文章编号:1006-2467(2024)04-0555-10 DOI: 10.16183/j.cnki.jsjtu.2022.470

基于群体划分的冠状病毒群体免疫优化算法

李博群, 孙志锋
(浙江大学 电气工程学院,杭州 310027)

摘 要:针对冠状病毒群体免疫优化(CHIO)算法收敛速度慢、求解精度低的问题,提出一种基于群体划分的冠状病毒群体免疫优化(SD-CHIO)算法.基于适应度均匀原则将初始群体划分为两部分,即全局寻优个体与局部寻优个体.对于全局寻优个体,在其位置更新中加入差分变异与漫反射变异策略,分别用来增强全局寻优个体之间的交流与群体多样性,从而提高算法的全局搜索能力.对于局部寻优个体,在其位置更新中引入一种自适应快速收敛策略:基于增量法进行精英预测,并加入一种自适应收敛系数使局部寻优个体能快速收敛至精英解,以提升算法的局部搜索能力.数值实验表明:SD-CHIO 能够有效提高原算法的收敛速度与精度,并表现出明显优于其他元启发式算法的全局与局部搜索能力以及一定的工程价值.

关键词:冠状病毒群体免疫优化算法;群体划分;自适应快速收敛;差分变异;漫反射变异
中图分类号:TP301 **文献标志码**:A

A Coronavirus Herd Immunity Optimizer Based on Swarm Division

LI Boqun, *SUN Zhifeng*
(College of Electrical Engineering, Zhejiang University, Hangzhou 310027, China)

Abstract: Aimed at the drawbacks of coronavirus herd immunity optimizer (CHIO), i. e., slow convergence speed and low optimization accuracy, a CHIO based on swarm division (SD-CHIO) is proposed. Based on the principle of uniform fitness, the initial swarm is divided into two parts, i. e., exploration individuals and exploitation individuals. For exploration individuals, differential mutation and diffuse reflection mutation are adopted in position update in order to enhance the communication among exploration individuals and swarm diversity respectively, so as to improve the exploration capability of the algorithm. For exploitation individuals, an adaptive fast convergence strategy is proposed in position update: elite prediction is conducted based on the incremental method, and an adaptive convergence coefficient is employed to ensure that exploitation individuals can quickly converge to the elite solution, which improves the exploitation capability of the algorithm. The numerical experiments demonstrate that SD-CHIO significantly improves the convergence speed and accuracy of the conventional algorithm, exhibiting better exploration and exploitation capabilities than other meta-heuristic algorithms do as well as certain value in engineering.

Keywords: coronavirus herd immunity optimizer (CHIO); swarm division; adaptive fast convergence; differential mutation; diffuse reflection mutation

---

收稿日期:2022-11-25 **修回日期**:2023-01-30 **录用日期**:2023-03-03
基金项目:国家自然科学基金重点项目(91647209),浙江省自然科学基金项目(LGG18F030005),宁波市重大专项(2022Z176)资助项目
作者简介:李博群(1997-),硕士生,从事智能控制算法研究.
通信作者:孙志锋,副教授;E-mail: eeszf@zju.edu.cn.

元启发式算法(Meta-Heuristic Algorithm, MA)是一种模拟动植物群体行为或自然物理机制,通过迭代实现方案寻优的高鲁棒性智能算法,适用于多变量、非线性、多极值的复杂优化问题求解[1-2].其中,粒子群优化[3](Particle Swarm Optimization, PSO)算法、差分进化(Differential Evolution, DE)算法[4]与人工蜂群(Artificial Bee Colony, ABC)算法[5]为最常用的传统元启发式算法.近年来涌现出很多新型元启发式算法,如蜻蜓算法[6]、正余弦算法[7]、鲸鱼优化算法[8](Whale Optimization Algorithm, WOA)、樽海鞘群算法[9](Salp Swarm Algorithm, SSA)等. Al-Betar 等[10]受冠状病毒传播机制的启发,于 2021 年提出一种冠状病毒群体免疫优化(Coronavirus Herd Immunity Optimizer, CHIO)算法.CHIO 具有参数少、结构简单、易于实现等优点[11-12],在车辆路径规划[11]、电源配置[13]、医学诊断[14]、配电网重构[15]等研究领域得到良好应用,具有重要的工程意义.但相对于很多元启发式算法,该算法的收敛速度与精度仍较低[10].

为了提高 CHIO 的搜索性能,部分学者给出相应的改进方案. Dalbah 等[11]提出一种用于有能力约束车辆路径问题(Capacitated Vehicle Routing Problem, CVRP)的增强型 CHIO,引入一种修复策略,即在 CHIO 初始化及进化过程中不断调整 CVRP 中的客户分布,有效维持优化过程中 CVRP 解的可行性;但该算法仅适用于 CVRP 问题,适用范围小,且在收敛速度与精度方面与很多元启发式算法相比无明显优势. Alweshah 等[14]在 CHIO 中引入一种贪婪交叉算子,即随机选择两个个体进行位置信息交叉变异来产生更优秀的子代,大大提高原算法的全局搜索性能;然而该算法在处理具有连续搜索空间的优化问题时,需要将待优化变量重新编码为整型数值,操作繁琐. Naderipour 等[15]提出一种改进 CHIO(Improved Coronavirus Herd Immunity Optimizer, ICHIO)算法,采用一种非线性递减惯性权重调整位置更新方程,能有效提高原算法的收敛速度与求解精度.

基于上述研究,提出一种基于群体划分的 CHIO(CHIO Based on Swarm Division, SD-CHIO)算法.基于适应度均匀原则,将初始群体划分为全局寻优个体与局部寻优个体.在原算法的"病毒休眠"阶段,利用全局寻优个体与局部寻优个体的作用分别提高算法的全局搜索能力与局部搜索能力.最后,基于数值实验验证了 SD-CHIO 的有效性.

1 冠状病毒群体免疫优化算法

医学界根据患者感染冠状病毒的时间段与症状轻重,将个体划分为 3 类:感染个体、易感个体与免疫个体. CHIO 模仿冠状病毒在这 3 类个体间的传染机制,构建位置更新公式[10]如下:

$$x_{i,j}^{(t+1)} = \begin{cases} x_{i,j}^{(t)} \pm R_{\mathrm{D}}\left(x_{i,j}^{(t)} - x_{S_1\{M_1\},j}^{(t)}\right), & 0 \leqslant r < \dfrac{B_{\mathrm{r}}}{3} \\ x_{i,j}^{(t)} \pm R_{\mathrm{D}}\left(x_{i,j}^{(t)} - x_{S_2\{M_2\},j}^{(t)}\right), & \dfrac{B_{\mathrm{r}}}{3} \leqslant r < \dfrac{2B_{\mathrm{r}}}{3} \\ x_{i,j}^{(t)} \pm R_{\mathrm{D}}\left(x_{i,j}^{(t)} - x_{S_3\{M_3\},j}^{(t)}\right), & \dfrac{2B_{\mathrm{r}}}{3} \leqslant r < B_{\mathrm{r}} \\ x_{i,j}^{(t)}, & B_{\mathrm{r}} \leqslant r \leqslant 1 \end{cases} \tag{1}$$

式中:$x_{i,j}^{(t)}$($i=1, 2, \cdots, N$;$j=1, 2, \cdots, D$;$t=1, 2, \cdots, T$)为第 t 次迭代中,第 i 个个体在第 j 维的位置,N、D、T 分别为群体规模、搜索空间维度与总迭代次数;感染个体、易感个体与免疫个体的编号分别组成集合 S_1、S_2 与 S_3;M_1、M_2 与 M_3 分别为随机选择的感染个体、随机选择的易感个体与最优免疫个体的编号在 S_1、S_2 与 S_3 中的序号;随机数 r、$R_{\mathrm{D}} \sim U(0, 1)$;基本繁殖率 $B_{\mathrm{r}} \in (0, 1)$. B_{r} 为病毒传播与休眠阶段的分界线,当 $r < B_{\mathrm{r}}$ 时,为"病毒传播"阶段,此时病毒在感染个体、易感个体与免疫个体间扩散,个体位置会受 $x_{S_1\{M_1\},j}^{(t)}$、$x_{S_2\{M_2\},j}^{(t)}$ 或 $x_{S_3\{M_3\},j}^{(t)}$ 的扰动而发生变化;当 $r \geqslant B_{\mathrm{r}}$ 时,为"病毒休眠"阶段,此时病毒停止扩散,个体位置将不受其他个体的影响而保持不变.每个个体的位置都将基于式(1)迭代 T 次,之后取 N 个个体中最优个体的位置作为优化结果.

