Module **parameterize**

QMMMReBind : Quantum Mechanics – Molecular Mechanics (*QMMM*) forcefield *Re*paramaterisation of the *Bind*ing site for the receptor-ligand complexes

Functions

```
def OPLS_LJ(system)
```

```
def copy_file(source, destination)
```

```
def dihedral_energy(x, k1, k2, k3, k4=0)
```

```
def dot_product(u_PA, eig_AB)
```

```
def error_function(delta_qm, delta_mm)
```

```
def error_function_boltzmann(delta_qm, delta_mm, T)
```

```
def fit_params(qm_scan_file, load_topology, system_xml, method)
```

```
def force_angle_constant(atom_A, atom_B, atom_C, bond_lengths, eigenvalues,
                         eigenvectors, coords, scaling_1, scaling_2)
```

```
def force_angle_constant_special_case(atom_A, atom_B, atom_C, bond_lengths, eigenvalues,
                                      eigenvectors, coords, scaling_1, scaling_2)
```

```
def force_constant_bond(atom_A, atom_B, eigenvalues, eigenvectors, coords)
```

```
def gen_init_guess(qm_scan_file, load_topology, system_xml)
```

```
def generate_mm_pdbs(qm_scan_file, template_pdb)
```

```
def generate_xml_from_charged_pdb_sdf(system_pdb, system_init_sdf, system_sdf,
                                      num_charge_atoms, index_charge_atom_1,
                                      charge_atom_1, system_xml)
```

This function generates an openforcefield xml file from the pdb file via SDF file and openforcefield.

```
def generate_xml_from_pdb_sdf(system_pdb, system_sdf, system_xml)
```

This function generates an openforcefield xml file from the pdb file

```
def get_dihedrals(qm_scan_file)
```

```
def get_mm_potential_energies(qm_scan_file, load_topology, system_xml)
```

```
def get_non_torsion_mm_energy(system_pdb, load_topology, system_xml)
```

```
def get_qm_energies(qm_scan_file)
```

```
def get_tor_params(qm_scan_file, template_pdb, load_topology, system_xml, method)
```

```
def get_torsional_lines(template_pdb, system_xml, qm_scan_file, load_topology, method,
                        dihedral_text_file)
```

```
def list_diff(list_1, list_2)
```

```
def list_hartree_kcal(list_)
```

```
def list_kJ_kcal(list_)
```

```
def list_to_dict(lst)
```

```
def objective_function(k_array, x, delta_qm)
```

```
def remove_mm_files(qm_scan_file)
```

```
def reverse_list(lst)
```

```
def scale_list(list_)
```

```
def search_in_file(file: str, word: str) -> list
```

Search for the given string in file and return lines containing that string along with line numbers

```
def torsiondrive_input_to_xyz(psi_input_file, xyz_file)
def u_PA_from_angles(atom_A, atom_B, atom_C, coords)
```

```
def uniq(input_)
```

```
def unit_vector_N(u_BC, u_AB)
```

```
def xyz_to_pdb(xyz_file, coords_file, template_pdb, system_pdb)
```

Classes

```
class GuestAmberXMLAmber (charge, num_charge_atoms, charge_atom_1, index_charge_atom_1,
                          system_pdb='guest_init_II.pdb', system_mol2='guest.mol2',
                          system_in='guest.in', system_frcmod='guest.frcmod',
                          prmtop_system='guest.prmtop', inpcrd_system='guest.inpcrd',
                          system_leap='guest.leap', system_xml='guest_init.xml',
                          system_smi='guest.smi', system_sdf='guest.sdf',
                          system_init_sdf='guest_init.sdf', index_charge_atom_2=' ',
                          charge_atom_2=' ', charge_parameter_file='guest_charges.txt',
                          system_qm_pdb='guest_init_II.pdb',
                          bond_parameter_file='guest_bonds.txt',
                          angle_parameter_file='guest_angles.txt',
                          system_qm_params_file='guest_qm_params.txt',
                          reparameterised_intermediate_system_xml_file='guest_intermedia
                          te_reparameterised.xml',
                          system_xml_non_bonded_file='guest_xml_non_bonded.txt',
                          system_xml_non_bonded_reparams_file='guest_xml_non_bonded_repa
                          rams.txt',
                          reparameterised_system_xml_file='guest_reparameterised.xml',
                          non_reparameterised_system_xml_file='guest_init.xml',
                          prmtop_system_non_params='guest_non_params.prmtop',
                          inpcrd_system_non_params='guest_non_params.inpcrd',
                          prmtop_system_params='guest_params.prmtop',
                          inpcrd_system_params='guest_params.inpcrd',
                          load_topology='openmm')
```

A class used to generate a template force field XML file for the ligand in order regenerate the reparametrised forcefield XML file.

This class contains methods to generate a template XML force field through openforcefield. XML template generation can be obtained through different file formats such as PDB, SDF, and SMI. Methods support charged ligands as well. Reparameterized XML force field files are then generated from the template files. Different energy components such as the bond, angle, torsional and non-bonded energies are computed for the non-reparametrized and the reparameterized force fields. Difference between the non-reparameterized and reparameterized force field energies can then be analyzed. …

Attributes

charge : `int`
Charge of the ligand.

num_charge_atoms : `int`
Number of charged atoms in the molecule.

charge_atom_1 : `int`
Charge on the first charged atom.

index_charge_atom_1 : `int`
Index of the first charged atom.

system_pdb : `str` , optional
Ligand PDB file with atom numbers beginning from 1.

system_mol2 : `str` , optional
Ligand Mol2 file obtained from PDB file.

system_in : `str` , optional
Prepi file as required by antechamber.

system_frcmod : `str` , optional
FRCMOD file as required by antechamber.

prmtop_system : `str` , optional
Topology file obtained from the ligand PDB.

inpcrd_system : `str` , optional
Coordinate file obtained from the Ligand PDB using the command saveamberparm.

system_leap : `str` , optional
Amber generated leap file for generating and saving topology and coordinate files.

system_xml : `str` , optional
Serialized XML force field file of the ligand.

system_smi : `str` , optional

Ligand SMILES format file.

system_sdf : `str` , optional
Ligand SDF (structure-data) format file.

system_init_sdf : `str` , optional
Ligand SDF (structure-data) format file. This file will be generated only if the ligand is charged.

index_charge_atom_2 : `int` , optional
Index of the second charged atom of the ligand.

charge_atom_2 : `int` , optional
Charge on the second charged atom of the ligand.

charge_parameter_file : `str` , optional
File containing the charges of ligand atoms and their corresponding atoms.

system_qm_pdb : `str` , optional
Ligand PDB file with atom numbers beginning from 1.

bond_parameter_file : `str` , optional
Text file containing the bond parameters for the ligand.

angle_parameter_file : `str` , optional
Text file containing the angle parameters of the ligand.

system_qm_params_file : `str` , optional
A text file containing the QM obtained parameters for the ligand.

reparameterised_intermediate_system_xml_file : `str` , optional
XML foce field file with bond and angle parameter lines replaced by corresponding values obtained from the QM calculations.

system_xml_non_bonded_file : `str` , optional
A text file to write the NonBondedForce Charge Parameters from the non-parametrised system XML file.

system_xml_non_bonded_reparams_file : `str` , optional
Text file containing the non-bonded parameters parsed from the XML force field file.

reparameterised_system_xml_file : `str` , optional
Reparameterized force field XML file obtained using openforcefield.

non_reparameterised_system_xml_file : `str` , optional
Non-reparameterized force field XML file obtained using openforcefield.

prmtop_system_non_params : `str`, optional

Amber generated topology file saved from the non-reparameterized force field XML file for the ligand.

inpcrd_system_non_params : `str`, optional

Amber generated coordinate file saved from the non-reparameterized force field XML file for the ligand.

prmtop_system_params : `str`, optional

Amber generated topology file saved from the reparameterized force field XML file for the ligand.

inpcrd_system_params : `str`, optional

Amber generated coordinate file saved from the reparameterized force field XML file for the ligand.

load_topology : `str`, optional

Argument to specify how to load the topology. Can either be "openmm" or "parmed".

