magnetochemistry

MDPI

Supplementary Materials

Mapping the magnetic anisotropy inside a Ni_4 Cubane Spin Cluster using polarized neutron diffraction

Olga Iasco, Yuri Chumakov, Frédéric Guégan, Béatrice Gillon*, Marc Lenertz, Alexandre Bataille, Jean-François Jacquot and Dominique Luneau

* Correspondence: beatrice.gillon@cea.fr; Tel.: +33-016-908-2791

Table S1. Components of the eigenvectors of the susceptibility tensor from PND for the cluster, individual ions and antiferromagnetic pairs in the $(\vec{i},\vec{j},\vec{k})$ cartesian basis set.

			x_1	x_2	x_3
cluster	[Ni_4]	$\chi 1$	-0.592	0.806	0
		$\chi 2$	0.806	0.592	0
		$\chi 3$	0	0	1
ions	Ni1	$\chi 1$	0	1	0
		$\chi 2$	1	0	0
		$\chi 3$	0	0	1
	Ni2	$\chi 1$	0	1	0
		$\chi 2$	1	0	0
		$\chi 3$	0	0	1
	Ni3	$\chi 1$	-0.390	0.921	0
		$\chi 2$	0	0	1
		$\chi 3$	0.921	0.390	0
	Ni4	$\chi 1$	-0.646	0.764	0
		$\chi 2$	0.764	0.646	0
		$\chi 3$	0	0	1
pairs	Ni1-Ni4	$\chi 1$	-0.638	0.770	0
		$\chi 2$	0.770	0.638	0
		$\chi 3$	0	0	1
	Ni2-Ni3	$\chi 1$	0	1	0
		$\chi 2$	0	0	1
		$\chi 3$	1	0	0

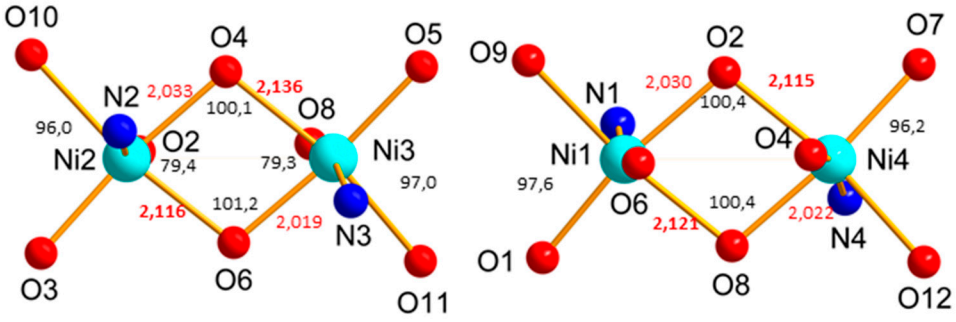

Figure S1. Geometry of the two antiferromagnetic NiO_2Ni bridges. The distance between the Ni ions is equal to 3.19 Å and bridging angle close to 100 deg.

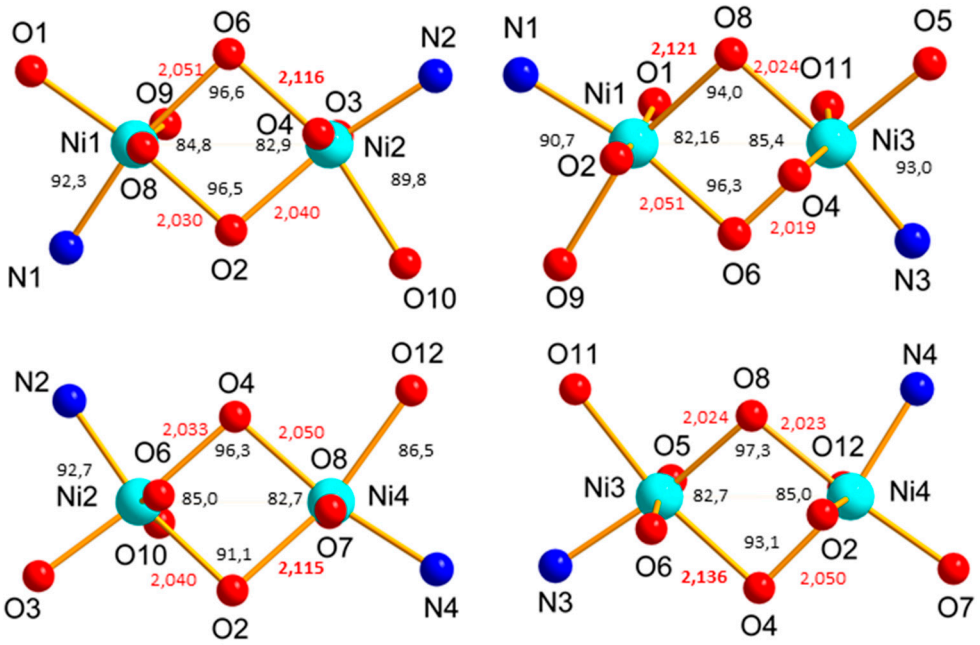

Figure S2. Geometry of the four ferromagnetic NiO_2Ni bridges.

