Supplementary material

Table S1: Computed harmonic vibrational frequencies of the monomers with CCSD(T)-F12a/aug-cc-pVTZ method.

Frequency	C-bound	O-bound	Cross-bound	S1	S2	S3
$\nu(O_1C_1O_1)$ asym stretch	2387.79	2386.52	2388.02	3311.64	2386.38	2387.58
$\nu(C_2O_3)$ stretch	2159.51	2149.73	2152.56	1513.40	2152.09	2152.40
$\nu_s(O_1C_1O_2)$ sym stretch	1348.14	1347.91	1347.60	1319.99	1347.47	1348.06
$\delta_{oop}(O_1C_1O_2)$ in plane bend	670.56	669.92	670.01	i721.19	668.56	669.38
$\delta(O_1C_1O_2)$ out of plane bend	663.99	667.34	669.67	537.39	667.54	668.92
$\delta(C_1C_2O_3)$ inter-molecular stretch	106.42	67.34	68.15	i511.49	48.43	59.97
$\nu(C_1C_2)$ in plane rock	63.87	63.79	41.90	472.11	38.87	41.22
$\delta_{oop}(C_1C_2O_3)$ in plane rock	59.50	40.75	25.81	445.98	24.28	10.65
τ inter-molecular torsion	32.14	27.35	23.08	102.60	i37.85	i7.79

1 Program 1

```
program pot_co-co2

implicit none
integer :: i, j, k
integer, parameter:: nc=134, nr=63, l1max=10, l2max=6, mmax=l1max+l2max
real(8) :: rr, theta1, theta2, phi,theta1d, theta2d, phid, v, facdeg, pi,
   dr
real(8) :: start, finish
real(8) :: x(nr), z(nr), z2(nr), ainp(nr,nc), z3(nr,nc)
real(8) :: k1, k2, zp1, zpn, x0, dx, xint, zint, som
real(8) :: th1, th2, ph, som2, sph, aparity, cfit(nc),fact(0:301)

common /COEF/ainp, x
common /spl/ z3
common /factorial/fact
!dimension x0(4)

call cpu_time(start)

pi=4.d0*datan(1.d0)
facdeg=pi/180.d00

open (5,file="coef.inp")
read(5,*) ainp
close(5)

open (6,file="r.inp")
read(6,*) x
close(6)

call factrl(fact)

zp1=0.0; zpn=0.0;
do i=1,nc
z=ainp(:,i)
call spline(x,z,nr,zp1,zpn,z2)
z3(:,i)=z2
enddo

open (8,file="res.out")


! Give the values of (R,th1,th2,ph), [a0.degrees]
rr= 6.6d0;
theta1d=90.0d0; theta2d=0.0d0; phid=0.0d0;
theta1=facdeg*theta1d; theta2=facdeg*theta2d; phi=facdeg*phid;
call pot(rr,theta1,theta2,phi,v)
write(8,'(4f10.3,␣2x,2f15.4)') rr, theta1d, theta2d, phid, v


end program pot_co-co2


subroutine pot(rr,th1,th2,ph,v)
implicit none
integer, parameter:: nc=134, nr=63, l1max=10, l2max=6, mmax=l1max+l2max
```

```
real(8) :: x(nr), z(nr), z2(nr), v
real(8) :: rr, k1, k2, zp1, zpn, x0, dx, xint, zint, ainp(nr,nc), som, pi
real(8) :: th1, th2, ph, som2, sph, aparity, cfit(nc), z3(nr,nc)
integer:: n,i, lt, l2mx, lmx, l1, l2, lmi, l, m, mx, nmax, ii
real(kind=8) :: fact(0:301),w3js, w6js
common /COEF/ainp, x
common /spl/ z3
common /factorial/fact
!dimension x4(4)

pi=4.d0*datan(1.d0)




xint=rr;


zp1=0.0; zpn=0.0;
do i=1,nc
z=ainp(:,i)
z2=z3(:,i)
call splint(x,z,z2,nr,xint,zint)
cfit(i)=zint
enddo


som2=0; som=0;
lt=0;
do l1=0,l1max,2
l2mx=min0(l1,l2max)
do l2=0,l2max!,2
lmx=l1+l2
lmx=min0(lmx,mmax)
lmi=iabs(l1-l2)
do l=lmi,lmx,2
lt=lt+1
mx=min0(l1,l2)
som=w3js(l1,l2,l,0,0,0,fact)*sph(l1,0,th1,0.d0)*sph(l2,0,th2,0.d0)
do m=1,mx,1
som=som+2.d0*APARITY(m)*w3js(l1,l2,l,m,-m,0,fact)*sph(l1,m,th1,0.d0)*sph(
   l2,m,th2,ph)
enddo
som2=som2+som*dfloat(2*l+1)*dsqrt(1.d0/(4.d0*pi))*cfit(lt)
enddo
enddo
enddo

v=som2

end subroutine pot(rr,th1,th2,ph,v)

SUBROUTINE spline(x,y,n,yp1,ypn,y2)
INTEGER n,NMAX
REAL(8) :: yp1,ypn,x(n),y(n),y2(n)
PARAMETER (NMAX=500)
INTEGER ::i,k
REAL(8):: p,qn,sig,un,u(NMAX)
if (yp1.gt..99e30) then
```

```
y2(1)=0.
