

```

C*****
C   GAUSSIAN NETWORK MODEL (GNM) PROGRAM
C*****
C
C   WRITTEN AND ARRANGED BY TANER Z SEN 2003
C   THE MATERIAL AND ASSISTANCE PROVIDED BY
C   ROBERT L. JERNIGAN, ANDRZEJ KLOCZKOWSKI
C   IVET BAHAR, ALPAY TEMIZ
C
C*****
C   VARIABLES
C*****

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

PARAMETER (NR=175)
REAL XX(NR), YY(NR), ZZ(NR), XXX, YYY, ZZZ
REAL BETA(NR), BBB
REAL CONT(NR, NR)
REAL W(NR), V(NR, NR)
DIMENSION INDX(NR)
REAL INVCNT(NR, NR), CROSS(NR, NR)
INTEGER RESNUM, NZERO
INTEGER CUTOFFSQ
INTEGER MODSTART, MODEND, MODSTART_I, MODEND_I
REAL MSF(NR)
CHARACTER ATNAME*4

C   DUMMIES

INTEGER DINT, DINT2, ICA
REAL DIFXX, DIFY, DIFFZZ, DIST, DSUM, DSUM1, DSUM2, DSUM3, DSUMW, IDUM
CHARACTER DUMMY6*6, DUMMY3*3

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

RESNUM=NR
CUTOFFSQ=49
EIGENCUT=1E-5

C*****
C   FILES
C*****

C   THIS IS THE ONLY INPUT FILE
OPEN(50, FILE='1AQB.PDB')

OPEN(60, FILE='CENTERS.GNM')
OPEN(61, FILE='BETA_ALL_MODES.TXT')
OPEN(62, FILE='CONTACTS.TXT')
OPEN(63, FILE='CROSSCORR_ALL_MODES.TXT')
OPEN(64, FILE='MS_FLUC.TXT')
OPEN(65, FILE='CROSS_CERTAIN_MODES.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 ') GOTO 330
    BACKSPACE(50)
    READ(50,55) DUMMY6,DINT,ATNAME,DUMMY3,DINT,XXX,YYY,ZZZ,DINT,BBB

    IF(ATNAME.EQ.' CA ') THEN
        XX(ICA)=XXX
        YY(ICA)=YYY
        ZZ(ICA)=ZZZ
        BETA(ICA)=BBB
        ICA=ICA+1
    ENDIF
    GOTO 320

55  FORMAT(A6,1X,I4,1X,A4,1X,A3,2X,I4,5X,3F8.3,3X,I3,F6.2)

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
    ENDIF

    DO 10 I=1,RESNUM
10  WRITE(60,200) I,XX(I),YY(I),ZZ(I),BETA(I)
200 FORMAT(I3,3F8.3,F7.3)

C*****
C  INITIALIZATION OF CONTACT MATRIX
C*****

    DO 20 I=1,RESNUM
    DO 20 J=1,RESNUM
20  CONT(I,J)=0.

C*****
C  CALCULATION OF BONDED AND NONBONDED INTERACTIONS
C*****

    DO 30 I=1,RESNUM-1
    DO 30 J=I+1,RESNUM
        DIFXX=XX(I)-XX(J)
        DIFYY=YY(I)-YY(J)
        DIFZZ=ZZ(I)-ZZ(J)
        DIST=DIFXX*DIFXX+DIFYY*DIFYY+DIFZZ*DIFZZ

        IF(DIST.LE.CUTOFFSQ) THEN
            CONT(I,J)=-1.
            CONT(J,I)=-1.
        ENDIF

30  CONTINUE

C*****
C  CONSTRUCTING THE DIAGONAL ELEMENTS OF CONT MATRIX
C*****

    DO 40 I=1,RESNUM
        DSUM=0.
        DO 35 J=1,RESNUM
            DSUM=DSUM+CONT(I,J)
35  CONTINUE
        CONT(I,I)=-DSUM

40  CONTINUE

C*****

```

```

C      SAVING CONTACT MATRIX
C*****

      DO 800 I=1,RESNUM
      DO 800 J=1,RESNUM
        WRITE (62,190) I,J,CONT(I,J)
800    CONTINUE
190    FORMAT(2I5,F8.3)

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

      CALL SVDCMP (CONT,RESNUM,RESNUM,RESNUM,RESNUM,W,V)

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

      CALL INDEXX (RESNUM,W,INDX)

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

C*****
C      CALCULATING INVERSE CONNECTIVITY WITH ALL MODES
C*****

      DO 60 I=1,RESNUM
      DO 60 J=1,RESNUM
        INVCONT(I,J)=0.
C      K SPECIFIES WHICH MODES WE ARE USING
      DO 60 K=1,RESNUM
        IF(W(INDX(K)).GT.EIGENCUT) THEN
          INVCONT(I,J)=INVCONT(I,J)+CONT(I,INDX(K))*V(J,INDX(K))
+
          /W(INDX(K))
        ENDIF
60    CONTINUE

      DO I=1,RESNUM
      DO J=1,RESNUM
        CROSS(I,J)=INVCONT(I,J)/SQRT(INVCONT(I,I)*INVCONT(J,J))
        WRITE (63,190) I,J,CROSS(I,J)
      ENDDO
      ENDDO

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

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

C*****
C      COMPARE MEAN-SQUARE FLUCTUATIONS WITH B-FACTORS
C*****

      DO 70 I=1,RESNUM
        WRITE (61,210) I,INVCONT(I,I)*DSUM2/DSUM1,BETA(I)
70    CONTINUE

210   FORMAT(I3,2F7.2)

```



```

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=700)
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
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(l.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(l)
        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
45      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)
        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)
46      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)
47      continue
48      rv1(l)=0.0
49      rv1(k)=f
        w(k)=x
50      continue
51      3 continue
52      return
53      END

```

```

C*****
C      FUNCTION pythag(a,b)
C*****
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

```

```

C*****
C      SUBROUTINE INDEXX(N,ARRIN,INDX)
C*****
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
```