phys. stat. sol. (b) **243**, No. 5, 973–988 (2006) / **DOI** 10.1002/pssb.200541328

Feature Article

Implementation of linear-scaling plane wave density functional theory on parallel computers

Chris-Kriton Skylaris[*, 1], **Peter D. Haynes**[2], **Arash A. Mostofi**[3], and **Mike C. Payne**[2]

[1] Physical & Theoretical Chemistry Laboratory, South Parks Road, University of Oxford, Oxford OX1 3QZ, UK
[2] Theory of Condensed Matter group, Cavendish Laboratory, University of Cambridge, J. J. Thomson Avenue, Cambridge CB3 0HE, UK
[3] Department of Materials Science and Engineering, Massachusetts Institute of Technology, Cambridge MA, USA

Received 30 September 2005, revised 9 December 2005, accepted 16 December 2005
Published online 2 February 2006

PACS 31.15.Ew, 71.15.–m, 71.15.Dx, 71.15.Mb

We describe the algorithms we have developed for linear-scaling plane wave density functional calculations on parallel computers as implemented in the ONETEP program. We outline how ONETEP achieves plane wave accuracy with a computational cost which increases only linearly with the number of atoms by optimising directly the single-particle density matrix expressed in a psinc basis set. We describe in detail the novel algorithms we have developed for computing with the psinc basis set the quantities needed in the evaluation and optimisation of the total energy within our approach. For our parallel computations we use the general Message Passing Interface (MPI) library of subroutines to exchange data between processors. Accordingly, we have developed efficient schemes for distributing data and computational load to processors in a balanced manner. We describe these schemes in detail and in relation to our algorithms for computations with a psinc basis. Results of tests on different materials show that ONETEP is an efficient parallel code that should be able to take advantage of a wide range of parallel computer architectures.

1 Introduction

First principles electronic structure calculations are typically very computer intensive. As they are applied to progressively larger systems the need to use parallel computers has become increasingly necessary. However, even density functional theory (DFT) [1] which combines a high degree of accuracy with computational efficiency suffers from a significant scaling problem. The computational effort of conventional implementations increases with the cube of the number of atoms in the calculation. This fact negates to a great extent the benefits of parallel computing for increasing the scale of the simulations. The recent emergence of linear-scaling or $\mathcal{O}(N)$ methods [2] which promise to perform DFT computations with a computational cost which increases only linearly with the number of atoms, and with no significant loss in accuracy, will allow the extension of DFT studies to larger systems. Linear-scaling methods are more complicated than the conventional methods but can take full advantage of parallel computing

[*] Corresponding author: e-mail: chris-kriton.skylaris@chem.ox.ac.uk

resources. For this reason their development has often advanced hand-in-hand with the development of parallel computer algorithms for their implementation. As the essence of linear-scaling is that the calculation can be divided into parts associated with spatially localised regions [2], linear-scaling methods naturally lend themselves more easily to a parallel implementation than conventional cubic-scaling methods which deal with delocalised quantities. Thus, in linear-scaling methods it is natural to map the spatially localised parts onto processors and this is the approach that has been used to parallelise most such codes. For example, in the CONQUEST code [3] the parallelisation of data involves the distribution of the localised "support functions" to processors, while in the SIESTA code [4] it is the localised numerical atomic orbitals that are distributed to processors and in Gaussian function codes the localised Gaussian functions are distributed [5]. The recognition of this similarity at a high level between apparently very different linear-scaling approaches has also been the motivation for a recent work [6] that aims to create a parallelisation interface that could be used with an arbitrary localised function method. Yet, even though linear-scaling methods have much in common at a high level, they do differ greatly in their practical details and consequently every parallel implementation is unique as it seeks to take the best advantage of all the features of the method (basis set, sparse matrix format, etc.) in order to achieve optimum performance.

The ONETEP program [7] which we have developed for linear-scaling DFT calculations has been shown to be capable of performing highly accurate calculations on a variety of materials [8]. The method was developed from the beginning with parallel computers in mind. From a purely phenomenological point of view the ONETEP method combines concepts from plane-wave codes [9, 10], where all quantities are delocalised, and from localised function codes such as Gaussian basis function codes [11] and numerical atomic orbital codes [4]. Nevertheless, we have very little in common with these methods from the point of view of implementation. Consequently, we have developed entirely novel algorithms for parallel computations with the ONETEP approach. The purpose of this paper is to give an overview of these algorithms.

Our programming approach has been to write code using standard FORTRAN 95 combined with calls to the Message Passing Interface (MPI) [12] library set of subroutines. This approach is highly portable as it allows parallel calculations to be performed through message passing on both shared memory and distributed memory computer architectures. It is based on the Multiple Instruction Multiple Data (MIMD) paradigm and in our implementation, as in most MIMD programming examples [13], we run the same copy of the program on each processor but both the data and the operations are distributed amongst the processors.

We begin with Section 2 where we describe the procedures followed for total energy optimisation in ONETEP. In Section 3 we describe the algorithms we have developed for the calculation of the most computationally intensive quantities needed in order to compute and optimise the total energy. In Section 4 that follows we develop our data parallelisation strategy which takes advantage of real space localisation. Section 5 describes how we combine the data parallelisation with communication to perform computations in parallel with an arbitrary number of processors. Finally we show in Section 6 examples of the performance of our parallel implementation on a variety of materials, and we finish with some conclusions in Section 7.

2 Optimisation of total energy in the ONETEP code

To understand the total energy optimisation in the ONETEP code it is helpful to compare it and contrast it with conventional plane wave approaches and with other linear-scaling approaches. As in many linear-scaling DFT approaches that are based on the density matrix [14–19] and aim for high accuracy, the density matrix in ONETEP [7] is represented as a quadratic form

$$\rho(\boldsymbol{r},\boldsymbol{r}') = \sum_{\alpha\beta} \phi_\alpha(\boldsymbol{r})\, K^{\alpha\beta} \phi^*_\beta(\boldsymbol{r}') \,, \qquad (1)$$

where the $\{\phi_\alpha\}$ are a set of spatially localised, nonorthogonal generalised Wannier functions (NGWFs) [20] and the matrix $\boldsymbol{K}$ is called the density kernel [21]. With this form the exponential decay of the density matrix which is encountered in insulating materials [22–26] is imposed in a practical manner as the NGWFs are strictly localised in spherical regions and the $\boldsymbol{K}$ matrix is made sparse by enforcing the condition $K^{\alpha\beta} = 0$ when $|\boldsymbol{R}_\alpha - \boldsymbol{R}_\beta| > r_{\text{cut}}$, where $\boldsymbol{R}_\alpha$ and $\boldsymbol{R}_\beta$ are the centres of the localisation regions of NGWFs $\phi_\alpha(\boldsymbol{r})$ and $\phi_\beta(\boldsymbol{r})$, respectively. The parameter r_{cut} is a predetermined cut-off threshold whose value depends on the material under study. In the code its value can be set with the input parameter *kernel_cutoff* in units of a_0.

The electronic charge density $n(\boldsymbol{r})$ is equal to the diagonal elements of the density matrix $(n(\boldsymbol{r}) = \rho(\boldsymbol{r}, \boldsymbol{r}))$ and the total energy can be considered as a functional of the density kernel and the NGWFs

$$E[n] = E[\boldsymbol{K}, \{\phi_\alpha\}]\,. \tag{2}$$

In contrast, conventional cubic-scaling plane wave calculations express the energy as a functional $E[\{\psi_i\}]$ of the molecular orbitals $\{\psi_i\}$ which are orthogonal and delocalised and minimise it iteratively by e.g. a conjugate gradients (CG) procedure [9]. In the "all bands" variant this minimisation is performed in a self-consistent manner so that the effective potential is also constantly updated. After each CG step a correction is applied which restores the orthonormality of the molecular orbitals. As no localisation constraints are applied each orbital is free to extend over the entire volume of the simulation cell. In the ONETEP approach on the other hand the NGWFs are forced to respect strict localisation constraints but no orthogonality constraints. The orthogonality of the molecular orbitals as well as the requirement for occupancies to be either zero or unity is now transformed into the constraint of idempotency $\rho^2 = \rho$ which is imposed on the density matrix. Subject to this constraint and also the normalisation condition $\left(\int \rho(\boldsymbol{r}, \boldsymbol{r})\, \mathrm{d}\boldsymbol{r} = N_{\text{e}}, \text{ where } N_{\text{e}} \text{ is the total number of electrons}\right)$, the energy of Eq. (2) is minimised both with respect to the density kernel and the NGWFs. As we have previously described [20], we achieve this by using a procedure which involves two nested loops which can be interpreted as two constrained-search steps:

$$E_{\min} = \min_{\{\phi_\alpha\}} L\,[\{\phi_\alpha\}] \quad (\text{outer loop})\,, \tag{3}$$

with

$$L[\{\phi_\alpha\}] = \min_{\boldsymbol{K}} E\,[\boldsymbol{K},\{\phi_\alpha\}] \quad (\text{inner loop})\,, \tag{4}$$

where the minimisation with respect to $\boldsymbol{K}$ in Eq. (4) ensures that L of Eq. (3) is a functional of the NGWFs only.

