EGS5 sample user code (ucnaicgv.f)
Response calculation of NaI detector
(August 12 2013, Draft)

Hideo Hirayama and Yoshihito Namito

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

Contents

1. Combinatorial Geometry (CG) 1
- 1.1. Body Definition . . . 1
- 1.2. Region Definition . . . 1
- 1.3. Example of Region Description . . . 2

2. Outlines of sample user code uccg_phantom.f 4
- 2.1. CG input data . . . 4

3. Details of user code 4
- 3.1. Main program: Step 1 . . . 4
  - 3.1.1. Include lines and specification statements: . . . 4
  - 3.1.2. `open` statement: . . . 6
- 3.2. Step 2: Pegs5-call . . . 6
- 3.3. Step 3: Pre-hatch-call-initialization . . . 7
- 3.4. Step 4: Determination-of-incident-particle-parameters . . . 8
- 3.5. Step 5: hatch-call . . . 9
- 3.6. Step 6: Initialization-for-howfar . . . 9
- 3.7. Step 7: Initialization-for-ausgab . . . 9
- 3.8. Step 8: Shower-call . . . 9
  - 3.8.1. Statistical uncertainty: . . . 11
  - 3.8.2. Output of results: . . . 11
- 3.9. `Subroutine ausgab` . . . 12
- 3.10. `Subroutine howfar` . . . 13

4. Comparison of speed between ucnai.f and ucnaicgv.f 14

5. Exercise problems 14
- 5.1. Problem 1 : Calculation for NaI detector . . . 14
- 5.2. Problem 2 : Ge detector calculation . . . 14
- 5.3. Problem 3 : Air ionization chamber calculation . . . 14

6. Answer for exercises 15
- 6.1. Problem 1 . . . 15
- 6.2. Problem 2 . . . 18
- 6.3. Problem 3 . . . 19

1. Combinatorial Geometry (CG)

1.1. Body Definition

Following bodies are supported in CG for EGS [1] .

1. Rectangular Parallelepiped (RPP)
   Specify the maximum and minimum values of x-, y-, and z-coordinates that bound a rectangular parallelepiped 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 $\theta 2$=2π, respectively.

Table 1: Data required to described each body type.

Body Type	Number	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			

1.2. 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. Here, region and zone are used as the same meaning. A special operator notations involving the symbols (+), (−), and (OR) is used to describe the intersections 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.

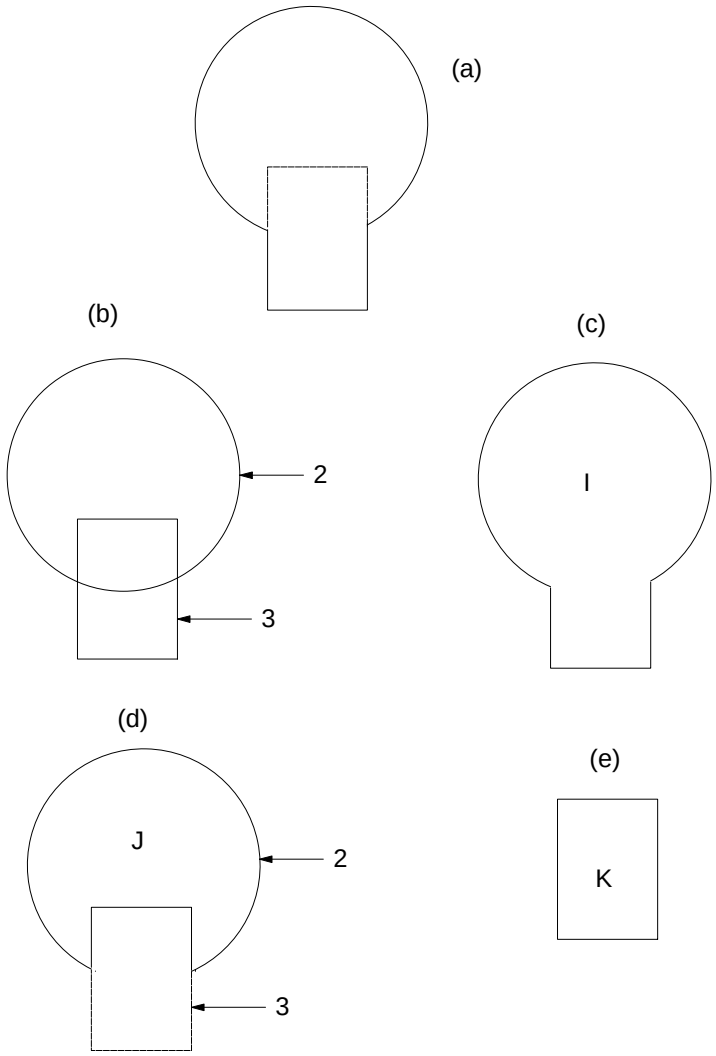

Figure 1: Examples of Combinatorial Geometry Method.

1.3. 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 shaded region A and the unshaded 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.

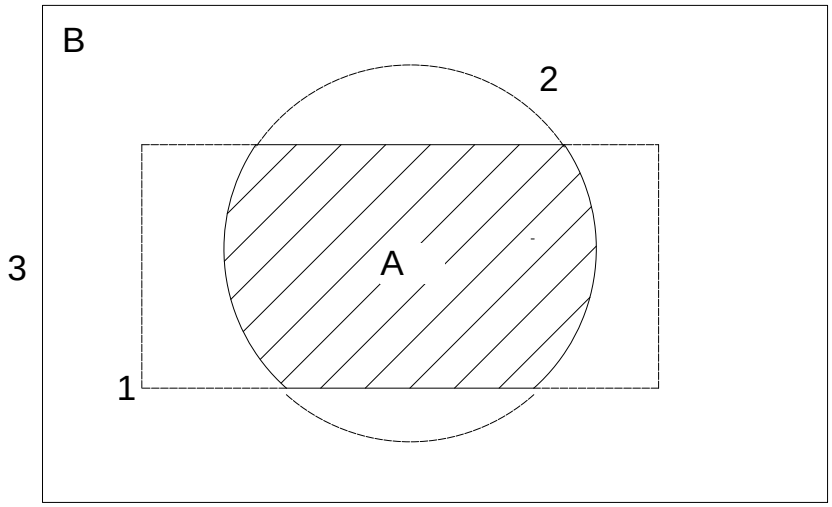

Figure 2: Use of OR operator.

2. Outlines of sample user code uccg_phantom.f

`ucnaicgv.f` is the egs5 user code to calculate the NaI detector response using CG. Input data of CG are written on the input data read from unit 4.

2.1. CG input data

Geometry are defined by the combination of several cylinders as shown 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
                   7.62         3.81
RCC     2          0.00          0.0         -0.5          0.00          0.0
                   8.12         4.31
RCC     3          0.00          0.0         -0.6          0.00          0.0
                   8.72         4.41
RCC     4          0.00          0.0         7.62          0.00          0.0
                    0.5         4.31
RCC     5          0.00          0.0         -5.6          0.00          0.0
                  18.72         9.41
RCC     6           0.0          0.0        -10.0           0.0          0.0
                   30.0         12.0
END
 Z1        +1
 Z2        +2       -1
 Z3        +3       -2       -4
 Z4        +4
 Z5        +5       -3
 Z6        +6       -5
END
 1    0    2    3    4    0
```

1. Source conditions
   - Source photon energy is 1.253 MeV.
   - Pencil beam normally along Z-axis to a point of (0.0, 0.0, -5.0).
2. Results obtained
   (a) Particle trajectory data (`egs5job.pic`)
   (b) Calculated result (`egs5job.out`)
      - Material data used in the calculation.
      - Information set to each region.
      - Peak and total efficiency of the detector and their uncertainties.
      - Detector response.
      - Particle spectra entering to NaI detector region.

3. Details of user code

3.1. Main program: Step 1

3.1.1. Include lines and specification statements: egs5 is written in Fortran 77. The size of arguments is defined in other files and included by using 'include line'. Various commons used inside egs5 are also included by the same way.

Include files related with egs5 are put on the `include` directory and those related with pegs5 are put on the `pegscommons` directory. Those for each user code including geometry related are put on the `auxcommons` directory. These files are linked by running egs5run script.

This is the most different feature with EGS4 at which the size of arguments can be modified inside an user code with Mortran macro. If it is necessary to modify the size of arguments used in

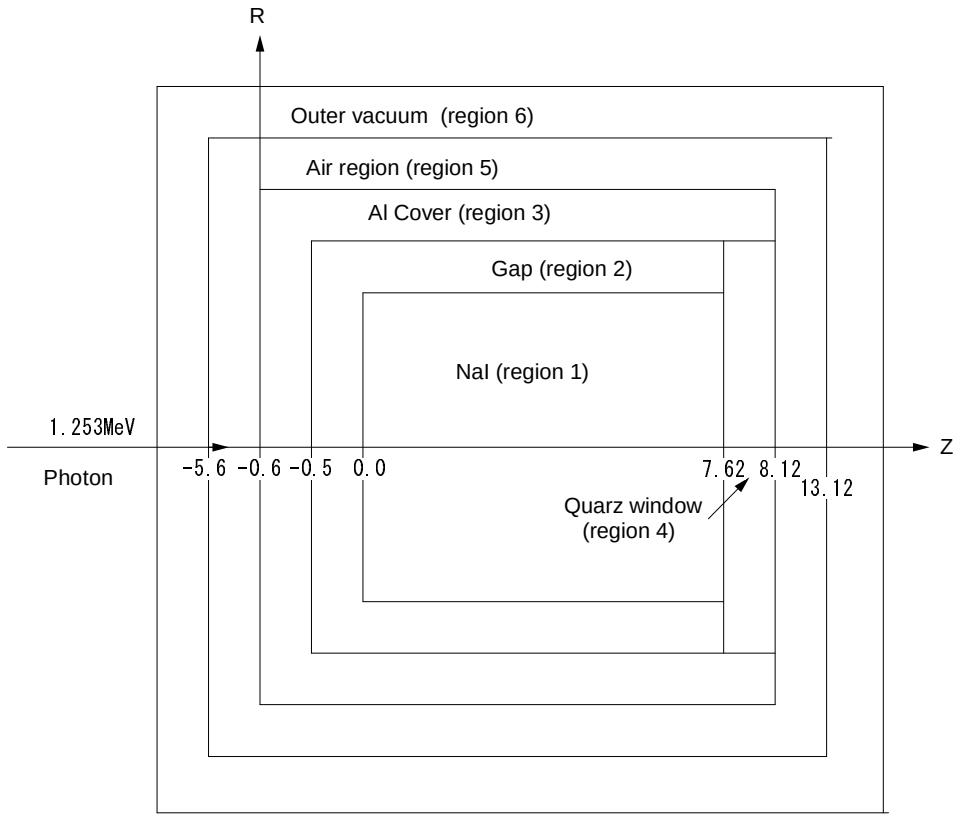

Figure 3: Geometry of ucnaicgv.f

egs5, you must modify the related parameter in 'egs5/include/egs5_h.f'. The parameters related to each user code are defined in 'egs5/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_bounds.f'
      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 included in each common are used inside the main program.*

Next is include lines not directly related to egs5 like geometry related.

