

```

C*****
C   ANISOTROPIC NETWORK MODEL (ANM) PROGRAM
C*****
C   VERSION 4 9/27/05
C   WRITTEN/ARRANGED BY TANER Z SEN
C
C   PLEASE REFER TO THE FOLLOWING PAPERS FOR MORE INFO:
C
C   ANM:
C   Atilgan AR, Durell SR, Jernigan RL, Demirel MC, Keskin O,
C   Bahar I, Biophys. J., 80:505-15, 2001.
C
C   GNM:
C   Bahar I, Atilgan AR, Erman B, Fold. & Des., 2:173-81, 1997
C
C*****
C   VARIABLES
C*****

C   NR: NUMBER OF RESIDUES
C   CUTOFF: CUT-OFF RADIUS
C   EIGENCUT: CUT-OFF TO DECIDE ZERO EIGENVALUE(S)

PARAMETER (NR=104,GAMMA=1.0)
REAL X(NR),Y(NR),Z(NR)
CHARACTER CNAM(NR)*3
REAL BETA(NR),HBETA(NR)
REAL HESS(NR*3,NR*3)
REAL INVHESS(NR*3,NR*3)
REAL W(NR*3),V(NR*3,NR*3)
DIMENSION INDX(NR*3)
REAL INVCONT(NR,NR)
INTEGER RESNUM,RES3,ANUM(NR),RNUM(NR)
INTEGER NUMZERO,STARTMOD,ENDMOD
REAL CUTOFFSQ,CUTOFF
REAL BX,BY,BZ,DIS2,EIGENCUT
REAL FLUCX2(NR),FLUCY2(NR),FLUCZ2(NR)
REAL FLUCX(NR),FLUCY(NR),FLUCZ(NR)
REAL FX2N(NR),FY2N(NR),FZ2N(NR)
CHARACTER*4 ATNAME,ANAME(NR)
CHARACTER*1 A1,CHA(NR)
REAL MEANB,MEANF,CORR,CORRF2,CORRB2

C   DUMMIES

INTEGER DINT,DINT2,ICA
REAL DIFXX,DIFYY,DIFFZZ,DIST,DSUM,DSUM1,DSUM2,IDUM
CHARACTER DUMMY6*6,DUMMY3*3

C*****
C   PARAMETERS
C*****

RESNUM=NR
RES3=NR*3

C   This is the only parameter we are using:
C   radius threshold to decide if two residues/nucleotides are
C   connected
C   CUTOFF=13

