

egs5 サンプルプログラム (ucxyz_phantom.f)
人体中の線量分布計算
(Draft, July 20, 2004)

平山 英夫、波戸 芳仁

〒 305-0801 茨城県つくば市大穂 1 - 1
高エネルギー加速器研究機構

Contents

1. サンプルプログラム ucxyz_phantom.fの概要	1
2. ユーザーコードの内容	2
2.1. メインプログラム	2
2.1.1. include 文及び型式宣言:	2
2.1.2. open 文:	3
2.1.3. subroutine getvoxel の call:	4
2.1.4. 計算モードの選択:	4
2.1.5. 入射粒子のパラメータ:	4
2.1.6. X線源の種類を増やす方法:	5
2.1.7. 輸送計算:	6
2.1.8. 統計誤差:	8
2.1.9. 計算結果の出力:	9
2.2. subroutine getvoxel	10
2.3. subroutine ausgab	11
2.4. subroutine howfar	13
3. 実習課題	14
3.1. 実習課題 1 : 線源を Cs-137 に変更する	14
3.2. 実習課題 2 : 線源を Co-60 に変更する	14
3.3. 実習課題 3 : 肺のモデルに変更する	14
3.4. 実習課題 4 : 腫瘍を含む肺	14
3.5. 実習課題 5 : 金属の挿入	14
3.6. その他	14
4. 実習課題の解答例	15
4.1. 実習課題 1	15
4.2. 実習課題 2	15
4.3. 実習課題 3	15
4.4. 実習課題 4	16
4.5. 実習課題 5	16

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方向のビン数 22
- Y方向のビン数 3
- X方向のビン数 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

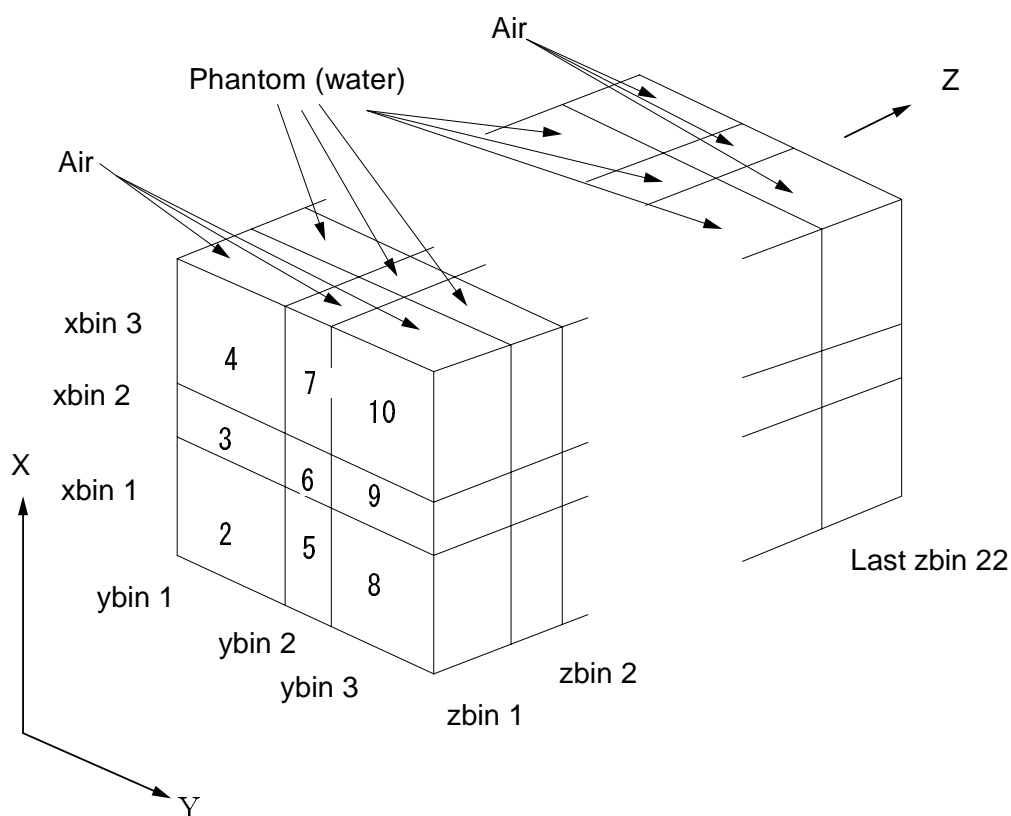


Figure 1: ucxyz_phantom.fのジオメトリー。

2. 線源条件

- 粒子のエネルギーは、isemode=0の時は、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、isemode=1の時は、ユニット4の入力データを用いてサンプリングする。
- 点等方線源:位置は、人体表面からの距離 (SPOSI) で指定
- ビームサイズ: 人体表面で XHBEAM*2 × YHBEAM*2 のビーム。XHBEAM 及び YHBEAM は、キー入力。

3. 計算モード

以下の2つのモードがある。目的によって切り替えて使用する。

- 飛跡表示システムを使って、飛跡を表示させるためのデータを作成するモード (imode=0)
egs5job.pic に飛跡データを出力
- 多くのヒストリーを使用して線量分布を計算するモード (imode=1)
egs5job.out に結果を出力

4. 得られる情報

(a) 飛跡表示モード (imode=0)

- 飛跡情報 (egs5job.pic)
- コンソール上に、ファントム中心の 1cm × 1cm の領域の深度線量分布 (1cm 単位)
- コンソール上に、後方散乱係数 (ファントム中心の 1cm × 1cm の領域での、ファントムがない場合の照射線量に対するファントムがある場合の照射線量の比)。照射線量は、光子のエネルギー束と空気エネルギー吸収係数から計算

(b) 線量分布計算モード (imode=1)

- 使用する物質に関するデータ
- 各リージョンに関する情報
- 定義した平板に関するデータ
- サンプルした X 線スペクトルと xray.dat から読み込んだスペクトルの比較
- ヒストリー数、ビームサイズ
- ファントム中心の 1cm × 1cm の領域の深度線量分布 (1cm 単位)
- 後方散乱係数
- ユニット 4 から入力する出力指定情報により、各リージョン中での吸収線量とその統計誤差

2. ユーザーコードの内容

2.1. メインプログラム

2.1.1. include 文及び型式宣言: egs5 は、Fortran で書かれているので、egs5 やジオメトリーや、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 では、egs5 に直接関係する include 関係のファイルは、egs ディレクトリーの include ディレクトリと、ジオメトリー関係のサブルーティンを含め、ユーザー固有のユーザーコードに関連した include 関係ファイルは、ユーザーのワーキングディレクトリー(以下では、user_dir とする。)のサブディレクトリー user_auxcommon ディレクトリーとリンクすることにより、使用できるようにしている。*

この点が、Morfran 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。従って、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、egs5.0/include/egs5_h.f 内の、その他の場合は、user_dir/user_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_elec.in.f'
include 'include/egs5_media.f'
include 'include/egs5_misc.f'
```

*これらの設定は、egs5run スクリプトで設定される。

```

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' 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。[†]

次の、設定は、ジオメトリ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 'user_auxcommons/edata.f'
include 'user_auxcommons/etaly1.f'
include 'user_auxcommons/geoxyz.f'
include 'user_auxcommons/instuf.f'
include 'user_auxcommons/lines.f'
include 'user_auxcommons/nfac.f'
include 'user_auxcommons/watch.f'

```

次の、etaly2.f は、準 egs5 的な扱いになっている common である。

```

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

```

特定のユーザーコード内で使用する common を次に定義する。

```

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

```

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。飛跡表示情報ファイルは、EGS4 では、ユニット 9 を使用していたが、egs5 では、39 に変更した。

```

!-----
!      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 afeter
!      getrz etc.  Unit for pict must be 39.
!-----

```

```

open(unit= 1,file='egs5job.out',status='unknown')
open(unit= 2,file='xray.dat',status='old')    ! Data of source x-ray
open(UNIT= 4,FILE='egs5job.inp',STATUS='old')
open(39,FILE='egs5job.pic',STATUS='unknown')

```

unit 2 の open 文は、X 線データを、xray.dat ファイルから読み込むことを定義しているものである。

[†]EGS4 の COMIN マクロに対応する扱いである。

2.1.3. subroutine getvoxel の call: 飛跡表示用データの型式定義 (この例では、pict32 を使用することを前提にしているので、nprec=1) の後で、2つの subroutine call を行う。最初のサブルーチンは、各種の counter パラメータをクリアするものである。2番目の getvoxel (名称は、ユーザーコードにより異なる。) が、egs5 で新たに必要なサブルーチンである。このサブルーチンをどの様に設定するかは、個々のユーザーコードにより異なるが、最低必要な機能は、pegs の実行と、subroutine hatch を call することである。このユーザーコードでは、物質の指定、カットオフエネルギー、オプション、入射粒子等の設定を全てこのサブルーチンにおいて、ユニット 4 から読み込むデータで設定する様にしている。

```
!-----
!      Define pict data mode.
!-----
      nprec=1

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

!      =====
!      call getvoxel(nreg)
!      =====
```

2.1.4. 計算モードの選択: このユーザーコードでは、最初に説明した様に、飛跡表示用データ作成モード (imode=0) と線量計算モードの両方対応している。モードの選択は、次により、キーボードより入力したデータで行う。

```
      write(6,100)
100  FORMAT(' Key in mode. 0:trajectory display, 1:dose calculation')
      read(5,*) imode
```

線量計算モードにおいては、getvoxel で作成した物質データ及び、X-Y 平面の中心領域の各リージョンの物質情報を出力するようにしている。

2.1.5. 入射粒子のパラメータ: 入射粒子のパラメータを設定する。線源の位置は、キーボードからの入力指定する。

