1 Modification of the TALYS file 'massdis.f'.

When using the fission model 'fymodel 4' in TALYS, the user has to specify the keyword 'massdis' in order for TALYS to calculate the fission fragment mass distribution. The subroutine 'massdis' must be modified if TALYS is to choose the files specifically provided by the user. To achieve this, the new variable 'yieldfileid' is passed along to the subroutine 'massdis' via the subroutine 'talys.cmb'. An if-statement is added to 'massdis.f' to check the 'yieldfileid' variable. If the variable has the default value, TALYS uses the ordinary files in the Y_{ff} - library. If the value is not the default value, then that value is added to the Y_{ff} file name just before the suffix '.ff'. In this way, TALYS can use the file provided by the user in its simulations. The modifications are highlighted in yellow.

```
       subroutine massdis
c
c +---------------------------------------------------------------------
c  Author:  Arjan Koningc  Date : May 22, 2021
c  Task :  Fission fragment yields
c  Edited: Peter Karlsson; Date: May 7 2023
c +---------------------------------------------------------------------
c
c ******************** Editing information ********************************
c  Added if condition to check if user supplied a value to the
c  common variable 'yieldfileid'.
c  If no: do nothing. (default keyword value = 'novalue')
c  if yes: Add string 'yieldfileid' to the fission fragment yield file name
c  in order to read specific yieldfile when multi-threading.
c *************************************************************************

c ****************** Declarations and common blocks ********************
c
      include "talys.cmb"
      include "gef.cmb"
      integer numZff,numNff,numpair
      parameter (numZff=80,numNff=150,numpair=2000)
      logical lexist
      character*2 Sfile
      character*3 massstring
      character*6 nucstring
      character*8 Estring
      character*13 fffile
      character*132 gefpath,ffpath,Efile,ffname,Yfile(2)
      character*132 string
      real Exfis(1000),xsfis(1000),Jfis,fisepsA,fisepsB,
     + partfisxs,sumpre,sumpost,Fmulti,beldm1(136,203),
     + ushell1(136,203),xstabtot(numZff,numNff),
     + xstabcomp(0:numZ,0:numN,numZff,numNff),
     + popffEx(numZff,numNff,0:numpop),
     + popffJ(numZff,numNff,0:numJ),Ebin(0:numpop),
     + term,Etabtot(numZff,numNff,1000),partfisJ(0:numJ),
     + Jtabtot(numZff,numNff,100),sumJ,sumxs,sumE,sum,
     + xsfisFF,Ytabtot(numZff,numNff),Eff(0:numenin),
     + Y(2,numpair),TKE0(2,numpair),TXE0(2,numpair),
     + Elight(2,numpair),Wlight(2,numpair),
     + Eheavy(2,numpair),Y0,ELL,EH,dEL,dEH,Extab(2),
     + Wheavy(2,numpair),Efac,Efftab(2),TK,TX,Effrel
      integer iz,ia,in,i,j,gefwrite,Zcomp,Ncomp,Z,A,odd,Zix,Nix,k,
     + nexend,iskip,istep,nex,nen,type,iza,Nff,
     + nexgef,Jgef,nb,parity,Zlight(2,numpair),Nfftab,
     + Alight(2,numpair),Zheavy(2,numpair),Afile,
     + Aheavy(2,numpair),Ntotal(2),izL,iaL,inL,izH,iaH,inH
c
c ************************** Mass yields *******************************
c
c xsfistot : total fission cross section
c nummass : number of masses
c xsApre : pre-neutron emission cross section
c xsApost : post-neutron emission corrected cross section
c yieldApre : pre-neutron emission fission yield
```

```
c yieldApost : post-neutron emission corrected fission yield
c numelem : number of elements
c numZff : number of Z of fission fragments
c numNff : number of N of fission fragments
c fffile : fission fragment file
c xsZApre : pre-neutron emission isotopic cross section
c xsZApost : post-neutron emission corrected isotopic cross section
c yieldZApre : pre-neutron emission isotopic yield
c yieldZApost: post-neutron emission corrected isotopic yield
c fymodel : fission yield model, 1: Brosa 2: GEF
c path : directory containing structure files to be read
c gefpath : path for GEF files
c popffEx : energy population of FF
c popffJ : spin population of FF
c flagoutfy : flag for output detailed fission yield calculation
c gefwrite : integer for output detailed fission yield calculation
c Rfiseps : ratio for limit for fission cross section per nucleus
c fiseps : limit for fission cross section per excitation energy bin
c
c Initialization
c
      do ia=1,nummass
        xsApre(ia)=0.
        xsApost(ia)=0.
        yieldApre(ia)=0.
        yieldApost(ia)=0.
        do type=0,numpar
          nuA(type,ia)=0.
