A stochastic algorithm for function minimization

DONGCAI SU

School of Communications, Jilin University, Changchun, Jilin 130021, China

E-mail: suntree4152@gmail.com

JUNWEI DONG

Bradley Department of Electrical & Computer Engineering, Virginia Polytechnic Institute and State University, VA 22043, USA

E-mail: djwei@vt.edu

ZUDUO ZHENG

Department of Civil & Environmental Engineering, Hohai University and Arizona State University, 1115 E. Lemon St. 401 E Apt., Tempe, AZ 85281 USA

Email: zhengzuduo@gmail.com Phone: +1-480-727-9805

Abstract

Focusing on what an optimization problem may comply with, the so-called convergence conditions have been proposed and sequentially a stochastic optimization algorithm named as DSZ algorithm is presented in order to deal with both unconstrained and constrained optimizations. Its principle is discussed in the theoretical model of DSZ algorithm, from which we present a practical model of DSZ algorithm. Practical model's efficiency is demonstrated by comparison with similar algorithms such as Enhanced Simulated Annealing (ESA), Monte Carlo Simulated Annealing (MCS), Sniffer Global Optimization (SGO), Directed Tabu Search (DTS), and Genetic Algorithm (GA), using a set of well-known both unconstrained and constrained optimization test cases. Meanwhile, further attention goes to the strategy how to optimize the high-dimensional unconstrained problems using DSZ algorithm.

Keywords: Global optimization, unconstrained optimization, constrained optimization

1 Introduction

Assuming the optimization task is to minimize the objective function, and point p is the global solution. Regarding to the condition which the objective functions concerned in this paper satisfy: (section 2.1):

"The smaller a point's function value, the higher probability that this point is closer to p."

DSZ algorithm (section 2.4) is presented involving two strategies:

1. Point set evolution according to its function value;
2. Shrinking operation according to the shrinking coefficient c is employed during the point set evolution.

According to the computational experiments (section 2.5, 4.4), the efficiency of DSZ algorithm is encouraging. Furthermore, the strategy of handling high dimensional, unconstrained problem was discussed in section 3

2 Optimization Principles and Procedure

2.1 Conditions on the optimization problems

The considered objective function is denoted as $f(x), x:[x_1,\cdots,x_i,\cdots,x_n]$ where $x_i(1\le i\le n)$ is real number. Let D the region of $x_i(1\le i\le n)$, where D: $l\le x\le u$, $(l_i\le x_i\le u_i, 1\le i\le n)$. And $o=\frac{l+u}{2}$ is the center of D;

Without losing generality, the optimization task is to search for minima of $f(x)$. The corresponding solution for the global minima is $p:[p_1,\cdots,p_{i,}\cdots,p_n],(1\le i\le n)$.

$D_k^x=(k(D-o)+x)\cap D,(x\in D,0<k\le 2)$, whose central point is x, the ratio of similitude between D_k^x and D is k.

$$r_k^x=\max\left(\frac{2|x_i-p_i|}{k(u_i-l_i)},1\le i\le n\right),\ (x\in D,0<k\le 2)$$

Optimization problems considered in this paper shall meet the following three conditions:

(1) Continuity: Given that ε is a positive number however small, $x1$ and $x2$ are any two points in D, if $0<\|x1-x2\|<\delta$, When δ is sufficiently small, the $f(x)$ always satisfies $|f(x1)-f(x2)|<\varepsilon$.

(2) Convergence Conditions I: $\theta(k)$ is a stochastic value which is correlated to k ($k\in(0,2]$), defined as $\theta(k)=\frac{\|p-x'\|}{\|p-x\|}$ where $p\in D_k^x$, and x' as a random point in D_k^x is subjected to $f(x')<f(x)$. Then for any positive integer N_0, which is large enough, and $N>N_0$, we have

$$\prod_{i=1}^{N}\theta(k_i)<\varepsilon'$$

where ε' is a positive number however small, and $k_i(1\le i\le N)$ is any number from $(0,2]$.

(3) Convergence Conditions II: Assume $\lambda_0\in(0,1)$ and $P_0\in(0,1)$ are fixed constants and x' is a point randomly selected in D_k^x. If $p\in D_k^x$ and $r_k^x\ge\lambda_0$, then $prob[f(x')<f(x)]\ge P_0$, where $prob[*]$ denotes the probability of event *.

2.2 The Theoretical Model

The theoretical procedure of DSZ algorithm is shown below:

(1) Initializing: set j=1 and initialize the iterative number and the maximum iteration (max_iter); randomly generate x_1 from region D and initialize $k_1\in(0,2]$ to enforce $p\in D_{k_1}^{x_1}$ and $r_{k_1}^{x_1}\ge\lambda_0$;

(2) Randomly generate x' from region $D_{k_j}^{x_j}$. If $f(x')<f(x_j)$, then let $x_{j+1}=x'$, otherwise, let $x_{j+1}=x_j$;

(3) Choose c_j to ensure $p\in D_{k_{j+1}}^{x_{j+1}}$ and $r_{k_{j+1}}^{x_{j+1}}\ge\lambda_0$, where $k_{j+1}=c_jk_j$;

(4) Let j=j+1, return to step (2) until j reaches max_iter ;

(5) Take $x_{\max\_iter}$ as the optimum solution.

