文章编号: 1001-0920(2017)07-1196-07 DOI: 10.13195/j.kzyjc.2016.0687

基于双变异模式协同的自适应微分进化算法

王世豪[1,2], 杨红雨[1,2†], 李玉贞[3], 韩松臣[1,2], 杨 波[2]

(1. 四川大学 空天科学与工程学院，成都 610065；2. 四川大学 国家空管自动化系统技术重点实验室，成都 610065；3. 上海电器科学研究所，上海 200063)

摘 要: 针对微分进化算法(DE)易陷入局部最优解、进化后期收敛速度慢、求解精度低等缺点,结合DE/rand/1和DE/best/1两种变异模式分别具有全局探索能力和局部开发能力的优点,引入精英存档策略和控制参数自适应策略,提出一种双变异模式协同自适应微分进化(DMCSaDE)算法. 15个典型benchmark测试函数的实验结果表明,DMCSaDE能够有效提高算法的全局探索能力和局部开发能力,避免早熟收敛,大大提高算法的收敛性能和鲁棒性,同时,精英种群的大小对DMCSaDE的优化性能具有明显的影响.

关键词: 微分进化；全局优化；精英存档；控制参数自适应

中图分类号: TP18 **文献标志码:** A

Self-adaptive differential evolution algorithm with dual mutation modes collaboration

WANG Shi-hao[1,2], *YANG Hong-yu*[1,2†], *LI Yu-zhen*[3], *HAN Song-chen*[1,2], *YANG Bo*[2]

(1. School of Aeronautics and Astronautics，Sichuan University，Chengdu 610065，China；2. National Key Laboratory of Air Traffic Control Automation System Technology，Sichuan University，Chengdu 610065，China；3. Shanghai Electrical Apparatus Research Institute，Shanghai 200063，China)

Abstract: The differential evolution(DE) algorithm has some disadvantages, such as easy to fall into local optimal solutions, slow convergence speed and low convergence precision in the later stage of evolution. By means of combining DE/rand/1 mutation mode with global exploration ability and DE/best/1 mutation mode with local exploitation ability, and introducing the elite archive strategy and control parameters adaptation strategy, a self-adaptive differential evolution with dual mutation modes collaboration(DMCSaDE) is proposed. A total of 15 typical benchmark test functions are used to perform comparative experiments. The experimental results show that DMCSaDE can effectively improve the global exploration ability and local exploitation ability, avoid the premature convergence and greatly improve the convergence performance and robustness. Additionally, the size of elite population has a significant effect on the optimization performance of DMCSaDE.

Keywords: differential evolution；global optimization；elite archive；control parameters adaptation

0 引 言

DE是由Storn等[1]针对实参优化问题而提出的一种全局优化算法. DE原理简单,控制参数少,简单易实现. 目前,DE已经广泛应用于函数优化、神经网络优化、工业控制、图像处理等众多科研和工程领域[2-4]. DE作为进化算法的一个重要分支,与其他进化算法有着相似的计算步骤,包括变异、交叉和选择操作,而且DE也是基于群体和适应度这两个概念. 然而,DE与其他进化算法一样容易陷入局部最优, 存在早熟收敛现象. 文献[5-6]的研究表明,DE的优化性能主要受变异模式和控制参数(种群规模NP、缩放因子F和交叉概率CR)的影响. 通常情况下,当使用DE求解优化问题时,需要事先确定采用的变异策略以及设置合适的控制参数.

DE有许多不同的变异模式,每种变异模式都有自己的优势和劣势. 为了充分利用变异模式的优势并克服其劣势,相关研究人员提出了许多基于变异模式的改进DE算法. 文献[7]提出的SaDE综合运用4种变异模式,且控制参数F和CR服从正态分布;文献[8]提出的JADE算法通过对DE/current-to-best/1模式

收稿日期: 2016-06-01；**修回日期:** 2016-08-22.

基金项目: 国家自然科学基金项目(71573184)；民航科技项目(20150228).

作者简介: 王世豪(1987−), 男, 博士生, 从事智能计算及优化的研究；杨红雨(1967−), 女, 教授, 博士生导师, 从事智能计算及优化、空中交通管理等研究.

†通讯作者. E-mail: yanghongyu@scu.edu.cn

进行改进,引入了一种新的变异模式DE/current-to-pbest/1;文献[9]通过对变异策略和控制参数进行组合,提出了EPSDE算法,在EPSDE中,不同的变异模式组成一个资源池;文献[10]利用3种变异模式和3种控制参数设置的随机结合,提出了组合DE算法(CoDE),在CoDE中,对于每个个体,均有3个实验向量生成;文献[11]通过引入一种基于等级的变异模式提出了Rank-JADE算法;文献[12]提出的SAS-DE算法能够自适应地结合多种变异模式;文献[13]提出的DMPSADE可以根据个体之间的竞争动态地调整变异模式.

另外,也有许多改进DE侧重于控制参数自适应策略. 文献[14]提出了一种模糊自适应DE算法,该算法利用模糊逻辑控制来调节控制参数F和CR;文献[15]提出的jDE算法,为每一个个体X_i分配一个F_i和CR_i,而且在进化过程中根据两个指定的阈值τ_1和τ_2动态地调整F_i和CR_i;文献[16]提出的aDE(adaptive DE)通过对比后代个体的适应值$f(x_i^{t+1})$和父代种群的平均适应值f_{avg}^t来调整F和CR;文献[17]在JADE的基础上引入了一种自适应种群规模减小机制,进而提出了JADE-APTS算法; 文献[18]提出的ESADE利用自适应参数的均匀分布和柯西分布生成控制参数值.