```
      write(6,180)
180  FORMAT(' Key in source position from phantom surface in cm')
      read(5,*) sposi
```

粒子のエネルギーについては、isemode で、xray.dat から読み込んだ 100kV の X 線スペクトルを使用するか、egs5job.inp で設定した値を使用するかどうかを決める。

```
!-----
!      Source energy sampling mode
!      isemode=0 use xray.dat
!      isemode=1 use egs5job.inp
!-----
      isemode=0
```

imode=0 の時は、xray.dat からデータを読み込み、読み込んだ確立分布関数 (pdf) から、累積分布関数 (cdf) を計算する。読み込むデータは、ビン数 (nofebin)、ビンのエネルギー幅 (deltae:MeV 単位)、各ビンの X 線数 (sspec) である。

ファントム表面での照射野の半値幅は、X-方向と Y-方向それぞれ別にキーボードから指定する。入力された値に伴い、Z-方向の方向余弦の最小値を計算する。

```
!-----
!      Key in half width and height at phantom surface
!-----
      write(6,220)
220  FORMAT(' Key in half width of beam at phantom surface in cm.')
```

```

        read(5,*) xhbeam
        write(6,230)
230   FORMAT(' Key in half height of beam at phantom surface in cm.')
```

```

        read(5,*) yhbeam
        radma2=xhbeam*xhbeam+yhbeam*yhbeam
        wimin=sposi/dsqrt(sposi*sposi+radma2)
```

ヒストリー数 (ncases) は、コンソールから入力するようになっている。ncases の値として、0 を入力すると計算が終了する。0 以外の値を入力すると、新たなバッチとして処理される。

2.1.6. X 線源の種類を増やす方法: ucxyz_phantom.f X 線源スペクトルは、1 個しかないが、複数の線源を用意したい場合には、以下のようにする。

1. X 線源数の変更

```

        real*8
        * depeh(LIMAX,LJMAX,LKMAX),depeh2(LIMAX,LJMAX,LKMAX),
        * dose(LIMAX,LJMAX,LKMAX),doseun(LIMAX,LJMAX,LKMAX),
        * ebint(201),nofebin(1),deltae(1),sspec(1,201),ecdf(201),
        * saspec(201)
```

で nofebin(1),deltae(1),sspec(1,201) 中の、1 を使用する X 線源の数に変更する。また、201 を、使用する X 線源中で、最も多いピン数の値に変更する。

2. xray.dat に、新たな線源に関するデータ (ピン数 (nofebin)、ピンのエネルギー幅 (deltae:MeV 単位)、各ピンの X 線数 (sspec)) を追加する。
3. データの読み込み及び X 線源を選択する部分を変更する。例えば、60kV, 80kV 及び 100kV から選択する場合 (スペクトルデータは、60kV, 80kV, 100kV の順に書かれているとする) には、

```

!-----
!       Read spectrum pdf
!-----
        do i=1,1
            read(2,*) nofebin(i)
            read(2,*) deltae(i)
            read(2,*) (sspec(i,ie),ie=1,nofebin(i))
        end do

!-----
!       Select source type
!-----
190   write(6,200)
200   FORMAT(' Key in source type. 1:100kV')
        read(5,*) ixtype
        if (ixtype.eq.0.or.ixtype.gt.1) then
            write(6,210)
210   FORMAT(' IXTYPE must be >0 <= $NXTYPE.')
```

```

        go to 190
        end if
```

を、

```

!-----
!       Read spectrum pdf
!-----
        do i=1,3
            read(2,*) nofebin(i)
            read(2,*) deltae(i)
            read(2,*) (sspec(i,ie),ie=1,nofebin(i))
        end do

!-----
!       Select source type
```

```

!-----
190  write(6,200)
200  FORMAT(' Key in source type. 1:100kV, 2:80kV, 3:100kV')
      read(5,*) ixtype
      if (ixtype.eq.0.or.ixtype.gt.3) then
          write(6,210)
210  FORMAT(' IXTYPE must be >0 <= 3.')
          go to 190
      end if

```

に変更する。

4. 出力部で、線源に関する部分 (569-572 行目) を変更する。

```

      write(1,390) sposi
390  FORMAT(/' Absorbed energy inside phantom for 100 kV X-ray'/ ' So
      *urce position ',F10.1,' cm from phantom surface'/ ' Within 1cm x 1
      *cm area after 5 cm air')

```

を、

```

      if (ixtype.eq.1) then
          ixen=60
      elseif (ixtype.eq.2) then
          ixen=80
      else
          ixen=100
      end if
      write(1,390) ixen,sposi
390  FORMAT(/' Absorbed energy inside phantom for ',I4,'kV X-ray'/
      *      ' Source position ',F10.1,' cm from phantom surface'/
      *      ' Within 1cm x 1cm area after 5 cm air')

```

5. 新たに使用する ixen を integer 宣言文に加える。

2.1.7. 輸送計算: 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、エネルギー分布を持った X 線が出るので、線源光子の方向、sposi が空気の厚さ (5cm) より長い場合の空気層の表面の位置での位置及びエネルギー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。

```

      do jhist=1,ncases
!-----
!      Start of CALL SHOWER loop
!-----
          icases=j
!-----
!      Determine direction (isotropic)
!-----
280  call randomset(w0)
      win=w0*(1.0-wimin)+wimin
      call randomset(phai0)
      phai=pi*(2.0*phai0-1.0)
      sinth=dsqrt(1.00-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 280
      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.DO
        yin=0.DO
        zin=-sposi
    end if

do i=1,imax
    if (xbound(i+1).gt.xin) go to 290
end do

290 do j=1,jmax
    if (ybound(j+1).gt.yin) go to 300
end do

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

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

    if (isemode.eq.0) then   ! use xray.dat
        call randomset(ei0)
        do ie=2,nsebin
            if (ei0.lt.ecdft(ie)) then
                go to 310
            end if
        end do

310    if (ie.gt.nsebin) then
        ie=nsebin
    end if
    saspec(ie)=saspec(ie)+1.DO
    ekin=ebint(ie-1)+(ei0-ecdft(ie-1))*(ebint(ie)-ebint(ie-1))/
*   (ecdft(ie)-ecdft(ie-1))
    wtin = 1.0
    else
        ! use egs5job.inp
        ! Monoenergetic case
        if (isamp .eq. 0) then
            ekin = ekein
            wtin = 1.0
        else if (isamp .eq. 1) then   ! Sample discrete energy from CDF
            call randomset(rnnow)
            i=0
312    continue
            i = i + 1
            if(ecdf(i) .le. rnnow) go to 312
            ekin = ebin(i)
            wtin = 1.0
        else if (isamp .eq. 2) then   ! Sample DIRECTLY from CDF
            call edistr(ekin)
            wtin = 1.0
        else if (isamp .eq. 3) then   ! Sample UNIFORMLY on energy
            ! interval and WEIGHT
            call randomset(rnnow)
            ekin = esam1 + rnnow*delsam
            isam = 0
314    continue
            isam = isam + 1
            if (ekin .lt. ebin(isam)) go to 316
            go to 314
316    continue
            wtin = epdf(isam)
        end if
    end if
end if

```

```

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

latchi=0

! -----
! Print first NWRITE or NLINES, whichever comes first
! -----
if (ncount .le. nwrite .and. ilines .le. nlines) then
  ilines = ilines + 1
  write(6,320) etot,xin,yin,zin,uin,vin,win,iqin,irin,idin
320  FORMAT(4G15.7/3G15.7,3I5)
end if

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

! Added for energy balance tests (SJW)
if(DABS(eparte + epartd - ekin)/ekin .gt. 1.d-10) then
  write(*,330) icases, eparte, epartd
330  FORMAT('Error on # ',I6,' Escape = ',F9.5,' Deposit = ',F9.5)
endif

! -----
! Sum variable and its squqre.
! -----

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+fexpss
fexpss2s=fexpss2s+fexpss*fexpss
fexpss=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
                                ! -----

```

2.1.8. 統計誤差: x をモンテカルロ計算で計算したい量(スコアする量)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:

$$\bar{x} = \frac{1}{N} \sum_{i=1}^N x_i \quad (1)$$

- x_i の分散値を以下の式から求める。:

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

- \bar{x} の分散値は、

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

となる。

- 統計誤差として、

$$R = s_{\bar{x}}/\bar{x} \simeq \left[\frac{1}{N} \left(\frac{\overline{x^2}}{\bar{x}^2} - 1 \right) \right]^{1/2} \quad (4)$$

を用いる。

このために、ヒストリー毎に、計算すべき量とその自乗の値を保存している。

2.1.9. 計算結果の出力: 得られた結果を処理して打ち出す処理を行う。線量計算モードで、xray.dat から粒子のエネルギーをサンプリングする場合には、最初に、サンプリングした X 線のスペクトルと、元の X 線スペクトルとの比較を出力し、その後、線源の条件 (線源のタイプ、位置)、ヒストリー数を出力する。

