基于搜索偏好知识的复杂多模差分进化算法

陈作汉[1,2]，曹 洁[1,2*]，赵付青[1]，张建林[1]

(1. 兰州理工大学计算机与通信学院 兰州 730050；2. 甘肃省制造业信息化工程技术研究中心 兰州 730050)

【摘要】针对复杂多模优化问题，提出一种基于搜索偏好知识的差分进化算法 PKLSHADE。PKLSHADE 将先验搜索偏好知识注入到种群的进化过程，在不同的进化阶段对种群的多样性和集约性区分考虑，进化早期重视差分扰动以增强算法的全局开发能力，进化后期更多围绕当前最优解进行局部精细搜索。同时，基于搜索偏好知识的变异策略能够实现差分进化算法全局开发和局部搜索的自适应平滑过渡，避免两搜索阶段的硬切换。在 CEC2017 复杂混合多模函数上的实验结果及统计分析表明，PKLSHADE 在最优解的精度、算法的稳定性等方面均优于 LSHADE、EBLSHADE、jSO 及 AMECoDEs 等近年来的优秀差分进化算法。

关 键 词 差分进化； 复杂多模优化； 变异策略； 偏好知识

中图分类号 TP18 **文献标志码** A doi:10.12178/1001-0548.2020153

Complex Multimodal Differential Evolution Algorithm Based on Search Preference Knowledge

CHEN Zuo-han[1,2], CAO Jie[1,2*], ZHAO Fu-qing[1], and ZHANG Jian-lin[1]

(1. College of Computer and Communication, Lanzhou University of Technology Lanzhou 730050;
2. Gansu Engineering Research Center of Manufacturing Informationization Lanzhou 730050)

Abstract A differential evolutionary (DE) algorithm based on search preference knowledge — PKLSHADE (i.e. preference-knowledge-based LSHADE) is proposed for complex multimodal optimization. PKLSHADE applies prior search preference knowledge in the evolutionary process of the population and differentiates the diversity and convergence of the population at different evolutionary stages, i.e., the importance is attached to the perturbation in the early stages of evolution to enhance the global development of the algorithm, and more local searches centering around the current optimal solution are carried out in the later stage. At the same time, the mutation strategy based on search preference knowledge can realize the global development of differential evolutionary algorithm and the smooth adaptive transition of local search to avoid the direct switch of the two search stages. The experimental results on the complex multimodal functions of CEC2017 show that PKLSHADE is superior to recent excellent algorithms such as LSHADE, EBLSHADE, jSO and AMECoDEs in terms of the accuracy of the optimal solution and the stability of the algorithm.

Key words differential evolutionary (DE); multimodal optimization; mutation strategies; preference knowledge

差分进化算法 (DE)[1] 是一种基于种群的进化算法，算法利用种群个体之间的差分信息形成扰动实现变异，通过父、子代个体适应度竞争选择新一代个体，经过差分变异、交叉和选择等操作迭代寻优。具有实现简单、控制参数较少和高效等优点，广泛应用于各领域。

为提升标准 DE 算法的优化性能，国内外学者对其进行了大量研究和改进[2]。这些改进可以分为两个方面，一方面是在 DE 算法的内在运行机制上进行改进，如 SHADE[3]、LSHADE[4]、jSO[5]、EBLSHADE[6]、AMECoDEs[7] 等。这些研究从 DE 算法控制参数的自适应、变异策略和种群个体保优选择等方面进行改进。另一方面，与其他优秀算法结合，利用其他算法的优点从初始种群质量、变异

收稿日期：2020－03－24；修回日期：2020－07－15
基金项目：国家自然科学基金 (61663023，61763028)
作者简介：陈作汉 (1978－)，男，博士生，主要从事智能优化算法和无线传感网络等方面的研究.
通信作者：曹洁，E-mail：caoj@lut.edu.cn

策略等方面提升标准 DE算法的寻优性能[8-10]。

尽管 DE 算法已经被学者广泛研究，但依然缺乏对问题隐性知识的考虑，存在容易陷入局部最优，出现早熟收敛或找不到全局最优解，求解复杂多模问题比较困难等情况。

文献 [11] 的研究表明 DE 算法在进化的不同阶段，有必要采取不同的开发 (exploitation) 和探测 (exploration) 策略，对于全局开发和局部搜索给予不同的重视程度。此类算法通过设置一个阈值切换全局开发和局部搜索两个阶段，取得了一定的效果。然而，DE 算法的进化过程从全局开发到局部搜索是一个复杂渐进过程，与具体优化问题的解空间特征及规模等密切相关，确定一个合理的阈值比较困难。另一方面，一个阈值是否适用于众多不同的优化问题也值得商榷，此外新增一个待定参数，也增加了算法的复杂性和不确定性。

本文针对复杂多模函数优化问题，提出一种基于搜索偏好知识的 DE 算法，将搜索偏好知识注入到种群的演化过程，在不同进化时期，对种群的分散性和集约性给予不同的重视程度。在进化初期，更重视差分扰动加强算法的全局开发能力，进化后期重视当前最优解，在当前最优解附近进行局部搜索。通过全局开发和局部搜索的自适应平滑切换平衡全局开发和局部搜索。

1 经典 DE 算法

DE 算法是一种群体智能优化算法，首先在解的取值范围内采用实数编码随机产生一定数量的初始种群，然后经过差分变异、交叉和选择操作生成新一代种群，直到达到预设最大迭代次数或求解精度则终止迭代。

1) 初始化

设种群规模为N_p，问题的维数为 D，初始化表示如下：

$$\boldsymbol{X}^0=(x_1^0,x_2^0,\cdots,x_{N_p}^0) \tag{1}$$

$$\boldsymbol{x}_i^0=(x_{i,1}^0,x_{i,2}^0,\cdots,x_{i,D}^0)\quad i\in\{1,2,\cdots N_p\} \tag{2}$$

$$x_{i,j}^0=x_{\min,j}+\text{rand}[0,1](x_{\max,j}-x_{\min,j}) \tag{3}$$

式中，$\boldsymbol{X}^0$是初始种群；$\boldsymbol{x}_i^0$是种群的个体；rand[0,1]是 0～1 之间的随机数；$x_{\max,j}$和$x_{\min,j}$是上下界。

2) 差分变异

差分变异操作是 DE 算法最显著的特征，通过种群个体间的差分项作用于其他个体得到变异向量，根据变异向量生成方法的不同，形成了多种变异策略，部分具有代表性的变异策略如下：

