Efficient computation of the coupling matrix in Time-Dependent Density Functional Theory [*]

Emmanuel Lorin de la Grandmaison [†] Shivaraju B. Gowda [‡] Yousef Saad [‡]
Murilo L. Tiago [§] James R. Chelikowsky [§]

March 31, 2006

Abstract

We present an efficient implementation of the computation of the coupling matrix arising in time-dependent density functional theory. The two important aspects involved, solution of Poisson's equation and the assembly of the coupling matrix, are investigated in detail and proper approximations are used. Poisson's equation is solved in the reciprocal space and bounded support of the wave functions are exploited in the numerical integration. Experiments show the new implementation is efficient by an order of magnitude when compared with a standard real-space code. The method is tested to compute optical spectra of realistic systems with hundreds of atoms from first principles. Details of the formalism and implementation are provided and comparisons with a standard real-space code are reported.

1 Introduction

A successful approach that is common in electronic structure calculations is to use *ab initio* pseudopotentials within density functional theory (DFT) [1]. This approach has been used to predict mechanical, chemical, and electronic properties of many classes of solids, liquids, molecules, and more. In density functional theory, the original N-electron problem is converted into an effective one-electron problem, where all non-classical electron interactions (namely, exchange and correlation) are replaced by an additive one-electron potential that is a functional of the charge density. While this mapping is formally exact, it is approximate in practice because the exact functional is unknown. A common approximation to this functional, which is used in this paper, is the Local Density Approximation (LDA) where the exchange-correlation functional is defined as a local function of the charge density. A second important approximation made in this approach is the use of pseudopotentials. In pseudopotential theory, only valence electrons, the ones forming chemical bonds, are treated explicitly. The effect of core electrons is accounted for by replacing the true atomic potential with an effective "pseudopotential". The pseudopotential is a smooth, slowly-varying potential, whereas the true atomic potential is rapidly varying and has a $\sim 1/r$ singularity

[*]Work supported by NSF grants ITR-0082094, DMR-0325218, by DOE under Grants DE-FG02-03ER25585, DE-FG02-03ER15491, and by the Minnesota Supercomputing Institute

[†]Département de Mathématiques, Université d'Orsay, 91405 Orsay (France) and Centre de Mathématiques et de Leurs Applications, Ecole Normale Supérieure de Cachan, 94235 Cachan (France)

[‡]Department of Computer Science and Engineering, Institute for the Theory of Advanced Materials in Information Technology, Digital Technology Center, University of Minnesota, Minneapolis, MN 55455

[§]Department of Chemical Engineering and Materials Science, Institute for the Theory of Advanced Materials in Information Technology, Digital Technology Center, University of Minnesota, Minneapolis, MN 55455

at the nuclear position. In addition, the removal of core electrons from the picture, results in a much smaller number of eigenvectors to compute.

However, because traditional time-independent DFT is inherently a ground-state theory [1] it is not suitable for the computation of electronic excitations in general, and optical properties in particular. A generalized, *time-dependent* DFT (TDDFT) formalism has been developed which allows the computation of excited state properties. This approach was applied successfully to a wide range of atoms and small molecules [2–7]. Optical properties of larger systems of molecules, clusters, or "quantum dots" (small fragments of bulk material) in the range of many hundreds of atoms are much harder to address despite their outstanding importance in physics. This is primarily due to severe computational limitations.

The recent article [5] demonstrated how this pseudopotential-TDDFT formalism can enable the calculation of properties for systems with several hundreds of atoms. Parallel computing was a major part of the success of this method, and so enhancements of the parallel efficiency of the method undertaken in the article [4] resulted in significant additional gains. The parallel code in these two articles uses a real-space implementation of a frequency-domain formulation of the TDDFT equations [3]. One difference between the present approach and existing numerical implementations of TDDFT (see *e.g.*, Refs. [6, 7]) is the absence of an explicit basis set (plane wave or atomic orbital basis) in which the Kohn-Sham equations are solved. Instead, we calculate DFT eigenstates on a grid in real space. Numerical convergence can be easily controlled by two parameters: one defining the extent of the real-space domain and another defining the density of grid points [5, 8, 9].

The main computational problem in implementing the TDDFT approach lies in the computation of a coupling matrix K. This matrix is dense and symmetric and its dimension n_K is of the order of the square of the number of states considered. Each row of the matrix requires the solution of a Poisson equation. In [5] and [4] these equations were handled by a preconditioned Conjugate Gradient method. Most of the time to complete the calculation is spent in solving these systems. The main contribution of this paper is an efficient implementation of the computation of the coupling matrix to handle the computational complexity of TDDFT. To this end, we use the Fast Fourier Transform to solve Poisson's equation and exploit the fact that the wave functions have bounded support to reduce the amount of computation.

This paper is organized as follows. In section 2 we present the formalism along with a direct method to compute the coupling-matrix based on a simple plane wave approach. The advantage of this method is that it exploits features of the physical and mathematical problems. In Section 3 we look at the simplified formulation of the coupling matrix. Then in Section 4, algorithms available for solving the Poisson equation are evaluated. In Section 5 we deal with parallelization aspects after studying the complexity and also discuss the methods of approximation to gain in computational speed. In section 6 some numerical results showing the optical spectra of different molecules are presented, which are computed with our method and finally, using a parallel version of the code, we present CPU-time comparisons with the real-space method of [4].

2 Formalism

We follow [5] and [4] and use the frequency-domain formulation of TDDFT as discussed by Casida [3]. In the following we give a brief background of the method. Details and physical justifications may be found in [3, 5, 10, 11].

The procedure begins with the solution of the time-independent Schrödinger equation, using the pseudo-potential DFT formulation [1] in the domain Ω,

$$\left(-\frac{\triangle}{2} + \sum_{R_a} V_{ps}(\mathbf{r} - \mathbf{R}_a) + V_H(\rho(\mathbf{r})) + V_{xc}(\rho(\mathbf{r}))\right) \psi_n(\mathbf{r}) = \varepsilon_n \psi_n(\mathbf{r}). \tag{1}$$

In the above equation, ψ_n and ε_n denote the n^{th} single-electron eigenfunction and eigenvalue respectively. The charge density, denoted by $\rho(\mathbf{r})$, is the sum of squares of the eigenfunctions $\psi_n(r)$ associated with the occupied states. Finally, V_H is the Hartree (electron-electron) interaction potential, V_{ps} is the pseudo-potential which represents the interactions between ions located in $\mathbf{R}_a$ and V_{xc} is the exchange-correlation potential. The above equation is solved for all occupied states and a number on unoccupied states resulting in a set of eigenpairs ε_j, ψ_j. The eigenfunctions are then used to assemble the coupling matrix K in the form:

$$K_{ij,kl} = 2\int_\Omega \int_\Omega \bar{\psi}_i(\mathbf{r})\psi_j(\mathbf{r}) \left(\frac{1}{|\mathbf{r} - \mathbf{r}'|} + \frac{dV_{xc}(\rho(\mathbf{r}))}{d\rho(\mathbf{r})}\delta(\mathbf{r} - \mathbf{r}')\right) \psi_k(\mathbf{r}')\bar{\psi}_l(\mathbf{r}') d\mathbf{r} d\mathbf{r}', \tag{2}$$

where Ω is the physical domain, which is a subset of $\mathbb{R}^3$. Given this, the optical absorption spectrum of a system of molecules is computed as,

$$QF_I = \omega_I^2 F_I, \tag{3}$$

where

$$Q_{ij,kl} = \delta_{ik}\delta_{jl}\omega_{kl}^2 + 2\sqrt{\lambda_{ij}\omega_{ij}} K_{ij,kl} \sqrt{\lambda_{kl}\omega_{kl}} \tag{4}$$

with $\lambda_{ij} = f(\varepsilon_i) - f(\varepsilon_j)$ is the difference between occupied states (f denotes the Fermi-Dirac function) and $\omega_{ij} = \varepsilon_i - \varepsilon_j$. The term ω_{ij} is directly related to the optical transition energy.

