基于 PSO 算法的 SOR 最优松弛因子选取研究

薛 丹,姚若侠
(陕西师范大学 计算机科学学院,陕西 西安 710119)

摘 要:目前选取逐次超松弛迭代法(SOR)最优松弛因子的基本思路是:在区间(0,2)上,根据确定的分割策略,选取分割点的值作为松弛因子来计算相应的 SOR 迭代次数,将小于预设的 SOR 迭代次数阈值的松弛因子作为最优解返回,例如二分比较法、黄金分割法、逐步搜索法等,其缺陷在于不易找到全局最优松弛因子且对参数依赖较大。为克服传统策略解决该问题的不足,受粒子群优化算法及其在不同场景成功应用的启发,提出利用基本粒子群优化算法(bPSO)、简化粒子群优化算法(sPSO)、带极值扰动粒子群优化算法(tPSO)和带极值扰动的简化粒子群优化算法(tsPSO)来搜索 SOR 迭代法最优松弛因子。通过对两个不同的线性方程组的实证测试,验证了四种算法在选取 SOR 最优松弛因子问题上的有效性。

关键词:粒子群优化算法;简化粒子群优化算法;带极值扰动粒子群优化算法;SOR 迭代法;最优松弛因子
中图分类号:TP301 **文献标识码**:A **文章编号**:1673-629X(2020)12-0015-06
doi:10.3969/j.issn.1673-629X.2020.12.003

Study on Selecting SOR Optimal Relaxation Factor with Particle Swarm Optimization

XUE Dan, YAO Ruo-xia
(School of Computer Science, Shaanxi Normal University, Xi'an 710119, China)

Abstract: At present, the basic idea of selecting SOR optimal relaxation factor is as follows: in the interval (0,2), the value of a split point is selected as the relaxation factor to calculate the corresponding SOR iteration number, and the relaxation factor less than the preset SOR iteration threshold is returned as the optimal solution, such as dichotomous comparison method, golden section method, stepwise search method, and so on. However, this strategy is hard to find the global optimal one and heavily depends on parameter setting. In order to solve the problem above, inspired by the particle swarm optimization and its successful application in different scenes, we propose to use the basic particle swarm optimization (bPSO), the simple particle swarm optimization (sPSO), the extremum disturbed particle swarm optimization (tPSO) and the extremum disturbed and simple particle swarm optimization (tsPSO) for finding SOR optimal relaxation factor. By testing the two different linear equations, we verify the validity of four algorithms in selecting SOR optimal relaxation factor.

Key words: particle swarm optimization; simple particle swarm optimization; extremum disturbed particle swarm optimization; SOR iterate algorithm; optimal relaxation factor

0 引 言

在科学和工程领域中,许多重要的问题常常可以归结为对大型稀疏线性代数系统的求解问题。这类线性代数系统的系数矩阵阶数较高,采用直接法进行 Gauss 消去或者是矩阵的三角分解,计算量大并且复杂度高,对于这种情况,通常采用迭代法求解[1]。迭代法是指从一个初始向量出发,按照给定的迭代公式,逐次迭代来逼近方程组的解,因此,迭代法是否收敛,收敛速度以及迭代次数成为衡量迭代算法非常重要的指标。逐次超松弛迭代法(successive overrelaxation, SOR)是对 Gauss-Seidel(G-S)迭代法的改进,由 Frankel 和 Young 提出,随后被广泛应用于求解核反应扩散、石油天然气存储、天气预测等大型实际问题的数学模型[2]。求解方程组 $Ax = b$ 的 SOR 迭代公式如下[1]:

$$x_i^{(k)} = x_i^{(k-1)} + \frac{w}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i}^{n} a_{ij}x_j^{(k-1)}\right) \tag{1}$$

其中,$k = 1,2,\cdots$ 表示每一步迭代计算;$i = 1,2,\cdots n$,x_i 表示解向量 x 的各个分量。松弛因子 w 是影响 SOR