          EaverageA(type,ia)=0.
        enddo
      enddo
      do in=1,numneu
        do iz=1,numelem
          if (iz.gt.numelem.or.in.gt.numneu) cycle
          xsZApre(iz,in)=0.
          xsZApost(iz,in)=0.
          yieldZApre(iz,in)=0.
          yieldZApost(iz,in)=0.
          do nex=0,1
            xsfpex(iz,in,nex)=0.
            yieldfpex(iz,in,nex)=0.
            fpratio(iz,in,nex)=0.
          enddo
          Excff(iz,in)=0.
          dExcff(iz,in)=0.
          TKE(iz,in)=0.
          do type=0,6
            nuZA(type,iz,in)=0.
            EaverageZA(type,iz,in)=0.
          enddo
        enddo
      enddo
      do i=1,2
        Ntotal(i)=0
        do k=1,numpair
          Zlight(i,k)=0
          Alight(i,k)=0
          Zheavy(i,k)=0
          Aheavy(i,k)=0
          Elight(i,k)=0.
          Wlight(i,k)=0.
          Eheavy(i,k)=0.
          Wheavy(i,k)=0.
          Y(i,k)=0.
          TKE0(i,k)=0.
          TXE0(i,k)=0.
        enddo
      enddo
      xstotpre=0.
        xstotpost=0.
        yieldtotpre=0.
        yieldtotpost=0.
        do type=0,6
          Pdisnuav(type)=0.
          do i=0,numnu
            Pdisnu(type,i)=0.
          enddo
          nubar(type)=0.
        enddo
        do i=1,numenin
          Eff(i)=0
        enddo
        do iz=1,numZff
          do in=1,numNff
            if (iz.gt.numelem.or.in.gt.numneu) cycle
            xstabtot(iz,in)=0.
            do nex=1,1000
              Etabtot(iz,in,nex)=0.
            enddo
            do J=1,100
              Jtabtot(iz,in,J)=0.
            enddo
            do nex=0,numpop
              popffEx(iz,in,nex)=0.
            enddo
            do J=0,numJ
              popffJ(iz,in,J)=0.
            enddo
            do Zcomp=0,maxZ
              do Ncomp=0,maxN
                xstabcomp(Zcomp,Ncomp,iz,in)=0.
              enddo
            enddo
          enddo
        enddo
  c
  c Only do FY calculation if the fission cross section is large enough
  c
        fpeps=Rfiseps*xsfistot
        if (fpeps.eq.0.) return
        if (fymodel.eq.2.or.fymodel.eq.3) then
          gefpath=trim(path)//'fission/gef/'
          if (flagoutfy) then
            gefwrite=1
          else
            gefwrite=0
          endif
  c
  c Read nuclear structure information for GEF
  c
  c beldm1: binding energy from liquid drop model
  c fisepsA : fission tolerance
  c fisepsB : fission tolerance
  c ushell1: shell correction
  c
          open (unit=4,file=trim(gefpath)//'beldm.dat',status='old')
          read(4,*) beldm1
          close(4)
          do i=1,203
            do j=1,136
             beldm(i,j)=beldm1(j,i)
            end do
          end do
          open (unit=4,file=trim(gefpath)//'ushell.dat',status='old')
          read(4,*) ushell1
          close(4)
          do i=1,203
            do j=1,136
```

```
           ushel(i,j)=ushell1(j,i)
          end do
        end do
        open (unit=4,file=trim(gefpath)//'nucprop.dat',status='old')
        do i=1,3885
          read(4,*) (RNucTab(i,j),j=1,8)
        end do
        close(4)
      endif
c
c Loop over nuclides
c
c Zcomp : charge number index for compound nucleus
c maxZ : maximal number of protons away from the initial
c compound nucleus
c Ncomp : neutron number index for compound nucleus
c maxN : maximal number of neutrons away from the initial
c compound nucleus
c ZZ,Z : charge number of residual nucleus