u(1)=0.
else
y2(1)=-0.5
u(1)=(3./(x(2)-x(1)))*((y(2)-y(1))/(x(2)-x(1))-yp1)
endif
do 11 i=2,n-1
sig=(x(i)-x(i-1))/(x(i+1)-x(i-1))
p=sig*y2(i-1)+2.
y2(i)=(sig-1.)/p
u(i)=(6.*((y(i+1)-y(i))/(x(i+1)-x(i))-(y(i)-y(i-1))/(x(i)-x(i-1)))/(x(i+1)
   -x(i-1))-sig*u(i-1))/p
11 continue
if (ypn.gt..99e30) then
qn=0.
un=0.
else
qn=0.5
un=(3./(x(n)-x(n-1)))*(ypn-(y(n)-y(n-1))/(x(n)-x(n-1)))
endif
y2(n)=(un-qn*u(n-1))/(qn*y2(n-1)+1.)
do 12 k=n-1,1,-1
y2(k)=y2(k)*y2(k+1)+u(k)
12 continue
return
END

SUBROUTINE splint(xa,ya,y2a,n,x,y)
INTEGER n
REAL(8):: x,y,xa(n),y2a(n),ya(n)
INTEGER k,khi,klo
REAL(8) :: a,b,h
klo=1
khi=n
1 if (khi-klo.gt.1) then
k=(khi+klo)/2
if(xa(k).gt.x)then
khi=k
else
klo=k
endif
goto 1
endif
h=xa(khi)-xa(klo)
if (h.eq.0.) pause 'bad␣xa␣input␣in␣splint'
a=(xa(khi)-x)/h
b=(x-xa(klo))/h
y=a*ya(klo)+b*ya(khi)+((a**3-a)*y2a(klo)+(b**3-b)*y2a(khi))*(h**2)/6.
return
END

!
   ....................................................................


double precision FUNCTION APARITY(I)
IMPLICIT DOUBLE PRECISION (A-H,O-Z)
APARITY=1.d0
IF ((I/2)*2.NE.I) APARITY=-1.d0
```

```
RETURN
END
!
   .................................................................

!**************************************************************
double precision FUNCTION PLM(LIN,MINa,COSTH)
IMPLICIT real*8 (a-h,o-z)
if (ABS(COSTH).GT.1.0d00)THEN
WRITE(6,*)'***ILLEGAL␣ARGEMENT␣TO␣PLM␣X=',COSTH
STOP
ENDIF
! SAVE IN LOCAL VARIABLES
L=LIN
M=IABS(MINa)
X=COSTH
!
IF(M.GT.L)THEN
PLM=0.0d00
ENDIF
LMAX=L
IF (M.GT.0) GO TO 5
!
PLM=1.0d0
PM2=0.d0
XL=0.d0
DO 2 L=1,LMAX
XL=XL+1.0d0
PP=((2.d0*XL-1.d0)*X*PLM-(XL-1.d0)*PM2)/XL
PM2=PLM
2 PLM=PP
GO TO 9000
!
5 IMAX=2*M
RAT=1.d0
AI=0.d0
DO 6 I=2,IMAX,2
AI=AI+2.d0
6 RAT=RAT*((AI-1)/AI)
! Y=SIN(THETA)
Y=SQRT(1.0d0-X*X)
PLM=SQRT(RAT)*(Y**M)
PM2=0.d0
LOW=M+1
XL=LOW-1
DO 10 L=LOW,LMAX
XL=XL+1.0d0
AL=DBLE((L+1)*(L-1))
AL=1.d0/AL
AL2=DBLE((L+M-1)*(L-M-1))*AL
AL=SQRT(AL)
AL2=SQRT(AL2)
PP=(2.d0*XL-1.D0)*X*PLM*AL-PM2*AL2
PM2=PLM
10 PLM=PP
PLM=PLM*AAPARITY(MINa)
!
!9000 PLM=PLM*SQRT(XL+0.5d0)
9000 PLM=PLM
```

```
RETURN
END
!
   ................................................................................

double precision FUNCTION AAPARITY(I)
IMPLICIT DOUBLE PRECISION (A-H,O-Z)
AAPARITY=1.d0
IF ((I/2)*2.NE.I) AAPARITY=-1.d0
RETURN
END
!
   ................................................................................

