分区交叉差分进化算法及其约束优化

刘荣辉[1,2] 郑建国[1]

(东华大学管理学院 上海 200051)[1] (河南城建学院计算机科学与工程系 平顶山 467036)[2]

摘 要 差分进化算法处理复杂高维优化问题时存在收敛速度慢和精度不高的缺陷,为此提出了分区交叉差分进化算法。利用柯西分布随机数设计两个动态算子,分别生成缩放因子和交叉因子用于进化中,并对进化进行合理的分区,不同区段根据不同的配置利用算子生成相应的交叉因子。同时为了加快收敛速度,采用了新的变异策略,对寻优的方向加以引导。对经典Benchmark函数进行了仿真测试,结果显示,本算法的收敛速度与优化准确率均有显著提高。同时提供了算法处理约束问题的解决方案,并检验了方案的可行性。

关键词 差分进化,柯西随机数,分区交叉,参数控制,约束优化

中图法分类号 TP18 **文献标识码** A

Subarea Crossover Differential Evolution Algorithm and its Constrained Optimization

LIU Rong-hui[1,2] ZHENG Jian-guo[1]

(School of Management, Donghua University, Shanghai 200051, China)[1]

(Department of Computer Science and Engineering, Henan University of Urban Construction, Pingdingshan 467036, China)[2]

Abstract Differential evolution algorithm in solving complex functions with high dimensions has shown some weaknesses, such as low convergence speed and low precision. Therefore a new self-adapting differential evolution algorithm with subarea crossover was proposed. Two operators based on Cauchy distribution random number were adapted and their associated control parameter values were gradually self-adapted in this algorithm. The evolution process was divided into two subareas with different configuration values for operators. At the same time, to enhance the convergent speed, a new mutation strategy was introduced to guide searching direction. Benchmark problems were used to verify this algorithm. The result of simulation indicates that it is improved significantly both in convergence speed and precision of optimization. The algorithm also provides a solution to deal with the constrained optimization problems and the feasibility is also verified by two examples.

Keywords Differential evolution, Cauchy random number, Subarea crossover, Parameter control, Constrained optimization

1 引言

差分进化算法(DE)[1]是采用实数编码的基于群体差异实现全局优化的一种较新算法,其针对高维、非线性及不可微等优化问题时表现出极强的生命力,在不同领域得到了广泛的应用[2-4]。但是,DE算法也存在处理某些复杂高维多峰值函数时收敛速度过慢、收敛精度不高的缺陷。为了改善DE的优化性能,国内外学者针对参数设计了不同自适应变异策略[5-8]。虽然这些方法在一定程度上提高了算法的性能,但复杂高维问题上仍易陷入局部最优,收敛精度的问题依然不能得到满意的解决。

为此本文引入分区交叉进化策略,有针对性地设计了两个算子,并对进化过程进行合理分区,每个区段利用算子采用不同的配置生成交叉因子,同时根据配置参数采用算子生成动态缩放因子。为了增加寻优的方向性,采用了一种加强局部寻优的变异策略。对典型测试函数进行了优化实验,结果显示分区交叉进化策略在提高算法的收敛速度和精度方面拥有一定的优势。

2 差分进化算法

设 $X_{i,G}=(x_{i,G}^1,x_{i,G}^2,\cdots,x_{i,G}^D)$, $i=1,2,\cdots,NP$ 是种群中一个 D 维向量(又称为个体),代表 D 维的优化参数向量,是 DE 中第 G 代的第 i 个体,NP 代表种群大小[1,6]。

2.1 种群初始化

第0代种群的每个个体 $X_{i,0}$ 中每个优化参数在指定的 $[X_{\text{low}},X_{\text{up}}]$ 值域内随机地产生。

$$x_{i,0}^j=x_{\text{low}}^j+rand(x_{\text{up}}^j-x_{\text{low}}^j),1\leqslant j\leqslant D,1\leqslant i\leqslant NP \quad (1)$$

式中,$rand$ 代表在[0,1]间产生的随机数,$x_{i,0}^j$ 表示第 i 个个体的第 j 维分量,初始化种群随机生成 NP 个个体。

2.2 变异操作(Mutation)

存在第 G 代的目标个体 $X_{i,G}$ 和变异个体 $V_{i,G}=(v_{iG}^1,v_{iG}^2,\cdots,v_{iG}^D)$,以一定的变异方式进行变异。变异的策略可以有多

到稿日期:2011-03-23 返修日期:2011-06-03 本文受国家自然科学基金(70971020)资助。

刘荣辉(1972—),博士生,讲师,主要研究方向为智能数据处理和数据挖掘,E-mail:liurh_126@126.com;郑建国(1962—),教授,博士生导师,主要研究方向为智能决策与数据挖掘、智能信息处理等。

