# 透镜成像反学习策略在粒子群算法中的应用

喻 飞[1],李元香[1],魏 波[1],徐 星[2],赵志勇[1]
(1.武汉大学软件工程国家重点实验室,武汉大学计算机学院,湖北武汉 430072;2.景德镇陶瓷学院信息工程学院,江西景德镇 333403)

**摘 要:** 在 PSO 中引入反向学习策略(Opposite-Based Learning)可使粒子在搜寻过程中总能找到当前解的反向位置,增加了接近全局最优解的机会.然而,OBL 仅在演化初期作用显著,在演化后期则需通过变异等手段来提高其"开发"能力.针对该问题,基于透镜成像原理,引入缩放因子和搜索半径两个可调参数进一步平衡了算法的"探索"和"开发"能力.实验表明该策略能够提高种群多样性和收敛性能.

**关键词:** 反向学习; 粒子群算法; 透镜成像
**中图分类号:** TP18 **文献标识码:** A **文章编号:** 0372-2112 (2014)02-0230-06
**电子学报 URL:** http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2014.02.004

# The Application of a Novel OBL Based on Lens Imaging Principle in PSO

YU Fei[1], LI Yuan-xiang[1], WEI Bo[1], XU Xing[2], ZHAO Zhi-yong[1]
(1. *State key Lab of Software Engineer, College of Computer Science, Wuhan University, Wuhan, Hubei* 430072, *China*;
2. *College of Information and Engineering, Jingdezhen Ceramic Institute, Jingdezhen Jiangxi* 333403, *China*)

**Abstract:** By introducing opposition-based learning (OBL) in PSO, particles are enabled to find an opposite position that is closer to the global optimization solution. However, OBL only makes a good performance on the initial phrase of the evolution, while at later stages it needs to be combined with other techniques (e.g. Cauchy mutation) to improve its ability of "exploration". In this paper, a novel OBL based on the principle of lens imaging is proposed. It uses two parameters (i.e., zoom factor and the factor of search radius), which will achieve a better balance between PSO's "exploration" and "exploitation" abilities. The simulation shows that the novel OBL possesses better convergence rate and convergence effect.

**Key words:** opposite-based learning; particle swarm optimization; imaging lenses

## 1 引言

Kennedy[1]提出的粒子群算法(Particle Swarm Optimization,PSO)是一种典型的群体智能优化算法,已广泛应用于多个领域[2~4].PSO 受鸟群觅食行为的启发,使用了一种简单的机制进行全局寻优.然而,PSO 与大多数随机算法一样存在收敛速度慢、易陷入局部最优等缺点.目前的改进方法主要是在算法的"探索"与"开发"能力之间取得一定的平衡.

反向学习(Opposition-based Learning,OBL)策略被 Tizhoosn[5]引入群智能算法,在 DE 算法中取得了较好的效果[6].Han[7]和 Wang[8]先后把 OBL 应用用于 PSO 的改进,均取得了较好的效果.然而,OBL 在 PSO 中的应用其显著作用主要在算法执行的前期,后期的效果不明显.因为在演化后期,大部分粒子集中到小范围区域,反向点所在区域也集中于此,即反向点的适应值并不比原值好多少,所以在演化后期试图通过反向的方式很难跳出当前的极值区域.Wang[8]引入了柯西变异策略对当前解进行小范围的扰动,增强算法逃离局部最优解的能力,并对其进行了一般化[9].虽然 OBL 在 PSO 中的应用有效提高了算法的"探索"能力,但仍需要借助柯西变异[8]、精英选择[10]等方法提高其"开采"能力.为了在解决高维问题时避免 PSO 算法陷入局部最优解,减少对其它方法的依赖,Omran[11]做了一定的尝试,仅使用 OBL 来提高算法的"开发"能力.

本文基于透镜成像原理设计了一种适用于 PSO 算法的透镜反向(lensOBL)规则.该规则在演化后期对反向位置进一步扩展,增强粒子逃逸极值区域的能力,提高反向策略的效率.lensOBL 引入的"缩放因子"和"搜索半径"两个参数保证了反向学习策略在整个搜索期间都能进行"宏观"和"微观"两方面的调节.该方法不需要引入其他变异方法,所需参数较少.实验表明,该方法进一

收稿日期:2013-03-04;修回日期:2013-06-21;责任编辑:蓝红杰
基金项目:国家自然科学基金(No.61070009);国家科技支撑计划(No.2012BAH25F02 );江西省自然(青年)科学基金(No.2012BAB211036)

步实现了“探索”与“开发”能力的平衡.

