Numer. Math. Theor. Meth. Appl.
doi: 10.4208/nmtma.OA-2020-0040

Vol. **14**, No. 1, pp. 261-284
February 2021

An Implicit Solver for the Time-Dependent Kohn-Sham Equation

Lei Yang[1], Yedan Shen[2], Zhicheng Hu[3,*] and Guanghui Hu[2,4]

[1] *Faculty of Information Technology, Macau University of Science and Technology, Macao SAR, China*
[2] *Department of Mathematics, University of Macau, Macao SAR, China*
[3] *Department of Mathematics, College of Science, Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu Province, China*
[4] *Zhuhai UM Science & Technology Research Institute, Zhuhai, Guangdong Province, China*

Received 9 March 2020; Accepted (in revised version) 25 June 2020

Abstract. The implicit numerical methods have the advantages on preserving the physical properties of the quantum system when solving the time-dependent Kohn-Sham equation. However, the efficiency issue prevents the practical applications of those implicit methods. In this paper, an implicit solver based on a class of Runge-Kutta methods and the finite element method is proposed for the time-dependent Kohn-Sham equation. The efficiency issue is partially resolved by three approaches, i.e., an h-adaptive mesh method is proposed to effectively restrain the size of the discretized problem, a complex-valued algebraic multigrid solver is developed for efficiently solving the derived linear system from the implicit discretization, as well as the OpenMP based parallelization of the algorithm. The numerical convergence, the ability on preserving the physical properties, and the efficiency of the proposed numerical method are demonstrated by a number of numerical experiments.

AMS subject classifications: 35Q41, 81Q05, 65M60, 65M55, 65M50

Key words: Time-dependent Kohn-Sham equation, implicit midpoint scheme, finite element methods, h-adaptive mesh methods, complex-valued algebraic multigrid methods.

1. Introduction

Suppose that there is an electronic structure system consisting of M nuclei and N electrons. The evolution of this many-body system in the nonrelativistic sense is fundamentally controlled by the time-dependent Schrödinger equation (TDSE)

$$i\frac{\partial}{\partial t}\Psi = H\Psi \quad \text{in } \mathbb{R}^3, \tag{1.1}$$

*Corresponding author. *Email address:* `huzhicheng@nuaa.edu.cn` (Z. Hu) where i denotes the imaginary unit, H consists of the kinetic energy operator for each particle as well as the classical Coulomb interactions between each pair of particles, and $\Psi := \Psi(\vec{X}_1,\dots,\vec{X}_M,\vec{x}_1,\dots,\vec{x}_N,t)$ is the high dimensional wavefunction depending on the position of each particle and a time variable. It is this high dimensionality of the wavefunction Ψ which makes the analysis and computation on the TDSE very challenging. The time-dependent Kohn-Sham (TDKS) density functional theory is one of the most successful approximation models towards partially overcoming this challenge, which can be written as

$$\begin{aligned} i\frac{\partial}{\partial t}\psi_j &= \left(-\frac{1}{2}\nabla^2 - \sum_l \frac{z_l}{|\vec{x}-\vec{R}_l|} + \int \frac{\rho(\vec{x}',t)}{|\vec{x}-\vec{x}'|}d\vec{x}' + v_{ALDA}(\rho)\right)\psi_j \\ &=: \left(-\frac{1}{2}\nabla^2 + V_{KS}\right)\psi_j, \quad j=1,\dots,N, \end{aligned} \tag{1.2}$$

where $\rho(\vec{x},t) = \sum_j |\psi_j(\vec{x},t)|^2$ is the time-dependent electron density, z_l and $\vec{R}_l$ for $l=1,\dots,M$ denote the nuclear charge and position of the l-th nucleus, and V_{KS} denotes the Kohn-Sham potential consisting of the external potential, the Hartree potential, as well as the exchange-correlation potential, respectively. Here, an adiabatic approximation for the exchange-correlation potential, denoted by v_{ALDA}, is considered. Guaranteed by the Runge-Gross theorem [22], the time-dependent electron density $\rho(\vec{x},t)$ is used as a fundamental variable to represent an evolved many-body system. It is noted that the electron density ρ is a four dimensional variable in a three dimensional space. This huge reduction of the dimension brings the possibility on quality analysis and simulation for the many-body system. So far, the TDKS equation has been widely used in a variety of applications such as material science, nano-optics, and attosecond science, etc. Please refer to [21] and references therein for more details on the application of the TDKS equation.

There have been lots of numerical methods in the market to solve the TDKS equation in the time domain, people may refer to [3,7,14] and references therein for detail. People may also refer to [11,16,28] for numerical methods of Schrödinger equation. Among those grid-based numerical methods, the finite difference methods [1], the finite element methods [3,8,9,17,18,27], the discontinuous Galerkin methods [20], the wavelet methods [12] etc. are popular for the spatial discretization, while there are Runge-Kutta methods, commutator-free Magnus expansion methods, etc. for the temporal discretization. It is worth mentioning that the comparison of the performance of those time propagators, including the linear multistep methods, can be found from a recent paper [14]. However, it should be pointed out that the memory issue of the solver is missed there, and that many factors would affect the performance of those solvers, for example, the performance of the linear solver for the implicit methods. Due to their advantage on the memory requirement, the single step methods such as the Runge-Kutta methods have attracted much attention in solving the time-dependent problems. Furthermore, some implicit one-step solvers have the property on well preserving the physical structure of the TDKS equation. These advantages make the solvers such as the implicit midpoint scheme very popular in the practical simulations. It is noted that some solvers using the explicit pseudospectral-splitting approach are also competitive [4, 24], since the Laplace operator in these solvers can be treated in Fourier space very efficient. However, when the practical problem with nonperiodic boundary conditions or the all-electron model is needed in the simulation, the finite element method with nonuniform mesh grids would be more attractive.

In this paper, we propose a numerical solver for the TDKS equation, with the implicit midpoint scheme for the temporal discretization and the finite element method for the spatial discretization. Towards improving the efficiency of the simulations, the following three approaches are studied in detail. First of all, an h-adaptive mesh method is employed in the algorithm to dynamically control the total amount of the grid points of the mesh. This is an attractive strategy for numerically solving the TDKS equation, especially when the all-electron model is considered in the simulation. An adaptive process is designed in this paper following [26], in which the distribution of the numerical error is generated by a heuristic *a posteriori* error estimation, while the efficient operations on the local refinement of the mesh grids as well as the solution update between two finite element spaces are guaranteed with the help of Hierarchy Geometry Tree (HGT) from [19]. Secondly, a complex-valued algebraic multigrid (AMG) solver is designed for efficiently solving the derived linear system. It is mainly the efficiency of the solver for the linear system which determines the efficiency of the implicit solver, since the linear system needs to be solved for over hundreds of thousands of times in a classic simulation for high harmonic generation. With the finite element discretization, it is known that the condition number of the matrix in the linear system is inversely proportional to h^2, where h means the size of the mesh grids. This implies that in the case of numerical discretization of an all-electron model, a quality preconditioner is required to effectively reduce the condition number, so that the linear system in the implicit method can be solved efficiently. In this paper, we follow [10] to use the so-called K formulation to express the complex-valued system by a blocked real-valued form. Then a complex-valued AMG is developed for solving the linear system, in which the restriction and prolongation operators are designed following [6], and a block Gauss-Seidel iterative method is used to damp out the high frequency part of the numerical error in each level of the AMG. The numerical experiments show successfully that the convergence behavior of the solver is not sensitive to the condition number of the matrix in the linear system. Finally, the algorithm will be parallelized by OpenMP technique, to further accelerate the simulation by fully utilizing the hardware resource, i.e., a Dell Precision 7920 Tower workstation, with dual Intel(R) Xeon(R) Gold 6136 CPU @ 3.00GHz (total 24 cores), and 250 Gigabytes memory. The greedy coloring algorithm is used in our work to help resolving the race condition issues in, for example, the formation of the total stiffness matrix.

In the following parts of this paper, a full description of the numerical discretization of the TDKS equation, including the implicit midpoint rule for the temporal discretization as well as the finite element method for the spatial discretization, is given in next section. In Section 3, three issues on the efficient implementation of the implicit method, i.e., an h-adaptive mesh method, a complex-valued AMG solver, as well as the OpenMP parallelization of the algorithm, are discussed in detail. In Section 4, the numerical convergence of the proposed method, as well as the performance of the method in solving the TDKS equation, are checked by a variety of numerical experiments. The conclusion and the future work are given in Section 5 finally.

2. Numerical discretization of the TDKS equation

The temporal discretization of the TDKS equation is introduced firstly, including the motivation on applying the Runge-Kutta methods with Gauss-Legendre collocation points, and a description of the midpoint scheme of the TDKS equation.

2.1. Structures of the TDKS equation and the implicit midpoint scheme

It is known that the following three properties are hold for the TDSE:

- It is a Hamiltonian system;
- Conservation of the probability, i.e., $\frac{\partial N(t)}{\partial t} = 0$, where $N(t) = \int \rho(\vec{x},t)d\vec{x}$ represents the total number of the electrons;
- The propagation of the system is time-reversal symmetry.

