Computer Physics Communications 183 (2012) 2581–2588

Contents lists available at SciVerse ScienceDirect

Computer Physics Communications

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

Real-space density functional theory and time dependent density functional theory using finite/infinite element methods

Alejandro Soba *, Edgar Alejandro Bea, Guillaume Houzeaux [1], Hadrien Calmet, José María Cela

Computer Applications in Science & Engineering, Barcelona Supercomputing Center, c/Jordi Girona, 31, E-08034 Barcelona, Spain

ARTICLE INFO

Article history:
Received 23 June 2011
Received in revised form
3 July 2012
Accepted 10 July 2012
Available online 17 July 2012

Keywords:
DFT
TD-DFT
Finite element method
Infinite element method
Electronic *ab initio* method

ABSTRACT

We present a numerical approach using the finite element method to discretize the equations that allow getting a first-principles description of multi-electronic systems within DFT and TD-DFT formalisms. A strictly local polynomial function basis set is used in order to represent the entire real-space domain. Infinite elements are introduced to model the infinite external boundaries in the case of Hartree's equation. The diagonal mass matrix is obtained using a close integration rule, reducing the generalized eigenvalue problem to a standard one. This framework of electronic structure calculation is embedded in a high performance computing environment with a very good parallel behavior.

1. Introduction

There exist several numerical approaches to density functional theory (DFT) based on real-space discretization of the Kohn–Sham equations that result in very useful tools for first-principles electronic structure calculations. The basic idea of all those treatments is either the use of a strictly local function basis set to discretize the real space, or the direct use of some kind of discretization for the differential equations, or a hybrid combination of both. For example, the SIESTA method [1,2] uses a numerical linear combination of atomic orbitals (LCAO) basis set and a uniform regular real-space grid to calculate potentials and matrix elements. The basis functions and the electron density are projected on a real-space grid, in order to calculate the Hartree and exchange–correlation potentials and matrix elements. The density gradient operator is discretized and evaluated by using the finite difference method. Poisson's equation is typically solved using Fast Fourier Transforms (FFT) on a periodic unit cell by a supercell construction. As an alternative, a finite difference scheme can also be used on a Cartesian grid together with fixed boundary conditions, obtained from the multipole expansion of the charge density. This can be done in $\mathcal{O}(N)$ operations, in contrast with the FFT that scales as $\mathcal{O}(N \log N)$. The size of the grid is determined by the number of plane waves which can be represented without aliasing. No explicit orthogonalization of the LCAO basis set is needed. In its place the minimization of a modified energy functional that produces orthogonal wave functions is used. Due to the sparsity of the Hamiltonian and overlap matrices minimal basis sets are needed and order-N methods (algorithms) are used with a number of operations that scale linearly with the size of the system, i.e. $\mathcal{O}(N)$ scaling.

As another alternative, the finite difference method is used to solve *ab initio* electronic problems [2–4]. For example, OCTOPUS [3] provides a very useful tool to approach both stationary and time-dependent problems: all the quantities are represented on a uniform real-space grid and almost all equations are fully solved on it, except for Poisson's equation for the Hartree potential, which is treated using FFT (as in SIESTA). This program is well suited for solving periodic and confined electronic quantum systems with time-dependent classical electromagnetic fields. In order to start the self-consistent cycle (SCF), an initial trial wave function can be generated from a linear combination of atomic pseudo-orbital basis set. In this approach, the sparsity of the Hamiltonian matrix is determined by the finite difference stencil for the kinetic energy and by the non-local projectors of the pseudo-potentials. In the time-dependent DFT formalism, the maximum time step of the wave function propagation in real-time is limited by the grid spacing. Another tool that doesn't use an explicit basis set to solve the Kohn–Sham equation in real space is PARSEC (Pseudopotential Algorithm for Real-Space Electronic Calculations) [5,6]. This tool handles a large variety of boundary conditions and reaches a great accuracy and parallel efficiency in the SCF using a non-linear subspace filtering method. OCTOPUS and PARSEC, were

* Corresponding author.
E-mail addresses: soba@cnea.gov.ar (A. Soba), guillaume.houzeaux@bsc.es (G. Houzeaux).
[1] Tel.: +34 934054291.

0010-4655/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.cpc.2012.07.007

both designed to reach an efficient massive parallelization, making them highly effective for the treatment of very large systems.

In the real-space approach the use of a regular mesh would be desirable for computational considerations because it produces a sparse, local, and highly structured Hamiltonian matrix, which enables the effective use of iterative numerical methods, very convenient from the point of view of parallel computation. However, to obtain accurate solutions, mesh refinements in the vicinity of the nuclei positions are required in many cases, and non-regular meshes are also necessary. Another possibility is the use of adaptive curvilinear coordinates [7,8]. This approximation improves the efficiency without losing the computational advantages of a regular mesh, by a change of coordinates that condenses mesh grid points in the vicinity of a nucleus to an adaptive mesh. Anyway, the use of the mesh refinement and the change of coordinates both increase the complexity of the calculus if pseudo-potentials are used.

Several approaches based on the finite element method (FEM) have been reported, in all-electron [9–11] and pseudo-potential approximations [12,13] as well. These works show the adequacy of the FEM to the requirements of the electronic structure calculations. Among other advantages the FEM gives the possibility of using non-uniform and unstructured meshes [11] near the atomic positions, i.e. the possibility to refine the mesh [12], without the need of making coordinate changes. Besides, it will be shown below that there is a great flexibility in the selection of the boundary conditions. Finally, the efficiency of the parallelization via the mesh partition technique is very high. By using METIS [14], as we did in our implementation, the partition of the domain is well-balanced by the number of degrees of freedom, without losing the advantages of the unstructured meshes in parallel environments.

A general numerical approach based on FEM to TD-DFT and DFT equations is proposed in this paper, focusing on some points related to the resolution of difficulties not treated in the literature (to the author's knowledge). In Section 2 we expose the use of the FEM to solve the DFT equations with particular emphasis on the boundary condition treatments. Section 3 presents some convergence studies, the eigen solver and the parallelization scheme of the self-consistency cycle. Section 4 is devoted to a single and preliminary numerical approach to TD-DFT equations and implementation details are described. Finally, in the last section, we make a summary and list some conclusions of this work.

