Gradient–Controlled, Typical–Distance Clustering for Global Optimization.

I. G. Tsoulos and I. E. Lagaris*
Department of Computer Science, University of Ioannina
P.O.Box 1186, Ioannina 45110 - GREECE

Abstract

We present a stochastic global optimization method that employs a clustering technique which is based on a typical distance and a gradient test. The method aims to recover all the local minima inside a rectangular domain. A new stopping rule is used. Comparative results on a set of test functions are reported.

Keywords: *Stochastic Global optimization, Multistart, Stopping rules.*

1 Introduction

The task of locating all the local minima of a continuous function inside a bounded domain, is frequently required in several scientific as well as practical problems. Methods that seek for the global minimum only, are not appropriate in this case. On the other hand, methods that retrieve all the local minima belong to the class of global optimization methods. In mathematical terms the problem 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. An obvious method that suggests itself is the so called *Multistart*. The main idea is to sample points at random from S and to initiate from each one a local search to retrieve a minimum. The method is inefficient since it rediscovers the same minima over and over. If by X^* we denote the (initially empty) set of the distinct local minima found so far, *Multistart* proceeds as follows:

Step–0: Set $i = 0$ and $X^* = \emptyset$

Step–1: Sample x at random from S

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

Step–2: Apply a local search procedure L, starting from x and concluding at a local minimum x^*

Step–3: Check if a new minimum is recovered and if so add it to X^*, i.e.:
If $x^* \ni X^*$ **then**
increment: $i \leftarrow i + 1$
set: $x_i^* = x^*$
add: $X^* \leftarrow X^* \cup \{x_i^*\}$
Endif

Step–4: If a stopping rule applies STOP, otherwise go to Step-1

Good stopping rules are important and should combine reliability and economy. A reliable rule is one that stops only when all minima have been collected with certainty. An economical rule is one that does not spend a huge number of local searches to detect that all minima have been found. The **"region of attraction"** or the **"basin"** of a local minimum associated with a local search procedure L is defined as:

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

where $L(x)$ is the minimizer returned when the local search procedure 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 $A \subseteq R^n$. Since by definition the regions of attraction do not overlap, i.e. $A_i \cap A_j = \emptyset$ for $i \neq j$, then 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:

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

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 $m(A_i) \neq 0$.

The main disadvantage of *Multistart* is that the local search may be initiated from points belonging to the same basin of attraction, leading to the discovery of the same minimizer over and over, wasting so computational resources. A way of curing that, would be to design algorithms capable of recognizing if a point belongs to the basin of an already found minimizer and hence avoid starting a local search from there. This however can be achieved only within a probabilistic framework, since otherwise it would mean the apriori knowledge of the different regions of attraction. The clustering approach aims to form clusters of points that belong to the same region of attraction and then start a local search from only one point of each cluster. To form these

clusters one starts from a uniform sample and then applies either a *reduction* (Becker and Lago (1970)), in which case a certain fraction of points with the highest function values is removed, or a *concentration* (Törn (1978)), where a few number of steepest descent steps are applied to every point of the sample, aiming to increase the sample density around the minima. Such methods were devised in Boender et. al. (1982) and Kan and Timmer Part I (1987) and paved the way toward the more efficient methods of Multilevel-Single Linkage (MLSL) (Kan and Timmer Part II (1987)) and Topographical MLSL (Törn and Viitanen (1994), Ali and Storey (1994)).
The new algorithm, described in this article, attempts to find all local minima inside a bounded set, that are potentially global. These local minima are obtained by a local search procedure starting from suitably chosen points in a properly maintained sample. The algorithm tries to identify the regions of attraction as economically as possible and employs a clustering procedure without the explicit formation of the clusters. We adopt a modified topographical technique that is based on a gradient test rather than on comparing function values. We also calculate a typical distance that depends on the objective function topology in order to set a realistic scale.

2 Gradient–Controlled, Typical–Distance Clustering (GTC)

We proceed by describing the two key elements that control the clustering, i.e. the typical distance and the gradient test.

