Quantum Dissipative Dynamics

User manual

F.M.G.J. Coppens
P.-G. Reinhard
M. Seve Dinh
E. Suraud
M. Vincendon

February 7, 2019

Contents

1 Prerequisites

To successfully obtain and install the code the following minimum list of requirements has to be obtained

- Internet connection
- Git
- Fortran compiler
- C compiler *(optional)*.

2 Installation

This section will concern itself with how and where to get the code and how to compile it. For the examples treated here the most basic settings are choses so as to minimise the risk of complications. For the full list of compilation parameters and supported libraries, please consult the *QDD Reference Manual*.

2.1 Obtaining the code

To obtain the code, only one command in a terminal window is required. Change to the directory where you want the top level directory of the code to reside and execute

```
$ git clone https://github.com/erltls2018/QDD.git
```

This will create a directory '**QDD**' that contains the software package. The directory

```
/path/you/chose/QDD/
```

will be referred to in this guide as the '**$QDD_ROOT**'. After the download is complete navigate to the Fortran source directory:

```
$ cd $QDD_ROOT/src/qdd
```

2.2 Compilation

To get everything up and running as easy and fast as possible we will compile the QDD package with the default '**Makefile**'. This means QDD will be using:

- Fast Fourier transforms from the *Netlib FFTPACK*
- The *GNU Fortran* compiler **gfortran**, that can be downloaded here: https://gcc.gnu.org/wiki/GFortran
- No OpenMP parallelisation
- No MPI parallelisation

Different compilers, Fast Fourier libraries and parallelisation options can be chosen as well. This information can be found in the *QDD Reference Manual*.

If all prerequisites are met, simply execute

```
$ make
```

After the build process is finished the executable '**qdd**' will be in the "bin"-directory

```
$ $QDD_ROOT/bin/qdd
```

TABLE 1: Minimum set of input files.

`for005`	top level file containing the calculation identifier '`<name>`'. This can be any string of characters, e.g. '`h2o`' or '`na8`'.
`for005.<name>`	top level file containing the main parameters of the calculation using Fortran's *namelist*-mechanism. For description of all these input parameters see Tables 2, 3, 4, 6, 7, 8, and 11.
`for005ion.<name>`	top level file containing the locations and types of the ions.

3 Basic I/O structure of a ground state calculation

3.1 Input files & input parameters

Each calculation will have its own directory and in it its own set of input files where all the details and the initial conditions of the system to be calculated are set. While the calculation is running, screen output and output files are generated that give information about the current state of the system while it is running and also about the final state when the calculation is finished. There are a minimum of 3 files required to start a calculation. They are listed in Table 1.

For the input file '`for005.<name>`' there is a minimum set of input parameters. The ones related to a static calculation are listed and explained in Tables 2, 3 and 4. A complete list of all the input parameters can be found in the *QDD Reference Manual*.

The input file '`for005ion.<name>`' contains information about the ion cores. Each line in the file represents one ion core and each line is composed of several fields like so

$$\underbrace{x_n \;\; y_n \;\; z_n}_{①} \quad \underbrace{cen_n}_{②} \quad \underbrace{ijk}_{③} \quad \underbrace{r_n}_{④} \quad \underbrace{\mathcal{S}_n}_{⑤}$$

① are the (x, y, z)-coordinates of ion core n

② is the chemical element number in the periodic table of ion-core n

③ is the node ordering in repeat initialisation, where $(i, j, k) \in \{x, y, x\}$

④ is radius of the initial Gaussian around ion-core n

⑤ is start spin for initialisation at ion-core n

For example, an ion input file for H_2O (*say* **for005ion.H2O**) could look like

```
0.22835   0.00000   0.00000  8  xyz  1.0   1
-0.91350   1.47420  0.00000  1  xyz  1.0  -1
-0.91350  -1.47420  0.00000  1  xyz  1.0  -1
```

Examples of these input files can be found in '**$QDD_ROOT/examples/**'. In the following sections, example calculations of the covalent molecule water will be treated in ever increasing complexity.

3.2 Output files

During a calculation output files are generated and stored in the same directory as where the **qdd** binary is executed. For a ground state calculation the output files are listed in Table 5.

4 An example static calculation for H_2O

The input files for the water molecule can be found in

```
$QDD_ROOT/examples/H2O/
```

The first and simplest calculation is a ground state calculation with no ionic relaxation. The ion-cores are fixed in position while the electronic density relaxes around then. You will find the files for this calculation in

```
$QDD_ROOT/examples/H2O/static-no_ionmd/
```

As discussed in the previous section the 3 input files that you will find here are '**for005**', '**for005.H2O**' and '**for005ion.H2O**'. One can either copy these files to any desired location of, or run the '**qdd**' executable directly inside this directory, by executing

```
$ cd /path/to/the/for005*/files
$ $QDD_ROOT/bin/qdd > terminal.out 2> error.out
```

This will save the terminal output on the screen to the file '**terminal.out**' and any errors that might be generated during the calculation to '**error.out**'. Depending on the speed of your machine, after a few minutes you should have a file listing similar to this

