

# Large Scale Finite-Temperature Real-time Time Dependent Density Functional Theory Calculation with Hybrid Functional on ARM and GPU Systems

Rongrong Liu

*State Key Lab of Processors,  
Institute of Computing Technology,  
Chinese Academy of Sciences,  
University of Chinese Academy  
of Sciences, Beijing, China  
liurongrong21s@ict.ac.cn*

Zhuoqiang Guo

*State Key Lab of Processors,  
Institute of Computing Technology,  
Chinese Academy of Sciences,  
University of Chinese Academy  
of Sciences, Beijing, China  
guozhuoqiang20z@ict.ac.cn*

Qiuchen Sha

*State Key Lab of Processors,  
Institute of Computing Technology,  
Chinese Academy of Sciences,  
University of Chinese Academy  
of Sciences, Beijing, China  
shaqiuchen22s@ict.ac.cn*

Tong Zhao

*State Key Lab of Processors,  
Institute of Computing Technology,  
Chinese Academy of Sciences,  
University of Chinese Academy  
of Sciences, Beijing, China  
zhaotong@ict.ac.cn*

Haibo Li

*School of Mathematics and  
Statistics, The University of  
Melbourne, Melbourne, Australia  
haibo.li@unimelb.edu.au*

WeiHu

*School of Computer Science  
and Technology, University of  
Science and Technology of  
China, Hefei, China  
whuustc@ustc.edu.cn*

Lijun Liu

*Department of Mechanical Engineering,  
Graduate School of Engineering,  
Osaka University,  
Osaka, Japan  
liu@mech.eng.osaka-u.ac.jp*

Guangming Tan

*State Key Lab of Processors,  
Institute of Computing Technology,  
Chinese Academy of Sciences,  
University of Chinese Academy  
of Sciences, Beijing, China  
tgm@ict.ac.cn*

Weile Jia

*State Key Lab of Processors,  
Institute of Computing Technology,  
Chinese Academy of Sciences,  
University of Chinese Academy  
of Sciences, Beijing, China  
jiaweile@ict.ac.cn*

**Abstract**—Ultra-fast electronic phenomena originating from finite temperature, such as nonlinear optical excitation, can be simulated with high fidelity via real-time time dependent density functional theory (rt-TDDFT) calculations with hybrid functional. However, previous rt-TDDFT simulations of real materials using the optimal gauge-known as the parallel transport gauge-have been limited to low-temperature systems with band gaps. In this paper, we introduce the parallel transport-implicit midpoint (PT-IM) method, which significantly accelerates finite-temperature rt-TDDFT calculations of real materials with hybrid function. We first implement PT-IM with hybrid functional in our plane wave code PWFT, and optimized it on both GPU and ARM platforms to build a solid baseline code. Next, we propose a diagonalization method to reduce computation and communication complexity, and then, we employ adaptively compressed exchange (ACE) method to reduce the frequency of the most expensive Fock exchange operator. Finally, we adopt the ring based method and the shared memory mechanism to overlap computation and communication and alleviate memory consumption respectively. Numerical results show that our optimized code can reach 3072 atoms for rt-TDDFT simulation with hybrid functional at finite temperature on 192 computing nodes, the time-to-solution for one time step is 429.3s, which is 41.4 times faster compared to the baseline.

**Index Terms**—rt-TDDFT, High-performance computing

## I. INTRODUCTION

Real-time time-dependent density functional theory (rt-TDDFT) [1]–[5] is a widely used approach in electronic excitation calculations, gaining research attention with the growing experimental focus on ultrafast electronic phenomena in materials science. It can be used in a spectrum of applications, including ion collisions [6], the light absorption spectrum [7], laser-induced demagnetization and phase transitions [8], charge transfer, dynamics of excited carriers, and chemical reactions [9]. Recent studies [10]–[16] have illuminated the capacity of laser excitation to initiate structural phase transitions and charge density wave excitations, along with revealing that many interactions in catalysis, previously assumed to be adiabatic, actually proceed via non-adiabatic mechanisms, necessitating electron excitation simulations for accurate analysis. These developments mark a significant shift in materials science simulations, pushing the boundaries beyond conventional ground state DFT calculation.

However, rt-TDDFT simulations of real materials still face two challenges: first, rt-TDDFT calculations are constrained by

the precision and stability requisites of ordinary differential equation (ODE) integrators, restricting the time step to the sub-attosecond domain to ensure the accuracy of dynamics. Consequently, explicit time integrators, notably the fourth-order Runge-Kutta (RK4) method, are frequently favored over implicit methods like the Crank-Nicolson (CN) scheme, owing to their operational efficiency and ease of implementation. The parallel transport gauge formalism can find the slowest oscillating orbitals, allowing for the effective utilization of implicit integrators with significantly larger step sizes. The parallel transport Crank-Nicolson (PT-CN) scheme, in particular, has been shown to extend the feasible time step to approximately 50 attoseconds while maintaining accuracy comparable to the RK4 method [17]. However, the current PT-CN scheme is only applicable for systems with band gaps, which means PT-CN cannot be applied to metallic or finite-temperature systems where electrons are fractionally occupied. Secondly, semi-local exchange-correlation functionals, like the Local Density Approximation (LDA) [18], [19] and Generalized Gradient Approximation (GGA) [20], fall short in accurately describing excited states and band gaps, which can lead to incorrect behavior such as the emergence of nonphysical exciton in rt-TDDFT calculations. Hybrid functionals [21], [22] within Density Functional Theory (DFT) offer a solution by mixing a portion of the Fock exchange integral with semi-local functionals, improving accuracy for electronic structures. Particularly, range-separated hybrid functionals [23], [24] have shown to match optical absorption spectra accurately, comparable to results from more demanding methods like the Bethe-Salpeter equation based on GW calculations [25]. Thus, combining rt-TDDFT with hybrid functionals represents a powerful approach for accurately modeling exciton excitation and charge transfer, marking a notable advance in materials science simulations.

Unfortunately, both finite-temperature rt-TDDFT and hybrid functional require significantly higher computational costs compared to traditional ground state DFT calculations with semi-local functionals. Specifically, rt-TDDFT can be orders of magnitude slower than conventional ground state molecular dynamics simulations, attributed to its smaller time steps. Similarly, hybrid functionals are substantially slower than semi-local exchange-correlation functionals due to the evaluation of the Fock exchange term. Consequently, literature on plane wave-based real material rt-TDDFT simulation with hybrid functional, even for modest systems comprising a few atoms, is scarce, let alone for larger systems containing thousands of atoms [26], [27]. Nonetheless, for a multitude of applications, such as excited state charge transfer and laser-induced structural phase transitions in nanostructures like quantum dots and wires, simulating large systems is indispensable. Moreover, complete basis sets such as plane waves are more favorable in describing excited states in rt-TDDFT calculations. All these considerations pose challenges to rt-TDDFT calculations of real materials at finite temperatures.

The latest developments in high-performance computing and the parallel transport gauge formalism have offers opportu-

nities for overcoming these challenges, both from algorithmic and hardware perspectives. Recently, An et al. proposed a parallel transport formalism for rt-TDDFT at finite temperatures [28], demonstrating its potential of extending the time step length significantly beyond the limitations of the Runge-Kutta 4th order (RK4) method in a one-dimension problem. On the hardware side, with the improvement of architecture and manufacturing processes, the computing performance has been enhanced at a faster pace compared to that of memory bandwidth and network bandwidth, resulting in an increasingly wider gap between the former and the latter two. This raises a highly intriguing question: can hybrid functional rt-TDDFT calculations be accelerated on many-core systems, such as ARM or GPU platforms?

In this paper, we present our highly scalable and efficient implementation of finite-temperature rt-TDDFT calculation with hybrid functionals, and optimized for both ARM and GPU platforms using the planewave code PWDFT. Our major contributions are as follows:

- We implement finite-temperature rt-TDDFT algorithm PT-IM with hybrid functional for 3D real materials and optimized it on ARM and GPU platforms using OpenMP and CUDA, respectively.
- We further proposed a matrix diagonalization method to reduce the computational complexity from  $O(N^4)$  to  $O(N^3)$  and significantly decreased the frequency of the most expensive hybrid functional calculations using the Adaptively Compressed Exchange (ACE) method.
- Additionally, we proposed an Asynchronous ring-based method and utilized a shared memory mechanism to optimize the network communication and reduce memory consumption.
- Testing results of a 384-atom silicon system show that compared to the baseline, our optimized code achieves a speedup of 55.15 and 41.44 times on ARM and GPU platforms, respectively. Our optimized code can also scale up to 960 nodes on Fugaku (46080 ARM cores) to simulate a silicon system of 1536 atoms and can reach 3072 atoms (12288 electrons) on 768 A100 GPUs.

This paper is organized as follows: we review the rt-TDDFT algorithm PT-IM in Sec. II. Then a baseline implementation of PT-IM with hybrid functional is shown in Sec. III. We further optimize the PT-IM in Sec. IV. Machine configuration and physical systems are listed in Sec. V and Sec. VI. The physical and performance results are shown in Sec. VII and Sec. VIII, respectively. The conclusion is drawn in Sec. IX.