```
!     ----------------------
!     Auxiliary-code COMMONs
!     ----------------------
```

*This is corresponding to COMIN macros in EGS4.

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

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

!     ------------------
!     cg related COMMONs
!     ------------------
      include 'auxcommons/geom_common.f' ! geom-common file
      integer irinn
```

The last `include` statement is related to CG.

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

```
      common/totals/                          ! Variables to score
     * depe,deltae,spec(3,50),maxpict
      real*8 depe,deltae,spec
      integer maxpict
```

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

3.1.2. `open` statement: At the top of executable statement, it is necessary to open files used in the user code. Due to the new feature that pegs is called inside each user code, user 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. The unit used in the subroutine 'plotxyz' and 'geomout' used to keep and output trajectory information is changed from '9' to '39' for this reason.

```
!     ----------
!     Open files
!     ----------
      open(6,FILE='egs5job.out',STATUS='unknown')
      open(4,FILE='egs5job.inp',STATUS='old')
      open(39,FILE='egs5job.pic',STATUS='unknown')
```

`counters_out` is the subroutine to set various counters to 0.

3.2. Step 2: Pegs5-call

Define the number of materials used in the user code as `nmed`.

Material names used in egs are defined after initialize some general variables by calling `subroutine block_set`. The material name defined here must be included in the material produced by pegs5 using input data read from unit 25. Data of 24 characters should be supplied for each line of `medarr`.

Characteristic dimension which is related to the minimum region size like diameter, length or thickness for each material is set as `chard`.

`Subroutine pegs5` is called after above setting.

```
      nmed=4
      if(nmed.gt.MXMED) then
        write(6,'(A,I4,A,I4,A/A)')
     *     ' nmed (',nmed,') larger than MXMED (',MXMED,')',
     *     ' MXMED in iclude/egs5_h.f must be increased.'
        stop
      end if

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

```
!     --------------------------------
!     define media before calling PEGS5
!     --------------------------------

      medarr(1)='NAI                       '
      medarr(2)='AL                        '
      medarr(3)='QUARTZ                    '
      medarr(4)='AIR-AT-NTP                '

      do j=1,nmed
        do i=1,24
          media(i,j)=medarr(j)(i:i)
        end do
      end do

      chard(1) = 7.62d0       !  automatic step-size control
      chard(2) = 0.1d0
      chard(3) = 0.5d0
      chard(4) = 5.0d0

      write(6,fmt="('chard =',5e12.5)") (chard(j),j=1,nmed)

!     ---------------------------------
!     Run KEK PEGS5 before calling HATCH
!     ---------------------------------
      write(6,100)
100   FORMAT(' PEGS5-call comes next'/)

!     ==========
      call pegs5
!     ==========
```

3.3. Step 3: Pre-hatch-call-initialization

The `npreci` is used to specify format for particle trajectories data and it is set to 2 in this user code for CGview. After initializing CG related parameters, `subroutine geomgt` is called to read CG input data and output CG information for CGview. `CSTA` and `CEND` are written before and after CG related data, respectively. The `ifto` which defines output unit of cg-data is set to 39 as the unit of trajectory data file for CGview. The number of region, `NREG`, is set by `izonin`.

```
      write(6,*) 'Read cg-related data'

!-----------------------------------------------
!     initilize cg related parameter
!-----------------------------------------------
      npreci=3     ! PICT data mode for CGView in free format

      ifti = 4     ! Input unit number for cg-data
      ifto = 39    ! Output unit number for PICT

      write(6,fmt="(' CG data')")
      call geomgt(ifti,6)  ! Read in CG data
      write(6,fmt="(' End of CG data',/)")

      if(npreci.eq.3) write(ifto,fmt="('CSTA-FREE-TIME')")
      if(npreci.eq.2) write(ifto,fmt="('CSTA-TIME')")

      rewind ifti
      call geomgt(ifti,ifto)! Dummy call to write geom info for ifto
      write(ifto,110)
110   FORMAT('CEND')

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

The material assignment is read in from input file (`egs5job.data`). Egs cut-off energy and

various option flags are set to each region except vacuum regions. In this user code, K & L-edge fluorescence option is turn-on.

After setting the seed, initialize the Ranlux random number generator.

```
!   Read material for each region from egs5job.data
      read(4,*) (med(i),i=1,nreg)

!   Set option except vacuum region
      do i=1,nreg-1
        if(med(i).ne.0) then
          iphter(i) = 0    ! Switches for PE-angle sampling
          iedgfl(i) = 1    ! K & L-edge fluorescence
          iauger(i) = 0    ! K & L-Auger
          iraylr(i) = 0    ! Rayleigh scattering
          lpolar(i) = 0    ! Linearly-polarized photon scattering
          incohr(i) = 0    ! S/Z rejection
          iprofr(i) = 0    ! Doppler broadening
          impacr(i) = 0    ! Electron impact ionization
        end if
      end do

!     ---------------------------------------------------------
!     Random number seeds.  Must be defined before call hatch
!     or defaults will be used.  inseed (1- 2^31)
!     ---------------------------------------------------------
      luxlev = 1
      inseed=1
      write(6,120) inseed
120   FORMAT(/,' inseed=',I12,5X,
     *         ' (seed for generating unique sequences of Ranlux)')

!     =============
      call rluxinit  ! Initialize the Ranlux random-number generator
!     =============
```

3.4. Step 4: Determination-of-incident-particle-parameters

Various source parameters like energy, position and direction are set. In this user code, 1.253 MeV photons normally incident along Z-axis to a point of (0.0, 0.0, -5.0). In CG geometry, it is possible to set `irin` automatically by setting `irin=0` .

```
! Define initial variables for incident particle normally incident
! on the slab
      iqin=0             ! Incident particle charge - photons
      ekein=1.253        ! Incident particle kinetic energy
      xin=0.0            ! Source position
      yin=0.0
      zin=-5.0
      uin=0.0            ! Moving along z axis
      vin=0.0
      win=1.0
      irin=0             ! Starting region (0: Automatic search in CG)
      wtin=1.0           ! Weight = 1 since no variance reduction used

!     pdf data for many source
      deltae=0.05        ! Energy bin of response

!---------------------------------------
!     Get source region from cg input data
!---------------------------------------
!
      if(irin.le.0.or.irin.gt.nreg) then
        call srzone(xin,yin,zin,iqin+2,0,irin)
        if(irin.le.0.or.irin.ge.nreg) then
          write(6,fmt="(' Stopped in MAIN. irin = ',i5)")irin
          stop
        end if
        call rstnxt(iqin+2,0,irin)
```

```
      end if
```

3.5. Step 5: hatch-call

Set `emaxe=0.D0` to get minimum upper energy of electrons in the material used, and then `subroutine hatch` is called.

Output the material data and parameters of each region to the result file (unit 1). Output the number of regions and the material number of each region to the trajectory file (unit 39).

```
      emaxe = 0.D0 ! dummy value to extract min(UE,UP+RM).

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

3.6. Step 6: Initialization-for-howfar

Define various parameters used for the geometry definition in this step. This part is not necessary in the case of using CG.

3.7. Step 7: Initialization-for-ausgab

The energy bin width is calculated from the source energy and the number of energy bin (50). `ncases` is a history number and `maxpict` is a number of histories to store trajectory data.

```
!     Energy bin width
      deltae=ekein / 50

!     Zero the variables
      depe=0.D0
      pefs=0.D0
      pef2s=0.D0
      tefs=0.D0
      tef2S=0.D0
      do j=1,50
        phs(j)=0.D0
        ph2s(j)=0.D0
        do ntype=1,3
          spec(ntype,j)=0.D0
          specs(ntype,j)=0.D0
          spec2s(ntype,j)=0.D0
        end do
      end do

!     Set histories
      ncases=10000
!     Set maximum number for pict
      maxpict=50
```

3.8. Step 8: Shower-call

`Subroutine shower` is called `ncases` times. Before `shower call-loop`, a batch number is written on the trajectory display file.

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

Summation of weight squared of above variables together with spectrum information are also stored for statistical analysis.

