二维泊松方程的遗传 PSOR 改进算法*

彭武[1)†] 何怡刚[1)2)] 方葛丰[3)] 樊晓腾[3)]

1) (湖南大学电气与信息工程学院, 长沙 410082)

2) (合肥工业大学电气与自动化工程学院, 合肥 230009)

3) (电子测试技术国防科技重点实验室, 青岛 266555)

(2012 年 5 月 27 日收到; 2012 年 8 月 31 日收到修改稿)

针对二维泊松方程在实际应用过程中几种常用方法存在计算量大、易发散、局部收敛等不足, 提出了一种改进算法. 该算法基于并行超松弛迭代法, 采用遗传算法对松弛因子进行全局寻优, 解决了超松弛迭代法求解泊松方程时最佳松弛因子难以确定的问题. 构建了多目标适应度函数, 优化了遗传算子参数, 分析了算法的计算量、计算时间与误差精度, 与传统方法进行了对比研究. 结果表明: 松弛因子对泊松方程求解的速度与精度影响显著; 改进算法能减少迭代次数, 节省计算时间, 加快方程的求解; 算法适合于求解计算量较大、精度要求较高的时域有限差分方程, 而且精度要求越高, 算法的性能越好, 节省的时间也越多.

关键词: 泊松方程, 遗传算法, 并行超松弛迭代法, 有限差分法

PACS: 03.50.De, 41.20.Cv, 02.70.Bf **DOI:** 10.7498/aps.62.020301

1 引 言

泊松方程是数学中一个常见的椭圆型偏微分方程, 诸多物理现象如电磁, 热传导和流体力学等, 都可用泊松方程来描述 [1]. 泊松方程的常用解法有 3 种: 第 1 种是格林函数法 [2]. 格林函数可以将微分方程边值问题转化为积分方程问题, 但对于有限域的泊松方程, 因为找不到其对应的格林函数, 故该法求解比较困难. 第 2 种是分离变量法 [3]. 分离变量法是求解数学物理方程中应用最广泛的一种方法, 但该法在应用时坐标系的选择有一定限制, 当所求解模型的边界面与某坐标系统的坐标面相符合, 或者至少分段地与坐标面相符合时才能使用. 第 3 种是有限差分法 [4]. 该方法是最早且很好的求解方法, 对于研究抛物形和椭圆形问题, 受到人们的关注和重视. 但有限差分方法对于椭圆形问题的逼近往往需要求解较大的稀疏矩阵, 数据处理也很复杂.

众所周知, 泊松方程的有限差分逼近往往构成一个较大的线性方程组, 对于这样大规模的计算, 迭代方法 [5−8] 被认为是求解此类方程组最为合适的方法. 考虑到成本与速度, 大容量的并行机和有效的数值并行方法及并行算法起到了非常重要的作用. 并行超松弛迭代算法 [9] (parallel successive over-relaxation iteration, PSOR) 因其有明显的并行性, 能大大提高计算效率、节省计算时间、减少迭代次数, 是一种比常用迭代方法如雅可比 (Jacobi)、高斯-赛德尔 (Gauss-Seidel, G-S)、逐次超松弛迭代 (successive over-relaxation iteration, SOR) 更优的方法. 同时遗传算法 [10,11] (genetic algorithm, GA) 借鉴达尔文的物竞天择、优胜劣汰、适者生存的自然选择和自然遗传机理, 通过模拟生物自然进化过程搜索最优解, 具有高度并行、随机、全局自适应搜索的特点. 本文提出将遗传算法与 PSOR 结合的改进方法, 既保留了上述几种方法的优点, 也克服

* 国家杰出青年科学基金 (批准号: 50925727)、国家自然科学基金 (批准号: 60876022, 61102039, 51107034)、湖南省科技计划项目 (批准号: 2011J4, 2011JK2023)、国防预研重大项目 (批准号: C1120110004)、广东省教育部产学研计划 (批准号: 2009B090300196) 和中央高校基本科研业务费资助的课题.

† 通讯作者. E-mail: pengwu@hnu.edu.cn

了它们的缺陷, 在保证计算精度的前提下, 提高了方程的收敛速度, 节省了计算时间.

本文介绍了 PSOR 算法求解二维泊松方程的优缺点; 分析了二维泊松方程 PSOR 算法的改进方法, 利用多目标 GA[12,13] 对松弛因子进行全局寻优, 对遗传算子的参数进行了优化配置; 把改进的方法与 Jacobi, Gauss-Seidel, SOR 以及 PSOR 进行对比, 分析了各算法的性能, 最后给出数值算例验证算法的有效性和精确性.

2 二维泊松方程 PSOR 算法

如图 1 所示, 用正方形网格划分场域 D, 步长为 h, 中心节点 0 上的位函数用 $\varphi_{i,j}$ 表示, 节点 1, 2, 3 和 4 上的位函数分别用 $\varphi_{i+1,j}$, $\varphi_{i,j+1}$, $\varphi_{i-1,j}$ 和 $\varphi_{i,j-1}$ 表示, 场域中其他各网格点电位表示依此类推.

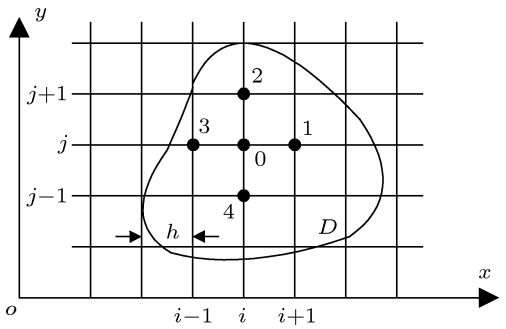

图 1 平面场域网格节点

在场域 D 上, 电位的二维泊松方程为

$$\frac{\partial^2\varphi}{\partial x^2}+\frac{\partial^2\varphi}{\partial y^2}=-F. \tag{1}$$

这里 F 表示电场中的 $\dfrac{\rho}{\varepsilon}$ 或磁场中的 μJ, 边界条件为

$$u(x,y)=g(x,y),\quad (x,y)\in\partial D. \tag{2}$$

其中 ∂D 为场域 D 的边界, $g(x,y)$ 是边界上的电位函数.