```

```
CUTOFFSQ=CUTOFF*CUTOFF
EIGENCUT=1E-5
```

```
C*****
```

```
C   FILES
```

```
C*****
```

```
C   THIS IS THE ONLY INPUT FILE
```

```
C   this file can be obtained from Brookhaven Protein Data Bank
```

```
C   http://www.rcsb.org/pdb
```

```
OPEN(50,FILE='lhrc.pdb')
```

```
OPEN(60,FILE='centers.txt')
```

```
OPEN(61,FILE='beta.txt')
```

```
OPEN(62,FILE='flucxyz.txt')
```

```
OPEN(66,FILE='eigenvalues.txt')
```

```
C*****
```

```
C   READ ALPHA CARBONS COORDINATES, AND B-FACTORS
```

```
C*****
```

```
310 READ(50,'(A6)') DUMMY6
    IF(DUMMY6.NE.'ATOM  ') GOTO 310
    BACKSPACE(50)
```

```
ICA=1
```

```
320 READ(50,'(A6)') DUMMY6
    IF(DUMMY6.NE.'ATOM  ') THEN
      IF(DUMMY6.EQ.'END  ') THEN
        GOTO 330
      ELSE
        GOTO 320
      ENDIF
    ENDIF
    BACKSPACE(50)
    READ(50,55) DUMMY6,D1INT,ATNAME,DUMMY3,A1,D2INT,XXX,YYY,ZZZ,R1,BBB
```

```
IF(ATNAME.EQ.' CA ') THEN
```

```
ANUM(ICA)=D1INT
```

```
ANAME(ICA)=ATNAME
```

```
CNAM(ICA)=DUMMY3
```

```
CHA(ICA)=A1
```

```
RNUM(ICA)=D2INT
```

```
X(ICA)=XXX
```

```
Y(ICA)=YYY
```

```
Z(ICA)=ZZZ
```

```
BETA(ICA)=BBB
```

```
ICA=ICA+1
```

```
ENDIF
```

```
GOTO 320
```

```
55  FORMAT(A6,I5,1X,A4,1X,A3,1X,A1,I4,4X,3F8.3,2F6.2)
```

```
330 IF(RESNUM.NE.(ICA-1)) THEN
    WRITE(*,*) 'THERE IS A PROBLEM WITH THE NUMBER OF RESIDUES!'
    WRITE(*,*) 'GIVEN RESNUM=',RESNUM,'CALCULATED RESNUM=',ICA-1
    GOTO 666
  ENDIF
```

```

DO 10 I=1,RESNUM
10 WRITE(60,200) I,X(I),Y(I),Z(I),BETA(I)

200 FORMAT(I3,3F8.3,F7.3)

C*****
C  INITIALIZATION OF HESSIAN MATRIX
C*****

DO 20 I=1,RES3
DO 20 J=1,RES3
20 HESS(I,J)=0.

C*****
C  CREATION OF HESSIAN MATRIX
C*****

DO J=1,RESNUM
DO K=1,RESNUM
BX=X(J)-X(K)
BY=Y(J)-Y(K)
BZ=Z(J)-Z(K)
DIS2=BX*BX+BY*BY+BZ*BZ

IF(J.NE.K.AND.DIS2.LE.CUTOFFSQ) THEN
C  FIRST: CREATION OF Hii
HESS(3*J-2,3*J-2)=HESS(3*J-2,3*J-2)+GAMMA*BX*BX/DIS2
HESS(3*J-1,3*J-1)=HESS(3*J-1,3*J-1)+GAMMA*BY*BY/DIS2
HESS(3*J,3*J)=HESS(3*J,3*J) +GAMMA*BZ*BZ/DIS2

HESS(3*J-2,3*J-1)=HESS(3*J-2,3*J-1)+GAMMA*BX*BY/DIS2
HESS(3*J-2,3*J)=HESS(3*J-2,3*J) +GAMMA*BX*BZ/DIS2
HESS(3*J-1,3*J-2)=HESS(3*J-1,3*J-2)+GAMMA*BY*BX/DIS2
HESS(3*J-1,3*J)=HESS(3*J-1,3*J) +GAMMA*BY*BZ/DIS2
HESS(3*J,3*J-2)=HESS(3*J,3*J-2) +GAMMA*BX*BZ/DIS2
HESS(3*J,3*J-1)=HESS(3*J,3*J-1) +GAMMA*BY*BZ/DIS2

C  SECOND: CREATION OF Hij
HESS(3*J-2,3*K-2)= -GAMMA*BX*BX/DIS2
HESS(3*J-1,3*K-1)= -GAMMA*BY*BY/DIS2
HESS(3*J,3*K)= -GAMMA*BZ*BZ/DIS2

HESS(3*J-2,3*K-1)= -GAMMA*BX*BY/DIS2
HESS(3*J-2,3*K)= -GAMMA*BX*BZ/DIS2
HESS(3*J-1,3*K-2)= -GAMMA*BY*BX/DIS2
HESS(3*J-1,3*K)= -GAMMA*BY*BZ/DIS2
HESS(3*J,3*K-2)= -GAMMA*BX*BZ/DIS2
HESS(3*J,3*K-1)= -GAMMA*BY*BZ/DIS2
ENDIF
ENDDO
ENDDO

C*****
C  SINGULAR VALUE DECOMPOSITION TO GET RID OF ZERO EIGENVALUES
C*****

CALL SVDCMP(HESS,RES3,RES3,RES3,RES3,W,V)

C*****
C  PUTTING THE EIGENVALUES IN ASCENDING ORDER