Example directory listing

```
-rw------- 1 coppens   25 Feb  1 11:55 dx
-rw------- 1 coppens   34 Feb  1 11:57 error.out
-rw------- 1 coppens    4 Feb  1 11:52 for005
-rw------- 1 coppens 1.2K Feb  1 11:53 for005.H2O
-rw------- 1 coppens  160 Nov 28 11:12 for005ion.H2O
-rw------- 1 coppens  46K Feb  1 11:57 for006.0H2O
-rw------- 1 coppens 2.0K Feb  1 11:57 infosp.H2O
-rw------- 1 coppens   13 Feb  1 11:55 nx
-rw------- 1 coppens 1.3K Feb  1 11:55 poptions.H2O
-rw------- 1 coppens 2.3K Feb  1 11:57 pstat.H2O
-rw------- 1 coppens  41M Feb  1 11:57 rsave.H2O
-rw------- 1 coppens 138K Feb  1 11:57 terminal.out
```

For this calculation, the relevant observables are all in the file '**pstat.H2O**':

pstat.H2O

```
final protocol of static for IFSICP=  2
level:  1  spin,occup,ekin,esp,variance =  1  1.00000  1.47822 -2.32460  5.4639E-10
level:  2  spin,occup,ekin,esp,variance =  1  1.00000  2.96866 -1.46691  1.0425E-09
level:  3  spin,occup,ekin,esp,variance =  1  1.00000  3.03747 -1.18004  5.8210E-10
level:  4  spin,occup,ekin,esp,variance =  1  1.00000  3.36471 -1.02293  2.8506E-10
level:  5  spin,occup,ekin,esp,variance =  1  0.00000  0.80152 -0.43357  1.0256E-03
level:  6  spin,occup,ekin,esp,variance = -1  1.00000  1.47822 -2.32460  5.4639E-10
level:  7  spin,occup,ekin,esp,variance = -1  1.00000  2.96866 -1.46691  1.0425E-09
level:  8  spin,occup,ekin,esp,variance = -1  1.00000  3.03747 -1.18004  5.8210E-10
level:  9  spin,occup,ekin,esp,variance = -1  1.00000  3.36471 -1.02293  2.8506E-10
level: 10  spin,occup,ekin,esp,variance = -1  0.00000  0.80152 -0.43357  1.0256E-03
binding energy  =  -33.5155334
total variance  =  6.7185E-10
sp pot, sp kin, rearr, nonlocal=  -33.68712   21.69813   -1.10795    0.00000
e_coul: i-i , e-i , e-e , total=   13.79125 -105.00801   41.43268  -49.78408
mon.:   8.00
dip.in  :    0.00000    0.00000    0.00000
dip.out :   -0.05972    0.00000    0.00000
quadrupole moments:
xx,yy,zz:     0.8554     0.9404     0.7523
xy,zx,zy:     0.0000    -0.0000    -0.0000
spindip.:     0.0000     0.0000    -0.0000
omegam,rhops,N_el,rhomix:     0.0000     0.0000     0.0000     0.0000
```

The most important information in this file will be discussed in the next section.

4.1 Results and observables

From the file contents above we can immediately read off the following observables

Energies The most relevant energies can be readily read from the file. They are the total binding energy: **`binding energy`**; the energies of the electronic orbitals (kinetic and eigen energy): **`ekin, esp`**; the total energy of all the occupied s.p. orbitals **`sp pot`**; the kinetic energy of all the occupied s.p. orbitals **`sp kin`** and the different types of Coulomb interaction (ion-ion, electron-ion and electron-electron) **`e_coul: i-i, e-i, e-e, total`**

Ionisation potential From the list of occupied and unoccupied orbitals one can immediately read-off the ionisation-potential energy

HOMO–LUMO gap this can be easily obtained from the s.p. orbital list by finding the highest occupied state and the lowest unoccupied state and taking the difference.

Multipole moments The monopole-, dipole- and quadrupole are labeled **`mon., dip-in, dip-out`** and **`quadrupole moments`** respectively.

4.2 Ion-core relaxation using pseudo-dynamics

To allow the ion cores to move and relax into their equilibrium position we need to start a pseudo-dynamical calculation. During this calculation the ion-cores will move due to a dynamic force generated by the electron-ion potential.

Starting a calculation like this, we need to make a few modifications to out input files. A set of example files for H_2O is available in

`$QDD_ROOT/examples/H2O/static-ionmd/`

In this example the oxygen and two hydrogen ion cores are placed along he y-axis and the hydrogen cores at a slightly too small distance away from the oxygen core. So both the angle and the distances are wrong.

Make sure that in the **GLOBAL** namelist in **for005.H2O**

```
imob = 1
```

and that in the **DYNAMIC** namelist the following parameters are present and set to

```
ionmdtyp = 1
icooltyp = 1
ifredmas = 1
nabsorb = 0, jesc = 0
itmax = 25000
jpos = 10
jvel = 10
```

Make sure **ipsptype**$\neq$**0** and that **itmax** is set to a high enough value for the relaxation to finish. In this case the relaxation takes less than 1 fs. In this case $1\,\mathrm{fs}/(\hbar \times 0.0125\,\mathrm{Ry}^{-1})$=1654 iterations would be enough. Also it is better not to use any absorbing points (**nabsorb=0**, this automaticall implies **jesc=0**), and use reduced masses for the ions in the dynamics (**ifredmas=1**). You can monitor the progress of this minimisation by investigating **pposion.H2O** and **pvelion.H2O**.