收稿日期:2020-01-02 **修回日期**:2020-05-06
基金项目:国家自然科学基金面上项目(11471004,61673251)
作者简介:薛 丹(1991-),女,硕士,研究方向为智能计算、大数据分析;姚若侠,教授,博导,研究方向为符号计算、智能计算、模式识别。

迭代法的关键，w 选取恰当，可以使 SOR 迭代法快速收敛，而不恰当的 w 可能减缓 SOR 迭代法的收敛速度甚至导致发散。因此，研究如何选取最优的松弛因子 w 在数值代数领域具有重要意义。

定理 1[1]：若 SOR 迭代收敛，则松弛因子 w 需满足 $w \in (0,2)$。

文献[3]指出，通过数学公式确定最优松弛因子 w 具有一定难度，因而另辟蹊径提出通过计算机算法选取最优松弛因子，即在区间(0,2)上分别采用二分比较法、黄金分割法、逐步搜索法选取最优松弛因子 w。通过对这三种算法的研究和分析，发现传统的分割算法等虽然容易实现，但却对参数设置有较强的依赖，且无法保证在(0,2)区间上可以成功选取全局最优松弛因子。因此，该文根据群智能优化算法可以在解空间根据迭代规则智能靠近最优解的思路，拟通过粒子群优化算法(particle swarm optimization，PSO)来实现在(0,2)区间上选取 SOR 迭代法最优松弛因子 w，以进一步推动通过计算机算法选取 SOR 最优松弛因子 w 的研究发展。

1 粒子群优化算法

粒子群优化算法是由 Eberhart 和 Kennedy 于 1995 年提出的一类基于群智能的随机优化算法[4]。PSO 算法的基本思想来源于对鸟群捕食行为的研究：一群鸟在随机搜寻食物，如果这个区域里只有一块食物，那么找到食物的最简单有效的策略是，搜寻目前离食物最近的鸟的周围区域。粒子群优化算法自提出以来，由于形式简单，实现容易，需要调整的参数较少，已被广泛应用于多个学科和工程领域[5-6]。

1.1 PSO 算法

PSO 算法首先初始化一组种群数为 n 的随机粒子，每个随机粒子代表 D 维解空间的一个解，粒子 i 在 d 维的位置用 x_{id} 表示，粒子 i 在 d 维的速度用 v_{id} 表示，粒子通过改变 v_{id} 来改变自己的位置，不断靠近潜在最优解。v_{id} 的更改受到两个极值的影响：一个极值是粒子本身找到的最优解，称为个体极值，记为 pbest_{id}；另一个极值是整个种群找到的最优解，称为全局极值，记为 gbest_d[4]。v_{id} 体现了粒子在群体中追随潜在最优解流动的速度和方向，文献[4]最早提出的粒子群追踪潜在最优解的进化方程如下式所示[4]：

$$v_{id}^{t+1} = v_{id}^{t} + c_1 r_1(\text{pbest}_{id} - x_{id}) + c_2 r_2(\text{gbest}_d - x_{id}) \quad (2)$$

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1} \quad (3)$$

其中，$i = 1,2,\cdots,n$；$d = 1,2,\cdots,D$；r_1 和 r_2 是[0,1]之间的随机数；学习因子 c_1、c_2 为非负常数，通常取 $c_1 = c_2 = 2$；$v_{id} \in [-v_{\max}, v_{\max}]$，$v_{\max}$ 是用户设定的常数，表示速度更改的最大值。进化终止条件为预设的最大迭代次数或预定的最小适应度阈值，该算法也被称为全局版 PSO 算法。

1.2 PSO 算法的发展

全局版 PSO 算法虽然形式简单、易于理解、容易实现且能快速收敛至潜在最优解，但存在容易陷入局部极值、进化后期收敛速度慢等问题，许多学者也一直致力于研究如何提高 PSO 算法的寻优能力。1998 年，Yuhui Shi 和 Russell Eberhart 通过给式(3)添加惯性权重(weight)来优化 PSO 算法的能力，并通过实验得出结论：当惯性权重在区间[0.9,1.2]时，PSO 算法有较高的概率可以搜索到全局最优解[7]，修改后的进化方程如式(4)和式(5)[7]式所示：

$$v_{id}^{t+1} = \text{weight} * v_{id}^{t} + c_1 r_1(\text{pbest}_{id} - x_{id}) + c_2 r_2(\text{gbest}_d - x_{id}) \quad (4)$$