subroutine imprime(imp,n,l,K,text)
implicit real(kind=8) (a-h,o-z)
integer, parameter :: dp = selected_real_kind(15, 307)
real(kind=dp) , intent(in), dimension(0:n-1,0:l-1) :: K
integer , intent(in) :: imp, n, l
character(len=10), intent(in) :: text
integer :: irest,i,j
write(imp,*) text
irest=(l/10)
do j=1,irest
write(imp,'(10I14)') (i,i=(j-1)*10+1,10*j)
do i=0,n-1
write(imp,'(I3,10E14.6)') i+1,K(i,(j-1)*10:j*10-1)
enddo
enddo
if (irest*10 .NE. n) then
write(imp,'(10I14)') (i,i=(j-1)*10+1,l-1)
do i=0,n-1
write(imp,'(I3,10E14.6)') i+1,K(i,irest*10:l-1)
enddo
endif
end subroutine imprime

function sph (l, m, thrad, phirad)
implicit none
integer, parameter :: wp = kind(1.0d0)
real(wp), parameter :: pi = 3.14159265358979323846_wp
integer, intent(in) :: l, m
real(wp), intent(in) :: thrad, phirad
integer :: itmin1, itmin2, itmin, itmax1, itmax2, itmax, it, iphase, &
ia, ib, ic
real(wp) :: sqrt_fac, sumt, denom, term, dlm0, const, cosb2, sinb2
real(wp) :: sph, exphi
real(wp), external :: fac10

cosb2 = dcos(thrad/2.0_wp)
sinb2 = dsin(thrad/2.0_wp)


itmin1 = 0
itmin2 = m
itmin = max(itmin1,itmin2)
itmax1 = l+m
itmax2 = l
```

```
itmax = min(itmax1,itmax2)
! write (6,'(10X,A,2I6)') ' itmin, itmax = ', itmin, itmax
sqrt_fac = dsqrt( fac10(l+m) * fac10(l-m) * fac10(l) * fac10(l) )
!
sumt = 0.0_wp
do it = itmin, itmax
iphase = (-1)**it
ia = l + m - it
ib = l - it
ic = it - m
! write (6,'(10X,A,5I6)') ' it, iphase, ia, ib, ic = ', it, iphase, ia, ib
   , ic
denom = fac10(ia) * fac10(ib) * fac10(it) * fac10(ic)
term = iphase * cosb2**(ia+ib) * sinb2**(it+ic) / denom
sumt = sumt + term
end do
dlm0 = sqrt_fac * sumt
const = dsqrt( (2.0_wp *l + 1.0_wp) / (4.0_wp * pi) )
exphi = dcos( dfloat(m) * phirad )
sph = const * exphi * dlm0
!
return
end function sph


function fac10 (n)
implicit none
integer, parameter :: wp = kind(1.0d0)
!----------------------------------------------
! formal arguments
!----------------------------------------------
integer, intent(in) :: n
!----------------------------------------------
! local variables
!----------------------------------------------
integer :: i
real(wp) :: fac10, q
! ---------------------------------------------
if (n == 0) then
fac10 = 1.0_wp
else
fac10 = 1.0_wp
q = 1.0_wp
do i = 1, n
fac10 = fac10 * q / 10.0_wp
q = q + 1.0_wp
end do
endif
!
return
end function fac10

function w3js(j1,j2,j3,m1,m2,m3,fact)
integer :: m1, m2, m3, j1, j2, j3
integer :: ia, ib, ic, id, ie, im, ig, ih, z, zmin, zmax, jsum
real(kind=8) :: w3js, denom, cc, cc1, cc2
real(kind=8),intent(in) :: fact(0:301)

w3js = 0.d0
```

```
if (m1+m2+m3 /= 0) goto 1000
ia = j1 + j2
if (j3 > ia) goto 1000
ib = j1 - j2
if (j3 < abs(ib)) goto 1000
jsum = j3 + ia
ic = j1 - m1
id = j2 - m2

if (abs(m1) > j1) goto 1000
if (abs(m2) > j2) goto 1000
if (abs(m3) > j3) goto 1000
ie = j3 - j2 + m1
im = j3 - j1 - m2
zmin = max0(0,-ie,-im)
ig = ia - j3
ih = j2 + m2
zmax = min0(ig,ih,ic)
cc = 0.d0
do z = zmin, zmax, 2
denom = fact(z/2)*fact((ig-z)/2)*fact((ic-z)/2)*fact((ih-z)/2)*&
fact((ie+z)/2)*fact((im+z)/2)
if (mod(z,4) /= 0) denom = -denom
cc = cc + 1.d0/denom
enddo
cc1 = fact(ig/2)*fact((j3+ib)/2)*fact((j3-ib)/2)/fact((jsum+2)/2)
cc2 = fact((j1+m1)/2)*fact(ic/2)*fact(ih/2)*fact(id/2)*fact((j3-m3)/2)*
   fact((j3+m3)/2)
cc = cc * sqrt(cc1*cc2)
if (mod(ib-m3,4) /= 0) cc = -cc
w3js = cc
if (abs(w3js) < 1.d-8) w3js = 0.d0
1000 return
end function w3js

function w6js(j1,j2,j3,l1,l2,l3,fact)
integer :: j1,j2,j3,l1,l2,l3
