DOI:10.19651/j.cnki.emt.2210205

基于螺旋搜索机制的行星搜索算法*

司书千 窦震海 王梓辰 董 军
(山东理工大学电气与电子工程学院 淄博 255000)

摘 要: 螺旋搜索机制的全局搜索能力强,广泛用于萤火虫算法及鲸鱼搜索算法,但其收敛速度慢,收敛精度低,局部搜索能力较差。通过改变收敛范围较小时的搜索模式,提出了局部螺旋搜索来提高其局部搜索能力,并引入变异操作提高其局部搜索能力,提出了行星搜索算法。通过对单峰及多峰值测试函数对该算法进行验证。结果表明行星搜索算法在收敛速度、搜索精度及局部搜索能力等方面较粒子群算法、萤火虫算法及鲸鱼搜索算法等有明显提升。

关键词: 螺旋搜索;搜索范围;行星搜索算法;全局收敛性;收敛精度
中图分类号: TP181 **文献标识码:** A **国家标准学科分类代码:** 520.10

Planet search algorithm based on spiral search mechanism

Si Shuqian Dou Zhenhai Wang Zichen Dong Jun
(School of Electrical and Electronic Engineering, Shandong University of Technology, Zibo 255000, China)

Abstract: Spiral search mechanism has strong global search ability and is widely used in firefly and whale search algorithms, but its convergence speed is slow, the convergence accuracy is low, and the local search ability is poor. By changing the search mode with small convergence range, a local spiral search is proposed to improve its local search ability, and a mutation operation is introduced to improve its local search ability, and a planet search algorithm is proposed. The algorithm is verified by single-peak and multi-peak test functions. The results show that the planet search algorithm is better than particle swarm optimization, firefly algorithm and whale search algorithm in convergence speed, search accuracy and local search ability.

Keywords: spiral search;search scope;planet search algorithm;global convergence;convergence precision

0 引 言

螺旋搜索机制由 Mirjalili 提出,用于表征飞蛾扑火算法(moth-flame optimization, MFO)以及鲸鱼搜索算法(whale optimization algorithm,WOA)的寻优过程[1-2]。螺旋搜索机制是粒子围绕当前最优点进行螺旋收缩搜索,从而寻找到最优值的一种搜索机制。相比于粒子群算法(particle swarm optimization, PSO)及遗传算法(genetic algorithm,GA)[3]的搜索机制,利用螺旋搜索机制的算法全局搜索能力较强[4-5]。

李安东等[6]采用改进的鲸鱼搜索算法对测试函数进行求解,其相较 PSO 等群体优化算法可以更快的得到精度较高的全局最优点,证明了螺旋搜索机制的全局搜索能力较强;尚猛在[7]中采用应用螺旋搜索机制的鲸鱼搜索算法的算法对物流配送中心选址模型进行求解,验证螺旋搜索机制在对多变量模型进行求解时较 PSO 等群体智能算法的收敛速度更快;张达敏[8]等利用 Circle 混沌序列生成初始化种群,提升种群的均匀分布程度。以上研究成果虽然对 WOA 算法进行了一定程度上的改进,但是在收敛精度以及收敛速度方面仍有一定程度的提升空间,尤其是一些基准测试函数的仿真结果并不十分理想。

但螺旋搜索机制确定螺旋中心后,缺少跳出局部最优的机制,故容易陷入局部最优点,通过采用混沌映射和改进其收敛因子可以改进这一缺点,但效果并不明显[5,9]。针对这一缺陷,本文采用对粒子进行变异以及改变其收敛范围较小时的搜索方式,提出一种基于螺旋搜索机制的行星搜索算法。通过对单峰及多峰值测试函数对本文所提算法与粒子群算法及鲸鱼优化算法等算法进行仿真实验,验证其收敛速度、搜索精度及局部搜索能力。

