DOI:10.16136/j.joel.2023.10.0397

基于高斯云改进的混沌麻雀搜索算法与应用

顾嘉城,龙英文*,吉明明,郑 旸

(上海工程技术大学 电子电气工程学院,上海 201620)

摘要:传统的麻雀搜索算法(sparrow search algorithm,SSA)在寻优过程中存在易陷入局部最优,以及搜索能力不足的问题。为了解决上述问题,提出了一种基于高斯云改进的混沌改进麻雀搜索算法(improved sparrow search algorithm,ISSA)。首先,利用伯努利混沌映射初始化种群以提高算法初始种群质量;其次,在发现者位置更新中引入自适应高斯云变异策略来提高算法在迭代过程中的全局搜索能力;最后,利用 t 分布反向学习策略对最优位置进行随机反向学习,以提高算法跳出局部最优的能力。在仿真实验中将本算法与其他 4 种基本算法利用 13 种基准测试函数进行对比实验,同时与其他的 ISSAs 进行对比。实验结果表明,本算法具有良好的收敛性以及精度,且全局探索能力相较于原算法大大提高。并将 ISSA 应用于 Kapur 熵多阈值图像分割任务中,结果表明,ISSA 相较于其他 4 种基本算法有着更高的分割精度。

关键词:麻雀搜索算法(SSA);群智能优化算法;自适应高斯云;混沌映射;图像分割
中图分类号:TP391 **文献标识码**:A **文章编号**:1005-0086(2023)10-1047-12

Improved chaotic sparrow search algorithm and application based on Gaussian cloud

GU Jiacheng, LONG Yingwen*, JI Mingming, ZHENG Yang

(School of Electronic and Electrical Engineering, Shanghai University of Engineering Science, Shanghai 201620, China)

Abstract: The traditional sparrow search algorithm (SSA) has the problems that it is easy to fall into the local optimum and the search ability is insufficient in the process of optimization. In order to solve the above problems, an improved sparrow search algorithm (ISSA) based on Gaussian cloud improvement is proposed. First, Bernoulli chaotic mapping is used to initialize the population to improve the initial population quality of the algorithm; secondly, an adaptive Gaussian cloud mutation strategy is introduced in the update of the finder position to improve the global search ability of the algorithm in the iterative process; finally, the reverse t distribution learning strategy is used to perform random reverse learning on the optimal position to improve the algorithm's ability to jump out of the local optimum. In the simulation experiment, this algorithm is compared with other four basic algorithms with 13 benchmark functions, and compared with other ISSAs. The experimental results show that the algorithm has good convergence and accuracy, and the global exploration ability is greatly improved compared with the original algorithm. The ISSA is applied to the Kapur entropy multi-threshold image segmentation task, and the results show that ISSA has higher segmentation accuracy than the other four basic algorithms.

Key words: sparrow search algorithm (SSA); swarm intelligence optimization algorithm; adaptive Gaussian cloud; chaotic mapping; image segmentation

0 引 言

最优化问题是工程应用中最基本的问题。解决最优化问题指的是在满足一定条件的基础下,通过计算求得最佳的结果的过程。传统的优化方法,例如梯度下降法等虽然简单,但难以解决多峰

* E-mail:gu_jiacheng798@163.com
收稿日期:2022-05-27 修订日期:2022-08-27
基金项目:国家自然科学基金(61603241)资助项目

函数的寻优。为解决上述等问题，大量的研究人员以自然界生物群体的行为规律为参照，提出了各种群智能优化算法，有鲸鱼优化算法（whale optimization algorithm，WOA）[1]、飞蛾火焰优化算法（moth-flame optimization algorithm，MFO）[2]、正余弦优化算法（sine cosine algorithm，SCA）[3]、灰狼优化算法（grey wolf optimization algorithm，GWO）[4]等。

研究人员在上述的算法中，通过引入诸如高斯变异[5,6]、柯西变异[7]、非线性收敛因子等[8]，对其进行优化以获得更好的寻优效果，并应用于诸多场景之上。XIONG等[9]利用WOA来进行居民用电量的预测，谷小兵等[10]利用改进的飞蛾扑火算法进行浆液循环泵组合优化，王海芳等[11]利用飞蛾扑火算法进行MRI图像分割，刘丽娟等[12]利用SCA应用于WSN节点部署优化，DUTTA等[13]利用GWO来优化电机PID控制参数。

薛建凯于2020年提出了麻雀搜索算法（sparrow search algorithm，SSA）[14]，该算法是受到自然界麻雀的觅食与反捕食的行为为启发。SSA的优点是参数设置容易以及收敛速度快。但是与其他的群智能优化算法相比，存在SSA在解决复杂问题时会提前陷入局部最优的不足。

为了弥补SSA的不足，提出了一种基于自适应高斯云变异的混沌SSA。通过实验表明，改进麻雀算法（improved sparrow search algorithm，ISSA）能够提高算法的全局寻优能力和局部开发能力，寻优性能也得到了增强。

1 基本SSA

为建立SSA的数学模型，假设共有D只麻雀组成种群群体，其在搜索空间中的位置可表示为$\boldsymbol{X}=[x_1,x_2,\cdots,x_D]^{\mathrm{T}}$，每个列向量表示每只麻雀的位置向量。那么，所有麻雀的适应度值函数$\boldsymbol{Fx}$表示为：

$$\boldsymbol{Fx}=[f(x_1),f(x_2),f(x_3),\cdots,f(x_D)]^{\mathrm{T}}\text{，}(1)$$

式中，$f(x)$代表每只麻雀的适应度值。

SSA将种群个体分为3种，分别是1）发现者、2）警戒者、3）跟随者，他们各自按照各自的分工来进行搜索。发现者提供搜索方向和搜索区域，警戒者负责监视觅食区域，跟随者负责跟随发现者进行觅食。

1）发现者大约占据种群总数的10%—20%，发现者的位置按式(2)进行更新：

$$x_{i,j}^{t+1}=\begin{cases}x_{i,j}^{t}*\exp\left(\dfrac{-i}{\alpha*iter_{\max}}\right),R_2<ST\\ x_{i,j}^{t}+\boldsymbol{Q}*\boldsymbol{L},R_2>ST\end{cases}\text{。}(2)$$

式中，$x_{i,j}^{t+1}$是种群迭代到t时第i个的个体在第j维的位置；α是一个于(0，1]中均匀分布的随机数；$iter_{\max}$是最大迭代次数；Q是服从正态分布的随机数；$\boldsymbol{L}$是一个$1\times d$的矩阵，其每一个元素皆赋值为1；ST是警戒阈值，其取值范围为[0.5，1]；R_2为警戒值，其取值范围为[0，1]。式(2)用于进行大范围的全局搜索，给麻雀种群确定大致的搜索方向。

2）警戒者的位置根据式(3)进行更新：

$$x_{i,j}^{t+1}=\begin{cases}x_{\mathrm{best}}^{t}+\beta*\left|x_{i,j}^{t}-x_{\mathrm{best}}^{t}\right|,f_i>f_{\mathrm{g}}\\ x_{i,j}^{t}+K*\dfrac{\left|x_{i,j}^{t}-x_{\mathrm{worst}}^{t}\right|}{(f_i-f_{\mathrm{w}})+\varepsilon},f_i=f_{\mathrm{g}}\end{cases}\text{。}(3)$$

式中，x_{best}^{t}为全局最优位置；β是一个随机数，其服从均值为0，方差为1的正态分布；ε是一个为了避免分母为0而设置的极小值；f_i、f_{g}、f_{w}分别第i只麻雀的适应度、当前全局最优适应度、全局最差适应度。在式(3)中，警戒者能够在靠近局部最优位置时选取一个远离当前位置的新的位置，以增强算法跳出局部最优的能力。

