

doi:10.11835/j.issn.1000-582X.2017.05.012

# 改进的粒子群算法及在数值函数优化中应用

李建平, 宫耀华, 卢爱平, 李盼池

(东北石油大学 计算机与信息技术学院, 黑龙江 大庆 163318)

**摘要:**为提高粒子群算法的优化能力,提出了一种改进的粒子群优化算法。在该算法中,采用 Beta 分布初始化种群,采用逆不完全伽马函数更新惯性权重,在速度更新式中,引入了基于差分进化的新算子,对于粒子的越界处理,采用了基于边界对称映射的新方法。以 50 个不同类型的数值函数作为优化实例,基于威尔柯克斯符号秩检验的测试结果表明,该算法明显优于普通粒子群优化算法、差分进化算法、人工蜂群优化算法和量子行为粒子群算法。

**关键词:**粒子群优化;Beta 分布函数;逆不完全伽马函数;数值优化;算法设计

中图分类号:TP183

文献标志码:A

文章编号:1000-582X(2017)05-095-09

## Application of improved particle swarm optimization to numerical function optimization

LI Jianping, GONG Yaohua, LU Aiping, LI Panchi

(School of Computer and Information Technology, Northeast Petroleum University,  
Daqing 163318, Heilongjiang, P.R.China)

**Abstract:** To enhance the optimization ability of the particle swarm optimization (PSO), an improved PSO algorithm was proposed in this paper. In the proposed approach, the Beta distribution function is used to initialize population, and the inverse incomplete gamma function is used to update the inertia weight. For adjustment of velocity, a new operator based on differential evolution is introduced. For cross-border processing of particles, a new method based on boundary symmetry mapping is designed. With taking 50 different types of benchmark functions as optimization examples, the experimental results based on the Wilcoxon-Signed rank test show that the proposed algorithm is obviously superior to the common PSO, differential evolution, artificial bee colony algorithm and quantum-behaved particle swarm optimization algorithm.

**Keywords:** particle swarm optimization; Beta distribution function; inverse incomplete gamma function; numerical optimization; algorithm design

粒子群优化算法(particle swarm optimization, PSO)是 1995 年由 Eberhart 和 Kennedy 提出一种模拟鸟群捕食行为的群智能优化算法<sup>[1-2]</sup>。PSO 具有操作简单,可调参数少的优点,目前已广泛应用于函数优化、神经网络训练、模糊系统控制以及其他相关的工程应用领域<sup>[3-4]</sup>。关于 PSO 性能的改进,目前主要有以下几种策略:一是基于算法参数的选择<sup>[5-7]</sup>;二是基于粒子位置及速度的更新规则<sup>[8-9]</sup>;三是与其它算法的融合,如

收稿日期:2016-10-21

基金项目:中国石油科技创新基金资助项目(2016D-5007-0302)。

Supported by PetroChina Innovation Foundation (2016D-5007-0302).

作者简介:李建平(1976-),男,副教授,博士,主要从事群智能优化算法方向研究,(E-mail)leejp@126.com。

混沌 PSO<sup>[10]</sup>、模糊 PSO<sup>[11]</sup>、多子群 PSO<sup>[12]</sup>、模拟退火 PSO<sup>[13]</sup>等。四是赋予粒子新特性,例如文献[14]提出一种基于非均匀变异和多阶段扰动的粒子群优化算法;文献[15]在改进的算法中建立了一种新的粒子群模型,在该模型中的粒子具有视野受阻且认知自身目标的特性;文献[16]提出一种基于粒子分类和按维动态改变权重的改进粒子群算法。以上这些改进中,比较典型的是对于惯性因子的改进。文献[17]详细考察了惯性因子的各种取值对算法性能的影响,文献[18]研究了动态惯性因子(随迭代步数线性或非线性下降)对性能的影响,这些改进使 PSO 性能均有不同程度的提高。

为了进一步提高 PSO 的优化能力,研究从粒子群初始化方法、惯性因子的动态更新策略、粒子的速率更新式、位置的越界处理 4 个方面,提出相应的改进措施,以期提高传统 PSO 的优化性能。采用 50 个不同类型和维度的标准测试函数作为优化对象,并通过与其他同类方法对比,结果验证了提出方法的有效性。

