Journal of Computational Physics 229 (2010) 2339–2363

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

# A massively-parallel electronic-structure calculations based on real-space density functional theory

Jun-Ichi Iwata [a,*], Daisuke Takahashi [a], Atsushi Oshiyama [a,b], Taisuke Boku [a], Kenji Shiraishi [a], Susumu Okada [a], Kazuhiro Yabana [a]

[a] Center for Computational Sciences, University of Tsukuba, 1-1-1 Tenoudai, Tsukuba, Ibaraki 305-8577, Japan
[b] Department of Applied Physics, School of Engineering, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan

ARTICLE INFO

*Article history:*
Received 22 January 2009
Received in revised form 24 November 2009
Accepted 24 November 2009
Available online 5 December 2009

*Keywords:*
Real-space
Finite-difference
Density functional
Electronic-structure
Large scale
Parallel

ABSTRACT

Based on the real-space finite-difference method, we have developed a first-principles density functional program that efficiently performs large-scale calculations on massively-parallel computers. In addition to efficient parallel implementation, we also implemented several computational improvements, substantially reducing the computational costs of $O(N^3)$ operations such as the Gram–Schmidt procedure and subspace diagonalization. Using the program on a massively-parallel computer cluster with a theoretical peak performance of several TFLOPS, we perform electronic-structure calculations for a system consisting of over 10,000 Si atoms, and obtain a self-consistent electronic-structure in a few hundred hours. We analyze in detail the costs of the program in terms of computation and of inter-node communications to clarify the efficiency, the applicability, and the possibility for further improvements.



## 1. Introduction

First-principles calculations based on the density functional theory (DFT) [1,2] have been performed on a variety of materials and have provided important microscopic information for physical properties based on quantum theory [3–5]. Thus, among various theoretical methodologies, DFT is a top choice at present for clarification and prediction of phenomena in condensed matter. The popularity of DFT is due to its relatively low computational costs and its reasonable accuracy, which often favor it over elaborate and highly accurate quantum chemistry approaches such as the configuration-interaction [6] or the diffusion Monte-Carlo approaches [7]. Although there are a few report that the exceptionally large-scale DFT calculations have been achieved [8,9], the usual target systems to which DFT can be applied easily are still limited to medium-sized systems consisting of hundreds to one thousand of atoms. Recent research interests in condensed matter and the material sciences require DFT calculations for much larger systems such as nanoscale systems. For instance, in semiconductor science, the typical size of metal–oxide–semiconductor field-effect transistors is on the nanometer scale. For such systems quantum mechanical simulations are important for understanding device characteristics [10,11]. Furthermore, in the life sciences, correlations between atomic structures and the bio-functions of in vivo proteins are hard to clarify without the help of simulations based on quantum theory [12,13]. Since nanoscale systems consist of 10,000–100,000 atoms or more, it is imperative to perform drastically large-scale calculations based on DFT to address these important subjects, and it is impossible to carry out such large-scale calculations with traditional computational programs. Thus, several computational approaches have been investigated intensively.

* Corresponding author.
*E-mail address:* iwata@comas.frsc.tsukuba.ac.jp (J.-I. Iwata).

0021-9991/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2009.11.038

One promising approach is the order-$N$ method [14] in which mathematical problems are re-formulated so as to utilize the possible localized nature of unitary-transformed wave functions or density matrices. However, another approach exists in which the conventional order-$N^3$ exact formulation is adopted and drastic improvements are achieved in the performance of the computer codes through restructuring and tuning for state-of-the-art computer architectures.

For the latter approach, the real-space finite-difference pseudopotential method proves to be a key ingredient because the methods are suitable for parallel computations. The real-space method for first-principles electronic-structure calculations was first proposed by Chelikowsky et al. in 1994 [15], and then, various developments and applications have been performed by many researchers [8,16–21]. Recently the real-space method have been applied for unprecedentedly large size Si nanocrystals by Zhou et al. [8]. In the real-space methods, singular ionic potentials are replaced by smoother pseudopotentials [22], and Schrödinger-type quantum mechanical equations are discretized on three-dimensional spatial grids and the solutions are produced by treating them as finite-difference (FD) equations. In principle, the matrix of the real-space formulation is sparse and fast Fourier transformation (FFT) is unnecessary for Hamiltonian matrix operations. The FFT-free character contrasts with the conventional plane-wave methods [23,24], and provides a great advantage by easing the communication burden in parallel computations.

To date, the real-space method have achieved the calculations for the systems of thousands of atoms in the order-$N^3$ exact formulation [8,9]. The real-space method is also promising for much larger systems containing 10,000–100,000 atoms. Therefore it is important to understand the computational details of the state-of-the-art real-space method, including the algorithms, implementations, and the performances, for further development of the first-principles calculations with the next-generation supercomputers. In this paper, we present a detailed description of our real-space DFT (RSDFT) code developed recently to overcome the size limitation of our computing system by utilizing the power of massively-parallel computing. We also investigate in detail the computational costs of RSDFT code to clarify how to study large systems using next-generation supercomputers.

The algorithm employed in our code is rather conventional, so that the computational costs scale as $O(N^3)$. However, there are several benefits to develop the code based on the conventional algorithm. First, we already know its applicability, accuracy, and suitable choices of computational parameters such as cut-off energies and sampling $k$ points. Second, we are able to concentrate on particular problems in large-scale calculations, such as numerical precision, convergence behavior, and the reliability of the total-energy calculation itself. The present study focuses on the computational aspects in large-scale real-space calculations. Thus, the programming techniques developed for RSDFT are also applicable to other methods, e.g. order-$N$ methods.

In Section 2, we briefly review the basics and fundamental equations of DFT. In Section 3, we present a similar formulation to that of Section 2, but in a discrete space for real-space FD calculations. Although the Sections 2 and 3 are somewhat repetitious for the specialists of the DFT calculations, we add them by the following reason; for future development of the first-principles calculations, we must need the help of the specialists of computer and computational sciences to bring out the best performance of the supercomputers, and we aim to remove the barrier at the entrance of the DFT calculations for the non-DFT specialists. In Section 4, we summarize the computational parameters and the several initial configurations for the parallel computation. In Section 5, we describe the overall algorithm of our RSDFT code and the details of several main subroutines. We introduce a new algorithm to accelerate $O(N^3)$ operations in Gram–Schmidt orthogonalization and in subspace diagonalization. In Section 6, we present the performance tests of our code and analyze the costs of computation and communication. In Section 7, we show several practical applications dealing with Si crystal, nanometer-scale Si quantum dots, and Si nanowires. Finally, we present a summary and conclusion in Section 8. Acronyms used in this paper are summarized in Table 1.

## 2. Density functional theory

The ultimate purpose of DFT calculations is to minimize an energy functional $E[\rho]$ with respect to the electron density $\rho$. Following the standard Kohn–Sham DFT formalism [2], we introduce the orbital $\phi_n$, and assume that the electron density is expressed as a sum of the absolute square of the orbitals

$$\rho(\mathbf{r}) = \sum_{n=1}^{M_B} f_n |\phi_n(\mathbf{r})|^2 \tag{1}$$

and the minimization is performed with respect to the orbitals. In Eq. (1), $M_B$ is the total number of orbitals and $f_n$ is the occupation number for the $n$th orbital. The explicit form of the energy functional in terms of the orbitals is

$$\begin{aligned} E[\phi] = {} & \sum_{n=1}^{M_B} f_n \int d\mathbf{r} \phi_n^*(\mathbf{r}) \left( -\frac{1}{2}\nabla^2 \right) \phi_n(\mathbf{r}) + \frac{1}{2} \int d\mathbf{r} \int d\mathbf{r}' \frac{\rho(\mathbf{r})\rho(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|} + E_{xc}[\rho] + \int d\mathbf{r} v_L(\mathbf{r})\rho(\mathbf{r}) \\ & + \sum_{n=1}^{M_B} f_n \int d\mathbf{r} \int d\mathbf{r}' \phi_n^*(\mathbf{r}) v_{NL}(\mathbf{r},\mathbf{r}') \phi_n(\mathbf{r}'), \end{aligned} \tag{2}$$

where the density $\rho$ is given by Eq. (1), and $E_{xc}$ is the exchange-correlation energy. $E_{xc}$ represents many-electron effects and its exact form is unknown. However, several approximate functionals are available [25,26]. The $v_L$ and $v_{NL}$ are the local and

**Table 1**
List of acronyms used throughout this paper.

| Acronym | Meaning |
|---|---|
| DFT | Density functional theory |
| FD | Finite-difference |
| RSDFT | Real-space density functional theory (the name of our code described in this paper) |
| GS | Gram–Schmidt |
| CG | Conjugate-gradient |
| SD | Subspace diagonalization |
| RQ | Rayleigh quotient |
| SCF | Self-consistent field |
| MPI | Message passing interface |
| BLAS | Basic Linear Algebra Subprograms |
| SCALAPACK | Scalable Linear Algebra Package |
| PACS-CS | Parallel array computer system for computational sciences (the name of the massively-parallel cluster at Center for Computational Sciences, University of Tsukuba) |
| DOS | Density of states |

non-local pseudopotentials that describe the interaction between ions and valence electrons [22]. The use of the pseudopotential is crucial for reducing the computational costs, because it removes the core states from the calculation, and makes the potentials smooth. Typically, the non-local potential is further approximated as [27]

$$v_{NL}(\mathbf{r},\mathbf{r}') = \sum_{a=1}^{N}\sum_{l=0}^{L_a}\sum_{m=-l}^{l} C_{alm}p_{alm}(\mathbf{r})p^*_{alm}(\mathbf{r}'), \tag{3}$$

where $N$ is the number of atoms, $L_a$ is the maximum angular momentum (typically 0–2), $C_{alm}$ is the coefficient, and $p_{alm}$ is the projector function that is localized within a spherical region $\Omega_a$ around each atom. Explicitly, $p_{alm}$ is

$$p_{alm}(\mathbf{r})\begin{cases} \neq 0 & (\mathbf{r}\in\Omega_a),\\ = 0 & (\mathbf{r}\notin\Omega_a).\end{cases} \tag{4}$$

The energy minimization must be performed under the ortho-normalization constraints

$$\int d\mathbf{r}\phi^*_m(\mathbf{r})\phi_n(\mathbf{r}) = \delta_{mn}. \tag{5}$$

Therefore, it is convenient to recast the minimization problem from the constrained form into the unconstrained form by introducing the Lagrange multipliers $\varepsilon_{mn}$:

$$\frac{\delta}{\delta\phi^*_m}\left\{E[\phi] - \sum_{m=1}^{M_B}\sum_{n=1}^{M_B}\varepsilon_{mn}\int d\mathbf{r}\phi^*_m(\mathbf{r})\phi_n(\mathbf{r})\right\} = 0. \tag{6}$$

From Eq. (6), we obtain the Kohn–Sham equation for the orbitals

$$\left(-\frac{1}{2}\nabla^2 + v_h(\mathbf{r}) + v_{xc}(\mathbf{r}) + v_L(\mathbf{r}) + \hat{v}_{NL}\right)\phi_n(\mathbf{r}) = \sum_{n=1}^{M_B}\varepsilon_{mn}\phi_n(\mathbf{r}), \tag{7}$$

where the operator $\hat{v}_{NL}$ is defined as

$$\hat{v}_{NL}\phi(\mathbf{r}) = \int d\mathbf{r}' v_{NL}(\mathbf{r},\mathbf{r}')\phi(\mathbf{r}'). \tag{8}$$

In Eq. (7), $v_h$ is the Hartree (or Coulomb) potential which can be written in integral form as

$$v_h(\mathbf{r}) = \int d\mathbf{r}'\frac{\rho(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|} \tag{9}$$

or in differential form as

$$\nabla^2 v_h(\mathbf{r}) = -4\pi\rho(\mathbf{r}) \tag{10}$$

in which case it is called the Poisson equation. In Eq. (7), $v_{xc}$ is the exchange-correlation potential which is defined as a functional derivative of the exchange-correlation energy

$$v_{xc}(\mathbf{r}) = \frac{\delta E_{xc}}{\delta\rho(\mathbf{r})}. \tag{11}$$

Typically, the approximate exchange-correlation functionals [25,26] are given their explicit functional form, so that the functional derivatives can be calculated in a straightforward manner.

Since the Lagrange multipliers can be expressed in the diagonal form through a unitary transformation of the orbitals, the Kohn–Sham equation can be rewritten in the simpler form

$$\left(-\frac{1}{2}\nabla^2 + v_h(\mathbf{r}) + v_{xc}(\mathbf{r}) + v_L(\mathbf{r}) + \hat{v}_{NL}\right)\phi_n(\mathbf{r}) = \varepsilon_n\phi_n(\mathbf{r}). \tag{12}$$

Solving Eq. (12) is the main task of DFT calculations. Since $v_h$ and $v_{xc}$ depend on the density, and, through Eq. (1), also depend on the orbitals, we must solve Eq. (12) self-consistently.

In addition to the orbitals, the atomic coordinates $\{\boldsymbol{R}_1, \boldsymbol{R}_2, \ldots\}$ are also variables of the DFT energy functional. Thus, we can perform the atomic-structure optimization by minimizing the energy functional with respect to the atomic coordinates. Usually, the energy functional has several local minima where the gradient vector with the coordinate of atom $\alpha$

$$\boldsymbol{g}_\alpha = -\frac{\partial E}{\partial \boldsymbol{R}_\alpha} \tag{13}$$

is zero for all $\alpha$, and each local minimum corresponds to a (meta-)stable structure of the system. Once the Kohn–Sham equation is solved for a fixed atomic configuration, then the gradient vector can be calculated using

$$\boldsymbol{g}_\alpha = -\int d\boldsymbol{r}\rho(\mathbf{r})\frac{\partial v_L(\mathbf{r})}{\partial \boldsymbol{R}_\alpha} - \sum_{n=1}^{M_B} f_n \int d\boldsymbol{r}\int d\boldsymbol{r}'\phi_n^*(\mathbf{r})\frac{\partial v_{NL}(\mathbf{r},\mathbf{r}')}{\partial \boldsymbol{R}_\alpha}\phi_n(\mathbf{r}'). \tag{14}$$

This is called the Helmann–Feynman force [28], and is suitable for numerical force calculations because the derivatives of the local and non-local pseudopotentials are easily obtained from their analytical expressions.

