# Cryana Tutorial

In this tutorial you can find the way to get Cryana from ParamagneticCyana2.1.

At the end of the tutorial we explain Cryana new commands.

# Files

We refer to original ParagneticCyana2.1 files.

### New files:

*crystaldata.f*

*getcrydip.f*

### Replaced files

*findzerocoords.f*

*psegrad.f*

*pseviol.f*

### Modified files

*copy.f*

*cyana.for*

*cyanadata.f*

*fcn.f*

*grad.f*

*readf.f*

*violst.f*

*viosta.f*

# Details on modified files:

***copy.f***

add the subroutine:

c ==================================================================

c COPPSE: Copy pcs.

c

c Mauro und Enrico

c ------------------------------------------------------------------

subroutine coppse (i,j)

c

use cyanadata

c

implicit real(kind=selected\_real\_kind(12)) (a-h,o-z)

include 'pse.incl'

indxdip(j)=indxdip(i)

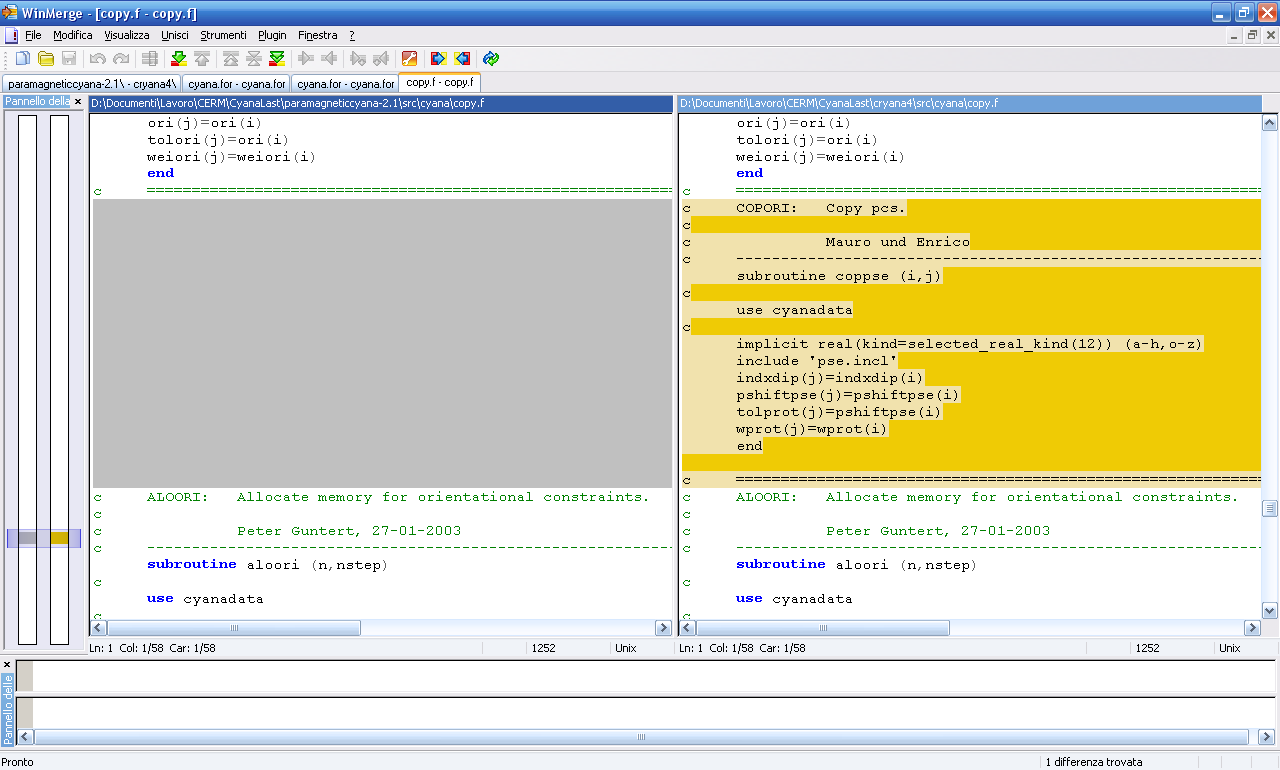
pshiftpse(j)=pshiftpse(i)

tolprot(j)=pshiftpse(i)

wprot(j)=wprot(i)

end

c ==================================================================



***cyana.for***

1. add

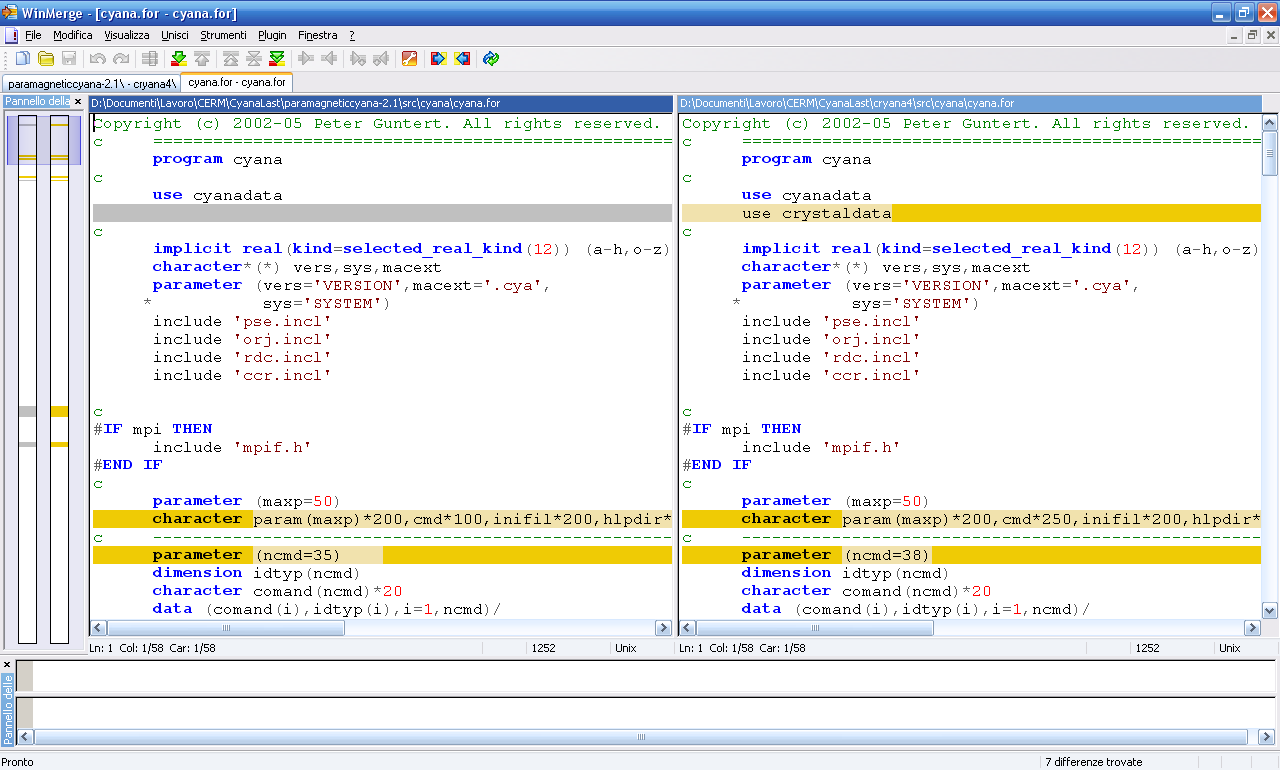
use crystaldata

1. change

cmd\*100 🡪 cmd\*250

1. add 3 parameters

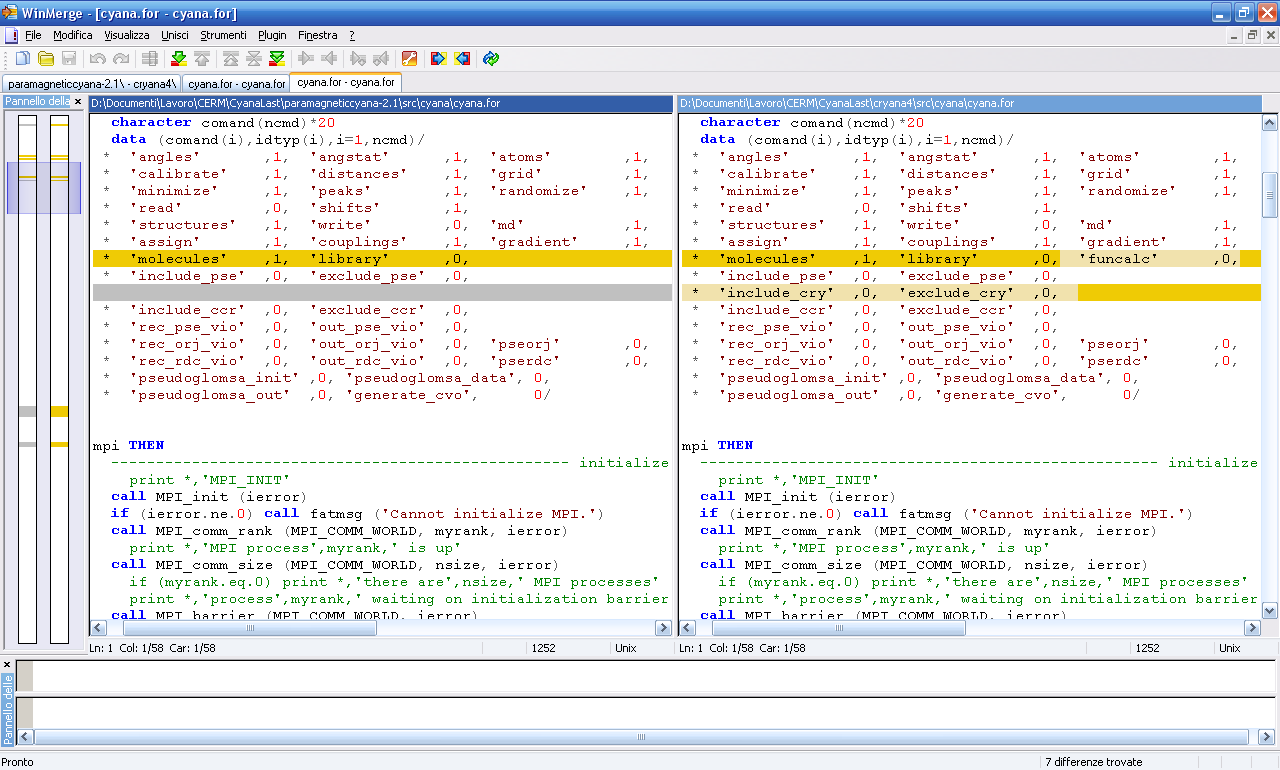
ncmd=38

******

1. add

'funcalc' ,0,

\* 'include\_cry' ,0, 'exclude\_cry' ,0,

******

1. add

case ('include\_cry')

