计算机科学
COMPUTER SCIENCE

密度泛函微扰理论中响应密度矩阵的迭代求解算法研究

刘人儁，徐直前，商红慧，张云泉

引用本文

刘人儁，徐直前，商红慧，张云泉.密度泛函微扰理论中响应密度矩阵的迭代求解算法研究[J]．计算机科学，2023，50(6)：81-85.

LIU Renyu，XU Zhiqian，SHANG Honghui，ZHANG Yunquan. Study of Iterative Solution Algorithm of Response Density Matrix in Density Functional Perturbation Theory [J]. Computer Science，2023，50(6)：81-85.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.220500252

密度泛函微扰理论中响应密度矩阵的迭代求解算法研究

刘人儒[1,2]　徐直前[2,3]　商红慧[2]　张云泉[2]
1 大连海洋大学信息工程学院　辽宁 大连 116023
2 中国科学院计算技术研究所计算机体系结构国家重点实验室　北京 100190
3 中国科学院大学计算机与控制学院　北京 100190
(liurenyudlou@163.com)

摘　要　针对密度泛函微扰理论中响应密度矩阵的计算问题,提出了一种全新的 Sternheimer 方程的并行求解方法,即通过共轭梯度算法和矩阵直接分解算法对 Sternheimer 方程进行求解,并且在第一性原理的分子模拟软件 FHI-aims 中实现了这两种算法。实验结果表明采用共轭梯度算法和矩阵直接分解算法的计算结果精度较高,相比传统方法的计算结果误差较小,且具有可扩展性,验证了新的 Sternheimer 方程中线性方程求解的正确性和有效性。

关键词:密度泛函;线性方程;迭代算法
中图法分类号　TP311.5

Study of Iterative Solution Algorithm of Response Density Matrix in Density Functional Perturbation Theory

LIU Renyu[1,2],XU Zhiqian[2,3],SHANG Honghui[2] and ZHANG Yunquan[2]
1 College of Information Engineering,Dalian Ocean University,Dalian,Liaoning 116023,China
2 State Key Laboratory of Computer Archintecture,Institute of Computing Technology,Chinese Academy of Sciences,Beijing 100190,China
3 School of Computer and Control Engineering,University of Chinese Academy of Sciences,Beijing 100190,China

Abstract　For the problem of calculating the response density matrix in density-functional perturbation theory(DFPT),a new parallel solution method for the Sternheimer equation is proposed,i.e.,the Sternheimer equation is solved by the conjugate gradient algorithm and the matrix direct decomposition algorithm,and the two algorithms are implemented in the first-principles molecular simulation software FHI-aims. Experimental results show that the computational results using conjugate gradient algorithm and matrix direct decomposition algorithm are more accurate,with less error than those of traditional methods,and scalable,which verifies the correctness and validity of the solution of linear equations in the new Sternheimer equation.

Keywords　Density-functional theory,Linear equations,Iterative algorithm

1　引言

密度泛函理论(Density-Functional Theory,DFT)是一种通过电子密度来研究多电子体系电子结构的量子力学方法,在化学、物理以及材料科学中都有着非常广泛的应用[1-4]。它可以计算总能量及其一阶导数的信息,而高阶物理性质(如极化率、振动频率)则需要通过密度泛函微扰理论(Density-Functional Perturbation Theory,DFPT)或耦合扰动自洽场来计算(Coupled Perturbed Self-consistent Field,CPSCF)[5-7]。目前已有多种 DFPT 方法,根据不同的原子基组使用情况可将其划分为两类:平面波方法(Planewave,PW)以及原子轨道线性组合方法(Linear Combination of Atomic Orbitals,LACO)[8-10]。PW 方法的基组是完备而弥散的,相反,LACO 方法的局域性较好,可以将单粒子算子表示为稀疏矩阵,并且已经被证明能够成功描述涉及几千个原子的复杂系统。