2. The stationary Kohn–Sham equation's treatment

In density functional theory, the total energy of an electronic system is expressed as a functional of the charge density as,

$$E\left[\rho(\boldsymbol{r})\right] = T_s\left[\rho(\boldsymbol{r})\right] + \int d\boldsymbol{r}\, V_{eff}\left(\left[\rho\right];\boldsymbol{r}\right)\rho(\boldsymbol{r}) \tag{1}$$

where T_s is the kinetic energy of a noninteracting electronic system under an effective external potential given by

$$V_{eff}\left(\left[\rho\right];\boldsymbol{r}\right) = V_{ion}(\boldsymbol{r}) + V_{hartree}\left(\left[\rho\right];\boldsymbol{r}\right) + V_{xc}\left(\left[\rho\right];\boldsymbol{r}\right) \tag{2}$$

this being the sum of ionic (Coulombian nuclear or pseudo-ionic), Hartree and exchange–correlation potentials, respectively. The effective potential depends on the density.

In this framework, also named stationary DFT, the ground state is a Slater determinant of orbitals ψ_k^{KS} obtained by minimizing the total energy by the self-consistent solution of Kohn–Sham equations [15],

$$\left\{-\frac{1}{2}\nabla^2 + V_{eff}(\boldsymbol{r})\right\}\psi_k(\boldsymbol{r}) = \epsilon_k\psi_k(r) \tag{3}$$

and the electronic density is built according to

$$\rho(\boldsymbol{r}) = \sum_{k=1}^{nodes} Nocc_k\,|\psi_k(\boldsymbol{r})|^2 \tag{4}$$

i.e., using the wave functions of each Kohn–Sham state k with occupation number $Nocc_k$.

To solve the set of Eqs. (1)–(4), all involved quantities are represented in the real cartesian space, discretized on a finite element mesh. The Kohn–Sham orbitals ψ_k^{KS} are approximated using local basis functions ϕ_j such that

$$\psi(\boldsymbol{r}) = \sum_{j=1}^{nodes}\phi_j(\boldsymbol{r})\varphi_j \tag{5}$$

where φ_j are the nodal unknowns of the problem. In the FEM, the basis functions ϕ_j are defined only within individual elements and have continuity with the neighboring elements. The number of nodes per element and their relative internal positions depend on the family of elements selected. In this work, we use three-dimensional hexahedral with interpolation functions of 27 nodes (cubic) or of 64 nodes (biquadratic) per element. After applying the discretization to Eq. (3), we end up with a sparse system for the nodal unknowns [16,17], treatable as a generalized eigenvalue problem over the entire domain, of the form:

$$\mathbf{H}\boldsymbol{\phi} = \epsilon\mathbf{M}\boldsymbol{\phi} \tag{6}$$

where the coefficients H_{ij} of the Hamiltonian matrix $\mathbf{H}$, and the coefficients M_{ij} of the mass matrix $\mathbf{M}$, are computed as:

$$H_{ij} = \frac{1}{2}\int_\Omega \nabla\varphi_i\nabla\varphi_j d\Omega + \int_\Omega \varphi_i V_{eff}\varphi_j d\Omega, \qquad M_{ij} = \int_\Omega \varphi_i\varphi_j d\Omega. \tag{7}$$

The different steps to build the Hamiltonian matrix and the boundary conditions of the problem will be given below.

2.1. Hartree potential

As usual in DFT, the Hartree potential operator is represented through the solution of the Poisson equation for a charge distribution $\rho(\boldsymbol{r})$,

$$-\nabla^2 V_{hartree} = 4\pi\,\rho(\boldsymbol{r}). \tag{8}$$

The discretization of this equation is similar to the kinetic energy operator shown in the first member of Eq. (7). After integration by parts, we end up with a set of linear equations for the nodal unknowns

$$\mathbf{L}\mathbf{v}_{hartree} = \mathbf{d} \tag{9}$$

where the coefficients of the Laplacian operator $\mathbf{L}$ and right-hand side $\mathbf{d}$ are computed as

$$L_{ij} = \int_\Omega \nabla\varphi_i\nabla\varphi_j d\Omega, \qquad d_i = 4\pi\int_\Omega \rho(\boldsymbol{r})\varphi_i d\Omega. \tag{10}$$

We must provide to these sets of equations some boundary conditions. We implement two options to do that: the first possibility consists in calculating the value of the potential in the boundary of a bounded simulation domain by making use of the multipole expansion of the charge density inside it. The potential for points in the boundary's domain is obtained from:

$$V_{hartree}\left(\partial\Omega\right) = \sum_{l=0}^{l_{max}}\sum_{m=-l}^{l}\frac{4\pi}{2l+1}\frac{1}{r^{l+1}}Y_{lm}\left(\partial\Omega\right)Q_{lm} \tag{11}$$

$$Q_{lm} = \int_\Omega r^l Y_{lm}(r)\rho(r)d\Omega \tag{12}$$

where Q_{lm} is the m-component of the l-pole moment (usually monopole, dipole, up to quadrupole is enough), and Y_{lm} are the respective spherical harmonics.

In many problems of quantum mechanics, we face unbounded domains where several relevant physical magnitudes decay to zero at infinity. For magnitudes in where the decay towards the domain boundary is exponential, the use of a relatively big domain and finite elements of a different size provides adequate solutions and great accuracy by discretizing with small elements in the main region and large elements far away. However, in the case of the Poisson equation, for example, where the solutions can decay as $1/r$, this approach is not the best one. In the last two decades, the infinite element method has emerged as an alternative solution technique [18–20] and we implement this option to model the far-field region in Eq. (8). The main advantage of the use of this kind of element is that the solution of the Poisson equation is numerically exact, i.e. the representation by infinite elements on the external part of the domain implies a mapping from boundary nodes to infinity. Thus, the computation of the multipole integrals is not necessary. A mesh with infinite elements on all the external boundaries is used to solve the Hartree contributions in Eq. (8), while a mesh with finite elements is considered for the problem defined in Eq. (7).

