# I-ReaxFF 使用手册

# Author: Guo Feng

# Email: [fengguo@lcu.edu.cn](mailto:fengguo@lcu.edu.cn) or

# [gfeng.alan@foxmail.com](mailto:gfeng.alan@foxmail.com)

目录

[1 I-ReaxFF软件安装](#_Toc23307_WPSOffice_Level1) [PAGEREF \_Toc23307\_WPSOffice\_Level1 \h4](#_Toc23307_WPSOffice_Level1)

[2 I-ReaxFF简介](#_Toc7187_WPSOffice_Level1) [PAGEREF \_Toc7187\_WPSOffice\_Level1 \h4](#_Toc7187_WPSOffice_Level1)

[3 I-ReaxFF使用参数说明](#_Toc30926_WPSOffice_Level1) [PAGEREF \_Toc30926\_WPSOffice\_Level1 \h8](#_Toc30926_WPSOffice_Level1)

[部分参数详细说明](#_Toc5850_WPSOffice_Level2) [PAGEREF \_Toc5850\_WPSOffice\_Level2 \h9](#_Toc5850_WPSOffice_Level2)

[4 I-ReaxFF使用案例](#_Toc9454_WPSOffice_Level1) [PAGEREF \_Toc9454\_WPSOffice\_Level1 \h11](#_Toc9454_WPSOffice_Level1)

[5 训练结果](#_Toc13521_WPSOffice_Level1) [PAGEREF \_Toc13521\_WPSOffice\_Level1 \h14](#_Toc13521_WPSOffice_Level1)

[参考文献：](#_Toc12962_WPSOffice_Level1) [PAGEREF \_Toc12962\_WPSOffice\_Level1 \h15](#_Toc12962_WPSOffice_Level1)

## 

## 1 I-ReaxFF软件安装

I-ReaxFF软件主要运行的操作系统为Linux系统，Window系统也能安装使用，但效率低下，不推荐。 I-ReaxFF使用python程序语言开发，少量代码使用cython开发。使用了少部分python的第三方库，首先要对这些依赖库进行安装。如NumPy, 可视化库matplotlib, ase (Atomic Simulation Environment)模块, 谷歌开源机器学习框架TensorFlow。以上模块均可通过python包管理器pip命令安装，如安装NumPy：

pip install numpy -user -upgrade

cython也可通过pip包管理器进行安装：

pip install cython -user -upgrade

在安装完以上依赖包以后，通过setup.py安装I-ReaxFF，

python setup.py install -user

## 2 I-ReaxFF简介

ReaxFF是化学反应的分子动力学力场，有着广泛的应用，如Pt-O体系的研究1,2 ，由van Duin3 于2001年开发，应用于含能材料体系的经典力学的反应性力场。ReaxFF利用键级思想，最早由Tersoff及Berenner在Tersoff势及REBO势中使用。基于键级思想，化学键在不同的化学环境中具有不同的能量，如近邻原子数，原子距离等，未修正的键级的计算公式（1-3）：

