file-parsing

October 16, 2023

0.1 File paring lesson

The lesson is about searching data files.

[2]:
```
ethanol_file = 'data/outfiles/ethanol.out'
print(ethanol_file)
```

```
data/outfiles/ethanol.out
```

[3]:
```
type(ethanol_file)
```

[3]:
```
str
```

[4]:
```
outfile = open(ethanol_file, 'r')
data = outfile.readlines()
outfile.close()
```

[6]:
```
# Using skills you already learned, fugure out how many lines were in the file.
# The key is remembering how many readinglines works.
number_lines = len(data)
print(number_lines)
```

```
270
```

[7]:
```
print(data)
```

```
['\n', '
-----------------------------------------------------------------------\n', '
Psi4: An Open-Source Ab Initio Electronic Structure Package\n', '
Psi4 1.1 release\n', '\n', '                         Git: Rev {HEAD} add49b9
\n', '\n', '\n', '    R. M. Parrish, L. A. Burns, D. G. A. Smith, A. C.
Simmonett,\n', '    A. E. DePrince III, E. G. Hohenstein, U. Bozkaya, A. Yu.
Sokolov,\n', '    R. Di Remigio, R. M. Richard, J. F. Gonthier, A. M. James,\n',
'    H. R. McAlexander, A. Kumar, M. Saitow, X. Wang, B. P. Pritchard,\n', '
P. Verma, H. F. Schaefer III, K. Patkowski, R. A. King, E. F. Valeev,\n', '
F. A. Evangelista, J. M. Turney, T. D. Crawford, and C. D. Sherrill,\n', '    J.
Chem. Theory Comput. in press (2017).\n', '    (doi:
10.1021/acs.jctc.7b00174)\n', '\n', '
-----------------------------------------------------------------------\n',
'\n', '\n', '    Psi4 started on: Tuesday, 27 June 2017 12:10PM\n', '\n', '
Process ID:  10591\n', '    PSIDATADIR: /Users/armcdona/psi4conda/share/psi4\n',
'    Memory:     500.0 MiB\n', '    Threads:    1\n', '    \n', '  ==> Input
File <==\n', '\n',
'--------------------------------------------------------------------------\n',
'#! You can write anything you want; this is a test\n', '\n', 'molecule ethanol
{\n', 'H        0.011242      0.005860      0.004848\n', 'C       -0.013516
-0.004055      1.099557\n', 'H        1.026838      0.005859      1.441119\n',
'H       -0.508712      0.905913      1.449714\n', 'C       -0.729377
-1.238026      1.605747\n', 'H       -1.761756     -1.258491      1.244036\n',
'H       -0.732480     -1.258491      2.699654\n', 'O       -0.054692
-2.392519      1.128673\n', 'H       -0.536299     -3.165575      1.469220\n',
'}\n', '\n', 'set basis cc-pVDZ\n', "energy('scf')------------------------------
-------------------------------------------\n", '\n', '*** tstart() called on
csm-armcdona-m1\n', '*** at Tue Jun 27 12:10:00 2017\n', '\n', '   => Loading
Basis Set <=\n', '\n', '    Name: CC-PVDZ\n', '    Role: ORBITAL\n', '
Keyword: BASIS\n', '    atoms 1, 3-4, 6-7, 9 entry H          line    20 file
/Users/armcdona/psi4conda/share/psi4/basis/cc-pvdz.gbs \n', '    atoms 2, 5
entry C          line   130 file /Users/armcdona/psi4conda/share/psi4/basis/cc-
pvdz.gbs \n', '    atoms 8              entry O          line   190 file
/Users/armcdona/psi4conda/share/psi4/basis/cc-pvdz.gbs \n', '\n', '    There are
an even number of electrons - assuming singlet.\n', '    Specify the
multiplicity in the molecule input block.\n', '\n', '\n', '
---------------------------------------------------------\n', '
SCF\n', '            by Justin Turney, Rob Parrish, and Andy Simmonett\n', '
RHF Reference\n', '                        1 Threads,    500 MiB Core\n', '
---------------------------------------------------------\n', '\n', '  ==>
Geometry <==\n', '\n', '    Molecular point group: c1\n', '    Full point group:
C1\n', '\n', '    Geometry (in Angstrom), charge = 0, multiplicity = 1:\n',
'\n', '       Center              X                  Y                   Z
Mass       \n', '    ------------   -----------------  -----------------
-----------------  -----------------\n', '           H          0.278612764252
1.265047047666    -1.274211449480     1.007825032070\n', '           C
0.253854764252     1.255132047666    -0.179502449480    12.000000000000\n', '
