SIESTA-PEXSI: massively parallel method for efficient and accurate *ab initio* materials simulation without matrix diagonalization

IOP Publishing Journal of Physics: Condensed Matter
J. Phys.: Condens. Matter **26** (2014) 305503 (15pp) doi:10.1088/0953-8984/26/30/305503

SIESTA-PEXSI: massively parallel method for efficient and accurate *ab initio* materials simulation without matrix diagonalization

Lin Lin[1], **Alberto García**[2], **Georg Huhs**[3] **and Chao Yang**[1]

[1] Computational Research Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA
[2] Institut de Ciència de Materials de Barcelona, (ICMAB-CSIC), Campus de la UAB, E-08193 Bellaterra, Spain
[3] Computer Applications in Science & Engineering, Barcelona Supercomputing Center, Campus Nord UPC, c/Gran Capità, 2-4, 08034 Barcelona, Spain

E-mail: linlin@lbl.gov

Received 1 May 2014, revised 9 June 2014
Accepted for publication 11 June 2014
Published 10 July 2014

Abstract
We describe a scheme for efficient large-scale electronic-structure calculations based on the combination of the pole expansion and selected inversion (PEXSI) technique with the SIESTA method, which uses numerical atomic orbitals within the Kohn–Sham density functional theory (KSDFT) framework. The PEXSI technique can efficiently utilize the sparsity pattern of the Hamiltonian and overlap matrices generated in SIESTA, and for large systems it has a much lower computational complexity than that associated with the matrix diagonalization procedure. The PEXSI technique can be used to evaluate the electron density, free energy, atomic forces, density of states and local density of states without computing any eigenvalue or eigenvector of the Kohn–Sham Hamiltonian. It can achieve accuracy fully comparable to that obtained from a matrix diagonalization procedure for general systems, including metallic systems at low temperature. The PEXSI method is also highly scalable. With the recently developed massively parallel PEXSI technique, we can make efficient use of more than 10 000 processors on high performance machines. We demonstrate the performance and accuracy of the SIESTA-PEXSI method using several examples of large scale electronic structure calculations, including 1D, 2D and bulk problems with insulating, semi-metallic, and metallic character.

Keywords: Kohn–Sham density functional theory, fast algorithm, atomic orbitals, large scale simulation, parallel computation

(Some figures may appear in colour only in the online journal)

1. Introduction

Kohn–Sham density functional theory (KSDFT) is the most widely used framework for electronic-structure calculations, and plays an important role in the analysis of electronic, structural and optical properties of molecules, solids and nano-structures. The efficiency of KSDFT depends largely on the computational cost associated with the evaluation of the electron charge density for a given potential within a self-consistent field (SCF) iteration. The most straightforward way to perform such an evaluation is to partially diagonalize the Kohn–Sham Hamiltonian by computing a set of eigenvectors corresponding to the algebraically smallest eigenvalues of the Hamiltonian. The complexity of this approach is $\mathcal{O}(N_e^3)$, where N_e is the number of electrons in the atomistic system of interest. As the number of atoms or electrons in the system increases, the cost of diagonalization becomes prohibitively expensive.

Although linear scaling algorithms [1–8] are attractive alternatives for improving the efficiency of KSDFT, they rely on using the nearsightedness principle [9, 10], which asserts that the density perturbation induced by a local change in the external potential decays away from where the perturbation is applied. One can then truncate elements of the density matrix away from the diagonal. Such truncation can be in practice applied only to insulating systems whose density matrix elements decay exponentially away from the diagonal, but not to metallic systems at low temperature, for which the density matrix elements decay only algebraically away from the diagonal.

The recently developed pole expansion and selected inversion (PEXSI) method [11–15] provides an alternative way for solving the Kohn–Sham problem without using a diagonalization procedure, and without invoking the nearsightedness principle to truncate density matrix elements. Compared to existing techniques, the PEXSI method has a few salient features: (1) PEXSI expresses physical quantities such as electron density, free energy, atomic forces, density of states and local density of states in terms of a spectral projector which is evaluated without computing any eigenvalues or eigenvectors. (2) The computational cost of the PEXSI technique scales at most as $\mathcal{O}(N_e^2)$. The actual computational cost depends on the dimensionality of the system: the cost for quasi-1D systems such as nanotubes is $\mathcal{O}(N_e)$ i.e. linear scaling; for quasi-2D systems such as graphene and surfaces (slabs) is $\mathcal{O}(N_e^{1.5})$; for general 3D bulk systems is $\mathcal{O}(N_e^2)$. This favorable scaling hinges on the sparse character of the Hamiltonian and overlap matrices, but not on any fundamental assumption about the localization properties of the single particle density matrix. (3) The PEXSI technique can be accurately applied to general materials systems including small gapped systems and metallic systems, and remains accurate at low temperatures. (4) The PEXSI method has a two-level parallelism structure and is by design highly scalable. The recently developed massively parallel PEXSI technique can make efficient usage of 10 000 ∼ 100 000 processors on high performance machines. (5) PEXSI can be controlled with a few input parameters, and can act nearly as a black-box substitution of the diagonalization procedure commonly used in electronic structure calculations.

In order to benefit from the PEXSI method, the Hamiltonian and overlap matrices must be sparse. This requirement is satisfied if localized discretization is used for representing the Kohn–Sham Hamiltonian by a finite sized matrix. Examples of localized discretization include numerical atomic orbitals [16–20], Gaussian type orbitals [21, 22], the finite difference [23] and finite element [24] methods, adaptive curvilinear coordinates [25], optimized nonorthogonal orbitals [1–3] and adaptive local basis functions [26]. In contrast, the planewave basis set is not localized and therefore cannot directly benefit from the PEXSI method. Even though they are formally localized, the number of degrees of freedom per atom associated with methods such as the finite difference and the finite element is usually much larger than that associated with other methods such as numerical atomic orbitals, leading to an increase of the preconstant factor in the computational cost. Therefore the finite difference and the finite element methods may not benefit as much from the PEXSI technique as those based on numerical atomic orbitals. We note also that the use of hybrid functionals with an orbital-based exact exchange term [27, 28] may significantly impact the sparsity pattern of the Hamiltonian matrix and increase the computational cost.

In previous work [14], the applicability of the PEXSI method was demonstrated for accelerating atomic orbital based electronic structure calculations. With the sequential implementation of the PEXSI method, it was possible to perform electronic structure calculations accurately for a nanotube containing 10 000 atoms discretized by a single-ζ (SZ, minimal) basis, and to perform geometry optimization for a nanotube that contains more than 1000 atoms with a double-ζ plus polarization (DZP) basis. However, the sequential implementation does not benefit from the inherent parallelism in the PEXSI method, and therefore leads to limited or no improvement for general electronic structure calculations.

The contribution of this paper is twofold: (1) We present the SIESTA-PEXSI method, which combines the SIESTA method [29, 30] based on numerical atomic orbitals and the recently developed massively parallel PEXSI method [15]. The SIESTA-PEXSI method can be efficiently scalable to more than 10 000 processors. We provide performance data for a range of systems, including strong and weak scaling characteristics, and illustrate the crossover points beyond which the new approach is more efficient than diagonalization. The accuracy of the result obtained from the SIESTA-PEXSI method is nearly indistinguishable from the result obtained from the diagonalization method. (2) We develop a hybrid scheme of density of states estimation and Newton's method to obtain the chemical potential. We demonstrate that the scheme is highly efficient and robust with respect to the initial guess, with or without the presence of gap states. The SIESTA-PEXSI approach has been implemented as a new solver in SIESTA, with built-in heuristics that balance efficiency and accuracy, but at the same time offering full control by the user.

This paper is organized as follows. In section 2, we describe the massively parallel PEXSI technique, and how to integrate it with the SIESTA method. We also present a new method to update the chemical potential. In section 3, we report the performance of the SIESTA-PEXSI method on several problems.

Throughout the paper, we use $\mathrm{Im}(A)$ to denote the imaginary part of a complex matrix A. We use H, S to denote the discretized Hamiltonian matrix and the corresponding overlap matrix obtained from a basis set Φ such as numerical atomic orbitals. Similarly $\hat{\gamma}(x, x')$ denotes the single particle density matrix operator, and the corresponding electron density is denoted by $\rho(x)$. The matrix Γ denotes the single particle density matrix represented in the Φ basis. In PEXSI, Γ and related matrices are approximated by a finite P-term pole expansion, denoted by Γ_P, $\Gamma_P^{\mathcal{F}}$, Γ_P^E respectively. However, to simplify notation, we will drop the subscript P and simply use Γ, $\Gamma^{\mathcal{F}}$, Γ^E to denote the approximated matrices unless otherwise noted.

2. Theory and practical implementation

2.1. Basic formulation

The ground-state electron charge density $\rho(x)$ of an atomistic system can be obtained from the self-consistent solution to the Kohn–Sham equations

$$\widehat{H}[\rho(x)]\psi_i(x)=\psi_i(x)\varepsilon_i, \tag{1}$$

where $\widehat{H}$ is the Kohn–Sham Hamiltonian that depends on $\rho(x)$, $\{\psi_i(x)\}$ are the Kohn–Sham orbitals which in turn determine the charge density by

$$\rho(x)=\sum_i^{\infty}|\psi_i(x)|^2 f_i \tag{2}$$

with occupation numbers f_i that can be chosen according to the Fermi-Dirac distribution function

$$f_i=f_\beta(\varepsilon_i-\mu)=\frac{2}{1+\mathrm{e}^{\beta(\varepsilon_i-\mu)}}, \tag{3}$$

where μ is the chemical potential chosen to ensure that

$$\int\rho(x)\,\mathrm{d}x=N_e, \tag{4}$$

and β is the inverse of the temperature, i.e., $\beta=1/(k_\mathrm{B}T)$ with k_B being the Boltzmann constant.

The most straightforward method to solve the Kohn–Sham problem is to expand the orbitals ψ_i as a linear combination of a finite number of basis functions $\{\varphi_j\}$, and thus recast (1) as a (generalized) eigenvalue problem within an iterative procedure to achieve self-consistency in the charge density. The computational complexity of this approach is $\mathcal{O}(N^3)$, where N is the number of basis functions and is generally proportional to the number of electrons N_e or atoms in the system to be studied. This approach becomes prohibitively expensive when the size of the system increases.