若用有限差分逼近来求解泊松方程, 最为常见的是标准的五点公式

$$\varphi_{i,j}=\frac{1}{4}\left(\varphi_{i+1,j}+\varphi_{i,j+1}+\varphi_{i-1,j}+\varphi_{i,j-1}+h^2F\right). \tag{3}$$

Jacobi 法用前一次迭代得到网络格点位函数值作为下一次迭代时的设定值, 即 (i,j) 点在 $(n+1)$ 次迭代时的位函数值 $\varphi_{i,j}^{(n+1)}$ 为

$$\varphi_{i,j}^{(n+1)}=\frac{1}{4}\left(\varphi_{i+1,j}^{(n)}+\varphi_{i,j+1}^{(n)}+\varphi_{i-1,j}^{(n)}+\varphi_{i,j-1}^{(n)}+h^2F\right). \tag{4}$$

Jacobi 法需要两个内存单元, 分别存储两次相邻迭代的近似值, 因而占用的内存大, 收敛速度较慢.

Gauss-Seidel 法将网格节点位函数左下方的新值代替旧值, 使收敛速度得到提高, 其迭代格式为

$$\varphi_{i,j}^{(n+1)}=\frac{1}{4}\left(\varphi_{i+1,j}^{(n)}+\varphi_{i,j+1}^{(n)}+\varphi_{i-1,j}^{(n+1)}+\varphi_{i,j-1}^{(n+1)}+h^2F\right). \tag{5}$$

但是当网格的节点数目很大时, 此法的收敛速度仍然很慢.

SOR 法进行了一些改进, 引入了松弛因子, 降低了计算机存储量. 但缺点是 SOR 法没有明显的并行性, 而且最佳松弛因子选取困难, 其迭代格式为

$$\varphi_{i,j}^{(n+1)}=(1-\omega)\varphi_{i,j}^{(n)}+\frac{\omega}{4}\left(\varphi_{i+1,j}^{(n)}+\varphi_{i,j+1}^{(n)}+\varphi_{i-1,j}^{(n+1)}+\varphi_{i,j-1}^{(n+1)}+h^2F\right), \tag{6}$$

式中, 权数 ω 称为松弛因子或加速收敛因子, $1<\omega<2$. 松弛因子 ω 决定了迭代解收敛的速度, 合适的收敛因子能加快收敛速度, 而不合适的收敛因子将降低收敛速度甚至发散.

PSOR 的思想是将场域 D 先分成 $P_1\times P_2$ 个非重叠子区域 (如图 2 所示, 其中 $P_1=P_2=2$), 用 $P_1\times P_2$ 台处理机对每个子域分别进行超松弛迭代. 具体方法为构造四个 SOR 迭代格式 [14,15]:

$$\varphi_{i,j}^{(n+1)}=(1-\omega)\varphi_{i,j}^{(n)}+\frac{\omega}{4}\left(\varphi_{i+1,j}^{(n+1)}+\varphi_{i,j+1}^{(n+1)}+\varphi_{i-1,j}^{(n)}+\varphi_{i,j-1}^{(n)}+h^2f_{i,j}\right), \tag{7a}$$

$$\varphi_{i,j+1}^{(n+1)}=(1-\omega)\varphi_{i,j+1}^{(n)}+\frac{\omega}{4}\left(\varphi_{i+1,j+1}^{(n+1)}+\varphi_{i,j+2}^{(n)}+\varphi_{i-1,j+1}^{(n)}+\varphi_{i,j}^{(n+1)}+h^2f_{i,j+1}\right), \tag{7b}$$

$$\varphi_{i+1,j}^{(n+1)}=(1-\omega)\varphi_{i+1,j}^{(n)}+\frac{\omega}{4}\left(\varphi_{i+2,j}^{(n)}+\varphi_{i+1,j+1}^{(n+1)}+\varphi_{i,j}^{(n+1)}+\varphi_{i+1,j-1}^{(n)}+h^2f_{i+1,j}\right), \tag{7c}$$

$$\varphi_{i+1,j+1}^{(n+1)}=(1-\omega)\varphi_{i+1,j+1}^{(n)}+\frac{\omega}{4}\left(\varphi_{i+2,j+1}^{(n)}+\varphi_{i+1,j+2}^{(n)}+\varphi_{i,j+1}^{(n+1)}+\varphi_{i+1,j}^{(n+1)}+h^2f_{i+1,j+1}\right). \tag{7d}$$

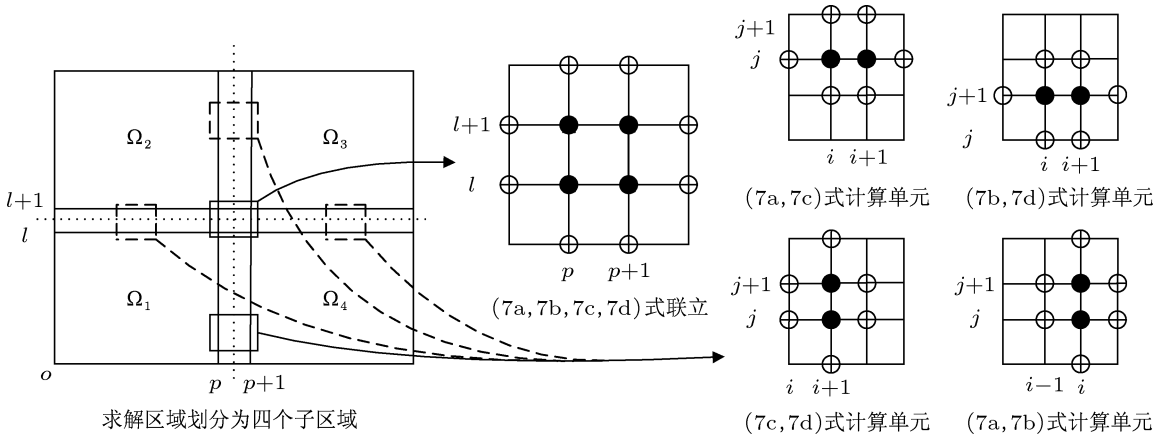

图 2 PSOR 算法格式

图中算法格式分为三步: 第一步, 用 (7a), (7b), (7c), (7d) 式分别迭代 Ω_1, Ω_2, Ω_3, Ω_4 子区域上的网格内点; 第二步, 用 (7a), (7b), (7c), (7d) 式联立迭代内边界相交的四个点, 公式 (7a), (7c); (7b), (7d); (7c), (7d); (7a), (7b) 分别迭代内边界上其他节点. 第三步, 验证收敛性, 若收敛则停止迭代, 否则转向第一步.