种[6-8]，这里列出了两种常用的方式：

1) DE/rand/1/bin

$$V_{i,G}=X_{a,G}+F.(X_{b,G}-X_{c,G}) \tag{2}$$

2) DE/best/2/bin

$$V_{i,G}=X_{best,G}+F.(X_{a,G}-X_{b,G})+F.(X_{c,G}-X_{d,G}) \tag{3}$$

式中，$a,b,c,d\in\{1,\cdots,NP\}$，是伴随目标个体 $X_{i,G}$ 产生的不同于 i 的互不相等的随机整数。$X_{best,G}$ 为 G 代最优个体，而 $F\in[0,1]$ 通常是一个实数，被称为缩放因子，用来控制差分向量的缩放。

2.3 交叉操作(Crossover)

变异个体 $V_{i,G}$ 和目标个体 $X_{i,G}$ 由下面的方案[1]进行交叉，产生候选个体：

$$u_{i,G}^{j}=\begin{cases}v_{i,G}^{j}, & rand_j\leqslant CR \text{ or } j=randn_i\\ x_{i,G}^{j}, & \text{otherwise}\end{cases} \tag{4}$$

式中，$rand_j$ 是位于[0,1]均匀分布的随机实数，而 $randn_i$ 是属于{1,2,…,D}随机产生的维数索引号，其保证了至少一位由变异向量贡献。交叉概率因子 CR 是个常数，是位于[0,1]内的一个实数。

2.4 选择操作(Selection)

选择操作采用的是“贪婪”策略[1]，经变异以及交叉操作后生成的候选个体 $U_{i,G}$ 与目标个体 $X_{i,G}$ 进行竞争：

$$X_{i,G+1}=\begin{cases}U_{i,G}, & f(U_{i,G})>f(X_{i,G})\\ X_{i,G}, & f(U_{i,G})\leqslant f(X_{i,G})\end{cases} \tag{5}$$

式中，f 是适应度函数。在 $U_{i,G}$ 和 $X_{i,G}$ 中选择适应度值最优作为第 $G+1$ 代个体，替换原目标个体。需要指出的是，式(5)适合处理最小化问题。本文差分进化算法(DE)是指采用式(2)、式(4)和式(5)构成的算法[1,6-8]。

3 分区交叉 DE 算法

候选个体的产生与控制参数 F 和 CR 密切相关，其策略的优劣对算法的性能有较大的影响。对上述式(2)、式(3)选择的不同，会对算法产生不同的影响。DE/rand/1/bin 采用得最为广泛[6-8]，能够很好保持种群多样性，利于收敛于全局最优。采用式(3)则拥有较快的收敛速度[7]，但种群差异性缺失严重，易陷入局部最优。为此本文提出上述二者的变种：一种既能保持种群多样性，又拥有一定收敛速度的改进变异策略。

3.1 改进的变异策略

改进的变异策略也可以拥有/1/bin 和/2/bin 不同的形式：

$$V_{i,G+1}=X_{rand\text{-}best,G}+F.(X_{a,G}-X_{b,G}) \tag{6}$$

$$V_{i,G+1}=X_{rand\text{-}best,G}+F.(X_{a,G}-X_{b,G})+F.(X_{c,G}-X_{d,G}) \tag{7}$$

$$V_{i,G+1}=X_{rand\text{-}best,G}+F.(X_{a,G}-X_{b,G})+F.(X_{c,G}-X_{c,G}) \tag{8}$$

式中，a,b,c 和 d 分别是从当前种群中随机选取的 m 个不同随机变量中互不相同个体的编号。$X_{rand\text{-}best,G}$ 是 m 个个体中函数适应度值最优的个体。F 是缩放因子，控制差分的缩放程度。这样式(6)—式(8)不仅能保持一定的速度，还能保持种群的多样性。经过试验检验，式(6)—式(8)中 m 位于[3,1.5 * NP]范围之内为宜。式(6)、式(8)效果要优于式(7)，本文试验采用效果更好的式(8)。

3.2 参数控制

柯西分布和正态分布具有相似性，柯西分布也是统计学中应用较为广泛的分布。在垂直方向上，和正态分布相比，柯西分布更接近垂直轴；而在水平方向，柯西分布并不比高斯分布更接近水平轴。与正态分布相比，虽然其较小的中心部分是一个显著的缺点，但可以发挥其两翼性的特长，加大寻优的范围，从而更易摆脱局部最优点的干扰。

为了打破进化的停滞状态，跳出局部最优点，扩大寻优范围，在此利用柯西分布随机数来生成如下算子：

$$F=cauchyrand(uF,0.05),rand_1<0.1 \tag{9}$$

$$CR=cauchyrand(uCR,0.05),rand_2<0.1 \tag{10}$$

式(9)是 F 的生成算子。其中 $rand_1$ 是位于[0,1]之间的随机数，$cauchyrand$ 代表柯西分布随机数函数，0.05 代表偏差。同样式(10)是产生 CR 的算子，$rand_2$ 也是位于[0,1]之间的随机数。同时把 F 和 CR 的取值范围控制在[0,1]之间。