H          1.294208764252     1.265046047666     0.162059550520
1.007825032070\n', '           H         -0.241341235748     2.165100047666
0.170654550520     1.007825032070\n', '           C         -0.462006235748
0.021161047666     0.326687550520    12.000000000000\n', '           H
-1.494385235748     0.000696047666    -0.035023449480     1.007825032070\n', '
H         -0.465109235748     0.000696047666     1.420594550520
1.007825032070\n', '           O          0.212678764252    -1.133331952334
-0.150386449480    15.994914619560\n', '           H         -0.268928235748
-1.906387952334     0.190160550520     1.007825032070\n', '\n', '  Running in c1
symmetry.\n', '\n', '  Rotational constants: A =      1.19639  B =      0.31014
C =      0.27136 [cm^-1]\n', '  Rotational constants: A =  35866.92932  B =
9297.74675  C =   8135.03541 [MHz]\n', '  Nuclear repulsion =
81.804973313181932\n', '\n', '  Charge       = 0\n', '  Multiplicity = 1\n', '
Electrons    = 26\n', '  Nalpha       = 13\n', '  Nbeta        = 13\n', '\n', '
```

```
==> Algorithm <==\n', '\n', '  SCF Algorithm Type is DF.\n', '  DIIS
enabled.\n', '  MOM disabled.\n', '  Fractional occupation disabled.\n', '
Guess Type is SAD.\n', '  Energy threshold   = 1.00e-06\n', '  Density threshold
= 1.00e-06\n', '  Integral threshold = 0.00e+00\n', '\n', '  ==> Primary Basis
<==\n', '\n', '  Basis Set: CC-PVDZ\n', '    Blend: CC-PVDZ\n', '    Number of
shells: 36\n', '    Number of basis function: 72\n', '    Number of Cartesian
functions: 75\n', '    Spherical Harmonics?: true\n', '    Max angular momentum:
2\n', '\n', '   => Loading Basis Set <=\n', '\n', '    Name: (CC-PVDZ AUX)\n', '
Role: JKFIT\n', '    Keyword: DF_BASIS_SCF\n', '    atoms 1, 3-4, 6-7, 9 entry H
line    50 file /Users/armcdona/psi4conda/share/psi4/basis/cc-pvdz-jkfit.gbs
\n', '    atoms 2, 5          entry C          line   120 file
/Users/armcdona/psi4conda/share/psi4/basis/cc-pvdz-jkfit.gbs \n', '    atoms 8
entry O          line   220 file /Users/armcdona/psi4conda/share/psi4/basis/cc-
pvdz-jkfit.gbs \n', '\n', '  ==> Pre-Iterations <==\n', '\n', '
-------------------------------------------------------\n', '    Irrep   Nso
Nmo     Nalpha   Nbeta   Ndocc  Nsocc\n', '
-------------------------------------------------------\n', '     A         72
72       0       0       0       0\n', '
-------------------------------------------------------\n', '    Total      72
72      13      13      13       0\n', '
-------------------------------------------------------\n', '\n', '  ==>
Integral Setup <==\n', '\n', '  ==> DFJK: Density-Fitted J/K Matrices <==\n',
'\n', '    J tasked:                  Yes\n', '    K tasked:
Yes\n', '    wK tasked:                  No\n', '    OpenMP threads:
1\n', '    Integrals threads:           1\n', '    Memory (MB):
375\n', '    Algorithm:                 Core\n', '    Integral Cache:
NONE\n', '    Schwarz Cutoff:          1E-12\n', '    Fitting Condition:
1E-12\n', '\n', '   => Auxiliary Basis Set <=\n', '\n', '  Basis Set: (CC-PVDZ
AUX)\n', '    Blend: CC-PVDZ-JKFIT\n', '    Number of shells: 126\n', '
Number of basis function: 348\n', '    Number of Cartesian functions: 393\n', '
Spherical Harmonics?: true\n', '    Max angular momentum: 3\n', '\n', '  Minimum
eigenvalue in the overlap matrix is 6.1615207161E-03.\n', '  Using Symmetric
Orthogonalization.\n', '\n', '  SCF Guess: Superposition of Atomic Densities via
on-the-fly atomic UHF.\n', '\n', '  ==> Iterations <==\n', '\n', '
Total Energy        Delta E     RMS |[F,P]|\n', '\n', '   @DF-RHF iter   0:
-155.10365249375823   -1.55104e+02   4.53015e-02 \n', '   @DF-RHF iter   1:
-154.01069491612421    1.09296e+00   4.72706e-03 \n', '   @DF-RHF iter   2:
-154.08072318140574   -7.00283e-02   1.85857e-03 DIIS\n', '   @DF-RHF iter   3:
-154.08948719255255   -8.76401e-03   6.74889e-04 DIIS\n', '   @DF-RHF iter   4:
-154.09117177055145   -1.68458e-03   1.58963e-04 DIIS\n', '   @DF-RHF iter   5:
