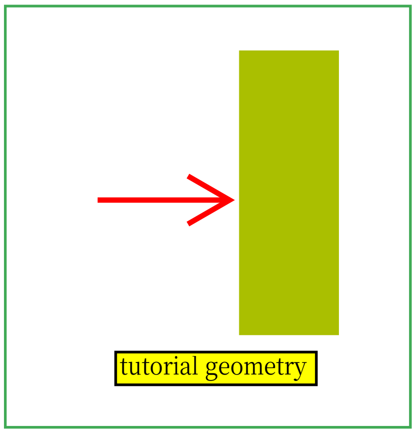


Chapter 3

A SERIES OF SHORT EGS5 TUTORIALS

EGS is a powerful system which can be used to produce very complex Monte Carlo simulations. In spite of some complexity, the user's interface with the system is, in principle, very simple. In the following series of tutorial programs, we use various aspects of the user interface in what we refer to as "EGS5user codes." In these user codes we will introduce some basic scoring techniques. Formal documentation in the form of EGS5 and PEGS user manuals can be found in Appendices B and C, respectively.

These tutorials are written under the assumption that the reader is generally familiar with the contents of the EGS5 and PEGS user manuals, although a complete understanding of the manuals is not required. In fact, the purpose of these tutorials is to make these manuals more understandable. Although the programs presented here are very simple in construction, it should become clear that with various extensions (generally of a bookkeeping nature), a wide range of powerful programs can be constructed from these tutorial examples. For brevity, we sometimes present only partial source listings of these user codes in the following sections. The complete source code for each tutorial can be found in the EGS5 distribution. Note also that the results from these tutorial programs may be slightly different on machines with different word lengths, different floating-point hardware, or different compiler optimizations.

3.1 Tutorial 1 (Program tutor1.f)

The geometry of the first seven tutorials is the same. Namely, a semi-infinite slab of material is placed in a vacuum and a pencil beam of photons or electrons is incident normal to the surface. The slab is in the X-Y plane and the particles are incident at the origin traveling along the Z-axis. In the first problem, a beam of 20 MeV electrons is incident on a 1 mm thick plate of tantalum.

In order to use EGS5 to answer the question “What comes out the far side of the plate?”, we have created the user code (**tutor1.f**) shown below. Also provided is the PEGS5 input file required for this run (see Appendix C for a description of how to construct PEGS5 input files).

```
!***********************************************************************
!
!                       **************
!                       *            *
!                       *  tutor1.f  *
!                       *            *
!                       **************
!
!  An EGS5 user code. It lists the particles escaping from the back
!  of a 1 mm Ta plate when a pencil beam of  20 MeV electrons
!  is incident on it normally.
!
!  For SLAC-R-730/KEK Report 2005-8: A simple example which 'scores'
!  by listing particles
!
!  The following units are used: unit 6 for output
!***********************************************************************
!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_media.f'
      include 'include/egs5_misc.f'
      include 'include/egs5_thresh.f'
      include 'include/egs5_useful.f'
      include 'include/egs5_usersc.f'
      include 'include/randomm.f'

!     bounds contains ecut and pcut
!     media contains the array media
!     misc contains med
!     thresh contains ae and ap
!     useful contains RM
!     usersc contains estepe and estepe2

      common/geom/zbound
      real*8 zbound
!     geom passes info to our howfar routine

      real*8 ein,xin,yin,zin,                        ! Arguments
     *       uin,vin,win,wtin
      integer iqin,irin

      integer i,j                                    ! Local variables
      character*24 medarr(1)

!     ----------
!     Open files
!     ----------
      open(UNIT= 6,FILE='egs5job.out',STATUS='unknown')

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

!-----------------------------------------------------------------------
! Step 2: pegs5-call
!-----------------------------------------------------------------------
!     ==============
      call block_set                 ! Initialize some general variables
!     ==============

!     --------------------------------
!     define media before calling PEGS5
!     --------------------------------
      nmed=1
      medarr(1)='TA                      '

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

! nmed and dunit default to 1, i.e. one medium and we work in cm

      chard(1) = 0.1d0       !  optional, but recommended to invoke
                             !  automatic step-size control

!     ---------------------------------------------
!     Run KEK version of PEGS5 before calling HATCH
!     (method was developed by Y. Namito - 010306)
!     ---------------------------------------------
```

```
      write(6,100)
100   FORMAT(' PEGS5-call comes next'/)

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

!-----------------------------------------------------------------------
! Step 3: Pre-hatch-call-initialization
!-----------------------------------------------------------------------
      nreg=3
!     nreg : number of region

      med(1)=0
      med(3)=0
      med(2)=1
! Vacuum in regions 1 and 3, ta in region 2
      ecut(2)=1.5
! Terminate electron histories at 1.5 MeV in the plate
      pcut(2)=0.1
! Terminate   photon histories at 0.1 MeV in the plate
!             Only needed for region 2 since no transport elsewhere
!             ecut is total energy = 0.989   MeV kinetic energy

!     --------------------------------
!     Set parameter estepe and estepe2
!     --------------------------------
      ~~estepe=0.10~~
      ~~estepe2=0.20~~
      ~~write(6,110) estepe, estepe2~~
~~110   FORMAT(1X,'ESTEPE at EKMAX: ',F10.5,' (estepe)',~~
     ~~*       /,1X,'ESTEPE at ECUT:  ',F10.5,' (estepe2)')~~

!     ------------------------------------------------------
!     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 20 MeV beam of electrons incident
! perpendicular to the slab
      iqin=-1
!            Incident charge - electrons
!            20 MeV kinetic energy
      ein=20.511
      xin=0.0
      yin=0.0
      zin=0.0
!     Incident at origin
      uin=0.0
      vin=0.0
      win=1.0
!            Moving along z axis
      irin=2
!            Starts in region 2, could be 1
!            weight = 1 since no variance reduction used
      wtin=1.0
!     Weight = 1 since no variance reduction used

!-----------------------------------------------------------------------
! Step 5:   hatch-call
!-----------------------------------------------------------------------
! Maximum total energy of an electron for this problem must be
! defined before hatch call
      emaxe = ein

      write(6,130)
130   format(/' Start tutor1'/' 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)

!    Pick up cross section data for ta
      write(6,150) ae(1)-0.511, ap(1)
150   format(/' Knock-on electrons can be created and any electron ',
     *'followed down to' /T40,F8.3,' MeV kinetic energy'/
     *' Brem photons can be created and any photon followed down to',
     */T40,F8.3,' MeV')
! Compton events can create electrons and photons below these cutoffs

!-----------------------------------------------------------------------
! Step 6:  Initialization-for-howfar
!-----------------------------------------------------------------------
      zbound=0.1
!     plate is 1 mm thick

!-----------------------------------------------------------------------
! Step 7:  Initialization-for-ausgab
!-----------------------------------------------------------------------
! Print header for output - which is all ausgab does in this case
      write(6,160)
160   format(/T19,'Kinetic energy(MeV)',T40,'charge',T48,
     *'angle w.r.t. z axis-degrees')

!-----------------------------------------------------------------------
! Step 8:  Shower-call
!-----------------------------------------------------------------------
! Initiate the shower 10 times
      do i=1,10
        write(6,170) i
170     format(' Start history',I4)
        call shower(iqin,ein,xin,yin,zin,uin,vin,win,irin,wtin)
!-----------------------------------------------------------------------
! Step 9:  Output-of-results
!-----------------------------------------------------------------------
!  Note output is at the end of each history in subroutine ausgab
      end do
      stop
      end
!-------------------------last line of main code------------------------

!-------------------------------ausgab.f--------------------------------
!-----------------------------------------------------------------------
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12
! ----------------------------------------------------------------------
! Required subroutine for use with the EGS5 Code System
! ----------------------------------------------------------------------
!***********************************************************************
!
!  In general, ausgab is a routine which is called under a series
!  of well defined conditions specified by the value of iarg (see the
!  egs5 manual for the list).  This is a particularly simple ausgab.
!  Whenever this routine is called with iarg=3 , a particle has
!  been discarded by the user in howfar
!  we get ausgab to print the required information at that point
!
```

```
!***********************************************************************
      subroutine ausgab(iarg)

      implicit none

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

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

      integer iarg                                            ! Arguments

      real*8 angle,ekine                          ! Local variables

      if (iarg.eq.3) then
!  Angle w.r.t. z axis in degrees
        angle=acos(w(np))*180./3.14159
        if (iq(np).eq.0) then
          ekine=e(np)
        else
          ekine=e(np)-0.511
!  Get kinetic energy
        end if
        write(6,100) ekine,iq(np),angle
100     format(T21,F10.3,T33,I10,T49,F10.1)
      end if
      return
      end

!--------------------------last line of ausgab.f------------------------

!-------------------------------howfar.f--------------------------------
!-----------------------------------------------------------------------
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12
! ----------------------------------------------------------------------
! Required (geometry) subroutine for use with the EGS5 Code System
!***********************************************************************
!
! The following is a general specification of howfar
!   given a particle at (x,y,z) in region ir and going in direction
!   (u,v,w), this routine answers the question, can the particle go
!   a distance ustep without crossing a boundary
!           If yes, it merely returns
!           If no, it sets ustep=distance to boundary in the current
!           direction and sets irnew to the region number   on the
!           far side of the boundary (this can be messy in general!)
!
!   The user can terminate a history by setting idisc>0. here we
!   terminate all histories which enter region 3 or are going
!   backwards in region 1
!
!                   |                |
!   Region 1        |   Region 2    |       Region 3
!                   |               |
!   e- =========>   |               | e- or photon ====>
!                   |               |
!   vacuum          |     Ta        |       vacuum
!
!***********************************************************************
      subroutine howfar

      implicit none

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

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

      common/geom/zbound
      real*8 zbound
!     geom passes info to our howfar routine

      real*8 tval                              ! Local variable

      if (ir(np).eq.3) then
        idisc=1
        return
!  Terminate this history: it is past the plate
!  We are in the Ta plate - check the geometry
      else if (ir(np).eq.2) then
        if (w(np).gt.0.0) then
!  Going forward - consider first since  most frequent
!  tval is dist to boundary in this direction
          tval=(zbound-z(np))/w(np)
          if (tval.gt.ustep) then
            return
!  Can take currently requested step
          else
            ustep=tval
            irnew=3
            return
          end if
!    end of w(np)>0 case
!    Going back towards origin
        else if (w(np).lt.0.0) then
!    Distance to plane at origin
          tval=-z(np)/w(np)
          if (tval.gt.ustep) then
            return
!    Can take currently requested step
          else
            ustep=tval
            irnew=1
            return
          end if
!    End w(np)<0 case
!    Cannot hit boundary
        else if (w(np).eq.0.0) then
          return
        end if
!  End of region 2 case
!  In regon with source
!  This must be a source particle on z=0 boundary
      else if (ir(np).eq.1) then
        if (w(np).gt.0.0) then
          ustep=0.0
          irnew=2
          return
        else
!  It must be a reflected particle-discard it
          idisc=1
          return
        end if
!  End region 1 case
      end if
      end
!--------------------------last line of howfar.f------------------------
```

```
ELEM
 &INP EFRACH=0.05,EFRACL=0.20,
      IRAYL=0,IBOUND=0,INCOH=0,ICPROF=0,IMPACT=0 &END
TA                             TA
TA
ENER
 &INP AE=1.50,AP=0.10,UE=20.611,UP=20.0 &END
TEST
 &INP  &END
PWLF
 &INP  &END
DECK
 &INP  &END
```

