Journal of System Simulation

Volume 35 | Issue 1 Article 2

1-18-2023

Chameleon Swarm Algorithm for Segmental Variation Learning of Population and S-type Weight

Damin Zhang
1.School of Big Data & Information Engineering, Guizhou University, Guiyang 550025, China;,
1203813362@qq.com

Yi Wang
1.School of Big Data & Information Engineering, Guizhou University, Guiyang 550025, China;,
ywang_gzu@163.com

Linna Zhang
2.School of Mechanical Engineering, Guizhou University, Guiyang 550025, China;

Chameleon Swarm Algorithm for Segmental Variation Learning of Population and S-type Weight

Abstract

Abstract: It is the best choice for intelligent algorithms to be applied to specific fields to explore strong searching ability, good reliability and stability.In this paper, aiming at the defects of chameleon swarm algorithm, such as unstable solution, low convergence accuracy and unbalanced search and development, a chameleon swarm algorithm (RMSCSA) based on population diversity segmental mutation learning and S-type weight is proposed. *The refraction mirror learning strategy (RML) is introduced to make the chameleon more consistent with the observation in nature and enhance its diversity. The introduction of segmental variation of population diversity can keep the individuals with poor fitness and guide them to the optimal position. The introduction of S-type decreasing weight makes it further balance the global search and explore ability of the algorithm, and obtains the advantage of S-type decreasing weight factor through convergence analysis.* The classical test function and CEC 2017 competition function are used for performance verification, and the results show that the three strategies have better optimization accuracy and efficiency for CSA. In order to compare the performance of different algorithms, the results of 30 independent runs are statistically analyzed by Wilcoxon rank-sum test, Friedman's test and Holm follow-up test. The analysis shows that the three strategies introduced have better optimization ability compared with CSA.

Keywords

chameleon swarm algorithm(CSA), refraction mirror learning(RML), diversity variation, S-type decreasing weight, statistical analyzed

Recommended Citation

Damin Zhang, Yi Wang, Linna Zhang. Chameleon Swarm Algorithm for Segmental Variation Learning of Population and S-type Weight[J]. Journal of System Simulation, 2023, 35(1): 11-26.

第 35 卷第 1 期 系统仿真学报© Vol. 35 No. 1
2023 年 1 月 Journal of System Simulation Jan. 2023

种群分段变异学习和S型权重变色龙群算法

张达敏[1]，王义[1*]，张琳娜[2]
(1. 贵州大学 大数据与信息工程学院，贵州 贵阳 550025；2. 贵州大学 机械工程学院，贵州 贵阳 550025)

摘要：探索寻优能力强、良好的可靠性和稳定性是智能算法应用到具体领域中的最佳选择。针对变色龙群算法存在求解不稳定、收敛精度低下和搜索开发之间不平衡等缺陷，提出一种种群多样性分段变异学习和S型权重的变色龙群算法(RMSCSA)。*引入折射镜像学习(refraction mirror learning，RML)策略使变色龙更符合自然界中的观察，增强它的多样性；引入种群多样性分段变异使适应度较差的个体得到保留，并引导它向最优位置靠近；S型递减权重的引入让它进一步平衡算法的全局搜索和开发能力，通过收敛性分析得出S型递减权重因子的优势。*利用经典函数集和CEC 2017函数集进行性能验证，结果表明3种策略比CSA具有更好寻优精度和效率。通过对独立运行30次的结果进行Wilcoxon秩和检验、Friedman’s以及Holm后续检验统计分析，结果表明引入的3种策略与CSA相比都有更好的寻优能力。

关键词：变色龙群算法；折射镜像学习；多样性变异；S型递减权重；统计分析
中图分类号：TP301.6 文献标志码：A 文章编号：1004-731X(2023)01-0011-16
DOI: 10.16182/j.issn1004731x.joss.21-0968
引用格式: 张达敏, 王义, 张琳娜. 种群分段变异学习和S型权重变色龙群算法[J]. 系统仿真学报, 2023, 35(1): 11-26.
Reference format: Zhang Damin, Wang Yi, Zhang Linna. Chameleon Swarm Algorithm for Segmental Variation Learning of Population and S-type Weight[J]. Journal of System Simulation, 2023, 35(1): 11-26.

Chameleon Swarm Algorithm for Segmental Variation Learning of Population and S-type Weight

Zhang Damin[1], *Wang Yi*[1*], *Zhang Linna*[2]
(1. School of Big Data & Information Engineering, Guizhou University, Guiyang 550025, China;
2. School of Mechanical Engineering, Guizhou University, Guiyang 550025, China)

Abstract: It is the best choice for intelligent algorithms to be applied to specific fields to explore strong searching ability, good reliability and stability. In this paper, aiming at the defects of chameleon swarm algorithm, such as unstable solution, low convergence accuracy and unbalanced search and development, a chameleon swarm algorithm (RMSCSA) based on population diversity segmental mutation learning and S-type weight is proposed. *The refraction mirror learning strategy (RML) is introduced to make the chameleon more consistent with the observation in nature and enhance its diversity. The introduction of segmental variation of population diversity can keep the individuals with poor fitness and guide them to the optimal position. The introduction of S-type decreasing weight makes it further balance the global search and explore ability of the algorithm, and obtains the advantage of S-type decreasing weight factor through convergence analysis.* The classical test function and CEC 2017 competition function are used

收稿日期：2021-09-16 修回日期：2021-11-10
基金项目：国家自然科学基金(62062021，61872034)；贵州省科学技术基金(黔科合基础[2020]1Y254)；贵州省自然科学基金(黔科合基础[2019]1064)
第一作者：张达敏(1967-)，男，教授，博士，研究方向为智能优化计算，认知无线电，机器学习。E-mail：1203813362@qq.com
通讯作者：王义(1997-)，男，苗族，硕士生，研究方向为智能计算，机器学习。E-mail：ywang_gzu@163.com

for performance verification, and the results show that the three strategies have better optimization accuracy and efficiency for CSA. In order to compare the performance of different algorithms, the results of 30 independent runs are statistically analyzed by Wilcoxon rank-sum test, Friedman's test and Holm follow-up test. The analysis shows that the three strategies introduced have better optimization ability compared with CSA.

Keywords: chameleon swarm algorithm(CSA); refraction mirror learning(RML); diversity variation; S-type decreasing weight; statistical analyzed

0 引言

智能算法一直是深受优化计算和控制理论等相关领域学者的关注。传统优化计算中，大部分以基于网格搜索(grid search, GS)[1]、K-Means[2]聚类分析等经典数值计算方法。随着信息社会进步的需求，各类优化问题的求解也变得越来越复杂，迫切需要找到应对问题的方案。近年来，基于仿生学的群体智能算法因机制灵活、时间复杂度低和算法结构简单等优点受到各相关领域研究者的青睐。如经典且具有代表性的有粒子群算法(particle swarm optimization, PSO)[3]、遗传算法(genetic algorithm, GA)[4]、差分进化算法(differential evolution, DE)[5]、蚁群算法(ant colony optimization, ACO)[6]等。

随着几种经典智能算法的提出，研究人员不断通过其他生物的社会行为研究提出新算法，如Mirjalili等根据樽海鞘群体的领导者和跟随着的引领行为提出樽海鞘群算法(salp swarm algorithm, SSA)[7]，Seyedali等观察鲸鱼气泡“结网”和变螺旋寻食方式的启发提出鲸鱼优化算法(whale optimization algorithm, WOA)[8]以及Konstantinos等根据雄雌蜉蝣的婚配吸引行为提出蜉蝣算法(mayfly algorithm, MA)[9]等，这些算法的提出都为解决复杂问题提供新思路。

变色龙群算法(chameleon swarm algorithm, CSA)[10]是Malik等在沙漠中观察变色龙寻食受启发提出。研究人员发现变色龙相比于其他生物有着不同的优点：眼睛能进行360°旋转观察猎物，并使用它们高速发射的粘性舌头捕捉猎物。这项优势增强了它发现猎物的概率和捕捉食物的能力。变色龙群算法获取食物主要有3个阶段：猎物搜索、眼睛旋转发现猎物和猎物捕捉。

CSA与其他算法相比具有不同的搜索方式，但仿生算法本身易陷入局部最优、精度低等特点难以改变。针对这些缺陷，研究人员对仿生算法不足做了相应的改进，目的是提高收敛速度、精度、算法的性能以及平衡它们搜索和开发能力。如何庆等[11]利用柯西变异和均匀分布的思想，对蝗虫算法位置搜索易陷入局部最优进行改进，有效提高算法的搜索能力；吴文海等[12]利用种群的邻域选择思想平衡算法全局搜索和局部开发，并利用概率的方法结合广义反向学习策略，加速算法的收敛潜能；范千等[13]引入折射的思想，将樽海鞘链寻食过程折射化，增强多样性改善樽海鞘群算法的寻优能力；Roshan等[14]提出变异策略和权重分布，分析裸鼹鼠算法搜索能力和对历史位置迭代的敏感依赖性分析，最终选择最好的策略改进算法，对高维复杂问题的求解提供良好的解决方案。

本文针对CSA的近似解精度不高和搜索性能低下的缺点，提出种群多样性分段变异学习和S型权重的变色龙群算法(RMSCSA)。3种策略为：①引入折射镜像学习策略(refraction mirror learning, RML)的增强求解的多样性，获得更多候选解提高算法的寻优能力；②种群多样性分段变异策略的引入使寻食过程中保持种群的多样性，平衡算法的搜索和开发。引入共生量加强群体间信息交流合作，从而提高开发能力；③上一次迭代的位置和速度信息会为下一次迭代起指引作用，文中提出一种S型递减权重因子，使前期对上一

DOI: 10.16182/j.issn1004731x.joss.21-0968

次位置的依赖性较强，提高全局搜索能力；后期削减依赖性趋于自由搜索提高开发能力，并通过机理性分析S型递减权重的优势。最后，理论证实该算法没有提高时间复杂度，引入优化测试函数集和IEEE CEC竞赛集对比文中提出的3种策略了算法的有效性和稳定性；另外对算法进行统计检验，用统计学的视野证明与其他算法的显著性差异，并与近些年提出的改进算法进行对比。多项分析表明，提出的算法具有较好的寻优精度、寻优性能和求解稳定性。

