egs5 sample user code (uccg_nai.f)
Response calculation of NaI detector
(Draft, July 28, 2004)

Hideo Hirayama and Yoshihito Namito

KEK, High Energy Accelerator Research Organization
1-1, Oho, Tsukuba, Ibaraki, 305-0801 Japan

Contents

1. PRESTA-CG

2. Combinatorial geometry (cg)

2.1. Body Definition

Following bodies are supported in PRESTA-CG*.

1. Rectangular Parallel-piped (RPP)
   Specify the maximum and minimum values of x-, y-, and z-coordinates that bound a rectuangular parallel-piped whose six sides are perpendicular to the coordinate axis.

2. Sphere (SPH)
   Specify the components of the radius vector **V** to the center of sphere and the radius R of the sphere.

3. Right Circular Cylinder (RCC)
   Specify the components of a radius vector **V** to the center of one base, the components of a vector **H** from the center of that base to the other base, and the radius of the cylinder.

4. Truncated Right Angle Cone (TRC)
   Specify the components of a radius vector **V** to the center of one base, the components of a vector **H** from the center of that base to the center of the other base, and the radii R1 and R2 of the lower and upper bases, respectively.

5. Torus (TOR)
   Specify the components of a radius vector **V** to the center of the torus, and the torus is configured parallel to one of the axis. R1 is the length between the center of torus and the center of tube, and R2 is the radius of the tube. Also, input the direction number of torus (n: x/y/z = 1/2/3). Furthermore, input starting angle $\theta 1$ and ending angle $\theta 2$ of the sector for the calculation of a part of torus. For the calculation of "complete" torus, set $\theta 1$=0, and $\theta 2$=$2pi$, respectively.

Table 1 Data required to described each body type.

Body Type	Inp. #	Real Data defining Particular Body					
RPP	#	Xmin	Xmax	Ymin	Ymax	Zmin	Zmax
SPH	#	Vx	Vy	Vz	R		
RCC	#	Vx	Vy	Vz	Hx	Hy	Hz
		R					
TRC	#	Vx	Vy	Vz	Hx	Hy	Hz
		R1	R2				
TOR	#	Vx	Vy	Vz	R1	R2	
		$\theta 1$	$\theta 2$	n			

2.2. Region Definition

The basic technique for description of the geometry consists of defining the location and shape of the various zones in term of the intersections and unions of the geometric bodies. A special operator notations involving the symbols (+), (−), and (OR) is used to describe the intersections

*Please see Appendix A of *JNC TN1410 2002-001* by T. Torii and T. Sugita[1].

and unions. These symbols are used by the program to construct information relating material descriptions to the body definitions.

If a body appears in a region description with a (+) operator, it means that the region being described is wholly contained in the body. If a body appears in a region description with a (−) operator, it means that the region being described is wholly outside the body. If body appears with an (OR) operator, it means that the region being described includes all points in the body. OR may be considered as a union operator. In some instances, a region may be described in terms of subregion lumped together by (OR) statements. Subregions are formed as intersects and then the region is formed by union of these subregions. When (OR) operators are used there are always two or more of them, and they refer to all body numbers following them, either (+) or (−). That is, all body numbers between "OR's" or until the end of the region cards for that region are intersected together before OR's are performed.

2.3. Example of Region Description

Consider an object composed of a sphere and a cylinder as shown in Fig. 1. To describe the object, one takes a spherical body (2) penetrated by a cylindrical body (3) (see Fig. 1). If the materials in the sphere and cylinder are the same, then they can be considered as one region, say region I (Fig. 1c). The description of region I would be

$$\mathrm{I} = +2\mathrm{OR} + 3.$$

This means that a point is in region I if it is either body 2 or inside body 3.

If different material are used in the sphere and cylinder, then the sphere with a cylindrical hole in it would be given a different region number (say J) from one cylinder (K).

The description of region J would be (Fig. 1d):

$$\mathrm{J} = +2 - 3.$$

This means that points in region J are all those points inside body 2 which are not inside body 3.

The description if region K is simply (Fig. 2e):

$$\mathrm{K} = +3.$$

That is, all points in region K lie inside body 3.

Combination of more than two bodies and similar region descriptions could contain a long string of (+), (−), and (OR) operators. It is important however to remember that **every spatial point in the geometry must be located in one and only one region.**

As a more complicated example of the use of the (OR) operator, consider the system shown in Fig. 2 consisting of the shared region A and the unshared region B. These regions can be described by the two BOX's, bodies 1 and 3, and the RCC, body 2. The region description would be

$$\mathrm{A} = +1 + 2$$

and

$$\mathrm{B} = +3 - 1\mathrm{OR} + 3 - 2.$$

Notice that OR operator refers to all following body numbers until the next OR operator is reached.

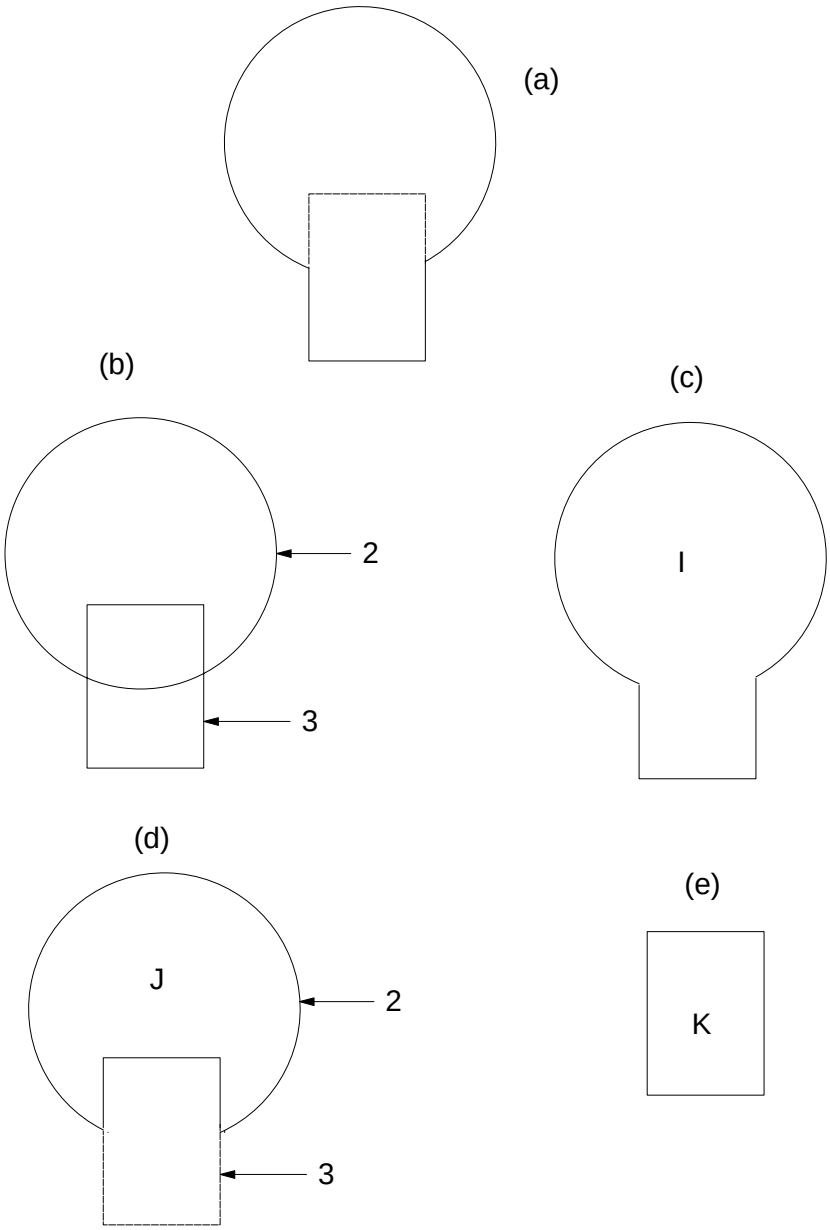

Figure 1: Examples of Combinatorial Geometry Method.

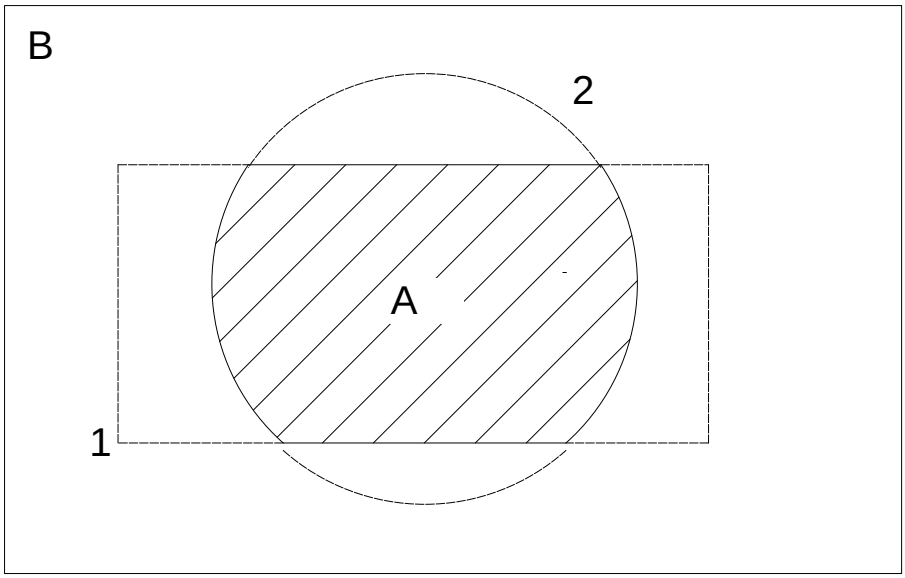

Figure 2: Use of OR operator.

3. Outlines of sample user code uccg_phantom.f

uccg_nai.f is the egs5 user code to calculate the same problem with ucrz_nai.f using cg. Input data of cg are written at the top of the input data from unit 4.

3.1. Input data for cg

Each region is defined by cylinder and planes in ucrz_nai.f. On the other hand, in uccg_nai.f, each region is defined by the combination of various size of cylinder in Fig. 3.

The input data for this geometry can be written as follows.

```
RCC     1 0.00        0.0          0.0          0.00        0.0        8.72
          4.41
RCC     2 0.00        0.0          0.1          0.00        0.0        8.12
          4.31
RCC     3 0.00        0.0          0.6          0.00        0.0        7.62
          3.81
RCC     4 0.00        0.0          8.22         0.00        0.0        0.5
          4.31
RCC     5 0.00        0.0         -1.0          0.00        0.0       10.0
          5.00
END
Z1              +1    -2    -4
Z2              +2    -3
Z3              +3
Z4              +4
Z5              +5    -1
END
```

1. Source condition

   - Source photon energy is sampled by using data read from unit 4 at subroutine getcg.
   - 1.332 MeV photon beam incident on the center of detector.