2.1 The Typical Distance

The typical distance is calculated by:

$$r_t = \frac{1}{M}\sum_{i=1}^{M}|x_i - L(x_i)| \tag{6}$$

where x_i are starting–points for the local search L, and M is the number of the performed local searches. This distance, is problem dependent (unlike the critical distance of Kan (Kan and Timmer Part I (1987), Kan and Timmer Part II (1987))) and yields an estimate for the mean basin–radius. To see this more clearly, note that if w is the number of the distinct minima found so far and if M_l is the number of times the local search discovered the minimizer x_l^*, a basin radius may be defined as:

$$R_l \equiv \frac{1}{M_l}\sum_{j=1}^{M_l}|x_l^{(j)} - x_l^*| \tag{7}$$

where $\{x_l^{(j)},\ \ j=1,\cdots,M_l\} = \{x_i,\ \ i=1,\cdots,M\} \cap A_l$, i.e. $L(x_l^{(j)}) = x_l^*$.
Since $M = \sum_{l=1}^{w} M_l$, a mean radius over the basins is given by:

$$< R > \equiv \sum_{l=1}^{w}\frac{M_l}{M}R_l = \frac{1}{M}\sum_{l=1}^{w}\sum_{j=1}^{M_l}|x_l^{(j)} - x_l^*| \tag{8}$$

Comparing eqs. (6),(7) and (8), it follows that $r_t = < R >$.

2.2 The Gradient Test

The gradient test originates from the observation that around a local minimum x^*, where a cluster develops, the objective function can be approximated by:

$$f(x) \approx f(x^*) + \frac{1}{2}(x - x^*)^T B^*(x - x^*) \qquad (9)$$

where B^* is the matrix of second derivatives (Hessian) at the minimum. Taking the gradient of the above we obtain: $\nabla f(x) \approx B^*(x - x^*)$. Similarly for any other point y in the neighborhood of x^*, $\nabla f(y) \approx B^*(y - x^*)$ and hence by subtracting and multiplying from the left by $(x - y)^T$ we obtain:

$$(x - y)^T[\nabla f(x) - \nabla f(y)] \approx (x - y)^T B^*(x - y) > 0 \qquad (10)$$

since B^* is positive definite.

2.3 The Starting–Point Set

At the start of each iteration, points are sampled from S, and they are all considered as starting–point candidates. A rejection mechanism is adopted in order to limit the number of starting points. A point x, ceases to be a candidate, when there exists a neighbor point p, satisfying $|x - p| < r_t$ and $(x - p)^T(\nabla f(x) - \nabla f(p)) > 0$ and in addition there exists a minimizer x^* with $\max\{|x - x^*|, |p - x^*|\} \leq R_x$, and $(x - x^*)^T \nabla f(x) > 0$ and $(p - x^*)^T \nabla f(p) > 0$, where

$$R_x = \max_i \{|x_i - L(x_i)|\} \qquad (11)$$

is the maximum distance an initial point was sampled away from its associated local minimizer. Note that mutual exclusion is not allowed. Namely, if a point x is excluded from the starting point set, due to the presence of point p, then point p cannot be excluded due to the presence of point x.

2.4 Algorithmic presentation

The algorithm performs iterations. At iteration k let $D^{(k)}$ be the working set of points, $X^{*(k)}$ the set of local minima retrieved so far, and $r_t^{(k)}$ the typical distance. Initially set $k = 0, r_t^{(0)} = 0, R_x = 0, D^{(0)} = X^{*(0)} = \emptyset$. The k^{th} iteration is described below.

2.4.1 GTC Algorithm

1. Sample N points $x_i, \quad i = 1, \cdots, N$ from S.
2. Set $X^{*(k+1)} = X^{*(k)}$, and $D^{(k+1)} = \{x_1, x_2, \cdots, x_N\} \cup X^{*(k+1)}$
3. $\forall\ i = 1, \cdots, N$
   Find the set $V = \{p_{i1}, \cdots, p_{iq}\} \subset D^{(k+1)}$ of the q nearest neighbors of x_i, that have not been excluded from being starting point candidates due to x_i's presence.

If $\exists\ p_{ij} \in V$ and $x^* \in X^{*(k+1)}$ such that:
$|x_i - p_{ij}| < r_t^{(k)}$ and $(x_i - p_{ij})^T(\nabla f(x_i) - \nabla f(p_{ij})) > 0$ and
$|x_i - x^*| < R_x$ and $(x_i - x^*)^T \nabla f(x_i) > 0$ and
$|p_{ij} - x^*| < R_x$ and $(p_{ij} - x^*)^T \nabla f(p_{ij}) > 0$
Then
x_i is not a starting–point.
Else
x_i is a starting–point. Start a local search $y = L(x_i)$.
Compute the typical distance using eq. (6) and R_x using eq. (11).
If y is a new minimum, **then**
Update: $X^{*(k+1)} \leftarrow X^{*(k+1)} \cup \{y\}$
$D^{(k+1)} \leftarrow D^{(k+1)} \cup \{y\}$
Endif
Endif

