Journal of System Simulation

Volume 34 | Issue 3 Article 7

3-22-2022

An Improved Atomic Search Algorithm

Jianfeng Li
School of Electrical and Electronic Engineering, Harbin University of Science and Technology, Harbin 150000, China;, 3021411795@qq.com

Di Lu
School of Electrical and Electronic Engineering, Harbin University of Science and Technology, Harbin 150000, China;, ludizeng@hrbust.edu.cn

Hexiang Li
School of Electrical and Electronic Engineering, Harbin University of Science and Technology, Harbin 150000, China;

An Improved Atomic Search Algorithm

Abstract

Abstract: The atom search algorithm (ASO) is a new optimization algorithm proposed by imitating the movement of atoms in the natural world. *An improved atomic search algorithm (IASO) is proposed to address the problems of prematureness and slow convergence of ASO in solving complex functions. IASO adds the binding force generated by the historical optimal solution of individual atoms to correct the acceleration of ASO and enhance the global search capability. The two multiplier coefficients are adaptively updated to coordinate the algorithm's global search and local development capabilities. The Gaussian mutation strategy is used to re-update the atomic position and improve the ability to jump out of precocity.* Carrying out simulation experiments on 14 benchmark functions and comparing other algorithms, IASO shows superior performance in terms of convergence speed and convergence accuracy.

Keywords

atomic optimization algorithm, function optimization, self-adaptation, Gaussian mutation, convergence accuracy, test function

第 34 卷第 3 期 系统仿真学报© Vol. 34 No. 3
2022 年 3 月 Journal of System Simulation Mar. 2022

一种改进的原子搜索算法

李建锋，卢迪*，李贺香
(哈尔滨理工大学 电气与电子工程学院，黑龙江 哈尔滨 150000)

摘要：原子搜索算法（atom search algorithm，ASO）是模仿自然界中原子运动而提出的一种新型优化算法，针对ASO在求解复杂函数时存在易早熟及收敛速度慢的问题，*提出了一种改进ASO算法（improved atomic search algorithm，IASO）。IASO加入了原子个体历史最优解产生的约束力来修正ASO的加速度，增强全局搜索能力。自适应更新2个乘数系数来协调算法的全局搜索和局部开发能力。适时采用高斯变异策略来重新更新原子位置，提高跳出早熟的能力。*对14个基准函数进行仿真实验，对比其他算法，IASO在收敛速度、收敛精度方面表现出优越的性能。

关键词：原子优化算法；函数优化；自适应；高斯变异；收敛精度；测试函数
中图分类号：TP301.6；TP391.9　　文献标志码：A　　文章编号：1004-731X(2022)03-0490-13
DOI: 10.16182/j.issn1004731x.joss.20-0824

An Improved Atomic Search Algorithm

Li Jianfeng, *Lu Di**, *Li Hexiang*
(School of Electrical and Electronic Engineering, Harbin University of Science and Technology, Harbin 150000, China)

Abstract: The atom search algorithm (ASO) is a new optimization algorithm proposed by imitating the movement of atoms in the natural world. *An improved atomic search algorithm (IASO) is proposed to address the problems of prematureness and slow convergence of ASO in solving complex functions. IASO adds the binding force generated by the historical optimal solution of individual atoms to correct the acceleration of ASO and enhance the global search capability. The two multiplier coefficients are adaptively updated to coordinate the algorithm's global search and local development capabilities. The Gaussian mutation strategy is used to re-update the atomic position and improve the ability to jump out of precocity.* Carrying out simulation experiments on 14 benchmark functions and comparing other algorithms, IASO shows superior performance in terms of convergence speed and convergence accuracy.

Keywords: atomic optimization algorithm; function optimization; self-adaptation; Gaussian mutation; convergence accuracy; test function

引言

在当今世界快速发展的过程中，最优化问题一直是人类在生产生活中不可避免的主流研究方向，人们在对优化问题的探索过程中受到自然灵感的启示而提出了许多元启发式算法[1]，它们模仿自然界物理现象、生物的行为或进化过程来解决现实中的优化问题[2]。基于进化方向的有遗传算法(genetic algorithm，GA)[3]、差分进化算法(differential evolution，DE)[4]；基于群智能行为方向的有灰狼算法(grey wolf optimizer algorithm，GWO)[5]、鲸鱼算法(whale optimization algorithm，WOA)[6]、狮群算法(loin swarm optimization，LSO)[7]；基于物理现象的有万有引力搜索算法(gravitational search algorithm，GSA)[8]、黑洞(black-hole，BH)[9]算法。

收稿日期：2020-10-26　　修回日期：2021-02-06
第一作者：李建锋(1994-)，男，硕士生，研究方向为机器人路径规划。Email：3021411795@qq.com
通讯作者：卢迪(1971-)，女，博士，教授，研究方向为数据融合、图像处理。Email：ludizeng@hrbust.edu.cn

尽管出现了许多新的优化方法，没有哪种优化算法可以解决所有的优化问题。遗传算法存在过早收敛的弊端[10]；差分进化算法存在局部最优和搜索停滞的问题[11]；万有引力搜索算法则有求解精度不高、早熟现象的存在[12]；飞蛾-火焰优化算法(moth-flame optimization algorithm，MFO)的单一搜索机制不适合解决复杂问题[13]；鲸鱼算法则在解决大规模优化问题时准确度不高[14]。

原子搜索算法(atom search algorithm，ASO)[15]是根据原子的运动规律，通过种群中各个原子之间的相互作用力来指导群体进行智能优化搜索。算法结构简单，参数较少。受粒子群算法记忆思想[16]的启示，提出了一种的改进原子搜索算法(improved atomic search algorithm, IASO)，进一步提升了算法的整体性能。IASO 引入了原子个体历史最优解产生的共价键约束力，利用原子种群中的信息交换和对自身行为的认知，通过优良信息来更新原子个体的加速度，增加了新生个体的多样性。为了平衡全局最优解和个体最优解的权重，对 2 个系数因子进行了自适应更新，提高了算法的收敛速度。最后，通过相邻 3 次迭代结果的变异系数大小来判断原子群体的收敛情况，如果变异系数过小，则采用高斯扰动重新更新原子位置，避免算法陷入早熟。通过求解 14 个函数优化问题验证其有效性，对比 GWO，WOA，MFO，ASO，IASO，该算法在稳定性和收敛精度方面均表现优异。

1 ASO 算法原理

ASO 是根据分子动力学中原子的物理运动规律建立的算法模型，根据牛顿第二定律，如果 F_i 是作用在第 i 个原子上的相互作用力，而 G_i 是作用在第 i 个原子上的约束力，该原子的质量为 m_i，那么第 i 个原子的加速度为

$$a_i = \frac{F_i + G_i}{m_i} \tag{1}$$

下面对 F_i，G_i，m_i 3 个值的求解进行阐述。

1.1 相互作用力 F_i^d

相互作用力 F_i^d 表示周围原子对当前原子 i 的作用力之和，其求解公式为

$$F_i^d(t) = \sum_{j \in K_{\text{best}}} rand_j F_{ij}^d(t) \tag{2}$$

式中：t 为当前迭代次数；$rand_j$ 为[0,1]上的随机数；d 为原子所在的维数；K_{best} 为对原子 i 产生作用力的原子的集合；$F_{ij}^d(t)$ 为第 t 次迭代中第 j 个原子对原子 i 的伦纳德·琼斯(L-J)势作用力[17]。

