DGDFT: A massively parallel method for large scale density functional theory calculations

Wei Hu, Lin Lin, and Chao Yang

Citation: The Journal of Chemical Physics **143**, 124110 (2015); doi: 10.1063/1.4931732
View online: http://dx.doi.org/10.1063/1.4931732
View Table of Contents: http://scitation.aip.org/content/aip/journal/jcp/143/12?ver=pdfcov
Published by the AIP Publishing

Articles you may be interested in

Optical absorbance of doped Si quantum dots calculated by time-dependent density functional theory with partial electronic self-interaction corrections
J. Chem. Phys. **137**, 144301 (2012); 10.1063/1.4755995

Robust acceleration of self consistent field calculations for density functional theory
J. Chem. Phys. **134**, 134109 (2011); 10.1063/1.3574836

Linear-scaling method for calculating nuclear magnetic resonance chemical shifts using gauge-including atomic orbitals within Hartree-Fock and density-functional theory
J. Chem. Phys. **127**, 054103 (2007); 10.1063/1.2749509

Comparison of two genres for linear scaling in density functional theory: Purification and density matrix minimization methods
J. Chem. Phys. **122**, 084114 (2005); 10.1063/1.1853378

Improving harmonic vibrational frequencies calculations in density functional theory
J. Chem. Phys. **106**, 10175 (1997); 10.1063/1.474047

This article is copyrighted as indicated in the article. Reuse of AIP content is subject to the terms at: http://scitation.aip.org/termsconditions. Downloaded to IP: 169.229.58.198 On: Tue, 29 Sep 2015 17:50:29

DGDFT: A massively parallel method for large scale density functional theory calculations

Wei Hu,[1,a)] Lin Lin,[1,2,b)] and Chao Yang[1,c)]
[1]*Computational Research Division, Lawrence Berkeley National Laboratory, Berkeley, California 94720, USA*
[2]*Department of Mathematics, University of California, Berkeley, California 94720, USA*

(Received 30 June 2015; accepted 8 September 2015; published online 29 September 2015)

We describe a massively parallel implementation of the recently developed discontinuous Galerkin density functional theory (DGDFT) method, for efficient large-scale Kohn-Sham DFT based electronic structure calculations. The DGDFT method uses adaptive local basis (ALB) functions generated on-the-fly during the self-consistent field iteration to represent the solution to the Kohn-Sham equations. The use of the ALB set provides a systematic way to improve the accuracy of the approximation. By using the pole expansion and selected inversion technique to compute electron density, energy, and atomic forces, we can make the computational complexity of DGDFT scale at most quadratically with respect to the number of electrons for both insulating and metallic systems. We show that for the two-dimensional (2D) phosphorene systems studied here, using 37 basis functions per atom allows us to reach an accuracy level of 1.3×10^{-4} Hartree/atom in terms of the error of energy and 6.2×10^{-4} Hartree/bohr in terms of the error of atomic force, respectively. DGDFT can achieve 80% parallel efficiency on 128,000 high performance computing cores when it is used to study the electronic structure of 2D phosphorene systems with 3500-14 000 atoms. This high parallel efficiency results from a two-level parallelization scheme that we will describe in detail. © *2015 AIP Publishing LLC.* [http://dx.doi.org/10.1063/1.4931732]

I. INTRODUCTION

Kohn-Sham (KS) density functional theory (DFT)[1,2] is the most widely used methodology for performing *ab initio* electronic structure calculations to study the structural and electronic properties of molecules, solids, and nanomaterials. However, until recently, routine DFT calculations are only limited to small systems because they have a relatively high complexity ($O(N^{2-3})$) with respect to the system size N. As the system size increases, the cost of traditional DFT calculations becomes prohibitively expensive. Therefore, it is still challenging to routinely use DFT calculations to treat large-scale systems that may contain thousands to tens of thousands of atoms. Although various linear scaling $O(N)$ methods[3–5] have been proposed for improving the efficiency of DFT calculations, they rely on the nearsightedness principle, which leads to exponentially localized density matrices in real-space for systems with a finite energy gap or at finite temperature. Furthermore, most of the existing linear scaling DFT codes, such as SIESTA,[6] OPENMX,[7] HONPAS,[8] CP2K,[9] and CONQUEST,[10] are based on the contracted and localized basis sets in the real-space, such as Gaussian-type orbitals or numerical atomic orbitals.[4] The accuracy of methods based on such contracted basis functions is relatively difficult to improve systematically compared to conventional uniform basis sets, for example, plane wave basis set,[11] while the disadvantage of using uniform basis sets is the relatively large number of basis functions per atom.

Another practical challenge of DFT calculations is related to the implementation of various approaches to take full advantage of the massive parallelism available on modern high performance computing (HPC) architectures. Conventional DFT codes, especially, plane wave codes, such as VASP,[12] QUANTUM ESPRESSO,[13] and ABINIT,[14] have relatively limited parallel scalability even for large-scale systems involving thousands of atoms. The strong scaling performance (i.e., the change of computational speed with respect to the number of processing units for a problem of fixed size) of these planewave based codes is often poor when the number of computational cores exceeds a few thousands. Nonetheless, improvements have been made in the past decade. For instance, Qbox[15] demonstrated high scalability to over 1000 atoms using 131 072 cores on the IBM Blue Gene/L architecture (of which a factor of 8 is due to k-point parallelization). On the other hand, several DFT software packages based on contracted basis functions have achieved high parallel performance using linear scaling techniques for insulating systems. Examples include CP2K[16] and CONQUEST,[17] in which linear scaling is achieved based on parallel sparse matrix multiplication.[18–20] CP2K has demonstrated calculations on 96 000 water molecular scaling to 46 656 cores.[16] CONQUEST has demonstrated scaling to over 2 000 000 atoms on 4096 cores.[17]

The recently developed discontinuous Galerkin density functional theory (DGDFT)[21] aims at reducing the number of basis functions per atom while maintaining accuracy comparable to that of the planewave basis set. This is achieved by using an adaptive local basis (ALB) set. One unique feature of the ALB set is that each ALB function is strictly localized in a certain element in the real space, and is discontinuous from

a)whu@lbl.gov
b)linlin@math.berkeley.edu
c)cyang@lbl.gov

 © 2015 AIP Publishing LLC

This article is copyrighted as indicated in the article. Reuse of AIP content is subject to the terms at: http://scitation.aip.org/termsconditions. Downloaded to IP: 169.229.58.198 On: Tue, 29 Sep 2015 17:50:29

the point of view of the global domain. The continuous Kohn-Sham orbitals and density are assembled from the discontinuous basis functions using the discontinuous Galerkin (DG) method.[22,23] The ALB set takes both atomic and chemical environmental effects into account, and can be systematically improved just by increasing one number (number of ALBs per element). For the two-dimensional phosphorene systems studied here, our numerical results show that using 37 basis functions per atom allows us to reach an accuracy level of 1.3×10^{-4} Hartree/atom in terms of the error of energy and 6.2×10^{-4} Hartree/bohr in terms of the error of atomic force, respectively. We remark that the idea of generating localized basis functions on the fly has also been explored in other electronic structure software packages such as ONETEP[24] and recently BigDFT,[25] where localized basis functions are continuous and are improved through an optimization procedure. Filter diagonalization[26] is another approach for contracting basis functions and has been applied to contract Gaussian type functions. The filter diagonalization requires choosing trial functions, and the choice of trial functions may not be straightforward for an initial set of fine basis functions such as planewaves, finite elements, or wavelets.

The solution produced by DGDFT is also fully consistent with the solution of standard Kohn-Sham equations in the limit of a complete basis set, and the error can be measured by *a posteriori* error estimators.[27]

The strict spatial locality guarantees that the ALBs can form an orthonormal basis set and the overlap matrix is therefore an identity matrix, which requires only the solution of a standard eigenvalue problem rather than a generalized eigenvalue problem. The discrete Kohn-Sham Hamiltonian matrix is sparse. Furthermore, the sparsity pattern bears a resemblance to a block version of finite difference stencils, and facilitates parallel implementation. The sparse discrete Hamiltonian matrix allows DGDFT to take advantage of the pole expansion and selected inversion (PEXSI) technique.[28–30] The PEXSI method overcomes the $O(N^3)$ scaling limit for solving Kohn-Sham DFT, and scales at most as $O(N^2)$ even for metallic systems at room temperature. In particular, the computational complexity of the PEXSI method is only $O(N)$ for quasi 1D systems, and is $O(N^{1.5})$ for quasi two-dimensional(2D) systems. This also makes the DGDFT methodology particularly suitable for analyzing low-dimensional (quasi 1D and 2D) systems regardless whether the system is a metal, a semiconductor, or an insulator,[29] different from the near-sightedness assumption in standard linear scaling DFT calculations.

In this paper, we describe a massively parallel DGDFT method, based on a two-level parallelization strategy. This strategy results directly from the domain decomposition nature of the DGDFT method in which the global computational domain is partitioned into a number of subdomains from which the ALBs are generated. We demonstrate the accuracy and high parallel efficiency of DGDFT by using 2D phosphorene monolayers with 3500-14 000 atoms as examples to show that DGDFT can take full advantage of up to 128 000 cores on a high performance computer.

The rest of paper is organized as follows. After brief introduction of the DGDFT method, we describe several important implementation considerations of the massively parallel DGDFT method in Section II. We demonstrate the numerical performance of DGDFT for 2D phosphorene monolayers in Section III, followed by the conclusion in Section IV.

II. DGDFT METHODOLOGY AND PARALLEL SCHEME

A. Adaptive local basis set in a discontinuous Galerkin framework

Since the main focus of this paper is to describe a highly efficient parallel implementation of the DGDFT methodology, we will not provide detailed theoretical derivation of the DGDFT method that can be found in Ref. 21. We will briefly introduce the basic work flow of DGDFT, and the main steps and major computational bottlenecks that require parallelization. For detailed explanation of technical details, we refer readers to Ref. 21 and a forthcoming publication[31] concerning the total energy and the force calculation in DGDFT. Throughout the paper we consider Γ-point calculation only. Therefore, the Kohn-Sham orbitals are real. We assume the system is closed shell. Spin degeneracy is omitted to simplify the notation in this section, but is correctly accounted for in the DGDFT code and the numerical simulation.