在 LCAO 方法中,传统计算 DFPT 的方法需要对未占位电子态求和[11]。然而在计算原子极化率时,施特恩海默方法(Sternheimer)则可以避免对未占位态的求和[12]。Sternheimer 方法由 Sternheimer 首次提出并使用,该方法是线性扰动理论的结果。Sternheimer 方法可以用来计算不同的属性,如密度泛函微扰理论中实数和复数的电介质等[13]。这一应用在原子物理学中首次被提出,在 PW 方法中也同样有效。Sternheimer 方法已被推广到线性响应的含时密度泛函理论(Time-Dependent Density Functional Theory,TDDFT)中[14],

到稿日期:2022-05-27　返修日期:2022-09-22
基金项目:国家重点研发计划资助:"面向复杂装备的 CAE 云服务平台研发"项目(2020YFB1709500)
This work was supported by the Research and Development of CAE Cloud Service Platform for Complex Equipment funded by National Key Research and Development Program of China(2020YFB1709500).
通信作者:商红慧(shanghonghui@ict.ac.cn)

但这种线性响应的 TDDFT 方法只能给出与电场扰动有关的极化率。通过求解 Sternheimer 线性方程，尤其是在 LCAO 全电子全势框架下，仍然无法得到关于原子位移和电场扰动的 DFPT 计算公式。这里我们使用一个在 LCAO 公式中推导的 Sternheimer 方法，采用迭代法和直接法对重新推导得出的响应密度矩阵进行求解，并在第一性原理分子模拟程序 FHI-aims 中实现，同时将使用共轭梯度算法和矩阵直接分解算法的计算结果和传统的计算结果进行比较，验证了该 Sternheimer 方法的有效性和可靠性。

2 计算方法和数值模拟

2.1 基于 Sternheimer 的响应密度矩阵

在密度泛函理论中，可利用哈密顿量对应的能量泛函求解基态电子，从而求解基态性质。为求解能量泛函，通常采用单粒子轨道即 Khon-Sham 方程的结果。Khon-Sham 单粒子方程的矩阵形式可写为：

$$\sum_v \boldsymbol{H}_{\mu v}^{(0)} \boldsymbol{C}_{vi}^{(0)} = \varepsilon_i^{(0)} \sum_v \boldsymbol{S}_{\mu v}^{(0)} \boldsymbol{C}_{vi}^{(0)} \tag{1}$$

若对系统施加外部扰动，则式(1)的一阶响应可表示为：

$$\sum_v (\boldsymbol{H}_{\mu v}^{(0)} - \varepsilon_i^{(0)} \boldsymbol{S}_{\mu v}^{(0)}) \boldsymbol{C}_{vi}^{(1)} = \varepsilon_i^{(0)} \sum_v \boldsymbol{S}_{\mu v}^{(1)} \boldsymbol{C}_{vi}^{(0)} - \sum_v (\boldsymbol{H}_{\mu v}^{(1)} - \varepsilon_i^{(1)} \boldsymbol{S}_{\mu v}^{(0)}) \boldsymbol{C}_{vi}^{(0)} \tag{2}$$

在矩阵形式中，可以写成：

$$\boldsymbol{H}^{(0)} \boldsymbol{C}^{(1)} - \boldsymbol{S}^{(0)} \boldsymbol{C}^{(1)} \boldsymbol{E}^{(0)} = \boldsymbol{S}^{(1)} \boldsymbol{C}^{(0)} \boldsymbol{E}^{(0)} - \boldsymbol{H}^{(1)} \boldsymbol{C}^{(0)} + \boldsymbol{S}^{(0)} \boldsymbol{C}^{(0)} \boldsymbol{E}^{(1)} \tag{3}$$