2.3 Optimization Principle

Theorem:

According to the theoretical model, if the maximum iteration max_iter is sufficiently large, then $|f(p)-f(x_{\max\_iter})|<\varepsilon$, where ε is a positive number however small, $x_{\max\_iter}$ is the optimum solution.

Proof:

Let $[x_{\tau(1)},\cdots,x_{\tau(i)},\cdots,x_{\tau(M)}]$ be the maximum subset of $[x_1,\cdots,x_i,\cdots,x_{\max\_iter}]$ which holds:

(ⅰ) $\tau(i)\in[1,\max\_iter]$ is a positive integer and $\tau(i+1)>\tau(i)$, $(1\le i\le M-1)$;

(ⅱ) $f(x_{\tau(i+1)})<f(x_{\tau(i)})$.

According to Convergence Conditions II as $p\in D_{k_j}^{x_j}, r_{k_j}^{x_j}\ge\lambda_0(1\le j\le\max\_iter)$, when max_iter is sufficient large, we have

$M\ge P_0\times\max\_iter>N_0+1$.

Also, from the definition of $\theta(k)$ in Convergence Conditions I, we have

$$\theta(k_{\tau(i)})=\frac{\left\|p-x_{\tau(i+1)}\right\|}{\left\|p-x_{\tau(i)}\right\|},(1\le i\le M-1).$$

Thus, $\left\|x_{\max\_iter}-p\right\|\le d_{\max}^{D}\prod_{i=1}^{M-1}\theta(k_{\tau(i)})$, where $d_{\max}^{D}$ is the maximum distance between p and any other point in region D.

According to Convergence Conditions I as well as the fact that $M-1>N_0$, we have

$$\left\|x_{\max\_iter}-p\right\|<\varepsilon' d_{\max}^{D}<\delta,$$

Where $\varepsilon' < \frac{\delta}{d_{\max}^{D}}$.

Due to the continuity condition, it can be found:

$| f(p) - f(x_{\max\_iter}) | < \varepsilon$,

Where ε is a positive however small number, which proves that $x_{\max\_iter}$ is actually the optimum solution.

2.4 The Practical Model

Because of the uncertainty of point p before optimization, values of k_1 and $c_j (1 \le j \le \max\_iter - 1)$ cannot be precisely determined so as to satisfy the conditions required in step (1) and (3) of the theoretical procedure. To avoid this difficulty, the practical procedure of DSZ algorithm is instead proposed here.

First, initialize $k_1 = 2$ so that $\forall x_1 \in D$,we have $p \in D_{k_1}^{x_1}$.[the definition of D_k^x is in section 2.1]. And $c_j = c, (1 \le j \le \max\_iter)$ is assigned in our research, where c (0<c<1) is the shrinking coefficient given in the initialization step of the algorithm, which is related to the specific optimization objective function. Unfortunately, $p \in D_{k_j}^{x_j}, (1 \le j \le \max\_iter)$ will be no longer guaranteed by doing so. Concerning to enclose p in search scope as much as possible during the process of DSZ algorithm, the practical model of DSZ algorithm is designed as bellow:

(1) Let j=1 and randomly generate m points in D to form the initial set $S_1 : \left[s_1^1, \cdots, s_i^1, \cdots, s_m^1 \right]$; initialize k_1 =2, the shrinking coefficient c (0<c<1) and the maximum iteration max_iter;

(2) For each point $s_i^j (1 \le i \le m)$ in S_j , a corresponding random point $s_i^j{}'$ is generated from $D_{k_j}^{s_i^j}$, thus a new set is formed as $S_j' : [s_1^j{}', \cdots, s_i^j{}', \cdots s_m^j{}']$;

(3) From $S_j \cup S_j'$, choose m points according to their objective function values as the new set S_{j+1} .The maximal value of these chosen points' function values should be smaller than the minimal value of the rest points' function values;

(4) Let $k_{j+1} = c \times k_j$, (0<c<1);

(5) Let j=j+1, return to step (2) until j=max_iter;

(6) Choose the solution $x0$ from $S_{\max\_iter}$ which minimizes f(x) as the output;

For certain optimization functions, if we can choose values of c , m and max_iter to hold

$$\|x0 - p\| \le 2c^{\max\_iter-1} d_{\max}^{D} \qquad (1)$$

Then $\|x0 - p\|$ will be controlled by varying $c^{\max\_iter-1}$, which meets the need to control the optimization precision.

