Journal of Shanghai University (English Edition)
ISSN 1007-6417, *Vol.*4, *No.*2(*Jun.*2000), *pp* 92~95

Improvement of Pure Random Search in Global Optimization

PENG Jian-ping[1], *SHI Ding-hua*[2]
1. *Shanghai Municipal Commission of Science and Technology, Shanghai* 200003, *China*
2. *College of Sciences, Shanghai University, Shanghai* 200436, *China*

Abstract In this paper, the improvement of pure random search is studied. By taking some information of the function to be minimized into consideration, the authors propose two stochastic global optimization algorithms. Some numerical experiments for the new stochastic global optimization algorithms are presented for a class of test problems.

Key words random search, global optimization, stochastic global optimization algorithm

1 Introduction

Consider the following global optimization problem:

$$\min_{x\in L} f(x),$$

where f is a real-valued continuous function over L, $L \subset R^d$ is of the form $L = \Pi_{i=1}^{d}[a_i, b_i]$, with $a_i, b_i \in R, a_i < b_i$. Denote $f_* = \min_{x\in L} f(x) = f(x_*)$ and define $S_\varepsilon = [f_*, f_* + \varepsilon]$ as a stopping criterion in the following algorithms, where ε is a sufficiently small positive number.

A stochastic global optimization algorithm is concerned with using randomly generated points to approximate optimal solution. The simplest stochastic global optimization algorithm is the so-called pure random search(PRS), which was named early in 1958[1]. This algorithm generates a sequence of independent and uniformly distributed points in the feasible region. When a stopping criterion is met, the best point of the sequence generated thus far is used as an approximation to the optimal solution[2~5]. In 1981, on the basis of Ref. [6] Rubinstein reached the important conclusion that if the objective function f is continuous, then the PRS converges to the global optimal solution with probability 1[7]. However, in 1992, Zabinsky and Smith proved that the PRS itself is inefficient for the reason that the simple device of forcing monotone value improvement on PRS achieves an exponential improvement in iterations required[8]. Thus, the PRS will be unable to implement when the dimension of the problem is very large. To overcome this drawback, researchers have done a lot of work. First, in 1978, W. L. Price proposed a controlled random search (CRS)[9], in which the random search and mode search are combined in the optimizing process. It does not require f to be differentiable and continuous. The CRS is simple to implement. The probability of success depends on the parameters of CRS and the properties of the problem. In 1983, Price presented another improving algorithm CRS2 for CRS[10], and correspondingly, CRS is called CRS1. Comparing with CRS1, CRS2 is easier to implement, the needed memory is less and it converges to the optimum faster. In 1987, Price obtained again two improving algorithms based on CRS2[11], called CRS3 and CCRS, respectively. CRS3 combines the global search with a local optimization algorithm[12] for accelerating the speed of convergence. CCRS is a parallel scheme of CRS3. It also aims at speeding up the convergence process. In 1988, C. Mohan and Kuanker Shanker provided four modified versions of controlled random search to make the algorithms reliable and convergent faster[13].

Considering the stochastic global optimization algorithms and their improvements, we found that most of them, such as PRS, adaptive search[14] and simulated annealing[15] *etc.*, did not make full use of the characteristics of the objective function. We think that if the objective function satisfies some conditions such as continuity, the Lipschitz condition or differentiability *etc.*, the regions containing the minimal points are easy to be determined. If the algorithm generates many points in

Received Jun. 29, 1999; Revised Sep. 13, 1999
Supported by the Science Foundation of Shanghai Municipal Commission of Education

the regions of this kind, the probability of success will be large. Motivated by this idea, we try to modify PRS further. And we get two algorithms, called the improving pure random search-1(IPRS-1) and improving pure random search-2(IPRS-2), respectively. The difference between them lies in the designs for different kinds of functions. The greatest merit of these new algorithms is that the distribution of the sample is decided directly by the characteristic of the objective function itself. The information of the objective function f is thus fully used.

In the following sections, we will give the steps of PRS in Section 2. The improved pure random research-1 and improved pure random research-2 are described in Sections 3 and 4, respectively. The numerical results are presented in Section 5.

2 Pure Random Research

The steps of PRS are as follows.

Pure Random Search(PRS)

Step 0 Set $n=0$.

Step 1 Generate a point x_n uniformly distributed in L, and set $y_n=f(x_n)$.