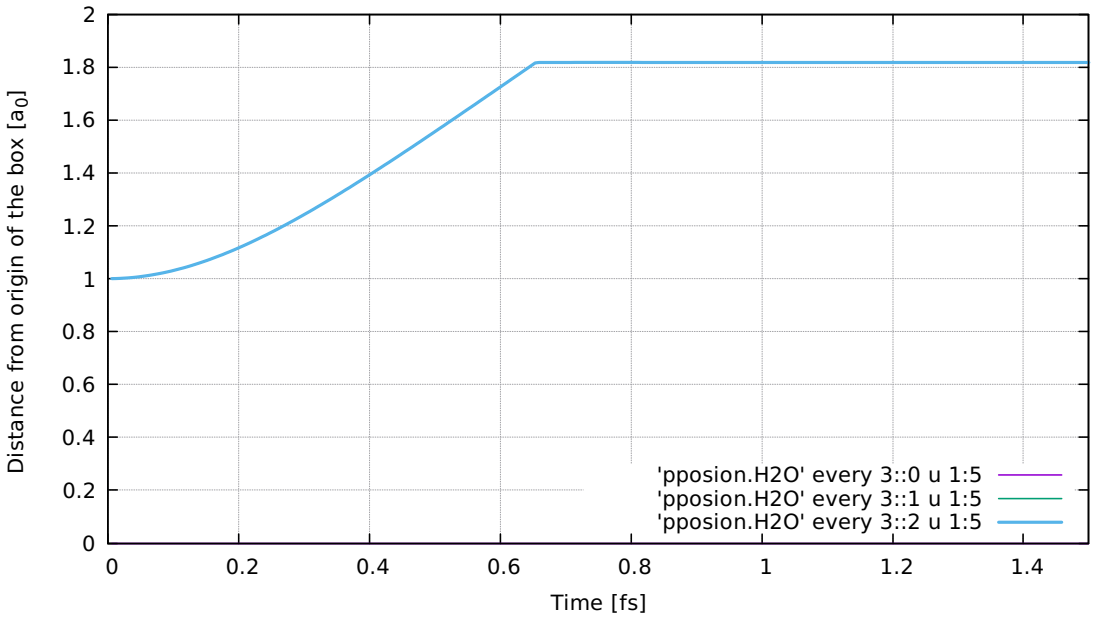

FIGURE 1: Shows the kinetic energy of the ion cores of oxygen and the two hydrogen atoms respectively as a function of time. After about 0.65 fs the ion cores have reached their equilibrium positions.

You can control the interval in number of iterations that the current position and velocity of the ions are printed to these files with **jpos** and **jvel**. Figures 1 and 2 shows the positions and kinetic energy of the ions during relaxation.

For more detailed information on the dynamic input parameters used here see Tables 6 and 8.

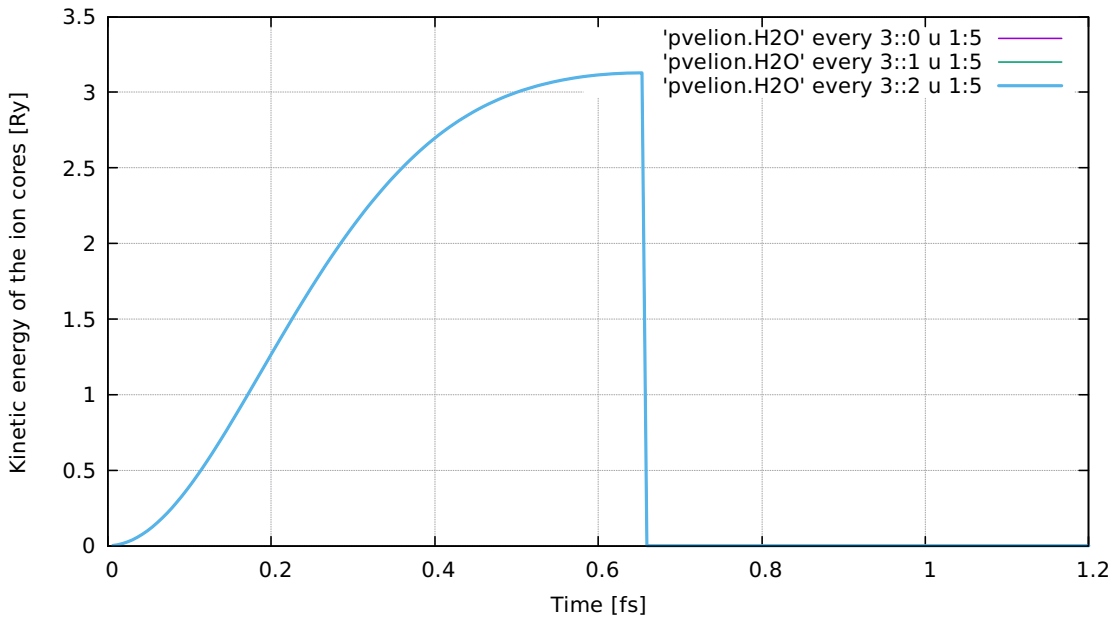

FIGURE 2: Shows the distance of the ion cores of oxygen and the two hydrogen atoms to the origin of the calculation box as a function of time. After about 0.65 fs the ion cores have reached their equilibrium positions.

5 Basic I/O structure of dynamic calculations

In this section the major input parameters will be listed and briefly explained as well as listing of all the generated output files.

5.1 Input parameters in the DYNAMIC namelist

A list of all the input parameters in the **DYNAMIC** namelist can be found in Tables 6, 7 and 8.