## II. BACKGROUND

### A. The parallel transport-implicit midpoint (PT-IM) method

Real-time time-dependent density functional theory solves the following time-dependent equation:

$$i\partial_t\Psi(t) = H(t, P(t))\Psi(t). \quad (1)$$

Here  $\Psi(t) = [\psi_1(t), \dots, \psi_N(t)]$  is the collection of electron wavefunctions (also called electron orbitals), and  $N$  is the

number of total electron states (spin degeneracy omitted).  $P(t)$  is the density matrix, which defined as  $P(t) = \Psi(t)\sigma(t)\Psi^*(t)$  [28].  $\Psi^*$  is the Hermitian conjugate of  $\Psi$  and  $\sigma(t)$  is the occupation number matrix.

In pure states (low temperature),  $\sigma(t) = \sigma(0) = I_N$ . So  $P(r, r') = \sum_i^N \psi_i(r)\sigma_i\psi_i^*(r')$ , and  $\sigma_i$  is either one (occupied) or zero (unoccupied). In real material rt-TDDFT simulation, the initial state  $\sigma(0)$  is required to be a mixed state. For instance, for metallic systems or semiconductors at finite temperatures, the wavefunctions can be fractionally occupied by the Fermi-Dirac distribution. In such mixed states,

$$P(r, r') = \sum_{i,j=1}^N \psi_i(r)\sigma_{ij}\psi_j^*(r'). \quad (2)$$

The corresponding rt-TDDFT equation 1 can be equivalently reformulated using a series of unitarily transformed orbitals. Physical observables, including the density matrix, remain unchanged under such unitary transformations, a property known as gauge invariance. This invariance enables the pursuit of an optimal gauge. Recent advancements have pinpointed such an optimal gauge [28], implicitly defined by the subsequent equation:

$$\begin{aligned} i\partial_t\Phi(t) &= (I - \tilde{P}(t))H(t, P(t))\Phi(t), \\ i\partial_t\sigma(t) &= [(\Phi^*(t)H(t, P(t))\Phi(t), \sigma(t)], \end{aligned} \quad (3)$$

where  $\Phi(t)$  oscillates much slower by choosing the optimal gauge ( $\Phi(t) = \Psi(t)U(t)$ ). Coupled with the implicit midpoint (IM) rule (also known as the Gauss-Legendre method of order 2), the shorthand notations are introduced:

$$\Phi_{n+\frac{1}{2}} = \frac{\Phi_{n+1} + \Phi_n}{2}, \sigma_{n+\frac{1}{2}} = \frac{\sigma_{n+1} + \sigma_n}{2}, \quad (4)$$

and accordingly,

$$\begin{aligned} \tilde{P}_{n+\frac{1}{2}} &= \Phi_{n+\frac{1}{2}}(\Phi_{n+\frac{1}{2}}^* \Phi_{n+\frac{1}{2}})^{-1} \Phi_{n+\frac{1}{2}}^*, \\ P_{n+\frac{1}{2}} &= \Phi_{n+\frac{1}{2}} \sigma_{n+\frac{1}{2}} \Phi_{n+\frac{1}{2}}^*, \\ H_{n+\frac{1}{2}} &= H(t_{n+\frac{1}{2}}, P_{n+\frac{1}{2}}), \end{aligned} \quad (5)$$

the parallel transport-implicit midpoint scheme (PT-IM) at each time step reads:

$$\begin{aligned} \Phi_{n+1} &= \Phi_n - i\Delta_t(I - \tilde{P}_{n+\frac{1}{2}})H_{n+\frac{1}{2}}\Phi_{n+\frac{1}{2}}, \\ \sigma_{n+1} &= \sigma_n - i\Delta_t[(\Phi_{n+\frac{1}{2}}^* H_{n+\frac{1}{2}} \Phi_{n+\frac{1}{2}}), \sigma_{n+\frac{1}{2}}]. \end{aligned} \quad (6)$$

If  $\{\Phi_{n+1}; \sigma_{n+1}\}$  is chosen to be the unknowns, then equation 3 can be viewed as a fixed point equation in the abstract form

$$x = T(x). \quad (7)$$

### B. Fock exchange operator

The Hamiltonian has the following operators when hybrid functionals are used:

$$H[P] = -\frac{1}{2}\Delta + V_{ext}(t) + V_{Hxc}[P(t)] + \alpha V_x[P(t)]. \quad (8)$$

Here  $V_{ext}(t)$  is the time-dependent external potential and  $V_{Hxc}$  consists of the Hartree potential and the local part

of the exchange-correlation potential. Without the term  $V_x$ , the functional is considered semilocal. This paper focuses on hybrid functional rt-TDDFT calculations, where  $V_x$ , called the Fock exchange operator, is an integral operator with kernel  $V_x[P](r, r') = -P(r, r')K(r, r')$ . In this context,  $K(r, r')$  denotes the kernel for the (possibly screened) electron interaction, and  $\alpha$  represents a mixing fraction (usually  $\alpha = 0.25$ ).

In hybrid functional calculations, in pure states, each set of multiplications  $V_x[P]\Phi$  requires the following operations:

$$(V_x[P]\phi_j)(r) = -\sum_{i=1}^N \phi_i(r)\sigma_i \int K(r, r')\phi_i^*(r')\phi_j(r')dr'. \quad (9)$$

In mixed states at finite temperatures, with  $P(r, r')$  in the form(2), each set of multiplications  $V_x[P]\Phi$  takes the following form:

$$(V_x[P])\phi_j(r) = -\sum_{i,k=1}^N \sigma_{ik}\phi_i(r) \int K(r, r')\phi_k^*(r')\phi_j(r')dr'. \quad (10)$$

In planewave basis, for  $V_x$  applied to a single orbital, in pure states, it can be calculated via solving  $N^2$  Poisson-type equations. In mixed states, however, it amounts to solving  $N^3$  Poisson equations. And then we need to do this for all  $N$  orbitals to obtain  $V_x\Phi$ . If we denote the number of discrete lattice points in real space by  $N_g$ , in pure states, the total cost of  $V_x\Phi$  is  $O(N_g \log N_g N^2) \sim O(N^3)$  and in mixed states the cost is  $O(N_g \log N_g N^3) \sim O(N^4)$ . Notably, in pure states, the time taken by the Fock exchange operator already accounts for over 95% of the total time, meaning that, for the same quantum system, the solution time of hybrid functional DFT is more than 20 times that of semi-local functional DFT [26]. In mixed states, the computational complexity of the Fock exchange operator increases by an order. This implies that performing rt-TDDFT with hybrid functionals on large systems at finite temperatures is prohibitively expensive.

### III. BASELINE IMPLEMENTATION OF PT-IM WITH HYBRID FUNCTIONAL ON GPU AND ARM PLATFORMS

Since there is no prior plane-wave implementation of the PT-IM method, the primary task of this paper is to develop an efficient baseline version of PT-IM. This section details our approach to implementing an efficient PT-IM method on GPU and ARM platforms using MPI combined with OpenMP/CUDA in the PWDFT package [26]. Note that OpenMP and GPU acceleration have been adopted for ground state electronic structure calculations in several software packages, including ABINIT [29], PWmat [30], [31], Quantum ESPRESSO [32], VASP [33], BigDFT [34], NWChem [35]. To achieve a better acceleration, our optimization efforts extended beyond the computationally expensive Fock exchange operator to include acceleration of additional components, such as residual calculations and wavefunction mixing.

Alg.1 outlines a single time propagation step of the PT-IM method. First, the initial values of  $\Phi_{n+1}$  and  $\sigma_{n+1}$  are

---

**Algorithm 1:** One time propagation step for PT-IM method with hybrid functional

---

**Input:**  $\Phi_n$  and  $\sigma_n$

**Output:**  $\Phi_{n+1}$  and  $\sigma_{n+1}$

- 1 Suppose  $\{\Phi_{n+1}, \sigma_{n+1}\} = T(\{\Phi_n, \sigma_n\})$ ;
  - 2 Calculate  $\rho_{n+1}^{in}$  from  $\Phi_{n+1}$  and  $\sigma_{n+1}$ ;
  - 3 **for**  $k = 1, 2, \dots$  **do**
  - 4   Calculate  $\Phi_{n+\frac{1}{2}}$  and  $\sigma_{n+\frac{1}{2}}$  refer to (4);
  - 5   Calculate  $\rho_{n+\frac{1}{2}}$  from  $\Phi_{n+\frac{1}{2}}$  and  $\sigma_{n+\frac{1}{2}}$ ;
  - 6   Update  $H_{n+\frac{1}{2}}$ ;
  - 7   Update  $\{\Phi_{n+1}, \sigma_{n+1}\}$  refer to (6);
  - 8   Update  $\Phi_{n+1}$  and  $\sigma_{n+1}$  by Anderson mixing;
  - 9   Evaluate the residual  $R_f$  of (6);
  - 10   Calculate  $\rho_{n+1}^{out}$  from  $\Phi_{n+1}$  and  $\sigma_{n+1}$ ;
  - 11   Jump out of the loop when the density change is sufficiently small;
  - 12 **end**
  - 13 Orthogonalize  $\Phi_{n+1}$  and conjugate symmetrize  $\sigma_{n+1}$ ;
- 

evaluated to obtain the intermediate wavefunctions  $\Phi_{n+\frac{1}{2}}$  and occupation number matrix  $\sigma_{n+\frac{1}{2}}$ . Next, we calculate the physical quantities at these intermediate moments:  $\rho_{n+\frac{1}{2}}$  and  $H_{n+\frac{1}{2}}$ , which are ultimately used to update the new  $\{\Phi_{n+1}, \sigma_{n+1}\}$  refer to (6), involving the calculation of the Fock exchange operator. Anderson mixing [36] of the wavefunctions and charge density are employed to accelerate the convergence of the fixed-point problem. When the residual of  $\rho$  is sufficiently small, the SCF iteration can be terminated. In practice, we find that the SCF convergence can also be controlled by the convergence of the charge density. In Alg.1, the most time-consuming part is the Fock exchange operator. Therefore, we will focus on optimizing it in the following. Other important computation modules include electron density, residual, and Anderson mixing.

