```fortran
C     =================================================================
C     SUBROUTINE PDD_SEN       :       perform PDD expansion, evaluate design sensitivity
c
C
C
C       N = 21
C       Y0 = total mass of design
C     Y1 = Nmin - Nc
c     Nmin           :   the minumum fatigue crack-initiation life
C     Nc             :   design threshold of loading cycles
c
C       X(1:21) -> TRUNCATED GAUSSIAN DISTRIBUTION
C
C
C       NOTE:  ALL CALCULATIONS ARE DONE IN THE GAUSSIAN SPACE
C     =================================================================
c     Copyright (c) 2010-2014 [Xuchun Ren], All Rights Reserved.



      SUBROUTINE PDD_SEN (XMU_IN,XMU_IN2, XSIG_IN, NSAMP,NBAS,
     1   NBAS_SM, NGAUSS,
     1 NGAUSS_SM, IID,  NGAUSS_GQ, IFLAG_Y, IFLAG_MCS, MOMSEN_OUT,
     3   IFLAG_AIJK, NVAR)

        USE RNNOF_INT
        USE RNSET_INT
        USE GQRUL_INT
        USE UMACH_INT
        USE GAMMA_INT

        USE DFLIB
        INCLUDE "link_fnl_shared.h"
CC     SUBROUTINE TO EVALUATE THE OBJECTIVE FUNCTION AND CONSTRAINTS
C     FOR EXAMPLE 4.1 OF PAPER "ROBUST DESIGN WITH ARBITRARY DISTRIBUTIONS USING GAUSS-TYPE QUADRATURE FORMULA" BY S.H. LEE, W. CHEN, ETC.
C       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
        IMPLICIT REAL*8 (A-H,O-Z)
        DOUBLE PRECISION XMU_IN(200),XMU_IN2(200), XSIG_IN(200),
        1      MOMSEN_OUT(30,2)
        REAL*8 IND
        REAL*8 X(30),XMU(30),SIG(30),XX(50),QX(50),QW(50),QXFIX(2),
        1          NMU(30), NSIG(30), SYUI(30), TMPA1(30), TMPA2(30),
     2          SMU_A(200),SSIG_A(200),SMU_A_TMP(200),SSIG_A_TMP(200)
      REAL*8 XX_SM(50),QX_SM(50),QW_SM(50),QXFIX_SM(2),X1(30)
        REAL*8 YR1(:,:),YR2(:,:,:,:),YR3(:,:,:,:,:,:),H(:,:,:)
        REAL*8 SMR1(:,:),SMR2(:,:,:,:),SMR3(:,:,:,:,:,:),SMH(:,:,:),
     1      SGR1(:,:),SGR2(:,:,:,:),SGR3(:,:,:,:,:,:),SGH(:,:,:)
        REAL*8 ALFA(:,:),PSIF(:,:),BITA(:,:,:,:),GAMA(:,:,:,:,:,:)
        REAL*8 ALFA_SM(:,:),PSIF_SM(:,:),BITA_SM(:,:,:,:),
        1          GAMA_SM(:,:,:,:,:,:),
     2      ALFA_SG(:,:),PSIF_SG(:,:),BITA_SG(:,:,:,:),
        1          GAMA_SG(:,:,:,:,:,:), PSIF_N(:,:),
     3      ALFA_SMIID(:,:),ALFA_SGIID(:,:),SCMU(:,:,:),SCSIG(:,:,:),
     4      YPDD(:,:,:), H_GQ(:,:,:)
      REAL*8 QXN(30,50),QWN(30,50),XXN(30,50),RDD_MU(30), XXN_SM(30,50),
     1      QWN_SM(30,50),QXN_SM(30,50)
      INTEGER NGAUSS(30), NBAS(30), NGAUSS_MAX, NBAS_MAX, NGAUSS_SM(30),
     1      NBAS_SM(30), NGAUSS_SM_MAX, NBAS_SM_MAX, I_TEMP


      ALLOCATABLE YR1,YR2,YR3,H,ALFA,PSIF,BITA,GAMA,SMR1,SMR2,SMR3,SMH,
     1              ALFA_SM,PSIF_SM,BITA_SM,GAMA_SM,
     2      SGR1,SGR2,SGR3,SGH,PSIF_N,
     3              ALFA_SG,PSIF_SG,BITA_SG,GAMA_SG,
     4          ALFA_SGIID, ALFA_SMIID,SCMU,SCSIG,YPDD,H_GQ

        REAL*8 A(30,30),B(30,30),AIJK(25,25,5,5),
        1      AIJKC(2,25,25,5,5)
        INTEGER NBAS_A(30), NO_GAUSS
      REAL*8 AF1(25,5), BT1(25,25,5,5), GM1(25,25,25,5,5,5),
     1  ALFA_OUT(25,5), X_TEMP(30),AIJK_TEMP(25,25,5,5)





C      INTEGER N
        COMMON /SR02/ XMU,SIG, NMU, NSIG, NYQ, N
        COMMON /XR01/ N_F, N_GF, N_GC, N_EF,N_EC, N_EF1, N_EC1, N_EF2, N_EC2
      COMMON /XR02/ AIJKC
      COMMON /XR06/ NO_GAUSS




        CHARACTER(80) MSG,MSG1
        MSG = 'PROGRAM NSAMP NBAS NBAS_SM NGAUSS NGAUSS_SM FN IID NYQ
     1         NGAUSS_GQ'
        MSG1 = 'IID = 0(INDEPENDENT), 1(IID), 2(DEPENDNENT) '

C
       RDD_MU = 0.


       NO_GAUSS = 0



      X1 = 0.
      NBAS_A = 0.
      NBAS_A(1:N) = NBAS(1:N)
C       N = 2
        PI = 4.*DATAN(1.0D0)
        DO I = 1,N
        XMU(I) = 0.
        SIG(I) = 1.
        ENDDO

      DO I = 1,N
            NMU(I) = XMU_IN2(I)
              NSIG(I) = XSIG_IN(I)
        ENDDO

        NGAUSS_MAX=MAXVAL(NGAUSS)
        NBAS_MAX=MAXVAL(NBAS)
        NGAUSS_SM_MAX=MAXVAL(NGAUSS_SM)
        NBAS_SM_MAX=MAXVAL(NBAS_SM)
```

```
      I_TEMP = 1
      DO I = 1, N
        I_TEMP = I_TEMP*NGAUSS(I)* NGAUSS_SM(I)
        IF (I_TEMP .NE. 0) I_TEMP = 1
      ENDDO

        IF (I_TEMP .EQ. 0) THEN
         PRINT*, "NGAUSS =0 OR NGAUSS_SM = 0"
         STOP
      ENDIF

        CALL DGQRUL (NGAUSS(1),4,0.,0.,0,QXFIX,QX,QW)
        CALL DGQRUL (NGAUSS_SM(1),4,0.,0.,0,QXFIX_SM,QX_SM,QW_SM)
        DO J = 1, N
      DO I = 1,NGAUSS(1)
        QXN(J,I) = QX(I)
        QWN(J,I) = QW(I)/DSQRT(PI)
        XXN(J,I) = QX(I)*DSQRT(2.D0)
        ENDDO
        DO I =1, NGAUSS_SM(1)
        QXN_SM(J,I) = QX_SM(I)
        QWN_SM(J,I) = QW_SM(I)/DSQRT(PI)
      XXN_SM(J,I)  = QX_SM(I)*DSQRT(2.D0)
        ENDDO
        ENDDO




        ALLOCATE( YR1(N,NGAUSS_MAX),YR2(N,N,NGAUSS_MAX,NGAUSS_MAX),
        1  H(N,NBAS_MAX,NGAUSS_MAX),
     &           ALFA(N,NBAS_MAX),PSIF(N,NBAS_MAX),PSIF_N(N,NBAS_MAX),
     1      BITA(N,N,NBAS_MAX,NBAS_MAX),
     &           YR3(N,N,N,NGAUSS_MAX,NGAUSS_MAX,NGAUSS_MAX),
     &           GAMA(N,N,N,NBAS_MAX,NBAS_MAX,NBAS_MAX),
     &   SMR1(N,NGAUSS_SM_MAX),SMR2(N,N,NGAUSS_SM_MAX,NGAUSS_SM_MAX),
     &   SMH(N,NBAS_SM_MAX,NGAUSS_SM_MAX),ALFA_SM(N,NBAS_SM_MAX),
     &   PSIF_SM(N,NBAS_SM_MAX),
     &   BITA_SM(N,N,NBAS_SM_MAX,NBAS_SM_MAX),
     &           SMR3(N,N,N,NGAUSS_SM_MAX,NGAUSS_SM_MAX,NGAUSS_SM_MAX),
     &           GAMA_SM(N,N,N,NBAS_SM_MAX,NBAS_SM_MAX,NBAS_SM_MAX),
     &   SGR1(N,NGAUSS_SM_MAX),SGR2(N,N,NGAUSS_SM_MAX,NGAUSS_SM_MAX),
     &   SGH(N,NBAS_SM_MAX,NGAUSS_SM_MAX),ALFA_SG(N,NBAS_SM_MAX),
     &   PSIF_SG(N,NBAS_SM_MAX),
     &   BITA_SG(N,N,NBAS_SM_MAX,NBAS_SM_MAX),
     &           SGR3(N,N,N,NGAUSS_SM_MAX,NGAUSS_SM_MAX,NGAUSS_SM_MAX),
     &           GAMA_SG(N,N,N,NBAS_SM_MAX,NBAS_SM_MAX,NBAS_SM_MAX),
     &    ALFA_SGIID(N,NBAS_SM_MAX),ALFA_SMIID(N,NBAS_SM_MAX) )

      ALLOCATE(H_GQ(N,NBAS_MAX,NGAUSS_GQ),SCMU(NGAUSS_GQ,
     1  NGAUSS_GQ,NGAUSS_GQ),SCSIG(NGAUSS_GQ,NGAUSS_GQ,NGAUSS_GQ),
     1           YPDD(NGAUSS_GQ,NGAUSS_GQ,NGAUSS_GQ))


C
C       INITIALIZE COEFFICIENTS
C


        Y0 = 0.



      IF (IID .EQ. 1) THEN
C         IF (IID .EQ. 1 .AND. (NGAUSS .NE. NGAUSS_SM .OR.
C     1            NBAS .NE. NBAS_SM)) THEN
C         PRINT*,' NGAUSS != NGAUSS_SM  or NBAS != NBAS_SM '
C         STOP
C         ENDIF
      ENDIF


        DO I = 1,N
         DO J = 1,NBAS(I)
         ALFA(I,J) = 0.
         ENDDO
        ENDDO

        DO I1 = 1,N
         DO I2 = 1,N
          DO J1 = 1,NBAS(I1)
           DO J2 = 1,NBAS(I2)
           BITA(I1,I2,J1,J2) = 0.
           ENDDO
          ENDDO
         ENDDO
        ENDDO



        DO I1 = 1,N
         DO I2 = 1,N
          DO I3 = 1,N
           DO J1 = 1,NBAS(I1)
            DO J2 = 1,NBAS(I2)
             DO J3 = 1,NBAS(I3)
             GAMA(I1,I2,I3,J1,J2,J3) = 0.
             ENDDO
            ENDDO
           ENDDO
          ENDDO
         ENDDO
        ENDDO

        ALFA = 0.
        BITA = 0.
        GAMA = 0.
```

```
C       CALCULATE/STORE RESPONSE AT MEAN INPUT (YM  SSM)
C
        DO I = 1,N
        X(I) = RDD_MU(I)
        ENDDO
        IF (IFLAG_AIJK .EQ. 0)  THEN
        CALL RESPON_FEA (X,YM, IFLAG_Y)
C      CALL RESPON_FEA (X,YM, IFLAG_Y)
        NO_GAUSS = NO_GAUSS + 1
        IF (IFLAG_Y .EQ. 1) N_EF1 = N_EF1 +1
        IF (IFLAG_Y .EQ. 2) N_EC1 = N_EC1 +1
        ELSEIF(IFLAG_AIJK .EQ. 1) THEN
        CALL RESPON_AIJK(NBAS_A,AIJKC(IFLAG_Y,:,:,:,:), X, YM)
        ELSE
        PRINT*, 'WRONG IFLAG = ', IFLAG
        STOP
        ENDIF


C       CALL SCOR (X,SMM,SGM)

C
C       CALCULATE/STORE UNIVARIATE RESPONSES AT GAUSS POINTS (YR1)
C
      YR1 = 0.
        DO IX = 1,N
         DO I = 1,N
         X(I) = RDD_MU(I)
         ENDDO

          DO IGAUSS = 1,NGAUSS(IX)
          X(IX) = XXN(IX,IGAUSS)
            IF (IFLAG_AIJK .EQ. 0)  THEN
            CALL RESPON_FEA (X,Y, IFLAG_Y)
C          CALL RESPON_FEA (X,Y, IFLAG_Y)

            NO_GAUSS = NO_GAUSS + 1
          IF (IFLAG_Y .EQ. 1) N_EF1 = N_EF1 +1
            IF (IFLAG_Y .EQ. 2) N_EC1 = N_EC1 +1

            ELSEIF(IFLAG_AIJK .EQ. 1) THEN
            CALL RESPON_AIJK(NBAS_A,AIJKC(IFLAG_Y,:,:,:,:), X, Y)

            ELSE
            PRINT*, 'WRONG IFLAG_AIJK = ', IFLAG_AIJK
            STOP
            ENDIF

          YR1(IX,IGAUSS) = Y
          ENDDO
        ENDDO





C
C       CALCULATE/STORE BIVARIATE RESPONSES AT GAUSS POINTS (YR2)
C
      YR2 = 0.
      IF (NVAR .GE. 2) THEN
        DO IX = 1,N-1
         DO JX = IX+1,N
          DO I = 1,N
          X(I) = RDD_MU(I)
          ENDDO

C                HERE CALCULATE THE VALUE OF Y AT THE CASE ONLY VARIABLE X(IX) AND X(JX) NONZERO  IN NGAUSS X NGAUSS INTEGRAL POINTS
           DO IGAUSS = 1,NGAUSS(IX)
            DO JGAUSS = 1,NGAUSS(JX)
            X(IX) = XXN(IX,IGAUSS)
            X(JX) = XXN(JX,JGAUSS)
            IF (IFLAG_AIJK .EQ. 0)  THEN
            CALL RESPON_FEA (X,Y, IFLAG_Y)
C          CALL RESPON_FEA (X,Y, IFLAG_Y)
            NO_GAUSS = NO_GAUSS + 1
            IF (IFLAG_Y .EQ. 1) N_EF2 = N_EF2 +1
            IF (IFLAG_Y .EQ. 2) N_EC2 = N_EC2 +1
            ELSEIF(IFLAG_AIJK .EQ. 1) THEN
            CALL RESPON_AIJK(NBAS_A,AIJKC(IFLAG_Y,:,:,:,:), X, Y)
            ELSE
            PRINT*, 'WRONG IFLAG_AIJK = ', IFLAG_AIJK
            STOP
            ENDIF

            YR2(IX,JX,IGAUSS,JGAUSS) = Y

            ENDDO
           ENDDO
         ENDDO
        ENDDO
        ENDIF






C
C
C       CALCULATE/STORE TRIVARIATE RESPONSES AT GAUSS POINTS (YR3)
C
      YR3 = 0.
      IF (NVAR .EQ. 3) THEN
        DO IX = 1,N-2
         DO JX = IX+1,N-1
          DO KX = JX+1,N
           DO I = 1,N
           X(I) = RDD_MU(I)
           ENDDO
```

```
            DO IGAUSS = 1,NGAUSS(IX)
             DO JGAUSS = 1,NGAUSS(JX)
              DO KGAUSS = 1,NGAUSS(KX)
              X(IX) = XXN(IX,IGAUSS)
              X(JX) = XXN(JX,JGAUSS)
              X(KX) = XXN(KX,KGAUSS)
                IF (IFLAG_AIJK .EQ. 0)  THEN
                CALL RESPON_FEA (X,Y, IFLAG_Y)
C              CALL RESPON_FEA (X,Y, IFLAG_Y)
                NO_GAUSS = NO_GAUSS + 1
                IF (IFLAG_Y .EQ. 1) N_EF2 = N_EF2 +1
                IF (IFLAG_Y .EQ. 2) N_EC2 = N_EC2 +1
                ELSEIF(IFLAG_AIJK .EQ. 1) THEN
                CALL RESPON_AIJK(NBAS_A,AIJKC(IFLAG_Y,:,:,:,:), X, Y)
                ELSE
                PRINT*, 'WRONG IFLAG_AIJK = ', IFLAG_AIJK
                STOP
                ENDIF
              YR3(IX,JX,KX,IGAUSS,JGAUSS,KGAUSS) = Y
              ENDDO
             ENDDO
            ENDDO

          ENDDO
         ENDDO
        ENDDO
        ENDIF




C
C       CALCULATE/STORE BASIS FUNCTIONS AT GAUSS POINTS (H)

        DO I = 1,N
         DO J = 1,NBAS(I)
          DO L = 1,N
          X(L) = RDD_MU(L)
          ENDDO
           DO IGAUSS = 1,NGAUSS(I)
           X(I) = XXN(I,IGAUSS)
         CALL BASIS_NORMAL (X,PSIF,N,NBAS(I))
        H(I,J,IGAUSS) = PSIF(I,J)
           ENDDO
         ENDDO
        ENDDO



C
C       CALCULATE COEFFICIENT Y0
C


        YU = 0.