1 变色龙群算法

CSA的提出是受变色龙生物发现食物和捕食行为而得到的启发。与变色龙算法(chameleon algorithm, CA)[15]不同的是，CSA能够解决约束问题和全局数值优化问题，而CA是一种自底向上的聚类算法，在数据挖掘场景中能够动态对数据集进行分类。变色龙群算法的主要过程如下：

设位置矢量为$\boldsymbol{X}$，种群大小为n，在d维空间搜索。其位置向量$\boldsymbol{X}$的$n\times d$矩阵为

$$\boldsymbol{X}=\begin{bmatrix} x_{11} & \cdots & x_{1d} \\ \vdots & & \vdots \\ x_{n1} & \cdots & x_{nd} \end{bmatrix} \tag{1}$$

变色龙维度空间的位置初始化表示为

$$x_{ij}=lb+r(ub-lb) \tag{2}$$

式中：ub为搜索空间的上边界；lb为搜索空间的下边界；r为取值(0, 1)的随机数。

1.1 猎物搜索

变色龙群体在觅食过程中的运动行为通过以下表达式进行食物的发现：

$$x_{ij}^{t+1}=\begin{cases} x_{ij}^{t}+p_1r_2(P_{ij}^{t}-G_j)+p_2r_1(G_j-x_{ij}^{t}), & r\geqslant P_p \\ x_{ij}^{t}+\mu(r_3(ub-lb)+lb)\mathrm{sgn}(rand-0.5), & r<P_p \end{cases} \tag{3}$$

式中：P_p为感知食物的概率。若$r\geqslant P_p$，变色龙可以根据在搜索空间中观察到的猎物来改变自己的位置。同理若$r<P_p$变色龙在寻找猎物时，会在不同方向和区域随机探索搜索空间，这使它们很大可能感知到附近的目标猎物。为衡量随机和观测之间的平衡搜索，取$P_p=0.1$；$\mathrm{sgn}(rand-0.5)$用于控制旋转方向，取值为+1或−1。式(3)各参数的说明见表1。

表1 猎物搜索的参数说明

Table 1 Parameter description of prey search

参数名称	说明
x_{ij}^{t}	变色龙i在j维空间中，第t次迭代的位置
P_{ij}^{t}	变色龙i在j维空间经t次迭代的最佳位置
G_j	变色龙在t次迭代的全局最优位置
p_1, p_2	用于控制算法开发能力的正系数
r_1, r_2, r_3, r	取值范围为(0, 1)的随机数
P_p	变色龙的感知概率
μ	随t控制的搜索能力参数
$\mathrm{sgn}(rand-0.5)$	控制旋转方向

参数μ的表达式为

$$\mu=\mathrm{e}^{(-\alpha t/T)^3} \tag{4}$$

文献[10]已对CSA的参数敏感性进行了分析，当$\alpha=3.5$有较好的搜索性能，本文取3.5。T为最大迭代次数，t为当前迭代次数。

1.2 变色龙眼睛的转动

变色龙的显著优势是它能够利用眼睛的旋转特征识别猎物位置，能360°旋转发现猎物，用眼睛模拟变色龙猎物发现行为，会根据猎物的位置来更新自己的位置，这一过程建模为数学关系为

$$x_i^{t+1}=\boldsymbol{m}(x_i^{t}-\bar{x}_i^{t})+\bar{x}_i^{t} \tag{5}$$

$$\boldsymbol{m}=R(\theta,\ \boldsymbol{V}_{z_1,\,z_2}) \tag{6}$$

$$\theta=r\cdot\mathrm{sgn}(rand-0.5)\pi \tag{7}$$

式中：$\bar{x}_i^{t}$为在t次迭代的中心位置；x_i^{t}为当前位置；x_i^{t+1}为经旋转矩阵翻译后生成的新位置；$\boldsymbol{m}(x_i^{t}-\bar{x}_i^{t})$为搜索空间当前位置和中心位置与旋转矩阵变换形成的位置；$\boldsymbol{m}$为旋转矩阵；z_1、z_2为搜索空间的两个正交向量；$\boldsymbol{V}_{z_1,z_2}$为z_1、z_2形成的正交矩阵；R为旋转矩阵的生成，生成过程见文献[10]。式(7)为变色龙眼睛转动生成的方向角，r为(0, 1)的随机数，$\mathrm{sgn}(rand-0.5)$控制变色龙左右旋

转的方向，取值为+1或−1，因此方向角的范围为[−π, π]。综上描述，变色龙眼睛转动发现猎物主要有4个阶段：①计算原始变色龙的位置中心；②确定猎物位置的旋转矩阵；③经旋转矩阵变换形成变换的位置；④得到的变换位置并结合原始中心位置得到变换后的位置。变色龙眼睛全方位旋转发现猎物这一优势，极大提高算法发现猎物的概率。

1.3 猎物捕捉

与其他生物不同的是，变色龙具有非常长的舌头，是很好的捕食工具。当猎物离变色龙较近时，变色龙这时会用舌头攻击猎物结束狩猎。由此，变色龙向猎物扑去时舌头速度的数学模型为

$$v_{ij}^{t+1}=v_{ij}^{t}+c_1r_1(G_j-x_{ij}^{t})+c_2r_2(P_{ij}^{t}-x_{ij}^{t}) \tag{8}$$

式中：v_{ij}^{t+1}为在经$t+1$迭代后更新的速度；v_{ij}^{t}为第t次迭代的速度；x_{ij}^{t}为第t次迭代的位置；G_j为全局最优位置；P_{ij}^{t}为局部最优位置；r_1和r_2为(0, 1)的随机数；c_1和c_2为G_j和P_{ij}^{t}的认知因子。

$$x_{ij}^{t+1}=x_{ij}^{t}+\frac{(v_{ij}^{t})^2-(v_{ij}^{t-1})^2}{2a} \tag{9}$$

式中：v_{ij}^{t}为当前的速度；v_{ij}^{t-1}为上一次迭代的速度；x_{ij}^{t}为当前的变色龙位置；a为加速度，根据运动学中推导的。加速度a的表达式为

$$a=2\,590(1-\mathrm{e}^{-\lg t}) \tag{10}$$

式中：t为当前的迭代次数。加速度会随t的变化而非线性变化。

2 种群多样性分段变异学习和S型权重的变色龙群算法

2.1 融合镜像学习的分段学习策略

近些年也有很多学者在研究智能算法中利用反向学习方法[16]，反向学习会生成可行解的反向解，对反向解评价选择更好的候选解。镜像学习[17]和反向学习不同的是：镜像学习策略是利用光的镜像实现种群整体突变。折射镜像学习策略(RML)结合了光的折射定律和镜像选择特性归纳出的改进策略，能增强算法的多样性寻找更好的候选解。折射镜像学习策略如图1所示。

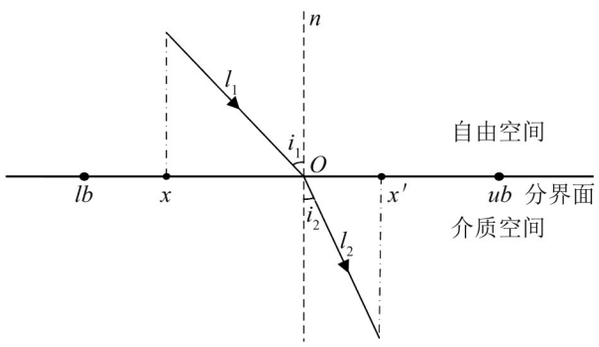

图1 折射镜像学习图

Fig. 1 Refraction mirror learning diagram

假设在自然空间区域中，设入射角与法线n的夹角为i_1，出射角与法线n的夹角为i_2，入射光线和出射光线的长度分别为l_1和l_2。折射定律为

$$n_0\sin i_1=n_1\sin i_2 \tag{11}$$

式中：n_0为自由空间的折射率，通常取1；n_1为介质折射率，通常$n_1>1$。由几何关系：

$$\sin i_1=\frac{(ub+lb)/2-x}{l_1} \tag{12}$$

$$\sin i_2=\frac{x'-(ub+lb)/2}{l_2} \tag{13}$$

结合折射定律，令入射光线长度与出射光线长度$l_1/l_2=k$，整合得

$$kn_1=\frac{(ub+lb)/2-x}{x'-(ub+lb)/2} \tag{14}$$

再反解x'，得到

$$x'=\frac{(ub+lb)}{2}+\frac{(ub+lb-2x)}{2kn_1} \tag{15}$$

取$k=n_1=1$时，有

$$x'=ub+lb-x \tag{16}$$

显然当k、n_1为1时，式(16)只为折射镜像学习的特例，候选解的方式较固定。而RML按照取值不同可以调整参数获得动态候选解，增加算法跳出局部最优的概率。为增强解的搜索多样性提高寻优能力，以一定概率交替执行镜像学习和RML策略，让候选解能随参数动态更新，既保留镜像学习的优势，又能充分发挥RML的优势。本

DOI: 10.16182/j.issn1004731x.joss.21-0968

文分段搜寻表达式如下：

$$x'=\begin{cases} ub+lb-x, & rand<AP \\ \dfrac{ub+lb}{2}+\dfrac{ub+lb-2x}{2kn_1}, & rand\geqslant AP \end{cases} \tag{17}$$

式中：AP为选择阈值概率，生成随机数高于AP选用RML，低于AP则选择镜像学习。经多次独立实验，$AP=0.35$时具有最佳的寻优效果。

2.2 种群多样性分段变异

2.2.1 多样性变异

良好的多样性能提高算法的收敛能力。文献[18]报道的种群多样性概念对遗传算法改进，实验证明该方法用于遗传算法有更快的收敛速度和精度，提高了全局搜索能力。本文多样性变异步长借鉴自适应指数概率密度分布$f(x)=te^{-tx}$, $x\in[0,\ t-1]$，为一个广义分布的值。表达式为

$$te^{-t}-e^{-t}-t+1=0 \tag{18}$$

对式(18)广义推导[19]，得到多样性变异算子为

$$\lambda(t)=\frac{\lg(1-rand\cdot(1-e^{-t}))}{t} \tag{19}$$

式中：$\lambda(t)$为多样性变异步长；t为迭代次数。

2.2.2 种群多样性分段变异的位置更新

群体经一次迭代，会产生适应度较好较差的变色龙个体。对于适应度较好的个体，是群体的精英，容易搜索到空间中的猎物，因而对于这类变色龙个体不需要进行变异；对于适应度较差的变色龙个体，本文并未直接淘汰。为提升算法的开发能力，保证算法收敛性，利用多样性变异引导群体向猎物靠近或指引它们发现猎物；对于适应度一般的个体，利用局部变异轻微调整，使子群向精英个体学习。