#### A. Data distribution

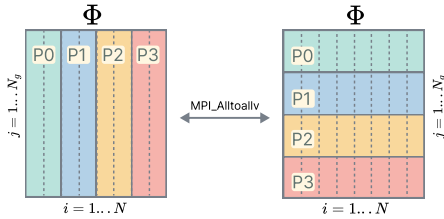


Fig. 1. The parallel distribution of wavefunction  $\Phi$  (left: band-index parallelization; right: grid-point parallelization). Note that  $\text{MPI\_Alltoallv}$  is required to transpose between the two parallelization schemes.

Fig. 1 shows the two primary parallelization schemes used in PWDF. First, the wavefunction  $\Phi$  can be distributed over the columns (band-index parallelization as shown in Fig. 1 left). This is particularly efficient for the calculation of  $H\Phi$  and hybrid functional since different MPI tasks can perform fast Fourier transformations (FFT) independently. The second

parallelization scheme is to distribute the wavefunction  $\Phi$  over the rows (grid-point parallelization as shown in Fig. 1 right,  $G$  is the grid in Fourier space). grid-point parallelization is efficient for the calculation of the overlap matrix  $S = \Phi^* H \Phi$  over matrix-matrix multiplication. Note that  $\text{MPI\_Alltoall}$  is required to transpose between these two parallelization schemes. And since we focus on large systems with more than a few hundred atoms, only one  $\Gamma$  point is needed. Therefore, K-point parallelization is omitted in this paper.

#### B. Evaluation of the Fock exchange operator

---

**Algorithm 2:** The Fock exchange operator calculation in mixed states

---

**Input:**  $\Phi$  and  $\sigma$

**Output:**  $V_x \Phi$

- 1 Let  $V_x \Phi$  be distributed by band-index parallelization and initialized to zero and  $\phi_{temp}$  is a temp variable;
  - 2 **for**  $k = 1, N$  **do**
  - 3   **if** the current process holds  $\phi_k$  **then**
  - 4     Broadcast  $\phi_k$  to all processes;
  - 5   **end**
  - 6   **for**  $i = 1, N$  **do**
  - 7     **if** the current process holds  $\phi_i$  **then**
  - 8       Broadcast  $\phi_i$  to all processes;
  - 9     **end**
  - 10    **for**  $j = 1, N$  **do**
  - 11     **if** the current process holds  $\phi_j$  **then**
  - 12        $\phi_{temp} = \phi_k^* \odot \phi_j$ ;
  - 13        $\phi_{temp} = \text{inplace forward FFT}(\phi_{temp})$ ;
  - 14        $\phi_{temp} = K(r, r') \phi_{temp}$ ;
  - 15        $\phi_{temp} = \text{inplace inverse FFT}(\phi_{temp})$ ;
  - 16        $V_x \phi_j = V_x \phi_j + \sigma_{ik} \phi_{temp} \odot \phi_i$ ;
  - 17     **end**
  - 18    **end**
  - 19 **end**
- 

In PT-IM implementation, the evaluation of the Fock exchange operator is the most time-consuming part and is repeatedly performed within the matrix-vector multiplication  $H\Phi$ . Alg. 2 details the evaluation of the Fock exchange operator ( $V_x[P]\Phi$ ) in mixed-state rt-TDDFT calculation. As discussed earlier, wavefunction  $\Phi$  is distributed in band-index parallelization to efficiently perform FFTs. Each wavefunction  $\phi_k$  has to be broadcast to all MPI tasks via  $\text{MPI\_Bcast}$ . Then the Fock exchange operator is evaluated as shown in Equ. 10. Due to the introduction of the occupation matrix  $\sigma_{i,k}$ , a triple loop is needed in calculating the Fock exchange operator among wavefunctions  $\phi_i$ ,  $\phi_j$  and  $\phi_k$ . This requires a total number of  $N^3$  FFTs, leading to a computational complexity of  $O(N^3 N_g \log N_g)$ , where  $N_g$  is the number of grid points and  $N$  is the number of electrons. This computational complexity is higher than that in zero-temperature rt-TDDFT or ground-state calculations, which requires only  $N^2$  FFTs since

the occupation matrix  $\sigma$  is diagonal, and only two-electron interaction between  $\phi_i$  and  $\phi_j$  are evaluated (as shown in Eq. 9).

In our baseline implementation, we take the following steps to optimize the calculation of the Fock exchange operator.

**(a) Band-by-band implementation.** First, we implement the Fock exchange operator in a band-by-band manner, and CUFFT and FFTW are utilized on GPU and ARM platforms. The gaps between the FFT invoke are filled via CUDA customized kernels or OpenMP accelerated computations. Note that no CPU-GPU synchronization is during the calculation.

**(b) Multi-batch implementation.** For GPU platform, we further utilize a multi-batch strategy to enhance its bandwidth utilization. Each A100 GPU has a bandwidth of 1.5 TB/s and the band-by-band implementation cannot fully exploit the hardware limit. To improve the performance, instead of sending the data  $\Phi^*\Phi$  one by one, we perform multi-batch operations in customized CUDA kernels, cuFFT, FFT, and MPI\_Bcast to fully saturate the memory and network bandwidth. Especially, the multi-batch implementation can also reduce the latency of CPU-GPU kernel launch. The batch size is set to 16. We find that the multi-batch implementation can greatly improve the performance of the Fock exchange operator compared to the band-by-band implementation.

### C. Other calculations

As Amdahl's law indicates, all calculations have to be optimized to achieve a desirable speedup. Thus in our PWDFT implementation, we have moved almost all calculations to the GPU and ARM cores besides the computationally intensive Fock exchange operator.

**1. Charge density evaluation.** The charge density  $\rho$  is calculated via  $\sum_{i,j=1}^N \phi_i \sigma_{ij} \phi_j^*$  in the PT-IM method. The introduction of occupation matrix  $\sigma_{i,j}$  has increased the computational complexity of charge density calculation from  $O(N^2 \log N_g)$  (ground state) to  $O(N^3 \log N_g)$  due to the interaction for each  $i, j$  pair of the wavefunctions. Note that the wavefunctions  $\Phi$  have to be communicated across all MPI tasks via MPI\_Bcast due to the parallel distribution as detailed in Sec. III-A, and all calculations are evaluated either via efficient libraries such as CUFFT/FFTW or hand written CUDA kernels/OpenMP.

**2. Anderson mixing and orthogonalization.** The Anderson mixing in PT-IM solves the least square problem for each wavefunction and  $\sigma$ . Note that the least square problem can be very small ( $20 \times 20$  in our implementation), thus the main computation is the evaluation of the overlap matrix that can be efficiently calculated via grid-point parallelization. Our implementation, requires 20 copies of the wavefunctions, which can cost lots of HBM if stored in the GPU. Thus in our GPU implementation, all wavefunctions are stored on the CPU to save GPU memory footprint and then copied to GPU for matrix-matrix multiplication to obtain the overlap matrix. The orthogonalization step is also accelerated via calling efficient libraries and hand-optimized kernels.

In summary, we develop a solid baseline version of PT-IM within the PWDFT package by incorporating the optimizations described above.

## IV. FURTHER OPTIMIZATIONS

A solid baseline version of PT-IM is implemented within the PWDFT package on the GPU and ARM platform, as detailed in Sec. III. However, despite our efforts in optimizing almost all calculations with multi-threaded parallelism and GPU, our baseline code still encounters several challenges: Firstly, it suffers from surging computation and communication complexity introduced by the occupation matrix  $\sigma$ , as delineated in (2)(10). Particularly, the number of FFTs in the evaluation of the Fock exchange operator grows from  $N^2$  (ground state) to  $N^3$  (PT-IM). Since the Fock exchange operator is required in each  $H\Phi$  calculation, we will have to calculate  $25 V_x \Phi$  in each time step on average. Moreover, the communications cost can be optimized via computation-communication overlap and asynchronous communication. In this section, we will introduce how to address the challenges above through step-by-step optimizations.

