免比例因子 F 的差分进化算法

张晓伟[1,2],刘三阳[1]
(1. 西安电子科技大学数学科学系,陕西西安 710071;2. 电子科技大学应用数学学院,四川成都 610054)

摘 要: 比例因子 F 的合适赋值常会大大改善差分进化算法的求解性能,但是如何给值是个麻烦的事情. 本文给出了二种免比例因子 F 的差分进化算法. 算法将每一个个体视为带电粒子,利用之间的吸引、排斥机制,确定个体在差分方向上移动的长度,依此免去比例因子 F 设置的麻烦. 通过和两种 PSO 算法以及其它四种不同赋值策略的算法的数值试验比较,表明提出的算法相比其它相比较的算法有更好的求解性能.

关键词: 类电磁机制; 全局优化; 粒子群优化; 差分进化
中图分类号: TP18; O224 **文献标识码**: A **文章编号**: 0372-2112 (2009) 06-1318-06

Differential Evolution Without the Scale Factor F

ZHANG Xiao-wei[1,2], LIU San-yang[1]
(1. *Department of Mathematical Sciences, Xidian University, Xi'an, Shaanxi* 710071, *China*;
2. *School of Applied Mathematics, University of Electronic Science and Technology of China, Chengdu, Sichuan* 610054, *China*)

Abstract: A fit setting of the scale factor F can usually improve greatly the performance of differential evolution, however, how to set is nuisance. Two differential evolutions without scale factor F are presented in the paper. The algorithms look upon each individuals as a charged particle and utilize the attraction-repulsion mechanism of the particles to decide on the step length of the motion of the individual in the direction of the difference for the purpose of avoiding the setting of the scale factor F. The comparisons of numerical experiments among the proposed algorithms, two PSO algorithms and four other algorithms with the different setting strategies are done, which show that the performance of the proposed algorithms outperform other compared algorithms.

Key words: electromagnetism-like mechanism; global optimization; particle swarm optimization; differential evolution1

1 引言

差分进化算法(Differential Evolution,DE)于 1995 由 Storn 和 Price 提出[1],该算法和其他进化类算法主要的不同在于差分变异的使用. 针对种群中的每一个个体,首先利用随机选取的其它个体间的"差分"信息得到试验个体,然后将试验个体和该个体的每一个分量以概率进行离散交叉得到候选个体,再以"贪婪"选择机制决定哪个个体进入下一代种群. DE 算法简单、鲁棒性高、易于程序实现、全局收敛能力强. 数值试验也表明 DE 算法对参数不太敏感,但是一个合适的参数会大大改善算法的求解性能. 研究者已经在参数比例因子 F 的改进方面做了很多工作. 这些改进的主要特点是参数在 DE 迭代过程中是动态变化的,不是事先取之于[0.4,1]中的某个值. 如 Abbass[2]用 F 取为均值为 0、方差为 1 的 DE 去求解多目标优化问题,即 $F\sim N(0,1)$. Qin 和 Suganthan 在 2005 的工作中[3],F 也取为正态分布随机数,不过这里 $F\sim N(0.5,0.3)$. 数值实然表明具有正态分布 F 的 DE 一般要比标准的 DE 有效,而且 Zaharie 也证明在这种情形下 DE 会以概率收敛到问题的全局最优解[4]. 然而,Ronkkonen 和 Lampinen 发现具有 Gauss 分布的 F 并不会本质的增加 DE 的求解性能[5]. 2007 年 Kim 等则将 F 修改为 $F=a+b\cdot\mathrm{rand}(0,1)$,修改的 DE 在一些工程问题求解中比标准 DE 鲁棒、有效,这里 rand(0,1)表示[0,1]之间均匀分布的随机数,a,b 是大于零的实数,而且 $a+b<1$[6]. 比例因子 F 的产生现在没有什么更好的方式,总的来说,F 值越大,种群趋向于探索,反之,则有利于局部搜索,这是一对矛盾. [0.4,1]是 F 的经验取值范围,0.5 是常用到的一个值,数值实验表明其能较好的平衡矛盾间的关系[1].