## 1 基本粒子群算法

设在  $n$  维空间中的  $N$  个粒子组成一个种群。其中,第  $i$  个粒子位置  $X_i$ 、速度  $V_i$ 、自身搜索到的最优位置  $P_i^l$ 、整个种群搜索到的最优位置  $P_g$  分别记为: $X_i=(x_{i1}, \dots, x_{in})$ ;  $V_i=(v_{i1}, \dots, v_{in})$ ;  $P_i^l=(p_{i1}, \dots, p_{in})$ ;  $P_g=(p_{g1}, \dots, p_{gn})$ 。将  $X_i$  代入目标函数可计算其目标值。种群的初始化方法为  $X_{ij}=X_{\min,j}+(X_{\max,j}-X_{\min,j})\times \text{rand}$ ,  $\text{rand}$  为在区间  $(0,1)$  均匀分布的随机数,粒子的更新策略为

$$V_i(t+1)=\omega V_i(t)+c_1 r_1 (P_i^l - X_i(t))+c_2 r_2 (P_g - X_i(t)), \quad (1)$$

$$X_i(t+1)=X_i(t)+V_i(t+1), \quad (2)$$

其中  $i=1,2,\dots,N$ ;  $\omega$  为惯性因子;  $c_1$  为自身因子;  $c_2$  为全局因子;  $r_1, r_2$  是  $(0,1)$  之间随机数。对种群中每个粒子应用式(1)、(2)循环迭代,可使整个种群逐步逼近全局最优解。为便于叙述,将式(1)重写为如下形式

$$V_i(t+1)=\omega V_i(t)+[\Phi](P_i - X_i(t)), \quad (3)$$

其中

$$P_i = \text{diag}\left(\frac{c_1 r_1^1}{c_1 r_1^1 + c_2 r_1^2}, \dots, \frac{c_1 r_n^1}{c_1 r_n^1 + c_2 r_n^2}\right) P_i^l + \text{diag}\left(\frac{c_2 r_1^2}{c_1 r_1^1 + c_2 r_1^2}, \dots, \frac{c_2 r_n^2}{c_1 r_n^1 + c_2 r_n^2}\right) P_g, \quad (4)$$

$$[\Phi] = \text{diag}(c_1 r_1^1 + c_2 r_1^2, \dots, c_1 r_n^1 + c_2 r_n^2), \quad (5)$$

为使 PSO 收敛,所有粒子应逼近式(4)定义的  $P_i$ 。

## 2 改进的粒子群算法

笔者提出的改进粒子群优化(improved pso, IPSO),其改进措施主要针对如下 4 个方面:粒子群初始化、惯性因子、速率更新、位置越界。

### 2.1 粒子群初始化

优化过程,实质上是在解空间搜索最优解的过程,群智能优化的优势在于,群体成员相互协作共同向着最优解逼近,即“围攻”最优解。因此,种群初始位置的设置就在一定程度上影响着搜索的进程,传统的按均匀分布随机生成初始位置的方法,简单易行,但不利于形成对最优解的“包围”。为使初始位置有效包围最优解,提出基于 Beta 分布的种群初始化方法。

Beta 分布函数的定义为

$$\beta(x) = \frac{x^{a-1} (1-x)^{b-1}}{B(a,b)}, 0 < x < 1, \quad (6)$$

其中分母是  $\beta$  函数,定义如下

$$B(a,b) = \int_0^1 t^{a-1} (1-t)^{b-1} dt, \quad (7)$$

其中  $a=b<1$ 。

在这种情况下,密度函数的形状是一个对称的 U 型。此时有这样的结果,候选解最有可能位于搜索空间的边界附近,因此全局最优解被较好的“包围”在初始粒子种群内。大量实验表明,当  $a=b=0.8$  时,初始化效果较好,此时该分布在区间  $(0,1)$  的概率密度函数图像如图 1 所示。

令粒子个数为  $N$ ,优化空间为  $D$  维,初始化后的候选解为  $X_i$ 。根据  $\beta$  概率分布对  $X_i$  的每一维按下式赋初值。

$$X_{ij} = X_{\min,j} + (X_{\max,j} - X_{\min,j}) \times \text{betarnd}(a, b, 1, 1), \tag{8}$$

其中  $i=1,2,\dots,N; j=1,2,\dots,D$ ,betarnd 为 MATLAB 中产生 Beta 分布随机数的内置函数。

在普通 PSO 粒子初始化方法中,采用均匀分布的随机数,此时的初始化结果近似在搜索空间中均匀赋初值,不利于形成对最优解的包围,而基于 Beta 分布的种群初始化方法克服了这一不足。

### 2.2 惯性因子的设计

在 PSO 中,惯性因子对算法性能具有重要影响,若惯性因子大于 1,将很快导致算法发散,若惯性因子小于 0,将很快导致算法停滞。通常在算法初期,主要关注全局探索,这时需要使搜索过程迈大步,以实现大范围寻优,因此惯性因子应较大;而在算法后期,主要侧重局部开发,此时需要使搜索过程迈小步,以避免错过全局最优解,此时惯性因子应较小。为合理实现探索和开发两阶段的平衡,提出基于逆不完全伽马函数的惯性因子调整方法。下面首先给出逆不完全伽马函数的定义及特性。