```

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

      if (imode.ne.0) then
        write(1,370)
370      FORMAT(//' Comparison between sampled spectrum and original data
*'/ 23X,'   Sampled   Probability',25X,'   Sampled   Probability'
* )
        do ie=2,nsebin,2
          write(1,380) ebint(ie),saspec(ie),ecdft(ie)-ecdft(ie-1),
*
*   ebint(ie+1), saspec(ie+1),ecdft(ie+1)-ecdft(ie)
380      FORMAT(1X,G9.3,' MeV(upper)-- ',2G12.5,3X, '; ',G9.3,' MeV(upp
*er)-- ',2G12.5)
          end do

          write(1,390) sposi
390      FORMAT(/' Absorbed energy inside phantom for 100 kV X-ray'/ ' So
*urce position ',F10.1,' cm from phantom surface'/ ' Within 1cm x 1
*cm area after 5 cm air')
          write(1,400) ncases, xhbeam, yhbeam
400      FORMAT(1X,I8,' photons normally incident from front side'/ ' Hal
*f width of beam is ',G15.5,'cm for X and ',G15.5,'cm for Y')
          end if

```

その後、各ボクセルの吸収線量とその統計的な誤差を求める。得られた結果を基に、ファントムの中心領域での Z-方向の吸収線量分布を出力する。また、入射粒子による照射線量、ファントム表面での照射線量及び後方散乱係数とそれぞれの誤差を求めて、出力する。

getvoxel でユニット 4 からの入力データに基づき設定した出力領域を、指定した方法 (Z-scan 又は X-scan) で出力する。

2.2. subroutine getvoxel

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、ジオメトリ情報、入射粒子に関する情報、初期乱数の設定等をユニット 4 から読み込み、必要な設定を行い、pegs を実行し、その後に subroutine hatch を call する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 : 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。
5. Record 5 : X-方向の平面の位置
 - maxx > 0 の時
maxx + 1 個、1 行に 1 カ所ずつ値を指定
 - 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
6. Record 6 : Y-方向の平面の位置
7. Record 7 : Z-方向の平面の位置
8. Record 8 : 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。
9. Record 8a : Record 8 で、medtmp ≠ 0 の場合には、各種オプション設定を指定する。(0: off, 1:on)
 - ipeangsw Switches for PE-angle sampling
 - iedgesw K & L-edge fluorescence
 - iraysw Rayleigh scattering
 - ipolarsw Linearly-polarized photon scattering
 - incohrsw S /Z rejection
 - iprofrsw Doppler broadening
 - mpacrsw electron impact ionization
10. Record 9 : 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。
11. Record 10 : 入射粒子の X-方向の範囲指定 (xlower, xupper)。xlower=xupper=0 の時は、X-方向の領域の中心のビームとする。
12. Record 11 : 入射粒子の Y-方向の範囲指定 (ylower, yupper)。
13. Record 12 : 入射粒子の各軸に対する角度 (thetaz,thetax,thetay) の指定。
thetaz = 0 の時は、他の角度は 90 と扱う。thetaz ≠ 0 で、thetax = thetay = 0 の時は、thetax を cos(thetax) が最大となる値に設定する。thetax ≠ 0 の時は、cos(thetax) の可能な最大値を超えていないかどうかを調べ、越えている場合は、最大値になる角度に設定。
14. Record 13 : 最初の random number seed(ixx, jxx) を指定。
ixx = 0, の時は、ixx= 123457 に、jxx = 0, の時は、jxx= 654321 に設定する。
15. Record 14 : ヒストリー数 (ncases) の指定。

16. Record 15 : 入射粒子の運動エネルギー (ekein:MeV), 電荷 (iqin) 及びサンプリング方法 (isamp) の指定。
isamp=0 の時は、ekein の単一エネルギー、isamp=1 の時は、離散エネルギーからサンプリングを行い、isamp=2 の時は、スペクトルからサンプリングを行い、isamp=3 の時は、一様サンプリングを行いウエイトを使用する手法を用いることを意味する。isamp ≠ 0 の時は、以下のデータが必要である。
17. Record 15a : 最低エネルギーの指定。(isamp>1 の時)
18. Record 15b : 各エネルギービンの最大値 (ebin(i)) とビンに対応する確率 (epdf(i)) の指定。
blank又は、0.0 は、指定の終了を意味する。
19. Record 16 : 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
20. Record 17 : 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トングパラメータ (ibrspl,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
ibrspl=0 スプリットング使用せず
ibrspl=1 nbrspl にスプリットング
21. Record 18 : 電子輸送に使用するパラメータ (estepe,estepe2) の指定

2.3. 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 オプションの伴う処理、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。

飛跡表示モードの場合は、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

! added SJW for particle by particle energy balance
    if(irl.eq.1) then
        eparte = eparte + edepwt
    else
        epartd = epartd + edepwt
    end if

    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-iold).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
            end if
            esing=e(np)
            dcon=encoa(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

! -----
! Output particle information for plot
! -----
    if (imode.eq.0) then
        call plotxyz(iarg,np,iq(np),x(np),y(np),z(np),e(np),ir(np),
*           w(np))
    end if

    return

```

end

2.4. 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 形状のルーチンを使用している。

3. 実習課題

3.1. 実習課題1：線源を Cs-137 に変更する

線源を、Cs-137 の単一エネルギー光子 (0.662MeV) に変える。

3.2. 実習課題2：線源を Co-60 に変更する

線源を Co-60 に変え、1.173MeV と 1.332MeV 光子を同じ確率で発生させる。

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 $0.3\text{g}/\text{cm}^3$) とし、その背後の 3cm の人体組織がある体系に変更する。線源は、元の X 線とする。

3.4. 実習課題4：腫瘍を含む肺

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

3.5. 実習課題5：金属の挿入

ファントムから 5cm-6cm の領域を鉄に変える。線源は、元の X 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

4.1. 実習課題 1

1. 線源エネルギー選択モード isemode=0 を isemode=1 に変更する。
2. ucxyz_phantom.data の 34 行目の ekinin の値を 0.662 に変更する。
3. ucxyz_phantom.data を別な名前で保存し、egs5run の実行時にユニット 4 のファイルとして、保存したファイル名を指定する。

4.2. 実習課題 2

1. isemode=1 の状態で、ucxyz_phantom.data の 34 行目の ekinin の値を 1.332 に、isamp を 1 に変更する。
2. 上記の後に、

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

を挿入する。

3. ucxyz_phantom.data を別な名前で保存し、egs5run の実行時にユニット 4 のファイルとして、保存したファイル名を指定する。

4.3. 実習課題 3

1. 線源エネルギー選択モードを isemode=0 に戻す。
2. ucxyz_phantom.data の 18 行目のデータを '1.0, 20' から '1.0, 16' に、22-25 行目の

```
1,3,1,3, 2,21, 1, 0.000, 0.00, 0.00          tissue
 1 1 0 0 0 0 0          peang,edge,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          peang,edge,ray,pola,incoh,prof,impac
```

を

```
1,3,1,3, 2, 4, 1, 0.000, 0.00, 0.00          tissuey
 1 1 0 0 0 0 0          peang,edge,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          peang,edge,ray,pola,incoh,prof,impac
1,3,1,3,15,17, 1, 0.300, 0.00, 0.00          tissue
 1 1 0 0 0 0 0          peang,edge,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          peang,edge,ray,pola,incoh,prof,impac
```

に変更する。

3. ucxyz_phantom.data を別な名前で保存し、egs5run の実行時にユニット 4 のファイルとして、保存したファイル名を指定する。
4. 飛跡表示のジオメトリとして、肺の前後の面を追加

```
do j=1,8
  pcoord(1,j)=0.0
  pcoord(2,j)=0.0
  pcoord(3,j)=0.0
  pnorm(1,j)=0.0
  pnorm(2,j)=0.0
```

```

    pnorm(3,j)=0.0
end do

pcoord(3,1)=0.0
pnorm(3,1)=1
pcoord(3,2)=zbound(5)
pnorm(3,2)=1
pcoord(3,3)=zbound(15)
pnorm(3,3)=1
pcoord(3,4)=zbound(kmax)
pnorm(3,4)=1
pcoord(2,5)=ybound(1)
pnorm(2,5)=1.0
pcoord(2,6)=ybound(jmax+1)
pnorm(2,6)=1.0
pcoord(1,7)=xbound(1)
pnorm(1,7)=1.0
pcoord(1,8)=xbound(imax+1)
pnorm(1,8)=1.0

call geomout(0,8)

```

4.4. 実習課題4

1. ucxyz_phantom.data の 18 行目のデータを '1.0, 20' から '1.0, 16' に、22-25 行目の

```

1,3,1,3, 2,21, 1, 0.000, 0.00, 0.00      tissue
 1 1 0 0 0 0 0      peang,edge,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      peang,edge,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      peang,edge,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      peang,edge,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      peang,edge,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      peang,edge,ray,pola,incoh,prof,impac
1,3,1,3,15,17, 1, 0.300, 0.00, 0.00      tissue
 1 1 0 0 0 0 0      peang,edge,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      peang,edge,ray,pola,incoh,prof,impac

```

に変更する。

2. ucxyz_phantom.data を別な名前で保存し、egs5run の実行時にユニット 4 のファイルとして、保存したファイル名を指定する。

4.5. 実習課題5

1. ucxyz_phantom.inp に次のデータを追加し、別な名前で保存する。

```

ELEM
  &INP IAPRIM=1,EFRACH=0.05,EFRACL=0.20,IRAYL=1,IBOUND=0,INCOH=0,
      ICPROF=0,IMPACT=0 /END
FE-IAPRIM      FE
FE
ENER
  &INP AE=0.521,AP=0.010,UE=2.511,UP=2.0 /END
PWLF

```

```

&INP /END
DECK
&INP /END
ELEM

```

2. ucxyz_phantom.data の 2 行目のデータを '2' から '3' に変え、4 行目の後に

```
FE-IAPRIM                                media(j,3) (24A1)
```

を挿入する。

3. 下記の 22-23 行目

```

1,3,1,3, 2,21, 1, 0.000, 0.00, 0.00      tissue
 1  1  0  0  0  0  0      peang,edge,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      peang,edge,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      peang,edge,ray,pola,incoh,prof,impac
1,3,1,3, 7, 7, 3, 0.000, 0.00, 0.00      Fe
 1  1  0  0  0  0  0      peang,edge,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      peang,edge,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      peang,edge,ray,pola,incoh,prof,impac

