Computer Physics Communications 267 (2021) 108045

Contents lists available at ScienceDirect

Computer Physics Communications

www.elsevier.com/locate/cpc

Many-core acceleration of the first-principles all-electron quantum perturbation calculations ☆

Honghui Shang [a,*], Xiaohui Duan [c,*], Fang Li [b,*], Libo Zhang [b], Zhiqian Xu [a], Kan Liu [c], Haiwen Luo [a], Yingrui Ji [a], Wenxuan Zhao [a], Wei Xue [c], Li Chen [a], Yunquan Zhang [a]

[a] State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[b] National Supercomputer Center in Wuxi, Wuxi, China
[c] Tsinghua University, Beijing, China

ARTICLE INFO

Article history:
Received 4 November 2020
Received in revised form 14 May 2021
Accepted 24 May 2021
Available online 2 June 2021

Keywords:
Density-functional perturbation theory
Many-core architecture
Linear scaling
MPI
Numeric atomic orbitals

ABSTRACT

The first-principles quantum perturbation theory, also called density-functional perturbation theory (DFPT), is the state-of-the-art formalism to directly link the experimental response properties of the materials with the quantum modeling of the electrons. Here in this work, we present an implementation of all-electron DFPT for massively parallel Sunway many-core architectures to accelerate DFPT calculation. We have paid special attention to the calculation of the response density matrix, the real-space integration of the response density as well as the response Hamiltonian matrix. We also employ the fast and massively parallel linear scaling scheme together with the load balance algorithm for the DFPT calculations to improve the scalability. Using the above approaches, the accurate first-principles quantum perturbation calculations can be extended over millions of cores.

© 2021 Published by Elsevier B.V.

1. Introduction

The physical properties measured in the experiments are directly related to the quantum response/perturbation of the system. Such response properties, related to the second or higher-order derivatives of the total energy can be calculated within the uniform quantum mechanical framework by means of density-functional perturbation theory (DFPT) [1,2]. Such quantum perturbation theory can provide the accurate predictions for many fundamental physical phenomena, such as super-conductivity [3], vibrational frequencies or phonon dispersions [4], polarizability [5,6], harmonic and anharmonic Raman spectra [7], and much more [2].

In order to calculate the above properties theoretically, the DFPT method has been implemented in the computational packages. Depending on the different form of basis function, the different numerical method is chosen. The basis set can be the plane-wave [8,9], the uniform real-space grids [10], the periodic sinc functions [11], the b-spline functions [12], the finite elements [13], or the wavelets [14]. Although the above basis sets can be converged systematically, the oscillatory behavior near the atomic nucleus, as shown in Fig. 1(a), cannot be accurately represented because of too heavy computation (e.g. 10^5 plane waves are needed for one core orbital). As a result, when using the above basis sets, the pseudization methods [15] using pseudo-potentials or projector-augmented wave (PAW) have been introduced, in which the core potential has been replaced with a 'fake' one. Although the pseudo-potentials have been carefully constructed to keep the valence part to be consistent with the all-electron method, the information of the core shells is still missing. In order to consider the core and valence states on the equal footing, the all-electron approaches have been developed, e.g. linearized augmented plane wave (LAPW) [16], linear muffin-tin orbital (LMTO) [17] methods and all-electron Gaussian type orbital (GTO) [18] or all-electron numerical atomic orbitals method (NAO) [19,20]. Such all-electron methods can achieve better precision compared with the pseudization method [15].

For codes using all-electron atomic orbitals, such as Gaussian basis set, the DFPT has been mainly implemented to treat finite, isolated systems [21,22], and only a few literature reports exist for the treatment of periodic boundary conditions [23] with only the Γ-point perturbations. The implementation of the DFPT in all-electron scheme for extended systems is not trivial [4,6]. This is because for periodic systems, the perturbations in DFPT destroy the periodic boundary conditions, and the atomic displacements cause a change of the entire basis set, the construction of the related matrix elements is complex and not as straightforward as

☆ The review of this paper was arranged by Prof. Blum Volker.

* Corresponding authors.
E-mail addresses: shanghui.ustc@gmail.com, shanghonghui@ict.ac.cn (H. Shang), sunrise_duan@126.com (X. Duan), lifang56@163.com (F. Li).

https://doi.org/10.1016/j.cpc.2021.108045
0010-4655/© 2021 Published by Elsevier B.V.

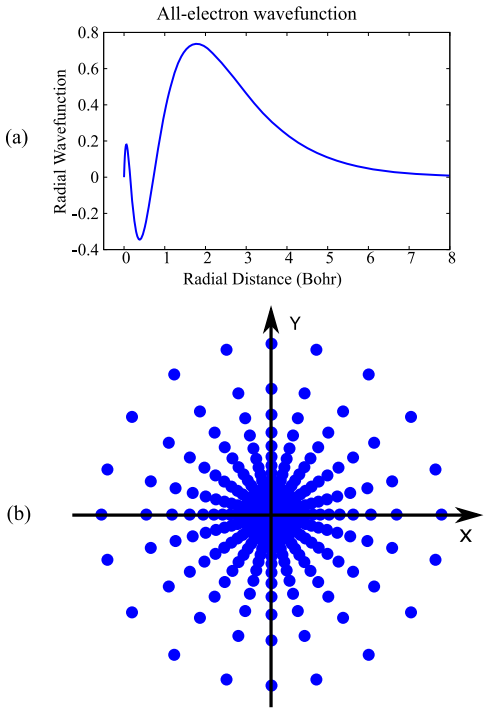

Fig. 1. (a) The all-electron wavefunction of the 3s orbital of the silicon atom; (b) The atom-centered grid used for the all-electron scheme, here we only illustrate the grids in the XY plane for simple.

expected. Therefore, compared to DFT, the parallelization and the corresponding optimizations are much more complicated, and the implementations of DFPT using all-electrons scheme for both the finite (molecules) and extended (periodic) systems are rare and lacking subsequent optimization, for example, the implementation using linear muffin tin orbitals [24] or linearized augmented plane-waves [25,26], or Gaussian basis set for only the electric field perturbation [18]. The all-electron, full-potential, numerical atomic orbitals based FHI-AIMS package [4,6,20,27] is the only massively scalable code, which can perform the all-electrons DFPT calculation for atomic perturbation and electric field perturbation to deal with both the finite (molecules) and extended (periodic) systems on the equal footing. Here in this work, the many-cores optimization of the DFPT method in FHI-AIMS package are presented to improve the computational efficiency. In our scheme, the all-electron atomic orbitals are discretized using the atom-centered grid [28] as illustrated in Fig. 1(b), in order to treat the all-electron full-potential systems.

