```
      DOUBLE PRECISION FUNCTION fch(t,y)
      IMPLICIT DOUBLE PRECISION (a-h,o-z)
      DOUBLE PRECISION y(nuu),sqe(48)
      LOGICAL log1
      SAVE fchcr
      COMMON /SCALE/ amgut, susy, amz, amw
      COMMON /INTZER/ log1  !  Initializes fchcr
      COMMON /TRMIN/ vtrue,pb(2)
      COMMON /ZIPER/ izip(48)
      COMMON /NUEQ/ nuu
      DATA fchcr /1.0d20/
CC
CC
       q=amgut*dexp(-t/2.d0)
       tinc=2.d0*dlog(amgut/amz) - t
C
CC
       call vborn(91,.false.,y,y(32),y(33),veff0)
       call dv1(91,.false.,y,0.d0,0.d0,t,wmeg)
C
C      call dv1(91,.true.,y,y(32),y(33),t,veff1)  ! Log printing
       call dv1(91,.false.,y,y(32),y(33),t,veff1) ! No log printing
C
       vsmal=wmeg+vtrue
       fch=veff1 / (veff0-vsmal) + 1.d0
       if(fch.gt.0.d0)then
         if(log1 .or. fch.lt.fchcr)then
           fchcr=fch
           log1=.false.
         endif
       endif
C
C
C   All large numbers are rescaled with 1.d23
C
C      write(91,*)'   '
C      write(91,*)'Q = ',q,'G_cr = ',fchcr
C      write(91,'("Tinc = ",f21.16,"   G = ",f21.16)')tinc,fch
C      write(91,'("v0-w-vtr = ",f21.16)')(veff0-vsmal)/1.d23
C      write(91,'("veff0-wmeg = ",f21.16)')(veff0-wmeg)/1.d23
C      write(91,'("veff1 = ",f21.16)')veff1 / 1.d23
C      write(91,'("(v0-w)/veff1 = ",f15.7)')(veff0-wmeg)/veff1
C      write(91,'("VFULL = ",f21.16/)')(veff0-wmeg+veff1)/1.d23
C
C   Files For Gnuplot
C
       call fdmass(y,y(32),y(33),sqe)
       con=fcond(t,sqe)
       avn=0.d0
       do   ig=1,48
           if(izip(ig).ne.0.d0)then
             if(sqe(ig).ne.0.d0)avn=avn+1.d0
           endif
       enddo
C       print*,avn
C       Pause
       call round(y,y(32),y(33),t,r1)
CC
       write(90,*)tinc,fch
       write(89,*)tinc,(veff0-vsmal)/1.d23, veff1/1.d23
       write(88,*)tinc, dabs( veff1/(veff0-wmeg) )
       write(87,*)tinc,r1
```



```
       write(86,*)tinc, dsqrt( con/avn )
CC
       return
       END
```

```
      DOUBLE PRECISION FUNCTION fcond(t,sqe)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DOUBLE PRECISION sqe(48)
      COMMON /SCALE/amgut, susy, amz, amw
      COMMON /ZIPER/izip(48)
CC
       con=0.d0
       do   k1=1,48
        if(izip(k1).ne.0.d0)then
C
            if(sqe(k1).ne.0.d0)then
               pscon=(2.d0*dlog(dsqrt(dabs(sqe(k1))))+T
     #         -2.d0*dlog(amgut))**2
               con=con+pscon
            endif
C
        endif
       enddo
C
C
       fcond=con
       return
       END
```

```
      SUBROUTINE FUNCT(T,Y,YPRIME)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DOUBLE PRECISION Y(NUU), YPRIME(NUU)
      COMMON/IZO/IZF,IZR
      COMMON/NUEQ/NUU
CC
      LR=IZF
      LRX=IZR
CC
      DO   J=1,NUU
         YPRIME(J)=
     #   FAP(LR,LRX,J,T,Y(1),Y(2),Y(3),Y(4),Y(5),Y(6),Y(7),Y(8),Y(9),
     #   Y(10),Y(11),Y(12),Y(13),Y(14),Y(15),Y(16),Y(17),Y(18),Y(19),
     #   Y(20),Y(21),Y(22),Y(23),Y(24),Y(25),Y(26),Y(27),Y(28),Y(29),
     #   Y(30),Y(31),Y(32),Y(33))
      ENDDO
CC
C-------------------------
       call vaver(t,y)
C-------------------------
      RETURN
       END
```

```
*==> global.f <==
************************************************************************
*                                                                      *
*  FILES                                                               *
*                                                                      *
*  evol.rg   :   Output numbers from routine out.f                     *
*  susyin    :   I.C. at Mgut                                          *
*  log.true  :   Useful quantities (logs etc) at True min.             *
*  true_min  :   True minimum configuration                            *