In a parallel computing environment an integral involves a global communication event, so that their reduction results a great advantage from the computational point of view. That is why the use of infinite elements instead of Eqs. (11)–(12) is a good option. Here, we use the original formulation proposed by Bettes [21] for the infinite elements. There, the domain is extended to the infinite by substituting the classical interpolation function with an alternative set, which maps the finite domain into the infinite one. A detailed list of these functions for numerous types of elements is shown in [21]. Taking the Lagrange family of elements, in the general three dimensional box domain, we can have three types of infinite elements depending on the number of infinite directions: one (in a face), two (in the edges) or three (in the vertices). In Table 1 we list the general mapping functions for the single case of just one direction. These functions can be used to build cases of more dimensions, by direct product [17]. A great advantage in the use of these mapping functions is that a Gauss integration rule could be used as in the finite case. Just for clarification, taking the case of one infinite direction, the elements have one infinite coordinate (x) (in practice x is very large), an application for the finite domain $-1 \leq \xi \leq 1$ onto one semi-infinite $x_1 \leq x \leq \infty$ is defined, using a mapping function $\xi = f(x)$. Then

$$x = \sum_{i=1}^{nodes} Z_i x_i \tag{13}$$

with the introduction of this change into the integral equation, the derivatives are included into the Jacobian matrix

$$\frac{\partial x}{\partial \xi} = \sum_{i=1}^{nodes} \frac{\partial Z_i}{\partial \xi} x_i. \tag{14}$$

A common mapping is $\xi = 1 - 1/(x - x_0)x_0$ being the pole of the element (i.e. the origin of the decay). If the infinite direction is the opposite semi-axis, the mapping needs to be redefined taking into account this sign change. For the case of more than one dimension, the shape functions in the infinite direction are multiplied by the one-dimensional functions in the finite direction as usual. The mapping functions for the nodes in the "infinite" point are not constructed. Eqs. (15) and (16) show the schematic application of infinite elements into Eq. (8). As explained previously, the infinite shape functions appear only in the elemental Jacobian matrix calculation. For a hexahedron, the contribution $L_{ij}^{(e)}$ of an element (e) to L_{ij} is

$$L_{ij}^{(e)} = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{\infty} \nabla\varphi_i \nabla\varphi_j J^{-1} d\Omega \tag{15}$$

Table 1
Parents (φ_k) and infinite (Z_k) shape functions for one dimension and positive infinite position $\xi = 1.0$.

Node, k	ξ_k	φ_k	Z_k
1	-1	$\frac{-9\xi^3+9\xi^2+\xi-1}{16}$	$\frac{9\xi^2-1}{4(1-\xi)}$
2	$-1/3$	$\frac{27\xi^3-9\xi^2-27\xi+9}{16}$	$-\frac{(\xi+1)(\xi-1)}{1-\xi}$
3	$1/3$	$\frac{-27\xi^3-9\xi^2+27\xi+9}{16}$	$\frac{(\xi+1)(3\xi+1)}{4(1-\xi)}$
4	1	$\frac{9\xi^3+9\xi^2-\xi-1}{16}$	–

with

$$J = \begin{vmatrix} \frac{\partial Z_i}{\partial \xi} & \frac{\partial Z_j}{\partial \xi} & \frac{\partial Z_k}{\partial \xi} \\ \frac{\partial Z_i}{\partial \eta} & \frac{\partial Z_j}{\partial \eta} & \frac{\partial Z_k}{\partial \eta} \\ \frac{\partial Z_i}{\partial \zeta} & \frac{\partial Z_j}{\partial \zeta} & \frac{\partial Z_k}{\partial \zeta} \end{vmatrix}. \tag{16}$$

The main drawback of the infinite elements used is the need for creating a double mesh: an internal mesh entirely formed with finite elements dedicated to solve the Kohn–Sham equation, and a second mesh, adding to the first one an external shell built with infinite elements to solve the Poisson equation. This double mesh, in order to use the same numbering for the internal mesh in the SCF cycle and for the Poisson domain, needs to be numbered in a non-correlative way. But while the bandwidth of the internal one maintains a good behavior as usual, there is an increase of the bandwidth in the external mesh used for the Poisson equation. This disadvantage is not related to the use of infinite elements; it is just the trouble of our implementation of the mesh generator. The use of a single meshing for each separate problem (Eqs. (8) and (10)) gives optimal results. A possible solution to reduce the mentioned bandwidth is the introduction of an algorithm of mesh renumbering in our implementations, a point to do in the future. Fig. 1 shows some solutions to the Poisson equation using the quadrupole approximation in the boundary and the infinite element method implemented in this work. The infinite elements map to 100 and 10,000 atomic units in order to study the differences as a function of "position" of the infinite. While the solution near the center of the domain does not exhibit differences (these points are not shown in the graph), these differences increase as the distance grows from the center. As we can see in the graph, for the kind of systems solved here, a change of two orders of magnitude in the position of the infinite does not shows appreciable differences in the solution.

2.2. Ionic potential

In an all-electron approach, the ionic potential is the sum of the Coulomb potential generated by each atomic nucleus present in the system,

$$V_{ion}(\boldsymbol{r}) = \frac{Z}{|\boldsymbol{r} - \boldsymbol{R}|} \tag{17}$$

where Z and $\boldsymbol{R}$ are the nuclear charge and the position of a nucleus, respectively. To exemplify an all-electron calculation, Table 2 presents the results for an all-electron Si atom calculated by our method and by the FHI98 pseudo-potential (PP in advance) package [22].

