

3D-GTDSE: A GPU-based code for solving 3D-TDSE in Cartesian coordinates

Ke Peng(彭科), Aihua Liu(刘爱华), Jun Wang(王俊), and Xi Zhao(赵曦)
Citatio: Chin. Phys. B, 2025, 34 (**9**): 094203. DOI: 10.1088/1674-1056/adee00
http://cpb.iphy.ac.cn; http://iopscience.iop.org/cpb

What follows is a list of articles you may be interested in

Elliptically polarized high-order harmonic generation of Ar atom in an intense laser field

Jie Hu(胡杰), Yi-Chen Wang(王一琛), Qiu-Shuang Jing(景秋霜), Wei Jiang(姜威), Ge-Wen Wang(王革文), Yi-Wen Zhao(赵逸文), Bo Xiao(肖礴), Hong-Jing Liang(梁红静), and Ri Ma(马日)
Chin. Phys. B, 2024, 33 (**5**): 054208. DOI: 10.1088/1674-1056/ad2604

Calibration of quantitative rescattering model for simulating vortex high-order harmonic generation driven by Laguerre-Gaussian beam with nonzero orbital angular momentum

Jiaxin Han(韩嘉鑫), Zhong Guan(管仲), Beiyu Wang(汪倍羽), and Cheng Jin(金成)
Chin. Phys. B, 2023, 32 (**12**): 124210. DOI: 10.1088/1674-1056/acfa86

Tailoring OAM spectrum of high-order harmonic generation driven by two mixed Laguerre-Gaussian beams with nonzero radial nodes

Beiyu Wang(汪倍羽), Jiaxin Han(韩嘉鑫), and Cheng Jin(金成)
Chin. Phys. B, 2023, 32 (**12**): 124208. DOI: 10.1088/1674-1056/ad0144

Spectral shift of solid high-order harmonics from different channels in a combined laser field

Dong-Dong Cao(曹冬冬), Xue-Fei Pan(潘雪飞), Jun Zhang(张军), and Xue-Shen Liu(刘学深)
Chin. Phys. B, 2023, 32 (**3**): 034204. DOI: 10.1088/1674-1056/aca4bf

Effect of laser focus in two-color synthesized waveform on generation of soft x-ray high harmonics

Yanbo Chen(陈炎波), Baochang Li(李保昌), Xuhong Li(李胥红), Xiangyu Tang(唐翔宇), Chi Zhang(张弛), and Cheng Jin(金成)
Chin. Phys. B, 2023, 32 (**1**): 014203. DOI: 10.1088/1674-1056/ac8cea

---

COMPUTATIONAL PROGRAMS FOR PHYSICS

3D-GTDSE: A GPU-based code for solving 3D-TDSE in Cartesian coordinates

Ke Peng(彭科)[1], Aihua Liu(刘爱华)[2,†], Jun Wang(王俊)[2,‡], and Xi Zhao(赵曦)[1,§]

[1] *School of Physics and Information Technology, Shaanxi Normal University, Xi'an* 710062, *China*
[2] *Institute of Atomic and Molecular Physics, Jilin University, Changchun* 130012, *China*

(Received 24 April 2025; revised manuscript received 25 June 2025; accepted manuscript online 10 July 2025)

We present a graphics processing units (GPU) parallelization based three-dimensional time-dependent Schrödinger equation (3D-TDSE) code to simulate the interaction between single-active-electron atom/molecule and arbitrary types of laser pulses with either velocity gauge or length gauge in Cartesian coordinates. Split-operator method combined with fast Fourier transforms (FFT) is used to perform the time evolution. Sample applications in different scenarios, such as stationary state energies, photon ionization spectra, attosecond clocks, and high-order harmonic generation (HHG), are given for the hydrogen atom. Repeatable results can be obtained with the benchmark program PCTDSE, which is a 3D-TDSE Fortran solver parallelized using message passing interface (MPI) library. With the help of GPU acceleration and vectorization strategy, our code running on a single NVIDIA 3090 RTX GPU can achieve about 10 times faster computation speed than PCTDSE running on a 144 Intel Xeon CPU cores server with the same accuracy. In addition, 3D-GTDSE can also be modified slightly to simulate non-adiabatic dynamics involving the coupling of nuclear and electronic wave packets, as well as pure nuclear wave packet dynamics in the presence of strong laser fields within 3 dimensions. Additionally, we have also discussed the limitations and shortcomings of our code in utilizing GPU memory. The 3D-GTDSE code provides an alternative tool for studying the ultrafast nonlinear dynamics under strong laser fields.

Keywords: GPU parallelization, high-order harmonic generation (HHG), time-dependent Schrödinger equation (TDSE), wave packet dynamics

PACS: 42.65.Ky, 32.80.Rm, 31.15.xv

DOI: 10.1088/1674-1056/adee00 **CSTR:** 32038.14.CPB.adee00

1. Introduction

Over the last decades, the development of extreme light sources has allowed experimentalists to probe and control a series of nonlinear dynamics of electrons triggered by strong laser pulses at even attosecond time scales,[1–4] such as photon ionization momentum distribution, high-order harmonic generation (HHG), and photon ionization spectra.[5–12] From a computational perspective, all the information of such dynamics can be retrieved from the wavefunctions upon numerical *ab initio* solving of the time-dependent Schrödinger equation (TDSE).[13,14] Theoretical discussions of the interaction between target systems and laser pulses can be restricted to the single active electron approximation (SAEA) if multi-electron effects can be neglected or are not of concern.[15–18] However, due to the computational costs required for solving the TDSE growing exponentially with respect to the degrees of freedom, simulating large sizes of even 3D space is still a challenging task. For example, large-scale nano-systems have only been simulated successfully under the Born–Oppenheimer approximation for single active electron problems.[19] Various parallelized schemes for solving the TDSE have been developed to accelerate the computations. The emergence of central processing unit (CPU)-based parallel architectures allowed the development of high performance programming languages (Fortran, Python, or C-like languages) and the successful usage of message passing interface (MPI[20]) and open multi-processing (OpenMP[21]), such as Qprop,[22] QPC-TDSE,[23] SemiGlobalCpp,[24] tRecX,[25] RMT,[26] SCID-TDSE,[27] and MP-CITDSE.[28] Meanwhile, the use of a heterogeneous computational environment, which involves the use of GPUs for higher parallelism with a lower-cost computing system, presents an alternative choice.

The main advantage of GPU architectures is their large number of physical cores and broadband memory bandwidth. Such advantages make GPUs an attractive architecture to run computationally intensive algorithms, such as solving large-sized TDSEs. The GPU parallelization model has so far appeared with different programming levels such as compute unified device architecture (CUDA[29]), OpenCL,[30] and OpenACC.[31] Based on these programming models, many TDSE solvers have been developed: HeTDSE[32] is an OpenACC based *ab initio* numerical simulation package to solve

†Corresponding author. E-mail: aihualiu@jlu.edu.cn
‡Corresponding author. E-mail: wangjun86@jlu.edu.cn
§Corresponding author. E-mail: zhaoxi719@snnu.edu.cn

© 2025 Chinese Physical Society and IOP Publishing Ltd. All rights, including for text and data mining, AI training, and similar technologies, are reserved.

the full-dimensional TDSE of helium subjected to a strong laser pulse. The B-spline basis sets are employed to construct the radial part of the wavefunction, while the spherical harmonic function is used to express the angular part. CLTDSE[33] is an OpenCL-based single active electron 3D TDSE solver. Similar to HeTDSE, CLTDSE expands the wavefunction in spherical coordinates, which can easily distinguish the contribution of each quantum state $|nlm\rangle$ to the dynamics, and converges quickly if the photon-induced transition occurs between states with small azimuthal quantum number l. However, when strong infrared (IR) laser pulses are used to excite the system, then large l and corresponding magnetic quantum number m are needed, which slows down the convergence. Additionally, the above codes are restricted to linearly polarized light in the current version.

Cartesian coordinates, which are favorable for scalability and generality, have the ability to overcome the above restrictions. QnDynCUDA[34] is a CUDA-based TDSE code running on a 2D Cartesian-coordinate space. Some real-time time-dependent density functional theory (RT-TDDFT[35]) programs, such as Octopus[36,37] and QDD,[38] which solve $N\times$ 3D Kohn–Sham equations for N-electron systems in 3D Cartesian-coordinate space, also provide a GPU parallelization option.

Here in this work, we provide a GPU-parallelized TDSE solver, 3D-GTDSE, running on 3D Cartesian coordinates. 3D-GTDSE can simulate the electron dynamics of an SAEA atom or molecule interacting with arbitrary types of laser pulses, with or without the dipole approximation,[39,40] such as HHG, photoionization spectra, attoclock momentum distribution, and photon-induced excitation between bound states. We apply the MATLAB GPU parallelization scheme to accelerate our program, as MATLAB offers the opportunity to accelerate our codes with minimal effort and without extensive knowledge of GPU hardware architectures or CUDA, OpenCL, and OpenACC programming skills. Additionally, MATLAB provides other toolboxes that can further enhance GPU computing efficiency, such as array operations. By leveraging the extensibility and generality of the Cartesian-coordinate system, our code can be easily modified to extend to four-dimensional or higher-dimensional spaces. It can also readily simulate nuclear wave packet molecular dynamics processes or the coupled nuclear–electronic wave packet dynamics under the nonadiabatic approximation.

