DMFTwDFT Tutorial

Hyowon Park

May 2019

1 Introduction

This DMFTwDFT program performs DFT+DMFT calculations combining various DFT codes. Our code is based on the papers of XXX. Our code can perform k-point paralellization calculations using MPI. And currently the vasp code is modified to perform charge-self-consistent DFT+DMFT calculations and the wannier projection part is improved by implementing the k-point parallelization (KPAR=number of cores).

This DMFTwDFT program mainly computes the local Green's function (G loc.out) and the hybridization function (Delta.out) from the DMFT self energy (sig.inp). For a complete DMFT self-consistent calculation, one must solve the Andson impurity problem using an impurity solver. Our code is interfaced with the ctqmc impurity solver developed by Prof. Kristjan Haule in Rutgers University. The ctqmc code can be downloaded from http://hauleweb.rutgers.edu/tutorials/Tutorial0.html with tutorials.

2 DFT+DMFT calculation procedure

2.1 DFT calculation

First, one should perform a typical DFT self-consistent calculation using a DFT code which is interfaced with wannier90 code. Here, I will show an example of the VASP code, but it can be generalized to arbitrary DFT codes including Siesta. One should also note that the DFT calculation must be a non-spin-polarized calculation (ISPIN=1 in VASP) even when one wants to compute spin-polarized DFT+DMFT calculations. This is because we want to have a non-spin-polarized Wannier calculation for DFT+DMFT.

2.2 Wannier90 calculation

Once you obtain the converged charge density and Kohn-Sham wavefunction from DFT, the next step is to perform the maximally localized Wannier function (MLWF) calculation using wannier90 code. The details of how to use the code can be found in www.wannier.org. Here, one should generate the DMFT

subspace of the MLWFs by using the "Projection" technique within a certain energy window.

For example, in the $LaNiO_3$ case with the rhombohedral structure, the wannier90.win file can be constructed as follows.

```
dis win min = -0.3014
dis win max = 10.6986
num wann = 28
num iter = 100

begin projections
f=0.00000000,0.00000000,0.00000000:l=2:z=-0.29712138,0.77011009,0.56449032:x=-
0.52491861,-0.62540726,0.57692379
f=0.50000000,0.50000000,0.50000000:l=2:z=0.29712138,0.77011009,-0.56449032:x=0.52491861,-
0.62540726,-0.57692379
O:p
end projections
```

Here, the energy window is specified by [dis win min : dis win max], and it is usually determined from the band structure. In this case, Ni d and O p bands are entangled in a range of -8eV to 3eV. Since the Fermi energy from a DFT calculation is 7.6986eV, the energy window is chosen as above. The number of wannier functions is 28 since we have 10 $d-$orbial and 18 $p-$orbtal wannier functions.

The projection orbitals are needed for the initial guess of Wannier orbitals and, for Ni $d-$orbitals, one needs to choose the projection axis to be aligned to the local Ni-O octahedron axis. This is because we want to minimize the off-diagonal terms of the Hamiltonian in the $d-$orbital basis, since we are using the ctqmc impurity solver and the calculation of off-diagonal terms will be very inefficient. One can use "generate win.py" file to obtain this projection axis. Using this win file, one should converge the maximal localization of wannier obitals and obtain wannier90.chk and wannier90.eig files.

2.3 DMFT+DFT calculation

The next step is to perform DMFT+DFT calculations using the output files of DFT+wannier calculations. Copy -input.py file will copy necessary output files for the inputs of DMFT+DFT runs. :
python Copy-input.py path-to-DFT-folder

For non-charge-self-consistent calculations (Niter=1), DFT-mu.out, DMFT-mu.out, INPUT.py, OSZICAR, RUNDMFT.py, sig.inp, wannier90.chk, wannier90.eig files are needed. For charge-self-consistent runs (Niter>1), INCAR, KPOINTS, OUTCAR, POSCAR, POTCAR, wannier90.win, wannier90.amn, WAVECAR files are additionally needed.