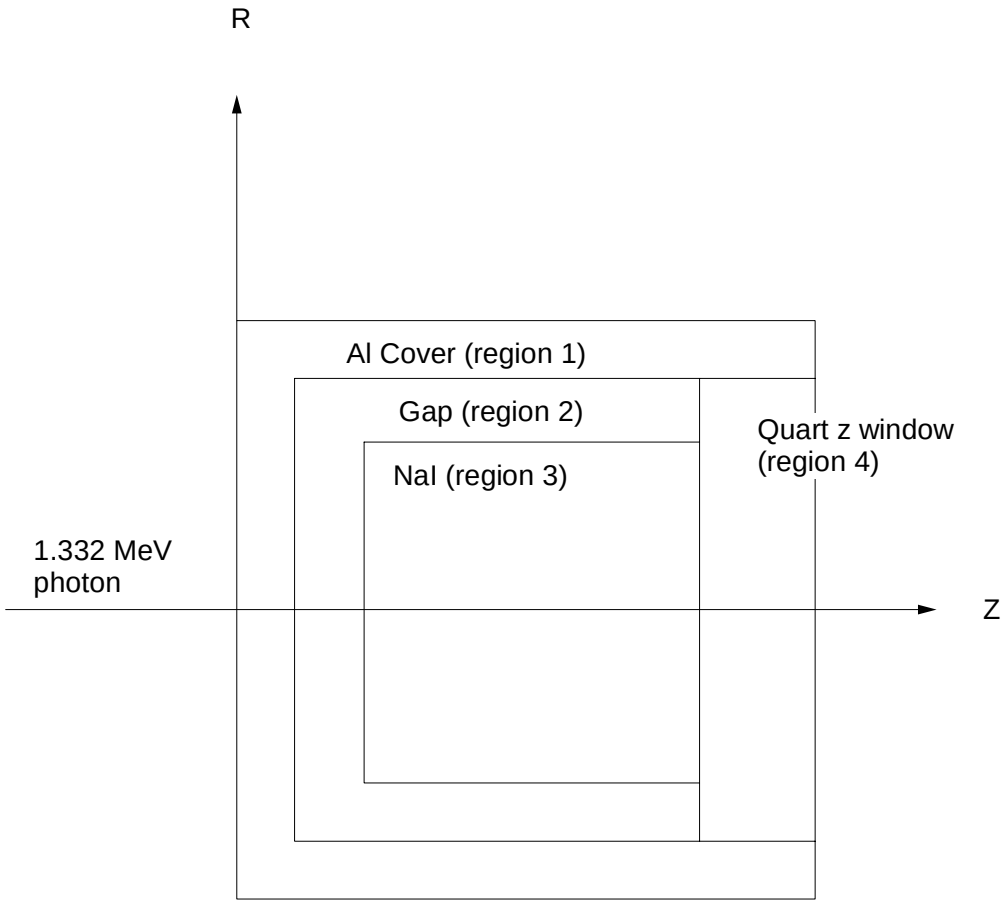

Figure 3: Geometry of uccg_nai.f

2. Results obtained

- Information of material used
- Material assignment to each region
- Peak and total efficiency
- Pulse height distribution
- Spectra of photon, electron and positron entering to NaI from outside

4. Details of user code

4.1. Main program

4.1.1. Include lines and specification statements: egs5 is written in Fortran 77. The size of arguments is defined at other files and included by using 'include line'. Various commons used inside egs5 are also included by the same way. Include files related directory with egs5 are put on the sub-directory ('include' directory) of egs5 directory (currently egs5.0). Those for each user including geometry related are put on the subdirectory ('user_auxcommon' directory) of user directory (currently kek_sample). These files are linked by running egs5run script.

This is the most different feature with EGS4 at which the side of arguments can be modified inside an user code with Mortran macro. If it is necessary to modify the side of arguments used in egs5, you must modify the related parameter in 'egs5.0/include/egs5_h.f'. The parameters related to each user are defined in 'kek_sampl/user_auxcommons/aux_h.f'.

First parts is include lines related egs5.

```
implicit none
```

```
!     ------------
!     EGS5 COMMONs
!     ------------
      include 'include/egs5_h.f'                ! Main EGS "header" file

      include 'include/egs5_edge.f'
      include 'include/egs5_media.f'
      include 'include/egs5_misc.f'
      include 'include/egs5_switches.f'
      include 'include/egs5_uphiot.f'
      include 'include/egs5_useful.f'
      include 'include/randomm.f'
```

`include 'include/egs5_h.f'` is always necessary. Other parts are only necessary when variables including at each common are used inside the main program.[†]

Next is include lines not directly related to egas5 like geometry related.

```
!     ----------------------
!     Auxiliary-code COMMONs
!     ----------------------
      include 'user_auxcommons/aux_h.f'   ! Auxiliary-code "header" file

      include 'user_auxcommons/edata.f'
      include 'user_auxcommons/etaly1.f'
      include 'user_auxcommons/instuf.f'
      include 'user_auxcommons/lines.f'
      include 'user_auxcommons/watch.f'

      include 'auxcommons/etaly2.f'      !  Added SJW for energy balance

!     ------------------
!     cg related COMMONs
!     ------------------
      include 'user_auxcommons/cg/tvalcg.f'
      include 'user_auxcommons/cg/zondta.f'
      include 'user_auxcommons/cg/rppdta.f'
      include 'user_auxcommons/cg/sphdtac.f'
      include 'user_auxcommons/cg/rccdta.f'
      include 'user_auxcommons/cg/trcdta.f'
      include 'user_auxcommons/cg/tordta.f'
```

Next `etaly2.f` is the semi-egs5 `common` and put at the `egs5.0/auxcommons` directory. The last 7 `include` statements are related to cg.

`common` used inside the user code is defined next.

```
      common/totals/                            ! Variables to score
     * depe(20),faexp,fexps,imode,ndet,nreg
      real*8 depe,faexp,fexps
      integer imode,ndet,nreg
```

By `implicit none` at the top, it is required to declare all data by a type declaration statement.

4.1.2. `Open` statement: At the top of executable statement, it is necessary to open units used in the user code. Due to the new feature that pegs is called inside each user code, it must be careful to the unit number used. The unit number from 7 to 26 are used inside 'pegs' and close at the end of 'pegs'. These units, therefore, must be re-open after calling pegs. It is better not to use these unit in the user code.

```
!     ----------
!     Open files
!     ----------
      open(UNIT= 4,FILE='egs5job.inp',STATUS='old')
      open(UNIT= 6,FILE='egs5job.out6',STATUS='unknown')
```

[†]This is corresponding to COMIN macros in EGS4.

4.1.3. `Call subroutine getcg`: Call `subroutine geomgt` to read cg input data and output cg related information.

Next subroutine is called to clear various counter parameters.

`Subroutine getcg` which is called next is the new subroutine used to run pegs as a part of user code and call `subroutine hatch`.

In the `subroutine getcg`, material used, egs5 cut-off energy, various option flag, geometry related data etc. will be set by reading data from unit 4.

```
!-----------------------------------------------
!     initialize cg related parameter
!-----------------------------------------------
      itbody=0
      irppin=0
      isphin=0
      irccin=0
      itorin=0
      itrcin=0
      izonin=0
      izonad=0
      itverr=0
      igmmax=0
      ifti = 4
      ifto = 6
      call geomgt(ifti,ifto,igmmax,itbody)

!-------------------------------
!     Get nreg from cg input data
!-------------------------------
      nreg=izonin
      if (nreg.gt.mxreg) then
        write(6,100) nreg,mxreg
100     FORMAT(' NREG(=',I12,') must be less than MXREG(=',I12,')' /' Yo
     *u must change MXREG in include/egs5_h.f.')
        stop
      end if

!     ====================
      call counters_out(0)
!     ====================

!     ================
      call getcg(nreg)
!     ================
```

4.1.4. Parameters setting and initialization: If `uin=vin=win=0.0`, `isot` is set to 1 as the flag for isotropic source.

An energy bin width is calculated from an incident kinetic energy and the number of bin.

Number of histories per batch (`ncaspb`) is calculated from batch number (`nbatch`) and number of histories (`ncases`). The uncertainty of calculated result is estimated from the deviation between the results at each batch.

```
      ndet=1


! ----------------------------------------
! Set isotropic source flag if uin=vin=win=0
! ----------------------------------------
      isot=0                ! monodirectional
      if (uin+vin+win.eq.0.0) isot=1

!     Energy bin width
      deltae=ekein / 50

!     Zero the variables
      depe=0.D0
      pef=0.D0
      tef=0.D0
```

```
      do j=1,50
        ph(J)=0.D0
        do nd=1,ndet
          spg(nd,j)=0.D0
          spe(nd,j)=0.D0
          spp(nd,j)=0.D0
        end do
      end do

!     Set number of batch and histories per batch
      nbatch = 50
      ncaspb = ncases / nbatch
      nofbat = 0
```

4.1.5. Transport calculation: Subroutine shower is called ncasepb times at each batch and repeated nbatch times.

Source energy is sampled based on the data read from unit 4 at subroutine getcg.

If some energy deposited at NaI, adds weight as total efficiency. If its energy is larger than 99.9% of source kinetic energy, treat as total absorption peak and adds weight as peak efficiency. Bin number corresponding absorbed energy is calculated to assign pulse height.

Average values for all variables are calculated at each batch.

```
                                          ! -------------------------
      do nofbat=1,nbatch                  ! Start of batch -loop
      do icases=1,ncaspb                  ! Start of CALL SHOWER loop
                                          ! -------------------------
!       ----------------------
!       Select incident energy
!       ----------------------
        eparte = 0.d0                  ! Initialize some energy-balance
        epartd = 0.d0                  !      tallying parameters (SJW)

          if (isamp .eq. 0) then          ! Monoenergetic case
            ekin = ekein
            wtin = 1.0
          else if (isamp .eq. 1) then     ! Sample discrete energy from CDF
            call randomset(rnnow)
            i=0
110         continue
            i = i + 1
            if(ecdf(i) .le. rnnow) go to 110
            ekin = ebin(i)
            wtin = 1.0
          else if (isamp .eq. 2) then     ! Sample DIRECTLY from CDF
            call edistr(ekin)
            wtin = 1.0
          else if (isamp .eq. 3) then     ! Sample UNIFORMLY on energy
            call randomset(rnnow)         ! interval and WEIGHT
            ekin = esam1 + rnnow*delsam
            isam = 0
120         continue
            isam = isam + 1
            if (ekin .lt. ebin(isam)) go to 130
            go to 120
130         continue
            wtin = epdf(isam)
          end if

        wtsum = wtsum + wtin              ! Keep running sum of weights
        etot = ekin + iabs(iqin)*RM       ! Incident total energy (MeV)
        availke = etot + iqin*RM       ! Available K.E. (MeV) in system
        totke = totke + availke                ! Keep running sum of KE

        if (isot.eq.1.0) then         ! Sample isotropically (forward only).
          call randomset(rnnow)
          win = 1.D0 - rnnow
          vin = sqrt(1.D0 - win*win)
```

```
        end if

!       ------------------------------------------------
!       Print first NWRITE or NLINES, whichever comes first
!       ------------------------------------------------
        if (ncount .le. nwrite .and. ilines .le. nlines) then
          ilines = ilines + 1
          write(6,140) etot,xin,yin,zin,uin,vin,win,iqin,irin,idin
140       FORMAT(4G15.7/3G15.7,3I5)
        end if

!       ============================================================
        call shower (iqin,etot,xin,yin,zin,uin,vin,win,irin,wtin)
!       ============================================================

!       Added for energy balance tests (SJW)
        if(DABS(eparte + epartd - ekin)/ekin .gt. 1.d-10) then
          write(6,150) icases, eparte, epartd
150       FORMAT('Error on # ',I6,' Escape = ',F9.5,' Deposit = ',F9.5)
        endif

!      If some energy is deposited inside detector add pulse-height
!      and efficiency.

       if (depe .gt. 0.D0) then
         ie=depe/deltae + 1
         if (ie .gt. 50)  ie = 50
         ph(ie)=ph(ie)+wtin
         tef=tef + wtin
         if(depe .ge. ekin*0.999) pef=pef +wtin
         depe = 0.D0
       end if

        ncount = ncount + 1         ! Count total number of actual cases

!                           ======================
        if (iwatch .gt. 0) call swatch(-1,iwatch)
!                           ======================

                                                  ! ------------------------
      end do                                      ! End of CALL SHOWER loop
                                                  ! ------------------------

!  Calculate average value for this BATCH
      do ie=1,50
        phpb(ie,nofbat) = ph(ie) /ncaspb
        ph(ie)=0.D0
      end do
      pefpb(nofbat)=pef / ncaspb
      tefpb(nofbat)=tef /ncaspb
      pef=0.D0
      tef=0.D0
      do nd=1,ndet
        do ie=1,50
          spgpb(nd,ie,nofbat)=spg(nd,ie)/ncaspb !photon spectrum
          spepb(nd,ie,nofbat)=spe(nd,ie)/ncaspb !electron spectrum
          spppb(nd,ie,nofbat)=spp(nd,ie)/ncaspb !positron spectrum
          spg(nd,ie)=0.D0
          spe(nd,ie)=0.D0
          spp(nd,ie)=0.D0
        end do
      end do
                                                  ! -------------------
      end do                                      ! End of batch loop
                                                  ! -------------------
```

4.1.6. Statistical uncertainty: The uncertainty of obtained, x, is estimated using the method used in MORCE-CG in this user code.