1 行星搜索算法

1.1 算法原理

本文借鉴太阳系的运行规律,利用当前迭代次数下的

收稿日期:2022-06-04
* 基金项目:山东省自然科学基金、山东省自然科学基金青年项目、淄博市重点研发计划项目资助

最优点作为恒星的位置,其他粒子围绕恒星做螺旋运动;在下一次迭代中,用新的最优点代替当前迭代次数下的恒星位置,最终形成稳定的"星系",从而找到最优点。

本文利用高斯变异[10],对搜索过程中适应度较差的粒子进行变异操作,以提高算法的搜索能力;并采用局部螺旋搜索,对当前迭代次数下适应度较好的粒子改变其迭代形式,使其在小范围做圆周搜索,以提高其局部搜索能力,使算法不易陷入局部最优,且在螺旋搜索机制上使算法的精度得到提升。

1.2　算法模型

本文提出的行星搜索算法的实现目标,是由混乱"星球"最终形成稳定的星系。鉴于此,这里首先确定当前粒子种群中的最优点 $P_{t,\mathrm{best}}$ 作为"恒星",其他粒子作为"行星",并围绕"恒星"做螺旋搜索,而"星球"的迭代方程可为:

$$p_{t+1,i} = L \cdot e^{r} \cdot \cos(2\pi r) + p_{t,\mathrm{best}} \tag{1}$$

其中,$P_{t,i}$ 为第 t 次迭代过程中"星球"的位置;$P_{t,\mathrm{best}}$ 代表第 t 次迭代过程中适应度最好的"星球"的位置;$L = | P_{t,\mathrm{best}} - P_{t,i} |$ 表示恒星与行星之间的距离;r 为[−1,1]的随机数,$t \in (1,\mathrm{maxgen})$。

粒子运行过程中,适应度较好的粒子会吸引附近的"星球"作为"卫星",并做螺旋运动,即:

$$\begin{cases} p_{t+1,i,j} = L\cos(2\pi r) + p_{t,k\mathrm{good},j}, & j \neq a \\ p_{t+1,i,j} = L\sin(2\pi r) + p_{t,k\mathrm{good},j}, & j = a \end{cases} \tag{2}$$

其中,$P_{t,k\mathrm{good}}$ 为当前迭代次数下除最优点外适应度较好的点,其中 $k \in (1, length(P)/10)$,L 表示"卫星"到"行星"的距离,a 为[1,n]的随机整数。

运行过程中适应度较差的"星球",会作为"流星"采用变异进行迭代,以提高其全局搜索能力,其迭代的表示式为:

$$p_{t+1,\beta} = p_{t,\beta} e^{l} \tag{3}$$

其中,$P_{t,\beta}$ 为当前迭代次数下适应度较差的"星球";$l = 2(1 - t/(r \times \mathrm{maxgen}))$为[−1,1]的随机数;$r$ 为[−1,1]的随机数。

当 $L < D$ 时,说明星系达到稳定,行星不再做螺旋收缩运动,而做螺旋运动,相应的迭代公式为:

$$\begin{cases} p_{t+1,i,j} = L\cos(2\pi r) + p_{t,\mathrm{best},j}, & j \neq a \\ p_{t+1,i,j} = L\sin(2\pi r) + p_{t,\mathrm{best},j}, & j = a \end{cases} \tag{4}$$

其中,$D = (1 - t/\mathrm{maxgen})b$;$b$ 代表系统稳定时"行星"与"恒星"之间的距离。

图 1 中,虚线圆范围代表稳定后"星球"的分布,当系统稳定后,粒子做螺旋运动,系统在最优点附近的局部搜索能力得到加强,寻优精度得到提高。

1.3　行星搜索算法流程

步骤 1)初始化参数:最大迭代次数 maxgen、"星球"行星粒子数 sizepop 等;

步骤 2)初始化行星的位置 P_i,计算种群中各行星粒子的适应度,并据适应度大小,选出当前种群中最优的行星个体,作为恒星 P_{best};

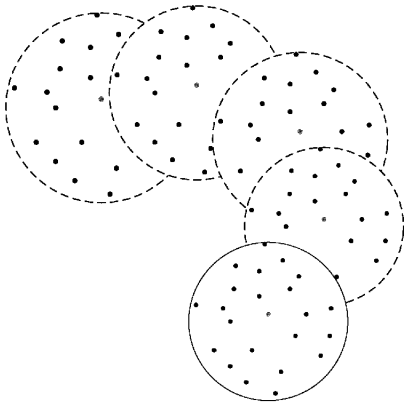

图 1　系统稳定后"星球"分布图

步骤 3)判断本次迭代中各行星粒子与恒星之间的距离 L,以决定粒子迭代方式为螺旋收缩搜索还是螺旋搜索,并根据式(1)和(4)迭代行星粒子新的位置 $P_{t+1,i}$;