其中，$\boldsymbol{E}^{(0)}$ 是包含特征值 ε_i 的对角矩阵，$\boldsymbol{E}^{(1)}$ 是包含特征值 ε_i 的响应对角矩阵[11]。

$$\boldsymbol{E}^{(1)} = \boldsymbol{C}^{(0)\dagger} \boldsymbol{H}^{(1)} \boldsymbol{C}^{(0)} - \boldsymbol{C}^{(0)\dagger} \boldsymbol{S}^{(1)} \boldsymbol{C}^{(0)} \boldsymbol{E}^{(0)} \tag{4}$$

在 Sternheimer 方法中，将式(3)的特征值的响应加入式(4)中，可得：

$$\boldsymbol{H}^{(0)} \boldsymbol{C}^{(1)} - \boldsymbol{S}^{(0)} \boldsymbol{C}^{(1)} \boldsymbol{E}^{(0)} = -(1 - \boldsymbol{S}^{(0)} \boldsymbol{P}^{(0)}) (\boldsymbol{H}^{(1)} \boldsymbol{C}^{(0)} - \boldsymbol{S}^{(1)} \boldsymbol{C}^{(0)} \boldsymbol{E}^{(0)}) \tag{5}$$

式(5)则是一个线性方程($\boldsymbol{A}x = b$)，因其左侧行列式消失，所以有无穷多个解。

根据其基组的大小，这样的线性方程可以用共轭梯度算法(Conjugate Gradient，CG)以及矩阵直接分解法进行求解。

2.2 计算方法

对于线性方程 $\boldsymbol{A}x = b$，主要有两种求解方法，即迭代法和直接法。迭代法包括共轭梯度算法即 CG 算法[15]，直接法通常指矩阵直接分解法，即 LU 分解法。迭代法是通过某种极限过程去不断逼近真解，其中共轭梯度算法的基本思想是利用上一次的搜索方向和目标函数在当前迭代位置的负梯度向量进行线性组合，再构造一个新的方向，与之前迭代产生的搜索方向共轭，直至求出最小点。共轭梯度算法所需的存储量小，计算方便，并且不需要外来参数，是求解线性方程最有效的方法之一。相比之下，LU 分解算法较为简单，它通过将矩阵分解为上三角和下三角矩阵乘积的形式来计算。LU 分解算法常被用来解线性方程或计算行列式。当矩阵 $\boldsymbol{A}$ 为正定时通常采用 CG 算法[16]。我们在 FHI-aims 程序中实现了迭代算法中的 CG 算法以及 LU 分解算法。

2.2.1 共轭梯度算法

在共轭梯度算法中，梯度表示为：

$$g_k = \boldsymbol{A} x_k - b = \nabla \varphi(x_k) \tag{6}$$

式(6)可以写成：

$$g_k = \boldsymbol{A} x_k - b = \boldsymbol{A}(x_0 + Q_k y_k) - b = -r_0 + (Q_k T_k + r_k \mathrm{e}_k^{\mathrm{T}}) y_k \tag{7}$$

因为 $Q_k T_k y_k = \beta_0 Q_k \mathrm{e}_1 = r_0$，所以有

$$g_k = (\mathrm{e}_k^{\mathrm{T}} y_k) r_k \tag{8}$$

因为 r_i 是 q_{i+1} 的倍数，g_i 是相互正交的，为证明 $g_k = \nu_k r_k$，则必须验证 $\mathrm{e}_k^{\mathrm{T}} y_k = \nu_k$，

$$T_k y_k = (L_k D_k)(L_k^{\mathrm{T}} y_k) = \beta_0 \mathrm{e}_1 \tag{9}$$

从式(9)中可得 $L_k^{\mathrm{T}} y_k = v_k$ 且 $(L_k D_k) v_k = \beta_0 \mathrm{e}_1$，搜索方向 $c_1, \cdots, c_k$ 满足一种不同的正交性质，因此有：

$$c_i^{\mathrm{T}} A c_j = \begin{cases} 0, & \text{if } i \neq j \\ d_j, & \text{if } i = j \end{cases} \tag{10}$$

