





```

call SiteBasis()

!CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
! EIGENVALUES and EIGENSTATES of the XXZ model
!CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
allocate(EigI(dimTotal))
allocate(VecI(dimTotal,dimTotal))
allocate(work(7*dimTotal))
call HamiltonianXXZ()
CALL DSYEV('V','U',dimTotal,VecI,dimTotal,EigI,WORK,7*dimTotal,INFO)

!CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
!c AVERAGE IPR
!CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
aveIPR=0.0d0
DO i=1,dimTotal
  aux=0.0d0
  DO j=1,dimTotal
    aux=aux+VecI(j,i)**4
  Enddo
  IPR=1.0d0/aux
  aveIPR=aveIPR+IPR
ENDDO
aveIPR=aveIPR/dfloat(dimTotal)

! Output (create file for a loop in Jz)
write(*,*)
write(*,*) 'IPR averaged over all eigenstates'
write(*,*) aveIPR

!CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
! HISTOGRAM with fixed bin size
!CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
Emin = dfloat( int(EigI(1))-1 )
Emax = dfloat( int(EigI(dimTotal)+1) )
NofBINS = int((Emax-Emin)/bin)
NofBINSPLUS=NofBINS+1

allocate(Edge(NofBINSPLUS))
DO i=1,NofBINS+1
  Edge(i)=Emin + bin*dfloat(i-1)
Enddo

allocate(NinBin(NofBINS))

```

```

Do i=1,NofBINS
  NinBin(i) = 0
Enddo

Do i=1,dimTotal
  do k=1,NofBINS
    If(EigI(i) >= Edge(k) .AND. EigI(i) < Edge(k+1)) then
      NinBin(k) = NinBin(k)+1
    Endif
  enddo
Enddo
! Output data for Histogram
  write(30,130) Edge(1),0
Do k=1,NofBINS
  write(30,130) Edge(k),NinBin(k)
  write(30,130) Edge(k+1),NinBin(k)
Enddo
130  Format (E18.9,I6)

```

```

!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
!c END END END END END END END END END END END END END END
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
!      STOP
      END

```

```

!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
!cccccccccccccccccccc SUBROUTINES SUBROUTINES SUBROUTINES cccccccccccc
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

```

```

!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
! ***** DIMENSION OF SUBSPACE *****
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
  subroutine SectorDimension()
  use variables
  implicit none

  facL=1

```

```

Do i=chain-upspins+1,chain
  facL=facL*i
Enddo

facU=1
Do i=2,upspins
  facU=facU*i
Enddo

!c END of SUBROUTINE for DIMENSION OF SUBSPACE
return
end subroutine SectorDimension

!CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
! ***** WRITING THE SITE-BASIS *****
!CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
subroutine SiteBasis()
use variables
implicit none

INTEGER (kind=4) :: ib,jb

logical mtc
integer (kind=4) :: in(chain),m2,h

mtc=.false.

!c INITIALIZATION
Do ib=1,dimTotal
  Do jb=1,chain
    basis(ib,jb)=0
  enddo
enddo

ii=1

71 call NEXKSB(chain,upspins,in,mtc,m2,h)
Do jb=1,upspins
  basis(ii,in(jb))=1

```

```

        Enddo
        ii=ii+1
        if(mtc) goto 71

!c END of SUBROUTINE for SITE-BASIS
        return
        end subroutine SiteBasis
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccc
!c SUBROUTINE to get the COMBINATIONS
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccc
        subroutine NEXKSB(n,k,a,mtc,m2,h)
        integer (kind=4) :: n,k,a(n),m2,h,jn
        logical mtc

        if(.not.mtc) then
        m2=0
        h=k
        go to 50
        endif
        if(m2.lt.n-h) h=0
        h=h+1
        m2=a(k+1-h)
50 do jn=1,h
        a(k+jn-h)=m2+jn
        enddo
        mtc=a(1).ne.n-k+1

        return
        end subroutine nexksb
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccc
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccc

!cccccccccccccccccccccccccccccccccccccccccccccccccccccccc
!
! SUBROUTINE to write the XXZ HAMILTONIAN in the SITE-BASIS
!
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccc

        subroutine HamiltonianXXZ()

        use variables
        implicit none

```

```

INTEGER (kind=4) :: tot,DifferentSite(chain)

!c INITIALIZATION
  Do i=1,dimTotal
    Do j=1,dimTotal
      VecI(i,j)=0.0d0
    Enddo
  Enddo

! DIAGONAL ELEMENTS
*****
  Do i=1,dimTotal
!cccccccccccccccc NN ccccccccccccccccccccccccccccccccc
    Do j=1,chain-1
      VecI(i,i)=VecI(i,i)+(JzI/4.d0)*(-1.0d0)**(basis(i,j)+basis(i,j+1))
    enddo
!cccccccccccccccccccccccccccccccccccccccccccccccccccccccc
  ! CLOSING i=1,dimTotal
    enddo
! END of DIAGONAL
*****

! OFF-DIAGONAL ELEMENTS
*****
  Do i = 1, dimTotal-1
    Do j = i+1, dimTotal

      tot = 0
      Do k = 1, chain
        DifferentSite(k)=0
      Enddo

      Do k = 1, chain
        If( basis(i,k).ne.basis(j,k) ) then
          tot = tot + 1
          DifferentSite(tot)=k
        Endif
      Enddo

      IF(tot.EQ.2) then
!cccccccccccccccc NN ccccccccccccccccccccccccccccccccc
      IF((DifferentSite(2) - DifferentSite(1)).EQ.1) then
        VecI(i,j)=VecI(i,j)+JxyI/2.0d0
        VecI(j,i)=VecI(i,j)
      
```