　　　　　　　 　　　　　　 （1）

　　　　　　　 （2）

　　　　　　 （3）

、、取值范围均为0～1。如图1所示。

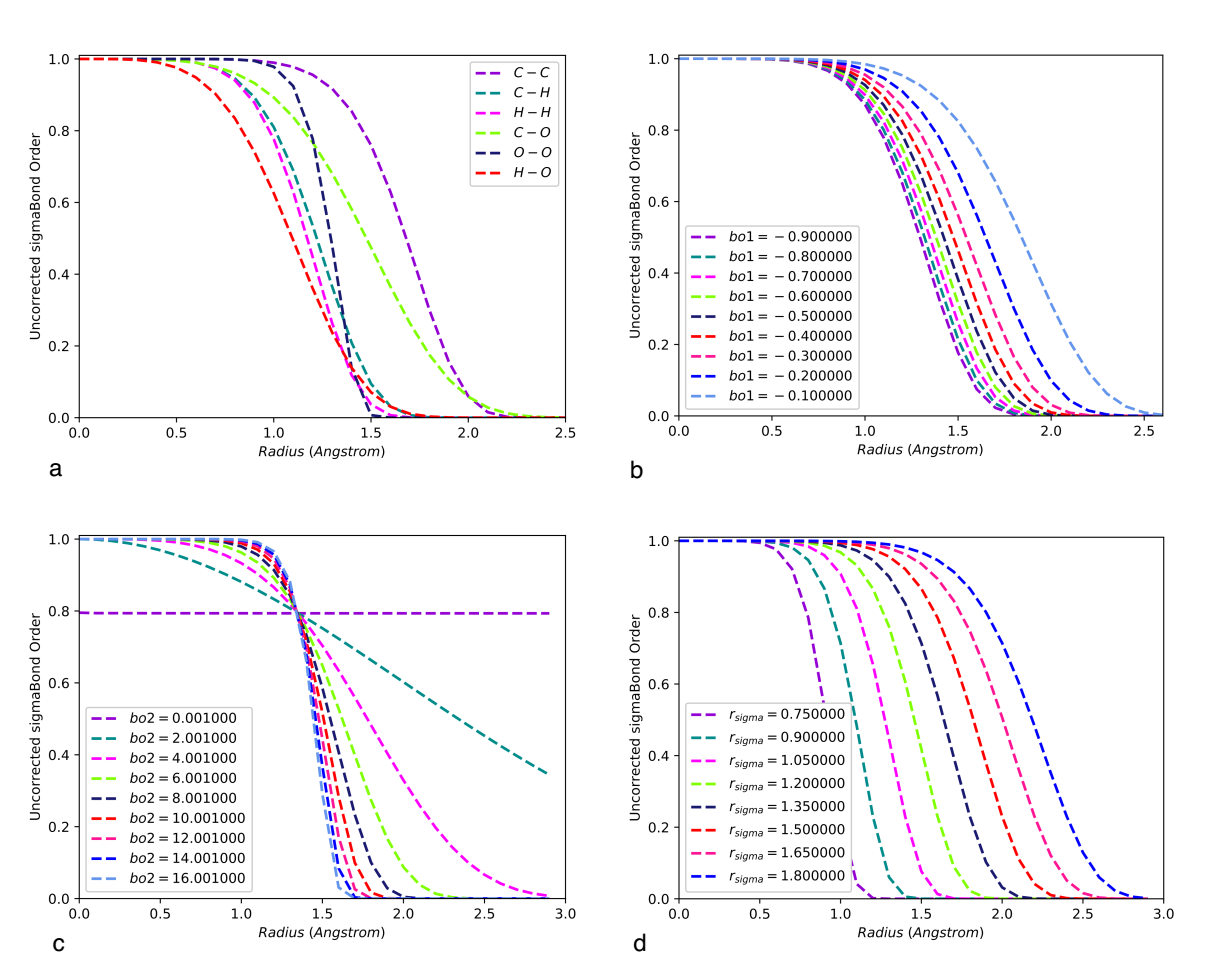


图1. 未修正键级值随r、参数、、的变化曲线。

键级值、、进一步被函数修正，如公式（4-7）所示：

(4)

(5)

(6)

(7)

公式（4-7）均为两元函数，即形成化学键的两个原子的值。值为原子所形成化学键的键级之和，由（8）式计算：

　　 　　　　　　　　　 　 　　　　　　　　　 　 (8)

在程序设计过程中，使用了大量的矩阵运算代替循环来提高程序的执行效率，如为（）的二维矩阵，而的计算可通过TensorFlow中的求和函数计算得到： = tf.reduce\_sum(,0),其中第二个参数为求和的维度。函数图象如图2所示。