DE/rand/1：

$$\boldsymbol{V}_i^t=\boldsymbol{X}_{r1}^t+F(\boldsymbol{X}_{r2}^t-\boldsymbol{X}_{r3}^t) \tag{4}$$

DE/best/1：

$$\boldsymbol{V}_i^t=\boldsymbol{X}_{\text{best}}^t+F(\boldsymbol{X}_{r1}^t-\boldsymbol{X}_{r2}^t) \tag{5}$$

DE/current-to-best/1:

$$\boldsymbol{V}_i^t=\boldsymbol{X}_i^t+F(\boldsymbol{X}_{\text{best}}^t-\boldsymbol{X}_i^t)+F(\boldsymbol{X}_{r1}^t-\boldsymbol{X}_{r2}^t) \tag{6}$$

DE/rand/2:

$$\boldsymbol{V}_i^t=\boldsymbol{X}_{\text{best}}^t+F(\boldsymbol{X}_{r1}^t-\boldsymbol{X}_{r2}^t)+F(\boldsymbol{X}_{r3}^t-\boldsymbol{X}_{r4}^t) \tag{7}$$

DE/best/1:

$$\boldsymbol{V}_i^t=\boldsymbol{X}_{r1}^t+F(\boldsymbol{X}_{r2}^t-\boldsymbol{X}_{r3}^t)+F(\boldsymbol{X}_{r4}^t-\boldsymbol{X}_{r5}^t) \tag{8}$$

式中，r_1,r_2,r_3,r_4,r_5随机从$[1,N_p]$中选取且与i互不相等；F为差分项缩放因子；$\boldsymbol{X}_{\text{best}}^t$为第$t$代的最优解。

3) 交叉

差分变异向量与目标向量进行交叉操作生成实验向量。DE 算法常见的交叉方式有二项式交叉和指数交叉。二项式交叉过程如下：

$$\boldsymbol{U}_{i,j}^t=\begin{cases}\boldsymbol{V}_{i,j}^t & j=j_{\text{rand}}\text{或}\text{rand}(0,1)\leqslant C_r\\ \boldsymbol{X}_{i,j}^t & \text{其他}\end{cases} \tag{9}$$

式中，C_r是交叉概率；j_{rand}从$[1,D]$之间随机选取；交叉过程保证$\boldsymbol{U}_{i,j}^t$至少有一个分量由$\boldsymbol{V}_{i,j}^t$贡献。

4) 选择

选择操作从目标向量 (父代) 与实验向量 (子代) 之间根据适应度值选择优秀个体进入下一代。

$$\boldsymbol{U}_{i,j}^t=\begin{cases}\boldsymbol{U}^t & f(\boldsymbol{U}^t)\leqslant f(\boldsymbol{X}_i^t)\\ \boldsymbol{X}_i^t & \text{其他}\end{cases} \tag{10}$$

对于最小化问题，如果新的实验向量的适应度函数值小于或等于目标向量的适应度值，则代表实验向量更优，选其进入下一代。

不同的差分变异操作具有不同寻优机理，如 LSHADE 采用的 current-to-best/1 差分变异策略，在整个进化过程，差分项$\boldsymbol{X}_{r1}^t-\boldsymbol{X}_{r2}^t$始终与向当前最优解学习项$\boldsymbol{X}_{\text{best}}^t-\boldsymbol{X}_i^t$保持同等权重 F，如图 1 所示。

除 DE/current-to-best/1 外， DE/rand/2 和 DE/best/2 的变异策略类似，都对差分项和向当前最优解学习项给予同一权重因子 F，即算法在运行全周期对解空间的全局开发和在最优解附近小范围内搜索一直同等重视。更为合理的选择尤其是对于多模优化问题，应该在进化前期更重视解空间的大范围

开发，尽量全面探测解空间，进化后期则主要在已找到的最优解附近进行精细搜索，加快算法收敛。

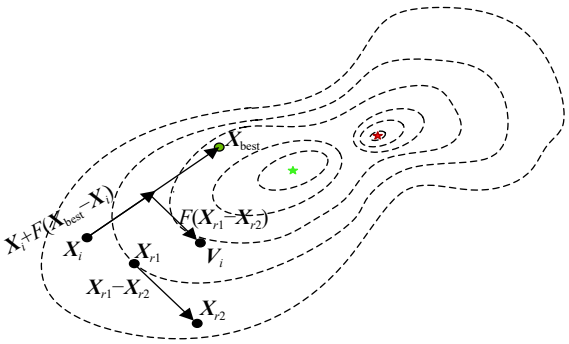


图 1　current-to-best/1 差分策略

2　PKLSHADE 算法

为了更好地平衡 DE 算法的全局开发和局部探索能力，本文在 LSHADE 算法的基础上提出一种基于搜索偏好知识的算法 PKLSHADE(preference-knowledge-based LSHADE)，采用当前评价次数与总评价次数比例权重实现全局开发到局部搜索的自适应平滑切换，并在理论上对算法的收敛性进行分析和证明。最后，基于 CEC 2017 标准测试集上的复杂混合多模函数对算法的效果进行仿真实验验证。

2.1　基于搜索偏好知识的变异策略

为使算法在进化初期加大全局开发能力，增大扰动以全面探测解空间，进化后期减少扰动，在已找到的最优解附近精细搜索加快算法收敛，设计新的差分变异策略如下：

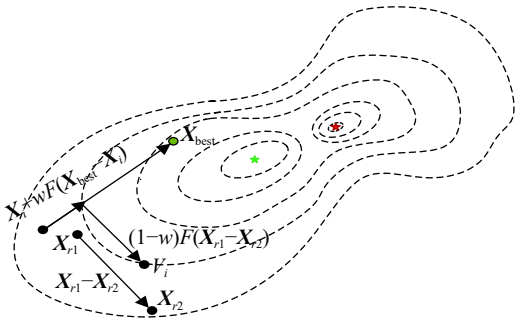
$$\boldsymbol{V}_i^t=\boldsymbol{X}_i^t+\omega F(\boldsymbol{X}_{\text{best}}^t-\boldsymbol{X}_i^t)+(1-\omega)F(\boldsymbol{X}_{r1}^t-\boldsymbol{X}_{r2}^t) \tag{11}$$

$$\omega=0.2+0.8\times\frac{\text{nfes}}{\text{max\_nfes}} \tag{12}$$

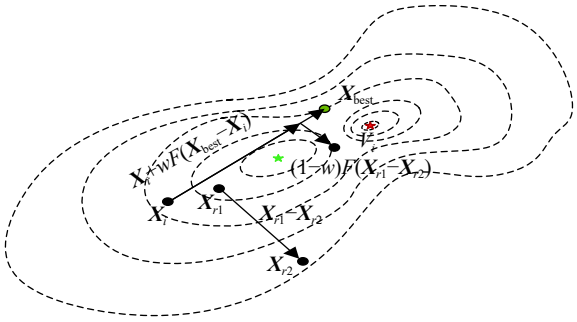
式中，nfes是当前评价次数；max_nfes为最大评价次数；ω 为前期权重因子。

