MANUAL

QuickFFmn 2016

by
Shaohong L. Li, and Donald G. Truhlar
Department of Chemistry and Supercomputing Institute,
University of Minnesota, Minneapolis, MN 55455-0431, USA

QuickFFmn is an extension of QuickFF1.0.1
by
Louis Vanduyfhuys, Steven Vandenbrande, Toon Verstraelen,
R. Schmid, M. Waroquier, V. Van Speybroeck
Center for Molecular Modeling, Ghent University,
Technologypark 903, 9052 Zwijnaarde, Belgium

Program version date: June 9, 2016
Documentation version date: August 24, 2016

Table of Contents
1. Introduction........................................................................................................................................ 2
2. Citation............................................................................................................................................... 2
3. Extra capabilities added in QuickFFmn 2016.................................................................................... 2
4. Additional changes in QuickFFmn 2016........................................................................................... 2
5. Installation......................................................................................................................................... 2
6. Using the extra capabilities................................................................................................................ 3
7. Test sets.............................................................................................................................................. 3
8. Sample Python code for constructing FF potential............................................................................ 3
9. Modifications to the code................................................................................................................... 4
10. Additional references......................................................................................................................... 4
1. Introduction
QuickFF (http://molmod.github.io/QuickFF) is a Python package for deriving force fields from ab initio input data. This version of QuickFF, called QuickFFmn 2016, is developed at the University of Minnesota based on QuickFF1.0.1 developed at the Ghent University, Belgium, with extra capabilities implemented.

2. Citation

3. Extra capabilities added in QuickFFmn 2016
- Simons-Parr-Finlan (SPF) potential for bond stretches,
  \[ U(R) = \frac{1}{2} k \left( \frac{R - R_e}{R} \right)^2 \]
  where \( U \) is potential energy, \( R \) is a bond length, \( k \) and \( R_e \) are parameters.
- Harmonic-cosine potential for valence bends,
  \[ U(\theta) = \frac{1}{2} k (\cos \theta - \cos \theta_e)^2 \]
  where \( U \) is potential energy, \( \theta \) is a bond angle, \( k \) and \( \theta_e \) are parameters.

4. Additional changes in QuickFFmn 2016
- Sulfur is deemed an “important” atom besides C, N, O for estimating atom types at ‘high’ level.
- Manual definition of atom connectivity can be passed via quickff.System.from_files function.

5. Installation
Install the original QuickFF1.0.1 and its dependencies (see http://molmod.github.io/QuickFF; one copy of QuickFF1.0.1 is provided), then replace the following files with the provided ones in the src_qffmn directory:
- fftable.py
- model.py
- perturbation.py
- program.py
- system.py
- terms.py
- tools.py
6. Using the extra capabilities
To use SPF and/or harmonic-cosine potential, add the following parameter(s) when calling system.determine_ics_from_topology (see sample code in Section 8):

```python
stretch_pot_kind='spf'
bend_pot_kind='harmcos'
```

7. Test sets
The following test sets are provided in the testset/ directory. (See readme.txt in the test sets for more details.)

- water_harmonic: constructing valence force field (bond stretches + bending) for water using original harmonic terms.
- water_new: constructing valence force field (bond stretches + bending) for water using newly implemented SPF and harmonic-cosine terms.

8. Sample Python code for constructing FF potential
```
from quickff import *

#--- Defining the system ---

#Read system from input files
system = System.from_files(['gaussian.fchk'])

#Guess atom types
system.guess_ffatypes('high')

#Determine internal coordinates from topology
system.determine_ics_from_topology(stretch_pot_kind='spf', bend_pot_kind='harmcos')

#--- Defining the model and program ---

model = Model.from_system(system, ai_project=True)
program = Program(system, model)

#--- Constructing the force field ---

#Estimate rest angle and multiplicity of dihedral potentials from geometry
model.val.determine_dihedral_potentials(system)

#Determine the coordinates of the perturbation trajectories
trajectories = program.generate_trajectories()

#Estimate all pars for bonds, bends and opdists
fttab = program.estimate_from_pt(trajectories)

#Refine force constants using a Hessian LSQ cost
fttab = program.refine_cost()

#--- Generating output ---
```
fftab.print_screen()
fftab.dump_ffit2('pars_ffit2.txt')
fftab.dump_yaff('pars_yaff.txt')

9. Modifications to the code
Search for #SHLL for modifications and comments in the following files:

- ffitable.py
- model.py
- perturbation.py
- program.py
- system.py
- terms.py
- tools.py

10. Additional references