不完全伽马函数  $\gamma(\lambda, a) = \int_0^{\lambda} e^{-t} t^{a-1} dt$  关于积分上限的逆函数,称为逆不完全伽马函数。在 Matlab 中的调用函数为 `gammaincinv( $\lambda, a$ )`,其中  $\lambda (\lambda \geq 0)$  是随机变量,  $a > 0$  是分辨率参数。当  $\lambda$  取定值,  $a$  在区间  $(0, 1]$  均匀取值时,  $(1/\lambda)\text{gammaincinv}(\lambda, 1-a)$  的特性如图 2 所示。

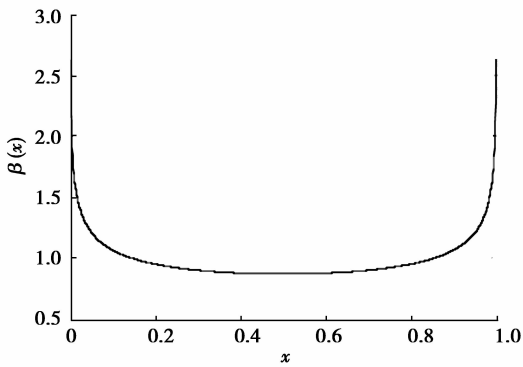


图 1 Beta 分布函数图像 ( $a=b=0.8$ )

Fig.1 Beta distribution function with  $a=b=0.8$

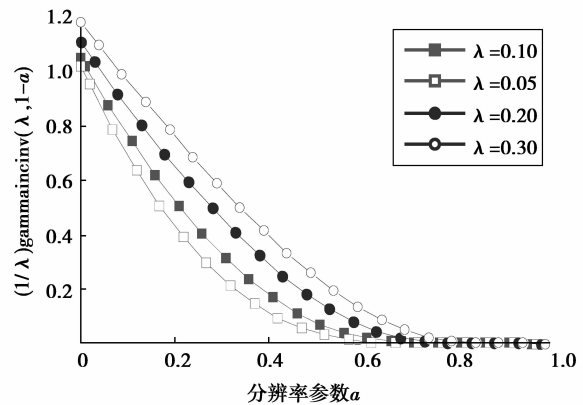


图 2  $(1/\lambda)\text{gammaincinv}(\lambda, 1-a)$  在  $a \in (0, 1]$  时的特性

Fig.2 The characteristics of  $(1/\lambda)\text{gammaincinv}(\lambda, 1-a)$

从图 2 可以得到这样的启示,若使  $\lambda$  取定值,而使  $a$  随迭代步数以等步长  $\Delta a$  从 0 线性增加到 1,则  $(1/\lambda)\text{gammaincinv}(\lambda, 1-a)$  具有从略大于 1 单调下降到接近于 0 的特性,这恰好符合对于调整惯性因子的要求。

经过初步探索,取  $\lambda=0.1, a=t/\text{MaxItr}$ ,其中  $t$  为当前迭代步数,  $\text{MaxItr}$  为限定迭代步数。此时逆不完全伽马函数具有这样一个优良特性:在迭代步数的前一半,接近线性下降,而在后一半,接近指数下降。用这样的函数来动态调整惯性因子,能够较好地实现探索与开发之间的平衡,进而有效避免早熟收敛<sup>[19]</sup>。

根据以上思路,惯性因子的调整方法可表述为下式。

$$\omega(t) = \omega_{\min} + \frac{\omega_{\max} - \omega_{\min}}{\lambda} \text{gammaincinv}\left(\lambda, 1 - \frac{t}{\text{MaxItr}}\right), \tag{9}$$

式中,  $\omega_{\max}$ ,  $\omega_{\min}$  分别为惯性因子最大值和最小值。

在普通粒子群算法速率更新式(1)中, 惯性因子  $\omega$  取定值, 或者全程随迭代步数线性下降, 就获得探索和开发之间的平衡而言, 显然不如基于逆不完全伽马函数的调整方法理想。

### 2.3 速度更新式的设计

对于速率的更新, 受差分进化思想启发, 引入了一个新算子, 该算子可表述为: 各自身最优粒子的算术平均值与当前粒子的差, 再乘以一个幅度随迭代步数单调下降的随机数, 如下式所示。

$$\delta_i(t) = \left(1 - \frac{t}{\text{MaxItr}}\right)^\alpha \times \text{rand} \times \left(\frac{1}{N} \sum_{i=1}^N P_i^L - X_i\right), \quad (10)$$

式中 rand 为在区间(0,1)均匀分布的随机数。

该算子的作用效果与  $\alpha$  的取值有关, 为便于分析该算子的特性, 首先给出  $(1-t/\text{MaxItr})^\alpha$  在不同  $\alpha$  取值下随迭代步数的变化情况, 如图 3 所示。

