

# サンプルユーザーコード ucphantomgv

平山 英夫、波戸 芳仁  
KEK, 高エネルギー加速器研究機構

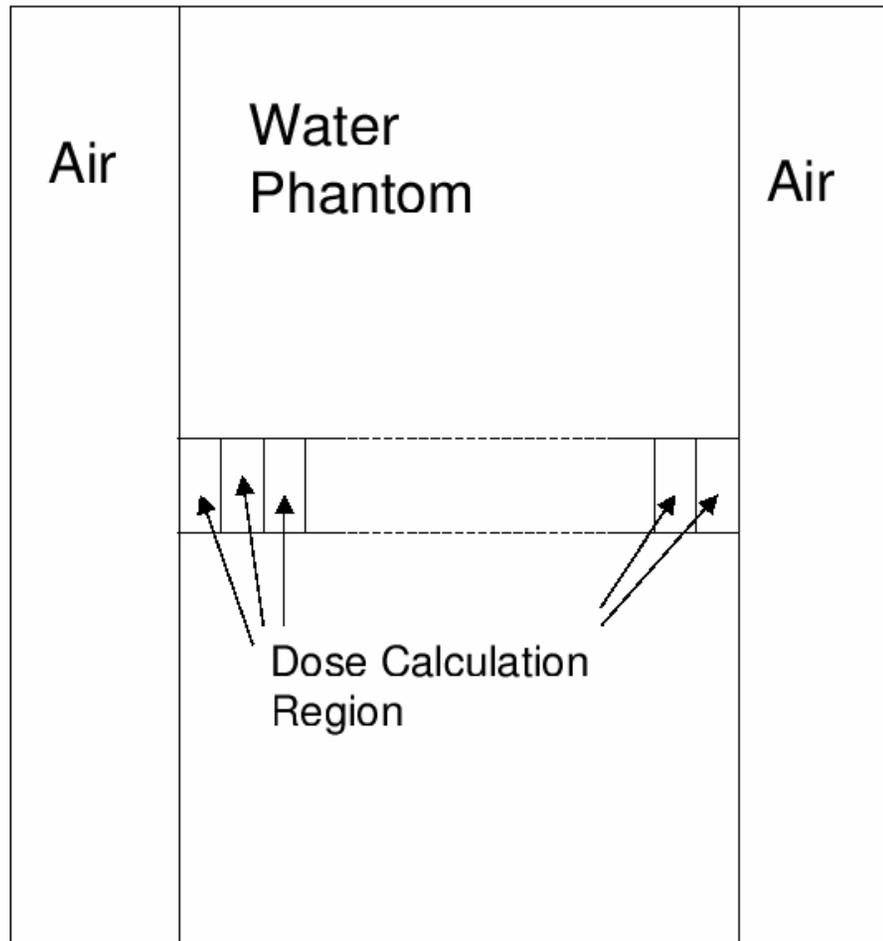
テキスト: phantomcgv.pdf,  
Egs5\_user\_manual.pdf

ユーザーコードで利用可能な変数、  
オプションについては  
egs5\_user\_manualを参照

# ucphantomcgv.f

- 計算課題: 水ファントム中での吸収線量の計算
- 形状: CG形状(RPP: 直方体)
- 線源光子エネルギー: 1.253MeV
- 点等方線源の位置 SPOSI (10cm)
- ファントム表面でのX-方向の半値幅 (xhbeam=1cm)、Y-方向の半値幅 (yhbeam=1cm)
- 得られる結果
  - 飛跡表示データ(CGView): egs5job.pic
  - 計算結果: egs5job.out

# Outside Vacuum Region



All bodies are written as RPP

# Step 1: Initialization

- egs5及びpegs5で使われているcommonは、それぞれincludeディレクトリー及びpegscommonsディレクトリーのファイルを ”include”文で取り込む
- 著者から提供されたジオメトリー関係などのユーザーコードのみで使用されるcommonは、auxcommonsディレクトリーのファイルをinclude文で取り込む