        DO IX = 1,N
         DO IGAUSS = 1,NGAUSS(IX)
         YU = YU + YR1(IX,IGAUSS)*QWN(IX,IGAUSS)
       ENDDO
        ENDDO




C
        YB = 0.
        DO IX = 1,N-1
         DO JX = IX+1,N
          DO IGAUSS = 1,NGAUSS(IX)
           DO JGAUSS = 1,NGAUSS(JX)
           YB = YB + YR2(IX,JX,IGAUSS,JGAUSS)*QWN(IX,IGAUSS)*
     &        QWN(JX,JGAUSS)
         ENDDO
          ENDDO
         ENDDO
        ENDDO



C
        YT = 0.
        DO IX = 1,N-2
         DO JX = IX+1,N-1
          DO KX = JX+1,N
           DO IGAUSS = 1,NGAUSS(IX)
            DO JGAUSS = 1,NGAUSS(JX)
             DO KGAUSS = 1,NGAUSS(KX)
             YT = YT + YR3(IX,JX,KX,IGAUSS,JGAUSS,KGAUSS)*
     &          QWN(IX,IGAUSS)*QWN(JX,JGAUSS)*
     &          QWN(KX,KGAUSS)
           ENDDO
            ENDDO
           ENDDO
          ENDDO
         ENDDO
        ENDDO



C
C
      NT = N
      IF (NVAR .EQ. 3) THEN
      Y0 = YT - DFLOAT(NT-3)*YB + DFLOAT((NT-2)*(NT-3))*YU/2.
     &    - DFLOAT((NT-1)*(NT-2)*(NT-3))*YM/6.

      ELSE IF (NVAR .EQ. 2) THEN
      Y0 =  YB
     &    - DFLOAT((NT-2))*YU + DFLOAT((NT-2)*(NT-1))*YM/2.
      ELSE IF (NVAR .EQ. 1) THEN
      Y0 =  YU
     &    - DFLOAT((NT-1))*YM
      ENDIF
```

```
         DO I = 1,N
          DO J = 1,NBAS(I)
C
          YYU = 0.
           DO IGAUSS = 1,NGAUSS(I)
           YYU = YYU + YR1(I,IGAUSS)*H(I,J,IGAUSS)*QWN(I,IGAUSS)
           ENDDO
C

C    DO 10 IX = 1,N-1
C    DO 10 JX = IX+1,N
C    IF (IX .NE. I .AND. JX .NE. I) GO TO 10
C     DO IGAUSS = 1,NGAUSS(IX)
C      DO JGAUSS = 1,NGAUSS(JX)
C      IF (IX .EQ. I) KGAUSS = IGAUSS
C      IF (JX .EQ. I) KGAUSS = JGAUSS
C      YYB = YYB + YR2(IX,JX,IGAUSS,JGAUSS)*H(I,J,KGAUSS)*
C     &           (QW(IGAUSS)/DSQRT(PI))*(QW(JGAUSS)/DSQRT(PI))
C      ENDDO
C     ENDDO
C10    CONTINUE
C
          YYB = 0.
            DO 10 IX = 1,N-1
            DO 10 JX = IX+1,N
            IF (IX .NE. I .AND. JX .NE. I) GO TO 10
             DO IGAUSS = 1,NGAUSS(IX)
              DO JGAUSS = 1,NGAUSS(JX)
              IF (IX .EQ. I) KGAUSS = IGAUSS
              IF (JX .EQ. I) KGAUSS = JGAUSS
              YYB = YYB + YR2(IX,JX,IGAUSS,JGAUSS)*H(I,J,KGAUSS)*
      &            QWN(IX,IGAUSS)*QWN(JX,JGAUSS)
              ENDDO
             ENDDO
10          CONTINUE

C
C
           YYT = 0.
            DO 20 IX = 1,N-2
            DO 20 JX = IX+1,N-1
            DO 20 KX = JX+1,N
            IF (IX .NE. I .AND. JX .NE. I .AND. KX .NE. I) GO TO 20
             DO IGAUSS = 1,NGAUSS(IX)
              DO JGAUSS = 1,NGAUSS(JX)
               DO KGAUSS = 1,NGAUSS(KX)
               IF (IX .EQ. I) MGAUSS = IGAUSS
               IF (JX .EQ. I) MGAUSS = JGAUSS
               IF (KX .EQ. I) MGAUSS = KGAUSS
               YYT = YYT + YR3(IX,JX,KX,IGAUSS,JGAUSS,KGAUSS)*H(I,J,MGAUSS)*
      &             QWN(IX,IGAUSS)*QWN(JX,JGAUSS)*
      &             QWN(KX,KGAUSS)
                     ENDDO
              ENDDO
             ENDDO
20          CONTINUE
C
C  --------------  IJNME PDD EQ.(26) (23)
       NT = N
       IF (NVAR .EQ. 3) THEN
       ALFA(I,J) = YYT - DFLOAT(NT-3)*YYB + DFLOAT((NT-2)*(NT-3))*YYU/2.

       ELSE IF (NVAR .EQ. 2) THEN
       ALFA(I,J) =  YYB
       &     - DFLOAT((NT-2))*YYU
       ELSE IF (NVAR .EQ. 1) THEN
       ALFA(I,J) =  YYU
       ENDIF

c          ALFA(I,J) = YYT - DFLOAT(NT-3)*YYB + DFLOAT((NT-2)*(NT-3))*YYU/2.
C
          ENDDO
         ENDDO






C
C       CALCULATE COEFFICIENTS BITA(I1,I2,J1,J2)
C
         DO I1 = 1,N-1
          DO I2 = I1+1,N
           DO J1 = 1,NBAS(I1)
            DO J2 = 1,NBAS(I2)
C
           YYYB = 0.
             DO IGAUSS = 1,NGAUSS(I1)
              DO JGAUSS = 1,NGAUSS(I2)
              YYYB = YYYB +
      &  YR2(I1,I2,IGAUSS,JGAUSS)*H(I1,J1,IGAUSS)*H(I2,J2,JGAUSS)*
      &  QWN(I1,IGAUSS)*QWN(I2,JGAUSS)
               ENDDO
              ENDDO
```

```
C
          YYYT = 0.
           DO 30 IIX = 1,N-2
           DO 30 JJX = IIX+1,N-1
           DO 30 KKX = JJX+1,N
           IF (IIX .NE. I1 .AND. IIX .NE. I2 .AND.
    &        JJX .NE. I1 .AND. JJX .NE. I2 .AND.
    &        KKX .NE. I1 .AND. KKX .NE. I2) GO TO 30
           IF (IIX .EQ. I1 .AND. (JJX .EQ. I2 .OR. KKX .EQ. I2)) GO TO 31
           IF (JJX .EQ. I1 .AND. (IIX .EQ. I2 .OR. KKX .EQ. I2)) GO TO 31
           IF (KKX .EQ. I1 .AND. (IIX .EQ. I2 .OR. JJX .EQ. I2)) GO TO 31
           GO TO 30
31        DO IGAUSS = 1,NGAUSS(IIX)
             DO JGAUSS = 1,NGAUSS(JJX)
              DO KGAUSS = 1,NGAUSS(KKX)
              IF (IIX .EQ. I1) MGAUSS1 = IGAUSS
              IF (JJX .EQ. I1) MGAUSS1 = JGAUSS
              IF (KKX .EQ. I1) MGAUSS1 = KGAUSS
              IF (IIX .EQ. I2) MGAUSS2 = IGAUSS
              IF (JJX .EQ. I2) MGAUSS2 = JGAUSS
              IF (KKX .EQ. I2) MGAUSS2 = KGAUSS
              YYYT = YYYT + YR3(IIX,JJX,KKX,IGAUSS,JGAUSS,KGAUSS)*
    &       H(I1,J1,MGAUSS1)*H(I2,J2,MGAUSS2)*QWN(IIX,IGAUSS)*
    &       QWN(JJX,JGAUSS)*QWN(KKX,KGAUSS)
                  ENDDO
             ENDDO
            ENDDO
30         CONTINUE
C

         NT = N
      IF (NVAR .EQ. 3) THEN
      BITA(I1,I2,J1,J2) = YYYT - DFLOAT(NT-3)*YYYB

      ELSE IF (NVAR .EQ. 2) THEN
      BITA(I1,I2,J1,J2) =  YYYB
      ELSE IF (NVAR .EQ. 1) THEN
      BITA(I1,I2,J1,J2) =  0.
      ENDIF
C          BITA(I1,I2,J1,J2) = YYYT - DFLOAT(NT-3)*YYYB

C       WRITE(*,*) I1, I2, J1, J2,   BITA(I1,I2,J1,J2)
C
           ENDDO
          ENDDO
         ENDDO
        ENDDO




C
C
C       CALCULATE COEFFICIENTS GAMA(I1,I2,I3,J1,J2,J3)
C
        DO I1 = 1,N-2
         DO I2 = I1+1,N-1
          DO I3 = I2+1,N
           DO J1 = 1,NBAS(I1)
            DO J2 = 1,NBAS(I2)
             DO J3 = 1,NBAS(I3)
C
             GAMA(I1,I2,I3,J1,J2,J3) = 0.
            DO IGAUSS = 1,NGAUSS(I1)
               DO JGAUSS = 1,NGAUSS(I2)
                DO KGAUSS = 1,NGAUSS(I3)
              GAMA(I1,I2,I3,J1,J2,J3) = GAMA(I1,I2,I3,J1,J2,J3) +
    &               YR3(I1,I2,I3,IGAUSS,JGAUSS,KGAUSS)*
    &         H(I1,J1,IGAUSS)*H(I2,J2,JGAUSS)*H(I3,J3,KGAUSS)*
    &         QWN(I1,IGAUSS)*
    &         QWN(I2,JGAUSS)*QWN(I3,KGAUSS)
                    ENDDO
               ENDDO
              ENDDO
C
             ENDDO
            ENDDO
           ENDDO
          ENDDO
         ENDDO
        ENDDO




C      =============================================
C        NOW CALCULATE THE Q-TH (Q up to 3rd) MOMENT BY ANALYTICAL EXPRESSION using direct pdd of response function
C      =============================================

        PRINT*, 'q-th (Q up to 3rd) MOMENT  OF Y BY ANALYTICAL EXPRESSION
     1    USING DIRECT pdd to response function y  IS  ....'
C

C      ===================
C      MOMENT OF ANALYTICAL
C      ===================

C      ----    1ST MOMENT
      RM1 = Y0
      PRINT*,  '--------------------------------------------'
      WRITE(*,*) 'RM1 = ', RM1


C      ----    2ND MOMENT

          RM2 = Y0**2.0

          DO I = 1,N
              DO J = 1,NBAS(I)
                RM2 = RM2 + ALFA(I,J)**2.0
              ENDDO
          ENDDO
          PRINT*,  '--------------------------------------------'
          WRITE(*,*) 'RM2 (UNIVARITE) = ', RM2
          RM2U = RM2
```

```
          DO I1 = 1,N-1
          DO I2 = I1+1,N
              DO J1 = 1,NBAS(I1)
              DO J2 = 1,NBAS(I2)
                RM2 = RM2 + BITA(I1,I2,J1,J2)**2.0
              ENDDO
              ENDDO
          ENDDO
          ENDDO

          PRINT*,  '-------------------------------------------'
          WRITE(*,*) 'RM2 (BIVARITE) = ', RM2
          RM2B = RM2

          DO I1 = 1,N-2
          DO I2 = I1+1,N-1
          DO I3 = I2+1,N
              DO J1 = 1,NBAS(I1)
              DO J2 = 1,NBAS(I2)
              DO J3 = 1,NBAS(I3)
                RM2 = RM2 + GAMA(I1,I2,I3,J1,J2,J3)**2.0
              ENDDO
              ENDDO
              ENDDO
          ENDDO
          ENDDO
          ENDDO

          RM2T = RM2

          PRINT*,  '-------------------------------------------'
          WRITE(*,*) 'RM2 (TRIVARITE) = ', RM2




C    ----     3RD MOMENT
C    -------    FOR THIS PROBLEM (EXAMPLE 4.2 OF SH LEE), THERE IS NO ANALYTICAL EXPRESSION SO FAR, SO THE FOLLOWING CALCULATION IS NOT RIGHT
C    -------    HOWEVER SINCE, IN THIS PROBLEM, THERE IS NO 3RD MOMENT REQUESTED, SO IT DOESN'S MATTER TO KEEP THE FOLLOWING




C   ======================
C   SENSITIVITY OF ANALYTICAL
C   =====================


      DO IMU = 1,N
      PRINT *, 'IMU = ', IMU

        SM0 = 0.
        SG0 = 0.
C
C  ------- FOR ONLY FIRST TWO VARIABLES, NEED SENSITIVITY
        DO I = 1,N
         DO J = 1,NBAS_SM(I)
         ALFA_SM(I,J) = 0.
         ALFA_SG(I,J) = 0.
         ENDDO
        ENDDO


        DO I1 = 1,N
         DO I2 = 1,N
          DO J1 = 1,NBAS_SM(I1)
           DO J2 = 1,NBAS_SM(I2)
           BITA_SM(I1,I2,J1,J2) = 0.
           BITA_SG(I1,I2,J1,J2) = 0.
           ENDDO
          ENDDO
         ENDDO
        ENDDO

        DO I1 = 1,N
         DO I2 = 1,N
          DO I3 = 1,N
           DO J1 = 1,NBAS_SM(I1)
            DO J2 = 1,NBAS_SM(I2)
             DO J3 = 1,NBAS_SM(I3)
             GAMA_SM(I1,I2,I3,J1,J2,J3) = 0.
             GAMA_SG(I1,I2,I3,J1,J2,J3) = 0.
             ENDDO
            ENDDO
           ENDDO
          ENDDO
         ENDDO
        ENDDO
        ALFA_SM = 0.
        BITA_SM = 0.
        GAMA_SM = 0.
      ALFA_SG = 0.
        BITA_SG = 0.
        GAMA_SG = 0.
        ALFA_SMIID = 0.
        ALFA_SGIID = 0.

C
C
C       CALCULATE/STORE CONSTANT TERM OF SCORE FUNCTION AT GAUSS POINTS
C

        DO I = 1,N
        X(I) = RDD_MU(I)
        ENDDO
        CALL SCOR (X,SMU_A,SSIG_A)
        SMM = SMU_A(IMU)
        SGM = SSIG_A(IMU)

C
C
C       CALCULATE/STORE UNIVARIATE SCORE FUNCTION AT GAUSS POINTS
C
```

```
        DO IX = 1,N
         DO I = 1,N
         X(I) = RDD_MU(I)
         ENDDO
          DO IGAUSS = 1,NGAUSS_SM(IMU)
          X(IMU) = XXN_SM(IMU,IGAUSS)
C         X1 = X(IX)
          CALL SCOR (X,SMU_A, SSIG_A)
          S_MU = SMU_A(IMU)
          S_SIG = SSIG_A(IMU)
          SMR1(IX,IGAUSS) = S_MU
          SGR1(IX,IGAUSS) = S_SIG
          ENDDO
        ENDDO

C
C
C       CALCULATE/STORE BIVARIATE SCORE FUNCTION AT GAUSS POINTS
C


        DO IX = 1,N-1
         DO JX = IX+1,N
          DO I = 1,N
          X(I) = RDD_MU(I)
          ENDDO
           DO IGAUSS = 1,NGAUSS_SM(IMU)
            DO JGAUSS = 1,NGAUSS_SM(IMU)
            X(IMU) = XXN_SM(IMU,IGAUSS)
            X(IMU) = XXN_SM(IMU,JGAUSS)
C           X1 = X(IX)
          CALL SCOR (X,SMU_A, SSIG_A)
          S_MU = SMU_A(IMU)
          S_SIG = SSIG_A(IMU)
            SMR2(IX,JX,IGAUSS,JGAUSS) = S_MU
            SGR2(IX,JX,IGAUSS,JGAUSS) = S_SIG
            ENDDO
           ENDDO
         ENDDO
        ENDDO

        SMR2 = 0.

