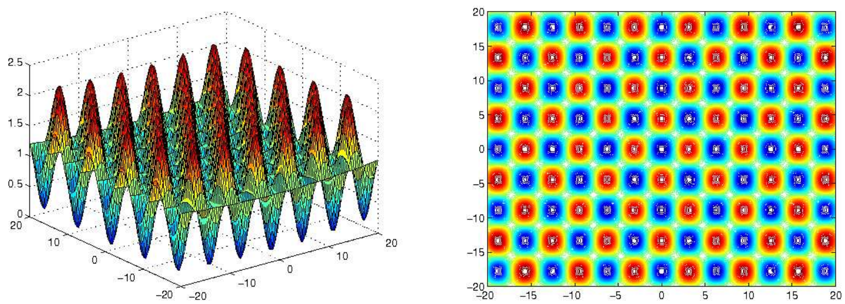
(b) Contour plot

Figure 8: Carrom table test function



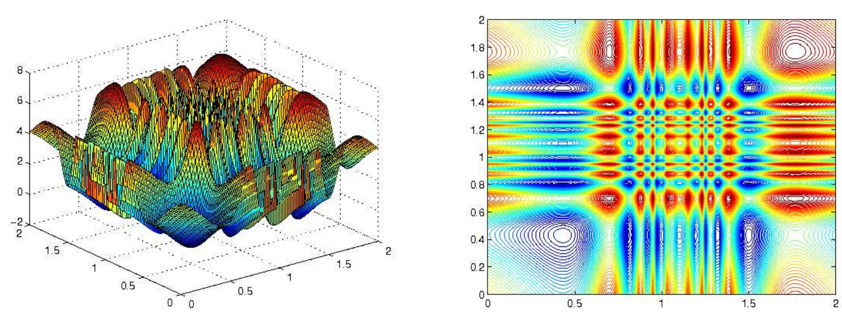
(a) Surface plot (b) Contour plot

Figure 9: Giunta's test function



(a) Surface plot (b) Contour plot

Figure 10: Griewanks's test function



(a) Surface plot (b) Contour plot

Figure 11: Guillin Hills test function

6.7 Holder test function ([41])

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

$$f(x_1, x_2) = -\cos x_1 \cos x_2 e^{1-\frac{\sqrt{x_1^2+x_2^2}}{\pi}}$$

6.8 Langermanns's test function ([46])

This test function has 270 minima inside $[0, 7]^2$.

$$f(x_i) = \sum_{k=0}^{5} c_k e^{\sigma_k} \ \cos \lambda_k$$

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 $\lambda_k = \sum_{i=1}^{n} \pi (x_i - a_k)^2$.

6.9 Levy's 3rd test function ([48])

This test function has 527 minima inside $[-10, 10]^2$.

$$f(x_1, x_2) = \sum_{k=1}^{5} k \cos\left((k-1)x_1 + k\right) \sum_{k=1}^{5} k \cos\left((k+1)x_2 + k\right)$$

6.10 Levy's 5th test function ([48])

This test function has 508 minima inside $[-10, 10]^2$.

$$f(x_1, x_2) = f_{Levy3}(x_1, x_2) + (x_1 + 1.42513)^2 + (x_2 + 0.80032)^2$$

6.11 Liang's test function [49]

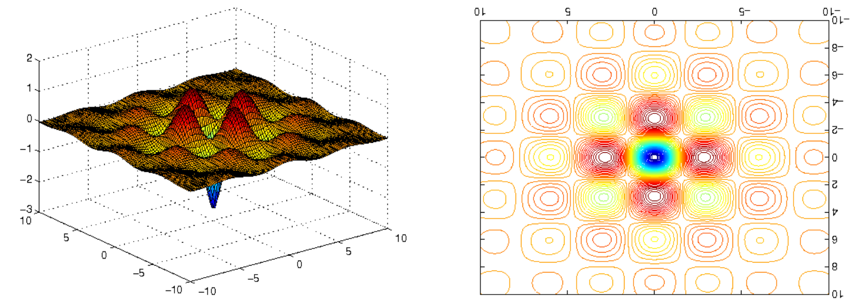
This test function has 236 local minima inside $[1, 4]^2$.

$$\begin{aligned} f(x_1, x_2) = & - \ (x_1 \sin(20x_2) + x_2 \sin(20x_1))^2 \cosh\left(\sin(10x_1)x_1\right) \\ & - \ (x_1 \cos(20x_2) - x_2 \sin(10x_1))^2 \cosh\left(\cos(10x_2)x_2\right) \end{aligned}$$