- Assume that the calculation calls for N "incident" particle histories.
- Split the "N" histories into n statistical batches of N/n histories each. The calculated quantity for each of these batches is called x_i.
- Calculate the mean value of x:

$$\overline{x} = \frac{1}{N}\sum_{i=1}^{n} x_i \tag{1}$$

- Estimate the variance associate with the distribution of the x_i:

$$s_x^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i^2 - \overline{x}^2) \tag{2}$$

- The estimated variance of $\overline{x}$ is the standard variance of the mean:

$$s_{\overline{x}}^2 = \frac{s_x^2}{n} \tag{3}$$

- Report FSD(fractional standard deviation) as the statistical error:

$$\mathrm{FSD} = s_{\overline{x}}/\overline{x} \tag{4}$$

4.1.7. Output of results: After finishing all histories, obtained results are analyzed and written on output file. Average values and their statistical uncertainty `FSD` are calculated form the average results per batch.

```
!     ------------------------------------
!     Calculate average and its deviation
!     ------------------------------------

!     ---------------
!     Peak efficiency
!     ---------------
      avpe = 0.D0
      desci2 = 0.D0
      do j = 1, nbatch
        avpe = avpe + pefpb(j)/nbatch
        desci2 = desci2 + pefpb(j)*pefpb(j)/nbatch
      end do
      sigpe = sqrt((desci2 - avpe*avpe)/(nbatch-1))
      avpe = avpe*100.0
      sigpe = sigpe*100.0
      write(6,210) avpe,sigpe
210   FORMAT(' Peak efficiency =',G15.5,'+-',G15.5,' %')


!     ----------------
!     Total efficiency
!     ----------------
      avte = 0.D0
      desci2 = 0.D0
      do j = 1, nbatch
        avte = avte + tefpb(j)/nbatch
        desci2 = desci2 + tefpb(j)*tefpb(j)/nbatch
      end do
      sigte = sqrt((desci2 - avte*avte)/(nbatch-1))
      avte = avte*100.0
      sigte = sigte*100.0
      write(6,220) avte,sigte
220   FORMAT(' Total efficiency =',G15.5,'+-',G15.5,' %')

!     -------------------------
!     Pulse height distribution
!     -------------------------
```

```
      write(6,230)
230   FORMAT(/' Pulse height distribution ')
      do ie=1,50
        elow=deltae*(ie-1)
        eup=deltae*ie
        if (elow .gt. ekein ) go to 990

        avph = 0.D0
        desci2 = 0.D0
        do j = 1, nbatch
          avph = avph + phpb(ie,j)/nbatch
          desci2 = desci2 + phpb(ie,j)*phpb(ie,j)/nbatch
        end do
        sigph = sqrt((desci2 - avph*avph)/(nbatch-1))
        avph = avph/deltae
        sigph= sigph/deltae
        write(6,240) eup,avph,sigph
240     FORMAT(' E (upper-edge --',G10.4,' MeV )=',G15.5,'+-',G15.5,
     *         ' counts/MeV/incident');
      end do

990   continue
```

Spectra of particles incident on NaI detector are also analyzed and output.

4.2. Subroutine getcg

`Subroutine getcg` used to define material used, its density, egs5 cut-off energy, various optional flag applied to each region, data for source particle etc. and call `subroutine hatch`.

The data read from unit 4 are as follows.

1. Record 1 : Title (within 80 characters)
2. Record 2 : Number of media in problem (nmed)
3. Record 3 : Media names (j=1,24, i=1,nmed lines)
4. Record 4 : Set material for region from irlinl to ielinh.
   medtmp : material number
   rhotmp : If rhotmp=0.0, the default value for that medium is used.
   ecutin, pcutin : KINETIC energy cutoffs for electrons and photons, respectively, in MeV. If > 0, ecut(i) and pcut(i) are set. Otherwise ae and ap are used (default).
   If medium not 0, following option is set to the regions above. (0: off, 1:on)
5. Record 4a : irlinl,irlinu,,medtmp, rhotmp, ecutin, pcutin
   - ipeangsw Switches for PE-angle sampling
   - iedgesw K & L-edge fluorescence
   - iraysw Rayleigh scattering
   - ipolarsw Linearly-polarized photon scattering
   - incohrsw S /Z rejection
   - iprofrsw Doppler broadening
   - mpacrsw electron impact ionization
6. Record 5 : Incident X,Y,Z coordinates (cm) (xin, yin, zin)
7. Record 6 : Incident region
8. Record 7 : Incident direction cosines (uin,vin,win) If uin=vin=win=0, it means isotropic source.
9. Record 8 : Starting random number seeding.
   If ixx = 0, ixx is set to 123457.
   If jxx = 0, jxx is set to 654321.
10. Record 9 : Number of cases (ncases).

11. Record 10 : Kinetic energy (MeV), charge of incident beam, and sampling switch. If isamp=0, a monoenergetic beam (ekein) will be used. Otherwise, a spectrum input must follow (Records 14a through 14b), which will be sampled from discrete energy (isamp=1), directly (isamp=2) or uniformly over the energy range (isamp=3) with weighting factor.

12. Record 10a : Only required when $isamp > 1$ (see above).

13. Record 10b : Only required when usamp¿0(see above). ebin(i) is the 'top-edge' of each energy bin (MeV) and epdf(i) is the corresponding probability for the bin. For example, a cross section (mb) can be used for epdf (but do not divide it by dE). The last card is a delimiter and should be blank (or contain 0.0). The i-subscript runs from 1 to nebin (nebin calculated after the delimiter).

14. Record 11 : Switch for tracking events with swatch: (0=No, 1=each interaction, 2=each step)

15. Record 12 : Switches for bremsstrahlung and pair production ANGLE SAMPLING, and brems-strahlung SPLITTING:

    ibrdst=0 No (use default: theta=m/E)
    ibrdst=1 Yes (recommended)
    iprdst=0 No (use default: theta=m/E)
    iprdst=1 Yes (low-order distribution)
    iprdst=2 Yes (recommended)
    ibrspl=0 No splitting
    ibrspl=1 Apply splitting (nbrspl=splitting factor)

16. Record 18 : Parameters used for charged particle transport (estepe,estepe2).

4.3. `Subroutine ausgab`

`Subroutine ausgab` is a subroutine to score variables that user want to calculate.

Include lines and specification statements are written at first by the same way used at the main program/

After the treatment related `iwatch` option, value of the stack number (np) is checked not to exceed the pre-set maximum value.

When iarg < 5, absorbed energy at the region nreg (outside the system) and other regions are summed separately to check energy balance at each history.

If the material number 1, NaI region, absorbed energy per step is added as the energy deposition at the detector.

If a particle enters to NaI region from outside, score energy information corresponding to each particle type.

```
!     ------------------------
!     Set some local variables
!     ------------------------
      irl = ir(np)
      iql = iq(np)
      edepwt = edep*wt(np)

!     ------------------------------------------------------------
!     Keep track of energy deposition (for conservation purposes)
!     ------------------------------------------------------------
      if (iarg .lt. 5) then
        esum(iql+2,irl,iarg+1) = esum(iql+2,irl,iarg+1) + edepwt
        nsum(iql+2,irl,iarg+1) = nsum(iql+2,irl,iarg+1) + 1


!  added SJW for particle by particle energy balance
        if(irl.eq.nreg) then
          eparte = eparte + edepwt
        else
          epartd = epartd + edepwt
        endif
      end if
```

```
!     ------------------------------------------
!     Score energy deposition inside NaI detector
!     ------------------------------------------
      if (med(irl). eq. 1) then
        depe = depe + edepwt

!      ------------------------------------------------
!      Score particle information if it enters from outside
!      ------------------------------------------------
       if (irl .ne. irold .and. iarg .eq. 0) then
         if (iql .eq. 0) then             ! photon
           ie = e(np)/deltae +1
           if(ie .gt. 50) ie = 50
           spg(1,ie) = spg(1,ie) + wt(np)
         elseif (iql .eq. -1) then        ! electron
           ie = (e(np) - RM)/deltae +1
           if(ie .gt. 50) ie = 50
           spe(1,ie) = spe(1,ie) + wt(np)
         else                             ! positron
           ie = (e(np) - RM)/deltae +1
           if(ie .gt. 50) ie = 50
           spp(1,ie) = spp(1,ie) + wt(np)
         end if
       end if
      end if


!     ---------------------------------------------------------------
!     Print out stack information (for limited number cases and lines)
!     ---------------------------------------------------------------
      if (ncount .le. nwrite .and. ilines .le. nlines) then
        ilines = ilines + 1
        write(6,101) e(np),x(np),y(np),z(np),u(np),v(np),w(np),
     *               iql,irl,iarg
 101    FORMAT(4G15.7/3G15.7,3I5)
      end if

!     ---------------------------------------------------------------
!     Print out particle transport information (if switch is turned on)
!     ---------------------------------------------------------------
!                          ========================
      if (iwatch .gt. 0) call swatch(iarg,iwatch)
!                          ========================

      return

      end
```

4.4. Subroutine howfar

At `subroutine howfar`, a distance to the boundary of region is checked. If the distance to the boundary is shorter than the distance to the next point, the distance to the next point is replaced with the distance to the boundary and new region `irnew` is set to the region number to which particle will enter.

If `idisc` is set to 1 by user, the treatment to stop following will be done in this subroutine.

Calculation to a distance to the boundary is done by using the various subroutines related cg in `uccg_nai.f`.

5. Conmparison of speed between ucrz_nai.f and uccg_nai.f

Cg geometry is suitable to treat a complex geometry than the cylinder-plane geometry etc. On the other hand, cg needs more cpu time. For example, uccg_nai.f needs 2.1 times longer cpu time than ucrz_nai.f for the same problem.[‡]

[‡]Sppedup for CG almost about 1.3 in this case, whic uses relatively small number of bodies, was provided by T. Sugita.

6. Exercise problems

6.1. Problem 1 : Calculation for NaI detector

Study variation by changing input data at the following cases.

1. Change the source to 0.662 MeV photons from ^{137}Cs.
2. Change source energy to 1.173 and 1.332 MeV photons from ^{60}Co.
3. Increase detector thickness twice for ^{60}Co source.
4. Change to isotropic source for ^{137}Cs.

6.2. Problem 2 : Ge detector calculation

Change detector to Ge from NaI and compare its peak and total efficiencies with NaI detector of same size for ^{137}Cs source.

6.3. Problem 3 : Air ionization chamber calculation

Change detector to air at 20° and 1 atm and calculate absorbed energy for ^{137}Cs source. Air region have 3.81 cm diameter and 7.62 cm length and is surrounded by 0.1 cm aluminum wall.

Calculate output of this chamber (Coulomb/source) using W-value of air and (33.97eV/pair) and the electron charge magnitude 1.602×10^{-19}C/e.

7. Answer for exercise

7.1. Problem 1

1. ^{137}Cs source
   - Change `ekein` value to 0.662 at 36 lines of `uccg_nai.data`.
   - Save `uccg_nai.data` as the different name and assign as the file name for unit 4.
2. ^{60}Co source
   - Change `isamp` to 1 at 36 lines of `uccg_nai.data`.
   - Add following data after 36 lines.

```
1.117,    1.0              discrete energy 1
1.332,    1.0,             discrete energy 2
0.0,      0.0,             end of set energy
```

   - Save `uccg_nai.data` as the different name and assign as the file name for unit 4.
3. Increase NaI detector length twice for ^{60}Co
   - Change cg-related data in data file created for above problem as follows.

```
RCC    1 0.00      0.0        0.0        0.00      0.0      16.44
         4.41
RCC    2 0.00      0.0        0.1        0.00      0.0      15.84
         4.31
RCC    3 0.00      0.0        0.6        0.00      0.0      15.34
         3.81
RCC    4 0.00      0.0       15.94       0.00      0.0       0.5
         4.31
RCC    5 0.00      0.0       -1.0        0.00      0.0      20.0
         5.00
END
Z1           +1   -2   -4
Z2           +2   -3
Z3           +3
Z4           +4
Z5           +5   -1
END
```

- Save as the different name and assign as the file name for unit 4.

4. Point isotropic source
   - Change `win` value at 33 line to 0.0 and `ekein` value to 0.662 at 36 lines of `uccg_nai.data`.
   - Save `uccg_nai.data` as the different name and assign as the file name for unit 4.