2.5 Experimental Test

The efficiency of DSZ algorithm has been tested by using a set of well known functions. It includes eight classes of problems: Branin [2, 3, 8-14], Eason [4, 8, 9, 13], Goldstein-Price [2, 4, 8-13], Shubert [3, 4, 8, 9, 11-14], Hartmann [1, 2, 4, 8, 9, 13], Rosenbrock [1, 4, 8, 9, 12, 13], Shekel [1, 2, 4, 8-10, 12, 13], and Zakharov [1, 4, 8, 9, 13]. According to Hedar and Fukushima [8], "the characteristics of these test functions are diverse enough to cover many kinds of difficulties that arise in global optimization problems." Other three functions: rastrigin [8, 10], six hump [8, 15, 16], coville [15] have also been tested. Since the global minima is known for each of these functions, as did in [1, 4, 8, 13], we define sufficiently close by

$$|f^* - \tilde{f}| < \varepsilon_1 |f^*| + \varepsilon_2$$

Where f^* refers to the global minima of the objective function, and $\tilde{f}$ refers to the result achieved by DSZ algorithm, ε_1 and ε_2 are set to 10^{-4} and 10^{-6}, respectively. Thus for each execution of DSZ algorithm, if the above inequity is satisfied, we say DSZ algorithm is successful for this execution. Table 1 shows the results of the tests, and the function evolutions "max_iter×m" is limited within $2\text{x}10^5$. For each test function, the success ratio ρ is defined as the number of successes out of 100 independent executions. The optimization error is expressed as

$$er = \begin{cases} \dfrac{\left| f^* - \tilde{f} \right|}{|f^*|}, (f^* \neq 0) \\ \left| \tilde{f} \right|, (f^* = 0) \end{cases}$$

Med(er) in the table stands for the median of the er set that obtained from the 100 independent executions of DSZ algorithm.

Table 1 DSZ algorithm experimental results for a set of test functions

Function	Dimension n	ρ	Med(er)	max_iter	Initial sample : m	c^max_iter
Branin	2	100	1.2063e-6	60	10	10^(-4)
Rastrigin-2	2	89	1.1908e-7	500	10	10^(-4)
Easom	2	99	3.1528e-6	100	10	10^(-4)
Goldstein-price	2	100	3.0373e-7	50	10	10^(-4)
Shubert	2	99	9.8209e-7	50	10	10^(-4)
Six-hump	2	100	2.7558e-5	50	10	10^(-4)
Colville	4	94	3.2967e-8	2000	20	10^(-5)
Hartmann-3	3	84	3.0660e-7	300	10	10^(-4)
Hartmann-6	6	87	6.2395e-7	300	10	10^(-4)
Rosenbrock-2	2	96	1.1481e-9	300	20	10^(-4)
Rosenbrock-5	5	——				
Rosenbrock-10	10	——				
Skekel-(4,5)	4	74	4.3520e-7	300	10	10^(-4)
Skekel-(4,7)	4	95	3.4821e-6	300	10	10^(-4)
Skekel(4,10)	4	97	5.0862e-7	300	10	10^(-4)
Zakharov-5	5	100	8.0444e-9	300	10	10^(-5)
Zakharov-10	10	100	2.5854e-8	1000	10	10^(-5)
Zakharov-50	50	100	1.0649e-8	20000	10	10^(-6)

Next, the test results are compared with some selected existing optimization algorithms. The comparison results are demonstrated in Table 2.

From Table 1, except Rosenbrock-5 and Rosenbrock-10, DSZ algorithm has optimized the test functions with success ratio of 74% or above. It is also seen from Table 2 that DSZ algorithm is competitive with respect to optimization

success ratio. It is worth pointing out that according to the procedure of the practical model, the process of generating new set S' in the step (2) and function value calculation for S' in the step (3) can easily be achieved through the parallel computing design. In that sense, if m parallel computational components are applied, the complexity of the computational time can be reduced to $O(max\_iter)$.

Table 2 Comparison between DSZ algorithm and other optimizations

Function	Enhanced Simulated Annealing (ESA)	Monte Carlo Simulated Annealing (MCSA)	Sniffer Global Optimization (SGO)	Directed Tabu Search (DTS)	DSZ algorithm
Branin	--	100/557	100/205	100/212	100/600
Easom	——	——		82/223	99/1000
Goldstein-price	100/783	99/1186	100/664	100/230	100/500
Shubert	——	——		92/274	99/500
Hartmann-3	100/698	100/1224	99/534	100/438	84/3000
Hartmamnn-6	100/1638	62/1914	99/1760	83/1787	87/3000
Rosenbrock-2	——	——	——	100/254	96/6000
Rosenbrock-5	——	——	——	85/1684	——
Rosenbrock-10	——	——	——	85/9037	——
Shekel-(4,5)	54/1487	54/3910	90/3695	75/819	74/3000
Shekel(4,7)	54/1661	64/3421	96/2655	65/812	95/3000
Shekel(4,10)	50/1363	81/3078	95/3070	52/828	97/3000
Zakharov-5	——	——	——	100/1003	100/3000
Zakharov-10	——	——	——	100/4032	100/10000

Note: "*/*" stands for "Optimization success number/Number of function evolutions".