K_{best} 的定义如式(3)所示：

$$K_{\text{best}}(t) = N - (N-2) \cdot \sqrt{\frac{t}{T}} \tag{3}$$

式中：N 为原子总个数；T 为总迭代次数。

$F_{ij}^d(t)$ 定义如式(4)所示：

$$F_{ij}^d(t) = -\eta(t)\left[2\left(h_{ij}(t)\right)^{13} - \left(h_{ij}(t)\right)^{7}\right] \tag{4}$$

式中：$\eta(t)$ 为用于调整排斥区域或吸引区域的深度函数，它的定义如式(5)所示：

$$\eta(t) = -\alpha\left(1 - \frac{t-1}{T}\right)^3 \mathrm{e}^{-\frac{20t}{T}} \tag{5}$$

式中：α 为深度权重。

h_{ij} 在文献[15]中的标准定义如式(6)所示；

$$h_{ij} = \frac{r_{ij}(t)}{\sigma(t)} \tag{6}$$

式中：σ 为长度尺度，表示碰撞直径；r_{ij} 为 2 个原子之间的欧几里得距离。

$r_{ij}(t)$ 和 $\sigma(t)$ 的定义如式(7)所示：

$$\begin{aligned} r_{ij}(t) &= \left\| x_i(t), x_j(t) \right\|_2 \\ \sigma(t) &= \left\| x_i(t), \frac{\sum_{j \in K_{\text{best}}(t)} x_j(t)}{K_{\text{best}}(t)} \right\|_2 \end{aligned} \tag{7}$$

式中：x_i 和 x_j 分别为原子 i 和原子 j 的位置。

实际上 h_{ij} 的值是在一定的范围内取值，在文献[15]中，为了改善勘探范围，作者将表现为斥力的 h 的下限设置为 $h_{\min}$=1.1，上限设置为 $h_{\max}$=2.4，并且在 $h_{\min}$ 的基础上增加了一个正弦扰动函数 $g(t)$，具体定义如式(8)所示；

第 34 卷第 3 期 系统仿真学报 Vol. 34 No. 3
2022 年 3 月 Journal of System Simulation Mar. 2022

$$h_{ij}(t)=\begin{cases}h_{\min}+g(t) & \dfrac{r_{ij(t)}}{\sigma(t)}<h_{\min}\\ \dfrac{r_{ij(t)}}{\sigma(t)} & h_{\min}\leqslant\dfrac{r_{ij(t)}}{\sigma(t)}\leqslant h_{\max}\\ h_{\max} & \dfrac{r_{ij(t)}}{\sigma(t)}>h_{\max}\end{cases} \tag{8}$$

式中：$g(t)$的定义如式(9)所示：

$$g(t)=0.1\sin\left(\frac{\pi}{2}\cdot\frac{t}{T}\right) \tag{9}$$

1.2 共价键约束力 G_i^d

在分子动力学中的几何约束在原子运动中也起着重要作用。在ASO中，为了突出种群最佳原子的引导作用，假设每个原子与种群最佳原子具有一个共价键，因此每个原子都要受到最佳原子的约束力作用，其求解公式如式(10)所示。

$$G_i^d=\beta e^{\frac{-20t}{T}}\left(x_{\text{best}}^d(t)-x_i^d(t)\right) \tag{10}$$

式中：β为系数因子；$x^d{}_{\text{best}}(t)$为第t次迭代中种群最佳原子位置；$x_i^d(t)$为原子第t次迭代的当前位置。

1.3 原子加速度 a_i^d

结合相互作用力和几何约束力，在时间t的第i个原子在第d维上的加速度如式(11)所示：

$$a_i^d=\frac{F_i^d+G_i^d}{m_i(t)}=$$
$$-\alpha\left(1-\frac{t-1}{T}\right)^3 e^{-\frac{20t}{T}}\sum_{j\in K_{\text{best}}}\frac{rand_j\left[2\left(h_{ij}(t)\right)^{13}-\left(h_{ij}(t)\right)^7\right]}{m_i(t)}+$$
$$\beta e^{-\frac{20t}{T}}\frac{\left(x_{\text{best}}^d(t)-x_i^d(t)\right)}{m_i} \tag{11}$$

第t次迭代中第i个原子的质量$m_i(t)$由当前种群个体的适应度值大小决定，其定义如式(12)所示:

$$M_i(t)=e^{-\frac{f_i(t)-f_{\min}(t)}{f_{\max}(t)-f_{\min}(t)}}$$
$$m_i(t)=\frac{M_i(t)}{\sum_{j=1}^{N}M_i(t)} \tag{12}$$

式中：$f_i(t)$表示第i个原子的适应度值；$f_{\max}(t)$和$f_{\min}(t)$分别表示原子种群中最大适应度值和最小适应度值。

1.4 位置 $x_i^d(t)$更新

在每一次迭代过程中，根据得到的加速度更新原子i的速度和位置，更新公式为如式(13)所示：

$$v_i^d(t+1)=rand_i^d v_i^d(t)+a_i^d(t)$$
$$x_i^d(t+1)=x_i^d(t)+v_i^d(t+1) \tag{13}$$

式中：v_i^d为原子的速度；x_i^d为原子位置；$rand_i^d$为[0,1]之间的随机数。

ASO算法的流程如图1所示。

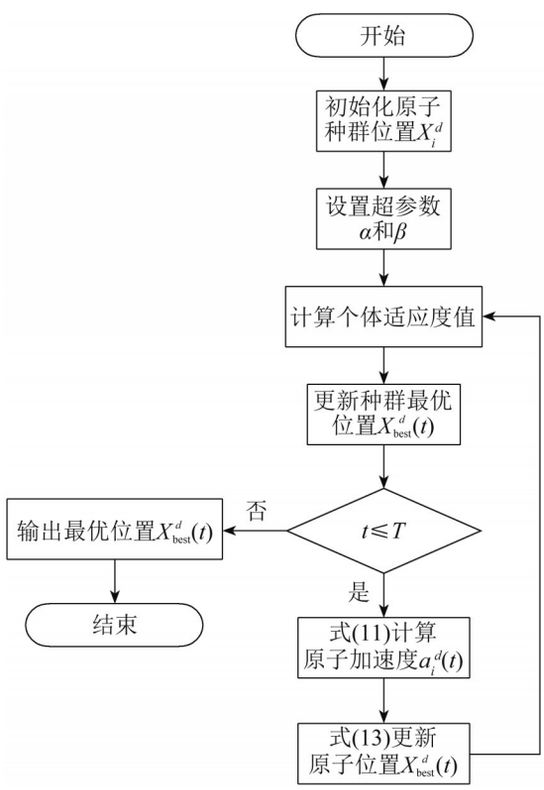


图1 ASO算法实现流程图
Fig. 1 ASO algorithm implementation flowchart

2 改进的原子搜索算法（IASO）

2.1 引入新的共价键约束力 P_i^d

ASO每次迭代中种群最佳原子对周围原子存在一个共价键约束力G_i^d，由式(10)可知它是种群最佳原子位置与当前原子位置的差值，这种机制提高了全局最优信息对当前原子的引导作用，加强了原子个体之间的协同合作。为了更好地更新原子加速度方向，本文引入原子对自身的认知，加入原子个