CHIO 中,3 种个体可以相互转变.当易感个体的位置变差且低于平均水平时,将会转变为感染个体.同样,当感染个体的位置变好且高于平均水平时,将会转变为免疫个体.可以看出,较优个体一般都集中在易感个体与免疫个体中.

2 基于群体划分的冠状病毒群体免疫优化算法

在"病毒休眠"阶段,CHIO 采用保持个体位置不变的策略.该策略在前期很难让个体跳出局部最优,后期不利于个体的局部收敛,会弱化 CHIO 的全局与局部搜索能力.为了克服这一缺陷,提出一种 SD-CHIO 算法,包含如下改进.

2.1 群体划分策略

引入群体划分策略,即将群体划分为全局寻优

个体与局部寻优个体. 在“病毒休眠”阶段，全局寻优个体与局部寻优个体采用不同的位置更新方式分别提高 CHIO 的全局与局部搜索能力.

基于适应度均匀原则进行群体划分. 首先，基于初始群体的适应度对每个个体进行排序：

$$O_0 = \underset{i=1,2,\cdots,N}{\text{sort}}(\text{fobj}(x_{i,j}^{(0)})) \tag{2}$$

式中：fobj(·)为适应度计算函数，适应度越小表示该个体的位置越优；$\underset{i=1,2,\cdots,N}{\text{sort}}(\cdot)$表示对 N 个个体的适应度进行升序排序，其输出 O_0 为排序后个体的编号集合.

然后，对全局寻优个体与局部寻优个体进行选择. 定义全局寻优个体的编号集合 I_1 与局部寻优个体的编号集合 I_2 如下：

$$I_1 = O_0\{2, 4, 6, \cdots, 2\lfloor N/2 \rfloor\} \tag{3}$$

$$I_2 = O_0\{1, 3, 5, \cdots, 2\lceil N/2 \rceil - 1\} \tag{4}$$

式中：$\lceil\cdot\rceil$与$\lfloor\cdot\rfloor$分别表示向上取整与向下取整. 为了均衡全局与局部寻优个体的适应度，将 O_0 中序号为偶数与奇数的元素分别赋予 I_1 与 I_2，以避免两种个体适应度差距较大对全局与局部搜索能力中一种能力的单方面不利影响. 将最优个体编号 $O_0\{1\}$ 赋予 I_2，以降低算法在前期落入其他较差局部最优解的可能性.

2.2　全局寻优个体位置更新

为了增强个体之间的交流以提高算法的全局搜索能力，受差分进化算法的启发，引入“差分变异”策略[4]. 定义全局寻优个体 $I_1\{i_G\}$ ($i_G = 1,2,\cdots,\lfloor N/2 \rfloor$) 的位置 $x_{I_1\{i_G\},j}^{(t)}$ 对应的差分变异量为

$$\sigma_{I_1\{i_G\},j}^{(t)} = x_{I_1\{\zeta_1\},j}^{(t)} + R_D(x_{I_1\{\zeta_2\},j}^{(t)} - x_{I_1\{\zeta_3\},j}^{(t)}) \tag{5}$$

式中：ζ_1、ζ_2 与 ζ_3 为 1～$\lfloor N/2 \rfloor$中随机选取的 3 个互不相同且不为 i_G 的整数. 将其他全局寻优个体 $I_1\{\zeta_1\}$ 的位置 $x_{I_1\{\zeta_1\},j}^{(t)}$ 赋值给差分变异量 $\sigma_{I_1\{i_G\},j}^{(t)}$，并采用其他全局寻优个体 $I_1\{\zeta_2\}$ 与 $I_1\{\zeta_3\}$ 的位置差，即差分向量$[x_{I_1\{\zeta_2\},j}^{(t)} - x_{I_1\{\zeta_3\},j}^{(t)}]$对 $\sigma_{I_1\{i_G\},j}^{(t)}$ 进行扰动，以达到增强个体之间交流的目的.

同时，当算法落入局部最优时，如果没有针对性的策略让其及时跳出局部最优，也会对算法的全局搜索能力造成负面影响. 为了提高 CHIO 跳出局部最优的能力，模仿自然界中的漫反射现象，提出一种“漫反射变异”，其示意图如图 1 所示.

图 1 中，为了便于分析，建立平面直角坐标系 d_wOh_w. 图中：ε、x、Q 均为省略上、下标的简写. 从点 $(x_{I_1\{i_G\},j}^{(t)}, x_{I_1\{i_G\},j}^{(t)})$ 向水面上的 O 点，即入射点$(Q_j^{(t)}, 0)$射入光线，入射光线与 h_w 轴呈 45°. 水面波动会出

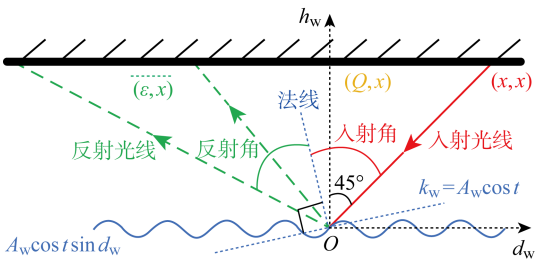

图 1　漫反射变异的示意图

Fig. 1　Diagram of diffuse reflection mutation

现漫反射现象，即反射光线也存在波动，因此反射光线最终射向点$(\varepsilon_{I_1\{i_G\},j}^{(t)}, x_{I_1\{i_G\},j}^{(t)})$，$\varepsilon_{I_1\{i_G\}}^{(t)}$ 即 $x_{I_1\{i_G\}}^{(t)}$ 对应的漫反射变异量. 构造函数 $A_w\cos t\sin d_w$ 来描述水面的形状，其中 A_w 为水面波动的最大幅值. 由此可得 O 点处水面的斜率为

$$k_w = \left.\frac{\mathrm{d}A_w\cos t\sin d_w}{\mathrm{d}d_w}\right|_{d_w=0} = A_w\cos t$$

t 为迭代次数. 根据入射角等于反射角等几何关系可进一步得到：

$$\frac{-\dfrac{1}{k_w} - \dfrac{Q_j^{(t)} - x_{I_1\{i_G\},j}^{(t)}}{Q_j^{(t)} - \varepsilon_{I_1\{i_G\},j}^{(t)}}}{1 - \dfrac{Q_j^{(t)} - x_{I_1\{i_G\},j}^{(t)}}{k_w(Q_j^{(t)} - \varepsilon_{I_1\{i_G\},j}^{(t)})}} = \frac{1+\dfrac{1}{k_w}}{1-\dfrac{1}{k_w}} \tag{6}$$