```
!     Write batch number
      write(39,fmt="('0    1')")

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

                                              ! -------------------------
      do i=1,ncases                           ! Start of shower call-loop
                                              ! -------------------------
!       ----------------------
!       Select incident energy
!       ----------------------

        wtin = 1.0

        wtsum = wtsum + wtin               ! Keep running sum of weights
        etot = ekein + 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

!       ----------------------
!       Select incident angle
!       ----------------------

!       ---------------------------------------------------
!       Print first NWRITE or NLINES, whichever comes first
!       ---------------------------------------------------
        if (ncount .le. nwrite .and. ilines .le. nlines) then
          ilines = ilines + 1
          write(6,280) etot,xin,yin,zin,uin,vin,win,iqin,irinn,idin
280       FORMAT(7G15.7,3I5)
        end if

!       ------------------------------------------------------------
!       Compare maximum energy of material data and incident energy
!       ------------------------------------------------------------
        if(etot+(1-iabs(iqin))*RM.gt.emaxe) then
          write(6,fmt="(' Stopped in MAIN.',
     1    ' (Incident kinetic energy + RM) > min(UE,UP+RM).')")
          stop
        end if

!       ---------------------------------------------------
!       Verify the normalization of source direction cosines
!       ---------------------------------------------------
        if(abs(uin*uin+vin*vin+win*win-1.0).gt.1.e-6) then
          write(6,fmt="(' Following source direction cosines are not',
     1    ' normalized.',3e12.5)")uin,vin,win
          stop
        end if

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

!       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
           phs(ie)=phs(ie)+wtin
           ph2s(ie)=ph2s(ie)+wtin*wtin
           tefs=tefs + wtin
           tef2s=tef2s + wtin*wtin
           if(depe .ge. ekein*0.999) then
             pefs=pefs +wtin
             pef2s=pef2s +wtin*wtin
           end if
```

```
          depe = 0.D0
        end if
        do ntype=1,3
          do ie=1,50
            specs(ntype,ie)=specs(ntype,ie)+spec(ntype,ie)
            spec2s(ntype,ie)=spec2s(ntype,ie)+
     *          spec(ntype,ie)*spec(ntype,ie)
            spec(ntype,ie)=0.D0
          end do
        end do

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

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

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

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

       call plotxyz(99,0,0,0.D0,0.D0,0.D0,0.D0,0,0.D0,0.D0)

       write(39,fmt="('9')")          ! Set end of batch for CG View
```

3.8.1. Statistical uncertainty: The uncertainty of obtained, x, is estimated using the method used in MCNP in this user code.

- Assume that the calculation calls for N "incident" particle histories.
- Assume that x_i is the result at the i-th history.
- Calculate the mean value of x :

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

- Estimate the variance associated with the distribution of x_i:

$$s^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \overline{x})^2 \simeq \overline{x^2} - (\overline{x})^2 \quad (\overline{x^2} = \frac{1}{N}\sum_{i=1}^{N} x_i^2). \tag{2}$$

- Estimate the variance associated with the distribution of $\overline{x}$:

$$s_{\overline{x}}^2 = \frac{1}{N}s^2 \simeq \frac{1}{N}[\overline{x^2} - (\overline{x})^2] \tag{3}$$

- Report the statistical error as:

$$s_{\overline{x}} \simeq [\frac{1}{N}(\overline{x^2} - \overline{x}^2)]^{1/2} \tag{4}$$

3.8.2. Output of results: After finishing all histories, obtained results are analyzed and written on output file. Average values and their statistical uncertainty are calculated form the sum of weight and the sum of weight squared.

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

!     ---------------
```

```
!     Peak efficiency
!     ---------------
      avpe = pefs/ncount
      pef2s=pef2s/ncount
      sigpe=dsqrt((pef2s-avpe*avpe)/ncount)
      avpe = avpe*100.0
      sigpe = sigpe*100.0
      write(6,350) avpe,sigpe
350   FORMAT(' Peak efficiency =',G11.4,'+-',G9.2,' %')

!     ----------------
!     Total efficiency
!     ----------------
      avte = tefs/ncount
      tef2s = tef2s/ncount
      sigte = dsqrt((tef2s-avte*avte)/ncount)
      avte = avte*100.0
      sigte = sigte*100.0
      write(6,360) avte,sigte
360   FORMAT(' Total efficiency =',G11.4,'+-',G9.2,' %')

!     --------------------------
!     Pulse height distribution
!     --------------------------
      write(6,370)
370   FORMAT(/' Pulse height distribution ')
      do ie=1,50
        elow=deltae*(ie-1)
        eup=deltae*ie

        avph = phs(ie)/ncount
        ph2s(ie)=ph2s(ie)/ncount
        sigph=dsqrt((ph2s(ie)-avph*avph)/ncount)
        avph = avph/deltae
        sigph= sigph/deltae
        write(6,380) eup,avph,sigph
380     FORMAT(' E (upper-edge --',G10.4,' MeV )=',G15.5,'+-',G15.5,
     *         ' counts/MeV/incident');
       end do
```

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

3.9. 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.

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, energy information corresponding to each particle type is scored.

```
!     ------------------------
!     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
      end if
!     ----------------------------------------------------------------
!     Print out particle transport information (if switch is turned on)
!     ----------------------------------------------------------------
!                       ========================
      if (iwatch .gt. 0) call swatch(iarg,iwatch)
!                       ========================
      if(iarg .ge. 5) return

!     ------------------------------------------
!     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
            ntype=1
            ie = e(np)/deltae +1
            if(ie .gt. 50) ie = 50
          elseif (iql .eq. -1) then        ! electron
            ntype=2
            ie = (e(np) - RM)/deltae +1
            if(ie .gt. 50) ie = 50
          else                             ! positron
            ntype=3
            ie = (e(np) - RM)/deltae +1
            if(ie .gt. 50) ie = 50
          end if
          spec(ntype,ie) = spec(ntype,ie) + wt(np)
        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(6,100) e(np),x(np),y(np),z(np),u(np),v(np),w(np),
     *               iql,irl,iarg
 100    FORMAT(4G15.7/3G15.7,3I5)
      end if

!     ----------------------------------
!     Output particle information for plot
!     ----------------------------------
      if (ncount.le.maxpict) then
        call plotxyz(iarg,np,iq(np),x(np),y(np),z(np),e(np),ir(np),
     *       wt(np),time(np))
      end if

      return

      end
```

3.10. Subroutine howfar

As far as CG is used, it is not necessary for user to change subroutine howfar at all.

For user's convenience, outline of subroutine howfar is described. 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 `ucnaicgv.f`.

4. Comparison of speed between ucnai.f and ucnaicgv.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, ucnaicgv.f needs 1.6 times longer cpu time than ucnai.f for the same problem. [2]

5. Exercise problems

5.1. Problem 1 : Calculation for NaI detector

Study variation of the results for the following cases.

1. Change to isotropic source of 1.253 MeV.
2. Change the source to 0.662 MeV photons from Cs-137.
3. Change source energy to 1.173 and 1.332 MeV photons of Co-60.
4. Increase detector thickness twice for 1.253 MeV photons.

5.2. 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 1.253 MeV photons.

5.3. Problem 3 : Air ionization chamber calculation

Change detector to air at 20°C and 1 atm and calculate absorbed energy for 1.253 MeV pencil beam photon. Air region have 7.62 cm diameter and 7.62 cm length and is surrounded by 0.5 cm aluminum wall. Air layer of 5 cm thickness exist outside aluminum wall.

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

6. Answer for exercises

It is recommended to save `egs5job.out` which is the result of `ucnaicgv.f` for 10,000 histories with a different file name like `nai.out` for comparisons with the results of following problems.

6.1. Problem 1

1. ^{137}Cs source

   - `cp ucnaicgv.f ucnaicgv1.f`
     You must use `copy ucnaicgv.f ucnaicgv1.f` or copy function of Windows in the case of DOS.
   - `cp ucnaicgv.data ucnaicgv1.data`
   - `cp ucnaicgv.inp ucnaicgv1.inp`
   - Modify `ucnaicgv1.f` as follows:
     - Modify source photon energy.
       Change
       ```
       ekein=1.253        ! Incident particle kinetic energy
       ```
       to
       ```
       ekein=0.662        ! Incident particle kinetic energy
       ```
   - Run `ucnaicgv1.f` by egs5run.

     - In the case of Linux or Cygwin
       Enter `ucnaicgv1` as the user code.
       Simply enter "return" as the file name for unit 4 and 25.
       Enter 1 for "Does this user code read from the terminal?".
     - In the case of DOS
       `egs5run ucnaicgv1`
   - Compare the calculated results with `nai.out`. Peak efficiency and total efficiency represent the results. These values are shown in the table below.

     Efficiencies before and after the modification

	Peak efficiency (P_{eff})	Total efficiency (T_{eff})
ucnaicgv	37.7 ± 0.5 %	76.5 ± 0.4 %
ucnaicgv1	59.2 ± 0.5 %	87.1 ± 0.3 %

     Here user should be careful on the point that efficiency value can have fluctuation on the order of statistical error depending on computer and compiler.