- Program for the calculation of single crystal magnetisation from PND or ab initio data

```
!*******************************************************************************
!                                    CalcM
!                                Frédéric Guégan
!                                  march 2016
! Program for the calculation of single crystal magnetisation from PND or ab initio data
!
! Frédéric GUEGAN, Université de Lyon
! frederic.guegan@univ-lyon1.fr
!
! Please refer to the manual (PDF) for any detail on the calculations.
!
```

```
!*******************************************************************************
PROGRAM CalcM
implicit none

    character(LEN=80)         :: choice,input,orient,output,filename,sortie1,sortie2,sym
    real(8),  parameter :: pi=3.14159265358979d0
    character(len=80)         :: formdat
    integer                   :: N,M,P
    real(8), dimension(:,:),allocatable    :: fields
    real(8), dimension(3,3)            :: chi,Mccslcri,Mmolcal,Mcrimol
    real(8), dimension(3,3)            :: Mcriccsl,Mcalmol, Mmolcri
    real(8), dimension(:,:),allocatable    :: moments,symop
    real(8),dimension(:),allocatable   :: meanmom
    real(8), dimension(3)          :: fieldi,momveci,fieldcry,mat1,mat2
    real(8)                   :: momi
    real(8)                   :: time1,time2,time3,tottime
    integer                   :: i,j
    CHARACTER(LEN=80) :: at0cry,at1cry,at2cry,at3cry,at0cal,at1cal,at2cal,at3cal ! atoms
labels in the .inp file
    real(8), dimension(3)     :: Ybcry, X1cry, X2cry, X3cry ! central and other atoms
positions in the lattice cell
    real(8), dimension(3)     :: Ybcal, X1cal, X2cal, X3cal ! same, in the calculation
frame
    real(8), dimension(3,3)   :: chical ! susceptibility tensor
    real(8), dimension(3) :: v1ccsl,v2ccsl,v3ccsl ! CCSL basis vectors

write(6,*)
'*******************************************************************************'
write(6,*) '                                          CalcM
'
write(6,*) '                                    Frédéric Guégan
'
write(6,*) '                                      march 2016
'
write(6,*) ' '
write(6,*) ' Program for the calculation of single crystal magnetisation from PND or ab
'
write(6,*) ' initio data '
write(6,*) ' '
write(6,*)
'*******************************************************************************'

write(6,*) ' '
write(6,*) ' Which kind of data would you like to use : PND(p) or Ab Initio(a) ?'
read(5,*) choice
write(6,*) ' Nom du fichier input (sans extension .inp) ? '
read(5,*) input
write(6,*) ' Nom du fichier orientations (sans extension .dat) ? '
read(5,*) orient
write(6,*) ' Nom du fichier output (sans extension .out) ? '
read(5,*) output
write(6,*) ' Name of the symmetry file (without extension .sym) ? '
read(5,*) sym

CALL CPU_TIME(time1)
! Now we read the symmetry file
    filename = TRIM(sym)//'.sym'
    open(unit=10,file=TRIM(filename),action='read')
    read(unit=10,fmt=*) M
```

```
     allocate(symop(M,9)) ! we set the size of the array containing the symmetry operations
     do i=1,M
          read(unit=10,fmt=*)
symop(i,1),symop(i,2),symop(i,3),symop(i,4),symop(i,5),symop(i,6),symop(i,7),symop(i,8),symo
p(i,9)
     end do ! we read the symmetry matrices
     close(10)
! We get the orientation data
     filename = TRIM(orient)//'.dat'
     OPEN(unit=12,file=TRIM(filename),action='read')
     read(unit=12,fmt=*) N
     allocate(fields(N,4)) ! we define the number of orientations to consider
     P=N*M
     allocate(moments(P,3)) ! and the total number of computed values
     allocate(meanmom(N)) ! and the number of mean magnetisation (for the whole symmetries)

     do i=1,N
          read(unit=12,fmt=*) fields(i,1),fields(i,2),fields(i,3),fields(i,4)
     end do
! Here we read the orientations of fields and stocked them in the array 'fields'
     close(12)

! Now we get the information contained within the input file.
     filename = TRIM(input)//'.inp'
     OPEN(unit=11,file=TRIM(filename),action='read')
     read(unit=11,fmt=*) at0cry,Ybcry(1),Ybcry(2),Ybcry(3)
     read(unit=11,fmt=*) at1cry,X1cry(1),X1cry(2),X1cry(3)
     read(unit=11,fmt=*) at2cry,X2cry(1),X2cry(2),X2cry(3)
     read(unit=11,fmt=*) at3cry,X3cry(1),X3cry(2),X3cry(3) ! Atoms positions in the lattice

     read(unit=11,fmt=*) at0cal,Ybcal(1),Ybcal(2),Ybcal(3)
     read(unit=11,fmt=*) at1cal,X1cal(1),X1cal(2),X1cal(3)
     read(unit=11,fmt=*) at2cal,X2cal(1),X2cal(2),X2cal(3)
     read(unit=11,fmt=*) at3cal,X3cal(1),X3cal(2),X3cal(3) ! Same in the calculation frame

     read(unit=11,fmt=*)
chical(1,1),chical(1,2),chical(1,3),chical(2,1),chical(2,2),chical(2,3),chical(3,1),chical(3
,2),chical(3,3)
     ! susceptibility tensor

     read(unit=11,fmt=*) v1ccsl(1),v1ccsl(2),v1ccsl(3)
     read(unit=11,fmt=*) v2ccsl(1),v2ccsl(2),v2ccsl(3)
     read(unit=11,fmt=*) v3ccsl(1),v3ccsl(2),v3ccsl(3) ! CCSL vectors in the lattice cell
     CLOSE(11)


! We prepare the matrices

     do i=1,3
          Mcriccsl(i,1)=v1ccsl(i)
          Mcriccsl(i,2)=v2ccsl(i)
          Mcriccsl(i,3)=v3ccsl(i)