3.3 算法实现描述

分区交叉差分进化(SCDE)算法伪码如下：

```
Step1  设定种群数 NP、维数 D 以及分区相关参数 uF,uCR1,uCR2,
       u 的初始化设置。初始化 m=0.1 * NP。变量的指定搜索空
       间范围为[X_low, X_up],指定一个最大的迭代次数 Gen,迭代计
       数器 G=1;
Step2  采用式(1)随机产生 NP 个体;
Step3  While G ≦ Gen //或者其他指定条件
Step4    For i 从 1 到 NP
  Step4.1  从当前种群中随机选择 m 个个体;
  Step4.2  根据式(9)利用 uF 生成 F;
  Step4.3  IF G ≦ u * Gen then//分区交叉处理
             根据式(10)利用 uCR1 生成 CR;
           ELSE
             根据式(10)利用 uCR2 生成 CR;
           End IF //分区交叉处理
  Step4.4  利用生成的 F 根据式(8)进行变异操作;
  Step4.5  根据选中的 CR 用式(4)进行交叉操作;
  Step4.6  判断越界,则在指定区域内随机生成新个体;
  Step4.7  用式(5)进行选择操作;
  Step4.8  End For;
Step5  迭代计数器 G 加 1;
Step6  End While。
```

3.4 算法收敛性分析

Solis 等[9]针对随机优化算法提出了以概率 1 收敛于全局最优解的充分条件。为了便于分析，下面对其主要结论进行重述。

假设 1　设 $f(X)$ 都是最小优化函数，则要求算法在进化过程父代和子代形成一个单调非递增序列，也即 $f_{G+1}^{*}\leqslant f_{G}^{*}$，其中 $G\geqslant 0$ 表示进化的代数，$f_G^*=f(X_G)$ 表示 G 代种群的最优解，$X_G\in U$ 是 G 代种群的序列，U 为变量可行域。

假设 2　对于 U 的任意 Borel 子集 A 有 $\vee(A)>0$，则有 $\prod_{G=0}^{\infty}(1-\mu_G(A))=0$ 成立，$\vee(A)$ 为子集 A 的 NP 维闭包，$\mu_G(A)$ 为相关采样策略得到 A 中部分元素的概率。

定理 1　设目标函数 f 为可测函数，同时 U 为可测子集，$\{X_G,G\geqslant 0\}$ 为算法生成的种群序列，并且同时满足假设 1 和假设 2，则 $\lim_{G\to\infty}\{f(X^*)=f^*\cap X^*\in X_G\}=1$ 成立，算法全局收敛于最优解 f^*，其中 X^* 为全局最优解的集合。

证明：由定理 1 可知，只需证明 SCDE 满足假设 1 和假设

2 即可。

1) SCDE 还是采用标准 DE 的选择操作式(4),由于其采用"贪婪"的选择策略,显然满足假设 1。

2) 设 $X_{i,G}$ 为第 i 个个体经过 G 代进化后的结果,由式(5)—式(7)进行变异,两次变异 $X_{i,G}$ 和 $X_{i,G+1}$ 的间隔代数是有限的,并且两次变异内个体中分量的更新相互之间没有任何影响。由于 $X_{i,G}$ 是基于变异策略随机产生的,同时算法保证了 $X_{i,G}$ 越界后在定义域内随机产生一个,则由文献[13]中结论可知,NP 个个体的样本空间的并必然包含 U,$U\subseteq\bigcup_{i=1}^{NP}M_{i,G}$,其中 $M_{i,G}$ 为 G 代个体 $X_{i,G}$ 的样本空间的支撑集。对于给定 U 的任意 Borel 子集 A,只要存在 $V(A)>0$,即 A 和其导集 A' 的并集为非空真子集,则可得 $V(A)\subseteq U$,种群内总能搜索到一个最小域 δ 完全覆盖 A,故 $\prod_{G=0}^{\infty}(1-\mu_G(A))=0$,假设 2 得证。

4 实验与分析

为了说明本算法的有效性,和比较著名自适应动态差分算法 jDE[6]、SaDE[8] 以及标准差分算法 DE(rand/1/bin),通过利用上述文献实验中采用的函数进行实验并加以比较分析。实验结果受 Matlab 精度所限,1E-324 即表示为"0"(实验平台 Matlab 7.5,当小于这个值时显示 0)。

比较对象采用原文献配置:jDE($\tau_1=\tau_2=0.1$,$F_l=0.1$,$F_u=0.9$),Sade($LP=50$);基本差分算法 DE 的参数设置:$F=0.5$,$CR=0.9$;SCDE 设置:$u=0.4$,$uF=0.5$;$uCR1=0.1$;$uCR2=0.7$。

4.1 测试函数

限于篇幅,本实验选用几个经典 Benchmark 进行实验比较。Benchmark 函数(f_1-f_6)用于 10 维(10-D)、30 维(30-D)和 100 维(100-D)实验,具体的函数定义[6,8]如下:

1)Sphere 函数

$f_1=\sum_{i=1}^{D}x_i^2$,取值范围为$[-100,100]^D$,最小值为 0,对应极值点为(0,…,0)。

2)Schwefel2.22 函数

$f_2=\sum_{i=1}^{D}|x_i|+\prod_{i=1}^{D}|x_i|$,取值范围为$[-10,10]^D$,最小值为 0,对应极值点为(0,…,0)。