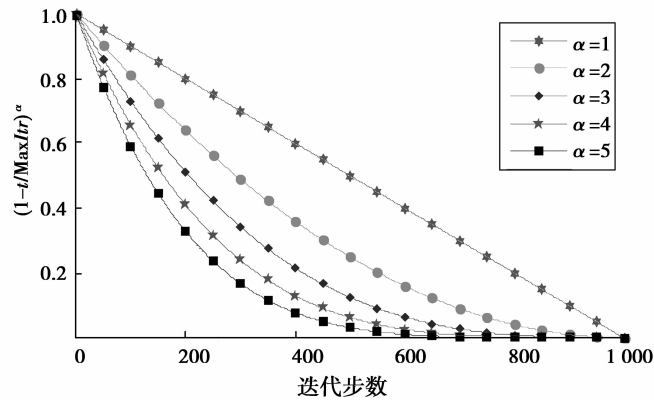


图 3 不同  $\alpha$  取值下  $(1-t/\text{MaxItr})^\alpha$  的动态特性

Fig.3 The characteristics of  $(1-t/\text{MaxItr})^\alpha$  under different  $\alpha$

由上图可知, 在算法的初期, 该算子可引领种群向着自身最优粒子的平均值逼近, 有利于全局探索; 随着迭代步数的增加, 该算子的作用指数减小, 有利于局部开发, 从而可有效实现探索和开发之间的合理平衡。大量实验表明, 当  $\alpha=5$  时优化性能最佳。因此在后续的数值函数优化中, 将该算子的参数固定为  $\alpha=5$ 。综上所述, 改进的速度更新式可表述为下式

$$V_i(t+1) = \omega(t)V_i(t) + c_1 r_1 (P_i^L - X_i(t)) + c_2 r_2 (P_g - X_i(t)) + \delta_i(t). \quad (11)$$

与普通粒子群算法速率更新式(1)对比可知, 改进的速率更新式中增加了一项幅度随迭代步数指数下降的随机项  $\delta_i(t)$ , 该随机项的核心作用在于进一步获得探索和开发之间的合理平衡。

### 2.4 粒子位置的越界处理

目前, 在绝大多数群智能优化算法中, 个体位置的越界处理均采用“若越出边界, 就等于边界”的简单方法, 该方法的优点是实现简单, 但由于边界点肯定不是全局最优解, 因此这种越界处理方式对优化进程不利。

鉴于此, 采用了基于边界对称映射的新方法。具体处理方法为: 令  $[X_{\min,j}, X_{\max,j}]$  为第  $j$  维寻优区间, 若  $X_{ij}$  越界, 则按下式处理。

$$X'_{ij} = \begin{cases} \min(X_{\max,j}, 2X_{\min,j} - X_{ij}), & \text{if } X_{ij} < X_{\min,j}, \\ \max(X_{\min,j}, 2X_{\max,j} - X_{ij}), & \text{if } X_{ij} > X_{\max,j}, \end{cases} \quad (12)$$

上述越界处理方法与普通 PSO 越界处理方法的差别可用图 4 直观表示。

尽管对传统 PSO 进行了多项改进, 但就控制参数的个数而言, 仅比传统 PSO 多了 2 个参数:  $\omega_{\max}$ ,  $\omega_{\min}$ 。

### 2.5 终止条件

终止条件可设置为精度阈值, 即当最优解达到某种精度要求后代终止; 也可以设置为限定步数, 即当达到限定步数之后, 不论是否满足精度要求迭代均终止, 研究选择限定步数作为终止条件。

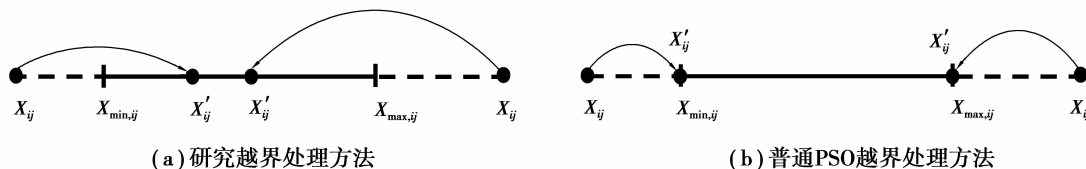


图 4 基于边界对称映射的越界处理方法

Fig.4 Cross boundary treatment based on symmetry mapping

### 2.6 改进 PSO 的算法流程

笔者提出的改进 PSO 算法的实施方案如下:

Step1 参数初始化,包括:种群规模,空间维数,全局因子,自身因子,惯性因子的上、下限,限定迭代步数。

Step2 按式(8)采用 Beta 分布随机数初始化种群;

Step3 按式(9)、(10)计算  $w(t)$  和  $\delta_i(t)$ ,按式(11)计算  $V_i(t+1)$ ,按式(2)计算  $X_i(t+1)$ ,按式(12)执行越界处理,计算目标函数值。