This user code produces the following output file called **egs5job.out6** (a copy of this file is included with the EGS5 distribion, named **tutor1.out**).

```
 PEGS5-call comes next

 ESTEPE at EKMAX:    0.10000 (estepe)
 ESTEPE at ECUT:     0.20000 (estepe2)

 inseed=           1      (seed for generating unique sequences of Ranlux)
```

```
ranlux luxury level set by rluxgo : 1     p=  48
ranlux initialized by rluxgo from seed           1

Start tutor1
Call hatch to get cross-section data

HATCH-call comes next

EGS SUCCESSFULLY 'HATCHED' FOR ONE MEDIUM.

Knock-on electrons can be created and any electron followed down to
                                         0.989 MeV kinetic energy
Brem photons can be created and any photon followed down to
                                         0.100 MeV
```

```
                 Kinetic energy(MeV)  charge  angle w.r.t. z axis-degrees
Start history   1
                        3.125           0            1.4
                        0.172           0            1.7
                       14.779          -1           19.9
Start history   2
                        5.841           0           39.7
                        0.337           0           40.2
                       11.707          -1           33.3
Start history   3
                        3.507           0           21.6
                       14.378          -1           58.6
Start history   4
                        2.446           0            1.4
                       15.503          -1           11.7
Start history   5
                        0.341           0            1.4
                       17.119          -1           31.9
Start history   6
                       17.843          -1           31.1
Start history   7
                        1.480           0           30.1
                       16.508          -1           30.1
Start history   8
                        0.188           0           34.7
                       17.381          -1           33.9
Start history   9
                        0.874           0            1.4
                        1.409           0            1.6
                        0.496           0           64.8
                       13.599          -1           64.5
Start history  10
                       17.440          -1           69.2
```

By keeping track of many of these histories, we could answer a lot of questions about what comes out the far side of the plate, but it should be recognized that these are all bookkeeping extensions to the problem – the physics itself already accomplished with EGS5 and the relatively small amount of user code listed above. The scoring routine for this problem is the simplest possible; namely, it outputs on the file some of the parameters of the various particles leaving the plate.

In addition, this user code includes examples of the following items that are discussed in detail in the EGS5 User Manual (Appendix B).

- The use of include statements to use values defined by parameter statements and to allow easy insertion of `COMMONS`.
- The technique required in order to define the array `MEDIA`.
- The definition of calling `PEGS5` to produce material data used by user code.
- The definition of seeds for the RANLUX random number generator.
- The definition of calling parameters for the `SHOWER` routine.
- A very simple `AUSGAB` routine.
- A simple `HOWFAR` routine.

3.2 Tutorial 2 (Program tutor2.f)

In this example we use the same geometry as above, but we want the fraction of the incident energy that is reflected from, transmitted through, and deposited in the plate. The coding is essentially the same as tutor1 except that `COMMON/SCORE/` and a new array `ENCORE` are defined at Step 1 in the sequence of steps required in the construction of a user code `MAIN` program, as described in the EGS5 User Manual of Appendix B. The latter is initialized to zero (Step 7) and subsequently output on the file (Step 9). The `AUSGAB` routine is considerably different as shown below.

```
!-------------------------------ausgab.f-------------------------------
!-----------------------------------------------------------------------
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12
! ---------------------------------------------------------------------
! Required subroutine for use with the EGS5 Code System
! ---------------------------------------------------------------------
!***********************************************************************
!
! In this AUSGAB routine for TUTOR2, we score the energy deposited
!  in the various regions. This amounts to the total energy
!  reflected, deposited and transmitted by the slab.
!
```

```
!  For IARG=0, an electron or photon step is about to occur and we
!  score the energy deposited, if any. Note that only electrons
!  deposit energy during a step, and due to our geometry, electrons
!  only take steps in region 2 - however there is no need to check.
!  For IARG=1,2 and 4, particles have been discarded for falling
!  below various energy cutoffs and all their energy is deposited
!  locally (in fact EDEP = particles kinetic energy).
!  For IARG=3, we are discarding the particle since it is in
!  region 1 or 3, so score its energy.
!
!***********************************************************************
      subroutine ausgab(iarg)

      implicit none

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

      include 'include/egs5_epcont.f'    ! epcont contains edep
      include 'include/egs5_stack.f'     ! stack contains x, y, z, u, v,
                                         ! w, ir and np

      common/score/escore(3)
      real*8 escore

      integer iarg                                          ! Arguments

      integer irl                                     ! Local variables

      if (iarg.le.4) then
        irl=ir(np)
!   Pick up current region number
        escore(irl)=escore(irl)+edep
      end if
      return
      end
!--------------------------last line of ausgab.f------------------------
```

AUSGAB is still very simple since all we need to do is to keep track of the energy deposited in the three regions. The variable EDEP (available through COMMON/EPCONT/) contains the energy deposited during a particular step for a variety of different IARG-situations, as described in the comments above and further elaborated upon in Appendix B. In this example, but not always, we can sum EDEP for any value of IARG up to 4. The following is the output provided by **tutor2** (named tutor2.out in distribution file).

```
 PEGS5-call comes next

 ESTEPE at EKMAX:    0.10000 (estepe)
 ESTEPE at ECUT:     0.20000 (estepe2)
```

```
inseed=           1      (seed for generating unique sequences of Ranlux)
ranlux luxury level set by rluxgo : 1     p=  48
ranlux initialized by rluxgo from seed           1

Start tutor2
Call hatch to get cross-section data

HATCH-call comes next

EGS SUCCESSFULLY 'HATCHED' FOR ONE MEDIUM.

Knock-on electrons can be created and any electron followed down to
                                         0.989 MeV kinetic energy
Brem photons can be created and any photon followed down to
                                         0.100 MeV

Fraction of energy reflected from plate=            0.857%
Fraction of energy deposited in plate=             12.622%
Fraction of energy transmitted through plate=      86.521%
                                                -----------
Total fraction of energy accounted for=           100.000%
```

3.3 Tutorial 3 (Program tutor3.f)

The geometry in this example is similar to the previous two but the problem is very different. Here we investigate the energy response function for a 2.54 cm thick slab of NaI when a 5 MeV beam of photons is incident on it. In this case the final scoring and binning is done at the end of each history (*i.e.*, after all the descendants from each initial photon have been tracked completely). The following shows the change required (at Step 8 and 9) and the new `AUSGAB` routine.