The goal of the DGDFT method is to find the minimizer of the Kohn-Sham energy functional, which is a nonlinear functional in terms of N single particle wavefunctions (orbitals) ψ_i. The minimizer of this functional is sought in a self-consistent field (SCF) iteration which produces electron density ρ that satisfies a fixed point (or self-consistent) nonlinear map. In a standard SCF iteration, a linear eigenvalue problem needs to be solved in each SCF cycle. As discussed in Ref. 21, the solution of this linear eigenvalue problem is usually the computational bottleneck for solving Kohn-Sham DFT.

In the DGDFT method, we partition the global computational domain Ω into a number of subdomains (called elements), denoted by $E_1, \ldots, E_M$. An example of partitioning the global domain of a model problem into a number of 2D elements is given in Fig. 1. We use $\mathcal{T}$ to denote the collection of all elements. In the current version of DGDFT, we use periodic boundary conditions to treat both molecule and solids. However, the method can be relatively easily generalized to other boundary conditions such as Dirichlet boundary condition. Therefore, each surface of the element must be shared between two neighboring elements, and $\mathcal{S}$ denotes the collection of all the surfaces.

Each Kohn-Sham orbital is expanded into a linear combination of ALBs $\varphi_{K,j}$, i.e.,

$$\psi_i(x) = \sum_{K=1}^{M} \sum_{j=1}^{J_K} c_{i;K,j} \varphi_{K,j}(x), \tag{1}$$

where $\varphi_{K,j}(x)$ is the jth ALB in the element E_K that has nonzero support only in E_K, and J_K is the total number of ALBs in E_K. The basis function $\varphi_{K,j}$ is not necessarily continuous across the boundary of E_K. Because each $\varphi_{K,j}$ is assumed to be square integrable within E_K and is zero outside of E_K, its inner product with other function can be defined in terms of an L^2 inner product defined on E_K. As a result, a natural global inner product between quantities such as ψ_i, which we

This article is copyrighted as indicated in the article. Reuse of AIP content is subject to the terms at: http://scitation.aip.org/termsconditions. Downloaded to IP: 169.229.58.198 On: Tue, 29 Sep 2015 17:50:29

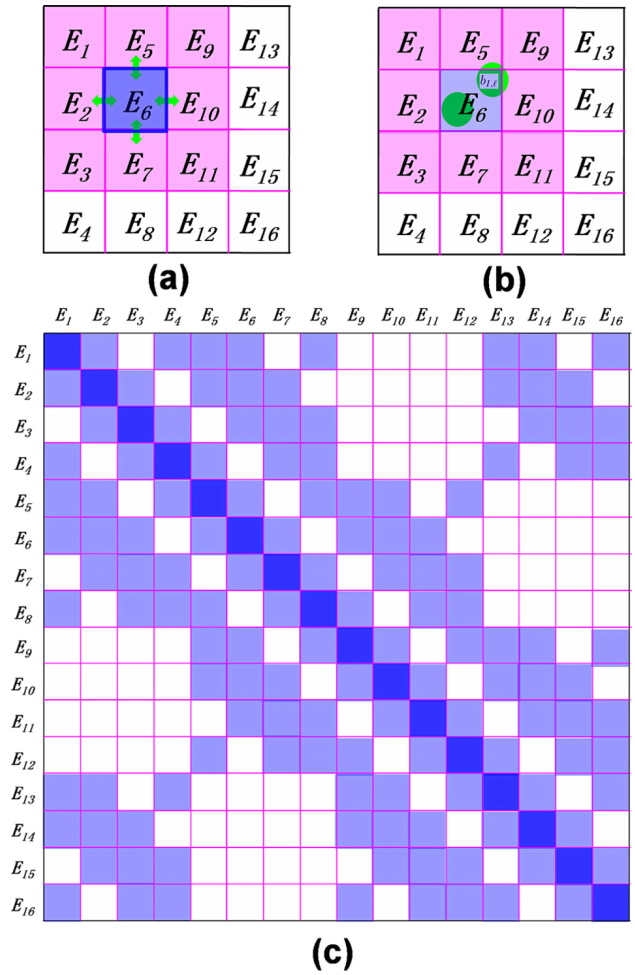

FIG. 1. A model system in 2D partitioned into 16 (4×4) equal sized elements. (a) An extended element Q_6 associated with the element E_6, and Q_6 includes elements E_1, E_2, E_3, E_5, E_6, E_7 E_9, E_{10}, and E_{11}. The four surfaces of E_6 on which boundary integrals are computed are highlighted by arrows. (b) Two possible cases of projection vectors of the nonlocal pseudopotential, one contained in an element, and another contained in multiple (here 4) elements. (c) The block structure of DG Hamiltonian matrix with blocks with nonzero values highlighted.

denote by $\langle \cdot, \cdot \rangle_{\mathcal{T}}$, can be taken as the sum of local L^2 inner products defined for all elements. Similarly, we can also define a global surface inner product, denoted by $\langle \cdot, \cdot \rangle_{\mathcal{S}}$, as the sum of local L^2 surface inner products defined on all surfaces of all elements. The ALB set is orthonormal, i.e., for $K, K' = 1, \ldots, M$, $j = 1, \ldots, J_K$, $j' = 1, \ldots, J_{K'}$,

$$\langle \varphi_{K,j}, \varphi_{K',j'} \rangle_{\mathcal{T}} = \delta_{K,K'} \delta_{j,j'}. \tag{2}$$

Because the expansion of ψ_i contains at least two ALBs localized in two adjacent elements with a shared surface on which neither ALB is continuous, the values of ψ_i on two sides of the surface can be different. This difference calls for the notion of average of gradient $\{\{\nabla \psi_i\}\}$ and the concept of jump of function value $[\![\psi_i]\!]$ defined on the surface. The use of average and jump operators distinguishes DGDFT from other KS-DFT solvers.

Using these notations, the total energy functional to be minimized with respect to the N Kohn-Sham orbitals $\{\psi_i\}$ subject to orthonormality condition can be written as

$$\begin{aligned} E_{\mathrm{DG}}(\{\psi_i\}) = & \frac{1}{2} \sum_{i=1}^{N} \langle \nabla \psi_i, \nabla \psi_i \rangle_{\mathcal{T}} + \langle V_{\mathrm{eff}}, \rho \rangle_{\mathcal{T}} \\ & + \sum_{I=1}^{N_A} \sum_{\ell=1}^{L_I} \gamma_{I,\ell} \sum_{i=1}^{N} \left| \langle b_{I,\ell}(\cdot - R_I), \psi_i \rangle_{\mathcal{T}} \right|^2 \\ & - \sum_{i=1}^{N} \langle \{\{\nabla \psi_i\}\}, [\![\psi_i]\!] \rangle_{\mathcal{S}} \\ & + \alpha \sum_{i=1}^{N} \langle [\![\psi_i]\!], [\![\psi_i]\!] \rangle_{\mathcal{S}}, \end{aligned} \tag{3}$$

where we use V_{eff} to denote the effective one-body potential (including local pseudopotential V_{loc}, Hartree potential V_H, and the exchange-correlation potential $V_{\mathrm{xc}}[\rho]$), the terms that contain $b_{I,\ell}$ and $\gamma_{I,\ell}$ correspond to the nonlocal pseudopotential in the Kleinman-Bylander form,[32] and α is an adjustable penalty parameter to ensure that Eq. (3) has a well defined ground state energy. For each atom I, there are L_I functions $\{b_{I,\ell}\}$, called projection vectors of the nonlocal pseudopotential. The parameters $\{\gamma_{I,\ell}\}$ are real valued scalars. We refer the readers to Ref. 21 for more detailed explanation of the notation and theory.

The procedure for constructing the ALBs and a detailed example of the ALBs will be given later in this subsection. For now we just treat Eq. (1) as an ansatz for representing the Kohn-Sham orbitals. Minimizing the coefficients $\{c_{i;K,j}\}$ subject to orthonormality condition gives rise to the Euler-Lagrange equation of Eq. (3) which is a linear eigenvalue problem,

$$\sum_{K,j} H^{\mathrm{DG}}_{K',j';K,j} c_{i;K,j} = \lambda_i c_{i;K',j'}. \tag{4}$$

Here, H^{DG} is the discrete Kohn-Sham Hamiltonian matrix, with matrix entries given by

$$\begin{aligned} H^{\mathrm{DG}}_{K',j';K,j} = & \Big(\frac{1}{2} \langle \nabla \varphi_{K,j'}, \nabla \varphi_{K,j} \rangle_{\mathcal{T}} + \langle \varphi_{K,j'}, V_{\mathrm{eff}} \varphi_{K,j} \rangle_{\mathcal{T}} \Big) \delta_{K,K'} \\ & + \Big(\sum_{I,\ell} \gamma_{I,\ell} \langle \varphi_{K',j'}, b_{I,\ell} \rangle_{\mathcal{T}} \langle b_{I,\ell}, \varphi_{K,j} \rangle_{\mathcal{T}} \Big) \\ & + \Big(-\frac{1}{2} \langle [\![\varphi_{K',j'}]\!], \{\{\nabla \varphi_{K,j}\}\} \rangle_{\mathcal{S}} \\ & - \frac{1}{2} \langle \{\{\nabla \varphi_{K',j'}\}\}, [\![\varphi_{K,j}]\!] \rangle_{\mathcal{S}} \\ & + \alpha \langle [\![\varphi_{K',j'}]\!], [\![\varphi_{K,j}]\!] \rangle_{\mathcal{S}} \Big). \end{aligned} \tag{5}$$

The H^{DG} matrix can be naturally partitioned into matrix blocks as sketched in Fig. 1. We call the submatrix $H^{\mathrm{DG}}_{K',:;K,:}$ of size $J_{K'} \times J_K$ the (K', K)th matrix block of H^{DG}, or $H^{\mathrm{DG}}_{K';K}$ for short. The terms are grouped by three parentheses on the right hand side of Eq. (5) to reflect different contributions to the DG Hamiltonian matrix, and will be treated differently in our parallel implementation of the method. The first group originates from the kinetic energy and the local pseudopotential, and only contributes to the diagonal blocks $H^{\mathrm{DG}}_{K,K}$. The second group comes from nonlocal pseudopotentials, and contributes to both the diagonal and off-diagonal blocks of H^{DG}. Since a projection vector of the nonlocal pseudopotential is spatially localized, we require the dimension of every element along each direction (usually on the order of 6–8 bohrs) to be larger

This article is copyrighted as indicated in the article. Reuse of AIP content is subject to the terms at: http://scitation.aip.org/termsconditions. Downloaded to IP: 169.229.58.198 On: Tue, 29 Sep 2015 17:50:29

than the size of the nonzero support of each projection vector (usually on the order of 2–4 bohrs). Thus, the nonzero support of each projection vector can overlap with at most 2^d elements as shown in Fig. 1(b) ($d = 1,2,3$). As a result, each nonlocal pseudopotential term may contribute both to the diagonal and the off-diagonal blocks of H^{DG}. The third group consists of contributions from boundary integrals, and can also contribute to both the diagonal and off-diagonal blocks of H^{DG}. Each boundary term involves only two neighboring elements by definition as plotted in Fig. 1(a). In summary, H^{DG} is a sparse matrix and the nonzero matrix blocks correspond to interactions between neighboring elements (Fig. 1).