3 Strategy of Handling High Dimensional, Unconstrained problems

As mentioned in [5], the unconstrained and bound constrained problems in higher dimensions are one of challenges in the near future. In practice, when the optimization problems are complicated functions with high dimensional independent variables, chose parameters m, c and max_iter to meet equation (1) in section 2.4 may introduce unacceptable computational cost.

The subsection will discuss a strategy to handle high-dimensional, unconstrained problems subject to particular condition.

3.1 Condition on the High-dimensional, Unconstrained Problem

The considered objective function is denoted as $f(x)$, $x:[x_1,\cdots,x_i,\cdots,x_n]$ where $x_i(1\le i\le n)$ is real number. D is the region of the independent variables, where D: $l\le x\le u$, $(l_i\le x_i\le u_i, 1\le i\le n)$.

If $p1$, $p2$ are any two random points in D_k^x $(\forall x\in D, 0<k\le 2)$, and $f(p1)<f(p2)$, then we have $prob[|p_i-p1_i|<|p_i-p2_i|]>0.5$ for any $i(1\le i\le n)$.

3.2 Distribution of the Results

Now we study distribution of DSZ algorithm's output for each dimension of the coordinate $x0_i,(1\le i\le n)$ by the predetermined proper value of c, m and max_iter.

Assuming that parameters c, m and max_iter are chosen for acceptable computational cost, after independently executing DSZ algorithm to optimize the target function for K times, we then get a set of independent results:

$\{x0_i^k \mid 1\le i\le n, 1\le k\le K\}$, where $x0_i^k$ denotes the i^{th} coordinate of the k^{th} time execution result.

$x0_i$'s distribution in only $[p_i, u_i]$ $(1 \le i \le n)$ will be discussed here, because its distribution in $[l_i, p_i]$ is subjected to the same manner.

Let $t_i = 2c^{\max\_iter-1}(u_i - l_i)$. Evenly divide $[p_i, u_i]$ by t_i into finite subintervals. Let $q_{j'}$ ($j' \ge 1$) be the number of points from set $\left\{x0_i^k \mid 1 \le k \le K\right\}$ that falls into the j'^{th} subinterval. When the algorithm runs into the maximum iteration, the adjacent subintervals $q_{j'}$ and $q_{j'+1}$ would have the transitional relationship as shown in Figure 1.

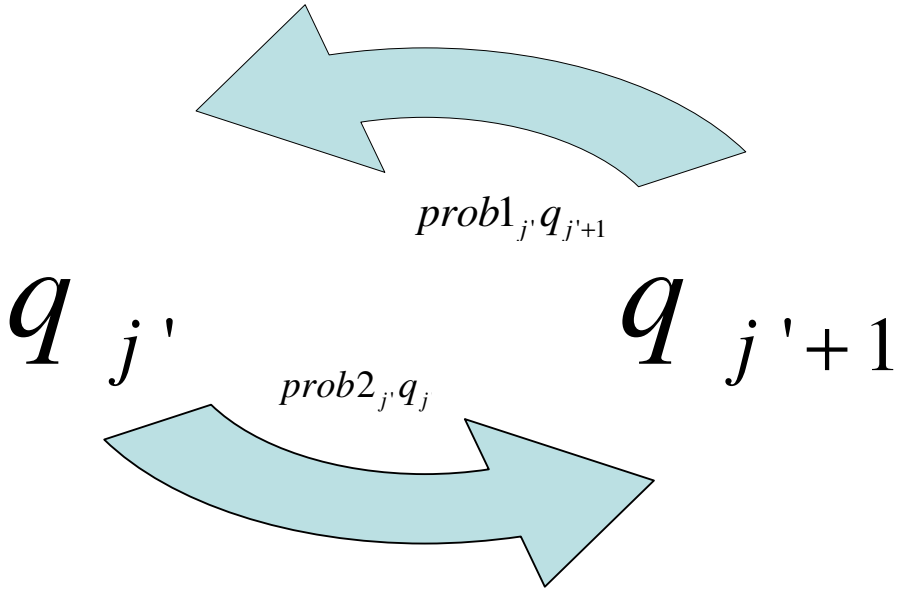

Figure 1 Transition relationship between $q_{j'}, q_{j'+1}$ at the final iteration cycle

According to the practical model, the values of set $\{S_{\max\_iter}\}$ are smaller than or equal to the ones of the set $\{S_{\max\_iter-1}\}$. According to condition in section 3.1, it gives $prob1_{j'} \ge prob2_{j'}$ because p_i lies in the first subinterval of $[p_i, u_i]$. When the transition reaches its equilibrium, we have

$$q_{j'+1} = \left(\frac{prob2_{j'}}{prob1_{j'}}\right) q_{j'}$$

Thus $q_{j'} = \prod_{i=1}^{j'-1} r_i q_1$, $j' > 1$, where $r_i = \frac{prob2_i}{prob1_i}$, $0 < r_i \le 1$. If r_i is a constant number equal to r, which is independent of the subinterval where it located, then we have:

$$q_{j'} = r^{j'-1} q_1$$

Furthermore, if $0 < r < 1$, and $c^{\max\_iter}$ is set to a sufficiently small number, which also leads to a sufficiently small t_i, the final output result in the i^{th} coordinate $x0_i$ would yield to the exponential distribution within the interval $[l_i, u_i]$. With the maximum probability occurring at the subinterval containing p_i, it has a decreasing negative exponential distribution along both sides of this subinterval.