3)Rosenbrock 函数

$f_3=\sum_{i=1}^{D}(100(x_{i+1}-x_i^2)+(x_i-1)^2)$,取值范围为$[-30,30]^D$,最小值为 0,对应极值点为(1,…,1)。

4)Rastrigrin 函数

$f_4=\sum_{i=1}^{D}(x_i^2-10\cos(2\pi x_i)+10)$,取值范围为$[-5.12,5.12]^D$,最小值为 0,对应极值为(0,…,0)。

5)Generalized Penalized 函数

$$f_5(x)=\frac{\pi}{D}\{10\sin^2(\pi y_i)+\sum_{i=1}^{D-1}(y_i-1)^2\cdot[1+10\sin^2(\pi y_{i+1})]+(y_D-1)\}+\sum_{i=1}^{30}u(x_i,10,100,4)$$

取值范围为$[-50,50]^D$,最小值为 0,对应极值点为(−1,…,−1)。

$$u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m, & x_i>a,\\ 0, & -a\leqslant x_i\leqslant a\\ k(-x_i-a)^m, & x_i<-a\end{cases}$$

$y_i=1+\frac{1}{4}(x_i+1)$

6)Griewank 函数

$f_6=\frac{1}{4000}\sum_{i=1}^{D}(x_i)^2-\prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$,取值范围为$[-600,600]^D$,最小值为 0,对应极值点为(0,…,0)。

4.2 30 维及 100 维 Benchmark 问题效率实验及分析

为了比较的公平性,根据文献[6,8]指定迭代次数(见表 1、表 2),统一在 Matlab7.5 平台下做实验。针对表 1 和表 2 分别设置维数 $D=30$,种群 $NP=100$;维数 $D=100$,种群 $NP=400$。

表 1 30-D Benchmark 函数 SCDE 和其它 DE 效率实验结果比较

F	Gen	SCDE Mean (Std Dev)	SaDE Mean (Std Dev)	jDE Mean (Std Dev)	DE(rand/1) Mean (Std Dev)
f_1	1500	**4.6E-78** (**3.5E-78**)	8.3E-22 (2.3E-21)^t	5.5E-20 (5.6E-20)^t	4.6E-16 (1.5E-16)^t
f_2	2000	**1.5E-57** (**1.2E-57**)	7.2E-13 (3.7E-13)^t	8.4E-17 (4.1E-17)^t	4.7E-11 (3.0E-11)^t
f_3	20000	**8.0E-28** (**1.3E-28**)	4.9E-17 (2.3E-16)^t	8.0E-02 (5.6E-01)^t	8.0E-02 (5.6E-01)^t
f_4	5000	**0.0E+00** (**0.0E+00**)	9.5E+01 (1.4E+01)^t	0.0E+00 (0.0E+00)^t	8.4E+01 (9.3E+00)^t
f_5	1500	**1.6E-32** (**6.7E-48**)	8.6E-22 (3.5E-21)^t	4.2E-21 (2.6E-21)^t	5.4E-17 (7.8E-18)^t
f_6	3000	**0.0E+00** (**0.0E+00**)	0.0E+00 (0.0E+0)^t	0.0E+00 (0.0E+00)^t	0.0E+00 (0.0E+00)^t

表 2 100-D Benchmark 函数 SCDE 和其它 DE 效率实验结果比较

F	Gen	SCDE Mean (Std Dev)	SaDE Mean (Std Dev)	jDE Mean (Std Dev)	DE(rand/1) Mean (Std Dev)
f_1	2000	**3.6E-49** (**3.4E-49**)	9.3E-06 (1.2E-05)t	3.9E-05 (1.8E-05)^t	1.1E+01 (2.7E+00)^t
f_2	3000	**1.2E-41** (**4.7E-42**)	4.4E-05 (3.9E-05)^t	2.8E-06 (3.5E-07)^t	7.5E-01 (1.2E-01)^t
f_3	20000	**5.5E-28** (**3.0E-28**)	2.2E+01 (3.9E+01)^t	9.1E-03 (2.5E-02)^t	3.5E+01 (6.5E-01)^t
f_4	9000	**2.8E+00** (**1.4E+00**)	6.0E+02 (4.2E+01)^t	2.3E+02 (8.0E+1)^t	8.1E+02 (2.3E+1)^t
f_5	3000	**4.7E-33** (**2.1E-34**)	3.7E-13 (1.8E-13)^t	3.3E-10 (8.8E-11)^t	1.2E-02 (2.9E-03)^t
f_6	3000	**0.0E+00** (**0.0E+00**)	3.1E-09 (5.2E-09)^t	3.4E-10 (8.6E-11)^t	5.3E-02 (1.4E-02)^t