## 2　标准粒子群算法

标准 PSO 算法将优化问题的求解过程看成是在 $D$ 维搜索空间中进行搜寻.问题的解是搜索空间中没有质量没有体积的粒子,粒子在搜索空间中以一定速度飞行.$N$ 个粒子中第 $i$ 个粒子的位置表示为 $\boldsymbol{X}_i=(x_{i1},x_{i2},\cdots,x_{iD})$,其飞行速度用 $\boldsymbol{V}_i=(v_{i1},v_{i2},\cdots,v_{iD})$表示.粒子根据自身的经验(个体历史最好值)和其它粒子的经验(全局历史最好值)来调整自己的飞行.个体历史最好位置(即有最好的适应值)表示为 $\boldsymbol{p}_i=(p_{i1},p_{i2},\cdots,p_{iD})$,记为 ***pbest***.全局历史最好值用 $\boldsymbol{p}_g$ 表示($g$ 为全局最好位置粒子的索引号),记为 ***gbest***.在寻优迭代过程中,第 $i$ 个粒子的 $d$ 维($1\leqslant d\leqslant D$)速度及位置按以下公式进行更新:

$$v_{id}^{k+1}=\omega v_{id}^{k}+c_1r_1(pbest_{id}^{k}-x_{id}^{k})+c_2r_2(gbest_{d}^{k}-x_{id}^{k})$$
$$x_{id}^{k+1}=x_{id}^{k}+v_{id}^{k+1}\qquad(1)$$

其中,$v_{id}^{k}$和 $x_{id}^{k}$表示 $k$ 次迭代时粒子 $i$ 速度和位置向量的 $d$ 维分量;$pbest_{id}^{k}$和 $gbest_{id}^{k}$分别表示粒子 $i$ 历史最优位置和种群当前最优位置向量的 $d$ 维分量,$c_1$ 和 $c_2$ 为学习因子,通常取 2,$r_1$ 和 $r_2$ 是介于(0,1)的随机数.$\omega$ 为惯性权重,通常有 $\omega\in[0,1]$,$\omega$ 用于维护全局和局部搜索能力的平衡,提高算法收敛性能.

## 3　基于椭圆透镜成像原理的粒子群算法

### 3.1　反向学习

在 PSO 中应用 OBL 是基于这样的假设:对于解空间中的解 $X$,总能在当前空间中找到其对应的反向解 $X^*$,如果反向解优于 $X$,则用反向解代替原来的解作为当前最优解.显然,反向解提供了一次接近全局最优解的机会.

**定义 1**　反向数:实数 $x\in[a,b]$,其反向数 $x^*$ 定义为 $x^*=a+b-x$.

**定义 2**　反向点:假设 $\boldsymbol{X}(x_1,x_2,\cdots,x_D)$为 $D$ 维空间中的一点,且 $x_j\in[a_j,b_j]$,$j\in1,2,\cdots,n$,则 $\boldsymbol{X}$ 的反向点表示为$\boldsymbol{X}^*(x_1^*,x_2^*,\cdots,x_D^*)$,其中 $x_j^*=a_j+b_j-x_j$.

以一维空间为例,即存在这样的函数 $g(x)$,使得每一个 $x\in[a,b]$,都有 $x^*=g(x)$.转换空间的中心位置在$[-\frac{a+b}{2},\frac{a+b}{2}]$之间随机分布[9].在随机算法中使用 OBL 增加了找到解的机会,可使算法快速收敛[12].如图 1 所示,粒子的寻优边界随着算法的执行进行不断变化,最终收敛至解的位置(实心点为解的位置,空心点为当前解 $x$ 及其反向解 $x^*$的位置).

在 PSO 中,OBL 在种群初始化中发挥了较大的作用.然而在演化后期,由于粒子大量聚集在局部最优值附近,OBL 作用不明显.另外,Wang 等人[6,7,9]在 PSO、DE 等算法中以一定概率使用 OBL,进一步增大了随机性,算法收敛速度放慢.

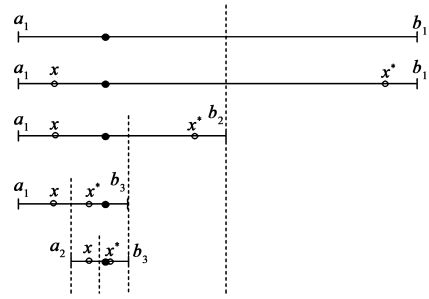



图1　一维空间反向寻解过程

### 3.2　基于透镜成像原理的反向学习

基于透镜的反向学习规则对 OBL 策略进行扩展,可以解决上述问题.

