KEK Report 2016-2
June 2016
R/D

β-ray Spectrum Data for egs5 based on ICRU-56 or RADAR

Y. Kirihara, H. Hirayama and Y. Namito

High Energy Accelerator Reserach Organization

High Energy Accelerator Research Organization (KEK) Notices for KEK Report 2016-2 and its included software or data base

Use: This report and its included software or data base should be used for non-commercial purposes only.
Contact KEK regarding commercial use.

KEK disclaimer of liability: KEK makes no representations or warranties, express or implied, nor assumes any liability for the use of this report or its contents, including software and data base.

Maintenance of notices: In the interest of clarity regarding the origin and status of this report and its included software or data base, this and all the preceding KEK notices are to: (1) remain affixed to any copy or derivative of this report or its software and data base made or distributed by the recipient of this report or its software and data base; and (2) be affixed to any copy of a document or any software and data base made or distributed by the recipient that contains a copy or derivative of this report or its software and data base.

For the information on the copyright of the EGS5 Code System, please visit the URL below.
http://rcwww.kek.jp/research/egs/egs5.html

KEK Reports are available from

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

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

β-ray Spectrum Data for egs5 based on ICRU-56 or RADAR

Y. Kirihara, H. Hirayama and Y. Namito

High Energy Accelerator Reserach Organization

Contents

Japanese Parts **1**

1 はじめに **2**

2 ICRU-56 データ **2**

3 RADAR データ **2**

4 ICRU-56 データと RADAR データの比較 **2**

5 egs5 用データベース **2**
5.1 ICRU-56 データ 2
5.2 RADAR-56 データ 3

6 Sample user code **4**
6.1 `ucicru56.f` 4
6.2 `ucradar.f` 6

7 データベース及びユーザーコードのダウンロード **7**

English Parts **9**

1 Introduction **10**

2 ICRU-56 data **10**

3 RADAR data **10**

4 Comparison between ICRU-56 data and RADAR data **10**

5 β-ray data base for egs5 **11**
5.1 ICRU-56 data 11
5.2 RADAR-56 data 11

6 Sample user code **12**
6.1 `ucicru56.f` 12
6.2 `ucradar.f` 14

7 Download of the data base and sample user codes **15**

Table and figure **17**

Appendix: Full listings of `ucicrp56.f` **28**

Appendix: Main programme of `ucradar.f` **39**

Appendix: Full listings of `ucradar.f` **45**

ICRU-56又はRADARに基づく egs5用β線スペクトルデータ

(Japanese Parts)

1 はじめに

線源の作り方で示されている様に、β線源はγ線源と異なりスペクトルは連続である。連続型の過程のサンプリングでは、一般には直接サンプリングは難しい。近似的な方法であるが、スペクトルの形が与えられている場合にどの様な場合にも適用できる方法は、横軸 (この場合は、エネルギー) を等間隔に区分し、その区間の積分値の全領域の積分値に対する割合を確率密度関数とし、乱数により対応するエネルギー区間をサンプリングし、エネルギー区間内では、一様分布として直線内挿によりエネルギーを決定する方法である。このためには、核種毎のデータが必要である。線源の作り方では、ICRU Report 56[1] のデータ (以下、「ICRU-56 データ」という。) を使った Sr-90 を扱っている。他の核種の場合には、ICRU Report 56 等からデータを作成することが必要となる。本レポートでは、ICRP Report 56 に掲載されているデータ及びより多くの核種が含まれている BNL National Nuclear Data Center から公開されている「RADAR - The Decay Data」[2] を使って作成した egs5 用の β 線スペクトルデータとその使い方を紹介する。

2 ICRU-56 データ

ICRU-56 には、36 核種の β 線スペクトルデータが掲載されている。スペクトルデータは、β 線の最大エネルギーを E_{max} として、E_{max} の値と E/E_{max} を 40 等分したときの、崩壊当たりに放出される単位 E/E_{max} 当たりの β 線数で構成されている。各区分の β 線数を 40 で割った値が区分当たりの β 線数となる。

3 RADAR データ

BNL National Nuclear Data Center から多くの核種の β 線スペクトルデータが、「RADAR - The Decay Data[2]」として EXCEL の表の形で公開されている。β 線のスペクトルデータは、Health Physics に掲載された [3]429 核種と、BNL のレポート [4] に掲載された 34 核種が含まれている。文献 [3] に基づくデータは、β 線の最大エネルギーまでの区間を 20 等分し、各エネルギー領域とその領域での崩壊当たりの放出数が示されている。文献 [4] に基づくデータは、エネルギーが等間隔でなく、分点数も核種により異なる。