为进一步提高DE的优化性能, 并使控制参数的选择独立于优化问题,本文提出一种DMCSaDE(self-adaptive differential evolution with dual mutation modes collaboration)算法. DMCSaDE综合运用DE/rand/1和DE/best/1两种变异模式分别具有的全局探索能力和局部开发能力,并在两种变异模式中引入精英存档策略,同时,对每个个体的控制参数F_i和CR_i采用自适应策略. 选取15个benchmark函数对DMCSaDE进行实验研究,并与基本DE(DE/rand/1/bin，DE/best/1/bin)和改进DE算法(jDE、aDE、SaDE、JADE、EPSDE、CoDE、Rank-JADE、JADE-APTS和DMPSADE)进行比较,同时,分析了精英种群的大小对DMCSaDE优化性能的影响. 实验结果表明,本文提出的DMCSaDE能够有效地避免早熟收敛,全局优化能力和鲁棒性得到了显著提高.

1　DE算法

考虑D维实数空间$S \subset R^D$为待优化问题$\min f(x)$的搜索空间. NP个D维实数向量$X_{i,t} = \{x_{i1,t}, x_{i2,t}, \cdots, x_{iD,t}\} \in S\,(i = 1, 2, \cdots, \mathrm{NP})$构成一代种群$X_t = \{X_{1,t}, X_{2,t}, \cdots, X_{\mathrm{NP},t}\}$,其中$t = 0, 1, \cdots, T$表示进化代数. 因此,DE的变异、交叉以及选择操作可如下表示:

1) 变异. 对于每个个体$X_{i,t}$,按照下式生成变异向量$V_{i,t+1} = [v_{i_1,t+1}, v_{i_2,t+1}, \cdots, v_{i_D,t+1}]$:

$$V_{i,t+1} = X_{r_1,t} + F(X_{r_2,t} - X_{r_3,t}). \tag{1}$$

其中:r_1、r_2和r_3是$[1, \mathrm{NP}]$之间的随机整数,且$r_1 \neq r_2 \neq r_3 \neq i$,因此种群规模NP至少为4;缩放因子$F$是一个介于$[0, 2]$之间的实数.

2) 交叉. 为提高种群的多样性,按下式生成实验向量$U_{i,t+1} = [u_{i_1,t+1}, u_{i_2,t+1}, \cdots, u_{i_D,t+1}]$:

$$u_{i_j,t+1} = \begin{cases} v_{i_j,t+1}, \ \mathrm{rand}_{i_j} \leqslant \mathrm{CR} \text{ or } j = j_{\mathrm{rand}}; \\ x_{i_j,t}, \ \text{otherwise}. \end{cases} \tag{2}$$

其中:$\mathrm{CR} \in [0, 1]$为交叉概率;rand_{ij}为$[0, 1]$均匀分布的随机数,决定了$U_{i,t+1}$的第j个元素是由变异向量$V_{i,t+1}$还是目标向量$X_{i,t}$所贡献;j_{rand}为$[1, D]$之间的随机整数,它确保$U_{i,t+1}$至少从$V_{i,t+1}$中获取一个元素,避免进化停滞.

3) 选择. DE采用贪婪的选择策略,选择操作可描述如下:

$$X_{i,t+1} = \begin{cases} U_{i,t+1}, \ f(U_{i,t+1}) < f(X_{i,t}); \\ X_{i,t}, \ \text{otherwise}. \end{cases} \tag{3}$$

当且仅当实验向量$U_{i,t+1}$的适应值小于$X_{i,t}$的适应值时,新个体$X_{i,t+1}$才被设置为$U_{i,t+1}$;否则,$X_{i,t+1}$保持$X_{i,t}$进入下一代.

2　DMCSaDE算法

全局探索和局部开发这对矛盾制约着DE的优化性能. 全局探索表示搜索解空间内不同区域的能力,局部开发表示接近最优解的能力. 文献[19]认为DE的局部开发能力强,而全局探索能力弱,从而容易陷入局部最优,产生早熟收敛. 当DE早熟收敛时,种群进化停滞,种群中的每个个体均会出现聚集现象,此时种群中个体的相似程度比较大,它也表明当前种群具有较差的多样性,而利用式(1)所求得的变异向量$V_{i,t+1}$与目标向量$X_{i,t}$之间的差异并不明显,从而使个体发现更优解的概率较小,很难摆脱局部最优. 因此,通过调控种群多样性可以有效提高全局探索能力,使算法跳出局部最优解.

DE有多种变异模式,每种变异模式都有自身的特点. 比如,DE/rand/1强调全局探索能力,而DE/best/1强调局部搜索能力. 两种变异模式分别为

$$V_{i,t+1} = X_{r_1,t} + F(X_{r_2,t} - X_{r_3,t}), \tag{4}$$

$$V_{i,t+1} = X_{\mathrm{best},t} + F(X_{r_2,t} - X_{r_3,t}). \tag{5}$$

由式(4)可知,DE/rand/1的变异向量$V_{i,t+1}$由3个互不相同的随机个体组成,这样有利于保持种群的多样性,因此全局探索能力强,但收敛速度较慢;由式(5)可知,DE/best/1的变异向量$V_{i,t+1}$由当前种群中最优

个体$X_{\rm best}$引导,因此局部搜索能力强,收敛速度快,但容易出现早熟收敛.