3）跟随者的位置根据式(4)进行更新：

$$x_{i,j}^{t+1}=\begin{cases}\boldsymbol{Q}*\exp\left(\dfrac{x_{\mathrm{worst}}^{t}-x_{i,j}^{t}}{i^2}\right),i>\dfrac{n}{2}\\ x_p^{t+1}+\left|x_{i,j}^{t}-x_p^{t+1}\right|*\boldsymbol{A}^{+}*L,i\leqslant\dfrac{n}{2}\end{cases}\text{。}(4)$$

式中，x_p^{t+1}为当前种群最优位置；$\boldsymbol{A}$是一个$1\times d$的矩阵，其每一个元素皆赋值为1或者-1，$\boldsymbol{A}^{+}=\boldsymbol{A}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}})^{-1}$；$n$是麻雀种群的大小；$i\leqslant n/2$时，跟随者种群根据发现者的位置进行觅食动作；当$i>n/2$时，麻雀种群没有获取到食物，其必须另寻他处觅食。在式(4)中，跟随者用于跟随发现者，进行局部寻优，使算法能够更快收敛于最优值。

2 自适应高斯变异的混沌SSA

2.1 伯努利混沌初始化种群

在伯努利（Bernoulli）混沌初始化种群中，种群初始化是否均匀是决定算法寻优效果的一个重要因素。因此引入混沌映射来进行算法种群的初始化，此举可提高初始种群多样性，以改善全局搜索能力。其中，Tent映射[15]以及Logistic映射[16]是最常见的混沌模型，通常用来初始化种群。图1和图2分别为Logistic映射和Tent映射生成值的频率图。

由图1和图2可知，Logistic映射在[0，0.1]和[0.9，1]的取值概率较大，Tent映射的概率分布均匀

度较差。图 3 所示为 Bernoulli 映射[17]的频率分布图,可以看出,相较于 Logistic 映射和 Tent 映射,Bernoulli 映射分布更均匀。

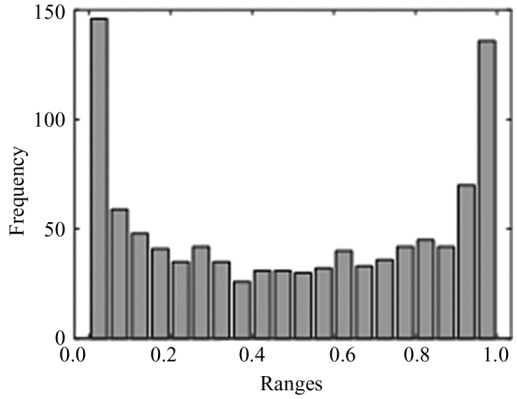


图 1　Logistic 映射频率分布

Fig. 1　Frequency distribution of Logistic mapping

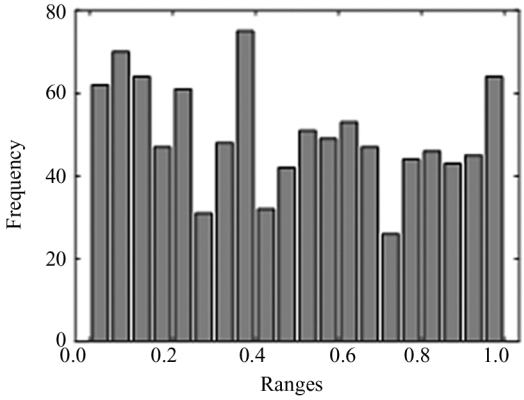


图 2　Tent 映射频率分布

Fig. 2　Frequency distribution of Tent mapping

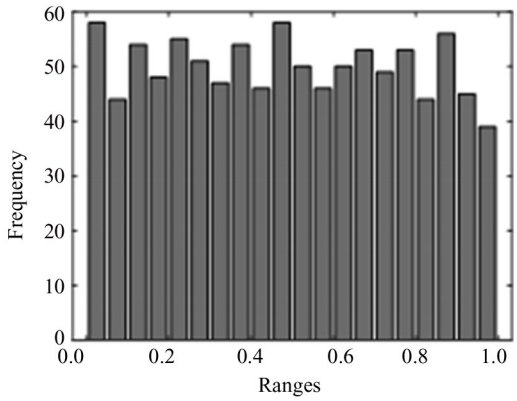


图 3　Bernoulli 映射频率分布

Fig. 3　Frequency distribution of Bernoulli mapping

为了提升初始解质量,本文采用 Bernoulli 映射混沌算子初始化种群,其迭代公式如下:

$$z_{k+1}=\begin{cases}z_k/(1-\lambda), z_k\in(0,1-\lambda]\\(z_k-1+\lambda)/\lambda, z_k\in(1-\lambda,1)\end{cases},\quad(5)$$

式中,k 为混沌迭代次数,λ 一般取 0.4。将产生的混沌序列根映射到解的搜索空间内:

$$x_{id}=x_{\mathrm{L}}+(x_{\mathrm{U}}-x_{\mathrm{L}})\times z_{id},\quad(6)$$

式中,x_{id} 为第 i 只麻雀在第 d 维的位置,x_{U} 和 x_{L} 分别为搜索空间的上下限,z_{id} 为式(5)产生的混沌值。

2.2　自适应高斯云变异策略

原始 SSA 中,发现者在 $R_2<ST$ 时将根据其当前位置进行更新,其移动范围如式所示:

$$y=\exp\left(\frac{-x}{\alpha T}\right),\quad(7)$$

式中,x 为迭代次数,y 为位置变化值。如图 4 所示,随着 x 变大,y 从(0,1)逐渐变为(0,0.4),麻雀在每一维的位置都在变小。由于 SSA 中发现者作为优势种群引导 SSA 的种群进行搜索,其搜索范围决定了 SSA 的寻优精度;然而在原算法中发现者的搜索范围随着迭代的进行而变小,其会导致算法在迭代后期的种群多样性下降,从而使算法容易陷入局部最优。

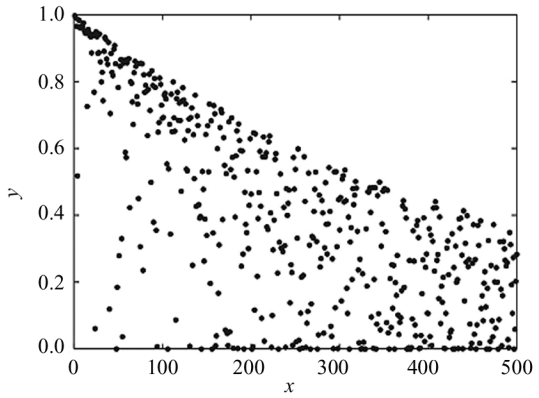


图 4　位置变量分布

Fig. 4　Location variable distribution

高斯云模型是一种基于高斯分布所生成的随机云模型,其能够有效模拟自然界的不确定性与分形现象[18],高斯云由许多个云滴组成。云模型可以通过 3 个参数来进行表征,他们分别是期望 Ex、熵 En 和超熵 He 。其中,期望 Ex 反应了云滴在空间内分布的期望,熵 En 反应了云滴的不确定性,超熵 He 则反应了熵 En 的不确定性,exp 是以自然常数 e 为底的指数函数。其隶属度函数如式(8)所示:

$$\mu=\exp[-(x-Ex)^2/(2(En')^2)]。\quad(8)$$

图 5 展示了 3 种参数状态下生成的高斯云模型。由图 5 可知,在期望 Ex 一定时,He 的值越大,高斯云分布越分散;En 越大,则高斯云的取值范围越宽泛。

利用高斯云模型具有随机性与模糊性的特点,

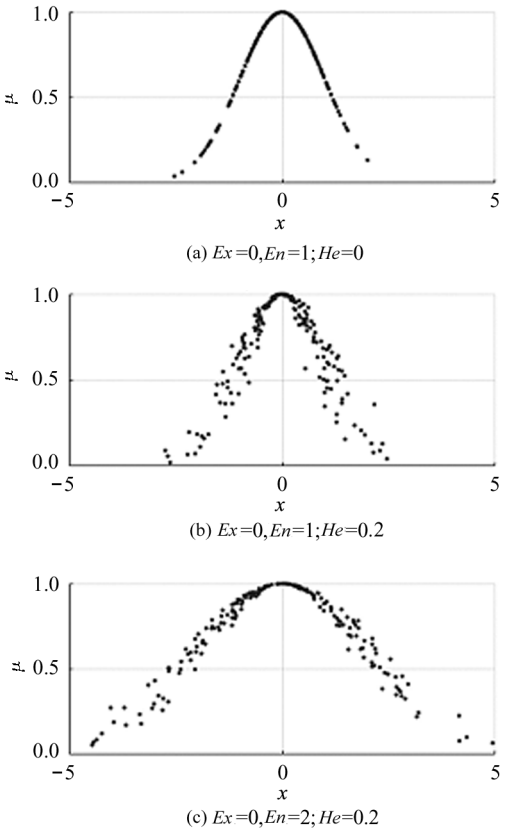
提出一种自适应高斯云变异策略来改进SSA。通过生成一个变异位置，使SSA能在一个小范围内进行领域搜索，同时有一定几率生成一个远离当前位置的新位置，使SSA能够在更大的范围内进行搜索寻优，提高了种群多样性，使算法拥有更优秀的跳出局部最优的能力，能够更好地进行全局搜索，提高了搜索速度并加快了算法的收敛速度。其位置更新式如下所示：

$$x_i^{t+1}=\begin{cases}x_i^t\cdot(1+\mathrm{Gauss}(0,\delta^2,0.1\delta^2)), R_2<ST\\ x_i^t+Q\cdot L, R_2\geqslant ST\end{cases},\tag{9}$$

式中，$\mathrm{Gauss}(0,\delta^2,0.1\delta^2)$表示参数满足$(Ex,En,He)=(0,\delta^2,0.1\delta^2)$的高斯云随机数，$\delta^2$定义如下：

$$\delta^2=\begin{cases}1, f_i\leqslant f_s\\ \exp\left(\dfrac{f_s-f_i}{|f_i|+e}\right), f_i>f_s\end{cases},\tag{10}$$

式中，f_i为第i只麻雀的适应度，f_s为随机选取发现者的适应度，e为无穷小常数。由上式可知，当第i只发现者的适应度优于第s只发现者，即$f_i\leqslant f_s$时，麻雀将向着更广阔的区域探索，避免陷入局部极值；否则，它会缩小搜索区域，增强局部位置的精细搜索。超熵$0.1\delta^2$则保证了变异解的模糊性，进一步提高了算法的搜索能力。



(a) Ex=0,En=1;He=0

(b) Ex=0,En=1;He=0.2

(c) Ex=0,En=2;He=0.2

图5　高斯云分布

Fig. 5　Gaussian cloud distribution

2.3　t分布随机反向学习

t分布的曲线形态和自由度参数n有关，其概率密度函数如式(11)所示：

$$p_t(x)=\frac{\Gamma\left(\frac{n+1}{2}\right)}{\sqrt{n\pi}\Gamma\left(\frac{n}{2}\right)}\cdot\left(1+\frac{x^2}{n}\right)^{-\frac{n+1}{2}}。\tag{11}$$

式中，$\Gamma\left(\frac{n+1}{2}\right)$表示第二型欧拉积分。$t$分布的概率密度图如图6所示，其中$N(0,1)$为标准正态分布。从图中可以看出，当自由度$n$较小时，$t$分布相较于正态分布更平缓。伴随$n$值的增大，$t$分布更接近正态分布。算法在迭代过程中，通过改变$n$值会获得不同幅度的扰动。

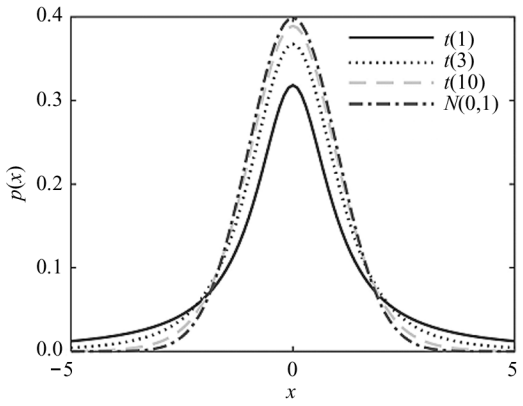


图6　t分布概率密度图

Fig. 6　t distribution probability density map

反向学习策略[19]最初是TIZHOOSH等在2015年提出的一种运用于群智能领域的改进策略，其根据当前解来生成一个反向解，进一步提高了算法的搜索能力，降低了算法陷入局部最优的可能性。生成反向学习解计算式如下所示：

$$x_d=UB+LB-x_t,\tag{12}$$

式中，UB和LB分别为搜索空间的上界和下界，x_t为当前解。而反向解与当前解的关系缺乏随机性，为了进一步提高反向解的随机性，以提高算法的种群多样性，避免算法陷入局部最优，采用t分布反向学习策略对麻雀位置进行扰动，其式如下：

$$x_{td}=UB+LB-t(iter)\cdot x_t,\tag{13}$$

式中，x_{td}为t分布反向学习后麻雀的位置，$t(iter)$为以当前迭代次数$iter$为变量的t分布。

由于不能保证变异后麻雀适应度值更优，利用贪婪原则[20]选择是否更新麻雀位置。即只有当变异

后麻雀的适应度值更优时,才对麻雀位置进行替换。式如下:

$$x_{\text{best}}=\begin{cases}x_{td},f(x_{td})<f(xb_d^t)\\xb_d^t,f(x_{td})\geqslant f(xb_d^t)\end{cases},\tag{14}$$

式中,x_{best}为贪婪选择后的最优麻雀位置。

2.4　ISSA 描述

基于自适应高斯云变异策略和 t 分布扰动策略的 ISSA 伪代码如下:

输入:搜索空间和目标函数

输出:最优解

1. 初始化种群规模、警戒者 PD 与侦察者 SD 比例、警戒阈值、最大迭代次数 Max_T 等

2. 根据式(5)生成 Bernoulli 初始种群

3. 计算初始适应度值以及当前拥有最优和最劣适应度值的麻雀种群位置

4. for $i=1$:Max_ T

5. for $j=1$: $PD*N$

6. 根据式(9)计算发现者位置

7. end for

8. for $j=PD*N:N$

9. 根据式(4)计算跟随者位置

10. end for

11. 在种群中随机选取$SD*N$只麻雀作为侦察者

12. for $j=1:SD*N$

13. 根据式(3)计算警戒者位置

14. end for

15. 计算每只个体适应度值并排序,找到当前最优个体和最劣个体

16. 根据式(13)计算当前最优个体的变异位置

17. 根据式(14)进行贪婪选择

18. end for

3　仿真实验与分析

3.1　实验设计

为了测试 ISSA 的寻优性能,利用 13 个基准函数对其进行测试,其中 F1—F7 为单峰高维测试函数,用以检验 ISSA 算法的寻优能力,F8—F13 为多峰多维测试函数,用以检测 ISSA 算法跳出局部最优的能力,维度数值分别取 $d=10$、$d=30$、$d=100$。为了避免随机因素对实验结果的影响,提高实验的准确性,所有算法将重复运行 30 次,并取 30 次寻优的平均值与标准差作为实验结果进行分析,实验结果在表 2 中列出。13 个基准测试函数如表 1 所示。