表 1 是 30-D Benchmark f_1-f_6 的实验结果,实验采用阈值 $u=0.4$,按照表中的迭代次数,独立运行 50 次,取其最优结果的均值和标准差(统计测试分析可以参考文献[10],表 1、表 2 采用双尾 t 检验,自由度 49,置信水平 $a=0.05$)。从表 1 可以看出,标准差分进化算法针对全部所列 Benchmark(f_1-f_6)问题很难找到全局最优点。对于单模态函数 f_1 和 f_3,SCDE 和 SaDE 优化的效果要好于 jDE 和 DE。其中 SCDE 在 f_2,f_3 和 f_4 函数上的实验效果要好于其他 3 者,特别针对 f_1 和 f_3 函数,其收敛的精度遥遥领先。经典的难以收敛的 f_3 函数(Rosenbrock)独立运行 50 次,20000 次迭代,其 30-D 收敛的均值和标准差分别达到 8.0E-28 和 1.3E-28,实验的结果远优于其比较对象。

在拥有多个局部最优点的多模态 f_4-f_6 问题上,SCDE 也拥有较明显的优势,其最终的实验结果相当或好于其比较对象。尤其在 f_5 函数上表现得较为突出。

表 2 的 100-D 实验结果,除了对 f_1,f_2 继续拥有较高的

精度之外，对 f_3 函数 20000 次迭代也有优良的表现，50 次迭代收敛精度远超其比较对象。针对 f_4 函数 4 个比较对象，实验的结果都不太理想，但 SCDE 总体的结果同样优于其比较对象，在 f_5 函数上也是如此。f_6 上指定的迭代次数和对应的 30-D 实验结果类似，50 次独立运行均收敛于全局最优。但其他比较对象，在 f_6 函数对应的实验结果则不太理想，没有一个能在指定的迭代次数全部收敛。

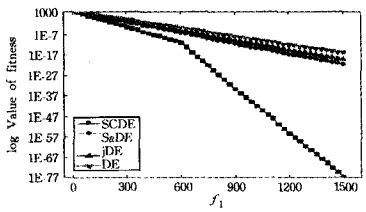


图 1　30 维 f_1 函数实验结果对比图

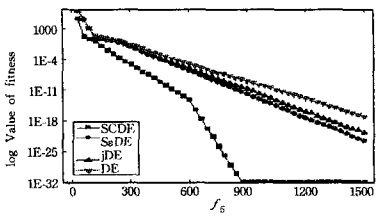


图 2　30 维 f_5 函数实验结果对比图

图 1 和图 2 是 30-D(NP=100) f_1，f_5 收敛情况比较图，可以看出 SCDE 收敛的速度依次明显快于 SaDE，jDE 和 DE。其由于采用了分段设置，进化阶段前 40%速度不是太快，后 60%由于采用了加速的参数配置，从图 1 和图 2 可以看出，进化过程从 40%部分起收敛有个明显的加速过程。

4.3　全局收敛实验比较分析

文献[8]是一个经典的自适应差分进化算法，拥有较精确的全局收敛性。下面本算法及 DE(rand/1/bin，具体的配置及实验结果见表 3)与 SaDE 和 jDE，针对 10-D 和 30-D f_1－f_4 进行比较。种群在此无论维数的高低统一设定 NP=50，最大迭代次数针对 10-D 统一设置为 100 000 次，针对 30-D 则全部设置为 300 000 次[8]。MGS 表示达到全局最优解时的平均迭代次数，即独立运行 30 次，求其均值。SR 表示搜索到全局最优解的成功率。

从表 3 中可以看出，对于 f_1－f_6 函数，10-D 和 30-D 实验的结果是 5 个比较对象在 f_1 函数上均能收敛到全局最优点，SCDE 实验的评价次数明显少于其它 4 个比较对象。在 30-D f_1 函数上，SCDE 相比其它比较对象，虽然都 100%能收敛于全局最优，但 SCDE 的 MGS 明显占优。f_2 函数上 30-D 同样拥有显著的优势。对于 10-Df_3 函数，有 SCDE，SaDE 和 jDE 100%收敛于全局最优，SCDE 表现稍好于 SaDE。30-D f_3 函数其它比较对象均不能 100%收敛于全局最优，但 SCDE 表现得稍好一些。

表 3　10-D 及 30-D Benchmark 函数收敛性实验比较

D	F	SCDE		SaDE		jDE		DE F=0.9 CR=0.1		DE F=0.5 CR=0.3	
		MGS	SR	NFS	SR	NFS	SR	NFS	SR	NFS	SR
10	f_1	4072	100%	5859	100%	7078	100%	12013	100%	6887	100%
	f_2	10021	100%	12359	100%	12618	100%	21832	100%	12292	100%
	f_3	21712	100%	30642	100%	3568	100%	—	0%	—	0%
	f_4	369	100%	11646	100%	831	100%	19200	100%	696	100%
	f_5	5670	100%	6232	100%	7274	100%	14273	100%	8134	100%
	f_6	543	100%	35272	100%	1081	100%	1366	100%	865	100%
30	f_1	12332	100%	15091	100%	19353	100%	44780	100%	20092	100%
	f_2	22353	100%	28734	100%	33826	100%	75201	100%	34662	100%
	f_3	72353	100%	—	90%	—	85%	—	0%	—	0%
	f_4	1171	100%	86575	100%	4624	100%	3190	100%	14233	0%
	f_5	12457	100%	15279	100%	27428	100%	63752	100%	33478	100%
	f_6	829	100%	1095	100%	1436	100%	2998	100%	1372	100%