     end do ! Mcriccsl is OK

     do i=1,3
          Mcrimol(i,1)=X1cry(i)-Ybcry(i)
          Mcrimol(i,2)=X2cry(i)-Ybcry(i)
          Mcrimol(i,3)=X3cry(i)-Ybcry(i)
     end do ! Mcrimol is OK
```

```
    do i=1,3
        Mcalmol(i,1)=X1cal(i)-Ybcal(i)
        Mcalmol(i,2)=X2cal(i)-Ybcal(i)
        Mcalmol(i,3)=X3cal(i)-Ybcal(i)
    end do ! Mcalmol is OK

    call inverse(Mcalmol,Mmolcal,3)
    call inverse(Mcrimol,Mmolcri,3)
    call inverse(Mcriccsl,Mccslcri,3)

! and we inversed the three previous matrices
! Nota: they may not be all used, but we may assume their computation is not a tremendously
demanding task

    if(choice.EQ.'p') then
        call PND(P,N,M,fields,chical,Mcriccsl,moments,symop,meanmom)
    else if(choice.EQ.'a') then
        call Abinit(P,N,M,fields,chical,Mccslcri,Mcrimol,Mmolcal,moments,symop,meanmom)
    else
    write(6,*) 'Unrecognised data label. Please check.'
    end if

sortie1 = TRIM(output)//'1.out'
OPEN(unit=13,file=TRIM(sortie1),action='write',status='new')
write(13,*)
'********************************************************************************'
write(13,*) '                                        CalcM
'
write(13,*) '                                   Frédéric Guégan
'
write(13,*) '                                     march 2016
'
write(13,*) ' '
write(13,*) ' Program for the calculation of single crystal magnetisation from PND or ab
'
write(13,*) ' initio data '
write(13,*) ' '
write(13,*)
'********************************************************************************'
write(13,*) ' Implemented by :'
write(13,*) ' '
write(13,*) ' Frédéric Guégan, University of Lyon 1 '
write(13,*) ' frederic.guegan@univ-lyon1.fr'
write(13,*) ' '
write(13,*) '  Team Crystallography and Molecular Engineering (Prof. D. Luneau) '
write(13,*) '  Multimaterials and Interfaces Laboratory (LMI, UMR CNRS-UCBL 5615)'
write(13,*) ' '
write(13,*) '  Team Chemometrics and Theoretical Chemistry (Prof. C. Morell) '
write(13,*) '  Lyon Institute for Analytical Sciences (ISA, UMR CNRS-UCBL-ENS 5280)'
write(13,*) ' '
write(13,*)
'********************************************************************************'
write(13,*) ' This program is freely distributed and modifiable by anyone. I do not
guarantee '
write(13,*) ' the exactitude of the calculations (though I did my best). Please feel free to
'
write(13,*) ' contact me if you find any error in the code !'
```

```
write(13,*)
'*******************************************************************************'
write(13,*) ' '
write(13,*) ' '
write(13,*) 'Type of data: '
    if(choice.EQ.'p') then
        write(13,*) '    PND experiment '
    else if(choice.EQ.'a') then
        write(13,*) '    Ab initio data '
    end if
write(13,*) ' '

write(13,*) 'Printing the matrices used for the transformation:'

write(13,*) 'Mcalmol :'
    do i=1,3
        write(13,*) Mcalmol(i,1),Mcalmol(i,2),Mcalmol(i,3)
    end do
write(13,*) ' '

write(13,*) 'Mmolcal :'
    do i=1,3
        write(13,*) Mmolcal(i,1),Mmolcal(i,2),Mmolcal(i,3)
    end do
write(13,*) ' '

write(13,*) 'Mcriccsl :'
    do i=1,3
        write(13,*) Mcriccsl(i,1),Mcriccsl(i,2),Mcriccsl(i,3)
    end do
write(13,*) ' '

write(13,*) 'Mccslcri :'
    do i=1,3
        write(13,*) Mccslcri(i,1),Mccslcri(i,2),Mccslcri(i,3)
    end do
write(13,*) ' '

write(13,*) 'Mcrimol :'
    do i=1,3
        write(13,*) Mcrimol(i,1),Mcrimol(i,2),Mcrimol(i,3)
    end do
write(13,*) ' '

write(13,*) 'Mmolcri'
    do i=1,3
        write(13,*) Mmolcri(i,1),Mmolcri(i,2),Mmolcri(i,3)
    end do
write(13,*) ' '
write(13,*) ' '
write(13,*) ' Results for all symmetries'
write(13,*) ' '

write(13,*) ' Theta                    Mcalc                       H norm'
    do i=1,P
        write(13,*) ' ',moments(i,1),' ',moments(i,2),' ',moments(i,3)
    end do

call CPU_TIME(time2)
```

```
tottime=time2-time1

write(13,*) ' '
write(13,*) 'Total time of execution (s):',tottime
    close(13)

sortie2 = TRIM(output)//'2.out'
OPEN(unit=14,file=TRIM(sortie2),action='write',status='new')
write(14,*)
'********************************************************************************'
write(14,*) '                                        CalcM
'
write(14,*) '                                    Frédéric Guégan
'
write(14,*) '                                      march 2016
'
write(14,*) ' '
write(14,*) ' Program for the calculation of single crystal magnetisation from PND or ab
'
write(14,*) ' initio data '
write(14,*) ' '
write(14,*)
'********************************************************************************'
write(14,*) ' Implemented by :'
write(14,*) ' '
write(14,*) ' Frédéric Guégan, University of Lyon 1 '
write(14,*) ' frederic.guegan@univ-lyon1.fr'
write(14,*) ' '
write(14,*) '  Team Crystallography and Molecular Engineering (Prof. D. Luneau) '
write(14,*) '  Multimaterials and Interfaces Laboratory (LMI, UMR CNRS-UCBL 5615)'
write(14,*) ' '
write(14,*) '  Team Chemometrics and Theoretical Chemistry (Prof. C. Morell) '
write(14,*) '  Lyon Institute for Analytical Sciences (ISA, UMR CNRS-UCBL-ENS 5280)'
write(14,*) ' '
write(14,*)