## 3. DFT on three-dimensional grid space

To numerically minimize the energy functional, we introduce a three-dimensional spatial grid and consider the minimization problem within the discrete space of $M_L$ grid points. Then the orbitals, density, and potentials are expressed as column vectors whose elements are the value at each grid point. For example,

$$\vec{\phi} = \begin{pmatrix} \phi^1 \\ \vdots \\ \phi^i \\ \vdots \\ \phi^{M_L} \end{pmatrix}, \tag{15}$$

where the $i$th element is the value at the grid point $\mathbf{r}_i$:

$$\phi^i = \phi(\mathbf{r}_i). \tag{16}$$

In this discrete scheme, the energy functional can be written as

$$\begin{aligned} E[\phi^1, \phi^2, \ldots, \phi^{M_L}] = &-\frac{1}{2}\sum_{n=1}^{M_B} f_n \sum_{i=1}^{M_L}\sum_{j=1}^{M_L} \phi_n^{i*} L_{ij}\phi_n^j \Delta\Omega - \frac{4\pi}{2}\sum_{i=1}^{M_L}\sum_{j=1}^{M_L}\rho^i L_{ij}^{-1}\rho^j\Delta\Omega + E_{xc}\left[\rho^1, \rho^2, \ldots, \rho_L^M\right] + \sum_{i=1}^{M_L} v_L^i \rho^i \Delta\Omega \\ &+ \sum_{n=1}^{M_B} f_n \sum_{i=1}^{M_L}\sum_{j=1}^{M_L} \phi_n^{i*} V_{ij}^{NL}\phi_n^j \Delta\Omega, \end{aligned} \tag{17}$$

where $\Delta\Omega$ is the volume element. The matrix $\{L_{ij}\}$ is the Laplacian in discrete space; i.e., an FD operator. The formula for the higher-order FD coefficients, from which the matrix $\{L_{ij}\}$ is constructed, is given in Appendix A, and the Laplacian in oblique-coordinate systems is given in Appendix B. $V^{NL}$ is the matrix for the non-local pseudopotential whose operation is defined as

$$\sum_{j=1}^{M_L} V_{ij}^{NL}\phi^j = \sum_{a=1}^{M_I}\sum_{l=0}^{L_a}\sum_{m=-l}^{l} C_{alm} p_{alm}^i \sum_{j\in\Omega_a} p_{alm}^{j*}\phi^j \Delta\Omega. \tag{18}$$

The second term in the right-hand side of Eq. (17) represents the Hartree energy. This form can be derived from that the Hartree potential $\vec{v}_h$, which satisfies the Poisson equation (in discrete space):

$$L\vec{v}_h = -4\pi\vec{\rho}. \tag{19}$$

The density is defined as

$$\rho^i = \sum_{n=1}^{N} f_n \left|\phi_n^i\right|^2. \tag{20}$$

Ortho-normalization constraints for the orbitals can be written as

$$\sum_{i=1}^{M_L} \phi_n^{i*} \phi_m^i \Delta\Omega = \delta_{mn}. \tag{21}$$

Minimizing the following functional,

$$\frac{\partial}{\partial \phi_n^{i*}} \left\{ E[\phi^1, \phi^2, \ldots, \phi^{M_L}] - \sum_{m=1}^{M_B} \sum_{n=1}^{M_B} \varepsilon_{mn} \sum_{i=1}^{M} \phi_n^{i*} \phi_m^i \Delta\Omega \right\} = 0, \tag{22}$$

we finally reach the eigenvalue equation for the orbitals

$$H\vec{\phi}_n = \varepsilon_n \vec{\phi}_n, \tag{23}$$

where

$$H = -\frac{1}{2}L + V + V^{NL} \tag{24}$$

is the Hamiltonian, $L$ is a sparse matrix of the FD operator, $V^{NL}$ is the matrix of the non-local operator Eq. (18), and $V$ is the diagonal matrix whose elements are the sum of the local potentials $\vec{v}_h$, $\vec{v}_{xc}$, and $\vec{v}_L$.

The Helmann–Feynman force of an atom $\alpha$ can be calculated from

$$\boldsymbol{g}_\alpha = -\sum_{i=1}^{M_L} \rho^i \frac{\partial v_L^i}{\partial \boldsymbol{R}_\alpha} \Delta\Omega - \sum_{n=1}^{M_B} f_n \sum_{i=1}^{M_L} \sum_{j=1}^{M_L} \phi^{i*} \frac{\partial V_{ij}^{NL}}{\partial \boldsymbol{R}_\alpha} \phi^j \Delta\Omega\Delta\Omega. \tag{25}$$

## 4. Computational set up and parallelization

The systems for which we chose to perform DFT calculations may be classified into two categories: (1) a system with periodic boundary conditions for orbitals and (2) a system with decaying boundary conditions. Crystalline solids or supercell geometries are typical of the former, and molecules or clusters belong to the latter. For both systems, we first must set the number $N$, the position, and the species of each atom in the unit cell. Since we are usually interested in the behavior of chemically active valence electrons, we treat the nucleus and inert core electrons together as an ion, and the interaction between the ion and valence electrons is described with the pseudopotential [22], as explained above. The pseudopotentials are prepared by pre-processing DFT calculations, and stored to external files. In the RSDFT code, we begin by reading the pseudopotential data from the external files, and then we prepare the pseudopotentials on each grid point by interpolation. Usually, we do not use the raw data for the interpolation, but rather the data preconditioned by Fourier-filtering [29] or the double-grid method [30]. These preconditionings are essential to reduce the so-called egg-box effect which is a spurious energy dependence on atomic position relative to neighboring grid points. Another preconditioning method exists in which the high-frequency Fourier components are perfectly filtered out by constructing the pseudopotentials in reciprocal space. This method is standard in plane-wave DFT code, but it is only applicable to periodic systems. However, localization in reciprocal space corresponds to delocalization in real space, and thus the integral region for the inner products extends over the unit cell. Consequently, the computational costs for the pseudopotential operation become $O(N^3)$. With our code, we have the choice of the two (for molecular or cluster systems) or three (for periodic systems) preconditioning methods.

From the number of atoms and their species, we get the number of valence electrons needed to neutralize the positive ionic charges. To perform the calculations for charged systems, we give the number of excess or deficit electrons as an input (a fractional number is also allowed). The total number of electrons $N_e$ in the system is the sum of them. The number of orbitals $M_B$ is determined as $M_B \geqslant N_e/2$ (we concentrate on the spin-unpolarized case in this paper). The occupation numbers $f_n$ in Eq. (1) are determined from the corresponding eigenvalues of the Kohn–Sham equation, so that

$$\sum_{n=1}^{M_B} f_n = N_e, \quad f_n = \frac{2}{1 + \exp\left(\frac{\varepsilon_n - \varepsilon_F}{k_B T}\right)}, \tag{26}$$

where $\varepsilon_F$ is the Fermi energy, which determines the border between occupied and unoccupied Kohn–Sham eigen states. $k_B T$ is a fictitious temperature which controls the broadening of the occupation.

Once the ionic coordinates are specified, we can construct an initial electron density as a superposition of the pseudo-atom densities, which are accompanied with the pseudopotential calculation [22]. Initial Hartree and exchange-correlation potentials are calculated from the initial electron density, and initial orbitals are prepared using a random-number generator and ortho-normalized by Gram–Schmidt procedure.

For solids, or all periodic systems, the unit cell, which is a parallelepiped, is the domain of the orbitals, and the orbitals are periodically connected from the unit cell to the adjacent unit cell. The unit cell is specified by the three lattice vectors $\vec{a}_1$, $\vec{a}_2$, and $\vec{a}_3$. In the real-space method, we define the spatial grid by dividing each lattice vector by an integer, so that the grid-spacing $H_i$ $(i = 1, 2, 3)$ takes the form

$$H_i = \frac{|\vec{a}_i|}{M_i}. \tag{27}$$

The total number of grid points is $M_L = M_1 M_2 M_3$, and the volume element is $\Delta\Omega = \Omega/M_L$, where $\Omega$ is the volume of the unit cell. The Kohn–Sham equations for periodic systems must be solved under the boundary condition for the $\vec{a}_1$ direction

$$\phi(i + M_1, j, k) = \phi(i, j, k). \tag{28}$$

Similar conditions apply for the $\vec{a}_2$ and $\vec{a}_3$ directions. It is easy to divide the unit cell, which is a parallelepiped domain, into equal-shaped sub-domains, making the number of grid points equal in each sub-domain. This is essential to achieve good load balances in parallel computations.

For molecules, or all isolated systems such as atoms and clusters, we introduce a finite domain outside of which the orbitals are essentially zero. To surround the molecular geometry with a sufficiently large vacuum region, we choose a spherically or cylindrically shaped domain. We employ a Cartesian coordinate system in which the origin is set at the center of the sphere (or cylinder), and then we divide the *x*-, *y*-, and *z*-directions by a grid-spacing $H$ and use the grid points within the domain. Contrary to the case for a parallelepiped unit cell, it is difficult to divide a spherical or cylindrical domain into equal-shaped sub-domains. In this case, we are obliged to vary the shape of each sub-domain so that the number of grid points in each sub-domain remains as nearly equal as possible.

In addition to three-dimensional periodic systems (solids) and zero-dimensional isolated systems (molecules), there are one-dimensional systems (wires and tubes) and two-dimensional systems (surfaces and interfaces) in nature, and these are also important targets in the condensed matter physics and the material science. Usually, these systems are treated in supercell models with periodic boundary conditions or cluster models with decaying boundary conditions. Although these models have provided many productive results, the boundary conditions are obviously artificial and we have to take care of the effect of the artifacts for concluding physical results. The natural boundary conditions for the one- and two-dimensional systems are combinations of the periodic and the decaying ones. Contrary to the conventional plane-wave method, the real-space method can easily realize such mixed boundary conditions suitable for one- and two-dimensional systems. We just refer to [31,32] for more information on such methods.

## 5. Details of algorithms

Minimization of the energy functional is two-fold. One minimization is with respect to the electron density or the orbitals, and the other minimization is with respect to the atomic coordinates. In our computational code, the minimization with respect to atomic coordinates contains the minimization with respect to the orbitals for fixed atomic coordinates as an internal procedure. This corresponds to the Born–Oppenheimer approximation or the adiabatic approximation. The most time-consuming operation is the minimization of the energy functional with respect to the orbitals. This operation is equivalent to solving the Kohn–Sham equation (12) or (23). In this section, we focus on our computational strategy to solve Eq. (23), and explain several main subroutines in detail. The method for the energy functional minimization with respect to the atomic coordinates will be summarized in Appendix C.

Eq. (23) appears as a standard eigenvalue problem for the Hermitian matrix of Eq. (24). However, since the Kohn–Sham Hamiltonian depends on its eigenfunctions through the density equation (1), we must satisfy the following *self-consistency* conditions for the electron density

$$\rho^{\text{output}}[\rho^{\text{input}}] - \rho^{\text{input}} = 0 \tag{29}$$

or the same condition can be written in terms of the potential

$$V[\rho^{\text{output}}] - V[\rho^{\text{input}}] = 0. \tag{30}$$

In these equations, $\rho^{\text{input}}$ and $\rho^{\text{output}}$ are the input and output electron densities, respectively. The latter is constructed from the updated orbitals using Eq. (1). These equations are solved using Pulay's direct inversion in the iterative subspace method [33] or using Broyden's methods [34].

First, we consider the minimization of the energy functional with respect to the orbitals under a fixed potential. This is equivalent to solving the eigenvalue problem equations (7) or (12). It is essential to note that, in the Kohn–Sham formulation of DFT, only the small number of eigen-states corresponding to electron-occupied states is required for constructing the electron density through Eq. (1). Therefore, the subspace iteration method [35] is suitable to solve the Kohn–Sham eigenvalue problem. The conjugate-gradient method [36,37] and the residual-minimization method [24] are often used in the electronic-structure calculations, but the effort to find the best algorithm have been continued. The Chebyshef-filtering method proposed by Zhou et al. [8] seems promising for large-scale electronic-structure calculations. Many other algorithms also exist and they have been compared for the characteristics and the performances have been investigated in Ref. [38].

Since the performance of the algorithms generally depend on the problems and tunings, it is difficult to say which one is the best for the electronic-structure calculations at present. The present RSDFT code is based on a modified version of the conjugate-gradient algorithm described in Ref. [36]. In order to find the subspace spanned by the eigenvectors associated with the smallest $M_B$ eigenvalues of the Hamiltonian in $M_L$ dimension, we minimize the Rayleigh quotients (RQ),

$$\varepsilon_n = \frac{\langle \phi_n | H | \phi_n \rangle}{\langle \phi_n | \phi_n \rangle}. \tag{31}$$

Here we introduce for convenience the bra-ket notation, defined as

$$\begin{aligned} \langle \phi_n | H | \phi_n \rangle &= {}^t\vec{\phi}_n^* H \vec{\phi}_n \Delta\Omega, \\ \langle \phi_n | \phi_n \rangle &= {}^t\vec{\phi}_n^* \vec{\phi}_n \Delta\Omega. \end{aligned} \tag{32}$$

Since a local RQ minimum corresponds to an eigen-solution of the equation

$$H\vec{\phi}_n = \varepsilon_n \vec{\phi}_n, \tag{33}$$

we can obtain an approximate subspace through the minimization of RQ. We perform the minimization using the conjugate-gradient (CG) method, but other approaches may be used to obtain an approximate subspace [24].

After the RQ minimization, we ortho-normalize the orbitals using the Gram–Schmidt (GS) procedure, and then we refine the subspace through subspace diagonalization (SD) (or Rayleigh–Ritz projection). Approximate eigenvectors are obtained as the resultant Ritz vectors, and are used as the initial vectors for the next iteration. We call the above procedure the Rayleigh–Ritz (RR) iteration, and we perform this iteration until the subspace converges sufficiently to be that spanned by the eigenvectors.

We now address the issue of self-consistent calculations. During the RR iteration, we also perform another iterative calculation that ensures self-consistency among the orbitals, density, and potentials, as mentioned above. We update the density and the potentials after the GS procedure, and then we perform SD with the updated potentials. We have experienced that the convergence behavior of self-consistent calculations improves when the update is performed between the GS and the SD procedures. We call the RR iteration with the update of the density and the potentials, the self-consistent field (SCF) iteration. The flow chart of the SCF iteration is summarized in Fig. 1.