The rest of this paper is organized as follows: In Sec. 2, the basic first-principle quantum perturbation algorithms are introduced. Then we show our optimization strategies in Sec. 3. Performance is finally described and analyzed in Sec. 4. Sec. 5 concludes the paper.

2. The first-principles quantum perturbation theory

The first-principles perturbation calculation is the key to get the static or dynamic response physical properties. Here we only briefly summarize the first-principles quantum perturbation approach. The starting point is the zero order Kohn-Sham equation, which is the single-particle form of the Schrödinger equation

$$\hat{h}_{KS}\psi_i = \left[\hat{t}_s + \hat{v}_{ext}(r) + \hat{v}_H + \hat{v}_{xc}\right]\psi_i = \epsilon_i\psi_i \,, \tag{1}$$

for the Kohn-Sham Hamiltonian $\hat{h}_{KS}$. In Eq. (1) $\hat{t}_s$ is the single particle kinetic operator, $\hat{v}_{ext}$ the (external) electron-nuclear potential, $\hat{v}_H$ the Hartree potential, and $\hat{v}_{xc}$ the exchange-correlation potential. Solving Eq. (1) yields the Kohn-Sham single particle states ψ_i and their eigenenergies ϵ_i. In our numerical implementation, the wavefunction ψ_i is expanded in terms of numeric atom-centered orbitals (NAOs) [20,29,30] $\chi_\mu(\mathbf{r})$

$$\psi_i(\mathbf{r}) = \sum_\mu C_{\mu i}\,\chi_\mu(\mathbf{r}) \,, \tag{2}$$

The zero order expansion coefficients $C_{\mu i}$ need to be determined by solving the generalized algebraic eigenvalue problem

$$H^{(0)}C^{(0)} = S^{(0)}C^{(0)}E^{(0)} \,, \tag{3}$$

whereby $E^{(0)}$ denotes the diagonal matrices containing the eigenvalues ϵ_i, and the ground state Hamiltonian matrix $H^{(0)}$, coefficients matrix $C^{(0)}$ and overlap matrix $S^{(0)}$ are all of size $N_{orb} \times N_{orb}$, where N_{orb} is the number of orbitals in the whole system.

In order to perform the quantum perturbation calculation, the above ground state form of the Schrödinger equation need to be perturbed with the external response, hence, the response form of Eq. (3) within the first order is

$$\begin{aligned} &H^{(0)}C^{(1)} - S^{(0)}C^{(1)}E^{(0)} - S^{(1)}C^{(0)}E^{(0)} \\ &= -H^{(1)}C^{(0)} + S^{(0)}C^{(0)}E^{(1)} \,, \end{aligned} \tag{4}$$

whereby $E^{(0)}$ and $E^{(1)}$ denote the diagonal matrices containing the eigenvalues ϵ_i and $\epsilon_i^{(1)}$ respectively, $H^{(1)}$ is the response Hamiltonian matrix, $C^{(1)}$ is the response coefficients matrix and $S^{(1)}$ is the response overlap matrix. The Eq. (4) are called Sternheimer equation [31], which is the key to get the corresponding response density matrix (here f_i denotes the occupation number of eigenstate)

$$P^{(1)}_{\mu,\nu} = \sum_i f(\epsilon_i)\left(C^{(1)}_{\mu,i}C^{(0)}_{\nu,i} + C^{(0)}_{\mu,i}C^{(1)}_{\nu,i}\right) , \tag{5}$$

in the density-functional perturbation theory (DFPT) [1,2,4,6,32].

The flowchart of DFPT is described as follows: After the ground state calculation with DFT is completed, the response of the overlap matrix is calculated. Then the DFPT cycle begins by using an initial guess for the response of the density matrix $P^{(1)}$, which then allows to construct the respective density $n^{(1)}(\mathbf{r})$. The associated response of the electrostatic potential $V^{(1)}_{es,tot}(\mathbf{r})$ is calculated by solving the Poisson equation in real space. The response Hamiltonian $H^{(1)}$ is calculated with the response density and potential. In turn, all these ingredients then allow to set up the Sternheimer equation, the solution of which allows to update the response of the expansion coefficients $C^{(1)}$. Using a linear or more efficient Pulay-mixing scheme [33] to accelerate the convergence, we iteratively repeat the DFPT loop until self-consistency is reached, i.e., until the changes in $P^{(1)}$ become smaller than a user-given threshold. Finally, the physical properties are evaluated using the converged response density matrix.

3. Implementation and optimization

3.1. The SW26010 many-core processor

The Sunway TaihuLight supercomputer comprises of 40,960 nodes (10,649,600 cores) with the peak performance over 125 PFLOPS. Each node has one SW26010 many-core processor, as shown in Fig. 2. One SW26010 processor contains 4 core-groups (CGs), with 65 cores in each CG, and in total 260 cores. Each CG contains one management processing element (MPE), one cluster of computing processing elements (CPEs) and one memory controller (MC). The MPE within each CG is used for computations,

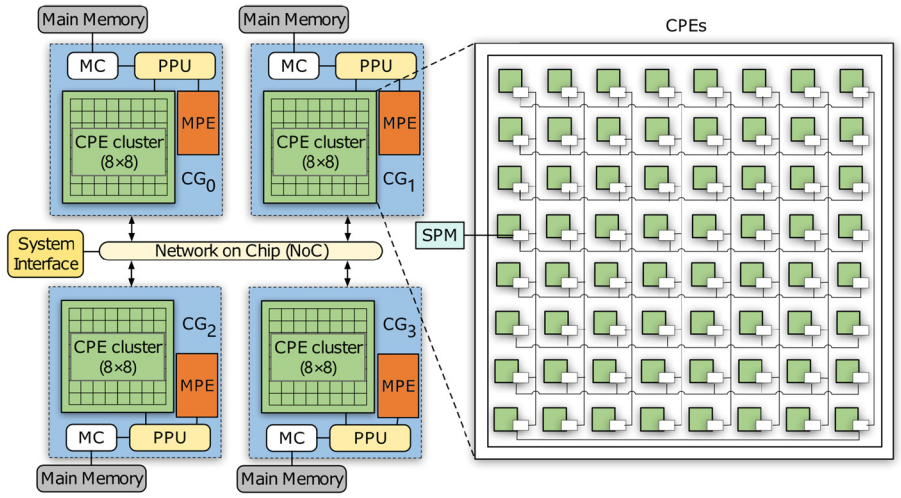

Fig. 2. The architecture of the many-core SW26010 CPU.