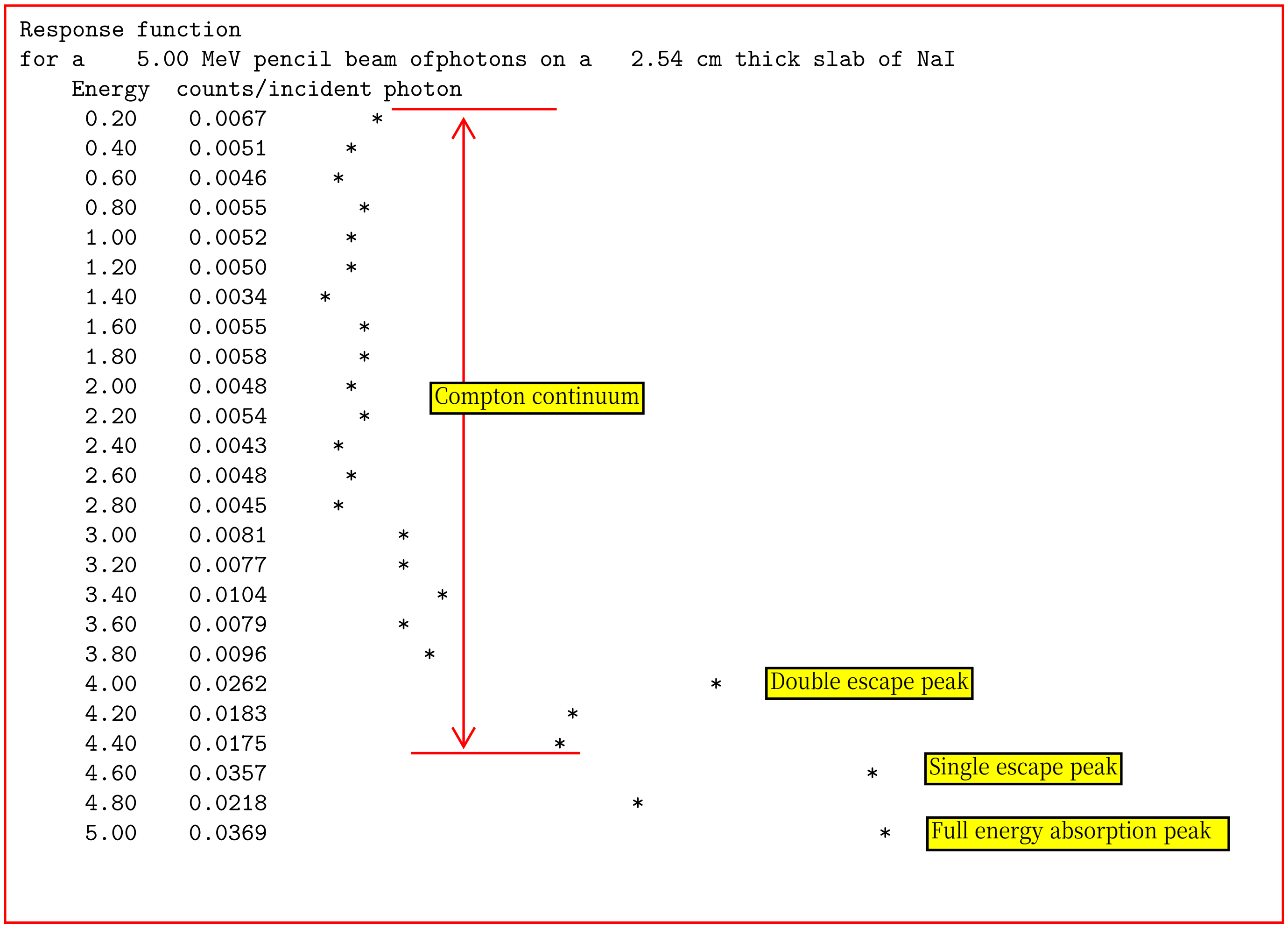
```
!-----------------------------------------------------------------------
! Step 8:  Shower-call
!-----------------------------------------------------------------------
! Initiate the shower ncase times
      ncase=10000
      do i=1,ncase
        ehist = 0.0
!  Zero energy deposited in this history
        call shower(iqin,ein,xin,yin,zin,uin,vin,win,irin,wtin)
!  Increment bin corresponding to  energy deposited in this history
        ibin= min0 (int(ehist/bwidth + 0.999), 25)
        if (ibin.ne.0) then
          ebin(ibin)=ebin(ibin)+1
        end if
      end do
```

```
!-----------------------------------------------------------------------
! Step 9:  Output-of-results
!-----------------------------------------------------------------------
! Pick up maximum bin for normalization
      binmax=0.0
      do j=1,25
        binmax=max(binmax,ebin(j))
      end do
      write(6,160) ein,zbound
160   format(/' Response function'/' for a',F8.2,' MeV pencil beam of',
     *'photons on a',F7.2,' cm thick slab of NaI'/ T6,
     *'Energy  counts/incident photon')
      do j=1,48
        line(j)=' '
      end do
! Blank entire output array
      do j=1,25
        icol=int(ebin(j)/binmax*48.0+0.999)
        if (icol.eq.0) icol=1
        line(icol)='*'
!  Load output array at desired location
        write(6,170) bwidth*j,ebin(j)/float(ncase),line
170     format(F10.2,F10.4,48A1)
        line(icol)=' '
!  Reblank
      end do

      stop
      end
!-------------------------last line of main code------------------------

!-------------------------------ausgab.f--------------------------------
!-----------------------------------------------------------------------
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12
! ----------------------------------------------------------------------
! Required subroutine for use with the EGS5 Code System
! ----------------------------------------------------------------------
!***********************************************************************
!
! In this AUSGAB routine for TUTOR3, we score the energy deposited
! in the detector region, region 2
!
!  For IARG=0, an electron or photon step is about to occur and we
!  score the energy deposited, if any. Note that only electrons
!  deposit energy during a step, and due to our geometry, electrons
!  only take steps in region 2 - however there is no need to check
!   this here
!  For IARG=1,2 and 4,particles have been discarded for falling below
!  various energy cutoffs and all their energy is deposited locally
!  (in fact EDEP = particles kinetic energy). This only happens in
!  region 2.  For IARG=3, we are discarding the particle since it is
!   in region 1 or 3, so we do not score its energy
!
!  EHIST keeps track of the total energy deposited during each
!  history. In the main routine it is zeroed at the start of each
!  history and binned at the end of each history.
!**********************************************************************
      subroutine ausgab(iarg)

      implicit none

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

      include 'include/egs5_epcont.f'    ! epcont contains edep
      include 'include/egs5_stack.f'     ! stack contains x, y, z, u, v,
                                         ! w, ir and np

      common/score/ehist
      real*8 ehist

      integer iarg                                         ! Arguments

      if (iarg.le.2 .or. iarg.eq.4) then
        ehist=ehist + edep
      end if

      return
      end

!--------------------------last line of ausgab.f------------------------
```

The following is the output provided by tutor3 (named tutor3.out in distribution file).

```
 PEGS5-call comes next

 ESTEPE at EKMAX:    0.10000 (estepe)
 ESTEPE at ECUT:     0.20000 (estepe2)

 inseed=           1      (seed for generating unique sequences of Ranlux)
 ranlux luxury level set by rluxgo : 1     p=  48
 ranlux initialized by rluxgo from seed           1

 Start tutor3
 Call hatch to get cross-section data

 HATCH-call comes next

 EGS SUCCESSFULLY 'HATCHED' FOR ONE MEDIUM.
```

```
Knock-on electrons can be created and any electron followed down to
                                         0.189 MeV kinetic energy
Brem photons can be created and any photon followed down to
                                         0.010 MeV
```



```
Response function
for a    5.00 MeV pencil beam ofphotons on a   2.54 cm thick slab of NaI
    Energy  counts/incident photon
     0.20    0.0067        *
     0.40    0.0051      *
     0.60    0.0046     *
     0.80    0.0055       *
     1.00    0.0052      *
     1.20    0.0050      *
     1.40    0.0034    *
     1.60    0.0055       *
     1.80    0.0058       *
     2.00    0.0048      *
     2.20    0.0054       *
     2.40    0.0043     *
     2.60    0.0048      *
     2.80    0.0045     *
     3.00    0.0081          *
     3.20    0.0077          *
     3.40    0.0104             *
     3.60    0.0079          *
     3.80    0.0096            *
     4.00    0.0262                                  *
     4.20    0.0183                       *
     4.40    0.0175                      *
     4.60    0.0357                                              *
     4.80    0.0218                            *
     5.00    0.0369                                               *
```

3.4 Tutorial 4 (Program tutor4.f)

This program examines the dependence of EGS5 results on electron step-size. Recall that for electrons with low initial energies, the limitations inherent in the EGS4 transport mechanics mandated that the user specify quite small electron step-sizes (defined in terms of the fractional energy loss over a step, `ESTEPE`) in order to assure converged results. As noted in Chapter 2, the transport mechanics algorithm of EGS5 naturally mitigates these dependencies. In addition, EGS5 provides several prescriptions for user selection of step-sizes based on material and region geometries.

As in EGS4, the program **tutor4.f** is based on **tutor2.f**, but with a 2 mm slab of silicon as the medium and 2.0 MeV for the incident electron energy. In addition to scoring transmitted, deposited and reflected energy, the number of transmitted and reflected electrons are tallied in **tutor4.f**. The example problem actually consists of four distinct runs, two using step-size specification based on fractional energy loss (EFRACH and EFRACL) and two using a user specified “characteristic dimension” for the given medium.

Recall that multiple-scattering step-sizes in EGS5 are defined in terms of the scattering strength accumulated over the given distance, and can be specified by the parameters EFRACH and EFRACL, which correspond to the fractional energy loss over the step at the upper energy range of the problem UE and the lower energy limit AE, respectively. Thus EFRACH and EFRACL, which are set in the PEGS input file, **pgs5job.pegs5inp**, correspond to ESTEPE from EGS4. In the first **tutor4.f** example, the material in the slab is named “SI with long steps” and EFRACH and EFRACL are set to 0.30. It should be noted that these values were chosen to be artificially high (the defaults in EGS5 are 0.05 and 0.20, respectively) to help illustrate step-size dependence, since on this problem, EGS5 shows little dependence on step-size with the default values). In the second example, the material is switched to one named “SI with short steps,” and EFRACH and EFRACL are set to be 0.01 and 0.02, respectively. (Thus this example problem also illustrates a method for creating different “media” which are actually the same material with different physics options invoked.)

In the third pass through **tutor4.f**, the first material is again used, but the step-size is selected by the specification of a “characteristic dimension,” CHARD. When CHARD is positive, EGS5 ignores step-sizes which correspond to EFRACH and EFRACL and instead automatically chooses values of the initial scattering strength which provide converged values for electron tracklength (to within 1-2% accuracy) for electrons impinging on an semi-infinite cylinder of diameter CHARD. The final example in **tutor4.f** demonstrates the mechanism for specifying smaller steps by specifying region dependent scale factors K1HSCL and K1LSCL, if greater than 1-2% accuracy is required.

