2.5 Million-Atom *Ab Initio* Electronic-Structure Simulation of Complex Metallic Heterostructures with DGDFT

Wei Hu[‡†], Hong An[‡§*], Zhuoqiang Guo[¶†], Qingcai Jiang[‡†], Xinming Qin[‡†], Junshi Chen[‡§], Weile Jia[¶*], Chao Yang[‖], Zhaolong Luo[‡], Jielan Li[‡], Wentiao Wu[‡], Guangming Tan[¶], Dongning Jia[§], Qinglin Lu[**], Fangfang Liu[**], Min Tian[††], Fang Li[‡‡], Yeqi Huang[‡], Liyi Wang[‡], Sha Liu[‡] and Jinlong Yang[‡*]

[‡] University of Science and Technology of China, Hefei, Anhui, China

[§] Pilot National Laboratory for Marine Science and Technology (Qingdao), China

[¶] Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China

[‖] School of Mathematical Sciences, Peking University, Beijing, China

[**] Institute of Software, Chinese Academy of Sciences, Beijing, China

[††] Qilu University of Technology, Shandong Computer Science Center, Jinan, Shangdong, China

[‡‡] National Research Center of Parallel Computer Engineering and Technology, Beijing, China

***Abstract*—Over the past three decades, *ab initio* electronic structure calculations of large, complex and metallic systems are limited to tens of thousands of atoms in computational accuracy and efficiency on leadership supercomputers. We present a massively parallel discontinuous Galerkin density functional theory (DGDFT) implementation, which adopts adaptive local basis functions to discretize the Kohn-Sham equation, resulting in a block-sparse Hamiltonian matrix. A highly efficient pole expansion and selected inversion (PEXSI) sparse direct solver is implemented in DGDFT to achieve $\mathcal{O}(N^{1.5})$ scaling for quasi two-dimensional systems. DGDFT allows us to compute the electronic structures of complex metallic heterostructures with 2.5 million atoms (17.2 million electrons) using 35.9 million cores on the new Sunway supercomputer. The peak performance of PEXSI can achieve 64 PFLOPS ($\sim$5% of theoretical peak), which is unprecedented for sparse direct solvers. This accomplishment paves the way for quantum mechanical simulations into mesoscopic scale for designing next-generation electronic devices.**

***Index Terms*—First-principles density functional theory, *ab initio* electronic structures, discontinuous Galerkin method, pole expansion and selected inversion algorithm, new Sunway supercomputer, complex metallic mesoscale heterostructures, next-generation devices**

I. Justification for Prize

Record 2.5-million-atom (100x improvement w.r.t. current state-of-the-art) *ab initio* electronic structure simulations for large-scale complex metallic heterostructures (200 nm, mesoscopic scale). PEXSI can reach an unprecedented 64 PFLOPS ($\sim$5% of theoretical peak) for sparse direct solvers. Corresponding time-to-solution can be three orders of magnitude faster than the current state-of-the-art.

†These authors contributed equally to this work.

*Corresponding Author: Hong An (han@ustc.edu.cn), Weile Jia (jiaweile@ict.ac.cn), Jinlong Yang (jlyang@ustc.edu.cn).

II. Performance Attributes

Performance attribute	Our submission
Category of achievement	Time-to-solution, scalability
Type of method used	Discontinuous Galerkin DFT
Results reported on basis of	Whole application including I/O
Precision reported	Double precision
System scale	Measured on whole system
Measurements	Timers, FLOP count

III. Overview of the Problem

First-principles materials simulation is the most accurate and effective quantum-mechanical methodology to explore the *ab initio* electronic structures for designing new high-efficiency energy materials and electronic devices. For new quantum multifunctional materials and next-generation electronics [1], [2], nanoscopic ($<$ 10 nm) and mesoscopic ($>$ 100 nm) heterostructures [3] with complex atomic structures and electronic properties have been proposed as strong candidates for solar cells, battery electrodes, field-effect transistors (FETs) [2], PN junctions and diodes, due to their superior electronic properties (e.g., bandgap opening, band alignment and charge transfer) as shown in Fig. 1(a). For example, as one of the most important two-dimensional (2D) materials, graphene and its interfaces with metals have attracted much attention in graphene FETs [4] because of its high mobility. Magic-angle twisted bilayer graphene (MATBG) [5] with the moiré superlattice can trap electrons on the flat graphene surface, which provides a good platform to investigate high-temperature superconductivity and topological insulator for next-generation FET channel materials [4]. However, even for the first maximum magic angle (1.1°) in MATBG, the smallest unitcell (10 nm) contains more than 10K atoms. Thus, multilayer MATBG systems with smaller magic angles in supercells readily reach up to 1M atoms ($>$ 100 nm) in real

SC22, November 13-18, 2022, Dallas, Texas, USA
978-1-6654-5444-5/22/$31.00 ©2022 IEEE

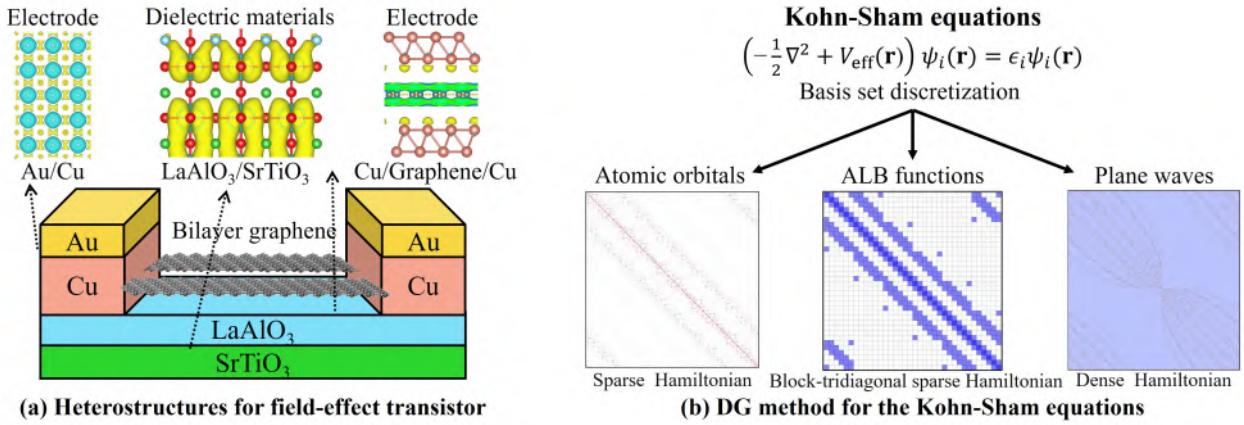

Fig. 1. (a) Four types of heterostructures for graphene FETs (Au/Cu, Cu/graphene/Cu (Cu/G/Cu), $LaAlO_3/SrTiO_3$ (LAO/STO) and bilayer graphene (BG)). (b) Three types of basis sets for discretizing the KS equations, including atomic orbitals, plane waves and adaptive local basis functions, with sparse, dense and block-sparse Hamiltonian metrics, respectively.

applications [2]. Metal alloys, such as lithium-sodium (Li/Na) and gold-copper (Au/Cu), act as electrodes in batteries [6] and FETs [1], especially light metal Li/Na alloy processes strong quantum effect on the battery performance. As dielectric substrates in FETs, a 2D superconductor electron gas [7] has been reported at the heterointerface between the two insulating oxides $LaAlO_3$ (LAO) and $SrTiO_3$ (STO) through spontaneous and piezoelectric polarization. Therefore, the ability to simulate *ab initio* electronic structures of these heterostructures is the key to the design of entire next-generation electronics at mesoscopic scale ($>$ 100 nm), such as FETs [1], [4], and is the heart of atomistic technology computer-aided design (TCAD) [2].

With an excellent balance between numerical accuracy and computational efficiency, the Kohn-Sham (KS) density functional theory (DFT) [8], [9] has become the most popular quantum-mechanical methodology to describe *ab initio* electronic structures of molecular and solid systems [10] in condensed matter physics, computational chemistry and materials science. However, most of the conventional DFT calculations have been performed for nanoscopic heterostructures ($<$ 1K atoms), which are far from the size of mesoscopic heterostructures ($>$ 100 nm and 1M atoms) in the entire next-generation electronics [2]. For example, low-angle ($< 10°$) surface dislocations [11] and grain boundaries in mesoscopic heterostructures always contain more than 10K atoms [2], such as Au/Cu, Li/Na, Cu/G/Cu, LAO/STO, multilayer graphene and MATBG (Fig. 1(a)).

A. Computational Challenges

A conventional DFT calculation for an N-atom system performs $\mathcal{O}(N^3)$ floating point operations and requires an $\mathcal{O}(N^2)$ memory footprint. In order to solve the KS equations, we need to select a basis set to represent the solution. For molecular systems, localized basis sets such as Gaussian-type orbitals [12] and numerical atomic orbitals [13] are preferred for low-scaling eigensolvers, especially linear-scaling methods [13]. For periodic solid systems, especially for metal oxides and perovskites containing heavy metal elements, such as LAO/STO, plane-waves [14]–[16] are preferred for cubic-scaling eigensolvers. However, the number of plane waves is typically $N_{\text{PW}} > 10^8$ even for moderate-scale systems ($<$ 5K atoms) [14], [16], which explicitly requires 10^{24} floating point operations and 10^{16} bytes memory usage. Such high computational cost and large memory usage hinder in large systems (10K atoms). Therefore, it is challenging to develop accurate, efficient and low-scaling methods for large-scale plane-wave DFT calculations especially for complex metallic systems.