C
C
C       CALCULATE/STORE TRIVARIATE SCORE FUNCTION AT GAUSS POINTS
C

        DO IX = 1,N-2
         DO JX = IX+1,N-1
          DO KX = JX+1,N
           DO I = 1,N
           X(I) = RDD_MU(I)
           ENDDO
            DO IGAUSS = 1,NGAUSS_SM(IMU)
             DO JGAUSS = 1,NGAUSS_SM(IMU)
              DO KGAUSS = 1,NGAUSS_SM(IMU)
              X(IMU) = XXN_SM(IMU,IGAUSS)
              X(IMU) = XXN_SM(IMU,JGAUSS)
              X(IMU) = XXN_SM(IMU,KGAUSS)
C           X1 = X(IX)
            CALL SCOR (X,SMU_A, SSIG_A)
              S_MU = SMU_A(IMU)
              S_SIG = SSIG_A(IMU)
              SMR3(IX,JX,KX,IGAUSS,JGAUSS,KGAUSS) = S_MU
              SGR3(IX,JX,KX,IGAUSS,JGAUSS,KGAUSS) = S_SIG
              ENDDO
             ENDDO
            ENDDO
          ENDDO
         ENDDO
        ENDDO
        SMR3 = 0.

C
C       CALCULATE/STORE BASIS FUNCTIONS AT GAUSS POINTS (H)
C
        DO I = 1,N
         DO J = 1,NBAS_SM(IMU)
          DO L = 1,N
          X(L) = RDD_MU(L)
          ENDDO
           DO IGAUSS = 1,NGAUSS_SM(IMU)
           X(IMU) = XXN_SM(IMU,IGAUSS)
           CALL BASIS_NORMAL (X,PSIF_SM,N,NBAS_SM(IMU))
           CALL BASIS_NORMAL (X,PSIF_SG,N,NBAS_SM(IMU))
          SMH(I,J,IGAUSS) = PSIF_SM(IMU,J)
          SGH(I,J,IGAUSS) = PSIF_SG(IMU,J)
           ENDDO
         ENDDO
        ENDDO

C
C       CALCULATE COEFFICIENT Y0
C



        SMU = 0.
        SGU = 0.
        DO IX = 1,N
         DO IGAUSS = 1,NGAUSS_SM(IMU)
         SMU = SMU + SMR1(IX,IGAUSS)*QWN_SM(IMU,IGAUSS)
         SGU = SGU + SGR1(IX,IGAUSS)*QWN_SM(IMU,IGAUSS)
       ENDDO
        ENDDO

        SMB = 0.
        SGB = 0.
        DO IX = 1,N-1
         DO JX = IX+1,N
          DO IGAUSS = 1,NGAUSS_SM(IMU)
          DO JGAUSS = 1,NGAUSS_SM(IMU)
           SMB = SMB + SMR2(IX,JX,IGAUSS,JGAUSS)*(QWN_SM(IMU,IGAUSS))*
     &        (QWN_SM(IMU,JGAUSS))
           SGB = SGB + SGR2(IX,JX,IGAUSS,JGAUSS)*(QWN_SM(IMU,IGAUSS))*
     &        (QWN_SM(IMU,JGAUSS))

         ENDDO
```

```fortran
          ENDDO
         ENDDO
        ENDDO


        SMT = 0.
        SGT = 0.
        DO IX = 1,N-2
         DO JX = IX+1,N-1
          DO KX = JX+1,N
           DO IGAUSS = 1,NGAUSS_SM(IMU)
            DO JGAUSS = 1,NGAUSS_SM(IMU)
             DO KGAUSS = 1,NGAUSS_SM(IMU)
             SMT = SMT + SMR3(IX,JX,KX,IGAUSS,JGAUSS,KGAUSS)*
     &           (QWN_SM(IMU,IGAUSS))*(QWN_SM(IMU,JGAUSS))*
     &           (QWN_SM(IMU,KGAUSS))
             SGT = SGT + SGR3(IX,JX,KX,IGAUSS,JGAUSS,KGAUSS)*
     &           (QWN_SM(IMU,IGAUSS))*(QWN_SM(IMU,JGAUSS))*
     &           (QWN_SM(IMU,KGAUSS))

           ENDDO
            ENDDO
           ENDDO
          ENDDO
         ENDDO
        ENDDO

C     DOUBLE CHECK SM0 Y0 'S FORMULAR FROM PAPER ??
C     DUE TO IN IID CASE, SM0 AND SG0 ARE ONE VARIABLES FUNCTION, SO THE EXPRESSION CAN BE
C     SM0 = SMU - (N-1)*SMM
C     SG0 = SGU - (N-1)*SGM


      NT = 1
      SM0 = (SMU - (NT-1.)*SMM)/DFLOAT(N)
      SG0 = (SGU - (NT-1.)*SGM)/DFLOAT(N)

C      SG0 = (SGU - (N-1.)*SGM)

C   SM0 = SMT - DFLOAT(N-3)*SMB + DFLOAT((N-2)*(N-3))*SMU/2.
C    &      - DFLOAT((N-1)*(N-2)*(N-3))*SMM/6.
       ttttt = SGT - DFLOAT(NT-3)*SGB + DFLOAT((NT-2)*(NT-3))*SGU/2.
     &      - DFLOAT((NT-1)*(NT-2)*(NT-3))*SGM/6.
C

C
C       CALCULATE COEFFICIENTS ALFA(I,J)
C

C
C       CALCULATE COEFFICIENTS ALFA_SM(I,J), ALFA_SG(I,J)
```

```fortran
C
C      CHECK SSMU, COZ SHOULD BE  = 18.XX
         DO I = IMU,IMU
          DO J = 1,NBAS_SM(IMU)
C
          SSMU = 0.
          SSGU = 0.
           DO IGAUSS = 1,NGAUSS_SM(IMU)
           SSMU = SSMU + SMR1(IMU,IGAUSS)*SMH(IMU,J,IGAUSS)*QWN_SM(IMU,IGAUSS)
           SSGU = SSGU + SGR1(IMU,IGAUSS)*SGH(IMU,J,IGAUSS)*QWN_SM(IMU,IGAUSS)
         ENDDO
C
           SSMB = 0.
           SSGB = 0.
            DO 110 IX = 1,N-1
            DO 110 JX = IX+1,N
            IF (IX .NE. I .AND. JX .NE. I) GO TO 110
             DO IGAUSS = 1,NGAUSS_SM(IMU)
              DO JGAUSS = 1,NGAUSS_SM(IMU)
              IF (IX .EQ. I) KGAUSS = IGAUSS
              IF (JX .EQ. I) KGAUSS = JGAUSS
              SSMB = SSMB + SMR2(IMU,IMU,IGAUSS,JGAUSS)*SMH(IMU,J,KGAUSS)*
     &            (QWN_SM(IMU,IGAUSS))*(QWN_SM(IMU,JGAUSS))
              SSGB = SSGB + SGR2(IMU,IMU,IGAUSS,JGAUSS)*SGH(IMU,J,KGAUSS)*
     &            (QWN_SM(IMU,IGAUSS))*(QWN_SM(IMU,JGAUSS))

              ENDDO
             ENDDO
110         CONTINUE
C
C
C      ALFA_SM EVERY DIMENSION NOT EQUAL   DOUBLE CHECK HERE
           SSMT = 0.
           SSGT = 0.
            DO 120 IX = 1,N-2
            DO 120 JX = IX+1,N-1
            DO 120 KX = JX+1,N
            IF (IX .NE. I .AND. JX .NE. I .AND. KX .NE. I) GO TO 120
             DO IGAUSS = 1,NGAUSS_SM(IMU)
              DO JGAUSS = 1,NGAUSS_SM(IMU)
               DO KGAUSS = 1,NGAUSS_SM(IMU)
              IF (IX .EQ. I) MGAUSS = IGAUSS
              IF (JX .EQ. I) MGAUSS = JGAUSS
              IF (KX .EQ. I) MGAUSS = KGAUSS
               SSMT = SSMT + SMR3(IMU,IMU,IMU,IGAUSS,JGAUSS,KGAUSS)*
     1        SMH(IMU,J,MGAUSS)
     &            *(QWN_SM(IMU,IGAUSS))*(QWN_SM(IMU,JGAUSS))*
     &            (QWN_SM(IMU,KGAUSS))
               SSGT = SSGT + SGR3(IMU,IMU,IMU,IGAUSS,JGAUSS,KGAUSS)*
     1        SGH(IMU,J,MGAUSS)
     &            *(QWN_SM(IMU,IGAUSS))*(QWN_SM(IMU,JGAUSS))*
     &            (QWN_SM(IMU,KGAUSS))
                 ENDDO
              ENDDO
             ENDDO
120         CONTINUE


            ALFA_SM(I,J) = SSMU
            ALFA_SG(I,J) = SSGU
C
          ENDDO
         ENDDO


C  ----------    FORMULA 14   OF LR.PDF   ----------

             DO I = 1,N
              DO J = 1,NBAS_SM(IMU)
                IF (IID .EQ. 1) THEN
                ALFA_SMIID(I,J) = ALFA_SM(1,J)
                ALFA_SGIID(I,J) = ALFA_SG(1,J)
                ELSE IF (IID .EQ. 0) THEN
                ALFA_SMIID(I,J) = ALFA_SM(I,J)
                ALFA_SGIID(I,J) = ALFA_SG(I,J)
                END IF
              ENDDO
             ENDDO

C
C        CALCULATE COEFFICIENTS BITA_SM(I1,I2,J1,J2), BITA_SG(I1,I2,J1,J2)
C
         DO I1 = 1,N-1
          DO I2 = I1+1,N
           DO J1 = 1,NBAS_SM(IMU)
            DO J2 = 1,NBAS_SM(IMU)
C
           SSSMB = 0.
           SSSGB = 0.
             DO IGAUSS = 1,NGAUSS_SM(IMU)
              DO JGAUSS = 1,NGAUSS_SM(IMU)
              SSSMB = SSSMB +
     &      SMR2(I1,I2,IGAUSS,JGAUSS)*SMH(IMU,J1,IGAUSS)*
     &       SMH(IMU,J2,JGAUSS)*
     &      (QWN_SM(IMU,IGAUSS))*(QWN_SM(IMU,JGAUSS))
              SSSGB = SSSGB +
     &      SGR2(I1,I2,IGAUSS,JGAUSS)*SGH(IMU,J1,IGAUSS)*
     &       SGH(IMU,J2,JGAUSS)*
     &      (QWN_SM(IMU,IGAUSS))*(QWN_SM(IMU,JGAUSS))


              ENDDO
             ENDDO
```

```fortran
C
           SSSMT = 0.
           SSSGT = 0.
           DO 130 JJX = IIX+1,N-1
           DO 130 KKX = JJX+1,N
           IF (IIX .NE. I1 .AND. IIX .NE. I2 .AND.
     &       JJX .NE. I1 .AND. JJX .NE. I2 .AND.
     &       KKX .NE. I1 .AND. KKX .NE. I2) GO TO 130
           IF (IIX .EQ. I1 .AND. (JJX .EQ. I2 .OR. KKX .EQ. I2)) GO TO 131
           IF (JJX .EQ. I1 .AND. (IIX .EQ. I2 .OR. KKX .EQ. I2)) GO TO 131
           IF (KKX .EQ. I1 .AND. (IIX .EQ. I2 .OR. JJX .EQ. I2)) GO TO 131
           GO TO 130
131        DO IGAUSS = 1,NGAUSS_SM(IMU)
             DO JGAUSS = 1,NGAUSS_SM(IMU)
              DO KGAUSS = 1,NGAUSS_SM(IMU)
              IF (IIX .EQ. I1) MGAUSS1 = IGAUSS
              IF (JJX .EQ. I1) MGAUSS1 = JGAUSS
              IF (KKX .EQ. I1) MGAUSS1 = KGAUSS
              IF (IIX .EQ. I2) MGAUSS2 = IGAUSS
              IF (JJX .EQ. I2) MGAUSS2 = JGAUSS
              IF (KKX .EQ. I2) MGAUSS2 = KGAUSS
              SSSMT = SSSMT + SMR3(IIX,JJX,KKX,IGAUSS,JGAUSS,KGAUSS)*
     &      SMH(IMU,J1,MGAUSS1)*SMH(IMU,J2,MGAUSS2)*
     &      (QWN_SM(IMU,IGAUSS))*
     &      (QWN_SM(IMU,JGAUSS))*(QWN_SM(IMU,KGAUSS))

              SSSGT = SSSGT + SGR3(IIX,JJX,KKX,IGAUSS,JGAUSS,KGAUSS)*
     &      SGH(IMU,J1,MGAUSS1)*SGH(IMU,J2,MGAUSS2)*
     &      (QWN_SM(IMU,IGAUSS))*
     &      (QWN_SM(IMU,JGAUSS))*(QWN_SM(IMU,KGAUSS))

                  ENDDO
             ENDDO
            ENDDO
130        CONTINUE
C
           BITA_SM(I1,I2,J1,J2) = SSSMT - DFLOAT(N-3)*SSSMB
           BITA_SG(I1,I2,J1,J2) = SSSGT - DFLOAT(N-3)*SSSGB

C       WRITE(*,*) I1, I2, J1, J2,   BITA(I1,I2,J1,J2)
C
           ENDDO
          ENDDO
         ENDDO
        ENDDO


C
C
C       CALCULATE COEFFICIENTS GAMA_SM(I1,I2,I3,J1,J2,J3), GAMA_SG(I1,I2,I3,J1,J2,J3)
C
        DO I1 = 1,N-2
         DO I2 = I1+1,N-1
          DO I3 = I2+1,N
           DO J1 = 1,NBAS_SM(IMU)
            DO J2 = 1,NBAS_SM(IMU)
             DO J3 = 1,NBAS_SM(IMU)
C
             GAMA_SM(I1,I2,I3,J1,J2,J3) = 0.
             GAMA_SG(I1,I2,I3,J1,J2,J3) = 0.
            DO IGAUSS = 1,NGAUSS_SM(IMU)
               DO JGAUSS = 1,NGAUSS_SM(IMU)
                DO KGAUSS = 1,NGAUSS_SM(IMU)
              GAMA_SM(I1,I2,I3,J1,J2,J3) = GAMA_SM(I1,I2,I3,J1,J2,J3) +
     &              SMR3(I1,I2,I3,IGAUSS,JGAUSS,KGAUSS)*
     &        SMH(IMU,J1,IGAUSS)*SMH(IMU,J2,JGAUSS)*SMH(IMU,J3,KGAUSS)*
     &       (QWN_SM(IMU,IGAUSS))*
     &       (QWN_SM(IMU,JGAUSS))*(QWN_SM(IMU,KGAUSS))

              GAMA_SG(I1,I2,I3,J1,J2,J3) = GAMA_SG(I1,I2,I3,J1,J2,J3) +
     &              SGR3(I1,I2,I3,IGAUSS,JGAUSS,KGAUSS)*
     &        SGH(IMU,J1,IGAUSS)*SGH(IMU,J2,JGAUSS)*SGH(IMU,J3,KGAUSS)*
     &       (QWN_SM(IMU,IGAUSS))*
     &       (QWN_SM(IMU,JGAUSS))*(QWN_SM(IMU,KGAUSS))

                    ENDDO
               ENDDO
              ENDDO
C
             ENDDO
            ENDDO
           ENDDO
          ENDDO
         ENDDO
        ENDDO







      IF (IID .EQ. 1) THEN
C          PRINT*,  'INDEPENDENT IDENTICAL DISTRIBUTION ...'
C          NBAS_SEN = MIN(NBAS, NBAS_SM)
C          SENSM = N* SM0
C          SENSG = N*SG0
C          DO I = 1,N
C              DO J = 1,NBAS_SEN
C                SENSM =  SENSM + ALFA(I,J)*ALFA_SMIID(I,J)
C                SENSG =  SENSG + ALFA(I,J)*ALFA_SGIID(I,J)
C              ENDDO
C          ENDDO
C
C
C          WRITE(*,*) 'SENSM = ', SENSM
C          WRITE(*,*) 'SENSG = ', SENSG
      ELSE IF (IID .EQ. 0) THEN
          PRINT*,  'INDEPENDENT BUT NOT INDENTICAL DISTRIBUTION '
c     1              NOT READY YET ... '
```

```
C          NBAS_SEN = MIN(NBAS, NBAS_SM)
C          SENSM = N* SM0
C          SENSG = N*SG0
C          DO I = 1,N
C              DO J = 1,NBAS_SEN
C                SENSM =  SENSM + ALFA(I,J)*ALFA_SM(I,J)
C                SENSG =  SENSG + ALFA(I,J)*ALFA_SG(I,J)
C              ENDDO
C          ENDDO
C
      ELSE IF (IID .EQ. 2) THEN
          PRINT*,  'DEPENDENT DISTRIBUTION '
c     1              NOT READY YET ... '
C          NBAS_SEN = MIN(NBAS, NBAS_SM)
C          SENSM = N* SM0
C          SENSG = N*SG0
C          DO I = 1,N
C              DO J = 1,NBAS_SEN
C                SENSM =  SENSM + ALFA(I,J)*ALFA_SM(I,J)
C                SENSG =  SENSG + ALFA(I,J)*ALFA_SG(I,J)
C              ENDDO
C          ENDDO
C
      END IF


C     ============================================
C       NOW CALCULATE THE SENSIVITY OF Q-TH (Q up to 2nd)  MOMENT BY ANALYTICAL EXPRESSION using direct pdd of response function
C     ============================================
        PRINT*, 'the sensitivity q-th (Q up to 2nd) MOMENT  OF Y BY ANALYTICAL
     1   EXPRESSION  USING DIRECT pdd to response function y  IS  ....'
C


C     ==================
C     MOMENT OF ANALYTICAL
C     ==================


C     ----    1ST MOMENT


          PRINT*,  'INDEPENDENT IDENTICAL DISTRIBUTION ...'