Following is the source code (except for subroutine HOWFAR, which is not changed from **tutor2.f**) used by **tutor4.f**.

```
!***********************************************************************
!
!                       **************
!                       *            *
!                       *  tutor4.f  *
!                       *            *
!                       **************
!
!  An EGS5 user code. It lists the particles escaping from the back
!  of a 2 mm Si plate when a pencil beam of 2 MeV electrons
!  is incident on it normally.
!
!  For SLAC-R-730/KEK Report 2005-8: A simple example which scores
!  reflected, deposited, and transmitted particles and energy and
!  demonstrates step-size selection
!
!  The following units are used: unit 6 for output
!***********************************************************************
!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_media.f'
      include 'include/egs5_misc.f'
      include 'include/egs5_thresh.f'
      include 'include/egs5_useful.f'
      include 'include/egs5_usersc.f'
      include 'include/randomm.f'

!     bounds contains ecut and pcut
!     media contains the array media
!     misc contains med
!     thresh contains ae and ap
!     useful contains RM
!     usersc contains estepe and estepe2

      common/geom/zbound
      real*8 zbound
!     geom passes info to our howfar routine

      common/score/escore(3),iscore(3)
      real*8 escore
      integer iscore

      real*8 ein,xin,yin,zin,                          ! Arguments
     *       uin,vin,win,wtin
      integer iqin,irin

      real*8 anorm,total                        ! Local variables
      real
     * tarray(2),tt,tt0,tt1,cputime
      integer loop,i,j,ncase
      character*24 medarr(2)

      real etime

!     ----------
!     Open files
!     ----------
      open(UNIT= 6,FILE='egs5job.out',STATUS='unknown')

      do loop = 1,4

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

!-----------------------------------------------------------------------
! Step 2: pegs5-call
!-----------------------------------------------------------------------
!     ==============
      call block_set                 ! Initialize some general variables
!     ==============

! nmed and dunit default to 1, i.e. one medium and we work in cm

      if(loop.eq.3) then
        chard(1) = 0.20d0      !  optional, but recommended to invoke
        chard(2) = 0.20d0      !  automatic step-size control
      else if(loop.eq.4) then
        chard(1) = 0.20d0      !  optional, but recommended to invoke
        chard(2) = 0.20d0      !  automatic step-size control
        k1Hscl(2) = 0.25d0
        k1Lscl(2) = 0.25d0      !  to reduce step size in a region
      else
        chard(1) = 0.00d0      !  optional, but recommended to invoke
        chard(2) = 0.00d0      !  automatic step-size control
      endif

      write(6,100) loop, chard(1)
100   FORMAT(72('*'),/,
     *'Initializing EGS5, loop = ',I1,': charD = ',f5.2,/,
     *72('*'),/)

      if(loop.eq.1) then

!     --------------------------------
!     define media before calling PEGS5
!     --------------------------------
      nmed=2
      medarr(1)='SI with long steps      '
      medarr(2)='SI with short steps     '

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

```
!     --------------------------------------------
!     Run KEK version of PEGS5 before calling HATCH
!     (method was developed by Y. Namito - 010306)
!     --------------------------------------------

      write(6,110)
110   FORMAT(' PEGS5-call comes next'/)

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

      endif

      if(loop.lt.3) then
        write(6,120) loop,medarr(loop)
120   FORMAT(' Using media number ',i1,', ',a24,' for this run',/)
      else if(loop.eq.4) then
        write(6,130) k1Hscl(2),k1Lscl(2)
130   FORMAT(' Scaling step-sizes by ',F4.2,' and ',F4.2,' at upper ',
     * ' and lower energy limits',/)
      endif

!-----------------------------------------------------------------------
! Step 3: Pre-hatch-call-initialization
!-----------------------------------------------------------------------
      nreg=3
!     nreg : number of region

      med(1)=0
      med(3)=0
      if(loop.eq.2) then
        med(2)=2
      else
        med(2)=1
      endif
! Vacuum in regions 1 and 3, Si in region 2
      ecut(2)=0.700
! Terminate electron histories at .700 MeV in the plate
      pcut(2)=0.010
! Terminate   photon histories at 0.01 MeV in the plate
!             Only needed for region 2 since no transport elsewhere
!             ecut is total energy = 0.189   MeV kinetic energy

!     -------------------------------
!     Set parameter estepe and estepe2
!     -------------------------------
      estepe=0.01
      estepe2=0.05
      write(6,140) estepe, estepe2
140   FORMAT(1X,'ESTEPE at EKMAX: ',F10.5,' (estepe)',
    *       /,1X,'ESTEPE at ECUT:  ',F10.5,' (estepe2)')

!     ------------------------------------------------------
!     Random number seeds.  Must be defined before call hatch
!     or defaults will be used.  inseed (1- 2^31)
!     ------------------------------------------------------
      luxlev=1
      inseed=1
      kount=0
      mkount=0
      do i = 1, 25
        isdext(i) = 0
      end do
      write(6,150) inseed
150   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 2 MeV beam of electrons incident
! perpendicular to the slab
      iqin=-1
!            Incident charge - electrons
!            2 MeV kinetic energy
      ein=2.d0 + RM
      xin=0.0
      yin=0.0
      zin=0.0
!     Incident at origin
      uin=0.0
      vin=0.0
      win=1.0
!            Moving along z axis
      irin=2
!            Starts in region 2, could be 1
!            weight = 1 since no variance reduction used
      wtin=1.0
!     Weight = 1 since no variance reduction used

!-----------------------------------------------------------------------
! Step 5:   hatch-call
!-----------------------------------------------------------------------
! Maximum total energy of an electron for this problem must be
! defined before hatch call
      emaxe = ein
      write(6,160)
160   FORMAT(/' Start tutor4'/' 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(6,170)
170   FORMAT(/,' HATCH-call comes next',/)

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

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

!    Pick up cross section data for ta
      write(6,180) ae(1)-RM, ap(1)
180   FORMAT(/' Knock-on electrons can be created and any electron ',
     *'followed down to' /T40,F8.3,' MeV kinetic energy'/
     *' Brem photons can be created and any photon followed down to',
     */T40,F8.3,' MeV')
! Compton events can create electrons and photons below these cutoffs

!-----------------------------------------------------------------------
! Step 6:  Initialization-for-howfar
!-----------------------------------------------------------------------
      zbound=0.2
!     plate is 2 mm thick

!-----------------------------------------------------------------------
! Step 7:  Initialization-for-ausgab
!-----------------------------------------------------------------------
      do i=1,3
        iscore(i)=0
        escore(i)=0.d0
!  Zero scoring array before starting
      end do

!-----------------------------------------------------------------------
! Step 8:  Shower-call
!-----------------------------------------------------------------------
      tt=etime(tarray)
      tt0=tarray(1)
```

```
! Initiate the shower ncase times
      ncase=50000
      do i=1,ncase
        call shower(iqin,ein,xin,yin,zin,uin,vin,win,irin,wtin)
      end do

      tt=etime(tarray)
      tt1=tarray(1)
      cputime=tt1-tt0

!-----------------------------------------------------------------------
! Step 9:  Output-of-results
!-----------------------------------------------------------------------
      write(6,190) cputime,ncase
190   FORMAT('CPU time = ',1X,G15.5,' sec for ',I8,' cases')

      anorm = 100./float(ncase)
      write(6,200) iscore(1)*anorm,iscore(3)*anorm
200   FORMAT(/,
     *' Fraction of electrons reflected from plate=',T50,F10.1,'%',/,
     *' Fraction of electrons transmitted through plate=',T50,F10.1,'%')

! Normalize to % of total input energy
      anorm = 100./((ein-RM)*float(ncase))
      total=0.0
      do i=1,3
        total=total+escore(i)
      end do
      write(6,210) (escore(i)*anorm,i=1,3),total*anorm
210   FORMAT(/,/,
     *  ' Fraction of energy reflected from plate=',T50,F10.1,'%'
     */ ' Fraction of energy deposited in plate=',T50,F10.1,'%'/
     *' Fraction of energy transmitted through plate=',T50,F10.1,'%'/
     *T50,11('-')/' Total fraction of energy accounted for=', T50,
     *F10.1,'%'/)

      end do  ! do four times through

      stop
      end
!-------------------------last line of main code------------------------

!-------------------------------ausgab.f--------------------------------
!-----------------------------------------------------------------------
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12
! ----------------------------------------------------------------------
! Required subroutine for use with the EGS5 Code System
! ----------------------------------------------------------------------
!***********************************************************************
!
! In this AUSGAB routine for TUTOR4, we score the energy deposited
!  in the various regions and count transmitted and reflected
!  electrons.
!
!  For IARG=0, an electron or photon step is about to occur and we
!  score the energy deposited, if any. Note that only electrons
!  deposit energy during a step, and due to our geometry, electrons
!  only take steps in region 2 - however there is no need to check.
!  For IARG=1,2 and 4, particles have been discarded for falling
!  below various energy cutoffs and all their energy is deposited
!  locally (in fact EDEP = particles kinetic energy).
!  For IARG=3, we are discarding the particle since it is in
!  region 1 or 3, so score its energy, and if it is an electron,
!  score it's region.
!
!***********************************************************************
      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_stack.f'

      common/score/escore(3), iscore(3)
      real*8 escore
      integer iscore

      integer iarg                                          ! Arguments

      integer irl                                     ! Local variables

      if (iarg.le.4) then
        irl=ir(np)
!   Pick up current region number
        escore(irl)=escore(irl)+edep
!   Pick up energy deposition/transmission/reflection
        if (iarg.eq.3 .and. iq(np).eq.-1) then
          iscore(irl)=iscore(irl)+1
!   Pick up electron transmission/reflection
        end if
      end if
      return
      end

!--------------------------last line of ausgab.f------------------------
```

Following is the PEGS input file required by **tutor4.f**, which specifies two versions of the same material (silicon), one with long step-sizes and one with short step-sizes.