After constructing the H^{DG} matrix and solving eigenvalue problem (4), the electron density required to evaluate the effective potential V_{eff} in Eq. (3) can be obtained from

$$\rho(x) = \sum_{i=1}^{N} |\psi_i(x)|^2. \tag{6}$$

It is well known that (6) is not the only way to compute the electron density. An alternative approach which does not require computing eigenvalues and eigenvectors of H_{DG} is the PEXSI method.[28–30] To make use of the PEXSI method, we need to express $\rho(x)$ in terms of selected blocks of the density matrix represented in the ALB set (or density matrix for short in the following discussion). To be more precise, this density matrix has the form

$$P_{K,j;K',j'} = \sum_{i=1}^{N} c_{i;K,j} c_{i;K',j'}, \tag{7}$$

and can be accurately approximated as a matrix function of H_{DG} without knowing $c_{i;K,j}$ explicitly.

Using the density matrix, we can express the electron density as

$$\begin{aligned}\rho(x) &= \sum_{K=1}^{M}\sum_{j=1}^{J_K}\sum_{K'=1}^{M}\sum_{j'=1}^{J_{K'}} \varphi_{K,j}(x)\varphi_{K',j'}(x)\left(\sum_{i=1}^{N} c_{i;K,j}c_{i;K',j'}\right)\\ &= \sum_{K=1}^{M}\sum_{j=1}^{J_K}\sum_{j'=1}^{J_K} \varphi_{K,j}(x)\varphi_{K,j'}(x)P_{K,j;K,j'}.\end{aligned} \tag{8}$$

Here, we have used the fact that each function $\varphi_{K,j}(x)$ is strictly localized in the element E_K to eliminate the cross terms involving both K and K'. As a result, the selected blocks, or more specifically, the diagonal blocks of the density matrix $P_{K;K} \equiv P_{K,:;K,:}$ are needed to evaluate the electron density. This is a key feature of this expression that makes it possible to use the PEXSI method as will be discussed in Section II C. After self-consistency of the electron density is achieved in the SCF iteration, the total energy and atomic forces can also be evaluated using the PEXSI method.

Eq. (8) assumes that the system is at zero temperature. For finite temperature treatment that is needed for metallic systems, the electron density should be computed as

$$\rho(x) = \sum_{K=1}^{M}\sum_{j=1}^{J_K}\sum_{K'=1}^{M}\sum_{j'=1}^{J_{K'}} \varphi_{K,j}(x)\varphi_{K',j'}(x)\left(\sum_{i=1}^{N} f_i c_{i;K,j}c_{i;K',j'}\right). \tag{9}$$

Here $f_i = \frac{1}{1+e^{\beta(\lambda_i-\mu)}}$ is the Fermi-Dirac distribution, β is the inverse of temperature, and μ is the chemical potential adjusted so that the total number of electrons is N. The Fermi-Dirac distribution can be derived by minimizing the Mermin free energy functional,[33] and a detailed derivation in the discontinuous Galerkin framework has been given in Ref. 34.

We have demonstrated that high accuracy in the total energy and atomic forces can be achieved with a very small number (4–40) of basis functions per atom in DGDFT, compared to fully converged planewave calculations.[21,31] Besides selected blocks of the density matrix (DM), the Helmholtz free energy and the atomic force can be evaluated by computing selected blocks of the free energy density matrix (FDM) and the energy density matrix (EDM), respectively. The technique for computing EDM and FDM is very similar to that for computing the DM.[29]

One notable feature of the ALB set is that it is generated on the fly, and is adaptive not only to the atomic but also the environmental information. In order to construct ALBs, we introduce, for each element E_K, an extended element Q_K that contains E_K and a buffer region surrounding E_K. We define $V_{\mathrm{eff}}^{Q_K} = V_{\mathrm{eff}}|_{Q_K}$ to be the restriction of the effective potential at the current SCF step to Q_K, and $V_{\mathrm{nl}}^{Q_K} = V_{\mathrm{nl}}|_{Q_K}$ to be the restriction of the nonlocal potential to Q_K. These restricted potentials define a local Kohn-Sham linear eigenvalue problem on each extended element Q_K,

$$\begin{aligned}\widetilde{H}_{\mathrm{eff}}^{Q_K}\widetilde{\varphi}_{K,j} &= \left(-\frac{1}{2}\Delta + V_{\mathrm{eff}}^{Q_K} + V_{\mathrm{nl}}^{Q_K}\right)\widetilde{\varphi}_{K,j} = \lambda_{K,j}\widetilde{\varphi}_{K,j},\\ \int_{Q_K} \widetilde{\varphi}_{K,j}(x)\widetilde{\varphi}_{K,j'}(x)\,\mathrm{d}x &= \delta_{jj'}.\end{aligned} \tag{10}$$

This linear eigenvalue problem can be discretized by using traditional basis sets such as plane waves and solved by an iterative method such as the locally optimal block preconditioned conjugate gradient (LOBPCG) method. We remark that SCF iterations need not and should not be performed within each element Q_K. The lowest J_K eigenvalues $\{\lambda_{K,j}\}_{j=1}^{J_K}$ and the corresponding eigenfunctions $\{\widetilde{\varphi}_{K,j}\}_{j=1}^{J_K}$ are computed on Q_K. We then restrict $\{\widetilde{\varphi}_{K,j}\}_{j=1}^{J_K}$ from Q_K to E_K. The truncated vectors are not necessarily orthonormal. Therefore, we orthonormalize the set of truncated eigenvectors to obtain $\{\varphi_{K,j}\}_{j=1}^{J_K}$. We then set each $\varphi_{K,j}$ to zero outside of E_K, so that it is defined over the entire domain, but is in general discontinuous across the boundary of E_K. These functions constitute the ALB set that we use to represent the Kohn-Sham Hamiltonian. Because they satisfy orthonormality condition (2), the overlap matrix corresponding to the ALB set is an identity matrix. Hence, no generalized eigenvalue problem with a potentially ill-conditioned overlap matrix needs to be solved. For more details of the construction of ALBs, such as the restriction of the potential and the choice of boundary conditions for the local eigenvalue problem (10), we refer readers to Ref. 21.

As an example, Fig. 2 shows the ALBs of a 2D phosphorene monolayer with 140 phosphorus atoms (P_{140}). This is a two-dimensional system, and the global computational domain is partitioned into 64 equal sized elements along the Y and Z directions, respectively. For instance, the extended element Q_{10} associated with the element E_{10} contains elements E_1, E_2,

This article is copyrighted as indicated in the article. Reuse of AIP content is subject to the terms at: http://scitation.aip.org/termsconditions. Downloaded to IP: 169.229.58.198 On: Tue, 29 Sep 2015 17:50:29

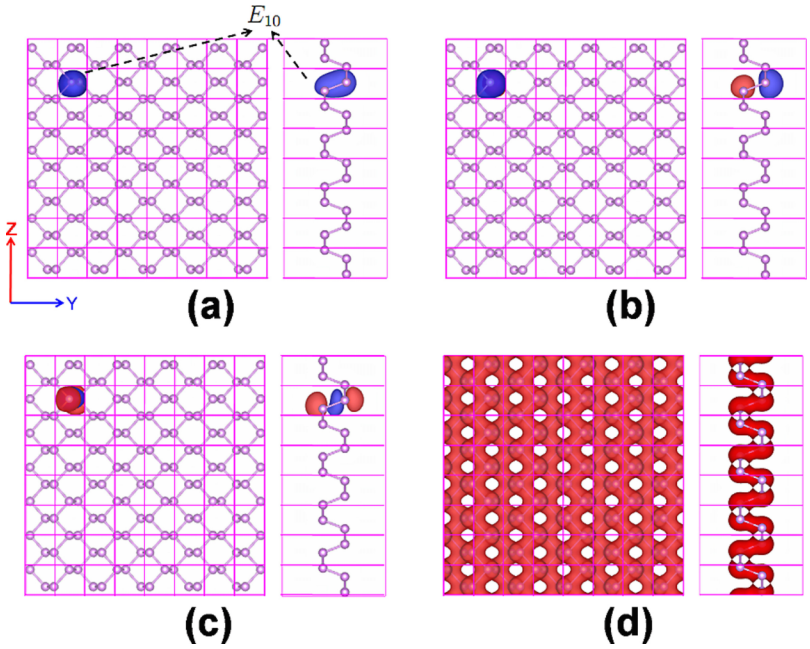

FIG. 2. The isosurfaces (0.04 Hartree/bohr^3) of the first three ALB functions belonging to the tenth element (E_{10}), (a) ϕ_1, (b) ϕ_2, (c) ϕ_3, and (d) the electron density ρ across in top and side views in the global domain in the example of P_{140}. The red and blue regions indicate positive and negative isosurfaces, respectively. There are 64 elements and 80 ALB functions in each element in the P_{140} system.

E_3, E_9, E_{11}, E_{17}, E_{18}, and E_{19}. We show the isosurfaces of the first three ALB functions for this element in Figs. 2(a)–2(c). Each ALB function shown is strictly localized inside E_{10} and is therefore discontinuous across the boundary of elements. On the other hand, each ALB function is delocalized across a few atoms inside the element since they are obtained from eigenfunctions of local Kohn-Sham Hamiltonian. Although the basis functions are discontinuous, the electron density is well-defined and is very close to be a continuous function in the global domain (Fig. 2(d)).

To summarize, the flowchart of the DGDFT method for solving Kohn-Sham DFT is shown in Fig. 3.