c AA,A : mass number of residual nucleus
c Zindex,Zix : charge number index for residual nucleus
c Nindex,Nix : neutron number index for residual nucleus
c maxex : maximum excitation energy bin for compound nucleus
c iskip,istep: help variables
c xsbinary : cross section from initial compound to residual nucleus
c Ex : excitation energy
c partfisxs : partial fission cross section
c fisfeedex : fission contribution from excitation energy bin
c brosafy : subroutine for fission fragment yields based on Brosa
c model
c disa : normalised fission fragment mass yield per excitation
c energy bin
c disacor : normalised fission product mass yield per excitation
c energy bin
c disaz : normalised fission fragment isotope yield
c per excitation energy bin
c disazcor : normalised fission product isotope yield
c per excitation energy bin
c gefran : number of random events for GEF calculation
c Exfis: excitation energy for fission
c
      do 20 Zcomp=0,maxZ
        do 25 Ncomp=0,maxN
          Z=ZZ(Zcomp,Ncomp,0)
          A=AA(Zcomp,Ncomp,0)
          odd=mod(A,2)
          Zix=Zindex(Zcomp,Ncomp,0)
          Nix=Nindex(Zcomp,Ncomp,0)
          if (xsfeed(Zcomp,Ncomp,-1).le.fpeps) goto 25
          if (Zcomp.eq.0.and.Ncomp.eq.0) then
            nexend=maxex(Zcomp,Ncomp)+1
          else
            nexend=maxex(Zcomp,Ncomp)
          endif
          fisepsA=fpeps/max(3*maxex(Zcomp,Ncomp),1)
          iskip=0
          istep=4
          if (fymodel.eq.2) then
            do 30 nex=1,1000
              Exfis(nex)=0.
              xsfis(nex)=0.
   30 continue
            nen=0
          endif
          do 40 nex=nexend,0,-1
            if (Zcomp.eq.0.and.Ncomp.eq.0.and.
     + nex.eq.maxex(Zcomp,Ncomp)+1) then
              excfis=Etotal
              partfisxs=xsbinary(-1)
              do 42 J=0,numJ
                partfisJ(J)=0.
                do 44 parity=-1,1,2
                  partfisJ(J)=partfisJ(J)+
     + fisfeedJP(Zcomp,Ncomp,nex,J,parity)
   44 continue
   42 continue
            else
              if (mod(iskip,istep).ne.0) then
                iskip=iskip+1
                goto 40
              endif
              if (nex-istep+1.lt.0) goto 40
              if (Ex(Zcomp,Ncomp,nex-istep+1).ge.30.) then
                partfisxs=0.
                do 45 J=0,numJ
                  partfisJ(J)=0.
   45 continue
                do 50 i=0,istep-1
                  partfisxs=partfisxs+fisfeedex(Zcomp,Ncomp,nex-i)
                  do 55 J=0,numJ
                    do 57 parity=-1,1,2
                      partfisJ(J)=partfisJ(J)+
     + fisfeedJP(Zcomp,Ncomp,nex-i,J,parity)
   57 continue
   55 continue
   50 continue
                if (partfisxs.ne.0) then
                  excfis=0.
                  do 60 i=0,istep-1
                    excfis=excfis+fisfeedex(Zcomp,Ncomp,nex-i)*
     + Ex(Zcomp,Ncomp,nex-i)
   60 continue
                  excfis=excfis/partfisxs
                endif
                iskip=1
              else
                excfis=Ex(Zcomp,Ncomp,nex)
                partfisxs=fisfeedex(Zcomp,Ncomp,nex)
                do 62 J=0,numJ
                  partfisJ(J)=0.
                  do 64 parity=-1,1,2
                    partfisJ(J)=partfisJ(J)+
     + fisfeedJP(Zcomp,Ncomp,nex,J,parity)
   64 continue
   62 continue
              endif
            endif
            if (partfisxs.gt.fisepsA) then
c
c Brosa
c Normalization: sum over disa = 2.