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1} \quad (5)$$

很多学者将式(4)和式(5)称为基本粒子群优化算法(basic particle swarm optimization，bPSO)。

2006 年，胡旺等提出简化粒子群优化算法(simple particle swarm optimization，sPSO)[8]，sPSO 算法仅由粒子位置控制进化过程。实验证明，该算法能够有效避免粒子群进化后期收敛速度变慢和精度低等问题[8]。sPSO 算法的进化方程为[8]：

$$x_{id}^{t+1} = \text{weight} * x_{id}^{t} + c_1 r_1(\text{pbest}_{id} - x_{id}^{t}) + c_2 r_2(\text{gbest}_d - x_{id}^{t}) \quad (6)$$

同时，为了克服 bPSO、sPSO 算法在进化后期容易陷入局部极值的缺点，提出了带极值扰动的粒子群优化算法(extremum disturbed particle swarm optimization，tPSO)[8]和带极值扰动的简化粒子群优化算法(extremum disturbed and simple particle swarm optimization，tsPSO)[8]。tPSO、tsPSO 算法的基本思路是：在粒子群进化方程中，通过扰动因子 r_3、r_4 分别对个体极值和全局极值添加随机扰动，以提升算法跳出局部极值的能力。具体扰动策略为：将粒子群的进化停滞步数作为随机扰动的触发条件，用 t_0 和 t_g 分别表示个体极值和全局极值进化停滞步数，用 T_0 和 T_g 分别表示个体极值和全局极值需要扰动的停滞步数阈值。T_0 和 T_g 的具体取值由用户设定，当 $t_0 \leqslant T_0$ 时，个体极值不需要扰动，$r_3 = 1$；否则对个体极值进行随机扰动，$r_3 = U(0,1)$。同理，当 $t_g \leqslant T_g$ 时，全局极值不需要扰动，$r_4 = 1$；否则对全局极值进行随机扰动，$r_4 = U(0,1)$。tPSO 算法的进化方程为：

$$v_{id}^{t+1} = \text{weight} * v_{id}^{t} + c_1 r_1(r_3 * \text{pbest}_{id} - x_{id}) + c_2 r_2(r_4 * \text{gbest}_d - x_{id}) \quad (7)$$

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1} \quad (8)$$

tsPSO 算法的进化方程为：

$$x_{id}^{t+1} = \text{weight} * x_{id}^{t} + c_1 r_1 (r_3 * \text{pbest}_{id} - x_{id}^{t}) + c_2 r_2 (r_4 * \text{gbest}_{d} - x_{id}^{t}) \tag{9}$$

扰动因子 r_3、r_4 的取值分别为:

$$r_3 = \begin{cases} 1, t_0 \leqslant T_0 \\ U(0,1), t_0 > T_0 \end{cases} \tag{10}$$

$$r_4 = \begin{cases} 1, t_g \leqslant T_g \\ U(0,1), t_g > T_g \end{cases} \tag{11}$$

通过对粒子群优化算法的研究及应用现状的调查和分析[9-14],选取 bPSO、sPSO、tPSO、tsPSO 等四个算法分别应用在(0,2)区间上针对不同方程组 $Ax = b$ 选取 SOR 迭代法的最优松弛因子 w,实验具体设计、部署和结果分析将在下文详细介绍。

2　实验与分析

2.1　实验环境及数据准备

实验选取两个不同的线性方程组[1,3,15],应用以上四个算法对两个线性方程组分别求出最优 SOR 松弛因子 w,在 SOR 迭代过程中,假定当 $\max|\Delta x_i| = \max|x_i^{(k+1)} - x_i^{(k)}| < 10^{-6}, i = (1,2,\cdots,n)$ 时迭代终止,如果 SOR 迭代 500 次仍无法满足迭代终止条件,则认为 SOR 发散。