在算法运行的前期权重因子ω较小，差分变异算子差分项$\boldsymbol{X}_{r1}^t-\boldsymbol{X}_{r2}^t$在变异中贡献较大，即扰动大，可以引导算法在较大范围内探测，如图 2a 所示。进化后期ω逐渐增大，$\boldsymbol{X}_{\text{best}}^t-\boldsymbol{X}_i^t$项的权重变大，意味着差分扰动变小，个体在当前最优解附近进行小范围的局部搜索，如图 2b 所示。

根据式 (11)，$\boldsymbol{V}_i^t$的结果由$\boldsymbol{X}_i^t$、$F(\boldsymbol{X}_{\text{best}}^t-\boldsymbol{X}_i^t)$和$F(\boldsymbol{X}_{r1}^t-\boldsymbol{X}_{r2}^t)$3 项构成，与 DE/current-to-best/1 策略的组成相同，而当ω较小时，本文的变异策略又与 DE/rand/1 策略接近。由此可见，本文的差分变异策略兼有 DE/rand/1 和 DE/current-to-best/1 的特点。



a. 进化前期



b. 进化后期

图 2　不同阶段变异策略对比

2.2　PKLSHADE 算法

PKLSHADE 算法基本流程和 LSHADE 一致，主要步骤如下：

1) 初始化，设置种群规模 N_p，问题维数 D，交叉率 C_r，最大评价次数 max_nfes，在取值范围内根据式 (3) 随机生成初始种群X^0；

2) 变异，随机选择$\boldsymbol{X}_i^t$、$\boldsymbol{X}_{r1}^t$和$\boldsymbol{X}_{r2}^t$，根据式 (11) 进行变异；

3) 交叉，采用式 (9) 进行交叉操作；

4) 选择，采用式 (10) 进行选择操作；

5) 终止判断，若nfes < max_nfes则返回步骤 2) 继续循环，否则结束循环，输出结果。

2.3　PKLSHADE 收敛性分析

DE 算法收敛性的方法有基于 Markov 链[12]、随机泛函分析理论[13] 和无群乘积理论[14] 等。下面以 Markov 链相关理论分析 PKLSHADE 算法的收敛性。

差分进化的迭代搜索过程包括差分变异、交叉、选择 3 个算子，这 3 个算子是相互独立的随机过程，可被认为是随机映射。

设$S=R^D$为被搜索的解空间，S^{N_P}为决策空间，f为适应度函数。

定义 1　差分变异算子：PKLSHADE 随机从S^{N_P}中选取$\boldsymbol{X}_i$、$\boldsymbol{X}_{r1}$和$\boldsymbol{X}_{r2}$，根据式 (11) 产生$\boldsymbol{V}_i$，是随

机映射，记作：

$$T_m : S^{N_p} \to S$$

其概率分布为：

$$\begin{aligned} &P\{T_m(\boldsymbol{X}) = \boldsymbol{V}_i\} = \\ &\sum P(T_m^1(\boldsymbol{X}) = \{\boldsymbol{X}_{r1}, \boldsymbol{X}_{r2}, \boldsymbol{X}_i, \boldsymbol{X}_{\text{best}}\}) \times \\ &P(T_m^2(\boldsymbol{X}_{r1}, \boldsymbol{X}_{r2}, \boldsymbol{X}_i, \boldsymbol{X}_{\text{best}}) = \boldsymbol{V}_i) = \\ &\sum P(T_m^1(\boldsymbol{X}) = \{\boldsymbol{X}_{r1}, \boldsymbol{X}_{r2}, \boldsymbol{X}_i, \boldsymbol{X}_{\text{best}}\}) \end{aligned} \tag{13}$$

式中，$\boldsymbol{X}_{r1}, \boldsymbol{X}_{r2}, \boldsymbol{X}_i, \boldsymbol{X}_{\text{best}} \in S^4$。

定义 2 交叉算子：交叉过程根据式 (9) 由$\boldsymbol{V}_i$和$\boldsymbol{X}_i$采用二项交叉产生新个体，属于随机映射，记作：

$$T_c : S^2 \to S$$

其概率分布为：

$$P\{T_c(\boldsymbol{X}_i, \boldsymbol{V}_i) = \boldsymbol{U}_i\} = \begin{cases} 0 & \boldsymbol{U}_i = \boldsymbol{X}_i \\ mC_D^k C_r^k (1 - C_r)^{D-k} & \text{其他} \\ C_r^N + 1/N_p & \boldsymbol{U}_i = \boldsymbol{V}_i \end{cases} \tag{14}$$

式中，$1 < k < D$为发生交叉的元素个数；m为交叉后产生的新个体数。

定义 3 选择算子：原目标向量$\boldsymbol{X}_i$和新实验向量$\boldsymbol{U}_i$进行贪婪选择，记作：

$$T_s : S^2 \to S$$

选择算子选择适应度更小的个体进入下一代种群，种群保留新个体$\boldsymbol{U}_i$的概率为：

$$P\{T_c(\boldsymbol{X}_i, \boldsymbol{U}_i) = \boldsymbol{U}_i\} = \begin{cases} 0 & f(\boldsymbol{U}_i) \leqslant f(\boldsymbol{X}_i) \\ 1 & \text{其他} \end{cases} \tag{15}$$

选择算子确保进化过程保留优秀解，该过程不可逆。由此，PKLSHADE 进化过程记作：

$$\begin{aligned} &\boldsymbol{X}(t+1) = \\ &\{\boldsymbol{X}_i(t+1) = T_s T_c T_m(\boldsymbol{X}(t)), i = 1, 2, \cdots, N_p\} \end{aligned} \tag{16}$$

定理 1 PKLSHADE 算法进化过程中，由于采用贪婪选择策略，种群个体适应度迭代序列单调非递增，即$F(\boldsymbol{X}(t+1) \leqslant F(\boldsymbol{X}(t))$，显然进化方向单调。

证明：

1) Markov 性：PKLSHADE 新一代个体的产生过程包括差分变异T_s、交叉T_c与选择T_m3 个环节，表示如下：

$$\boldsymbol{X}(t+1) = T(\boldsymbol{X}(t)) = T_s T_c T_m(\boldsymbol{X}(t)) \tag{17}$$

