**DFT嵌入势能构造 V1.0**

**By 陈亦林**

1. **替换pyscf中对应文件（替换前一定要备份被替换的原文件）**

**将步骤0文件夹中的hf.py，uhf.py, rohf.py替换到****pyscf/scf/ 路径下**

**将步骤0文件夹中的rks.py，uks.py, roks.py替换到****pyscf/dft/ 路径下**

**需要注意，在替换以后，PySCF的“对程性”设置将bug，暂时还没有修**

**即，不能使用“mol.symmetry = True”**

**对称性必须一直处于关闭状态**

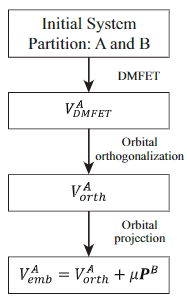
**并且所有Pyscf涉及到HF和DFT的计算，都需要在输入文件里增加“hf\_mo\_occ.txt”，“hf\_mo\_occ\_a.txt”，“hf\_mo\_occ\_b.txt”三个空文件，否则会报错，报错了自己加上对应的空文件就可以了（见下面3-2）**

**目前暂时的解决方案有两个：**

1. **备份好Pyscf中被替换前的原文件，在需要对称性计算时，把文件替换回去。（嵌入势的构造是不需要对称性的，但是平时单独体系的计算可能需要）**
2. **修bug，目前猜测在pyscf/scf/hf\_symmetry.py和pyscf/dft/rks\_symmetry.py文件中可能可以解决**
3. **总步骤**
   1. 指定系统中的“嵌入区域”原子，其余原子为“环境区域”
   2. 较大smearing width的DMFET（0.02或者其他数值，需要自己测试）
   3. 较小smearing width的DMFET（0.005），收敛完成，检查最终的molden文件的分子占据数，应该都是整数
   4. （不一定需要）对于有简并轨道的，或者在做完DMFET以后，始终存在较大的“嵌入区域”和“环境区域”轨道交叠的，要在交叠轨道空间中做正交化，因此这一步是可选的，一般不做
   5. 基于上述得到的“嵌入区域”的密度矩阵dma，和“嵌入区域”的密度矩阵dmb，构造投影算符P，选择投影项miu，完成投影和最终嵌入势的构造

**PS：在实例代码中都有非常详细的注释，请务必仔细看，这里只解释框架和基本流程，阅读时请配合代码和注释看！！！！**

1. **简单流程图**



1. **第一次计算需要怎么做**

（以乙醇为例C2H6O）

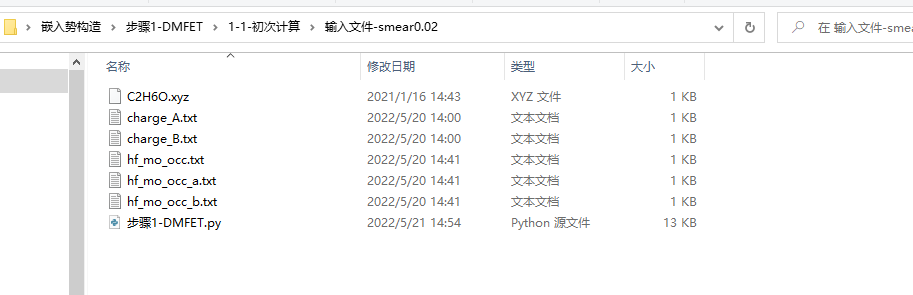
**3-1.** 打开其XYZ坐标文件，根据其行号（从0开始），指定想要的原子为“嵌入区域”



如上，我指定了0号的O原子和8号的H原子为“嵌入区域”，其余原子就自动为“环境区域”

在实例中填入[0, 8]

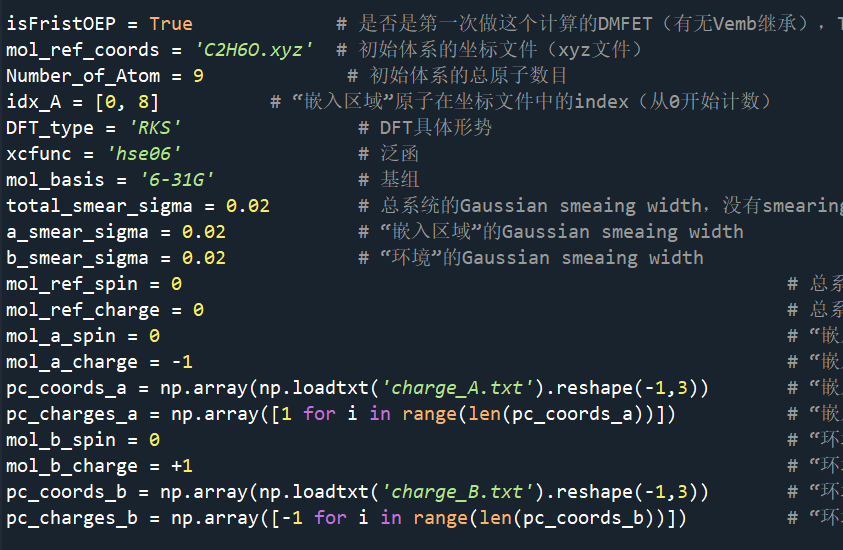
**3-2.** 在1-1的输入文件中



charge\_A.txt和charge\_B.txt，是选定区域以后，你要在嵌入区域和环境区域中加入“Capping Charges”的具体坐标(X,Y,Z)