The input parameters for DFT+DMFT runs will be stored in INPUT.py file. After you have all input files from DFT runs, now you need to generate

self-energy file, sig.inp. If you already have this from the previous run, you can copy from it. Otherwise, one can have zero self-energy by generating using sigzero.py file.

Once you have all input files, you can excute the program using RUNDMFT.py file. During the run, dmft.x, ctqmc, and dft excutables will be run using mpi. Therefore one should put the mpi commands in para-com.dat. For example, one can put the following line in submit script.
echo mpirun -machinefile PBS-NODEFILE -n XX > para-com.dat

3 Input parameters

Here, I explain input parameters for DMFT+DFT runs.

3.1 Niter

Niter means the total loop of DFT+DMFT.

3.2 Nit

Nit means the total loop of DMFT self-consistent calculations.

3.3 Ndft

Ndft means the total loop of DFT self-consistent calculations.

3.4 n-tot

n-tot means the total number of electrons in the Wannier subspace. For exmaple, in LaNiO3, we have 2 Ni ions with 7 d-electrons/Ni and 6 O ions with 6 p-electrons/O, therefore totally 50 electrons.

3.5 nf

nf means the possible occupancy of d- or f- electrons in a correlated atom. This is used for the initial guess of self-energy. One can put the DFT occupancy of the correlated atom or the nominal electron number.

3.6 nspin

nspin means the number of spins in DMFT calculations. nspin=2 means the spin-polarized calculation. Although nspin=2, the DFT calculation must be non-spin-polarized calculation.

3.7 atomnames

atomnames means the name of atoms where the Wannier projection will be taken.

3.8 orbs

orbs means the name of Wannier orbitals in each atom in atomnames.

3.9 L rot

L rot means whether the wannier projection axis will be rotated along the local axis. 1: rotated, 0: non-rotated.

3.10 cor at

cor at is the list of all correlated atoms in the material. For LaNiO3, it is Ni1 and Ni2.

3.11 cor orb

cor orb is the list of all correlated orbitals in each correlated atom.

3.12 U

U is the local Hubbard interaction on cor at.

3.13 J

J is the Hund's coupling.

3.14 alpha

alpha is the double counting correction parameter. alpha=0 means the conventional double counting in the fully localized limit.

3.15 mix-sig

mix-sig is the mixing parameter between the previous and the current self energies.

3.16 q

q is the number of k-points while doing DMFT self-consistent calculations. We are using the Wannier interpolation technique, therefore large numbers of q-points will be possible. Usually 2-3 times larger than DFT k-points will be necessary for better convergence.

3.17 ewin

ewin is the energy window for Wannier projection with respect to the DFT fermi energy.

4 Output files

The main output files of DFT+DMFT runs will be INFO-ITER, INFO-ENERGY, INFO-KSUM, INFO-DM, INFO-TIME. The examples of LaNiO3 runs for both N.C.S.C(Niter=1) and C.S.C(Niter>1) can be found in run-exmaples directory.

4.1 INFO-ITER

INFO-ITER records all iteration information necessary for monitoring convergence. For example, LaNiO3 DMFT calculation (Niter=1) will show something like below.

```
1 10 7.798655 7.797084 1.669025 1.644641 -68.440528 -68.086689 0.000000
1 11 7.798642 7.797683 1.668703 1.644087 -68.444524 -68.088970 0.000000
1 12 7.798855 7.796977 1.669338 1.644704 -68.446219 -68.090861 0.000000
1 13 7.798939 7.797447 1.669546 1.644125 -68.446738 -68.096304 0.000000
1 14 7.798821 7.797382 1.669375 1.644365 -68.443074 -68.092242 0.000000
1 15 7.798739 7.797317 1.669131 1.644268 -68.442809 -68.091764 0.000000
```

To check the convergence of DFT+DMFT calculation, one must check if the local lattice quantity and the impurity quantity are converging (getting equal). Here, the first number is total iteraction steps and the second number is DMFT iteration steps. The third and fourth number compares the d-occupancy from local lattice calculaton and the impurity calcualtion of ctqmc. The fifth and sixth numbers compare the self-energy at w=oo - Vdc for lattice and impurity. The seventh and eighth nubmers are the total energy computed using the Migdal-Galisky method and the ctqmc sampling. The last number is the charge difference between two consecutive steps.

