KEK Internal 2011-8
December 2011
R/D

Lecture Notes of Dose distribution calculation inside phantom with Voxel

(Modified at June 27, 2016)

H. Hirayama and Y. Namito

High Energy Accelerator Reserach Organization

KEK Reports are available from

High Energy Accelerator Research Organization (KEK)
1-1 Oho, Tsukuba-sh
Ibaraki-ken, 305-0801
JAPAN

Phone: +81-29-864-5137
Fax: +81-29-864-4604
E-mail: irdpub@mail.kek.jp
Internet: http://www.kek.jp

Lecture Notes of Dose distribution calculation inside phantom with Voxel

H. Hirayama and Y. Namito

High Energy Accelerator Reserach Organization

Contents

egs5サンプルプログラム (ucxyz_phantom.f)
ファントム中の線量分布計算 (Voxel形状)

(Japanese Parts)

1　サンプルプログラム ucxyz_phantom.f の概要

ucxyz_phantom.f は、以下の計算を行うユーザコードである。

1. 形状 (第 1 図)

   - ３次元ボクセル形状
   - Z 方向のボクセル数　22
   - Y 方向のボクセル数　3
   - X 方向のボクセル数　3
   - 人体を一様な水でモデル化　X-, Y-方向 30cm, 深さ 20cm
   - 人体の前後に 5cm の空気

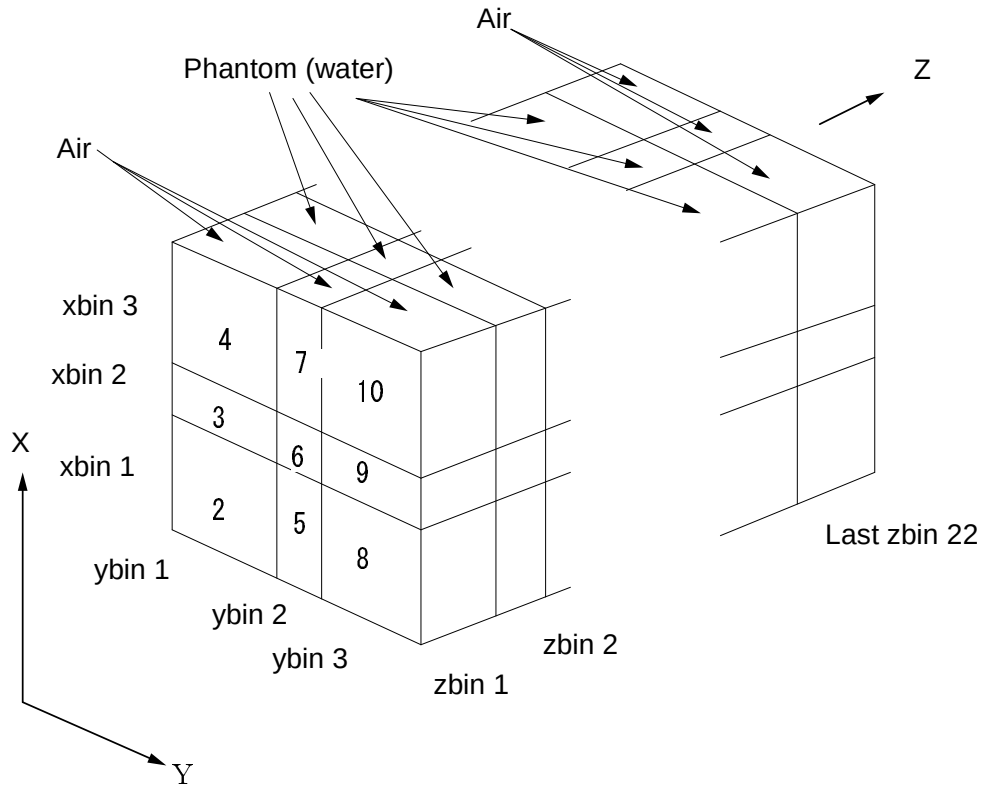

Figure 1: ucxyz_phantom.f のジオメトリー。

2. 線源条件

   - 入射粒子は、エネルギー 1.253MeV の光子
   - 点等方線源:位置は、人体表面からの距離 (SPOSI=10cm)
   - ビームサイズ：人体表面で XHBEAM(=1cm)*2 × YHBEAM(=1cm)*2 のビーム。

3. 得られる情報

   (a) CGView 用飛跡情報 (`egs5job.pic`)

   (b) 計算結果 (`egs5job.out`)

   - 使用する物質に関するデータ

- 各リージョンに関するデータ
- 定義した平板に関するデータ
- ヒストリー数、ビームサイズ
- ファントム中心の 1cm × 1cm の領域での深度線量分布 (1cm 単位)
- 後方散乱係数
- 各リージョンの吸収エネルギー割合

2 ユーザーコードの内容

2.1 メインプログラム Step 1

2.1.1 include 文及び型式宣言

egs5 は、Fortran で書かれているので、egs5 やジオメトリー等ユーザーコードで使われている変数の配列の大きさは、別のファイルに parameter 文で指定し、include 機能によりユーザーコードに取り入れている。common についても、同じく include 機能を用いている。

egs5 に直接関係する include 関係のファイルは、include/ディレクトリ（egs に関係するもの）、pegscommons/ディレクトリ（pegs に関係するもの）および auxcommons/ディレクトリ（egs5 の著者から提供しているジオメトリー関係のサブルーティン等ユーザーコードにのみ関係するもの）とリンクすることにより使用できるようにしている。[1]

この点が、Mortran のマクロ機能により、ユーザーコードで再設定できた EGS4 の場合と最も異なることである。配列の大きさを変更する場合には、egs5 に直接関係する場合は、include/egs5_h.f 内の、その他の場合は、auxcommons/aux_h.f の当該 parameter 文の値を変更することになる。

最初の設定は、egs に直接関連する include 文である。

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

include 'include/egs5_bounds.f'
include 'include/egs5_edge.f'
include 'include/egs5_elecin.f'
include 'include/egs5_media.f'
include 'include/egs5_misc.f'
include 'include/egs5_stack.f'
include 'include/egs5_thresh.f'
include 'include/egs5_uphiot.f'
include 'include/egs5_useful.f'
include 'include/egs5_usersc.f'
include 'include/randomm.f'
```

include 'include/egs5_h.f' は、必ず必要であるが、それ以外の common に関連する include 文は、メインプログラムで、使用する可能性があるものだけで良い。[2]

次の設定は、ジオメトリー関係のサブルーティン及びユーザー固有のユーザーコードに関連する include 文である。

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

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

特定のユーザーコード内で使用する common を次に定義する。

[1] これらの設定は、egs5run スクリプトで設定される。
[2] EGS4 の COMIN マクロに対応する扱いである。

```
      common/score/                                ! Variables to score
     *            depe(LIMAX,LJMAX,LKMAX),faexp,fexps,maxpict
      real*8 depe,faexp,fexps
      integer maxpict
```

メインプログラムの先頭で、implicit none 宣言をしているので、メインプログラムで使用している全ての変数の型式宣言をする必要がある。

2.1.2 open 文

実行文の先頭で、使用するユニットを open する。egs5 では、pegs をプログラムの一部として含む構造を標準としている。pegs の実行に伴い、ユニット 7-26 は、close されることから、メインプログラムで open していても、pegs 実行後に、再度 open することが必要となる。そのため、ユニット 7-26 の使用を避ける方が良い。ユニット 39 は、飛跡情報の出力ファイルである。

```
!-----------------------------------------------------------------
!     Units 7-26 are used in pegs and closed.  It is better not
!     to use as output file. If they are used must be re-open after
!     getvoxel 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')
```

その後、各カウンターをリセットするサブルーティン counters_out(0) を call する。

2.2 メインプログラム Step 2: Pegs5-call

物質データ数、各物質の名前、各物質の characteristic dimension、ボクセルを設定する平板データ、リージョンの設定と、各リージョンへの物質及びオプション設定を egs5job.inp から読込むサブルーティン getvoxel(ifto) を call する。ifto は、getvoxel で読み込んだデータを出力するファイル番号である。

その後、サブルーティン pegs5 を call する。

```
      ifto = 6      ! Output unit in getvoxel
!     ===================
      call getvoxel(ifto)
!     ===================

!     ------------------------------
!     Run PEGS5 before calling HATCH
!     ------------------------------
      write(6,*) 'PEGS5-call comes next'

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

2.3 メインプログラム Step 3: Pre-hatch-call-initialization

飛跡データファイルのフォーマットを指定する npreci を設定する。このユーザーコードでは、CGView のフリーフォーマットを使用するので 3 を指定する。使用する飛跡表示システムに対応した形状データ及び各リージョンの物質番号を飛跡データファイルに出力する。

```
!-------------------------------------------
!     Define pict data mode.
!-------------------------------------------
!     npreci 1: for PICT32
!            2: for CGview
!            3: for CGview in free format
      npreci=3
```

```
      if(npreci.eq.3) write(39,fmt="('GSTA-FREE-TIME')")
      if(npreci.eq.2) write(39,fmt="('GSTA-TIME')")
      write(39,fmt="('SLAB')")
      write(39,fmt="(I6)") imax+1
      write(39,fmt="(I6)") jmax+1
      write(39,fmt="(I6)") kmax+1
      write(39,fmt="(4F15.4)") (xbound(j),j=1,imax+1)
      write(39,fmt="(4F15.4)") (ybound(j),j=1,jmax+1)
      write(39,fmt="(4F15.4)") (zbound(j),j=1,kmax+1)
      write(39,fmt="('GEND')")

      write(39,fmt="('MSTA')")
      write(39,fmt="(I4)") nreg
      write(39,fmt="(15I4)") (med(i),i=1,nreg)
      write(39,fmt="('MEND')")
```

Ranlux 乱数のシード inseed の値を設定し、初期化する。

```
!     ---------------------------------------------------------
!     Random number seeds.  Must be defined before call hatch
!     or defaults will be used.  inseed (1- 2^31)
!     ---------------------------------------------------------
      luxlev = 1
      inseed=1
      write(6,100) inseed
100   FORMAT(/,' inseed=',I12,5X,
     *         ' (seed for generating unique sequences of Ranlux)')

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

2.4 メインプログラム Step 4: Determination-of-incident-particle-parameters

線源からファントム表面までの距離、その他の線源パラメータを設定する。

```
!-------------------------------------------------
!     Define source position from phantom surface.
!-------------------------------------------------
!     Source position from phantom surface in cm.
      sposi=10.0

      iqin=0             ! Incident charge - photons
      ekein=1.253        ! Kinetic energy of source photon
      etot=ekein + abs(iqin)*RM
      xin=0.D0
      yin=0.D0
      zin=-sposi
      uin=0.D0
      vin=0.D0
      win=1.D0
      wtin=1.D0

!-----------------------------------------
!     Half width and height at phantom surface
!-----------------------------------------
!     X-direction half width of beam at phantom surface in cm.
      xhbeam=1.0
!     Y-direction half height of beam at phantom surface in cm.
      yhbeam=1.0
      radma2=xhbeam*xhbeam+yhbeam*yhbeam
      wimin=sposi/dsqrt(sposi*sposi+radma2)
```

2.5 メインプログラム Step 5: hatch-call

最大電子エネルギー（全エネルギー）を表す emaxe を設定後に subroutine hatch を call する。
hatch で読み込まれた物質データや、リージョンに設定した情報を確認のために出力する。

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

      write(6,110)
110   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(6,120)
120   FORMAT(/,' HATCH-call comes next',/)

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

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

!-----------------------------------------------------------------------
!     Output medium and region information to file for calculation mode.
!-----------------------------------------------------------------------
      write(6,*) ' Quantities associated with each media:'
      do j=1,nmed
        write(6,130) (media(i,j),i=1,24)
130     FORMAT(/,1X,24A1)
        write(6,140) rhom(j),rlcm(j)
140     FORMAT(5X,' Rho=',G15.7,' g/cm**3     RLC=',G15.7,' cm')
        write(6,150) ae(j),ue(j),ap(j),up(j)
150     FORMAT(5X,' AE=',G15.7,' MeV    UE=',G15.7,' MeV'/ 5X,' AP=',G
     *  15.7,' MeV    UP=',G15.7,' MeV')
      end do

      write(6,160)
160   FORMAT(/' Information of medium and cut-off for central region')
      i=imax/2+1
      j=jmax/2+1
      do k=1,kmax
        irl=1+i+(j-1)*imax+(k-1)*ijmax
        if (med(irl).eq.0) then
          write(6,170) k,irl
170       FORMAT(' Medium(',I3,'-th z bin, region:',I5,')= Vacuum')
        else
          write(6,180) k,irl,(media(ii,med(irl)),ii=1,24),
     *                 ecut(irl),pcut(irl),rhor(irl)
180       FORMAT(' Medium(',I3,'-th z bin, region:',I5,
     *           ')=',24A1,/5X,'ECUT=',G10.5,' MeV, PCUT=',
     *           G10.5, ' MeV, density=',F10.3)
        end if
      end do
```

2.6 メインプログラム Step 6: Initialization-for-howfar

普通のユーザーコードでは、このステップで形状を指定するための情報（平板、円筒、球等）を記述するが、本ユーザーコードでは getvoxel で形状を指定しているので、このステップで記述する事項はない。

2.7 メインプログラム Step 7: Initialization-for-ausgab

ausgab に必要な設定を行う。

計算する量の初期化、使用する検出器数、ヒストリー数、飛跡表示ファイルにデータを出力するヒストリー数の設定を行う。飛跡データファイルに、バッチ番号 (1) を出力する。

```
      ncount = 0
      ilines = 0
      nwrite = 10
      nlines = 25
      idin = -1
      totke = 0.
      wtsum = 0.

!---------------------
!     Clear variables
!---------------------
!     Zero the dose
      do k=1,kmax
        do j=1,jmax
          do i=1,imax
            depe(i,j,k)=0.D0
            depeh(i,j,k)=0.D0
            depeh2(i,j,k)=0.D0
          end do
        end do
      end do

      faexp=0.D0
      faexps=0.D0
      faexp2s=0.D0
      fexps=0.D0
      fexpss=0.D0
      fexps2s=0.D0

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

!-------------------------
!     History number
!-------------------------
!     History number
      ncases=100000
!     Maximum history number to write trajectory data
      maxpict=50

      write(39,fmt="('0    1')")
```

2.8 メインプログラム Step 8: Shower-call

設定したヒストリー数 (ncases) だけ `subroutine shower` を `call` し、egs5 を使用する部分である。ucxyz_phantom.f では、`sposi` の位置に、等方線源があり、そこから照射野内に、1.253MeV の光子が出るので、線源光子の方向及び sposi が空気の厚さ (5cm) より長い場合の空気層の表面での位置を決めるルーチンが加わっている。

各ヒストリー毎に、エネルギーバランス (入射運動エネルギーと、体系内外の吸収エネルギーの和が等しいこと) をチェックを行っている。

各ヒストリー終了後、平均値とその分散計算のために、計算対象量の値とその自乗をそれぞれ加算する。

```
                                          ! -------------------------
      do jhist=1,ncases                   ! Start of CALL SHOWER loop
                                          ! -------------------------
        icases=jhist
!-------------------------------------
!       Determine direction (isotropic)
!-------------------------------------
200     call randomset(w0)
        win=w0*(1.0-wimin)+wimin
        call randomset(phai0)
        phai=pi*(2.0*phai0-1.0)
        sinth=dsqrt(1.D0-win*win)
        uin=dcos(phai)*sinth
```

```
        vin=dsin(phai)*sinth
        dis=sposi/win
        xpf=dis*uin
        ypf=dis*vin
        if (dabs(xpf).gt.xhbeam.or.dabs(ypf).gt.yhbeam) go to 200
        if (sposi.gt.zbound(2)-zbound(1)) then
          disair=(sposi-(zbound(2)-zbound(1)))/win
          xin=disair*uin
          yin=disair*vin
          zin=zbound(1)
        else
          xin=0.D0
          yin=0.D0
          zin=-sposi
        end if

        do i=1,imax
            if (xbound(i+1).gt.xin) go to 210
        end do

210     do j=1,jmax
          if (ybound(j+1).gt.yin) go to 220
        end do

!       -------------
!       Input region
!       -------------
220     k=1
        irin=1+i+(j-1)*imax

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

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

        latchi=0

!       ---------------------------------------------------
!       Print first NWRITE or NLINES, whichever comes first
!       ---------------------------------------------------
        if (ncount .le. nwrite .and. ilines .le. nlines) then
          ilines = ilines + 1
          write(6,230) etot,xin,yin,zin,uin,vin,win,iqin,irin,idin
230       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    ' normarized.',3e12.5)")uin,vin,win
          stop
        end if

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

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

!---------------------------------
!      Sum variable and its square.
!---------------------------------

        do k=1,kmax
          do j=1,jmax
            do i=1,imax
              depeh(i,j,k)=depeh(i,j,k)+depe(i,j,k)
              depeh2(i,j,k)=depeh2(i,j,k)+depe(i,j,k)*depe(i,j,k)
              depe(i,j,k)=0.D0
            end do
          end do
        end do

        faexps=faexps+faexp
        faexp2s=faexp2s+faexp*faexp
        faexp=0.0
        fexpss=fexpss+fexps
        fexps2s=fexps2s+fexps*fexps
        fexps=0.0

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

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

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

2.8.1 統計誤差

x をモンテカルロ計算で計算したい量（スコアーする量）とする。モンテカルロ計算の結果には、その統計誤差が必要である。ucxyz_phantom.f では、次のような MCNP で使用している方法を採用している。

- ヒストリー数を N とする。
- x_i を i 番目のヒストリーの結果とする。
- x の平均値を計算する:

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

- x_i の分散値を以下の式から求める。:

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

- $\overline{x}$ の分散値は、

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

  となる。

- 統計誤差として、

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

  を用いる。

先の計算すべき量とその自乗の和は、上記の処理を行うために行っている。

2.9 メインプログラム Step 9: Output of results

得られた結果を処理して打ち出す。最初に線源の条件 (線源のタイプ、位置)、ヒストリー数を出力する。その後、注目する領域での平均吸収線量とその統計誤差を求め出力する。

```
      write(1,280) sposi
280   FORMAT(/' Absorbed energy inside phantom for 1.253MeV photon'/
     *' Source position ',F10.1,' cm from phantom surface'/
     *' Within 1cm x 1 cm area after 5 cm air')

      write(1,290) ncases, xhbeam, yhbeam
290   FORMAT(1X,I8,' photons normally incident from front side'/
     *' Half width of beam is ',G15.5,'cm for X and ',G15.5,'cm for Y')