7.2. Problem 2

1. Replace `NAI-IAPRIM` at 20 lines at `uccg_nai.data` to `GE-IAPRIM`.
2. Save `uccg_nai.data` as the different name and assign as the file name for unit 4.
3. Replace NaI related data from 1 to 13 lines of `uccg_nai.inp` to the following data.

```
ELEM
 &INP IAPRIM=1,EFRACH=0.05,EFRACL=0.20,
      IRAYL=1,IBOUND=0,INCOH=0,ICPROF=0,IMPACT=0 /END
GE-IAPRIM                      GE
GE
ENER
 &INP AE=0.521,AP=0.0100,UE=2.511,UP=2.0 /END
TEST
 &INP  /END
PWLF
 &INP  /END
DECK
 &INP  /END
```

4. Save `uccg_nai.inp` as the different name and assign as the file name for unit 25.

7.3. Problem 3

1. Modify `uccg_nai.f` as follows.
   - Add `depepb(50)` as real*8 local variable.
   - Change write statement concerning geometry as follows.
   - Change write statement concerning detector as follows,

```
      tdet=7.62
      rdet=3.81
      tcov=0.1
      rtcov=0.1
      write(6,200) tdet,rdet,tcov,rtcov
200   FORMAT(/' Detector length=',G15.5,' cm'/
     *       ' Detector radius=',G15.5,' cm'/
     *       ' Al cover thickness=',G10.2,' cm'/
     *       ' Al cover side thickness=',G10.2,' cm'/)
```

   - Add routines to calculate average absorbed energy and its `FSD` at air region.

```
!     ----------------------
!     Absorbed energy in air
!     ----------------------
      avab = 0.D0
      desci2 = 0.D0
      do j = 1, nbatch
        avab = avab + depepb(j)/nbatch
        desci2 = desci2 + depepb(j)*depepb(j)/nbatch
      end do
      sigab = sqrt((desci2 - avab*avab)/(nbatch-1))
      write(6,210) avab,sigab
210   FORMAT(' Absorbed energy in air =',G15.5,'+-',G15.5,' MeV/photon')
      avab = avab /33.97D-6 *1.602D-19
      sigab= sigab /33.97D-6 *1.602D-19
      write(6,215) avab,sigab
215   FORMAT(' Output current =',G15.5,'+-',G15.5,' C/photon')
```

- Add `avab,sigav` to local variables as `real*8`.

2. Make input data file for unit 4 as follows.

```
  RCC    1 0.00      0.0        0.0        0.00      0.0      7.82
           3.81
  RCC    2 0.00      0.0        0.1        0.00      0.0      7.72
           3.92
  RCC    3 0.00      0.0       -1.0        0.00      0.0     10.0
           5.00
  END
  Z1           +1   -2
  Z2           +2
  Z3           +3   -1
  END
0.662 MeV photon on Air ionization chamber
        2                          nmed
AIR-AT-NTP-IAPRIM                  media(j,1) (24A1)
AL-IAPRIM                          media(j,2) (24A1)
    1    1    2    0.     0.561      0.0      Al
    1    1    0    0    0    0    0   peang,edge,ray,pola,incoh,prof,impac
    2    2    1    0.     0.561     0.0       Air
    1    1    0    0    0    0    0   peang,edge,ray,pola,incoh,prof,impac
    0                                    end of define
       0.0       0.0       0.0 xin,yin,zin
         1                          irin
       0.0       0.0       1.0 uin,vin,win
       0      0                   ixx, jxx
    100000                         ncases (I10)
    0.662        0         0    ekein(mev),iqin,isamp
    0                              iwatch
    1    2    0    0              ibrdst,iprdst,ibrspl,nbrspl
      0.10      0.20              estepe and estepe2
```

3. Make data file for unit 25 as follows.

```
MIXT
 &INP NE=3,RHO= 1.2050E-03,RHOZ= 0.78,0.2103,0.0094,IAPRIM=1,
      EFRACH=0.05,EFRACL=0.20,IRAYL=1,IBOUND=0,INCOH=0,
      ICPROF=0,IMPACT=0 /END
AIR-AT-NTP-IAPRIM             AIR-GAS
N  O  AR
ENER
 &INP AE=0.521,AP=0.010,UE=2.511,UP=2.0 /END
PWLF
 &INP  /END
DECK
 &INP  /END
ELEM
 &INP IAPRIM=1,EFRACH=0.05,EFRACL=0.20,
      IRAYL=1,IBOUND=0,INCOH=0,ICPROF=0,IMPACT=0 /END
AL-IAPRIM                     AL
AL
ENER
 &INP AE=0.521,AP=0.010,UE=2.511,UP=2.0 /END
TEST
 &INP  /END
PWLF
 &INP  /END
DECK
 &INP  /END
```

References

[1] T. Torii and T. Sugita, "Development of PRESTA-CG Incorprating Combinatorial Geometry in EGS4/PRESTA", *JNC TN1410 2002-201*, Japan Nuclear Cycle Development Institute (2002).

