Real-time adaptive finite element solution of time-dependent Kohn–Sham equation

Gang Bao a,b, Guanghui Hu c,d,*, Di Liu b

a Department of Mathematics, Zhejiang University, Hangzhou 310027, China
b Department of Mathematics, Michigan State University, East Lansing, MI 48824, USA
c Department of Mathematics, University of Macau, Macau S.A.R., China
d UM Zhuhai Research Institute, Zhuhai, Guangdong, China

Article history:
Received 21 May 2014
Received in revised form 8 October 2014
Accepted 26 October 2014
Available online 30 October 2014

Keywords:
Time-dependent Kohn–Sham
Finite element methods
Crank–Nicolson
Mesh adaptive methods
Multigrid for complex system

ABSTRACT

In our previous paper (Bao et al., 2012 [1]), a general framework of using adaptive finite element methods to solve the Kohn–Sham equation has been presented. This work is concerned with solving the time-dependent Kohn–Sham equations. The numerical methods are studied in the time domain, which can be employed to explain both the linear and the nonlinear effects. A Crank–Nicolson scheme and linear finite element space are employed for the temporal and spatial discretizations, respectively. To resolve the trouble regions in the time-dependent simulations, a heuristic error indicator is introduced for the mesh adaptive methods. An algebraic multigrid solver is developed to efficiently solve the complex-valued system derived from the semi-implicit scheme. A mask function is employed to remove or reduce the boundary reflection of the wavefunction. The effectiveness of our method is verified by numerical simulations for both linear and nonlinear phenomena, in which the effectiveness of the mesh adaptive methods is clearly demonstrated.

© 2014 Elsevier Inc. All rights reserved.

1. Introduction

In 1984, Runge and Gross [24] discovered the one-to-one correspondence between time-dependent one-body density \( n(\mathbf{x}, t) \) and time-dependent one-body potentials \( v_{\text{ext}}(\mathbf{x}, t) \) in a system under certain quite general conditions, for a given initial state. Based on Runge–Gross theorem, the time-dependent electron density becomes a fundamental variable of the system, which can generate all other physical properties of the system. Analogous to the Kohn–Sham system, the time-dependent electron density can be obtained by solving a fictitious non-interacting system, the time-dependent Kohn–Sham system, instead of the original interacting one. Consequently, the dimension of the problem may be dramatically reduced, and the theoretical and numerical study of large electronic system becomes possible.

Since then, a great deal of studies have been conducted on developing numerical methods for TDKS equation. In the perturbative regime, the time-dependent linear response theory (TDLRT) is one of the most popular methods to study the response of the system to an external potential \( \delta v_{\text{ext}}(\mathbf{x}', t') \). For the method, once the linear density response function \( \chi(\mathbf{x}, \mathbf{x}', t - t') \) is obtained, the first-order response of all properties derivable from the density with respect to any scalar field can be calculated smoothly. Such linear density response function can be derived straightforwardly from the ground-state of the system. Hence, the TDLRT is widely used in the calculations of photoabsorption spectra, the excited state, etc.,

* Corresponding author.
E-mail address: huyangemei@gmail.com (G. Hu).

http://dx.doi.org/10.1016/j.jcp.2014.10.052
0021-9991/© 2014 Elsevier Inc. All rights reserved.
of the given electronic structures. In the cases that the higher-order components in the external potential can no longer be omitted, the time-dependent higher-order response function can be introduced with an analogous derivation to the linear case. Similar to the TDLRT, this higher-order density response function gives the desired high-order response of all properties derivable from the density. To date, the TDLRT and related higher-order methods have been widely used in the computational chemistry community, and have been realized in a lot of software such as Siesta [28], GPAW [21,34], and Octopus [5]. However, the drawback of the TDLRT is also quite apparent. From the derivation of the linear and higher-order density response functions, we know that the ground-state perturbation of the system is required. This means that the variation of the time-dependent electron density cannot be very far away from the ground-state electron density, or else the reliability would be destroyed. This drawback limits the applications of TDLRT on the study of the non-perturbative regime of a given system where the time-dependent electron density could vary dramatically in a large domain. Another issue about the TDLRT, from the computational point of view, is that a high number of unoccupied orbitals needs to be calculated to guarantee the accuracy and reliability of TDLRT, which increases the difficulty of the convergence on the ground-state calculation, especially for a large and complex electronic structure system.

Developing numerical methods for real-time time-dependent Kohn–Sham equation (RT-TDKS) can resolve the above mentioned issues of TDLRT effectively. In a RT-TDKS simulation, the system is evolved in the time domain by solving TDKS equation, and the time-dependent electron density $n(\vec{x},t)$ is recorded at each time instant. Then all time-dependent properties of the system can be derived from $n(\vec{x},t)$, according to the Runge–Gross theorem. From the description above, we can see that there is no restriction on the time-dependent external potential $v_{ext}(\vec{x},t)$, and the item that we need to record, say, the time-dependent electron density $n(\vec{x},t)$, only depends on the occupied orbitals. Furthermore, both linear effect such as photoabsorption spectra and nonlinear effect such as high harmonic generation can be numerically studied by using the above process. Therefore, it appears that RT-TDKS methods are more powerful than TDLRT methods. However, there are several challenges on developing effective numerical methods for RT-TDKS equation. From the model point of view, so far, an effective and reliable time-dependent exchange-correlation potential $v_{xc}[n](\vec{x},t)$ is still unavailable. Theoretically, this quantity is unknown analytically, and is global in both time and space. In practice, the approximations such as the adiabatic local density approximation (ALDA), are used in the simulation. Although ALDA is local in both time and space, and has good performance in the perturbative simulations, its performance in the non-perturbative simulations is unacceptable, especially when the external field is sufficiently strong. Hence, to develop a generally effective local approximation of $v_{xc}[n](\vec{x},t)$ is crucial for the study of RT-TDKS equation. On the other hand, from the computational point of view, more computational resources are needed by RT-TDKS methods compared with TDLRT methods. Unlike TDLRT methods which only focus on an interested frequency of the response of the system, RT-TDKS methods can produce the response of the system at all frequencies in a single simulation. It makes RT-TDKS methods more demanding on the memory and CPU time. In addition, since the nonlinear property of the equation, and the fact that the time propagator of TDKS equation is actually a time-ordered exponential operator, to develop efficient numerical methods for RT-TDKS equation is quite nontrivial. In spite of these challenges, the advantages of RT-TDKS methods have already motivating a lot of research on related fields.

There have been a lot of pioneer works on developing numerical methods for RT-TDKS equation. The related algorithms have also been realized in several libraries, such as QBox [14,26] where the plane-wave method is used, Siesta [28] where linear combination of atomic orbitals (LCAO) basis sets are adopted, GPAW [21,34] where the grid-based projector-augmented wave method is used, and Octopus [5] where the finite difference method is used. In these libraries, the frameworks of the algorithm for RT-TDKS are well-established, and excellent results are presented from simulating photoabsorption spectra of the electronic structures to calculating nonlinear phenomena such as high harmonic generation and multiphoton ionization. Although these existing methods and libraries have been widely applied, there is still one common potential issue on their practical applications, i.e., the methods are all rely on the regularity of the domain and/or the structured property of the mesh. In addition, various boundary conditions could appear in the simulation with complex configuration. To formulate boundary condition is still a nontrivial challenge for plane-wave method. As one kind of real-space methods, the finite element methods have the potential to deliver more robust numerical solvers on handling various boundary conditions and domain with complex geometry, even compared with finite difference methods mentioned above which is another kind of real-space methods. In addition, since the finite element methods have been widely used in other research fields such as computational fluid dynamics, computational electromagnetics, etc., there have been a lot of mature techniques for developing efficient methods such as a posteriori error estimations, multigrid methods, etc., which can benefit the development of the efficient methods for RT-TDKS equation. People may refer to [23] for the finite element methods in the electronic structure calculations. Although the finite element methods have been developed for the ground-state calculations of the electronic system in [29,35,2,1,3,10,17], etc., little is known about the application of finite element methods for the RT-TDKS. In [27], a finite element method solver for time-dependent and stationary Schrödinger equations is proposed. However, only one-dimensional problem is considered there. In [6], the finite-difference/finite-element time–space discretization is utilized for the TDKS. Although the applications for the three-dimensional carbon nanotube is presented there, only one-dimensional problem is actually solved with the mode approach. The full three-dimensional discretization for the TDKS equation may be very complicated and CPU-time demanding in the implementation. However, it is meaningful for the practical applications, especially for the system with a general structure.