Parameters

charge : `int`

Charge of the ligand.

num_charge_atoms : `int`

Number of charged atoms in the molecule.

charge_atom_1 : `int`

Charge on the first charged atom.

index_charge_atom_1 : `int`

Index of the first charged atom.

system_pdb : `str`, optional

Ligand PDB file with atom numbers beginning from 1.

system_mol2 : `str`, optional

Ligand Mol2 file obtained from PDB file.

system_in : `str`, optional

Prepi file as required by antechamber.

system_frcmod : `str`, optional

FRCMOD file as required by antechamber.

prmtop_system : `str`, optional

Topology file obtained from the ligand PDB.

inpcrd_system : `str` , optional
Coordinate file obtained from the Ligand PDB using the command saveamberparm.

system_leap : `str` , optional
Amber generated leap file for generating and saving topology and coordinate files.

system_xml : `str` , optional
Serilazed XML force field file of the ligand.

system_smi : `str` , optional
Ligand SMILES format file.

system_sdf : `str` , optional
Ligand SDF (structure-data) format file.

system_init_sdf : `str` , optional
Ligand SDF (structure-data) format file. This file will be generated only if the ligand is charged.

index_charge_atom_2 : `int` , optional
Index of the second charged atom of the ligand.

charge_atom_2 : `int` , optional
Charge on the second charged atom of the ligand.

charge_parameter_file : `str` , optional
File containing the charges of ligand atoms and their corresponding atoms.

system_qm_pdb : `str` , optional
Ligand PDB file with atom numbers beginning from 1.

bond_parameter_file : `str` , optional
Text file containing the bond parameters for the ligand.

angle_parameter_file : `str` , optional
Text file containing the angle parameters of the ligand.

system_qm_params_file : `str` , optional
A text file containing the QM obtained parameters for the ligand.

reparameterised_intermediate_system_xml_file : `str` , optional
XML foce field file with bond and angle parameter lines replaced by corresponding values obtained from the QM calculations.

system_xml_non_bonded_file : `str` , optional

A text file to write the NonBondedForce Charge Parameters from the non-parametrised system XML file.

system_xml_non_bonded_reparams_file : `str` , optional

Text file containing the non-bonded parameters parsed from the XML force field file.

reparameterised_system_xml_file : `str` , optional

Reparameterized force field XML file obtained using openforcefield.

non_reparameterised_system_xml_file : `str` , optional

Non-reparameterized force field XML file obtained using openforcefield.

prmtop_system_non_params : `str` , optional

Amber generated topology file saved from the non-reparameterized force field XML file for the ligand.

inpcrd_system_non_params : `str` , optional

Amber generated coordinate file saved from the non-reparameterized force field XML file for the ligand.

prmtop_system_params : `str` , optional

Amber generated topology file saved from the reparameterized force field XML file for the ligand.

inpcrd_system_params : `str` , optional

Amber generated coordinate file saved from the reparameterized force field XML file for the ligand.

load_topology : `str` , optional

Argument to specify how to load the topology. Can either be "openmm" or "parmed".

Methods

```
def analyze_diff_energies(self)
```

Compares the energies of the ligand obtained from the non-parameterized and the parameterized force field files.

```
def generate_xml_antechamber(self)
```

Generates an XML forcefield file from the PDB file through antechamber.

```
def generate_xml_from_charged_pdb_sdf(self)
```

Generates an XML forcefield file for a singly charged ligand molecule from the SDF file through openforcefield.

```
def generate_xml_from_doubly_charged_pdb_sdf(self)
```

Generates an XML forcefield file for a singly charged ligand molecule from the SDF file through openforcefield.

```
def generate_xml_from_pdb_sdf(self)
```

Generates an XML forcefield file from the SDF file through openforcefield.

```
def generate_xml_from_pdb_smi(self)
```

Generates an XML forcefield file from the SMILES file through openforcefield.

```
def save_amber_params(self)
```

Saves amber generated topology files for the ligand.

```
def write_reparameterised_system_xml(self)
```

Writes a reparameterised XML force field file for the ligand.

```
def write_system_params(self)
```

Saves the parameters obtained from the QM log files in a text file.

```
class HostAmberXMLAmber (system_pdb='host.pdb', system_xml='host.xml',
                         sim_output='sim_output.pdb', sim_steps=1000,
                         charge_parameter_file='host_qm_surround_charges.txt',
                         system_qm_pdb='host_qm.pdb',
                         bond_parameter_file='host_qm_bonds.txt',
                         angle_parameter_file='host_qm_angles.txt',
                         system_qm_params_file='host_qm_params.txt',
                         reparameterised_intermediate_system_xml_file='host_intermediate
                         _reparameterised.xml',
                         system_xml_non_bonded_file='host_xml_non_bonded.txt',
                         system_xml_non_bonded_reparams_file='host_xml_non_bonded_repara
                         ms.txt',
                         reparameterised_system_xml_file='host_reparameterised.xml',
                         non_reparameterised_system_xml_file='host.xml',
                    prmtop_system_non_params='host_non_params.prmtop',
                    inpcrd_system_non_params='host_non_params.inpcrd',
                    prmtop_system_params='host_params.prmtop',
                    inpcrd_system_params='host_params.inpcrd',
                    load_topology='openmm')
```

A class used to generate a template force field XML file for the receptor in order regenerate the reparametrised forcefield XML file.

This class contains methods to generate a template XML force field through openforcefield. Re-parameterized XML force field files are then generated from the template files. Different energy components such as the bond,angle, torsional and non-bonded energies are computed for the non-reparametrized and the reparameterized force fields. Difference between the non-reparameterized and reparameterized force field energies can then be analyzed. ...

Attributes

system_pdb : `str`, optional
Receptor PDB file with atom numbers beginning from 1.

system_xml : `str`, optional
Serilazed XML force field file of the receptor.

sim_output : `str`, optional
PDB file containing the trajectory coordinates for the OpenMM simulation.

sim_steps : `str`, optional
Number of steps in the OpenMM MD simulation.

charge_parameter_file : `str`, optional
File containing the charges of receptor atoms and their corresponding atoms.

system_qm_pdb : `str`, optional
Receptor QM region's PDB file with atom numbers beginning from 1.

bond_parameter_file : `str`, optional
Text file containing the bond parameters for the receptor.

angle_parameter_file : `str`, optional
Text file containing the angle parameters of the receptor.

system_qm_params_file : `str`, optional
A text file containing the QM obtained parameters for the receptor.

reparameterised_intermediate_system_xml_file : `str`, optional

XML foce field file with bond and angle parameter lines replaced by corresponding values obtained from the QM calculations.

system_xml_non_bonded_file : `str` , optional
: A text file to write the NonBondedForce Charge Parameters from the non-parametrised system XML file.

system_xml_non_bonded_reparams_file : `str` , optional
: Text file containing the non-bonded parameters parsed from the XML force field file.

reparameterised_system_xml_file : `str` , optional
: Reparameterized force field XML file obtained using openforcefield.

non_reparameterised_system_xml_file : `str` , optional
: Non-reparameterized force field XML file obtained using openforcefield.

prmtop_system_non_params : `str` , optional
: Amber generated topology file saved from the non-reparameterized force field XML file for the receptor.

inpcrd_system_non_params : `str` , optional
: Amber generated coordinate file saved from the non-reparameterized force field XML file for the receptor.

prmtop_system_params : `str` , optional
: Amber generated topology file saved from the reparameterized force field XML file for the receptor.

inpcrd_system_params : `str` , optional
: Amber generated coordinate file saved from the reparameterized force field XML file for the receptor.

load_topology : `str` , optional
: Argument to specify how to load the topology. Can either be "openmm" or "parmed".