为了使漫反射变异不存在过大偏差，定义入射点横坐标为所有全局寻优个体位置的平均值，即

$$Q_j^{(t)} = \frac{\sum_{i_g=1}^{\lfloor N/2 \rfloor} x_{I_1\{i_g\},j}^{(t)}}{\lfloor N/2 \rfloor}$$

基于式(6)，可求得的表达式为

$$\varepsilon_{I_1\{i_G\},j}^{(t)} = \frac{1}{\lfloor N/2 \rfloor}\sum_{i_g=1}^{\lfloor N/2 \rfloor} x_{I_1\{i_g\},j}^{(t)} + \frac{1+2A_w\cos t - A_w^2\cos^2 t}{1-2A_w\cos t - A_w^2\cos^2 t} \times \left(\frac{1}{\lfloor N/2 \rfloor}\sum_{i_g=1}^{\lfloor N/2 \rfloor} x_{I_1\{i_g\},j}^{(t)} - x_{I_1\{i_G\},j}^{(t)}\right) \tag{7}$$

采用漫反射变异策略，个体有更多机会探索未搜索过的领域，有利于提高群体的多样性，增强算法的全局搜索能力.

基于上述两种变异策略，构造全局寻优个体的位置更新公式如下：

$$x_{I_1\{i_G\},j}^{(t+1)}\big|_{j=1,2,\cdots,D} = \begin{cases} x_{I_1\{i_G\},j}^{(t)}, & 0 \leqslant p_j < 0.5 \\ \sigma_{I_1\{i_G\},j}^{(t)}, & 0.5 \leqslant p_j < 0.75 \\ \varepsilon_{I_1\{i_G\},j}^{(t)}, & 0.75 \leqslant p_j \leqslant 1 \end{cases} \tag{8}$$

式中：随机数 $p_j \sim U(0, 1)$，在每个维度 j 都会更新一次. $p_j < 0.5$ 时，保持位置不变；$p_j \geqslant 0.5$ 时，加入

差分变异与漫反射变异策略来增强个体之间的交流并提高群体的多样性.

2.3 局部寻优个体位置更新

为了提高算法收敛到最优解的速度与精度以加强算法的局部搜索能力,引入一种“自适应快速收敛”策略.基于该策略,构造局部寻优个体 $I_2\{i_L\}$ 的位置更新公式如下:

$$x_{I_2\{i_L\},j}^{(t+1)}=(1\pm R_D\psi_L^{(t)})x_{I_2\{i_L\},j}^{(t)}+[1\pm(R_D\pm B_s)\psi_L^{(t)}](\eta_j^{(t)}-x_{I_2\{i_L\},j}^{(t)}) \quad (9)$$

式中:随机数 $B_s\sim U(0,1)$;$\psi_L^{(t)}$ 为自适应收敛系数;个体的位置向精英解 $\eta_j^{(t)}$ 转移.$\eta_j^{(t)}$ 越优,个体就能更快收敛到较优位置,那么算法的收敛速度将会得到提升.

CHIO 中,最优个体一般存在于易感个体或免疫个体中.为了保证精英解 $\eta_j^{(t)}$ 足够优秀以加快算法的收敛速度,基于易感与免疫个体中最优个体的位置 $\lambda_j^{(t)}$ 以及增量法构造精英解预测公式,即 $\eta_j^{(t)}$ 的表达式为

$$\eta_j^{(t)}=\lambda_j^{(t)}+R_D\gamma_\Delta\left(1-\frac{t}{T}\right)(\lambda_j^{(t)}-\lambda_j^{(t-1)}) \quad (10)$$

式中:引入易感与免疫个体中最优个体的位置增量 $(\lambda_j^{(t)}-\lambda_j^{(t-1)})$ 对 $\lambda_j^{(t)}$ 进行补偿,来提前预测潜在最优解,即精英解 $\eta_j^{(t)}$;γ_Δ 为预测步长,用来对位置增量的幅度进行调整.系数 $\left(1-\frac{t}{T}\right)$ 用来逐渐压缩位置增量的幅度,避免在后期产生过大预测误差从而对收敛精度产生不利影响.

为了提高算法的收敛精度,定义式(9)中的自适应收敛系数 $\psi_L^{(t)}$ 为一种单调递减的凸函数.基于反余切函数构造 $\psi_L^{(t)}$ 的表达式如下:

$$\psi_L^{(t)}=(\beta_1-\beta_2)\sqrt{\frac{4}{\pi}\operatorname{arccot}\frac{t}{T}-1}+\beta_2 \quad (11)$$

式中:β_1、β_2 分别为 $\psi_L^{(t)}$ 的初值与终值.为了验证 $\psi_L^{(t)}$ 的凹凸性,求其二阶导数

$$\frac{\mathrm{d}^2\psi_L^{(t)}}{\mathrm{d}t^2}=$$

$$-\frac{4(\beta_1-\beta_2)}{\pi^2T^2}\frac{1+\frac{\pi t}{T}-\frac{4t}{T}\operatorname{arccot}\frac{t}{T}}{\left[1+\left(\frac{t}{T}\right)^2\right]^2\left(\frac{4}{\pi}\operatorname{arccot}\frac{t}{T}-1\right)^{\frac{3}{2}}}\leqslant$$

$$-\frac{4(\beta_1-\beta_2)}{\pi^2T^2}\frac{1+\frac{\pi t}{T}-\frac{4t}{T}\frac{\pi}{2}\left(1-\frac{t}{2T}\right)}{\left[1+\left(\frac{t}{T}\right)^2\right]^2\left(\frac{4}{\pi}\operatorname{arccot}\frac{t}{T}-1\right)^{\frac{3}{2}}}=$$

$$-\frac{4(\beta_1-\beta_2)}{\pi^2T^2}\times$$

$$\frac{\pi\left(\frac{t}{T}-\frac{1}{2}\right)^2+1-\frac{\pi}{4}}{\left[1+\left(\frac{t}{T}\right)^2\right]^2\left(\frac{4}{\pi}\operatorname{arccot}\frac{t}{T}-1\right)^{\frac{3}{2}}}<0 \quad (12)$$

式中:$\psi_L^{(t)}$ 的二阶导数小于 0,因此 $\psi_L^{(t)}$ 为凸函数.

根据式(11)可画出 $\psi_L^{(t)}$ 的图像,如图 2 所示.可以看出,随着迭代次数 t 的增加,$\psi_L^{(t)}$ 斜率的绝对值越来越大,表明 $\psi_L^{(t)}$ 在加速减小,有助于快速缩小局部寻优个体在 $\eta_j^{(t)}$ 周围的搜索区域,从而有效提高收敛精度,增强算法的局部搜索能力.

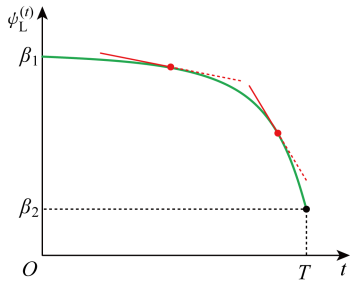

图 2 自适应收敛系数 $\psi_L^{(t)}$ 的图像

Fig. 2 Graph of adaptive convergence coefficient $\psi_L^{(t)}$

2.4 SD-CHIO 的实现

对标准 CHIO 进行上述改进后,给出 SD-CHIO 的伪代码,其中行序号粗体的伪代码为改进步骤,具体如下.