For three dimensional simulations, Lehtovaara et al. extend their all-electron finite element solver for the Kohn–Sham equation [17] to the RT-TDKS case [18]. Excellent numerical results can be observed there. To generate an appropriate mesh, an elegant method is introduced in [17,18], say, the mesh is created by merging highly nonuniform but symmetric atomic
meshes to a global, nonuniform, and unstructured mesh. Similar mesh generation methods can also be observed from [29,25]. The meshes generated by these methods can handle ground-state calculations, and the perturbative simulations very well. However, there is a difficulty for using such fixed mesh in non-perturbative simulations since the time-dependent electron density could change dramatically during the simulation. As a result, a sufficiently dense mesh in a sufficiently large domain around the nuclei might be necessary for a reliable simulation. In some case, such domain could be very large, and it is a dynamic process for the variation of the electron density during the simulation. Hence, from the computational point of view, mesh adaptive methods, including the local refinement and coarsening, offer a better choice rather than a global dense mesh. In the simulation, the mesh density is adjusted dynamically according to the numerical results with the mesh adaptive methods, and computational resource can be potentially saved. Although there have been a lot of works on developing mesh adaptive methods for ground-state calculations [10,35,8], rare of them is related to the RT-TDKS simulations.

In this paper, we develop an adaptive finite element method for RT-TDKS simulations, based on our previous work in [1]. There are two main components in a numerical discretization for the TDKS equation, the spatial discretization and the temporal discretization. For the spatial discretization of the TDKS equation, we follow [1] to employ the linear finite element method. For the temporal discretization of the TDKS equation, a second-order semi-implicit Crank–Nicolson (SOICN) scheme is adopted, which conserves the unitary and the time-reversal symmetry of the time propagator well. Since the TDKS equation is a complex equation, the linear system derived is actually a complex-valued one. We will study this complex-valued system by separating the wavefunction into the real part and the imaginary part, and the final matrix derived by this way is a large block one. Since this block matrix is nonsymmetric, we use the bi-conjugate gradient (BiCG) method to solve the linear system. Other than the nonlocal part of the pseudopotential in Kleinman–Bylander form, all four submatrices of the large block matrix actually have the same sparsity pattern. Based on this observation, an algebraic multigrid (AMG) method is developed in this paper for efficiently solving the block matrix. The numerical results show that the derived AMG solver works very well. As discussed above, the mesh adaptive method could potentially improve the efficiency of the RT-TDKS simulations. In this paper, we follow our previous work in [1] to use the hierarchical geometry tree (HGT) for the mesh data, and introduce a heuristic a posteriori error estimator to handle the mesh refinement and coarsening. With the mesh adaptive methods, meshes with a large diameter can be used close to the boundary of the domain. Consequently, a sufficiently large computational domain can be used for a RT-TDKS simulation, without significantly increasing the number of grid points. Furthermore, our solution will not be very sensitive to the usage of the absorbing boundary conditions. In our simulation, the simple mask function method [19] serves as the absorbing boundary condition.

This paper is arranged as follows. In the next section, the TDKS model is briefly summarized. Then the details of the numerical discretization of the TDKS equation are presented in Section 3. We discuss several numerical issues in Section 4. In Section 5, a variety of numerical simulations are presented to show the effectiveness and reliability of our numerical method. Finally, we conclude the research and discuss some future directions.

2. Time-dependent Kohn–Sham equation

The time-dependent Schrödinger equation can be read as

\[ i \frac{\partial}{\partial t} \Psi = H \Psi, \]

where \( \Psi = \psi(\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_N, t) \) is the wavefunction which depends on the positions of \( N \) electrons at time \( t \), \( i \) is the imaginary unit, and \( H \) is the Hamiltonian operator. It is obviously too expensive to solve (1) directly because of the high dimensionality of \( \Psi \). Consequently, to reduce the dimension of Eq. (1) has great meaning for the practical applications.

The Runge–Gross theorem [24] asserts that all observables can be calculated with the knowledge of the one-body density [11]. Analogous to the idea of Kohn and Sham [16], the time-dependent density of the system can be obtained by solving the following time-dependent Kohn–Sham (TDKS) equation

\[ i \frac{\partial}{\partial t} \psi_j(\vec{x}, t) = -\frac{\nabla^2}{2} + v_{KS}[n](\vec{x}, t) \psi_j(\vec{x}, t), \quad j = 1, 2, \ldots, N \]

where \( \psi_j(\vec{x}, t) \) is the \( j \)-th wavefunction, and \( v_{KS}[n](\vec{x}, t) \) is the Kohn–Sham potential. The density of the original interacting system \( n(\vec{x}, t) \), and the time-dependent Kohn–Sham wavefunctions \( \psi_j(\vec{x}, t) \) have the relationship

\[ n(\vec{x}, t) = \sum_{j=1}^{N} |\psi_j(\vec{x}, t)|^2. \]

Compared with the Schrödinger equation (1), the spatial dimension of TDKS equation (2) is 3, which is a huge reduction compared with 3N of TDSE (1). According to Runge–Gross theorem, once the time-dependent electron density of the system \( n(\vec{x}, t) \) is known, all other physical variables of the system can be generated from \( n(\vec{x}, t) \), explicitly or implicitly.

There are three components in the Kohn–Sham potential \( v_{KS}[n](\vec{x}, t) \): The external potential \( v_{ext}(\vec{x}, t) \), the Hartree potential \( v_H[n](\vec{x}, t) \), and the exchange-correlation potential \( v_{xc}[n](\vec{x}, t) \), defined as

\[ v_{KS}[n](\vec{x}, t) = v_{ext}(\vec{x}, t) + v_H[n](\vec{x}, t) + v_{xc}[n](\vec{x}, t). \]
In the ground-state calculations, \( v_{\text{ext}} \) only accounts for the Coulomb attraction between the electrons and the nuclei. For the time-dependent simulations, there might be an extra perturbation field acting on the system in \( v_{\text{ext}} \). For example, a laser field \( v_{\text{laser}} \). The Hartree potential, \( v_H[n](\vec{x}, t) = \frac{n(\vec{x}, t)}{|\vec{x} - \vec{x}'|} \), accounts for the electrostatic interaction between electrons. To efficiently generate the Hartree potential, the following Poisson problem

\[
-\nabla^2 v_H = 4\pi n(\vec{x}, t),
\]
\[
v_H = 0, \quad |\vec{x}| \to \infty,
\]

is solved with efficient numerical methods such as multigrid methods instead of evaluating the original integral.

