

EGS5 サンプルプログラム (ucnaicgv.f)
NaI 検出器の応答計算 (cg Version)
(August 12, 2013 Draft)

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

目次

| | |
|---|-----------|
| 1. Combinatorial Geometry (CG) | 1 |
| 1.1. Body の定義 | 1 |
| 1.2. リージョンの定義 | 1 |
| 1.3. リージョン定義の例 | 2 |
| 2. サンプルプログラム ucnaicgv.f の概要 | 4 |
| 2.1. CG 入力データ | 4 |
| 3. ユーザーコードの内容 | 5 |
| 3.1. メインプログラム: Step 1 | 5 |
| 3.2. メインプログラム: Step 2 | 7 |
| 3.3. メインプログラム: Step 3 | 7 |
| 3.4. メインプログラム: Step 4 | 9 |
| 3.5. メインプログラム: Step 5 | 9 |
| 3.6. メインプログラム: Step 6 | 10 |
| 3.7. メインプログラム: Step 7 | 10 |
| 3.8. メインプログラム: Step 8 | 10 |
| 3.8.1. 統計誤差: | 12 |
| 3.9. メインプログラム: Step 9 | 13 |
| 3.10. Subroutine ausgab | 14 |
| 3.11. Subroutine howfar | 15 |
| 4. ucnai.f と ucnaicgv.f の計算速度の比較 | 15 |
| 5. 実習課題 | 16 |
| 5.1. 実習課題 1 : NaI 検出器の計算 | 16 |
| 5.2. 実習課題 2 : Ge 検出器の計算 | 16 |
| 5.3. 実習課題 3 : 空気電離箱の計算 | 16 |
| 6. 実習課題の解答例 | 17 |
| 6.1. 実習課題 1 | 17 |
| 6.2. 実習課題 2 | 21 |
| 6.3. 実習課題 3 | 22 |

1. Combinatorial Geometry (CG)

1.1. Body の定義

EGS 用 CG [1] では、以下のような立体 (Body) を使用する事ができる。

1. 直方体 (RPP)

x -, y - と z -方向の最小値及び最大値で定義する。各面はいずれかの軸と平行である。

2. 球 (SPH)

球の中心を示すベクトル \mathbf{V} と半径で定義する。

3. 円筒 (RCC)

円筒の底面の中心を示すベクトル \mathbf{V} と、中心からの高さベクトル \mathbf{H} 及び円筒の半径で定義する。

4. 円錐台 (TRC)

円錐の底面の中心を示すベクトル \mathbf{V} 、底面中心からの上面中心への高さベクトル \mathbf{H} 、及び底面と上面のそれぞれの半径 $R1$ 及び $R2$ で定義する。

5. トーラス (TOR)

いずれかの軸に平行なトーラスの中心を示すベクトル \mathbf{V} 、トーラス中心から、チューブの中心までの距離 $R1$ 、チューブの半径 $R2$ 及びトーラスの方向を示す番号、(n : $x/y/z = 1/2/3$) で定義する。更に、トーラスの始まりの角度 $\theta1$ と終わりの角度 $\theta2$ を指定する。トーラス全体を使用する場合には、 $\theta1=0$, 及び $\theta2=2\pi$ とする。

表 1: 各形状の立体とその記述のためのデータ

| 形状 | 通番 | 各形状の立体を定義するデータ | | | | | |
|-----|----|----------------|-----------|------|------|------|------|
| RPP | # | Xmin | Xmax | Ymin | Ymax | Zmin | Zmax |
| SPH | # | Vx | Vy | Vz | R | | |
| RCC | # | Vx | Vy | Vz | Hx | Hy | Hz |
| | | R | | | | | |
| TRC | # | Vx | Vy | Vz | Hx | Hy | Hz |
| | | R1 | R2 | | | | |
| TOR | # | Vx | Vy | Vz | R1 | R2 | |
| | | $\theta1$ | $\theta2$ | n | | | |

1.2. リージョンの定義

各リージョンは、body の組み合わせにより定義する。組み合わせには、特別な記号、+、- 及び OR が使われる。

+ 記号の後に body 番号が書かれた場合には、body の内側の領域がリージョンとなる。一方、- 記号の後に body 番号が書かれた場合には、body の外側の領域がリージョンとなる。body 番号の後に+又は- 記号と body 番号が続く場合には、間に AND 記号があるのと同じである。従って、+1 +2 は、body 1 の内側でなおかつ body 2 の内側を意味するので、body 1 と body 2 の重なった領域となる。一方、+1 -2 は、body 1 の内側でなおかつ body 2 の外側を意味するので、body 1 の領域

中で body 2 と重なっていない領域を意味することになる。Body 番号が OR 記号の後に書かれた場合は、OR 記号は結合記号として使用される。リージョンが、OR 記号で結合したサブリージョンの組み合わせで定義される場合もある。2つ以上の OR 記号が使われる場合、OR の機能は、OR 記号の間及び OR 記号からリージョン定義の行の最後までに含まれる全ての body 番号に、+ や - 記号に関係なく適用される。

1.3. リージョン定義の例

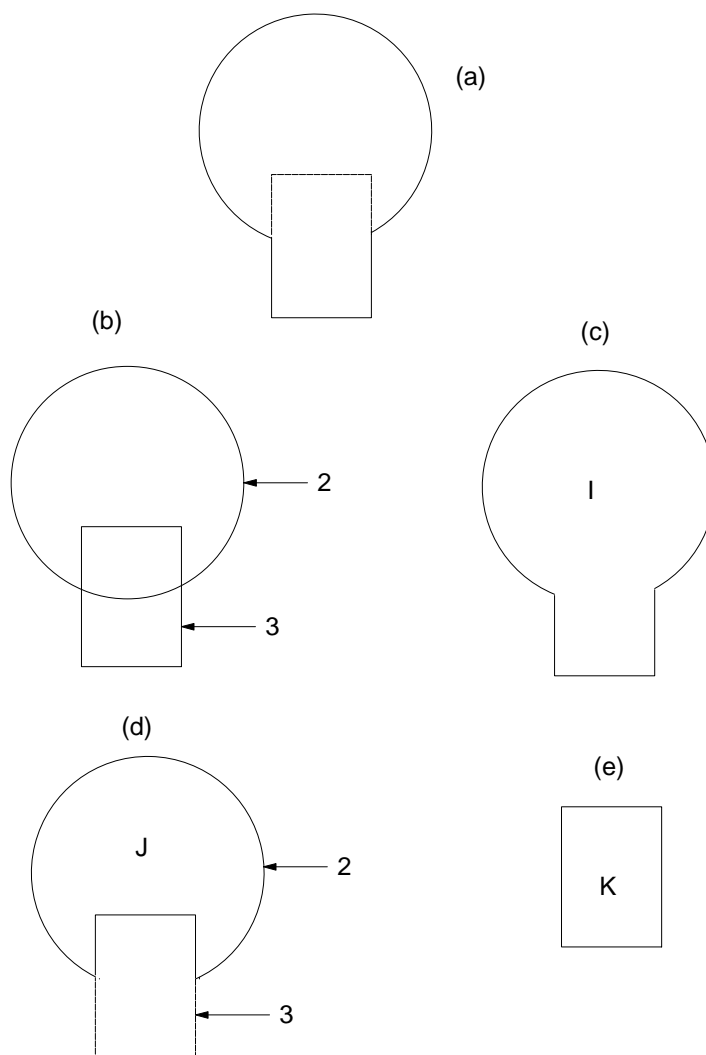


図 1: Combinatorial Geometry の例

第 1 図に示すような、球 (body 2) に円筒 (body 3) が挿入している様な体系を考える。もし、球と円筒の物質が同じであれば、リージョン I (図 1c) の様に一つのリージョンとする事ができる。リージョン I は、

$$I = +2OR + 3$$

と記述する。これは、リージョン I が、body 2 か body 3 のどちらかに属する領域であることを意味している。

球と円筒が異なった物質の場合、円筒部を除外した球には、円筒部のリージョン番号 (K) と異なったリージョン番号を付ける (例えば J)。

リージョン J (図 1d) は、

$$J = +2 - 3$$

と記述する。これは、body 2 に属するが、body 3 に属さない領域を意味する。

リージョン K (図 2e) は、単に

$$K = +3$$

と記述する。これは、body 3 の属する領域を意味する。

2つ以上の body を組み合わせる場合には、+、- や OR 記号を含む長い記述となる。しかしながら、形状中の全ての点は、どれか一つのリージョンとして定義される様にしなければならない。

