Dynamical Mean Field Theory for strongly correlated electrons

TOOLbox series, nov 2021

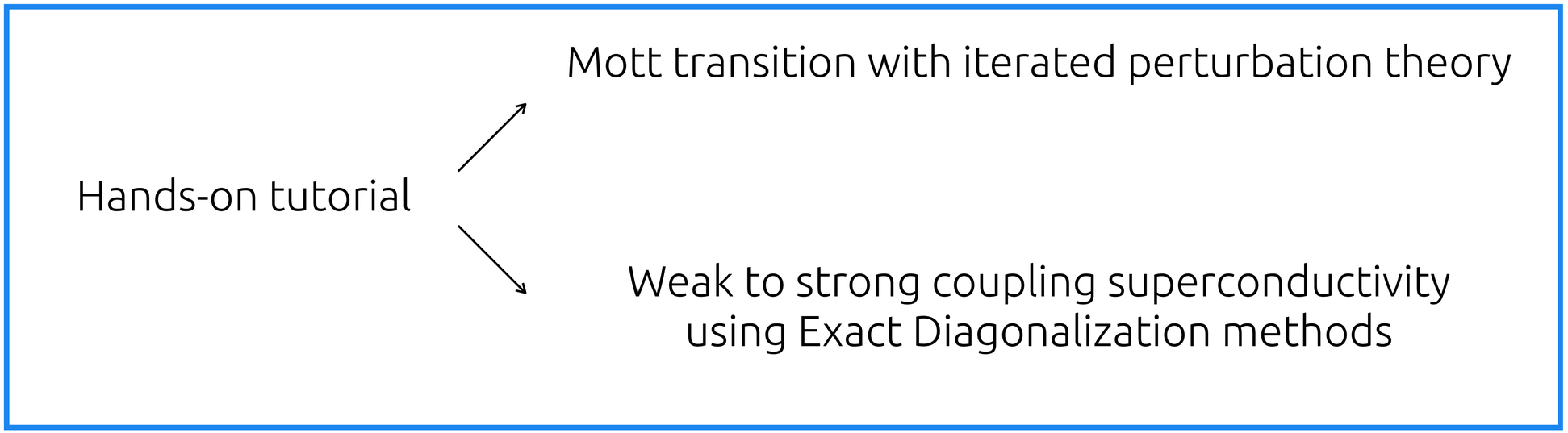
Giacomo Mazza
DQMP, Unige

Monday 22

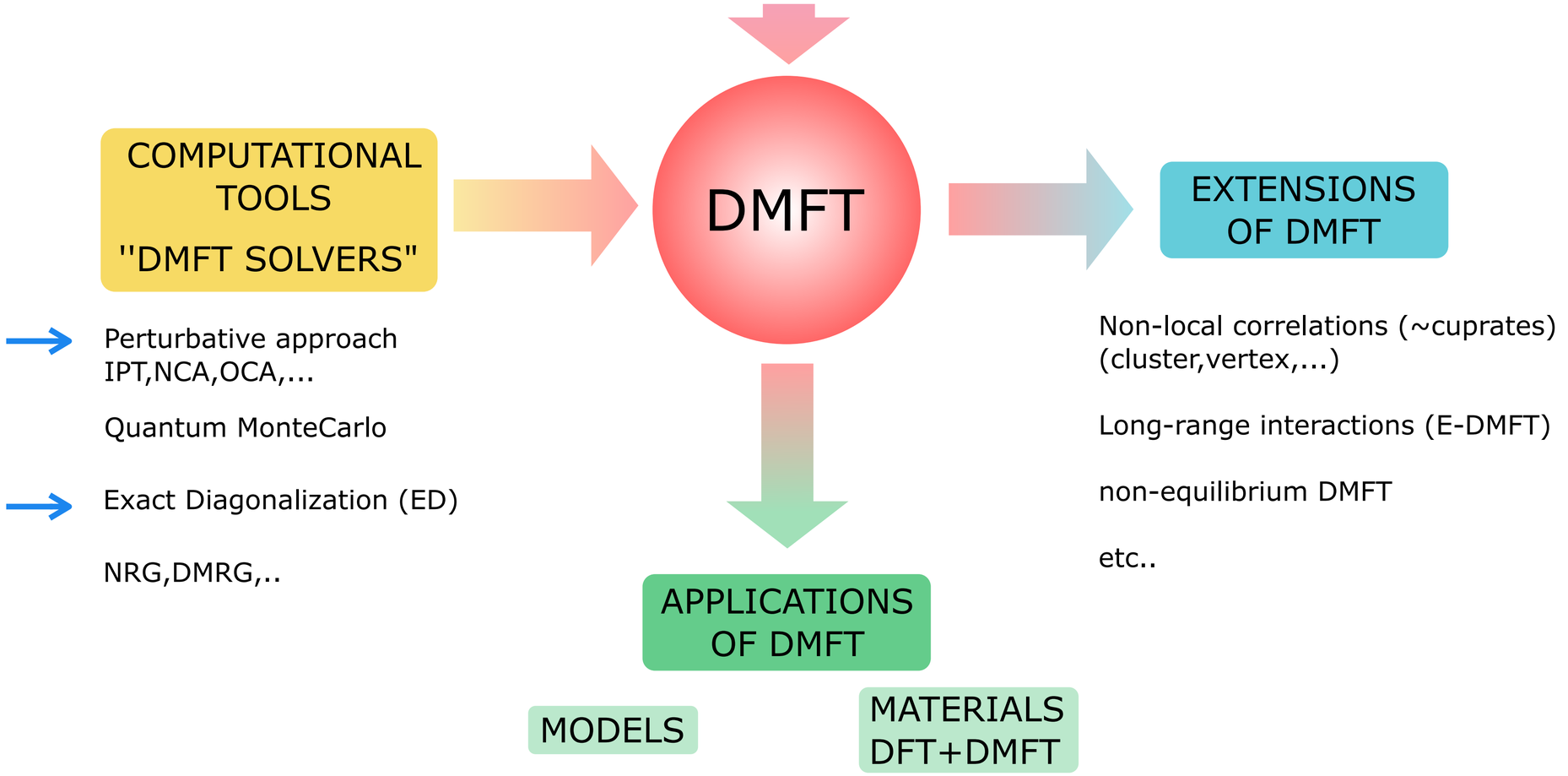
Introduction to the basics concepts of the Dynamical Mean Field Theory

(instruction for the hands-on tutorial)

Tuesday 23



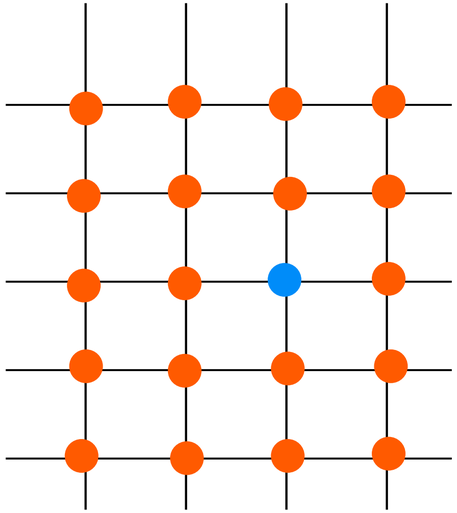
General concepts of statistic physics
(single site representation, limit of infinite dimensions,..)



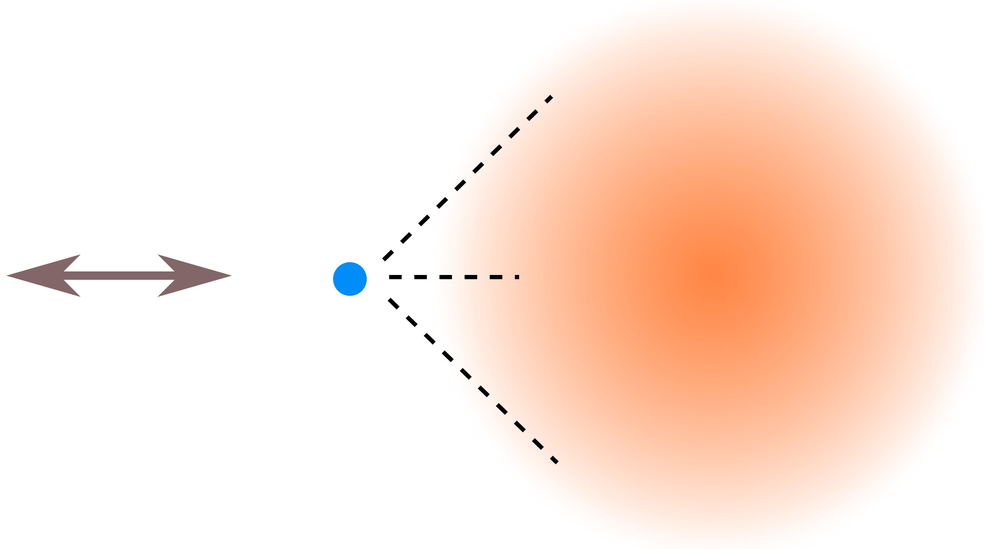
Adapted from A. Georges
Fermions en interaction: Introduction a la theorie du champ moyen dynamique. College de France, 2019 (online)

$$H_{\text{Hubbard}} = -t \sum_{\langle \mathbf{R}\mathbf{R}' \rangle \sigma} c^{\dagger}_{\mathbf{R}\sigma} c_{\mathbf{R}'\sigma} + U \sum_{\mathbf{R}} n_{\mathbf{R}\uparrow} n_{\mathbf{R}\downarrow} - \mu \sum_{\mathbf{R}\sigma} c^{\dagger}_{\mathbf{R}\sigma} c_{\mathbf{R}\sigma},$$

Lattice problem



Single-site representation



Compare with static mean-field

$$H_{\text{Ising}} = -J \sum_{\langle ij \rangle} \sigma_i \sigma_j - \sum_i h_i \sigma_i$$

	STATIC MeanField	DYNAMICAL MeanField
LOCAL OBSERVABLE	local magnetization $m_i = \langle \sigma_i \rangle$	local Greens function $G_{ii}(\tau) = -\langle T_\tau c_{i\sigma}(\tau) c^\dagger_{i\sigma}(0) \rangle$
LOCAL REPRESENTATION	spin in a magnetic field $m_i = \tanh\left(\beta h_i^{\text{eff}}\right)$	atom in a bath $G^{-1}_{ii}(\tau) = \mathcal{G}_0^{-1}(\tau) - \Sigma_{\text{imp}}[\mathcal{G}_0]$
SELF CONSISTENCY RELATION	$h_i^{\text{eff}} = h_i + J \sum_{\langle ij \rangle} m_j$	$\Sigma_{\mathbf{k}}(\omega) = \Sigma_{imp}(\omega)$

$$\mathcal{S}^{(i)}_{\rm imp} = \int d\tau d\tau' c^\dagger_{i\sigma} \mathcal{G}_0^{-1}(\tau-\tau') c_{i\sigma}(\tau') + \int d\tau H^{(i)}_{int}(\tau) \qquad H^{(i)}_{int} = U n_{i\uparrow} n_{i\downarrow}$$

$$\mathcal{G}_0^{-1}(\tau-\tau') = G^{-1}_{at,0}(\tau-\tau') - \Delta(\tau-\tau'). \qquad G_{at,0}(i\omega_n) = i\omega_n + \mu$$

$$G_{ii,\sigma}(\tau-\tau') = -\langle T_\tau c_{i\sigma}(\tau) c^\dagger_{i\sigma}(\tau') \rangle_{\mathrm{imp}[\mathcal{G}_0]}$$

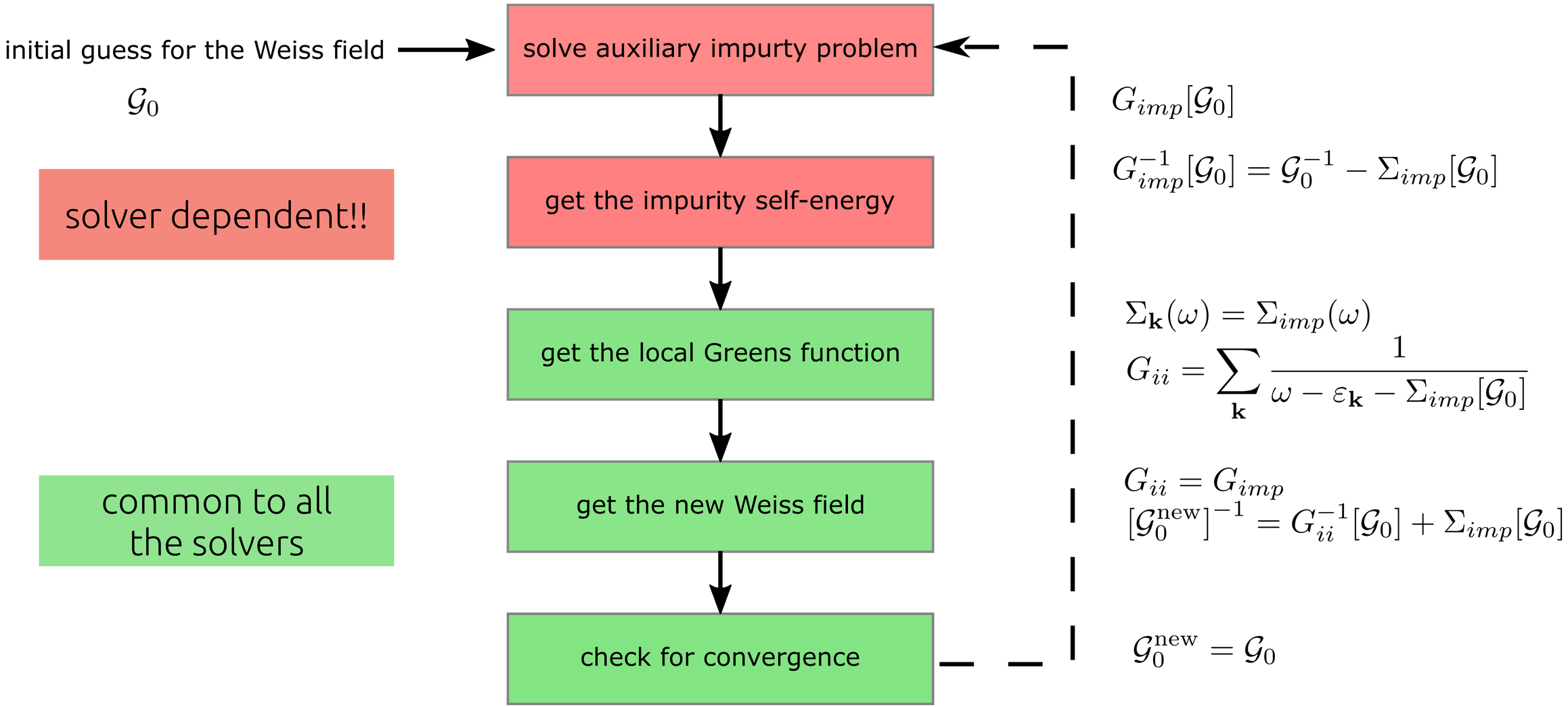
$$G^{-1}_{ii\sigma}(\tau-\tau') = \mathcal{G}_0^{-1}(\tau-\tau') - \Sigma_{\rm imp}[\mathcal{G}_0]$$

DMFT approximation

$$\boxed{\Sigma_{\mathbf{k}} = \Sigma_{\rm imp}[\mathcal{G}_0]}$$

exact for $z \to \infty$

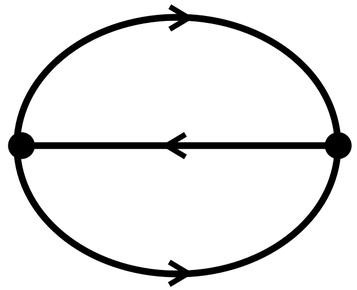
$$G_{ii}(\omega) = \sum_{\mathbf{k}} \frac{1}{\omega - \varepsilon_{\mathbf{k}} - \Sigma_{\mathbf{k}}(\omega)}$$


initial guess for the Weiss field
$\mathcal{G}_0$
solver dependent!!
common to all the solvers
solve auxiliary impurty problem
get the impurity self-energy
get the local Greens function
get the new Weiss field
check for convergence
$G_{imp}[\mathcal{G}_0]$
$G^{-1}_{imp}[\mathcal{G}_0] = \mathcal{G}_0^{-1} - \Sigma_{imp}[\mathcal{G}_0]$
$\Sigma_{\mathbf{k}}(\omega) = \Sigma_{imp}(\omega)$
$G_{ii} = \sum_{\mathbf{k}} \frac{1}{\omega - \varepsilon_{\mathbf{k}} - \Sigma_{imp}[\mathcal{G}_0]}$
$G_{ii} = G_{imp}$
$[\mathcal{G}_0^{\rm new}]^{-1} = G^{-1}_{ii}[\mathcal{G}_0] + \Sigma_{imp}[\mathcal{G}_0]$
$\mathcal{G}_0^{\rm new} = \mathcal{G}_0$

Iterated Perturbation Theory (IPT)

Approximate solution of the impurity problem

$$\Sigma_{\mathrm{imp}}(\tau) = U^2 \mathcal{G}_0(\tau)\mathcal{G}_0(-\tau)\mathcal{G}_0(\tau)$$

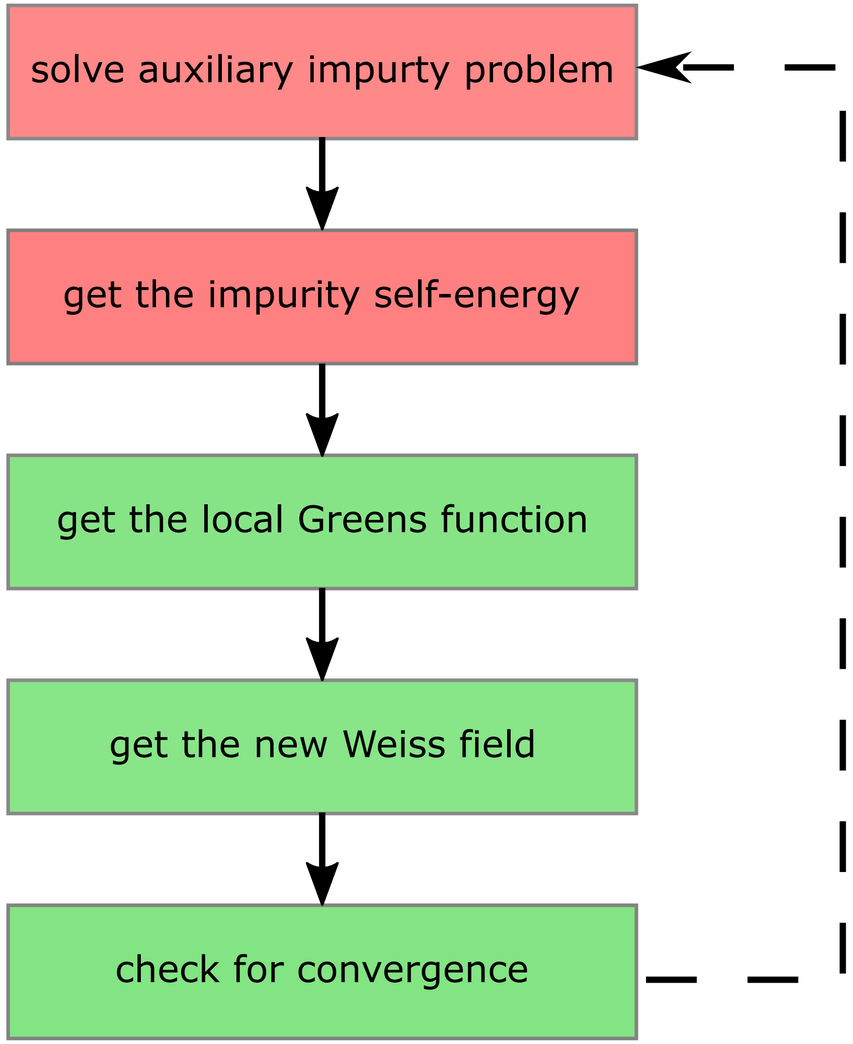


- Mott transition in the single-band Hubbard model
- Useful to get used with the DMFT loop
- Imaginary frequency or real frequency
- Little practical use beyond the single-band Hubbard model

ipt_hm_matsubara.f90

wm(:) !Matsubara frequencies
Sigma(:) !impurity Self-energy
fg0(:) !Weiss field at loop iloop
fg0_prev(:) !Weiss field at loop iloop-1
fg(:) !local greens function

```
!get or read first sigma
call  get_initial_function(Sigma,"Sigma.restart")
!dmft loop:
iloop=0 ; converged=.false.
do while(.not.converged.AND.iloop<nloop)
   iloop=iloop+1
   write(*,"(A,i5)",advance="no")"DMFT-loop",iloop
   !SELF-CONSISTENCY:
   do i=1,L
      zeta  = xi*wm(i) - sigma(i)
      fg(i) = gfbethe(wm(i),zeta,D)
   enddo
   !
   fg0_prev= fg0                        !
   fg0     = one/(one/fg + sigma)
   if(iloop>1)fg0 = wmix*fg0 + (1.d0-wmix)*fg0_prev !mix to avoid loops
   !
   !IMPURITY SOLVER: fg0-->sigma
   call solve_ipt_mats(sigma)
   !
   dens = ipt_measure_dens_matsubara(fg)
   z    = ipt_measure_zeta_matsubara(sigma,fg0)
   docc = ipt_measure_docc_matsubara(sigma,fg0)
   write(*,"(3F15.9,3x)",advance="no")dens,docc,z
   !Check CONVERGENCE on the Weiss Field:
   converged=check_convergence(fg0)
enddo
```

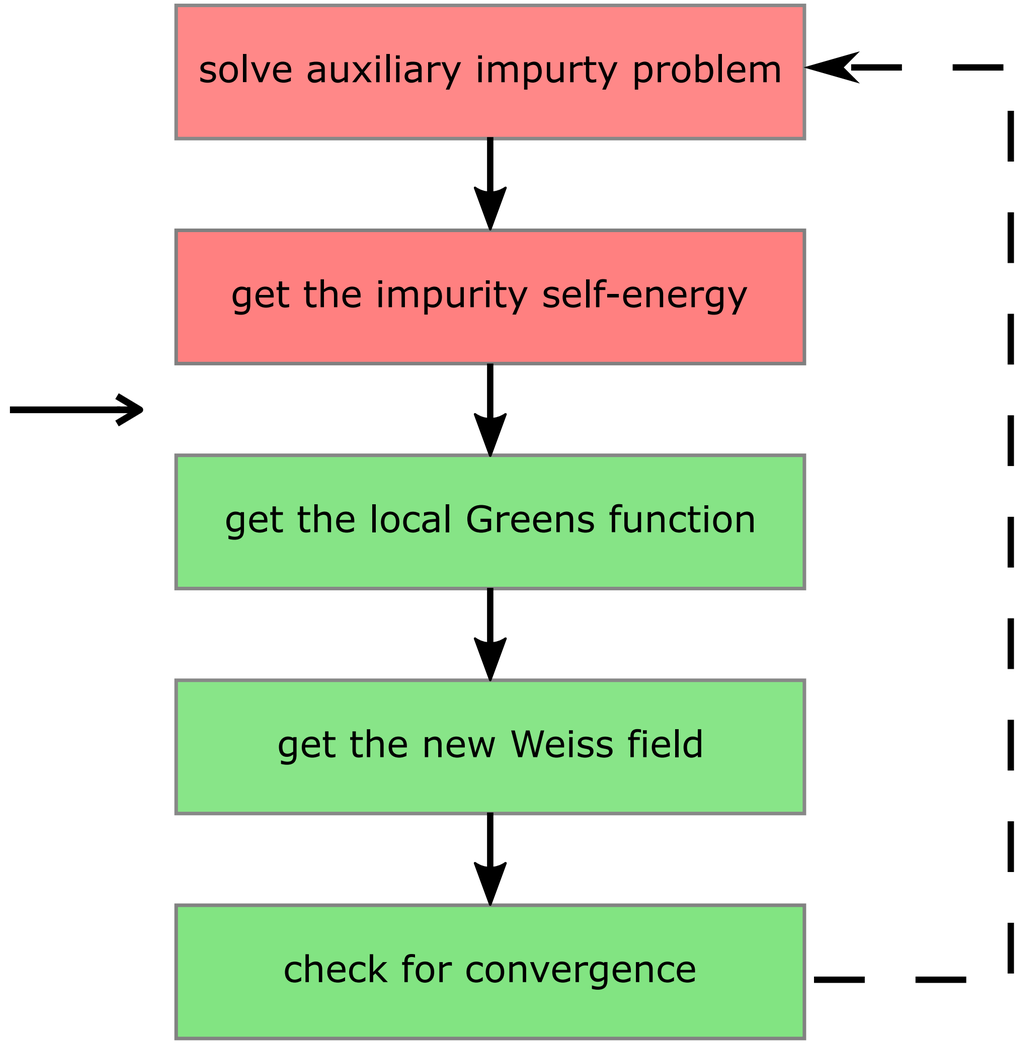


ipt_hm_matsubara.f90

```
wm(:)        !Matsubara frequencies
Sigma(:)     !impurity Self-energy
fg0(:)       !Weiss field at loop iloop
fg0_prev(:)  !Weiss field at loop iloop-1
fg(:)        !local greens function
```