4 ICRU-56 データと RADAR データの比較

ICRU-56 に含まれている 36 核種について RADAR データと β 線スペクトルの比較を行った。SLAC-TN-92-1[5] で公開されている簡易式の計算プログラム (BetaCDF code) を用いてで計算した値も参考値として比較した。両者にデータがある 32 核種の比較を第 1-6 図に示す。

^{210}Bi 以外の核種については、ICRU-56 データと RADAR データは数%以内で一致している。^{210}Bi については、RADAR データと SLAC-TN-92-1 とは一致しているが、ICRU Report 56 とは明らかに異なっている。2007 年の出された ICRP-107[7] のデータの元データである JAERI 1347[6] も ICRU Report 56 と異なり、RADAR データと一致することから、RADAR データの方が正しいと思われる。

^{56}Mn, ^{95}Zr, ^{99}Mo, ^{124}Sb, ^{131}I, ^{134}Cs, ^{137}Cs, ^{140}La, ^{141}Ce 及び ^{143}Ce について、SLAC-TN-92-1 のデータが、他のデータとスペクトルの形が異なっているのは、マルチエネルギーを無視しているためである。

5 egs5 用データベース

5.1 ICRU-56 データ

ICRU-56 データを用いて、核種毎に以下の構造を持つデータファイルを作成した。

1. 1 行目　核種の説明 20 文字

2. 2 行目　電子 (-1) か陽電子 (1) の識別データ、β の最大エネルギー (E_{max})

3. 3 から 42 行目　単位 E/E_{max} エネルギー区分の崩壊当たりの放出数

^{90}Sr の場合の例を以下に示す。

```
ICRU5-56 Sr-90 Beta-
-1, 0.546
1.597
1.538
1.532
1.526
1.518
1.509
1.500
1.490
1.479
1.466
1.453
1.439
1.422
1.404
1.384
1.361
1.335
1.306
1.274
1.238
1.198
1.154
1.106
1.053
0.997
0.935
0.870
0.801
0.729
0.654
0.577
0.498
0.420
0.343
0.268
0.198
0.135
0.081
0.038
0.010
```

egs5 用のデータとしては、表 1 に示す核種が含まれている。

5.2 RADAR-56 データ

文献 [4] に基づくデータは、エネルギーが等間隔でなく、分点数も異なることから、エネルギー分点に対応するエネルギーを決めることが困難である。文献 [3] のデータが大部分であることから、文献 [3] に基づくデータのみを用いて egs5 用のデータファイルを作成した。スペクトルデータは、電子と陽電子を区別していないので、両方の崩壊モードを持つ核種は、データから除外した。

以上の条件で、核種毎に以下の構造を持つデータファイルを作成した。

1. 1 行目　核種の説明 20 文字

2. 2 行目　電子 (-1) か陽電子 (1) の識別データ

3. 3 から 22 行目　エネルギー区分の上限値と対応する崩壊当たりの放出数

^{90}Sr の場合の例を以下に示す。

```
RADAR Sr-90 Beta-
-1
0.0273,7.79E-02
0.0546,7.60E-02
0.0819,7.50E-02
0.1092,7.40E-02
0.1365,7.30E-02
0.1638,7.17E-02
0.1911,7.01E-02
0.2184,6.80E-02
0.2457,6.53E-02
0.2730,6.19E-02
0.3003,5.78E-02
0.3276,5.27E-02
0.3549,4.68E-02
0.3822,4.01E-02
0.4095,3.27E-02
0.4368,2.48E-02
0.4641,1.71E-02
0.4914,9.75E-03
0.5187,4.28E-03
0.5460,1.01E-03
```

egs5 用のデータとしては、表 2,3 に示す核種が含まれている。

6 Sample user code

6.1 ucicru56.f

ucicru56.f は、ucsource.f の枠組みで egs5 用の ICRU-56 データを使用したユーザーコードである。ucicru56.f では、使用する β 線を放出する核種を、キーボードから入力するようにしている。ICRU-56 データを含むディレクトリーは、egs5run を実行しているディレクトリーにあることを前提としている。入力する核種名は、第 1 表にある表記方法である。例えば、^{90}Sr の場合には、Sr-90 と入力する。