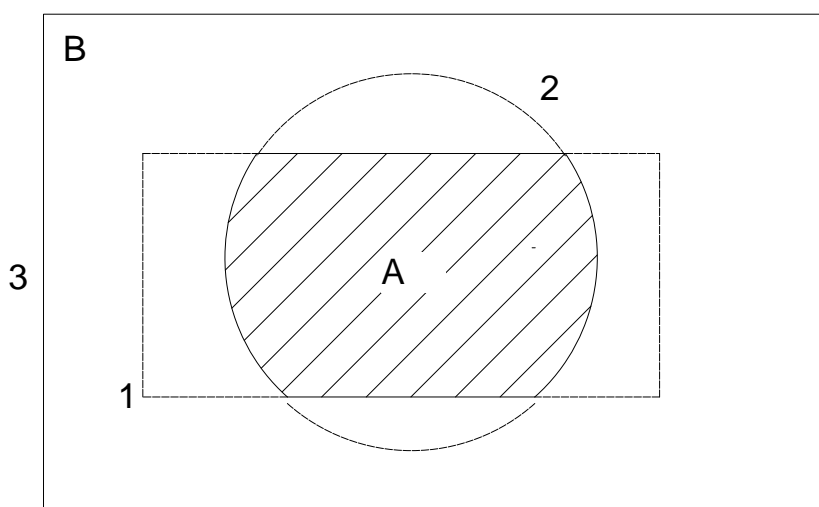


図 2: Use of OR operator.

OR 記号を使ったもっと複雑な例として、第 2 図の斜線部の領域 A と斜線を引いていない領域 B を考える。これらのリージョンは、2つの直方体 (body 1 と 3) と、一つの円筒 (body 2) で記述される。それぞれのリージョンは、

$$A = +1 + 2$$

そして

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

と記述する。OR 記号は、次に OR 記号が現れるまで、それに続く全ての body 番号に適用される事に注意する必要がある。

2. サンプルプログラム ucnaicgv.fの概要

ucnaicgv.fは、CG を使って形状を記述するユーザコードである。CG 入力データは、ユニット4のデータファイルに記述する。

2.1. CG 入力データ

図3に示すように円筒の組み合わせで体系を定義している。

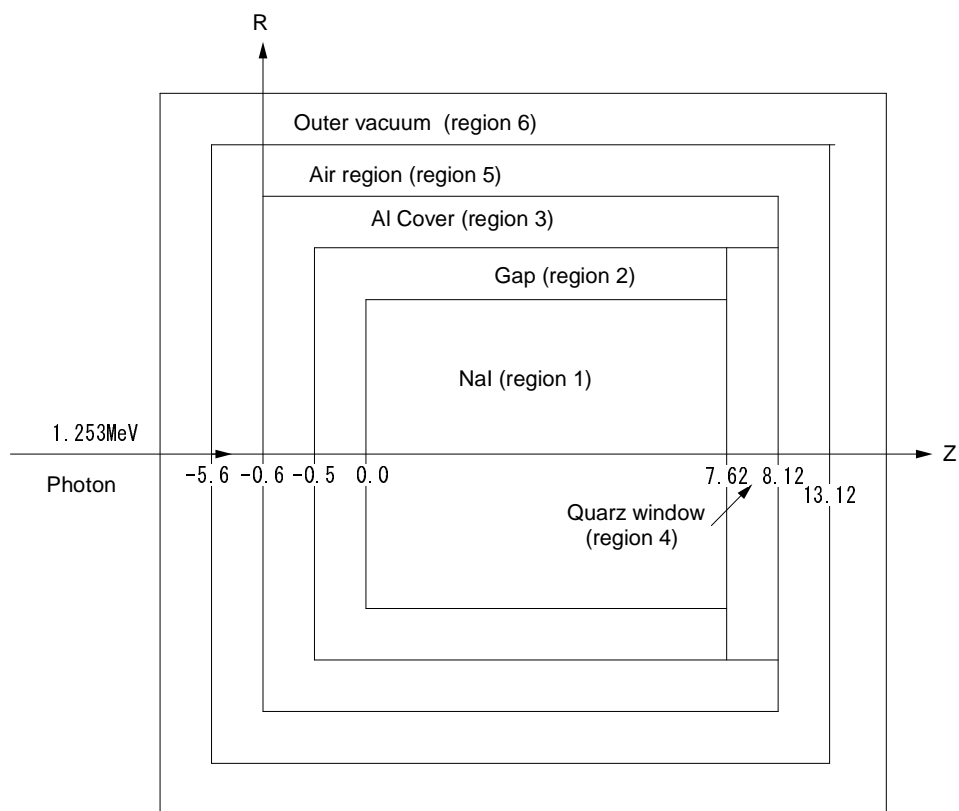


図 3: ucnaicgv.f のジオメトリー

この形状の入力データは、以下のように記述する。

| | | | | | | |
|-----|---|-------|------|-------|------|-----|
| RCC | 1 | 0.00 | 0.0 | 0.0 | 0.00 | 0.0 |
| | | 7.62 | 3.81 | | | |
| RCC | 2 | 0.00 | 0.0 | -0.5 | 0.00 | 0.0 |
| | | 8.12 | 4.31 | | | |
| RCC | 3 | 0.00 | 0.0 | -0.6 | 0.00 | 0.0 |
| | | 8.72 | 4.41 | | | |
| RCC | 4 | 0.00 | 0.0 | 7.62 | 0.00 | 0.0 |
| | | 0.5 | 4.31 | | | |
| RCC | 5 | 0.00 | 0.0 | -5.6 | 0.00 | 0.0 |
| | | 18.72 | 9.41 | | | |
| RCC | 6 | 0.0 | 0.0 | -10.0 | 0.0 | 0.0 |
| | | 30.0 | 12.0 | | | |

```

END
Z1      +1
Z2      +2      -1
Z3      +3      -2      -4
Z4      +4
Z5      +5      -3
Z6      +6      -5
END
1      0      2      3      4      0

```

最後の行は、対応するリージョンの物質番号である。

1. 線源条件

- 入射粒子は、エネルギー 1.253MeV の光子
- 入射粒子の角度は、Z-軸に沿って (0.0, 0.0, -5.0) に垂直入射

2. 得られる情報

- CGview 用の飛跡データ (egs5job.pic)
- 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情報
 - ピーク及び全検出効率
 - レスポンス
 - NaI 検出器領域の入ってくる、光子、電子、陽電子のスペクトル

3. ユーザーコードの内容

3.1. メインプログラム: Step 1

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 に直接関係する `include` 関係のファイルは、`include/ディレクトリ` (egs に関するもの)、`pegscommons/` (pegs に関するもの) および `auxcommons/` (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。*

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。従って、egs5 に直接関係する配列の大きさを変更場合は、`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。

```

implicit none
!
! -----
! EGS5 COMMONs
! -----
include 'include/egs5_h.f'           ! Main EGS "header" file

```

*これらの設定は、egs5run スクリプト又は egs5run.bat で設定される。

```

include 'include/egs5_bounds.f'
include 'include/egs5_edge.f'
include 'include/egs5_media.f'
include 'include/egs5_misc.f'
include 'include/egs5_switches.f'
include 'include/egs5_uphiot.f'
include 'include/egs5_useful.f'
include 'include/randomm.f'

```

`include 'include/egs5_h.f'` 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、ジオメトリー関係のサブルーティン等ユーザーコードに関連する `include` 文である。

```

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

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

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

```

最後の `include` 文が、CG に関連したもので、CG を使用する場合には常にこの表現とする。ユーザーコード内で使用する `common` を次に定義する。

```

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

実行文の先頭で、使用するユニットを `open` する。egs5 では、pegs5 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。

```

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

```

ユニット 39 は、飛跡情報の出力ファイルである。

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。

[†]EGS4 の COMIN マクロに対応する扱いである。

3.2. メインプログラム: Step 2

ユーザーコードで使用する物質数を `nmed` で定義する。

物質データ及び各物質の Characteristic Dimension を設定した後で、`pegs5` を call する。`medarr` のデータは必ず 24 文字分を指定する必要がある。物質名が 24 文字未満の場合には空白を補って、合計 24 文字とする。Characteristic Dimension は、当該物質で構成されるリージョンの最も小さいサイズ (1 cm × 1 cm × 1 cm の立方体であれば 1cm) に設定する。

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

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

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

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

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

chard(1) = 3.81d0             ! automatic step-size control
chard(2) = 0.1d0
chard(3) = 0.5d0
chard(4) = 5.0d0

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

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

! =====
! call peps5
! =====
```