所有 i 和 j 都满足 $1 \leqslant i < j \leqslant k$。

因为 $Q_k = C_k L_k^{\mathrm{T}}$ 且 $T_k = Q_k^{\mathrm{T}} \boldsymbol{A} Q_k$，因此可得：

$$T_k = L_k (C_k^{\mathrm{T}} \boldsymbol{A} C_k) L_k^{\mathrm{T}} \tag{11}$$

由于 $T_k = L_k D_k L_k^{\mathrm{T}}$ 是从 LDL^{T} 中分解得来，因此有

$$D_k = C_k^{\mathrm{T}} \boldsymbol{A} C_k \tag{12}$$

列分区 $C_k = [c_1 | \cdots | c_k]$ 证明了 $c_i^{\mathrm{T}} \boldsymbol{A} c_j = [D_k]_{ij}$，因此搜索方向 $c_1, \cdots, c_k$ 是共轭的。如果把 c_k 看作一个搜索方向，ρ_k 作为搜索的步长，且这些量都是可以缩放的，则搜索方向的更新方法为：

$$c_k = q_k - \ell_{k-1} c_{k-1} \tag{13}$$

因为 q_k 是 g_{k-1} 的倍数，所以有(搜索方向 k) = g_{k-1} + 标量 × (搜索方向 $k-1$)，可以表示为：

$$p_k = g_{k-1} + \tau_{k-1} p_{k-1} \tag{14}$$

该式符合

$$\boldsymbol{A} p_k = \boldsymbol{A} g_{k-1} + \tau_{k-1} \boldsymbol{A} p_{k-1} \tag{15}$$

又因为有

$$\tau_{k-1} = -\frac{p_{k-1} \boldsymbol{A} g_{k-1}}{p_{k-1}^{\mathrm{T}} \boldsymbol{A} p_{k-1}} \tag{16}$$

且

$$p_k^{\mathrm{T}} \boldsymbol{A} g_{k-1} = p_k^{\mathrm{T}} \boldsymbol{A} p_k \tag{17}$$

因为 ρ_k 是 c_k 的倍数，所以有

$$x_k = x_{k-1} - \mu_k p_k \tag{18}$$

在方程两边加 $\boldsymbol{A}$ 再减去 $\boldsymbol{B}$ 可得

$$g_k = g_{k-1} - \mu_k \boldsymbol{A} p_k \tag{19}$$

通过式(17)可得

$$\mu_k = \frac{g_{k-1}^{\mathrm{T}} g_{k-1}}{g_{k-1}^{\mathrm{T}} \boldsymbol{A} p_k} = \frac{g_{k-1}^{\mathrm{T}} g_{k-1}}{p_k^{\mathrm{T}} \boldsymbol{A} p_k} \tag{20}$$

从方程 $g_{k-1} = g_{k-2} - \mu_{k-1} \boldsymbol{A} p_{k-1}$ 和 $g_{k-1}^{\mathrm{T}} g_{k-2} = 0$ 中可以得到

$$g_{k-1}^{\mathrm{T}} g_{k-1} = -\mu_{k-1} g_{k-1}^{\mathrm{T}} \boldsymbol{A} p_{k-1} \tag{21}$$

$$g_{k-2}^{\mathrm{T}} g_{k-2} = \mu_{k-1} g_{k-2}^{\mathrm{T}} \boldsymbol{A} p_{k-1} = \mu_{k-1} p_{k-1}^{\mathrm{T}} \boldsymbol{A} p_{k-1} \tag{22}$$

将式(21)和式(22)代入式(16)可得：

$$\tau_{k-1} = \frac{g_{k-1}^{\mathrm{T}} g_{k-1}}{g_{k-2}^{\mathrm{T}} g_{k-2}} \tag{23}$$