The eigenvalues and eigenfunctions of the matrix $Q(\omega)$ are then computed and, finally, the oscillator strengths associated with the transition energy ω_I are obtained as follows:

$$f_I = \frac{2}{3} \sum_{\beta=\{x,y,z\}} |B_\beta^T R^{1/2} F_I|. \tag{5}$$

Here, $R_{ij,kl}(\omega) = \delta_{ik}\delta_{jl}\lambda_{kl}\omega_{kl}$ and $(B_\beta)_{ij} = \int \psi_i^*(\mathbf{r})\beta\psi_j(\mathbf{r})d\mathbf{r}$ for the three cartesian directions $\beta = \{x, y, z\}$.

This formalism leads to three distinct computational blocks as shown in Fig. 1. The first block computes the DFT solution to give the eigenvalues, occupation states and wave functions. Then the coupling matrix is constructed using the output of the first block. Finally the eigenvalues and eigenvectors of the dense matrix Q are determined to compute the optical absorption spectrum of the system. Our implementation divides the three blocks as separate program units with the intermediate outputs written to a file (in binary format), which is subsequently read as an input in the next block. All the three blocks are implemented in parallel, using MPI for communication. The implementation of the parallel DFT solution can be found in [12]. In this paper we concentrate on the parallel implementation of the construction of the coupling matrix. The matrix Q is symmetric, dense and real. SCALAPACK [13] is used for computing its eigenvalues and eigenvectors in parallel.

3 Coupling matrix construction

The evaluation of the coupling matrix given in Eq. (2), is the most expensive part of a TDDFT calculation. Computing the matrix K consists of evaluating a large number of entries, each of which is very costly to compute.

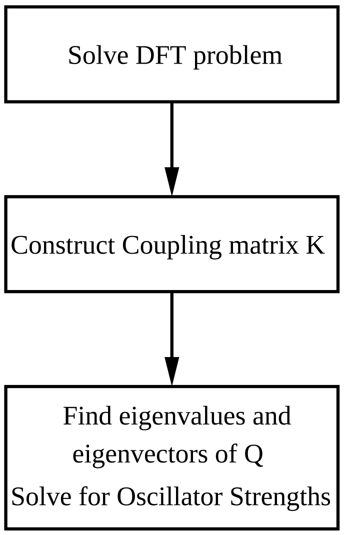
Solve DFT problem
Construct Coupling matrix K
Find eigenvalues and eigenvectors of Q
Solve for Oscillator Strengths

Figure 1: Flowchart for finding oscillator strengths using TDDFT.

Equation (2) consists of a sum of two integrals which can be computed separately. The first, double integral can be rewritten as:

$$\int_\Omega\int_\Omega \bar{\psi}_i(\mathbf{r})\psi_j(\mathbf{r})\frac{1}{|\mathbf{r}-\mathbf{r}'|}\psi_k(\mathbf{r}')\bar{\psi}_l(\mathbf{r}')d\mathbf{r}d\mathbf{r}' = \int_\Omega \psi_k(\mathbf{r}')\bar{\psi}_l(\mathbf{r}')\int_\Omega \frac{1}{|\mathbf{r}-\mathbf{r}'|}\bar{\psi}_i(\mathbf{r})\psi_j(\mathbf{r})d\mathbf{r}d\mathbf{r}' \ . \tag{6}$$

If we define

$$\rho_{ij}(\mathbf{r}) \equiv \bar{\psi}_i(\mathbf{r})\psi_j(\mathbf{r}), \tag{7}$$

and

$$\Phi_{ij}(\mathbf{r}') \equiv \int \frac{1}{|\mathbf{r}-\mathbf{r}'|}\bar{\psi}_i(\mathbf{r})\psi_j(\mathbf{r})d\mathbf{r}, \tag{8}$$

then we see that

$$\Phi_{ij}(\mathbf{r}') = \int \frac{\rho_{ij}(\mathbf{r})}{|\mathbf{r}-\mathbf{r}'|}d\mathbf{r}. \tag{9}$$

With appropriate boundary conditions, this is well-known to be the solution of the Poisson equation:

$$\nabla^2\Phi_{ij}(\mathbf{r}') = -4\pi\rho_{ij}(\mathbf{r}'). \tag{10}$$

Note that ρ_{ij} is not a physical charge density and Φ_{ij} is not a physical potential. However, they do have charge and potential units, respectively, and obey the same Poisson relation that a "true" charge density and Coulomb potential do. Solving the Poisson equation (10) using fast solvers may be more efficient than evaluating Φ_{ij} from the evaluation of the integral Eq. (9) by a "direct summation". There is indeed a wide variety of fast Poisson solvers available for solving Eq. (10) on regular meshes.

Once $\Phi_{ij}(\mathbf{r}')$ is found by solving the Poisson equation, we end up with the following form of Eq. (2):

$$K_{ij,kl} = 2\int_\Omega \rho_{lk}(\mathbf{r}')\Phi_{ij}(\mathbf{r}')d\mathbf{r}' + 2\int_\Omega \rho_{ij}(\mathbf{r})\frac{dV_{xc}[\rho(\mathbf{r})]}{d\rho(\mathbf{r})}\rho_{lk}(\mathbf{r})d\mathbf{r}. \tag{11}$$

The two single integrals in Eq. (11) must be computed by direct summation but this can be done for both of them at the same time:

$$K_{ij,kl} = 2\int_\Omega \Big[\Phi_{ij}(\mathbf{r}) + \rho_{ij}(\mathbf{r})\frac{dV_{xc}[\rho(\mathbf{r})]}{d\rho(\mathbf{r})}\Big]\rho_{lk}(\mathbf{r})d\mathbf{r}. \tag{12}$$

In practice, the computation of K is performed row-wise, and consists of two phases. First, $\Phi_{ij}(\mathbf{r}')$ is obtained from the solution of (10). Then, the integral Eq. (12) is evaluated. Equation (10) must be solved only *once per row*, *i.e.*, once per pair i, j, whereas Eq. (12) needs to be evaluated for each matrix entry, *i.e.*, for each pair of pairs of indices $\{(i,j),(k,l)\}$. Fig. 2 illustrates the assembly of the coupling matrix.