由于采用了并行技术, 计算机各处理器间通信与计算时间重叠, 获得了较为理想的加速效率. 因此, PSOR 用于计算机求解泊松问题时, 可以减少迭代次数, 节省计算机程序运行的时间, 缺点是最佳因子选择困难.

最佳收敛因子的取值因问题而异, 目前只有少数模型问题, 才有确定使收敛速度达到最优的松弛因子 (最佳因子) 的理论公式 [16], 在一般的情况下, 最佳收敛因子只能凭借经验取值, 因此如何快速选取最佳因子成为 PSOR 法的关键. 基于此目的, 本文对 PSOR 法进行改进, 采用多目标 GA 对松弛因子进行全局寻优, 形成新的算法 (GPSOR), 解决了最佳因子选择困难的问题.

3 多目标遗传 PSOR 改进算法的设计

为了使遗传算法获得良好的性能, 对各参数进行优化配置是十分必要的. 遗传算法所涉及的要素有参数编码、初始群体的设定、适应度函数的设计、遗传操作的设计和控制参数的设定等.

3.1 适应度函数及编码

适应度函数值的大小用来区分每个个体的优劣, 并判定个体是否进入下一代. 本文采用遗传算法对最佳收敛因子 ω 进行全局寻优, 确定 ω 在区间 (0, 2) 上的最优值. 由以上分析知 ω 在很大程度上影响迭代收敛的速度与精度. 因此适应度函数应包含迭代次数 K 以及收敛精度这两个因素, 考虑双目标适应度函数为

$$\begin{aligned} F^1 &= K, \\ F^2 &= \max\left|\varphi_{ij}^{(K)}(\omega) - \varphi_{ij}^{(K-1)}(\omega)\right|. \end{aligned} \tag{8}$$

则标准适应度为

$$F_{\text{fit}} = c_1 F^1 + c_2 F^2. \tag{9}$$

式中 c_1, c_2 为加权系数, 满足 $c_1 + c_2 = 1$, 且 $c_1 > 0$, $c_2 > 0$. 如果 ω 选取合适, 随着迭代的进行, F^2 将逐渐接近于零, 是一个减函数, 而迭代次数 K 逐渐变大, F^1 是一个增函数. (9) 式将这两个函数进行加权求和, 可以得到一个凸函数, 根据凸函数的性质可知该函数唯一极值点即为最佳的松弛因子. 结合遗传算法适应度最大化原则, 对标准适应度作如下修改:

$$F_{\text{fit}} = \frac{1}{1 + c_1 F^1 + c_2 F^2}. \tag{10}$$

调整后的适应度在 (0, 1) 区间内, 具有更好的鉴别力. 由于 F^1 与 F^2 数量级不一致, 因此进一步对 F^1 与 F^2 进行规范化, 令权系数 c_1, c_2 满足

$$\frac{c_1}{c_2} = \frac{\max F_n^2}{\max K_n}, (n = 1, 2, \cdots, N). \tag{11}$$

式中 N 为种群大小; $\max F_n^2$ 表示当前代的最大误差; $\max K_n$ 表示当前代迭代次数的最大值, 它与 GA 寻优时间成正比. 若 $\max K_n$ 过大, GA 算法用时太长, 将降低 GA 的作用; 若 $\max K_n$ 过小,GA 将提前收敛而得不到最佳因子. 因此, 在保证最佳因子优先的前提下, $\max K_n$ 应尽量取一个较小的值. 规范化有利于均衡 F^1 与 F^2 的比例, 使适应度函数更趋合理, 规范化后的适应度为

$$c_1+c_2=1,\frac{c_1}{c_2}=\frac{\max F_n^2}{\max K_n},(n=1,2,\cdots,N),$$
$$F_{\text{fit}}=\frac{1}{1+c_1K+c_2\max\left|\varphi_{ij}^{(K)}(\omega)-\varphi_{ij}^{(K-1)}(\omega)\right|}. \tag{12}$$

由 (12) 式可知, 在算法早期, F^2 占据主导作用, 算法的主要目的是寻找精度较高的收敛因子; 而在算法晚期, F^1 占据主导作用, 算法的主要目的是寻找收敛速度较快的收敛因子. 因此, 该适应度函数不仅具有凸函数的性质符合适应度函数的选取原则, 而且综合了 "准" 和 "快" 的特点, 有利于 GA 快速进行全局寻优.

对于染色体的编码, 由于二进制编码个体长度大, 在进行数值优化时精度不高, 且稳定性不如实数编码, 因此本文采用实数编码方法对染色体进行编码.

3.2 遗传算子

为了获得良好的新一代群体实现群体进化, 进一步对遗传算子进行设计, 这是遗传策略的主要组成部分, 也是调整和控制进化过程的基本工具.

3.2.1 选择算子

选择操作是遗传算法中环境对个体适应性的评价方式, 也是实现群体优良基因传播的基本方式. 考虑双目标适应度函数为凸函数, 具有唯一极值的特点, 选择算子应该选用截断选择方法. 该方法只允许最优秀的个体才能选为父体, 截断选择的参数为截断阈值 T, $10\%\leqslant T\leqslant 50\%$, 即有 1-$T$ 的个体没有机会被选择, 因此算法收敛速度快. 结合没有重串的稳态繁殖法 (steady state reproduction without duplicates), 在形成新一代种群时, 首先保留上一代中优质的个体, 然后检查新加入的个体与种群中现有个体是否重复, 如果重复就舍弃. 这种做法增大了个体在种群中的分布区域, 提高了种群的多样性, 避免了不必要的适应度计算.