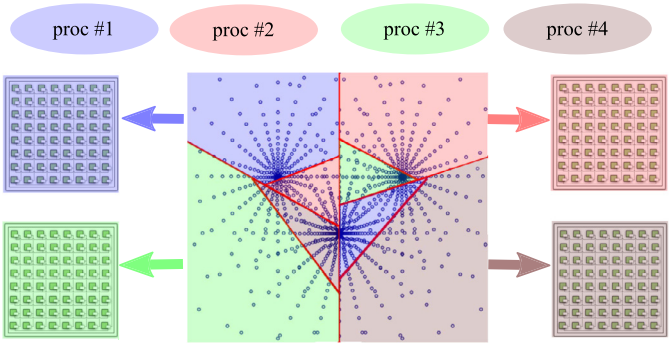

Fig. 3. Two level parallelization for the grid integration. In the first process-level parallelization, the batches are loaded on each process as labeled by different colors in this figure. In the thread-level parallelization, the grids inside each batch are loaded on the CPEs to make many-core acceleration. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

management and communication. The CPEs is organized as an 8×8 mesh (64 cores) and is designed to maximize the aggregated computing power and to minimize the complexity of the micro-architecture. Each SW26010 processor contains 32 GB memory, with 8 GB memory in each CG. Each MPE has a 32 KB L1 instruction cache, a 32 KB L1 data cache, and a 256 KB L2 cache for both instruction and data. Each CPE has a 16 KB L1 instruction cache, and a 64 KB scratch pad memory (SPM, also called the Local Data Memory (LDM)), which serves the same function as the L1 cache, but in a user-controlled way. The MPE and the CPEs within the same CG share the same memory which is controlled by the MC. For the network, within the processor, the 4 CGs are connected using a network on chip (NoC). The 256 processors inside a supernode are fully connected through a customized network board, then the supernodes are connected with the central network switches.

3.2. Two-level parallelization strategy

In this work, all the perturbation properties are calculated within the discretized three-dimensional physical grids, which is suitable for massively parallel implementations. The non-evened atom-centered grids for a three-atom system are illustrated in Fig. 3. In such atom-centered grid, a partition procedure for the grids is firstly made for each atom, and then the single-center (atom) grids are further separated into radial and angular parts, that radially the atom-centered grid consists of several spherical integration shells with radial integration weight w_{rad} [34], and on these shells, angular integration points are distributed in such a way that spherical harmonics up to a certain order are integrated exactly by using the Lebedev grids [35], with angular integration weights w_{ang}. In order to partition this non-even grid meshes in an efficient way, the grid adapted cut-plane method is used to make batches [36], as shown in Fig. 3 with different colors. The procedure to obtain batches is as follows: Firstly the center of the mass for all the points are computed; Secondly, the direction of the cut-plane are gotten by computing the normal of a plane; Thirdly, the position of the cut-plane are computed to divide all the points into two even-sized sets and the full points are split into two subsets (batches) using the cut-plane. The above procedure is repeated until getting all the batches with the desired size (around 100 to 300 points within each batch).

Two-levels of the parallelization has been adopted for the calculations of the numerical integration: The first level parallelization is performed over the batches, which are distributed across all MPI processes to achieve good parallel scalability using the adapted batch distribution algorithm to achieve load balance, as discussed in the following section; The second level parallelization is performed inside each batch, in which, the calculation of the integration is distributed on the cores within CPEs by looping over the number of the points and the number of the computed basis functions. The acceleration on CPEs can further improve the performance. The load balancing for the integration is achieved by eventually distributing the integration points over the MPI processes. The batches/grids are distributed according to the current summation of the points (use linked list) in each process, the new batch is always sent to the process with the minimal number of points, in this way, the load balance for the grid integration is achieved [36].

During the integration calculation, e.g. response Hamiltonian matrix, as shown in Fig. 4, each batch (labeled with a different color) is calculated locally for one part of the matrix element, and finally the matrix element is summed up using MPI_Allreduce collective communication.

3.3. The $O(N)$ solver to get the first order density matrix

In the traditional way [21] to get the response density matrix, we have the $O(N^3)$ computational scaling because of the including of the dense matrix (expansion coefficients matrix) multiplication. In order to reduce the $O(N^3)$ scaling to $O(N)$ scaling, the dense matrix operations need to be avoided, hence the purification related method [37] is a promising choice. In the purification

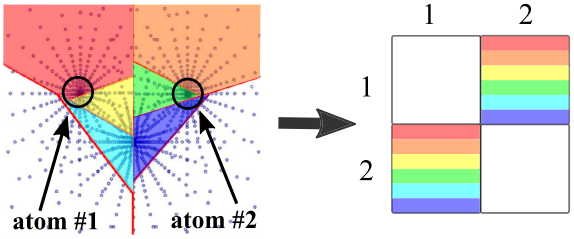

Fig. 4. The parallel integration method for the response Hamiltonian matrix element between two atoms. Here we assume one atom has one orbital, so we have a 2×2 matrix on the right side. The integration between atoms-1 and atoms-2 gives the matrix elements M(1,2) and M(2,1). The integration is performed locally in each batch (labeled with different color) and the final first order Hamiltonian matrix element is calculated by merging the matrix element over the corresponding batches.

Algorithm 1 The linear scaling TC2-DFPT algorithm, where N_{occ} is the number of occupied states, $\varepsilon_{min}(H)/\varepsilon_{max}(H)$ denotes the minimal/maximum boundary for the eigenvalues of Hamiltonian matrix $H^{(0)}$, which is estimated with Gershgorin's Circle Theorem.

Require:
$H^{(0)}$: zero order Hamiltonian
$S^{(0)}$: zero order Overlap
$H^{(1)}$: first order Hamiltonian
$P^{(1)}$: first order density matrix

$H^{(0)}_{orth} \leftarrow S^{(0)-\frac{1}{2}}\ H^{(0)}\ S^{(0)-\frac{1}{2}}$
$H^{(1)}_{orth} \leftarrow S^{(0)-\frac{1}{2}}\ H^{(1)}\ S^{(0)-\frac{1}{2}}$
estimate $\varepsilon_{min}(H^{(0)}_{orth}), \varepsilon_{max}(H^{(0)}_{orth})$
$X^{(0)}_0 \leftarrow (\varepsilon_{max} I - H^{(0)}_{orth})/(\varepsilon_{max} - \varepsilon_{min})$
$X^{(1)}_0 \leftarrow (-H^{(1)}_{orth})/(\varepsilon_{max} - \varepsilon_{min})$
for $n \leftarrow 0, max$ **do**
  if $Tr[X^{(0)}_n] - N_{occ} < 0$ **then**
    $X^{(0)}_{n+1} \leftarrow 2X^{(0)}_n - (X^{(0)}_n)^2$
    $X^{(1)}_{n+1} \leftarrow 2X^{(1)}_n - X^{(1)}_n X^{(0)}_n - X^{(0)}_n X^{(1)}_n$
  else
    $X^{(0)}_{n+1} \leftarrow (X^{(0)}_n)^2$
    $X^{(1)}_{n+1} \leftarrow X^{(1)}_n X^{(0)}_n + X^{(0)}_n X^{(1)}_n$
  end if
  Error $\leftarrow \max(X^{(0)}_{n+1} - X^{(0)}_n)$
  if Error $\leq$ tolerance **then**
    break
  end if
end for
$P^{(0)} \leftarrow S^{(0)-\frac{1}{2}}\ X^{(0)}_n\ S^{(0)-\frac{1}{2}}$
$P^{(1)} \leftarrow S^{(0)-\frac{1}{2}}\ X^{(1)}_n\ S^{(0)-\frac{1}{2}}$