In a pseudo-potential approximation, the ionic potential involves retaining the angular momentum dependence by truncating the sum on angular momentum channels and decomposing into

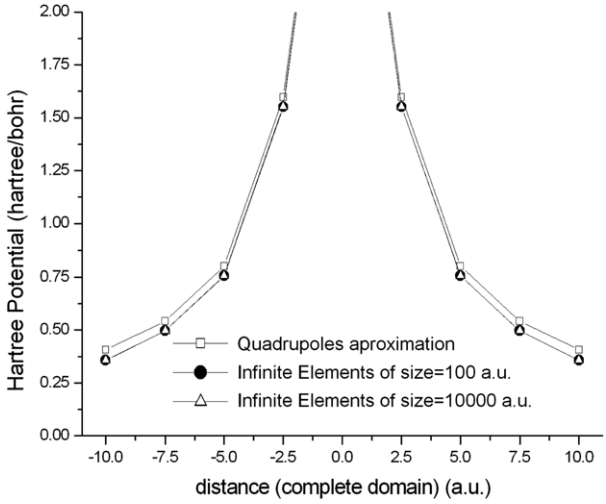
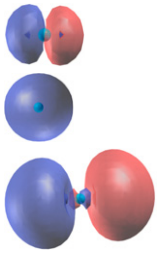


Fig. 1. Solution in $(0, 0, z)$ axis, of the Poisson equation with different sizes of "infinite" elements and quadrupole approximation in boundaries.

Table 2
All-electron calculation for Si atom. Total energy and eigenenergies in Hartree units.

	FHI98	Current work
Total energy	−288.8216	−289.3259
Orbital 1s	−65.3573	−65.079
Orbital 2s	−5.0987	−5.053
Orbital 2p	−3.5136	−3.502
Orbital 3s	−0.3998	−0.398
Orbital 3p	−0.1530	−0.154



two terms, a local term, that grossly treats the higher angular momentum components and a nonlocal term in the fully separable Kleinman–Bylander form [23], that both depends explicitly on angular momentum channels and directly on the pseudo-orbitals of the semilocal pseudo-potential. The ionic pseudo-potential takes the form [23],

$$V_{ion}^{PP}(\boldsymbol{r}) = V_{local} + \sum_{l=0}^{l_{max}} \sum_{m=-l}^{m=l} \frac{\left|\delta V_l \psi_{lm}^{PP}\right\rangle \left\langle \psi_{lm}^{PP} \delta V_l\right|}{\left\langle \psi_{lm}^{PP} \left|\delta V_l\right| \psi_{lm}^{PP}\right\rangle} \tag{18}$$

where $\delta V_l = V_l - V_{local}$ is the l-channel semilocal pseudo-potential and ψ_{lm}^{PP} is the atomic reference pseudo-eigenfunction with quantum indexes lm. Computationally, the local part does not involve extra calculation, but the non-local contributions require angular momentum projections over a spherical regime, centered at each nucleus. After applying the discretization to the non-local term, each *bra* and *ket* contribution provides a vector. Then each term in the summation is a single scalar product within the expectation value of V_l. We use the pseudo-potential approximation of Troullier and Martin [22,24]. Fig. 2 shows the PP calculation for the BeH_2 molecule.

2.3. Exchange–correlation potential

The exchange–correlation functional used is the standard local density approximation (LDA), which provides two contributions to the potential:

$$V_{xc}(r) = \frac{\partial E_{xc}^{LDA}[\rho(\boldsymbol{r})]}{\partial \rho} = \epsilon_{xc}(\rho(r)) + \rho(r) \frac{\partial \epsilon_{xc}[\rho(\boldsymbol{r})]}{\partial \rho} \tag{19}$$

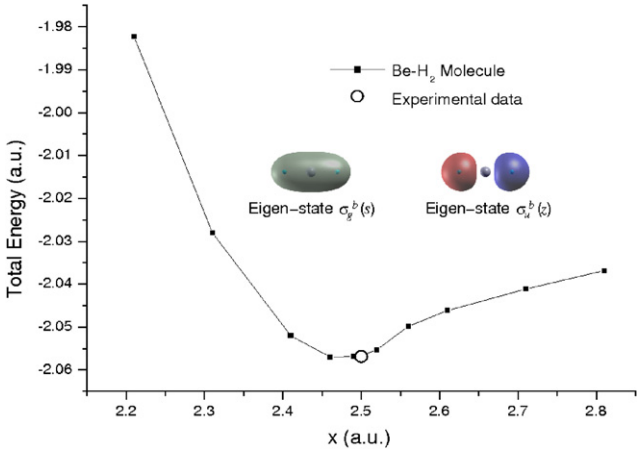


Fig. 2. PP calculation of BeH_2 molecule. The experimental value was taken from [25].

with $\epsilon_{xc}(r) = \epsilon_{ex}(r) + \epsilon_c(r)$ where $\epsilon_{ex}(r) = -\frac{3}{4}\left(\frac{3}{\pi}\right)^{\frac{1}{3}} \rho(r)^{\frac{1}{3}}$ and for the correlation contribution the Ceperley and Alder analytic forms are used [3,26].

2.4. Total energy calculation and external potential contribution

After convergence is reached, the total energy of the system is the sum of the earlier magnitudes: kinetic, ionic, Hartree and exchange–correlation contributions, to which we need to add the negative ion–ion contribution,

$$E_{tot} = E_T + E_{ion} + 0.5E_{hartree} + E_{xc} - E_{ion-ion}. \tag{20}$$

Each electronic energy term is obtained from the converged self-consistent electronic density and the ion–ion interaction energy is calculated as [3],

$$E_{ion-ion} = \sum_{i=1} \sum_{j>i} \frac{Z_i Z_j}{\left|\boldsymbol{R}_i - \boldsymbol{R}_j\right|} \tag{21}$$

summing all pairs of atomic nuclei present in the system.