In addition, we have also discussed the limitations and shortcomings of our code, particularly regarding GPU memory usage. As is well known, data transfer between GPUs in multi-GPU nodes must go through the PCIe bus, which introduces significant latency compared to multi-CPU parallelism. For algorithms that require extensive data exchange, this results in a noticeable reduction in the efficiency of GPU parallelization. Currently, our code does not implement multi-GPU parallelization for the FFT operations during the time evolution process. Addressing this limitation will be a focus of our future work. As a result, the performance of our code is constrained by the total available GPU memory. However, for tasks that involve relatively low data exchange, such as scanning potential energy curves, our code does offer the option for multi-GPU parallel execution.

The rest of this paper is organized as follows. In Section 2 we present the theoretical background on 3D-GTDSE. In Section 3, we explain how MATLAB accelerates our code. In Section 4, we exhibit an overview of the package structure. In Section 5, we show several test applications of 3D-GTDSE and compare our results with other codes, especially with another 3D TDSE code PCTDSE[41] which runs on Cartesian coordinates as well. In Section 6, we compare the parallel efficiency of 3D-GTDSE with PCTDSE and give the memory usage of the program under different grid sizes. We present our conclusions in Section 7.

Atomic units are used throughout unless stated otherwise.

2. Theory

2.1. Splitting-operator method

A system wavefunction $\psi(\boldsymbol{r},t)$ with SAEA evolving with time can be described by TDSE

$$\mathrm{i}\frac{\partial\psi(\boldsymbol{r},t)}{\partial t}=\hat{H}(t)\,\psi(\boldsymbol{r},t), \tag{1}$$

where $\hat{H}(t)$ is the Hamiltonian operator. Within dipole approximation, the velocity gauge (VG) $\hat{H}(t)$ reads

$$\hat{H}(t)=\frac{[\boldsymbol{p}+\boldsymbol{A}(t)]^2}{2}+V(\boldsymbol{r}), \tag{2}$$

where $\boldsymbol{p}=-\mathrm{i}\nabla$ is the canonical momentum operator, $V(\boldsymbol{r})$ is the Coulomb potential, $\boldsymbol{A}(t)$ is the vector potential, which is associated with the electric field by $\boldsymbol{E}(t)=-\partial\boldsymbol{A}(t)/\partial t$. Another formally equivalent choice is the length gauge (LG) Hamiltonian

$$\hat{H}(t)=\frac{-\nabla^2}{2}+\boldsymbol{r}\cdot\boldsymbol{E}(t)+V(\boldsymbol{r}). \tag{3}$$

Equation (1) has a formal solution expressed by the time-evolution operator

$$\psi(\boldsymbol{r},t)=\exp\left(-\mathrm{i}\int_{t_0}^{t}\hat{H}(t_1)\,\mathrm{d}t_1\right)\psi(\boldsymbol{r},t_0), \tag{4}$$

which connects the wavefunction at any time t and initial time t_0. The propagation of wavefunction can be implemented by the split-operator method[42]

$$\begin{aligned}&\psi(\boldsymbol{r},t_0+\Delta t)\\ &=\mathrm{e}^{-\mathrm{i}\frac{1}{4}(\boldsymbol{p}+\boldsymbol{A})^2\Delta t}\,\mathrm{e}^{-\mathrm{i}V\Delta t}\,\mathrm{e}^{-\mathrm{i}\frac{1}{4}(\boldsymbol{p}+\boldsymbol{A})^2\Delta t}\psi(\boldsymbol{r},t_0)+\mathrm{O}(\Delta t)^3\end{aligned} \tag{5}$$

and FFT.[43] Details about the split-operator method and FFT can be found in Ref. [44].

2.2. Initial state preparation

The 3D-GTDSE provides the initial state through the imaginary time evolution (ITE) method.[45] The basic idea of ITE is replacing the real time step $\mathrm{d}t$ with an imaginary time $-\mathrm{i}\,\mathrm{d}t$ in Eq. (5), implementing the time propagation with the laser-free Hamiltonian $\hat{H}_0 = \boldsymbol{p}^2/2 + V(\boldsymbol{r})$ from a guessed wavefunction $\psi_{\text{guess}}(\boldsymbol{r}, t=0)$. Repeat this evolution till the eigenvalues $E = \frac{\langle\psi(\boldsymbol{r})|\hat{H}_0|\psi(\boldsymbol{r})\rangle}{\langle\psi(\boldsymbol{r})|\,\psi(\boldsymbol{r})\rangle}$ of the wavefunctions converge.

2.3. Mask function

When numerically solving the TDSE in a fixed-size box, an absorbing function $M(\boldsymbol{r})$ is needed to avoid the unphysical reflection of the wave function at the boundary. The widely used form is

$$M(\boldsymbol{r}) = \begin{cases} 1, & \boldsymbol{r} < \boldsymbol{r}_0, \\ \sin^{\frac{1}{8}}\left(\frac{\pi(\boldsymbol{r}_m - \boldsymbol{r})}{2(\boldsymbol{r}_m - \boldsymbol{r}_0)}\right), & \boldsymbol{r}_0 < \boldsymbol{r} < \boldsymbol{r}_m, \\ 0, & \text{else}. \end{cases} \tag{6}$$

Here, $\boldsymbol{r}_0$ to $\boldsymbol{r}_m$ is the damping zone. The damped zone has to be far enough away from the nucleus so that the physical system is undisturbed by the absorbing boundary.

2.4. High harmonic generation

We can calculate the expectation value of the induced dipole moment $\langle\boldsymbol{r}(t)\rangle$, dipole velocity $\langle\dot{\boldsymbol{r}}(t)\rangle$, or dipole acceleration $\langle\ddot{\boldsymbol{r}}(t)\rangle$ to get HHG, both in LG and VG. In VG, we have

$$\langle\boldsymbol{r}(t)\rangle = \langle\psi_V(\boldsymbol{r},t)\,|\boldsymbol{r}|\,\psi_V(\boldsymbol{r},t)\rangle, \tag{7}$$

$$\langle\dot{\boldsymbol{r}}(t)\rangle = \langle\psi_V(\boldsymbol{p},t)\,|\boldsymbol{p}+\boldsymbol{A}(t)|\,\psi_V(\boldsymbol{p},t)\rangle, \tag{8}$$

$$\langle\ddot{\boldsymbol{r}}(t)\rangle = \left\langle\psi_V(\boldsymbol{r},t)\left|-\frac{\partial V(\boldsymbol{r})}{\partial r} - \boldsymbol{E}(t)\right|\psi_V(\boldsymbol{r},t)\right\rangle. \tag{9}$$

While in LG, the formulas are

$$\langle\boldsymbol{r}(t)\rangle = \langle\psi_L(\boldsymbol{r},t)\,|\boldsymbol{r}|\,\psi_L(\boldsymbol{r},t)\rangle, \tag{10}$$

$$\langle\dot{\boldsymbol{r}}(t)\rangle = \langle\psi_L(\boldsymbol{p},t)\,|\boldsymbol{p}|\,\psi_L(\boldsymbol{p},t)\rangle, \tag{11}$$

$$\langle\ddot{\boldsymbol{r}}(t)\rangle = \left\langle\psi_L(\boldsymbol{r},t)\left|-\frac{\partial V(\boldsymbol{r})}{\partial r}\right|\psi_L(\boldsymbol{r},t)\right\rangle + \boldsymbol{E}(t). \tag{12}$$

The HHG spectrum is given by

$$S_\zeta(\omega) = \frac{1}{2\pi}\left|\int_{-\infty}^{\infty}\zeta(t)\,\mathrm{e}^{\mathrm{i}\omega t}\,\mathrm{d}t\right|^2, \tag{13}$$

where $\varsigma(t)$ is $\langle\boldsymbol{r}\rangle$, $\langle\dot{\boldsymbol{r}}\rangle$ or $\langle\ddot{\boldsymbol{r}}\rangle$.

2.5. Photoelectron momentum distribution and photoelectron spectrum

In order to get the photoelectron momentum distribution (PMD) $f(\boldsymbol{p}, t_\mathrm{f})$ at time t_f, we gather the wave function absorbed by the boundary at each intermediate time t_c:

$$f(\boldsymbol{p},t_\mathrm{f}) = \left|\sum_{t_\mathrm{c}}\mathrm{e}^{-\mathrm{i}\int_{t_\mathrm{c}}^{t_\mathrm{f}}\frac{[\boldsymbol{p}+\boldsymbol{A}(t)]^2}{2}\mathrm{d}t}\,\tilde{\psi}_{\text{flux}}(\boldsymbol{r},t_\mathrm{c})\right|^2, \tag{14}$$

where $\psi_{\text{flux}}(\boldsymbol{r},t_\mathrm{c}) = (1-M(\boldsymbol{r}))\psi(\boldsymbol{r},t_\mathrm{c})$, $\tilde{\psi}_{\text{flux}}(\boldsymbol{r},t_\mathrm{c})$ is the Fourier transform of $\psi_{\text{flux}}(\boldsymbol{r},t_\mathrm{c})$, and $\mathrm{e}^{-\mathrm{i}\int_{t_\mathrm{c}}^{t_\mathrm{f}}\frac{[\boldsymbol{p}+\boldsymbol{A}(t)]^2}{2}\mathrm{d}t}$ is the Volkov evolution operator.[41]

The photoelectron spectrum can be obtained from PMD $f(\boldsymbol{p},t_\mathrm{f})$

$$P(E,t_\mathrm{f}) = \int f\left(\frac{\boldsymbol{p}^2}{2} = E, t_\mathrm{f}\right)\mathrm{d}\boldsymbol{p}. \tag{15}$$

2.6. Non-dipole apporximation

The 3D-GTDSE can also simulate the interaction between a target and a vector potential without dipole approximation (ND). By making a gauge transformation of the wavefunction from VG $\psi_V(\boldsymbol{r},t)$ to LG $\psi_L(\boldsymbol{r},t)$:

$$\psi(\boldsymbol{r},t) = \mathrm{e}^{\mathrm{i}F(\boldsymbol{r},t)}\psi_L(\boldsymbol{r},t). \tag{16}$$