'********************************************************************************'
write(14,*) ' This program is freely distributed and modifiable by anyone. I do not
guarantee '
write(14,*) ' the exactitude of the calculations (though I did my best). Please feel free to
'
write(14,*) ' contact me if you find any error in the code !'
write(14,*)
'********************************************************************************'
write(14,*) ' '
write(14,*) ' '
write(14,*) 'Type of data: '
    if(choice.EQ.'p') then
        write(14,*) '    PND experiment '
    else if(choice.EQ.'a') then
        write(14,*) '    Ab initio data '
    end if
write(14,*) ' '

write(14,*) 'Printing the matrices used for the transformation:'

write(14,*) 'Mcalmol :'
    do i=1,3
        write(14,*) Mcalmol(i,1),Mcalmol(i,2),Mcalmol(i,3)
```

```
    end do
write(14,*) ' '

write(14,*) 'Mmolcal :'
    do i=1,3
        write(14,*) Mmolcal(i,1),Mmolcal(i,2),Mmolcal(i,3)
    end do
write(14,*) ' '

write(14,*) 'Mcriccsl :'
    do i=1,3
        write(14,*) Mcriccsl(i,1),Mcriccsl(i,2),Mcriccsl(i,3)
    end do
write(14,*) ' '

write(14,*) 'Mccslcri :'
    do i=1,3
        write(14,*) Mccslcri(i,1),Mccslcri(i,2),Mccslcri(i,3)
    end do
write(14,*) ' '

write(14,*) 'Mcrimol :'
    do i=1,3
        write(14,*) Mcrimol(i,1),Mcrimol(i,2),Mcrimol(i,3)
    end do
write(14,*) ' '

write(14,*) 'Mmolcri'
    do i=1,3
        write(14,*) Mmolcri(i,1),Mmolcri(i,2),Mmolcri(i,3)
    end do
write(14,*) ' '
write(14,*) ' '
write(14,*) ' Results for all symmetries'
write(14,*) ' '

write(14,*) ' Theta                    Mcalc                        H norm'
    do i=1,N
        write(14,*) ' ',moments(i,1),' ',meanmom(i),' ',moments(i,3)
    end do

call CPU_TIME(time3)

tottime=time3-time1

write(14,*) ' '
write(14,*) 'Total time of execution (s):',tottime
    close(14)

end program CalcM

subroutine PND(P,N,M,fields,chi,Mcriccsl,moments,symop,meanmom)
! Subroutine for the calculation of the magnetisation induced by a field applied along a
given direction, from the local susceptibility tensor (deduced from PND experiments) and in
the linear approximation : m = XH.
! Arguments: N, number of field orientations ; fields, array of the field orientations; chi,
susceptibility tensor ; Mcriccsl, matrix of the CCSL vectors in the lattice basis.
! Output of subroutine : array of calculated moments versus theta, and norm of the magnetic
field (control, should be 1).
```

```
implicit none

     character(len=80)          :: formdat,sym
     integer                    :: N,M,P
     real(8), dimension(N,4)            :: fields
     real(8), dimension(3,3)            :: chi,Mcriccsl,Opsym,chisym,tOpsym,tmp
     real(8), dimension(P,3)            :: moments
     real(8), dimension(N)          :: meanmom
     real(8), dimension(M,9)            :: symop
     real(8), dimension(3)          :: fieldi,momveci,fieldcry
     real(8)                    :: momi
     integer                    :: i,j,k,l,t


write(6,*)
'*******************************************************************************'
write(6,*) '                                              PND
'
write(6,*) '                                       Frédéric Guégan
'
write(6,*) '                                         march 2016
'
write(6,*) ' '
write(6,*) ' Evaluation of the single crystal magnetisation from PND experiments '
write(6,*) ' '
write(6,*)
'*******************************************************************************'

write(6,*) ' Format of the orient.dat file (CCSL:yes, crystal:no)'
read(5,*) formdat
     if(formdat.EQ.'yes') then
     do i=1,N
          meanmom(i)=0
     end do
     do j=1,M
          do i=1,N
               fieldi(1)=fields(i,2)
               fieldi(2)=fields(i,3)
               fieldi(3)=fields(i,4) ! we first assign to fieldi the direction of the field
at point i
                    do k=1,3
                         l=k+3
                         t=k+6
                         Opsym(1,k) = symop(j,k)
                         Opsym(2,k) = symop(j,l)
                         Opsym(3,k) = symop(j,t) ! we built the matrix for each symm operation
(loop in j)
                    end do
!                   do l=1,3
!                        do t=1,3
!                             tOpsym(l,t)=Opsym(t,l) ! we transpose the symmetry matrix
!                        end do
!                   end do
                  call inverse(Opsym,tOpsym,3)
                    write(6,*) tOpsym(1,1),tOpsym(1,2),tOpsym(1,3)
                    write(6,*) tOpsym(2,1),tOpsym(2,2),tOpsym(2,3)
                    write(6,*) tOpsym(3,1),tOpsym(3,2),tOpsym(3,3)
                    tmp=MATMUl(chi,Opsym) ! we begin the symmetry transformation of chi
                    chisym=MATMUL(tOpsym,tmp)
```

```
                momveci=MATMUL(chisym,fieldi)
                momi=momveci(1)*fieldi(1)+momveci(2)*fieldi(2)+momveci(3)*fieldi(3) !
scalar product of vec(m) and vec(H)
                k=i+(j-1)*N
                moments(k,1)=fields(i,1)
                moments(k,2)=momi