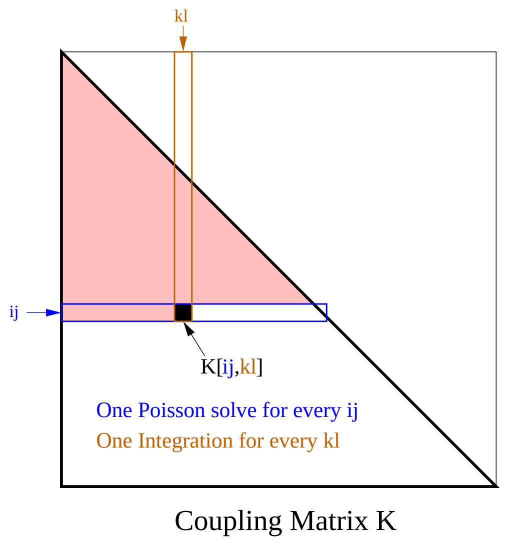

Figure 2: Construction of the coupling matrix.

The dimension *kdim* of the coupling matrix is of the order of the square of the number of states considered. The coupling matrix is symmetric, so only the lower triangular part is assembled. The calculation of each row is independent of the others, so it can be easily parallelized. Wave functions are localized in real space, *i.e.*, they have a small support and this property can be exploited to reduce the cost of computing the integral (11).

4 Solution of Poisson's Equation

The solution of the Poisson equation constitutes the biggest part of the cost of the coupling matrix construction. A three-dimensional grid with constant grid spacing and constant coefficients is used. Furthermore, the coupling matrix is built in parallel, by having each processor compute a number of rows of the matrix. As a result the Poisson equation is solved on a single processor. This is the

Gauss Elimin.	PCG	FFT	Multigrid
$N^{7/3}$	$N^{3/2}$	$N \log N$	N

Table 1: Comparison of the computational complexity for several fast Poisson solvers. The complexity for Gaussian Elimination assumes a 3-D mesh leading to a banded matrix of band $N^{2/3}$.

best situation for the use of a "Fast Poisson Solver". A Fast Poisson Solver is any technique which can solve the Poisson equation in $\mathcal{O}(N \log N)$ operations where N is the number of points in the discretization.

Table 1 shows the complexity of different algorithms for solving Poisson's equation. The class of "real space methods" work directly in physical space. The Poisson equation is discretized by, e.g., finite differences and the resulting linear system is solved using an iterative method or a multigrid technique. In contrast, planewave methods work in the Fourier (or reciprocal) space. In this case, the Poisson equation is trivially solved because the Fourier modes diagonalize the Laplacian.

In [4], a real space method with Preconditioned Conjugate Gradient (PCG) method was used to solve the resulting linear systems. Among the conclusions in [4] is the fact that the Conjugate Gradient is somewhat ill-adapted to the problem at hand. The discretization uses high-order finite differences. This has a negative impact on sparsity as well as convergence. Preconditioning was not helpful as in other cases because the modest gain in the number of steps to converge was counterbalanced by the additional cost of applying the preconditioner. Multigrid methods might be a better choice, but they too face some difficulties. The first comes again from the use of high order finite difference discretization – which standard MG methods do not handle. The second is a practical problem: a proper implementation would be quite complex due to nonuniform data access patterns and the use of additional memory.

Plane wave basis methods are attractive for computing Φ_{ij} since they exploit the fast Fourier transform. Discrete truncated sets of plane wave bases on the reciprocal lattice is a natural basis set for periodic systems. Hence, they are the preferred method in density functional theory (DFT) for infinite periodic systems such as bulk solids. This method is very popular because of the use of the highly optimized vendor supplied FFT routines. However, for finite localized systems that are not periodic, such as slabs or molecular clusters, the use of a discrete set of plane waves will implicitly generate unwanted periodic images of the cell being studied. Nevertheless, discrete plane wave basis sets are often used for studying finite systems because of the available efficient implementation of the FFT, and errors in the potential are usually ignored, or diminished by using a supercell. However, this error might be substantial and might affect the equilibrium structure and dynamics of weakly bounded molecules or clusters. Several methods are proposed for treating this problem while at the same time taking advantage of the excellent speed of FFT packages. The spurious effects of the periodic cells can be removed and appropriate boundary conditions applied with some modifications. This usually involves additional computations and complications.

Several methods have been proposed to minimize or eliminate the effects of the periodic images for localized systems. Castro and Rubio [14] investigated cut-off method and multipole correction method and compared them to the real-space approach. The cut-off based methods consider a bigger cell and implement a cut-off between the values of the periodic cells. Multipole-correction methods, treat long range potential analytically and short range potential using FFT's. In this Section we describe the uncorrected plane wave method first, and next we will discuss methods to remove the spurious interaction of the periodic images.

4.1 Computation of uncorrected $\mathbf{\Phi}_{ij}$

Consider the term Φ_{ij}, which is the solution of equation (10). The Fourier based Fast Poisson Solver discussed earlier is formally equivalent to applying a forward discrete Fourier transform (denoted by $\mathcal{F}$) followed by an inverse Fourier transform (denoted by $\mathcal{F}^{-1}$):

$$\Phi_{ij}(\mathbf{r}) = 4\pi\mathcal{F}^{-1}\ \left[\ \mathcal{F}(\Psi_i\bar{\Psi}_j)(\mathbf{k})/|\mathbf{k}|^2\ \right](\mathbf{r}). \tag{13}$$

A method based on the above approach, using FFT, would give satisfactory results in term of CPU-time and precision. However, it can be improved by using some specific features of the wave functions and the Fourier transform. To begin with, it is important to address an issue related to the feasibility of our approach. Since wave functions are functions with nearly bounded support, their Fourier transforms never vanish at infinity (Paley-Wiener theorem), *i.e.*, they cannot theoretically be constrained to be in a bounded domain. However, since the wave functions are sufficiently differentiable, and decrease to zero fast enough at infinity[1], their Fourier transform does decrease to zero at infinity. In practice, wave functions have nearly bounded support and they verify these assumptions.

Since $\Psi_i\bar{\Psi}_j$ are functions with nearly bounded support, their Fourier transform is also such that:

$$\mathcal{F}(\Psi_i\bar{\Psi}_j)(\mathbf{k}) = \sum_{\mathbf{r}} e^{i\mathbf{k}.\mathbf{r}}(\Psi_i\bar{\Psi}_j)(\mathbf{r}) = \sum_{\mathbf{r}\,\in\,\mathrm{Supp}(\Psi_i\bar{\Psi}_j)} e^{i\mathbf{k}.\mathbf{r}}(\Psi_i\bar{\Psi}_j)(\mathbf{r}).$$

This remark can be exploited to reduce the domain on which to compute the Fourier transform, that is to reduce the frequency cut-off in the Fourier expansion. For the inverse Fourier transform the argument is slightly different. The inverse Fourier transform gives the Hartree potential (C is a positive constant):

$$\Phi_{ij}(\mathbf{r}) = C\sum_{\mathbf{k}} e^{-i\mathbf{k}.\mathbf{r}}\mathcal{F}(\Psi_i\bar{\Psi}_j)(\mathbf{k})/|\mathbf{k}|^2. \tag{14}$$

As $\mathcal{F}(\Psi_i\bar{\Psi}_j)(\mathbf{k})$ is bounded (regularity of the wave functions) and the general term $1/|\mathbf{k}|^2$ series converges, we can truncate the sum in (14) by dropping those terms with $\mathbf{k}$ for $|\mathbf{k}|$ large enough (say $|\mathbf{k}| > K_{\mathrm{max}}$). Note that the orthogonality of the wave functions implies that $\mathcal{F}(\Psi_i\bar{\Psi}_j)(\mathbf{0}) = \mathbf{0}$, so there is no division by zero when $\mathbf{k} = \mathbf{0}$ and $i \neq j$.