!      ---------------------------------------
!      Calculate average and its uncertainties
!      ---------------------------------------

        do k=1,kmax
          do j=1,jmax
            do i=1,imax
              irl=1+i+(j-1)*imax+(k-1)*ijmax
              amass=(xbound(i+1)-xbound(i))*
     *              (ybound(j+1)-ybound(j))*
     *              (zbound(k+1)-zbound(k))*rhor(irl)
              dose(i,j,k)=depeh(i,j,k)/ncases
              depeh2(i,j,k)=depeh2(i,j,k)/ncases
              doseun(i,j,k)=dsqrt((depeh2(i,j,k)-
     *                     dose(i,j,k)*dose(i,j,k))/ncases)
              dose(i,j,k)=dose(i,j,k)*1.602D-10/amass
              doseun(i,j,k)=doseun(i,j,k)*1.602D-10/amass
            end do
          end do
        end do

!-----------------------------------------------
!     Print out the results of central phantom
!-----------------------------------------------
      i=imax/2+1
      j=jmax/2+1
      do kkk=2,kmax-1
        depths=zbound(kkk)
        depthl=zbound(kkk+1)
        irl=1+i+(j-1)*imax+(kkk-1)*ijmax
        write(6,300) depths,depthl,(media(ii,med(irl)),ii=1,24),
     *  rhor(irl),dose(i,j,kkk),doseun(i,j,kkk)
300     FORMAT(' At ',F4.1,'--',F4.1,'cm (',24A1,',rho:',F8.4,')=',
     *  G13.5,'+-',G13.5,'Gy/incident')
      end do

!-----------------------------------------------
!     Calculate average exposure and its deviation
!-----------------------------------------------
      area=(xbound(i+1)-xbound(i))*(ybound(j+1)-ybound(j))
      faexpa=faexps/ncases
      faexp2s=faexp2s/ncases
      faexrr=dsqrt((faexp2s-faexpa*faexpa)/ncases)
      faexpa=faexpa*1.6E-10/area
      faexrr=faexrr*1.6E-10/area
      fexpsa=fexpss/ncases
      fexps2s=fexps2s/ncases
      fexerr=dsqrt((fexps2s-fexpsa*fexpsa)/ncases)
      fexpsa=fexpsa*1.6E-10/area
      fexerr=fexerr*1.6E-10/area
      if (faexpa.gt.0.0) then
        bsfa=fexpsa/faexpa
        bsferr=bsfa*dsqrt((faexrr/faexpa)**2.+(fexerr/fexpsa)**2.)
        write(6,310) faexpa,faexrr,fexpsa,fexerr,bsfa,bsferr
```

```
310     FORMAT(/' Exposure in free air (using mu_en) =', G15.5,'+-',G15.
     *  5,' Gy/incident'/ ' Exposure at phantom surface (using mu_en) ='
     *  , G15.5,'+-',G15.5,'Gy/incident'/ ' Backscattering factor =',G15
     *  .5,'+-',G15.5)
      else
        write(6,320) faexpa,faexrr,fexpsa,fexerr
320     FORMAT(/' Exposure in free air (using mu_en) =', G15.5,'+-',G15.
     *  5,' Gy/incident'/ ' Exposure at phantom surface (using mu_en) ='
     *  , G15.5,'+-',G15.5,'Gy/incident')
      end if
```

getvoxel で設定した出力領域の吸収線量とその割合を　指定した方法 (Z-scan 又は X-scan) で出力する。

2.10 subroutine getvoxel(ifto)

getvoxel は、ボクセル形状の問題について、使用する物質データ、各リージョンに設定する物質とオプション、ジオメトリー情報、ボクセルデータの出力方法の指定をユニット 4 から読み込み、必要な設定を行うサブルーチンである。

ユニット 4 から読み込むデータは以下の様になっている。

1. Record 1 : タイトル情報 (80 文字)
2. Record 2 : 使用する物質数 (nmed)
3. Record 3 : 物質名 : 24 文字で指定する。pegs 入力データの名前と対応が必要。
4. Record 4 : 各物質の characteristic dimension
5. Record 5 : X-, Y-, Z-方向のボクセル数 (maxx,maxy,maxz)
   それぞれの値が正の時は、ボクセル数を、値が負の時は、その絶対値が、等間隔指定の組数を意味する。
6. Record 6 : X-方向の平面の位置
   - maxx >0 の時
     maxx + 1 個の X-方向の平面の位置を、1 行に 1 カ所ずつ指定する。
     maxx が、x-方向のビン数の配列数である LIMAX より大きい場合には、その旨のメッセージを出力し、プログラムを止める。配布時には 22 に設定されている LIMAX より大きい maxx にしたい時は、auxcommon/au_h.f 中の LIMAX の値を変更する。
   - maxx < 0 の時
     最も小さい X-方向の位置を指定、その後、abs(maxx) ペアの ボクセル幅とボクセル数を 1 行に 1 組ずつ指定
     maxx 読み込み時に LIMAX との比較が出来ないので、この段階で設定するビン数が、LIMAX を超えていないかを調べ、超えた段階で、その旨のメッセージを出力し、プログラムを止める。配布時には 22 に設定されている LIMAX より大きい maxx にしたい時は、auxcommon/au_h.f 中の LIMAX の値を変更する。
7. Record 7 : Y-方向の平面の位置
8. Record 8 : Z-方向の平面の位置
9. Record 9 : 全てのリージョンを物質番号 1 として、密度、ecut, pcut 及び各種オプション設定を指定する。(0: off, 1:on)

ipeangsw	Switches for PE-angle sampling
iedgeflsw	K & L-edge fluorescence
iaugersw	K & L-edge Auger
iraysw	Rayleigh scattering
ipolarsw	Linearly-polarized photon scattering
incohrsw	S/Z rejection
iprofrsw	Doppler broadening
mpacrsw	electron impact ionization

10. Record 10 :特定のリージョン (il, iu, jl,ju, kl, ku で指定。il $\leq$ i $\leq$ iu, ji $\leq$ i $\leq$ ju, ku $\leq$ k $\leq$ ku の領域が対象) の物質、密度、ecut, pcut の指定
il=iu=0 のデータは、指定モードの終了を意味する。

11. Record 10a : `medtmp` が 0 でない場合には、各種オプション設定を指定する。(0: off, 1:on)

12. Record 11: 結果を出力するリージョン (il, iu, jl,ju, kl, ku で指定。il $\leq$ i $\leq$ iu, ji $\leq$ i $\leq$ ju, ku $\leq$ k $\leq$ ku の領域) と、スキャンの方向 (izscan: izscan $\neq$ 0 の時は、Z-方向のスキャン、それ以外は X-方向のスキャン) を指定する。

13. Record 12 : トラッキング状況を設定するフラグ (iwatch) の指定。
iwatch= 0:トラッキングなし。
iwatch= 1:反応毎のトラッキング、iwatch= 2:ステップ毎のトラッキング

14. Record 13 : 制動輻射 (ibrdst) 及び電子対生成 (iprdst) の際の角度分布オプションの設定及びスプリッティングパラメータ (ibrspl0,nbrspl) の設定。

ibrdst=0	制動輻射で、デフォルト値 ($\theta = m/E$) を使用
ibrdst=1	制動輻射で、サンプリング使用 (recommended)
iprdst=0	電子対生成で、デフォルト値 ($\theta = m/E$) を使用
iprdst=1	電子対生成で、low-order distribution を使用
iprdst=2	電子対生成で、推奨のサンプリングを使用
ibrspl0=0	スプリッティング使用せず
ibrspl0=1	nbrspl にスプリッティング

2.11 `subroutine ausgab`

`ausgab` は、ユーザが求める情報をスコアするサブルーチンである。最初に、メインプログラムと同様に、include 文及びローカル変数の型式宣言を行う。

`iwatch` オプションに伴う処理、スタック番号が、最大値を超えていないことの確認後、途中結果の出力をする。

iarg <5 の場合には、リージョン 1 とそれ以外のリージョンでの吸収エネルギー及び、リージョンが 1 以外の時は、各ボクセルでの吸収エネルギーを計算する。

更に、光子が、ファントム表面を横切った場合かどうかの判定を行い、横切ったと判断した場合には、面エネルギー束と空気のエネルギー吸収計数から、ファントム表面での空気吸収線量を計算する。光子が、Z-軸に対して逆に進んだことがない場合 (ファントムが無い場合のファントム表面位置) には、同様な方式で、ファントム無しの空気の吸収線量を計算する。この計算のため、`w(np)` が負になった場合には、`latch(np))` を 1 にセットし、ファントム無しの計算に加えないようにしている。

ヒストリー数が、飛跡表示ヒストリーの設定数 (`maxpict`) より小さい場合は、粒子の情報を記録する `subroutine plotxyz` を呼ぶ。

```
!     ------------------------------------------------------------
!     Print out particle transport information (if switch is turned on)
!     ------------------------------------------------------------
!                       ========================
```

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

!  --------------------------------
!  Keep track of how deep stack gets
!  --------------------------------
        if (np.gt.MXSTACK) then
          write(6,100) np,MXSTACK
100       FORMAT(//' In AUSGAB, np=',I3,' >= maximum stack',
     *         ' allowed which is',I3/1X,79('*')//)
          stop
        end if
!     ------------------------
!     Set some local variables
!     ------------------------
      irl = ir(np)
      iql = iq(np)
      edepwt = edep*wt(np)

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

!     ------------------------------------------------------------
!     Keep track of energy deposition (for conservation purposes)
!     ------------------------------------------------------------
      if (iarg .gt. 5) return

      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

      i=mod(irl-1,imax)
      if (i.eq.0) i=imax
      k=1+(irl-1-i)/ijmax
      j=1+(irl-1-i-(k-1)*ijmax)/imax

      if (irl.gt.1.and.edep.ne.0.D0) then

        depe(i,j,k)=depe(i,j,k)+edepwt
      end if

!-------------------------------
!     Check cross phantom surface
!-------------------------------
      if(i.eq.imax/2+1.and.j.eq.jmax/2+1) then  ! X-Y central region
        if (abs(irl-irold).eq.ijmax.and.iq(np).eq.0) then
          if ((w(np).gt.0.0.and.k.eq.2).or.
     *     (w(np).le.0.0.and.k.eq.1)) then
            if (dabs(w(np)).ge.0.0349) then
              cmod=dabs(w(np))
            else
              cmod=0.01745
            end if
            esing=e(np)
            dcon=encoea(esing)            ! PHOTX data
            fexps=fexps+e(np)*dcon*wt(np)/cmod
            if (w(np).lt.0.0) latch(np)=1
            if (w(np).gt.0.0.and.latch(np).eq.0) then
              faexp=faexp+e(np)*dcon*wt(np)/cmod
            end if
          end if
        end if
      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),
     *       w(np))
      end if

      return

      end
```

2.12 subroutine howfar

howfar は、粒子の進行方向でのリージョン境界までの距離を計算し、反応点までの距離との比較をし、境界までの距離の方が短い場合には粒子の移動距離を境界までの距離に置き換え、リージョンが変わるという処理を行う。

その他に、howfar では、ユーザが粒子の追跡を止める設定を行う。(idisc=1;) 通常は、粒子が、検討している領域の外に出て追跡を終了する場合にこの設定を行う。

ucxyz_phantom.f では、汎用の voxel 形状用の howfar を使用している。

```
      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'

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

                                          ! Auxiliary-code COMMONs
      include 'auxcommons/geoxyzv.f'
      include 'auxcommons/instuf.f'

      real*8                                              ! Local variables
     * dist,dnearl

      integer
     * irl,irx,iry,irz

      irl = ir(np)

      if (irl .le. 0) then
        write(6,*) 'Stopped in howfar with irl <= 1'
        stop
      end if

      if (irl .eq. 1) then
        idisc = 1  ! ------------------------------------------------
        return     ! Particle outside geometry - return to ELECTR/PHOTON
      end if       ! ------------------------------------------------

!     ---------------------------
!     Get irx, iry and irz indices
!     ---------------------------
      irx=mod(irl-1,imax)
      if (irx.eq.0) irx=imax
      irz=1+(irl-1-irx)/ijmax
      iry=1+(irl-1-irx-(irz-1)*ijmax)/imax

      dnearl = 1.D10

!     -----------------
!     Check Z-direction
!     -----------------
      dnearl=min(dnearl,(zbound(irz+1)-z(np)),(z(np)-zbound(irz)))
```

```
      if (w(np) .gt. 0.0) then
        dist = (zbound(irz+1)-z(np))/w(np)
        if (dist .lt. ustep) then
          ustep=dist
          if (irz .ne. kmax) then
            irnew=irl+ijmax
          else
            irnew=1
          end if
        end if
      else if (w(np) .lt. 0.0) then
        dist = -(z(np) - zbound(irz))/w(np)
        if (dist .lt. ustep) then
          ustep = dist
          if (irz .ne. 1) then
            irnew=irl-ijmax
          else
            irnew = 1
          end if
        end if
      end if

!     -----------------
!     Check X-direction
!     -----------------
      dnearl=min(dnearl,(xbound(irx+1)-x(np)),(x(np)-xbound(irx)))
      if (u(np) .gt. 0.0) then
          dist = (xbound(irx+1)-x(np))/u(np)
          if (dist .lt. ustep) then
              ustep=dist
              if (irx .ne. imax) then
                  irnew=irl+1
              else
                  irnew=1
              end if
          end if
      else if (u(np) .lt. 0.0) then
          dist = -(x(np) - xbound(irx))/u(np)
          if (dist .lt. ustep) then
              ustep = dist
              if (irx .ne. 1) then
                  irnew=irl-1
              else
                  irnew = 1
              end if
          end if
      end if

!     -----------------
!     Check Y-direction
!     -----------------
      dnearl=min(dnearl,(ybound(iry+1)-y(np)),(y(np)-ybound(iry)))
      if (v(np) .gt. 0.0) then
          dist = (ybound(iry+1)-y(np))/v(np)
          if (dist .lt. ustep) then
              ustep=dist
              if (iry .ne. jmax) then
                  irnew=irl+imax
              else
                  irnew=1
              end if
          end if
      else if (v(np) .lt. 0.0) then
          dist = -(y(np) - ybound(iry))/v(np)
          if (dist .lt. ustep) then
              ustep = dist
              if (iry .ne. 1) then
                  irnew=irl-imax
              else
                  irnew = 1
```

```
        end if
    end if
end if

dnear(np)=dnearl

                                                  ! ----------------------
return                                            ! Return to ELECTR/PHOTON
                                                  ! ----------------------
end
```

3　実習課題

3.1　実習課題１：線源を Co-60 に変更する。

線源を Co-60 に変え、1.173MeV と 1.333MeV 光子を同じ確率で発生させる。

3.2　実習課題２：線源を 100kV の X 線に変更する。

線源光子のエネルギーを 100kV の X 線 (スペクトルデータは、xray.dat から読み込み) データを用いてサンプリングする。

3.3　実習課題３：肺のモデルに変更する。

前面から 3cm を通常の人体組織、3-13cm を肺（密度 0.3g/cm^3) とし、その背後に 3cm の人体組織がある体系に変更する。線源は、100kVX 線とする。

3.4　実習課題４：腫瘍を含む肺のモデルに変更する。

肺の前面から 3cm の位置に、厚さ 2cm の腫瘍を設定する。腫瘍の領域の密度を通常の水とする。腫瘍は、X-, Y-方向全域に拡がっていると仮定する。線源は、100kVX 線とする。

3.5　実習課題５：ファントム中に金属を含むモデルに変更する。

厚さ 20cm のファントムにおいて深さ 5cm-6cm の領域を鉄に変える。線源は、100kVX 線とする。

3.6　その他

上記に加えて、以下のような試みも考えられる。

- 線源として、他のエネルギーの X 線を使用する
- 光子だけでなく、電子入射の可能にする
- 挿入した金属の厚さを 1cm と異なる厚さにする
- 腫瘍の面積を限定する

4 実習課題の解答例

比較のために、ucxyz_phantom.f を実行し、計算結果 (egs5job.out, egs5job.pic) を別な名称のファイル名 (例えば、xyz_phantom.out, xyz_phantom.pic) で保存しておく。

4.1 実習課題 1

1. cp ucxyz_phantom.f ucxyz_phantom1.f
2. cp ucxyz_phantom.data ucucxyz_phantom1.data
3. cp ucucxyz_phantom.inp ucucxyz_phantom1.inp
4. ucucxyz_phantom1.f を以下のように修正する。

   - 線源データのための配列を追加する。

     ```
           real*8
          * depeh(LIMAX,LJMAX,LKMAX),depeh2(LIMAX,LJMAX,LKMAX),
          * dose(LIMAX,LJMAX,LKMAX),doseun(LIMAX,LJMAX,LKMAX)
     ```

     を

     ```
           real*8
          * depeh(LIMAX,LJMAX,LKMAX),depeh2(LIMAX,LJMAX,LKMAX),
          * dose(LIMAX,LJMAX,LKMAX),doseun(LIMAX,LJMAX,LKMAX)
          * ,esbin(MXEBIN),espdf(MXEBIN),escdf(MXEBIN)
     ```

     に変更。

   - 線源エネルギーデータの数を示す変数を追加する。

     ```
           integer
          * i,ii,iii,icases,idin,idose,ie,ipage,irl,j,jhist,jj,jl,ju,k,
          * kkk,nlist,nperpg
     ```

     を

     ```
           integer
          * i,ii,iii,icases,idin,idose,ie,ipage,irl,j,jhist,jj,jl,ju,k,
          * kkk,nlist,nperpg,nsebin
     ```

     に変更。

   - 線源データファイルの open 文を追加する。

     ```
           open(6,file='egs5job.out',status='unknown')
     ```

     を

     ```
           open(6,file='egs5job.out',status='unknown')
           open(2,file='co60.inp',status='unknown')
     ```

     に変更。

   - co60.inp は、線源のエネルギーとその確率密度関数で以下の内容のファイルであり、配布ファイルに含まれている。

     ```
     1.173,1.333
     0.5,0.5
     ```

   - 線源データの読み込みと cdf を作成するルーチンの追加。

     ```
     !     Source position from phantom surface in cm.
           sposi=10.0
     ```

     を

```
!     Source position from phantom surface in cm.
      sposi=10.0

      nsebin=2            ! Number of source energy bins
      read(2,*) (esbin(i),i=1,nsebin)
      read(2,*) (espdf(i),i=1,nsebin)
!---------------------------
!     Calculate CDF from pdf
!---------------------------
      tnum=0.D0
      do ie=1,nsebin
        tnum=tnum+espdf(ie)
      end do

      escdf(1)=espdf(1)/tnum
      do ie=2,nsebin
        escdf(ie)=escdf(ie-1)+espdf(ie)/tnum
      end do
```

に変更。

- 線源の最大運動エネルギーの変更。

```
      ekein=1.253         ! Kinetic energy of source photon
```