Initialise the loop

```
!get or read first sigma
call  get_initial_function(Sigma,"Sigma.restart")
!dmft loop:
iloop=0 ; converged=.false.
do while(.not.converged.AND.iloop<nloop)
   iloop=iloop+1
   write(*,"(A,i5)",advance="no")"DMFT-loop",iloop
   !SELF-CONSISTENCY:
   do i=1,L
      zeta  = xi*wm(i) - sigma(i)
      fg(i) = gfbethe(wm(i),zeta,D)
   enddo
   !
   fg0_prev= fg0                        !
   fg0     = one/(one/fg + sigma)
   if(iloop>1)fg0 = wmix*fg0 + (1.d0-wmix)*fg0_prev !mix to avoid loops
   !
   !IMPURITY SOLVER: fg0-->sigma
   call solve_ipt_mats(sigma)
   !
   dens = ipt_measure_dens_matsubara(fg)
   z    = ipt_measure_zeta_matsubara(sigma,fg0)
   docc = ipt_measure_docc_matsubara(sigma,fg0)
   write(*,"(3F15.9,3x)",advance="no")dens,docc,z
   !Check CONVERGENCE on the Weiss Field:
   converged=check_convergence(fg0)
enddo
```



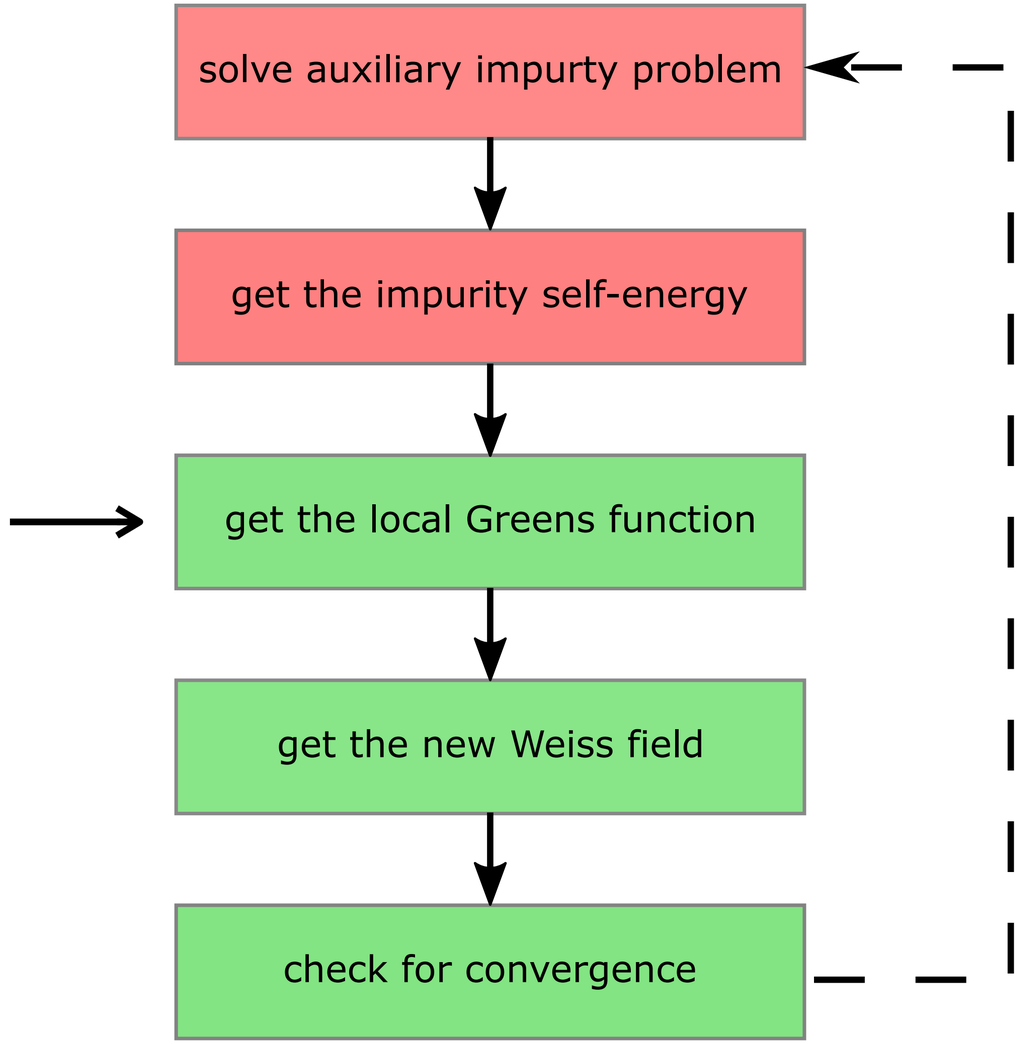
ipt_hm_matsubara.f90

wm(:) !Matsubara frequencies
Sigma(:) !impurity Self-energy
fg0(:) !Weiss field at loop iloop
fg0_prev(:) !Weiss field at loop iloop-1
fg(:) !local greens function

```
!get or read first sigma
call  get_initial_function(Sigma,"Sigma.restart")
!dmft loop:
iloop=0 ; converged=.false.
do while(.not.converged.AND.iloop<nloop)
   iloop=iloop+1
   write(*,"(A,i5)",advance="no")"DMFT-loop",iloop
   !SELF-CONSISTENCY:
   do i=1,L
      zeta  = xi*wm(i) - sigma(i)
      fg(i) = gfbethe(wm(i),zeta,D)
   enddo
   !
   fg0_prev= fg0                          !
   fg0     = one/(one/fg + sigma)
   if(iloop>1)fg0 = wmix*fg0 + (1.d0-wmix)*fg0_prev !mix to avoid loops
   !
   !IMPURITY SOLVER: fg0-->sigma
   call solve_ipt_mats(sigma)
   !
   dens = ipt_measure_dens_matsubara(fg)
   z    = ipt_measure_zeta_matsubara(sigma,fg0)
   docc = ipt_measure_docc_matsubara(sigma,fg0)
   write(*,"(3F15.9,3x)",advance="no")dens,docc,z
   !Check CONVERGENCE on the Weiss Field:
   converged=check_convergence(fg0)
enddo
```

Local greens function

$$G(i\omega_n) = \int d\epsilon \frac{\rho_{\rm bethe}(\epsilon)}{i\omega_n - \epsilon - \Sigma(i\omega)}$$



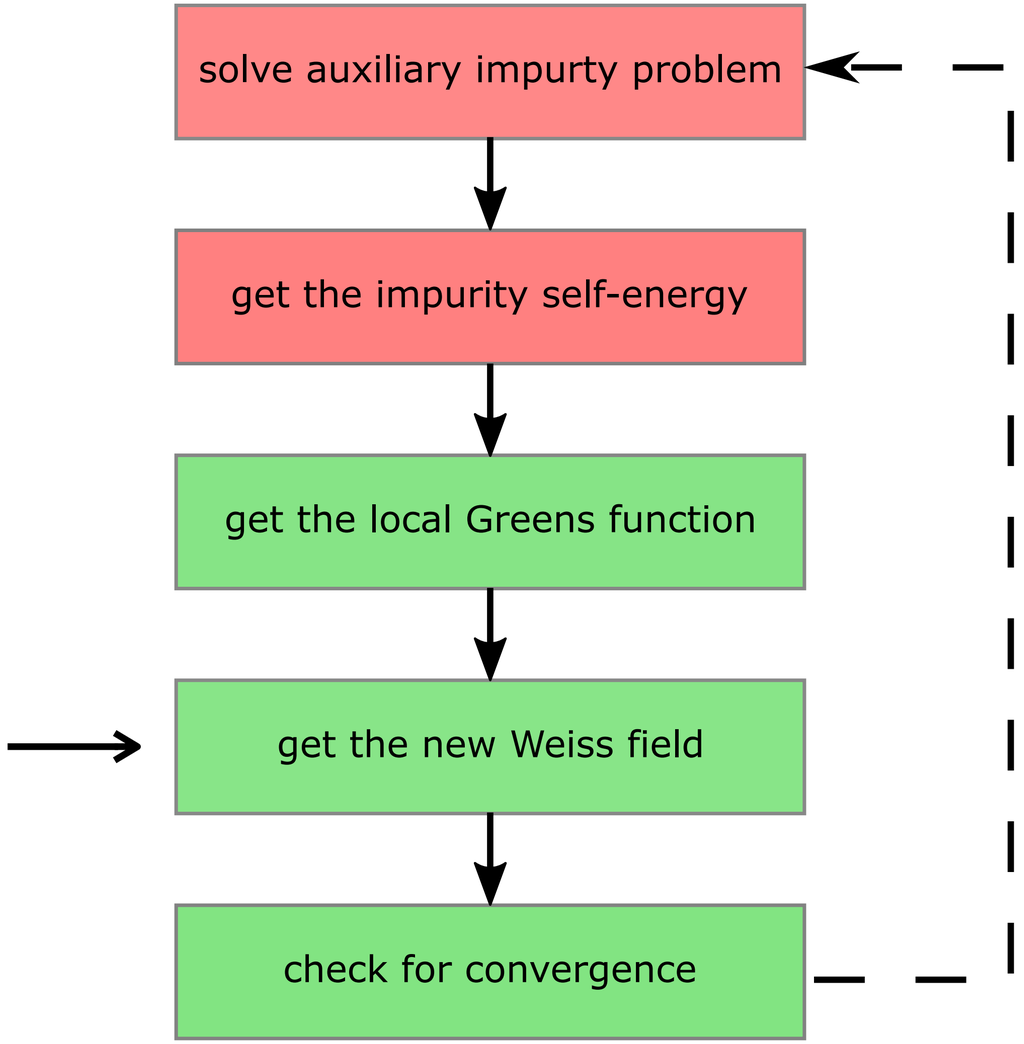
ipt_hm_matsubara.f90

wm(:) !Matsubara frequencies
Sigma(:) !impurity Self-energy
fg0(:) !Weiss field at loop iloop
fg0_prev(:) !Weiss field at loop iloop-1
fg(:) !local greens function

```
!get or read first sigma
call  get_initial_function(Sigma,"Sigma.restart")
!dmft loop:
iloop=0 ; converged=.false.
do while(.not.converged.AND.iloop<nloop)
   iloop=iloop+1
   write(*,"(A,i5)",advance="no")"DMFT-loop",iloop
   !SELF-CONSISTENCY:
   do i=1,L
      zeta  = xi*wm(i) - sigma(i)
      fg(i) = gfbethe(wm(i),zeta,D)
   enddo
   !
   fg0_prev= fg0                          !
   fg0     = one/(one/fg + sigma)
   if(iloop>1)fg0 = wmix*fg0 + (1.d0-wmix)*fg0_prev !mix to avoid loops
   !
   !IMPURITY SOLVER: fg0-->sigma
   call solve_ipt_mats(sigma)
   !
   dens = ipt_measure_dens_matsubara(fg)
   z    = ipt_measure_zeta_matsubara(sigma,fg0)
   docc = ipt_measure_docc_matsubara(sigma,fg0)
   write(*,"(3F15.9,3x)",advance="no")dens,docc,z
   !Check CONVERGENCE on the Weiss Field:
   converged=check_convergence(fg0)
enddo
```

$$\mathcal{G}_0^{\text{new}}(i\omega_n) = \frac{1}{G^{-1}(i\omega_n) + \Sigma(i\omega_n)}$$

new Weiss field

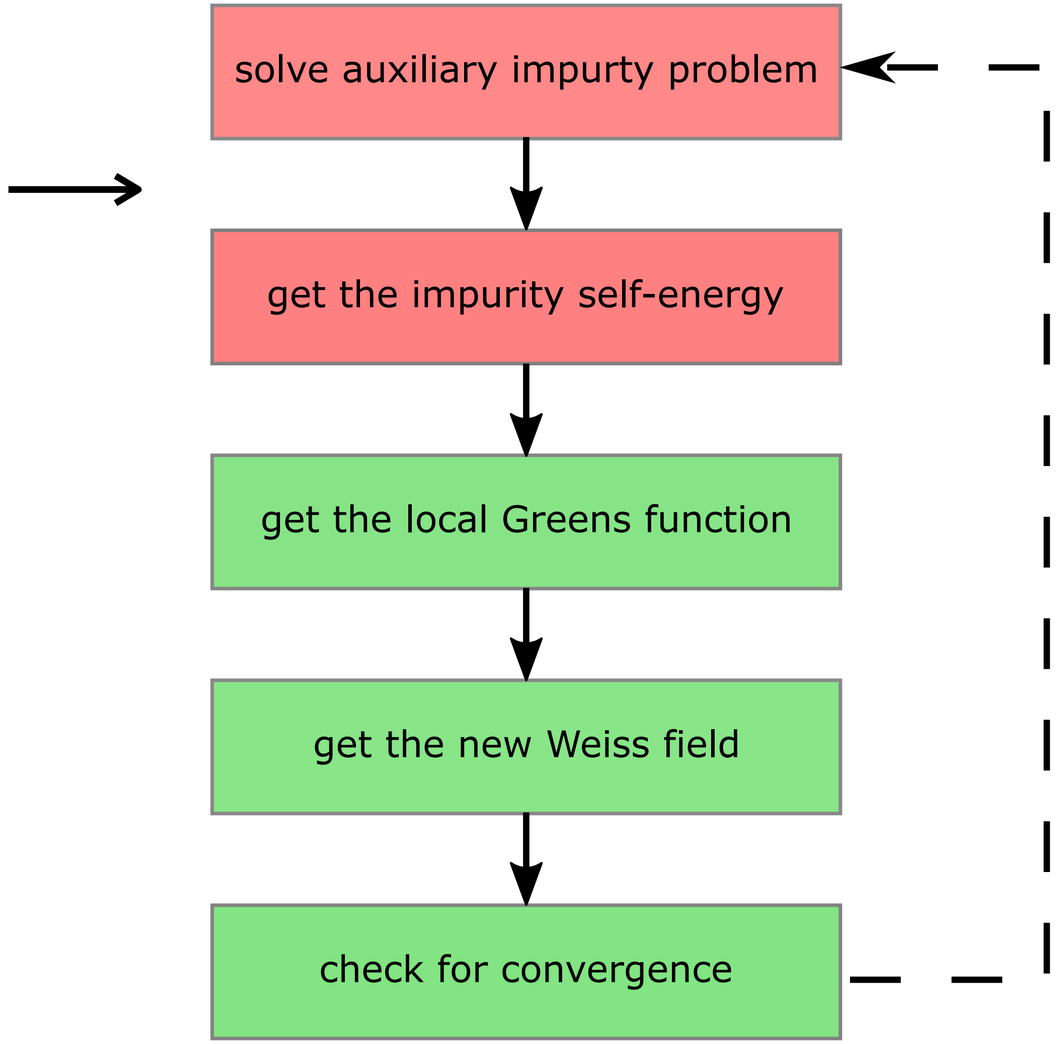


ipt_hm_matsubara.f90

```
wm(:)        !Matsubara frequencies
Sigma(:)     !impurity Self-energy
fg0(:)       !Weiss field at loop iloop
fg0_prev(:)  !Weiss field at loop iloop-1
fg(:)        !local greens function
```

```
!get or read first sigma
call  get_initial_function(Sigma,"Sigma.restart")
!dmft loop:
iloop=0 ; converged=.false.
do while(.not.converged.AND.iloop<nloop)
   iloop=iloop+1
   write(*,"(A,i5)",advance="no")"DMFT-loop",iloop
   !SELF-CONSISTENCY:
   do i=1,L
      zeta  = xi*wm(i) - sigma(i)
      fg(i) = gfbethe(wm(i),zeta,D)
   enddo
   !
   fg0_prev= fg0                          !
   fg0     = one/(one/fg + sigma)
   if(iloop>1)fg0 = wmix*fg0 + (1.d0-wmix)*fg0_prev !mix to avoid loops
   !
   !IMPURITY SOLVER: fg0-->sigma
   call solve_ipt_mats(sigma)
   !
   dens = ipt_measure_dens_matsubara(fg)
   z    = ipt_measure_zeta_matsubara(sigma,fg0)
   docc = ipt_measure_docc_matsubara(sigma,fg0)
   write(*,"(3F15.9,3x)",advance="no")dens,docc,z
   !Check CONVERGENCE on the Weiss Field:
   converged=check_convergence(fg0)
enddo
```

get the new self-energy

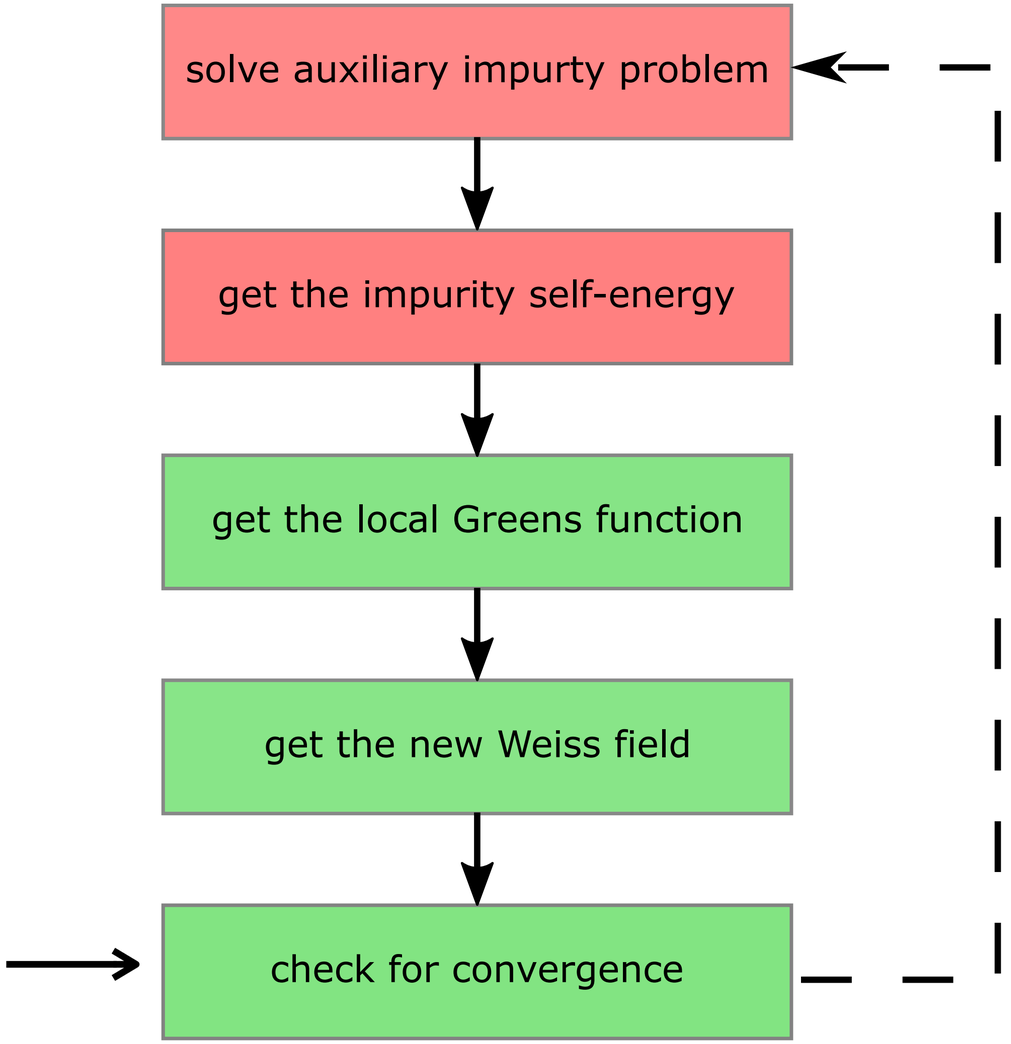


ipt_hm_matsubara.f90

```
wm(:)        !Matsubara frequencies
Sigma(:)     !impurity Self-energy
fg0(:)       !Weiss field at loop iloop
fg0_prev(:)  !Weiss field at loop iloop-1
fg(:)        !local greens function
```

```
!get or read first sigma
call  get_initial_function(Sigma,"Sigma.restart")
!dmft loop:
iloop=0 ; converged=.false.
do while(.not.converged.AND.iloop<nloop)
   iloop=iloop+1
   write(*,"(A,i5)",advance="no")"DMFT-loop",iloop
   !SELF-CONSISTENCY:
   do i=1,L
      zeta  = xi*wm(i) - sigma(i)
      fg(i) = gfbethe(wm(i),zeta,D)
   enddo
   !
   fg0_prev= fg0                          !
   fg0     = one/(one/fg + sigma)
   if(iloop>1)fg0 = wmix*fg0 + (1.d0-wmix)*fg0_prev !mix to avoid loops
   !
   !IMPURITY SOLVER: fg0-->sigma
   call solve_ipt_mats(sigma)
   !
   dens = ipt_measure_dens_matsubara(fg)
   z    = ipt_measure_zeta_matsubara(sigma,fg0)
   docc = ipt_measure_docc_matsubara(sigma,fg0)
   write(*,"(3F15.9,3x)",advance="no")dens,docc,z
   !Check CONVERGENCE on the Weiss Field:
   converged=check_convergence(fg0)
enddo
```