### A. Algorithm innovation

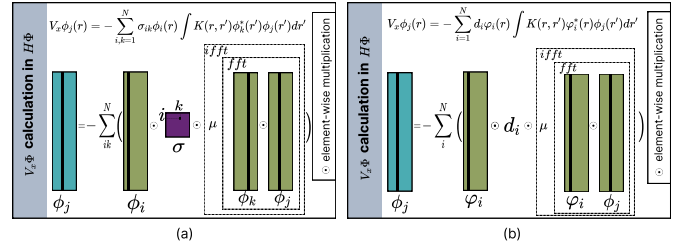


Fig. 2. Evaluation of the Fock exchange operator. (a) Baseline. (b) Accelerated by diagonalization.

#### 1) Reduce the complexity of Fock exchange operator and density calculation by occupation matrix diagonalization:

In PT-IM, the occupation number matrix  $\sigma$  introduces extra computation into the evaluation of charge density and Fock exchange operator. For example, in PT-IM the Fock exchange operator evaluation requires  $10^9$  FFTs for a physical system with  $10^3$  orbitals. The extensive computational cost hinders the time-to-solution of PT-IM to more than 30 minutes per time step for a physical system of 384 atoms. One key observation is that  $\sigma$  is a Hermitian matrix, whose eigenvectors are orthogonal to each other. Hence, we can diagonalize it:

$$\sigma_t = Q D Q^*. \quad (11)$$

Here  $D$  is a diagonal matrix with diagonal elements  $d_1, d_2, \dots, d_N$ . We can set  $\varphi = \Phi Q$ , so density matrix can be written as:

$$P(r, r') = \sum_i^N \varphi_i(r) d_i \varphi_i^*(r'). \quad (12)$$

Meanwhile, the result of  $V_x$  applied to an orbital  $\phi_j$  then is given by:

$$(V_x[P])\phi_j(r) = -\sum_i^N d_i \varphi_i(r) \int K(r, r') \varphi_i^*(r') \phi_j(r') dr', \quad (13)$$

As illustrated in Fig. 2(b), the only additional overhead introduced is the single diagonalization of  $\sigma$  after each update and the basis set transformation of the wavefunctions during the calculation of density and exchange operators. The number of FFTs in  $V_x\Phi$  calculation is greatly reduced from  $O(N^3)$  to  $O(N^2)$ , decreasing from a triple loop to a double loop, and communication volume from  $O(N_g N^2)$  to  $O(N_g N)$ . Fig. 2 shows the comparison between the naive and optimized versions of the Fock exchange operator. Similarly, the number of FFTs in the calculation of charge density can also be reduced from  $O(N^2)$  to  $O(N)$ .

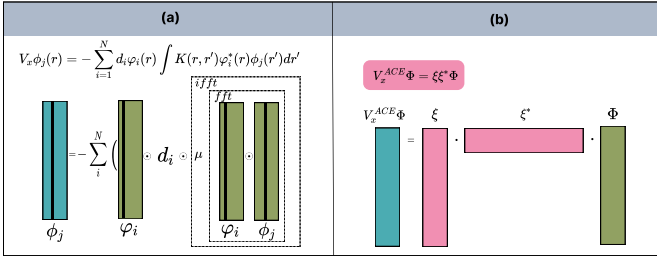


Fig. 3. Evaluation of  $V_x\Phi$ . (a) Direct two-electron integral. (b) ACE operator.

2) *Reduce the frequency of Fock exchange operator by adaptively compressed exchange (ACE)*: The Fock exchange operator remains the most computationally intensive part after introducing the occupation matrix diagonalization, i.e., it still takes 90% of the total time for a silicon system of 384 atoms. One way to further optimize it is to reduce the frequency of the Fock exchange operator by adopting the ACE formulation, which is introduced by Lin [37]. The construction of the low-rank ACE operator is as follows:

$$W_i(r) = (V_x \phi_i)(r) = (V_X^{ACE} \phi_i)(r), \quad (14)$$

$$V_X^{ACE}(r, r') = -\sum_{i=1}^{N_e} \xi_k(r) \xi_k(r').$$

More theoretical details on  $W$  and  $\xi$  are described in Ref. [37]. Fig. 2 shows the computational procedure of both two-electron integral and ACE operator.

To integrate the ACE method into PT-IM, two ACE operators are required due to the implicit midpoint rule:  $V_{x_n}^{ACE}$  and  $V_{x_{n+\frac{1}{2}}}^{ACE}$ . Those ACE operators are incorporated into PT-IM via a double self-consistent field (SCF) loop, where both  $V_{x_n}^{ACE}$  and  $V_{x_{n+\frac{1}{2}}}^{ACE}$  are constructed in the outer SCF. During the evaluation of  $H\Phi$  of the inner SCF, the ACE operators  $V_X^{ACE}$  can replace the previous Fock exchange operator, transforming the previously two-electron integral into more efficient matrix-matrix multiplications of size  $N_g \times N$ . Fig. 3(b) shows a detailed workflow of the PT-IM-ACE. Note that ACE operator

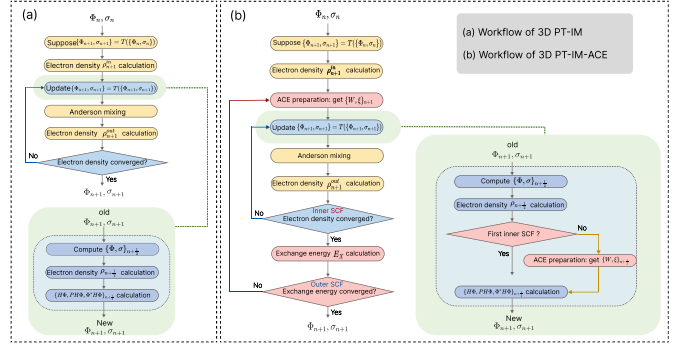


Fig. 4. One time step propagation of the rt-TDDFT using (a) PT-IM (b) PT-IM-ACE with double loop to reduce the frequency of Fock exchange operator.

can greatly reduce the frequency of the Fock exchange operator application. For example, to fully converge for a silicon system of 384 atoms, an average of 25 SCF steps are required, meaning that 25 Fock exchange operators are evaluated in the previous implementation (Fig. 4(a)). With the introduction of ACE operator, it takes about 5 outer SCF iterations, with each outer SCF averaging 13 inner SCF iterations. This optimization reduces the number of Fock exchange operator calculation by 20, or 80%, in a single time-step propagation.

### B. System innovation

Since all computational intensive parts have been migrated to GPU/ARM in Sec. III, we focus on the communication time and memory footprint in this subsection. The most time-consuming MPI operation is the wavefunction MPI\_Bcast in the evaluation of the Fock exchange operator. Fig. 5(a) illustrates a naive implementation of MPI\_Bcast with 4 MPI tasks. In this setup, 4 steps of MPI\_Bcast are performed to evaluate the Poisson-like equation for all wavefunction pairs (i,j). To reduce the communication time associated with the wavefunctions, we perform several steps of optimization.

1) *Ring-based point-to-point pattern*: We propose a ring-based point-to-point (p2p) communication pattern, as shown in Fig. 5 (b). In this approach, wavefunctions are rotated among processes through point-to-point MPI communications. Within each step, MPI tasks send and receive wavefunctions from its adjacent processes.

The ring-based method offers distinct advantages over the conventional broadcast approach in both communication pattern and latency. Unlike broadcasting, which requires global communication and can impact the entire network, the ring-based approach limits communication to neighboring processes. This localized communication significantly reduces network load and minimizes congestion. Additionally, in terms of latency, the ring-based method ensures that each communication step occurs within a single hop, which is highly advantageous in most network topologies. As a result, this method greatly improves scalability by reducing communication burdens and times.

2) *Asynchronous ring-based method*: Furthermore, the performance of the ring-based method can be substantially



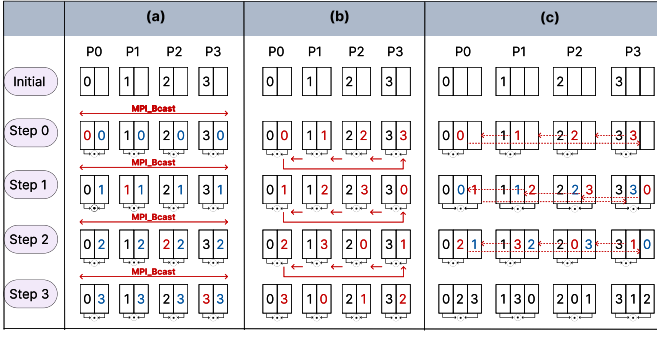


Fig. 5. Communication pattern of wavefunctions across 4 processes. (a). Bcast-based method. (b). Ring-based point-to-point pattern. (c). Asynchronous ring-based method. The red two-way arrow solid line indicates MPI\_Bcast communication, and the red one-way arrow solid line is point-to-point communication. The dashed red one-way arrow stands for asynchronous point-to-point communication.  $\odot$  denotes element-wise multiplication between two wavefunctions.

enhanced by leveraging asynchronous execution to overlap communication with computation. The process is shown in Fig. 5(c). In each step, a process asynchronously transmits its local wavefunctions (or those received in the previous step) to the next neighboring process while simultaneously beginning to asynchronously receive wavefunctions from the previous neighbor. After completing these initial computations, the process waits for the communication phase to finish before proceeding to the next step. This iterative process consists of *mpisize* steps.