根据式 (13)～(15)，T_m、T_c与T_s都与迭代次数t无关，只与$\boldsymbol{X}(t)$相关，即第$t+1$代种群状态只和第t代种群相关，与第t代种群之前种群的状态无关，是 Markov 链。

2) 齐次性：因为 PKLSHADE 算法的差分变异、交叉与选择环节均与迭代次数t无关，故其k步转移概率$P_{ij}^k = P\{\boldsymbol{X}(t+k) = j | \boldsymbol{X}(t) = i\}$与$t$无关，即 PKLSHADE 算法种群迭代过程满足齐次性，证毕。

PKLSHADE 算法的转移概率为：

$$\begin{aligned} &P\{T(\boldsymbol{X}(t))_i = \boldsymbol{X}(t+1)\} = \\ &\sum\sum\sum P\{T_m(\boldsymbol{X}(t)) = \{\boldsymbol{X}_{r1}, \boldsymbol{X}_{r2}, \boldsymbol{X}_i, \boldsymbol{X}_{\text{best}}\}\} \times \\ &P\{T_c(\boldsymbol{X}_i(t), \boldsymbol{V}_i) = \boldsymbol{U}_i\} \times \\ &P\{T_s(\boldsymbol{X}_i(t), \boldsymbol{U}_i) = \boldsymbol{X}_i(t+1)\} = \\ &\prod_{i=1}^{N_p} P\{T(\boldsymbol{X}(t))_i = \boldsymbol{X}_i(t+1)\} \end{aligned} \tag{18}$$

$\forall \boldsymbol{X}(t) \in S^{N_p}, \exists \{\boldsymbol{X}_{r1}, \boldsymbol{X}_{r2}, \boldsymbol{X}_i, \boldsymbol{X}_{\text{best}}\} \in S^{N_p}, \boldsymbol{V}_i, \boldsymbol{U}_i \in S^{N_p}$，显然：

$$\begin{aligned} &P\{T_m(\boldsymbol{X}(t)) = \{\boldsymbol{X}_{r1}, \boldsymbol{X}_{r2}, \boldsymbol{X}_i, \boldsymbol{X}_{\text{best}}\} > 0 \\ &P\{T_c(\boldsymbol{X}_i(t), \boldsymbol{V}_i) = \boldsymbol{U}_i\} > 0 \\ &P\{T_s(\boldsymbol{X}_i(t), \boldsymbol{U}_i) = \boldsymbol{X}_i(t+1)\} > 0 \end{aligned}$$

因此，转移概率$P\{T(\boldsymbol{X}(t))_i = \boldsymbol{X}(t+1)\} > 0$，假设 PKLSHADE 算法 Markov 链种群序列的转移概率记作$P\{\boldsymbol{X}, \boldsymbol{Y}\} = P\{\boldsymbol{X}(t+1) = \boldsymbol{Y} | \boldsymbol{X}(t) = \boldsymbol{X}\}$，则有如下定理：

定理 2 PKLSHADE 种群序列$\{\boldsymbol{X}(t), t = 0, 1, 2, \cdots\}$以概率 1 收敛于解空间内全局最优解集$E^*$的子集$E_\delta^* = \{\boldsymbol{Y} = (y_1, y_2, \cdots, y_{N_p}), y_i \in E^*\}$，即：

$$\lim_{n \to \infty} P\{\boldsymbol{X}(t) \in E_\delta^* | \boldsymbol{X}(0) = \boldsymbol{X}_0\} = 1$$

证明：假设$\boldsymbol{X}^*$是$f(x)$的唯一全局最优解，由式 (13) 和式 (14) 可得：

1) 如果$\boldsymbol{X}, \boldsymbol{Y} \in E_\delta^*$，则$P\{\boldsymbol{X}, \boldsymbol{Y}\} > 0$，$P\{\boldsymbol{Y}, \boldsymbol{X}\} > 0$，即两状态互通$\boldsymbol{X} \leftrightarrow \boldsymbol{Y}$；

2) 如果$\boldsymbol{X} \in E_\delta^*$，而$\boldsymbol{Y} \notin E_\delta^*$，则$P\{\boldsymbol{X}, \boldsymbol{Y}\} = 0$，即$\boldsymbol{X} \nleftrightarrow \boldsymbol{Y}$。

因此，E_δ^*为正常返的非周期不可约闭集。对于任意初始解，存在极限分布：

$$\lim_{n \to \infty} P\{\boldsymbol{X}(t) = \boldsymbol{Y} | \boldsymbol{X}(0) = \boldsymbol{X}_0\} = \begin{cases} \pi(\boldsymbol{Y}) & \boldsymbol{Y} \in E_\delta^* \\ 0 & \boldsymbol{Y} \in E_\delta^* \end{cases}$$

即$\boldsymbol{X}(t)$必然会进入E_δ^*内，且满足某一极限概率分布$\pi(\boldsymbol{Y})$，所以$\lim_{n \to \infty} P\{\boldsymbol{X}(t) \in E_\delta^* | \boldsymbol{X}(0) = \boldsymbol{X}_0\} = 1$，证毕。

3 实验结果与分析

为了评估 PKLSHADE 算法的性能，实验选用 CEC2017[15] 标准测试集的复杂混合多模函数f_{11}～f_{30}进行验证。其中f_{11}～f_{20}为混合函数 (hybrid function)，f_{21}～f_{30}为复合函数 (composition function)。

函数维数 D 取 10，所有算法在所选测试函数上独立运行 51 次。

3.1　PKLSHADE 算法参数分析

本文算法基于 LSHADE 算法，有 4 个参数分别是N_p、H、μF与μC_r，关于参数的说明详见文献 [4]。选择合理的参数初值，对于算法的性能具有重要影响，本文采用 DOE(design of experiment)[16] 方法进行分析。每个参数设置 3 个水平值，如表 1 所示。

表 1　PKLSHADE 参数水平设置

参数	水平		
	1	2	3
N_p	50	100	200
H	5	10	15
μF	0.2	0.5	0.8
μC_r	0.2	0.5	0.8

根据正交表$L_9(3^4)$对每组参数组合独立运行 30 次，算法所得平均性能作为响应值，结果如表 2 所示。

表 2　PKLSHADE 参数水平及响应

参数				平均响应
N_p	H	μF	μC_r	
50	5	0.2	0.2	1.77×10^4
50	10	0.5	0.5	1.29×10^4
50	15	0.8	0.8	1.51×10^4
100	5	0.5	0.8	1.94×10^4
100	10	0.8	0.2	2.90×10^4
100	15	0.2	0.5	1.67×10^4
200	5	0.8	0.5	1.60×10^4
200	10	0.2	0.8	1.89×10^4
200	15	0.5	0.2	2.90×10^4