method, only the zero/first order density and Hamiltonian matrices are used for the matrix multiplication, so the sparse matrix multiplication can be applied since both the density and Hamiltonian matrices can be written in the sparse matrix form. Here we use the orthogonal formulation of the second order trace-correcting purification (TC2) method for DFPT [38], which is called TC2-DFPT. Within this approach, the first order density matrix $P^{(1)}$ is explicitly constructed from the first order Hamiltonian matrix $H^{(1)}$. As shown in Algorithm 1, the response density matrix is calculated only with sparse matrix-matrix multiplication, which is the key to achieve linear scaling, and provides the base for computing the response density matrix explicitly and rapidly. There are two parameters to control the simulation error of the TC2-DFPT method. One parameter is the filter, which refers to the threshold to determine which matrix elements can be treated as zero. The other parameter is the tolerance, which is the convergence-threshold which compared the band energy between the current iteration and the last iteration.

The MPI level distributed memory parallelization of the sparse matrix-matrix multiplication can be performed with 3D (or 2.5D) SpGEMM algorithm [39,40]. In this algorithm, each matrix is distributed along the cubic $\sqrt{P/c} \times \sqrt{P/c} \times c$ processes grid, where

Algorithm 2 The load balance method in sparse matrix-matrix multiplication.

Require:
n: matrix dimension
index_lookup: permutation array
A : input sparse matrix
A' : load-balanced sparse matrix

for $i \leftarrow 1, n$ **do**
  index_lookup(i) $\leftarrow i$
end for
for $i \leftarrow n, 1$ **do**
  $j \leftarrow$ randint$(1, n)$
  swap(index_lookup(n), index_lookup(j))
end for
Broadcast index_lookup to all processes
let P_{row} and P_{col} be empty matrices
for $i \leftarrow 1, n$ **do**
  $P_{row}(i, index\_lookup(i)) \leftarrow 1$
  $P_{col}(index\_lookup(i), i) \leftarrow 1$
end for
$A' \leftarrow P_{row} A P_{col}$

$1 < c < \sqrt[3]{P}$ and P is the number of the total processes. Then each matrix is broadcasted and multiplied locally to compute a contribution to a local result matrix, and finally the matrix is summed up using non-blocking all-to-all collective communication. It should be noted that, since the input matrices are usually not well load balanced, the elements are redistributed by permuting the rows and columns in order to achieve the load balance in the sparse matrix-matrix multiplication, as shown in Algorithm 2.

3.4. Many-core acceleration strategy

For many-core acceleration, we use athread to achieve a suitable mapping of the DFPT code in FHI-AIMS onto the Sunway processors. Since the on-chip fast buffer and the available memory bandwidth of the Sunway processor are relatively limited, the memory usage during the porting is significantly more challenging [41]. The major loops are paralleled in CPE cluster architecture, and the frequently-accessed variables are copied into the local fast buffer of the CPE.

For the calculations of the first order density and first order Hamiltonian within one processor, the first loop is over the number of batches (n_{batches}) allocated in the current processor. Then inside one batch, another loop is over the number of the points (n_{points}) in this batch, which can be paralleled with athread. Here in this work, the athread acceleration is used for the n1_part2 as shown in Algorithm 3 and for the H1_part1 in Algorithm 4.

Besides the calculation of the loops over n_points, the dense matrix-matrix multiplication (DGEMM) is another performance bottleneck in the calculation of integrations. The parallel blocked matrix-matrix multiplication algorithm [42] is used to achieve high performance. The tile block of the matrix is loaded to the CPEs via high bandwidth asynchronized DMA (Direct Memory Access) mechanism at one time to accelerate the data transformation to CPEs, in which a double-buffering technique is used to hide the non-negligible transferring time. Then the SIMD (Single Instruction Multiple Data) assembly codes are adopted to further improve the performance by using the available 256-bit SIMD vector registers in the SW26010 processor [43]. Such optimized DGEMM is applied in the calculations of n1_part1 as shown in Algorithm 3 and the H2_part1 in Algorithm 4.

We have also ported the TC2-DFPT scheme to the Sunway many-core architecture. Here we accelerate the local sparse matrix-matrix multiplication with swSparse, a sparse library for Sunway many-core architecture using athread, which can gain an average speedup of 9.2×/10.4× compared to the original version for the sparsity of 18.9%/4.5%.

Algorithm 3 The many-core optimization of the calculation of first order density.

Require:
$P^{(1)}$: first order density matrix $n^{(1)}$: first order density

1: **for** $i \leftarrow 1, n_{\text{batches}}$ **do**
2: get n_{point} and n_{compute} in current batch
3: $P^{(1)}_{\text{local}} \leftarrow$ global sparse matrix $P^{(1)}$
4: dgemm($P^{(1)}_{\text{local}}$, wave, M) ▷ **n1_part1**
5:
6: **for** $j \leftarrow 1, n_{\text{points}}$ **do** ▷ **n1_part2**
7: $n^{(1)}_{local}(j) \leftarrow \sum_{k}^{n_{\text{compute}}} M(k, j)\ \text{wave}(k, j)$
8: **end for**
9: global $n^{(1)} \leftarrow n^{(1)}_{local}$
10: **end for**

Algorithm 4 The many-core optimization of the calculation of first order Hamiltonian.

Require:
$V^{(1)}$: first order potential
$H^{(1)}$: first order Hamiltonian

1: **for** $i \leftarrow 1, n_{\text{batches}}$ **do**
2: get n_{point} and n_{compute} in current batch
3: $V^{(1)}_{\text{local}} \leftarrow$ global sparse matrix $V^{(1)}$
4: **for** $j \leftarrow 1, n_{\text{points}}$ **do** ▷ **H1_part1**
5: **for** $k \leftarrow 1, n_{\text{compute}}$ **do**
6: $M(k, j) = w(j)V^{(1)}_{\text{local}}(j)\ \text{wave}(k, j)$
7: **end for**
8: **end for**
9: dgemm(M, wave, $H^{(1)}_{\text{local}}$) ▷ **H1_part2**
10: global sparse $H^{(1)} \leftarrow H^{(1)}_{\text{local}}$
11: **end for**

4. Performance results

4.1. Validation

In order to validate the linear scaling TC2-DFPT method, we show the convergence behavior of the linear scaling TC2-DFPT method for a system containing 386 atoms in Fig. 5. Here we use the "tier 1" basis set, light grid setting,and the LDA functional [44]. The benchmark polarizability is calculated with the $O(N^3)$ method. The polarizability difference ($\Delta\alpha$=$\alpha - \alpha_{benchmark}$) calculated with various setting (filter, tolerance) are plotted, which clearly shows a rapid convergence to the benchmark result. The error at set-3 is already 0.0022 Bohr^3 (5.7$\times$ 10^{-6} Bohr^3/atom) and if we increase to use set-5, the error reduces to 3$\times$ 10^{-5} Bohr^3 (7.8$\times$ 10^{-8} Bohr^3/atom).