类电磁机制算法是由 Birbil 和 Fang 于 2003 年提出的一种模拟电磁场中的吸引和排斥机制的随机全局优化算法[7]. 该算法将种群中的每个个体比作带电粒子,然后按一定的准则使得搜索粒子朝最优解移动. 这种思想来源于电磁理论中带电粒子间的吸引与排斥机制. 数值试验表明类电磁机制算法能够快速地收敛到较低维

收稿日期:2008-04-22;修回日期:2009-02-23
基金项目:国家自然科学资金资助(No. 60674108,60705004);综合业务网理论及关键技术国家重点试验室基金资助(No. ISN02080003)

数问题的全局最优解. 当迭代次数趋于无穷时,种群中至少有一个个体以概率 1 移动到全局最优的 ε 领域内[8]. 本文则是借助类电磁算法带电粒子间相互吸引、排除机制给出了二种免参数 F 的差分进化算法. 算法主要的特点如下:(1) 避免了比例因子 F 的设置;(2) 避免了标准 DE 向"上山"方向错误前进的可能. 最后的数值试验表明提出的算法比相比较的算法更有效.

2　类电磁机制算法

根据电磁理论中的吸引—排斥机制(库仑定理),类电磁机制算法首先根据问题的目标函数值确定出种群中每个个体(带电粒子)所带的电荷量. 电荷量的大小决定了该粒子对其他粒子的吸引或者排斥的强弱程度,即目标函数值越小,吸引力就越强,反之,排斥力就越大. 然后算法计算每一个粒子所受到其它粒子施加于它的电磁力的合力,以此来确定该粒子下一步移动的方向.

计算每一个粒子所受的合力的方法同电磁力的一样,就是通过将受到的其他粒子的力进行矢量叠加. 比如,粒子 X_1 优于粒子 X_2,而粒子 X_1 劣于粒子 X_3,那么粒子 X_2 将对粒子 X_1 有一个排斥力 $F_{1,2}$,而粒子 X_3 对粒子 X_1 有一个吸引力 $F_{1,3}$,根据平行四边形法则,粒子 X_1 所受的合力 $F_1 = F_{1,2} + F_{1,3}$,如图 1.

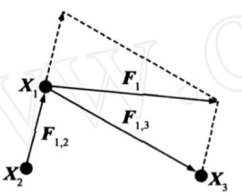

图1　粒子X_1到的合力F_1

算法 1　类电磁机制算法(EM)

Step1　初始化种群. 若令 X_i^k 表示粒子 X_i 的第 k 个分量,$[L, U]$为 X_i^k 的取值范围,那么种群将按照下列方式产生:

$$X_i^k = l + \mathrm{rand}(0,1) \cdot (U - L),\ k = 1, \cdots, n$$

Step2　停机条件(最大迭代数)是否满足?

Step3　局部搜索. 每个粒子在每一维按照一定的步长在规定的搜索次数内进行搜索,一旦得到了一个更好的解就停止对该维的搜索. 局部搜索为种群的全局搜索提供了有效的局部信息.

Step4　计算合力. 每个粒子 X_i 所带的电荷量 Q_i 决定着该粒子所受的吸引力或者排斥力的大小,按照式(1)计算.

$$Q_i = \exp\left(- n \cdot \frac{f(X_i) - f(X_{best})}{\sum_{j=1}^{POP} (f(X_j) - f(X_{best}))} \right) \tag{1}$$

这里,X_{best}表示种群中最好的个体,即所带电荷量最大的粒子. POP 表示种群的大小. n 是种群的维数. Delta 为实数,用以控制搜索步长. LSITER 为最大搜索次数. 在计算了种群中所有粒子的电荷量之后,就要确定粒子 X_i 所受的合力 F_i(其它 POP - 1 粒子对它施加的力的合力). 对所有的 $i \neq j (i = 1, \cdots, POP)$,$X_i$ 受到 X_j 的电磁力 $F_{i,j}$如下计算:

$$F_{i,j} = \begin{cases} (X_j - X_i) \dfrac{Q_i Q_j}{\| X_j - X_i \|^2}, & \text{if } f(X_j) \leq f(X_i) \\ (X_i - X_j) \cdot \dfrac{Q_i Q_j}{\| X_j - X_i \|^2}, & \text{if } f(X_j) > f(X_i) \end{cases} \tag{2}$$

一旦计算出每一个粒子 $X_j (j \neq i)$ 施加在 X_i 上的电磁力 $F_{i,j}$后,再根据式(3)算出粒子 X_i 所受的合力 F_i.

$$F_i = \sum_{\substack{j=1 \\ j \neq i}}^{POP} F_{i,j} \tag{3}$$

Step5　个体移动. 转 Step2.