通过表 2 可以计算得到各参数在每一水平值下的平均响应值并得到参数主效应图，然后根据极差确定 4 个参数对算法影响等级排序。表 3 的等级排序结果说明参数μC_r对算法影响最大，然后依次是N_p和μF，H对算法性能的影响相对最小。

表 3　PKLSHADE 参数平均响应及等级

水平	参数			
	N_p	H	μF	μC_r
1	**15 245**	**17 677**	**17 785**	25 252
2	21 712	20 306	20 448	**15 220**
3	21 320	20 294	20 045	17 805
极差	6 466	2 629	2 663	10 031
等级	2	4	3	1

根据图 3 可见，PKLSHADE 算法的最佳参数组合是$N_p = 50$，$H = 5$，$\mu F = 0.2$，$\mu C_r = 0.5$。

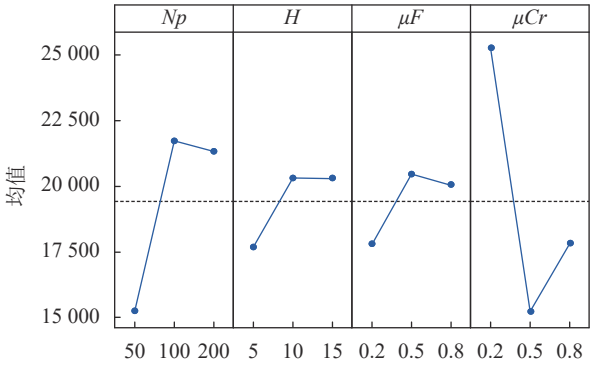


图 3　参数主效应图

3.2　PKLSHADE 与其他算法对比

对比算法选用目前具有代表性的种群线性下降及参数自适应的 LSHADE 算法、自适应多精英引导的 AMECoDEs 算法、LSHADE 算法的改进算法 jSO 和 EBLSHADE，以上算法采用 current-to-best/1、current-to-pbest-w/1、current-to-ord-pbest/1 和current-to-nbest/1 和 current-to-pbest/1 等优秀的变异策略。

表 4 是本文算法与对比算法 51 次独立运行所得平均结果，包括最优解的平均值 (Mean) 和标准差 (std)。

从表 4 可以看出所有算法在f_{26}函数上均找到相同的最优解。LSHADE 与 jSO 算法在f_{11}函数上、jSO 与 EBLSHADE 算法在f_{14}函数上、LSHADE 与 EBLSHADE 算法在f_{20}函数上、PKLSHADE 与 AMECoDEs 算法在f_{28}函数上都找到相同的最优解。在其余 15 个函数中，PKLSHADE 在 7 个函数上 (f_{21}～f_{25}、f_{27}、f_{30}) 都优于其他对比算法；而 LSHADE、AMECoDEs、jSO 和 EBLSHADE 算法分别有 1、0、4 和 3 个函数优于其他算法。

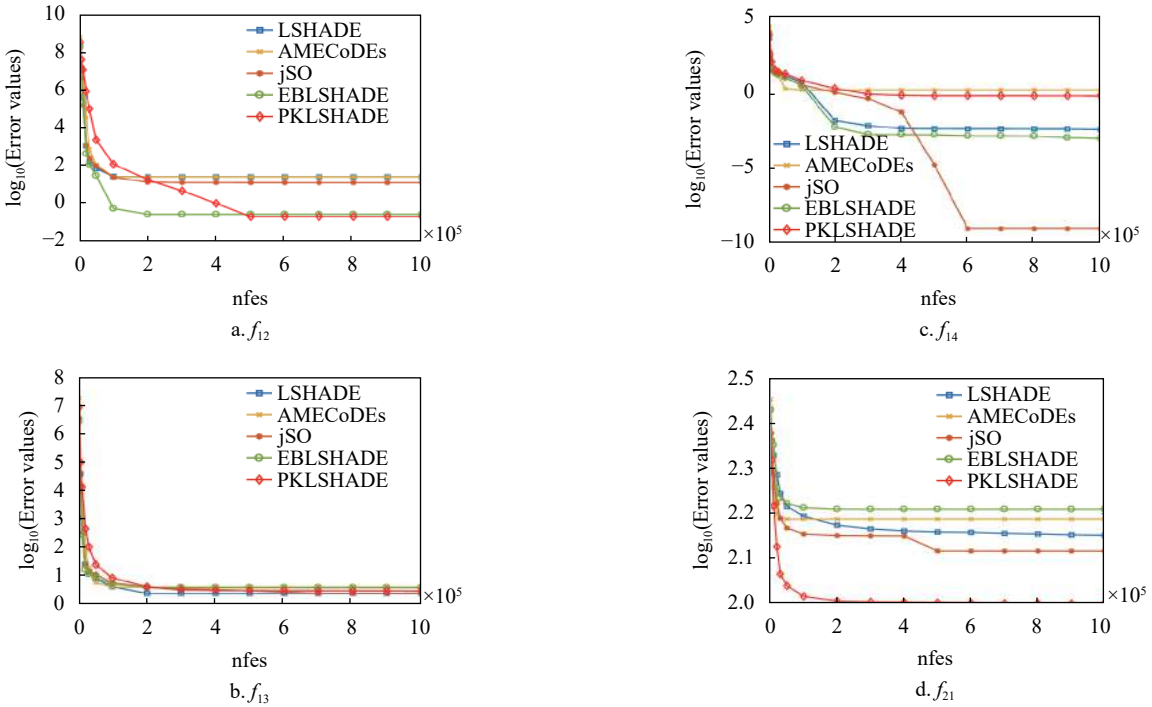
如果将所有算法找到最优解的函数个数 (包括共同取得最优解的情况) 进行统计，在全部 20 个函数中，PKLSHADE、LSHADE、AMECoDEs、jSO 和 EBLSHADE 算法分别有 9、4、2、7 和 6 个函数找到相对最优解，PKLSHADE 依然有一定的优势。