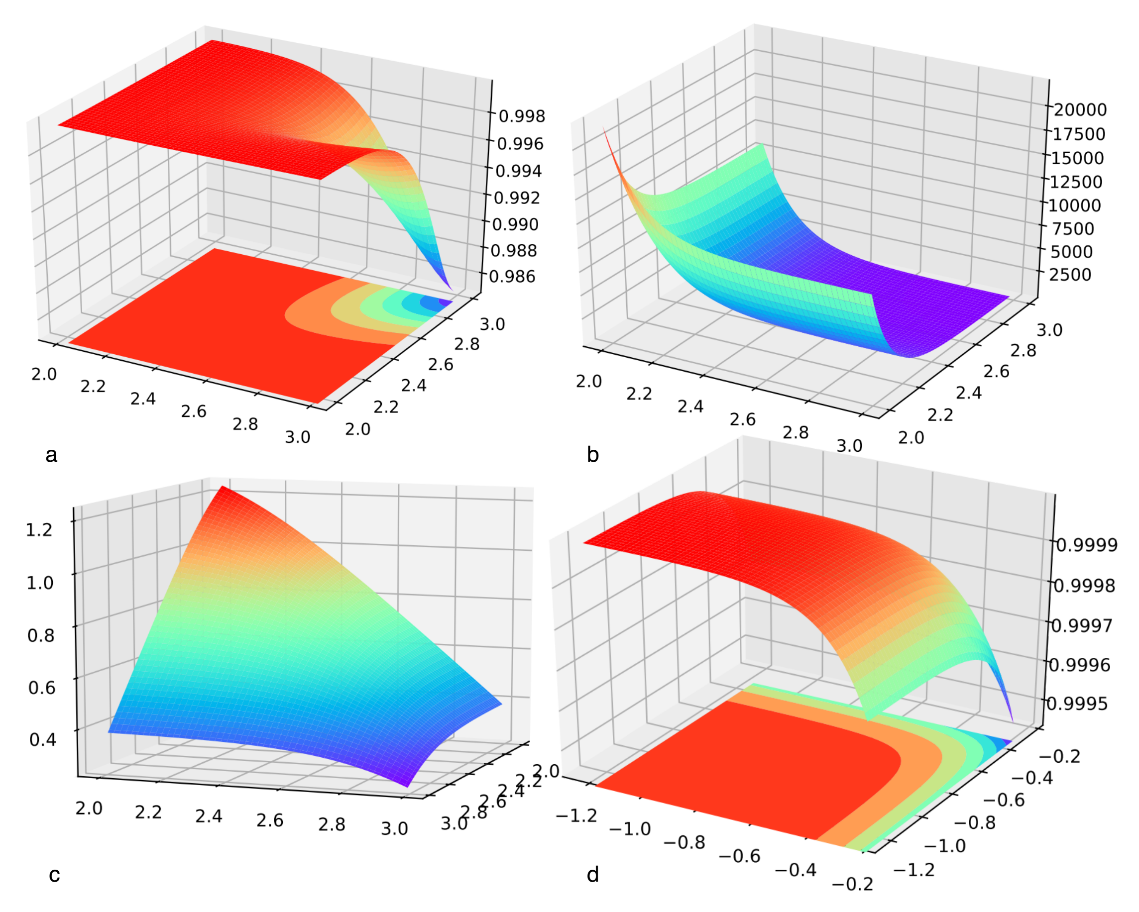
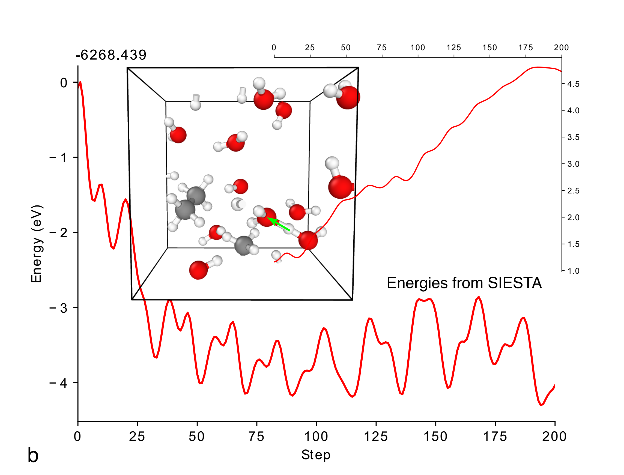
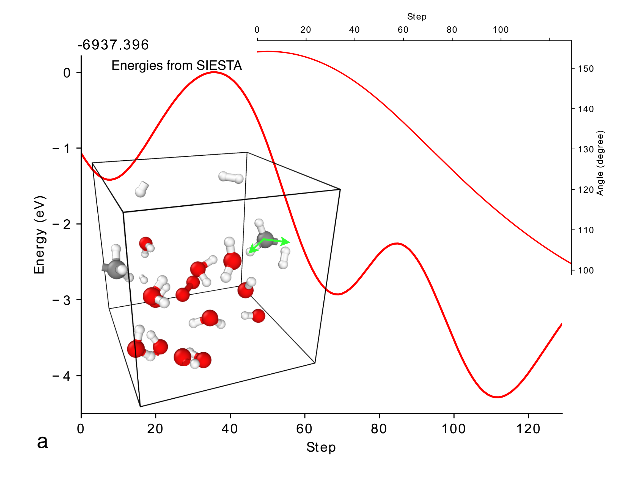