        NBAS_SEN = MIN(NBAS_MAX, NBAS_SM(IMU))
        SENSM1 = SM0*Y0
        SENSG1 = SG0*Y0
        DO I = IMU,IMU
            DO J = 1,NBAS_SEN
              SENSM1 =  SENSM1 + ALFA(I,J)*ALFA_SMIID(I,J)
              SENSG1 =  SENSG1 + ALFA(I,J)*ALFA_SGIID(I,J)
            ENDDO
        ENDDO
        PRINT*,  '---------(sensitivity of 1st moment)-------------'
        WRITE(*,*) 'SENSM1  = ', SENSM1
        WRITE(*,*) 'SENSG1 = ',  SENSG1






C     ----    2ND MOMENT AND SENSITIVITY OF 2ND MOMENT

          SENSM2 = Y0**2.0*SM0 + SM0*(RM2-Y0**2.0) +
     1              2*Y0*(SENSM1 - SM0*Y0)
          SENSG2 = Y0**2.0*SG0 + SG0*(RM2-Y0**2.0) +
     1              2*Y0*(SENSG1 - SG0*Y0)


C     U*U*U



      DO I1=1,N
      DO K1=1,N
      DO M1=IMU,IMU
          DO J1=1,NBAS(I1)
          DO L1=1,NBAS(K1)
          DO N1=1,NBAS_SM(IMU)

          IF (I1 .EQ. K1 .AND. K1 .EQ. M1) THEN
            IS = (J1+L1+N1)/2
            IF(MOD(J1+L1+N1, 2) .EQ. 0 .AND. J1 .LE. IS
     1        .AND. L1 .LE. IS  .AND. N1 .LE. IS ) THEN
        SENSM2 = SENSM2 + ALFA(I1,J1)*ALFA(K1,L1)*ALFA_SMIID(M1,N1)*
     1      DSQRT(GAMMA(1.0+DFLOAT(J1))*GAMMA(1.0+DFLOAT(L1))*
     2      GAMMA(1.0+DFLOAT(N1)))/(GAMMA(1.0+DFLOAT(IS-J1))
     2      *GAMMA(1.0+DFLOAT(IS-L1))*GAMMA(1.0+DFLOAT(IS-N1)))

        SENSG2 = SENSG2 + ALFA(I1,J1)*ALFA(K1,L1)*ALFA_SGIID(M1,N1)*
     1      DSQRT(GAMMA(1.0+DFLOAT(J1))*GAMMA(1.0+DFLOAT(L1))*
     2      GAMMA(1.0+DFLOAT(N1)))/(GAMMA(1.0+DFLOAT(IS-J1))
     2      *GAMMA(1.0+DFLOAT(IS-L1))*GAMMA(1.0+DFLOAT(IS-N1)))
            ENDIF
          ENDIF

          ENDDO
          ENDDO
          ENDDO
      ENDDO
      ENDDO
      ENDDO

          PRINT*,  '----------(sensitivity of 2ND moment)------------'
          WRITE(*,*) 'SENSM2 (UNIVARITE) = ', SENSM2
          WRITE(*,*) 'SENSG2 (UNIVARITE) = ', SENSG2
          SENSM2U = SENSM2
          SENSG2U = SENSG2


C     B*U*U
```

```
      DO I1=1,N-1
      DO I2=I1+1,N
      DO K1=1,N
      DO M1=IMU,IMU
          DO J1=1,NBAS(I1)
          DO J2=1,NBAS(I2)
          DO L1=1,NBAS(K1)
          DO N1=1,NBAS_SM(IMU)

          IF (I1 .EQ. K1 .AND. I2 .EQ. M1 ) THEN
            IF (J1 .EQ. L1 .AND. J2 .EQ. N1) THEN
            SENSM2 =  SENSM2 + BITA(I1,I2,J1,J2)*ALFA(K1,L1)*
     1                  ALFA_SMIID(M1,N1)
            SENSG2 =  SENSG2 + BITA(I1,I2,J1,J2)*ALFA(K1,L1)*
     1                  ALFA_SGIID(M1,N1)
            ENDIF
          ENDIF
          IF (I2 .EQ. K1 .AND. I1 .EQ. M1 ) THEN
            IF (J2 .EQ. L1 .AND. J1 .EQ. N1) THEN
            SENSM2 =  SENSM2 + BITA(I1,I2,J1,J2)*ALFA(K1,L1)*
     1                  ALFA_SMIID(M1,N1)
            SENSG2 =  SENSG2 + BITA(I1,I2,J1,J2)*ALFA(K1,L1)*
     1                  ALFA_SGIID(M1,N1)
            ENDIF
          ENDIF


          ENDDO
          ENDDO
          ENDDO
          ENDDO
      ENDDO
      ENDDO
      ENDDO
      ENDDO

C     U*B*U

      DO I1=1,N
      DO K1=1,N-1
      DO K2=K1+1,N
      DO M1=IMU,IMU
          DO J1=1,NBAS(I1)
          DO L1=1,NBAS(K1)
          DO L2=1,NBAS(K2)
          DO N1=1,NBAS_SM(IMU)

          IF (K1 .EQ. I1 .AND. K2 .EQ. M1 ) THEN
            IF (L1 .EQ. J1 .AND. L2 .EQ. N1) THEN
            SENSM2 =  SENSM2 + BITA(K1,K2,L1,L2)*ALFA(I1,J1)*
     1                  ALFA_SMIID(M1,N1)
            SENSG2 =  SENSG2 + BITA(K1,K2,L1,L2)*ALFA(I1,J1)*
     1                  ALFA_SGIID(M1,N1)
            ENDIF
          ENDIF
          IF (K2 .EQ. I1 .AND. K1 .EQ. M1 ) THEN
            IF (L2 .EQ. J1 .AND. L1 .EQ. N1) THEN
            SENSM2 =  SENSM2 + BITA(K1,K2,L1,L2)*ALFA(I1,J1)*
     1                  ALFA_SMIID(M1,N1)
            SENSG2 =  SENSG2 + BITA(K1,K2,L1,L2)*ALFA(I1,J1)*
     1                  ALFA_SGIID(M1,N1)
            ENDIF
          ENDIF


          ENDDO
          ENDDO
          ENDDO
          ENDDO
      ENDDO
      ENDDO
      ENDDO
      ENDDO

C     B*B*U
      DO I1=1,N-1
      DO I2=I1+1,N
      DO K1=1,N-1
      DO K2=K1+1,N
      DO M1=IMU,IMU
          DO J1=1,NBAS(I1)
          DO J2=1,NBAS(I2)
          DO L1=1,NBAS(K1)
          DO L2=1,NBAS(K2)
          DO N1=1,NBAS_SM(IMU)

          IF (I1 .EQ. K1 .AND. I1 .EQ. M1 ) THEN
           IS = (J1+L1+N1)/2
            IF(MOD(J1+L1+N1, 2) .EQ. 0 .AND. J1 .LE. IS
     1       .AND. L1 .LE. IS  .AND. N1 .LE. IS ) THEN
            IF (I2 .EQ. K2 .AND. J2 .EQ. L2) THEN
             SENSM2 = SENSM2 + BITA(I1,I2,J1,J2)*BITA(K1,K2,L1,L2)
     1         *ALFA_SMIID(M1,N1)*
     1     DSQRT(GAMMA(1.0+DFLOAT(J1))*GAMMA(1.0+DFLOAT(L1))*
     2     GAMMA(1.0+DFLOAT(N1)))/(GAMMA(1.0+DFLOAT(IS-J1))
     2     *GAMMA(1.0+DFLOAT(IS-L1))*GAMMA(1.0+DFLOAT(IS-N1)))

             SENSG2 = SENSG2 + BITA(I1,I2,J1,J2)*BITA(K1,K2,L1,L2)
     1         *ALFA_SGIID(M1,N1)*
     1     DSQRT(GAMMA(1.0+DFLOAT(J1))*GAMMA(1.0+DFLOAT(L1))*
     2     GAMMA(1.0+DFLOAT(N1)))/(GAMMA(1.0+DFLOAT(IS-J1))
     2     *GAMMA(1.0+DFLOAT(IS-L1))*GAMMA(1.0+DFLOAT(IS-N1)))

            ENDIF
            ENDIF
          ENDIF

          IF (I2 .EQ. K2 .AND. I2 .EQ. M1 ) THEN
           IS = (J2+L2+N1)/2
            IF(MOD(J2+L2+N1, 2) .EQ. 0 .AND. J2 .LE. IS
     1       .AND. L2 .LE. IS  .AND. N1 .LE. IS ) THEN
            IF (I1 .EQ. K1 .AND. J1 .EQ. L1) THEN
             SENSM2 = SENSM2 + BITA(I1,I2,J1,J2)*BITA(K1,K2,L1,L2)
     1         *ALFA_SMIID(M1,N1)*
     1     DSQRT(GAMMA(1.0+DFLOAT(J2))*GAMMA(1.0+DFLOAT(L2))*
     2     GAMMA(1.0+DFLOAT(N1)))/(GAMMA(1.0+DFLOAT(IS-J2))
     2     *GAMMA(1.0+DFLOAT(IS-L2))*GAMMA(1.0+DFLOAT(IS-N1)))
```

```
             SENSG2 = SENSG2 + BITA(I1,I2,J1,J2)*BITA(K1,K2,L1,L2)
     1          *ALFA_SGIID(M1,N1)*
     1     DSQRT(GAMMA(1.0+DFLOAT(J2))*GAMMA(1.0+DFLOAT(L2))*
     2     GAMMA(1.0+DFLOAT(N1)))/(GAMMA(1.0+DFLOAT(IS-J2))
     2     *GAMMA(1.0+DFLOAT(IS-L2))*GAMMA(1.0+DFLOAT(IS-N1)))


            ENDIF
            ENDIF
          ENDIF







          ENDDO
          ENDDO
          ENDDO
          ENDDO
          ENDDO
      ENDDO
      ENDDO
      ENDDO
      ENDDO
      ENDDO


C          PRINT*,  '-----------------------------------------'
C          WRITE(*,*) 'SENSM2 (BIVARITE) = ', SENSM2
C          WRITE(*,*) 'SENSG2 (BIVARITE) = ', SENSG2
      SENSM2B = SENSM2
      SENSG2B = SENSG2






C     T*U*U, UTU, UUT are zero

      SENSM3 = SENSM2
      SENSG3 = SENSG2
C     T*B*U
      DO I1=1,N-2
      DO I2=I1+1,N-1
      DO I3=I2+1,N
      DO K1=1,N-1
      DO K2=K1+1,N
      DO M1=IMU,IMU
          DO J1=1,NBAS(I1)
          DO J2=1,NBAS(I2)
          DO J3=1,NBAS(I3)
          DO L1=1,NBAS(K1)
          DO L2=1,NBAS(K2)
          DO N1=1,NBAS_SM(IMU)

          IF (I1 .EQ. K1 .AND. I2 .EQ. K2 .AND. I3 .EQ. M1 ) THEN
            IF (J1 .EQ. L1 .AND. J2 .EQ. L2 .AND. J3 .EQ. N1) THEN
            SENSM3  =  SENSM3  + GAMA(I1,I2,I3,J1,J2,J3)*
     1              BITA(K1,K2,L1,L2)*ALFA_SMIID(M1,N1)

            SENSG3  =  SENSG3  + GAMA(I1,I2,I3,J1,J2,J3)*
     1              BITA(K1,K2,L1,L2)*ALFA_SGIID(M1,N1)
            ENDIF
          ENDIF
          IF (I1 .EQ. K1 .AND. I2 .EQ. M1 .AND. I3 .EQ. K2 ) THEN
            IF (J1 .EQ. L1 .AND. J2 .EQ. N1 .AND. J3 .EQ. L2) THEN
            SENSM3  =  SENSM3  + GAMA(I1,I2,I3,J1,J2,J3)*
     1              BITA(K1,K2,L1,L2)*ALFA_SMIID(M1,N1)

            SENSG3  =  SENSG3  + GAMA(I1,I2,I3,J1,J2,J3)*
     1              BITA(K1,K2,L1,L2)*ALFA_SGIID(M1,N1)
            ENDIF
          ENDIF




          IF (I1 .EQ. M1 .AND. I2 .EQ. K1 .AND. I3 .EQ. K2 ) THEN
            IF (J1 .EQ. N1 .AND. J2 .EQ. L1 .AND. J3 .EQ. L2) THEN
            SENSM3  =  SENSM3  + GAMA(I1,I2,I3,J1,J2,J3)*
     1              BITA(K1,K2,L1,L2)*ALFA_SMIID(M1,N1)

            SENSG3  =  SENSG3  + GAMA(I1,I2,I3,J1,J2,J3)*
     1              BITA(K1,K2,L1,L2)*ALFA_SGIID(M1,N1)
            ENDIF
          ENDIF




          ENDDO
          ENDDO
          ENDDO
          ENDDO
          ENDDO
          ENDDO
      ENDDO
      ENDDO
      ENDDO
      ENDDO
      ENDDO
      ENDDO


C     B*T*U
      DO I1=1,N-1
      DO I2=I1+1,N
      DO K1=1,N-2
      DO K2=K1+1,N-1
      DO K3=K2+1,N
      DO M1=IMU,IMU
          DO J1=1,NBAS(I1)
          DO J2=1,NBAS(I2)
          DO L1=1,NBAS(K1)
          DO L2=1,NBAS(K2)
          DO L3=1,NBAS(K3)
          DO N1=1,NBAS_SM(IMU)
```

```fortran
          IF (K1 .EQ. I1 .AND. K2 .EQ. I2 .AND. K3 .EQ. M1 ) THEN
            IF (L1 .EQ. J1 .AND. L2 .EQ. J2 .AND. L3 .EQ. N1) THEN
            SENSM3 =  SENSM3 + GAMA(K1,K2,K3,L1,L2,L3)*BITA(I1,I2,J1,J2)
     1              *ALFA_SMIID(M1,N1)

            SENSG3 =  SENSG3 + GAMA(K1,K2,K3,L1,L2,L3)*BITA(I1,I2,J1,J2)
     1              *ALFA_SGIID(M1,N1)
            ENDIF
          ENDIF
          IF (K1 .EQ. I1 .AND. K2 .EQ. M1 .AND. K3 .EQ. I2 ) THEN
            IF (L1 .EQ. J1 .AND. L2 .EQ. N1 .AND. L3 .EQ. J2) THEN
            SENSM3 =  SENSM3 + GAMA(K1,K2,K3,L1,L2,L3)*BITA(I1,I2,J1,J2)
     1              *ALFA_SMIID(M1,N1)

            SENSG3 =  SENSG3 + GAMA(K1,K2,K3,L1,L2,L3)*BITA(I1,I2,J1,J2)
     1              *ALFA_SGIID(M1,N1)
            ENDIF
          ENDIF




          IF (K1 .EQ. M1 .AND. K2 .EQ. I1 .AND. K3 .EQ. I2 ) THEN
            IF (L1 .EQ. N1 .AND. L2 .EQ. J1 .AND. L3 .EQ. J2) THEN
            SENSM3 =  SENSM3 + GAMA(K1,K2,K3,L1,L2,L3)*BITA(I1,I2,J1,J2)
     1              *ALFA_SMIID(M1,N1)

            SENSG3 =  SENSG3 + GAMA(K1,K2,K3,L1,L2,L3)*BITA(I1,I2,J1,J2)
     1              *ALFA_SGIID(M1,N1)
            ENDIF
          ENDIF


          ENDDO
          ENDDO
          ENDDO
          ENDDO
          ENDDO
          ENDDO
      ENDDO
      ENDDO
      ENDDO
      ENDDO
      ENDDO
      ENDDO


C     T*T*U
      DO I1=1,N-2
      DO I2=I1+1,N-1
      DO I3=I2+1,N
      DO K1=1,N-2
      DO K2=K1+1,N-1
      DO K3=K2+1,N
      DO M1=IMU,IMU
          DO J1=1,NBAS(I1)
          DO J2=1,NBAS(I2)
          DO J3=1,NBAS(I3)
          DO L1=1,NBAS(K1)
          DO L2=1,NBAS(K2)
          DO L3=1,NBAS(K3)
          DO N1=1,NBAS_SM(IMU)


          IF (M1 .EQ. I1 .AND. M1 .EQ. K1 ) THEN

            IS = (J1+L1+N1)/2

            IF( ( MOD(J1+L1+N1, 2) .EQ. 0 .AND. J1 .LE. IS
     1        .AND. L1 .LE. IS  .AND. N1 .LE. IS)) THEN

            IF( I2 .EQ. K2 .AND. I3 .EQ. K3 ) THEN
            IF( J2 .EQ. L2 .AND. J3 .EQ. L3)  THEN
             SENSM3 = SENSM3 + GAMA(I1,I2,I3,J1,J2,J3)*
     1          GAMA(K1,K2,K3,L1,L2,L3)*
     1          ALFA_SMIID(M1,N1)*
     1     DSQRT(GAMMA(1.0+DFLOAT(J1))*GAMMA(1.0+DFLOAT(L1))
     2          *GAMMA(1.0+DFLOAT(N1)) )/(GAMMA(1.0+DFLOAT(IS-J1))
     2     *GAMMA(1.0+DFLOAT(IS-L1))*GAMMA(1.0+DFLOAT(IS-N1)) )

             SENSG3 = SENSG3 + GAMA(I1,I2,I3,J1,J2,J3)*
     1          GAMA(K1,K2,K3,L1,L2,L3)*