c
              if (fymodel.eq.1) then
                call brosafy(Zix,Nix)
                term=0.5*partfisxs
                do 70 ia=1,A
                  xsApre(ia)=xsApre(ia)+term*disa(ia)
                  xsApost(ia)=xsApost(ia)+term*disacor(ia)
                  do 72 iz=1,Z
                    in=ia-iz
                    if (in.lt.1.or.in.gt.numneu) goto 72
                    xsZApre(iz,in)=xsZApre(iz,in)+term*disaz(ia,iz)
                    xsZApost(iz,in)=xsZApost(iz,in)+term*disazcor(ia,iz)
   72 continue
   70 continue
              endif
c
c GEF
```

```
c
c xsfis: fission cross section
c xsfisFF: fission cross section per FF
c
       if (fymodel.eq.2) then
        nen=nen+1
        Exfis(nen)=excfis
        xsfis(nen)=partfisxs
       endif
c
c GEF + TALYS evaporation
c Normalization: sum over Ytab, Etab, Jtab = 1
c
c flagfisout : flag for output of fission information
c Jfis : spin of fissioning system
c partfisJ : partial fission spin distribution
c xstabtot: total cross section from GEF
c Jtabtot: total spin from GEF
c xstabcomp: Z, N cross section from GEF
c Ytabtot: yield from GEF
c
       if (fymodel.eq.3.and.A.le.350) then
        fisepsB=fisepsA/(5*maxJ(Zcomp,Ncomp,nex))*0.5
        do 75 J=0,maxJ(Zcomp,Ncomp,nex)
         if (partfisJ(J).lt.fisepsB) goto 75
         Jfis=real(J)+0.5*odd
         call gefsub(Z,A,excfis,Jfis)
         write(*,*) " FF excitation for Z= ",
  + Z," A= ",A," Ex= ",excfis," J= ",Jfis," xs= ",
  + partfisJ(J)," N_cases:",N_cases
         do 80 iza=1,N_cases
          iz=NZMkey(iza,3)
          in=NZMkey(iza,2)
          if (iz.gt.numZff.or.in.gt.numNff) goto 80
          term=Ytab(iza)*partfisJ(J)
          xstabtot(iz,in)=xstabtot(iz,in)+term
          xstabcomp(Zcomp,Ncomp,iz,in)=
  + xstabcomp(Zcomp,Ncomp,iz,in)+term
          do 85 nexgef=1,1000
           Etabtot(iz,in,nexgef)=Etabtot(iz,in,nexgef)+
  + term*Etab(iza,nexgef)
 85 continue
          do 87 Jgef=1,100
           Jtabtot(iz,in,Jgef)=Jtabtot(iz,in,Jgef)+
  + term*Jtab(iza,Jgef)
 87 continue
 80 continue
 75 continue
       endif
      endif
 40 continue
c
c GEF
c Normalization: sum over ysum,yAz= 2. * sigma_fission
c
c flagffspin: flag to use spin distribution in initial population
c Fmulti: factor for multi-chance fission
c
     if (fymodel.eq.2.and.A.le.350) then
      call geftalys(real(Z),real(A),nen,Exfis,xsfis,gefwrite,
  + gefran)
      do ia=1,A
       xsApre(ia)=xsApre(ia)+0.5*ysum(ia)
       xsApost(ia)=xsApost(ia)+0.5*ysump(ia)
       if (ia.le.200) then
        do iz=1,Z
         in=ia-iz
         if (in.ge.1.and.in.le.numneu) then
          xsZApre(iz,in)=xsZApre(iz,in)+0.5*yAZ(ia,iz)
```

```
                    yieldZApre(iz,in)=yAZ(ia,iz)
                    xsZApost(iz,in)=xsZApost(iz,in)+0.5*yAZp(ia,iz)
                  endif
                enddo
              endif
            enddo
            if (xsfistot.gt.0.) then
              Fmulti=Ncomp*xsfeed(Zcomp,Ncomp,-1)
              do i=0,numnu
                if (ann_sum(i).gt.0.)
     + Pdisnu(1,i)=Pdisnu(1,i)+(Fmulti+ann_sum(i))/xsfistot
              enddo
              do ia=1,A
                if (anpre_sum(ia).gt.0.)