基于以上分析,本文结合DE/rand/1和DE/best/1两种变异模式分别具有的全局探索能力和局部开发能力,在两种变异模式中引入精英存档策略,提出一种DMCSaDE算法.

精英存档策略的基本原理是将当前种群Pop(NP个个体)按照个体的适应值从小到大划分为精英种群P和非精英种群Q. P中保留前NEP个较优的个体(适应值较小),Q保留剩余的NP − NEP个个体. P和Q满足$P\bigcup Q = {\rm Pop}$,且$P\bigcap Q = \varnothing$. 当DMCSaDE采用DE/rand/1模式时,参与变异的个体$X_{r_1,t}$和$X_{r_2,t}$来自P,$X_{r_3,t}$来自Q. 当DMCSaDE采用DE/best/1模式时,参与变异的个体$X_{r_2,t}$来自P,$X_{r_3,t}$来自Q. 由此可得,DMCSaDE的变异和交叉操作如下所示:

$$V_{i,t+1} = \begin{cases} X^P_{r_1,t}+F_{i,t}(X^P_{r_2,t}-X^Q_{r_3,t}),\ \tau < 1-\left(\dfrac{t}{T}\right)^2; \\ X_{{\rm best},t}+F_{i,t}(X^P_{r_2,t}-X^Q_{r_3,t}),\ {\rm otherwise}; \end{cases} \tag{6}$$

$$u_{i_j,t+1} = \begin{cases} v_{i_j,t+1},\ {\rm rand}_{i_j} \leqslant {\rm CR}_{i,t}\ {\rm or}\ j = j_{\rm rand}; \\ x_{i_j,t},\ {\rm otherwise}. \end{cases} \tag{7}$$

其中:τ是$[0,1]$之间的随机数,它决定了变异向量$V_{i,t+1}$由DE/rand/1模式或DE/best/1模式生成;t为进化代数计数器;T为算法的最大进化代数. 此外,每个个体都有自己的控制参数$F_{i,t}$和${\rm CR}_{i,t}$,且随种群进化而自适应调整.

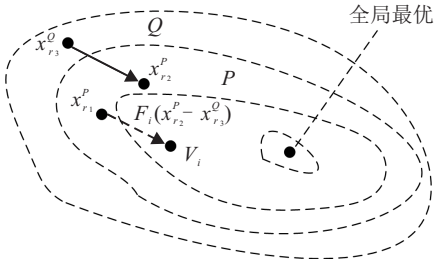


图1 DMCSaDE的DE/rand/1模式(进化初期)

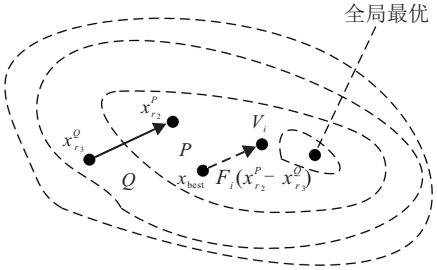


图2 DMCSaDE的DE/best/1模式(进化后期)

图1和图2所示的分别为采用精英存档策略改进后的DE/rand/1和DE/best/1变异模式,两者主要区别在于参与变异操作的个体$X^P_{r_1}$和$X_{\rm best}$;两者共同的部分为差分向量$X^P_{r_2}-X^Q_{r_3}$,从矢量运算的角度出发,差分向量朝着更优个体$X^P_{r_2}$的矢量方向进行搜索,更加有利于提高收敛速度和求解精度.

新的变异模式(6)表明,进化前期,$1-(t/T)^2$的值较大,τ小于$1-(t/T)^2$的概率较大,图1所示的DE/rand/1模式更多地被选用,个体$X^P_{r_1}$、$X^P_{r_2}$和$X^Q_{r_3}$共同参与变异操作,能够有效地保持种群多样性,从而尽可能地对搜索空间进行全方位探索以发现更多有“希望”的区域,避免了算法早熟收敛;进化后期,$1-(t/T)^2$的值在不断减小,τ大于$1-(t/T)^2$的概率在不断增加,图2所示的DE/best/1模式被更多地用来提高算法的局部开发能力,该模式以$X_{\rm best}$为引导,将对探索到的有“希望”的区域进行开发以提高求解精度,同时也加快了收敛速度.

精英种群P主要用来提高算法的收敛速度,而非精英种群Q主要用来调节种群多样性. 当P的大小NEP = 0或NEP = NP时,参与变异的个体$X_{r_1,t}$、$X_{r_2,t}$和$X_{r_3,t}$随机来自种群Pop,DMCSaDE的收敛速度和求解精度将降低. 当P的大小NEP = 1时,DE/rand/1模式中的个体$X_{r_1,t}$相当于$X_{{\rm best},t}$,而$X_{r_2,t}$和$X_{r_3,t}$来自Q,此时DE/rand/1模式退化为DE/best/1模式,不利于保持种群多样性,DMCSaDE的求解精度将下降. 由此可知,过大的NEP有利于保持种群多样性,但影响DMCSaDE的局部搜索能力,不利于算法收敛;过小的NEP不利于保持种群多样性,影响DMCSaDE的全局探索能力. 本文建议${\rm NEP} \in [0.3, 0.5]\cdot {\rm NP}$(详见第3.2节).