图2. 函数随成键两个原子值的变化曲面。

如正确描述化学键断裂与形成过程，需恰当的截断函数，我们采用了如式（9）所示的截断函数：

　　　　　　　　　（9）



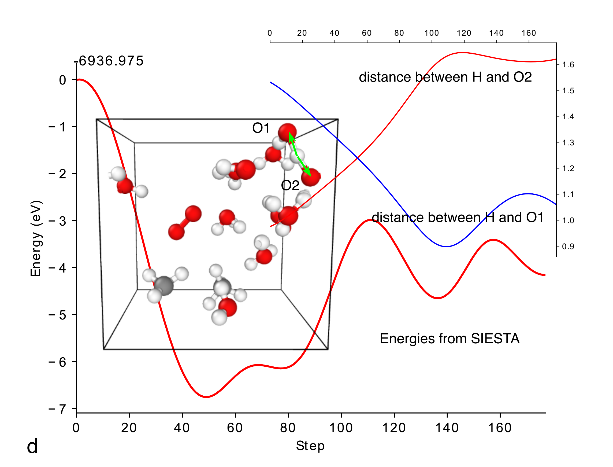
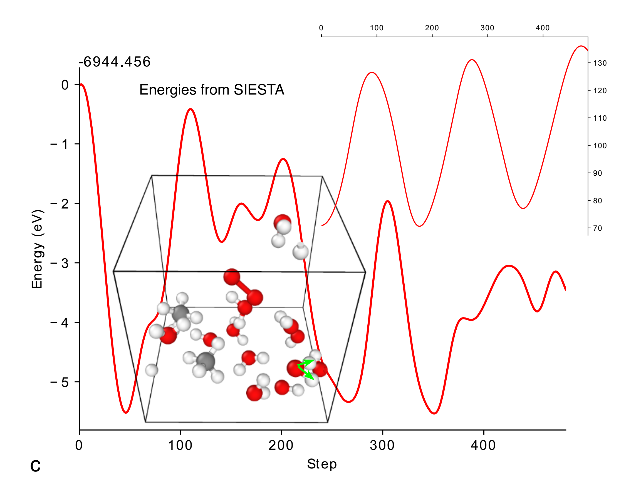


图3. 分子结构的键角、键长的变化、能量值的变化。

I-ReaxFF可自动从第一性原理动力学（AIMD）的计算结果中收集数据做为训练集，图3展示了变量的变化及能量的变化曲线。

## 3 I-ReaxFF使用参数说明

从I-ReaxFF中引入训练模块：

import irff.reax import ReaxFF

训练数据集定义：

Direcs={'nm2\_23':'/media/feng/NETAC/siesta/nm2\_23',

'nm2\_24':'/media/feng/NETAC/siesta/nm2\_24'}

Direcs中定义了两个目录文件，这两个目录均是包含SIESTA-MD (AIMD)计算结果的文件夹，定义ReaxFF对象，’ffield’为初始参数文件。一般使用如下：

rn = ReaxFF(libfile='ffield',

　 direcs=Direcs,

　 dft='siesta',

　 atomic=True,

　 bore=0.5,

　 opt=None,

　 cons=None,

　 pkl=True,

　 batch\_size=50,

　 losFunc='n2',

　 bo\_penalty=100000.0)

可使用ReaxFF对象的run模块开始训练：

rn.run(learning\_rate=1.0e-4,step=1000000,print\_step=10,writelib=1000)