综合上述阐述，本文提出的多样性变异策略应用于分段子群的表达式为

$$x_{ij}^t=\begin{cases} x_{ij}^t, & F(x_i)\leqslant 0.15N \\ x_{ij}^t+rand\cdot(\lambda(t)P_{\text{best}}-R), & 0.15N<F(x_i)\leqslant 0.6N \\ x_{ij}^t+rand\cdot(\lambda(t)G_{\text{best}}-R), & F(x_i)>0.6N \end{cases} \tag{20}$$

式中：N为变色龙群体总的数量；$F(x_i)$为经迭代后，按照适应度排名的序列值，根据适应度值选择具体搜索方式。$R=(x_{ij}^t+x_{i-1,j}^t)/2$为共生量，目的是变色龙个体$i$和$i-1$相互作用，在社会部分引入组合模式加强个体间信息交流与合作，增强进化的关联性，提升算法搜索和开发能力。

2.3 S型递减惯性权重因子

2.3.1 S型递减动态惯性权重

惯性权重体现了当前搜索更新的位置与前一次迭代位置的指引关系，具有继承前一次位置更新的经验和能力。较大的惯性权重具有全局搜索能力，较小的惯性权重对局部搜索有较好开发能力[19]。通常，迭代初期尽可能需要较大的惯性权重增强算法的全局搜索能力；迭代后期，需要以较小的惯性权重保持优良的局部搜索能力，即后期的重点探索开发能力。结合Salgotra等[20]提出的S型函数，符合前期权重大，后期权重相对小的特征。基于此本文改进S型递减函数的惯性权重，表达式为

$$\omega(t)=\omega_{\max}+\frac{(\omega_{\min}-\omega_{\max})}{1+\exp(-u(T-t-h\cdot gen))} \tag{21}$$

$$u=10^{\lg(gen)-2} \tag{22}$$

式中：$\omega_{\max}$为最大权重；$\omega_{\min}$为最小权重；t为当前迭代；T为最大迭代次数；h为一个随机数，取值为(0, 1)；gen为进化因子，取值为50[14]。结合CSA的更新特点，将S型递减动态权重引入速度和位置更新的表达式如下：

$$x_i^{t+1}=m(\omega x_i^t-\bar{x}_i^t)+\bar{x}_i^t \tag{23}$$

$$v_{ij}^{t+1}=\omega v_{ij}^t+c_1r_1(G_j-x_{ij}^t)+c_2r_2(P_{ij}^t-x_{ij}^t) \tag{24}$$

$$x_{ij}^{t+1}=\omega x_{ij}^t+((v_{ij}^t)^2-(v_{ij}^{t-1})^2)/2a \tag{25}$$

由式(23)~(25)，速度和位置更新由2部分组成：惯性部分和社会认知部分。t次迭代速度/位置会直接影响$t+1$次迭代更新，即惯性部分对位置更新的依赖性较大。前期，算法依靠先验信息的指导搜索能快速锁定最优解的局部位置，即对t次迭代信息有较大的依赖性，此时权重值应较大，

使算法前期快速全局搜索；随着迭代进行，若t次迭代时陷入局部最优解，$t+1$更容易陷入局部最优，此时利用惯性权重削减t次先验学习的继承，主要依靠认知部分使个体间进行信息交流合作，提高算法开发能力。

2.3.2 微分方程下的收敛性分析

吴凡等[21]利用运动学和微分方程推导了粒子群算法(PSO)的收敛性分析，理论证明了惯性权重对粒子群算法收敛速度和搜索能力的影响。CSA在猎物捕捉阶段的速度更新表达式一致，文中引用吴凡等的理论研究成果，对几种不同的惯性权重加以分析。微分方程下整理的位置表达式为

$$x_{ij}^{t}=\mathrm{e}^{\frac{\omega-1}{2}t}\sqrt{A^2+B^2}\sin(kt+\theta) \tag{26}$$

$$\theta=\arctan(A/B) \tag{27}$$

式中：A、B为它的通解得到的常数；ω为惯性权重因子；k为微分方程的虚部，详细推导过程见文献[21]。Malik的研究中引入自适应线性递减的惯性权重[10]，经收敛性分析发现存在一定缺陷。文献[22]试图利用反向思维提出随迭代次数S型递增的惯性权重，图2仿真上述人员所提的几种权重收敛趋势。

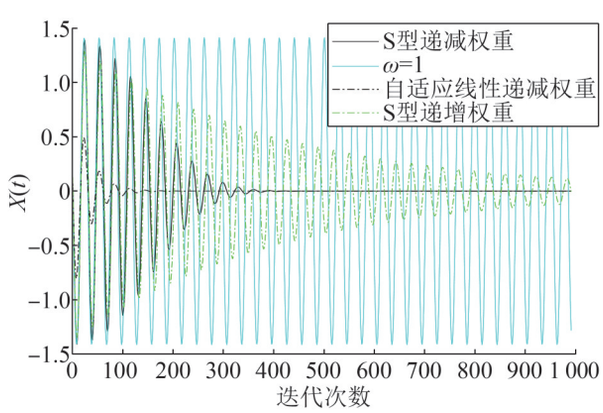

图2 不同惯性权重的搜索趋势
Fig. 2 Search trend of different inertial weights

由图2，当$\omega=1$时整个过程保持原始搜索，完全依赖于前一次更新的位置，所以效果不佳；自适应线性递减的惯性权重在前期收敛快，约100次后就进入局部搜索。若以这种方式持续搜索，前期缺少种群多样性，会淘汰适应度较差个体，易陷入局部最优；而S型递增惯性权重从图中看出，尽管搜索空间在渐渐变小，但相比开发能力依然在缺失，在后期难以搜索到最优解；文中S型递减权重相对较好，原因是前期全局搜索能力和后期开发能力之间的交替较好，不会对适应性差的个体淘汰，搜索和开发也能达到较好平衡。

2.4 RMSCSA算法实现流程及时间复杂度分析

2.4.1 RMSCSA算法的实现流程

RMSCSA的算法实现流程图如图3所示。

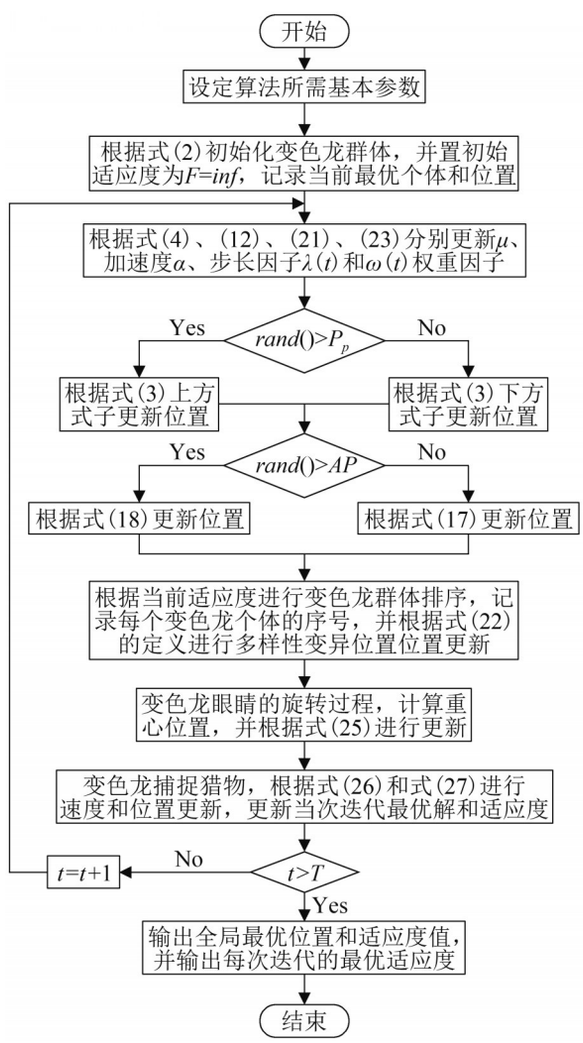

图3 RMSCSA算法流程
Fig. 3 Flow chart of RMSCSA algorithm

DOI: 10.16182/j.issn1004731x.joss.21-0968

2.4.2 时间复杂度分析

算法的时间复杂度是衡量算法执行速度和效率的关键指标，所以本文对RMSCSA进行时间复杂度分析是有必要的。假设变色龙的群体数量为n，搜索维度为d，最大迭代次数为T。根据时间复杂度的计算方法，初始化的时间复杂度为$O(n\times d)$。

迭代环内，定义的参数，如物理加速度a，μ，变异步长λ以及S型惯性权重因子ω，复杂度均为$O(1)$。

猎物搜索阶段，这一过程为猎物搜索，其复杂度为$O(n\times d)$。引入上述策略后，在位置更新呢下方增加了if语句，复杂度不变，同为$O(n\times d)$。引入位置的分段变异后，下方位置更新，其复杂度为$O(1)$。所以这一阶段时间复杂度为$O(n\times d+1)$。

变色龙眼睛转动阶段，对每个变色龙个体的位置更新，由于旋转变换过程较复杂，设生成旋转矩阵时间为t_1，翻译位置和生成新位置的时间设为t_2。所以变色龙眼睛转动阶段复杂度为$O(n\times(t_1+t_2))$。

猎物捕捉阶段，主要更新的公式为式(26)和(27)。其时间复杂度为$O(n\times d+n)$。

综上，RMSCSA的时间复杂度为$T(n)=O(n\times d+T\times(4+n\times d+1+n\times(t_1+t_2)+n\times d+n))$。简化后，复杂度为$T(n)=O((T+1)\times n\times d)$。由此可看出，RMSCSA与CSA保持同一量级，并未增加算法复杂度。

3 实验测试与分析

为了权衡以上策略的有效性和可靠性，需对算法进行对比分析。本文设置2个对比实验，利用经典测试函数为有效性分析和CEC 2017竞赛集对比性能。设文中引入RML策略的变色龙群算法记为(RCSA)、种群多样性分段变异策略的变色龙群算法记为(MCSA)、S型递减权重策略的变色龙群算法记为(SCSA)及混合策略的变色龙群算法记为(RMSCSA)。实验环境均在Windows10操作系统，12G运行内存，版本为MATLAB 2018a进行测试。