体历史最优解——单个原子在历次迭代中的最佳位置叫原子个体历史最优解。在每一次迭代中的个体原子位置都与原子个体历史最优位置进行比较，如果当前个体原子位置优于它的个体历史最优位置，则把当前个体原子位置更新为个体历史最优位置；相反，则不更新。假设每个原子个体在迭代中产生的个体历史最优解对周围原子也存在一个共价键约束力，因此原子的加速度受种群最佳原子产生的共价键约束力和个体历史最优原子产生的共价键约束力共同影响。原子可以利用种群信息和自身经验，通过信息融合更新加速度，提高了算法的全局探索能力。原子个体历史最优解产生的共价键约束力定义为P_i^d，其表达式如式(14)所示：

$$P_i^d=\lambda e^{\frac{-20t}{T}}\left(x_p^d(t)-x_i^d(t)\right) \tag{14}$$

式中：λ为系数因子；$x_p{}^d$代表第t次迭代中原子i的历史最优位置。

2.2 自适应调整系数因子

引入原子个体历史最优解产生的共价键约束力P_i^d之后，原子加速度a_i^d的公式更新为如式(15)所示：

$$a_i=\frac{F_i^d+G_i^d+P_i^d}{m_i}=$$
$$-\alpha\left(1-\frac{t-1}{T}\right)^3 e^{-\frac{20t}{T}}\sum_{j\in K_{\text{best}}}\frac{rand_j\left[2\left(h_{ij}(t)\right)^{13}-\left(h_{ij}(t)\right)^7\right]}{m_i}+$$
$$\beta e^{-\frac{20t}{T}}\frac{\left(x_{\text{best}}^d(t)-x_i^d(t)\right)}{m_i}+\lambda e^{-\frac{20t}{T}}\frac{\left(x_p^d(t)-x_i^d(t)\right)}{m_i} \tag{15}$$

式中：β和λ都是超参数，对算法的寻优速度和优化结果有着重要的影响。可以看出，β代表了原子向全局历史最优解运动能力的权重；λ代表了原子对自己的认可程度。这2个参数反映了对种群信息和个体信息的接受程度，不同阶段需要设置不同大小。在算法搜索初期，为了增强原子的全局搜索能力，让λ取较大的值，β取较小的值，这样可以使原子在整个可行空间进行搜索。在搜索后期，λ取较小的值，β取较大的值，可以使原子快速收敛于最优值。所以随着迭代次数的增加，λ逐渐变大，β逐渐变小，基于此本文对两2个参数采取自适应更新策略，平衡全局信息与局部信息，提高信息利用率。β和λ的自适应更新公式为

$$\beta=\beta_{\max}+\left(\beta_{\min}-\beta_{\max}\right)\cdot\frac{T-t}{T-1}$$
$$\lambda=\lambda_{\min}+\left(\lambda_{\max}-\lambda_{\min}\right)\cdot\frac{T-t}{T-1} \tag{16}$$

式中：$\beta_{\min}$，λ_{min}分别为参数的最小值；$\beta_{\max}$，$\lambda_{\max}$为最大值。可知β和λ都在[0，1]之间变化。

2.3 高斯变异策略

ASO优化算法在更新原子全局最优位置的过程中，原子通常会表现出早熟的现象，整个原子种群进入搜索停滞，陷入局部极值。为了加快原子的收敛速度，提高原子跳出早熟的能力，将高斯变异策略[18]引入IASO算法法的原子位置更新中，加入高斯变异因子对原子位置进行扰动，使之产生新的位置继续进行更新。在对原子位置进行高斯变异之前，首先进行变异系数判断，变异系数是衡量一组数据离散程度的一个归一化量度，变异系数的定义为如下式(17)所示：

$$c_v=\frac{\delta}{\mu} \tag{17}$$

式中：c_v为变异系数；δ为标准差；μ为平均值。

IASO算法提前设置一个变异系数阈值c_{std}，为了保持算法的效率，变异系数不宜过大，本文设置c_{std}=0.1，算法前期迭代收敛性一般较好，迭代中后期易陷入局部最优解，在算法进行T/5次迭代之后，再根据变异系数适时采用高斯变异。对相邻3次的迭代结果求取高斯变异系数$stdY(t)$，其求解如式(18)所示：

$$stdY(t)=\frac{\sqrt{\sum_{i=t-2}^{t}\left(y_{\text{best}}^i-meanY(t)\right)^2}}{meanY(t)} \tag{18}$$

式中：t为当前迭代次数；y^i_{best}为第i次迭代的最优解；$meanY(t)$为3次迭代结果y_{best}的平均值，其定义如式(19)所示：

$$meanY(t)=\frac{\sum_{i=t-1}^{t}y_{\text{best}}^i}{3} \tag{19}$$

第 34 卷第 3 期 系统仿真学报 Vol. 34 No. 3
2022 年 3 月 Journal of System Simulation Mar. 2022

如果 $stdY(t)>cstd$, 说明每次迭代的最优解 y_{best} 变化较大，算法的收敛性仍然较好，此时对原子位置更新不采用高斯扰动；如果 $stdY(t)<cstd$,则说明迭代结果 y_{best} 变化不大，算法陷入了局部停滞，此时对原子的位置进行高斯变异扰动更新，提高原子的多样性，跳出局部极值。加入高斯变异后的原子位置更新公式如式(20)所示：

$$\overline{x_i^d(t)} = r_1 \otimes x_i^d(t) + Gaussian \otimes \left(x_{best}^d(t) - x_i^d\right) \quad (20)$$

式中：r_1 为服从 0-1 之间均匀分布的随机数；$Gaussian\sim N(0,1)$，$x_i^d(t)$ 为第 t 次迭代原子位置；$\overline{x_i^d}(t)$ 为加高斯扰动后的原子位置。IASO 算法流程如图 2 所示。