3.2.2 交叉算子

算术交叉算子是实数编码遗传算法中应用最广泛的一种算子, 其采用的交叉方法是线性插值. 比如在两个体 ω_1, ω_2 之间进行算术交叉, 则交叉运算所产生出的两个新个体 ω_1^*, ω_2^* 为

$$\begin{aligned}\omega_1^*&=a\omega_1+(1-a)\omega_2,\\ \omega_2^*&=(1-a)\omega_1+a\omega_2,\end{aligned} \tag{13}$$

其中 a 是 (0, 1) 区间内的随机数. 由方程组可知, 无论 a 选择何值, 交叉后新个体不会超出两个旧个体所界定的范围, 其好处是新个体特征可控, 加快了局部寻优的计算速度.

3.2.3 变异算子

变异本身是一种局部随机搜索, 使遗传算法具有局部的随机搜索能力以及保持种群的多样性, 防止出现过早收敛. 本文基于实值编码采用非均匀实值变异算子

$$\omega'=\begin{cases}\omega+\Delta(t,\omega_{\max}-\omega), & d>0.5,\\ \omega-\Delta(t,\omega-\omega_{\min}), & d\leqslant 0.5,\end{cases} \tag{14}$$

式中 $\omega_{\max}$ 和 $\omega_{\min}$ 分别为 ω 的上界和下界值, t 为当前进化代数, ω 为变异前的值, ω' 为变异后的值, d 为 [0, 1] 区间的随机数, $\Delta(t,y)$ 为

$$\Delta(t,y)=y\cdot r\cdot(1-t/W)^b, \tag{15}$$

这里 r 为 [0,1] 区间的随机数, W 为最大进化代数; b 为确定非均匀度的参数. 由 (14) 式可知, 进化初期, 变异步长较大, 因此有利于扩大编码空间搜索的广度, 提高模式重组能力, 使最佳因子迅速跳到极值点附近; 进化后期, 变异步长较小, 有利于提高个体生存能力, 在极值点附近逐步求精. 因此,(14) 式可以有效提高 GA 的性能.

3.2.4 交叉概率和变异概率

GA 参数中交叉概率 P_{c} 和变异概率 P_{m} 的选择也是影响遗传算法行为和性能的关键所在. Srinvivas 等提出一种自适应遗传算法 (adaptive GA, AGA)[17], P_{c} 和 P_{m} 能够随适应度自动改变, 其好处是在保护最优个体的同时加快了较差个体的淘汰速度, P_{c} 和 P_{m} 计算表达式如下:

$$P_{\text{c}}=\begin{cases}P_{\text{c1}}-\dfrac{(P_{\text{c1}}-P_{\text{c2}})\left(f'-f_{\text{avg}}\right)}{f_{\max}-f_{\text{avg}}}, & f'\geqslant f_{\text{avg}},\\ P_{\text{c1}}, & f'<f_{\text{avg}},\end{cases} \tag{16a}$$

$$P_{\mathrm{m}}=\begin{cases}P_{\mathrm{m1}}-\dfrac{(P_{\mathrm{m1}}-P_{\mathrm{m2}})(f_{\max}-f)}{f_{\max}-f_{\mathrm{avg}}}, & f\geqslant f_{\mathrm{avg}},\\ P_{\mathrm{m1}}, & f<f_{\mathrm{avg}},\end{cases} \tag{16b}$$

式中 $f_{\max}$ 为每代群体最大适应度值; f_{avg} 为每代群体平均适应度值; f' 为两个交叉个体较大的适应度值; f 为变异个体的适应度值; $P_{\mathrm{c1}}=0.9$, $P_{\mathrm{c2}}=0.6$, $P_{\mathrm{m1}}=0.1$, $P_{\mathrm{m2}}=0.001$.

4 数值和仿真分析

为了验证改进算法的有效性, 考虑如下二维泊松方程:

$$\frac{\partial^2\varphi}{\partial x^2}+\frac{\partial^2\varphi}{\partial y^2}=4x\cos(x)+\left(5-x^2-y^2\right)\sin(x),$$
$$\Omega: x^2+y^2-1=0,\quad (x,y\in[-1,1]). \tag{17}$$

边界 $\partial\Omega$ 满足 Dirichlet 边界条件, $u=0$ V, 方程的精确解为

$$\varphi(x,y)=\left(x^2+y^2-1\right)\sin(x). \tag{18}$$

用数值方法解泊松方程时, Jacobi, Gauss-Seidel 法没有最佳收敛因子问题, 在相同收敛精度条件下重复计算, 其迭代次数基本不变; 对于 SOR, PSOR 法, 在迭代次数为 100, 网格划分为 100×100 条件下, ω 选取 1.72 时 [18], 算法精度分别是 1.284×10^{-3} 和 9.130×10^{-4}; ω 选取 1.80 时, 算法精度分别是 8.732×10^{-4} 和 7.557×10^{-4}, 后者的收敛效果比前者更好, 表明最佳因子不能依靠经验取得. 为了快速获得最佳因子, 设定种群大小 $N=20$, 染色体长度为 15 位 (ω 保留 15 位有效数字), 进化代数 $W=10$, 变异算子 $b=2$, 截断阈值 $T=10\%$, 网格划分 $h^2=100\times100$. 图 3 所示为遗传算法进化过程.

