MULTI-BAND QMC: FORTRAN CODES EXPLAINED.

We start review of the code with parameter file which is used in most subroutines of the program.
"param.dat" looks as :

```
      IMPLICIT REAL*8 (A-H,O-Z)
      parameter(L=16,Iwmax=2**10,Nom=Iwmax+1)
      parameter(Nlm=2,Nd=2*Nlm,Nf=Nlm*(Nd-1),Ns=1)
      complex*16 ci
      parameter (ci=(0.d0,1.d0) )
      parameter (pi=3.1415926535898d0)
      common/global/Beta,Zero,One,Two
```

$Beta$ is $\beta = 1/T$, T is temperature.
$Zero$, One, Two are 0, 1, 2 correspondingly.
Where L is number of time slices, $Iwmax$ is number of ω points.
Nlm is number of orbitals,
Nd is maximum degeneracy (spin degeneracy "2" is taken into account). One can also say that is is maximum number of GFs treated in the system.
$Nf = 0.5 * Nd * (Nd - 1)$ is number of Ising spins.
Ns is real spins number (for AF case $Ns = 2$ and paramagnetic situation is treated with $Ns = 1$).
Meaning of ci, pi is clear.

```
      SUBROUTINE QMC(g0,g,dtau,u,iscf,idum,acc,nrat,nswpIN,nn)
C ----------  ----------  ----------  ----------  ----------  ----------
c     IMPLICIT REAL*8 (A-H,O-Z)
      include 'param.dat'
      include 'mpif.h'
      DOUBLE PRECISION
     &   g0(L,Nd),   g(L,Nd)
     & , ga(L,L,Nd), gx(L,L,Nd), gst(L,L,Nd)
     & , v(L),       vn(L,Nf)
     & , del(L,L)
     & , xg(-L:L),   xgf(-L:L),  G00(-L:L),  xs(0:L)
     & , xf(Nd,Nf)
     & , u(Nf),      xlam(Nf)
     & , docc(Nf),   nn(Nf)
c
      DIMENSION s(2*L,1,Nf) ! it is to save memory space
      INTEGER ipoint(Nf,2)
      INTEGER myid,ierr,numprocs,master
C
      common/ising/ vn
      common/ifs/   if1,iff
      common/varmpi/myid,ierr,numprocs,master
C ----------  ----------  ----------  ----------  ----------  ----------
```

Now we describe input data.
Weiss Green function (GF) $\mathcal{G}_0(\tau)$ is input function into QMC subroutine and $G(\tau)$ is output. In the program they are $g0(L, Nd)$ and $g(L, Nd)$ correspondingly. (L, Nd) means that GSs are defined in τ space (L slices) and for each degeneracy (Nd).
$ga(L, L, Nd), gx(L, L, Nd), gst(L, L, Nd)$ are working GSs they depend on τ and τ' and this is why all of them have (L, L, Nd) structure (time, time, Number_(of)_degeneracy). ga corresponds to G' in formulae above, gx is G and gst is used for storage (gst = Green Function STorage) of temporary GFs on each sweep.
Next line contains two arrays $v(L)$ and $vn(L, Nf)$. The first one is an auxiliary array to form array of Ising fields

$vn(L, Nf)$.
Array $del(L, L)$ is $L \times L$ identity matrix .
$xg(-L : L), xgf(-L : L), G00(-L : L)$ are temporary arrays used in "wrap-around" operation only.
$xf(Nd, Nf)$ contains the sign of the interaction (ie. +=ferro and -=antiferro) given the channel $(1 \rightarrow Nd)$ and the pseudo spin $(1 \rightarrow Nf)$. It is used only in clean update routine.
Matrix of Coulomb interactions is contained in $U(Nf)$ array.
And $xlam(Nf)$ contains information about all exponents used in Hubbard-Stratonovich transformations.
$docc(Nlm)$ is used to deal with double occupancy and it is unimportant for QMC itself.
Finally, $ipoint(Nf, 2)$ is enumeration array. It is used in calculation of determinants ratio.
Next step will be initialization.

```
      if(idum.eq.0) idum=123457
      nsweep=nswpIN/numprocs
      polar=0.5d0
      ndirty=100
      ncor=2
      if(iscf.eq.1) then
        nwarm=100
      else
        nwarm=30
      endif
```