3.3. メインプログラム: Step 3

飛跡データファイルのフォーマットを指定する `nprec` を設定する。このユーザーコードでは、フリーフォーマットの 3 を指定する。計算結果の出力ファイルに、CG データの開始を示す `CG data` を書き込み、その後 CG の入力データを読み込み、`cg` データを指定したファイル (この場合は、6) に出力する処理を行うサブルーティン `geomgt` を call する。その後、CG データの終了を意味する `End of CG data` を出力する。次に、飛跡データファイルに必要な情報を出力する。出力ユニットである

ifto は、39 に設定している。PICT のデータモードを示す文字列 (CSTA-FREE 又は、CSTA) を出力し、再度 subroutine geomgt により CG データを飛跡データファイルに出力する。最後に CG データの終了を意味する CEND を出力する。これらの処理後、cg データから、リージョン総数である nreg を引き出す。

CG を使用する場合には、この部分が必ず必要であり、変更する必要はない。

```

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

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

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

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

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

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

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

物質番号の設定を、CG データの最後で定義した情報から読み込んだ後、各リージョンの物質番号、
egs カットオフエネルギー、オプションの設定（このユーザーコードでは、光電子の角度分布、特性
X 線の発生）を行う。
anlux 乱数のシード inseed の値を設定し、初期化する。

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

!   Set option except vacuum region

do i=2,nreg-2
  if(med(i).ne.0) then
    iphter(i) = 0      ! Switches for PE-angle sampling
    iedgfl(i) = 1      ! K & L-edge fluorescence
    iauger(i) = 0      ! K & L-Auger
    iraylr(i) = 0      ! Rayleigh scattering
    lpolar(i) = 0      ! Linearly-polarized photon scattering
    incohr(i) = 0      ! S/Z rejection
    iprofr(i) = 0      ! Doppler broadening
    impacr(i) = 0      ! Electron impact ionization
  end if
end do

!-----
!   Random number seeds. Must be defined before call hatch
!   or defaults will be used. inseed (1- 2^31)
!-----
luxlev = 1

```

```

inseed=1
write(6,120) inseed
120 FORMAT(/,' inseed=',I12,5X,
*          '(seed for generating unique sequences of Ranlux)')

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

```

3.4. メインプログラム: Step 4

入射粒子のパラメータを設定する。この例では、単一エネルギーの光子 (1.253MeV) が、垂直に検出器の中心に入射するとしている。なお CG を用いる場合には、`irin=0` と指定することにより `irin` を自動的に設定することができる。このユーザーコードでは線源位置は固定なので、`irin=0` の場合は、直後に `irin` を CG により求めるルーチンが書かれている。

```

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

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

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

```

3.5. メインプログラム: Step 5

`pegs` で作成した物質データの最高エネルギー (全エネルギー) の最小値 `emaxe` を `hatch` で求めるために、`emaxe` を 0 に設定後に、`subroutine hatch` を呼ぶ。`emaxe` は、入射粒子のエネルギーが、物質データの最高エネルギーを超えていないことをチェックするために用いる。`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ようにしている。

```

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

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

!      -----
!      Open files (before HATCH call)
!      -----

```

```

open(UNIT=KMPI,FILE='pgs5job.pegs5dat',STATUS='old')
open(UNIT=KMPO,FILE='egs5job.dummy',STATUS='unknown')

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

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

```

3.6. メインプログラム: Step 6

普通のユーザーコードでは、このステップで形状に関する情報（平板、円筒、球等）を記述するが、本ユーザーコードでは `cg` 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。

3.7. メインプログラム: Step 7

`ausgab` に必要な設定を行う。エネルギービンの幅を、入射エネルギーとビン数 (50) から計算する。`Ncases` は、ヒストリー数で、`maxpict` は、飛跡情報を記録するヒストリー数である。

```

!      Energy bin width
!      deltae=ekein / 50

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

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

```

3.8. メインプログラム: Step 8

飛跡表示ファイルにバッチ番号を出力した後で、`ncases` ヒストリーだけ `subroutine shower` を `call` する。入射粒子の運動エネルギーに対応する電子の全エネルギーが物質データで設定した最高エネルギー `emaxe` を超えていないか調べる。各ヒストリー毎に、`NaI` 領域での吸収エネルギーの有無を調べ、吸収エネルギーがある場合には、全検出効率の数に `wtin` を加え、そのエネルギーが入射粒子の 99.9%以上の時は、ピーク検出効率の数に `wtin` を加える。また、吸収エネルギーの値により、波高分布の対応するチャンネルに `wtin` を加える。統計誤差評価のために、粒子スペクトルを含め上記の変数のヒストリー毎の結果の自乗値の和を求めておく。

```

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

```

```

!
!
! if(iwatch.gt.0) call swatch(-99,iwatch)
!
!
! -----
do i=1,ncases ! -----
! Start of shower call-loop
! -----
!
! -----
! Select incident energy
! -----
!
wtin = 1.0

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

totke = totke + availke ! Keep running sum of KE

!
! -----
! Select incident angle
! -----
!
!
! -----
! Print first NWRITE or NLINES, whichever comes first
! -----
!
if (ncount .le. nwrite .and. ilines .le. nlines) then
  ilines = ilines + 1
280 write(6,280) etot,xin,yin,zin,uin,vin,win,iqin,irin,idin
  FORMAT(7G15.7,3I5)
end if
!
! -----
! Compare maximum energy of material data and incident energy
! -----
!
if(etot+(1-iabs(iqin))*RM.gt.emaxe) then
  write(6,fmt="(' Stopped in MAIN.',
1 ' (Incident kinetic energy + RM) > min(UE,UP+RM).')")
  stop
end if
!
! -----
! Verify the normalization of source direction cosines
! -----
!
if(abs(uin*uin+vin*vin+win*win-1.0).gt.1.e-6) then
  write(6,fmt="(' Following source direction cosines are not',
1 ' normalized.',3e12.5)")uin,vin,win
  stop
end if
!
! -----
! call shower (iqin,etot,xin,yin,zin,uin,vin,win,irin,wtin)
! -----
!
! If some energy is deposited inside detector add pulse-height
! and efficiency.
!
if (depe .gt. 0.D0) then
  ie=depe/deltae + 1
  if (ie .gt. 50) ie = 50
  phs(ie)=phs(ie)+wtin

```

```

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

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

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

!                                     ! -----
!                                     ! End of CALL SHOWER loop
!                                     ! -----

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

    call plotxyz(99,0,0,0.D0,0.D0,0.D0,0.D0,0,0,0.D0,0.D0)
    write(39,fmt="( '9' )")          ! Set end of batch for CG View

```

3.8.1. 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。ucnaicgv.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。:

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

- \bar{x} の分散値は、

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

となる。

- 統計誤差として、

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

を用いる。

先の計算した結果とその自乗の和は、上記の処理に用いる。

3.9. メインプログラム: Step 9

得られた結果を処理して打ち出す処理を行う。ピーク検出効率、全検出効率及び波高分布について、ヒストリー毎の結果の和と結果の自乗和から、平均値とその統計誤差を求めて出力する。

```

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

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

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

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

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