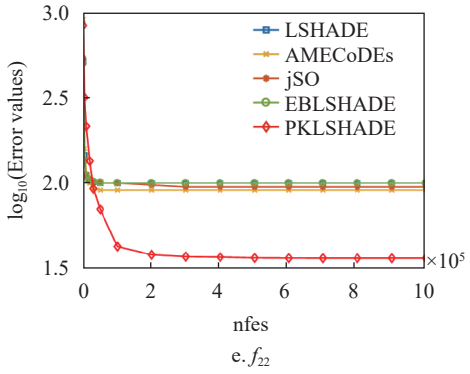
分析表 4 还可以发现，PKLSHADE 算法在复合函数f_{21}～f_{30}上优势明显，在混合函数f_{11}～f_{20}上虽然表现不是最好，但与其他对比算法没有显著性差异，部分结果依然优于除得到最优解算法以外的其它算法，如f_{13}、f_{18}和f_{20}，仅在f_{15}函数上与 AMECoDEs 算法的最优解接近，劣于其他算法。

表 4 PKLSHADE 与其他对比算法的最优解平均值与标准差

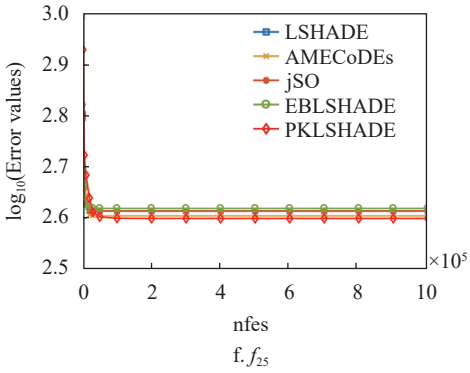
函数	PKLSHADEMean$_{std}$	LSHADEMean$_{std}$	AMECoDEsMean$_{std}$	jSOMean$_{std}$	EBLSHADEMean$_{std}$
f_{11}	8.004 5×10^{-1}$_{4.73\times10^{-1}}$	**0.000 0×10^{0}**$_{0.00\times10^{0}}$	6.964 7×10^{-1}$_{6.72\times10^{-1}}$	**0.000 0×10^{0}**$_{0.00\times10^{0}}$	9.5507E-03$_{3.02\times10^{-2}}$
f_{12}	3.059 3×10^{-1}$_{2.56\times10^{-1}}$	2.409 3×10^{1}$_{5.05\times10^{1}}$	1.367 1×10^{0}$_{3.45\times10^{0}}$	**1.457 0×10^{-1}**$_{1.33\times10^{-1}}$	2.081 4×10^{-1}$_{2.40\times10^{-1}}$
f_{13}	2.318 6×10^{0}$_{1.27\times10^{0}}$	3.386 0×10^{0}$_{2.34\times10^{0}}$	3.664 1×10^{0}$_{2.55\times10^{0}}$	**1.451 1×10^{0}**$_{2.22\times10^{0}}$	3.874 3×10^{0}$_{2.04\times10^{0}}$
f_{14}	8.650 8×10^{-1}$_{4.71\times10^{-1}}$	2.085 1×10^{-2}$_{6.47\times10^{-2}}$	2.070 0×10^{0}$_{3.21\times10^{0}}$	**0.000 0×10^{0}**$_{0.00\times10^{0}}$	**0.000 0×10^{0}**$_{0.00\times10^{0}}$
f_{15}	1.862 2×10^{-1}$_{9.46\times10^{-2}}$	5.833 1×10^{-2}$_{1.51\times10^{-1}}$	1.028 5×10^{-1}$_{1.59\times10^{-1}}$	4.989 2×10^{-2}$_{1.45\times10^{-1}}$	**3.149 7×10^{-3}**$_{5.53\times10^{-3}}$
f_{16}	2.068 4×10^{-1}$_{1.05\times10^{-1}}$	**3.864 5×10^{-2}**$_{3.87\times10^{-2}}$	4.749 7×10^{-1}$_{2.72\times10^{-1}}$	2.419 8×10^{-1}$_{2.08\times10^{-1}}$	1.858 7×10^{-1}$_{1.78\times10^{-1}}$
f_{17}	3.928 6×10^{-1}$_{2.84\times10^{-1}}$	1.056 5×10^{-2}$_{6.99\times10^{-3}}$	1.474 9×10^{0}$_{8.34\times10^{-1}}$	1.381 1×10^{-2}$_{9.04\times10^{-3}}$	**1.029 7×10^{-2}**$_{8.54\times10^{-3}}$
f_{18}	7.138 7×10^{-2}$_{7.60\times10^{-2}}$	7.862 8×10^{-2}$_{1.52\times10^{-1}}$	1.006 4×10^{-1}$_{3.14\times10^{-1}}$	**6.165 5×10^{-4}**$_{1.62\times10^{-3}}$	1.280 9×10^{-1}$_{2.06\times10^{-1}}$
f_{19}	2.625 9×10^{-2}$_{6.39\times10^{-3}}$	8.688 5×10^{-5}$_{2.75\times10^{-4}}$	7.704 0×10^{-2}$_{6.99\times10^{-2}}$	**0.000 0×10^{0}**$_{0.00\times10^{0}}$	6.013 0×10^{-6}$_{1.90\times10^{-5}}$
f_{20}	3.121 7×10^{-2}$_{9.87\times10^{-2}}$	**0.000 0×10^{0}**$_{0.00\times10^{0}}$	6.243 5×10^{-2}$_{1.32\times10^{-1}}$	6.243 5×10^{-2}$_{1.25\times10^{-1}}$	**0.000 0×10^{0}**$_{0.00\times10^{0}}$
f_{21}	**1.000 0×10^{2}**$_{3.44\times10^{-6}}$	1.416 5×10^{2}$_{5.29\times10^{1}}$	1.433 9×10^{2}$_{5.61\times10^{1}}$	1.619 4×10^{2}$_{5.06\times10^{1}}$	1.417 1×10^{2}$_{5.39\times10^{1}}$
f_{22}	**1.563 9×10^{1}**$_{3.21\times10^{1}}$	1.000 0×10^{2}$_{0.00\times10^{0}}$	9.030 0×10^{1}$_{3.17\times10^{1}}$	1.000 0×10^{2}$_{0.00\times10^{0}}$	1.000 0×10^{2}$_{0.00\times10^{0}}$
f_{23}	**2.756 0×10^{2}**$_{9.69\times10^{1}}$	3.009 6×10^{2}$_{1.57\times10^{0}}$	3.060 8×10^{2}$_{3.86\times10^{0}}$	3.008 0×10^{2}$_{1.23\times10^{0}}$	3.019 6×10^{2}$_{1.75\times10^{0}}$
f_{24}	**9.268 3×10^{1}**$_{2.31\times10^{1}}$	3.042 5×10^{2}$_{7.24\times10^{1}}$	2.889 3×10^{2}$_{9.96\times10^{1}}$	2.580 6×10^{2}$_{1.04\times10^{2}}$	3.290 2×10^{2}$_{3.62\times10^{0}}$
f_{25}	**3.978 8×10^{2}**$_{1.49\times10^{-1}}$	4.070 0×10^{2}$_{1.92\times10^{1}}$	4.073 0×10^{2}$_{1.97\times10^{1}}$	4.025 1×10^{2}$_{1.36\times10^{1}}$	4.163 6×10^{2}$_{2.38\times10^{1}}$
f_{26}	**3.000 0×10^{2}**$_{0.00\times10^{0}}$	**3.000 0×10^{2}**$_{0.00\times10^{0}}$	**3.000 0×10^{2}**$_{0.00\times10^{0}}$	**3.000 0×10^{2}**$_{0.00\times10^{0}}$	**3.000 0×10^{2}**$_{0.00\times10^{0}}$
f_{27}	**3.887 5×10^{2}**$_{3.06\times10^{-1}}$	3.895 2×10^{2}$_{0.00\times10^{0}}$	3.891 7×10^{2}$_{7.16\times10^{-1}}$	3.894 2×10^{2}$_{2.05\times10^{-1}}$	3.895 2×10^{2}$_{0.00\times10^{0}}$
f_{28}	**3.000 0×10^{2}**$_{0.00\times10^{0}}$	3.623 6×10^{2}$_{1.31\times10^{2}}$	**3.000 0×10^{2}**$_{0.00\times10^{0}}$	3.595 6×10^{2}$_{1.19\times10^{2}}$	4.163 0×10^{2}$_{1.50\times10^{2}}$
f_{29}	2.378 0×10^{2}$_{2.27\times10^{0}}$	2.309 6×10^{2}$_{2.20\times10^{0}}$	2.378 4×10^{2}$_{7.15\times10^{0}}$	2.309 7×10^{2}$_{1.51\times10^{0}}$	**2.296 0×10^{2}**$_{2.05\times10^{0}}$
f_{30}	**3.945 0×10^{2}**$_{6.46\times10^{-3}}$	8.211 3×10^{4}$_{2.58\times10^{5}}$	3.951 3×10^{2}$_{0.00\times10^{0}}$	3.954 3×10^{2}$_{4.01\times10^{-3}}$	3.993 2×10^{2}$_{1.52\times10^{1}}$

