Qbox User Guide

version 1.45.0

2008-09-15

François Gygi
fgygi@ucdavis.edu
http://eslab.ucdavis.edu

1 Introduction

Qbox is a First-Principles Molecular Dynamics code. It can be used to compute the electronic structure of atoms, molecules, solids and liquids within the Density Functional Theory (DFT) formalism. It can also be used to perform first-principles molecular dynamics (FPMD) simulations using forces computed within DFT. Qbox computes the solutions of the Kohn-Sham equations using a plane-wave basis set and norm-conserving pseudopotentials. It can perform constant-temperature and/or constant-pressure simulations.

2 Qbox distribution

Qbox is distributed in source form under the GPL license, and in executable form for specific platforms. For source distribution see http://eslab.ucdavis.edu/software/qbox . Qbox has been built and run on the following combinations of platforms and systems:

1. AMD Opteron 270 - Infinipath / CentOS /Rocks
2. Intel Xeon x86_64/ Fedora6
3. BlueGene/L SDSC, LLNL, ANL
4. IBM p655 Series, Datastar SDSC
5. Teragrid Mercury
6. Teragrid Cobalt
7. Teragrid Tungsten
8. BlueGene/P, ANL
9. Cray XT4 (NERSC franklin)

Some precompiled executables are available at http://eslab.ucdavis.edu .

3 Basic Qbox operation

3.1 Starting Qbox

On most platforms, Qbox is started using the `mpirun` command. On a Linux platform using the mpich implementation of MPI, the command is

```
mpirun -np <ntasks> qbox_exec input_file > output_file
```

The input file contains a list of commands that are read and executed by Qbox. The output file created by Qbox is a valid and well formed XML document. It can be used to extract various diagnostic information after the run is completed. During a Qbox run, the output file can be used to track the progress of the simulation.

3.2 Qbox commands

When used in interactive mode, Qbox prints a prompt

```
[qbox]
```

and waits for the user to type a command. When used in batch mode, Qbox reads input from an input file. Output is written on standard output (stdout) and can be redirected to a file. A Qbox input file consists of a sequence of Qbox commands with one command per line. Qbox commands are

angle	compute the angle formed by three atoms
atom	define an atom
compute_mlwf	compute maximally localized Wannier functions
constraint	define or delete a constraint
distance	compute the distance between two atoms
fold_in_ws	fold atoms into the Wigner-Seitz cell
help	print a short message about the use of a command
kpoint	add or remove k-points
list_atoms	print a list of currently defined atoms
list_species	print a list of currently defined atomic species
load	load a sample from a file previously saved
move	move atoms
print	print the value of a Qbox variable
quit	exit Qbox
randomize_wf	add random noise to the wavefunction coefficients
reset_vcm	set the velocity of the center of mass to zero
run	run MD or electronic optimization steps
save	save a sample on a file for later use
set	assign a value to a Qbox variable
species	define a new atomic species
status	print a summary of the current state
strain	impose a strain on the sample
torsion	compute the dihedral angle defined by four atoms
!(shell escape)	execute a shell command

Commands and their syntax are described in detail in the Section "Qbox commands".
If a command is read on input and is not in the above list, Qbox interprets it as the name of an input script and attempts to open a file having that name in the current working directory. If the file can be opened and is readable, Qbox starts interpreting each line of that file as its input. Qbox scripts can be nested. At the end of a script, Qbox returns to the previous script level and continues to read commands. At the end of the topmost level script, Qbox exits.
Unix commands can be issued within a Qbox input sequence using a shell escape "!" character at the beginning of a line. For example, the line

```
[qbox] !date
```

invokes the Unix `date` command.
Comments can be inserted in Qbox input by inserting a '#' character at the beginning of each comment line

```
[qbox] # print the list of all atoms
[qbox] list_atoms
```

Comments can also be added at the end of a command line by inserting a "#" character where the comment starts

```
[qbox] list_atoms # get a list of all atoms
```

3.3 Qbox variables

Calculation parameters such as the plane-wave energy cutoff are specified using Qbox variables. Qbox variables can be set using the `set` command. Their value can be printed using the `print` command. For example the command

```
set ecut 24
```

sets the variable `ecut` to the value 24. This causes the plane wave basis set to be resized to include all plane waves with a kinetic energy not exceeding 24 Rydberg. Other Qbox variables can be set similarly. The Qbox variables are

atoms_dyn	ionic dynamics control variable
cell	dimensions of the unit cell
cell_dyn	unit cell dynamics control variable
cell_lock	control of allowed unit cell motions
cell_mass	fictitious mass of the unit cell
charge_mix_coeff	mixing parameter for charge density update
charge_mix_rcut	Kerker screening for charge density update
debug	debug parameters (not for normal use)
dt	time step (a.u.)
ecut	plane wave energy cutoff (Ry)
ecutprec	energy cutoff of the preconditioner (Ry)
ecuts	energy cutoff of the confinement potential
emass	fictitious electronic mass (for CP dynamics)
ext_stress	externally applied stress (GPa)
fermi_temp	Fermi temperature (K)
nempty	number of empty states
net_charge	net charge of the system
nrowmax	maximum size of process grid columns
ref_cell	dimensions of the reference unit cell
stress	stress control variable
thermostat	thermostat control variable
th_temp	thermostat temperature (K)
th_time	thermostat time constant (a.u.)
th_width	thermostat width (K)
wf_diag	wavefunction diagonalization control variable
wf_dyn	wavefunction dynamics control variable
xc	exchange-correlation control variable

Qbox variables have a default value. Refer to the Section “Qbox variables” for details.