共轭梯度算法的伪代码描述如算法 1 所示。

算法 1 共轭梯度算法

Input：矩阵 **A**，右端向量 **b**

Output：近似解 x

1. r:=$\mathbf{A}x_0-\mathbf{b}$
2. for i:=1 to Maxter do
3. p:=r
4. ρ:=$p^T r$
5. if $\sqrt{\rho}<$ ethr then
6. Converged, exit loop
7. end if
8. p:=−p
9. if i>1 then
10. p:=p + $\frac{\rho}{\rho_{old}}p_{old}$
11. end if
12. t:=Ap
13. $\lambda:=-\frac{p^T r}{p^T t}$
14. $x_0:=x_0+\lambda p$
15. r:=r+λt
16. $p_{old}:=p$
17. $\rho_{old}:=\rho$
18. end for

2.2.2 LU 分解算法

假定矩阵 $\boldsymbol{A}$ 可以写成两部分矩阵相乘的形式，即 $\boldsymbol{A}=\boldsymbol{LU}$，$\boldsymbol{A}$ 被分解为上三角阵和下三角阵的乘积，因此 $\boldsymbol{A}x=\boldsymbol{b}$ 就可以写成：

$$\boldsymbol{LU}x=\boldsymbol{b} \tag{24}$$

通过 $\boldsymbol{L}y=\boldsymbol{b}$ 求解出 y，又因为 $\boldsymbol{U}x=y$，可以求解出 x。

有关并行程序的 LU 分解算法，可以通过直接调用并行分布式线性代数运算工具包 ScaLAPACK(Scalable LAPACK)来实现。ScaLAPACK 是一个并行计算的软件包，它采用消息传递机制来处理进程之间的通信，可以用来求解线性代数系统，如各种矩阵运算、线性方程组求解等。一般并行化代码进程都是一维，但在这里进程会从一维映射到二维，各个进程映射成一个二维数组，其中每一个进程网格都包含在一个上下文(Context)中，每个上下文可以分别通信并且互不影响，一个上下文可以包含一个或多个进程。使用 ScaLAPCK 求解线性代数的步骤为：首先初始化进程网格，然后将数据(矩阵或者向量)按照块状循环方式分布到进程网格，最后调用 ScaLAPACK 求解例程。

LU 分解算法的伪代码描述如算法 2 所示。

算法 2 LU 分解算法

Input：矩阵 **A**，右端向量 **b**

Output：近似解 x

1. **U**=**A**，**L**=**I**
2. for j=1:n−1 do
3. for i=j+1:n do
4. $l_{ij}=\mathbf{U}_{ij}/\mathbf{U}_{jj}$
5. for k=j:n do
6. $u_{ik}=u_{ik}-l_{ij}u_{jk}$
7. end for
8. end for
9. end for

3 实验结果与分析

为验证算法的可行性，我们在“天河二号”上进行了算例测试，将从准确性和计算时间两方面对实验结果进行分析，同时进行了算法的可扩展性的测试。测试环境的相关参数及配置如表 1 所列。

表 1 超级计算机“天河二号”相关参数

Table 1 Parameters of supercomputer TH-2

参数名称	参数值
资源名称	广州超算天河二号
CPU 型号	Intel Xeon E5-2692
节点核数	单节点 2 * 12 核
频率/GHz	2.200

3.1 正确性和一致性

本次实验中，所使用的算例为聚乙烯体系 $H(C_2H_4)_n$，如图 1 所示，这里重复的单元为阴影部分标记的 C_2H_4，白色原子代表氢原子，黑色原子代表碳原子。测试原子数的体系分别为 50，98，194，386，542，770，1 154。

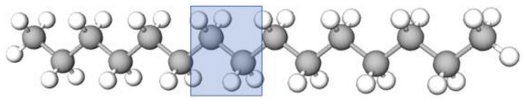

图 1 聚乙烯分子 $H(C_2H_4)_nH$

Fig. 1 Polyethylene molecule $H(C_2H_4)_nH$

为验证共轭梯度算法和矩阵直接分解算法的准确性，将共轭梯度算法和直接分解算法的 FHI-aims 程序的计算结果与传统计算方法进行对比，对比的结果均采用第一轮 CPSCF 的计算结果，均保留小数点后两位。计算结果如表 2 所列。