The result is that it is possible to work with a reduced box:

$$\Phi_{ij}(\mathbf{r}) \sim C\sum_{|\mathbf{k}|\leqslant K_{\mathrm{max}}} e^{-i\mathbf{k}.\mathbf{r}}\mathcal{F}(\Psi_i\bar{\Psi}_j)(\mathbf{k})/|\mathbf{k}|^2.$$

In practice one can reduce the domain on which to compute the fast Fourier transform: instead of working on the whole domain $\mathbf{\Omega}$, the computation of the Fourier transform and inverse Fourier transforms can be restricted to the subdomain $\mathbf{\Gamma} \subset \mathbf{\Omega}$ obtained from enforcing the restriction $|k| \leq K_{max}$.

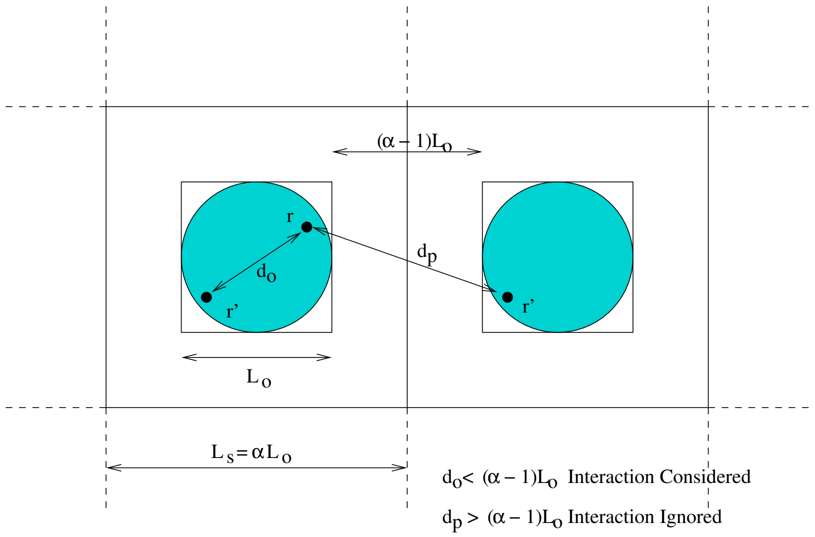
(α − 1)L$_o$
r
d$_o$
r'
d$_p$
r'
L$_o$
L$_s$=αL$_o$
d$_o$< (α − 1)L$_o$ Interaction Considered
d$_p$ > (α − 1)L$_o$ Interaction Ignored

Figure 3: Cutoff of Coulomb interaction between periodic images of a supercell.

4.2 Cut-off methods

Cut-off methods [14–16] are designed to correct spurious effects due to Coulomb interactions between periodic images, and usually require the use of bigger periodic cells. Using a large enough cell leads to a formally exact , but computationally expensive calculation. Equation (9) is an infinite convolution, but in practice can be evaluated as in equation (13). However a Fourier convolution of $\Phi(\mathbf{r})$, Eq. (13), includes interactions between points $\mathbf{r}$ and $\mathbf{r}'$ when both are on the same cell *and* when they belong to different cells. In order to avoid the last possibility, we impose a modification to the Hartree potential,

$$\Phi^{cut}(\mathbf{r}) = \int_{cell} \frac{\rho(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|} d\mathbf{r}'. \tag{15}$$

where the integral is performed only over the cell containing $\mathbf{r}$. This effectively defines a modified Coulomb interaction,

$$f^{cut}(\mathbf{r},\mathbf{r}') = \begin{cases} \dfrac{1}{|\mathbf{r}-\mathbf{r}'|} & \text{For } \mathbf{r},\ \mathbf{r}' \text{ in same cell} \\ 0 & \text{Otherwise} \end{cases} \tag{16}$$

Depending upon the cut-off considered, there are two types of methods: spherical cut-off and cubic cut-off [14]. Although both methods have comparable performance, the spherical cut-off method is easier to implement. As shown in Fig. 3, let L_o be the length of the original cell and $L_s = \alpha L_o$, be the size of the supercell, centered on the original one. For these cells, we can define the truncated Coulomb interaction:

$$f^{cut}(\mathbf{r},\mathbf{r}') = \begin{cases} \dfrac{1}{|\mathbf{r}-\mathbf{r}'|} & \text{For } |\mathbf{r}-\mathbf{r}'| < (\alpha-1)L_o \\ 0 & \text{Otherwise} \end{cases} \tag{17}$$

[1]Typically, wave functions belonging to a Schwartz space

The advantage of using a spherical cut-off is that the function f^{cut} in Eq. (17) has analytic Fourier transform: $\mathcal{F}(f^{cut})(\mathbf{k}) = 4\pi\left[1-\cos(\mathbf{k}(\alpha-1)L_o)\right]/\mathbf{k}^2$. After expanding the supercell, one can easily implement a spherical cutoff by means of the following replacement in Eq. (13):

$$\frac{1}{|\mathbf{k}|^2} \rightarrow \frac{1}{|\mathbf{k}|^2}\left[1-\cos(\mathbf{k}[\alpha-1]L_o)\right] \ .$$

In general, the choice $\alpha \geq 1+\sqrt{3}$ ensures that there is Coulomb interaction involving points $\mathbf{r}$ and $\mathbf{r}'$ on the same cell only. For the case of electron wave functions that are required to vanish outside a spherical domain of radius $L_o/2$ [12], one can choose $\alpha \geq 2$ and suppress exactly the interaction between images. In [14], it is shown that reducing the ratio of larger cell to the original cell to *e.g.* $\alpha = 1.7$ still gives a reasonable approximation for the solution of the Poisson's equation.