进化过程中,需要随时对P和Q进行更新维护,当且仅当生成的新个体$U_{i,t+1}$比父代个体$X_{i,t}$更优时才进行更新,更新操作存在以下两种情况:

1) 如果新个体$U_{i,t+1}$对应的父个体$X_{i,t}$已经存在于精英种群P,则直接用$U_{i,t+1}$替换$X_{i,t}$;

2) 如果新个体$U_{i,t+1}$对应的父个体$X_{i,t}$不在P中,且$U_{i,t+1}$优于P中最差的个体$X_{{\rm worst},t}$(适应值最大),则用$U_{i,t+1}$替换$X_{{\rm worst},t}$,同时将Q中对应于$U_{i,t+1}$的个体替换为$X_{{\rm worst},t}$.

此外,进化过程中,如果任何个体$X_{i,t}$连续ST代不能生成一个更优个体(个体进化停滞),则表明该个体当前的控制参数$F_{i,t}$和${\rm CR}_{i,t}$不合适而应当重新设置;反之,则认为该个体当前的控制参数是合适的而应当保留进入下一代. 文献[16]建议当参数不合适需要重新设置时,采用随机调整的方法. 在文献[16]中,当后代个体的适应值$f(x_i^{t+1})$大于等于父代种群的平均适应值$f^t_{\rm avg}$时,需要重新生成参数,但此时并不能表明个体的进化受阻,反而可能会丢失有效的

参数值,造成搜索不够充分. 因此本文根据进化过程中个体的进化状态,提出如下一种控制参数自适应策略. 该策略能够使个体的当前参数值保留ST代,从而尽可能地对有“希望”的区域进行搜索,进而提高求解精度.

$$F_{i,t+1}=\begin{cases}F_{i,t},\ \mathrm{st}_i<\mathrm{ST};\\ F_l+r_1(F_u-F_l),\ \mathrm{st}_i\geqslant\mathrm{ST};\end{cases}\tag{8}$$

$$\mathrm{CR}_{i,t+1}=\begin{cases}\mathrm{CR}_{i,t},\ \mathrm{st}_i<\mathrm{ST};\\ \mathrm{CR}_l+r_2(\mathrm{CR}_u-\mathrm{CR}_l),\ \mathrm{st}_i\geqslant\mathrm{ST}.\end{cases}\tag{9}$$

其中: $F_{i,t}$ 和 $\mathrm{CR}_{i,t}$ 分别为个体 $X_{i,t}$ 当前的缩放因子和交叉概率, F_l 和 F_u 分别为缩放因子的下限和上限, CR_l 和 CR_u 分别为交叉概率的下限和上限. 根据文献[17]的建议,设置 $F_l=0.1$、$F_u=1.0$、$\mathrm{CR}_l=0.3$、$\mathrm{CR}_u=1.0$; r_1、r_2 为 $[0,1]$ 之间的随机数; st_i 为个体 $X_{i,t}$ 的进化停滞代数,整数ST为指定的阈值,根据文献[20]的建议,设置 $\mathrm{ST}=3$.

控制参数 $F_{i,t}$ 和 $\mathrm{CR}_{i,t}$ 的更新在变异操作之前,从而新的控制参数将影响到下一代个体的变异和交叉操作. 通过采用控制参数自适应策略,使控制参数的选择独立于具体的优化问题,提高了算法的适用性和鲁棒性.

3　实验及结果分析

为检验DMCSaDE算法的优化性能,选用15个典型的benchmark测试函数进行实验研究,如下所示:

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

$$f_2(x)=\sum_{i=1}^{D}|x_i|+\prod_{i=1}^{D}|x_i|;$$

$$f_3(x)=\sum_{i=1}^{D}\Big(\sum_{j=1}^{i}x_j\Big)^2;$$

$$f_4(x)=\max\{\,|x_i|,\ 1\leqslant i\leqslant D\};$$

$$f_5(x)=\sum_{i=1}^{D-1}(100(x_{i+1}-x_i^2)^2+(1-x_i)^2);$$

$$f_6(x)=\sum_{i=1}^{D}(\lfloor x_i+0.5\rfloor)^2;$$

$$f_7(x)=\sum_{i=1}^{D}ix_i^4+\mathrm{random}[0,1);$$

$$f_8(x)=\sum_{i=1}^{D}-x_i\sin(\sqrt{|x_i|})+D\cdot 418.982\,887\,272\,336\,9;$$

$$f_9(x)=\sum_{i=1}^{D}(10^6)^{\frac{i-1}{D-1}}x_i^2;$$

$$f_{10}(x)=-20\exp\Big(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}\Big)-\exp\Big(\frac{1}{D}\sum_{i=1}^{D}\cos 2\pi x_i\Big)+20+e;$$

$$f_{11}(x)=\sum_{i=1}^{D}\frac{x_i^2}{4\,000}-\prod_{i=1}^{D}\cos\Big(\frac{x_i}{\sqrt{i}}\Big)+1;$$

$$f_{12}(x)=1-\cos\Big(2\pi\sqrt{\sum_{i=1}^{D}x_i^2}\Big)+0.1\sqrt{\sum_{i=1}^{D}x_i^2};$$

$$f_{13}(x)=\sum_{i=1}^{D}\Big(0.5+\frac{\sin^2(\sqrt{x_i^2+x_{i+1}^2})-0.5}{(1+0.001(x_i^2+x_{i+1}^2))^2}\Big),\quad x_{D+1}=x_1;$$

$$f_{14}(x)=\frac{\pi}{D}\Big\{10\sin^2(\pi y_1)+\sum_{i=1}^{D-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})]+(y_D-1)^2\Big\}+\sum_{i=1}^{D}u(x_i,10,100,4),\ y_i=1+\frac{1}{4}(x_i+1);$$

$$f_{15}(x)=0.1\Big\{10\sin^2(3\pi x_1)+\sum_{i=1}^{D-1}(x_i-1)^2[1+\sin^2(3\pi x_{i+1})]+(x_D-1)^2[1+\sin^2(2\pi x_D)]\Big\}+\sum_{i=1}^{D}u(x_i,5,100,4).$$