Formally, the electronic-structure problem can be recast in terms of the one-particle density matrix defined by

$$\hat{\gamma}=\sum_{i=1}^{\infty}|\psi_i\rangle f_\beta(\varepsilon_i-\mu)\langle\psi_i|=f_\beta(\widehat{H}-\mu), \tag{5}$$

with μ chosen so that $\mathrm{Tr}[\hat{\gamma}]=N_e$. $\hat{\gamma}$ can thus be evaluated without the need for diagonalization, if the Fermi function is approximated by a linear combination of a number of simpler functions. This is the Fermi operator expansion (FOE) method [31].

While most of the FOE schemes require as many as $\mathcal{O}(\beta\Delta E)$ or $\mathcal{O}\left(\sqrt{\beta\Delta E}\right)$ terms of simple functions (with ΔE being the spectrum width), the recently developed pole expansion [11] is particularly promising since it requires only $\mathcal{O}(\log\beta\Delta E)$ terms of simple rational functions. The pole expansion has the analytic expression

$$f_\beta(\varepsilon-\mu)\approx\mathrm{Im}\sum_{l=1}^{P}\frac{\omega_l^\rho}{\varepsilon-(z_l+\mu)}, \tag{6}$$

We refer readers to [11, 14] for more details. The complex shifts $\{z_l\}$ and weights $\left\{\omega_l^\rho\right\}$ are determined only by β, ΔE and the number of poles P. All quantities in the pole expansion are known explicitly and their calculation takes negligible amount of time.

Following the derivation in [14], we can use (6) to approximate the single particle density matrix $\hat{\gamma}$ by its P-term pole expansion, denoted by $\hat{\gamma}_P$ as

$$\begin{aligned}\hat{\gamma}_P(x,x')&=\Phi(x)\,\mathrm{Im}\left(\sum_{l=1}^{P}\frac{\omega_l^\rho}{H-(z_l+\mu)S}\right)\Phi^T(x')\\&\equiv\Phi(x)\,\Gamma\Phi^T(x').\end{aligned} \tag{7}$$

where $\Phi=[\varphi_1,\cdots,\varphi_N]$ is a collective vector of basis functions, $S_{ij}=\langle\varphi_i|\varphi_j\rangle$, $H_{ij}=\langle\varphi_i|\widehat{H}|\varphi_j\rangle$, and Γ is an $N\times N$ matrix represented in terms of the Φ basis. To simplify our notation, we will drop the subscript P originating from the P-term pole expansion approximation unless otherwise noted.

It would seem that the need to carry out P matrix inversions in (8) would mean that the computational complexity of this approach is still close to the $\mathcal{O}(N^3)$ scaling of diagonalization. However, what is really needed in practice is just the electron density in real space, that is

$$\rho(x)=\Phi(x)\Gamma\Phi^T(x)=\sum_{ij}\Gamma_{ij}\,\varphi_j(x)\,\varphi_i(x). \tag{8}$$

When the basis functions $\varphi_i(x)$ are compactly supported in real space, the product of two functions $\varphi_i(x)$ and $\varphi_j(x)$ will be zero when they do not overlap. These i,j pairs can be excluded from the summation in equation (8). Consequently, we *only* need Γ_{ij} such that $\varphi_j(x)\varphi_i(x)\neq 0$ in equation (8). This set of Γ_{ij}'s is a subset of $\{\Gamma_{ij}|\,H_{ij}\neq 0\}$. To obtain these *selected elements*, we need to compute the corresponding elements of $(H-(z_l+\mu)S)^{-1}$ for all z_l. We emphasize that we compute the selected elements of the density matrix because only these elements are needed to compute physical quantities such as charge density, energy and forces, due to the localized character of the basis set. The computed selected elements of the density matrix are accurate, and should be regarded as if we performed a conventional $\mathcal{O}(N^3)$ calculation first, and then only kept the corresponding selected elements of the density matrix. In principle we could retrieve any matrix element of the density matrix, simply by enlarging the set of 'selected elements'. This process is fundamentally different from the usage of the 'near-sightedness' approximation, which throws away the information of the density matrix beyond the truncation region.

The recently developed selected inversion method [12, 13] provides an efficient way of computing the selected elements of an inverse matrix. For a (complex) symmetric matrix of the form $A=H-zS$, the selected inversion algorithm first constructs an LDL^T factorization of A, where L is a block lower triangular matrix called the Cholesky factor, and D is a block diagonal matrix. The computational scaling of the selected inversion algorithm is only proportional to the number of nonzero elements in the Cholesky factor L, which is $\mathcal{O}(N)$ for quasi-1D systems, $\mathcal{O}(N^{1.5})$ for quasi-2D systems, and $\mathcal{O}(N^2)$ for 3D bulk systems, thus achieving universal asymptotic improvement over the diagonalization method for systems of all dimensions. It should be noted that the selected inversion algorithm is an *exact* method for computing selected elements of A^{-1} if exact arithmetic is employed, and in practice the only source of error originates from the roundoff error. In particular, the selected inversion algorithm does not rely on any localization property of A^{-1}.

In addition to computing the charge density at a reduced computational complexity in each SCF iteration, we can also use this pole-expansion and selected-inversion (PEXSI) technique to compute the free energy $\mathcal{F}_{\text{tot}}$ and the atomic forces F_I efficiently without diagonalizing the Kohn–Sham Hamiltonian. It follows from the derivation in [14] that

$$\mathcal{F}_{\text{tot}} = \text{Tr}[\Gamma^{\mathcal{F}} S] + \mu N_e - \frac{1}{2}\iint \frac{\rho(x)\rho(y)}{|x-y|}\text{d}x\text{d}y + E_{\text{xc}}[\rho] - \int V_{\text{xc}}[\rho]\rho(x)\,\text{d}x, \tag{9}$$

$$F_I = -\frac{\partial \mathcal{F}_{\text{tot}}}{\partial R_I} = -\text{Tr}\left[\Gamma \frac{\partial H}{\partial R_I}\right] + \text{Tr}\left[\Gamma^E \frac{\partial S}{\partial R_I}\right]. \tag{10}$$

where the energy and free-energy density matrices Γ^E and $\Gamma^{\mathcal{F}}$ are given by pole expansions with *the same poles* as those used for computing the charge density:

$$\Gamma^{E,\mathcal{F}} = \text{Im}\sum_{l=1}^{P} \frac{\omega_l^{E,\mathcal{F}}}{H-(z_l+\mu)S}. \tag{11}$$

Since the trace terms in (9) and (10) require only the (i, j)th entries of $\Gamma^{E,\mathcal{F}}$ for (i, j) satisfying $S_{ij} \neq 0$ or $H_{ij} \neq 0$, the needed elements of the energy-density matrices can be computed without additional complexity.

2.2. Massively parallel PEXSI method

In addition to its favorable asymptotic complexity, the PEXSI method is also inherently more scalable than the standard approach based on matrix diagonalization when it is implemented on a parallel computer. The parallelism in PEXSI exists at two levels. First, the selected inversions associated with different poles (usually on the order of $40 \sim 60$) are completely independent. Second, each selected inversion itself can be parallelized by using the parallel selected inversion method called PSelInv [15].

A parallel selected inversion consists of the following steps:

(a) The rows and columns of the matrices H and S are reordered to reduce the number of nonzeros in the triangular factor of the LDL^T decomposition of $H - z\,S$.
(b) A parallel symbolic factorization of $H - z\,S$ is performed to identify the location of the nonzero matrix elements in L.
(c) The numerical LDL^T decomposition (or equivalent LU decomposition) of $H - zS$ is performed.
(d) The desired selected elements of $(H - z\,S)^{-1}$ are computed from L and D.

Step 1 can be performed in parallel by using the ParMETIS [32] or the PT-Scotch [33] software packages. Its cost is much smaller compared to the numerical factorization, and only needs to be done once. Although for symmetric matrices only LDL^T factorization is needed, the PEXSI package currently use the SuperLU_DIST software package [34] to perform steps 2 and 3 in parallel with LU factorization. The LU factorization contains equivalent information as in LDL^T factorization but can be twice as expensive in the worst case due to the lack of usage of the symmetry property of the matrix.

PEXSI and PSelInv has an independent data structure, and can be interfaced with other sparse direct solvers.

The cost of symbolic factorization in Step 2 is usually much lower than the numerical factorization. The numerical factorization procedure can be described in terms of the traversal of a tree called the *elimination tree*. Each node of the tree corresponds to a block of continuous columns of $H - zS$. A node $\mathcal{R}$ is the parent of a node $\mathcal{J}$ if and only if

$$\mathcal{R} = \min\left\{\mathcal{I} > \mathcal{J} \mid L_{\mathcal{I},\mathcal{J}} \text{ is a nonzero block}\right\}. \tag{12}$$

In SuperLU_DIST, each node is distributed among a subset of processors block cyclically. The traversal of the elimination tree proceeds from the leaves towards the root. The update of $\mathcal{I}$ is performed in parallel on a subset of processors assigned to $\mathcal{I}$ and its children nodes, and the main operations involved in the update are a number of dense matrix-matrix multiplications. In addition to parallelism within the update of a supernode, additional concurrency can be exploited in the traversal of different branches of the elimination tree for updating different supernodes.

All of these techniques can be used in Step 4 to compute selected elements of $(H - zS)^{-1}$. In this step, the elimination tree is traversed from the root down towards the leaves. As each node is traversed, selected elements of $(H - zS)^{-1}$ within the columns that are mapped to that node are computed through a number of dense matrix-matrix multiplications. Communication is needed among processors that are mapped to the node and its ancestors. Multiple nodes belonging to different branches of the elimination tree can be traversed simultaneously if the update of these nodes do not involve communications with the same ancestor.