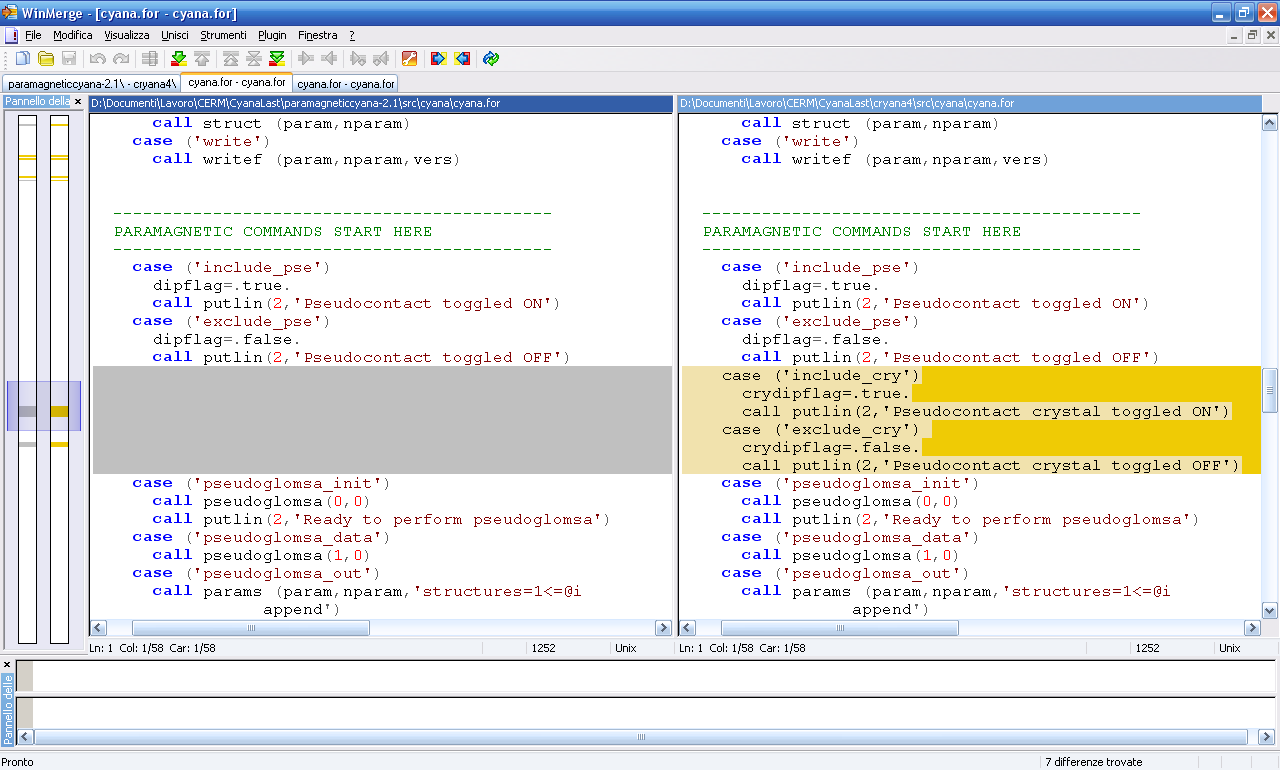
crydipflag=.true.

call putlin(2,'Pseudocontact crystal toggled ON')

case ('exclude\_cry')

crydipflag=.false.

call putlin(2,'Pseudocontact crystal toggled OFF')

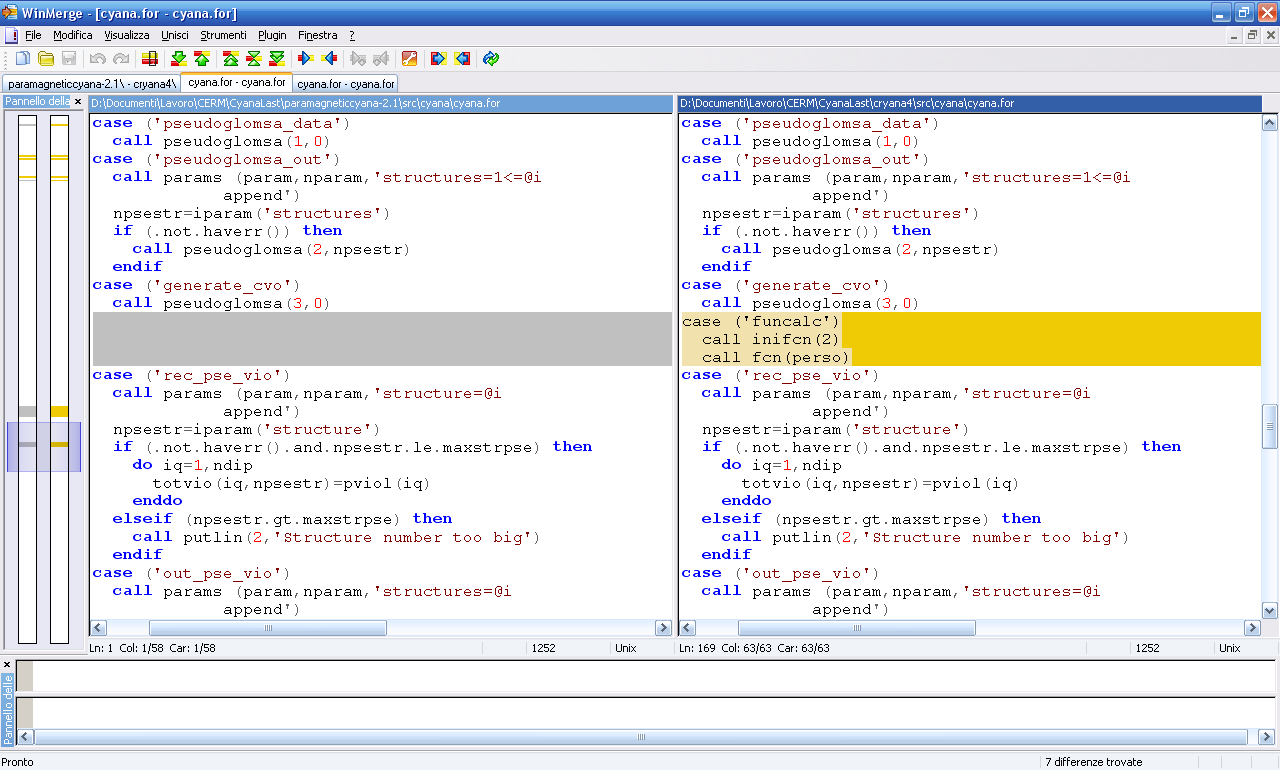
******

1. add

case ('funcalc')

call inifcn(2)

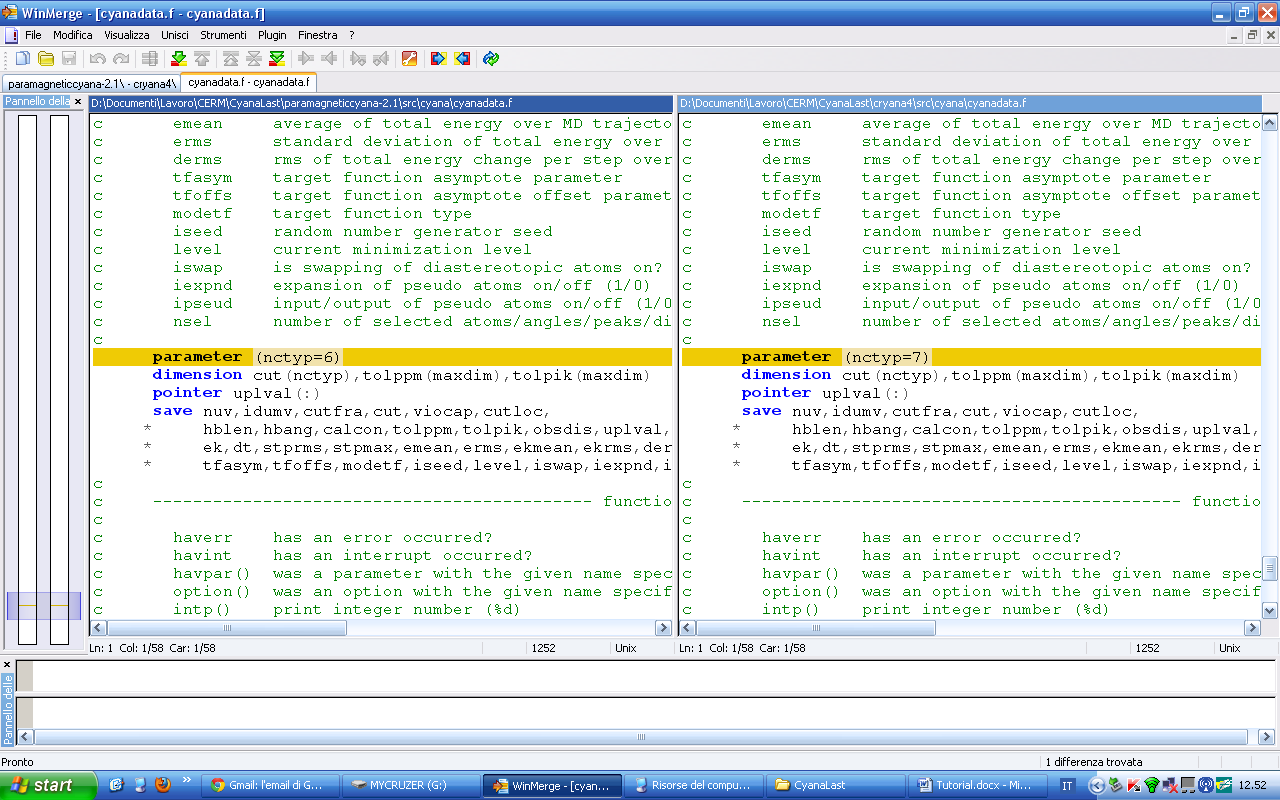
call fcn(perso)