Appendix 1 Full listings of uccg_nai.f

```
!*********************************************************************
!****************************  KEK, High Energy Accelerator Research  *
!****************************  Organization                           *
!*** u c c g _ n a i *********                                        *
!****************************     EGS5.0 USER CODE - 28 JUL 2004/1300 *
!*********************************************************************
!* This is a general User Code based on the cg geometry scheme.       *
!*********************************************************************
!                                                                     *
!  PROGRAMMERS:  H. Hirayama                                          *
!                Radiation Science Center                             *
!                Applied Science Laboratory                           *
!                KEK, High Energy Accelerator Research Organization   *
!                1-1, Oho, Tsukuba, Ibaraki, 305-0801                 *
!                Japan                                                *
!                                                                     *
!                E-mail:      hideo.hirayama@kek.jp                   *
!                Telephone:   +81-29-864-5489                         *
!                Fax:         +81-29-864-1993                         *
!                Based on ucrtz_sampl4 by Nelson and James.           *
!                                                                     *
!*********************************************************************
!*********************************************************************
! The uccg_nai.f User Code requires a data-input file                 *
! (e.g., uccg_nai.data) that is read by subroutine getcg (with        *
! instructions in its header).  The following shows the geometry for  *
! uccg_nai.data.                                                      *
! Input data for CG geometry must be written at the top of data-input *
! file. This user code corresponds to ucnai3cgp.mor for egs4.         *
!*********************************************************************
!                                                                     *
!             -------------------------------                         *
!             cg Geometry (uccg_nai example)                          *
!             -------------------------------                         *
!                                                                     *
!                                                                     *
!                R                                                    *
!                ^                                                    *
!                |                                                    *
!           +----+----+----+--------+------+---                       *
!           |                                                         *
!           |     Outer vacuum region                                 *
!           +    +----+----+--------+------+--- r=4.41 cm             *
!           |    |              Al            |                       *
!           |    +    +----+---------+------+--- R=4.31              *
!           |    |    |        Gap    |      |                       *
!           +    +    +    +---------+      +--- R=3.81              *
!           |    |    |    |         |      |                         *
!           |    | Al | Gap|   NaI   |Quartz|                         *
!  .33 MeV  |    |    |    |         |      |                         *
!      ==========>+---+----+---------+------+--------> Z              *
!     photons    0   0.1  0.6       8.22   8.72 cm                    *
!                                                                     *
!                                                                     *
!                                                                     *
!                                                                     *
!*********************************************************************
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12

!---------------------------------------------------------------------
!------------------------------ main code ----------------------------
!---------------------------------------------------------------------

      implicit none

!     ------------
!     EGS5 COMMONs
!     ------------
      include 'include/egs5_h.f'                ! Main EGS "header" file

      include 'include/egs5_edge.f'
      include 'include/egs5_media.f'
      include 'include/egs5_misc.f'
      include 'include/egs5_switches.f'
      include 'include/egs5_uphiot.f'
      include 'include/egs5_useful.f'
      include 'include/randomm.f'
```

```
!     ----------------------
!     Auxiliary-code COMMONs
!     ----------------------
      include 'user_auxcommons/aux_h.f'   ! Auxiliary-code "header" file

      include 'user_auxcommons/edata.f'
      include 'user_auxcommons/etaly1.f'
      include 'user_auxcommons/instuf.f'
      include 'user_auxcommons/lines.f'
      include 'user_auxcommons/watch.f'

      include 'auxcommons/etaly2.f'      !  Added SJW for energy balance

!     ------------------
!     cg related COMMONs
!     ------------------
      include 'user_auxcommons/cg/tvalcg.f'
      include 'user_auxcommons/cg/zondta.f'
      include 'user_auxcommons/cg/rppdta.f'
      include 'user_auxcommons/cg/sphdtac.f'
      include 'user_auxcommons/cg/rccdta.f'
      include 'user_auxcommons/cg/trcdta.f'
      include 'user_auxcommons/cg/tordta.f'

      common/totals/                          ! Variables to score
     * depe,deltae,spg(1,50),spe(1,50),spp(1,50),nreg
      real*8 depe,deltae,spg,spe,spp
      integer nreg

!**** real*8                                                  ! Arguments
      real*8 totke
      real*8 rnnow,etot
      real*8 esumt

      real*8                                              ! Local variables
     * availke,avpe,avph,avspe,avspg,avspp,avte,desci2,ekin,pef,
     * sigpe,sigte,sigph,sigspg,sigspe,sigspp,tef,wtin,wtsum

      real*8
     * ph(50),phpb(50,50),spgpb(1,50,50),spepb(1,50,50),
     * spppb(1,50,50),pefpb(50),tefpb(50)

      real                                                ! Local variables
     * elow,eup,rdet,rtcov,rtgap,tcov,tdet,tgap

      real
     * tarray(2),tt,tt0,tt1,cputime

      integer
     * icases,idin,isam,isot,nlist,
     * i,j,k,ireg,n,imed,ndet,nd,nbatch,ncaspb,nofbat,ie,
     * itbody,izonad,
     * igmmax,ifti,ifto


!     ----------
!     Open files
!     ----------
      open(UNIT= 4,FILE='egs5job.inp',STATUS='old')
      open(UNIT= 6,FILE='egs5job.out6',STATUS='unknown')

!-----------------------------------------------
!     initialize cg related parameter
!-----------------------------------------------
      itbody=0
      irppin=0
      isphin=0
      irccin=0
      itorin=0
      itrcin=0
      izonin=0
      izonad=0
      itverr=0
      igmmax=0
      ifti = 4
      ifto = 6
      call geomgt(ifti,ifto,igmmax,itbody)

!-------------------------------
!     Get nreg from cg input data
```

```
!-------------------------------
      nreg=izonin
      if (nreg.gt.mxreg) then
        write(6,100) nreg,mxreg
100     FORMAT(' NREG(=',I12,') must be less than MXREG(=',I12,')' /
     *  ' You must change MXREG in include/egs5_h.f.')
        stop
      end if

!     ====================
      call counters_out(0)
!     ====================

!     ================
      call getcg(nreg)
!     ================

      ncount = 0
      ilines = 0
      nwrite = 10
      nlines = 10
      idin = -1
      totke = 0.
      wtsum = 0.


!     =========================
      call ecnsv1(0,nreg,totke)
      call ntally(0,nreg)
!     =========================

      write(6,110)
110   FORMAT(//,' ENERGY/COORDINATES/DIRECTION COSINES/ETC.',/,
     *        6X,'E',16X,'X',14X,'Y',14X,'Z'/
     *        1X,'U',14X,'V',14X,'W',9X,'IQ',4X,'IR',3X,'IARG',/)

!                         ======================
      if (iwatch .gt. 0) call swatch(-99,iwatch)
!                         ======================

      ndet=1


! ----------------------------------------
! Set isotropic source flag if uin=vin=win=0
! ----------------------------------------
      isot=0                  ! monodirectional
      if (uin+vin+win.eq.0.0) then
        isot=1
        write(6,115)
115     FORMAT(' Isotropic source')
      end if

!     Energy bin width
      deltae=ekein / 50

!     Zero the variables
      depe=0.D0
      pef=0.D0
      tef=0.D0
      do j=1,50
        ph(J)=0.D0
        do nd=1,ndet
          spg(nd,j)=0.D0
          spe(nd,j)=0.D0
          spp(nd,j)=0.D0
        end do
      end do

!     Set number of batch and histories per batch
      nbatch = 50
      ncaspb = ncases / nbatch
      nofbat = 0

      tt=etime(tarray)
      tt0=tarray(1)

                                            ! -------------------------
      do nofbat=1,nbatch                    ! Start of batch -loop
      do icases=1,ncaspb                    ! Start of CALL SHOWER loop
```

```
                                          ! -------------------------
!       ----------------------
!       Select incident energy
!       ----------------------
        eparte = 0.d0                   ! Initialize some energy-balance
        epartd = 0.d0                   !      tallying parameters (SJW)

        if (isamp .eq. 0) then                      ! Monoenergetic case
          ekin = ekein
          wtin = 1.0
        else if (isamp .eq. 1) then     ! Sample discrete energy from CDF
          call randomset(rnnow)
          i=0
120       continue
          i = i + 1
          if(ecdf(i) .le. rnnow) go to 120
          ekin = ebin(i)
          wtin = 1.0
        else if (isamp .eq. 2) then     ! Sample DIRECTLY from CDF
          call edistr(ekin)
          wtin = 1.0
        else if (isamp .eq. 3) then     ! Sample UNIFORMLY on energy
          call randomset(rnnow)         ! interval and WEIGHT
          ekin = esam1 + rnnow*delsam
          isam = 0
130       continue
          isam = isam + 1
          if (ekin .lt. ebin(isam)) go to 140
          go to 130
140       continue
          wtin = epdf(isam)
        end if

        wtsum = wtsum + wtin              ! Keep running sum of weights
        etot = ekin + iabs(iqin)*RM       ! Incident total energy (MeV)
        availke = etot + iqin*RM        ! Available K.E. (MeV) in system
        totke = totke + availke                 ! Keep running sum of KE

        if (isot.eq.1) then          ! Sample isotropically (forward only).
          call randomset(rnnow)
          win = 1.D0 - rnnow
          vin = sqrt(1.D0 - win*win)
        end if

!       ------------------------------------------------
!       Print first NWRITE or NLINES, whichever comes first
!       ------------------------------------------------
        if (ncount .le. nwrite .and. ilines .le. nlines) then
          ilines = ilines + 1
          write(6,150) etot,xin,yin,zin,uin,vin,win,iqin,irin,idin
150       FORMAT(4G15.7/3G15.7,3I5)
        end if

!       ========================================================
        call shower (iqin,etot,xin,yin,zin,uin,vin,win,irin,wtin)
!       ========================================================

!       Added for energy balance tests (SJW)
        if(DABS(eparte + epartd - ekin)/ekin .gt. 1.d-10) then
          write(6,160) icases, eparte, epartd
160       FORMAT('Error on # ',I6,' Escape = ',F9.5,' Deposit = ',F9.5)
        endif

!       If some energy is deposited inside detector add pulse-height
!       and efficiency"

        if (depe .gt. 0.D0) then
          ie=depe/deltae + 1
          if (ie .gt. 50)  ie = 50
          ph(ie)=ph(ie)+wtin
          tef=tef + wtin
          if(depe .ge. ekin*0.999) pef=pef +wtin
          depe = 0.D0
        end if

        ncount = ncount + 1         ! Count total number of actual cases

!                           ======================
        if (iwatch .gt. 0) call swatch(-1,iwatch)
```

```
!                          ======================

                                                    ! ------------------------
      end do                                        ! End of CALL SHOWER loop
                                                    ! ------------------------

!  Calcurate average value for this BATCH
      do ie=1,50
        phpb(ie,nofbat) = ph(ie) /ncaspb
        ph(ie)=0.D0
      end do
      pefpb(nofbat)=pef / ncaspb
      tefpb(nofbat)=tef /ncaspb
      pef=0.D0
      tef=0.D0
      do nd=1,ndet
        do ie=1,50
          spgpb(nd,ie,nofbat)=spg(nd,ie)/ncaspb !photon spectrum
          spepb(nd,ie,nofbat)=spe(nd,ie)/ncaspb !electron spectrum
          spppb(nd,ie,nofbat)=spp(nd,ie)/ncaspb !positron spectrum
          spg(nd,ie)=0.D0
          spe(nd,ie)=0.D0
          spp(nd,ie)=0.D0
        end do
      end do
                                                    ! -------------------
      end do                                        ! End of batch loop
                                                    ! --------------------

      tt=etime(tarray)
      tt1=tarray(1)
      cputime=tt1-tt0
      write(6,170) cputime
170   format(/' Elapsed Time (sec)=',G15.5)

!                        ======================
      if (iwatch .gt. 0) call swatch(-88,iwatch)
!                        ======================

!     ---------------------
!     Write out the results
!     ---------------------