For sufficiently large problems, SuperLU_DIST can achieve substantial speedup in the numerical factorization when hundreds to thousands of processors are used. Similar or better speedup factors can be observed when selected elements of $(H - zS)^{-1}$ are computed from the distributed L and D factors. It has been shown that for large matrices a single selected inversion can scale to 4096 cores, and if 40 poles are used in the pole expansion of the Fermi-Dirac function, the total number of computational cores that can be efficiently utilized by the parallel PEXSI method is $40 \times 4096 \approx 160\,000$. We will give some more concrete examples in section 3 to demonstrate the performance of our implementation of the parallel PEXSI algorithm.

Compared to PEXSI, it is generally difficult to efficiently use that many computational cores in a dense matrix calculation algorithm such as those implemented in ScaLAPACK. The main bottleneck of the computation using ScaLAPACK is the reduction of a dense matrix to a tridiagonal matrix and the back transformation of the eigenvectors of a tridiagonal matrix, which are inherently sequential. Moreover, the cost of the diagonalization method scales cubically with respect to the matrix dimension. This limits the size of the matrix that can be handled by ScaLAPACK, as well as the parallel scalability on massively parallel computers. Although some recent progress has been made to make these transformations more efficient [35], making diagonalization scalable on more than tens of thousands of cores remains a challenging task.

2.3. Determination of the chemical potential

The chemical potential μ required in the pole expansion (8) is not known a priori. Its value must in general be determined so that the total number of electrons is appropriate:

$$N_e = N_\beta(\mu) = \mathrm{Tr}[\hat{\gamma}] = \mathrm{Tr}[\Gamma \Phi^T \Phi] = \mathrm{Tr}[\Gamma S]. \qquad (13)$$

As the right-hand-side is a non-decreasing function of μ, the chemical potential can be efficiently obtained by Newton's method, maybe combined with extra safeguards such as bisection. The required derivative $N'_\beta(\mu)$ can be computed with very little extra cost using the pole expansion of the derivative of the Fermi-Dirac distribution $f'_\beta(\varepsilon - \mu)$, which can be constructed, for the reasons illustrated in [14], by using the same shifts z_l as those in (6):

$$N'_\beta(\mu) = \mathrm{Tr}[\Gamma^d S], \qquad (14)$$

$$\Gamma^d = \mathrm{Im}\sum_{l=1}^{P} \frac{\omega_l^d}{H - (z_l + \mu)S}. \qquad (15)$$

When Newton's method is used, the convergence of μ is rapid near the correct chemical potential. However, the standard Newton's method may not be robust enough when the initial guess is far away from the correct chemical potential. It may give, for example, too large a correction when $N'_\beta(\mu)$ is close to zero, as when μ is near the edge or in the middle of a band gap.

One way to overcome the above difficulty is to use an approximation to the function $N_\beta(\varepsilon)$ to narrow down the region in which the correct μ must lie. This function can be seen effectively as a (temperature smeared) cumulative density of states, counting the number of eigenvalues in the interval $(-\infty, \varepsilon)$.

We can evaluate $N_\infty(\varepsilon)$, its zero-temperature limit, without computing any eigenvalues of (H, S). Instead, we perform a matrix decomposition of the shifted matrix $H - \varepsilon S = LDL^T$, where L is unit lower triangular and D is diagonal. It follows from Sylvester's law of inertia [36], which states that the inertia (the number of negative, zero and positive eigenvalues) of a real symmetric matrix does not change under a congruent transform, that D has the same inertia as that of $H - \varepsilon S$. Hence, we can obtain $N_\infty(\varepsilon)$ by simply counting the number of negative entries in D. Note that the matrix decomposition $H - \varepsilon S = LDL^T$ can be computed efficiently by using a sparse LDL^T or LU factorization in real arithmetic. It requires fewer floating point operations than the complex arithmetic direct sparse factorization used in PEXSI.

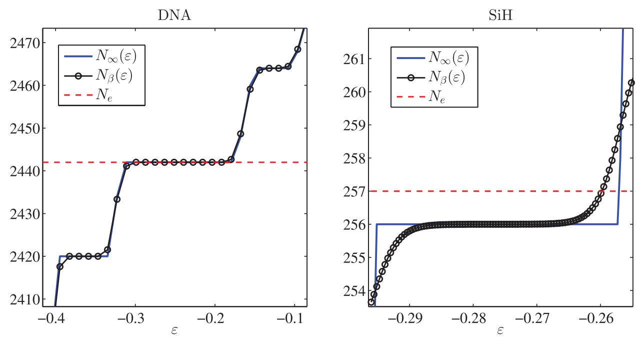


Figure 1. Number of electrons at zero temperature $N_\infty(\varepsilon)$ (blue solid line), at 300 K $N_\beta(\varepsilon)$ (black line with circles), and exact number of electrons N_e (red dashed line) for (*a*) DNA-1 (*b*) SiH. (unit of ε: Hartree).

To estimate $N_\beta(\mu)$ for a finite β, we use the identity

$$N_\beta(\mu) = \int_{-\infty}^{\infty} f_\beta(\varepsilon - \mu)\, \mathrm{d}N_\infty(\varepsilon), \qquad (16)$$

and perform an integration by parts to obtain

$$N_\beta(\mu) = -\int_{-\infty}^{\infty} f'_\beta(\varepsilon - \mu) N_\infty(\varepsilon)\, \mathrm{d}\varepsilon. \qquad (17)$$

The integral in (17) can be evaluated numerically by sampling $f'_\beta(\varepsilon - \mu)$ and $N_\infty(\varepsilon)$ at a number of quadrature points $\{\varepsilon_m\}_{m=1}^{Q}$ and performing a weighted sum of $f'_\beta(\varepsilon_m - \mu)N_\infty(\varepsilon_m)$ for $m = 1, \ldots, Q$. The Q evaluations of $\{N_\infty(\varepsilon_m)\}_{m=1}^{Q}$ can be performed simultaneously using the LDL^T factorization-based inertia counting procedure described above, with Q groups of processors. Since the derivative of the Fermi-Dirac function is sharply peaked, we can approximate $N_\beta(\mu)$ in a given interval by sampling $N_\infty(\varepsilon)$ in a slightly wider interval. Figure 1 shows the number of electrons at zero temperature $N_\infty(\varepsilon)$ obtained from inertia counting procedure, and the interpolated finite temperature profile $N_\beta(\varepsilon)$ at 300 K for a DNA system (with finite gap) and a SiH system (with zero gap) near the Fermi energy.

While the finite temperature smearing effect is negligible for insulators at 300 K, it is more pronounced for metals and leads to a more smooth $N_\beta(\varepsilon)$ which is suitable for applying Newton's method to find the chemical potential. On the other hand, the inertia counting procedure obtains the global profile of the cumulative density of states, and does not suffer from the problem of being trapped in intermediate band gaps.

This approximation to the function $N_\beta(\varepsilon)$ is then available for use in determining the approximate placement of the true μ by simple root finding: $N_\beta(\mu) = N_e$. In practice, we need to start from an interval which is large enough to contain the chemical potential. For systems with a gap, the interval should also contain the gap edges. In this case, two roots are sought: $N_\beta(\varepsilon_l) = N_e - \delta N$ and $N_\beta(\varepsilon_h) = N_e + \delta N$, where δN is small, say 0.1, and so ε_l and ε_h will be estimates of the band edges from which μ can be determined as $\mu = 1/2(\varepsilon_l + \varepsilon_h)$.

The fidelity of $N_\beta(\varepsilon)$, and thus the quality of the estimation of μ, depends on the density with which the interval can be sampled, which, with a fixed number Q of sampling points for $N_\infty(\varepsilon)$, will increase as the interval is narrowed. In practice, the above procedure is repeated with progressively smaller intervals until either the μ estimate stabilizes or the size of the interval is small enough. Unless the original interval is very large, 2 or 3 inertia counts are enough to provide an adequate μ from which to start the PEXSI process, and the number of inertia counts can be one or even zero (i.e. the inertia counting procedure is turned off) in subsequent SCF iterations. Newton's method in the solver then takes over for the final refining of μ.

The hybrid procedure of inertia counting and Newton's method is efficient and robust for both insulating and metallic systems. This will be further demonstrated in section 3.

2.4. Calculation of density of states and localized density of states

The density of states $g(\varepsilon)$ is defined as

$$g(\varepsilon) = 2\sum_i \delta(\varepsilon - \varepsilon_i). \tag{18}$$

The factor of two comes from spin degeneracy. As advanced in the previous section, the cumulative density of states (CDOS)

$$C(\varepsilon) = \int_{-\infty}^{\varepsilon} g(\varepsilon')\,\mathrm{d}\varepsilon'. \tag{19}$$

is exactly the function $N_\infty(\varepsilon)$ discussed there. To evaluate $g(\varepsilon)$, we sample $N_\infty(\varepsilon)$ at a set of $\{\varepsilon_l\}$ using the inertia counting procedure, and use an appropriate interpolation scheme to approximate $N_\infty(\varepsilon)$ for other values of ε. The approximation to the DOS is then obtained by numerical differentiation. We remark that inertia counting is not the only diagonalization-free method for computing the DOS. Other methods that make use of matrix vector multiplications only are also possible (see e.g. a recent review [37]).

Another physical quantity that can be easily approximated via selected inversion is the localized density of states (LDOS), defined as

$$g(x, \varepsilon) = 2\sum_i \delta(\varepsilon - \varepsilon_i)\,|\psi_i(x)|^2. \tag{20}$$

and representing in fact the contribution to the charge density of the states with eigenvalues in the vicinity of ε, as filtered by the δ function.

When a finite basis set Φ is used, the LDOS can be represented as

$$g(x, \varepsilon) = 2\Phi(x)\,C\delta(\varepsilon - \Xi)\,C^T\Phi^T(x). \tag{21}$$

where C is a matrix of the coefficients of the expansion in Φ of the orbitals $\psi_i(x)$, and Ξ is a diagonal matrix of the eigenvalues. It follows from the Sokhotski–Plemelj formula [38]

$$\lim_{\eta\to 0+} \frac{1}{\varepsilon + \mathrm{i}\eta} = \mathrm{PV}(1/\varepsilon) - \mathrm{i}\pi\delta(\varepsilon), \tag{22}$$

that

$$\delta(\varepsilon - \varepsilon_i) = \lim_{\eta\to 0+} -\frac{1}{\pi}\mathrm{Im}\,\frac{1}{\varepsilon + \mathrm{i}\eta - \varepsilon_i}, \tag{23}$$

where symbol PV in (22) stands for the Cauchy principal value.