     + nuA(1,ia)=nuA(1,ia)+(Fmulti+anpre_sum(ia))/xsfistot
              enddo
              nubar(1)=nubar(1)+(Fmulti+anMean_sum)/xsfistot
            endif
          endif
   25 continue
   20 continue
      do type=0,6
        if (parskip(type)) cycle
        sum=0.
        do 90 i=0,numin
          sum=sum+Pdisnu(type,i)
   90 continue
        if (sum.gt.0.) then
          do 95 i=0,numin
            Pdisnu(type,i)=Pdisnu(type,i)/sum
   95 continue
        endif
      enddo
c
c GEF + TALYS evaporation (fymodel 3)
c or
c yields + TALYS evaporation (fymodel 4)
c
c Ebin: energy of bin
c Etabtot: tabulated energy
c sumJ: sum over spin distribution
c Jgef: counter
c nexbeg: first energy index
c nexend: last energy index
c nexgef: counter
c
      if (fymodel.ge.3) then
        Ebin(0)=0.
        do i=1,numpop
          Ebin(i)=0.1*i
        enddo
        if (fymodel.eq.3) then
          sumxs=0.
          do 131 iz=1,numZff
            do 132 in=1,numNff
              if (iz.gt.numelem.or.in.gt.numneu) goto 132
              if (xstabtot(iz,in).eq.0.) goto 132
              sumxs=sumxs+xstabtot(iz,in)
              sumE=0.
              do nexgef=1,1000
                sumE=sumE+Etabtot(iz,in,nexgef)
              enddo
              if (sumE.gt.0.) then
                do nexgef=1,1000
                  Etabtot(iz,in,nexgef)=Etabtot(iz,in,nexgef)/sumE
                enddo
              endif
              sumJ=0.
              do Jgef=1,100
              sumJ=sumJ+Jtabtot(iz,in,Jgef)
            enddo
            if (sumJ.gt.0.) then
              do Jgef=1,100
                Jtabtot(iz,in,Jgef)=Jtabtot(iz,in,Jgef)/sumJ
              enddo
            endif
132 continue
131 continue
        if (sumxs.gt.0.) then
          do iz=1,numZff
            do in=1,numNff
              if (iz.gt.numelem.or.in.gt.numneu) cycle
              Ytabtot(iz,in)=xstabtot(iz,in)/sumxs
            enddo
          enddo
        endif
        do iz=1,numZff
          do in=1,numNff
            if (iz.gt.numelem.or.in.gt.numneu) cycle
            ia=iz+in
            xsfisFF=xsfistot*Ytabtot(iz,in)
            xsApre(ia)=xsApre(ia)+xsfisFF
            xsZApre(iz,in)=xsZApre(iz,in)+xsfisFF
            yieldZApre(iz,in)=Ytabtot(iz,in)
            do nexgef=1,1000
              popffEx(iz,in,nexgef)=popffEx(iz,in,nexgef)+
   + xsfisFF*Etabtot(iz,in,nexgef)
            enddo
            do J=1,30
              popffJ(iz,in,J)=Jtabtot(iz,in,J)
            enddo
          enddo
        enddo
        do 160 iz=1,Z
          do 170 ia=iz+1,A
            in=ia-iz
            if (in.gt.numneu) goto 170
            if (xsZApre(iz,in).eq.0.) goto 170
            sumJ=0.
            do J=1,30
              sumJ=sumJ+popffJ(iz,in,J)
            enddo
            if (sumJ.gt.0.) then
              do J=1,30
                popffJ(iz,in,J)=popffJ(iz,in,J)/sumJ
              enddo
            endif
            nb=numpop
            do 165 nex=100,numpop
              if (popffEx(iz,in,nex).lt.1.e-9) then
                nb=nex
                goto 167
              endif
165 continue
167 nb=max(nb,1)
            fffile=’ff000000.ex’
            write(fffile(3:5),’(i3.3)’) iz
            write(fffile(6:8),’(i3.3)’) ia
            open (unit=1,file=fffile,status=’replace’)
            if (flagffspin) then
              write(1,*) nb,30,1," Yff= ",yieldZApre(iz,in),
   + " xs= ",xsZApre(iz,in)
              do nex=0,nb
                write(1,’(f10.5,30es12.5)’) Ebin(nex),
   + (0.5*popffEx(iz,in,nex)*popffJ(iz,in,J),
   + J=1,30)
              enddo
            else
```

```