The Hamiltonian structure of the equation can be seen clearly by defining the Hamiltonian function

$$\mathcal{H}\big(\Psi^r,\Psi^i\big) = \frac{1}{2}\big\langle\Psi^r|H\Psi^r\big\rangle + \frac{1}{2}\big\langle\Psi^i|H\Psi^i\big\rangle, \tag{2.1}$$

where $\Psi = \Psi^r + i\Psi^i$ with Ψ^r and Ψ^i representing the real and imaginary parts of the wavefunction Ψ, respectively, and $< \cdot|\cdot >$ is the standard bra-ket notation in the quantum mechanics. Then the system (1.1) can be reformulated as

$$\begin{cases} \dfrac{\partial}{\partial t}\Psi^r = \dfrac{\partial \mathcal{H}}{\partial \Psi^i} \\ \dfrac{\partial}{\partial t}\Psi^i = -\dfrac{\partial \mathcal{H}}{\partial \Psi^r} \end{cases} \qquad \text{in } \mathbb{R}^3, \tag{2.2}$$

which formally is a Hamiltonian system.

The conservation of the probability is a natural requirement in the evolution of the quantum system, i.e., the total number of the electrons should be kept unchanged during the whole process. In other words, if the electron number is given by $N(t_0)$ at the initial time t_0, then $N(t) = N(t_0)$ should be always correct for any later time instant $t > t_0$. This is equivalent to impose the condition $\frac{\partial N(t)}{\partial t} = 0$ on $N(t)$. This property is preserved well by the Schrödinger equation by the observation that

$$\frac{\partial N(t)}{\partial t} = \int_{\mathbb{R}^3} \frac{\partial \rho(\vec{x},t)}{\partial t} d\vec{x} = i\left(\int_{\mathbb{R}^3} (H\Psi)^\star \Psi d\vec{x} - \int_{\mathbb{R}^3} \Psi^\star (H\Psi) d\vec{x}\right). \tag{2.3}$$

It can be seen that Hermitian property of the Hamiltonian H from the Schrödinger equation guarantees $\frac{\partial N(t)}{\partial t} = 0$, i.e., the conservation of the probability.

It can be easily checked that for a given Hamiltonian which is not explicitly dependent on time, if the wavefunction $\Psi(t)$ is the solution of the Schrödinger equation (1.1), then the function $\Psi^\star(-t)$ will satisfy the governing equation

$$i\frac{\partial}{\partial t}\Psi^\star(-t) = H\Psi^\star(-t). \tag{2.4}$$

The above property for the time-dependent Schrödinger equation is called T-symmetry.

Since the time-dependent density functional theory is a substitution for the many-body TDSE in the case that the time-dependent exchange-correlation potential is known exactly, the TDKS equation shares the same properties mentioned above with the TDSE. People may refer to a recent paper [14] for the related discussion. Consequently, in the temporal discretization of the TDKS equation, the above properties need to be preserved well by the numerical scheme, to make the numerical results physical.

Since their advantages on resolving the stiff problems and on the storage, the single step methods have been popular in the practical simulations, for examples, the implicit trapezoidal scheme (also known as Crank-Nicolson scheme) and the implicit midpoint scheme. It can be checked easily that both schemes

i) are of $\mathcal{O}(\Delta t^2)$, where Δt represents the step size in the temporal discretization,

ii) preserve the conservation of the probability,

iii) satisfy the time-reversal symmetry.

However, the implicit midpoint scheme is symplectic, while the implicit trapezoidal scheme is not. Consequently, the implicit midpoint scheme is more desirable in the simulations, in which the better performance for the long term simulations can be expected. In the following, we will use the implicit midpoint scheme to describe the temporal discretization of the TDKS equation.

Remark 2.1. The propagators from both implicit trapezoidal scheme and implicit midpoint scheme are *unitary*, from which the conservation of the probability can be shown.

Suppose that ψ_n represents the wavefunction in the TDKS equation (1.2) at the time instant t_n, and that the next time instant is given by $t_{n+1} := t_n + \Delta t$ with Δt the step size. Then the implicit midpoint scheme of the TDKS equation is given by

$$\psi_{n+1} = \frac{2i + \Delta t H(\rho_{n+\frac{1}{2}})}{2i - \Delta t H(\rho_{n+\frac{1}{2}})}\psi_n =: U_n^{n+1}\psi_n, \tag{2.5}$$

where $H(\rho)$ stands for the Hamiltonian operator depending on the electron density ρ, and $\rho_{n+\frac{1}{2}} = \Sigma_j|\psi_{n+\frac{1}{2},j}|^2$, in which $\psi_{n+\frac{1}{2},j}$ represents the evaluation of the j-th wavefunction at the time instant $t_{n+\frac{1}{2}} := t_n + \frac{\Delta t}{2}$.

One issue for the above discretization (2.5) is its nonlinearity since the unknown quantity $\rho_{n+\frac{1}{2}}$. A classic approach for resolving the nonlinearity is to introduce a prediction-correction process, as the follows:

Algorithm 2.1: Prediction-correction process on solving (2.5).

```
    Data: $\psi_n$, an initial $\psi_{n+1}(:= \psi_n)$, an auxiliary $\tilde{\psi}$, and $tolerance$
    Result: $\psi_{n+1}$
1   Calculate an initial $\rho_{n+\frac{1}{2}}$;
2   do
3   |   Let $\tilde{\psi} = \psi_{n+1}$;
4   |   Solve the Eq. (2.5) to get an updated $\psi_{n+1}$;
5   |   Update $\rho_{n+\frac{1}{2}}$;
6   while $||\psi_{n+1} - \tilde{\psi}|| > tolerance$;
```

In the *while* loop in the above algorithm, the quantity $\rho_{n+\frac{1}{2}}$ is updated by $\frac{1}{2}(\rho_{n+1}+\rho_n)$ with the updated ψ_{n+1} in our implementation. It is noted that the numerical error introduced in this approximation is consistent with the implicit midpoint rule. For the initial $\rho_{n+\frac{1}{2}}$ in Step 1, we may simply use ρ_n for the approximation. However, this rough approximation may cause more correction steps to obtain an accurate ρ_{n+1}.

There are two approaches to improve the initial approximation of $\rho_{n+\frac{1}{2}}$, i.e., solving the TDKS equation on the interval $[t_n, t_{n+\frac{1}{2}}]$ with some scheme, and the extrapolation methods. Basically, extrapolation methods would be faster since no linear system needs to be solved. However, the extrapolation methods would need more storage since more previous solutions are needed in the implementation. With better initial guess for $\rho_{n+\frac{1}{2}}$, a more accurate ρ_{n+1} and faster convergence of the prediction-correction process can be expected. People may refer to [15] for more details about the prediction-correction methods.

In our implementation, the implicit Euler scheme is used in the prediction step, while a maximum number for the correction steps is also introduced to control the efficiency of the simulation. In this case, the *while* loop would end either the condition listed in Step 6 in the algorithm becomes false, or the number of the correction steps exceeds the given maximum number.

2.2. The finite element discretization of TDKS equation

In the TDKS equation (1.2), the Hamiltonian consists of kinetic energy operator $-\frac{\nabla^2}{2}$, the external potential $-\sum_l \frac{z_l}{|\vec{x}-\vec{R}_l|}$, the Hartree potential

$$V_H = \int \frac{\rho(\vec{x}', t)}{|\vec{x}' - \vec{x}|} d\vec{x}',$$

and the exchange-correlation potential v_{ALDA}.

For the convenience of the description, we rewrite (2.5) as follows,

$$\begin{cases} 2\psi^r_{n+1} - \Delta t H\big(\rho_{n+\frac{1}{2}}\big)\psi^i_{n+1} = 2\psi^r_n + \Delta t H\big(\rho_{n+\frac{1}{2}}\big)\psi^i_n, \\ \Delta t H\big(\rho_{n+\frac{1}{2}}\big)\psi^r_{n+1} + 2\psi^i_{n+1} = -\Delta t H\big(\rho_{n+\frac{1}{2}}\big)\psi^r_n + 2\psi^i_n. \end{cases} \tag{2.6}$$

To derive the finite element discretization for the above system, we firstly introduce the standard Sobolev space in a given domain $\Omega \subset \mathbb{R}^3$ by $H^1 := W^1_2(\Omega)$, and define $V := \{\phi \in H^1(\Omega) : \phi = 0 \text{ on } \partial\Omega\}$. Then the variational form of (2.6) is given by: To find $\psi_{n+1} \in V$ such that

$$\begin{cases} \quad 2\displaystyle\int_\Omega \psi^r_{n+1}\phi d\vec{x} - \Delta t \int_\Omega \left(\frac{1}{2}\nabla\psi^i_{n+1}\cdot\nabla\phi + V_{KS}\psi^i_{n+1}\phi\right) d\vec{x} \\ = 2\displaystyle\int_\Omega \psi^r_n\phi d\vec{x} + \Delta t \int_\Omega \left(\frac{1}{2}\nabla\psi^i_n\cdot\nabla\phi + V_{KS}\psi^i_n\phi\right) d\vec{x}, \quad \forall\phi\in V, \\ \quad \Delta t\displaystyle\int_\Omega \left(\frac{1}{2}\nabla\psi^r_{n+1}\cdot\nabla\phi + V_{KS}\psi^r_{n+1}\phi\right) d\vec{x} + 2\int_\Omega \psi^i_{n+1}\phi d\vec{x} \\ = -\Delta t\displaystyle\int_\Omega \left(\frac{1}{2}\nabla\psi^r_n\cdot\nabla\phi + V_{KS}\psi^r_n\phi\right) d\vec{x} + 2\int_\Omega \psi^i_n\phi d\vec{x}, \quad \forall\phi\in V. \end{cases} \tag{2.7}$$