Here, F is an undetermined function. In the following, we will rigorously derive the form that F must satisfy. Inserting Eq. (16) into TDSE

$$\mathrm{i}\frac{\partial}{\partial t}\psi(\boldsymbol{r},t) = \left[-\frac{\nabla^2}{2} - \mathrm{i}\boldsymbol{A}\cdot\nabla + \frac{\boldsymbol{A}^2}{2} + V(\boldsymbol{r})\right]\psi(\boldsymbol{r},t), \tag{17}$$

we have

$$\begin{aligned} &\mathrm{i}\frac{\partial}{\partial t}\psi(\boldsymbol{r},t) \\ &= \left\{\mathrm{i}\frac{\partial}{\partial t}\mathrm{e}^{\mathrm{i}F(\boldsymbol{r},t)}\right\}\psi_L(\boldsymbol{r},t) + \mathrm{e}^{\mathrm{i}F(\boldsymbol{r},t)}\left\{\mathrm{i}\frac{\partial}{\partial t}\psi_L(\boldsymbol{r},t)\right\} \\ &= -\dot{F}\,\mathrm{e}^{\mathrm{i}F}\psi_L + \mathrm{i}\,\mathrm{e}^{\mathrm{i}F}\dot{\psi}_L, \end{aligned} \tag{18}$$

$$\begin{aligned} &\left[-\frac{\nabla^2}{2} - \mathrm{i}\boldsymbol{A}\cdot\nabla + \frac{\boldsymbol{A}^2}{2} + V(\boldsymbol{r})\right]\mathrm{e}^{\mathrm{i}F(\boldsymbol{r},t)}\psi_L(\boldsymbol{r},t) \\ &= \alpha + \beta + \chi, \end{aligned} \tag{19}$$

where

$$\alpha = -\frac{\nabla^2}{2}\left\{\mathrm{e}^{\mathrm{i}F(\boldsymbol{r},t)}\psi_L(\boldsymbol{r},t)\right\}, \tag{20}$$

$$\beta = -\mathrm{i}\boldsymbol{A}\cdot\nabla\left\{\mathrm{e}^{\mathrm{i}F(\boldsymbol{r},t)}\psi_L\right\}, \tag{21}$$

$$\chi = \left(V + \frac{\boldsymbol{A}^2}{2}\right)\mathrm{e}^{\mathrm{i}F(\boldsymbol{r},t)}\psi_L. \tag{22}$$

The first term

$$\begin{aligned} \alpha &= -\frac{\nabla^2}{2}\cdot\nabla\left\{\mathrm{e}^{\mathrm{i}F}\psi_L\right\} \\ &= -\frac{\nabla}{2}\cdot\left\{\mathrm{e}^{\mathrm{i}F}\nabla\psi_L + \psi_L\nabla\left(\mathrm{e}^{\mathrm{i}F}\right)\right\} \\ &= -\frac{\nabla}{2}\cdot\left\{\mathrm{e}^{\mathrm{i}F}\nabla\psi_L + \mathrm{i}\psi_L\mathrm{e}^{\mathrm{i}F}\nabla F\right\} \\ &= -\frac{1}{2}\left\{\nabla\cdot\left[\mathrm{e}^{\mathrm{i}F}\nabla\psi_L\right] + \nabla\cdot\left[\mathrm{i}\psi_L\mathrm{e}^{\mathrm{i}F}\nabla F\right]\right\} \\ &= -\frac{1}{2}\{\left(\nabla\mathrm{e}^{\mathrm{i}F}\right)\cdot\nabla\psi_L + \mathrm{e}^{\mathrm{i}F}\nabla^2\psi_L \\ &\quad + \left[\nabla\left(\mathrm{i}\psi_L\mathrm{e}^{\mathrm{i}F}\right)\right]\cdot\nabla F + \mathrm{i}\psi_L\mathrm{e}^{\mathrm{i}F}\nabla^2 F\} \end{aligned}$$

$$
\begin{aligned}
&= -\frac{1}{2}\{\mathrm{i}\,\mathrm{e}^{\mathrm{i}F}\nabla F\cdot\nabla\psi_L+\mathrm{e}^{\mathrm{i}F}\nabla^2\psi_L\\
&\quad+\left[\mathrm{i}\,\mathrm{e}^{\mathrm{i}F}\nabla\psi_L+\mathrm{i}\psi_L\nabla\mathrm{e}^{\mathrm{i}F}\right]\cdot\nabla F+\mathrm{i}\psi_L\mathrm{e}^{\mathrm{i}F}\nabla^2F\}\\
&= -\frac{1}{2}\{2\mathrm{i}\,\mathrm{e}^{\mathrm{i}F}\nabla F\cdot\nabla\psi_L+\mathrm{e}^{\mathrm{i}F}\nabla^2\psi_L\\
&\quad-\psi_L\mathrm{e}^{\mathrm{i}F}(\nabla F)^2+\mathrm{i}\psi_L\mathrm{e}^{\mathrm{i}F}\nabla^2F\},
\end{aligned}\tag{23}
$$

the second term

$$
\begin{aligned}
\beta &= -\mathrm{i}\boldsymbol{A}\cdot\nabla\left\{\mathrm{e}^{\mathrm{i}F}\psi_L\right\}\\
&= -\mathrm{i}\boldsymbol{A}\cdot\left\{\mathrm{e}^{\mathrm{i}F}\nabla\psi_L+\mathrm{i}\,\mathrm{e}^{\mathrm{i}F}\psi_L\nabla F\right\}\\
&= -\mathrm{i}\,\mathrm{e}^{\mathrm{i}F}\boldsymbol{A}\cdot\nabla\psi_L+\mathrm{e}^{\mathrm{i}F}\psi_L\boldsymbol{A}\cdot\nabla F,
\end{aligned}\tag{24}
$$

then we have

$$
\begin{aligned}
&\alpha+\beta+\chi\\
&= -\frac{1}{2}\left\{2\mathrm{i}\,\mathrm{e}^{\mathrm{i}F}\nabla F\cdot\nabla\psi_L+\mathrm{e}^{\mathrm{i}F}\nabla^2\psi_L\right.\\
&\quad\left.-\psi_L\mathrm{e}^{\mathrm{i}F}(\nabla F)^2+\mathrm{i}\psi_L\mathrm{e}^{\mathrm{i}F}\nabla^2F\right\}\\
&\quad-\mathrm{i}\,\mathrm{e}^{\mathrm{i}F}\boldsymbol{A}\cdot\nabla\psi_L+\mathrm{e}^{\mathrm{i}F}\psi_L\boldsymbol{A}\cdot\nabla F\\
&\quad+\left(V+\frac{\boldsymbol{A}^2}{2}\right)\mathrm{e}^{\mathrm{i}F}\psi_L\\
&= \mathrm{i}\frac{\partial}{\partial t}\psi(\boldsymbol{r},t)\\
&= -\dot{F}\,\mathrm{e}^{\mathrm{i}F}\psi_L+\mathrm{i}\,\mathrm{e}^{\mathrm{i}F}\dot{\psi}_L-\mathrm{i}\,\mathrm{e}^{\mathrm{i}F}\nabla F\cdot\nabla\psi_L-\frac{1}{2}\mathrm{e}^{\mathrm{i}F}\nabla^2\psi_L\\
&\quad+\frac{1}{2}\psi_L\mathrm{e}^{\mathrm{i}F}(\nabla F)^2-\frac{\mathrm{i}}{2}\psi_L\mathrm{e}^{\mathrm{i}F}\nabla^2F\\
&\quad-\mathrm{i}\,\mathrm{e}^{\mathrm{i}F}\boldsymbol{A}\cdot\nabla\psi_L+\mathrm{e}^{\mathrm{i}F}\psi_L\boldsymbol{A}\cdot\nabla F+\left(V+\frac{\boldsymbol{A}^2}{2}\right)\mathrm{e}^{\mathrm{i}F}\psi_L\\
&= -\dot{F}\,\mathrm{e}^{\mathrm{i}F}\psi_L+\mathrm{i}\,\mathrm{e}^{\mathrm{i}F}\dot{\psi}_L\\
&\quad-\mathrm{i}\nabla F\cdot\nabla\psi_L-\frac{1}{2}\nabla^2\psi_L+\frac{1}{2}\psi_L(\nabla F)^2-\frac{\mathrm{i}}{2}\psi_L\nabla^2F\\
&\quad-\mathrm{i}\boldsymbol{A}\cdot\nabla\psi_L+\psi_L\boldsymbol{A}\cdot\nabla F+\left(V+\frac{\boldsymbol{A}^2}{2}\right)\psi_L+\dot{F}\psi_L\\
&= \mathrm{i}\dot{\psi}_L,
\end{aligned}\tag{25}
$$

finally we have

$$
\begin{aligned}
\mathrm{i}\dot{\psi}_L = \Bigg\{&-\frac{1}{2}\nabla^2-\mathrm{i}\nabla F\cdot\nabla-\mathrm{i}\boldsymbol{A}\cdot\nabla+\frac{1}{2}(\nabla F)^2\\
&-\frac{\mathrm{i}}{2}\nabla^2F+\boldsymbol{A}\cdot\nabla F+\frac{\boldsymbol{A}^2}{2}+\dot{F}+V\Bigg\}\psi_L.
\end{aligned}\tag{26}
$$

Function F should satisfy the following equation:

$$
\begin{aligned}
&-\mathrm{i}\nabla F\cdot\nabla-\mathrm{i}\boldsymbol{A}\cdot\nabla+\frac{1}{2}(\nabla F)^2\\
&-\frac{\mathrm{i}}{2}\nabla^2F+\boldsymbol{A}\cdot\nabla F+\frac{\boldsymbol{A}^2}{2}=0,
\end{aligned}\tag{27}
$$

the solution is

$$
\nabla F=-\boldsymbol{A},\tag{28}
$$

then F is

$$
F=-\int_C\boldsymbol{A}(\boldsymbol{r},t)\cdot\mathrm{d}\boldsymbol{r},\tag{29}
$$

where C is a path from a reference point $\boldsymbol{r}_0$ to the point $\boldsymbol{r}$.