3.3 Test Experiments

Test function in this section is $f(x) = \sum_{i=1}^{n} x_i, 0 \le x_i \le 1$. The optimization task is to determine the minimum value of $f(x)$. As to this function, condition in section 3.1 is satisfied.

In this optimization, the dimension of x is $n = 10^3$. The algorithm parameters are set as max_iter=40001, m=1 and the shrinking constant c=$10^{-3/40000}$. When the algorithm runs into the last iteration, the variable subintervals are scaled to the magnitude of $c^{\max\_iter-1} = 10^{-3}$. After all the input parameters are set up, independently running 10 times gives a 10×1000 output matrix M. Each row of M refers to the result of one independent execution of DSZ algorithm. Because of the specialty of the function $f(x)$ considered in this paper, all elements in M can be regarded as the output of any dimension $x0_i (1 \le i \le n)$ after independently running the DSZ algorithm for 10×1000 times. Hence, histograms for each dimension of the variables are shown in Figure 2 by using all the elements of M.

Figure 3 shows the plot of all the elements fell into interval [0, 0.1], and the length of its subinterval is 10^{-3}. Figure 3 indicates the majority of points fall into the region of [0, 0.02]. Thus if we zoom in this region, the corresponding result is shown in Figure 4. The logarithmic curve of Figure 4 is shown in Figure 5. The smooth trend of curve in Figure 5 discloses that the diagram in Figure 3 approximates an exponential distribution, as what has been derived in the previous section.

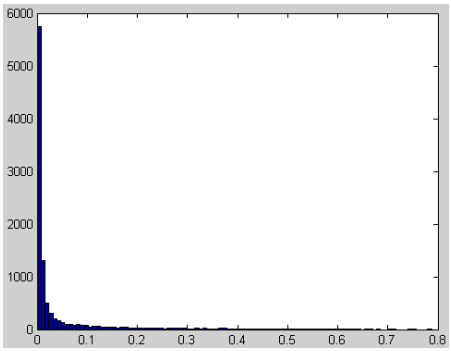

Figure 2 The histogram for all the elements of M

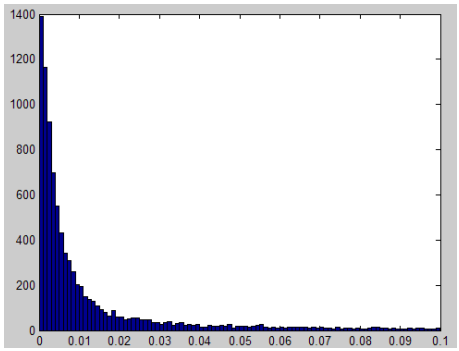

Figure 3 Histogram of all the elements fell into interval [0, 0.1]

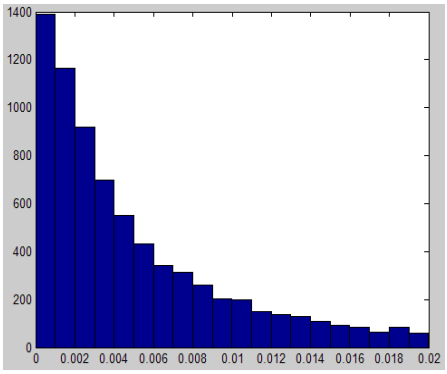

Figure 4 Histogram of all the elements fell into interval [0, 0.02]

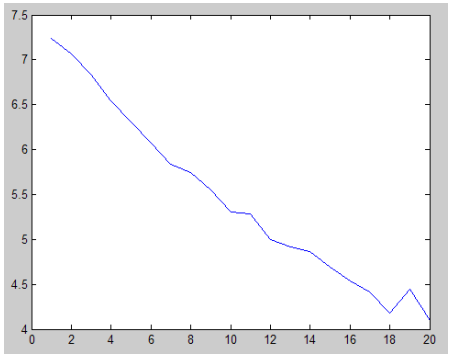

Figure 5 The logarithm curve of figure 4. x axis: subinterval number; y axis: log value of the number of points fell in each subinterval

3.4 Strategy

According to the distribution of the output results in each coordinate $x0_i (1 \le i \le n)$ of DSZ algorithm discussed above, we describe the strategy of using DSZ algorithm to optimize certain high-dimensional, unconstrained problems as following:

Initialize parameters c, m and max_iter at acceptable computational cost. Run DSZ algorithms to optimize the objective function independently for K times. Examine the histograms of the output solution set at each dimension $\left\{x0_i^k \mid 1 \le k \le K\right\}$ ($1 \le i \le n$) .Use the information histograms provide (the subinterval in which majority of set $\left\{x0_i^k \mid 1 \le k \le K\right\} (1 \le i \le n)$ locate) to narrow the interval, in which $p_i (1 \le i \le n)$ locates, hence narrow the region D_p where

p locates. Obviously, $p \in D_p \subset D$. Use D_p as the new region for the variables: $D^{new} \leftarrow D_p$. Repeat the above steps until D_p becomes sufficiently small.