                write(1,*) nb,0,1," Yff= ",yieldZApre(iz,in),
     + " xs= ",xsZApre(iz,in)
                do nex=0,nb
                  write(1,'(f10.5,es12.5)') Ebin(nex),popffEx(iz,in,nex)
                enddo
              endif
              close(1)
  170 continue
  160 continue
        endif
c
c fymodel 4: Okumura model - read in yields and excitation energies
c ffmodel : fission fragment model,
c 1: GEF (database by Ali Al-Adili and Fredrik Nordstroem)
c 2: HF3D (Okumura) (database by Toshihiko Kawano)
c 3: SPY (Okumura) (Jean-Francois Lemaitre)
c
        if (fymodel.eq.4) then
          if (yieldfile(1:1).eq.' ') then
            Effrel=Etotal
            ffpath=trim(path)//'fission/ff/'
            if (ffmodel.eq.0) ffname='local'
            if (ffmodel.eq.1) ffname='gef'
            if (ffmodel.eq.2) ffname='hf3d'
            if (ffmodel.eq.3) ffname='spy'
            ffpath=trim(ffpath)//trim(ffname)//'/'
            massstring=' '
            Afile=Ainit
            Sfile=nuc(Zinit)
            write(massstring,'(i3.3)') Afile
            nucstring=trim(Sfile)//massstring//isochar(Lisoinp)
            ffpath=trim(ffpath)//trim(nucstring)//'/'
            Efile=trim(ffpath)//trim(nucstring)//'_'//trim(ffname)//'.E'
            inquire (file=Efile,exist=lexist)
            if (.not.lexist) then
              write(*,'(" TALYS-error: Non-existent FF file ",a)')
     + trim(Efile)
              stop
            endif
            open (unit=1,file=Efile,status='unknown')
            i=1
  152 read(1,*,end=154) Eff(i)
            i=i+1
            goto 152
  154 close (1)
            Nff=i-1
            if (Effrel.le.Eff(1)) then
              Efftab(1)=Eff(1)
              Nfftab=1
            else
              Nfftab=2
              if (Effrel.ge.Eff(Nff)) then
                Efftab(1)=Eff(Nff)
                Efftab(2)=Emaxtalys
                Extab(2)=Efftab(1)
              else
                call locate(Eff,0,Nff,Effrel,nen)
                Efftab(1)=Eff(nen)
                Efftab(2)=Eff(nen+1)
                Extab(2)=Eff(nen+1)
              endif
              Efac=(Effrel-Efftab(1))/(Efftab(2)-Efftab(1))
            endif
            Extab(1)=Efftab(1)
            do k=1,Nfftab
              Estring=' '
              write(Estring,'(es8.2)') Extab(k)
              Estring(5:5)='e'
              if (yieldfileid == 'novalue') then
              Yfile(k)=trim(ffpath)//trim(nucstring)//'_'//Estring//
   + 'MeV_'//trim(ffname)//'.ff'
            else
              Yfile(k)=trim(ffpath)//trim(nucstring)//'_'//Estring//
  + 'MeV_'//trim(ffname)//'_'//trim(yieldfileid)//'.ff'
            endif
          enddo
        else
          Nfftab=1
          Yfile(1)=trim(yieldfile)
        endif
        do k=1,Nfftab
          write(*,'(/," Fission fragment yields read from ",a)')
   + trim(Yfile(k))
          inquire (file=Yfile(k),exist=lexist)
          if (.not.lexist) then
            write(*,'(" TALYS-error: Non-existent FF file ",a)')
   + trim(Yfile(k))
            stop
          endif
          open (unit=1,file=Yfile(k),status='unknown')
          read(1,'(///13x,i6)') Ntotal(k)
          read(1,'(a)') string
          do i=1,Ntotal(k)
            read(1,'(a)',end=164) string
            read(string,*) Zlight(k,i),Alight(k,i),
   + Zheavy(k,i),Aheavy(k,i),Y(k,i),TKE0(k,i),TXE0(k,i),
   + Elight(k,i),Wlight(k,i),Eheavy(k,i),Wheavy(k,i)
          enddo
164 close (1)
        enddo
        write(*,'(/," Fission fragment pairs")')
        write(*,'(" Zl Al Zh Ah Yield TKE TXE ",
   + " ELight dElight Eheavy dEheavy")')
        do i=1,Ntotal(Nfftab)