measure observable
heck convergence

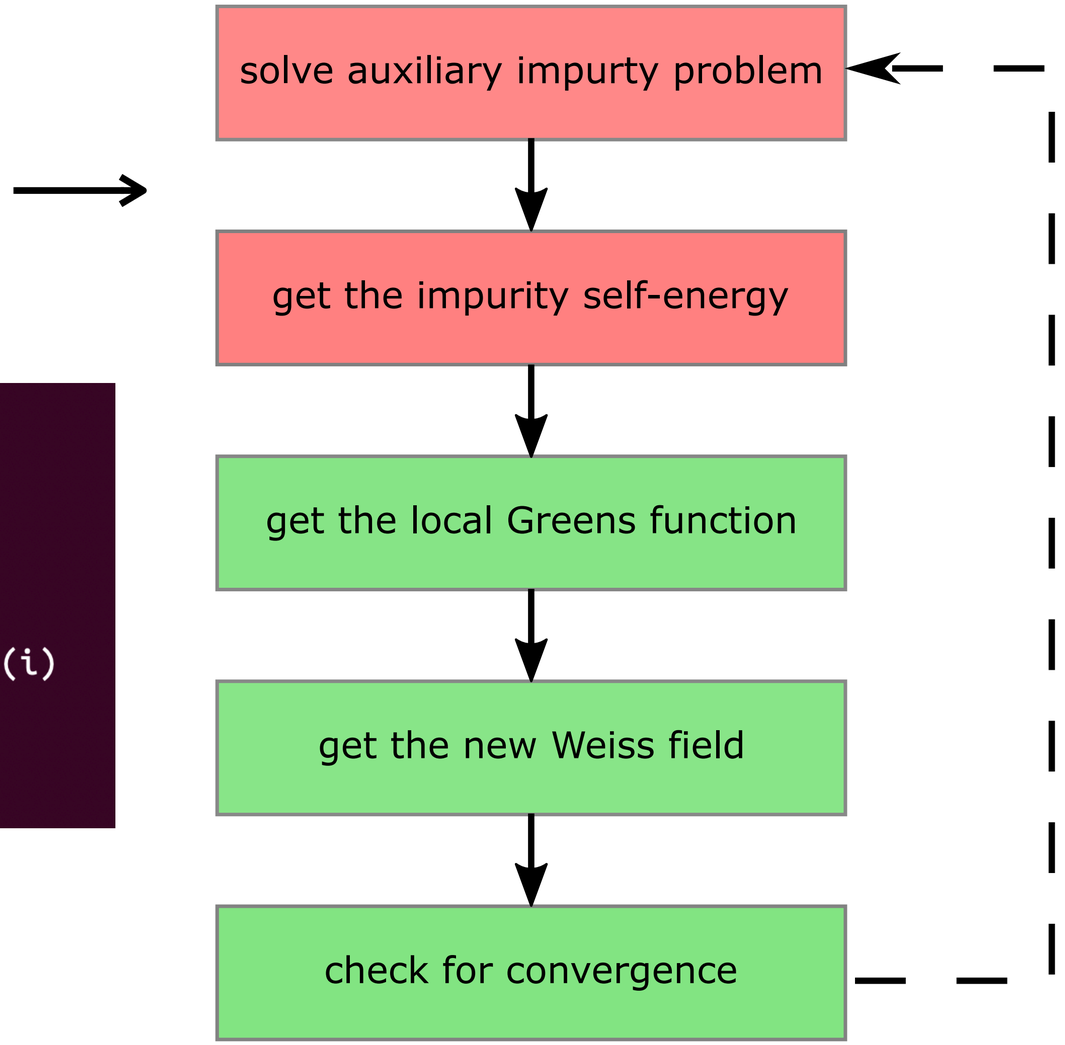


ipt_hm_matsubara.f90

The solver part

```
   !
   !IMPURITY SOLVER: fg0-->sigma
   call solve_ipt_mats(sigma)
   !
```

```
!PURPOSE  : Solve 2^nd order perturbation theory
subroutine solve_ipt_mats(sigma)
  complex(8),dimension(L) :: sigma
  real(8),dimension(0:L)  :: fg0_tau
  real(8),dimension(0:L)  :: sigma_tau
  call fftgf_iw2tau(fg0,fg0_tau(0:),beta)
  forall(i=0:L)sigma_tau(i)=Uloc*Uloc*fg0_tau(i)*fg0_tau(L-i)*fg0_tau(i)
  call fftgf_tau2iw(sigma_tau(0:),Sigma,beta)
  Sigma=xi*dimag(Sigma)
end subroutine solve_ipt_mats
```



ipt_hm_matsubara.f90

inputIPT.conf

```
&IPT_VARIABLE                                    !
NLOOP=300,   max number of DMFTloops              !
ULOC=2.0000000000000000     ,                     !
BETA=100.00000000000000     ,                     !
L=8192 ,
DMFT_ERROR=1.0000000000000000E-005,              !
WMIX=0.5d0,
/                                                !
```

ipt_hm_matsubara.f90

inputIPT.conf

```
&IPT_VARIABLE                                    !
NLOOP=300,                                        !
ULOC=2.0000000000000000      ,   Hubbard U        !
BETA=100.00000000000000      ,                    !
L=8192 ,
DMFT_ERROR=1.0000000000000000E-005,              !
WMIX=0.5d0,
/                                                !
```

ipt_hm_matsubara.f90

inputIPT.conf

```
&IPT_VARIABLE                                    !
NLOOP=300,                                        !
ULOC=2.0000000000000000      ,                    !
BETA=100.00000000000000      , Inverse temperature !
L=8192 ,
DMFT_ERROR=1.0000000000000000E-005,              !
WMIX=0.5d0,
/                                                !
```

ipt_hm_matsubara.f90

inputIPT.conf

```
&IPT_VARIABLE                                    !
NLOOP=300,                                        !
ULOC=2.0000000000000000     ,                     !
BETA=100.00000000000000     ,                     !
L=8192 ,                        number matsubara freq
DMFT_ERROR=1.0000000000000000E-005,               !
WMIX=0.5d0,
/                                                 !
```

ipt_hm_matsubara.f90

inputIPT.conf

```
&IPT_VARIABLE                                    !
NLOOP=300,                                        !
ULOC=2.0000000000000000     ,                     !
BETA=100.00000000000000     ,                     !
L=8192 ,
DMFT_ERROR=1.000000000000000E-005,               !  Threshold for convergence
WMIX=0.5d0,
/                                                !
```

ipt_hm_matsubara.f90

inputIPT.conf

```
&IPT_VARIABLE                                    !
NLOOP=300,                                        !
ULOC=2.0000000000000000     ,                     !
BETA=100.00000000000000     ,                     !
L=8192 ,
DMFT_ERROR=1.0000000000000000E-005,              !
WMIX=0.5d0,                    Mixing parameter
/                                                 !
```

ipt_hm_matsubara.f90

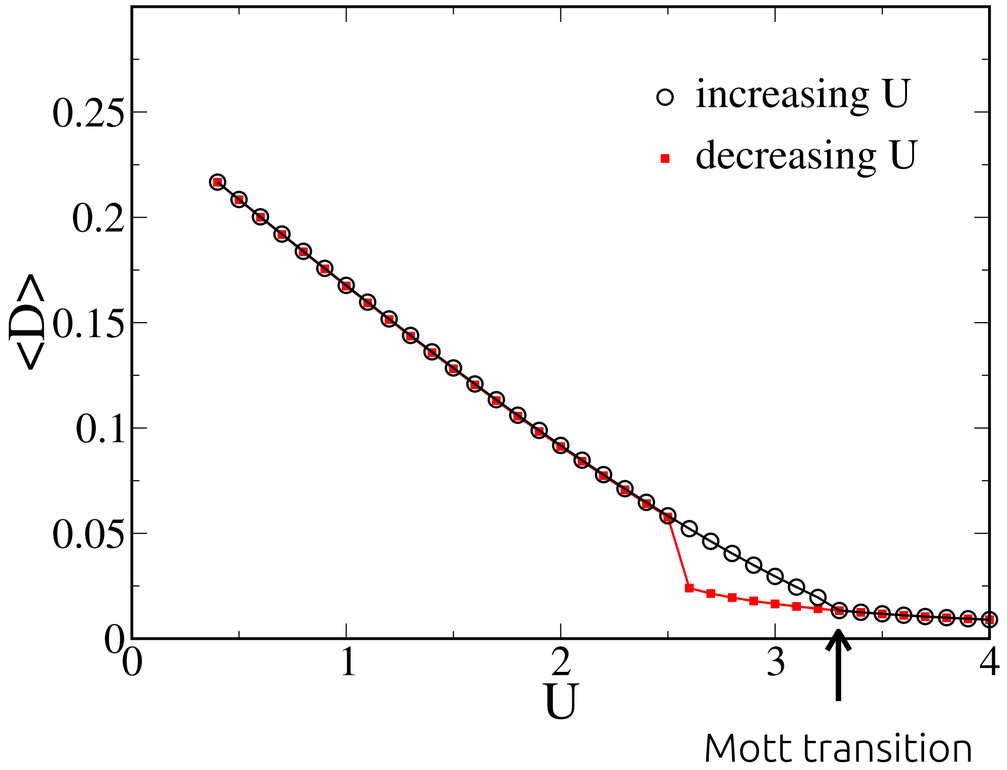
Output

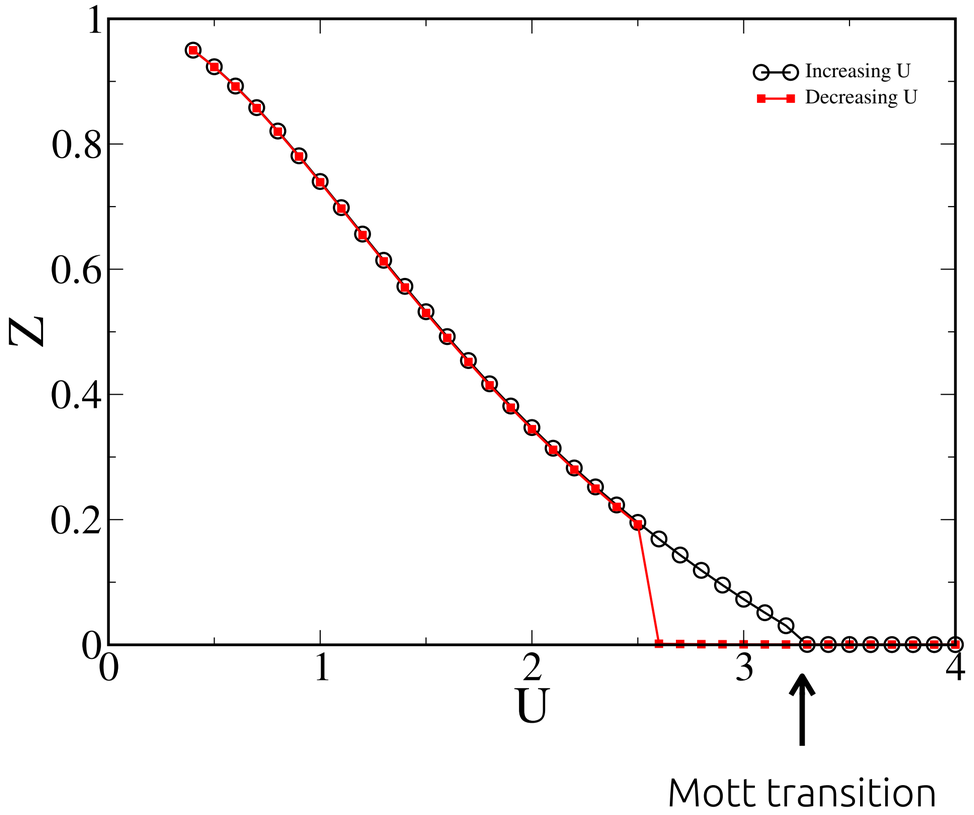
```
&IPT_VARIABLE
 NLOOP=300         ,
 ULOC=  2.0000000000000000     ,
 BETA=  100.00000000000000     ,
 L=8192         ,
 DMFT_ERROR=  1.0000000000000001E-005,
 WMIX= 0.50000000000000000     ,
 /                    ⟨n_iσ⟩          ⟨n_i↑n_i↓⟩      Z = (1 − ∂Σ'/∂ω |ω=0)^−1
DMFT-loop    1    0.500000000    0.111249673    0.433552979   error=  7.3308646E-02
DMFT-loop    2    0.500000000    0.103827550    0.400879447   error=  9.3510057E-03
DMFT-loop    3    0.500000000    0.099227494    0.380916970   error=  5.7538610E-03
DMFT-loop    4    0.500000000    0.096365233    0.368402243   error=  3.6022560E-03
DMFT-loop    5    0.500000000    0.094572993    0.360421234   error=  2.2814657E-03
DMFT-loop    6    0.500000000    0.093443732    0.355276608   error=  1.4563598E-03
DMFT-loop    7    0.500000000    0.092728302    0.351938947   error=  9.3469056E-04
DMFT-loop    8    0.500000000    0.092272972    0.349765678   error=  6.0212279E-04
DMFT-loop    9    0.500000000    0.091982091    0.348347913   error=  3.8888586E-04
DMFT-loop   10    0.500000000    0.091795699    0.347422266   error=  2.5161438E-04
DMFT-loop   11    0.500000000    0.091675965    0.346817817   error=  1.6299983E-04
DMFT-loop   12    0.500000000    0.091598896    0.346423189   error=  1.0568477E-04
DMFT-loop   13    0.500000000    0.091549208    0.346165647   error=  6.8564084E-05
DMFT-loop   14    0.500000000    0.091517130    0.345997648   error=  4.4500045E-05
DMFT-loop   15    0.500000000    0.091496399    0.345888114   error=  2.8890053E-05
DMFT-loop   16    0.500000000    0.091482989    0.345816731   error=  1.8759520E-05
DMFT-loop   17    0.500000000    0.091474309    0.345770233   error=  1.2182984E-05
DMFT-loop   18    0.500000000    0.091468686    0.345739956   error=  7.9127131E-06
DMFT-loop   19    0.500000000    0.091465042    0.345720248   error=  5.1395346E-06
```

Output files

```
[giacomo@hyperion 12:
G0_iw.ipt
G_iw.ipt
Sigma.restart
Sigma_iw.ipt
inputIPT.conf
observables_last.ipt
```

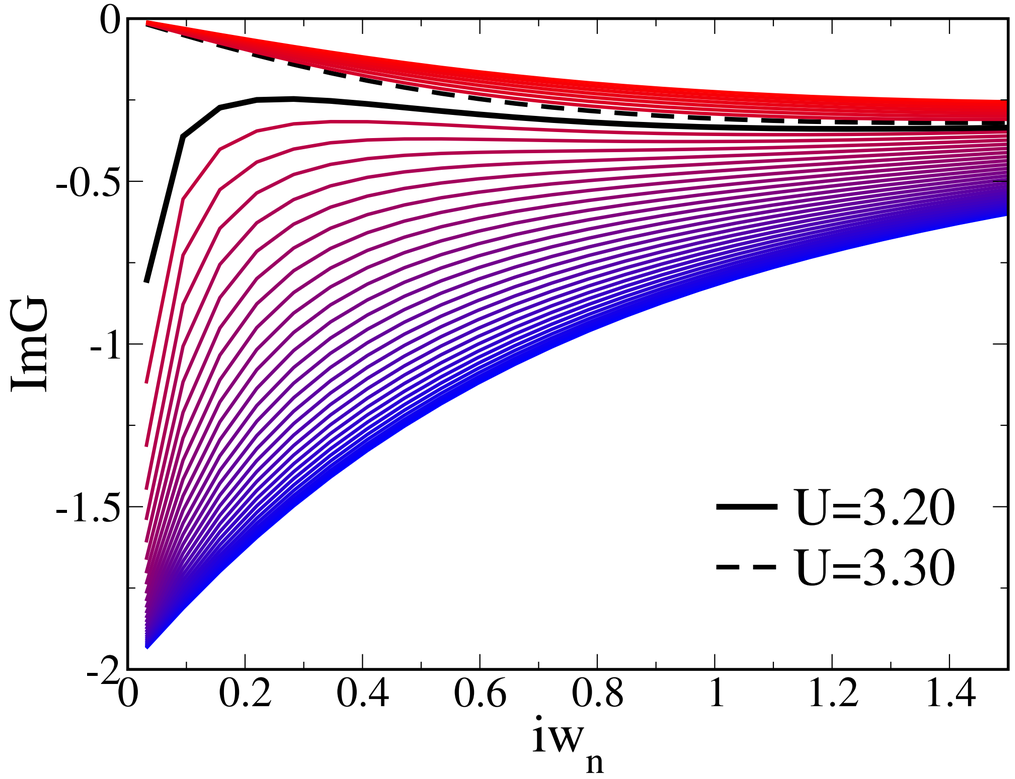
Example: run the code at different U adiabatically following the solution

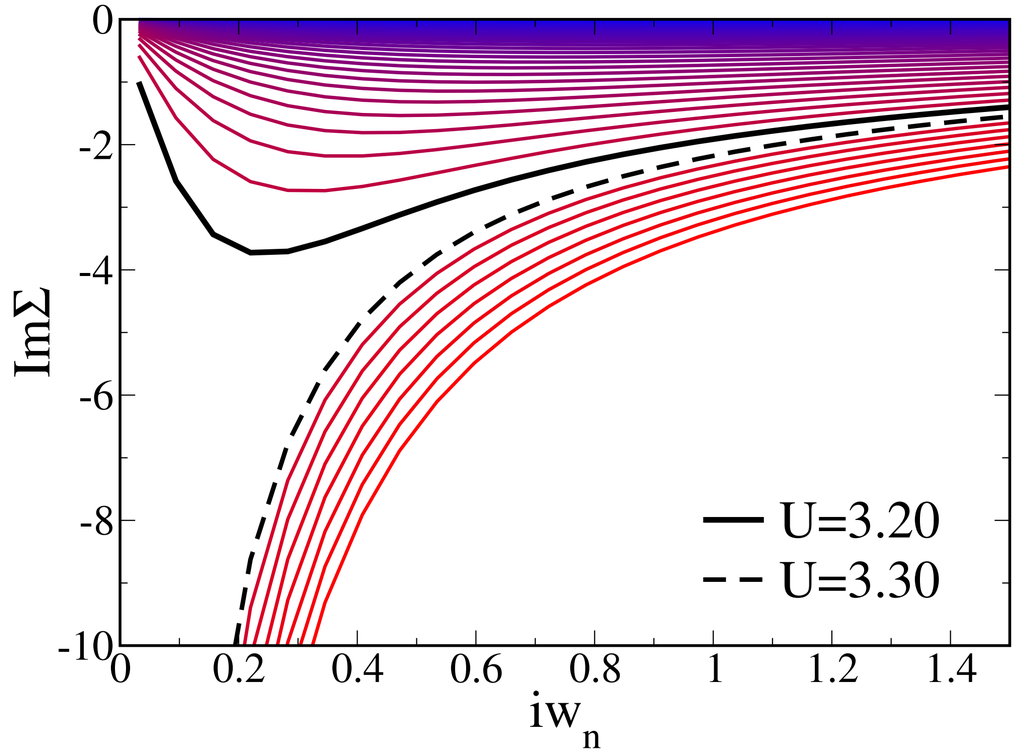




For fun: change the temperature and check that the Mott transition becomes first order

Green’s function and self-energies VS imaginary frequencies





For fun: change the temperature and check that the Mott transition becomes first order

ipt_hm_real.f90

```
!dmft loop:
iloop=0 ; converged=.false.
do while(.not.converged.AND.iloop<nloop)
   iloop=iloop+1
   write(*,"(A,i5,A)",advance="no")"DMFT-loop",iloop," "
   !
   !
   !SELF-CONSISTENCY:
   do i=1,L
      zeta = dcmplx(wr(i),eps) - sigma(i)
      fg(i) = gfbether(wr(i),zeta,D)
   enddo
   !
   fg0_prev= fg0                                    !
   fg0     = one/(one/fg + sigma)                   !
   if(iloop>1)fg0 = wmix*fg0 + (1.d0-wmix)*fg0_prev !mix to avoid loops
   !
   !IMPURITY SOLVER: fg0-->sigma
   call solve_ipt_real(sigma)
   !
   !Check CONVERGENCE on the Weiss Field:
   converged=check_convergence(fg0)
enddo
```

$i\omega_n \rightarrow \omega + i0^+$

real freq solver

```
&IPT_VARIABLE                                      !
NLOOP=300,                                          !
ULOC=2.0000000      ,                   !
BETA=100.00000000000000     ,                      !
L=2000,
DMFT_ERROR=1.0000000000000001E-004,                 !
WMIX=0.5d0,
EPS=0.01d0,
/                                                   !
```

real frequency discretisation

Imaginary broadening

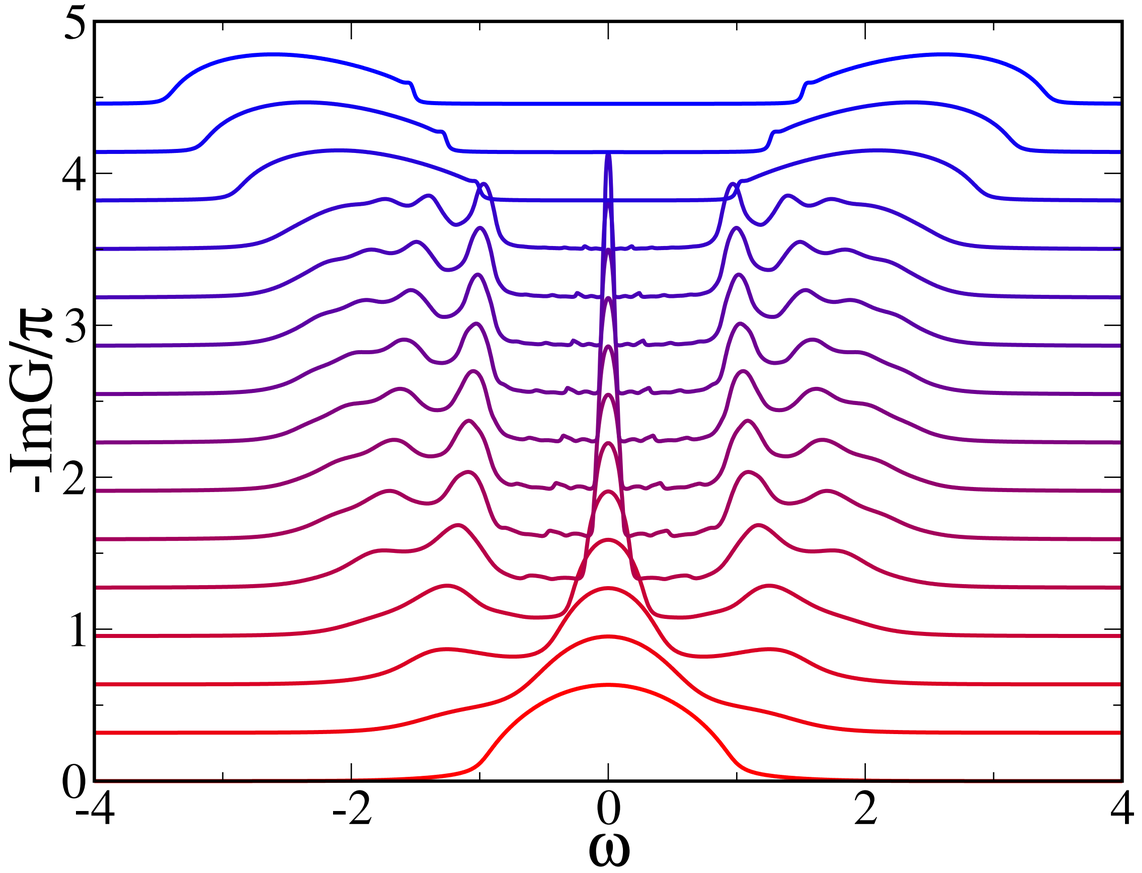
```
!PURPOSE  : Solve 2^nd order perturbation theory
subroutine solve_ipt_real(sigma)
  complex(8),dimension(L) :: sigma
  integer                 :: ix,iy,iz
  real(8)                 :: sum1,sum2
  real(8),dimension(L)    :: reS,imS
  !
  call getAs
  call getPolarization
  !
  do ix=1,L
     sum1=zero
     sum2=zero
     do iy=1,L
        iz= iy_m_ix(iy,ix)
        if(iz>0)then
           sum1=sum1+A0p(iy)*P1(iz)*mesh
           sum2=sum2+A0m(iy)*P2(iz)*mesh
        end if
     enddo
     imS(ix)=-Uloc*Uloc*(sum1+sum2)*pi
  enddo
  reS   = kronig(imS,wr,L)
  sigma = reS + xi*imS
end subroutine solve_ipt_real
```

real freq solver

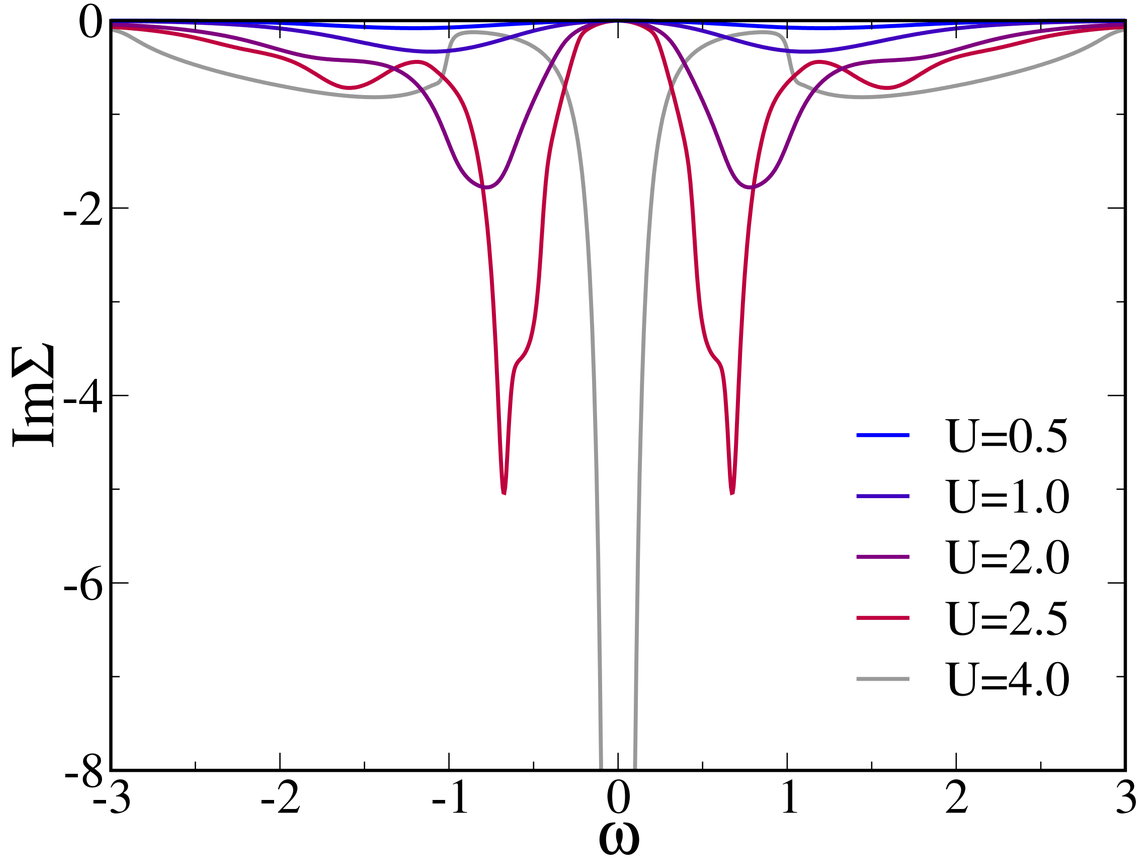
Frequency convolutions
In the sunset diagram