-154.09128893560870   -1.17165e-04   4.54749e-05 DIIS\n', '   @DF-RHF iter   6:
-154.09130079349944   -1.18579e-05   1.31995e-05 DIIS\n', '   @DF-RHF iter   7:
-154.09130170057145   -9.07072e-07   2.83994e-06 DIIS\n', '   @DF-RHF iter   8:
-154.09130176573018   -6.51587e-08   7.05545e-07 DIIS\n', '\n', '  ==> Post-
Iterations <==\n', '\n', '    Orbital Energies (a.u.)\n', '
-----------------------\n', '\n', '    Doubly Occupied:
\n', '\n', '       1A    -20.546800     2A    -11.275095     3A    -11.219194
\n', '       4A     -1.348904     5A     -1.008482     6A     -0.830928  \n', '
```

```
7A     -0.687787     8A     -0.642241     9A     -0.566722  \n', '      10A
-0.530314    11A     -0.519527    12A     -0.481883  \n', '      13A
-0.433845  \n', '\n', '    Virtual:
\n', '\n', '      14A      0.180940    15A      0.215018    16A      0.246070
\n', '      17A      0.251127    18A      0.267963    19A      0.292938  \n', '
20A      0.381996    21A      0.396231    22A      0.603706  \n', '      23A
0.615847    24A      0.664225    25A      0.752044  \n', '      26A
0.757291    27A      0.804155    28A      0.864841  \n', '      29A
0.872855    30A      0.889502    31A      0.896591  \n', '      32A
0.907855    33A      0.940941    34A      1.131195  \n', '      35A
1.155522    36A      1.172297    37A      1.258058  \n', '      38A
1.331160    39A      1.426371    40A      1.468358  \n', '      41A
1.517174    42A      1.567678    43A      1.599506  \n', '      44A
1.691808    45A      1.783132    46A      1.866083  \n', '      47A
1.890906    48A      1.892378    49A      1.915291  \n', '      50A
1.939375    51A      1.972358    52A      1.983727  \n', '      53A
2.003589    54A      2.100718    55A      2.199911  \n', '      56A
2.243703    57A      2.282336    58A      2.316926  \n', '      59A
2.469974    60A      2.516843    61A      2.524986  \n', '      62A
2.586064    63A      2.756948    64A      2.775491  \n', '      65A
2.808122    66A      2.899507    67A      2.915802  \n', '      68A
3.330513    69A      3.407987    70A      3.474254  \n', '      71A
3.601498    72A      4.107129  \n', '\n', '    Final Occupation by Irrep:\n', '
A \n', '    DOCC [    13 ]\n', '\n', '  Energy converged.\n', '\n', '  @DF-RHF
Final Energy:  -154.09130176573018\n', '\n', '   => Energetics <=\n', '\n', '
Nuclear Repulsion Energy =             81.8049733131819323\n', '    One-Electron
Energy =                -371.6091065733360210\n', '    Two-Electron Energy =
135.7128314944239378\n', '    DFT Exchange-Correlation Energy =
0.0000000000000000\n', '    Empirical Dispersion Energy =
0.0000000000000000\n', '    PCM Polarization Energy =
0.0000000000000000\n', '    EFP Energy =
0.0000000000000000\n', '    Total Energy =
-154.0913017657301793\n', '\n', '\n', '\n', 'Properties will be evaluated at
0.000000,   0.000000,   0.000000 Bohr\n', '\n', 'Properties computed using the
SCF density matrix\n', '\n', '  Nuclear Dipole Moment: (a.u.)\n', '     X:
-0.8398      Y:     2.6103      Z:     0.5939\n', '\n', '  Electronic Dipole
Moment: (a.u.)\n', '     X:     0.2905      Y:    -2.6019      Z:    -0.2054\n',
'\n', '  Dipole Moment: (a.u.)\n', '     X:    -0.5493      Y:     0.0084
Z:     0.3884     Total:     0.6728\n', '\n', '  Dipole Moment: (Debye)\n', '
X:    -1.3962      Y:     0.0213      Z:     0.9873     Total:     1.7101\n',
'\n', '\n', '*** tstop() called on csm-armcdona-m1 at Tue Jun 27 12:10:01
2017\n', 'Module time:\n', '\tuser time   =       1.10 seconds =       0.02
minutes\n', '\tsystem time =       0.06 seconds =       0.00 minutes\n',
'\ttotal time  =          1 seconds =       0.02 minutes\n', 'Total time:\n',
'\tuser time   =       1.10 seconds =       0.02 minutes\n', '\tsystem time =
0.06 seconds =       0.00 minutes\n', '\ttotal time  =          1 seconds =
0.02 minutes\n', '\n', '*** Psi4 exiting successfully. Buy a developer a
beer!\n']
```