每个粒子 X_i 将沿着其合力 F_i 的方向按式(4)移动.

$$X_i = X_i + \mathrm{rand}(0,1) \cdot \frac{F_i}{\| F_i \|} \cdot R \tag{4}$$

在式(4)中,R 为一个向量,其每一个分量定义了对应的向上边界 U 或者下边界 L 移动的可行步长. 即,

$$R^k = \begin{cases} U - X_i^k, & \text{if } F_i^k > 0 \\ X_i^k - L, & \text{if } F_i^k < 0 \end{cases} \tag{5}$$

这里,F_i^k 是 F_i 的第 k 个分量,合力 F_i 被单位化是为了保证个体在可行域内移动. 关于这方面更详细的内容,可阅读 Birbil 和 Fang 的文章[7,8].

3　免参数 F 的差分进化算法

DE 算法有几种变形,最常用的为 DE/ Rand/ 1. 对于一般的 DE/ a/ b,其含义如下:a 表示基向量(个体)选择方式;b 表示差分向量的个数. 例如:

DE/ Rand/ 1　$V = X_{r1} + F \cdot (X_{r2} - X_{r3})$

DE/ Rand/ 2　$V = X_{r1} + F \cdot (X_{r2} - X_{r3} + X_{r4} - X_{r5})$

DE/ Best/ 2　$V = X_{best} + F \cdot (X_{r2} - X_{r3} + X_{r4} - X_{r5})$

这里,$r1 \neq r2 \neq r3 \neq r4 \neq r5$,均为$[1, POP]$间的随机整数.

在按照式(1)～(3)力的计算中,对每一个个体(带电粒子),其所受到的力是由其它的 POP - 1 个个体决定. 一种 EM 算法的改进是,力由种群中随机选取的某一个个体来确定. 这种方法出现在文献[9]中,见式(6). 作者用其求解了项目进度安排问题. 这个做法保留了 EM 算法的精髓,而且计算简单. 但是 EM 算法一个主要的缺点是一但最好点陷入局部最优,那么在很少的代数内整个种群会聚集于此,出现"早熟",这从另一方面也说明 AEM 有较强的局部搜索能力.

$$F_{i,j} = Q_{i,j}(X_j - X_i) = \frac{f(X_i) - f(X_j)}{f(X_{worst}) - f(X_{best})} (X_j - X_i) \tag{6}$$

这里 X_{worst}表示当前种群中最差的个体. 在我们的算法

中基向量 X_{r1} 受到的力有两种方法来计算,即式(7)及式(8).

$$V = X_{r0} + Q_{r1,r2}\cdot(X_{r2} - X_{r1}) = X_{r0} + \frac{f(X_{r1}) - f(X_{r2})}{f(X_{worst}) - f(X_{best})}\cdot(X_{r2} - X_{r1}) \quad (7)$$

$$V = X_{r1} + Q_{r1,r2}(X_{r2} - X_{r1}) + Q_{r1,r3}(X_{r3} - X_{r1}) = X_{r1} + \left[\frac{f(X_{r1}) - f(X_{r2})}{f(X_{worst}) - f(X_{best})}\cdot(X_{r2} - X_{r1}) + \frac{f(X_{r1}) - f(X_{r3})}{f(X_{worst}) - f(X_{best})}\cdot(X_{r3} - X_{r1})\right] \quad (8)$$

这里一个自然问题是为何不按照 $V = X_{r1} + F_{r1,r2}$ 计算,这主要是出于下面的考虑.这个式子等于下式:

$$X_{r1} + Q_{r1,r2}\cdot(X_{r2} - X_{r1}) = (1 - Q_{r1,r2})\cdot X_{r1} + Q_{r1,r2}\cdot X_{r2} \quad (9)$$

从式(6)可看出 $Q_{i,j}\in[-1,1]$,因此对于式(9),试验个体成为 X_{r1} 和 X_{r2} 的仿射组合的一部分,或者是 $2X_{r1} - X_{r2}$ 和 X_{r2} 的凸组合.这是很平凡的一种做法,也失去了 DE 的本质.

从式(7)、(8)可以看出,其有类似于 DE/Rand/1 和 DE/Rand/2 中的差分方程的形式,且没有比例因子 F,或者可将带电量看作比例因子 F,但其会根据选择的个体的函数值解析的算出.预先的数值试验表明,式(8)一般能够给出较好的试验个体,而且还有一个隐含的特点,那就是可能避免 DE 向"上山"方向错误前进的可能.详见下面描述.