5.2 Output files

A list of all the output files and their description can be found in Tables 9 and 10.

6 An example dynamic calculation for H_2O

6.1 Excitation by applying a LASER pulse

In this example we will excite the water molecule by applying a resonant and off-resonant LASER pulse. The example input files can be found in

```
$QDD_ROOT/examples/H2O/dynamic/
```

The most important parameters that control the laser excitation are the following

```
itft = 3
deltat = 36.0
e1x = 0.0
e1y = 0.0
e1z = 1.0
omega = 0.79
e0 = 0.08
```

The be on-resonance we need an **omega** of 0.79 Ry which corresponds to 10.7 eV. In this case the pulse shape is a $\cos^2$, the pulse length is 36 fs and the pulse polarisation is in the z-direction. A list of all the excitation parameters can be found in Table 7.

To calculate the off-resonant case, simply change the value of **omega** to another value, e.g. 0.691 Ry, or 9.4 eV.

6.2 Results and observables

Three basic observables, the ionisation, absorption and dipole moment are shown in Figures 3 and 4 (the green curves). Figure 3 shows the time evolution of the three key observables considered in the present study. The dipole signal (lower panel) starts like an off resonant response where laser pulse and dipole signal follow the same pattern. At about 15 fs, we see a sudden increase in response and the dipole signal quickly deviates from the laser pulse, a typical pattern of resonant excitation.

7 Electronic relaxation with RTA

7.1 Additional DYNAMIC input parameters concerning RTA

One can simply use the input files from the previous section at

```
$QDD_ROOT/example/H2O/dynamic/
```

and enable the RTA procedure by choosing an iteration interval for an RTA step to happen. If want an RTA-step to happen every 1000 iteration set

```
jrtaint = 1000
```

A list of all the input parameters in the **DYNAMIC** namelist related to RTA can be found in Table 11.

7.2 Output files

A list of all the output files and their description can be found in Table 12.

7.3 The effect of dissipation on resonant and off-resonant excitation of H_2O

Of particular interest is the performance of RTA as compared to LDA. In Figure 3 for the first 15 fs there is practically no difference between LDA and RTA. Dissipation depends strongly on internal excitation energy and is thus inactive in the early stages, where not yet enough energy has been absorbed (middle panel). Later on, the difference grows dramatic. Dipole oscillations last very long after the pulse in the case of LDA, while RTA produces considerable attenuation.

Dissipation produces also a different evolution of energy absorption (middle panel) to the extent that more energy can flow into the system. The mechanism is a suppression of induced emission. If a system undergoes dipole oscillations in resonance with the laser field photon emission is coherently amplified and at a certain stage more energy is emitted than absorbed. This is the mechanism known as Rabi oscillations in simple two-level systems, and it leads to the observed oscillations in E_{abs}. Unlike in two-level systems, the photon couples here to several dipole modes which reduces the amplitude of the Rabi oscillations. Dissipation, furthermore, transfers energy out of the resonant dipole channel into intrinsic excitation as seen from the then smaller dipole amplitude (lower panel). This reduces induced emission and thus gives the system new capacity to swallow more energy. It is a

question of subtle energy balance in the system how this surplus on energy is distributed.

The upper panel of Figure 3 shows the time evolution of ionisation. Dissipation in RTA distracts electrons from their way to direct emission and redirects them to the system enhancing its intrinsic energy. Thus ionisation is with RTA systematically lower than with LDA. Another difference is seen in the long-time behaviour. In RTA, ionisation levels off after the pulse is over while it carries on in LDA. The latter is related to the long lasting dipole oscillations which continue to feed the emission channel.

Figure 4 shows time evolution for off-resonant excitation of H_2O. In fact, we see that the case shows traces of resonance response in the very early stages. But later on, the dipole signal (lower panel) follows in the average the pulse envelope and, most important, it nearly disappears with the pulse dying out. Being not perfectly off-resonant, the system's modes remain somewhat excited as one can see from energy absorption (middle panel) not fully returning to zero. It is much smaller than in the resonant case, but it is not returning fully back to zero which indicates some energy transfer away from the coherent dipole oscillations into other systems modes. The energy thus absorbed is distributed into continuum modes which leads to direct ionisation (see upper panel) and into bound modes which leaves eventually some final intrinsic excitation.