を

```
      ekein=esbin(nsebin) ! Maximum kinetic energy}
```

に変更する。

- 線源エネルギーのサンプリングルーチンを追加する。

```
        ekin=ekein
```

を

```
        call randomset(rnnow)
        do ie=1,nsebin
          if(rnnow.le.escdf(ie)) go to 1000
        end do
1000    ekin=esbin(ie)
```

に変更。

- 入射エネルギー出力部を以下のように変更する。

```
280   FORMAT(/' Absorbed energy inside phantom for 1.253MeV photon'/
```

を

```
280   FORMAT(/' Absorbed energy inside phantom for Co-60 photon'/
```

に変更。

5. ucxyz_phantom1.f を egs5run で実行する。

   - Linux 又は Cygwin の場合
     ユーザーコード名として、ucxyz_phantom1 を, ユニット 4 及びユニット 25 のファイル名には，何も入力しないでリターンする。
     ”Does this user code read from the terminal?”に対して 1 を入力する。
   - DOS の場合
     egs5run ucxyz_phantom1
   - ucxyz_phantom1 等が、egs5run.bat を実行しているディレクトリーと別なディレクトリーにある場合は、ディレクトリー名を記載する。DOS の場合、ディレクトリーの識別子は、/ ではなく ¥であるので、間違わないように注意する。

6. 計算が終了したら、egs5job.out を調べ、平均エネルギーが 1.253MeV 近くになっていることを確認する。また、各値が 1.253MeV の場合と異なることを確認する。

4.2 実習課題２

1. cp ucxyz_phantom1.f ucxyz_phantom2.f
2. cp ucxyz_phantom1.data ucxyz_phantom2.data
3. cp ucxyz_phantom1.inp ucxyz_phantom2.inp
4. ucxyz_phantom2.f を以下のように修正する。
   - 線源のエネルギービン幅を定義する変数を追加する。

     ```
           real*8 bsfa,bsferr,faexps,faexp2s,faexrr,fexpss,fexps2s,fexerr,
          *       faexpa,fexpsa
     ```

     を

     ```
           real*8 bsfa,bsferr,faexps,faexp2s,faexrr,fexpss,fexps2s,fexerr,
          *       faexpa,fexpsa,deltaes
     ```

     に変更する。
   - サンプリングした線源のスペクトル情報のための変数を追加する。

     ```
           real*8
          * depeh(LIMAX,LJMAX,LKMAX),depeh2(LIMAX,LJMAX,LKMAX),
          * dose(LIMAX,LJMAX,LKMAX),doseun(LIMAX,LJMAX,LKMAX)
          * ,esbin(MXEBIN),espdf(MXEBIN),escdf(MXEBIN)
     ```

     を

     ```
           real*8
          * depeh(LIMAX,LJMAX,LKMAX),depeh2(LIMAX,LJMAX,LKMAX),
          * dose(LIMAX,LJMAX,LKMAX),doseun(LIMAX,LJMAX,LKMAX)
          * ,esbin(MXEBIN),espdf(MXEBIN),escdf(MXEBIN),saspec(MXEBIN)
     ```

     に変更する。
   - 線源情報のファイルを変更する。

     ```
           open(2,file='co60.inp',status='unknown')
     ```

     を

     ```
           open(2,file='xray.dat',status='old')  ! Data of source x-ray
     ```

     に変更。
   - xray.dat は、以下のデータファイルで、配布ファイルに含まれている。

     ```
       201
       0.0005
         0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
         0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
         0.,   15.,  472.,  410.,  595.,  675.,  642.,  477.,
       498.,  492.,  504.,  610.,  611.,  551.,  637.,  702.,
       711.,  994., 1130., 1338., 1618., 1860., 2393., 2887.,
      3250., 3766., 4337., 4972., 5586., 6152., 6849., 7200.,
      8078., 8446., 8850., 9129., 9675.,10419.,11907.,12607.,
     13196.,13542.,13940.,13999.,13922.,13409.,13136.,13141.,
     13594.,13916.,14347.,14525.,14496.,14621.,14658.,14818.,
     14745.,14730.,14589.,14217.,14097.,13794.,13924.,13665.,
     13650.,13430.,13260.,12862.,12587.,12227.,12255.,12117.,
     11551.,11343.,11187.,10859.,10604.,10266.,10085., 9768.,
      9519., 9232., 9147., 8760., 8600., 8263., 8150., 7907.,
      7574., 7296., 7058., 6815., 6769., 6505., 6511., 6279.,
      6160., 6751., 7016., 7988., 8860., 9176., 9348., 9177.,
      7496., 5690., 4512., 4105., 3851., 3574., 3494., 3337.,
      3202., 3115., 3177., 2989., 3326., 3356., 3441., 3403.,
      2873., 2569., 2263., 2008., 1815., 1661., 1490., 1469.,
      1435., 1242., 1210., 1183., 1210., 1104., 1034., 1052.,
     ```

```
922.,  904.,  866.,  842.,  860.,  824.,  726.,  714.,
688.,  600.,  587.,  610.,  497.,  485.,  481.,  395.,
403.,  385.,  334.,  363.,  343.,  348.,  259.,  270.,
247.,  247.,  262.,  207.,  182.,  210.,  194.,  152.,
130.,  114.,  150.,  113.,  139.,   90.,   76.,   59.,
 52.,   34.,   34.,   31.,   11.,   23.,   12.,   12.,
  4.
```

201 は、エネルギービン数、0.0005 は、エネルギービンの幅 (MeV) である。それ以降の数字は、各エネルギービンに対応する X 線の発生数であり、積分した値で割ると確率密度関数となる。エネルギーの最小値は 0.0 としている。

- 線源データの読込部を変更する。

```
nsebin=2            ! Number of source energy bins
read(2,*) (esbin(i),i=1,nsebin)
read(2,*) (espdf(i),i=1,nsebin)
```

を

```
read(2,*) nsebin            ! Number of source energy bins
read(2,*) deltaes           ! Source energy bin width in MeV
read(2,*) (espdf(i),i=1,nsebin)
```

に変更。[3]

- cdf 作成関連部分 (ビン数、エネルギービンに対応するエネルギーの設定、cdf の作成) を変更する。

```
escdf(1)=espdf(1)/tnum
do ie=2,nsebin
  escdf(ie)=escdf(ie-1)+espdf(ie)/tnum
end do
```

を

```
nsebin=nsebin+1
esbin(1)=0.d0
escdf(1)=espdf(1)/tnum
do ie=2,nsebin
  esbin(ie)=(ie-1)*deltaes
  escdf(ie)=escdf(ie-1)+espdf(ie)/tnum
end do
```

に変更。

- サンプリングスペクトル情報を初期化する。

```
fexps2s=0.D0
```

を

```
fexps2s=0.D0

do ie=1,nsebin
  saspec(ie)=0.D0
end do
```

に変更。

- 線源エネルギーのサンプリング部を変更する。

[3] この問題のように、配列の引数となる変数の値を変更する場合には、ユーザーコード完成後に、まずデバッガー機能を含めてコンパイル、実行を行い、配列範囲外アクセスが起きないことを確認するべきである。方法としては、”egs5run” と入力するところで”egs5run db”と入力する。これにより、デバッガー機能を含めたコンパイルが行われる。つぎに”egs5job.exe”と入力して、計算を実行する。配列範囲外アクセスが起きなければ計算は通常通り終了する。(追加的なメッセージはなにも表示されない)　配列範囲外アクセスが起きた場合には、ソースのどの行で、どの配列の何番目の要素に不正なアクセスが行われたかが表示されるので、ソースの当該部分を修正する。なお、デバッガーを含めてコンパイルした場合実行速度が低下するので、デバッガーの使用はプログラム変更の場合のみとする方がよい。

```
        call randomset(rnnow)
        do ie=1,nsebin
          if(rnnow.le.escdf(ie)) go to 1000
        end do
1000    ekin=esbin(ie)
```

を

```
        call randomset(rnnow)
        do ie=1,nsebin
          if(rnnow.le.escdf(ie)) go to 1000
        end do
1000    if (ie.gt.nsebin) then
          ie=nsebin
        end if
        saspec(ie)=saspec(ie)+1.D0
        if (escdf(ie).eq.escdf(ie-1)) then
          ekin=esbin(ie-1)
        else
          ekin=esbin(ie-1)+(rnnow-escdf(ie-1))*(esbin(ie)-esbin(ie-1))/
     *         (escdf(ie)-escdf(ie-1))
        end if
```

に変更。

- 体系に入射したエネルギーチェックのルーチンの後に、サンプリングした線源スペクトルを出力する文を追加する。

```
!-----------------------------
!     Sampled source spectrum
!-----------------------------
```

を

```
!-----------------------------
!     Sampled source spectrum
!-----------------------------
      do ie=2,nsebin
        saspec(ie)=saspec(ie)/float(ncases)
      end do

      write(6,272)
272   FORMAT(/' Comparison between sampled spectrum and pdf'
     *  /23X,'   Sampled      pdf     ',25X,'   Sampled       pdf     '
     *  )
      do ie=2,nsebin,2
        if(ie.eq.nsebin) then
          write(6,274) esbin(ie),saspec(ie),escdf(ie)-escdf(ie-1)
274       FORMAT(1X,G9.3,' MeV(upper)-- ',2G12.5)
        else
          write(6,276) esbin(ie),saspec(ie),escdf(ie)-escdf(ie-1),
     *    esbin(ie+1), saspec(ie+1),escdf(ie+1)-escdf(ie)
276       FORMAT(1X,G9.3,' MeV(upper)-- ',2G12.5,3X, '; ',G9.3,
     *           ' MeV(upper)-- ',2G12.5)
        end if
      end do
```

に変更。

- 線源情報の出力部を変更する。

```
280   FORMAT(/' Absorbed energy inside phantom for Co-60 photon'/
```

を

```
280   FORMAT(/' Absorbed energy inside phantom for 100kV X-ray'/
```

に変更。

5. ucphantomcgv2.inp を変更する。

```
&INP AE=0.521,AP=0.0100,UE=2.011,UP=1.5 /END
```

を

```
&INP AE=0.521,AP=0.0100,UE=0.711,UP=0.2 /END
```

に変更 (2 カ所)。

6. ucphantomcgv2.f を egs5run で実行する。

   - Linux 又は Cygwin の場合
     ユーザーコード名として、ucxyz_phantom2 を, ユニット 4 及びユニット 25 のファイル名には，何も入力しないでリターンする。
     ”Does this user code read from the terminal?”に対して 1 を入力する。
   - DOS の場合
     egs5run ucxyz_phantom2

7. 計算が終了したら、egs5job.out を調べ、平均エネルギーがおおよそ 40keV になっていることを確認する。また、サンプリングされた線源スペクトルと、線源スペクトルの pdf を比較する。

8. CGView を使用して、phantom.pic との飛跡の違いを確認する。

4.3 実習課題３

1. cp ucxyz_phantom2.data ucxyz_phantom3.data

2. cp ucxyz_phantom2.inp ucxyz_phantom3.inp

3. ucxyz_phantom3.data を以下のように修正する。

   (a) Z-方向のボクセル数を変更する。

   ```
   1.0,    20                      voxel width, number of voxels
   ```

   を

   ```
   1.0,    16                      voxel width, number of voxels
   ```

   に変更する。

   (b) リージョンへの物質指定等を変更する。

   ```
   1,3,1,3, 2,21,  1,  0.000, 0.00, 0.00        tissue
     1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
   1,3,1,3,22,22,   2,  0.00,  0.00, 0.00       air
     1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
   ```

   を

   ```
   1,3,1,3, 2, 4,  1,  0.000, 0.00, 0.00         tissue
     1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
   1,3,1,3, 5,14,  1,  0.300, 0.00, 0.00         lung
     1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
   1,3,1,3,15,17,  1,  0.000, 0.00, 0.00         tissue
     1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
   1,3,1,3,18,18,   2,  0.00,  0.00, 0.00        air
     1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
   ```

   に変更する。

4. run5again を実行する。

- Linux の場合
  ユニット 4 のファイル名として ucxyz_phantom3 を入力し、ユニット 25 のファイル名は、何も入力しないでリターンする。
  Enter name of the executable に、何も入力しないでリターンする。
- DOS の場合
  run5again ucxyz_phantom3

5. 計算が終了したら、egs5job.out を調べ、肺の領域の密度が設定通りになっていることを確認する。また、線量分布が一様なファントムの場合と異なることを確認する。

4.4 実習課題４

1. cp ucxyz_phantom3.data ucxyz_phantom4.data
2. cp ucxyz_phantom3.inp ucxyz_phantom4.inp
3. ucxyz_phantom4.data を以下のように修正する。

   (a) リージョンへの物質指定等を変更する。

```
1,3,1,3, 5,14,  1,  0.300, 0.00, 0.00         lung
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
```

   を

```
1,3,1,3, 5, 7,  1,  0.300, 0.00, 0.00         lung
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
1,3,1,3, 8, 9,  1,  0.000, 0.00, 0.00         tumor
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
1,3,1,3,10,14,  1,  0.300, 0.00, 0.00         lung
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
```

   に変更する。

4. run5again を実行する。
   - Linux の場合
     ユニット 4 のファイル名として ucxyz_phantom4 を入力し、ユニット 25 のファイル名は、何も入力しないでリターンする。
     Enter name of the executable に、何も入力しないでリターンする。
   - DOS の場合
     run5again ucxyz_phantom4
5. 計算が終了したら、egs5job.out を調べ、腫瘍のヶ所の密度が設定通りになっていることを確認する。また、線量分布が一様なファントムの場合と異なることを確認する。

4.5 実習課題５

1. cp ucxyz_phantom4.data ucxyz_phantom5.data
2. cp ucxyz_phantom4.inp ucxyz_phantom5.inp
3. ucxyz_phantom4.data を以下のように修正する。

   (a) 物質データ数を’2’ から ’3’ に変更する。

```
          2                       nmed (I10)
```

   を

```
          3                       nmed (I10)
```

に変更する。

(b) 物質 (鉄) を追加する。

```
AIR-AT-NTP                      media(j,2) (24A1)
```

を

```
AIR-AT-NTP                      media(j,2) (24A1)
FE                              media(j,3) (24A1)
```

を変更する。

(c) 鉄の characteric dimension を追加する。

```
   1.0,   1.0                   chard
```

を

```
   1.0,   1.0, 1.0              chard
```

に変更する。

(d) リージョンへの物質指定等を変更する。

```
1,3,1,3, 2, 4,  1,  0.000, 0.00, 0.00          tissue
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
1,3,1,3, 5, 7,  1,  0.300, 0.00, 0.00          lung
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
1,3,1,3, 8, 9,  1,  0.000, 0.00, 0.00          tumor
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
1,3,1,3,10,14,  1,  0.300, 0.00, 0.00          lung
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
1,3,1,3,15,17,  1,  0.000, 0.00, 0.00          tissue
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
```

を

```
1,3,1,3, 2, 6,  1,  0.000, 0.00, 0.00          tissue
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
1,3,1,3, 7, 7,  3,  0.000, 0.00, 0.00          iron
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
1,3,1,3, 8,21,  1,  0.000, 0.00, 0.00          tissue
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
```

に変更する。

4. ucxyz_phantom5.inp を以下のように変更する。

(a) 以下の鉄のデータを追加する。

```
ELEM
 &INP IRAYL=1 /END
FE                              FE
FE
ENER
 &INP AE=0.521,AP=0.010,UE=2.011,UP=1.5 /END
PWLF
 &INP  /END
DECK
 &INP  /END
```

5. run5again を実行する。

- Linux の場合
  ユニット 4 のファイル名として ucxyz_phantom5 を入力し、ユニット 25 のファイル名は、何も入力しないでリターンする。
  Enter name of the executable に、何も入力しないでリターンする。

- DOS の場合
  `run5again ucxyz_phantom5`

6. 計算が終了したら、egs5job.out を調べ、腫瘍のヶ所の密度が設定通りになっていることを確認する。また、線量分布が一様なファントムの場合と異なることを確認する。

egs5 sample user code (ucxyz_phantom.f)
Dose distribution calculation inside phantom with Voxel

(English Parts)

1 Outlines of sample user code ucxyz_phantom.f

ucxyz_phantom.f is the egs5 user code to perform following calculations.

1. Geometry (Fig. 1)
   - 3-dimensional volume element (voxel) geometry
   - number of z-direction bin 22
   - number of y-direction bin 3
   - number of x-direction bin 3
   - phantom is modeled with water of 30cmx30cm area and 20cm depth.
   - 5cm air region exits at before and after phantom.

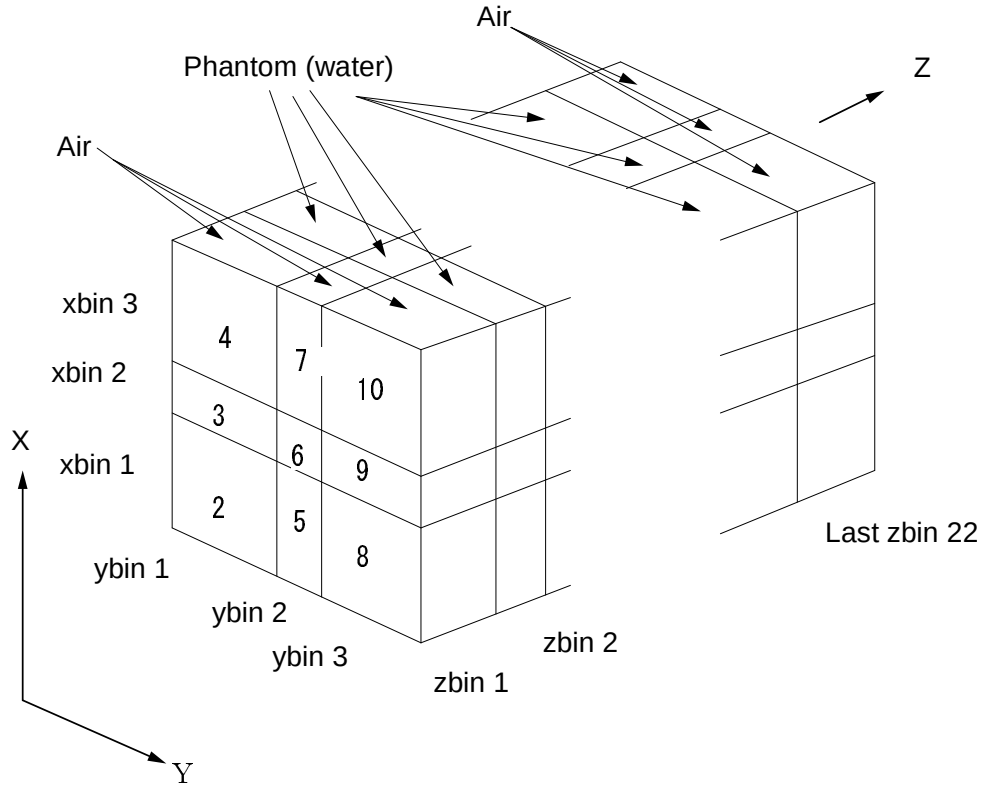

Figure 1: geometry of ucxyz_phantom.f.