# 配列の大きさの指定

- commonで使用されている変数の配列の大きさは、parameter文で指定
  - egs5で使用されているcommonの変数は、  
include/egs5\_h.f
  - ユーザーコードでのみ使用されるcommonの変数は、  
auxcommns/aux\_h.f
- commonと同じようにinclude文により取り込まれる。
- 配列の大きさを変更する場合は、parameter文の変数を変更する

```
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'
```

egs5 common に含まれる変数をメインプログラム等のプログラム単位で使用する場合は、include文で当該commonを指定

**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'**

ジオメトリー関係等ユーザーコード  
のみで使用されるcommon

CG関係のcommonで、CGを使用する場合には常に必要(変更無し)

**include 'auxcommons/geom\_common.f' ! geom-common file  
integer irinn**

In include/egs5\_h.f

**! Maximum number of different media (excluding vacuum)**

**integer MXMED**

**parameter (MXMED = 4)**

物質の数を増やしたい場合には、この数値を変更する。

include/egs5\_misc.f

**common/MISC/**

**! Miscellaneous COMMON**

**\* rhor(MXREG), dunit,**

**\* med(MXREG), iraylr(MXREG), lpolar(MXREG), incohr(MXREG),**

**\* iprofr(MXREG), impacr(MXREG),**

**\* kmpi, kmpo, noscat**

**real\*8**

**\* rhor, dunit**

**integer**

**\* med, iraylr, lpolar, incohr, iprofr, impacr, kmpi, kmpo, noscat**

**common/totals/**

**! Variables to score**

**\* depe(20),faexp,fexps,maxpict,ndet**  
**real\*8 depe,faexp,fexps**  
**integer maxpict,ndet**

このユーザーコード固有  
のcommon

main programで使用する倍精度の実数

**!\*\*\*\* real\*8**

**! Local variables**

**real\*8**

**\* area,availke,depthl,depths,dis,disair,ei0,ekin,elow,eup,**  
**\* phai0,phai,radma2,sinth,sposi,tnum,vol,w0,wimin,wtin,wtsum,**  
**\* xhbeam,xpf,yhbeam,ypf**

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

**\* faexpa,fexpsa**

**real\*8**

**\* depeh(20),depeh2(20),dose(20),dose2(20),doseun(20)**

main programで使用する単精度の実数

**real**

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

main programで使用する整数

**integer**

\* **i,ii,ibatch,icases,idin,ie,ifti,ifto,imed,ireg,isam,**

\* **ixtype,j,k,kdet,nnn**

物質名に使用する文字変数(24文字)

**character\*24 medarr(2)**

# Open文

- ユーザーコードから、pegsを実行するのに伴い、ユニット7-26は、pegsで close されることから、メインプログラムで open していても、pegs実行後に、再度 open することが必要となる。そのため、ユニット7-26の使用を避ける方が良い。
- ユニット39は、飛跡情報の出力ファイルである。

# Step 2:pegs5-call

- 物質データ及び各物質のcharacteristic distanceを設定した後で、pegs5をcallする。

```
nmed=2  
medarr(1)='WATER-IAPRIM-PHOTX'  
medarr(2)='AIR-AT-NTP-IAPRIM'
```

pegs5で作成する物質データの名前。pegs5の入力データ(ユニット24から読み込み)と対応

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

各物質のcharacteristic dimension

当該物質のリージョンで中、最も小さいサイズを指定

```
chard(1) = 1.0d0 ! automatic step-size control  
chard(2) = 1.0d0
```

# Step 3:Pre-hatch-call-initialization

```
!-----  
!   Define pict data mode.  
!-----
```

```
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
```

CG関連の処理を行う部分。

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

CGを使用する場合は、変更しない。

```
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
```

# CG形状(RPP:直方体で構成)

- ファントム前の空気層
- ファントムの領域
- ファントム内の線量計算をする領域
- ファントム後の空気層
- 体系全体を覆う領域(計算終了の領域を定義するために設定)