$idum$ is an orbitrary number to initialize random generator.
$nsweep$ is number of sweeps done per processor.
$ndirty$ is number of runs before the clean update which is necessary to check that the precision is not degraded.
$polar$ defines ratio of spin up and down.
$ncor=$?
Next we initialize del, gst and xf arrays .

```
      do in=1,Nf
          docc(in)=0.d0
            nn(in)=0.d0
      enddo
C
      do j=1,L
         do  i=1,L
            del(i,j)=0.d0
            del(i,i)=1.d0
         enddo
      enddo
c
      do kc=1,Nd
       do j=1,L
         do i=1,L
            gst(i,j,kc)=0.d0
         enddo
        enddo
      enddo
c
c initialize: xf
c xf contains in the sign of the interaction (ie. +=ferro
c and -=antiferro) given the channel (1-Nd) and the pseudo spin (1-Nf)
    do i=1,Nd
     do j=1,Nf
      xf(i,j)=0.d0
     enddo
    enddo
```

```
C
c       Sigma(i,ij)  =  1 if i<j
c                    = -1 if i>j
        ij=0
        do i=1,Nd-1
         do j=i+1,Nd
          ij=ij+1
          xf(i,ij)=1.d0
          xf(j,ij)=-1.d0
         enddo
        enddo

c initialization: ipoint(:2)
c first entry (1 -> Nf) is which pseudo spin flips
c and the second entry (1 -> 2) corresponds to
c 1 to ferro-coupling (+), and 2 to antiferro (-)
        ij=0
        do i=1,Nd-1
         do j=i+1,Nd
          ij=ij+1
          ipoint(ij,1)=i
          ipoint(ij,2)=j
         enddo
        enddo
C Structure of ipoint:
C 1 1 1 1 2 2 2 3 3 4
C 2 3 4 5 3 4 5 4 5 5
```

Before main loop we store all values of $\lambda_{mm'}$ in exponents (9).

```
    do i=1,Nf
          z=dtau*u(i)/2.d0
          z=exp(z)
          xlam(i)=dlog(z+dsqrt(z**2-1.d0))
    enddo
```

Next line is random generator initialization and following loops transform physical GF depending on τ into GF depending on τ and τ' plus we change sign of GF (historical reason).

```
c      initialization: random numbers generator
      r=ranw(idum)
c// Minus - change for QMC G-notation //
    do kc=1,Nd
         do i=0,L-1
          g00(i)=-g0(i+1,kc)
         enddo
         do i=1,L-1
          g00(-i)=-g00(L-i)
         enddo
c
     do  ir=1,L
      do  is=1,L
           gx(is,ir,kc)=g00(is-ir)
      enddo
     enddo
    enddo
c
```

Here we generate (initialize) field of auxiliary Ising spins $vn(L, Nf)$ for each time slice.

```
      if(iscf.eq.1) then
    do iv=1,Nf
     xlam0=xlam(iv)
     call initial(xlam0,v,polar,idum)
     do i=1,L
      vn(i,iv)=v(i)
     enddo
    enddo
      endif
```

We do first clean update (full matrix invertion) and produce $G' \equiv ga$.

```
    do kc=1,Nd
          call gnewclean(ga,vn,gx,del,xf,kc)
    enddo
```

Read comments to learn meaning of kx, irr, $nrat$.
Here is very important point: START of the **MAIN** loop of simulations!

```
C ----------  ----------  ----------  ----------  ----------  ----------
c Main loop does nsweeps: a clean update comes after ndirty dirty updates
c
c kx= # of measurements
c irr= # of accepted flips (after the warm-up)
c nrat= # of negative det encountered
C ----------  ----------  ----------  ----------  ----------  ----------
    kx=0
    irr=0
    nrat=0
    kxmax=0
C
        DO 2 k=1,nsweep
         kk=mod(k,ndirty)
         kcor=mod(k,ncor)
         DO 5 j=1,L
         DO 55 iv=1,Nf
```