    moments(k,3)=sqrt(fieldi(1)*fieldi(1)+fieldi(2)*fieldi(2)+fieldi(3)*fieldi(3))
                meanmom(i)=meanmom(i)+momi/(M*1.0)
        end do
    end do
    else if(formdat.EQ.'no') then
        do j=1,M
            do i=1,N
                fieldcry(1)=fields(i,2)
                fieldcry(2)=fields(i,3)
                fieldcry(3)=fields(i,4)

                fieldi=MATMUL(Mcriccsl,fieldcry)

                do k=1,3
                    l=k+3
                    m=k+6
                    Opsym(1,k) = symop(j,k)
                    Opsym(2,k) = symop(j,l)
                    Opsym(3,k) = symop(j,m)
                end do
                do l=1,3
                    do m=1,3
                        tOpsym(l,m)=Opsym(m,l) ! we transpose the symmetry matrix
                    end do
                end do
                tmp=MATMUl(chi,Opsym) ! we begin the symmetry transformation of chi
                chisym=MATMUL(tOpsym,tmp)

                momveci=MATMUL(chisym,fieldi)
                momi=momveci(1)*fieldi(1)+momveci(2)*fieldi(2)+momveci(3)*fieldi(3) !
scalar product of vec(m) and vec(H)
                k=i+(j-1)*N
                moments(k,1)=fields(i,1)
                moments(k,2)=momi

    moments(k,3)=sqrt(fieldi(1)*fieldi(1)+fieldi(2)*fieldi(2)+fieldi(3)*fieldi(3))
                meanmom(i)=meanmom(i)+momi/(M*1.0)
            end do
        end do
    else
        write(6,*) 'Unrecognised format identifier, please check.'
    end if

end subroutine PND

subroutine Abinit(P,N,M,fields,chi,Mmolcri,Mcalmol,Mcriccsl,moments,symop,meanmom)
! Subroutine for the calculation of the magnetisation induced by a field applied along a
given direction, from ab initio calculated susceptibility tensor and in the linear
approximation : m = XH.
! Arguments: N, number of field orientations ; fields, array of the field orientations; chi,
susceptibility tensor ; Mcricssl, matrix of the CCSL vectors in the lattice basis; Mmolcri,
```

```
inverse of the matrix of three vectors (based on 4 non collinear atoms) in the lattice cell
; Mcalmol, inverse of the same matrix but in the basis of the calculation (orthonormal
frame).
! Output of subroutine : array of calculated moments versus theta, and norm of the magnetic
field (control, should be 1).

implicit none
     character(len=80)         :: formdat
     integer                   :: N,P,M
     real(8), dimension(N,4)            :: fields
     real(8), dimension(3,3)            ::
chi,Mcriccsl,tMcriccsl,Mcalmol,Mmolcri,mat1,mat2,mat3,mat4
     real(8), dimension(3,3)            :: chiccsl,Opsym,tOpsym,chisym,tmp
     real(8), dimension(p,3)            :: moments
     real(8), dimension(N)          :: meanmom
     real(8), dimension(M,9)            :: symop
     real(8), dimension(3)          :: fieldi,momveci,fieldcry
     real(8)                   :: momi,norm
     integer                   :: i,j,l,t,k

write(6,*)
'*******************************************************************************'
write(6,*) '                                        Abinit
'
write(6,*) '                                    Frédéric Guégan
'
write(6,*) '                                      march 2016
'
write(6,*) ' '
write(6,*) ' Evaluation of the single crystal magnetisation from Ab initio calculations '
write(6,*) ' '
write(6,*)
'*******************************************************************************'

write(6,*) ' Format of the orient.dat file (CCSL:yes, crystal:no)'
read(5,*) formdat
     if(formdat.EQ.'yes') then
     do i=1,N
          meanmom(i)=0
     end do
          mat1=MATMUL(Mcalmol,Mmolcri) ! matrix from the calculated frame to the crystal one
               do i=1,3
                    do j=1,3
                         mat2(i,j)=mat1(j,i)
                    end do
               end do ! mat2 = t(mat1)
               do i=1,3
                    do j=1,3
                         tMcriccsl(i,j)=Mcriccsl(j,i)
                    end do
               end do ! we transpose Mccslcri
          mat3=MATMUL(mat2,chi)
          mat4=MATMUL(mat3,mat1)
          mat1=MATMUL(mat4,Mcriccsl)
          chiccsl=MATMUL(tMcriccsl,mat1)
          norm=chiccsl(1,1)*(chiccsl(2,2)*chiccsl(3,3)-chiccsl(3,2)*chiccsl(2,3))-&
               chiccsl(1,2)*(chiccsl(2,1)*chiccsl(3,3)-chiccsl(3,1)*chiccsl(2,3))+&
               chiccsl(1,3)*(chiccsl(2,1)*chiccsl(3,2)-chiccsl(3,1)*chiccsl(2,2))
          write(6,*) norm
```

```
         do j=1,M
              do i=1,N
                   fieldi(1)=fields(i,2)
                   fieldi(2)=fields(i,3)
                   fieldi(3)=fields(i,4) ! we first assign to fieldi the direction of the
field at point i

                   do k=1,3
                        l=k+3
                        t=k+6
                        Opsym(1,k) = symop(j,k)
                        Opsym(2,k) = symop(j,l)
                        Opsym(3,k) = symop(j,t) ! we built the matrix for each symm operation
(loop in j)
                   end do
                   do l=1,3
                        do t=1,3
                             tOpsym(l,t)=Opsym(t,l) ! we transpose the symmetry matrix
                        end do