*  sigma     :   Tr(s) , Tr(|s|) one loop corrections to m1, m2        *
*                                                                      *
*                                                                      *
*  COMMON /RENTR/    :  Renormalization scale for TRUE min. (fixed)    *
*                       trn,qsl is the scale which appears explicitly  *
*                       in logarithms. This common is initialized      *
*                       in block data subpr.                           *
*  COMMON /SOFCOM/   :  Soft parameters and "corrected" vevs at Qrn    *
*  COMMON /NIT/      :  Communicate I.C. to evalsoft routine.          *
*  COMMON /QUEUE/    : Pass the value of logical lpr into function ftr2*
*                      where the effective potential is calculated.    *
*                      (lpr controls printing facilities)              *
*  COMMON /TRMIN/    : True minimum                                    *
*                                                                      *
*                                                                      *
*  idata     :   # of data points                                      *
*  ao        :   A_0                                                   *
*  amoo      :   M_0                                                   *
*  am12      :   M_1/2                                                 *
*  efapto    :   tanb                                                  *
*  ttop      :   Pole mass of top quark                                *
*  reject    :   Rejecting unphysical sitations (Logical)              *
*                                                                      *
*                                                                      *
*           1   mu > 0                                                 *
*         /                                                            *
*   j ---                                                              *
*         \                                                            *
*           2   mu < 0                                                 *
*                                                                      *
*             1   No thres.                                            *
*           /                                                          *
*   nai ---                                                            *
*           \                                                          *
*             2   Thresholds                                           *
*                                                                      *
*   MODULES                                                            *
*                                                                      *
*  initcnd  -->  Prints I.C. at Mgut in all files                      *
*  signmu   -->  Prints headers for mu>0 or mu<0                       *
*  xbrs     -->  Prints the renormal. scale for TRUE min.              *
*                Mgut is computed inside evalsoft so we have to call   *
*                xbrs after calling evalsoft.                          *
************************************************************************
      PROGRAM global
      IMPLICIT DOUBLE PRECISION (a-h,o-z)
      PARAMETER(idata=6,NU=39)
      DOUBLE PRECISION xcom(NU),vev(2),ftr2
      DOUBLE PRECISION hyps(NU)
      LOGICAL reject,lpr
      COMMON /SOFCOM/ xcom
      COMMON /RENTR/ trn,qsl
      COMMON /NIT/ ao,amoo,am12
```



```
      COMMON /SCALE/ amgut,susy,amz,amw
      COMMON /TRMIN/ yb,pb(2)
      COMMON /QUEUE/ lpr
      EXTERNAL ftr2
      DATA reject /.false./, naim /1/   !  naim always 1 due to PZZ corr.