For a given spin configuration we make a spin flip and calculate the determinant ratio. Formulae (20), (21), (22), (23) are implemented. If determinant ratio (20) is negative we count such cases via $nrat$.

```
           dv=2.d0*vn(j,iv)
c// calculates the determinant ratio //
           kup=ipoint(iv,1)
           kdw=ipoint(iv,2)
           ratup=1.d0+(1.d0-ga(j,j,kup))*(dexp(-dv)-1.d0)
           ratdw=1.d0+(1.d0-ga(j,j,kdw))*(dexp(+dv)-1.d0)
           rat=ratup*ratdw
C
           if(rat.lt.0.)then
              nrat=nrat+1
           end if
```

Now we calculate determinant rat ratio (20) and compare it with a random number r and if $rat < r$ we accept spin flip, count it, and inset change into Ising spin configuration matrix vn.

```
            rat=rat/(1.d0+rat)
c// for num rec
        r=ranw(idum)
            if(rat.gt.r)then
               if(k.gt.nwarm)then
                  irr=irr+1
               end if
```

We store the successful spin flip.

```
c
               vn(j,iv)=-vn(j,iv)
```

Depending on kk value (if it is equal to $ndirty$ ($kk = 0$) then clean update should be done).

```
               if(kk.eq.0)then
              call gnewclean(ga,vn,gx,del,xf,kup)
              call gnewclean(ga,vn,gx,del,xf,kdw)
               else
                  call gnew(ga,vn,j,iv,del,1.d0,kup)
                  call gnew(ga,vn,j,iv,del,-1.d0,kdw)
               endif
            endif
55        CONTINUE
5       CONTINUE
```

Two loops in τ and in Ising spin are finished and we can store our measurement for a sweep. We store values of GF gst and $docc$ – double occupancies.

```
c// store the measurements //
    if(kcor.eq.0.and.k.gt.nwarm)then
    kx=kx+1
    do kc=1,Nd
c// store GF //
      do jx=1,L
            do ix=1,L
              gst(ix,jx,kc)=gst(ix,jx,kc)+ga(ix,jx,kc)
            enddo
      enddo
    enddo
c
C ----------  ----------  ----------  ----------  ----------  ----------
          ij=0
        do ik=1,2*Nlm-1
         do jk=ik+1,2*Nlm
                ij=ij+1
          do ix=1,L
C
           gak  =1.-ga(ix,ix,ik)
           gakN =1.-ga(ix,ix,jk)
C
           docc(ij)=docc(ij)+gak*gakN
C
          enddo
```

```
         enddo
        enddo
C ----------  ----------  ----------  ----------  ----------  ----------
c
    endif
c
2    CONTINUE  ! end of the main QMC loop
```

After all simulations finished we can compute final GF, double occupancy, as well as the acceptance rate. Below we sum up GF and double accupancy.

```
C ----------  ----------  ----------  ----------  ----------  ----------
c// acceptance rate :
      acc=dfloat(irr)/dfloat((nsweep-nwarm)*L*Nf)
c// normalization of GF //
      kxmax=kx*numprocs
      do kc=1,Nd
    do jx=1,L
          do ix=1,L
            gst(ix,jx,kc)=gst(ix,jx,kc)/dfloat(kxmax)
      enddo
    enddo
      enddo
C
         do k=1,Nf
            nn(k)=docc(k)/dfloat(kxmax*L)
         enddo
```

Getting back physical GF from QMC GF: $(\tau, \tau') \to (\tau - \tau')$.

```
c// wrap-around //
    DO kc=1,Nd
c// negative times (-) //
          do j=0,L-1
            xg(-j)=0.
            do i=1,L-j
              xg(-j)=xg(-j)+gst(i,i+j,kc)
            enddo
          enddo
c// positive times (+) //
          do j=1,L-1
            xg(j)=0.
            do i=1,L-j
              xg(j)=xg(j)+gst(i+j,i,kc)
            enddo
          enddo
c// it is enforced the time "antiperiodicity" //
        do i=1,L-1
          xgf(i)=(xg(i)-xg(i-L))/dfloat(L)
          xgf(i-L)=-xgf(i)
        enddo
        xgf(0)=xg(0)/dfloat(L)
        xgf(-L)=-xgf(0)
```