对于驼峰函数 SHCB(见附录 f_0),它的最优解为(-0.08984,0.71265)和(0.08984,-0.71265),最优值为-1.031628.我们给出了3组试验点.

(1) $X_{r1} = (-0.61462, 0.081834)$, $X_{r2} = (0.15779, -0.30533)$, $X_{r3} = (0.0040769, -0.80874)$.由式(8)计算后,得到 $V = (0.14606, -0.68055)$,见图2.

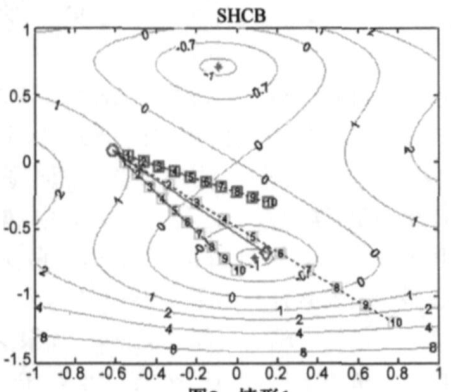

图2 情形1

(2) $X_{r1} = (0.15961, 0.48913)$, $X_{r2} = (0.28105, 0.86676)$, $X_{r3} = (0.94169, -0.23207)$.计算后得到 $V = (-0.4222, 0.97067)$,见图3.

(3) $X_{r1} = (-0.39, -0.91221)$, $X_{r2} = (-0.15301, 0.28698)$, $X_{r3} = (0.13566, -0.58573)$.计算后得到 $V = (-0.12891, -0.54275)$,见图4.

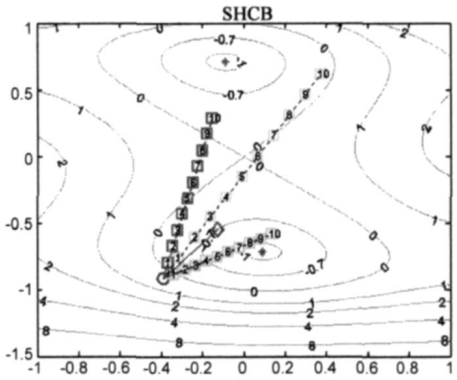

图3 情形2

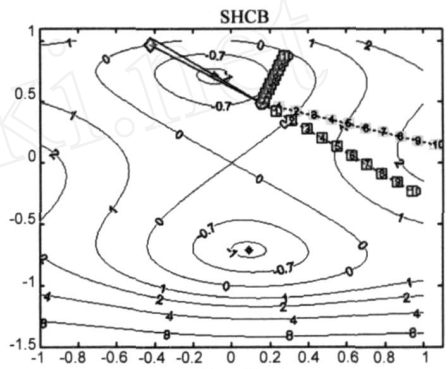

图4 情形3

图2~4中给出了 SHCB 函数在 $[-1,1]^2$ 中的等值线,其对应的函数值也被给出.星号表示最优解,圆圈表示 X_{r1},外侧的两个方框10分别表示 X_{r2} 和 X_{r3},两个实直线表示基向量 X_{r1} 在所受到作用力 $F_{r1,r2}$ 和 $F_{r1,r3}$ 方向上移动的长度,菱形表示由式(8)得到的试验个体 V,即就是合力.外侧的两列方框分别表示 F 取不同值时,下面的式(10)中右端基向量 X_{r1} 后面的两项,这里比例因子 F 的取值范围为 $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1\}$.

$$V = X_{r1} + F\cdot[(X_{r2} - X_{r1}) + (X_{r3} - X_{r1})] = X_{r1} + F\cdot(X_{r2} - X_{r1}) + F\cdot(X_{r3} - X_{r1}) \quad (10)$$

在这两列方框中间的一列方框表示其对应的试验个体 V 的值,它们之间的对应关系可以通过方框中的标号对应.式(8)不同于 DE/Rand/2 中的差分方程,但若将 $X_{r2} - X_{r1}$ 和 $X_{r3} - X_{r1}$ 分别看作一个整体,而忽略内部间的作用关系,那么其可以看作是 DE/Rand/1 中的差分方程.之所以式(8)和式(10)来比较,是因为两者结构相似,若有差异,则在图上的表现就一目了然,而且式(10)也没有背离 DE 的本质.