图 4 给出了统一评价次数下 PKLSHADE 与其他算法在收敛性方面部分函数的对比，其中 nfes 是评价次数，Error values 是所有算法获得的最优解与标准测试函数已知最优解之间的误差值，该值越小，说明算法的搜索性能越好。可以看出，PKLSHADE 算法具有较好的收敛性，在函数 f_{21}、f_{22}上解的精度和收敛速度均表现突出。对于函数 f_{12}和 f_{14}，虽然由于前期重视全局开发使得收敛效果弱于其它算法，但在进化后期更重视局部搜索机制的作用下，收敛精度能够迅速追平甚至超过其他算法。在函数 f_{13}与 f_{25}上 PKLSHADE 与其他算法接近。





e. f_{22}

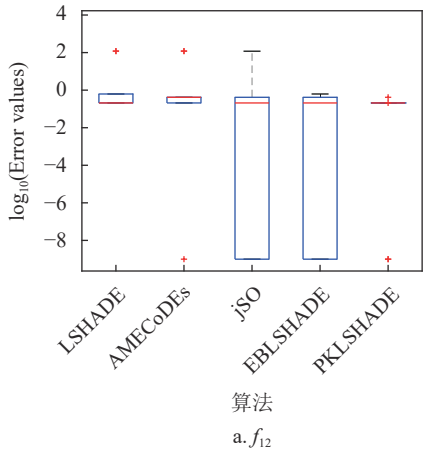


f. f_{25}

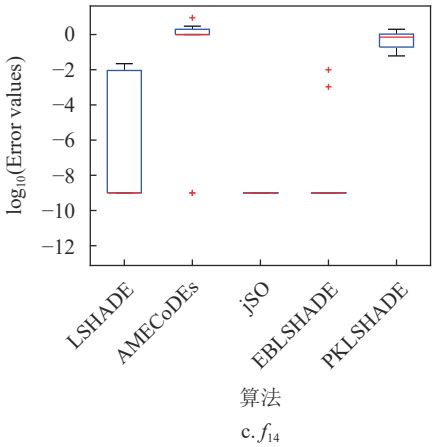
图 4　收敛曲线对比

图 5 是 PKLSHADE 算法与其他对比算法运行分布特性的盒图对比，包含反映寻优结果的最大值、最小值、中位值等分布信息。可以看出除 f_{22}函数外，PKLSHADE 算法在函数f_{12}、f_{21}和f_{25}上均具有最窄的分布，有很好的稳定性；在函数f_{13}和f_{14}上也有较好的分布特性，接近最优对比算法的结果。

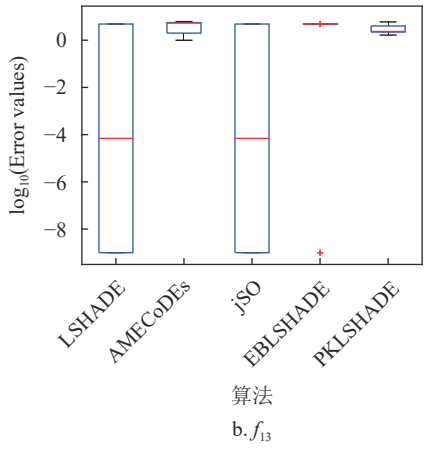
为进一步验证 PKLSHADE 算法的优势，对所有算法的优化结果进行 Wilcoxon 非参数假设检验分析，显著性水平a取 0.05 和 0.1，对应置信度区间 95% 和 90%。结果如表 5 所示。其中，Z 是非参数检验中的统计量，p-value 是置信度值。从表 5 可以看出 PKLSHADE 算法相对 LSHADE、AMECoDEs 和 EBLSHADE 存在显著性差异，明显优于上述 3 个算法；与 jSO 算法相比虽然没有显著性差异，即 PKLSHADE 与 jSO 算法的搜索性能相当，但由对应 $R-$的值 91 大于 $R+$的值 62 可知在所有 20 个测试函数上，PKLSHADE 的总体寻优效果依然优于 jSO 算法。