6.12 Piccioni's test function ([50])

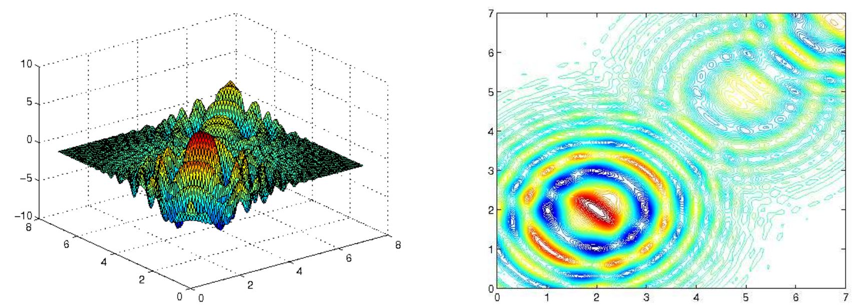
This test function has 28 minima inside $[-5, 5]^2$.

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



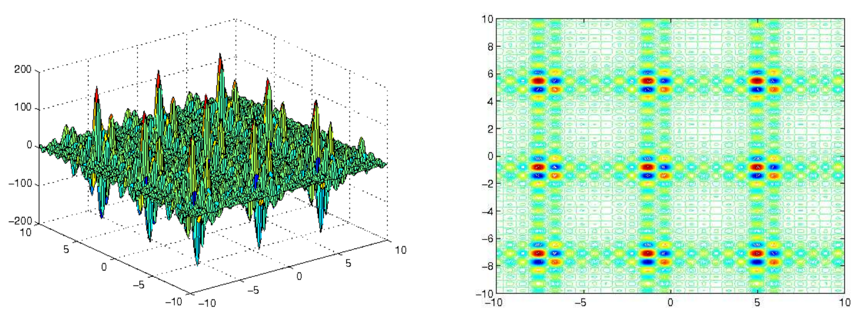
(a) Surface plot (b) Contour plot

Figure 12: Holder-like test function



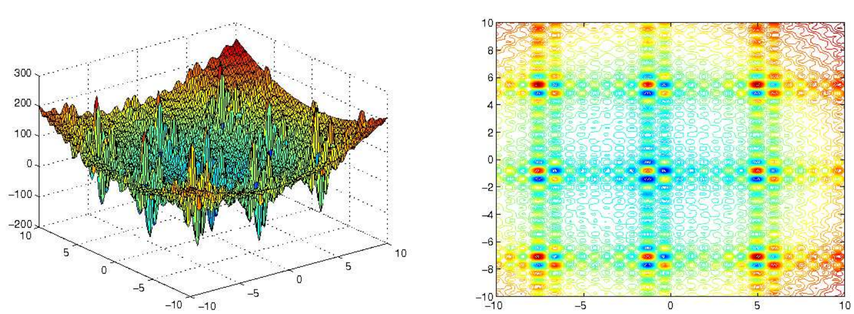
(a) Surface plot (b) Contour plot

Figure 13: Lagermanns's test function



(a) Surface plot (b) Contour plot

Figure 14: Levy's No 3 test function



(a) Surface plot (b) Contour plot

Figure 15: Levy's No 5 test function

6.13 Rastrigin's test function ([51])

This test function has 49 minima inside $[-1, 1]^2$.

$$f(x) = 10n + \sum_{i=1}^{n} \left(x_i^2 - 10\cos(2\pi x_i)\right)$$

6.14 Voglis's Test Function

This test function has 61 minima inside $[-25, 25]^2$.

$$f(x) = \alpha_0 \left(\frac{1}{2}x^T Q_0 x + x^T d_0\right) + \sum_{i=1}^{80} \alpha_k e^{-\frac{1}{2}x^T Q_k x + x^T d_k}$$

Function dimension $n = 2$, Q_j specific positive definite $2x2$ matrices, d_j 2-dimensional vectors and α_j appropriate scaling constants.

6.15 Schaffer's Test Function ([41])

This test function has 95 minima inside $[-3, 3]^2$.

$$f(x_1, x_2) = 0.5 + \frac{\sin(x_1^2 + x_2^2)^2 - 0.5}{\left(1 + 0.001(x_1^2 + x_2^2)\right)^2} + 0.1\sin(10x_1) + 0.1\sin(10x_2)$$

6.16 Shubert's Test Function ([52])

This test function has 400 minima inside $[-10, 10]^2$.

$$f(x) = -\sum_{i=1}^{n}\sum_{j=1}^{5} j\sin((j+1)x_i + j)$$

6.17 M0 Test Function ([52])

This test function has 66 minima inside $[-5, 1]^2$.