B. Two level parallelization strategy

The DGDFT framework naturally lends itself to a two level parallelization strategy. At the coarse grained level, we distribute work among different processors by elements. We call this level of concurrency the *inter-element* parallelization. Within each element, the eigenvalue solvers on each local (extended) domain and the construction of the DG Hamiltonian matrix can be further parallelized. This level of parallelization is called the *intra-element* parallelization. We use the Message Passing Interface (MPI) to handle data communication.

We assume that the total number of processors is $P_{\text{tot}} = M \times P_e$, where M is the number of elements, and P_e is the number of processors assigned to each element. We partition these processors into a 2D logical processor grid following a column major order as shown in Fig. 4. We call this 2D processor grid the global processor grid to distinguish it from other processor grids employed in various parts of the massively parallel DGDFT method. Each row (column) communicator of this grid is called a global row (column) communicator. The processors with ranks $(K-1)P_e + 1$ to KP_e $(K = 1, \ldots, M)$ are in the Kth global column processor group, and are assigned to element E_K for intra-element parallelization. Similarly, the processors with ranks $i, P_e + i, \ldots, (M-1)P_e + i$ $(i = 1, \ldots, P_e)$ are in the ith global row processor group. When

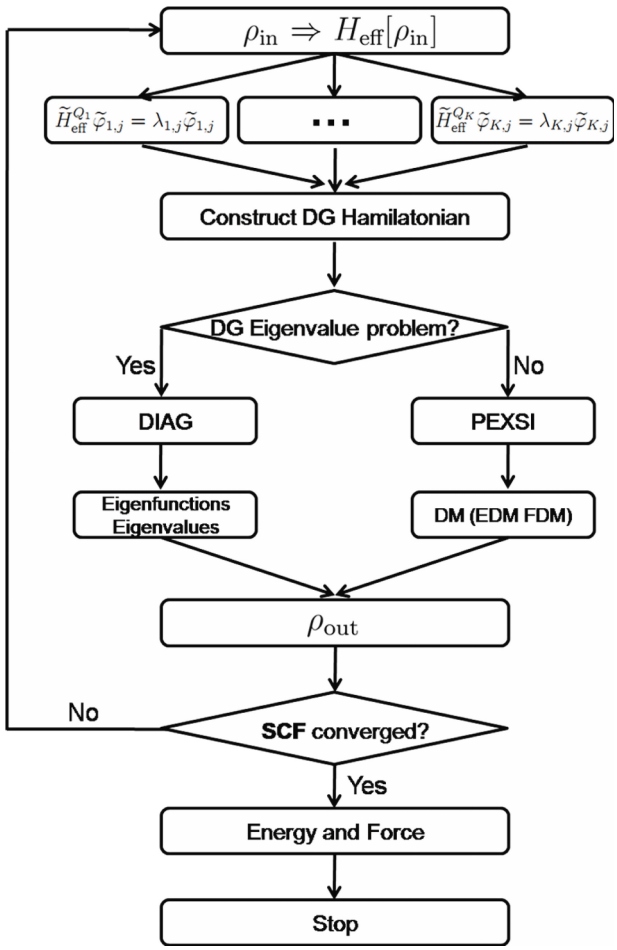

FIG. 3. Flowchart of the DGDFT method. There are three time consuming steps: the generation of ALB functions, the construction of DG Hamiltonian matrix, and the evaluation of the approximate charge density by either diagonalizing the DG Hamiltonian (DIAG) or by using the PEXSI method. H_{eff} and $\widetilde{H}_{\text{eff}}^{Q_K}$ represent the global Kohn-Sham Hamiltonian operator for a given electron density, and the local Kohn-Sham Hamiltonian on the extended element Q_K, respectively.

This article is copyrighted as indicated in the article. Reuse of AIP content is subject to the terms at: http://scitation.aip.org/termsconditions. Downloaded to IP: 169.229.58.198 On: Tue, 29 Sep 2015 17:50:29

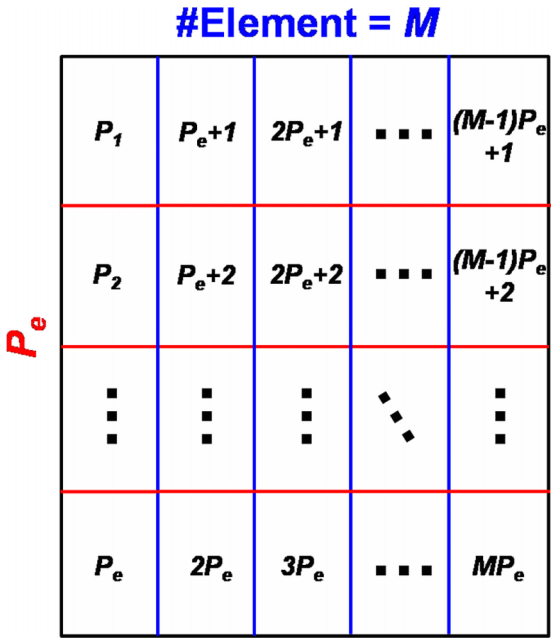

FIG. 4. Global processor grid for implementing the two level parallelization in the DGDFT method. M is the number of elements in the systems. P_e is the number of processors given for each element.

a very large number of processors are used, it is important to avoid global all-to-all communication as much as possible in order to reduce communication cost and achieve parallel scalability. Therefore, by dividing processors into column and row processor groups, we can restrict most of the communication within a global column or row group.

Based on the intra-element parallelization strategy to be detailed below, the maximum number of processors that can be effectively used for a single element depends on the number of basis functions to be generated on a single element, which is usually on the order of 100. It can be as large as a few hundreds. The level of concurrency that can be achieved in the inter-element parallelization is determined by the number of elements. The maximum number of processors that can be employed by DGDFT is therefore determined by the number of basis functions per element multiplied by the number of elements, which is equal to the dimension of the DG Hamiltonian matrix. For example, the P_{140} system contains 140 phosphorus atoms and is partitioned into 64 (8×8) elements. There are about 2 atoms in each element and we use 50 basis per element. The maximum number of processors that can be effectively used for this system is 3600. For the 2D phosphorene test problems (P_{3500} and P_{14000}) used in this work, we partition the phosphorene systems into 1600 (40×40) and 6400 (80×80) elements, respectively. The maximum number of processors that can be effectively used for these systems is 80 000 and 320 000, respectively. Currently, we are often limited by the number of processors available on the existing high performance computers such as the Edison system at NERSC. The maximum number of processors we can use is 128 000.

1. Generation of the ALB functions

For each extended element, the computation of eigenfunctions for the local Kohn-Sham Hamiltonian can be parallelized in a way similar to the parallelization of a standard planewave based Kohn-Sham DFT solver. The local Kohn-Sham orbitals form a local dense matrix Ψ_K of dimension $N_g \times J_K$. Here, J_K is the number of ALBs to be computed, and N_g is the total number of grid points required to represent each local Kohn-Sham orbital in the real space, which is determined from the kinetic energy cutoff $E_{\text{cut}}^{\text{wfc}}$ by the following rule:

$$(N_g)_i = \frac{\sqrt{2E_{\text{cut}}^{\text{wfc}}}L_i}{\pi}. \tag{11}$$

Here, L_i is the length of the extended element along the ith coordinate direction. The total number of grid points is $N_g = \prod_{i=1}^{3}(N_g)_i$. In a typical calculation, $N_g \sim 10^6$ and $J_K \sim 10^2$ as shown in Fig. 5.

On each extended element, we use the LOBPCG method[35] to solve the local Kohn-Sham orbitals. The main operations involved in the LOBPCG solver are (1) The application of the local Hamiltonian operator $\widetilde{H}_{\text{eff}}^{Q_K}$ to Ψ_K. (2) Dense matrix-matrix multiplication of the form $\Phi^T\Psi$, where matrices Φ, Ψ have the same dimension as the Kohn-Sham orbitals Ψ_K. (3) Dense matrix-matrix multiplication of the form ΦC, where Φ has the same dimension as Ψ_K, and C is a small matrix on the order of $J_K \times J_K$. (4) The diagonalization of matrices of size $O(J_K)$ using the Rayleigh-Ritz procedure. (5) The application of a preconditioner operator.

It should be noted that these different types of operations require different data distribution and task parallelization strategies. Operation (1) requires applying the Laplacian operator via the Fast Fourier Transform (FFT), the local pseudopotential operator, and the nonlocal pseudopotential operator to Ψ_K. This can be done easily if the orbitals (i.e., columns of Ψ_K) are partitioned along the column direction (see Fig. 5(a)). Since each processor holds all entries of an orbital, the FFT can be done in the same way as in a sequential implementation. We further assume that each processor associated with the extended element Q_K has a copy of the local pseudopotential $V_{\text{eff}}^{Q_K}$, and the nonlocal pseudopotential $V_{\text{nl}}^{Q_K}$. Therefore, all operations described in (1) can be performed in the same way as those in a sequential implementation.

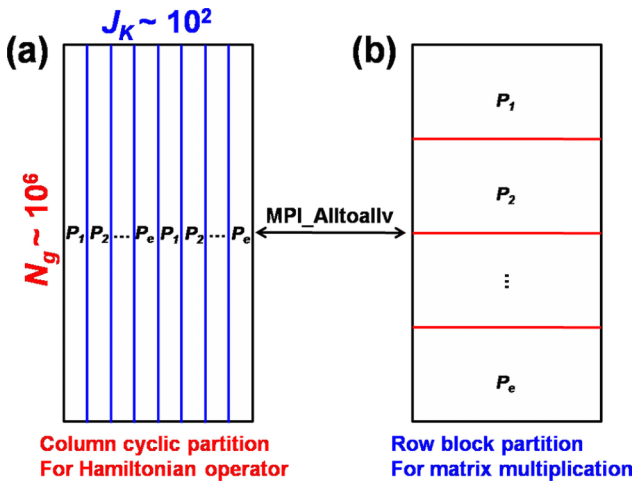

FIG. 5. Two different types of partition of Ψ_K. The partition shown in (a) is used for computing $\widetilde{H}_{\text{eff}}^{Q_K}\Psi_K$. The partition shown in (b) is used when matrix-matrix multiplications of the form $\Phi^T\Psi_K$ and $\Psi_K C$ are performed.