4.2. Single node performance

The speedups of different DFPT kernels within one SW26010 processor (i.e. 4 CGs, 260 cores) are presented in this part. The speedups are measured by the calculation time of 4 MPE divide by the calculation time of 4 MPE and the 256 CPEs. We use silicon solid as an example, for which we compute the three diagonal components of the polarizability tensor and the dielectric constant using LDA functional, the 'tight' numerical setting and 'tier 3' basis set, with 7×7×7 **k**-points in the primitive unit cell.

In Fig. 6, we show the speedup for the first order density ($n^{(1)}$). The upper panel shows the result for the first part of $n^{(1)}$ while the lower panel shows the result for the second part of $n^{(1)}$. The speedup depends on two parameters: n_{point} and n_{compute}. The n_{point} is the number of the points in one batch, and the n_{compute} is the number of the basis functions in one batch. In the first part, a local dense density matrix multiplication (dgemm) is performed. Here we use the athread optimized dgemm function to accelerate the calculation. The speedup ratio for this kernel increases from

label	filter	tolerance	$\Delta\alpha$ (Bohr^3)
set-1	10^{-6}	10^{-5}	0.0487
set-2	10^{-8}	10^{-4}	0.0355
set-3	10^{-7}	10^{-5}	0.0022
set-4	10^{-8}	10^{-5}	0.00006
set-5	10^{-8}	10^{-6}	0.00003

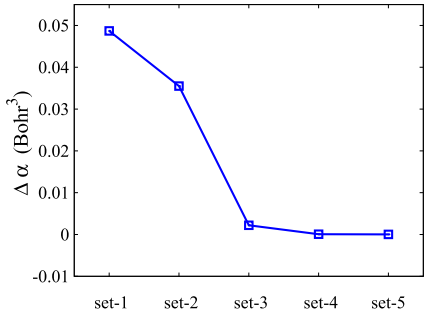

Fig. 5. Convergence behavior of the $O(N)$ TC2-DFPT method for the polarizability difference ($\Delta\alpha$=$|\alpha - \alpha_{\text{benchmark}}|$) with respect to numerical parameters (filter, tolerance) as shown in the upper Table. Here the parameter filter refers to the threshold to determine which matrix elements can be treated as zero while the parameter tolerance means the convergence-threshold within one SCF cycle.

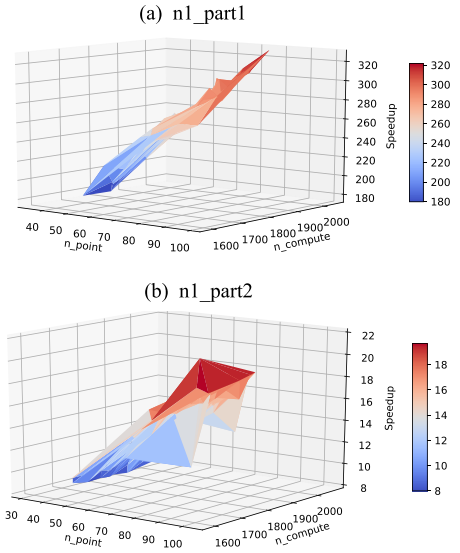

Fig. 6. The performance speedup for the first order density ($n^{(1)}$) within one Sunway many-core processor. The speedup is plotted with respect to n_{point} and n_{compute} in every local batch. The upper panel(a) shows the speedup of the first part of the $n^{(1)}$ calculation and the lower panel (b) shows the speedup of the second part of the $n^{(1)}$ calculation.

175$\times$ to 328$\times$ with increasing the two parameters n_{point}/n_{compute} from 34/1834 to 78/2029. The high speedup of the dgemm kernel comes from the optimization with architecture-aware methodologies to exploit data locality and the effective memory bandwidth on SW26010 processor. [43] Similarly, the performance speedup for the second part of $n^{(1)}$ is also related to the number of the local points (n_{point}) since we use athread for the n_{point} loop. In addition, the computation amount inside the n_{point} loop is decided by the number of the computed basis function (n_{compute}) as there is a dot product calculation for the vectors of length n_{compute}. The second part of $n^{(1)}$ would improve the performance by 8 to 22 times with increasing the two parameters n_{point}/n_{compute} from 34/1834 to 103/1589.

In Fig. 7, the speedup for the calculation of the first order Hamiltonian ($H^{(1)}$), is shown. In the upper panel of Fig. 7, the speedup of the first part of the $H^{(1)}$ is plotted with respect to n_{point} and n_{compute}. Here the speedup is increased from 16 to 25

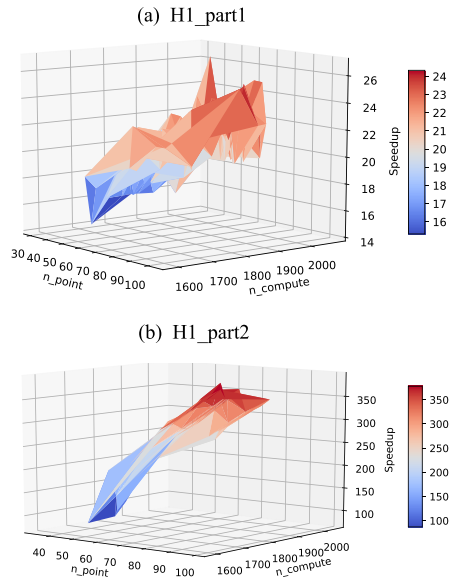

Fig. 7. The performance speedup for the first order Hamiltonian ($H^{(1)}$) within one Sunway many-core processor. The speedup is plotted with respect to n_{point} and $n_{compute}$ in every local batch. The upper panel(a) shows the speedup of the first part of the $H^{(1)}$ calculation and the lower panel (b) shows the speedup of the second part of the $H^{(1)}$ calculation.