$$f(x) = \sin(2.2\pi x_1 + \frac{\pi}{2})\frac{2 - x_2}{2}\frac{3 - x_1}{2} + \sin(\frac{\pi}{2}x_2^2 + \frac{\pi}{2})\frac{2 - x_2}{2}\frac{3 - x_1}{2}$$

6.18 M3 Test Function ([52])

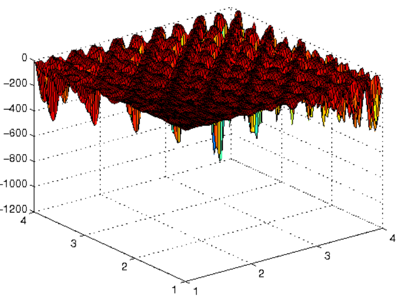
This test function has 26 minima inside $[-2, 2]^2$.

$$f(x) = -(x_2^2 - 4.5x_2^2)x_1x_2 - 4.7\cos(3x_1 - x_2^2(2 + x_1))\sin(2.5\pi * x_1) + (0.3 * x1)^2$$

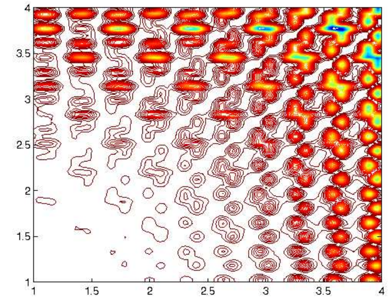
6.19 Siam Problem 4 Function ([53])

This test function has 600 minima inside $[-1, 1]^2$.

