PyCDFT

0.5

Generated by Doxygen 1.8.3.1

Wed Oct 3 2018 13:11:45

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

Class Documentation

3.1 pycdft.common.atom.Atom Class Reference

An atom in a specific cell.

Public Attributes

- sample

  the sample where the atom lives.
- symbol

  chemical symbol of the atom.

3.1.1 Detailed Description

An atom in a specific cell.

An atom can be initialized by and exported to an ASE Atom object.
All physical quantities are in atomic unit.

```
abs_coord (np.ndarray, shape = [3]): absolute coordinate.
cry_coord (np.ndarray, shape = [3]): crystal coordinate.
```

Extra attributes are welcomed to be attached to an atom.

3.1.2 Member Data Documentation

3.1.2.1 pycdft.common.atom.Atom.sample

the sample where the atom lives.

3.1.2.2 pycdft.common.atom.Atom.symbol

chemical symbol of the atom.

The documentation for this class was generated from the following file:

- atom.py

3.2 pycdft.cdft.CDFTSolver Class Reference

Constrained DFT solver.

Public Member Functions

- def solve

  Solve CDFT SCF or optimization problem.
- def solve_scf

  Iteratively solve the CDFT problem.
- def solve_scf_with_new_V

  Given V for all constraints, solve the KS problem.
- def solve_scf_for_dW_by_dV

  Wrapper function for solve_scf_with_new_V returning dW/dV.
- def solve_opt

  Relax the structure under constraint.

Public Attributes

- job

  "scf" or "opt".
- sample

  the whole system for which CDFT calculation is performed.
- dft_driver

  the interface to DFT code (e.g.
- maxcscf

  maximum number of CDFT iterations.
- maxstep

  maximum geometry optimization steps.
- F_tol

  force threshold for optimization.

3.2.1 Detailed Description

Constrained DFT solver.

Vc_tot (float array, shape == [vspin, n1, n2, n3]): total constraint potential as a sum of all constraints defined on all fragments.

3.2.2 Member Function Documentation

3.2.2.1 def pycdft.cdft.CDFTSolver.solve (*self*)

Solve CDFT SCF or optimization problem.

3.2.2.2 def pycdft.cdft.CDFTSolver.solve_opt (*self*)

Relax the structure under constraint.

A force from constraint potential is added to DFT force during optimization.

3.2.2.3 def pycdft.cdft.CDFTSolver.solve_scf (*self*)

Iteratively solve the CDFT problem.

An outer loop (implemented below) is performed to maximize the free energy w.r.t. Lagrangian multipliers for all constrains, an inner loop (casted to a KS problem and outsourced to DFT code) is performed to minimize the free energy w.r.t. charge density.

3.2.2.4 def pycdft.cdft.CDFTSolver.solve_scf_for_dW_by_dV (*self, V*)

Wrapper function for solve_scf_with_new_V returning dW/dV.

3.2.2.5 def pycdft.cdft.CDFTSolver.solve_scf_with_new_V (*self, Vs*)

Given V for all constraints, solve the KS problem.

3.2.3 Member Data Documentation

3.2.3.1 pycdft.cdft.CDFTSolver.dft_driver

the interface to DFT code (e.g.

Qbox or PW).

3.2.3.2 pycdft.cdft.CDFTSolver.F_tol

force threshold for optimization.

3.2.3.3 pycdft.cdft.CDFTSolver.job

"scf" or "opt".

3.2.3.4 pycdft.cdft.CDFTSolver.maxcscf

maximum number of CDFT iterations.

3.2.3.5 pycdft.cdft.CDFTSolver.maxstep

maximum geometry optimization steps.

3.2.3.6 pycdft.cdft.CDFTSolver.sample

the whole system for which CDFT calculation is performed.

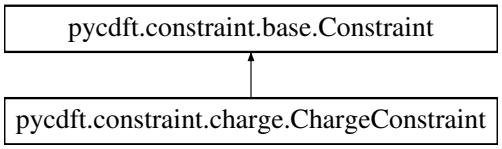
The documentation for this class was generated from the following file:

- cdft.py

3.3 pycdft.constraint.charge.ChargeConstraint Class Reference

Constraint on the absolute electron number of a fragment.

Inheritance diagram for pycdft.constraint.charge.ChargeConstraint:



Additional Inherited Members

3.3.1 Detailed Description

Constraint on the absolute electron number of a fragment.

Extra attributes: fragment a fragment of the whole system.