integer :: ia, ib, ic, id, ie, iif, ig, ih, sum1, sum2, sum3, sum4
integer :: w, wmin, wmax, ii, ij, ik
real(kind=8) :: w6js, omega, denom, theta1, theta2, theta3, theta4, theta
real(kind=8),intent(in) :: fact(0:301)

w6js = 0.d0
ia = j1 + j2
if (ia < j3) goto 1000
ib = j1 - j2
if (abs(ib) > j3) goto 1000
ic = j1 + l2
if (ic < l3) goto 1000
id = j1 - l2
if (abs(id) > l3) goto 1000
ie = l1 + j2
if (ie < l3) goto 1000
iif = l1 - j2
if (abs(iif) > l3) goto 1000
ig = l1 + l2
if (ig < j3) goto 1000
ih = l1 - l2
if (abs(ih) > j3) goto 1000
```

```
sum1=ia + j3
sum2=ic + l3
sum3=ie + l3
sum4=ig + j3
wmin = max0(sum1, sum2, sum3, sum4)
ii = ia + ig
ij = j2 + j3 + l2 + l3
ik = j3 + j1 + l3 + l1
wmax = min0(ii,ij,ik)
omega = 0.d0
do w = wmin, wmax, 2
denom = fact((w-sum1)/2)*fact((w-sum2)/2)*fact((w-sum3)/2)&
*fact((w-sum4)/2)*fact((ii-w)/2)*fact((ij-w)/2)&
*fact((ik-w)/2)
if (mod(w,4) /= 0) denom = -denom
omega = omega + fact(w/2+1) / denom
enddo
theta1 = fact((ia-j3)/2)*fact((j3+ib)/2)*fact((j3-ib)/2)&
/fact(sum1/2+1)
theta2 = fact((ic-l3)/2)*fact((l3+id)/2)*fact((l3-id)/2)&
/fact(sum2/2+1)
theta3 = fact((ie-l3)/2)*fact((l3+iif)/2)*fact((l3-iiF)/2)&
/fact(sum3/2+1)
theta4 = fact((ig-j3)/2)*fact((j3+ih)/2)*fact((j3-ih)/2)&
/fact(sum4/2+1)
theta = theta1 * theta2 * theta3 * theta4
w6js = omega * sqrt(theta)
if (abs(w6js) < 1.d-8) w6js = 0.d0
1000 return
end function w6js

function w9js(j1,j2,j3,j4,j5,j6,j7,j8,j9,fact)
integer :: j1,j2,j3,j4,j5,j6,j7,j8,j9
integer :: i, kmin, kmax, k
real(kind=8) :: x, s, x1, x2, x3, w9js, w6js
real(kind=8),intent(in) :: fact(0:301)

kmin = abs(j1-j9)
kmax = j1 + j9
i = abs(j4-j8)
if (i > kmin) kmin = i
i = j4 + j8
if (i < kmax) kmax = i
i = abs(j2-j6)
if (i > kmin) kmin = i
i = j2 + j6
if (i < kmax) kmax = i
x = 0.d0
do k = kmin, kmax, 2
s = 1.d0
if (mod(k,2) /= 0) s = -1.d0
x1 = w6js(j1,j9,k,j8,j4,j7,fact)
x2 = w6js(j2,j6,k,j4,j8,j5,fact)
x3 = w6js(j1,j9,k,j6,j2,j3,fact)
x = x + s*x1*x2*x3*dfloat(k+1)
enddo
w9js = x
return
end function w9js
```

```
subroutine factrl(fact)
integer :: i
real(kind=8),intent(out) :: fact(0:301)
fact(0) = 1.d0
do i=1,301
fact(I) = fact(I-1) * dble(I)
enddo
END subroutine factrl

Makefile


pot: pot_co-co2.o
gfortran $(CFLAGS) -o pot pot_co-co2

pot_co-co2.o: pot_co-co2.o
gfortran $(CFLAGS) -c pot_co-co2

clean:
rm -f *.o pot
```

2 Program 2

```
program code
use arg
  real(kind=8) :: xval,pi,facdeg
  x1=20.d0
  x2=10.d0
  x3=10.d0
  x4=10.d0
  pi=4.d0*datan(1.d0)
  facdeg=pi/180.d00
  call poten(xval)
                       write(*,*) x1,x2/facdeg,x3/facdeg,x4/facdeg,xval
end program code
subroutine poten(val)
use arg
!  real(kind=dp) , intent(in)   :: x1,x2,x3,x4
  real(kind=dp) , intent(out)   :: val
  integer , allocatable, dimension(:) :: iwork
  real(kind=dp),allocatable, dimension(:) :: work
  integer :: i, j, kk, l, nr, inf
  integer        :: k,key,neval,ier,limit,lenw,last
  real, dimension(2) :: tarray
  real(kind=dp) :: resultat, f, f1, f2, ff, An, aur, convR
  real(kind=dp) :: a,b,eps1,eps2,abserr
  integer :: parite, irest,iprint, imp, output, input, ipot,npas
  logical :: printw
  character(len=40)     :: nomfich
  real(kind=dp)  ::  r(5), theta1(5), theta2(5), phi(5), x(4,5)
  real(kind=dp)  ::  fp(4), fse(4,4), fpk(4), uk(4)
  real(kind=dp)  ::  T(4,4), fsm(4,4), g(4)
  integer  ::  is(4,4)
  real(kind=dp)  ::  pi, facdeg
  call factrl(fact)
! x1=20.d0
! x2=10.d0
!  x3=10.d0
!  x4=10.d0

nomfich='tableau'
!