3. Cycle of self-consistent and parallelization

3.1. Eigen solver

Once the local matrices created by each element are assembled into global matrices, a generalized eigenvalue system of the form (6) is built, where $\boldsymbol{\phi}/\epsilon$ is the auto-state (auto-energy) of the problem. In the case of the FEM a sparse system is obtained. In DFT, the Kohn–Sham equation is non-linear in density and it is solved by self-consistent cycles to reach the electronic ground state. This self-consistent procedure is stopped when the differences between the densities of two consecutive cycles are less than a certain tolerance. There are a lot of tools for solving this kind of problem when the matrix $\mathbf{M}$ is positive definite [27] or semi-definite [28]. If the system does not have this property, a direct solver taken for example from EisPACK [29] can be used. For some cases, especially for big matrices, an iterative method can be more convenient from the point of view of the time for calculation. These methods give us a few eigenvalues (usually the small ones) [30,31].

The FEM provides a natural way to simplify the eigenproblem, Eq. (6), making its resolution easier working a little with the algebra involved in the resolution of the problem. Each local matrix associated to each finite element is obtained after evaluating the integral of each contribution to the Hamiltonian. As we explain in Section 2.1 the integration uses general Gauss methods using

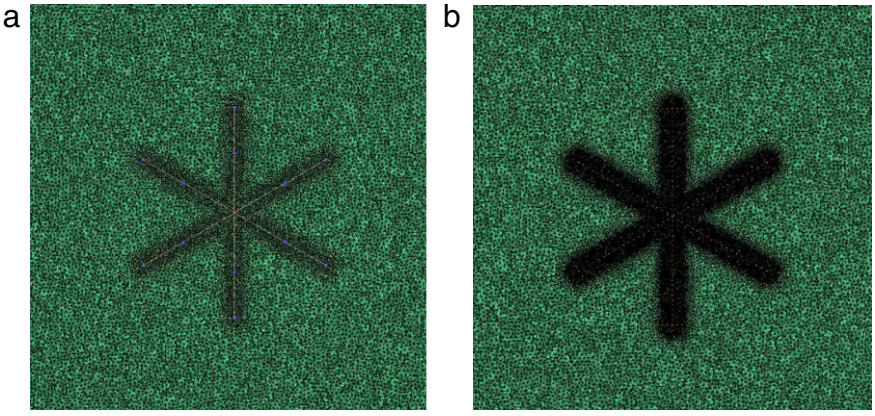


Fig. 3. Two examples of refinements in the plane of the molecule: (a) 2.5M elements and (b) 3.7M elements.

standard quadrature points. But, in the case of the mass matrix, using a close integration rule, diagonal matrices are obtained [17]. Thus the generalized eigenproblem (Eq. (6)) is reduced to a single problem, less computationally expensive to solve. To do that we use a deflated conjugate gradient solver [31–34], from which solution we obtain the small eigenvalues needed.

3.2. Analysis of convergence

In order to analyze the efficiency of the regular/irregular mesh used, the use of infinite elements or not, and the extension of the domain, we computed the convergence time for the Poisson equation and the eigen solver. Table 3 presents a resume of the most significant results. The logarithmic mesh means that we use refinement in the central atom following a logarithmic rule. For this calculus we use hexahedral elements with cubic interpolation function (27 nodes by element). With increasing degrees of freedom better results in less iterations are obtained. The cost is the increase in time, of course. In cases of chosen infinite elements (last file of Table 3 as an example), we can reduce the internal domain meshed (reducing the number of nodes), in comparison with the calculus with regular Dirichlet boundary condition. In the internal mesh we use logarithmic refinement too. The added external shell of infinite elements increases the cost of the Poisson solver in terms of CPU time. But as we explained, this is due to the discontinuous numbering of our mesh. Other tests with a continuous numbering give results similar to the ones obtained with a standard mesh. The number of iterations to reach convergence is reduced too, but not significantly.

The use of non-refined meshes, even in a small physical system, will not reach convergence with a number of nodes similar to those listed in the table. To obtain convergence with a homogenous grid size, we need to increase the degrees of freedom by one order of magnitude approximately (10^7).

3.3. Parallelization

The parallelization of the solver is based on a mesh partitioning technique using a Master–Slave strategy. Mesh partitioning is done by METIS [14] and parallelized with an MPI-based strategy. Fig. 3 shows an interesting refinement example used to solve the system benzene. The plane of the molecule is plotted in order to show the grade of refinements. The use of METIS to make the subdivision of the domain allows splitting it by numbers of elements, not by size of the domain. As we do not need to pass any additional information to each process, we divided the domain weighting by the amount of calculation to give to each processor. This point is very advantageous with respect to finite differences

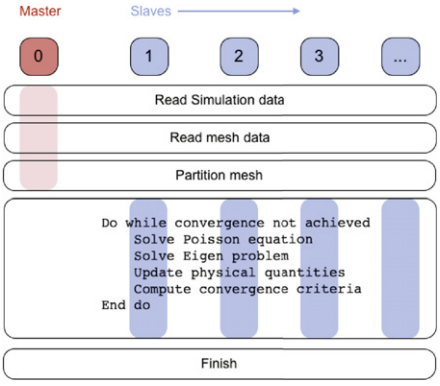


Fig. 4. Parallelization scheme.

because in the last case it is necessary to send a ghost point to the neighboring processors in order to complete the calculation. Although it is not discussed in this paper, it is worth mentioning that a second and deeper degree of parallelization is achieved by opening threads in internal loops using OpenMP directives, resulting in a hybrid-parallelization strategy. Iterative solvers are used to solve all the equations. The Master–Slave strategy consists of the following. The Master reads the simulation and mesh data, performs its partition and dumps some output files (for example of the convergence residuals). The slaves build the local element matrices (Hamiltonian, mass matrix or Laplacian: we refer to all of this as the LHS) and right-hand side (RHS in case of the solution of the Poisson equation) and are in charge of the resulting system solution in parallel. The scheme is illustrated in Fig. 4.