4.3 Long-range and short-range splitting

Another method to circumvent the effect of interaction between the local cell and its neighbors is to use a technique described, e.g., in [17], and called FALR (Fourier Analysis for Long-Range). FALR allows to treat separately the long-range and short-range parts of the potential. Specifically, the long-range part of the potential is treated analytically while the short-range part is treated using the Fourier transform. Formally, we solve the following systems:

$$\begin{cases} -\triangle\Phi_{ij}^{(\text{long})}(\mathbf{r}) = 4\pi\rho_{ij}^{(\text{long})}(\mathbf{r}), \\ -\triangle\Phi_{ij}^{(\text{short})}(\mathbf{r}) = 4\pi\rho_{ij}^{(\text{short})}(\mathbf{r}) = 4\pi\left(\rho_{ij}(\mathbf{r}) - \rho_{ij}^{(\text{long})}(\mathbf{r})\right), \end{cases} \quad (18)$$

and then add the results, $\Phi = \Phi^{(\text{long})} + \Phi^{(\text{short})}$, to obtain the final solution. Because we treat

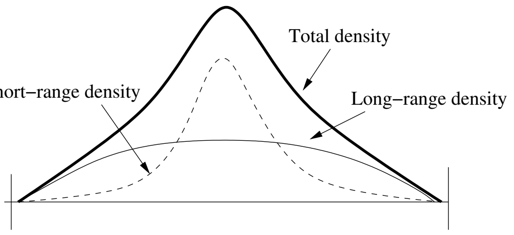

Figure 4: Long-range part and short range part of the density.

only the short-range part by discrete Fourier transform, the interactions between the different copies of the potential can be reduced. The way to determine "analytically" the long-range part of the potential is to use a multipole expansion. Thus, there are three steps required and they can be summed-up as follows:

1. First express the solution Φ_{ij} of the Poisson equation as a multipole expansion.

2. For the long-range part of the potential Φ_{ij} we have to build an associated long-range density from $\Psi_i\bar{\Psi}_j$.

3. For the short-part of $\Psi_i\bar{\Psi}_j$ we use a Fourier and inverse Fourier transform as explained in the previous section to obtain the short-range part of the potential Φ_{ij}.

We now provide a few additional details. Recall that we want to solve (10) knowing that the solution of this equation outside the support of $\Psi_i\bar{\Psi}_j$ can be expressed in spherical coordinates by a multipole expansion. Denoting by q_{lm} the spherical multipole moments, Y_{lm} the spherical harmonics, and (r, θ, ϕ) the spherical coordinates, this expression is:

$$\Phi_{ij}(\mathbf{r}) = \sum_{l=0}^{l=\infty} \sum_{m=-l}^{m=+l} \frac{4\pi}{1+2l} \frac{q_{lm}}{r^{l+1}} Y_{lm}(\theta, \phi), \ \mathbf{r} \in \left(\text{Supp}(\Psi_i\bar{\Psi}_j)\right)^{\text{C}} \tag{19}$$

with:

$$q_{lm} = \int \Psi_i(\mathbf{r})\bar{\Psi}_j(\mathbf{r}) Y_{lm}(\theta, \phi) r^l d\mathbf{r} \quad \text{and} \quad Y_{lm}(\theta, \phi) = \sqrt{\frac{1+2l}{4\pi} \frac{(l-m)!}{(l+m)!}} P_l^m(\cos\theta) e^{im\phi}. \tag{20}$$

In the above expression, P_l^m are the standard Legendre polynomials. In practice, only a finite expansion is used to extract the long-range part of the solution.

Next, we need to define a long-range density function from $\Psi_i\bar{\Psi}_j$ denoted by $f_{ij}^{(\text{long})}$. Since we consider long distance interactions, it can be assumed that the charge density is point-like and spherical symmetry can be assumed. As a result the three-dimensional Poisson equation becomes a one-dimensional radial equation.

Then the long-range density $f_{ij}^{(\text{long})}$ is expanded in spherical harmonics as follows:

$$f_{ij}^{(\text{long})} = \sum_{l=0}^{l=\bar{l}} \sum_{m=-l}^{m=+l} \alpha_{lm} f_l(|\mathbf{r}|) Y_{lm}(\theta, \phi)$$

where this expansion is such that its multipole moments are the same as those of the expansion of $f_{ij}^{(\text{long})}$, *i.e.*, $q_{lm}^{(\text{long})} = q_{lm}$. Details can be found in [17].

4.4 Practical computation of Φ_{ij}

When implementing the FFT-based solution of the Poisson's equation, several opportunities for optimization are found. Since the wave functions are real, its Fourier transform is Hermitian, so f^{cut} needs to be applied only on half of the domain, which saves arithmetic operations and memory. Usually the FFT software is efficient for input sizes equal to the product of small prime numbers. So it would be beneficial to consider the size of the supercell to be one which corresponds to such a number. For the present implementation FFTw [18] was used. A preliminary experiment with ESSL library on IBM platforms showed that it performs much better than FFTw. The problem associated with the IBM ESSL library is that it has a set of allowed values for the input size N, unlike the FFTw library which can take an arbitrary input size. The initial input grid size is an odd number in our implementation and cannot work with even numbers, so we plan IBM ESSL library in our future versions. However, dealing with this problem is worth the effort because of the high performance of this software.

A supercell is described by a scalar α which is the ratio of the supercell size to that of the original cell. We initially conducted some experiments to determine some optimal value for α. It was noted that $\alpha = 1.7$ gives sufficiently reliable results [19]. The function f^{cut} in Eq. (17), is only dependent on the grid. Since we use the same grid for all the Poisson's equation solutions, we calculate f^{cut} only once before the the coupling matrix assembly loop and use it with matrix multiplication subsequently for all ij. Table 2 summarizes the solution of the Poisson's equation.

```
*Algorithm* **Solution of Poisson's Equation**
Calculate $f^{cut}$ for the 3D grid
Start of Coupling matrix assembly loop
        /*For each ij*/
1. Apply forward Fourier transform
        $\mathcal{F}(\rho_{ij})(\mathbf{k})$ .
2. Multiply by $f^{cut}$
        $\mathcal{F}(\rho_{ij})(\mathbf{k})f^{cut}$.
3. Apply inverse Fourier transform
        $\mathcal{F}^{-1}(\mathcal{F}(\rho_{ij})(\mathbf{k})f^{cut})$
End of Coupling matrix assembly loop
```

Table 2: Solution of the Poisson's equation.

5 Assembling the coupling-matrix

To improve performance, we evaluate Eq. (2) in two steps. First, we compute the kl independent kernel of the integral:

$$\ker_{ij}(\mathbf{r}) = \Big[\Phi_{ij}(\mathbf{r}) + \rho_{ij}(\mathbf{r})\frac{dV_{xc}[\rho(\mathbf{r})]}{d\rho(\mathbf{r})}\Big]. \tag{21}$$

This needs to be done only once per row. Furthermore, in this kernel, $\rho(\mathbf{r})$ is constant throughout the construction of the coupling matrix because it is set at the DFT stage. The analytic form of $V_{xc}[\rho(\mathbf{r})]$, and therefore $dV_{xc}[\rho(\mathbf{r})]/d\rho(\mathbf{r})$, is known. This term is therefore computed once and for all during the setup of the program. Our implementation used the Ceperley-Alder exchange-correlation functional [20]. Following [5] we have slightly modified its analytic parametrization to assure a continuous derivative.