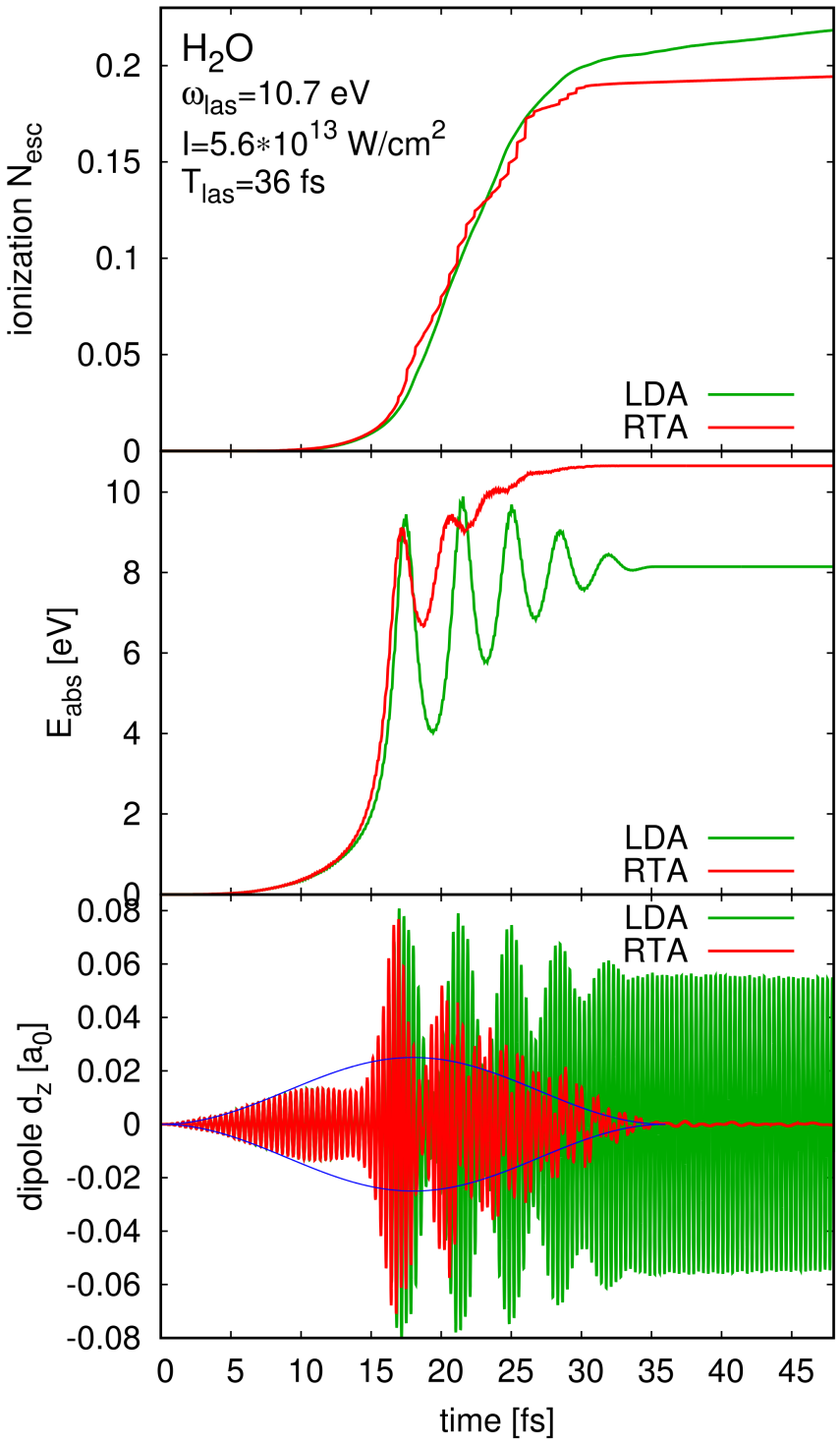

FIGURE 3: Comparison of LDA and RTA time evolution of three basic observables for H_2O: dipole moment perpendicular to molecular plane (lower), energy absorbed from laser field (middle), and net ionisation (upper). In the lower panel, the pulse envelope is indicated in addition to the dipole moments. The system was excited by a laser pulse with frequency $\omega_{\text{las}} = 10.7$ eV, intensity $I = 5.6 \times 10^{13}$ W/cm^2, and total pulse length $T_{\text{pulse}} = 36$ fs.
Image from Phys. Plasmas ***25****, 031905 (2018).*