4 Applications in the Constrained Optimization Context

4.1 Condition on the optimization problem

The optimization target is function $f(x)$, $x:[x_1,\cdots,x_i,\cdots,x_n]$ where $x_i(1 \le i \le n)$ is real number. Define D: $l \le x \le u$, $(l_i \le x_i \le u_i, 1 \le i \le n)$ as the region for variable x. E$\subset D$ is the constrained region on x which is nonempty. The target function $f(x)$ also satisfies the three conditions described in section 2.1, namely, (1) Continuity; (2) Convergence condition I; (3) Convergence condition II.

Without losing generality, the task is to look for the global minima of $f(x)$. The corresponding solution for the global minima is $p:[p_1,\cdots,p_{i,}\cdots,p_n],\ (1 \le i \le n)$.

4.2 Optimization Methodology

For the constrained optimization problems, DSZ algorithm employs almost the same approach as done for the unconstrained optimization.

In the constrained optimization context, firstly, randomly generate N legal points in E by evenly sampling siz points in D, the set of legal points is denoted as S_1. In this way, on the one hand, S_1 is randomly distributed in the region E; on the other hand, the ratio of volume between space E and D could be estimated by N/*siz*.

Thus, if initialize k_1 as $k_1 = \left(\frac{N}{siz}\right)^{1/n}$ (n is the dimension of variable x), the entire E space can be expected to covered by using the initial random set $\bigcup_{x \in S_1} D_{k_1}^x$.

After that, the same method of processing the unconstrained optimization is

adopted. One thing noted is that illegal points should be removed once they appear during the process.

4.3 Procedure in the Constrained Optimization Context

(1) Let j=1, and randomly generate m legal points by sampling *siz* points randomly in region D to form the initial set $S_1:\left[s_1^1,\cdots,s_i^1,\cdots,s_m^1\right]$, thus $S_1 \subset E$. Initialize $k_1=\left(\frac{m}{siz}\right)^{1/n}$, shrinking coefficient c (0<c<1), and the maximum iteration max_iter;

(2) For each point $s_i^j,(1\le i\le m)$ in S_j, a corresponding random point $s_i^j{}'$ is generated from $D_{k_j}^{s_i^j}$, thus new set is formed as $S_j{}':[s_1^j{}',\cdots,s_i^j{}',\cdots s_m^j{}']$;

(3) From set $S_j \cup S_j{}'$, choose m legal points according to their objective function values as the new set S_{j+1}. The maximal value of these chosen m points' function values should smaller than the minimal value of the rest legal points' function values;

(4) Let $k_{j+1}=c\times k_j$ (0<c<1);

(5) Let j=j+1, return step (2) until j=max_iter;

(6) Choose the solution from $S_{\max\_iter}$ which yields $f(x)$ minima as the output.

4.4 Test Results

In this section, DSZ algorithm is applied to 11 test cases from reference [6], as shown in Table 3.

Table 3 Summary of test case

Function	N	Type of f	ρ	LI	NE	NI	A
Min G1	13	Quadratic	0.0111%	9	0	0	6
Max G2	K	Nonlinear	99.8474%	0	0	2	1
Max G3	K	Polynomial	0.0000%	0	1	0	1
Min G4	5	Quadratic	52.1230%	0	0	6	2
Min G5	4	Cubic	0.0000%	2	3	0	3

Min G6	2	Cubic	0.0066%	0	0	2	2
Min G7	10	Quadratic	0.0003%	3	0	5	6
Max G8	2	Nonlinear	0.8560%	0	0	2	0
Min G9	7	Polynomial	0.5152%	0	0	4	2
Min G10	8	Linear	0.0010%	3	0	3	6
Min G11	2	Quadratic	0.0000%	0	1	0	1

Note: LI-linear inequalities, NE-nonlinear Equalities, NI-Nonlinear inequalities, A-active constraints and ρ - feasibility; for both G2 and G3, k=50.

For each test case we list number of variables n, type of the function f, the relative size of the feasible region in the search space given by ρ, the number of constraints of each categories such as LI (linear inequalities), NE (nonlinear equations) and NI (nonlinear inequalities). The feasibility ρ is determined experimentally in reference [17] by calculating the percentage of feasible solutions among the 1,000,000 randomly generated individuals.

Table 5 and Table 6 are the comparisons between DSZ algorithm and genetic algorithm [6, 7]. The independent variables of function G2 and G3 are 20 dimensions and 10 dimensions, respectively. Parameters of DSZ algorithm are shown in Table 4.