```
ELEM
 &INP EFRACH=0.30,EFRACL=0.30,
      IRAYL=0,IBOUND=0,INCOH=0,ICPROF=0,IMPACT=0 &END
SI with long steps                                 SI
SI
ENER
 &INP AE=0.700,AP=0.010,UE=2.521,UP=2.1 &END
PWLF
 &INP  &END
DECK
 &INP  &END
ELEM
 &INP EFRACH=0.01,EFRACL=0.01,
      IRAYL=0,IBOUND=0,INCOH=0,ICPROF=0,IMPACT=0 &END
SI with short steps                                SI
SI
ENER
 &INP AE=0.700,AP=0.010,UE=2.521,UP=2.1 &END
PWLF
 &INP  &END
DECK
 &INP  &END
```

Following is the output produced by **tutor4.f**, showing that the severe step-size dependence exhibited by EGS4 on this problem is greatly diminished in EGS5. Recall that in the **tutor4** example of EGS4, runs using the default step-size algorithm predicted 1.3% reflection and 49.21.0%, EGS4 returned values of 6.4% reflection and 61.3% transmission. In contrast, EGS5 runs using very long multiple scattering steps (corresponding to 30% energy loss over the steps, loop 1 of the problem) yield values for reflection (8.3%) and transmission (66.6%) which are fairly close to EGS5 results using 1% energy loss (7.6% and 64.3% for reflection and transmission, respectively, loop 2 of the problem). This demonstrates the power of the modified random hinge approach. In addition, the significant discrepancies between the 1% energy loss results from EGS4 and EGS5 further illustrate the shortcomings of the EGS4 transport mechanics model, even at very small electron step sizes.

The results from the third loop of this example show that the expected 1-2are determined automatically by EGS5 using the characteristic dimension of 2 mm for this problem. Note that for this distance and at this energy, the step-sizes selected by EGS5 for silicon are roughly 15-20 times as large as those used in the 1% energy loss run (loop two), thus providing significant speedup (a factor of 4, as seen from the output) for calculations at this level of accuracy.

The results from loop four of this problem demonstrate that by using `K1HSCL` and `K1LSCL` (see Appendix B for details) to scale step-sizes, a higher degree of accuracy can be recovered without the complete loss of the efficiency provided by the characteristic dimension method.

```
***********************************************************************
Initializing EGS5, loop = 1: charD =  0.00
```

```
**********************************************************************

 PEGS5-call comes next

 Using media number 1, SI with long steps       for this run

 ESTEPE at EKMAX:    0.01000 (estepe)
 ESTEPE at ECUT:     0.05000 (estepe2)

 inseed=          1      (seed for generating unique sequences of Ranlux)
 ranlux luxury level set by rluxgo : 1     p=  48
 ranlux initialized by rluxgo from seed           1

 Start tutor4
 Call hatch to get cross-section data

 HATCH-call comes next

 EGS SUCCESSFULLY 'HATCHED' FOR     2 MEDIA.
 WARNING in RMSFIT: no characteristic dimension input for media   1
 Using old data from pgs5job.msfit with:
 efrach, efrachl =      3.0000E-01     3.0000E-01
 WARNING in RMSFIT: no characteristic dimension input for media   2
 Using old data from pgs5job.msfit with:
 efrach, efrachl =      1.0000E-02     1.0000E-02

 Knock-on electrons can be created and any electron followed down to
                                         0.189 MeV kinetic energy
 Brem photons can be created and any photon followed down to
                                         0.010 MeV
CPU time =       24.557     sec for    50000 cases

 Fraction of electrons reflected from plate=          8.3%
 Fraction of electrons transmitted through plate=     66.6%


 Fraction of energy reflected from plate=             4.0%
 Fraction of energy deposited in plate=              59.3%
 Fraction of energy transmitted through plate=       36.7%
                                                -----------
 Total fraction of energy accounted for=            100.0%

**********************************************************************
Initializing EGS5, loop = 2: charD =  0.00
**********************************************************************

 Using media number 2, SI with short steps      for this run

 ESTEPE at EKMAX:    0.01000 (estepe)
 ESTEPE at ECUT:     0.05000 (estepe2)
 inseed=           1      (seed for generating unique sequences of Ranlux)
 ranlux luxury level set by rluxgo : 1     p=  48
 ranlux initialized by rluxgo from seed           1

 Start tutor4
 Call hatch to get cross-section data

 HATCH-call comes next

 EGS SUCCESSFULLY 'HATCHED' FOR     2 MEDIA.
 WARNING in RMSFIT: no characteristic dimension input for media   1
 Using old data from pgs5job.msfit with:
 efrach, efrachl =      3.0000E-01     3.0000E-01
 WARNING in RMSFIT: no characteristic dimension input for media   2
 Using old data from pgs5job.msfit with:
 efrach, efrachl =      1.0000E-02     1.0000E-02

 Knock-on electrons can be created and any electron followed down to
                                         0.189 MeV kinetic energy
 Brem photons can be created and any photon followed down to
                                         0.010 MeV
CPU time =       130.60     sec for    50000 cases

 Fraction of electrons reflected from plate=          7.6%
 Fraction of electrons transmitted through plate=    64.3%


 Fraction of energy reflected from plate=             3.4%
 Fraction of energy deposited in plate=              62.1%
 Fraction of energy transmitted through plate=       34.5%
                                                -----------
 Total fraction of energy accounted for=            100.0%

************************************************************************
Initializing EGS5, loop = 3: charD =  0.20
************************************************************************

 ESTEPE at EKMAX:    0.01000 (estepe)
 ESTEPE at ECUT:     0.05000 (estepe2)

 inseed=           1      (seed for generating unique sequences of Ranlux)
 ranlux luxury level set by rluxgo : 1     p=  48
 ranlux initialized by rluxgo from seed           1

 Start tutor4
 Call hatch to get cross-section data

 HATCH-call comes next

 EGS SUCCESSFULLY 'HATCHED' FOR     2 MEDIA.
 Knock-on electrons can be created and any electron followed down to
                                         0.189 MeV kinetic energy
 Brem photons can be created and any photon followed down to
                                         0.010 MeV
CPU time =       28.263     sec for    50000 cases

 Fraction of electrons reflected from plate=            7.9%
 Fraction of electrons transmitted through plate=      65.1%


 Fraction of energy reflected from plate=               3.5%
 Fraction of energy deposited in plate=                61.1%
 Fraction of energy transmitted through plate=         35.3%
                                                 -----------
 Total fraction of energy accounted for=              100.0%

************************************************************************
Initializing EGS5, loop = 4: charD =  0.20
************************************************************************

 Scaling step-sizes by 0.25 and 0.25 at upper  and lower energy limits

 ESTEPE at EKMAX:    0.01000 (estepe)
 ESTEPE at ECUT:     0.05000 (estepe2)

 inseed=           1      (seed for generating unique sequences of Ranlux)
 ranlux luxury level set by rluxgo : 1     p=  48
 ranlux initialized by rluxgo from seed           1

 Start tutor4
 Call hatch to get cross-section data

 HATCH-call comes next

 EGS SUCCESSFULLY 'HATCHED' FOR     2 MEDIA.

 Knock-on electrons can be created and any electron followed down to
                                         0.189 MeV kinetic energy
 Brem photons can be created and any photon followed down to
                                         0.010 MeV
CPU time =       53.031     sec for    50000 cases

 Fraction of electrons reflected from plate=            7.5%
 Fraction of electrons transmitted through plate=      64.1%


 Fraction of energy reflected from plate=               3.3%
 Fraction of energy deposited in plate=                62.4%
 Fraction of energy transmitted through plate=         34.4%
                                                 -----------
 Total fraction of energy accounted for=              100.0%
```

3.5 Tutorial 5 (Program tutor5.f)

In this program we give an example that includes Rayleigh scattering and which makes use of a variable called `LATCH` (contained in `COMMON/STACK/`). `LATCH` can be set for any particle on the "stack" of particles being transported, and it is passed on to all its progeny. This provides a simple procedure for keeping track of the histories of particles. In this case we make use of `LATCH` to keep track of how often photons from an incident 50 keV beam are Compton or Rayleigh scattered while passing through a 0.5 cm slab of water.

The program also demonstrates the use of the `IAUSFL` array of flags (in `COMMON/EPCONT/`). By setting the appropriate flags, the user can cause the EGS5 system to call the `AUSGAB` subroutine in any combination of 31 well specified situations (see Appendix B). By default, EGS calls `AUSGAB` only 5 out of the possible 31 situations. Here, by setting `IAUSFL(18)` and `IAUSFL(24)` from 0 (default) to 1 in the main program, we cause EGS to call `AUSGAB` with `IARG=17` and `IARG=23` (*i.e.*, just before a Compton or a Rayleigh scattering event, respectively). We make use of these calls to set some flags associated with each photon rather than for scoring any variables. A complete listing of **tutor5.f**, except for `HOWFAR` routine which is similar to the other examples, is given below.