Combining equations (21) and (23), we obtain the following alternative expression for the LDOS:

$$\begin{aligned} g(x, \varepsilon) &= -\frac{2}{\pi}\lim_{\eta\to 0+} \Phi(x)\,C(\varepsilon + \mathrm{i}\eta - \Xi)^{-1}\,C^T\Phi^T(x) \\ &= \frac{2}{\pi}\lim_{\eta\to 0+} \Phi(x)\,[H - (\varepsilon + \mathrm{i}\eta)S]^{-1}\,\Phi^T(x). \end{aligned} \tag{24}$$

Note that equation (24) allows us to compute the LDOS without using any eigenvalue or eigenvector. In practice, we take η to be a small positive number, and the local DOS can be approximated by

$$g(x, \varepsilon) \approx \frac{2}{\pi}\sum_{ij} \varphi_i(x)\,\varphi_j(x)\,[H - (\varepsilon + \mathrm{i}\eta)S]_{ij}^{-1}, \tag{25}$$

which is similar to the computation of the electron density. Again only the selected elements of the matrix $[H - (\varepsilon + \mathrm{i}\eta)\,S]^{-1}$ are needed for the computation of the LDOS for each ε, and the selected elements can be obtained efficiently by the PSelInv procedure.

2.5. Interface to SIESTA and heuristics for enhancing the efficiency

SIESTA is a density functional theory code which uses finite-range atomic orbitals to discretize the Kohn–Sham problem [29], and thus handles internally sparse H, S, and single-particle density matrices. SIESTA is therefore well suited to implement the PEXSI method along the lines explained above.

The PEXSI method can be directly integrated into SIESTA as a new kind of electronic-structure solver. Conceptually, the interface between SIESTA and PEXSI is straightforward. The existing SIESTA framework takes care of setting up the basis set and of constructing the sparse H and S matrices at each iteration of the self-consistent-field (SCF) cycle. H and S are passed to the PEXSI module, which returns the density matrix Γ and, optionally, the energy-density matrix Γ^E (needed for the calculation of forces) and the $\Gamma^{\mathcal{F}}$ matrix that can be used to estimate the electronic entropy. SIESTA then computes the charge density to generate a new Hamiltonian to continue the cycle, until convergence is achieved. Energies and forces are computed as needed.

The details of the interface are controlled by a number of parameters which provide flexibility to the user, especially in regard to the bracketing and tolerances involved in the determination of the chemical potential, and in the context of parallel computation. We list some of the more relevant parameters in table 1, but we should note that even finer control is possible by other, more specialized parameters.

The initial interval $[\mu_{lb}, \mu_{ub}]$ should be large enough to contain the true chemical potential. If it does not, the code will automatically expand it until μ is properly bracketed, but it is obviously more efficient to start the process with an appropriate interval, even if it is relatively large, and use the program's refining features without the need for backtracking. Note that, due to the implicit reference energy used by SIESTA, μ is typically negative and within a Rydberg of zero, so the specification of $[\mu_{lb}, \mu_{ub}]$ should not be a problem

Table 1. Main SIESTA-PEXSI parameters

Parameter	Purpose
$[\mu_{lb}, \mu_{ub}]$	Initial guess of the lower and upper bounds of the chemical potential μ
n_{IC}	The number of SCF steps in which inertial counting is used to narrow down the interval in which μ lies.
tol_{NE}	Tolerance on number of electrons
T	Electronic temperature
P	Number of poles
ppp	Number of processors used per pole
np_0	The number of processors used for non-PEXSI operations

in practice even when nothing is known about the electronic structure of the system. In fact the program can choose an appropriately wide starting interval if the user does not indicate one.

The bracketing interval can be refined by the use of the inertia counting procedure detailed in section 2.3. This is particularly important in the first few SCF steps in which the approximate electron density (and consequently the Hamiltonian) is far from converged. The parameter n_{IC} controls directly the number of SCF steps for which this procedure is followed. Beyond those, the PEXSI solver is directly invoked without further refinement. There is also the possibility of linking the use of inertia counting refinement to the convergence level of the calculation.

Because the chemical potential tends to oscillate in the first few SCF steps, a completely robust method should, in principle, search for a bracket afresh at every step, starting from a sufficiently wide interval. However, it is wasteful to completely ignore the previous bracketing when the SCF cycle reaches a more stable region, so we developed a heuristic to allow SIESTA-PEXSI to reuse the bracket determined in the previous SCF step, under various conditions related to the convergence level and to whether or not inertia counting is still used as a safeguard. We set the μ search interval for a new SCF step to be slightly larger (by a value that can also be controlled) than the final μ interval used in the previous SCF step. For some systems, an estimation of the change in the band-structure energy caused by the change in H can be used to shift the bracket across iterations. If μ ever falls out of the bracketing interval, the algorithm recovers automatically by expanding the interval appropriately.

When the search interval is deemed appropriate, we invoke the PEXSI solver with a starting μ equal to the mid-point of the interval, and use the solver's built-in Newton's method to refine μ until the error in the total number of electrons is below tol_{NE}.

The tolerance in the computed number of electrons tol_{NE} is a key parameter of the PEXSI module, and it should be set to an appropriately small value to guarantee the necessary accuracy in the results. But there is no advantage in setting the tolerance too low in the early stages of the SCF cycle. Hence, the favored mode of operation of SIESTA-PEXSI is to use an on-the-fly tolerance level which ranges from a coarse value at the start and progressively (in tandem with the reduction of a typical convergence-level metric, which can also control the coarseness of the bracketing) decreases towards the desired fine level tol_{NE}. Typically, no more than 3–5 PEXSI solver iterations are needed to achieve a high accuracy in N_e, and in practice the use of an adaptive tolerance level with proper bracketing means that just one or at most two solver iterations are enough for most of the steps in a complete SCF cycle. For gapped systems, with the starting μ well into the gap, the solver iteration can be turned off without affecting the accuracy of the results, leading to significant savings.

The total cost of a SCF cycle includes the LDL^T factorizations required in the inertia counting bracket refinements, in addition to the cost of the PEXSI solver. As the cost of an inertia counting invocation is typically much lower than a solver iteration, it is clear that a strategy that uses at least a few inertia counting steps to properly bracket μ before invoking the solver will be in fact cheaper than one which incurs the extra costs associated with a bad guess of μ. The overall cost of the algorithm would depend on the uniformity of the convergence to self-consistency and on the electronic structure of the system (i.e, whether or not it has a gap). We will report the actual cost of μ search for a variety of systems in the next section.

The design of SIESTA-PEXSI provides some flexibility to users in terms of the usage of computational resources. If the user's goal is a low time-to-solution on a large machine, the number of processors per pole ppp should be increased as much as possible. The overall number of processors Np should be set to $P \times ppp$, where P is the number of poles, to achieve complete parallelization over poles. If the goal is to minimize the number of processors involved in the computation, then ppp should be set to the minimum number that allows the problem to fit in memory. It should be noted that the memory requirements of the PEXSI approach are significantly lower than those of the diagonalization-based algorithm, as the relevant matrices are handled in their original sparse form, and not converted to dense form as in ScaLAPACK. As the parallel efficiency over the number of poles is nearly perfect, the total cost (time $\times$ Np) does not depend on the total number of processors used, so the minimum-cost strategy can be used with a useful range of machine sizes: from total parallelization over poles in medium-to-large machines, to serial calculation of poles in small machines.

The number of processors that can be profitably used by the PEXSI module is typically larger than the number of processors that the non-solver part of SIESTA needs (as it is itself very efficient, having been coded for essentially O(N) operation). Hence the non-solver operations in the SIESTA side use a subset of the processors available, specified by the np_0 parameter in table 1. Appropriate logic is in place to orchestrate the data movement and control flow.

3. Numerical results

In this section we report the performance and accuracy achieved by the massively parallel SIESTA-PEXSI method for computing the ground state energy and atomic forces of several systems. We also show some other capabilities of

Table 2. Test problems used in numerical experiments.

Name	Type	Description
DNA	1D insulating	DNA. The basic unit (715 atoms) contains two base pairs of an A-DNA double helix. Replicating this unit along the axis of rotation results in several instances of this problem of different sizes, with quasi-one-dimensional character. The largest instance considered contains 25 units and is 76 nanometers long.
C-BN	2D semi-metallic	A layer of boron nitride (BN) on top of a graphene sheet. Several instances of this problem are generated by varying the orientation of the BN sheet relative to the graphene layer within an appropriate periodic cell. The largest example considered of this quasi-two-dimensional system has 12 700 atoms.
H_2O	3D insulating	Liquid water. The basic repeating unit contains 64 molecules, and is generated by taking a snapshot of a molecular-dynamics run with the TIP4P force field. Appropriate supercells can be generated, with the largest having 8000 molecules, or 24 000 atoms.
Al	3D metallic	A bulk Al system generated from a 8 × 8 × 8 supercell of the primitive *FCC* unit cell. The positions of the atoms are perturbed by small random displacements to break the symmetry. This is a typical metallic system.
SiH	3D, special	A bulk Si system with 64 atoms, with an H interstitial impurity. The Fermi level in this system is pinned by the position of an H-derived level within the gap.

Table 3. Characteristics of the test examples in terms of the number of atoms, physical dimension of each system, the corresponding size of the Hamiltonian H, and the percentage of nonzero elements in H and in the L and U factors. Also given is the length scale l, which is, in the case of DNA, the length of the cell in the direction of rotation axis of the strand, for C-BN the length of a side of the unit cell, and for H_2O the side length of the cube.