B. Conventional Solutions

There are mainly two solutions to accelerate large-scale DFT calculations. The most common solution is to develop efficient low-scaling eigensolvers for sparse Hamiltonian matrix within localized basis sets for reducing the computational cost and memory usage. The most notable examples are linear-scaling methods [13], such as divide-and-conquer (DAC) methods [17] and fragment molecular orbital (FMO) methods [18]. For large-scale complex and metallic systems, several efficient cubic-scaling eigensolvers, such as partial Rayleigh-Ritz [19] method and Chebyshev polynomial filtered subspace iteration (CheFSI) [20] method, have been proposed. Another solution is high performance computing (HPC) for accelerating large-scale DFT calculations on modern supercomputers. Based on these low-scaling methods implemented with localized basis sets, several highly efficient DFT software packages have been developed, such as CP2K [21], CONQUEST [22], OPENMX [23], ONETEP [24], and FHI-aims [25], and SIESTA [26], due to the local data communication of sparse Hamiltonian matrix (Fig. 1 (b)). However, most of linear-scaling DFT methods are limited to total energy calculations and strongly rely on the nearsightedness principle

in molecules, semiconductors and insulators (difficult for complex metallic systems). Furthermore, the numerical accuracy and computational efficiency of linear-scaling methods also depend on the parameters of localized basis sets, and are difficult to be improved systematically compared to complete basis sets (plane waves). But cubic-scaling plane-wave DFT codes, such as VASP [27] and QUANTUM ESPRESSO [15], require a large number of basis set for high accuracy and are difficult to achieve large-scale DFT calculations ($>$ 10K atoms) due to large all-to-all data communications of dense Hamiltonian matrix (Fig. 1 (b)). To date, standard plane-wave calculations are still limited to thousands of atoms in Qbox [14] and PWDFT [16].

IV. Current State of the Art

The rapid development of modern supercomputers enables the HPC as a powerful tool to accelerate large-scale DFT calculations. Several massively parallel DFT softwares have been developed, such as Qbox [14], LS3DF [28], RSDFT [29], CONQUEST [22], FHI-aims [25], and DFT-FE [11]. For example, large-scale linear-scaling DFT calculations (1M atoms) have been performed in CONQUEST [22] on the Cray supercomputer. But Qbox [14] as the only plane-wave code, wins the Gordon Bell prize of 2006, which can scale up to 128K CPU cores on the BlueGene/L supercomputer for small-scale systems (1K atoms). In particular, as one of Gordon Bell finalists in 2019, a finite-element method has been proposed in DFT-FE [11] to enable fast, accurate and massively parallel cubic-scaling (CheFSI) DFT calculations on large-scale metallic systems (11K Mg atoms).

A. DGDFT Methodology

The discontinuous Galerkin (DG) method [32] is a powerful tool to discretize partial differential equations, such as classical mechanics in the Hamilton-Jacobi equation [33] and fluid mechanics in the Navier-Stokes equation [34]. In 2012, Lin et al. first proposed [35] to adopt the DG method for KS equations and then developed discontinuous Galerkin density functional theory (DGDFT) [20], [30], [31], [36], which aims to combine the advantages of both atomic orbitals and plane waves (Fig. 1 (b)). DGDFT is unique in two aspects: an adaptive local basis (ALB) set [35] and a sparse direct solver (pole expansion and selected inversion (PEXSI) [37] method). DGDFT uses the ALB functions [30] to discretize the KS equations. As a new type of orthogonal, localized, and complete basis sets, the ALB functions are efficient and systematically improvable not only for total energy calculations, but also for atomic forces and vibrational frequencies [36], which can be used for a wide range of applications such as geometry optimization and molecular dynamics simulation. The ALB function is strictly localized in a subdomain in the real space, and it has two benefits: (1) In the basis construction step, we can deal with each subdomain element separately(as shown in Fig. 2(a)). (2) The corresponding DG Hamiltonian matrix keeps a block-sparse structure during the self-consistent field (SCF) iteration (Fig. 2(c)). The former helps us in introducing a divide-and-conquer method in generating ALB functions for the Hamiltonian matrix, while the latter is critical in utilizing the PEXSI solver to reduce the computational complexity. Differing from the linear scaling methods relying on the nearsightedness principle, DGDFT is universal for both semiconducting and metallic systems. With the sparse direct solver PEXSI, we realize low-scaling cost $\mathcal{O}(N^{1.5})$ for quasi 2D complex metallic systems with plane-wave precision in DGDFT. Therefore, DGDFT is able to take full advantage of the massive parallelism available on modern heterogeneous supercomputers for high-precision plane-wave DFT calculations. In particular, DGDFT has been demonstrated scaling to 128,000 CPU cores on Edison [30] and 8,519,680 processing cores (131,072 core groups) on Sunway TaihuLight [31] for performing large-scale DFT calculations on metallic systems ($>$ 10K atoms). Furthermore, DGDFT and PEXSI have been used to design multifunctional materials (Solar cells [38] and quantum dots [39]) and explain the new phenomenon in experiments (Photoactivity [40] and hydrogen evolution [41]).

The flowchart of the DGDFT methodology is given in Fig. 2(e). There are four time-consuming parts in DGDFT:

(1) Generating ALB functions: We first decompose the global domain into a series of subdomains (called elements in Fig. 2(a)). Then we can deal with each subdomain without communication and gather all basis sets to solve the global KS equations [35]. We partition a 2D graphene system with 180 carbon atoms (G180) into 16 equal-sized elements in a 2D 4×4 mesh. We define an extended element Q_6 composed of a central element E_6 and 8 neighboring elements. We solve small local KS equations $H^{Q_k}\phi_{k,j}^{Q_k} = \lambda_{k,j}^{Q_k}\phi_{k,j}^{Q_k}$ (H^{Q_k} and $\phi_{k,j}^{Q_k}$ are the local Hamiltonian and KS orbitals on the extended element Q_k). Then we restrict and truncate these eigenfunctions into the central element E_6 and obtain a new set of ALB functions $\{\phi_{k,j}\}_{j=1}^{J_b}$ (J_b is the number of ALB functions in each element). The ALB function plotted in Fig. 2(a) is strictly localized inside E_4 and is therefore discontinuous across the boundary of elements. But the global electron density looks continuous as shown in Fig. 2(b). Therefore, the ALB functions combine the advantages of both atomic orbitals (localization and truncation) and plane waves (orthogonality and completeness) [30] for global KS equations, resulting in a sparse-block DG Hamiltonian (Fig. 2(c)). Furthermore, the ALB functions are adaptively generated on-the-fly during each SCF. Such features make the DG Hamiltonian keep unchanged in a sparse correlation-tridiagonal structure during the SCF iterations even for complex metallic systems.

(2) Constructing DG Hamiltonian: For global KS equations, the KS orbitals $\psi_i(r)$ are expanded over ALB functions $\psi_i(r) = \sum_{k=1}^{M}\sum_{j=1}^{J_b} C_{i;k,j}\phi_{k,j}(r)$. Within the ALB functions, solving the global KS equations becomes a linear eigenvalue problem

$$\sum_{k,j} H^{\mathrm{DG}}_{k',j';k,j} C_{i;k,j} = \lambda_i C_{i;k',j'}, \tag{1}$$

where H^{DG} is a block-tridiagonal sparse DG Hamiltonian matrix. The nonzero matrix blocks correspond to interactions

TABLE I

PERFORMANCE COMPARISON OF MASSIVELY PARALLEL DFT SOFTWARE PACKAGES ON MODERN HETEROGENEOUS SUPERCOMPUTERS. THE DFT METHODS INCLUDE CUBIC-SCALING PLANE-WAVE (PW) AND LOCALIZED REAL-SPACE (RS) BASIS SETS, NUMERICAL ATOMIC ORBITALS (NAOS), AND LINEAR-SCALING (LS) SOLVERS. LS3DF, RSDFT, CONQUEST AND FHI-AIMS EXPLOIT LS EIGENSOLVERS. DGDFT AND DFT-FE CAN USE CHEFSI EIGENSOLVER. QBOX AND PWDFT ADOPT CUBIC-SCALING CONJUGATE GRADIENT EIGENSOLVERS (DAVIDSON AND PPCG).

Code	Year	Basis	Eigensolver	System	#atoms	Machine	Architecture	Scale	Peak (FLOPS)
LS3DF [28]	2008	PW	LS	ZnTeO	16K	BlueGene/R	PowerPC	131K cores	108T
RSDFT [29]	2011	RS	LS	Si	107K	K computer	SPARC64	442K cores	3.1P
CONQUEST [22]	2020	NAOs	LS	Si	1M	K computer	SPARC64	200K cores	\
FHI-aims [25]	2021	NAOs	LS	Polyethylene ($H[C_2H_4]_nH$)	500k	New Sunway	sw26010 pro	40M cores	468.5P
Qbox [14]	2006	PW	Davidson	Mo	1K	BlueGene/L	PowerPC	128K cores	207T
DGDFT [30]	2015	DG-PW	PEXSI	Phosphorene	14K	Edison	Intel Xeon	128K cores	\
PWDFT [16]	2017	PW	PPCG	Si, H_2O, AlSi	5K	Cori	Intel Xeon	8K cores	\
DGDFT [20]	2018	DG-PW	CheFSI	Lithium battery	28K	Cori	Intel Xeon	39K cores	\
DFT-FE [11]	2019	RS-PW	CheFSI	Mg	11K	Summit	V100	23K GPUs	46P
DGDFT [31]	2021	DG-PW	CheFSI	Graphene	13K	Sunway TaihuLight	sw26010	8.5M cores	\
DGDFT (This work)	2022	DG-PW	PEXSI	MATBG, Li/Na, Cu/G/Cu, LAO/STO	2.5M	New Sunway	sw26010 pro	40M cores	64.0P