In a finite element implementation, only two kinds of communications are necessary between sub-domains. The first type of communication consists of exchanging arrays between neighbors with MPISendrecv. The strategy is the following. (1) For each slave, compute elemental LHS and RHS for each element. (2) For each slave, assemble (scatter) elemental RHS and LHS into global LHS and RHS. (3) Exchange RHS of boundary nodes, the nodes belonging to more than one sub-domain, and sum the contribution. (4) The operations of an iterative solver are matrix–vector multiplications. Then, for each slave, compute matrix–vector multiplication. (5) Exchange the results on the boundary nodes, as was done for the RHS. The second type of communication is global and of reduce type with MPIReduce. It is used to compute: (1) the convergence residual: the sum over all the nodes of the domain. Residuals are required to check the different convergence tolerances of the scheme. (2) Scalar products: they take the part of iterative solvers. There are some critical points of the parallelization strategy. In order to have an efficient algorithm

Table 3
Convergence analysis for total energy calculation of the ground state in the system Methano. Different grid sizes and several grades of refinement are tested, either with or without infinite elements outside of the central domain. Convergence error, cpu time, convergence time and number of iterations are compared.

Degrees of freedom	Domain	Poisson (s)	Eigen solver (s)	Total energy (hartree)	Iter.
$1 \cdot 10^5$	Cube 20 a.u. log.	197	525.35	$-0.89E_0$	31
$2.3 \cdot 10^5$	Cube 20 a.u. log.	300.92	523.36	$-0.95E_0$	31
$5 \cdot 10^5$	Cube 20 a.u. log.	545.0	890	$-0.97E_0$	30
$2 \cdot 10^6$	Cube 20 a.u. log.	980	1450	$-0.98E_0$	28
Total mesh $1.1 \cdot 10^6$ + Internal mesh $8.0 \cdot 10^5$	Cube 10 a.u. log. + Inf. elements 1000 a.u.	2090	400	$-E_0$	26

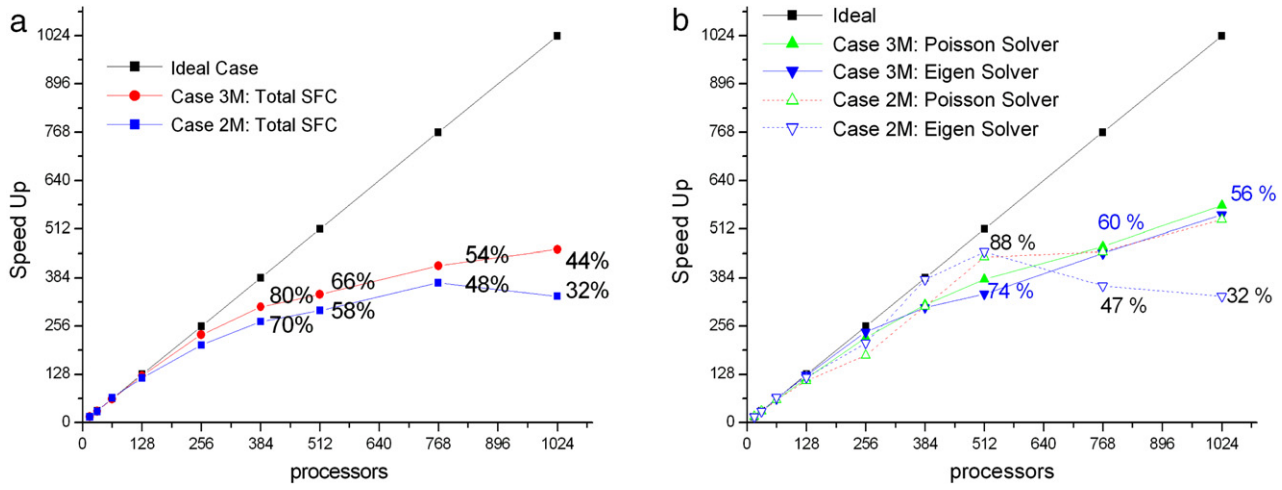


Fig. 5. Scalability measures for a mesh of 2M and 3M nodes. (a) Total time from the complete SFC. (b) Poisson and eigen solver times.

to run on thousands of processors, some important aspects of the parallelization must be carefully treated: mesh partitioning, node numbering and communication scheduling. The details of the implementation of these points are given in [35].

The scalability of the code has been studied maintaining the size of the problem fixed while the number of processors used in the simulation has been increased (hard scaling), and increasing the size of the problem, in order to see the relation with a bigger number of processors (soft scaling). In Fig. 5(a) and (b), we plot the speed up of the complete self-consistency cycle and the separate measures taken for the Poisson and the eigensystem solvers respectively. In all measures the hard scaling exhibits the expected lost of speed up, except for the case of the 3M mesh, as more work is carried out in the elemental loops. In the case of Fig. 5(b), when the independent time is taken for each solver, both measures show better results on average in the 2M case than in the 3M, but only up to 512 processors; after this number, the 3M cases show better results as expected. In the case of complete SCF, this improvement is shown for any number of processors. It should be mentioned that we have very few elements per CPU in the rightmost part of the graphs. In fact, for 512 CPUs on the 2M mesh, we get on average 3900 elements per CPU.

4. Time-dependent DFT

The time-dependent Kohn–Sham equation takes the form [36,37],

$$\hat{H}(t)\psi(\boldsymbol{r},t) = \left[-\frac{1}{2}\nabla^2 + V_{\mathit{eff}}(\boldsymbol{r},t)\right]\psi(\boldsymbol{r},t) = i\frac{\partial\psi(\boldsymbol{r},t)}{\partial t} \tag{22}$$

where this equation describes the evolution of the wave function ψ in a system perturbed by a changing external potential and so defined by a time-dependent Hamiltonian operator, $\hat{H}(t)$. In order to determine completely the dynamics of the system, there is required an initial value condition fixed by the stationary Kohn–Sham wave function.