Example	Atoms	N	s_H (%)	s_{LU} (%)	l (nm)
DNA-1	715	7183	6.8	23	3.1
DNA-4	2860	28732	1.7	6.2	12
DNA-9	6435	64647	0.75	2.9	28
DNA-16	11440	114928	0.42	1.7	49
DNA-25	17875	179575	0.27	1.1	76
C-$BN_{2.3}$	1988	25844	5.9	36	5.5
C-$BN_{1.43}$	3874	50362	3.0	24	7.7
C-$BN_{0.57}$	7988	103844	1.5	15	11
C-$BN_{0.00}$	12770	166010	0.91	11	14
H_2O-8	1536	11776	2.3	28	2.5
H_2O-27	5184	39744	0.69	18	3.7
H_2O-64	12288	94208	0.29	12	5.0
H_2O-125	24000	184000	0.15	8.4	6.2
Al	512	6656	36	94	1.6
SiH	65	833	74	97	1.1

SIESTA-PEXSI that may be useful for characterizing the electronic properties of materials.

To demonstrate that SIESTA-PEXSI can handle different types of systems, we choose five different test problems for our numerical experiments, including insulating, semi-metallic, and metallic systems, and covering all the relevant dimensions. Table 2 gives a brief description of each system. The atomic configurations of DNA-4, C-$BN_{2.3}$ and H_2O-8 are given in figure 2.

Our calculations were performed on the Edison system at the National Energy Research Scientific Computing (NERSC) center. Each node consists of two twelve-core Intel 'Ivy Bridge' 2.4 GHz processors and has 64 gigabytes (GB) of DDR3 1866 MHz memory.

The number of atoms in the various instances of the problems listed on table 2, the sizes of unit cells, as well as the matrix sizes and the sparsity (i.e., the percentage of nonzero elements) of the corresponding H matrices and their L and U factors are given in table 3. Note that the number of nonzero elements in L+U is the same as the number of nonzero elements in L+D+L^T if an equivalent LDL^T factorization is to be used.

Table 4. Deviations of the energy and forces obtained from SIESTA-PEXSI with respect to those from a diagonalization-based calculation.

System	P	tol_{NE}	err_E(eV/atom)	err_F(eV Å^{-1})
DNA-1	50	10^{-4}	2×10^{-8}	2×10^{-6}
C-$BN_{2.3}$	40	10^{-3}	3×10^{-6}	2×10^{-5}
H_2O-8	40	10^{-4}	3×10^{-6}	6×10^{-5}
Al	40	10^{-3}	6×10^{-6}	2×10^{-6}
SiH	40	10^{-3}	10^{-4}	7×10^{-5}
SiH	60	10^{-4}	6×10^{-6}	7×10^{-5}

We use DNA-a to denote a DNA strand with a unit cells, C-BN_α to denote C-BN layers in which the Boron-Nitride layer is rotated by α degrees relative to the graphene sheet (for this set of systems a lower α implies a larger system size), and H_2O-n to denote a box of liquid water with n unit cells. The various instances of DNA, C-BN, and H_2O are used to test the performance (including parallel scaling) and accuracy of SIESTA-PEXSI, while the smaller Al and SiH systems are used mainly to showcase the accuracy of SIESTA-PEXSI and the effectiveness of the hybrid inertia counting plus Newton's method for finding the correct chemical potential for metallic systems.

In all cases we use a DZP basis set, which results in 13 orbitals per atom for C, N, B, O, and P, and 5 orbitals per H atom.

3.1. Accuracy of the SIESTA-PEXSI approach

We now report the accuracy of SIESTA-PEXSI in terms of the computed energies and atomic forces. We measure the accuracy of the energy by examining the difference err_E between the free energy computed by SIESTA-PEXSI and that computed by the standard SIESTA approach in which free energies are obtained from density matrices constructed from the eigenvectors of (H, S). A similar metric is used for assessing the accuracy of atomic forces. We denote by err_F the maximum force difference among all atoms.

In table 4 we list the differences of energy per atom and force for fully converged calculations for a number of test problems with indication of the number of poles used in the pole expansion and of the target tolerance for electron count used in the chemical potential search.

In all our tests we set the electronic temperature to 300 K.

As we can see from this table, the errors in the energy computed by SIESTA-PEXSI are typically of the order of 10^{-6} eV per atom. The maximum error in atomic forces is of the order of 10^{-5} eV Å^{-1}. Both are sufficiently small for most applications, and can be achieved with a modest number of poles and reasonable accuracy tolerance on μ. For SiH, P=60 and $tol_{ne} = 10^{-4}$ bring the error in the energy per atom to the μ eV level. In this more delicate case, one needs the extra accuracy to locate precisely the Fermi level, which is pinned in a gap state.

3.2. *Efficiency of the SIESTA-PEXSI approach*

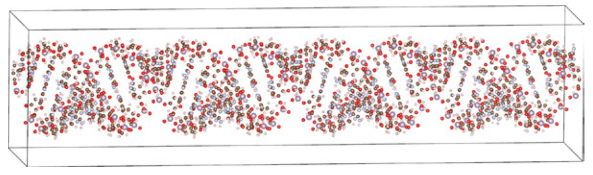
We now report the performance of SIESTA-PEXSI in comparison to that of the standard SIESTA approach in which density matrices are obtained from the eigenvectors of (H, S) computed by the ScaLAPACK diagonalization procedure based on the suite of subroutines pdpotrf, pdsyngst, pdsyevd/pdsyevx, and pdtrsm for transformation to a standard eigenvalue problem, solution, and back transformation, respectively.

We use instances of the DNA, C-BN and H_2O problems to test both the strong scaling of the solver, which is measured by the change in wallclock time as a function of the number of processors used to solve a problem of fixed size, and the weak scaling, which is measured by the change of wall clock time as we increase in tandem both the problem size and the number of processors used in the computation.

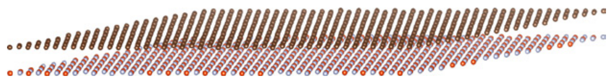
Our timing measurements refer to a single SCF step. For completeness they include also the time used to setup the Hamiltonian and the overlap matrices, but this is in any case a very small fraction of the total (<5%) for these systems. The symbolic factorization is inexpensive compared to numerical factorization and selected inversion, and can be computed once for the entire SCF iteration using only the sparsity structure of H and S, and the time for symbolic factorization is excluded in the timings. It should be noted that the data for the diagonalization-based method include the time involved in constructing the density and energy-density matrices from the eigenvectors, needed at each SCF step.

The cost of the diagonalization method is determined directly by H and S and the internal, basically hardwired operating parameters in ScaLAPACK. The chemical potential is computed from the list of eigenvalues. In contrast, SIESTA-PEXSI determines the chemical potential iteratively in each SCF step, and thus the computational load depends on the actual sequence of bracketing and refining of μ followed. In order to provide an appropriate reference with which to compare the diagonalization-based results, our SIESTA-PEXSI timings include one (H_2O) or two (DNA and C-BN) inertia counting cycles to narrow down the search interval for the chemical potential and one call of the PEXSI solver.

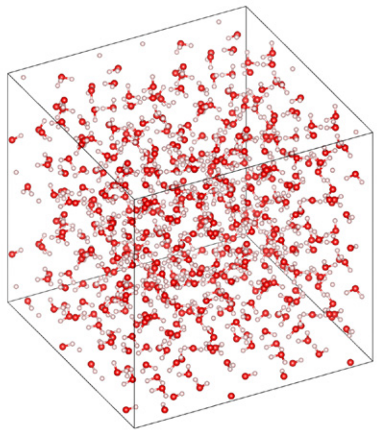
As mentioned above, one or two calls of the PEXSI solver are typically needed for an appropriately precise computation



(a)



(b)

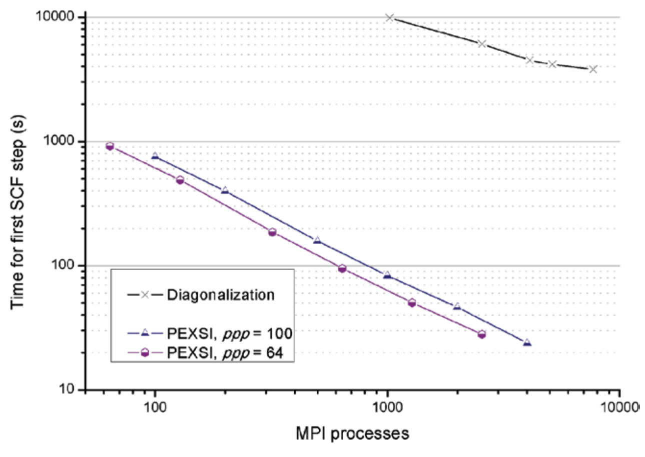


(c)

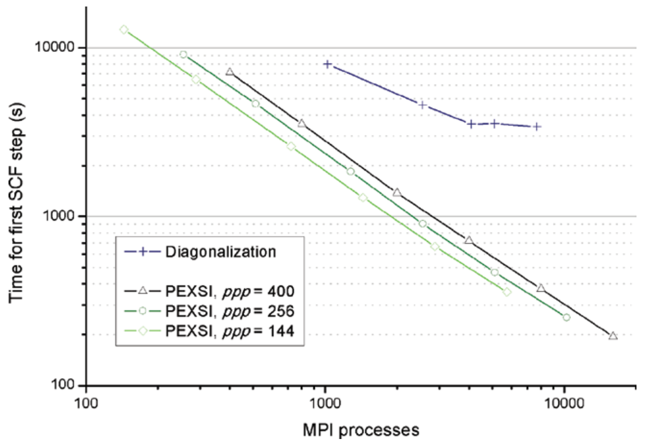
Figure 2. The atomic configuration of (a) the DNA-4 system (b) the C-BN$_{2.3}$ system and (c) the H_2O-8 system.

of μ as the SCF cycle unfolds. Given a good strategy for keeping a tight bracketing of μ, the final steps in the cycle close to convergence are likely to need just one call, and the same behavior is expected for most of the cycle when a good guess of the starting electronic structure can be provided, as in molecular-dynamics or geometry-optimization simulation. It should also be noted that later steps in the SCF cycle will not need the inertia counting procedure. Our timings for SIESTA-PEXSI are thus representative on average of the computational effort expected for a given system. In some cases, the actual effort will be higher by a small factor, and in others (as in systems with a gap) might even be smaller.