Overlapping communication with computation can further reduce the total runtime, significantly improving program performance. While this technique can also be applied to Bcast-based method, its effectiveness depends on whether computation or communication takes longer. As noted in Sec. VIII-D, our tests show that communication time exceeds computation time, meaning that communication ultimately determines the total runtime. The broadcast method generally increases communication time, thereby reducing the benefits of overlapping computation with communication.

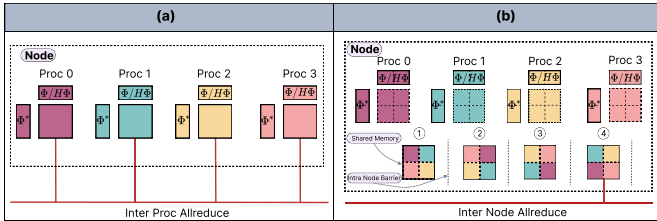


Fig. 6. Calculation for  $\Phi^* \Phi$  or  $\Phi^* H \Phi$  across 4 processes. (a) The original. (b) optimized by shared memory mechanism. The matrix with dashed border has not been allocated memory.

3) *Reduce memory footprint using shared memory mechanism*: The idea is to use inter-process shared memory to reduce memory usage. In the original implementation, certain matrices, such as  $\sigma$ , and intermediate results like  $\Phi^* \Phi$  and  $\Phi^* H \Phi$ , are not memory scalable. Consider a system with 768 silicon atoms and 1920 electronic orbitals, characterized by a

substantial grid size of  $N_g = 324000$ . When using more than 168 processes, the memory advantage of scalable matrices, such as wavefunctions, diminishes. At this stage, the memory consumed by non-scalable square matrices becomes significant and cannot be ignored.

Our key idea is to use shared memory between processes to store these matrices, as they are identical across all processes. Shared memory is allocated using the MPI SHM Extension [38], allowing processes on the same computing node to share the same matrix. If  $p$  processes are launched on a node, the memory usage for these matrices is reduced to  $1/p$  of the original amount. This optimization primarily enables the simulation of larger systems.

As for performance improvement, although inter-process allreduce was replaced with inter-node allreduce, reducing the number of processes involved in communication to one-quarter of the original, there was no significant performance gain. This lack of improvement is due to the introduction of remote memory access in NUMA systems. When the runtime, the  $\Phi \Phi$  and  $\Phi H \Phi$  matrices we stored in the shared memory were allocated to a single NUMA node in the physical memory, which would lead to the remote memory access problem when the computing cores in the other three NUMA nodes accessed this matrix, resulting in a loss of performance. In fact, with our method, by sacrificing little computing performance, we reduce the number of communication process, communication volume and memory footprint of this part to a quarter of the original. This enables us to scale up to a larger size.

## V. MACHINE CONFIGURATION

All our tests are performed on both ARM and GPU platforms. The first machine is Fugaku, an ARM many-core supercomputer currently ranked fourth in the Top500 list [39], with a theoretical peak performance of 537.21 PFLOPS. Fugaku is comprised of 158,976 computing nodes interconnected through a 6D-torus network. Each node is equipped with one A64FX ARM CPU, which has four core memory groups (CMGs). Each CMG has 13 cores (1 for OS and 12 for compute) and 8GB of HBM2 memory (32GB HBM2 per node). Additionally, each computing core supports 512-bit SVE vector instructions, allowing an A64FX to reach a theoretical peak performance of 3.38 TFLOPS at 2.2GHz, with a theoretical memory bandwidth of 1024GB/s.

The second platform is a GPU cluster featuring NVIDIA A100 GPUs. Each computing node is equipped with one ARM-based Kunpeng-920 CPU, 256GB of DDR4 memory, and 4 NVIDIA A100 GPUs. Each Kunpeng-920 CPU has 128 cores distributed across four NUMA domains, each supporting 128-bit NEON vector instructions. Each A100 GPU accelerator offers a theoretical peak performance of 9.7 TFLOPS (19.5 TFLOPS with tensor cores) and 40GB of HBM2 memory, achieving a theoretical bandwidth of 1.5TB/s. The CPU and GPUs are interconnected via a PCIe bus with a bi-directional bandwidth of 64GB/s. The computing nodes are interconnected through a fat-tree network.

## VI. PHYSICAL SYSTEM

Silicon systems ranging from 48 to 3072 atoms, corresponding to the supercell constructed from  $1 \times 1 \times 3$  to  $6 \times 8 \times 8$  unit cells. Each simple cubic unit cell consists of 8 silicon atoms with the lattice constant being 5.43 Å. In our accuracy tests, the number of extra states is set to  $N_{atom}$ , and it is set to  $\frac{1}{2}N_{atom}$  in all other tests.

In our tests, the external potential is a laser pulse shown in Fig. 7(a), and its wavelength is 380 nm. The total simulation time is 30 fs, with a time step of 50 as for both PT-IM and PT-IM-ACE methods. The stopping criteria is set to  $1.0 \times 10^{-6}$  for electron density and exchange energy errors. We use the SG15 Optimized Norm-Conserving Vanderbilt (ONCV) pseudopotentials [40], [41] and HSE06 functionals [23] in all tests. The kinetic energy cutoff is set to 10 Hartree and the temperature is set to 8000K. The average number of outer and inner SCFs is 5 and 13, respectively. The maximum Anderson mixing dimension is set to 20.

For the system with 1536 atoms, the number of grid points for a wavefunction is  $N_g = 60 \times 90 \times 120 = 648,000$ . This corresponds to a charge density grid  $120 \times 180 \times 240$ . The Fock exchange operator is evaluated on the wavefunction grid. The number of orbitals is  $N = 1536 \times 2 + \frac{1}{2} \times 1536 = 3840$ .

## VII. PHYSICAL RESULTS

In this section, we test the accuracy of the optimized code and describe the motion of electrons during the rt-TDDFT simulation.

### A. Accuracy

The accuracy is evaluated using the dipole moment along the x-direction and the total energy, as shown in Fig. 7. For the 380 nm laser case, Fig. 7 demonstrates that the results of PT-IM-ACE with a 50 as time step fully match those obtained using the RK4 method with a time step 100 times smaller. Furthermore, the enlarged section in Fig. 7 confirms that PT-IM-ACE provides a very good approximation to the electron dynamics compared to RK4 during the final 100 time steps (25-30 fs), regardless of whether the system is in a pure or mixed state. It is important to note that electrons already exhibit fractional occupation at the beginning of the finite temperature (8000K) rt-TDDFT simulation.

### B. Electrons motions

The motion of electrons in finite temperature rt-TDDFT is shown in Fig. 8, with the laser pulse shown in Fig. 7(a). Initially, the occupation number matrix  $\sigma_t$  is depicted in Fig. 8(c), with elements from 0 to 1 indicating the probability of electron occupying each orbital. During the simulation, the variation of the off-diagonal element  $\sigma_t(0, 2)$  over time is shown in Fig. 8(a), demonstrating the stochastic nature of electron motion. Meanwhile, as an example of diagonal elements, the variation of  $\sigma_t(22, 22)$  over time is shown in Fig. 8(b), increasing as the external field is strengthening (10-15 fs). This indicates that the stronger the external laser field, the more active the electrons are. Enhanced electron activity

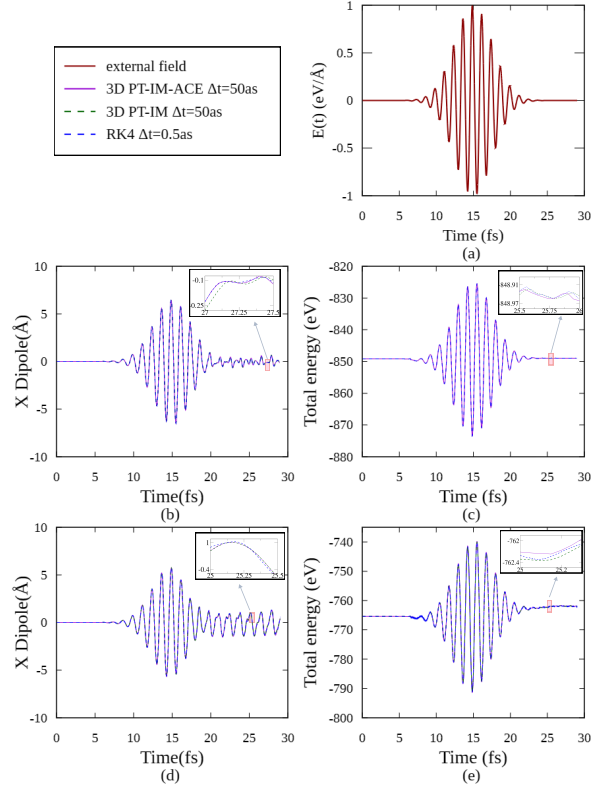


Fig. 7. Electron dynamics of an 8 atom silicon system under a laser pulse with 380 nm. (a) Electric field along the x direction. (b) Dipole moment along the x direction in pure states. (c) Total energy in pure states. (d) Dipole moment along the x direction in mixed states (Total states = 24). (e) Total energy in mixed states (Total states = 24).

in laser fields implies that modifying material's electronic structures and band properties can significantly affect their optical responses, offering new avenues for optoelectronic device innovation.