实验环境信息如下:Win 10 操作系统,Intel® Core (TM)i7-8550U CPU 处理器,8 GB 内存,Python 开发语言,Python3.5 解释器版本,PyCharm2018.1.4 开发环境。

算法的参数设置如下:种群粒子数=3,进化世代数阈值=50,$c_1 = c_2 = 2$,惯性系数:weight=0.9,bPSO、tPSO 算法中速度更改的最大值 $v_{\max} = 0.3$;sPSO、tsPSO 算法中粒子位置更改的最小值=0.001,最大值=1.999;tPSO、tsPSO 算法中个体极值需要扰动的停滞步数阈值 $T_o = 3$,全局极值需要扰动的停滞步数阈值 $T_g = 5$。实验中用到的两个形为 $Ax = b$ 的线性方程组数据[1,3,16]如下:

$$A[1] = \begin{bmatrix} 5 & 1 & -1 & -2 \\ 2 & 8 & 1 & 3 \\ 1 & -2 & -4 & -1 \\ -1 & 3 & 2 & 7 \end{bmatrix}$$

$$b[1] = [-2, -6, 6, 12]^{\mathrm{T}}$$

$$S[1] = [1, -2, -1, 3]^{\mathrm{T}}$$

$$A[2] = \begin{bmatrix} 4 & -1 & 0 & -1 & 0 & 0 \\ -1 & 4 & -1 & 0 & -1 & 0 \\ 0 & -1 & 4 & 0 & 0 & -1 \\ -1 & 0 & 0 & 4 & -1 & 0 \\ 0 & -1 & 0 & -1 & 4 & -1 \\ 0 & 0 & -1 & 0 & -1 & 4 \end{bmatrix}$$