We introduce the following notation for the description of the finite element discretization. First of all, $\Omega \subset \mathbb{R}^3$ is used to denote the computational domain, and $\partial\Omega$ is its boundary. For this domain Ω, we have a tetrahedron mesh $\mathcal{T}$ which completely covers the domain Ω. The mesh $\mathcal{T}$ consists of a set of nonoverlapped tetrahedron elements, i.e., $\mathcal{T} = \{T_k\}_{k=1}^{N_{tet}}$, where N_{tet} is the total number of the tetrahedron elements in the mesh $\mathcal{T}$. Then following the definition introduced by Ciarlet, on tetrahedron elements $T_k, k = 1,\ldots,N_{tet}$, we define the finite element $(T_k, \mathcal{P}_1, \mathcal{N})$, where $\mathcal{P}_1$ is the set of all first order polynomials in three variables, and $\mathcal{N}$ is the set of nodal variables. With the above notations, we can define the C^0 finite dimensional subspace $V_\mathcal{T}$ of V.

Then the discretized variational form of (2.7) is given by: To find $\psi_{n+1,\mathcal{T}} \in V_\mathcal{T}$ such that

$$\begin{cases} \quad 2\displaystyle\sum_k\int_{T_k} \psi^r_{n+1,\mathcal{T}}\phi d\vec{x} - \Delta t \sum_k\int_{T_k} \left(\frac{1}{2}\nabla\psi^i_{n+1,\mathcal{T}}\cdot\nabla\phi + V_{KS}\psi^i_{n+1,\mathcal{T}}\phi\right) d\vec{x} \\ = 2\displaystyle\sum_k\int_{T_k} \psi^r_{n,\mathcal{T}}\phi d\vec{x} + \Delta t \sum_k\int_{T_k} \left(\frac{1}{2}\nabla\psi^i_{n,\mathcal{T}}\cdot\nabla\phi + V_{KS}\psi^i_{n,\mathcal{T}}\phi\right) d\vec{x}, \quad \forall\phi\in V_\mathcal{T}, \\ \quad \Delta t\displaystyle\sum_k\int_{T_k} \left(\frac{1}{2}\nabla\psi^r_{n+1,\mathcal{T}}\cdot\nabla\phi + V_{KS}\psi^r_{n+1,\mathcal{T}}\phi\right) d\vec{x} + 2\sum_k\int_{T_k} \psi^i_{n+1,\mathcal{T}}\phi d\vec{x} \\ = -\Delta t\displaystyle\sum_k\int_{T_k} \left(\frac{1}{2}\nabla\psi^r_{n,\mathcal{T}}\cdot\nabla\phi + V_{KS}\psi^r_{n,\mathcal{T}}\phi\right) d\vec{x} + 2\sum_k\int_{T_k} \psi^i_{n,\mathcal{T}}\phi d\vec{x}, \quad \forall\phi\in V_\mathcal{T}. \end{cases} \tag{2.8}$$

Following the K formulation proposed in [10], the unknowns are organized as

$$\left(\psi^{r,1}_{n+1,\mathcal{T}}, \psi^{i,1}_{n+1,\mathcal{T}}, \psi^{r,2}_{n+1,\mathcal{T}}, \psi^{i,2}_{n+1,\mathcal{T}}, \ldots, \psi^{r,N_{gp}}_{n+1,\mathcal{T}}, \psi^{i,N_{gp}}_{n+1,\mathcal{T}}\right)^\top \tag{2.9}$$

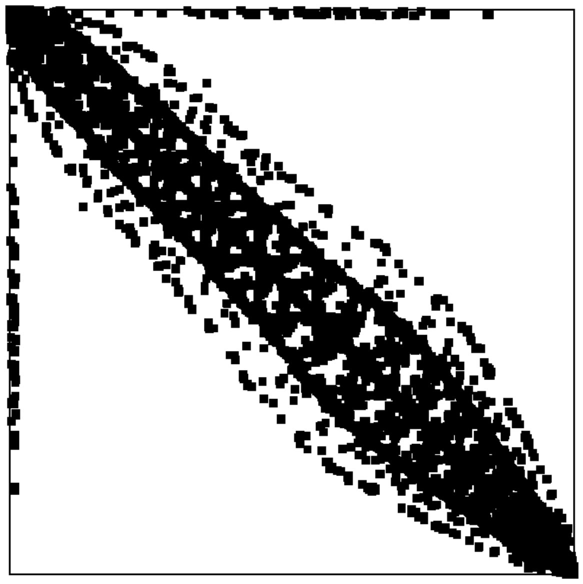


Figure 1: The sparsity pattern of the matrix.

with N_{gp} the total number of the grid points, and the pattern of the coefficient matrix has the form shown in Fig. 1.

In the K formulation form used in this paper, each black square ▪ in Fig. 1 represents a 2×2 submatrix of the form

$$\begin{bmatrix} a & -b \\ b & a \end{bmatrix} \tag{2.10}$$
$$= \begin{bmatrix} 2\int \phi_j \phi_k d\vec{x} & -\Delta t \int \left(\frac{1}{2} \nabla \phi_j \cdot \nabla \phi_k + V_{KS} \phi_j \phi_k \right) d\vec{x} \\ \Delta t \int \left(\frac{1}{2} \nabla \phi_j \cdot \nabla \phi_k + V_{KS} \phi_j \phi_k \right) d\vec{x} & 2\int \phi_j \phi_k d\vec{x} \end{bmatrix},$$

according to the discretized weak form (2.8). It is noted that the determinant of the above matrix is $a^2 + b^2$, hence that each submatrix (2.10) is always invertible.

3. Three approaches for accelerating the simulations

The efficiency is a main issue preventing the widely application of the implicit solvers in the practical simulations. In the following, focusing on the proposed method in last section, we introduce three approaches for improving the efficiency of the implementation.

3.1. The h-adaptive mesh method

A classic process of using the adaptive mesh methods consists of the following steps, i.e.,

$$\cdots \text{solve} \cdots \text{estimate} \cdots \text{mark} \cdots \text{refine} \cdots$$

In this work, the above process means that the TDKS equation (1.2) is solved firstly on the current finite element space, then the distribution of the numerical error is estimated, then the tetrahedron elements in the mesh are marked according to the error estimation. Finally, a new finite element space is built on the new mesh by locally refining or coarsening the elements in the old mesh, and the new solutions are obtained by the interpolation. In our implementation, we follow [19] to adopt the hierarchy geometry tree to manage the local refinement/coarsening operations of the mesh grids. It is noted that an efficient interpolation operation can be obtained based on this hierarchy geometry tree. People may also refer to [2,3] for the detail.

In the following, we first propose a heuristic *a posteriori* error indicator for each tetrahedron element in the mesh, then introduce how the h-adaptive module is embedded in the numerical method proposed in the last section.

It is noted that in [26], a residual type *a posteriori* error estimator for each element has been developed for the parabolic equations. Although they are neither parabolic nor hyperbolic for the TDSE/TDKS, we still borrow the idea in [26] to generate the error indicator, which consists of two components, i.e., temporal error and spatial error. Suppose that for a given wavefunction ψ, ψ_n and ψ_{n+1} represent the wavefunction at the time instants t_n and t_{n+1}, respectively. Then the temporal error is denoted by

$$\eta_t = \left(\Delta t \sum_{T_k \in \mathcal{T}} \int_{T_k} \left(\nabla \psi_{n+1} - \nabla \psi_n \right)^2 d\vec{x} \right)^{\frac{1}{2}}, \tag{3.1}$$

while the spatial error is denoted by

$$\begin{aligned} \eta_h = \Bigg(& \sum_{T_k \in \mathcal{T}} \Delta t \Delta h_{T_k}^2 \int_{T_k} \left(V_{KS} \psi_{n+\frac{1}{2}} - \frac{1}{\Delta t} \left(\psi_{n+1} - \psi_n \right) \right)^2 d\vec{x} \\ & + \sum_{e \in \mathcal{E}} \Delta t \Delta h_e \int_e \left(\frac{1}{2} \mathcal{J}_e \left(\frac{1}{2} \nabla \psi_n \cdot \vec{n}_e + \frac{1}{2} \nabla \psi_{n+1} \cdot \vec{n}_e \right) \right)^2 ds \Bigg)^{\frac{1}{2}}. \end{aligned} \tag{3.2}$$

In the above formula, the Kohn-Sham potential V_{KS} is evaluated by $\rho_{n+\frac{1}{2}}$ since the implicit midpoint scheme is used in our method. The $\mathcal{E}$ represents the set of all triangle faces in the mesh $\mathcal{T}$. The $\mathcal{J}_e(\nabla \psi \cdot \vec{n}_e)$ represents the jump of the gradient of the wavefunction ψ across the edge e along the normal direction of e with $\vec{n}_e$ the unit normal vector.