Finally we have the ND TDSE in LG

$$
\mathrm{i}\frac{\partial\psi(\boldsymbol{r},t)}{\partial t}=\left\{\frac{-\nabla^2}{2}+\int_C\boldsymbol{E}(\boldsymbol{r},t)\cdot\mathrm{d}\boldsymbol{r}+V(\boldsymbol{r})\right\}\psi(\boldsymbol{r},t).\tag{30}
$$

In the current version, solving ND TDSE can only be done in LG.

3. Accelerate the calculations by MATLAB

MATLAB provides several ways to accelerate code, particularly through GPU computing and optimized array operations. Here is a detailed description of how MATLAB achieves this in this work.

3.1. GPU computing

MATLAB offers GPU arrays, which are designed to store and process data on a GPU. By simply converting standard MATLAB arrays to GPU arrays using the *gpuArray* function, we can take advantage of GPU acceleration. A sample which is widely used in our code is as follows:

```
A = rand(1000, 1000);
%define a 1000x1000 matrix
gpuA = gpuArray(A);
%copy the matrix A from CPU to GPU memory
B = fft(gpuA);
% Perform Fast Fourier Transform on GPU
```

When 3D-GTDSE performs operations on the GPU arrays, MATLAB automatically executes these operations on the GPU, taking advantage of GPU's parallel processing capabilities.

3.2. Array operations

MATLAB is optimized for array operations, which can lead to significant performance improvements. By vectorizing code, MATLAB can take advantage of its optimized numerical libraries and multi-threaded computation capabilities, such as vectorization, which involves rewriting loops and iterative operations as array operations. MATLAB is optimized to handle operations on entire arrays at once, which reduces the overhead associated with loop control and takes advantage of low-level optimizations. A sample is given to present how vectorization works in 3D-GTDSE.

```
for i = 1:N
    B(i) = A(i) * 2;
end

% Vectorized approach
B = A * 2; % This is much faster in MATLAB
```

Besides, MATLAB's core strength lies in its ability to perform matrix operation functions efficiently. Functions like *sum* (which is widely used in our code) for calculating the sum of array elements are highly optimized. A comparison of the performance of using the *sum* function versus using a *for loop* for this task is given, using the *sum* function:

```
% Generate a large array of random numbers
N = 1e6; % Number of elements
A = rand(1, N);

% Measure time taken by sum function
tic;
totalSum = sum(A);
timeSum = toc;

fprintf('Using sum function:\n');
fprintf('Total sum: %.6f\n', totalSum);
fprintf('Time taken: %.6f seconds\n', timeSum);
```

then using *for loop*:

```
timeSum = toc;

fprintf('Using sum function:\n');
fprintf('Total sum: %.6f\n', totalSum);
fprintf('Time taken: %.6f seconds\n', timeSum);
```

This example runs on our personal computer and outputs look like this:

```
% Initialize the sum variable
totalSumLoop = 0;

% Measure time taken by for loop
tic;
for i = 1:N
    totalSumLoop = totalSumLoop + A(i);
end
timeLoop = toc;

fprintf('\nUsing for loop:\n');
fprintf('Total sum: %.6f\n', totalSumLoop);
fprintf('Time taken: %.6f seconds\n', timeLoop);
```

We can find that using the *sum* function would get about two orders of magnitude faster acceleration. Such improvement will be more evident if a higher-dimensional matrix $B(i, j, ...)$ is employed.

3.3. Combining GPU computing and array operations

Combining the above techniques can yield even greater performance improvements. For instance, performing vectorized operations on GPU arrays can harness both the power of MATLAB's optimized array handling and the parallel processing capabilities of GPUs:

```
A = rand(1000, 1000);
gpuA = gpuArray(A);
% Vectorized GPU operations

gpuB = gpuA * 2;
 % Element-wise multiplication on GPU
```

4. Description of the program

The whole 3D-GTDSE program is written in MATLAB and users do not need to install it.

The flow diagram of 3D-GTDSE can be found in Fig. 1. The *main.m* file serves to call and run our program. Running 3D-GTDSE needs the following two steps:

Step 1: Set the path to the source code that needs to be written to the header of the *main.m* program code via the *addpath('the source code path')* function.

Step 2: Select one specific subprogram and write down its file name after *addpath('the source code path')*. When the selected subprogram runs, the rest is commented out.

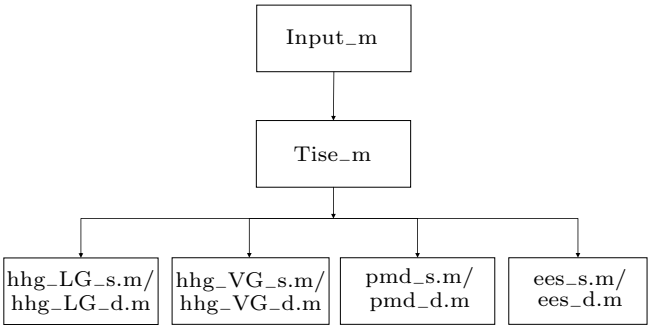


Fig. 1. Flow chart of the programmes. See text for details.

The subprograms are divided into six main categories:

(1) *tise.m* is for calculating the ground state wave function.

(2) *hhg_VG_s.m*, *hhg_VG_d.m* are for calculating HHG spectra in the velocity gauge with single and double precision, respectively.

(3) *hhg_LG_s.m*, *hhg_LG_d.m* are for calculating HHG spectra in the length gauge with single and double precision, respectively.

(4) *pmd_s.m*, *pmd_d.m* are for calculating the PMD with single and double precision, respectively.

(5) *ees_s.m*, *ees_d.m* are for calculating the PES with single and double precision, respectively.

(6) *Input.m* is used to set up the simulation box, time and space steps as well as the laser parameters. Details are given as follows: x_0 and x_m represent the minimal position and maximal position in the x-direction, respectively. y_0 and y_m represent the minimal position and maximal position in the y-direction, respectively. z_0 and z_m represent the minimal position and maximal position in the z-direction, respectively. N_x, N_y, and N_z represent the numbers of grid points in the x, y, and z directions, respectively. To avoid the Coulomb singularity, ensure that the grid points in each dimension are set to even numbers, or at least maintain an even number of grid points in one dimension. dt is the time step in the evolutionary process. t_0 is the starting time of the evolutionary process. t_f is the end

time of the evolutionary process. The above parameters are all in atomic units. I is the laser field intensity, which is in the units of W/cm^2. λ is the wavelength of the laser, which is in the units of nm. $N_{\rm gpu} = 1, 2, \ldots$ are the GPU indices that users want to call to calculate, if the server contains more than one GPU.

5. Example

In this section, we will give examples of 3D-GTDSE to show the feasibility of the code. All calculations run on NVIDIA GeForce RTX 3090 GPU. All the test examples are prepared in "*../3D-GTDSE/test/*" subdirectory.

5.1. Ground state calculation

In this example, we obtain the ground state of the hydrogen atom using the imaginary time evolution method, as introduced in Subsection 2.2.

We perform the calculation in a $20 \times 20 \times 20$ simulation box with a grid size of 512^3. The time step is set to 0.001. Table 1 shows some of the details of the evolutionary process. Figure 2 illustrates the real-time variation of the ground state energy and neighboring energy differences throughout the imaginary time evolution. We can see that the energy converges at about 4000 steps and achieves a precision of 10^{-10} after 19897 iteration steps of imaginary time. The machine time spent is about 1.60 h.

Our method is based on a Cartesian-coordinate system, which may encounter issues with Coulomb singularities during grid discretization. To address this, our code allows for either the use of a soft-core Coulomb potential or skipping the singularities by appropriately choosing the grid points.

Table 1. Calculation steps vs. the instantaneous eigenvalues and errors.

Step	Instantaneous eigenvalues	Instantaneous ΔE
1	0.2056678287	$4.0646703870\times10^{-3}$
2	0.2018800268	$3.7878019028\times10^{-3}$
3	0.1982747862	$3.6052405405\times10^{-3}$
4	0.1948121714	$3.4626148681\times10^{-3}$
5	0.1914682688	$3.3439026095\times10^{-3}$
6	0.1882266846	$3.2415841550\times10^{-3}$
7	0.1850752889	$3.1513957480\times10^{-3}$
8	0.1820046494	$3.0706395092\times10^{-3}$
9	0.1790071705	$2.9974788564\times10^{-3}$
10	0.1760765745	$2.9305959941\times10^{-3}$
⋮	⋮	⋮
19888	–0.4976741950	$1.0067113809\times10^{-11}$
19889	–0.4976741950	$1.0060507982\times10^{-11}$
19890	–0.4976741950	$1.0052847443\times10^{-11}$
19891	–0.4976741950	$1.0045630994\times10^{-11}$
19892	–0.4976741950	$1.0037970455\times10^{-11}$
19893	–0.4976741950	$1.0030198894\times10^{-11}$
19894	–0.4976741950	$1.0023204489\times10^{-11}$
19895	–0.4976741950	$1.0015988039\times10^{-11}$
19896	–0.4976741951	$1.0008438522\times10^{-11}$
19897	–0.4976741951	$1.0001666162\times10^{-11}$

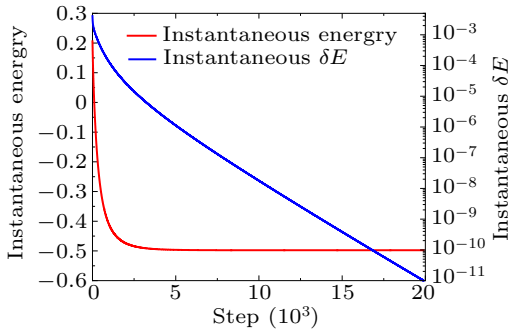


Fig. 2. Real-time variation of the ground state energy and neighboring energy difference of the hydrogen atom throughout the imaginary time evolution.