Linear-scaling approaches proposed by other groups [27] consist of three nested loops. In that case the inner loop of Eq. (4) is further divided into two nested loops. Thus in the inner-most loop the band structure energy $E_{\text{BS}} = \sum_{\alpha\beta} K^{\alpha\beta} H_{\beta\alpha}$ is optimised in a non-self-consistent manner (i.e. by ignoring the dependence of $\boldsymbol{H}$ on $\boldsymbol{K}$), using either the Li–Nunes–Vanderbilt (LNV) approach [28] and/or the Palser–Manolopoulos approach [29], in the middle loop self-consistency is imposed by density mixing and in the outer loop the total energy is minimised with respect to the NGWFs. The three-loop procedure is more complicated and non-variational. In contrast, the two-loop procedure we follow is by construction fully variational and shares the same stability and advantages as the ensemble DFT method of Marzari-Vanderbilt–Payne [30] to which it is closely related. For our self-consistent minimisation of the energy with respect to $\boldsymbol{K}$, which is described in Eq. (4), we use our own modified version of the LNV (or its variant by Millam and Scuseria [31]) method combined with the penalty functional method of Haynes and Payne [32].

3 Algorithms for calculations with a psinc basis set

As ONETEP is, to our knowledge, the only electronic structure code using a psinc basis set [20, 33, 34] we have developed from scratch efficient algorithms for the computation of the total energy and all the quantities required for its optimisation. The psinc functions are highly localised and each one of them is centred on a point of a regular grid in real space, yet they are orthogonal. The orthogonality of the basis set combined with the fact that it is constructed from plane waves allows us to employ a very efficient length-scale preconditioning scheme [34] which makes the number of self-consistent steps needed to converge the energy to a given threshold per atom small and independent of the number of atoms. Figure 1 demonstrates this fact with several examples of calculations with ONETEP on different kinds of materials. The number of iterations depends slightly on the material but it always remains small and independent of the number of atoms. This non-trivial characteristic is absolutely necessary in order to maintain $\mathcal{O}(N)$ cost throughout the calculation on the total energy since the cost per NGWF iteration of Eq. (3) is already $\mathcal{O}(N)$.

Our NGWFs are expanded in the psinc basis and localisation is imposed by restricting their expansion coefficients to be nonzero only within predefined spherical localisation regions as shown on the right hand side of Fig. 2. In the code these coefficients can be stored directly in the simulation cell, or in the FFT box [35] or in terms of "parallelepipeds" or "PPDs" as shown in Fig. 2. The FFT box representation allows us to use the machinery of conventional plane wave calculations to perform momentum space

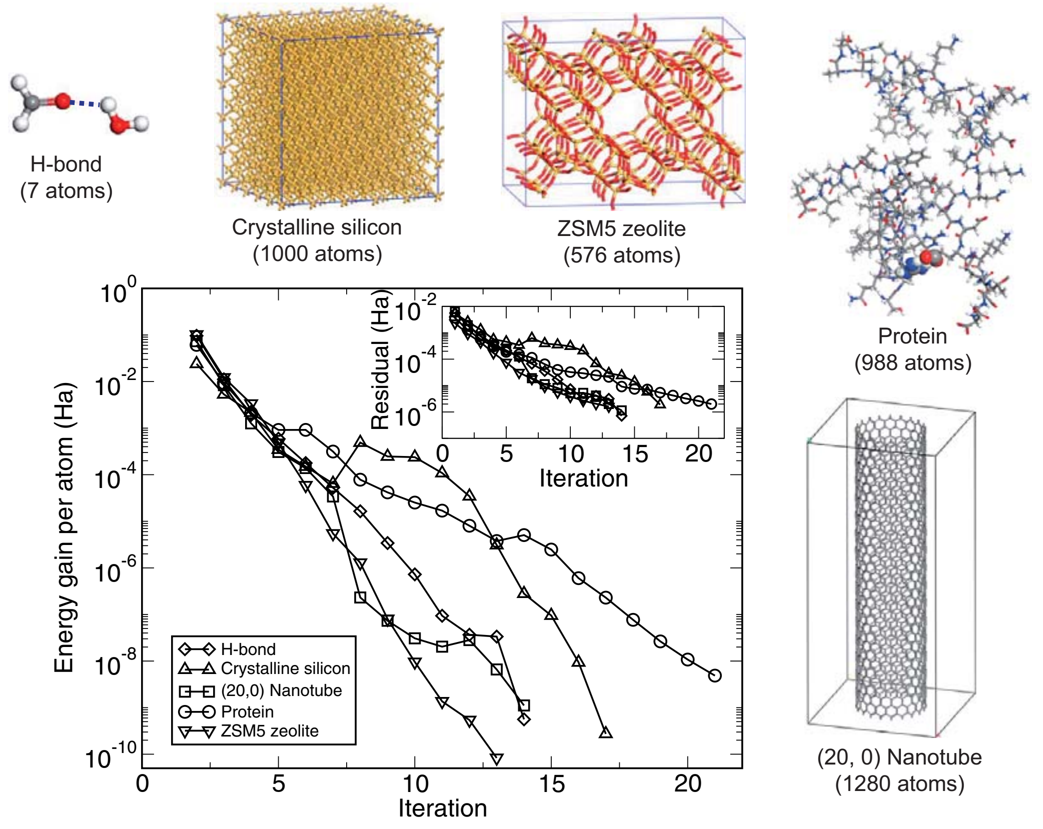


Fig. 1 Self-consistent convergence in calculations with ONETEP for a variety of materials. The main plot shows the change in the total energy (in Hartrees per atom) as a function of the iteration number for NGWF optimisation as in Eq. (3). The inset shows the root mean square value of the NWGF gradient at each such iteration.

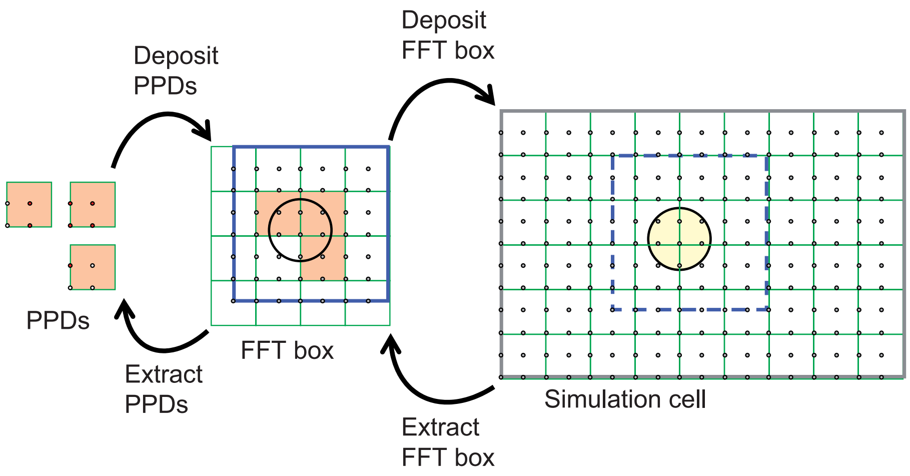


Fig. 2 From left to right: An NGWF stored in the compact, memory-efficient PPD representation can be deposited to or extracted from an FFT box. The FFT box makes possible operations in momentum space with computing cost which is independent of system size. The FFT box can in turn be deposited to or extracted from the simulation cell in real space. The localisation sphere of the NGWF and the subdivision in PPDs of the psinc grid in the FFT box and the simulation cell are also shown.

operations on NGWFs in a miniature simulation cell with a computing cost which is small and independent of the size of the entire simulation cell [36]. We are able to efficiently "extract" an NGWF from the simulation cell to an FFT box or conversely "deposit" it from an FFT box to the simulation cell, a procedure which we have formally defined previously in terms of simulation cell to FFT box projection operators [20, 36]. The simulation cell is subdivided into an integer number of PPDs along each lattice vector direction in such a way that each PPD is commensurate with the simulation cell and contains a small portion of the psinc grid, in a similar fashion to the domain decomposition approach that is commonly used in parallel implementations of real space methods [15, 19, 37–43]. The FFT box also contains a portion of the psinc grid in a fashion commensurate with the simulation cell. Apart from this condition, the FFT box can be placed anywhere in the simulation cell and is not required to be subdivided to an integer number of PPDs. The PPDs serve as a very compact, memory-efficient way of storing the NGWFs and are also the form in which the NGWFs are communicated between processors as we discuss later in Section 5. We have developed algorithms for efficiently extracting from an FFT box the PPDs that contain psinc grid points within the localisation region of an NGWF and for the converse process of depositing these PPDs to an FFT box.