4. Increment $k \leftarrow k + 1$

3 The Double-Box Stopping Rule

The termination rule introduced is tightly coupled to the sampling scheme. A larger box S_2 is constructed that contains S and such that $m(S_2) = 2 \times m(S)$. At every iteration the N points in S are collected, by sampling uniformly from S_2 and rejecting points not contained in S. After k iterations, let the accumulated number of points sampled from S_2 be M_k, kN of which are contained in S. The quantity then: $\delta_k \equiv \frac{kN}{M_k}$ has an expectation value $<\delta>= \frac{1}{k}\sum_{i=1}^{k}\delta_i$ that asymptotically, i.e. for large k, tends to $\frac{m(S)}{m(S_2)} = \frac{1}{2}$. The variance given by $\sigma^2(\delta) =<\delta^2> - <\delta>^2$ tends to zero as $k \to \infty$. The deviation $\sigma(\delta)$ is a measure of the uncovered fraction of the search space. We permit iterating without finding new minima until $\sigma^2 < \frac{1}{2}\sigma_{last}^2$, where σ_{last} is the standard deviation at the iteration during which the most recent minimum was found. With the above definitions the algorithm can be stated as:

1. Initially set $\alpha = 0$
2. Sample N points as described above.
3. Calculate $\sigma^2(\delta)$
4. Apply an iteration of the GTC algorithm, neglecting its first step and using instead the N points already sampled.
5. If one or more new minima are found, set: $\alpha = \sigma^2/2$ and repeat from step 2.
6. STOP if $\sigma^2 < \alpha$, otherwise repeat from step 2.

4 Numerical experiments

Several experiments were performed with a host of test functions. Each experiment was repeated ten times, each time with a different seed for the random number generator. The reported results are averages over the ten runs. We compare our method

to Multistart, MLSL (Kan and Timmer Part II (1987)) and TMLSL (Ali and Storey (1994)), using in all methods the stopping rule employed by Kan in Kan and Timmer Part II (1987). Namely, the algorithm stops if $\frac{w(M-1)}{M-w-2} < w + \frac{1}{2}$, M being the total number of sample points and w the number of retrieved minimizers. Note that the local procedure used, is an implementation of BFGS with a line search and analytically calculated gradient. We also compare to Multistart and to TMLSL using in all methods the introduced in this article "Double Box Stopping Rule" (DBSR). In the TMLSL and GTC experiments the number of nearest neighbors considered was set equal to one.
We report results in tables. We list the objective function name, the number of existing minima, the number of retrieved minima, the number of function evaluations, the number of gradient evaluations and the CPU-time in seconds, under the headings FUNCTION, NOEM, NORM, NOFC, NOGC, and TIME correspondingly.

4.1 Comparing the termination rules

Comparison of Tables I and II, shows the superiority of the DBSR. The CPU time in the case of Multistart, scales linearly with the number of function and gradient evaluations. This however is not true with MLSL, since as the sample size increases, the task of finding the points within the range of the critical radius, becomes heavier and heavier. In the case of TMLSL and GTC, where the sample size increases only at the pace of the number of minima, the collection of the nearest neighbors does not appear to be of major concern. Hence for problems requiring many iterations, generating thus an oversized sample, MLSL is not suggested. For this reason we have not performed further experiments with MLSL using our new DBSR. The advantage of our new stopping rule in TMLSL and GTC is clear as may be readily deduced comparing Tables IV and V (TMLSL) and Tables VI and VII (GTC).

4.2 Comparing GTC to Multistart

Multistart offers a basis for comparison. It is simple to code and has been extensively used. There is an overhead in the performance of GTC, that is due to the process of nearest neighbor searching. This is infered from the quoted CPU-times that are in disharmony with the numbers of functional/gradient evaluations. For instance in Tables I and VI for the *Shubert* function, *Multistart* performs 7.7×10^6 functional and gradient evaluations corresponding to a CPU-time of only 112 seconds, while GTC performs only 187846 functional and 501505 gradient calls requiring 334 seconds. The extra time needed by GTC (and the same holds true for TMLSL) is for the nearest neighbor search. For time consuming objective functions this overhead will hardly be noticed. The important feature to pay attention at, is clearly the number of calls to the objective function and its gradient. By inspection of the corresponding tables, GTC is by far superior to *Multistart*.