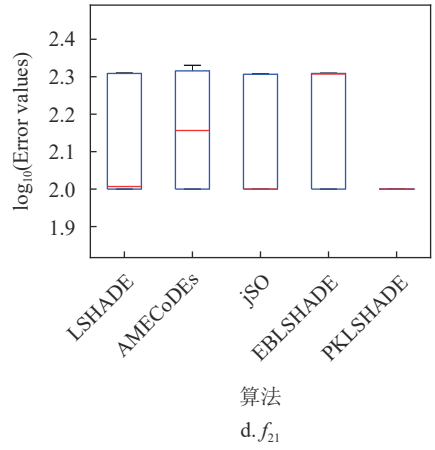
a. f_{12}



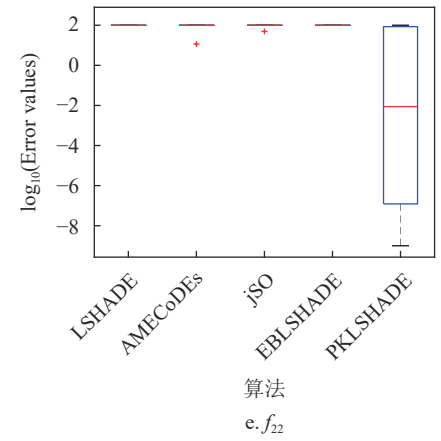
c. f_{14}



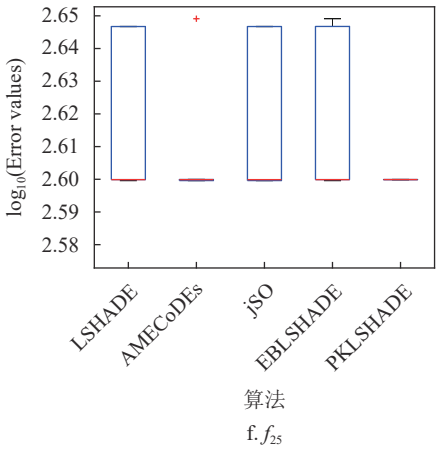
b. f_{13}



d. f_{21}



e. f_{22}



f. f_{25}

图 5 部分函数分布盒图对比

表 5 Wilcoxon 非参数检验

算法	*R*+	*R*−	*Z*	*p*-value	*a*=0.05	*a*=0.1
LSHADE	46	144	−1.972	0.049	**Yes**	**Yes**
AMECoDEs	9	111	−2.897	0.004	**Yes**	**Yes**
jSO	62	91	−0.686	0.492	**No**	**No**
EBLSHADE	54	136	−1.650	0.099	**Yes**	**Yes**

4 结 束 语

本文提出了一种基于搜索知识偏好的差分进化算法，在进化的初期重视差分扰动以便在更大范围内探测解空间，搜索更多潜在的最优解所在区域，进化后期则在前期所得最优解的引导下，更重视局部小范围搜索，以获得精度更高的解并加快算法收敛，提出的算法能够在全局开发和局部搜索之间平滑过渡，能够很好地平衡二者的关系。

同时，本文还基于 Markov 理论对提出算法的收敛性进行理论分析和证明，并通过 DOE 实验设计方法分析算法最优参数组合。在 CEC 2017 复杂混合多模函数上的实验表明，本文算法在求解精度、收敛性及稳定性等方面都有一定提升。

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] DAS S, MULLICK S S, SUGANTHAN P N. Recent advances in differential evolution - an updated survey[J]. Swarm and Evolutionary Computation, 2016, 27: 1-30.
[3] TANABE R, FUKUNAGA A. Success-history based parameter adaptation for differential evolution[C]//IEEE Congress on Evolutionary Computation (CEC 2013). Cancun: IEEE, 2013: 71-78.
[4] TANABE R, FUKUNAGA A. Improving the search performance of SHADE using linear population size reduction[C]//IEEE Congress on Evolutionary Computation (CEC 2014). Beijing: IEEE, 2014: 1658-1665.
[5] BREST J, MAUCEC M S, BOSKOVIC B. Single objective real-parameter optimization: Algorithm jSO[C]//IEEE Congress on Evolutionary Computation (CEC 2017). Spain: IEEE, 2017: 1311-1318.
[6] MOHAMED A W, HADI A A, JAMBI K M. Novel mutation strategy for enhancing SHADE and LSHADE algorithms for global numerical optimization[J]. Swarm and Evolutionary Computation, 2019, 50(11): 1-14.
[7] CUI L, LI G, ZHU Z, et al. Adaptive multiple- elites-guided composite differential evolution algorithm with a shift mechanism[J]. Information Sciences, 2018, 422(1): 122-143.
[8] HE Xiao-yu, ZHOU Yu-ren. Enhancing the performance of differential evolution with covariance matrix self-adaptation [J]. Applied Soft Computing, 2018, 64(3): 227-243.
[9] WANG Shi-hao, LI Yu-zhen, YANG Hong-yu. Self-adaptive mutation differential evolution algorithm based on particle swarm optimization[J]. Applied Soft Computing, 2019, 81(8): 1-22.
[10] ALI I M, ESSAM D, KASMARIK K. A novel design of differential evolution for solving discrete traveling salesman problems[J]. Swarm and Evolutionary Computation, 2020, 52(2): 1-17.
[11] LIU Z Z, WANG Y, YANG S X, et al. Differential evolution with a two-stage optimization mechanism for numerical optimization[C]//IEEE Congress on Evolutionary Computation (CEC 2016). Vancouver: IEEE, 2016: 3170-3177.
[12] HU Zhong-bo, XIONG Sheng-wu, SU Qing-hua. Finite Markov chain analysis of classical differential evolution algorithm[J]. Journal of Computational and Applied Mathematics, 2014, 268(11): 121-134.
[13] 贺毅朝, 王熙照, 刘坤起, 等. 差分演化的收敛性分析与算法改进[J]. 软件学报, 2010, 21(5): 875-885.
HE Yi-chao, WANG Xi-zhao, LIU Kun-qi, et al. Convergent analysis and algorithmic improvement of differential evolution[J]. Journal of Software, 2010, 21(5): 875-885.
[14] HU Z, XIONG S, SU Q, et al. Sufficient conditions for global convergence of differential evolution algorithm[J]. Journal of Applied Mathematics, 2013(1): 1044-1065.
[15] AWAD N H, ALI M Z, SUGANTHAN P N, et al. Problem definitions and evaluation criteria for the CEC 2017 special session and competition on single objective real-parameter numerical optimization[R]. Vancouver: IEEE, 2016.
[16] ANGELA M D, DANIEL T V. Design and analysis of experiments[M]. Cham: Springer, 2017.

编 辑 漆 蓉