3.1　与基本DE和改进DE进行比较

将DMCSaDE与基本DE(DE/rand/1/bin和DE/best/1/bin)、jDE和aDE进行比较. 对所有算法, $\mathrm{NP}=100$, 独立实验次数为30. 根据文献[15]的建议,对于DE/rand/1/bin,设置 $F=0.5$ 和 $\mathrm{CR}=0.9$. 根据文献[10]的建议,对于DE/best/1/bin,设置 $F=0.8$ 和 $\mathrm{CR}=0.9$. 对于DMCSaDE,精英种群大小 $\mathrm{NEP}=30$, $\mathrm{ST}=3$. 其他算法的参数设置详见具体文献. 表1给出了各算法所求最优解的均值(Mean)和达到指定收敛精度时所需的平均进化代数(MNEG)以及实验成功率(SR),其中N/A表示Not Applicable. MNEG和SR主要用于比较算法的收敛速度和稳定性. 此外,图3给出了部分函数的进化曲线. 横轴和纵轴fit分别表示进化代数和log10(平均最优适应值). 图3(b)、图3(c)和图3(d)的图例与图3(a)相同.

由表1和图3可得: 1)就所求最优解的精度来看(Mean): DMCSaDE对所有测试函数求得的最优解均值都要好于其他算法. 2)就算法的收敛速度来看(MNEG): DMCSaDE能够以最少的进化代数达到指定的收敛精度;而DE/rand/1bin、DE/best/1/bin、jDE和aDE表现较差,特别是对于多峰函数. 3)就算法的稳定性来看(SR): DMCSaDE的稳定性最高; DE/rand/1/bin和DE/best/1/bin的表现最差,这是由于前者在有

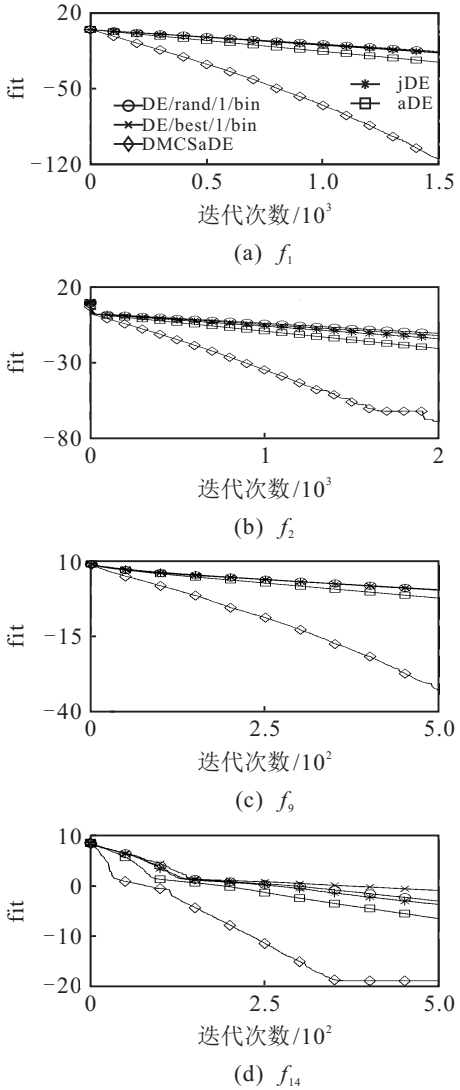
限的进化代数内收敛太慢,而后者采用贪婪策略导致早熟收敛;jDE和aDE要好于DE/rand/1/bin和DE/best/1/bin,但对于具有多个局部最优解的多峰函数,优化结果并不理想.

综上所述,DMCSaDE不仅具有很高的收敛精度和稳定性,而且具有很快的收敛速度,这主要是由于DMCSaDE采用了双变异模式(DE/rand/1和DE/best/1)协同策略进而提高了种群的多样性,同时精英存档策略和控制参数自适应策略的引入不仅加快了算法的收敛速度,也提高了算法的稳定性.

此外,有7个函数与CEC2005的测试函数相同,不同的是这些函数在CEC2005中经过了偏移和旋转变换. 为进一步验证DMCSaDE的有效性,将其与SaDE、JADE、EPSDE、CoDE、Rank-JADE、JADE-APTS和DMPSADE进行对比. DMCSaDE的参数设置不变,其他算法的设置见具体文献. 独立实验次数和函数维度均为30,最大进化代数为3 000. f_5和f_{13}的搜索空间为$[-100,100]$,其他不变. 表2给出了各算法所求最优解的均值. 从表2可以看出,DMCSaDE的优化结果好于其他参与比较的算法.