在图2中,只有当 $F = \{0.4, 0.5, 0.6, 0.7\}$ 时,由式(10)得到的试验个体较好,其它的取值将会得到更差的试验个体.然而由式(8)得到的试验个体与全局最优解

很接近.同样类似的情形出现在图 3 中.图 4 中式(8)的优越性表现的非常突出.虽然 $F = 0.2$ 时其值与式(10)的相当,但是式(10)向一个错误的"上山"方向前进.

下面我们给出两种免比例因子 F 的差分进化算法(DE1/ F 和 DE2/ F).

算法 2　免比例因子 F 的差分进化算法(DE1/ F)

Step1　初始化参数.种群个数 POP,最大代数 MI,当前代数 ITER = 1,ε,交叉概率 C_r.

Step2　初始化种群.形成种群 P_{ITER}.

Step3　如果 ITER > MI **或** $|f(X_{worst}) - f(X_{best})| < \varepsilon$,**则停止并输出最优解.**

Step4　对每个 $X_i \in P_{ITER}$**按照式(7)计算试验个体** V,**这里** $i \neq r1 \neq r2 \neq r3$.**若** V **不在问题的可行区域内,这一步将重复计算,直到满足约束为止.**

Step5　如下执行交叉操作.

$$Y_i^j = \begin{cases} X_i^j, \text{if rand}(0,1) \leq C_r \text{ and } j \neq i \\ V^j, \text{else} \end{cases} \quad (11)$$

这里上标 j 表示相应个体的第 j 个分量.

Step6　"贪婪"选择.如果 $f(Y) < f(X_i)$,**则** $X_i = Y$. ITER = ITER + 1,**转** Step3.

算法 3　免比例因子 F 的差分进化算法(DE2/ F)

算法 3 和算法 2 除了在 Step4 中试验个体产生不用式(7)而用式(8)外,其它的步骤和算法 2 完全一致.在停机准测中 $|f(X_{worst}) - f(X_{best})| < \varepsilon$ 是为了计算的数值稳定,因为在个体电量的计算中 $f(X_{worst}) - f(X_{best})$ 做了分母.

4　数值试验

我们对提出的算法 DE1/ F 和 DE2/ F 在 Matlab7 环境下进行了数值模拟,参数设置都为 POP = 70, $C_r = 0.9$, $\varepsilon = 1E-100$.算法对表 1 中 10 个测试函数分别独立运行 20 次,在求解结果的平均最优值、最优值的方差以及评估次数上分别和下面 6 种算法进行了比较.表 1 给出了 10 个测试函数的搜索区间、全局最优解和全局最优值以及相应的求解维数.

表 1　$f_1 - f_{10}$求解维数、搜索区间、全局最优解及全局最优值

函数	维数	搜索区间	全局最优解	全局最优值
f_1	30	$[-100\ 100]^n$	$(0,\cdots,0)$	0
f_2	30	$[-32\ 32]^n$	$(0,\cdots,0)$	0
f_3	30	$[1.28\ 1.28]^n$	$(0,\cdots,0)$	0
f_4	30	$[-50\ 50]^n$	$(1,\cdots,1)$	0
f_5	30	$[-50\ 50]^n$	$(0,\cdots,0)$	0
f_6	30	$[-600\ 600]^n$	$(0,\cdots,0)$	0
f_7	30	$[-5.12\ 5.12]^n$	$(0,\cdots,0)$	0
f_8	100	$[-5\ 10]^n$	$(1,\cdots,1)$	0
f_9	30	$[-10\ 10]^n$	$(0,\cdots,0)$	0
f_{10}	30	$[-100\ 100]^n$	$(0,\cdots,0)$	0

文献[10]和文献[11]分别提出了一种改进的粒子群进化算法 HPSO 和 CPSO,算法对 $f_1 - f_{10}$分别运行 50 次后,给出了求解结果.计算的结果见文献[10～12]、表 2 和表 3.提出的算法和这些结果做了直接的比较.为了说明提出算法在免比例因子 F 方面的优越性,算法还和标准 DE(DE/ rand/ 1)、$F = 0.4 + 0.4 \cdot \text{rand}(0,1)$ 的 DEF[6]、$F \sim N(0,1)$ 的 DEG0[2]以及 $F \sim N(0.5, 0.3)$ 的 DEG0.5[3]进行比较.公平起见,参数设置都为:POP = 70, $F = 0.5$, $C_r = 0.9$;DEF、DEG0 和 DEG0.5 的交叉操作按式(12)进行.