******

***cyanadata.f***

1. change

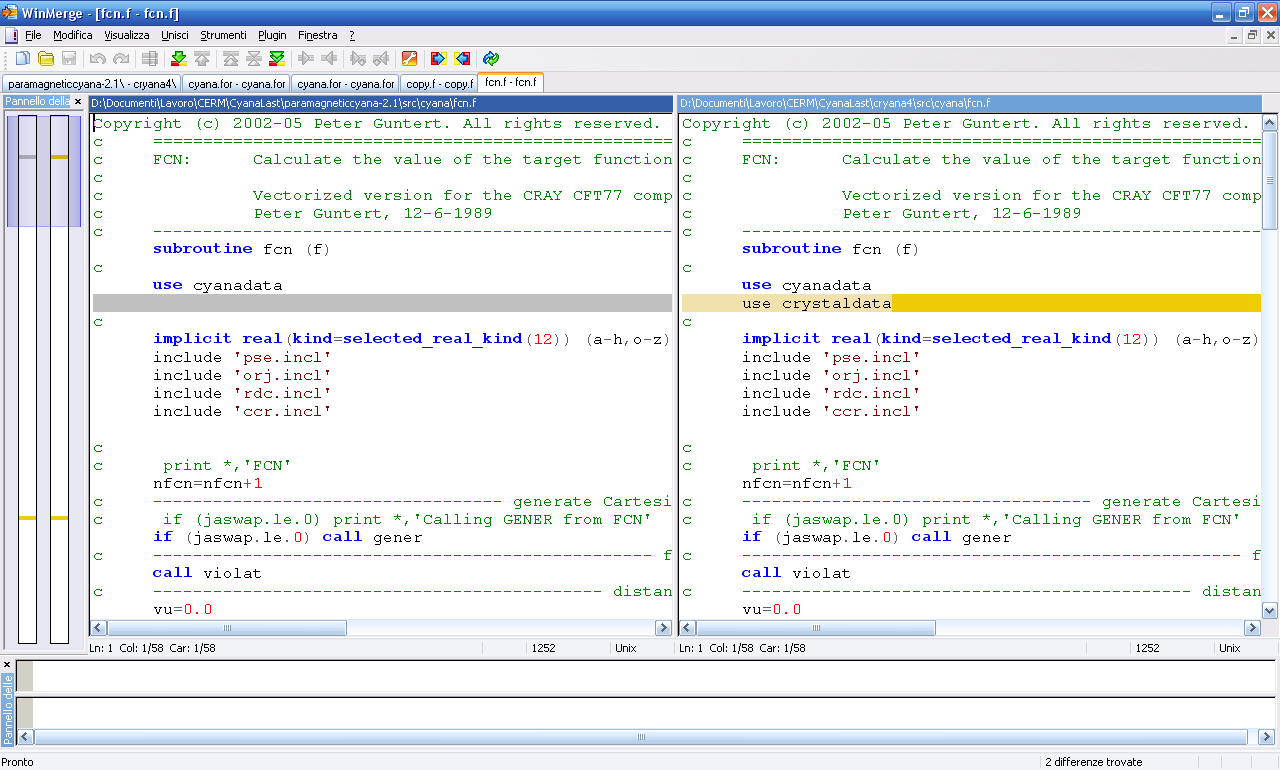
nctyp=6 🡪 nctyp=7

******

***fcn.f***

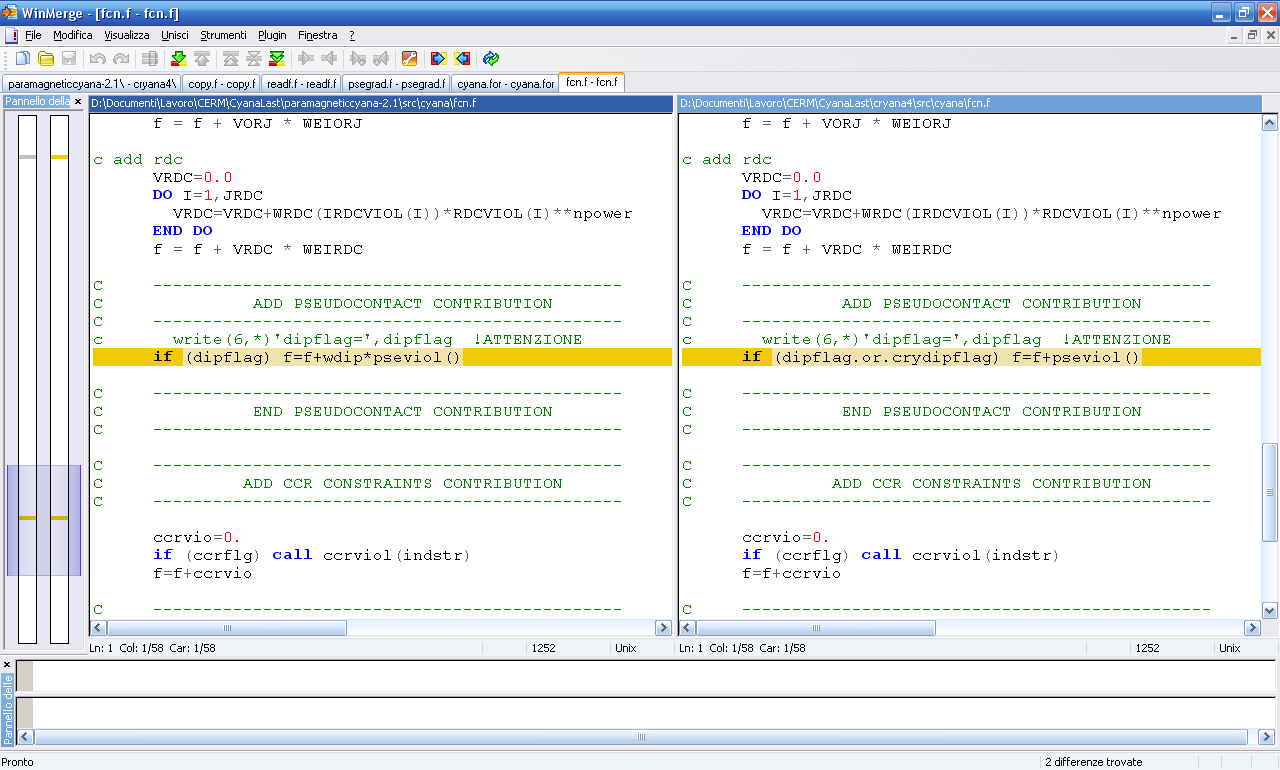
1. add

use crystaldata

******

1. change

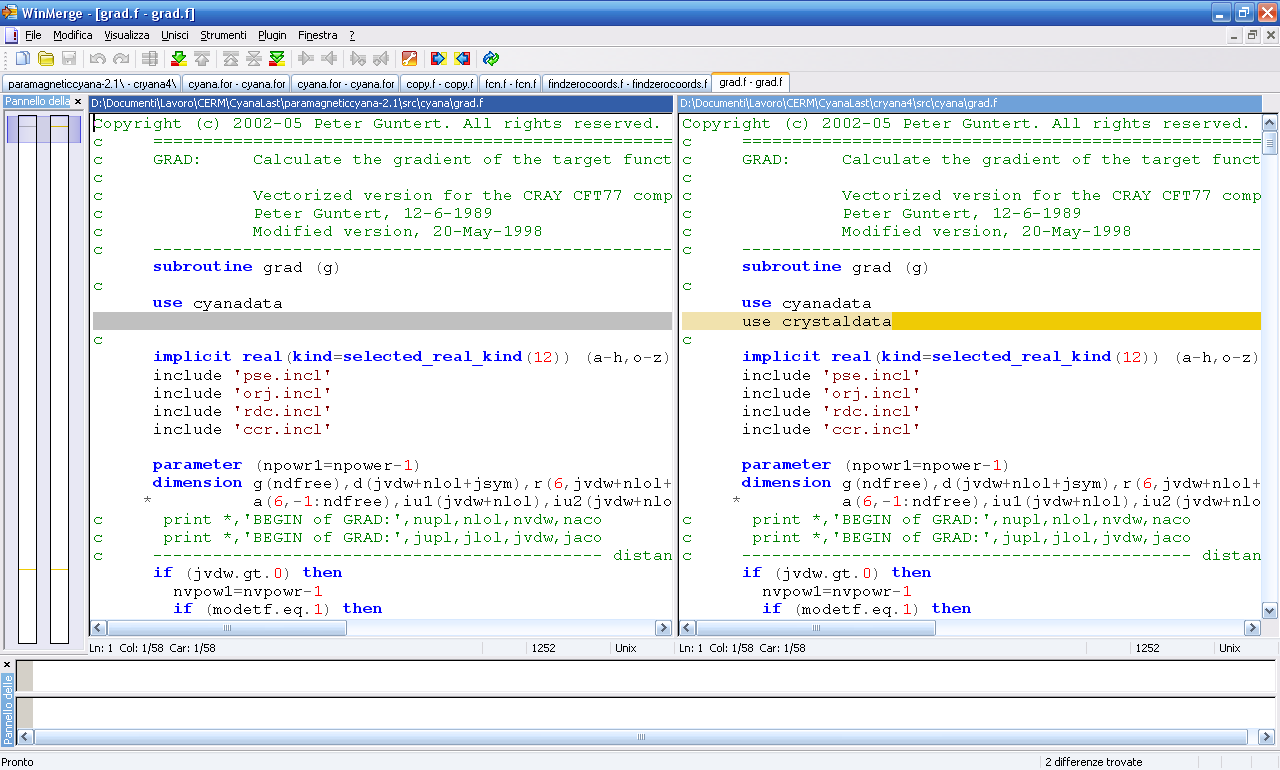
if (dipflag) f=f+wdip\*pseviol() 🡪 if (dipflag.or.crydipflag) f=f+pseviol()