2. ^{60}Co source

   - `cp ucnaicgv.f ucnaicgv2.f`
   - `cp ucnaicgv.data ucnaicgv2.data`
   - `cp ucnaicgv.inp ucnaicgv2.inp`
   - Modify `ucnaicgv2.f` as follows:
     - Modify the maximum electron kinetic energy used.
       Change
       ```
       ekein=1.253        ! Incident particle kinetic energy
       ```
       to
       ```
       ekein=1.333        ! Incident particle kinetic energy
       ```

– Add sampling routines for source photon energy sampling.
Just after

```
!       ----------------------
!       Select incident energy
!       ----------------------
```

add followings.

```
        call randomset(rnnow)
        if(rnnow.le.0.5) then
          ekein=1.173
        else
          ekein=1.333
        end if
```

– Modify output statement concerning the source energy.
Change

```
      write(6,340) ekin
340   FORMAT(' Results for ',G15.5,'MeV photon'/)
```

to

```
      write(6,340)
340   FORMAT(' Results for Co-60 gamma-ray (1.173 and 1.333 MeV)'/)
```

- Run `ucnaicgv2.f` by egs5run.

  – In the case of Linux or Cygwin
  Enter `ucnaicgv2` as the user code.
  Simply enter "return" as the file name for unit 4 and 25.
  Enter 1 for "Does this user code read from the terminal?".
  – In the case of DOS
  `egs5run ucnaicgv2`
- Compare the calculated results with `nai.out`. Confirm that there are 2 peaks in the response corresponding to the source energies.

3. Isotropic source

- `cp ucnaicgv.f ucnaicgv3.f`
- `cp ucnaicgv.data ucnaicgv3.data`
- `cp ucnaicgv.inp ucnaicgv3.inp`
- Modify `ucnaicgv3.f` as follows:
  – Add a source direction sampling routine.
  Change

```
!       ----------------------
!       Select incident angle
!       ----------------------
```

to

```
!       ----------------------
!       Select incident angle
!       ----------------------
275       call randomset(rnnow)
          zi0=rnnow
          call randomset(rnnow)
          xi0=2.0*rnnow-1.0
          call randomset(rnnow)
          yi0=2.0*rnnow-1.0
          rr0=dsqrt(xi0*xi0+yi0*yi0+zi0*zi0)
          if(rr0.gt.1.0) go to 275
          win = zi0/rr0
          uin = xi0/rr0
          vin = yi0/rr0
```

- Run `ucnaicgv3.f` by egs5run.

  - In the case of Linux or Cygwin
    Enter `ucnaicgv3` as the user code.
    Simply enter "return" as the file name for unit 4 and 25.
    Enter 1 for "Does this user code read from the terminal?".
  - In the case of DOS
    `egs5run ucnaicgv3`

- Check the trajectories by CGview using `egs5job.pic`. Confirm that source photons were emitted isotropically.
- Compare the calculated results with `nai.out`.
  Example of result: P_{eff}=3.6 ± 0.2 %, T_{eff}=9.3 ± 0.3 %

4. Increase NaI detector thickness twice

- `cp ucnaicgv.f ucnaicgv4.f`
- `cp ucnaicgv.data ucnaicgv4.data`
- `cp ucnaicgv.inp ucnaicgv4.inp`
- Modify `ucnaicgv4.f` as follows:
  - Increase the detector length.
    Change

    ```
    tdet=7.62
    ```

    to

    ```
    tdet=7.62*2.0
    ```

  - This modification is not directly related to the calculation in the case of using cg. This is used to output the detector size on the result file. The real change of the geometry is done by the next change of cg data.
- Modify `ucnaicgv4.data` as follows:

```
RCC    1         0.00          0.0          0.0         0.00          0.0
                15.24         3.81
RCC    2         0.00          0.0         -0.5         0.00          0.0
                15.74         4.31
RCC    3         0.00          0.0         -0.6         0.00          0.0
                16.34         4.41
RCC    4         0.00          0.0        15.24         0.00          0.0
                  0.5         4.31
RCC    5         0.00          0.0         -5.6         0.00          0.0
                26.34         9.41
RCC    6          0.0          0.0        -10.0          0.0          0.0
                36.74         12.0
END
 Z1       +1
 Z2       +2      -1
 Z3       +3      -2      -4
 Z4       +4
 Z5       +5      -3
 Z6       +6      -5
END
 1    0    2    3    4    0
```

- Check `ucnaicgv4.data`.

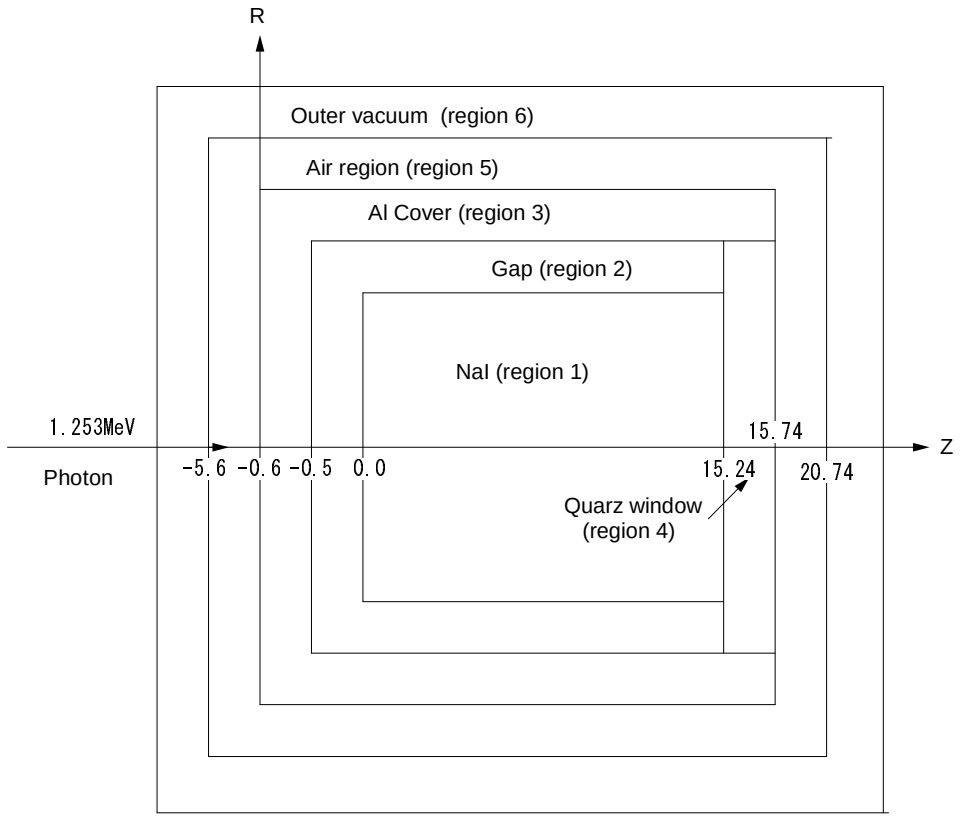

Figure 4: Geometry of ucnaicgv4.f

  - Check `ucnaicgv4.data` by using CGView as follows;
    Select "Making geometry data" of File option.
    Select Open File and assign `ucnaicgv4.data` by changing file type to "all files".
    Geometry is displayed when you select OK.
    Select "Geometry Check" of Environment option.
    Select "Check Start".
- Run `ucnaicgv4.f` by egs5run.
  - In the case of Linux or Cygwin
    Enter `ucnaicgv4` as the user code.
    Simply enter "return" as the file name for unit 4 and 25.
    Enter 1 for "Does this user code read from the terminal?".
  - In the case of DOS
    `egs5run ucnaicgv4`
- Compare the calculated results with `nai.out`.
  Example of the result: P_{eff}=54.0 ± 0.5%, T_{eff}=93.6 ± 0.3 %

6.2. Problem 2

1. `cp ucnaicgv.f ucnaicgv5.f`
2. `cp ucnaicgv.data ucnaicgv5.data`
3. `cp ucnaicgv.inp ucnaicgv5.inp`
4. Modify `ucnaicgv5.f` as follows:
   - Modify the material data used.

Change

```
medarr(1)='NAI                     '
```

to

```
medarr(1)='GE                      '
```

`Here, data of 24 characters must be specified as medarr(1).`

User should be careful on the point that data of 24 characters are necessary for `medarr` .

5. Modify `ucnaicgv5.inp` as follows:

```
Modify
COMP
 &INP NE=2,RHO=3.67, PZ=1,1,IRAYL=1 /END
NAI                              NAI
NA I
```

to

```
ELEM
 &INP IRAYL=1 /END
GE                               GE
GE
```

User should be careful on the point that the second "`GE`" must start from 31-th column.