Based on (3.1) and (3.2), the error indicator in each element $T_k \in \mathcal{T}$ is given by

$$\begin{aligned}\eta_{T_k} = \Bigg(&\Delta t \int_{T_k} \big(\nabla\psi_{n+1} - \nabla\psi_n\big)^2 d\vec{x} \\ &+ \Delta t \Delta h_{T_k}^2 \int_{T_k} \left(V_{KS}\psi_{n+\frac{1}{2}} - \frac{1}{\Delta t}\big(\psi_{n+1} - \psi_n\big)\right)^2 d\vec{x} \\ &+ \frac{1}{2}\Delta t \Delta h_e \int_{e\in\partial T_k} \left(\frac{1}{2}\mathcal{J}_e\left(\frac{1}{2}\nabla\psi_n \cdot \vec{n}_e + \frac{1}{2}\nabla\psi_{n+1}\cdot\vec{n}_e\right)\right)^2 ds\Bigg)^{\frac{1}{2}}. \end{aligned} \tag{3.3}$$

It is noted that the factor $\frac{1}{2}$ in front of the third term of the above formula means equally distributing the error from the edge $e \in \partial T_k$ to its two neighbor tetrahedron elements.

Remark 3.1. In (3.2) and (3.3), the triangles locating on the domain boundary $\partial\Omega$ are not considered. The error estimation on those triangles is not trivial since the Dirichlet boundary condition is employed in the simulation. However, since the mask function technique is used to handle the issue of the reflection of the wavefunction around the domain boundary, it would be a reasonable assumption that there is no jump of the gradient of the wavefunction across the domain boundary. Hence, it is also reasonable to ignore the corresponding contribution on the error indicator.

The algorithm given below shows how the adaptive refinement of the mesh grids is embedded in Algorithm 2.1.

Algorithm 3.1: Prediction-correction process with adaptive mesh method on solving (2.5).

```
  Data: $\psi_n$, an initial $\psi_{n+1}(:= \psi_n)$, an auxiliary $\tilde{\psi}$, and $tolerance$
  Result: $\psi_{n+1}$
1 Calculate an initial $\rho_{n+\frac{1}{2}}$;
2 do
3     Let $\tilde{\psi} = \psi_{n+1}$;
4     Solve the Eq. (2.5) to get an updated $\psi_{n+1}$;
5     if First correction step then
6         Implement the adaptive mesh method;
7     end
8     Update $\rho_{n+\frac{1}{2}}$;
9 while $||\psi_{n+1} - \tilde{\psi}|| > tolerance$;
```

In the above algorithm, the adaptive mesh method is implemented in the first correction step, since with a small Δt and a quality $\rho_{n+\frac{1}{2}}$, the wavefunction in this stage is actually accurate enough. So the adaptive mesh method is only implemented once here in the whole prediction-correction process.

3.2. The algebraic multigrid solvers

Two AMG solvers are needed in our framework, one is a real-valued AMG for the generation of the Hartree potential in the Hamiltonian, and the other one is a complex-valued AMG for the linear system derived from the implicit scheme for the TDKS equation.

In the generation of the Hartree potential in the Hamiltonian, the following Poisson equation needs to be solved with the given electron density, i.e.,

$$\begin{cases} -\nabla^2 V_H = 4\pi\rho & \text{in } \Omega, \\ V_H = V_H^b & \text{on } \partial\Omega, \end{cases} \tag{3.4}$$

where V_H^b represents the Hartree potential on the domain boundary, and the multipole expansion is used here for the approximation. People may refer to [3, 25] and references therein for more detail.

Suppose that the linear system derived from the above Poisson equation is given by

$$A_h u_h = f_h, \tag{3.5}$$

where the subscript h denotes the size of the mesh for obtaining the stiffness matrix A_h. In an abstract description, the iteration method tries to generate a sequence $(u_h^{(n)})_{n=0}^{\infty}$ such that $u_h = \lim_{n\to\infty}(u_h^{(n)})$.

By introducing the error $e_h^{(n)} = u_h - u_h^{(n)}$, the residual $r_h^{(n)} = f_h - A_h u_h^{(n)}$ from the n-th approximation $u_h^{(n)}$, and by approximating the defect equation $A_h e_h^{(n)} = r_h^{(n)}$ with

$$\hat{A}_h \hat{e}_h^{(n)} = r_h^{(n)}, \tag{3.6}$$

the next approximation $u_h^{(n+1)}$ is obtained by

$$u_h^{(n+1)} = \left(I_h - \hat{A}_h^{-1} A_h\right) u_h^{(n)} + \hat{A}_h^{-1} f_h. \tag{3.7}$$

Two properties are expected from $\hat{A}_h$, i.e., $\hat{A}_h$ should be simple so that the system $\hat{A}_h \hat{e}_h^{(n)} = r_h^{(n)}$ can be solved efficiently, and the spectral radius of the iteration matrix $I_h - \hat{A}_h^{-1} A_h$ is less than 1 so that the iteration is convergent.

In the multigrid framework, the Eq. (3.6) is designed by discretizing (3.4) on a coarse mesh through the restriction operator R_h^H and the prolongation operator P_H^h, i.e.,

$$A_H R_h^H e_h^{(n)} =: A_H e_H^{(n)} = r_H := R_h^H r_h^{(n)}. \tag{3.8}$$

Then the new approximation $u_h^{(n+1)}$ can be expressed by

$$u_h^{(n+1)} = \left(I_h - P_H^h A_H^{-1} R_h^H A_h\right) u_h^{(n)} + P_H^h A_H^{-1} R_h^H f_h. \tag{3.9}$$

Based on the above description, the two-grid iteration method is given in Algorithm 3.2 below.

Algorithm 3.2: A two-grid iteration.

```
Data: $A_h$, $f_h$, initial guess $u_h^{(0)}$, $tol$, $n_s$, $n_m$, and $N_m$.
Result: $u_h$.
1 while $||r_h^{(n)}||_2 > tol$ or $n_m < N_m$ do
2     Implement certain classic iteration on the Eq. (3.5) for $n_s$ times;
3     Build the Eq. (3.8) by using $R_h^H$;
4     Solve the Eq. (3.8);
5     Update the solution with (3.9) by using $P_H^h$;
6     Implement certain classic iteration on the Eq. (3.5) for $n_s$ times;
7     Let $n_m = n_m + 1$;
8 end
```

If we use S_h to denote certain classic iterative operator, the iteration matrix for Algorithm 3.2 becomes

$$M_{tg} = S_h \left(I_h - P_H^h A_H^{-1} R_h^H A_h \right) S_h. \tag{3.10}$$

It is noted that the convergence of the above two-grid algorithm for the linear system derived from the model problem (3.4) has been studied in detail, please refer to [5] and references therein. It is also noted that the two-grid method can be recursively used, which delivers a multigrid algorithm.

It is noted that in our implementation, the generation of the restriction and prolongation operators in this paper follows [6], the Gauss-Seidel iteration is used for the smoother, and the implementation of the real-valued AMG follows [19].

Besides solving the Poisson equation (3.4), the AMG method is also employed in solving the complex-valued linear system derived in our implicit numerical method. We follow [10] to use K formulation for representing the complex-valued matrix, with the sparsity pattern shown in Fig. 1.

It can be observed that the sparsity pattern of the complex-valued matrix is same to the real-valued one for the Poisson equation (3.4). Consequently, the AMG solver described above for the real-valued linear system can be reused, with the modification introduced by the K formulation. Briefly, the generation of both the restriction and prolongation operators, and the implementation of the iteration scheme in AMG will be in "block" style, since each ▪ in Fig. 1 represents a 2×2 matrix. In our implementation, a block Gauss-Seidel iteration scheme is used in the complex-valued AMG.

3.3. OpenMP parallelization of the algorithm

OpenMP is a mature technique for enhancing the implementation efficiency of the algorithm, by fully utilizing the hardware resource in a workstation. The research on improving the finite element codes by OpenMP keeps active [23].

In our work, we mainly use OpenMP in the two operations to improve the simulation. The first one is in the assembly of the stiffness matrix, and the second one is in the implementation of AMG solvers.

The utilization of the OpenMP in the assembly of the stiffness matrix is not straightforward since the possible data racing in writing the global matrix. An effective approach on resolving the data racing is to group those nonadjacent elements, then OpenMP can be used directly in each group for the assembly operation. In our method, a greedy algorithm is used to generate several groups in each of which every element is not adjacent to any other elements. It is worth mentioning that the computational complexity of the greedy algorithm is $\mathcal{O}(N_{tet})$, and the implementation of the algorithm is efficient with the help of the HGT data structure for the mesh. For the OpenMP in AMG solver, it can be trivially used in the matrix-vector product. In addition, if the Jacobi iteration is used as the smoother, the parallelization of the iteration can also be realized trivially.