Given only a single parameter, the psinc kinetic energy cut-off which is defined [8] in a fashion similar to the kinetic energy cut-off of plane wave basis sets, ONETEP determines the psinc grid, the subdivision of the simulation cell into PPDs and the size of the FFT box in such a way as to maximise computational speed, subject to geometrical constraints [35]. The psinc kinetic energy cut-off is specified with the input parameter *cutoff_energy* which can be given either in units of eV or Hartree.

In general, the action of a quantum mechanical operator on an NGWF results in its delocalisation over the entire volume of the FFT box [35]. This however poses no practical difficulty in achieving $\mathcal{O}(N)$ cost per self-consistent NGWF iteration as we will see in the remainder of this section where we describe how we compute efficiently the most computationally expensive quantities required for the total energy and its optimisation with the conjugate gradients approach. A rigorous mathematical justification for the following computations has been published elsewhere [20, 36] but the operations as described here and the order in which they are performed are optimised for speed rather than for clarity. In the figures that follow we will no longer be depicting the psinc grid but only the "coarser" PPD representation. Each of the algorithms we describe in the following Subsections is divided into "level 1" and "level 2" opera-

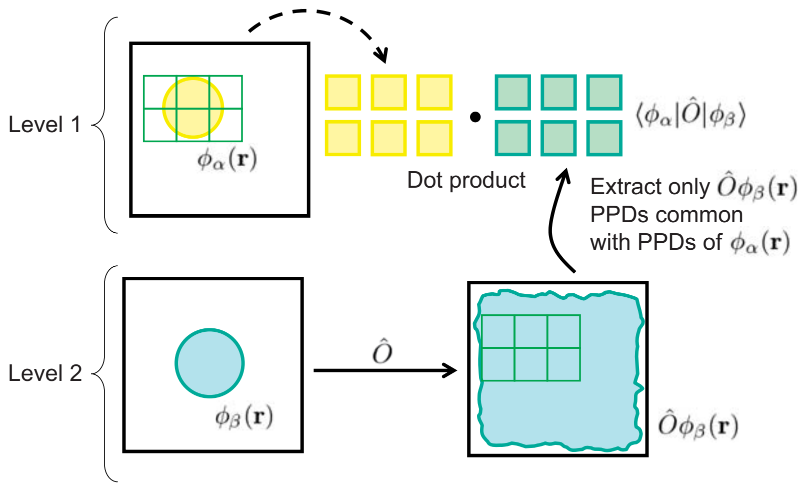

Level 1
$\phi_\alpha(\mathbf{r})$
Dot product
$\langle\phi_\alpha|\hat{O}|\phi_\beta\rangle$
Extract only $\hat{O}\phi_\beta(\mathbf{r})$ PPDs common with PPDs of $\phi_\alpha(\mathbf{r})$
Level 2
$\phi_\beta(\mathbf{r})$
$\hat{O}$
$\hat{O}\phi_\beta(\mathbf{r})$

Fig. 3 Calculation of $\langle\phi_\alpha|\hat{O}|\phi_\beta\rangle$. $\hat{O}$ is applied to $\phi_\beta(\boldsymbol{r})$ only once and causes it to delocalise over the entire volume of its FFT box. The integral is calculated as a dot product over the PPDs that belong to the localisation region of $\phi_\alpha(\boldsymbol{r})$.

tions. Level 1 operations involve mixing NGWFs from several overlapping localisation spheres while level 2 operations are carried out on a single NGWF localisation sphere.

3.1 Operator matrix elements

The calculation of the matrix elements $\langle\phi_\alpha|\hat{O}|\phi_\beta\rangle$ of the operator $\hat{O}$ in the NGWF representation is schematically described in Fig. 3. The operator $\hat{O}$ can be any of the individual components (kinetic energy, local potential, etc.) of the Hamiltonian. The $\hat{O}\phi_\beta(\boldsymbol{r})$ quantity which is delocalised over the entire volume of its FFT box is generated (level 2 task) once and for all and is kept in memory. Due to the orthogonality and cardinality of the psinc basis the $\langle\phi_\alpha|\hat{O}|\phi_\beta\rangle$ integral is simply equal to a dot product between the psinc grid points in common betweeen $\phi_\alpha(\boldsymbol{r})$ and $\hat{O}\phi_\beta(\boldsymbol{r})$ [20]. Since $\hat{O}\phi_\beta(\boldsymbol{r})$ is spread over all the psinc points of the FFT box, this in turn means that we need to take the dot product between just the points which fall within the localisation sphere of $\phi_\alpha(\boldsymbol{r})$. For efficiency, this operation is performed over the entire PPDs that belong to the localisation region of $\phi_\alpha(\boldsymbol{r})$. This is described in Fig. 3 as the level 1 operation because it is repeated for every single NGWF that overlaps $\phi_\beta(\boldsymbol{r})$. For the sake of clarity the function $\phi_\alpha(\boldsymbol{r})$ is also depicted inside an FFT box. However, it is never placed there in practice since it is only required in its readily available PPD representation.

3.2 Charge density

The generation of the charge density as the diagonal element of the density matrix of Eq. (1) is schematically described in Fig. 4. If we rewrite the charge density in the following form

$$n(\boldsymbol{r}) = \sum_\alpha n(\boldsymbol{r};\alpha) \quad \text{where} \quad n(\boldsymbol{r};\alpha) = \phi_\alpha(\boldsymbol{r}) \sum_\beta K^{\alpha\beta}\phi_\beta(\boldsymbol{r}) \,, \tag{5}$$

we just need to generate each $n(\boldsymbol{r};\alpha)$ component. This process involves two FFT boxes. In the first FFT box we place $\phi_\alpha(\boldsymbol{r})$ while in the second FFT box we accumulate the sum $\sum_\beta K^{\alpha\beta}\phi_\beta(\boldsymbol{r})$ which runs over all NGWFs which overlap with $\phi_\alpha(\boldsymbol{r})$ including itself, hence these are level 1 operations. Then level 2 operations follow where the contents of each FFT box are Fourier-interpolated to a psinc grid with twice

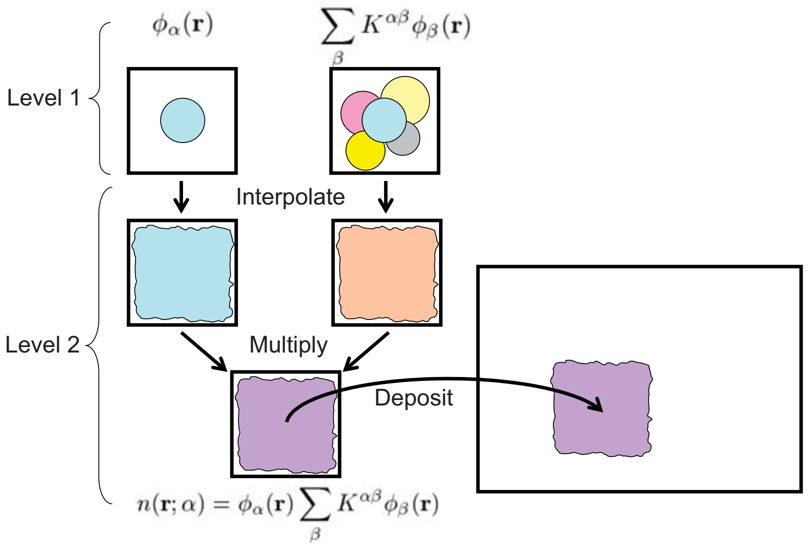

$\phi_\alpha(\mathbf{r})$
$\sum_\beta K^{\alpha\beta}\phi_\beta(\mathbf{r})$
Level 1
Interpolate
Level 2
Multiply
Deposit
$n(\mathbf{r};\alpha) = \phi_\alpha(\mathbf{r})\sum_\beta K^{\alpha\beta}\phi_\beta(\mathbf{r})$