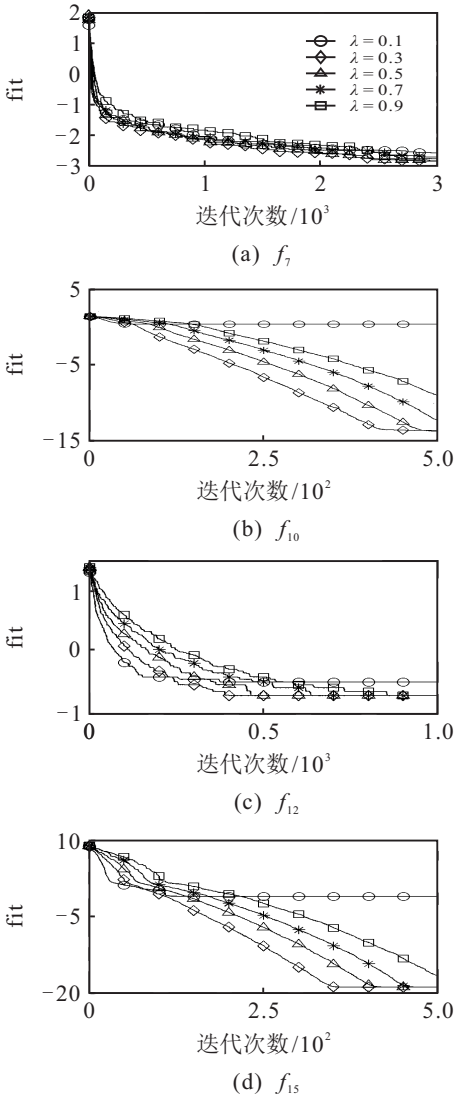
3.2 参数NEP对DMCSaDE的影响

在第2节中,初步探讨了精英种群P的大小NEP对DMCSaDE的影响. 但如何针对不同的优化问题,选择合适范围内的NEP才是研究的关键. 为此,通过设置不同的NEP来进一步研究其对DMCSaDE优化性能的影响,进而确定合适NEP的取值范围. 种群大小$\mathrm{NP}=100$,独立实验次数为30,DMCSaDE的参数设置同3.1节,设置$\mathrm{NEP}=\lambda\cdot\mathrm{NP},\lambda\in\{0.1,0.3,0.3,0.7,0.9\}$. 表3给出了DMCSaDE在不同NEP下所求解得的最优解均值Mean和达到指定收敛精度所需的平均进化代数均值MNEG以及实验成功率SR,其中最优(Best)和次优(Second)结果分别用粗体和斜体表示. 同时,表3统计了最优解和次优解的个数. 图4也给出了部分函数的进化曲线. 横轴和纵轴fit分别表示进化代数和log10(平均最优适应值). 图4(b)、图4(c)和图4(d)的图例与图4(a)相同.



(a) f_1 (b) f_2 (c) f_9 (d) f_{14}

图3 不同算法对测试函数进行优化后的平均最优适应值进化曲线



(a) f_7 (b) f_{10} (c) f_{12} (d) f_{15}

图4 DMCSaDE在不同NEP下的平均最优适应值进化曲线

表 1 基本DE、jDE、aDE和DMCSaDE的优化结果

函数	T	DE/rand/1/bin		DE/best/1/bin		jDE		aDE		DMCSaDE	
		Mean	MNEG(SR)/%	Mean	MNEG(SR)/%	Mean	MNEG(SR)/%	Mean	MNEG(SR)/%	Mean	MNEG(SR)/%
f_1	1 500	2.63e-16	932(100)	4.09e-17	891(100)	1.12e-17	870(100)	3.77e-26	625(100)	**1.32e-114**	202(100)
f_2	2 000	1.27e-11	1 557(100)	4.46e-13	1 381(100)	4.09e-15	1 221(100)	1.34e-21	871(100)	**2.46e-69**	299(100)
f_3	5 000	5.96e-12	3 914(100)	2.32e-13	3 576(100)	4.39e-03	N/A(0)	4.64e-05	N/A(0)	**8.96e-52**	1 345(100)
f_4	5 000	2.60e-01	N/A(3)	4.37e-09	N/A(83)	8.62e-10	4 466(100)	6.99e-05	N/A(60)	**3.63e-21**	1 915(100)
f_5	3 000	5.72e-03	N/A(0)	2.65e-01	N/A(60)	9.56e+00	N/A(0)	1.06e+01	N/A(0)	**9.32e-28**	996(100)
f_6	1 500	**0.00e+00**	389(100)	6.66e-01	N/A(57)	**0.00e+00**	309(100)	**0.00e+00**	277(100)	**0.00e+00**	82(100)
f_7	3 000	4.62e-03	147 4(100)	7.58e-03	N/A(87)	5.39e-03	1 734(100)	5.16e-03	1 645(100)	**1.14e-03**	476(100)
f_8	1 000	7.46e+03	N/A(0)	1.42e+03	N/A(0)	5.24e+02	N/A(0)	9.27e-09	N/A(83)	**0.00e+00**	618(100)
f_9	500	4.24e+00	N/A(0)	2.55e+00	N/A(0)	2.82e+00	N/A(0)	6.37e-03	N/A(0)	**7.09e-34**	240(100)
f_{10}	500	2.84e-02	N/A(0)	7.67e-02	N/A(0)	1.16e-02	N/A(0)	5.35e-04	N/A(0)	**2.88e-14**	287(100)
f_{11}	1 000	5.41e-09	N/A(73)	8.21e-03	N/A(53)	2.41e-09	N/A(93)	2.52e-14	674(100)	**0.00e+00**	212(100)
f_{12}	1 000	2.85e-01	688(100)	3.43e-01	732(100)	3.10e-01	728(100)	2.90e-01	636(100)	**1.99e-01**	207(100)
f_{13}	500	3.28e-07	N/A(0)	1.81e-07	N/A(0)	1.14e-07	N/A(0)	1.89e-10	413(100)	**0.00e+00**	134(100)
f_{14}	500	8.63e-04	N/A(0)	1.17e-01	N/A(0)	1.86e-04	N/A(0)	2.85e-07	N/A(0)	**1.35e-19**	198(100)
f_{15}	500	5.52e-03	N/A(0)	1.34e-02	N/A(0)	1.53e-03	N/A(0)	2.12e-06	N/A(0)	**1.29e-19**	210(100)