## VIII. PERFORMANCE RESULTS AND ANALYSIS

In this section, we perform detailed performance tests, including step-by-step performance improvements, strong scaling, weak scaling, and communication analysis. On the ARM platform, all our tests are conducted with four MPI ranks per node, matching the A64FX's four NUMA architectures. Each process launches 12 threads to manage the 12 computing cores within a CMG, with access to 8GB of memory. On the GPU platform, each compute node is equipped with four A100 GPUs, so we initiate four MPI ranks per node, with each process controlling one A100 GPU. Consequently, each process has access to 64GB of host memory and 40GB of GPU memory.

### A. Step-by-step performance improvement

On both platforms, step-by-step performance improvements are evaluated using a 384 silicon atom system on 240 ARM nodes and 24 GPU nodes. Results are shown in Fig. 9. The baseline test (BL) represents the original PTIM algorithm, accelerated by OpenMP and GPU on two respective platforms.



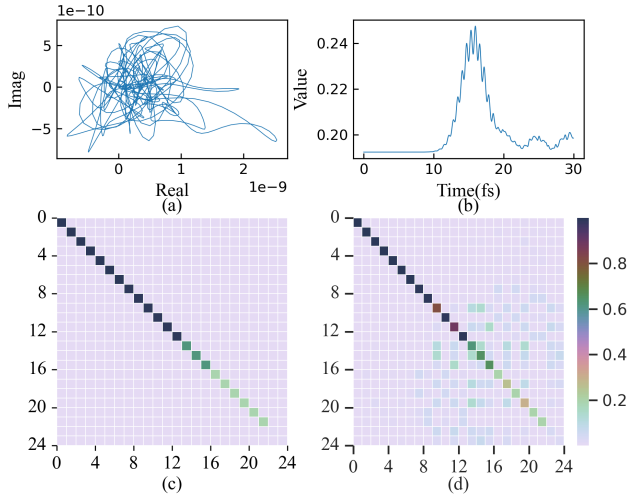


Fig. 8. States evolution of an 8-atom silicon system under laser pulse irradiation over 30 fs with 4 processes. (a) Relationship between the real and imaginary parts of the off-diagonal element  $\sigma_t(0, 2)$  over 30 fs. (b) Variation of the diagonal element  $\sigma_t(22, 22)$  over 30 fs. (c) Initial  $\sigma_t$ . (d) Final  $\sigma_t$ .

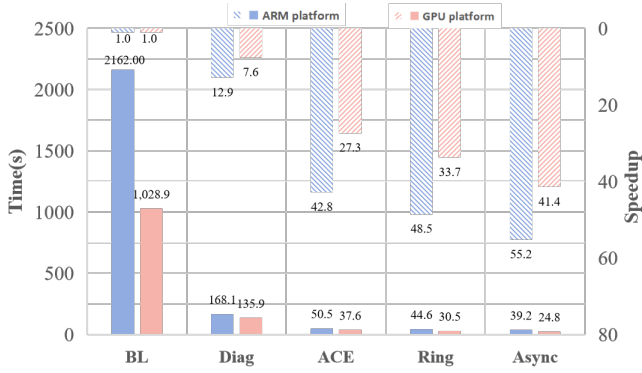


Fig. 9. Step-by-step performance improvement for one time step on GPU/ARM platforms with 384 silicon atom system using 240/24 nodes on ARM/GPU platform. The baseline is the results calculated by the initial PTIM method accelerated by OpenMP and GPU.

1) *Occupation matrix diagonalization*: As shown in Fig. 9, the occupation matrix diagonalization method, labeled as "Diag" in the figures, is introduced in Sec. IV-A1. For the 384 atom system, this method accelerated per step performance of the PT-IM in PWDFt by 12.86x on the ARM and 7.57x on the GPU platform. In Sec. IV-A1, we describe the diagonalization algorithm, which reduces the computational complexity of  $V_x\Phi$ . As system size  $N$  increases, the importance of  $V_x\Phi$  calculation grows, and the benefit of complexity reduction becomes more significant. Therefore, the speedup of the diagonalization of occupation matrix is more substantial for larger systems.

2) *ACE method*: The ACE operator method, introduced in Sec. IV-A2 and labeled as "ACE" in Fig. 9, reduces the number of computations for  $V_x\Phi$  from 25 to 5. On the ARM/GPU platform, for the 384 atom system, the computation time of  $H\Phi$  decrease from 148.5s/110.6s to 6s/20.3s, with the total ACE preparation time being 23s/17.4s. As shown in Fig. 9,

the ACE operator accelerates the per-step time by 3.3x/3.6x.

3) *Ring-based method*: The performance gain from the ring-based method, detailed in Sec. IV-B1 and denoted as a "Ring", is shown in Fig. 9. Compared to the Bcast-based method in the previous step, it accelerates the 384 atom system by 1.13x/1.23x on the ARM/GPU platform.

4) *Asynchronous ring-based method*: The asynchronous ring-based method is detailed in Sec. IV-B2 and is denoted as "Async" in Fig. 9. Compared to the ring-based method in the previous step, it gains a speedup of 1.14x/1.23x on the ARM/GPU platform for the 384 atom system.

Further details on the MPI communication time across different methods will be analyzed in more detail in Sec. VIII-D. We remark that the more nodes used, the greater the improvement of communication optimization.

## B. Strong scaling

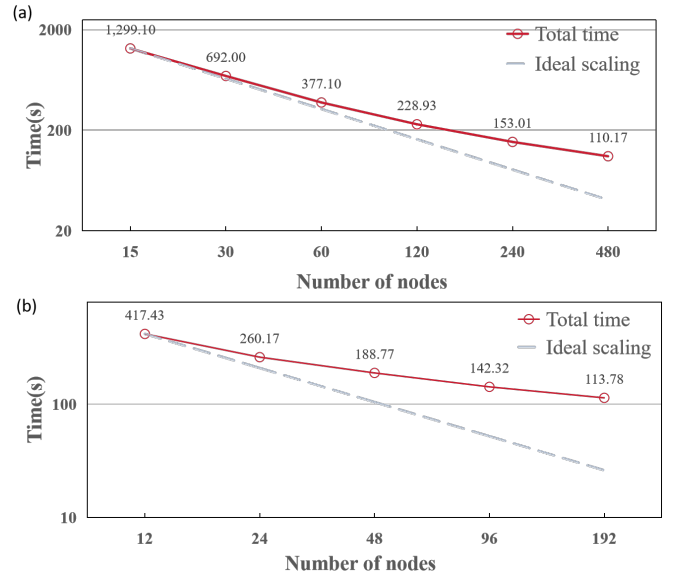


Fig. 10. Strong scaling: wall clock time per 50 as for silicon systems on two HPC systems. The "ideal scaling" here scales as  $O(N)$ . (a). Strong scaling with 768 silicon atom system on ARM platform. (b). Strong scaling with 1536 silicon atom system on GPU platform.