      write(6,180) ncount,ncases,totke,iseed1,iseed2
180   FORMAT(//,' Ncount=',I10,' (actual cases run)',/,
     *       ' Ncases=',I10,' (number of cases requested)',/,
     *       ' TotKE =',G15.5,' (total KE (MeV) in run)'/
     *       ' Last iseed1 =',I12,', iseed2 =',I12)

      if (totke .le. 0.D0) then
        write(6,190) totke,availke,ncount
190     FORMAT(//,' Stopped in MAIN with TotKE=',G15.5,/,
     *         ' AvailKE=',G15.5, /,' Ncount=',I10)
        stop
      end if

      tdet=7.62
      rdet=3.81
      tcov=0.1
      rtcov=0.1
      tgap=0.5
      rtgap=0.5
      write(6,200) tdet,rdet,tcov,rtcov,tgap,rtgap
200   FORMAT(/' Detector length=',G15.5,' cm'/
     *       ' Detector radius=',G15.5,' cm'/
     *       ' Al cover thickness=',G10.2,' cm'/
     *       ' Al cover side thickness=',G10.2,' cm'/
     *       ' Front gap =',G10.2,' cm'/' Side gap =',G10.2,' cm'/)

      if (isamp.eq.0) then
        write(6,210) ekin
210     FORMAT(' Results for ',G15.5,'MeV photon'/)
      else if (isamp.eq.1) then
        write(6,212) ekein
212     FORMAT(' Source eneygy is sampled from discrete ons.'/
     *         ' Higest energy is ',G15.5,'MeV'/)
      else if (isamp.eq.2) then
        write(6,214)
214     FORMAT(' Source eneygy is sampled DIRECTLY from CDF'/)
```

```
      else
        write(6,216)
216     FORMAT(' Source eneygy is sampled UNIFORMLY on energy interval'/
     *         ' and use Weight'/)
      end if

!     ----------------------------------
!     Calculate average and its deviation
!     ----------------------------------

!     ---------------
!     Peak efficiency
!     ---------------
      avpe = 0.D0
      desci2 = 0.D0
      do j = 1, nbatch
        avpe = avpe + pefpb(j)/nbatch
        desci2 = desci2 + pefpb(j)*pefpb(j)/nbatch
      end do
      sigpe = sqrt((desci2 - avpe*avpe)/(nbatch-1))
      avpe = avpe*100.0
      sigpe = sigpe*100.0
      write(6,220) avpe,sigpe
220   FORMAT(' Peak efficiency =',G15.5,'+-',G15.5,' %')


!     ----------------
!     Total efficiency
!     ----------------
      avte = 0.D0
      desci2 = 0.D0
      do j = 1, nbatch
        avte = avte + tefpb(j)/nbatch
        desci2 = desci2 + tefpb(j)*tefpb(j)/nbatch
      end do
      sigte = sqrt((desci2 - avte*avte)/(nbatch-1))
      avte = avte*100.0
      sigte = sigte*100.0
      write(6,230) avte,sigte
230   FORMAT(' Total efficiency =',G15.5,'+-',G15.5,' %')

!     --------------------------
!     Pulse height distribution
!     --------------------------
      write(6,240)
240   FORMAT(/' Pulse height distribution ')
      do ie=1,50
        elow=deltae*(ie-1)
        eup=deltae*ie
        if (elow .gt. ekein ) go to 260

        avph = 0.D0
        desci2 = 0.D0
        do j = 1, nbatch
          avph = avph + phpb(ie,j)/nbatch
          desci2 = desci2 + phpb(ie,j)*phpb(ie,j)/nbatch
        end do
        sigph = sqrt((desci2 - avph*avph)/(nbatch-1))
        avph = avph/deltae
        sigph= sigph/deltae
        write(6,250) eup,avph,sigph
250     FORMAT(' E (upper-edge --',G10.4,' MeV )=',G15.5,'+-',G15.5,
     *         ' counts/MeV/incident');
      end do

260    continue

!     ------------------------------------------------------------
!     Particle spectrum.  Incident particle spectrum to detector.
!     ------------------------------------------------------------
      write(6,270)
270   FORMAT(/' Particle spectrum crossing the detector plane'/
     *       30X,'particles/MeV/source photon'/
     *       ' Upper energy',11X,'  Gamma',18X,' Electron',
     *       14X,' Positron')

      do nd=1,ndet
        do ie=1,50
```

```
          elow=deltae*(ie-1)
          eup=deltae*ie
          if (elow .gt. ekein ) go to 290

!     ---------------------------------
!     Gamma spectrum per MeV per source
!     ---------------------------------

          avspg = 0.D0
          desci2 = 0.D0
          do j = 1, nbatch
            avspg = avspg + spgpb(nd,ie,j)/nbatch
            desci2 = desci2 + spgpb(nd,ie,j)*spgpb(nd,ie,j)/nbatch
          end do
          sigspg = sqrt((desci2 - avspg*avspg)/(nbatch-1))
          avspg = avspg/deltae
          sigspg= sigspg/deltae

!     ------------------------------------
!     Electron spectrum per MeV per source
!     ------------------------------------

          avspe = 0.D0
          desci2 = 0.D0
          do j = 1, nbatch
            avspe = avspe + spepb(nd,ie,j)/nbatch
            desci2 = desci2 + spepb(nd,ie,j)*spepb(nd,ie,j)/nbatch
          end do
          sigspe = sqrt((desci2 - avspe*avspe)/(nbatch-1))
          avspe = avspe/deltae
          sigspe= sigspe/deltae

!     ------------------------------------
!     Positron spectrum per MeV per source
!     ------------------------------------

          avspp = 0.D0
          desci2 = 0.D0
          do j = 1, nbatch
            avspp = avspp + spppb(nd,ie,j)/nbatch
            desci2 = desci2 + spppb(nd,ie,j)*spppb(nd,ie,j)/nbatch
          end do
          sigspp = sqrt((desci2 - avspp*avspp)/(nbatch-1))
          avspp = avspp/deltae
          sigspp= sigspp/deltae

          write(6,280) eup,avspg,sigspg,avspe,sigspe,avspp,sigspp
280       FORMAT(G10.5,' MeV--',3(G12.5,'+-',G12.5))
        end do
      end do

290   continue
!     ============================
      call ecnsv1(1,nreg,totke)
      call ntally(1,nreg)
!     ============================

!     ====================
      call counters_out(1)
!     ====================

!     -----------
!     Close files
!     -----------
      close(UNIT=4)
      close(UNIT=6)

      stop

      end

!-------------------------last line of main code-------------------------

!------------------------------getcg.f---------------------------------
! Version:   040630-1300                                      KEK-LSCAT
! Reference: KEK Internal 2000-1
!-----------------------------------------------------------------------
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12
```

```
! ------------------------------------------------------------------------
! Auxiliary subroutine for use with the EGS5 Code System
! ------------------------------------------------------------------------
! This is a data-entry subprogram for use with a cg geometry.
! The data input is similar to that in ucrz.
! However, this version is designed specifically to utilize
! cg geometry.
! ------------------------------------------------------------------------
!
! -------------------
! SUBROUTINE ARGUMENT
! -------------------
!  nreg      Number of regions in geometry (determined by data input).
!
! ----------------
! UNIT ASSIGNMENTS
! ----------------
!  Unit 4    Input file.
!  Unit 6    Output file.
!  Unit 8    Echoes input cross-section data (assign a null file).
!  Unit 12   Input cross-section file from PEGS5.
!
! ----------
! INPUT FILE
! ----------
!  ============================================================
!  CG geometry related data must be written before following data.
!  ============================================================
!  --------
!  Record 1  title (80A1)        Title line.
!  --------
!  Record 2  nmed                Number of media in problem.
!  --------
!  Record 3  media(j,i) (24A1)   Media names (j=1,24, I=1,nmed lines).
!  --------
!  --------
!  Record 4  irlinl,irlinu,medtmp, rhotmp, ecutin, pcutin
!  -------- (3I5,3F10.3)         Set material for region from irlinl to ielinh.
!                                medtmp : material number
!                                rhotmp : If rhotmp=0.0, the default
!                                value for that medium is used.
!                                ecutin, pcutin : KINETIC energy cutoffs
!                                for electrons and photons, respectively,
!                                in MeV. If > 0, ecut(i) and pcut(i) are
!                                set. Otherwise ae and ap are used (default).
!                                irlinl =0 means end of define.
!
!                                If medtmp not 0, following data follows.
!  ---------
!  Record 4a ipeangsw,           Switches for PE-angle sampling,
!  --------- iedgesw,            K & L-edge fluorescence,
!            iraysw,             Rayleigh scattering,
!            ipolarsw,           Linearly-polarized photon scattering,
!            incohrsw,           S/Z rejection,
!            iprofrsw,           Doppler broadening,
!            impacrsw            electron impact ion-ization (0=off, 1=on).
!            (7I5)
!...+....1....+....2....+....3....+....4....+....5....+....6....+....7..
!  --------
!  Record 5  xin,yin,zin         Incident X,Y,Z coordinates (cm).
!  --------
!  Record 6  irin                Incident region.
!  ---------
!  Record 7  uin,vin,win         Incident direction cosines (U,V,W).
!  ---------                     If uin=vin=win=0, isotropic.
!  ---------
!  Record 8  ixx,jxx             Starting random number seeding.
!  ---------                     If ixx = 0, ixx is set to 123457.
!                                If jxx = 0, jxx is set to 654321.
!  ---------
!  Record 9  ncases              Number of cases.
!  ---------
!  Record 10 ekein,iqin,isamp    Kinetic energy (MeV), charge of inci-
!  ---------                     dent beam, and sampling switch.  If
!                                isamp=0, a monoenergetic beam (ekein)
!                                will be used.  Otherwise, a spectrum
!                                input must follow (Records 10a through
!                                10b), which will be sampled from discrete
!                                energy (isamp=1), directly (isamp=2) or
!                                uniformly over the energy range (isamp=3)
```

```
!                                with weighting factor.
!  ----------
!  Record 10a ebinmin            Only required when isamp>1(see above).
!                                Lowest energy (MeV) in spectrum.
!  ----------
!  Record 10b ebin(i),epdf(i)    Only required when usamp>0(see above).
!  ----------                    ebin(I) is the 'top-edge' of each
!                                energy bin (MeV) and epdf(i) is the
!                                corresponding probability for the bin.
!                                For example, a cross section (mb) can
!                                be used for epdf (but do not divide it
!                                by dE).  The last card is a delimiter
!                                and should be blank (or contain 0.0).
!                                The i-subscript runs from 1 to nebin
!                                (nebin calculated after the delimiter)
! ---------
! Record 11 iwatch               Switche for tracking events with swatch:
! ---------                        (0=No, 1=each interaction,
!                                   2=each step)
! ---------
! Record 12 ibrdst,iprdst,       Switches for bremsstrahlung and pair
! --------- ibrspl,nbrspl        production ANGLE SAMPLING, and brems-
!                                strahlung SPLITTING:
!                                ibrdst=0 No (use default: theta=m/E)
!                                       1 Yes (recommended)
!                                iprdst=0 No (use default: theta=m/E)
!                                       1 Yes (low-order distribution)
!                                       2 Yes (recommended)
!                                ibrspl=0 No
!                                       1 Yes (NBRSPL=splitting factor)
! --------
! Record 13 estepe,estepe2
! ---------
! ------------------------------------------------------------------------

      subroutine getcg(nreg)

      implicit none

      include 'include/egs5_h.f'                ! Main EGS "header" file

      include 'include/egs5_bounds.f'    ! COMMONs required by EGS5 code
      include 'include/egs5_brempr.f'
      include 'include/egs5_edge.f'
      include 'include/egs5_eiicom.f'
      include 'include/egs5_elecin.f'
      include 'include/egs5_media.f'
      include 'include/egs5_misc.f'
      include 'include/egs5_switches.f'
      include 'include/egs5_thresh.f'
      include 'include/egs5_useful.f'
      include 'include/egs5_userpr.f'
      include 'include/egs5_usersc.f'
      include 'include/egs5_uservr.f'
      include 'include/egs5_userxt.f'

      include 'pegscommons/mscom.f'                          ! PEGS common

      include 'user_auxcommons/aux_h.f'   ! Auxiliary-code "header" file

      include 'user_auxcommons/edata.f'
      include 'user_auxcommons/instuf.f'
      include 'user_auxcommons/watch.f'

      include 'include/randomm.f'         ! Additional (non-EGS5) COMMON

      integer nreg                                            ! Arguments

      real*8                                            ! Local variables
     * ecutmn,ek0,
     * ecutin,deg2rad,
     * pcutin,rhotmp,
     * therad,totphi

      integer i,iexp,ii,iiz,irlin,irlinl,irlinu,ixx,izn,j,
     *        jxx,k,n,medtmp,moreOutput,ner,nrbin,nzbin
```

```
      data deg2rad/0.01745329/
      data moreOutput/0/       ! Change this from 0 to 1 for more output

      write(6,1100)
1100  FORMAT(//,T25,'+----------------------------------------+',
     *        /,T25,'| EGS5 User Code using subroutine Getcg  |',
     *        /,T25,'+----------------------------------------+',
     *        /,T25,'|  NOTE:  cg geometry.                   |',
     *        /,T25,'+----------------------------------------+',
     *        //)

!  SJW 02-May-2002  New subroutine calls to initialize data no
!  longer set in block data because of size issues

!     ==============