```

その後、同様にして NaI 検出器に入射した粒子のスペクトルについて、平均と統計誤差を求めて出力する。

3.10. 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。

iarg < 5 の場合には、それぞれのリージョンでの吸収エネルギーを計算する。

物質番号が 1(NaI) の時は、ステップ中での吸収エネルギーを、検出器の吸収エネルギーに加える。検出器中で更に、粒子が外部から検出器部に入ってきた場合かどうかの判定を行い、外部から入ってきた粒子の場合は、粒子に応じて、エネルギー毎の集計用配列変数に wt を加算する。ヒストリー数が maxpict 以下の時は、飛積情報を出力するために plotxyz を call する。

```
! -----
! Set some local variables
! -----
!
! irl = ir(np)
! iql = iq(np)
! edepwt = edep*wt(np)
!
! -----
! Keep track of energy deposition (for conservation purposes)
! -----
!
! if (iarg .lt. 5) then
!     esum(iql+2,irl,iarg+1) = esum(iql+2,irl,iarg+1) + edepwt
!     nsum(iql+2,irl,iarg+1) = nsum(iql+2,irl,iarg+1) + 1
! end if
!
! -----
! Print out particle transport information (if switch is turned on)
! -----
!
! if (iwatch .gt. 0) call swatch(iarg,iwatch)
!
! if(iarg. .ge. 5) return
!
! -----
! Score energy deposition inside NaI detector
! -----
!
! if (med(irl) .eq. 1) then
!     depe = depe + edepwt
!
! -----
! Score particle information if it enters from outside
! -----
!
! if (irl .ne. irold .and. iarg .eq. 0) then
!     if (iql .eq. 0) then
!         ! photon
!         ntype=1
!         ie = e(np)/deltae + 1
!         if(ie .gt. 50) ie = 50
!     elseif (iql .eq. -1) then
!         ! electron
!         ntype=2
!         ie = (e(np) - RM)/deltae + 1
!         if(ie .gt. 50) ie = 50
!     else
!         ! positron
!         ntype=3
!         ie = (e(np) - RM)/deltae + 1
!         if(ie .gt. 50) ie = 50
!     end if
!     spec(ntype,ie) = spec(ntype,ie) + wt(np)
! end if
! end if
!
! -----
```



```

!      Print out stack information (for limited number cases and lines)
!      -----
      if (ncount .le. nwrite .and. ilines .le. nlines) then
          ilines = ilines + 1
          write(6,100) e(np),x(np),y(np),z(np),u(np),v(np),w(np),
*              iql,irl,iarg
100    FORMAT(7G15.7,3I5)
      end if

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

      return

      end

```

3.11. Subroutine howfar

CG を利用するかぎりユーザーが **howfar** を変更する必要は一切ない。

以下、参考のため **howfar** の機能を述べる。**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、ユーザー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。

4. ucnaif と ucnaicgv.f の計算速度の比較

複雑な形状の計算を行う場合には、**cg** は相対的に容易であるが、反面、円筒平板形状の **howfar** に比べ、計算時間が長いという問題がある。対象とする問題によって、違いは異なるが、円筒平板形状を使用している **ucnaif** と **ucnaicgv.f** で全く同じ条件の計算を行うと、**ucnaif** の方が 1.6 倍速いという結果が得られている。[2]

5. 実習課題

5.1. 実習課題 1 : NaI 検出器の計算

次のように変更して、ピーク検出効率及び全検出効率の変化を調べよ。

1. 線源を、Cs-137 の単一エネルギー光子 (0.662MeV) に変える。
2. 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. 1.253MeV 線源について、一方向 (Z-方向) のみに放出している線源光子を、等方線源に変更する。
4. 1.253MeV 一方向線源で、検出器の有感領域の厚さを 2 倍する。

5.2. 実習課題 2 : Ge 検出器の計算

検出器を、Ge に変更して、同じ大きさの NaI と、1.253MeV 線源に対するピーク及び全検出効率と比較せよ。

5.3. 実習課題 3 : 空気電離箱の計算

検出器を、 20° 、1 気圧の空気に変え、1.253MeV 線源に対して、吸収エネルギーを求めよ。検出器の途中のギャップを除き、3 インチ直径で 3 インチ長さの空気の領域の周辺に厚さ、5mm の Al がある形状とする。Al カバーの周辺に厚さ 5cm の空気層があるとする。

空気の W 値 (33.97eV/pair) を用いて、入射光子 1 個当たりのこの電離箱の出力 (Coulomb/source) を求めよ。電荷素量を、 1.602×10^{-19} C/e とする。

6. 実習課題の解答例

比較のために、ucnaicgv.f の計算モードで、ヒストリー数 10000 の場合の結果 (egs5job.out) を別な名称のファイル名 (例えば、nai.out) で保存しておく。

6.1. 実習課題 1

1. Cs-137 線源

- cp ucnaicgv.f ucnaicgv1.f
これは、UNIX 又は Cygwin の場合である。DOS の場合は、
copy ucnaicgv.f ucnaicgv1.f
又は、Windows 上でファイルのコピーを行う。以下の操作でも同様。
- cp ucnaicgv.data ucnaicgv1.data
- cp ucnaicgv.inp ucnaicgv1.inp
- ucnaicgv1.f を以下のように変更する。
 - ucnaicgv1.f 中の
ekein=1.253 ! Incident particle kinetic energy
を
ekein=0.662 ! Incident particle kinetic energy
に変更する。
- ucnaicgv1.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、ucnaicgv1 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 ucnaicgv1
 - ucnaicgv1 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。
- 計算が終了したら、egs5job.out を調べ、1.253MeV の結果と比較する。計算結果をもっともよく表す指標として、ピーク効率と全効率を下の表に示す。

| 変更前後の効率 | | |
|-----------|----------------------------|--------------------------|
| | ピーク効率 (P_{eff}) | 全効率 (T_{eff}) |
| ucnaicgv | $37.7 \pm 0.5 \%$ | $76.5 \pm 0.4 \%$ |
| ucnaicgv1 | $59.2 \pm 0.5 \%$ | $87.1 \pm 0.3 \%$ |

なお、計算機、コンパイラーの違いにより、計算の状況がかわり、数値は統計誤差程度の変動を示すことがありうる。

2. Co-60 source

- cp ucnaicgv.f ucnaicgv2.f
- cp ucnaicgv.data ucnaicgv2.data

- `cp ucnaicgv.inp ucnaicgv2.inp`
- `ucnaicgv2.f` を以下のように変更する。
 - 最高エネルギーの変更


```
ekein=1.253      ! Incident particle kinetic energy
```

 を


```
ekein=1.333      ! Incident particle kinetic energy
```

 に変更する。
 - 線源のエネルギー決定部分の変更


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

 の直後に


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

 を追加する。
 - 入射エネルギー出力部の変更


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

 を


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

 に変更する。
- `ucnaicgv2.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、`ucnaicgv2` 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 ucnaicgv2
```
- 計算が終了したら、`egs5job.out` を調べ、1.253MeV の結果と比較する。波高分布に2つの入射エネルギーに対応したピークがあることを確認する。

3. 等方線源

- `cp ucnaicgv.f ucnaicgv3.f`
- `cp ucnaicgv.data ucnaicgv3.data`
- `cp ucnaicgv.inp ucnaicgv3.inp`
- `ucnaicgv3.f` を以下の様に変更する。
 - 2π での等方角度分布をサンプリングする文を追加する。


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

```

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

```

に変更。

- ucnaicgv3.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、ucnaicgv3 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 ucnaicgv3
- 計算が終了したら、CGview の”体系・飛跡データの読み込み”で、egs5job.pic を選択し、等方線源となっていることを確認する。
- egs5job.out を調べ、ビーム入射の場合と結果を比較する。
結果の例: $P_{\text{eff}}=3.6 \pm 0.2 \%$, $T_{\text{eff}}=9.3 \pm 0.3 \%$

4. 長さ 2 倍の NaI

- cp ucnaicgv.f ucnaicgv4.f
- cp ucnaicgv.data ucnaicgv4.data
- cp ucnaicgv.inp ucnaicgv4.inp
- ucnaicgv4.f を以下のように変更する。
 - 検出器の長さの変更