```

```

C*****
      CALL INDEXX(RES3,W,INDX)

      DO 700 I=1,RES3
        WRITE(66,*) I,W(INDX(I))
700    CONTINUE

C*****
C      SAVING THE EIGENVECTORS
C*****

      NEIG=RES3
      LN=RES3

      OPEN(44,FILE='eigenvectors.TXT')
      WRITE(44,*) NEIG,LN
      DO I=1,NEIG
        WRITE(44,*) W(INDX(I))
      ENDDO
C      THE FIRST SET OF DATA IS: J=1 AND THEN I=1,LN (V(1,1) V(2,1) etc.
      WRITE(44,*) ((V(I,INDX(j)), I=1,LN),J=1,NEIG)

C*****
C      CALCULATING ZERO EIGENVALUES
C*****
      NUMZERO=0
      DO K=1,RES3
        IF(W(K).LE.EIGENCUT) THEN
          NUMZERO=NUMZERO+1
        ENDIF
      ENDDO

C      NUMZERO should be equal to 6 due the our connectivity definition
      WRITE(*,*) 'ZERO EIGENVALUES=',NUMZERO

C*****
C      CALCULATING INVERSE CONNECTIVITY  INVCONT
C*****

      DO I=1,RES3
        DO J=1,RES3
          INVHESS(I,J)=0.
          DO K=1,RES3
            IF(W(K).GT.EIGENCUT) THEN
              INVHESS(I,J)=INVHESS(I,J)+V(I,K)*V(J,K)/W(K)
            ENDIF
          ENDDO
        ENDDO
      ENDDO

      JJJ=0
      DO I=1,RES3
        IF(INVHESS(I,I).LT.0) JJJ=JJJ+1
      ENDDO
      WRITE(*,*) 'NEGATIVE ELEMENTS IN THE DIAGONAL:',JJJ

```

```

C*****
C      CALCULATION OF MEAN-SQUARE FLUCTUATIONS
C*****

      DO I=1,RESNUM
        FLUCX2(I)=INVHESS((I-1)*3+1,(I-1)*3+1)
        FLUCY2(I)=INVHESS((I-1)*3+2,(I-1)*3+2)
        FLUCZ2(I)=INVHESS((I-1)*3+3,(I-1)*3+3)
        HBETA(I)=FLUCX2(I)+FLUCY2(I)+FLUCZ2(I)
      ENDDO

C*****
C      NORMALIZATION
C*****

      DSUM1=0.
      DSUM2=0.
      DO 65 I=1,RESNUM
        DSUM1=DSUM1+HBETA(I)
        DSUM2=DSUM2+BETA(I)
65     CONTINUE

      WRITE(*,*) 'DSUM1=',DSUM1,'DSUM2=',DSUM2

      MEANB=0.
      MEANF=0.
      DO 70 I=1,RESNUM
        HBETA(I)=HBETA(I)*DSUM2/DSUM1
        FLUCX2(I)=FLUCX2(I)*DSUM2/DSUM1
        FLUCY2(I)=FLUCY2(I)*DSUM2/DSUM1
        FLUCZ2(I)=FLUCZ2(I)*DSUM2/DSUM1

C      NORMALIZATION OF FLUCTUATIONS

      FX2N(I)=FLUCX2(I)-HBETA(I)/3
      FY2N(I)=FLUCY2(I)-HBETA(I)/3
      FZ2N(I)=FLUCZ2(I)-HBETA(I)/3

      WRITE(62,220) I,FLUCX2(I),FLUCY2(I)
+           ,FLUCZ2(I),I,FX2N(I),FY2N(I),FZ2N(I)

C      COMPARING MEAN-SQUARE FLUCTUATIONS WITH B-FACTORS

      WRITE(61,210) I,HBETA(I),BETA(I)
      MEANF=MEANF+HBETA(I)
      MEANB=MEANB+BETA(I)

70     CONTINUE

      MEANF=MEANF/RESNUM
      MEANB=MEANB/RESNUM

210    FORMAT(I3,2F7.2)
220    FORMAT(I3,1X,3F7.2,1X,I3,1X,3F7.2)

C*****
C      CALCULATING THE CORRELATIONS
C*****