The final step in the construction of a row of the coupling matrix is performing the integration given by

$$K_{ij,kl} = \sum_{\mathbf{r}} \Big[\ker_{ij}(\mathbf{r})\Big]\psi_k(\mathbf{r})\bar{\psi}_l(\mathbf{r}). \tag{22}$$

One could, in principle, use sophisticated integration algorithms to this end. However, we invariably found that for a grid spacing that was small enough to guarantee convergence of the DFT part of the calculation, direct summation was sufficiently accurate.

Finally, it is straightforward to overwrite $K_{ij,kl}$ with $Q_{ij,kl}$ using Eq. (4). Once this has been calculated, computing each element of $K_{ij,kl}$ within a given row is a simple matter of computing $\psi_k(\mathbf{r})\bar{\psi}_l(\mathbf{r})$ and summing over all grid points: It is important to note that because of our use of a real-space grid, all wave functions are real and therefore both $K_{ij,kl}$ and $Q_{ij,kl}$ are inherently symmetric, so we *only compute and store their upper triangular half.*

5.1 Exploiting the bounded support of the wave functions

This section discusses the evaluation of the coefficients of coupling-matrix, given by

$$K_{ij,kl} = \int_\Omega \left(\Psi_i(\mathbf{r})\bar{\Psi}_j(\mathbf{r})\frac{dV_{\text{XC}}(\mathbf{r})}{d\rho(\mathbf{r})} + \Phi_{ij}(\mathbf{r})\right)\Psi_k(\mathbf{r})\bar{\Psi}_l(\mathbf{r})d\mathbf{r}. \tag{23}$$

This computation is performed by a classical integration formula, as discussed in the previous section. The fact that wave functions are functions with nearly bounded support can be exploited to reduce the cost of the summation. Indeed, it is only necessary to sum the intersection of $\mathrm{Supp}(\Psi_k)$ and $\mathrm{Supp}(\Psi_l)$, where $\mathrm{Supp}(\Psi_k) = \{\mathbf{r} \in \Omega, \Psi_k(\mathbf{r}) \neq 0\}$.

$$K_{ij,kl} = 2 \int_{\mathrm{Supp}(\Psi_k)\,\cap\,\mathrm{Supp}(\Psi_l)} \left(\Psi_i(\mathbf{r})\bar{\Psi}_j(\mathbf{r}) \frac{dV_{\mathrm{XC}}(\mathbf{r})}{d\rho(\mathbf{r})} + \Phi_{ij}(\mathbf{r}) \right) \Psi_k(\mathbf{r})\bar{\Psi}_l(\mathbf{r}) d\mathbf{r}.$$

In this paper a rigorous exploitation of this observation was not implemented. An easier approach, from an implementation viewpoint, was adopted in which the decaying property of the charge density is exploited. It is observed that the charge density is essentially zero in a fairly large part of the physical domain Ω. Since the wave functions of all occupied states vanish wherever $\rho(\mathbf{r}) = 0$, it is only necessary to perform the integration where $\rho(\mathbf{r}) > \varepsilon$, where ε is some small number, *i.e.*,

$$K_{ij,kl} = 2 \int_{\{\mathbf{r}\ |\ \rho(\mathbf{r}) > \varepsilon\}} \left(\Psi_i(\mathbf{r})\bar{\Psi}_j(\mathbf{r}) \frac{dV_{\mathrm{XC}}(\mathbf{r})}{d\rho(\mathbf{r})} + \Phi_{ij}(\mathbf{r}) \right) \Psi_k(\mathbf{r})\bar{\Psi}_l(\mathbf{r}) d\mathbf{r}.$$

It is easy and inexpensive to build a list of all the r-points which satisfy the condition $\rho(\mathbf{r}) > \varepsilon$ initially. During integration, a smaller summation is then computed using this list.

5.2 Computational costs and parallelization aspects

Let $kdim$ be the number of transitions included, that is the dimension of the coupling-matrix, and n the number of grid-points. Note that $kdim$ is of order M^2, where M is the number of eigenfunctions. More precisely, we have $kdim$ transitions $\Psi_i\Psi_j$ included, for some (i, j) in $[1, M]^2$. The algorithm computes $kdim$ forward and inverse Fourier transforms, at the cost of $\mathcal{O}(\ kdim\ n \log n)$ operations. Then, it computes the $\mathcal{O}(kdim^2)$ integrals (23) at the cost of n operations each. In conclusion, without any reduction of the box, *i.e.*, of the number of Fourier modes, the number of operations required by the method is of the order,

$$C_1\ kdim^2\ n + C_2\ kdim\ n \log(n).$$

As was already noted, the cost of multipole expansions is negligible compared with the Fourier transforms.

It was remarked in [4] that in the best cases $kdim$ behaves as $\mathcal{O}(n)$ and in the worse cases as $\mathcal{O}(n^2)$ depending of the physical system.

Even after reducing the frequency cut-off in the Fourier expansion, that is reducing n, the order remains large so that parallelization of the method is essential. The coefficients of the coupling-matrix are independent so that parallelization is very simple *via* message-passing. We have parallelized the code so that all processors compute an approximatively equivalent number of terms in the coupling matrix. The principle is to make a decomposition in rows of the upper triangular part of the coupling-matrix such that the sum of the terms computed by each processor is approximatively the same. Of course the larger the matrix is the more effective the parallelization is, see Figure 5. A second point concerning the distribution of the data has to be discussed. Indeed the memory occupied by the wave functions and the coupling matrix is very large so that it is also necessary to distribute the data. More precisely, on each processor one computes and allocates only a part of the coupling matrix so that, except for the root processor, we do not allocate the whole coupling-matrix. Nevertheless, because of the complexity of the computation of the coupling matrix, we create and allocate the totality of the wave functions for each processor.

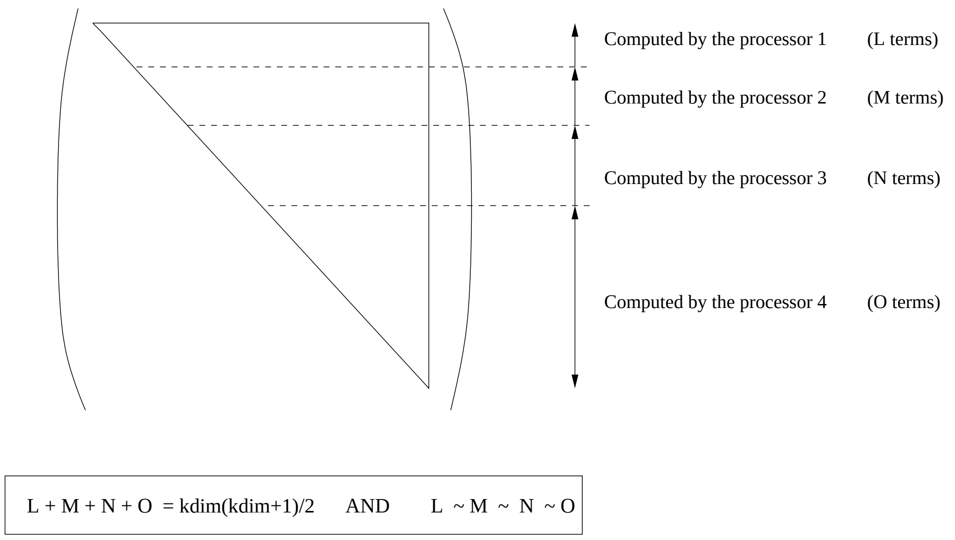
Computed by the processor 1 (L terms)
Computed by the processor 2 (M terms)
Computed by the processor 3 (N terms)
Computed by the processor 4 (O terms)
L + M + N + O = kdim(kdim+1)/2 AND L ~ M ~ N ~ O

Figure 5: Parallelization of the construction of the coupling matrix for 4 processors.