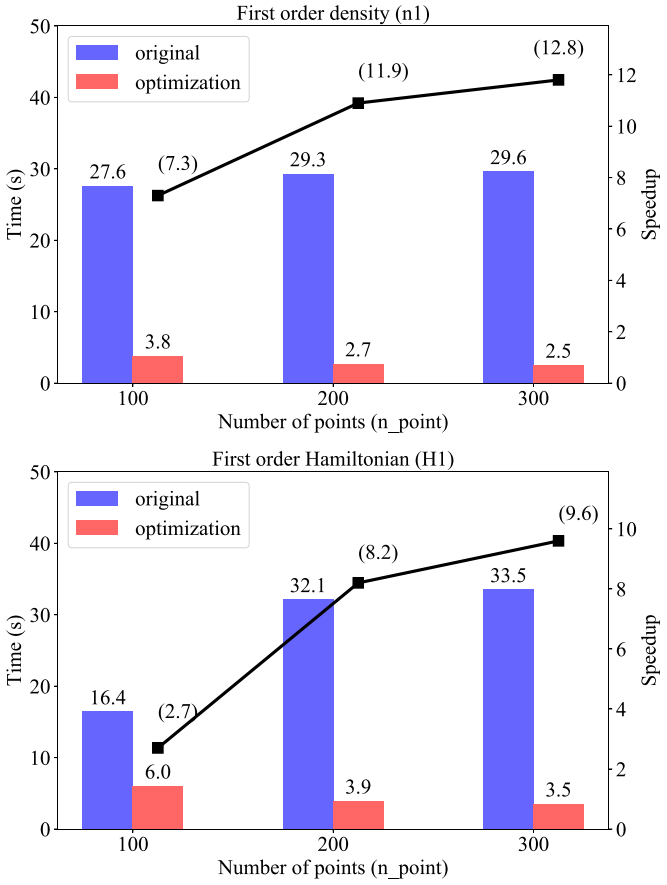

Fig. 8. The time and performance speedup for the calculation of the first order density (upper panel) and the first order Hamiltonian (lower panel) in one DFPT cycle with respect to the average number of the points in every batch. The blue bars are the original time while the red bars refer to the many-core optimized time. The speedup values are labeled with black squares and are annotated in the parentheses.

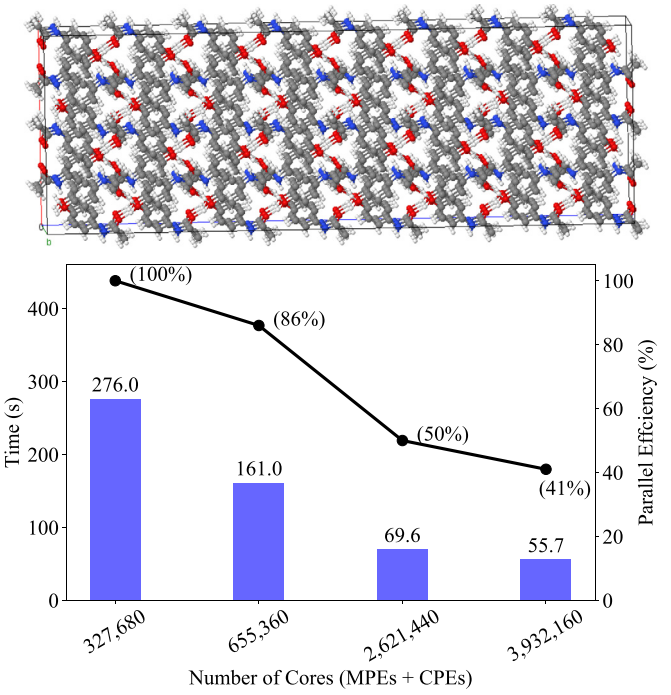

Fig. 9. The strong scalability for the computation time of the paracetamol crystal containing 3840 atoms (38208 basis). The blue bars is the DFPT time per cycle, and the black line is the parallel efficiency. Simulation time and parallel efficiency values are annotated on the top of bars and in the parentheses respectively.

with increasing the two parameters $n_{point}/n_{compute}$ from 32/1851 to 73/2043. Then in the second part, the matrix multiplication is performed for the local dense matrix which is again optimized with athread. The speedup ratio for this kernel increases from 69× to 394× with increasing the two parameters $n_{point}/n_{compute}$ from 36/1721 to 88/1752.

After evaluating the performance speedup for the kernel functions in the local batch calculation, we begin to discuss the integration calculation for the whole/global calculation of first order Hamiltonian and the first order density. In such calculations, the local matrixes need to be merged into the global matrices, and the data need to be copied between the MPE and the CPEs. As a result, the speedup is reduced compared with the individual kernel function. However, it is also clearly shown that, the speedup is increased with increasing the average number of points in every batch. Fig. 8 gives the time for the calculation of the whole first order density in one DFPT cycle with respect to the average number of the points in every batch. The speedup changes from 7.3 to 12.8 with increasing the average batch size from 100 to 300. In Fig. 8, the time for the calculation of the whole first order Hamiltonian is given, the many-core optimization improves the performance by 2.7 to 9.6 times with increasing the average batch size from 100 to 300 which demonstrates again the performance number depends on the number of points per CPE.

4.3. Scalability results

Fig. 9 shows the results of strong scalability for DFPT calculations with LDA functional, “light” basis set on the Sunway Taihu-Light supercomputer. The system used is the paracetamol crystal (form II) containing 3,840 atoms in the unit cell (38,208 basis functions). The number of the Sunway processes increases from 5,120 all the way to 61,440 for the calculation (the number of cores ranging from 327,680 to 3,932,160). The parallel efficiency is around 41% at 61,440 processes (3,932,160 cores). The efficiency is reduced

due to the batches distributed per process is not enough for the computation.

5. Conclusion

We have presented the numerical algorithms and parallel schemes of the first-principles quantum perturbation simulations within an all-electron, numeric atom-centered orbitals framework to reach a good performance over millions of cores on Sunway many-core architectures. Two levels of parallelization have been adopted to utilize the nature of the physical problems and the many-core architecture. Moreover, the linear-scaling scheme is used to achieve high performance. The scalability of the DFPT code for the molecular crystal is excellent. To the best of our knowledge, this is the first reported quantum perturbation calculation that can scale to over millions of cores. The proposed parallelization schemes and novel algorithms in this work can be generalized to a variety of other types of many-core architectures such as GPU, for example, the matrix-matrix multiplication (DGEMM) in the calculation of the first order density and the first order Hamiltonian can use the NVIDIA's cuBLAS library, which has already interfaced to the main FHI- aims code.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported by National Natural Science Foundation of China (Grant No. 22003073), CARCH 4205 and CARCH 4411. This work was also supported by the European Union's Horizon 2020 Research and Innovation Programme (Grant No. 951786), the NOMAD CoE.