In the next section, the performance of the proposed acceleration methods, as well as the performance of the implicit solver for the TDKS equation, will be checked by a number of numerical experiments.

4. Numerical experiments

The performance of the proposed numerical method will be shown in this section by a number of the numerical experiments. We focus on three aspects, i.e., the efficiency of the proposed complex-valued AMG solver for the linear system from the implicit numerical method, the effectiveness of the h-adaptive mesh method in solving the TDKS equation, as well as the performance of the proposed implicit numerical solver in the long term simulations.

The hardware for the simulations is a Dell Precision T7920 Tower, with two Intel(R) Xeon(R) Gold 6136 CPU @ 3.00GHz (total 24 cores), and with 256 gigabytes memory. The software for the simulation is a C++ library AFEABIC [2, 3] developed and maintained by the authors. It is noted that we also use AFEABIC to calculate the ground state of the given electronic structure, which is used in this paper as the initial condition for the simulations of the TDKS equation.

In all simulations, the computational domain is a ball with the radius 50 au, and the initial tetrahedral mesh is generated by Gmsh [13]. The ground state of the given electronic structure is obtained by using an h-adaptive finite element method proposed in [2] for the Kohn-Sham density functional theory, which has been realized in AFEABIC.

4.1. The performance of the AMG solver for the complex-valued system

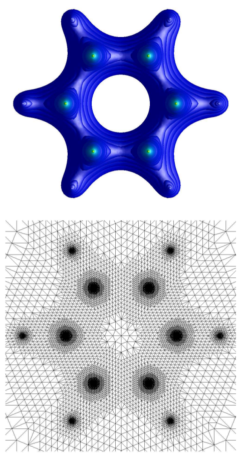
To show the effectiveness of the proposed AMG solver, several atoms and molecules are tested, and the results are shown in Table 1. The simulation process for each case in the table can be described as follows. First of all, the ground state of the given atom

Table 1: Comparison on the CPU time (millisecond) on solving the linear system by block Gauss-Seidel iteration and by multigrid iteration, respectively. The number in $(\cdot)$ is the iteration numbers used.

Stru.	DOFs	Block GS		MG	
Helium	153,761	310,630 ms	(6,900)	117,300 ms	(42)
Lithium	357,142	1,705,477 ms	(14,900)	206,213 ms	(49)
Beryllium	590,998	5,272,625 ms	(25,000)	297,384 ms	(46)
Boron	769,228	8,363,926 ms	(32,100)	341,742 ms	(50)
LiH	104,469	301,458 ms	(9,100)	121,277 ms	(50)
Li_2	159,206	452,573 ms	(9,200)	132,246 ms	(47)
BeH_2	155,548	548,938 ms	(11,400)	168,883 ms	(61)
Li_9 (BCC)	344,430	723,390 ms	(6300)	176,541 ms	(45)
CH_4	398,155	5,806,147 ms	(41,200)	649,829 ms	(144)
H_2O	508,908	10,600,067 ms	(58,200)	916,032 ms	(175)
C_6H_6	1,950,290	29,832,561 ms	(40,900)	1,354,015 ms	(90)

or molecule is obtained by an h-adaptive finite element method [2]. Then with this ground state as the initial state, the electronic structure system is propagated forward by using the proposed numerical method in this paper. We record the CPU time (in millisecond, ms) and the iteration steps needed for solving a complex-valued linear system by using the proposed complex-valued AMG solver. As a comparison, the corresponding results from the block Gauss-Seidel iteration are also listed. It is noted that the number shown in the parenthesis is the total number of the iteration steps. In all cases, the L_2 norm of the residual of the system is used to design the stop criterion, and the tolerance is 1.0×10^{-12}. In addition, the parameter n_s in Algorithm 3.2 is 3 in all simulations.

Before introducing the results in Table 1, it is noted that the ground state for each given atom or molecule is calculated accurately with our h-adaptive finite element method in [2], which consists of a self-consistent field (SCF) iteration for the linearization of the Kohn-Sham equation, and the locally optimal blocked preconditioned conjugate gradient (LOBPCG) method for solving the generalized eigenvalue problem. In all cases, the residual for each eigenpair is used to design the stop criterion for LOBPCG method with the tolerance 1.0×10^{-8}, and the L_2 norm of the difference from two adjacent iterations is used to design the stop criterion for SCF iteration with the tolerance 1.0×10^{-4}. Finally, in the process of the h-adaptive refinement of the mesh, the simulation is stopped when the total energy of the electronic structure changes within 0.1% compared with the total energy obtained from last mesh. As an example, Fig. 2 shows the results from the ground state simulation for benzene molecule, including the isosurfaces of the electron density and the mesh slice on the x-y plane around the molecule (in Fig. 2, left two subfigures), and the convergence history of the total energy of the benzene molecule with the h-adaptive refinement of the mesh (in Fig. 2, right figure). It is noted that the reference value for the total energy of the



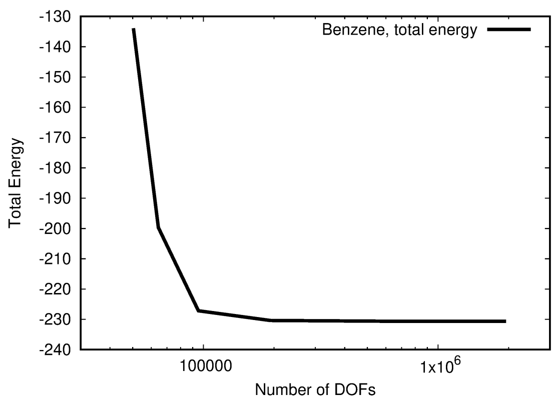


Figure 2: Calculation of the ground state for benzene molecule. Upper left: isosurface of the electron density. Bottom left: the mesh profile on the x-y plane around the molecule. Right: the convergence of the total energy of the benzene with the h-adaptive refinement of the mesh.

benzene molecule is -230.85 au from CCCBDB, from which we can see the numerical convergence of our simulation for benzene molecule.

Now we are ready to introduce the results in Table 1, and two main observations can be made from the table. First of all, the proposed complex-valued AMG solver in this paper significantly accelerates solving the linear system, compared with the block Gauss-Seidel iteration, for all cases. In most cases, no more than 10% CPU time is needed by AMG solver to solve the linear system, compared with the block Gauss-Seidel iteration. It is worth mentioning that in benzene case, the percentage is only around 4.5%. The second observation is that with the same parameters such as the tolerance for the stop criterion, the performance of the AMG solver is more stable than that of the block Gauss-Seidel iteration, for different electronic structure and total amount of the grid points in the mesh. This can be seen clearly from those numbers in the parentheses, i.e., the variance of the iteration numbers from block Gauss-Seidel iteration is 309898000, while it is 2066.1 for the proposed AMG solver in this paper.

It is known from the classical analysis for the multigrid method that the performance of the method is not sensitive to the condition number of the matrix for the convergent cases. We would like to mention that the complex-valued AMG solver proposed in this paper also has such property, which can be shown clearly in Table 2. It is noted that the parameter n_s in Algorithm 3.2 is 6 in all simulations in the table.

It is noted that in Table 2, for each atom or molecule, the different number of the grid points is from using different tolerance for h-adaptive refinement of the mesh in each simulation, and the tolerance for each simulation is given in the parenthesis. The nonsensitivity of the proposed AMG solver to the condition number of the matrix can be observed clearly from the above table, i.e., the change of the total number of the iteration needed for different simulations is small.

Before getting involved in the next subsection, we would also like to deliver the

Table 2: The total iteration numbers needed by AMG for each electronic structure with different number of the grid points.

Stru.	Grid Points	Iter.	Stru.	Grid Points	Iter.
Li	70,963 (5.0×10^{-3})	38	CH_4	137,533 (2.0×10^{-2})	68
	118,311 (3.0×10^{-3})	36		398,155 (1.0×10^{-2})	89
	357,142 (1.0×10^{-3})	36		575,040 (8.0×10^{-3})	86
	469,682 (8.0×10^{-4})	37		963,547 (5.0×10^{-3})	78
LiH	104,469 (5.0×10^{-3})	37	H_2O	50,908 (2.0×10^{-2})	102
	172,878 (3.0×10^{-3})	30		255,467 (1.0×10^{-2})	89
	544,501 (1.0×10^{-3})	29		644,037 (8.0×10^{-3})	82
	692,392 (8.0×10^{-4})	28		1,044,008 (5.0×10^{-3})	84

results of a study on the role of the number n_s in Algorithm 3.2 played in the complex-valued AMG solver. It is known that the smoother S_h in Algorithm 3.2 is used to damp out the high frequency part of the numerical error. Below in Table 3, the results of the performance of the proposed AMG solver obtained from different n_s in Algorithm 3.2 are listed for several atoms and molecules.