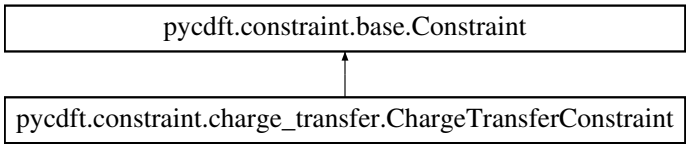
The documentation for this class was generated from the following file:

- charge.py

3.4 pycdft.constraint.charge_transfer.ChargeTransferConstraint Class Reference

Constraint on electron number difference between a donor and an acceptor fragment.

Inheritance diagram for pycdft.constraint.charge_transfer.ChargeTransferConstraint:



Additional Inherited Members

3.4.1 Detailed Description

Constraint on electron number difference between a donor and an acceptor fragment.

Extra attributes: donor donor fragment. acceptor acceptor fragment.

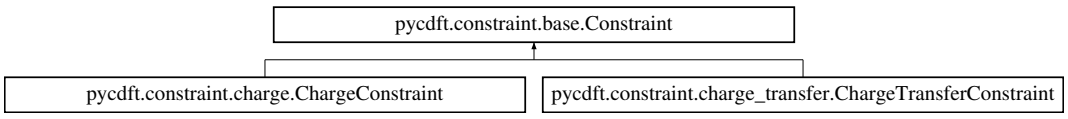
The documentation for this class was generated from the following file:

- charge_transfer.py

3.5 pycdft.constraint.base.Constraint Class Reference

Constraint.

Inheritance diagram for pycdft.constraint.base.Constraint:



Public Member Functions

- def __init__
- def dW_by_dV

  The derivative of free energy with respect to V.
- def update_structure

  Update the constraint with new structure.
- def update_w

  Update the weight with new structure.
- def update_N

  Update the electron number or electron number difference.
- def update_Vc

  Update constraint potential.
- def update_Fc

  Update constraint force.

Public Attributes

- sample

  the whole system.
- N0

  the target value for the electron number or electron number difference,
- V_init

  Initial guess or bracket for V, used for certain optimization algorithms.
- N_tol

  convergence threshold for N - N0 (= dW/dV).
- V

  Lagrangian multiplier associate with the constraint.
- N

  the electron number or electron number difference for this constraint

3.5.1 Detailed Description

Constraint.

3.5.2 Constructor & Destructor Documentation

3.5.2.1 def pycdft.constraint.base.Constraint.__init__ (*self, sample*)

Parameters

V_init	initial guess for V.

3.5.3 Member Function Documentation

3.5.3.1 def pycdft.constraint.base.Constraint.dW_by_dV (*self*)

The derivative of free energy with respect to V.

dW/dV = dr w_i(r) n(r) - N0 = N - N0

3.5.3.2 def pycdft.constraint.base.Constraint.update_Fc (*self*)

Update constraint force.

3.5.3.3 def pycdft.constraint.base.Constraint.update_N (*self*)

Update the electron number or electron number difference.

3.5.3.4 def pycdft.constraint.base.Constraint.update_structure (*self*)

Update the constraint with new structure.

3.5.3.5 def pycdft.constraint.base.Constraint.update_Vc (*self*)

Update constraint potential.

3.5.3.6 def pycdft.constraint.base.Constraint.update_w (*self*)

Update the weight with new structure.

3.5.4 Member Data Documentation

3.5.4.1 pycdft.constraint.base.Constraint.N_tol

convergence threshold for N - N0 (= dW/dV).

3.5.4.2 pycdft.constraint.base.Constraint.sample

the whole system.

3.5.4.3 pycdft.constraint.base.Constraint.V

Lagrangian multiplier associate with the constraint.

3.5.4.4 pycdft.constraint.base.Constraint.V_init

Initial guess or bracket for V, used for certain optimization algorithms.

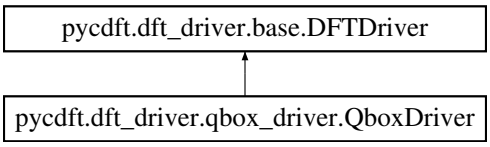
The documentation for this class was generated from the following file:

- constraint/base.py

3.6 pycdft.dft_driver.base.DFTDriver Class Reference

DFT driver.