3.4 Structure of a Qbox script

A Qbox script starts with commands defining the sample being simulated. For example, the sequence of commands

```
set cell 20 0 0  0 20 0  0 0 20
species silicon silicon.xml
atom Si1 silicon  3.000  0.000  0.000
atom Si2 silicon  0.000  2.000  0.000
atom Si3 silicon -3.000  0.000  0.000
atom Si4 silicon  0.000 -2.000  0.000
```

defines a sample in a cubic unit cell of size 20 a.u. (Bohr radii). The sample consists of a 4-atom silicon cluster. The definition of the species "silicon" is given in the file `silicon.xml.` A species definition document contains all the information needed to describe an atomic species, including the pseudopotential used to represent the electron-ion interaction. Species can also be defined by a URI as in the following command

```
species carbon http://www.example.org/species/carbon.xml
```

Species definition documents are XML documents that follow the syntax defined by the `http://www.quantum-simulation.org` XML Schema definition.

4 Examples of use of Qbox

4.1 Electronic structure of a Si_4 cluster

The following example shows how to use Qbox to compute the electronic ground state of a silicon 4-atom cluster. Approximate input atomic coordinates are used. The electronic ground state is computed using 200 iterations of the preconditioned steepest descent algorithm with Anderson acceleration (PSDA).

```
# example: electronic structure of si4
set cell 20 0 0  0 20 0  0 0 20
species silicon silicon.xml
atom Si1 silicon  3.500  0.000  0.000
atom Si2 silicon  0.000  2.000  0.000
atom Si3 silicon -3.000  0.000  0.000
atom Si4 silicon  0.500 -2.000  0.000
set ecut 12
set wf_dyn PSDA
set ecutprec 4
randomize_wf
run 0 200
```

4.2 Structure optimization

The following example shows how to optimize the structure of a Si_4 cluster. The electronic ground state is computed using 200 iterations of the preconditioned steepest descent algorithm (PSDA). The structure is then optimized using 50 steps of the steepest descent with Anderson acceleration (SDA) algorithm. Five steps of electronic optimization are performed before each ionic step.

```
# example: structure optimization of Si4
set cell 20 0 0  0 20 0  0 0 20
species silicon silicon.xml
atom Si1 silicon  3.500  0.000  0.000
atom Si2 silicon  0.000  2.000  0.000
atom Si3 silicon -3.500  0.000  0.000
atom Si4 silicon  0.000 -2.000  0.000
set ecut 12
set wf_dyn PSDA
set ecutprec 4
randomize_wf
run 0 200
```

```
# start structure optimization
set atoms_dyn SDA
set dt 100
run 50 5
```

4.3 Molecular dynamics simulation

The following example shows how to perform a Born-Oppenheimer molecular dynamics simulation. The electronic ground state is first computed using 200 iterations of the preconditioned steepest descent algorithm (PSDA). A Born-Oppenheimer MD is then done by running 50 ionic steps with 5 electronic iterations at each ionic step.

```
# example: molecular dynamics of si4
set cell 20 0 0  0 20 0  0 0 20
species silicon silicon.xml
atom Si1 silicon  3.500  0.000  0.000
atom Si2 silicon  0.000  2.000  0.000
atom Si3 silicon -3.500  0.000  0.000
atom Si4 silicon  0.000 -2.000  0.000
set ecut 12
set wf_dyn PSDA
set ecutprec 4
randomize_wf
run 0 200
# start molecular dynamics
set atoms_dyn MD
set dt 40
run 50 5
```

4.4 Saving and restarting a simulation

The following example shows how to save the simulation sample on a file at the end of a simulation and how to restart from the saved file. In the first script, the electronic ground state is computed using 200 iterations of the preconditioned steepest descent algorithm (PSDA). A Born-Oppenheimer MD is then done by performing 50 ionic steps with 5 electronic iterations at each ionic step. The sample is saved at the end of the first script on file si4.xml. In the second script, the sample is loaded from the restart file and 50 additional MD steps are performed.

```
# script 1: electronic ground state and 50 MD steps
set cell 20 0 0  0 20 0  0 0 20
species silicon silicon.xml
atom Si1 silicon  3.500  0.000  0.000
atom Si2 silicon  0.000  2.000  0.000
atom Si3 silicon -3.500  0.000  0.000
atom Si4 silicon  0.000 -2.000  0.000
set ecut 12
set wf_dyn PSDA
set ecutprec 4
randomize_wf
run 0 200
# perform 10 MD steps
set atoms_dyn MD
set dt 40
run 50 5
```

```
# save the sample
save si4.xml


# script 2: restart from saved sample and
# perform 50 MD steps
load si4.xml
set wf_dyn PSDA
set ecutprec 4
set atoms_dyn MD
run 50 5
# save the sample
save si4.xml
```

4.5 Changing energy cutoff on the fly

The size of sample files can become large when simulating large systems. Although Qbox uses a parallel I/O algorithm to save sample data, the process of writing data to a file can be slow, especially on platforms that do not offer a parallel file system. This problem can be alleviated by saving the sample with reduced resolution for wavefunctions. The value of the variable `ecut` can be changed using the `set ecut` command at any time during the simulation. When the value of `ecut` changes, Qbox interpolates wavefunctions onto the plane wave basis defined by the new value. The size of the sample file can be reduced by changing the value of `ecut` to a small value before using the `save` command. When loading a sample that was saved with reduced resolution, the original resolution can be restored by first redefining the value of `ecut` and then reoptimizing the wavefunctions.
The ability to change energy cutoff on the fly is also useful to accelerate a ground state calculation by starting the calculation at low cutoff and gradually increasing the cutoff to the final desired value. This procedure can be compared to the full approximation scheme of the multigrid method.

5 Qbox commands

angle

NAME

angle --compute the value of the angle defined by the positions of three atoms

SYNOPSIS

angle *atom1 atom2 atom3*

DESCRIPTION

The angle command prints the value of the angle formed by the three atoms given as arguments. The names *atom1*, *atom2* and *atom3* must refer to the names of atoms currently defined in the sample.