RPP	1	-15.0	15.0	-15.0	15.00	-5.0	0.00	← 空気層
RPP	2	-15.0	15.0	-15.0	15.00	0.0	20.00	← ファントム
RPP	3	-0.5	0.5	-0.5	0.50	0.0	1.00	線量計算を したい領域 を定義する ためのbody
RPP	4	-0.5	0.5	-0.5	0.50	1.0	2.00	
RPP	5	-0.5	0.5	-0.5	0.50	2.0	3.00	
RPP	6	-0.5	0.5	-0.5	0.50	3.0	4.00	
RPP	7	-0.5	0.5	-0.5	0.50	4.0	5.00	
RPP	8	-0.5	0.5	-0.5	0.50	5.0	6.00	
RPP	9	-0.5	0.5	-0.5	0.50	6.0	7.00	
RPP	10	-0.5	0.5	-0.5	0.50	7.0	8.00	
RPP	11	-0.5	0.5	-0.5	0.50	8.0	9.00	

RPP	17	-0.5	0.5	-0.5	0.50	14.0	15.00
RPP	18	-0.5	0.5	-0.5	0.50	15.0	16.00
RPP	19	-0.5	0.5	-0.5	0.50	16.0	17.00
RPP	20	-0.5	0.5	-0.5	0.50	17.0	18.00
RPP	21	-0.5	0.5	-0.5	0.50	18.0	19.00
RPP	22	-0.5	0.5	-0.5	0.50	19.0	20.00
RPP	23	-0.5	0.5	-0.5	0.50	0.0	20.00
RPP	24	-15.0	15.0	-15.0	15.00	20.0	25.00
RPP	25	-20.0	20.0	-20.0	20.00	-20.0	40.00

線量計算を  
したい領域を  
定義するた  
めのbody

線量計算の全領域  
を包含するbody

背後の空気層

体系全体を覆う  
body

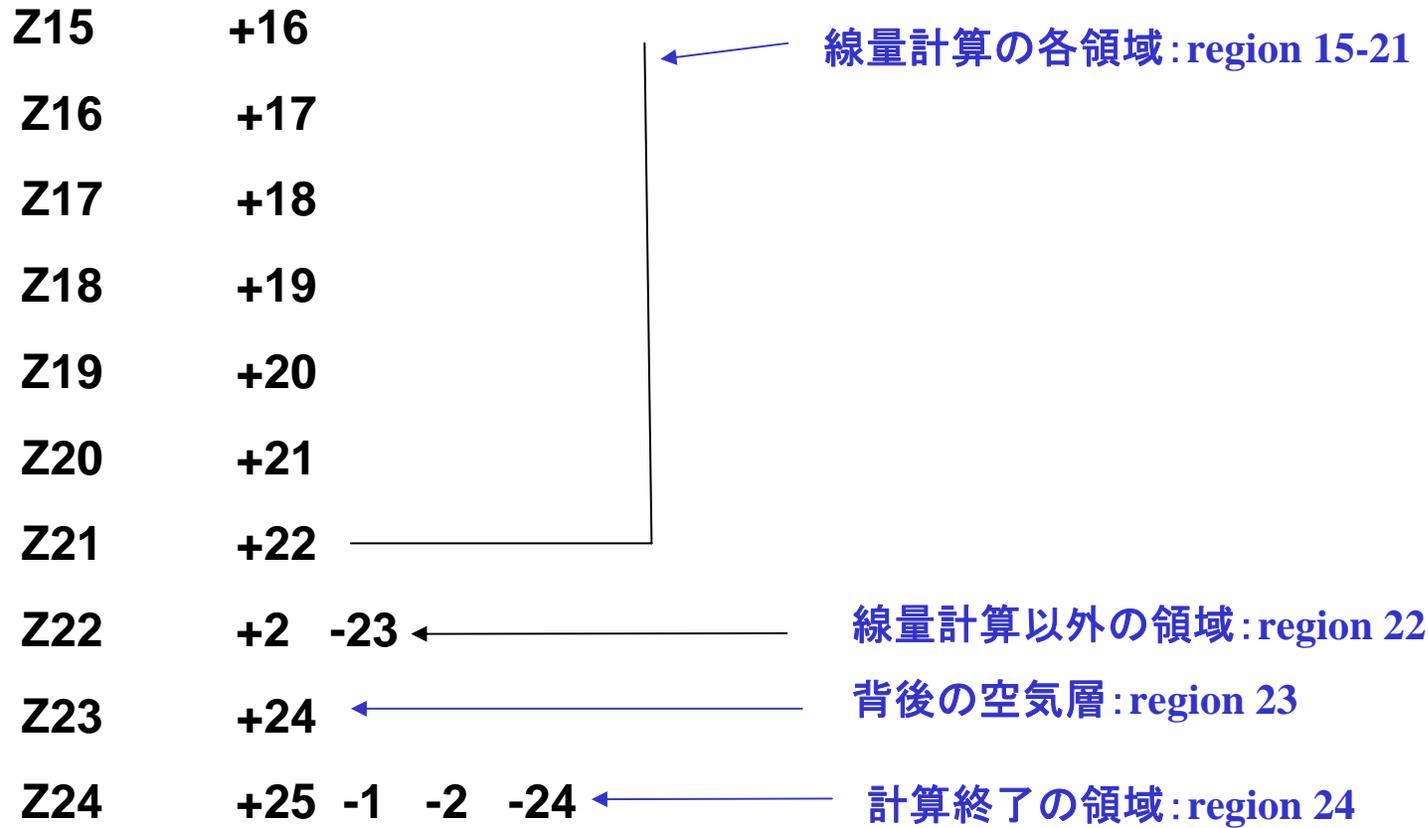
Z1 +1  
Z2 +3  
Z3 +4  
Z4 +5  
Z5 +6  
Z6 +7  
Z7 +8  
Z8 +9  
Z9 +10  
Z10 +11  
Z11 +12  
Z12 +13  
Z13 +14  
Z14 +15