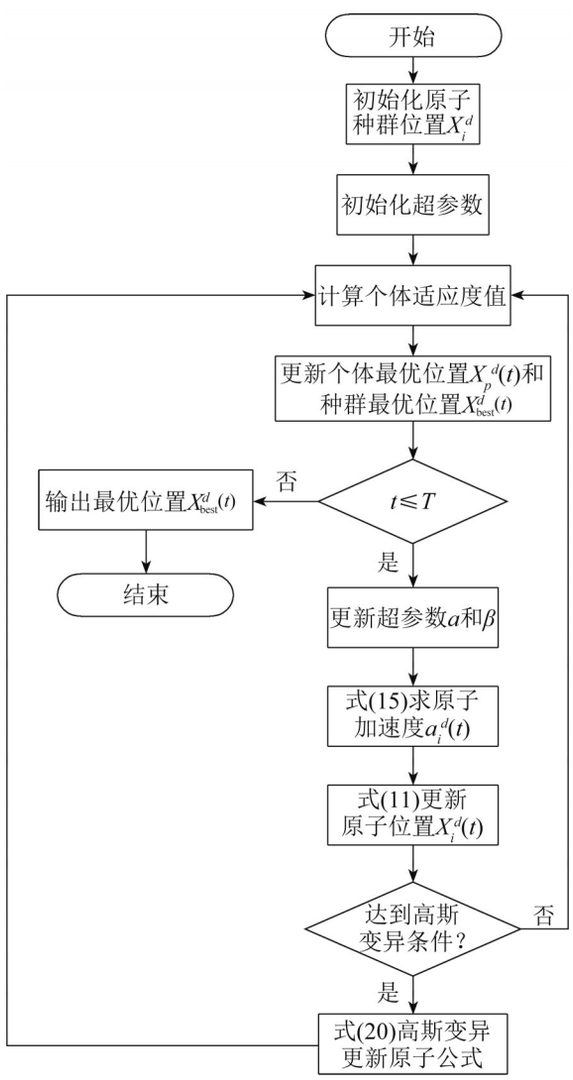


图 2　IASO 算法实现步骤图
Fig. 2　IASO algorithm implementation flowchart

3　仿真分析

3.1　实验环境和参数设置

为了验证 IASO 算法的性能，分别与 GWO，MFO，WOA，ASO在同一仿真环境下实验，环境参数为 AMD A4-6210 APU with 4G RAM，1.8 GHz，Window10 64 位操作系统，仿真软件为 Matlab 2016b。算法参数设置如表 1 所示。

表 1　算法参数设置表
Table 1　Algorithm parameter setting table

算法	参数
GWO	收敛因子 α 由 2 线性减少至 0
MFO	对数螺旋线形状常数 b=1,t 由-1 线性减少到-2
WOA	收敛因子 α 由 2 线性减少至 0，螺旋常数 b=1
ASO	深度权重 α=50，乘数系数 β=0.2
IASO	深度权重 α=50,系数因子 β_{min}=0.1，λ_{min}=0.1，β_{max}=0.9，λ_{max}=0.9，变异系数 c_{std}=0.1

GWO 参数参考文献[5]设置，MFO 参数参考文献[6]设置，WOA 参数参考文献[13]设置，ASO 参数参考文献[15]设置。

3.2　标准测试函数

实验选取了 14 个基准测试函数进行实验，选取的 14 个基准函数具有多样性，这样得到的测试结果可以更客观、更全面地反映算法寻优能力。函数表达式(Benchmark function)维数(Dim)、函数的取值范围(Range)及其理论值(f_{min})相关信息如表 2 所示。其中 f_1~f_6 是高维单峰函数，单峰函数在定义域内只有一个极值点，主要用于测试算法的全局收敛性。f_7~f_{11} 是高维多峰函数，多峰函数则是在定义域内有多个局部极值点，用来检测算法跳出局部最优和避免早熟的能力。f_{12}~f_{14} 低维多峰函数，同高维多峰测试函数一样，低维多峰测试函数也可用于检验算法全局搜索的性能。

表2 基准测试函数

Table 2 Benchmark test functions

Benchmark function	Dim	Feature	Range	$f_{\min}$
$f_1(x)=\sum_{i=1}^{n}x_i^2$	30	[−100,100]	[−100,100]	0
$f_2(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	30	[−10,10]	[−10,10]	0
$f_3(x)=\sum_{i=1}^{n}\left(\sum_{j=1}^{i}x_j\right)^2$	30	[−100,100]	[−100,100]	0
$f_4(x)=\max_i\{\lvert x_i\rvert,1\leqslant i\leqslant D\}$	30	[−100,100]	[−100,100]	0
$f_5(x)=\sum_{i=1}^{n}(\lvert x_i+0.5\rvert)^2$	30	[−100,100]	[−100,100]	0
$f_6(x)=\sum_{i=1}^{n}x_i^4+rand(0,1)$	30	[−1.28,1.28]	[−1.28,1.28]	0
$f_7(x)=\sum_{i=1}^{n}-x_i\sin\left(\sqrt{\lvert x_i\rvert}\right)$	30	[−500,500]	[−500,500]	−418.982 9×5
$f_8(x)=\sum_{i=1}^{n}\left[x_i^2-10\cos(2\pi x_i)+10\right]$	30	[−5.12,5.12]	[−5.12,5.12]	0
$f_9(x)=-20\exp\left[-0.2\sqrt{\frac{1}{n}\sum_{i-1}^{n}x_i^2}-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i\right)\right]+20+\mathrm{e}$	30	[−32,32]	[−32,32]	0
$f_{10}(x)=\frac{1}{4\,000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	30	[−600,600]	[−600,600]	0
$f_{11}(x)=0.1\{\sin^2(3\pi x_1)+\sum_{i-1}^{n}(x_i-1)^2[1+\sin^2(3\pi x_1+1)]+(x_n-1)^2[1+\sin^2(2\pi x_n)]\}+\sum_{i=1}^{n}u(x_i,5,100,4)$	30	[−50,50]	[−50,50]	0
$f_{12}(x)=\left(\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right)^{-1}$	2	[−65,65]	[−65,65]	1
$f_{13}(x)=\sum_{i=1}^{11}\left[a_i-\frac{x_1(b_i^2+b_ix_2)}{b_i^2+b_ix_3+x_4}\right]^2$	4	[−5,5]	[−5,5]	0.000 30
$f_{14}=-\sum_{i=1}^{7}\left[(\boldsymbol{X}-\boldsymbol{a}_i)(\boldsymbol{X}-\boldsymbol{a}_i)^{\mathrm{T}}+C_i\right]^{-1}$	4	[0.10]	[0.10]	−10.402 8