Step4 判断终止条件,若满足则结束,否则转 Step3。

上述流程可用图 5 直观描述。

## 3 对比实验

### 3.1 测试函数

为了充分验证提出改进算法的优势,采用了 50 个标准测试函数,这些测试函数均为极小值优化,每个测试函数的解析表达式、自变量取值范围、参数取值、精确极小值等信息均请参阅文献[20],其中  $f_1 - f_5$  为单峰可分离函数(unimodal separable, US);  $f_6 - f_{17}$  为单峰不可分离函数(unimodal non-separable, UN);  $f_{18} - f_{26}$  为多峰可分离函数(multimodal separable, MS);  $f_{27} - f_{50}$  为多峰不可分离函数(multimodal non-separable, MN)。

### 3.2 参数设置

用上述 50 个不同类型和维度的标准测试函数作为优化对象,用 IPSO 与基本 PSO、差分进化(DE)、人工蜂群(ABC)和量子行为粒子群(QPSO)算法的优化结果进行对比。为使对比公平,所有算法的种群规模均取 50,迭代次数均取 1 000。其中基本 PSO 的控制参数取  $c_1 = c_2 = 2.0$ ,惯性因子从 0.9 到 0.4 随迭代步数线性减小;IPSO 控制参数  $c_1 = c_2 = 2.0$ ,惯性因子最大值  $w_{max} = 0.9$ ,最小值  $w_{min} = 0.4$ ;DE 的缩放因子取  $\alpha = 0.6$ ,交叉概率取  $CR = 0.8$ ;ABC 的控制参数  $limit = 50$ ;QPSO 的控制参数取  $\alpha = 0.5$ 。为降低优化结果的随机性,对每个测试函数,分别用 5 种算法独立优化 50 次,以 50 次优化的平均结果作为对比指标。

### 3.3 结果对比

以 50 次独立优化的平均结果、平均时间作为对比指标。就优化的平均结果而言,对于 F1~F5,IPSO 优于 PSO 和 DE 的有 4 个,优于 ABC 的有 3 个,优于 QPSO 的有 2 个;对于 F6~F17,IPSO 优于 PSO 的有 9 个,优于 DE 的有 5 个,优于 ABC 的有 8 个,优于 QPSO 的有 10 个;对于 F18~F26,IPSO 的优化性能不够好,优于 PSO 的仅有 2 个,优于 DE 的仅有 4 个,优于 QPSO 的仅有 3 个,优于 ABC 的仅有 1 个;对于 F27~F50,IPSO 优于 PSO 的有 16 个,优于 DE 的有 18 个,优于 ABC 的有 9 个,优于 QPSO 的有 10 个。5 种算法

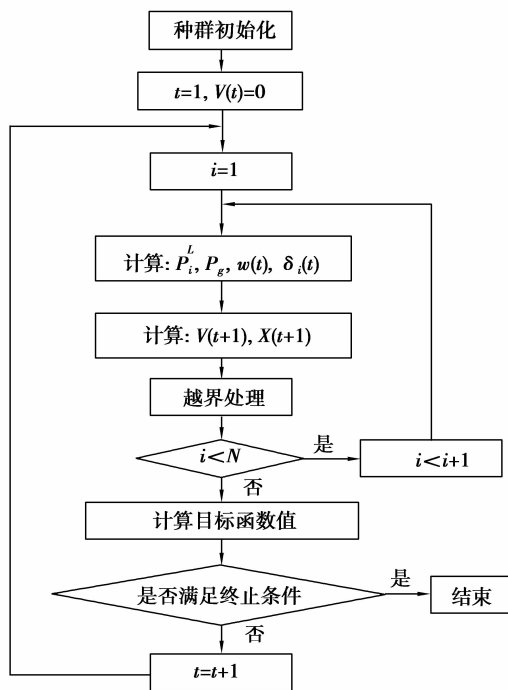


图 5 改进 PSO 的算法流程

Fig.5 Algorithm flow of improved PSO

优化的平均结果对比如图 6 所示,平均时间对比如图 7 所示。由于函数 14,15,23,50 各种方案优化得到的目标函数值相当大,为使图示清晰,图 6 中这 4 个函数没有给出。

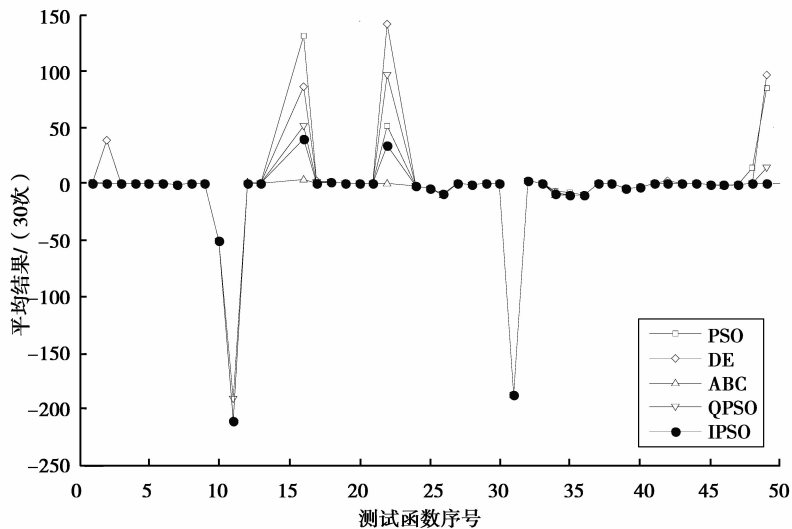


图 6 五种算法在相同迭代步数下的平均优化结果对比

Fig.6 Comparison of average optimal results of five algorithms under the same number of iteration steps