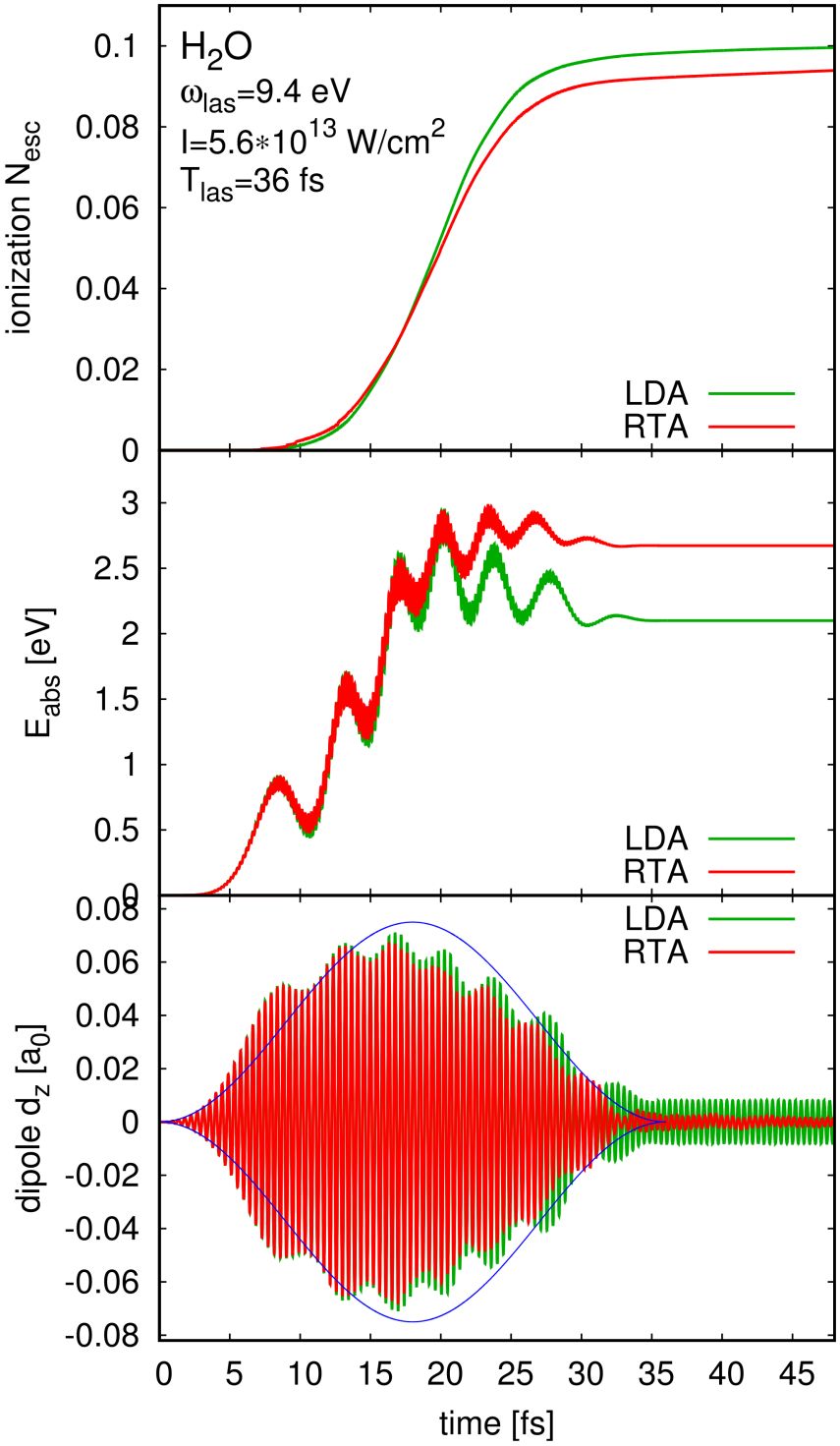

FIGURE 4: Same as Figure 3 but off-resonance at $\omega_{\text{las}} = 9.4$ eV.
*Image from Phys. Plasmas **25**, 031905 (2018).*

TABLE 2: System choice definitions in the 'GLOBAL' namelist in **for005.<name>**

The **GLOBAL** namelist	
concerning system choices	
kxbox, **kybox**, **kzbox**	number of grid points in the $\{x, y, z\}$-direction. A typical value is 64 points in each direction. The box sizes must fulfil the condition: **kxbox** $\geq$ **kybox** $\geq$ **kzbox**.
kstate	maximum number of possible single-particle (s.p.) states (can be larger than **nclust**)
numspin	number of spin components 1 $\rightarrow$ spin averaged (possible problem for ADSIC) 2 $\rightarrow$ full spin treatment)
nclust	number of QM electrons; if set to 0 or a negative value (charge) this will be automatically calculated: **nclust** $= \sum_{i=1}^{n_{ion}} Z_{ion}$ = charge, where Z_{ion} is the charge of each ion
nion	number of cluster ions
nspdw	number of spin down electrons
nion2	selects type of ionic background 0 $\rightarrow$ jellium background 1 $\rightarrow$ background from ionic pseudo-potentials 2 $\rightarrow$ background read in from **potion.dat**
radjel	Wigner-Seitz radius of jellium background
surjel	surface thickness of jellium background
bbeta	quadrupole deformation of jellium background
gamma	triaxiality of jellium background
dx,dy,dz	grid spacing (in Bohr) for the 3D numerical grid. If negative, this will be set to an optimal value, a value for **kxbox** will be suggested in the file '**nx**', the code stops and has to be restarted. The grid size is defined before compilation in **params.F90** and it has to correlate with the pseudo-potentials corresponds to ecut in solid state
imob	global switch to allow ionic motion (if set to 1)
isurf	switch for Ar or MgO surface (isurf=1 activates surface)
iDielec	switch to dielectic support
xDielec	x below which dielectric zone is activated
epsDi	dielectric constant in the dielectric zone
rotclustx,y,z	vector fo angle of initial rotation of ions

TABLE 3: Wave function initialisation in the GLOBAL namelist in **for005.<name>**

The **GLOBAL** namelist	
concerning the initialisation of wave functions	
b2occ	deformation for initial harmonic oscillator wf's
gamocc	triaxiality for initial harmonic oscillator wf's
deocc	shift of initial Fermi energy (determines nr. of states)
shiftWFx	shift of initial wave functions in x-direction
ishiftCMtoOrigin	switch to shift centre of mass of cluster to origin
ispinsep	initialise wave functions with some spin asymmetry
init_lcao	choice of basis for wave function initialisation 0 $\rightarrow$ harmonic oscillator functions (centre can be moved by **shiftWFx**) 1 $\rightarrow$ atomic orbitals: wave functions are centred at ionic sites

TABLE 4: Convergence parameters in the GLOBAL namelist in **for005.<name>**

The **GLOBAL** namelist	
concerning convergence issues	
e0dmp	damping parameter for static solution of Kohn-Sham equations (typically about the energy of the lowest bound state)
epswf	step size for static solution of Kohn-Shahm equations (of order of 0.5)
epsoro	required variance to terminate static iteration (of order 10^{-5})

TABLE 5: Output files generated during a *static* calculation

dx	grid spacing in units of Bohr.
nx	this file contains a suggested value for **kxbox** when **dx** is set to a negative value
for006.0<name>	protocol file.
infosp.<name>	energy and variances at given iteration numbers determined by the variable **jinfo** in the **DYNAMIC** namelist.
poptions.<name>	this file contains an overview of the chosen options on solvers, compiler options, etc.
pstat.<name>	contains the final information about the single particle energies, spins, variances, occupation numbers, monopole-, dipole- and quadrupole moments, etc.