The real frequency solver gives access at the spectral functions

Formation of the Mott gap



Fermi liquid VS non-fermi liquid self-energies

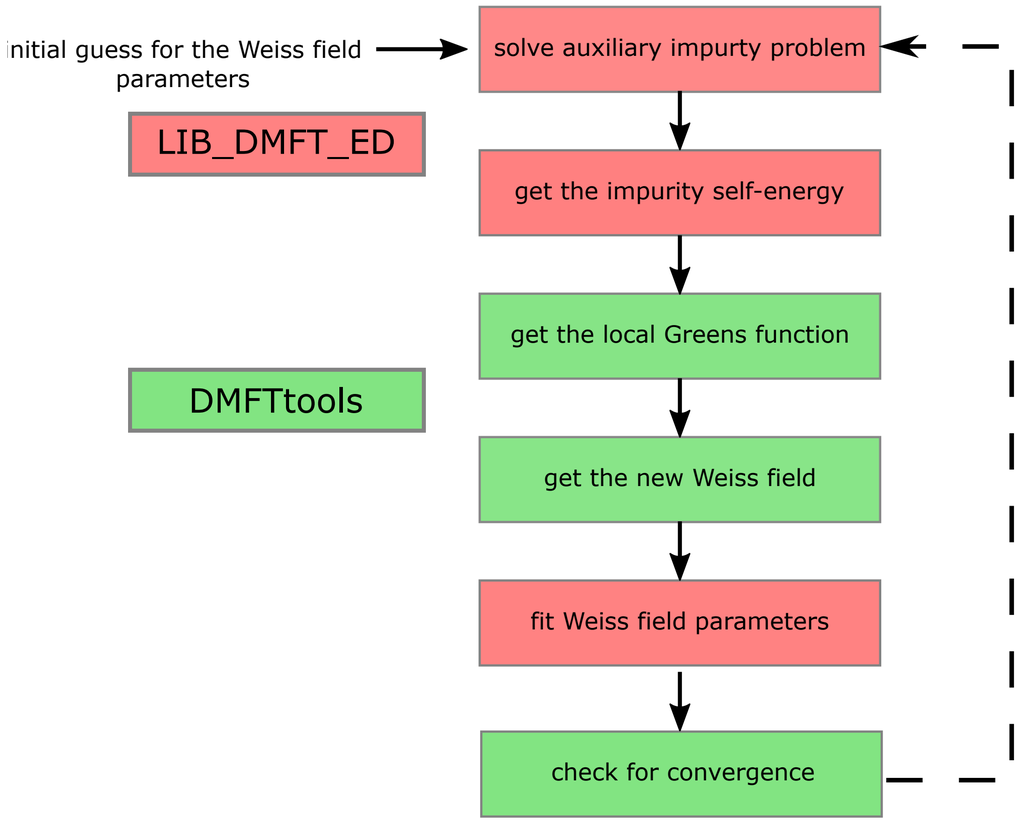


Exact Diagonalization solver

$$H_{imp} = U n_{i\uparrow} n_{i\downarrow} - \mu \sum_{\sigma} n_{i\sigma} + \sum_{l\sigma} v_l c^{\dagger}_{i\sigma} d_{l\sigma} + h.c. + \sum_{l} \varepsilon_l d^{\dagger}_{l\sigma} d_{l\sigma}$$

$$\Delta(i\omega_n) = \sum_{l} \frac{|v_l|^2}{i\omega_n - \varepsilon_l}$$

$$\mathcal{G}_0^{-1}(i\omega_n) = i\omega_n + \mu - \Delta(i\omega_n)$$

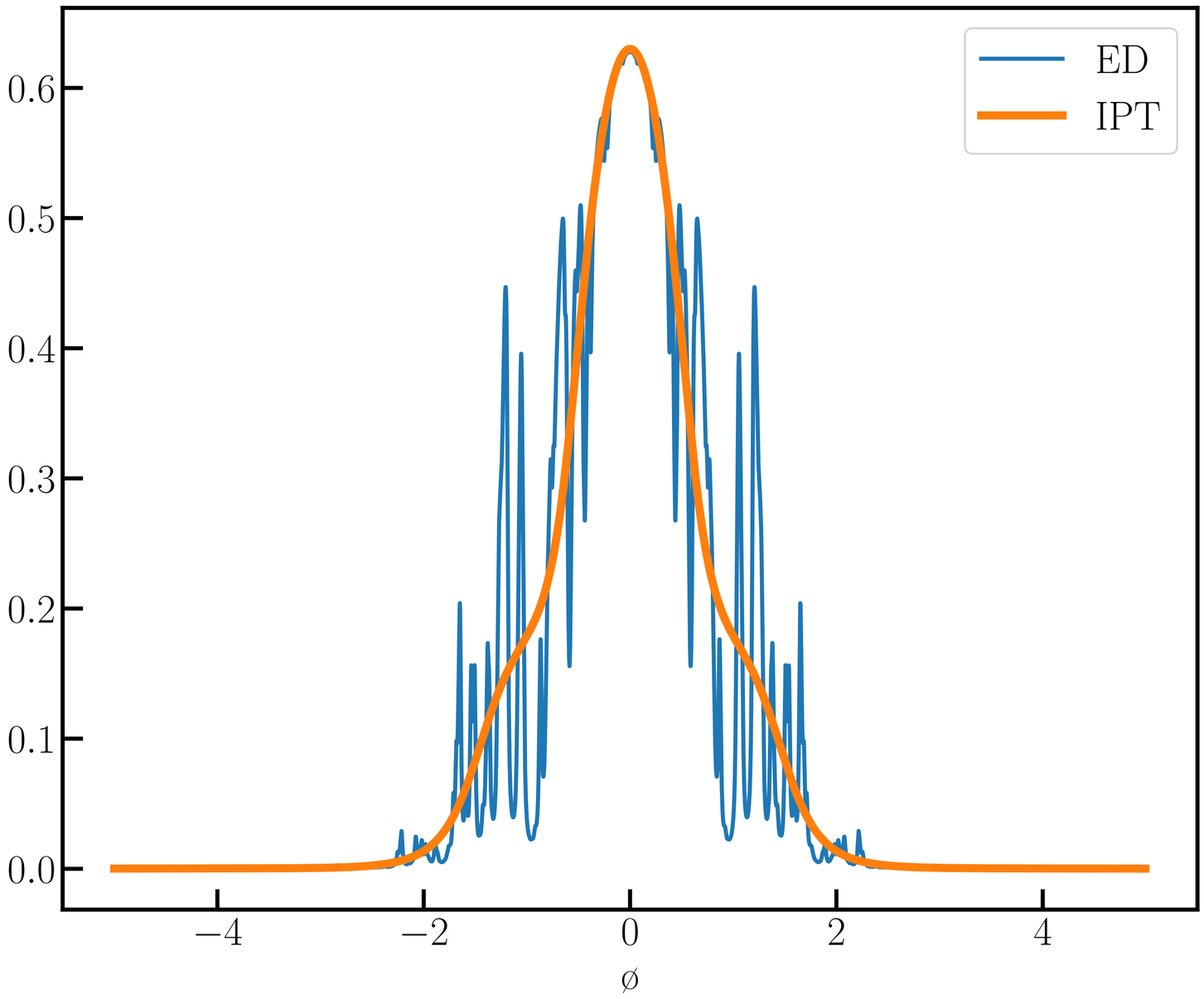


1. Find the ground state of H_imp → Lanczos method
2. Compute the Green's function

Minimize a cost function to get the new bath parameters

$$\chi^2[\{\varepsilon_l, v_l\}] = \sum_{\omega_n} w_n |\Delta(i\omega_n; \{\varepsilon_l, v_l\}) - \Delta_{\text{target}}(i\omega_n)|^2$$

!!!! ED is not good for real axis calculations !!!!


ED
IPT
0.6
0.5
0.4
0.3
0.2
0.1
0.0
−4
−2
0
2
4
ø

DMFT equations for superconductivity

Replace all the Green's function by matrix Green's functions in the Nambu representation

$$\mathbf{G}_{ii} = \begin{pmatrix} -\langle T_\tau c_{i\uparrow}(\tau) c^\dagger_{i\uparrow}(\tau')\rangle & -\langle T_\tau c^\dagger_{i\uparrow}(\tau) c^\dagger_{i\downarrow}(\tau')\rangle \\ -\langle T_\tau c_{i\downarrow}(\tau) c_{i\uparrow}(\tau')\rangle & -\langle T_\tau c^\dagger_{i\downarrow}(\tau) c_{i\downarrow}(\tau')\rangle \end{pmatrix} = \begin{pmatrix} G_{\uparrow\uparrow}(\tau-\tau') & F_{\uparrow\downarrow}(\tau-\tau') \\ \overline{F}_{\downarrow\uparrow}(\tau-\tau') & \overline{G}_{\downarrow\downarrow}(\tau-\tau') \end{pmatrix}$$

$G_{\uparrow\uparrow}(\tau-\tau')$ $\quad$ $\overline{G}_{\downarrow\downarrow}(\tau-\tau')$ $\quad$ Normal Green's function

$F_{\uparrow\downarrow}(\tau-\tau')$ $\quad$ $\overline{F}_{\downarrow\uparrow}(\tau-\tau')$ $\quad$ Anomalous Green's function

Same for Weiss field and Self-energy

$$H^{SC}_{imp} = U n_{i\uparrow} n_{i\downarrow} - \mu \sum_\sigma n_{i\sigma} + \sum_{l\sigma} v_l c^\dagger_{i\sigma} d_{l\sigma} + h.c. + \sum_l \varepsilon_l d^\dagger_{l\sigma} d_{l\sigma} + \sum_l \Delta_l d^\dagger_{l\uparrow} d^\dagger_{l\downarrow} + h.c$$

extra parameter in the impurity Hamiltonian
Allow for SC symmetry breaking in the Weiss field

ed_ahm_bethe.f90 driver (*) to solve the (attractive) Hubbard model on the bethe lattice

$$H = -t \sum_{ij} c^{\dagger}_{i\sigma} c_{i\sigma} - |U| \sum_{i} n_{i\uparrow} n_{i\downarrow} - \mu \sum_{i\sigma} n_{i\sigma}$$

Prototype model for unconventional (ie non BCS) superconductivity (s-wave)

(*) driver = small program that makes use of the routines in the Libraries DMFTtools and LIB_DMFT_ET

Different physical problems may need slightly different sets of instructions

If you want to study a different problem then you may need to write a specific driver

The driver ed_ahm_bethe.f90 is quite general and it represents a good starting point for any physical problem

****If you want to use this code in your research please cite ***

Amaricci et al, "EDIpack: A parallel exact diagonalization package for quantum impurity problems" arXiv 2105.06806

ed_ahm_bethe.f90 solve the attractive Hubbard model on the be the Lattice

```
!
!Allocate local functions (Nambu)
allocate(Gmats(2,Nspin,Nspin,Norb,Norb,Lmats))
allocate(Smats(2,Nspin,Nspin,Norb,Norb,Lmats))
allocate(Weiss(2,Nspin,Nspin,Norb,Norb,Lmats))
!
allocate(Greal(2,Nspin,Nspin,Norb,Norb,Lreal))
allocate(Sreal(2,Nspin,Nspin,Norb,Norb,Lreal))
!
allocate(Ebethe(1,Le))
Ebethe(1,:) = linspace(-Wband,Wband,Le,mesh=de)
!
allocate(Dbethe(1,Le))
Dbethe(1,:) = dens_bethe(Ebethe(1,:),wband)*de
!
allocate(Hloc(1,1,1,1))
Hloc        = 0d0
!
allocate(H0(1))
H0=0d0
!
!Setup solver
Nb=ed_get_bath_dimension()
allocate(Bath(Nb))
allocate(Bath_prev(Nb))
call ed_init_solver(bath)
```

Allocate normal AND anomalous
Greens Functions/Weiss field/Self energy

ed_ahm_bethe.f90 solve the attractive Hubbard model on the be the Lattice

```
!
!Allocate local functions (Nambu)
allocate(Gmats(2,Nspin,Nspin,Norb,Norb,Lmats))
allocate(Smats(2,Nspin,Nspin,Norb,Norb,Lmats))
allocate(Weiss(2,Nspin,Nspin,Norb,Norb,Lmats))
!
allocate(Greal(2,Nspin,Nspin,Norb,Norb,Lreal))
allocate(Sreal(2,Nspin,Nspin,Norb,Norb,Lreal))
!
allocate(Ebethe(1,Le))
Ebethe(1,:) = linspace(-Wband,Wband,Le,mesh=de)
!
allocate(Dbethe(1,Le))
Dbethe(1,:) = dens_bethe(Ebethe(1,:),wband)*de
!
allocate(Hloc(1,1,1,1))
Hloc        = 0d0
!
allocate(H0(1))
H0=0d0
!
!Setup solver
Nb=ed_get_bath_dimension()
allocate(Bath(Nb))
allocate(Bath_prev(Nb))
call ed_init_solver(bath)
```

Set the density of states of the Bethe lattice

Set the energy of the atomic levels (here 0)

ed_ahm_bethe.f90 solve the attractive Hubbard model on the be the Lattice

```
!
!Allocate local functions (Nambu)
allocate(Gmats(2,Nspin,Nspin,Norb,Norb,Lmats))
allocate(Smats(2,Nspin,Nspin,Norb,Norb,Lmats))
allocate(Weiss(2,Nspin,Nspin,Norb,Norb,Lmats))
!
allocate(Greal(2,Nspin,Nspin,Norb,Norb,Lreal))
allocate(Sreal(2,Nspin,Nspin,Norb,Norb,Lreal))
!
allocate(Ebethe(1,Le))
Ebethe(1,:) = linspace(-Wband,Wband,Le,mesh=de)
!
allocate(Dbethe(1,Le))
Dbethe(1,:) = dens_bethe(Ebethe(1,:),wband)*de
!
allocate(Hloc(1,1,1,1))
Hloc        = 0d0
!
allocate(H0(1))
H0=0d0
!
!Setup solver
Nb=ed_get_bath_dimension()
allocate(Bath(Nb))
allocate(Bath_prev(Nb))
call ed_init_solver(bath)
```

"baths" = $\{\varepsilon_l, v_l, \Delta_l\}$

Allocate the bath at loop iloop
Allocate the bath at loop iloop+1

Set-up and init the solver

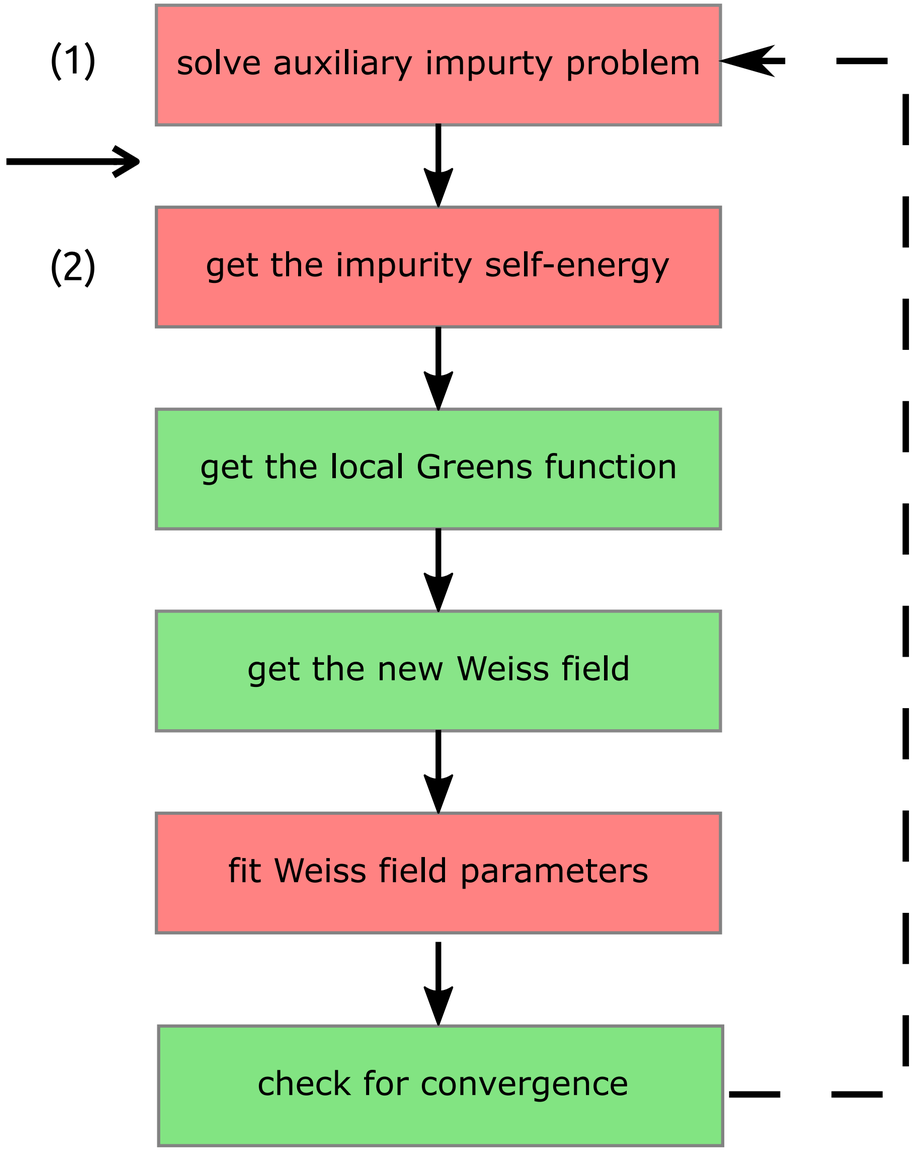
ed_ahm_bethe.f90 solve the attractive Hubbard model on the be the Lattice

```
!DMFT loop
iloop=0;converged=.false.
do while(.not.converged.AND.iloop<nloop)
   iloop=iloop+1
   call start_loop(iloop,nloop,"DMFT-loop")
   !Solve the EFFECTIVE IMPURITY PROBLEM (first w/ a guess for the bath)
   call ed_solve(bath,Hloc)
   !Retrieve impurity self-energies (normal, anomalous)
   call ed_get_Sigma_matsubara(Smats(1,:,:,:,:,:))
   call ed_get_Self_matsubara(Smats(2,:,:,:,:,:))
   call ed_get_Sigma_realaxis(Sreal(1,:,:,:,:,:))
   call ed_get_Self_realaxis(Sreal(2,:,:,:,:,:))
   call ed_get_dens(dens,iorb=1)

   !Compute the local gfs:
   call dmft_gloc_matsubara(Ebethe,Dbethe,H0,Gmats,Smats)

   call dmft_self_consistency(Gmats(1,:,:,:,:,:),Gmats(2,:,:,:,:,:),&
        Smats(1,:,:,:,:,:),Smats(2,:,:,:,:,:),&
        Weiss(1,:,:,:,:,:),Weiss(2,:,:,:,:,:),&
        Hloc,trim(cg_scheme))
   call dmft_print_gf_matsubara(Gmats(1,:,:,:,:,:),"Gloc",iprint=1)
   call dmft_print_gf_matsubara(Gmats(2,:,:,:,:,:),"Floc",iprint=1)
   call dmft_print_gf_matsubara(Weiss(1,:,:,:,:,:),"Weiss",iprint=1)
   call dmft_print_gf_matsubara(Weiss(2,:,:,:,:,:),"fWeiss",iprint=1)
   !Perform the self-consistency fitting the new bath
   call ed_chi2_fitgf(Weiss,bath,ispin=1)
   !if it holds apply symmetrizations
   if(phsym)call ed_ph_symmetrize_bath(bath,save=.true.)
   if(normal_bath)call ed_enforce_normal_bath(bath,save=.true.)
   !MIXING:
   if(iloop>1)Bath = wmixing*Bath + (1.d0-wmixing)*Bath_prev
   Bath_prev=Bath
   !Check convergence (if required change chemical potential)
   converged = check_convergence(Weiss(1,1,1,1,1,:),dmft_error,nsuccess,nloop,reset=.false.)
   if(nread/=0.d0)call ed_search_variable(xmu,dens,converged)
   !Close this DMFT loop
   call end_loop
enddo
```