Formally, the solution of Eq. (22) may be written as,

$$\begin{aligned}\psi(\boldsymbol{r},t) &= \hat{U}(t,0)\psi(\boldsymbol{r},0)\\ &= \mathcal{T}\exp\left(-i\int_0^t d\tau\hat{H}(\tau)\right)\psi(\boldsymbol{r},0)\end{aligned} \tag{23}$$

where $\hat{U}$ is the so-called evolution operator, or time propagator, and $\mathcal{T}$ exp is the time-ordered exponential. In practice, it is convenient to split the interval $[0,t]$ into smaller time intervals, and technically we make it with a constant time-step, Δt. So, in a short propagation,

$$\begin{aligned}&\psi(\boldsymbol{r},t+\Delta t)\\ &= \hat{U}(t+\Delta t,t)\psi(\boldsymbol{r},t) \approx \exp\left(-i\Delta t\hat{H}(\tau)\right)\psi(\boldsymbol{r},t)\end{aligned} \tag{24}$$

the time dependence of $\hat{H}$ is alleviated and the norm of the T exponential argument is reduced (it increases linearly with Δt). Besides, if the perturbation is small, the equations can be linearized and the linear response formalism could be used obtaining the first-order density–density response in the frequency domain [3,11]. In our case, following a simple formalism exploring the possibility to use the FEM in all the fields related to TDDFT, an explicit integration is used. This approximation requires knowing the Hamiltonian at times $t \leqslant \tau \leqslant t+\Delta t$, where it is unknown. Assuming the simplest approximation to the exponential propagator, time-reversal symmetry can be enforced which defines [38],

$$\hat{U}(t+\Delta t,t) \approx \exp\left(-i\frac{\Delta t}{2}\hat{H}(t+\Delta t)\right)\exp\left(-i\frac{\Delta t}{2}\hat{H}(t)\right). \tag{25}$$

The Hamiltonian at time $t+\Delta t$ must be either extrapolated or approximated with a previous evolution.

In order to do that, the exponential of $\hat{H}$ needs to be calculated. In the present work, the Taylor expansion of the exponential is

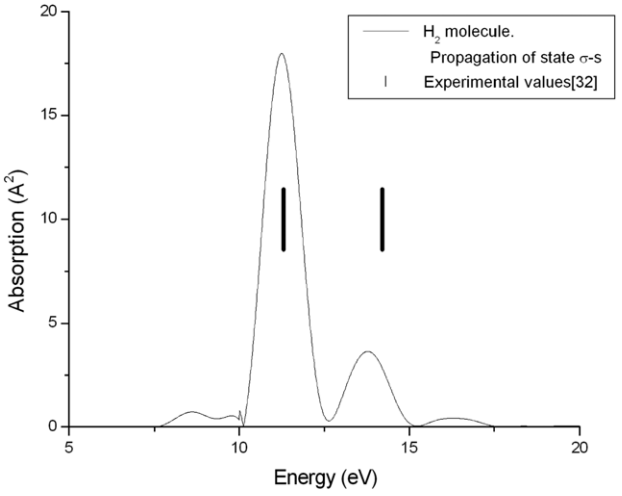


Fig. 6. Absorption spectra of the H_2 molecule from a initial kick applied along its molecular axis (z direction), which is calculated by our implementation TD-DFT. The experimental lines are shown for comparison.

truncated at order four:

$$\exp(-i\Delta t\hat{H}) \approx \sum_{n=0}^{4} \frac{(-i\Delta t)^n}{n!}\hat{H}^n. \tag{26}$$

In the FEM formulation we need to solve for each eigenstate an implicit time-evolution set of equations of the form $\hat{H}\psi^{t+1} = \psi^t$ where the known state is at time t and the unknown state is at the next time step. This implicit equation set is solved using preconditioned conjugated gradient methods. Only some examples with small molecules and simple perturbations are treatable until now. We apply it to the H_2 molecule with a small kick at $t = 0$ in the orthogonal direction to the molecular axis. With this simple perturbation, we can obtain an approximate absorption spectra, exhibiting the two characteristic peaks [39]. (See Fig. 6.)

5. Conclusion

Following previous studies, we developed a numerical approach for solving the DFT and TD-DFT equations that describe multi-electronic systems. The method is based on the representation on real space discretized with infinite and finite elements and it is embedded in a HPC environment. The focus was placed on two aspects: versatility and high performance computing. It shows a high capability and potential to solve big systems with great performance and accuracy, allowing the use of hundreds of processors almost linearly. The use of the finite element method, both for all-electron and pseudo-potential approximations, showed a very comfortable adaptability; besides that, the domain decomposition follows a natural way of reduced communications and bottleneck effects. The introduction of infinite elements in order to extend the numerical domain of Poisson's equation to infinity has also shown the versatility of the FEM applied to such types of systems. Needless to say, for clarity, the inclusion of the infinite element method doesn't mean a better selection than a multipole expansion to set Dirichlet boundary conditions for Poisson's equation. The inclusion of infinite elements does not improve accuracy or speed, because the domain should be large enough so that no further Dirichlet condition is worth setting in the eigenvalue problem. The advantage, in a parallel environment, is the elimination of the communications required by the second approach. Finally, the approach presented here is far from being a robust tool yet. The TD-DFT implementation is still under development and various improvements need to be added: more sophisticated approximations for the evolution operator, implementation of the linear response theory for small perturbations, and parallelization over states to improve the scalability [3], are some of the points that require development.

Acknowledgments

The authors acknowledge the European Union through the FP7 e-I3 ETSF (Contract No. 211956) project and to all the members of the CASE group in Barcelona.