TABLE 6: Dynamical paramaters in the DYNAMIC namelist in **for005.<name>**

The **DYNAMIC** namelist	
numerical and physical parameters for statics and dynamics	
dt1	time step for propagating electronic wave functions, $\frac{\Delta t}{\Delta x^2} \leq 1$
ismax	maximum number of static iterations
idyniter	switch to s.p. energy as E0DMP for 'iter>idyniter'
ifhamdiag	diagonalization of m.f. Hamiltonian in static step (presently limited to fully occupied configurations)
isitmax	nr. of imaginary-time steps to improve static solution
itmax	number of time steps for electronic propagation
ifexpevol	exponential evolution 4. order instead of TV splitting
iffastpropag	accelerated time step in TV splitting (for pure electron dynamics, interplay with absorbing b.c. ??)
irest	switch to restart dynamics from file 'save'
istat	switch to read wavefunctions from file 'rsave' • it continues static iteration for 'ismax>0' • it starts dynamics from these wf's for 'ismax=0'
idenfunc	choice of density functional for LDA 1 → Perdew & Wang 1992 (default setting) 2 → Gunnarson & Lundquist 3 → only exchange in LDA
isave	saves results after every 'isave' steps on file 'rsave' in and after static iteration on file 'save' in dynamic propagation
ipseudo	switch for using pseudo-densities to represent substrate atoms
ipsptype	type of pseudopotentials: 0 = soft local (errf); 1 = full Goedecker; 2 = local Goedecker; 3 = read from file goed.asci (no need to specify) ; 4 = semicore read from file goed.asci
directenergy	**.true.** = direct computation of energy (only for **LDA**, **Slater**, **KLI**)
ifsicp	selects type of self-interaction correction 0 = pure LDA, 1 = SIC-GAM, 2 = ADSIC; 3 = SIC-Slater; 4 = SIC-KLI; 5 = exact exchange; 6 = inactive; 7 = localized SIC; 8 = full SIC (double set). IFSICP=7 or 8 requires switch **twostsic=1** in **define.h**. Option IFSICP=7 needs yet testing.
icooltyp	type of cooling (0=none, 1=pseudo-dynamics, 2=steepest descent, 3=Monte Carlo)
ifredmas	switch to use reduced mass for ions in dynamics
ionmdtyp	ionic propagation (0=none, 1=leap-frog, 2=velocity Verlet)
ntref	nr. time step after which absorbing bounds are deactivated
nabsorb	number of absorbing points on boundary (0 switches off)
powabso	power of absorbing boundary conditions
ispherabso	switch to spherical mask in absorbing bounds

TABLE 7: Dynamical paramaters in the DYNAMIC namelist in **for005.<name>**

The **DYNAMIC** namelist	
way of excitation	
centfx	initial boost of electronic wavefuncftions in x-direction
centfy	initial boost of electronic wavefuncftions in y-direction
centfz	initial boost of electronic wavefuncftions in z-direction
tempion	initial temperature of cluster ions
ekmat	initial kinetic energy of substrate atom (boost in x, in eV)
itft	choice of shape of laser pulse 1 = ramp laser pulse, sine switching on/off 2 = gaussian laser pulse 3 = $\cos^2$ pulse
tnode	time (in fs) at which pulse computation starts
deltat	length of ramp pulse (**itft = 1**), in fs
tpeak	time (in fs, relative to **tnode**) at which peak is reached (for **itft** = 1 and 2, pulse length becomes 2*__tpeak__)
omega	laser frequency (in Ry)
e0	laser field strength in Ry/Bohr
e1x,e1y,e1z	orientation of pulse
e0_2	field strength of second laser pulse (only **itft=3**)
phase2	phase of second pulse
omega2	frequency of second pulse
tstart2	initial ime of second pulse
tpeak2	peak time of 2. pulse (pulse length is **2*tpeak2**)
iexcit	modus of excitation (0=shifts, 1=rotation)
iangmo	switch to compute angular momentum
irotat	axis of rotation for excitation (x=1,y=2,z=2,xyz=4)
phirot	angle of rotation for excitation (in units of degree)
phangle	angle of "rotation" into a $1ph$ state
phphase	phase of "rotation" into a $1ph$ state
nhstate,npstate	nr. of hole and particle state for $1ph$ excitation this $1ph$ option can only be run from **istat=1**
eproj	energy of incoming projectile (= last ion in the list)
vpx,vpy,vpz	direction of the incoming projectile
taccel	time span over which the projectile is accelerated to **eproj** for **taccel=0** one has to use **init_lcao=1**

TABLE 8: Parameters that control observables and output in the DYNAMIC namelist in **for005.<name>**