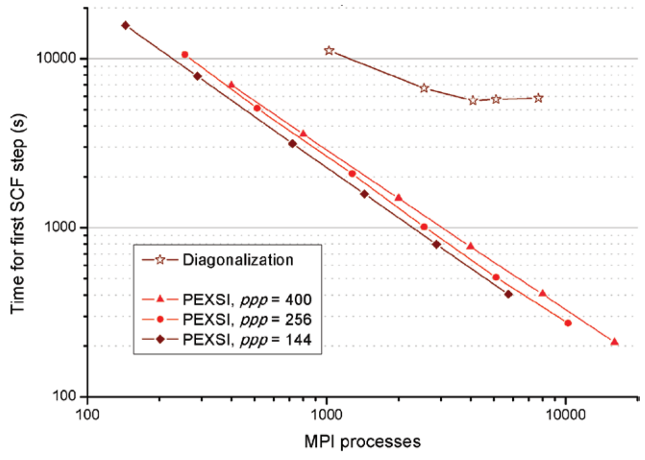
In figure 3, we plot the wallclock time required to complete the first SCF step as a function of the total number of processors used, for both SIESTA-PEXSI and the standard diagonalization method in SIESTA, when they are used to solve the DNA-25, C-BN$_{0.00}$, and H_2O-125 problems. In our experiments we used P = 40 poles in the pole expansion approximation of the Fermi-Dirac function and varying degrees of concurrency over poles: $np = k \times ppp$, where $k \in \{1, 2, 5, 10, 20, 40\}$. When k = 1, we simply loop serially over poles and perform parallel selected inversion on ppp processors at each pole. When $k = P$ = 40, full concurrency is achieved. For each test problem we connect all measurements that correspond to the same ppp with a line. The nearly perfect scaling exhibited by these lines reflects the embarrassingly parallel nature of pole expansion. The further reduction in the wallclock time



(a) DNA-25



(b) C-BN$_{0.00}$



(c) H_2O-125

Figure 3. Parallel strong scaling of SIESTA-PEXSI and the diagonalization approach when they are applied to DNA-25, C-BN$_{0.00}$ and H_2O-125 problems.

upon increasing *ppp* depends for a given system on its size and the degree of sparsity of the Cholesky factor L (or in L and U factors in the LU factorization).

We observe that increasing *ppp* from 64 to 100 leads to additional reduction in wallclock time by a factor of 1.2 for the DNA-25 problem. The L matrix of DNA is very sparse. The L matrices of C-BN and H_2O are less sparse, and more processors can be effectively used to reduce the wallclock time of selected inversion. We observe that increasing *ppp* from 144 to 400 leads to an additional speedup of 1.8 for C-BN$_{0.00}$, and of 2 for H_2O-125.

In figure 3, we also plot the wallclock time used by the standard diagonalization-based procedure. The computational complexity of diagonalization does not depend on the sparsity of the H and S matrices but only on their size, which is similar for all three systems, and therefore the performance of the method on these problems is comparable. The diagonalization curves in figure 3 start at around 1000 processors because this is the minimum number that would allow these problems to fit in memory, as the dense form of H and S is needed by the algorithm. A reasonably good parallel scaling can be observed up to 4000 processors, but the performance degrades after that.

We can also see that when around 1000 processors are used in the computation, SIESTA-PEXSI is approximately one order of magnitude faster than the diagonalization method for the C-BN$_{0.0}$ and H_2O-125 problems, and approximately two orders of magnitude faster for the DNA-25 problem. The performance gap between SIESTA-PEXSI and diagonalization widens as the number of processors used in the computation increases. This is due to the relatively limited scalability of ScaLAPACK, and to the fact that PEXSI can more efficiently utilize a large number of cores thanks to the two-level parallelism.

Even though there is a clear advantage of SIESTA-PEXSI in being able to use large numbers of processors, we should point out that its smaller memory footprint means that it can operate also with relatively small numbers of processors. Thus on Edison we can solve the DNA-25 and C-BN$_{0.00}$ problems by SIESTA-PEXSI with as few as 144 processors for C-BN$_{0.00}$ and 64 processors for DNA-25. For the DNA-25 problem, running SIESTA-PEXSI on 64 processors is still more than four times faster than running the standard diagonalization procedure on 5120 processors.

To measure the weak scaling of SIESTA-PEXSI, and compare it with that of the diagonalization-based procedure, we ran both solvers on multiple instances of the DNA, C-BN and H_2O systems with different sizes. We adjust the number of processors so that it is approximately proportional to the dimension of the H and S matrices, i.e., as the problem size increases, we use more processors to solve the larger problem. For diagonalization, the number of processes for each system size are chosen so that there are approximately 40–50 orbitals per processor.

For PEXSI, in all weak scaling tests, we use $P = 40$ poles and fully exploit the concurrency at the pole expansion level, i.e., the selected inversions associated with different poles are carried out simultaneously on different groups of *ppp* processors, for all sizes. This means that the increase in processor count with problem size is achieved with progressively higher *ppp* values.

In table 5 we report the wallclock time t_{iter} used to complete the first SCF step, and use the t_{iter} for the smallest size in each DNA, C-BN and H_2O series as the basis for measuring weak scaling. If we increase the number of processors by a factor of

Table 5. Configurations and times for the first SCF iteration, as presented in figure 5.

	Diagonalization		Siesta-PEXSI		
Example	Proc.	t_{iter} (s)	Proc.	t_{iter} (s)	t_{ideal} (s)
DNA-1	128	7.1	360	3.8	3.8
DNA-4	512	123	1000	6.8	5.5
DNA-9	1280	606	1960	10.9	6.3
DNA-16	2560	2005	2560	16.6	8.6
DNA-25	4096	4118	4000	24.4	8.6
$C\text{-}BN_{2.3}$	720	96.5	1440	69.2	69.2
$C\text{-}BN_{1.43}$	1280	393	2560	136	105
$C\text{-}BN_{0.57}$	2560	1422	5680	188	140
$C\text{-}BN_{0.00}$	4096	3529	10 240	248	157
H_2O-8	256	16.8	640	12.3	12.3
H_2O-27	720	272	2560	47.3	35.0
H_2O-64	2048	1375	5760	126	87.5
H_2O-125	4096	5641	10 240	314	188

α when the problem size is increased by a factor of β, the ideal weak scaling factor (in terms of the wall clock time) is

$$s_w = \frac{\beta}{\alpha},$$

for DNA problems due to the quasi-1D nature. The ideal weak scaling factors are $\frac{\beta^{3/2}}{\alpha}$ and $\frac{\beta^2}{\alpha}$ for C-BN and H_2O systems, due to the quasi-2D and 3D nature of the systems, respectively. Table 5 shows the ideal time for the PEXSI runs computed using those ideal weak scaling factors for each instance $t_{\text{ideal}} = t_0 \times s_w$, where t_0 is the wallclock time for the smallest instance in each series. The weak-scaling for diagonalization is close to the expected cubic scaling $s_w = \frac{\beta^3}{\alpha}$, and the corresponding ideal time is not shown for simplicity.

Another way to study the weak scaling of the SIESTA-PEXSI approach is to examine how the total computational cost, which is the product of the wallclock time and the number of processors (i.e., the first two columns for each method in table 5), changes with respect to the problem size. For perfect weak scaling, the change in cost should match that predicted by the computational complexity of the sequential algorithm. We plot the cost involved in solving each instance of the DNA, C-BN and H_2O problems in Figure 5, which shows that the data for each series falls approximately on a line. The (fitted) slope of the cost for DNA, which includes both factorization and selected inversion, is observed to be approximately 1.3, higher than the $O(N)$ ideal scaling for quasi-1D systems.

For C-BN the slope of the fitting line is approximately 1.7, which is a bit larger than the $O(N^{1.5})$ ideal scaling for quasi-2D systems. The slope of the line for the H_2O series is approximately 2.18, again slightly larger than the expected $O(N^2)$ scaling for 3D systems. The observed degradation in parallel scaling (relatively larger for DNA) comes about because neither selected inversion nor factorization (which has the same asymptotic complexity) can scale perfectly due to the complicated data communication and task dependency involved.

It should be noted also that the ideal weak-scaling data in table 5 depends on the size and processor count chosen for the smallest instance of each series. We have already mentioned that all systems considered in our timings have been run with complete parallelization over poles. This limits the gains from increasing processor counts to the actual efficiency of selected-inversion and factorization within a pole, as a function of *ppp*. Roughly, the computational load for a given instance will depend on the number of non-zeros in the Cholesky factor L, which is plotted as a function of matrix size in figure 4. It can be seen that C-BN has the highest such non-zero density, closely followed by H_2O. This is because C-BN has a more densely packed structure than H_2O, and thus a larger prefactor, but there is a clear trend for a faster increase in the number of non-zeros in the Cholesky factor for H_2O, in agreement with the asymptotic complexity of 3D and quasi-2D systems, respectively. The DNA system has significantly fewer non-zero values in L for a given size of H, and so it can use a relatively smaller number of processors per pole efficiently.

We can also see from figure 5 that the crossover point at which SIESTA-PEXSI becomes more efficient (in terms of cost) than the standard diagonalization procedure in SIESTA is around matrix dimension Nx = 7000 (∼ 700 atoms) for

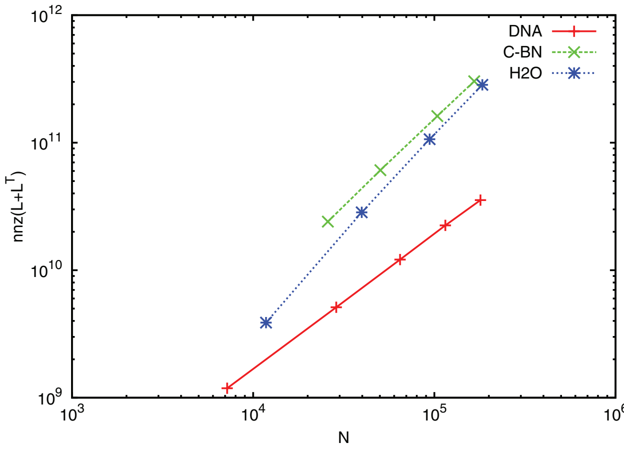


Figure 4. Number of non-zeros in the Cholesky factor of H for the systems shown on figure 5, as a function of the dimension of H.