```

tdet=7.62

```

 を


```

tdet=7.62*2.0

```

 に変更する。
 - この修正は、計算とは直接関係しないことである。CG を使用する場合、CG 入力データから検出器のサイズ等を直感できないので、どのような形状の計算か判るようになるためのものである。実際の形状の変更は、ucnaicgv4.data で行う。
- ucnaicgv4.data を以下の様に変更する。

| | | | | | | |
|-----|---|-------|------|------|------|-----|
| RCC | 1 | 0.00 | 0.0 | 0.0 | 0.00 | 0.0 |
| | | 15.24 | 3.81 | | | |
| RCC | 2 | 0.00 | 0.0 | -0.5 | 0.00 | 0.0 |
| | | 15.74 | 4.31 | | | |

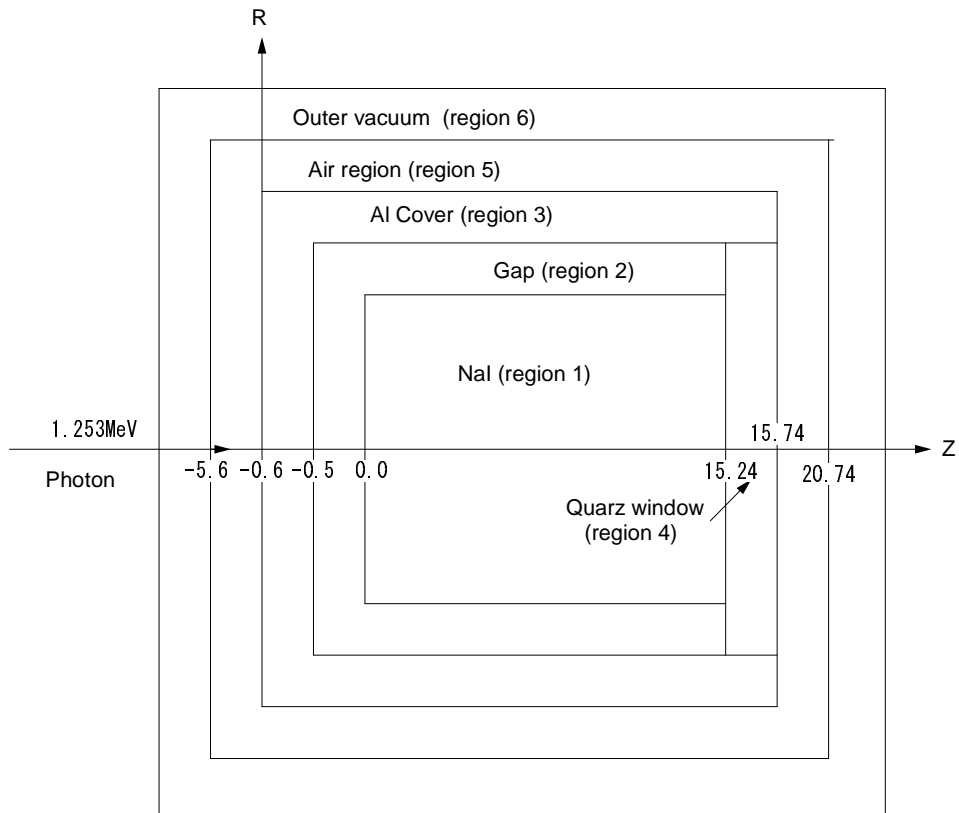


図 4: Geometry of ucnaicgv4.f90

```

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

```

- 作成した ucnaicgv4.data をチェックする。
 - CGview の”体型データ作成”のファイル作成を選択。
 - ファイルの種類として”すべてのファイル”とし、ucnaicgv4.data を選択する。
 - 表示を選択し、設定通りに形状となっていることを確認する。
 - ”設定”の”体型定義確認”を実施する。
- ucnaicgv4.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、ucnaicgv4 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 ucnaicgv4

- 計算が終了したら、CGview の”体系・飛跡データの読み込み”で、egs5job.pic を選択し、検出器の長さが2倍になっていることを確認する。
- egs5job.out を調べ、元の長さの結果と比較する。
結果の例: $P_{\text{eff}}=54.0 \pm 0.5\%$, $T_{\text{eff}}=93.6 \pm 0.3 \%$

6.2. 実習課題2

1. cp ucnaicgv.f ucnaicgv5.f
2. cp ucnaicgv.data ucnaicgv5.data
3. cp ucnaicgv.inp ucnaicgv5.inp
4. ucnaicgv5.f を以下のように変更する。

- 物質の変更

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

を

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

に変更する。ここで medarr のデータとして、24 文字を指定していることに注意せよ。

5. ucnaicgv5.inp の変更

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

を

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

に変更する。(1 行に2度、GE と書く場合は31 カラム目から書くことに注意せよ。)

6. ucnaicgv5.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、ucnaicgv5 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 ucnaicgv5
```

7. 計算が終了したら、egs5job.out を調べ、NaI の結果と比較する。
結果の例: $P_{\text{eff}}=39.8 \pm 0.5 \%$, $T_{\text{eff}}=87.7 \pm 0.3 \%$

6.3. 実習課題 3

1. cp ucnaicgv.f ucioncgv.f
2. cp ucnaicgv.data ucioncgv.data
3. cp ucnaicgv.inp ucioncgv.inp
4. ucioncgv.f を以下のように修正する。

- 計算結果の統計処理で使用する変数を追加

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

を

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

に変更。

- 使用する物質数の変更。

```
nmed=4
```

を

```
nmed=2
```

に変更。

- 物質名の変更。

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

を

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

に変更。

- 物質の数の減少にともない不要となった chard 指定文を削除。

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

を削除。

- 初期化変数の追加

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


を

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

に変更

- ヒストリー数を 100,000 に増やす。

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

を o

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

に変更する。

- 空気中での吸収エネルギーを合計するルーチンの追加。

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

を

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

に変更

- 円筒と平板数の減少に伴い、計算終了後の形状に関する出力部を変更する。

```
tdet=7.62
rdet=3.81
tcov=0.1
rtcov=0.1
tgap=0.5
rtgap=0.5
write(6,330) tdet,rdet,tcov,rtcov,tgap,rtgap
330 FORMAT(/' Detector length=',G15.5,' cm'/
*        ' Detector radius=',G15.5,' cm'/
*        ' A1 cover thickness=',G10.2,' cm'/
*        ' A1 cover side thickness=',G10.2,' cm'/
*        ' Front gap =',G10.2,' cm'/' Side gap =',G10.2,' cm'/)
```

を

```
tdet=7.62
rdet=3.81
tcov=0.5
rtcov=0.5
write(6,330) tdet,rdet,tcov,rtcov
330 FORMAT(/' Detector length=',G15.5,' cm'/
*        ' Detector radius=',G15.5,' cm'/
*        ' A1 cover thickness=',G10.2,' cm'/
*        ' A1 cover side thickness=',G10.2,' cm'/)
```

に変更する。

- 空気中での平均吸収エネルギーとその統計誤差を計算する以下のルーチン、及び出力を計算するルーチンを加える。

```

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

を

! -----
! Absorbed energy in air
! -----
avab = depes/ncount
depe2s = depe2s/ncount
sigab = sqrt((depe2s - avab*avab)/ncount)
write(6,362) avab,sigab
362 FORMAT(' Absorbed energy in air =',G10.3,'+-',G9.2,' MeV/photon')
avab = avab /33.97D-6 *1.602D-19
sigab= sigab /33.97D-6 *1.602D-19
write(6,364) avab,sigab
364 FORMAT(' Output current =',G10.3,'+-',G9.2,' C/photon')

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

```

に変更する。

- 空気が検出器と外側の2つの領域に存在するので、`ausgab`における検出器中での吸収エネルギーを指定する条件を変更する。

```

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

を

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

```

に変更する。

5. `ucioncgv.data` を以下のように変更する。

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

6. `ucioncgv.inp` を以下のように変更する。

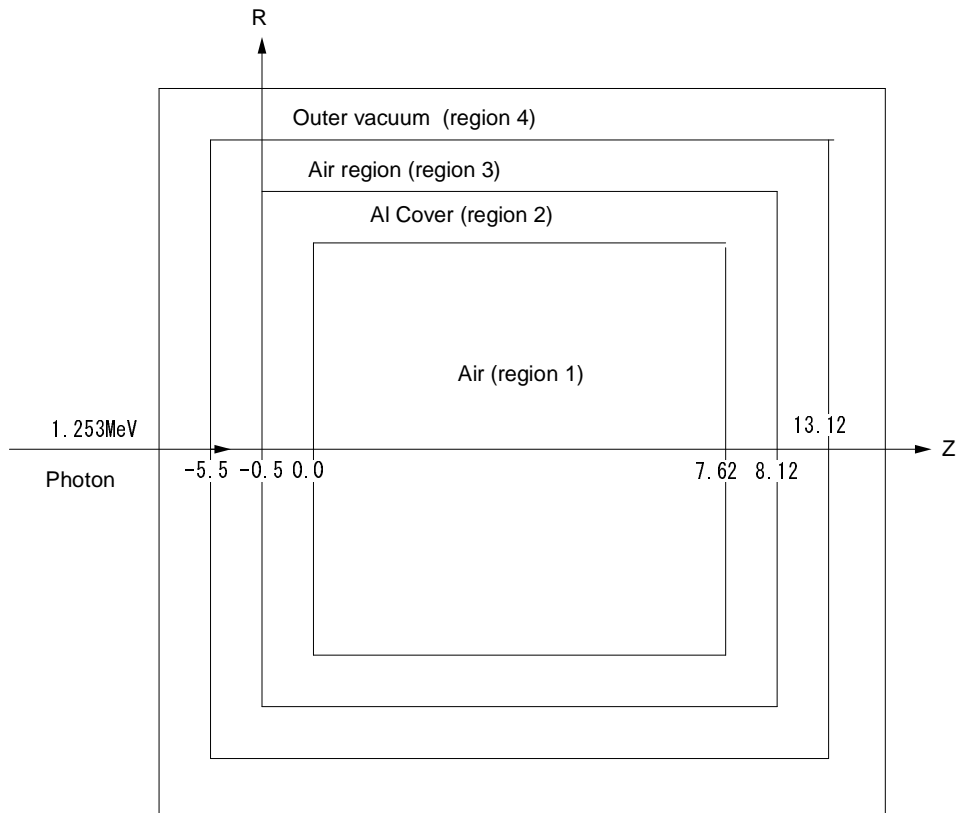


図 5: Geometry of ucioncgv.f

```

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