```

```

CORR=0.
CORRF2=0.
CORRB2=0.
DO I=1,RESNUM
  CORR=CORR+(HBETA(I)-MEANF)*(BETA(I)-MEANB)
  CORRF2=CORRF2+(HBETA(I)-MEANF)*(HBETA(I)-MEANF)
  CORRB2=CORRB2+(BETA(I)-MEANB)*(BETA(I)-MEANB)
ENDDO
CORR=CORR/SQRT(CORRF2*CORRB2)
WRITE(*,*) 'Correlation: ',CORR

WRITE(*,*) 'Program finished successfully!'

666  STOP
      END

C*****
C  ALL OF THE FOLLOWING FUNCTIONS ARE TAKEN FROM
C  PRESS, W.H. ET AL. "NUMERICAL RECIPES IN FORTRAN 77",
C  CAMBRIDGE UNIVERSITY PRESS, 2001
C*****
C
C  SINGULAR VALUE DECOMPOSITION
C
C*****

SUBROUTINE SVDCMP(a,m,n,mp,np,w,v)
INTEGER m,mp,n,np,NMAX
REAL a(mp,np),v(np,np),w(np)
PARAMETER (NMAX=70000)
C  USES pythag
INTEGER i,its,j,jj,k,l,nm
REAL anorm,c,f,g,h,s,scale,x,y,z,rv1(NMAX),pythag
g=0.0
scale=0.0
anorm=0.0
do 25 i=1,n
  l=i+1
  rv1(i)=scale*g
  g=0.0
  s=0.0
  scale=0.0
  if(i.le.m)then
    do 11 k=i,m
      scale=scale+abs(a(k,i))
11    continue
    if(scale.ne.0.0)then
      do 12 k=i,m
        a(k,i)=a(k,i)/scale
        s=s+a(k,i)*a(k,i)
12    continue
      f=a(i,i)
      g=-sign(sqrt(s),f)
      h=f*g-s
      a(i,i)=f-g
      do 15 j=1,n
        s=0.0
        do 13 k=i,m

```

```

        s=s+a(k,i)*a(k,j)
13      continue
        f=s/h
        do 14 k=i,m
            a(k,j)=a(k,j)+f*a(k,i)
14      continue
15      continue
        do 16 k=i,m
            a(k,i)=scale*a(k,i)
16      continue
        endif
    endif
    w(i)=scale *g
    g=0.0
    s=0.0
    scale=0.0
    if((i.le.m).and.(i.ne.n))then
        do 17 k=1,n
            scale=scale+abs(a(i,k))
17      continue
        if(scale.ne.0.0)then
            do 18 k=1,n
                a(i,k)=a(i,k)/scale
                s=s+a(i,k)*a(i,k)
18      continue
            f=a(i,1)
            g=-sign(sqrt(s),f)
            h=f*g-s
            a(i,1)=f-g
            do 19 k=1,n
                rv1(k)=a(i,k)/h
19      continue
            do 23 j=1,m
                s=0.0
                do 21 k=1,n
                    s=s+a(j,k)*a(i,k)
21      continue
                do 22 k=1,n
                    a(j,k)=a(j,k)+s*rv1(k)
22      continue
23      continue
                do 24 k=1,n
                    a(i,k)=scale*a(i,k)
24      continue
            endif
        endif
    endif
    anorm=max(anorm,(abs(w(i))+abs(rv1(i))))
25 continue
    do 32 i=n,1,-1
        if(i.lt.n)then
            if(g.ne.0.0)then
                do 26 j=1,n
                    v(j,i)=(a(i,j)/a(i,1))/g
26      continue
                do 29 j=1,n
                    s=0.0
                    do 27 k=1,n
                        s=s+a(i,k)*v(k,j)
27      continue
                do 28 k=1,n

```

```

                v(k,j)=v(k,j)+s*v(k,i)
28         continue
29         continue
        endif
        do 31 j=1,n
            v(i,j)=0.0
            v(j,i)=0.0
31         continue
        endif
        v(i,i)=1.0
        g=rv1(i)
        l=i
32        continue
do 39 i=min(m,n),1,-1
    l=i+1
    g=w(i)
    do 33 j=1,n
        a(i,j)=0.0
33        continue
        if(g.ne.0.0)then
            g=1.0/g
            do 36 j=1,n
                s=0.0
                do 34 k=1,m
                    s=s+a(k,i)*a(k,j)
34                continue
                    f=(s/a(i,i))*g
                    do 35 k=i,m
                        a(k,j)=a(k,j)+f*a(k,i)
35                    continue
36                continue
                do 37 j=i,m
                    a(j,i)=a(j,i)*g
37                continue
            else
                do 38 j= i,m
                    a(j,i)=0.0
38                continue
            endif
            a(i,i)=a(i,i)+1.0
39        continue
do 49 k=n,1,-1
    do 48 its=1,30
        do 41 l=k,1,-1
            nm=l-1
            if((abs(rv1(l))+anorm).eq.anorm) goto 2
            if((abs(w(nm))+anorm).eq.anorm) goto 1
41        continue
1        c=0.0
        s=1.0
        do 43 i=1,k
            f=s*rv1(i)
            rv1(i)=c*rv1(i)
            if((abs(f)+anorm).eq.anorm) goto 2
            g=w(i)
            h=pythag(f,g)
            w(i)=h
            h=1.0/h
            c= (g*h)
            s=-(f*h)