表 2 不同体系下不同方法的计算结果

Table 2 Calculation results of different methods under different systems

原子数	benchmark	LU	CG
50	59.54	59.54	59.54
98	118.91	118.91	118.91
194	237.67	237.67	237.67
386	475.19	475.19	475.11
542	668.17	668.17	668.06
770	950.21	950.21	950.06
1 154	1 425.24	1 425.24	1 425.01

图 2 中 benchmark 代表传统计算方法，LU 为直接分解算法，CG 为共轭梯度算法。从图 2 和表 3 可以看出，在小规模体系计算中(原子数小于 100)，其他 3 种方法与传统方法的计算结果基本一致。随着模拟体系规模增大，LU 方法与传统计算方法的误差几乎为 0，CG 方法与传统计算方法的误差也较小，说明迭代算法中的 CG 方法和直接算法中的 LU 方法准确性都较高。

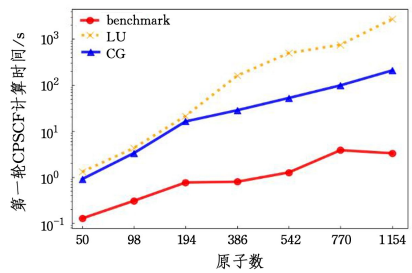

图 2 不同体系下不同方法的计算时间

Fig. 2 Calculation time of different methods under different systems

表 3 相对误差的绝对值

Table 3 Absolute value of relative error

原子数	benchmark	LU	CG
50	—	10^{-5}	10^{-5}
98	—	10^{-5}	10^{-5}
194	—	10^{-5}	10^{-5}
386	—	10^{-5}	0.000168
542	—	10^{-5}	0.000165
770	—	10^{-5}	0.000158
1154	—	10^{-5}	0.000161

下面比较在不同体系下不同方法的计算时间，模拟条件与上文保持一致，计算时间如图 3 所示。

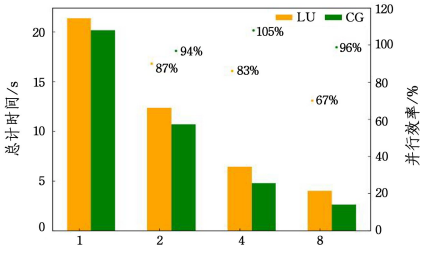

图 3 两种算法的强可扩展性的并行效率

Fig. 3 Parallel efficiencies of strong scalability of two algorithms

从图 3 中可以观察到，CG 方法和 LU 方法与传统 u 方法相比计算耗时均更长，传统 u 方法计算时间最短。可以看出与 LU 方法相比，CG 方法在计算时间上更有优势，在计算较小的原子体系时 CG 方法和 LU 方法的计算时间较为接近，但在计算更大的原子体系时，CG 方法的计算时间比 LU 方法更短。

3.2 共轭梯度算法的精度测试

为验证迭代法中 CG 方法的准确性，我们在不同条件下测试了该程序计算结果的精度。

首先，CG 方法采用 386 个原子的体系进行测试，分别更改算法的迭代次数，令其收敛精度分别为 10^{-5}，10^{-4}，10^{-3}，均采用第一轮 CPSCF 的计算结果，将不同精度下的结果与传统计算方法 u 方法的结果进行对比，实验结果如表 4 所列。