RELATED INFORMATION

list_atoms, distance, torsion

atom

NAME

atom --add an atom to the current sample

SYNOPSIS

atom *name species x y z [vx vy vz]*

DESCRIPTION

The atom command adds an atom to the current sample. The *name* argument can be any character string but must differ from all the other names of atoms in the current sample. The *species* argument must refer to an atomic species previously defined using the species command. The position of the atom is specified by its coordinates *x*, *y*, *z* in atomic units (Bohr). Optionally, the velocity of the atom can be specified by its components *vx*, *vy*, *vz* in atomic units (Bohr/atomic-unit-of-time). (One atomic-unit-of-time is 0.02418885 fs).

RELATED INFORMATION

list_atoms, species

compute_mlwf

NAME

compute_mlwf –compute maximally localized Wannier functions

SYNOPSIS

compute_mlwf

DESCRIPTION

The compute_mlwf command transforms the current wavefunctions into maximally localized Wannier functions following the algorithm in Computer Physics Communications **155** , p. 1 (2003) and a one-sided iterative Jacobi algorithm for simultaneous diagonalization. The position of Wannier centers and the corresponding spreads are printed on output. The value of the electronic, ionic and total dipole are printed. The iterative methods stops when the decrease of the spread is sufficiently small between two iterations, or when a maximum number of iterations is reached. In that latter case, the compute_mlwf command can be issued again to try to improve the convergence of the spread minimization. After execution of the compute_mlwf command, the wavefunctions are maximally localized.

RELATED INFORMATION

constraint

NAME

constraint --define a constraint on atomic positions

SYNOPSIS

constraint distance *atom1 atom2 distance [velocity]*

constraint angle *atom1 atom2 atom3 angle [velocity]*

constraint torsion *atom1 atom2 atom3 atom4 angle [velocity]*

DESCRIPTION

The constraint command add an entry to the constraint set. The constraint can be either a distance, angle or torsion constraint. The atom names must refer to atoms previously defined using the atom command.

RELATED INFORMATION

list_atoms, angle, distance, torsion

distance

NAME

distance --print the distance between two atoms

SYNOPSIS

distance *atom1 atom2*

DESCRIPTION

The distance command prints the value of the distance separating two atoms. In periodic samples, the printed distance is not necessarily the *minimal* distance between the two atoms.

RELATED INFORMATION

angle, torsion

fold_in_ws

NAME

fold_in_ws –fold all atoms within the Wigner-Seitz cell

SYNOPSIS

fold_in_ws

DESCRIPTION

The fold_in_ws command moves atoms by multiples of the unit cell lattice vectors so that all atoms are withing the Wigner-Seitz cell.

RELATED INFORMATION

help

NAME

help --print a brief help message about a command

SYNOPSIS

help *[command]*

DESCRIPTION

The help command prints a short description of the *command* given as an argument. When used without arguments, prints a list of valid commands.

RELATED INFORMATION

kpoint

NAME

kpoint –define and manage the set of k-points used in the calculation of the electronic structure.

SYNOPSIS

kpoint list

kpoint add *kx ky kz weight*

kpoint delete *kx ky kz*

DESCRIPTION

The kpoint command is used to add and delete k-points to the set of k-points used in the electronic structure calculation. The kpoint is defined by a vector on the basis of the reciprocal lattice vectors. If reciprocal lattice vectors are $\mathbf{b}_1$ $\mathbf{b}_2$ and $\mathbf{b}_3$ the k-point defined by the numbers (*kx, ky, kz)* on the command line is $kx * \mathbf{b}_1 + ky * \mathbf{b}_2 + kz * \mathbf{b}_3$. For example, the *X* point of the Brillouin Zone for an FCC lattice is specified as *kx=0.5, ky=0.5, kz=0.0.*

The list of all currently defined k-points can be printed using the command `kpoint list`.

The sum of the *weight* arguments must add up to 1.0. This is currently not checked by Qbox.

Some k-points can be defined with zero weight. In that case, the electronic wavefunctions and eigenvalues are computed at these points, but they are not included in the calculation of the charge density.

By default, Qbox starts with a k-point set containing a single k-point: *k=(0,0,0) (*the Γ point) with a weight of 1.0. When defining a k-point set, it is necessary to first *delete* the Γ point before defining other k-points. This is due to two possible reasons:

1) The desired k-point set does not contain the Γ point.

2) The desired k-point set contains the Γ point, but with a weight different from 1.0. In this case, the Γ point must be deleted and then added in order to be defined with the correct weight. The only way to change the weight of a k-point is to delete it and define it again.

Qbox does not perform any symmetrization of the charge density to reduce the number of k-points to the irreducible wedge of the Brillouin Zone. The full set of k-points must be defined (except for the k, -k symmetry, i.e. If *k* is included in the set, then -*k* need not be included).

Modifying the k-point set erases all wavefunctions. It is not possible to modify the k-point set after running a calculation or after loading a sample without resetting the wavefunctions.

Note: When deleting a kpoint, the arguments *kx, ky, kz* are compared to the coordinates of the k-points currently defined. Since comparisons of floating point numbers are unreliable, the kpoint delete command will delete any k-point located within a radius of 10^{-6} of the vector (*kx, ky, kz*). Similarly, when adding a new k-point, the kpoint add command will exit without defining the new k-point and print a warning message if a previously defined k-point is located within a radius of 10^{-6} of the new kpoint.