The exchange-correlation potential \( v_{\text{xc}}[n](\vec{x}, t) \) accounts for all the nontrivial many-body effects in the system. Similar to the ground-state calculation, the time-dependent \( v_{\text{xc}}[n](\vec{x}, t) \) has no analytical expression, hence the approximation becomes necessary. In ground-state calculation, \( v_{\text{xc}} \) is written as a functional derivative of the exchange-correlation energy, which follows from a variational derivation of the Kohn–Sham equation starting from the total energy. However, this is not the case for time-dependent density functional theory (TDDFT) because of the causality [13]. By using Keldish formalism, van Leeuwen [31] introduced a new action function \( \tilde{A} \), and its exchange-correlation part \( \tilde{A}_{\text{xc}} \) had the following relationship with time-dependent \( v_{\text{xc}} \)

\[
v_{\text{xc}}[n](\vec{x}, t) = \frac{\delta \tilde{A}_{\text{xc}}}{\delta n(\vec{x}, \tau)} n(\vec{x}, t),
\]

where \( \tau \) is the Keldish pseudo-time. In the simulation, the simplest adiabatic local density approximation (ALDA) is adopted, and the exchange-correlation potential \( v_{\text{xc}}[n](\vec{x}, t) \) is given by

\[
v_{\text{ALDA}}[n](\vec{x}, t) = v_{\text{ALDA}}[n](\vec{x}),
\]

where \( v_{\text{ALDA}}[n](\vec{x}) \) is the local density approximation. With this approximation, the \( v_{\text{ALDA}}[n](\vec{x}) \) is obviously local both in time and in space, and hence can be efficiently implemented. However, \( v_{\text{ALDA}} \) is only proper for calculating the ground-state of the electronic structures. Consequently, ALDA is reasonable only for simulating a system with weak external field. In a simulation with sufficiently large external potential, such term in the Hamiltonian can be removed.

3. Numerical discretization of the TDKS equation

The numerical discretization of the TDKS equation includes the temporal and spatial discretizations. For the spatial discretization, we follow [1] to use the linear finite element space. The temporal discretization is more complicated, the derivation is given in the following subsection.

3.1. Temporal discretization

There are two key issues in the time discretization for the TDKS equation: The time propagator and the size of the time step. In this section, we first briefly summarize the properties the time propagator should satisfy. Then the time propagator used in this paper is introduced. For the simplification, the spatial variable is dropped in this subsection.

Define a linear time propagator \( U(t, t_0) \) for the Schrödinger equation (1) such that the wavefunction at an instant \( t \), say \( \phi(t) \), can be expressed as

\[
\phi(t) = U(t, t_0)\phi(t_0)
\]

where \( \phi(t_0) \) is given as the initial state of the wavefunction at \( t_0 \). Then the main task in this section is to find out the time propagator \( U(t, t_0) \), which is determined by the following Schrödinger-like equation

\[
\frac{\partial}{\partial t} U(t, t_0) = H(t)U(t, t_0).
\]

By introducing a time-ordering operator \( T \), the formal solution of above equation can be expressed as [20]

\[
U(t, t_0) = T \exp \left\{-i \int_{t_0}^{t} H(\tau) d\tau \right\}
\]

Although this is a concise mathematical expression for \( U(t, t_0) \), it is too complicated to use in the practical problems. However, it can be verified easily that the time-propagator (9) satisfies the following three properties

- Unitary, \( U^*(t, t_0) = U^{-1}(t, t_0) \),
- Time-reversal symmetry, \( U^{-1}(t, t_0) = U(t_0, t) \),
- \( U(t_2, t_0) = U(t_2, t_1)U(t_1, t_0) \).
where \( t_0, t_1, t_2 \) are three instants, and \( U^* \) is the conjugate of \( U \). The first property preserves the probability of the wavefunction with the time evolution, which is very important for the stability of the numerical method. The second property implies that we can obtain \( \phi(t + \Delta t/2) \) either by propagating \( \phi(t) \) forward \( \Delta t/2 \) or by propagating \( \phi(t + \Delta t) \) backward \( \Delta t/2 \). Theoretically, to obtain the wavefunction at any instant \( t \), (7) can be used directly once the linear operator \( U(t_0,t_0) \) is obtained exactly. However, in the implementation, the largest size of the time step \( \Delta t_{\text{max}} \) is limited by the following formula

\[
\Delta t_{\text{max}} \approx 1/\omega_{\text{max}}.
\]

where \( \omega_{\text{max}} \) stands for the maximum frequency in the field. Consequently, if \( \Delta t = |t - t_0| \) in (7) is too large, which is unfortunately the case for most practical problems, the \( \phi(t) \) from (7) is unreliable. The last property above helps us to avoid this difficulty. With this property, we can use the partition \( \{[t_i, t_{i-1}]\}_{i=1}^{n} \) of the interval \( t = [t_0, t_n] \) to express \( \phi(t) \) as

\[
\phi(t) = \prod_{i=1,2,...,n} U(t_i, t_{i-1}) \phi(t_0).
\]

In this case, \( \max|t_i - t_{i-1}|, \quad i = 1, 2, \ldots, n \leq \Delta t_{\text{max}} \) will guarantee the stability and reliability of the numerical methods.

The remaining task is to approximate the propagator \( U(t_0,t_0) \) in (9) accurately and efficiently. Note that there is an integral appeared in the expression \( U(t_0,t) \) with the integration interval \([t_0, t]\). To numerically approximate this integral, it is inevitable to evaluate \( H(\tau) \) for \( \tau \in (t_0, t] \). In fact, we could use rectangle rule with the left endpoint of the interval to avoid the evaluation of \( H(\tau) \) for \( \tau \in (t_0, t] \). However, the derived time propagator does not satisfy the properties mentioned above. Since it is unknown a priori for \( H(\tau) \), the extrapolation technique is a reasonable choice for the approximation. For example, the extrapolation is implemented in Octopus [5] which delivered excellent numerical results. One issue of the extrapolation method is the requirement on the storage. To extrapolate an \( H(\tau) \) on \([t_0, t]\), a number of historical records for the Hamiltonian before \( t_0 \) are needed. The better approximation for \( H(\tau) \) we want, the more records we need. Hence, using the extrapolation method will make us to face the problem of balancing the numerical accuracy and the algorithm efficiency. Besides the extrapolation method, we can also resort to the predictor–corrector method to avoid using the historical records.

Before introducing the predictor–corrector method, we first give the approximation of the time propagator \( U(t + \Delta t, t) \) in this paper. The derivation of the approximation propagator is quite simple. First, the trapezoid rule is used to approximate the integral \( \int_{t}^{t+\Delta t} H(\tau) \, d\tau \), then the Padé approximation \( P_{1,1} = \frac{(2 + x)}{(2 - x)} \) gives the following formula

\[
U(t + \Delta t, t) \approx \frac{1 - \frac{1}{4} \Delta t(H(t) + H(t + \Delta t))}{1 + \frac{1}{4} \Delta t(H(t) + H(t + \Delta t))}.
\]

It is trivial to verify that this approximation satisfies the unitary and time-reversal symmetry. In the above formula, \( H(t) \) is known, while \( H(t + \Delta t) \) depends on the desired quantity \( \phi(t + \Delta t) \) in the calculation, hence is not known a priori. The following predictor–corrector method is adopted to effectively evaluate this quantity. First, a rough \( \tilde{\phi}(t + \Delta t) \) is evaluated with the formula

\[
\tilde{\phi}(t + \Delta t) = \tilde{U}(t + \Delta t, t)\phi(t), \quad \tilde{U}(t + \Delta t, t) = \frac{1 - \frac{1}{2} \Delta t H(t)}{1 + \frac{1}{2} \Delta t H(t)}.
\]

Then a rough \( \tilde{H}(t + \Delta t) \) is available from \( \tilde{\phi}(t + \Delta t) \). We use \( \tilde{H}(t + \Delta t) \) instead of the exact \( H(t + \Delta t) \) in (11) to calculate a new \( \tilde{\phi}(t + \Delta t) \), then a new \( \tilde{H}(t + \Delta t) \). We repeat this process till a consistent status is arrived.

The flow chart of the predictor–corrector method is given by Algorithm 1 below.