Fig. 10 (a)(b) shows the strong scaling of one time step using the optimized PT-IM method for a 768-atom silicon system on the ARM platform and a 1536-atom system on the GPU platform, respectively. On the ARM platform, the parallel efficiency is 36.8% when increasing the number of nodes by 32 times. On the GPU platform, the same increase yields a parallel efficiency of 22.9%. The dropping of the parallel efficiency are highly related to the data communication and computational efficiency. First, as the number of nodes increases, communication time grows due to MPI\_Sendrecv in ring-based method and MPI\_Allreduce operations required by the Rayleigh-Ritz procedure. For example, on the ARM platform, when increasing from 15 nodes to 480 nodes, the MPI\_Sendrecv grows by a factor of 1.5, from 4.7 seconds to 7.1 seconds and the MPI\_Allreduce grows by a factor of 1.4,

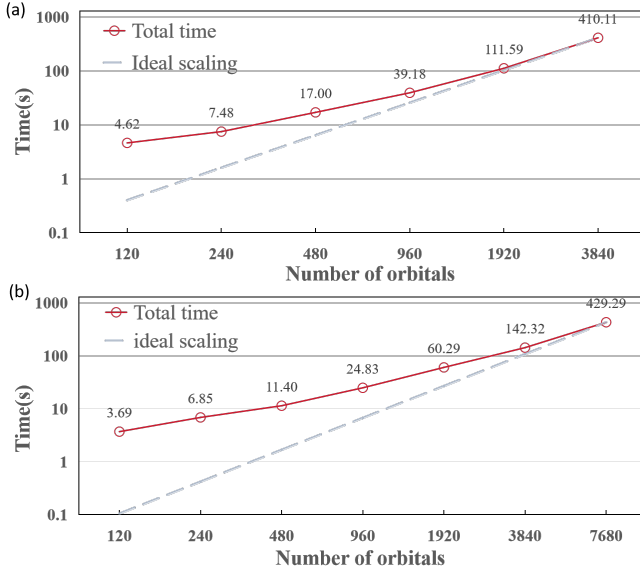


Fig. 11. Weak scaling: wall clock time per 50 as for silicon systems on two HPC systems. The “ideal scaling” here scales as  $O(N^2)$  on both platforms. (a). Weak scaling on the ARM platform, from 48 atoms to 1536 atoms. The number of nodes used is always set to 1/4 of the number of total orbitals in the calculation. (b). Weak scaling on the GPU platform, from 48 atoms to 3072 atoms. The number of nodes is used are always set to 1/40 of the number of total orbitals in the calculation.

from 2.6 seconds to 3.7 seconds. On the GPU platform, when increasing from 12 nodes to 192 nodes, the MPI\_Sendrecv grows by a factor of 1.6, from 6.3 seconds to 10.1 seconds and the MPI\_Allreduce grows by a factor of 1.5, from 2.9 seconds to 4.3 seconds. Second, the computational efficiency decreases as per-node workload scales. For example, on the ARM platform, when computing resources expand by 32 times, the computing efficiency drops to 40% of the original. On the GPU platform, when computing resources expand by 16 times, the computing efficiency reduces to 26% of the original.

Compared to the GPU platform, the optimized PWDFT demonstrates higher parallel efficiency on the ARM platform. This is primarily due to two factors. First, the ratio of theoretical peak performance to peak bandwidth is lower on the ARM platform (3.4 Flop/Byte) compared to the GPU platform (6.5 Flop/Byte), allowing for better performance since PWDFT is bandwidth-bound. Second, the ARM platform features a 6D torus network architecture, which provides superior network performance.

### C. Weak scaling

Fig. 11 shows the weak scaling of the optimized PT-IM code on both ARM and GPU platforms. We find that the system size is primarily constrained by the memory capacity of the hardware. For example, our optimized PWDFT can accommodate only 1536 atoms on 960 computing nodes on Fugaku, limited by the 8GB memory capacity of each NUMA node. On the GPU platform, due to the number of nodes we can access, PWDFT can only scale up to 3072 atoms

on 192 nodes. Even with more nodes, the current machine configuration is unable to handle 6144 silicon atoms because of memory limitations. The simulation of 3072 atoms already consumes over 80% of the available global memory and 75% theoretical peak bandwidth per process. This also indicates that PT-IM is a memory-bandwidth bounded problem. If our GPUs have larger global memory, it might be possible to double the simulation scale.

The computational complexity of hybrid functional rt-TDDFT simulation scales as  $O(N^3)$ . Notably, on both platforms, when the number of orbitals is relatively low, doubling it results in a much smaller increase in computing time than the theoretical fourfold. However, as the system size grows, the time required to double the computational workload approaches to the theoretical four times increase. This is because, in smaller systems, less time is spent on the Fock exchange operator  $V_x \Phi$ . As the system scale scales up, the relative time spent on these operations increases, eventually becoming the dominant factor in the overall simulation time. For a smaller system with 192 atoms, simulating one time step on the GPU platform using 12 nodes takes 11.40 seconds, meaning that each femtosecond of simulation requires approximately 3.5 minutes. For a larger system with 3072 atoms, simulating one time step with 192 computing nodes takes 429.29 seconds, implying that each femtosecond of simulation takes about 2.5 hours.

### D. Communication analysis

In this section, we evaluate the MPI communication time after system optimizations. Table. I shows results from 1536-atom tests on ARM and GPU platforms, using 960 and 96 nodes, respectively. A ‘-’ indicates that no such communication occurs in the program.

This subsection focuses on communication, and the optimization methods prior to ACE do not directly reduce communication. Therefore, we start communication optimization analysis from PT-IM after ACE optimization.

On the ARM/GPU platform, the communication time for wavefunctions using the bcst-based method accounts for 74%/83% of the total MPI communication time. The ring-based method reduces the corresponding time from 67.22s/64.85s to 30.1s/20.54s. For asynchronous ring communication, after overlapping computation and communication, the time spent on MPI\_Wait is 20.13/10.1 seconds on the ARM/GPU platforms, respectively.

We note that on two platforms, after asynchronous communication, MPI\_Wait time is greater than zero, indicating that communication time exceeds computation time. Even with the overlap of computation and communication, communication remains the bottleneck, highlighting the importance of replacing the bcst-based method with the ring-based mechanism.

In Table. I, the proportion of communication time on the GPU platform is higher than that on the ARM platform, even though the number of processes on the GPU platform is only one tenth of that on the ARM platform. This can be attributed to two factors. First, although the computational workload per

TABLE I  
MPI COMMUNICATION TIME WITH THE OPTIMIZED METHODS FOR A BIG SYSTEM OF 1536 SILICON ATOMS ON BOTH PLATFORMS

		Alltoallv (s)	Sendrecv (s)	Wait (s)	Allgatherv (s)	Allreduce (s)	Bcast (s)	Total communication time(s)	Communication ratio(%)
<b>ARM platform</b>	ACE	9.04	-	-	0.17	14.19	67.22	90.62	18.92
	Ring	9.03	30.1	-	0.17	14.21	0.03	53.54	12.73
	Async	9.18	-	20.13	0.17	14.18	0.03	43.69	10.65
<b>GPU platform</b>	ACE	7.95	-	-	0.47	4.99	64.85	78.26	25.72
	Ring	7.35	20.54	-	0.47	4.46	0.89	33.71	21.13
	Async	7.64	-	10.1	0.47	4.28	0.82	23.31	16.38

process on the GPU is ten times that on the ARM platform, the computational power per process on the GPU platform (9.7 TFLOPS) is 11.5 times greater than the ARM platform (0.84 TFLOPS). Second, our GPU cluster is not equipped with NVLink and does not support GPUDirect communication, which negatively impacts communication time. On the GPU platforms equipped with NVLink, such as Summit, the communication performance of our program will be further improved [42].

## IX. CONCLUSION

In this paper, we first implement a three-dimensional PT-IM algorithm in PWDFT, enabling finite-temperature rt-TDDFT simulation with hybrid functional. In terms of algorithms, we propose a diagonalization method to reduce computation and communication complexity, nearing pure state levels. Additionally, we employ the ACE method to significantly reduce the frequency of the most expensive Fock exchange operator. In terms of computer architecture, the ring-based method is utilized to optimize communication patterns and extensively reduce the communication load. Performance is further enhanced by overlapping computation and communication. The shared memory mechanism is used to alleviate the memory consumption. The correctness of our implementation is proved in the physical result. The step-by-step performance improvement test reveals our optimization achieving speeds up of 51.15 and 41.44 times speed up on ARM and GPU platforms, respectively. The strong scaling results show that when nodes scale 32 and 16 times, time-to-solution was accelerated by 11.79x/3.67X, on ARM and GPU platforms, respectively. The simulation system is respectively extended to 1536/3072 atoms(6144/12288 electrons) on two platforms. Our work paves the way for large-scale rt-TDDFT simulation for real material with finite temperatures.