10-D 实验的结果是 5 个比较对象在 f_4－f_6 函数上均能 100%收敛到全局最优点，但 SCDE 平均迭代次数均是最少；同样，30-D SaDE 在 f_4－f_6 函数上所有的比较对象都能 100%收敛全局最优，SCDE 在这 3 个函数同样拥有绝对领先的优势。特别是在 f_4 和 f_6 函数，无论 10-D 和 30-D 上，SCDE 均拥有远优于其比较对象的收敛效果。

4.4　SCDE 参数分析

标准差分进化算法为了达到最好的优化效果，往往针对不同的问题需要差异性地调整 F 和 CR 的值。因此，为了减少人为的参数调整，需要根据优化问题的不同自适应地调整参数。为此，本文改进算法利用柯西分布随机数引入了两个算子，一个算子用于生成 F，另外一个算子用于生成 CR。把进化过程分为两个分区，分别利用算子产生不同的 CR 值。其 CR 前半程以 0.1 为中心，可有效保证种群的多样性；后半程 CR 围绕 0.7，则可以加快收敛的速度。

采用了分区交叉式的参数控制差分进化算法，实验结果显示，无论对单模态还是多模态，以及高、低维复杂的优化问题，均有良好稳定的表现。虽然本算法需要设置的主要参数有 4 个，但都可以看作是常数。如有必要，只需调整分段阈值 u 即可。阈值 u 增大，可以提高种群多样性。如果对迭代次数要求不高，可以采用该方案获得较好的效果；阈值 u 变小，可以提高收敛的速度(比如说 f_1)，但其影响到种群多样性，对有些函数反而更容易陷入局部最优(比如 f_4 函数等)。其他 4 个参数都被认为是常数，对其进行简单的调整，也可对某些函数起到较好的收敛效果，比如设置 u=0，uF=0.4，$uCR2$=0.7，对 f_1 函数有很好的收敛效果，但对其它函数表现就要差一些。

5　SCDE 的约束优化处理及应用

约束优化问题(constrained optimization problems，COPs)广泛存在于工程和管理领域，由于其难以求解，因而求解约束优化问题具有十分重要的理论和实际意义[11]。

和求解无约束优化问题不同，当优化问题伴随着许多线性和非线性、等式和不等式约束条件时，其处理的过程将变得

更加复杂。为了使 SCDE 能够解决约束优化问题，需要辅助一个约束处理机制。罚函数法是通过对目标函数增加惩罚项，从而将约束优化问题转换为不带约束的优化问题进行处理的，由于简单易行得到了一定的应用。但其能否找到最优解，往往取决于设置合适的惩罚系数。该系数经常与问题相关，参数如何设置又是一个面临的难题[12]。为了避免设置惩罚系数，Deb[11,13]提出了基于个体成对比较的 3 条选择准则。但该准则的主要缺陷是可行解优先，难以发挥不可行解的作用。不可行解将很难进入群体，从而导致了种群多样性降低。Jiménez 和 Verdegay[11,14]提出了一种形如多目标优化中使用的 min-max 表示方法，该法中的个体比较准则类似于 Deb[13]所提出的个体比较准则。

经过试验，SCDE 采用文献[14]提供的 3 个准则作为选择操作，能够获得最佳的约束优化效果。该准则如下。

准则 1　两个比较的个体中，两者均为可行解，则目标函数适应度值较优者胜出。

准则 2　两个比较的个体中，一个个体为可行解，另外一个个体为不可行解时，可行解胜出。

准则 3　当在两个比较的个体中，两个个体均不为可行解，分别得到各自个体中最大违反约束的值，两个值中较小的解胜出，也即最大最小者获准进入下一代。

将上述规则定义为函数 $g(X_1,X_2)$。如果函数值为 1，则表示 X_1 进入下一代，否则 X_2 进入下一代。则新的选择操作为：

$$X_{i,G+1}=\begin{cases}U_{i,G}, & g(U_{i,G},X_{i,G})==1\\ X_{i,G}, & \text{otherwise}\end{cases} \tag{11}$$

分区差分进化算法选择操作如采用式(11)，则在无需其他额外参数设置的情况下可以方便地处理约束优化问题，并能达到较好的优化效果。

考虑经典二次规划(quadratic programming，QP)和其对偶二次规划(dual quadratic programming，DQP)问题[15]。