**Algorithm 1: The predictor–corrector process.**

**Data:** \( \phi(t), \Delta t, \text{MITER}, \text{TOL} \)

**Result:** \( \phi(t + \Delta t) \)

Evaluate the Hamiltonian \( H(t) \) from \( \phi(t) \);
Let \( \text{iter} = 0 \), and appropriately initialize \( \tilde{H}(t + \Delta t) \);
Use (12) to get \( \tilde{\phi}(t + \Delta t) \), and \( \tilde{H}(t + \Delta t) \);

while \( \text{iter} < \text{MITER} \) and \( ||\tilde{H}(t + \Delta t) - \tilde{H}(t + \Delta t)|| > \text{TOL} \) do
\( \tilde{H}(t + \Delta t) = \tilde{H}(t + \Delta t) \);
Update \( \tilde{\phi}(t + \Delta t) \) using (11) and \( \tilde{H}(t + \Delta t) \);
Update \( \tilde{H}(t + \Delta t) \) from \( \tilde{\phi}(t + \Delta t) \);
\( \text{iter}++ \);
end

Let \( \phi(t + \Delta t) = \tilde{\phi}(t + \Delta t) \);
The numerical accuracy may benefit from a small Tol and a large MITER in the above algorithm. However, the demand for the computational resource increases in the meantime. In our simulation, MITER is set as 2, which generates quite good results.

As we mentioned, the maximum of the size of the time step \( \Delta t_{\text{max}} \) and the maximum of the frequency \( \omega_{\text{max}} \) have the following inversely proportional relationship \( \Delta t_{\text{max}} \approx 1/\omega_{\text{max}} \), while for a real-space discretization, \( \omega_{\text{max}} \) is determined by the mesh spacing \( \Delta h \) as \( \omega_{\text{max}} = 2\pi^2/\Delta h^2 \). Hence \( \Delta t_{\text{max}} \) can be obtained by

\[
\Delta t_{\text{max}} = \frac{\Delta h^2}{2\pi^2}.
\]

(13)

To guarantee the stability of the algorithm, it is reasonable to set the size of the time step as \( \Delta t = \alpha \Delta t_{\text{max}}, \alpha \in (0, 1] \).

3.2. Spatial discretization

For a Kohn–Sham system which contains \( N \) wavefunctions, \( \phi_j, j = 1, 2, \ldots, N \), each of these wavefunctions obeys (2). In the following description of the spatial discretization, the subscript of the wavefunction will be dropped for the simplification. We will use \( \phi \) as a wavefunction. The size of the time step is denoted by \( \Delta t \).

In our simulation, the time propagator (11) is adopted, and (2) becomes

\[
\phi(\tilde{x}, t + \Delta t) = \frac{1 - \frac{1}{4} \Delta t (H(\tilde{x}, t) + H(\tilde{x}, t + \Delta t))}{1 + \frac{1}{4} \Delta t (H(\tilde{x}, t) + H(\tilde{x}, t + \Delta t))} \phi(\tilde{x}, t).
\]

(14)

Note that the wavefunction \( \phi \) is a complex-valued variable, which means that the derived algebraic system from the above equation is a complex-valued one. In the simulation, we study Eq. (14) by separating the real part and the imaginary part of the wavefunction. Specifically, assume that \( \phi = \phi^{\text{real}} + i\phi^{\text{imag}} \), then the above equation can be reformulated as the following system

\[
\begin{align*}
\phi^{\text{real}}(\tilde{x}, t + \Delta t) &= \phi^{\text{real}}(\tilde{x}, t) - \frac{1}{4} \Delta t H(\tilde{x}, t) \phi^{\text{imag}}(\tilde{x}, t + \Delta t) \\
&\quad + \frac{1}{4} \Delta t H(\tilde{x}, t + \Delta t) \phi^{\text{imag}}(\tilde{x}, t + \Delta t) \\
\phi^{\text{imag}}(\tilde{x}, t + \Delta t) &= \phi^{\text{imag}}(\tilde{x}, t) - \frac{1}{4} \Delta t H(\tilde{x}, t) \phi^{\text{real}}(\tilde{x}, t + \Delta t) \\
&\quad + \frac{1}{4} \Delta t H(\tilde{x}, t + \Delta t) \phi^{\text{real}}(\tilde{x}, t + \Delta t) \quad \forall \psi \in \mathcal{V}
\end{align*}
\]

(15)

Denote

\[
\mathcal{V} = \{ \psi : \psi \in H^1(\Omega), \psi|_{\partial \Omega} = 0 \}
\]

where \( H^1(\Omega) = W^1_2(\Omega) \) is the traditional Sobolev space, and \( \Omega \) is the physical domain of the electronic system. It is noted that \( \psi|_{\partial \Omega} = 0 \) in the definition of \( \mathcal{V} \) is not an appropriate condition, since the wavefunction could reach the boundary of \( \Omega \) with the influence of the external perturbation. Simply zero-Dirichlet boundary condition will result in the numerical reflection of the wavefunction which may affect the reliability of the simulation. This issue can be resolved by introducing the absorbing boundary condition, which will be describe detailedly in the next section. With the space \( \mathcal{V} \), the variational form of (15) can be read as: Find \( \phi^{\text{real}}, \phi^{\text{imag}} \in \mathcal{V} \) such that

\[
\begin{align*}
\phi^{\text{real}}(\tilde{x}, t + \Delta t), \psi &= \frac{1}{4} \Delta t H(\tilde{x}, t + \Delta t) \phi^{\text{imag}}(\tilde{x}, t + \Delta t), \psi \\
&\quad - \frac{1}{4} \Delta t H(\tilde{x}, t + \Delta t) \phi^{\text{real}}(\tilde{x}, t + \Delta t), \psi \\
&\quad + \frac{1}{4} \Delta t H(\tilde{x}, t) \phi^{\text{imag}}(\tilde{x}, t), \psi \\
&\quad - \frac{1}{4} \Delta t H(\tilde{x}, t) \phi^{\text{real}}(\tilde{x}, t), \psi \\
\phi^{\text{imag}}(\tilde{x}, t + \Delta t), \psi &= \frac{1}{4} \Delta t H(\tilde{x}, t + \Delta t) \phi^{\text{real}}(\tilde{x}, t + \Delta t), \psi \\
&\quad - \frac{1}{4} \Delta t H(\tilde{x}, t + \Delta t) \phi^{\text{imag}}(\tilde{x}, t + \Delta t), \psi \\
&\quad + \frac{1}{4} \Delta t H(\tilde{x}, t) \phi^{\text{real}}(\tilde{x}, t), \psi \\
&\quad - \frac{1}{4} \Delta t H(\tilde{x}, t) \phi^{\text{imag}}(\tilde{x}, t), \psi \\
&\quad \forall \psi \in \mathcal{V}
\end{align*}
\]

(17)

Suppose that \( \Omega \) is divided into a set of tetrahedron \( T \) with \( T_i \) as its element and \( X_i \) as the vertex of \( T_i \). Let \( V_h \subset \mathcal{V} \) be the linear finite element space which is built on \( T \). The variational form (17) can be approximated as: Find \( \phi^{\text{real}}_h, \phi^{\text{imag}}_h \in V_h \) such that
\[
\begin{align*}
&\phi_h^{\text{real}}(\tilde{x}, t + \Delta t), \psi = -\frac{1}{4}\Delta t \ H(\tilde{x}, t) + H(\tilde{x}, t + \Delta t) \ \phi_h^{\text{imag}}(\tilde{x}, t + \Delta t), \psi \\
&= \phi_h^{\text{real}}(\tilde{x}, t), \psi + \frac{1}{4}\Delta t \ H(\tilde{x}, t) + H(\tilde{x}, t + \Delta t) \ \phi_h^{\text{imag}}(\tilde{x}, t), \psi, \quad \forall \psi \in V_h \\
&\phi_h^{\text{imag}}(\tilde{x}, t + \Delta t), \psi + \frac{1}{4}\Delta t \ H(\tilde{x}, t) + H(\tilde{x}, t + \Delta t) \ \phi_h^{\text{real}}(\tilde{x}, t + \Delta t), \psi \\
&= \phi_h^{\text{imag}}(\tilde{x}, t), \psi - \frac{1}{4}\Delta t \ H(\tilde{x}, t) + H(\tilde{x}, t + \Delta t) \ \phi_h^{\text{real}}(\tilde{x}, t), \psi, \quad \forall \psi \in V_h
\end{align*}
\]

So far, we have arrived at the full discretization of TDKS equation. We now discuss how to solve the derived linear system efficiently. According to the structure of (18), the matrix in the final linear system is obviously non-symmetric. We will develop an algebraic multigrid method to solve this system, and the details on this method is given in the next section.