3.1 实验参数设置

实验选取变色龙群体的数量为$n=60$，最大迭代次数为$T=1\,000$。每个测试函数均独立重复实验30次，便于数据统计学差异分析。表2为各算法的主要参数。

表2 各算法的主要参数设置
Table 2 Main parameter Settings of each algorithm

算法	主要参数
CSA	P_p=0.1, α=3.5, p_1=0.25, $p_2=1.5$, $c_1=1.75$, $c_2=1.75$
RCSA	$n_1=1.0\sim1.5$, $k=1$, $AP=0.35$
MCSA	$n=50$, N为种群大小
SCSA	$gen=51$, $\omega_{max}=0.95$, $\omega_{min}=0.55$
SSA	--
WOA	$a_{max}=2$, $a_{min}=0$

3.2 实验结果及分析

实验1：实验引用14个复杂测试函数作为目标函数验证算法有效性和可靠性。选取测试函数类型为6个单峰(Unimodal)函数、4个多峰(Multimodal)函数及4个固定维(Fix-dimension)函数，函数特征为：可分(Separable)和不可分(Non-Separable)。本文利用不同函数特征，不同维度和不同类型的函数全面验算算法性能。14个函数特征信息见表3。

表4记录了单峰函数和多峰函数在维度$d=$10, 30, 50维的平均值、标准差和独立运行的平均耗时。固定维函数$F_{11}\sim F_{14}$维度较低，但函数特征较为复杂，所以对低维复杂特征的函数对比也能突显算法的有效性，可说明算法在复杂特征函数上的搜索性能。表5记录了4个固定维函数的平均值、标准差和独立运行的平均耗时的对比结果，加强算法在固定维函数的有效性对比。

表3 14个经典测试函数信息

Table 3 Information of 14 classical test functions

函数名称	函数公式	特征	取值空间	最优取值
Sphere	$F_1(X)=\sum_{i=1}^{n}x_i^2$	US	[−100, 100]	0
Schwefel 2.22	$F_2(X)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	US	[−100, 100]	0
Schwefel 1.2	$F_3(X)=\sum_{i=1}^{n}(\sum_{j=1}^{n}x_j)^2$	UN	[−100, 100]	0
Schwefel 2.21	$F_4(X)=\max_{1\leqslant i\leqslant n}\{\lvert x_i\rvert\}$	US	[−100, 100]	0
Rosenbrock	$F_5(X)=\sum_{i=1}^{n}[100(x_{i+1}-x_i^{\,2})^2+(1-x_i^{\,2})^2]$	UN	[−30, 30]	0
Quartic	$F_6(X)=\sum_{i=1}^{n}ix_i^4+random[0,\ 1)$	US	[−1.28, 1.28]	0
Rastrigin	$F_7(X)=\sum_{i=1}^{n}x_i^{\,2}-10\cos(2\pi x_i+10)$	MS	[−5.12, 5.12]	0
Ackley	$F_8(X)=-20\exp\left(-\frac{0.2\sqrt{\sum_{i=1}^{n}x_i^2}}{n}\right)-\exp\left(\sum_{i=1}^{n}\cos\left(\frac{2\pi x_i}{n}\right)\right)+20+e$	MN	[−32, 32]	0
Griewank	$F_9(X)=\frac{1}{4\,000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	MN	[−600, 600]	0
Penalized 2	$F_{10}(X)=0.1\{\sin^2 3\pi x_1+\sum_{i=1}^{n-1}(x_i-1)^2\,[1+\sin^2 3\pi x_{i+1}]+(x_n-1)^2+\sum_{i=1}^{n}u(x_i,\ 5,100,\ 4)\}$	MN	[−50, 50]	0
Foxholes	$F_{11}(X)=[\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{j=1}^{2}(x_i-a_{ij})^6}]^{-1}$	MN	[−65.53, 65.53]	≈1
Kowalik	$F_{12}(X)=\sum_{i=1}^{11}[a_i-\frac{x_1(b_i^{\,2}+b_ix_2)}{b_i2+b_ix_3x_4}]^2$	MN	[−5, 5]	3.075×10^{-4}
Branin	$F_{13}(X)=(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6)^2+10\left(1-\frac{1}{8\pi}\right)\cos x_1+10$	MN	$x_1\in[-5,\ 10]$ $x_2\in[0,\ 15]$	0.398
Hartman	$F_{14}(X)=-\sum_{i=1}^{4}c_i\exp[-\sum_{j=1}^{6}a_{ij}\ (x_j-p_{ij})^2]$	MS	[0,1]	−3.32

表4 不同维度下各策略的测试函数对比

Table 4 Comparison of test functions of different policies in different dimensions

函数	算法名	$d=10$			$d=30$			$d=50$		
		平均值	标准差	耗时/s	平均值	标准差	耗时/s	平均值	标准差	耗时/s
F_1	CSA	5.71×10^{-10}	1.44×10^{-10}	1.66	3.37×10^{-7}	1.31×10^{-7}	6.60	3.28×10^{-6}	5.32×10^{-7}	10.67
	RCSA	2.97×10^{-90}	1.57×10^{-89}	1.82	3.71×10^{-86}	1.45×10^{-85}	6.89	6.03×10^{-89}	3.28×10^{-88}	12.46
	MCSA	2.2×10^{-163}	6.4×10^{-162}	1.65	4.3×10^{-117}	9.9×10^{-117}	6.71	1.2×10^{-116}	6.6×10^{-116}	13.55
	SCSA	4.48×10^{-78}	2.37×10^{-77}	1.71	1.31×10^{-34}	4.56×10^{-32}	7.01	3.97×10^{-21}	3.38×10^{-20}	11.62
	RMSCSA	**0**	**0**	1.88	**0**	**0**	7.51	**0**	**0**	11.65