Methods

```
def analyze_diff_energies(self)
```

Compares the energies of the ligand obtained from the non-parameterized and the parameterized force field files.

def save_amber_params(self)

Saves amber generated topology files for the ligand.

def serialize_system(self)

def write_reparameterised_system_xml(self)

Writes a reparameterised XML force field file for the ligand.

def write_system_params(self)

Saves the parameters obtained from the QM log files in a text file.

class MergeHostGuestTopology (host_prmtop, guest_prmtop, host_inpcrd, guest_inpcrd, system_prmtop, system_inpcrd)

This class is used to merge the host and guest topology and coordinate files.

…

Attributes

host_prmtop : `str`
Topology file of the receptor.

guest_prmtop : `str`
Topology file of the ligand.

host_inpcrd : `str`
Coordinate file of the receptor.

guest_inpcrd : `str`
Coordinate file of the ligand.

system_prmtop : `str`
Topology file of the receptor - ligand complex.

system_inpcrd : `str`
Coordinate file of the receptor - ligand complex.

Parameters

host_prmtop : `str`
Topology file of the receptor.

guest_prmtop : `str`
: Topology file of the ligand.

host_inpcrd : `str`
: Coordinate file of the receptor.

guest_inpcrd : `str`
: Coordinate file of the ligand.

system_prmtop : `str`
: Topology file of the receptor - ligand complex.

system_inpcrd : `str`
: Coordinate file of the receptor - ligand complex.

Methods

```
def merge_topology_files(self)
```

Merge the host and guest topology and coordinate files.

```
class ParameterizeGuest (vibrational_scaling, xyz_file='guest_coords.xyz',
                         coordinate_file='guest_coordinates.txt',
                         unprocessed_hessian_file='guest_unprocessed_hessian.txt',
                         bond_list_file='guest_bond_list.txt',
                         angle_list_file='guest_angle_list.txt',
                         hessian_file='guest_hessian.txt',
                         atom_names_file='guest_atom_names.txt',
                         bond_parameter_file='guest_bonds.txt',
                         angle_parameter_file='guest_angles.txt',
                         charge_parameter_file='guest_charges.txt',
                         guest_pdb='guest_init_II.pdb',
                         proper_dihedral_file='proper_dihedrals.txt')
```

A class used to obtain force field parameters for the ligand (bond, angle and charge parameters) from QM calculations.

This class contain methods to process the output files of the Gaussian QM output files (.chk, .fchk and .log files). Methods in the class extract the unprocessed hessian matrix from the Gaussian QM calculations, processes it and uses the Modified Seminario Method to ontain the bond and angle parameters. The class also extracts the QM charges from the log file.

…

Attributes

vibrational_scaling : `float`
Vibrational scaling factor for the DFT deficits/anharmonicities in the bond parameters.

xyz_file : `str` , optional
XYZ file for ligand coordinates obtained from its corresponding formatted checkpoint file.

coordinate_file : `str` , optional
Text file containing the ligand coordinates (extracted from the formatted checkpoint file).

unprocessed_hessian_file : `str` , optional
Unprocessed hessian matrix of the ligand obtained from the formatted checkpoint file.

bond_list_file : `str` , optional
Text file containing the bond information of the ligand extracted from the log file.

angle_list_file : `str` , optional
Text file containing the angle information of the ligand extracted from the log file.

hessian_file : `str` , optional
Processed hessian matrix of the ligand.

atom_names_file : `str` , optional
Text file containing the list of atom names from the fchk file.

bond_parameter_file : `str` , optional
Text file containing the bond parameters for the ligand obtained using the Modified Seminario method.

angle_parameter_file : `str` , optional
Text file containing the angle parameters of the ligand obtained using the Modified Seminario method..

charge_parameter_file : `str` , optional
Text file containing the QM charges of the ligand.

guest_pdb : `str` , optional
Ligand PDB file with atom numbers beginning from 1.

proper_dihedral_file : `str` , optional

A text file containing proper dihedral angles of the ligand.

Parameters

vibrational_scaling : `float`
Vibrational scaling factor for the DFT deficits/anharmonicities in the bond parameters.

xyz_file : `str`, optional
XYZ file for ligand coordinates obtained from its corresponding formatted checkpoint file.

coordinate_file : `str`, optional
Text file containing the ligand coordinates (extracted from the formatted checkpoint file).

unprocessed_hessian_file : `str`, optional
Unprocessed hessian matrix of the ligand obtained from the formatted checkpoint file.

bond_list_file : `str`, optional
Text file containing the bond information of the ligand extracted from the log file.

angle_list_file : `str`, optional
Text file containing the angle information of the ligand extracted from the log file.

hessian_file : `str`, optional
Processed hessian matrix of the ligand.

atom_names_file : `str`, optional
Text file containing the list of atom names from the fchk file.

bond_parameter_file : `str`, optional
Text file containing the bond parameters for the ligand obtained using the Modified Seminario method.

angle_parameter_file : `str`, optional
Text file containing the angle parameters of the ligand.

charge_parameter_file : `str`, optional
Text file containing the QM charges of the ligand.

guest_pdb : `str`, optional
Ligand PDB file with atom numbers beginning from 1.

proper_dihedral_file : `str`, optional
A text file containing proper dihedral angles of the ligand.

Methods

```
def get_atom_names(self)
```

Saves a list of atom names from the formatted checkpoint file.

```
def get_bond_angle_params(self)
```

Saves the bond and angle parameter files obtained from the formatted checkpoint file.

```
def get_bond_angles(self)
```

Saves a text file containing bonds and angles from the gaussian log file.

```
def get_charges(self)
```

Saves the atomic charges in a text file obtained from the Gaussian log file.

```
def get_hessian(self)
```

Extracts hessian matrix from the unprocessed hessian matrix and saves into a new file.

```
def get_proper_dihedrals(self)
```

Saves proper dihedral angles of the ligand in a text file.

```
def get_unprocessed_hessian(self)
```

Saves a text file of the unprocessed hessian matrix from the formatted checkpoint file.

```
def get_xyz(self)
```

Saves XYZ file from the formatted checkpoint file.

```
class ParameterizeHost (vibrational_scaling, xyz_file='host_qm_coords.xyz',
                        coordinate_file='host_qm_coordinates.txt',
                        unprocessed_hessian_file='host_qm_unprocessed_hessian.txt',
                        bond_list_file='host_qm_bond_list.txt',
                        angle_list_file='host_qm_angle_list.txt',
                        hessian_file='host_qm_hessian.txt',
                        atom_names_file='host_qm_atom_names.txt',
                        bond_parameter_file='host_qm_bonds.txt',
                    angle_parameter_file='host_qm_angles.txt',
                    charge_parameter_file='host_qm_surround_charges.txt',
                    host_qm_pdb='host_qm.pdb')
```

A class used to obtain force field parameters for the QM region of the receptor (bond, angle and charge parameters) from QM calculations.