!write(*,*)nom
      An=6.023d23         ! Avogadro number
      aur = 0.52917706D0  ! 1 au = aur Angstrom
       convR=aur*1.d-8           ! bohr to cm
  pi=4.d0*datan(1.d0)
  facdeg=pi/180.d00
  input=8
  output=9
  imp=output
  ipot=10
       do i=1,4
       do j=1,4
         is(i,j)=0
       enddo
       enddo
       do i=1,4
         is(i,i)=1
         uk(i)=0.d0
       enddo
 100          format(f9.4,f10.4,f10.4,f10.4,f20.9)
!  open (input,file=trim(nomfich)//".dat",status="old")
  open (ipot,file=trim(nomfich)//".pot",status="old")
!  open (output,file=trim(nomfich)//".out",status="unknown")
  read(ipot,*) lamax,nmax
!  write(output,*)'lamax,nmax=   ',lamax, nmax
  lamax=lamax-1
  nmax=nmax-1
  allocate(ri(0:nmax),h(0:nmax))
  allocate(fi(0:nmax,0:lamax),fs(0:nmax,0:lamax))
  allocate(vvl(0:lamax))
  allocate(ll1(0:lamax),ll2(0:lamax),ll(0:lamax))
           do ir=nmax,0,-1
           read(ipot,*) ri(ir)
           enddo

  do l=0,lamax
     read(ipot,*) ll1(l),ll2(l),ll(l)
  irest=((nmax+1)/100)
    do j=1,irest
       READ(ipot,*) fi((j-1)*100:j*100-1,l)
    enddo
    if (irest*100 .NE. nmax+1) then
       READ(ipot,*) fi(irest*100:nmax,l)
    endif
  enddo
do i=0,nmax-1
  h(i)=ri(i+1)-ri(i)
enddo

!  write(output,*)'lamax,nmax=   ',lamax, nmax
!  lamax=lamax-1
!fi=fi*(2.d0*mu)/219474.63067d0
  do l=0,lamax
     call spline1(ri,fi(:,l),fs(:,l),nmax)
  enddo
        nr=56 !251
       nt1=13
       nt2=13
       np=4
       ntintm=91  !251
       nrintm=151
       npintm=91
       iflag2=1
       do i=1,4
       do j=1,4
         is(i,j)=0
       enddo
       enddo
       do i=1,4
         is(i,i)=1
       enddo
!                     write(*,*) 'r     ,  theta1   , theta2   ,  phi?'
!                     read(*,*) x1, x2, x3, x4
                    x2=x2*facdeg
                    x3=x3*facdeg
                    x4=x4*facdeg
                       val=ff()
!                       write(output,*) x1,x2/facdeg,x3/facdeg,x4/facdeg,
   val
!
!********************************************************************
!
end subroutine poten
double precision function ff()
use arg
real(kind=dp) :: resultat,som,An,bk,T,u,aur,cKcm,BE,cmJ,pi,facdeg,y,
   APARITY
integer        :: i,k,l,m,mx
  real(kind=16) ::  w3js, w6js

      An=6.023d23         ! Avogadro number
      bk = 1.380662d-23   ! en J.K-1
      conv = 0.61285d-8   ! from au to cm3.s-1
      u = 1.660565d-27    ! en kg
      aur = 0.52917706D0  ! 1 au = aur Angstrom
      cKcm = 0.69502      ! 1 K = cm-1
      BE = 1.7204         ! en cm-1
      cmJ = 1.9865d-23    ! 1 cm-1 = conve  Joul
       pi = acos(-1.d0)
       expo= 1.d-20             !AÂš to m2
       expo2=1.d6               !m3 to cm3
       convR=aur*1.d-8           ! bohr to cm
       cof1 = expo*dsqrt(8.d0/(redmu*u*pi))
       cof = cof1/(dsqrt(bk)*bk)

  T=303.15d0
  pi=4.d0*datan(1.d0)
  facdeg=pi/180.d00
!  phi=x*facdeg
     call pot(ri,fi(:,:),fs(:,:),h,nmax,lamax,x1,vvl)
  resultat=0.d0
!  write(*,*) 'phi=', phi
  do i=0,lamax
```

```
!  write(*,*) 'lll=', ll1(i),ll2(i),ll(i),x1,x2/facdeg,x3/facdeg,x4/facdeg
     mx=min0(ll1(i),ll2(i))
     som=APARITY(ll1(i))*dble(w3js(2*ll1(i),2*ll2(i),2*ll(i),0,0,0,fact))*
        y(ll1(i),0,x2,0.d0)*y(ll2(i),0,x3,0.d0)
!  write(*,*)'som', som!,y(ll1(i),0,x2,0.d0),y(ll2(i),0,x3,0.d0),w3js(2*
   ll1(i),2*ll2(i),2*ll(i),0,0,0,fact),APARITY(ll1(i))
! write(*,*) 'vvl=', i, som, vvl(i),fact(5)
  do m=1,mx,1
     som=som+2.d0*APARITY(m+ll1(i))*dble(w3js(2*ll1(i),2*ll2(i),2*ll(i),2*
        m,-2*m,0,fact))*y(ll1(i),m,x2,0.d0) &
            *y(ll2(i),m,x3,x4)
  enddo
  resultat=resultat+vvl(i)*som*dfloat(2*ll(i)+1)*dsqrt(1.d0/(4.d0*pi))
! write(*,*) 'som=', i, som, vvl(i)
  enddo
! write(*,*) 'pot=', r,theta1/facdeg,theta2/facdeg,phi/facdeg,resultat

 ff=resultat
! write(*,*) ff
!ff=facdec*facdeg*facdeg
!ff=1.d00