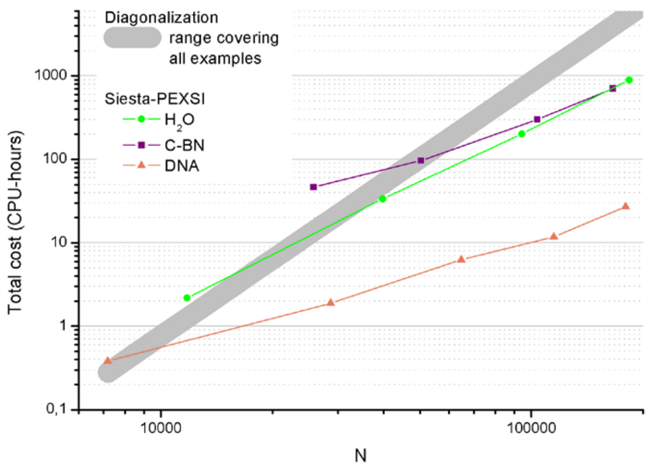


Figure 5. Weak scaling based on computational cost (time × number of processors) for the first SCF step for the H_2O, C-BN, and DNA examples. The points correspond to the various problem sizes as listed in table 3. The times for diagonalization show some variations due to differences in the construction of H and the building of the Γ matrix, and are represented by the gray stripe. The configurations and results can be found in more detail in table 5.

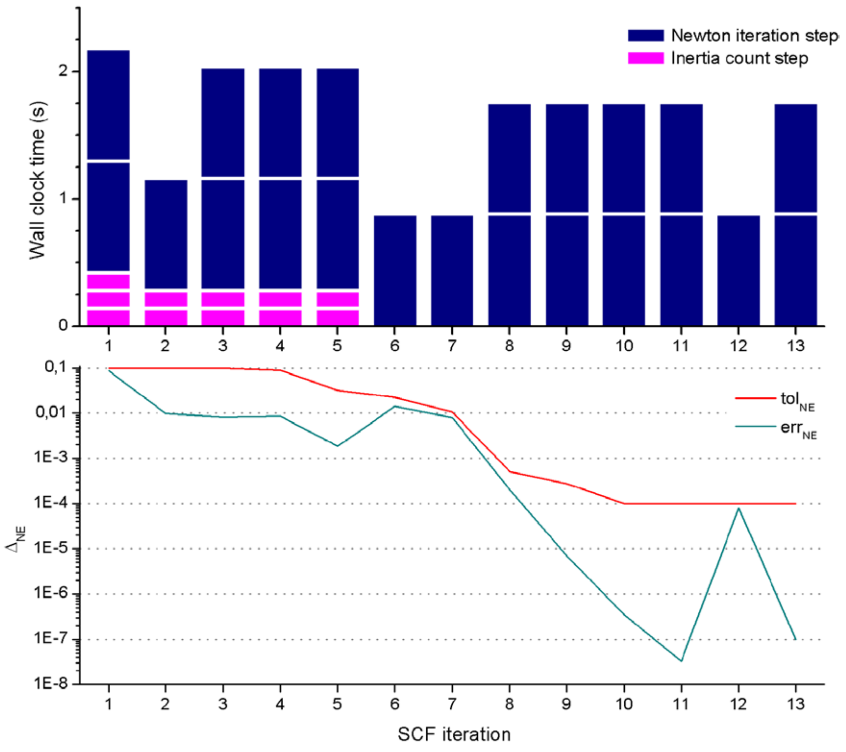


Figure 6. Convergence history in the SCF cycle for the SiH system. The top graph shows the time per SCF step and a breakdown into the number of inertia counting and PEXSI solver iteration (Newton iteration) steps. The bottom diagram draws the (adaptive) target tolerance and the actual error in the number of electrons at the end of each SCF step.

the DNA problem, $N = 50\,000$ (~ 4000 atoms) for the C-BN problem and $N = 40\,000$ (~ 5000 atoms) for the H_2O problem.

The actual orderings of the crossover points depend on the dimensionality (through the power of N in the scaling), and also on a prefactor which is system dependent.

Since more processors can be used efficiently, the advantage of SIESTA-PEXSI in time-to-solution performance is even more pronounced than the benefit from the reduction of total work load, and is already evident for smaller systems, as can be seen in table 5.

The conclusions regarding the performance comparison between PEXSI and diagonalization, being based on the asymptotic scaling of the algorithm, remain largely valid even taking into account the efficiency improvements that can been obtained by refactoring some of the internals of ScaLAPACK [35].

3.3. The search for chemical potential

As we discussed in section 2.3, the current implementation of SIESTA-PEXSI uses a combination of an inertia counting bracketing technique and Newton's method to determine the chemical potential μ that satisfies the nonlinear equation $N_\beta(\mu) = N_e$, where $N_\beta(\mu)$ is defined in (13). In this section, we demonstrate the effectiveness of this approach for both metallic and insulating systems.

The metallic example is the SiH system with 257 electrons. Since our calculation is spin-restricted, the odd number of electrons is solely controlled through the fractional occupation of the H-derived state in the gap. Any small perturbation of the chemical potential can change the number of electrons, and the perturbation is temperature dependent (we use 300 K). Despite the relatively small system size, this can be considered as the most difficult case for finding the chemical potential without having access to the eigenvalues. Figure 6 illustrates the number of inertia counting cycles and PEXSI solver steps needed to obtain a chemical potential that satisfies $|N_\beta(\mu) - N_e| < 10^{-4}$ for the converged SiH system when using 60 poles. The calculation starts from a wide initial μ search interval of (−27.0, 0.0) eV, and uses an adaptive electron tolerance which starts at a coarse $tol_{Ne} = 0.1$ and progressively tightens towards the target value of 10^{-4} as the deviation from self-consistency in the H matrix elements moves towards its tolerance target of 10^{-5} Ry. The number of inertia counting steps needed to narrow down the interval that contains the true chemical potential is roughly 2 ~ 3 in the first few SCF steps.

After the 5th iteration, the inertia counting procedure is turned off. The number of PEXSI solver steps needed is at most 3 in the first few SCF steps, and it becomes 2 after the 6th SCF step. Figure 6 also shows the absolute error $|N_\beta(\mu) - N_e|$ at each SCF step, which is always below the set tolerance.

Our test for insulating systems is the DNA-1 system with 2442 electrons, at 300 K and using 50 poles. Figure 7 illustrates the number of inertia counting and PEXSI solver steps needed to reduce $|N_\beta(\mu) - N_e|$ to a level below a final tolerance of 10^{-4}, starting at a coarser tolerance of 1. The calculation also starts from a wide initial interval of (−27.0, 0.0) eV. Usually two inertia counting steps are needed in the first few SCF iterations. The second iteration requires one more due to

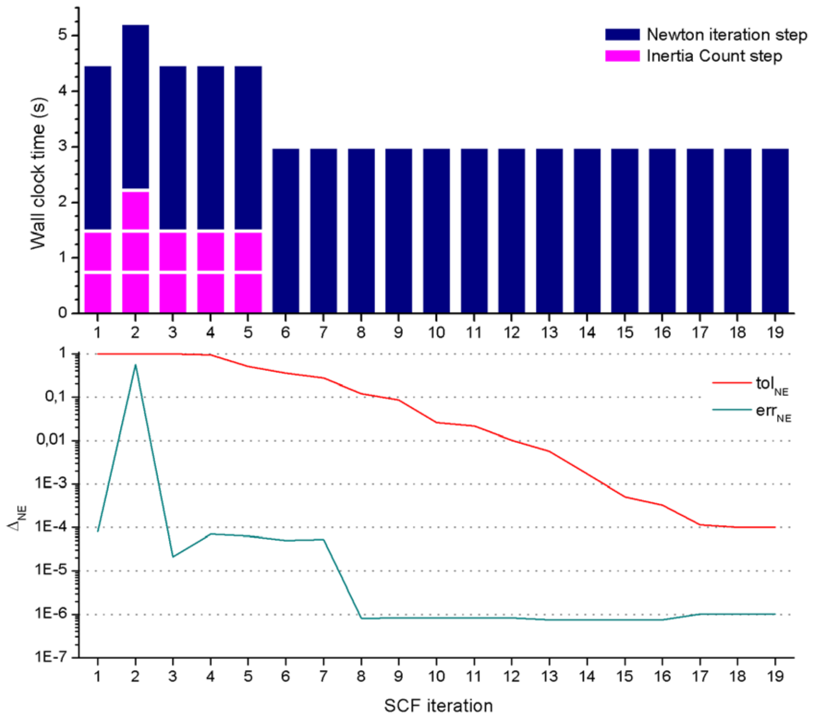


Figure 7. Convergence history in the SCF cycle for the DNA-1 system. The top graph shows the time per SCF step and a breakdown into the number of inertia counting and PEXSI solver iteration (Newton iteration) steps. The bottom diagram draws the (adaptive) target tolerance and the actual error in the number of electrons at the end of each SCF step.