***grad.f***

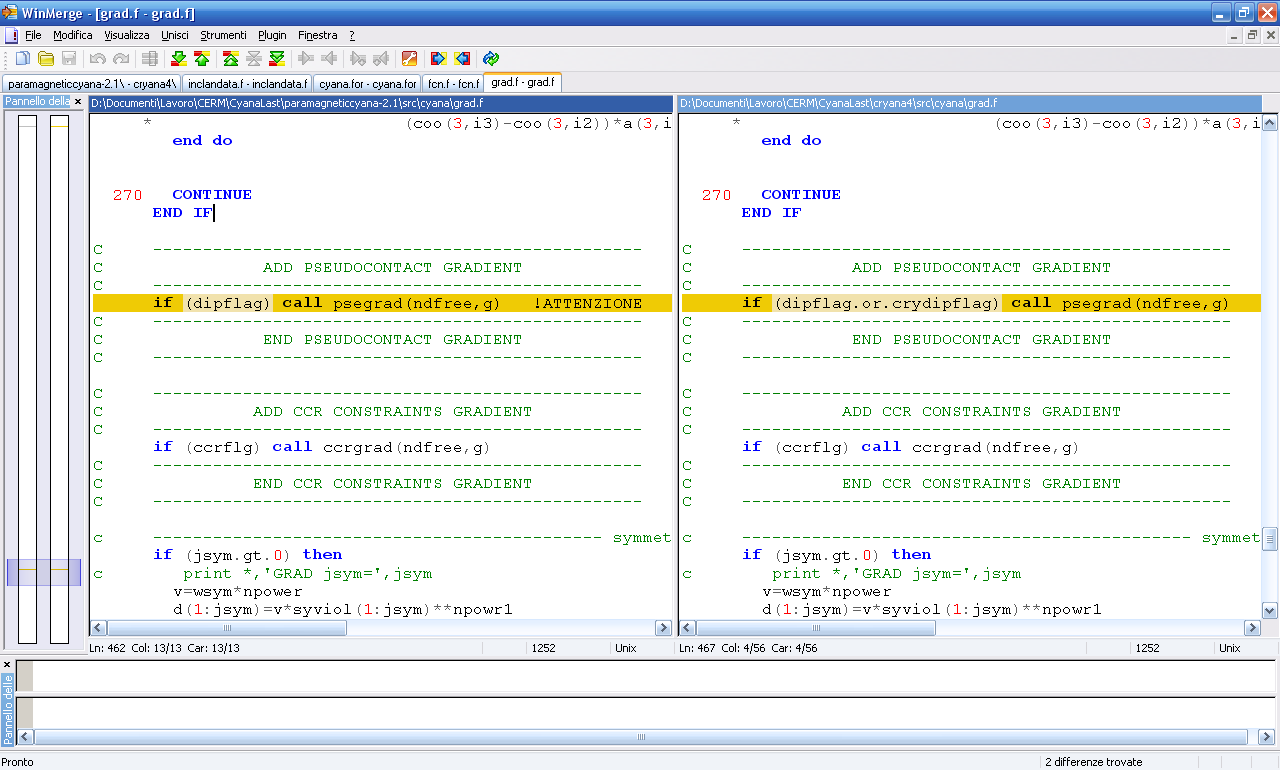
1. add

use crystaldata

******

1. change

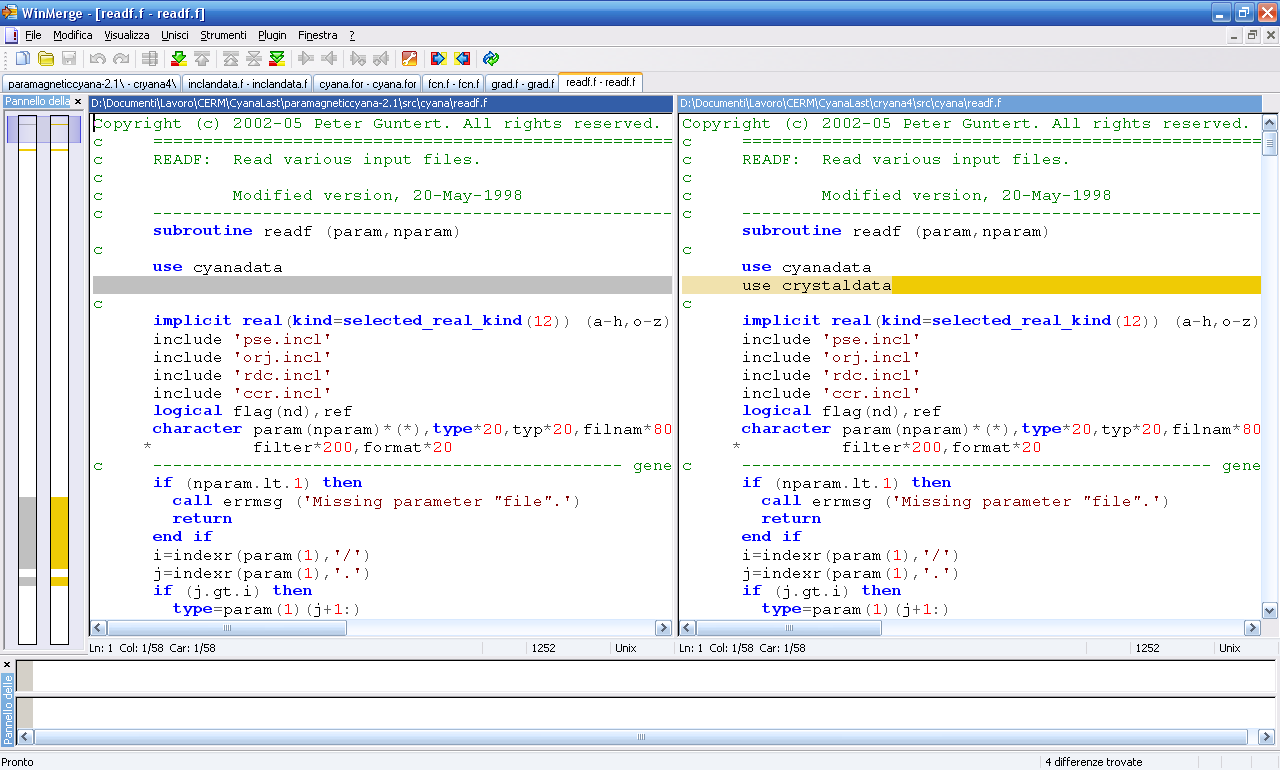
if (dipflag) call psegrad(ndfree,g) 🡪 if (dipflag.or.crydipflag) call psegrad(ndfree,g)

******

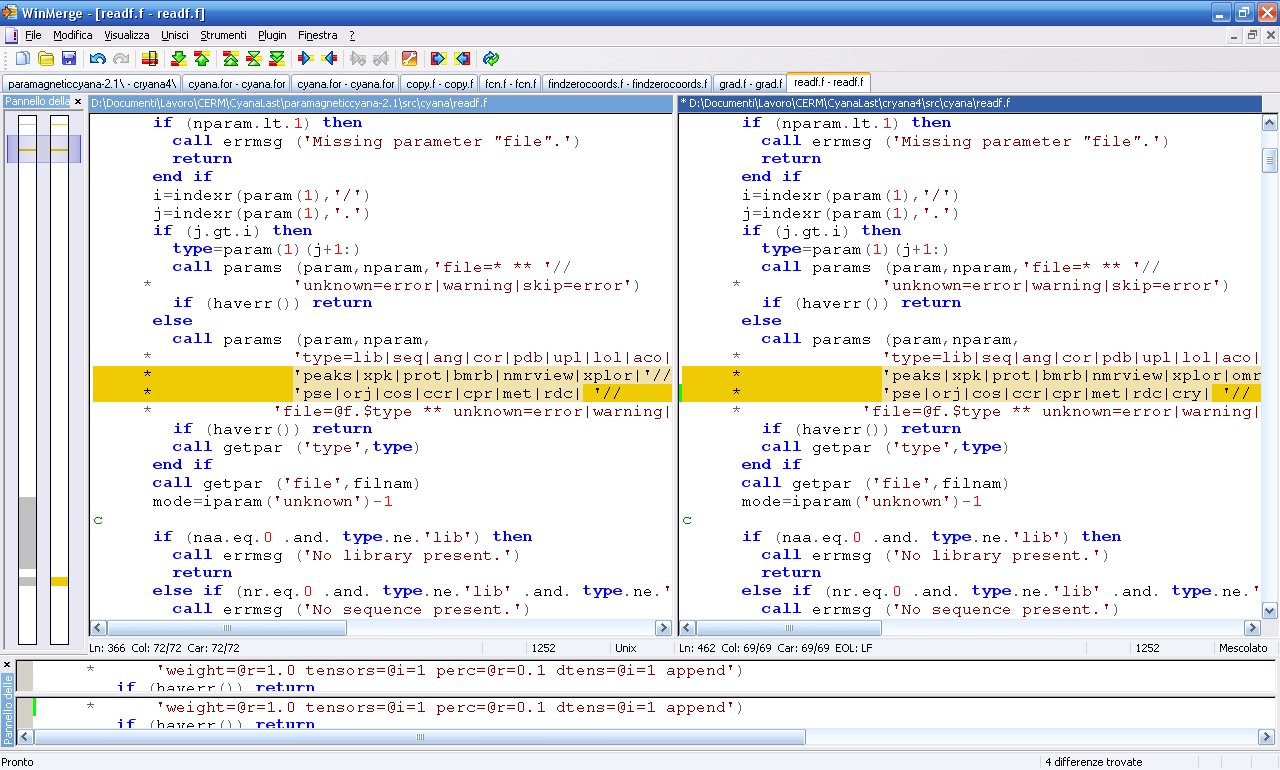
***readf.f***

1. add

use crystaldata

******

1. add omr and cry

******

1. add cry case

c ------------------------------------------------ pseudo

case ('cry')

call params (param,nparam,'tolerance=@r=0.4 crystal=@i=1 '//

\* 'weight=@r=1.0 tensors=@i=1 perc=@r=0.1 dtens=@i=1 '//

\* 'OMweight=@r=10.0 OMtol=@r=0.1 '//

\* 'celldimx=@r=0.0 celldimy=@r=0.0 celldimz=@r=0.0 '//

\* 'symmetry=none|P21212|P212121|P222|P2221|=none append')

if (haverr()) return

OMflag=.false.

crytoldip=rparam('tolerance')

cryncrystal=iparam('crystal')

crywdip=rparam('weight')

cryntenso=iparam('tensors')

crypseperc=rparam('perc')

cryNDTENSO=iparam('dtens')

cellDim(1)=rparam('celldimx')

cellDim(2)=rparam('celldimy')

cellDim(3)=rparam('celldimz')

call getpar ('symmetry',symmetry)

crystalmode=cryncrystal

if (cellDim(1).eq.0.0.or.cellDim(2).eq.0.0.or.cellDim(3).eq.0.0)

\* then

CALL ERRMSG('Please insert crystal cell dimensions '//

\* 'celldimx, celldimy, celldimz')

stop

end if

select case (cryncrystal)

case (0)

SolidState=.false.

case (1)

SolidState=.true.

call putlin (2,'Crystal mode')

!inizializzazione matrici di rotazione per pcs con stato solido

call initRotMatrix

case (2)

SolidState=.true.

OMflag=.true.

OMweight = rparam('OMweight')

OMtol = rparam('OMtol')

call putlin (2,'Crystal mode with origin '//

\* 'forced alignment with internal metal')

call putlin (2,'Do not forget to upload a matrix '//

\* 'by a .omr file')

call initRotMatrix

case default

CALL ERRMSG('wrong crystal mode! select 0 or 1')

end select

if (cryNDTENSO.ne.1.and.cryntenso.ne.1) then

write(6,\*) 'OPTION NOT YET IMPLEMENTED'

stop

end if

if (cryNDTENSO\*cryntenso.gt.maxtenscry/2) then

CALL ERRMSG('Too many tensors!')

stop

endif

if (.not.option('append')) ndipcry=0

n=ndipcry

call getcrydip(1,mode)

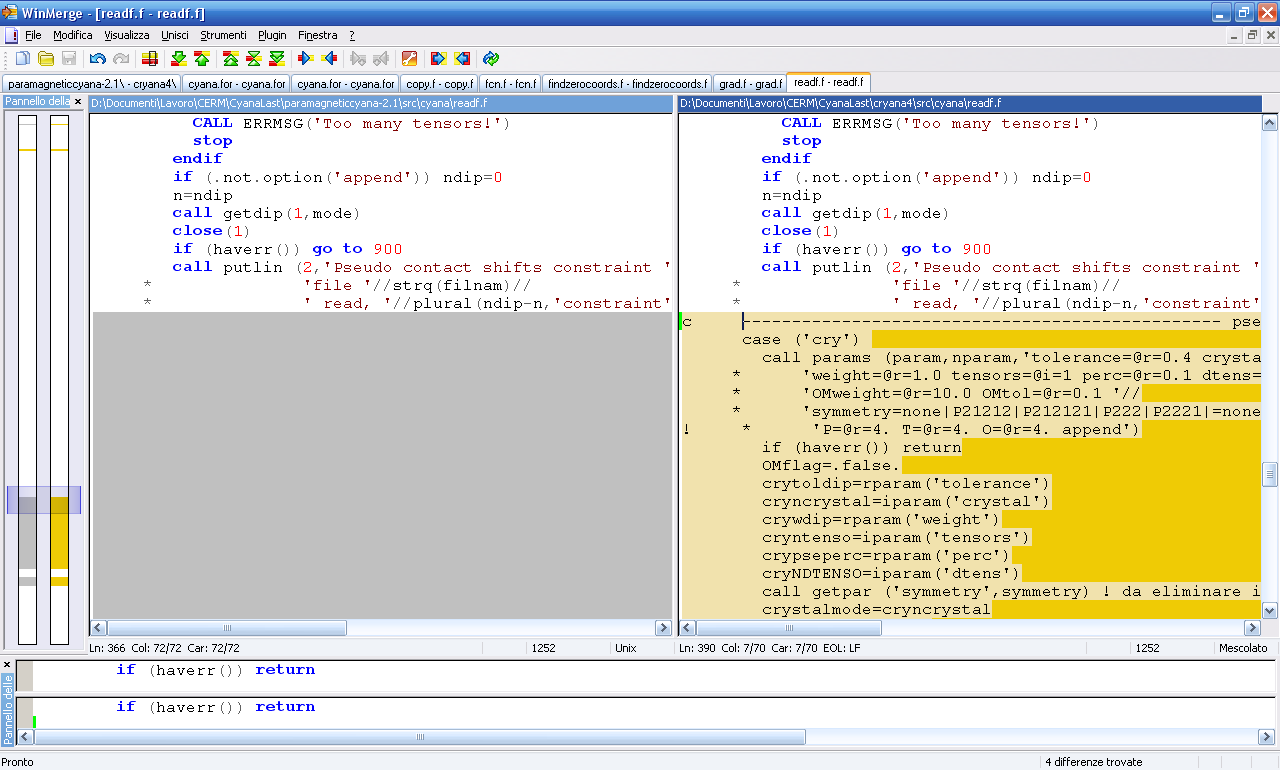
close(1)