6. Run `ucnaicgv5.f` by egs5run.

   - In the case of Linux or Cygwin
     Enter `ucnaicgv5` as the user code.
     Simply enter "return" as the file name for unit 4 and 25.
     Enter 1 for "Does this user code read from the terminal?".
   - In the case of DOS
     `egs5run ucnaicgv5`

7. Compare the calculated results with `nai.out`. Example of the result: P_{eff}=39.8 $\pm$ 0.5 %, T_{eff}=87.7 $\pm$ 0.3 %

6.3. Problem 3

1. `cp ucnaicgv.f ucioncgv.f`
2. `cp ucnaicgv.data ucioncgv.data`
3. `cp ucnaicgv.inp ucioncgv.inp`
4. Modify `ucioncgv.f` as follows:

   - Add variables used.
     Change

     ```
     * xi0,yi0,zi0
     ```

     to

     ```
     * xi0,yi0,zi0,avab,depes,depe2s,sigab
     ```

Change

```
     * xi0,yi0,zi0
```

to

```
     * xi0,yi0,zi0,avab,depes,depe2s,sigab
```

- Modify the number of materials used.
  Change

```
      nmed=4
```

  を

```
      nmed=2
```

- Modify the name of materials.
  Change

```
      medarr(1)='NAI                    '
      medarr(2)='AL                     '
      medarr(3)='QUARTZ                 '
      medarr(4)='AIR-AT-NTP             '
```

  to

```
      medarr(1)='AIR-AT-NTP             '
      medarr(2)='AL                     '
```

- Delete unnecessary chard for non existing material number. Delete following lines

```
      chard(3) = 0.5d0
      chard(4) = 5.0d0
```

- Modify variables to be initialized.
  Change

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

  to

```
!     Zero the variables
      depe=0.D0
      depes=0.D0
      depe2s=0.D0
```

- Increase history number to 100,000. Change

```
!     Set histories
      ncases=10000
```

  to

```
!     Set histories
      ncases=100000
```

- Add routines to sum absorbed energy in air.
  Change

```
        if (depe .gt. 0.D0) then
          ie=depe/deltae + 1
```

  to

```
        if (depe .gt. 0.D0) then
          depes=depes+depe
          depe2s=depe2s+depe*depe
          ie=depe/deltae + 1
```

- Modify statements related geometry output.

  Change

```
      tdet=7.62
      rdet=3.81
      tcov=0.1
      rtcov=0.1
      tgap=0.5
      rtgap=0.5
      write(6,330) tdet,rdet,tcov,rtcov,tgap,rtgap
330   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'/)
```

  to

```
      tdet=7.62
      rdet=3.81
      tcov=0.5
      rtcov=0.5
      write(6,330) tdet,rdet,tcov,rtcov
330   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 calculated average absorbed energy of air and its statistical error.

  Change

```
!     -------------------------
!     Pulse height distribution
!     -------------------------
      write(6,370)
```

  to

```
!     ----------------------
!     Absorbed energy in air
!     ----------------------
      avab = depes/ncount
      depe2s = depe2s/ncount
      sigab = sqrt((depe2s - avab*avab)/ncount)
      write(6,362) avab,sigab
362   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(6,364) avab,sigab
364   FORMAT(' Output current =',G15.5,'+-',G15.5,' C/photon')

!     -------------------------
!     Pulse height distribution
!     -------------------------
      write(6,370)
```

- Modify score condition at the detector region in ausgab to distinguish from air outside detector.

```
      if (med(irl) .eq. 1) then
        depe = depe + edepwt
```

  to

```
if (irl .eq. 1) then
  depe = depe + edepwt
```

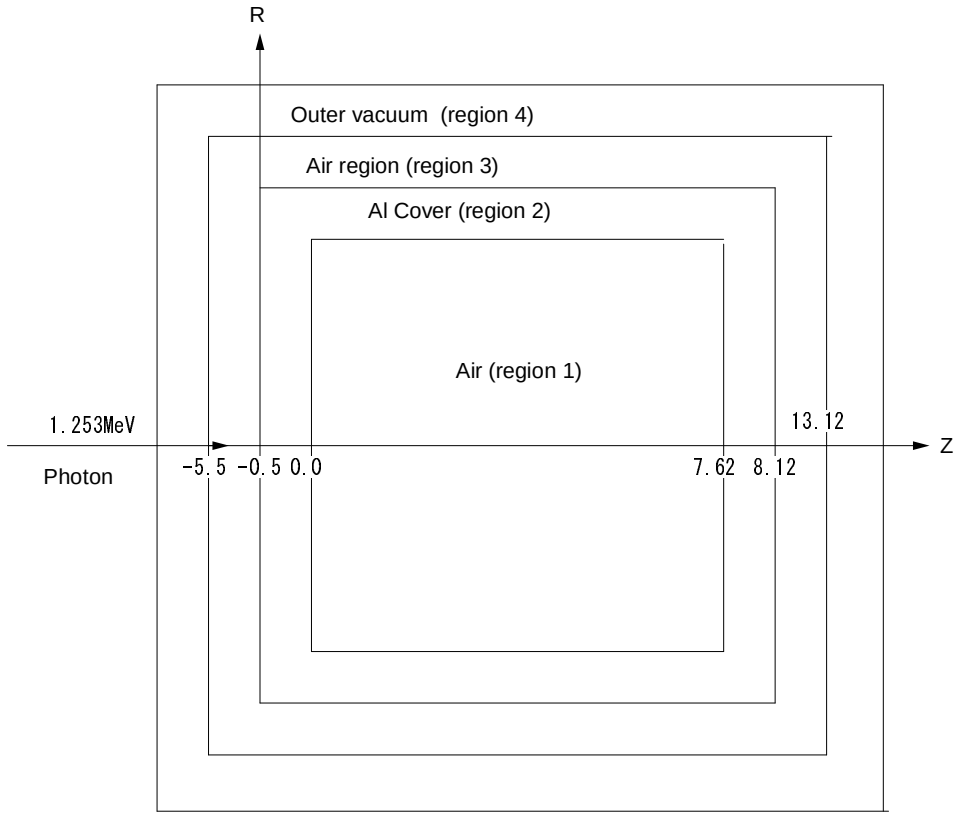

Figure 5: Geometry of ucioncgv.f

5. Modify `ucioncgv.data` as follows:

```
RCC    1        0.00          0.0          0.0          0.00          0.0
                7.62         3.81
RCC    2        0.00          0.0         -0.5          0.00          0.0
                8.62         4.31
RCC    3        0.00          0.0         -5.5          0.00          0.0
               18.62         9.31
RCC    4        0.00          0.0         -6.0          0.00          0.0
               20.62        10.31
END
Z1         +1
Z2         +2       -1
Z3         +3       -2
Z4         +4       -3
END
  1    2    1    0
```

6. Check `ucioncgv.data`.

   - `cp ucioncgv.data ucioncgv.geo`.
   - Check `ucioncgv.geo` by using CGView as follows;
     Select "Making geometry data" of File option.
     Select Open File and assign `ucioncgv.geo`".
     Geometry is displayed when you select OK.

Select "Geomtry Check" of Environment option.
Select "Check Start".

7. Modify `ucioncgv.inp` as follows.

```
MIXT
 &INP NE=3,RHO= 1.2929E-03,RHOZ= 0.755,0.232,0.013,
      GASP=0.93174, IRAYL=1 /END
AIR-AT-NTP                    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 IRAYL=1 /END
AL                            AL
AL
ENER
 &INP AE=0.521,AP=0.010,UE=2.511,UP=2.0 /END
PWLF
 &INP  /END
DECK
 &INP  /END
```

8. Run `ucioncgv.f` by egs5run.

   - In the case of Linux or Cygwin
     Enter `ucioncgv` as the user code.
     Simply enter "return" as the file name for unit 4 and 25.
     Enter 1 for "Does this user code read from the terminal?".
   - In the case of DOS
     `egs5run ucioncgv`

9. Check the trajectories by CGview using `egs5job.pic`.

10. Check the calculated results.
    Example of result are shown in the table below.

Result of ionization chamber calculation	
Peak efficiency P_{eff} (%)	0.0
Total efficiency T_{eff} (%)	0.73 ± 0.3e-1
Absorbed energy (MeV/γ)	0.15e-3 ± 0.7e-5
Output (C/γ)	0.69e-18 ± 0.3e-19

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).

[2] T. Sugita, T. Torii, A. Takamura, "Incorporating Combinatorial Geometry to the EGS5 Code and Its Speed-Up", Twelfth EGS User's Meeting in Japan, KEK Proc. **2005-10**, 7-21, (KEK, Tsukuba, 9 - 11 Aug. 2005).

Appendix 1 Full listings of `ucnaicgv.f`

```
!*****************************************************************************
!*****************************  KEK, High Energy Accelerator Research  *
!*****************************  Organization                           *
!*** u c n a i c g v *********                                         *
!*****************************    EGS5.0 USER CODE -  28 Jul 2012/1430 *
!*****************************************************************************
!* This is a general User Code based on the cg geometry scheme.        *
!*****************************************************************************
!                                                                     *
!  PROGRAMMERS:  H. Hirayama                                          *
!                Applied Research Laboratory                          *
!                KEK, High Energy Accelerator Research Organization   *
!                1-1, Oho, Tsukuba, Ibaraki, 305-0801                 *
!                Japan                                                *
!                                                                     *
!                E-mail:      hideo.hirayama@kek.jp                   *
!                Telephone:   +81-29-864-5451                         *
!                Fax:         +81-29-864-4051                         *
!                                                                     *
!                Y. Namito                                            *
!                Radiation Science Center                             *
!                Applied Research Laboratory                          *
!                KEK, High Energy Accelerator Research Organization   *
!                1-1, Oho, Tsukuba, Ibaraki, 305-0801                 *
!                Japan                                                *
!                                                                     *
!                E-mail:      yoshihito.namito@kek.jp                 *
!                Telephone:   +81-29-864-5489                         *
!                Fax:         +81-29-864-1993                         *
!                                                                     *
!*****************************************************************************
!*****************************************************************************
! The ucnaicgv.f User Code requires a cg-input file only              *
! (e.g., ucnaicgv.data).                                              *
! The following shows the geometry for ucnaicg.data.                  *
! Input data for CG geometry must be written at the top of data-input  *
! file together with material assignment to each region.  Cg-data can  *
! be checked by CGview.                                               *
! This user code corresponds to ucnai3cgp.mor for egs4.               *
! Use Ranlux random number generator.                                 *
!*****************************************************************************
!                                                                     *
!              ------------------------                               *
!              cg Geometry (ucnaicgv)                                 *
!              ------------------------                               *
!                                                                     *
!                                                                     *
!                 R                                                   *
!                 ^                                                   *
!                 |                                                   *
!            +----+----+----+--------+------+---                      *
!            |                                                        *
!            |     Outer vacuum region                                *
!            +    +----+----+--------+------+-----+ R=9.41            *
!            |    |         Air                  |                   *
!            |    |    +---------------------+     + R=4.41          *
!            |    |    |    Al cover         |     |                 *
!            |    +    +    +-----------+---+     + R=4.31            *
!            |    |    |    | Gap       |   |     |                   *
!            +    +    +    +   +-------+   +     + R=3.81            *
!            |    |    |    |   |       |Quartz   |                   *
!            |    |    |    |   |  NaI  |   |     |                   *
! 1.253 MeV  |    |    |    |   |       |   |     |                   *
!       ============>--+----+---+-------+---+-----+--------> Z        *
!     photons  -5.6 -0.6 -0.5 0.0    7.62 8.12  13.12                 *
!                 -5.0                                                *
!                                                                     *
!                                                                     *
!                                                                     *
!*****************************************************************************
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12