```

7. ucioncgv.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、ucioncgv 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 ucioncgv
```

8. 計算が終了したら、CGview の「体系・飛跡データの読み込み」で、egs5job.pic を選択し、形状及び飛跡が NaI の場合と違っていることを確認する。
9. egs5job.out を調べ、計算したい情報が得られていることを確認する。結果の例を下の表に示す。

電離箱の計算結果

| | |
|----------------------------|--------------------------------------|
| ピーク効率 P_{eff} (%) | 0.0 |
| 全効率 T_{eff} (%) | $0.73 \pm 0.3\text{e-}1$ |
| 吸収エネルギー (MeV/ γ) | $0.15\text{e-}3 \pm 0.7\text{e-}5$ |
| 出力 (C/ γ) | $0.69\text{e-}18 \pm 0.3\text{e-}19$ |

参考文献

- [1] T. Torii and T. Sugita, “Development of PRESTA-CG Incorporating Combinatorial Geometry in EGS4/PRESTA”, *JNC TN1410 2002-201*, Japan Nuclear Cycle Development Institute (2002).
- [2] T. Sugita, T. Torii, A. Takamura, “Incorporating Combinatorial Geometry to the EGS5 Code and Its Speed-Up”, Twelfth EGS User’s Meeting in Japan, KEK Proc. **2005-10**, 7-21, (KEK, Tsukuba, 9 - 11 Aug. 2005).

Appendix 1 Full listings of ucnaicgv.f

```

*****
***** KEK, High Energy Accelerator Research
***** Organization
*** ucnaicgv *****
***** EGS5.0 USER CODE - 28 Jul 2012/1430 *
*****
! This is a general User Code based on the cg geometry scheme.
*****
PROGRAMMERS: H. Hirayama
              Applied Research Laboratory
              KEK, High Energy Accelerator Research Organization
              1-1, Oho, Tsukuba, Ibaraki, 305-0801
              Japan
              E-mail: hideo.hirayama@kek.jp
              Telephone: +81-29-864-5451
              Fax: +81-29-864-4051
              Y. Namito
              Radiation Science Center
              Applied Research Laboratory
              KEK, High Energy Accelerator Research Organization
              1-1, Oho, Tsukuba, Ibaraki, 305-0801
              Japan
              E-mail: yoshihito.namito@kek.jp
              Telephone: +81-29-864-5489
              Fax: +81-29-864-1993
*****
! The ucnaicgv.f User Code requires a cg-input file only
! (e.g., ucnaicgv.data).
! The following shows the geometry for ucnaicgv.data.
! Input data for CG geometry must be written at the top of data-input
! file together with material assignment to each region. Cg-data can
! be checked by CGview.
! This user code corresponds to ucnaic3cgp.mor for egs4.
! Use Ranlux random number generator.
*****
-----
cg Geometry (ucnaicgv)
-----
      R
      |
      +-----+-----+-----+-----+
      | Outer vacuum region | R=9.41
      | Air                  | R=4.41
      | Al cover             | R=4.31
      | Gap                  | R=3.81
      | NaI                  |
      | Quartz               |
      +-----+-----+-----+-----+
1.253 MeV photons >-----> Z
      -5.6 -0.6 -0.5 0.0 7.62 8.12 13.12
      -5.0
*****
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12
-----
main code
-----
! Step 1: Initialization
-----

implicit none

```

```

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

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

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

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

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

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

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

real*8                               ! Local variables
* availke,avpe,avph,avspe,avspg,avssp,avte,desci2,pefs,pef2s,
* rr0,sigpe,sigte,sigph,sigspg,sigspe,sigssp,tefs,tef2s,wtin,wsum,
* xi0,yi0,zi0

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

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

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

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

character*24 medarr(MXMED)

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

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

! =====

```

```

    call counters_out(0)
    =====
!-----
! Step 2: pegs5-call
!-----
!-----
! Define media before calling PEGS5
!-----
    nmed=4
    if(nmed.gt.MXMED) then
        write(6,'(A,I4,A,I4,A/A)')
    *      ' nmed (' nmed,') larger than MXMED (' ,MXMED,')',
    *      ' MXMED in iclude egs5_h.f must be increase.'
        stop
    end if
!-----
! =====
! call block_set                      ! Initialize some general variables
! =====
    medarr(1)='NAI'                      '
    medarr(2)='AL'                       '
    medarr(3)='QUARTZ'                   '
    medarr(4)='AIR-AT-NTP'                '
    do j=1,nmed
        do i=1,24
            media(i,j)=medarr(j)(i:i)
        end do
    end do
    chard(1) = 7.62d0                    ! automatic step-size control
    chard(2) = 0.1d0
    chard(3) = 0.5d0
    chard(4) = 5.0d0
    write(6,fmt="( 'chard =',5e12.5)" (chard(j),j=1,nmed)
!-----
! Run KEK PEGS5 before calling HATCH
!-----
    write(6,100)
100  FORMAT('PEGS5-call comes next'/)
!-----
! =====
! call pegs5
! =====
!-----
! Step 3: Pre-hatch-call-initialization
!-----
    write(6,*) 'Read cg-related data'
!-----
!-----
! Initialize CG related parameters
!-----
    npreci=3      ! PICT data mode for CGView in free format
    ifti = 4      ! Input unit number for cg-data
    ifto = 39     ! Output unit number for PICT
    write(6,fmt="( ' CG data' )")
    call geomgt(ifti,6) ! Read in CG data
    write(6,fmt="( ' End of CG data',/ )")
    if(npreci.eq.3) write(ifto,fmt="( 'CSTA-FREE-TIME' )")
    if(npreci.eq.2) write(ifto,fmt="( 'CSTA-TIME' )")
    rewind ifti
    call geomgt(ifti,ifto)! Dummy call to write geom info for ifto
    write(ifto,110)
110  FORMAT('CEND')
!-----
!-----
! Get nreg from cg input data
!-----
    nreg=izonin

```

```

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

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

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

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

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

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

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

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

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

! -----
! Open files (before HATCH call)
! -----
  open(UNIT=KMPI,FILE='pgs5job.pegs5dat',STATUS='old')
  open(UNIT=KMPO,FILE='egs5job.dummy',STATUS='unknown')
  write(6,140)

```



