1 Introduction

scc is the last part of the core level. It performs the self-consistent TB+U calculation and calculate the Fermi-level of the system. Even though you are not going to deal with a TB+U problem, it it still necessary to run this task because it still contains other necessary information to perform tasks in application level.

2 Dictionary

2.1 Input

scc.HubU This parameter sets the Hubbard U of each state. To give U on a state, just input the state label and its Hubbard U, e.g [1,2;2,2] means state 1 and 2 both have Hubbard U=2. Here we only consider the direct Hubbard term which reads $H_{Hub} = Un_i n_i$ no matter what basis you are using. This is equivalent to consider the F_0 Slaster integral only in full Hubbard potetial calculation. You can leave this parameter empty if you are not going to perform TB+U calculation.

scc.Charge This parameter sets the initial charge distribution of the self-consistent iteration. Just input the average charge number of each state. If no TB+U calculation is performed, the distribution doesn't matter because system will get the correct distribution after one iteration. However, their sum, i.e. the total charge, has to be correct because it affects the position of Fermi level. If there are three states, then [0.6,0.4,1] means each their initial average charge is 0.6, 0.4 and 1 respectively.

scc.Mixing This parameter sets the mixing parameter of self-consistent iterations. The relation between the new and old density matrix is: $\rho_{new} = (1 - \alpha)\rho_{in} + \alpha\rho_{out}$. scc.Mixing gives the value of α.

scc.Iteration This parameter sets the maximal number of iterations. If self-consistency is not reached after the maximal loops, PiLab will stop.

scc.Converge This parameter sets the criterion of self-consistency. Once the absolute values of the difference between output and input density matrices are smaller than it, self-consistency is reached.

scc.Mesh This parameter tells PiLab how to mesh the k-space. For example, [10,15,20] means to divide the reciprocal b_1, b_2 and b_3 into 10, 15 and 20 equal parts. The more dense of the grid, the more accurate Fermi level you will get. For insulator, you can use less dense grid. However, for metal, dense grid is needed. In most calculation, you might not care about the accuracy of Fermi level. If so, just input a very sparse grid, say [10,10,10].

scc.Temperature This parameter sets the temperature of the Fermi-Dirac distribution when calculating the Fermi-level. For insulator, it usually requires a larger value to get accurate result. This parameter doesn't mean an excitation calculation. It is just a parameter to estimate the Fermi-level. By default, PiLab sets it 100 which should be appropriate for most cases.

scc.Memory This parameter tells PiLab how to use the memory. If your scc.Mesh is very large, then the memory might not enough to store all calculation results. In this case, you will need to tune this parameter. By default, this parameter is set to 'normal'. If you find error message about memory, change it to 'max'. If it is still not working, set it to 'HDD'. HDD will use the hard disk to temporarily store calculated data which will lead the calculation very slow. Therefore, unless needed, it is strongly recommend to use a reasonable mesh, so that 'nomral' or 'max' are already enough.

2.2 Output

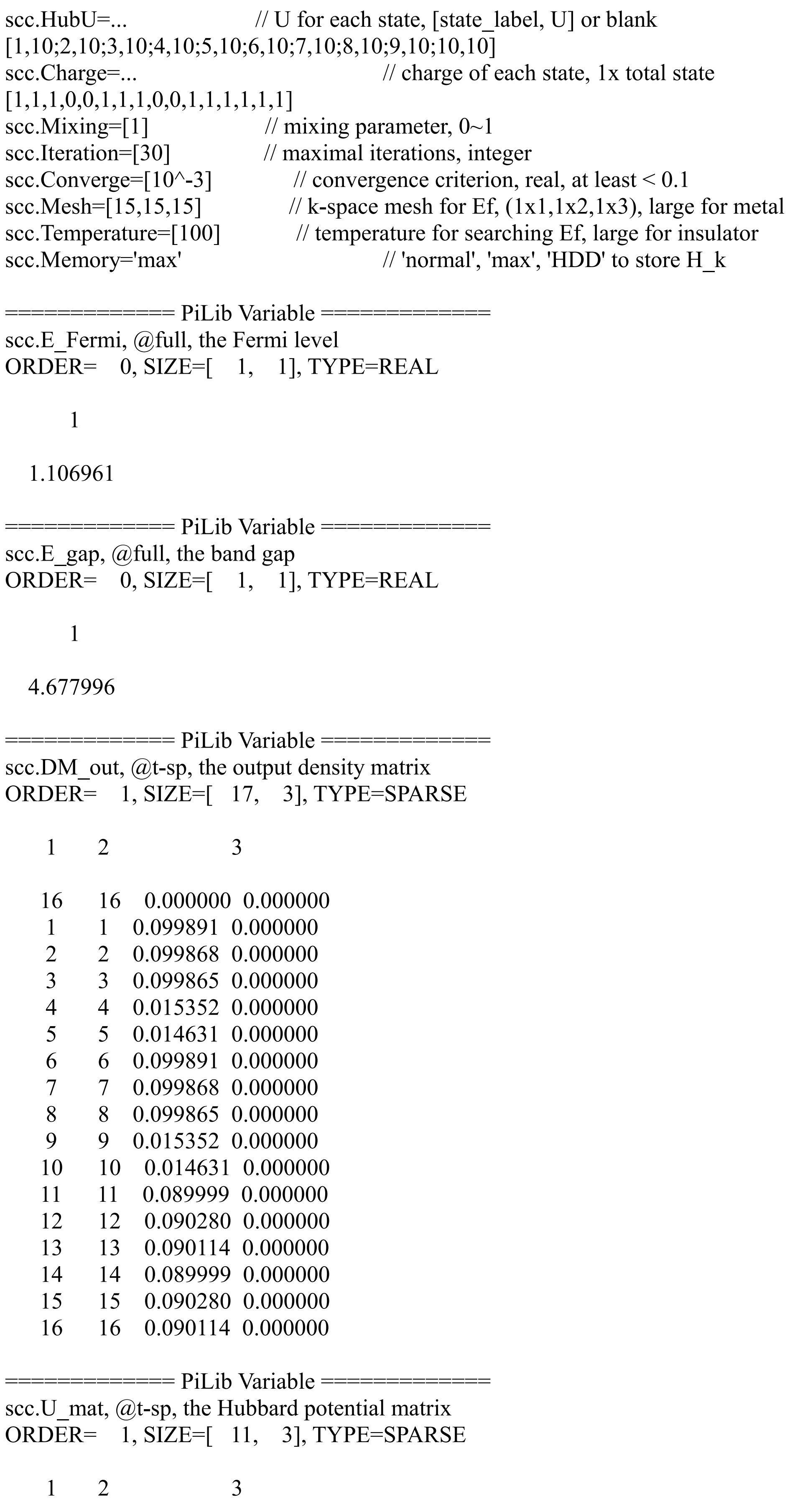
scc.E_Fermi This variable gives the Fermi level. For metal, this result is reliable only if scc.Mesh is dense enough. You can check its accuracy by observing the convergence of the Fermi level when varying scc.Mesh.