4. Numerical issues

4.1. Mesh adaptive methods

The mesh adaptive method used in this work follows the one in our previous work [1]. We use the hierarchical geometry tree (HGT) to manage the mesh data. There are a few advantages of using such tree type data structure to handle the mesh. First of all, locally refining or coarsening the mesh grids corresponds to growing new leaf nodes or cutting the leaf nodes from the HGT, which can be done quite efficiently. Secondly, the HGT data structure gives the leaf nodes, i.e., the geometry elements, from the old tree and the new tree a belonging-to relation, which can generate a highly efficient interpolation operation between two meshes. This is crucial for a time-dependent simulation. For details of HGT data structure, please refer to [1] and references therein.

One of the most important issues for the mesh adaptive methods is the generation of an effective indicator for each element in the mesh. The mesh will be locally refined or coarsened in terms of these indicators. From the efficiency point of view, the indicator should satisfy two requirements. First, the indicator should give reliable information on where to refine or coarsen. Second, the calculation of the indicator should be easy and not demanding on computational resource. In [1], we design the indicator by using the gradient of the wavefunction. This is a reasonable choice since the wavefunctions vary dramatically in the vicinity of the nucleus, especially for the core electrons when an all-electron calculation is implemented. The usage of the gradient of the wavefunction makes the density of the mesh grids around the nucleus greater than those in the region away from the nucleus. Hence the wavefunctions can be resolved well, and excellent results can be observed from [1]. However, the success of such indicator is based on the fact that the singularity introduced by the external potential \( v_{\text{ext}} \) also appears in the vicinity of the nucleus. The \( v_{\text{ext}} \) in an all-electron Kohn–Sham equation is given by

\[
v_{\text{ext}} = -\frac{Z_j}{|\tilde{x} - \tilde{R}_j|}, \quad j = 1, 2, \ldots, M,
\]

where \( Z_j \) and \( \tilde{R}_j \) stand for the charge and position of the \( i \)-th nucleus, respectively. \( M \) is the number of the nuclei. Obviously, this term is quite singular around \( \tilde{R}_j \). Consequently, in the Kohn–Sham simulations, the gradient of wavefunction based indicator works very well.

However, it is conceivable that the indicator in [1] might not work well in a time-dependent simulation. When the external perturbation added on the system is sufficiently weak, the wavefunction will only vary around its ground-state. In this case, the gradient-based indicator can still resolve the singularity introduced by \( v_{\text{ext}} \) well. But it is not the case when a sufficiently strong external field such as a short pulse laser is added on the system. Driven by the laser field, the electrons will move back and forth around the nucleus in a very large range, or even the atom will be ionized sometimes. Hence, using the gradient-based indicator in the mesh adaptive methods will fail on resolving the singularity introduced by \( v_{\text{ext}} \). On the other hand, the problem we solved in this paper is a time-dependent one, the temporal discretization of the time-dependent equation will also introduce the numerical error. Hence, it is reasonable that the indicator contains the information from the temporal discretization.

As shown in Section 3, we solve the complex system by separating the real and the imaginary parts of the wavefunction, i.e., we solve the following system instead of the original equation

\[
\begin{align*}
\frac{\partial}{\partial t} \phi^{\text{real}} &= -\frac{1}{2} \nabla^2 + v_{\text{KS}} \phi^{\text{imag}} \\
-\frac{\partial}{\partial t} \phi^{\text{imag}} &= -\frac{1}{2} \nabla^2 + v_{\text{KS}} \phi^{\text{real}}.
\end{align*}
\]

Hence, we use the following heuristic a posteriori error estimator as the error indicator in the simulation,
where \( \phi_j^{\text{real},n} \) and \( \phi_j^{\text{img},n} \) stand for the real and imaginary parts of the \( j \)-th wavefunction at the time level \( n \), respectively. In the above formula, \( J_e (\tilde{n}_e \cdot \nabla \phi_j) = (\nabla \phi_j|_{T_k} \cdot \tilde{n}_{e,kl} + \nabla \phi_j|_{T_i} \cdot \tilde{n}_{e,ik}) \) denotes the jump of the gradient of the wavefunction across the face \( e \) which is the common face of two tetrahedron elements \( T_k \) and \( T_i \). \( \tilde{n}_{e,kl} \) and \( \tilde{n}_{e,ik} \) stand for the unit outward normals on the face \( e \) of \( T_k \) and \( T_i \), respectively. The parameter \( h_e \) stands for the diameters of the face \( e \). The term \( R_{T_k}^{\text{real}} \) denotes the element residual of the first equation in (19), and has the following form

\[
R_{T_k}^{\text{real}} = \frac{1}{2} \frac{1}{2} \Delta \phi_j^{\text{img},n} - v_{KS} \phi_j^{\text{img},n} + \frac{1}{2} \Delta \phi_j^{\text{img},n-1} - v_{KS} \phi_j^{\text{img},n-1} + \frac{\phi_j^{\text{real},n} - \phi_j^{\text{real},n-1}}{\Delta t} ,
\]

and \( h_T \) stands for the diameter of the element \( T_k \). The term \( R_{T_k}^{\text{img}} \) is given similarly.

The a posteriori error estimation (20) is obtained following [32]. In fact, \( J_e \) and \( R_{T_k} \) describe the edge and element residuals, respectively, while the last two terms in (20) describe the temporal residual. In the numerical examples in next section, we can see that such error indicator works very well in the simulations. The theoretical analysis of the validity of the error estimation (20) will be reported in our forthcoming paper.

Remark. To make the transition of the adaptive simulation from the ground-state case to the time-dependent case smooth, we actually use the following error indicator in the ground-state calculation

\[
\eta_{T_k} = \sum_{i}^{N} \frac{1}{2} h_e J_e (\tilde{n}_e \cdot \nabla \phi_j) \frac{1}{2} \Delta \phi_j^{\text{img},n} + \frac{1}{2} \Delta \phi_j^{\text{img},n-1} - v_{KS} \phi_j^{\text{img},n} - v_{KS} \phi_j^{\text{img},n-1} + \frac{\phi_j^{\text{real},n} - \phi_j^{\text{real},n-1}}{\Delta t} ,
\]

where \( J_e (\tilde{n}_e \cdot \nabla \phi) \) again denotes the jump of the gradient of the wavefunction. The term \( R_{T_k}^{\text{GS}} \) denotes the element residual of the Kohn–Sham equation with the form

\[
R_{T_k} = \epsilon_i \psi_i^h + \frac{1}{2} \nabla^2 \psi_i^h - v_{KS}(\tilde{x}) \psi_i^h ,
\]

where \( \epsilon_i \) stands for the \( i \)-th eigenvalue of the Kohn–Sham equation, and \( v_{KS}(\tilde{x}) \) is the Kohn–Sham effective potential.

4.2. An algebraic multigrid solver for the complex symmetric system

If the partial differential equation (14) is directly discretized by using a complex space, the resulting algebraic system is a complex symmetric one. In our study, we separate the real part and the imaginary part of the wavefunctions, and use the same real finite element space to discretize both variables. Consequently, an equivalent real system from (18) can be written as the following form

\[
A B \phi^{\text{real}}(\tilde{x}, t + \Delta t) = B A \phi^{\text{real}}(\tilde{x}, t) = f^{\text{real}}
\]

\[
B A \phi^{\text{img}}(\tilde{x}, t + \Delta t) = -B A \phi^{\text{img}}(\tilde{x}, t) = f^{\text{img}}
\]

where \( A \) is a mass matrix, and \( B \) is a matrix obtained from the discretization of \((-\frac{1}{2} \Delta + v_{KS}[n])\phi \).