if (haverr()) go to 900

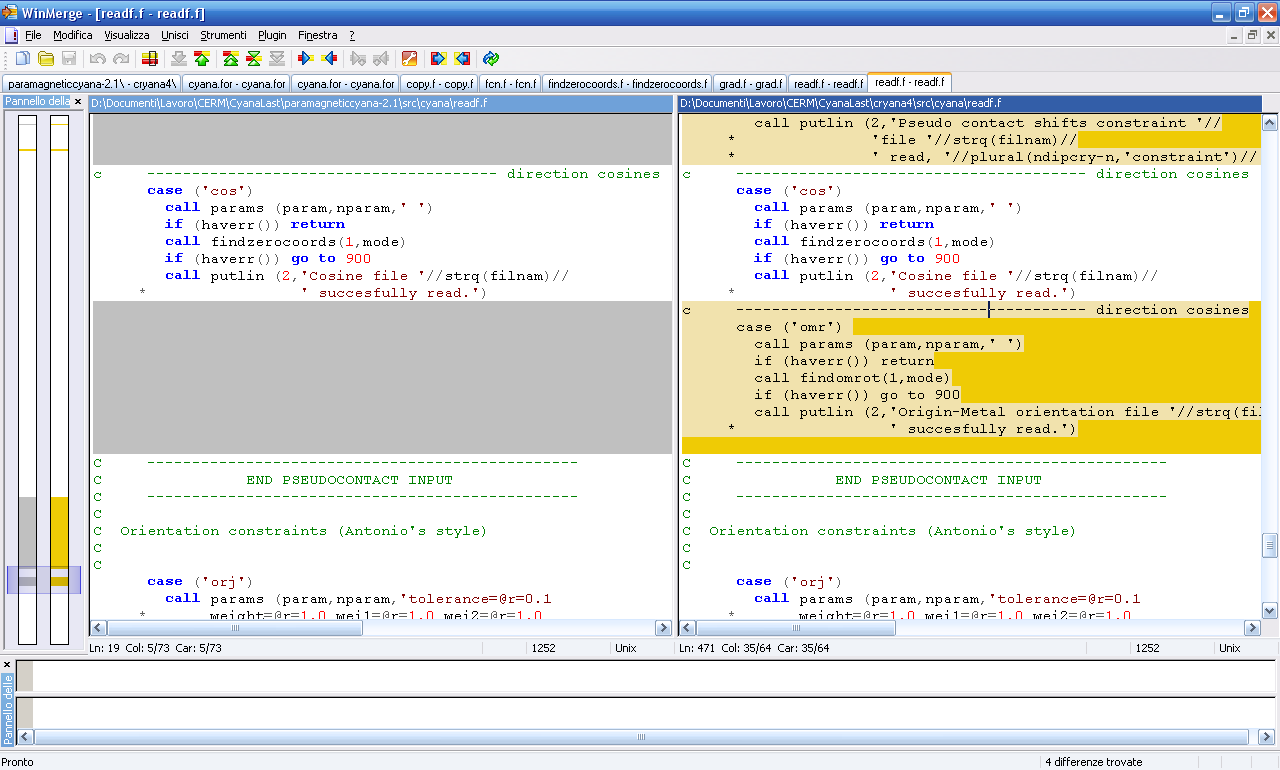
call putlin (2,'Pseudo contact shifts constraint '//

\* 'file '//strq(filnam)//

\* ' read, '//plural(ndipcry-n,'constraint')//'.')

******

1. add omr case

******

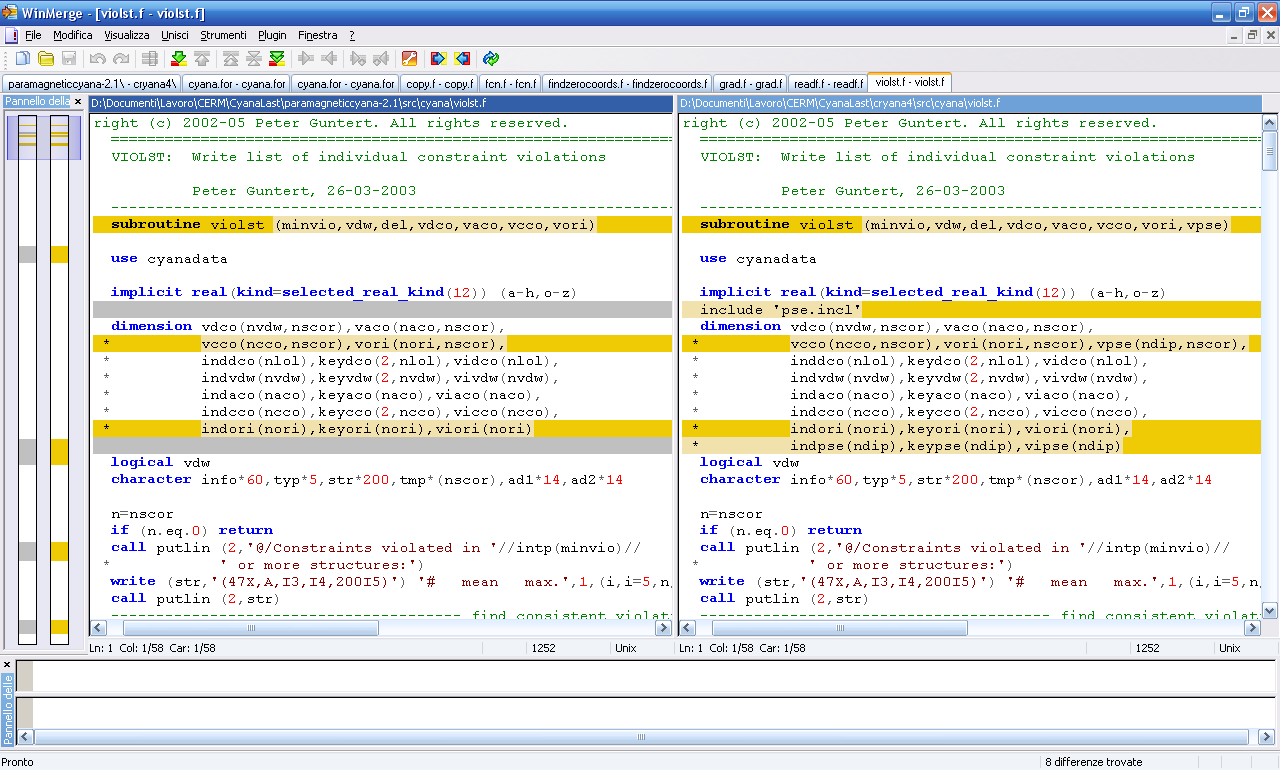
***violst.f***

1. add

include ‘pse.incl’

1. add variables

vpse(ndip,nscor),indpse(ndip),keypse(ndip),vipse(ndip)

******

1. add

nvpse=0

cut(7)=toldip

do i=1,ndip

nvio=count((vpse(i,1:n).gt.cut(7)))

if (count((vpse(i,1:n).gt.cut(7))).ge.minvio) then

nvpse=nvpse+1

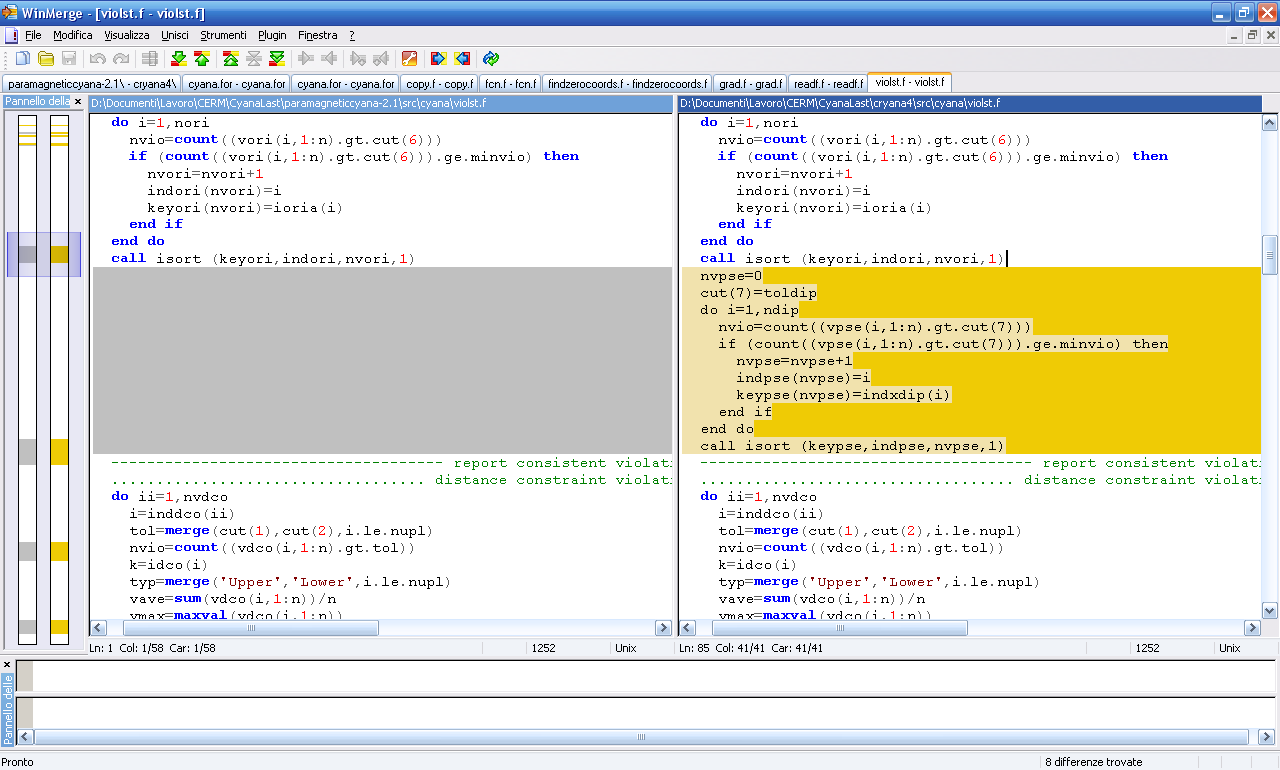
indpse(nvpse)=i

keypse(nvpse)=indxdip(i)

end if

end do

call isort (keypse,indpse,nvpse,1)

******

1. add

c ................................... pseudocontact shift violations

do ii=1,nvpse

i=indpse(ii)

nvio=count((vpse(i,1:n).gt.cut(7)))

vave=sum(vpse(i,1:n))/n

vmax=maxval(vpse(i,1:n))

vipse(ii)=vave

tmp=' '

do j=1,n

if (vpse(i,j).gt.cut(7))