算法一:SD-CHIO 伪代码

1. 参数初始化:N, D, T, B_r, A_w, γ_Δ, β_1, β_2.
2. 群体位置与状态初始化.
3. 初始群体适应度计算.
4. 利用式(2)初始群体适应度排序.
5. 利用式(3)、(4)计算 I_1, I_2.
6. $t=1$.
7. WHILE ($t\leqslant T$)
8. 根据群体适应度选取 $\lambda_j^{(t)}$.
9. FOR[1]($i=1$ TO N)
10. IF[1]($0\leqslant r<B_r/3$)
11. $x_{i,j}^{(t+1)}=x_{i,j}^{(t)}\pm R_D(x_{i,j}^{(t)}-x_{S_1\{M_1\},j}^{(t)})$.
12. ELSE_IF ($B_r/3\leqslant r<2B_r/3$)
13. $x_{i,j}^{(t+1)}=x_{i,j}^{(t)}\pm R_D(x_{i,j}^{(t)}-x_{S_2\{M_2\},j}^{(t)})$.
14. ELSE_IF ($2B_r/3\leqslant r<B_r$)
15. $x_{i,j}^{(t+1)}=x_{i,j}^{(t)}\pm R_D(x_{i,j}^{(t)}-x_{S_3\{M_3\},j}^{(t)})$.
16. ELSE
17. IF[2]($i\in I_1$) //全局寻优个体位置更新
18. 利用式(5)计算 $\sigma_{i,j}^{(t)}$.
19. 利用式(7)计算 $\varepsilon_{i,j}^{(t)}$.

20. FOR[2] ($j=1$ TO D)
21. $p_j=$ rand.
22. IF[3] ($0\leqslant p_j<0.5$)
23. $x_{i,j}^{(t+1)}=x_{i,j}^{(t)}$.
24. ELSE_IF ($0.5\leqslant p_j<0.75$)
25. $x_{i,j}^{(t+1)}=\sigma_{i,j}^{(t)}$. //差分变异
26. ELSE
27. $x_{i,j}^{(t+1)}=\varepsilon_{i,j}^{(t)}$. //漫反射变异
28. END IF[3]
29. END FOR[2]
30. ELSE //局部寻优个体位置更新
31. 利用式(11)计算 $\psi_{\mathrm{L}}^{(t)}$.
32. 利用式(10)计算 $\eta_j^{(t)}$.
33. 利用式(9)计算 $x_{i,j}^{(t+1)}$. //基于精英预测的自适应收敛
34. END IF[2]
35. END IF[1]
36. 群体状态更新与适应度计算.
37. END FOR[1]
38. $t\leftarrow t+1$.
39. END WHILE
40. RETURN 历史最优个体的位置.

SD-CHIO 的复杂度主要由群体初始化、位置更新与适应度计算 3 个过程决定. 其中，群体初始化的复杂度为 $O(N)$，位置更新的复杂度为 $O(DNT)$，适应度计算的复杂度为 $O(NT)$，因此 SD-CHIO 的总复杂度为 $O(N(1+DT+T))$. 因为 SD-CHIO 中的改进部分并不增加原算法 CHIO 的适应度计算次数，所以其复杂度与 CHIO 以及 PSO、差分进化、WOA、SSA、ICHIO 算法基本持平，且低于 ABC 算法.

3　数值实验及分析

为了验证 SD-CHIO 的有效性，采用基准测试函数以及工程问题对其进行性能测试. 数值实验在 AMD Ryzen 5 3500U with Radeon Vega Mobile Gfx 2.10 GHz 8 GB 内存 MATLAB2018b 条件下实现. 在 SD-CHIO 中，设定 $B_{\mathrm{r}}=0.01$，因为该情况下原算法 CHIO 的综合寻优性能最佳[10]；其他参数设置为 $A_{\mathrm{w}}=0.1$，$\gamma_{\Delta}=0.2$，$\beta_1=2$，$\beta_2=0.01$.

3.1　函数测试

选取 12 个典型测试函数，如表 1 所示. 其中 $F_1\sim F_4$ 为单峰函数[8, 10]，可用来检验算法的局部搜索能力；$F_5\sim F_8$ 为多峰函数[10, 16]，可用来测试算法的全局搜索能力. $F_9\sim F_{12}$ 选自 CEC'2008[17] 与 CEC'2022[18] 函数优化竞赛，为带偏移量 $\boldsymbol{o}$ 与旋转量 $\boldsymbol{M}$ 的复杂函数，可测试算法在复杂优化问题中的性能. 表中：random[0,1)表示[0,1)的随机数.

为分别测试 SD-CHIO 中各改进策略的性能，基于各改进策略创建 3 种算法：CHIO-1、CHIO-2 与 CHIO-3. CHIO-1 中只有局部寻优个体作用；CHIO-2 中只有全局寻优个体作用且不采用漫反射变异，即在式(8)中 $0.75\leqslant p_j\leqslant 1$ 的位置更新部分也采用差分变异；CHIO-3 中只有全局寻优个体作用，且差分变异与漫反射变异均被采用. 基于函数 F_9 对 CHIO、CHIO-1、CHIO-2、CHIO-3 与 SD-CHIO 进行测试. 为了公平对比，设置每种算法的初始解相同，且将每种算法的迭代参数均设为 $D=30$，$N=30$，$T=500$. 为了降低实验的偶然性，令每种算法独立运行 30 次，之后得出 5 种算法的平均收敛曲线，如图 3 所示.

由图 3 可知，当只有局部寻优个体作用时，CHIO-1 收敛曲线在前期($t=1\sim40$)的下降速率远大于 CHIO，因此 CHIO-1 的收敛速度在前期较 CHIO 有极大提升；但在后期($t=460\sim500$)，由于没有全局寻优个体作用，算法极易落入局部最优，所以 CHIO-1 的收敛精度较 CHIO 只有很小的提高. 当只有全局寻优个体作用且只采用差分变异策略时，由于差分变异策略可以加强算法中个体之间的交流，从而提高算法的全局搜索能力，所以 CHIO-2 后期收敛精度较 CHIO 与 CHIO-1 有极大提升. 当只有全局寻优个体作用且差分变异与漫反射变异策略均被采用时，由于漫反射变异策略可以提高算法跳出局部最优的能力，所以 CHIO-3 后期收敛精度较 CHIO-2 进一步提升. 当全局寻优个体与局部寻优个体均作用时，相当于在 CHIO-3 跳出局部最优后加入局部寻优个体进行局部优化，使 SD-CHIO 的收敛速度与收敛精度较 CHIO-3 得到进一步提高，局部搜索能力增强. 经验证，上述改进策略可以大大增强原算法的全局与局部搜索能力.