EXAMPLE

Define a set of 8 k-points for a simple cubic or orthorhombic cell. This k-point set is equivalent to doubling the cell in all three directions

```
kpoint delete 0 0 0
# Gamma point
kpoint add  0.0  0.0  0.0   0.125
# X point
kpoint add  0.5  0.0  0.0   0.125
kpoint add  0.0  0.5  0.0   0.125
kpoint add  0.0  0.0  0.5   0.125
# R point
kpoint add  0.5  0.5  0.5   0.125
# M point
kpoint add  0.5  0.5  0.0   0.125
kpoint add  0.5  0.0  0.5   0.125
kpoint add  0.0  0.5  0.5   0.125
```

Define the *X* and *L* points in the Brillouin Zone of a FCC lattice, with zero weight:

```
# a1 = (a/2, a/2, 0)
# a2 = (0, a/2, a/2)
# a3 = (a/2, 0, a/2)
# b1 = (2*pi/a) ( 1.0,  1.0, -1.0)
# b2 = (2*pi/a) (-1.0,  1.0,  1.0)
# b3 = (2*pi/a) ( 1.0, -1.0,  1.0)
# X point
# X = (2*pi/a) ( 1.0, 0.0, 0.0 ) = 0.5 * ( b1 + b3 )
kpoint 0.5 0.0 0.5   0.0000
# L point
# L = (2*pi/a) ( 0.5, 0.5, 0.5 ) = 0.5 * ( b1 + b2 + b3 )
kpoint add 0.5 0.5 0.5   0.0000
```

RELATED INFORMATION

list_atoms

NAME

list_atoms --print a list of atoms currently defined in the sample

SYNOPSIS

list_atoms

DESCRIPTION

The list_atoms command prints a list of all atoms currently defined.

RELATED INFORMATION

list_species, atom

list_species

NAME

list_species --print a list of all species currently defined

SYNOPSIS

list_species

DESCRIPTION

The list_species command prints a list of all species currently defined. For each species, the parameters of the corresponding pseudopotential are printed.

RELATED INFORMATION

list_atoms, species

load

NAME

load --load a sample from an XML sample document

SYNOPSIS

load *URI*

DESCRIPTION

The load command loads a simulation sample defined in an XML document provided by the *URI* argument. The *URI* can be a local file, in which case Qbox will open and read the file. If *URI* is a *URL* (e.g. http://www.quantum-simulation.org/examples/samples/ch4.xml) Qbox will download the document from the corresponding web site.

Qbox sample documents must conform to the XML Schema definition provided at

http://www.quantum-simulation.org

RELATED INFORMATION

save

move

NAME

move --change the position of an atom

SYNOPSIS

move *atom* to *x y z*

move *atom* by *dx dy dz*

DESCRIPTION

The move command moves an atom to an absolute position specified by *x*, *y*, *z* or by a relative displacement specified by *dx, dy, dz*. Positions or displacements must be given in atomic units (Bohr)

RELATED INFORMATION

print

NAME

print --print the current value of a Qbox variable

SYNOPSIS

print *variable*

DESCRIPTION

The print command prints the current value a Qbox variable. For a list of variables, see the section "Qbox variables".

RELATED INFORMATION

set

quit

NAME

quit --exit Qbox

SYNOPSIS

quit

DESCRIPTION

The quit command exits Qbox without saving any information about the sample. To save the current sample, see the save command.

RELATED INFORMATION

save

randomize_wf

NAME

randomize_wf --add a random perturbation to electronic wavefunctions

SYNOPSIS

randomize_wf *[amplitude]*

DESCRIPTION

The randomize_wf command adds random numbers to the Fourier coefficients of the electronic wavefunction. The *amplitude* argument can be used to change the intensity of the perturbation.

The randomize_wf command is used at the beginning of an electronic structure calculation when the symmetry of the atomic coordinates is high. In such situations, the iterative algorithms used to compute the electronic ground state can converge to saddle points of the energy functional instead of true minima. Using randomize_wf introduces a slight symmetry breaking which is sufficient to avoid high-symmetry saddle points.

RELATED INFORMATION

reset_vcm

NAME

reset_vcm --reset the velocity of the center of mass to zero

SYNOPSIS

reset_vcm

DESCRIPTION

The reset_vcm command modifies the velocity of all atoms so as to ensure that the velocity of the center of mass is zero. The current value of the velocity of the center of mass is printed by the status command

RELATED INFORMATION

run

NAME

run --update electronic wavefunctions and/or atomic positions

SYNOPSIS

run *niter*

run *niter nitscf*

run *niter nitscf nite*

DESCRIPTION

The run command starts a simulation in which atomic positions and/or electronic states are updated. The algorithms used to update the atomic positions and electronic states are determined by the variables atoms_dyn and wf_dyn. The parameters are defined as follows

- *niter* The number of ionic steps to be performed, i.e. steps during which atomic positions are updated. This number can be zero if only electronic wavefunction updates are desired.

- *nitscf* The number of self-consistent electronic iterations. The charge density is updated at the beginning of each self-consistent iteration.
- *nite* The number of electronic iterations performed between updates of the charge density

The run command can be used in the following ways

- run *niter*

  Perform *niter* ionic steps. Before each ionic step, the electronic states are updated once, and the charge density is updated (i.e. both parameters *nitscf* and *nite* default to 1). Using “run 0” computes the total energy without modifying the electronic wavefunction.

- run *niter nitscf*

  Perform *niter* ionic steps. Before each ionic step, the charge density is updated *nitscf* times. Before each update of the charge density, the electronic states are updated once (i.e. the parameter *nite* defaults to 1.

- run *niter nitscf nite*

  Perform *niter* ionic steps. Before each ionic step, the charge density is updated *nitscf* times. Before each charge density update, the electronic states are updated *nite* times.

Example 1

```
run 0 100
```

is used to perform 100 self-consistent electronic optimization steps. Only electronic optimization is performed. This the most common way of computing the electronic ground state.

Example 2

```
run 50 10
```

is used to perform 50 ionic steps, with 10 charge density updates before each ionic step. Before each charge density update, the electronic states are updated once.

Example 3

```
run 50 5 10
```

is used to performed 50 ionic steps. The charge density is updated 5 times before each ionic step. The electronic states are updated 10 times before each charge density update.

If the variable cell_dyn is not set to LOCKED, the unit cell parameters are updated each time the atomic positions are updated. In that case, the stress variable must be set to ON so that the stress tensor is computed before the cell parameters are updated.