## REFERENCES

- [1] X. Andrade, J. Alberdi-Rodriguez, D. A. Strubbe, M. J. Oliveira, F. Nogueira, A. Castro, J. Muguerza, A. Arruabarrena, S. G. Louie, A. Aspuru-Guzik *et al.*, "Time-dependent density-functional theory in massively parallel computer architectures: the octopus project," *Journal of Physics: Condensed Matter*, vol. 24, no. 23, p. 233202, 2012.
- [2] G. Onida, L. Reining, and A. Rubio, "Electronic excitations: density-functional versus many-body green's-function approaches," *Reviews of modern physics*, vol. 74, no. 2, p. 601, 2002.
- [3] E. Runge and E. K. Gross, "Density-functional theory for time-dependent systems," *Physical review letters*, vol. 52, no. 12, p. 997, 1984.
- [4] C. A. Ullrich, "Time-dependent density-functional theory: concepts and applications," 2011.
- [5] X. Wu, A. Selloni, and R. Car, "Order-n implementation of exact exchange in extended insulating systems," *Physical Review B*, vol. 79, no. 8, p. 085102, 2009.
- [6] Z. Wang, S.-S. Li, and L.-W. Wang, "Efficient real-time time-dependent density functional theory method and its application to a collision of an ion with a 2d material," *Physical review letters*, vol. 114, no. 6, p. 063004, 2015.
- [7] S. A. Fischer, C. J. Cramer, and N. Govind, "Excited state absorption from real-time time-dependent density functional theory," *Journal of chemical theory and computation*, vol. 11, no. 9, pp. 4294–4303, 2015.
- [8] W.-H. Liu, J.-W. Luo, S.-S. Li, and L.-W. Wang, "Microscopic force driving the photoinduced ultrafast phase transition: Time-dependent density functional theory simulations of irte 2," *Physical Review B*, vol. 102, no. 18, p. 184308, 2020.
- [9] M. E. Casida and M. Huix-Rotllant, "Progress in time-dependent density-functional theory," *Annual review of physical chemistry*, vol. 63, pp. 287–323, 2012.
- [10] S. Cushing, "Plasmonic hot carriers skip out in femtoseconds," *Nature Photonics*, vol. 11, no. 12, pp. 748–749, 2017.
- [11] M. Lucchini, S. A. Sato, A. Ludwig, J. Herrmann, M. Volkov, L. Kasmi, Y. Shinohara, K. Yabana, L. Gallmann, and U. Keller, "Attosecond dynamical franz-keldysh effect in polycrystalline diamond," *Science*, vol. 353, no. 6302, pp. 916–919, 2016.
- [12] A. Moulet, J. B. Bertrand, T. Klostermann, A. Guggenmos, N. Karpowicz, and E. Goulielmakis, "Soft x-ray excitonics," *Science*, vol. 357, no. 6356, pp. 1134–1138, 2017.
- [13] F. Schlaepfer, M. Lucchini, S. A. Sato, M. Volkov, L. Kasmi, N. Hartmann, A. Rubio, L. Gallmann, and U. Keller, "Attosecond optical-field-enhanced carrier injection into the gas conduction band," *Nature Physics*, vol. 14, no. 6, pp. 560–564, 2018.
- [14] M. Schultze, K. Ramasesha, C. Pemmaraju, S. Sato, D. Whitmore, A. Gandman, J. S. Prell, L. Borja, D. Prendergast, K. Yabana *et al.*, "Attosecond band-gap dynamics in silicon," *Science*, vol. 346, no. 6215, pp. 1348–1352, 2014.
- [15] S. Tan, A. Argondizzo, J. Ren, L. Liu, J. Zhao, and H. Petek, "Plasmonic coupling at a metal/semiconductor interface," *Nature Photonics*, vol. 11, no. 12, pp. 806–812, 2017.
- [16] M. Zürich, H.-T. Chang, L. J. Borja, P. M. Kraus, S. K. Cushing, A. Gandman, C. J. Kaplan, M. H. Oh, J. S. Prell, D. Prendergast *et al.*, "Direct and simultaneous observation of ultrafast electron and hole dynamics in germanium," *Nature communications*, vol. 8, no. 1, p. 15734, 2017.
- [17] W. Jia, D. An, L.-W. Wang, and L. Lin, "Fast real-time time-dependent density functional theory calculations with the parallel transport gauge," *Journal of Chemical Theory and Computation*, vol. 14, no. 11, pp. 5645–5652, 2018.
- [18] D. M. Ceperley and B. J. Alder, "Ground state of the electron gas by a stochastic method," *Physical review letters*, vol. 45, no. 7, p. 566, 1980.
- [19] J. P. Perdew and A. Zunger, "Self-interaction correction to density-functional approximations for many-electron systems," *Physical review B*, vol. 23, no. 10, p. 5048, 1981.
- [20] J. P. Perdew, K. Burke, and M. Ernzerhof, "Generalized gradient approximation made simple," *Physical review letters*, vol. 77, no. 18, p. 3865, 1996.
- [21] K. Raghavachari, "Perspective on "density functional thermochemistry. iii. the role of exact exchange" becke ad (1993) j chem phys 98: 5648–52," *Theoretical Chemistry Accounts*, vol. 103, pp. 361–363, 2000.
- [22] J. P. Perdew, M. Ernzerhof, and K. Burke, "Rationale for mixing exact exchange with density functional approximations," *The Journal of chemical physics*, vol. 105, no. 22, pp. 9982–9985, 1996.

- [23] J. Heyd, G. E. Scuseria, and M. Ernzerhof, "Hybrid functionals based on a screened coulomb potential," *The Journal of chemical physics*, vol. 118, no. 18, pp. 8207–8215, 2003.
- [24] J. Heyb, G. Scuseria, and M. Ernzerhof, "Erratum: Hybrid functionals based on a screened coulomb potential," *J. Chem. Phys.*, vol. 124, no. 21, p. 219906, 2006.
- [25] M. Del Ben, C. Yang, Z. Li, F. H. da Jornada, S. G. Louie, and J. Deslippe, "Accelerating large-scale excited-state gw calculations on leadership hpc systems," in *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE, 2020, pp. 1–11.
- [26] W. Jia, L.-W. Wang, and L. Lin, "Parallel transport time-dependent density functional theory calculations with hybrid functional on summit," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2019, pp. 1–23.
- [27] M. Alducin, R. D. Muiño, and J. Juaristi, "Non-adiabatic effects in elementary reaction processes at metal surfaces," *Progress in Surface Science*, vol. 92, no. 4, pp. 317–340, 2017.
- [28] D. An, D. Fang, and L. Lin, "Parallel transport dynamics for mixed quantum states with applications to time-dependent density functional theory," *Journal of Computational Physics*, vol. 451, p. 110850, 2022.
- [29] X. Gonze, F. Jollet, F. A. Araujo, D. Adams, B. Amadon, T. Applencourt, C. Audouze, J.-M. Beuken, J. Bieder, A. Bokhanchuk *et al.*, "Recent developments in the abinit software package," *Computer Physics Communications*, vol. 205, pp. 106–131, 2016.
- [30] W. Jia, Z. Cao, L. Wang, J. Fu, X. Chi, W. Gao, and L.-W. Wang, "The analysis of a plane wave pseudopotential density functional theory code on a gpu machine," *Computer Physics Communications*, vol. 184, no. 1, pp. 9–18, 2013.
- [31] W. Jia, J. Fu, Z. Cao, L. Wang, X. Chi, W. Gao, and L.-W. Wang, "Fast plane wave density functional theory molecular dynamics calculations on multi-gpu machines," *Journal of Computational Physics*, vol. 251, pp. 102–115, 2013.
- [32] J. Romero, E. Phillips, G. Ruetsch, M. Fatica, F. Spiga, and P. Giannozzi, "A performance study of quantum espresso's pwscf code on multi-core and gpu systems," in *High Performance Computing Systems. Performance Modeling, Benchmarking, and Simulation: 8th International Workshop, PMBS 2017, Denver, CO, USA, November 13, 2017, Proceedings 8*. Springer, 2018, pp. 67–87.
- [33] M. Hacene, A. Anciaux-Sedrakian, X. Rozanska, D. Klahr, T. Guignon, and P. Fleurat-Lessard, "Accelerating vasp electronic structure calculations using graphic processing units," *Journal of computational chemistry*, vol. 33, no. 32, pp. 2581–2589, 2012.
- [34] L. E. Ratcliff, A. Degomme, J. A. Flores-Livas, S. Goedecker, and L. Genovese, "Affordable and accurate large-scale hybrid-functional calculations on gpu-accelerated supercomputers," *Journal of Physics: Condensed Matter*, vol. 30, no. 9, p. 095901, 2018.
- [35] M. Valiev, E. J. Bylaska, N. Govind, K. Kowalski, T. P. Straatsma, H. J. J. Van Dam, D. Wang, J. Nieplocha, E. Aprà, T. L. Windus *et al.*, "Nwchem: A comprehensive and scalable open-source solution for large scale molecular simulations," *Computer Physics Communications*, vol. 181, no. 9, pp. 1477–1489, 2010.
- [36] D. G. Anderson, "Iterative procedures for nonlinear integral equations," *Journal of the ACM (JACM)*, vol. 12, no. 4, pp. 547–560, 1965.
- [37] L. Lin, "Adaptively compressed exchange operator," *Journal of chemical theory and computation*, vol. 12, no. 5, pp. 2242–2249, 2016.
- [38] M. Brinskiy and M. Lubin, "An introduction to mpi-3 shared memory programming," Available <https://software.intel.com/enus>, 2017.
- [39] "Top 500 list," <https://www.top500.org/lists/top500/2023/11/>, 11 2023.
- [40] M. Schlupf and F. Gygi, "Optimization algorithm for the generation of oncv pseudopotentials," *Computer Physics Communications*, vol. 196, pp. 36–44, 2015.
- [41] D. Hamann, "Optimized norm-conserving vanderbilt pseudopotentials," *Physical Review B*, vol. 88, no. 8, p. 085117, 2013.
- [42] W. Elwasif, W. Godoy, N. Hagerty, J. A. Harris, O. Hernandez, B. Joo, P. Kent, D. Lebrun-Grandie, E. Maccarthy, V. Melesse Vergara *et al.*, "Application experiences on a gpu-accelerated arm-based hpc testbed," in *Proceedings of the HPC Asia 2023 Workshops*, 2023, pp. 35–49.