**定义 3**　基点:若 $D$ 维空间中存在若干点 $\boldsymbol{o}_1,\boldsymbol{o}_2,\cdots\boldsymbol{o}_m$,对于任意一点 $\boldsymbol{X}(x_1,x_2,\cdots,x_D)$与其反向点 $X^*(x_1^*,x_2^*,\cdots,x_D^*)$到 $\boldsymbol{o}_i$ 的欧氏距离分别为 $l_i$ 和 $l'_i$,令 $k=l_i/l'_i$,且 $k=1,2,\cdots,n$,则 $\boldsymbol{o}_i$ 被称为 $\boldsymbol{X}$ 与 $\boldsymbol{X}^*$ 在 $k=i$ 时的基点,$k$ 称为缩放因子.

粒子在空间中寻找反向点的过程可以看成透镜成像的过程.以一维空间为例,假设坐标轴上区间 $[a,b]$ 上 $x$ 处有高度为 $h$ 的物体,基点位置 $o$(本文取基点位置为$\frac{a+b}{2}$)上放置焦距为 $r$ 的透镜,其像的高度为 $h'$,成像原理如图 2 所示.

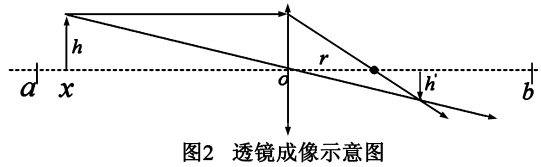



图2　透镜成像示意图

图 2 中,$x$ 以 $o$ 为基点找到了对应的反向点 $x^*$,由透镜成像原理我们不难得出以下公式:

$$\frac{(a+b)/2-x}{x^*-(a+b)/2}=\frac{h}{h'}\qquad(2)$$

$$\frac{r}{x^*-(a+b)/2-r}=\frac{h}{h'}\qquad(3)$$

显然 $h/h'=k$,即缩放因子 $k$ 与物和像之间的比例关系对应.对于式(2)进行变换即可得到反向点 $x^*$ 的计算公式如下:

$$x^*=(a+b)/2+(a+b)/2k-x/k\qquad(4)$$

当 $k=1$ 时,上式即为中心位置在$[-\frac{a+b}{2},\frac{a+b}{2}]$的 OBL.将上式推广至 $D$ 维空间,粒子在搜索空间中的位置为 $\boldsymbol{X}(x_1,x_2,\cdots,x_D)$,根据反向点的定义,$\boldsymbol{X}^*(x_1^*,x_2^*,\cdots,x_D^*)$为解空间中与之对应的反向点.如果适应值 $f(\boldsymbol{X}^*)$比 $f(\boldsymbol{X}^*)$更好些,则用反向点 $\boldsymbol{X}^*$替代 $\boldsymbol{X}$,从而使

粒子在更好的位置上获得信息以调整自身的飞行.

然而,粒子的反向点可能一起落入极值点区域,如图 3 所示:

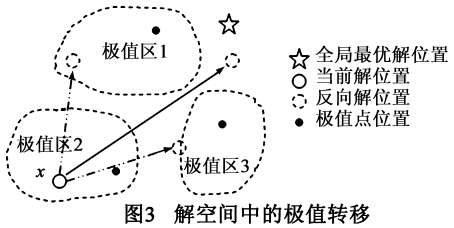



**图3 解空间中的极值转移**

显然,一旦反向点也落入极值区域,所有的粒子会迅速向该极值点靠近.因此,需要设法让粒子的反向位置落在更广的范围内,使粒子逃逸极值区域[13].

对式(3)进行变换可得到如下公式:

$$x^* = (1 + 1/k)r + \frac{a+b}{2} \tag{5}$$

即对于固定的缩放因子,若搜索半径已知则可用式(5)表示反向点.

联立式(4)和式(5)可得:

$$r = \frac{a + b - 2x}{2(k+1)} \tag{6}$$

在 PSO 中,初始种群粒子的反向搜索半径 $r$ 可以通过计算所有粒子对应的反向搜索半径获得.当缩放因子固定时,可以通过调节搜寻半径,改变搜寻的范围,在一定程度上使反向点逃逸极值点.

**定理 1** 基点固定,反向搜索半径 $r$ 固定时,缩放因子 $k$ 越大,反向点离基点越近,反之则越远.

**定理 2** 基点固定,缩放因子 $k$ 固定时,反向搜索半径 $r$ 越大,反向点离基点越远,反之则越近.

定理 1 和定理 2 的证明从式(5)不难得出.