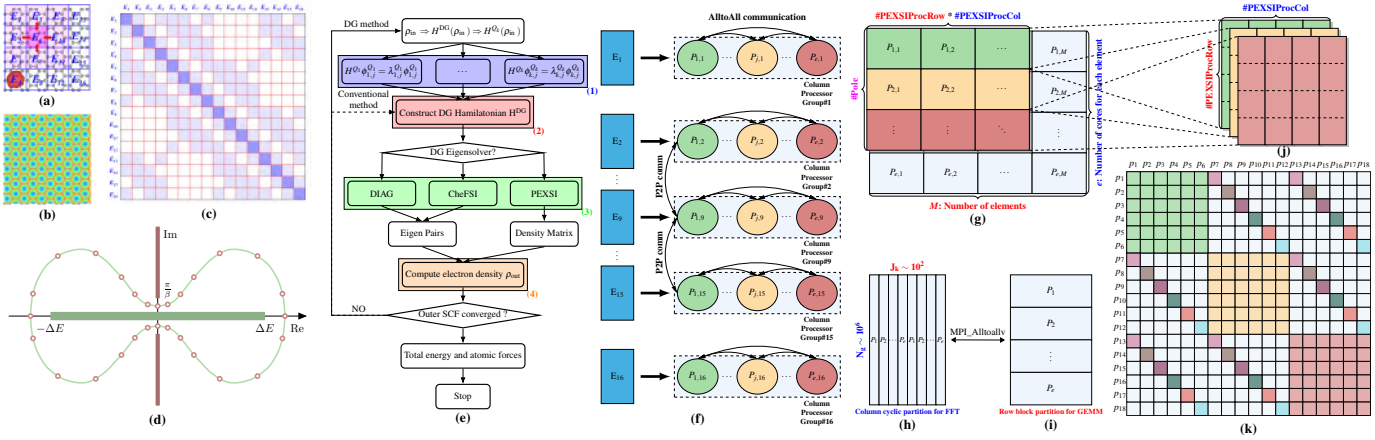

Fig. 2. The main steps in DGDFT. (a) A 2D graphene system (G180) partitioned into 16 (4×4) elements. (b) The electron density. (c) The block-sparse DG Hamiltonian matrix (G180). (d) Placement of poles used in PEXSI. (e) Flowchart of DGDFT. (f) Two-level parallelization in DGDFT, including (g) 2D MPI grid, (h) 1D band parallelization with column cyclic partition (FFT) and (i) 1D grid parallelization with row block partition (GEMM). (j) The processor grid in PEXSI. (k) Data communication heat map in DGDFT and PEXSI, the same color denotes a collective communication group.

between neighboring elements are shown in Fig. 2(c).

(3) Diagonalizing DG Hamiltonian: The conventional method is to directly diagonalize the DG Hamiltonian by using the PDSYEVD subroutine in ScaLAPACK (referred to as DIAG). We present two efficient methods to solve this eigenvalue problem, including the PEXSI [37] and CheFSI [20] algorithms. The CheFSI algorithm is a cubic-scaling diagonalization method (also used in DFT-FE [11]) and has been demonstrated in DGDFT only scaling to 131,072 core groups on the Sunway TaihuLight supercomputer [31].

In DGDFT, the sparse direct solver PEXSI [37] is designed to exploit the sparsity of the Hamiltonian once the DG Hamiltonian is constructed. PEXSI expands the density matrix with a linear combination of Green's functions

$$P \approx \sum_{l=1}^{Q} \mathrm{Im}\left[\omega_l (H^{\mathrm{DG}} - z_l I)^{-1}\right], \qquad (2)$$

where the integration weights ω_l and shifts z_l are chosen carefully so that the number of expansion terms Q is proportional to $\log(\beta \Delta E)$, where β is the inverse temperature and ΔE is the spectrum width of H^{DG} (Fig. 2(d)). In practice, 40 to 120 poles are already enough to produce accurate results. The evaluation of one pole can be parallelized up to $10,000$ MPI tasks, and different poles can be embarrassingly parallelized. Since only the nonzero blocks of $(H^{\mathrm{DG}} - z_l I)^{-1}$ are required to evaluate the density matrix, an efficient selected inversion algorithm can be used in solving the problem. In PEXSI, we first perform parallel sparse LU factorization of the shifted DG Hamiltonians $((H - z_l I))$ using state-of-the-art LU factorization package SuperLU_DIST [42], and then perform selected inversion on the lower triangular matrix L with PSelInv [43]. FHI-aims [25] and CP2K [21] also use the PEXSI eigensolver, but their implementations of PEXSI is currently not highly scalable ($<$ 10K CPU cores) within atomic basis sets.

(4) Computing electron density: After solving the eigenvalue problem, the electron density, total energy [35] and atomic forces [36], can be individually computed on each element.

B. Two-Level Parallelization Strategy

We remark that a two-level parallelization strategy is utilized in DGDFT as given in Fig. 2(f). The MPI tasks are arranged in 2D grids, and both intra-element and Hamiltonian matrix

(inter-element) are distributed and parallelized in the same fashion (Fig. 2(g)). Note that the communication patterns of both intra- and inter-element parallelization are fixed due to the fact that DG Hamiltonian matrix structure remains unchanged throughout the SCF iterations. We combine the techniques of OS.write, OS.read, container and serialization in C++, to efficiently simplify the communication process (Fig. 2(g) and (k)) and improve the communication efficiency in DGDFT.

The first level is the intra-element column parallelization for evaluating each extended element in inner SCF iterations. As shown in Fig. 2(g), each column of MPI processes holds an individual extended element, and constructs the corresponding fragment of the DG Hamiltonian matrix over ALB functions. For local KS equations, we implement a plane-wave module named PWDFT [16] by using less than $P_e < 200$ cores on each extended element. The data layout of local KS orbitals $\Phi^{Q_k} \in \mathbb{R}^{N_r^{Q_k} \times J_b}$ is rearranged between column cyclic partition (Fig. 2(h)) and row block partition (Fig. 2(i)) for efficiently performing FFT and GEMM, respectively, with a local MPI_Alltoallv for data format conversion.

The second level is the inter-element row parallelization for outer SCF iterations, where DG Hamiltonian matrix H_{DG} is diagonalized and most computationally intensive. Three methods are implemented in DGDFT: Direct diagonalization (DIAG), CheFSI and PEXSI [20], [30]. From such a two-level parallelization strategy, the minimum number $N_{\min}$ and maximum number $N_{\max}$ of MPI processes used in DGDFT are computed as $N_{\min} = M$ and $N_{\max} = MJ_b = N_b$. By using this two-level parallelization strategy, DGDFT is highly scalable on 100K cores on heterogeneous supercomputers [30], [31].

In PEXSI, each pole requires a 2D grid of size (#PEXSIProcRow) $\times$ (#PEXSIProcCol) as shown in Fig. 2(j). This parallelization scheme restricts each pole to be evaluated within M MPI tasks, and at most P_e poles can be processed in parallel, where P_e is the number of MPIs in evaluating a single element (Fig. 2(g)). This parallelization scheme will be improved in Sec. V. In PEXSI, multiple (N_μ) Fermi operators are evaluated in parallel to dynamically keep a rigorous upper and lower bound of the true chemical potential, so that the chemical potential reaches convergence along with the SCF iteration [44]. Thus the total number of poles reaches $N_\mu \times N_{pole}$. We remark that the baseline DGDFT and PEXSI (SuperLU_DIST7.2 and PSelInv) are linked with many-core accelerated libraries.

V. Innovations

Our main contribution is a massively parallel DGDFT, which can scale up to 102,400 computing nodes on new Sunway to study complex metallic heterostructures (2.5M atoms), reaching mesoscale for the first time. This is mainly achieved by adapting PEXSI to take advantage of the block-sparse Hamiltonian matrix. By using PEXSI, we overcome the cubic scaling of conventional eigensolvers, and reduce the computational complexity to $O(N^{1.5})$ and $O(N^2)$ for two- and three-dimensional systems, respectively. We optimize the computationally intensive parts with both algorithmic and system innovations, and our optimized sparse direct solver PEXSI can reach 64.0 PFLOPS ($\sim$5.0% of the peak) on almost entire Sunway supercomputer. The corresponding time-to-solution takes 931.0 seconds per SCF for a system with 2.5-million atoms, which is estimated to be more than 1000 times faster than the current state-of-the-art.

A. Algorithmic Innovation

In this section, we mainly focus on the optimization of the sparse direct solver PEXSI, which is composed of two parts: LU factorization (**SuperLU_DIST**) and parallel selected inversion (**PSelInv**) as discussed in Sec. IV. We remark that optimizations of the sparse matrix operations, such as SPMV, LU factorization, are notoriously difficult on modern supercomputers due to their low arithmetic intensity relative to the amount of data movement and indirect addressing. These features are often in conflict with the computation/bandwidth intensive nature of many-core architecture. For example, the current state-of-the-art sparse-matrix benchmark HPCG records can only reach 16 PFLOPS, 2.9 PFLOPS and 5.9 PFLOPS on Fugaku, Summit and new Sunway supercomputers, respectively. These correspond to of 3.6%, 1.5% and 0.5% of the peak performance, respectively [45], [46]. To harness the computing power provided by many-core architecture, the following algorithmic innovations in terms of data locality, granularity and parallelization scheme are adopted:

- We introduce a specific blocked data scheme to take advantage of the structured sparse pattern of DG Hamiltonian matrix. This greatly boosts the performance of the PEXSI solver on the new sw26010 pro chip.
- We carefully adjust the granularity of supernodal representation of PEXSI solver to adapt to both the sparse pattern of DG Hamiltonian and hardware characteristics of computation, bandwidth and network of the new Sunway architecture.
- We propose a new parallelization scheme to decouple the intra-element and inter-element parallelization to improve the numerical efficiency of the DGDFT code.

1) Blocked LU factors for Hermitian matrix: In this subsection, we focus on the optimization of SuperLU_DIST, and similar optimization techniques can be applied to selected inversion **PSelInv**. The key steps of LU factorization for a generalized sparse matrix H stored with supernodal representation and distributed in a two-dimensional block cyclic scheme are illustrated in Fig. 3(a). Note that we skip the look-ahead of L and U factors for simplicity reasons. As shown in Fig. 3(a), first the L panel is factorized (LFactor), then L panel is broadcast to the right. Next the U panel is factorized by TRSV (UPFactor), followed by the downward broadcast of U panel. The last step is the computationally intensive Schur complement update (SchurUpdate) that is carried out via matrix-matrix multiplication. Note that maintaining data locality is a fundamental challenge in implementing sparse matrix linear algebra due to the unstructured computation and indirect memory access, especially on many-core architectures.