3.3 算法的有效性分析

3.3.1 共价键约束力 P_i^d

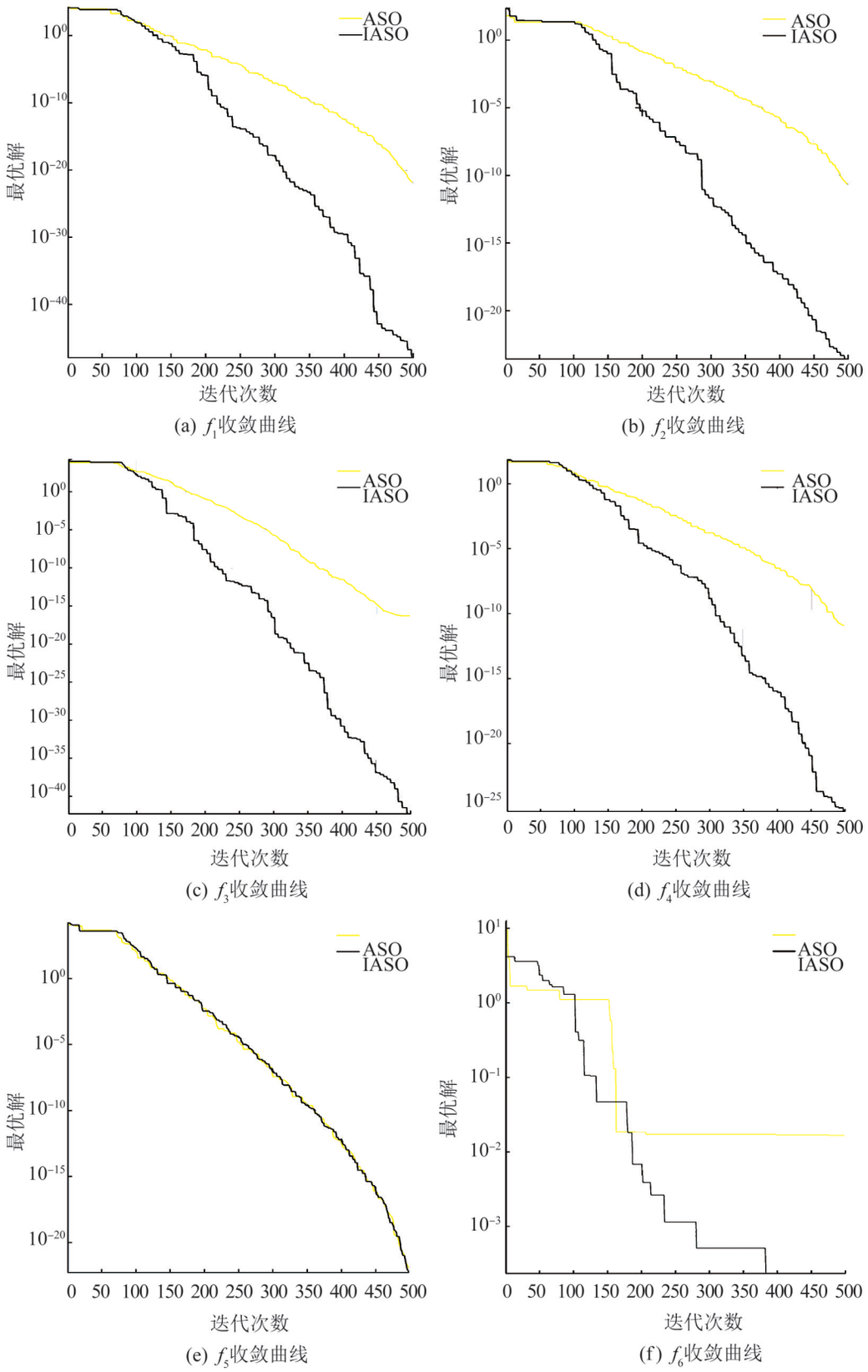
引入共价键约束力P_i^d之后，不仅原子有了记忆思想，原子加速度更新的可行空间变大。而且产生了新的超参数λ，它是衡量原子个体历史最优解在更新加速度时的所占权重大小。本算法对超参数β和λ进行了自适应更新，这2个改进主要提高算法的全局收敛性，为此用f_1~f_6是高维单峰函数来验证此改进的有效性。对比算法为ASO, 群规模设为N=30，迭代次数固定为T=500，比较算法的收敛精度，以30次运算的平均值来刻画收敛曲线。收敛曲线如图3所示。

由图3可以看出，除了f_5函数以外，改进的算法在其他单峰函数的收敛速度均表现出优越的性能。说明此改进是有效的。

3.3.2 高斯变异策略

高斯变异主要增强算法跳出局部最优能力，本节在ASO的基础上只加入高斯变异扰动，测试高维多峰函数f_7~f_{11}来验证改进的有效性，参数同3.3.1节，收敛曲线如图4所示。

第 34 卷第 3 期 系统仿真学报 Vol. 34 No. 3
2022 年 3 月 Journal of System Simulation Mar. 2022



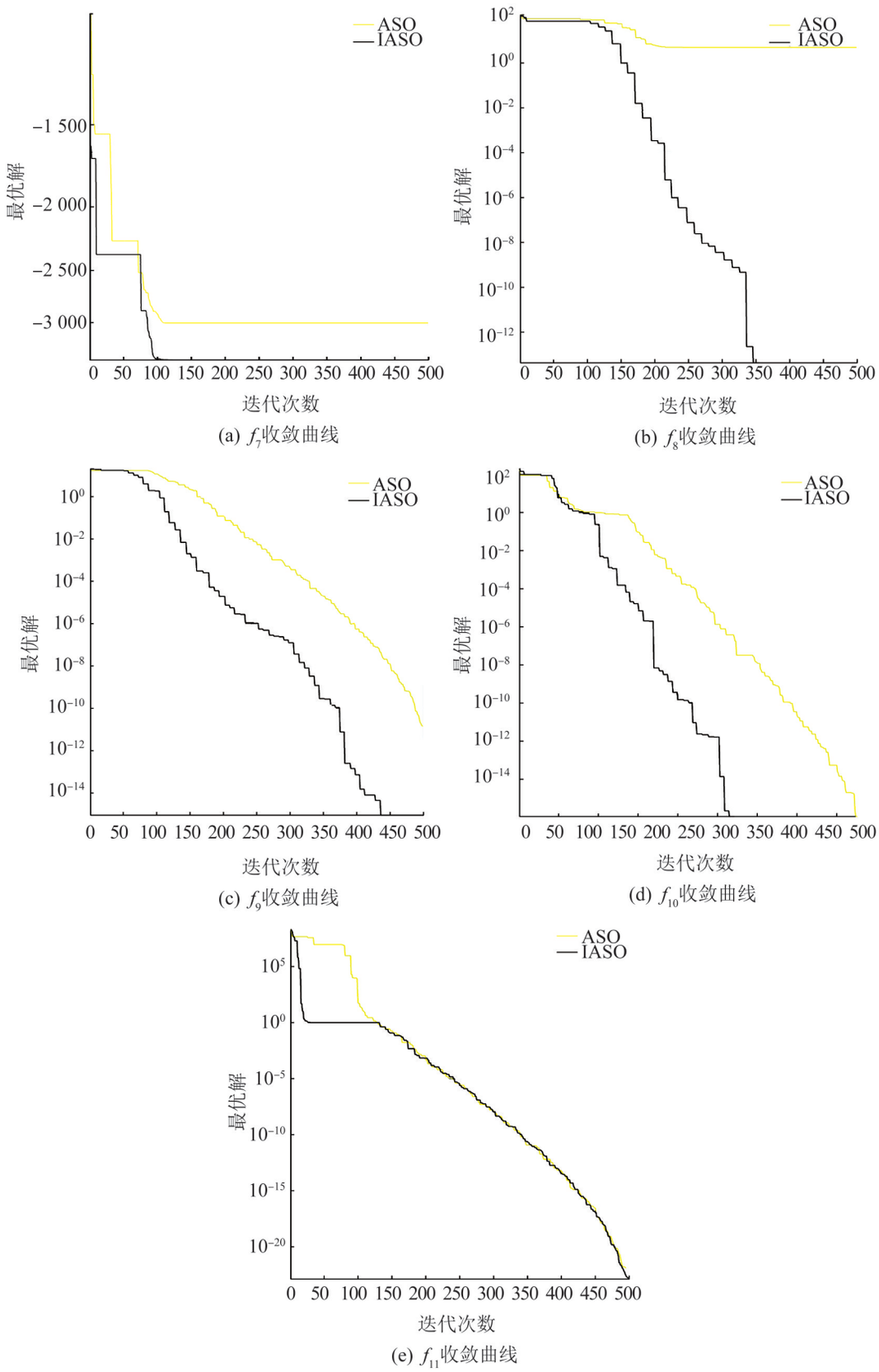
(a) f_1收敛曲线 (b) f_2收敛曲线

(c) f_3收敛曲线 (d) f_4收敛曲线

(e) f_5收敛曲线 (f) f_6收敛曲线

图3 单峰函数的收敛曲线

Fig. 3 Convergence curve of unimodal function



(a) f_7收敛曲线　(b) f_8收敛曲线

(c) f_9收敛曲线　(d) f_{10}收敛曲线

(e) f_{11}收敛曲线

图4　多峰函数的收敛曲线

Fig. 4　Convergence curve of multimodal function

https://dc-china-simulation.researchcommons.org/journal/vol34/iss3/7
DOI: 10.16182/j.issn1004731x.joss.20-0824