Atomic positions are updated before the charge density and electronic states are updated. As a consequence, the atomic positions and the electronic structure are consistent at the end of a run command.

RELATED INFORMATION

atoms_dyn, wf_dyn

save

NAME

save --save the current sample into a file

SYNOPSIS

save *filename*

DESCRIPTION

The save command saves the current sample into a file in XML format. The format used conforms to the XML Schema defined at http://www.quantum-simulation.org. The information saved includes the dimensions of the unit cell, atomic positions and velocities, the electronic wavefunction, and optionally the time derivative of the wavefunction.

RELATED INFORMATION

load

set

NAME

set --assign a value to a Qbox variable

SYNOPSIS

set *variable value [value ...]*

DESCRIPTION

The set command assigns the value(s) *value* to the variable *variable*. Some variables (e.g. cell) are multivalued, in which case the set command requires multiple arguments.

RELATED INFORMATION

print

species

NAME

species --define a new atomic species and add it to the list of currently known species

SYNOPSIS

species *name URI*

DESCRIPTION

The species command defines a new atomic species under the name *name*. The newly defined species is added to the list of currently known species. The definition is read from the given *URI*. If the *URI* is a local file, Qbox opens and reads it. If the *URI* is a *URL* (e.g.

http://www.quantum-simulation.org/examples/species/hydrogen_pbe.xml) Qbox downloads the species definition from the corresponding web site. The species definition document must conform to the species XML Schema definition given at http://www.quantum-simulation.org

RELATED INFORMATION

list_species, atom

status

NAME

status --print status of the current sample

SYNOPSIS

status

DESCRIPTION

The status command prints a brief summary of the characteristics of the current sample.

RELATED INFORMATION

strain

NAME

strain –apply strain on the system

SYNOPSIS

strain [-atomsonly] [-inverse] uxx uyy uzz uxy uyz uxz

DESCRIPTION

The strain command modifies the shape of the unit cell and modifies the atomic positions to impose a strain defined by the component of the symmetric strain tensor u. Using the -inverse flag causes the inverse transformation to be applied. Using -atomsonly changes the postitions of atoms without affecting the unit cell

RELATED INFORMATION

torsion

NAME

torsion --print the value of the torsion angle defined by the positions of four atoms

SYNOPSIS

torsion *atom1 atom2 atom3 atom4*

DESCRIPTION

The torsion command prints the value of the angle (dihedral) defined by the four atoms given as arguments. The names *atom1*, *atom2 atom3* and *atom4* must refer to the names of atoms currently defined in the sample.

RELATED INFORMATION

list_atoms, angle, distance

! (shell escape)

NAME

! (shell escape) --execute a Unix command from withing Qbox

SYNOPSIS

! *command [arguments]*

DESCRIPTION

The ! command executes the command given as an argument in a Unix shell.

RELATED INFORMATION

6 Qbox variables

atoms_dyn

NAME

atoms_dyn --atom dynamics control variable

DESCRIPTION

The atoms_dyn variable determines the algorithm used to update atomic positions during a simulation. The following values are allowed:

- LOCKED: Ionic forces are not computed and atomic positions are not updated. This is the default.
- SD: Steepest Descent. Ionic forces are computed and atomic positions are updated using the steepest descent algorithm

  $$\vec{R}(t+dt)=\vec{R}(t)+\frac{dt^2}{m}\vec{F}(t)$$

  where F is the force, m is the ionic mass and dt is the time step (see variable dt).
- SDA: Steepest Descent with Acceleration. A line minimization algorithm is used to find a minimum satisfying the Wolfe conditions before changing the descent direction. Using this algorithm requires a full calculation of the electronic ground state at each ionic step, i.e. *nitscf* > 1 in the run command.
- CG: Conjugate Gradient. A line minimization algorithm is used to find a minimum satisfying the Wolfe conditions before changing the descent direction. The Polak-Ribiere formula is used to update descent directions. Using this algorithm requires a full calculation of the electronic ground state at each ionic step, i.e. *nitscf* > 1 in the run command. Using this algorithm requires a full calculation of the electronic ground state at each ionic step, i.e. *nitscf* > 1 in the run command.
- MD: Molecular dynamics. Ionic forces are computed and ionic positions are updated using the Verlet algorithm

  $$\vec{R}(t+dt)=2\vec{R}(t)-\vec{R}(t-dt)+\frac{dt^2}{m}\vec{F}(t)$$

  This algorithm can be used to perform either Born-Oppenheimer molecular dynamics (using *nitscf* > 1 in the run command) or Car-Parrinello molecular dynamics (*nitscf*=1). See also the wf_dyn variable.

ALLOWED VALUES

LOCKED, SD, SDA, CG, MD

RELATED INFORMATION

dt

cell

NAME

cell --unit cell parameters

DESCRIPTION

The cell variable contains the coordinates of the three lattice vectors $\vec{a}_{1,} \vec{a}_{2,} \vec{a}_3$ defining the unit cell. The unit cell is defined using the set command with the following arguments

set cell a_{1x} a_{1y} a_{1z} a_{2x} a_{2y} a_{2z} a_{3x} a_{3y} a_{3z}

The lattice vectors $\vec{a}_{1,} \vec{a}_{2,} \vec{a}_3$ form the columns of the 3x3 lattice parameter matrix *A*

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{32} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

The default value of all cell parameters is zero.

ALLOWED VALUES

positive real numbers

RELATED INFORMATION

ref_cell

cell_dyn

NAME

cell_dyn --cell dynamics control variable

DESCRIPTION

The cell_dyn variable determines the algorithm used to update the unit cell during a simulation. The following values are allowed:

- LOCKED: The unit cell is not updated. This is the default.
- SD: Steepest Descent. The unit cell is updated using the computed stress tensor and the steepest descent algorithm

$$a_{ij}(t+dt) = a_{ij}(t) + \frac{dt^2}{m_\Omega} \Omega \left(\sigma(t) - \sigma^{ext}(t) \right) A^{-T}(t)$$

where $\sigma(t)$ is the stress tensor, $\sigma^{ext}(t)$ is the externally applied stress (variable ext_stress), Ω is the volume of the unit cell, m_Ω is the mass of the unit cell (variable cell_mass), *A* is the 3x3 lattice parameter matrix (see cell variable) and *dt* is the time step (variable dt).