续表

函数	算法名	$d=10$			$d=30$			$d=50$		
		平均值	标准差	耗时/s	平均值	标准差	耗时/s	平均值	标准差	耗时/s
F_2	CSA	1.51×10^{-6}	6.84×10^{-6}	1.67	2.47	3.13	6.58	1.31×10^{2}	8.40×10	10.54
	RCSA	4.78×10^{-44}	2.57×10^{-43}	1.81	2.75×10^{-43}	1.49×10^{-42}	6.61	5.84×10^{-45}	3.18×10^{-44}	11.28
	MCSA	5.13×10^{-84}	2.32×10^{-83}	2.10	2.37×10^{-61}	6.22×10^{-61}	7.56	2.23×10^{-65}	1.09×10^{-64}	11.25
	SCSA	2.35×10^{-35}	6.32×10^{-33}	1.71	6.19×10^{-18}	4.48×10^{-17}	6.84	2.36×10^{-13}	1.84×10^{-12}	11.22
	RMSCSA	$\mathbf{1.3\times10^{-197}}$	$\mathbf{3.2\times10^{-195}}$	2.15	$\mathbf{1.1}\times10^{-188}$	$\mathbf{1.1}\times10^{-187}$	6.77	$\mathbf{2.12\times10^{-99}}$	$\mathbf{7.9\times10^{-99}}$	11.32
F_3	CSA	3.92×10^{-7}	9.31×10^{-7}	1.77	7.88	9.53	7.00	4.39×10^{3}	1.09×10^{3}	11.02
	RCSA	8.64×10^{-91}	3.11×10^{-90}	2.03	2.23×10^{-83}	1.22×10^{-82}	6.95	2.95×10^{-82}	1.16×10^{-81}	10.99
	MCSA	5.7×10^{-113}	4.5×10^{-112}	1.88	1.14×10^{-75}	5.39×10^{-75}	7.51	4.55×10^{-77}	2.49×10^{-76}	11.43
	SCSA	2.03×10^{-79}	3.72×10^{-78}	1.81	9.42×10^{-31}	1.69×10^{-29}	7.22	3.38×10^{-37}	2.94×10^{-35}	11.33
	RMSCSA	**0**	**0**	2.02	**0**	**0**	7.35	$\mathbf{3.0\times10^{-281}}$	$\mathbf{9.1\times10^{-281}}$	11.19
F_4	CSA	1.14×10^{-5}	7.20×10^{-6}	1.67	3.50×10^{-3}	6.50×10^{-3}	6.69	2.25×10^{-2}	2.14×10^{-2}	10.50
	RCSA	4.75×10^{-45}	2.54×10^{-44}	1.83	5.65×10^{-43}	3.09×10^{-42}	6.89	5.78×10^{-45}	2.28×10^{-44}	10.79
	MCSA	2.41×10^{-80}	8.67×10^{-80}	2.16	2.33×10^{-37}	1.27×10^{-36}	8.12	5.08×10^{-51}	5.77×10^{-50}	10.20
	SCSA	3.79×10^{-44}	1.39×10^{-44}	1.72	2.61×10^{-17}	2.63×10^{-18}	6.88	7.59×10^{-15}	3.36×10^{-16}	9.95
	RMSCSA	$\mathbf{4.5\times10^{-178}}$	$\mathbf{3.6\times10^{-178}}$	2.11	$\mathbf{6.3\times10^{-182}}$	$\mathbf{3.3\times10^{-181}}$	6.91	$\mathbf{5.7\times10^{-178}}$	$\mathbf{2.9\times10^{-176}}$	10.07
F_5	CSA	1.95	4.68	1.71	2.46	2.59	6.69	2.51×10^{2}	3.81×10^{2}	10.59
	RCSA	4.06×10^{-7}	7.73×10^{-6}	2.01	1.81×10^{-5}	4.04×10^{-5}	7.03	5.29×10^{-5}	1.03×10^{-4}	11.83
	MCSA	$8.\ 2\times10^{-4}$	7.54×10^{-3}	2.06	2.89×10^{-2}	1.70×10^{-2}	7.24	4.0×10^{-7}	$\mathbf{3.31\times10^{-5}}$	11.56
	SCSA	1.11	7.51×10^{-1}	1.72	2.86	2.14×10^{-1}	7.89	4.87	4.89	11.21
	RMSCSA	$\mathbf{1.40\times10^{-9}}$	$\mathbf{1.62\times10^{-9}}$	2.39	$\mathbf{1.92\times10^{-9}}$	$\mathbf{3.21\times10^{-7}}$	7.21	$\mathbf{2.63\times10^{-7}}$	8.61×10^{-5}	11.31
F_6	CSA	3.55	3.02	1.68	1.26×10^{2}	2.21×10	6.62	5.45×10^{2}	5.36×10	10.52
	RCSA	3.26×10^{-5}	4.31×10^{-5}	1.96	2.49×10^{-5}	2.28×10^{-5}	6.71	2.65×10^{-5}	3.95×10^{-5}	12.51
	MCSA	4.17×10^{-5}	2.27×10^{-5}	2.21	2.53×10^{-5}	$\mathbf{2.14\times10^{-6}}$	6.55	1.38×10^{-7}	1.03×10^{-6}	11.85
	SCSA	1.21×10^{-2}	4.10×10^{-1}	1.95	2.98×10^{-2}	5.58×10^{-1}	6.55	5.36×10^{-2}	1.17×10^{-2}	11.29
	RMSCSA	$\mathbf{3.61\times10^{-7}}$	$\mathbf{1.25\times10^{-7}}$	2.44	$\mathbf{2.62\times10^{-6}}$	1.77×10^{-5}	6.52	$\mathbf{1.37\times10^{-7}}$	$\mathbf{1.00\times10^{-6}}$	11.18
F_7	CSA	9.44×10^{-1}	1.15	1.67	1.39×10	2.75×10	6.05	3.11×10^{2}	1.94×10	10.63
	RCSA	**0**	**0**	1.96	**0**	**0**	6.56	**0**	**0**	11.02
	MCSA	**0**	**0**	2.12	**0**	**0**	6.71	**0**	**0**	11.43
	SCSA	3.85×10^{-98}	2.95×10^{-97}	1.81	2.55×10^{-97}	1.78×10^{-96}	6.96	**0**	**0**	11.41
	RMSCSA	**0**	**0**	2.09	**0**	**0**	6.31	**0**	**0**	12.69
F_8	CSA	2.92×10^{-5}	2.32×10^{-6}	1.71	1.02	1.54	6.03	1.91	1.02×10^{-1}	10.64
	RCSA	$\mathbf{8.88\times10^{-16}}$	$\mathbf{8.88\times10^{-16}}$	2.00	$\mathbf{8.88\times10^{-16}}$	$\mathbf{8.88\times10^{-16}}$	6.21	$\mathbf{8.88\times10^{-16}}$	$\mathbf{8.88\times10^{-16}}$	12.08
	MCSA	$\mathbf{8.88\times10^{-16}}$	$\mathbf{8.88\times10^{-16}}$	2.05	$\mathbf{8.88\times10^{-16}}$	$\mathbf{8.88\times10^{-16}}$	6.05	$\mathbf{8.88\times10^{-16}}$	$\mathbf{8.88\times10^{-16}}$	10.66
	SCSA	$\mathbf{8.88\times10^{-16}}$	$\mathbf{8.88\times10^{-16}}$	1.77	2.04×10^{-12}	2.47×10^{-12}	6.22	7.26×10^{-13}	5.47×10^{-12}	11.03
	RMSCSA	$\mathbf{8.88\times10^{-16}}$	$\mathbf{8.88\times10^{-16}}$	2.07	$\mathbf{8.88\times10^{-16}}$	$\mathbf{8.88\times10^{-16}}$	6.55	$\mathbf{8.88\times10^{-16}}$	$\mathbf{8.88\times10^{-16}}$	10.68
F_9	CSA	5.65×10^{-1}	4.37×10^{-1}	1.72	1.33×10^{-4}	3.12×10^{-4}	6.15	2.79×10^{-3}	6.19×10^{-3}	10.63
	RCSA	**0**	**0**	1.89	**0**	**0**	6.04	**0**	**0**	11.00
	MCSA	**0**	1.6×10^{-311}	2.00	4.05×10^{-211}	3.87×10^{-209}	6.29	1.4×10^{-266}	9.5×10^{-264}	11.52
	SCSA	**0**	**0**	1.91	**0**	**0**	6.22	**0**	**0**	10.99
	RMSCSA	**0**	**0**	2.00	**0**	**0**	6.49	**0**	**0**	10.71
F_{10}	CSA	1.90×10^{-11}	8.75×10^{-11}	1.97	9.53×10^{-8}	3.60×10^{-8}	6.56	3.70×10^{-4}	2.01×10^{-3}	11.26
	RCSA	1.51×10^{-8}	1.44×10^{-7}	2.10	$\mathbf{1.12\times10^{-10}}$	1.74×10^{-8}	7.06	6.95×10^{-8}	1.17×10^{-8}	12.28
	MCSA	4.21×10^{-23}	2.20×10^{-20}	2.30	2.99×10^{-8}	5.83×10^{-8}	6.89	4.99×10^{-8}	1.66×10^{-9}	12.09
	SCSA	3.85×10^{-10}	7.06×10^{-8}	2.03	2.56×10^{-8}	1.31×10^{-8}	6.66	4.27×10^{-5}	4.98×10^{-4}	11.76
	RMSCSA	$\mathbf{3.51\times10^{-26}}$	$\mathbf{1.18\times10^{-24}}$	2.51	1.57×10^{-10}	$\mathbf{1.18\times10^{-8}}$	7.05	$\mathbf{7.55\times10^{-13}}$	$\mathbf{1.43\times10^{-12}}$	11.81

表5 固定维度下的优化对比

Table 5 Optimization comparison in fixed dimensions

函数	算法名称	平均值	标准差	耗时/s	维度
F_{11}	CSA	9.979×10^{-1}	2.23×10^{-4}	1.28	2
	RCSA	9.980×10^{-1}	5.90×10^{-11}	1.45	2
	MCSA	9.981×10^{-1}	6.20×10^{-2}	1.38	2
	SCSA	9.980×10^{-1}	2.55×10^{-8}	1.45	2
	RMSCSA	$\mathbf{9.982\times10^{-1}}$	$\mathbf{4.38\times10^{-16}}$	1.55	2
F_{12}	CSA	5.110×10^{-4}	4.99×10^{-3}	1.27	4
	RCSA	3.117×10^{-4}	5.53×10^{-3}	1.36	4
	MCSA	3.083×10^{-4}	6.58×10^{-3}	1.92	4
	SCSA	3.100×10^{-4}	2.87×10^{-4}	1.37	4
	RMSCSA	$\mathbf{3.079\times10^{-4}}$	$\mathbf{2.38\times10^{-6}}$	1.81	4
F_{13}	CSA	3.974×10^{-1}	1.55×10^{-2}	0.22	2
	RCSA	$\mathbf{3.978\times10^{-1}}$	1.20	0.21	2
	MCSA	3.986×10^{-1}	8.21×10^{-2}	0.29	2
	SCSA	$\mathbf{3.978\times10^{-1}}$	3.62×10^{-5}	0.24	2
	RMSCSA	$\mathbf{3.978\times10^{-1}}$	$\mathbf{2.98}\times10^{-6}$	0.36	2
F_{14}	CSA	−3.322	3.01×10^{-2}	1.37	6
	RCSA	−3.324	$\mathbf{5.80\times10^{-9}}$	1.54	6
	MCSA	**−3.320**	1.99×10^{-1}	1.79	6
	SCSA	−3.322	6.75×10^{-2}	1.97	6
	RMSCSA	**−3.320**	7.25×10^{-8}	1.94	6

由表4的实验结果，从耗时上看，不同维度下所需的时间有着很大的差异。对于10维搜索空间而言，单次独立运行的时间集中在1.65~2.3 s；随着搜索维度的增加，在30维搜索空间上，运行时间的差异变得明显，主要集中在6~8 s；50维函数下，运行时间普遍超过10 s。由此，维度的不同对算法的运行时间存在很大的差异，算法运算量也会随维度呈指数增加使求解变得困难。由表4看出，耗时最少的是CSA，主要原因之一是它没有增加策略，复杂度相对较小。综上，从不同维度函数下，CSA的计算时间略低于其他算法，但总体差异不大，时间差异在可接受范围内。

对于平均值和标准差，不同维度的平均值存在差异。其主要方面也是随着搜索维度的增加，求解更加困难，复杂度提高导致搜索效率下降。但从表4可看出，随着维度逐渐增大，算法收敛精度略有降低，但相比于CSA来说并没有大幅度降低。函数F_1~F_4下，RMSCSA相比于其他几种算法具有很好的性能和方差。另外，对于函数F_7~F_9以及F_1、F_4而言，在不同维度的搜索都具有较为理想的寻优能力，且随着复杂度增大没有影响搜索的性能；对CSA而言，大部分随着维度的增加精度逐渐降低甚至和目标优化值偏离程度严重。文献[10]中CSA在低维空间下有很好的寻优性能但忽视了维度增大的考验。本文的结果在低维寻优上与对应文献相差无几，但随维度增大后，CSA的缺陷渐渐出现。综上分析，表中的CSA在标准差和平均值与其它改进策略的算法对比有效性和寻优能力都较差，标准差较大，算法求解不稳定。

根据表5固定维函数上看，CSA的性能相比其它算法优势不明显；对于其他算法，虽然都与目标函数值存在差异，但总体离目标值偏离程度较小，只有函数F_{14}的标准差劣于RCSA，但差异较小。且本文F_{11}~F_{14}函数虽然维度低，但隐含各种参数使低维函数的求解也较为复杂，而引入RMSCSA相比求解稳定性较好，说明RMSCSA在低维求解困难函数上具有较好的寻优能力。综上所述，本文提出的RMSCSA和引入改进策略的算法都具备良好寻优能力，且从单峰、多峰、固定维及不同维度下都相比CSA算法提升显著，算法有效性得以保证。

图4绘制出部分不同维度和不同函数的平均收敛曲线，进一步能从图形上直观看出各种算法策略的迭代收敛情况。便于直观观察，纵坐标显示适应度值取10为底的对数，横坐标以200为间隔取值。其中，画出10、30、50维函数图形各3个。

由图4，RMSCSA相比与其他算法而言有更快的平均收敛速度。从迭代初期可看出SCSA和RMSCSA算法就渐渐体现较快的搜索性能，反映出本文提出的S型惯性权重因子发挥较好的搜索优势；随着迭代进行，各算法搜索效率逐渐拉开，层次更清晰，说明其他策略的引入也变得有效，同时表明引入的不同策略对算法的寻优能力有不同程度的提高。