     1          ALFA_SGIID(M1,N1)*
     1     DSQRT(GAMMA(1.0+DFLOAT(J1))*GAMMA(1.0+DFLOAT(L1))
     2          *GAMMA(1.0+DFLOAT(N1)) )/(GAMMA(1.0+DFLOAT(IS-J1))
     2     *GAMMA(1.0+DFLOAT(IS-L1))*GAMMA(1.0+DFLOAT(IS-N1)) )
            ENDIF
            ENDIF

            ENDIF
          ENDIF


          IF (M1 .EQ. I2 .AND. M1 .EQ. K2 ) THEN

            IS = (J2+L2+N1)/2

            IF( ( MOD(IS, 2) .EQ. 0 .AND. J2 .LE. IS
     1        .AND. L2 .LE. IS  .AND. N1 .LE. IS)) THEN

            IF( I1 .EQ. K1 .AND. I3 .EQ. K3 ) THEN
            IF( J1 .EQ. L1 .AND. J3 .EQ. L3)  THEN
             SENSM3 = SENSM3 + GAMA(I1,I2,I3,J1,J2,J3)*
     1          GAMA(K1,K2,K3,L1,L2,L3)*
     1          ALFA_SMIID(M1,N1)*
     1     DSQRT(GAMMA(1.0+DFLOAT(J2))*GAMMA(1.0+DFLOAT(L2))
     2          *GAMMA(1.0+DFLOAT(N1)) )/(GAMMA(1.0+DFLOAT(IS-J2))
     2     *GAMMA(1.0+DFLOAT(IS-L2))*GAMMA(1.0+DFLOAT(IS-N1)) )

             SENSG3 = SENSG3 + GAMA(I1,I2,I3,J1,J2,J3)*
     1          GAMA(K1,K2,K3,L1,L2,L3)*
     1          ALFA_SGIID(M1,N1)*
     1     DSQRT(GAMMA(1.0+DFLOAT(J2))*GAMMA(1.0+DFLOAT(L2))
     2          *GAMMA(1.0+DFLOAT(N1)) )/(GAMMA(1.0+DFLOAT(IS-J2))
     2     *GAMMA(1.0+DFLOAT(IS-L2))*GAMMA(1.0+DFLOAT(IS-N1)) )

            ENDIF
            ENDIF
```

```
            ENDIF
          ENDIF

          IF (M1 .EQ. I3 .AND. M1 .EQ. K3 ) THEN

            IS = (J3+L3+N1)/2

            IF( ( MOD(IS, 2) .EQ. 0 .AND. J3 .LE. IS
     1        .AND. L3 .LE. IS  .AND. N1 .LE. IS)) THEN

            IF( I1 .EQ. K1 .AND. I2 .EQ. K2 ) THEN
            IF( J1 .EQ. L1 .AND. J2 .EQ. L2)  THEN
             SENSM3 = SENSM3 + GAMA(I1,I2,I3,J1,J2,J3)*
     1           GAMA(K1,K2,K3,L1,L2,L3)*
     1           ALFA_SMIID(M1,N1)*
     1      DSQRT(GAMMA(1.0+DFLOAT(J3))*GAMMA(1.0+DFLOAT(L3))
     2           *GAMMA(1.0+DFLOAT(N1)) )/(GAMMA(1.0+DFLOAT(IS-J3))
     2      *GAMMA(1.0+DFLOAT(IS-L3))*GAMMA(1.0+DFLOAT(IS-N1)) )

             SENSG3 = SENSG3 + GAMA(I1,I2,I3,J1,J2,J3)*
     1           GAMA(K1,K2,K3,L1,L2,L3)*
     1           ALFA_SGIID(M1,N1)*
     1      DSQRT(GAMMA(1.0+DFLOAT(J3))*GAMMA(1.0+DFLOAT(L3))
     2           *GAMMA(1.0+DFLOAT(N1)) )/(GAMMA(1.0+DFLOAT(IS-J3))
     2      *GAMMA(1.0+DFLOAT(IS-L3))*GAMMA(1.0+DFLOAT(IS-N1)) )


            ENDIF
            ENDIF

            ENDIF
          ENDIF



          ENDDO
          ENDDO
          ENDDO
          ENDDO
          ENDDO
          ENDDO
          ENDDO
      ENDDO
      ENDDO
      ENDDO
      ENDDO
      ENDDO
      ENDDO
      ENDDO

       SENSM2T = SENSM3
       SENSG2T = SENSG3



          PRINT*,  '----------------------------------------'
          WRITE(*,*) 'SENSM3 (TRIVARITE) = ', SENSM3
          WRITE(*,*) 'SENSG3 (TRIVARITE) = ', SENSG3






C
C        NOW START SIMULATION FOR UNIVARIATE/BIVARIATE/DIRECT MCS
C
C        PRINT*, 'STARTING SIMULATION ....'
C
C      PFU = FAULURE PROBABLITY OF UNIVARIATE APPROX,
C      PFB =                       BIVARIATE
C      PFT =                       TRIVARIATE
C      PFMCS =                     DIRECT MCS
C      SMOMU = MOMENT CALC BY UNIVARIATE APPROX
C      SMUSU = SENSITIVITY TO MU OF MOMENT CALC BY UNIVARIATE APPROX
C      SSIGSU = SENSITIVITY TO SIG OF MOMENT CALC BY UNIVARIATE APPROX
C      SMOMB = MOMENT CALC BY BIVARIATE APPROX
C      SMUSB = SENSITIVITY TO MU OF MOMENT CALC BY BIVARIATE APPROX
C      SSIGSB = SENSITIVITY TO SIG OF MOMENT CALC BY BIVARIATE APPROX
C      SMOMT = MOMENT CALC BY TRIVARIATE APPROX
C      SMUST = SENSITIVITY TO MU OF MOMENT CALC BY TRIVARIATE APPROX
C      SSIGST = SENSITIVITY TO SIG OF MOMENT CALC BY TRIVARIATE APPROX

         PFU = 0.
         PFB = 0.
         PFT = 0.
         PFMCS = 0.
         SMOMU = 0.
         SMUSU = 0.
         SSIGSU = 0.
         SMOMB = 0.
         SMUSB = 0.
         SSIGSB = 0.
         SMOMT = 0.
         SMUST = 0.
         SSIGST = 0.

         Y_EXC_MU = 0.
         Y_EXC_SIG = 0.
         SENS_EXC_1MU = 0.
         SENS_EXC_2MU = 0.

C
         KK = 1
      KSTEP = NSAMP/10
         ISEED = 1234567
```

```fortran
        CALL RNSET(ISEED)
c       OPEN(108,FILE='C.TXT')
        DO 100 ISAMP = 1,NSAMP
C
        DO I = 1,N
C     This function generates a pseudorandom number from a standard normal distribution.
        X(I) = DRNNOF()
        ENDDO
      DO I =1, N
        CALL BASIS_NORMAL (X,PSIF,N,NBAS(I))
        PSIF_N(I,:) = PSIF(I,:)
        ENDDO
C
C       UNIVARIATE APPROXIMATION
C


        SYU = Y0
        DO I = 1,N
         DO J = 1,NBAS(I)
         SYU = SYU + ALFA(I,J)*PSIF_N(I,J)
         ENDDO
        ENDDO
        IND = 0.
        IF (SYU .LE. 0.) IND = 1.
        PFU = PFU + IND

C   SYU = 0.
C
C   DO I = 1,N
C    SYUI(I) = 0.
C    DO J = 1,NBAS
C    SYUI(I) = SYUI(I)+ ALFA(I,J)*PSIF(I,J)
C    ENDDO
C    SYU = SYU + SYUI(I)
C   ENDDO
C   SYU = SYU + Y0
C   IND = 0.
C   IF (SYU .LE. 0.) IND = 1.
C   PFU = PFU + IND
C
C
C       UNIVARIATE APPROXIMATION OF MOMENT AND SENSITIVITY
C     SMUU  ---- UNIVIATE APPROXIMATION/EXACT VALUE OF SCORE FUNCTION OF MU
C     SSIGU  ---- UNIVIATE APPROXIMATION/EXACT VALUE OF SCORE FUNCTION OF SIG  (MAY CALL SCOR () TO GET THE EXACT VALUE)
C     FOR IDD CASE
        SMUU =  SM0
        SSIGU = SG0

       DO I = 1,N
           DO J = 1,NBAS_SM(IMU)
            SMUU = SMUU + ALFA_SMIID(I,J)*PSIF_N(I,J)
            SSIGU = SSIGU + ALFA_SGIID(I,J)*PSIF_N(I,J)
        ENDDO
           ENDDO



c       SMUU =  SMUU_EX
c       SSIGU = SSIGU_EX








C      IF (IID .NE. 1) THEN
C        PRINT*, 'NOT READY FOR IID != 1 SIMULATION'
C        STOP
C      ENDIF

      CALL PDF_IID(X,YPDF)
      SMOMU1 = SMOMU1 + SYU
      SMUSU1 = SMUSU1 + SMUU*SYU
      SSIGSU1 = SSIGSU1 + SSIGU*SYU

      SMOMU1 = SMOMU
      SMOMU2 = SMOMU**2.0
      SMUSU2 = SMUSU2 + SMUU2*SYU
      SSIGSU = SSIGSU + SSIGU*SYU
C          PRINT*,  '----------------------------------------'
C          WRITE(*,*) 'SMOMU1 (UNIVARITE) = ', SENSM2
C          WRITE(*,*) 'SMOMU1 (UNIVARITE) = ', SENSG2
CC
C       BIVARIATE APPROXIMATION
C
        SYB = SYU
        DO I1 = 1,N-1
         DO I2 = I1+1,N
          DO J1 = 1,NBAS(I1)
           DO J2 = 1,NBAS(I2)
           SYB = SYB + BITA(I1,I2,J1,J2)*PSIF(I1,J1)*PSIF(I2,J2)
           ENDDO
          ENDDO
         ENDDO
        ENDDO
        IND = 0.
        IF (SYB .LE. 0.) IND = 1.
        PFB = PFB + IND
C       WRITE(108,'(I8,F18.6)') ISAMP , SYB
C
C     BIVARIATE APPROXIMATE FOR MOMENT AND SENSITIVITY
C
      SMOMB = SMOMB + SYB
      SMUSB = SMUSB + SMUU*SYB
      SSIGSB = SSIGSB + SSIGU*SYB


C
C
C       TRIVARIATE APPROXIMATION
C
```

```
         SYT = SYB
         DO I1 = 1,N-2
          DO I2 = I1+1,N-1
           DO I3 = I2+1,N
            DO J1 = 1,NBAS(I1)
             DO J2 = 1,NBAS(I2)
              DO J3 = 1,NBAS(I3)
              SYT = SYT + GAMA(I1,I2,I3,J1,J2,J3)*PSIF(I1,J1)*PSIF(I2,J2)*
      &           PSIF(I3,J3)
              ENDDO
             ENDDO
            ENDDO
           ENDDO
          ENDDO
         ENDDO
         IND = 0.
         IF (SYT .LE. 0.) IND = 1.
         PFT = PFT + IND

C
C      BIVARIATE APPROXIMATE FOR MOMENT AND SENSITIVITY
C
       SMOMT = SMOMT + SYT
       SMUST = SMUST + SMUU*SYT
       SSIGST = SSIGST + SSIGU*SYT



C
C        DIRECT MCS
C

         IND = 0.
         CALL RESPON_FEA (X,Y, IFLAG_Y)
         Y_EXC_MU = Y_EXC_MU + SYMCS
         Y_EXC_SIG = Y_EXC_SIG + SYMCS**2.0

         IF (IMU .EQ.  2) THEN
C          PRINT *, '----'
         ENDIF
       CALL SCOR (X,SMU_A_TMP,SSIG_A_TMP)
         SENS_EXC_1MU = SENS_EXC_1MU + SYMCS * SMUU
         SENS_EXC_2MU = SENS_EXC_2MU + SYMCS**2.0 * SMUU

         IF (SYMCS .LE. 0.) IND = 1.
         PFMCS = PFMCS + IND
C
C        PRINT*, Y0,SYU,SYB,SYT,SYMCS
C
       IF (ISAMP .EQ. KSTEP*KK) GO TO 28
       GO TO 100
28     KKK = KSTEP*KK
C        PRINT*, KKK, ' SAMPLES .....'
       KK = KK + 1
C
100      CONTINUE
C
```

```
C
C
        PFU = PFU/DFLOAT(NSAMP)
        PFB = PFB/DFLOAT(NSAMP)
        PFT = PFT/DFLOAT(NSAMP)
        PFMCS = PFMCS/DFLOAT(NSAMP)

      SMOMU = SMOMU/(DFLOAT(NSAMP))
      SMUSU = SMUSU/(DFLOAT(NSAMP))
      SSIGSU = SSIGSU/(DFLOAT(NSAMP))

      SMOMB = SMOMB/(DFLOAT(NSAMP))
      SMUSB = SMUSB/(DFLOAT(NSAMP))
      SSIGSB = SSIGSB/(DFLOAT(NSAMP))


      SMOMT = SMOMT/(DFLOAT(NSAMP))
      SMUST = SMUST/(DFLOAT(NSAMP))
      SSIGST = SSIGST/(DFLOAT(NSAMP))

        Y_EXC_MU = Y_EXC_MU/(DFLOAT(NSAMP))
        Y_EXC_SIG = Y_EXC_SIG/(DFLOAT(NSAMP))
        SENS_EXC_1MU = SENS_EXC_1MU/(DFLOAT(NSAMP))
        SENS_EXC_2MU = SENS_EXC_2MU/(DFLOAT(NSAMP))


C
C     PRINT STATISTICS
C
C   PRINT*,'EXAMPLE:'
C   PRINT*
C   PRINT*, 'UNIVARIATE:'
C   PRINT*
C   PRINT*, 'PF = ', PFU
C   PRINT*
C
C   PRINT*, 'BIVARIATE:'
C   PRINT*
C   PRINT*, 'PF = ', PFB
C   PRINT*
CC
C   PRINT*, 'TRIVARIATE:'
C   PRINT*
C   PRINT*, 'PF = ', PFT
C   PRINT*
CC

C      PRINT*, 'SMOMU = '
C      WRITE (*,*) SMOMU
C      PRINT*, 'SMUSU = '
C      WRITE (*,*) SMUSU
C      PRINT*, 'SSIGSU = '
C      WRITE (*,*) SSIGSU
C
C      PRINT*, 'SMOMB = '
C      WRITE (*,*) SMOMB
C      PRINT*, 'SMUSB = '
C      WRITE (*,*) SMUSB
C      PRINT*, 'SSIGSB = '
C      WRITE (*,*) SSIGSB
C
C      PRINT*, 'SMOMT = '
C      WRITE (*,*) SMOMT
C      PRINT*, 'SMUST = '
C      WRITE (*,*) SMUST
C      PRINT*, 'SSIGST = '
C      WRITE (*,*) SSIGST
C
C      PRINT*, 'Y_EXC_MU  = '
C      WRITE (*,*) Y_EXC_MU
C      PRINT*, 'Y_EXC_SIG  = '
C      WRITE (*,*) Y_EXC_SIG
C      PRINT*, 'SENS_EXC_1MU  = '
C      WRITE (*,*) SENS_EXC_1MU
C      PRINT*, 'SENS_EXC_2MU  = '
C      WRITE (*,*) SENS_EXC_2MU
C



C   PRINT*, 'DIRECT MCS:'
C   PRINT*
C   PRINT*, 'PFMCS = ', PFMCS
C   PRINT*
CC
C   PRINT*,'U.OUT, B.OUT, T.OUT, MCS.OUT OPENED FOR SAMPLE VALUES'
C   PRINT*, 'OF UNIVARIATE, BIVARIATE, TRIVARIATE, AND MCS SOLUTIONS'
CC
C
      IF (NVAR .EQ. 3) THEN
      RM2_OUT = RM2T
      SENSM2_OUT = SENSM2T
      SENSG2_OUT = SENSG2T
      ELSE IF (NVAR .EQ. 2) THEN
      RM2_OUT = RM2B
      SENSM2_OUT = SENSM2B
      SENSG2_OUT = SENSG2B
      ELSE IF (NVAR .EQ. 1) THEN
      RM2_OUT = RM2U
      SENSM2_OUT = SENSM2U
      SENSG2_OUT = SENSG2U
      ENDIF

      IF (IFLAG_MCS .EQ. 0) THEN
        MOMSEN_OUT(1,1) = RM1
        MOMSEN_OUT(1,2) = RM2_OUT
        MOMSEN_OUT(IMU+1,1) = SENSM1
        MOMSEN_OUT(IMU+1,2) = SENSM2_OUT
      ELSE IF(IFLAG_MCS .EQ. 1) THEN
        MOMSEN_OUT(1,1) = Y_EXC_MU
        MOMSEN_OUT(1,2) = Y_EXC_SIG
        MOMSEN_OUT(IMU+1,1) = SENS_EXC_1MU
        MOMSEN_OUT(IMU+1,2) = SENS_EXC_2MU
      ENDIF



C  ---------------------------------------------------
C
```

```
C   GAUSS QUADRATURE TO CALCULATE THE SENSITIVITY AND MOMENT
C
C  ----------------------------------------------------

C       PRINT*, 'STARTING GAUSS QUADRATURE ....'