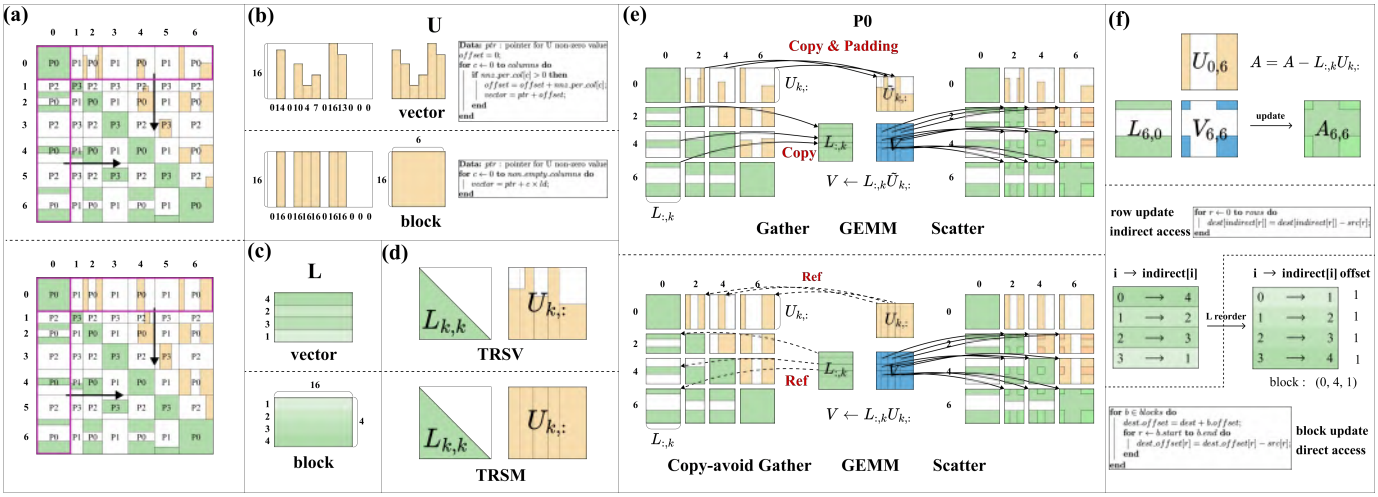

Fig. 3. The blocked LU factors and corresponding optimization steps in optimized SuperLU_DIST. (a) Generalized sparse matrix(upper) and Hamiltonian matrix generated in DGDFT (below). (b) previous (upper) and blocked (below) data format for U factor. (c) Previous (upper) and blocked (below) data format for L factor. (d) U panel factorization, upper: baseline, below: optimized. (e) SchurUpdate, upper: baseline, below: optimized. (f) SchurUpdate Scatter, left: baseline, right: optimized.

One key observation is that the Hamiltonian matrix in DGDFT is Hermitian and nonzero matrix elements are in structured dense blocks, as shown in Fig. 3(a) (below). Therefore, we introduce a blocked format for LU factors to enhance the data locality in both computation and memory access. Our idea improves the data parallelism by storing both L and U factors in blocks rather than vectors, as shown in Fig. 3(b) and (c). In the new data format, the L factors (Fig. 3(c)) are reorganized from random order to a continuous storage, which will benefit the scatter of SchurUpdate as will be discussed below. Fig. 3(b) shows the comparison between previous vector and the new block U data formats for the U factors. Next we show how to improve the performance of LU factorization by taking advantage of the new block data format.

A. SchurUpdate gather. First, we optimize the gather of L and U factors in SchurUpdate. The three steps of the SchurUpdate for a generalized sparse matrix H are shown in Fig. 3(e) (upper): (1) The nonzero blocks of both $L_{:,k}$ and $U_{k,:}$ are gathered to form extra dense matrix $L_{:,k}$ and $\tilde{U}_{k,:}$. Note that in the generalized implementation, extra zeros are filled in to generate a dense matrix $\tilde{U}_{k,:}$, as illustrated in Fig. 3(c). (2) GEMM subroutine is invoked to perform dense matrix-matrix multiplication on $L_{:,k}$ and $U_{k,:}$ to get V. (3) Schur complement is updated by scattering back matrix V. Note that in the optimized code, step 2 is accelerated by calling our optimized SWBLAS library specially customized for small matrix size in DGDFT (Sec. V-B1). Then the gathered operations of both L and U can be avoided by keeping the corresponding address and size of $L_{:,k}$ and $U_{k,:}$ factors in our new block data format and the zero fill-in step can be eliminated, as shown in Fig. 3(e)(below).

B. SchurUpdate scatter The indirect addressing of matrix V in the SchurUpdate scatter is optimized to direct memory access in the new block data format. As shown in Fig. 3(f), the matrix V keeps the same order of the L factors. Previously, the rows of the L factor are arranged in a random order, indirect addressing is required to map between V and the corresponding position in matrix A(Fig. 3(f)). Our optimized SuperLU_DIST, however, can skip indirect addressing due to the block format of the V matrix. A detailed pseudocode is shown in Fig. 3(f) (below).

C. UPFactor. With U factors stored in blocks, we can improve the efficiency of U panel factorization by replacing TRSV with TRSM, as shown in Fig. 3(d). This significantly improves the performance of the U panel factorization by a factor of 100 on the new Sunway platform due to data reuse and more computationally intensive of TRSM subroutine. Consequently, an overall speedup of about 3 times can be achieved (see Sec. VII) compared to the baseline, and we remark that this is mainly contributed by the novel block data format of U factors.

D. SNodeFactor. The factorization of the diagonal supernode is implemented in a sequential way in the baseline SuperLU_DIST, and it is accelerated with our optimized kernel, which incorporates optimization techniques such as fused panel factorization with data residing in the local data memory (LDM), recursive blocked factorization and balanced data distribution. Our optimized supernode factorization further reduces the computation time, and leads to a 7% performance boost.

Our optimized SuperLU_DIST can be $\sim$ 7 times faster than the baseline. This is achieved by adapting the new block data format to improve the data locality in both memory access and computation. We find that both computation time and communication time are reduced by 83% and 90%, respectively (Sec. VII). We remark that the communication time reduction is due to the reduction of the MPI wait time

caused by data dependence, and our optimized SuperLU_DIST does not change the data dependence or communication graph compared to the baseline. The optimizations above can also be applied to the selected inversion **PSelInv**. For example, the indirect addressing of the Schur complement in **PSelInv** can also be improved by applying blocked memory access. We remark that our optimization techniques above can also be applied to all sparse Hermitian matrices and benefit other scientific computing fields with sufficiently regular sparsity patterns.

2) Supernode granularity: The performance of the PEXSI solver is highly related to the supernodal representation. The choice of the supernode size not only depends on the sparsity pattern of the matrix, but also takes into account of the hardware characteristics such as the computational power, memory bandwidth and network latency/bandwidth, etc. Thus the default supernode size represents a trade-off between hardware limitations and sparsity patterns of the corresponding matrix. For example, the supernode size changes from 128 in SuperLU_DIST 6.1 to 256 in SuperLU_DIST 7.2 to embrace the evolution of many-core architecture. Since the Hamiltonian matrix generated from ALB functions has special block-structure patterns and the computation/bandwidth/network on the new Sunway supercomputer are unique, it is important to re-evaluate the best supernode size to exploit the performance provided by the new Sunway platform.

We perform a study of the performance of the PEXSI solver with a typical Hamiltonian matrix generated from $103,680$-atom graphene system. The dimensions of the Hamiltonian matrix is $552,960$, and number of nonzeros is $919,756,800$ (0.3% sparse), respectively. The corresponding performance of the PEXSI solver w.r.t. the supernode size is shown in Table II. We find that the best performance is achieved when the supernode size is 320, which is higher than default of SuperLU_DIST. 7.2 (256). This is due to the fact the sw26010 pro has a higher FLOPS/Byte ratio compared to other many-core architectures such as NVIDIA GPU. In the tests shown at Sec. VII, we set the supernode size to 320 to achieve the best performance on the new Sunway platform.

TABLE II
PERFORMANCE (TFLOPS) OF PEXSI SOLVER W.R.T. THE SUPERNODE SIZE ON 24 COMPUTING NODES FOR HAMILTONIAN IN GRAPHENE (103,680 ATOMS).

Supernode size	64	128	256	320	384	448	512
G103680	5.3	10.4	12.8	13.0	12.1	12.7	12.0

3) Decouple intra- and inter-element parallelization : Previously, both generating ALBs on extended elements and using PEXSI to compute the electron density share the same two-dimensional parallelization scheme, as shown in Fig. 4(a). M extended elements are parallelized among the columns of MPI tasks, and sparse Hamiltonian matrix $H^{\mathrm{DG}} - z_l I$ is constructed and embarrassingly parallelized by rows. However, this parallelization scheme is not optimal for achieving high efficiency in massively parallel computing. As shown in Fig. 4(a), when using 480 MPI tasks to evaluate a system of 120 elements E_i, a 4×120 MPI grid is formed. Although the extended elements are efficiently parallelized, the $N_{pole} \times N_{\mu}$ poles can only be parallelized in groups of 4. When $N_{\mu} = 2$ and $N_{pole} = 120$, each row MPI tasks has to sequentially process 60 poles.

A new parallelization scheme is developed to efficiently exploit the computing power in new Sunway. The data distribution of extended elements and PEXSI are decoupled to resolve the parallelization dilemma (Fig. 4(b)). The MPI tasks are first organized into M groups to generate ALBs on M extended elements. Then the Hamiltonian matrix H is constructed and re-distributed into $N_{\mu} \times N_{pole}$ groups to exploit the parallelism of poles. For example, the 120×4 grid will be reorganized into 240×2 grid to parallelize the 240 poles ($N_{mu} = 2$ and $N_{pole} = 120$). In Sec. VII, the $N_{\mu} \times N_{pole}$ groups are always parallelized first to fully exploit the power of Sunway.