```
!***********************************************************************
!
!                      **************
!                      *            *
!                      *  tutor5.f  *
!                      *            *
!                      **************
!
!  An EGS5 user code which scores the number and average energy of the
!  primary, Rayleigh scattered and Compton scattered photons passing
!  through a  5 cm thick slab of water when a 50 keV pencil beam of
!  photons is incident normally
!
!
!  For SLAC-R-730/KEK Report 2005-8:  Example of including Rayleigh
!         scattering, and use of the LATCH feature
!
!  The following units are used: unit 6 for output
!***********************************************************************
!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_epcont.f'
      include 'include/egs5_media.f'
      include 'include/egs5_misc.f'
      include 'include/egs5_stack.f'
      include 'include/egs5_thresh.f'
      include 'include/egs5_useful.f'
      include 'include/egs5_usersc.f'
      include 'include/randomm.f'

!     bounds contains ecut and pcut
!     epcont contains iausfl
!     media contains the array media
!     misc contains med
!     stack contains latchi
!     thresh contains ae and ap
!     useful contains RM
!     usersc contains estepe and estepe2

      common/geom/zbound
      real*8 zbound
!     geom passes info to our howfar routine

      common/score/count(3),entot(3)
      real*8 count,entot

      real*8 ein,xin,yin,zin,             ! Arguments
     *       uin,vin,win,wtin
      integer iqin,irin

      real*8 anorm                             ! Local variables
      integer i,j,ncase
      character*24 medarr(1)

!     ----------
!     Open files
!     ----------
      open(UNIT= 6,FILE='egs5job.out',STATUS='unknown')

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

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

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

!     ---------------------------------
!     define media before calling PEGS5
!     ---------------------------------
      nmed=1
      medarr(1)='H2O                     '

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

! nmed and dunit default to 1, i.e. one medium and we work in cm

      chard(1) = 0.5d0       !  optional, but recommended to invoke
                             !  automatic step-size control

!     ---------------------------------------------
!     Run KEK version of PEGS5 before calling HATCH
!     (method was developed by Y. Namito - 010306)
!     ---------------------------------------------
      write(6,100)
100   FORMAT(' PEGS5-call comes next'/)

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

!-----------------------------------------------------------------------
! Step 3: Pre-hatch-call-initialization
!-----------------------------------------------------------------------
      nreg=3
!     nreg : number of region

      med(1)=0
      med(3)=0
      med(2)=1
! Regions 1 and 3 are vacuum, region 2, H2O
      ecut(2)=1.5
!     Terminate electron histories at 1.5 MeV in the slab
      pcut(2)=0.010
!     Terminate   photon histories at 0.01 MeV in the slab
      iraylr(2)=1
!     Turn on rayleigh scattering in the slab
! Note, above three parameters need to be set for all regions in which
! there is particle transport - just region 2 in this case
!     -------------------------------
!     Set parameter estepe and estepe2
!     -------------------------------
      estepe=0.10
      estepe2=0.20
      write(6,110) estepe, estepe2
110   FORMAT(1X,'ESTEPE at EKMAX: ',F10.5,' (estepe)',
     *       /,1X,'ESTEPE at ECUT:  ',F10.5,' (estepe2)')

!     ---------------------------------------------------
!     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 50 keV beam of photons normally incident
! on the slab
      iqin=0
!     Incident photons
!             50 keV
      ein=0.050
      xin=0.0
      yin=0.0
      zin=0.0
!     Incident at origin
      uin=0.0
      vin=0.0
      win=1.0
!     Moving along z axis
      irin=2
!     Starts in region 2, could be 1
      wtin=1.0
!     weight = 1 since no variance reduction used
      latchi=0
!     latch set to zero at start of each history

!-----------------------------------------------------------------------
! Step 5:   hatch-call
```

```
!-----------------------------------------------------------------------
! Maximum total energy of an electron for this problem must be
! defined before hatch call
      emaxe = ein + RM

      write(6,130)
130   format(/' Start tutor5'/' 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)

!    Pick up cross section data for water
      write(6,150) ae(1)-0.511, ap(1)
150   format(/' Knock-on electrons can be created and any electron ',
     *'followed down to' /T40,F8.3,' MeV kinetic energy'/
     *' Brem photons can be created and any photon followed down to',
     */T40,F8.3,' MeV')
! Compton events can create electrons and photons below these cutoffs

!-----------------------------------------------------------------------
! Step 6:  Initialization-for-howfar
!-----------------------------------------------------------------------
      zbound=0.5
!     Plate is 0.5 cm thick

!-----------------------------------------------------------------------
! Step 7:  Initialization-for-ausgab
!-----------------------------------------------------------------------
      do i=1,3
        count(i)=0.0
        entot(i)=0.0
!  Zero scoring array at start
      end do

!  We want to set flags in ausgab every time a rayleigh scattering
!  or Compton scattering occurs. Set the flags in iausfl(comin
!  epcont) to signal the  egs system to make the appropriate calls
      iausfl(18)=1
      iausfl(24)=1

!-----------------------------------------------------------------------
! Step 8:  Shower-call
!-----------------------------------------------------------------------
! Initiate the shower ncase times
      ncase=10000
      do i=1,NCASE
        call shower(iqin,ein,xin,yin,zin,uin,vin,win,irin,wtin)
      end do

!-----------------------------------------------------------------------
! Step 9:  Output-of-results
!-----------------------------------------------------------------------
! Normalize to % of photon number
      anorm = 100./float(ncase)
      do i=1,3
        if (count(i).ne.0) then
          entot(i)=entot(i)/count(i)
!    Get average energies
        end if
      end do
      write(6,160) ein*1000.,zbound, pcut(2), (anorm*count(i),entot(i),
     *i=1,3)
160   format(/' For',F6.1,' keV photons incident on',F4.1,'cm of H2O',
     *' with PCUT=',F5.3,' MeV' //' Transmitted primaries=',T40,F8.2,
     *'%  ave energy=',F10.3,' MeV'// ' Fraction Rayleigh scattering=',
     *T40,F8.2,'%  ave energy=',F10.3,' MeV' //
     *' Fraction Compton scattering only=',T40,F8.2,'%  ave energy=',
     *F10.3, ' MeV'//)

      stop
      end
!-------------------------last line of main code------------------------

!-------------------------------ausgab.f--------------------------------
!-----------------------------------------------------------------------
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12
! ----------------------------------------------------------------------
! Required subroutine for use with the EGS5 Code System
! ----------------------------------------------------------------------
!***********************************************************************
!
!  In this AUSGAB routine for TUTOR5 we both set flags whenever there is
!  a scattering event and then count histories when they have come
!  through the slab , according to what kind of scattering they have
!  undergone.
!  The logic is as follows
!   set FLAG1 if a Compton event occurs
!   set FLAG2 if a Rayleigh event occurs
!  The FLAGS are the units and thousands digits in the parameter LATCH
!
!  When a history is terminated, increment various counters according
!  to whether no flags are set - i.e. its a primary, FLAG2 is set,
!  i.e. it has Rayleigh scattered or FLAG1 is set and FLAG2 is not set
!   i.e. only  Compton scattering has occurred.
!***********************************************************************
      subroutine ausgab(iarg)

      implicit none

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

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

      common/score/count(3),entot(3)
      real*8 count,entot

      integer iarg                                          ! Arguments

      integer jj                                       ! Local variable

      if (iarg.eq.17) then
!  A Compton scatter is about to occur
        latch(np)=latch(np)+1
      else if (iarg.eq.23) then
!  A Rayleigh scatter is about to occur
        latch(np)=latch(np)+1000
!   If a history has terminated because leaving the slab, score it
!  Particle has left slab
      else if (iarg .eq. 3) then
        if (ir(np).eq.3 .or. ir(np) .eq. 1) then
!    It is transmitted or reflected
          jj=0
          if (latch(np) .eq. 0) then
!      No scattering - a primary
            jj=1
          else if (mod(latch(np),10000)-mod(latch(np),100) .ne. 0) then
!      at least one Rayleigh scatter
            jj=2
          else if (mod(latch(np),100) .ne. 0) then
!      at least one Compton scatter without Rayleigh
            jj=3
!      debug
          else
            write(6,1080) jj,latch(np)
1080        format(' jj,latch(np)=',2I10)
          end if
          if (jj .ne. 0) then
```

```
            count(jj)=count(jj) + 1.
            entot(jj) = entot(jj) + e(np)
          end if
!    End region 3 block
        end if
!  End iarg 3 block
      end if
      return
      end
!--------------------------last line of ausgab.f------------------------
```

Following is the output provided by **tutor5.f**.

```
 PEGS5-call comes next

 ESTEPE at EKMAX:    0.10000 (estepe)
 ESTEPE at ECUT:     0.20000 (estepe2)

 inseed=           1      (seed for generating unique sequences of Ranlux)
 ranlux luxury level set by rluxgo : 1      p=  48
 ranlux initialized by rluxgo from seed           1

 Start tutor5
 Call hatch to get cross-section data

 HATCH-call comes next

 RAYLEIGH OPTION REQUESTED FOR MEDIUM NUMBER  1

 EGS SUCCESSFULLY 'HATCHED' FOR ONE MEDIUM.

 Knock-on electrons can be created and any electron followed down to
                                         0.010 MeV kinetic energy
 Brem photons can be created and any photon followed down to
                                         0.010 MeV

 For  50.0 keV photons incident on 0.5cm of H2O with PCUT=0.010 MeV

 Transmitted primaries=                  88.89%  ave energy=     0.050 MeV

 Fraction Rayleigh scattering=            0.95%  ave energy=     0.049 MeV

 Fraction Compton scattering only=        8.60%  ave energy=     0.046 MeV
```

3.6 Tutorial 6 (Program tutor6.f)

One of the important features of the EGS5 Code System is that the user has direct control over the geometry in which the radiation transport takes place, and rather complex geometries can be described in a very simple manner with the aid of geometry subprograms. Subroutine HOWFAR, in the previous examples, was made more detailed than necessary for demonstration purposes. One can greatly simplify things, particularly for the simple slab geometry case, as shown in the following excerpts from **tutor6**.