References

[1] X. Gonze, Phys. Rev. B 55 (1997) 10337–10354, https://doi.org/10.1103/PhysRevB.55.10337, http://link.aps.org/doi/10.1103/PhysRevB.55.10337.

[2] S. Baroni, S. de Gironcoli, A. Dal Corso, P. Giannozzi, Rev. Mod. Phys. 73 (2001) 515–562, https://doi.org/10.1103/RevModPhys.73.515, http://link.aps.org/doi/10.1103/RevModPhys.73.515.

[3] M. Cardona, Sci. Technol. Adv. Mater. 7 (2006) S60–S66, https://doi.org/10.1016/j.stam.2006.03.009, http://stacks.iop.org/1468-6996/7/i=S1/a=A14?key=crossref.3e9584da5d8fcc9366fa1dd1c568d7a1.

[4] H. Shang, C. Carbogno, P. Rinke, M. Scheffler, Comput. Phys. Commun. 215 (2017) 26–46, https://doi.org/10.1016/j.cpc.2017.02.001, http://www.sciencedirect.com/science/article/pii/S0010465517300437, https://linkinghub.elsevier.com/retrieve/pii/S0010465517300437.

[5] P. Giannozzi, S. de Gironcoli, P. Pavone, S. Baroni, Phys. Rev. B 43 (1991) 7231–7242, https://doi.org/10.1103/PhysRevB.43.7231, http://link.aps.org/doi/10.1103/PhysRevB.43.7231.

[6] H. Shang, N. Raimbault, P. Rinke, M. Scheffler, M. Rossi, C. Carbogno, New J. Phys. 20 (2018) 073040, https://doi.org/10.1088/1367-2630/aace6d, http://stacks.iop.org/1367-2630/20/i=7/a=073040?key=crossref.b45b8680fc0308226fe0611417a68450.

[7] N. Raimbault, V. Athavale, M. Rossi, Phys. Rev. B, Condens. Matter Mater. Phys. 3 (2019) 053605, https://doi.org/10.1103/PhysRevMaterials.3.053605, https://link.aps.org/doi/10.1103/PhysRevMaterials.3.053605, arXiv:1901.10587.

[8] P. Giannozzi, S. Baroni, N. Bonini, M. Calandra, R. Car, C. Cavazzoni, D. Ceresoli, G.L. Chiarotti, M. Cococcioni, I. Dabo, A. Dal Corso, S. de Gironcoli, S. Fabris, G. Fratesi, R. Gebauer, U. Gerstmann, C. Gougoussis, A. Kokalj, M. Lazzeri, L. Martin-Samos, N. Marzari, F. Mauri, R. Mazzarello, S. Paolini, A. Pasquarello, L. Paulatto, C. Sbraccia, S. Scandolo, G. Sclauzero, A.P. Seitsonen, A. Smogunov, P. Umari, R.M. Wentzcovitch, J. Phys. Condens. Matter 21 (2009) 395502, http://www.quantum-espresso.org.

[9] G. Kresse, J. Hafner, Phys. Rev. B 47 (1993) 558–561, https://doi.org/10.1103/PhysRevB.47.558, https://link.aps.org/doi/10.1103/PhysRevB.47.558.

[10] X. Andrade, D. Strubbe, U. De Giovannini, A.H. Larsen, M.J.T. Oliveira, J. Alberdi-Rodriguez, A. Varas, I. Theophilou, N. Helbig, M.J. Verstraete, L. Stella, F. Nogueira, A. Aspuru-Guzik, A. Castro, M.A.L. Marques, A. Rubio, Phys. Chem. Chem. Phys. 17 (2015) 31371–31396, https://doi.org/10.1039/C5CP00351B.

[11] C.-K. Skylaris, P.D. Haynes, A.A. Mostofi, M.C. Payne, J. Chem. Phys. 122 (2005) 084119, https://doi.org/10.1063/1.1839852.

[12] D.R. Bowler, R. Choudhury, M.J. Gillan, T. Miyazaki, Phys. Status Solidi B, Basic Solid State Phys. 243 (2006) 989–1000, https://doi.org/10.1002/pssb.200541386, https://onlinelibrary.wiley.com/doi/pdf/10.1002/pssb.200541386.

[13] S. Das, P. Motamarri, V. Gavini, B. Turcksin, Y.W. Li, B. Leback, in: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC '19, Association for Computing Machinery, New York, NY, USA, 2019.

[14] S. Mohr, L.E. Ratcliff, P. Boulanger, L. Genovese, D. Caliste, T. Deutsch, S. Goedecker, J. Chem. Phys. 140 (2014) 204110, https://doi.org/10.1063/1.4871876, http://www.ncbi.nlm.nih.gov/pubmed/24880269.

[15] K. Lejaeghere, G. Bihlmayer, T. Bjorkman, P. Blaha, S. Blugel, V. Blum, D. Caliste, I.E. Castelli, S.J. Clark, A. Dal Corso, S. de Gironcoli, T. Deutsch, J.K. Dewhurst, I. Di Marco, C. Draxl, M. Du ak, O. Eriksson, J.A. Flores-Livas, K.F. Garrity, L. Genovese, P. Giannozzi, M. Giantomassi, S. Goedecker, X. Gonze, O. Granas, E.K.U. Gross, A. Gulans, F. Gygi, D.R. Hamann, P.J. Hasnip, N.A.W. Holzwarth, D. Iu an, D.B. Jochym, F. Jollet, D. Jones, G. Kresse, K. Koepernik, E. Kucukbenli, Y.O. Kvashnin, I.L.M. Locht, S. Lubeck, M. Marsman, N. Marzari, U. Nitzsche, L. Nordstrom, T. Ozaki, L. Paulatto, C.J. Pickard, W. Poelmans, M.I.J. Probert, K. Refson, M. Richter, G.-M. Rignanese, S. Saha, M. Scheffler, M. Schlipf, K. Schwarz, S. Sharma, F. Tavazza, P. Thunstrom, A. Tkatchenko, M. Torrent, D. Vanderbilt, M.J. van Setten, V. Van Speybroeck, J.M. Wills, J.R. Yates, G.-X. Zhang, S. Cottenier, Science 351 (2016) aad3000, https://doi.org/10.1126/science.aad3000, http://science.sciencemag.org/content/351/6280/aad3000.abstract.

[16] G.K.H. Madsen, P. Blaha, K. Schwarz, E. Sjöstedt, L. Nordström, Phys. Rev. B 64 (2001) 195134, https://doi.org/10.1103/PhysRevB.64.195134, https://link.aps.org/doi/10.1103/PhysRevB.64.195134.

[17] M. Methfessel, C.O. Rodriguez, O.K. Andersen, Phys. Rev. B 40 (1989) 2009–2012, https://doi.org/10.1103/PhysRevB.40.2009, https://link.aps.org/doi/10.1103/PhysRevB.40.2009.