B. System Innovation

1) Customized libraries: The xMath-BLAS library is customized towards the small-size block matrix in DGDFT. The GEMM function is modified in the following aspects: (1) customized blocking and data-thread mapping, (2) specifically designed software pipeline for matrices of different sizes and shapes, and (3) fine-tuned assembly kernels with instructions carefully arranged to maximize instruction level parallelism. Compared to the previously more generalized SWBLAS, our optimized version is 1.2 times faster on a non-transposed DGEMM for a tall-and-skinny matrix-matrix multiplication with M = 7388, N = 120, and K = 120. The xMath-LAPACK library is also customized for the non-blocked factorization with fully fused kernels. By keeping the panel data in the LDM of the CPEs, we are able to not only reduce the data movement overhead significantly by removing all the intermediate DMA operations, but also implement highly efficient SIMD kernels. The xMath-FFT [47] is also specially optimized for the DGDFT code. To construct the ALB basis, wavefunctions are mapped from the real space to the reciprocal space with many FFT operations. Note that the wavefunctions have conjugate symmetry at Γ point. Our customized xMath-FFT library takes advantage of the physical symmetry and save half of the communication and computation.

To make the transposition between uniform grids and Legendre-Gauss-Lobatto (LGL) grids [48] efficiently, we adopt a high performance tensor transpose library provided by BaGuaLu [49]. In this way, the grids in the middle direction can be transposed to the outer direction, thus enabling us to perform level-3 BLAS function instead of level-2 function.

2) Network Optimization: In the previous implementation of (**PSelInv**), restricted collective communication such as broadcast of $\mathcal{L}$ is implemented via asynchronous point-to-point MPI communication in flat tree (FTree), binary tree (BTree) or shifted binary tree (SBTree) modes [43], as shown in Fig. 4(c-e). Note that SBTree has greatly improved the load balance of **PSelInv** by heuristically preventing one MPI task from

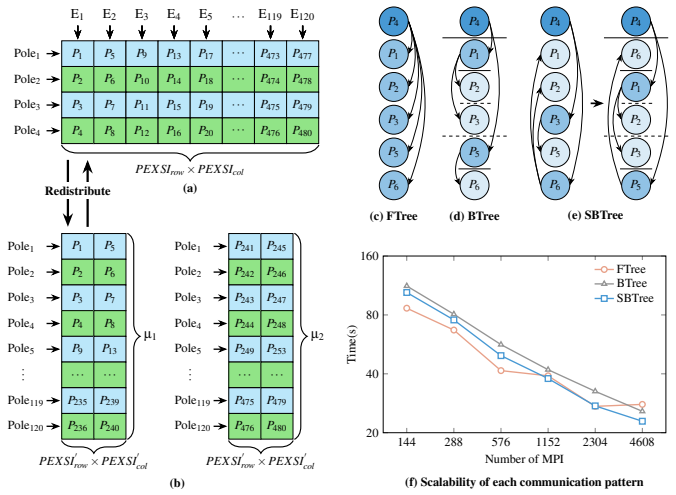

Fig. 4. Parallelization scheme and network optimization. (a) Previous and (b) new parallelization scheme. Communication pattern of (c) FTree, (d) BTree, and (e) SBTree in **PSelInv**. (f) Scalability of three tree-based modes on new Sunway.

participating in multiple collective communications, hence improving the scalability of selected inversion. In particular, testing results on Edison supercomputer [43] equipped with a Dragonfly network topology show that SBTree outperforms others when more than 16 MPI tasks are used. However, the new Sunway supercomputer employs a fat tree network topology, and scaling behavior is quite different from Edison. As shown in Fig. 4(f), we find that naive FTree implementation can be 20% faster than the shifted binary tree when using 144 MPI tasks, and SBTree outperforms FTree when more than $4,608$ MPIs are used. The enhanced scalability of FTree implementation on the new Sunway is due to the fact that (a) network bandwidth of a single Sunway processor is 200Gps and not easily congested like Edison, (b) fat-tree topology is more suitable for flat-tree communication compared to the dragonfly topology.

3) Data Structure, Setup and I/O: To construct the DG Hamiltonian, we constantly need to look up the pseudopotential of a given atom. The original implementation of DGDFT, which targets general-purpose CPUs, implements the look-up through std::map from C++ Standard Template Library, which is implemented through trees. However, it suffers from poor performance due to its indirect access, especially when the system expends. Moreover, the CPEs on sw26010 pro do not support allocating from the main memory shared across cores, making the std::map inaccessible on CPEs and hindering the acceleration of pseudopotential lookup. To address the above issues, we exploit the fact that there are practically no more than 200 different types of atoms and thus replace the std::map with a small array. By indexing the array with the atom number, we can acquire the pseudopotential in one direct access. Such a mechanism is also more CPE-friendly: the LD Cache makes loading the pseudopotential efficient in case of direct indexing. These methods greatly boost the performance of the DG Hamultionian step by a factor of 10 compared to our previous design.

Previously the I/O of the atomic coordinates, pseudopotential file and control parameters are read in by all MPI tasks and compared in a self-contained routine named electronic structure data format (ESDF). To reduce the I/O contention and setup time, we employ a key-value format and read-in-and-broadcast mechanism. It greatly improves the I/O contention. The corresponding setup and I/O time is reduced from $16,481.0$ to 5.9 seconds when using $165,888$ MPI tasks for $207,360$-atom system.

TABLE III
INPUT TIME COMPARISON (IN SECOND) OF ESDF FORMAT AND THE K-V FORMAT.

#atom	2,880	6,480	11,520	46,080	207,360	2,457,600
ESDF	4.9	43.7	118.2	1020.0	16481.0	-
Key-value	0.5	0.5	0.8	1.7	5.9	68.2

VI. How Performance Was Measured

A. Physical and Chemical Test Systems

We construct atomic structures for four types of metallic heterostructures, including Li/Na, Cu/G/Cu, LAO/STO and graphene systems (MG, BG and MATBG), with negligible lattice mismatch between corresponding two components. Both the scale ($>$ 100K atoms) and complexity (metallic systems) of these systems make it very difficult for conventional DFT calculations, for example, linear-scaling methods implemented within localized atomic basis sets lose their efficacy since nearsightedness no longer works for metallic systems. Furthermore, the cubic-scaling and quadratic memory requirement of conventional solvers [11] limit both computational time and system size accessible on modern supercomputers. We list the key parameters of these systems in Table IV. The norm-conserving pseudo-potentials [50] (Hartwigsen-Goedecker-Hutter (HGH) and SG15 Optimized Norm-Conserving Vanderbilt (ONCV)) are supported by DGDFT, and we use exchange-correlation functional of local density approximation of Goedecker-Teter-Hutter (LDA-PZ) [51] to describe the electronic structures of these systems.

TABLE IV
COMPUTATIONAL PARAMETERS OF THE LARGEST SYSTEMS FOR MG, BG, LI/NA, CU/G/CU AND LAO/STO IN DGDFT, INCLUDING THE NUMBER OF ATOMS N_a, THE ENERGY CUTOFF E_{CUT} (HA), THE NUMBER OF GRID POINTS N_r, THE NUMBER OF THE ALB FUNCTIONS J_b IN EACH ELEMENT, THE NUMBER OF ELEMENTS M, THE NUMBER OF TOTAL ALB FUNCTIONS N_b, THE MINIMUM ($N_{\min} = M$) AND MAXIMUM ($N_{\max} = MJ_b = N_b$) NUMBERS OF MPI PROCESSES.

Systems	N_a	E_{cut}	N_r	J_b	M	N_b
MG	103,680	30.0	233,472,000	120	18,432	2,211,840
BG	207,360	30.0	212,336,640	120	18,432	2,211,840
Li/Na	2,508,800	10.0	1,560,674,304	120	28,224	3,386,880
Cu/G/Cu	115,200	20.0	92,897,280	120	13,824	1,658,880
LAO/STO	128,000	30.0	174,587,904	120	9,216	1,105,920

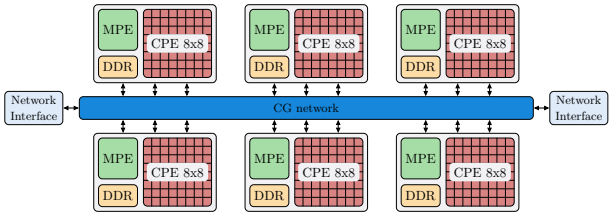

Fig. 5. Architecture of one computing node in the new Sunway supercomputer.

B. Systems and Environment

All numerical tests are performed on new Sunway supercomputer, which consists of 107,520 computing nodes with a theoretical peak performance of 1.5 EFLOPS. The computing nodes are connected via a fat-tree network topology and every 256 nodes constitute a supernode (Fig. 5). Each node is equipped with one sw26010 pro chip and 96 GB memory that can be divided into 6 core groups (CGs, 16 GB memory each CG). Each CG has 1 Manage Processing Element (MPE) and 64 Computing Processing Elements (CPEs) organized as 8×8 grid in Cmesh network. Each CPE has its own instruction cache and data storage that can be configured as LDM or Local Data Cache (LD Cache). Data transfer between main memory and LDM, main memory and LD Cache are achieved by direct memory access (DMA) and load/store instructions respectively. Both MPE and CPE use in-house SW64 instruction set.

In the optimized DGDFT code, we embrace an MPI+SACA programming model to have explicitly control over CPEs and 6 MPI tasks are launched (each binds to one CG) on each computing node to fully exploit the computing power of the sw26010 pro. Fortran compiler swgfortran, C++ compiler swg++, MPI wrapper compiler mpic++ and many-core programming extension SWUC [52] are used. We customize the FFT, BLAS, and LAPACK libraries for better performance, as discussed in Sec. V-B1.

C. Measurement

The total floating point operations (FLOPs) of the performed calculations is collected via counting the effective FLOPs, which is less than the actual FLOPs executed in PEXSI and DGDFT. The following criteria are used to measure the performance of the DGDFT.