If you perform charge-self-consistent calculations, you will see something like below.

```
22 1 8.013450 8.008304 1.497393 1.475141 -68.094131 -67.725218 0.004445
23 1 8.013099 8.008603 1.497078 1.474757 -68.081961 -67.715811 0.006589
24 1 8.012837 8.009002 1.496692 1.474657 -68.072933 -67.715758 0.011620
25 1 8.012751 8.009157 1.497266 1.474415 -68.067251 -67.713353 0.011801
26 1 8.012565 8.008943 1.497044 1.474825 -68.064955 -67.714023 0.006227
27 1 8.012501 8.009220 1.497517 1.474323 -68.065745 -67.714654 0.007282
28 1 8.012436 8.009272 1.497843 1.474266 -68.067957 -67.714704 0.010110
29 1 8.012442 8.009222 1.498471 1.474300 -68.067595 -67.716171 0.012198
30 1 8.012360 8.008898 1.498468 1.474642 -68.068845 -67.713375 0.009332
```

Now you can see that we performed total iteration of 30 steps with 1 DMFT step. The d-occupancy is increased to near 8.0 and the last number is updated for the charge difference.

5 Post-processing tools

5.1 analytic continuation - maximum entropy

Since the ctqmc impurity solver samples the self energy (sig.inp) on the imaginary axis, one should perform the analytic continuation to obtain the self energy on the real axis. Here, we use the maximum entropy method to perform this analytic continuation. The source file can be compiled in post-tools/maxent-source directory.

The procedure of performing max-ent is as follows.

1. One should compile the maxent-source codes and copy *.so files and *.py files to the bin directory. And one should make an empty directory for max-ent calculation maybe inside the DMFT run directory.
2. Copy a few last self energy data sig.inp.XXX to the directory.
3. Run "sigaver.py" to take average of the self energies. ex) sigaver.py sig.inp.*
4. One should copy maxent-params.dat file from the source directory.
5. Now perform "maxent-run.py sig.inpx". The analytically continued self-energy will be stored at "Sig.out" file.

5.2 density of states

Once you get the self energy on the real axis (Sig.out), you are now ready to compute the density of states and band structure using DMFT.

1. One can make a new empty directory and copy necessary files. One can "Copy-input.py path-of-DMFT-results -dos"
2. Now one should make sig.inp-real file from Sig.out of maxent result. One can copy Sig.out and do "./Interpol-sig-real.py"
3. Now run dmft-dos.x to obtain G-loc.out on the real axis. Ex) mpirun -n XX dmft-dos.x

6 Charge-self-consistent calculations interfaced to DFT+wannier90 codes

This DMFTwDFT package can be easily interfaced to any DFT codes which are implementing wannier90 calculations. As the Wannier functions are used as basis functions for DMFT regardless of the specific DFT basis sets, our DFT+DMFT calculations can be efficiently interfaced to various DFT codes. Our DMFTwDFT code has a library mode (similar to wannier90 code) such that any DFT codes can call the fortran subroutine to obtain the necessary information to update charge density within DFT+DMFT. Specifically, one can pass the k-points information within DFT to the subroutine Compute-DMFT and can obtain DMFT weight and Unitary matrix at each k-point for computing the charge density $\rho(r)$.

The formalism for computing charge density within DFT+DMFT is explained briefly below. The details of this derivation and the application to the PAW formalism can be found in Phys. Rev. B 90, 235103 (2014). The charge density $\rho(r)$ within DFT+DMFT can be given by

$$\rho(\mathbf{r}) = \sum_{i \notin W} \rho_i^{DFT}(\mathbf{r}) + \sum_{i,j \in W} \rho_{ij}^{DMFT}(\mathbf{r}) \tag{1}$$

where ρ^{DMFT} is the charge density within the DMFT subspace, W is the energy window for constructing Wannier functions, and i,j are the band indices.