Fig. 4 Calculation of a $n(\boldsymbol{r};\alpha)$ component of the charge density. The interpolation ensures that no aliasing takes place during the multiplication of the contents of the FFT boxes. The charge density is eventually accumulated in the simulation cell as the sum of all $n(\boldsymbol{r};\alpha)$ components.

as many points along each lattice vector direction. This interpolation ensures that the subsequent real space multiplication of the contents of the two FFT boxes will not introduce any aliasing errors, as a consequence of the fact that the psinc representation is entirely equivalent to a plane wave representation. After the interpolation the $n(\boldsymbol{r};\alpha)$ component is obtained as the product of the contents of the two FFT boxes and is deposited into the simulation cell. The process is repeated for the remaining $n(\boldsymbol{r};\alpha)$ components until the entire electronic charge density has been constructed in the simulation cell.

3.3 NGWF gradient

The calculation of the gradient of the energy with respect to the NGWF expansion coefficients in the psinc basis is schematically described in Fig. 5. Depending on tensor corrections [44] and various types of preconditioning, several variants for the NGWF gradient are possible, some of which have been described in the literature [17, 20, 45]. All variants have the following general form

$$\left.\frac{\delta E}{\delta\phi_\alpha}\right|_{\boldsymbol{r}} = \sum_\beta [\hat{H}\phi_\beta](\boldsymbol{r})\, A^\beta_\alpha + \sum_\gamma \phi_\gamma(\boldsymbol{r})\, B^\gamma_{\ \alpha}\,, \tag{6}$$

where the actual form of the matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ determines the gradient variant. The level 1 part of the calculation of $\delta E/\delta\phi_\alpha$ involves two FFT boxes, one for each of the terms of Eq. (6). Each of these terms is accumulated into its FFT box over the NGWFs that overlap with $\phi_\alpha(\boldsymbol{r})$. Then the level 2 operations follow where the contents of the two FFT boxes are added together and the PPDs belonging to the localisation sphere of $\phi_\alpha(\boldsymbol{r})$ are extracted from the resulting sum. Next we apply a stage of “shaving” of the extracted PPDs where the NGWF gradient values corresponding to psinc functions outside the localisation sphere of $\phi_\alpha(\boldsymbol{r})$ are zeroed. This ensures that the localisation of $\phi_\alpha(\boldsymbol{r})$ within its sphere will be preserved when the gradient is used in a conjugate gradient optimisation step. Finally the “shaved” gradient of the total energy with respect to $\phi_\alpha(\boldsymbol{r})$ in its PPD representation is stored and the procedure is repeated until the gradient with respect to all remaining NGWFs is also obtained.

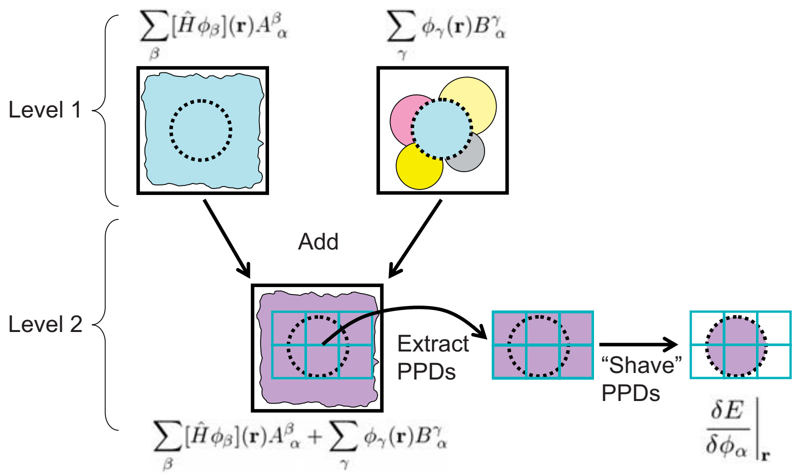

$\sum_{\beta}[\hat{H}\phi_{\beta}](\mathbf{r})A^{\beta}_{\alpha}$
$\sum_{\gamma}\phi_{\gamma}(\mathbf{r})B^{\gamma}_{\alpha}$
Level 1
Add
Level 2
Extract PPDs
"Shave" PPDs
$\sum_{\beta}[\hat{H}\phi_{\beta}](\mathbf{r})A^{\beta}_{\alpha}+\sum_{\gamma}\phi_{\gamma}(\mathbf{r})B^{\gamma}_{\alpha}$
$\left.\frac{\delta E}{\delta\phi_{\alpha}}\right|_{\mathbf{r}}$

Fig. 5 Calculation of the gradient of the energy with respect to the expansion coefficients of $\phi_{\alpha}(\boldsymbol{r})$ in the psinc basis. The resulting gradient is stored in the PPD representation after it is "shaved" to ensure that it will preserve the localisation of $\phi_{\alpha}(\boldsymbol{r})$.

4 Data parallelisation strategy

We will now turn our attention to issues specific to the parallel implementation of ONETEP by first examining the distribution of data to processors. It is critical that the distribution of data is done in such a way that load balancing is achieved in parallel computations. On the one hand the similarities of the ONETEP code with conventional real space methods would suggest the distribution of data according to the well-tested domain decomposition approach, making use of the subdivision of the simulation cell volume into PPDs. On the other hand, as we saw in Section 3, many of our computational algorithms use the FFT box to compute various quantities in momentum space with plane wave techniques. The FFT

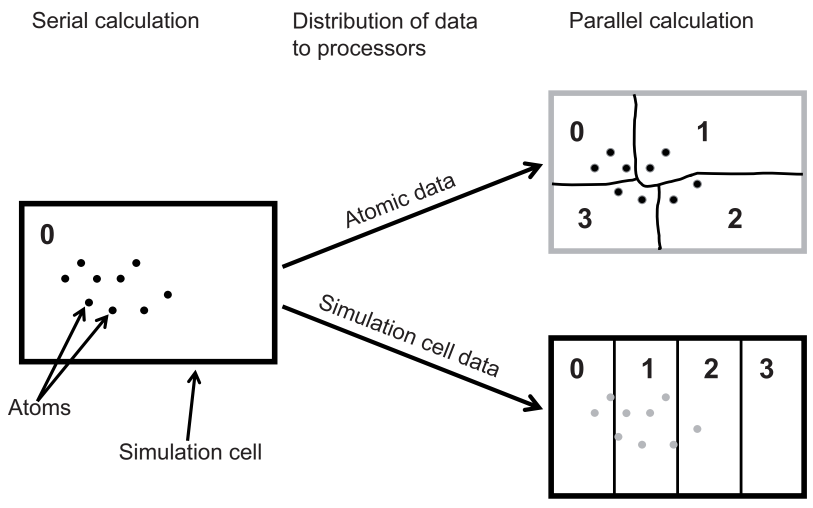


Fig. 6 Schematic example of the ONETEP parallelisation strategy for atomic and simulation cell data. On the left hand side a serial calculation is shown performed on one processor (0) which holds the data for all the atoms and for the entire simulation cell volume. On right hand side the same calculation is performed in parallel on four processors (numbered from 0 to 3). Each processor now holds only a cluster of closely-spaced atoms and a slab of the simulation cell volume.

box is not compatible with the PPD subdivision of the simulation cell, and in any case there would be no benefit in distributing its data as it is constructed so that its volume is a small constant multiple of the volume per atom which is independent of the volume of the simulation cell and the total number of atoms.