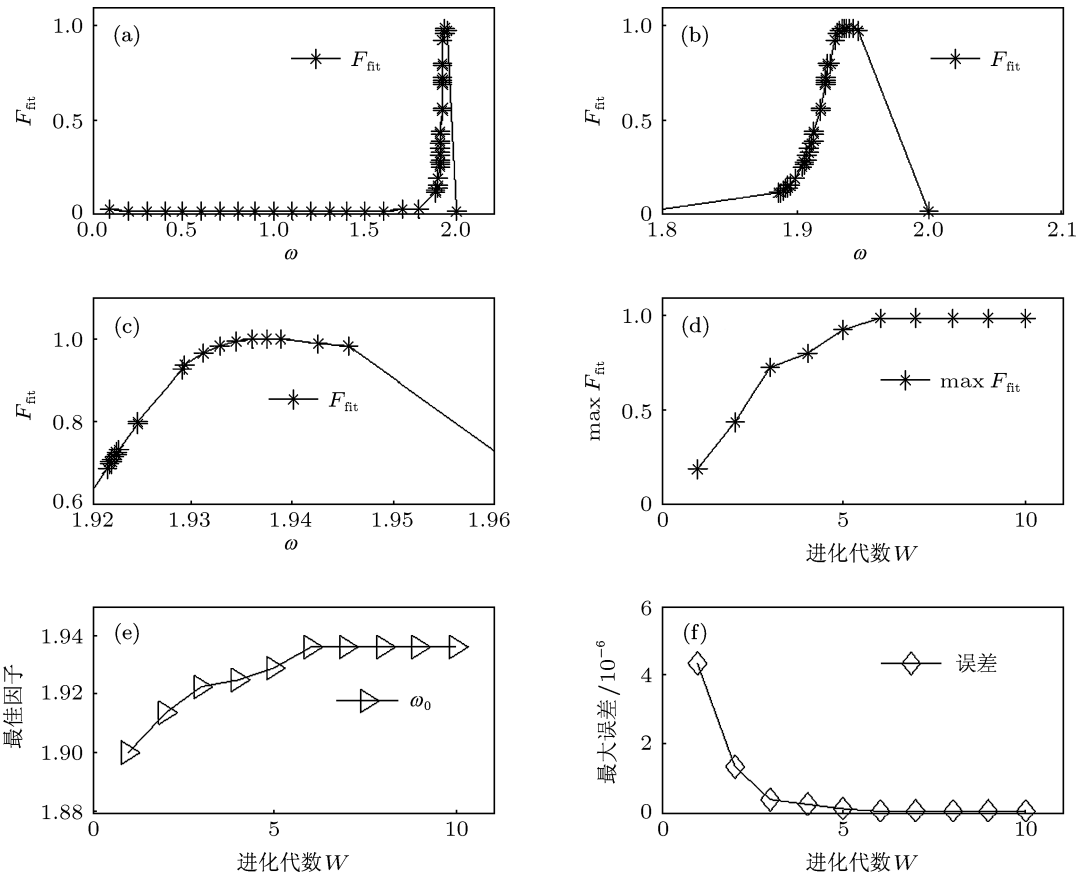

图 3 历代种群进化曲线 (a)—(c) 种群适应度曲线; (d) 每代最大适应度变化曲线; (e) 最佳因子变化曲线; (f) 每代最大误差变化曲线

图 3(a), (b) 和 (c) 表明适应度函数是一个凸函数, 具有唯一极值, 且极值对应的松弛因子在 1.94 附近, 这个特点与理论分析是一致的. 由图 3(d) 知, 随着种群的进化, 种群的适应度在不断增加, 但到进化后期, 其值基本不变, 维持在 0.99—1 范围内, 表明进化是一个择优的过程, 此外适

应度最大且稳定不变是 GA 算法收敛的表现. 图 3(e), (f) 分别是进化过程中每代的最佳因子与最大迭代误差的变化曲线, 可见随着种群的进化, 误差逐渐减小, 充分体现了 “适者生存” 的遗传思想, 也符合进化的原理. 经过 10 代进化, GA 寻找最佳因子的过程用时 $T_{\mathrm{GA}} = 28.576$ s, 找到的最佳因子 ω_0 为 1.93533759483030, 对应的适应度为 0.998138358465302, 而第 9 代的最大适应度为 0.998138358465302, 第 10 代的最大适应度为 0.998138358465302, 两者适应度相差已经不超过 10^{-10}, 因此可以认为 ω_0 就是全局最佳因子, 进化结果表明 GA 算法取得了预期效果.

为了考察 GA 的性能, 还必须分析 GA 算法的计算量与计算时间等问题. GA 算法寻优用时 $T_{\mathrm{GA}} = 28.576$ s, 其计算量 C_{GA} 用下式表示:

$$C_{\mathrm{GA}} = \frac{h_1^2}{h_2^2} \times N \times W \times \max K_n, \quad (n = 1, 2, \cdots, N), \tag{19}$$

式中 h_1 为 GA 寻优过程中网格的划分步长, GA 在 100×100 的网格下寻找最佳因子, 则 $h_1 = 100$; h_2 为实际求解方程时 Ω 的划分, 如网格划分为 200×200, 则 $h_2 = 200$; N 是种群大小, W 是进化代数. 需要指出的是, C_{GA} 是一个相对值, 因为在分析传统算法的计算量时用到许多不同的网格, 迭代次数相同但实际上计算量不同, 如 Jacobi 在 200×200 的网格下迭代 1000 次, 迭代量记为 1000, 而 GA 算法在 100×100 网格下迭代 1000 次, 迭代量则不能记为 1000, 因为必须考虑网格划分的比例, 所以采用 (19) 式来描述 GA 的计算量, 本文中其值为 $C_{\mathrm{GA}} = 15000$.

由图 3 知, 最佳因子大致在 1.94 附近跳动, 根据定理, SOR 迭代收敛的必要条件是 $0 < \omega < 2$, 因此为保证算法的收敛, 最佳因子不能超出 (0, 2) 区间. 图 4 所示是 ω 分别取最优解, 1.990, 1.995 和 2.0 时方程的逼近效果, 可以看出最优解与非最优解的性能有很大差别.

值得一提的是, 这些非最优解的逼近效果与地貌特征类似, 表明可以用某些非最优解来对复杂地理环境进行建模, 至于这些非最优解的分布与特性有待进一步研究, 本文不做讨论. 因此, 本文提出的遗传 PSOR (GPSOR) 算法一方面得到了全局最优解, 用于快速、精确地求解二维泊松方程; 另一方面, 得到了一些位于最优解附近的非最优解, 如 $\omega = 1.990, 1.995, 2.0$ 等, 应用到复杂地貌的建模则不失为一种方法, 这也是采用 GA 求解泊松方程时所具有的新特征.