(1) find the ground state of H_imp

(2) compute GF

solver part

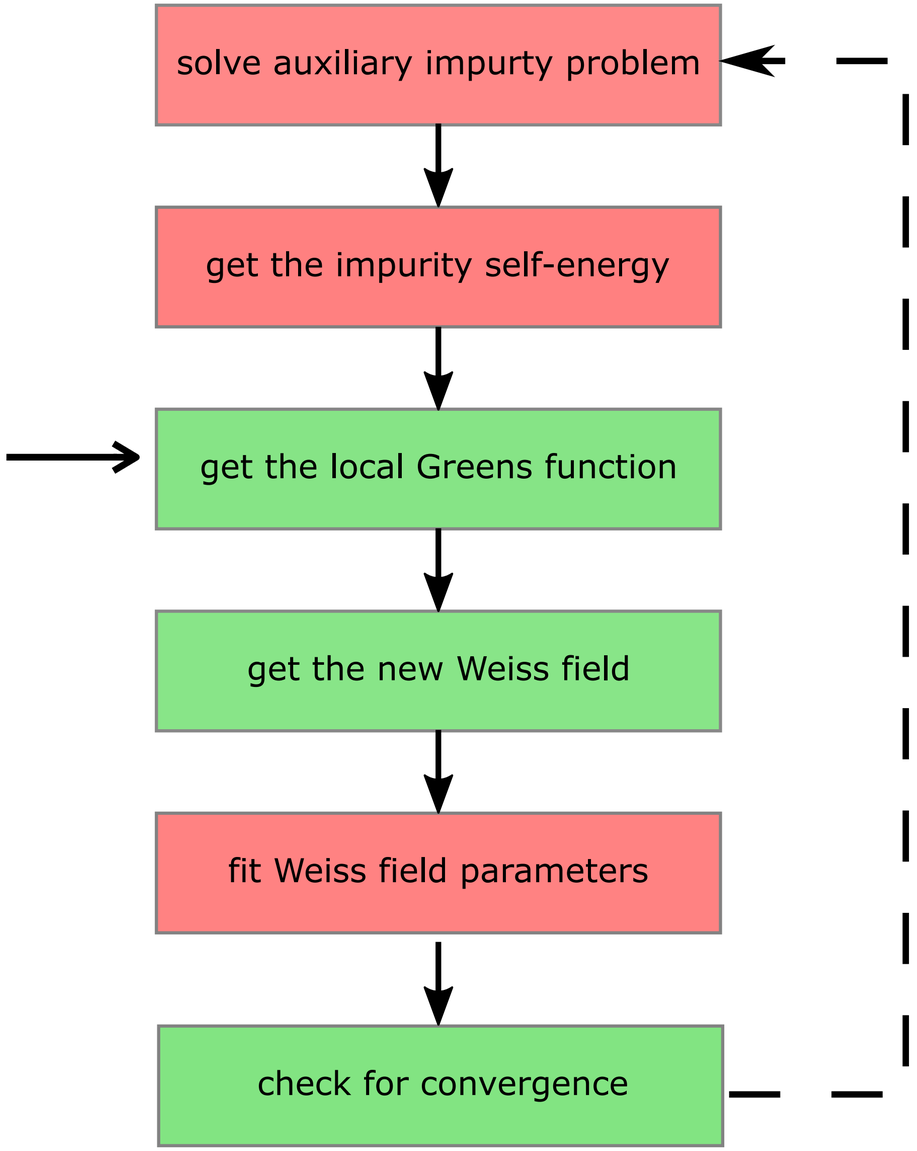


ed_ahm_bethe.f90 solve the attractive Hubbard model on the be the Lattice

```
!DMFT loop
iloop=0;converged=.false.
do while(.not.converged.AND.iloop<nloop)
   iloop=iloop+1
   call start_loop(iloop,nloop,"DMFT-loop")
   !Solve the EFFECTIVE IMPURITY PROBLEM (first w/ a guess for the bath)
   call ed_solve(bath,Hloc)

   !Retrieve impurity self-energies (normal, anomalous)
   call ed_get_Sigma_matsubara(Smats(1,:,:,:,:,:))
   call ed_get_Self_matsubara(Smats(2,:,:,:,:,:))
   call ed_get_Sigma_realaxis(Sreal(1,:,:,:,:,:))
   call ed_get_Self_realaxis(Sreal(2,:,:,:,:,:))
   call ed_get_dens(dens,iorb=1)

   !Compute the local gfs:
   call dmft_gloc_matsubara(Ebethe,Dbethe,H0,Gmats,Smats)

   call dmft_self_consistency(Gmats(1,:,:,:,:,:),Gmats(2,:,:,:,:,:),&
        Smats(1,:,:,:,:,:),Smats(2,:,:,:,:,:),&
        Weiss(1,:,:,:,:,:),Weiss(2,:,:,:,:,:),&
        Hloc,trim(cg_scheme))
   call dmft_print_gf_matsubara(Gmats(1,:,:,:,:,:),"Gloc",iprint=1)
   call dmft_print_gf_matsubara(Gmats(2,:,:,:,:,:),"Floc",iprint=1)
   call dmft_print_gf_matsubara(Weiss(1,:,:,:,:,:),"Weiss",iprint=1)
   call dmft_print_gf_matsubara(Weiss(2,:,:,:,:,:),"fWeiss",iprint=1)
   !Perform the self-consistency fitting the new bath
   call ed_chi2_fitgf(Weiss,bath,ispin=1)
   !if it holds apply symmetrizations
   if(phsym)call ed_ph_symmetrize_bath(bath,save=.true.)
   if(normal_bath)call ed_enforce_normal_bath(bath,save=.true.)
   !MIXING:
   if(iloop>1)Bath = wmixing*Bath + (1.d0-wmixing)*Bath_prev
   Bath_prev=Bath
   !Check convergence (if required change chemical potential)
   converged = check_convergence(Weiss(1,1,1,1,1,:),dmft_error,nsuccess,nloop,reset=.false.)
   if(nread/=0.d0)call ed_search_variable(xmu,dens,converged)
   !Close this DMFT loop
   call end_loop
enddo
```

$$G(i\omega_n) = \int d\epsilon \frac{\rho_{\text{bethe}}(\epsilon)}{i\omega_n - \epsilon - \Sigma(i\omega)}$$

Get the local GF

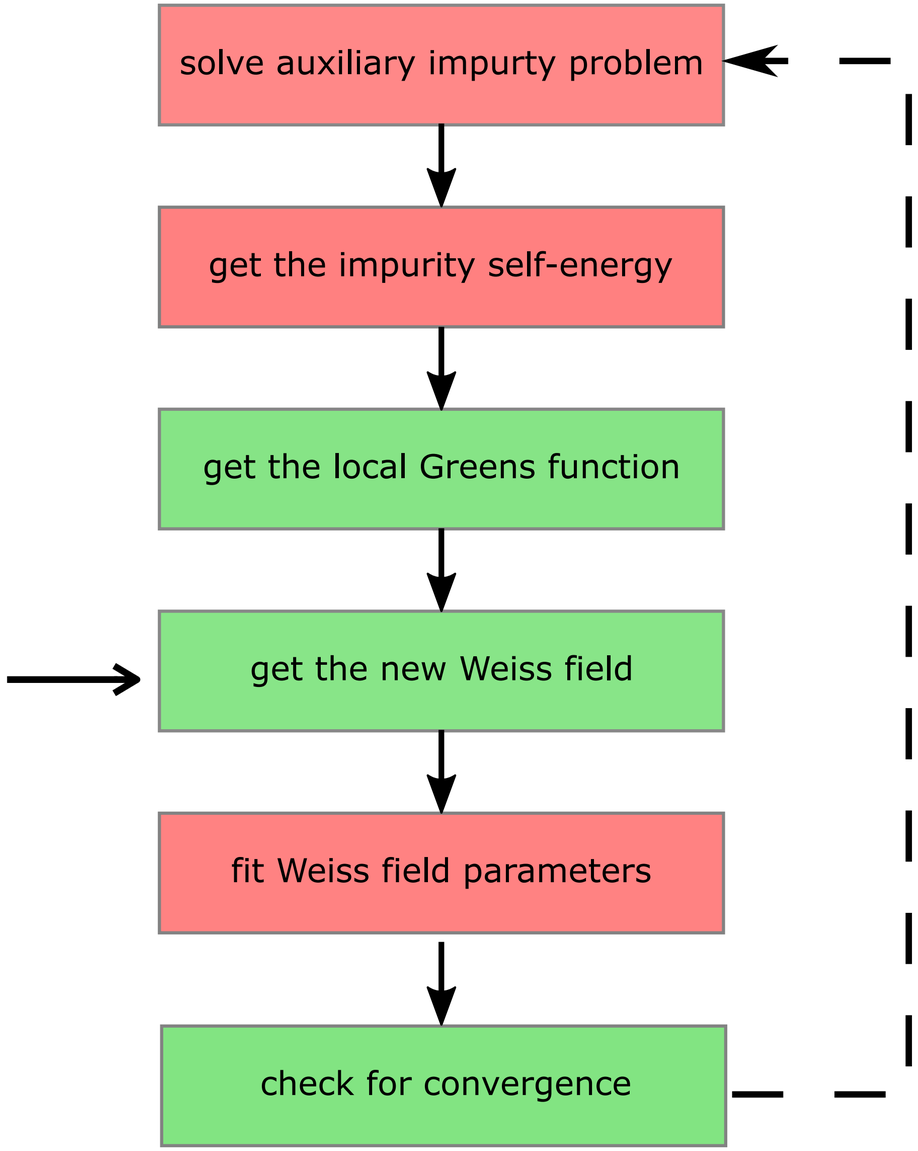


ed_ahm_bethe.f90 solve the attractive Hubbard model on the be the Lattice

```
!DMFT loop
iloop=0;converged=.false.
do while(.not.converged.AND.iloop<nloop)
   iloop=iloop+1
   call start_loop(iloop,nloop,"DMFT-loop")
   !Solve the EFFECTIVE IMPURITY PROBLEM (first w/ a guess for the bath)
   call ed_solve(bath,Hloc)

   !Retrieve impurity self-energies (normal, anomalous)
   call ed_get_Sigma_matsubara(Smats(1,:,:,:,:,:,:))
   call ed_get_Self_matsubara(Smats(2,:,:,:,:,:,:))
   call ed_get_Sigma_realaxis(Sreal(1,:,:,:,:,:,:))
   call ed_get_Self_realaxis(Sreal(2,:,:,:,:,:,:))
   call ed_get_dens(dens,iorb=1)

   !Compute the local gfs:
   call dmft_gloc_matsubara(Ebethe,Dbethe,H0,Gmats,Smats)

   call dmft_self_consistency(Gmats(1,:,:,:,:,:,:),Gmats(2,:,:,:,:,:,:),&
        Smats(1,:,:,:,:,:,:),Smats(2,:,:,:,:,:,:),&
        Weiss(1,:,:,:,:,:,:),Weiss(2,:,:,:,:,:,:),&
        Hloc,trim(cg_scheme))
   call dmft_print_gf_matsubara(Gmats(1,:,:,:,:,:,:),"Gloc",iprint=1)
   call dmft_print_gf_matsubara(Gmats(2,:,:,:,:,:,:),"Floc",iprint=1)
   call dmft_print_gf_matsubara(Weiss(1,:,:,:,:,:,:),"Weiss",iprint=1)
   call dmft_print_gf_matsubara(Weiss(2,:,:,:,:,:,:),"fWeiss",iprint=1)
   !Perform the self-consistency fitting the new bath
   call ed_chi2_fitgf(Weiss,bath,ispin=1)
   !if it holds apply symmetrizations
   if(phsym)call ed_ph_symmetrize_bath(bath,save=.true.)
   if(normal_bath)call ed_enforce_normal_bath(bath,save=.true.)
   !MIXING:
   if(iloop>1)Bath = wmixing*Bath + (1.d0-wmixing)*Bath_prev
   Bath_prev=Bath
   !Check convergence (if required change chemical potential)
   converged = check_convergence(Weiss(1,1,1,1,1,:),dmft_error,nsuccess,nloop,reset=.false.)
   if(nread/=0.d0)call ed_search_variable(xmu,dens,converged)
   !Close this DMFT loop
   call end_loop
enddo
```

$$\mathcal{G}_0^{\rm new}(i\omega_n) = \frac{1}{G^{-1}(i\omega_n) + \Sigma(i\omega_n)}$$

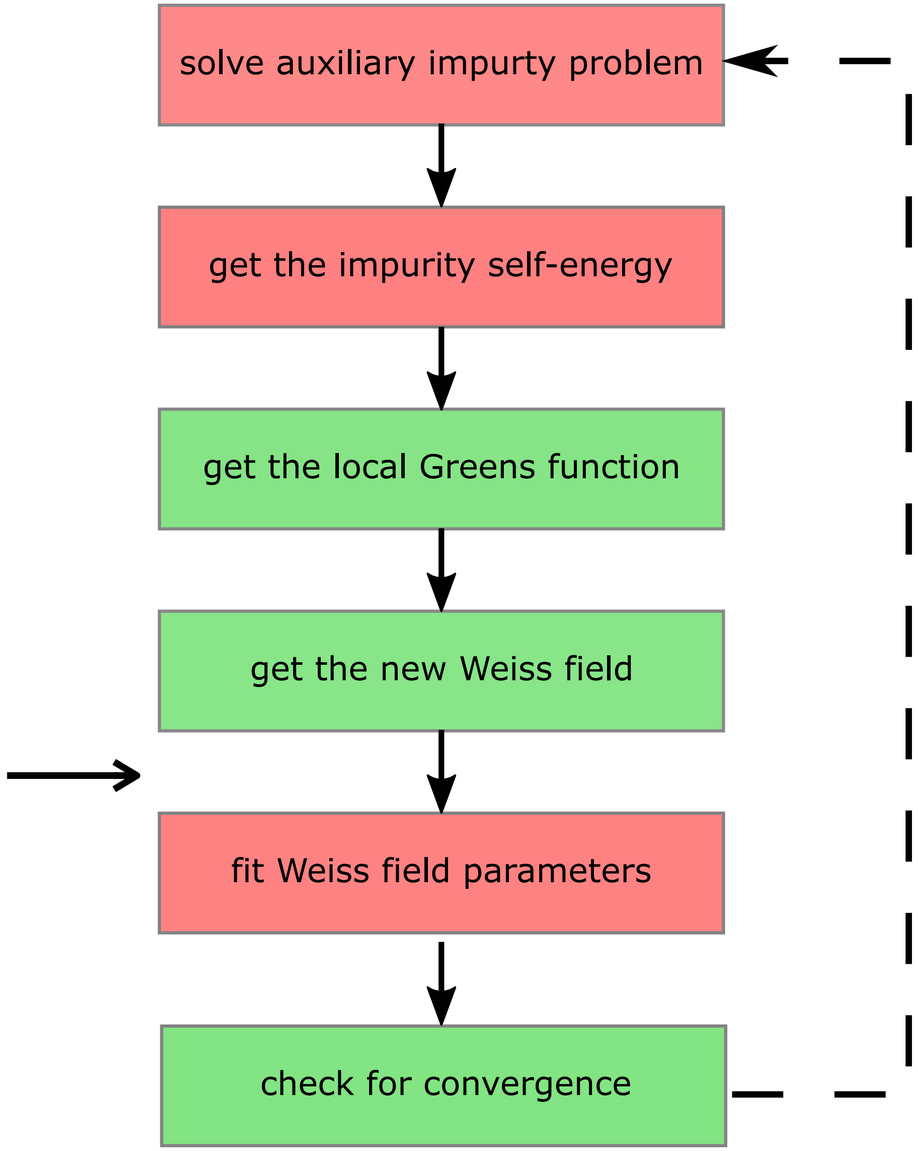
Get the new Weiss field



ed_ahm_bethe.f90 solve the attractive Hubbard model on the be the Lattice

```
!DMFT loop
iloop=0;converged=.false.
do while(.not.converged.AND.iloop<nloop)
   iloop=iloop+1
   call start_loop(iloop,nloop,"DMFT-loop")
   !Solve the EFFECTIVE IMPURITY PROBLEM (first w/ a guess for the bath)
   call ed_solve(bath,Hloc)

   !Retrieve impurity self-energies (normal, anomalous)
   call ed_get_Sigma_matsubara(Smats(1,:,:,:,:,:,:))
   call ed_get_Self_matsubara(Smats(2,:,:,:,:,:,:))
   call ed_get_Sigma_realaxis(Sreal(1,:,:,:,:,:,:))
   call ed_get_Self_realaxis(Sreal(2,:,:,:,:,:,:))
   call ed_get_dens(dens,iorb=1)

   !Compute the local gfs:
   call dmft_gloc_matsubara(Ebethe,Dbethe,H0,Gmats,Smats)

   call dmft_self_consistency(Gmats(1,:,:,:,:,:,:),Gmats(2,:,:,:,:,:,:),&
        Smats(1,:,:,:,:,:,:),Smats(2,:,:,:,:,:,:),&
        Weiss(1,:,:,:,:,:,:),Weiss(2,:,:,:,:,:,:),&
        Hloc,trim(cg_scheme))
   call dmft_print_gf_matsubara(Gmats(1,:,:,:,:,:,:),"Gloc",iprint=1)
   call dmft_print_gf_matsubara(Gmats(2,:,:,:,:,:,:),"Floc",iprint=1)
   call dmft_print_gf_matsubara(Weiss(1,:,:,:,:,:,:),"Weiss",iprint=1)
   call dmft_print_gf_matsubara(Weiss(2,:,:,:,:,:,:),"fWeiss",iprint=1)
   !Perform the self-consistency fitting the new bath
   call ed_chi2_fitgf(Weiss,bath,ispin=1)
   !if it holds apply symmetrizations
   if(phsym)call ed_ph_symmetrize_bath(bath,save=.true.)
   if(normal_bath)call ed_enforce_normal_bath(bath,save=.true.)
   !MIXING:
   if(iloop>1)Bath = wmixing*Bath + (1.d0-wmixing)*Bath_prev
   Bath_prev=Bath
   !Check convergence (if required change chemical potential)
   converged = check_convergence(Weiss(1,1,1,1,1,:),dmft_error,nsuccess,nloop,reset=.false.)
   if(nread/=0.d0)call ed_search_variable(xmu,dens,converged)
   !Close this DMFT loop
   call end_loop
enddo
```

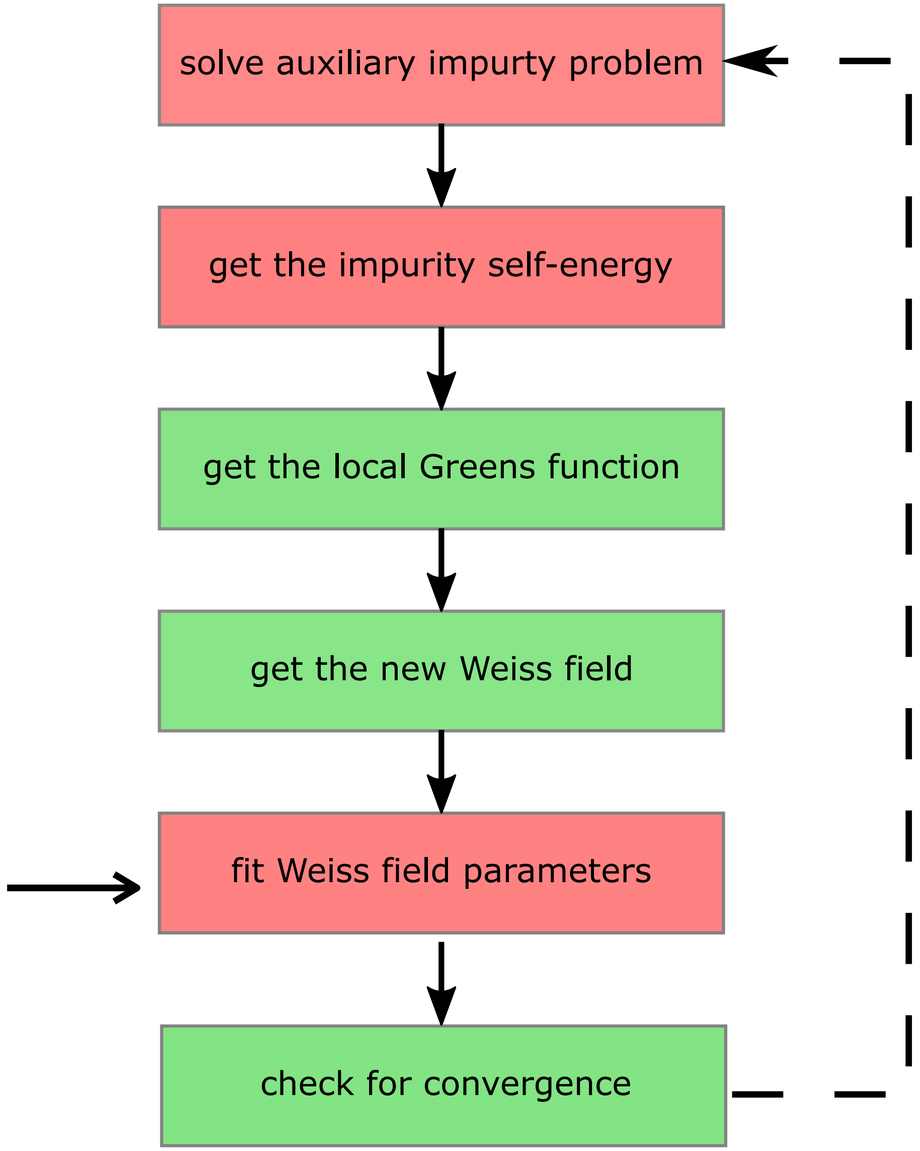
Printout GF and WF



ed_ahm_bethe.f90 solve the attractive Hubbard model on the be the Lattice

```
!DMFT loop
iloop=0;converged=.false.
do while(.not.converged.AND.iloop<nloop)
   iloop=iloop+1
   call start_loop(iloop,nloop,"DMFT-loop")
   !Solve the EFFECTIVE IMPURITY PROBLEM (first w/ a guess for the bath)
   call ed_solve(bath,Hloc)

   !Retrieve impurity self-energies (normal, anomalous)
   call ed_get_Sigma_matsubara(Smats(1,:,:,:,:,:,:))
   call ed_get_Self_matsubara(Smats(2,:,:,:,:,:,:))
   call ed_get_Sigma_realaxis(Sreal(1,:,:,:,:,:,:))
   call ed_get_Self_realaxis(Sreal(2,:,:,:,:,:,:))
   call ed_get_dens(dens,iorb=1)

   !Compute the local gfs:
   call dmft_gloc_matsubara(Ebethe,Dbethe,H0,Gmats,Smats)

   call dmft_self_consistency(Gmats(1,:,:,:,:,:,:),Gmats(2,:,:,:,:,:,:),&
        Smats(1,:,:,:,:,:,:),Smats(2,:,:,:,:,:,:),&
        Weiss(1,:,:,:,:,:,:),Weiss(2,:,:,:,:,:,:),&
        Hloc,trim(cg_scheme))
   call dmft_print_gf_matsubara(Gmats(1,:,:,:,:,:,:),"Gloc",iprint=1)
   call dmft_print_gf_matsubara(Gmats(2,:,:,:,:,:,:),"Floc",iprint=1)
   call dmft_print_gf_matsubara(Weiss(1,:,:,:,:,:,:),"Weiss",iprint=1)
   call dmft_print_gf_matsubara(Weiss(2,:,:,:,:,:,:),"fWeiss",iprint=1)
   !Perform the self-consistency fitting the new bath
   call ed_chi2_fitgf(Weiss,bath,ispin=1)
   !if it holds apply symmetrizations
   if(phsym)call ed_ph_symmetrize_bath(bath,save=.true.)
   if(normal_bath)call ed_enforce_normal_bath(bath,save=.true.)
   !MIXING:
   if(iloop>1)Bath = wmixing*Bath + (1.d0-wmixing)*Bath_prev
   Bath_prev=Bath
   !Check convergence (if required change chemical potential)
   converged = check_convergence(Weiss(1,1,1,1,1,:),dmft_error,nsuccess,nloop,reset=.false.)
   if(nread/=0.d0)call ed_search_variable(xmu,dens,converged)
   !Close this DMFT loop
   call end_loop
enddo
```