2. Source conditions
   - Source photon energy is 1.253 MeV.
   - A point isotropic source exits at sposi(=10cm) from a phantom surface.
   - Half-beam size at the phantom surface for x-direction is xhbeam(=1cm) and y-direction is yhbeam(=1cm).
3. Results obtained
   (a) Data of information of particle trajectories (`egs5job.pic`)
   (b) Calculated result (`egs5job.out`)

- Information of material used
- Material assignment to each region
- Plane data defined
- Dose distributions and their uncertainties at central phantom (1cm × 1cm) area
- Back scattering factor at the phantom surface (1cm × 1cm area at the phantom center)
- Dose distributions inside the phantom and their uncertainties.

2 Details of user code

2.1 Main program: Step 1

2.1.1 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 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.

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

include 'include/egs5_bounds.f'
include 'include/egs5_edge.f'
include 'include/egs5_elecin.f'
include 'include/egs5_media.f'
include 'include/egs5_misc.f'
include 'include/egs5_switches.f'
include 'include/egs5_stack.f'
include 'include/egs5_thresh.f'
include 'include/egs5_uphiot.f'
include 'include/egs5_useful.f'
include 'include/randomm.f'
```

`include 'include/egs5_h.f'` is always necessary. Other parts are only necessary when variables including at each common are used inside the main program.[1]

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

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

include 'user_auxcommons/edata.f'
include 'user_auxcommons/etaly1.f'
include 'user_auxcommons/geoxyzv.f'
include 'user_auxcommons/instuf.f'
include 'user_auxcommons/lines.f'
include 'user_auxcommons/nfac.f'
include 'user_auxcommons/watch.f'
```

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

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

```
      common/score/                                ! Variables to score
     *            depe(LIMAX,LJMAX,LKMAX),faexp,fexps,maxpict
      real*8 depe,faexp,fexps
      integer maxpict
```

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

2.1.2 `open` statement

At the top of executable statement, it is necessary to open units used in the user code. Due to the new feature that pegs is called inside each user code, it must be careful to the unit number used. The unit number from 7 to 26 are used inside 'pegs' and close at the end of 'pegs'. These units, therefore, must be re-open after calling pegs. It is better not to use these unit in the user code. The unit used in the subroutine 'plotxyz' and 'geomout' used to keep and output trajectory information is '39' for this reason.

```
!-----------------------------------------------------------------
!     Units 7-26 are used in pegs and closed.  It is better not
!     to use as output file. If they are used must be re-open after
!     getvoxel 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')
```

2.2 Step 2: Pegs5-call

Call subroutine `getvoxel` (name of subroutine and its function is different depending on each user code) which read material number, material name, their characteristic dimensions, plane data for voxels and assignment of material and various option flag etc. from unit 4.

`Subroutine pegs5` is called after above setting.

```
!-----------------------------------------------
!     Define pict data mode.
!-----------------------------------------------
      ifto = 6      ! Output unit in getvoxel
!     ====================
      call getvoxel(ifto)
!     ====================

!     ------------------------------
!     Run PEGS5 before calling HATCH
!     ------------------------------
      write(6,*) ' PEGS5-call comes next'

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

2.3 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. As mentioned before, this user code has 2 calculation mode. Output geometry data and material assignments to the trajectory display data file.

```
!-----------------------------------------------
!     Define pict data mode.
!-----------------------------------------------
!     npreci 1: for PICT32
!            2: for CGview
!            3: for CGview in free format
      npreci=3
```

```
      if(npreci.eq.3) write(39,fmt="('GSTA-FREE-TIME')")
      if(npreci.eq.2) write(39,fmt="('GSTA-TIME')")
      write(39,fmt="('SLAB')")
      write(39,fmt="(I6)") imax+1
      write(39,fmt="(I6)") jmax+1
      write(39,fmt="(I6)") kmax+1
      write(39,fmt="(4F15.4)") (xbound(j),j=1,imax+1)
      write(39,fmt="(4F15.4)") (ybound(j),j=1,jmax+1)
      write(39,fmt="(4F15.4)") (zbound(j),j=1,kmax+1)
      write(39,fmt="('GEND')")

      write(39,fmt="('MSTA')")
      write(39,fmt="(I4)") nreg
      write(39,fmt="(15I4)") (med(i),i=1,nreg)
      write(39,fmt="('MEND')")
```

Next initialize the Ranlux random number generator.

```
!     ------------------------------------------------------
!     Random number seeds.  Must be defined before call hatch
!     or defaults will be used.  inseed (1- 2^31)
!     ------------------------------------------------------
      luxlev = 1
      inseed=1
      write(6,100) inseed
100   FORMAT(/,' inseed=',I12,5X,
     *         ' (seed for generating unique sequences of Ranlux)')

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

2.4 Step 4: Determination-of-incident-particle-parameters

Various source parameters like energy, position (distance from a phantom surface) and a half width at phantom surface are set.

```
!-------------------------------------------------
!     Define source position from phantom surface.
!-------------------------------------------------
!     Source position from phantom surface in cm.
      sposi=10.0

      iqin=0             ! Incident charge - photons
      ekein=1.253        ! Kinetic energy of source photon
      etot=ekein + abs(iqin)*RM
      xin=0.D0
      yin=0.D0
      zin=-sposi
      uin=0.D0
      vin=0.D0
      win=1.D0
      wtin=1.D0

!-----------------------------------------
!     Half width and height at phantom surface
!-----------------------------------------
!     X-direction half width of beam at phantom surface in cm.
      xhbeam=1.0
!     Y-direction half height of beam at phantom surface in cm.
      yhbeam=1.0
      radma2=xhbeam*xhbeam+yhbeam*yhbeam
      wimin=sposi/dsqrt(sposi*sposi+radma2)
```

2.5 Step 5: hatch-call

Maximum total energy of electrons is defined as `emaxe`, and then `subroutine hatch` is called.

Output the material data and parameters of each region to the result file (unit 6).

```
! Define possible maximum total energy of electron before hatch
      emaxe = ekein + RM

      write(6,110)
110   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(6,120)
120   FORMAT(/,' HATCH-call comes next',/)

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

2.6 Step 6: Initialization-for-howfar

Define various parameters used for the geometry definition in this step. In this user code, this part is done at subroutine `getvoxel`.

2.7 Step 7: Initialization-for-ausgab

Initialize various variables to be calculated and set the number of detectors and a number of histories to calculate (`ncases`) and to store trajectory data (`maxpict`).

```
      ncount = 0
      ilines = 0
      nwrite = 10
      nlines = 25
      idin = -1
      totke = 0.
      wtsum = 0.

!---------------------
!     Clear variables
!---------------------
!     Zero the dose
      do k=1,kmax
        do j=1,jmax
          do i=1,imax
            depe(i,j,k)=0.D0
            depeh(i,j,k)=0.D0
            depeh2(i,j,k)=0.D0
          end do
        end do
      end do

      faexp=0.D0
      faexps=0.D0
      faexp2s=0.D0
      fexps=0.D0
      fexpss=0.D0
      fexps2s=0.D0

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

!-------------------------
!     History number
!-------------------------
!     History number
      ncases=100000
```

```
!     Maximum history number to write trajectory data
      maxpict=50

      write(39,fmt="('0    1')")
```

2.8 Step 8: Shower-call

In this part, `subroutine shower` is called 'ncases' (history number). Before calling `shower`, various source parameters are sampled. In this used code, it is supposed that a point isotropic point source exits at `sposi` cm from the phantom surface. If `sposi` is larger than 5cm (air thickness in front of the phantom), starting source position at the surface of air region is determined considering the beam width at the phantom surface.

At each history, energy balance between the kinetic energy of source and absorbed energy in all region defined. Summation of weight squared of variables to be calculated together with spectrum information are also stored for statistical analysis.

```
                                              ! -------------------------
      do jhist=1,ncases                       ! Start of CALL SHOWER loop
                                              ! -------------------------
        icases=jhist
!-------------------------------------
!       Determine direction (isotropic)
!-------------------------------------
200     call randomset(w0)
        win=w0*(1.0-wimin)+wimin
        call randomset(phai0)
        phai=pi*(2.0*phai0-1.0)
        sinth=dsqrt(1.D0-win*win)
        uin=dcos(phai)*sinth
        vin=dsin(phai)*sinth
        dis=sposi/win
        xpf=dis*uin
        ypf=dis*vin
        if (dabs(xpf).gt.xhbeam.or.dabs(ypf).gt.yhbeam) go to 200
        if (sposi.gt.zbound(2)-zbound(1)) then
          disair=(sposi-(zbound(2)-zbound(1)))/win
          xin=disair*uin
          yin=disair*vin
          zin=zbound(1)
        else
          xin=0.D0
          yin=0.D0
          zin=-sposi
        end if

        do i=1,imax
            if (xbound(i+1).gt.xin) go to 210
        end do

210     do j=1,jmax
          if (ybound(j+1).gt.yin) go to 220
        end do

!       -------------
!       Input region
!       -------------
220     k=1
        irin=1+i+(j-1)*imax

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

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

```
        totke = totke + availke                 ! Keep running sum of KE

        latchi=0

!       ------------------------------------------------
!       Print first NWRITE or NLINES, whichever comes first
!       ------------------------------------------------
        if (ncount .le. nwrite .and. ilines .le. nlines) then
          ilines = ilines + 1
          write(6,230) etot,xin,yin,zin,uin,vin,win,iqin,irin,idin
230       FORMAT(4G15.7/3G15.7,3I5)
        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 normarization 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    ' normarized.',3e12.5)")uin,vin,win
          stop
        end if

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

!---------------------------------
!      Sum variable and its square.
!---------------------------------

        do k=1,kmax
          do j=1,jmax
            do i=1,imax
              depeh(i,j,k)=depeh(i,j,k)+depe(i,j,k)
              depeh2(i,j,k)=depeh2(i,j,k)+depe(i,j,k)*depe(i,j,k)
              depe(i,j,k)=0.D0
            end do
          end do
        end do

        faexps=faexps+faexp
        faexp2s=faexp2s+faexp*faexp
        faexp=0.0
        fexpss=fexpss+fexps
        fexps2s=fexps2s+fexps*fexps
        fexps=0.0

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

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

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

2.8.1 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:

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

2.8.2 Step 9: Output of results

Obtained results from `ncases` histories are analyzed and outputted in this part.

```
      write(1,280) sposi
280   FORMAT(/' Absorbed energy inside phantom for 1.253MeV photon'/
     *' Source position ',F10.1,' cm from phantom surface'/
     *' Within 1cm x 1 cm area after 5 cm air')

      write(1,290) ncases, xhbeam, yhbeam
290   FORMAT(1X,I8,' photons normally incident from front side'/
     *' Half width of beam is ',G15.5,'cm for X and ',G15.5,'cm for Y')

!      ---------------------------------------
!      Calculate average and its uncertainties
!      ---------------------------------------

        do k=1,kmax
          do j=1,jmax
            do i=1,imax
              irl=1+i+(j-1)*imax+(k-1)*ijmax
              amass=(xbound(i+1)-xbound(i))*
     *              (ybound(j+1)-ybound(j))*
     *              (zbound(k+1)-zbound(k))*rhor(irl)
              dose(i,j,k)=depeh(i,j,k)/ncases
              depeh2(i,j,k)=depeh2(i,j,k)/ncases
              doseun(i,j,k)=dsqrt((depeh2(i,j,k)-
     *                       dose(i,j,k)*dose(i,j,k))/ncases)
              dose(i,j,k)=dose(i,j,k)*1.602D-10/amass
              doseun(i,j,k)=doseun(i,j,k)*1.602D-10/amass
            end do
          end do
        end do

!-----------------------------------------------
!     Print out the results of central phantom
!-----------------------------------------------
      i=imax/2+1
      j=jmax/2+1
      do kkk=2,kmax-1
        depths=zbound(kkk)
        depthl=zbound(kkk+1)
        irl=1+i+(j-1)*imax+(kkk-1)*ijmax
        write(6,300) depths,depthl,(media(ii,med(irl)),ii=1,24),
```

```
     *  rhor(irl),dose(i,j,kkk),doseun(i,j,kkk)
300     FORMAT(' At ',F4.1,'--',F4.1,'cm (',24A1,',rho:',F8.4,')=',
     *  G13.5,'+-',G13.5,'Gy/incident')
      end do

!-------------------------------------------------
!     Calculate average exposure and its deviation
!-------------------------------------------------
      area=(xbound(i+1)-xbound(i))*(ybound(j+1)-ybound(j))
      faexpa=faexps/ncases
      faexp2s=faexp2s/ncases
      faexrr=dsqrt((faexp2s-faexpa*faexpa)/ncases)
      faexpa=faexpa*1.6E-10/area
      faexrr=faexrr*1.6E-10/area
      fexpsa=fexpss/ncases
      fexps2s=fexps2s/ncases
      fexerr=dsqrt((fexps2s-fexpsa*fexpsa)/ncases)
      fexpsa=fexpsa*1.6E-10/area
      fexerr=fexerr*1.6E-10/area
      if (faexpa.gt.0.0) then
        bsfa=fexpsa/faexpa
        bsferr=bsfa*dsqrt((faexrr/faexpa)**2.+(fexerr/fexpsa)**2.)
        write(6,310) faexpa,faexrr,fexpsa,fexerr,bsfa,bsferr
310     FORMAT(/' Exposure in free air (using mu_en) =', G15.5,'+-',G15.
     *  5,' Gy/incident'/ ' Exposure at phantom surface (using mu_en) ='
     *  , G15.5,'+-',G15.5,'Gy/incident'/ ' Backscattering factor =',G15
     *  .5,'+-',G15.5)
      else
        write(6,320) faexpa,faexrr,fexpsa,fexerr
320     FORMAT(/' Exposure in free air (using mu_en) =', G15.5,'+-',G15.
     *  5,' Gy/incident'/ ' Exposure at phantom surface (using mu_en) ='
     *  , G15.5,'+-',G15.5,'Gy/incident')
      end if
```

The average absorbed dose and its uncertainty at each voxel are calculated. The depth distribution at the central area of the phantom and back scattering factor obtained from exposure at the phantom surface with and without phantom are printed.

The scan data at each Z- or X-bin which is defined in subroutine `getvoxel` are also printed in the dose calculation mode.

2.9 Subroutine getvoxel

Subroutine `getvoxel` is used to define material used, its density, egs5 cut-off energy, various optional flag applied to each region, data for voxel geometry related etc. and call subroutine `hatch`.

The data read from unit 4 are as follows.

1. Record 1 : Title (within 80 characters)
2. Record 2 : Number of media in problem (nmed)
3. Record 3 : Media names (j=1,24, i=1,nmed lines)
4. Record 4 : Characteristic dimension for each material
5. Record 5 : Number of voxel in the X-, Y- and Z-directions (maxx,maxy,maxz). If < 0, it means that number of equally spaced boundaries will be input.
6. Record 6 : xbound
   i.e. repeat the following replacing (i and x), (j and y) and (k and z) respectively.
   - if maxx > 0 input, one per line, the maxx + 1 x boundaries. if maxx is larger than LIMAX, stop program after output warning information. If you want use larger maxx than default value of LIMAX (=22), you must modify LIMAX in `auxcommon/au_h.f`.

- if maxy < 0 input smallest x boundary, followed by abs(maxx) pairs one per line: voxl width, # voxls with this width.

  If maxx finally set from input data, is larger than LIMAX, stop program after output warning information. If you want use larger maxx than default value of LIMAX (=22), you must modify LIMAX in `auxcommon/au_h.f`.

7. Record 7 : ybound
8. Record 8 : zbound
9. Record 9 : Set density, ecut, pcut and various options (0: off, 1:on) to all regions supposing medium is 1.

ipeangsw	Switches for PE-angle sampling
iedgeflsw	K & L-edge fluorescence
iaugersw	K & L-edge Auger
iraysw	Rayleigh scattering
ipolarsw	Linearly-polarized photon scattering
incohrsw	S/Z rejection
iprofrsw	Doppler broadening
mpacrsw	electron impact ionization

10. Record 10 : Line is repeated until a blank line found.
    For all voxels with $il \leq i \leq iu$, $jl \leq j \leq ju$, $kl \leq k \leq ku$ the medium used is medtmp and the density used is rhotmp. If rhotmp=0.0, the default value for that medium is used (faster than entering default density here). If iu and il are zero, it means the end of define.
11. Record 10a:If medium not 0, options are set to the regions above.
    (0: off, 1:on)
12. Record 11 : Regions for which the dose will be output.
    IZSCAN non-zero to get z-scan per page, otherwise output is an x-scan per page.
13. Record 12 : Switch for tracking events with swatch:
    (0=No, 1=each interaction, 2=each step)
14. Record 13 : Switches for bremsstrahlung and pair production ANGLE SAMPLING, and brems-strahlung SPLITTING:

ibrdst=0	No (use default: theta=m/E)
ibrdst=1	Yes (recommended)
iprdst=0	No (use default: theta=m/E)
iprdst=1	1 Yes (low-order distribution)
iprdst=2	2 Yes (recommended)
ibrspl0=0	No splitting
ibrspl0=1	Apply splitting (nbrspl=splitting factor)

2.10 `Subroutine ausgab`

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

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

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

When $iarg < 5$, absorbed energy at the region 1 (outside the system) and other regions are summed separately to check energy balance at each history. If region is not 1, absorbed energy per step is added to that at the region of current particle exits.

If photon crosses the phantom surface at the central region, energy absorption of air is calculated from energy fluence of photon and mass attenuation coefficient of air. Energy absorption of air without phantom is corresponding those by photons never scattered backward. For this purpose, `latch(np))` is set to 1 if `w(np) < 0`.

If a history number is less than `maxpict`, `subroutine plotxyz` which is record and output trajectory related information is called.

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

!  --------------------------------
!  Keep track of how deep stack gets
!  --------------------------------
        if (np.gt.MXSTACK) then
          write(6,100) np,MXSTACK
100       FORMAT(//' In AUSGAB, np=',I3,' >= maximum stack',
     *         ' allowed which is',I3/1X,79('*')//)
          stop
        end if
!     ------------------------
!     Set some local variables
!     ------------------------
      irl = ir(np)
      iql = iq(np)
      edepwt = edep*wt(np)

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

!     ----------------------------------------------------------
!     Keep track of energy deposition (for conservation purposes)
!     ----------------------------------------------------------
      if (iarg .gt. 5) return

      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

      i=mod(irl-1,imax)
      if (i.eq.0) i=imax
      k=1+(irl-1-i)/ijmax
      j=1+(irl-1-i-(k-1)*ijmax)/imax

      if (irl.gt.1.and.edep.ne.0.D0) then

        depe(i,j,k)=depe(i,j,k)+edepwt
      end if

!--------------------------------
!     Check cross phantom surface
!--------------------------------
      if(i.eq.imax/2+1.and.j.eq.jmax/2+1) then  ! X-Y central region
        if (abs(irl-irold).eq.ijmax.and.iq(np).eq.0) then
          if ((w(np).gt.0.0.and.k.eq.2).or.
     *     (w(np).le.0.0.and.k.eq.1)) then
           if (dabs(w(np)).ge.0.0349) then
             cmod=dabs(w(np))
           else
             cmod=0.01745
           end if
           esing=e(np)
           dcon=encoea(esing)            ! PHOTX data
           fexps=fexps+e(np)*dcon*wt(np)/cmod
           if (w(np).lt.0.0) latch(np)=1
```

```
            if (w(np).gt.0.0.and.latch(np).eq.0) then
              faexp=faexp+e(np)*dcon*wt(np)/cmod
            end if
          end if
        end if
      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),
     *       w(np),time(np))
      end if

      return

      end
```

2.11 Subroutine howfar

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

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

Calculation to a distance to the boundary is done by the general treatment for voxel geometry in `ucxyz_phantom.f`.