                   end do
                   tmp=MATMUl(chiccsl,Opsym) ! we begin the symmetry transformation of chi
                   chisym=MATMUL(tOpsym,tmp)

                   momveci=MATMUL(chisym,fieldi) ! evaluation of the magnetisation in the
CCSL lattice
                   momi=momveci(1)*fieldi(1)+momveci(2)*fieldi(2)+momveci(3)*fieldi(3) !
scalar product of vec(m) and vec(H) : magnetisation along the field direction
                   k=i+(j-1)*N
                   moments(k,1)=fields(i,1)
                   moments(k,2)=momi

     moments(k,3)=sqrt(fieldi(1)*fieldi(1)+fieldi(2)*fieldi(2)+fieldi(3)*fieldi(3))
                   meanmom(i)=meanmom(i)+momi/(M*1.0)
              end do
         end do
     else if(formdat.EQ.'no') then
     do i=1,N ! we do the same thing as previously, but we also change the frame for the
field (cry -> CCSL)
         meanmom(i)=0
     end do
         mat1=MATMUL(Mcalmol,Mmolcri) ! matrix from the calculated frame to the crystal one
              do i=1,3
                   do j=1,3
                        mat2(i,j)=mat1(j,i)
                   end do
              end do ! mat2 = t(mat1)
              do i=1,3
                   do j=1,3
                        tMcriccsl(i,j)=Mcriccsl(j,i)
                   end do
              end do ! we transpose Mccslcri
         mat3=MATMUL(mat2,chi)
         mat4=MATMUL(mat3,mat1)
         mat1=MATMUL(mat4,Mcriccsl)
         chiccsl=MATMUL(tMcriccsl,mat1)
         norm=chiccsl(1,1)*(chiccsl(2,2)*chiccsl(3,3)-chiccsl(3,2)*chiccsl(2,3))-&
              chiccsl(1,2)*(chiccsl(2,1)*chiccsl(3,3)-chiccsl(3,1)*chiccsl(2,3))+&
              chiccsl(1,3)*(chiccsl(2,1)*chiccsl(3,2)-chiccsl(3,1)*chiccsl(2,2))
```

```
         write(6,*) norm

         do j=1,M
             do i=1,N
                  fieldcry(1)=fields(i,2)
                  fieldcry(2)=fields(i,3)
                  fieldcry(3)=fields(i,4) ! we first assign to fieldi the direction of the
field at point i
                  fieldi=MATMUL(Mcriccsl,fieldcry)
                  do k=1,3
                       l=k+3
                       m=k+6
                       Opsym(1,k) = symop(j,k)
                       Opsym(2,k) = symop(j,l)
                       Opsym(3,k) = symop(j,m)
                  end do
                  do l=1,3
                       do m=1,3
                            tOpsym(l,m)=Opsym(m,l) ! we transpose the symmetry matrix
                       end do
                  end do
                  tmp=MATMUl(chiccsl,Opsym) ! we begin the symmetry transformation of chi
                  chisym=MATMUL(tOpsym,tmp)
                  momveci=MATMUL(chisym,fieldi) ! evaluation of the magnetisation in the
CCSL lattice
                  momi=momveci(1)*fieldi(1)+momveci(2)*fieldi(2)+momveci(3)*fieldi(3) !
scalar product of vec(m) and vec(H) : magnetisation along the field directio
                  k=i+(j-1)*N
                  moments(k,1)=fields(i,1)
                  moments(k,2)=momi

     moments(k,3)=sqrt(fieldi(1)*fieldi(1)+fieldi(2)*fieldi(2)+fieldi(3)*fieldi(3))
                  meanmom(i)=meanmom(i)+momi/(M*1.0)
             end do
         end do

     else
         write(6,*) 'Unrecognised format identifier, please check.'
     end if

end subroutine Abinit

subroutine inverse(a,c,n)
!============================================================
! Inverse matrix
! Method: Based on Doolittle LU factorization for Ax=b
! Alex G. December 2009
!------------------------------------------------------------
! input ...
! a(n,n) - array of coefficients for matrix A
! n      - dimension
! output ...
! c(n,n) - inverse matrix of A
! comments ...
! the original matrix a(n,n) will be destroyed
! during the calculation
!============================================================
implicit none
integer n
```

```
double precision a(n,n), c(n,n)
double precision L(n,n), U(n,n), b(n), d(n), x(n)
double precision coeff
integer i, j, k

! step 0: initialization for matrices L and U and b
! Fortran 90/95 aloows such operations on matrices
L=0.0
U=0.0
b=0.0

! step 1: forward elimination
do k=1, n-1
   do i=k+1,n
      coeff=a(i,k)/a(k,k)
      L(i,k) = coeff
      do j=k+1,n
         a(i,j) = a(i,j)-coeff*a(k,j)
      end do
   end do
end do

! Step 2: prepare L and U matrices
! L matrix is a matrix of the elimination coefficient
! + the diagonal elements are 1.0
do i=1,n
  L(i,i) = 1.0
end do
! U matrix is the upper triangular part of A
do j=1,n
  do i=1,j
    U(i,j) = a(i,j)
  end do
end do

! Step 3: compute columns of the inverse matrix C
do k=1,n
  b(k)=1.0
  d(1) = b(1)
! Step 3a: Solve Ld=b using the forward substitution
  do i=2,n
    d(i)=b(i)
    do j=1,i-1
      d(i) = d(i) - L(i,j)*d(j)
    end do
  end do
! Step 3b: Solve Ux=d using the back substitution
  x(n)=d(n)/U(n,n)
  do i = n-1,1,-1
    x(i) = d(i)
    do j=n,i+1,-1
      x(i)=x(i)-U(i,j)*x(j)
    end do
    x(i) = x(i)/u(i,i)
  end do
! Step 3c: fill the solutions x(n) into column k of C
  do i=1,n
    c(i,k) = x(i)
  end do
```

```
  b(k)=0.0
end do
end subroutine inverse
```