表 2 DMCSaDE与其他改进DE的优化结果

函数	SaDE	JADE	EPSDE	CoDE	Rank-JADE	JADE-APTS	DMPSADE	DMCSaDE
f_1	2.96e-30	**0.00e+00**	1.02e-29	3.12e-31	**0.00e+00**	**0.00e+00**	**0.00e+00**	**0.00e+00**
f_3	9.33e-02	4.15e-27	4.12e-22	4.25e-09	7.03e-27	2.10e-29	2.34e-05	**1.50e-29**
f_5	8.24e+01	6.70e-01	1.23e+00	9.86e-01	1.04e+00	8.59e+00	6.23e-01	**3.58e-01**
f_9	8.48e+05	2.01e+04	9.28e+06	1.87e+05	1.46e+04	7.43e+03	3.50e+05	**3.89e+03**
f_{10}	2.11e+01	2.10e+01	2.11e+01	**2.01e+01**	2.10e+01	2.09e+01	2.11e+01	**2.01e+01**
f_{11}	5.23e-03	4.10e-03	7.44e-03	5.11e-03	4.97e-03	3.05e-03	4.61e-04	**5.85e-06**
f_{13}	2.22e+01	2.18e+01	2.30e+01	2.20e+01	2.17e+01	1.19e+01	2.19e+01	**8.27e+00**

表 3 DMCSaDE在不同NEP下的优化结果

函数	T	$\lambda=0.1$		$\lambda=0.3$		$\lambda=0.5$		$\lambda=0.7$		$\lambda=0.9$	
		Mean	MNEG(SR)/%	Mean	MNEG(SR)/%	Mean	MNEG(SR)/%	Mean	MNEG(SR)/%	Mean	MNEG(SR)/%
f_1	1 500	**1.43e-153**	137(100)	*1.32e-114*	202(100)	1.94e-93	290(100)	5.41e-80	385(100)	2.92e-60	499(100)
f_2	2000	**1.59e-78**	216(100)	*2.46e-69*	299(100)	2.92e-59	421(100)	2.46e-45	558(100)	1.16e-37	722(100)
f_3	5 000	**3.38e-76**	777(100)	*8.96e-52*	1 345(100)	9.81e-41	1 881(100)	4.97e-30	2 488(100)	3.25e-17	3 400(100)
f_4	5 000	1.85e-17	2 490(100)	3.63e-21	1 915(100)	*9.55e-24*	1 962(100)	**9.21e-24**	2 104(100)	4.48e-21	2 478(100)
f_5	3 000	2.31e-20	868(100)	9.32e-28	996(100)	*8.69e-28*	1 333(100)	**6.21e-28**	1 643(100)	2.02e-21	N/A(67)
f_6	1 500	4.00e+00	N/A(0)	**0.00e+00**	82(100)	**0.00e+00**	112(100)	**0.00e+00**	152(100)	**0.00e+00**	204(100)
f_7	3 000	2.66e-03	902(100)	**1.14e-03**	476(100)	*1.47e-03*	790(100)	1.93e-03	700(100)	1.70e-03	1 225(100)
f_8	1 000	9.14e+02	N/A(0)	**0.00e+00**	618(100)	**0.00e+00**	739(100)	**0.00e+00**	850(100)	3.55e+02	N/A(0)
f_9	500	**8.39e-43**	169(100)	*7.09e-34*	240(100)	3.61e-26	303(100)	3.38e-20	359(100)	2.86e-14	421(100)
f_{10}	500	2.11e+00	N/A(0)	*2.88e-14*	287(100)	**2.29e-14**	347(100)	7.11e-13	406(100)	1.16e-09	470(100)
f_{11}	1 000	2.52e-02	N/A(33)	**0.00e+00**	212(100)	**0.00e+00**	309(100)	**0.00e+00**	375(100)	**0.00e+00**	474(100)
f_{12}	1 000	3.33e-01	135(100)	*1.99e-01*	207(100)	**1.99e-01**	263(100)	1.99e-01	357(100)	1.99e-01	440(100)
f_{13}	500	7.03e-16	91(100)	**0.00e+00**	134(100)	**0.00e+00**	184(100)	**0.00e+00**	229(100)	3.70e-16	274(100)
f_{14}	500	2.77e-01	N/A(67)	**1.35e-19**	198(100)	*1.35e-19*	244(100)	1.35e-19	297(100)	2.92e-18	345(100)
f_{15}	500	7.78e-02	N/A(33)	**1.29e-19**	210(100)	*1.29e-19*	268(100)	1.29e-19	306(100)	1.87e-17	363(100)
Best(Second)		4(0)		7(6)		6(5)		6(0)		2(0)	

从表3和图4可以得出:$\lambda = 0.1$,即$\mathrm{NEP} = 10$时,对于单峰函数$f_1 \sim f_3$和f_9,DMCSaDE表现出最高的求解精度和收敛速度;对于多峰函数,DMCSaDE明显表现出求解精度低,稳定性差等缺点. $\lambda = 0.9$时,DMCSaDE的求解精度最低,且收敛速度最慢. $\lambda = 0.7$时,DMCSaDE的优化效果要好于$\lambda \in \{0.1, 0.9\}$. 相比之下,$\lambda \in \{0.3, 0.5\}$时,DMCSaDE所求得的最优解和次优解个数最多,且仍保持很高的收敛速度和稳定性. 因此,过大或过小的NEP都将降低DMCSaDE的优化性能. 当$\lambda \in [0.3, 0.5]$,即$\mathrm{NEP} \in [0.3, 0.5] \cdot \mathrm{NP}$时,DMCSaDE能够针对大多数优化问题发挥出最好的优化性能. 此外,对于高维优化问题,较小的NP将不利于算法寻优,而较大的NP虽然能够提高求解精度,但会增加算法的运算量. 根据文献[1]的建议,$\mathrm{NP} \in [3, 10] \cdot D$.