```

に変更する。

4. ucxyz_phantom.data を別な名前で保存し、egs5run の実行時にユニット 4 のファイルとして、保存したファイル名を指定する。


```

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

include 'user_auxcommons/edata.f'
include 'user_auxcommons/etaly1.f'
include 'user_auxcommons/geoxyz.f'
include 'user_auxcommons/instuf.f'
include 'user_auxcommons/lines.f'
include 'user_auxcommons/nfac.f'
include 'user_auxcommons/pladta.f'
include 'user_auxcommons/voxel.f'
include 'user_auxcommons/watch.f'

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

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

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

!**** real*8                                     ! Local variables
real*8
* amass,availke,depthl,depths,dis,disair,ei0,ekin,elow,eup,
* phai0,phai,radma2,rnow,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),
* ebint(201),nofebin(1),deltae(1),sspec(1,201),ecdf(201),
* saspec(201)

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

integer
* i,ii,iii,icases,idin,idose,ie,ifti,ifto,igmmax,imed,ipage,ireg,
* irl,isemode,isam,ixtype,j,jhist,jj,jl,ju,k,kkk,nlist,nnn,
* nperpg,nreg,nsebin

! -----
! 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
getrz etc.  Unit for pict must be 39.
-----

open(unit= 1,file='egs5job.out',status='unknown')
open(unit= 2,file='xray.dat',status='old')  ! Data of source x-ray
open(UNIT= 4,FILE='egs5job.inp',STATUS='old')
open(39,FILE='egs5job.pic',STATUS='unknown')

! -----
! Define pict data mode.
! -----
npreci=1

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

! =====
! call getvoxel(nreg)
! =====

! -----
! Selection mode form Keyboard.
! -----

```

```

100  write(6,100)
      FORMAT(' Key in mode. 0:trajectory display, 1:dose calculation')
      read(5,*) imode

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

!-----
! Set parameter for PICT32
!-----
      xmin=xbound(1)-1.0
      xmax=xbound(imax+1)+1.0
      ymin=ybound(1)-1.0
      ymax=ybound(imax+1)+1.0
      zmin=-5.0+zbound(1)
      zmax=zbound(kmax+1)+2.0

      do j=1,6
         pcoord(1,j)=0.0
         pcoord(2,j)=0.0
         pcoord(3,j)=0.0
         pnorm(1,j)=0.0
         pnorm(2,j)=0.0
         pnorm(3,j)=0.0
      end do

      pcoord(3,1)=0.0
      pnorm(3,1)=1
      pcoord(3,2)=zbound(kmax)
      pnorm(3,2)=1
      pcoord(2,3)=ybound(1)
      pnorm(2,3)=1.0
      pcoord(2,4)=ybound(jmax+1)
      pnorm(2,4)=1.0
      pcoord(1,5)=xbound(1)
      pnorm(1,5)=1.0
      pcoord(1,6)=xbound(imax+1)
      pnorm(1,6)=1.0

      call geomout(0,6)
      fnorm=dmax1(xmax-xmin+2,ymax-ymin+2,zmax-zmin)
      write(39,1200) xmin,xmax,ymin,ymax,zmin,zmax,fnorm
1200  FORMAT(7E10.3)

!-----
! Output medium and region information to file for calculation mode.
!-----
      if (imode.ne.0) then
110    write(1,110)
          FORMAT(' Quantities associated with each media:')
          do j=1,nmed
120      write(1,120) (media(i,j),i=1,24)
          FORMAT(/,1X,24A1)
130      write(1,130) rho(j),rlc(j)
          FORMAT(5X,' Rho=',G15.7,' g/cm**3      RLC=',G15.7,' cm')
          write(1,140) ae(j),ue(j),ap(j),up(j)
140      *  FORMAT(5X,' AE=',G15.7,' MeV      UE=',G15.7,' MeV' / 5X,' AP=',G
          *    15.7,' MeV      UP=',G15.7,' MeV')
          end do

150    write(1,150)
          FORMAT(/' Information of medium and cut-off for central region')
          i=imax/2+1
          j=jmax/2+1
          do k=1,kmax
             irl=i+i+(j-1)*imax+(k-1)*ijmax
             if (med(irl).eq.0) then
160      *  write(1,160) k,irl
          *    FORMAT(' Medium(',I3,'-th z bin, region:',I5,
          *          ')= Vacuum')

```

```

        else
          write(1,170) k,irl,(media(ii,med(irl)),ii=1,24),
*          ecut(irl),pcut(irl),rhor(irl)
170      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

    end if

!-----
!   Define source position from phantom surface.
!-----
    write(6,180)
180    FORMAT(' Key in source position from phantom surface in cm')
    read(5,*) sposi

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

!-----
!   Clear variables
!-----
!   Zero the dose
    do k=1,kmax
      do j=1,jmax
        do i=1,imax
          depe(i,j,k)=0.D0
          deph(i,j,k)=0.D0
          deph2(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
    fexpss2s=0.D0

    do i=1,201
      saspec(i)=0.D0
    end do

    iii=0

!-----
!   Source energy sampling mode
!   isemode=0 use xray.dat
!   isemode=1 use egs5job.inp
!-----
    isemode=0

    if (isemode.eq.0) then      ! use xray.dat
!-----
      Read spectrum pdf
!-----
      do i=1,1
        read(2,*) nofebin(i)
        read(2,*) deltae(i)
        read(2,*) (sspec(i,ie),ie=1,nofebin(i))
      end do

!-----
!   Select source type
!-----
190    write(6,200)
200    FORMAT(' Key in source type. 1:100kV')
    read(5,*) ixtype
    if (ixtype.eq.0.or.ixtype.gt.1) then
      write(6,210)
210    FORMAT(' IXTYPE must be >0 <= $NXTYPE.')
      go to 190
    end if

!-----

```

```

! Calculate CDF for selected source
!-----
      nsebin=nofebin(ixtype)
      tnum=0.D0
      do ie=1,nsebin
        tnum=tnum+sspec(ixtype,ie)
      end do

      ecdf(1)=0.0
      do ie=2,nsebin
        ecdf(ie)=ecdf(ie-1)+sspec(ixtype,ie)/tnum
      end do

!-----
! Make energy bin table
!-----
      do ie=1,nsebin
        ebint(ie)=(ie-1)*deltae(ixtype)
      end do
end if

!-----
! Source condition redefine
!-----
      xin=0.D0
      yin=0.D0
      zin=-sposi
      uin=0.D0
      vin=0.D0
      win=1.D0

!-----
! Key in half width and height at phantom surface
!-----
      write(6,220)
220  FORMAT(' Key in half width of beam at phantom surface in cm.')
```

```

      read(5,*) xhbeam
      write(6,230)
230  FORMAT(' Key in half height of beam at phantom surface in cm.')
```

```

      read(5,*) yhbeam
      radma2=xhbeam*xhbeam+yhbeam*yhbeam
      wimin=sposi/dsqrt(sposi*sposi+radma2)

      write(6,240)
240  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)
!
!-----
! Key in history number
!-----
250  write(6,260)
260  FORMAT(' Key in number of cases (0 means end of calculation.)')
```

```

      read(5,*) ncases
      if (ncases.eq.0) go to 1330

      iii=iii+1

      close(39,status='keep')
      open(39,file='egs5job.pic',access='append')
      write(39,270) iii
270  FORMAT('0',I5)

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

      do jhist=1,ncases
!-----
! Start of CALL SHOWER loop
!-----
        icses=j

!-----
! Determine direction (isotropic)
!-----
280  call randomset(w0)
      win=w0*(1.0-wimin)+wimin
      call randomset(phai0)
      phai=pi*(2.0*phai0-1.0)

```



```

sinh=dsqrt(1.D0-win*win)
uin=dcos(phai)*sinh
vin=dsin(phai)*sinh
dis=sposi/win
xpf=dis*uin
ypf=dis*vin
if (dabs(xpf).gt.xhbeam.or.dabs(ypf).gt.yhbeam) go to 280
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 290
end do

290 do j=1,jmax
  if (ybound(j+1).gt.yin) go to 300
end do

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

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

   if (isemode.eq.0) then  ! use xray.dat
     call randomset(ei0)
     do ie=2,nsebin
       if (ei0.lt.ecdft(ie)) then
         go to 310
       end if
     end do

310   if (ie.gt.nsebin) then
       ie=nsebin
     end if
     saspec(ie)=saspec(ie)+1.D0
     ekin=ebint(ie-1)+(ei0-ecdft(ie-1))*(ebint(ie)-ebint(ie-1))/
*    (ecdft(ie)-ecdft(ie-1))
     wtin = 1.0
   else
     ! use egs5job.inp
     if (isamp .eq. 0) then  ! Monoenergetic case
       ekin = ekein
       wtin = 1.0
     else if (isamp .eq. 1) then  ! Sample discrete energy from CDF
       call randomset(rnnow)
       i=0
312       continue
       i = i + 1
       if(ecdft(i) .le. rnnow) go to 312
       ekin = ebin(i)
       wtin = 1.0
     else if (isamp .eq. 2) then  ! Sample DIRECTLY from CDF
       call edistr(ekin)
       wtin = 1.0
     else if (isamp .eq. 3) then  ! Sample UNIFORMLY on energy
       ! interval and WEIGHT
       call randomset(rnnow)
       ekin = esam1 + rnnow*delsam
314       isam = 0
       continue
       isam = isam + 1
       if (ekin .lt. ebin(isam)) go to 316
       go to 314
316       continue
       wtin = epdf(isam)
     end if