In order to accommodate these requirements we adopt a two-pronged data parallelisation strategy which is outlined in Fig. 6. We distinguish two types of data: "atomic data" and "simulation cell data" and distribute them to processors in different ways. To the category of atomic data belong quantities that are clearly associated with a particular atom such as the NGWFs in their PPD representation, the pseudopotentials, the atomic coordinates, etc. Each atomic quantity is allocated in its entirety to a processor. This approach simplifies and facilitates amongst other things depositing NGWFs to FFT boxes which are also allocated in their entirety to single processors. To the category of simulation cell data belong quantities which by construction extend over the entire simulation cell, such as the charge density and the Hartree, ionic and exchange-correlation potentials. As can be seen from Fig. 6 the distribution of simulation cell data simply consists of dividing the simulation cell into slabs of equal volume along one lattice vector direction and distributing the psinc grid of each slab to a particular processor. The distribution of atomic data is somewhat more involved. Atoms are allocated to processors so that the number of NGWFs on each processor is approximately equal. Of course there are many possible ways to achieve this which do not necessarily always lead to the balancing of the computational load or communication. In order to ensure that these important balancing requirements are always satisfied, the atoms are distributed to processors in the order in which they are visited by a "space-filling" fractal curve [5]. This approach is effective irrespective of the order in which the atoms are given in the input file or the structure of the material under study and results in the near-optimal allocation to each processor of groups of atoms which are in close proximity in space as is schematically shown in Fig. 6.

5 Communication model & parallel calculations

Using the data parallelisation strategy of the previous section in combination with calls to MPI subroutines we are able to perform in parallel the computations that we described in Section 3. The separation of the tasks of Section 3 into level 1 and level 2 operations has special practical importance for the parallel implementation. During the level 1 stages a processor X performs operations on groups of atomic data that in general can belong to other processors, hence these atomic data will need to be communicated to the processor X. All the NGWFs that overlap with the current NGWF of X need to be considered and the precomputed sparsity pattern of the NGWF overlap matrix is used to predict which NGWFs need to be fetched from other processors to X. Our communication model, which takes advantage of this sparsity, can be seen in Fig. 7. The purpose of this model is to consider all pairs of processors and exchange data between them when needed. As shown in Fig. 7 the model runs through the blocks of the processor-processor overlap matrix. For each such block, the already known sparsity pattern of the NGWF overlap matrix is used to predict and only calculate quantities that are non-zero. The communication model consists of $2N_P - 1$ "super-steps" where N_P is the number of processors. The first super-step involves the diagonal blocks only, which are also the most dense. As all information is on the same processor, only computation takes place without any communication. Each one of the remaining super-steps deals with each of the off-diagonal bands of blocks. This approach allows us to use only point-to-point communication and hence maintain communication requirements which can scale to an arbitrary number of processors. Crucial to this is our parallelisation strategy for atomic data which ensures that the non-zero values of the (by construction) sparse NGWF overlap matrix are clustered around its diagonal. The right hand side of Fig. 7 confirms this with an example of the sparsity pattern of the NGWF overlap matrix from a 3000-atom protein molecule. On the left side of the figure the super-steps are presented in detail. For each block the row-processor sends data and the column-processor receives and computes. In general, during a super-step a processor is involved in sending data, receiving data and computing. To improve performance we use "non-blocking" communication during the send operations. This allows us to interleave communication and computation and avoid the delays involved with the sender processors waiting

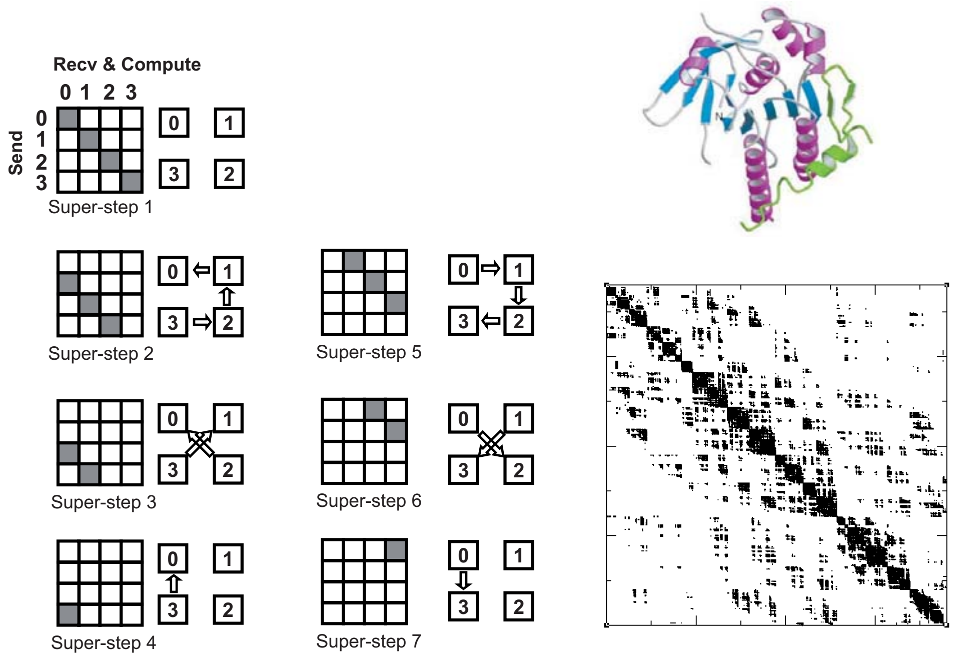


Fig. 7 Example of the ONETEP communication model on four processors (numbered 0–3). On the left the seven super-steps that are performed are shown. These allow point-to-point communication between all pairs of processors. The processor pairs that are active in each super-step are shown as shaded blocks in a matrix of processor-processor blocks and the direction of communication is indicated by arrows next to them. On the right, the actual sparsity pattern of the NGWF overlap matrix of a 3000-atom protein is shown. Only about 1% of the total number of elements are non-zero (shown in black). Most of them are clustered along the diagonal of the matrix which in turn implies that the bulk of the communication and computation will be performed during super-steps which involve diagonal or near-diagonal processor-processor blocks.

for the send operations to complete. Of course, no data is sent when the NGWF in the current send row does not overlap with the NGWF in the current receive column. The NGWFs are always communicated in their compact PPD representation, and as soon as they are received, they are expanded to the form required for the type of calculation to be performed.

At first one may think that once a row-NGWF has been received, a processor would be able to use it in all computations with the local column-NGWFs with which it overlaps. In most cases this is not possible as it would lead to excessively large memory requirements due to the fact that each of the local column-NGWFs needs to be kept its own FFT box. The size of the FFT box in a calculation is determined by the maximum NGWF radius and the psinc kinetic energy cut-off. As an example, in routine calculations an FFT box could contain 75 × 75 × 75 psinc points (double precision numbers) and would occupy about 3.2 MB of memory. To avoid this potential memory bottleneck we keep only a batch N_{batch} of FFT boxes on each processor at any one time so that functions in FFT boxes are generated only when needed, in a fashion reminiscent of direct SCF algorithms [46]. Thus, on top of the supersteps of Fig. 7 there is also a series of "batch-steps" which take place until all the local NGWFs $N_{\text{NGWF}}^{(X)}$ of each processor X have been processed through its batch of FFT boxes.

Let us now look closer into one of the processor-processor blocks of Fig. 7 and examine the events that take place during the super-step loop which is nested inside the batch-step loop. In particular Fig. 8 describes the generation of the $\langle \phi_\alpha | \hat{O} | \phi_\beta \rangle$ matrix elements on processor X. The procedure involves three nested loops: The outermost loop takes place over batches of NGWFs until all $N_{\text{NGWF}}^{(X)}$ NGWFs of X and

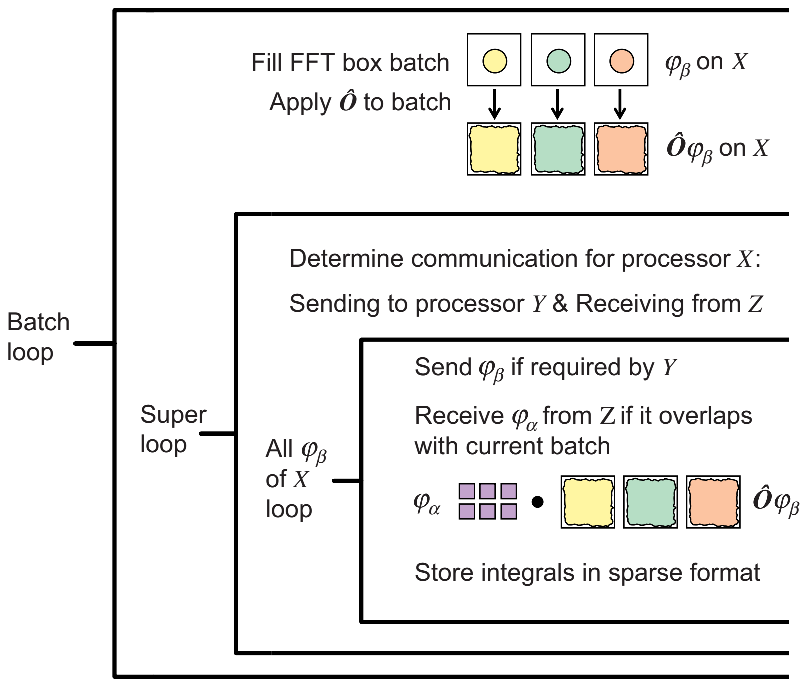

Fill FFT box batch
Apply $\hat{O}$ to batch
φ_β on X
$\hat{O}\varphi_\beta$ on X
Determine communication for processor X:
Sending to processor Y & Receiving from Z
Batch loop
Super loop
All φ_β of X loop
Send φ_β if required by Y
Receive φ_α from Z if it overlaps with current batch
φ_α
$\hat{O}\varphi_\beta$
Store integrals in sparse format