      DATA nai,ttop /1, 174.d0/
      DATA kun,temp,it,stval /97, 0.d0, 80, 100.d0/
C-------------------------------------------------
C
C   Evolution output file
C
C     open(57,file='evol.rg')
C-------------------------------------------------
C      + is for mu>0  and - is for mu<0
C         write headings for ast.out
C-------------------------------------------------
C     OPEN(14,FILE='TAU81')
C     OPEN(21,FILE='RADY81')
C     OPEN(13,FILE='BEMU81')
C     OPEN(17,FILE='LOG81')
C     open(76,file='EXP')
C     OPEN(11,FILE='susyout')
C     OPEN(25,FILE='ast+.out')
C     OPEN(23,FILE='LLT+')
C     OPEN(26,FILE='ast-.out')
C     OPEN(27,FILE='LLT-')
C     OPEN(45,FILE='sin^2+')
C     OPEN(46,FILE='sin^2-')
C     OPEN(47,FILE='bounds+')
C     OPEN(48,FILE='bounds-')
C
      open(20,file='SUSY')   !  Write only in the 3-rd loop
      open(22,file='SIN')    !             >>
      open(24,file='susyin')
      open(67,file='data')
      open(98,file='log_true')
      open(97,file='true_min')
      open(96,file='pass_tests')  !  Write for all loops
      open(95,file='sigma')
      open(94,file='num.irr')
      open(93,file='log.irr')
      open(92,file='caf.out')
      open(91,file='chf.out')
C-----------------------------------------
C   Gnuplot Files
C-----------------------------------------
      open(90,file='ratio')
      open(89,file='sing')
      open(88,file='pertur')
      open(87,file='roundoff')
      open(86,file='avlog')
C
C     write(25,*)'#Ao   Mo      M1/2    as(M_z)full  mu>0'
C     write(26,*)'#Ao   Mo      M1/2    as(M_z)full  mu<0'
C     write(23,*)'#Ao   Mo      M1/2    LLT       mu>0'
C     write(27,*)'#Ao   Mo      M1/2    LLT       mu<0'
C     write(45,*)'#Ao   Mo      M1/2    sin^2(MSbar)       mu>0'
C     write(46,*)'#Ao   Mo      M1/2    sin^2(MSbar)       mu<0'
C
C---------------------------------
C   Header in SUSY file
C---------------------------------
```

```
      call hdiu20(NAI,NAIM)
C-----------------------------------------------------
C   Renormalization scale (explicit) is initialized
C   in block data subpr
C-----------------------------------------------------
      do   kdata = 1,idata
          read(24,*)ao,amoo,am12,efapto
          call initcnd(ao,amoo,am12,efapto)
          do  j = 1,2
             call signmu(j)
C------------------------------------------------------
             call string(j,efapto,TTOP,hyps,nai,reject)
C------------------------------------------------------
             call evalsoft(j,efapto,TTOP,xcom,nai,reject)
             if(reject)then
               reject=.false.
             else
               trn=2.d0*dlog(amgut/qsl)
               call xbrs(97,trn)
C---------------------------------------------
C   Finding TRUE minimum value from iterations
C   Use automatic format for comparison reasons.