!  write(*,*) r,theta1,theta2,phi
!  write(*,*) 'phi=', phi, ff, resultat
return

end function ff
subroutine spline1(x,f,fs,n)
  implicit real(kind=8) (a-h,o-z)
integer, parameter :: dp = 8
!integer, parameter :: dp = selected_real_kind(15, 307)
real(kind=dp) , allocatable, dimension(:,:) :: a
real(kind=dp) , intent(in), dimension(0:n)   :: x, f
real(kind=dp) , intent(out), dimension(0:n)   :: fs
real(kind=dp) , allocatable, dimension(:)   :: b, h
integer  , intent(in)    :: n
integer        :: i,j,k
real(kind=dp) :: z,xp
allocate(a(0:n,1:3))
allocate(b(0:n))
allocate(h(0:n))
a(:,:)=0.d0
do i=0,n-1
  h(i)=x(i+1)-x(i)
enddo
do i=1,n
  a(i,2)=2.d0*(h(i)+h(i-1))
enddo
a(0,2)=1.d0
a(n,2)=1.d0
do i=0,n-2
  a(i+1,1)=h(i)
enddo
a(n,1)=0.d0
do i=2,n
  a(i-1,3)=h(i-1)
enddo
a(0,3)=0.d0
do i=1,n-1
  b(i)=6.d0*((f(i+1)-f(i))/h(i)-(f(i)-f(i-1))/h(i-1))
```

```fortran
enddo
b(0)=0.d0
b(n)=0.d0
do i=1,n
  a(i,1)=a(i,1)/a(i-1,2)
  a(i,2)=a(i,2)-a(i,1)*a(i-1,3)
enddo
do i=1,n
  b(i)=b(i)-a(i,1)*b(i-1)
enddo
fs(n)=b(n)/a(n,2)
do i=n-1,0,-1
  fs(i)=(b(i)-a(i,3)*fs(i+1))/a(i,2)
enddo
end subroutine spline1
subroutine pot(x,f,fs,h,n,lmax,xp,z)
  implicit real(kind=8) (a-h,o-z)
integer, parameter :: dp = 8
!integer, parameter :: dp = selected_real_kind(15, 307)
real(kind=dp) , intent(in), dimension(0:n)   :: x
real(kind=dp) , intent(in), dimension(0:n,0:lmax)   :: fs, f
real(kind=dp) , intent(in), dimension(0:n)   :: h
real(kind=dp) , intent(in)  :: xp
real(kind=dp)   :: eps=0.0000001d0, Ai,Bi,Ao,Bo
real(kind=dp) , intent(out), dimension(0:lmax) :: z
integer   ,  intent(in)     :: n,lmax
integer        :: i,k,l
k=0
if((xp>=x(0)-eps).and.(xp<=x(n)+eps)) then
  do i=0,n-1
    if (xp>x(i)) k=i
  enddo
else
!  print *,   'pot  :  revoir x !!!!',i,x(i),k
  if(xp>=x(n)+eps) then
do l=0,lmax
   Bo=dlog(f(n-1,l)/f(n,l))/dlog(x(n)/x(n-1))
   Ao=f(n,l)*x(n)**Bo
   z(l)=Ao/(xp**Bo)
enddo
do l=0,lmax
   if(f(n-1,l)*f(n,l)<=0.d0.or.f(n-1,l)/f(n,l)<=1.0d0.or.Bo>8.0d1.or.Bo
      <=4.d0)z(l)=0.d0
enddo
!  z=z/100.d0
!  z=0.d0
!write(2,*)xp,z,Ao,Bo
  else
do l=0,lmax
   Bi=dlog(f(0,l)/f(1,l))/(x(1)-x(0))
   Ai=f(0,l)*dexp(Bi*x(0))
   z(l)=Ai*dexp(-Bi*xp)
enddo
do l=0,lmax
   if(f(1,l)*f(0,l)<=0.d0)z(l)=0.d0
enddo
!write(2,*)z,Ai,Bi
  endif
!  z=0.d0
```

```
  return
!  stop
endif
do l=0,lmax
z(l)=fs(k,l)*((((x(k+1)-xp)**3)/(6.d0*h(k)))-((h(k)/6.d0)*(x(k+1)-xp)))
z(l)=z(l)+fs(k+1,l)*((((xp-x(k))**3)/(6.d0*h(k)))-((h(k)/6.d0)*(xp-x(k))))
z(l)=z(l)+(f(k,l)/h(k))*(x(k+1)-xp)
z(l)=z(l)+(f(k+1,l)/h(k))*(xp-x(k))
enddo
!print *,   'l intervalle est:',k
!print *,   'pi(x)=',z
end subroutine pot
double precision function Y(l,m,theta,phi)
implicit real(kind=8) (a-h,o-z)
integer, parameter :: dp = 8
integer,intent(in) :: l, m
real (kind=8), intent(in) :: theta, phi
integer :: i
real (kind=dp) ::  pi, fac
!pi=8.0d00*datan(1.d00)
Y=PLM(l,iabs(m),dcos(theta))*dcos(dfloat(m)*phi)
!Y=dsqrt(1.d00/(pi))*PLM(l,iabs(m),dcos(theta))*dcos(dfloat(m)*phi)
!Y=APARITY((m+iabs(m))/2)*Y
end function Y
!------------------------------------------------------------------
! This function calculates the 3-j symbol
! J_i and M_i have to be twice the actual value of J and M
!------------------------------------------------------------------
      function w3js(j1,j2,j3,m1,m2,m3,fact)
      integer :: m1, m2, m3, j1, j2, j3
            integer :: ia, ib, ic, id, ie, im, ig, ih, z, zmin, zmax, jsum
            real(kind=16) :: w3js, denom, cc, cc1, cc2
      real(kind=16),intent(in) :: fact(0:301)

            w3js = 0.d0
            if (m1+m2+m3 /= 0) goto 1000
            ia = j1 + j2
            if (j3 > ia) goto 1000
            ib = j1 - j2
            if (j3 < abs(ib)) goto 1000
            jsum = j3 + ia
            ic = j1 - m1
            id = j2 - m2

            if (abs(m1) > j1) goto 1000
                  if (abs(m2) > j2) goto 1000