步骤 4)在本次迭代中适应度较好的"行星"粒子,会吸引附近粒子做"卫星"并进行迭代,根据式(2),迭代后确定粒子位置 $P_{t+1,i}$;

步骤 5)将本次迭代中适应度较差的行星粒子根据式(3)进行变异操作;

步骤 6)计算所有粒子的适应度,并找出本次迭代过程中适应度最好的粒子作为"恒星";

步骤 7)结束条件的判定。停止迭代的标准是当前已迭代到预先设置的最大次数,或已满足最小误差要求,于是输出最优解和全局最优个体;若判定不满足结束条件,则返回步骤 3)。

2　实验结果与分析

仿真验证通过对测试函数的测试来进行,分别针对单峰值及多峰值函数分别验证其有效性及可行性。

2.1　算法实验环境及参数设定

为验证行星搜索算法的有效性和可行性,这里采用 12 个基准测试函数输入该算法,以检测它的性能。算法仿真实验是在 Intel(R) Core(TM) i7-8750H CPU、2.2 GHz 主频、8 GB 内存电脑上完成的,操作系统为 Windows10 (64 位),编程软件是 MATLAB R2016b;算法种群规模设置为 30,迭代次数是 500 次。

2.2　测试函数

验证算法用的 12 个典型基准测试函数,提供在表 1 中,它们均取自文献[2],其中,f1～f7 为单峰测试函数;f8～f12 为非线性多峰测试函数。不同形式的测试函数,可用于测试算法的优化效果和有效性。

为验证本文所提算法在多变量模型中的有效性,将测试函数的维数均设为 100,通过进行对测试函数的多次求解验证该函数在单峰值及多峰值函数中的有效性。

表 1　基准测试函数

函数	表达式	搜索范围	理论最优
Sphere	$f_1(x)=\sum_{i-1}^{n}x_i^2$	[−100,100]	0
Schwefel 2.22	$f_2(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	[−10,10]	0
Schwefel 1.2	$f_3(x)=\sum_{i=1}^{n}\left(\sum_{j-1}^{i}x_j\right)^2$	[−100,100]	0
Schwefel 2.21	$f_4(x)=\max\{\lvert x_i\rvert,\ 1\leqslant i\leqslant n\}$	[−100,100]	0
Rosenbrock	$f_5(x)=\sum_{i=1}^{n-1}[100(x_i-x_i^2)^2+(x_i-1)^2]$	[−30,30]	0
Step	$f_6(x)=\sum_{i-1}^{n}(\lvert x_i+0.5\rvert)^2$	[−100,100]	0
Quartic	$f_7(x)=\sum_{i=1}^{n}ix_i^4+random[0,1)$	[−1.28,1.28]	0
Schwefel 2.26	$f_8(x)=\sum_{i=1}^{n}-x_i\sin(\sqrt{\lvert x_i\rvert})$	[−500,500]	−418.982 9D
Rastrigin	$f_9(x)=\sum_{i-1}^{n}[-x_i^2-10\cos(2\pi x_i)+10]$	[−5.12,5.12]	0
Ackley	$f_{10}(x)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right)+20+e$	[−32,32]	0
Griewank	$f_{11}(x)=\frac{1}{4\ 000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	[−600,600]	0
Penalized	$f_{12}(x)=\frac{\pi}{n}\left\{\begin{array}{l}10\sin(\pi y_1)+\sum_{i=1}^{n-1}(y_i-1)^2[1-10\sin^2(\pi y_{i+1})]\\+(y_n-1)^2+\sum_{i=1}^{n}\mu(x_i,10,1\ 000,4)\end{array}\right\}$ $y_i=1+\frac{x_i+1}{4}\mu(x_i,a,k,m)=\begin{cases}k(x_i-a)^m, & x_i>a\\0, & -a<x_i<a\\k(-x_i-a)^m, & x_i<-a\end{cases}$	[−50,50]	0