缩放因子 $k$ 和反向搜索半径 $r$ 作为两个可调参数,在 PSO 搜索解的过程中充分发挥反向学习策略的作用.搜索半径 $r$ 是宏观调控因子,其较小的变化会引起搜索范围的大范围变化,便于提高算法的"探索"能力.而缩放因子 $k$ 是微观调控因子,主要是为了改善算法的"开发"能力.

本文采用线性减少缩放因子,其公式表示如下:

$$k_t = k_{\max} - \frac{k_{\max} - k_{\min}}{t_{\max}} t \tag{7}$$

其中 $k_{\max}$和 $k_{\min}$分别表示缩放因子最大值和最小值,$t_{\max}$为最大迭代次数,$t$ 为当前迭代次数.

### 3.3 透镜成像反向规则

**定义 4** 坡度:设 $\boldsymbol{P}_t = (\boldsymbol{X}_1, \boldsymbol{X}_2, \cdots, \boldsymbol{X}_N) \in R^N$ 为第 $t$ 代种群,粒子 $X_i \in \boldsymbol{P}_t$,用 $f_{avg}(t)$表示第 $t$ 代粒子的平均适应值,则第 $t$ 代种群的坡度 $m(t)$表示成以下公式:

$$m(t) = f_{avg}(t-1) - f_{avg}(t) \tag{8}$$

其中,$f_{avg}(t) = \sum_{i=1}^{i=N} f(\boldsymbol{X}_i)/N$.

当 $m(t) < 0$ 时,此时搜索到极值,为避免陷入局部最优,应给粒子逃逸此极值的力量.由 3.2 节可知缩放因子是线性递减的,因此搜索半径 $r_t$ 大于当前实际搜索半径,即通过公式(5)计算的反向点会落在当前搜索区域$[a, b]$之外的区域,在包含当前极值区域更广的搜索空间中进行搜索,增加粒子逃逸极值的几率.算法 1 给出了透镜反向规则 lensOBL 的描述.

**算法 1** 透镜反向规则 lensOBL

输入:第 $t$ 代种群 $P_t$,缩放因子 $k_t$ 和搜索半径 $r_t$,坡度 $m(t)$.

输出:用新种群替换第 $t$ 代种群.

```
for (i = 1 to N) do
    根据式(4)得粒子 X_i 的反向粒子 X_i^*;
    将反向粒子加入反向种群 OP_t;
end
if (m(n) < 0) && (k_t! = 0) && (r_t! = 0) do
    r = r_t * rand(0,1);
    for (i = 1 to N) do
        根据式(5)计算 X_i 的反向粒子 X_i^*;
        将反向粒子加入反向子种群 OPS_t;
    end
end
从集合{P_t, OP_t, OPS_t}中选择 N 个适应值最好的粒子替换种群 P_t
```

## 4 基于 lensOBL 的粒子群算法

lensOBL 反向规则有效使用的一个前提条件就是搜索边界的不断变化,基于 lensOBL 的粒子群算法(lensPSO)动态更新当前搜索空间边界,使用与文献[9]相同的更新策略,用公式表示如下:

$$a_j(t) = \min(X_{i,j}(t)), b_j(t) = \max(X_{i,j}(t)) \tag{9}$$

即 $a_j(t)$和 $b_j(t)$分别表示第 $t$ 代 $j$ 维的上下界.

基于透镜成像反学习的 PSO 算法描述如下:

**算法 2** 使用 lensOBL 反向规则的粒子群算法 lensPSO

输入:搜索空间 $S$ 和目标函数 $f$.

输出:最优解 $\zeta$

```
设置缩放因子 k_0;
随机生成 N 个粒子作为初始种群 P_0;
P_1 = lensOBL(P_0, k_0, 0, 0);
根据式(6)计算初始搜索半径 r_1;
t = 1;
while (Termination_test(P_t) = = False)
    按公式(7)更新缩放因子;
    按公式(8)计算种群坡度 m(t);
    按公式(9)更新当前位置边界 a 和 b;
    P_t = lensOBL(P_t, k_t, r_t, m(t));
    for (i = 1 to N) do
        按公式(1)计算粒子 X_i 的速度;
```

13)　　按公式(2)更新粒子 $\boldsymbol{X}_i$ 的位置;
14)　end
15)　更新 *pbest* 和 *gbest*;
16)　$t = t + 1$;
17)　end
18)　将第 $t$ 代的粒子最好位置作为解 $\zeta$ 输出;

## 5　实验及结果分析

为了验证 lensPSO 算法的优化性能,选择常用的 8 个 Benchmark 函数[4,14]:Sphere、Quadric、Rosenbrock、Rastrigin 、Griewank、Ackley、Schaffer、Schwelfel($f_1 - f_8$)进行数值试验,函数自变量范围按文献[4]取得,全局最优值均为 0.其中,$f_1 - f_3$ 为单峰函数,$f_4 - f_8$ 为多峰函数.