In the following subsections, we present in detail the computation and communication algorithms of several main subroutines.

### 5.1. Conjugate-gradient minimization

Let us consider that we have $M_B$ ortho-normalized orbitals as an initial guess for the $M_B$ eigenfunctions of the Hamiltonian, and then we update the orbitals so as to minimize the RQs of Eq. (31) while maintaining the ortho-normalization condition equation (21) to the extent possible. The algorithm of the CG minimization can be found elsewhere [4,36], and we also give a brief summary of CG and the preconditioning in Appendix D.

Although the ortho-normalization constraints can be imposed explicitly in the CG procedure, we omit them for several reasons. First, the computational cost for ortho-normalization is extremely high. Second, the update of each orbital can be performed independently, so that parallel computation becomes possible for each orbital. Third, at a local minimum, the RQs remain unchanged under first-order variations of the orbital, so that components of other eigenfunctions are hardly mixed with the eigenfunction being optimized in the CG procedure. We have found that the CG procedure without the ortho-normalization constraints actually works well.

The number of CG iterations for each orbital is limited to 3–5, because the convergence efficiency is unchanged for more iterations. Most of the computational costs are incurred by Hamiltonian operations, calculations of inner products, and the preconditioning operation. The operation of the Hamiltonian matrix is performed through a subroutine call, the details of which are the subject of the next subsection.

### 5.2. Hamiltonian operation

The Hamiltonian equation (24) is constructed from three operators: FD operators, and local and non-local potential operators. In Cartesian coordinates the operation of the FD Laplacian on an orbital is performed as

$$\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}\right)\phi(x_i, y_i, z_i) \approx \sum_{m=-M_D}^{M_D} C_m \phi(x_i + mH_x, y_i, z_i) + \sum_{m=-M_D}^{M_D} C_m \phi(x_i, y_i + mH_y, z_i) + \sum_{m=-M_D}^{M_D} C_m \phi(x_i, y_i, z_i + mH_z), \tag{34}$$

where $H_x,\ H_y,\ H_z$ are the grid spacings in each direction, and $C_m$'s are the coefficients of the FD. When the order $M_D$ formula is employed, we refer to the values of the orbital at $3 \times 2M_D$ points near the point $(x_i, y_i, z_i)$, so that the matrix of the FD Laplacian has at most only $6M_D + 1$ non-zero elements per row. This means that the FD matrix is sparse. The coefficients of the higher-order FD and its accuracy is discussed in Appendix A. In Fig. 2, we show what effects the grid-spacing choice and the FD order have on a total-energy calculation.

In the parallel FD computation, the orbital values near a boundary between two sub-domains are exchanged via the communications with the message passing interface (MPI) library. For a single orbital, the number of data sent to and received from a neighbor is $M_D \times M_s$, where $M_s$ is the number of grid points within the areas in contact between the two sub-domains.

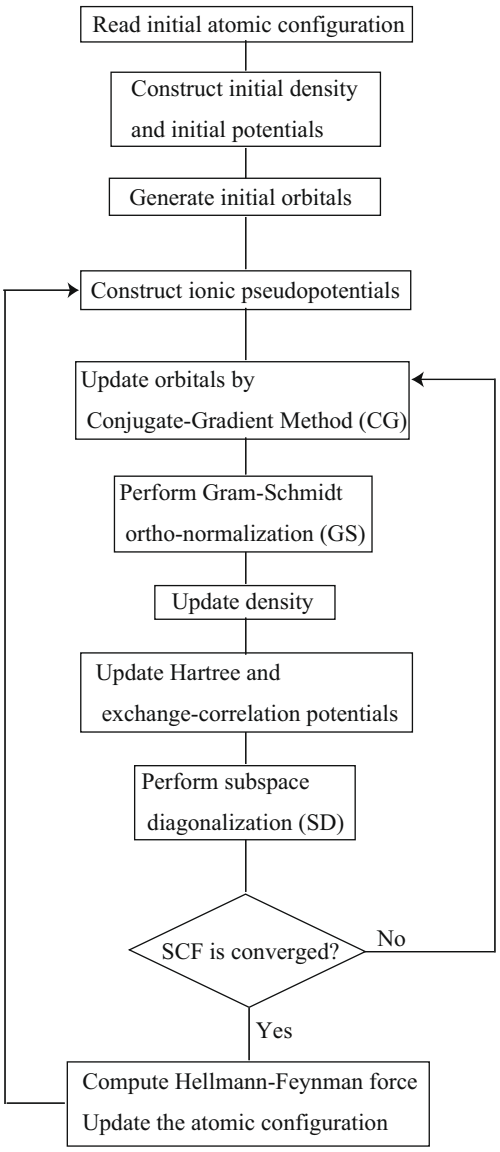


**Fig. 1.** Flow chart of the SCF calculation and the atomic-structure optimization.

Several possible manners exist for sending and receiving the data for $M_B$ orbitals: $M_B$ communications with $M_D \times M_s$ data, a single communication with $M_B \times M_D \times M_s$ data, or several communications with an appropriate set of packed data. The cost of the communications will be discussed later.

Since the local potential matrix is diagonal, its operation on an orbital is straightforward:

$$(V\vec{\phi})^i = v(\mathbf{r}_i)\phi(\mathbf{r}_i). \tag{35}$$

The operation of the non-local potential equation (18) is constructed in two steps. First, we compute an inner product between a projector function $p_{alm}$ and an orbital $\phi$,

$$\beta_{alm} = C_{alm} \int_{\Omega_a} d\mathbf{r} p_{alm}(\mathbf{r})\phi(\mathbf{r}) \approx C_{alm} \sum_{j \in \Omega_a} p^{j*}_{alm} \phi^j \Delta V \tag{36}$$

and then we compute the linear combination

$$(V_{NL}\vec{\phi})^i = \sum_{a=1}^{N} \sum_{l=0}^{L_a} \sum_{m=-l}^{l} p^i_{alm} \beta_{alm}, \tag{37}$$

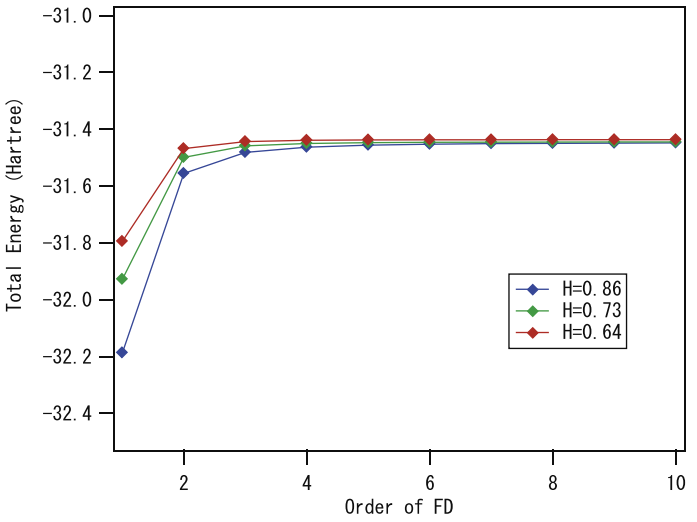


**Fig. 2.** Dependence of the total-energy on the order of the finite-difference for several grid-spacings $H$. The system is cubic Si in the diamond structure. The unit of the energy and the grid-spacing is in atomic unit.

where the summation index $a$ covers the number of ions $N$, and $L_a$ is usually taken to be 0–2. Thus, the non-local potential operator has $(L_a+1)^2 \times N$ terms. The matrix of the non-local operator is not sparse, but the operation can be performed through the two vector operations described above. Furthermore, the summation for the inner product equation (36) is always performed within a small region around each ion irrespective of the system size. Therefore, the computational cost of the non-local potential operation on an orbital is $O(N)$. The integral of the inner product equation (36) is performed within a small region. However, there is a case that the integral region extends over several nearest-neighbor nodes. In that case, MPI communications thus occur to perform the inner product within a group of the relevant nodes (sub-domains). The details of the communication cost will be discussed later.

### 5.3. Gram–Schmidt orthogonalization

After the CG minimization, we recover the ortho-normalization relations among the orbitals through the GS procedure. Since the GS procedure is widely applied in many fields in the computational sciences, several algorithms have been proposed to perform these calculations efficiently [39–41]. We have also developed a GS algorithm suitable for the RSDFT [42], and the computational cost of the GS is thereby substantially reduced. A brief explanation of the GS algorithm adopted in the present code is given below:

$$
\begin{aligned}
&\vec{\varphi}_1 = \vec{\phi}_1,\\
&\vec{\varphi}_2 = \vec{\phi}_2 - \vec{\varphi}_1\left(\vec{\varphi}_1^* \cdot \vec{\phi}_2\right),\\
&\vec{\varphi}_3 = \vec{\phi}_3 - \vec{\varphi}_1\left(\vec{\varphi}_1^* \cdot \vec{\phi}_3\right) - \vec{\varphi}_2(\vec{\varphi}_2^* \cdot \vec{\phi}_3),\\
&\vec{\varphi}_4 = \vec{\phi}_4 - \vec{\varphi}_1\left(\vec{\varphi}_1^* \cdot \vec{\phi}_4\right) - \vec{\varphi}_2\left(\vec{\varphi}_2^* \cdot \vec{\phi}_4\right) - \vec{\varphi}_3\left(\vec{\varphi}_3^* \cdot \vec{\phi}_4\right),\\
&\vec{\varphi}_5 = \vec{\phi}_5 - \vec{\varphi}_1\left(\vec{\varphi}_1^* \cdot \vec{\phi}_5\right) - \vec{\varphi}_2\left(\vec{\varphi}_2^* \cdot \vec{\phi}_5\right) - \vec{\varphi}_3\left(\vec{\varphi}_3^* \cdot \vec{\phi}_5\right) - \vec{\varphi}_4\left(\vec{\varphi}_4^* \cdot \vec{\phi}_5\right),\\
&\vec{\varphi}_6 = \vec{\phi}_6 - \vec{\varphi}_1\left(\vec{\varphi}_1^* \cdot \vec{\phi}_6\right) - \vec{\varphi}_2\left(\vec{\varphi}_2^* \cdot \vec{\phi}_6\right) - \vec{\varphi}_3\left(\vec{\varphi}_3^* \cdot \vec{\phi}_6\right) - \vec{\varphi}_4\left(\vec{\varphi}_4^* \cdot \vec{\phi}_6\right) - \vec{\varphi}_5\left(\vec{\varphi}_5^* \cdot \vec{\phi}_6\right),\\
&\quad\vdots
\end{aligned}
\tag{38}
$$

Eq. (38) is the usual algorithm of the GS procedure (we omitted the normalization steps for simplicity). At first sight, the operations seem to be constructed from inner products and scalar-by-vector products. However, a part of the operations can be grouped and regarded as matrix-by-matrix products. As an example, consider the case that we have three ortho-normalized orbitals $\{\vec{\varphi}_1, \vec{\varphi}_2, \vec{\varphi}_3\}$, and we want to ortho-normalize the remaining orbitals $\{\vec{\varphi}_4, \vec{\varphi}_5, \vec{\varphi}_6\}$. The first four terms of the right-hand side of the last three lines of Eq. (38) can be rewritten as

$$
(\vec{\phi}_4, \vec{\phi}_5, \vec{\phi}_6) - (\vec{\varphi}_1, \vec{\varphi}_2, \vec{\varphi}_3)\begin{pmatrix} {}^t\vec{\varphi}_1^* \\ {}^t\vec{\varphi}_2^* \\ {}^t\vec{\varphi}_3^* \end{pmatrix}(\vec{\phi}_4, \vec{\phi}_5, \vec{\phi}_6),
\tag{39}
$$

showing that we can perform part of the calculation using matrix products. The above rearrangement of the GS algorithm can improve the performance substantially, because matrix products can be performed extremely efficiently on modern computers by employing a highly tuned linear-algebra library (e.g. level 3 of Basic Linear Algebra Subprograms (BLAS)) [43].

In Fig. 3, we illustrate the algorithm described above. The region to compute is shown in Fig. 3 as a triangular part of an $(M_B-1)\times(M_B-1)$ matrix. In this triangle, we first define a $(M_B-1)/2\times(M_B-1)/2$ square in which matrix multiplication is possible. Two smaller triangles remain above and to the right side of the square. In these smaller triangles, we also define squares, and we continue the procedure recursively, until the size of the resulting square is less than $4\times 4$. The computational performance is superior for larger squares. However, for large-sized systems, there is a possibility that the largest square is not treatable due to memory limitations. In this case, we give as input the maximum square size, and the square part of the calculations are performed by dividing into several small squares of the desired size.

Our GS algorithm is termed the classical GS. Although the accuracy of classical GS is occasionally problematic, we confirm that it works well within double precision accuracy, even for a large-system with 20,000 orbitals. The size of the entire vector space, which is equal to the number of grid points $M_L$, is about 100 times larger than the number of orbitals $M_B$, and we apply the GS procedure only for the $M_B$ orbitals. In such a situation, the classical GS does not suffer from numerical inaccuracy.

### 5.4. Subspace diagonalization

After the CG and GS, we perform SD. In SD, we reduce the $M_L$-dimension eigenvalue problem an $M_B$-dimension problem,

$$\begin{pmatrix} H_{1,1} & H_{1,2} & \cdots & H_{1,MB} \\ H_{2,1} & H_{2,2} & \cdots & H_{2,MB} \\ \vdots & \vdots & & \vdots \\ H_{MB,1} & H_{MB,2} & \cdots & H_{MB,MB} \end{pmatrix} \begin{pmatrix} c_n^1 \\ c_n^2 \\ \vdots \\ c_n^{MB} \end{pmatrix} = \varepsilon_n \begin{pmatrix} c_n^1 \\ c_n^2 \\ \vdots \\ c_n^{MB} \end{pmatrix}, \tag{40}$$

where $H_{i,j}$ is

$$H_{i,j} = {}^t\vec{\varphi}_i^* H \vec{\varphi}_j \Delta\Omega. \tag{41}$$

We solve Eq. (40) for all eigenvalues and eigenvectors. Finally, we compute the Ritz vectors as

$$\vec{\phi}_n = c_n^1\vec{\varphi}_1 + c_n^2\vec{\varphi}_2 + \cdots + c_n^{MB}\vec{\varphi}_{MB}. \tag{42}$$

We adopt the $M_B$ Ritz vectors as the approximate eigenvectors, and also use them as initial vectors for the next RR iteration.