第 34 卷第 3 期 系统仿真学报 Vol. 34 No. 3
2022 年 3 月 Journal of System Simulation Mar. 2022

图 4 可以看出，除了 f_{11} 函数以外，加入高斯变异之后的算法针对多极值函数也表现出较强的收敛性。也证明了改进的有效性。

3.3.3 融合实验

将所有改进融合进行测试实验。与 GWO, MFO,WOA,ASO 4 算法进行对比试验。因为算法中每一次运行的结果是随机的，为了比较的公平性原则，所有算法的种群规模设为 N=30，每个对比算法在所有的测试函数中将独立运行 30 次，迭代次数固定为 T=500，比较算法的收敛精度。表 3 是测试函数的测试结果，其中 *Best*，*Mean*，*Worst* 和 *Std* 分别表示在 30 次独立实验中的最优解、平均解、最差解和运行结果的标准差，D 表示函数的维数，D=30。

表 3　测试函数实验结果
Table 3　Experimental results of test function

Function	Result	Algorithm				
		GWO	MFO	WOA	ASO	IASO
f_1	*Best*	**0**	0.000 1E-11	**0**	0.003 6E-21	**0**
	Worst	0.786 4E-58	0.144 1E-11	0.907 2E-80	0.100 0E-21	**0.166 5E-110**
	Mean	0.045 1E-58	0.028 5E-11	0.030 2E-80	0.034 6E-21	**0.005 6E-110**
	Std	0.146 0E-58	0.041 2E-11	0.165 6E-80	0.027 8E-21	**0.030 4E-110**
f_2	*Best*	0.000 3E-31	0.003 8E-7	**0**	0.008 0E-9	**0**
	Worst	0.365 6E-31	0.206 8E-7	0.171 2E-52	0.168 4E-9	**0.777 5E-80**
	Mean	0.054 3E-31	0.055 3E-7	0.016 9E-52	0.043 6E-9	**0.025 9E-80**
	Std	0.083 3E-31	0.053 7E-7	0.042 7E-52	0.032 0E-9	**0.141 9E-80**
f_3	*Best*	**0**	**0**	1.484 2	1.176 7E-17	**0**
	Worst	0.101 7E-27	0.005 1	6.981 9	0.007 8E-17	**0.285 1E-142**
	Mean	0.008 2E-27	0.008 3	4.309 4	0.000 4E-17	**0.018 1E-142**
	Std	0.023 4E-27	0.019 1	1.171 1	0.001 5E-17	**0.068 9E-142**
f_4	*Best*	**0**	0.001 6	0.002 5	0.070 3E-11	**0**
	Worst	0.196 2E-16	1.551 9	16.573 5	0.992 3E-11	**0.654 1E-72**
	Mean	0.021 7E-16	0.243 9	1.383 9	0.346 1E-11	**0.021 8E-72**
	Std	0.041 4E-16	0.296 7	3.447 6	0.249 7E-11	**0.119 4E-72**
f_5	*Best*	**0**	**0**	0.000 1	**0**	**0**
	Worst	0.252 2E-5	0.152 5E-12	0.010 3	**0.116 7E-22**	0.654 5E-22
	Mean	0.016 8E-5	0.236 0E-12	0.001 1	**0.033 8E-22**	0.205 1E-22
	Std	0.063 8E-5	0.033 3E-12	0.001 8E-2	**0.029 9E-22**	0.172 3E-22
f_6	*Best*	0.000 1E-3	0.003 3	0.013 4E-2	0.002 7	**0.263 3E-4**
	Worst	0.001 5E-3	0.024 4	0.003 0E-2	0.043 0	**3.147 1E-4**
	Mean	0.000 6E-3	0.009 3	0.002 9E-2	0.013 5	**1.204 0E-4**
	Std	0.000 4E-3	0.005 9	0.389 3	0.009 2	**0.745 0E-4**
f_7	*Best*	−3.182 7E+3	−3.833 0E+3	−4.189 6E+3	−3.538 4E+3	**−3.071 4E+3**
	Worst	−2.045 6E+3	−2.521 1E+3	−2.199 6E+3	−2.373 7E+3	**−2.886 5E+3**
	Mean	−2.702 4E+3	−3.213 9E+3	−3.467 7E+3	−2.783 6E+3	**−3.587 0E+3**
	Std	0.293 2E+3	0.339 6E+3	0.637 6E+3	0.283 7E+3	**0.278 8E+3**
f_8	*Best*	0	4.974 8	0	1.989 9	**0**
	Worst	6.486 2E-13	78.743 5	4.175 1E-13	16.914 3	**0**
	Mean	1.103 1E-13	25.316 9	3.180 2E-13	8.656 1	**0**

续表

Function	Result	Algorithm				
		GWO	MFO	WOA	ASO	IASO
	Std	2.116 5E-13	15.587 8	1.043 9E-13	3.223 0	**0**
	Best	0.444 1E-14	0	0.008 9E-13	0.070 0E-10	**0.888 2E-15**
	Worst	0.799 4E-14	19.950 2E-6	0.151 0E-13	0.433 8E-10	**0.888 2E-15**
f_9	*Mean*	0.680 9E-14	0.719 9E-6	0.050 3E-13	0.197 5E-10	**0.888 2E-15**
	Std	0.170 3E-14	3.644 4E-6	0.033 7E-13	0.087 5E-10	**0**
	Best	0	0.017 2	0	0	**0**
	Worst	0.094 5E-14	0.666 4	0.330 7E-14	0.027 0E-13	**0**
f_{10}	*Mean*	0.024 6E-14	0.155 5	0.048 3E-14	0.004 4E-13	**0**
	Std	0.021 4E-14	0.133 2	0.087 5E-14	0.007 8E-13	**0**
	Best	0.000 0	0.000 0	0.000 8	0.004 4	**0**
	Worst	0.201 5	0.011 0	0.211 3	0.302 1	**0.804 3E-22**
f_{11}	*Mean*	0.022 9	0.003 3	0.042 2	0.055 0	**0.211 1E-22**
	Std	0.049 9	0.005 1	0.049 5	0.061 0	**0.190 8E-22**
	Best	0.998 0	0.998 0	0.998 0	0.998 0	**0.998 0**
	Worst	12.670 5	7.874 0	10.763 2	0.001 4	**1.992 0**
f_{12}	*Mean*	3.676 0	2.249 2	3.025 8	3.968 9	**1.064 5**
	Std	3.875 0	1.918 4	3.354 9	1.398 4	**0.252 2**
	Best	0.000 3	0.000 6	0.000 3	0.716 2	**0.000 3**
	Worst	0.056 6	0.020 4	0.001 5	0.001 4	**0.000 7**
f_{13}	*Mean*	0.005 7	0.002 1	0.000 7	0.001 0	**0.000 5**
	Std	0.012 2	0.003 9	0.000 3	0.000 2	**0.000 1**
	Best	**−**10.402 7	**−**10.402 9	**−**10.401 0	**−**10.402 9	**−10.402 9**
	Worst	−5.127 3	−1.837 6	−2.765 3	**−**2.751 9	**−5.128 8**
f_{14}	*Mean*	**−**10.225 8	−6.457 5	−8.493 2	**−**6.147 9	**−10.227 1**
	Std	0.962 9	3.095 1	2.653 7	1.396 9	**0.962 9**