3 Exercise problems

3.1 Problem 1 : Change source energy

Change source energy to 1.173 and 1.332 MeV photons from ^{60}Co.

3.2 Problem 2 : Change source energy to 100kV X-rays. (Spectrum data are read from xray.dat)

3.3 Problem 3 : Change to lung model

Set surface 3 cm of phantom as the normal tissue (water), 3 to 13 cm as the lung (water with 0.3 g cm^{-3}) and 13-16cm as the normal tissue.
Source is the X-ray read from `xray.dat)`.

3.4 Problem 4 : Lung with tumor

`Set tumor region at 3 to 5cm from the lung surface as the normal tissue.`

3.5 Problem 5 : Inset iron inside phantom

`Replace 5 to 6 cm region of the phantom with iron.`

3.6 Other problems

`In addition above, following problems are also useful as exercises.`

- `Use other X-ray sources`
- `Change incident particle to an electron`
- `Change thickness of iron`
- `Calculate for limited area of tumor`

4 Answer for exercises

It is recommended to run ucxyz_phantom.f and to save egs5job.out, egs5job.pict which are the results with different file names like xyz_phantom.out, xyz_phantom.pict for comparisons with the results of following problems.

4.1 Problem 1

1. cp ucxyz_phantom.f ucxyz_phantom1.f
2. cp ucxyz_phantom.data ucxyz_phantom1.data
3. cp ucxyz_phantom.inp ucxyz_phantom1.inp
4. Modify ucxyz_phantom1.f as follows:

   - Add variables for source data.
     Chanfge

```
      real*8
     * depeh(LIMAX,LJMAX,LKMAX),depeh2(LIMAX,LJMAX,LKMAX),
     * dose(LIMAX,LJMAX,LKMAX),doseun(LIMAX,LJMAX,LKMAX)
```

     to

```
      real*8
     * depeh(LIMAX,LJMAX,LKMAX),depeh2(LIMAX,LJMAX,LKMAX),
     * dose(LIMAX,LJMAX,LKMAX),doseun(LIMAX,LJMAX,LKMAX)
     * ,esbin(MXEBIN),espdf(MXEBIN),escdf(MXEBIN)
```

   - Add variable for a number of source energy data.
     Change

```
      integer
     * i,ii,iii,icases,idin,idose,ie,ipage,irl,j,jhist,jj,jl,ju,k,
     * kkk,nlist,nperpg
```

     t を

```
      integer
     * i,ii,iii,icases,idin,idose,ie,ipage,irl,j,jhist,jj,jl,ju,k,
     * kkk,nlist,nperpg,nsebin
```

   - Add open statement for source data file.
     Change

```
      open(6,file='egs5job.out',status='unknown')
```

     to

```
      open(6,file='egs5job.out',status='unknown')
      open(2,file='co60.inp',status='unknown')
```

   - co60.inp is the data file including source gamma-ray energies and their pdf for Co-60 as follows:

```
1.173,1.333
0.5,0.5
```

   - Add statements to read source data and to create cdf from pdf data.
     Change

```
!     Source position from phantom surface in cm.
      sposi=10.0
```

     to

```
!     Source position from phantom surface in cm.
      sposi=10.0

      nsebin=2            ! Number of source energy bins
      read(2,*) (esbin(i),i=1,nsebin)
      read(2,*) (espdf(i),i=1,nsebin)
!---------------------------
!     Calculate CDF from pdf
!---------------------------
      tnum=0.D0
      do ie=1,nsebin
        tnum=tnum+espdf(ie)
      end do

      escdf(1)=espdf(1)/tnum
      do ie=2,nsebin
        escdf(ie)=escdf(ie-1)+espdf(ie)/tnum
      end do
```

- Modify the maximum electron kinetic energy used.

  Change

```
      ekein=1.253        ! Kinetic energy of source photon
```

  to

```
      ekein=esbin(nsebin) ! Maximum kinetic energy}
```

- Add sampling routines for source photon energy sampling. Change

```
        ekin=ekein
```

  to

```
        call randomset(rnnow)
        do ie=1,nsebin
          if(rnnow.le.escdf(ie)) go to 1000
        end do
1000    ekin=esbin(ie)
```

- Modify output statement concerning the source energy.

  Change

```
280   FORMAT(/' Absorbed energy inside phantom for 1.253MeV photon'/
```

  to

```
280   FORMAT(/' Absorbed energy inside phantom for Co-60 photon'/
```

5. Run ucxyz_phantom1.f by egs5run.

   - In the case of Linux or Cygwin
     Enter ucxyz_phantom1 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 ucxyz_phantom1

6. Check egs5job.out to confirm average source energy is nearly equal to 1.253MeV. Compare the obtained results with xyz_phantom.out.

4.2 Problem 2

1. cp ucxyz_phantom1.f ucxyz_phantom2.f
2. cp ucxyz_phantom1.data ucxyz_phantom2.data
3. cp ucxyz_phantom1.inp ucxyz_phantom2.inp

4. Modify ucxyz_phantom2.f as follows:

- Add variable for a source energy bin.
  Change

```
      real*8 bsfa,bsferr,faexps,faexp2s,faexrr,fexpss,fexps2s,fexerr,
     *       faexpa,fexpsa
```

  to

```
      real*8 bsfa,bsferr,faexps,faexp2s,faexrr,fexpss,fexps2s,fexerr,
     *       faexpa,fexpsa,deltaes
```

- Add variable to score a sampled source spectrum.
  Change

```
      real*8
     * depeh(LIMAX,LJMAX,LKMAX),depeh2(LIMAX,LJMAX,LKMAX),
     * dose(LIMAX,LJMAX,LKMAX),doseun(LIMAX,LJMAX,LKMAX)
     * ,esbin(MXEBIN),espdf(MXEBIN),escdf(MXEBIN)
```

  to

```
      real*8
     * depeh(LIMAX,LJMAX,LKMAX),depeh2(LIMAX,LJMAX,LKMAX),
     * dose(LIMAX,LJMAX,LKMAX),doseun(LIMAX,LJMAX,LKMAX)
     * ,esbin(MXEBIN),espdf(MXEBIN),escdf(MXEBIN),saspec(MXEBIN)
```

- Modify a file name for source.
  Change

```
      open(2,file='co60.inp',status='unknown')
```

  to

```
      open(2,file='xray.dat',status='old')  ! Data of source x-ray
```

- xray.dat is a file including following data.

```
  201
  0.0005
    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
    0.,   15.,  472.,  410.,  595.,  675.,  642.,  477.,
  498.,  492.,  504.,  610.,  611.,  551.,  637.,  702.,
  711.,  994., 1130., 1338., 1618., 1860., 2393., 2887.,
 3250., 3766., 4337., 4972., 5586., 6152., 6849., 7200.,
 8078., 8446., 8850., 9129., 9675.,10419.,11907.,12607.,
13196.,13542.,13940.,13999.,13922.,13409.,13136.,13141.,
13594.,13916.,14347.,14525.,14496.,14621.,14658.,14818.,
14745.,14730.,14589.,14217.,14097.,13794.,13924.,13665.,
13650.,13430.,13260.,12862.,12587.,12227.,12255.,12117.,
11551.,11343.,11187.,10859.,10604.,10266.,10085., 9768.,
 9519., 9232., 9147., 8760., 8600., 8263., 8150., 7907.,
 7574., 7296., 7058., 6815., 6769., 6505., 6511., 6279.,
 6160., 6751., 7016., 7988., 8860., 9176., 9348., 9177.,
 7496., 5690., 4512., 4105., 3851., 3574., 3494., 3337.,
 3202., 3115., 3177., 2989., 3326., 3356., 3441., 3403.,
 2873., 2569., 2263., 2008., 1815., 1661., 1490., 1469.,
 1435., 1242., 1210., 1183., 1210., 1104., 1034., 1052.,
  922.,  904.,  866.,  842.,  860.,  824.,  726.,  714.,
  688.,  600.,  587.,  610.,  497.,  485.,  481.,  395.,
  403.,  385.,  334.,  363.,  343.,  348.,  259.,  270.,
  247.,  247.,  262.,  207.,  182.,  210.,  194.,  152.,
  130.,  114.,  150.,  113.,  139.,   90.,   76.,   59.,
   52.,   34.,   34.,   31.,   11.,   23.,   12.,   12.,
    4.
```

At the above data, a first 201 is the number of energy bins and next 0.0005 is the energy bin width in MeV. Following numbers corresponds to number of X-rays per energy bin. The lower energy corresponding the first bin is 0.0.

- Modify the parts of data read.

  Change

```
      nsebin=2            ! Number of source energy bins
      read(2,*) (esbin(i),i=1,nsebin)
      read(2,*) (espdf(i),i=1,nsebin)
```

  to

```
      read(2,*) nsebin            ! Number of source energy bins
      read(2,*) deltaes           ! Source energy bin width in MeV
      read(2,*) (espdf(i),i=1,nsebin)
```

[2]

- Modify the number of cdf bin. [3]

  Change

```
      escdf(1)=espdf(1)/tnum
      do ie=2,nsebin
        escdf(ie)=escdf(ie-1)+espdf(ie)/tnum
      end do
```

  to

```
      nsebin=nsebin+1
      esbin(1)=0.d0
      escdf(1)=espdf(1)/tnum
      do ie=2,nsebin
        esbin(ie)=(ie-1)*deltaes
        escdf(ie)=escdf(ie-1)+espdf(ie)/tnum
      end do
```

- Initialize sampled X-ray spectrum.

  Change

```
      fexps2s=0.D0
```

  to

```
      fexps2s=0.D0

      do ie=1,nsebin
        saspec(ie)=0.D0
      end do
```

[2]If it is necessary to change the value of the argument, you must check "out of bound" error by running egs5 with debug option as foolows. In the case of Unix or Cygwin, key in "egs5run db" instead of "egs5run". Next, execute programe by "egs5job.exe". In the case of DOS, execute "egs5run_db ucxyz_phantom2".

Modify `egs5run.bat` to use debug mode and save as `egs5run_db.bat` for this purpose.
If "out of bound error" occurs, the line number and the argument coused error is displayed. Running egs5 with debug option needs mor CPU time than usual way and therefore is used only for this kind of check.

[3]If it is necessary to change the value of the argument, you must check "out of bound" error by running egs5 with debug option as foolows. In the case of Unix or Cygwin, key in "egs5run db" instead of "egs5run". Next, execute programe by "egs5job.exe". In the case of DOS, execute "egs5run_db ucphantomcgv2".

Modify `egs5run.bat` to use debug mode and save as `egs5run_db.bat` for this purpose.
If "out of bound error" occurs, the line number and the argument coused error is displayed. Running egs5 with debug option needs mor CPU time than usual way and therefore is used only for this kind of check.

- Modify source energy sampling statements.

  Change

```
        call randomset(rnnow)
        do ie=1,nsebin
          if(rnnow.le.escdf(ie)) go to 1000
        end do
1000    ekin=esbin(ie)
```

  to

```
        call randomset(rnnow)
        do ie=1,nsebin
          if(rnnow.le.escdf(ie)) go to 1000
        end do
1000    if (ie.gt.nsebin) then
          ie=nsebin
        end if
        saspec(ie)=saspec(ie)+1.D0
        if (escdf(ie).eq.escdf(ie-1)) then
          ekin=esbin(ie-1)
        else
          ekin=esbin(ie-1)+(rnnow-escdf(ie-1))*(esbin(ie)-esbin(ie-1))/
     *         (escdf(ie)-escdf(ie-1))
        end if
```

- Add statements to output sampled X-ray spectrum.

  Change

```
!-----------------------------
!     Sampled source spectrum
!-----------------------------
```

  to

```
!-----------------------------
!     Sampled source spectrum
!-----------------------------
      do ie=2,nsebin
        saspec(ie)=saspec(ie)/float(ncases)
      end do

      write(6,272)
272   FORMAT(/' Comparison between sampled spectrum and pdf'
     *  /23X,'   Sampled      pdf     ',25X,'   Sampled       pdf     '
     *  )
      do ie=2,nsebin,2
        if(ie.eq.nsebin) then
          write(6,274) esbin(ie),saspec(ie),escdf(ie)-escdf(ie-1)
274       FORMAT(1X,G9.3,' MeV(upper)-- ',2G12.5)
        else
          write(6,276) esbin(ie),saspec(ie),escdf(ie)-escdf(ie-1),
     *    esbin(ie+1), saspec(ie+1),escdf(ie+1)-escdf(ie)
276       FORMAT(1X,G9.3,' MeV(upper)-- ',2G12.5,3X, '; ',G9.3,
     *           ' MeV(upper)-- ',2G12.5)
        end if
      end do
```

- Modify output format for the source information.

  Change

```
280   FORMAT(/' Absorbed energy inside phantom for Co-60 photon'/
```

  to

```
280   FORMAT(/' Absorbed energy inside phantom for 100kV X-ray'/
```

5. Modify ucxyz_phantom2.inp as follows:
   Change 2 places of

```
 &INP AE=0.521,AP=0.0100,UE=2.011,UP=1.5 /END
```

to

```
 &INP AE=0.521,AP=0.0100,UE=0.711,UP=0.2 /END
```

6. Run ucxyz_phantom2.f by egs5run.

   - In the case of Linux or Cygwin
     Enter ucxyz_phantom2 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 ucxyz_phantom2

7. Check egs5job.out to confirm average source energy is nearly equal to 40keV. Compare the sampled spectrum with pdf. Compare the absorbed dose distribution with xyz_phantom.out.

8. Check the trajectories using CGview.

4.3 Problem 3

1. cp ucxyz_phantom2.data ucxyz_phantom3.data
2. cp ucxyz_phantom2.inp ucxyz_phantom3.inp
3. Modify ucxyz_phantom3.data as follows:

   (a) Modify the number of voxel at Z-direction.
   Change

```
1.0,    20                      voxel width, number of voxels
```

   to

```
1.0,    16                      voxel width, number of voxels
```

   (b) Modify the material assignment etc.
   Change

```
1,3,1,3, 2,21,  1,  0.000, 0.00, 0.00        tissue
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
1,3,1,3,22,22,   2,  0.00,  0.00, 0.00       air
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
```

   to

```
1,3,1,3, 2, 4,  1,  0.000, 0.00, 0.00         tissue
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
1,3,1,3, 5,14,  1,  0.300, 0.00, 0.00         lung
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
1,3,1,3,15,17,  1,  0.000, 0.00, 0.00         tissue
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
1,3,1,3,18,18,   2,  0.00,  0.00, 0.00        air
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
```

4. Execute run5again.

   - In the case of Linux or Cygwin
     Enter ucxyz_phantom3 as the file name for unit 4 and simply enter "return" as the file name for unit 25.
     Enter simply return for "Enter name of the executable".

- In the case of DOS
  run5again ucxyz_phantom3

5. Confirm the density at the lung regions. Compare the absorbed dose distribution with xyz_pantom.out.

6. Check the trajectories using CGview.

4.4 Problem 4

1. cp ucxyz_phantom3.data ucxyz_phantom4.data

2. cp ucxyz_phantom3.inp ucxyz_phantom4.inp

3. Modify ucxyz_phantom4.data as follows.

   (a) Modify the density at tumor regions.
   Change

```
1,3,1,3, 5,14,  1,  0.300, 0.00, 0.00         lung
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
```

   to

```
1,3,1,3, 5, 7,  1,  0.300, 0.00, 0.00         lung
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
1,3,1,3, 8, 9,  1,  0.000, 0.00, 0.00         tumor
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
1,3,1,3,10,14,  1,  0.300, 0.00, 0.00         lung
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
```

4. Execute run5again.
   Enter ucxyz_phantom4 as the file name for unit 4 and simply enter "return" as the file name for unit 25.

5. Enter simply return for "Enter name of the executable".

6. Confirm the density at the tumor regions. Compare the absorbed dose distribution with xyz_pantom.out.

4.5 Ploblem 5

1. cp ucxyz_phantom4.data ucxyz_phantom5.data

2. cp ucxyz_phantom4.inp ucxyz_phantom5.inp

3. Modify ucxyz_phantom4.data as follows.

   (a) Modify the number of materials from '2' to '3'.
   Change

```
         2                    nmed (I10)
```

   to

```
         3                    nmed (I10)
```

   (b) Add new material (Iron).
   Change

```
AIR-AT-NTP                    media(j,2) (24A1)
```

   to

```
AIR-AT-NTP                    media(j,2) (24A1)
FE                            media(j,3) (24A1)
```

(c) Add characteristic dimension of iron.

Change

```
   1.0,   1.0                  chard
```

to

```
   1.0,   1.0, 1.0             chard
```

(d) Modify material assignment etc. related the "tumor" regions.

Change

```
1,3,1,3, 2, 4,  1,  0.000, 0.00, 0.00         tissue
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
1,3,1,3, 5, 7,  1,  0.300, 0.00, 0.00         lung
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
1,3,1,3, 8, 9,  1,  0.000, 0.00, 0.00         tumor
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
1,3,1,3,10,14,  1,  0.300, 0.00, 0.00         lung
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
1,3,1,3,15,17,  1,  0.000, 0.00, 0.00         tissue
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
```

to

```
1,3,1,3, 2, 6,  1,  0.000, 0.00, 0.00         tissue
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
1,3,1,3, 7, 7,  3,  0.000, 0.00, 0.00         iron
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
1,3,1,3, 8,21,  1,  0.000, 0.00, 0.00         tissue
  1  1  0  0  0  0  0  0       peang,edgefl,auger,ray,pola,incoh,prof,impac
```

4. Modify ucxyz_phantom5.inp as follows.

(a) Add input data for iron.

```
ELEM
 &INP IRAYL=1 /END
FE                              FE
FE
ENER
 &INP AE=0.521,AP=0.010,UE=2.011,UP=1.5 /END
PWLF
 &INP  /END
DECK
 &INP  /END
```

5. Execute run5again.
   Enter ucxyz_phantom5 as the file name for unit 4 and simply enter "return" as the file name for unit 25.