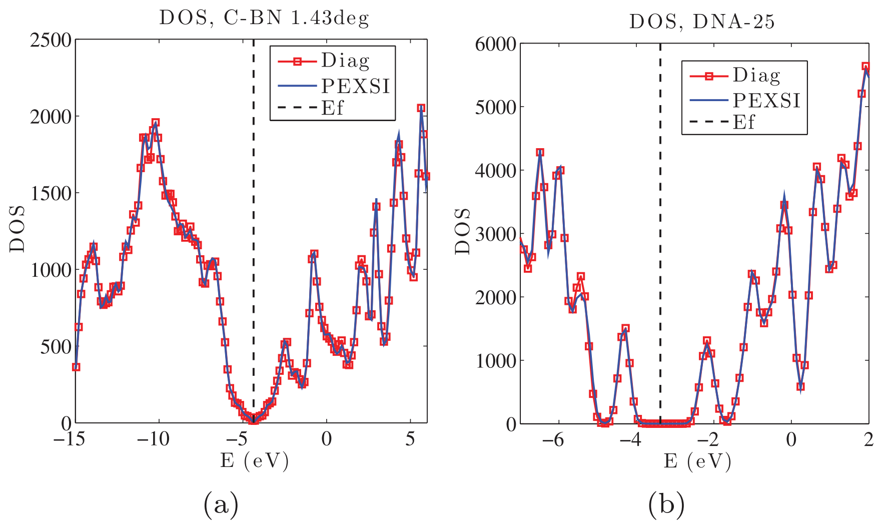


Figure 8. DOS around the Fermi level for the 1.43 degree C-BN and DNA-25 systems. Figure 9 presents the LDOS for the SiH system for an energy interval of width 0.4 eV around the Fermi level, showing the state due to the added H atom in the bulk Si system.

a large jump in H. After the 5th iteration, the inertia counting procedure is completely turned off. Throughout the SCF cycle, which ends when the deviation from self-consistency in H is below 10^{-5} Ry, just one PEXSI solver call is needed to provide a solution with a number of electrons within the tolerance. This is because μ is kept within the gap by the bracketing procedure.

3.4. DOS and LDOS

In section 2.4, we described how to use SIESTA-PEXSI to obtain spectral information of the atomistic system such as DOS and LDOS without computing eigenvalues and eigenvectors of (H, S). In figure 8(*a*) we plot the DOS obtained from the standard diagonalization method and the inertia counting procedure implemented in SIESTA-PEXSI for the C-BN$_{1.43}$ system near the Fermi level (‘E_f’). We observe nearly perfect agreement between the DOS curves obtained from these two approaches.

Figure 8(*b*) shows the DOS near the Fermi level for the DNA-25 system using both the inertia counting method and diagonalization. The usage of PEXSI for evaluating DOS without obtaining eigenvalues is also significant for systems at large size. For the DNA-25 system, using 64 *ppp* and 3200

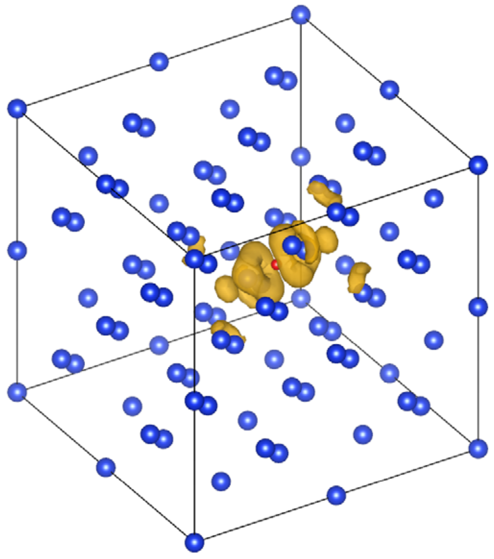


Figure 9. Isosurface of the LDOS around the Fermi level for the interstitial H atom in the Si crystal.

processors to evaluate 200 points in the DOS near the Fermi energy (each group of 64 processors evaluate 4 points of cumulative DOS with inertia counting) only takes 34 s, while diagonalization using the same number of processors takes 4865 seconds. It should be noted that the performance of PEXSI can be further improved by simply using more cores: after the SCF converges, one can restart the calculation to compute the DOS using the converged density matrix, and use 12 800 processors to compute 200 points in the DOS in parallel. In such case, the wall clock can be reduced to 9 seconds.

4. Conclusion

We have combined the pole expansion and selected inversion (PEXSI) technique with the SIESTA method for Kohn–Sham density functional theory (KDSFT) calculation. The resulting SIESTA-PEXSI method can efficiently use more than 10 000 processors, and is particularly suitable for performing large scale *ab initio* materials simulation on high performance parallel computers. The SIESTA-PEXSI method does not compute eigenvalues or eigenvectors of the Kohn–Sham Hamiltonian, and its accuracy is fully comparable to that obtained from the standard matrix diagonalization based SIESTA calculation for general systems, including insulating, semi-metallic, and metallic systems at low temperature.

The current implementation of SIESTA-PEXSI does not yet support spin-polarized systems, but this can be achieved with minor changes to the code. Furthermore, the code does not support **k**-point sampling, as in principle it is not needed for large-enough systems. However, in some cases, as in graphene and similar semi-metallic systems, an appropriate computation of the spectral properties such as the DOS does need **k**-point sampling even when very large supercells are used. We will address this problem in future work, which requires modifying PEXSI to handle complex Hermitian H matrices. In such case the shifted matrix $H - zS$ is no longer symmetric but is only structurally symmetric. This further development will also be useful for *ab initio* study of non-collinear magnetism and spin-orbit coupling effects.

Acknowledgments

This work was partially supported by the Laboratory Directed Research and Development Program of Lawrence Berkeley National Laboratory under the US Department of Energy contract number DE-AC02-05CH11231, by Scientific Discovery through Advanced Computing (SciDAC) program funded by the US Department of Energy, Office of Science, Advanced Scientific Computing Research and Basic Energy Sciences, by the Center for Applied Mathematics for Energy Research Applications (CAMERA), which is a partnership between Basic Energy Sciences and Advanced Scientific Computing Research at the US Department of Energy (LL and CY), by the European Community's Seventh Framework Programme [FP7/2007-2013] under the PRACE Project grant agreement number 283493 (G. H.), by the Spanish MINECO through grants FIS2009-12721-C04-03, FIS2012-37549-C05-05 and CSD2007-00050 and by Generalitat de Catalunya (2014 SGR 301) (AG).

References

[1] Bowler D R, Miyazaki T and Gillan M J 2002 *J. Phys.: Condens. Matter* **14** 2781
[2] Fattebert J L and Bernholc J 2000 *Phys. Rev.* B **62** 1713
[3] Hine N D, Haynes P D, Mostofi A A, Skylaris C K and Payne M C 2009 *Comput. Phys. Commun.* **180** 1041
[4] Yang W 1991 *Phys. Rev. Lett.* **66** 1438
[5] Li X-P, Nunes R W and Vanderbilt D 1993 *Phys. Rev.* B **47** 10891
[6] McWeeny R 1960 *Rev. Mod. Phys.* **32** 335
[7] Goedecker S 1999 *Rev. Mod. Phys.* **71** 1085
[8] Bowler D R and Miyazaki T 2012 *Rep. Prog. Phys.* **75** 036503
[9] Kohn W 1996 *Phys. Rev. Lett.* **76** 3168
[10] Prodan E and Kohn W 2005 *Proc. Natl Acad. Sci.* **102** 11635
[11] Lin L, Lu J, Ying L and E W 2009 *Chin. Ann. Math.* **30** 729
[12] Lin L, Lu J, Ying L, Car R and E W 2009 *Commun. Math. Sci.* **7** 755
[13] Lin L, Yang C, Meza J, Lu J, Ying L and E W 2011 *ACM. Trans. Math. Softw.* **37** 40
[14] Lin L, Chen M, Yang C and He L 2013 *J. Phys.: Condens. Matter* **25** 295501
[15] Jacquelin M, Lin L and Yang C 2014 arXiv:1404.0447
[16] Junquera J, Paz O, Sanchez-Portal D and Artacho E 2001 *Phys. Rev.* B **64** 235111
[17] Chen M, Guo G C and He L 2010 *J. Phys.: Condens. Matter* **22** 445501
[18] Kenny S D, Horsfield A P and Fujitani H 2000 *Phys. Rev.* B **62** 4899
[19] Ozaki T 2003 *Phys. Rev.* B **67** 155108
[20] Blum V, Gehrke R, Hanke F, Havu P, Havu V, Ren X, Reuter K and Scheffler M 2009 *Comput. Phys. Commun.* **180** 2175
[21] Frisch M, Pople J and Binkley J 1984 *J. Chem. Phys.* **80** 3265
[22] VandeVondele J, Krack M, Mohamed F, Parrinello M, Chassaing T and Hutter J 2005 *Comput. Phys. Commun.* **167** 103
[23] Chelikowsky J, Troullier N and Saad Y 1994 *Phys. Rev. Lett.* **72** 1240
[24] Tsuchida E and Tsukada M 1995 *Phys. Rev.* B **52** 5573

[25] Tsuchida E and Tsukada M 1998 *J. Phys. Soc. Japan.* **67** 3844
[26] Lin L, Lu J, Ying L and E W 2012 *J. Comput. Phys.* **231** 2140
[27] Becke A D 1993 *J. Chem. Phys.* **98** 5648
[28] Adamo C and Barone V 1999 *J. Chem. Phys.* **110** 6158
[29] Soler J M, Artacho E, Gale J D, García A, Junquera J, Ordejón P and Sánchez-Portal D 2002 *J. Phys.: Condens. Matter* **14** 2745
[30] Artacho E *et al* 2008 *J. Phys.: Condens. Matter* **20**
[31] Goedecker S 1993 *Phys. Rev.* B **48** 17573
[32] Karypis G and Kumar V 1998 *J. Parallel Distrib. Comput.* **48** 71
[33] Chevalier C and Pellegrini F 2008 *Parallel Comput.* **34** 318
[34] Li X and Demmel J 2003 *ACM Trans. Math. Softw.* **29** 110
[35] Auckenthaler T, Blum V, Bungartz H J, Huckle T, Johanni R, Krämer L, Lang B, Lederer H and Willems P R 2011 *Parallel Comput.* **37** 783
[36] Sylvester J J 1852 *Phil. Mag.* **4** 138
[37] Lin L, Saad Y and Yang C 2013 arXiv:1308.5467
[38] Plemelj J and Radok J R M 1964 *Problems in the Sense of Riemann and Klein* (New York: Interscience)