4 结 论

本文研究了DE算法的早熟收敛问题,通过结合DE/rand/1和DE/best/1两种变异模式的优点,提出了一种DMCSaDE算法. 同时,在DMCSaDE中引入了精英存档策略和控制参数自适应策略,这使得算法的收敛速度、求解精度以及稳定性得到了显著提高. 15个常用于优化算法比较的基准函数的实验结果表明了DMCSaDE的整体优化性能要好于DE/rand/1/bin、DE/best/1/bin、jDE、aDE、SaDE、JADE、EPSDE、CoDE、Rank-JADE、JADE-APTS和DMPSADE. 此外,分析了参数NEP对DMCSaDE性能的影响,并给出了NEP的取值范围.

参考文献(References)

[1] Storn R, Price K. Differential evolution: A simple and efficient adaptive scheme for global optimization over continuous spaces[R]. California: University of California, Berkeley, 1995.

[2] Ghosh A, Datta A, Ghosh S. Self-adaptive differential evolution for feature selection in hyperspectral image data[J]. Applied Soft Computing, 2013, 13(4): 1969-1977.

[3] Marcic T, Stumberger B, Stumberger G. Differential evolution based parameter identification of a line-start IPM synchronous motor[J]. IEEE Trans on Industrial Electronics, 2014, 61(11): 5921-5929.

[4] Kadhar K M A, Baskar S, Amali S M J. Diversity controlled self-Adaptive differential evolution based design of non-fragile multivariable PI controller[J]. Engineering Applications of Artificial Intelligence, 2015, 46: 209-222.

[5] Gamperle R, Muller S D, Koumoutsakos P. A parameter study for differential evolution[C]. WSEAS Int Conf on Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation. New York: WSEAS, 2002: 293-298.

[6] Caponio A, Neri F, Tirronen V. Super-fit control adaptation in memetic differential evolution frameworks[J]. Soft Computing, 2009, 13(8): 811-831.

[7] Qin A K, Huang V L, Suganthan P N. Differential evolution algorithm with strategy adaption for global numerical optimization[J]. IEEE Trans on Evolutionary Computation, 2009, 13: 398-417.

[8] Zhang J, Sanderson A C. JADE: Adaptive differential evolution with optional external archive[J]. IEEE Trans on Evolutionary Computation, 2009, 13(5): 945-958.

[9] Mallipeddi R, Suganthan P, Pan Q, et al. Differential evolution algorithm with ensemble of parameters and mutation strategies[J]. Applied Soft Computing, 2011, 11(2): 1679-1696.

[10] Wang Y, Cai Z, Zhang Q. Differential evolution with composite trial vector generation strategies and control parameters[J]. IEEE Trans on Evolutionary Computation, 2011, 15(1): 55-66.

[11] Gong W, Cai Z. Differential evolution with ranking-based mutation operators[J]. IEEE Trans on Cybernetics, 2013, 43: 2066-2081.

[12] Elsayed S M, Sarker R A, Essam D L. A self-adaptive combined strategies algorithm for constrained optimization using differential evolution[J]. Applied Mathematics and Computation, 2013, 241: 267-282.

[13] Fan Q, Yan X. Self-adaptive differential evolution algorithm with discrete mutation control parameters[J]. Expert Systems with Applications, 2015, 42(3): 1551-1572.

[14] Liu J, Lampinen J. A fuzzy adaptive differential evolution algorithm[J]. Soft Computing, 2005, 9(6): 448-462.

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

[16] Nasimul N, Danushka B, Hitoshi I. An adaptive differential evolution algorithm[C]. IEEE Congress on Evolutionary Computation. New Orleans: IEEE Press, 2011: 2229-2236.

[17] Zhu W, Tang Y, Fang J, et al. Adaptive population tuning scheme for differential evolution[J]. Information Sciences, 2013, 223: 164-191.

[18] 赵志伟, 杨景明, 呼子宇, 等. 基于一次指数平滑法的自适应差分进化算法[J]. 控制与决策, 2016, 31(5): 790-796.
(Zhao Z W, Yang J M, Hu Z Y, et al. Self-adaptive differential evolution algorithm based on exponential smoothing[J]. Control and Decision, 2016, 31(5): 790-796.)

[19] Chakrabory U K, Das S, Konar A. Differential evolution with local neighborhood[C]. IEEE Congress on Evolutionary Computation. Vancouver: IEEE Press, 2006: 2042-2049.

[20] 胡旺, 李志蜀. 一种更简化而高效的粒子群优化算法[J]. 软件学报, 2007, 18(4): 861-866.
(Hu W, Li Z S. A simple and more effective particle swarm optimization algorithm[J]. J of Software, 2007, 18(4): 861-866.)

（责任编辑：齐 霁）