表 1　基准测试函数

Tab. 1　Benchmark function

Function	d	Range	Theory optimal value
$F_1(x)=\sum_{i=1}^{n}x_i^2$	10/30/100	$[-100,100]$	0
$F_2(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	10/30/100	$[-10,10]$	0
$F_3(x)=\sum_{i=1}^{n}(\sum_{j=1}^{i}x_j^2)^2$	10/30/100	$[-100,100]$	0
$F_4(x)=\max_i\{\lvert x_i\rvert,1\leqslant i\leqslant n\}$	10/30/100	$[-100,100]$	0
$F_5(x)=\sum_{i=1}^{n-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	10/30/100	$[-30,30]$	0
$F_6(x)=\sum_{i=1}^{n}(x_i+0.5)^2$	10/30/100	$[-1.28,1.28]$	0
$F_7(x)=\sum_{i=1}^{n}ix_i^4+\text{rand}[0,1]$	10/30/100	$[-1.28,1.28]$	0
$F_8(x)=\sum_{i=1}^{n}-x_i+\sin\sqrt{\lvert x_i\rvert}$	10/30/100	$[-500,500]$	$-418.9829D$
$F_9(x)=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)+10]$	10/30/100	$[-5.12,5.12]$	0
$F_{10}(x)=-20\exp\left(-0.2\sqrt{\frac{1}{2}\sum_{i=1}^{n}x_i}\right)-\exp[\frac{1}{2}\sum_{i=1}^{n}\cos(2\pi x_i)]+20+e$	10/30/100	$[-32,32]$	0
$F_{11}(x)=1+\frac{1}{4\,000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)$	10/30/100	$[-600,600]$	0

续表 1

Continued Tab. 1

Function	d	Range	Theory optimal value
$F_{12}(x)=\frac{\pi}{n}\{10\sin(\pi y_1\}+\sum_{i=1}^{n-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})+\sum_{i=1}^{n-1}u(x_i,10,100,4)]$, where $y_i=1+\frac{\pi_i+1}{4}$, $u(x_i,a,k,m)\begin{cases}K(x_i-a)^m, \text{if } x_i>a\\0,-a\leqslant x_i\geqslant a\\K(-x_i-a)^m,-a\leqslant x_i\end{cases}$	10/30/100	[−50,50]	0
$F_{13}(x)=0.1\left(\sin^2(3\pi x_1)+\sum_{i=1}^{n}(x_i-1)^2[1+\sin^2(3\pi x_i+1)]+(x_n-1)^2+\sin^2(2\pi x_n)\right)+\sum_{i=1}^{n}u(x_i,5,100,4)$	10/30/100	[−50,50]	0

3.2 算法对比分析

通过对表 2 的分析可知，ISSA 算法在相同的条件下对所选的 13 个基准测试函数的寻优能力要优于所对比的另外 4 种算法。在单峰高维测试函数 F1—F7 的测试结果中，ISSA 算法对 F1—F4 测试函数在 $d=10$、$d=30$、$d=100$ 的寻优结果均领先于其他几个对比算法，且具有显著的优越性。F6、F7 测试函数的结果中，ISSA 的寻优结果均领先于另外 3 个对比算法，仅在 $d=100$ 时以一个微小的差距落后于 SSA 算法。而在 F5 测试函数的结果中，虽然 ISSA 的寻优结果略差于 SSA 算法的寻优结果，但是相比于另外 3 个对比算法来说其寻优结果均领先几个数量级。在多峰高维测试函数 F8—F13 的测试结果中，对于 F9、F11 测试函数，ISSA 和 SSA 算法都能够收敛到最优值。而在 F8、F10、F13 测试函数中，ISSA 算法均能够取得最佳的寻优效果。

以上仿真测试的结果表明了 ISSA 算法在 10 维、30 维、50 维的计算条件下能够表现出较好的寻优性能，其寻优结果相比于其他 4 种对比算法有显著的优越性。

表 2 基准测试函数优化结果

Tab. 2 Benchmark test function optimization results

		Mean	Std	Mean	Std	Mean	Std
		$d=10$		$d=30$		$d=100$	
F1	ISSA	$\mathbf{0.00\times10^{0}}$	$\mathbf{0.00\times10^{0}}$	$\mathbf{0.00\times10^{0}}$	$\mathbf{0.00\times10^{0}}$	$\mathbf{0.00\times10^{0}}$	$\mathbf{0.00\times10^{0}}$
	GWO	8.43×10^{-57}	3.58×10^{-56}	1.44×10^{-27}	1.90×10^{-27}	1.47×10^{-12}	1.02×10^{-12}
	WOA	9.66×10^{-77}	5.29×10^{-76}	5.46×10^{-74}	2.96×10^{-73}	1.15×10^{-71}	6.22×10^{-71}
	SSA	9.11×10^{-34}	3.48×10^{-33}	1.33×10^{-32}	5.79×10^{-32}	4.95×10^{-35}	2.56×10^{-34}
	SCA	3.24×10^{-11}	1.66×10^{-10}	8.55×10^{0}	1.11×10^{1}	9.98×10^{3}	5.04×10^{3}
F2	ISSA	$\mathbf{8.03\times10^{-118}}$	$\mathbf{4.40\times10^{-117}}$	$\mathbf{1.90\times10^{-121}}$	$\mathbf{8.05\times10^{-121}}$	$\mathbf{1.52\times10^{-117}}$	$\mathbf{8.30\times10^{-117}}$
	GWO	1.26×10^{-19}	1.85×10^{-19}	1.66×10^{-9}	1.00×10^{-9}	4.06×10^{-4}	1.43×10^{-4}
	WOA	1.24×10^{-31}	3.15×10^{-31}	1.24×10^{-29}	4.38×10^{-29}	1.96×10^{-29}	3.66×10^{-29}
	SSA	3.77×10^{-25}	1.29×10^{-24}	1.09×10^{-21}	5.90×10^{-21}	4.76×10^{-23}	2.60×10^{-22}
	SCA	1.27×10^{-5}	2.13×10^{-5}	3.38×10^{-1}	6.84×10^{-1}	1.96×10^{1}	1.12×10^{1}
F3	ISSA	$\mathbf{6.37\times10^{-181}}$	$\mathbf{0.00\times10^{0}}$	$\mathbf{1.64\times10^{-148}}$	$\mathbf{9.00\times10^{-148}}$	$\mathbf{2.16\times10^{-160}}$	$\mathbf{1.18\times10^{-159}}$
	GWO	6.43×10^{-14}	1.93×10^{-13}	7.67×10^{-2}	1.23×10^{-1}	4.03×10^{3}	2.89×10^{3}
	WOA	7.83×10^{2}	1.05×10^{3}	6.58×10^{4}	1.95×10^{4}	1.07×10^{6}	3.98×10^{5}
	SSA	1.23×10^{-13}	4.80×10^{-13}	1.08×10^{-10}	5.88×10^{-10}	3.27×10^{-12}	1.70×10^{-11}
	SCA	5.58×10^{-1}	1.67×10^{0}	1.35×10^{4}	5.66×10^{3}	2.90×10^{5}	7.96×10^{4}
F4	ISSA	$\mathbf{7.79\times10^{-85}}$	$\mathbf{4.27\times10^{-84}}$	$\mathbf{2.95\times10^{-92}}$	$\mathbf{1.61\times10^{-91}}$	$\mathbf{7.99\times10^{-98}}$	$\mathbf{3.64\times10^{-97}}$
	GWO	2.44×10^{-10}	9.25×10^{-10}	1.01×10^{-3}	7.95×10^{-4}	6.70×10^{0}	4.35×10^{0}
	WOA	6.77×10^{0}	1.08×10^{1}	5.42×10^{1}	2.69×10^{1}	7.28×10^{1}	2.52×10^{1}
	SSA	6.44×10^{-9}	3.02×10^{-8}	2.42×10^{-8}	9.63×10^{-8}	3.44×10^{-8}	1.72×10^{-7}
	SCA	6.91×10^{-2}	1.36×10^{-1}	4.77×10^{1}	1.08×10^{1}	9.20×10^{1}	3.02×10^{0}