2.3　实验结果

算法验证中，在相同迭代次数、种群规模及运行次数下，将本文提出的行星搜索算法与粒子群算法[11]、遗传算法[12]、飞蛾扑火算法、鲸鱼优化算法[13]以及麻雀搜索算法(sparrow search algorithm，SSA)[14-15]的进行了对比，基于仿真计算结果，画出了它们的收敛曲线和收敛精度，具体如表 2 和图 4 所示。

由表 2 可看出，相比于 PSO、GA、MFO、WOA 以及 SSA 算法，本文提出的行星搜索算法的平均收敛精度以及最优值均是更优的，且其方差也远小于其他算法，说明行星搜索算法的寻优能力较强，稳定性较好。

从图 2 可以看出，相比于经典的 PSO、GA 算法，行星搜索算法、MFO 以及 WOA 算法等利用螺旋搜索机制的算法，不论在单峰值函数还是多峰值函数下，均可得到收敛精度更高的结果，这是因为，螺旋搜索机制在寻优过程中具有强大的全局搜索能力。

且从图 2(c)～(f)中可看出，相比于同样利用了螺旋搜索机制的 MFO 和 WOA 算法，行星搜索算法不易陷入局部最优点，这是因为，该算法确定的“卫星”以及“彗星”搜索机制，可提高系统“星球”变异的能力，能使算法跳出局部最优点，提高系统的全局寻优能力。

相比于同样运用了螺旋搜索机制的 WOA 算法，行星搜索算法的收敛精度有了进一步提高，说明本文所提出的局部螺旋搜索机制，可加强系统“星球”在最优点附近的寻优能力，使算法的局部搜索能力上得到提升。而相比于寻优效果较强 SSA 算法，行星搜索算法在测试函数下均可找