From Table 3, it can be observed from all cases that with the increment of the parameter n_s in Algorithm 3.2, the iteration steps needed for solving the linear system decrease. For example, in the simulation of H_2O molecule, when we use $n_s = 3$, the number of the iteration steps is 175, and the corresponding CPU time is $916,032$ ms. When we increase the parameter n_s from 3 to 9, the number of the iteration steps becomes 78, and the CPU time becomes $598,884$ ms. That means such change of the parameter n_s allows us to use around 45% iteration steps and around 65% CPU time to solve the same linear system. However, it does not mean that the efficiency can always be improved by increasing n_s. For example, in the same case for H_2O molecule, $n_s = 18$ brings us smaller number of the iteration steps, but more CPU time. A similar observation can be made for all cases in the above table. It should be pointed out that the best choice of the parameter n_s in Algorithm 3.2 should be problem dependent. However, from the results shown in Table 3 and based on our numerical experience, $n_s = 9$ is fairly a good choice. Hence, in the following simulations, we will keep using this value.

So far, we have discussed the performance of the proposed AMG solver in solving the linear system derived from the implicit temporal discretization. In the following, the effectiveness of the proposed h-adaptive mesh method will be demonstrated by two examples.

4.2. The effectiveness of the h-adaptive mesh method

We use two artificial examples to show the effectiveness of the proposed h-adaptive mesh method, as follows.

In the first example, the molecule is an H_2, with the initial positions for two nuclei

Table 3: The performance of the proposed AMG solver with different n_s in Algorithm 3.2.

Stru.	n_s	Iter.	CPU time (ms)	Stru.	n_s	Iter.	CPU time (ms)
Helium	3	42	117,300	H_2O	3	175	916,032
	6	31	91,027		6	102	670,123
	9	**25**	**85,237**		**9**	**78**	**598,884**
	12	22	81,692		12	66	618,059
	15	20	80,420		15	59	607,581
	18	19	85,462		18	54	622,983
	21	18	86,785		21	50	641,419
Lithium	3	49	206,213	CH_4	3	144	649,829
	6	36	177,185		6	89	493,016
	9	**31**	**182,271**		**9**	**71**	**460,736**
	12	27	180,271		12	62	467,982
	15	25	188,784		15	56	470,987
LiH	3	50	121,277	H_2O	3	175	916,032
	6	37	95,607		6	102	670,123
	9	**31**	**87,030**		**9**	**78**	**598,884**
	12	28	87,707		12	66	618,059
	15	25	86,412		15	59	607,581
	18	24	87,727		18	54	622,983
	21	22	86,938	Li_9	3	45	176,541
	24	21	87,845	BCC	6	35	166,746
	27	21	93,874		**9**	**29**	**160,826**
	30	20	93,544		12	26	177,077
	33	19	93,906		15	24	180,401

$(-0.7209, 0.0, 0.0)$ and $(0.7209, 0.0, 0.0)$, respectively. The simulation consists of two processes. Firstly, the ground state of the H_2 molecule is obtained by solving the Kohn-Sham equation with an h-adaptive finite element method in AFEABIC [2]. In the second process, the TDKS equation is solved with the proposed numerical method in this paper. To make the dynamics of the system nontrivial, the positions of two nuclei change to $(0.0, -0.7209, 0.0)$ and $(0.0, 0.7209, 0.0)$, respectively, at the initial time.

The dynamics of the H_2 molecule is shown in Fig. 3. In the top row, the iso-surfaces of the electron density of the molecule are shown at the time instants $t = 0, 0.05, 0.1, 0.2$, respectively, while they are the corresponding mesh grids around the molecule in the bottom row. Please note that the mesh on the x-y plane is shown here for clearly demonstrating the dynamics change of the mesh grids. It can be observed clearly that with a sudden change of the positions of two nuclei, the distribution of the electron density changes with the time evolution, under the Coulomb interaction. With the help of the proposed h-adaptive mesh method, this transition process is resolved very well.

The second example is similar to the first one, in which the molecule becomes a bo-

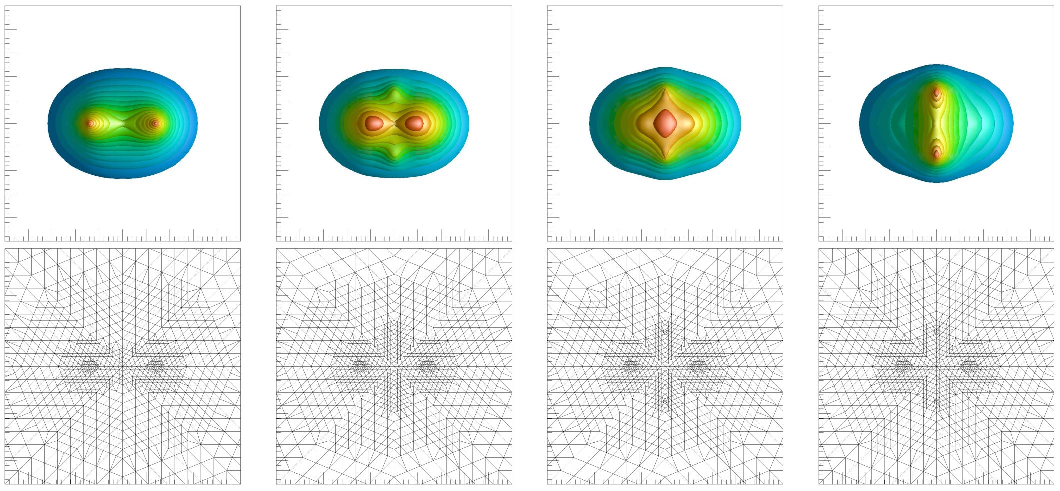


Figure 3: The isosurfaces of an H_2 molecule (top), and the corresponding mesh around the molecule (bottom), at $t = 0, 0.05, 0.1, 0.2$ (from left to right), respectively. The results are restricted in the box $[-2.5au, 2.5au] \times [-2.5au, 2.5au]$ in the x-y plane.

rane molecule (BH_3). The position of the nucleus of the boron atom is the origin point $(0, 0, 0)$, while the initial positions of the nuclei for three hydrogen atoms are $(0., 2.248773926, 0.)$, $(1.947551604, -1.124386963, 0.)$, and $(-1.947551604, -1.124386963, 0.)$, respectively. After the ground state of the borane molecule is obtained, the nuclei positions for three hydrogen atoms are changed to $(0., -2.248773926, 0.)$, $(1.947551604, 1.124386963, 0.)$, and $(-1.947551604, 1.124386963, 0.)$, i.e. their symmetric points w.r.t. the x-axis in the x-y plane. Then the dynamics of the electron density is obtained by solving the TDKS equation with the proposed numerical method. The numerical results at the time instants $t = 0, 0.36, 0.6$, and 1.0, are shown in Fig. 4. Again, it can be observed that the transition process caused by the sudden change of the positions of the nuclei is resolved very well, with our h-adaptive mesh method.

It is worth mentioning that the proposed complex-valued AMG solver works very stable in both simulations, i.e., in the whole simulation process of the H_2 molecule, only around 5 steps of the multigrid iteration are needed to solve the complex-valued linear system with the stop criterion 1.0×10^{-10} for the residual, while it is 15 multigrid iteration steps for the borane molecule.

4.3. The performance on long term simulations

In this subsection, we test the performance of the proposed numerical framework on long-term simulation of the time dependent Kohn-Sham equation. To check the ability of the numerical scheme on preserving the structures mentioned in Section 2.1, two kinds of the numerical experiments are implemented below.

The first kind of numerical experiments is to show the property of the numerical scheme on preserving the symplectic structure of the system. It is noted from (2.1) that

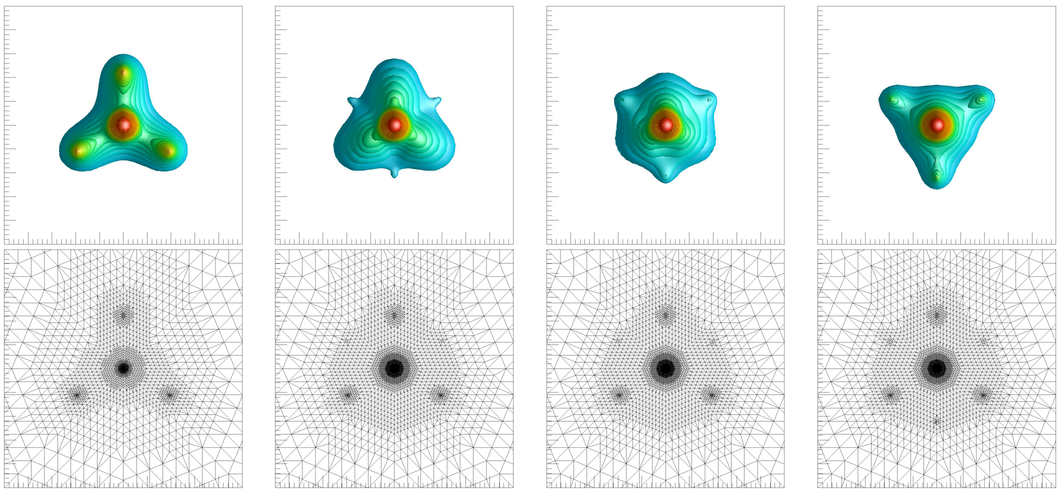


Figure 4: The isosurfaces of a Borane (BH_3) molecule (top), and the corresponding mesh around the molecule (bottom), at $t = 0, 0.36, 0.6, 1.0$ (from left to right), respectively. The results are restricted in the box $[-5au, 5au] \times [-5au, 5au]$ in the x-y plane.

the conservation of the Hamiltonian function is equivalent to the conservation of the total energy of the system. Hence, the simulation process can be described as follows. First of all, the ground state of the electronic system is calculated by using an h-adaptive finite element framework. Then by using this ground state as an initial condition, the electronic structure system is propagated freely. In this case, both the norm of the wavefunction and the total energy of the system should be preserved. The ability of the proposed numerical method in this paper on preserving the above properties is well demonstrated by the results of the H_2 molecule. The ground state of the H_2 molecule is obtained first. Then without the initial perturbation, the system is propagated by solving the TDKS equation with the proposed numerical method for over 1200 au in time, and the conservation of the total energy and the total number of the electrons is preserved very well. Please see Fig. 5.