Inheritance diagram for pycdft.dft_driver.base.DFTDriver:



Public Member Functions

- def reset

  Reset the DFT code.
- def set_Vc

  Set the constraint potential Vc in DFT code.
- def run_scf

  Order the DFT code to perform SCF calculation under the constraint.
- def run_opt

  Order the DFT code to run one structure relaxation step.
- def get_rho_r

  Fetch the charge density from the DFT code, write to self.sample.rhor.
- def get_force

  Fetch the DFT force from the DFT code.
- def set_Fc

  Set the constraint force in the DFT code.
- def get_structure

  Fetch the structure from the DFT code, write to self.sample.
- def get_wfc

  Fetch the wavefunction from the DFT code.

Public Attributes

- sample

  the whole system for which CDFT calculation is performed.
- istep

  current geometry optimization step.
- icscf

  current constrained SCF step.
- output_path

  output file path.

3.6.1 Detailed Description

DFT driver.

3.6.2 Member Function Documentation

3.6.2.1 def pycdft.dft_driver.base.DFTDriver.get_force (*self*)

Fetch the DFT force from the DFT code.

3.6.2.2 def pycdft.dft_driver.base.DFTDriver.get_rho_r (*self*)

Fetch the charge density from the DFT code, write to self.sample.rhor.

3.6.2.3 def pycdft.dft_driver.base.DFTDriver.get_structure (*self*)

Fetch the structure from the DFT code, write to self.sample.

3.6.2.4 def pycdft.dft_driver.base.DFTDriver.get_wfc (*self*)

Fetch the wavefunction from the DFT code.

3.6.2.5 def pycdft.dft_driver.base.DFTDriver.reset (*self, output_path*)

Reset the DFT code.

3.6.2.6 def pycdft.dft_driver.base.DFTDriver.run_opt (*self*)

Order the DFT code to run one structure relaxation step.

3.6.2.7 def pycdft.dft_driver.base.DFTDriver.run_scf (*self*)

Order the DFT code to perform SCF calculation under the constraint.

Returns when SCF calculation is finished.

3.6.2.8 def pycdft.dft_driver.base.DFTDriver.set_Fc (*self*)

Set the constraint force in the DFT code.

3.6.2.9 def pycdft.dft_driver.base.DFTDriver.set_Vc (*self, Vc*)

Set the constraint potential Vc in DFT code.

```
Given constraint potential Vc as an array of shape [vspin, n1, n2, n3],
this method send the constraint potential to the DFT code.
```

Parameters

Vc	the constraint potential.

3.6.3 Member Data Documentation

3.6.3.1 pycdft.dft_driver.base.DFTDriver.icscf

current constrained SCF step.

3.6.3.2 pycdft.dft_driver.base.DFTDriver.istep

current geometry optimization step.

3.6.3.3 pycdft.dft_driver.base.DFTDriver.output_path

output file path.

3.6.3.4 pycdft.dft_driver.base.DFTDriver.sample

the whole system for which CDFT calculation is performed.

The documentation for this class was generated from the following file:

- dft_driver/base.py

3.7 pycdft.common.fragment.Fragment Class Reference

A part of the system to which constraints may apply.

Public Attributes

- sample

  sample.

3.7.1 Detailed Description

A part of the system to which constraints may apply.

atoms (list of Atom): list of atoms belonging to the fragment. w (numpy array, shape = [n1, n2, n3]: Hirshfeld weight function.

3.7.2 Member Data Documentation

3.7.2.1 pycdft.common.fragment.Fragment.sample

sample.

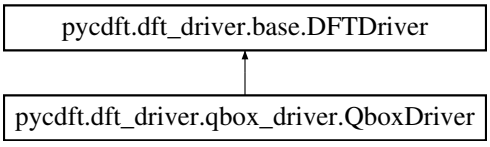
The documentation for this class was generated from the following file:

- fragment.py

3.8 pycdft.dft_driver.qbox_driver.QboxDriver Class Reference

DFT driver.

Inheritance diagram for pycdft.dft_driver.qbox_driver.QboxDriver:



Public Member Functions

- def wait_for_lock_file

  Wait for Qbox lock file to appear.
- def run_cmd

  Order Qbox to run given command.
- def set_Vc

  Write Vc in cube format, then send set vext command to Qbox.
- def copy_output

  Copy Qbox output file to self.output_path.
- def run_scf

  Run SCF calculation in Qbox.
- def run_opt

  Run geometry optimization in Qbox.
- def get_rho_r

  Implement abstract fetch_rhor method for Qbox.
- def get_force

  Implement abstract fetch_force method for Qbox.
- def set_Fc

  Implement abstract set_force method for Qbox.
- def get_structure

  Implement abstract fetch_structure method for Qbox.
- def clean

  Clean qb_cdft.in qb_cdft.out qb_cdft.in.lock.
- def get_wfc

  Parse wavefunction from Qbox.