```

```

end if

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

latchi=0

! -----
! Print first NWRITE or N_LINES, whichever comes first
! -----
if (ncount .le. nwrite .and. ilines .le. nlines) then
  ilines = ilines + 1
  write(6,320) etot,xin,yin,zin,uin,vin,win,iqin,irin,idin
320   FORMAT(4G15.7/3G15.7,3I5)
end if

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

! Added for energy balance tests (SJW)
if(DABS(eparte + epartd - ekin)/ekin .gt. 1.d-10) then
  write(*,330) icases, eparte, epartd
330   FORMAT('Error on # ',I6,' Escape = ',F9.5,' Deposit = ',F9.5)
endif

!-----
! 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+fexpss
fexpss2s=fexpss2s+fexpss*fexpss
fexpss=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(6,340) cputime
340   format(' Elapsed Time (sec)=',G15.5)

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

! -----
! Write out the results
! -----
write(6,350) ncount,ncases,totke,iseed1,iseed2
350   FORMAT('/', ' Ncount=',I10,' (actual cases run)',/,
*         ' Ncases=',I10,' (number of cases requested)',/,
*         ' TotKE =',G15.5,' (total KE (MeV) in run)'/
*         ' Last iseed1 =',I12,', iseed2 =',I12)

if (totke .le. 0.D0) then
  write(6,360) totke,availke,ncount

```

```

360   FORMAT(//,' Stopped in MAIN with TotKE=',G15.5,/,
*       ' AvailKE=',G15.5, /, ' Ncount=',I10)
      stop
      end if

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

      if (imode.ne.0) then
        write(1,370)
370     FORMAT(//' Comparison between sampled spectrum and original data
*'/ 23X,' Sampled Probability',25X,' Sampled Probability'
* )
        do ie=2,nsebin,2
          write(1,380) ebint(ie),saspec(ie),ecdft(ie)-ecdft(ie-1),
* ebint(ie+1), saspec(ie+1),ecdft(ie+1)-ecdft(ie)
380     FORMAT(1X,G9.3,' MeV(upper)-- ',2G12.5,3X,' ; ',G9.3,' MeV(upp
*er)-- ',2G12.5)
        end do

        if (isemode.eq.0) then
          write(1,390) sposi
390     FORMAT(/' Absorbed energy inside phantom for 100 kV X-ray'/
* ' Source position ',F10.1,' cm from phantom surface'/
* ' Within 1cm x 1 cm area after 5 cm air')
        else
          write(1,395) sposi
395     FORMAT(/' Absorbed energy inside phantom for source ',
* 'defined in egs5job.inp '/
* ' Source position ',F10.1,' cm from phantom surface'/
* ' Within 1cm x 1 cm area after 5 cm air')
        end if

        write(1,400) ncases, xhbeam, yhbeam
400     FORMAT(1X,I8,' photons normally incident from front side'/ ' Hal
*f width of beam is ',G15.5,'cm for X and ',G15.5,'cm for Y')
        end if

!-----
!   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(6,410) depths,depthl,(media(ii,med(irl)),ii=1,24),
* rhor(irl),dose(i,j,kkk),doseun(i,j,kkk)
410     FORMAT(' At ',F4.1,'--',F4.1,' cm (' ,24A1,' ,rho:',F8.4,')=',
* G13.5,'+-',G13.5,'Gy/incident')

```

```

        if (imode.ne.0) then
            write(1,410) depths,depth1,(media(ii,med(irl)),ii=1,24),
*          rhor(irl),dose(i,j,kkk),doseun(i,j,kkk)
        end if
    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
        fexp2s=fexp2s/ncases
        fexerr=dsqrt((fexp2s-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(6,430) faexpa,faexrr,fexpsa,fexerr,bsfa,bsferr
            write(1,430) faexpa,faexrr,fexpsa,fexerr,bsfa,bsferr
430          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(6,440) faexpa,faexrr,fexpsa,fexerr
            write(1,440) faexpa,faexrr,fexpsa,fexerr
440          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
!-----

        if (imode.ne.0) then
            do idose=1,idgrp          ! Loop over groups of regions to analyse
                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(1,460) Title
460          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
                            j1=j
                            ju=min(j+3,jdosu(idose))
                            write(1,470) xbound(i),xbound(i+1),i
470          FORMAT(/T15,'For x=',F10.3,' to',F10.3,5X,'i=',I3)
                            write(1,480) (ybound(jj),jj=j1,ju+1)
480          FORMAT(' ybounds:',F7.3,F12.3,3F17.3)
                            write(1,490)(jj,jj=j1,ju)
490          FORMAT(T10,'j=',t17,5(I4,13X))
                            write(1,500) zbound(kdosl(idose))
500          FORMAT(' zbounds(',F10.3,')')
                            do k=kdosl(idose),kdosu(idose)
                                write(1,510) zbound(k+1),k,(dose(i,jj,k),
*                                min(99.9, 100.*doseun(i,jj,k)/dose(i,jj,k)),
*                                jj=j1,ju)
510          FORMAT(F8.3,I4,4(1PE11.3,'-',OPF4.1,'%') )
                                if (mod(k,5).eq.0) then
                                    write(1,520)
520          FORMAT(' ')
                                end if
                            end do
                        end do
                    end do
                end if
            end do
        end if
    end do

```

```

        if(mod(ipage,nperpg).eq.0.and.(ju.ne.jdosu(idose).
*         or.i.ne.idosu(idose))) then
            write(1,460) 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=idosl(idose)-idosl(idose)
    i=i+i/5+7
    nperpg=60/i      ! Number of sets of bins per page
    write(1,460) Title
    ipage=1

    do k=kdosl(idose),kdosu(idose)
        do j=jdosl(idose),jdosu(idose),4
            jl=j
            ju=min(j+3,jdosu(idose))
530         write(1,530) zbound(k),zbound(k+1),k
            FORMAT(//T15,'for z=',F10.3,' to',F10.3,5X,'k=',I3)
540         write(1,540) (ybound(jj),jj=jl,ju+1)
            FORMAT(' Ybounds:',F7.3,F12.3,3F17.3)
550         write(1,550) (jj, jj=jl,ju)
            FORMAT(T10,'j=',T17,5(I4,13X))
560         write(1,560) xbound(idosl(idose))
            FORMAT(' Xbounds (',F10.3,')')

            do i=idosl(idose),idosu(idose)
                write(1,570) xbound(i+1),i,(dose(i,jj,k),
*                 min(99.9, 100.*doseun(i,jj,k)/dose(i,jj,k)),
*                 jj=jl,ju)
570             FORMAT(F8.3,I4,4(1PE11.3,'-',0PF4.1,'%') )
                if (mod(i,5).eq.0) then
580                 write(1,580)
                    FORMAT(' ')
                end if
            end do

            if(mod(ipage,nperpg).eq.0.and.(ju.ne.jdosu(idose).
*             or.k.ne.kdosu(idose))) then
                write(1,460) 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
end if          ! end od imode=1

!-----
! Write end of batch information
!-----
590 write(39,590)
    FORMAT('9')
    call plotxyz(99,0,0,0.D0,0.D0,0.D0,0.D0,0,0.D0)
    close(UNIT=39,status='keep')
    go to 250

1330 if (imode.ne.0) then
! =====
! call ecnsv1(nlist,nreg,totke)
! =====
end if

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

! -----
! Close files
! -----
close(UNIT=1)
close(UNIT=4)

```

```
stop
end
```

```
-----last line of main code-----
-----getvoxel.f-----
```

```
Version: 030831-1300 KEK-LSCAT
Reference: KEK Internal 2000-1
```

```
23456789|123456789|123456789|123456789|123456789|123456789|123456789|12
```

```
-----
Auxiliary subroutine for use with the EGS5 Code System
-----
```

```
This is a data-entry subprogram for use with a 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 indexes (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 < zbound(k+1)    k <= kmax
-----
```

```
SUBROUTINE ARGUMENT
```

```
nreg      Number of regions in geometry (determined by data input).
```

```
UNIT ASSIGNMENTS
```

```
Unit 1    Output summary and results
Unit 4    Input file.
Unit 6    Output file.
Unit 8    Echoes input cross-section data (assign a null file).
Unit 12   Input cross-section file from PEGS5.
```

```
INPUT FILE
```

```
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  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 5  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 pr/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 reg
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 6   ybound
-----
Record 7   zbound
-----
Record 8   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 8a  ipeangsw,      Switches for PE-angle sampling,
!           iedgesw,      K & L-edge fluorescence,
!           iraysw,       Rayleigh scattering,
!           ipolarsw,     Linearly-polarized photon scattering,
!           incohsw,      S/Z rejection,
!           iprofrsw,     Doppler broadening,
!           impacrsw     electron impact ionization (0=off, 1=on).
-----
Record 9   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 10  xlower, xupper
-----
!           Boundaries of beam in x direction, in cm
!           If xlower is zero, a value near middle
!           is taken. If XUPPER is zero, no extent
!           in X direction.
-----
Record 11  ylower,yupper   As for X direction.
-----
Record 12  thetaz,thetax,thetay
-----
!           thetaz: angle of beam to z axis (0 is normal) in degrees.
!           If thetaz is zero, others assumed normal(i.e.90 deg).
!           If thetaz is non-zero - and others both are zero.
!           thetax is as large as possible - i.e. max cos allowed,
!           and thetay is 90 deg.
!           If thetax is non-zero, it may be reduced if too large,
!           and thetay will be chosen to normalize the direction
!           cosines.
-----
Record 13  ixj,jxx         Starting random number seeding.
!           If ixj = 0, ixj is set to 123457.
!           If jxx = 0, jxx is set to 654321.
-----
Record 14  ncases         Number of cases.
-----
Record 15  ekein,iqin,isamp
!           Kinetic energy (MeV), charge of inci-
!           dent beam, and sampling switch. If
!           isamp=0, a monoenergetic beam (ekein)
!           will be used. Otherwise, a spectrum
!           input must follow (Records 15a through
!           15b), which will be sampled from discrete
!           energy (isamp=1), directly (isamp=2) or
!           uniformly over the energy range (isamp=3)
!           with weighting factor.
-----
Record 15a ebinmin       Only required when isamp>1(see above).
!           Lowest energy (MeV) in spectrum.
-----
Record 15b ebin(i),epdf(i) Only required when isamp>0(see above).

