基于模拟退火的 CRS 算法

汤 丹 [1]

摘要 对非线性规划问题提出了一种算法，该算法把模拟退火算法应用到 CRS 算法中，根据模拟退火算法每一次迭代都体现集中和扩散两个策略的平衡的特点，使 CRS 算法更能够搜索到全局最优解，而不会陷入局部最优解. 最后把提出的算法应用到两个典型的函数优化问题中.

关键词 CRS 算法，模拟退火算法，全局优化，接受概率

中图分类号 O221

数学分类号 90C70, 90C05

CRS Algorithm Based on the Simulated Annealing

TANG Dan[1]

Abstract In this paper, a new algorithm is proposed for the nonlinear programming problems. The algorithm applied the simulated annealing to the CRS algorithm. According to the simulated annealing reflects on concentrates and proliferates in each iteration. Enables the CRS algorithm to search the global optimization more easier rather than the local optimization. Finally, the proposed algorithm is applied to two typical function optimization problems, and the numerical results illustrate the accuracy and efficiency of the algorithm.

Keywords CRS algorithm, simulated annealing, global optimization, accept probability

Chinese Library Classification O223

2010 Mathematics Subject Classification 90C70, 90C05

0 引 言

我们考虑如下的全局优化问题

$$\begin{array}{l}\min\ f(x)\\ \text{s.t. } x\in S\end{array} \tag{1}$$

其中 $x\in R^n$, $f(x):R^n\to R$ 是实函数， $S=\{x\in R^n: a_i\leqslant x_i\leqslant b_i, i=1,2,\cdots,n\}$, a_i 和 b_i 为变量 x 的第 i 个分量的下界和上界. 针对问题 (1) 的求解已有许多算法，如遗传算法、粒子群算法和模拟退火算法等，本文主要利用 CRS 算法求解.

CRS 算法是一种直接搜索启发式算法，它最初是由 Price 于 1978 年提出的. 目前为止，关于 CRS 算法的算法有六种 [1−5], Nelson 等人提出了一个改进的 CRS 算法 [6], 并应

收稿日期： 2011 年 6 月 8 日.

1. 华侨大学经济与金融学院，福建泉州 362001; College of Economics and Finance, Huaqiao University, Quanzhou Fujian 362001, China

用到飞机的翼栅设计中. CRS 算法具有操作简单，易于程序设计等特点，但是在应用中容易产生局部最优解，因此我们提出了一种基于模拟退火的 CRS 算法.

本文所提出的改进的基本思想是：在产生新的测试点并计算其函数值，如果它比最坏点的函数值还大，那么根据模拟退火算法，计算其接受概率 Q, 若 $\mathrm{rand}(0,1) \leqslant Q$, 则接受这个测试点及其函数值并替换最坏点及其函数值，然后更新温度，重新选择新的测试点. 从而提高了算法搜索到全局最优值的能力. 本文只以 CRS 算法中具有代表性的 CRS2 算法为例进行研究，对于其它版本的 CRS 算法也可类似研究. 最后把改进的 CRS 算法应用到两个典型的函数优化问题中，数值结果表明，改进的 CRS 算法是可行的、有效的.

1 CRS2 算法的描述

CRS2 算法描述如下：

Step 1 从解空间 S 中随机选取 N 个样本点，生成群体 $P = \{x_1, x_2, \cdots, x_N\}$, 计算每个点的函数值.

Step 2 从 P 中确定最好点 x^{best} 和最坏点 x^{bad} 及它们的目标函数值 f_{best}, f_{bad}, 使得

$$f_{\mathrm{best}} = \min_{x \in P} f(x), \quad f_{\mathrm{bad}} = \max_{x \in P} f(x).$$

若算法终止条件满足 $(f_{\max} - f_{\min} < 10^{-6})$, 则停止.

Step 3 从 P 中随机选取 n 个点 $R_2, R_3, \cdots, R_{n+1}$, 计算中心

$$G = \frac{1}{n}(x_{\mathrm{best}} + \sum_{j=2}^{n} R_j)$$

及试验点

$$\tilde{x} = 2G - R_{n+1}.$$

若 $\tilde{x} \notin S$ ，则转 Step 3 ，否则计算 $f(\tilde{x})$, 若 $f(\tilde{x}) > f_{\max}$ ，则转 Step 3.

Step 4 令 $P = P \cup \{\tilde{x}\} \backslash \{x_{\mathrm{bad}}\}$, 转 Step 2.

2 算 法

3.1 接受概率

在内循环中，新的测试点 $\tilde{x}$ 产生后，并计算出其函数值 $f(\tilde{x})$. 接受它的概率为：

$$Q = \exp(-(f(\tilde{x}) - f_{\max}))/T \tag{2}$$

其中 T 为在产生新的测试点 $\tilde{x}$ 时的温度，接受概率与温度之间成反比.

3.2 算法的描述

本文所提出的算法描述如下：

Step 1 从解空间 S 中随机选取 N 个样本点，生成群体 $P = \{x_1, x_2, \cdots, x_N\}$, 计算每个点的函数值，确定初始温度 T.