为了比较 SD-CHIO 与其他元启发式算法的搜索性能，基于 $F_1\sim F_{12}$ 对 SD-CHIO 以及传统元启发式算法(DE、PSO、ABC)、新型元启发式算法(WOA、SSA)、CHIO 及其改进算法 ICHIO 进行函数测试对比实验. 实验中，以综合性能最优为导向设置每种元启发式算法的非共有参数. 为了保证公平性，对于同一函数，设置每种元启发式算法的初始解

表 1 基准测试函数

Tab. 1 Function of benchmark test

函数名称	函数表达式	搜索范围
Sphere	$F_1(x)=\sum_{j=1}^{D}x_j^2$	[−100, 100]
Schwefel P2.21	$F_2(x)=\max_{j=1,2,\cdots,D}\lvert x_j\rvert$	[−100, 100]
Schwefel P2.22	$F_3(x)=\sum_{j=1}^{D}\lvert x_j\rvert+\prod_{j=1}^{D}\lvert x_j\rvert$	[−10, 10]
Noise	$F_4(x)=\sum_{j=1}^{D}jx_j^4+\mathrm{random}[0,1)$	[−100, 100]
Rastrigin	$F_5(x)=\sum_{j=1}^{D}[x_j^2-10\cos(2\pi x_j)+10]$	[−5.12, 5.12]
Griewank	$F_6(x)=\frac{1}{4\,000}\sum_{j=1}^{D}x_j^2-\prod_{j=1}^{D}\cos\frac{x_j}{\sqrt{j}}+1$	[−600, 600]
Salomon	$F_7(x)=-\cos\left(2\pi\sqrt{\sum_{j=1}^{D}x_j^2}\right)+0.1\sqrt{\sum_{j=1}^{D}x_j^2}+1$	[−100, 100]
Ackley	$F_8(x)=-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{j=1}^{D}x_j^2}\right)-\exp\left[\frac{1}{D}\sum_{j=1}^{D}\cos(2\pi x_j)\right]+20+\mathrm{e}$	[−32, 32]
Shifted Griewank	$F_9(x)=\frac{1}{4\,000}\sum_{j=1}^{D}(x_j-o_j)^2-\prod_{j=1}^{D}\cos\frac{x_j-o_j}{\sqrt{j}}+1$	[−600, 600]
Shifted Ackley	$F_{10}(x)=-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{j=1}^{D}(x_j-o_j)^2}\right)-\exp\left[\frac{1}{D}\sum_{j=1}^{D}\cos 2\pi(x_j-o_j)\right]+20+\mathrm{e}$	[−32, 32]
Shifted+Rotated Rastrigin	$F_{11}(x)=\sum_{j=1}^{D}[z_j^2-10\cos(2\pi z_j)+10],\quad \boldsymbol{z}=\boldsymbol{M}[0.051\,2(\boldsymbol{x}-\boldsymbol{o})]$	[−100, 100]
Shifted+Rotated Levy	$F_{12}(x)=\sin^2(\pi w_1)+\sum_{j=1}^{D-1}(w_j-1)^2[1+10\sin^2(\pi w_j-1)]+(w_D-1)^2[1+\sin^2(2\pi w_D)]$, $w_j=1+\frac{z_j-1}{4},\quad \boldsymbol{z}=\boldsymbol{M}[0.051\,2(\boldsymbol{x}-\boldsymbol{o})]$	[−100, 100]

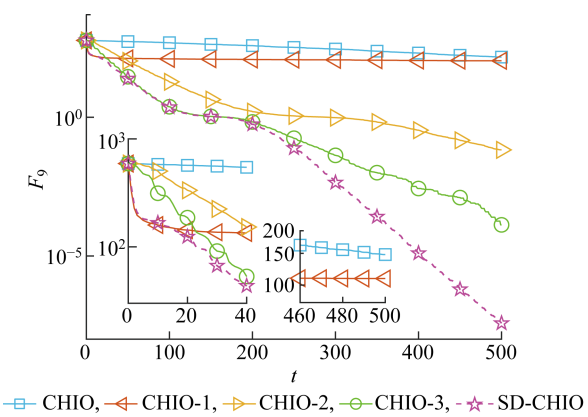

图 3 基于不同改进策略的 CHIO 的平均收敛曲线

Fig. 3 Average convergence curves of CHIOs based on different strategies for improvement

相同，且将每个元启发式算法的迭代参数均设置为 $N=30$；$T=500$；$D=30$（$F_1\sim F_{10}$），10（$F_{11}\sim F_{12}$）. 为了降低偶然性，对于同一函数，让每种元启发式算法独立运行 30 次，并记录最终求得的最优值的平均值（μ）与标准差（s）作为算法的评估指标，如表 2 所示. 其中，最优数据加粗显示.

由表 2 可见，在单峰函数 $F_1\sim F_4$ 的测试中，SD-CHIO 不仅在 F_1 中收敛到 μ 的理论最优值 0，而且在 $F_2\sim F_4$ 中的收敛精度也优于其他元启发式算法，说明 SD-CHIO 大大提高了原算法 CHIO 的局部搜索能力，且其局部搜索能力优于改进算法 ICHIO 与其他元启发式算法. 在多峰函数 $F_5\sim F_8$ 的测试中，SD-CHIO 在 $F_5\sim F_7$ 中均取得 μ 的理论