**“hf\_mo\_occ.txt”，“hf\_mo\_occ\_a.txt”，“hf\_mo\_occ\_b.txt”不用管，是空文件，但是开始所有的PYSCF计算中都要有这三个文件名的空文件，否则会报错，报错了自己加上对应的空文件就可以了**

**3-3.** 在1-1的‘步骤1-DMFET.py’的设置



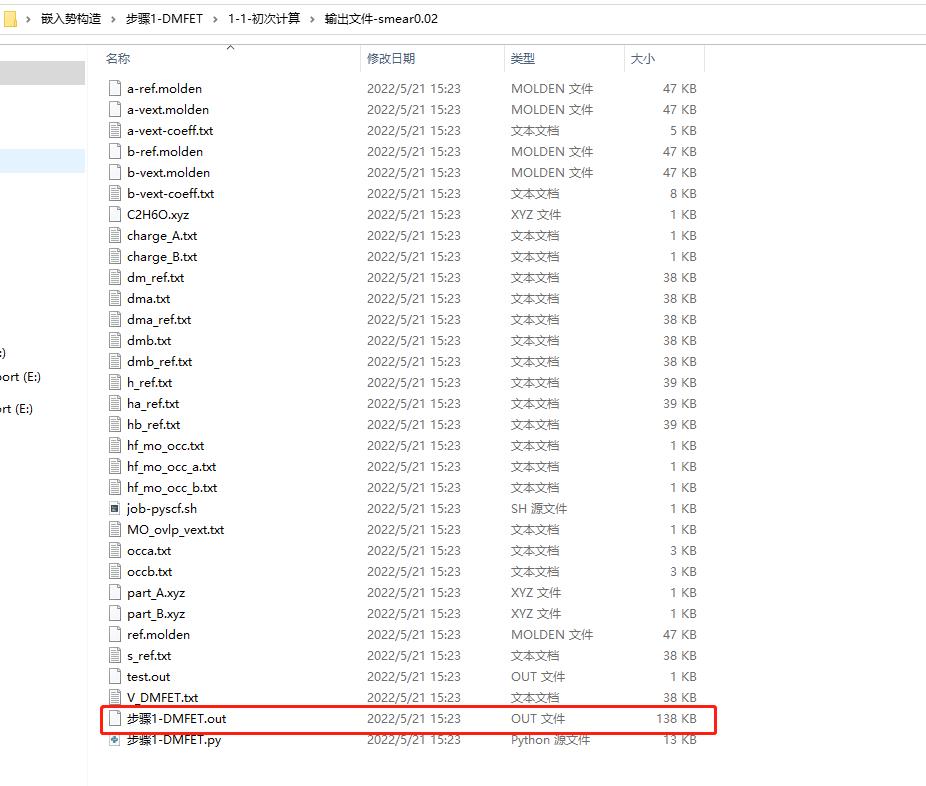
由于是第一次计算

**请将 isFristOEP设置为True**

**其余截图部分，均需要自己设置参数**

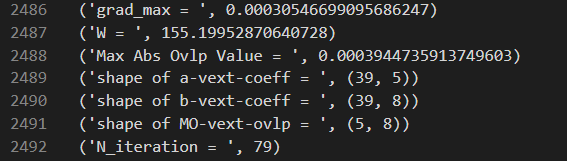
参考代码里面详细的注释来就可以了

**3-4.** 开始第一次DMFET计算，得到了1-1输出文件



先查看OUT文件，这个文件记录了OEP的迭代过程：

示意一下里面要关心的参数(代码注释里有)



Grad\_max是当前的Gradient矩阵元中的最大绝对值

W是当前的泛函数值

“Max Abs Ovlp Value”是当前“嵌入区域”与“环境区域”轨道的最大交叠数值

N\_iteration是当前OEP的迭代次数

在该计算结束后



V\_DMFET.txt就存储了当前的嵌入势Vemb

1. **续算**

见1-2

将1-1的输出文件整体copy到新的文件夹，作为新的续算输入文件

只需要改动一个地方就可以续算：

**将步骤1-DMFET.py第一行的isFirstOEP赋值为False即可**

smearing width的数值也可以自行改变：

保持原值，或者减小都可以

示例1-2是基于1-1的输出，减小了smearing width为0.005，最终完成了势能构建，收敛了。

在该计算结束后



V\_DMFET.txt就存储了当前的嵌入势Vemb

1. **轨道正交化**

轨道正交化只适用于简并轨道，或者出于一些原因，一直要保持较大smearing width的特殊情况。

由于这样的实际例子非常难找，并且代码因此具有极高的特异性，所以建议把原理搞清楚，然后一个体系一个体系的单独设计和处理

这里附上的是NV色心的一部分原代码和数据

基本思路是：

通过Reformorbitals.py重新构造出了正交化后的dma与dmb，然后以改dma为新的目标，做嵌入区域a单独的OEP

1. **轨道投影**

轨道投影是通用的

将1-2的输出文件作为新的输入文件，运行“步骤3-投影.py”即可完成投影

在该计算结束后



**Vemb\_with\_P.txt就存储了最终加了投影项的嵌入势**

**至此，DFT嵌入势全部构造完成**