$$b[2] = [0,5,0,6, -2,6]^{\mathrm{T}}$$

$$S[2] = [1,2,1,2,1,2]^{\mathrm{T}}$$

其中,$S[i], i = 1,2$ 是方程组的精确解。

2.2　算法设计

应用 bPSO、sPSO、tPSO、tsPSO 算法在(0,2)区间上对上述两个线性方程组分别求最优松弛因子 w,算法的程序流程如图 1 ~ 图 4 所示。

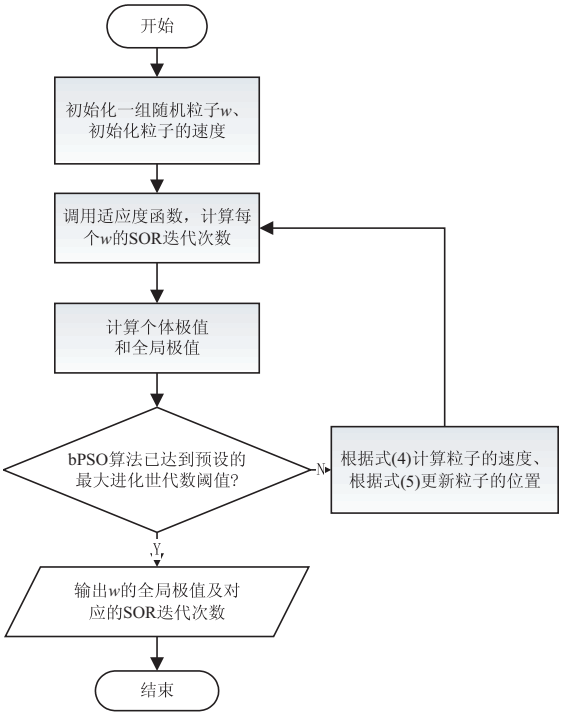


图 1　bPSO 算法程序流程

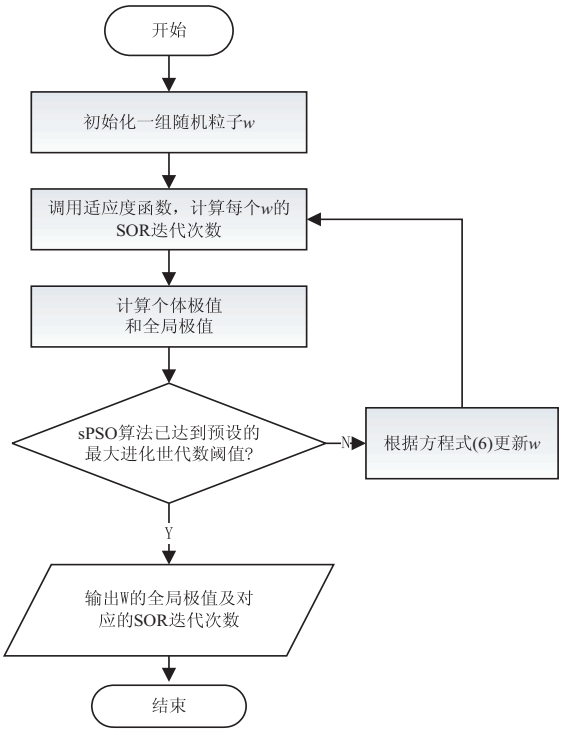


图 2　sPSO 算法程序流程

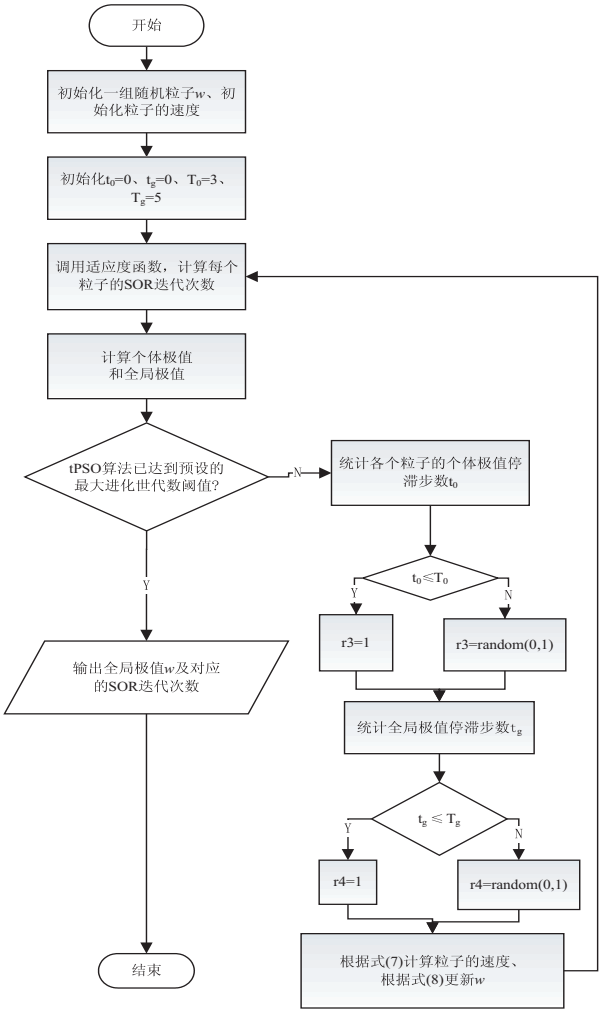


图 3 tPSO 算法程序流程

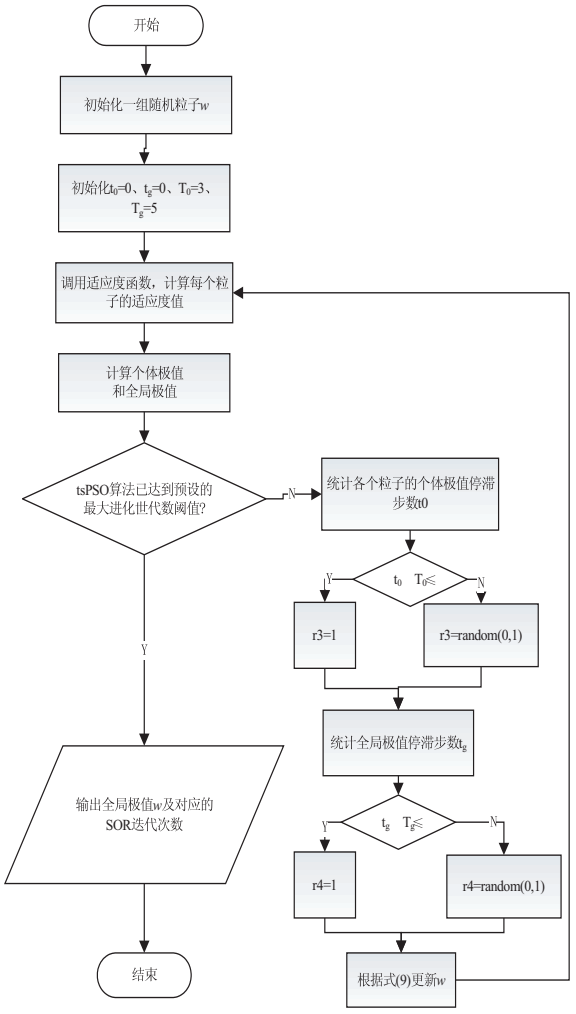