- **Time-to-solution**, defined as per SCF time, the wall clock time used for calculating a single SCF loop. The "per SCF time" includes all the time used in a single SCF loop (IO included). Setup time, such as the setup of the system and MPI initialization and finalization, is not included.
- **Peak performance**, defined as $\frac{\text{PEXSI total FLOPs}}{\text{PEXSI solver time}}$. We remark that PEXSI is a sparse direct solver and notoriously difficult to optimize on many-core architecture. For example,the current state-of-the-art HPCG can only achieve 27 PFLOPS on the entire Sunway platform.
- **Sustained performance**, defined as $\frac{\text{PEXSI total FLOPs}}{\text{total wall clock time}}$. The "total wall clock time" includes the whole application running time (including IO).

D. Numerical Accuracy

Because the ALB functions can systematically approach the complete basis limit, the numerical accuracy of DGDFT only depends on the energy cutoff and the number of the ALB functions, and is comparable to standard plane-wave DFT calculations (QUANTUM ESPRESSO [15]), which has already been validated in our previous works [20], [30], [36]. In our experiments, we set the energy cutoff and the number of the ALB functions are chosen to ensure, for example, the total energy of G180 is accurate to 10^{-4} Ha/atom and atomic forces are accurate to 10^{-3} Ha/Bohr respectively.

VII. Performance Results

We show the performance improvement of SuperLU_DIST in Sec. VII-A, and the overall performance of DGDFT will be discussed in Sec. VII-B. The baseline codes for comparison are DGDFT and PEXSI linked with many-core libraries such as FFT, BLAS, and LAPACK etc.

A. Small System

In this subsection, we report the performance improvement by adapting the optimization techniques introduced in Sec. V. We focus on the performance of the SuperLU_DIST package for simplicity reasons, and we remark that similar optimizations are also applied to **PSelInv**. The benchmark Hamiltonian matrices are generated from two typical metallic systems: one $12,960$-atom graphene and one $86,400$-atom Li/Na system. Both Hamiltonian matrices share the same dimension of $69,120$ and the number of nonzero elements is $3,456,000$ (sparsity: 0.07%). A minimum of 4 MPI tasks(2×2 grid, 4 CGs) are used to store the matrices. The baseline for comparison is SuperLU_DIST 7.2 accelerated with SWBLAS on CPEs. We remark that the total FLOPs of the LU factorization for Li/Na (34.1 TFLOPs) is about 2 times that of the bilayer graphene (14.8 TFLOPs) system due to various fill-in behavior in the symbolic factorization.

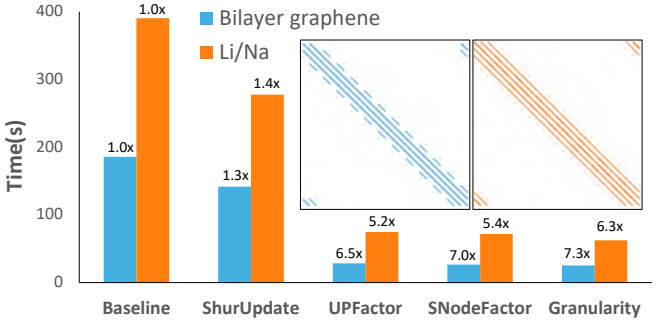

Fig. 6. Step-by-step improvement of the optimizations for bilayer graphene (12,960 atoms) and Li/Na (86,400 atoms).

A. SchurUpdate. In the optimized SuperLU_DIST, the "copy-and-fill-in" of gathering L and U factors is substituted with pointer reference (Fig. 3(e)), thus the corresponding data-movement time is eliminated. Note that the time for "Gather" takes no time (0%) in the final results, thus it is not included in Table V. Meanwhile, the scatter of the result matrix V (Fig. 3(e)) is optimized from indirect addressing to consecutively block memory access. The optimization of the SchurUpdate leads to a speedup factor of 1.3 and 1.4 for bilayer graphene and Li/Na systems, respectively.

B. UPFactor. Now we focus on the optimization of the U panel factorization since it becomes the dominant part and takes $32\% - 40\%$ of total time for the graphene and Li/Na Hamiltonian. We can exploit the block structure of the U panel by substituting vector-wise factorization (TRSV) with matrix factorization (TRSM), and the results show that a speedup factor of ~ 100 can be achieved. Compared to the baseline, the optimized version now reaches an overall speedup factor of 6.5 and 5.2 for bilayer graphene and Li/Na systems, respectively. We remark that the speedup of U panel factorization not only reduces the computational time, but also reduces the MPI wait time of other ranks, as discussed in Sec. V-A.

C. SNodeFactor. As detailed in Sec. V-A, factorization of the diagonal supernode is optimized by utilizing customized kernels. A speedup factor of 1.4 is achieved since the diagonal supernode is small (Dimension: 256). Compared to the baseline, we find that the overall speedup factor is increased to 7.0 and 5.4 for bilayer graphene and Li/Na systems, respectively.

D. Granularity. As discussed in Sec. V-A, an optimal choice of the supernode size is used in our optimized code to replace the default parameter in the baseline. And as shown in Fig. 6, the overall speedup factors increase to 7.3 and 6.3 for Li/Na and graphene matrices, respectively.

Finally we achieve 0.54 and 0.44 TFLOPS for the LU factorization of the $12,960$-atom bilayer graphene and the $86,400$-atom Li/Na matrices on 4 CGs, reaching 5.7% and 4.7% of the theoretical peak, respectively. A percentage of time consumed by critical kernels in total execution time is shown at Table V. We find that GEMM operation takes more time in the graphene system (28% percent) than Li/Na system (24%), and communication time takes more time in the Li/Na (38%) than bilayer graphene (22%). Note that the percentage of communication time for the Li/Na system grows fast in strong scaling tests, and peak performance drops from 5% to 1.2% when scaling from 4 to 64 MPI tasks. We remark that even the GEMM operations in SchurUpdate are still memory-bound due to the small block size(average size of the matrix is: 2019, 256, 2019.). The bottleneck resource is memory bandwidth, and profiling results show that the optimized PEXSI reaches 80% of the theoretical limit of memory bandwidth.

B. Scaling towards Large-Scale Systems

A. Strong Scaling. Fig. 7 shows the strong scaling of "time-to-solution" of graphene (G2880 and G72000) ranging from 576 CGs to 576,000 CGs. First we show that the speedup of

TABLE V
PERCENTAGE OF MAJOR PARTS IN THE OPTIMIZED SUPERLU_DIST WITH 4 MPI TASKS FOR BILAYER GRAPHENE (BG) AND LI/NA WITH 12,960 AND 86,400 ATOMS, RESPECTIVELY.

System	GEMM	UPFactor	LFactor	Comm	LookAhead	Scatter
BG	28%	2%	9%	22%	20%	19%
Li/Na	24%	1%	4%	38%	17%	16%

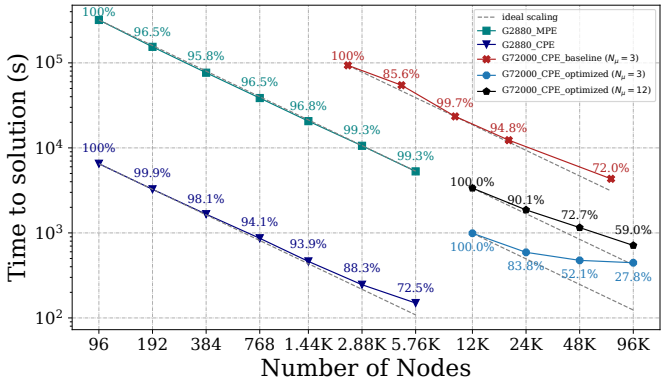

Fig. 7. Strong scaling of two graphene systems (G2880 and G72000) with MPE, CPE (baseline) and optimized CPE versions of DGDFT.

DGDFT on CPE (baseline) over MPE is about 50 times with a 2880-atom graphene system due to the accelerated many-core libraries. In all following tests, we use the CPE version as baseline(Fig. 7 red line). Testing results on a 72,000-atom graphene system (N_μ=3) show that our optimized DGDFT can be >10 times faster than the baseline. We remark that the parallelization scheme in the baseline is sub-optimal in exploiting the parallelism of poles, thus leading to a worse time-to-solution (but better parallel efficiency due to worse peak performance). As shown in Fig. 7, the optimized DGDFT has a parallel efficiency of 27.8% when scaling from 72,000 to 576,000 MPIs. The corresponding peak performance of PEXSI reaches 9.58 PFLOPS (5.7% of peak) to 38.95 PFLOPS (2.9% of peak). We remark that the time-to-solution is 445.8 seconds per SCF for G72000 when using high-precision parameters (E_{cut} = 30.0 Ha and N_b/N_a = 24 ALB functions), which contains $432,000$ electrons and is 3.43 times larger than the current state-of-the-art [11] ($105,08$ Mg atoms, $126,096$ electrons, 142.7 seconds per SCF, peak 46 PFLOPS). Based on the cubic scaling of conventional DFT methods, an estimation of the time-to-solution for 72,000-atom graphene (432K electrons) is $3.43^3 \times 142.7 \approx 5738$ seconds and our optimized DGDFT can be 12.9 times faster than the current state-of-the-art.

The black line in Fig. 7 show the scaling of DGDFT in a computationally intensive scenario when 12 chemical potential points are used (4 times FLOPs than N_μ=3). Each pole's evaluation is fixed on a 20×20 processor grid, and strong scaling is carried out by parallelizing the poles. We find that the parallel efficiency is 61% when scaling from 72,000 to 576,000 MPIs, and the peak performance is maintained at

$\sim 5.7\%$.

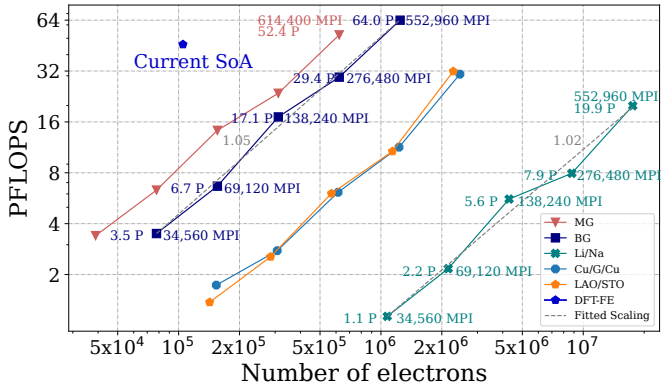

Fig. 8. Weak scaling of PEXSI w.r.t. the number of electrons for MG, BG, Li/Na, Cu/G/Cu and LAO/STO.