This article is copyrighted as indicated in the article. Reuse of AIP content is subject to the terms at: http://scitation.aip.org/termsconditions. Downloaded to IP: 169.229.58.198 On: Tue, 29 Sep 2015 17:50:29

The most efficient way to parallelize operations (2) and (3) is to partition Ψ_K by row blocks as shown in Fig. 5(b). This is because for operation (2), one can compute the matrix inner product of each block locally on each processor, and then use MPI_Allreduce among the P_e processors to sum up local products of size $J_K \times J_K$. For operation (3), there is no communication at all if all P_e processors have a local copy of the C matrix. Partitioning Ψ_K by columns would incur more communication cost and make this part of the computation less scalable. Since in each LOBPCG iteration we apply the Hamiltonian operator to Ψ_K once, but perform 12 matrix-matrix multiplications of operation type (2) and 7 matrix-matrix multiplications of operation type (3), it is worthwhile to switch back and forth between a column partition and row partition of Ψ_K in between the first and other types of operations performed in each LOBPCG iteration. This can be achieved by using a MPI_Alltoallv call.

For operation (4), since J_K is usually on the order of hundreds in practice, we solve the Rayleigh-Ritz problem and perform subspace diagonalization locally on each processor. Numerical experiments indicate that this sequential part usually takes around or less than 1 s.

Finally, in our implementation we use the preconditioner proposed in Ref. 36 to accelerate the convergence of LOBPCG. The preconditioner can be easily applied to different orbitals simultaneously without communication. Thus, a column partition of Ψ_K is suitable for applying the preconditioner in parallel.

Once the Kohn-Sham orbitals Ψ_K are constructed through LOBPCG, they are restricted from the extended element Q_K to the element E_K. After orthonormalizing columns of the restricted Ψ_K, we obtain the ALBs denoted by $\Phi_K = [\varphi_{K,1}, \ldots, \varphi_{K,J_K}]$. We remark that it is not necessary to compute the local wavefunctions Ψ_K to full accuracy before the electron density becomes self-consistent in the SCF iteration. As the accuracy of the electron density improves during the SCF cycle, a more accurate Ψ_K can be obtained from running a few iterations of the LOBPCG procedure that uses the Ψ_K returned from the previous SCF iteration as a starting guess. In practice, we find that using 3 preconditioned LOBPCG iterations per SCF iteration is often sufficient to achieve rapid convergence of the SCF procedure. Our numerical results indicate that the overall procedure for generating the ALBs is highly efficient. For instance, for the phosphorene P_{3500} system, the total time for generating the ALBs is only 1.35 and 0.71 s by using 12 800 and 25 600 processors, respectively.

2. Construction of the DG Hamiltonian

Due to the spatial locality of the ALBs, the DG Hamiltonian matrix in Eq. (5) is a sparse matrix and has a block structure that can be naturally distributed among different column processor groups assigned to different elements as shown in Fig. 1(c), i.e., the processors assigned to the element E_K assemble the Kth block row of H^{DG}. The construction of the DG Hamiltonian matrix consists of the evaluation of volume integrals within each element and surface integral at the boundary of different elements. To achieve high accuracy, all integrals in (5) are evaluated by a Gaussian quadrature defined on a Legendre-Gauss-Lobatto (LGL) grid in each element. The gradients of ALBs sampled on the LGL grid points and denoted by $\nabla\Phi_K$ can be evaluated by applying differentiation matrices to Φ_K along the x, y, z directions, respectively. To compute $\nabla\Phi_K$ efficiently, both Φ_K and $\nabla\Phi_K$ should be partitioned and distributed by columns among processors assigned to E_K. We refer readers to, e.g., Ref. 37 for details of constructing differentiation matrices associated with Gaussian quadratures.

The construction of the Kth block row of H^{DG} consists of the following three steps that correspond to the three terms grouped by parentheses on the right hand side of Eq. (5). (1) Compute contributions to the diagonal matrix blocks from the kinetic energy and V_{eff} terms. (2) Compute contributions to the diagonal and off-diagonal matrix blocks from the nonlocal pseudopotential term. (3) Compute contributions to the diagonal and off-diagonal matrix blocks from the boundary integral terms.

In step (1), it is generally more efficient to partition columns of Φ_K among different processors as shown in Fig. 5(a) because applying a differentiation matrix, or local pseudopotential V_{eff} to different columns of Φ_K requires no communication. The inner products $\langle\cdot,\cdot\rangle_{\mathcal{T}}$, however, are evaluated as vector inner products (or a matrix-matrix multiplication when several integrals are evaluated simultaneously). A block row partition is more efficient for these types of operations. Collective communication is required to sum up local products. To accommodate both data distribution schemes in this step, we use the MPI_Alltoallv call within a global column processor group to convert Φ_K and $\nabla\Phi_K$ from column partition to row partition.

More sophisticated data communication is needed for step 2. This is because the projection vector of a nonlocal pseudopotential may be distributed among several elements (up to 8, see Fig. 1(b)). If the nonzero support of the projection operator is completely within one element E_K, this projection operator only contributes to one diagonal block $H^{\mathrm{DG}}_{K;K}$. Otherwise, it contributes also to some off-diagonal matrix blocks $H^{\mathrm{DG}}_{K';K}$ for all $E_{K'}$'s that contain a distributed portion of the projection vector. Unlike the local pseudopotential V_{eff}, the nonlocal pseudopotential does not need to be updated during the SCF iteration. Therefore, it is efficient for processors associated with the element E_K to own a distributed portion of the projection vector $b_{I,\ell}(x)$ on the LGL grid constructed on E_K, if $b_{I,\ell}(x)$ does not vanish on E_K. This allows the inner product $\langle b_{I,\ell}, \varphi_{K,j}\rangle_{\mathcal{T}}$ to be entirely evaluated on processors associated with E_K without further inter-element communication. The computation of the matrix element $\left(\sum_{I,\ell}\gamma_{I,\ell}\langle\varphi_{K',j'}, b_{I,\ell}\rangle_{\mathcal{T}}\langle b_{I,\ell}, \varphi_{K,j}\rangle_{\mathcal{T}}\right)$, however, requires data communication between processors associated with E_K and $E_{K'}$, on which $b_{I,\ell}$ does not vanish. This is done by communicating the inner products of the form $\langle b_{I,\ell}, \varphi_{K,j}\rangle_{\mathcal{T}}$ among such neighboring elements. Since the size of such inner products is independent of the LGL grid size, the communication volume of this step is relatively low. For inter-element parallelization, we use asynchronous data communication routines with MPI_Isend and MPI_Irecv for efficient data communication.

The boundary integrals that appear in step 3 also require inter-element communication. However, the inter-element data

This article is copyrighted as indicated in the article. Reuse of AIP content is subject to the terms at: http://scitation.aip.org/termsconditions. Downloaded to IP: 169.229.58.198 On: Tue, 29 Sep 2015 17:50:29

communication only occurs among two neighboring elements $E_K, E_{K'}$ if the two elements share a common surface S. It should be noted that the computation of average and jump operators only requires values of functions on the surface S, and therefore only the function values of Φ_K and $\Phi_{K'}$ together with their gradients $\nabla\Phi_K$ and $\nabla\Phi_{K'}$ restricted to S need to be computed. These calculations require much lower communication volume compared to those required in volume integrals. We also remark that the communication performed in this step can be overlapped with computation to further reduce the cost of communication. In particular, we first launch the communication needed to carry out step 3 before starting to perform the computational tasks in step 1, which does not require inter-element data communication.

Numerical results indicate that the overall procedure for constructing the DG Hamiltonian matrix is highly efficient. For instance, for the phosphorene P_{3500} system, the total wall clock time used to construct the DG Hamiltonian matrix is only 1.11 and 0.84 s when the construction is carried out on 12 800 and 25 600 processors, respectively.

3. Computation of the electron density

As discussed in Section II A, the electron density can be assembled from the eigenvector coefficients $\{c_{i;K,j}\}$, or from the diagonal matrix blocks of the density matrix $P_{K;K}$ directly. Both options are supported in DGDFT. When the eigenvector option is used, we diagonalize the DG Hamiltonian matrix (referred to as the "DIAG" method) by using the ScaLAPACK software package.[38] In order to use ScaLAPACK, we need to convert the block row partition of the DG Hamiltonian matrix (see Fig. 1) to the 2D block cyclic data distribution scheme required by ScaLAPACK. The eigenvectors returned from ScaLAPACK, which are stored in the 2D block cyclic format, are redistributed according to the block row partition used to distribute H^{DG}. We developed a routine to perform these conversions. Such conversion is *the only* operation in DGDFT that involves MPI communication among, in principle, all the available processors. All other MPI communications are performed either within the global row processor group or the global column processor group. We note that a similar conversion procedure also can be found in other electronic structure software packages such as SIESTA[6] and CP2K[16] when ScaLAPACK is used.

Once the eigenvectors are redistributed by elements, the electron density can be evaluated locally on each element. The global electron density implies the collection of the density defined on each local element. Such global electron density is never collected to be stored on a single processor, but is distributed across processors in a global row processor group.

It should be noted that for large systems, all ScaLAPACK diagonalization routines (such as the divided and conquer routine PDSYEVD currently used in DGDFT) have limited parallel scalability. They often cannot make efficient use of the maximum number of processors (which can be 10 000–100 000 or more) that can be used by other parts of the DGDFT calculation. Therefore, we often need to restrict the ScaLAPACK calculation to a subset of processors to avoid getting sub-optimal performance.

C. Pole expansion and selected inversion method

When the electron density is computed from the expression given in (8), we use the recently developed PEXSI method[28–30] to compute the diagonal blocks of the density matrix. This technique avoids the diagonalization procedure which has an $O(N^3)$ complexity. It is accurate for both insulating and metallic systems. Furthermore, the computational complexity of the PEXSI method is only $O(N)$ for 1D systems, $O(N^{1.5})$ for 2D systems, and $O(N^2)$ for 3D systems. Therefore, the PEXSI method is particularly well suited for studying electronic structures of large scale low-dimensional (1D and 2D) systems.[39,40]

The PEXSI method is based on approximating the density matrix by a linear combination of Green's functions, i.e.,

$$P \approx \sum_{l=1}^{Q} \mathrm{Im}\left[\omega_l (H^{\mathrm{DG}} - z_l I)^{-1}\right], \tag{12}$$

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}. In practical calculations, we observe that it is often sufficient to choose ΔE to be much smaller than the true spectrum width of H^{DG}, thanks to the exponential decay of the Fermi-Dirac function above the Fermi energy. In most cases, it is sufficient to choose $Q = 40 \sim 80$. If we only need the diagonal blocks of the density matrix P, we do not need to compute the entire inverse of $H^{\mathrm{DG}} - z_l I$. Only the diagonal blocks of $(H^{\mathrm{DG}} - z_l I)^{-1}$ need to be computed, and these diagonal blocks can be computed efficiently by using the selected inversion method.[28] The use of selected inversion leads to favorable complexity of the PEXSI method.