As we mentioned before, to guarantee the numerical accuracy and the stability, the predictor–corrector method given by Algorithm 1 is implemented in each time step. Even let \( MITER = 1 \), it means that this kind of system needs to be solved twice in each step. Then an efficient solver for this kind of systems is crucial for the simulations, because tens of thousands or even more time steps may be needed in one single simulation.

It is obvious that the above system is not symmetric. Hence the Bi-CGSTAB method [33] could be a natural choice to solve (23). However, when the condition number of the matrix is large, the convergence rate and the accuracy of Bi-CGSTAB method could be negatively affected. Unfortunately, this is the case for our \( h \)-adaptive finite element solver. With the local refinement and coarsening implementation, the mesh size varies dramatically from the vicinity of the nucleus to the boundary region of the computational domain, which enlarges the condition number of the matrix. To reduce the condition number and accelerate the convergence, a preconditioning process becomes necessary. In [18], an effective preconditioner, \((\alpha A + T)^{-1}\), is presented. Here \( \alpha \) is a constant, \( A \) is the mass matrix, and \( T \) is the stiff matrix. This real preconditioner
Fig. 1. Left: A point \( p_i \) in \( \mathcal{C} \) and its neighbors \( p_{j} \) to \( p_{f} \). Right: A point \( p_{f} \) in \( \mathcal{F} \), and its neighbors \( p_{i} \) to \( p_{c} \).

works well there. Because the matrix \((\alpha A + T)^{-1}\) is symmetric positive-definite (SPD), the Bi-CGSTAB with a multigrid preconditioner could be a competitive solver for (23).

We develop an algebraic multigrid (AMG) method in this paper to solve (23). Since it is proposed, the AMG method has proved to be an efficient method, theoretically and practically, see [4] and references therein. The idea of the multigrid method is based on the following observation. For a given linear system which is derived from a partial differential equation on a given mesh, when an iterative method such as Jacobi iteration is used to solve it, the numerical error with high frequency can be removed much more efficiently than that with low frequency. Hence, the strategy of the multigrid method is to project the residual equation on the fine mesh to the one on the coarse mesh. After solving the equation on the coarse mesh, the solution will be interpolated back to the one on the fine mesh, then the solution of the original system will be corrected accordingly. This is a typical process of the multigrid method. In terms of the above process, two questions need to be well answered for designing a quality multigrid solver. The first question is how to transfer functions between different grids (projection and interpolation), and the second one is what iterative method (smoother) we should use on each grid. To better explain our method for the system (23), let’s consider the Poisson equation below.

\[
-\nabla^2 v_H = 4\pi n(x, t), \quad \bar{x} \in \Omega \\
v_H|_{\partial\Omega} = v.
\]

The above Poisson is solved in our time-dependent simulation to generate the Hartree potential in each time step. The boundary value \( v \) here is given approximately by multipole expansion method, see [1]. Suppose that the final system obtained by linear finite element discretization of (24) is \( L v_{H,b} = b \). Then each row of the matrix \( L \) corresponds to a grid point in the mesh. To generate the coarse system from the fine system, we need to split the grid points into two sets, i.e., the set \( \mathcal{F} \) containing the points which do not belong to the coarse mesh, and the set \( \mathcal{C} \) containing the points which belong to the coarse mesh. The sets \( \mathcal{C} \) and \( \mathcal{F} \) satisfy the following relations,

\[
\mathcal{C} + \mathcal{F} = \mathcal{S}, \quad \mathcal{C} \cap \mathcal{F} = \emptyset,
\]

where \( \mathcal{S} \) and \( \emptyset \) stand for the set of all grid points on the fine mesh and empty set, respectively. Our strategy to choose \( \mathcal{C} \) and \( \mathcal{F} \) is similar to [7]. The algorithm can be described as follows. We move a point \( p_c \) in \( \mathcal{S} \) to \( \mathcal{C} \), and move all its neighbor points which are still in \( \mathcal{S} \) to \( \mathcal{F} \). Fig. 1 (left one) shows the point \( p_c \) and its neighbor points \( p_{j} \), \( i = 1, 2, \ldots, 5 \). We repeat the above process till \( \mathcal{S} \) is empty. For the interpolation, we design it by average. For instance, if a point \( p_j \) in \( \mathcal{F} \) directly connect to 5 points in \( \mathcal{C} \) (see Fig. 1, right one), then the updated solution \( v_{H,f} \) on the fine mesh is given by the summation of each point on the coarse mesh with an equal weight \( 1/5 \). With this strategy, an interpolation matrix \( P \) can be generated. There are different ways to design the projection. We simply use the transpose of \( P \), i.e., \( P^T \) to project the solution on the coarse mesh to the fine mesh. In summary, if we use \( r = b - Lv_{H,b} \) to denote the residual on the fine mesh, then the residual equation is given by \( L e_b = r \), and the related linear system for the residual equation on the coarse mesh is defined by \( L e_b = \hat{r} \), where \( L = P^TLP, \hat{e}_b = Pe_b \), and \( \hat{r} = Pr \). After we get \( \hat{e}_b \), the correction of the solution \( v_{H,b} \) on the fine mesh can be given as \( P^T \hat{e}_b \). The above method is a kind of two-grid method. When we do the projection of the system recursively, then interpolate the correction level by level, we get a multigrid method. In the implementation, we use Gauss–Seidel iterative method as the smoother to damp out the high frequency error before the projection and after the interpolation in each level, and only 2 or 3 iterations is implemented. In our simulations, such AMG solver works very well in solving (24).

In solving the system (23), it is obviously that the one-to-one correspondence between a row in (23) and a grid point in the mesh is no longer available. Hence a direct use of the above process to construct the projection and interpolation can no longer work either. However, since the same linear finite element space is used to discretize both the real and imaginary equations, the matrices \( A \) and \( B \) in (23) have the same shape. Consequently, an idea to design the projection and interpolation for (23) is as follows. First, the interpolation \( P \) is generated from the matrix \( A \) exactly according to the process mentioned above. Then the interpolation for (23) is given as

\[
P = \begin{bmatrix} P & 0 \\ 0 & P \end{bmatrix}
\]
The transpose of the above matrix $P^T$ is used as the projection. With $P$ and $P^T$, the functions can be transferred smoothly between different grids.

For the smoother, the Gauss-Seidel iterative scheme is implemented 2 to 3 times to damp out the high-frequency part of the error before the projection and after the interpolation. In the numerical simulation, the V-cycle iteration is employed for the multigrid method.

In the numerical simulations in Section 5, the above multigrid method works quite well. The residual of the system (23) can be reduced to $10^{-8}$ by only a couple of V-cycle iterations.

**Remark.** In the case when the pseudopotential is used in the simulation, a nonlocal part in the pseudopotential could be involved. Consequently, the shape of the matrix $B$ will be different from the shape of the matrix $A$. In this case, we may still use the projection $P^T$ and the interpolation $P$ generated from $A$ on $B$, and some preliminary results show that the effect of such operation is quite well. The other choice is to regenerate the pseudopotential according to the position of the grid points. It could be better since the accurate pseudopotential is involved in the coarse system. More tests are needed for this case.

### 4.3. The absorbing boundary conditions

The treatment of the boundary condition for the TDDFT simulations is crucial for the numerical accuracy and the reliability of the numerical solutions. Unlike the DFT simulations, simply using the Dirichlet boundary condition in a TDDFT simulation may cause the wave reflection when the wave arrives at the domain boundary. Consequently, an appropriate absorbing boundary condition, which can remove or significantly reduce the wave reflection, becomes important for the reliability of the simulations.