C---------------------------------------------
               vev(1) = xcom(32)
               vev(2) = xcom(33)
               write(97,'(//,5X,"From iterations True min. at :")')
               write(97,*)' v1 = ',xcom(32)
               write(97,*)' v2 = ',xcom(33)
               write(97,*)' Veff = ',ftr1(vev)
               write(97,'(///)')
C-------------------------------------------------
C   Finding TRUE minimum value with numerical tools
C-------------------------------------------------
               lpr=.false.   !  *** No printing from xamebsa
               call xamebsa(kun,temp,it,ftr2,stval,pb,yb)
               lpr=.true.    !  *** Log check
               zs=ftr2(pb)
C              temptr=1.d6
C              iiter=16
C              call xamebsa(temptr,iiter)
             endif
          enddo
          call sepmu
      enddo
      END

*==> hdiu20.f <==
************************************************************************
*    Calculates both with thresholds and without them. The point is    *
*    that in the function F(I,II,J,..   ) :                            *
*    I=1, II=1  NO THRESHOLDS AT ALL                                   *
*    I=2, II=1  NO THRESHOLDS AT GAUGE COUPLINGS,YES ELSEWHERE         *
*    I=2, II=2  THRESHOLDS TO EVERYTHING                               *
*    I=1, II=2  THRESHOLDS AT GAUGE + YUKAWAS                          *
*    ONLY THE ABOVE COMBINATIONS ARE ALLOWED FOR THE SWITCHES          *
*                                                                      *
*    In the absence of threshold effects, in the RGEs, the "B,mu"      *
*    decouple from the rest of the RGEs but not with thresholds        *
*                                                                      *
*    Allowed combinations are:                                         *
*    NAI=1, NAIM=1 ( No thresholds )                                   *
*    NAI=2, NAIM=1 (Thresholds elsewhere except gauge couplings )      *
```



```
*    NAI=2, NAIM=2 (Thresholds to everything)                          *
*    NAI=1, NAIM=2 (Thresholds only in couplings)                      *
*    NAIH=1, NAIH=2 (No high energy thresholds, Yes high energy        *
*    thresholds, LOOK at FUNCTION NA(J) for more details               *
*    NRO=1 Yes Non Renormalizable Operators                            *
*    NRO=0 No Non Renormalizable Operators                             *
************************************************************************
      SUBROUTINE hdiu20(NAI,NAIM)
C
      WRITE(20,'(/,"****  NAI = ",I2,"   NAIM = ",I2)') NAI,NAIM
      if(NAI.EQ.1 .AND. NAIM.EQ.1)then
         WRITE(20,'(" ***   NO THRESHOLDS APPLY TO RGEs  ***")')
      elseif(NAI.EQ.1 .AND. NAIM.EQ.2)then
         WRITE(20,'(" *** THRESHOLDS ONLY IN RGEs OF COUPLINGS",
     #  " GAUGE + YUKAWAS *** ")')
      elseif(NAI.EQ.2 .AND. NAIM.EQ.1)then
         WRITE(20,'(" *** THRESHOLDS ONLY IN RGEs OF  MASSES",
     #  " + YUKAWAS *** ")')
      elseif(NAI.EQ.2 .AND. NAIM.EQ.2)then
         WRITE(20,'(" *** THRESHOLDS IN RGEs OF EVERYTHING  ***")')