The PEXSI method can scale to 10 000–100 000 processors. This has recently been demonstrated in the massively parallel SIESTA-PEXSI method.[30,39] SIESTA-PEXSI uses local atomic orbitals to discretize the Kohn-Sham Hamiltonian. Because DGDFT is designed to take advantage of massively parallel computers, the high scalability of PEXSI, in addition to its lower asymptotic complexity, makes it a more attractive option compared to the diagonalization option. When the PEXSI option is used, DGDFT can often scale to 10 000–100 000 processors, and be used to solve the electronic structure problems with more than 10 000 atoms efficiently.[40]

In order to use PEXSI in DGDFT, the DG Hamiltonian matrix needs to be redistributed from a block row distribution format to a distributed compressed sparse column format (DCSC). When the DCSC format is used to distribute a sparse Hamiltonian matrix among M processors, each processor holds roughly $\lfloor n/M \rfloor$ columns stored in a compressed sparse column (CSC) format, where n is the dimension of the matrix. The diagonal blocks of the density matrix returned from PEXSI, which are stored in DCSC format, are converted back to the block row partition format used to represent the electron density.

Because the selected inversions of the shifted DG Hamiltonians $(H^{\mathrm{DG}} - z_l I)$ are independent of each other for different poles z_l, we can carry out these selected inversions among different global row processor groups. Therefore, unlike the data redistribution scheme used for ScaLAPACK. As a result,

This article is copyrighted as indicated in the article. Reuse of AIP content is subject to the terms at: http://scitation.aip.org/termsconditions. Downloaded to IP: 169.229.58.198 On: Tue, 29 Sep 2015 17:50:29

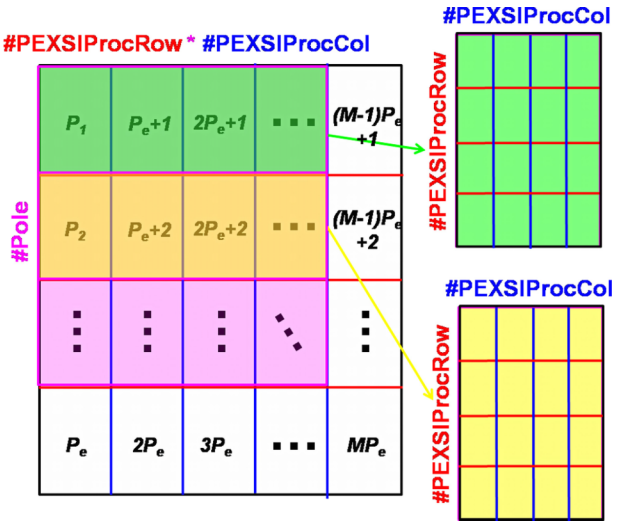

FIG. 6. The processor grid used in the PEXSI method. The processors used in PEXSI are usually a subset of all available processors. #Pole represents the number of poles used in the PEXSI method. The inset shows that each 1D global row processor group is further partitioned into a 2D processor group used for each selected inversion.

the procedure for redistributing H^{DG} required by PEXSI uses asynchronous communication only within a global row processor group.

Because each parallel selected inversion requires processors to be arranged logically into a 2D grid of size (#PEXSIProcRow) × (#PEXSIProcCol), the processors within each global row processor group are reconfigured in PEXSI as shown in Fig. 6. Note that restricting the selected inversion at each pole to processors belonging to a global processor row group limits the maximum number of processors that can be used for each selected inversion to the number of elements M. However, for systems of large size, M can be easily 1000 or more. Therefore, such a configuration does not severely limit the number of processors that can be effectively used for selected inversion. If the number of processors (P_e) within each global column processors group is larger than the number of poles Q, all selected inversions required in pole expansion (12) can be carried out simultaneously. The number of processors that can effectively be used in PEXSI is MQ. If $Q > P_e$, we have to process at most P_e poles at a time, and repeat the process until selected blocks of $(H - z_l I)^{-1}$ have been computed for all z_l's. In this case and when all M processors in a global processor row group are used for selected inversion, all MP_e processors can be effectively used by PEXSI.

III. COMPUTATIONAL RESULTS

In this section, we report the performance and accuracy of DGDFT when it is applied to 2D phosphorene systems of different sizes.

Phosphorene, a new 2D elemental monolayer,[41–45] has received considerable amount of interest recently after it has been experimentally isolated through mechanical exfoliation from bulk black phosphorus. Phosphorene exhibits some remarkable electronic properties superior to graphene, a well known elemental sp^2-hybridized carbon monolayer.[46–48] For example, phosphorene is a direct semiconductor with a high hole mobility.[41] It has the drain current modulation up to 10^5 in nanoelectronics.[42] These remarkable properties have already been used for wide applications in field effect transistors[43] and thin-film solar cells.[44] Furthermore, up to now, phosphorene is the only stable elemental 2D material which can be mechanically exfoliated in experiments[41] besides graphene. Therefore, it can potentially be used as an alternative to graphene in the future and lead to faster semiconductor electronics.

Fig. 2 shows the atomic configuration of 2D phosphorene monolayer in a 5 × 7 supercell (P_{140}). Other phosphorene models in very large supercells involving thousands or tens of thousands of atoms, such as the 25 × 35 (P_{3500}) and 50 × 70 (P_{14000}) supercells, which we use as test problems in this work, are not shown here. The vacuum space in the X and Y directions is about 10 Å to separate the interactions between neighboring slabs in phosphorene.

All calculations are performed on the Edison platform available at the National Energy Research Scientific Computing (NERSC) center. Edison has 5462 Cray XC30 nodes. Each node has 24 cores partitioned among two Intel Ivy Bridge processors. Each 12-core processor runs at 2.4 GHz, and has 64 GB (Gigabyte) of memory per node. The maximum number of available cores is 131 088 on Edison. In all calculations, we utilize all 24 cores on a computational node.

A. Computational accuracy

We use the conventional plane wave software package ABINIT[14] as a reference to check the accuracy of results from DGDFT. The same exchange-correlation functional of the local density approximation of Goedecker, Teter, Hutter (LDA-Teter93)[49] and the Hartwigsen-Goedecker-Hutter (HGH) norm-conserving pseudopotential[50] are adopted in both ABINIT and DGDFT software packages. We remark that we also implemented generalized gradient functionals such as the Perdew-Burke-Ernzerhof (PBE)[51] functional in DGDFT. DGDFT has been used to demonstrate a 2 × 1 edge reconstruction in armchair phosphorene nanoribbons by using *ab initio* molecular dynamics calculations with the PBE functional.[40]

We first check the accuracy of the total energy and the atomic force of the DGDFT method by using P_{140} shown in Fig. 2 as an example. To simplify our discussion, we define the total energy error per atom ΔE (Hartree/atom) and maximum atomic force error ΔF (Hartree/bohr) as

$$\Delta E = (E^{\mathrm{DGDFT}} - E^{\mathrm{ABINIT}})/N_A$$

and

$$\Delta F = \max_I |F_I^{\mathrm{DGDFT}} - F_I^{\mathrm{ABINIT}}|,$$

respectively, where N_A is the total number of atoms. E^{DGDFT} and E^{ABINIT} represent the total energy computed by DGDFT and ABINIT, respectively, and F_I^{DGDFT} and F_I^{ABINIT} represent the Hellmann-Feynman force on the Ith phosphorus atom in P_{140} computed by DGDFT and ABINIT, respectively. The ABINIT results are obtained by setting the energy cutoff to 200 Hartree for the wavefunction to ensure full convergence. The kinetic energy cutoff (denoted by Ecut) in the DGDFT method

This article is copyrighted as indicated in the article. Reuse of AIP content is subject to the terms at: http://scitation.aip.org/termsconditions. Downloaded to IP: 169.229.58.198 On: Tue, 29 Sep 2015 17:50:29

is used to define the grid size for computing the ALBs as is in standard Kohn-Sham DFT calculations using planewave basis sets. Ecut is also directly related to the LGL integration grid defined on each element and used to perform numerical integration as needed to construct the DG Hamiltonian matrix. The number of LGL grids per direction is set to be twice the number of grid points calculated using Eq. (11) with the same Ecut.

Table I shows that the total energy and atomic forces produced by the DGDFT method are highly accurate compared to the ABINIT results. In particular, the total energy error ΔE can be as small as 3.39×10^{-6} Hartree/atom if the DIAG method is used and 8.12×10^{-5} Hartree/atom if the PEXSI method is used, respectively. Here, 50 poles are used and the accuracy of PEXSI can be further improved by increasing the number of poles. The maximum error of the atomic force can be as small as 1.06×10^{-4} Hartree/bohr when DIAG is used and 1.06×10^{-4} Hartree/bohr when PEXSI is used. These results are obtained when only a relatively small number of ALB functions per atom are used to construct the global DG Hamiltonian. The energy cutoff for constructing the ALBs is set to 200 Hartree in this case. Note that the accuracy of total energy and atomic force in DGDFT depends on both the energy cutoff for local wavefunctions defined on an extended element and the number of ALB functions. We can see from Table I that the accuracy of the total energy and atomic forces both improves as the energy cutoff and the number of ALB functions increases. We also find that the use of the Hellmann-Feynman force can result in accurate force calculation, despite that the contribution from the Pulay force[52] is not included.

The penalty parameter α in Eq. (3) penalizes large jumps of the Kohn-Sham orbitals across element surfaces, so that the resulting electron density is close to being continuous and the numerical scheme is stable. The α-dependence of the errors of the total energy and the atomic force is shown in Fig. 7. We set the energy cutoff to 100 Hartree for generating ALBs, and use 91.43 ALB functions per atom (200 ALB functions per element) to obtain highly accurate result compared to that obtained from ABINIT. We find that the choice of α values has minor effect on the accuracy of the energy and the force.