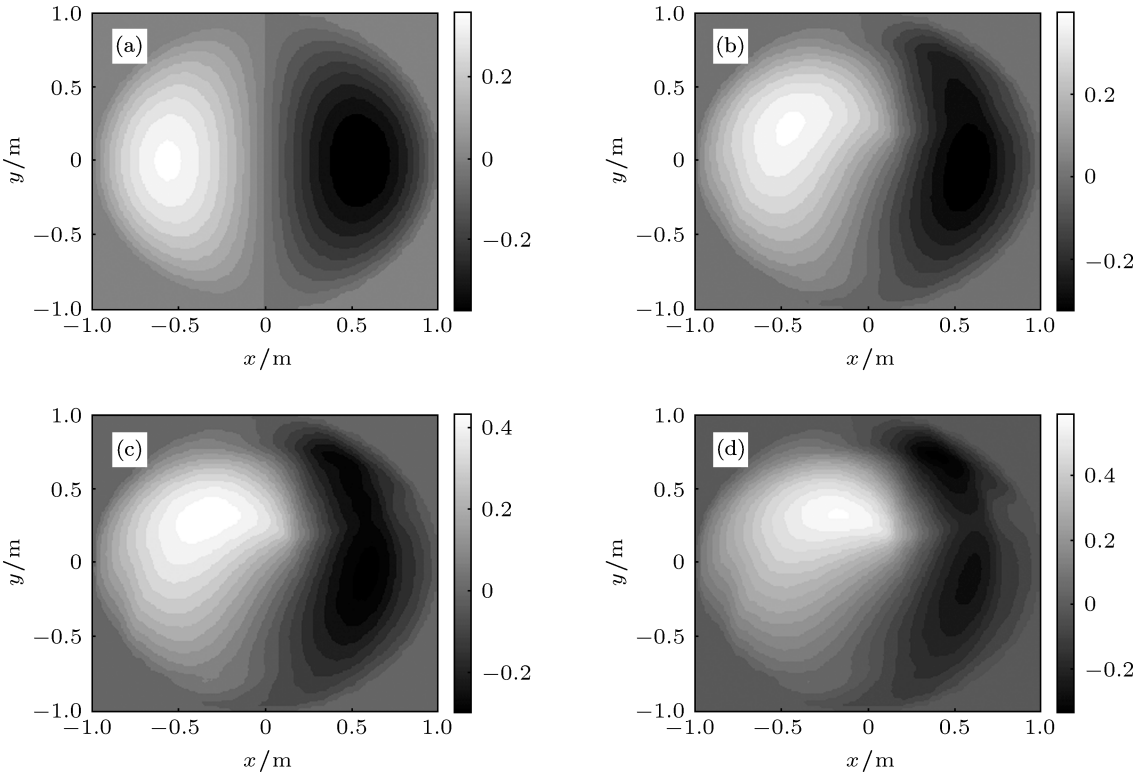

图 4 不同松弛因子逼近效果 (a) 最优解; (b) $\omega = 1.990$; (c) $\omega = 1.995$; (d) $\omega = 2.0$

表 1 相同误差精度下算法迭代量对比

算法	迭代量 C/次						
	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}
Jacobi	13639	19955	26270	32585	38901	45216	51531
Gauss-Seidel	7890	11526	17152	25101	33118	41135	49152
SOR	6384	9407	14523	20665	26812	32959	39106
PSOR	5357	8849	12933	17678	24757	28214	34392
GPSOR	438	536	633	729	826	923	4788

注: $C_{\mathrm{GA}} = 15000$.

表 2 相同误差精度下算法时间对比

算法	算法时间 T/s						
	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}
Jacobi	50.208	72.526	94.290	118.080	139.515	161.683	184.051
Gauss-Seidel	30.571	43.307	64.835	92.065	121.918	152.124	173.600
SOR	26.289	37.764	57.130	80.407	103.881	128.215	151.432
PSOR	6.814	9.905	17.127	30.913	42.632	58.907	75.278
GPSOR	1.823	2.189	2.553	2.928	3.2920	3.649	4.022

注: $T_{\mathrm{GA}} = 28.576$.

为考察各算法在不同精度以及不同网格划分下的性能, 用 Jacobi, Gauss-Seidel, SOR, PSOR 以及 GPSOR 分别求解 (17) 式, 对计算量、误差以及计算时间进行对比, 见表 2 至表 4 (其中表 2 至表 4 网格划分为 200×200, 表 2 至表 4 中 SOR 及 PSOR 参数 ω 取 1.3, GPSOR 中 ω 采用本文 GA 计算得到的值, 即 ω_0).

由表 1 及表 2 可知, 在误差精度为 10^{-6} 情况下, $C_{\mathrm{jacobi}} = 13639$, $T_{\mathrm{jacobi}} = 50.208$; $C_{\mathrm{PSOR}} = 5357$, $T_{\mathrm{PSOR}} = 6.814$; $C_{\mathrm{GPSOR}} = 438$, $T_{\mathrm{GPSOR}} = 1.823$, 表面上看 GOSOR 用时及计算量最少, 但 $T_{\mathrm{GPSOR}} = 1.823$ 是事先进行 GA 寻优条件下得到的值, 而 $T_{\mathrm{GA}} = 28.576$, 所以从总的计算量和计算时间来看 GPSOR 方法并没有优势. 随着精度要求的提高, 传统方法计算量迅速增加, 而 GPSOR 计算量并没有显著变化, 误差为 10^{-9} 后 GPSOR 开始具有优势, 精度越高, 优势越明显, 表明对于精度要求较高的方程, GPSOR 比传统方法更好.

表 3 表明, 随着迭代的进行, 传统方法误差变化不明显, 收敛性能相对平稳缓和; GPSOR 前期误差较大, 随着迭代的进行, 误差迅速变小, 到 250 次迭代之后就已经优于其他算法, 这表明 GPSOR 方法具有加速收敛的特性, 能加快方程的求解, 这与传统方法平缓的收敛性质明显不同.

表 4 表明, 当网格划分较小且精度要求不高时, 传统方法计算更简便, 但当网格划分超过 128×128 时, GPSOR 法表现更好, 而且网格划分越细, GPSOR 优势越大, 节省的时间也越多, 如: 512×512 条件下

$$T_{\mathrm{GPSOR}} = \frac{1}{17}T_{\mathrm{jacobi}} = \frac{1}{10}T_{\mathrm{G\text{-}S}} = \frac{1}{9}T_{\mathrm{SOR}} = \frac{1}{6}T_{\mathrm{PSOR}}. \tag{20}$$

表 4 还表明最佳因子在不同的网格划分下都获得了比较满意的收敛效果, 因此在实际应用中, 可以考虑在粗网格下进行 GA 寻优, 从而减少计算量, 获得更快的寻优速度.