ファントム前の空気: region 1



線量計算の各領域: region 2-14



## 各リージョンへの物質、各種オプションの設定

! Read material for each region from egs5job.data

```
read(4,*) (med(i),i=1,nreg)
```

! Set option except vacuum region

ファントムリージョンで、光電子の角度分布、特性X線、レイリー散乱オプションを設定

```
do i=2,nreg-2
```

```
  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) = 1  ! 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
```

## リージョン毎に設定できるオプション

<b>ecut, pcut</b>	カットオフエネルギー (全エネルギー)
<b>iphtr</b>	光電子の角度分布のサンプリング
<b>iedgfl</b>	K & L-特性X線の発生
<b>iauger</b>	K & L-オージェ電子の発生
<b>iraylr</b>	レイリー散乱
<b>lpolar</b>	光子散乱での直線偏光
<b>incohr</b>	S/Z rejection
<b>iprofr</b>	ドップラー広がり
<b>impacr</b>	電子衝突電離

# 乱数(ranlux乱数)

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

異なったinseed毎に、重複しない乱数を発生することが可能

並列計算の場合に有効

# Step 4: 入射粒子のパラメーター設定

!-----  
! Define source position from phantom surface.

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

iqin=0           ! Incident charge - photons  
ekein=1.235       ! Kinetic energy of source photon  
etot=ekein + abs(iqin)\*RM

xin=0.D0

yin=0.D0

粒子の種類、エネルギー位置、方向

zin=-sposi

入射粒子の属するリージョン (irin=0; cg 情報から計算して決定)

uin=0.D0

vin=0.D0

win=1.D0

irin=0   ! Starting region (0: Automatic search in CG)

## ファントム表面でのX及びYの半 値幅の設定

!-----

! Half width and height at phantom surface

!-----

! X-direction half width of beam at phantom surface in cm.

xhbeam=1.0

! Y-direction half height of beam at phantom surface in cm.

yhbeam=1.0

radma2=xhbeam\*xhbeam+yhbeam\*yhbeam

wimin=sposi/dsqrt(sposi\*sposi+radma2)