TABLE I. The accuracy of DGDFT in terms of the total energy error per atom ΔE (Hartree/atom) and the maximum atomic force error ΔF (Hartree/bohr) in the DIAG and PEXSI methods with different kinetic energy cutoff Ecut (Hartree), and number of ALB functions per atom, compared with converged ABINIT calculations. #ALB means the number of ALB functions per atom.

DGDFT P_{140}		DIAG		PEXSI	
Ecut	#ALB	ΔE	ΔF	ΔE	ΔF
20	91.43	5.22×10^{-4}	4.03×10^{-3}	3.22×10^{-4}	4.03×10^{-3}
40	18.28	4.51×10^{-2}	5.97×10^{-2}	4.57×10^{-2}	5.97×10^{-2}
40	27.42	6.67×10^{-4}	2.51×10^{-3}	6.85×10^{-4}	2.52×10^{-3}
40	36.57	1.34×10^{-4}	6.16×10^{-4}	1.59×10^{-4}	6.18×10^{-4}
40	45.71	7.00×10^{-5}	4.00×10^{-4}	6.44×10^{-5}	5.23×10^{-4}
40	91.43	-4.32×10^{-7}	5.93×10^{-4}	1.34×10^{-4}	5.93×10^{-4}
100	91.43	1.59×10^{-5}	1.97×10^{-4}	8.04×10^{-5}	1.97×10^{-4}
200	91.43	3.39×10^{-6}	1.06×10^{-4}	8.12×10^{-5}	1.06×10^{-4}

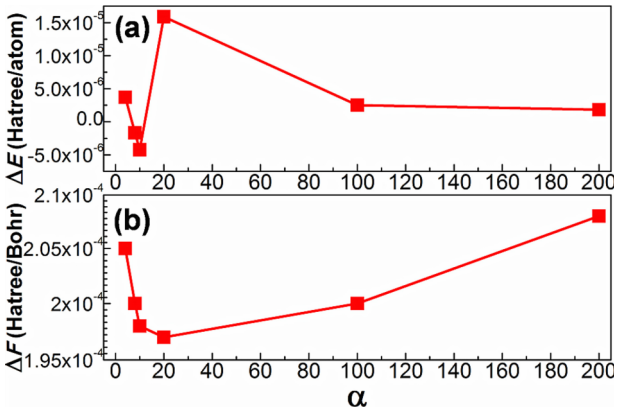

FIG. 7. (a) The total energy error per atom ΔE (Hartree/atom) and (b) the maximum atomic force error ΔF (Hartree/bohr) in the example of P_{140} as a function of penalty parameter α by using DGDFT with the DIAG method. We set the energy cutoff to 100 Hartree for ALBs and 91.43 ALB functions per atom (200 ALB functions per element) to obtain highly accurate results compared to that obtained by ABINIT.

When α is adjusted from 2.0 to 200.0, the change of the error in total energy is less than 1.50×10^{-5} Hartree/atom and the error in atomic forces changes from 1.97×10^{-4} (α = 20.0) to 2.08×10^{-4} (α = 200.0) Hartree/bohr. In the numerical simulation, we use α = 20.0 for 2D phosphorene systems.

In the following parallel efficiency tests, we set the energy cutoff to 40 Hartree for ALBs and 36.57 ALB functions per atom (80 ALB functions per element), which achieves good compromise between accuracy and computational efficiency. For this particular choice of the energy cutoff and the number of ALB functions, we are able to keep the total energy error under 1×10^{-4} Hartree/atom and atomic force error under 1×10^{-3} Hartree/bohr for 2D phosphorene systems.

B. Parallel efficiency

To illustrate the parallel scalability of the DGDFT method, we demonstrate the performance of three main steps in each SCF iteration as shown in Fig. 3: (a) the generation of ALB functions, (b) the construction of DG Hamiltonian matrix, and (c) the evaluation of the approximate charge density, energy, and atomic forces by either diagonalizing the DG Hamiltonian (DIAG) or by using the PEXSI technique. Note that there are some additional steps such as the computation of energy, charge mixing or potential mixing, and intermediate data communication. The cost of these extra steps is included in the total wall clock time.

Fig. 8 shows the strong parallel scaling of these three individual steps, as well as the overall DGDFT method, for two large-scale 2D phosphorene monolayers (P_{3500} and P_{14000}) in terms of the wall clock time per SCF step. For P_{3500}, we tested the performance using both the DIAG and PEXSI methods for the evaluation of charge density during SCF. But for P_{14000}, we only use the PEXSI method, since the DIAG method is too expensive for systems of such size (the dimension of the matrix is 512 000).

The wall clock time of the first two steps is independent of whether PEXSI or DIAG is used. Figs. 8(a) and 8(b) show that they both scale well with respect to the number of cores used in

This article is copyrighted as indicated in the article. Reuse of AIP content is subject to the terms at: http://scitation.aip.org/termsconditions. Downloaded to IP: 169.229.58.198 On: Tue, 29 Sep 2015 17:50:29

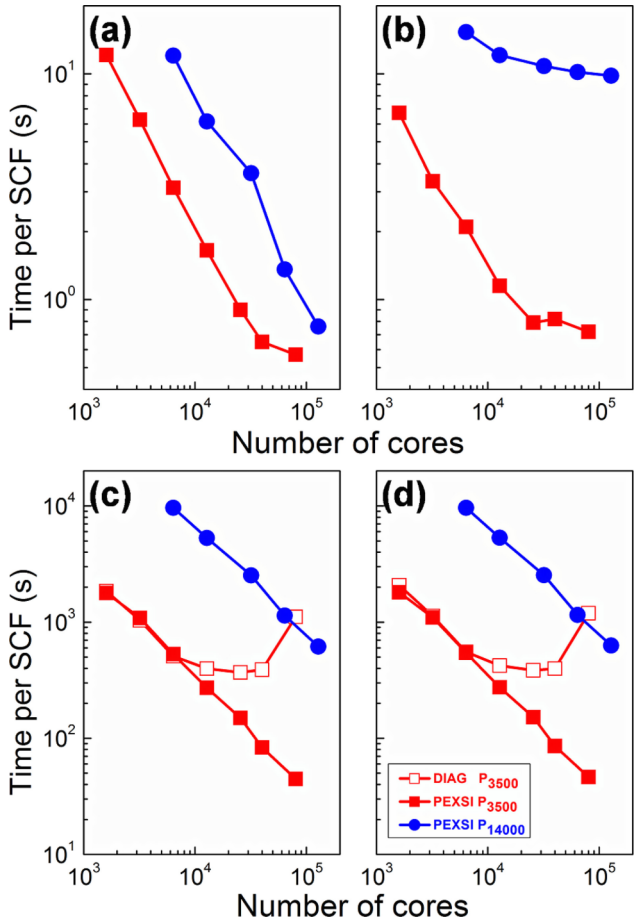

FIG. 8. The wall clock time with respect to the number of computational cores used for 2D phosphorene monolayer systems of different sizes (P_{3500} and P_{14000}). Strong scaling of (a) the generation of ALB functions step, (b) the DG Hamiltonian matrix construction step, (c) the evaluation of the approximate charge density, and (d) the overall computational time, including the time for all previous steps and the remaining overhead. The reported wall clock time is for one SCF iteration. The timing and scaling shown in (c) and (d) depend on whether DIAG (hollow markers) or PEXSI (solid markers) is used to evaluate physical quantities such as charge density, energy, and forces.

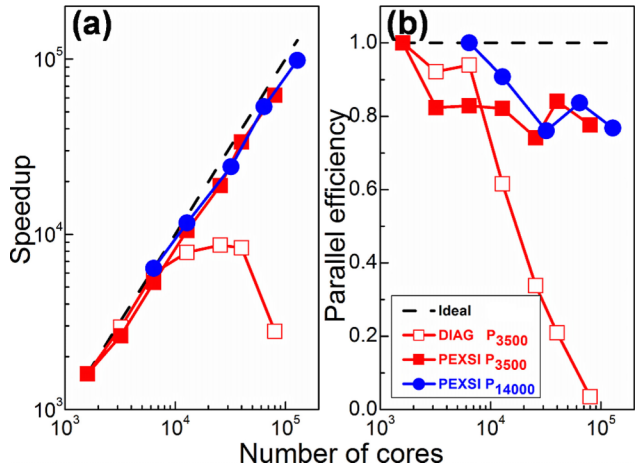

FIG. 9. The (a) speedup and (b) parallel efficiency with respect to the number of cores used for the computation for 2D phosphorene monolayer systems of different sizes (P_{3500} and P_{14000}).

the computation for all test problems we used. The exception is the construction of the DG Hamiltonian which does not scale beyond 10^4 processors for P_{14000}. We find that the only routine does not scale well is the inter-element communication of the boundary values of Φ_K and $\nabla\Phi_K$, which is currently implemented via asynchronous communication. Since the volume of the asynchronous communication is proportional to the system size, for a large system the communication volume may exceed the size of the MPI buffer which leads to sub-optimal performance of the asynchronous data communication. Nonetheless, the cost of the generation of ALBs and the construction of the DG matrix is much less compared to that for computing the electron density from the DG Hamiltonian.

Figs. 8(c) and 8(d) show that the evaluation of the approximate charge density using the DG Hamiltonian matrix dominates the total wall clock time per SCF iteration in the DGDFT methodology for systems of large sizes. For large-scale 2D phosphorene systems P_{3500} and P_{14800}, the PEXSI method can effectively reduce the wall clock time compared to the DIAG method in the DGDFT methodology. Furthermore, using the DIAG method with ScaLAPACK[38] appears to limit the strong parallel scalability to at most 10 000 cores on the Edison. Increasing the cores beyond that can lead to an increase in wall clock time. In contrast, the PEXSI method exhibits highly scalable performance. It can make efficient use of about 100 000 cores on Edison for P_{3500} and P_{14000}. For the phosphorene P_{3500} system, the total computational time is only 46.45 s for one SCF iteration when DG-PEXSI is carried out on 80 000 cores, which is about 10 times faster than the DIAG method. The most expensive part is 44.60 s to evaluate the approximate charge density with the PEXSI method. For the phosphorene P_{14000} system, DG-PEXSI needs 627.71 s for one SCF iteration (616.30 s for the PEXSI part) when DG-PEXSI is carried out on 128 000 cores, which is more expensive than linear scaling DFT methods due to the $O(N^{1.5})$ scaling feature of the PEXSI method. In the future, we will implement the hybrid MPI/OpenMP parallelization for DG-PEXSI to further reduce the wall time.