表 2 与其他算法的性能比较

函数	指标	行星搜索算法	PSO	GA	MFO	WOA	SSA
	Mean	0.00×10^{0}	1.89×10^{-1}	6.82×10^{2}	9.79×10^{1}	5.29×10^{-113}	3.68×10^{-6}
f1	Std	0.00×10^{0}	8.34×10^{-2}	5.07×10^{2}	4.07×10^{1}	4.76×10^{-113}	3.25×10^{-6}
	Best	0.00×10^{0}	9.60×10^{2}	1.19×10^{2}	5.27×10^{1}	$9.62\times10^{12}1$	6.79×10^{8}
	Mean	0.00×10^{0}	2.35×10^{0}	2.67×10^{1}	2.57×10^{0}	3.67×10^{-65}	1.22×10^{-4}
f2	Std	0.00×10^{0}	7.75×10^{-1}	9.66×10^{0}	1.91×10^{0}	3.30×10^{-65}	5.97×10^{-5}
	Best	0.00×10^{0}	1.49×10^{0}	1.60×10^{1}	4.50×10^{1}	9.84×10^{69}	5.54×10^{5}
	Mean	0.00×10^{0}	7.77×10^{0}	3.99×10^{3}	2.67×10^{4}	1.36×10^{-21}	2.24×10^{-4}
f3	Std	0.00×10^{0}	4.08×10^{0}	2.82×10^{3}	1.62×10^{4}	1.22×10^{-21}	1.95×10^{-4}
	Best	0.00×10^{0}	3.24×10^{0}	8.63×10^{2}	8.68×10^{3}	8.69×10^{-25}	7.45×10^{-6}
	Mean	0.00×10^{0}	1.38×10^{1}	6.66×10^{1}	4.99×10^{1}	4.47×10^{-19}	9.26×10^{-5}
f4	Std	0.00×10^{0}	5.29×10^{0}	3.06×10^{1}	3.69×10^{1}	4.01×10^{-19}	8.26×10^{-5}
	Best	0.00×10^{0}	7.95×10^{0}	3.26×10^{1}	8.87×10^{0}	9.22×10^{-22}	9.13×10^{-7}
	Mean	5.36×10^{20}	4.50×10^{2}	1.27×10^{2}	1.33×10^{5}	2.87×10^{1}	7.63×10^{16}
f5	Std	4.82×10^{-20}	1.89×10^{2}	6.30×10^{1}	1.18×10^{5}	1.44×10^{1}	6.86×10^{-16}
	Best	0.00×10^{0}	2.40×10^{2}	5.72×10^{1}	1.55×10^{3}	1.27×10^{1}	5.95×10^{26}
	Mean	9.20×10^{10}	1.82×10^{1}	2.10×10^{3}	7.06×10^{1}	3.11×10^{1}	8.87×10^{7}
f6	Std	8.28×10^{-10}	7.94×10^{-2}	1.01×10^{3}	4.86×10^{1}	2.09×10^{-1}	7.96×10^{-7}
	Best	4.38×10^{-15}	9.40×10^{-2}	9.80×10^{2}	1.66×10^{1}	7.84×10^{-2}	2.15×10^{-9}
	Mean	1.20×10^{-4}	9.86×10^{-2}	2.74×10^{-2}	2.44×10^{-1}	1.05×10^{-3}	3.62×10^{-4}
f7	Std	1.07×10^{-4}	8.12×10^{-2}	1.21×10^{-2}	1.80×10^{-1}	5.06×10^{-4}	3.20×10^{-4}
	Best	5.90×10^{-7}	8.36×10^{-3}	1.40×10^{-2}	4.40×10^{-2}	4.86×10^{-4}	6.05×10^{-6}
	Mean	-1.26×10^{4}	-7.56×10^{3}	-3.46×10^{3}	-5.16×10^{3}	-6.87×10^{3}	-9.02×10^{3}
f8	Std	1.85×10^{1}	9.63×10^{2}	8.37×10^{2}	1.98×10^{3}	1.44×10^{3}	7.20×10^{2}
	Best	-1.26×10^{4}	-8.63×10^{3}	-4.39×10^{3}	-7.36×10^{3}	-8.47×10^{3}	-9.82×10^{3}
	Mean	0.00×10^{0}	1.17×10^{2}	2.31×10^{2}	2.09×10^{2}	0.00×10^{0}	1.30×10^{6}
f9	Std	0.00×10^{0}	3.13×10^{1}	6.30×10^{1}	1.65×10^{2}	0.00×10^{0}	1.16×10^{6}
	Best	0.00×10^{0}	8.27×10^{1}	1.61×10^{2}	2.59×10^{1}	0.00×10^{0}	9.40×10^{-9}
	Mean	8.88×10^{16}	3.18×10^{0}	1.95×10^{1}	4.79×10^{0}	4.44×10^{15}	1.62×10^{7}
f10	Std	7.99×10^{-16}	1.91×10^{0}	1.08×10^{1}	2.92×10^{0}	3.91×10^{-15}	1.46×10^{-7}
	Best	3.55^{-23}	1.06×10^{0}	7.47×10^{0}	1.55×10^{0}	9.21×10^{-17}	8.36×10^{-12}
	Mean	0.00×10^{0}	1.13×10^{0}	5.09×10^{1}	1.92×10^{0}	0.00×10^{0}	4.73×10^{-8}
f11	Std	0.00×10^{0}	2.75×10^{-1}	1.17×10^{1}	9.45×10^{-1}	0.00×10^{0}	4.26×10^{-8}
	Best	0.00×10^{0}	8.23×10^{-1}	3.79×10^{1}	8.70×10^{-1}	0.00×10^{0}	9.63×10^{-14}
	Mean	2.03×10^{-6}	6.31×10^{-2}	2.76×10^{0}	9.05×10^{2}	9.64×10^{-3}	1.17×10^{-5}
f12	Std	1.83×10^{6}	2.78×10^{2}	1.43×10^{0}	7.25×10^{2}	8.08×10^{3}	9.90×10^{6}
	Best	5.98×10^{-12}	3.22×10^{-2}	1.17×10^{0}	9.90×10^{1}	6.61×10^{-4}	7.00×10^{-7}