半値幅に対応した $\theta$ に対応する $\cos \theta$

# Step 5: hatch-call

- 電子・陽電子の全エネルギーの最大値をemaxeを0.d0に設定し、hatch を call する。(hatchで、emaxeを計算する。)
- 読み込んだ情報を確認するために、物質データ及び各リージョンの情報を出力する

```
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
```

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

# Step 6: Initialization-for-howfar

- ユーザーコードで使用する形状データを設定する
  - 平板、円筒、球などに関するデータ
- CGを使用しているこのユーザーコードでは、形状に関するデータは、cg入力データとしてstep 6以前に処理しているので、このstepで設定することはない

飛跡情報ファイルへの物質データ等の出力

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

# Step 7: Initialization-for-ausgab

- 計算で求める量の初期化
- 中心領域で、線量計算をするリージョンの数
- 計算したいヒストリー数(ncases)、飛跡データを記録するヒストリー数の設定

```
!-----  
! History number
```

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

```
! Maximum history number to write trajectory data  
maxpict=100  
iwatch=0
```

飛跡情報ファイルへのバッチ番号出力

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

## Step 8: Shower-call

- ncases数のヒストリー実行する
- 各ヒストリー毎に、線源情報(粒子の種類、エネルギー、位置、方向)を設定
- 線源粒子の座標から、線源粒子のリージョン番号をサーチ

```
240  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 240
      if (sposi.gt.5.0) then
        disair=(sposi-5.0)/win
        xin=disair*uin
        yin=disair*vin
        zin=-5.D0
      else
        xin=0.D0
        yin=0.D0
        zin=-sposi
      end if
```

線源の方向と位置の決定

ファントム表面での位置を計算し、設定した半値幅の領域からはみ出した場合には、サンプリングをやり直す

線源の位置が空気層の外側の場合、空気層の入り口での位置を入射粒子の位置として設定

入射粒子の位置から、その場所のリージョン番号を求める  
irin=0なので、ここでリージョン番号が設定される

!-----  
! **Get source region from cg input data**  
!-----

```
if(irin.le.0.or.irin.gt.nreg) then
  call srzone(xin,yin,zin,iqin+2,0,irinn)
  if(irinn.le.0.or.irinn.ge.nreg) then
    write(6,fmt="(' Stopped in MAIN. irinn = ',i5)")irinn
    stop
  end if
  call rstnxt(iqin+2,0,irinn)
else
  irinn=irin
end if
```

## 入射エネルギー及び方向余弦の規格化のチェック

```
! -----  
! Compare maximum energy of material data and incident energy  
! -----  
if(etot+(1-iabs(iqin))*RM.gt.emaxe) then  
  write(6,fmt="(' Stopped in MAIN.',  
1  ' (Incident kinetic energy + RM) > min(UE,UP+RM).')")  
  stop  
end if  
  
! -----  
! Verify the normarization of source direction vector  
! -----  
if(abs(uin*uin+vin*vin+win*win-1.0).gt.1.e-6) then  
  write(6,fmt="(' Following source direction vector is not',  
1  ' normarized.',3e12.5)")uin,vin,win  
  stop  
end if
```

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

計算したい量の平均値とその分散を求めるために、ヒストリー毎の値とその自乗を加える

```
do kdet=1,ndet  
  depeh(kdet)=depeh(kdet)+depe(kdet)  
  depeh2(kdet)=depeh2(kdet)+depe(kdet)*depe(kdet)  
  depe(kdet)=0.0  
end do  
  
faexps=faexps+faexp  
faexp2s=faexp2s+faexp*faexp  
faexp=0.0  
fexpss=fexpss+fexps  
fexp2s=fexp2s+fexps*fexps  
fexps=0.0
```