There are mainly two common approaches to implement the absorbing boundary conditions. The first one is the negative imaginary potential (NIP) method. In the NIP method, a negative imaginary potential $v_{nip}$ is introduced in the Hamiltonian operator. This potential only works near the boundary of the computational domain, and it is 0 in the interior of the domain. For example, define a sphere which surrounds the electronic structure, and its radius is $R$. The following potential can be used in the NIP method.

$$v_{nip} = \begin{cases} 0, & r \leq R, \\ -i\alpha|\mathbf{r} - \mathbf{R}|, & r > R. \end{cases}$$

(25)

Recently, a new negative imaginary kinetic energy (NIK) method is proposed in [18]. This method is one kind of PML method, with some modifications from the standard PML implementation. After adding an additional negative imaginary function $\sigma(\mathbf{x})$ to the kinetic energy term in the Hamiltonian, the weak form of the kinetic energy becomes

$$\frac{1}{2} + i\sigma(\mathbf{x}) \nabla \phi \cdot \nabla \psi d\Omega,$$

(26)

where $\sigma(\mathbf{x})$ is a parametrization function decreasing from zero towards negative infinity. Unlike the standard PML implementation, the above NIK method will not introduce the advection term, which may potentially enhance the stability of the algorithm.

In this paper, we use the following mask function method to handle the boundary condition. After each time step the new wavefunction $\phi(\mathbf{x}, t + \Delta t)$ is obtained from $\phi(\mathbf{x}, t)$. Then the amplitude of the wavefunction is reduced gradually towards the boundary by applying a mask function $M(\mathbf{x})$. Hence, the wavefunction for next time propagation step has the following form,

$$\phi(\mathbf{x}, t + \Delta t) = M(\mathbf{x})\phi(\mathbf{x}, t + \Delta t).$$

In the implementation, the mask function $M(\mathbf{x})$ is defined as follows,

$$M = \begin{cases} 1, & |\mathbf{x}| < R_{in}, \\ \cos\left(\frac{\pi}{2} \frac{|\mathbf{x}| - R_{in}}{R_{out} - R_{in}}\right), & R_{in} \leq |\mathbf{x}| \leq R_{out}, \\ 0, & R_{out} < |\mathbf{x}|. \end{cases}$$

(27)

The selection of parameters $R_{in}$ and $R_{out}$ depends on the problems.

Unlike the NIP and NIK methods which modify the Hamiltonian operator, the mask function approach operates the wavefunction directly. Hence, it is simpler on the implementation. The mask function has been widely used in TDDFT simulations, we refer to [20] and references therein for more details.

**Remark.** With the mesh adaptive methods, we could use quite large computational domain in the simulation, while the total number of the mesh grids can be controlled since only trouble regions need dense mesh grids. So the absorbing boundary condition can be dropped if a sufficiently large domain is used, and the reliability of the numerical results can potentially be benefit from removing this artificial process. This can be observed from our numerical simulations in Section 5.
5. Numerical experiments

In this section, the effectiveness of proposed numerical method will be tested by a variety of simulations for both linear and nonlinear effects. The main parameters for the hardware are Intel(R) Core(TM) i5-3470 CPU @ 3.20 GHz (4 cores, 6 M cache), and 8 Gb memory. The software we used in the simulation is AFEABIC [1,2], which is implemented in C++. AFEABIC supports both all-electron and pseudopotential calculations.

5.1. Calculation of photoabsorption spectra

One of the most important applications of TDDFT is to calculate the photoabsorption spectra [11]. In this subsection, the photoabsorption spectra of some atoms and molecules are calculated by our numerical method to verify its reliability. It is noted that the simulations in this subsection belong to the perturbative regime, which means that the variation of the electron density is weak, and just around the ground-state one. Hence, the mesh adaptive method is only used in the generation of the ground-state of the system, and is turned off during the time-dependent simulations. The numerical results show that such strategy works very well.

The process of the simulations for calculating photoabsorption spectra of a given system is as follows. First, the ground state of the electronic system is calculated. To do so, the numerical method proposed by the authors in [1] is employed. Then the ground state wavefunctions are perturbed slightly. Generally, each of the single-particle Kohn–Sham wavefunction is perturbed by multiplying a phase $e^{i\vec{k} \cdot \vec{r}}$, and the initial state for TDDFT calculation is written as

$$\psi_f (t = 0^+) = e^{i\vec{k} \cdot \vec{r}} \psi_f (t = 0^-),$$

where $\psi_f (t = 0^-)$ stands for the ground-state wavefunction. This phase-factor shifts the momentum of the electrons, giving then a coherent velocity field that causes the appearance of a polarization as the system evolves in time [20]. In the simulation, a sufficiently small $\vec{k}$ is needed to keep the response of the system linear and dipolar. After initializing the system, those Kohn–Sham wavefunctions are further propagated during a finite time. Then the dynamics of the system can be analyzed in terms of the time-dependent induced dipole moment of the electron cloud

$$D(t) = \sum_{i=1}^{N} \psi_i^*(t) \vec{x} \psi_i(t) d\Omega,$$

where $\psi_i(t)$ stands for the $i$-th time-dependent Kohn–Sham wavefunction. Finally, the linear photoabsorption spectra are obtained from the Fourier transform of $D(t)$.

To test the numerical convergence of our method, the photoabsorption spectrum of a single hydrogen atom is calculated. It is noted that there is only one electron in a Hydrogen atom. Hence, we actually solve the Schrödinger and time-dependent Schrödinger equations instead of Kohn–Sham and time-dependent Kohn–Sham equation in this example, which means that only kinetic energy operator and external potential are considered in the Hamiltonian. In this example, the all-electron calculation is employed, and the size of the computational domain is $[-9.9]^3$ (atomic unit is used here). To clearly show the numerical convergence of our method, three successively refined meshes are used in the simulations. The initial uniform mesh contains 2457 grid points, which means that the initial mesh size is around 1.33 au. The following two successively refined meshes contain 17,969 and 137,313 grid points, which means that the sizes of two meshes are around 0.69 au and 0.35 au, respectively. Since we focus on the numerical convergence of our method, the mask function is turned off in the simulations. For the size of the time step, we use (13) to determine it as around 0.016 au for the most dense mesh case, then the same value is used in the other two simulations. The end time of the simulation is around 241 au. In all three simulations, the kick strength in (28) is 0.01 au. Since this is not a large kick strength for a hydrogen atom in an all-electron simulation, the most strong peak in the spectrum should appear around 10.2 eV. The numerical results are shown in Fig. 2 (left one), where the numerical convergence of our method is shown clearly. With the successively refinement of the mesh, the position of the strongest peak in the spectrum moves to the 10.2 eV.

It is noted that there are weak peaks after the strong one in each spectrum in Fig. 2 (left one). This can be explained by the nonphysical reflection of the wavefunction from the boundary. These nonphysical peaks can be effectively restrained by using mask function method. To show the effect of the mask function, the following simulations are implemented. The size of the computational domain is still $[-9.9]^3$ au, and the same mesh partition, i.e., 17,969 mesh grids (mesh size 0.69 au), is used in all three simulations. To make the comparison more clearly, a longer simulation time is used here, i.e., the end time of the simulation is around 447 au now. There is no mask function used in the first simulation, and the result is given by the narrow dashed line in Fig. 2 (right one). In the second simulation, the mask function (27) is used with $R_m = 8$ au and $R_{out} = 9$ au, and the result is given by the wide dashed line in Fig. 2 (right one). In the third simulation, the mask function (27) is used with $R_m = 6$ au and $R_{out} = 9$ au, and the result is given by the solid line in Fig. 2 (right one). Two observations can be made from the figure. First, without mask function, several nonphysical peaks in the spectrum can be observed obviously. With the help of mask function, these nonphysical peaks are effectively restrained, and stronger mask function gives better effect on restraining the nonphysical peaks. However, the second observation is that the strong peak shifts to a higher energy position after we strengthen the artificial absorption, which is negative for a simulation. A similar phenomenon has been also reported in [18]. To deliver a reliable result, an appropriate setup is needed to balance the above
Fig. 2. Photoabsorption spectra of hydrogen atom obtained from our method. Left: The comparison of the spectra obtained from three successively refined meshes. Right: The comparison of the spectra when different mask function is used in the same simulation.