[8]:

```
for line in data:
    if 'Final Energy' in line:
        energy_line = line
        print(energy_line)
```

```
  @DF-RHF Final Energy:  -154.09130176573018
```

[10]:

```
print(energy_line)
```

```
  @DF-RHF Final Energy:  -154.09130176573018
```

[11]:

```
# Now we are going to parse even more to pull out just the number

split_save = energy_line.split()
print(split_save)
```

```
['@DF-RHF', 'Final', 'Energy:', '-154.09130176573018']
```

[12]:

```
energy = split_save[3]
print(energy)
```

```
-154.09130176573018
```

[13]:

```
energy = float(energy)
```

[14]:

```
print(energy)
```

```
-154.09130176573018
```

[15]:

```
type(energy)
```

[15]:

```
float
```

[16]:

```
energy + 50
```

[16]:

```
-104.09130176573018
```

[17]:

```
import glob  #nothing happens
```

[19]:

```
file_location = 'data/outfiles/*.out'
print(file_location)
```

```
data/outfiles/*.out
```

[21]:

```
filenames = glob.glob(file_location)
```

[23]:

```python
print(filenames)
```

```
['data/outfiles/butanol.out', 'data/outfiles/decanol.out',
'data/outfiles/ethanol.out', 'data/outfiles/heptanol.out',
'data/outfiles/hexanol.out', 'data/outfiles/methanol.out',
'data/outfiles/nonanol.out', 'data/outfiles/octanol.out',
'data/outfiles/pentanol.out', 'data/outfiles/propanol.out']
```

[25]:

```python
for file in filenames:
    outfile = open(file, 'r')
    data = outfile.readlines()
    outfile.close()
    for line in data:
        if 'Final Energy' in line:
            energy_line = line
            split_save = energy_line.split()
            energy = float(split_save[3])
            print(file, energy)
```

```
data/outfiles/butanol.out -232.1655798347283
data/outfiles/decanol.out -466.3836241400086
data/outfiles/ethanol.out -154.09130176573018
data/outfiles/heptanol.out -349.27397687072676
data/outfiles/hexanol.out -310.2385332251633
data/outfiles/methanol.out -115.04800861868374
data/outfiles/nonanol.out -427.3465180082815
data/outfiles/octanol.out -388.3110864554743
data/outfiles/pentanol.out -271.20138119895074
data/outfiles/propanol.out -193.12836249728798
```

[26]:

```python
first_file = filenames[0]
print(first_file)
```

```
data/outfiles/butanol.out
```

[27]:

```python
line_split = first_file.split('/')
print(line_split)
```

```
['data', 'outfiles', 'butanol.out']
```

[28]:

```python
outfile_name = line_split[2]
print(outfile_name)
```

```
butanol.out
```

[29]:

```python
name_split = outfile_name.split('.')
print(name_split)
```

```
['butanol', 'out']
```

[30]:

```
molecule_name = name_split[0]
print(molecule_name)
```

```
butanol
```

[31]:

```
type(molecule_name)
```

[31]:

```
str
```

[41]:

```
datafile = open('energies.txt','w+')
for file in filenames:
    #get molecule name
    line_split = file.split('/')
    outfile_name = line_split[2]
    name_split = outfile_name.split('/')
    molecule_name = name_split[0]

    outfile = open(file, 'r')
    data = outfile.readlines()
    outfile.close()
    for line in data:
        if 'Final Energy' in line:
            energy_line = line
            split_save = energy_line.split()
            energy = float(split_save[3])
            datafile.write(F'{molecule_name} : {energy:4f} \n')

datafile.close()
```

[ ]:

```
```