Since the matrix appearing in Eq. (40) is Hermitian, we need only compute the matrix elements for the lower (or upper) triangular part:

$$\begin{array}{ccccc} {}^t\vec{\varphi}_1^*\vec{h}_1 & & & & \\ {}^t\vec{\varphi}_2^*\vec{h}_1 & {}^t\vec{\varphi}_2^*\vec{h}_2 & & & \\ {}^t\vec{\varphi}_3^*\vec{h}_1 & {}^t\vec{\varphi}_3^*\vec{h}_2 & {}^t\vec{\varphi}_3^*\vec{h}_3 & & \\ {}^t\vec{\varphi}_4^*\vec{h}_1 & {}^t\vec{\varphi}_4^*\vec{h}_2 & {}^t\vec{\varphi}_4^*\vec{h}_3 & {}^t\vec{\varphi}_4^*\vec{h}_4 & \\ {}^t\vec{\varphi}_5^*\vec{h}_1 & {}^t\vec{\varphi}_5^*\vec{h}_2 & {}^t\vec{\varphi}_5^*\vec{h}_3 & {}^t\vec{\varphi}_5^*\vec{h}_4 & {}^t\vec{\varphi}_5^*\vec{h}_5 \\ & & \cdots & & \end{array}. \tag{43}$$

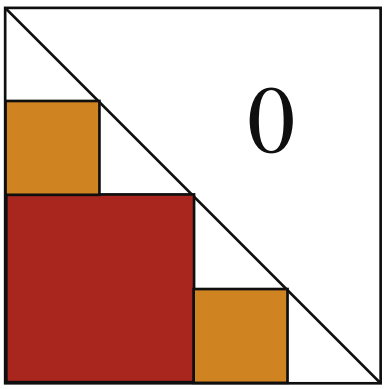


**Fig. 3.** A schematic illustration for an idea to perform the GS calculation in matrix-by-matrix product form recursively. We have to compute the lower triangular part of $M_B$-dimensional matrix, and the squares represent the parts where matrix-by-matrix computations are possible.

In this equation, $\vec{h}_i = H\vec{\varphi}_i \Delta\Omega$. The number of matrix elements is $\frac{1}{2}M_B(M_B+1)$, so the computational cost for matrix construction scales as $O\left(M_L M_B^2\right) \sim O(N^3)$.

The first three terms of the last three lines of Eq. (43) can be rewritten as

$$\begin{pmatrix} {}^t\vec{\varphi}_3 \\ {}^t\vec{\varphi}_4 \\ {}^t\vec{\varphi}_5 \end{pmatrix} (\vec{h}_1 \quad \vec{h}_2 \quad \vec{h}_3) \tag{44}$$

so that, as done for GS, a part of the calculation can be gathered and performed as matrix products instead of inner products. The construction of the Ritz vectors is also performed as matrix products:

$$(\vec{\phi}_1 \quad \vec{\phi}_2 \quad \cdots \quad \vec{\phi}_{MB}) = (\vec{\varphi}_1 \quad \vec{\varphi}_2 \quad \cdots \quad \vec{\varphi}_{MB})(\vec{c}_1 \quad \vec{c}_2 \quad \cdots \quad \vec{c}_{MB}). \tag{45}$$

The computational cost for the Ritz vectors also scales as $O\left(M_L M_B^2\right) \sim O(N^3)$.

Before constructing the Ritz vectors, we diagonalize the dense $M_B$-dimensional Hermitian matrix. Although the dimension of the matrix ($M_B$) is only 1% of the dimension of the original Hamiltonian matrix ($M_L$), the memory requirement and computational costs of the diagonalization become prohibitively high for extremely large systems. Therefore, we also perform a parallel computation for the $M_B$-dimensional diagonalization. For this purpose, we employ the divide-and-conquer eigensolver for Hermitian (or symmetric) matrices in the Scalable Linear Algebra Package (SCALAPACK) library [44]. In order to use SCALAPACK, distribute a part of the matrix to each node, and gather the distributed eigenvector to a node. Thus, we need additional MPI_REDUCE and MPI_BCAST communications to construct the matrix elements and Ritz vectors, respectively.

## 6. Performance tests

The predominant portion of the total computational time in RSDFT calculations results from the SCF iterations. Therefore, in this section we investigate in detail the costs of computations and communications for SCF iterations. To this end, we present several test calculations performed by the RSDFT code for periodic systems. The test systems are cubic Si crystals in the diamond structure, with sizes of 512, 1000, 1728, 2744, and 4096 atoms. Only the gamma point is sampled for the Brillouin zone integration, allowing us to define the orbitals and related variables as double precision real numbers. The grid-spacing is chosen as 0.45 Å. These computational parameters are found to provide fairly accurate results for the electronic-structure. The maximum number of iterations in the CG routine is fixed at 3. The calculations are performed on a massively-parallel cluster, the PACS-CS, at University of Tsukuba, the properties of which can be found elsewhere [45].

As shown in Fig. 4, most of the computational time is spent for the GS, SD, and CG routines in a single step of the SCF iterations. The computational costs of GS and SD routines scale as $O(N^3)$. The CG routine is constructed from Hamiltonian operations, preconditioning processes, and the formation of vector inner products. All these computational costs scale as $O(N^2)$. However, the computational time of the CG routine is comparable to that of $O(N^3)$ routines, because the performance of the computations in the CG routine is substantially lower than those in the $O(N^3)$ routines, in which most of the computations can be performed with level 3 BLAS.

In the following subsections, we discuss the costs of computations and communications incurred by the main subroutines in more detail through theoretical estimations and practical test calculations.

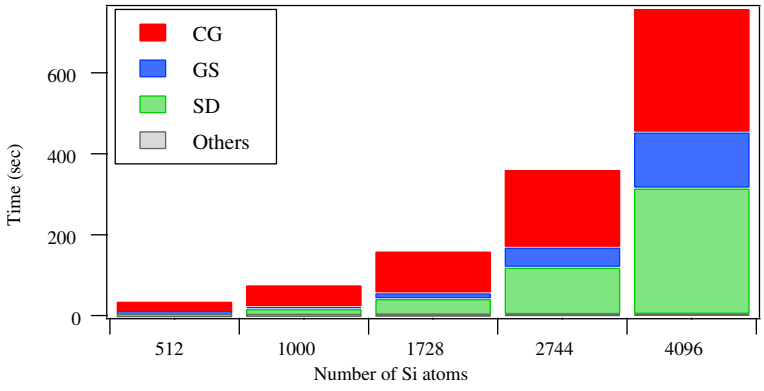


**Fig. 4.** Computational time for a single step of the SCF iterations and the items for each subroutine. The test system is cubic Si in diamond structure with the number of atoms of 512–4096.

### 6.1. Gram–Schmidt ortho-normalization

The number of floating point operations (FLOPs) for GS using $N_P$ processors can be written as

$$\frac{M_B(M_B-1)}{2}\left(4\frac{M_L}{N_P}+1\right)+M_B\left(3\frac{M_L}{N_P}+1\right), \tag{46}$$

where the first term and the second term come from orthogonalization and normalization, respectively. Since both $M_B$ and $M_L$ are proportional to the number of atoms $N$, the computational cost of GS scales as $O\left(M_B^2 M_L\right) \sim O(N^3)$. When the orbitals are double-complex variables, the number of FLOPs in (46) increases four times.

In the GS procedure, we need to compute $M_B(M_B-1)/2+M_B$ inner products of orbitals of $M_L$-dimension, and the inner products require inter-node communications for MPI_ALLREDUCE. Assuming the binary-tree algorithm, the cost of the communication for an MPI_ALLREDUCE can be written as

$$T_{\text{Allreduce}} = (L + D/B) \times \log_2 N_P, \tag{47}$$

where $L$ is the latency (in s), $D$ is the data size (in Bytes), $B$ is the band width (in Bytes/s), and $N_P$ is the number of CPUs. In the GS algorithm described in Section 5.3, the number of calls for MPI_ALLREDUCE is $2M_B$ which is the same as that in the naive algorithm of classical GS. For the $2M_B$-times MPI_ALLREDUCE, the total data size is

$$8\frac{M_B(M_B-1)}{2}+8M_B \quad \text{(Bytes)}. \tag{48}$$

In the double-complex calculations, the data size is twice that of Eq. (48). The total cost of communication for MPI_ALLREDUCE thus becomes

$$T_{\text{Allreduce}} = \left(2M_B \times L + \frac{8M_B(M_B-1)/2+8M_B}{B}\right) \times \log_2 N_P, \tag{49}$$

which scales as $O\left(M_B^2 \log_2 N_P\right) \sim O(N^2 \log_2 N_P)$. From Eqs. (46) and (49), we notice that as the number of CPUs increases, the cost of the computation decreases as $\sim 1/N_P$, while the cost of the communication increases as $\sim \log_2 N_P$. Thus, a crossover point exists at a certain number of CPUs where the cost of communication becomes higher than that of computation.

In Figs. 5 and 6, we show the time required for computation and for communication in the GS routine for the systems with 512 atoms and 1728 atoms, respectively. For the 512-atom system, the communication time becomes longer than the computational time at 128 CPUs. In other words, the granularity of the computation becomes too small by the parallelization with 128 CPUs. For the 1728-atom system, the computational cost increases by about $(1728/512)^3 \approx 38$ times of that of the 512-atom system, and the communication time is always smaller than the computational time up to parallelization with 256 CPUs (see Fig. 6).

### 6.2. Hamiltonian operation

The operation of the Hamiltonian matrix on an orbital is constructed from the finite-difference, local potential, and non-local potential operators. When the order of the finite-difference is $M_D$, the number of FLOPs for the finite-difference is

$$9\frac{M_L}{N_P}M_D M_B. \tag{50}$$

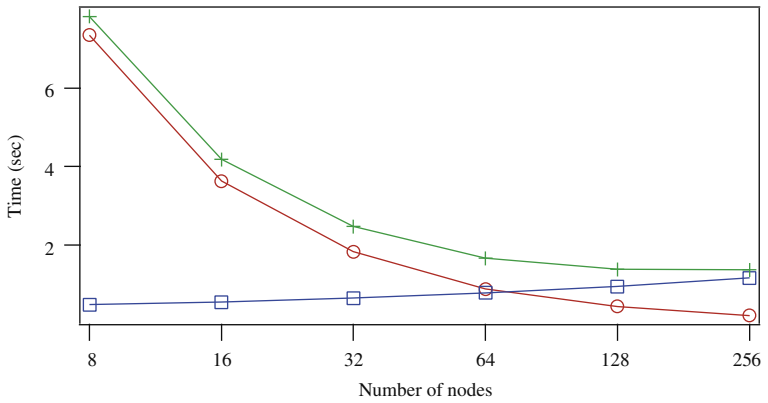


**Fig. 5.** Times for GS calculation for the bulk Si crystal of 512 atoms. The plots with squares, circles, and crosses represent the times for communication, computation, and the sum of the two, respectively.

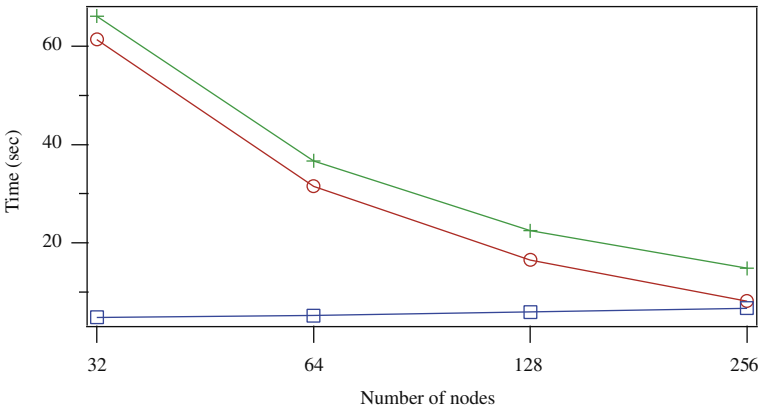


**Fig. 6.** Times for GS calculation for the bulk Si crystal of 1728 atoms. The plots with squares, circles, and crosses represent the times for communication, computation, and the sum of the two, respectively.

In order to exchange the boundary-values with the nearest-neighbor processors, the amount of data sent and received per direction is

$$D = 8\left(\frac{M_L}{N_P}\right)^{2/3} \times M_D \times M_B \quad \text{(Bytes)}. \tag{51}$$

The data is exchanged only with the six nearest neighbors irrespective of the number of nodes. Thus, the total cost of the communication is

$$T = 6 \times \left( L \times M_B + 8\left(\frac{M_L}{N_P}\right)^{2/3} M_D M_B \times \frac{1}{B} \right). \tag{52}$$

The local potential, which is the sum of the ionic pseudopotentials, Hartree potential, and exchange-correlation potential, is a diagonal matrix, and the operation is straightforward. The number of FLOPs is

$$2\frac{M_L}{N_P} \times M_B \tag{53}$$

and the operations are purely local; i.e., no inter-node communication takes place.

To estimate the computational costs for the non-local operations, we need to know the number of non-local operators and the number of grid points within a range of each operator. However, since these parameters depend on the other parameters in complex ways, we use approximate values that are estimated from the configuration of the atoms in each sub-domain of the unit cell and the range of each non-local operator. The form of the non-local operator is described in Eq. (37), and the total number of FLOPs for the non-local potential operations for $M_B$ orbitals can be written as

$$4 \times \frac{N N_{prj} \Omega_a}{\Delta V} \frac{1}{N_P} \times M_B, \tag{54}$$

where $N_{prj}$, which typically ranges from 1 to 16, is the number of projector functions per atom. $\Omega_a$ is a spherical region, where the integration of an inner product equation (36) is performed. It can be defined using a cut-off radius $R_a$ as

$$\Omega_a = \frac{4\pi}{3} R_a^3. \tag{55}$$

$\Omega_a/\Delta V$ is the number of grid points within this region. When the integral region extends over several nodes, MPI-communication is necessary to complete the inner products. As shown in the schematic illustration in Fig. 7, the number of relevant nodes that share the integral regions is almost unchanged with respect to the total number of nodes, as long as the size of each sub-domain is not too small. Since the data size is also insensitive to the number of nodes, the communication cost is almost constant irrespective of the number of nodes required for parallelization. We employ MPI_ISEND and MPI_IRECV for the inner product computations instead of MPI_ALLREDUCE, because the integral regions are closed within the nearest few nodes. The communication cost can thus be estimated as

$$T = 6 \times \left( L + \frac{D}{B} \right) \times M_B, \tag{56}$$

where, as discussed above, the data size $D$ is almost constant.