ALLOWED VALUES

LOCKED, SD

RELATED INFORMATION

cell, ref_cell, cell_lock

cell_lock

NAME

cell_lock --cell dynamics constraints control variable

DESCRIPTION

The cell_lock variable is used to restrict the possible changes of the unit cell parameters. The allowed values are

- OFF: No restriction on the unit cell parameters are enforced. This is the default.
- A: The lattice vector $\vec{a}_1$ is fixed.
- B: The lattice vector $\vec{a}_2$ is fixed.
- C: The lattice vector $\vec{a}_3$ is fixed.
- AB: The lattice vectors $\vec{a}_1$ and $\vec{a}_2$ are fixed.
- AC: The lattice vectors $\vec{a}_1$ and $\vec{a}_3$ are fixed.
- BC: The lattice vectors $\vec{a}_2$ and $\vec{a}_3$ are fixed.
- ABC: All lattice vectors are fixed. This is equivalent to cell_dyn = LOCKED.
- S: The shape of the unit cell is preserved. Lattice vectors can change length but not direction.
- AS: The lattice vector $\vec{a}_1$ is fixed and the shape of the unit cell is preserved.
- BS: The lattice vector $\vec{a}_2$ is fixed and the shape of the unit cell is preserved.
- CS: The lattice vector $\vec{a}_3$ is fixed and the shape of the unit cell is preserved.
- ABS: The lattice vectors $\vec{a}_1$ and $\vec{a}_2$ are fixed and the shape of the unit cell is preserved.
- ACS: The lattice vectors $\vec{a}_1$ and $\vec{a}_3$ are fixed and the shape of the unit cell is preserved.
- BCS: The lattice vectors $\vec{a}_2$ and $\vec{a}_3$ are fixed and the shape of the unit cell is preserved.
- R: The aspect ratio of the unit cell is preserved. All lattice vectors are rescaled by the same constant.

ALLOWED VALUES

OFF, A, B, C, AB, AC, BC, ABC, S, AS, BS, CS, ABS, ACS, BCS, R

RELATED INFORMATION

cell, cell_dyn, cell_mass

cell_mass

NAME

cell_mass --mass of the unit cell

DESCRIPTION

The cell_mass variable is the mass of the unit cell in atomic units (in which the mass of a carbon atom is 12.0). The default value is 10000.

ALLOWED VALUES

positive real values

RELATED INFORMATION

cell, cell_dyn

charge_mix_coeff

NAME

charge_mix_coeff --charge density mixing coefficient

DESCRIPTION

The charge density mixing coefficient determines the mixing scheme used to update the electronic charge density during self-consistent iterations. The new charge density is computed from the current and old charge density according to

$$\rho_{new} = (1-\alpha)\,\rho_{old} + \alpha\,\rho_{current}$$

where α is the charge mixing coefficient. The default value is 0.5. Values smaller than one can be used to accelerate the convergence of self-consistent iteration in metallic systems.

ALLOWED VALUES

real values in the interval [0,1]

RELATED INFORMATION

charge_mix_rcut

charge_mix_rcut

NAME

charge_mix_rcut --charge mixing cutoff radius

DESCRIPTION

The charge_mix_rcut variable is used to define the range of the Coulomb interaction in the Kerker charge density mixing scheme. If the value of charge_mix_rcut is non-zero and if self-consistent iterations are used, the new charge density is computed from the current and old charge density according to

$$\rho_{new} = \rho_{old} + \delta\rho$$

$$\delta\rho(\boldsymbol{G}) = \alpha \frac{\boldsymbol{G}^2}{\boldsymbol{G}^2 + q_0^2} \left(\rho_{current}(\boldsymbol{G}) - \rho_{old}(\boldsymbol{G})\right)$$

where $q_0 = 2\pi / r_c$ and r_c is the charge mixing cutoff radius. The default value of charge_mix_cutoff is 10 a.u.

ALLOWED VALUES

positive real numbers

RELATED INFORMATION

charge_mix_coeff

debug

NAME

debug --debug parameters

DESCRIPTION

The debug variable is used to pass debug parameters to Qbox. It is not intended for normal use.

ALLOWED VALUES

character strings

RELATED INFORMATION

dt

NAME

dt --simulation time step

DESCRIPTION

The dt variable is the simulation time step in atomic units of time (1 a.u. of time = 0.02418885 fs). The default value is 3 a.u.

ALLOWED VALUES

non-negative real numbers

RELATED INFORMATION

atoms_dyn, wf_dyn, cell_dyn

ecut

NAME

ecut --plane-wave basis energy cutoff

DESCRIPTION

The ecut variable defines the size of the plane wave basis used to define the electronic wavefunctions. It must given in Rydberg units. The wavefunction plane wave basis consists of all plane waves having a kinetic energy smaller than E_{cut} . The charge density and the total potential are described using a larger basis set that includes all plane waves with a kinetic energy smaller than $4E_{cut}$. The default value of ecut is zero, in which case the plane wave basis contains one basis function--the plane wave of wavevector $\boldsymbol{G}=0$.