!-----------------------------------------------------------------------------
!------------------------------ main code ------------------------------------
!-----------------------------------------------------------------------------
!-----------------------------------------------------------------------------
! Step 1: Initialization
!-----------------------------------------------------------------------------

      implicit none
```

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

      include 'include/egs5_bounds.f'
      include 'include/egs5_brempr.f'
      include 'include/egs5_edge.f'
      include 'include/egs5_media.f'
      include 'include/egs5_misc.f'
      include 'include/egs5_thresh.f'
      include 'include/egs5_uphiot.f'
      include 'include/egs5_useful.f'
      include 'include/egs5_usersc.f'
      include 'include/egs5_userxt.f'
      include 'include/randomm.f'

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

      include 'auxcommons/edata.f'
      include 'auxcommons/etaly1.f'
      include 'auxcommons/instuf.f'
      include 'auxcommons/lines.f'
      include 'auxcommons/nfac.f'
      include 'auxcommons/watch.f'

!     ------------------
!     cg related COMMONs
!     ------------------
      include 'auxcommons/geom_common.f' ! geom-common file
      integer irinn

      common/totals/                            ! Variables to score
     * depe,deltae,spec(3,50),maxpict
      real*8 depe,deltae,spec
      integer maxpict

!**** 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,pefs,pef2s,
     * rr0,sigpe,sigte,sigph,sigspg,sigspe,sigspp,tefs,tef2s,wtin,wtsum,
     * xi0,yi0,zi0

      real*8
     * phs(50),ph2s(50),specs(3,50),spec2s(3,50)

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

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

      integer
     * i,icases,idin,ie,ifti,ifto,ii,iiz,imed,ireg,isam,
     * izn,nlist,j,k,n,ner,ntype

      character*24 medarr(MXMED)

!     ----------
!     Open files
!     ----------
!-----------------------------------------------------------------
!     Units 7-26 are used in pegs and closed.  It is better not
!     to use as output file. If they are used, they must be opened
!     after getcg etc. Unit for pict must be 39.
!-----------------------------------------------------------------

      open(6,FILE='egs5job.out',STATUS='unknown')
      open(4,FILE='egs5job.inp',STATUS='old')
      open(39,FILE='egs5job.pic',STATUS='unknown')
```

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

!-----------------------------------------------------------------------
! Step 2: pegs5-call
!-----------------------------------------------------------------------

!     ---------------------------------
!     Define media before calling PEGS5
!     ---------------------------------

      nmed=4
      if(nmed.gt.MXMED) then
        write(6,'(A,I4,A,I4,A/A)')
     *     ' nmed (',nmed,') larger than MXMED (',MXMED,')',
     *     ' MXMED in iclude/egs5_h.f must be increased.'
        stop
      end if

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

      medarr(1)='NAI                     '
      medarr(2)='AL                      '
      medarr(3)='QUARTZ                  '
      medarr(4)='AIR-AT-NTP              '

      do j=1,nmed
        do i=1,24
          media(i,j)=medarr(j)(i:i)
        end do
      end do

      chard(1) = 7.62d0       !  automatic step-size control
      chard(2) = 0.1d0
      chard(3) = 0.5d0
      chard(4) = 5.0d0

      write(6,fmt="('chard =',5e12.5)") (chard(j),j=1,nmed)

!     ----------------------------------
!     Run KEK PEGS5 before calling HATCH
!     ----------------------------------
      write(6,100)
100   FORMAT('PEGS5-call comes next'/)

!     ==========
      call pegs5
!     ==========

!-----------------------------------------------------------------------
! Step 3: Pre-hatch-call-initialization
!-----------------------------------------------------------------------
      write(6,*) 'Read cg-related data'

!-------------------------------------------------
!     Initialize CG related parameters
!-------------------------------------------------
      npreci=3     ! PICT data mode for CGView in free format

      ifti = 4     ! Input unit number for cg-data
      ifto = 39    ! Output unit number for PICT

      write(6,fmt="(' CG data')")
      call geomgt(ifti,6)  ! Read in CG data
      write(6,fmt="(' End of CG data',/)")

      if(npreci.eq.3) write(ifto,fmt="('CSTA-FREE-TIME')")
      if(npreci.eq.2) write(ifto,fmt="('CSTA-TIME')")

      rewind ifti
      call geomgt(ifti,ifto)! Dummy call to write geom info for ifto
      write(ifto,110)
110   FORMAT('CEND')

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

```
!   Read material for each region from egs5job.data
      read(4,*) (med(i),i=1,nreg)

!   Set option except vacuum region
      do i=1,nreg-1
        if(med(i).ne.0) then
          iphter(i) = 1    ! Switches for PE-angle sampling
          iedgfl(i) = 1    ! K & L-edge fluorescence
          iauger(i) = 0    ! K & L-Auger
          iraylr(i) = 0    ! Rayleigh scattering
          lpolar(i) = 0    ! Linearly-polarized photon scattering
          incohr(i) = 0    ! S/Z rejection
          iprofr(i) = 0    ! Doppler broadening
          impacr(i) = 0    ! Electron impact ionization
        end if
      end do

!     ------------------------------------------------------------
!     Random number seeds.  Must be defined before call hatch
!     or defaults will be used.  inseed (1- 2^31)
!     ------------------------------------------------------------
      luxlev = 1
      inseed=1
      write(6,120) inseed
120   FORMAT(/,' inseed=',I12,5X,
     *         ' (seed for generating unique sequences of Ranlux)')

!     =============
      call rluxinit  ! Initialize the Ranlux random-number generator
!     =============

!-----------------------------------------------------------------------
! Step 4:  Determination-of-incident-particle-parameters
!-----------------------------------------------------------------------
! Define initial variables for incident particle normally incident
! on the slab
      iqin=0             ! Incident particle charge - photons
      ekein=1.253        ! Incident particle kinetic energy
      xin=0.0            ! Source position
      yin=0.0
      zin=-5.0
      uin=0.0            ! Moving along z axis
      vin=0.0
      win=1.0
      irin=0             ! Starting region (0: Automatic search in CG)
      wtin=1.0           ! Weight = 1 since no variance reduction used

!     pdf data for many source
      deltae=0.05        ! Energy bin of response

!-----------------------------------------
!     Get source region from cg input data
!-----------------------------------------
!
      if(irin.le.0.or.irin.gt.nreg) then
        call srzone(xin,yin,zin,iqin+2,0,irin)
        if(irin.le.0.or.irin.ge.nreg) then
          write(6,fmt="(' Stopped in MAIN. irin = ',i5)")irin
          stop
        end if
        call rstnxt(iqin+2,0,irin)
      end if


!-----------------------------------------------------------------------
! Step 5:   hatch-call
!-----------------------------------------------------------------------
!-----------------------------------------------------------------------
      emaxe = 0.D0 ! dummy value to extract min(UE,UP+RM).

      write(6,130)
130   format(/' Call hatch to get cross-section data')

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

```
      write(6,140)
140   FORMAT(/,' HATCH-call comes next',/)

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

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

! ------------------------------------------------------------
! Print various data associated with each media (not region)
! ------------------------------------------------------------
      write(6,150)
150   FORMAT(/,' Quantities associated with each MEDIA:')
      do j=1,nmed
        write(6,160) (media(i,j),i=1,24)
160     FORMAT(/,1X,24A1)
        write(6,170) rhom(j),rlcm(j)
170     FORMAT(5X,' rho=',G15.7,' g/cu.cm     rlc=',G15.7,' cm')
        write(6,180) ae(j),ue(j)
180     FORMAT(5X,' ae=',G15.7,' MeV    ue=',G15.7,' MeV')
        write(6,190) ap(j),up(j)
190     FORMAT(5X,' ap=',G15.7,' MeV    up=',G15.7,' MeV',/)
      end do