在f_1~f_6高维单峰测试函中，从表3可以看出，除f_5函数外，IASO在最优值、最差值、平均值还是标准差，都获得了比其他算法更好的结果，收敛精度也优于其他算法，说明了IASO算法的稳定性及其在求解高维空间问题中是有效的、可行的。在f_7~f_{11}高维多峰测试函数中，对于f_7~f_{11}函数，IASO算法优于其他算法，其中f_8，f_{10}函数，IASO算法均达到了理论最优值。表明IASO算法在解决高维多峰测试函数上性能优越，其具有很强的全局搜索能力。在3个低维高峰函数中，IASO的运行结果也优于其他算法。总体来说IASO在大部分测试函数中均展示了优越的性能。图5则为GWO，MFO，WOA，ASO，IASO这5个算法的部分收敛图。从图5可以清楚地看到，IASO在取得全局最优值的同时拥有了较快的收敛速度，可以看出IASO算法相较于其他算法有较强的竞争优势。

3.4 Wilcoxon秩和检验

为了消除实验数据的偶然性并表明实验结果的显著性，对上述仿真结果进行了显著性水平为 α=0.05 的 Wilcoxon 秩和检验[19]。Wilcoxon秩和检验用于判断验证2组数据之间有无显著的差别。

(1) H_0：对比算法求解结果无差异。

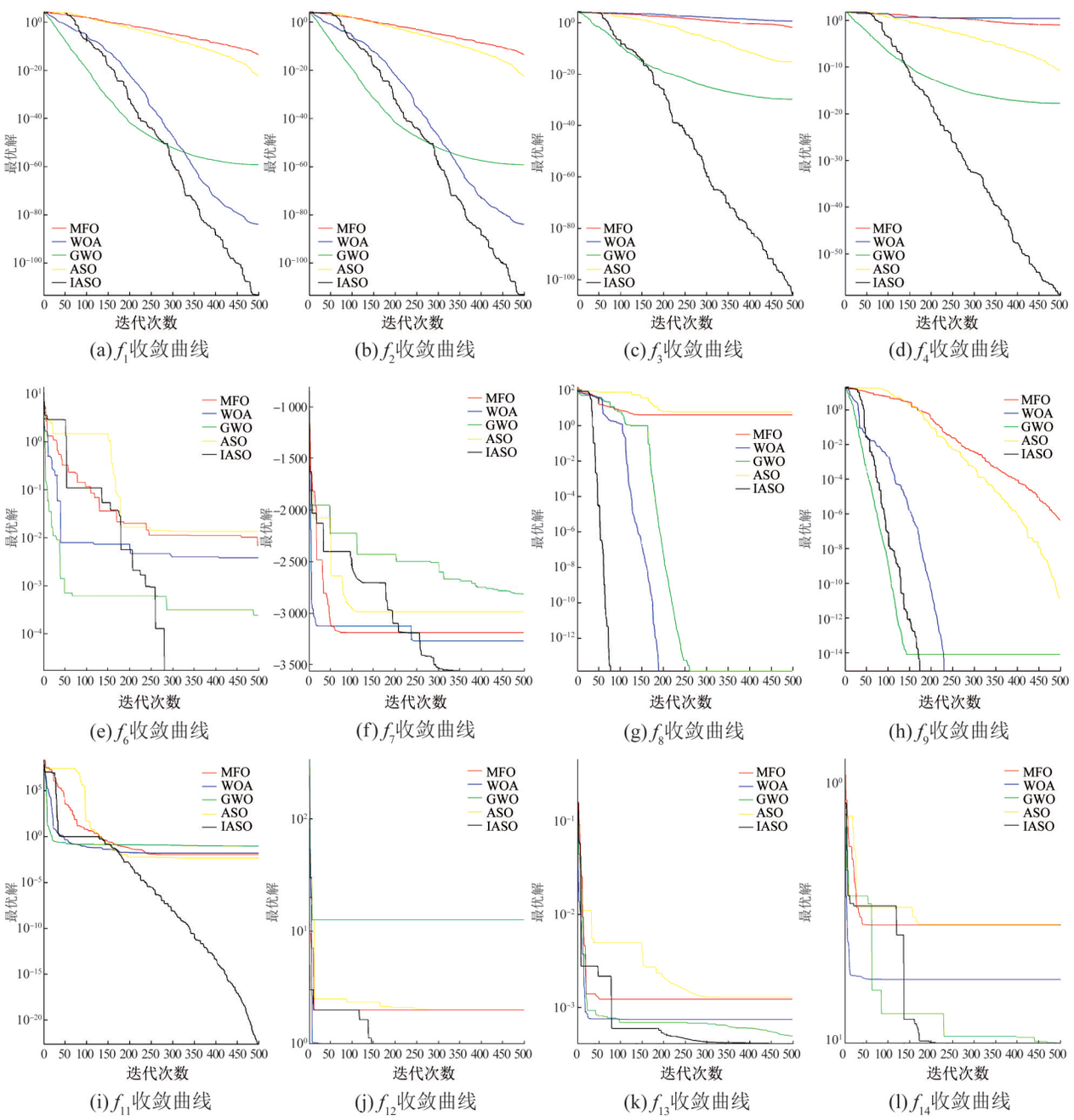
(2) H_1：对比算法求解结果有差异。

设定假设检验阈值为0.05，当显著性 P>0.05

第 34 卷第 3 期 系统仿真学报 Vol. 34 No. 3
2022 年 3 月 Journal of System Simulation Mar. 2022

时，接受零假设H_0，算法没有显著性差异；反之，当显著性P<0.05时，接受非零假设H_1，算法有显著性差异。将IASO的30次函数求解结果与GWO，MFO，GWA，ASO的30次求解结果进行秩和检验。IASO与其他4种算法相比的P值测试结果见表4所示。

由表4可知；IASO与其他4种算法相比，大部分函数的求解结果均呈现显著性差异，其中，对于函数f_5，IASO与ASO无显著性差异；对于函数f_7，IASO与WOA无显著性差异;对于函数f_{12}，IASO与MFO无显著性差异;对于函数f_{14}，IASO与MFO，WOA无显著性差异。总体来看，IASO的表现优异。