Step 2 若在该温度达到内循环停止条件，转 Step 5, 从 P 中确定最好点 x^{best} 和最坏点 x^{bad} 及它们的目标函数值 f_{best}, f_{bad}, 使得

$$f_{\text{best}} = \min_{x\in P}\ f(x), \quad f_{\text{bad}} = \max_{x\in P}\ f(x).$$

若算法终止条件满足 ($f_{\max} - f_{\min} < 10^{-6}$), 则停止.

Step 3 从 P 中随机选取 n 个点 $R_2, R_3, \cdots, R_{n+1}$ ，计算中心

$$G = \frac{1}{n}\left(x_{\text{best}} + \sum_{j=2}^{n} R_j\right)$$

及试验点

$$\tilde{x} = 2G - R_{n+1}.$$

若 $\tilde{x} \notin S$ ，则转 Step 3, 否则计算 $f(\tilde{x})$, 若 $f(\tilde{x}) < f_{\max}$, 则转 Step 5.

Step 4 若 $f(\tilde{x}) < f_{\max}$, 按公式 (2) ，计算 $\tilde{x}$ 的接受概率，若 $\text{rand}(0,1) > Q$, 则转 Step 6.

Step 5 令 $P = P \cup \{\tilde{x}\}\backslash\{x_{\text{bad}}\}$.

Step 6 $T = \alpha T$, 转 Step 2.

其中 $\text{rand}(0,1)$ 是 $0 \sim 1$ 的随机数， $\alpha \in (0,1)$.

3.3 算法参数说明

样本点个数 N 的选取： N 的值越大，越能够进行全局搜索，因此找到全局最优值的可能性越高. 另一方面， N 的值越大，要求计算机的存储量越大，因此算法的收敛速度越慢，对于许多的目标函数，选取一个较小的 N 值就能够找到全局最优值. 但是，在某些情况下，用所给的 N 值找到的值不是全局最优值，而是局部最优值. 在一般的情况下，在 CRS2 中， N 的取值为 $N = 10\ (n+1)$, 其中 n 为变量的维数。

$\alpha \in (0,1)$ 且其大小可以不断变化，根据应用者的要求自定义.

3 数值实验

为了验证算法的可行性和有效性，现选择 2 个典型函数优化问题进行测试：

例 1[7] Shekel Function(SQRIN)

$$\begin{aligned}&\min\ f(x) = -\sum_{i=1}^{m}\left[(x-\alpha_i)(x-\alpha_i)^{\mathrm{T}} + c_i\right]^{-1}\\ &\text{s.t. } 0 \leqslant x_j \leqslant 10,\ j = 1,2,\cdots,n\end{aligned}$$

其中 α_i 和 c_i 的值参见表 1.

其最优解和最优值是 $\min(f(x_{loc-opt})) \approx f(\alpha_i) \approx 1/c_i,\ \ 1 \leqslant i \leqslant m.$

例 2[7] Hartman Function

$$\begin{aligned}&\min\ f(x) = -\sum_{i=1}^{m} c_i \exp\left(\sum_{j=1}^{n}\alpha_{ij}(x_j - p_{ij})^2\right)\\ &\text{s.t. } 0 \leqslant x_j \leqslant 1,\ j = 1,2,\cdots,n\end{aligned}$$

其中 p_i 是第 i 个局部最优值的近似位置， α_i 是在第 i 个局部最优值与 Hessian 矩阵的特征值成比例的值， c_i >0 是第 i 个局部最优值的高度 (深度).

当 n=3,6 时， p_i ， α_i ， c_i 的取值参见表 2 和表 3.

表 1

i	α_i				c_i
1	4	4	4	4	0.1
2	1	1	1	1	0.2
3	8	8	8	8	0.2
4	6	6	6	6	0.4
5	3	7	3	7	0.4
6	2	9	2	9	0.6
7	5	5	3	3	0.3
8	8	1	8	1	0.7
9	6	2	6	2	0.5
10	7	3.6	7	3.6	0.5

表 2 $m = 4, n = 3$ 时的 p_i， α_i， c_i 的取值

i	α_i			c_i	p_i		
1	3	10	30	1	0.3689	0.117	0.2673
2	0.1	10	35	1.2	0.4699	0.4387	0.747
3	3	10	30	3	0.1091	0.8732	0.5547
4	0.1	10	35	3.2	0.03815	0.5743	0.8828

当 $n = 3$ 时最优解和最优值是：

$f(0.114, 0.556, 0.852) = -3.86.$

表 3 $m = 4, n = 6$ 时的 p_i， α_i， c_i 的取值

i	α_i						c_i	p_i					
1	10	3	17	3.5	1.7	8	1	0.1312	0.1696	0.5569	0.0124	0.8283	0.5886
2	0.05	10	17	0.1	8	14	1.2	0.2329	0.4135	0.8307	0.3736	0.1004	0.9991
3	3	3.5	1.7	10	17	8	3	0.2348	0.1451	0.3522	0.2883	0.3047	0.6650
4	17	8	0.05	10	0.1	14	3.2	0.4047	0.8828	0.8732	0.5743	0.1091	0.0381

当 $n = 6$ 时最优解和最优值是： $f(0.201, 0.15, 0.447, 0.275, 0.311, 0.657) = -3.32.$

对于例题 1 的 $m = 5, 7, 10$ 的三种情况, 分别应用本文算法和 CRS2 算法进行计算 5 次， $m = 5, 7$ 时均有一次搜索不到全局最优值和最优解，本文的算法与 CRS2 算法计算结果比较分别见表 4 、表 5. 对于例题 2 的 $n = 3, 6$ 两种情况，分别用本文算法和 CRS2 算法进行计算 5 次， $n = 6$ 时有一次搜索不到全局最优值和最优解，本文的算法与 CRS2 算法计算结果比较分别见表 6 、表 7.