The second kind of the numerical experiments is devoted to the conservation of the norm of the wavefunction under an initial perturbation on the system. Specifically, the ground state of the electronic structure is calculated first. Then the phase of the ground state wavefunction is shifted by $\psi = e^{-ikz}\psi$, where k is the amplitude of the perturbation, and z denotes the polarization direction. After that, we let the system evolve freely, and record the time-dependent dipole moment, and the norm of the wavefunction.

In Fig. 6, it obviously is seen that the number of the electrons for a helium atom, which should be 2, is preserved very well even for the long term simulations. The history of the time dependent dipole is also recorded in the simulation, and is shown in Fig. 6 (bottom one). Similar results can be found for the simulations of the LiH molecule (Fig. 7), and the methane molecule (Fig. 8).

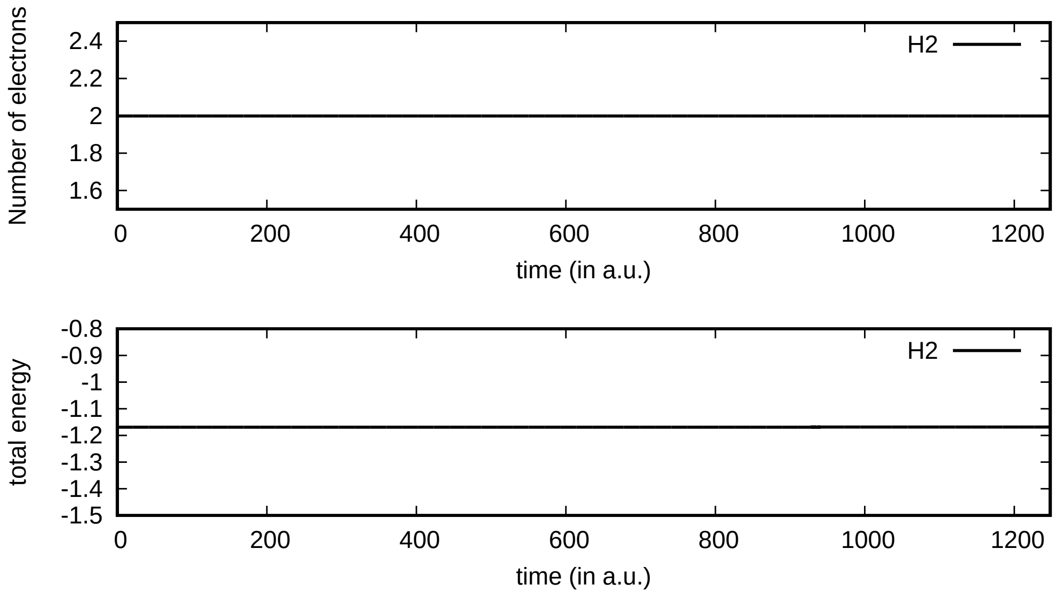


Figure 5: The results for H_2 molecule. Top: Number of the electrons in the system with the time evolution, which should be 2 for the H_2 molecule. Bottom: the history of the total energy with the time evolution.

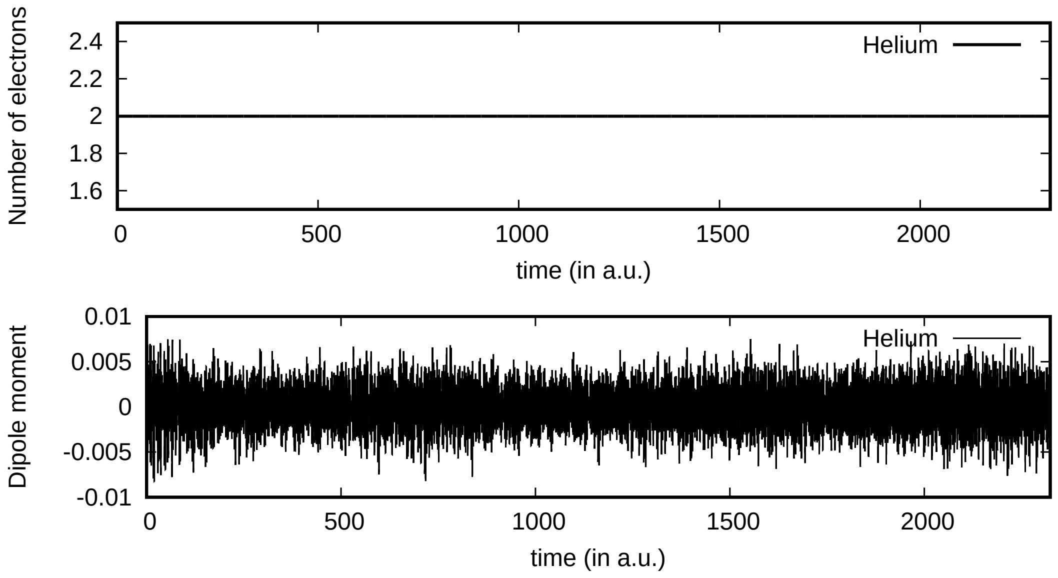


Figure 6: Top: Number of the electrons in the system with the time evolution, which should be 2 for the helium atom. Bottom: the history of the time dependent dipole in the simulation.

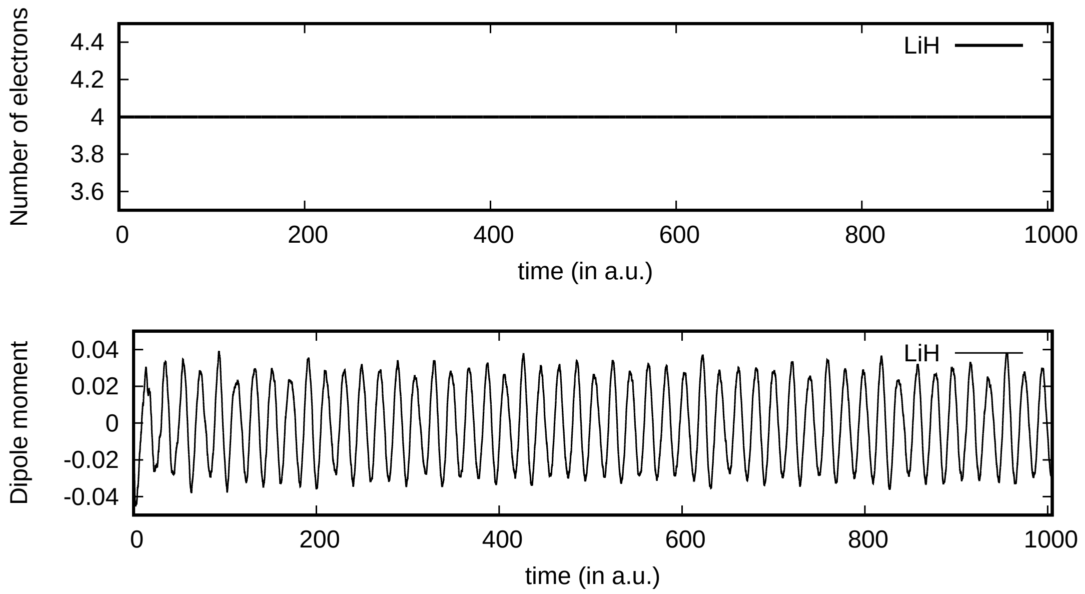


Figure 7: Top: Number of the electrons in the system with the time evolution, which should be 4 for the LiH molecule. Bottom: the history of the time dependent dipole in the simulation.