scc.E_gap This variable gives the band gap. Note that, even for metallic system, it may still gives a small gap due to enviably numerical error. The same, this result is reliable only if scc.Mesh is dense enough.

scc.DM_out This variable outputs the self-consistent density matrix. Note that, **this variable will automatically be reloaded as the initial input density matrix every time scc is performed**. Therefore, if you keep it or manually change this variable, **scc** will ignore scc.Charge and use it as initial input automatically.

scc.U_mat This variable gives the Hubbard potential. Since we only consider direct coulomb, this Hubbard potential matrix is always diagonal no matter in what basis as mentioned before.

scc.H_onsite This variable gives the full onsite Hamiltonia. $H_{onsite} = hop.onsite\_E + hop.LS\_mat + scc.U\_mat$



```
scc.HubU=...                    // U for each state, [state_label, U] or blank
[1,10;2,10;3,10;4,10;5,10;6,10;7,10;8,10;9,10;10,10]
scc.Charge=...                                  // charge of each state, 1x total state
[1,1,1,0,0,1,1,1,0,0,1,1,1,1,1,1]
scc.Mixing=[1]                  // mixing parameter, 0~1
scc.Iteration=[30]              // maximal iterations, integer
scc.Converge=[10^-3]            // convergence criterion, real, at least < 0.1
scc.Mesh=[15,15,15]             // k-space mesh for Ef, (1x1,1x2,1x3), large for metal
scc.Temperature=[100]           // temperature for searching Ef, large for insulator
scc.Memory='max'                                // 'normal', 'max', 'HDD' to store H_k

============= PiLib Variable =============
scc.E_Fermi, @full, the Fermi level
ORDER=    0, SIZE=[    1,    1], TYPE=REAL

         1

   1.106961

============= PiLib Variable =============
scc.E_gap, @full, the band gap
ORDER=    0, SIZE=[    1,    1], TYPE=REAL

         1

   4.677996

============= PiLib Variable =============
scc.DM_out, @t-sp, the output density matrix
ORDER=    1, SIZE=[   17,    3], TYPE=SPARSE

      1      2                  3

     16     16    0.000000  0.000000
      1      1    0.099891  0.000000
      2      2    0.099868  0.000000
      3      3    0.099865  0.000000
      4      4    0.015352  0.000000
      5      5    0.014631  0.000000
      6      6    0.099891  0.000000
      7      7    0.099868  0.000000
      8      8    0.099865  0.000000
      9      9    0.015352  0.000000
     10     10    0.014631  0.000000
     11     11    0.089999  0.000000
     12     12    0.090280  0.000000
     13     13    0.090114  0.000000
     14     14    0.089999  0.000000
     15     15    0.090280  0.000000
     16     16    0.090114  0.000000

============= PiLib Variable =============
scc.U_mat, @t-sp, the Hubbard potential matrix
ORDER=    1, SIZE=[   11,    3], TYPE=SPARSE

      1      2                  3
```

Figure 1: page 1 of NiO_scc.plb

```
    16      16    0.000000  0.000000
     1       1   -0.498900  0.000000
     2       2   -0.498700  0.000000
     3       3   -0.498700  0.000000
     4       4    0.346500  0.000000
     5       5    0.353700  0.000000
     6       6   -0.498900  0.000000
     7       7   -0.498700  0.000000
     8       8   -0.498700  0.000000
     9       9    0.346500  0.000000
    10      10    0.353700  0.000000

============= PiLib Variable =============
scc.H_onsite, @t-sp, H_onsite(hop.onsite_E+hop.LS_mat+scc.U_mat)
ORDER=    1, SIZE=[    17,     3], TYPE=SPARSE

     1       2                    3

    16      16    0.000000  0.000000
     1       1   -0.798900  0.000000
     2       2   -0.798700  0.000000
     3       3   -0.798700  0.000000
     4       4    0.346500  0.000000
     5       5    0.353700  0.000000
     6       6   -0.798900  0.000000
     7       7   -0.798700  0.000000
     8       8   -0.798700  0.000000
     9       9    0.346500  0.000000
    10      10    0.353700  0.000000
    11      11   -0.300000  0.000000
    12      12   -0.300000  0.000000
    13      13   -0.300000  0.000000
    14      14   -0.300000  0.000000
    15      15   -0.300000  0.000000
    16      16   -0.300000  0.000000
```

Figure 2: page 2 of NiO_scc.plb