Fig. 3. The photoabsorption cross section with an x-direction perturbation (left), and the photoabsorption cross section with a y-direction perturbation (right), for a dilithium molecule.

two kinds of effect of the mask function. The other way is to use a sufficiently large domain so that the artificial absorbing strategy can be dropped. In our examples for Na2 and Na4 molecules below, we will show that our mesh adaptive method allow us to implement such simulations efficiently.

The photoabsorption spectra of a dilithium molecule is simulated in our second example. In this simulation, the all-electron calculation is employed. After reaching the ground state of the dilithium molecule, we use process mentioned above to generate the photo absorption spectra, and the results from our algorithm are shown in Fig. 3. To clearly demonstrate the positions of the peaks in the photoabsorption cross section data, we display the results from an x-direction perturbation and a y-direction perturbation separately. We use the results shown in [15] to validate our solution. It is obvious that our results match those presented in [15] well.

The last two examples in this subsection are for the photoabsorption spectra of a Na2 molecule and a Na4 molecule. Different from the previous two examples where the all-electron calculations are implemented, these two examples employ the pseudopotential calculations. In our simulations, the Troullier-Martins norm conserving pseudopotential is employed, and the Kleinman-Bylander method is used to accelerate the implementation. The computational domain we used in the simulation is $[-100, 100] \times [-100, 100] \times [-100, 100]$, and the unit is au. Even using so large domain, there are only 3239 grid points in the mesh. With our adaptive method, the smallest mesh size is around 0.7 au in the vicinity of the nucleus, while the largest one is around 25 au, which gives a quality distribution of the mesh grids, see Fig. 4. Our numerical results agree with the experimental ones [30] very well, see Fig. 5. It is worth mentioning that no absorbing boundary condition is used in two simulations.

5.2. Calculation of high harmonic generation

High harmonic generation (HHG) is one of the typical nonlinear effects in the study of the response of a system to an intense external field. This nonlinear effect can be explained by a semi-classical three-step model. First of all, the electron is
treated quantum mechanically, say, the electron tunnel ionizes from the parent nucleus under a polarized laser pulse. Then the dynamics of the electron are modeled classically. Half an optical cycle after ionization, the electron will reverse direction as the electric field changes, and will accelerate back towards the parent nucleus. Finally, when the electron reaches to the parent nucleus, it might be captured by the nucleus, and the atom arrives at its ground state. In the meantime, pronounced signals at multiples of the driving frequency appear in the photoemission spectrum. Please refer to [9] and references therein for the detail of this three-step model.

The simulation process of HHG is similar to the perturbative ones in the last subsection. The ground state of the system is calculated first. Then the system evolves under a polarized laser pulse. In this process, the time-dependent dipole moment of the system at each time step is recorded. The harmonic order of the system can then be read from the power spectrum generated by related Fourier transform. Although the procedure looks simple, a quality numerical simulation of HHG is quite challenging, especially for a large system. Compared with the perturbative simulations where the time-dependent wavefunctions only vary around their ground-states, the variation of the wavefunctions during the HHG simulation could be very dramatic. Hence, to resolve the variation well, the following two things are needed. First, a sufficiently large domain is needed. If the domain is not large enough, the interesting variation of the electron density might be absorbed by the boundary, which result in an unreliable simulation. Secondly, the mesh should be sufficiently dense where needed to well record the variation. Consequently, to deliver a computable simulation, the domain size needs to be estimated first, then an acceptable mesh size is used based on the numerical experience. For example, in [12], to simulate the response of a CO₂ molecule under a short laser pulse field, a three-dimensional cubic cell with the dimension $z_{\text{max}} = 40 \ \text{au}$, and a uniform grid spacing of $\Delta z = 0.28 \ \text{au}$ is employed. This results in over twenty million grid points in the domain, and it is mentioned in [12] that the multiprocessor parallel computers from Compute Canada are used for the simulations. In the following numerical example, we will show that our adaptive methods might save computational resources in a similar simulation.

The numerical example in this subsection is the simulation of the harmonic generation for a lithium atom. The Troullier–Martin pseudopotential is used to remove the singularity introduced by core electron, and LDA and ALDA are employed
the exchange-correlation potential and time-dependent exchange-correlation potential, respectively. The laser is polarized parallel to the z-axis, and the corresponding potential is given by

\[ v_{\text{laser}} = z E_0 \sin^2\left(\frac{\pi t}{T}\right) \sin(\omega t), \]

where \( E_0 = 0.01 \) au stands for the amplitude of the electric field, \( T \) determines the laser pulse duration, \( \omega = 0.03 \) au is the frequency of the laser field. In the implementation, \( T = 2\pi/(5\omega) \) is used, and the laser pulse is given in Fig. 6 (left one).

The results from the above simulation are as follows. The time-dependent dipole moments are given in Fig. 6 (right one), and corresponding dipole power spectrum is given in Fig. 7. Since the inverse symmetry of the lithium atom, theoretically only odd multiples of the driving frequency are emitted. This can be verified from the positions of the peaks in Fig. 7. In addition, the decay of the magnitude of the power spectrum stops after the 11-th harmonic order, which partially verifies the plateau phenomenon observed from the physical experiments for HHG.

Fig. 8 shows the solutions at three snapshots with our adaptive method. For \( t = 0 \) au, it (left column in Fig. 8) shows the ground state of the lithium atom. As it evolves over time, the variation of the electron density becomes larger, hence when \( t = 400 \) au (middle column in Fig. 8), our adaptive method starts adjusting the density of the mesh grids according to the indicator. With the magnitude of the laser pulse approximately reaches at its maximum value at \( t = 594 \) au (right column in Fig. 8), the variation of the isosurface of the electron density become larger, and our adaptive method successfully refines the mesh where needed to keep the variation of the electron density reliable. It is worth mentioning that the domain size we used in this simulation is \([−150, 150] \times [−150, 150] \times [−150, 150]\), and atomic unit is used. With so large computational domain, the maximum number of the mesh grids during the simulation is around one hundred and ten thousand, which shows that our adaptive method could utilize the computational resource very reasonably.
6. Conclusion

We develop a numerical framework of using $h$-adaptive finite element methods to solve TDKS equation. To optimize the mesh density during the time-dependent simulation, a heuristic a posteriori error estimation is introduced based on the temporal and spatial residual of the equation. To efficiently solve the derived block matrix, an algebraic multigrid method is designed, and it works very well. To restrain the unphysical oscillation, the mask function is used to absorb the wavefunctions which arrive at the boundary. It is worth mentioning that with our adaptive method, sufficiently large computational domain could be used to avoid using the absorbing boundary conditions. Our numerical examples for the photoabsorption spectra calculations for Na2 and Na4 molecules verify this very well. Finally, an HHG simulation for a lithium atom is given by using our adaptive methods, the numerical results confirm the theoretical ones very well, and the variation of the mesh grids during the simulation successfully show the effectiveness of our adaptive method.

A lot of excellent numerical results in [17,22,25], etc., have shown the advantage of using high-order finite element methods for Kohn–Sham equation. One of our future works is to develop a high-order adaptive finite element method for TDKS equation. We believe that a high-order method could enhance the numerical accuracy and efficiency. However, a lot of numerical issues need to be taken care before we can obtain a quality high-order solver. We will report related results in other work. Another future research is on the numerical methods for time-dependent current-density functional theory, which can be coupled with Maxwell’s equations to explain many useful physical phenomenon in the nano-optics.

Acknowledgements

We would like to thank the anonymous reviewers for their helpful comments. The work of G. Bao was supported in part by the NSF grants DMS-0968360, DMS-1211292, the ONR grant N00014-12-1-0319, a Key Project of the Major Research Plan of NSFC (No. 91130004), and a special research grant from Zhejiang University. The research of G.H. Hu was supported in part by SRG024-FST12-13R-HGH, MRG016/HGH/2013/FST, MYRG2014-00111-FST from University of Macau, 085/2012/A3 from FDCT of Macao S.A.R., and National Natural Science Foundation of China (Grant No. 11401608). The research of D. Liu was supported by NSF Focused Research Group grant DMS-096836.

References