Fig. 8 Parallel computation of $\langle\phi_\alpha|\hat{O}|\phi_\beta\rangle$ integrals. Non-zero elements in the columns of the $\langle\phi_\alpha|\hat{O}|\phi_\beta\rangle$ matrix that correspond to NGWFs belonging to processor X are computed using the communication model of Fig. 7. In the inner-most loop the integrals are generated as in Fig. 3 for the serial calculation the only difference now being the fact that processor X holds a batch of FFT boxes (three in this example) with $\hat{O}\phi_\beta(\boldsymbol{r})$ functions at any one time.

all other processors are exhausted. The middle loop is the super-step loop over processor-processor blocks of Fig. 7 during which the direction of point-to-point communication for each processor is decided. In our example processor X is only allowed to send data to processor Y and receive data from processor Z. The innermost loop involves the actual operations of communication and computation. Each local NGWF of X is sent to Y if needed and an NGWF from the rows of the current Z–X processor-procesor block is received from Z if it is predicted to produce a non-zero $\langle\phi_\alpha|\hat{O}|\phi_\beta\rangle$ integral with any of the $\hat{O}\phi_\beta(\boldsymbol{r})$ functions in the current batch of X (which have been already generated during the batch step). In the example of Fig. 8 the size of the batch is 3. In practice, ONETEP contains the "expert-level" input parameters *locpot_int_batchsize*, *kinetic_int_batchsize*, *density_batchsize* and *ngwf_grad_batchsize* with which the user can set the batch sizes for the calculation of the local potential integrals, kinetic energy integrals, charge density and NGWF gradient respectively. The default value for these variables, which was used to produce all results reported in this paper, is 10.

Apart from the issue of communication between atomic data that we have examined so far in this section, the need for communication of simulation cell data between processors also arises. For example, the calculation of the Hartree potential is performed by solving the Poisson equation over the entire simulation cell subject to periodic boundary conditions. The charge density, which is generated as described in Subsection 3.2, is Fourier transformed to momentum space using an algorithm for fast Fourier transforms on parallel computers which is compatible with our parallelisation strategy for simulation cell data (Fig. 6) and can scale to an arbitrary number of processors. The Hartree potential is then obtained in momentum space by multiplication of the charge density by the momentum representation of $1/r$ as is common in plane wave codes [9]. It is then transferred to real space by an inverse fast Fourier transform. A closely related procedure is followed for the calculation of derivatives of the charge density which are

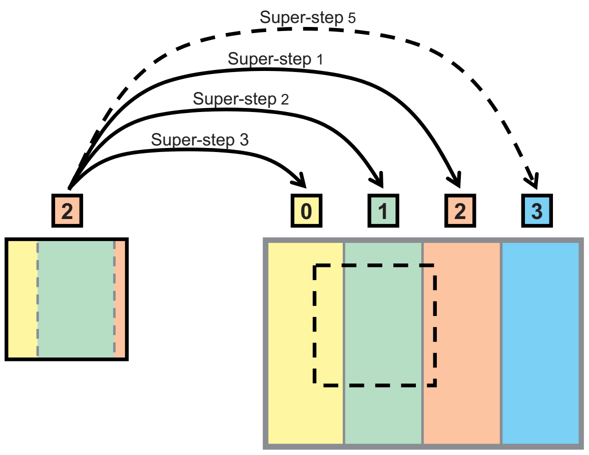


Fig. 9 Depositing an FFT box to the simulation cell during a parallel calculation. In this example on four processors (labelled 0–3) the contents of the FFT box of processor 2 are distributed to slabs of the simulation cell belonging to processors 0, 1, and 2. The supersteps of Fig. 7 during which this data transfer takes place are noted. For clarity the FFT boxes of the remaining processors which are also concurrently depositing data to the simulation cell are not shown.

required for the generation of the exchange-correlation potentials and energy [47] of the various Generalised Gradient Approximation (GGA) functionals implemented in ONETEP.

Finally, there is communication between atomic data and simulation cell data. The need for communication between the two data types arises when the contents of an FFT box, which is treated as atomic data and is located on one processor, need to be deposited to the simulation cell whose volume is divided into slabs distributed to different processors as shown in Fig. 6. This is exactly the state of affairs in the level 2 part of the calculation of the charge density of Fig. 4. The number of such FFT box to simulation cell transfers that a processor X needs to perform during the calculation of the charge density is equal to its local number of NGWFs, $N_{\text{NGWF}}^{(X)}$. For each such transfer a sequence of processor-processor communication steps analogous to those of Fig. 7 takes place. Of course now there is no batch loop or NGWF loop, so only the middle loop of Fig. 8 remains. A graphic illustration of this process is given in Fig. 9 which shows an example of an FFT box belonging to processor 2 and the communication super-steps during which its contents are deposited to the simulation cell slabs of different processors. Concurrently with this, all other processors are also depositing the contents of their FFT boxes to the simulation cell, but this is not depicted in Fig. 9 for the sake of clarity. Again here communication happens only when it is needed: in the example processor 2 does not send any data to processor 3 since its FFT box does not contain any part of the simulation cell slab of processor 3. Also, there is obviously no communication when processor 2 deposits to its own slab (diagonal blocks in the scheme of Fig. 7). The opposite procedure is followed when extracting quantities from the simulation cell to the FFT box. Such an example is the extraction of the Hartree potential in order to construct the Hamiltonian operator and evaluate its action on a function in an FFT box as is necessary in the calculation of the Hamiltonian matrix elements (Fig. 3) and the NGWF gradient (Fig. 5).

6 Results and discussion

The parallel implementation of ONETEP is completely general and should be able to scale to an arbitrary number of processors on a variety of parallel platforms. In practice we need to have more than one atom per processor for the communication not to dominate the total computational time. However we have observed that only 10 atoms per processor is already enough for good parallel scaling in most cases. In Fig. 10 we report the speed-up that we achieve in the total time taken for one self-consistent NGWF iteration (Eq. (3)) as a function of increasing numbers of processors. We show calculations run on 4, 8, 16, 32, 48 and 64 processors. We focus here on two examples, an 800-atom chiral boron nitride nanotube and a 1403-atom polyglycine peptide in a globular conformation. We observe that both curves show an almost linear speed-up up to 64 processors. The speed-ups we achieve remain substantial even when we get to 64 processors with 79% of the ideal value for the case of the nanotube and 72% in the case of the

