Quantum Dissipative Dynamics

User manual

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

February 4, 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.

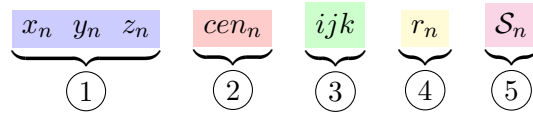| for005	top level file containing the calculation identifier '`<name>`'. This can be any string of characters, e.g. '`h2o`' or '`na8`'.
for005.`<name>`	top level file containing the main parameters of the calculation using Fortran's *namelist*-mechanism. For description of all these input parameters see Tables 2, 3, 4, 6, 7, 8, and 10.
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

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

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 illustrative ground state calculation: H_2O

The input files for the water molecule can be found in

`$QDD_ROOT/samples/H2O/`

As discussed in the previous section the 3 files that you will find here are '`for005`', '`for005.h2o`' and '`for005ion.h2o`' respectively. You can either copy these files to a location of your choosing, or simply run the '`qdd`' executable in each 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

```
-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
```

In the file '`pstat.H2O`' you will find

```
final protocol of static for IFSICP=  2
level:  1  spin,occup,ekin,esp,variance =  1  1.00000   1.31912 -2.33723   5.8472E-10
level:  2  spin,occup,ekin,esp,variance =  1  1.00000   2.51534 -1.59199   5.3699E-10
level:  3  spin,occup,ekin,esp,variance =  1  1.00000   2.46317 -1.25959   5.3194E-10
level:  4  spin,occup,ekin,esp,variance =  1  1.00000   2.74713 -1.11479   5.7191E-10
```

```
level:  5  spin,occup,ekin,esp,variance =  1  0.00000   0.71328 -0.42203   7.7313E-07
level:  6  spin,occup,ekin,esp,variance =  1  0.00000   0.73195 -0.26481   2.9550E-03
level:  7  spin,occup,ekin,esp,variance =  1  0.00000   0.30830 -0.21468   5.6298E-03
level:  8  spin,occup,ekin,esp,variance =  1  0.00000   0.31976 -0.21151   6.3903E-03
level:  9  spin,occup,ekin,esp,variance =  1  0.00000   0.35431 -0.15832   7.9917E-03
level: 10  spin,occup,ekin,esp,variance =  1  0.00000   0.20732 -0.12894   1.4731E-03
level: 11  spin,occup,ekin,esp,variance = -1  1.00000   1.31912 -2.33723   5.8472E-10
level: 12  spin,occup,ekin,esp,variance = -1  1.00000   2.51534 -1.59199   5.3699E-10
level: 13  spin,occup,ekin,esp,variance = -1  1.00000   2.46317 -1.25959   5.3194E-10
level: 14  spin,occup,ekin,esp,variance = -1  1.00000   2.74713 -1.11479   5.7191E-10
level: 15  spin,occup,ekin,esp,variance = -1  0.00000   0.71328 -0.42203   7.7313E-07
level: 16  spin,occup,ekin,esp,variance = -1  0.00000   0.73195 -0.26481   2.9550E-03
level: 17  spin,occup,ekin,esp,variance = -1  0.00000   0.30830 -0.21468   5.6298E-03
level: 18  spin,occup,ekin,esp,variance = -1  0.00000   0.31976 -0.21151   6.3903E-03
level: 19  spin,occup,ekin,esp,variance = -1  0.00000   0.35431 -0.15832   7.9917E-03
level: 20  spin,occup,ekin,esp,variance = -1  0.00000   0.20732 -0.12894   1.4731E-03
binding energy  =  -32.6355092
total variance  =   5.5684E-10
sp pot, sp kin, rearr, nonlocal=  -30.69673   18.08954   -1.06211    0.00000
e_coul: i-i , e-i , e-e , total=   13.54907  -97.56193   39.39447  -44.61839
mon.:   8.00
dip.in  :    0.00000    0.00000    0.00000
dip.out :   -0.06362    0.00000    0.00000
quadrupole moments:
xx,yy,zz:     0.9037     0.9913     0.7949
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
```

4.1   Results and observables

Other than the terminal output there are also a number of other files created during and when the calculation is finished.

5   Basic I/O structure of dynamic calculations

6   Example dynamic calculations

7   Electronic relaxation with RTA

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

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. NOT SURE ABOUT THIS. THE VALUE IN THIS FILE SEEMS TO CONFLICT WITH THE VALUE SET IN THE INPUT FILE
`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

`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
`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 10: 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 $\Delta t = $`jrtaint`$*$`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 11: 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