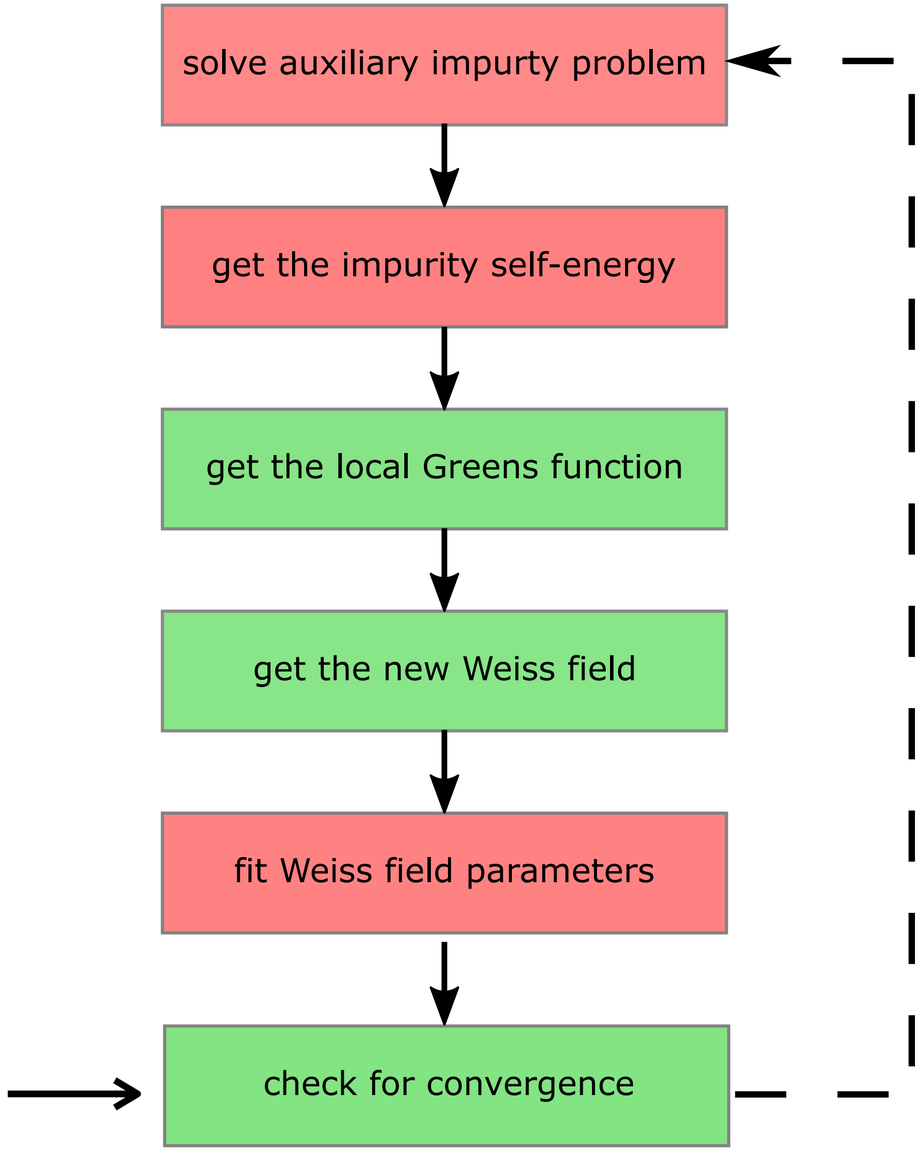
Get the new bath Parameters



ed_ahm_bethe.f90 solve the attractive Hubbard model on the be the Lattice

```
!DMFT loop
iloop=0;converged=.false.
do while(.not.converged.AND.iloop<nloop)
   iloop=iloop+1
   call start_loop(iloop,nloop,"DMFT-loop")
   !Solve the EFFECTIVE IMPURITY PROBLEM (first w/ a guess for the bath)
   call ed_solve(bath,Hloc)

   !Retrieve impurity self-energies (normal, anomalous)
   call ed_get_Sigma_matsubara(Smats(1,:,:,:,:,:,:))
   call ed_get_Self_matsubara(Smats(2,:,:,:,:,:,:))
   call ed_get_Sigma_realaxis(Sreal(1,:,:,:,:,:,:))
   call ed_get_Self_realaxis(Sreal(2,:,:,:,:,:,:))
   call ed_get_dens(dens,iorb=1)

   !Compute the local gfs:
   call dmft_gloc_matsubara(Ebethe,Dbethe,H0,Gmats,Smats)

   call dmft_self_consistency(Gmats(1,:,:,:,:,:,:),Gmats(2,:,:,:,:,:,:),&
        Smats(1,:,:,:,:,:,:),Smats(2,:,:,:,:,:,:),&
        Weiss(1,:,:,:,:,:,:),Weiss(2,:,:,:,:,:,:),&
        Hloc,trim(cg_scheme))
   call dmft_print_gf_matsubara(Gmats(1,:,:,:,:,:,:),"Gloc",iprint=1)
   call dmft_print_gf_matsubara(Gmats(2,:,:,:,:,:,:),"Floc",iprint=1)
   call dmft_print_gf_matsubara(Weiss(1,:,:,:,:,:,:),"Weiss",iprint=1)
   call dmft_print_gf_matsubara(Weiss(2,:,:,:,:,:,:),"fWeiss",iprint=1)
   !Perform the self-consistency fitting the new bath
   call ed_chi2_fitgf(Weiss,bath,ispin=1)
   !if it holds apply symmetrizations
   if(phsym)call ed_ph_symmetrize_bath(bath,save=.true.)
   if(normal_bath)call ed_enforce_normal_bath(bath,save=.true.)
   !MIXING:
   if(iloop>1)Bath = wmixing*Bath + (1.d0-wmixing)*Bath_prev
   Bath_prev=Bath
   !Check convergence (if required change chemical potential)
   converged = check_convergence(Weiss(1,1,1,1,1,:),dmft_error,nsuccess,nloop,reset=.false.)
   if(nread/=0.d0)call ed_search_variable(xmu,dens,converged)
   !Close this DMFT loop
   call end_loop
enddo
```

Convergence check

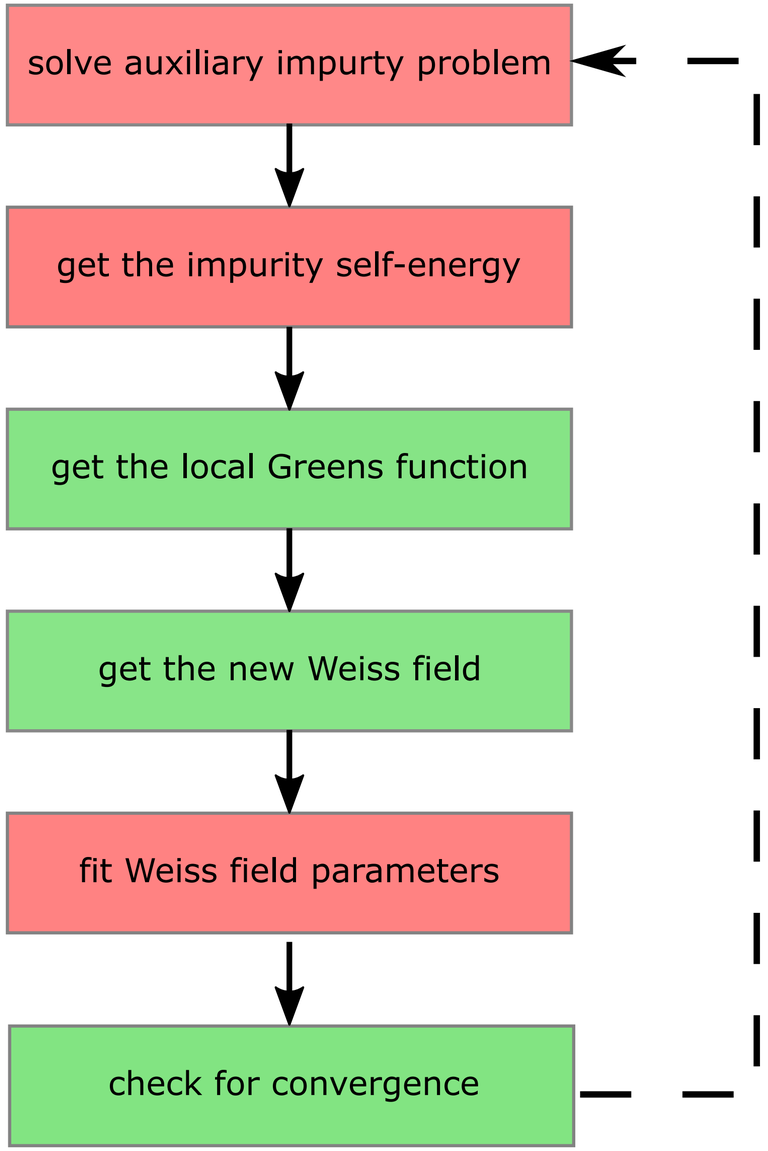


ed_ahm_bethe.f90 solve the attractive Hubbard model on the be the Lattice

```
   call end_loop
enddo
```

@convergence get GF in real frequency; Compute the kinetic energy

```
!Compute the local gfs:
call dmft_gloc_realaxis(Ebethe,Dbethe,H0,Greal,Sreal)
call dmft_print_gf_realaxis(Greal(1,:,:,:,:,:),"Gloc",iprint=1)
call dmft_print_gf_realaxis(Greal(2,:,:,:,:,:),"Floc",iprint=1)
!Compute the Kinetic Energy:
call dmft_kinetic_energy(Ebethe,Dbethe,H0,Smats(1,:,:,:,:,:),Smats(2,:,:,:,:,:))
!
```



```
WBAND=1.000000000                       !Bethe Lattice bandwidth
LE=1000                                 !Number of energy levels for Bethe DOS integration
WMIXING=7.500000000E-01                 !Mixing bath parameter
PHSYM=T                                 !Flag to enforce p-h symmetry of the bath.
NORMAL=F                                !Flag to enforce no symmetry braking in the bath.
NORB=1                                  !Number of impurity orbitals (max 5).
NBATH=6                                 !Number of bath sites:(normal=>Nbath per orb)(hybrid=>Nbath total)(replica=>Nbath=Nreplica)
NSPIN=1                                 !Number of spin degeneracy (max 2)
NPH=0                                   !Max number of phonons allowed (cut off)
BATH_TYPE=normal                        !flag to set bath type: normal (1bath/imp), hybrid(1bath), replica(1replica/imp)
ULOC=-2.050000000,0.d0,0.d0,0.d0,0.d0   !Values of the local interaction per orbital (max 5)
UST=0.d0                                !Value of the inter-orbital interaction term
JH=0.d0                                 !Hunds coupling
JX=0.d0                                 !S-E coupling
JP=0.d0                                 !P-H coupling
NLOOP=100                               !Max number of DMFT iterations.
NSUCCESS=1                              !Number of successive iterations below threshold for convergence
DMFT_ERROR=1.000000000E-05              !Error threshold for DMFT convergence
SB_FIELD=1.000000000E-01                !Value of a symmetry breaking field for magnetic solutions.
DELTASC=2.000000000E-02                 !Value of the SC symmetry breaking term.
BETA=1000.000000000                     !Inverse temperature, at T=0 is used as a IR cut-off.
XMU=0.d0                                !Chemical potential. If HFMODE=T, xmu=0 indicates half-filling condition.
G_PH=0.d0,0.d0,0.d0,0.d0,0.d0           !Electron-phonon coupling constant
W0_PH=0.d0                              !Phonon frequency
SPIN_FIELD_X=0.d0,0.d0,0.d0,0.d0,0.d0   !magnetic field per orbital coupling to X-spin component
SPIN_FIELD_Y=0.d0,0.d0,0.d0,0.d0,0.d0   !magnetic field per orbital coupling to Y-spin component
SPIN_FIELD_Z=0.d0,0.d0,0.d0,0.d0,0.d0   !magnetic field per orbital coupling to Z-spin component
EXC_FIELD=0.d0,0.d0,0.d0,0.d0           !external field coupling to exciton order parameters
PAIR_FIELD=0.d0,0.d0,0.d0,0.d0,0.d0     !pair field per orbital coupling to s-wave order parameter component
ED_MODE=superc                          !Flag to set ED type: normal=normal, superc=superconductive, nonsu2=broken SU(2)
ED_DIAG_TYPE=lanc                       !flag to select the diagonalization type: 'lanc' for Lanczos/Davidson, 'full' for Full diagonalization method
ED_FINITE_TEMP=F                        !flag to select finite temperature method. note that if T then lanc_nstates_total must be > 1
ED_TWIN=F                               !flag to reduce (T) or not (F,default) the number of visited sector using twin symmetry.
ED_SECTORS=F                            !flag to reduce sector scan for the spectrum to specific sectors +/- ed_sectors_shift.
ED_SECTORS_SHIFT=1                      !shift to ed_sectors
ED_SPARSE_H=T                           !flag to select  storage of sparse matrix H (mem--, cpu++) if TRUE, or direct on-the-fly H*v product (mem++, cpu--) if
ED_TOTAL_UD=T                           !flag to select which type of quantum numbers have to be considered: T (default) total Nup-Ndw, F orbital based Nup-Ndw
ED_SOLVE_OFFDIAG_GF=F                   !flag to select the calculation of the off-diagonal impurity GF. this is T by default if bath_type/=normal
ED_PRINT_SIGMA=T                        !flag to print impurity Self-energies
ED_PRINT_G=T                            !flag to print impurity Greens function
ED_PRINT_G0=T                           !flag to print non-interacting impurity Greens function
ED_ALL_G=T                              !flag to evaluate all the components of the impurity Green`s functions irrespective of the symmetries
ED_VERBOSE=3                            !Verbosity level: 0=almost nothing --> 5:all. Really: all
```

```
WBAND=1.000000000                       !Bethe Lattice bandwidth
LE=1000                                 !Number of energy levels for Bethe DOS integration
WMIXING=7.500000000E-01                 !Mixing bath parameter
PHSYM=T                                 !Flag to enforce p-h symmetry of the bath.
NORMAL=F                                !Flag to enforce no symmetry braking in the bath.
NORB=1                                  !Number of impurity orbitals (max 5).
NBATH=6                                 !Number of bath sites:(normal=>Nbath per orb)(hybrid=>Nbath total)(replica=>Nbath=Nreplica)
NSPIN=1                                 !Number of spin degeneracy (max 2)
NPH=0                                   !Max number of phonons allowed (cut off)
BATH_TYPE=normal                        !flag to set bath type: normal (1bath/imp), hybrid(1bath), replica(1replica/imp)
ULOC=-2.050000000,0.d0,0.d0,0.d0,0.d0   !Values of the local interaction per orbital (max 5)
UST=0.d0                                !Value of the inter-orbital interaction term
JH=0.d0                                 !Hunds coupling
JX=0.d0                                 !S-E coupling
JP=0.d0                                 !P-H coupling
NLOOP=100                               !Max number of DMFT iterations.
NSUCCESS=1                              !Number of successive iterations below threshold for convergence
DMFT_ERROR=1.000000000E-05              !Error threshold for DMFT convergence
SB_FIELD=1.000000000E-01                !Value of a symmetry breaking field for magnetic solutions.
DELTASC=2.000000000E-02                 !Value of the SC symmetry breaking term.
BETA=1000.000000000                     !Inverse temperature, at T=0 is used as a IR cut-off.
XMU=0.d0                                !Chemical potential. If HFMODE=T, xmu=0 indicates half-filling condition.
G_PH=0.d0,0.d0,0.d0,0.d0,0.d0           !Electron-phonon coupling constant
W0_PH=0.d0                              !Phonon frequency
SPIN_FIELD_X=0.d0,0.d0,0.d0,0.d0,0.d0   !magnetic field per orbital coupling to X-spin component
SPIN_FIELD_Y=0.d0,0.d0,0.d0,0.d0,0.d0   !magnetic field per orbital coupling to Y-spin component
SPIN_FIELD_Z=0.d0,0.d0,0.d0,0.d0,0.d0   !magnetic field per orbital coupling to Z-spin component
EXC_FIELD=0.d0,0.d0,0.d0,0.d0           !external field coupling to exciton order parameters
PAIR_FIELD=0.d0,0.d0,0.d0,0.d0,0.d0     !pair field per orbital coupling to s-wave order parameter component
```

MODEL AND PARAMETERS OF DMFT LOOP

```
ED_MODE=superc                          !Flag to set ED type: normal=normal, superc=superconductive, nonsu2=broken SU(2)
ED_DIAG_TYPE=lanc                       !flag to select the diagonalization type: 'lanc' for Lanczos/Davidson, 'full' for Full diagonalization method
ED_FINITE_TEMP=F                        !flag to select finite temperature method. note that if T then lanc_nstates_total must be > 1
ED_TWIN=F                               !flag to reduce (T) or not (F,default) the number of visited sector using twin symmetry.
ED_SECTORS=F                            !flag to reduce sector scan for the spectrum to specific sectors +/- ed_sectors_shift.
ED_SECTORS_SHIFT=1                      !shift to ed_sectors
ED_SPARSE_H=T                           !flag to select  storage of sparse matrix H (mem--, cpu++) if TRUE, or direct on-the-fly H*v product (mem++, cpu--) if
ED_TOTAL_UD=T                           !flag to select which type of quantum numbers have to be considered: T (default) total Nup-Ndw, F orbital based Nup-Ndw
ED_SOLVE_OFFDIAG_GF=F                   !flag to select the calculation of the off-diagonal impurity GF. this is T by default if bath_type/=normal
ED_PRINT_SIGMA=T                        !flag to print impurity Self-energies
ED_PRINT_G=T                            !flag to print impurity Greens function
ED_PRINT_G0=T                           !flag to print non-interacting impurity Greens function
ED_ALL_G=T                              !flag to evaluate all the components of the impurity Green`s functions irrespective of the symmetries
ED_VERBOSE=3                            !Verbosity level: 0=almost nothing --> 5:all. Really: all
```

```
WBAND=1.000000000                          !Bethe Lattice bandwidth
LE=1000                                    !Number of energy levels for Bethe DOS integration
WMIXING=7.500000000E-01                    !Mixing bath parameter
PHSYM=T                                    !Flag to enforce p-h symmetry of the bath.
NORMAL=F                                   !Flag to enforce no symmetry braking in the bath.
NORB=1                                     !Number of impurity orbitals (max 5).
NBATH=6                                    !Number of bath sites:(normal=>Nbath per orb)(hybrid=>Nbath total)(replica=>Nbath=Nreplica)
NSPIN=1                                    !Number of spin degeneracy (max 2)
NPH=0                                      !Max number of phonons allowed (cut off)
BATH_TYPE=normal                           !flag to set bath type: normal (1bath/imp), hybrid(1bath), replica(1replica/imp)
ULOC=-2.050000000,0.d0,0.d0,0.d0,0.d0      !Values of the local interaction per orbital (max 5)
UST=0.d0                                   !Value of the inter-orbital interaction term
JH=0.d0                                    !Hunds coupling
JX=0.d0                                    !S-E coupling
JP=0.d0                                    !P-H coupling
NLOOP=100                                  !Max number of DMFT iterations.
NSUCCESS=1                                 !Number of successive iterations below threshold for convergence
DMFT_ERROR=1.000000000E-05                 !Error threshold for DMFT convergence
SB_FIELD=1.000000000E-01                   !Value of a symmetry breaking field for magnetic solutions.
DELTASC=2.000000000E-02                    !Value of the SC symmetry breaking term.
BETA=1000.000000000                        !Inverse temperature, at T=0 is used as a IR cut-off.
XMU=0.d0                                   !Chemical potential. If HFMODE=T, xmu=0 indicates half-filling condition.
G_PH=0.d0,0.d0,0.d0,0.d0,0.d0              !Electron-phonon coupling constant
W0_PH=0.d0                                 !Phonon frequency
SPIN_FIELD_X=0.d0,0.d0,0.d0,0.d0,0.d0      !magnetic field per orbital coupling to X-spin component
SPIN_FIELD_Y=0.d0,0.d0,0.d0,0.d0,0.d0      !magnetic field per orbital coupling to Y-spin component
SPIN_FIELD_Z=0.d0,0.d0,0.d0,0.d0,0.d0      !magnetic field per orbital coupling to Z-spin component
EXC_FIELD=0.d0,0.d0,0.d0,0.d0              !external field coupling to exciton order parameters
PAIR_FIELD=0.d0,0.d0,0.d0,0.d0,0.d0        !pair field per orbital coupling to s-wave order parameter component
```

MODEL AND PARAMETERS OF DMFT LOOP

```
ED_MODE=superc                             !Flag to set ED type: normal=normal, superc=superconductive, nonsu2=broken SU(2)
ED_DIAG_TYPE=lanc                          !flag to select the diagonalization type: 'lanc' for Lanczos/Davidson, 'full' for Full diagonalization method
ED_FINITE_TEMP=F                           !flag to select finite temperature method. note that if T then lanc_nstates_total must be > 1
ED_TWIN=F                                  !flag to reduce (T) or not (F,default) the number of visited sector using twin symmetry.
ED_SECTORS=F                               !flag to reduce sector scan for the spectrum to specific sectors +/- ed_sectors_shift.
ED_SECTORS_SHIFT=1                         !shift to ed_sectors
ED_SPARSE_H=T                              !flag to select  storage of sparse matrix H (mem--, cpu++) if TRUE, or direct on-the-fly H*v product (mem++, cpu--) if
ED_TOTAL_UD=T                              !flag to select which type of quantum numbers have to be considered: T (default) total Nup-Ndw, F orbital based Nup-Nd
ED_SOLVE_OFFDIAG_GF=F                      !flag to select the calculation of the off-diagonal impurity GF. this is T by default if bath_type/=normal
ED_PRINT_SIGMA=T                           !flag to print impurity Self-energies
ED_PRINT_G=T                               !flag to print impurity Greens function
ED_PRINT_G0=T                              !flag to print non-interacting impurity Greens function
ED_ALL_G=T                                 !flag to evaluate all the components of the impurity Green`s functions irrespective of the symmetries
ED_VERBOSE=3                               !Verbosity level: 0=almost nothing --> 5:all. Really: all
```

ED-SOLVER

```
WBAND=1.000000000                       !Bethe Lattice bandwidth
LE=1000                                 !Number of energy levels for Bethe DOS integration
WMIXING=7.500000000E-01                 !Mixing bath parameter
PHSYM=T                                 !Flag to enforce p-h symmetry of the bath.
NORMAL=F                                !Flag to enforce no symmetry braking in the bath.
NORB=1                                  !Number of impurity orbitals (max 5).
NBATH=6                                 !Number of bath sites:(normal=>Nbath per orb)(hybrid=>Nbath total)(replica=>Nbath=Nreplica)
NSPIN=1                                 !Number of spin degeneracy (max 2)
NPH=0                                   !Max number of phonons allowed (cut off)
BATH_TYPE=normal                        !flag to set bath type: normal (1bath/imp), hybrid(1bath), replica(1replica/imp)
ULOC=-2.050000000,0.d0,0.d0,0.d0,0.d0   !Values of the local interaction per orbital (max 5)
UST=0.d0                                !Value of the inter-orbital interaction term
JH=0.d0                                 !Hunds coupling
JX=0.d0                                 !S-E coupling
JP=0.d0                                 !P-H coupling
NLOOP=100                               !Max number of DMFT iterations.
NSUCCESS=1                              !Number of successive iterations below threshold for convergence
DMFT_ERROR=1.000000000E-05              !Error threshold for DMFT convergence
SB_FIELD=1.000000000E-01                !Value of a symmetry breaking field for magnetic solutions.
DELTASC=2.000000000E-02                 !Value of the SC symmetry breaking term.
BETA=1000.000000000                     !Inverse temperature, at T=0 is used as a IR cut-off.
XMU=0.d0                                !Chemical potential. If HFMODE=T, xmu=0 indicates half-filling condition.
G_PH=0.d0,0.d0,0.d0,0.d0,0.d0           !Electron-phonon coupling constant
W0_PH=0.d0                              !Phonon frequency
SPIN_FIELD_X=0.d0,0.d0,0.d0,0.d0,0.d0   !magnetic field per orbital coupling to X-spin component
SPIN_FIELD_Y=0.d0,0.d0,0.d0,0.d0,0.d0   !magnetic field per orbital coupling to Y-spin component
SPIN_FIELD_Z=0.d0,0.d0,0.d0,0.d0,0.d0   !magnetic field per orbital coupling to Z-spin component
EXC_FIELD=0.d0,0.d0,0.d0,0.d0           !external field coupling to exciton order parameters
PAIR_FIELD=0.d0,0.d0,0.d0,0.d0,0.d0     !pair field per orbital coupling to s-wave order parameter component
ED_MODE=superc                          !Flag to set ED type: normal=normal, superc=superconductive, nonsu2=broken SU(2)
ED_DIAG_TYPE=lanc                       !flag to select the diagonalization type: 'lanc' for Lanczos/Davidson, 'f
ED_FINITE_TEMP=F                        !flag to select finite temperature method. note that if T then lanc_nstat
ED_TWIN=F                               !flag to reduce (T) or not (F,default) the number of visited sector using
ED_SECTORS=F                            !flag to reduce sector scan for the spectrum to specific sectors +/- ed_s
ED_SECTORS_SHIFT=1                      !shift to ed_sectors
ED_SPARSE_H=T                           !flag to select  storage of sparse matrix H (mem--, cpu++) if TRUE, or direct on-the-fly H*v product (mem++, cpu--) if
ED_TOTAL_UD=T                           !flag to select which type of quantum numbers have to be considered: T (default) total Nup-Ndw, F orbital based Nup-Ndw
ED_SOLVE_OFFDIAG_GF=F                   !flag to select the calculation of the off-diagonal impurity GF. this is T by default if bath_type/=normal
ED_PRINT_SIGMA=T                        !flag to print impurity Self-energies
ED_PRINT_G=T                            !flag to print impurity Greens function
ED_PRINT_G0=T                           !flag to print non-interacting impurity Greens function
ED_ALL_G=T                              !flag to evaluate all the components of the impurity Green`s functions irrespective of the symmetries
ED_VERBOSE=3                            !Verbosity level: 0=almost nothing --> 5:all. Really: all
```