      endif
C
      return
      END

*==> initcnd.f <==
      SUBROUTINE initcnd(ao,amoo,am12,efapto)
      IMPLICIT DOUBLE PRECISION (a-h,o-z)
      PARAMETER(ID1=14)    !  Fix it as you like
      DIMENSION iu1(ID1)
      DATA (iu1(k),k=1,ID1)/22, 86, 87, 88, 89, 90, 91, 92, 93,
     #                      94, 95, 96, 97, 98/
CC
      do   ir=1,ID1
          write(iu1(ir),'(34("-"),/,"A0 = ",f10.4,/,
     #  "M0 = ",f10.4,/,"M1/2 = ",f10.4,/,
     #  "tanb = ",f10.4,/,34("-"),//)')ao,amoo,am12,efapto
      enddo
      return
      END

*==> signmu.f <==
      SUBROUTINE signmu(i)
      PARAMETER(ID2=15)    !  Fix it as you like
      DIMENSION mu2(ID2)
      DATA (mu2(n),n=1,ID2)/22, 86, 87, 88, 89, 90, 91, 92, 93,
     #                      94, 95, 96, 97, 98, 20/
CC
      do   k=1,ID2
      if(i.eq.1)write(mu2(k),'(//,24X,"*** mu > 0  *** ",//)')
      if(i.eq.2)write(mu2(k),'(//,24X,"*** mu < 0  *** ",//)')
      enddo
CC
      return
      END

*==> sepmu.f <==
      SUBROUTINE sepmu
      PARAMETER(ID2=15)  !  Fix it as you like
      DIMENSION mu2(ID2)
      DATA (mu2(n),n=1,ID2)/22, 86, 87, 88, 89, 90, 91, 92, 93,
     #                      94, 95, 96, 97, 98, 20/
```

```
CC
       do   jr=1,ID2
           write(mu2(jr),'(/,34(">",X),/)')
       enddo
      return
      END

*==> xbrs.f <==
      SUBROUTINE xbrs(iu,tbr)
      IMPLICIT DOUBLE PRECISION (a-h,o-z)
      COMMON /SCALE/ amgut,susy,amz,amw
      qbr=amgut*dexp(-tbr/2.d0)
      write(iu,'(/,"Qsl = ",f10.4,//)')qbr
      return
      END

*==> bld_sector.f <==
      BLOCK DATA sector
      IMPLICIT DOUBLE PRECISION (a-h,o-z)
      DIMENSION icut(44),izip(48)
      COMMON /KOFTIS/ icut
      COMMON /ZIPER/ izip
      COMMON /RENTR/ t,q
      COMMON /XMARG/ rmar1,rmar2
      COMMON/FUTURE/smvector,smhcolor,smsigma,smhfprime
C
C  Top-stop contribution
C
C     DATA (icut(i),i=1,44)
C    #/0,0,0, 0,0,1, 0,0,0, 0,0, 2*0, 4*0, 0,0,0,
C    # 2*0,2*0,2*0, 2*0,2*0,2*1, 2*0,2*0,2*0, 2*0,2*0,2*0/
C     DATA (izip(j),j=1,48)
C    #/0,0,0, 0,0,1, 0,0,0, 0,0, 4*0, 4*0, 0,0,0,
C    # 2*0,2*0,2*0, 2*0,2*0,2*1, 2*0,2*0,2*0, 8*0/
C
C  Full sector contribution
C
      DATA (icut(i),i=1,44)
     #/1,1,1, 1,1,1, 1,1,1, 1,1, 2*1, 4*1, 1,1,1,
     # 2*1,2*1,2*1, 2*1,2*1,2*1, 2*1,2*1,2*1, 2*1,2*1,2*1/
      DATA (izip(j),j=1,48)
     #/1,1,1, 1,1,1, 1,1,1, 1,1, 4*1, 4*1, 1,1,1,
     # 2*1,2*1,2*1, 2*1,2*1,2*1, 2*1,2*1,2*1, 8*1/
C
      DATA smvector,smhcolor,smsigma,smhfprime
     #    /1.d0, 3.d0, 3.d0, 5.d0/
      DATA t,q /50.d0,200.d0/
      DATA rmar1,rmar2 /1.d6, 1.d6/
      END
```

```
      SUBROUTINE round(sf,x1,x2,ts,r1)
      IMPLICIT DOUBLE PRECISION (a-h,o-z)
      DIMENSION izip(48),psuvf(48)
      DOUBLE PRECISION sf(nuu),vcf(48),sqe(48)
      SAVE vcf
      COMMON /SCALE/ amgut,susy,amz,amw
      COMMON /wur/ wur,pi
      COMMON /ZIPER/ izip
      COMMON /NUEQ/ nuu
      DATA (vcf(jc),jc=1,48)
     # /3*-4.d0, 6*-12.d0, 6.d0, 3.d0, 8*-2.d0, 9*2.d0,
     # 12*6.d0, 8*1.d0/
CC
      call fdmass(sf,x1,x2,sqe)