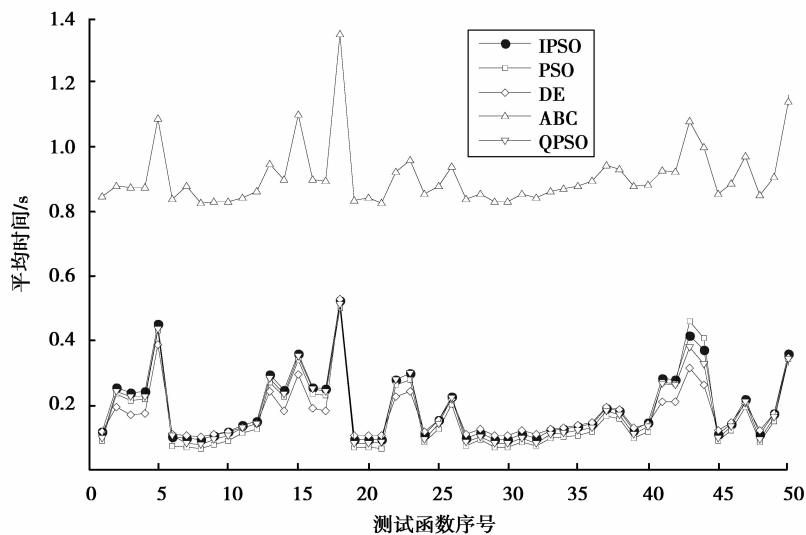


图 7 五种算法在相同迭代步数下的平均运行时间对比

Fig.7 Average running time comparison of five algorithms under the same number of iteration steps

就平均运行时间而言,IPSO 略长于除 ABC 以外的其他算法,这是由于引入了逆不完全伽马函数和一个基于差分进化的新算子导致了运算量增加的缘故。

为便于对比平均运行时间,首先将这 50 个测试函数的平均时间再取平均值,然后将该平均值作为对比的指标。IPSO 的平均时间为 PSO 的 1.118 9 倍,为 DE 的 1.097 2 倍,为 ABC 的 0.212 7 倍,为 QPSO 的 1.066 7 倍。为充分展示 IPSO 的优良性能,进一步考察相同运行时间下的对比结果。根据 IPSO 与其他算法平均时间的比值,首先将 PSO、DE、ABC、QPSO 的迭代步数分别设置为 1 200、1 200、500、1 200,然后对所有函数重新优化,每个测试函数分别用 5 种算法优化 50 次,然后取 50 次优化的平均结果作为对比指标,优化结果对比如图 8 所示。实验结果表明,IPSO 的性能与图 1 中展示的结果区别不大,仍然明显优于其他算法。

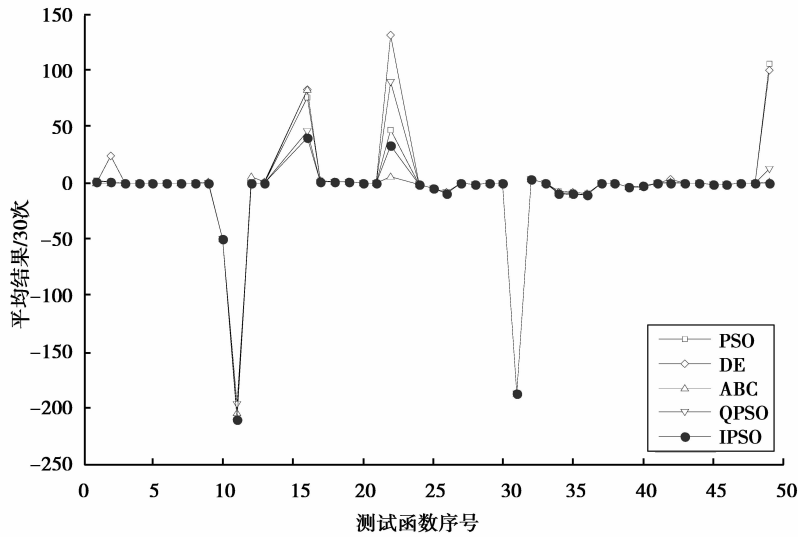


图 8 5 种算法在相同运行时间下的平均优化结果对比

Fig.8 Comparison of average optimal results of five algorithms under the same running time