          izL=Zlight(Nfftab,i)
          iaL=Alight(Nfftab,i)
          izH=Zheavy(Nfftab,i)
          iaH=Aheavy(Nfftab,i)
          inL=iaL-izL
          inH=iaH-izH
          ia=iaL+iaH
          if (inL.lt.1.or.inL.gt.numneu) cycle
          if (inH.lt.1.or.inH.gt.numneu) cycle
          if (ia.ne.Ainit) cycle
          ELL=Elight(Nfftab,i)
          dEL=Wlight(Nfftab,i)
          EH=Eheavy(Nfftab,i)
          dEH=Wheavy(Nfftab,i)
          Y0=Y(Nfftab,i)
          TK=TKE0(Nfftab,i)
          TX=TXE0(Nfftab,i)
          if (Nfftab.eq.2) then
            do j=1,Ntotal(1)
              if (Zlight(1,j).eq.izL.and.Alight(1,j).eq.iaL) then
                ELL=Elight(1,j)+Efac*(Elight(2,i)-Elight(1,j))
                dEL=Wlight(1,j)+Efac*(Wlight(2,i)-Wlight(1,j))
                Y0=Y(1,j)+Efac*(Y(2,i)-Y(1,j))
                TK=TKE0(1,j)+Efac*(TKE0(2,i)-TKE0(1,j))
                TX=TXE0(1,j)+Efac*(TXE0(2,i)-TXE0(1,j))
              endif
              if (Zheavy(1,j).eq.izH.and.Aheavy(1,j).eq.iaH) then
                EH=Eheavy(1,j)+Efac*(Eheavy(2,i)-Eheavy(1,j))
                dEH=Wheavy(1,j)+Efac*(Wheavy(2,i)-Wheavy(1,j))
                Y0=Y(1,j)+Efac*(Y(2,i)-Y(1,j))
                TK=TKE0(1,j)+Efac*(TKE0(2,i)-TKE0(1,j))
                TX=TXE0(1,j)+Efac*(TXE0(2,i)-TXE0(1,j))
              endif
            enddo
            endif
            xsfisFF=xsfistot*Y0
            xsApre(iaL)=xsApre(iaL)+xsfisFF
            xsApre(iaH)=xsApre(iaH)+xsfisFF
            xsZApre(izL,inL)=xsZApre(izL,inL)+xsfisFF
            xsZApre(izH,inH)=xsZApre(izH,inH)+xsfisFF
            Excff(izL,inL)=ELL
            Excff(izH,inH)=EH
            dExcff(izL,inL)=dEL
            dExcff(izH,inH)=dEH
            TKE(izL,inL)=TK
            TKE(izH,inH)=TK
            write(*,'(4i4,7es12.5)') izL,iaL,izH,iaH,Y0,TK,TX,
     + ELL,dEL,EH,dEH
          enddo
        endif
c
c fymodel 5: General model - read in full population per fission fragment
c
        if (fymodel.eq.5) then
          fffile='ff000000.ex'
          do 180 iz=1,Z
            do 190 ia=iz+1,A
              write(fffile(3:5),'(i3.3)') iz
              write(fffile(6:8),'(i3.3)') ia
              inquire (file=fffile,exist=lexist)
              if (lexist) then
                open (unit=1,file=fffile,status='old')
                in=ia-iz
                if (in.le.numneu) then
                  read(1,*) nb
                  do nex=0,nb
                    read(1,'(f10.5,es12.5)') Ebin(nex),
     + popffEx(iz,in,nex)
                  enddo
                endif
                close (1)
              endif
  190 continue
  180 continue
        endif
      endif
c
c Normalization to fission yields (sum = 2)
c
c sumpre: sum over pre-neutron FP's
c
      sumpre=0.
      sumpost=0.
      do ia=1,Atarget
        sumpre=sumpre+xsApre(ia)
        sumpost=sumpost+xsApost(ia)
      enddo
      do 220 iz=1,Ztarget
        do 230 ia=iz+1,Atarget
          in=ia-iz
          if (iz.gt.numelem.or.in.gt.numneu) goto 230
          if (xsZApre(iz,in).eq.0.) goto 230
          if (sumpre.gt.0.) yieldZApre(iz,in)=2.*xsZApre(iz,in)/sumpre
          yieldApre(ia)=yieldApre(ia)+yieldZApre(iz,in)
          yieldtotpre=yieldtotpre+yieldZApre(iz,in)
          xstotpre=xstotpre+xsZApre(iz,in)
          if (fymodel.le.2) then
            if (sumpost.gt.0.) yieldZApost(iz,in)=
     + 2.*xsZApost(iz,in)/sumpost
            yieldApost(ia)=yieldApost(ia)+yieldZApost(iz,in)
            yieldtotpost=yieldtotpost+yieldZApost(iz,in)
            xstotpost=xstotpost+xsZApost(iz,in)
          endif
230 continue
220 continue
    return
    end
```