续表 2

Continued Tab. 2

		Mean	Std	Mean	Std	Mean	Std
		$d=10$		$d=30$		$d=100$	
F5	ISSA	3.16×10^{-4}	1.17×10^{-3}	**2.75×10^{-3}**	8.04×10^{-3}	1.26×10^{-2}	3.21×10^{-2}
	GWO	6.92×10^{0}	5.82×10^{-1}	2.72×10^{1}	6.12×10^{-1}	9.83×10^{1}	4.51×10^{-1}
	WOA	2.20×10^{1}	7.68×10^{1}	2.84×10^{1}	3.45×10^{-1}	9.83×10^{1}	1.42×10^{-1}
	SSA	**2.90×10^{-4}**	**9.71×10^{-4}**	3.24×10^{-3}	**7.98×10^{-3}**	**9.85×10^{-3}**	**1.76×10^{-2}**
	SCA	7.98×10^{0}	1.03×10^{0}	7.95×10^{5}	1.16×10^{6}	2.07×10^{8}	8.10×10^{7}
F6	ISSA	**2.85×10^{-6}**	**3.84×10^{-6}**	**1.37×10^{-5}**	**2.43×10^{-5}**	7.55×10^{-5}	1.74×10^{-4}
	GWO	1.67×10^{-2}	6.36×10^{-2}	8.98×10^{-1}	3.60×10^{-1}	1.18×10^{1}	1.05×10^{0}
	WOA	2.96×10^{-2}	5.16×10^{-2}	8.29×10^{-1}	3.01×10^{-1}	6.34×10^{0}	1.27×10^{0}
	SSA	3.87×10^{-6}	7.85×10^{-6}	3.02×10^{-5}	6.02×10^{-5}	6.46×10^{-5}	1.04×10^{-4}
	SCA	4.86×10^{-1}	1.72×10^{-1}	1.55×10^{2}	1.87×10^{2}	1.90×10^{4}	9.70×10^{3}
F7	ISSA	**3.18×10^{-4}**	3.35×10^{-4}	**5.47×10^{-4}**	**4.06×10^{-4}**	**3.85×10^{-4}**	**2.52×10^{-4}**
	GWO	4.23×10^{-4}	**2.59×10^{-4}**	3.83×10^{-3}	1.48×10^{-3}	1.48×10^{-2}	4.16×10^{-3}
	WOA	2.22×10^{-3}	2.74×10^{-3}	6.01×10^{-3}	5.86×10^{-3}	8.92×10^{-3}	1.15×10^{-2}
	SSA	4.97×10^{-4}	2.90×10^{-4}	7.50×10^{-4}	7.14×10^{-4}	6.61×10^{-4}	6.65×10^{-4}
	SCA	3.08×10^{-3}	3.37×10^{-3}	4.30×10^{-1}	5.07×10^{-1}	2.09×10^{2}	9.94×10^{1}
F8	ISSA	**-3.81×10^{3}**	4.34×10^{2}	**-1.23×10^{4}**	**$3.19\times10^{+2}$**	**-4.14×10^{4}**	**2.00×10^{2}**
	GWO	-2.68×10^{3}	3.58×10^{2}	-6.11×10^{3}	8.20×10^{2}	-1.51×10^{4}	3.37×10^{3}
	WOA	-3.32×10^{3}	6.30×10^{2}	-9.93×10^{3}	1.64×10^{3}	-3.35×10^{4}	5.90×10^{3}
	SSA	-2.84×10^{3}	6.05×10^{2}	-9.50×10^{3}	2.47×10^{3}	-3.62×10^{4}	5.50×10^{3}
	SCA	-2.10×10^{3}	**1.90×10^{2}**	-3.75×10^{3}	3.32×10^{2}	-6.64×10^{3}	5.84×10^{2}
F9	ISSA	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**
	GWO	2.45×10^{0}	3.35×10^{0}	8.02×10^{0}	6.69×10^{0}	4.08×10^{1}	2.98×10^{1}
	WOA	4.32×10^{0}	1.23×10^{1}	3.79×10^{-15}	2.08×10^{-14}	3.79×10^{-15}	2.08×10^{-14}
	SSA	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**
	SCA	5.83×10^{0}	1.30×10^{1}	6.38×10^{1}	4.31×10^{1}	2.97×10^{2}	1.37×10^{2}
F10	ISSA	**8.88×10^{-16}**	**0.00×10^{0}**	**8.88×10^{-16}**	**0.00×10^{0}**	**8.88×10^{-16}**	**0.00×10^{0}**
	GWO	2.68×10^{-14}	4.95×10^{-15}	1.09×10^{-8}	3.70×10^{-9}	3.14×10^{-4}	9.02×10^{-5}
	WOA	4.56×10^{-15}	3.43×10^{-15}	5.98×10^{-15}	2.75×10^{-15}	7.16×10^{-15}	3.58×10^{-15}
	SSA	1.18×10^{-13}	5.08×10^{-13}	2.43×10^{-14}	9.22×10^{-14}	1.08×10^{-13}	5.19×10^{-13}
	SCA	5.74×10^{-1}	2.01×10^{0}	1.49×10^{1}	7.32×10^{0}	1.90×10^{1}	3.77×10^{0}
F11	ISSA	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**
	GWO	3.29×10^{-2}	3.08×10^{-2}	8.15×10^{-3}	1.05×10^{-2}	1.54×10^{-2}	2.43×10^{-2}
	WOA	1.21×10^{-1}	1.85×10^{-1}	3.70×10^{-18}	2.03×10^{-17}	**0.00×10^{0}**	**0.00×10^{0}**
	SSA	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**
	SCA	1.60×10^{-1}	1.95×10^{-1}	2.24×10^{0}	1.37×10^{0}	1.82×10^{2}	9.55×10^{1}
F12	ISSA	**1.89×10^{-6}**	4.06×10^{-6}	3.31×10^{-6}	**5.81×10^{-6}**	1.15×10^{-6}	1.97×10^{-6}
	GWO	5.12×10^{-3}	1.01×10^{-2}	5.45×10^{-2}	2.88×10^{-2}	3.87×10^{-1}	8.39×10^{-2}
	WOA	1.95×10^{-1}	9.94×10^{-1}	4.84×10^{-2}	3.56×10^{-2}	9.80×10^{-2}	4.06×10^{-2}
	SSA	2.11×10^{-6}	**3.21×10^{-6}**	**1.72×10^{-6}**	6.10×10^{-6}	**8.94×10^{-7}**	**1.95×10^{-6}**
	SCA	1.13×10^{-1}	3.97×10^{-2}	6.43×10^{5}	1.69×10^{6}	5.17×10^{8}	2.28×10^{8}