Additional Inherited Members

3.8.1 Detailed Description

DFT driver.

Extra attributes: init_cmd initialization command for Qbox. scf_cmd command for running constrained SCF. opt_cmd command for running geometry optimization.

3.8.2 Member Function Documentation

3.8.2.1 def pycdft.dft_driver.qbox_driver.QboxDriver.copy_output (*self*)

Copy Qbox output file to self.output_path.

3.8.2.2 def pycdft.dft_driver.qbox_driver.QboxDriver.get_force (*self*)

Implement abstract fetch_force method for Qbox.

3.8.2.3 def pycdft.dft_driver.qbox_driver.QboxDriver.get_rho_r (*self*)

Implement abstract fetch_rhor method for Qbox.

Send plot charge density commands to Qbox, then parse charge density.

3.8.2.4 def pycdft.dft_driver.qbox_driver.QboxDriver.get_structure (*self*)

Implement abstract fetch_structure method for Qbox.

3.8.2.5 def pycdft.dft_driver.qbox_driver.QboxDriver.get_wfc (*self*)

Parse wavefunction from Qbox.

3.8.2.6 def pycdft.dft_driver.qbox_driver.QboxDriver.run_cmd (*self, cmd*)

Order Qbox to run given command.

3.8.2.7 def pycdft.dft_driver.qbox_driver.QboxDriver.run_opt (*self*)

Run geometry optimization in Qbox.

3.8.2.8 def pycdft.dft_driver.qbox_driver.QboxDriver.run_scf (*self*)

Run SCF calculation in Qbox.

3.8.2.9 def pycdft.dft_driver.qbox_driver.QboxDriver.set_Fc (*self*)

Implement abstract set_force method for Qbox.

3.8.2.10 def pycdft.dft_driver.qbox_driver.QboxDriver.set_Vc (*self, Vc*)

Write Vc in cube format, then send set vext command to Qbox.

3.8.2.11 def pycdft.dft_driver.qbox_driver.QboxDriver.wait_for_lock_file (*self*)

Wait for Qbox lock file to appear.

The documentation for this class was generated from the following file:

- qbox_driver.py

3.9 pycdft.common.sample.Sample Class Reference

The physical system to be simulated.

Public Member Functions

- def update_weights

  Update weights with new structure.
- def compute_eigr

  Compute e^{-iGr} array where r is coordinate of atom.
- def compute_rhoatom_g

  Compute charge density for an atom with specific coordinate in cell.
- def compute_rhoatom_grad_r

Compute nuclear gradient for atom.

- def ase_cell

  Get an ASE Atoms object of current cell.
- def show

  Visualize the structure by VESTA.
- def save

  Save the structure to file.
- def export

  Export the structure to various formats.
- def nel

  Compute # of electrons according to certain pseudopotential family.

Public Attributes

- omega

  cell volume.
- vspin

  number of spin channels (1 or 2) for constraint potential.
- Ed

  DFT energy.
- Ec

  Constraint energy.
- W

  free energy.
- Fd

  DFT force.
- Fc

  Constraint force.
- Fw

  -grad(W).

3.9.1 Detailed Description

The physical system to be simulated.

All physical quantities are in atomic unit.

Variables ending with _r are defined on G space grid. Variables ending with _g are defined on G space grid. Variables ending with _rd are defined on radial grid in R space; variables ending with _d correspond to all G vectors with different norm. They are used to compute atomic densities.

```
R (np.ndarray, shape = [3, 3]): the real space lattice vectors of the system.
G (np.ndarray, shape = [3, 3]): the reciprocal space lattice vectors of the system.
```

3.9.2 Member Function Documentation

3.9.2.1 def pycdft.common.sample.Sample.ase_cell (*self*)

Get an ASE Atoms object of current cell.

3.9.2.2 def pycdft.common.sample.Sample.compute_eigr (*self, atom*)

Compute e^{-iGr} array where r is coordinate of atom.

3.9.2.3 def pycdft.common.sample.Sample.compute_rhoatom_g (*self, atom*)

Compute charge density for an atom with specific coordinate in cell.