B. Weak Scaling. Fig. 8 shows the weak scaling of PEXSI w.r.t the number of electrons for four complex metallic systems (Li/Na, Cu/G/Cu, LAO/STO, MG and BG) . Note that PEXSI is the most computationally intensive part and theoretically scales as $\mathcal{O}(N^{1.5})$ for 2D systems. The performance of PEXSI scales almost perfectly up to $552,960$ MPI tasks for both BG with high-precision parameters (E_{cut} = 30.0 Ha and N_b/N_a = 10.7 basis functions per atom) and Li/Na with low-precision parameters (E_{cut} = 10.0 Ha and N_b/N_a = 1.3 basis functions per atom). This indicates that PEXSI solver can reach higher FLOPS with no intrinsic obstacles when solving a bigger problem on future HPC platform. Note that Li/Na reaches lower peak than BG due to more MPI communication as discussed in Sec. VII-A. **For a $207,360$-atom bilayer graphene, our optimized DGDFT can achieve a time-to-solution of 931.0 seconds per SCF iteration, and PEXSI can reach 64.0 PFLOPS ($\sim$5.0% of theoretical peak) on $92,160$ computing nodes (36 million cores), which is unprecedented for sparse direct solvers.** As far as we know, this is for the first time a sparse direct solver, which is notoriously difficult on modern many-core architecture, is successfully employed in extreme-scale DFT calculations. As a comparison, the best performance of sparse matrix benchmark HPCG can achieve 5 PFLOPS (27 PFLOPS if breaking the HPCG rules) on new Sunway [46]. We remark that a sustained performance (Setup and I/O time included) of 23 PFLOPS (1.8% of the peak) is achieved for 207, 360-atom bilayer graphene on $92,160$ computing nodes.

The arithmetic intensity of our sparse direct solver PEXSI is far lower than conventional solvers such as iterative or explicitly diagonalization methods ($\mathcal{O}(N^3)$). Furthermore, the quadratic memory footprint of conventional solvers also limit the system scale accessible on modern supercomputers. For example, the current state-of-the-art metallic system is a 11K-atom Mg system with a time-to-solution of 142 seconds per SCF and 46 PFLOPS (27.8% peak of 3,800 GPU nodes) on Summit. To reach a physical system of 2.5M atom (17.2M electrons) as shown in Fig. 8, conventional methods require a much bigger supercomputer and time-to-solution is estimated to be $2,209,147$ seconds according to the cubic scaling. This indicates that our DGDFT can be 2054 times faster than the current state-of-the-art for systems with 2.5M atom.

VIII. Implications

A. Scientific Applications

The moiré electrons in MATBG [5] form ordered quantum dot arrays, paving the way for next-generation devices, such as graphene FETs [4]. However, such large-scale complex and metallic systems ($>$ 10K) are too difficult to investigate by conventional cubic-scaling DFT calculations.

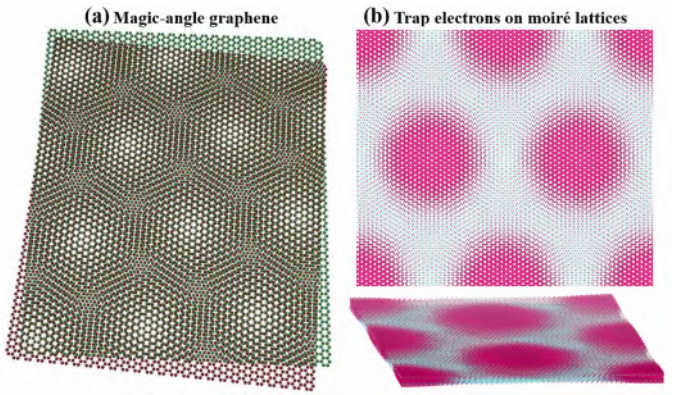

Fig. 9. Quantum electronic structures of MATBG, involving (a) atomic structures, (b) top and side views of local density of states (LDOS) at the Fermi level.

We use DGDFT to investigate *ab initio* electronic structures of MATBG (10K carbon atom) as shown in Fig. 9. We calculate the local density of states (LDOS) of MATBG and observe the moiré superlattices trap a number of localized electrons at the Fermi level in MATBG due to strong quantum electron-correlation effect. Our DFT simulations can serve as the first step towards *ab initio* understanding of superconducting and correlated insulating behaviors observed in experiments [5] for next-generation devices [4].

B. Outlook of DGDFT

For quantum mechanics, we are trying to develop the DG framework into carrying out complex electronic structure calculations such as Hartree-Fock, post-Hartree-Fock, and quantum computational chemistry. For atomistic TCAD [2], simulating the next-generation FETs at atomic level (10$\sim$100 nm), such as Fin, Gate-All-Around and Multi-Bridge Channel FETs, is still challenging, because their atomic structures and electronic properties are too complex and too large in DFT calculations. We plan to incorporate the non-equilibrium Green's functions formalism [2] in DGDFT to simulate the quantum transport properties in FETs.

To date, leadship supercomputers worldwide vary in system design (ARM A64FX for Fugaku and AMD GPUs for Frontier). Since each architecture has its own proprietary programming model such as CUDA, RoCM, etc, one major challenge is the performance portability. Though we only

showcase the optimization of DGDFT on the new Sunway, we remark that the corresponding optimization strategies can be analogously applied to other many-core architectures such as GPUs. Furthermore, the optimization shown in this work helps improve the performance of DGDFT by two orders of magnitude compared to the case of Edison [30] and Sunway TaihuLight [31]. To our understanding, these optimization techniques can be applied to other platforms with no obstacle.

C. Outlook for Post-E Supercomputers

The past three decades have witnessed tremendous success of massively parallel computing, and the mainstream design philosophy is driven by compute-bound linear algebra applications such as LINPACK.As a result, every year the computing power of high-end supercomputers has steadily increased by around a factor of two, paving the way to the exascale computing era now. Meanwhile, the gap among computing power, memory bandwidth and network bandwidth is growing, for example, the FLOPS/Byte ratio of the fastest supercomputer has increased from 5 to 30 in the last decade. Bounded by the hardware technology and power wall, supercomputers in the near future will most likely keep the inertia of pursuing FLOPS with many-core architecture, e.g., the tensor cores with higher FLOPS/Byte ratio. This trend has greatly shaped the landscape of HPC algorithms to dense linear algebra with a divide-and-conquer style, which can be reflected by sub-domain communication pattern and dense computations of the Gordon Bell Prize winning applications in recent years [11], [53]. On the other hand, supercomputers today are no longer performance-friendly for algorithms with frequent global communication or inconsistent memory access, especially sparse linear algebra operations. It is difficult to exploit data locality only from the perspective of hardware architecture for dense computations. The challenge is to exploit more and more application/input specific feature to "create" data locality. In order to boost the peak performance on the Sunway many-core architecture, we exploit the block structure of the Hamiltonian matrix, which is common in many real-world physical problems. We believe that the insights gained in this work can be helpful for other scientific applications with similar pattern. In fact, our algorithmic innovations appeal to the emerging concept of "Octopodes" [54] for improving overall co-design cycle of future supercomputers. And our DGDFT can be one potential candidate for further pushing the limit of quantum mechanical simulations into macroscopic scale (1000 nm) on post-E supercomputers.

ACKNOWLEDGMENT

All numerical experiments are performed on the new Sunway supercomputer. This work is partly supported by National Key Research and Development Program of China (2016YFA0200600, 2021YFB0300600), National Science Foundation of China (22173093, 12131002, T2125013, 22288201, 22003061), Innovation Program for Quantum Science and Technology (2021ZD0303306) and Anhui Initiative in Quantum Information Technologies (AHY090400), CAS Project for Young Scientists in Basic Research (YSBR-005) and Network Information Project of Chinese Academy of Sciences (CASWX2021SF-0103). We thank Prof. Lin Lin (University of California, Berkeley and Lawrence Berkeley National Laboratory), Prof. Chao Yang (Lawrence Berkeley National Laboratory), Prof. Weinan E (Peking University), Prof. Yi Luo (University of Science and Technology of China), Prof. Lin-Wang Wang (Institute of Semiconductors, Chinese Academy of Sciences) and Dr. Long Wang for helpful discussions.

REFERENCES

[1] G. Iannaccone, F. Bonaccorso, and et al., "Quantum engineering of transistors based on 2D materials heterostructures," *Nat. Nanotechnol.*, vol. 13, pp. 183–191, 2018.

[2] A. N. Ziogas, T. Ben-Nun, and et al., "A data-centric approach to extreme-scale ab initio dissipative quantum transport simulations," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2019, pp. 1–13.

[3] A. K. Geim and I. V. Grigorieva, "Van der waals heterostructures," *Nature*, vol. 499, pp. 419–425, 2013.

[4] J. R. Prance and M. B. Shalom, "Building devices in magic-angle graphene," *Nat. Nanotechnol.*, vol. 16, pp. 745–746, 2021.

[5] Y. Cao, V. Fatemi, and et al., "Unconventional superconductivity in magic-angle graphene superlattices," *Nature*, vol. 556, no. 7699, pp. 43–50, 2018.

[6] J. K. Stark, Y. Ding, and P. A. Kohl, "Dendrite-free electrodeposition and reoxidation of lithium-sodium alloy for metal-anode battery," *J. Electrochem. Soc.*, vol. 158, no. 10, p. A1100, 2011.

[7] A. Ohtomo and H. Y. Hwang, "A high-mobility electron gas at the $LaAlO_3$/$SrTiO_3$ heterointerface," *Nature*, vol. 427, pp. 423–426, 2004.

[8] P. Hohenberg and W. Kohn, "Inhomogeneous electron gas," *Phys Rev*, vol. 136, p. B864, 1964.