续表 2

Continued Tab. 2

		Mean	Std	Mean	Std	Mean	Std
		$d=10$		$d=30$		$d=100$	
F13	ISSA	**4.51×10^{-6}**	**8.32×10^{-6}**	**2.21×10^{-5}**	**5.63×10^{-5}**	**2.63×10^{-5}**	**4.92×10^{-5}**
	GWO	1.73×10^{-2}	3.69×10^{-2}	8.08×10^{-1}	2.61×10^{-1}	7.61×10^{0}	5.82×10^{-1}
	WOA	6.76×10^{-2}	7.62×10^{-2}	9.00×10^{-1}	3.88×10^{-1}	3.62×10^{0}	1.16×10^{0}
	SSA	6.50×10^{-6}	8.82×10^{-6}	2.85×10^{-5}	6.60×10^{-5}	3.38×10^{-5}	5.87×10^{-5}
	SCA	3.85×10^{-1}	8.40×10^{-2}	1.12×10^{6}	2.41×10^{6}	8.82×10^{8}	3.41×10^{8}

3.3 算法收敛的对比分析

为了能够更加直观地展示 ISSA 算法的寻优性能，在图 7 中，(a)—(f)分别代表 5 种算法在 F1、F2、F6、F7、F9、F10 的收敛曲线图。其中，图中横坐标为迭代次数，纵坐标为平均适应度的 log 值。

对于测试函数F1、F2，ISSA均能以最快的收敛

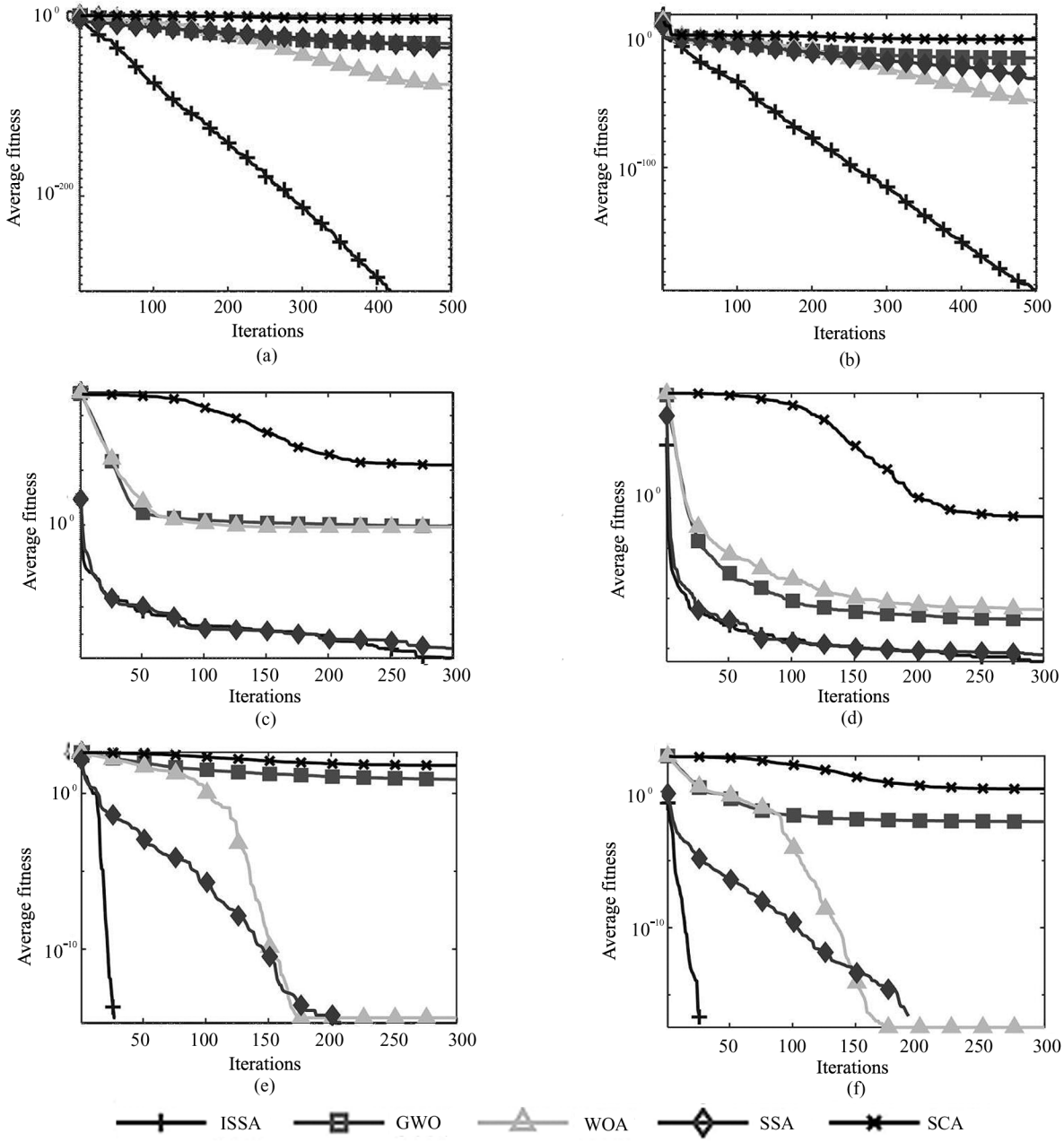


图 7　5 种算法的收敛曲线对比

Fig. 7　Comparison of convergence curves of five algorithms

速度进行迭代。对于测试函数 F6、F7,ISSA 明显有着极高的寻优性能,并能取得最好的寻优结果。对于函数 F9、F11,ISSA 均能在迭代 50 次之前收敛到理论最优值,且收敛精度最高。

综上,ISSA 在求解不同测试函数时,具有更快的收敛速度及精度。

3.4　与其他 ISSAs 比较

为了进一步体现本文改进策略的优越性,与文献[21]中 ISSA1 进行比较,两种算法的基本参数设置也以该文献为准。其中,得到 13 个基准测试函数在 30 维下的实验结果如表 3 所示。

由表3可知,对于函数F1、F2、F3、F4,相较于

表 3　ISSA 与 ISSA1 性能对比

Tab. 3　Performance comparison between ISSA and ISSA1

Function	ISSA1		ISSA	
	Mean	Std	Mean	Std
F1	1.00×10^{-87}	5.50×10^{-87}	$\mathbf{0.00\times10^{0}}$	$\mathbf{0.00\times10^{0}}$
F2	2.84×10^{-59}	1.55×10^{-58}	$\mathbf{0.00\times10^{0}}$	$\mathbf{0.00\times10^{0}}$
F3	4.43×10^{-34}	1.69×10^{-33}	$\mathbf{0.00\times10^{0}}$	$\mathbf{0.00\times10^{0}}$
F4	7.40×10^{-42}	2.91×10^{-41}	$\mathbf{0.00\times10^{0}}$	$\mathbf{0.00\times10^{0}}$
F5	7.30×10^{-3}	3.08×10^{-2}	$\mathbf{6.80\times10^{-3}}$	$\mathbf{2.26\times10^{-2}}$
F6	$\mathbf{8.15\times10^{-4}}$	7.23×10^{-4}	8.45×10^{-4}	$\mathbf{7.10\times10^{-4}}$
F7	-8.94×10^{3}	$\mathbf{2.86\times10^{3}}$	$\mathbf{-1.21\times10^{4}}$	6.37×10^{2}
F8	$\mathbf{0.00\times10^{0}}$	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
F9	8.88×10^{-16}	0.00×10^{0}	8.88×10^{-16}	0.00×10^{0}
F10	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
F11	1.65×10^{-6}	2.87×10^{-6}	6.61×10^{-6}	9.90×10^{-6}
F12	8.19×10^{-5}	2.04×10^{-4}	$\mathbf{5.54\times10^{-5}}$	$\mathbf{1.58\times10^{-4}}$
F13	1.46×10^{-6}	2.68×10^{-6}	8.35×10^{-6}	1.70×10^{-5}