选择标准 PSO 算法和 GOPSO 算法[9]与本文提出的 lensPSO 算法进行对比实验.文献[11]中提出的 iPSO 是一种仅使用反向学习策略的改进算法,也将其与本算法对比.算法参数设置如下:惯性权重 $w = 0.7298$, $c_1 = c_2 = 1.4$,lensPSO 中的缩放因子初始值设置为 $k = 0.75$.种群规模均为 40,函数维数为 30,每次运行 3000 代,每个函数独立运行 30 次.

### 5.1　lensPSO 在 Benchmark 函数上的表现

从图 5($a$)($b$)($c$)中可以看出,在处理单峰函数上,lensPSO 具有明显的优势.对于简单的 Sphere 函数和 Quadric 函数,lensPSO 算法表现出快速收敛能力.对于复杂的 Rosenbrock 函数,虽然函数山谷提供的信息少,但是 lensPSO 算法仍然能避免陷入局部最优值,算法能迅速找到搜索方向,快速收敛.这是因为当粒子集中在极值附近时,搜索半径 $r$ 的变动,使算法能在较大范围内寻找粒子的反向点,搜索到新的极值,并替换当前找到的最好值,从而使粒子向新的极值点靠近,挣脱了极值点的束缚,在粒子向新的极值点靠近时再次随机改变搜索半径 $r$,增加粒子靠近全局最优点的机会.

统计算法在 Benchmark 函数上独立运行 30 次所得到的最好值、最差值、平均值和标准差的结果见表 1.可

**表 1　函数测试数据**

| 函数 | 算法 | 最小值 | 中间值 | 平均值 | 方差 | 最大值 |
|---|---|---|---|---|---|---|
| $f_1$ | PSO | 4.11E + 00 | 1.29E + 01 | 1.26E + 01 | 5.52E + 00 | 2.12E + 01 |
| | GOPSO | 6.72E - 01 | 3.91E + 00 | 3.64E + 00 | 2.07E + 00 | 7.26E + 00 |
| | iPSO | 5.91E - 29 | 3.98E - 15 | 1.01E - 03 | 3.02E - 03 | 9.05E - 03 |
| | lensPSO | 2.74E - 152 | 7.20E - 61 | 7.22E - 31 | 2.16E - 30 | 6.49E - 30 |
| $f_2$ | PSO | 1.361 | 92.42 | 236.7 | 20.93 | 396.25 |
| | GOPSO | 1.54E + 00 | 4.80E + 00 | 1.05E + 03 | 2.56E + 03 | 3.49E + 00 |
| | iPSO | 2.17E - 38 | 3.78E - 21 | 9.49E - 11 | 2.12E - 10 | 7.91E - 11 |
| | lensPSO | 1.40E - 106 | 2.90E - 26 | 1.95E - 18 | 4.35E - 18 | 1.62E - 18 |
| $f_3$ | PSO | 1.74E + 01 | 3.08E + 01 | 2.90E + 01 | 8.56E + 00 | 3.70E + 01 |
| | GOPSO | 1.45E + 01 | 2.54E + 01 | 2.46E + 01 | 7.73E + 00 | 3.33E + 01 |
| | iPSO | 2.89E + 01 | 2.90E + 01 | 2.90E + 01 | 1.41E - 02 | 2.90E + 01 |
| | lensPSO | 2.31E - 02 | 5.20E - 02 | 5.65E - 02 | 3.54E - 02 | 9.89E - 02 |
| $f_4$ | PSO | 1.17E + 0217 | 2.07E + 02 | 1.90E + 02 | 5.05E + 01 | 2.31E + 02 |
| | GOPSO | 6.14E + 01 | 9.97E + 01 | 9.37E + 01 | 2.29E + 01 | 1.14E + 02 |
| | iPSO | 2.67E - 07 | 8.78E - 01 | 8.32E - 01 | 6.54E - 01 | 1.57E + 00 |
| | lensPSO | 0 | 0 | 8.25E - 10 | 1.65E - 09 | 3.30E - 09 |
| $f_5$ | PSO | 2.13E + 01 | 1.24E + 02 | 1.11E + 02 | 7.82E + 01 | 2.31E + 02 |
| | GOPSO | 1.43E + 01 | 3.04E + 01 | 3.94E + 01 | 2.95E + 01 | 9.50E + 01 |
| | iPSO | 1.29E + 00 | 6.30E + 00 | 3.36E + 01 | 6.87E + 01 | 1.74E + 02 |
| | lensPSO | 0 | 2.20E - 10 | 1.01E - 09 | 1.44E - 09 | 3.22E - 09 |
| $f_6$ | PSO | 9.81E + 00 | 1.49E + 01 | 1.48E + 01 | 3.29E + 00 | 1.81E + 01 |
| | GOPSO | 5.15E + 00 | 1.26E + 01 | 1.16E + 01 | 4.07E + 00 | 1.60E + 01 |
| | iPSO | 6.24E + 00 | 1.64E + 01 | 1.44E + 01 | 6.93E + 00 | 2.09E + 01 |
| | lensPSO | 8.88E - 16 | 8.88E - 16 | 7.76E - 10 | 1.73E - 09 | 3.88E - 09 |
| $f_7$ | PSO | 6.91E + 00 | 9.37E + 00 | 9.75E + 00 | 2.22E + 00 | 1.31E + 01 |
| | GOPSO | 2.11E + 00 | 4.25E + 00 | 4.15E + 00 | 1.71E + 00 | 6.50E + 00 |
| | iPSO | 3.07E - 01 | 1.61E + 00 | 3.02E + 00 | 2.79E + 00 | 7.09E + 00 |
| | lensPSO | 4.60E - 02 | 7.30E - 02 | 9.82E - 02 | 7.30E - 02 | 2.27E - 01 |
| $f_8$ | PSO | 7.69E + 02 | 9.11E + 02 | 8.66E + 02 | 7.13E + 01 | 9.25E + 02 |
| | GOPSO | 2.12E + 02 | 2.54E + 02 | 2.70E + 02 | 4.75E + 01 | 3.21E + 02 |
| | iPSO | 7.16E + 02 | 7.30E + 02 | 7.64E + 02 | 8.71E + 01 | 9.19E + 02 |
| | lensPSO | 1.61E + 02 | 2.03E + 02 | 2.10E + 02 | 4.02E + 01 | 2.71E + 02 |