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

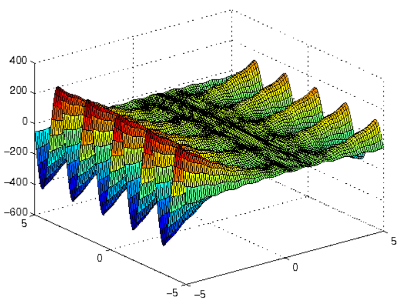


(a) Surface plot

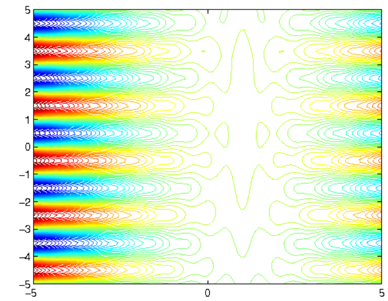


(b) Contour plot

Figure 16: Liangs's test function

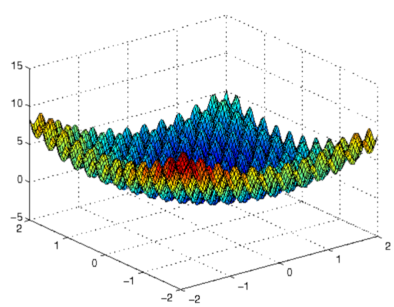


(a) Surface plot

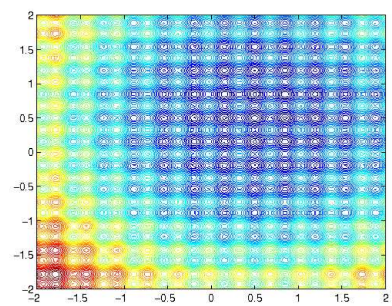


(b) Contour plot

Figure 17: Piccioni's test function

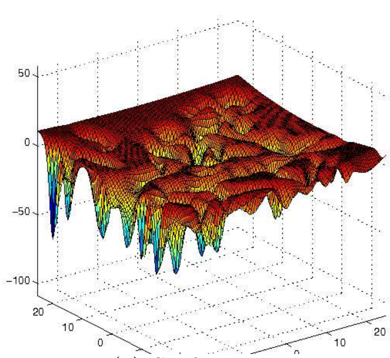


(a) Surface plot

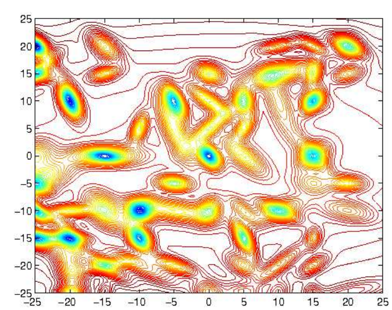


(b) Contour plot

Figure 18: Rastrigin's test function

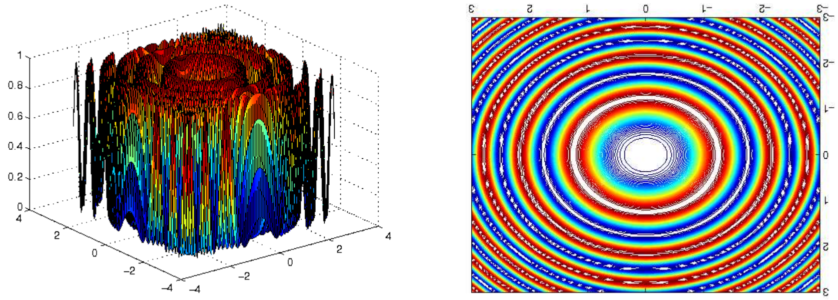


(a) Surface plot



(b) Contour plot

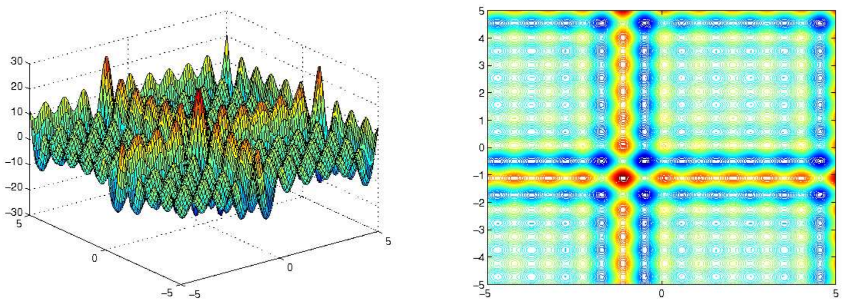
Figure 19: Voglis 's test function



(a) Surface plot

(b) Contour plot

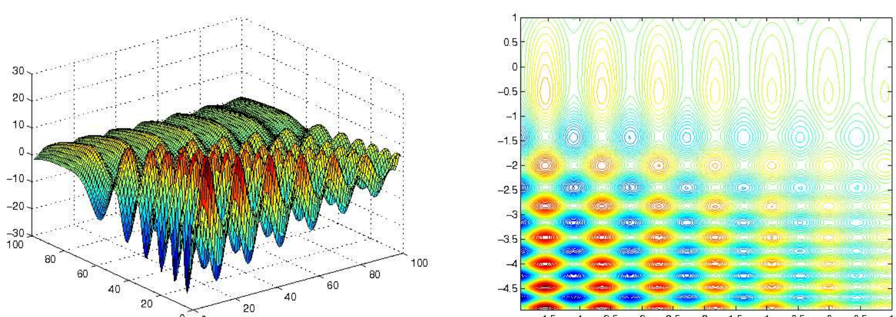
Figure 20: Schaffer's test function



(a) Surface plot

(b) Contour plot

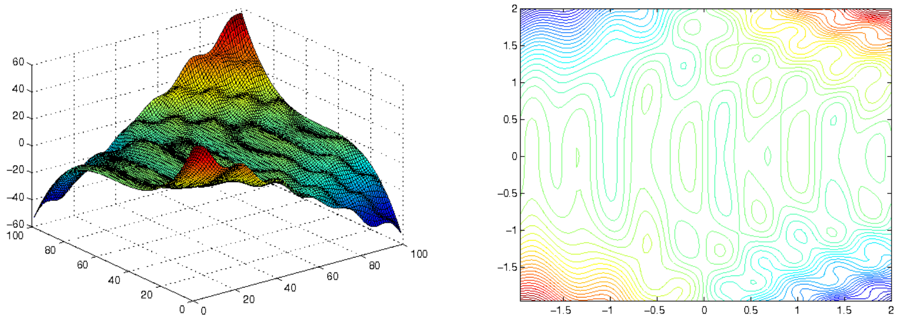
Figure 21: Shubert's test function



(a) Surface plot

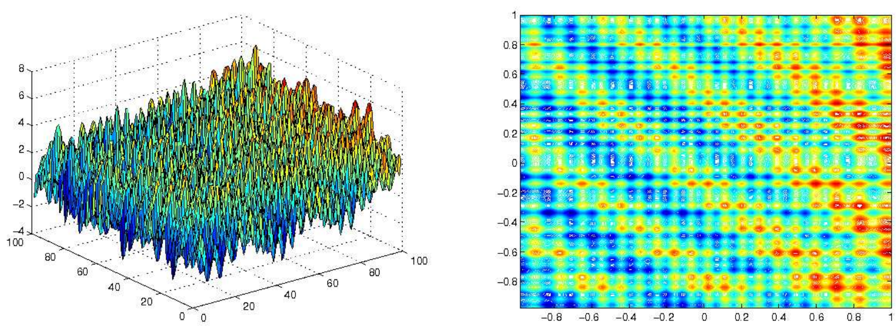
(b) Contour plot

Figure 22: M0 test function



(a) Surface plot (b) Contour plot

Figure 23: M3 test function



(a) Surface plot (b) Contour plot

Figure 24: Siam Problem 4 test function