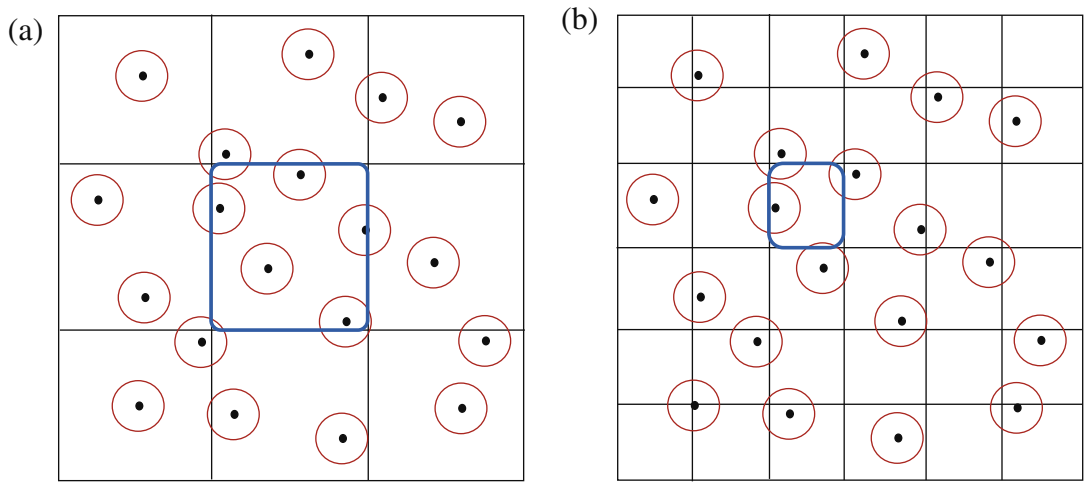


**Fig. 7.** Schematic illustrations to explain the inter-node communication for non-local potential operations. We consider the two-dimensional case for simplicity: (a) is the case for 9-node parallelization, and (b) is the case for 25-node parallelization. The circles represent the range of the non-local operators. In both cases, the number of data exchanged with a neighbor node is not so different.

In Fig. 8, we show the computational time for the three parts of the Hamiltonian operation for the system of 1728 Si atoms. The majority of the time is spent for the finite-difference and the non-local potential operations. For the finite-difference operation in the 1728-atom system, the cost of communication is always higher than the cost of computation, and both costs decrease as the number of nodes increase, as expected from Eqs. (50) and (52). For the non-local potential operation, the computational cost decreases as the number of nodes increases, while the communication cost is almost constant irrespective of the number of nodes.

Fig. 9 shows the performance of the Hamiltonian operation for several system sizes. Since the granularity of the computation increases, performance improves for the larger systems. However, the FLOPs per second (FLOPS) are significantly low compared with the theoretical peak performance value of $5.6 \times N_P$ (GFLOPS), leaving open the possibility of further improving the algorithms. At present, the computational time for the Hamiltonian operation is comparable to that of $O(N^3)$ parts in spite of its $O(N^2)$ scaling.

### 6.3. Subspace diagonalization

This routine is implemented by first constructing the matrix, then solving the eigenvalue problem, and finally constructing the Ritz vectors. The computational cost of each part scales as $O(N^3)$, and inter-node communication occurs in each part. In addition to using MPI, SCALAPACK [44] is also used to solve the eigenvalue problem. Accordingly, the matrices are divided into $N_P$ submatrices, and the resulting eigenvectors are stored in divided form in each node.

For matrix construction, we use the same technique as that used in the GS routine, so that most of the computation can be performed with level 3 BLAS. The number of FLOPs is the sum of

$$\frac{M_B(M_B+1)}{2}\left(2\frac{M_L}{N_P}+1\right) \tag{57}$$

and the FLOPs for Hamiltonian operation discussed in the previous subsection.

Since the matrices have to be divided for the SCALAPACK routine, we use MPI_REDUCE instead of MPI_ALLREDUCE for computing the matrix elements. The number of calls to MPI_REDUCE is almost the same as the number of nodes for parallelization, and the total data size is

$$D = 8\frac{M_B(M_B+1)}{2} \quad \text{(Bytes)}. \tag{58}$$

Therefore, the cost of communication can be estimated as

$$T_{\text{reduce}} = \log_2 N_P \times \left(N_P \times L + 8\frac{M_B(M_B+1)}{2}\frac{1}{B}\right). \tag{59}$$

For the eigenvalue problem, we use the divide-and-conquer eigensolver PDSYEVD and PDHEEVD in SCALAPACK for double precision real calculations and double complex calculations, respectively. With this solver, we compute all the $M_B$ eigenvalues and eigenvectors. A theoretical estimation of the computational cost of the divide-and-conquer eigensolver is $4M_B^3$.

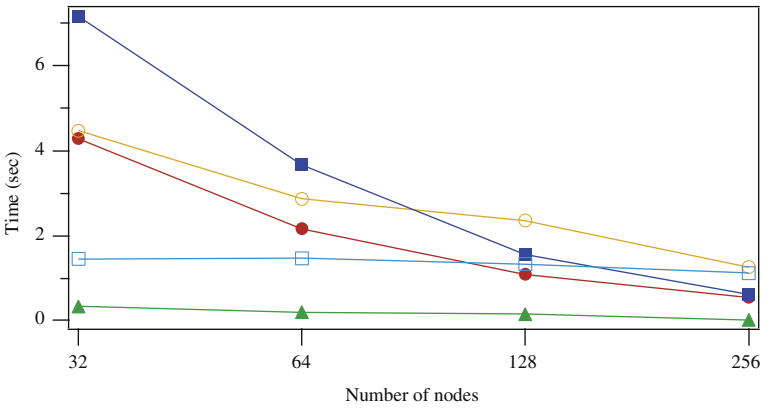


**Fig. 8.** Times for the three operations in the Hamiltonian for the system of cubic Si crystal of 1728 atoms. The bold and open symbols mean the time for computation and communication, respectively, and the square, circle, and triangle symbols mean the time for finite-difference, non-local potential, and local potential operation, respectively.

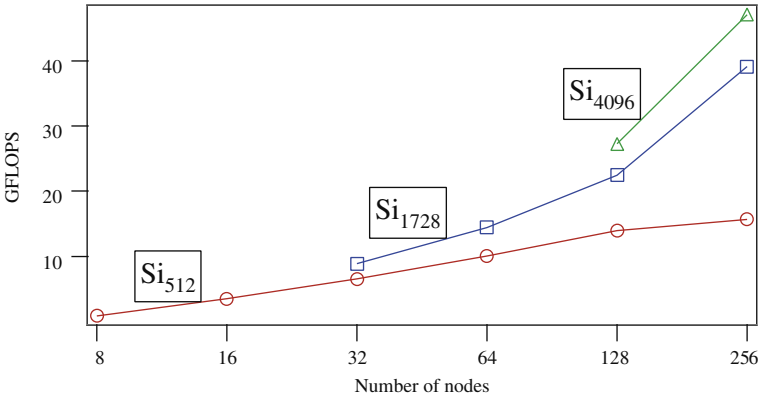


**Fig. 9.** Computational performance of the Hamiltonian operation for the system size of 512, 1728, and 4096 Si atoms. The theoretical peak performance is $5.6 \times N_p$ (GFLOPS), where $N_p$ is the number of nodes.

Fig. 10 shows the performance of the PDSYEVD routine for several system sizes. The parallel efficiency is not good because the performance is significantly affected by the manner in which the matrix is divided.

After solving the eigenvalue problem, we construct the Ritz vectors from the eigenvectors and the orbitals. Since the eigenvectors are obtained using SCALAPACK, they are stored in a divided form in each node. Therefore, we perform MPI_BCAST to distribute a portion of the eigenvectors to all the nodes. The communication cost to accomplish this is estimated as

$$T_{\mathrm{Bcast}} = \log_2 N_P \times \left( L \times N_P + \frac{8M_B^2}{B} \right). \tag{60}$$

The number of FLOPs is

$$2 \times \frac{M_L}{N_P} M_B^2. \tag{61}$$

This computation can also be performed in matrix product form with level 3 BLAS.

In Fig. 11, we show the computational time for several parts of the SD routine. The construction of matrix and Ritz vectors is the most time-consuming part in SD routine. However, since the parallel efficiency of SCALAPACK routine unsatisfactory, the computational cost of the PDSYEVD routine becomes highest at 256 parallel nodes. As expected from Eqs. (59) and (60), the time for MPI_REDUCE and MPI_BCAST increases monotonically as the number of nodes increases.

In Fig. 12, we show the total performance of the SD routine. The GFLOPS/node is 1–3, which is quite good compared with the theoretical peak performance of 5.6 GFLOPS/node.

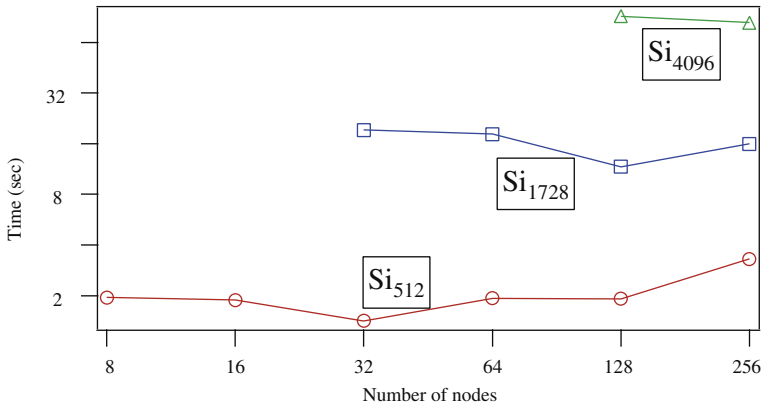


**Fig. 10.** Parallel efficiency of the divide-and-conquer eigensolver PDSYEVD, a subroutine in the SCALAPACK library. The ordinate shows time required for solving the eigenvalue problem with each number of nodes in the abscissa. Computational times for the three systems are shown, namely 512-, 1728-, and 4096-atom systems, and the dimension of the matrix for each system is 1024, 3556, and 8192, respectively.

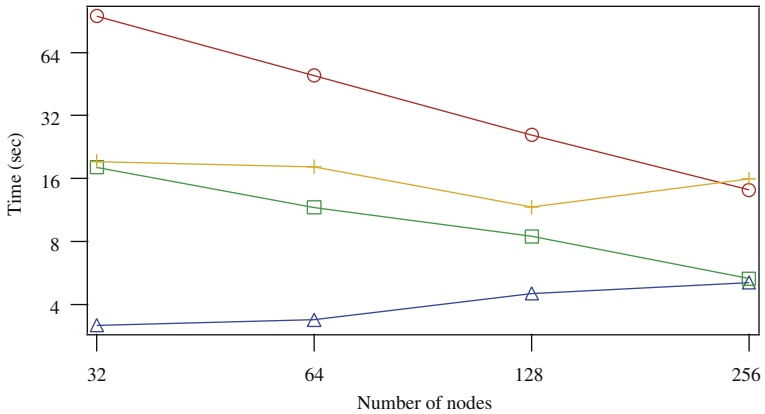


**Fig. 11.** Times for computation and communication in the SD routine for the system of 1728 Si atoms. Circles, crosses, and squares represent the computational time for the construction of matrix, diagonalization (PDSYEVD), and the construction of Ritz vectors, respectively. Triangle symbols represent the time for communication (MPI_REDUCE and MPI_BCAST).

### 6.4. Conjugate-gradient method

In addition to the SCF iteration, we also perform an iterative calculation in the CG routine. The CG iteration is constructed from Hamiltonian operations, the preconditioning operation, and several vector operations including inner products (the inner products also need inter-node communications through MPI_ALLREDUCE). Although the computational costs scale as $O(N^2)$, the performance is unsatisfactory because most of the operations are constructed from vector sums and products, and the communication costs are relatively high. In particular, the preconditioning operation is almost the same as the FD operation in the Hamiltonian, and the number of calls is a few times larger than that of the Hamiltonian. Consequently, the cost for MPI_ISEND and MPI_IRECV becomes high, and the efficiency of parallelization drops. Fig. 13 shows the practical performance of the CG routine, as well as parts of several main operations.

## 7. Practical applications

We begin this section by showing an example of the application of our RSDFT code to a crystalline Si system with 4096 atoms. The crystalline lattice is diamond structure with a lattice constant of 43.4 Å. The grid-spacing is chosen to be 0.45 Å, which corresponds to the lattice constant divided into 96 segments. Thus, the total number of grid points is $96^3 = 884,736$ points. For the ionic potentials, we employ the norm-conserving pseudopotential [22] in the separable approximate form [27], and we treat only the valence orbitals; i.e., the $8192 + \alpha$ orbitals, where $\alpha$ represents additional unoccupied orbitals, of which there are few. The computation is performed on the PACS-CS at the University of Tsukuba [45], and the unit cell is divided into 884 sub-domains by using 256 nodes. The initial orbitals are prepared with a random-number generator, and ortho-normalized using the GS procedure. The initial electron density is constructed from the sum of the pseudo-atomic

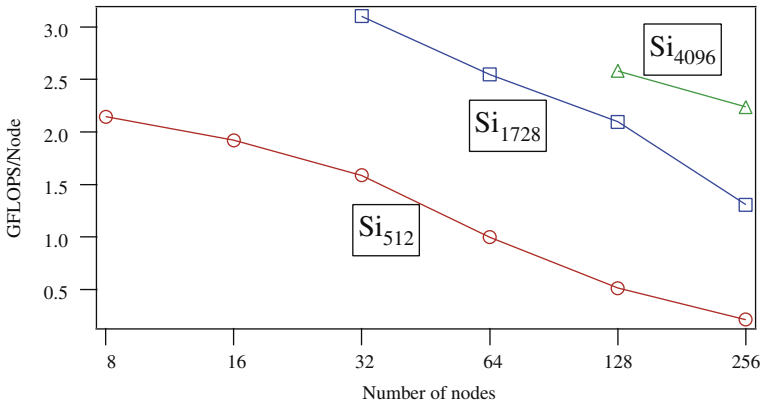


**Fig. 12.** Computational performance of SD routine for several system sizes. The theoretical peak performance is 5.6 (GFLOPS/node).