Step 2 If a stopping criterion is met, stop. Otherwise, increase n and return to Step 1.

3 Improved Pure Random Search-1

We assume in addition, that f has not too many local minima in a very small region. The improving pure random search-1 is designed as follows.

Improved Pure Random Search-1(IPRS-1)

Step 0 Set $n=0$.

Step 1 Choose a positive integer m divide L into m pieces, denoted by Δ_i, where $i=1,2,\cdots,m$. Generate a point x_i uniformly distributed in every Δ_i. Compute $f(x_i)$and denote

$$y_0=\min_{1\leqslant i\leqslant m} f(x_i).$$

Step 2 If a stopping criterion is met, stop.

Step 3 Allocate a probability interval U_i in $[0,1]$ for every Δ_i, which have the length l_i

$$l_i=\frac{e^{-f(x_i)}}{\sum_{i=1}^{m}e^{-f(x_i)}},$$

where $\bigcup_{i=1}^{m}U_i=[0,1]$, $\sum_{i=1}^{m}l_i=1$. Determine r, $1\leqslant r\leqslant m$, such that

$$l_r=\max_{1\leqslant i\leqslant m} l_i.$$

Step 4 Increase n.

Step 5 Generate a random number ξ uniformly distributed in $[0,1]$.

Step 6 Choose a U_j such that $\xi\in U_j$, then generate a point z_j uniformly distributed in the corresponding Δ_j. Denote $y_n=f(z_j)$.

If $f(z_j)<f(x_j)$, set $x_j=z_j$, return to Step 2.

If $f(z_j)=f(x_j)$, return to Step 4.

Otherwise, let

$$l_r=l_r+\frac{e^{-f(x_j)}}{\sum_{i=1}^{m}e^{-f(x_i)}}-\frac{e^{-f(z_j)}}{\sum_{i=1,i\neq j}^{m}e^{-f(x_i)}+e^{-f(z_j)}}$$

and

$$l_j=\frac{e^{-f(z_j)}}{\sum_{i=1,i\neq j}^{m}e^{-f(x_i)}+e^{-f(z_j)}},$$

then return to Step 4.

When the function satisfies a slightly more restrictive conditions, IPRS-1 almost ensures that the generated points concentrate near x_*. That leads to very large probability of catching the minimum. At the same time, the probability of generating the point with high function value is becoming smaller. This speeds up the process of reaching what we want.

If the function value varies within a large range in very small region or has many local minima in very small region, it is possible that the minimum will be lost in the running of IPRS-1. For this kind of functions, we change the generating rule. The resulting algorithm is called the improved pure random search-2(IPRS-2).

4 Improved Pure Random Search-2

Improved Pure Random Search-2(IPRS-2)

Step 0 Set $n=0$.

Step 1 Choose a positive integer m, divide L into m pieces, denoted by Δ_i, where $i=1,2,\cdots,m$. Generate a point x_i uniformly distributed in every Δ_i. Compute $f(x_i)$ and denote

$$y_0=\min_{1\leqslant i\leqslant m} f(x_i).$$

Step 2 If a stopping criterion is met, stop.

Step 3 Determine r such that

$$f(x_r)=\min_{1\leqslant i\leqslant m} f(x_i).$$

Step 4 Allocate a probability interval U_i in $[0,1]$

for every Δ_i, which have the length l_i,

$$l_i = \frac{e^{-f(x_i)}}{\sum_{i=1}^{m} e^{-f(x_i)}} = l(\geqslant \frac{1}{m})$$

and

$$l_1 = l_2 = \cdots = l_{r-1} = l_{r+1} = \cdots = l_m = \frac{1-l}{m-1},$$

where

$$\bigcup_{i=1}^{m} U_i = [0,1], \quad \sum_{i=1}^{m} l_i = 1.$$

Step 5 Increase n.

Step 6 Generate a random number ξ uniformly distributed in $[0,1]$.

Step 7 Choose an U_j such that $\xi \in U_j$, then generate a point z_j uniformly distributed in the corresponding Δ_j. Denote $y_n = f(z_j)$.

If $f(z_j) < f(x_j)$, set $x_j = z_j$, return to Step 2.

Otherwise, return to Step 5.

The IPRS-2 only ensures that the probability of generating points near the current minimum is large, and the probabilities of generating other points are the same. This prevents from being trapped in local minimum, to some extent, for the function with many local minimum in very small regions.