```

```

! -----
!                                     ebin(i) is 'discrete energy' with epdf(i)
!                                     for isamp=1. ebin (i) is 'top-edge' of
!                                     each energy bin (MeV) and epdf(i) is the
!                                     corresponding probability for the bin
!                                     for isamp > 1.
!                                     For example, a cross section (mb) can
!                                     be used for epdf (but do not divide it
!                                     by dE). The last card is a delimiter
!                                     and should be blank (or contain 0.0).
!                                     The i-subscript runs from 1 to nebin
!                                     (nebin calculated after the delimiter)
! -----
! Record 16  iwatch                    Switch for tracking events with swatch:
! -----
!                                     (0=No, 1=each interaction,
!                                     2=each step)
! -----
! Record 17  ibrdst,iprdst,            Switches for bremsstrahlung and pair
! -----  ibrspl,nbrspl                production ANGLE SAMPLING, and brems-
!                                     strahlung SPLITTING:
!                                     ibrdst=0 No (use default: theta=m/E)
!                                     1 Yes (recommended)
!                                     iprdst=0 No (use default: theta=m/E)
!                                     1 Yes (low-order distribution)
!                                     2 Yes (recommended)
!                                     ibrspl=0 No
!                                     1 Yes (NBRSP=splitting factor)
! -----
! Record 18  estepe,estepe2
! -----

```

```

-----
subroutine getvoxel(nreg)
implicit none
include 'include/egs5_h.f'           ! Main EGS "header" file
include 'include/egs5_bounds.f'     ! COMMONs required by EGS5 code
include 'include/egs5_brempr.f'
include 'include/egs5_edge.f'
include 'include/egs5_eicom.f'
include 'include/egs5_elec.in.f'
include 'include/egs5_media.f'
include 'include/egs5_misc.f'
include 'include/egs5_switches.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 'user_auxcommons/aux_h.f'   ! Auxiliary-code "header" file
include 'user_auxcommons/edata.f'   ! Auxiliary-code COMMONs
include 'user_auxcommons/geoxyz.f'
include 'user_auxcommons/instuf.f'
include 'user_auxcommons/voxel.f'
include 'user_auxcommons/watch.f'

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

integer nreg                         ! Arguments

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,jiinu,k,kl,ku,maxbd,maxx,maxy,
* maxx,medtmp,moreOutput,n,ner,ngroup,nn,nnn

```



```

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

write(6,1100)
write(1,1100)
1100 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
! =====

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

! -----
! Record 1: title
! -----
read(4,101) title
101  FORMAT(80A1)
write(6,102) title
write(1,102) title
102  FORMAT(8x,71('-'))/'$TITLE: '/'+' ,3X,80A1/8X,71('-')

! -----
! Record 2: nmed
! -----
read(4,*) nmed
if (nmed.eq.0 .or. nmed .gt. MXMED) then
104  write(6,104) nmed
      FORMAT(' *** Stopped in Getvoxel with nmed=',I5,' > MXMED')
      stop
end if
write(6,105) nmed
write(1,105) nmed
105  FORMAT(' Number of media : ',I5,/)

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

! -----
! Record 4: 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(6,109) maxx,maxy,maxz
write(1,109) maxx,maxy,maxz
109  FORMAT ('+',3I6);

maxbd=LIMAX
write(6,110)
write(1,110)

```

```

110  FORMAT(/T20,'Input boundaries in the x direction')
! -----
! Record 5  xbound
! -----
      if (maxx.gt.0) then                ! Just pick up boundaries.
        do i=1,maxx
          write(6,111) i
          write(1,111) i
111      FORMAT(' Small boundary for region(',I3,') ')
          read(4,*) xbound(i)
          if (i.ne.1.and.xbound(i).le.xbound(i-1)) then
            write(6,112)
            write(1,112)
112      FORMAT(' Boundary out of order*****')
          end if
          write(6,113) xbound(i)
          write(1,113) xbound(i)
113      FORMAT('+',T10,F12.3)
        end do
        write(6,114) maxx+1
        write(1,114) maxx+1
114      FORMAT(' Outer boundary for region(',I3,') ')
        read(4,*) xbound(maxx+1)
        write(6,115) xbound(maxx+1)
        write(1,115) xbound(maxx+1)
115      FORMAT('+',T10,F12.3)
      else                                ! Input groups of region.
        write(6,116)
        write(1,116)
116      FORMAT(' Initial boundary: ')
        read(4,*) xbound(1)
        write(6,117) xbound(1)
        write(1,117) xbound(1)
117      FORMAT('+',F12.3)
        ngroup=-maxx
        maxx=0
        do igrup=1,ngroup
          write(6,118)
          write(1,118)
118      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(6,119) width,nn
          write(1,119) width,nn
119      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(6,120)
            write(1,120)
120      FORMAT(T15,'*** No. of X-direction reduced ***')
          end if
          maxx=maxx+nnn
        end do
        write(6,121) (xbound(i),i=1,maxx+1)
        write(1,121) (xbound(i),i=1,maxx+1)
121      FORMAT(' Boundaries'/(6F12.3))
      end if

      imax=maxx

      maxbd=LJMAX
      write(6,130)
      write(1,130)
130      FORMAT(/T20,'Input boundaries in the y direction')
! -----
! Record 6  ybound
! -----
      if (maxy.gt.0) then                ! Just pick up boundaries.
        do i=1,maxy
          write(6,111) i
          write(1,111) i
          read(4,*) ybound(i)
          if (i.ne.1.and.ybound(i).le.ybound(i-1)) then

```

```

        write(6,112)
        end if
        write(6,113) ybound(i)
        write(1,113) ybound(i)
    end do
    write(6,114) maxy+1
    write(1,114) maxy+1
    read(4,*) ybound(maxy+1)
    write(6,115) ybound(maxy+1)
    write(1,115) ybound(maxy+1)
else
    ! Input groups of region.
    write(6,116)
    write(1,116)
    read(4,*) ybound(1)
    write(6,117) ybound(1)
    write(1,117) ybound(1)
    ngroup=-maxy
    maxy=0
    do igrup=1,ngroup
        write(6,118)
        write(1,118)
        read(4,*) width,nn
        if(nn.le.0) nn=1
        if(width.le.0.0) width=1.DO
        write(6,119) width,nn
        write(1,119) 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(6,120)
            write(1,120)
        end if
        maxy=maxy+nnn
    end do
    write(6,121) (ybound(i),i=1,maxy+1)
    write(1,121) (ybound(i),i=1,maxy+1)
end if

jmax=maxy

maxbd=LKMAX
write(6,140)
write(1,140)
140  FORMAT(/T20,'Input boundaries in the z direction')
! -----
! Record 7   zbound
! -----
if (maxz.gt.0) then
    ! Just pick up boundaries.
    do i=1,maxz
        write(6,111) i
        write(1,111) i
        read(4,*) zbound(i)
        if (i.ne.1.and.zbound(i).le.zbound(i-1)) then
            write(6,112)
            write(1,112)
        end if
        write(6,113) zbound(i)
        write(1,113) zbound(i)
    end do
    write(6,114) maxz+1
    write(1,114) maxz+1
    read(4,*) zbound(maxz+1)
    write(6,115) zbound(maxz+1)
    write(1,115) zbound(maxz+1)
else
    ! Input groups of region.
    write(6,116)
    write(1,116)
    read(4,*) zbound(1)
    write(6,117) zbound(1)
    write(1,117) zbound(1)
    ngroup=-maxz
    maxz=0
    do igrup=1,ngroup
        write(6,118)
        write(1,118)