ALLOWED VALUES

non-negative real numbers

RELATED INFORMATION

ecutprec

NAME

ecutprec --preconditioning energy cutoff

DESCRIPTION

The ecutprec variable defines the energy cutoff used in the preconditioner for electronic structure optimization. Corrections to the electronic wavefunctions are preconditioned in Fourier space using a diagonal preconditioning matrix *K* whose elements are defined by

$$k_{ij} = \delta_{ij} k(G)$$

$$k(G) = \begin{cases} \dfrac{1}{2E_{cut}^{prec}} & \dfrac{1}{2}G^2 < E_{cut}^{prec} \\ \dfrac{1}{2}E & \dfrac{1}{2}E \geq E_{cut}^{prec} \end{cases}$$

The value of ecutprec must be given in Rydberg units. Preconditioning is only used if the wf_dyn variable is set to either PSD or PSDA. If ecutprec = 0, no preconditioning is used. The default value of ecutprec is zero. Effective preconditioning is achieved with ecutprec < ecut/2 or ecutprec < ecut/4. Small values of ecutprec (< 5 Ry) can lead to instabilities.

ALLOWED VALUES

positive real numbers smaller than or equal to ecut.

RELATED INFORMATION

ecut, wf_dyn

ecuts

NAME

ecuts --energy cutoff for stress confinement potential

DESCRIPTION

The ecuts variable defines the energy cutoff used in a confinement potential in Fourier space. The confinement potential is used when computing the stress tensor with variable cell size in order to ensure constant resolution as the unit cell changes size. The confinement energy is

$$E_{conf} = \frac{1}{2} \sum_{n,\boldsymbol{G}} |c_{n\boldsymbol{G}}|^2 G^2 f(G)$$

where

$$f(g) = f_s \left(1 - \frac{1}{1 + \exp((G^2/2 - E_{cut}^s)/\sigma_s)} \right)$$

$$f_s = 2 \quad \text{and} \quad \sigma_s = \frac{1}{2} \quad .$$

The default value of ecuts is zero, in which case the confinement potential is not used.

For a detailed description of the use of confinement potentials in constant pressure simulations, see

1. P. Focher, G. L. Chiarotti, M. Bernasconi, *et al*. Structural Phase-Transformations Via 1St-Principles Simulation, Europhys. Lett. **26** (5): 345-351 (1994).

2. M. Bernasconi, G. L. Chiarotti, P. Focher, *et al*. First-Principle Constant-Pressure Molecular-Dynamics, Journal Of Physics And Chemistry Of Solids **56** (3-4): 501-505 (1995).

ALLOWED VALUES

positive real numbers smaller than or equal to ecut

RELATED INFORMATION

stress, cell_dyn

emass

NAME

emass --fictitious electronic mass for Car-Parrinello simulations

DESCRIPTION

The emass variable defines the fictitious electronic mass used in Car-Parrinello simulations. The default value is zero, in which case the fictitious electronic mass used in the calculation is $m_e = 2 E_{cut} dt^2$. The value of emass is only relevant if the variable wf_dyn is set to MD (i.e. if the wavefunction dynamics is Car-Parrinello). It is ignored otherwise.

ALLOWED VALUES

positive real values

RELATED INFORMATION

wf_dyn

ext_stress

NAME

ext_stress --external stress

DESCRIPTION

The ext_stress variable determines the value of the externally applied stress. The ext_stress variable must be set using the following syntax

set ext_stress $\sigma_{xx} \sigma_{yy} \sigma_{zz} \sigma_{xy} \sigma_{yz} \sigma_{xz}$

where the values of the elements of the stress tensor must be given in GPa units. The external stress can be positive or negative. The default value is zero.

ALLOWED VALUES

real numbers

RELATED INFORMATION

stress

fermi_temp

NAME

fermi_temp --Fermi temperature for fractionally occupied states

DESCRIPTION

The fermi_temp variable determines the value of the Fermi temperature used in the calculation of occupation factors for fractionally occupied states. The value must be given in Kelvin. The default value is zero.

ALLOWED VALUES

non-negative real numbers

RELATED INFORMATION

nempty

nempty

NAME

nempty --number of empty electronic states

DESCRIPTION

The nempty variable determines the number of electronic states that are included in the calculation in addition to the number of states needed to accomodate the total number of electrons. If nempty is non-zero, the eigenvalues and eigenvectors of the Kohn-Sham hamiltonian are computed at each electronic iteration and the charge density is recomputed from the eigenvectors using a Fermi distribution. The default value is of nempty is zero.

ALLOWED VALUES

non-negative integers

RELATED INFORMATION

fermi_temp

net_charge

NAME

net_charge --net charge of the system

DESCRIPTION

The net_charge variable is used to control the total amount of electronic charge in the calculation. If net_charge = 0, the total number of electrons is determined from the sum of the valence charges given in the species definition files and the system is neutral. If net_charge = -1, an extra electron is added to the system. If net_charge = 1, an electron is removed from the system. The default value is zero.

ALLOWED VALUES

integers

RELATED INFORMATION

nrowmax

NAME

nrowmax --maximum number of process grid rows

DESCRIPTION

The nrowmax variable determines the shape of the process grid used in parallel calculations. It is used to optimize performance when large numbers of parallel tasks are used. The default value is 32.

ALLOWED VALUES

positive integers

RELATED INFORMATION

ref_cell

NAME

ref_cell --reference unit cell

DESCRIPTION

The ref_cell variable determines the size of the reference unit cell. The reference unit cell is used in constant pressure calculations to ensure constant resolution of the basis set as the unit cell changes size. The unit cell must always be enclosed in the reference unit cell during a constant pressure calculation.

ALLOWED VALUES

positive real numbers

RELATED INFORMATION

cell

stress

NAME

stress --stress calculation control variable

DESCRIPTION

The stress variable determines whether the stress tensor is calculated. The possible values are ON and OFF. The default is OFF.