DOI: 10.16182/j.issn1004731x.joss.21-0968

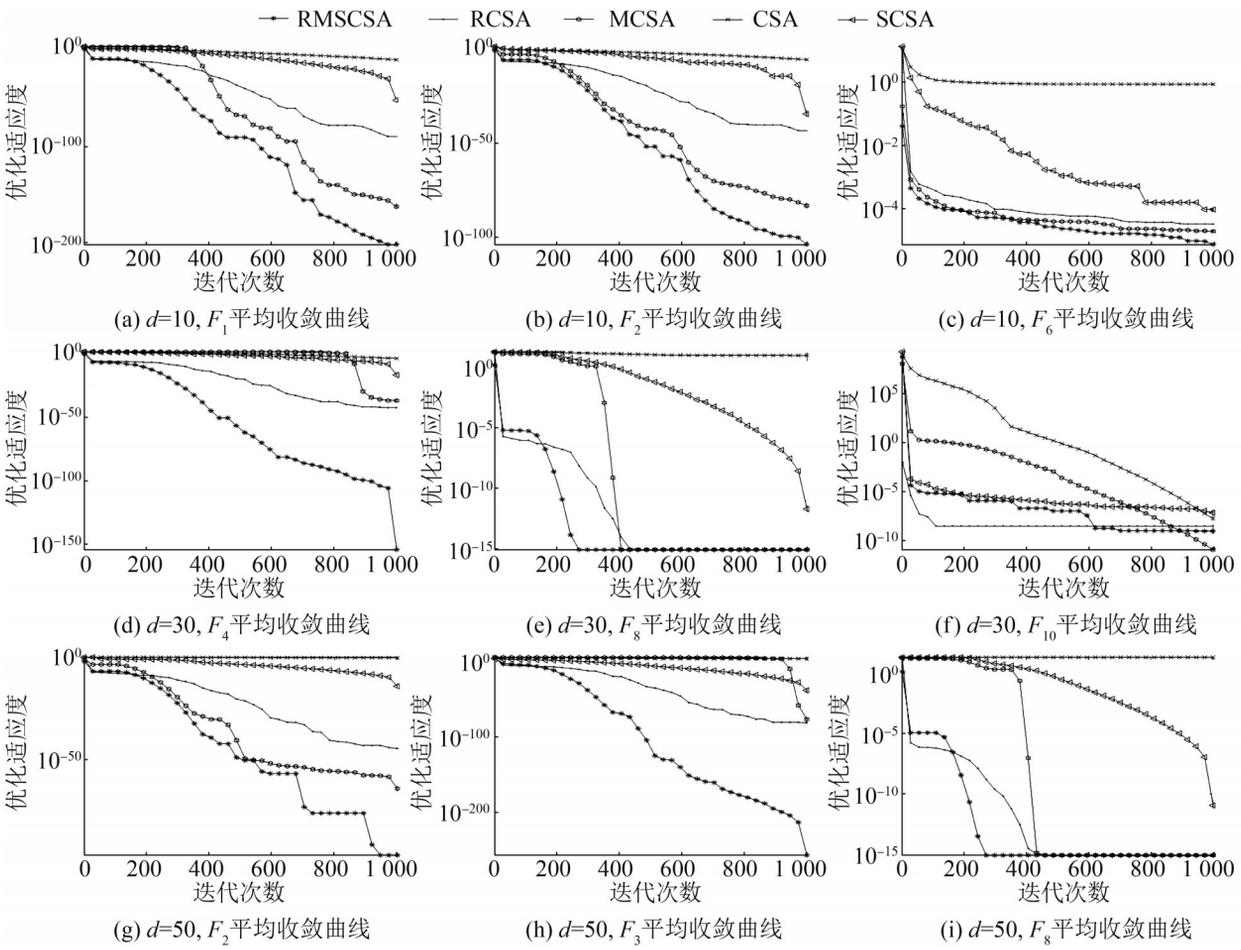

图4　不同维度下测试函数下的平均收敛曲线

Fig. 4　Average convergence curves of test functions in different dimensions

实验2：为进一步体现本文所提RMSCSA的优越性，实验2引入IEEE CEC 2017竞赛集验证算法性能。根据不同函数类型均匀选取10个函数进行对比，包括单峰、多峰、混合(Hybrid)和复合(Composition)竞赛函数。另外，还引入近些年国内外研究较多的樽海鞘群算法(SSA)[7]和鲸鱼优化算法(WOA)[8]进行对比，竞赛函数的具体信息见表6。

表7记录竞赛集函数对比的优化结果，包括各种算法策略的平均值、标准差及耗时。由于IEEE函数竞赛集组合、嵌入各类函数的复杂特征导致目标函数的求解十分困难，对这类函数的求解难以找到目标最优值，只能与其他算法对比找到相对最优值，得出算法在困难函数上的寻优性能。先从耗时上看，耗时最短的是WOA和SSA算法；这2种算法相比于CSA复杂度低，算法执行的速度较快。但从每个函数优化的平均值和标准差的对比，整体劣于CSA和本文改进策略的效果；因此，虽然算法的执行时间快，但求解的性能和稳定性较差。

表6　CEC 2017测试集函数信息

Table 6　CEC 2017 test set function information

函数名	维度	取值空间	特征	优化最优值
CEC04	30	[−100, 100]	MS	400
CEC05	30	[−100, 100]	MN	500
CEC08	30	[−100, 100]	MN	800
CEC16	30	[−100, 100]	HF	1 600
CEC17	30	[−100, 100]	HF	1 700
CEC20	30	[−100, 100]	HF	2 000
CEC21	30	[−100, 100]	CF	2 100
CEC23	30	[−100, 100]	CF	2 300
CEC27	30	[−100, 100]	CF	2 700
CEC29	30	[−100, 100]	CF	2 900

表 7 CEC 2017 测试下各种算法性能对比

Table 7 Performance comparison of various algorithms under CEC 2017 test

函数名	评价指标	RMSCSA	CSA	RCSA	MCSA	SCSA	SSA	WOA
CEC04	平均值	**4.705 2×10²**	2.120 8×10³	7.353 7×10²	4.724 4×10²	7.489 6×10²	6.425 5×10²	6.890 7×10²
	标准差	7.298×10⁻⁴	1.628×10	**1.587**×10⁻⁴	1.698×10	1.093×10²	2.055×10	6.773×10
	耗时/s	4.08	6.32	2.69	3.13	2.01	1.25	0.83
CEC05	平均值	**5.209 7×10²**	1.006 5×10	6.885 6×10²	7.511 1×10²	5.385 6×10²	6.970 5×10²	8.454 6×10²
	标准差	**1.625×10⁻⁵**	2.013×10	1.526×10	2.296×10	2.160×10	3.852×10	5.344×10
	耗时/s	3.88	6.44	2.67	3.14	2.01	1.61	0.75
CEC08	平均值	**9.235 5×10²**	1.237 0×10³	1.237 0×10³	1.005 9×10³	1.037 9×10³	9.958 0×10²	1.129 4×10³
	标准差	**2.684×10⁻⁵**	6.200×10⁻⁴	1.954×10⁻³	2.192×10	1.573×10	4.910 2×10	6.337×10
	耗时/s	3.91	2.91	2.71	4.19	2.26	1.53	0.88
CEC16	平均值	**3.258 6×10³**	6.706 5×10³	3.749 8×10³	4.503 9×10³	3.450 6×10³	1.855 9×10⁴	4.921 2×10³
	标准差	**3.105×10²**	2.176×10³	2.112×10³	3.051×10²	3.220×10²	3.771 5×10³	4.472×10²
	耗时/s	3.96	3.73	2.66	4.26	2.84	1.32	0.77
CEC17	平均值	2.356 9×10³	2.499 6×10⁴	4.446 5×10³	3.004 0×10³	**2.282 3×10³**	2.966 2×10³	2.986 8×10³
	标准差	**3.144×10**	3.334×10⁴	2.781×10²	8.944×10	1.395×10²	2.836×10²	2.991×10²
	耗时/s	4.11	3.96	2.78	4.37	2.93	1.55	0.56
CEC20	平均值	**2.630 2×10³**	3.646 7×10³	3.410 6×10³	3.012 0×10³	2.758 8×10³	2.795 5×10³	3.127 9×10³
	标准差	**1.361×10**	1.592×10²	9.995×10	7.638×10	2.135×10²	5.541×10³	1.823×10²
	耗时/s	4.14	3.85	2.81	4.40	2.90	1.97	0.57
CEC21	平均值	2.610 0×10³	2.926 4×10³	2.849 0×10³	2.604 0×10³	**2.526 2×10³**	2.917 2×10³	2.831 9×10³
	标准差	**6.886**	5.267×10	3.944×10	2.114×10	1.941×10	5.541 0×10²	3.544×10
	耗时/s	4.26	3.82	3.23	4.38	3.07	1.97	0.42
CEC23	平均值	**3.211 7×10³**	4.012 8×10³	3.781 0×10³	3.257 7×10³	3.893 0×10³	3.698 8×10³	3.987 5×10³
	标准差	4.617×10	2.371×10²	1.298×10²	**4.601×10**	2.389×10²	4.629×10³	1.256×10²
	耗时/s	2.95	3.76	3.16	3.45	3.36	1.45	0.95
CEC27	平均值	**3.060 0×10³**	5.167 4×10³	3.508 0×10³	5.611 0×10³	3.302 8×10³	4.305 7×10³	4.558 1×10³
	标准差	**2.015×10**	3.219×10³	1.143×10²	5.111×10²	3.083×10²	2.609×10	8.075×10
	耗时/s	4.06	4.13	4.46	3.17	4.35	1.30	0.56
CEC29	平均值	**3.652 9×10³**	2.614 1×10⁴	8.485 1×10³	8.485 8×10³	4.419 8×10³	4.202 8×10³	5.034 0×10³
	标准差	**3.505**	2.067×10⁴	5.081×10	5.081×10³	2.094×10²	2.024×10²	5.363×10²
	耗时/s	5.70	4.09	3.07	3.07	4.12	1.28	0.42

再从平均值和标准差上对比，RMSCSA 较 MCSA 和 SCSA 略好，且 RMSCSA 为未最优解的频率较稳定。RMSCSA 中部分函数的性能虽劣于其它算法，总体寻优效果好。从平均值分析，只有 CEC17 和 21 劣于其他算法，其余 8 个函数上求解效果最好，可看出具有十分明显的优势。总体上看，除 RMSCSA 外，求解结果相对较好的是 SCSA、RCSA，求解性能最差为 CSA 和 SSA。但仅从平均值考虑是不充分的，而需要把标准差因素考虑其中衡量平均值的总体偏离程度。