表 4 $m = 5$ 时，本文算法与算法计算结果的比较

CRS 算法		本文算法	
最优解	最优值	最优解	最优值
(3.9999,4.0001,3.9999,4.0001)	−10.1531	(3.9999,4.0001,4.0000,4.0001)	−10.1531
(0.9999,1.0002,1.0000,1.0001)	−5.05551	(4.0000,4.0001,4.0000,4.0000)	−10.1531
(4.0000,4.0002,4.0000,4.0001)	−10.1531	(4.0000,4.0001,3.9999,4.0001)	−10.1532
(3.9999,4.0002,4.0000,4.0001)	−10.1532	(4.0000,4.0002,4.0000,4.0002)	−10.1531
(4.0000,4.0000,4.0000,4.0002)	−10.1531	(4.0001,4.0000,3.9999,4.0001)	−10.1532

表 5 $m = 7$ 时，本文算法与算法计算结果的比较

CRS 算法		本文算法	
最优解	最优值	最优解	最优值
(4.0004,4.0007,3.9995,3.9995)	−10.4029	(4.0006,4.0007,3.9995,3.9995)	−10.4029
(4.0006,4.0006,3.9995,3.9996)	−10.4029	(4.0005,4.0007,3.9994,3.9996)	−10.4029
(3.0007,7.0008,3.0002,7.0005)	−2.7658	(4.0005,4.0006,3.9994,3.9995)	−10.4029
(4.0005,4.0007,3.9994,3.9996)	−10.4029	(4.0005,4.0006,3.9995,3.9996)	−10.4029
(4.0006,4.0006,3.9994,3.9995)	−10.4029	(4.0006,4.0007,3.9995,3.9996)	−10.4029

表 6 $n = 6$ 时 CRS 算法计算结果

最优解	最优值
(0.2013,0.1467,0.4764,0.2754,0.3116,0.6574)	−3.3219
(0.1074,1.1328,0.8312,0.0312,0.3156,0.6254)	−2.6523
(0.2017,0.1502,0.4770,0.2753,0.3115,0.6572)	−3.3222
(0.2018,0.1500,0.4766,0.2754,0.3115,0.6572)	−3.3222
(0.2018,0.1500,0.4767,0.2753,0.3116,0.6572)	−3.3222

表 7 $n = 6$ 时本文算法计算结果

最优解	最优值
(0.2018,0.1497,0.4769,0.2752,0.3117,0.6573)	−3.3222
(0.2013,0.1503,0.4769,0.2753,0.3115,0.6272)	−3.3222
(0.2013,0.1500,0.4767,0.2752,0.3116,0.6573)	−3.3222
(0.2015,0.1498,0.4766,0.2753,0.3115,0.6573)	−3.3222
(0.2015,0.1500,0.4769,0.2754,0.3116,0.6571)	−3.3222

从以上的表中，我们可以看出，改进的算法比 CRS2 算法更能准确地搜索到全局最优解和最优值.

4 结 束 语

为了提高算法的搜索到全局最优值的能力，我们给出了基于模拟退火的 CRS2 算法，该算法主要是以一定的概率接受测试点，使之不会陷入局部最优解. 数值结果表明改进的算法是可行的、有效的.

参 考 文 献

[1] Price W L. A Controlled Random Search Procedure for Global Optimization, in Towards Global Optimization 2 [M]. North-Holland Publishing Company, Amsterdam, Holland, 1978.
[2] Price W L. Global Optimization by Controlled Random Search [J]. *Journal of Optimization Theory and Applications*, 1983, **40**: 333-348.
[3] Price W L. Global Optimization Algorithms for a CAD Worksstation [J]. *Journal of Optimization Theory and Application*, 1987, **55**: 133-146.
[4] Ali M M, Storey C. Modified Controlled Random Search algorithms [J]. *International Journal of Computer Mathematics*, 1994, **53**: 229-235.
[5] Ali M M, TÖrn A, Viitanen S. A Numerical Comparison of Some Modified Controlled Random Search Algorithms [J]. *Journal of Global Optimization*, 1997, **11**: 377-385.
[6] Nelson M F. An Improved Controlled Random Search Algorithm for Inverse Airfoil Cascade Design [C]. The 6^{th} World Congresses of Structural and Multidisciplinary Optimization, International Society for Structural and Multidisciplinary Optimization, Brazil, 2005.
[7] 王凌. 智能优化算法及其应用 [M]. 北京：清华大学出版社， 2003.