3.9.2.4 def pycdft.common.sample.Sample.compute_rhoatom_grad_r (*self, atom*)

Compute nuclear gradient for atom.

3.9.2.5 def pycdft.common.sample.Sample.export (*self, fmt =* "qb", *pseudos =* None)

Export the structure to various formats.

3.9.2.6 def pycdft.common.sample.Sample.nel (*self, pseudos =* "SG15")

Compute # of electrons according to certain pseudopotential family.

3.9.2.7 def pycdft.common.sample.Sample.save (*self, fname*)

Save the structure to file.

3.9.2.8 def pycdft.common.sample.Sample.show (*self*)

Visualize the structure by VESTA.

3.9.2.9 def pycdft.common.sample.Sample.update_weights (*self*)

Update weights with new structure.

3.9.3 Member Data Documentation

3.9.3.1 pycdft.common.sample.Sample.Ec

Constraint energy.

3.9.3.2 pycdft.common.sample.Sample.Ed

DFT energy.

3.9.3.3 pycdft.common.sample.Sample.Fc

Constraint force.

Fc = sum_k V_k int grad(w_k(r)) n(r) dr

3.9.3.4 pycdft.common.sample.Sample.Fd

DFT force.

3.9.3.5 pycdft.common.sample.Sample.Fw

-grad(W).

Fw = Fd + Fc.

3.9.3.6 pycdft.common.sample.Sample.omega

cell volume.

3.9.3.7 pycdft.common.sample.Sample.vspin

number of spin channels (1 or 2) for constraint potential.

Note that

3.9.3.8 pycdft.common.sample.Sample.W

free energy.

W = Ed + Ec - sum_k V_k N_k

The documentation for this class was generated from the following file:

- sample.py

3.10 pycdft.common.wfc.Wavefunction Class Reference

Container class for Kohn-Sham wavefunction.

Public Member Functions

- def skb2idx

  Get internal index from (spin, kpoint, band) index.
- def idx2skb

  Get (spin, kpoint, band) index from internal index.
- def normalize

  Normalize psi(r).

3.10.1 Detailed Description

Container class for Kohn-Sham wavefunction.

A wavefunction is defined as a collection of KS orbitals, each uniquely labeled by three spin (0 or 1), k point index and band index. To facilitate distributed storage and access of a wavefunction on multiple processors, each KS orbital is also uniquelly labeled by an internal index. Internal index (idx) is generated by following pattern: for ispin in range(nspin): for ikpt in range(nkpt): for ibnd in range(nbnd[ispin, ikpt]): idx ++

Currently, k points are not fully supported.

Public attributes: psi_r R space KS orbitals defined on a R space grid described by self.wgrid. psi_g G space KS orbitals defined on a G space grid described by self.wgrid.

```
Above quantities can be accessed like dicts. They can be indexed with either
an integer (internal index) or a 3-tuple of integers (spin, kpoint, band index).
After been indexed, the corresponding quantity (numpy array) of a
specific KS orbital is returned.
```

sample sample upon which the wavefunction is defined. wgrid wavefunction grid. dgrid charge density grid.

nspin # of spin channel. 1: spin unpolarized; 2: spin polarized. nkpt # of k points. nbnd # of bands. norb total # of orbitals on all spins, kpoints. occ occupation numbers. shape: (nspin, nkpt, nbnd).

Private attributes: idx_skb_map internal index -> (spin, kpoint, band) index map skb_idx_map (spin, kpoint, band) index -> internal index map

```
Above maps can be accessed by skb2idx and idx2skb methods.
```

3.10.2 Member Function Documentation

3.10.2.1 def pycdft.common.wfc.Wavefunction.idx2skb (*self, idx*)

Get (spin, kpoint, band) index from internal index.

3.10.2.2 def pycdft.common.wfc.Wavefunction.normalize (*self, psir*)

Normalize psi(r).

3.10.2.3 def pycdft.common.wfc.Wavefunction.skb2idx (*self, ispin, ikpt, ibnd*)

Get internal index from (spin, kpoint, band) index.

The documentation for this class was generated from the following file:

- wfc.py

3.11 pycdft.common.wfc.WfcManager Class Reference

Helper class to manage a collection of quantities like psi(r) or psi(G).

3.11.1 Detailed Description

Helper class to manage a collection of quantities like psi(r) or psi(G).

The collection can be indexed by either an internal index or a (spin, kpoint, band) index.

The documentation for this class was generated from the following file:

- wfc.py

Index