This class contain methods to process the output files of the Gaussian QM output files (.chk, .fchk and .log files). Methods in the class extract the unprocessed hessian matrix from the Gaussian QM calculations, processes it and uses the Modified Seminario Method to ontain the bond and angle parameters. The class also extracts the QM charges from the log file.

...

Attributes

vibrational_scaling : `float`
Vibrational scaling factor for the DFT deficits/anharmonicities in the bond parameters.

xyz_file : `str` , optional
XYZ file for ligand coordinates obtained from its corresponding formatted checkpoint file.

coordinate_file : `str` , optional
Text file containing the receptor coordinates (extracted from the formatted checkpoint file).

unprocessed_hessian_file : `str` , optional
Unprocessed hessian matrix of the receptor obtained from the formatted checkpoint file.

bond_list_file : `str` , optional
Text file containing the bond information of the receptor extracted from the log file.

angle_list_file : `str` , optional
Text file containing the angle information of the receptor extracted from the log file.

hessian_file : `str` , optional
Processed hessian matrix of the receptor.

atom_names_file : `str` , optional
Text file containing the list of atom names from the fchk file.

bond_parameter_file : `str` , optional
Text file containing the bond parameters for the receptor obtained using the Modified Seminario method.

angle_parameter_file : `str` , optional
Text file containing the angle parameters of the receptor.

charge_parameter_file : `str` , optional
Text file containing the QM charges of the receptor.

host_qm_pdb : `str` , optional
PDB file for the receptor's QM region.

Parameters

vibrational_scaling : `float`
Vibrational scaling factor for the DFT deficits/anharmonicities in the bond parameters.

xyz_file : `str` , optional
XYZ file for ligand coordinates obtained from its corresponding formatted checkpoint file.

coordinate_file : `str` , optional
Text file containing the receptor coordinates (extracted from the formatted checkpoint file).

unprocessed_hessian_file : `str` , optional
Unprocessed hessian matrix of the receptor obtained from the formatted checkpoint file.

bond_list_file : `str` , optional
Text file containing the bond information of the receptor extracted from the log file.

angle_list_file : `str` , optional
Text file containing the angle information of the receptor extracted from the log file.

hessian_file : `str` , optional
Processed hessian matrix of the receptor.

atom_names_file : `str` , optional
Text file containing the list of atom names from the fchk file.

bond_parameter_file : `str` , optional

Text file containing the bond parameters for the receptor obtained using the Modified Seminario method.

angle_parameter_file : `str` , optional

Text file containing the angle parameters of the receptor.

charge_parameter_file : `str` , optional

Text file containing the QM charges of the receptor.

host_qm_pdb : `str` , optional

PDB file for the receptor's QM region.

Methods

def get_atom_names(self)

Saves a list of atom names from the formatted checkpoint file.

def get_bond_angle_params(self)

Saves the bond and angle parameter files obtained from the formatted checkpoint file.

def get_bond_angles(self)

Saves a text file containing bonds and angles from the gaussian log file.

def get_charges(self)

Saves the atomic charges in a text file obtained from the Gaussian log file.

def get_hessian(self)

Extracts hessian matrix from the unprocessed hessian matrix and saves into a new file.

def get_unprocessed_hessian(self)

Saves a text file of the unprocessed hessian matrix from the formatted checkpoint file.

def get_xyz(self)

Saves XYZ file from the formatted checkpoint file.

```
class PrepareGaussianGuest (charge, multiplicity, guest_pdb='guest_init_II.pdb',
                            n_processors=12, memory=50, functional='B3LYP',
                            basis_set='6-31G', optimisation='OPT', frequency='FREQ',
                            add_keywords_I='Integral=(Grid=UltraFine)',
                            add_keywords_II='Pop(MK,ReadRadii)',
                            add_keywords_III='IOp(6/33=2,6/42=6)',
                            gauss_out_file='guest.out', fchk_out_file='guest_fchk.out')
```

A class used to prepare the QM engine input file (Gaussian) for the ligand and run QM calculations with appropriate keywords.

This class contain methods to write an input file (.com extension) for the QM engine. It then runs a QM calculation with the given basis set and functional. Checkpoint file is then converted to a formatted checkpoint file. Output files (.log, .chk, and .fhck) will then be used to extract ligand's force field parameters.

...

Attributes

charge : `int`
Charge of the ligand.

multiplicity : `int`
Spin Multiplicity (2S+1) of the ligand where S represents the total spin of the ligand.

guest_pdb : `str` , optional
Ligand PDB file with atom numbers beginning from 1.

n_processors : `int` , optional
Number of processors to be used for Gaussian program to run and set in %NProcShared command of Gaussian.

memory : `int` , optional
Memory (in GB) to be used set in %Mem command of Gaussian.

functional : `str` , optional
Exchange/Correlation or hybrid functional to use in the Gaussian QM calculation.

basis_set : `str` , optional
Basis set to use for the Gaussian QM calculation.

optimisation : `str` , optional
set to "OPT" to perform a geometry optimization on the ligand specified in the system; else set to an empty string.

frequency : `str` , optional

set to "FREQ" for Gaussian to perform a frequency calculation; else set to an empty string.

add_keywords_I : `str` , optional

Specifies the integration grid.

add_keywords_II : `str` , optional

Specifies the Gaussian described options for generating population.

add_keywords_III : `str` , optional

Used to include the IOp keyword (to set the internal options to specific values) in the Gaussian command.

gauss_out_file : `str` , optional

This file contains the output script obtained after running the Gaussian QM calculation.

fchk_out_file : `str` , optional

Formatted checkpoint file obtained from the checkpoint file using formchk command.

Parameters

charge : `int`

Charge of the ligand.

multiplicity : `int`

Spin Multiplicity (2S+1) of the ligand where S represents the total spin of the ligand.

guest_pdb : `str` , optional

Ligand PDB file with atom numbers beginning from 1.

n_processors : `int` , optional

Number of processors to be used for Gaussian program to run and set in %NProcShared command of Gaussian.

memory : `int` , optional

Memory (in GB) to be used set in %Mem command of Gaussian.

functional : `str` , optional

Exchange/Correlation or hybrid functional to use in the Gaussian QM calculation.

basis_set : `str` , optional

Basis set to use for the Gaussian QM calculation.

optimisation : `str` , optional
set to "OPT" to perform a geometry optimization on the ligand specified in the system; else set to an empty string.

frequency : `str` , optional
set to "FREQ" for Gaussian to perform a frequency calculation; else set to an empty string.

add_keywords_I : `str` , optional
Specifies the integration grid.

add_keywords_II : `str` , optional
Specifies the Gaussian described options for generating population.

add_keywords_III : `str` , optional
Used to include the IOp keyword (to set the internal options to specific values) in the Gaussian command.

gauss_out_file : `str` , optional
This file contains the output script obtained after running the Gaussian QM calculation.

fchk_out_file : `str` , optional
Formatted checkpoint file obtained from the checkpoint file using formchk command.