! ------------------------------------------------------
! Print media and cutoff energies assigned to each region
! ------------------------------------------------------
      do i=1,nreg
        if (med(i) .eq. 0) then
          write(6,200) i
200       FORMAT(' medium(',I3,')=vacuum')
        else
          write(6,210) i,(media(ii,med(i)),ii=1,24),ecut(i),pcut(i)
210       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(6,220) izn
220           FORMAT('   X-ray information for Z=',I3)
              write(6,230) (ekx(ii,izn),ii=1,10)
230           FORMAT('   K-X-ray energy in keV',/,
     *               4G15.5,/,4G15.5,/,2G15.5)
              write(6,240) (elx1(ii,izn),ii=1,8)
240           FORMAT('   L-1 X-ray in keV',/,4G15.5,/,4G15.5)
              write(6,250) (elx2(ii,izn),ii=1,5)
250           FORMAT('   L-2 X-ray in keV',/,5G15.5)
              write(6,260) (elx3(ii,izn),ii=1,7)
260           FORMAT('   L-3 X-ray in keV',/,4G15.5,/,3G15.5)
            end do
          end if
        end if
      end do

      write(39,fmt="('MSTA')")
      write(39,fmt="(i4)") nreg
      write(39,fmt="(15i4)") (med(i),i=1,nreg)
      write(39,fmt="('MEND')")

!-----------------------------------------------------------------------
! Step 6:  Initialization-for-howfar
!-----------------------------------------------------------------------
!-----------------------------------------------------------------------
! Step 7:  Initialization-for-ausgab
!-----------------------------------------------------------------------

      ncount = 0
      ilines = 0
      nwrite = 10
```

```
      nlines = 10
      idin = -1
      totke = 0.
      wtsum = 0.
      iwatch=0

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

      write(6,270)
270   FORMAT(//,' Energy/Coordinates/Direction cosines/etc.',/,
     *        6X,'e',14X,'x',14X,'y',14X,'z',
     *        14X,'u',14X,'v',14X,'w',11X,'iq',3X,'ir',1X,'iarg',/)

!     Energy bin width
      deltae=ekein / 50

!     Zero the variables
      depe=0.D0
      pefs=0.D0
      pef2s=0.D0
      tefs=0.D0
      tef2S=0.D0
      do j=1,50
        phs(j)=0.D0
        ph2s(j)=0.D0
        do ntype=1,3
          spec(ntype,j)=0.D0
          specs(ntype,j)=0.D0
          spec2s(ntype,j)=0.D0
        end do
      end do

!     Set histories
      ncases=10000
!     Set maximum number for pict
      maxpict=50

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

!-----------------------------------------------------------------------
! Step 8:  Shower-call
!-----------------------------------------------------------------------
!     Write batch number
      write(39,fmt="('0    1')")

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

                                          ! -------------------------
      do i=1,ncases                       ! Start of shower call-loop
                                          ! -------------------------
!       ----------------------
!       Select incident energy
!       ----------------------

        wtin = 1.0

        wtsum = wtsum + wtin              ! Keep running sum of weights
        etot = ekein + iabs(iqin)*RM       ! Incident total energy (MeV)
        if(iqin.eq.1) then              ! Available K.E. (MeV) in system
          availke = ekein + 2.0*RM       ! for positron
        else                           ! Available K.E. (MeV) in system
          availke = ekein               ! for photon and electron
        end if

        totke = totke + availke                ! Keep running sum of KE

!       ----------------------
!       Select incident angle
!       ----------------------

!       ------------------------------------------------
!       Print first NWRITE or NLINES, whichever comes first
!       ------------------------------------------------
        if (ncount .le. nwrite .and. ilines .le. nlines) then
```

```
          ilines = ilines + 1
          write(6,280) etot,xin,yin,zin,uin,vin,win,iqin,irin,idin
280       FORMAT(7G15.7,3I5)
        end if

!       ------------------------------------------------------------
!       Compare maximum energy of material data and incident energy
!       ------------------------------------------------------------
        if(etot+(1-iabs(iqin))*RM.gt.emaxe) then
          write(6,fmt="(' Stopped in MAIN.',
     1    ' (Incident kinetic energy + RM) > min(UE,UP+RM).')")
          stop
        end if

!       ------------------------------------------------------
!       Verify the normalization of source direction cosines
!       ------------------------------------------------------
        if(abs(uin*uin+vin*vin+win*win-1.0).gt.1.e-6) then
          write(6,fmt="(' Following source direction cosines are not',
     1    ' normalized.',3e12.5)")uin,vin,win
          stop
        end if

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

!       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
           phs(ie)=phs(ie)+wtin
           ph2s(ie)=ph2s(ie)+wtin*wtin
           tefs=tefs + wtin
           tef2s=tef2s + wtin*wtin
           if(depe .ge. ekein*0.999) then
             pefs=pefs +wtin
             pef2s=pef2s +wtin*wtin
           end if
           depe = 0.D0
         end if
         do ntype=1,3
           do ie=1,50
             specs(ntype,ie)=specs(ntype,ie)+spec(ntype,ie)
             spec2s(ntype,ie)=spec2s(ntype,ie)+
     *           spec(ntype,ie)*spec(ntype,ie)
             spec(ntype,ie)=0.D0
           end do
         end do

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

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

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

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

       call plotxyz(99,0,0,0.D0,0.D0,0.D0,0.D0,0,0.D0,0.D0)

       write(39,fmt="('9')")          ! Set end of batch for CG View

       tt=etime(tarray)
       tt1=tarray(1)
       cputime=tt1-tt0
       write(6,300) cputime
300    format(' Elapsed Time (sec)=',G15.5)

!-------------------------------------------------------------------------
! Step 9:  Output-of-results
!-------------------------------------------------------------------------
```

```
      write(6,310) ncount,ncases,totke
310   FORMAT(/,' Ncount=',I10,' (actual cases run)',/,
     *       ' Ncases=',I10,' (number of cases requested)',/,
     *       ' TotKE =',G15.5,' (total KE (MeV) in run)')

      if (totke .le. 0.D0) then
        write(6,320) totke,availke,ncount
320     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(6,330) tdet,rdet,tcov,rtcov,tgap,rtgap
330   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'/)

      write(6,340) ekein
340   FORMAT(' Results for ',G15.5,'MeV photon'/)

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

!     ---------------
!     Peak efficiency
!     ---------------
      avpe = pefs/ncount
      pef2s=pef2s/ncount
      sigpe=dsqrt((pef2s-avpe*avpe)/ncount)
      avpe = avpe*100.0
      sigpe = sigpe*100.0
      write(6,350) avpe,sigpe
350   FORMAT(' Peak efficiency =',G11.4,'+-',G9.2,' %')

!     ----------------
!     Total efficiency
!     ----------------
      avte = tefs/ncount
      tef2s = tef2s/ncount
      sigte = dsqrt((tef2s-avte*avte)/ncount)
      avte = avte*100.0
      sigte = sigte*100.0
      write(6,360) avte,sigte
360   FORMAT(' Total efficiency =',G11.4,'+-',G9.2,' %')

!     --------------------------
!     Pulse height distribution
!     --------------------------
      write(6,370)
370   FORMAT(/' Pulse height distribution ')
      do ie=1,50
        elow=deltae*(ie-1)
        eup=deltae*ie

        avph = phs(ie)/ncount
        ph2s(ie)=ph2s(ie)/ncount
        sigph=dsqrt((ph2s(ie)-avph*avph)/ncount)
        avph = avph/deltae
        sigph= sigph/deltae
        write(6,380) eup,avph,sigph
380     FORMAT(' E (upper-edge --',G10.4,' MeV )=',G15.5,'+-',G15.5,
     *         ' counts/MeV/incident')
       end do

!     ------------------------------------------------------------
!     Particle spectrum.  Incident particle spectrum to detector.
!     ------------------------------------------------------------
      write(6,400)
400   FORMAT(/' Particle spectrum crossing the detector plane'/
```

```
     *       30X,'particles/MeV/source photon'/
     *       ' Upper energy',11X,'  Gamma',18X,' Electron',
     *       14X,' Positron')

      do ie=1,50
        elow=deltae*(ie-1)
        eup=deltae*ie

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

        avspg = specs(1,ie)/ncount
        spec2s(1,ie)=spec2s(1,ie)/ncount
        sigspg=dsqrt((spec2s(1,ie)-avspg*avspg)/ncount)
        avspg=avspg/deltae
        sigspg= sigspg/deltae

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

        avspe = specs(2,ie)/ncount
        spec2s(2,ie)=spec2s(2,ie)/ncount
        sigspe=dsqrt((spec2s(2,ie)-avspe*avspe)/ncount)
        avspe= avspe/deltae
        sigspe= sigspe/deltae

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

        avspp = specs(3,ie)/ncount
        spec2s(3,ie)=spec2s(3,ie)/ncount
        sigspp=dsqrt((spec2s(3,ie)-avspp*avspp)/ncount)
        avspp= avspp/deltae
        sigspp= sigspp/deltae

        write(6,410) eup,avspg,sigspg,avspe,sigspe,avspp,sigspp
410     FORMAT(G10.5,' MeV--',3(G12.5,'+-',G12.5))
      end do

      nlist=1


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

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

      stop

      end

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

!------------------------------ausgab.f--------------------------------
! Version:   080708-1600
! 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 AUSGAB to:
!
!   1) Score energy deposition
!   2) Score particle information enter to detector from outside
!   3) Print out particle transport information
!   4) call plotxyz if imode=0

! ---------------------------------------------------------------------
```

```
      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 'auxcommons/aux_h.f'   ! Auxiliary-code "header" file