6 Numerical results

In order to validate and compare the methods described in this paper we now present some numerical examples. We compare the optical spectra obtained by the Fourier space method with the real space method. We concentrate on the construction of the coupling matrix (i.e second block in Fig. 1) and compare it to the results and timings obtained from the real-space code [4]. The remaining two blocks (i.e Construction of the wave functions and Computation of the oscillator strengths) are already implemented and we use them as a black-box.

The code is written in C++ and MPI and runs on `Linux`, `Sun`, and the `IBM/SP`. The parallel tests have been executed on the `IBM/SP2` at the Minnesota Supercomputer Institute (MSI) (`www.msi.umn.edu`) of the University of Minnesota. The `IBM/SP2` has 82 nodes, with four 375 Ghz power3 processors, sharing 4 GB of memory. O3 optimization flags are used with the compilation. The FFT routines are implemented using FFTw library [18].

After computing the wave functions, the code performs the following tasks:

- Read the wave functions, the eigenvalues and the occupied states (ε_i, ψ_i, $f(\varepsilon_i)$).
- Compute the density and oscillator strengths at LDA level, B_β (see section 1).
- Compute the exchange-correlation potential V_{xc}.
- Compute the coupling matrix.

Once the coupling matrix K is constructed the dense eigenvalue problem is solved using ScaLAPACK and oscillator strengths are computed.

6.1 Optical Absorption of Hydrogenated Silicon

Hydrogenated silicon clusters are a class of nanostructures obtained by simply cutting a roughly spherical chunk of crystalline silicon, with diameter of up to 20 $\mathring{A}$. The effect of surface states on the absorption spectrum can be reduced by passivating the outer atomic layer. Clusters with

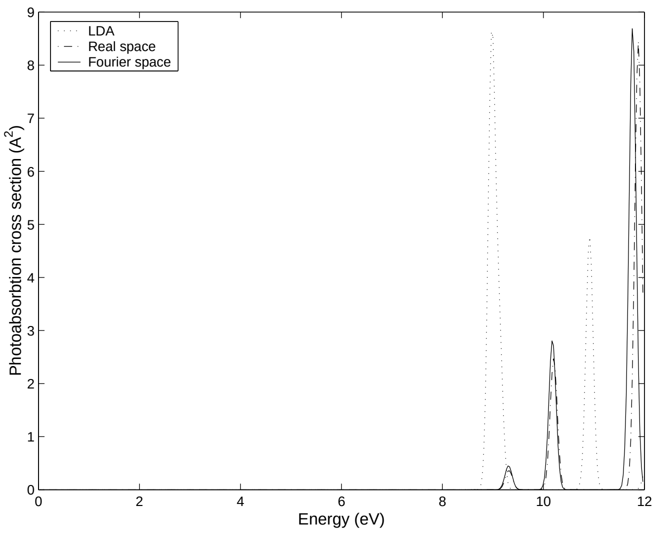

Figure 6: Optical absorption spectra for SiH4.

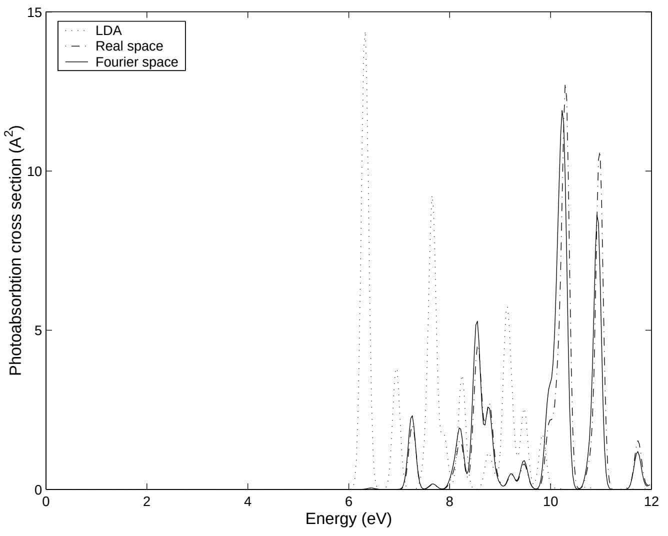

Figure 7: Optical absorption spectra for Si5H12.

sufficiently large diameter retain many of the electronic properties of bulk silicon. By reducing its diameter, we can observe the behavior of various electronic properties such as the fundamental electronic gap and absorption spectrum as function of system size. For example, the LDA absorption threshold (defined as the energy difference between the highest occupied molecular orbital and lowest unoccupied molecular orbital obtained in DFT-LDA) is expected to increase smoothly from the bulk limit, around 1 eV, to the range of 5 to 8 eV for small clusters. The optical spectrum is also blue-shifted as the cluster diameter is reduced [5].

We constructed quasi-spherical silicon quantum dots on the computer from "shells" of equidistant silicon atoms situated around a central atom. The silicon atoms were placed in sites corresponding to the bulk structure of silicon, where each silicon atom is surrounded by four other

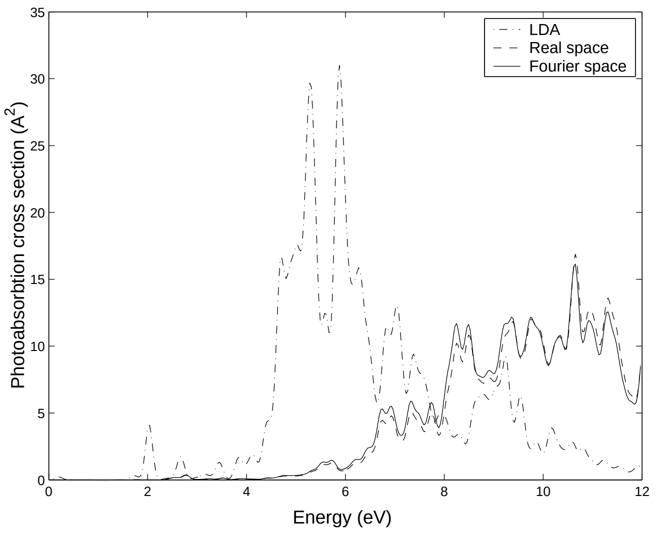

Figure 8: Optical absorption spectra for Si34H36.

silicon atoms possessing tetrahedral symmetry. This results in dangling bonds for silicon atoms on the outer shell, which have "missing" neighbors. Each dangling bond was then eliminated by adding a surface hydrogen atom, thus removing optically active states associated with the surface. Finally, we relaxed the outer layers of the dot by moving the atoms so as to minimize the forces acting on the hydrogen cap atoms.