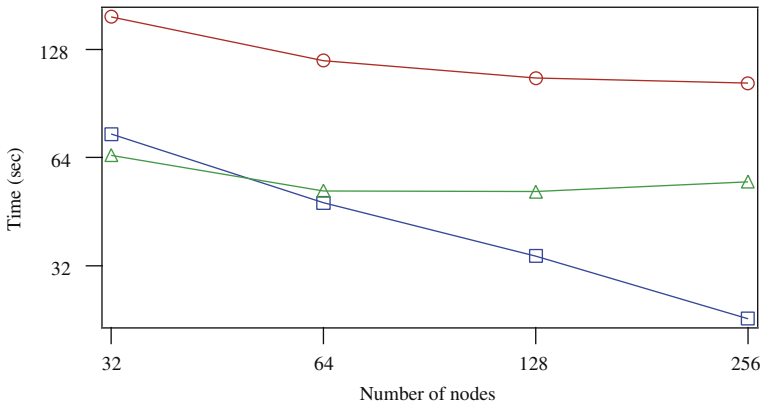


**Fig. 13.** Computational times for CG routine for the system of 1728 Si atoms. Circles mean the total computational time, and the squares and triangles mean the times for Hamiltonian operations and preconditionings, respectively.

densities [22], and the initial potential is computed from the initial electron density. Fig. 14 shows the convergence behavior of the RSDFT calculation for $Si_{4096}$. The convergence is judged by the norm of the potential difference,

$$\|V^{(i)} - V^{(i-1)}\|^2 < \eta, \tag{62}$$

where $V^{(i)}$ is the local potential of the $i$th iteration, and $\eta > 0$ is a convergence criterion. For the first 10 iterations, the local potential is fixed at the initial potential, and then the self-consistent iteration is performed by updating the potential. As

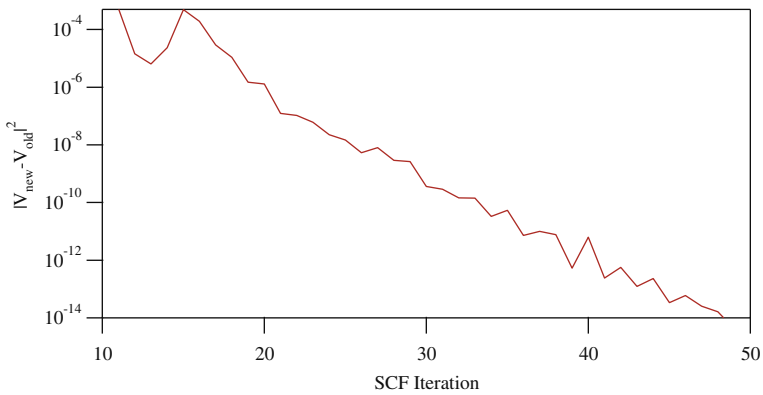


**Fig. 14.** Convergence behavior of the SCF calculation for the bulk crystalline Si of 4096 atoms.

shown in Fig. 14, 40–50 iterations are needed to achieve sufficient convergence. In Table 2, we show the computational time and performance of the main subroutines. Comparing the time for the CG and GS subroutines, we notice that the computational time for $O(N^2)$ and $O(N^3)$ routines are comparable, because the $O(N^3)$ computation can be performed under extremely high performance conditions due to level 3 BLAS. Since the computational time for the single SCF iteration is 776 s, and about 50 iterations are necessary for convergence, we obtain a self-consistent electronic-structure of the 4096 Si atom system in approximately 10 h.

Next, we present examples the RSDFT code applied to nanometer size Si clusters. The system sizes range from a few hundred atoms to 10,000 atoms. To our knowledge, this is the first time to perform the first-principles calculation for a system of over 10,000 Si atoms without any assumptions or limitations on the symmetry of the system. The first-principles real-space calculations for Si nanocrystals of over 800 atoms was achieved by Ogüt et al. in 1997 [46], and the real-space calculations for Si nanocrystals of thousands of Si atoms was achieved by Zhou et al. by introducing a novel Chebyshef-filtered subspace iteration method and employing the symmetry of the systems in 2006 [8].

Each cluster is terminated by H atoms at the surface, so that the systems have additional H atoms. Grid-spacing is chosen as 0.45 Å, which is the same as that used in the calculation of the 4096-atom bulk Si. We have confirmed that our results of the band gaps are changed less than 0.01 eV by reducing the grid-spacing to 0.38 Å. In Fig. 15, we show the density-of-states (DOS) for 5000- and 6000-atom Si clusters. The DOSs are not significantly different, and these are almost the same as that of the bulk DOS [47]. The similar result that the DOS of the Si cluster of thousands of atoms is very close to that of the bulk has also been found by Zhou et al. [8]. In Fig. 16, we show the dependence of the band gap energies on system size. In the infinite-size limit, the band gap should converge to the bulk band gap. Fig. 16 contains two curves: one is the band gap obtained from the difference between the highest-occupied and the lowest-unoccupied Kohn–Sham eigenvalues, and the other is the band gap obtained from the ΔSCF calculations [3]. Explicitly, the latter is obtained by the following formula

$$E_g^{\Delta\mathrm{SCF}} = E(N_e+1) + E(N_e-1) - 2E(N_e), \tag{63}$$

where $E(N_e+1)$, $E(N_e-1)$, and $E(N_e)$ represent the total energies of single negative, single positive, and neutral charge-state systems, respectively. As shown in Fig. 16, the band gap energies decrease as the system size increases, and for the largest cluster ($Si_{10701}H_{1996}$) the band gap energy is 0.66 eV from the Kohn–Sham eigenvalue difference and 1.07 eV from the ΔSCF calculation. The Kohn–Sham eigenvalue difference is close to the Kohn–Sham band gap of bulk Si, while the ΔSCF gap is close to the experimental band gap of 1.17 eV in Si crystals. However, the ΔSCF gap does not yet appear to converge. We have analyzed several quantities as a function of the cluster size and conclude that the ΔSCF band gap becomes the Kohn–Sham gap at the infinite-size limit. Details will be published elsewhere. In the precedent study by Zhou et al., for the largest cluster, they have obtained the band gap of 0.7 eV higher than that of their extrapolated value at the bulk limit 0.63 eV [8]. While in our calculations, the bulk limit is obtained as 0.53 eV, and the band gap of the largest cluster is 0.54 eV higher than that of the

**Table 2**
Computational time and performances for the bulk crystalline Si of 4096 atoms. For the SD routine, the time and performance of the SCALAPACK routine is also showed separately (in the parentheses).

| | Time (s) | MFLOPS/node |
|---|---|---|
| CG | 304.6 | 78.77 |
| GS | 138.1 | 3359 |
| SD (PDSYEVD) | 311.9 (83.2) | 2269 (103.2) |
| Total (1-SCF) | 776.9 | 1539 |

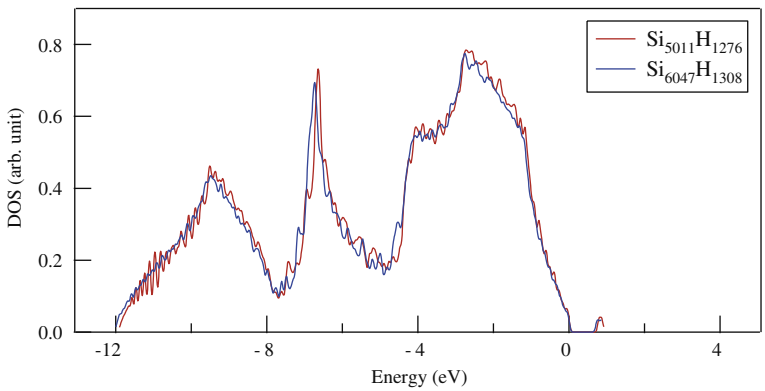


**Fig. 15.** Density of states for $Si_{5011}H_{1276}$ and $Si_{6047}H_{1308}$ clusters. The zero energy corresponds to the highest-occupied state.

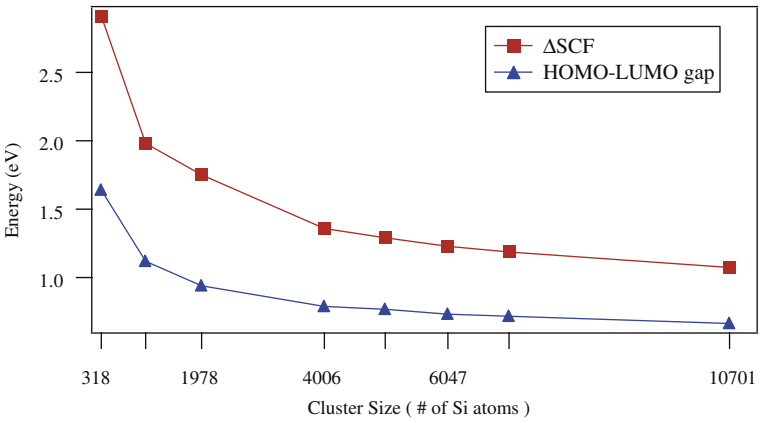


**Fig. 16.** Cluster-size dependence of the band gap energy obtained from the Kohn–Sham eigenvalues difference (triangles) and the delta-SCF method (squares).

bulk. By considering the slight differences of the bulk limit of the band gap and the size of the largest cluster, the present result is essentially the same as that previously reported by Zhou et al. [8].

The computational time for the largest 10,000-atom cluster is summarized in Table 3. The total number of grid points $M_L$ is 3,402,059, and the number of orbitals $M_B$ is 22,432. The computations are performed on 1024 nodes of PACS-CS (peak performance is 5.7 TFLOPS), and the memory requirement is 1.1 GB/node. Thus, we have demonstrated that a self-consistent electronic-structure calculation for a 10,000-atom Si system can be accomplished in about 5 days.

Finally, we present examples of the calculations for Si nanowire systems. Since the Si nanowires are promising candidates for a channel material of the field-effect transistor in the next generation [48], it becomes urgent to understand their electronic-structures in detail. Several models are examined here: Si nanowires of 4 and 8 nm diameters, and a Si nanowire of 10 nm diameter with surface roughness. The perfect nanowires of 4 and 8 nm diameter consist of 341 and 1361 Si atoms, and additional 84 and 164 terminating H atoms, respectively. The rough nanowire of 10 nm diameter consists of 12,822 Si atoms and 1544 terminating H atoms; the details of the construction of the roughness model are described below. The wire axis is chosen as [100], and periodic boundary condition is imposed along the wire axis. The period is just the lattice constant of bulk Si (5.43 Å) for the 4 and 8 nm diameter wires, and the 6 times of the lattice constant for the rough wire. Periodic boundary condition is also imposed in the lateral directions with 7 Å separation between the nanowires in adjacent supercells. For Brillouin zone integration, 4**k** points are sampled along the wire axis for the 4 and 8 nm diameter wires, and only the one ($\Gamma$) point is sampled for the rough wire.

The surface roughness is introduced as a random fluctuation of the radius of the nanowire around a average radius. In cylindrical coordinates, the radius of the rough wire is expressed as follows:

$$R(z,\theta) = R_0 + \Delta(z,\theta), \tag{64}$$

where $R_0$ is the average radius and $\Delta$ is the radius fluctuation. The distribution of the random fluctuation is characterized by assuming the following autocorrelation function [49,50],

$$\langle \Delta(\mathbf{r})\Delta(\mathbf{r}+\mathbf{r}')\rangle = \Delta_m^2 e^{-r'/L_m}, \tag{65}$$

where $\Delta_m$ is the root-mean-square of the radius fluctuation, and $L_m$ is the correlation length. In our calculations, we took 0.3, 0.54, and 5 nm for $\Delta_m$, $L_m$, and $R_0$, respectively.

In Fig. 17, we show the self-consistent electronic charge density and the atomic configuration of the Si nanowire with surface roughness. Obviously the system has no symmetry. Therefore, the calculation must be performed without any help of symmetry operation which is utilized to reduce the size of the matrix. The model system consists of 14,366 atoms, and the total number of grid points is 4,718,592. The total computational time to obtain the self-consistent electronic-structure was 292 h, including 93 iteration steps, with 1024 nodes of the PACS-CS.

**Table 3**
Computational times for 1-SCF step for $Si_{10701}H_{1996}$ cluster.

| | Time (s) |
|---|---|
| CG | 2680 |
| GS | 1184 |
| SD | 2350 |
| Total (1-SCF) | 6781 |

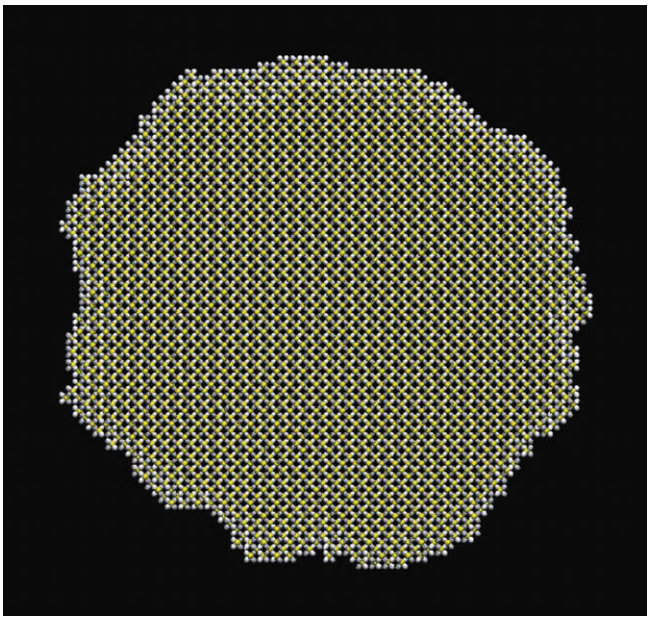

**Fig. 17.** Cross-sectional view of the self-consistent electronic charge density and the atomic configuration of the Si nanowire with surface roughness. The yellow balls represent Si atoms, and the silvers represent an isosurface of the electronic charge density.