5 Numerical Experiments

In this section, we first describe some test problems in detail.

Test problems

1. $f(x) = 9x_1^2 + 36x_1x_2 + 52x_2^2 + 30(x_1^2 + 4x_2^2 + 2x_1 - 16)^2,$
$-4 \leqslant x_1 \leqslant 4, \quad -2 \leqslant x_2 \leqslant 2.$

2. $f(x) = \frac{1}{2}x_1^2 + \frac{1}{2}(1 - \cos(2x_1)) + x_2^2,$
$-10 \leqslant x_1 \leqslant 10, \quad -10 \leqslant x_2 \leqslant 10.$

3. $f(x) = \frac{1}{4}x_1^4 - \frac{1}{2}x_1^2 + \frac{1}{10}x_1 + \frac{1}{2}x_2^2,$
$-2 \leqslant x_1 \leqslant 2, \quad -2 \leqslant x_2 \leqslant 2.$

4. Goldstein/Price

$$f(x) = (1 + (x_1 + 2x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 28x_2 + 12x_1x_2 + 12x_2^2))(30 + (2x_1 - 6x_2)^2(18 - 32x_1 + 12x_1^2 + 96x_2 - 72x_1x_2 + 108x_2^2)),$$
$$-2 \leqslant x_1 \leqslant 2, \quad -1 \leqslant x_2 \leqslant 1.$$

5. Branin

$$f(x) = (x_2 - 1.275x_1^2 + 5x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})\cos(\pi x_1) + 10,$$
$$-1.5 \leqslant x_1 \leqslant 3.5, \quad 0 \leqslant x_2 \leqslant 15.$$

6. $f(x) = 0.1(\sin^2(3\pi x_1) + \sum_{j=1}^{d-1}(x_j - 1)^2(1 + \sin^2(3\pi x_{j+1})) + (x_d - 1)^2(1 + \sin^2(2\pi x_d))),$
$-10 \leqslant x_j \leqslant 10, \quad j = 1, \cdots, d.$

There test problems are taken from Ref. [14]. The results of running IPRS-1 and IPRS-2 are all average over 50 runs, where ε is taken to be 0.01. Tables 1 and 2 show the number of the problems as P, the number of division of L as m, the average number of iterations as $E[N]$. We also show the number of function evaluations, where available, for the best results reported in literature for a particular problem. One should be careful in comparing the algorithms in this way, since different algorithms use different stopping criteria and each algorithm is designed for some restricted class of optimization problems.

Table 1 Results for test problems

P	IPRS-1		IPRS-2		Best other
	m	$E[N]$	m	$E[N]$	
1	8×16	7112.1	8×16	8426.1	547.4[1]
	16×32	1996.5	16×32	5156.0	
	20×40	1300.7	20×40	3522.0	
	40×80	268.6	40×80	784.3	
2	10×10	835.2	10×10	1027.6	421.1[1]
	20×20	248.3	20×20	452.7	10286[4]
	40×40	112.0	40×40	320.1	
	50×50	148.6	50×50	323.6	
3	4×4	1529.3	4×4	1104.4	114.9[1]
	8×8	483.4	8×8	739.3	3402[4]
	20×20	251.0	20×20	249.0	
	40×40	69.7	40×40	106.4	
4	4×8	12126.0	4×8	10669.0	621.2[1]
	8×16	4351.0	8×16	6128.3	148[2]
	12×24	2167.3	12×24	2471.2	
	20×40	1096.3	20×40	824.9	
5	5×15	698.9	5×15	1892.5	717.6[1]
	10×30	1188.0	10×30	2214.0	160[3]
	20×60	1007.7	20×60	5661.5	
	30×90	878.4	30×90	5131.6	

Table 2 Results for test problem 6 with different dimensions

d	IPRS-1		IPRS-2	
	m	$E[M]$	m	$E[M]$
1	5	82.6	5	77.3
	10	53.8	10	53.5
	20	41.7	20	44.3
	40	24.1	40	39.5
2	5×5	62.5	5×5	47.3
	10×10	41.3	10×10	67.78
	20×20	16.7	20×20	83.3
	40×40	0	40×40	0
3	5×5×5	1986.6	5×5×5	2728.9
	10×10×10	478.5	10×10×10	5153.5
	15×15×15	469.9	15×15×15	4112.3
4	5×5×5×5	6764.6	5×5×5×5	14960.0