Methods

def get_fchk(self)

Converts the Gaussian checkpoint file (.chk) to a formatted checkpoint file (.fchk).

def run_gaussian(self)

Runs the Gaussian QM calculation for the ligand locally.

def write_input(self)

Writes a Gaussian input file for the ligand.

class PrepareGaussianHost (charge, multiplicity, host_qm_pdb='host_qm.pdb', n_processors=12, memory=50, functional='B3LYP', basis_set='6-

```
31G', optimisation='OPT', frequency='FREQ',
add_keywords_I='Integral=(Grid=UltraFine)',
add_keywords_II='Pop(MK,ReadRadii)',
add_keywords_III='IOp(6/33=2,6/42=6)',
gauss_out_file='host_qm.out',
fchk_out_file='host_qm_fchk.out')
```

A class used to prepare the QM engine input file (Gaussian) for the receptor and run QM calculations with appropriate keywords.

This class contain methods to write an input file (.com extension) for the QM engine. It then runs a QM calculation with the given basis set and functional. Checkpoint file is then converted to a formatted checkpoint file. Output files (.log, .chk, and .fhck) will then be used to extract receptors's force field parameters.

...

Attributes

charge : `int`
Charge of the receptor.

multiplicity : `int`
Spin Multiplicity (2S+1) of the receptor where S represents the total spin of the receptor.

host_qm_pdb : `str` , optional
PDB file of the receptor's QM region with atom numbers beginning from 1.

n_processors : `int` , optional
Number of processors to be used for Gaussian program to run and set in %NProcShared command of Gaussian.

memory : `int` , optional
Memory (in GB) to be used set in %Mem command of Gaussian.

functional : `str` , optional
Exchange/Correlation or hybrid functional to use in the Gaussian QM calculation.

basis_set : `str` , optional
Basis set to use for the Gaussian QM calculation.

optimisation : `str` , optional
set to "OPT" to perform a geometry optimization on the receptor specified in the system; else set to an empty string.

frequency : `str` , optional

set to "FREQ" for Gaussian to perform a frequency calculation; else set to an empty string.

add_keywords_I : `str`, optional
Specifies the integration grid.

add_keywords_II : `str`, optional
Specifies the Gaussian described options for generating population.

add_keywords_III : `str`, optional
Used to include the IOp keyword (to set the internal options to specific values) in the Gaussian command.

gauss_out_file : `str`, optional
This file contains the output script obtained after running the Gaussian QM calculation.

fchk_out_file : `str`, optional
Formatted checkpoint file obtained from the checkpoint file using formchk command.

Parameters

charge : `int`
Charge of the receptor.

multiplicity : `int`
Spin Multiplicity (2S+1) of the receptor where S represents the total spin of the receptor.

host_qm_pdb : `str`, optional
PDB file of the receptor's QM region with atom numbers beginning from 1.

n_processors : `int`, optional
Number of processors to be used for Gaussian program to run and set in %NProcShared command of Gaussian.

memory : `int`, optional
Memory (in GB) to be used set in %Mem command of Gaussian.

functional : `str`, optional
Exchange/Correlation or hybrid functional to use in the Gaussian QM calculation.

basis_set : `str`, optional
Basis set to use for the Gaussian QM calculation.

optimisation : `str`, optional

set to "OPT" to perform a geometry optimization on the receptor specified in the system; else set to an empty string.

frequency : `str`, optional

set to "FREQ" for Gaussian to perform a frequency calculation; else set to an empty string.

add_keywords_I : `str`, optional

Specifies the integration grid.

add_keywords_II : `str`, optional

Specifies the Gaussian described options for generating population.

add_keywords_III : `str`, optional

Used to include the IOp keyword (to set the internal options to specific values) in the Gaussian command.

gauss_out_file : `str`, optional

This file contains the output script obtained after running the Gaussian QM calculation.

fchk_out_file : `str`, optional

Formatted checkpoint file obtained from the checkpoint file using formchk command.

Methods

```
def get_fchk(self)
```

Converts the Gaussian checkpoint file (.chk) to a formatted checkpoint file (.fchk).

```
def run_gaussian(self)
```

Runs the Gaussian QM calculation for the receptor locally.

```
def write_input(self)
```

Writes a Gaussian input file for the receptor QM region.

```
class PrepareGaussianHostGuest (charge, multiplicity, guest_pdb='guest_init_II.pdb',
                                host_qm_pdb='host_qm.pdb', n_processors=12, memory=50,
                                functional='B3LYP', basis_set='6-31G', optimisation='',
frequency='', add_keywords_I='Integral=
(Grid=UltraFine)', add_keywords_II='Pop(MK,ReadRadii)',
add_keywords_III='IOp(6/33=2,6/42=6)',
gauss_system_out_file='system_qm.out',
fchk_system_out_file='system_qm_fchk.out',
host_guest_input='host_guest.com',
qm_guest_charge_parameter_file='guest_qm_surround_charge
s.txt',
qm_host_charge_parameter_file='host_qm_surround_charges.
txt',
qm_guest_atom_charge_parameter_file='guest_qm_atom_surro
und_charges.txt')
```

A class used to prepare the QM engine input file (Gaussian) for the receptor - ligand complex and run the QM calculations with the appropriate keywords.

This class contain methods to write an input file (.com extension) for the QM engine for the receptor - ligand complex. It then runs a QM calculation with the given basis set and functional. Checkpoint file is then converted to a formatted checkpoint file. Output files (.log, .chk, and .fhck) will then be used to extract charges for the ligand and the receptor.

...

Attributes

charge : `int`

Total charge of the receptor - ligand complex.

multiplicity : `int`

Spin Multiplicity (2S+1) of the ligand where S represents the total spin of the ligand.

guest_pdb : `str` , optional

Ligand PDB file with atom numbers beginning from 1.

host_qm_pdb : `str` , optional

PDB file for the receptor's QM region.

n_processors : `int` , optional

Number of processors to be used for Gaussian program to run and set in %NProcShared command of Gaussian.

memory : `int` , optional

Memory (in GB) to be used set in %Mem command of Gaussian.

functional : `str`, optional
    Exchange/Correlation or hybrid functional to use in the Gaussian QM calculation.

basis_set : `str`, optional
    Basis set to use for the Gaussian QM calculation.

optimisation : `str`, optional
    set to "OPT" to perform a geometry optimization on the ligand specified in the system; else set to an empty string.

frequency : `str`, optional
    set to "FREQ" for Gaussian to perform a frequency calculation; else set to an empty string.

add_keywords_I : `str`, optional
    Specifies the integration grid.

add_keywords_II : `str`, optional
    Specifies the Gaussian described options for generating population.

add_keywords_III : `str`, optional
    Used to include the IOp keyword (to set the internal options to specific values) in the Gaussian command.

gauss_system_out_file : `str`, optional
    This file contains the output script obtained after running the Gaussian QM calculation.

fchk_system_out_file : `str`, optional
    Formatted checkpoint file obtained from the checkpoint file using formchk command.

host_guest_input : `str`, optional
    Gaussian input file (.com extension) for the receptor - ligand QM region.

qm_guest_charge_parameter_file : `str`, optional
    File containing the charges of ligand atoms and their corresponding atoms. Charge obtained are the polarised charged due to the surrounding receptor's region.

qm_host_charge_parameter_file : `str`, optional
    File containing the charges of the QM region of the receptor.

qm_guest_atom_charge_parameter_file : `str`, optional
    File containing the charges of ligand atoms. Charge obtained are the polarised charged due to the surrounding receptor's region.