driver

MODEL AND PARAMETERS OF DMFT LOOP

```
WBAND=1.000000000                          !Bethe Lattice bandwidth
LE=1000                                    !Number of energy levels for Bethe DOS integration
WMIXING=7.500000000E-01                    !Mixing bath parameter
PHSYM=T                                    !Flag to enforce p-h symmetry of the bath.
NORMAL=F                                   !Flag to enforce no symmetry braking in the bath.
NORB=1                                     !Number of impurity orbitals (max 5).
NBATH=6                                    !Number of bath sites:(normal=>Nbath per orb)(hybrid=>Nbath total)(replica=>Nbath=Nreplica)
NSPIN=1                                    !Number of spin degeneracy (max 2)
NPH=0                                      !Max number of phonons allowed (cut off)
BATH_TYPE=normal                           !flag to set bath type: normal (1bath/imp), hybrid(1bath), replica(1replica/imp)
ULOC=-2.050000000,0.d0,0.d0,0.d0,0.d0      !Values of the local interaction per orbital (max 5)
UST=0.d0                                   !Value of the inter-orbital interaction term
JH=0.d0                                    !Hunds coupling
JX=0.d0                                    !S-E coupling
JP=0.d0                                    !P-H coupling
NLOOP=100                                  !Max number of DMFT iterations.
NSUCCESS=1                                 !Number of successive iterations below threshold for convergence
DMFT_ERROR=1.000000000E-05                 !Error threshold for DMFT convergence
SB_FIELD=1.000000000E-01                   !Value of a symmetry breaking field for magnetic solutions.
DELTASC=2.000000000E-02                    !Value of the SC symmetry breaking term.
BETA=1000.000000000                        !Inverse temperature, at T=0 is used as a IR cut-off.
XMU=0.d0                                   !Chemical potential. If HFMODE=T, xmu=0 indicates half-filling condition.
G_PH=0.d0,0.d0,0.d0,0.d0,0.d0              !Electron-phonon coupling constant
W0_PH=0.d0                                 !Phonon frequency
SPIN_FIELD_X=0.d0,0.d0,0.d0,0.d0,0.d0      !magnetic field per orbital coupling to X-spin component
SPIN_FIELD_Y=0.d0,0.d0,0.d0,0.d0,0.d0      !magnetic field per orbital coupling to Y-spin component
SPIN_FIELD_Z=0.d0,0.d0,0.d0,0.d0,0.d0      !magnetic field per orbital coupling to Z-spin component
EXC_FIELD=0.d0,0.d0,0.d0,0.d0              !external field coupling to exciton order parameters
PAIR_FIELD=0.d0,0.d0,0.d0,0.d0,0.d0,0.d0   !pair field per orbital coupling to s-wave order parameter component
ED_MODE=superc                             !Flag to set ED type: normal=normal, superc=superconductive, nonsu2=broken SU(2)
ED_DIAG_TYPE=lanc                          !flag to select the diagonalization type: 'lanc' for Lanczos/Davidson, 'f
ED_FINITE_TEMP=F                           !flag to select finite temperature method. note that if T then lanc_nstat
ED_TWIN=F                                  !flag to reduce (T) or not (F,default) the number of visited sector using
ED_SECTORS=F                               !flag to reduce sector scan for the spectrum to specific sectors +/- ed_s
ED_SECTORS_SHIFT=1                         !shift to ed_sectors
ED_SPARSE_H=T                              !flag to select  storage of sparse matrix H (mem--, cpu++) if TRUE, or direct on-the-fly H*v product (mem++, cpu--) if
ED_TOTAL_UD=T                              !flag to select which type of quantum numbers have to be considered: T (default) total Nup-Ndw, F orbital based Nup-Ndw
ED_SOLVE_OFFDIAG_GF=F                      !flag to select the calculation of the off-diagonal impurity GF. this is T by default if bath_type/=normal
ED_PRINT_SIGMA=T                           !flag to print impurity Self-energies
ED_PRINT_G=T                               !flag to print impurity Greens function
ED_PRINT_G0=T                              !flag to print non-interacting impurity Greens function
ED_ALL_G=T                                 !flag to evaluate all the components of the impurity Green`s functions irrespective of the symmetries
ED_VERBOSE=3                               !Verbosity level: 0=almost nothing --> 5:all. Really: all
```

Number of orbitals
And number of sites in the bath

MODEL AND PARAMETERS OF DMFT LOOP

```
WBAND=1.000000000                          !Bethe Lattice bandwidth
LE=1000                                    !Number of energy levels for Bethe DOS integration
WMIXING=7.500000000E-01                    !Mixing bath parameter
PHSYM=T                                    !Flag to enforce p-h symmetry of the bath.
NORMAL=F                                   !Flag to enforce no symmetry braking in the bath.
NORB=1                                     !Number of impurity orbitals (max 5).
NBATH=6                                    !Number of bath sites:(normal=>Nbath per orb)(hybrid=>Nbath total)(replica=>Nbath=Nreplica)
NSPIN=1                                    !Number of spin degeneracy (max 2)
NPH=0                                      !Max number of phonons allowed (cut off)
BATH_TYPE=normal                           !flag to set bath type: normal (1bath/imp), hybrid(1bath), replica(1replica/imp)
ULOC=-2.050000000,0.d0,0.d0,0.d0,0.d0      !Values of the local interaction per orbital (max 5)
UST=0.d0                                   !Value of the inter-orbital interaction term
JH=0.d0                                    !Hunds coupling
JX=0.d0                                    !S-E coupling
JP=0.d0                                    !P-H coupling
NLOOP=100                                  !Max number of DMFT iterations.
NSUCCESS=1                                 !Number of successive iterations below threshold for convergence
DMFT_ERROR=1.000000000E-05                 !Error threshold for DMFT convergence
SB_FIELD=1.000000000E-01                   !Value of a symmetry breaking field for magnetic solutions.
DELTASC=2.000000000E-02                    !Value of the SC symmetry breaking term.
BETA=1000.000000000                        !Inverse temperature, at T=0 is used as a IR cut-off.
XMU=0.d0                                   !Chemical potential. If HFMODE=T, xmu=0 indicates half-filling condition.
G_PH=0.d0,0.d0,0.d0,0.d0,0.d0              !Electron-phonon coupling constant
W0_PH=0.d0                                 !Phonon frequency
SPIN_FIELD_X=0.d0,0.d0,0.d0,0.d0,0.d0      !magnetic field per orbital coupling to X-spin component
SPIN_FIELD_Y=0.d0,0.d0,0.d0,0.d0,0.d0      !magnetic field per orbital coupling to Y-spin component
SPIN_FIELD_Z=0.d0,0.d0,0.d0,0.d0,0.d0      !magnetic field per orbital coupling to Z-spin component
EXC_FIELD=0.d0,0.d0,0.d0,0.d0              !external field coupling to exciton order parameters
PAIR_FIELD=0.d0,0.d0,0.d0,0.d0,0.d0,0.d0   !pair field per orbital coupling to s-wave order parameter component
ED_MODE=superc                             !Flag to set ED type: normal=normal, superc=superconductive, nonsu2=broken SU(2)
ED_DIAG_TYPE=lanc                          !flag to select the diagonalization type: 'lanc' for Lanczos/Davidson, 'f
ED_FINITE_TEMP=F                           !flag to select finite temperature method. note that if T then lanc_nstat
ED_TWIN=F                                  !flag to reduce (T) or not (F,default) the number of visited sector using
ED_SECTORS=F                               !flag to reduce sector scan for the spectrum to specific sectors +/- ed_s
ED_SECTORS_SHIFT=1                         !shift to ed_sectors
ED_SPARSE_H=T                              !flag to select  storage of sparse matrix H (mem--, cpu++) if TRUE, or direct on-the-fly H*v product (mem++, cpu--) if
ED_TOTAL_UD=T                              !flag to select which type of quantum numbers have to be considered: T (default) total Nup-Ndw, F orbital based Nup-Ndw
ED_SOLVE_OFFDIAG_GF=F                      !flag to select the calculation of the off-diagonal impurity GF. this is T by default if bath_type/=normal
ED_PRINT_SIGMA=T                           !flag to print impurity Self-energies
ED_PRINT_G=T                               !flag to print impurity Greens function
ED_PRINT_G0=T                              !flag to print non-interacting impurity Greens function
ED_ALL_G=T                                 !flag to evaluate all the components of the impurity Green`s functions irrespective of the symmetries
ED_VERBOSE=3                               !Verbosity level: 0=almost nothing --> 5:all. Really: all
```

Interaction Hamiltonian

MODEL AND PARAMETERS OF DMFT LOOP

```
WBAND=1.000000000                          !Bethe Lattice bandwidth
LE=1000                                    !Number of energy levels for Bethe DOS integration
WMIXING=7.500000000E-01                    !Mixing bath parameter
PHSYM=T                                    !Flag to enforce p-h symmetry of the bath.
NORMAL=F                                   !Flag to enforce no symmetry braking in the bath.
NORB=1                                     !Number of impurity orbitals (max 5).
NBATH=6                                    !Number of bath sites:(normal=>Nbath per orb)(hybrid=>Nbath total)(replica=>Nbath=Nreplica)
NSPIN=1                                    !Number of spin degeneracy (max 2)
NPH=0                                      !Max number of phonons allowed (cut off)
BATH_TYPE=normal                           !flag to set bath type: normal (1bath/imp), hybrid(1bath), replica(1replica/imp)
ULOC=-2.050000000,0.d0,0.d0,0.d0,0.d0      !Values of the local interaction per orbital (max 5)
UST=0.d0                                   !Value of the inter-orbital interaction term
JH=0.d0                                    !Hunds coupling
JX=0.d0                                    !S-E coupling
JP=0.d0                                    !P-H coupling
NLOOP=100                                  !Max number of DMFT iterations.
NSUCCESS=1                                 !Number of successive iterations below threshold for convergence
DMFT_ERROR=1.000000000E-05                 !Error threshold for DMFT convergence
SB_FIELD=1.000000000E-01                   !Value of a symmetry breaking field for magnetic solutions.
DELTASC=2.000000000E-02                    !Value of the SC symmetry breaking term.
BETA=1000.000000000                        !Inverse temperature, at T=0 is used as a IR cut-off.
XMU=0.d0                                   !Chemical potential. If HFMODE=T, xmu=0 indicates half-filling condition.
G_PH=0.d0,0.d0,0.d0,0.d0,0.d0              !Electron-phonon coupling constant
W0_PH=0.d0                                 !Phonon frequency
SPIN_FIELD_X=0.d0,0.d0,0.d0,0.d0,0.d0      !magnetic field per orbital coupling to X-spin component
SPIN_FIELD_Y=0.d0,0.d0,0.d0,0.d0,0.d0      !magnetic field per orbital coupling to Y-spin component
SPIN_FIELD_Z=0.d0,0.d0,0.d0,0.d0,0.d0      !magnetic field per orbital coupling to Z-spin component
EXC_FIELD=0.d0,0.d0,0.d0,0.d0              !external field coupling to exciton order parameters
PAIR_FIELD=0.d0,0.d0,0.d0,0.d0,0.d0,0.d0   !pair field per orbital coupling to s-wave order parameter component
ED_MODE=superc                             !Flag to set ED type: normal=normal, superc=superconductive, nonsu2=broken SU(2)
ED_DIAG_TYPE=lanc                          !flag to select the diagonalization type: 'lanc' for Lanczos/Davidson, 'f
ED_FINITE_TEMP=F                           !flag to select finite temperature method. note that if T then lanc_nstat
ED_TWIN=F                                  !flag to reduce (T) or not (F,default) the number of visited sector using
ED_SECTORS=F                               !flag to reduce sector scan for the spectrum to specific sectors +/- ed_s
ED_SECTORS_SHIFT=1                         !shift to ed_sectors
ED_SPARSE_H=T                              !flag to select  storage of sparse matrix H (mem--, cpu++) if TRUE, or direct on-the-fly H*v product (mem++, cpu--) if
ED_TOTAL_UD=T                              !flag to select which type of quantum numbers have to be considered: T (default) total Nup-Ndw, F orbital based Nup-Ndw
ED_SOLVE_OFFDIAG_GF=F                      !flag to select the calculation of the off-diagonal impurity GF. this is T by default if bath_type/=normal
ED_PRINT_SIGMA=T                           !flag to print impurity Self-energies
ED_PRINT_G=T                               !flag to print impurity Greens function
ED_PRINT_G0=T                              !flag to print non-interacting impurity Greens function
ED_ALL_G=T                                 !flag to evaluate all the components of the impurity Green`s functions irrespective of the symmetries
ED_VERBOSE=3                               !Verbosity level: 0=almost nothing --> 5:all. Really: all
```

Error in DFMT loops

MODEL AND PARAMETERS OF DMFT LOOP

```
WBAND=1.000000000                       !Bethe Lattice bandwidth
LE=1000                                 !Number of energy levels for Bethe DOS integration
WMIXING=7.500000000E-01                 !Mixing bath parameter
PHSYM=T                                 !Flag to enforce p-h symmetry of the bath.
NORMAL=F                                !Flag to enforce no symmetry braking in the bath.
NORB=1                                  !Number of impurity orbitals (max 5).
NBATH=6                                 !Number of bath sites:(normal=>Nbath per orb)(hybrid=>Nbath total)(replica=>Nbath=Nreplica)
NSPIN=1                                 !Number of spin degeneracy (max 2)
NPH=0                                   !Max number of phonons allowed (cut off)
BATH_TYPE=normal                        !flag to set bath type: normal (1bath/imp), hybrid(1bath), replica(1replica/imp)
ULOC=-2.050000000,0.d0,0.d0,0.d0,0.d0   !Values of the local interaction per orbital (max 5)
UST=0.d0                                !Value of the inter-orbital interaction term
JH=0.d0                                 !Hunds coupling
JX=0.d0                                 !S-E coupling
JP=0.d0                                 !P-H coupling
NLOOP=100                               !Max number of DMFT iterations.
NSUCCESS=1                              !Number of successive iterations below threshold for convergence
DMFT_ERROR=1.000000000E-05              !Error threshold for DMFT convergence
SB_FIELD=1.000000000E-01                !Value of a symmetry breaking field for magnetic solutions.
DELTASC=2.000000000E-02                 !Value of the SC symmetry breaking term.
BETA=1000.000000000                     !Inverse temperature, at T=0 is used as a IR cut-off.
XMU=0.d0                                !Chemical potential. If HFMODE=T, xmu=0 indicates half-filling condition.
G_PH=0.d0,0.d0,0.d0,0.d0,0.d0           !Electron-phonon coupling constant
W0_PH=0.d0                              !Phonon frequency
SPIN_FIELD_X=0.d0,0.d0,0.d0,0.d0,0.d0   !magnetic field per orbital coupling to X-spin component
SPIN_FIELD_Y=0.d0,0.d0,0.d0,0.d0,0.d0   !magnetic field per orbital coupling to Y-spin component
SPIN_FIELD_Z=0.d0,0.d0,0.d0,0.d0,0.d0   !magnetic field per orbital coupling to Z-spin component
EXC_FIELD=0.d0,0.d0,0.d0,0.d0           !external field coupling to exciton order parameters
PAIR_FIELD=0.d0,0.d0,0.d0,0.d0,0.d0,0.d0 !pair field per orbital coupling to s-wave order parameter component
ED_MODE=superc                          !Flag to set ED type: normal=normal, superc=superconductive, nonsu2=broken SU(2)
ED_DIAG_TYPE=lanc                       !flag to select the diagonalization type: 'lanc' for Lanczos/Davidson, 'f
ED_FINITE_TEMP=F                        !flag to select finite temperature method. note that if T then lanc_nstat
ED_TWIN=F                               !flag to reduce (T) or not (F,default) the number of visited sector using
ED_SECTORS=F                            !flag to reduce sector scan for the spectrum to specific sectors +/- ed_s
ED_SECTORS_SHIFT=1                      !shift to ed_sectors
```

Temperature(*)
chemical potential
symmetry breaking fields

MODEL AND PARAMETERS OF
DMFT LOOP

(*) Temperature: here Temperature is just an IR cut-off for smooth fermi functions; See below for ACTUAL FINITE T calculations

ED_MODE = super/normal

ED_FINITE_TEMP =F/T

```
BETA=1000.000000000                !Inverse temperature, at T=0 is used as a IR cut-off.
XMU=0.d0                           !Chemical potential. If HFMODE=T, xmu=0 indicates half-filling condition.
G_PH=0.d0,0.d0,0.d0,0.d0,0.d0      !Electron-phonon coupling constant
W0_PH=0.d0                         !Phonon frequency
SPIN_FIELD_X=0.d0,0.d0,0.d0,0.d0,0.d0  !magnetic field per orbital coupling to X-spin component
SPIN_FIELD_Y=0.d0,0.d0,0.d0,0.d0,0.d0  !magnetic field per orbital coupling to Y-spin component
SPIN_FIELD_Z=0.d0,0.d0,0.d0,0.d0,0.d0  !magnetic field per orbital coupling to Z-spin component
EXC_FIELD=0.d0,0.d0,0.d0,0.d0      !external field coupling to exciton order parameters
PAIR_FIELD=0.d0,0.d0,0.d0,0.d0,0.d0  !pair field per orbital coupling to s-wave order parameter component
ED_MODE=superc                     !Flag to set ED type: normal=normal, superc=superconductive, nonsu2=broken SU(2)
ED_DIAG_TYPE=lanc                  !flag to select the diagonalization type: 'lanc' for Lanczos/Davidson, 'full' for Full diagonalization method
ED_FINITE_TEMP=F                   !flag to select finite temperature method. note that if T then lanc_nstates_total must be > 1
ED_TWIN=F                          !flag to reduce (T) or not (F,default) the number of visited sector using twin symmetry.
ED_SECTORS=F                       !flag to reduce sector scan for the spectrum to specific sectors +/- ed_sectors_shift.
ED_SECTORS_SHIFT=1                 !shift to ed_sectors
ED_SPARSE_H=T                      !flag to select  storage of sparse matrix H (mem--, cpu++) if TRUE, or direct on-the-fly H*v product (mem++, cpu--) if FALSE
ED_TOTAL_UD=T                      !flag to select which type of quantum numbers have to be considered: T (default) total Nup-Ndw, F orbital based Nup-Ndw
ED_SOLVE_OFFDIAG_GF=F              !flag to select the calculation of the off-diagonal impurity GF. this is T by default if bath_type/=normal
ED_PRINT_SIGMA=T                   !flag to print impurity Self-energies
ED_PRINT_G=T                       !flag to print impurity Greens function
ED_PRINT_G0=T                      !flag to print non-interacting impurity Greens function
ED_ALL_G=T                         !flag to evaluate all the components of the impurity Green`s functions irrespective of the symmetries
ED_VERBOSE=3                       !Verbosity level: 0=almost nothing --> 5:all. Really: all
ED_HW_BATH=2.000000000             !half-bandwidth for the bath initialization: flat in -hwband:hwband
ED_OFFSET_BATH=1.000000000E-01     !offset for the initialization of diagonal terms in replica bath: -offset:offset
LMATS=4096                         !Number of Matsubara frequencies.
LREAL=5000                         !Number of real-axis frequencies.
LTAU=1024                          !Number of imaginary time points.
LFIT=1000                          !Number of Matsubara frequencies used in the \Chi2 fit.
LPOS=100                           !Number of points for the lattice PDF.
```

```
CHIPAIR_FLAG=F                          !Flag to activate pair susceptibility calculation.
CHIEXCT_FLAG=F                          !Flag to activate excitonis susceptibility calculation.
HFMODE=T                                !Flag to set the Hartree form of the interaction (n-1/2). see xmu.
EPS=1.000000000E-02                     !Broadening on the real-axis.
CUTOFF=1.000000000E-09                  !Spectrum cut-off, used to determine the number states to be retained.
GS_THRESHOLD=1.000000000E-09             !Energy threshold for ground state degeneracy loop up
LANC_METHOD=arpack                      !select the lanczos method to be used in the determination of the spectrum  ARPACK (default), LANCZOS (T=0 only), DVDSON (no M
LANC_NSTATES_SECTOR=1                   !Initial number of states per sector to be determined.
LANC_NSTATES_TOTAL=1                    !Initial number of total states to be determined.
LANC_NSTATES_STEP=2                     !Number of states added to the spectrum at each step.
LANC_NCV_FACTOR=10                      !Set the size of the block used in Lanczos-Arpack by multiplying the required Neigen (Ncv=lanc_ncv_factor*Neigen+lanc_ncv_add)
LANC_NCV_ADD=0                          !Adds up to the size of the block to prevent it to become too small (Ncv=lanc_ncv_factor*Neigen+lanc_ncv_add)
LANC_NITER=1024                         !Number of Lanczos iteration in spectrum determination.
LANC_NGFITER=400                        !Number of Lanczos iteration in GF determination. Number of momenta.
LANC_TOLERANCE=1.000000000E-18          !Tolerance for the Lanczos iterations as used in Arpack and plain lanczos.
LANC_DIM_THRESHOLD=1024                 !Min dimension threshold to use Lanczos determination of the spectrum rather than Lapack based exact diagonalization.
CG_METHOD=1                             !Conjugate-Gradient method: 0=NR, 1=minimize.
CG_GRAD=0                               !Gradient evaluation method: 0=analytic (default), 1=numeric.
CG_FTOL=1.000000000E-05                 !Conjugate-Gradient tolerance.
CG_STOP=0                               !Conjugate-Gradient stopping condition: 0-3, 0=C1.AND.C2, 1=C1, 2=C2 with C1=|F_n-1 -F_n|<tol*(1+F_n), C2=||x_n-1 -x_n||<tol*(
x_n||).
CG_NITER=500                            !Max. number of Conjugate-Gradient iterations.
CG_WEIGHT=1                             !Conjugate-Gradient weight form: 1=1.0, 2=1/n , 3=1/w_n.
CG_SCHEME=delta                         !Conjugate-Gradient fit scheme: delta or weiss.
CG_POW=2                                !Fit power for the calculation of the Chi distance function as 1/L*|G0 - G0and|**cg_pow
CG_MINIMIZE_VER=F                       !Flag to pick old/.false. (Krauth) or new/.true. (Lichtenstein) version of the minimize CG routine
CG_MINIMIZE_HH=1.000000000E-04          !Unknown parameter used in the CG minimize procedure.
JZ_BASIS=F                              !
JZ_MAX=F                                !
```

```
CHIPAIR_FLAG=F                          !Flag to activate pair susceptibility calculation.
CHIEXCT_FLAG=F                          !Flag to activate excitonis susceptibility calculation.
HFMODE=T                                !Flag to set the Hartree form of the interaction (n-1/2). see xmu.
EPS=1.000000000E-02                     !Broadening on the real-axis.
CUTOFF=1.000000000E-09                  !Spectrum cut-off, used to determine the number states to be retained.
GS_THRESHOLD=1.000000000E-09            !Energy threshold for ground state degeneracy loop up
LANC_METHOD=arpack                      !select the lanczos method to be used in the determination of the spectrum. ARPACK (default), LANCZOS (T=0 only), DVDSON (no M
LANC_NSTATES_SECTOR=1                   !Initial number of states per sector to be determined.
LANC_NSTATES_TOTAL=1                    !Initial number of total states to be determined.
LANC_NSTATES_STEP=2                     !Number of states added to the spectrum at each step.
LANC_NCV_FACTOR=10                      !Set the size of the block used in Lanczos-Arpack by multiplying the required Neigen (Ncv=lanc_ncv_factor*Neigen+lanc_ncv_add)
LANC_NCV_ADD=0                          !Adds up to the size of the block to prevent it to become too small (Ncv=lanc_ncv_factor*Neigen+lanc_ncv_add)
LANC_NITER=1024                         !Number of Lanczos iteration in spectrum determination.
LANC_NGFITER=400                        !Number of Lanczos iteration in GF determination. Number of momenta.
LANC_TOLERANCE=1.000000000E-18          !Tolerance for the Lanczos iterations as used in Arpack and plain lanczos.
LANC_DIM_THRESHOLD=1024                 !Min dimension threshold to use Lanczos determination of the spectrum rather than Lapack based exact diagonalization.
CG_METHOD=1                             !Conjugate-Gradient method: 0=NR, 1=minimize.
CG_GRAD=0                               !Gradient evaluation method: 0=analytic (default), 1=numeric.
CG_FTOL=1.000000000E-05                 !Conjugate-Gradient tolerance.
CG_STOP=0                               !Conjugate-Gradient stopping condition: 0-3, 0=C1.AND.C2, 1=C1, 2=C2 with C1=|F_n-1 -F_n|<tol*(1+F_n), C2=||x_n-1 -x_n||<tol*(
x_n||).
CG_NITER=500                            !Max. number of Conjugate-Gradient iterations.
CG_WEIGHT=1                             !Conjugate-Gradient weight form: 1=1.0, 2=1/n , 3=1/w_n.
CG_SCHEME=delta                         !Conjugate-Gradient fit scheme: delta or weiss.
CG_POW=2                                !Fit power for the calculation of the Chi distance function as 1/L*|G0 - G0and|**cg_pow
CG_MINIMIZE_VER=F                       !Flag to pick old/.false. (Krauth) or new/.true. (Lichtenstein) version of the minimize CG routine
CG_MINIMIZE_HH=1.000000000E-04          !Unknown parameter used in the CG minimize procedure.
JZ_BASIS=F                              !
JZ_MAX=F                                !
```