      smax=fmax(sqe,12,48,48)
C
      do   jw=1,48
          psuvf(jw)=0.d0
      enddo
      ff1 = 0.d0
      do  i6=1,48
         if(izip(i6).ne.0)then
           psuvf(i6) = vcf(i6)*u(smax , sqe(i6) , ts )
     #      /(64.d0*pi*pi)
           ff1 = ff1+psuvf(i6)
         endif
      enddo
C
      psmax=psuvf(1)
      do   jt=2,48
          if(psuvf(jt).gt.psmax)psmax=psuvf(jt)
      enddo
C
      r1=dabs(ff1/psmax)
CC
      return
      END
```

```
      SUBROUTINE sear(p,ql,qr,hmax,tol,torl)
      IMPLICIT DOUBLE PRECISION (a-h,o-z)
      PARAMETER (nu=39,iw=28+21*nu,iww=23+21*nu)
      PARAMETER (n=2)
      LOGICAL log1
      DOUBLE PRECISION fch
      DOUBLE PRECISION pparam(iww),sparam(iw),faram(iw)
      DOUBLE PRECISION p(n),xcom(nu),xcom1(nu)
      COMMON /SOFCOM/ xcom  !  At ren. true ie Mz
      COMMON /SCALE/amgut,susy,amz,amw
      COMMON /INTZER/ log1
      EXTERNAL fon,fch
CC
      x1 = p(1)
      x2 = p(2)
CC
C     write(91,*)'**************************'
C     write(91,*)'x1 = ',x1
C     write(91,*)'x2 = ',x2
C     write(91,*)'**************************'
CC
      do  it=1,nu
         xcom1(it)=xcom(it)
      enddo
      xcom1(32)=x1
      xcom1(33)=x2
C*********************************************************
C          tol=.0000000005d0  !  Default (5.d-10)
C
           sqi=2.d0*dlog(amgut/amz)
           sqf=2.d0*dlog(amgut/ql)
           ifl=0
           call d02caf(sqi,sqf,nu,xcom1,tol,fon,pparam,ifl)
C*********************************************************
C     torl=.0000000005d0   !  Default 5.d-10
C     ifail=0    !  NAG HARD FAILURE ....
C
      irel=0
      si=sqf
      sf=2.d0*dlog(amgut/qr)
      ifail=-1    !  NAG SOFT FAILURE ....
      log1=.true.
      call d02chf(si,sf,nu,xcom1,torl,irel,hmax,fon,fch,
     #  sparam,ifail)
CC
      return
      END
```

```
      DOUBLE PRECISION FUNCTION tfunk(p)
      IMPLICIT DOUBLE PRECISION (a-h,o-z)
      PARAMETER (nu=39,iw=28+21*nu,iww=23+21*nu)
      PARAMETER (n=2,rscal=90000.d0)
      DOUBLE PRECISION pparam(iww),sparam(iw),faram(iw)
      DOUBLE PRECISION p(n),xcom(nu),xcom1(nu),ycr(nu)
      LOGICAL lpx,fre,lcaf
      COMMON /SCALE/amgut,susy,amz,amw
      COMMON /SOFCOM/ xcom
      COMMON /RENTR/ tsig,qsl
      COMMON /SWITCH/fre
      COMMON /CRVAL/tcr,ycr
      COMMON /XUEUE/ lpx,lcaf
      COMMON /XMARG/ rmar1,rmar2
      EXTERNAL funct
C
      x1 = p(1)
      x2 = p(2)
C-----------------------------------------
C   Defining a 'Box' for the function Veff
C-----------------------------------------
     if(dabs(x1).lt.rmar1.and.dabs(x2).lt.rmar2)then
        if(dabs(x1).lt.5d2 .and. dabs(x2).lt.5.d2)then
          call vborn(93,.false.,xcom,x1,x2,veff0)
          call dv1(93,.false.,xcom,0.d0,0.d0,tsig,wmeg)
          call dv1(93,.false.,xcom,x1,x2,tsig,veff1)
        else
          do   it=1,nu
              xcom1(it)=xcom(it)
          enddo
          xcom1(32)=x1
          xcom1(33)=x2
C         torl=.0000000005d0
          torl=5.d-15
          si=tsig
          sf=2.d0*dlog(amgut/1.d10)
          ifail=0
          fre=.true.