```

```

do 42 j=1,m
  y=a(j,nm)
  z=a(j,i)
  a(j,nm)=(y*c)+(z*s)
  a(j,i)=-(y*s)+(z*c)
42  continue
43  continue
2  z=w(k)
  if(1.eq.k)then
    if(z.lt.0.0)then
      w(k)=-z
      do 44 j=1,n
        v(j,k)=-v(j,k)
44  continue
      endif
      goto 3
    endif
  if(its.eq.30) pause 'no convergence in svdcmp'
  x=w(1)
  nm=k-1
  y=w(nm)
  g=rv1(nm)
  h=rv1(k)
  f=((y-z)*(y+z)+(g-h)*(g+h))/(2.0*h*y)
  g=pythag(f,1.0)
  f=((x-z)*(x+z)+h*((y/(f+sign(g,f)))-h))/x
  c=1.0
  s=1.0
do 47 j=1,nm
  i=j+1
  g=rv1(i)
  y=w(i)
  h=s*g
  g=c*g
  z=pythag(f,h)
  rv1(j)=z
  c=f/z
  s=h/z
  f=(x*c)+(g*s)
  g=-(x*s)+(g*c)
  h=y*s
  y=y*c
do 45 jj=1,n
  x=v(jj,j)
  z=v(jj,i)
  v(jj,j)=(x*c)+(z*s)
  v(jj,i)=-(x*s)+(z*c)
45  continue
  z=pythag(f,h)
  w(j)=z
  if(z.ne.0.0)then
    z=1.0/z
    c=f*z
    s=h*z
  endif
  f=(c*g)+(s*y)
  x=-(s*g)+(c*y)
do 46 jj=1,m
  y=a(jj,j)
  z=a(jj,i)

```

```

        a(jj,j)= (y*c)+(z*s)
        a(jj,i)=- (y*s)+(z*c)
46     continue
47     continue
        rv1(1)=0.0
        rv1(k)=f
        w(k)=x
48     continue
3     continue
49     continue
return
END

FUNCTION pythag(a,b)
REAL a,b,pythag
REAL absa,absb
absa=abs(a)
absb=abs(b)
if(absa.gt.absb)then
    pythag=absa*sqrt(1.+(absb/absa)**2)
else
    if(absb.eq.0.)then
        pythag=0.
    else
        pythag=absb*sqrt(1.+(absa/absb)**2)
    endif
endif
return
END

SUBROUTINE INDEXX(N,ARRIN,INDX)
DIMENSION ARRIN(N),INDX(N)
DO 11 J=1,N
    INDX(J)=J
11 CONTINUE
IF(N.EQ.1)RETURN
L=N/2+1
IR=N
10 CONTINUE
    IF(L.GT.1)THEN
        L=L-1
        INDXT=INDX(L)
        Q=ARRIN(INDXT)
    ELSE
        INDXT=INDX(IR)
        Q=ARRIN(INDXT)
        INDX(IR)=INDX(1)
        IR=IR-1
        IF(IR.EQ.1)THEN
            INDX(1)=INDXT
            RETURN
        ENDIF
    ENDIF
    I=L
    J=L+L
20 IF(J.LE.IR)THEN
    IF(J.LT.IR)THEN
        IF(ARRIN(INDX(J)).LT.ARRIN(INDX(J+1)))J=J+1
    ENDIF

```

```
IF(Q.LT.ARRIN(INDX(J)))THEN
  INDX(I)=INDX(J)
  I=J
  J=J+J
ELSE
  J=IR+1
ENDIF
GO TO 20
ENDIF
INDX(I)=INDXT
GO TO 10
END
```