FIT OF BATH PARAMETERS

In the DATA/init_run/ directory

```
[giacomo@hyperion 15:28 init_run (master)]$ ls -1
hamiltonian.restart
inputAHM.conf
run.sh
```

A reasonable guess for the parameters in the impurity hamiltonian

Input file

Bash script to run the code recursively

$$H_{imp}^{SC} = U n_{i\uparrow} n_{i\downarrow} - \mu \sum_{\sigma} n_{i\sigma} + \sum_{l\sigma} v_l c_{i\sigma}^{\dagger} d_{l\sigma} + h.c. + \sum_{l} \varepsilon_l d_{l\sigma}^{\dagger} d_{l\sigma} + \sum_{l} \Delta_l d_{l\uparrow}^{\dagger} d_{l\downarrow}^{\dagger} + h.c$$

```
[giacomo@hyperion 15:31 init_run (master)]$ cat hamiltonian.restart
            #Ek_l1_s1                  Dk_l1_s1                  Vk_l1_s1
  -7.350264028261E-01   -1.165133274030E+00   -2.605904767951E-01
  -4.082465585322E-01   -3.651036562198E-01    1.500611517845E-01
  -5.539808117348E-02   -5.612699572416E-01    1.850262221085E-01
   5.539808117348E-02   -5.612699572416E-01    1.850262221085E-01
   4.082465585322E-01   -3.651036562198E-01    1.500611517845E-01
   7.350264028261E-01   -1.165133274030E+00   -2.605904767951E-01
[giacomo@hyperion 15:31 init_run (master)]$
```

Run a single DMFT loop, ie Nloop=1 and read the output

```
INIT SOLVER FOR
Summary:
-----------------------------------------
# of levels/spin      =              7
Total size            =             14
# of impurities       =              1
# of bath/impurity    =              6
# of Bath levels/spin =              6
# of  sectors         =             15
Ns_Orb                =              7
Ns_Ud                 =              1
Nph                   =              0
Largest Sector(s)     =           3432
-----------------------------------------
```

Initialize the bath

```
-----DMFT-loop    1 (max:  100)-----
(  0.0000,  0.0000)

           #Ek_l1_s1              Dk_l1_s1              Vk_l1_s1
  -7.350264028261E-01   -1.165133274030E+00   -2.605904767951E-01
  -4.082465585322E-01   -3.651036562198E-01    1.500611517845E-01
  -5.539808117348E-02   -5.612699572416E-01    1.850262221085E-01
   5.539808117348E-02   -5.612699572416E-01    1.850262221085E-01
   4.082465585322E-01   -3.651036562198E-01    1.500611517845E-01
   7.350264028261E-01   -1.165133274030E+00   -2.605904767951E-01
Diagonalize impurity problem:
Diagonalize impurity H:
        1-Solving sector:        1   sz:  -7  dim=                  1
Total time [h:m:s.ms]:   0:00:00.000

        2-Solving sector:        2   sz:  -6  dim=                 14
Total time [h:m:s.ms]:   0:00:00.000

        3-Solving sector:        3   sz:  -5  dim=                 91
Total time [h:m:s.ms]:   0:00:00.002

        4-Solving sector:        4   sz:  -4  dim=                364
Total time [h:m:s.ms]:   0:00:00.072
```

start the diagonalization

```
Total time [h:m:s.ms]:   0:00:03.390
# i       E_i           exp(-(E-E0)/T)       Sz     Sect     Dim
     1   -5.720735175477   1.000000000000E+00   0      8      3432
Egs =     -5.720735175477   0
Z   =      1.000000000000
Get impurity Greens functions:
Get G&F_l1_s1
From sector  :     8     0
        apply c^+_a,up:  1
          apply c_a,up: -1
          apply c_a,dw:  1
        apply c^+_a,dw: -1
apply c^+_a,up + c_a,dw  1
apply c_a,up + c^+_a,dw -1
Total time [h:m:s.ms]:   0:00:00.511
Get observables:
dens=    1.000000000000    1.000000000000
docc=    0.442171769401
phi =   -0.426362048662   -0.852724097325
Get local energy:
<Hint>  =   -0.384343538802
<V>     =   -0.884343538802
<E0>    =    0.000000000000
<Ehf>   =    0.500000000000
Dust    =    0.000000000000
Dund    =    0.000000000000
Dse     =    0.000000000000
Dph     =    0.000000000000
```

end of the diagonalization

use LANCZOS to compute GF

$\langle n_i \rangle$

$\langle n_{i\uparrow} n_{i\downarrow} \rangle$

$\langle c^\dagger_{i\uparrow} c^\dagger_{i\downarrow} \rangle$

energy $\langle H_{int} \rangle$

obsevables

```
Get local Matsubara Superc Green's function (no print)

  +ETA --> fort.6
  10% |ETA: 0:00:03.965 @21 November 2021  22:36:10
  20% |ETA: 0:00:03.853
  30% |ETA: 0:00:03.832                          get local GF
  40% |ETA: 0:00:02.726
  50% |ETA: 0:00:02.595 @21 November 2021  22:36:11
  60% |ETA: 0:00:01.944
  70% |ETA: 0:00:01.702
  80% |ETA: 0:00:00.952
  90% |ETA: 0:00:00.951
 100% |ETA: 0:00:00.000                                  print local GF/WF/Sigma
Total time [h:m:s.ms]:   0:00:04.770
Gloc dmft_gfio: write spin-orbital diagonal elements.
Floc dmft_gfio: write spin-orbital diagonal elements.
Weiss dmft_gfio: write spin-orbital diagonal elements.
fWeiss dmft_gfio: write spin-orbital diagonal elements.
\Chi2 fit with CG-minimize and CG-weight: 1 on: delta        FIT
chi^2|iter=   5.428583883E-08 |    18  <--  Orb1 Spin1
            #Ek_l1_s1               Dk_l1_s1               Vk_l1_s1
  -7.368862472260E-01   -1.163963969987E+00   -2.549103937907E-01
  -4.106578097561E-01   -3.644053221024E-01    1.580654889513E-01
  -5.575033227078E-02   -5.603308273612E-01    1.863142740882E-01
   5.570751126489E-02   -5.603379748694E-01    1.862918353871E-01
   4.106364322785E-01   -3.644328121463E-01    1.579171693689E-01
   7.368634304659E-01   -1.163974976009E+00   -2.548497602797E-01
^[[1;91merror=^[[0m  1.0000000E+00    new bath parameters  after fit
====================================
Total time [h:m:s.ms]:   0:00:08.910      END of the dmft loop
```

release Nloop=1 and
bring the DMFT loop at convergence

output of files

```
[giacomo@work-laptop 15:34 uloc-2.00 (master)]$ ls
Floc_l11_s1_iw.dat          eigenvalues_list.ed       impF0_l11_s1_realw.ed     inputAHM.conf
Floc_l11_s1_realw.dat       energy_info.ed            impF_l11_s1_iw.ed         observables_all.ed
Gloc_l11_s1_iw.dat          energy_last.ed            impF_l11_s1_realw.ed      observables_info.ed
Gloc_l11_s1_realw.dat       error.err                 impG0_l11_s1_iw.ed        observables_last.ed
Hseed                       fWeiss_l11_s1_iw.dat      impG0_l11_s1_realw.ed     parameters_info.ed
LOG.file                    fit_delta_l1_s1.ed        impG_l11_s1_iw.ed         parameters_last.ed
Weiss_l11_s1_iw.dat         fit_fdelta_l1_s1.ed       impG_l11_s1_realw.ed      scifor_version.inc
chi2fit_results_orb1_s1.ed  gfmatrix.restart          impSelf_l11_s1_iw.ed      sectors.restart
code_version.inc            hamiltonian.restart       impSelf_l11_s1_realw.ed   state_list.ed
dmft_kinetic_energy.dat     hamiltonian.used          impSigma_l11_s1_iw.ed     used.inputAHM.conf
dmft_kinetic_energy.info    impF0_l11_s1_iw.ed        impSigma_l11_s1_realw.ed
[giacomo@work-laptop 15:34 uloc-2.00 (master)]$
```

ex: local observables

$ cat observables_info.ed

```
[giacomo@work-laptop 16:09 uloc-2.00 (master)]$ cat observables_info.ed
#   1dens_1          2phi_1          3docc_1          4nup_1          5ndw_1          6mag_1          7s2          8egs
    9sz2_11         10n2_11         11z_1s1       13sig_1s1
```

$ cat observables_all.ed

```
[giacomo@work-laptop 16:10 uloc-2.00 (master)]$
[giacomo@work-laptop 16:10 uloc-2.00 (master)]$ cat observables_all.ed
   1.000000000    -0.426362049     0.442171769     0.500000000     0.500000000    -0.000000000     0.028914115    -5.720735175
   0.028914115     1.884343539     0.848682115    -0.000000065
   1.000000000    -0.424653041     0.441228190     0.500000000     0.500000000     0.000000000     0.029385905    -5.723766751
   0.029385905     1.882456381     0.845667019    -0.000000066
   1.000000000    -0.424274534     0.441053468     0.500000000     0.500000000    -0.000000000     0.029473266    -5.711271785
   0.029473266     1.882106935     0.844028263    -0.000000068
   1.000000000    -0.424079395     0.440967081     0.500000000     0.500000000     0.000000000     0.029516459    -5.703120288
   0.029516459     1.881934163     0.843045662    -0.000000069
   1.000000000    -0.423955505     0.440905505     0.500000000     0.500000000    -0.000000000     0.029547247    -5.701530335
   0.029547247     1.881811010     0.842643217    -0.000000069
   1.000000000    -0.423902823     0.440881897     0.500000000     0.500000000     0.000000000     0.029559051    -5.700595511
   0.029559051     1.881763794     0.842387332    -0.000000069
   1.000000000    -0.423877210     0.440871437     0.500000000     0.500000000     0.000000000     0.029564282    -5.700289121
   0.029564282     1.881742873     0.842247855    -0.000000069
   1.000000000    -0.423878366     0.440874708     0.500000000     0.500000000    -0.000000000     0.029562646    -5.699609901
   0.029562646     1.881749415     0.842169190    -0.000000069
   1.000000000    -0.423868434     0.440871112     0.500000000     0.500000000     0.000000000     0.029564444    -5.699478928
   0.029564444     1.881742224     0.842103932    -0.000000069
```

ex: local greens function

```
[giacomo@work-laptop 16:06 uloc-2.00 (master)]$
[giacomo@work-laptop 16:06 uloc-2.00 (master)]$ more Gloc_l11_s1_iw.dat
   3.1415926535897933E-003  -5.8464619279676357E-003   3.5516409962760981E-014
   9.4247779607693795E-003  -1.7535285504820551E-002   3.5539022354320882E-014
   1.5707963267948967E-002  -2.9211819302948156E-002   3.5234510621651172E-014
   2.1991148575128554E-002  -4.0867906932570239E-002   3.4722712986129389E-014
   2.8274333882308138E-002  -5.2495446968694640E-002   3.5384753937703350E-014
   3.4557519189487726E-002  -6.4086414480951412E-002   3.5313122055419949E-014
   4.0840704496667310E-002  -7.5632882113037023E-002   3.5617349185019381E-014
   4.7123889803846901E-002  -8.7127040591108998E-002   3.5608851750492526E-014
   5.3407075111026485E-002  -9.8561218546315860E-002   3.5544253629803124E-014
   5.9690260418206076E-002 -0.10992790154358732        3.6191108974698710E-014
   6.5973445725385660E-002 -0.12121975021676776        3.6046998177184653E-014
   7.2256631032565244E-002 -0.13242961741897513        3.6949074713483326E-014
   7.8539816339744828E-002 -0.14355056430665492        3.7058253872252617E-014
   8.4823001646924426E-002 -0.15457587528593181        3.7812655296395187E-014
```

$$\omega_n = \frac{\pi}{\beta}(2n+1) \qquad \mathrm{Im}G(i\omega_n) \qquad \mathrm{Re}G(i\omega_n)$$

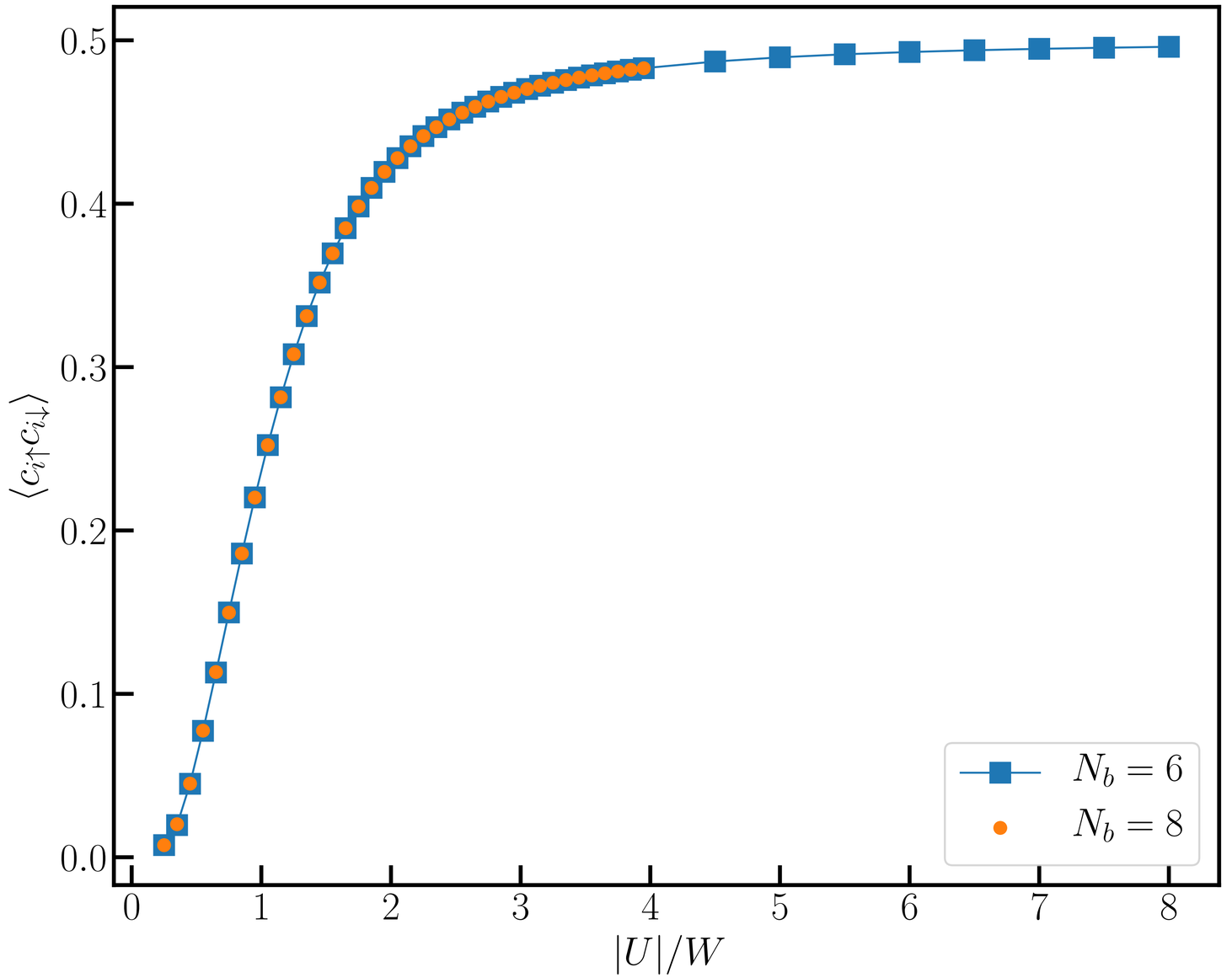
ex.: local energy $\langle H_i \rangle$

```
[giacomo@work-laptop 16:12 uloc-2.00 (master)]$ cat energy_info.ed energy_last.ed
#          1<Hi>  2<V>=<Hi-Ehf>        3<Eloc>         4<Ehf>         5<Dst>         6<Dnd>         7<Dse>         8<Dph>
   -0.381742224   -0.881742224    0.000000000    0.500000000    0.000000000    0.000000000    0.000000000    0.000000000
```

ex.: kinetic energy $K = -t \sum_{ij} \langle c^\dagger_{i\sigma} c_{j\sigma} \rangle$

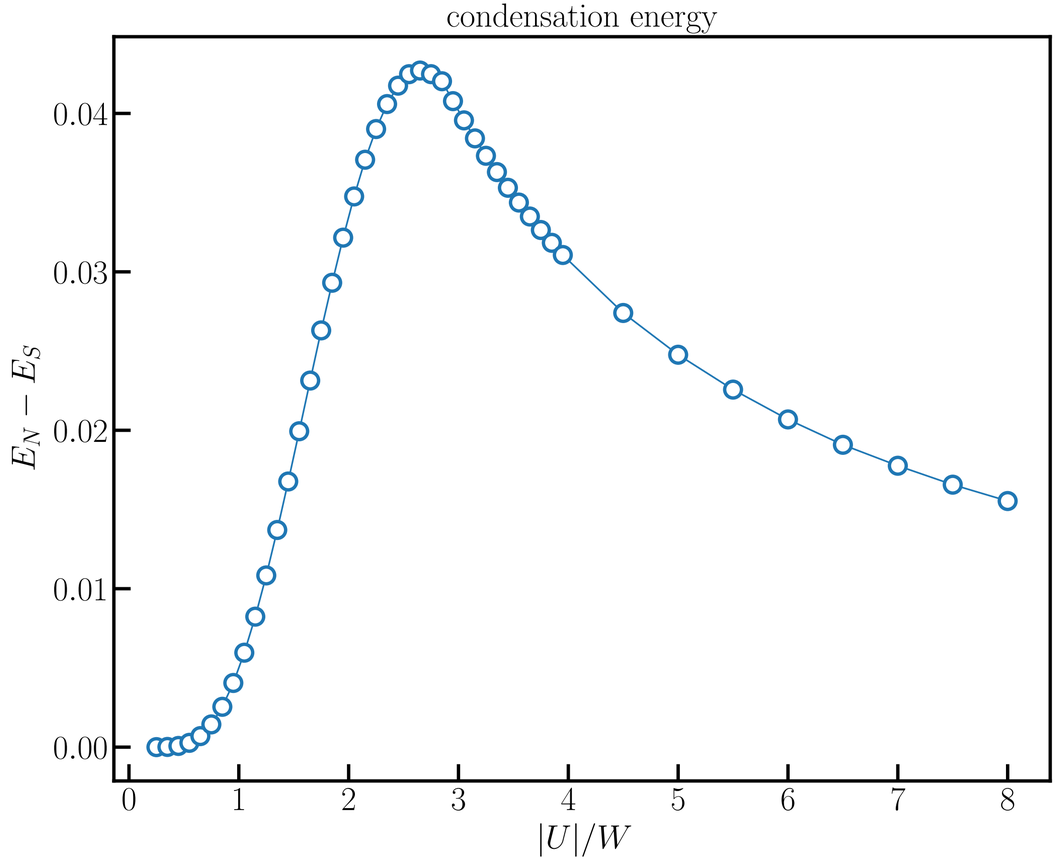
```
[giacomo@work-laptop 16:22 uloc-2.00 (master)]$ cat dmft_kinetic_energy.info dmft_kinetic_energy.dat
#           1<K>         2<Eloc>           3<K1>       4<Eloc1>
   -0.240864613    0.000000000   -0.240864613    0.000000000
```

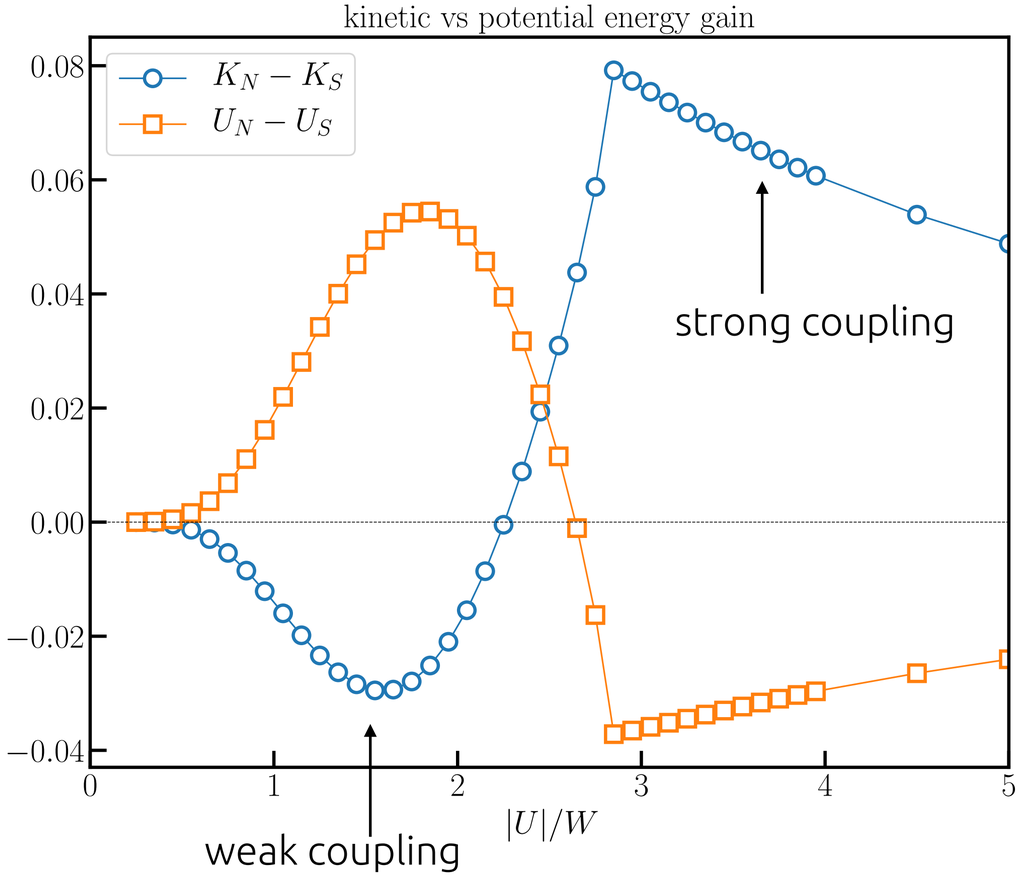
remember!!! total energy = kinetic energy + local energy


0.5
0.4
0.3
0.2
0.1
0.0
$\langle c_{i\uparrow} c_{i\downarrow} \rangle$
0
1
2
3
4
5
6
7
8
$|U|/W$
$N_b = 6$
$N_b = 8$

Make a run with ED_mode=superc and a run with ED_mode=normal to compute the condensation energy

follow the solution at different U and describe the weak to strong coupling crossover





for the braves: try a finite temperature calculation