ALLOWED VALUES

ON, OFF

RELATED INFORMATION

ext_stress, cell_dyn

thermostat

NAME

thermostat --thermostat control variable

DESCRIPTION

The thermostat variable determines what type of thermostat is used for constant temperature simulations. The choices are

- OFF: No thermostat is used. This is the default.
- SCALING: Scaling of velocities. At each MD step, the velocities of all atoms are rescaled as

  $$v_i = v_i(1 - \eta \, dt)$$

where

$$\eta = \frac{1}{\tau} \tanh \frac{T - T_{ref}}{\Delta}$$

τ is the thermostat time constant (variable th_time), T is the instantaneous temperature computed from the ionic kinetic energy, T_{ref} is the thermostat reference temperature (variable th_temp), and Δ is the thermostat temperature width (variable th_width).

- ANDERSEN: Andersen thermostat. The atoms are subjected to random collisions with particles drawn from a Maxwell distribution of velocities with temperature T_{ref}. The collision frequency is $1/\tau$ where τ is given by the variable th_time (see H. C. Andersen, J. Chem. Phys. **72**, 2384 (1980)).
- LOWE: Lowe thermostat. Pairs of atoms are subjected to random collisions with a particle drawn from a Maxwell distribution of velocities with temperature T_{ref}. The collision frequency is $1/\tau$ where τ is given by the variable th_time . The Lowe thermostat is similar to the Andersen thermostat but conserves total momentum (see C. P. Lowe, Europhys. Lett. **47**, 145 (1999)).

ALLOWED VALUES

OFF, SCALING, ANDERSON, LOWE

RELATED INFORMATION

th_temp, th_time, th_width,

th_temp

NAME

th_temp --thermostat reference temperature

DESCRIPTION

The th_temp variable determines the thermostat reference temperature. The default is zero. See thermostat.

ALLOWED VALUES

non-negative real numbers

RELATED INFORMATION

th_time, th_width, thermostat

th_time

NAME

th_time --thermostat time constant

DESCRIPTION

The th_time variable determines the thermostat time constant. The time constant is a measure of the time over which the thermostat adjusts the temperature. See thermostat for details. The value of th_time must be given in atomic units of time. The default value is 5000 a.u. (~ 120 fs).

ALLOWED VALUES

positive real numbers

RELATED INFORMATION

th_temp, th_width, thermostat

th_width

NAME

th_width --thermostat temperature window

DESCRIPTION

The th_width determines the value of the thermostat temperature width. See thermostat details. The value of th_width must be given in Kelvin units. The default value is 100 K.

ALLOWED VALUES

positive real numbers

RELATED INFORMATION

th_temp, th_time, thermostat

wf_diag

NAME

wf_diag --diagonalization control variable

DESCRIPTION

The wf_diag variable determines whether eigenvectors and/or eigenvalues of the hamiltonian are computed after each optimization of the electronic structure. The choices are

- T: True. Eigenvalues and eigenvectors are computed. Eigenvalues are printed on output in eV units.
- F: False. Eigenvalues and eigenvectors are not computed. This is the default.
- EIGVAL: Eigenvalues only. The eigenvalues are computed and printed, but eigenvectors are not computed.
- MLWF: Maximally localized Wannier Functions (MLWFs) are computed.
- MLWFC: The position of MLWF centers is computed and printed but MLWFs are not computed.

If empty states are added to the calculation (variable nempty > 0) the eigenvalues and eigenvectors are always computed.

ALLOWED VALUES

T, F, EIGVAL,MLWF,MLWFC

RELATED INFORMATION

wf_dyn

NAME

wf_dyn --wavefunction dynamics control variable

DESCRIPTION

The wf_dyn variable determines which algorithm is used to update the wavefunctions during electronic structure optimization. The choices are

- SD: Steepest Descent. This the default. Wavefunctions are updated as follows:

$$\psi^{(k+1)} = \psi^{(k)} - \alpha H \psi^{(k)}$$

where

$$\alpha = \begin{cases} \dfrac{1}{2E_{cut}} & m_e = 0 \\ \dfrac{dt^2}{m_e} & m_e > 0 \end{cases}$$

and m_e is the fictitious electronic mass (variable emass). By default, m_e=0.

- PSD: Preconditioned Steepest Descent. Wavefunctions are updated as follows:

$$\psi^{(k+1)} = \psi^{(k)} - \alpha K P_\perp H \psi^{(k)}$$

where K is a preconditioning matrix

$$k_{ij} = \delta_{ij} k(G)$$

$$k(G) = \begin{cases} \dfrac{1}{2E_{cut}^{prec}} & \dfrac{1}{2}G^2 < E_{cut}^{prec} \\ \dfrac{1}{2}E & \dfrac{1}{2}E \geq E_{cut}^{prec} \end{cases}$$

and $P_\perp = I - \sum_i |\psi_i\rangle\langle\psi_i|$ is a projector on the orthogonal complement of the subspace spanned by all wavefunctions. The preconditioning matrix depends on the value of E_{cut}^{prec} (variable ecutprec).

- PSDA: Preconditioned Steepest Descent with Anderson acceleration. Wavefunctions are updated as with the PSD option, and convergence is accelerated by the Anderson scheme (D. G. Anderson, JACM **12**, No 4, pp. 547-560 (1965)).
- MD: Molecular Dynamics. Wavefunctions are updated using the Car-Parrinello scheme

$$\psi_i^{(k+1)} = 2\psi_i^{(k)} - \psi_i^{(k-1)} - \frac{dt^2}{m_e} H \psi_i^{(k)} + \sum_j \Lambda_{ij} \psi_j^{(k)}$$

  where holonomic constraints are used to enforce orthogonality.
- LOCKED. Wavefunctions are not updated.

ALLOWED VALUES

SD, PSD, PSDA, MD, LOCKED

RELATED INFORMATION

other variables or commands

xc

NAME

xc --exchange-correlation functional control variable

DESCRIPTION

The xc variable determines which exchange-correlation functional is used in the electronic structure calculation. The choices are

- LDA: Local Density Approximation, Ceperley-Alder data. This is the default.
- PBE: Perdew-Burke-Ernzerhof GGA functional.

ALLOWED VALUES

LDA, PBE

RELATED INFORMATION