                  if (abs(m3) > j3) goto 1000
                  ie = j3 - j2 + m1
                  im = j3 - j1 - m2
                  zmin = max0(0,-ie,-im)
                  ig = ia - j3
                  ih = j2 + m2
                  zmax = min0(ig,ih,ic)
                  cc = 0.d0
                  do z = zmin, zmax, 2
                        denom = fact(z/2)*fact((ig-z)/2)*fact((ic-z)/2)*
                           fact((ih-z)/2)*&
                              fact((ie+z)/2)*fact((im+z)/2)
                  if (mod(z,4) /= 0) denom = -denom
```

```
                        cc = cc + 1.d0/denom
                  enddo
                  cc1 = fact(ig/2)*fact((j3+ib)/2)*fact((j3-ib)/2)/fact((
                     jsum+2)/2)
            cc2 = fact((j1+m1)/2)*fact(ic/2)*fact(ih/2)*fact(id/2)*fact((
               j3-m3)/2)*fact((j3+m3)/2)
            cc = cc * sqrt(cc1*cc2)
                  if (mod(ib-m3,4) /= 0) cc = -cc
                  w3js = cc
                  if (abs(w3js) < 1.d-8) w3js = 0.d0
1000             return
      end function w3js

!-----------------------------------------------------------------
! This function calculates the 3-j symbol
! J_i and M_i have to be twice the actual value of J and M
!-----------------------------------------------------------------
            function w6js(j1,j2,j3,l1,l2,l3,fact)
            integer :: j1,j2,j3,l1,l2,l3
            integer :: ia, ib, ic, id, ie, iif, ig, ih, sum1, sum2, sum3,
               sum4
            integer :: w, wmin, wmax, ii, ij, ik
      real(kind=16) :: w6js, omega, denom, theta1, theta2, theta3, theta4,
          theta
      real(kind=16),intent(in) :: fact(0:301)

            w6js = 0.d0
                  ia = j1 + j2
                  if (ia < j3) goto 1000
                  ib = j1 - j2
                  if (abs(ib) > j3) goto 1000
                  ic = j1 + l2
                  if (ic < l3) goto 1000
                  id = j1 - l2
                  if (abs(id) > l3) goto 1000
                  ie = l1 + j2
                  if (ie < l3) goto 1000
                  iif = l1 - j2
                  if (abs(iif) > l3) goto 1000
                  ig = l1 + l2
                  if (ig < j3) goto 1000
                  ih = l1 - l2
                  if (abs(ih) > j3) goto 1000
            sum1=ia + j3
            sum2=ic + l3
            sum3=ie + l3
            sum4=ig + j3
                  wmin = max0(sum1, sum2, sum3, sum4)
                  ii = ia + ig
                  ij = j2 + j3 + l2 + l3
                  ik = j3 + j1 + l3 + l1
                  wmax = min0(ii,ij,ik)
                  omega = 0.d0
                  do w = wmin, wmax, 2
                  denom = fact((w-sum1)/2)*fact((w-sum2)/2)*fact((w-sum3)
                     /2)&
                              *fact((w-sum4)/2)*fact((ii-w)/2)*fact((ij-w)
                                 /2)&
                              *fact((ik-w)/2)
                        if (mod(w,4) /= 0) denom = -denom
                        omega = omega + fact(w/2+1) / denom
                  enddo
            theta1 = fact((ia-j3)/2)*fact((j3+ib)/2)*fact((j3-ib)/2)&
                       /fact(sum1/2+1)
            theta2 = fact((ic-l3)/2)*fact((l3+id)/2)*fact((l3-id)/2)&
                       /fact(sum2/2+1)
            theta3 = fact((ie-l3)/2)*fact((l3+iif)/2)*fact((l3-iiF)/2)&
                       /fact(sum3/2+1)
            theta4 = fact((ig-j3)/2)*fact((j3+ih)/2)*fact((j3-ih)/2)&
                       /fact(sum4/2+1)
            theta = theta1 * theta2 * theta3 * theta4
                  w6js = omega * sqrt(theta)
            if (abs(w6js) < 1.d-8) w6js = 0.d0
1000             return
      end function w6js

      function w9js(j1,j2,j3,j4,j5,j6,j7,j8,j9,fact)
            integer :: j1,j2,j3,j4,j5,j6,j7,j8,j9