表 3 相同迭代次数下误差对比 (单位: 10^{-4})

算法	100 次	150 次	200 次	250 次	300 次	350 次	400 次
Jacobi	1.3255	1.2966	1.2693	1.2434	1.2189	1.1951	1.1725
Gauss-Seidel	2.5515	2.4497	2.3563	2.2687	2.1864	2.1088	2.0347
SOR	3.2514	3.0886	2.9400	2.8023	2.6739	2.5534	2.9856
PSOR	4.3948	3.8712	3.6031	3.3318	3.2769	3.0971	3.5571
GPSOR	22.7347	10.5582	4.6422	1.0408	0.32251	0.0851	0.0558

表 4 不同网格划分迭代量与运行时间对比 (误差为 10^{-6})

算法	迭代量 C/次				计算时间 T/s			
	128×128	256×256	512×512	1024×1024	128×128	256×256	512×512	1024×1024
Jacobi	6612	20087	55337	121493	9.986	125.927	1489.499	13171.286
Gauss-Seidel	3847	11703	33932	115535	6.168	73.907	917.443	11085.504
SOR	3179	9618	28258	39201	5.088	63.482	791.957	984.830
PSOR	1017	6617	15652	30673	2.849	37.079	527.971	759.916
GPSOR	339 (36221)	724 (9155)	2242 (2288)	5630 (572)	0.699	4.818	61.454	165.933

注: 括号内为 C_{GA} 值, $T_{\mathrm{GA}} = 28.576$.

图 5(a) 是各算法迭代次数的变化曲线, 在相同的误差精度下, GPSOR 具有更少的迭代次数, 并且随着误差等级的提高, 传统方法的计算量与 GPSOR 计算量之间的差别在逐渐扩大, GPSOR 的优势也更加明显; 图 5(b) 是各算法运行时间的对比曲线, 精度要求较低时, GPSOR 并不实用, 这是因为 GA 寻优需要占用时间, 而 Jacobi 和 Gauss-Seidel 并不需要, 所以在这种情况下 GPSOR 用时会比传统方法长. 随着精度要求的提高, GPSOR 算法开始具有优势, 而且精度越高, 优势越显著, 正是这种特点使得 GPSOR 特别适合于精度要求高的场合, 这也是传统算法所不及的; 图 5(c) 是迭代前期的误差曲线, 可见 GPSOR 曲线最陡, 下降速度最快, 在 250 次迭代之后就超过其他曲线, 表明 GPSOR 算法具有加速收敛的性质, 这也是区别于传统算法的重要特征; 图 5(d) 是迭代次数 $k = 300$ 时, 算法最优解与精确解的绝对误差图像, 可见 300 次迭代之后 GPSOR 算法误差已不超过 10^{-2}, 获得了良好的求解效果.

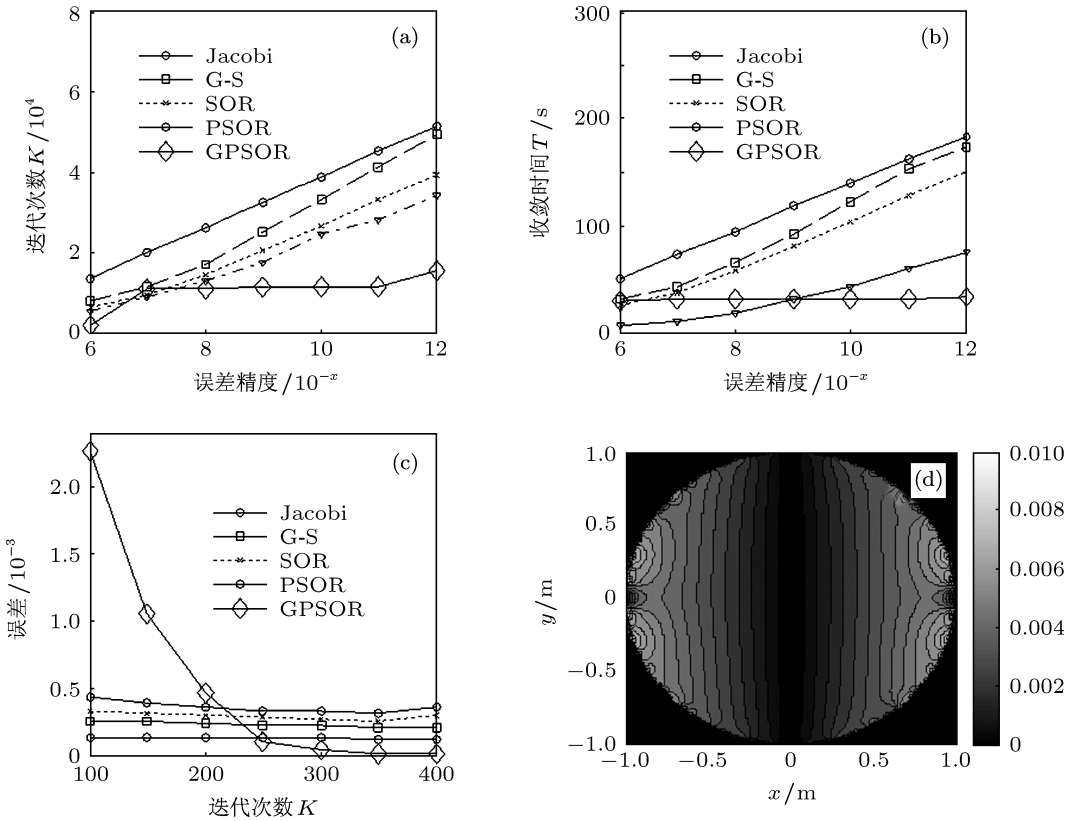

图 5 数据对比及仿真结果 (a) 迭代次数对比图; (b) 收敛时间对比图; (c) 误差对比图; (d) 最优解的绝对误差图

5 结 论

提出了一种 GPSOR 改进算法, 采用 GA 对 PSOR 松弛因子进行全局寻优, 解决了逐次松弛迭代法求解泊松方程最佳因子的快速选取问题. 讨论了 GPSOR 算法与传统算法的区别, 进行了实例验证. 算法实例表明了 GPSOR 方法具有加速收敛特性, 能加快方程的求解速度, 节省计算时间, 在相同的迭代次数下收敛精度更高, 与传统算法平缓的收敛性质有着较大的区别. 算法适合于求解计算量大、精度要求高的偏微分方程, 并且精度要求越高, 节省时间越多, 算法也越有效, 为电磁场波动、流体、热传导等泊松方程的快速以及高精度求解提供了一种新方法; 实验中还发现非最佳因子具有一定利用价值, 可以应用到解泊松方程以外的领域, 是今后工作中需要探索和研究的.