Fig. 9 shows the speedup and parallel efficiency with respect to the number of cores used for the computation for 2D phosphorene P_{3500} and P_{14000}. For the P_{3500} system, both the DIAG and PEXSI methods in DGDFT can keep highly parallel efficiency (90% for DIAG and 80% for PEXSI) with less than 10 000 cores. But for the DIAG method, further increase of the number of cores will lead to rapid loss of parallel efficiency. On the contrary, the PEXSI method is highly scalable, and its parallel efficiency is about 80% even when the number of cores increases beyond 80 000 for the P_{3500} system. For the large-scale P_{14000} system, we only use the PEXSI method in DGDFT, and we find that the parallel efficiency of the DGDFT-PEXSI method is around 80% when 128 000 cores are used on Edison (Edison has 131 088 cores in total).

IV. CONCLUSIONS

We described a massively parallel implementation of the DGDFT methodology that can be used to perform large-scale Kohn-Sham DFT calculations efficiently. We demonstrated the accuracy and efficiency of our parallel implementation. We

This article is copyrighted as indicated in the article. Reuse of AIP content is subject to the terms at: http://scitation.aip.org/termsconditions. Downloaded to IP: 169.229.58.198 On: Tue, 29 Sep 2015 17:50:29

show that for the 2D phosphorene systems studied here, using 37 basis functions per atom allows us to reach an accuracy level of 1.3×10^{-4} Hartree/atom in terms of the error of energy and 6.2×10^{-4} Hartree/bohr in terms of the error of atomic force, respectively. DGDFT can achieve 80% parallel efficiency on 128 000 high performance computing cores when it is used to study the electronic structure of 2D phosphorene systems with 3500-14 000 atoms. The high parallel efficiency results from two-level parallelization schemes that make use of several different types of data distribution, task scheduling, and data communication schemes. It also benefits from the use of the PEXSI method to compute electron density, energy, and atomic forces. The PEXSI method has a favorable computational complexity and is also amenable to a two-level parallelization scheme that enables it to achieve high parallel efficiency.

ACKNOWLEDGMENTS

This work is partially supported by the Scientific Discovery through Advanced Computing (SciDAC) Program funded by U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research and Basic Energy Sciences (W.H., L.L., and C.Y.), and by the Center for Applied Mathematics for Energy Research Applications (CAMERA), which is a partnership between Basic Energy Sciences and Advanced Scientific Computing Research at the U.S Department of Energy (L.L. and C.Y.). We thank the National Energy Research Scientific Computing (NERSC) center for the computational resources.

[1]P. Hohenberg and W. Kohn, Phys. Rev. **136**, B864 (1964).
[2]W. Kohn and L. J. Sham, Phys. Rev. **140**, B1133 (1965).
[3]S. Goedecker, Rev. Mod. Phys. **71**, 1085 (1999).
[4]H. Shang, H. Xiang, Z. Li, and J. Yang, Int. Rev. Phys. Chem. **29**, 665 (2010).
[5]D. R. Bowler and T. Miyazaki, Rep. Prog. Phys. **75**, 036503 (2012).
[6]J. M. Soler, E. Artacho, J. D. Gale, A. García, J. Junquera, P. Ordejón, and D. Sánchez-Portal, J. Phys.: Condens. Matter **14**, 2745 (2002).
[7]T. Ozaki and H. Kino, Phys. Rev. B **72**, 045121 (2005).
[8]X. Qin, H. Shang, H. Xiang, Z. Li, and J. Yang, Int. J. Quantum Chem. **115**, 647 (2014).
[9]J. VandeVondele, M. Krack, F. Mohamed, M. Parrinello, T. Chassaing, and J. Hutter, Comput. Phys. Commun. **167**, 103 (2005).
[10]M. J. Gillan, D. R. Bowler, A. S. Torralba, and T. Miyazaki, Comput. Phys. Commun. **177**, 14 (2007).
[11]G. Kresse and J. Furthmüller, Phys. Rev. B **54**, 11169 (1996).
[12]G. Kresse and J. Hafner, Phys. Rev. B **47**, 558(R) (1993).
[13]P. Giannozzi, S. Baroni, N. Bonini, M. Calandra, R. Car, C. Cavazzoni, D. Ceresoli, G. L. Chiarotti, M. Cococcioni, I. Dabo *et al.*, J. Phys.: Condens. Matter **21**, 395502 (2009).
[14]X. Gonze, B. Amadon, P.-M. Anglade, J.-M. Beuken, F. Bottin, P. Boulanger, F. Bruneval, D. Caliste, R. Caracas, M. Côté *et al.*, Comput. Phys. Commun. **180**, 2582 (2009).
[15]F. Gygi, IBM J. Res. Dev. **52**, 1 (2008).
[16]J. VandeVondele, U. Borštnik, and J. Hutter, J. Chem. Theory Comput. **8**, 3565 (2012).
[17]D. R. Bowler and T. Miyazaki, J. Phys.: Condens. Matter **22**, 074207 (2010).
[18]C. Saravanan, Y. Shao, R. Baer, P. N. Ross, and M. Head-Gordon, J. Comput. Chem. **24**, 618 (2003).
[19]E. H. Rubensson and E. Rudberg, J. Comput. Chem. **32**, 1411 (2011).
[20]U. Borštnik, J. VandeVondele, V. Weber, and J. Hutter, Parallel Comput. **40**, 47 (2014).
[21]L. Lin, J. Lu, L. Ying, and W. E, J. Comput. Phys. **231**, 2140 (2012).
[22]D. N. Arnold, SIAM J. Numer. Anal. **19**, 742 (1982).
[23]D. N. Arnold, F. Brezzi, B. Cockburn, and L. D. Marini, SIAM J. Numer. Anal. **39**, 1749 (2002).
[24]C. K. Skylaris, P. D. Haynes, A. A. Mostofi, and M. C. Payne, J. Chem. Phys. **122**, 084119 (2005).
[25]S. Mohr, L. E. Ratcliff, P. Boulanger, L. Genovese, D. Caliste, T. Deutsch, and S. Goedecker, J. Chem. Phys. **140**, 204110 (2014).
[26]M. J. Rayson and P. R. Briddon, Phys. Rev. B **80**, 205104 (2009).
[27]J. Kaye, L. Lin, and C. Yang, Commun. Math. Sci. **13**, 1741 (2015).
[28]L. Lin, J. Lu, L. Ying, R. Car, and W. E, Commun. Math. Sci. **7**, 755 (2009).
[29]L. Lin, M. Chen, C. Yang, and L. He, J. Phys.: Condens. Matter **25**, 295501 (2013).
[30]L. Lin, A. García, G. Huhs, and C. Yang, J. Phys.: Condens. Matter **26**, 305503 (2014).
[31]G. Zhang, L. Lin, W. Hu, C. Yang, and J. E. Pask, "Adaptive local basis set for Kohn-Sham density functional theory in a discontinuous Galerkin framework II: Force, vibration, and molecular dynamics calculations" (published online 2015).
[32]L. Kleinman and D. M. Bylander, Phys. Rev. Lett. **48**, 1425 (1982).
[33]N. D. Mermin, Phys. Rev. **137**, A1441 (1965).
[34]L. Lin, J. Lu, L. Ying, and W. E, J. Chem. Phys. **231**, 4515 (2012).
[35]A. V. Knyazev, SIAM J. Sci. Comput. **23**, 517 (2001).
[36]M. C. Payne, M. P. Teter, D. C. Allen, T. A. Arias, and J. D. Joannopoulos, Rev. Mod. Phys. **64**, 1045 (1992).
[37]L. N. Trefethen, *Spectral Methods in MATLAB* (SIAM, 2000), Vol. 10.
[38]T. Auckenthaler, V. Blum, H. J. Bungartz, T. Huckle, R. Johanni, L. Krämer, B. Lang, H. Lederer, and P. R. Willems, Parallel Comput. **37**, 783 (2011).
[39]W. Hu, L. Lin, C. Yang, and J. Yang, J. Chem. Phys. **141**, 214704 (2014).
[40]W. Hu, L. Lin, and C. Yang, Phys. Chem. Chem. Phys. (to be published).
[41]H. Liu, A. T. Neal, Z. Zhu, Z. Luo, X. Xu, D. Tománek, and P. D. Ye, ACS Nano **8**, 4033 (2014).
[42]J. Qiao, X. Kong, Z.-X. Hu, F. Yang, and W. Ji, Nat. Commun. **5**, 4475 (2014).
[43]L. Li, Y. Yu, G. Ye, Q. Ge, X. Ou, H. Wu, D. Feng, X. Chen, and Y. Zhang, Nat. Nanotech. **9**, 372 (2014).
[44]J. Dai and X. C. Zeng, J. Phys. Chem. Lett. **5**, 1289 (2014).
[45]W. Hu, T. Wang, and J. Yang, J. Mater. Chem. C **3**, 4756 (2015).
[46]K. S. Novoselov, A. K. Geim, S. V. Morozov, D. Jiang, Y. Zhang, S. V. Dubonos, I. V. Grigorieva, and A. A. Firsov, Scinece **306**, 666 (2004).
[47]A. K. Geim and K. S. Novoselov, Nat. Mater. **6**, 183 (2007).
[48]A. H. C. Neto, F. Guinea, N. M. R. Peres, K. S. Novoselov, and A. K. Geim, Rev. Mod. Phys. **18**, 109 (2009).
[49]S. Goedecker, M. Teter, and J. Hutter, Phys. Rev. B **54**, 1703 (1996).
[50]C. Hartwigsen, S. Goedecker, and J. Hutter, Phys. Rev. B **58**, 3641 (1998).
[51]J. P. Perdew, K. Burke, and M. Ernzerhof, Phys. Rev. Lett. **77**, 3865 (1996).
[52]P. Pulay, Chem. Phys. Lett. **73**, 393 (1980).

This article is copyrighted as indicated in the article. Reuse of AIP content is subject to the terms at: http://scitation.aip.org/termsconditions. Downloaded to IP: 169.229.58.198 On: Tue, 29 Sep 2015 17:50:29