```

140  FORMAT(/, ' HATCH-call comes next',/)
!
! =====
! call hatch
! =====
!
! -----
! Close files (after HATCH call)
! -----
!
! close(UNIT=KMPI)
! close(UNIT=KMPO)
!
! -----
! Print various data associated with each media (not region)
! -----
!
! write(6,150)
150  FORMAT(/, ' Quantities associated with each MEDIA:')
! do j=1,nmed
!     write(6,160) (media(i,j),i=1,24)
160  FORMAT(/,1X,24A1)
!     write(6,170) rhom(j),rlcm(j)
170  FORMAT(5X, ' rho=',G15.7, ' g/cu.cm      rlc=',G15.7, ' cm')
!     write(6,180) ae(j),ue(j)
180  FORMAT(5X, ' ae=',G15.7, ' MeV      ue=',G15.7, ' MeV')
!     write(6,190) ap(j),up(j)
190  FORMAT(5X, ' ap=',G15.7, ' MeV      up=',G15.7, ' MeV',/)
! end do
!
! -----
! Print media and cutoff energies assigned to each region
! -----
!
! do i=1,nreg
!     if (med(i) .eq. 0) then
200  write(6,200) i
!     FORMAT(' medium(',I3,')=vacuum')
!     else
!     write(6,210) i,(media(ii,med(i)),ii=1,24),ecut(i),pcut(i)
210  FORMAT(' medium(',I3,')=',24A1,
! *      'ecut=',G10.5, ' MeV, pcut=',G10.5, ' MeV')
!
! -----
! Print out energy information of K- and L-X-rays
! -----
!
!     if (iedgfl(i) .ne. 0) then          ! Output X-ray energy
!         ner = nne(med(i))
!         do iiz=1,ner
!             izn = zelem(med(i),iiz) ! Atomic number of this element
!             write(6,220) izn
220  FORMAT(' X-ray information for Z=',I3)
!             write(6,230) (ekx(ii,izn),ii=1,10)
230  FORMAT(' K-X-ray energy in keV',/,
! *          4G15.5,/,4G15.5,/,2G15.5)
!             write(6,240) (elx1(ii,izn),ii=1,8)
240  FORMAT(' L-1 X-ray in keV',/,4G15.5,/,4G15.5)
!             write(6,250) (elx2(ii,izn),ii=1,5)
250  FORMAT(' L-2 X-ray in keV',/,5G15.5)
!             write(6,260) (elx3(ii,izn),ii=1,7)
260  FORMAT(' L-3 X-ray in keV',/,4G15.5,/,3G15.5)
!         end do
!     end if
! end if
! end do
!
! write(39,fmt="( 'MSTA' )")
! write(39,fmt="(i4)") nreg
! write(39,fmt="(15i4)") (med(i),i=1,nreg)
! write(39,fmt="( 'MEND' )")
!
! -----
! Step 6: Initialization-for-howfar
! -----
!
! -----
! Step 7: Initialization-for-ausgab
! -----
!
! ncount = 0
! ilines = 0
! nwrite = 10
! nlines = 10
! idin = -1

```

```

totke = 0.
wtsum = 0.
iwatch=0

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

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

! Energy bin width
deltae=ekein / 50

! Zero the variables
depe=0.D0
pefs=0.D0
pef2s=0.D0
tefs=0.D0
tef2s=0.D0
do j=1,50
  phs(j)=0.D0
  ph2s(j)=0.D0
  do ntype=1,3
    spec(ntype,j)=0.D0
    specs(ntype,j)=0.D0
    spec2s(ntype,j)=0.D0
  end do
end do

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

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

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

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

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

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

wtin = 1.0

wtsum = wtsum + wtin
etot = ekein + iabs(iqin)*RM
if(iqin.eq.1) then
  availke = ekein + 2.0*RM
else
  availke = ekein
end if

totke = totke + availke

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

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

```

```

280     FORMAT(7G15.7,3I5)
       end if

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

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

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

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

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

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

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

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

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

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

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

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

! -----
! Step 9:  Output-of-results
! -----
       write(6,310) ncount,ncases,totke
310  FORMAT(/,' Ncount=',I10,' (actual cases run)',/,
*        ' Ncases=',I10,' (number of cases requested)',/,

```

```

*      ' TotKE =',G15.5,' (total KE (MeV) in run)')

if (totke .le. 0.D0) then
  write(6,320) totke,availke,ncount
320  FORMAT('/',' Stopped in MAIN with TotKE=',G15.5,/,
*      ' AvailKE=',G15.5, /, ' Ncount=',I10)
  stop
end if

tdet=7.62
rdet=3.81
tcov=0.1
rtcov=0.1
tgap=0.5
rtgap=0.5
write(6,330) tdet,rdet,tcov,rtcov,tgap,rtgap
330  FORMAT('/',' Detector length=',G15.5,' cm'/
*      ' Detector radius=',G15.5,' cm'/
*      ' Al cover thickness=',G10.2,' cm'/
*      ' Al cover side thickness=',G10.2,' cm'/
*      ' Front gap =',G10.2,' cm'/ ' Side gap =',G10.2,' cm'/)

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

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

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

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

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

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

! -----
! Particle spectrum. Incident particle spectrum to detector.
! -----
write(6,400)
400  FORMAT('/',' Particle spectrum crossing the detector plane'/
*      30X,'particles/MeV/source photon'/
*      ' Upper energy',11X,' Gamma',18X,' Electron',
*      14X,' Positron')

```

```

do ie=1,50
  elow=deltae*(ie-1)
  eup=deltae*ie
!
! -----
! Gamma spectrum per MeV per source
! -----
!
  avspg = specs(1,ie)/ncount
  spec2s(1,ie)=spec2s(1,ie)/ncount
  sigspg=dsqrt((spec2s(1,ie)-avspg*avspg)/ncount)
  avspg=avspg/deltae
  sigspg= sigspg/deltae
!
! -----
! Electron spectrum per MeV per source
! -----
!
  avspe = specs(2,ie)/ncount
  spec2s(2,ie)=spec2s(2,ie)/ncount
  sigspe=dsqrt((spec2s(2,ie)-avspe*avspe)/ncount)
  avspe= avspe/deltae
  sigspe= sigspe/deltae
!
! -----
! Positron spectrum per MeV per source
! -----
!
  avspg = specs(3,ie)/ncount
  spec2s(3,ie)=spec2s(3,ie)/ncount
  sigspp=dsqrt((spec2s(3,ie)-avspg*avspg)/ncount)
  avspg= avspg/deltae
  sigspp= sigspp/deltae
!
  write(6,410) eup,avspg,sigspg,avspe,sigspe,avspg,sigspp
410  FORMAT(G10.5,' MeV--',3(G12.5,'+-',G12.5))
end do

nlist=1

!
! =====
! call ecnsv1(nlist,nreg,totke)
! call ntally(nlist,nreg)
! =====
!
! =====
! call counters_out(1)
! =====
!
stop
end

!-----last line of main code-----
!-----ausgab.f-----
! Version: 080708-1600
! Reference: SLAC-265 (p.19-20, Appendix 2)
!-----
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12
!
! -----
! Required subroutine for use with the EGS5 Code System
! -----
! A AUSGAB to:
!
! 1) Score energy deposition
! 2) Score particle information enter to detector from outside
! 3) Print out particle transport information
! 4) call plotxyz if imode=0
! -----
!
subroutine ausgab(iarg)
implicit none

```

```

include 'include/egs5_h.f'           ! Main EGS "header" file
include 'include/egs5_epcont.f'     ! COMMONs required by EGS5 code
include 'include/egs5_misc.f'
include 'include/egs5_stack.f'
include 'include/egs5_useful.f'

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

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

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

integer                            ! Arguments
* iarg

real*8                             ! Local variables
* edepwt

integer
* ie,iql,irl,ntype

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

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

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

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

! -----
! Score particle information if it enters from outside
! -----
if (irl .ne. iold .and. iarg .eq. 0) then
  if (iql .eq. 0) then           ! photon
    ntype=1
    ie = e(np)/deltae +1
    if(ie .gt. 50) ie = 50
  elseif (iql .eq. -1) then     ! electron
    ntype=2
    ie = (e(np) - RM)/deltae +1
    if(ie .gt. 50) ie = 50
  else                          ! positron
    ntype=3
    ie = (e(np) - RM)/deltae +1
    if(ie .gt. 50) ie = 50
  end if
  spec(ntype,ie) = spec(ntype,ie) + wt(np)
end if
end if

```

```

! -----
! Print out stack information (for limited number cases and lines)
! -----
      if (ncount .le. nwrite .and. ilines .le. nlines) then
            ilines = ilines + 1
            write(6,100) e(np),x(np),y(np),z(np),u(np),v(np),w(np),
*           iql,irl,iarg
100      FORMAT(7G15.7,3I5)
      end if