图 4 tsPSO 算法程序流程

2.3 实验结果及分析

运行以上四个算法，分别对两个线性方程组在(0,2)区间各执行一次搜索 SOR 迭代法最优松弛因子 w，程序执行结果见表 1。由表 1 数据可知：

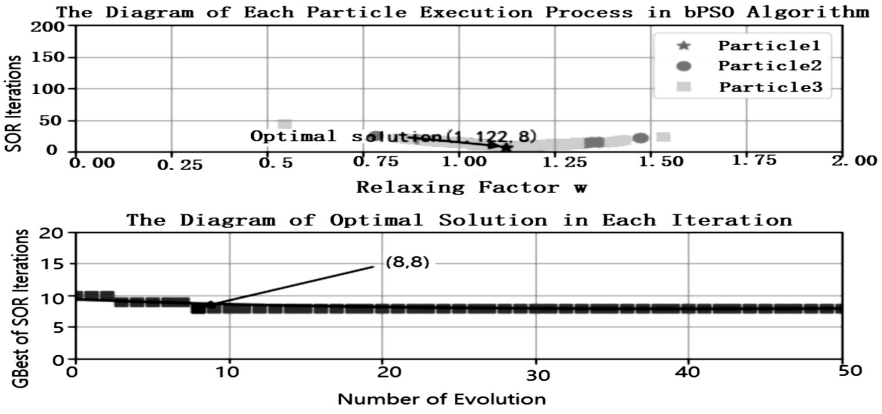
(1)实验所选 4 个算法，对于两个不同的线性方程组均能有效地在(0,2)区间找到 SOR 最优松弛因子，且搜索出的 w 对应的 SOR 迭代次数均较小，近似解也都满足与精确解之间的精度要求，即 $\max|\Delta x_i| = \max|x_i^{(k+1)} - x_i^{(k)}| < 10^{-6}$。

(2)相对于文献[3]中应用二分查找法、黄金分割法等搜索方程组 1 的 SOR 最优松弛因子 w，该文应用 bPSO、sPSO、tPSO、tsPSO 算法搜索的 w 对应的 SOR 迭代次数为 8 次，优于文献[3]中的 12 次。