Parameters

charge : `int`

Total charge of the receptor - ligand complex.

multiplicity : `int`

Spin Multiplicity (2S+1) of the ligand where S represents the total spin of the ligand.

guest_pdb : `str` , optional

Ligand PDB file with atom numbers beginning from 1.

host_qm_pdb : `str` , optional

PDB file for the receptor's QM region.

n_processors : `int` , optional

Number of processors to be used for Gaussian program to run and set in %NProcShared command of Gaussian.

memory : `int` , optional

Memory (in GB) to be used set in %Mem command of Gaussian.

functional : `str` , optional

Exchange/Correlation or hybrid functional to use in the Gaussian QM calculation.

basis_set : `str` , optional

Basis set to use for the Gaussian QM calculation.

optimisation : `str` , optional

set to "OPT" to perform a geometry optimization on the ligand specified in the system; else set to an empty string.

frequency : `str` , optional

set to "FREQ" for Gaussian to perform a frequency calculation; else set to an empty string.

add_keywords_I : `str` , optional

Specifies the integration grid.

add_keywords_II : `str` , optional

Specifies the Gaussian described options for generating population.

add_keywords_III : `str` , optional

Used to include the IOp keyword (to set the internal options to specific values) in the Gaussian command.

gauss_system_out_file : `str` , optional

This file contains the output script obtained after running the Gaussian QM calculation.

`fchk_system_out_file` : `str` , optional

Formatted checkpoint file obtained from the checkpoint file using formchk command.

`host_guest_input` : `str` , optional

Gaussian input file (.com extension) for the receptor - ligand QM region.

`qm_guest_charge_parameter_file` : `str` , optional

File containing the charges of ligand atoms and their corresponding atoms. Charge obtained are the polarised charged due to the surrounding receptor's region.

`qm_host_charge_parameter_file` : `str` , optional

File containing the charges of the QM region of the receptor.

`qm_guest_atom_charge_parameter_file` : `str` , optional

File containing the charges of ligand atoms. Charge obtained are the polarised charged due to the surrounding receptor's region.

Methods

```
def get_fchk(self)
```

Converts the Gaussian checkpoint file (.chk) to a formatted checkpoint file (.fchk).

```
def get_qm_host_guest_charges(self)
```

Extract charge information for the receptor - ligand QM region.

```
def run_gaussian(self)
```

Runs the Gaussian QM calculation for the ligand - receptor region locally.

```
def write_input(self)
```

Writes a Gaussian input file for the receptor - ligand QM region.

```
class PrepareQMMM (init_pdb, distance, num_residues, guest_resname,
                   cleaned_pdb='system.pdb', guest_init_pdb='guest_init.pdb',
                   host_pdb='host.pdb', guest_pdb='guest_init_II.pdb',
guest_xyz='guest_coord.txt', residue_list='residue_list.txt',
host_qm_atoms='host_qm.txt', host_mm_atoms='host_mm.txt',
host_qm_pdb='host_qm.pdb', host_mm_pdb='host_mm.pdb',
qm_pdb='qm.pdb', mm_pdb='mm.pdb',
host_mm_region_I_atoms='host_mm_region_I.txt',
host_mm_region_II_atoms='host_mm_region_II.txt',
host_mm_region_I_pdb='host_mm_region_I.pdb',
host_mm_region_II_pdb='host_mm_region_II.pdb')
```

A class used to segregate the QM and MM regions.

This class contains methods to remove the solvent, ions and all entities that are exclusive of receptor and the ligand. It also defines the Quantum Mechanical (QM) region and the Molecular Mechanical (MM) region based upon the distance of the ligand from the receptor and the chosen number of receptor residues. It is also assumed that the initial PDB file will have the receptor followed by the ligand.

…

Attributes

init_pdb : `str`
Initial PDB file containing the receptor-ligand complex with solvent, ions, etc.

cleaned_pdb : `str`
Formatted PDB file containing only the receptor and the ligand.

guest_init_pdb : `str`
A separate ligand PDB file with atom numbers not beginning from 1.

host_pdb : `str`
A separate receptor PDB file with atom numbers beginning from 1.

guest_resname : `str`
Three letter residue ID for the ligand.

guest_pdb : `str` , optional
Ligand PDB file with atom numbers beginning from 1.

guest_xyz : `str` , optional
A text file of the XYZ corordinates of the ligand.

distance : `float` , optional
The distance required to define the QM region of the receptor. This is the distance between the atoms of the ligand and the atoms of the receptor.

residue_list : `str` , optional

A text file of the residue numbers of the receptor within the proximity (as defined by the distance) from the ligand.

host_qm_atoms : `str` , optional
: A text file of the atom numbers of the receptors in the QM region.

host_mm_atoms : `str` , optional
: A text file of the atom numbers of the receptors in the MM region (all atoms except atoms in the QM region)

host_qm_pdb : `str` , optional
: PDB file for the receptor's QM region.

host_mm_pdb : `str` , optional
: PDB file for the receptor's MM region.

qm_pdb : `str` , optional
: PDB file for the QM region (receptor's QM region and the ligand).

mm_pdb : `str` , optional
: PDB file for the MM region.

host_mm_region_I_atoms : `str` , optional
: A text file of the atom numbers of the receptors in the MM region preceeding the QM region.

host_mm_region_II_atoms : `str` , optional
: A text file of the atom numbers of the receptors in the MM region following the QM region.

host_mm_region_I_pdb : `str` , optional
: PDB file of the receptor in the MM region preceeding the QM region.

host_mm_region_II_pdb : `str` , optional
: PDB file of the receptor in the MM region following the QM region.

num_residues : `int` , optional
: Number of residues required in the QM region of the receptor.

Parameters

init_pdb : `str`
: Initial PDB file containing the receptor-ligand complex with solvent, ions, etc.

cleaned_pdb : `str`
: Formatted PDB file containing only the receptor and the ligand.

guest_init_pdb : `str`

A separate ligand PDB file with atom numbers not beginning from 1.

host_pdb : `str`

A separate receptor PDB file with atom numbers beginning from 1.

guest_resname : `str`

Three letter residue ID for the ligand.

guest_pdb : `str`

Ligand PDB file with atom numbers beginning from 1.

guest_xyz : `str`

A text file of the XYZ corordinates of the ligand.

distance : `float`

The distance required to define the QM region of the receptor. This is the distance between the atoms of the ligand and the atoms of the receptor.

residue_list : `str`

A text file of the residue numbers of the receptor within the proximity (as defined by the distance) from the ligand.

host_qm_atoms : `str`

A text file of the atom numbers of the receptors in the QM region.

host_mm_atoms : `str`

A text file of the atom numbers of the receptors in the MM region (all atoms except atoms in the QM region)

host_qm_pdb : `str`

PDB file for the receptor's QM region.

host_mm_pdb : `str`

PDB file for the receptor's MM region.

qm_pdb : `str`

PDB file for the QM region (receptor's QM region and the ligand).

mm_pdb : `str`

PDB file for the MM region.

host_mm_region_I_atoms : `str`

A text file of the atom numbers of the receptors in the MM region preceeding the QM region.

host_mm_region_II_atoms : `str`

A text file of the atom numbers of the receptors in the MM region following the QM region.

host_mm_region_I_pdb : `str`
PDB file of the receptor in the MM region preceeding the QM region.

host_mm_region_II_pdb : `str`
PDB file of the receptor in the MM region following the QM region.

num_residues : `int`
Number of residues required in the QM region of the receptor.