到最优值,且搜索精度比 SSA 算法更高,说明利用螺旋搜索机制的行星搜索算法,可满足群体优化问题的求解。

行星搜索算法相比于 PSO、WOA 等其他算法,由于迭代“星球”的步长是跟“星球”与当前迭代步中“恒星”距离成正比,距离较远的“星球”能以较大步长进行寻优,加强其寻找全局最优点的能力,对距离较近的“星球”,能以较短步长寻优,以避免在最优点附近出现振荡,所以其收敛速度较快,可在迭代次数较少时就得到理想解。同时,依托局部螺旋加强算法在局部寻优方面的能力,从图 2(e)可看出,当寻优过程接近全局最优解时,局部螺旋搜索机制可使算法的局部寻优能力得到加强。通过设立“彗星”对“星球”进行变异操作,从图 2(e)~(g)可看到,本文提出的行星搜索算法的折点较多,说明其还具有较强的跳出局部最优的能力。

综上所述,本文提出的行星搜索算法,可有效提升算法收敛精度,比其他算法有更优越的寻优效果。

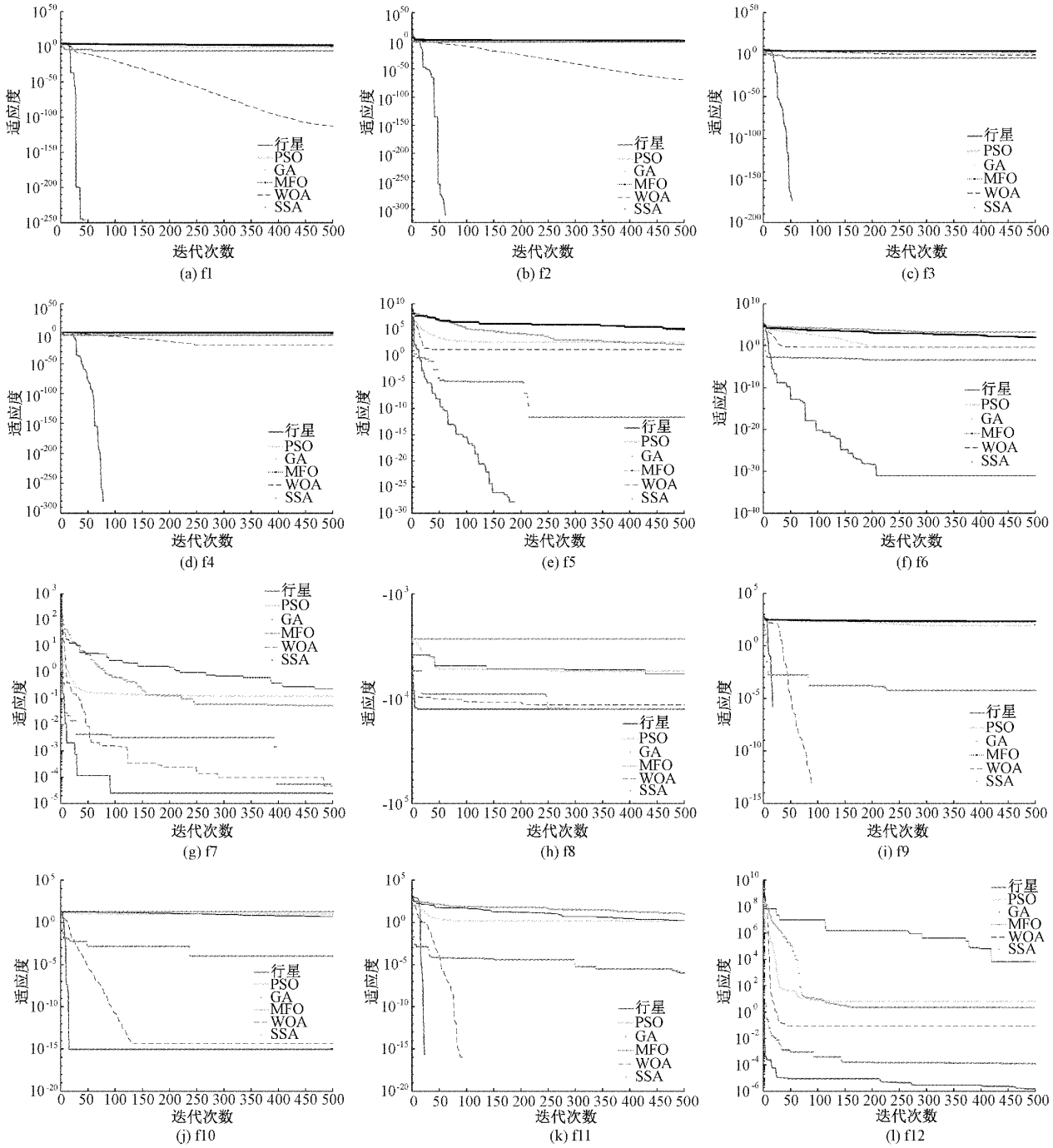

图 2　收敛曲线对比效果图

3　结　　论

本文基于螺旋搜索机制全局搜索能力较强的优点，加入变异操作，并针对其局部搜索能力较差提出在其搜索范围较小时采用局部螺旋搜索提高其搜索精度，提出行星搜索算法。经验证行星搜索算法对测试函数进行收敛性分析的效果显著，与其他同样应用螺旋搜索机制的飞蛾扑火算法以及鲸鱼搜索算法相比可以搜索到收敛精度更高的全局最优点。本文所提算法尚处于理论研究，仅利用测试函数进行了算法性能测试，将其应用于各种工程学问题的效果还有待验证。

参考文献

[1]　MIRJALILI S. Moth-flame optimization algorithm：A novel nature-inspired heuristic paradigm [J]. Knowledge-Based Systems，2015，89.

[2]　MIRJALILI S，LEWIS A. The whale optimization

algorithm[J]. Advances in Engineering Software, 2016,DOI:10.1016/j.advengsoft.2016.01.008.

[3] 许学彬,陈博桓,赵楠楠,等.基于 GA-BP 的改进高斯均值区域去噪技术[J].电子测量与仪器学报,2022,36(2):107-113.