In Fig. 18, we show the DOS of the rough and perfect Si nanowires, and also show the DOS of the bulk Si for comparison. Although the nanowires are typical one-dimensional systems, the DOS of the Si nanowire with large ($\geqslant$8 nm) diameter is almost same as that of the three-dimensional bulk Si. The band gaps of the nanowires are 0.81, 0.61, and 0.57 eV for the 4 nm diameter, 8 nm diameter, and the rough nanowire, respectively. The band gap approaches to the bulk value as the size of the nanowire increases. The value of the band gap of the largest nanowire, 0.57 eV, is very close to the bulk value, 0.53 eV.

In the DOS profile of the rough Si nanowire, we can find collapse of several peaks, which are not observed in the DOS of the bulk and the large (8 nm diameter) nanowire. For thinner (2 nm diameter) Si nanowires, it have been found that several sharp peaks related to the Van Hove singularities in the DOS profile disappeared by introducing surface roughness [50]. Our finding of the DOS variation of the rough nanowire may also be related to disappearance of some peaks contributing to the

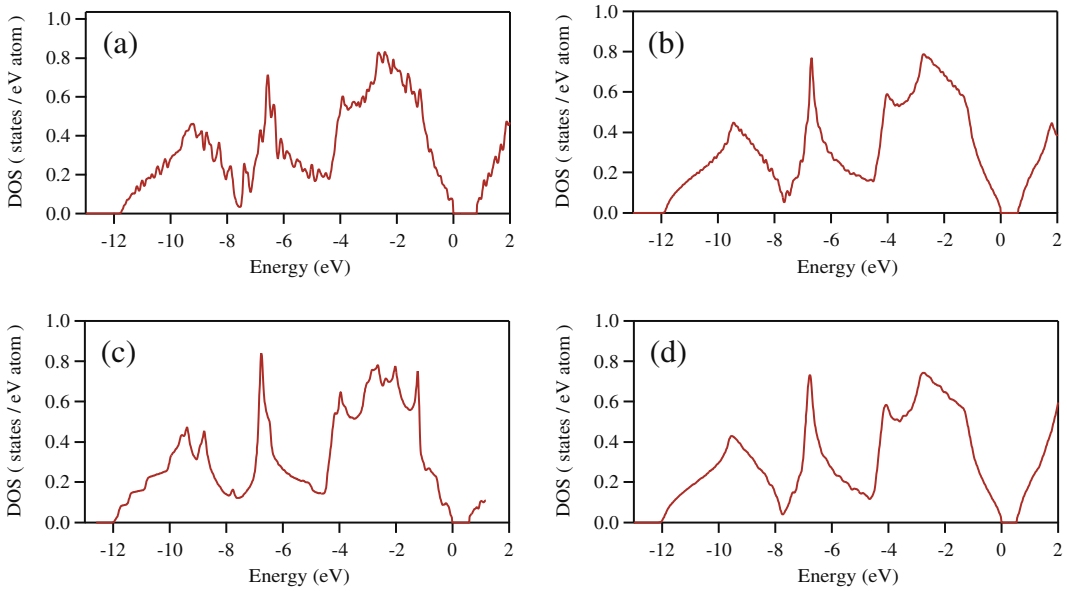


**Fig. 18.** Density of states for (a) Si nanowire of 4 nm diameter, (b) Si nanowire of 8 nm diameter, (c) Si nanowire with surface roughness, and (d) bulk Si. The energy 0 means the valence band top.

DOS. Although the band gap remains close to the bulk value, the influence of the surface roughness to the DOS seems to be large even in the large diameter nanowire.

## 8. Summary

We have developed RSDFT code suitable for massively-parallel computers, and have demonstrated that the code, which is based on the usual $O(N^3)$ formulation, can treat large systems. The code was developed to perform the computations using matrix products to the extent possible, which substantially improves the performance of $O(N^3)$ parts of the computations. We apply the code to 10,000 Si atom systems, and demonstrate that the self-consistent electronic-structure can be obtained within a few hundred hours using PACS-CS, which is a massively-parallel cluster with theoretical performance in the TFLOPS range. We have performed such large calculations without any assumption on the symmetry of the systems. Therefore, the present code is also applicable for amorphous systems and systems with structural roughness.

We are investigating several methods for further improvement of the RSDFT code. We have demonstrated that the real-space parallelization is highly efficient in our present code. In addition to that, parallelization on the orbital indices (orbital parallelization) is straightforward and very efficient for the CG routine. While in the GS routine, the orbital parallelization is not straightforward and unlikely to be efficient because of the complicated dependence among the orbitals and the additional communication. However, we have found the following fact from our theoretical estimate; as the number of processors increases, the total communication costs in the GS routine can be lowered by utilizing the orbital–parallel implementation in spite of the additional communications other than the MPI_ALLREDUCE. The implementation of the orbital parallelization has almost been completed, and the performance is now under investigation. Dealing with the multicore and the mutiprocessor architecture is also important ingredient to bring out the best performance of the recent, and the next generation, parallel computers. For the purpose, we are developing another version of the code, in which MPI is used for inter-node communication and OpenMP is used for intra-node communication, namely the hybrid-parallel version of the code.

We are also researching the best algorithm to solve the Kohn–Sham equation, or in other words the minimization of the energy functional, for further improvement of the electronic-structure calculations. It is desirable that the algorithm has no explicit orthogonalization in the minimization or the filtering process and has suitability for parallel computations. The recently-proposed Chebyshef-filtering method [8] satisfies such features and seems promising for large-scale calculations.

We should emphasize that the algorithmic improvements developed for this study is efficient not only for large-scale calculations, but also for reducing the time for medium-scale calculations; this is practically important for our usual work. In addition, the numerical techniques developed for this study are not limited to our code, but are also applicable to other methods such as the plane-wave method and the order-$N$ method. The synergy between computational sophistication and the development of the order-$N$ method are likely to make the first-principle calculations possible for systems with 100,000 or more atoms.

## Acknowledgments

We acknowledge the members of the COMAS-DFT meeting at University of Tsukuba, especially Prof. M. Sato, Prof. T. Sakurai, and Prof. A. Ukawa, for fruitful discussions and comments on the development of the RSDFT code. Numerical calculations for the present work have been carried out under the Interdisciplinary Computational Science Program at the Center for Computational Sciences, University of Tsukuba.

## Appendix A. Coefficients of the higher-order finite-difference approximation

Let us consider Taylor expansions of functions $f(x \pm mH)$ with respect to $mH$, which is the grid-spacing multiplied by an integer $m$:

$$\begin{aligned} f(x+mH) &= f(x) + f'(x)mH + f''(x)\frac{(mH)^2}{2!} + f'''(x)\frac{(mH)^3}{3!} + \cdots, \\ f(x-mH) &= f(x) - f'(x)mH + f''(x)\frac{(mH)^2}{2!} - f'''(x)\frac{(mH)^3}{3!} + \cdots. \end{aligned} \tag{66}$$

The coefficients of the higher-order finite-difference approximation for the first and second derivatives can be obtained by taking the appropriate linear combinations of $f(x \pm mH)$ so as to eliminate the higher-order terms in $O(H^n)$ ($n > 1$ for the first derivative and $n > 2$ for the second derivative, respectively).

The same coefficients can also be obtained through the Lagrange interpolation formula

$$f(x) \approx \sum_{m=-M_D}^{M_D} f(x_m) L_m^{2M_D}(x), \tag{67}$$

where $L_m^{2M_D}(x)$ is the $2M_D$-order polynomial

$$L_m^{2M_D}(x) = \frac{(x - x_{-M_D}) \cdots (x - x_{m-1})(x - x_{m+1}) \cdots (x - x_{M_D})}{(x_m - x_{-M_D}) \cdots (x_m - x_{m-1})(x_m - x_{m+1}) \cdots (x_m - x_{M_D})}. \tag{68}$$

The finite-difference calculation at the $i$th grid point is

$$\left.\frac{\partial f(x)}{\partial x}\right|_{x=x_i} \approx \sum_{m=-M_D}^{M_D} f(x_i + x_m) \left.\frac{\partial L_m^{2M_D}(x)}{\partial x}\right|_{x=x_i}, \tag{69}$$

$$\left.\frac{\partial^2 f(x)}{\partial x^2}\right|_{x=x_i} \approx \sum_{m=-M_D}^{M_D} f(x_i + x_m) \left.\frac{\partial^2 L_m^{2M_D}(x)}{\partial x^2}\right|_{x=x_i}, \tag{70}$$

where the coefficients are obtained from the derivatives of $L_m^{2M_D}(x)$.

By using the $2M_D + 1$ terms of $f(x \pm mH)$, we can eliminate the higher-order terms in Eq. (66) up to $O(H^{2M_D-1})$ and $O(H^{2M_D})$ for the first and second derivatives, respectively. Thus, the errors in the finite-difference approximation are $O(H^{2M_D-2})$, which is the same for both formulas.

## Appendix B. Oblique coordinate system and the finite-difference formula

In periodic systems, it is possible that the unit cell has a non-orthogonal shape. In this case, it is natural to take the coordinates along each lattice vector; i.e., the oblique-coordinate system instead of the Cartesian coordinate system. The coordinate transformation from $\vec{x}$-coordinates (Cartesian) to the $\vec{\xi}$-coordinates (oblique) is performed through a linear transformation,

$$\vec{\xi} = T\vec{x}. \tag{71}$$

The transformation matrix $T$ can be written as

$$T = \frac{1}{2\pi}(a_1\vec{b}_1, a_2\vec{b}_2, a_3\vec{b}_2), \tag{72}$$

where $\vec{b}_i$ is the reciprocal lattice vector and $a_i$ is the absolute length of the lattice vector; namely $a_i = |\vec{a}_i|$. The gradient operator and the Laplacian can be written in the $\vec{\xi}$-coordinate system as

$$\frac{\partial}{\partial x_i} = \sum_{j=1}^{3} \frac{b_j^{x_i}}{2\pi} \frac{\partial}{\partial \xi_j}, \tag{73}$$

$$\nabla^2 = \sum_{i=1}^{3} \frac{\partial^2}{\partial x_i^2} = \sum_{i=1}^{3} \sum_{j=1}^{3} \frac{(a_i\vec{b}_i) \cdot (a_j\vec{b}_j)}{4\pi^2} \frac{\partial^2}{\partial \xi_i \partial \xi_j}. \tag{74}$$

Contrary to the $\vec{x}$-coordinate system, the cross-terms appear in the expression for the Laplacian in the $\vec{\xi}$-coordinate system.

Finite-difference calculations are performed as

$$\frac{\partial f}{\partial \xi_1} \approx \sum_{m=-M_D}^{M_D} B_m f(\xi_1 + mH_1, \xi_2, \xi_3), \tag{75}$$

$$\frac{\partial^2 f}{\partial \xi_1^2} \approx \sum_{m=-M_D}^{M_D} C_m f(\xi_1 + mH_1, \xi_2, \xi_3). \tag{76}$$

The finite-difference formula for a cross-term can be written naively as

$$\frac{\partial^2}{\partial \xi_1 \partial \xi_2} \approx \sum_{m=-M_d}^{M_d} \sum_{n=-M_d}^{M_d} D_{m,n} f(\xi_1 + mH_1, \xi_2 + nH_2, \xi_3). \tag{77}$$

However, the formula is exactly the same as that obtained by applying twice the formula for the first-order derivative; i.e., the coefficient $D_{m,n}$ is written as $D_{m,n} = B_m \times B_n$.

## Appendix C. Structure optimization

In order to get a stable atomic structure using DFT, we perform the energy functional minimization with respect to the atomic coordinates as well as to the orbitals. The energy minimization with respect to the atomic coordinates can be performed using the CG method. Updating the position $\boldsymbol{R}_\alpha$ of atom $\alpha$ in the $i$th CG iteration is done using the Helmann–Feynman force $\boldsymbol{g}_\alpha$:

$$\boldsymbol{p}_\alpha^{(i)} = \boldsymbol{g}_\alpha^{(i)} + \gamma_i \boldsymbol{p}_\alpha^{(i-1)}, \tag{78}$$

$$\gamma_i = \frac{\boldsymbol{g}_\alpha^{(i)} \cdot \boldsymbol{g}_\alpha^{(i)}}{\boldsymbol{g}_\alpha^{(i-1)} \cdot \boldsymbol{g}_\alpha^{(i-1)}}, \quad \gamma_1 = 0, \tag{79}$$

$$\boldsymbol{R}_\alpha^{(i+1)} = \boldsymbol{R}_\alpha^{(i)} + \eta_i \boldsymbol{p}_\alpha^{(i)}, \tag{80}$$

$$\eta_i = \left\{ \eta | \min_\eta E\left[\boldsymbol{R}_\alpha^{(i)} + \eta \boldsymbol{p}_\alpha^{(i)}\right] \right\}. \tag{81}$$

The last step is the functional minimization along the $\mathbf{p}_i$ direction, which is the line-minimization. In the line minimization, we examine several points (atomic configurations) along the direction, and at each point we perform the self-consistent total-energy calculation.