为了更加客观公正的展示 IPSO 的优良性能,下面根据所有测试函数采用 IPSO 和 PSO、DE、ABC、QPSO 分别优化 50 次的结果,利用统计阈值  $\alpha=0.05$  进行秩和检验。该检验的原假设是:“对于同一函数,IPSO 的优化结果和其它算法的优化结果没有差异”。检验结果如表 1 所示,表中  $w$  列的“+、=、-”表示对于该行的函数,IPSO 在统计意义上分别优于、等于、劣于对比算法,NaN 表示 2 种算法对于该行的函数完全等价,为简洁直观,表中每类函数仅随机选取了 2 个。基于检验假设的统计结果表明,IPSO 有 20~40 个函数优于对比算法,而劣于对比算法的函数仅有 1~11 个。

从以上三类实验结果看,IPSO 表现出了相同的趋势。即对于单峰(变量可分离和不可分离)和多峰变量不可分离函数,IPSO 明显优于 PSO、DE、ABC 和 QPSO;而对于多峰变量可分离函数,IPSO 的优化性能不够理想。然而任何算法都不是万能的,不可能对于所有类型的问题都适用,这恰与无免费午餐定理的思想是一致的。对于 IPSO 表现出的优良特性,给出如下理论分析。

表 1 采用秩和检验的算法成对测试显著性结果对比 ( $\alpha=0.05$ )

Table 1 Comparison of pairwise test significance results based on rank sum test ( $\alpha=0.05$ )

序号	IPSO/PSO				IPSO/DE				IPSO/ABC				IPSO/QPSO			
	P	R <sub>1</sub>	R <sub>2</sub>	W	P	R <sub>1</sub>	R <sub>2</sub>	W	P	R <sub>1</sub>	R <sub>2</sub>	W	P	R <sub>1</sub>	R <sub>2</sub>	W
F3	7.07E-18	1275	3775	+	1.11E-05	1887	3163	+	7.07E-18	1275	3775	+	7.07E-18	1275	3775	+
F4	7.07E-18	1275	3775	+	1.79E-02	2181	2869	+	7.07E-18	1275	3775	+	7.07E-18	1275	3775	+
F8	4.73E-17	1307	3743	+	7.07E-18	3775	1275	-	7.07E-18	1275	3775	+	7.07E-18	1275	3775	+
F9	1.51E-02	2172	2878	+	3.31E-20	3775	1275	-	7.97E-18	1277	3773	+	4.32E-08	1730	3320	+
F21	NaN	1275	3775	=	NaN	1275	3775	=	3.31E-20	1275	3775	+	NaN	1275	3775	=
F22	2.92E-08	1720	3330	+	3.42E-12	1515	3535	+	7.07E-18	3775	1275	-	3.53E-17	1302	3748	+
F35	5.28E-04	1697	3353	+	7.88E-01	2071	2979	=	6.89E-13	1395	3655	+	9.94E-16	1375	3675	+
F36	3.38E-01	1704	3346	=	7.72E-01	1814	3236	=	8.61E-14	1393	3657	+	4.08E-16	1375	3675	+
+ / = / - 26/20/4																
21/18/11																
37/8/5																
40/9/1																

首先,从种群的初始化方法看,采用了基于 Beta 分布的随机数产生初始种群粒子位置的新方法。这种方法的优点在于,利用 Beta 分布函数的特性,能够以较大概率确保初始种群遍布于最优解周围,从而有利于使种群向着最优解逼近。而其他算法采用的基于均匀分布随机数的种群初始化方法,种群在优化空间的初始分布是随机的,不能呈现出对最优解的“包围”特性,从而在对最优解的逼近上略逊色于 IPSO。

第二,在搜索过程中,探索 and 开发两阶段的合理平衡往往是制约算法性能的关键因素。从 IPSO 惯性因子的更新方式看,采用了基于逆不完全伽马函数的新方法,逆不完全伽马函数的独特性能使惯性因子的下降趋势,较好的符合了算法在全局探索 and 局部开发两阶段之间的平衡,从而使优化性能得到进一步提高。

第三,从粒子的速率更新式看,研究在传统 PSO 速率更新的基础上,受差分进化思想的启发,增加了一个基于所有粒子自身最优平均值与当前粒子之差的新算子。在算法初期的全局探索阶段,该算子的作用幅度较大,有利于发现较好的目标区域;随着迭代步数逐渐减小,其作用幅度逐渐减小,直至为零,有利于在最优解附近局部开发,进一步保证了探索 and 开发之间的合理平衡。

第四,越界处理是所有群智能优化算法都面临的问题,该问题也对算法性能有较大影响,若不及时处理会使算法迅速发散,若处理不当则会使算法陷入早熟收敛。采用的基于边界点对称映射的粒子越界处理方法,使越界的粒子立即返回寻优空间,而不是停留在边界,这种处理方法也使寻优速度得到了进一步的提高。