\* tmp(j:j)=merge('\*','+',vpse(i,j).eq.vmax)

end do

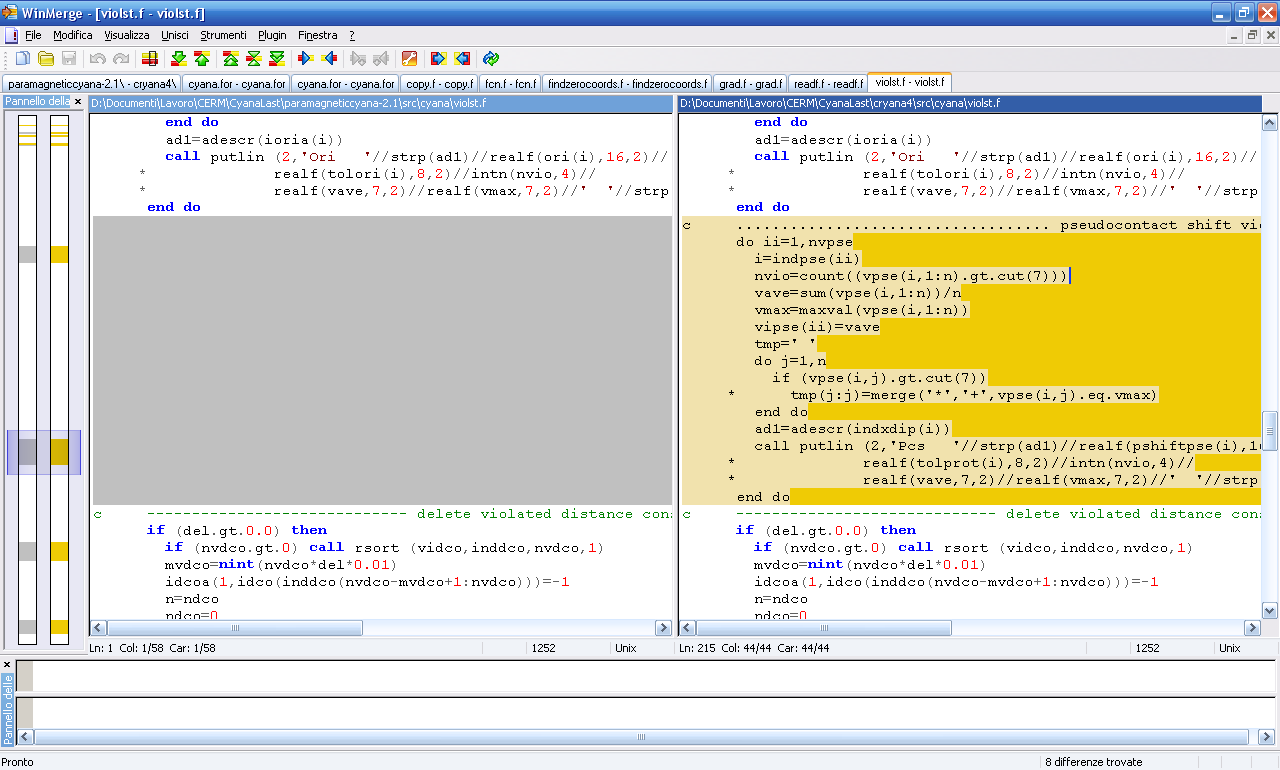
ad1=adescr(indxdip(i))

call putlin (2,'Pcs '//strp(ad1)//realf(pshiftpse(i),16,2)//

\* realf(tolprot(i),8,2)//intn(nvio,4)//

\* realf(vave,7,2)//realf(vmax,7,2)//' '//strp(tmp))

end do

******

1. add

c ------------------------ delete pseudocontact shifts constraints

if (nvpse.gt.0) call rsort (vipse,indpse,nvpse,1)

mvpse=nint(nvpse\*del\*0.01)

indxdip(indpse(nvpse-mvpse+1:nvpse))=-1

n=npse

npse=0

do i=1,n

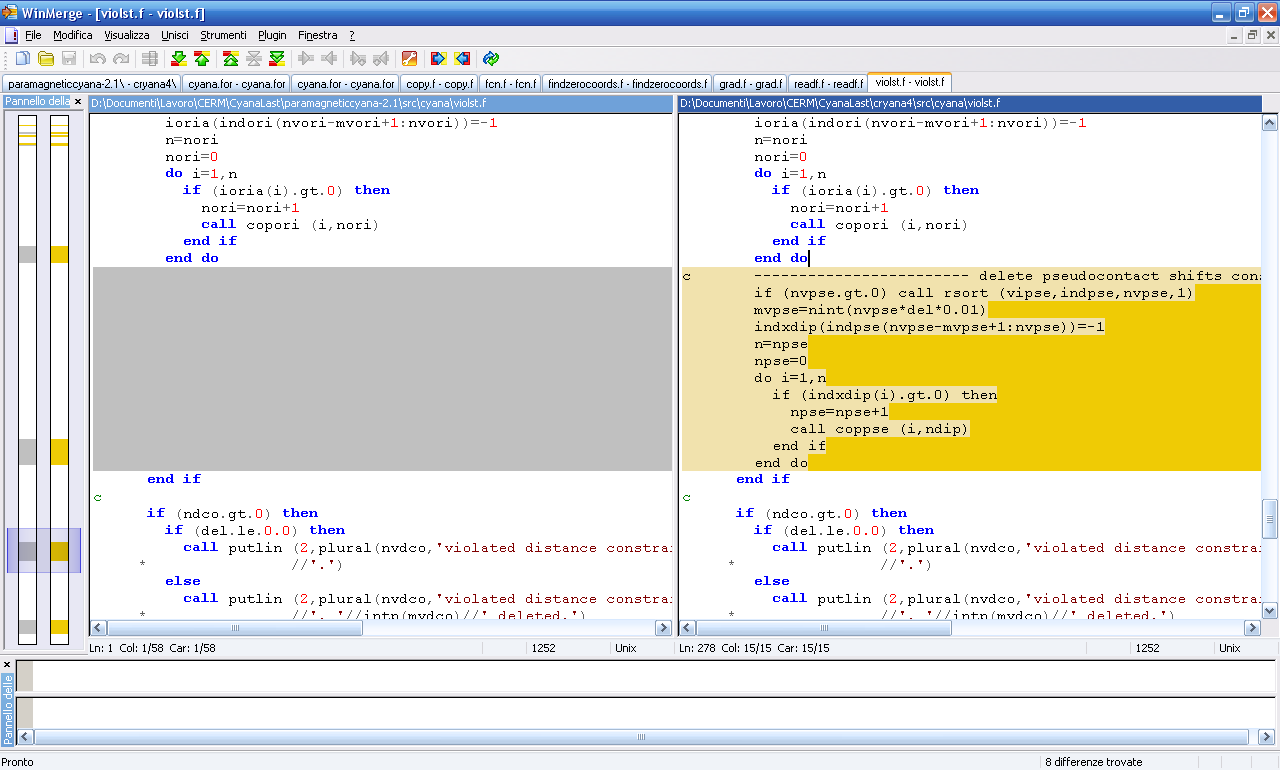
if (indxdip(i).gt.0) then

npse=npse+1

call coppse (i,ndip)

end if

end do

******

1. add

if (ndip.gt.0) then

if (del.le.0.0) then

call putlin (2,plural(nvpse,'violated pcs '//

\* 'constraint')//'.')

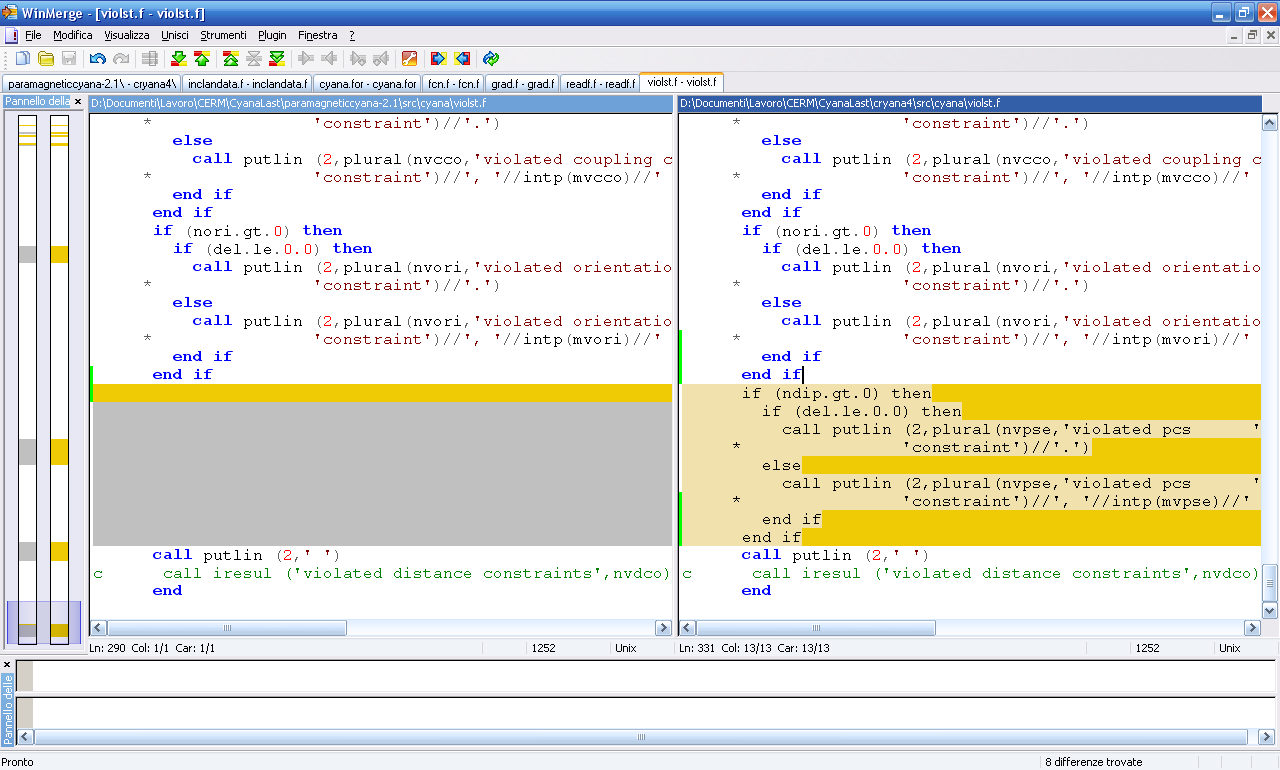
else

call putlin (2,plural(nvpse,'violated pcs '//

\* 'constraint')//', '//intp(mvpse)//' deleted.')

end if

end if

******

***viosta.f***

1. add

include ‘pse.incl’