ISSA1,ISSA 在平均值和标准差上提升了至少 33 个数量级。对函数 F5 和 F6,ISSA 和 ISSA1 都没有找到最优解,且两者的优化结果相差不大。对于函数 F7(理论最优解为 −12 569.487),ISSA 在收敛精度上提升较大。对于函数 F8、F9、F10,两种算法的优化结果相同,但实际上 ISSA 的收敛速度较 ISSA1 更快(由于篇幅原因,此处不再给出迭代曲线)。ISSA 和 ISSA1 在优化函数 F11 和 F12 时各有优势,两者的优化结果相近。总体而言,ISSA 的优化效果要好于 ISSA1,在优化不同函数时的适应性更强。

4　基于 ISSA 的多阈值图像分割

4.1　Kapur 熵多阈值图像分割原理

图像分割是图像处理特征的关键预处理技术之一,历经多年,目前已有多种图像分割技术被提出。而图像阈值分割法是一种高效、稳健、准确的图像分割方法[22],并且图像阈值分割法目前也被广泛地应用于医学、农业等多种应用场景。基于 Kapur 熵的图像多阈值分割是利用一组灰度阈值 $[t_{n1},t_{n2},\cdots,t_{ns}]$ 将图像划分为 $s+1$ 个区域 $\{A_0,A_1,\cdots,A_0\}$,图像的 Kapur 熵目标函数可被定义为:

$$H(t_{n1},t_{n2},\cdots,t_{ns})=H_0+H_1+\ldots+H_s\ , \tag{15}$$

式中,$H_0,H_1,\cdots,H_s$ 为不同区域的熵,定义如下:

$$H_0=\sum_{i=0}^{t_{n1}-1}\frac{p_i}{\omega_0}\cdot\ln(\frac{p_i}{\omega_0}),\omega_0=\sum_{i=0}^{t_{n1}-1}p_i\ , \tag{16}$$

$$H_1=\sum_{i=t_{n1}}^{t_{n2}-1}\frac{p_i}{\omega_1}\cdot\ln(\frac{p_i}{\omega_1}),\omega_1=\sum_{i=t_{n1}}^{t_{n2}-1}p_i\ , \tag{17}$$

$$H_s=\sum_{i=t_{ns}}^{L-1}\frac{p_i}{\omega_s}\cdot\ln(\frac{p_i}{\omega_s}),\omega_s=\sum_{i=t_{ns}}^{L-1}p_i\ , \tag{18}$$

式(16)—(18)中,p_i 为该灰度等级占全图灰度等级的百分比,$\omega_0,\omega_1,\cdots,\omega_s$ 为阈值区间内灰度等级之和,则最优分割阈值 $[t_{n1}{}^*,t_{n2}{}^*,\ldots,t_{ns}{}^*]$ 应满足:

$$[t_{n1}^*,t_{n2}^*,\cdots,t_{ns}^*]=\operatorname{argmax}\{H(t_{n1},t_{n2},\cdots,t_{ns})\}\ 。 \tag{19}$$

4.2　基于 ISSA 的图像分割

采用 ISSA 对图像进行 Kapur 熵图像分割的方式是:利用 ISSA 对式(15)进行迭代寻优,寻找一组 $[t_{n1},t_{n2},\cdots,t_{n0}]$ 使得图像的总 Kapur 熵值最大,以获得图像的最优分割阈值。

为验证提出的 ISSA 在多阈值图像分割的效果,选取了 Cameraman、Lake、Pepers 作为测试图片,以

5种算法来进行Kapur熵多阈值分割，测试图片如图8所示。分割阈值个数s分别取2、3、4、5个，其分割结果如表4所示。其中s为3、4、5的分割结果图如图9所示。

图8　测试图片

Fig. 8　Test image

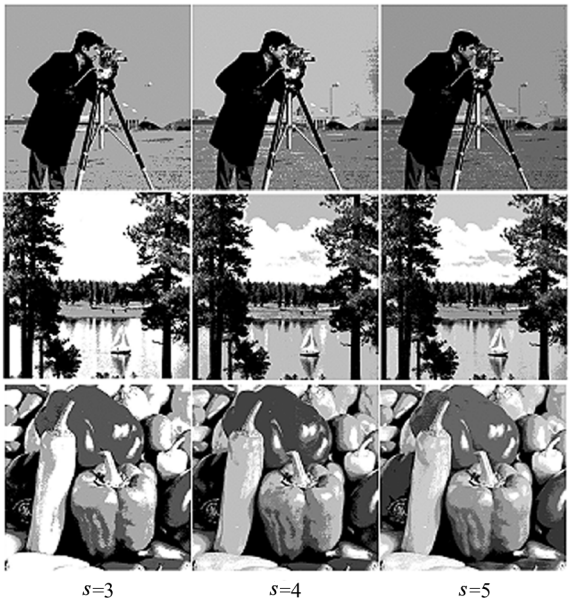


图9　ISSA分割结果

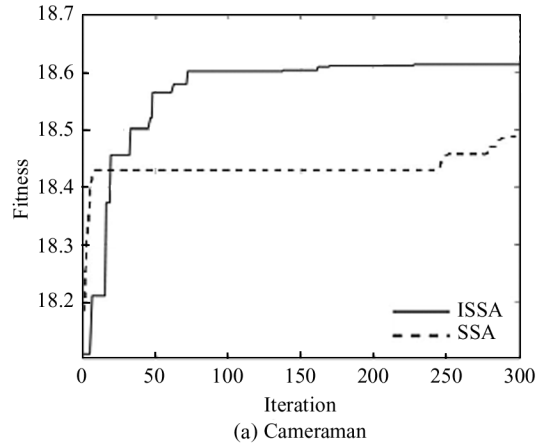
Fig. 9　ISSA segmentation result

由表4可知，在Cameraman测试图中，ISSA算法在阈值个数为3、4、5时均能获得最优的阈值分割结果。在Lake和Pepers测试图中，处理多阈值时，ISSA能够获得更好的分割效果。ISSA与SSA在进行图像分割时的迭代曲线如图10所示，其中横坐标为迭代次数，纵坐标为适应度值。可以看出ISSA有着更好的收敛性能以及更高的分割精度，说明了ISSA算法在多阈值图像分割任务中的有效性。

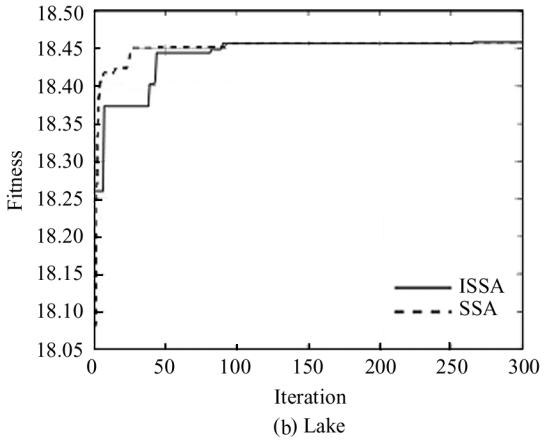
表4　阈值分割结果对比

Tab. 4　Comparison of threshold segmentation results

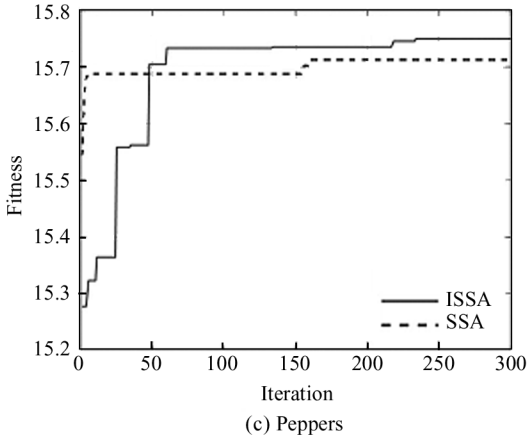
Test image	s	ISSA	SSA	GWO	WOA	SCA
Cameraman	2	**12.305**	12.256	**12.305**	**12.305**	12.343
	3	**15.423**	15.321	15.291	15.243	15.275
	4	**18.614**	18.268	18.083	18.373	18.336
	5	**21.424**	20.995	21.264	21.367	21.219
Lake	2	**12.547**	12.467	12.256	**12.547**	12.164
	3	**15.622**	15.225	15.614	15.133	15.339
	4	**18.457**	18.339	17.913	17.986	17.919
	5	**21.163**	21.116	20.976	20.963	20.999
Pepers	2	**10.852**	10.394	10.492	10.549	10.637
	3	**13.377**	13.127	12.968	12.951	13.319
	4	**15.750**	15.137	15.462	15.258	15.530
	5	**17.737**	17.696	17.413	17.534	17.147