To investigate the impact of the second approach on computational accuracy, we compared two sets of ground-state energy calculations for the hydrogen atom, illustrating the effect of skipping singularities. The two cases correspond to a large box (200×200×200) and a small box (20×20×20), with grid sizes of 511×511×512, 511×512×512, and 512^3, respectively.

For the large box case: The 512^3 grid skips all singularities; the converged ground-state energy is −0.4712. The 511×512×512 grid skips two singularities; the result is −0.4777. The 511×511×512 grid skips only one singularity; the result is −0.4937.

For the small box case: The 512^3 grid yields −0.497674. The 511×512×512 grid yields −0.497774. The 511×511×512 grid yields −0.497999.

These results indicate that skipping singularities can affect the accuracy of the hydrogen atom ground-state energy calculation. However, if the grid only skips the singularity in one direction, the accuracy remains reasonably acceptable.

Meanwhile, according to Ref. [41], the PCTDSE program running on a 16-core CPU server in parallel takes about 24 h to reach similar results under the same box size, grid size and the iteration steps.

5.2. HHG test

In this section, we present HHG calculations for the interaction of a linearly polarized laser with a hydrogen atom. The laser vector potential is chosen to have a sine-squared envelope

$$A(t) = \begin{cases} \frac{E_0}{\omega}\sin^2\left(\frac{\pi t}{T_1}\right)\sin(\omega t)e_z, & 0 \le t \le T_1, \\ 0, & \text{otherwise}, \end{cases} \tag{31}$$

where $E_0 = 0.0292$ corresponds to an intensity of 0.3×10^{14} W/cm^2, $\omega = 0.0569$ is the angular frequency, and $T_1 = 3T$ with $T = 2\pi/\omega$ being one optical period.

Due to the limitations of GPU memory size, the calculation is performed using a grid of size $256 \times 256 \times 1536$, covering ±20 a.u. in the x-direction, ±20 a.u. in the y-direction, and ±125 a.u. in the z-direction. The time step for this computation is set to 0.02, and there are 17501 steps in total. The

single-precision computation takes 0.59 h, while the double-precision computation takes 1.54 h.

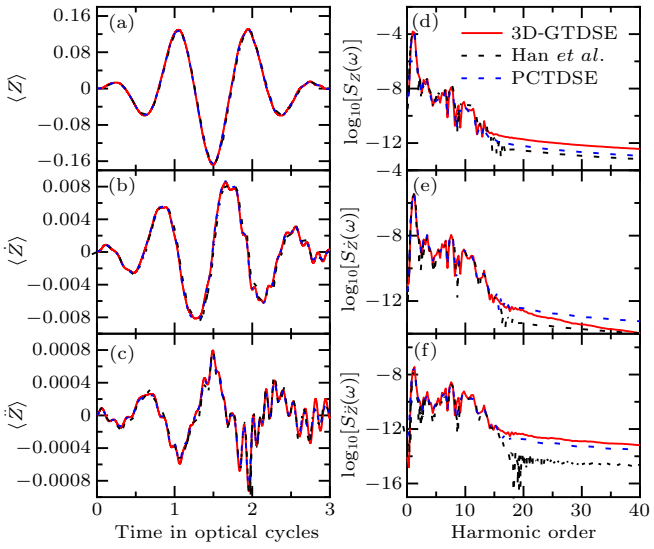


Fig. 3. Expectation values of dipole moment (a), dipole velocity (b), and dipole acceleration (c) as a function of time, along with the respective harmonic spectra (d), (e), and (f) versus harmonic order, for a laser of wavelength $\lambda = 800$ nm and an intensity of 0.3×10^{14} W/cm^2. The red solid lines are the results of our calculations, the blue dashed lines are the results from PCTDSE code,[41] and the black solid lines are the results from Ref. [46].

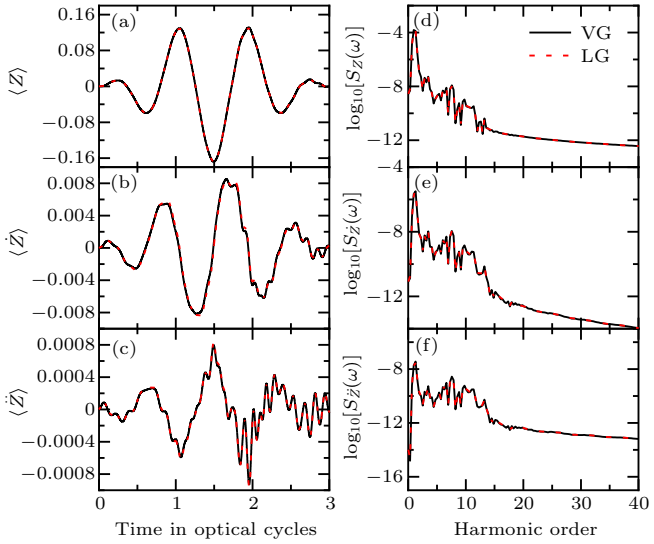


Fig. 4. Expectation values of dipole moment (a), dipole velocity (b), and dipole acceleration (c) as a function of time, along with the respective harmonic spectra (d), (e), and (f) versus harmonic order, for a laser of wavelength $\lambda = 800$ nm and an intensity of 0.3×10^{14} W/cm^2. The dashed red and solid black lines refer to the results in the length (LG) and velocity (VG) gauges, respectively.

We perform the HHG calculations following the method in Subsection 2.4. The results are shown in Fig. 3. Figures 3(a), 3(b), and 3(c) display the induced dipole $\langle \boldsymbol{r}(t) \rangle$, $\langle \dot{\boldsymbol{r}}(t) \rangle$, and $\langle \ddot{\boldsymbol{r}}(t) \rangle$, respectively. Figures 3(d), 3(e), and 3(f) show the corresponding HHG spectra. The red solid lines represent results from 3D-GTDSE, while the green and blue dashed lines are from Han *et al.*[46] and the PCTDSE code, respectively. These figures demonstrate that 3D-GTDSE achieves the same HHG results as the other two codes, confirming the validity of our code.

Furthermore, we compare the high harmonic generation in the velocity gauge with that in the length gauge, as shown in Fig. 4. We find that the length gauge (LG) and the velocity gauge (VG) provide the same results for $\langle \boldsymbol{r}(t) \rangle$, $\langle \dot{\boldsymbol{r}}(t) \rangle$, and $\langle \ddot{\boldsymbol{r}}(t) \rangle$, leading to spectra in the three forms that exhibit very similar shapes. This is because the laser intensity is much lower than the threshold for over-the-barrier ionization ($I_b = 1.4 \times 10^{14}$ W/cm^2), as investigated in Ref. [46].

5.3. Photoelectron momentum distribution

In this example, we perform simulations of the PMD as well as the offset angles for a hydrogen atom interacting with circularly polarized pulses. The simulation parameters are the same as those used in Refs. [41,47]. The vector potential has the form

$$A(t) = \begin{cases} -A_0 \cos^4(\frac{\omega t}{4}) \\ \times(\cos(\omega t)e_x + \sin(\omega t)e_y), & -T \le t \le T, \\ 0, & \text{otherwise} \end{cases} \quad (32)$$

with the laser's duration nearly single-cycle and central wavelength $\lambda = 800$ nm. The peak intensity is 0.8×10^{14} W/cm^2.

The results from 3D-GTDSE and PCTDSE can be found in Figs. 5(a) and 5(b), respectively. The two PMDs match well with each other.

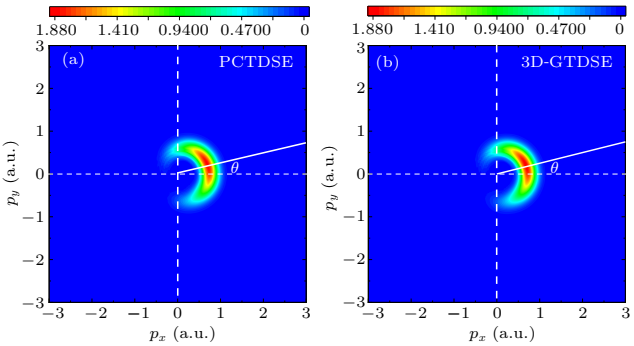


Fig. 5. Photoelectron momentum distribution in the polarization (x–y) plane at the laser intensity of 0.8×10^{14} W/cm^2. θ is the offset angle relative to the horizontal p_x direction. A comparisons of the momentum distribution diagram. (b) is from 3D-GTDSE code and (a) is from PCTDSE.

The so-called offset angle can be calculated from the average electron momentum distribution in the polarization plane[41,47]

$$\theta = \arctan\left(\frac{\int f(p, t_f) p_y \mathrm{d}p}{\int f(p, t_f) p_x \mathrm{d}p}\right). \quad (33)$$

By modifying the peak laser intensities and performing the same calculations, we obtain the offset angle θ as a function of the laser intensity in Fig. 6. Comparing the results with PCTDSE[41] and L. Torlina *et al.*,[47] we see that the three curves are essentially the same overall.