所有参数均可查询源程序得到具体的含义：

class ReaxFF(object):

def \_\_init\_\_(self,libfile='ffield',direcs={},

dft='ase',atoms=None,

cons=['val','vale'],

opt=None,optword='nocoul',

batch\_size=50,sample='uniform',

hbtol=0.01,

bore=0.5,

vdwcut=10.0,

hbshort=6.75,hblong=7.5,

interactive=False,

active=False,

ro\_scale=0.1,

clip\_op=True,

rcut=None,rcuta=None,re=None,

atomic=True,

spec=['C','H','O','N'],

grad\_clip=False,

norm=5.0,

sort=False,

pkl=True,

popSize=500,

fromPop=False,

board=False,

bo\_penalty=100000.0,

to\_train=True,

optMethod='ADAM',

maxstep=60000,

emse=0.9,

zw=1.0,

convergence=0.97,

losFunc='n2',

conf\_vale=None,

huber\_d=30.0,

ncpu=None):

''' version 3.0

　　　　　　　　Time: 2018-10-20

　　　　　　　　Intelligence ReaxFF Neual Network: Evoluting the Force Field

　　　　　　　　parameters on-the-fly 2017-11-01 '''

### 部分参数详细说明

**dft：**第一性原理分子动力学数据格式，可选择siesta、cpmd、nwchem，ase。如siesta动力学结果，ase为软件Atomic Simulation Environment标准TRAJ数据格式。

**cons:** 设定列表中的变量在优化过程中保持不变。

**opt:** 设定列表中的变量将被优化（cons和opt不能同时设置）。

**optword:** 可以设置为nocoul、novdw、notor、all，

**batch\_size:** 结构、能量的个数从AIMD中抽样。

**sample:**抽样方法，可先uniform，均一抽样，random随样抽样。

**hbtol:** 形成氢键键级最小值。

**hbshort, hblong:** 氢键结构X-H...Y中X-Y长度的取值范围。

**vdwcut:** vdw相互作用的截断半径。

**rcut,rcuta,re:** rcut两两相互作用截断半径，rcuta形成共价键角截断半径，re平衡键长。

**bore:** 在平衡键长处Bond-Order的最小值，推荐值0.5，在精度提高之后可适当调小比如0.3。

**interactive：**是否开启TensorFlow互动模式，在调式模式时使用。

**clip\_op：**是否对参数取值范围进行限制，默认为True。

**ro\_scale：**是否对参数rosi取值进行限制，如0.1,即rosi\*0.9<rosi<rosi\*1.1。

**atomic:** 零点能采用原子化模式。

**spec:** 结构中的化学元素种类，也可以从ffield中获取。

**grad\_clip:** 是否进行梯度裁剪，默认为False。

**norm:** 梯度裁剪的最大范数

**bo\_penalty：**促进收敛的选项。

**losFunc：**损失函数，可选n2、abs、mse、huber, 分别对应范数2、绝对差、均方差和结合了均方差和绝对值二者特长的huber式损失函数。

## 4 I-ReaxFF使用案例

### 使用SIESTA 分子动力学方法进行数据收集

#!/usr/bin/env python

from \_\_future\_\_ import print\_function

from irff.siesta import siesta\_md,siesta\_opt

from ase.io import read,write

def md(ncpu=20,T=2000,us='F',tstep=100,dt=1.0,gen='poscar.gen'):

''' AIMD '''

if exists('siesta.MDE') or exists('siesta.MD\_CAR'):

system('rm siesta.MDE siesta.MD\_CAR')

A = read(gen,index=-1)

print('\n- running siesta md ...')

siesta\_md(A,ncpu=ncpu,T=T,dt=dt,tstep=tstep,us=us)

if \_\_name\_\_ == '\_\_main\_\_':

aimd(ncpu=20,T=2000,us='F',tstep=100,dt=1.0,gen='poscar.gen')

### 调用训练模块进行训练

#!/usr/bin/env pythonfrom irff.reax import ReaxFF

direcs={'nm1\_1':'/media/feng/NETAC/siesta/nm1\_1',

　'nm1\_2':'/media/feng/NETAC/siesta/nm1\_2',

　'nm1\_3':'/media/feng/NETAC/siesta/nm1\_3',

　'nm1\_4':'/media/feng/NETAC/siesta/nm1\_4'｝

def train():

rn = ReaxFF(libfile='ffield',

direcs=direcs,

dft='siesta',

atomic=True,

bore=0.4,

optword='nocoul',

opt=None,

cons=None,

pkl=True,

batch\_size=50,

losFunc='n2',

bo\_penalty=100000.0)