4.3 Comparing GTC to TMLSL

The comparison of GTC to TMLSL is of main interest, since TMLSL is one of the most successful stochastic global optimization methods. We have used variations of TMLSL to solve problems in molecular mechanics and have developed code contained in the PANMIN package (Theos et al (2004)). Using Kan's original termination criterion, TMLSL stops prematurely in most cases (Table IV, *Shubert, Rastrigin, Gkls, Griewank, Hansen, Camel, Shekel*). Table VI shows that GTC also stops early in the case of Gkls, Camel and Shekel but collects more minimizers than TMLSL. Finally when the new DBSR is used, comparing Tables V and VII, we see that GTC misses only a few minima in the case of $Gkls(3, 100)$ and $Gkls(4, 100)$. (TMLSL misses a few more). The efficiency of each method depends on the number of function and gradient evaluations. GTC is significantly more efficient. On the average TMLSL needs twice as many functional/gradient evaluations and in some cases even four times as many.

TABLE I. Multistart, Kan's Stopping rule

FUNCTION	NOEM	NORM	NOFC	NOGC	TIME
Shubert	400	400	7713921	7713921	111.77
Rastrigin	49	49	116122	116122	0.94
GKLS(3,30)	30	15	5085	5085	0.25
GKLS(3,100)	100	56	82622	82622	10.09
GKLS(4,100)	100	25	26083	26083	4.67
Guilin(20,100)	100	100	1741346	1741346	287.54
Griewank-2	529	529	18941873	18941873	263.90
Hansen	527	527	13661461	13661461	227.74
Camel	6	6	2830	2830	0.02
Shekel-10	10	9	10701	10701	0.94

TABLE II. Multistart, Double Box Stopping rule

FUNCTION	NOEM	NORM	NOFC	NOGC	TIME
Shubert	400	400	615904	615904	8.94
Rastrigin	49	49	18661	18661	0.15
GKLS(3,30)	30	28	195546	195546	9.49
GKLS(3,100)	100	97	7467472	7467472	915.35
GKLS(4,100)	100	97	9890042	9890042	1775.5
Guilin(20,100)	100	100	1617099	1617099	267.09
Griewank-2	529	529	1832773	1832773	26.15
Hansen	527	527	543990	543990	9.07
Camel	6	6	4098	4098	0.03
Shekel-10	10	10	35832	35832	3.13

TABLE III. MLSL, Kan's Stopping rule

FUNCTION	NOEM	NORM	NOFC	NOGC	TIME
Shubert	400	159	128344	11634	248.30
Rastrigin	49	6	637	237	0.01
GKLS(3,30)	30	6	876	276	0.03
GKLS(3,100)	100	6	779	179	0.04
GKLS(4,100)	100	3	871	71	0.05
Guilin(20,100)	100	12	5776	1776	1.40
Griewank-2	529	4	506	106	0.01
Hansen	527	20	1636	560	0.04
Camel	6	3	499	99	0.01
Shekel-10	10	6	947	207	0.04

TABLE IV. TMLSL, Kan's Stopping rule

FUNCTION	NOEM	NORM	NOFC	NOGC	TIME
Shubert	400	165	154926	26306	125.24
Rastrigin	49	20	2597	969	0.20
GKLS(3,30)	30	9	821	521	0.05
GKLS(3,100)	100	12	1490	1004	0.19
GKLS(4,100)	100	3	535	135	0.08
Guilin(20,100)	100	99	400226	380386	89.15
Griewank-2	529	215	1470660	1246240	358.07
Hansen	527	423	528331	82885	602.15
Camel	6	4	415	215	0.01
Shekel-10	10	6	646	276	0.07

TABLE V. TMLSL, Double Box Stopping rule

FUNCTION	NOEM	NORM	NOFC	NOGC	TIME
Shubert	400	400	360416	61132	287.66
Rastrigin	49	49	27850	15056	1.90
GKLS(3,30)	30	29	125141	78494	11.26
GKLS(3,100)	100	94	16905021	9324132	3749.7
GKLS(4,100)	100	94	9719338	5156518	2635.5
Guilin(20,100)	100	100	1676572	1616692	354.28
Griewank-2	529	528	2006690	1693136	491.53
Hansen	527	527	808274	127356	921.24
Camel	6	6	3641	2881	0.07
Shekel-10	10	10	44162	39044	4.03