        DO I = 1, NGAUSS_GQ
        DO J = 1, NGAUSS_GQ
        DO K = 1, NGAUSS_GQ
          YPDD(I,J,K) = Y0
          SCMU(I,J,K) = 0.
          SCSIG(I,J,K) = 0.
        ENDDO
        ENDDO
        ENDDO

C
C
C     PFU = FAULURE PROBABLITY OF UNIVARIATE APPROX,
C     PFB =                       BIVARIATE
C     PFT =                       TRIVARIATE
C     PFMCS =                     DIRECT MCS
C     SMOMU = MOMENT CALC BY UNIVARIATE APPROX
C     SMUSU = SENSITIVITY TO MU OF MOMENT CALC BY UNIVARIATE APPROX
C     SSIGSU = SENSITIVITY TO SIG OF MOMENT CALC BY UNIVARIATE APPROX
C     SMOMB = MOMENT CALC BY BIVARIATE APPROX
C     SMUSB = SENSITIVITY TO MU OF MOMENT CALC BY BIVARIATE APPROX
C     SSIGSB = SENSITIVITY TO SIG OF MOMENT CALC BY BIVARIATE APPROX
C     SMOMT = MOMENT CALC BY TRIVARIATE APPROX
C     SMUST = SENSITIVITY TO MU OF MOMENT CALC BY TRIVARIATE APPROX
C     SSIGST = SENSITIVITY TO SIG OF MOMENT CALC BY TRIVARIATE APPROX

        SMOMU = 0.
        SMUSU = 0.
        SSIGSU = 0.
        SMOMB = 0.
        SMUSB = 0.
        SSIGSB = 0.
        SMOMT = 0.
        SMUST = 0.
        SSIGST = 0.

C



C       GET QR RULE
      CALL DGQRUL (NGAUSS_GQ,4,0.,0.,0,QXFIX,QX,QW)


        DO IGAUSS = 1,NGAUSS_GQ
        XX(IGAUSS) = QX(IGAUSS)*DSQRT(2.D0)
        ENDDO
C
C       CALCULATE/STORE BASIS FUNCTIONS AT GAUSS POINTS (H) AND SCORE (SC)FUNCTION AT GAUSS POINTS
C
        DO I = 1,N
         DO J = 1,NBAS(I)
          DO L = 1,N
          X(L) = RDD_MU(L)
          ENDDO
           DO IGAUSS = 1,NGAUSS_GQ
           X(I) = XX(IGAUSS)
C      THIS CALL HAVE LOT OF UNNECESSORY CALCULATIONS
           IF (I .EQ. 1 .OR. I .EQ. 2) THEN
              CALL BASIS_NORMAL (X,PSIF,N,NBAS(I))
           ELSE IF (I .EQ. 3) THEN
              CALL BASIS_BETA (X,PSIF,N,NBAS(I))
           ELSE IF (I. EQ. 4) THEN
              CALL BASIS_GUMBEL (X,PSIF,N,NBAS(I))
           ELSE IF (I .EQ. 5) THEN
              CALL BASIS_LOGN (X,PSIF,N,NBAS(I))
           ENDIF

         H_GQ(I,J,IGAUSS) = PSIF(I,J)
           ENDDO
         ENDDO
        ENDDO

C     CALCULATE SCORE FUNCTION AT GAUSS POINTS
C     SMUU_SUM = 0.
C          SSIGU_SUM = 0.
C           DO I = 1,N
C         DO IGAUSS = 1, NGAUSS_GQ
C          X1 = XX(IGAUSS)
C          CALL SCOR (X1,SMUU, SSIGU)
C          SCMU(I,IGAUSS)=SMUU
C          SCSIG(I,IGAUSS) = SSIGU
C       ENDDO
C       ENDDO
C

              DO IG = 1, NGAUSS_GQ
              DO JG = 1, NGAUSS_GQ
              DO KG = 1, NGAUSS_GQ
                  X1(1) = XX(KG)
                  CALL SCOR (X1,SMU_A, SSIG_A)
                  SMUU = SMU_A(IMU)
                  SSIGU = SSIG_A(IMU)

                  SCMU(IG,JG,KG) = SCMU(IG,JG,KG) + SMUU
                  SCSIG(IG,JG,KG) = SCSIG(IG,JG,KG) + SSIGU
                  X1(1) = XX(JG)
                  CALL SCOR (X1,SMU_A, SSIG_A)
                  SMUU = SMU_A(IMU)
                  SSIGU = SSIG_A(IMU)

                  SCMU(IG,JG,KG) = SCMU(IG,JG,KG) + SMUU
                  SCSIG(IG,JG,KG) = SCSIG(IG,JG,KG) + SSIGU
                  X1(1) = XX(IG)
                  CALL SCOR (X1,SMU_A, SSIG_A)
                  SMUU = SMU_A(IMU)
                  SSIGU = SSIG_A(IMU)

                  SCMU(IG,JG,KG) = SCMU(IG,JG,KG) + SMUU
                  SCSIG(IG,JG,KG) = SCSIG(IG,JG,KG) + SSIGU

            ENDDO
            ENDDO
            ENDDO
```

```
C           CALL BASIS (X,PSIF,N,NBAS)
C
C       GAUSS QUADRATURE VALUE BY UNIVARIATE APPROXIMATION
C
C

C  ---- CALCULATE Y VALUE AT GAUSS POINT BY UNIVARIATE APPROX
C           SYU = Y0


         DO I = 1,N
         DO J = 1,NBAS(I)
              DO IG = 1, NGAUSS_GQ
              DO JG = 1, NGAUSS_GQ
              DO KG = 1, NGAUSS_GQ
C                 YPDD(IG,JG,KG) = YPDD(IG,JG,KG) + ALFA(I,J)*H_GQ(I,J,KG)
                  YPDD(IG,JG,KG) = YPDD(IG,JG,KG) + ALFA(I,J)*H_GQ(I,J,KG)/3.
        1              + ALFA(I,J)*H_GQ(I,J,JG)/3.+ ALFA(I,J)*H_GQ(I,J,IG)/3.
              ENDDO
C             YPDD(IG,JG,KG) = YPDD(IG,JG,KG) + ALFA(I,J)*H_GQ(I,J,JG)
              ENDDO
C             YPDD(IG,JG,KG) = YPDD(IG,JG,KG) + ALFA(I,J)*H_GQ(I,J,IG)
              ENDDO
         ENDDO
         ENDDO




C   -------- SUM ALL THE GAUSSIAN POINT FOR UNIVARIATE APPROX
C


C   SMUSB = 0.
C   SSIGSB = 0.
C   SMOMT = 0.
C   SMUST = 0.
C   SSIGST = 0.
C
C
            DO IG = 1, NGAUSS_GQ
              DO JG = 1, NGAUSS_GQ
              DO KG = 1, NGAUSS_GQ
                  SMUSU = SMUSU + YPDD(IG,JG,KG)*SCMU(IG,JG,KG)*QW(IG)
        1              /DSQRT(PI)*QW(JG)/DSQRT(PI)*QW(KG)/DSQRT(PI)
                  SSIGSU = SSIGSU + YPDD(IG,JG,KG)*SCSIG(IG,JG,KG)*QW(IG)
        1              /DSQRT(PI)*QW(JG)/DSQRT(PI)*QW(KG)/DSQRT(PI)
              ENDDO
              ENDDO
              ENDDO



C  ---- CALCULATE Y VALUE AT GAUSS POINT BY BIVARIATE APPROX
C  ----- IJNME PDD EQ.(29), I1<I2  need double check

         DO I1 = 1,N-1
         DO I2 = I1+1,N
         DO J1 = 1,NBAS(I1)
         DO J2 = 1,NBAS(I2)
              DO IG = 1, NGAUSS_GQ
              DO JG = 1, NGAUSS_GQ
              DO KG = 1, NGAUSS_GQ
                  YPDD(IG,JG,KG) = YPDD(IG,JG,KG) + BITA(I1,I2,J1,J2)
        1                  *H_GQ(I1,J1,KG)*H_GQ(I2,J2,IG)/3.
     2              + BITA(I1,I2,J1,J2)
        1                  *H_GQ(I1,J1,KG)*H_GQ(I2,J2,JG)/3.
     2              + BITA(I1,I2,J1,J2)
        1                  *H_GQ(I1,J1,IG)*H_GQ(I2,J2,JG)/3.

C             YPDD(IG,JG,KG) = YPDD(IG,JG,KG) + BITA(I1,I2,J1,J2)
C   1                  *H_GQ(I1,J1,KG)*H_GQ(I2,J2,IG)
C     2              + BITA(I1,I2,J1,J2)
C   1                  *H_GQ(I1,J1,KG)*H_GQ(I2,J2,JG)
            ENDDO
C         YPDD(IG,JG,KG) = YPDD(IG,JG,KG)
C     2              + BITA(I1,I2,J1,J2)
C   1                  *H_GQ(I1,J1,IG)*H_GQ(I2,J2,JG)
            ENDDO
              ENDDO
         ENDDO
         ENDDO
         ENDDO
         ENDDO





C   -------- SUM ALL THE GAUSSIAN POINT FOR BIVARIATE APPROX
C


C   SMUST = 0.
C   SSIGST = 0.
C
C
            DO IG = 1, NGAUSS_GQ
              DO JG = 1, NGAUSS_GQ
              DO KG = 1, NGAUSS_GQ
                  SMUSB = SMUSB + YPDD(IG,JG,KG)*SCMU(IG,JG,KG)*QW(IG)
        1              /DSQRT(PI)*QW(JG)/DSQRT(PI)*QW(KG)/DSQRT(PI)
                  SSIGSB = SSIGSB + YPDD(IG,JG,KG)*SCSIG(IG,JG,KG)*QW(IG)
        1              /DSQRT(PI)*QW(JG)/DSQRT(PI)*QW(KG)/DSQRT(PI)
              ENDDO
              ENDDO
              ENDDO

C  ---- CALCULATE Y VALUE AT GAUSS POINT BY TRIVARIATE APPROX
C  ----- IJNME PDD EQ.(29), I1<I2<I3
```

```
         DO I1 = 1,N-2
         DO I2 = I1+1,N-1
         DO I3 = I2+1,N
         DO J1 = 1,NBAS(I1)
         DO J2 = 1,NBAS(I2)
         DO J3 = 1,NBAS(I3)
              DO IG = 1, NGAUSS_GQ
              DO JG = 1, NGAUSS_GQ
              DO KG = 1, NGAUSS_GQ
                  YPDD(IG,JG,KG) = YPDD(IG,JG,KG) + BITA(I1,I2,J1,J2)
        1                  *H_GQ(I1,J1,KG)*H_GQ(I2,J2,IG)*H_GQ(I3,J3,JG)/3.

              ENDDO
              ENDDO
              ENDDO
         ENDDO
         ENDDO
         ENDDO
         ENDDO
         ENDDO
         ENDDO





C   -------- SUM ALL THE GAUSSIAN POINT FOR TRIVARIATE APPROX
C


C   SMUST = 0.
C   SSIGST = 0.
C
C
            DO IG = 1, NGAUSS_GQ
              DO JG = 1, NGAUSS_GQ
              DO KG = 1, NGAUSS_GQ
                  SMUST = SMUST + YPDD(IG,JG,KG)*SCMU(IG,JG,KG)*QW(IG)
        1              /DSQRT(PI)*QW(JG)/DSQRT(PI)*QW(KG)/DSQRT(PI)
                  SSIGST = SSIGST + YPDD(IG,JG,KG)*SCSIG(IG,JG,KG)*QW(IG)
        1              /DSQRT(PI)*QW(JG)/DSQRT(PI)*QW(KG)/DSQRT(PI)
              ENDDO
              ENDDO
              ENDDO

C      PRINT*, 'GAUSS QUADRATURE RESULT: ------------------'
C      PRINT*, 'SMUSU = '
C      WRITE (*,*) SMUSU
C      PRINT*, 'SSIGSU = '
C      WRITE (*,*) SSIGSU
C
C      PRINT*, 'SMUSB = '
C      WRITE (*,*) SMUSB
C      PRINT*, 'SSIGSB = '
C      WRITE (*,*) SSIGSB
C
C      PRINT*, 'SMUST = '
C      WRITE (*,*) SMUST
C      PRINT*, 'SSIGST = '
C      WRITE (*,*) SSIGST


C     ----------- ENDDO OF DO I=1,IMU   ----------
      ENDDO



      A = 0.
      B = 0.
      DO I = 1,N
      A(I,1:20) = XMU_IN(I)
      B(I,1:20) =(/1,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19/)*
     1    XSIG_IN(I)**2.0
      ENDDO



      AF1 = 0.
      BT1 = 0.
      GM1 = 0.
      ALFA_OUT = 0.

      DO I = 1, N
      DO K = 1, NBAS_A(I)
      AF1(I,K) = ALFA(I,K)
      ALFA_OUT(I,K) = ALFA(I,K)
      ENDDO
      ENDDO

      DO I1 = 1, N-1
      DO I2 = I1+1, N
      DO K1 = 1, NBAS_A(I1)
      DO K2 = 1, NBAS_A(I2)
      BT1(I1,I2,K1,K2) = BITA(I1,I2,K1,K2)
      ENDDO
      ENDDO
      ENDDO
      ENDDO

      DO I1 = 1, N-2
      DO I2 = I1+1, N-1
      DO I3 = I2+1, N
      DO K1 = 1, NBAS_A(I1)
      DO K2 = 1, NBAS_A(I2)
      DO K3 = 1, NBAS_A(I3)
      GM1(I1,I2,I3,K1,K2,K3) = GAMA(I1,I2,I3,K1,K2,K3)
      ENDDO
      ENDDO
      ENDDO
      ENDDO
      ENDDO
      ENDDO
```

```fortran
C      SUBROUTINE A_IJK(Y0, ALFA, BITA, GAMA, A, B, NBAS, N, AIJK)
C      CALL A_IJK(Y0, A, B, NBAS_A, N, AIJK)
C      SUBROUTINE RESPON_PDD(Y0, ALFA, BITA, GAMA, NBAS, N1, X)
      IF (IFLAG_AIJK .EQ. 0) THEN
      CALL A_IJK(Y0, AF1, BT1, GM1, A, B, NBAS_A, N, AIJK)
      AIJKC(IFLAG_Y,:,:,:,:) = AIJK
      ENDIF

C      X_TEMP  = 97.