A selection of optical spectra obtained by computing the TDLDA spectra using Real-space and Fourier-space methods for three hydrogenated quantum dots is shown in Figs. 6–8. It starts with the smallest "quantum dot" one could construct using the above procedure - a SiH_4 molecule based on the central Si atom - and ends with $Si_{34}H_{36}$. For comparison, Figs. 6-8 also show absorption spectra of the same dots computed using conventional (time-independent) LDA, where the spectrum is based on LDA eigenvalue differences weighted by the matrix element between the associated eigenvectors.

The effectiveness of the TDLDA method in calculating the absorption spectra for the above molecules when compared to experiments is explained in detail in [4]. In this section we will use the real-space results as our reference spectrum. It is readily observed that our calculations agree with the real-space code for all the hydrogenated quantum dots considered.

6.2 Timings

Table 3 shows the wall clock runtime for the coupling matrix generation for Si34H36 with the planewave TDLDA code when compared with the real space code. The "Load balanced" time shows the time obtained by optimally load balancing the work between processors, among other improvements. The difference between this time and that of the implementation shown above it, accounts for several improvements made to the code, including but not limited to: (1) dynamic load balancing; (2) Fortran-90 coding instead of C++; and (3) use of BLAS 2 and BLAS 3 routines in assembling the coupling matrix by blocks. Further optimization may be possible and fine-tuning for different machines may lead to additional gains.

Method	Wall-Clock Run Time (hours)
Real Space Code	15:30
PW: Initial Implementation	3:30
PW: Load balanced	1:30

Table 3: Comparison of wall-clock runtime of the parallel TDLDA code (not including the generation of Kohn-Sham eigenvalues and wave functions) using Fourier space and Real Space for the Si34H36 test case running on 8 processors.

7 Conclusion

We have presented a method for computing the coupling matrix which arises in time-dependent density functional theory. The method is based on using a plane wave basis leading to fast solutions of the Poisson equation. A detailed description of the parallel implementation was given. The main advantage of this approach versus the use of preconditioned Conjugate Gradient method used previously [4] for handling the Poisson problems, is that it leads naturally to fast-solutions of the Poisson equations. The issue of the influence of near-by cells when solving Poisson's equation by FFT for non-periodic systems, can be handled in a number of ways, including the cut-off approach described in [14]. Though this increases the cost slightly, this increase is rather limited for sufficiently localized systems.

Among other strategies which lead to a significant reduction in the overall computational time is the use of local support for efficient integration. The numerical results indicate that the method is far superior to the real-space code.

Acknowledgments. We would like to thank Manuel Alemany, Russ Burdick, and Leeor Kronik for useful discussions and for their help with the DFT and TDDFT codes.

References

[1] See, e. g., M. L. Cohen and J. R. Chelikowsky, *Electronic Structure and Optical Properties of Semiconductors*, 2nd ed. (Springer-Verlag, Berlin 1989); J. R. Chelikowsky and M. L. Cohen, in *Handbook on Semiconductors*, T. S. Moss, Ed. , 2nd Edition (Elsevier, Amsterdam, 1992); W. E. Pickett, Comput. Phys. Reports **9**, 115 (1989); M. C. Payne, M. P. Teter, D. C. Allan, T. A. Arias, and J. D. Joannopulos, Rev. Mod. Phys. **64**, 1045 (1992).

[2] E. K. U. Gross, J. F. Dobson, and M. Petersilka, in *Density Functional Theory*, edited by R. F. Nalewajski, (Springer-Verlag, Berlin, 1996), p. 81; M. Petersilka, U. J. Gossmann, and E. K. U. Gross, Phys. Rev. Lett. **76**, 1212 (1996).

[3] M. E. Casida, in *Recent Advances in Density-Functional Methods*, Part I, edited by D. P. Chong (World Scientific, Singapore, 1995), p. 155; M. E. Casida, in *Recent Developments and Applications of Modern Density Functional Theory*, edited by J. M. Seminario (Elsevier, Amsterdam, 1996), p. 391.

[4] W. R. Burdick, Y. Saad, L. Kronik, I. Vasiliev, M. Jain, and J. R. Chelikowsky, "Parallel Implementation of Time-Dependent Density Functional Theory", CPC, **156**, 22-42 (2003).

[5] I. Vasiliev, S. Öğüt, and J. R. Chelikowsky, Phys. Rev. B **65**, 115416 (2002).

[6] R. E. Stratmann, G. E. Scuseria, and M. J. Frisch, J. Chem. Phys. **109**, 8218 (1998); R. Bauernschmitt and R. Ahlrichs, Chem. Phys. Lett. **256**, 454 (1996).

[7] S. J. A. van Gisbergen, C. Fonseca Guerra, and E. J. Baerends, J. Comp. Chem. **21**, 1512 (2000); S. J. A. van Gisbergen, J. G. Snijders, and E.J. Baerends, Comp. Phys. Commun. **118**, 119 (1999).

[8] J. R. Chelikowsky, N. Troullier, and Y. Saad, Phys. Rev. Lett **72**, 1240 (1994); J. R. Chelikowsky, N. Troullier, K. Wu, and Y. Saad, Phys. Rev. B **50**, 11355 (1994).

[9] Y. Saad, A. Stathoupolos, J. R. Chelikowsky, K. Wu, and S. Öğüt, BIT **36**, 563 (1996).

[10] I. Vasiliev, PhD Thesis, University of Minnesota (2001).

[11] I. Vasiliev, S. Öğüt, and J. R. Chelikowsky, Phys. Rev. Lett **86**, 1813 (2001).

[12] A. Stathopoulos, S. Öğüt, Y. Saad, J. R. Chelikowsky, and H. Kim, Comput. Sci. Eng., **2**, 19 (2000).

[13] L. S. Blackford, J. Choi, A. Cleary, E. D'Azevedo, J. Demmel, I. Dhillon, J. Dongarra, S. Hammarling, G. Henry, A. Petitet, K. Stanley, D. Walker, and R. C. Whaley. *ScaLAPACK Users Guide* (Society for Industrial and Applied Mathematics, Philadelphia, 1997).

[14] A. Castro , A. Rubio, and M.J. Stott, Canadian J. of Phys. **81**, 1151 (2003).

[15] G. Onida, L. Reining, R. W. Godby, R. Del Sole, and W. Andreoni, Phys. Rev. Lett. **75**, 818 (1995).

[16] M. R. Jarvis, I. D. White, R. W. Godby, and M. C. Payne, Phys. Rev. B **56**, 14972 (1997).

[17] G. Lauritsch and P.-G. Reinhard, International Journal of Modern Physics, **5** (1994).

[18] M. Frigo and S. G. Johnson, FFTW: An adaptive software architecture for the FFT, Proc. 1998 IEEE Intl. Conf. Acoustics Speech and Signal Processing, **3**, 1381 (1998).

[19] S. Gowda. Master's Thesis, University of Minnesota (2004).

[20] D. M. Ceperley and B. J. Alder, Phys. Rev. Lett. **45**, 566 (1980).