[1] Wang X Y, Zhang H M, Wang G Y, Song J J, Qin S S, Qu J T 2011 *Acta Phys. Sin.* **60** 027102 (in Chinese) [王晓艳, 张鹤鸣, 王冠宇, 宋建军, 秦珊珊, 屈江涛 2011 物理学报 **60** 027102]

[2] Shang Y, Huo B Z, Meng C N, Yuan J H 2010 *Acta Phys. Sin.* **59** 8178 (in Chinese) [尚英, 霍丙忠, 孟春宁, 袁景和 2010 物理学报 **59** 8178]

[3] Ji F Y, Zhang S L 2012 *Acta Phys. Sin.* **61** 080202 (in Chinese) [吉飞宇, 张顺利 2012 物理学报 **61** 080202]

[4] Ma J W, Yang H Z, Zhu Y P 2001 *Acta Phys. Sin.* **50** 1415 (in Chinese) [马坚伟, 杨慧珠, 朱亚平 2001 物理学报 **50** 1415]

[5] Liu S K, Fu Z T, Liu S D 2001 *Phys. Lett.* A **289** 69

[6] Kohno T, Kotakemori H, Nikia H 1997 *Linear Algebra Appl.* **267** 113

[7] Hadjidimos A 2000 *Journal of Computational and Applied Mathematics* **123** 77

[8] Smith B F, Bjorstad P E, Gropp W D 1996 *Domain Decomposition*: *Parallel Multilevel Methods for Elliptic Partial Differential Equations* (Cambridge: Cambridge University Press) p124

[9] Wang X B, Liang Z C, Wu Z S, 2012 *Acta Phys. Sin.* **61** 124104 (in Chinese) [王晓冰, 梁子长, 吴振森 2012 物理学报 **61** 124104]

[10] He J, Xu J Y, Yao X 2000 *IEEE Trans on Evolutionary Computation* **4** 295

[11] Dai D, Ma X K, Li F C, You Y 2002 *Acta Phys. Sin.* **51** 2459 (in Chinese) [戴栋, 马西奎, 李富才, 尤勇 2002 物理学报 **51** 2459]

[12] Zhao Z J, Zhen S L, Shang J N, Kong X Z 2007 *Acta Phys. Sin.* **56** 6760 (in Chinese) [赵知劲, 郑仕链, 尚俊娜, 孔宪正 2007 物理学报 **56** 6760]

[13] Dutta D, Dutta P, Sil J 2012 *Proceedings of the 1st International Conference on Recent Advances in Information Technology*, Dhanbad, India, March 15–17 2012 p548

[14] Sweilam N H, Moharram H M, Ahmed S 2012 *Proceedings of the 8th International Conference on Informatics and Systems*, Cairo, Egypt, May 14–16, 2012 p78

[15] Xu Q Y 2011 *Proceedings of the 2011 Inernational Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, Beijing, China, Octorber 10–12, 2011 p295

[16] Wang B Z 2002 *Computational electromagnetic* (Beijing: Science Press) p34 (in Chinese) [王秉中 2002 计算电磁学 (北京: 科学出版社) 第 34 页]

[17] Srinivas M, Patnaik L M 1994 *IEEE Trans. on SMC* **24** 656

[18] Xie Z C, Zhou Y Q 2009 *Mathematics in Practice and Theory* **39** 154 (in Chinese) [谢竹诚, 周永权 2009 数学的实践与认知 **39** 154]

An ameliorative algorithm of two-dimensional Poisson equation based on genetic parallel successive over-relaxation method*

Peng Wu[1)†] He Yi-Gang[1)2)] Fang Ge-Feng[3)] Fan Xiao-Teng[3)]

1) (*College of Electrical and Information Engineering, Hunan University, Changsha* 410082*, China*)

2) (*School of Electrical and Automation Engineering, Hefei University of Technology, Hefei* 230009*, China*)

3) (*The 41st Research Institute of China Electronics Technology Group Corporation, Qingdao* 266555*, China*)

(Received 27 May 2012; revised manuscript received 31 August 2012)

Abstract

There exist some disadvantages in the calculation of two-dimensional Poisson equation with several common methods. A new ameliorative algorithm is presented. It is based on a parallel successive over-relaxation (PSOR) method, by using the multi-objective genetic algorithm to search for optimal relaxation factor, with which the problem of optimal relaxation factor selection in PSOR is solved. The multi-objective fitness function is constructed, with which the genetic algorithm parameters are optimized. The analysis mainly focuses on algorithm computation, time cost and accuracy of error correction. The performance of the ameliorative algorithm is compared with those of Jacobi, Gauss-Seidel, Successive over relaxation iteration (SOR) and PSOR. Experimental results show that relaxation factor has a significant effect on the speed of solving Poisson equation, as well as the accuracy. The improved algorithm can increase the speed of iteration and obtain higher accuracy than traditional algorithm. It is suited for solving complicated finite difference time domain equations which need high accuracy. The higher the accuracy requirement, the better the performance of the algorithm is and the more computation time can also be saved.

Keywords: poisson equation, genetic algorithm, parallel successive over-relaxation method, finite difference method

PACS: 03.50.De, 41.20.Cv, 02.70.Bf **DOI:** 10.7498/aps.62.020301

* Project supported by the National Natural Fund for Distinguished Yong Scholar of China (Grant No. 50925727), the National Natural Science Foundation of China (Grant Nos. 60876022, 61102039, 51107034), the Hunan Province Science and Technology Program, China (Grant Nos. 2011J4, 2011JK2023), the National Defense Advanced Research Program (Grant No. C1120110004), the Joint Program of Guangdong Province and Ministry of Education of China on Industry, University and Research Institute Integration (Grant No. 2009B090300196), and the Central University Fund Basic Research and Operating Expenses Program.

† Corresponding author. E-mail: pengwu@hnu.edu.cn