CalcM

Program for the calculation of the single crystal magnetisation from PND or ab initio data

FRéDéRIC GUéGAN
Université Claude Bernard Lyon 1
frederic.guegan@univ-lyon1.fr

Version: 21/04/2016

About CalcM

CalcM is a program (written in Fortran 90) designed to compute the magnetisation one could get in an oriented single-crystal SQUID measurement from the magnetic susceptibility tensor obtained either in a Polarised Neutron Diffraction experiment (PND) or an *ab initio* calculation (typically from MOLCAS). The program is free and modifiable by anyone. Despite all my efforts, some mistakes might still be present in the code. Feel free to contact me if you find any !

ADDITION TO THE PREVIOUS VERSION : the different positions within the crystal lattice are now taken into account. A supplementary file (.sym) containing the matrices of the symmetry operations is requested.

1 - Principle and equations

Let's consider a typical oriented single-crystal magnetometry experiment : a given compound, whose susceptibility tensor is $\vec{\vec{\chi}}$, is placed in a magnetic field $\vec{H}$. The induced magnetisation $\vec{m}$ is then :

$$\vec{m} = \vec{\vec{\chi}} \times \vec{H} \quad (1)$$

where × represents the classical matrix product. Experimentally, one measures the magnetisation component along the direction of the magnetic field :

$$m = \vec{m}.\frac{\vec{H}}{H}. \quad (2)$$

For the sake of simplicity, in the following we will consider a unitary field (1 T), and drop the normalisation factor.

In a classical single-crystal SQUID measurement, one knows the orientation of the magnetic field with respect to the crystal lattice axes at any time. Typically, the measurement is performed by rotating the crystal around one of its crystal axes (or reciprocal axes). The field orientation can thus be described at any time by a rotation angle θ :

$$\vec{H} = f(\theta). \quad (3)$$

In order to determine the induced magnetisation, it is sufficient to know the susceptibility tensor in the same frame of reference as the field, and one simply needs to compute the product of equation (2). Note that this calculation actually needs to be performed for each independent molecule in the crystal lattice, the overall magnetic moment being simply the average of the magnetic moments for each position in the lattice. This is what **CalcM** does.

In the case of a PND measurement, the susceptibility tensor is expressed in a frame of reference $(\overrightarrow{v_1}, \overrightarrow{v_2}, \overrightarrow{v_3})$ such that :

- $\overrightarrow{v_1}$ is directed along $\overrightarrow{a^*}$,
- $\overrightarrow{v_2}$ is directed along $\vec{c}$,
- $\overrightarrow{v_3}$ completes the right-handed frame,

and the three vectors are unitary.

In the case of an *ab initio* calculation, the positions of the atoms must be given in a cartesian frame of reference, which then is used to express $\overleftrightarrow{\chi}$.

Thus, in the general case one needs to transform the χ tensor and/or the magnetic field, in order to express them in the same frame of reference, and this for every position in the lattice. In **CalcM**, we decided to choose the CCSL basis as a standard frame to perform the computations. We then seek to determine $\overleftrightarrow{\chi}_{CCSL}$ and $\vec{H}_{CCSL}$. In the following, we detail the operations that permit these change of frame, in the case of PND (subroutine "PND") and *ab initio* (subroutine "Abinit") data.

1.1 PND

As we already said, the tensor from a PND experiment is already expressed with respect to the CCSL basis (at least in Orphee). One then simply needs to ensure the field is also expressed in this basis. In the current version of **CalcM**, the field orientations can be given in the CCSL or crystal lattice frame of reference.

In the first case, the program simply computes the product:

$$m = \left(\overleftrightarrow{\chi}_{CCSL} \times \vec{H}_{CCSL}\right).\vec{H}_{CCSL} \quad (4)$$

straightforwardly from equation (2), for each of the possible positions in the lattice.

In the other case, we first need to transform the field from the crystalline frame to the CCSL basis. To do so, we use the change of frame matrix $\overleftrightarrow{P}_{cri \to CCSL}$ (whose columns are coordinates of the CCSL vectors expressed in the crystal cell) :

$$\vec{H}_{CCSL} = M_{cri\to CCSL} \times \vec{H}_{cri} = \begin{pmatrix} v_{1a} & v_{2a} & v_{3a} \\ v_{1b} & v_{2b} & v_{3b} \\ v_{1c} & v_{2c} & v_{3c} \end{pmatrix} \times \begin{pmatrix} H_a \\ H_b \\ H_c \end{pmatrix} \quad (5)$$

(a,b,c indices indicating the components along the corresponding crystallographic axes). Then, one simply needs to compute the product in equation (4).

Note : if the field is not unitary ($||\vec{H}_{CCSL}|| = 1$), one has to divide the computed moment in (4) by $||\vec{H}_{CCSL}||$.

1.2 Ab initio

In the case of an *ab initio* calculation, as we already indicated the susceptibility tensor is given in an orthonormal frame. Generally, the relation between the crystal and orthonormal frames is not plain. However, we know the positions of the atoms in both frames. For a given atom Xi, we will write as (x_i,y_i,z_i) its coordinates in the orthonormal frame (labelled "cal"), and (x_i^a,y_i^b,z_i^c) its fractional coordinates in the lattice cell.