Methods

```
def clean_up(self)
```

Reads the given PDB file, removes all entities except the receptor and ligand and saves a new pdb file.

```
def create_host_guest(self)
```

Saves separate receptor and ligand PDB files.

```
def get_guest_coord(self)
```

Saves a text file of the XYZ corordinates of the ligand.

```
def get_host_mm_region_atoms(self)
```

Saves a text file for the atoms of the receptor's MM region preceding the QM region and saves another text file for the atoms of the receptor's MM region folllowing the QM region.

```
def get_host_qm_mm_atoms(self)
```

Saves a text file of the atom numbers of the receptors in the QM region and MM region separately.

```
def get_qm_mm_regions(self)
```

Saves separate PDB files for the QM and MM regions. QM regions comprise the QM region of the receptor and the entire ligand where the MM region comprise the non-selected QM regions of the receptor.

```
def get_qm_resids(self)
```

Saves a text file of the residue numbers of the receptor within the proximity (as defined by the distance) from the ligand.

`def realign_guest(self)`

Saves a ligand PDB file with atom numbers beginning from 1.

`def save_host_mm_regions_pdbs(self)`

Saves a PDB file for the receptor's MM region preceding the QM region and saves another PDB file for the receptor's MM region folllowing the QM region.

`def save_host_pdbs(self)`

Saves a PDB file for the receptor's QM region and MM region separately.

```
class RunOpenMMSims (system_prmtop, system_inpcrd, system_pdb,
                    system_output='sim_output.pdb', sim_steps=1000)
```

A class used to run the OpenMM simulation on any specified system.

This class contains methods to run a MD simulation to confirm the proper structure of the reparameterized forcefield files.

…

Attributes

system_prmtop : `str`
Topology file of the system (receptor, ligand or receptor - ligand complex)

system_inpcrd : `str`
Coordinate file of the system (receptor, ligand or receptor - ligand complex)

system_pdb : `str`
PDB file of the system to run MD simulation (receptor, ligand or receptor - ligand complex).

sim_output : `str` , optional
PDB file containing the trajectory coordinates for the OpenMM simulation.

sim_steps : `str` , optional
Number of steps in the OpenMM MD simulation.

Parameters

system_prmtop : `str`
: Topology file of the system (receptor, ligand or receptor - ligand complex)

system_inpcrd : `str`
: Coordinate file of the system (receptor, ligand or receptor - ligand complex)

system_pdb : `str`
: PDB file of the system to run MD simulation (receptor, ligand or receptor - ligand complex).

sim_output : `str` , optional
: PDB file containing the trajectory coordinates for the OpenMM simulation.

sim_steps : `str` , optional
: Number of steps in the OpenMM MD simulation.

Methods

```
def run_openmm_prmtop_inpcrd(self)
```

Run a OpenMM MD simulation with prmtop and inpcrd file"

```
def run_openmm_prmtop_pdb(self)
```

Run a OpenMM MD simulation with prmtop and PDB file"

```
class TorsionDriveParams (num_charge_atoms, index_charge_atom_1, charge_atom_1,
                          tor_dir='torsion_dir',
                          reparameterized_torsional_params_file='reparameterized_torsional_params.txt', psi_input_file='torsion_drive_input.dat',
                          xyz_file='torsion_drive_input.xyz',
                          coords_file='torsion_drive_input.txt',
                          template_pdb='guest_init_II.pdb',
                          system_pdb='torsion_drive_input.pdb',
                          system_sdf='torsion_drive_input.sdf',
                          system_xml='torsion_drive_input.xml', qm_scan_file='scan.xyz',
                          load_topology='openmm', method='L-BFGS-B',
                          dihedral_text_file='dihedrals.txt',
                          system_init_sdf='torsion_drive_input_init.sdf',
                          reparameterised_system_xml_file='guest_reparameterised.xml',
                    reparameterised_torsional_system_xml_file='guest_torsional_rep
                    arameterized.xml')
```

This class is used to parameterize the torsional parameters of the ligand by fitting the torsional parameters obtained from torsiondrive calculations.

… Attributes

---

num_charge_atoms : `int`
Number of charged atoms in the molecule.

index_charge_atom_1 : int Index of the first charged atom.

charge_atom_1 : int Charge on the first charged atom.

tor_dir : str, optional Torsiondrive directory containing separate torsiondrive folders, each containing files for a separate torsiondrive calculation for a particular dihedral angle.

reparameterized_torsional_params_file : str, optional Text file containing the forcefield parameters for the ligand previously obtained without torsional reparameterization.

psi_input_file : str, optional Input file for psi4 QM engine.

xyz_file : str, optional XYZ file for ligand coordinates.

coords_file : str, optional Text file containing the XYZ coordinates of the ligand.

template_pdb : str, optional Guest PDB with atoms beginning from 1 to be used as a template PDB to retrieve atom indices and symbols.

system_pdb : str, optional PDB file for the torsiondrive torsion scans.

system_sdf : str, optional SDF file for the ligand.

system_xml : str, optional XML force field file for the ligand.

qm_scan_file : str, optional Output scan file for the torsiondrive scans.

load_topology : str, optional Argument to specify how to load the topology. Can either be "openmm" or "parmed".

method : str, optional Minimization method for fitting of torsional parameters.

dihedral_text_file : str, optional Dihedral information file for torsiondrive.

system_init_sdf : str, optional Ligand SDF (structure-data) format file. This file will be generated only if the ligand is charged.

reparameterised_system_xml_file : str, optional Reparameterized force field XML file obtained using openforcefield without torsional reparamaterization.

reparameterised_torsional_system_xml_file : str, optional XML force field file for the ligand obtained with torsional reparamaterization.

Parameters

num_charge_atoms : `int`

Number of charged atoms in the molecule.

index_charge_atom_1 : int Index of the first charged atom.

charge_atom_1 : int Charge on the first charged atom.

tor_dir : str, optional Torsiondrive directory containing separate torsiondrive folders, each containing files for a separate torsiondrive calculation for a particular dihedral angle.

reparameterized_torsional_params_file : str, optional Text file containing the forcefield parameters for the ligand previously obtained without torsional reparameterization.

psi_input_file : str, optional Input file for psi4 QM engine.

xyz_file : str, optional XYZ file for ligand coordinates.

coords_file : str, optional Text file containing the XYZ coordinates of the ligand.

template_pdb : str, optional Guest PDB with atoms beginning from 1 to be used as a template PDB to retrieve atom indices and symbols.

system_pdb : str, optional PDB file for the torsiondrive torsion scans.

system_sdf : str, optional SDF file for the ligand.

system_xml : str, optional XML force field file for the ligand.

qm_scan_file : str, optional Output scan file for the torsiondrive scans.

load_topology : str, optional Argument to specify how to load the topology. Can either be "openmm" or "parmed".

method : str, optional Minimization method for fitting of torsional parameters.

dihedral_text_file : str, optional Dihedral information file for torsiondrive.

system_init_sdf : str, optional Ligand SDF (structure-data) format file. This file will be generated only if the ligand is charged.

reparameterised_system_xml_file : str, optional Reparameterized force field XML file obtained using openforcefield without torsional reparamaterization.

reparameterised_torsional_system_xml_file : str, optional XML force field file for the ligand obtained with torsional reparamaterization.