6. Enter simply return for "Enter name of the executable".

7. Confirm the density at the tumor regions. Compare the absorbed dose distribution with xyz_pantom.out.

Appendix: Full listings of `ucxyz_phantom.f`

```
!*****************************************************************************
!****************************  KEK, High Energy Accelerator Research  *
!****************************  Organization                           *
!* u c x y z  _ p h a n t o m*                                        *
!****************************     EGS5.0 USER CODE - 20 Jan 2016/1000 *
!                              Apply region dependent bremss splitting.*
!*****************************************************************************
!* This is a general User Code based on the cg geometry scheme.        *
!*****************************************************************************
!                                                                     *
!  PROGRAMMERS:  H. Hirayama                                          *
!                Radiation Science Center                             *
!                Applied Science Laboratory                           *
!                KEK, High Energy Accelerator Research Organization   *
!                1-1, Oho, Tsukuba, Ibaraki, 305-0801                 *
!                Japan                                                *
!                                                                     *
!                E-mail:      hideo.hirayama@kek.jp                   *
!                Telephone:   +81-29-864-5489                         *
!                Fax:         +81-29-864-1993                         *
!                Based on xyzdos.mor.                                 *
!                                                                     *
! 09AUG2004: CDF creation parts are corrected.    H. Hirayama         *
!*****************************************************************************
!*****************************************************************************
! The ucxyz_phantom.f User Code requires a data-input file            *
! (e.g., ucxyz_phantom.data) that is read by subroutine getvoxel (with *
! instructions in its header).  The following shows the geometry for   *
! ucxyz_phantom.data.                                                 *
! Trajectory data can be defined for PICT32 or CGview by setting      *
! npreci 1, 2 or 3, respectively.                                     *
! This user code corresponds to ucphantom_rec.mor for egs4.           *
! Use Ranlux random number generator.                                 *
!*****************************************************************************
!                                                                     *
!              ------------------------------------------------       *
!              3-Dimensional X-Y-Z Geometry (ucxyz_phantom example)    *
!              ------------------------------------------------       *
!                                                                     *
!                      X (Y into page)                                *
!                      ^                                              *
!                      |                                              *
!                      |                                              *
!                      | Outer vacuum region                          *
!                +-----+---+---+-   -+---+----+    15.0cm             *
!        Vacuum  |     |   |   |     |   |    |                       *
!        region  +Air +H2O|Water (H2O)  |Air |                       *
!                |     |   |   |      |H2O|    |                       *
!                +-----+---+---+-   -+---+----+--   0.5                *
!                |     |   |   |     |   |    |                       *
!                |Air |H2O|H2O|      |H2O|Air |                       *
! 1.253MeV photon|     |   |   |     |   |    |                       *
!      =========>+-----+---+---+-   -+---+----+-----> Z                *
! photons       -5.0  0.0 1.0       19.0 20.0                         *
!                                         25.0                        *
!                                                                     *
!                                                                     *
!*****************************************************************************
!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_edge.f'
      include 'include/egs5_elecin.f'
      include 'include/egs5_media.f'
      include 'include/egs5_misc.f'
      include 'include/egs5_stack.f'
      include 'include/egs5_thresh.f'
```

```
      include 'include/egs5_uphiot.f'
      include 'include/egs5_useful.f'
      include 'include/egs5_usersc.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/geoxyzv.f'
      include 'auxcommons/instuf.f'
      include 'auxcommons/lines.f'
      include 'auxcommons/nfac.f'
      include 'auxcommons/voxel.f'
      include 'auxcommons/watch.f'

      common/score/                              ! Variables to score
     *            depe(LIMAX,LJMAX,LKMAX),faexp,fexps,maxpict
      real*8 depe,faexp,fexps
      integer maxpict

!**** real*8                                                 ! Arguments
      real*8 etot,totke
!**** integer
      integer ifto

!**** real*8                                             ! Local variables
      real*8
     * amass,areac,availke,depthl,depths,dis,disair,ei0,ekin,elow,eup,
     * phai0,phai,radma2,rnnow,sinth,sposi,tnum,w0,wimin,wtin,wtsum,
     * xhbeam,xpf,yhbeam,ypf

      real*8 bsfa,bsferr,faexps,faexp2s,faexrr,fexpss,fexps2s,fexerr,
     *       faexpa,fexpsa

      real*8
     * depeh(LIMAX,LJMAX,LKMAX),depeh2(LIMAX,LJMAX,LKMAX),
     * dose(LIMAX,LJMAX,LKMAX),doseun(LIMAX,LJMAX,LKMAX)

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

      integer
     * i,ii,iii,icases,idin,idose,ie,ipage,irl,j,jhist,jj,jl,ju,k,
     * kkk,nlist,nperpg

!     ----------
!     Open files
!     ----------
!-----------------------------------------------------------------------
!     Units 7-26 are used in pegs and closed.  It is better not
!     to use as output file. If they are used must be re-open after
!     getvoxel 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
!-----------------------------------------------------------------------------
      ifto = 6      ! Output unit in getvoxel
!     ==================
      call getvoxel(ifto)
!     ==================

!     ------------------------------
!     Run PEGS5 before calling HATCH
!     ------------------------------
      write(6,*) 'PEGS5-call comes next'

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

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

!-----------------------------------------------------------------------
! Step 3: Pre-hatch-call-initialization
!-----------------------------------------------------------------------
!-----------------------------------------------
!     Define pict data mode.
!-----------------------------------------------
!     npreci 1: for PICT32
!            2: for CGview
!            3: for CGview in free format
      npreci=3

      if(npreci.eq.3) write(39,fmt="('GSTA-FREE-TIME')")
      if(npreci.eq.2) write(39,fmt="('GSTA-TIME')")
      write(39,fmt="('SLAB')")
      write(39,fmt="(I6)") imax+1
      write(39,fmt="(I6)") jmax+1
      write(39,fmt="(I6)") kmax+1
      write(39,fmt="(4F15.4)") (xbound(j),j=1,imax+1)
      write(39,fmt="(4F15.4)") (ybound(j),j=1,jmax+1)
      write(39,fmt="(4F15.4)") (zbound(j),j=1,kmax+1)
      write(39,fmt="('GEND')")

      write(39,fmt="('MSTA')")
      write(39,fmt="(I4)") nreg
      write(39,fmt="(15I4)") (med(i),i=1,nreg)
      write(39,fmt="('MEND')")

!     ------------------------------------------------------------
!     Random number seeds.  Must be defined before call hatch
!     or defaults will be used.  inseed (1- 2^31)
!     ------------------------------------------------------------
      luxlev = 1
      inseed=1
      write(6,100) inseed
100   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 source position from phantom surface.
!-----------------------------------------------
!     Source position from phantom surface in cm.
      sposi=10.0

      iqin=0             ! Incident charge - photons
      ekein=1.253        ! Kinetic energy of source photon
      etot=ekein + abs(iqin)*RM
      xin=0.D0
      yin=0.D0
      zin=-sposi
      uin=0.D0
      vin=0.D0
      win=1.D0
      wtin=1.D0

!-------------------------------------------
!     Half width and height at phantom surface
!-------------------------------------------
!     X-direction half width of beam at phantom surface in cm.
      xhbeam=1.0
!     Y-direction half height of beam at phantom surface in cm.
      yhbeam=1.0
      radma2=xhbeam*xhbeam+yhbeam*yhbeam
      wimin=sposi/dsqrt(sposi*sposi+radma2)

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

      write(6,110)
110   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(6,120)
120   FORMAT(/,' HATCH-call comes next',/)

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

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

!-----------------------------------------------------------------------
!     Output medium and region information to file for calculation mode.
!-----------------------------------------------------------------------
      write(6,*) ' Quantities associated with each media:'
      do j=1,nmed
        write(6,130) (media(i,j),i=1,24)
130     FORMAT(/,1X,24A1)
        write(6,140) rhom(j),rlcm(j)
140     FORMAT(5X,' Rho=',G15.7,' g/cm**3     RLC=',G15.7,' cm')
        write(6,150) ae(j),ue(j),ap(j),up(j)
150     FORMAT(5X,' AE=',G15.7,' MeV    UE=',G15.7,' MeV'/ 5X,' AP=',G
     *  15.7,' MeV    UP=',G15.7,' MeV')
      end do

      write(6,160)
160   FORMAT(/' Information of medium and cut-off for central region')
      i=imax/2+1
      j=jmax/2+1
      do k=1,kmax
        irl=1+i+(j-1)*imax+(k-1)*ijmax
        if (med(irl).eq.0) then
          write(6,170) k,irl
170       FORMAT(' Medium(',I3,'-th z bin, region:',I5,')= Vacuum')
        else
          write(6,180) k,irl,(media(ii,med(irl)),ii=1,24),
     *                 ecut(irl),pcut(irl),rhor(irl)
180       FORMAT(' Medium(',I3,'-th z bin, region:',I5,
     *           ')=',24A1,/5X,'ECUT=',G10.5,' MeV, PCUT=',
     *           G10.5, ' MeV, density=',F10.3)
        end if
      end do

!-----------------------------------------------------------------------
! Step 6:  Initialization-for-howfar
!-----------------------------------------------------------------------
!       Initialization is done inside getvoxel

!-----------------------------------------------------------------------
! Step 7:  Initialization-for-ausgab
!-----------------------------------------------------------------------
      ncount = 0
      ilines = 0
      nwrite = 10
      nlines = 25
      idin = -1
      totke = 0.
      wtsum = 0.

!---------------------
!     Clear variables
!---------------------
!     Zero the dose
      do k=1,kmax
        do j=1,jmax
          do i=1,imax
            depe(i,j,k)=0.D0
            depeh(i,j,k)=0.D0
            depeh2(i,j,k)=0.D0
          end do
        end do
      end do
```

```
      faexp=0.D0
      faexps=0.D0
      faexp2s=0.D0
      fexps=0.D0
      fexpss=0.D0
      fexps2s=0.D0

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

!-------------------------
!     History number
!-------------------------
!     History number
      ncases=100000
!     Maximum history number to write trajectory data
      maxpict=50

      write(39,fmt="('0    1')")

      write(6,190)
190   FORMAT(//,' ENERGY/COORDINATES/DIRECTION COSINES/ETC.',/,
     *        6X,'E',16X,'X',14X,'Y',14X,'Z'/
     *        1X,'U',14X,'V',14X,'W',9X,'IQ',4X,'IR',3X,'IARG',/)

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

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

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

                                              ! -------------------------
      do jhist=1,ncases                       ! Start of CALL SHOWER loop
                                              ! -------------------------
        icases=jhist
!---------------------------------
!       Determine direction (isotropic)
!---------------------------------
200     call randomset(w0)
        win=w0*(1.0-wimin)+wimin
        call randomset(phai0)
        phai=pi*(2.0*phai0-1.0)
        sinth=dsqrt(1.D0-win*win)
        uin=dcos(phai)*sinth
        vin=dsin(phai)*sinth
        dis=sposi/win
        xpf=dis*uin
        ypf=dis*vin
        if (dabs(xpf).gt.xhbeam.or.dabs(ypf).gt.yhbeam) go to 200
        if (sposi.gt.zbound(2)-zbound(1)) then
          disair=(sposi-(zbound(2)-zbound(1)))/win
          xin=disair*uin
          yin=disair*vin
          zin=zbound(1)
        else
          xin=0.D0
          yin=0.D0
          zin=-sposi
        end if

        do i=1,imax
            if (xbound(i+1).gt.xin) go to 210
        end do

210     do j=1,jmax
          if (ybound(j+1).gt.yin) go to 220
        end do

!       -------------
!       Input region
!       -------------
220     k=1
```

```
        irin=1+i+(j-1)*imax

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

        latchi=0

!       ----------------------------------------------------
!       Print first NWRITE or NLINES, whichever comes first
!       ----------------------------------------------------
        if (ncount .le. nwrite .and. ilines .le. nlines) then
          ilines = ilines + 1
          write(6,230) etot,xin,yin,zin,uin,vin,win,iqin,irin,idin
230       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    ' normarized.',3e12.5)")uin,vin,win
          stop
        end if

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

!---------------------------------
!      Sum variable and its square.
!---------------------------------

        do k=1,kmax
          do j=1,jmax
            do i=1,imax
              depeh(i,j,k)=depeh(i,j,k)+depe(i,j,k)
              depeh2(i,j,k)=depeh2(i,j,k)+depe(i,j,k)*depe(i,j,k)
              depe(i,j,k)=0.D0
            end do
          end do
        end do

        faexps=faexps+faexp
        faexp2s=faexp2s+faexp*faexp
        faexp=0.0
        fexpss=fexpss+fexps
        fexps2s=fexps2s+fexps*fexps
        fexps=0.0

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

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

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

      tt=etime(tarray)
      tt1=tarray(1)
      cputime=tt1-tt0
      write(6,250) cputime
250   format(' Elapsed Time (sec)=',G15.5)
```

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

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

!     --------------------
!     Write out the results
!     --------------------
      write(6,260) ncount,ncases,totke
260   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(6,270) totke,availke,ncount
270     FORMAT(/,' Stopped in MAIN with TotKE=',G15.5,/,
     *         ' AvailKE=',G15.5, /,' Ncount=',I10)
        stop
      end if

!-----------------------------
!     Sampled source spectrum
!-----------------------------

      write(6,280) sposi
280   FORMAT(/' Absorbed energy inside phantom for 1.253MeV photon'/
     *' Source position ',F10.1,' cm from phantom surface'/
     *' Within 1cm x 1 cm area after 5 cm air')

      write(6,290) ncases, xhbeam, yhbeam
290   FORMAT(1X,I8,' photons normally incident from front side'/
     *' Half width of beam is ',G15.5,'cm for X and ',G15.5,'cm for Y')

!      ---------------------------------------
!      Calculate average and its uncertainties
!      ---------------------------------------

        do k=1,kmax
          do j=1,jmax
            do i=1,imax
              irl=1+i+(j-1)*imax+(k-1)*ijmax
              amass=(xbound(i+1)-xbound(i))*
     *              (ybound(j+1)-ybound(j))*
     *              (zbound(k+1)-zbound(k))*rhor(irl)
              dose(i,j,k)=depeh(i,j,k)/ncases
              depeh2(i,j,k)=depeh2(i,j,k)/ncases
              doseun(i,j,k)=dsqrt((depeh2(i,j,k)-
     *                     dose(i,j,k)*dose(i,j,k))/ncases)
              dose(i,j,k)=dose(i,j,k)*1.602D-10/amass
              doseun(i,j,k)=doseun(i,j,k)*1.602D-10/amass
            end do
          end do
        end do

!-----------------------------------------------
!     Print out the results of central phantom
!-----------------------------------------------
      i=imax/2+1
      j=jmax/2+1
      do kkk=2,kmax-1
        depths=zbound(kkk)
        depthl=zbound(kkk+1)
        irl=1+i+(j-1)*imax+(kkk-1)*ijmax
        write(6,300) depths,depthl,(media(ii,med(irl)),ii=1,24),
     *  rhor(irl),dose(i,j,kkk),doseun(i,j,kkk)
300     FORMAT(' At ',F4.1,'--',F4.1,'cm (',24A1,',rho:',F8.4,')=',
     *  G13.5,'+-',G13.5,'Gy/incident')
      end do

!-------------------------------------------------
!     Calculate average exposure and its deviation
!-------------------------------------------------
      area=(xbound(i+1)-xbound(i))*(ybound(j+1)-ybound(j))
      faexpa=faexps/ncases
```

```
      faexp2s=faexp2s/ncases
      faexrr=dsqrt((faexp2s-faexpa*faexpa)/ncases)
      faexpa=faexpa*1.6E-10/area
      faexrr=faexrr*1.6E-10/area
      fexpsa=fexpss/ncases
      fexps2s=fexps2s/ncases
      fexerr=dsqrt((fexps2s-fexpsa*fexpsa)/ncases)
      fexpsa=fexpsa*1.6E-10/area
      fexerr=fexerr*1.6E-10/area
      if (faexpa.gt.0.0) then
        bsfa=fexpsa/faexpa
        bsferr=bsfa*dsqrt((faexrr/faexpa)**2.+(fexerr/fexpsa)**2.)
        write(6,310) faexpa,faexrr,fexpsa,fexerr,bsfa,bsferr
310     FORMAT(/' Exposure in free air (using mu_en) =', G15.5,'+-',G15.
     *  5,' Gy/incident'/ ' Exposure at phantom surface (using mu_en) ='
     *  , G15.5,'+-',G15.5,'Gy/incident'/ ' Backscattering factor =',G15
     *  .5,'+-',G15.5)
      else
        write(6,320) faexpa,faexrr,fexpsa,fexerr
320     FORMAT(/' Exposure in free air (using mu_en) =', G15.5,'+-',G15.
     *  5,' Gy/incident'/ ' Exposure at phantom surface (using mu_en) ='
     *  , G15.5,'+-',G15.5,'Gy/incident')
      end if

!     ---------------------------
!     Write out the whole results
!     ---------------------------
      do idose=1,idgrp         ! Loop over groups of regions to analyze
        if (izscan(idose).ne.0) then   ! Do output with one Z scan per page
!         Number of sets of depth per page
          k = (kdosu(idose) - kdosl(idose))
          k = k + k/5 + 7
          nperpg = 60/k
          write(6,330) Title
330       FORMAT(/10X,80A1//T10,'xyz(V01) dose outputs  Gy.cm**2',
     *    '(or Gy/incident particle for 0 area)')
          ipage=1   ! Count how many zgroups printed on this page

          do i=idosl(idose),idosu(idose)
            do j=jdosl(idose),jdosu(idose),4
              jl=j
              ju=min(j+3,jdosu(idose))
              write(6,340) xbound(i),xbound(i+1),i
340           FORMAT(/T15,'For x=',F10.3,' to',F10.3,5X,'i=',I3)
              write(6,350) (ybound(jj),jj=jl,ju+1)
350           FORMAT(' ybounds:',F7.3,F12.3,3F17.3)
              write(6,360)(jj,jj=jl ,ju)
360           FORMAT(T10,'j=',t17,5(I4,13X))
              write(6,370) zbound(kdosl(idose))
370           FORMAT(' zbounds (',F10.3,')')
              do k=kdosl(idose),kdosu(idose)
                write(6,380) zbound(k+1),k,(dose(i,jj,k),
     *          min(99.9, 100.*doseun(i,jj,k)/dose(i,jj,k)),
     *          jj=jl,ju)
380             FORMAT(F8.3,I4,4(1PE11.3,'-',0PF4.1,'%') )
                if (mod(k,5).eq.0) then
                  write(6,390)
390               FORMAT(' ')
                end if
              end do

              if(mod(ipage,nperpg).eq.0.and.(ju.ne.jdosu(idose).
     *          or.i.ne.idosu(idose))) then
                write(6,330) Title
                ipage=1
              else
                ipage=ipage+1
              end if
            end do   ! end j-loop
          end do     ! end i-loop

        else                    ! Output x scans each page
          i=idosu(idose)-idosl(idose)
          i=i+i/5+7
          nperpg=60/i        ! Number of sets of bins per page
          write(6,330) Title
```