From Table 1, we can conclude that the IPRS-1 performs well in comparison with some other methods in literature. In all cases the optimal solutions were found within the desired accuracy in a reasonable amount of time and function evaluations. It is also clear that for the IPRS-1 and IPRS-2, in general, the more pieces L is divided into, the less the number of iterations is. Notice that Problem 5 has the opposite result. In this case, the function changes a little in a very large region containing the global minimum. Adding division number of L makes the useful region containing the global minimum to be divided into too many pieces. The probabilities of generating points in these pieces all keep large. Therefore, the probability of catching the global minimum becomes small. As a result, the number of iterations needed increases. Thus, for this kind of functions, one should not divide L into too many pieces so as to reach the global minimum fast. In all but Problem 4, the IPRS-1 is much better than the IPRS-2. The objective function of the Problem 4 has three local minimum in very small region. According to the principles of the algorithms, they should behave in this way. Table 2 shows that when $d \geqslant 2$, the iterations of the algorithms grow with d. Therefore, if d is not very large, the algorithms are available. Although we did not give the strict proofs of mathematical results for convergence and complexity for these two algorithms so far, we conjecture that the algorithms have, at least, better properties than the PRS. We will do the research further to set up solid mathematical foundations for the algorithms. In Tables 1 and 2, index (1) refers to the simulated annealing method (see Ref.[14]), (2) to multi-level single linkage, using a penalty function for handing the constraints (see Ref.[16]), (3) is random direction method (see Ref.[17]) and (4) is method based on stochastic differential equations (see Ref.[18]).

References

[1] Brooks S.H., A discussion of random methods for seeking maxima, *Operations Research*, 1958, **6**:244 - 251

[2] Anderssen R.S., Global optimization, In R.S. Anderssen, L.S. Jennings and D.M. Ryan(Editors), *Optimization*, University of Queensland Press, 1972

[3] Dixon L.C.W. and Szegö G.P., eds. *Towards Global Optimization* 2, North-Holland, Amsterdam, 1978

[4] Dixon L.C.W. and Szegö G.P., eds. *Towards Global Optimization* 1, North-Holland, Amsterdam, 1978

[5] Rinnooy Kan A.G.H. and Timmer G.T., Stochastic methods for global optimization. *American Journal of Mathematical and Management Sciences*, 1984, **4**:7 - 40

[6] Devroye L., Progressive global random search of continuous functions, *Mathematical programming*, 1978, **15**:330 - 342

[7] Rubinstein R.Y., *Simulation and Monte Carlo Method*, John Wiley and Sons, New York, 1981

[8] Zabinsky Z.B. and Smith R.L., Pure adaptive search in global optimization, *Mathematical Programming*, 1992, **53**:323 - 338

[9] Price W.L., A controlled random search procedure for global optimization. in: Dixon and Szegö, eds. *Towards Global Optimization* 2, North-Holland, Amsterdam, 1978

[10] Price W.L., Global optimization by controlled random search, *JOTA*, 1983, **40**:333 - 348

[11] Price W.L., Global optimization algorithms for CAD workstation, *JOTA*, 1987, **55**:133 - 146

[12] Nelder J.A. and Mead R., A simplex method for function minimization, *Computer Journal*, 1965, **7**:308 - 313

[13] Mohan C. and Kuanker Shanker, A numerical study of modified versions of controlled random search method for global optimization, *Intern. J. Computer Math.*, 1988, **23**:325 - 341

[14] Romeijn H.E., *Global Optimization by Random Walk Sampling Methods*, Ph D. Thesis, Erasmus University Rotterdam, 1992

[15] Kirkpatrick S., Gelatt C.D. Jr., Vecchi M.P., Optimization by simulated annealing, *Science*, 1983, **20**:671 - 680

[16] Timmer G.T., *Global Optimization: a Stochastic Approach*, Ph D. Thesis, Erasmus University Rotterdam, Rotterdam, The Netherlands, 1984

[17] Bremmerman H., A method of unconstrained global optimization, *Mathematical Biosciences*, 1970, **9**:1 - 15

[18] Aluffi-Pentini F., Parisi V. and Zirilli F., Global optimization and stochastic differential equations, *Journal of Optimization Theory and Applications*, 1985, **47**:1 - 16