Methods

```
def write_reparams_torsion_lines(self)
```

Saves a text file containing torsional parameters for the ligand obtained through openforcefield.

```
def write_reparams_torsion_lines_charged(self)
```

Saves a text file containing torsional parameters for a charged ligand obtained through openforcefield.

```
def write_torsional_reparams(self)
```

Generates a XML force field file for the ligand with reparameterized torsional parameters.

```
class TorsionDriveSims (charge, multiplicity,
                        reparameterised_system_xml_file='guest_reparameterised.xml',
                        torsion_xml_file='guest_torsion_xml.txt',
                        xyz_file='guest_coords.xyz',
                        psi_input_file='torsion_drive_input.dat', memory=50,
                        basis_set='STO-3G', functional='BLYP', iterations=2000,
                        method_torsion_drive='native_opt',
                        system_bonds_file='guest_bonds.txt', tor_dir='torsion_dir',
                        dihedral_text_file='dihedrals.txt',
                        template_pdb='guest_init_II.pdb',
                        torsion_drive_run_file='run_command', dihedral_interval=15,
                        engine='psi4', energy_threshold=0.001)
```

This class is used to create a filetree for torsion scan using torsionsdrive for the dihedral angles of the ligand.

This class creates a directory for carrying out torsiondrive calculations followed by fitting of torsional parameters. Methods in this class are used to run torsiondrive calculations either for all of the torsional angles, or for non-hydrogen / heavy atoms contributing to the torsional angle.

...

Attributes

charge : `int`
Charge of the ligand.

multiplicity : `int`
: Spin Multiplicity (2S+1) of the ligand where S represents the total spin of the ligand.

reparameterised_system_xml_file : `str` , optional
: Reparamaterixed XML force field for the ligand.

torsion_xml_file : `str` , optional
: A text file containing torsional parameters from reparametrised XML file.

xyz_file : `str` , optional
: XYZ file containing the coordinates of the guest molecule.

psi_input_file : `str` , optional
: Input file for psi4 QM engine.

memory : `int` , optional
: Memory (in GB) to be used.

basis_set : `str` , optional
: Basis set to use for the QM engine.

functional : `str` , optional
: Exchange/Correlation or hybrid Functional for the QM engine.

iterations : `int` , optional
: Maximum number of geometry optimization steps.

method_torsion_drive : `str` , optional
: The algorithm/package to use while running the torsiondrive scan. Using –native_opt uses QM program native constrained optimization algorithm and turns off geomeTRIC package.

system_bonds_file : `str` , optional
: Text file containing bond parameters for the ligand.

tor_dir : `str` , optional
: Torsiondrive directory containing separate torsiondrive folders, each containing files for a separate torsiondrive calculation for a particular dihedral angle.

dihedral_text_file : `str` , optional

Dihedral information file for torsiondrive.

template_pdb : `str` , optional

Guest PDB with atoms beginning from 1 to be used as a template PDB to retrieve atom indices and symbols.

torsion_drive_run_file : `str` , optional
bash file for torsiondrive calculations.

dihedral_interval : `int` , optional
Grid spacing for dihedral scan, i.e. every n degrees (where n is an integer), multiple values will be mapped to each dihedral angle.

engine : `str` , optional
Engine for running torsiondrive scan.

energy_threshold : `float` , optional
Only activate grid points if the new optimization is lower than the previous lowest energy (in a.u.).

Parameters

charge : `int`
Charge of the ligand.

multiplicity : `int`
Spin Multiplicity (2S+1) of the ligand where S represents the total spin of the ligand.

reparameterised_system_xml_file : `str` , optional
Reparamaterixed XML force field for the ligand.

torsion_xml_file : `str` , optional
A text file containing torsional parameters from reparametrised XML file.

xyz_file : `str` , optional
XYZ file containing the coordinates of the guest molecule.

psi_input_file : `str` , optional
Input file for psi4 QM engine.

memory : `int` , optional
Memory (in GB) to be used.

basis_set : `str` , optional
Basis set to use for the QM engine.

functional : `str` , optional
Exchange/Correlation or hybrid Functional for the QM engine.

iterations : `int` , optional
: Maximum number of geometry optimization steps.

method_torsion_drive : `str` , optional
: The algorithm/package to use while running the torsiondrive scan. Using – native_opt uses QM program native constrained optimization algorithm and turns off geomeTRIC package.

system_bonds_file : `str` , optional
: Text file containing bond parameters for the ligand.

tor_dir : `str` , optional
: Torsiondrive directory containing separate torsiondrive folders, each containing files for a separate torsiondrive calculation for a particular dihedral angle.

dihedral_text_file : `str` , optional

Dihedral information file for torsiondrive.

template_pdb : `str` , optional
: Guest PDB with atoms beginning from 1 to be used as a template PDB to retrieve atom indices and symbols.

torsion_drive_run_file : `str` , optional
: bash file for torsiondrive calculations.

dihedral_interval : `int` , optional
: Grid spacing for dihedral scan, i.e. every n degrees (where n is an integer), multiple values will be mapped to each dihedral angle.

engine : `str` , optional
: Engine for running torsiondrive scan.

energy_threshold : `float` , optional
: Only activate grid points if the new optimization is lower than the previous lowest energy (in a.u.).

Methods

```
def create_non_H_bonded_torsion_drive_dir(self)
```

Creates a directory for carrying out torsiondrive calculations for all non-hydrogen bonded torsional angles.

`def create_non_H_torsion_drive_dir(self)`

Creates a directory for carrying out torsiondrive calculations for all non-hydrogen torsional angles.

`def create_torsion_drive_dir(self)`

Creates a directory for carrying out torsiondrive calculations for all the proper dihedral angles.

`def run_torsion_sim(self)`

Run torsion scans using torsiondrive locally.

`def write_psi4_input(self)`

Writes a psi4 input QM file.

`def write_tor_params_txt(self)`

Saves a text file containing torsional parameters from the reparameterized XML force field file.

`def write_torsion_drive_run_file(self)`

Saves a bash file for running torsion scans for torsiondrive.

Index

Functions

Classes

GuestAmberXMLAmber

analyze_diff_energies

generate_xml_antechamber

generate_xml_from_charged_pdb_sdf

generate_xml_from_doubly_charged_pdb_sdf

generate_xml_from_pdb_sdf

generate_xml_from_pdb_smi

save_amber_params

write_reparameterised_system_xml

write_system_params

HostAmberXMLAmber

analyze_diff_energies

save_amber_params

serialize_system

write_reparameterised_system_xml

write_system_params

MergeHostGuestTopology

merge_topology_files

ParameterizeGuest

get_atom_names

get_bond_angle_params

get_bond_angles

get_charges

get_hessian

get_proper_dihedrals

get_unprocessed_hessian

get_xyz

ParameterizeHost

get_atom_names

get_bond_angle_params

get_bond_angles

get_charges

get_hessian

get_unprocessed_hessian

get_xyz

PrepareGaussianGuest

get_fchk

run_gaussian

write_input

PrepareGaussianHost

get_fchk

run_gaussian

write_input

PrepareGaussianHostGuest

get_fchk

get_qm_host_guest_charges

run_gaussian

write_input

PrepareQMMM

clean_up

create_host_guest

get_guest_coord

get_host_mm_region_atoms

get_host_qm_mm_atoms

get_qm_mm_regions

get_qm_resids

realign_guest

save_host_mm_regions_pdbs

save_host_pdbs

RunOpenMMSims

run_openmm_prmtop_inpcrd

run_openmm_prmtop_pdb

TorsionDriveParams

write_reparams_torsion_lines

write_reparams_torsion_lines_charged

write_torsional_reparams

TorsionDriveSims

create_non_H_bonded_torsion_drive_dir

create_non_H_torsion_drive_dir

create_torsion_drive_dir

run_torsion_sim

write_psi4_input

write_tor_params_txt

write_torsion_drive_run_file