## Appendix D. Preconditioned conjugate-gradient method

In order to minimize the RQ,

$$\varepsilon(\phi, \phi^*) = \frac{\langle \phi | H | \phi \rangle}{\langle \phi | \phi \rangle}, \tag{82}$$

we employ the preconditioned CG (PCG) method. We first compute the gradient vector

$$\vec{g} = -\frac{\partial \varepsilon}{\partial \vec{\phi}^*} = \frac{-1}{\langle \phi | \phi \rangle} (H\vec{\phi} - \varepsilon \vec{\phi}) \tag{83}$$

and then the following iterative calculation is performed for $i = 1$ to the maximum number of CG iterations $N_{CG}$:

$$\vec{p}_i = P\vec{g}_i + \gamma_i \vec{p}_{i-1}, \tag{84}$$

$$\gamma_i = \frac{\langle Pg_i | g_i \rangle}{\langle Pg_{i-1} | g_{i-1} \rangle}, \quad \gamma_1 = 0, \tag{85}$$

$$\vec{\phi}_{i+1} = \alpha_i \vec{\phi}_i + \beta_i \vec{p}_i. \tag{86}$$

In this expression, $P$ is the preconditioning matrix. The coefficients $\alpha_i$ and $\beta_i$ are chosen so as to minimize $\varepsilon(\phi_{i+1}, \phi_{i+1}^*)$, or, equivalently, to solve the following eigenvalue problem

$$\begin{pmatrix} \langle \phi_i | H | \phi_i \rangle & \langle \phi_i | H | p_i \rangle \\ \langle p_i | H | \phi_i \rangle & \langle p_i | H | p_i \rangle \end{pmatrix} \begin{pmatrix} \alpha_i \\ \beta_i \end{pmatrix} = \varepsilon \begin{pmatrix} \alpha_i \\ \beta_i \end{pmatrix}. \tag{87}$$

Typically, $N_{CG}$ is taken to be 2–4.

To see the effect of the preconditioning, let us consider the situation where we search the $n$th eigen-solution $\vec{\phi}$ of the Hamiltonian. The solution $\vec{\phi}$ can be expanded using the eigenfunctions of the Hamiltonian:

$$\vec{\phi} = \sum_a c_a \vec{\phi}_a = c_n \vec{\phi}_n + \sum_{a \neq n} c_a \vec{\phi}_a. \tag{88}$$

The gradient vector is expressed as

$$\vec{g} = \frac{-1}{\langle \phi | \phi \rangle} c_n (\varepsilon_n - \varepsilon) \vec{\phi}_n + \frac{-1}{\langle \phi | \phi \rangle} \sum_{a \neq n} c_a (\varepsilon_a - \varepsilon) \vec{\phi}_a. \tag{89}$$

If the preconditioning matrix is chosen as the inverse of the Hamiltonian $P \approx H^{-1}$, then the preconditioned gradient vector can be expressed as

$$\vec{\eta} = P\vec{g} = \frac{-1}{\langle \phi | \phi \rangle} c_n (\varepsilon_n - \varepsilon) H^{-1} \vec{\phi}_n + \frac{-1}{\langle \phi | \phi \rangle} \sum_{a \neq n} c_a (\varepsilon_a - \varepsilon) H^{-1} \vec{\phi}_a = \frac{-1}{\langle \phi | \phi \rangle} c_n \left(1 - \frac{\varepsilon}{\varepsilon_n}\right) \vec{\phi}_n + \frac{-1}{\langle \phi | \phi \rangle} \sum_{a \neq n} c_a \left(1 - \frac{\varepsilon}{\varepsilon_a}\right) \vec{\phi}_a. \tag{90}$$

When $\varepsilon \approx \varepsilon_n$, the first term of the right-hand side vanishes. When $\epsilon_a \gg \epsilon$, the summation of the second term becomes

$$-\frac{1}{\langle \phi | \phi \rangle} c_a \left(1 - \frac{\varepsilon}{\varepsilon_a}\right) \vec{\phi}_a \approx -\frac{1}{\langle \phi | \phi \rangle} c_a \vec{\phi}_a. \tag{91}$$

Thus, when we update the orbital according to

$$\vec{\phi}_{next} = \vec{\phi} + \alpha \vec{\eta}, \tag{92}$$

the components associated with the higher eigenvalues can be eliminated:

$$\vec{\phi}_{next} \approx d_n \vec{\phi}_n + \sum_{\varepsilon_a < \varepsilon} d_a \vec{\phi}_a. \tag{93}$$

Although this is not exactly the desired eigensolution $\vec{\phi}_n$, the surplus terms are also the components within the subspace that we are seeking. Therefore, we can sort the mixed orbitals $\vec{\phi}_{next}$ into the eigensolutions through the subsequent GS and SD procedures.

From the considerations discussed above, we learn that the preconditioning matrix should be constructed as the inverse of an operator that approximately reproduces the higher eigenvalues of the Hamiltonian. The kinetic energy operator is one such operator, and in the present code, we employ the kinetic energy operator approximated by $M_D = 1$ finite-difference as the preconditioning operator. The matrix inversion is also performed using the CG method, but the number of CG iterations is limited to three. Although three iterations can only give an approximate inverse, this works well as a preconditioning operator.

## References


[1] P. Hohenberg, W. Kohn, Inhomogeneous electron gas, Phys. Rev. 136 (1964) B864.
[2] W. Kohn, L.J. Sham, Self-consistent equations including exchange and correlation effects, Phys. Rev. 140 (1965) A1133.
[3] R.O. Jones, O. Gunnarsson, The density functional formalism, its applications and prospects, Rev. Mod. Phys. 61 (1989) 689.
[4] M.C. Payne, M.P. Teter, D.C. Allan, T.A. Arias, J.D. Joannopoulos, Iterative minimization techniques for ab initio total-energy calculations: molecular dynamics and conjugate gradients, Rev. Mod. Phys. 64 (4) (1992) 1045–1097.
[5] W. Kohn, Nobel lecture: electronic structure of matter – wave functions and density functionals, Rev. Mod. Phys. 71 (1999) 1253.
[6] J.M. Foster, F. Boys, Cannonical configurational interaction procedure, Rev. Mod. Phys. 32 (1960) 300.
[7] W.M.C. Foulkes, L. Mitas, R.J. Needs, G. Rajagopal, Quantum Monte Carlo simulations of solids, Rev. Mod. Phys. 73 (2001) 33.
[8] Y. Zhou, Y. Saad, M.L. Tiago, J.R. Chelikowsky, Parallel self-consistent calculations via Chebyshev-filtered subspace acceleration, Phys. Rev. E 74 (2006) 066704.
[9] T.-L. Chan, M.L. Tiago, E. Kaxiras, J.R. Chelikowsky, Size limits on doping phosphorus into silicon nanocrystals, Nano Lett. 8 (2008) 596–600.
[10] For a review, M.L. Lee, E.A. Fitzgerald, M.T. Bulsara, M.T. Currie, A. Lochtefeld, J. Appl. Phys. 97 (2005) 011101.
[11] N. Umezawa, K. Shiraishi, Y. Akasaka, A. Oshiyama, S. Inumiya, S. Miyazaki, K. Ohmori, T. Chikyow, T. Ohno, K. Yamabe, Y. Nara, K. Yamada, Chemical controllability of charge states of nitrogen-related defects in HfOxNy: first-principles calculations, Phys. Rev. B 77 (2008) 165130. and references there in.
[12] S. Raugei, F.L. Gervasio, P. Carloni, DFT modeling of biological systems, Phys. Status Solidi b 243 (2006) 2500–2515.
[13] K. Kamiya, M. Boero, M. Tateno, K. Shiraishi, A. Oshiyama, Possible mechanism of proton transfer through peptide groups in the H-pathway of the bovine cytochrome *c* oxidase, J. Am. Chem. Soc. 129 (2007) 9663–9673. and references there in.
[14] S. Goedecker, Linear scalin electronic structure methods, Rev. Mod. Phys. 71 (1999) 1085–1123.
[15] J.R. Chelikowsky, N. Troullier, K. Wu, Y. Saad, Higher-order finite-difference pseudopotential method – an application to diatomic-molecules, Phys. Rev. B 50 (1994) 11355–11364.
[16] K. Yabana, G.F. Bertsch, Time-dependent local-density approximation in real time, Phys. Rev. B 54 (1996) 4484–4487.
[17] E.L. Briggs, D.J. Sullivan, J. Bernholc, Real-space multigrid-based approach to large-scale electronic structure calculations, Phys. Rev. B 54 (1996) 14362–14375.
[18] T.L. Beck, Real-space mesh techniques in density functional theory, Rev. Mod. Phys. 72 (2000) 1041–1080.
[19] U.V. Waghmare, H. Kim, I.J. Park, N. Modine, P. Maragakis, E. Kaxiras, HARES: an efficient method for first-principles electronic structure calculations of complex systems, Comput. Phys. Commun. 137 (2001) 341–360.
[20] K. Hirose, T. Ono, Y. Fujimoto, S. Tsukamoto, First-Principles Calculation in Real-Space Formalism, Electronic Configurations and Transport Properties of Nanostructures, Imperial College Press, London, 2005.
[21] J.-I. Iwata, K. Shiraishi, A. Oshiyama, Large-scale density functional calculations on silicon divacancies, Phys. Rev. B 77 (2008) 115208.
[22] N. Troullier, J.L. Martins, Efficient pseudopotentials for plane-wave calculations, Phys. Rev. B 43 (1991) 1993.
[23] J. Yamauchi, M. Tsukada, S. Watanabe, O. Sugino, First-principles study on energetics of c-BN(0 0 1) reconstructed surfaces, Phys. Rev. B 54 (1996) 5586–5603.
[24] G. Kresse, J. Furthmüller, Efficient iterative schemes for ab initio total-energy calculations using a plane-wave basis set, Phys. Rev. B 54 (1996) 11169–11185.
[25] J.P. Perdew, A. Zunger, Self-interaction correction to density functional approximation for many-electron systems, Phys. Rev. B 23 (1981) 5048–5079.
[26] J.P. Perdew, K. Burke, Y. Ernzerhof, Generalized gradient approximation made simple, Phys. Rev. Lett. 77 (1996) 3865–3868.
[27] L. Kleinman, D.M. Bylander, Efficacious form for model pseudopotential, Phys. Rev. Lett. 48 (1982) 1425.
[28] R.P. Feynman, Forces in molecules, Phys. Rev. 56 (1939) 340–343.
[29] L.-W. Wang, Mask-function real-space implementations of nonlocal pseudopotentials, Phys. Rev. B 64 (2001) 201107.
[30] T. Ono, K. Hirose, Timesaving double-grid method for electronic-structure calculations, Phys. Rev. Lett. 82 (1999) 5016–5019.
[31] J. Han, M.L. Tiago, T.-L. Chan, J.R. Chelikowsky, Real space method for the electronic structure of one-dimensional periodic systems, J. Chem. Phys. 129 (2008) 144109.
[32] A. Natan, A. Benjamini, D. Naveh, L. Kronik, M.L. Tiago, S.P. Beckman, J.R. Chelikowsky, Real-space pseudopotential method for first principles calculations of general periodic and partially periodic systems, Phys. Rev. B 78 (2008) 075109.
[33] P. Pulay, Convergence acceleration of iterative sequences. The case of SCF iteration, Chem. Phys. Lett. 73 (1980) 393–398.
[34] D.D. Johnson, Modefied Broyden's method for accelerating convergence in self-consistent calculations, Phys. Rev. B 38 (1988) 12807–12813.
[35] F. Chaitin-Chatelin, Valeurs Propres de Matrices, Springer-Verlag, Tokyo, 2003 (Japanese translation).
[36] D.M. Bylander, L. Kleinman, S. Lee, Self-consistent calculations of the energy bands and bonding properties of $B_{12}C_3$, Phys. Rev. B 42 (1990) 1394.
[37] M.P. Teter, M.C. Payne, D.C. Allan, Solution of Schödinger's equation for large systems, Phys. Rev. B 40 (1989) 12255.
[38] C. Vömel, S.Z. Tomov, O.A. Marques, A. Canning, L.-W. Wang, J.J. Dongarra, State-of-the-art eigensolvers for electronic structure calculations of large scale nano-systems, J. Comput. Phys. 227 (2008) 7113–7124.
[39] D. Vanderstraeten, A stable and efficient parallel block Gram–Schmidt algorithm, in: Proceedings of the Fifth International Euro-Par Conference (Euro-Par 1999), Lecture Notes in Computer Science, vol. 1685, Springer-Verlag, 1999, pp. 1128–1135.
[40] F.J. Lingen, Efficient Gram–Schmidt orthonormalisation on parallel computers, Commun. Numer. Method Eng. 16 (2000) 57–66.
[41] J. Daniel, W.B. Gragg, L. Kaufman, G.W. Stewart, Reorthogonalization and stable algorithms for updating the Gram–Schmidt QR factorization, Math. Comput. 30 (1976) 772–795.
[42] T. Yokozawa, D. Takahashi, T. Boku, M. Sato, Efficient parallel implementation of classical Gram–Schmidt orthogonalization using matrix multiplication, in: Proceedings of the Fourth International Workshop on Parallel Matrix Algorithms and Applications (PMAA'06), 2006, pp. 37–38.
[43] J.J. Dongarra, J. Du Croz, S. Hammarling, R.J. Hanson, ACM Trans. Math. Softw. 16 (1990) 1.
[44] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. du Croz, A. Greenbaum, S. Hammarling, A. Mckenney, D. Sorense, Technical Report CS90-105, Computer Science Dept., University of Tennessee, 1990.
[45] The PACS-CS is constructed from 2560 nodes of LV Xeon 2.8 GHz processors, and the theoretical peak speed is 14.3 TFLOPS. <http://www.ccs.tsukuba.ac.jp/PACS-CS/>.

[46] S. Ogüt, J.R. Chelikowsky, S.G. Louie, Quantum confinement and optical gaps in Si nanocrystals, Phys. Rev. Lett. 79 (1997) 1770.
[47] M.T. Yin, M. Cohen, Theory of static structural properties, crystal stability, and phase transformations: application to Si and Ge, Phys. Rev. B 26 (1982) 5668.
[48] S.D. Suk, S.-Y. Lee, S.-M. Kim, E.-J. Yoon, M.-S. Kim, M. Li, C.W. Oh, K.H. Yeo, S.H. Kim, D.-S. Shin, K.-H. Lee, H.S. Park, J.N. Han, C.J. Park, J.-B. Park, D.-W. Kim, D. Park, B.-I. Ryu, High performance 5 nm radius Twin Silicon Nanowire MOSFET(TSNWFET): fabrication on bulk Si wafer, characteristics, and reliability, IEDM Tech. Dig., 2005.
[49] J. Wang, E. Polizzi, A. Ghosh, S. Datta, M. Lundstorm, Theoretical investigation of surface roughness scattering in sillicon nanowire transistors, Appl. Phys. Lett. 87 (2005) 043101.
[50] A. Lherbier, M.P. Persson, Y.-M. Niquet, F. Triozon, S. Roche, Quantum transport length in silicon-based semiconducting nanowires: surface roughness effects, Phys. Rev. B 77 (2008) 085301.