TABLE VI. GTC, Kan's Stopping rule

FUNCTION	NOEM	NORM	NOFC	NOGC	TIME
Shubert	400	400	187846	501505	334.08
Rastrigin	49	49	20657	24591	0.92
GKLS(3,30)	30	9	537	817	0.07
GKLS(3,100)	100	13	1103	1555	0.31
GKLS(4,100)	100	8	466	842	0.30
Guilin(20,100)	100	100	366280	382841	81.82
Griewank-2	529	529	4265130	4711645	806.58
Hansen	527	527	1223284	1731469	819.77
Camel	6	5	248	443	0.01
Shekel-10	10	9	4748	5046	0.52

TABLE VII. GTC, Double Box Stopping rule

FUNCTION	NOEM	NORM	NOFC	NOGC	TIME
Shubert	400	400	31674	59044	28.98
Rastrigin	49	49	4449	5090	0.16
GKLS(3,30)	30	30	80275	134247	16.15
GKLS(3,100)	100	97	4160897	5663373	1482.3
GKLS(4,100)	100	96	2564480	3878680	1437.7
Guilin(20,100)	100	100	792052	827892	177.68
Griewank-2	529	529	1032445	1140113	191.41
Hansen	527	527	82572	109020	42.28
Camel	6	6	844	1705	0.05
Shekel-10	10	10	20226	21597	2.25

4.4 Test Functions

We list the test functions used in our experiments, the associated search domains and the number of the existing local minima.

1. **Rastrigin**.
   $f(x) = x_1^2 + x_2^2 - cos(18x_1) - cos(18x_2)$
   $x \in [-1, 1]^2$ with 49 local minima.
2. **Shubert**.
   $$f(x) = -\sum_{i=1}^{2}\sum_{j=1}^{5} j\{sin[(j+1)x_i] + 1\}$$
   $x \in [-10, 10]^2$ with 400 local minima.
3. **GKLS**.
   $f(x) = Gkls(x, n, w)$, is a function with w local minima, described in Gaviano et al (2003).
   $x \in [-1, 1]^n, n \in [2, 100]$
4. **Guilin Hills**.
   $$f(x) = 3 + \sum_{i=1}^{n} c_i \frac{x_i + 9}{x_i + 10} sin\left(\frac{\pi}{1 - x_i + 1/(2k_i)}\right)$$
   $x \in [0, 1]^n$, $c_i > 0$, and k_i are positive integers. This function has $\prod_{i=1}^{n} k_i$ minima. In our experiments we chose $n = 20$ and arranged k_i so that the number of minima is 100.
5. **Griewank # 2**.
   $$f(x) = 1 + \frac{1}{200}\sum_{i=1}^{2} x_i^2 - \prod_{i=1}^{2}\frac{cos(x_i)}{\sqrt(i)}$$
   $x \in [-100, 100]^2$ with 529 minima.
6. **Hansen**.
   $$f(x) = \sum_{i=1}^{5} i\cos[(i-1)x_1 + i]\sum_{j=1}^{5} j\cos[(j+1)x_2 + j]$$
   $x \in [-10, 10]^2$ with 527 minima.
7. **Camel**.
   $$f(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$$
   $x \in [-5, 5]^2$ with 6 minima.
8. **Shekel-10**.
   $$f(x) = -\sum_{i=1}^{m}\left(\frac{1}{(x - A_i)(x - A_i)^T + c_i}\right)$$

where: $A = \begin{bmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \\ 2 & 9 & 2 & 9 \\ 5 & 5 & 3 & 3 \\ 8 & 1 & 8 & 1 \\ 6 & 2 & 6 & 2 \\ 7 & 3.6 & 7 & 3.6 \end{bmatrix}$ $c = \begin{bmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \\ 0.6 \\ 0.3 \\ 0.7 \\ 0.5 \\ 0.5 \end{bmatrix}$

$x \in [0, 10]^4$ with 10 minima.

5 Conclusions

We presented "GTC", a new stochastic Global Optimization method, that takes in account the topology of the objective function to effectively decide when to apply a local search. The mean basin radius, sets a realistic scale and this seems to be important. In addition a new stopping rule is introduced based on a space-covering rational, which works very well in practice. The comparison with well established Multistart based methods, favors GTC and hence encourages its application to hard practical problems.

References

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

Becker, R.W. and Lago, G.V.(1970), *A global optimization algorithm*, in Proceedings of the 8^{th} Allerton Conference on Circuits and Systems Theory.

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.

Gaviano, M. and Ksasov, D.E. and Lera, D. and Sergeyev, Y.D.(2003), *Software for generation of classes of test functions with known local and global minima for global optimization*, ACM Trans. Math. Softw. **29**, 469-480

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

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

Theos, F. V. and 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

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.

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