```

```

        read(4,*) width,nn
        if(nn.le.0) nn=1
        if(width.le.0.0) width=1.DO
        write(6,119) width,nn
        write(1,119) 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(6,120)
            write(1,120)
        end if
        maxz=maxz+nnn
    end do
    write(6,121) (zbound(i),i=1,maxz+1)
    write(1,121) (zbound(i),i=1,maxz+1)
end if

kmax=maxz

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

write(6,143) imax,jmax,kmax,nreg
write(1,143) imax,jmax,kmax,nreg
143  FORMAT(' imax, jmax, kmax, nreg =',4I8)

!      Check nreg
      if(nreg .gt. MXREG) then
          write(6,150) nreg
150   FORMAT(' *** Stopped in getvoxel with nreg=',I5,' > MXREG')
          stop
      end if
      write(6,155) nreg
      write(1,155) nreg
155   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
      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) = iedgesw
          iraylr(i) = iraysw
          lpolar(i) = ipolarsw
          incohr(i) = incohrsw
          iprofr(i) = iprofrsw
          impacr(i) = impacrs
      end do

!      -----
!      Record 8 il,iu, jl,ju, kl,ku, medtmp, rhotmp, ecutin, pcutin
!      ----- (7I5,3F10.0)          Line is repeated until a blank line found

200   write(6,190)
      write(1,190)
190   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 220

!      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(6,210) il,iu,jl,ju,kl,ku,medtmp,rhotmp
        write(1,210) il,iu,jl,ju,kl,ku,medtmp,rhotmp
210   FORMAT('+',3('(',I3,I4,')'), I4, F10.3)

        if (medtmp.ne.0) then
! -----
! Record 8a: ipeangsw,iedgesw,iraysw,ipolarsw,
!           incohrsw,iprofrsw,impacrsw
! -----
        read(4,*) ipeangsw,iedgesw,iraysw,ipolarsw,incohrsw,
        *         iprofrsw,impacrsw

        write(6,215) ecutin,pcutin,ipeangsw,iedgesw,iraysw,ipolarsw,
        *         incohrsw,iprofrsw,impacrsw
        write(1,215) ecutin,pcutin,ipeangsw,iedgesw,iraysw,ipolarsw,
        *         incohrsw,iprofrsw,impacrsw
215   FORMAT(' ecut =',G15.5,' and pcut =',G15.5/
        * ' ipeangsw=',I3,',iedgesw=',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) = iedgesw
                    iraylr(irl) = iraysw
                    lpolar(irl) = ipolarsw
                    incohr(irl) = incohrsw
                    iprofr(irl) = iprofrsw
                    impacrs(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 200

220   continue
! -----
! Record 9  il,iu, jl,ju, kl,ku,izscan
! -----
        write(6,230)
        write(1,230)
230   FORMAT(' 3 pairs defining lower,upper x,y,z indeces 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
        idgrp=idgrp+1
240   write(6,242)
        write(1,242)
242   FORMAT('$: ')
        read(4,245) idosl(idgrp),idosu(idgrp),jdosl(idgrp),jdosu(idgrp),
        *         kdosl(idgrp),kdosu(idgrp),izscan(idgrp)
245   FORMAT(7I5)

        if(idosl(idgrp).eq.0 .and. idosu(idgrp).eq.0) go to 255 ! 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(6,250) idosl(idgrp),idosu(idgrp),jdosl(idgrp),jdosu(idgrp),
*      kdosl(idgrp),kdosu(idgrp),izscan(idgrp)
write(1,250) idosl(idgrp),idosu(idgrp),jdosl(idgrp),jdosu(idgrp),
*      kdosl(idgrp),kdosu(idgrp),izscan(idgrp)
250  FORMAT('+',T5,3(I6,I4),I6)

go to 240

255  continue

idgrp=idgrp-1

if(idgrp.gt.LMXDOS) then
write(6,257) idgrp,LMXDOS
257  FORMAT(' idgrp(=',I5,') must be less than LMXDOS(=',I5,')'/
*      ' Or you must chnage LMXDOS in xyzdose_h.f')
end if

! -----
! Record 10  xinl, xinu
! -----
write(6,260)
write(1,260)
260  FORMAT(/' Specifications for parallel beam, incident on',
*      ' x-y surface'/' Incident on what range of x values? ');
read(4,*) xinl,xinu
if(xinl.eq.0) xinl = (xbound(imax+1)+xbound(1))/2.
! Enter near middle
if(xinl.lt.xbound(1)) xinl =xbound(1)

if(xinu.le.xinl) xinu=xinl ! Pencil beam

if(xinu.gt.xbound(imax+1)) xinu = xbound(imax+1)

if(xinl.gt.xbound(imax+1)) xinl = xbound(imax+1)

write(6,270) xinl,xinu
write(1,270) xinl,xinu
270  FORMAT('+',T10,2F10.3)

! Search for initial region x index range

xindel=xinu-xinl
ixinl=0
280  ixinl=ixinl+1
if(xbound(ixinl).le.xinl.and.xbound(ixinl+1).gt.xinl)
* go to 290
go to 280
290  ixinu=ixinl-1
300  ixinu=ixinu+1
if(xbound(ixinu).le.xinu.and.xbound(ixinu+1).ge.xinu)
* go to 310
go to 300

310  write(6,320) ixinl,ixinu
write(1,320) ixinl,ixinu
320  FORMAT(T40,'X index ranges over I=',I3,' to ',I4);

! -----
! Record 11  yinl, yinu
! -----
write(6,330)
write(1,330)
330  FORMAT(/'$Incident on what range of y values? ');
read(4,*) yinl,yinu
if(yinl.eq.0) yinl = (ybound(jmax+1)+ybound(1))/2.
! Enter near middle
if(yinl.lt.ybound(1)) yinl =ybound(1)

if(yinu.le.yinl) yinu=yinl ! Pencil beam

if(yinu.gt.ybound(jmax+1)) yinu = ybound(jmax+1)

if(yinl.gt.ybound(jmax+1)) yinl = ybound(jmax+1)

```

```

write(6,270) yinl,yinu
write(1,270) yinl,yinu

! Search for initial region y index range

yindel=yinu-yinl
jyinl=0
340 jyinl=jyinl+1
if(ybound(jyinl).le.yinl.and.ybound(jyinl+1).gt.yinl)
* go to 350
go to 340
350 jyinl=jyinl-1
360 jyinl=jyinl+1
if(ybound(jyinl).le.yinu.and.ybound(jyinl+1).ge.yinu)
* go to 370
go to 360

370 write(6,380) jyinl,jyinl
write(1,380) jyinl,jyinl
380 FORMAT(T40,'Y index ranges over I=',I3,' to ',I4);

! -----
! Record 12 thetaz,thetax,thetay
! -----

write(6,390)
write(1,390)
390 FORMAT(' Angle of beam to axis(in deg, 0 is normal): ')

read(4,*) thetaz,thetax,thetay
win = cos(thetaz*PI/180.0)
uin = cos(thetax*PI/180.0)
uin = min(uin, dsqrt(1.0-win*win)) ! Make sure not too big.
thetax = acos(uin)*180.0/PI
vin = sqrt(1.0-win*win-uin*uin)
thetay = acos(vin)*180.0/PI
! The input value of thetaz is not used.

write(6,400) thetaz,thetax,thetay
write(1,400) thetaz,thetax,thetay
400 FORMAT('+ ',3F10.2,' deg')

! -----
! Record 13: ix, jx
! -----

read(4,*) ix,jx
if (ix .eq. 0) ix = 123457 ! Default seed
if (jx .eq. 0) jx = 654321 ! Default seed

! -----
! Save the starting random-number seeds
! -----

iseed1=ix
iseed2=jx
write(6,410) iseed1,iseed2
write(1,410) iseed1,iseed2
410 FORMAT('/', ' iseed1=',I12,5X,' iseed2=',I12,
* (starting random-number seeds)')

! -----
! Record 14: ncases
! -----

read(4,*) ncases
write(6,420) ncases
write(1,420) ncases
420 FORMAT('/', ' ncases=',I12)

! -----
! Record 15: ekein,iqin,isamp
! -----

read(4,*) ekein,iqin,isamp

if (isamp .eq. 0) then ! -----
! Monoenergetic case
! -----

write(6,430) iqin,ekein
write(1,430) iqin,ekein
430 FORMAT('/', ' MONOENERGETIC case has been selected with:',
* '/', ' iqin=',I5,' (incident charge of beam)',
* '/', ' ekein=',G15.5,' MeV (incident kinetic energy)')

```

```

else if (isamp .gt. 0) then                                     ! -----
                                                             ! Energy spectrum case
                                                             ! -----
! -----
! Record 15a: ebinmin
! -----
if(isamp.ne.1) then
  read(4,*) ebinmin                                         ! Lowest energy in spectrum (MeV)
  write(6,440) iqin,ebinmin
  write(1,440) iqin,ebinmin
440  FORMAT(/,' Energy-SPECTRUM case has been selected with:',
*      /,' iqin=',I5,' (incident charge of beam)',
*      /,' ebinmin=',F10.3,' MeV (lowest energy bin)')
end if

if (isamp .eq. 1) then
  write(6,450) isamp
  write(1,450) isamp
450  FORMAT(' isamp =',I2,' (Sample from discrete energy)')
elseif (isamp .eq. 2) then
  write(6,455) isamp
  write(1,455) isamp
455  FORMAT(' isamp =',I2,' (DIRECT-sampling over energy range)')
else if (isamp .eq. 3) then
  write(6,460) isamp
  write(1,460) isamp
460  FORMAT(' isamp =',I2,
*      ' (UNIFORM-sampling over energy range) with WEIGHTING')
end if

! -----
! Record 15b: ebin(i),epdf(i)
! -----
i = 0
465  continue                                               ! -----
                                                             ! Start of energy-spectrum input loop
                                                             ! -----
  i = i + 1
  if (i .gt. MXEBIN) then
    write(6,470) i
    write(1,470) i
470  FORMAT(/,' Stopped in getrtz with I=',I6,' > MXEBIN')
    stop
  end if
! ebin(i) is each discrete energy (isamp=1) or
! top-edge of bin (imode >1)
  read(4,*) ebin(i),epdf(i)
  if (i .gt. 1 .and. ebin(i) .le. ebin(i-1)) then
    go to 485
  else if (i .eq. 1 .and. ebin(i) .le. ebinmin) then
    go to 475
  end if
  go to 465

475  continue                                               ! Reach here when a read-error occurs
  write(6,480)
480  FORMAT(/,' Stopped in getvoxel with spectrum read-error')
  stop

485  continue                                               ! Reach here when delimiter card has been read

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

  write(6,490) (i,ebin(i),epdf(i),ecdf(i),i=1,nebin)
  write(1,490) (i,ebin(i),epdf(i),ecdf(i),i=1,nebin)
490  FORMAT(/,' Bin      Upper energy  Probability  Cumulative ',
*      /,' #          (MeV)          Probability',