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

      return

      end

!-----last line of ausgab.f-----
!-----howfar.f-----
! Version: 070627-1600
! Reference: T. Torii and T. Sugita, "Development of PRESTA-CG
! Incorporating Combinatorial Geometry in EGS4/PRESTA", JNC TN1410 2002-201,
! Japan Nuclear Cycle Development Institute (2002).
! Improved version is provided by T. Sugita. 7/27/2004
!-----
!23456789|123456789|123456789|123456789|123456789|123456789|123456789|12
! -----
! Required (geometry) subroutine for use with the EGS5 Code System
! -----
! This is a CG-HOWFAR.
! -----

      subroutine howfar
      implicit none

c
      include 'include/egs5_h.f'          ! Main EGS "header" file
      include 'include/egs5_epcont.f'    ! COMMONs required by EGS5 code
      include 'include/egs5_stack.f'
      include 'auxcommons/geom_common.f' ! geom-common file

c
c
      integer i,j,jjj,ir_np,nozone,jty,kno
      integer irnear,irnext,irlold,irlfg,itvlf,ihitcg
      double precision xidd,yidd,zidd,x_np,y_np,z_np,u_np,v_np,w_np
      double precision tval,tval0,tval00,tval10,tvalmn,delhow
      double precision atvalttmp
      integer iq_np

c
      ir_np = ir(np)
      iq_np = iq(np) + 2

c
      if(ir_np.le.0) then
            write(6,*) 'Stopped in howfar with ir(np) <=0'
            stop
      end if

c
      if(ir_np.gt.izonin) then
            write(6,*) 'Stopped in howfar with ir(np) > izonin'
            stop
      end if

c
      if(ir_np.EQ.izonin) then
            idisc=1
            return
      end if

c
      tval=1.d+30
      itvalm=0

c
c
      body check
      u_np=u(np)
      v_np=v(np)
      w_np=w(np)

```

```

x_np=x(np)
y_np=y(np)
z_np=z(np)
c
do i=1,nbbody(ir_np)
  nozone=ABS(nbzone(i,ir_np))
  jty=itblty(nozone)
  kno=itblno(nozone)
c  rpp check
  if(jty.eq.ityknd(1)) then
    if(kno.le.0.or.kno.gt.irppin) go to 190
    call rppcg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c  sph check
  elseif(jty.eq.ityknd(2)) then
    if(kno.le.0.or.kno.gt.isphin) go to 190
    call sphcg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c  rcc check
  elseif(jty.eq.ityknd(3)) then
    if(kno.le.0.or.kno.gt.irccin) go to 190
    call rcccg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c  trc check
  elseif(jty.eq.ityknd(4)) then
    if(kno.le.0.or.kno.gt.itrcin) go to 190
    call trccg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c  tor check
  elseif(jty.eq.ityknd(5)) then
    if(kno.le.0.or.kno.gt.itorin) go to 190
    call torcg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c  rec check
  elseif(jty.eq.ityknd(6)) then
    if(kno.le.0.or.kno.gt.irecin) go to 190
    call reccg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c  ell check
  elseif(jty.eq.ityknd(7)) then
    if(kno.le.0.or.kno.gt.iellin) go to 190
    call ellcg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c  wed check
  elseif(jty.eq.ityknd(8)) then
    if(kno.le.0.or.kno.gt.iwedin) go to 190
    call wedcg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c  box check
  elseif(jty.eq.ityknd(9)) then
    if(kno.le.0.or.kno.gt.iboxin) go to 190
    call boxcg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c  arb check
  elseif(jty.eq.ityknd(10)) then
    if(kno.le.0.or.kno.gt.iarbin) go to 190
    call arbcg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c  hex check
  elseif(jty.eq.ityknd(11)) then
    if(kno.le.0.or.kno.gt.ihexin) go to 190
    call hexcg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c  haf check
  elseif(jty.eq.ityknd(12)) then
    if(kno.le.0.or.kno.gt.ihafin) go to 190
    call hafcg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c  tec check
  elseif(jty.eq.ityknd(13)) then
    if(kno.le.0.or.kno.gt.itecin) go to 190
    call teccg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c  gel check
  elseif(jty.eq.ityknd(14)) then
    if(kno.le.0.or.kno.gt.igelin) go to 190
    call gelcg1(kno,x_np,y_np,z_np,u_np,v_np,w_np)
c
c**** add new geometry in here
c
  end if
190 continue
end do
c
irnear=ir_np
if(itvalm.eq.0) then
  tval0=cgeps1
  xidd=x_np+tval0*u_np

```



```

yidd=y_np+tval0*v_np
zidd=z_np+tval0*w_np
310 continue
    if(x_np.ne.xidd.or.y_np.ne.yidd.or.z_np.ne.zidd) goto 320
    tval0=tval0*10.d0
    xidd=x_np+tval0*u_np
    yidd=y_np+tval0*v_np
    zidd=z_np+tval0*w_np
    go to 310
320 continue
c write(*,*) 'srzone:1'
c call srzone(xidd,yidd,zidd,iq_np,ir_np,irnext)

if(irnext.ne.ir_np) then
    tval=0.0d0
    irnear=irnext
else
    tval00=0.0d0
    tval10=10.0d0*tval0
    irlold=ir_np
    irlfg=0
330 continue
    if(irlfg.eq.1) go to 340
    tval00=tval00+tval10
    if(tval00.gt.1.0d+06) then
        write(6,9000) iq(np),ir(np),x(np),y(np),z(np),
& u(np),v(np),w(np),tval00
9000 format(' TVAL00 ERROR : iq,ir,x,y,z,u,v,w,tval=',
& 2I3,1P7E12.5)
        stop
    end if
    xidd=x_np+tval00*u_np
    yidd=y_np+tval00*v_np
    zidd=z_np+tval00*w_np
    call srzold(xidd,yidd,zidd,irlold,irlfg)
    go to 330
340 continue

tval=tval00
do j=1,10
    xidd=x_np+tval00*u_np
    yidd=y_np+tval00*v_np
    zidd=z_np+tval00*w_np
c write(*,*) 'srzone:2'
c call srzone(xidd,yidd,zidd,iq_np,irlold,irnext)
    if(irnext.ne.irlold) then
        tval=tval00
        irnear=irnext
    end if
    tval00=tval00-tval0
end do
if(ir_np.eq.irnear) then
    write(0,*) 'ir(np),tval=',ir_np,tval
end if
end if
else
do j=1,itvalm-1
do i=j+1,itvalm
    if(atval(i).lt.atval(j)) then
        atvaltmp=atval(i)
        atval(i)=atval(j)
        atval(j)=atvaltmp
    endif
enddo
enddo
itvlf=0
tvalmn=tval
do jjj=1,itvalm
    if(tvalmn.gt.atval(jjj)) then
        tvalmn=atval(jjj)
    end if
    delhow=cgeps2
    tval0=atval(jjj)+delhow
    xidd=x_np+tval0*u_np
    yidd=y_np+tval0*v_np
    zidd=z_np+tval0*w_np
410 continue
    if(x_np.ne.xidd.or.y_np.ne.yidd.or.z_np.ne.zidd) go to 420
    delhow=delhow*10.d0

```

```

        tval0=atval(jjj)+delhow
        xidd=x_np+tval0*u_np
        yidd=y_np+tval0*v_np
        zidd=z_np+tval0*w_np
420      go to 410
        continue
c      write(*,*) 'srzone:3'
        call srzone(xidd,yidd,zidd,iq_np,ir_np,irnext)
        if((irnext.ne.ir_np.or.atval(jjj).ge.1.).and.
&         tval.gt.atval(jjj)) THEN
            tval=atval(jjj)
            irnear=irnext
            itvlf=1
            goto 425
        end if
    end do
425      continue
        if(itvlf.eq.0) then
            tval=cgmnst
            xidd=x_np+tval0*u_np
            yidd=y_np+tval0*v_np
            zidd=z_np+tval0*w_np
430          continue
            if(x_np.ne.xidd.or.y_np.ne.yidd.or.z_np.ne.zidd) go to 440
                tval0=tval0*10.d0
                xidd=x_np+tval0*u_np
                yidd=y_np+tval0*v_np
                zidd=z_np+tval0*w_np
                go to 430
440          continue
            if(tvalmn.gt.tval0) then
                tval=tvalmn
            else
                tval=tval0
            end if
        end if
    end if
    ihitcg=0
    if(tval.le.ustep) then
        ustep=tval
        ihitcg=1
    end if
    if(ihitcg.eq.1) THEN
        if(irnear.eq.0) THEN
            write(6,9200) iq(np),ir(np),x(np),y(np),z(np),
&                u(np),v(np),w(np),tval
9200 format(' TVAL ERROR : iq,ir,x,y,z,u,v,w,tval=',2I3,1P7E12.5)
            idisc=1
            itverr=itverr+1
            if(itverr.ge.100) then
                stop
            end if
            return
        end if
        irnew=irnear
        if(irnew.ne.ir_np) then
            call rstnxt(iq_np,ir_np,irnew)
        endif
    end if
    return
end
!-----last line of subroutine howfar-----

```