表 4 不同迭代次数算法的精度

Table 4 Accuracy of algorithm with different number of iterations

收敛精度	u	CG
10^{-5}	475.18	475.18
10^{-4}	475.18	475.18
10^{-3}	475.18	475.18

将计算结果与表 2 中传统计算 u 方法的 386 体系的结果进行对比，计算结果保留小数点后两位，可以看出 CG 方法在不同迭代次数下的计算结果基本一致。

表 5 列出了 CG 方法在不同迭代次数下的计算时间，可以看出随着迭代次数的增加，CG 方法计算的时间也有所增加。

表 5 不同迭代次数下的计算时间

Table 5 Computation time for different number of iterations

收敛精度	CG 方法计算时间/s
10^{-5}	208.012
10^{-4}	185.227
10^{-3}	66.617

此外，为测试 CG 方法的精度，本文采用小规模的原子数量进行测试(原子数小于 100)，其计算结果如表 6 所列。

表 6 小规模原子数量下解方程时间及迭代次数

Table 6 Solution time and number of iterations for equations with small number of atoms

原子数	CG
50	0.534 s *Min iter*=2 *Max iter*=146
98	1.512 s *Min iter*=2 *Max iter*=309

原子数量为 50 时 CG 方法解方程总耗时为 0.534 s，原子数量为 98 时 CG 方法解方程总耗时为 1.512 s。*Min iter* 和 *Max iter* 分别为方程的最小和最大迭代次数。在原子数为 50 时 CG 方法的计算结果为 59.54，原子数为 98 时计算结果为 118.91。与表 2 中 u 方法的计算结果进行对比可以观察到，在计算小规模原子体系时，CG 方法和传统 u 方法的计算结果基本一致，且精度较高。

3.3 可扩展性测试

为分析引入迭代算法和直接算法的程序性能，本文从可扩展性测试进行了分析。在扩展性测试中，测试的原子数体系保持 1538 不变，所使用的 CPU 核数依次为 1 个、2 个、4 个和 8 个。分别对传统 LU 方法和 CG 方法进行测试，并且以 1 个核为基准计算其他测试组的计算比和程序的并行效率。测试结果如表 7 和表 8 所列。

表 7 LU 算法强扩展性测试结果

Table7 Test results of LU algorithm's strong scalability

CPU 核数	模拟用时/s	加速比	并行效率/%
1	21.316450250	1.00	100
2	12.314805750	1.73	87
4	6.411373862	3.32	83
8	3.966194642	5.37	67

表 8 CG 算法强扩展性测试结果

Table 8 Test results of CG algorithm's strong scalability

CPU 核数	模拟用时/s	加速比	并行效率/%
1	20.135521060	1.00	100
2	10.698882890	1.88	94
4	4.760458335	4.23	105
8	2.617948012	7.69	96

两种算法的并行效率如图 3 所示，结合表 7 表 8 可以看出当 CPU 核数逐渐增加时，模拟用时有所减少，两种算法都展现出较为可观的加速比和并行效率。同时比较 CG 方法和 LU 方法的扩展性，可观察到随着 CPU 核数的增加，LU 方法的并行效率在逐渐降低，这是因为增加 CPU 核数后会导致进程之间的通信时间开销较大，所以并行效率会逐渐降低，而 CG 方法的并行效率并没有因 CPU 核数的增加而受到影响，其最低并行效率依旧可以达到 94%，这也可以说明迭代算法中的 CG 算法更适用于求解本文中的线性方程，其效率明显更高。

结束语 在密度泛函微扰理论中，Sternheimer 方法通常被用来计算分子极化率、振动频率等的相关信息。本工作在

Sternheimer 方法的基础上，引入共轭梯度算法和矩阵直接分解算法来对相应密度矩阵进行求解，并成功地在 FHI-aims 程序中实现，比较了不同方法的准确性以及计算时间。实验结果表明，本文采用的迭代算法中的 CG 方法和直接分解算法中的 LU 方法的计算结果与传统计算 u 方法的计算结果基本一致，在计算较大原子体系时 CG 算法的计算时间更短且效率更高，同时其扩展性也明显优于 LU 方法。本文所使用的响应密度矩阵求解方法是可行的，其为量子微扰理论中的响应密度矩阵求解提供了有力支撑，但在计算较大原子体系时，其时间复杂度会更高，计算时间也更长，目前仅适用于密度泛函理论中 Sternheimer 方程的求解。因此在后续的工作中应考虑在计算较大体系时，解决迭代次数过多导致计算时间较长这一问题，对算法进行适当改进。