以看出,lensPSO 算法在各项数值上均优于其它算法.而对于 Sphere 函数和 Quadric 函数,lensPSO 算法精度远高于其它算法.对于 Rosenbrock 函数,由于函数特性,各算法均未找到最优值,但是 lensPSO 算法总能收敛于最优值附近,稳定性与 iPSO 相当,但其结果要好于其它算法.

对于多峰函数,从图 5($d$)($e$)($f$)($g$)中可以看出,对函数 $f_4$,仅使用反向策略的 GOPSO 与标准 PSO 性能相差不太大,iPSO 能够收敛至全局最优解,从表 1 中可以看出其收敛精度不及 lensPSO 算法.对于函数 $f_5$、$f_6$,虽然 GOPSO 和 iPSO 算法都能找到全局最优解所处的极值区域,但精度仍然不及 lensPSO 算法.对于函数 $f_7$,lensPSO 算法能搜索至全局最优解附近极值区域,其它三种算法则早熟.对于函数 $f_8$,这是一个典型的欺骗函数,其全局极值点周围的所有局部极值点远离它,iPSO 算法在搜索过程中陷入错误的收敛方向,而 lensPSO 算法性能则略好于 PSO 和 GOPSO.

## 5.2 缩放因子 $k$ 对算法效率的影响

缩放因子 $k$ 和搜索半径 $r$ 作为"宏观"和"微观"调控的两个参数,其取值对算法的效率有一定的影响.搜索半径 $r$ 作为微调的参数,其改变可看成是在当前极值附近的"扰动",具有随机性.为了分析缩放因子 $k$ 对算法的影响,实验中搜索半径 $r$ 按算法 2 中的方式取得.为验证参数 $k$ 对算法的影响,确定其合适的取值范围,采用不同的取值在测试函数上进行了试验,结果如图 4 所示.从图中可以看出,阈值取[0.6,1]时,算法有较好的表现.缩放因子 $k$ 和搜索半径 $r$ 的选择可以协调进行选择.实验中采用固定的缩放因子,取 $k=0.75$.缩放因子确定后,算法能快速收敛,但是也易陷入局部最