## 4 结 论

提出了一种改进的粒子群优化算法。该算法主要针对种群初始化方法、惯性因子的更新方法、粒子速率更新式的设计、例子的越界处理方法 4 个方面进行了改进。改进的目标是有效实现寻优过程中全局探索 and 局部开发 2 阶段的合理平衡,采用的主要方法是在惯性因子的更新式中引入了逆不完全伽马函数,在速率更新式中,引入了基于差分进化的新算子。标准函数极值优化的实验结果验证了提出方法的优越性,从而揭示出本文提出的改进措施是有效的可行的。

### 参考文献:

- [1] Kennedy J, Eberhart R C. Particle swarms optimization[C]// Proceedings of IEEE international conference on Neural Networks. Perth. WA, USA: IEEE, 1995: 1942-1948.
- [2] Shi Y H, Eberhart R C. A modified particle swarm optimizer[C]// Proceedings of the 1998 IEEE International Conference on Evolutionary Computation, Piscataway, USA: IEEE, 1998: 67-73.
- [3] Cui G Z, Li X G, Zhang X C, et al. The optimization of DNA encodings based on modified PSO/GA algorithm[J]. Chinese Journal of Computers, 2010, 33(2): 311-316.
- [4] Zhand C S, Sun J G, Ouyang D T, et al. A self-adaptive hybrid particle swarm optimization algorithm for flow shop scheduling problem[J]. Chinese Journal of Computers, 2009, 32(11): 2137-2146.
- [5] Jiang M, Luo Y P, Yang S Y. Stochastic convergence analysis and parameter selection of the standard particle swarm optimization algorithm[J]. Information Processing Letters, 2007, 102(1): 8-16.
- [6] Bergh F, Engelbrecht A P. A study of particle swarm optimization particle trajectories[J]. Information Science, 2006, 176(8): 937-971.
- [7] Cai X J, Cui Z H, Zeng J C, et al. Dispersed particle swarm optimization[J]. Information Processing Letters, 2008, 105(6): 231-235.
- [8] Lu Z S, Hou Z R. Particle swarm optimization with adaptive mutation[J]. Acta Electronica Sinica, 2004, 32(3): 416-420.
- [9] Liu Y, Qin Z, Shi Z W, et al. Center particle swarm optimization[J]. Neurocomputing, 2007, 70(4-6): 672-679.
- [10] Liu B, Wang L, Jin Y H, et al. Improved particle swarm optimization combined with chaos[J]. Chaos Solitons &



- Fractals, 2005, 25(5): 1261-1271.
- [11] Luo Q, Yi D Y. A co-evolving framework for robust particle swarm optimization[J]. Applied Mathematics and Computation, 2008, 199(2): 611-622.
- [12] Jiang Y, Hu T S, Huang C C, et al. An improved particle swarm optimization algorithm[J]. Applied Mathematics and Computation, 2007, 193(1): 231-239.
- [13] Li L L, Wang L, Liu L H. An effective hybrid PSOGA strategy for optimization and its application to parameter estimation[J]. Applied Mathematics and Computation, 2006, 179(1): 135-146.
- [14] 赵新超, 刘国莅, 刘虎球, 等. 基于非均匀变异和多阶扰动的粒子群优化算法[J]. 计算机学报, 2014, 37(9): 2058-2070.  
ZHAO Xinchao, LIU Guoli, PENG Daogang, et al. Particle swarm optimization algorithm based on non-uniform mutation and multiple stages perturbation[J]. Chinese Journal of Computers, 2014, 37(9): 2058-2070. (in Chinese)
- [15] 钱玉良, 张浩, 彭道刚, 等. 基于 EMD 调制和粒子群模型的发电机组轴心轨迹提纯[J]. 信息与控制, 2013, 42(2): 243-251.  
QIAN Yuliang, ZHANG Hao, PENG Daogang, et al. Purifying the orbit of generator unit based on EMD modulation and a particle swarm model[J]. Information and Control, 2013, 42(2): 243-251. (in Chinese)
- [16] 白国振, 荆鹏翔. 基于改进粒子群算法的并联机械手运动学参数辨识[J]. 信息与控制, 2015, 44(5): 545-551.  
BAI Guozhen, JING Pengxiang. Kinematic parameter identification of parallel manipulator based on improved particle swarm algorithm[J]. Information and Control, 2015, 44(5): 545-551. (in Chinese)
- [17] Bergh F, Engelbrecht A P. A study of particle swarm optimization particle trajectories[J]. Information Sciences, 2006, 176(8): 937-971.
- [18] Chatterjee A, Siarry P. Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization[J]. Computers & Operations Research, 2007, 33(3): 859-871.
- [19] Kerman. GSA: A gravitational search algorithm[J]. Information Sciences, 2009, 179(1): 2232-2248.
- [20] Berat D, Tamer O. A new metaheuristic for numerical function optimization: vortex search algorithm[J]. Information Sciences, 2015, 293(1): 125-145.

(编辑 侯 湘)