# GradientDescentOptimizer AdamOptimizer AdagradOptimizer RMSPropOptimizer

rn.sa(total\_step=1000,step=20000,astep=2000,

print\_step=10,writelib=1000)

rn.close()

if \_\_name\_\_ == '\_\_main\_\_':

train()

### 调用GULP分子动学软件使用训练过的参数进行分子动力学模拟

#!/usr/bin/env python

from \_\_future\_\_ import print\_function

from irff.gulp import write\_gulp\_in,xyztotraj

from ase.io import read,write

def nvt(T=350,time\_step=0.1,tot\_step=10000,gen='siesta.traj',index=-1,mode='w'):

''' a gulp MD run '''

A = read(gen,index=index)

write\_gulp\_in(A,runword='md qiterative conv',

T=T,

time\_step=time\_step,

tot\_step=tot\_step,

lib='reax')

print('\n- running gulp nvt ...')

system('gulp<inp-gulp>gulp.out')

xyztotraj('his.xyz',mode=mode)

if \_\_name\_\_ == '\_\_main\_\_':

nvt()

### 使用画图模块对训练中间过程进行可视化

#!/usr/bin/env python# -\*- coding: utf-8 -\*-from irff.plot.reax\_plbd import plbd

plbd(direcs={'nm2\_11':'/home/feng/siesta/nm2\_11'},

batch\_size=50,

bonds=[0,5]) # atoms index that forms bond

## 

图4.左上：未修正键级值，右上：键能值组分，左下：pow值，右下：指数值。

## 5 训练结果

### 使用I-ReaxFF训练的甲烷乙烷水合物氢键结构

Case study: CH4/CH3CH3 molecules enclosed by hydrogen bonding networks

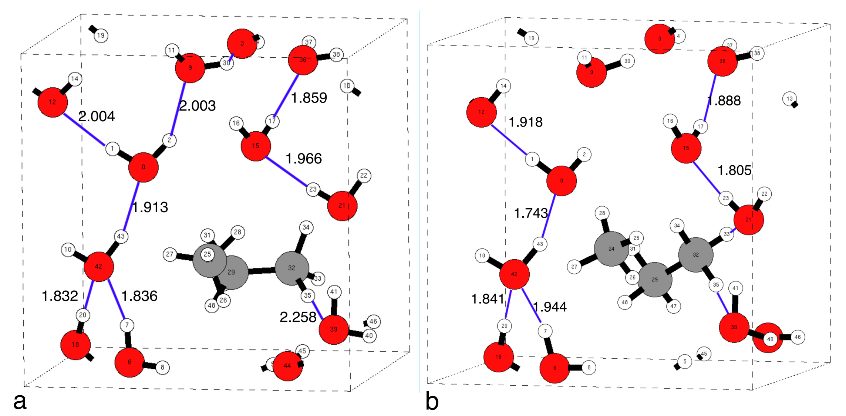


图5（a）为ReaxFF-MD结果，（b）为SIESTA-MD结果，可以看出，经过训练后的ReaxFF与SIESTA的计算结果很相近。

## 参考文献：

1. Senftle, T. P.; Hong, S.; Islam, M. M.; Kylasa, S. B.; Zheng, Y. X.; Shin, Y. K.; Junkermeier, C.; Engle-Herbert, R.; Janik, M. J.; Aktulga, H. M.; Verstraelen, A. T.; Grama, A.; van Duin, A. C. T. The ReaxFF reactive force-field: development, applications and future directions. *NPJ Comp. Mat.* 2016, 15011.

2. Donato, F.; Jochen, B.; Lehel, S.; Jonathan, E. M.; van Duin, A. C. T.; Timo, J. Development of a ReaxFF potential for Pt–O systems describing the energetics and dynamics of Pt-oxide formation. *Phys. Chem. Chem. Phys.* 2014, 16, 23118.

3. van Duin, A. C. T.; Dasgupta, S.; Lorant, F.; Goddard, W. A. ReaxFF: A Reactive Force Field for Hydrocarbons. *J. Phys. Chem. A* 2001, *105,* 9396-9409.