表 2　不同元启发式算法的函数测试结果

Tab. 2　Function test results of different MAs

函数	算法	μ/A	s/A	函数	算法	μ/A	s/A	函数	算法	μ/A	s/A
F_1	PSO	7.90×10^{-3}	1.15×10^{-2}	F_5	PSO	5.93×10^{1}	1.73×10^{1}	F_9	PSO	2.84×10^{-2}	2.04×10^{-2}
	DE	5.93×10^{-4}	3.42×10^{-4}		DE	1.54×10^{2}	9.87×10^{0}		DE	3.37×10^{-2}	6.18×10^{-2}
	ABC	2.43×10^{1}	1.52×10^{1}		ABC	2.38×10^{2}	1.25×10^{1}		ABC	1.20×10^{0}	9.48×10^{-2}
	WOA	1.48×10^{-74}	6.76×10^{-74}		WOA	3.79×10^{-15}	2.08×10^{-14}		WOA	1.12×10^{1}	3.58×10^{0}
	SSA	2.45×10^{-7}	7.25×10^{-7}		SSA	5.16×10^{1}	1.68×10^{1}		SSA	1.71×10^{-2}	1.49×10^{-2}
	CHIO	1.21×10^{4}	3.08×10^{3}		CHIO	1.71×10^{2}	1.75×10^{1}		CHIO	1.48×10^{2}	3.48×10^{1}
	ICHIO	5.69×10^{-43}	1.62×10^{-43}		ICHIO	2.07×10^{2}	7.68×10^{1}		ICHIO	1.28×10^{2}	2.35×10^{1}
	SD-CHIO	**0**	**0**		SD-CHIO	**0**	**0**		SD-CHIO	$\mathbf{3.76\times10^{-8}}$	$\mathbf{2.53\times10^{-8}}$
F_2	PSO	7.58×10^{0}	1.79×10^{0}	F_6	PSO	2.49×10^{-2}	1.78×10^{-2}	F_{10}	PSO	2.19×10^{0}	3.63×10^{0}
	DE	9.34×10^{0}	1.81×10^{0}		DE	3.44×10^{-2}	6.30×10^{-2}		DE	5.98×10^{-3}	1.46×10^{-3}
	ABC	6.48×10^{1}	4.07×10^{0}		ABC	1.22×10^{0}	9.54×10^{-2}		ABC	4.15×10^{0}	3.25×10^{-1}
	WOA	4.97×10^{1}	2.52×10^{1}		WOA	8.37×10^{-3}	4.59×10^{-2}		WOA	1.56×10^{1}	1.27×10^{0}
	SSA	1.25×10^{1}	3.51×10^{0}		SSA	1.64×10^{-2}	1.12×10^{-2}		SSA	2.71×10^{0}	8.22×10^{-1}
	CHIO	7.37×10^{1}	3.41×10^{0}		CHIO	1.13×10^{2}	3.12×10^{1}		CHIO	1.73×10^{1}	7.18×10^{-1}
	ICHIO	4.25×10^{-22}	9.60×10^{-23}		ICHIO	**0**	**0**		ICHIO	1.63×10^{1}	3.05×10^{-1}
	SD-CHIO	$\mathbf{9.51\times10^{-163}}$	**0**		SD-CHIO	**0**	**0**		SD-CHIO	$\mathbf{1.88\times10^{-5}}$	$\mathbf{5.49\times10^{-6}}$
F_3	PSO	1.39×10^{0}	3.45×10^{0}	F_7	PSO	9.68×10^{-1}	1.99×10^{-1}	F_{11}	PSO	1.62×10^{1}	2.78×10^{0}
	DE	4.89×10^{-3}	1.53×10^{-3}		DE	8.74×10^{-1}	1.03×10^{-1}		DE	3.72×10^{1}	$\mathbf{2.25\times10^{0}}$
	ABC	2.96×10^{-1}	7.37×10^{-2}		ABC	3.17×10^{0}	4.13×10^{-1}		ABC	3.66×10^{1}	5.15×10^{0}
	WOA	3.84×10^{-51}	1.51×10^{-50}		WOA	1.43×10^{-1}	6.26×10^{-2}		WOA	3.14×10^{1}	2.17×10^{1}
	SSA	2.08×10^{0}	1.49×10^{0}		SSA	1.90×10^{0}	4.60×10^{-1}		SSA	1.62×10^{1}	7.29×10^{0}
	CHIO	8.19×10^{4}	4.46×10^{5}		CHIO	1.84×10^{1}	1.30×10^{0}		CHIO	7.26×10^{1}	9.96×10^{0}
	ICHIO	2.87×10^{-22}	4.10×10^{-23}		ICHIO	4.46×10^{-1}	5.71×10^{-2}		ICHIO	4.93×10^{1}	8.15×10^{0}
	SD-CHIO	$\mathbf{4.26\times10^{-164}}$	**0**		SD-CHIO	**0**	**0**		SD-CHIO	$\mathbf{1.02\times10^{1}}$	2.81×10^{0}
F_4	PSO	4.73×10^{-2}	1.34×10^{-2}	F_8	PSO	5.26×10^{-1}	7.42×10^{-1}	F_{12}	PSO	2.99×10^{-3}	1.63×10^{-2}
	DE	1.60×10^{-1}	3.11×10^{-2}		DE	7.47×10^{-3}	2.10×10^{-3}		DE	1.39×10^{-12}	7.12×10^{-12}
	ABC	3.09×10^{-1}	8.71×10^{-2}		ABC	4.17×10^{0}	3.96×10^{-1}		ABC	3.07×10^{-5}	5.28×10^{-5}
	WOA	3.14×10^{-3}	2.68×10^{-3}		WOA	4.09×10^{-15}	3.00×10^{-15}		WOA	1.81×10^{0}	1.60×10^{0}
	SSA	2.38×10^{-1}	9.11×10^{-2}		SSA	2.53×10^{0}	8.36×10^{-1}		SSA	3.77×10^{-1}	4.89×10^{-1}
	CHIO	1.36×10^{2}	1.33×10^{2}		CHIO	1.67×10^{1}	8.05×10^{-1}		CHIO	2.77×10^{0}	6.26×10^{-1}
	ICHIO	1.65×10^{-3}	6.31×10^{-4}		ICHIO	1.13×10^{-14}	2.94×10^{-15}		ICHIO	1.36×10^{0}	3.81×10^{-1}
	SD-CHIO	$\mathbf{1.51\times10^{-4}}$	$\mathbf{8.50\times10^{-5}}$		SD-CHIO	$\mathbf{8.88\times10^{-16}}$	**0**		SD-CHIO	$\mathbf{4.90\times10^{-13}}$	$\mathbf{7.87\times10^{-13}}$

最优值 0，且在 F_8 中的收敛精度也优于其他元启发式算法，表明 SD-CHIO 的全局搜索能力较原算法 CHIO 同样得到很大提升，且其全局搜索能力同样优于改进算法 ICHIO 与其他元启发式算法. 在复杂函数 $F_9\sim F_{12}$ 的测试中，SD-CHIO 的收敛精度均优于其他元启发式算法，表明其在求解复杂优化问题时，相对于其他元启发式算法具有一定优势. 除 F_{11} 外，SD-CHIO 在所有函数测试中均能取得最小 s 值，且 7 次取得理论最优值 0，说明 SD-CHIO 相比于其他元启发式算法具有较强鲁棒性.

为了更加直观地比较各元启发式算法的收敛性能，给出 8 种元启发式算法的平均收敛曲线，如图 4 所示. 由图 4(a)～4(d)可知，对于单峰函数 $F_1\sim F_4$，自适应快速收敛策略大大提高了原算法前期收敛速度与后期收敛精度，同时也让 SD-CHIO 拥有 8 类元启发式算法中最高的收敛速度与精度. 由图 4(e)～4(h)可知，对于多峰函数 $F_5\sim F_8$，CHIO 明显陷入局部最优，导致后期收敛精度很低；而差分变异与漫反射变异策略让 SD-CHIO 更容易跳出局部最优，有效提升原算法后期的收敛精度，同时也让 SD-CHIO获得 8 类元启发式算法中最优的全局搜索能力. 图 4(i)～4(l)中，对于复杂函数 $F_9\sim F_{12}$，