The **DYNAMIC** namelist	
flags for observables	
iemomsRel	calculates multipole momentas of electron density relative to origin (0) or c.m. of cluster (1)
istinf	modulus for printing information in static iteration
ifspemoms	switch to compute and print spatial s.p. moments
iftransme	switch to compute and print transition m.elements
ifrhoint_time	switch to slices of integrated densities for all times
jstinf	modulus for printing information in dynamic
jinfo	modulus for printing dynamical information on **infosp.<name>**
jdip	modulus for printing dipole moments on **pdip.<name>**
jquad	modulus for printing quadrupole moments on **pquad.<name>**
jesc	modulus for printing ionization **pescel.<name>**
jenergy	modulus for printing energy information on **penergies.<name>**
iflocaliz	activates computation of Becke's localisation
jelf	modulus for analysing and printing electron localisation in dynamics various files are written of the form **pelf*.<name>**
iflocaliz	modulus for analysing and printing electron localisation in statics
jstinf	modulus for printing s.p. energies and variances
jpos	modulus for printing ionic positions on **pposion.<name>**
jvel	modulus for printing ionic velocities on **pvelion.<name>**
jstateoverlap	switch to compute overlap of static state with the state directly after dynamical initialisation

TABLE 9: Output files generated during a dynamic calculation

energies.<name>	historical, contains only the binding energy
forces.<name>	forces on ions, generated when ion molecular dynamics is active
<name>.bs	output suited for further processing by freeware which can make 3D structure plots of molecular configurations in the typical chemical style
pdip.<name>	dipole moment in x, y, z direction, versus time
penerclu.<name>	kinetic energy of the cluster in the x,y,z directions and total, versus time, at intervals commanded by the input parameter jener
pescel.<name>	proportion of electrons remaining, total number of electrons, number of electrons lost, versus time, at intervals commanded by the input parameter jesc
plaser.<name>	laser parameters Ex, Ey, Ez, power, laser energy, etc as a fonction of time
povlp.<name>	unused in this version
penergies.<name>	Various energies, versus time. The 26 detailed entries (single particle energy, rearrangement energy, etc..) are described in the output file itself. The total energy is at location 18.
pescOrb.<name>	Number of electrons lost per orbital, versus time, at intervals commanded by the input parameter jnorms
pkinenion.<name>	kinetic energy of the cluster in the x,y,z directions and total, versus time, at intervals commanded by the input parameter jpos
pPES.<name>	unused in this version

TABLE 10: Output files generated during a dynamic calculation (*continued*)

pposion.<name>	positions of the individual ions in x,y,z, and distance to center, versus time, at intervals commanded by the input parameter jpos
pproba.<name>	probabilities of charge states versus time, at time intervals commanded by input parameter jnorms
pprojdip.<name>	x, y, z pos of projectile versus time, at time intervals commanded by input parameter jdip
prhov.<name>	unused in this version
progstatus	Only a flag when dynamics are finished
pspenergies.<name>	single particle energies versus time, at time intervals commanded by input parameter jinfo
pspvariances.<name>	single particle energy variances versus time, at time intervals commanded by input parameter jinfo
pspvariancesp.<name>	single particle energy variances versus time (with correction by projection), at time intervals commanded by input parameter jinfo
ptempion.<name>	ion temperatures during ionic-core relaxation
pvelion.<name>	ion velocities during ionic-core relaxation, or dynamic calculation with molecular dynamics
rsave.<name>	This file contains all parameters of a static convergence to allow for a dynamic start without recomputing the statics: to use it set ismax=0 and istat=1
save.<name>	This file contains all parameters to allow for a dynamic start at time : to use it set irest>=0
Time	Number of points in the calculation box and used wall time to complete the given number of iterations

TABLE 11: Parameters that control RTA in the DYNAMIC namelist in **for005.<name>**

The **DYNAMIC** namelist	
flags for RTA	
jrtaint	Modulus for calling the RTA subroutine, i.e., nr. of TDLDA steps per one RTA step. Course time step Δt for RTA and fine time step for TDLDA **dt1** are related as Δt =**jrtaint**∗**dt1**.
rtamu	Parameter μ in front of the quadratic density constraint in the DCMF Hamiltonian (**??**)
rtamuj	Parameter μ_j in front of the quadratic current constraint in the DCMF Hamiltonian (**??**)
rtasumvar2max	Termination criterion ϵ_0 in the RTA step as used in figure **??**
rtaeps	Step size δ in the damping operator (cross ref to be defined) $\mathcal{D}$ for the RTA step.
rtae0dmp	Energy offset E_{00} in the damping operator (cross ref to be defined) $\mathcal{D}$ for the RTA step
rtasigee	In medium $e^- - e^-$ cross section used for the relaxation time (**??**)
rtars	Effective Wigner-Seitz radius r_s used for the relaxation time (**??**)
rtatempinit	The value **rtatempinit**/10 is used as lower value for the search of temperature in **SUBROUTINE ferm1**

TABLE 12: Output files that are specific to a RTA-enabled calculation

convergenceRTA	this file contains a log of the RTA iterations that contains information on the convergence details
peqstate	parameters for convergence of the dtmf process: current iteration number, cycles to convergence, variance, residual err. on density, residual err. on current, parameters mu, muj, energy achieved
prta	prints at each RTA step: time, entropy, laser energy and the mu and temperature of a fermi distribution fitted to the occupation numbers
pspeed.<name>	prints at each RTA step, along x axis, the reference density (spin up and down), achieved density (spin up and down), target x current, achieved x current