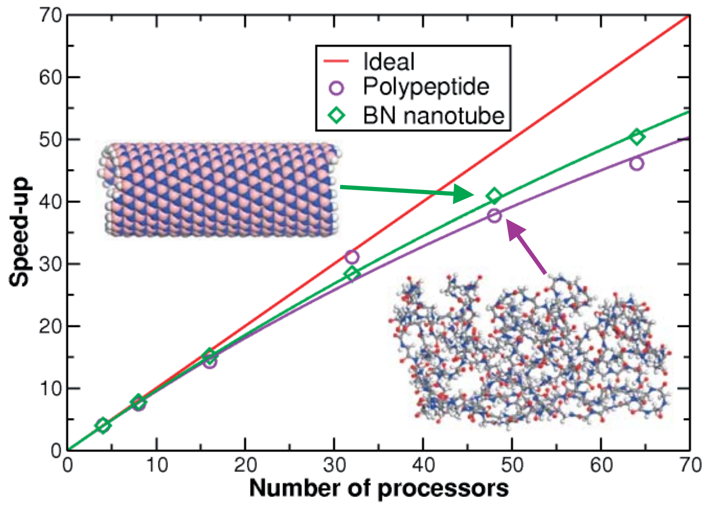
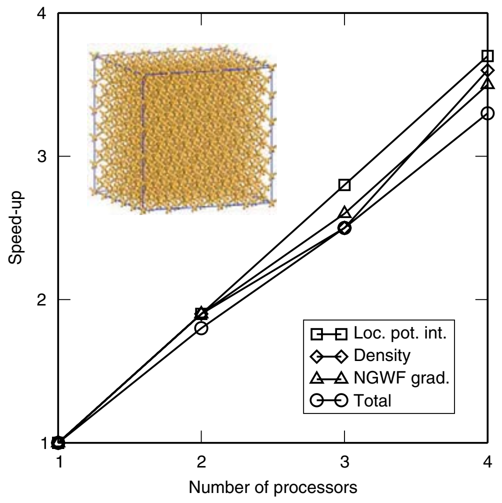


Fig. 10 Speed-up in the total wall-clock time taken for one NGWF iteration. These calculations were run on a Sun Fire F15K parallel computer.

polyglycine. The very regular structure of the nanotube leads to an ideal partitioning of the atomic data by our parallelisation strategy to achieve a near-optimal balance of the computation and communication load, hence the parallel speedup in this case is greatest. The irregular three-dimensional structure of the polyglycine is a challenging test for our parallel algorithms. While these irregularities are obvious in Fig. 10 it is particularly pleasing to note that the speed-ups remain high throughout its range from which we can conclude that the distribution of atoms to processors and hence the balancing of tasks is still done in a satisfactory way.

Let us now look at some further tests which examine the parallel scaling of each of the most computationally intensive components in a total energy calculation with ONETEP. We show in Fig. 11 results for a 1000-atom block of perfect crystalline silicon and a 988-atom protein generated on a single SUN V40z server with four AMD OPTERON processors with the same calculation parameters as we have used for these systems for our plane wave accuracy tests [8]. The speed-up as a function of the number of processors is given for the wall-clock time taken for the calculation of the local potential integrals according to Subsection 3.1, the electronic charge density according to Subsection 3.2, the NGWF gradient according to Subsection 3.3 and the total wall-clock time for one NGWF iteration. We can observe that the speed-ups we achieve increase essentially linearly with the number of processors and are very satisfactory for all the calculated quantities as on all four processors they lie in the region of 75–93% of the ideal speed-up. Again here we have one material with a perfectly ordered structure such as the silicon and one material with a completely random structure such as the protein and we observe that we get good speed-ups in both cases, as a further confirmation of the generality of our parallel algorithms.

A point which we have not discussed so far concerns the handling of matrices. We have developed our own special package for storing only the non-zero values of the sparse matrices encountered in ONETEP and performing algebraic operations with them. This package takes advantage of the atom-blocked structure of these matrices to achieve efficiency both in storage and in operations such as sparse matrix multiplication, which are performed in parallel. While operations with our sparse-stored matrices are parallelised, their storage is currently not parallelised and thus every processor holds in its memory an entire copy of each matrix. This is not an obstacle to parallel performance as we have seen from the results of this section but it does eventually limit the size of the molecules we can study. This limit is machine-dependent as it is determined by the memory available per processor. Nevertheless calculations with a few thousand atoms are well within the reach of the current version of ONETEP. As an example, the size of the matrix of Fig. 7 is 7600 × 7600 yet the memory it occupies stored in sparse format is only 4.4 MB.



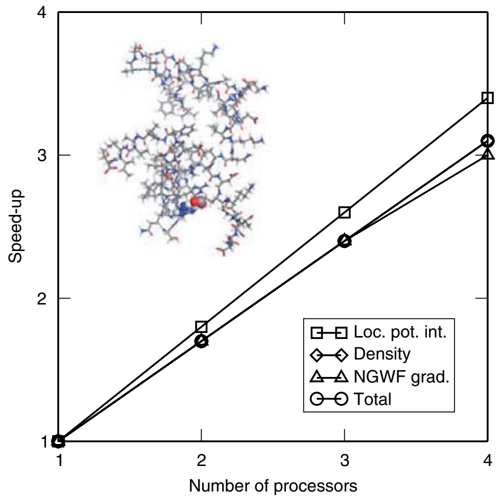


Fig. 11 Parallel speed-ups per one NGWF iteration on a SUN V40z server with four AMD OPTERON processors for a 1000-atom block of crystalline silicon (left) and a 988-atom protein (right). The speed-up as a function of the number of processors is given for the generation of the local potential integrals according to Subsection 3.1, the electronic charge density according to Subsection 3.2, the NGWF gradient according to Subsection 3.3 and the total wall-clock time taken for one NGWF iteration.

This rather modest amount of memory is possible because we always use only a minimal number of NGWFs per atom (i.e. one NGWF per hydrogen atom and four NGWFs per each second row atom such as carbon, oxygen, etc.). In contrast, even routine calculations with localised atomic orbitals basis set codes would need to use at least a "Double Zeta plus Polarisation (DZP)" basis in which case the size of this matrix would be 27500×27500 and it would take up 58 MB of memory. This results in a significant operational benefit in the ONETEP approach as the storage and linear algebra operation cost of matrices is kept as small as possible and the weight of the calculation is shifted to the NGWFs which are more straightforward to parallelise compared to the sparse matrices. The matrix memory benefit is of course partially offset by the larger memory requirements for storing the NGWFs as a psinc expansion (typically about 0.1 MB per NGWF) compared to the near-zero memory requirements for storing atomic orbitals in an atomic orbital code.

In the near future we intend to extend our sparse matrix storage and algebra package to a data-parallel model, effectively adding to the data parallelisation strategy of Fig. 6 a third branch corresponding to "matrix data" which will be combined with the implementation of spin polarisation. At this stage all the types of data in the code will be parallelised and this should enable ONETEP to perform accurate first principles calculations on tens of thousands of atoms.

7 Conclusions

We have presented our implementation for parallel computers of a linear-scaling plane wave density functional theory method. Our method which is implemented in the ONETEP program takes advantage of the decay properties of the single-particle density matrix to perform calculations where the computating cost increases only linearly with the number of atoms. The computational machinery behind this approach is novel as it makes use of the psinc basis set. We have presented the algorithms we have developed for the computation of the total energy and its optimisation with this basis set. In addition to these algorithms we have also developed a strategy for distributing to processors in a balanced manner the data and the computational load. By combining these algorithms with calls to the subroutines of the Message Passing Interface (MPI) library we are able to perform calculations which are portable to a wide range of parallel computer architectures. Tests on a variety of materials demonstrate the high parallel efficiency of ONETEP.

Acknowledgements C.-K. S. would like to thank the Royal Society for a University Research Fellowship. P. D. H. would like to thank Sidney Sussex College Cambridge for a Senior Research Fellowship. A. A. M. would like to thank Christ's College Cambridge for a Research Fellowship and the Royal Society for an equipment grant. The computing resources of the Cambridge-Cranfield High Performance Computing Facility have been essential for this work.