Table 4 Parameters of DSZ algorithm

Function	n	m	max_iter	$c^{\max\_iter}$
G1	13	30	1000	10^{-5}
G2	20	100	1000	10^{-3}
G3	10	20	100	10^{-3}
G4	5	20	1000	10^{-3}
G5	4			
G6	2	15	500	10^{-3}
G7	10	20	1000	10^{-3}
G8	2	10	500	10^{-3}
G9	7	20	1000	10^{-3}
G10	8	50	1000	10^{-3}

G11	2	5	20	10^{-5}

Note: n - dimensions of function's independent variables; m-the population size of the points set; max_iter- the maximum iterations; c- the shrinking coefficient.

Table 5 Test Results (1)

Function	Optimum value	GA in[6] （Experiment #3）			DSZ algorithm		
		Worst	Best	Average	Worst	Best	Average
G1	-15	-14.5732	-14.7184	-14.6478	-14.9606	-14.9999	-14.9609
G2	0.803553	0.78279	0.79486	0.78722	0.75686	0.80339	0.78671
G3	1.0	0.9960	0.9978	0.9970	1.0000	1.0000	1.0000
G4	-30665.5	-30645.6	-30661.5	-30653.1	-30664.7	-30665.1	-30665.0
G5	5126.4981						
G6	-6961.8	-6390.6	-6944.4	-6720.4	-6961.4	-6961.7	-6961.6
G7	24.306	26.182	25.090	25.545	24.702	24.401	24.493
G8	0.095828*	0.0958246	0.0958250	0.0958248	0.105459	0.105460	0.105459
G9	680.63	683.58	681.72	682.56	680.99	680.79	680.85
G10	7049.33	7685.8	7321.2	7498.6	7297.5	7053.4	7106.3
G11	0.75	0.75	0.75	0.75	0.75	0.75	0.75

Note: Independently run the genetic algorithm in [6] and DSZ algorithm 10 times, respectively; for the genetic algorithm, the maximum number of generations is 5,000, the population size is 70; for G3, k=10 and for G2, k=20. Parameters of DSZ algorithm are shown in Table 4.

Table 6 Test Results (2)

Function	Optimum value	GA in [7]			DSZ algorithm		
		Worst	Best	Median	Worst	Best	Median
G1	-15	-11.9999	-14.9999	-14.9997	-14.6051	-14.9999	-14.9950
G2	0.803553	0.672169	0.803190	0.755332	0.72074	0.80347	0.7851
G3	1.0	0.785582	1.00009	0.94899	1.0000	1.0000	1.0000
G4	-30665.5	-30652.0	-30665.5	-30663.4	-30664.7	-30665.2	-30665.0
G5	5126.4981	6112.22	5126.51	5172.53			
G6	-6961.8	-6954.32	-6961.78	-6959.57	-6961.4	-6961.8	-6961.7
G7	24.306	35.8820	24.4110	26.7357	24.914	24.340	24.448

G8	0.095828*	0.0958246	0.0958250	0.0958248	0.105459	0.105460	0.105460
G9	680.63	684.131	680.762	681.706	681.15	680.79	680.83
G10	7049.33	12097.4	7060.55	7723.17	7297.5	7053.4	7070.4
G11	0.75	0.8094	0.7490	0.7493	0.75	0.75	0.75

Note: Run the genetic algorithm in [7] and DSZ algorithm 50 times independently; for the genetic algorithm, the maximum number of generations is 5,000, the population size is 10; for G3, k=10 and for G2, k=20. Parameters of DSZ algorithm are shown in Table 4.

The test results clearly indicate that by using less than 1/5 of the iterations of genetic algorithm, DSZ algorithm is capable of determining better results than GA. It should be pointed out that DSZ algorithm has found a new solution x= [1.22780107298315, 3.74488659676571] for G8, which gives G8(x) =0.105460>0.095828. Besides, when the constraints contain equalities, DSZ algorithm handles them by transforming them into a format of inequalities or bound constrained problems. Take G3 for an example:

G3：Maximize $G3(x) = (\sqrt{n})^n \times \prod_{i=1}^{n} x_i$, where $\sum_{i=1}^{n} x_i^2 = 1, 0 \le x_i \le 1, for(1 \le i \le n)$.

G3 could be transformed into an equivalent form as following:

Maximize $G3(x) = (\sqrt{n})^n \times \prod_{i=1}^{n-1} x_i \times \sqrt{1 - \sum_{i=1}^{n-1} x_i^2}$, where $\sum_{i=1}^{n-1} x_i^2 \le 1, 0 \le x_i \le 1,$

$for(1 \le i \le n-1)$.

In this way, the n-dimensional constrained optimization problem with equalities is transformed into the (n-1)-dimensional constrained optimization problem with only inequalities.

Take G11 for another example.

G11：Maximize $G11(x) = x_1^2 + (x_2 - 1)^2$, where $x_2 - x_1^2 = 0, -1 \le x_i \le 1, (i = 1, 2)$.

G11 is equivalent to the problem below:

Maximize $G11(x) = x^2 + (x^2 - 1)^2$, where $-1 \le x \le 1$.