          call d02caf(si,sf,nu,xcom1,torl,funct,pparam,ifail)
          call vborn(93,.false.,ycr,ycr(32),ycr(33),veff0)
          call dv1(93,.false.,ycr,0.d0,0.d0,tcr,wmeg)
          call dv1(93,lpx,ycr,ycr(32),ycr(33),tcr,veff1)
        endif
C-------------------------------------------------------
C   Rescaling so as to have as small values as possible
C-------------------------------------------------------
      v=veff0-wmeg+veff1
      tfunk=v/rscal**4
C-----------------------------------------------
C
C  Printing facilities
C
         qcr=amgut*dexp(-tcr/2.d0)
         if(lpx)
     #    write(94,'(/"x1(Q) = ",d25.17/,"x2(Q) = ",d25.17,/,
   #  "Rn. scale = ",f26.17,/,"Vacen = ",d25.17,
   #  /,"Pure tree = ",d25.17,/,"Tree - Vacen = ",d25.17,/,
   #  "1-loop = ",d25.17,/,"v0/v1 = ",f10.5,/,"[v0-w]/v1 = ",
   #  f10.5,/,"VFULL = ",d25.17/')
     #    ycr(32),ycr(33),qcr,wmeg,veff0,veff0-wmeg,veff1,
     #    veff0/veff1,(veff0-wmeg)/veff1,v
C-----------------------------------------------
```



```
      else
         tfunk = 1.d200
      endif
      return
      END
```

```
      SUBROUTINE vaver(t,y)
      IMPLICIT DOUBLE PRECISION (a-h,o-z)
      PARAMETER(NU=39)
      DOUBLE PRECISION y(NU),sqe(48),ycr(NU)
      LOGICAL fre,lpx,lcaf
      SAVE conmin,icount
      COMMON /SCALE/ amgut, susy, amz, amw
      COMMON /CRVAL/ tcr,ycr
      COMMON /SWITCH/ fre
      COMMON /XUEUE/ lpx,lcaf
      COMMON /TRMIN/ vtrue,pb(2)
      DATA conmin /0.d0/
      DATA icount /0/  ! *** Print only ONCE ***
CC
      q=amgut*dexp(-t/2.d0)
      tinc=2.d0*dlog(amgut/amz) - t
C
      call fdmass(y,y(32),y(33),sqe)
      con=fcond(t,sqe)
C
      if(fre.or.con.lt.conmin)then
         conmin=con
         tcr=t
         do i=1,NU
           ycr(i)=y(i)
         enddo
         fre=.false.
      endif
C
      qcr=amgut*dexp(-tcr/2.d0)
C-----------------------------------------------------------------
C     Printing facilities
C-----------------------------------------------------------------
         if(lcaf .and. (icount/2)*2.eq.icount)then
           call vborn(92,.false.,y,y(32),y(33),veff0)
           call dv1(92,.false.,y,0.d0,0.d0,t,wmeg)
           call dv1(92,lcaf,y,y(32),y(33),t,veff1)
           v=veff0-wmeg+veff1
           vsmal=wmeg+vtrue
           ratio=1.d0 + veff1/(veff0-vsmal)
C
           write(92,'(/"x1(Q) = ",d25.17/,"x2(Q) = ",d25.17,/,
   #    "Rn. scale = ",f30.17,/,"Crit. scale = ",f30.17/,
   #    "Sum of Log^2 = ",f20.12/,"Critical value = ",f20.12/,
   #    "Tinc = ",f20.14/,"Vacen = ",d25.17,
   #    /,"Pure tree = ",d25.17,/,"Tree - Vacen = ",d25.17,/,
   #    "1-loop = ",d25.17,/,"v0/v1 = ",f10.5,/,
   #    "1 + v1/[v0-w-vtrue] = ",f10.5,/,"VFULL = ",
   #    d25.17///)')
      #      y(32),y(33),q,qcr,con,conmin, tinc,wmeg,veff0,
      #      veff0-wmeg,veff1,(veff0-wmeg)/veff1,ratio,v
C
C-----------------------------------------------------------------
         endif
C
      icount=icount+1
      return
      END
```