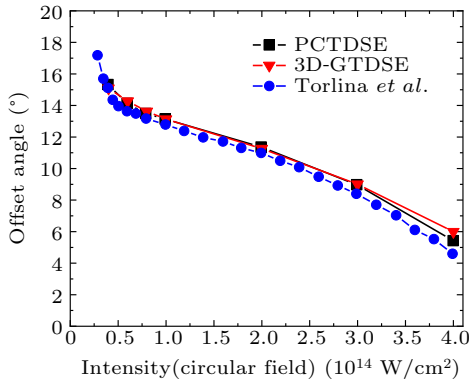


Fig. 6. Comparisons of the offset angles as a function of laser intensity, obtained from 3D-GTDSE (red triangle), PCTDSE (black square) and Torlina *et al.*[47] (blue circle).

The 3D-GTDSE calculations are made using an asymmetric grid of size $511 \times 511 \times 512$, covering a box size of ± 100 in each dimension. The time step is 0.02 and there are 18,001 time steps in total. The single precision computation takes 1.20 h for each offset angle calculation, while the double precision computation takes 3.76 h.

5.4. Photoelectron spectrum

In this example, we calculate the photoelectron spectrum after the interaction between strong infrared laser pulses and hydrogen atoms. The vector potential of a 10-cycle laser pulse with $\sin^2$ envelope is employed in this subsection:

$$A(t) = \begin{cases} \frac{E_0}{\omega}\sin^2(\frac{\omega t}{20})\sin(\omega t)e_z, & 0 \le t \le 20\pi/\omega, \\ 0, & \text{otherwise.} \end{cases} \tag{34}$$

Three cases with different peak intensities 4×10^{14} W/cm^2 ($E_0 = 0.1066$), 6×10^{14} W/cm^2 ($E_0 = 0.1308$) and 1×10^{15} W/cm^2 ($E_0 = 0.1688$) are tested. The wavelength is 800 nm ($\omega = 0.057$ a.u.).

Figures 7(a)–7(c) show our results for the three peak intensities, each of which gives a comparison with the result from Ref. [48] and PCTDSE.[41] The grid size is $675 \times 675 \times 676$ and the box covers ± 160 in each dimension. The time step for this calculation is set to 0.02 and there are 75001 steps in total, the single precision computation takes 7.10 h.

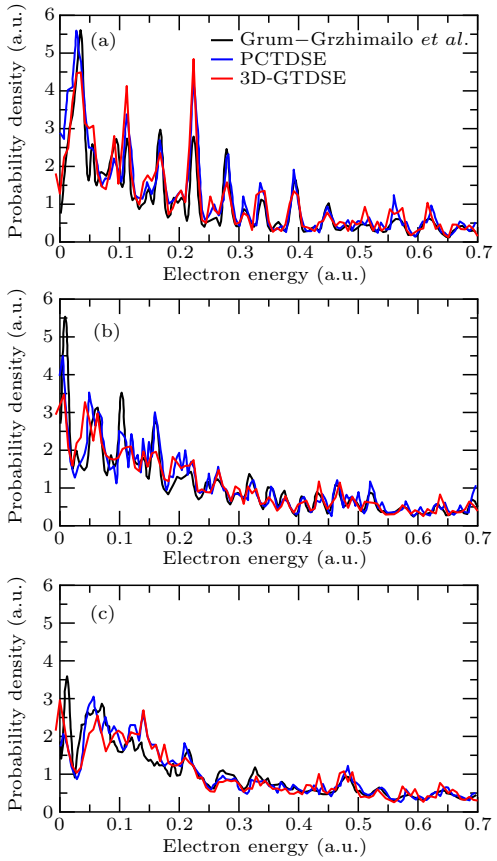


Fig. 7. Photoelectron energy spectrum of atomic hydrogen in a laser pulse as Eq. (34) at the three peak intensities of 4×10^{14} W/cm^2 ($E_0 = 0.1066$) (a), 6×10^{14} W/cm^2 ($E_0 = 0.1308$) (b) and and 1×10^{15} W/cm^2 ($E_0 = 0.1688$) (c), at the wavelength of 800 nm. The spectra of A. N. Grum-Grzhimailo *et al.*[48] (black) and PCTDSE (blue) are in good agreement with ours (red).

5.5. Extended functions

Here in this subsection, we provide some extended function examples in 3D-GTDSE, such as solving ND TDSE, nuclear-electron coupled TDSE, and 4-dimensional (4D) TISE.

5.5.1. ND-TDSE

The non-dipole approximation is essential for capturing the full range of physical phenomena that occur when a strong laser interacts with a target, particularly under conditions of high intensity, short wavelength, and complex target structure. These simulations with the non-dipole approximation allow more accurate and comprehensive predictions of experimental outcomes and a deeper understanding of the underlying physics.

We present an example of solving the ND TDSE in Eq. (30) to calculate HHG in Fig. 8. The space-dependent electric field component of the laser takes the form

$$E(z,t) = \begin{cases} E_0\sin^2\left(\frac{\omega t}{6}\right)\cos(k_z z - \omega t), & 0 \le t \le 6\pi/\omega, \\ 0, & \text{otherwise,} \end{cases} \tag{35}$$

where $E_0 = 0.0292$ corresponds to an intensity of 0.3×10^{14} W/cm^2, and $\omega = 0.0207$ is the angular frequency. The path integral of the space-position-dependent electric field is performed as follows (we choose the path from $(0, 0, z_m)$ to (x, y, z)):

$$\begin{aligned} &\int_C E(\boldsymbol{r},t)\cdot \mathrm{d}l \\ &= \int_0^x (E_x\boldsymbol{e}_x + E_y\boldsymbol{e}_y + E_z\boldsymbol{e}_z)\,\mathrm{d}x\boldsymbol{e}_x \\ &\quad + \int_0^y (E_x\boldsymbol{e}_x + E_y\boldsymbol{e}_y + E_z\boldsymbol{e}_z)\,\mathrm{d}y\boldsymbol{e}_y \\ &\quad + \int_{z_m}^z (E_x\boldsymbol{e}_x + E_y\boldsymbol{e}_y + E_z\boldsymbol{e}_z)\,\mathrm{d}z\boldsymbol{e}_z \\ &= \frac{E_0}{k_z}\sin^2\left(\frac{\omega t}{6}\right)\sin(k_z z - \omega t) \end{aligned}$$

$$+\frac{E_0}{k_z}\sin^2\left(\frac{\omega t}{6}\right)\sin(k_z z m-\omega t). \tag{36}$$

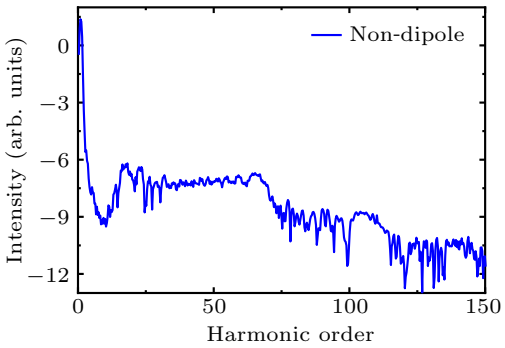


Fig. 8. HHG from interaction between a space-dependent laser field and H atom. The laser intensity is 3×10^{14} W/cm^2 and the wavelength is 2200 nm.

5.5.2. Electron–nuclear coupled TDSE

Accurate simulation of electron–nuclear coupled dynamics under a strong laser field allows for a more comprehensive understanding of processes such as multiphoton ionization and excitation, ultrafast bond-breaking dynamics, re-collision processes, and HHG in non-adiabatic conditions where the separation of electron and nuclear timescales is invalid.

In this example, we simulated the HHG process from the interaction between a hydrogen molecular ion (H_2^+) and an ultrashort, ultra-intense infrared laser field within the center-of-mass coordinate system. The modulation from 3D electron code to 2D (electron) + 1D (nuclei) is straightforward and can be found in the "*../3D-GTDSE/Extended code/TDSE_H2+/*" folder. The Hamiltonian is given by

$$H = H_0 + H_1, \tag{37}$$

$$\begin{aligned} H_0 = &-\frac{1}{2\mu_N}\frac{\partial^2}{\partial R^2}+-\frac{1}{2\mu_e}\frac{\partial^2}{\partial r^2}+\frac{q_1q_2}{R}\\ &-\frac{q_1}{\sqrt{\left(x+\frac{M_2}{M_1+M_2}R\right)^2+y^2+0.25}}\\ &-\frac{q_2}{\sqrt{\left(x-\frac{M_1}{M_1+M_2}R\right)^2+y^2+0.25}}, \end{aligned} \tag{38}$$

$$H_1 = -\frac{M_1-M_2}{M_1+M_2}E(t)\cdot R+\left(\frac{M_1+M_2+2m}{M_1+M_2+m}\right)E(t)\cdot r, \tag{39}$$

$$\begin{aligned} a(t) = &\langle\psi(r,R,t)|-\frac{\partial V_C}{\partial r}\\ &-\left(\frac{M_1+M_2+2m}{M_1+M_2+m}\right)E(t)\,|\psi(r,R,t)\rangle\,. \end{aligned} \tag{40}$$

Here, q_1 and q_2 represent the electric charges of the two atomic nuclei, while M_1 and M_2 denote their respective masses. The reduced mass of the nuclei is $\mu_N = M_1M_2/(M_1+M_2)$, and the reduced mass of the electrons is also $\mu_e = m(M_1+M_2)/(M_1+M_2+m)$. The HHG spectrum is given by

$$S(\omega)=\frac{1}{2\pi}\left|\int_{-\infty}^{\infty}a(t)\,e^{i\omega t}\,dt\right|^2. \tag{41}$$

We use a trapezoidal envelope 10-cycle laser field in the form

$$E(t)=E_0 f(t)\cos(\omega t), \tag{42}$$

where

$$f(t)=\begin{cases}\frac{t}{2T}, & 0<t<2T,\\ 1, & 2T<t<8T,\\ \frac{10T-t}{2T}, & 8T<t<10T,\\ 0, & \text{otherwise,}\end{cases} \tag{43}$$

$E_0 = 0.098$ corresponds to an intensity of 3.4×10^{14} W/cm^2, and $\omega = 0.057$.