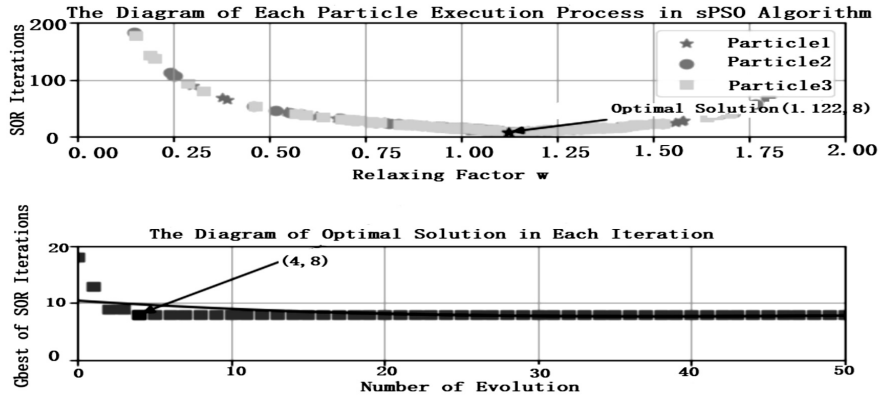
表 1 四种算法选取最优 w 的实验结果

方程编号	算法	w	SOR 迭代次数	$\max\|\Delta x_i\|$	近似解
1	bPSO	1.118	8	2.960×10^{-7}	$(1,-2,-1,3)^T$
	sPSO	1.140	8	6.698×10^{-7}	$(1,-2,-1,3)^T$
	tPSO	1.116	8	5.030×10^{-7}	$(1,-2,-1,3)^T$
	tsPSO	1.143	8	9.495×10^{-7}	$(1,-2,-1,3)^T$
2	bPSO	1.122	8	8.620×10^{-7}	$(1,2,1,2,1,2)^T$
	sPSO	1.122	8	8.620×10^{-7}	$(1,2,1,2,1,2)^T$
	tPSO	1.119	8	5.223×10^{-7}	$(1,2,1,2,1,2)^T$
	tsPSO	1.122	8	8.620×10^{-7}	$(1,2,1,2,1,2)^T$

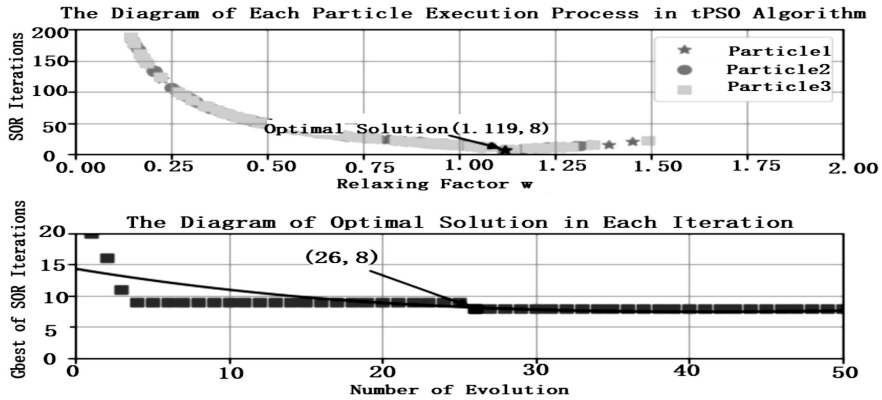
该文在实验编码过程中调用 Python 的 matplotlib 库，对粒子的执行过程进行了可视化展示。仅列出以上四个算法在搜索方程组 2 的 SOR 最优松弛因子时的执行过程，见图 5。