由表 7 可看出，RMSCSA 在 CEC04 和 CEC23 函数上的平均值最优，但标准差劣于 RCSA 和 MCSA，这 2 个函数的标准差从数值上看差异不大，处于同一数量级且数值接近，因此也无法拒绝标准差的差异一定劣于 RCSA 和 MCSA，从而也可证明总体而言 RMSCSA 标准差也趋于稳定且标准差最优的个数最多。综上，本文 RMSCSA 算法对于 CEC 竞赛集函数具有良好的寻优效果。

DOI: 10.16182/j.issn1004731x.joss.21-0968

3.3 与近年其他智能算法对比

近年来，众多研究者针对智能优化算法也有不少研究成果，且都有良好的搜索精度和较快的收敛速度。为了比较本文的RMSCSA与近3年较优秀的研究成果，对比RMSCSA和其他算法的优劣，引入多子群进化算法(MPEA)[20]、麻雀搜索算法(SSA)[23]、混合合作多目标优化的入侵杂草优化(HCMOIWO)[24]、柯西变异和均匀分布的蝗虫算法(HCUGOA)[11]、随机策略和反向学习自适应差分进化算法(GOBL-RNADE)[12]、全局收敛的鲸鱼优化算法(WOAEP)[25]、分数阶微积分的花粉算法(FO-FPA)[26]、自适应杂交差分进化裸鼹鼠算法(SHDNMRA)[14]和正弦余弦的乌鸦搜索算法(SCCSA)[27]等。为保持与其他算法比较一致，设置种群的大小为30，维度为30，最大迭代次数为1 000，独立运行30次。本文仅比较平均值，对比结果如表8所示。

由表8，对比算法的在同一函数的搜索精度较高，本文提出的RMSCSA也不逊于对比算法，可看出本文算法求解结果对比也相对较为显著。虽然RMSCSA不能保证每个函数都最优，但大部分都仅次于最优，排名靠前。综上所述，本文提出的RMSCSA和近3年其他优秀成果的对比也具备良好的搜索性能和收敛精度。

4 统计学检验分析

3.2节中对经典测试集和CEC 2017分析只是从平均值和标准差的视野去直观看待算法的性能。分析结果上看，部分算法的对比出现性能相当、优于和劣于，这都是偶然因素。平均值最优，但标准差存在差异，难以说明性能一定最好。针对这一分析不足，本文对独立重复的数据统计学分析，从统计学视野评估算法的性能是客观、公正的。

各算法在不同测试函数下独立运行30次的数据利用Wilcoxon秩和检验[28]进行统计，该方法能描述算法与其他方法的显著性。文中对经典函数集进行秩和检验，显著性水平为$\alpha=0.05$。若检验结果$p<0.05$则说明算法存在显著性差异，否则差异不明显。从表4中的结果上看，默认控制算法为RMSCSA，对比算法分别为RMSCSA和RCSA、RMSCSA和MCSA、RMSCSA和SCSA、RMSCSA和CSA，分别在表9中以p_1、p_2、p_3和p_4表示。由于F_1~F_{10}在不同维度下独立测试，因此表中显示的是不同维度p的平均值。文中“NA”字样表示对比无效，说明算法性能相当，“+/–/=”分别表示“优/劣/相当”。文中p值大于0.05或性能相当已用粗体标记。

表8 与其他智能算法的平均值比较

Table 8 Average comparison with other intelligent algorithms

算法	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}
RMSCSA	**0**	2.21×10^{-179}	**0**	$\mathbf{2.77\times10^{-181}}$	$\mathbf{1.92\times10^{-5}}$	$\mathbf{2.39\times10^{-6}}$	**0**	8.88×10^{-16}	**0**	5.52×10^{-9}
MPEA[20]	2.70×10^{-11}	2.77×10^{-91}	1.52×10^{-20}	1.04×10^{-68}	6.74×10^{-12}	2.35×10^{-1}	6.38×10^{-6}	2.84×10^{-13}	4.3×10^{-7}	2.27×10^{-5}
SSA[21]	**0**	2.12×10^{-259}	**0**	5.72×10^{-177}	9.24×10^{-4}	1.34×10^{-4}	**0**	8.88×10^{-16}	**0**	7.17×10^{-12}
HCMOIWO[22]	3.17×10^{-55}	3.66×10^{-51}	5.41×10^{-10}	9.34×10^{-13}	5.28×10^{-1}	8.41×10^{-3}	4.62×10^{-1}	1.86×10^{-11}	3.29×10^{-2}	1.35×10^{-3}
HCUGOA[23]	**0**	**0**	**0**	3.23×10^{-176}	1.78×10^{-3}	4.96×10^{-5}	**0**	8.88×10^{-16}	**0**	
GOBL-RNADE[24]	1.63×10^{-240}	1.80×10^{-126}	7.15×10^{-276}	6.26×10^{-115}	5.94×10^{-2}	7.70×10^{-2}	**0**	**0**	**0**	$\mathbf{2.28\times10^{-12}}$
WOAEP[25]	1.59×10^{-96}	1.27×10^{-48}	3.97×10^{-96}	1.46×10^{-49}	2.89×10	2.67×10^{-4}	**0**	8.88×10^{-16}	**0**	2.99
FO-FPA[26]	1.51×10^{-184}	5.04×10^{-93}	1.23×10^{-183}	9.97×10^{-93}	2.89×10	1.13×10^{-4}	**0**	8.88×10^{-16}	**0**	9.65×10^{-6}
SHDNMRA[18]	1.80×10^{-60}	1.80×10^{-25}	5.70×10^{-61}	8.20×10^{-24}	8.00×10^{-2}	6.40×10^{-4}	8.20×10^{-10}	8.20×10^{-10}	4.60×10^{-17}	3.02×10^{-7}
SCCSA[27]	9.22×10^{-69}	8.24×10^{-41}	4.31×10^{-13}	2.15×10^{-17}	5.90	1.33×10^{-3}	5.46	8.88×10^{-16}	1.34×10^{-2}	3.00

表9 经典测试函数下的秩和检验值
Table 9 Rank sum test values of classical test functions

函数	p_1	p_2	p_3	p_4
F_1	3.00×10^{-11}	3.01×10^{-11}	3.00×10^{-11}	3.00×10^{-11}
F_2	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
F_3	2.99×10^{-11}	2.99×10^{-11}	2.99×10^{-11}	2.99×10^{-11}
F_4	3.02×10^{-11}	3.00×10^{-11}	3.02×10^{-11}	3.00×10^{-11}
F_5	2.14×10^{-9}	4.04×10^{-11}	2.06×10^{-10}	1.16×10^{-10}
F_6	**0.282**	**0.365**	5.66×10^{-11}	3.02×10^{-11}
F_7	**NA**	**NA**	**NA**	1.21×10^{-12}
F_8	5.28×10^{-5}	3.99×10^{-3}	**NA**	1.21×10^{-12}
F_9	**NA**	3.68×10^{-3}	**NA**	1.21×10^{-12}
F_{10}	6.84×10^{-7}	3.02×10^{-11}	1.17×10^{-9}	1.21×10^{-12}
F_{11}	**0.4247**	3.02×10^{-11}	1.17×10^{-9}	1.21×10^{-12}
F_{12}	3.02×10^{-11}	1.49×10^{-4}	8.15×10^{-11}	8.48×10^{-9}
F_{13}	**0.595**	1.23×10^{-9}	2.78×10^{-11}	1.09×10^{-12}
F_{14}	7.70×10^{-8}	5.57×10^{-10}	3.02×10^{-11}	3.02×10^{-11}
+/–/=	9/3/2	12/1/1	11/0/3	14/0/0

由表9中Wilcoxon秩和检验结果，大部分与RMSCSA算法计算出的p值都要小于0.05；在F_7函数、F_8和F_9函数中有“NA”字样，即RMSCSA和RCSA、MCSA和SCSA性能相当，说明上述策略对2个函数求解十分敏感。RMSCSA对比RCSA，函数F_6、F_{11}和F_{13}的RMSCSA的性能劣于RCSA，说明RML策略的有效性比RMSCSA好，其他均与RMSCSA存在显著性差异。因此，根据Wilcoxon检验分析的结果，RMSCSA具有统计学上的有效性。

上述讨论为函数对比分析，为定量分析，下面用Friedman’s检验[29]可对不同策略进行定性分析，判决每种策略排名情况。该方法需先计算各算法在每个测试函数上的平均排名，计算出所有统计数据卡方值[30]与临界值进行比较，得出的临界值进行比较是否拒绝假设的依据。最后用Holm后续矫正检验[31]分析组间差异性。

各算法的平均排名计算公式为

$$R_j=\frac{1}{N}\sum_{i=1}^{N}r_i^j \tag{28}$$

式中：R_j为算法j的平均排名；N为函数个数；r_i^j为函数i中算法j的排名。实验1 F_1~F_{10}为不同维度，所以需要取3个测试的平均值。表10、11分别为实验1和实验2算法的平均排名，表12、13分别为Holm后续检验的结果。

表10 经典测试函数下算法平均排名
Table 10 Average ranking of algorithms under classical test functions

算法	R_1	R_2	R_3	$\bar{R}$	排名
RMSCSA	2.35	2.2	2.05	2.2	1
RCSA	4.7	2.2	2.45	3.116 666 7	3
MCSA	2.35	2.5	2.25	2.366 666 7	2
SCSA	2.85	3.3	3.35	3.166 666 7	4
CSA	2.75	4.8	4.9	4.15	5

表11 CEC 2017函数平均排名
Table 11 Average ranking of CEC 2017 functions

算法	R	排名
RMSCSA	1.8	1
CSA	6.75	7
RCSA	4.55	5
MCSA	3.9	4
SCSA	2.5	2
SSA	3.7	3
WOA	4.8	6

表12 经典函数Holm后续校验结果
Table 12 Subsequent verification results of classical function Holm

i	RMSCSA vs.	$\bar{z}$值	$\bar{p}$值	α/i	是否拒绝
4	CSA	3.677 0	2.36×10^{-4}	0.0125	是
3	SCSA	2.877 5	4.01×10^{-3}	0.016 666 7	是
2	RCSA	2.388 7	1.69×10^{-2}	0.025	是
1	MCSA	0.893 1	0.3725	0.05	否