[18] L. Maschio, B. Kirtman, R. Orlando, M. Rèrat, J. Chem. Phys. 137 (2012) 204113, https://doi.org/10.1063/1.4767438, http://www.ncbi.nlm.nih.gov/pubmed/23205987.

[19] B. Delley, J. Chem. Phys. 92 (1990) 508, https://doi.org/10.1063/1.458452, http://scitation.aip.org/content/aip/journal/jcp/92/1/10.1063/1.458452.

[20] V. Blum, R. Gehrke, F. Hanke, P. Havu, V. Havu, X. Ren, K. Reuter, M. Scheffler, Comput. Phys. Commun. 180 (2009) 2175–2196, https://doi.org/10.1016/j.cpc.2009.06.022, http://linkinghub.elsevier.com/retrieve/pii/S0010465509002033.

[21] J.A. Pople, R. Krishnan, H.B. Schlegel, J.S. Binkley, Int. J. Quant. Chem. 16 (1979) 225–241, https://doi.org/10.1002/qua.560160825.

[22] M. Frisch, M. Head-Gordon, J. Pople, Chem. Phys. 141 (1990) 189–196, https://doi.org/10.1016/0301-0104(90)87055-G, http://www.sciencedirect.com/science/article/pii/030101049087055G.

[23] A.F. Izmaylov, G.E. Scuseria, J. Chem. Phys. 127 (2007) 144106, https://doi.org/10.1063/1.2790024, http://www.ncbi.nlm.nih.gov/pubmed/17935385.

[24] S. Savrasov, D. Savrasov, Phys. Rev. B 54 (1996) 16487–16501, http://www.ncbi.nlm.nih.gov/pubmed/9985772.

[25] R. Yu, H. Krakauer, Phys. Rev. B 49 (1994) 4467–4477, https://doi.org/10.1103/PhysRevB.49.4467, http://link.aps.org/doi/10.1103/PhysRevB.49.4467.

[26] R. Kouba, A. Taga, C. Ambrosch-Draxl, L. Nordström, B. Johansson, Phys. Rev. B 64 (2001) 184306, https://doi.org/10.1103/PhysRevB.64.184306, http://link.aps.org/doi/10.1103/PhysRevB.64.184306.

[27] X. Ren, P. Rinke, V. Blum, J. Wieferink, A. Tkatchenko, A. Sanfilippo, K. Reuter, M. Scheffler, New J. Phys. 14 (2012) 053020, https://doi.org/10.1088/1367-2630/14/5/053020, http://stacks.iop.org/1367-2630/14/i=5/a=053020?key=crossref.351b343783c2c1df1596219a941a74eb.

[28] A.D. Becke, J. Chem. Phys. 88 (1988) 2547–2553, https://doi.org/10.1063/1.454033, http://scitation.aip.org/content/aip/journal/jcp/88/4/10.1063/1.454033.

[29] B. Delley, J. Chem. Phys. 94 (1991) 7245, https://doi.org/10.1063/1.460208, http://scitation.aip.org/content/aip/journal/jcp/94/11/10.1063/1.460208.

[30] B. Delley, J. Chem. Phys. 92 (1990) 508, https://doi.org/10.1063/1.458452, http://scitation.aip.org/content/aip/journal/jcp/92/1/10.1063/1.458452.

[31] R.M. Sternheimer, Phys. Rev. 96 (1954) 951–968, https://doi.org/10.1103/PhysRev.96.951, http://link.aps.org/doi/10.1103/PhysRev.96.951.

[32] X. Gonze, C. Lee, Phys. Rev. B 55 (1997) 10355–10368, https://doi.org/10.1103/PhysRevB.55.10355, http://link.aps.org/doi/10.1103/PhysRevB.55.10355.

[33] P. Pulay, Chem. Phys. Lett. 73 (1980) 393–398, https://doi.org/10.1016/0009-2614(80)80396-4, http://www.sciencedirect.com/science/article/pii/0009261480803964.

[34] J. Baker, J. Andzelm, A. Scheiner, B. Delley, J. Chem. Phys. 101 (1994) 8894–8902, https://doi.org/10.1063/1.468081, http://link.aip.org/link/?JCP/101/8894/1.

[35] B. Delley, J. Comput. Chem. 17 (1996) 1152–1155, https://doi.org/10.1002/(SICI)1096-987X(19960715)17:9<1152::AID-JCC7>3.0.CO;2-R.

[36] V. Havu, V. Blum, P. Havu, M. Scheffler, J. Comput. Phys. 228 (2009) 8367–8379, https://doi.org/10.1016/j.jcp.2009.08.008, http://linkinghub.elsevier.com/retrieve/pii/S0021999109004458.
[37] A.M.N. Niklasson, Phys. Rev. B 66 (2002) 155115, https://doi.org/10.1103/PhysRevB.66.155115, https://link-aps-org-443.webvpn.las.ac.cn/doi/10.1103/PhysRevB.66.155115.
[38] V. Weber, A.M.N. Niklasson, M. Challacombe, Phys. Rev. Lett. 92 (2004) 193002, https://doi.org/10.1103/PhysRevLett.92.193002; https://link.aps.org/doi/10.1103/PhysRevLett.92.193002, http://arxiv.org/abs/0312634.
[39] A. Azad, G. Ballard, A. Buluç, J. Demmel, L. Grigori, O. Schwartz, S. Toledo, S. Williams, SIAM J. Sci. Comput. 38 (2016) C624–C651, https://doi.org/10.1137/15M104253X, http://www.siam.org/journals/sisc/38-6/M104253.html.
[40] W. Dawson, T. Nakajima, Comput. Phys. Commun. 225 (2018) 154–165, https://doi.org/10.1016/j.cpc.2017.12.010, https://linkinghub.elsevier.com/retrieve/pii/S0010465517304150.
[41] X. Duan, P. Gao, T. Zhang, M. Zhang, W. Liu, W. Zhang, W. Xue, H. Fu, L. Gan, D. Chen, X. Meng, G. Yang, in: Proceedings - International Conference for High Performance Computing, Networking, Storage, and Analysis, SC 2018, 2018, pp. 148–159.
[42] K. Goto, R.A. van de Geijn, ACM Trans. Math. Softw. 34 (2008), https://doi.org/10.1145/1356052.1356053.
[43] L. Jiang, C. Yang, Y. Ao, W. Yin, W. Ma, Q. Sun, F. Liu, R. Lin, P. Zhang, in: 2017 46th International Conference on Parallel Processing, ICPP, 2017, pp. 422–431.
[44] J. Perdew, A. Zunger, Phys. Rev. B 23 (1981) 5048.