# 統計的な誤差評価

- $x$  をモンテカルロ計算によって求める量とする。
- MCNPで使用している誤差を評価する方法
  - 計算は  $N$  個の“入射”粒子について行われ、 $x_i$  は、 $i$ -番目のヒストリーの結果であるとする

$$\bar{x} = \frac{1}{N} \sum_{i=1}^N x_i \quad x_i \text{ の平均値}$$

$$s^2 = \frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2 \approx \overline{x^2} - (\bar{x})^2; (\overline{x^2} = \frac{1}{N} \sum_{i=1}^N x_i^2) \quad x_i \text{ の分散}$$

$$s_{\bar{x}}^2 = \frac{1}{N} s^2 \approx \frac{1}{N} [\overline{x^2} - \bar{x}^2] \quad \bar{x} \text{ の分散}$$

$$s_{\bar{x}} \approx \left[ \frac{1}{N} [\overline{x^2} - \bar{x}^2] \right]^{1/2} \quad \text{標準偏差}$$

## Step 9: Output-of-results

- 線源条件や、形状等の情報の出力
  - どのような計算であるかを示すために出力
  - cgの場合は、形状をデータから直接示すことが容易でないので、必要な情報を設定して出力する
- 平均値の和とその自乗の和から、求めたい量の平均値と誤差を計算し、出力する

# 吸收線量

```
area=1.D0*1.D0
do kdet=1,ndet
  vol=area*1.D0
  dose(kdet)=depeh(kdet)/ncases
  dose2(kdet)=depeh2(kdet)/ncases
  doseun(kdet)=dsqrt((dose2(kdet)-dose(kdet)*dose(kdet))/ncases)
  dose(kdet)=dose(kdet)*1.602E-10/vol
  doseun(kdet)=doseun(kdet)*1.602E-10/vol
  depths=kdet-1.0
  depthl=kdet
  write(6,320)depths,depthl,(media(ii,med(kdet+1)),ii=1,24),
* rhor(kdet+1),dose(kdet),doseun(kdet)
320  FORMAT(' At ',F4.1,'--',F4.1,'cm (',24A1,',rho:',F8.4,')=',
* G13.5,'+-',G13.5,'Gy/incident')
end do
```

# ausgab

- ausgab は、ユーザーが得たい情報を記録するサブルーチンである
- ファントム領域での吸収線量
- ファントム表面での照射線量

```
if (irl.ge.2.and.irl.le.nreg-3) then
  idet=irl-1
  if(idet.ge.1.and.idet.le.ndet) then
    depe(idet)=depe(idet)+edepwt/rhor(irl)
  end if
end if
```

線量計算の領域の粒子の場合、単位重量当たりの吸収線量を積算する

`rhor(irl)`は、当該リージョンの密度

## 照射線量の計算

```
if (abs(irl-irold).eq.1.and.iq(np).eq.0) then
  if((w(np).gt.0.0.and.irl.eq.2).or.(w(np).le.0.0.and.irl.eq.1))
* then
  if (dabs(w(np)).ge.0.0349) then
    cmod=dabs(w(np))
  else
    cmod=0.0175
  end if
  esing=e(np)
  dcon=encoa(esing)
  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
```

光子が面を横切った場合

ファントム前面の場合

平面粒子束: 単位面積を通過する粒子束の計算  $-\cos\theta$  の補正

エネルギーESINGの光子に対する空気の質量吸収係数

! PHOTX data

# howfar

- howfar は、egs にジオメトリーに関する情報を伝えるサブルーチン
- howfar は、ustep の途中で、リージョン境界があるかどうかを調べる。ある場合には、
  - ustep を境界までの距離に置き換える
  - irnew を粒子が入っていくリージョン番号に設定する
- 粒子が、ユーザーが追跡を止めたい領域(例: 体系外)に達したばあいには、idiscard フラグを1に設定する
- 使用するジオメトリルーティン毎に異なったhowfarとなる
  - cgを使用している場合は、このユーザーコードのhowfarを使用する

# 実習課題

- 実習課題1: 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。
- 実習課題2: 100kVのX線 (スペクトルデータは、xray.datから読み込み) データを用いてサンプリングする。
- 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 $0.3\text{g/cm}^3$ )とし、その背後に3cm の人体組織がある体系に変更する。線源は、元のX線とする。
- 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。
- 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。

## 変更記録

- 2009-6-24
  - ucphantomcgv.fとの整合性をとる