```
          ipage=1

          do k=kdosl(idose),kdosu(idose)
            do j=jdosl(idose),jdosu(idose),4
              jl=j
              ju=min(j+3,jdosu(idose))
              write(6,400) zbound(k),zbound(k+1),k
400           FORMAT(/T15,'for z=',F10.3,' to',F10.3,5X,'k=',I3)
              write(6,410) (ybound(jj),jj=jl,ju+1)
410           FORMAT(' Ybounds:',F7.3,F12.3,3F17.3)
              write(6,420) (jj, jj=jl,ju)
420           FORMAT(T10,'j=',T17,5(I4,13X))
              write(6,430) xbound(idosl(idose))
430           FORMAT(' Xbounds (',F10.3,')')

              do i=idosl(idose),idosu(idose)
                write(6,440) xbound(i+1),i,(dose(i,jj,k),
     *          min(99.9, 100.*doseun(i,jj,k)/dose(i,jj,k)),
     *          jj=jl,ju)
440             FORMAT(F8.3,I4,4(1PE11.3,'-',0PF4.1,'%') )
                if (mod(i,5).eq.0) then
                  write(6,fmt="(' ')")
                end if
              end do

              if(mod(ipage,nperpg).eq.0.and.(ju.ne.jdosu(idose).
     *          or.k.ne.kdosu(idose))) then
                write(6,330) Title
                ipage=1
              else
                ipage=ipage+1
              end if
            end do                ! end j-loop
          end do                  ! end k-loop
        end if                    ! end of x scan per page output
      end do                    ! end of idose loop

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

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

      stop

      end

!-------------------------last line of main code-------------------------
!----------------------------getvoxel.f--------------------------------
! Version:   161020-1000
!         Read material and geometry data from egs5job.inp
!         Apply for setting bremss splitting to each region.
!-----------------------------------------------------------------------
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12
! ---------------------------------------------------------------------
! Auxiliary subroutine for use with the EGS5 Code System
! ---------------------------------------------------------------------
! This is a data-entry subprogram for use with a general-purpose
! egs5 user code to do cartesian coordinate dose deposition studies.
! Every voxel (volume element) can have different materials and/or
! varying densities (for use with CT data).
! Basic pats of this subroutine related with geometry taken from
! xyzdos.mor.
! ---------------------------------------------------------------------
!
!   voxels are labeled by indicies (i,j,k) and defined by:
!        xbound(i) <= x < xbound(i+1)     i <= imax
!        ybound(j) <= y < ybound(j+1)     j <= jmax
!        zbound(k) <= z < zbiund(k+1)     k <= kmax
!
! -------------------
! SUBROUTINE ARGUMENT
! -------------------
!  nreg      Number of regions in geometry (determined by data input).
!
! ----------------
! UNIT ASSIGNMENTS
! ----------------
```

```
!  Unit 4       Input file.
!  Unit ifto    Output file.
!
! ----------
! INPUT FILE
! ----------
!  Record 1  title (80A1)        Title line.
!  --------
!  Record 2  nmed (I10)          Number of media in problem.
!  --------
!  Record 3  media(j,i) (24A1)   Media names (j=1,24, I=1,nmed lines).
!  --------                      Note that entire volume is initially
!                                set to medium.
! ---------
!  Record 4  chard               characteristic distance for each medium
! ---------
!  --------
!  Record 5   maxx, maxy, maxz   Number of voxels in the X,Y,Z directions
!  --------                      If <0, it means that number of equally
!                                spaced boundaries will be input.
!  --------
!  Record 6   xbound
!  --------
!                      i.e. repeat the following replacing (i and x) by
!                        (j and y) and (k and z) respectively.
!    if maxx > 0
!               input, one per line, the maxx + 1  x boundaries
!
!    if maxx < 0
!               input smallest x boundary, followed by  abs(maxx) pairs
!               one per line: voxel width, # voxels with this width
!
!     for example: starting at record 5
!                -1,-1,-1
!                0.0
!                1.0,16
!                0.0
!                1.0,16
!                0.0
!                1.0,16
!     defines a 16x16x16 cube of 1cm**3 voxels with a total of 4097 regions
!    or
!                -1,-1,3
!                0.0
!                1.0,16
!                0.0
!                1.0,16
!                0.0
!                5.0
!                10.0
!      defines a 16x16x10 cube with 1x1x5 cm voxels stacked 2 deep
!  --------
!  Record 7    ybound
!  --------
!  Record 8    zbound
!  --------
!  ---------
!  Record 9 ecutin,pcutin,       For medium 1 at first
!  ---------ipeangsw,           Switches for PE-angle sampling,
!           iedgeflsw,           K & L-edge fluorescence,
!           iaugersw,            K & L-Auger
!           iraysw,              Rayleigh scattering,
!           ipolarsw,            Linearly-polarized photon scattering,
!           incohrsw,            S/Z rejection,
!           iprofrsw,            Doppler broadening,
!           impacrsw             electron impact ionization (0=off, 1=on).
!  --------
! Record 10 il,iu, jl,ju, kl,ku, medtmp, rhotmp,ecutin,pcutin
!  --------                     Line is repeated until a blank line found
!                               All regions default to medium 1 with its
!                               default density unless changed here.
!                               For all voxels with
!                                IL <= I <= IU
!                                JL <= J <= JU
!                                KL <= K <= KU
!                          the medium used is MEDIUM and the density used is
!                          DENSITY. If DENSITY=0.0, the default value for that
!                          medium is used (faster than entering default density
!                          here).
!                          If iu and il are zero, it means the end of define.
!                                   If medium not 0, following option is set
!                                   to the regions above.
```

```
! ---------
! Record 10a ipeangsw,           Switches for PE-angle sampling,
! ---------- iedgeflsw,          K & L-edge fluorescence,
!            iaugersw,           K & L-Auger
!            iraysw,             Rayleigh scattering,
!            ipolarsw,           Linearly-polarized photon scattering,
!            incohrsw,           S/Z rejection,
!            iprofrsw,           Doppler broadening,
!            impacrsw            electron impact ionization (0=off, 1=on).
!
! --------
! Record 11  il,iu, jl,ju, kl,ku,izscan
! ---------                      Regions for which the dose will be output.
!                                IZSCAN non-zero to get z-scan per page,
!                                otherwise output is an x-scan per page.
!
! ---------
! Record 12 iwatch               Switch for tracking events with swatch:
! ---------                        (0=No, 1=each interaction,
!                                   2=each step)
! ---------
! Record 13 ibrdst,iprdst,       Switches for bremsstrahlung and pair
! --------- ibrspl0,nbrspl       production ANGLE SAMPLING, and brems-
!                                strahlung SPLITTING:
!                                ibrdst=0 No (use default: theta=m/E)
!                                       1 Yes (recommended)
!                                iprdst=0 No (use default: theta=m/E)
!                                       1 Yes (low-order distribution)
!                                       2 Yes (recommended)
!                               ibrspl0=0 No
!                                       1 Yes (NBRSPL=splitting factor)
! ------------------------------------------------------------------------

      subroutine getvoxel(ifto)

      implicit none

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

      include 'include/egs5_bounds.f'    ! COMMONs required by EGS5 code
      include 'include/egs5_brempr.f'
      include 'include/egs5_edge.f'
      include 'include/egs5_eiicom.f'
      include 'include/egs5_elecin.f'
      include 'include/egs5_media.f'
      include 'include/egs5_misc.f'
      include 'include/egs5_thresh.f'
      include 'include/egs5_uphiot.f'
      include 'include/egs5_useful.f'
      include 'include/egs5_userpr.f'
      include 'include/egs5_usersc.f'
      include 'include/egs5_uservr.f'
      include 'include/egs5_userxt.f'

      include 'pegscommons/mscom.f'                         ! PEGS common

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

      include 'auxcommons/edata.f'        ! Auxiliary-code COMMONs
      include 'auxcommons/geoxyzv.f'
      include 'auxcommons/instuf.f'
      include 'auxcommons/voxel.f'
      include 'auxcommons/watch.f'

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

      integer ifto                         ! Argument

      real*8                                          ! Local variables
     * ecutin,ecutmn,ek0,pcutin,rhotmp,totphi,
     * thetax,thetay,thetaz,xlower,
     * xupper,ylower,yupper,width

      integer i,igroup,ii,iiz,il,in,irl,iu,ixinu,
     * ixx,izn,j,jl,ju,jxx,jyinu,k,kl,ku,maxbd,maxx,maxy,
     * maxz,medtmp,moreOutput,n,ner,ngroup,nn,nnn,ibrspl0

      integer ipeangsw,iedgeflsw,iaugersw,iraysw,ipolarsw,incohrsw,
```

```
     *        iprofrsw,impacrsw

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

      write(ifto,100)
100   FORMAT(//,T25,'+----------------------------------------+',
     *        /,T25,'| EGS5 User Code using subroutine voxel  |',
     *        /,T25,'+----------------------------------------+',
     *        /,T25,'|  NOTE:  X-Y-Z Voxel geometry.          |',
     *        /,T25,'|         X-Y plane on the page          |',
     *        /,T25,'|         (X to the right, Y upwards,    |',
     *        /,T25,'|           Z out).                      |',
     *        /,T25,'+----------------------------------------+',
     *        //)

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

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

! ---------------
! Record 1: title
! ---------------
      read(4,110) title
 110  FORMAT(80A1)
      write(ifto,120) title
 120  FORMAT(8x,71('-')/'$TITLE: '/'+',3X,80A1/8X,71('-'))

! --------------
! Record 2: nmed
! --------------
      read(4,*) nmed
      if (nmed.eq.0 .or. nmed .gt. MXMED) then
        write(ifto,130) nmed
 130    FORMAT(' *** Stopped in Getvoxel with nmed=',I5,' > MXMED')
        stop
      end if
      write(ifto,140) nmed
 140  FORMAT(' Number of media : ',I5,/)

! ---------------
! Record 3: media
! ---------------
      do i=1,nmed
        read(4,150) (media(j,i),j=1,24)
 150    FORMAT(24A1)
        write(ifto,160) i,(media(j,i),j=1,24)
 160    FORMAT(' MEDIUM=',I5,' ==> ',24A1)
      end do

!     charD is defined to each medium.
! ----------------
! Record 4: chard
! ----------------
      read(4,*) (chard(i),i=1,nmed)

      write(ifto,*) 'chard =',(chard(i),i=1,nmed)

! ---------------------------
! Record 5: maxx, maxy, maxz
! ---------------------------
      read(4,*) maxx,maxy,maxz

!     Check bin-number.
      if (maxx.eq.0) maxx =1
      if (maxx.gt.LIMAX) maxx=LIMAX
      if (maxy.eq.0) maxy =1
      if (maxy.gt.LJMAX) maxy=LJMAX
      if (maxz.eq.0) maxz =1
      if (maxz.gt.LKMAX) maxz=LKMAX

      write(ifto,170) maxx,maxy,maxz
170   FORMAT ('+',3I6)

      maxbd=LIMAX
      write(ifto,180)
180   FORMAT(/T20,'Input boundaries in the x direction')
```

```
! -----------------
! Record 6   xbound
! -----------------
      if (maxx.gt.0) then            ! Just pick up boundaries.
        do i=1,maxx
          write(ifto,190) i
190       FORMAT(' Small boundary for region(',I3,')  ')
          read(4,*) xbound(i)
!         if (i.ne.1.and.xbound(i).le.xbound(i-1)) then
          if (i.ne.1) then
            if(xbound(i).le.xbound(i-1)) then
              write(ifto,200)
200           FORMAT(' Boundary out of order**************')
            end if
          end if
          write(ifto,210) xbound(i)
210       FORMAT('+',T10,F12.3)
        end do
        write(ifto,220) maxx
220     FORMAT(' Outer boundary for region(',I3,')  ')
        read(4,*) xbound(maxx+1)
        write(ifto,230) xbound(maxx+1)
230     FORMAT('+',T10,F12.3)
      else                           ! Input groups of region.
        write(ifto,240)
240     FORMAT(' Initial boundary: ')
        read(4,*) xbound(1)
        write(ifto,250) xbound(1)
250     FORMAT('+',F12.3)
        ngroup=-maxx
        maxx=0
        do igroup=1,ngroup
          write(ifto,260)
260       FORMAT(' Width in this group, no. of regions in group: ')
          read(4,*) width,nn
          if(nn.le.0) nn=1
          if(width.le.0.0) width=1.D0
          write(ifto,270) width,nn
270       FORMAT('+',F12.3,I5)
          nnn=min(nn,maxbd-maxx)
          if (nnn.ne.0) then
            do in=maxx+1,maxx+nnn
              xbound(in+1)=xbound(in)+width
            end do
          end if
          if (nn.ne.nnn) then
            write(ifto,280)
280         FORMAT(T15,'*** No. of X-direction reduced ***'/)
          end if
          maxx=maxx+nnn
        end do
        write(ifto,290) (xbound(i),i=1,maxx+1)
290     FORMAT(' Boundaries'/(6F12.3))
      end if

      imax=maxx

      maxbd=LJMAX
      write(ifto,300)
300   FORMAT(/T20,'Input boundaries in the y direction')

! -----------------
! Record 7   ybound
! -----------------
      if (maxy.gt.0) then            ! Just pick up boundaries.
        do i=1,maxy
          write(ifto,190) i
          read(4,*) ybound(i)
!          if (i.ne.1.and.ybound(i).le.ybound(i-1)) then
          if (i.ne.1) then
            if(ybound(i).le.ybound(i-1)) then
              write(ifto,200)
            end if
          end if
          write(ifto,210) ybound(i)
        end do
        write(ifto,220) maxy
        read(4,*) ybound(maxy+1)
        write(ifto,230) ybound(maxy+1)
      else                           ! Input groups of region.
```

```
        write(ifto,240)
        read(4,*) ybound(1)
        write(ifto,250) ybound(1)
        ngroup=-maxy
        maxy=0
        do igroup=1,ngroup
          write(ifto,260)
          read(4,*) width,nn
          if(nn.le.0) nn=1
          if(width.le.0.0) width=1.D0
          write(ifto,270) width,nn
          nnn=min(nn,maxbd-maxy)
          if (nnn.ne.0) then
            do in=maxy+1,maxy+nnn
              ybound(in+1)=ybound(in)+width
            end do
          end if
          if(nn.ne.nnn) then
            write(ifto,280)
          end if
          maxy=maxy+nnn
        end do
        write(ifto,290) (ybound(i),i=1,maxy+1)
      end if

      jmax=maxy

      maxbd=LKMAX
      write(ifto,310)
310   FORMAT(/T20,'Input boundaries in the z direction')

! -----------------
! Record 8   zbound
! -----------------
      if (maxz.gt.0) then            ! Just pick up boundaries.
        do i=1,maxz
          write(ifto,190) i
          read(4,*) zbound(i)
!          if (i.ne.1.and.zbound(i).le.zbound(i-1)) then
          if (i.ne.1) then
            if(zbound(i).le.zbound(i-1)) then
              write(ifto,200)
            end if
          end if
          write(ifto,210) zbound(i)
        end do
        write(ifto,220) maxz
        read(4,*) zbound(maxz+1)
        write(ifto,230) zbound(maxz+1)
      else                           ! Input groups of region.
        write(ifto,240)
        read(4,*) zbound(1)
        write(ifto,250) zbound(1)
        ngroup=-maxz
        maxz=0
        do igroup=1,ngroup
          write(ifto,260)
          read(4,*) width,nn
          if(nn.le.0) nn=1
          if(width.le.0.0) width=1.D0
          write(ifto,270) width,nn
          nnn=min(nn,maxbd-maxz)
          if (nnn.ne.0) then
            do in=maxz+1,maxz+nnn
              zbound(in+1)=zbound(in)+width
            end do
          end if
          if(nn.ne.nnn) then
            write(ifto,280)
          end if
          maxz=maxz+nnn
        end do
        write(ifto,290) (zbound(i),i=1,maxz+1)
      end if

      kmax=maxz

      ijmax = imax*jmax
      irmax = 1 + ijmax*kmax
      nreg = irmax

      write(ifto,320) imax,jmax,kmax,nreg
```

```
320   FORMAT(' imax, jmxa, kmax, nreg =',4I8)

!     Check nreg
      if (nreg .gt. MXREG) then
        write(ifto,330) nreg
330     FORMAT(' *** Stopped in getvoxel with nreg=',I5,' > MXREG')
        stop
      end if
      write(ifto,340) nreg
340   FORMAT(/,' number of region (nreg) =',I5,/,
     *         ' nreg includes outside vacuum region (regin=1)')

!     Set all region except 1 set to medium=1.
      med(1)=0

! ------------------------------------------------------------
! Record 9: ipeangsw,iedgeflsw,iaugersw,iraysw,
!           ipolarsw,incohrsw,iprofrsw,impacrsw for medium 1
! ------------------------------------------------------------
        read(4,*) ecutin,pcutin,ipeangsw,iedgeflsw,iaugersw,iraysw,
     *            ipolarsw,incohrsw,iprofrsw,impacrsw

        write(ifto,350) ecutin,pcutin,ipeangsw,iedgeflsw,iaugersw,
     *             iraysw,ipolarsw,incohrsw,iprofrsw,impacrsw
350     FORMAT(/' Medium 1'/' ecut =',G15.5,' and pcut =',G15.5/
     *' ipeangsw=',I3,',iedgeflsw=',I3,',iaugersw=',I3,'iraysw=',I3/
     *' ipolarsw=',I3,' ,incohrsw=',I3,',iprofrsw=',I3,',impacrsw=',I3)

      do i=2,irmax
        med(i)=1
        if (pcutin .gt. 0.)  pcut(i) = pcutin
        if (ecutin .gt. 0.)  ecut(i) = ecutin + RM
        iphter(i) = ipeangsw
        iedgfl(i) = iedgeflsw
        iauger(i) = iaugersw
        iraylr(i) = iraysw
        lpolar(i) = ipolarsw
        incohr(i) = incohrsw
        iprofr(i) = iprofrsw
        impacr(i) = impacrsw
      end do