参 考 文 献

[1] SHAM L J，KOHN W. One-particle properties of an inhomogeneous interacting electron gas [J]. Physical Review，1966，145(2)：561.

[2] ALZAHRANI A Z. Structural and electronic properties of graphene upon molecular adsorption：DFT comparative analysis [J]. Graphene Simulation，2011，1：21-38.

[3] CHAKRABORTY D，CHATTARAJ P K. Conceptual density functional theory based electronic structure principles [J]. Chemical Science，2021，12(18)：6264-6279.

[4] BESLEY N A. Modeling of the spectroscopy of core electrons with density functional theory [J]. Wiley Interdisciplinary Reviews：Computational Molecular Science，2021，11(6)：e1527.

[5] GONZE X. First-principles responses of solids to atomic displacements and homogeneous electric fields：Implementation of a conjugate-gradient algorithm [J]. Physical Review B，1997，55(16)：10337.

[6] GONZE X，LEE C. Dynamical matrices，Born effective charges，dielectric permittivity tensors，and interatomic force constants from density-functional perturbation theory [J]. Physical Review B，1997，55(16)：10355.

[7] BARONI S，DE GIRONCOLI S，DAL CORSO A，et al. Phonons and related crystal properties from density-functional perturbation theory [J]. Reviews of Modern Physics，2001，73(2)：515.

[8] OCHSENFELD C，HEAD-GORDON M. A reformulation of the coupled perturbed self-consistent field equations entirely within a local atomic orbital density matrix-based scheme [J]. Chemical Physics Letters，1997，270(5/6)：399-405.

[9] FRISCH M，HEAD-GORDON M，POPLE J. Direct analytic SCF second derivatives and electric field properties [J]. Chemical Physics，1990，141(2/3)：189-196.

[10] LIANG W Z，ZHAO Y，HEAD-GORDON M. An efficient approach for self-consistent-field energy and energy second derivatives in the atomic-orbital basis [J]. The Journal of Chemical Physics，2005，123(19)：194106.

[11] SHANG H，CARBOGNO C，RINKE P，et al. Lattice dynamics calculations based on density-functional perturbation theory in real space [J]. Computer Physics Communications，2017，215：26-46.

[12] STERNHEIMER R M. Electronic polarizabilities of ions from the Hartree-Fock wave functions [J]. Physical Review，1954，96(4)：951.

[13] MOTORNYI O，VAST N，TIMROV I，et al. Electron energy loss spectroscopy of bulk gold with ultrasoft pseudopotentials and the Liouville-Lanczos method [J]. Physical Review B，2020，102(3)：035156.

[14] HOFMANN F，SCHELTER I，KÜMMEL S. Efficiently evaluating the Krieger-Li-Iafrate and common-energy-denominator approximations in the frequency-dependent Sternheimer scheme [J]. Physical Review A，2019，99(2)：022507.

[15] KAMEARI A. Improvement of ICCG convergence for thin elements in magnetic field analyses using the finite-element method [J]. IEEE Transactions on Magnetics，2008，44(6)：1178-1181.

[16] HESTENES M R，STIEFEL E. Methods of conjugate gradients for solving linear systems [M]. Washington：NBS，1952.

LIU Renyu，born in 1997，postgraduate. His main research interests include computer application technology and so on.

SHANG Honghui，born in 1984，Ph.D，associate professor. Her main research interests include the development of the first-principles methods and their applications on the high-performance computer systems.

（责任编辑：何杨）