(a) Cameraman



(b) Lake



(c) Peppers

图10　分割收敛曲线

Fig. 10　Segmentation convergence curve

5 结　论

本文在基本 SSA 基础上,通过 Bernoulli 混沌初始化种群、自适应高斯云变异策略,以及引入 t 分布反向学习策略,在很大程度上改善了 SSA 收敛精度低、易陷入局部最优的缺陷。利用 13 个基准测试函数的实验结果表明,改进算法的收敛速度更快、精度更高、鲁棒性更强。最后将该算法应用到图像分割中,仿真实验验证了本文所提算法在实际工程应用中的可行性。

参考文献:

[1] SEYEDALI M, ANDREW L. The whale optimization algorithm[J]. Advances in Engineering Software, 2016, 95: 56-80.

[2] SEYEDALI M. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm[J]. Knowledge-Based Systems, 2015, 89: 230-251.

[3] SEYEDALI M. SCA: A sine cosine algorithm for solving optimization problems [J]. Knowledge-Based Systems, 2016, 96: 312-350.

[4] MIRJALILI S, MIRJALILI S M, LEWIIS A. Grey wolf optimizer[J]. Advances in Engineering Software, 2014, 69(3): 46-61.

[5] CHEN Z Y, ZHANG D M, XIN Z Y. Multi-subpopulation based symbiosis and non-uniform Gaussian mutation salp swarm algorithm [J]. Acta Automatica Sinica, 2022, 48(5): 1307-1317.
陈忠云,张达敏,辛梓芸.多子群的共生非均匀高斯变异樽海鞘群算法[J].自动化学报,2022,48(5):1307-1317.

[6] WEI X, FENG F. Optimization of empire competition algorithm based on Gauss-Cauchy mutation [J]. Computer Science, 2021, 48(S2): 142-146.
魏昕,冯锋.基于高斯-柯西变异的帝国竞争算法优化[J].计算机科学,2021,48(S2):142-146.

[7] ZHANG L L, MA Y CHEN Y L. Industrial control Intrusion detection based on CGWO optimization of Gaussian process [J]. Computer Engineering and Design, 2021, 42(12): 3351-3358.
张利隆,马垚,陈永乐.基于 CGWO 优化高斯过程的工控入侵检测[J].计算机工程与设计,2021,42(12):3351-3358.

[8] ZHAO C, WANG Y F, LIN L. State of charge estimation for lithium battery based on kernel extreme learning machine optimized by improved grey wolf algorithm[J]. Information and Control, 2021, 50(6): 731-739.
赵超,王延峰,林立.基于改进灰狼算法优化核极限学习机的锂电池动力电池荷电状态估计[J].信息与控制,2021,50(6):731-739.

[9] XIONG X, HU X, GUO H. A hybrid optimized grey seasonal variation index model improved by whale optimization algorithm for forecasting the residential electricity consumption[J]. Energy, 2021, 234: 318-423.

[10] GU X B, LI J Q, XU X, et al. Optimization of slurry circulating pump combination based on moths to flame algorithm[J]. Thermal Power Engineering, 2021, 36(6): 44-50.
谷小兵,李建强,徐贤等.基于飞蛾扑火算法的浆液循环泵组合优化[J].热能动力工程,2021,36(6):44-50.

[11] WAN H F, QI C F, ZHANG Y, et al. Knee MRI segmentation algorithm based on chaotic moths to flame optimization [J]. Journal of Northeastern University (Natural Science), 2020, 41(3): 326-331.
王海芳,祁超飞,张瑶,等.基于混沌飞蛾扑火优化的膝盖 MRI 分割算法[J].东北大学学报(自然科学版),2020,41(3):326-331.

[12] LIU L J, LIU D Y, LIU T T. Application of improved sine-cosine optimization algorithm in WSN coverage[J]. Mathematics in Practice and Theory, 2021, 51(11): 129-137.
刘丽娟,刘定一,刘婷婷.改进的正余弦优化算法在 WSN 覆盖中的应用[J].数学的实践与认识,2021,51(11):129-137.

[13] DUTTA P, NAYAK S K. Grey wolf optimizer based PID controller for speed control of BLDC motor[J]. Journal of Electrical Engineering & Technology, 2021, 16(2): 955-961.

[14] XUE J K, SHEN B. A novel swarm intelligence optimization approach: sparrow search algorithm [J]. Systems Science & Control Engineering, 2020, 8(1): 22-34.

[15] GAMBHIR G, MANDAL J K. Shared memory implementation and performance analysis of LSB steganography based on chaotic tent map [J]. Innovations in Systems and Software Engineering, 2021, 17(4): 56-63.

[16] CHENG L Z, SHIFEI D. A stochastic configuration network based on chaotic sparrow search algorithm [J]. Knowledge-Based Systems, 2021, 220: 315-331.

[17] CHIKUSHI R T M, BARROS R S M, SILVA M G N, et al. Using spectral entropy and bernoulli map to handle concept drift [J]. Expert Systems with Applications, 2021, 167: 114114.

[18] XU L, ZHANG Z Y, CHEN X, et al. Improved sparrow search algorithm based BP neural networks for aero-optical imaging deviation prediction[J]. Journal of Optoelectronics · Laser, 2021, 32(6): 653-658.

许亮,张紫叶,陈曦,等.基于改进麻雀搜索算法优化BP神经网络的气动光学成像偏移预测[J].光电子·激光,2021,32(6):653-658.

[19] TIZHOOSH H. Opposition-based learning: A new scheme for machine intelligence[C]//International Conference on Computational Intelligence for Modelling, Control & Automation, November 28-30, 2005, Vienna, Austria. New York: IEEE, 2005: 695-701.

[20] FAN W, MENG J, DU Y F, et al. LQR control of precision vibration isolation system based on improved particle swarm algorithm[J]. Electronic Measurement Technology, 2022, 45(2): 104-109.
范伟,孟江,杜永飞,等.基于改进粒子群算法的精密隔振系统LQR控制[J].电子测量技术,2022,45(2):104-109.

[21] MAO Q H, ZHANG Q, MAO C C, et al. Mixing sine and cosine algorithm with Lévy flying chaotic sparrow algorithm [J]. Journal of Shanxi University (Natural Science Edition), 2021, 44(6): 1086-1091.
毛清华,张强,毛承成等.混合正弦余弦算法和Lévy飞行的麻雀算法[J/OL].山西大学学报(自然科学版),2021,44(6):1086-1091.

[22] SOWJANYA K, INJETI S K. Investigation of butterfly optimization and gases Brownian motion optimization algorithms for optimal multilevel image thresholding[J]. Expert Systems With Applications 2021, 182: 165-172.

作者简介:

龙英文 (1974—),男,博士,副教授,硕士生导师,主要从事研究方向为人工智能与图像处理方面的研究.