Figure 9 shows the results for the electric field perpendicular and parallel to the nuclear vibration direction. We observe that the HHG signals in the parallel case are stronger than those in the perpendicular case, which is consistent with the electron density distribution of H_2^+.

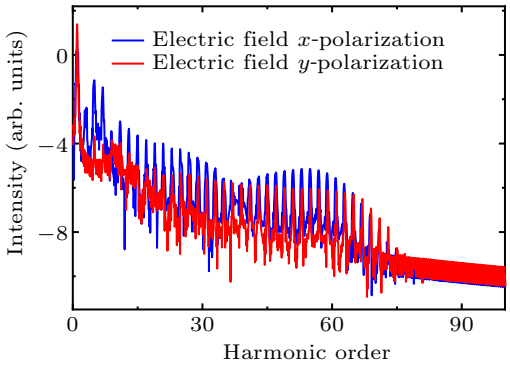


Fig. 9. HHG spectra from interaction between a trapezoidal envelope strong laser field and H_2^+. The simulations are performed in 2D (electron) + 1D (nuclei) space. The molecular axis is along the x-direction. The red line represents HHG from the electric field along the x-direction, while the blue line represents HHG from the electric field along the y-direction.

The 3D-GTDSE can also be modified to 1D (electron) + 2D (nuclei) or pure 3D nuclei code with little effort. These modifications can be left as an exercise for the user to complete on their own.

5.5.3. The 4D TISE

As illustrated in Refs. [34,41], one advantage of the Cartesian-coordinates based codes is that it easily allows the codes to be extended to higher degrees of freedom. In this example, we extended our 3D electron code to a 3D (electron) + 1D (nuclei) case. The Hamiltonian in the absence of a laser field is given by

$$\begin{aligned} H_0 = &-\frac{1}{2\mu_N}\frac{\partial^2}{\partial R^2}-\frac{1}{2\mu_e}\frac{\partial^2}{\partial r^2}+\frac{q_1q_2}{R}\\ &-\frac{q_1}{\left|r+\frac{M_2}{M_1+M_2}R\right|}-\frac{q_2}{\left|r-\frac{M_1}{M_1+M_2}R\right|}. \end{aligned} \tag{44}$$

Due to the GPU memory limitations (24G in total) we are using, we can only provide an example of solving the TISE for H_2^+. As GPU memory capacity increases, we believe that the memory limitations will become less significant.

This simulation is performed on a grid of size $256\times256\times256\times64$, covering $\pm10\times\pm10\times\pm10\times(0\sim5]$ in x, y, z and R axes, respectively. We obtain the energy of the ground state of H_2^+ as $E_{\min}=-0.5819$, and the average spacing of the nuclear vibrations as $R=2.1241$. Our results consist well with those from Refs. [49,50].

5.5.4. Parallel computing with multiple GPUs

Our code can achieve multi-GPU parallelization for computations that do not require large-scale data exchange, such as photoionization streaking camera simulations. Here, we present an example of multi-GPU parallelization: the calculation of the potential energy surface of the H_2^+ molecular ion. The field-free Hamiltonian is given by

$$H_0 = -\frac{1}{2\mu_N}\frac{\partial^2}{\partial R^2}+-\frac{1}{2\mu_e}\frac{\partial^2}{\partial r^2}+\frac{q_1q_2}{R} - \frac{q_1}{\sqrt{\left(x+\frac{M_2}{M_1+M_2}R\right)^2+y^2+0.6}} - \frac{q_2}{\sqrt{\left(x-\frac{M_1}{M_1+M_2}R\right)^2+y^2+0.6}}. \tag{45}$$

We compute various electronic states at different internuclear distances under the Born–Oppenheimer approximation. (Due to the use of Cartesian coordinates in our calculations, the orthogonality of electronic wave functions may cause state swapping during the computation of electronic states, requiring manual correction.) The example presents the calculation of the potential energy curve for the hydrogen molecular ion as shown in Fig. 10.

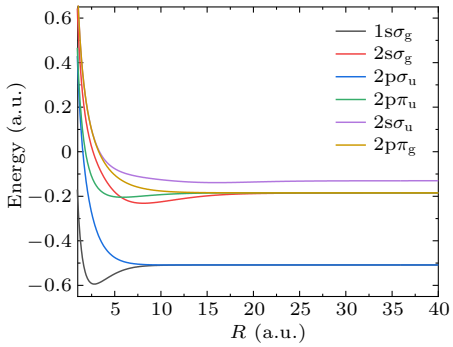


Fig. 10. The potential curves of several quantum states within the Born–Oppenheimer approximation.

6. Parallel scaling

In this section, we provide a comparison of HHG results between double precision and single precision calculations. An estimate of memory usage for different grid sizes is also provided. Finally, we compare the efficiency of the 3D-GTDSE and PCTDSE programs.

6.1. HHG from double precision and single precision

Here, we discuss single and double precision calculations. We find that the precision choice would strongly influence the GPU parallel scaling (about 5–6 times), while for CPU calculations, it only has an influence of about 30%–40% (since CPU parallel scaling is not concerned with this work, we do not show the CPU results, but such CPU calculations can be performed by the readers easily). Through many experimental simulations, we find that the results of single-precision calculation and double-precision calculation are basically the same as shown in Fig. 11, which gives the comparison of high harmonic generation between single-precision and double-precision calculations.

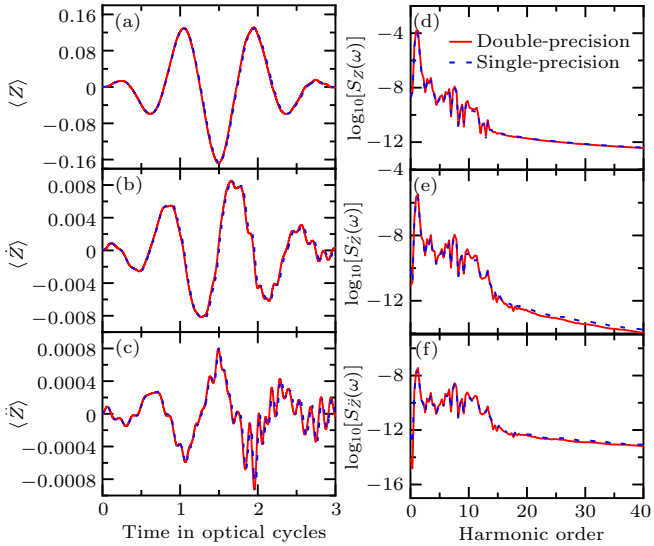


Fig. 11. Expectation values of dipole moment (a), dipole velocity (b), and dipole acceleration (c) as a function of time, along with the respective harmonic spectra (d), (e), and (f) versus harmonic order, for a laser of wavelength $\lambda=800$ nm and an intensity of 0.3×10^{14} W/cm^2. The dashed blue and solid red lines refer to the results of single-precision and double-precision calculations, respectively.

Besides, due to the limitation of the GPU memory size, calculations running with single-precision in large grid size spaces are an alternative choice, which will be shown in the next subsection.

6.2. Memory usage of GPU

In this subsection, we give the GPU memory usages of imaginary time evolution and time-dependent evolution in 3D space.

Table 2. GPU memory usage of imaginary time evolution. Total GPU memory of one GeForce RTX 3090 is 24268M.

Size-of-grid	Memory usage (double-precision)	Memory usage (single-precision)
64^3	532	514
128^3	784	640
256^3	2802	1650
512^3	16892	9718
600^3	23586	14516
650^3	–	17268
700^3	–	21448

From Tables 2 and 3, it can be seen that for a 3090 GPU with 24 GB of memory, the maximum box size for single-

precision calculations is 700^3. For double-precision calculations, the maximum box size for imaginary time evolution is 600^3, and for real-time evolution, the maximum is 512^3. If users require a larger box size, they can use a GPU with more memory for their calculations.

Table 3. GPU memory usage of time-dependent evolution. Total GPU memory of one GeForce RTX 3090 is 24268M.

Size-of-grid	Memory usage (double-precision)	Memory usage (single-precision)
64^3	558	526
128^3	824	658
256^3	2956	1724
512^3	19984	10240
600^3	–	16174
650^3	–	20428
700^3	–	22774

From this subsection and the comparison between single precision and double precision calculations, it is evident that using single precision, when accuracy requirements are met, significantly reduces the GPU memory usage.

6.3. Efficiency

Our 3D-GTDSE runs on a single NVIDIA RTX3090 GPU under both double and single precisions. PCTDSE calculations were carried out on the Tianhe-2 supercomputer system at the National University of Defense Technology (NUDT) with Intel Xeon 24, 48, 72, 112, 144 CPU cores respectively in Ref. [41] and the calculation times of PCTDSE are copied from Ref. [41] directly.

The comparison is given in Fig. 12. We can find that for double precision calculation, 3D-GTDSE is about two times faster than PCTDSE running on 144 CPU cores, while for single precision calculation, 3D-GTDSE is about one order faster. The results indicate that 3D-GTDSE has a good parallel scaling and efficiency.