表 2　DE1/ F、DE2/ F 及其它 6 种算法求解 $f_1 - f_5$ 的数值结果比较

函数	算法	平均值	方差	评估次数
f_1	HPSO	5.6193E-002	1.33657E-001	121145
	CPSO	1.3114E-001	2.7676E-001	120000
	DE	5.120949E-023	7.378043E-023	119070
	DEF	1.369509E-017	4.832588E-018	119070
	DEG0	1.652111E-021	4.262193E-021	119070
	DEG0.5	1.757227E-029	5.993760E-029	119070
	DE1/ F	3.713906E-028	2.096048E-028	119070
	DE2/ F	1.972022E-032	1.063446E-032	119070
f_2	HPSO	1.546E-004	4.752E-005	122067
	CPSO	2.773E-003	4.167E-003	150000
	DE	1.637446E-012	1.837765E-012	119070
	DEF	1.082795E-009	2.716100E-010	119070
	DEG0	5.137514E-001	7.039830E-001	119070
	DEG0.5	2.037179E-001	5.122931E-001	119070
	DE1/ F	2.255973E-014	2.276306E-015	119070
	DE2/ F	1.101341E-014	2.647339E-015	119070
f_3	HPSO	3.6245E-001	1.1288E-001	123342
	CPSO	6.4903E-001	3.0693E-001	120000
	DE	7.319620E-003	2.245911E-003	119070
	DEF	2.017413E-002	5.634481E-003	119070
	DEG0	1.953405E-002	1.088926E-002	119070
	DEG0.5	1.184403E-002	5.737578E-003	119070
	DE1/ F	1.630074E-002	3.586426E-003	119070
	DE2/ F	9.271430E-003	2.595265E-003	119070
f_4	HPSO	3.2465E-001	7.7820E-001	147962
	CPSO	2.0810E-001	2.7038E-001	150000
	DE	8.259483E-024	1.190187E-023	119070
	DEF	4.116632E-019	1.380280E-019	119070
	DEG0	7.168907E+001	3.201389E+002	119070
	DEG0.5	4.146740E-002	9.748170E-002	119070
	DE1/ F	4.807917E-030	7.238646E-030	119070
	DE2/ F	1.570545E-032	2.808012E-048	118755
f_5	HPSO	5.6735E-001	1.6146E-001	152548
	CPSO	9.7155E-001	9.0252E-001	150000
	DE	2.057781E-023	1.652255E-023	119070
	DEF	2.570083E-018	9.648983E-019	119070
	DEG0	5.323900E-001	1.246407E+000	119070
	DEG0.5	6.309113E+002	2.820643E+003	119070
	DE1/ F	5.265687E-029	4.698807E-029	119070
	DE2/ F	1.491530E-032	2.049890E-033	119070

表 3 DE1/ F、DE2/ F 及其它 6 种算法求解 f_6 - f_{10}的数值结果比较

函数	算法	平均值	方差	评估次数
f_6	HPSO	1.849E-002	1.237E-002	144149
	CPSO	4.0065E-001	2.260E-001	150000
	DE	3.698020E-004	1.653805E-003	119070
	DEF	3.053113E-016	4.804032E-016	119070
	DEC0	1.132178E-002	1.724979E-002	119070
	DEC0.5	6.154975E-003	1.069657E-002	119070
	DE1/ F	6.692625E-006	1.246370E-005	119070
	DE2/ F	4.789816E-006	1.920384E-005	99827
f_7	HPSO	2.185E-003	3.0375E-002	236695
	CPSO	4.3753E-001	3.2068E-001	250000
	DE	1.524641E+002	1.612246E+001	119070
	DEF	3.280431E-007	3.299214E-007	119070
	DEC0	2.004842E+001	6.656986E+000	119070
	DEC0.5	1.910320E+001	5.160876E+000	119070
	DE1/ F	4.974795E-002	2.224796E-001	119070
	DE2/ F	9.949591E-002	3.062419E-001	119070
f_8	HPSO	9.762345E+001	9.7896E-001	178792
	CPSO	6.9072E-001	2.8421E-001	200000
	DE	2.498767E+002	7.443217E+001	119070
	DEF	6.475810E+002	7.401500E+001	119070
	DEC0	5.958584E+002	2.339653E+002	119070
	DEC0.5	2.847018E+002	7.975838E+001	119070
	DE1/ F	2.649782E+002	7.378302E+001	119070
	DE2/ F	1.006276E+001	2.115634E+001	119070
f_9	HPSO	1.8091E-001	7.5567E-001	120316
	CPSO	8.4745E-001	4.9846E-001	120000
	DE	1.039846E-011	7.578122E-012	119070
	DEF	6.210984E-011	1.245983E-011	119070
	DEC0	1.081246E-014	1.778142E-014	119070
	DEC0.5	1.089285E-019	5.935102E-020	119070
	DE1/ F	2.455501E-017	6.197960E-018	119070
	DE2/ F	5.222028E-020	1.568907E-020	119070
f_{10}	HPSO	7.6024E-001	7.2903E-001	121378
	CPSO	5.9038E-001	1.372E-003	120000
	DE	3.670133E+000	2.401217E+000	119070
	DEF	1.594262E+000	2.004368E-001	119070
	DEC0	2.365727E+001	5.599349E+000	119070
	DEC0.5	2.021148E+001	6.861107E+000	119070
	DE1/ F	5.189996E-001	9.866108E-002	119070
	DE2/ F	5.916996E-002	1.818585E-002	119070