            integer :: i, kmin, kmax, k
            real(kind=16) :: x, s, x1, x2, x3, w9js, w6js
      real(kind=16),intent(in) :: fact(0:301)

            kmin = abs(j1-j9)
                  kmax = j1 + j9
                  i = abs(j4-j8)
                  if (i > kmin) kmin = i
                  i = j4 + j8
                  if (i < kmax) kmax = i
                  i = abs(j2-j6)
                  if (i > kmin) kmin = i
                  i = j2 + j6
                  if (i < kmax) kmax = i
                  x = 0.d0
                  do k = kmin, kmax, 2
                        s = 1.d0
                        if (mod(k,2) /= 0) s = -1.d0
                        x1 = w6js(j1,j9,k,j8,j4,j7,fact)
                  x2 = w6js(j2,j6,k,j4,j8,j5,fact)
                  x3 = w6js(j1,j9,k,j6,j2,j3,fact)
                  x = x + s*x1*x2*x3*dfloat(k+1)
                  enddo
            w9js = x
            return
      end function w9js

      subroutine factrl(fact)
            integer :: i
      real(kind=16),intent(out) :: fact(0:301)
            fact(0) = 1.d0
            do i=1,301
                        fact(I) = fact(I-1) * dble(I)
                  enddo
      END subroutine factrl

!
   ...............................................................................
```

```
        double precision FUNCTION APARITY(I)
        IMPLICIT DOUBLE PRECISION (A-H,O-Z)
        APARITY=1.d0
        IF ((I/2)*2.NE.I) APARITY=-1.d0
        RETURN
        END
!
   ................................................................................

!****************************************************************
       double precision FUNCTION PLM(LIN,MINa,COSTH)
        IMPLICIT real*8 (a-h,o-z)
        if (ABS(COSTH).GT.1.0d00)THEN
        WRITE(6,*)'***ILLEGAL␣ARGEMENT␣TO␣PLM␣X=',COSTH
        STOP
        ENDIF
        L=LIN
        M=IABS(MINa)
        X=COSTH
        IF(M.GT.L)THEN
        PLM=0.0d00
        ENDIF
        LMAX=L
        IF (M.GT.0) GO TO 5
        PLM=1.0d0
        PM2=0.d0
        XL=0.d0
        DO 2 L=1,LMAX
        XL=XL+1.0d0
        PP=((2.d0*XL-1.d0)*X*PLM-(XL-1.d0)*PM2)/XL
        PM2=PLM
2       PLM=PP
        GO TO 9000
5       IMAX=2*M
        RAT=1.d0
        AI=0.d0
        DO 6 I=2,IMAX,2
        AI=AI+2.d0
6       RAT=RAT*((AI-1)/AI)
        Y=SQRT(1.0d0-X*X)
        PLM=SQRT(RAT)*(Y**M)
        PM2=0.d0
        LOW=M+1
        XL=LOW-1
        DO 10 L=LOW,LMAX
        XL=XL+1.0d0
        AL=DBLE((L+1)*(L-1))
        AL=1.d0/AL
        AL2=DBLE((L+M-1)*(L-M-1))*AL
        AL=SQRT(AL)
        AL2=SQRT(AL2)
        PP=(2.d0*XL-1.D0)*X*PLM*AL-PM2*AL2
        PM2=PLM
10      PLM=PP
        PLM=PLM*AAPARITY(MINa)
!9000     PLM=PLM*SQRT(XL+0.5d0)
9000   PLM=PLM
        RETURN
        END
```

```
!
    ................................................................................

        double precision FUNCTION AAPARITY(I)
        IMPLICIT DOUBLE PRECISION (A-H,O-Z)
        AAPARITY=1.d0
        IF ((I/2)*2.NE.I) AAPARITY=-1.d0
        RETURN
        END
!
    ................................................................................

MODULE arg
  implicit real(kind=8) (a-h,o-z)
  integer, parameter :: dp = 8
  real(kind=dp), save:: x1, x2, x3, x4
  integer, save :: nmax, lamax
!implicit none
!  integer , dimension(0:lamax), save :: ll1, ll2, ll
!  real(kind=dp), dimension(0:nmax), save :: ri,h
!  real(kind=dp), dimension(0:lamax), save :: vvl
!  real(kind=dp), dimension(0:nmax,0:lamax), save :: fi,fs
  integer , dimension(:), allocatable, save :: ll1, ll2, ll
  real(kind=dp), dimension(:), allocatable, save :: ri,h
  real(kind=dp), dimension(:), allocatable, save :: vvl
  real(kind=dp), dimension(:,:), allocatable, save :: fi,fs
  real(kind=16), save :: fact(0:301)
  real(kind=dp), save :: yyR(5,5,5,5)
END MODULE arg
```