References

[1] W. Kohn and L. J. Sham, Self-consistent equations including exchange and correlation effects, Phys. Rev. **140**, A1133–A1138 (1965).
[2] S. Goedecker, Linear scaling electronic structure methods, Rev. Mod. Phys. **71**(4), 1085–1123 (1999).
[3] D. R. Bowler, T. Miyazaki, and M. J. Gillan, Recent progress in linear scaling ab initio electronic structure techniques, J. Phys.: Condens. Matter **14**, 2781–2798 (2002).
[4] J. M. Soler, E. Artacho, J. D. Gale, A. García, J. Junquera, P. Ordejón, and D. Sánchez-Portal, The SIESTA method for *ab initio* order-N materials simulation, J. Phys.: Condens. Matter **14**, 2745–2779 (2002).
[5] M. Challacombe, A general parallel sparse-blocked matrix multiply for linear scaling scf theory, Comput. Phys. Commun. **128**, 93–107 (2000).
[6] L. Seijo and Z. Barandiarán, Parallel, linear-scaling building-block and embedding method based on localized orbitals and orbital-specific basis sets, J. Chem. Phys. **121**(14), 6698–6709 (2004).
[7] C.-K. Skylaris, P. D. Haynes, A. A. Mostofi, and M. C. Payne, Introducing ONETEP: Linear-scaling density functional simulations on parallel computers, J. Chem. Phys. **122**, 084119 (2005).
[8] C.-K. Skylaris, P. D. Haynes, A. A. Mostofi, and M. C. Payne, Using ONETEP for accurate and efficient $\mathcal{O}(N)$ density functional calcualations, J. Phys.: Condens. Matter **17**, 5757–5769 (2005).
[9] M. C. Payne, M. P. Teter, D. C. Allan, T. A. Arias, and J. D. Joannopoulos, Iterative minimisation techniques for ab initio total-energy calculations: molecular dynamics and conjugate gradients, Rev. Mod. Phys. **64**(2), 1045–1097 (1992).
[10] S. J. Clark, M. D. Segall, C. J. Pickard, P. J. Hasnip, M. I. J. Probert, K. Refson, and M. C. Payne, First principles methods using CASTEP, Z. Kristallogr. **220**, 567–570 (2005).
[11] A. Willetts, L. Gagliardi, A. G. Ioannou, A. M. Simper, C.-K. Skylaris, S. Spencer, and N. C. Handy, MAGIC: An integrated computational environment for the modelling of heavy atom chemistry, Int. Rev. Phys. Chem. **19**(3), 327 (2000).
[12] Message Passing Interface Forum, http://www.mpi-forum.org/.
[13] P. Pacheco, Parallel Programming with MPI (Morgan Kaufmann, San Fransisco, CA, 1996).
[14] E. Hernández and M. J. Gillan, Self-consistent first-principles technique with linear scaling, Phys. Rev. B **51**(15), 10157–10160 (1995).
[15] C. M. Goringe, E. Hernández, M. J. Gillan, and I. J. Bush, Linear-scaling DFT-pseudopotential calculations on parallel computers, Comput. Phys. Commun. **102**, 1–16 (1997).
[16] P. D. Haynes and M. C. Payne, Localised spherical-wave basis set for $\mathcal{O}(N)$ total-energy pseudopotential calculations, Comput. Phys. Commun. **102**, 17–32 (1997).
[17] J.-L. Fattebert and J. Bernholc, Towards grid-based $\mathcal{O}(N)$ density-functional theory methods: Optimized nonorthogonal orbitals and multigrid acceleration, Phys. Rev. B **62**(3), 1713–1722 (2000).
[18] J.-L. Fattebert and F. Gygi, Linear scaling first-principles molecular dynamics with controlled accuracy, Comput. Phys. Commun. **162**, 24–36 (2004).
[19] J. E. Pask and P. A. Sterne, Real-space formulation of the electrostatic potential and total energy of solids, Phys. Rev. B **71**, 113101 (2005).
[20] C.-K. Skylaris, A. A. Mostofi, P. D. Haynes, O. Diéguez, and M. C. Payne, Nonorthogonal generalised Wannier function pseudopotential plane-wave method, Phys. Rev. B **66**, 035119 (2002).
[21] R. McWeeny, Some recent advances in density matrix theory, Rev. Mod. Phys. **32**(2), 335–369 (1960).
[22] W. Kohn, Analytic properties of Bloch waves and Wannier functions, Phys. Rev. **115**, 809 (1959).
[23] J. D. Cloizeaux, Energy bands and projection operators in a crystal: Analytic and asymptotic properties, Phys. Rev. **135**(3A), A685–A697 (1964).

[24] R. Baer and M. Head-Gordon, Sparsity of the density matrix in Kohn–Sham density functional theory and an assessment of linear system-size scaling methods, Phys. Rev. Lett. **79**(20), 3962–3965 (1997).

[25] S. Ismail-Beigi and T. A. Arias, Locality of the density matrix in metals, semiconductors and insulators, Phys. Rev. Lett. **82**(10), 2127–2130 (1999).

[26] L. He and D. Vanderbilt, Exponential decay properties of Wannier functions and related quantities, Phys. Rev. Lett. **86**(3), 5341–5344 (2001).

[27] D. R. Bowler, I. J. Bush, and M. J. Gillan, Practical methods or ab initio calculations on thousands of atoms, Int. J. Quantum Chem. **77**, 831–842 (2000).

[28] X. P. Li, R. W. Nunes, and D. Vanderbilt, Density-matrix electronic-structure method with linear system-size scaling, Phys. Rev. B **47**(16), 10891–10894 (1993).

[29] A. H. R. Palser and D. E. Manolopoulos, Canonical purification of the density matrix in electronic-structure theory, Phys. Rev. B **58**(19), 12704–12711 (1998).

[30] N. Marzari, D. Vanderbilt, and M. C. Payne, Ensemble density-functional theory for ab initio molecular dynamics of metals and finite-temperature insulators, Phys. Rev. Lett. **79**(7), 1337–1340 (1997).

[31] J. M. Millam and G. E. Scuseria, Linear scaling conjugate gradient density matrix search as an alternative to diagonalisation for first principles electronic structure calculations, J. Chem. Phys. **106**(13), 5569–5577 (1997).

[32] P. D. Haynes and M. C. Payne, Corrected penalty-functional method for linear-scaling calculations within density-functional theory, Phys. Rev. B **59**(19), 12173–12176 (1999).

[33] D. Baye and P.-H. Heenen, Generalised meshes for quantum mechanical problems, J. Phys. A, Math. Gen. **19**, 2041–2059 (1986).

[34] A. A. Mostofi, P. D. Haynes, C.-K. Skylaris, and M. C. Payne, Preconditioned iterative minimisation for linear-scaling electronic structure calculations, J. Chem. Phys. **119**(17), 8842 (2003).

[35] C.-K. Skylaris, A. A. Mostofi, P. D. Haynes, C. J. Pickard, and M. C. Payne, Accurate kinetic energy evaluation in electronic structure calculations with localized functions on real space grids, Comput. Phys. Commun. **140**, 315–322 (2001).

[36] A. A. Mostofi, C.-K. Skylaris, P. D. Haynes, and M. C. Payne, Total energy calculations on a real space grid with localized functions and a plane-wave basis, Comput. Phys. Commun. **147**, 788–802 (2002).

[37] E. L. Briggs, D. J. Sullivan, and J. Bernholc, Real-space multigrid-based approach to large-scale electronic structure calculations, Phys. Rev. B **54**(20), 14362–14375 (1996).

[38] T. L. Beck, Real-space mesh techniques in density-functional theory, Rev. Mod. Phys. **72**(4), 1041–1080 (2000).

[39] M. E. Tuckerman, D. A. Yarne, S. O. Samuelson, A. L. Hughes, and G. J. Martyna, Exploiting multiple levels of parallelism in molecular dynamics based calculations via modern techniques and software paradigms on distributed memory computers, Comput. Phys. Commun. **128**, 333–376 (2000).

[40] T. Torsti, M. Heiskanen, M. J. Puska, and R. M. Nieminen, MIKA: Multigrid-based program package for electronic structure calculations, Int. J. Quantum Chem. **91**, 171–176 (2003).

[41] M. M. G. Alemany, M. Jain, L. Kronik, and J. R. Chelikowsky, Real-space pseudopotential method for computing the electronic properties of periodic systems, Phys. Rev. B **69**, 075101 (2004).

[42] R. Schmid, Car-Parrinello simulations with a real space method, J. Comput. Chem. **25**(6), 799–812 (2004).

[43] J. J. Mortensen, L. B. Hansen, and K. W. Jacobsen, Real-space grid implementation of the projector augmented wave method, Phys. Rev. B **71**, 035109 (2005).

[44] E. Artacho and L. Miláns del Bosch, Nonorthogonal basis sets in quantum mechanics: Representations and second quantization, Phys. Rev. A **43**(11), 5770–5777 (1991).

[45] E. Hernández, M. J. Gillan, and C. M. Goringe, Linear-scaling density-functional-theory technique: The density-matrix approach, Phys. Rev. B **53**(11), 7147–7157 (1996).

[46] J. Almlöf, K. Korsel, and K. Faegri, Jr., Principles for a Direct SCF approach to LCAO-MO Ab-Initio Calculations, J. Comput. Chem. **3**, 385 (1982).

[47] J. A. White and D. M. Bird, Implementation of gradient-corrected exchange-correlation potentials in Car-Parrinello total-energy calculations, Phys. Rev. B **50**(7), 4954–4957 (1994).