```



```

*          /,(I4,3X,F10.3,2F16.4))
!
! -----
! Set up energy-sampling interval
! -----
esam1 = ebinmin
esam2 = ebin(nebin)
delsam = esam2 - esam1

write(6,500) esam1,esam2
write(1,500) esam1,esam2
500  FORMAT(/,' Energy-sampling interval is:',/,
*      '      esam1 =',G15.5,' MeV to esam2 =',G15.5,' MeV',/)
      else
510  write(6,510) isamp
      FORMAT(/,' Stopped in getvoxel with bad isamp=',I10)
      stop
      end if

! -----
! Record 16: iwatch
! -----
      read(4,*) iwatch

      write(6,520) iwatch
      write(1,520) iwatch
520  FORMAT(/,' SWATCH tracking switch: iwatch=',I2,
*      '      (0=off, 1=each interaction, 2=each step)')

! -----
! Record 17: ibrdst,iprdst,ibrspl,nbrspl
! -----
      read(4,*) ibrdst,iprdst,ibrspl,nbrspl
      write(6,530) ibrdst,iprdst,ibrspl,nbrspl
      write(1,530) ibrdst,iprdst,ibrspl,nbrspl
530  FORMAT(/,' IBRDST=',I2,/, ' IPRDST=',I2,/, ' IBRSPL=',I2, ' (NBR SPL='
*,I5,')')

      if (ibrspl .gt. 0) then
        if (nbrspl .gt. 0) then
          fbrspl = 1.0/float(nbrspl)
        else
          write(6,540) ibrspl,nbrspl
          write(1,540) ibrspl,nbrspl
540  FORMAT(/,' Stopped in Getvoxel with IBRSPL=',
*      '      I5,' and NBR SPL=',I5)
          stop
          end if
        end if

! -----
! Run KEK version of PEGS5 before calling HATCH
! (method was developed by Y. Namito - 010306)
! -----
      write(6,550)
550  FORMAT(/,' PEGS5NB3-call comes next',/)

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

! -----
! Open files (before HATCH call)
! -----
      open(UNIT=KMPI,FILE='pgs5job.peg5dat',STATUS='old')
      open(UNIT=KMPO,FILE='egs5job.dummy',STATUS='unknown')

      write(6,560)
560  FORMAT(/,' HATCH-call comes next',/)

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

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

```

```

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

ek0 = ekein

! -----
! Record 18: estepe,estepe2
! -----
      read(4,*) estepe, estepe2
      write(6,570) estepe, estepe2
      write(1,570) estepe, estepe2
570  FORMAT(/,1X,' ESTEPE at EKMAX: ',F10.0,' (estepe)',
*      /,1X,' ESTEPE at ECUT: ',F10.0,' (estepe2)')

! -----
! Print values used for efrac1 and efrac2
! -----
      write(6,*)
      write(6,*) ' EFRACL=',efrac1
      write(6,*) ' EFRACH=',efrac2

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

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

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

! -----
! Print various data associated with each media (not region)
! -----
      write(6,580)
580  FORMAT(/,' Quantities associated with each MEDIA:')
      do j=1,nmed
        write(6,590) (media(i,j),i=1,24)
590  FORMAT(/,1X,24A1)
        write(6,600) rho(j),rlc(j)
600  FORMAT(5X,' rho=',G15.7,' g/cu.cm rlc=',G15.7,' cm')
        write(6,610) ae(j),ue(j)
610  FORMAT(5X,' ae=',G15.7,' MeV ue=',G15.7,' MeV')
        write(6,620) ap(j),up(j)
620  FORMAT(5X,' ap=',G15.7,' MeV up=',G15.7,' MeV',/)
      end do

! -----
! Print media and cutoff energies assigned to each region
! -----
      if(moreOutput .eq.1) then

```

```

do i=2,nreg
  if (med(i) .eq. 0) then
    write(6,630) i,ecut(i),pcut(i)
    FORMAT(' medium(' ,I3,')=vacuum',18X,
630 *      'ecut=',G10.5,' MeV, pcut=',g10.5,' mev')
  else
    write(6,640) i,(media(ii,med(i)),ii=1,24),ecut(i),pcut(i)
    FORMAT(' medium(' ,I3,')=',24A1,
640 *      'ecut=',G10.5,' MeV, pcut=',G10.5,' MeV')
  !
  ! -----
  ! Print out energy information of K- and L-X-rays
  ! -----
  if (iedgfl(i) .ne. 0) then          ! Output X-ray energy
    ner = nne(med(i))
    do iiz=1,ner
      izn = zelem(med(i),iiz) ! Atomic number of this element
      write(6,650) izn
650      FORMAT(' X-ray information for Z=',I3)
      write(6,660) (ekx(ii,izn),ii=1,10)
660      FORMAT(' K-X-ray energy in keV',/,
*          4G15.5,/,4G15.5,/,2G15.5)
      write(6,670) (elx1(ii,izn),ii=1,8)
670      FORMAT(' L-1 X-ray in keV',/,4G15.5,/,4G15.5)
      write(6,680) (elx2(ii,izn),ii=1,5)
680      FORMAT(' L-2 X-ray in keV',/,5G15.5)
      write(6,690) (elx3(ii,izn),ii=1,7)
690      FORMAT(' L-3 X-ray in keV',/,4G15.5,/,3G15.5)
    end do
  end if
end if
end do
end if
return          ! -----
              ! Return to MAIN
              ! -----
end

```

!-----last line of getvoxel.f-----

```

!-----ausgab.f-----
! Version: 030831-1300
! Reference: SLAC-265 (p.19-20, Appendix 2)
!-----
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12
!
! Required subroutine for use with the EGS5 Code System
!-----
! A simple AUSGAB to:
!
! 1) Score energy deposition
! 2) Print out stack information
! 3) Print out particle transport information (if switch is turned on)
!
!-----

```

```

subroutine ausgab(iarg)

implicit none

include 'include/egs5_h.f'          ! Main EGS "header" file
include 'include/egs5_epcont.f'    ! COMMONs required by EGS5 code
include 'include/egs5_stack.f'

include 'user_auxcommons/aux_h.f'  ! Auxiliary-code "header" file
include 'user_auxcommons/etaly1.f' ! Auxiliary-code COMMONs
include 'user_auxcommons/geoxyz.f'
include 'user_auxcommons/lines.f'
include 'user_auxcommons/ntaly1.f'
include 'user_auxcommons/voxel.f'
include 'user_auxcommons/watch.f'

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

common/score/                      ! Variables to score
                                  ! Variables to score

```

```

*      depe(LIMAX,LJMAX,LKMAX),faexp,fexps,imode
real*8 depe,faexp,fexps
integer imode

integer                                ! Arguments
* iarg

real*8                                  ! Local variables
* cmod,dcon,edepwt,encoa,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

! added SJW for particle by particle energy balance
if(irl.eq.1) then
  eparte = eparte + edepwt
else
  epartd = epartd + edepwt
end if

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-iold).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
      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 (imode.eq.0) then
  call plotxyz(iarg,np,iq(np),x(np),y(np),z(np),e(np),ir(np),
*   w(np))
end if

return

end

```

```

!-----last line of ausgab.f-----
!-----howfar.f-----
! Version: 030831-1300
! Reference: SLAC-265 (p.19-20, Appendix 2)
!-----
!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/geoxyz
! 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 'include/egs5_switches.f'

```

```

include 'user_auxcommons/aux_h.f' ! Auxiliary-code "header" file
! Auxiliary-code COMMONs
include 'user_auxcommons/geoxyz.f'
include 'user_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. uestep) then
    uestep=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. uestep) then
    uestep = 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. uestep) then
    uestep=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. uestep) then
    uestep = 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
! Reference: SLAC-265 (p.19-20, Appendix 2)
!-----
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12
!
! real 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)
!-----
real function encoea(energy)

real 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 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=log(enmu(i+1))
    enm0=log(enmu(i))
    hnu1=log(hnu(i+1))
    hnu0=log(hnu(i))

    ene0=log(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-----
!-----encoew.f-----
! Version: 030831-1300
! Reference: SLAC-265 (p.19-20, Appendix 2)
!-----
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12
!
! real function encoew(energy)
! Function to evaluate the energy absorption coefficient of water.
! (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)
!-----
real function encoew(energy)

real hnu(36)/0.001,0.0015,0.002,0.003,0.004,0.005,0.006,0.008,
* 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 enmu(36)/4065., 1372., 615.2, 191.7, 81.91, 41.88,
* 24.05, 9.915, 4.944, 1.374, 0.5503, 0.1557,
* 0.06947,0.04223,0.03190,0.02597,0.02546,0.02764,
* 0.02967,0.03192,0.03279,0.03299,0.03284,0.03206,
* 0.03103,0.02965,0.02833,0.02608,0.02281,0.02066,
* 0.01915,0.01806,0.01658,0.01566,0.01441,0.01382/

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

integer i

if (energy.gt.hnu(36)) then
  encoew=enmu(36)
  return
end if
if (energy.lt.hnu(1)) then
  encoew=enmu(1)
  return
end if

do i=1,36
  if(energy.ge.hnu(i).and.energy.lt.hnu(i+1)) then
    enm1=log(enmu(i+1))
    enm0=log(enmu(i))
    hnu1=log(hnu(i+1))
    hnu0=log(hnu(i))

    ene0=log(energy)

```



```

        slope=(enm1-enm0)/(hnu1-hnu0)
        encoew=exp(enm0+slope*(ene0-hnu0))
        return
    end if
    if(energy.eq.hnu(i+1)) then
        encoew=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 ENCOEW*****',/, ' E=',G15.5,///)
    return
    end
!-----last line of encoew.f-----

```