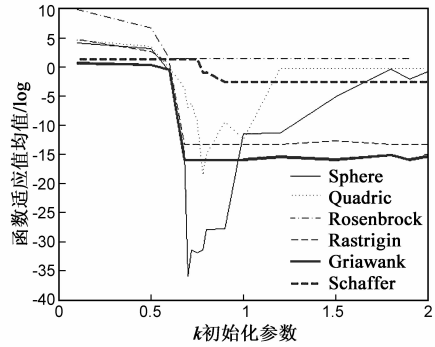



图4 不同 参数对算法性能的影响

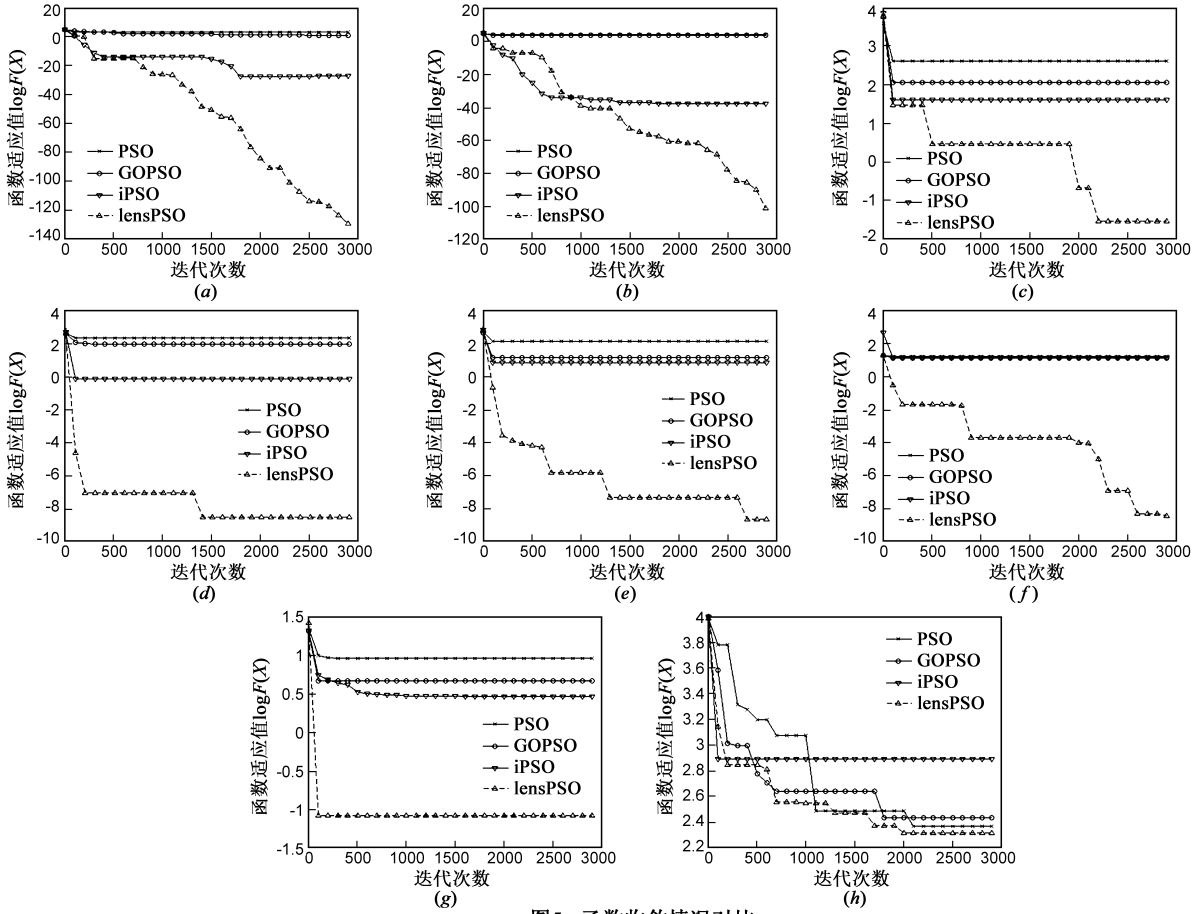



图5 函数收敛情况对比

优值.因此,在演化后期,坡度 $m(t)<0$ 时,"宏观"调控条件满足,则算法主要靠搜索半径 $r$ 进行调节.此时,粒子主要集中在极值点附近,搜索半径 $r$ 的调控增大了粒子反向点落在极值区域外的几率,使得搜索范围扩大,从而增加了粒子逃逸局部极值的机会,避免陷入极值区域.因此,缩放因子 $k$ 选择适当的值后,需要靠搜索半径 $r$ 来配合使用,从而达到较好的平衡状态.