      call block_set                ! Initialize some general variables
!     ==============

!     ================
      call region_init               ! Initialize some region variables
!     ================

! ---------------
! Record 1: title
! ---------------
      read(4,101) title
 101  FORMAT(80A1)
      write(6,102) title
 102  FORMAT(' TITLE:'/1X,80A1/)

! --------------
! Record 2: nmed
! --------------
      read(4,*) nmed
      if (nmed .gt. MXMED) then
        write(6,104) nmed
 104    FORMAT(' *** Stopped in Getcg with nmed=',I5,' > MXMED')
        stop
      end if
      write(6,105) nmed
 105  FORMAT(' nmed=',I5,/)

! ---------------
! Record 3: media
! ---------------
      do i=1,nmed
        read(4,106) (media(j,i),j=1,24)
 106    FORMAT(24A1)
        write(6,107) i,(media(j,i),j=1,24)
 107    FORMAT(' MEDIUM=',I5,' ==> ',24A1)
      end do

      do i=1,nreg                ! Set all regions to vacuum to begin with
        med(i) = 0
      end do

      write(6,108) ipeangsw,iedgesw,iraysw
108   FORMAT(//,' ipeangsw=',I5,
     *          ' Photoelectric-angle sampling (0=off, 1=on)',
     *         /,' iedgesw =',I5,
     *          ' K/L-edge switch (0=off, 1=on)',
     *         /,' iraysw  =',I5,
     *          ' Rayleigh scattering switch (0=off, 1=on)')

      write(6,109) ipolarsw,incohrsw,iprofrsw,impacrsw
109   FORMAT(//,' ipolarsw=',I5,
     *          ' Linearly polarized photon switch (0=off, 1=on)',
     *        /,' incohrsw=',I5,
     *          ' S/Z rejection switch (0=off, 1=on)',
     *        /,' iprofrsw=',I5,
     *          ' Doppler broadening switch (0=off, 1=on)',
     *        /,' impacrsw=',I5,
     *          ' Electron impact ionization switch (0=off, 1=on)')

      write(6,140) nreg-1
 140  FORMAT(/,' Assign medium and related flag for 1 to nreg-1 (='
     *  ,I5,')'/)
```

```
! ---------------------------------------------------------
!  Record 4  irlinl, irlinu, meptmp, rhotmp, ecutin, pcutin
!---------------------------------------------------------
!    ----------------------
!    Define to each region
!    ----------------------

142   continue
      read(4,143) irlinl,irlinu,medtmp,rhotmp,ecutin,pcutin
143   FORMAT(3I5,3F10.3)
      if (irlinl .eq. 0) go to 160

      if (medtmp.ne.0) then
! ----------------------------------------
! Record 4a: ipeangsw,iedgesw,iraysw,ipolarsw,
!            incohrsw,iprofrsw,impacrsw
! ----------------------------------------
        read(4,145) ipeangsw,iedgesw,iraysw,ipolarsw,incohrsw,
     * iprofrsw,impacrsw
145     FORMAT(7I5)

        write(6,146) irlinl,irlinu,medtmp,rhotmp,ecutin,pcutin
146     FORMAT(' Region from',I5,' to',I5,': medium =',I5,', rhoh=',
     *         G15.5/11X,' ecut =',G15.5,', pcut =',G15.5)

        write(6,150) ipeangsw,iedgesw,iraysw
150     FORMAT(11X,' iphter=',I3,3X,'iedgfl=',I3,3X,'iraylr=',I3)
        write(6,152) ipolarsw,incohrsw,iprofrsw,impacrsw
152     FORMAT(11X,' lpolar=',I3,3X,'incohr=',I3,3X,'iprofr=',I3,
     *       3X,'impacr=',I3)
      else
        write(6,153) irlin
153     FORMAT(' Region =',I5,' is vacuum')
      end if

      do irlin=irlinl,irlinu
        med(irlin)=medtmp
        if (medtmp.ne.0) then
          if(rhotmp. gt. 0.) then
            rhor(irlin) = rhotmp
          end if
          if (ecutin .gt. 0.) then
            ecut(irlin) = pcutin
          end if
          if (pcutin .gt. 0.) then
            pcut(irlin) = pcutin
          end if
          iphter(irlin) = ipeangsw
          iedgfl(irlin) = iedgesw
          iraylr(irlin) = iraysw
          lpolar(irlin) = ipolarsw
          incohr(irlin) = incohrsw
          iprofr(irlin) = iprofrsw
          impacr(irlin) = impacrsw
        end if
      end do
      go to 142

160   continue
! ---------------------
! Record 5: xin,yin,zin
! ---------------------
      read(4,*) xin,yin,zin

      write(6,180) xin,yin,zin
180   FORMAT(/,' xin=',G15.7,5X,'yin=',G15.7,5X,'zin=',G15.7
     *         /' (incident coordinates)')

! ---------------
! Record 5: irin
! ---------------
      read(4,*) irin
      write(6,190) irin
190   FORMAT(/,' irin=',I5,' (incident region)')

! ----------------------
! Record 6: uin,vin,win
! ----------------------
```

```
      read(4,*) uin,vin,win
      write(6,200) uin,vin,win
200   FORMAT(/,' uin=',G15.7,5X,'vin=',G15.7,5X,'win=',G15.7,
     *       ' (incident direction cosines)')

! SJW 02-May-2002  Not needed for EGS5
! ------------------
! Record 7: ixx,jxx
! ------------------
      read(4,*) ixx,jxx
      if (ixx .eq. 0)  ixx = 123457               ! Default seed
      if (jxx .eq. 0)  jxx = 654321               ! Default seed
      write(6,210) ixx,jxx
210   FORMAT(/,' ixx=',I12,5X,'jxx=',I12,
     *       ' (starting random-number seeds)')

!     -------------------------------------
!     Save the starting random-number seeds
!     -------------------------------------
       iseed1=ixx
       iseed2=jxx

!     ===========
!      call rmarin               ! Initialize the random-number generator
!     ===========

! -----------------
! Record 8: ncases
! -----------------
      read(4,*) ncases
      write(6,220) ncases
220   FORMAT(/,' ncases=',I12)

! --------------------------
! Record 9: ekein,iqin,isamp
! --------------------------
      read(4,*) ekein,iqin,isamp
                                                  ! ------------------
      if (isamp .eq. 0) then                      ! Monoenergetic case
                                                  ! ------------------
        write(6,230) iqin,ekein
230     FORMAT(/,' MONOENERGETIC case has been selected with:',
     *        //,' iqin=',I5,' (incident charge of beam)',
     *         /,' ekein=',G15.5,' MeV (incident kinetic energy)')

                                                  ! --------------------
      else if (isamp .gt. 0) then                 ! Energy spectrum case
                                                  ! --------------------
!       -------------------
!       Record 9a: ebinmin
!       -------------------
        if(isamp.ne.1) then
          read(4,*) ebinmin            ! Lowest energy in spectrum (MeV)
          write(6,240) iqin,ebinmin
240       FORMAT(/,' Energy-SPECTRUM case has been selected with:',
     *          //,' iqin=',I5,' (incident charge of beam)',
     *          /,' ebinmin=',F10.3,' MeV (lowest energy bin)')
        end if

        if (isamp .eq. 1) then
          write(6,245) isamp
245       FORMAT('   isamp =',I2,' (Sample from discrete energy)')
        elseif (isamp .eq. 2) then
          write(6,250) isamp
250       FORMAT('   isamp =',I2,' (DIRECT-sampling over energy range)')
        else if (isamp .eq. 3) then
          write(6,260) isamp
260       FORMAT('   isamp =',I2,
     *           ' (UNIFORM-sampling over energy range) with WEIGHTING')
        end if

!       ---------------------------
!       Record 9b: ebin(i),epdf(i)
!       ---------------------------
        i = 0
                                  ! ---------------------------------
265     continue                  ! Start of energy-spectrum input loop
                                  ! ---------------------------------
```

```
          i = i + 1
          if (i .gt. MXEBIN) then
            write(6,270) i
270         FORMAT(//,' Stopped in getcg with I=',I6,' > MXEBIN')
            stop
          end if
          read(4,*) ebin(i),epdf(i)       ! ebin(i) is top-edge of bin
          if (i .gt. 1 .and. ebin(i) .le. ebin(i-1)) then
            go to 285
          else if (i. eq. 1 .and. ebin(i) .le. ebinmin) then
            go to 275
          end if
        go to 265

275     continue                  ! Reach here when a read-error occurs
        write(6,280)
280     FORMAT(//,' Stopped in getcg with spectrum read-error')
        stop

285     continue          ! Reach here when delimiter card has been read

        nebin = i - 1                     ! Number of energy bins read in
        totphi = 0.
        do i=1,nebin
          totphi = totphi + epdf(i)
        end do
        ecdf(1) = epdf(1)/totphi
        do i=2,nebin
          ecdf(i) = ecdf(i-1) + epdf(i)/totphi
        end do

        write(6,290) (i,ebin(i),epdf(i),ecdf(i),i=1,nebin)
290     FORMAT(/,'  BIN    UPPER ENERGY   PROBABILITY     CUMULATIVE ',
     *         /,'   #        (MeV)                      PROBABILITY',
     *         /,(I4,3X,F10.3,2F16.4))

!       -------------------------------
!       Set up energy-sampling interval
!       -------------------------------
        esam1 = ebinmin
        esam2 = ebin(nebin)
        delsam = esam2 - esam1

        write(6,300) esam1,esam2
300     FORMAT(//,' Energy-sampling interval is:',/,
     *           '   esam1 =',G15.5,' MeV to esam2 =',G15.5,' MeV',/)
       else
         write(6,310) isamp
310      FORMAT(/,' Stopped in getcg with bad isamp=',I10)
         stop
       end if

! -----------------
! Record 10: iwatch
! -----------------
       read(4,*) iwatch
       write(6,350) iwatch
350    FORMAT(//,' SWATCH tracking switch: iwatch=',I2,
     *          ' (0=off, 1=each interaction, 2=each step)')

! --------------------------------------
! Record 11: ibrdst,iprdst,ibrspl,nbrspl
! --------------------------------------
       read(4,*) ibrdst,iprdst,ibrspl,nbrspl
       write(6,410) ibrdst,iprdst,ibrspl,nbrspl
410    FORMAT(/,' IBRDST=',I2,/,' IPRDST=',I2,/,' IBRSPL=',I2,' (NBRSPL='
     *,I5,')')

       if (ibrspl .gt. 0) then
         if (nbrspl .gt. 0) then
           fbrspl = 1.0/float(nbrspl)
         else
           write(6,420) ibrspl,nbrspl
420        FORMAT(//,' Stopped in Getcg with IBRSPL=',I5,' and NBRSPL=',
     *     I5)
           stop
         end if
       end if
```

```
!     ------------------------------------------
!     Run KEK version of PEGS5 before calling HATCH
!     (method was developed by Y. Namito - 010306)
!     ------------------------------------------
      write(6,430)
430   FORMAT(/,' PEGS5NB3-call comes next',/)

!     =============
      call pegs5nb3
!     =============

!     ------------------------------
!     Open files (before HATCH call)
!     ------------------------------
      open(UNIT=KMPI,FILE='pgs5job.pegs5dat',STATUS='old')
      open(UNIT=KMPO,FILE='egs5job.dummy',STATUS='unknown')

      write(6,440)
440   FORMAT(/,' HATCH-call comes next',/)

!     ==========
      call hatch
!     ==========

!     ------------------------------
!     Close files (after HATCH call)
!     ------------------------------
      CLOSE(UNIT=KMPI)
      CLOSE(UNIT=KMPO)

! SJW 02-May-2002  replace reading of PRESTA switches with
! estepe and estepe2, and call to presta_inputs with calls
! to check_limits and rmsfit

! Set minimum (total) energy
      ecutmn = 1.D10
      do i = 1,nreg
        if (ecut(i).gt.0.0) ecutmn=min(ecutmn,ecut(i))
      end do

      ek0 = ekein                          ! Set maximum (kinetic) energy

!     ===================================
!      call presta_inputs(nreg,ecutmn,ek0)    ! Do PRESTA inputs/summary
!     ===================================

! -------------------------
! Record 12: estepe,estepe2
! -------------------------
      read(4,*) estepe, estepe2
      write(6,450) estepe, estepe2
450   FORMAT(/,1X,' ESTEPE at EKMAX: ',F10.0,' (estepe)',
     *       /,1X,' ESTEPE at ECUT:  ',F10.0,' (estepe2)')

! ---------------------------------------
! Print values used for efracl and efrach
! ---------------------------------------
      write(6,*)
      write(6,*) '  EFRACL=',efracl
      write(6,*) '  EFRACH=',efrach

!     ===================================
      call check_limits(nreg,ecutmn,ek0)     ! Set energy step constants
!     ===================================

!     ===========
      call rmsfit                         ! read multiple scattering data
!     ===========

!-------------------------------------------------------------
! All of the input data should have been read in at this point,
! but check to make sure that the incident kinetic energy is
! below the limit set by PEGS (i.e., UE and UP) for all media.
!-------------------------------------------------------------
      do j=1,nmed
        if (ekein+RM .gt. ue(j)) then
          write(6,*)
     *      'Stopped in SUBROUTINE getcg with ekein + RM > ue(j):'
          write(6,*) '   j = ',j
          write(6,*) '   ekein + RM = ',ekein+RM
```

```
          write(6,*) '   ue(j) = ',ue(j)
          stop
        end if
        if (ekein .gt. up(j)) then
          write(6,*)
     *      'Stopped in SUBROUTINE getcg with ekein > up(j):'
          write(6,*) '   j = ',j
          write(6,*) '   ekein = ',ekein
          write(6,*) '   up(j) = ',up(j)
          stop
        end if
      end do

! ---------------------------------------------------------
! Print various data associated with each media (not region)
! ---------------------------------------------------------
      write(6,460)
460   FORMAT(/,' Quantities associated with each MEDIA:')
      do j=1,nmed
        write(6,470) (media(i,j),i=1,24)
470     FORMAT(/,1X,24A1)
        write(6,480) rho(j),rlc(j)
480     FORMAT(5X,' rho=',G15.7,' g/cu.cm     rlc=',G15.7,' cm')
        write(6,490) ae(j),ue(j)
490     FORMAT(5X,' ae=',G15.7,' MeV    ue=',G15.7,' MeV')
        write(6,500) ap(j),up(j)
500     FORMAT(5X,' ap=',G15.7,' MeV    up=',G15.7,' MeV',/)
      end do

! ---------------------------------------------------
! Print media and cutoff energies assigned to each region
! ---------------------------------------------------
      if(moreOutput .eq.1) then
        do i=1,nreg
          if (med(i) .eq. 0) then