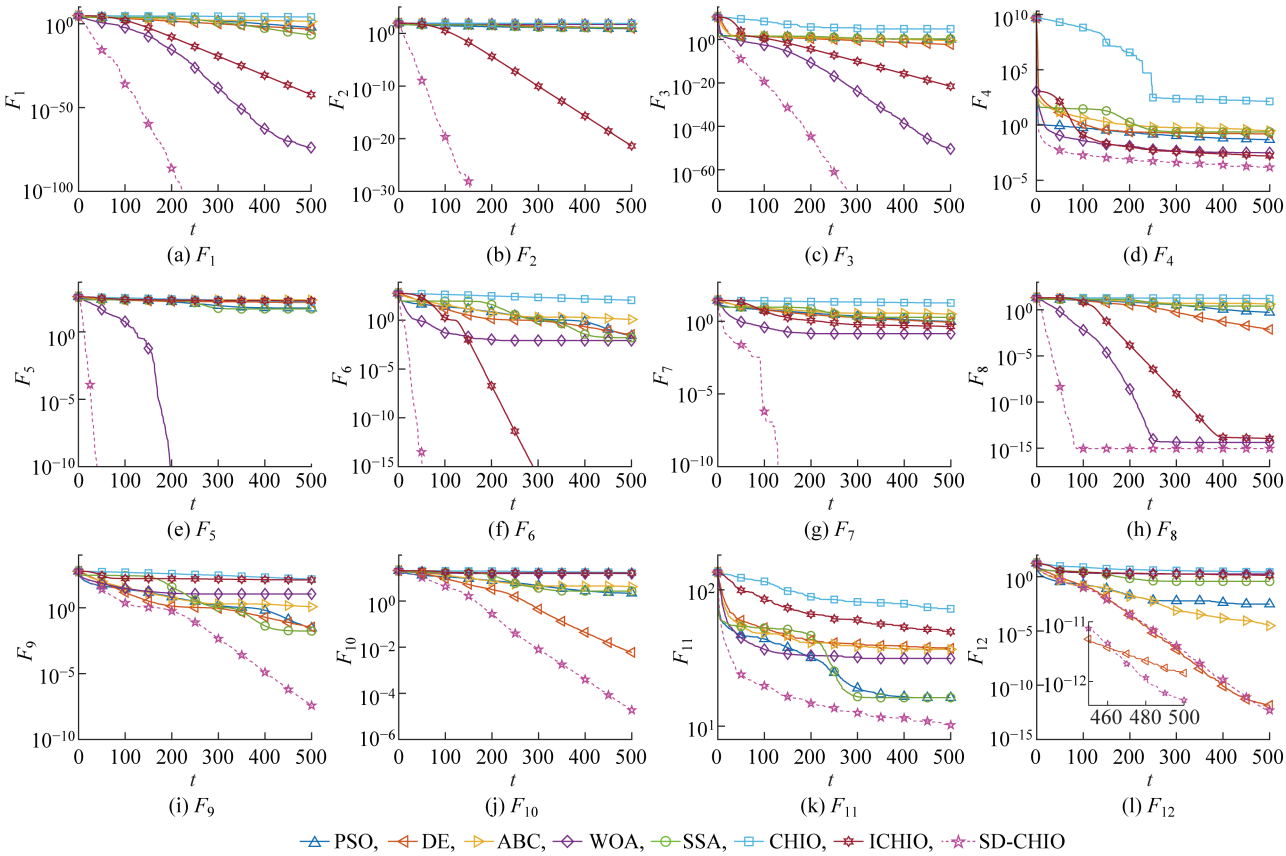

图 4 函数测试中不同元启发式算法的平均收敛曲线

Fig. 4 Average convergence curves of different MAs in function test

SD-CHIO与原算法以及其他元启发式算法相比更易跳出局部最优,因此其后期收敛精度最高.

3.2 工程应用

采用光伏电池参数辨识这一工程问题来测试SD-CHIO的实用性能.光伏电池的等效电路如图5所示[19].图中:I_{ph}、I_D、I_{sh}分别为光生电流、二极管正向电流与二极管泄露电流;R_{sh}、R_s分别为并联电阻和串联电阻;V_L与I_L分别为光伏电池的输出电压与输出电流.

根据图5可以推导出I_L的数学表达式为

$$I_L = I_{ph} - I_D - I_{sh} = I_{ph} - I_{sd}\left\{\exp\left[\frac{q_c(V_L + I_L R_s)}{A_c k_c T_c}\right] - 1\right\} - \frac{V_L + I_L R_s}{R_{sh}} \tag{13}$$

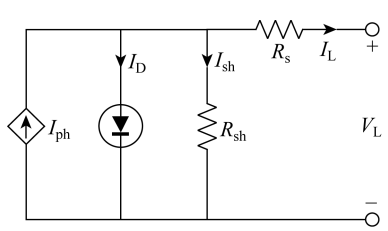

图 5 光伏电池等效电路

Fig. 5 Equivalent circuit of photovoltaic cell

式中:I_{sd}为二极管反向饱和电流;q_c、A_c、k_c、T_c分别为电子电荷量(1.602×10^{-19} C)、二极管理想因子、玻耳兹曼常量(1.380×10^{-23} J/K)与太阳电池工况下的绝对温度.

对一款直径为57 mm的商用(R. T. C France)硅太阳能电池进行参数辨识[19].在温度为33 ℃、光强为1 000 W/m^{2}的工况下,测得该电池的(I_{L,n_s}, V_{L,n_s})($n_s = 1, 2, \cdots, 26$)数据点作为辨识数据集,对参数I_{ph}、I_{sd}、A_c、R_s、R_{sh}进行辨识,参数实测值为$I_{ph}^* = 0.760\ 8$ A、$I_{sd}^* = 0.322\ 3$ A、$A_c^* = 1.483\ 7$、$R_s^* = 0.036\ 4\ \Omega$、$R_{sh}^* = 53.763\ 4\ \Omega$.为了衡量参数辨识的准确度,定义最终参数辨识值与实测值之间的差距即辨识误差如下:

$$E_{iden} = \frac{1}{5}\left(\frac{|I_{ph} - I_{ph}^*|}{I_{ph}^*} + \frac{|I_{sd} - I_{sd}^*|}{I_{sd}^*} + \frac{|A_c - A_c^*|}{A_c^*} + \frac{|R_s - R_s^*|}{R_s^*} + \frac{|R_{sh} - R_{sh}^*|}{R_{sh}^*}\right) \tag{14}$$

在参数辨识中,基于式(13)构建模型误差F_{pv},并以其作为目标函数,其表达式如下:

$$\left.\begin{aligned}&\min F_{\mathrm{pv}}=\sqrt{\frac{\sum_{n_s=1}^{26}\left|I_{\mathrm{ph}}-\frac{V_{\mathrm{L},n_s}+I_{\mathrm{L},n_s}R_{\mathrm{s}}}{R_{\mathrm{sh}}}-I_{\mathrm{L},n_s}-I_{\mathrm{sd}}\left\{\exp\left[\frac{q_{\mathrm{c}}(V_{\mathrm{L},n_s}+I_{\mathrm{L},n_s}R_{\mathrm{s}})}{A_{\mathrm{c}}k_{\mathrm{c}}T_{\mathrm{c}}}\right]-1\right\}\right|^2}{26}}\\&\text{s.t. }\ 0<I_{\mathrm{ph}}\leqslant 1\ \mathrm{A},\quad 0<I_{\mathrm{sd}}\leqslant 1\ \mu\mathrm{A},\quad 1\leqslant A_{\mathrm{c}}\leqslant 2,\quad 0<R_{\mathrm{s}}\leqslant 0.5\ \Omega,\quad 0<R_{\mathrm{sh}}\leqslant 100\ \Omega\end{aligned}\right\}\tag{15}$$

设置 SD-CHIO 与其他元启发式算法的初始解相同，迭代参数为 $N=30$、$T=500$，每种算法独立运行 30 次. 图 6 为 8 种算法的平均收敛曲线，可以看出，SD-CHIO 前期的收敛速度略低于 DE 与 ABC，但明显高于 CHIO 与 ICHIO；后期 CHIO 与 ICHIO 明显落入局部最优，而 SD-CHIO 成功跳出局部最优，且其求解结果在所有元启发式算法中具有最低的模型误差.

参数辨识结果如表 3 所示. 可见，SD-CHIO 得到最优 μ 与 s 值，表明其在参数辨识中较其他元启发式算法具有更强的全局与局部搜索能力以及鲁棒性. 同时，还给出每种元启发式算法在 30 次运行中得到的最小 F_{pv} 值，即 $F_{\mathrm{pv,min}}$. SD-CHIO 求得的 $F_{\mathrm{pv,min}}$ 值为 1.03×10^{-3} A，分别降低至 CHIO 与 ICHIO 的 22.25%与 10.49%；且该 $F_{\mathrm{pv,min}}$ 值对应的 E_{iden} 值在 8 种元启发式算法中最低，表明 SD-CHIO 辨识出的所有参数在整体上较其他元启发式算法更接近真实值.