```
!***********************************************************************
!
!                     **************
!                     *            *
!                     *  tutor6.f  *
!                     *            *
!                     **************
!
!  An EGS5 user code. It lists the particles escaping from the back
!  of a 1 mm Ta plate when a pencil beam of  20 MeV electrons
!  is incident on it normally.
!
!  NOTE: This program is the same as TUTOR1.f except that the
!        geometry subroutine (HOWFAR) is simplified by the use of
!        the general purpose geometry subroutines PLAN2P.
       .
       .
       .
!-----------------------------------------------------------------------
! Step 6:  Initialization-for-howfar
!-----------------------------------------------------------------------
! Define the coordinates and the normal vectors for the two planes.
! Information required by howfar (and auxiliary geometry subprograms)
! and passed through common/pladta/
!
! First plane (the x-y plane through the origin)
      pcoord(1,1)=0.0
      pcoord(2,1)=0.0
      pcoord(3,1)=0.0
! Coordinates
      pnorm(1,1) =0.0
      pnorm(2,1) =0.0
      pnorm(3,1)= 1.0
! Normal vectors
! Second plane (note: slab is 1 mm thick)
      pcoord(1,2)=0.0
      pcoord(2,2)=0.0
      pcoord(3,2)=0.1
```

```
! Coordinates
      pnorm(1,2) =0.0
      pnorm(2,2) =0.0
      pnorm(3,2)= 1.0
! Normal vectors
       .
       .
       .
!------------------------------howfar.f-------------------------------
!---------------------------------------------------------------------
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12
! --------------------------------------------------------------------
! Required (geometry) subroutine for use with the EGS5 Code System
!*********************************************************************
!
! The following is a general specification of howfar.  Essentially
! it is the same as that given in tutor1.f with the following
! exception: 1) Particles must be initially begin in region 2 and are
!               discarded when that enter region 1 or 3 (no check
!               is made on w(np)).
!            2) The coding is much simplified (i.e., modular)
!               As a result of using auxiliar geometry subprogram
!               plan2p (which calls plane1 and chgtr which require
!               commons epcont, pladta, and stack).
!
!   The user can terminate a history by setting idisc>0. Here we
!   terminate all histories which enter region 3 or are going
!   backwards in region 1
!
!                   |               |
!   Region 1        |   Region 2    |       Region 3
!                   |               |
!   e- =========>   |               | e- or photon ====>
!                   |               |
!   vacuum          |     Ta        |       vacuum
!
! DESCRIPTION - PLAN2P is generally called from subroutine HOWFAR
!   whenever a particle is in a region bounded by two planes that
!   ARE parallel.  Both subroutines PLANE1 and CHGTR are called
!   by PLAN2P (the second PLANE1 call is not made if the first
!   plane is hit, or if the trajectory is parallel).
!------------------------------------------------------------------
!     NPL1   = ID number assigned to plane called first (input)
!     NRG1   = ID number assigned to region particle trajectory
!              will lead into
!     ISD1   =  1 normal points towards current region (input)
!            = -1 normal points away from current region (input)
!     NPL2   = Same (but for plane called second)
!     NRG2   = Same (but for plane called second)
!     ISD2   = Same (but for plane called second)
!*********************************************************************
```

```
      subroutine howfar

      implicit none

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

      include 'include/egs5_epcont.f'    ! epcont contains irnew, ustep
                                         ! and idisc
      include 'include/egs5_stack.f'     ! stack contains x, y, z, u, v,
                                         ! w, ir and np

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

      include 'auxcommons/pladta.f'

      integer irl                                 ! Local variable

      irl=ir(np)              ! Set local variable
      if (irl.ne.2) then
        idisc=1       ! Terminate this history if not in plate
      else            ! We are in the Ta plate - check the geometry
        call plan2p(irl,irl+1,1,irl-1,irl-1,-1)
      end if

      return
      end

!--------------------------last line of howfar.f------------------------
```

The actual HOWFAR code is now less than 10 lines long, the rest consisting of COMMON's and comments. For a complete understanding of how PLAN2P and its related subroutines PLANE1 and CHGTR are called and used, the reader should refer to comments in the appropriate subroutine (distributed with the EGS5 Code System). For a description of the concepts involved in modeling geometry for Monte Carlo programs in general (and EGS4 in particular), the reader may wish to refer to the document "How to Code Geometry: Writing Subroutine HOWFAR," which is provided with the EGS5 distribution.

3.7 Tutorial 7 (Program tutor7.f)

In this program we give an example that includes K- and L-fluorescence photons which can be possible in any material including a compound or a mixture. Here we investigate the reflected photon spectrum from 1 cm of lead when a 100 keV beam of photons is incident on it. If the **IEDGFL** flag is set to 1, fluorescence photons can be produced after K- or L-photoelectric effect interactions in that region. A complete listing of **tutor7**, except `HOWFAR` routine which is the same as **tutor6.f**, is given below.

```
!***********************************************************************
!
!                        **************
!                        *            *
!                        *  tutor7.f  *
!                        *            *
!                        **************
!
! An EGS5 user code which scores the spectrum of reflection from
! 1.0 cm thick slab of lead when a 100 keV beam of photons is incident
! on it with or without fluorescence photons.
!
!  For SLAC-R-730/KEK Report 2005-8:  Example of including fluorescence
!
!  The following units are used: unit 6 for output
!***********************************************************************
!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_edge.f'
      include 'include/egs5_epcont.f'
      include 'include/egs5_media.f'
      include 'include/egs5_misc.f'
      include 'include/egs5_thresh.f'
      include 'include/egs5_useful.f'
      include 'include/egs5_usersc.f'
      include 'include/randomm.f'

!     bounds contains ecut and pcut
!     edge contains iedgfl
!     epcont contains iausfl
!     media contains the array media
!     misc contains med
!     thresh contains ae and ap
!     useful contains RM
!     usersc contains estepe and estepe2

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

      include 'auxcommons/pladta.f'

      common/score/bwidth,ebin(50)
      real*8 bwidth,ebin

      real*8 ein,xin,yin,zin,               ! Arguments
     *       uin,vin,win,wtin
      integer iqin,irin

      real*8 binmax                              ! Local variables
      integer i,icol,j,ncase
      character*24 medarr(1)
      character*4 line(48)

!     ----------
!     Open files
!     ----------
      open(UNIT= 6,FILE='egs5job.out',STATUS='unknown')

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

!-----------------------------------------------------------------------
! Step 2: pegs5-call
!-----------------------------------------------------------------------
!     ==============
      call block_set                 ! Initialize some general variables
!     ==============

!     --------------------------------
!     define media before calling PEGS5
!     --------------------------------
      nmed=1
      medarr(1)='PB                      '