ρ^{DMFT} can be also represented using DFT KS wavefunctions as follows.

$$\rho_{ij}^{DMFT}(\mathbf{r}) = \frac{1}{N_{\mathbf{k}}} \sum_{\mathbf{k}} n_{\mathbf{k}ij} \langle \psi_{\mathbf{k}i}^{KS} | \mathbf{r} \rangle \langle \mathbf{r} | \psi_{\mathbf{k}j}^{KS} \rangle, \tag{2}$$

where $n_{\mathbf{k}ij}$ is the DMFT occupancy matrix in the KS basis and $|\psi_{\mathbf{k}j}^{KS}\rangle$ is the KS wavefunction at momentum $\mathbf{k}$ and band i.

The sum over band indices i, j in Eq. 2 can be simplified to the sum over one index because the occupancy matrix n_{ij} is Hermitian and so can be written in terms of eigenvalues $w_{\mathbf{k}\lambda}$ and eigenfunctions ϕ_λ as

$$n_{\mathbf{k}ij} = \sum_{\lambda} U_{\mathbf{k}i\lambda}^{DMFT} \cdot w_{\mathbf{k}\lambda} \cdot U_{\mathbf{k}j\lambda}^{DMFT*} \tag{3}$$

where $U_{\mathbf{k}}^{DMFT}$ are unitary matrices whose rows are ϕ_λs. Using this eigendecomposition, the wavefunction ψ^{KS} is unitarily transformed to $\psi_{\mathbf{k}\lambda}^{DMFT}$ given by

$$\langle \mathbf{r} | \psi_{\mathbf{k}\lambda}^{DMFT} \rangle \quad = \quad \sum_{i} \langle \mathbf{r} | \psi_{\mathbf{k}i}^{KS} \rangle \cdot U_{\mathbf{k}i\lambda}^{DMFT}. \tag{4}$$

And finally, the charge density $\rho^{DMFT}(r)$ can be represented using the DMFT weights and DMFT wavefunctions using the subroutine to compute the charge density within any DFT codes.

$$\rho^{DMFT}(\mathbf{r}) = \frac{1}{N_{\mathbf{k}}} \sum_{\mathbf{k},\lambda} w_{\mathbf{k}\lambda} \langle \psi^{DMFT}_{\mathbf{k}\lambda} | \mathbf{r} \rangle \langle \mathbf{r} | \psi^{DMFT}_{\mathbf{k}\lambda} \rangle, \tag{5}$$

6.1 Compute-DMFT

Compute-DMFT subroutine has the following structure with input and output parameters. You can find this fortran file dmft lib.F90 in the src directory. The example of using this library can be found in library-mode-test directory.

subroutine Compute DMFT(n kpts loc,n wann,kpt dft,wght dft,band win loc,DMFT eval,DMFT evec)

integer, intent(in) :: n kpts loc,n wann
- n kpts loc is a variable for the number of k-points in DFT (It can be either k-points in IBZ or full BZ).
- n wann is the number of wannier orbitals (size of Wannier Hamiltonian).

real(kind=dp), intent(in) :: kpt dft(3,n kpts loc)
- kpt dft is the list of k-points in DFT using fractional coordinates.

real(kind=dp), intent(in) :: wght dft(n kpts loc)
- wght dft is the weight of each k-point in BZ. The sum of weights should be one.

integer,intent(out) :: band win loc(2,n kpts loc)
- band win loc is the band range of Wannier subspace W at each k-point. This will be needed for computing charge density within the subspace.

real(kind=dp), intent(out) :: DMFT eval(n wann,n kpts loc)
- DMFT eval is the eigenvalues $w_{\mathbf{k}\lambda}$ of the DMFT occupancy matrix $n_{\mathbf{k}ij}$.

complex(kind=dp), intent(out) :: DMFT evec(n wann,n wann,n kpts loc)
- DMFT evec is the DMFT eigenvectors $U^{DMFT}_{\mathbf{k}i\lambda}$.