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

      common/totals/                           ! Variables to score
     * depe,deltae,spec(3,50),maxpict
      real*8 depe,deltae,spec
      integer maxpict

      integer                                                ! Arguments
     * iarg

      real*8                                          ! Local variables
     * edepwt

      integer
     * ie,iql,irl,ntype

!     ------------------------
!     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
      end if

!     ---------------------------------------------------------------------
!     Print out particle transport information (if switch is turned on)
!     ---------------------------------------------------------------------
!                       ========================
      if (iwatch .gt. 0) call swatch(iarg,iwatch)
!                       ========================
      if(iarg .ge. 5) return

!     ------------------------------------------
!     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
            ntype=1
            ie = e(np)/deltae +1
            if(ie .gt. 50) ie = 50
          elseif (iql .eq. -1) then         ! electron
            ntype=2
            ie = (e(np) - RM)/deltae +1
            if(ie .gt. 50) ie = 50
          else                              ! positron
            ntype=3
            ie = (e(np) - RM)/deltae +1
            if(ie .gt. 50) ie = 50
          end if
          spec(ntype,ie) = spec(ntype,ie) + wt(np)
```

```
        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(6,100) e(np),x(np),y(np),z(np),u(np),v(np),w(np),
     *               iql,irl,iarg
 100    FORMAT(7G15.7,3I5)
      end if

!     ------------------------------------
!     Output particle information for plot
!     ------------------------------------
      if (ncount.le.maxpict) then
        call plotxyz(iarg,np,iq(np),x(np),y(np),z(np),e(np),ir(np),
     *       wt(np),time(np))
      end if

      return

      end

!-------------------------last line of ausgab.f-------------------------
!-------------------------------howfar.f--------------------------------
! Version:   070627-1600
! Reference: T. Torii and T. Sugita, "Development of PRESTA-CG
! Incorporating 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
c
      include 'include/egs5_h.f'       ! Main EGS "header" file
      include 'include/egs5_epcont.f'  ! COMMONs required by EGS5 code
      include 'include/egs5_stack.f'
      include 'auxcommons/geom_common.f' ! geom-common file
c
c
      integer i,j,jjj,ir_np,nozone,jty,kno
      integer irnear,irnext,irlold,irlfg,itvlfg,ihitcg
      double precision xidd,yidd,zidd,x_np,y_np,z_np,u_np,v_np,w_np
      double precision tval,tval0,tval00,tval10,tvalmn,delhow
      double precision atvaltmp
      integer iq_np
c
      ir_np = ir(np)
      iq_np = iq(np) + 2
c
      if(ir_np.le.0) then
        write(6,*) 'Stopped in howfar with ir(np) <=0'
        stop
      end if
c
      if(ir_np.gt.izonin) then
        write(6,*) 'Stopped in howfar with ir(np) > izonin'
        stop
      end if
c
      if(ir_np.EQ.izonin) then
        idisc=1
        return
      end if
c
      tval=1.d+30
      itvalm=0
c
c     body check
```

```
      u_np=u(np)
      v_np=v(np)
      w_np=w(np)
      x_np=x(np)
      y_np=y(np)
      z_np=z(np)
c
      do i=1,nbbody(ir_np)
        nozone=ABS(nbzone(i,ir_np))
        jty=itblty(nozone)
        kno=itblno(nozone)
c     rpp check
        if(jty.eq.ityknd(1)) then
          if(kno.le.0.or.kno.gt.irppin) go to 190
          call rppcg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c     sph check
        elseif(jty.eq.ityknd(2)) then
          if(kno.le.0.or.kno.gt.isphin) go to 190
          call sphcg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c     rcc check
        elseif(jty.eq.ityknd(3)) then
          if(kno.le.0.or.kno.gt.irccin) go to 190
          call rcccg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c     trc check
        elseif(jty.eq.ityknd(4)) then
          if(kno.le.0.or.kno.gt.itrcin) go to 190
          call trccg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c     tor check
        elseif(jty.eq.ityknd(5)) then
          if(kno.le.0.or.kno.gt.itorin) go to 190
          call torcg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c     rec check
        elseif(jty.eq.ityknd(6)) then
          if(kno.le.0.or.kno.gt.irecin) go to 190
          call reccg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c     ell check
        elseif(jty.eq.ityknd(7)) then
          if(kno.le.0.or.kno.gt.iellin) go to 190
          call ellcg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c     wed check
        elseif(jty.eq.ityknd(8)) then
          if(kno.le.0.or.kno.gt.iwedin) go to 190
          call wedcg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c     box check
        elseif(jty.eq.ityknd(9)) then
          if(kno.le.0.or.kno.gt.iboxin) go to 190
          call boxcg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c     arb check
        elseif(jty.eq.ityknd(10)) then
          if(kno.le.0.or.kno.gt.iarbin) go to 190
          call arbcg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c     hex check
        elseif(jty.eq.ityknd(11)) then
          if(kno.le.0.or.kno.gt.ihexin) go to 190
          call hexcg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c     haf check
        elseif(jty.eq.ityknd(12)) then
          if(kno.le.0.or.kno.gt.ihafin) go to 190
          call hafcg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c     tec check
        elseif(jty.eq.ityknd(13)) then
          if(kno.le.0.or.kno.gt.itecin) go to 190
          call teccg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c     gel check
        elseif(jty.eq.ityknd(14)) then
          if(kno.le.0.or.kno.gt.igelin) go to 190
          call gelcg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c
c**** add new geometry in here
c
       end if
  190  continue
      end do
c
```

```
      irnear=ir_np
      if(itvalm.eq.0) then
        tval0=cgeps1
        xidd=x_np+tval0*u_np
        yidd=y_np+tval0*v_np
        zidd=z_np+tval0*w_np
  310   continue
          if(x_np.ne.xidd.or.y_np.ne.yidd.or.z_np.ne.zidd) goto 320
          tval0=tval0*10.d0
          xidd=x_np+tval0*u_np
          yidd=y_np+tval0*v_np
          zidd=z_np+tval0*w_np
          go to 310
  320   continue
c       write(*,*) 'srzone:1'
        call srzone(xidd,yidd,zidd,iq_np,ir_np,irnext)
c
        if(irnext.ne.ir_np) then
          tval=0.0d0
          irnear=irnext
        else
          tval00=0.0d0
          tval10=10.0d0*tval0
          irlold=ir_np
          irlfg=0
  330     continue
          if(irlfg.eq.1) go to 340
            tval00=tval00+tval10
            if(tval00.gt.1.0d+06) then
              write(6,9000) iq(np),ir(np),x(np),y(np),z(np),
     &                      u(np),v(np),w(np),tval00
 9000 format(' TVAL00 ERROR : iq,ir,x,y,z,u,v,w,tval=',
     &       2I3,1P7E12.5)
              stop
            end if
            xidd=x_np+tval00*u_np
            yidd=y_np+tval00*v_np
            zidd=z_np+tval00*w_np
            call srzold(xidd,yidd,zidd,irlold,irlfg)
            go to 330
  340     continue
c
          tval=tval00
          do j=1,10
            xidd=x_np+tval00*u_np
            yidd=y_np+tval00*v_np
            zidd=z_np+tval00*w_np
c           write(*,*) 'srzone:2'
            call srzone(xidd,yidd,zidd,iq_np,irlold,irnext)
            if(irnext.ne.irlold) then
              tval=tval00
              irnear=irnext
            end if
            tval00=tval00-tval0
          end do
          if(ir_np.eq.irnear) then
            write(0,*) 'ir(np),tval=',ir_np,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
            endif
          enddo
        enddo
        itvlfg=0
        tvalmn=tval
        do jjj=1,itvalm
          if(tvalmn.gt.atval(jjj)) then
            tvalmn=atval(jjj)
          end if
          delhow=cgeps2
          tval0=atval(jjj)+delhow
          xidd=x_np+tval0*u_np
          yidd=y_np+tval0*v_np
```

```
          zidd=z_np+tval0*w_np
  410     continue
          if(x_np.ne.xidd.or.y_np.ne.yidd.or.z_np.ne.zidd) go to 420
            delhow=delhow*10.d0
            tval0=atval(jjj)+delhow
            xidd=x_np+tval0*u_np
            yidd=y_np+tval0*v_np
            zidd=z_np+tval0*w_np
          go to 410
  420     continue
c         write(*,*) 'srzone:3'
          call srzone(xidd,yidd,zidd,iq_np,ir_np,irnext)
          if((irnext.ne.ir_np.or.atval(jjj).ge.1.).and.
     &        tval.gt.atval(jjj)) THEN
            tval=atval(jjj)
            irnear=irnext
            itvlfg=1
            goto 425
          end if
        end do
  425   continue
        if(itvlfg.eq.0) then
          tval0=cgmnst
          xidd=x_np+tval0*u_np
          yidd=y_np+tval0*v_np
          zidd=z_np+tval0*w_np
  430     continue
          if(x_np.ne.xidd.or.y_np.ne.yidd.or.z_np.ne.zidd) go to 440
            tval0=tval0*10.d0
            xidd=x_np+tval0*u_np
            yidd=y_np+tval0*v_np
            zidd=z_np+tval0*w_np
            go to 430
  440     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(6,9200) iq(np),ir(np),x(np),y(np),z(np),
     &                  u(np),v(np),w(np),tval
 9200 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
        if(irnew.ne.ir_np) then
          call rstnxt(iq_np,ir_np,irnew)
        endif
      end if
      return
      end
!--------------------last line of subroutine howfar----------------------
```