CC      X_TEMP(1:5) = (/10,1,10000,800,1050/)
C      AIJK_TEMP = AIJKC(IFLAG_Y,:,:,:,:)
C      CALL EVAL_AIJK(NBAS_A, AIJK_TEMP,X_TEMP,Y)
C      CALL RESPON_AIJK(NBAS_A, AIJK_TEMP, X_TEMP, Y2)
C      CALL EVAL_PDD(Y0, AF1, BT1, GM1, NBAS_A, N, X_TEMP, Y_PDD)
C
C
C      PRINT*, 'IFLAG_Y = ', IFLAG_Y
C      PRINT*, 'Y = ', Y
C      PRINT*, 'Y2 = ', Y2
C      PRINT*, 'Y_PDD = ', Y_PDD
C      PRINT*, 'X_TEMP = ', X_TEMP(1:9)

      RETURN


      END
C
C       ----------------------------------------------------------
        SUBROUTINE RESPON(X,Y,IFLAG_Y)
C
C       CALCULATES Y FOR A SAMPLE OF X
C
      IMPLICIT REAL*8 (A-H,O-Z)
        REAL*8 X(30),Z(30),XMU(30),SIG(30), NMU(30),NSIG(30)
      REAL*8 Y
      INTEGER NYQ, N
        COMMON /SR02/ XMU,SIG, NMU, NSIG, NYQ, N
        COMMON /XR01/ N_F, N_GF, N_GC, N_EF,N_EC
C
C   DO I = 1,4
C   Z(I) = X(I)
C   ENDDO
      PI = 4.*DATAN(1.0D0)
      DO I =1, N
      Z(I) = NMU(I) + NSIG(I)*X(I)
      ENDDO





      IF (IFLAG_Y .EQ. 1) THEN
         Y = 0.0
         DO I =1,N
          Y = Y+Z(I)
         ENDDO
        N_EF = N_EF + 1
      ELSE IF (IFLAG_Y .EQ. 2) THEN
         Y = 0.0
         DO I =1,N
          Y = Y+Z(I)**2.0
         ENDDO
        N_EC = N_EC +1
      ELSE
      PRINT *, 'WRONG IFLAG_Y'
      STOP
      ENDIF


C
        RETURN
        END
C
C--------------------------------------------------------------
        SUBROUTINE SCOR(X,S_MU, S_SIG)
C
C       CALCULATES SCORE FUNCTION FOR A SAMPLE OF X
C
      IMPLICIT REAL*8 (A-H,O-Z)
        REAL*8 X(30),Z(30),XMU(30),SIG(30), NMU(30),NSIG(30)
      REAL*8 S_MU(200), S_SIG(200)
        INTEGER NYQ, N
        COMMON /SR02/ XMU,SIG, NMU, NSIG, NYQ, N

C     ONLY FIRST TWO VARIABLES ARE DESIGN VARIABLES, FROM 3-5 VARIABLES, NEED ADD NEW SCORE FUNCTION FOR THIS PROBLEM
      DO I=1,N
      Z(I) = NMU(I) + NSIG(I)*X(I)
      S_MU(I) =  1./NSIG(I) * ((Z(I)-NMU(I))/NSIG(I))
      S_SIG(I) =  1./NSIG(I) * (((Z(I)-NMU(I))/NSIG(I))**2.0  - 1.0)
      ENDDO

        RETURN
        END



C
C--------------------------------------------------------------
        SUBROUTINE PDF_IID(X, Y)
C
C       CALCULATES PDF VALUE FOR A SAMPLE OF X
C
      IMPLICIT REAL*8 (A-H,O-Z)
        REAL*8 X(30),Z(30),XMU(30),SIG(30), NMU(30),NSIG(30)
      REAL*8 S_MU, S_SIG
      INTEGER NYQ, N
        COMMON /SR02/ XMU,SIG, NMU, NSIG, NYQ, N

      Y = 1.
      PI = 4.*DATAN(1.0D0)
      DO I = 1,N
        Z(I) = NMU(I) + NSIG(I)*X(I)
        Y = Y * 1./(NSIG(I)*DSQRT(2*PI)) * EXP(-0.5*
     1       ((Z(I)-NMU(I))/NSIG(I))**2.0)

      ENDDO

        RETURN
        END
```

```
C
C         ------------------------------------------------------------
          SUBROUTINE BASIS_NORMAL(X,PSIF,N,NBAS)
C
C         DEFINES HERMITE POLYNOMIALS - NORMALIZED W.R.T. N(0,1) PDF
C         DO NOT INCLUDE THE 0-ORDER, BECAUSE THE 0-ORDER GIVES MEAN
C       ALREADY INCLUDED IN Y0
C
        IMPLICIT REAL*8 (A-H,O-Z)
          REAL*8 X(N),PSIF(N,NBAS)
C
          DO I = 1,N
C         IF (0 .LE. NBAS) PSIF(I,1) = 1.  ----------THIS IS WHY Y0 COME IN   ----  CAN WE AVOID CALCULATE Y0 BY INTRODUCING THIS BASIS?
          IF (1 .LE. NBAS) PSIF(I,1) = X(I)
          IF (2 .LE. NBAS) PSIF(I,2) = (X(I)**2.0 - 1.)/DSQRT(2.D0)
          IF (3 .LE. NBAS) PSIF(I,3) = (X(I)**3.0 - 3.*X(I))/
       &                             DSQRT(6.D0)
          IF (4 .LE. NBAS) PSIF(I,4) = (X(I)**4.0 - 6.*X(I)**2.0 + 3.)/
       &                             DSQRT(24.D0)
          IF (5 .LE. NBAS) PSIF(I,5) = (X(I)**5.0 - 10.*X(I)**3.0 + 15.*X(I))/
       &                             DSQRT(120.D0)
          IF (6 .LE. NBAS) PSIF(I,6) = (X(I)**6.0 - 15.*X(I)**4.0 + 45.*X(I)**2.0
       &                            -15.)/DSQRT(720.D0)
          IF (7 .LE. NBAS) PSIF(I,7) =(X(I)**7.0 - 21.*X(I)**5.0 + 105.*X(I)**3.0
       &                            -105.*X(I))/DSQRT(5040.D0)
          IF (8 .LE. NBAS) PSIF(I,8) =(X(I)**8.0 - 28.*X(I)**6.0 + 210.*X(I)**4.0
       &                      -420.*X(I)**2.0 + 105.)/DSQRT(40320.D0)
          IF (9 .LE. NBAS) PSIF(I,9) =(X(I)**9.0 - 36.*X(I)**7.0 + 378.*X(I)**5.0
       &             -1260.*X(I)**3.0 + 945.*X(I))/DSQRT(362880.D0)
          ENDDO
C
          RETURN
          END





C
C         ------------------------------------------------------------
          SUBROUTINE BASIS_LOGN(X,PSIF,N,NBAS)
C
C         DEFINES ORTHOGONAL POLYNOMIALS - LOG-NORMAL  PDF
C         DO NOT INCLUDE THE 0-ORDER, BECAUSE THE 0-ORDER GIVES MEAN
C       ALREADY INCLUDED IN Y0
C

C    -------------  JUST FOR MEAN = 1050, STD = 250  --------
C    ------ OTHER CASE NEED USING STIELTJES PROCESS RECALCULATE A AND B


        IMPLICIT REAL*8 (A-H,O-Z)
          REAL*8 X(N),PSIF(N,NBAS), B(10), A(10)
C


        DATA A/1.0500000000E+003, 1.2319457942E+003, 1.4384794447E+003,
       1        1.6726589984E+003, 1.9379098914E+003, 2.2380683981E+003,
       1        2.5774301819E+003, 2.9608045427E+003, 3.3935750281E+003,
       2        3.8817671531E+003/

        DATA B/1.0, 6.2500000000E+004, 1.5166686037E+005,
       1        2.7610394319E+005, 4.4690157519E+005, 6.7831622879E+005,
       1        9.8862908461E+005, 1.4012303609E+006, 1.9459876237E+006,
       2        2.6609711061E+006/




          DO I = 1,N
C         IF (0 .LE. NBAS) PSIF(I,1) = 1.  ----------THIS IS WHY Y0 COME IN
          IF (1 .LE. NBAS) PSIF(I,1) = (X(I) - A(1))/DSQRT(1.0D0*B(2))
          IF (2 .LE. NBAS) PSIF(I,2) = (PSIF(I,1)*(X(I) - A(2))
       1     -DSQRT(1.0D0*B(2)))/DSQRT(1.0D0*B(3))
          IF (3 .LE. NBAS) PSIF(I,3) = ((X(I)-A(3))*PSIF(I,2)-DSQRT(1.0D0*B(3))
       1     *PSIF(I,1))/DSQRT(1.0D0*B(4))

          IF (4 .LE. NBAS) PSIF(I,4) = ((X(I)-A(4))*PSIF(I,3)-DSQRT(1.0D0*B(4))
       1      *PSIF(I,2))/DSQRT(1.0D0*B(5))

          IF (5 .LE. NBAS) PSIF(I,5) = ((X(I)-A(5))*PSIF(I,4)-DSQRT(1.0D0*B(5))
       1      *PSIF(I,3))/DSQRT(1.0D0*B(6))
          IF (6 .LE. NBAS) PSIF(I,6) = ((X(I)-A(6))*PSIF(I,5)-DSQRT(1.0D0*B(6))
       1      *PSIF(I,4))/DSQRT(1.0D0*B(7))
          IF (7 .LE. NBAS) PSIF(I,7) = ((X(I)-A(7))*PSIF(I,6)-DSQRT(1.0D0*B(7))
       1      *PSIF(I,5))/DSQRT(1.0D0*B(8))
          IF (8 .LE. NBAS) PSIF(I,8) = ((X(I)-A(8))*PSIF(I,7)-DSQRT(1.0D0*B(8))
       1      *PSIF(I,6))/DSQRT(1.0D0*B(9))
          IF (9 .LE. NBAS) PSIF(I,9) = ((X(I)-A(9))*PSIF(I,8)-DSQRT(1.0D0*B(9))
       1      *PSIF(I,7))/DSQRT(1.0D0*B(10))
          ENDDO
C
          RETURN
          END



          SUBROUTINE DGQRUL_LOGN (NGAUSS,QX,QW)
C
C         CALCULATE GAUSS QUADRATURE RULE FOR LOG-NORMAL  PDF (WEIGHT)
C
C
C

C    -------------  JUST FOR MEAN = 1050, STD = 250  --------
C    ------ OTHER CASE NEED USING STIELTJES PROCESS RECALCULATE A AND B


        IMPLICIT REAL*8 (A-H,O-Z)
          REAL*8 B(10), A(10), QX(50), QW(50)
C
```

```fortran
      IF (NGAUSS .EQ. 1) THEN
        QX(1) =  1.0500000000E+003
        QW(1) =  1.0D0
      ELSEIF (NGAUSS .EQ. 2) THEN
        QX(1) =  8.7493517795E+002
        QW(1) =  6.7097744145E-001
        QX(2) =  1.4070106162E+003
        QW(2) =  3.2902255855E-001
      ELSEIF (NGAUSS .EQ. 3) THEN
        QX(1) =  7.7549702759E+002
        QW(1) =  3.8144102839E-001
        QX(2) =  1.1724219847E+003
        QW(2) =  5.7026328722E-001
        QX(3) =  1.7725062266E+003
        QW(3) =  4.8295684388E-002
      ELSEIF (NGAUSS .EQ. 4) THEN
        QX(1) =  7.0792306393E+002
        QW(1) =  2.1110467491E-001
        QX(2) =  1.0370001864E+003
        QW(2) =  6.0928607207E-001
        QX(3) =  1.4800750137E+003
        QW(3) =  1.7540991032E-001
        QX(4) =  2.1680859733E+003
        QW(4) =  4.1993426996E-003

      ELSEIF (NGAUSS .EQ. 5) THEN
        QX(1) =  6.5789978335E+002
        QW(1) =  1.1819942906E-001
        QX(2) =  9.4445049975E+002
        QW(2) =  5.3996393645E-001
        QX(3) =  1.3091174382E+003
        QW(3) =  3.1300785250E-001
        QX(4) =  1.8145879191E+003
        QW(4) =  2.8587099274E-002
        QX(5) =  2.6049384882E+003
        QW(5) =  2.4168271864E-004

      ELSE

      PRINT *, 'NGAUSS TOO BIG OR NEGATIVE! '

      STOP




      ENDIF



C
        RETURN
        END

C
C       -------------------------------------------------------------
        SUBROUTINE BASIS_GUMBEL(X,PSIF,N,NBAS)
C
C       DEFINES ORTHOGONAL POLYNOMIALS - GUMBEL  PDF
C       DO NOT INCLUDE THE 0-ORDER, BECAUSE THE 0-ORDER GIVES MEAN
C     ALREADY INCLUDED IN Y0
C

C   -------------  JUST FOR MEAN = 800, STD = 200  -------- MODIFY TO SUITABLE FOR STANDARD GUMBEL
C   ------ OTHER CASE NEED USING STIELTJES PROCESS RECALCULATE A AND B


      IMPLICIT REAL*8 (A-H,O-Z)
        REAL*8 X(N),PSIF(N,NBAS), B(10), A(10)
C

      DATA A/5.7721566490E-001, 2.0387416037E+000, 3.7776788547E+000,
     1       5.6086837306E+000, 7.4827936017E+000, 9.3823342409E+000,
     1       1.1298739237E+001, 1.3227148571E+001, 1.5164540892E+001,
     2       1.7108910081E+001/



      DATA B/ 1.0, 1.6449340668E+000, 5.1016518243E+000,
     1       1.0595458980E+001, 1.8091908282E+001, 2.7590004789E+001,
     1       3.9088817747E+001, 5.2587984894E+001, 6.8087369314E+001,
     2       8.5586897909E+001/





C      DATA A/8.0000000000E+002, 1.0279094199E+003, 1.2990781823E+003,
C     1        1.5846039113E+003, 1.8768514056E+003, 2.1730645576E+003,
C     1        2.4719075267E+003, 2.7726224445E+003, 3.0747381636E+003,
C     2        3.3779418511E+003/
C
C
C
C
C
C      DATA B/ 1.0, 4.0000000000E+004, 1.2405729633E+005,
C     1         2.5765066682E+005, 4.3994245475E+005, 6.7090846605E+005,
C     1         9.5052606752E+005, 1.2787864500E+006, 1.6556862840E+006,
C     2         2.0812237921E+006/
C


        DO I = 1,N
C       IF (0 .LE. NBAS) PSIF(I,1) = 1.  ----------THIS IS WHY Y0 COME IN
        IF (1 .LE. NBAS) PSIF(I,1) = (X(I) - A(1))/DSQRT(1.0D0*B(2))
        IF (2 .LE. NBAS) PSIF(I,2) = (PSIF(I,1)*(X(I) - A(2))
     1    -DSQRT(1.0D0*B(2)))/DSQRT(1.0D0*B(3))
        IF (3 .LE. NBAS) PSIF(I,3) = ((X(I)-A(3))*PSIF(I,2)-DSQRT(1.0D0*B(3))
     1    *PSIF(I,1))/DSQRT(1.0D0*B(4))

        IF (4 .LE. NBAS) PSIF(I,4) = ((X(I)-A(4))*PSIF(I,3)-DSQRT(1.0D0*B(4))
     1     *PSIF(I,2))/DSQRT(1.0D0*B(5))
```

```
        IF (5 .LE. NBAS) PSIF(I,5) = ((X(I)-A(5))*PSIF(I,4)-DSQRT(1.0D0*B(5))
     1       *PSIF(I,3))/DSQRT(1.0D0*B(6))
        IF (6 .LE. NBAS) PSIF(I,6) = ((X(I)-A(6))*PSIF(I,5)-DSQRT(1.0D0*B(6))
     1       *PSIF(I,4))/DSQRT(1.0D0*B(7))
        IF (7 .LE. NBAS) PSIF(I,7) = ((X(I)-A(7))*PSIF(I,6)-DSQRT(1.0D0*B(7))
     1       *PSIF(I,5))/DSQRT(1.0D0*B(8))
        IF (8 .LE. NBAS) PSIF(I,8) = ((X(I)-A(8))*PSIF(I,7)-DSQRT(1.0D0*B(8))
     1       *PSIF(I,6))/DSQRT(1.0D0*B(9))
        IF (9 .LE. NBAS) PSIF(I,9) = ((X(I)-A(9))*PSIF(I,8)-DSQRT(1.0D0*B(9))
     1       *PSIF(I,7))/DSQRT(1.0D0*B(10))
        ENDDO
C
        RETURN
        END


        SUBROUTINE DGQRUL_GUMBEL (NGAUSS,QX,QW)
C
C       CALCULATE GAUSS QUADRATURE RULE FOR GUMBEL  PDF (WEIGHT)
C
C
C