Let's consider a reference atom X0 (we may for instance choose an atom on the (0,0,0) position in either frame), and three others X_1, X_2, X_3 atoms, such that (X_0, X_1, X_2, X_3) are not coplanar. We can then define three non coplanar vectors $\overrightarrow{X_0X_i}$ (i = 1,2,3) based on these atoms, expressed in the crystal or orthonormal frame :

$$\overrightarrow{X_0X_i} = \begin{pmatrix} x_i - x_0 \\ y_i - y_0 \\ z_i - z_0 \end{pmatrix} = \begin{pmatrix} p_i \\ q_i \\ r_i \end{pmatrix} \quad \text{in the orthonormal frame} \quad (6)$$

$$\overrightarrow{X_0X_i} = \begin{pmatrix} x_i^a - x_0^a \\ y_i^b - y_0^b \\ z_i^c - z_0^c \end{pmatrix} = \begin{pmatrix} p_i^a \\ q_i^b \\ r_i^c \end{pmatrix} \quad \text{in the crystal frame.} \quad (7)$$

Since these vectors are non coplanar, they permit to pave the whole (3D) space : they define a new frame of reference, non necessarily orthonormal, called in the following "molecular frame" (label "mol"). We can then change from the orthonormal frame to the molecular one by the change of basis matrix :

$$M_{cal\to mol} = \begin{pmatrix} p_1 & p_2 & p_3 \\ q_1 & q_2 & q_3 \\ r_1 & r_2 & r_3 \end{pmatrix} \quad (8)$$

and from the crystal frame to the molecular one by :

$$M_{cri\to mol} = \begin{pmatrix} p_1^a & p_2^a & p_3^a \\ q_1^b & q_2^b & q_3^b \\ r_1^c & r_2^c & r_3^c \end{pmatrix}. \quad (9)$$

Obviously the inverse change of basis matrices are also defined, from the molecular frame to the crystal one :

$$M_{mol\rightarrow cri} = M_{cri\rightarrow mol}^{-1} \tag{10}$$

and from the molecular frame to the orthonormal one :

$$M_{mol\rightarrow cal} = M_{cal\rightarrow mol}^{-1}. \tag{11}$$

It is then possible to directly change from the calculated frame to the crystal cell :

$$M_{cal\rightarrow cri} = M_{cal\rightarrow mol} \times M_{mol\rightarrow cri} = M_{cal\rightarrow mol} \times M_{cri\rightarrow mol}^{-1}. \tag{12}$$

Furthermore, we also know the expression of the change of basis matrix between the crystal cell and the CCSL basis ($M_{cri\rightarrow CCSL}$, defined in equation (5)). We can then access the change of basis matrix from the calculated frame to the CCSL basis :

$$M_{cal\rightarrow CCSL} = M_{cal\rightarrow cri} \times M_{cri\rightarrow CCSL} \tag{13}$$

$$= M_{cal\rightarrow mol} \times M_{mol\rightarrow cri} \times M_{cri\rightarrow CCSL} \times M_{cri\rightarrow CCSL} \tag{14}$$

The susceptibility tensor in the CCSL basis thus reduces to :

$$\overleftrightarrow{\chi}_{CCSL} = {}^{t}M_{cal\rightarrow CCSL} \times \overleftrightarrow{\chi}_{cal} \times M_{cal\rightarrow CCSL} \tag{15}$$

Thus, one comes to the same situations as previously described (in section 1.1).

2 - Files structure

2.1 Input and output

The program involves 4 different files : a first file containing all the structural informations (".inp"), a file listing the magnetic field orientations (".dat"), a file containing all the symmetry operation matrices (".sym") and an output file (".out"). The choice of the kind of calculation (PND or *ab initio*) is made in interactive mode (input from terminal).

2.2 Structure of the .inp file

The four first lines contain the labels and positions of the X0, X1, X2 and X3 atoms in the crystal cell. The next four lines recall the same parameters in the orthonormal frame.

Note : in the case of PND-based calculations, these informations are not needed for the calculation of magnetisation. One is free to enter any values, in the correct format.

The next line contains the susceptibility tensor in the CCSL or orthonormal frame, written line by line. The three last lines contain the CCSL vectors, expressed in the lattice cell (line vectors).

Summary of the input format :

label0 X0a Y0b Z0c
label1 X1a Y1b Z1c
label2 X2a Y2b Z2c
label3 X3a Y3b Z3c
label0 X0 Y0 Z0
label1 X1 Y1 Z1
label2 X2 Y2 Z2
label3 X3 Y3 Z3
chi11 chi12 chi13 chi21 chi22 chi23 chi31 chi32 chi33
v1a v1b v1c
v2a v2b v2c
v3a v3b v3c

2.3 Structure of the .dat file

The first lien of the orientation file must contain the number of orientations to compute. Then, the orientations are given in the following format :

theta H1 H2 H3 with theta the rotation angle, and Hi the field components in the CCSL or crystal frame of reference.

2.4 Structure of the .sym file

The symmetry file structure is close to that of the .dat file : the first line gives the number of equivalent positions, and the following lines list the matrices associated to the considered symmetry operations :

U11 U12 U13 U21 U22 U23 U31 U32 U33 with U_{ij} the matrix associated to the symmetry operation expressed in the CCSL frame.

Example : in a monoclinic group $2/m$, the equivalent positions are found by applying the 2-fold axis along $\vec{b}$

$$U(2) = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}$$

We do not consider the mirror plane, as the successive application of the 2 fold axis and the inversion will give the same result (and inversion related positions are not needed, as they yield the same magnetic moment). The .sym file will thus write as

```
      2
1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0
-1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 -1.0
```

Note 2 : *a minima*, the file must contain the identity matrix (for the reference position).

2.5 Structure of the .out files

CalcM produces 2 output files which have their first lines in common. The first bloc recalls the program version, the type of data (*ab initio* or PND) and the matrices used by the program. Then the results are written in the last bloc. The first output file (filename1.out) gives a list of angles, magnetic moments and field norm, for each position in the lattice. The second output file (filename2.out) gives the same information, but averaged over all positions in the lattice.