!  --------
!  Record 10 il,iu, jl,ju, kl,ku, medtmp, rhotmp, ecutin, pcutin
!  -------- (7I5,3F10.0)       Line is repeated until a blank line found

360   write(ifto,370)
370   FORMAT(' Lower,upper i, j, k, medium, density')

      read(4,*) il,iu,jl,ju,kl,ku,medtmp,rhotmp,ecutin,pcutin
      if(il.eq.0 .and. iu.eq.0) go to 400

!     Check il etc.
      if(il.lt.0) il=1
      if(iu.lt.0 .or. iu.ge.imax) iu=imax
      if(jl.le.0) jl=1
      if(ju.le.0 .or. ju.ge.jmax) ju=jmax
      if(kl.le.0) kl=1
      if(ku.le.0 .or. ku.ge.kmax) ku=kmax

!     Check medtmp
      if(medtmp.lt.0 .or. medtmp.gt.nmed) medtmp=1

      write(ifto,380) il,iu,jl,ju,kl,ku,medtmp,rhotmp
380   FORMAT('+',3('(',I3,I4,')'), I4, F10.3)

      if (medtmp.ne.0) then
! ------------------------------------------------
! Record 10a: ipeangsw,iedgeflsw,iaugersw,iraysw,
!             ipolarsw,incohrsw,iprofrsw,impacrsw
! ------------------------------------------------
        read(4,*) ipeangsw,iedgeflsw,iaugersw,iraysw,ipolarsw,
     *            incohrsw,iprofrsw,impacrsw

        write(ifto,390) ecutin,pcutin,ipeangsw,iedgeflsw,iaugersw,
     *             iraysw,ipolarsw,incohrsw,iprofrsw,impacrsw
390     FORMAT(' ecut =',G15.5,' and pcut =',G15.5/
```

```
     *' ipeangsw=',I3,',iedgeflsw=',I3,',iaugersw=',I3,'iraysw=',I3/
     *' ipolarsw=',I3,' ,incohrsw=',I3,',iprofrsw=',I3,',impacrsw=',I3)

        do i=il,iu
          do j=jl,ju
            do k=kl,ku
              irl=1+i+(j-1)*imax+(k-1)*ijmax
              med(irl)=medtmp
              if (rhotmp.ne.0) rhor(irl)=rhotmp
              if (pcutin .gt. 0.)  pcut(irl) = pcutin
              if (ecutin .gt. 0.)  ecut(irl) = ecutin+ RM
              iphter(irl) = ipeangsw
              iedgfl(irl) = iedgeflsw
              iauger(irl) = iaugersw
              iraylr(irl) = iraysw
              lpolar(irl) = ipolarsw
              incohr(irl) = incohrsw
              iprofr(irl) = iprofrsw
              impacr(irl) = impacrsw
            end do
          end do
        end do
       else
         do i=il,iu
           do j=jl,ju
             do k=kl,ku
               irl=1+i+(j-1)*imax+(k-1)*ijmax
               med(irl)=0
             end do
           end do
         end do
       end if

       go to 360

400    continue
! --------
! Record 11  il,iu, jl,ju, kl,ku,izscan
! ---------
       write(ifto,410)
410    FORMAT(' 3 pairs defining lower,upper x,y,z indecses of dose',
      * 'regions'/' for which results are to be output'/
      * ' izscan non-zero : scan per page'/
      * ' One set of 6 per line, end with all zero')

       idgrp=0
420    idgrp=idgrp+1
       write(ifto,430)
430    FORMAT('$: ')
       read(4,*) idosl(idgrp),idosu(idgrp),jdosl(idgrp),jdosu(idgrp),
      *          kdosl(idgrp),kdosu(idgrp),izscan(idgrp)

       if(idosl(idgrp).eq.0 .and. idosu(idgrp).eq.0) go to 460  ! End of define.

       if(idosl(idgrp).le.0) idosl(idgrp)=1
       if(idosu(idgrp).le.0 .or. idosu(idgrp).ge.imax) idosu(idgrp)=imax
       if(jdosl(idgrp).le.0) jdosl(idgrp)=1
       if(jdosu(idgrp).le.0 .or. jdosu(idgrp).ge.jmax) jdosu(idgrp)=jmax
       if(kdosl(idgrp).le.0) kdosl(idgrp)=1
       if(kdosu(idgrp).le.0 .or. kdosu(idgrp).ge.kmax) kdosu(idgrp)=kmax

       write(ifto,450) idosl(idgrp),idosu(idgrp),jdosl(idgrp),
      *            jdosu(idgrp),kdosl(idgrp),kdosu(idgrp),izscan(idgrp)
450    FORMAT('+',T5,3(I6,I4),I6)

       go to 420

460    continue

       idgrp=idgrp-1

       if(idgrp.gt.LMXDOS) then
         write(ifto,470) idgrp,LMXDOS
470      FORMAT(' idgrp(=',I5,') must be less than LMXDOS(=',I5,')'/
      *         ' Or you must chnage LMXDOS in xyzdose_h.f')
       end if
```

```
! -----------------
! Record 12: iwatch
! -----------------
      read(4,*) iwatch

      write(ifto,790) iwatch
790   FORMAT(//,' SWATCH tracking switch: iwatch=',I2,
     *          ' (0=off, 1=each interaction, 2=each step)')

! ---------------------------------------
! Record 13: ibrdst,iprdst,ibrspl,nbrspl0
! ---------------------------------------
      read(4,*) ibrdst,iprdst,ibrspl0,nbrspl
      write(ifto,800) ibrdst,iprdst,ibrspl0,nbrspl
800   FORMAT(/,' IBRDST=',I2,/,' IPRDST=',I2,/,' IBRSPL0=',I2,
     *' (NBRSPL=',I5,')')

      if (ibrspl0 .gt. 0) then
        if (nbrspl .gt. 0) then
          fbrspl = 1.0/float(nbrspl)
          do i=1,nreg
            if(med(i).ne.0) ibrspl(i)=1  ! Apply splitting to all region
          end do
        else
          write(ifto,810) ibrspl0,nbrspl
810       FORMAT(//,' Stopped in Getvoxel with IBRSPL=',
     *         I5,' and NBRSPL=',I5)
          stop
        end if
      end if

                                                              ! --------------
      return                                                  ! Return to MAIN
                                                              ! --------------
      end

!-------------------------last line of getvoxel.f------------------------
!-------------------------------ausgab.f---------------------------------
! Version:   031108-1300
!            060801-1000
! Reference: SLAC-265 (p.19-20, Appendix 2)
!------------------------------------------------------------------------
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12

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

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

      subroutine ausgab(iarg)

      implicit none

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

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

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

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

                                          ! Variables to score
      common/score/                             ! Variables to score
     *       depe(LIMAX,LJMAX,LKMAX),faexp,fexps,maxpict
      real*8 depe,faexp,fexps
      integer maxpict
```

```
      integer                                                  ! Arguments
     * iarg

      real*8                                                ! Local variables
     * cmod,dcon,edepwt,encoea,esing

      integer i,irl,irx,iry,irz,iql,j,k

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

!  ---------------------------------
!  Keep track of how deep stack gets
!  ---------------------------------
        if (np.gt.MXSTACK) then
          write(6,100) np,MXSTACK
100       FORMAT(//' In AUSGAB, np=',I3,' >= maximum stack',
     *         ' allowed which is',I3/1X,79('*')//)
          stop
        end if
!     ------------------------
!     Set some local variables
!     ------------------------
      irl = ir(np)
      iql = iq(np)
      edepwt = edep*wt(np)

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

!     ------------------------------------------------------------
!     Keep track of energy deposition (for conservation purposes)
!     ------------------------------------------------------------
      if (iarg .gt. 5) return

      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

      i=mod(irl-1,imax)
      if (i.eq.0) i=imax
      k=1+(irl-1-i)/ijmax
      j=1+(irl-1-i-(k-1)*ijmax)/imax

      if (irl.gt.1.and.edep.ne.0.D0) then
        depe(i,j,k)=depe(i,j,k)+edepwt
      end if

!---------------------------------
!     Check cross phantom surface
!---------------------------------
      if(i.eq.imax/2+1.and.j.eq.jmax/2+1) then  ! X-Y central region
        if (abs(irl-irold).eq.ijmax.and.iq(np).eq.0) then
          if ((w(np).gt.0.0.and.k.eq.2).or.
     *     (w(np).le.0.0.and.k.eq.1)) then
            if (dabs(w(np)).ge.0.0349) then
              cmod=dabs(w(np))
            else
              cmod=0.01745
            end if
            esing=e(np)
            dcon=encoea(esing)            ! PHOTX data
            fexps=fexps+e(np)*dcon*wt(np)/cmod
            if (w(np).lt.0.0) latch(np)=1
            if (w(np).gt.0.0.and.latch(np).eq.0) then
              faexp=faexp+e(np)*dcon*wt(np)/cmod
            end if
          end if
        end if
```

```
      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:   160120-1000
! Reference: SLAC-265 (p.19-20, Appendix 2)
! use geoxyzv.f for geometry information instead of geoxyz.f
!-----------------------------------------------------------------------
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12

! ----------------------------------------------------------------------
! Required (geometry) subroutine for use with the EGS5 Code System
! ----------------------------------------------------------------------
!   HOWFAR routine to use with a generalized cartesian coordinate system
!   for voxel geometry.
!
!   Geometrical information is passed in common/geoxyzv
!    xbound(MXXPLNS+1),ybound(MXYPLNS+1),zbound(MXZPLNS+1),imax,jmax,
!    kmax,ijmax,irmax
!     xbound etc are the X, Y and Z boundaries defining the voxels
!     MXXPLNS etc are the maximum number of planes in each direction
!           as defined in the auxiliary-code header file.
!     imax etc are the actual number of elements in each direction for
!           this particular calculation
!     ijmax = imax*jmax   a useful number
!     irmax = 1 + ijmax*kmax  the total number of regions in the
!           current problem
!
!     Each voxel is defined by a triple of integers (i,j,j) (but called
!           irx,iry and irz in this routine) such that:
!
!           xbound(i) <= x < xbound(i+1)     1 < i < imax
!           ybound(j) <= y < ybound(j+1)     1 < j < jmax
!           zbound(k) <= z < zbound(k+1)     1 < k < kmax
!
!  The X axis is up the page, the Y axis to the right and Z into the page
!
!  The region number is defined as:
!       ir = 1 + i + (j-1)*imax + (k-1)*ijmax
!
!  The routine sets DNEAR  Note that in problems where the typical
!  step size is of the order of the region dimensions, then computing
!  DNEAR can decrease efficiency. In this case the two lines containing
!  DNEAR should be commented out
!
! ----------------------------------------------------------------------

      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'

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

                                          ! Auxiliary-code COMMONs
      include 'auxcommons/geoxyzv.f'
      include 'auxcommons/instuf.f'

      real*8                                           ! Local variables
     * dist,dnearl

      integer
     * irl,irx,iry,irz

      irl = ir(np)
```

```
      if (irl .le. 0) then
        write(6,*) 'Stopped in howfar with irl <= 1'
        stop
      end if

      if (irl .eq. 1) then
        idisc = 1  ! -----------------------------------------------------
        return     ! Particle outside geometry - return to ELECTR/PHOTON
      end if       ! -----------------------------------------------------

!     ---------------------------
!     Get irx, iry and irz indices
!     ---------------------------
      irx=mod(irl-1,imax)
      if (irx.eq.0) irx=imax
      irz=1+(irl-1-irx)/ijmax
      iry=1+(irl-1-irx-(irz-1)*ijmax)/imax

      dnearl = 1.D10

!     -----------------
!     Check Z-direction
!     -----------------
      dnearl=min(dnearl,(zbound(irz+1)-z(np)),(z(np)-zbound(irz)))
      if (w(np) .gt. 0.0) then
        dist = (zbound(irz+1)-z(np))/w(np)
        if (dist .lt. ustep) then
          ustep=dist
          if (irz .ne. kmax) then
            irnew=irl+ijmax
          else
            irnew=1
          end if
        end if
      else if (w(np) .lt. 0.0) then
        dist = -(z(np) - zbound(irz))/w(np)
        if (dist .lt. ustep) then
          ustep = dist
          if (irz .ne. 1) then
            irnew=irl-ijmax
          else
            irnew = 1
          end if
        end if
      end if

!     -----------------
!     Check X-direction
!     -----------------
      dnearl=min(dnearl,(xbound(irx+1)-x(np)),(x(np)-xbound(irx)))
      if (u(np) .gt. 0.0) then
          dist = (xbound(irx+1)-x(np))/u(np)
          if (dist .lt. ustep) then
              ustep=dist
              if (irx .ne. imax) then
                  irnew=irl+1
              else
                  irnew=1
              end if
          end if
      else if (u(np) .lt. 0.0) then
          dist = -(x(np) - xbound(irx))/u(np)
          if (dist .lt. ustep) then
              ustep = dist
              if (irx .ne. 1) then
                  irnew=irl-1
              else
                  irnew = 1
              end if
          end if
      end if

!     -----------------
!     Check Y-direction
!     -----------------
      dnearl=min(dnearl,(ybound(iry+1)-y(np)),(y(np)-ybound(iry)))
      if (v(np) .gt. 0.0) then
          dist = (ybound(iry+1)-y(np))/v(np)
```

```
          if (dist .lt. ustep) then
              ustep=dist
              if (iry .ne. jmax) then
                  irnew=irl+imax
              else
                  irnew=1
              end if
          end if
      else if (v(np) .lt. 0.0) then
          dist = -(y(np) - ybound(iry))/v(np)
          if (dist .lt. ustep) then
              ustep = dist
              if (iry .ne. 1) then
                  irnew=irl-imax
              else
                  irnew = 1
              end if
          end if
      end if

      dnear(np)=dnearl

                                                ! ------------------------
      return                                    ! Return to ELECTR/PHOTON
                                                ! ------------------------
      end

!-------------------------last line of howfar.f-------------------------
!-------------------------------encoea.f--------------------------------
! Version:   030831-1300
!-----------------------------------------------------------------------
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12
!
!     double precision function encoea(energy)
!     Function to evaluate the energy absorption coefficient of air.
!      (Tables and Graphs oh photon mass attenuation coefficients and
!      energy-absorption coefficients for photon energies 1 keV to
!      20 MeV for elements Z=1 to 92 and some dosimetric materials,
!      S. M. Seltzer and J. H. Hubbell 1995, Japanese Society of
!      Radiological Technology)
!-----------------------------------------------------------------------
      double precision function encoea(energy)

      real*8 hnu(38)/0.001,0.0015,0.002,0.003,0.0032029,0.0032029,
     *           0.004,0.005,0.006,0.008,0.01,0.015,0.02,0.03,0.04,
     *           0.05,0.06,0.08,0.10,0.15,0.2,0.3,0.4,0.5,0.6,0.8,1.0,
     *           1.25,1.5,2.0,3.0,4.0,5.0,6.0,8.0,10.0,15.0,20.0/

      real*8 enmu(38)/3599.,  1188.,  526.2,  161.4,  133.0,  146.0,
     *      76.36, 39.31,  22.70,  9.446,  4.742,  1.334,  0.5389,
     *      0.1537,0.06833,0.04098,0.03041,0.02407,0.02325,0.02496,
     *      0.02672,0.02872,0.02949,0.02966,0.02953,0.02882,0.02789,
     *      0.02666,0.02547,0.02345,0.02057,0.01870,0.01740,0.01647,
     *      0.01525,0.01450,0.01353,0.01311/

      real*8 energy,enm1,hnu1,ene0,slope

      integer i

      if (energy.gt.hnu(38)) then
        encoea=enmu(38)
        return
      end if
      if (energy.lt.hnu(1)) then
        encoea=enmu(1)
        return
      end if

      do i=1,38
        if(energy.ge.hnu(i).and.energy.lt.hnu(i+1)) then
          enm1=dlog(enmu(i+1))
          enm0=dlog(enmu(i))
          hnu1=dlog(hnu(i+1))
          hnu0=dlog(hnu(i))

          ene0=dlog(energy)
          slope=(enm1-enm0)/(hnu1-hnu0)
          encoea=exp(enm0+slope*(ene0-hnu0))
          return
        end if
```

```
        if(energy.eq.hnu(i+1)) then
          encoea=enmu(i+1)
          return
        end if
      end do

! If sort/interpolation cannot be made, indicate so by writing
! a comment and stopping here.
      write(6,100) energy
100   FORMAT(///,' *****STOPPED IN ENCOEA*****',/,' E=',G15.5,///)
      return
      end

!-------------------------last line of encoea.f-----------------------
!-----------------------------decoe.f---------------------------------
! Version:   100302-1000
!-------------------------------------------------------------------------
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12
! ------------------------------------------------------------------------
! Function to evaluate the ratio of ambient dose equivalent to air (Sv/Gy).
! Data tanken from ICRP pub 74 (1996).
! ------------------------------------------------------------------------
      double precision function decoe(energy)

      implicit none

      real*8 energy, slope
      integer i

      real*8 hnu(25)/
     *  0.01,0.015,0.02,0.03,0.04,0.05,0.06,0.08,
     *  0.10,0.15,0.2,0.3,0.4,0.5,0.6,0.8,1.0,1.5,2.0,
     *  3.0,4.0,5.0,6.0,8.0,10.0/

      real*8 enmu(25)/0.008,0.26,0.61,1.10,1.47,1.67,1.74,1.72,
     *  1.65,1.49,1.40,1.31,1.26,1.23,1.21,1.19,1.17,1.15,1.14,
     *  1.13,1.12,1.11,1.11,1.11,1.10/

      if(energy.gt.hnu(25)) then
        decoe=enmu(25)
        return
      end if

      if (energy.lt.hnu(1)) then
        decoe=enmu(1)
        return
      end if

      do i=1,25
        if(energy.ge.hnu(i).and.energy.lt.hnu(i+1)) then
          slope=(dlog(enmu(i+1))-dlog(enmu(i)))/
     *          (dlog(hnu(i+1))-dlog(hnu(i)))
          decoe=dlog(enmu(i))+slope*(dlog(energy)-dlog(hnu(i)))
          decoe=exp(decoe)
          return
        end if
        if(energy.eq.hnu(i+1)) then
          decoe=enmu(i+1)
          return
        end if
      end do

! If sort/interpolation cannot be made, indicate so by writing
! a comment and stopping here.

      write(3,100) energy
100   format(///,' **** Stopped in decoe ****',/,' E=',G15.5,///)
      stop

      return
      end

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