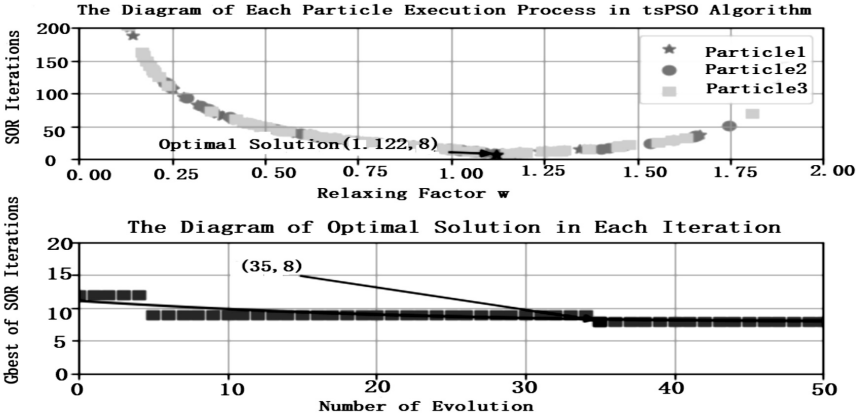
(a)bPSO 算法



(b)sPSO 算法



(c)tPSO 算法



(d)tsPSO 算法

图 5　四个算法对方程组 2 搜索 SOR 最优松弛因子执行过程

由图 5(a)可以看出,bPSO 算法搜索方程组 2 最优松弛因子 w 时,主要搜索区间集中在(0.5,1.5),在进化第 8 次时找到最优松弛因子 1.122,对应的 SOR 迭代次数为 8 次。图 5(b)反映了 sPSO 算法实际搜索区间在(0,2),覆盖的解区间的范围更广,粒子在进化第 4 次时找到最优松弛因子 1.122,对应的 SOR 迭代次数为 8 次。图 5(c)反映了 tPSO 算法主要搜索区间在(0,1.5),在进化第 26 次时找到最优松弛因子 1.119,对应的 SOR 迭代次数为 8 次。图 5(d)反映了 tsPSO 算法主要搜索区间在(0,2),几乎覆盖全部解区间,其结果最具全局最优性,且粒子在进化第 35 次时找到最优松弛因子 1.122,对应的 SOR 迭代次数为 8 次。

3 结束语

应用 bPSO、sPSO、tPSO、tsPSO 等四个算法分别对两个不同的线性方程组搜索 SOR 最优松弛因子 w,获得以下结论:

(1)bPSO、sPSO、tPSO、tsPSO 算法均成功实现在(0,2)区间上选取 SOR 全局最优松弛因子 w,且实验结果优于文献[3]中的二分比较法等算法。

(2)对于同一个方程组的应用中,bPSO 算法的实际搜索区间较小,容易陷入局部极值。sPSO 算法、tPSO 算法、tsPSO 算法,实际搜索范围相较 bPSO 算法有所扩大,其中 tsPSO 算法的搜索范围几乎覆盖了全部解区间,其搜索结果也更具全局最优性。

参考文献:

[1] 同济大学应用数学系. 工程数学数值分析与矩阵论(上册)[M]. 上海:同济大学出版社,2012.

[2] 沈海龙. 线性代数系统迭代解法与预条件方法研究[D]. 沈阳:东北大学,2013.

[3] 胡 枫,金远平. SOR 最优松弛因子选取方法研究[J]. 西南师范大学学报:自然科学版,2008,33(5):48-50.

[4] KENNEDY J, EBERHART R. Particle swarm optimization[C]//Proceedings of ICNN'95 - international conference on neural networks. Perth Australia:IEEE,1995:1942-1948.

[5] EBERHART R, SHI Y. Particle swarm optimization: developments, applications and resources[C]//Proceedings of the 2001 congress on evolutionary computation. Seoul South Korea:IEEE,2001:81-86.

[6] 杨 维,李歧强. 粒子群优化算法综述[J]. 中国工程科学,2004,6(5):87-94.

[7] SHI Y, EBERHART R. A modified particle swarm optimizer[C]//1998 IEEE international conference on evolutionary computation proceedings. Anchorage, USA:IEEE,1998:69-73.

[8] 胡 旺,李志蜀. 一种更简化而高效的粒子群优化算法[J]. 软件学报,2007,18(4):861-868.

[9] 宋文文,王 珺,杜 晔,等. 基于粒子群优化的数据中心负载均衡机制[J]. 南京邮电大学学报:自然科学版,2019,39(5):81-88.

[10] 郁 诺,刘 洋. 基于粒子群优化算法的高速网络可变结构节点位置控制[J]. 吉林大学学报:理学版,2019,57(5):1179-1184.

[11] 严宇翔,胡伍生. 粒子群优化算法在匈牙利水汽层析中的应用研究[J]. 测绘工程,2019,28(6):6-9.

[12] 王立志,慕晓冬,刘宏岚. 采用改进粒子群优化的 SVM 方法实现中文文本情感分类[J]. 计算机科学,2020,47(1):231-236.

[13] 谷 鹏,王大龙,张世仓. 基于粒子群优化的扩展卡尔曼滤波方法研究[J]. 工业控制计算机,2019,32(11):80-82.

[14] 李 鹏,李兵舰,亓 亮,等. 一种改进的粒子群优化算法及其在无人机航路规划中的应用[J]. 舰船电子对抗,2019,42(5):59-64.

[15] 北京大学数学系几何与代数教研室前代数小组. 高等代数[M]. 第 3 版. 北京:高等教育出版社,2006.