C   -------------  JUST FOR MEAN = 800, STD = 200  ---------------- MODIFY TO SUITABLE FOR STANDARD GUMBEL
C   ------ OTHER CASE NEED USING STIELTJES PROCESS RECALCULATE A AND B


      IMPLICIT REAL*8 (A-H,O-Z)
        REAL*8 B(10), A(10), QX(50),QW(50)
C




      IF (NGAUSS .EQ. 1) THEN
        QX(1) =  5.7721566490E-001
        QW(1) =  1.0D0
      ELSEIF (NGAUSS .EQ. 2) THEN
        QX(1) =  -1.6814757449E-001
        QW(1) =   7.4752726598E-001
        QX(2) =  2.7841048431E+000
        QW(2) =  2.5247273402E-001
      ELSEIF (NGAUSS .EQ. 3) THEN
        QX(1) =  -5.6941560564E-001
        QW(1) =  4.9626301710E-001
        QX(2) =  1.5209907666E+000
        QW(2) =  4.7986179911E-001
        QX(3) =  5.4420609624E+000
        QW(3) =  2.3875183786E-002
      ELSEIF (NGAUSS .EQ. 4) THEN
        QX(1) =  -8.3881616518E-001
        QW(1) =  3.2239218571E-001
        QX(2) =  8.4850412694E-001
        QW(2) =  5.8073605368E-001
        QX(3) =  3.5914517036E+000
        QW(3) =  9.5421314087E-002
        QX(4) =  8.4011801885E+000
        QW(4) =  1.4504465174E-003

      ELSEIF (NGAUSS .EQ. 5) THEN
        QX(1) =  -1.0394422129E+000
        QW(1) =  2.0981305071E-001
        QX(2) =  4.0696152279E-001
        QW(2) =  5.8838291141E-001
        QX(3) =  2.5724705542E+000
        QW(3) =  1.9122518250E-001
        QX(4) =  6.0068251043E+000
        QW(4) =  1.0508618090E-002
        QX(5) =  1.1538298487E+001
        QW(5) =  7.0237293503E-005

      ELSE

      PRINT *, 'NGAUSS TOO BIG OR NEGATIVE! '

      STOP




      ENDIF



C
        RETURN
        END



C
C       -----------------------------------------------------------
        SUBROUTINE BASIS_BETA(X,PSIF,N,NBAS)
C
C       DEFINES ORTHOGONAL POLYNOMIALS - BETA PDF
C       DO NOT INCLUDE THE 0-ORDER, BECAUSE THE 0-ORDER GIVES MEAN
C     ALREADY INCLUDED IN Y0
C

C   -------------  JUST FOR ALFA = 5, BETA = 5;  0<= X <=1 --------
C   ------ OTHER CASE NEED USING STIELTJES PROCESS RECALCULATE A AND B


      IMPLICIT REAL*8 (A-H,O-Z)
        REAL*8 X(N),PSIF(N,NBAS), B(10), A(10)
C



      DATA A/0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5/


      DATA B/ 1.0, 2.2727272727E-002, 3.4965034965E-002,
     1      4.2307692308E-002, 4.7058823529E-002, 5.0309597523E-002,
     2      5.2631578947E-002, 5.4347826087E-002, 5.5652173913E-002,
     3      5.6666666667E-002/




        DO I = 1,N
C       IF (0 .LE. NBAS) PSIF(I,1) = 1.  ----------THIS IS WHY Y0 COME IN
        IF (1 .LE. NBAS) PSIF(I,1) = (X(I) - A(1))/DSQRT(1.0D0*B(2))
        IF (2 .LE. NBAS) PSIF(I,2) = (PSIF(I,1)*(X(I) - A(2))
     1      -DSQRT(1.0D0*B(2)))/DSQRT(1.0D0*B(3))
        IF (3 .LE. NBAS) PSIF(I,3) = ((X(I)-A(3))*PSIF(I,2)-DSQRT(1.0D0*B(3))
     1      *PSIF(I,1))/DSQRT(1.0D0*B(4))

        IF (4 .LE. NBAS) PSIF(I,4) = ((X(I)-A(4))*PSIF(I,3)-DSQRT(1.0D0*B(4))
     1       *PSIF(I,2))/DSQRT(1.0D0*B(5))

        IF (5 .LE. NBAS) PSIF(I,5) = ((X(I)-A(5))*PSIF(I,4)-DSQRT(1.0D0*B(5))
     1       *PSIF(I,3))/DSQRT(1.0D0*B(6))
        IF (6 .LE. NBAS) PSIF(I,6) = ((X(I)-A(6))*PSIF(I,5)-DSQRT(1.0D0*B(6))
     1       *PSIF(I,4))/DSQRT(1.0D0*B(7))
        IF (7 .LE. NBAS) PSIF(I,7) = ((X(I)-A(7))*PSIF(I,6)-DSQRT(1.0D0*B(7))
     1       *PSIF(I,5))/DSQRT(1.0D0*B(8))
```

```fortran
     1      *PSIF(I,5))/DSQRT(1.0D0*B(8))
        IF (8 .LE. NBAS) PSIF(I,8) = ((X(I)-A(8))*PSIF(I,7)-DSQRT(1.0D0*B(8))
     1      *PSIF(I,6))/DSQRT(1.0D0*B(9))
        IF (9 .LE. NBAS) PSIF(I,9) = ((X(I)-A(9))*PSIF(I,8)-DSQRT(1.0D0*B(9))
     1      *PSIF(I,7))/DSQRT(1.0D0*B(10))
        ENDDO
C
        RETURN
        END


        SUBROUTINE DGQRUL_BETA (NGAUSS,QX,QW)
C
C       CALCULATE GAUSS QUADRATURE RULE FOR GUMBEL  PDF (WEIGHT)
C
C
C

C   -------------  JUST FOR ALFA = 5, BETA = 5;  0<= X <=1 --------
C   ------ OTHER CASE NEED USING STIELTJES PROCESS RECALCULATE A AND B


      IMPLICIT REAL*8 (A-H,O-Z)
        REAL*8 B(10), A(10), QX(50),QW(50)
C




      IF (NGAUSS .EQ. 1) THEN
        QX(1) =  0.5
        QW(1) =  1.0D0
      ELSEIF (NGAUSS .EQ. 2) THEN
        QX(1) =  3.4924432771E-001
        QW(1) =  5.0000000000E-001
        QX(2) =  6.5075567229E-001
        QW(2) =  5.0000000000E-001
      ELSEIF (NGAUSS .EQ. 3) THEN
        QX(1) =  2.5980776929E-001
        QW(1) =  1.9696969697E-001
        QX(2) =  5.0000000000E-001
        QW(2) =  6.0606060606E-001
        QX(3) =  7.4019223071E-001
        QW(3) =  1.9696969697E-001
      ELSEIF (NGAUSS .EQ. 4) THEN
        QX(1) =  2.0129742715E-001
        QW(1) =  7.6169789309E-002
        QX(2) =  3.9618876278E-001
        QW(2) =  4.2383021069E-001
        QX(3) =  6.0381123722E-001
        QW(3) =  4.2383021069E-001
        QX(4) =  7.9870257285E-001
        QW(4) =  7.6169789309E-002

      ELSEIF (NGAUSS .EQ. 5) THEN
        QX(1) =  1.6070082773E-001
        QW(1) =  3.0562430134E-002
        QX(2) =  3.2129662783E-001
        QW(2) =  2.4566134609E-001
        QX(3) =  5.0000000000E-001
        QW(3) =  4.4755244755E-001
        QX(4) =  6.7870337217E-001
        QW(4) =  2.4566134609E-001
        QX(5) =  8.3929917227E-001
        QW(5) =  3.0562430134E-002

      ELSE

      PRINT *, 'NGAUSS TOO BIG OR NEGATIVE! '

      STOP





      ENDIF



C
        RETURN
        END




C      MODULE GETAIJK
C      CONTAINS
      SUBROUTINE A_IJK(Y0, ALFA, BITA, GAMA, A, B, NBAS, N, AIJK)
c     1 IFLAG_DIST)
C      SUBROUTINE A_IJK( A, B, NBAS, N, AIJK)
C     SUBROUTINE TO GET COEFICIENTS A_IJK BY PDD COEFs
c     A, B ARE THREE TERMS RECURSION COEFs
C     AIJK X1^I*X2^J*X3^J

      USE UMACH_INT
      USE GAMMA_INT
      REAL*8 Y0, ALFA(25,5),BITA(25,25,5,5),GAMA(25,25,25,5,5,5)
C      REAL*8 Y0
C      REAL, POINTER ::  ALFA(:,:),BITA(:,:,:,:),GAMA(:,:,:,:,:,:)
      REAL*8 A(30,30),B(30,30), AIJK(25,25,5,5)
      REAL*8 PHI(30,20,20)
      INTEGER NBAS(30), N
C      ALLOCATABLE ALFA, BITA, GAMA

C      COMMON /XR03/ Y0, ALFA, BITA, GAMA



      PHI = 0.
      AIJK = 0.
```

```
      DO I = 1, N
      DO J = 1, NBAS(I)
        IF (J .EQ. 1) THEN
            PHI(I,J,1) =  -A(I,J)/DSQRT(1.0D0*B(I,J+1))
            PHI(I,J,2) =  1.0/DSQRT(1.0D0*B(I,J+1))
        ELSEIF (J .EQ. 2) THEN
            PHI(I,J,1) =  (-A(I,J)*PHI(I,J-1,1) -DSQRT(1.0D0*B(I,J)))
     1       /DSQRT(1.0D0*B(I,J+1))
            PHI(I,J,2) =  (PHI(I,J-1,1)-A(I,J)*PHI(I,J-1,2))
     1       /DSQRT(1.0D0*B(I,J+1))
            PHI(I,J,3) =  (PHI(I,J-1,2))
     1       /DSQRT(1.0D0*B(I,J+1))
        ELSE
           PHI(I,J,1) =  (-DSQRT(1.0D0*B(I,J))*PHI(I,J-2,1) -
     1      A(I,J)*PHI(I,J-1,1))/DSQRT(1.0D0*B(I,J+1))
           DO K = 2, J+1
           PHI(I,J,K) =  (-DSQRT(1.0D0*B(I,J))*PHI(I,J-2,K) -
     1      A(I,J)*PHI(I,J-1,K)+ PHI(I,J-1,K-1))/DSQRT(1.0D0*B(I,J+1))
           ENDDO
       ENDIF

      ENDDO
      ENDDO

C     TO DO :  CONSTANT TERM,  1ST ORDER TERM, 2ND ORDER TERM, ....
      AIJK(1,1,1,1) = Y0;
      DO I = 1, N
      DO J = 1, NBAS(I)
        AIJK(1,1,1,1) = AIJK(1,1,1,1) +
     1    ALFA(I,J)*PHI(I,J,1)
        DO K =2, NBAS(I)+1
        AIJK(I,1,K,1) = AIJK(I,1,K,1) + ALFA(I,J)*PHI(I,J,K)
        ENDDO

      ENDDO
      ENDDO




      DO I1 = 1, N-1
      DO I2 = I1+1, N
      DO J1 = 1, NBAS(I1)
      DO J2 = 1, NBAS(I2)
        AIJK(1,1,1,1) = AIJK(1,1,1,1) +
     1    BITA(I1,I2,J1,J2)*PHI(I1,J1,1)*PHI(I2,J2,1)

        DO I =1, N
        DO K =2, NBAS(I)+1
        IF (I1.EQ.I) THEN
        AIJK(I,1,K,1) = AIJK(I,1,K,1) +
     1   BITA(I1,I2,J1,J2)*(PHI(I1,J1,K)*PHI(I2,J2,1))
        ELSEIF (I2.EQ. I) THEN
        AIJK(I,1,K,1) = AIJK(I,1,K,1) +
     1   BITA(I1,I2,J1,J2)*(PHI(I1,J1,1)*PHI(I2,J2,K))
        ENDIF

        ENDDO
        ENDDO


        DO K1 =2, NBAS(I1)+1
        DO K2 =2, NBAS(I2)+1
        AIJK(I1,I2,K1,K2) = AIJK(I1,I2,K1,K2) +
     1   BITA(I1,I2,J1,J2)*PHI(I1,J1,K1)*PHI(I2,J2,K2)
        ENDDO
        ENDDO


      ENDDO
      ENDDO
      ENDDO
      ENDDO




      RETURN
      END

C       END MODULE GETAIJK

      SUBROUTINE EVAL_AIJK(NBAS, AIJK, X, Y)
C     SUBROUTINE TO EVALUATION FUNCTION ACCORDING THE COEFFICIENT A_IJK
C     AIJK X1^I*X2^J*X3^J

      REAL*8 AIJK(25,25,5,5)
      INTEGER NBAS(30)
C      REAL*8 X(20), Y, Z(20)
      REAL*8 X(30),Z(30),XMU(30),SIG(30), NMU(30),NSIG(30), Y
        COMMON /SR02/ XMU,SIG, NMU, NSIG, NYQ, N



       Z = X



      Y = AIJK(1,1,1,1)

      DO I = 1,N
      DO K = 2,NBAS(I)+1
      Y = Y + AIJK(I,1,K,1)*Z(I)**(K-1)
      ENDDO
      ENDDO

      DO I1 = 1,N-1
      DO I2 = I1+1,N
      DO K1 = 2,NBAS(I1)+1
      DO K2 = 2,NBAS(I2)+1
      Y = Y + AIJK(I1,I2,K1,K2)*Z(I1)**(K1-1)*Z(I2)**(K2-1)
      ENDDO
      ENDDO
      ENDDO
      ENDDO



      RETURN
```

```
      END


      SUBROUTINE RESPON_AIJK(NBAS, AIJK, X, Y)
C     SUBROUTINE TO EVALUATION FUNCTION ACCORDING THE COEFFICIENT A_IJK
C     AIJK X1^I*X2^J*X3^J

      REAL*8 AIJK(25,25,5,5)
      INTEGER NBAS(30)
C      REAL*8 X(20), Y, Z(20)
      REAL*8 X(30),Z(30),XMU(30),SIG(30), NMU(30),NSIG(30), Y
        COMMON /SR02/ XMU,SIG, NMU, NSIG, NYQ, N


      Z = NMU + NSIG*X

      Y = AIJK(1,1,1,1)

      DO I = 1,N
      DO K = 2,NBAS(I)+1
      Y = Y + AIJK(I,1,K,1)*Z(I)**(K-1)
      ENDDO
      ENDDO

      DO I1 = 1,N-1
      DO I2 = I1+1,N
      DO K1 = 2,NBAS(I1)+1
      DO K2 = 2,NBAS(I2)+1
      Y = Y + AIJK(I1,I2,K1,K2)*Z(I1)**(K1-1)*Z(I2)**(K2-1)
      ENDDO
      ENDDO
      ENDDO
      ENDDO



      RETURN
      END


C      MODULE GETAIJK
C      CONTAINS
      SUBROUTINE EVAL_PDD(Y0, ALFA, BITA, GAMA, NBAS, N1, X,Y)
c     1 IFLAG_DIST)
C      SUBROUTINE EVAL_PDD(Y0, ALFA, BITA, GAMA, NBAS, N)
C     SUBROUTINE TO GET FUNCTION VALUE BY PDD APPROXIMATION


      USE UMACH_INT
      USE GAMMA_INT
      REAL*8 Y0, ALFA(25,5),BITA(25,25,5,5),GAMA(25,25,25,5,5,5)
      REAL*8  H(30,20), PSIF(30,30),Z(30), XMU(30),SIG(30),
     1   NMU(30),NSIG(30), Y, X(30)
      INTEGER NBAS(30), N1

C      COMMON /XR03/ Y0, ALFA, BITA, GAMA
      COMMON /SR02/ XMU,SIG, NMU, NSIG, NYQ, N

       Z = X


      Z(1) =  (X(1)- NMU(1))/NSIG(1)
      Z(2) =  (X(2)- NMU(2))/NSIG(2)
      Z =  (X- NMU)/NSIG





C
C       CALCULATE/STORE BASIS FUNCTIONS AT GAUSS POINTS (H)


        DO I = 1,N
         DO J = 1,NBAS(I)

         CALL BASIS_NORMAL (Z,PSIF,N,NBAS(I))
         H(I,J) = PSIF(I,J)

         ENDDO
        ENDDO


      Y = Y0

      DO I = 1,N
      DO J = 1,NBAS(I)
      Y = Y + ALFA(I,J)*H(I,J)
      ENDDO
      ENDDO

      DO I1 = 1,N-1
      DO I2 = I1+1,N
      DO J1 = 1,NBAS(I1)
      DO J2 = 1,NBAS(I2)
      Y = Y + BITA(I1,I2,J1,J2)*H(I1,J1)*H(I2,J2)
      ENDDO
      ENDDO
      ENDDO
      ENDDO


      DO I1 = 1,N-2
      DO I2 = I1+1,N-1
      DO I3 = I2+1,N
      DO J1 = 1,NBAS(I1)
      DO J2 = 1,NBAS(I2)
      DO J3 = 1,NBAS(I3)
      Y = Y + GAMA(I1,I2,I3,J1,J2,J3)*H(I1,J1)*H(I2,J2)*H(I3,J3)
      ENDDO
      ENDDO
      ENDDO
      ENDDO
      ENDDO
      ENDDO




      RETURN
      END

C      END MODULE GETAIJK
```