表13 CEC 2017函数Holm后续校验结果
Table 13 CEC 2017 Holm subsequent verification results

i	RMSCSA vs.	z值	p值	α/i	是否拒绝
6	CSA	5.124	2.99×10^{-7}	0.008 333	是
5	WOA	3.105	1.903×10^{-3}	0.01	是
4	RCSA	2.847	4.413×10^{-3}	0.025	是
3	MCSA	2.617 4	8.86×10^{-3}	0.016 67	是
2	SSA	2.359 4	1.83×10^{-2}	0.025	是
1	SCSA	0.725	0.468 452	0.05	否

由表10，RMSCSA的综合排名为1，算法有效性最好。表12中经典函数对比的算法自由度为

DOI: 10.16182/j.issn1004731x.joss.21-0968

4，查表得临界卡方值为9.487 729，统计量的平均卡方值为27.615 7，p值为1.492 2×10^{-5}，远小于0.05的临界显著性水平，存在统计学差异；只有MCSA接受原假设，说明MCSA与RMSCSA的总体差异不明显，即MCSA策略的有效性更强。经过Holm后续矫正，性能排名RMSCSA>MCSA>RCSA>SCSA>CSA。由此，利用统计学方法通过矫正得出各种算法差异的程度，从而得出各算法有效性排名。

同样，表11中排名最好的是RMSCSA，其性能结果最好的为RMSCSA。同样将它作为控制算法进行Holm后续校验，校验结果如表13。

CEC 2017函数集中，自由度为6，查表可得临界卡方值为：12.591 6，统计量卡方值31.846；由卡方值求出的p值为1.746 5×10^{-5}，远小于显著性水平的p值，整体出现显著性差异。由表13的检验矫正结果，在CEC 2017测试集上除了SCSA以外均拒绝原假设，则说明只有SCSA的差异不明显，SCSA的性能接近RMSCSA。根据p值判定，SCSA性能接近RMSCSA，SSA次之，最差为CSA。从统计学视野分析实验1和实验2的有效性和性能对比，表明本文提出RMSCSA的有效性、可靠性及性能都是最优的。

5 结论

本文针对变色龙群算法求解精度低和算法的搜索和开发不平衡进行优化改进，在算法复杂度未明显提高情况下增强CSA的寻优能力，搜索和开发能力也得到更好的平衡，提升算法的稳定性和寻优能力。引入RML获取算法的动态解，增加算法跳出局部最优的概率；引入多样性分段变异策略对不同适应度区间的个体分段变异提高算法的搜索效率；引入S型惯性权重平衡算法前期的搜索和后期开发，提高算法全局寻优能力。最后，利用统计学差异分析论证算法在2个实验上的有效性和性能，以统计学的视野证实RMSCSA的性能的优越性。后续研究中，将它应用在认知异构通信网络的资源分配优化问题和各种NP难题，进一步探索算法在复杂优化问题求解的有效性和可靠性。

参考文献：

[1] Pramada S K, Archana S O, Sithara S, et al. Grid Search Based SVM Approach for Sea Level Rise[J]. IOP Conference Series: Earth and Environmental Science (S1755-1307), 2020, 581(1): 1-18.

[2] Ningsih S, Syahputra D. K-Means Algorithm for Clustering Third-party Funds of Conventional Banking [J]. Journal of Physics: Conference Series (S1742-6588), 2021, 1899(1): 18-36.

[3] Kennedy J, Eberhart R. Particle Swarm Optimization. [C]// International Conference on Neural Net-works. Perth, Australia: IEEE,1995: 1942-1948.

[4] Gold D E, Holland J H. Genetic Algorithms and Machine Learning[J]. Machine Learning (S0885-6125), 1988, 3(2/3): 95-99.

[5] Rainer S, Kenneth P. Differential Evolution: A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces[J]. Journal of Global Optimization (S0925-5001), 1997, 11(4): 341-359.

[6] Dorigo M, Maniezzo V, Colorni A. Ant System: Optimization by a Colony of Cooperating Agents[J]. IEEE Trans on Systems, Man, and Cybernetics. Part B, Cybernetics: a Publication of the IEEE Systems, Man, and Cybernetics Society (S1083-4427), 1996, 26(1): 29-41.

[7] 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 (S0965-9978), 2017, 114(6): 163-191.

[8] Mirjalili S, Andrew L. The Whale Optimization Algorithm [J]. Advances in Engineering Software (S0965-9978), 2016, 95: 51-67.

[9] Konstantinos Z, Stelios T. A Mayfly Optimization Algorithm[J]. Computers & Industrial Engineering (S0360-8352), 2020, 145(3): 1-26.

[10] Shehadeh B M. Chameleon Swarm Algorithm: A Bio-inspired Optimizer for Solving Engineering Design Problems[J]. Expert Systems with Applications (S0957-4174), 2021, 174: 1-24.

[11] 何庆, 林杰, 徐航. 混合柯西变异和均匀分布的蝗虫优化算法[J]. 控制与决策, 2021, 36(7): 1558-1568.
He Qing, Lin Jie, Xu Hang. Hybrid Cauchy Mutation and Uniform Distribution of Grasshopper Optimization

Algorithm[J]. Control & Decision, 2021, 36(7): 1558-1568.

[12] 吴文海, 郭晓峰, 周思羽, 等. 基于随机邻域策略和广义反向学习的自适应差分进化算法[J].系统工程与电子技术, 2021, 43(7): 1928-1942.
Wu Wenhai, Guo Xiaofeng, Zhou Siyu, et al. Adaptive Differential Evolution Algorithm Based on Stochastic Neighborhood Strategy and Generalized Reverse Learning[J]. Systems Engineering and Electronics, 2021, 43(7): 1928-1942.

[13] 范千, 陈振健, 夏樟华. 一种基于折射反向学习机制与自适应控制因子的改进樽海鞘群算法[J]. 哈尔滨工业大学学报, 2020, 52(10): 183-191.
Fan Qian, Chen Zhenjian, Xia Zhanghua. An Improved Salp Group Algorithm Based on Refraction Reverse Learning Mechanism and Adaptive Control Factor[J]. Journal of Harbin Institute of Technology, 2020, 52(10): 183-191.

[14] Roshan E S, Habil M K. Refraction and Reflection from the Interface of Anisotropic Materials[J]. Physical Scripta (S0031-8949), 2019, 94(8): 17-29.

[15] Karpis G, Han E H, Kumar V. Chameleon: Hierarchical Clustering using Dynamic Modeling (Cover Story) [J]. Computer (S0018-9162), 1999, 32(8): 68-75.

[16] Li Jingnan, Le Meilong. Improved Whale Optimization Algorithm based on Image Selection[J]. Trans of Nanjing University of Aeronautics and Astronautics (S1005-1120), 2020, 37(S1): 115-123.

[17] Deb K, Deb D. Analysing Mutation Schemes for Real-Parameter Genetic Algorithms[J]. Int. J. of Artificial Intelligence and Soft Computing (S1755-4950), 2014, 4 (1): 1-28.

[18] Shi Yuhui, Eberhart R C. A Modified Particle Swarm Optimizer[C]// IEEE Int Conf on Evolutionary Computation, 1998: 69-73.

[19] Pitono J, Soeprijanto A. Hybrid Optimization of Emission and Economic Dispatch by the Sigmoid Decreasing Inertia Weight Particle Swarm Optimization [J]. World Academy of ence Engineering & Technology (S0950-4125), 2009, 60: 315–320.

[20] Salgotra R, Singh U, Singh G, et al. A Self-Adaptive Hybridized Differential Evolution Naked Mole-Rat Algorithm for Engineering Optimization Problems[J]. Computer Methods in Applied Mechanics and Engineering (S0045-7825), 2021: 383-409.

[21] 吴凡, 洪思, 杨冰, 等. 曲线递增策略的自适应粒子群算法研究[J].计算机应用研究, 2021, 38(6): 1653-1656, 1661.
Wu Fan, Hong Si, Yang Bing, et al. Adaptive Particle Swarm Optimization based on Curve Increasing Strategy [J]. Application Research of Computers, 201,38(6):1653-1656, 1661.

[22] Liu H T, Du W, Guo Z X. A Multi-Population Evolutionary Algorithm with Single-Objective Guide for Many-objective Optimization[J]. Information Sciences (S0020-0255), 2019, 503: 39-60.

[23] Xue J K, Shen B. A Novel Swarm Intelligence Optimization Approach: Sparrow Search Algorithm[J]. Systems Science & Control Engineering (S2164-2583), 2020, 8(1): 22-34.

[24] Technology-Cybernetics. Solving Multi-objective Optimization Problems Using Hybrid Cooperative Invasive Weed Optimization With Multiple Populations [J]. Journal of Technology & Science (S1944-1894), 2018: 821-832.

[25] 冯文涛, 邓兵. 鲸鱼优化算法的全局收敛性分析及参数选择研究[J]. 控制理论与应用, 2021, 38(5): 641-651.
Feng Wentao, Deng Bing. Global Convergence Analysis and Parameter Selection of Whale Optimization Algorithm[J]. Control Theory and Applications, 2021, 38 (5): 641-651.

[26] Dalia Y, Mohamed A E, Mirjalili S. Fractional-order Calculus-based Flower Pollination Algorithm with Local Search for Global Optimization and Image Segmentation [J]. Knowledge-based Systems (S0950-7051), 2020, 197 (4): 105889.

[27] Soheyl K, Seyed H R. Sine-cosine Crow Search Algorithm: Theory and Applications[J]. Neural Computing and Applications (S0941-0643), 2019, 3(6): 1-18.

[28] Wilcoxon F. Individual Comparisons by Ranking Methods[J]. Biometrics Bulletin (S0099-4987), 1945, 1(6): 80-83.

[29] Kennet M C, Chokumnoyporn N, Sriwattana S, et al. Comparing Friedman Versus Mack-Skillings Data Analyses on duplicated rank data: a Case of Visual Color Intensity[J]. Journal of the Science of Food and Agriculture (S0022-5142), 2019, 99(13): 5696-5701.

[30] Qin H W, Fei Q H, Ma X Q, et al. A New Parameter Reduction Algorithm for Soft Sets Based on Chi-square Test[J]. Applied Intelligence (S0924-669X), 2021, 51 (11) : 7960-7972.

[31] Shih-Hsi L, Marjan M, Dejan H, et al. A Parameter Control Method of Evolutionary Algorithms using Exploration and Exploitation Measures with a Practical Application for fitting Sovova's Mass Transfer Model[J]. Applied Soft Computing Journal (S1568-4946), 2013, 13 (9): 3792-3805.