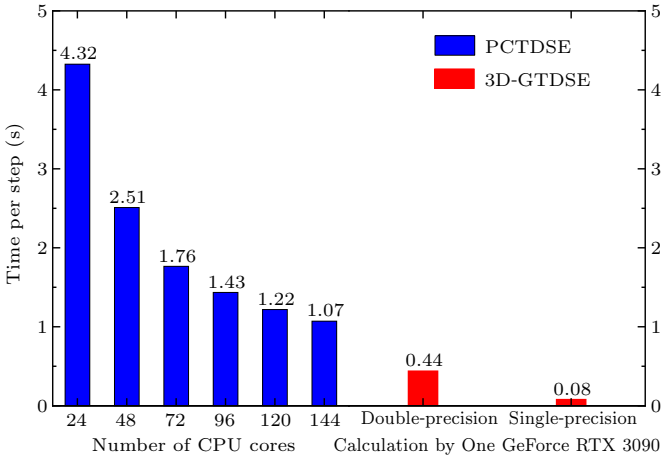


Fig. 12. The times required for PCTDSE to evolve at each time step with MPI acceleration from various numbers of CPU cores are presented as blue bars, these data are copied from Ref. [41] directly. The times required for 3D-GTDSE to evolve at each time step for double-precision and single-precision computations are given in red bars. The box sizes and grid sizes are totally the same in this figure.

7. Conclusions

In this work, we present a 3D TDSE solver in Cartesian coordinates accelerated with GPU and vectorization. The program is described in detail, and test examples of HHG, PES, PMD, as well as attoclock offset angles, are provided. Our showcases indicate that 3D-GTDSE is an efficient, easily expandable, and user-friendly tool for studying ultrafast nonlinear dynamics under strong laser fields.

We find that GPU memory limits the program's usage. Looking ahead, in the next version of the program, we will focus on developing multi-GPU parallel algorithms to distribute the total computational memory across multiple GPUs. Additionally, we will investigate ways to accelerate data transfer speeds between different GPUs.

Program availability

The source code is openly available in the Science Data Bank at https://doi.org/10.57760/sciencedb.27107.

Acknowledgements

Ke Peng thanks Jia Hao Liu and Bin Liu for helpful discussions on mathematical derivation and the program developing. This work was supported by the GHfund A (Grant No. ghfund202407013663), the Fundamental Research Funds for the Central Universities (Grant No. GK202207012), Shaanxi Province (Grant No. QCYRCXM-2022-241), the National Key Research and Development Program of China (Grant No. 2022YFE0134200), Guangdong Basic and Applied Basic Research Foundation (Grant No. 2025A1515011117), the Natural Science Foundation of Jilin Province (Grant No. 20220101016JC), and the National Natural Science Foundation of China (Grant Nos. 12374238, 11934004, and 11974230).

References

[1] Krausz F and Ivanov M 2009 *Rev. Mod. Phys.* **81** 163
[2] Drescher M, Hentschel M, Kienberger R, Uiberacker M, Yakovlev V, Scrinzi A, Westerwalbesloh T, Kleineberg U, Heinzmann U and Krausz F 2002 *Nature* **419** 803
[3] Goulielmakis E, Yakovlev V S, Cavalieri A L, Uiberacker M, Pervak V, Apolonski A, Kienberger R, Kleineberg U and Krausz F 2007 *Science* **317** 769
[4] Uiberacker M, Uphues T, Schultze M, Verhoef A J, Yakovlev V, Kling M F, Rauschenberger J, Kabachnik N M, Schroeder H and Lezius M 2007 *Nature* **446** 627
[5] Cui H F and Miao X Y 2020 *Chin. Phys. B* **29** 074203
[6] Grundmann S, Kircher M, Vela-Perez I, Nalin G and Doerner R 2020 *Phys. Rev. Lett.* **124** 233201
[7] Ivanov I 2014 *Phys. Rev. A* **90** 013418
[8] Higuet J, Ruf H, Thiré N, Cireasa R, Constant E, Cormier E, Descamps D, Mével E, Petit S, Pons B, *et al.* 2011 *Phys. Rev. A* **83** 053401
[9] Agostini P and DiMauro L F 2004 *Rep. Prog. Phys.* **67** 813
[10] Sansone G, Benedetti E, Calegari F, Vozzi C, Avaldi L, Flammini R, Poletto L, Villoresi P, Altucci C, Velotta R, *et al.* 2006 *Science* **314** 443
[11] Carrera J J, Tong X M and Chu S I 2006 *Phys. Rev. A* **74** 023404

[12] Aubert B, Barate R, Boutigny D, Couderc F, Gaillard J M, Hicheur A, Karyotakis Y, Lees J, Tisserand V, Zghiche A, *et al*. 2004 *Phys. Rev. D* **69** 111104

[13] Zhang L and Simos T E 2016 *Adv. Math. Phys.* **2016** 8181927

[14] Liu N and Jiang W 2018 *Adv. Comput. Math.* **44** 1235

[15] Awasthi M, Vanne Y V, Saenz A, Castro A and Decleva P 2008 *Phys. Rev. A* **77** 063403

[16] Saenz A, Awasthi M, Vanne Y, Castro A and Decleva P 2008 *APS Division of Atomic, Molecular and Optical Physics Meeting Abstracts* Vol. 39 pp. OPO-50

[17] Schild A and Gross E 2017 *Phys. Rev. Lett.* **118** 163202

[18] Nisoli M, Decleva P, Calegari F, Palacios A and Martín F 2017 *Chem. Rev.* **117** 10760

[19] Born M 1927 *Ann. Phys* **84** 457

[20] Walker D W and Dongarra J J 1996 *Supercomputer* **12** 56

[21] Chandra R, Dagum L, Kohr D, Maydan D, McDonald J and Menon R 2001 *Parallel Programming in OpenMP* (Morgan Kaufmann), ISBN: 1558606718

[22] Mosert V and Bauer D 2016 *Comput. Phys. Commun.* **207** 452

[23] Zhang Z H, Li Y, Mao Y J and He F 2023 *Comput. Phys. Commun.* **290** 108787

[24] Kozicki J 2023 *Comput. Phys. Commun* **291** 108839

[25] Scrinzi A 2022 *Comput. Phys. Commun.* **270** 108146

[26] Brown A C, Armstrong G S, Benda J, Clarke D D, Wragg J, Hamilton K R, Mašín Z, Gorfinkiel J D and van der Hart H W 2020 *Comput. Phys. Commun.* **250** 107062

[27] Patchkovskii S and Muller H 2016 *Comput. Phys. Commun.* **199** 153

[28] Forembski A and Nikolopoulos L A 2023 *Comput. Phys. Commun.* **291** 108820

[29] Farber R 2011 *CUDA application design and development* (Morgan Kaufmann)

[30] Munshi A 2009 2009 *IEEE Hot Chips* 21 *Symposium (HCS)* (IEEE) pp. 1–314

[31] Wienke S, Springer P, Terboven C and Mey D 2012 *Euro-Par 2012 Parallel Processing: 18th International Conference*, Rhodes Island, Greece, pp. 859–870

[32] Zhao X, Zhang G, Bai T, Wang J and Yu W W 2021 *Chin. Phys. B* **30** 073201

[33] Broin C O and Nikolopoulos L 2014 *Comput. Phys. Commun.* **185** 1791

[34] Dziubak T and Matulewski J 2012 *Comput. Phys. Commun.* **183** 800

[35] Chen H, McMahon J M, Ratner M A and Schatz G C 2010 *J. Phys. Chem. C* **114** 14384

[36] Marques M A L, Castro A, Bertsch G F and Rubio A 2003 *Comput. Phys. Commun.* **151** 60

[37] Castro A, Appel H, Oliveira M, Rozzi C A, Andrade X, Lorenzen F, Marques M A L, Gross E K U and Rubio A 2006 *Phys. Status. Solidi. (b)* **243** 2465

[38] Dinh P M, Vincendon M, Coppens F, Suraud E and Reinhard P G 2022 *Comput. Phys. Commun.* **270** 108155

[39] Milosevic N, Corkum P B and Brabec T 2004 *Phys. Rev. Lett.* **92** 013002

[40] He P L, Lao D and He F 2017 *Phys. Rev. Lett.* **118** 163203

[41] Fu Y, Zeng J and Yuan J 2017 *Comput. Phys. Commun.* **210** 181

[42] Bandrauk A D and Shen H 1991 *Chem. Phys. Lett.* **176** 428

[43] Gerlind P, Daniel P, Gabriele S and Manfred T 2018 *Numerical Fourier Analysis* (Springer International Publishing) pp. 231–303

[44] Feit M, Fleck Jr J and Steiger A 1982 *J. Comput. Phys.* **47** 412

[45] Lehtovaara L, Toivanen J and Eloranta J 2007 *J. Comput. Phys.* **221** 148

[46] Han Y C and Madsen L B 2010 *Phys. Rev. A* **81** 063430

[47] Torlina L, Morales F, Kaushal J, Ivanov I, Kheifets A, Zielinski A, Scrinzi A, Muller H G, Sukiasyan S, Ivanov M and Smirnova O 2015 *Nat. Phys.* **11** 503

[48] Grum-Grzhimailo A N, Abeln B, Bartschat K, Weflen D and Urness T 2010 *Phys. Rev. A* **81** 043408

[49] Flügge S and Franzen W 2012 *Practical quantum mechanics* (Springer Science & Business Media)

[50] Bressanini D, Mella M and Morosi G 1997 *Chem. Phys. Lett.* **272** 370