1. add variables

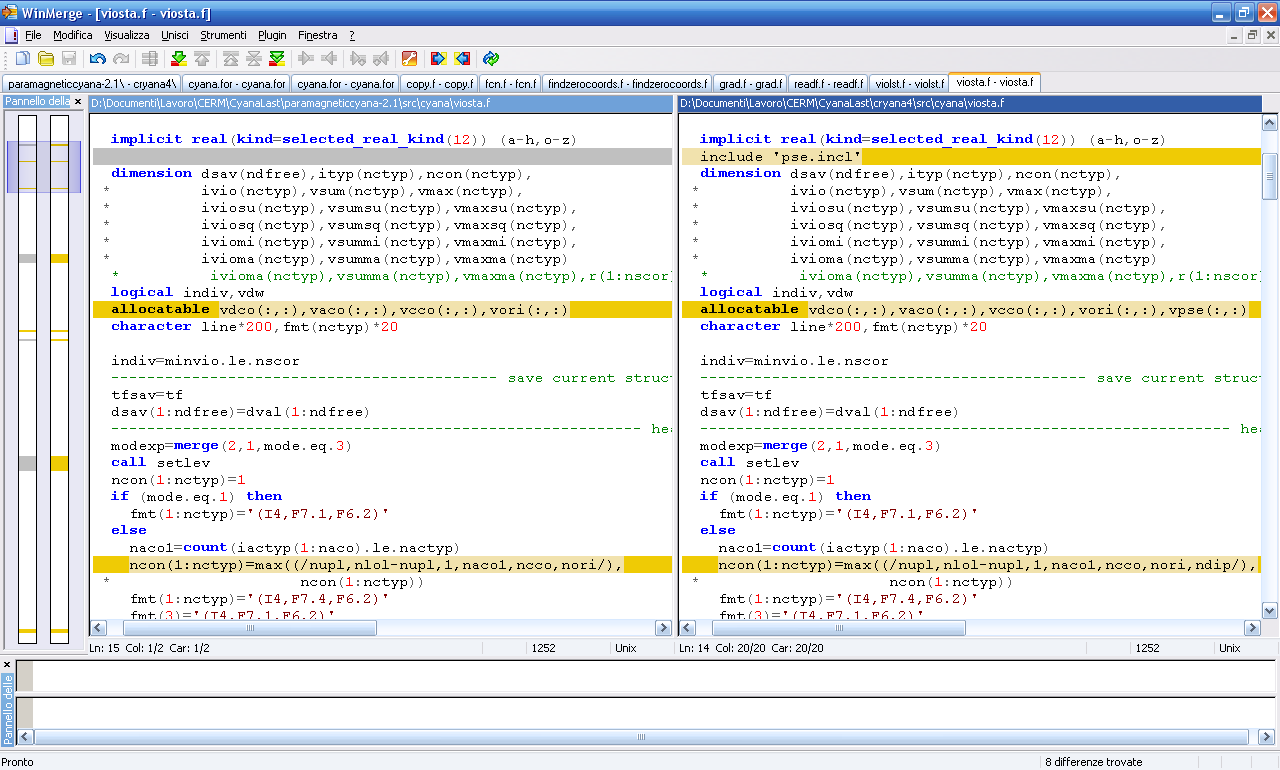
vpse(:,:),

1. change

ncon(1:nctyp)=max((/nupl,nlol-nupl,1,naco1,ncco,nori/),

🡪

ncon(1:nctyp)=max((/nupl,nlol-nupl,1,naco1,ncco,nori,ndip/),

******

1. add

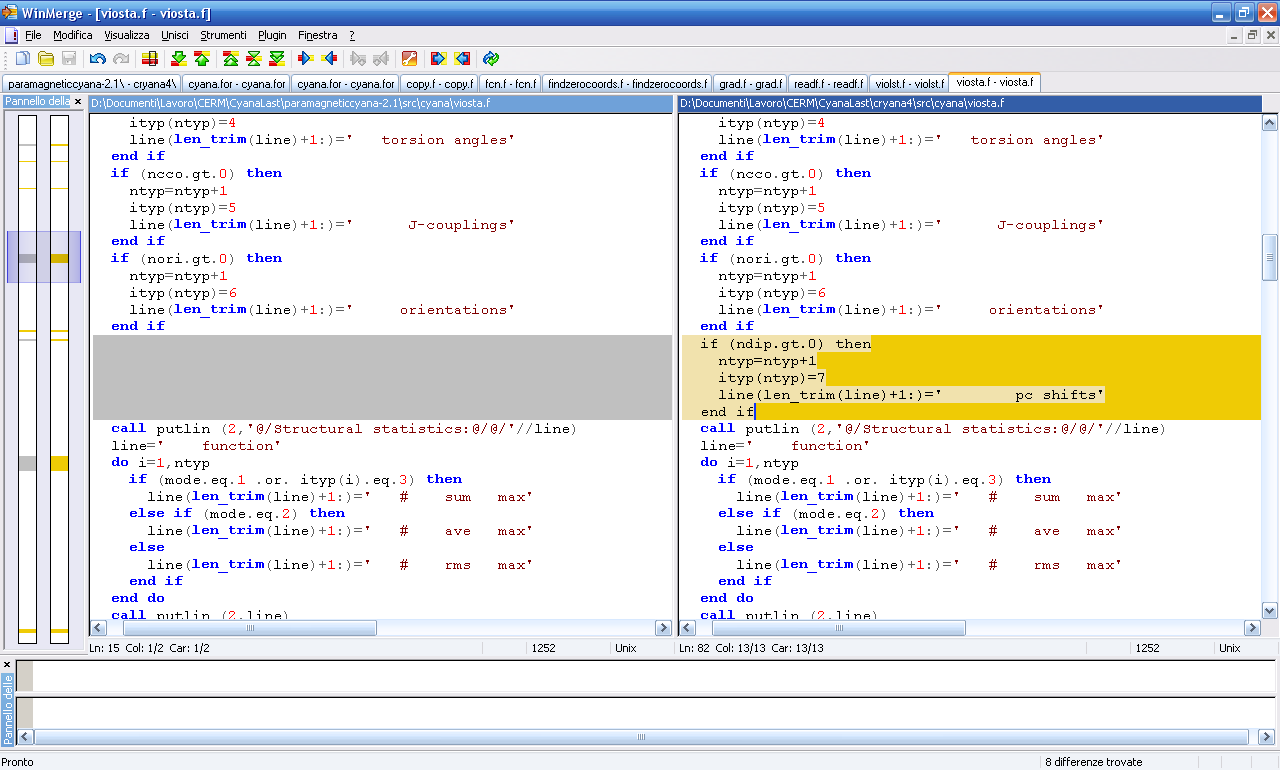
if (ndip.gt.0) then

ntyp=ntyp+1

ityp(ntyp)=7

line(len\_trim(line)+1:)=' pc shifts'

end if

******

1. change

allocate (vdco(nvdw,nscor),vaco(naco,nscor),

\* vcco(ncco,nscor),vori(nori,nscor))

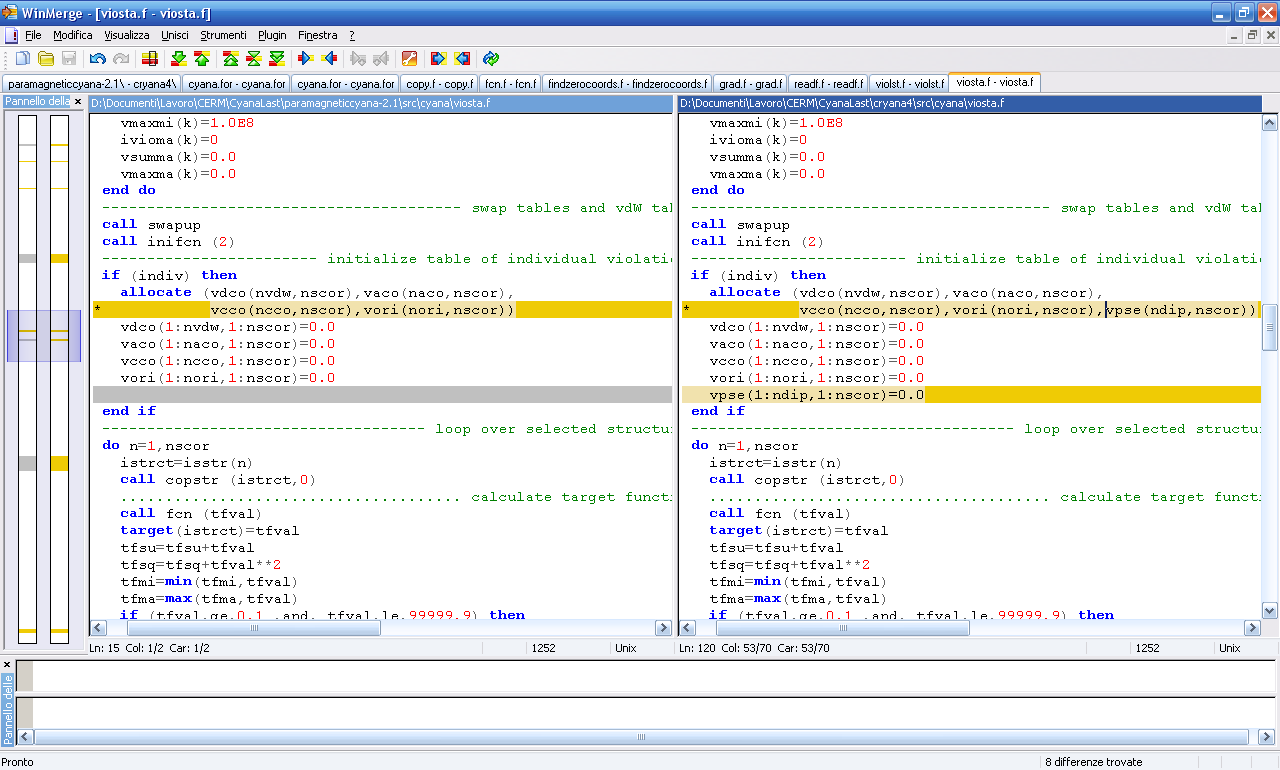
🡪

allocate (vdco(nvdw,nscor),vaco(naco,nscor),

\* vcco(ncco,nscor),vori(nori,nscor),vpse(ndip,nscor))

1. add

vpse(1:ndip,1:nscor)=0.0

******

1. add

k=7

do i=1,ndip ! jpsevio. ora glieli facciamo scrivere tutti

v=abs(pviol(i))

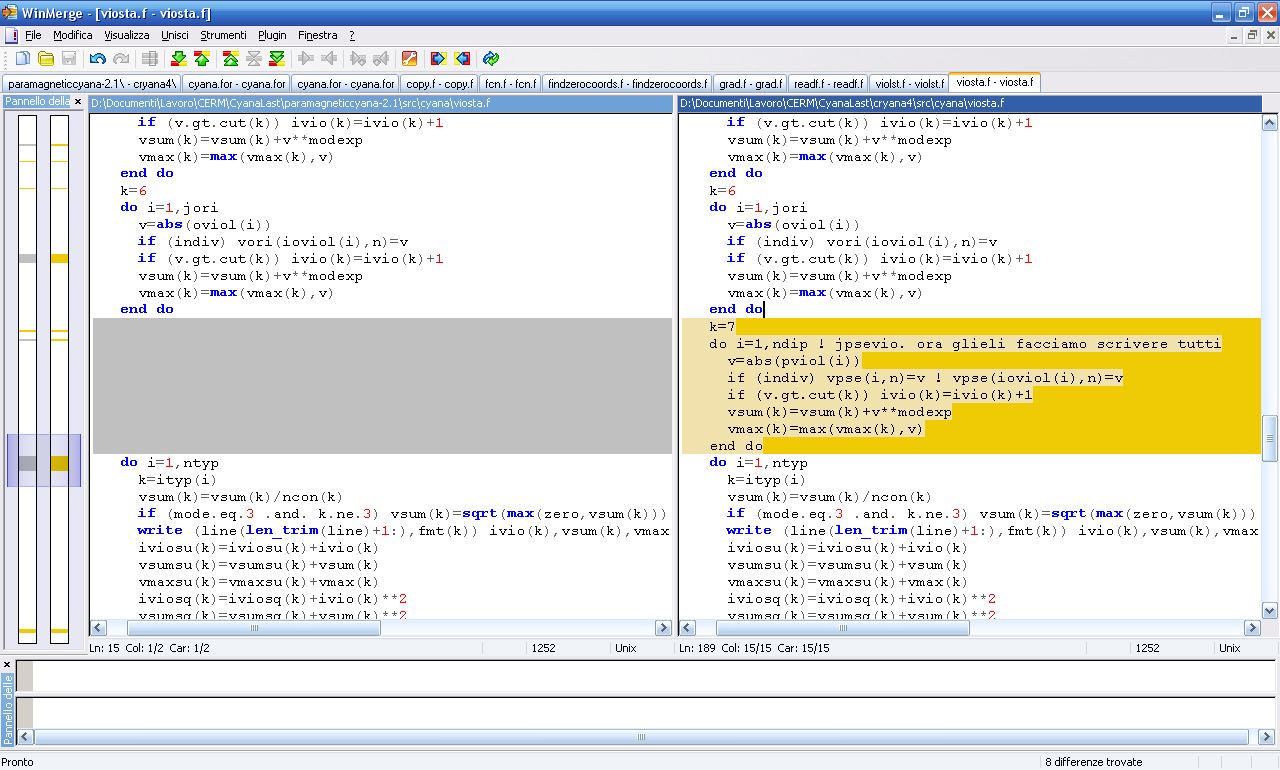
if (indiv) vpse(i,n)=v ! vpse(ioviol(i),n)=v