## 6　结束语

本文分析了反向学习策略在 PSO 算法应用中存在的问题,从透镜成像原理中得到启发,引入了缩放因子和搜索半径两个参数对 PSO 算法进行"微观"和"宏观"上的调控,进一步改善其"开发"与"探索"能力,实验表明该方法不需要借助于其它策略就能取得较好的结果,对 OBL 的应用有一定指导意义.

### 参考文献


[1] Kennedy J, Eberhart R C. Particle swarm optimization[A]. Proceedings of IEEE International Conference on Neural Networks [C]. Perth, Australia, 1995. 1942 – 1948.

[2] 田野,刘大有.求解流水车间调度问题的混合粒子群算法[J].电子学报,2011,39(5):087 – 1093.
Tian Ye, Liu Da-you. A hybrid particle swarm optimization method for flow shop scheduling problem[J]. Acta Electronica Sinica, 2011, 39(5): 1087 – 1093. (in Chinese)

[3] Z Zheng, Hock Soon S, Jixiang S. A hybrid particle swarm optimization with cooperative method for multi-object tracking [A]. Congress on Evolutionary Computation (CEC′2012)[C]. Washington, D C, USA: IEEE Press, 2012. 1 – 6.

[4] G Yue-Jiao, Z Jun, H S Chung. An efficient resource allocation scheme using particle swarm optimization[J]. IEEE Transactions on Evolutionary Computation, 2012, 16(6): 801 – 816.

[5] Tizhoosh H R. Opposition-based learning: A new scheme for machine intelligence[A]. Proceedings of International Conference on the Computational Intelligence for Modelling, Control and Automation, 2005 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce[C]. Washington, D C, USA: IEEE Press, 2005. 695 – 701.

[6] Rahnamayan S, Tizhoosh H R, M M A Salama. Opposition-based differential evolution[J]. IEEE Transactions on Evolutionary Computation, 2008, 12(1): 64 – 79.

[7] H Lin, H Xing-shi. A Novel opposition-based particle swarm optimization for noisy problems[A]. Proceedings of International Conference on the Natural Computation[C]. Piscataway: IEEE Press, 2007. 624 – 629.

[8] W Hui, L H hui, L Yong. Opposition-based particle swarm algorithm with cauchy mutation[A]. Congress on Evolutionary Computation (CEC′2007)[C]. Washington, D C, USA: IEEE Press, 2007. 4750 – 4756.

[9] Wang H, Wu Z, Rahnamayan S. Enhancing particle swarm optimization using generalized oppositionbased learning[J]. Information Sciences, 2011, 181(20): 4699 – 4714.

[10] Wang H, Wu Z, Rahnamayan S. Diversity Analysis of Opposition-Based Differential Evolution—an Experimental Study [M]. Advances in Computation and Intelligence, Springer Berlin Heidelberg, 2010. 95 – 102.

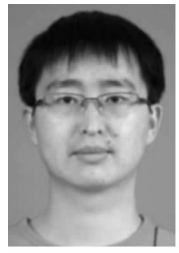
[11] Omran M G H, AL-Sharhan S. Using oppositionbased learning to improve the performance of particle swarm optimization [A]. Proceedings of the IEEE Swarm Intelligence Symposium (SIS′2008)[C]. Washington, D C, USA: IEEE Press, 2008. 1 – 6.

[12] Rahnamayan S, Tizhoosh H R, Salama M M A. Opposition versus randomness in soft computing techniques[J]. Applied Soft Computing, 2008, 8(2): 906 – 918.

[13] 陶新民,刘福荣,等.一种多尺度协同变异的粒子群优化算法[J].软件学报,2012,23(7):1805 – 1815.
Tao X M, Liu F R, et al. Multi-scale cooperrative mutation particle swarm optimization algorithm[J]. Journal of Software, 2012, 23(7): 1805 – 1815(in Chinese)

[14] He S, Wu Q H, Saunders J R. Group search optimizer: an optimization algorithm inspired by animal searching behavior [J]. IEEE Transactions on Evolutionary Computation, 2009, 13(5): 973 – 990.


### 作者简介

**喻　飞**　男,1981 年出生于湖北钟祥,武汉大学计算机学院博士研究生.研究方向为智能计算、自然计算.
E-mail: yufei@whu.edu.cn

**李元香**　男,1962 年出生于湖北监利,武汉大学计算机学院软件工程国家重点实验室教授,博士生导师,从事并行计算与演化计算的理论与应用研究.
E-mail: yxli@whu.edu.cn