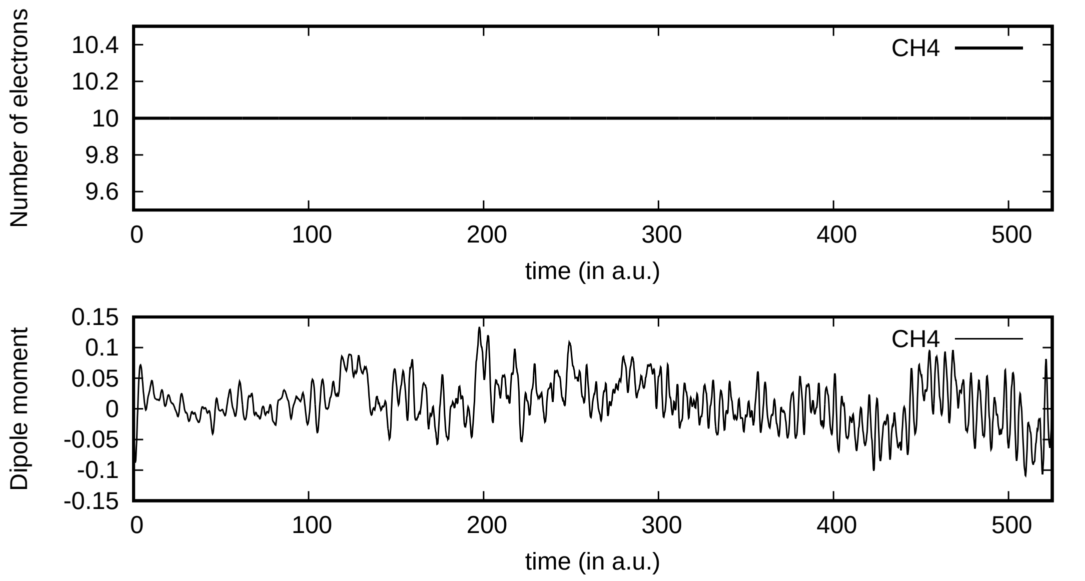


Figure 8: Top: Number of the electrons in the system with the time evolution, which should be 10 for the CH_4 molecule. Bottom: the history of the time dependent dipole in the simulation.

5. Conclusion

In this paper, an implicit solver is proposed for the TDKS equation. The solver consists of an implicit midpoint scheme for the temporal discretization, and a linear finite element scheme for the spatial discretization. To resolve the efficiency issue of the proposed implicit solver, a complex-valued AMG solver is designed for efficiently solving the linear system from the implicit scheme, an h-adaptive mesh method is developed based on a hierarchy geometry tree and a residual type *a posteriori* error estimation technique, and the algorithm is parallelized by OpenMP. A number of numerical experiments successfully show the effectiveness of the proposed method.

Quality long term simulation for a given electronic structure is necessary in a variety of applications such as the photonabsorption spectra calculation, simulations on the high harmonic generation, and molecular dynamics. A study towards these applications will be reported in our forthcoming paper.

Acknowledgements

The work of the first author was partially funded by the Science and Technology Development Fund, Macau SAR (Nos. 0070/2019/A2, 101/2016/A3), and National Natural Science Foundation of China (Grant No. 11701598). The work of the third author was partially supported by the National Natural Science Foundation of China (Grant No. 11601229), and the Natural Science Foundation of Jiangsu Province of China (Grant No. BK20160784). The work of the last author was partially supported by FDCT of Macao SAR (029/2016/A1), National Natural Science Foundation of China (Grant Nos. 11922120, 11871489, 11401608), Multi-Year research grant (2019-00154-FST, 2017-00189-FST) of University of Macau.

References

[1] X. Andrade, D. Strubbe, U. De Giovannini, A. H. Larsen, M. J. T. Oliveira, J. Alberdi-Rodriguez, A. Varas, I. Theophilou, N. Helbig, M. J. Verstraete, L. Stella, F. Nogueira, A. Aspuru-Guzik, A. Castro, M. A. L. Marques, and A. Rubio, *Real-space grids and the octopus code as tools for the development of new simulation approaches for electronic systems*, Phys. Chem. Chem. Phys., 17 (2015), 31371-31396.

[2] G. Bao, G. Hu, and D. Liu, *An h-adaptive finite element solver for the calculations of the electronic structures*, J. Comput. Phys., 231 (2012), 4967-4979.

[3] G. Bao, G. Hu, and D. Liu, *Real-time adaptive finite element solution of time-dependent Kohn-Sham equation*, J. Comput. Phys., 281 (2015), 743-758.

[4] A. Borzì, G. Ciaramella, and M. Sprengel, *Formulation and Numerical Solution of Quantum Control Problems*, Computational Science & Engineering, SIAM, 2017.

[5] A. Brandt and O. Livne, *Multigrid Techniques: 1984 Guide with Applications to Fluid Dynamics, Revised Edition*, Classics in Applied Mathematics, SIAM, 2011.

[6] M. Brezina, R. Falgout, S. MacLachlan, T. Manteuffel, S. McCormick, and J. Ruge, *Adaptive algebraic multigrid*, SIAM J. Sci. Comput., 27 (2006), 1261-1286.

[7] A. Castro, H. Appel, M. Oliveira, C. A. Rozzi, X. Andrade, F. Lorenzen, M. A. L. Marques, E. K. U. Gross, and A. Rubio, *Octopus: a tool for the application of time-dependent density functional theory*, Phys. Stat. Sol. (b), 243 (2006), 2465-2488.

[8] H. Chen, X. Dai, X. Gong, L. He, and A. Zhou, *Adaptive finite element approximations for Kohn-Sham models*, Multiscale Model. Simul., 12 (2014), 1828-1869.

[9] L. Chen and Y. Chen, *Multigrid method for poroelasticity problem by finite element method*, Adv. Appl. Math. Mech., 11 (2019), 1339-1357.

[10] D. Day and M. Heroux, *Solving complex-valued linear systems via equivalent real formulations*, SIAM J. Sci. Comput., 23 (2001), 480-498.

[11] X. Feng, H. Liu, and S. Ma, *Mass- and energy-conserved numerical schemes for nonlinear Schrödinger equations*, Commun. Comput. Phys., 26 (2019), 1365-1396.

[12] L. Genovese, B. Videau, M. Ospici, T. Deutsch, S. Goedecker, and J.-F. Méhaut, *Daubechies wavelets for high performance electronic structure calculations: The BigDFT project*, Comptes Rendus Mécanique, 339 (2011), 149-164. High Performance Computing.

[13] C. Geuzaine and J.-F. Remacle, *Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities*, Internat. J. Numer. Methods Engrg., 79 (2009), 1309-1331.

[14] A. Gómez Pueyo, M. A. L. Marques, A. Rubio, and A. Castro, *Propagators for the time-dependent Kohn-Sham equations: Multistep, Runge-Kutta, Exponential Runge-Kutta, and Commutator Free Magnus Methods*, J. Chem. Theory Comput., 14 (2018), 3040-3052. PMID: 29672048.

[15] D. Griffiths and D. J. Higham, *Numerical Methods for Ordinary Differential Equations*, Springer-Verlag, 2010.

[16] Y. Hong, J. Lu, J. Lin, and W. Chen, *Numerical simulation of non-linear Schrodinger equations in arbitrary domain by the localized method of approximate particular solution*, Adv. Appl. Math. Mech., 11 (2019), 108-131.

[17] B. Kanungo and V. Gavini, *Real time time-dependent density functional theory using higher order finite-element methods*, Phys. Rev. B, 100 (2019), 115148.

[18] I. Kwon and D. Y. Kwak, *Discontinuous bubble immersed finite element method for Poisson-Boltzmann equation*, Commun. Comput. Phys., 25 (2018), 928-946.

[19] R. Li, *On multi-mesh h-adaptive methods*, J. Sci. Comput., 24 (2005), 321-341.

[20] L. Lin, J. Lu, L. Ying, and W. E, *Adaptive local basis set for Kohn-Sham density functional theory in a discontinuous Galerkin framework I: Total energy calculation*, J. Comput. Phys., 231 (2012), 2140-2154.

[21] M. A. L. Marques, N. T. Maitra, F. M. S. Mogueira, E. K. U. Gross, and A. Rubio, eds., *Fundamentals of Time-Dependent Density Functional Theory*, Lecture Notes in Physics 837, Springer, 2012.

[22] E. Runge and E. K. U. Gross, *Density-functional theory for time-dependent systems*, Phys. Rev. Lett., 52 (1984), 997-1000.

[23] S. M. H. Sefidgar, A. R. Firoozjaee, and M. Dehestani, *Parallelization of torsion finite element code using compressed stiffness matrix algorithm*, Engineering with Computers, 2020.

[24] M. Sprengel, G. Ciaramella, and A. Borzì, *A COKOSNUT code for the control of the time-dependent Kohn-Sham model*, Comput. Phys. Commun., 214 (2017), 231-238.

[25] C. A. Ullrich, *Time-Dependent Density-Functional Theory*, Oxford University Press, 2011.

[26] R. Verfürth, *A Posteriori Error Estimation Techniques for Finite Element Methods*, Oxford University Press, 2013.

[27] L. YANG AND G. HU, *An adaptive finite element solver for demagnetization field calculation*, Adv. Appl. Math. Mech., 11 (2019), 1048-1063.

[28] J. ZHANG, D. LI, AND X. ANTOINE, *Efficient numerical computation of time-fractional nonlinear Schrödinger equations in unbounded domain*, Commun. Comput. Phys., 25 (2018), 218-243.