[4] 谭智钢,程静,王维庆.基于改进鲸鱼优化算法的微网系统能量优化管理[J].科学技术与工程,2021,21(32):13714-13720.

[5] 李赛宇.基于 Iterative 映射和非线性拟合的鲸鱼优化算法[J].重庆大学学报,2022:1-17.

[6] 李安东,刘升.混合策略改进鲸鱼优化算法[J].计算机应用研究,2022-02-20:1-8.

[7] 尚猛,康建英,曹峻玮,等.基于改进鲸鱼优化算法的物流配送中心选址策略[J].计算机应用与软件,2019,36(6):254-259.

[8] 张达敏,徐航,王依柔,等.嵌入 Circle 映射和逐维小孔成像反向学习的鲸鱼优化算法[J].控制与决策,2021,36(5):1173-1180.

[9] SHUXIAN D. Improved whale optimization algorithm based on the tent chaotic mapping and nonlinear convergence factor[J]. Journal of Physics: Conference Series,2020,1682(1).

[10] 李国全,高建宇,白天宇,等.基于 SVM 与改进型乌鸦搜索算法的风电功率预测方法[J].国外电子测量技术,2022,41(2):40-45.

[11] 邵亚璐,王鉴,赵飞飞,等.基于混沌加权粒子群算法的频率计权设计[J].电子测量技术,2022,45(5):75-79.

[12] 王震,刘瑞敏,朱阳光,等.一种求解 TSP 问题的改进遗传算法[J].电子测量技术,2019,42(23):91-96.

[13] 邹浩,李维刚,李阳,等.基于混沌收敛因子和惯性权重的鲸鱼优化算法[J].武汉科技大学学报,2022,45(4):304-313.

[14] 薛建凯.一种新型的群智能优化技术的研究与应用[D].上海:东华大学,2020.

[15] 王雨虹,王志中,付华,等.多策略改进麻雀算法与 BiLSTM 的变压器故障诊断研究[J].仪器仪表学报,2022,43(3):87-97.

作者简介

司书千,硕士研究生,主要研究方向为计算机启发式算法优化、冷热电联产微电网优化运行等。

E-mail:sdut_ssq@163.com