if (v.gt.cut(k)) ivio(k)=ivio(k)+1

vsum(k)=vsum(k)+v\*\*modexp

vmax(k)=max(vmax(k),v)

end do

******

1. change

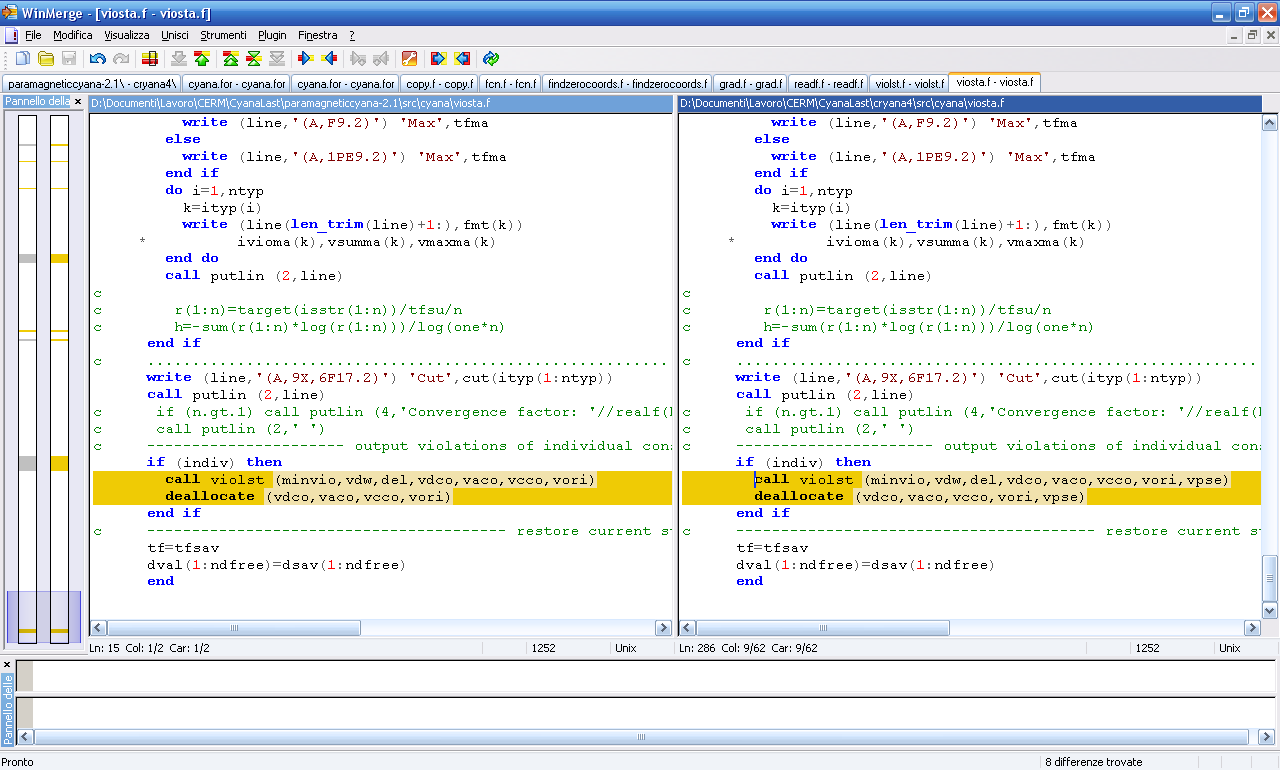
call violst (minvio,vdw,del,vdco,vaco,vcco,vori)

deallocate (vdco,vaco,vcco,vori)

🡪

call violst (minvio,vdw,del,vdco,vaco,vcco,vori,vpse)

deallocate (vdco,vaco,vcco,vori,vpse)

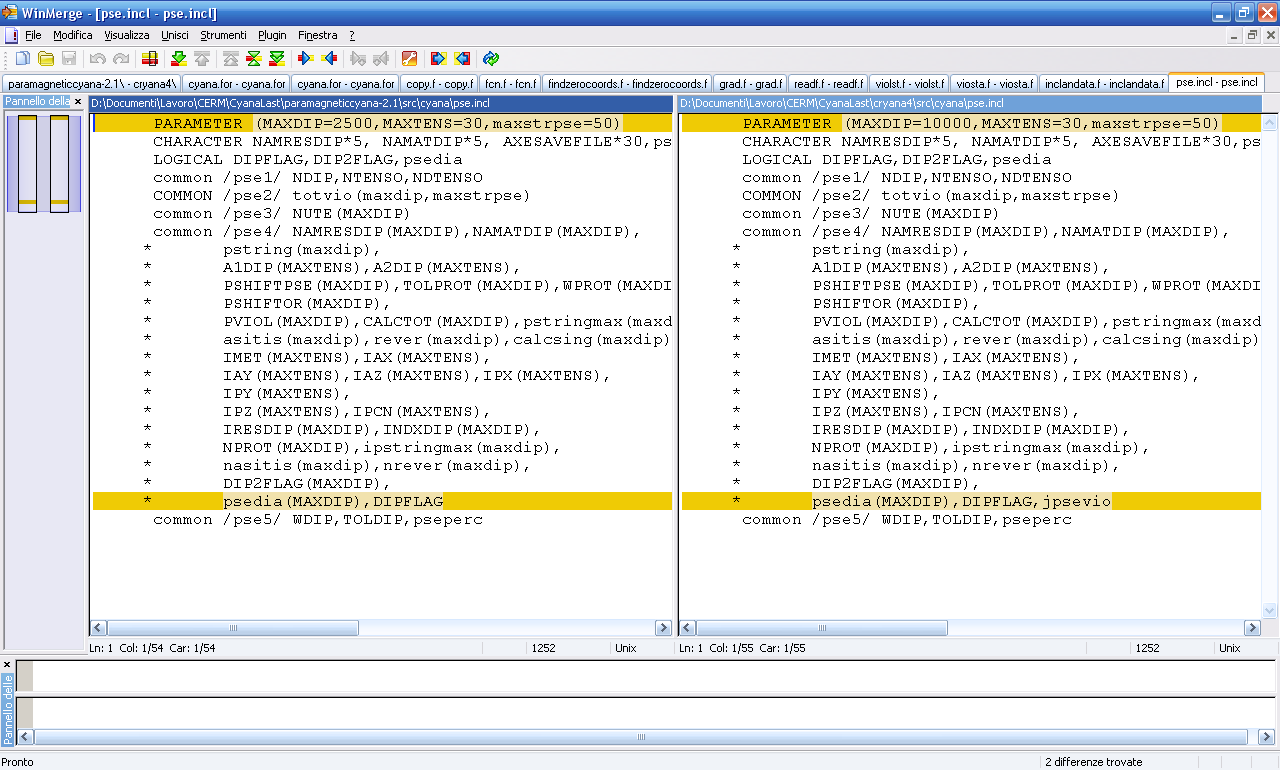
******

***pse.incl***

1. change

MAXDIP=2500 🡪 MAXDIP=10000

1. add variable jpsevio

******

# Commands

This section reviews syntax and parameters of the new commands introduced.

To read a pse file that is being used considering the contribution to pcs of external symmetric metals, a typical command is the following:

read cry observed\_pcs.pse tensors=2 weight=0.5 tolerance=0.2 perc=0.2 symmetry=P21212 ↵

celldimx=69.194 celldimy=62.564 celldimz=37.262

read cos anisoTensor.cos

Reading the cry file, tensors is the number of tensors present in the sequence, including the crystal origin tensor, and they must be specified at the end of the pse file (the origin have to be the last tensor in pse file).

The symmetry allowed at the moment are P222, P2221, P21212, P212121 .

celldimx, celldimy and celldimz are the dimensions of the asymmetric unit cell.

The meaning of other parameters is obvious.

The cos file is the file containing the information of susceptibility magnetic tensor.

If we need to fix a specific orientation for the origin with respect to the metal tensor orientation we can use the following command:

read cry observed\_pcs.pse tensors=2 weight=0.5 tolerance=0.2 perc=0.2 symmetry=P21212 ↵

celldimx=69.194 celldimy=62.564 celldimz=37.262 crystal=2 OMweight=100.0  
read cos anisoTensor.cos  
read omr XY.omr

The parameter crystal=2 (default is crystal=1) means that an omr file needs to be read to restrain the reciprocal orientation of the two tensors. The omr file is just a file containing a matrix. If M is the matrix containing in rows the coordinates of the unit vectors AX, AY and AZ defining the metal tensor orientations and O is the matrix containing in rows the coordinates of the unit vectors defining crystallographic origin orientations, the omr file contains the matrix MTO.

Other useful added commands are:

include\_cry

exclude\_cry

to include/exclude the contribution of the global pcs restraints in the calculation of the TF.

### An example of pse file

155 GLN C 0.30 1 0.20 1.50 1

155 GLN CA 0.52 1 0.20 1.50 1

155 GLN CB 0.30 1 0.20 1.50 1

155 GLN CG 0.31 1 0.20 1.50 1

155 GLN CD 0.24 1 0.20 1.50 1

156 SER C 0.39 1 0.20 1.50 1

156 SER CA 0.52 1 0.20 1.50 1

156 SER CB 0.15 1 0.20 1.50 1

157 LEU C 0.56 1 0.20 1.50 1

157 LEU CA 0.58 1 0.20 1.50 1

157 LEU CB 0.81 1 0.20 1.50 1

157 LEU CG 0.52 1 0.20 1.50 1

200 LTNS ME

40 LTNS ME

### An example of cos file

1.00000E+00 0.00000E+00 0.00000E+00

0.00000E+00 1.00000E+00 0.00000E+00

0.00000E+00 0.00000E+00 1.00000E+00

7.00370E-32 -2.33456E-32

The last two numbers of cos file are the axial and rhombic components of the susceptibility anisotropy tensor.

### An example of omr file

0.78169E+00 -0.39277E+00 0.48486E+00

-0.55380E+00 -7.92354E-02 0.82870E+00

-0.28648E+00 -0.91584E+00 -0.28045E+00