问题 1(QP)：

$$\text{Min } x_1^2+x_2^2+x_1x_2-30x_1-30x_2$$

$$\text{s. t. } \frac{5}{12}x_1-x_2+x_3-\frac{35}{12}=0$$

$$\frac{5}{2}x_1+x_2+x_4-\frac{35}{2}=0$$

$$x_5-x_1-5=0$$

$$x_2+x_6-5=0,0\leqslant x_i\leqslant 20,i=1,\cdots,6$$

问题 2(DQP)：

$$\text{Min } \frac{35}{12}y_1+\frac{35}{2}y_2+5y_3+5y_4-x_1^2-x_2^2-x_1x_2$$

$$\text{s. t. } \frac{5}{12}y_1+\frac{5}{2}y_2+y_3-2x_2-x_2+30\leqslant 0$$

$$-y_1+y_2+y_4-x_1-2x_2+30\leqslant 0$$

$$-10\leqslant y_i\leqslant 0,i=1,\cdots,4,0\leqslant x_i\leqslant 10,i=1,2$$

QP 问题的最优解为 $x^*=(5,5,5.833333,0,10,0)$，DQP 问题的最优解为 $y^*=(0,-6,0,-9)$，其最优解对应 x 的值则为 $x^*=(5,5)$，两个问题极值均为 -255。

对于 QP 包含的 4 个等式约束 $h_k(x)=0$，将其放松为不等式约束 $|h_k(x)|-\omega\leqslant 0$ 进行处理，取值参照文献[16]$\omega=0.0001$。2 个问题的求解均设置种群 $NP=100$，最大迭代次数 $Gen=10000$，独立运行 10 次取其最优解。QP 问题的最优解为 $x^*=(5,5,5.833333,1.2E\text{-}13,10,3.3E\text{-}13)$；DQP 问题的最优解为 $y^*=(-1.7E\text{-}15,-6,-5.0E\text{-}16,-9)$，$x^*=(5,5)$；二者的最优值均为 -255。从上述实验结果可以看出，本算法和约束处理机制相结合，可以方便地用于约束优化问题的解决。

结束语　差分进化由于可控参数少和寻优的高效性等优点，广泛应用于工程、管理以及科技等领域，但在处理高维复杂优化问题时表现不佳。为此本文提出了分区交叉差分进化算法。首先为了增加寻优的方向性，采用一种有效的变异策略，随机选择当前种群中 m 个个体，把其中的优势解对应的个体向量作为基向量，用于变异中。其次，本文提供了一种分区交叉的优化处理新思路，采用柯西分布随机数设计算子用于 F 的生成，同时在不同区段之中采用不同的配置，利用算子来生成 CR 用于进化。对经典的 Benchmark 函数 10 维、30 维和 100 维以及全局收敛性问题进行了实验分析，结果体现了本算法能明显提高收敛速度和精度。除此之外，本算法还可以结合解决约束优化问题，说明它具有很强的实用性。

参 考 文 献

[1] Storn R, Price K. Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces[J]. Journal of Global Optimization, 1997, 11(4): 341-359

[2] Zhang M, Luo W, Wang X. Differential evolution with dynamic stochastic selection for constrained optimization[J]. Information Sciences, 2008, 178(15): 3043-3074

[3] Pan Q K, Wang L, Qian B. A novel differential evolution algorithm for bi-criteria no-wait flow shop scheduling problems[J]. Computers & Operations Research, 2009, 36(8): 2498-2511

[4] 李庆良，雷虎民，邵雷，等. 一种基于差分进化算法的多模型建模方法[J]. 控制与决策，2010，25(12)：1866-1874

[5] Liu J, Lampinen J. A fuzzy adaptive differential evolution algorithm[J]. Soft Computing-A Fusion of Foundations, Methodologies and Applications, 2005, 9(6): 448-462

[6] Brest J, Greiner S, Boskovic B, et al. Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems[J]. IEEE Transactions on Evolutionary Computation, 2006, 10(6): 646-657

[7] Das S, Abraham A, Chakraborty U K, et al. Differential evolution using a neighborhood-based mutation operator[J]. IEEE Transactions on Evolutionary Computation, 2009, 13(3): 526-553

[8] Qin A K, Huang V L, Suganthan P N. Differential evolution algorithm with strategy adaptation for global numerical optimization[J]. IEEE Transactions on Evolutionary Computation, 2009, 13(2): 398-417

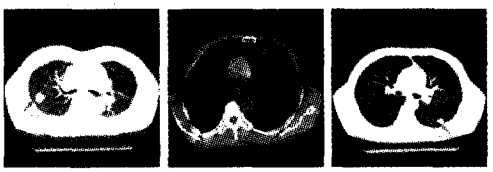
[9] Solis F J. Minimization by random search techniques[J]. Mathematics of Operations Research, 1981: 19-30

[10] Neri F, Tirronen V. Recent advances in differential evolution: a survey and experimental analysis[J]. Artificial Intelligence Review, 2010, 33(1): 61-106

[11] 王勇，蔡自兴，周育人，等. 约束优化进化算法[J]. 软件学报，2009，20(1)：11-29

[12] Runarsson T P, Yao X. Stochastic ranking for constrained evolutionary optimization[J]. IEEE Transactions on Evolutionary Computation, 2000, 4(3): 284-294

(下转第 304 页)



(a)病例1　(b)病例2　(c)病例3

图2　实验原图

采用肺结节检测的敏感性(Sensitivity),分析本文提出的基于局部阈值和聚类中心迭代的肺结节检测算法。分别使用2个不同的局部窗口分割ROIs,敏感性如表1所列。从表1可以看出,选择合适的局部窗口可以提高算法检测的敏感性。