References

[1] P. Ordejon, E. Artacho, J.M. Soler, Self.consistent order-N density-functional calculation for very large system, Physical Review B 53 (1996) 16.
[2] C.F. Sanz-Navarro, R. Grima, A. García, E.A. Bea, A. Soba, J.M. Cela, P. Ordejón, An efficient implementation of a QM–MM method in SIESTA, Theoretical Chemistry Accounts: Theory, Computation, and Modeling (Theoretica Chimica Acta) 128 (4–6) 825–833. http://dx.doi.org/10.1007/s00214-010-0816-5.
[3] A. Castro, M. Marques, et al., Octopus: a tool for the application of time-dependent density functional theory.
[4] J.R. Chelikowsky, N. Troullier, Y. Saad, Finite difference-pseudopotential method: electronic structure calculations without a basis, Physical Review Letters 72 (1994) 8.
[5] L. Kronik, A. Makmal, M. Tiago, M. Alemany, M. Jain, X. Huang, Y. Saad, J. Chelikowsky, PARSEC — the pseudopotential algorithm for real-space electronic structure calculations: recent advances and novel applications to nano-structures, Physica Status Solidi B 243 (2006) 1063.
[6] Yunkai Zhou, Yousef Saad, Murilo L. Tiago, James R. Chelikowsky, Parallel self-consistent-field calculations via Chebyshev-filtered subspace acceleration, Physical Review E 74 (2006).
[7] François Gygi, Electronic-structure calculations in adaptive coordinates, Physical Review B 48 (1993).
[8] N.A. Modine, Gil Zumbach, Efthimios Kaxiras, Adaptive-coordinate real-space electronic-structure calculations for atoms, molecules, and solids, Physical Review B 55 (1997) 16.
[9] S.R. White, J.W. Wilkin, M.P. Teter, Finite-element method for electronic structure, Physical Review B 39 (1989) 9.
[10] J.E. Pask, B.M. Klein, P.A Sterne, C.Y. Fong, Finite-elements methods in electronic-structure theory, Computer Physics Communications 135 (2001) 1–34.
[11] L. Lehtovaara, V. Havu, M. Puska, All-electron density functional theory and time-dependent density functional theory with high-order finite elements, Journal of Chemical Physics 131 (2009) 054103.
[12] D. Zhang, L. Shen, A. Zhou, X. Gong, Finite element method for solving Kohn–Sham equations based on self adaptative tetrahedral mesh, Physics Letters A 372 (2008).
[13] O. Certík, J. Vackar, J. Plesek, Density functional theory calculations using the finite element method, Proceedings of the Estonian Academy of Sciences 57 (2008) 3.
[14] Metis, family of multilevel partitioning algorithms. http://glaros.dtc.umn.edu/gkhome/views/metis.
[15] P. Hohenberg, W. Kohn, Physical Review 136 (1964).
[16] O.C. Zienkiewicz, R.L. Taylor, El método de los elementos finitos, 4a Edición, McGraw-Hill, Barcelona, España, 1994.
[17] Klaus-Jürgen Bathe, Finite Element Procedures, Prentice Hall, 1996.
[18] T.T. Abdel-Fattah, H.A. Hodhod, A.Y. Akl, A nobel formulation of infinite elements for static analysis, Computer and Structures 77 (2000).
[19] A. Houmat, Mapped Infinite p-element for two dimensional problems of unbounded domains, Computers and Geotechnics 35 (4) (2008) 608–615.
[20] L. Gavete, A. Ruiz, C. Manzano, Elementos infinitos con caida 1/r, Revista Internacional de Metodos Numericos para Calculo y Diseno en Ingenieria 6 (1990) 2.
[21] Peter Bettess, Infinite Elements, Penshaw Press, 1992.
[22] M. Fuchs, M. Scheffler, Ab initio pseudopotentials for electronic structure calculations of poly-atomic systems using density-functional theory, Computer Physics Communications 119 (1999).
[23] L. Kleinman, D.M. Bylander, Efficacious form for model pseudopotentials, Physical Review Letters 48 (1982) 1425.
[24] N. Troullier, J.L. Martins, Efficient pseudopotentials for plane-wave calculations, Physical Review B 43 (1991) 3.
[25] A. Shayesteh, K. Tereszchuk, P.F. Bernatha, R. Colin, Infrared emission spectra of BeH2 and BeD2, Journal of Chemical Physics 118 (2003) 8.
[26] R.G. Parr, W. Yang. (Eds.), Density Functional Theory of Atoms and Molecules, Oxford Science publications, 1989.
[27] http://www.netlib.org/lapack/.
[28] http://www.caam.rice.edu/software/ARPACK/.
[29] http://www.netlib.org/eispack/.
[30] Ho-Jong Jang, Preconditioned conjugate gradient method for large generalized eigenproblems, in: Trend in Mathematics, in: ICMS, vol. 4, 2001, p. 2.
[31] Kresse, Furthmüller, Efficient iterative schemes for ab initio total-energy calculations using a plane-wave basis set, Physical Review B 54 (1996) 11169.
[32] Y. Saad, J. Yeung, J. Erhel, F. Guyomarc'h, A deflated version of the conjugate gradient algorithm, SIAM Journal on Scientific Computing 21 (2000).

[33] R. Lohner, F. Mut, J. Cebral, R. Aubry, G. Houzeaux, Deflated preconditioned conjugate gradient solvers for the pressure-poisson equation: extensions and improvements, International Journal for Numerical Methods in Engineering 87 (1–5) (2011) 2–14.
[34] R. Aubry, F. Mut, R. Löhner, J.R. Cebral, Deflated preconditioned conjugate gradient solvers for the Pressure-Poisson equation, Journal of Computational Physics 227 (2008).
[35] G. Houzeaux, M. Vázquez, R. Aubry, J.M. Cela, A massively parallel fractional step solver for incompressible flows, Journal of Computational Physics 228 (2009).
[36] M. Marques, E.k.U Gross, Time dependent density functional theory, in: Annual Review of Physical Chemistry, vol. 55, pp. 427–455. Volume publication date June 2004.
[37] Time-Dependent Density Functional Theory, in: M.A.L. Marques, C.A. Ullrich, F. Nogueira, A. Rubio, K. Burke, E.K.U. Gross (Eds.), Series: Lecture Notes in Physics, vol. 706, 2006, Hardcover, XXXIV, 591 p.
[38] A. Castro, M. Marques, A. Rubio, Propagators for time-dependent Kohn–Sham equations, Journal of Chemical Physics 121 (2004) 8.
[39] J.I. Martínez, M. Isla, J.A. Alonso, Theoretical study of molecular hydrogen clusters, The European Physical Journal 43 (2007).