[9] W. Kohn and L. J. Sham, "Self-consistent equations including exchange and correlation effects," *Phys Rev*, vol. 140, p. A1133, 1965.

[10] L. Wang, "Divide-and-conquer quantum mechanical material simulations with exascale supercomputers," *Natl. Sci. Rev.*, vol. 1, no. 4, pp. 604–617, 2014.

[11] S. Das, P. Motamarri, and et al., "Fast, scalable and accurate finite-element based ab initio calculations using mixed precision computing: 46 pflops simulation of a metallic dislocation system," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC '19. New York, NY, USA: Association for Computing Machinery, 2019.

[12] M. J. Frisch, J. A. Pople, and J. S. Binkley, "Self-consistent molecular orbital methods 25. supplementary functions for gaussian basis sets," *J. Chem. Phys.*, vol. 80, pp. 3265–3269, 1984.

[13] D. R. Bowler and T. Miyazaki, "O(N) methods in electronic structure calculations," *Rep. Prog. Phys.*, vol. 75, p. 036503, 2012.

[14] F. Gygi, E. W. Draeger, M. Schulz, and et. al., "Large-scale electronic structure calculations of high-z metals on the bluegene/l platform," in *Proceedings of the 2006 ACM/IEEE Conference on Supercomputing*, ser. SC '06. New York, NY, USA: Association for Computing Machinery, 2006, p. 45–es.

[15] P. Giannozzi, S. Baroni, N. Bonini, and et al., "QUANTUM ESPRESSO: A modular and open-source software project for quantum simulations of materials," *J Phys: Condens Matter*, vol. 21, no. 39, p. 395502, 2009.

[16] W. Hu, L. Lin, and et al., "Adaptively compressed exchange operator for large-scale hybrid density functional calculations with applications to the adsorption of water on silicene," *J. Chem. Theory Comput.*, vol. 13, no. 3, pp. 1188–1198, 2017.

[17] W. Yang, "Electron density as the basic variable: a divide-and-conquer approach to the ab initio computation of large molecules," *J. Mol. Struct.:THEOCHEM*, vol. 255, pp. 461–479, 1992.

[18] Z. Zhao, J. Meza, and L. Wang, "A divide-and-conquer linear scaling three-dimensional fragment method for large scale electronic structure calculations," *J. Phys. Condens. Matter.*, vol. 20, pp. 294 203–294 210, 2008.

[19] V. Michaud-Rioux, L. Zhang, and H. Guo, "RESCU: A real space electronic structure method," *J. Comput. Phys.*, vol. 307, pp. 593–613, 2016.

[20] A. S. Banerjee, L. Lin, and et al., "Two-level chebyshev filter based complementary subspace method: Pushing the envelope of large-scale electronic structure calculations," *J. Chem. Theory Comput.*, vol. 14, pp. 2930–2946, 2018.

[21] T. D. Kühne, M. Iannuzzi, and et al., "Cp2k: An electronic structure and molecular dynamics software package-quickstep: Efficient and accurate electronic structure calculations," *J. Chem. Phys.*, vol. 152, no. 19, p. 194103, 2020.

[22] A. Nakata, J. S. Baker, S. Y. Mujahed, and et al., "Large scale and linear scaling dft with the conquest code," *J. Chem. Phys.*, vol. 152, no. 16, p. 164112, 2020.

[23] T. Ozaki and H. Kino, "Efficient projector expansion for the ab initio LCAO method," *Phys. Rev. B.*, vol. 72, no. 4, p. 045121, 2005.

[24] C.-K. Skylaris, P. D. Haynes, A. A. Mostofi, and M. C. Payne, "Introducing ONETEP: Linear-scaling density functional simulations on parallel computers," *J. Chem. Phys.*, vol. 122, no. 8, p. 084119, 2005.

[25] H. Shang, F. Li, Y. Zhang, and et al., "Extreme-scale ab initio quantum raman spectra simulations on the leadership hpc system in china," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2021, pp. 1–13.

[26] J. M. Soler, E. Artacho, J. D. Gale, and et al., "The SIESTA method for ab initio order-N materials simulation," *J Phys: Condens Matter*, vol. 14, no. 11, p. 2745, 2002.

[27] G. Kresse and J. Hafner, "Ab initio molecular dynamics for liquid metals," *Phys Rev B*, vol. 47, p. 558, 1993.

[28] L. Wang, B. Lee, and et al., "Linearly scaling (3D) fragment method for large-scale electronic structure calculations," in *Proceedings of the 2008 ACM/IEEE Conference on Supercomputing*, ser. SC '08. IEEE Press, 2008.

[29] Y. Hasegawa and J.-I. I. et al., "First-principles calculations of electron states of a silicon nanowire with 100,000 atoms on the k computer," in *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC '11. New York, NY, USA: Association for Computing Machinery, 2011.

[30] W. Hu, L. Lin, and C. Yang, "DGDFT: A massively parallel method for large scale density functional theory calculations," *J. Chem. Phys.*, vol. 143, no. 12, p. 124110, 2015.

[31] W. Hu, X. Qin, and et al., "High performance computing of dgdft for tens of thousands of atoms using millions of cores on sunway taihulight," *Sci. Bull.*, vol. 66, no. 2, pp. 111–119, 2021.

[32] B. Cockburn, G. Karniadakis, and C.Shu, "The development of discontinuous galerkin methods discontinuous galerkin methods," in *Berlin: Springer*. ACM, 2000, pp. 3–50.

[33] C. Hu and C. Shu, "A discontinuous galerkin finite element method for hamilton-jacobi equations," *SIAM J. Sci. Comput.*, vol. 12, no. 2, p. 666, 1999.

[34] I. Lomtev and G. E. Karniadakis, "A discontinuous galerkin method for the navier–stokes equations," *Int. J. Numer. Methods Fluids*, vol. 29, no. 5, p. 587, 1999.

[35] L. Lin, J. Lu, L. Ying, and W. E, "Adaptive local basis set for Kohn-Sham density functional theory in a discontinuous Galerkin framework I: Total energy calculation," *J. Comput. Phys.*, vol. 231, no. 4, pp. 2140–2154, 2012.

[36] G. Zhang, L. Lin, and et al., "Adaptive local basis set for kohn-sham density functional theory in a discontinuous galerkin framework II: Force, vibration, and molecular dynamics calculations," *J. Comput. Phys.*, vol. 335, pp. 426–443, 2017.

[37] L. Lin, M. Chen, C. Yang, and L. He, "Accelerating atomic orbital-based electronic structure calculation via pole expansion and selected inversion," *J Phys: Condens Matter*, vol. 25, p. 295501, 2013.

[38] W. Hu, L. Lin, and et al., "Edge-modified phosphorene nanoflake heterojunctions as highly efficient solar cells," *Nano Lett.*, vol. 16, p. 1675, 2016.

[39] W. Hu, Y. Huang, X. Qin, and et. al., "Room-temperature magnetism and tunable energy gaps in edge-passivated zigzag graphene quantum dots," *npj 2D Mater. Appl.*, vol. 3, p. 17, 2019.

[40] S. Liu, W. Hu, and et al., "$[Ti_{12}In_6O_{18}(OOCC_6H_5)_{30}]$: A multifunctional hetero-polyoxotitanate nanocluster with high stability and visible photoactivity," *Dalton Trans.*, vol. 46, no. 3, pp. 678–684, 2017.

[41] J. Zhang, W. Hu, and et al., "Stable heteropolyoxotitanate nanocluster for full solar spectrum photocatalytic hydrogen evolution," *J. Phys. Chem. C*, vol. 121, no. 34, pp. 18 326–18 332, 2017.

[42] X. S. Li and J. W. Demmel, "SuperLU_DIST: A scalable distributed-memory sparse direct solver for unsymmetric linear systems," *ACM Transactions on Mathematical Software (TOMS)*, vol. 29, no. 2, pp. 110–140, 2003.

[43] M. Jacquelin, L. Lin, and C. Yang, "PSelInv-A distributed memory parallel algorithm for selected inversion: The symmetric case," *ACM Transactions on Mathematical Software (TOMS)*, vol. 43, no. 3, pp. 1–28, 2016.

[44] W. Jia and L. Lin, "Robust determination of the chemical potential in the pole expansion and selected inversion method for solving kohn-sham density functional theory," *J. Chem. Phys.*, vol. 147, no. 14, p. 144107, 2017.

[45] https://www.top500.org, June 2022 (2022-06-01).

[46] Q. Zhu, H. Luo, and et al., "Enabling and scaling the hpcg benchmark on the newest generation sunway supercomputer with 42 million heterogeneous cores," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2021, pp. 1–13.

[47] Y. Zhao, Y. Ao, and et al., "General implementation of 1-d fft on the sunway 26010 processor," *Journal of Software (in Chinese)*, vol. 31, no. 10, pp. 3184–3196, 2020.

[48] E. Faccioli, F. Maggio, and et al., "2D and 3D elastic wave propagation by a pseudo-spectral domain decomposition method," *J. Seismol.*, vol. 1, no. 3, pp. 237–251, 1997.

[49] Z. Ma, J. He, and et al., "BaGuaLu: Targeting brain scale pretrained models with over 37 million cores," in *Proceedings of the 27th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, 2022, pp. 192–204.

[50] C. Hartwigsen, S. Goedecker, and J. Hutter, "Relativistic separable dual-space gaussian pseudopotentials from H to Rn," *Phys Rev B*, vol. 58, p. 3641, 1998.

[51] S. Goedecker, M. Teter, and J. Hutter, "Separable dual-space gaussian pseudopotentials," *Phys. Rev. B*, vol. 54, p. 1703, 1996.

[52] H. Cao and J. Chen, "Design and implementation of ShenWei universal C/C++," 2022. [Online]. Available: https://arxiv.org/abs/2208.00607

[53] W. Jia, H. Wang, and et al., "Pushing the limit of molecular dynamics with ab initio accuracy to 100 million atoms with machine learning," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC '20. IEEE Press, 2020.

[54] S. Matsuoka, J. Domke, and et. al., "Preparing for the future – Rethinking proxy apps," *arXiv:2204.07336*, 2022.