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

```
! nmed and dunit default to 1, i.e. one medium and we work in cm

      chard(1) = 1.0d0       !  optional, but recommended to invoke
                             !  automatic step-size control

!     ------------------------------------------------
!     Run KEK version of PEGS5 before calling HATCH
!     (method was developed by Y. Namito - 010306)
!     ------------------------------------------------
      write(6,100)
100   FORMAT(' PEGS5-call comes next'/)

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

!-----------------------------------------------------------------------
! Step 3: Pre-hatch-call-initialization
!-----------------------------------------------------------------------
      nreg=3
!     nreg : number of region

      med(1)=0
      med(3)=0
      med(2)=1
! Regions 1 and 3 are vacuum, region 2, lead
      iraylr(2)=1
!     Turn on rayleigh scattering in the slab
      iedgfl(2)=1
!     1: Turn on fluorescence production in the slab
!     0: Turn off fluorescence production in the slab
! Note, above three parameters need to be set for all regions in which
! there is particle transport - just region 2 in this case

!     --------------------------------
!     Set parameter estepe and estepe2
!     --------------------------------
      estepe=0.10
      estepe2=0.20
      write(6,110) estepe, estepe2
110   FORMAT(' ESTEPE at EKMAX: ',F10.5,' (estepe)',
     *       /' ESTEPE at ECUT:  ',F10.5,' (estepe2)')

!     ------------------------------------------------------
!     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 100 keV beam of photons normally incident
! on the slab
      iqin=0
!     Incident photons
!             100 keV
      ein=0.100
      xin=0.0
      yin=0.0
      zin=0.0
!     Incident at origin
      uin=0.0
      vin=0.0
      win=1.0
!     Moving along z axis
      irin=2
!     Starts in region 2, could be 1
      wtin=1.0
!     weight = 1 since no variance reduction used

!-----------------------------------------------------------------------
! Step 5:   hatch-call
!-----------------------------------------------------------------------
! Maximum total energy of an electron for this problem must be
! defined before hatch call
      emaxe = ein + RM

      write(6,130)
130   format(/' Start tutor7'/' 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)

!    Pick up cross section data for lead
      write(6,150) ae(1)-0.511, ap(1)
150   format(/' Knock-on electrons can be created and any electron ',
     *'followed down to' /T40,F8.3,' MeV kinetic energy'/
     *' Brem photons can be created and any photon followed down to',
     */T40,F8.3,' MeV')
! Compton events can create electrons and photons below these cutoffs

!-----------------------------------------------------------------------
! Step 6:  Initialization-for-howfar
!-----------------------------------------------------------------------
! Define the coordinates and the normal vectors for the two planes.
! Information required by howfar (and auxiliary geometry subprograms)
! and passed through common/pladta/
!
! First plane (the x-y plane through the origin)
      pcoord(1,1)=0.0
      pcoord(2,1)=0.0
      pcoord(3,1)=0.0
! Coordinates
      pnorm(1,1) =0.0
      pnorm(2,1) =0.0
      pnorm(3,1)= 1.0
! Normal vectors
! Second plane (note: slab is 1 cm thick)
      pcoord(1,2)=0.0
      pcoord(2,2)=0.0
      pcoord(3,2)=1.0
! Coordinates
      pnorm(1,2) =0.0
      pnorm(2,2) =0.0
      pnorm(3,2)= 1.0
! Normal vectors

!-----------------------------------------------------------------------
! Step 7:  Initialization-for-ausgab
!-----------------------------------------------------------------------
      do i=1,50
        ebin(i) = 0.0
!  Zero scoring array before starting
      end do
      bwidth = 0.002

!-----------------------------------------------------------------------
! Step 8:  Shower-call
```

```
!-----------------------------------------------------------------------
! Initiate the shower ncase times
      ncase=10000
      do i=1,NCASE
        call shower(iqin,ein,xin,yin,zin,uin,vin,win,irin,wtin)
      end do

!-----------------------------------------------------------------------
! Step 8:  Output-of-results
!-----------------------------------------------------------------------
! Use log10(10000.0) as maximum value
      binmax=dlog10(10000.d0)

      if (iedgfl(2).eq.1) then
        write(6,160) ein,pcoord(3,2)
160     format(/' Reflected photon spectrum'/' for a',F8.2,
     *  ' MeV pencil beam of photons on a',F7.2,
     *  ' cm thick slab of lead'/' with fluorescence photon'//T6,
     *  'Energy  counts/incident photon'/
     *  25X,' log(counts for 10^4 incident photons)')
      else
        write(6,170) ein,pcoord(3,2)
170     format(' Reflected photon spectrum'/' for a',F8.2,
     *  ' MeV pencil beam of photons on a',F7.2,
     *  ' cm thick slab of lead'/' without fluorescence photon'//T6,
     *  'Energy  counts/incident photon'/
     *  25X,' log(counts for 10^4 incident photons)')
      end if

      do j=1,48
        line(j)=' '
      end do
! Blank entire output array
      do j=1,50
        if(ebin(j).gt.0) then
          icol=
     *       int(dlog10(ebin(j)*10000.0/float(ncase))/binmax*48.0+0.999)
          if (icol.eq.0) icol=1
        else
          icol = 1
        endif
        line(icol)='*'
!  Load output array at desired location
        write(6,180) bwidth*j,ebin(j)/float(ncase),line
180     format(F10.4,F10.4,48A1)
        line(icol)=' '
!  Reblank
      end do

      stop
      end
```

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

!------------------------------ausgab.f-------------------------------
!-----------------------------------------------------------------------
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12
! ----------------------------------------------------------------------
! Required subroutine for use with the EGS5 Code System
! ----------------------------------------------------------------------
!***********************************************************************
!
!  In this AUSGAB routine for TUTOR7 we score photons reflected
!  from the slab (ir(np)=1 and iq(np)=0).
!***********************************************************************
      subroutine ausgab(iarg)

      implicit none

      include 'include/egs5_h.f'                ! Main EGS "header" file
      include 'include/egs5_stack.f'     ! COMMONs required by EGS5 code

      common/score/bwidth,ebin(50)
      real*8 bwidth,ebin

      integer iarg                                          ! Arguments

      integer ibin,irl                                ! Local variable

      irl=ir(np)                     ! Local variable
      if(irl.eq.1.and.iq(np).eq.0) then   ! Photon is reflected
!     Increment bin corresponding to photon energy
        ibin= min0 (int(e(np)/bwidth + 0.999), 50)
        if (ibin.ne.0) then
          ebin(ibin)=ebin(ibin)+1
        end if
      end if

      return
      end
!--------------------------last line of ausgab.f------------------------
```

The following is the output provided by **tutor7** with and without the fluorescence option (named tutor7_w.out and tutor7_wo.out in distribution file, respectively).

```
 PEGS5-call comes next

 ESTEPE at EKMAX:    0.10000 (estepe)
 ESTEPE at ECUT:     0.20000 (estepe2)

 inseed=           1      (seed for generating unique sequences of Ranlux)
```

```
ranlux luxury level set by rluxgo : 1     p=  48
ranlux initialized by rluxgo from seed           1

Start tutor7
Call hatch to get cross-section data

HATCH-call comes next

RAYLEIGH OPTION REQUESTED FOR MEDIUM NUMBER  1

EGS SUCCESSFULLY 'HATCHED' FOR ONE MEDIUM.

Knock-on electrons can be created and any electron followed down to
                                         0.010 MeV kinetic energy
Brem photons can be created and any photon followed down to
                                         0.001 MeV

Reflected photon spectrum
for a    0.10 MeV pencil beam of photons on a   1.00 cm thick slab of lead
with fluorescence photon

    Energy  counts/incident photon
                         log(counts for 10^4 incident photons)
   0.0020    0.0000*
   0.0040    0.0000*
   0.0060    0.0000*
   0.0080    0.0000*
   0.0100    0.0001*
   0.0120    0.0014             *
   0.0140    0.0037                  *
   0.0160    0.0004       *
   0.0180    0.0000*
   0.0200    0.0000*
   0.0220    0.0000*
   0.0240    0.0000*
   0.0260    0.0000*
   0.0280    0.0000*
   0.0300    0.0000*
   0.0320    0.0000*
   0.0340    0.0000*
   0.0360    0.0000*
   0.0380    0.0000*
   0.0400    0.0000*
   0.0420    0.0000*
   0.0440    0.0000*
   0.0460    0.0000*
   0.0480    0.0000*
   0.0500    0.0000*
   0.0520    0.0001*
   0.0540    0.0001*
   0.0560    0.0000*
```

20 L-X rays are included for Pb in egs5.

L-alpha-1 & L-alpha-2 10.5 keV

L-beta-1 & L-beta-2 12.6 keV

L-gamma-1 14.8 keV

$\updownarrow$

```
   0.0580    0.0000*
   0.0600    0.0006         *
   0.0620    0.0004       *
   0.0640    0.0002   *
   0.0660    0.0004       *
   0.0680    0.0008          *                     10 K-X rays are included for Pb in egs5
   0.0700    0.0008          *
   0.0720    0.0004       *
   0.0740    0.0490                                *   K-alpha-2 72.79 keV
   0.0760    0.0894                                   *  K-alpha-1 74.96 keV
   0.0780    0.0007          *
   0.0800    0.0011            *
   0.0820    0.0009           *
   0.0840    0.0001*
   0.0860    0.0378                              *  K-beta-1 84.92 keV
   0.0880    0.0091                       *       K-beta-2 87.34 keV
   0.0900    0.0001*
   0.0920    0.0000*
   0.0940    0.0000*
   0.0960    0.0000*
   0.0980    0.0000*
   0.1000    0.0005        *    Rayleigh Scattering


PEGS5-call comes next

ESTEPE at EKMAX:    0.10000 (estepe)
ESTEPE at ECUT:     0.20000 (estepe2)

inseed=          1       (seed for generating unique sequences of Ranlux)
ranlux luxury level set by rluxgo : 1     p=  48
ranlux initialized by rluxgo from seed          1

Start tutor7
Call hatch to get cross-section data

HATCH-call comes next

RAYLEIGH OPTION REQUESTED FOR MEDIUM NUMBER  1

EGS SUCCESSFULLY 'HATCHED' FOR ONE MEDIUM.

Knock-on electrons can be created and any electron followed down to
                                          0.010 MeV kinetic energy
Brem photons can be created and any photon followed down to
                                          0.001 MeV
Reflected photon spectrum
for a    0.10 MeV pencil beam of photons on a   1.00 cm thick slab of lead
without fluorescence photon

    Energy  counts/incident photon
```

```
                      log(counts for 10^4 incident photons)
0.0020    0.0000*
0.0040    0.0000*
0.0060    0.0000*
0.0080    0.0000*
0.0100    0.0000*
0.0120    0.0000*
0.0140    0.0000*
0.0160    0.0000*
0.0180    0.0000*
0.0200    0.0000*
0.0220    0.0000*
0.0240    0.0000*
0.0260    0.0001*
0.0280    0.0000*
0.0300    0.0000*
0.0320    0.0000*
0.0340    0.0000*
0.0360    0.0000*
0.0380    0.0000*
0.0400    0.0000*
0.0420    0.0000*
0.0440    0.0000*
0.0460    0.0000*
0.0480    0.0000*
0.0500    0.0000*
0.0520    0.0000*
0.0540    0.0000*
0.0560    0.0000*
0.0580    0.0000*
0.0600    0.0000*
0.0620    0.0000*
0.0640    0.0001*
0.0660    0.0000*
0.0680    0.0000*
0.0700    0.0001*
0.0720    0.0001*
0.0740    0.0012            *        71.9 keV @ 180 deg
0.0760    0.0013             *
0.0780    0.0008          *                                  Compton Scattering
0.0800    0.0008          *
0.0820    0.0002   *
0.0840    0.0003     *               83.6 keV @  90 deg
0.0860    0.0001*
0.0880    0.0000*
0.0900    0.0000*
0.0920    0.0000*
0.0940    0.0000*
0.0960    0.0000*
0.0980    0.0000*
0.1000    0.0008          *          Rayleigh scattering
```