$$Y_i^j=\begin{cases} V^j,\text{if rand}(0,1) \le C_r \text{ or } j=\text{irand} \\ X_i^j,\text{else} \end{cases} \quad (12)$$

这里 irand 是 1 到 POP 之间的随机整数.

为了与 HPSO 和 CPSO 的已有结果进行比较,其它 DE 类的算法的进化最大代数均为 MI = 1700,因为此时函数的评估次数小于或等于 119070. 这个值和 PSO 类算法的最小评估次数 120000 大致相当,便于比较.

f_1 是一个简单的超球面,有一个局部最优解,即全局最优. DE 类算法用了较少的评估次数,得到了精度更高的解和更小的方差. 其中,DE2/ F 求解结果最好. f_2 和 f_3 为有噪音的函数,有多个局部最优解,同样,DE 类好于 PSO 类. 对于 f_2,DE1/ F 和 DE2/ F 要好于其它算法,对于 f_3,DE 类算法求解结果相当,DE 和 DE2/ F 略好些. 具有罚项的测试函数 f_4 和 f_5 有许多局部最优解,除了 DEC0 外,其它 DE 类算法均能满意的全局最优解,在求解进度方面,DE2/ F 最好,DE1/ F 次之,再次是 DE,更次是 DEF. 而具有 Gauss 分布的 DEC0 和 DEC0.5 求解精度差.

f_6 和 f_7 分别是被经常用来测试进化算法全局搜索能力的 Griewank 函数和 Rastrigin 函数,其有许多局部最小点,在这些算法中,DEF 得到的结果最好,其次为 DE1/ F 和 DE2/ F. 2 维的 f_8 就是很难求解的 Banana 函数,CPSO 求解的结果最好,其次为 DE2/ F 和 HPSO,其余的相差不大. f_9 和 f_{10}的一个特点就是目标函数不可微,对于 f_9,DE2/ F 求解结果最好,且具有最小的方差. 对于 f_{10},DE2/ F 最好,其次为 DE1/ F,CPSO 和 HPSO,下来为 DE 和 DEF,DEC0 和 DEC0.5 求解结果最差.

从表 2 和表 3 我们可以看出,提出的免比例因子 F 的差分进化算法 DE1/ F 和 DE2/ F,尤其是 DE2/ F,相比其它算法有很强的优势.

5 结论

本文提出了两种免比例因子 F 的差分进化算法,该算法免去了差分向量比例系数 F 设置的麻烦,借用类电磁机制算法中带电粒子之间相互吸引、排斥机制,待变异个体能够自动确定其在差分向量方向移动的大小. 在多个复杂、困难优化函数的求解测试中以及和其它策略的比较中,提出算法表现出很好的鲁棒、高效性能,尤其是算法 DE2/ F. 继续的工作为:(1) 探究 DEF 高效求解 f_6,f_7 的原因,依此改善我们提出的算法;(2) 从理论上分析提出算法,给出收敛性证明.