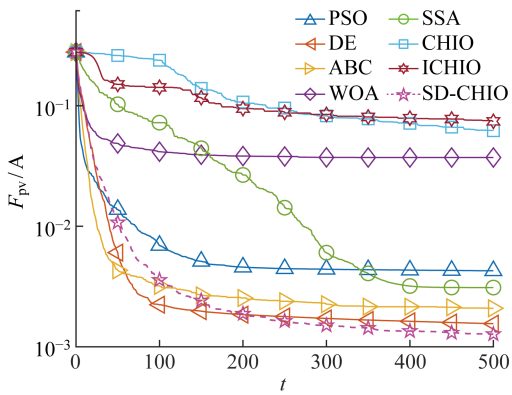

图 6　光伏电池参数辨识中不同元启发式算法的平均收敛曲线

Fig. 6　Average convergence curves of different MAs in parameter identification of photovoltaic cell

表 3　光伏电池参数辨识中不同元启发式算法的搜索结果

Tab. 3　Searching results of different MAs in parameter identification of photovoltaic cell

算法	μ/A	s/A	$F_{\mathrm{pv,\ min}}$/A	E_{iden}/%	I_{ph}/A	I_{sd}/μA	A_{c}	R_{s}/Ω	R_{sh}/Ω
PSO	4.27×10^{-3}	9.19×10^{-3}	1.55×10^{-3}	28.77	0.760 4	0.593 3	1.545 5	0.033 8	79.765 5
DE	1.56×10^{-3}	1.11×10^{-4}	1.42×10^{-3}	23.33	0.760 5	0.546 6	1.536 7	0.034 1	73.769 9
ABC	2.09×10^{-3}	2.65×10^{-4}	1.60×10^{-3}	27.32	0.759 9	0.581 2	1.543 5	0.033 7	77.778 3
WOA	3.72×10^{-2}	2.91×10^{-2}	1.36×10^{-3}	9.10	0.759 6	0.263 8	1.461 3	0.037 3	66.201 9
SSA	3.10×10^{-3}	3.34×10^{-3}	1.26×10^{-3}	19.23	0.760 0	0.457 4	1.517 4	0.035 1	79.733 8
CHIO	6.21×10^{-2}	4.14×10^{-2}	4.63×10^{-3}	31.37	0.764 6	0.598 6	1.546 6	0.034 8	87.131 5
ICHIO	7.42×10^{-2}	5.45×10^{-2}	9.82×10^{-3}	32.61	0.757 4	0.026 7	1.267 1	0.047 3	67.883 2
SD-CHIO	**1.27×10^{-3}**	**1.09×10^{-4}**	**1.03×10^{-3}**	**4.33**	0.760 7	0.359 1	1.492 4	0.036 0	58.306 5

4　结论

提出一种基于群体划分的冠状病毒群体免疫优化算法(SD-CHIO). 将初始群体划分为全局寻优个体与局部寻优个体两部分，并通过这两部分的作用来分别提高算法的全局与局部搜索能力. 通过基准函数与工程问题的测试得出如下结论：

(1) SD-CHIO 在单峰函数测试中，相比于原算法以及其他元启发式算法有更高的收敛速度与精度，表明其局部搜索能力相比于原算法大大增强.

(2) SD-CHIO 在多峰函数测试中，相比于原算法以及其他元启发式算法更易跳出局部最优解，在后期有更高求解精度，表明其全局搜索能力相比于原算法得到有效提高.

(3) 对于复杂函数的寻优，SD-CHIO 在所有元启发式算法中有最高求解精度，表明其在求解复杂优化问题时具有一定优势.

(4) 在函数测试中，SD-CHIO 求得的最优值标准差普遍小于其他元启发式算法，表明其具有较强鲁棒性.

(5) 在光伏电池参数辨识问题中，SD-CHIO 能较快求得具有较高精度的电池参数，表明该算法具

有一定的工程意义与实用价值.

参考文献：

[1] TAYARANI-N M H, YAO X, XU H M. Meta-heuristic algorithms in car engine design: A literature survey [J]. **IEEE Transactions on Evolutionary Computation**, 2015, 19(5): 609-629.

[2] 杜晓昕，王浩，崔连和，等. 基于聚类和探测精英引导的蜻蜓算法[J]. **浙江大学学报：工学版**，2022，56(5)：977-986.
DU Xiaoxin, WANG Hao, CUI Lianhe, *et al*. Dragonfly algorithm based on clustering and detection elite guidance[J]. **Journal of Zhejiang University (Engineering Science)**, 2022, 56(5): 977-986.

[3] BONYADI M R, MICHALEWICZ Z. Stability analysis of the particle swarm optimization without stagnation assumption[J]. **IEEE Transactions on Evolutionary Computation**, 2016, 20(5): 814-819.

[4] HAMZA N M, ESSAM D L, SARKER R A. Constraint consensus mutation-based differential evolution for constrained optimization[J]. **IEEE Transactions on Evolutionary Computation**, 2016, 20(3): 447-459.

[5] 杜振鑫，韩德志，刘广钟，等. 一种逐步加强开采的人工蜂群算法[J]. **上海交通大学学报**，2018，52(1)：96-102.
DU Zhenxin, HAN Dezhi, LIU Guangzhong, *et al*. Artificial bee colony algorithm with gradually enhanced exploitation[J]. **Journal of Shanghai Jiao Tong University**, 2018, 52(1): 96-102.

[6] MIRJALILI S. Dragonfly algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems[J]. **Neural Computing & Applications**, 2016, 27(4): 1053-1073.

[7] MIRJALILI S. SCA: A sine cosine algorithm for solving optimization problems[J]. **Knowledge-Based Systems**, 2016, 96: 120-133.

[8] MIRJALILI S, LEWIS A. The whale optimization algorithm [J]. **Advances in Engineering Software**, 2016, 95: 51-67.

[9] MIRJALILI S, GANDOMI A H, MIRJALILI S Z, *et al*. Salp swarm algorithm: A bio-inspired optimizer for engineering design problems[J]. **Advances in Engineering Software**, 2017, 114: 163-191.

[10] AL-BETAR M A, ALYASSERI Z A A, AWADALLAH M A, *et al*. Coronavirus herd immunity optimizer (CHIO)[J]. **Neural Computing & Applications**, 2021, 33(10): 5011-5042.

[11] DALBAH L M, AL-BETAR M A, AWADALLAH M A, *et al*. A modified coronavirus herd immunity optimizer for capacitated vehicle routing problem[J]. **Journal of King Saud University-Computer & Information Sciences**, 2022, 34(8): 4782-4795.

[12] DALBAH L M, AL-BETAR M A, AWADALLAH M A, *et al*. A coronavirus herd immunity optimization (CHIO) for travelling salesman problem [C]// **International Conference on Innovative Computing & Communications**. Singapore: Springer, 2022: 717-729.

[13] ALQARNI M. Sodium sulfur batteries allocation in high renewable penetration microgrids using coronavirus herd immunity optimization[J]. **Ain Shams Engineering Journal**, 2021, 13(2): 1-14.

[14] ALWESHAH M, ALKHALAILEH S, AL-BETAR M A, *et al*. Coronavirus herd immunity optimizer with greedy crossover for feature selection in medical diagnosis[J]. **Knowledge-Based Systems**, 2022, 235: 107629.

[15] NADERIPOUR A, ABDULLAH A, MARZBALI M H, *et al*. An improved corona-virus herd immunity optimizer algorithm for network reconfiguration based on fuzzy multi-criteria approach[J]. **Expert Systems with Applications**, 2022, 187: 115914.

[16] 夏学文，刘经南，高柯夫，等. 具备反向学习和局部学习能力的粒子群算法[J]. **计算机学报**，2015，38(7)：1397-1407.
XIA Xuewen, LIU Jingnan, GAO Kefu, *et al*. Particle swarm optimization algorithm with reverse-learning and local-learning behavior[J]. **Chinese Journal of Computers**, 2015, 38(7): 1397-1407.

[17] TANG K, YAO X, SUGANTHAN P N, *et al*. Benchmark functions for the CEC'2008 special session and competition on large scale global optimization [R]. Taiwan: National Chiao Tung University, 2007.

[18] KUMAR A, PRICE K V, MOHAMED A W, *et al*. Problem definitions and evaluation criteria for the CEC 2022 special session and competition on single objective bound constrained numerical optimization [R]. Singapore: Nanyang Technological University, 2021.

[19] XIONG G J, ZHANG J, YUAN X F, *et al*. Parameter extraction of solar photovoltaic models by means of a hybrid differential evolution with whale optimization algorithm [J]. **Solar Energy**, 2018, 176: 742-761.

（本文编辑：王历历）