The above examples show that the two-dimensional problem constrained by equalities can be transformed into the one-dimensional unconstrained problem. If constraints contain multiple nonlinear equalities e.g. G5, the nonlinear equation

toolkit shall be added to current version of DSZ algorithm so as to transform equality constraints into either inequality constraints or unconstrained conditions.

5 Summary

In the present paper a simple stochastic global optimization method, DSZ algorithm, has been proposed based on Convergence Conditions I and Convergence Conditions II, which are generally ignored by previous researchers. The experimental results demonstrate that DSZ algorithm is effective and capable of handling both unconstrained and constrained optimization problems.

As to DSZ algorithm's principle, we demonstrated its validity based on its theoretical model. For the practical model of DSZ algorithm, its validity and efficiency will need further investigation when applied to more complicated objective functions. Therefore, future work will continue on:

(1) Mechanism on practical DSZ algorithm and what conditions the objective function should meet to guarantee the efficiency of its optimization;

(2) In this paper we proposed a strategy of handling the high-dimensional optimization problem by using the histogram information of DSZ algorithm outputs to narrow the optimal region for p. The efficiency of this strategy needs to be testified by more high-dimensional and complicated cases;

(3) Although the current version of DSZ algorithm is capable to handle scenarios with inequality constraints, in order to cover the equality and inequality mixed conditions, a toolkit of equation solver would need to be included;

(4) DSZ algorithm is also expected to be applied to other optimization issues, e.g. discrete variable optimization and combination optimization in future.

Acknowledgement We are thankful to Professor Wael H. Abulshohoud in Jilin University-Lambton College (previous with The George Washington University) for his insightful comments and suggestions.

6 References

[1] P. Siarry and F. Durdin, "Enhanced simulated annealing for globally minimizing functions of many-continuous variables," ACM *Transactions on Mathematical Software (TOMS),* vol. 23, pp. 209-228, 1997.

[2] D. VANDERBILT and S. LOUIE, "A Monte Carlo simulated annealing approach to optimization over continuous variables," Journal *of Computational Physics (Print),* vol. 56, pp. 259-271, 1984.

[3] R. R. Butler and E. Slaminka, "An evaluation of the sniffer global optimization algorithm using standard test functions," *Journal of Computational Physics,* vol. 99, pp. 28-32, 3. 1992.

[4] A. R. Hedar and M. Fukushima, "Hybrid Simulated Annealing and Direct Search Method for Nonlinear Unconstrained Global Optimization," Optim*. Methods Software,* vol. 17, pp. 891-912, 2002.

[5] A. Neumaier, "Complete search in continuous global optimization and constraint satisfaction," Acta *Numerica,* vol. 13, pp. 271-369, 2004.

[6] S. Koziel and Z. Michalewicz, "Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization," Evil*. Compute,* vol. 7, pp. 19-44, Spring. 1999.

[7] S. Venkatraman and G. Yen, "A Generic Framework for Constrained Optimization Using Genetic Algorithms," Evolutionary *Computation, IEEE Transactions on,* vol. 9, pp. 424-435, 2005.

[8] A. R. Hedar and M. Fukushima, "Tabu Search directed by direct search methods for nonlinear global optimization star, open," Eur. *J. Oper. Res.,* vol. 170, pp. 329-349, 2006.

[9] M. J. Hirsch, C. N. Meneses, P. M. Pardalos and M. G. C. Resende, "Global optimization by continuous grasp," Optimization *Letters,* vol. 1, pp. 201-212, 2007.

[10] J. Barhen, "TRUST: A Deterministic Algorithm for Global Optimization," Science*,* vol. 276, pp. 1094-1097, 1997.

[11] D. Cvijovic and J. Klinowski, "Taboo Search: An Approach to the Multiple Minima Problem," Science*,* vol. 267, pp. 664-666, 1995.

[12] T. B. Trafalis and S. Kasap, "A novel metaheuristics approach for continuous global optimization," J. *Global Optimiz.,* vol. 23, pp. 171-190, 2002.

[13] A. Hedar and M. Fukushima, "Minimizing multimodal functions by simplex coding genetic algorithm," Optim*. Methods Software,* vol. 18, pp. 265–282, 2003.

[14] G. H. Koon and A. V. Sebald, "Some interesting test functions for evaluating evolutionary programming strategy," *Proceedings of the Fourth Annual Conference on Evolutionary Programming,* pp. 479–499, 1995.

[15] Z. Michalewicz, *Genetic Algorithms Data Structures= Evolution Programs.* Springer, 1996,

[16] R. Chelouah and P. Siarry, "A Continuous Genetic Algorithm Designed for the Global Optimization of Multimodal Functions," J. *Heuristics,* vol. 6, pp. 191-213, 2000.

[17] Z. Michalewicz and G. Nazhiyath, "Genocop III: a co-evolutionary algorithm for numerical optimizationproblems with nonlinear constraints," Evolutionary *Computation, 1995, IEEE International Conference on,* vol. 2.