Finally, we must restore the sign of GF!

```
c// change MINUS sign from QMC //
        do i=0,L-1
          g(i+1,kc)=-xgf(i)
    enddo
c
    ENDDO  ! kc=1,Nd
C ----------  ----------  ----------  ----------  ----------  ----------
```

And we are have the output: $g(L, Nd)$!

```
C ----------  ----------  ----------  ----------  ----------  ----------
      RETURN
      END
```

One can easily see correspondence between formulae (24) and (25) and subroutine "gnew" below.
The key line is line with label 1.
SUBROUTINE gnew(g,vn,j,iv,del,xflag,kc) $\longrightarrow$
SUBROUTINE gnew(G', V^m, τ , m', $\hat{I}$, ± 1, m) .
$\pm 1 \equiv \begin{cases} +1 & \rightarrow & \uparrow \\ -1 & \rightarrow & \downarrow \end{cases}$;

```
c **********  **********  **********  **********  **********  **********
c// dirty update of GF //
c
      SUBROUTINE gnew(g,vn,j,iv,del,xflag,kc)
      include 'param.dat'
      double precision g(L,L,Nd),vn(L,Nf),del(L,L),d(L,L)
c
        dv=xflag*2.*vn(j,iv)
        ee=exp(dv)-1.d0
        a=ee/(1.d0+(1.d0-g(j,j,kc))*ee)
        do i2=1,L
          do i1=1,L
1             d(i1,i2)=g(i1,i2,kc)+(g(i1,j,kc)-del(i1,j))*(a*g(j,i2,kc))
          enddo
        enddo
        do i2=1,L
          do i1=1,L
            g(i1,i2,kc)=d(i1,i2)
          enddo
        enddo
      RETURN
      END
```

To understand what is going on in this subroutine one should take a look at formulae (15), (16), (17), (18).
SUBROUTINE gnewclean(g,vn,gx,del,xf,kc) $\longrightarrow$
SUBROUTINE gnewclean(G',V^m,G,$\hat{I}$,$Q_{mm'}$,m)

```
C **********  **********  **********  **********  **********  **********
c// clean update of GF //
c
      SUBROUTINE gnewclean(g,vn,gx,del,xf,kc)
      include 'param.dat'
      double precision
     & , g(L,L,Nd), vn(L,Nf)
     & , b(L,L), binv(L,L), del(L,L)
     & , gx(L,L,Nd), ee(L)
```

```
     & , xf(Nd,Nf)
c
    do  i=1,L
    vv=0.d0
     do iv=1,Nf
      vv=vv+xf(kc,iv)*vn(i,iv)
     enddo
    ee(i)=dexp(vv)-1.d0
    enddo
c
      do j=1,L
         do  i=1,L
            b(i,j)=del(i,j)-ee(j)*(gx(i,j,kc)-del(i,j))
     enddo
      enddo
c
      call inverse(b,binv)
c
      do i1=1,L
         do i2=1,L
            xdum=0.d0
            do i=1,L
```

$G' = A^{-1}G$

```
               xdum=xdum+binv(i1,i)*gx(i,i2,kc)
        enddo
           g(i1,i2,kc)=xdum
          enddo
       enddo
c
      RETURN
      END
```

This subroutine is obvious. It creates a random sequence of L spins.

```
C **********  **********  **********  **********  **********  **********
c// initialize the vector v of Ising fields //
c
      subroutine initial(xlam,v,polar,idum)
      include 'param.dat'
      double precision v(L)
c
      do i=1,L
         s=1.d0
      r=ranw(idum)
         if(r.gt.polar) s=-1.d0
         v(i)=xlam*s
      enddo
      RETURN
      END
C **********  **********  **********  **********  **********  **********
```