            write(6,510) i,ecut(i),pcut(i)
510         FORMAT(' medium(',I3,')=vacuum',18X,
     *               'ecut=',G10.5,' MeV, pcut=',g10.5,' mev')
          else
            write(6,520) i,(media(ii,med(i)),ii=1,24),ecut(i),pcut(i)
520         FORMAT(' medium(',I3,')=',24A1,
     *               'ecut=',G10.5,' MeV, pcut=',G10.5,' MeV')
!           ---------------------------------------------
!           Print out energy information of K- and L-X-rays
!           ---------------------------------------------
            if (iedgfl(i) .ne. 0) then             ! Output X-ray energy
              ner = nne(med(i))
              do iiz=1,ner
                izn = zelem(med(i),iiz)  ! Atomic number of this element
                write(6,530) izn
530             FORMAT('   X-ray information for Z=',I3)
                write(6,540) (ekx(ii,izn),ii=1,10)
540             FORMAT('   K-X-ray energy in keV',/,
     *                 4G15.5,/,4G15.5,/,2G15.5)
                write(6,550) (elx1(ii,izn),ii=1,8)
550             FORMAT('   L-1 X-ray in keV',/,4G15.5,/,4G15.5)
                write(6,560) (elx2(ii,izn),ii=1,5)
560             FORMAT('   L-2 X-ray in keV',/,5G15.5)
                write(6,570) (elx3(ii,izn),ii=1,7)
570             FORMAT('   L-3 X-ray in keV',/,4G15.5,/,3G15.5)
              end do
            end if
          end if
        end do
      end if
                                                          ! --------------
      return                                              ! Return to MAIN
                                                          ! --------------
      end

!-------------------------last line of getcg.f-----------------------

!------------------------------ausgab.f--------------------------------
! Version:   030831-1300
! Reference: SLAC-265 (p.19-20, Appendix 2)
!----------------------------------------------------------------------
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12
```

```
! ------------------------------------------------------------------------
! Required subroutine for use with the EGS5 Code System
! ------------------------------------------------------------------------
! A simple AUSGAB to:
!
!   1) Score energy deposition
!   2) Print out stack information
!   3) Print out particle transport information (if switch is turned on)
!

! ------------------------------------------------------------------------

      subroutine ausgab(iarg)

      implicit none

      include 'include/egs5_h.f'                ! Main EGS "header" file

      include 'include/egs5_epcont.f'    ! COMMONs required by EGS5 code
      include 'include/egs5_misc.f'
      include 'include/egs5_stack.f'
      include 'include/egs5_useful.f'

      include 'user_auxcommons/aux_h.f'   ! Auxiliary-code "header" file

      include 'user_auxcommons/etaly1.f'         ! Auxiliary-code COMMONs
      include 'user_auxcommons/lines.f'
      include 'user_auxcommons/ntaly1.f'
      include 'user_auxcommons/watch.f'

      include 'auxcommons/etaly2.f'      !  Added SJW for energy balance

      common/totals/                            ! Variables to score
     * depe,deltae,spg(1,50),spe(1,50),spp(1,50),nreg
      real*8 depe,deltae,spg,spe,spp
      integer nreg

      integer                                                 ! Arguments
     * iarg

      real*8                                             ! Local variables
     * edepwt

      integer
     * ie,iql,irl

!     ------------------------
!     Set some local variables
!     ------------------------
      irl = ir(np)
      iql = iq(np)
      edepwt = edep*wt(np)

!     ------------------------------------------------------------
!     Keep track of energy deposition (for conservation purposes)
!     ------------------------------------------------------------
      if (iarg .lt. 5) then
        esum(iql+2,irl,iarg+1) = esum(iql+2,irl,iarg+1) + edepwt
        nsum(iql+2,irl,iarg+1) = nsum(iql+2,irl,iarg+1) + 1


!  added SJW for particle by particle energy balance
        if(irl.eq.nreg) then
          eparte = eparte + edepwt
        else
          epartd = epartd + edepwt
        endif
      end if

!     ---------------------------------------------
!     Score energy deposition inside NaI detector
!     ---------------------------------------------
      if (med(irl). eq. 1) then
        depe = depe + edepwt

!      ---------------------------------------------------
!      Score particle information if it enters from outside
!      ---------------------------------------------------
       if (irl .ne. irold .and. iarg .eq. 0) then
         if (iql .eq. 0) then             ! photon
```

```
            ie = e(np)/deltae +1
            if(ie .gt. 50) ie = 50
            spg(1,ie) = spg(1,ie) + wt(np)
          elseif (iql .eq. -1) then        ! electron
            ie = (e(np) - RM)/deltae +1
            if(ie .gt. 50) ie = 50
            spe(1,ie) = spe(1,ie) + wt(np)
          else                             ! positron
            ie = (e(np) - RM)/deltae +1
            if(ie .gt. 50) ie = 50
            spp(1,ie) = spp(1,ie) + wt(np)
          end if
        end if
      end if


!     ----------------------------------------------------------------
!     Print out stack information (for limited number cases and lines)
!     ----------------------------------------------------------------
      if (ncount .le. nwrite .and. ilines .le. nlines) then
        ilines = ilines + 1
        write(6,101) e(np),x(np),y(np),z(np),u(np),v(np),w(np),
     *               iql,irl,iarg
 101    FORMAT(4G15.7/3G15.7,3I5)
      end if

!     ----------------------------------------------------------------
!     Print out particle transport information (if switch is turned on)
!     ----------------------------------------------------------------
!                          ========================
      if (iwatch .gt. 0) call swatch(iarg,iwatch)
!                          ========================

      return

      end

!-------------------------last line of ausgab.f-------------------------
!-----------------------------howfar.f-----------------------------------
! Version:   040727-1300
! Reference: T. Torii and T. Sugita, "Development of PRESTA-CG
! Incorprating Combinatorial Geometry in EGS4/PRESTA", JNC TN1410 2002-201,
! Japan Nuclear Cycle Development Institute (2002).
! Improved version is provided by T. Sugita. 7/27/2004
!--------------------------------------------------------------------------
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12

! ------------------------------------------------------------------------
! Required (geometry) subroutine for use with the EGS5 Code System
! ------------------------------------------------------------------------
! This is a CG-HOWFAR.
! ------------------------------------------------------------------------

      subroutine howfar

      implicit none

      include 'include/egs5_h.f'
      include 'include/egs5_epcont.f'
      include 'include/egs5_stack.f'
      include 'include/egs5_thresh.f'

!     include 'user_auxcommons/aux_h.f'
      include 'user_auxcommons/cg/tvalcg.f'
      include 'user_auxcommons/cg/zondta.f'
      include 'user_auxcommons/cg/rppdta.f'
      include 'user_auxcommons/cg/sphdtac.f'
      include 'user_auxcommons/cg/rccdta.f'
      include 'user_auxcommons/cg/trcdta.f'
      include 'user_auxcommons/cg/tordta.f'

      real*8 atvaltmp,xidd,yidd,zidd                    ! Local variables

      real delhow,tval,tval0,tval10,tval00,tvalmn,udotau,udotav,
     *     udotaw,xiss,xl,yiss,yl,ziss,zl

      integer i,ihitcg,irl,irlfg,irlold,irnear,irnext,itvlfg,j,jjj
```

```
      IRL=IR(NP)
      IF (IRL.LT.1.OR.IRL.GE.IZONIN) THEN
        IDISC=1
        RETURN
      END IF
      TVAL=1.E+30
      ITVALM=0
      DO I=1,NBBODY(IRL)
        DO J=1,IRPPIN
          IF (ABS(NBZONE(I,IRL)).EQ.NBRPP(J)) THEN
            UDOTAU=U(NP)
            UDOTAV=V(NP)
            UDOTAW=W(NP)
            XL=X(NP)
            YL=Y(NP)
            ZL=Z(NP)
            CALL RPPCG1(J,XL,YL,ZL,UDOTAU,UDOTAV,UDOTAW)
          END IF
        end do
        DO J=1,ISPHIN
          IF (ABS(NBZONE(I,IRL)).EQ.NBSPH(J)) THEN
            UDOTAU=U(NP)
            UDOTAV=V(NP)
            UDOTAW=W(NP)
            XL=X(NP)
            YL=Y(NP)
            ZL=Z(NP)
            CALL SPHCG1(J,XL,YL,ZL,UDOTAU,UDOTAV,UDOTAW)
          END IF
        end do
        DO J=1,IRCCIN
          IF (ABS(NBZONE(I,IRL)).EQ.NBRCC(J)) THEN
            UDOTAU=U(NP)
            UDOTAV=V(NP)
            UDOTAW=W(NP)
            XL=X(NP)
            YL=Y(NP)
            ZL=Z(NP)
            CALL RCCCG1(J,XL,YL,ZL,UDOTAU,UDOTAV,UDOTAW)
          END IF
        end do
        DO J=1,ITRCIN
          IF (ABS(NBZONE(I,IRL)).EQ.NBTRC(J)) THEN
            UDOTAU=U(NP)
            UDOTAV=V(NP)
            UDOTAW=W(NP)
            XL=X(NP)
            YL=Y(NP)
            ZL=Z(NP)
            CALL TRCCG1(J,XL,YL,ZL,UDOTAU,UDOTAV,UDOTAW)
          END IF
        end do
        DO J=1,ITORIN
          IF (ABS(NBZONE(I,IRL)).EQ.NBTOR(J)) THEN
            UDOTAU=U(NP)
            UDOTAV=V(NP)
            UDOTAW=W(NP)
            XL=X(NP)
            YL=Y(NP)
            ZL=Z(NP)
            CALL TORCG1(J,XL,YL,ZL,UDOTAU,UDOTAV,UDOTAW)
          END IF
        end do
      end do
      IRNEAR=IRL
      IF (ITVALM.EQ.0) THEN
        TVAL0=1.E-4
        XISS=X(NP)+TVAL0*U(NP)
        YISS=Y(NP)+TVAL0*V(NP)
        ZISS=Z(NP)+TVAL0*W(NP)
2291    IF(X(NP).NE.XISS.OR.Y(NP).NE.YISS.OR.Z(NP).NE.ZISS) GO TO 2292
          TVAL0=TVAL0*10.
          XISS=X(NP)+TVAL0*U(NP)
          YISS=Y(NP)+TVAL0*V(NP)
          ZISS=Z(NP)+TVAL0*W(NP)
        GO TO 2291
2292    CONTINUE
        XIDD=DBLE(X(NP))+DBLE(TVAL0)*DBLE(U(NP))
        YIDD=DBLE(Y(NP))+DBLE(TVAL0)*DBLE(V(NP))
        ZIDD=DBLE(Z(NP))+DBLE(TVAL0)*DBLE(W(NP))
        CALL SRZONE(XIDD,YIDD,ZIDD,IRNEXT)
        IF (IRNEXT.NE.IRL) THEN
```

```
          TVAL=0.0
          IRNEAR=IRNEXT
        ELSE
          TVAL00=0.0
          TVAL10=10.0*TVAL0
          IRLOLD=IRL
          IRLFG=0
2301      IF (IRLFG.EQ.1) GO TO 2302
            TVAL00=TVAL00+TVAL10
            IF (TVAL00.GT.1.0E+06) THEN
              WRITE(6,2310)IQ(NP),IR(NP),X(NP),Y(NP),Z(NP), U(NP),V(NP),
     *        W(NP),TVAL00
2310          FORMAT(' TVAL00 ERROR : IQ,IR,X,Y,Z,U,V,W,TVAL=', 2I3,
     *        1P7E12.5)
              STOP
            END IF
            XIDD=DBLE(X(NP))+DBLE(TVAL00)*DBLE(U(NP))
            YIDD=DBLE(Y(NP))+DBLE(TVAL00)*DBLE(V(NP))
            ZIDD=DBLE(Z(NP))+DBLE(TVAL00)*DBLE(W(NP))
            CALL SRZOLD(XIDD,YIDD,ZIDD,IRLOLD,IRLFG)
          GO TO 2301
2302      CONTINUE
          TVAL=TVAL00
          DO J=1,10
            XIDD=DBLE(X(NP))+DBLE(TVAL00)*DBLE(U(NP))
            YIDD=DBLE(Y(NP))+DBLE(TVAL00)*DBLE(V(NP))
            ZIDD=DBLE(Z(NP))+DBLE(TVAL00)*DBLE(W(NP))
            CALL SRZONE(XIDD,YIDD,ZIDD,IRNEXT)
            IF (IRNEXT.NE.IRLOLD) THEN
              TVAL=TVAL00
              IRNEAR=IRNEXT
            END IF
            TVAL00=TVAL00-TVAL0
          end do
          IF (IRL.EQ.IRNEAR) THEN
            WRITE(0,*) 'IRL,TVAL=',IRL,TVAL
          END IF
        END IF
      ELSE
        DO J=1,ITVALM-1
          DO I=J+1,ITVALM
            IF ((ATVAL(I).LT.ATVAL(J))) THEN
              ATVALTMP=ATVAL(I)
              ATVAL(I)=ATVAL(J)
              ATVAL(J)=ATVALTMP
            END IF
          end do
        end do
        ITVLFG=0
        TVALMN=TVAL
        DO JJJ=1,ITVALM
          IF (TVALMN.GT.ATVAL(JJJ)) THEN
            TVALMN=ATVAL(JJJ)
          END IF
          DELHOW=1.E-4
          TVAL0=ATVAL(JJJ)+DELHOW
          XISS=X(NP)+TVAL0*U(NP)
          YISS=Y(NP)+TVAL0*V(NP)
          ZISS=Z(NP)+TVAL0*W(NP)
2361      IF(X(NP).NE.XISS.OR.Y(NP).NE.YISS.OR.Z(NP).NE.ZISS) GO TO 2362
            DELHOW=DELHOW*10.
            TVAL0=ATVAL(JJJ)+DELHOW
            XISS=X(NP)+TVAL0*U(NP)
            YISS=Y(NP)+TVAL0*V(NP)
            ZISS=Z(NP)+TVAL0*W(NP)
          GO TO 2361
2362      CONTINUE
          XIDD=DBLE(X(NP))+DBLE(TVAL0)*DBLE(U(NP))
          YIDD=DBLE(Y(NP))+DBLE(TVAL0)*DBLE(V(NP))
          ZIDD=DBLE(Z(NP))+DBLE(TVAL0)*DBLE(W(NP))
          CALL SRZONE(XIDD,YIDD,ZIDD,IRNEXT)
          IF ((IRNEXT.NE.IRL.OR.ATVAL(JJJ).GE.1.).AND.TVAL.GT.
     *    ATVAL(JJJ)) THEN
            TVAL=ATVAL(JJJ)
            IRNEAR=IRNEXT
            ITVLFG=1
            GOTO 2370
          END IF
        end do
2370    IF (ITVLFG.EQ.0) THEN
          TVAL0=1.E-4
          XISS=X(NP)+TVAL0*U(NP)
```

```
          YISS=Y(NP)+TVAL0*V(NP)
          ZISS=Z(NP)+TVAL0*W(NP)
2381      IF(X(NP).NE.XISS.OR.Y(NP).NE.YISS.OR.Z(NP).NE.ZISS) GO TO 2382
            TVAL0=TVAL0*10.
            XISS=X(NP)+TVAL0*U(NP)
            YISS=Y(NP)+TVAL0*V(NP)
            ZISS=Z(NP)+TVAL0*W(NP)
          GO TO 2381
2382      CONTINUE
          IF (TVALMN.GT.TVAL0) THEN
            TVAL=TVALMN
          ELSE
            TVAL=TVAL0
          END IF
        END IF
      END IF
      IHITCG=0
      IF (TVAL.LE.USTEP) THEN
        USTEP=TVAL
        IHITCG=1
      END IF
      IF (IHITCG.EQ.1) THEN
        IF (IRNEAR.EQ.0) THEN
          WRITE(6,2390)IQ(NP),IR(NP),X(NP),Y(NP),Z(NP),U(NP),V(NP),W(NP)
     *     ,TVAL
2390      FORMAT(' TVAL ERROR : IQ,IR,X,Y,Z,U,V,W,TVAL=',2I3,1P7E12.5)
          IDISC=1
          ITVERR=ITVERR+1
          IF (ITVERR.GE.100) THEN
            STOP
          END IF
          RETURN
        END IF
        IRNEW=IRNEAR
      END IF
      RETURN
      END

!--------------------last line of subroutine howfar---------------------
```