附录

$$f_0(x)=4x_1^2-2.1x_1^4+\frac{1}{3}x_1^6+x_1x_2-4x_2^2+4x_2^4$$

$$f_1(x)=\sum_{i=1}^{n}x_i^2$$

$$f_2(x)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right)+20+\exp(1)$$

$$f_3(x)=\sum_{i=1}^{n}i\cdot x_i^4+\text{random}[0,1)$$

$$f_4(x)=\frac{\pi}{n}\{10\sin^2(\pi y_1)+\sum_{i=1}^{n-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})]+(y_n-1)^2\}+\sum_{i=1}^{n}u(x_i,10,100,4)$$

$$y_i = 1 + \frac{x_i + 1}{4},\ i = 1, \cdots, n,$$

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

$$f_5(x) = \frac{1}{10}\{\sin^2(3\pi x_1) + \sum_{i=1}^{n-1}(x_i - 1)^2[1 + \sin^2(3\pi x_{i+1})] + (x_n - 1)^2[1 + \sin^2(2\pi x_n)]\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$$

$$f_6(x) = \sum_{i=1}^{n}\frac{x_i^2}{4000} - \prod_{i=1}^{n}\cos\frac{x_i}{\sqrt{i}} + 1$$

$$f_7(x) = 10n + \sum_{i=1}^{n}[x_i^2 - 10\cos(2\pi x_i)]$$

$$f_8(x) = \sum_{i=1}^{n-1}[100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2]$$

$$f_9(x) = \sum_{i=1}^{n}|x_i| + \prod_{i=1}^{n}|x_i|$$

$$f_{10}(x) = \max\{|x_i|, i = 1, 2, \cdots, n\}$$

参考文献:

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

[2] Abbass. The self-adaptive pareto differential evolution algorithm [A]. Proceedings of the IEEE Congress on Evolutionary Computation[C]. Honolulu, USA: IEEE Press, 2002. 831 - 836.

[3] Qin, Suganthan. Self-adaptive differential evolution algorithm for numerical optimization [A]. Proceedings of the IEEE Congress on Evolutionary Computation[C]. Edinburgh, USA: Institute of Electrical and Electronics Engineers Computer Society, 2005. 1785 - 1791.

[4] Zaharie. Critical values for the control parameters of differential evolution algorithms[A]. Eighth International MENDEL Conference on Soft Computing[C]. Brno, Czech Republic: Brno University of Technology, 2002. 62 - 67.

[5] Ronkkonen, Lampinen. On using normally distributed mutation step length for the differential evolution algorithm[A]. Ninth International MENDEL Conference on Soft Computing[C]. Brno, Czech Republic: Brno University of Technology, 2003. 11 - 18.

[6] Kim, Chong, Park, et al. Differential evolution strategy for constrained global optimization and application to practical engineering problems[J]. IEEE Transactions on Magnetics, 2007, 43(4): 1565 - 1568.

[7] Birbil, Fang. An electromagnetism-like mechanism for global optimization[J]. Journal of Global Optimization, 2003, 25(3): 263 - 282.

[8] Birbil, Fang, Sheu. On the convergence of a population-based global optimization[J]. Journal of Global Optimization, 2004, 30(3): 301 - 318.

[9] Debels, Reyck, Leus, et al. A hybrid scatter search/electromagnetism meta-heuristic for project scheduling[J]. European Journal of Operational Research, 2006, 169(2): 638 - 653.

[10] Ratnaweera, Halgamuge, Watson. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients[J]. IEEE Transactions on Evolutionary Computation, 2004, 8(3): 240 - 255.

[11] Bergh, Engelbrecht. A cooperative approach to particle swarm optimization[J]. IEEE Transactions on Evolutionary Computation, 2004, 8(3): 225 - 239.

[12] Wang Yuping, Dang Chuangyin. An evolutionary algorithm for global optimization based on level-set evolution and latin squares[J]. IEEE Transactions on Evolutionary Computation, 2007, 11(5): 579 - 595.

作者简介:

张晓伟　男,1979 年 11 月出生,博士研究生. 主要研究方向为进化计算及最优化理论与方法. 发表文章 10 余篇,其中被 SCI、EI、ISTP 检索 8 篇. Email: x. w. zhang @126. com

刘三阳　男,1959 年 11 月出生,博士,教授,博士生导师,国家教学名师. 主要从事应用数学,特别是最优化和运筹学的研究. 现任西安电子科技大学理学院院长、工业与应用数学研究所所长. 出书 10 部,在国内外重要刊物杂志上发表论文 300 多篇.

Email: liusanyang @126. com