表1　肺结节检测的敏感性

局部窗口大小 (w*h)	阳性结节的数目 (TPs)	假阴性结节的数目(FNs)	本文算法敏感性 (sensitivity)
31*31	32	18	0.64
51*51	39	11	0.78

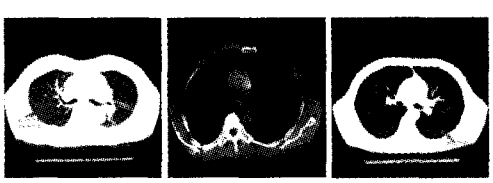
肺结节检测的敏感性定义为

$$Sensitivity=\frac{TP}{TP+FN} \tag{6}$$

式中,TP为检测的真阳性肺结节,FN为漏检真阳性肺结节。

本文造成检测敏感性低的主要原因有:(1)肺结节太小,医学征像和提取的特征很难与血管区分,从而造成小结节漏检。(2)算法是基于2D空间,在2D空间中,肺结节、血管和气管具有很多相似的特征,从而造成肺结节漏检。(3)结节因为与肺壁相粘连,分割出的区域形状不规则,也会产生漏检。

对3种类型的肺结节检测结果如图3所示。(a)为孤立性结节检测结果,(b)为低对比度结节检测结果,(c)为粘连肺壁结节检测结果。从实验检测结果可以观察到,本文提出的算法可以较好地检测肺孤立性肺结节、低对比度的肺结节和粘连肺壁性的肺结节。



(a)病例1检测结果　(b)病例2检测结果　(c)病例3检测结果

图3　检测的结果

结束语　本文提出了基于局部阈值和聚类中心迭代的肺结节检测算法。首先,采用局部阈值算法分割ROIs,并且计算ROIs的形态特征、灰度特征和纹理特征。其次,根据肺结节在CT图像的表现形式,采用基于规则的方法消除非结节的区域并对训练样本采用聚类中心迭代算法,分别求其肺结节聚类中心和非肺结节聚类中心。最后,根据欧式距离,对未知分类的候选结节进行分类。实验结果表明,本文提出的算法能够较好地检测出孤立性结节、低对比度结节和粘连肺壁结节。

参考文献

[1] American Cancer Society. Cancer facts and figures 2005[EB/OL]. http://www.cancer.org

[2] Liu Y, Yang J Z, et al. A Method of Pulmonary Nodule Detection Utilizing Multiple Support Vector Machines[C]// IEEE Internation Conference on Computer Application and System Modeling. 2010, v10-118-v1-1216

[3] Noriaki M, Hyoungseop K, Yoshiinori I, et al. Automatic Detection of Lung Nodules in Tmporal Subtraction Images by Use of Shape and Denisty Features [C] // IEEE Fourth Internation Conference on Innovative Computing. 2009:1288-1292

[4] Zhao B, Gamsu G, Ginsburg M S. Automatic detection of small lung nodules on CT utilizing a local density maximum algorithm [J]. J Appl Clin Med Phys, 2003, 4(3):248-260

[5] Li Q. Recent progress in computer-aided diagnosis of lung nodules on thin-section CT [J]. Computerized Medical Imaging and Graphics, 2007, 31(4/5):248-257

[6] Baek T, Kim J S, Na Y H, et al. Pulmonary Nodules: Automated Detection on CT Images with Morphologic Matching Algorithm-Preliminary Results [C]// Radiology. 2005, 236:259-299

[7] Dehmeshki J, Ye X, Lin X, et al. Automated detection of lung nodules in CT images using shape-based genetic algorithm [J]. Computerized Medical Imaging and Graphics, 2007, 31(6):408-417

[8] Dehmeshiki J, Ye X, et al. A Hybrid Approach for Automated Detection of Lung Nodules in CT Images [C]// IEEE International Symposium on Biomedical Imaging: Macro to Nano. 2006: 506-509

[9] Tong L J, Chen K, et al. Document Image Binarization Based on NFCM[C]// IEEE Confence Image and Signal Processing. 2009: 1-5

[10] Zhang J, Li B, Tian L F. Lung nodule classification combining rule-based and SVM [C]// 5th IEEE International conference on Bio-Inspired Computing: Theories and Appications (BIC-TA). 2010, 2:1033-1036

(上接第287页)

[13] Deb K. An efficient constraint handling method for genetic algorithms[J]. Computer Methods in Applied Mechanics and Engineering, 2000, 186(2-4):311-338

[14] Jiménez F, Verdegay J L. Evolutionary techniques for constrained optimization problems[C]// Zimmermann HJ, ed. Proc. of the 7th European Congress Intelligence Techniques and Soft Computing(EUFIT'99). Berlin: Springer-Verlag, 1999

[15] Ghasabi-Oskoei H, Mahdavi-Amiri N. An efficient simplified neural network for solving linear and quadratic programming problems[J]. Applied Mathematics and Computation, 2006, 175(1):452-464

[16] Runarsson T P, Yao X. Search biases in constrained evolutionary optimization[J]. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 2005, 35(2):233-243