(a) f_1收敛曲线 (b) f_2收敛曲线 (c) f_3收敛曲线 (d) f_4收敛曲线

(e) f_6收敛曲线 (f) f_7收敛曲线 (g) f_8收敛曲线 (h) f_9收敛曲线

(i) f_{11}收敛曲线 (j) f_{12}收敛曲线 (k) f_{13}收敛曲线 (l) f_{14}收敛曲线

图5 5种算法在12个函数上收敛曲线对比

Fig. 5 Convergence curve comparison of 5 algorithms on 12 functions

表4 Wilcoxon秩和检验结果
Table 4 Wilcoxon rank sum test results

function	Significance level P			
	GWO	MFO	WOA	ASO
f_1	0	0	0	0
f_2	0	0	0	0
f_3	0	0	0	0
f_4	0	0	0	0
f_5	0	0	0	**0.158 0**
f_6	0	0	0	0
f_7	0	0.021 6	**0.333 7**	0
f_8	0	0	0	0
f_9	0.000 2	0.001 2	0.000 3	0.001 2
f_{10}	0	0	0.000 1	0
f_{11}	0	0	0	0
f_{12}	0	0.075	0	0.001 9
f_{13}	0.004 9	0	0.001 2	0
f_{14}	0	**0.189 2**	0	0.219 5

注：加粗的数字为显著性 P>0.05 的情况，未加粗的数字为显著性P<0.05 的情况。

4 结论

作为一种新的智能优化算法，ASO优化算法仍然有很大提高收敛精度的优化空间，本文对ASO算法进行了3个方面的改进，提出一种新的优化算法IASO。实验结果表明,对于单峰高维测试函数和高维多峰测试函数，IASO算法均表现出较快的收敛性和较好的稳定性；对于多数固定维多峰函数，IASO依然具有较好的运行结果，从而验证了本文所提算法的有效性、可行性。最后通过Wilcoxon秩和检验，也进一步说明了仿真数据的可靠性。下一阶段拟将本算法在相关应用领域中展开研究。

参考文献：

[1] Mukesh Mann, Pradeep Tomar, Om Prakash Sangwan. Bio-Inspired Metaheuristics: Evolving and Prioritizing Software Test Data[J]. Applied Intelligence March 2018 (S0924-669X), 2018, 48(3): 687-702,

[2] 林诗洁, 董晨, 陈明志, 等 新型群智能优化算法综述[J]. 计算机工程与应用, 2018, 54(12): 6-14.
Lin Shijie, Dong Chen, Cheng Minzhi, et al. Summary of New Group Intelligent Optimization Algorithms[J]. Computer Engineering and Applications, 2018, 54(12): 6-14.

[3] Holland, John H. Building Blocks, Cohort Genetic Algorithms, and Hyperplane-Defined Functions[J]. Evolutionary Computation (S1063-6560) 2000, 8(4): 373-391.

[4] Rocca Paolo, Giacomo Oliveri, Andrea Massa. Differential Evolution as Applied to Electromagnetics[J]. IEEE Antennas & Propagation Magazine (S1558-4143), 2011, 53(1): 3058-3059.

[5] Mirjalili S, Mirjalili S M, Lewis A. Grey Wolf Optimizer [J]. Advances in Engineering Software (S0965-9978), 2014, 69(3): 46-61.

[6] Mirjalili S, Lewis A. The Whale Optimization Algorithm [J]. Advances in Engineering Software (S0965-9978), 2016, 95(5): 51-67.

[7] 刘生建, 杨艳, 周永权. 一种群体智能算法——狮群算法[J]. 模式识别与人工智能, 2018, 31(5): 431-441.
Liu Shengjian, Yang Yan, Zhou Yongquan. A Swarm Intelligence Algorithm-Lion Swarm Optimization[J]. Pattern Recognition and Artificial Intelligence, 2018, 31 (5): 431-441.

[8] Rashedi E, Nezamabadi-Pour H, Saryazdi S. GSA: A Gravitational Search Algorithm[J]. Information Sciences (S0020-0255), 2009, 179(13): 2232-2248.

[9] Hatamlou, Abdolreza. Black hole: A New Heuristic Optimization Approach for Data Clustering[J]. Information ences (S0020-0255), 2013, 222: 175-184.

[10] L Deng, P Yang, W Liu. An Improved Genetic Algorithm [C]// 2019 IEEE 5th International Conference on Computer and Communications (ICCC). Chengdu, China: IEEE, 2019: 47-51,.

[11] 丁青锋, 尹晓宇. 差分进化算法综述[J]. 智能系统学报, 2017, 12(4): 431-442.
Ding Qingfeng, Yin Xiaoyu. Research Survey Of Differential Evolution Algorithms[J]. CAAI Transactions on Intelligent Systems 2017, 12(4): 431-442.

[12] Song Z, Tang C, Chen X, et al. A Self-Adaptive Mechanism Embedded Gravitational Search Algorithm[C]// 2019 12th International Symposium on Computational Intelligence and Design (ISCID). Hangzhou, China: Zhejiang University Press 2019: 108-112.

[13] X Zhao, Y Fang, Z Ma, et al. An Ameliorated Moth-Flame Optimization Algorithm[C]// 2018 37th Chinese Control Conference (CCC). Wuhan: TCCT, 2018: 2372-2377.

[14] Sun Y, Wang X, Chen Y, et al. A modified whale Optimization Algorithm For Large-Scale Global Optimization Problems[J]. Expert Systems with

第 34 卷第 3 期 系统仿真学报 Vol. 34 No. 3
2022 年 3 月 Journal of System Simulation Mar. 2022

Applications (S0957-4174), 2018, 114: 563-577.

[15] Zhao W, Wang L, Zhang Z. A Novel Atom Search Optimization for Dispersion Coefficient Estimation In Groundwater[J]. Future Generation Computer Systems (S0167-739X), 2018, 91: 601-610.

[16] Yang W, Chen L, et al. Multi/Many-Objective Particle Swarm Optimization Algorithm Based on Competition Mechanism[J]. Computational Intelligence and Neuroence (S1687-5265), 2020, 2020: 1-26.

[17] Guo X Y, Brault P. Early Stages Of Silicon Nitride Film Growth Studied By Molecular Dynamics Simulations[J]. Surface Science (S0039-6028), 2001, 488(1/2): 133-140.

[18] 高庆吉, 王瑞雪, 谈政. 基于高斯变异和自适应参考点的MOPSO优化算法[J]. 计算机应用与软件, 2019, 36 (9): 255-259
Gao Qingji, Wang Ruixue, Tan Zheng. Multi-Objective Particle Swarm Optimization Based on Gaussian Mutation and Adaptive Reference Point[J]. Computer Applications and Software, 2019, 36(9): 255-259

[19] Derrac J, García S, Molina D, et al. A Practical Tutorial On The Use of Nonparametric Statistical Tests as a Methodology for Comparing Evolutionary and Swarm Intelligence Algorithms[J]. Swarm and Evolutionary Computation (S2210-6502), 2011, 1(1): 3-18