ICRU-56 データを使用することに関連した箇所は、以下の箇所である。

1. 変数の定義

```
      real*8                                                ! Local variables
     * availke,tnum,wtin,wtsum,xi0,yi0,zi0,ebmax,
     * spe(MXEBIN),ebeta(41),pbeta(41),cbeta(41),
     * emax

      integer
     * i,icases,idin,ie,ifti,ifto,ii,j,k,n,ner,nbtype,nbnum

      character*10 atom
      character*72 soinf
      character*72 filename
```

ebeta は、データベースの区分の上限運動エネルギー、pbeta は、データベースの区分ごとの崩壊当たりの放出数で、cbeta は、累積分布関数である。nbtype は、電子 (-1) と陽電子 (1) かを識別する変数、nbnum は、分点数 41 であり、41 を用いる。また、atom は、使用する核種名、soinf は、データベースの 1 行目に書かれている線源情報、filename は、入力した核種名を使って作成するデータファイル名である。

2. データファイルの open

```
      write(6,'(A/A,A)')
     *   ' Key in atomic number and mass number like Sr-90'

      read(5,*) atom
```

```
      filename='ICRU56/'//trim(adjustl(atom))//'.dat'

      open(3,file=filename,STATUS='old')
```

3. データの読み込み

```
! Read beta-ray spectrum data fron ICRU-56 data-base
      read(3,'(A72)') soinf
      read(3,*) nbtype,ebmax
      nbnum=41
      ebeta(1)=0.d0
      do i=2,nbnum
        read(3,*) pbeta(i)
      end do
```

エネルギー分点数を 41 とし、エネルギーと確率分布関数を 2 から nbnum までのデータとして読み込んでいるのは、ファイルのエネルギー値はエネルギービンの上限であるためである。

4. 確率密度関数と累積分布関数の算出

```
!-------------------------------------------
!     Calculate CDF and PDF from emission rate
!-------------------------------------------
      tnum=0.d0
      ebeta(1)=0.d0
      do ie=2,nbnum
        tnum=tnum+pbeta(ie)
        ebeta(ie)=ebmax*(ie-1)/40.d0
      end do

      cbeta(1)=0.d0
      pbeta(1)=0.d0
      do ie=2,nbnum
        cbeta(ie)=cbeta(ie-1)+pbeta(ie)/tnum
        pbeta(ie)=pbeta(ie)/tnum  ! pdf
      end do
      tnum=tnum/40.d0  ! Number of beta-rays per decay

      iqin=nbtype         ! Incident charge - electrons
      ekein=ebeta(nbnum)  ! Maximum kinetic energy
```

ビン毎の放出率から、確率密度関数と累積分布関数求める。その後、読み込んだデータに基づき入射粒子の電荷と最大運動エネルギーを設定する。

5. エネルギーのサンプリング

```
!       ------------------------
!       Determine source energy
!       ------------------------
        call randomset(rnnow)
        do ie=2,nbnum
          if(rnnow.le.cbeta(ie)) go to 1000
        end do
1000    if(ie.gt.nbnum) ie=nbnum
        ekein=ebeta(ie-1)+(rnnow-cbeta(ie-1))*(ebeta(ie)-ebeta(ie-1))
     *      /(cbeta(ie)-cbeta(ie-1))
```

確率分布関数は、合計すると 1 になるはずであるが、数値表現の精度のため、厳密に 1 にならない場合がある。ie が 41 を超えないようにするために、超えた場合には、nbnum (=41) に制限している。

β 線のエネルギーサンプリングに直接関係するのは以上である。結果を入射 β 線当たりとする場合には、ヒストリー全体での平均を求めればよい。一方、線源強度が $\mathrm{Bq/cm^2}$ 又は $\mathrm{Bq/cm^3}$ の場合で、単位放射能当たり ($\mathrm{Bq/cm^2}$ 又は $\mathrm{Bq/cm^3}$) の量を計算する場合には、入射 β 線当たりの結果に tnum を掛ける必要がある。

Appendix に、`ucicru56.f` を示す。

6.2 ucradar.f

ucradar.f は、ucsource.f の枠組みで egs5 用の RADAR データを使用したユーザーコードである。ucicru56.f と同様に、ucradar.f では、使用する β 線を放出する核種を、キーボードから入力するようにしている。RADAR データを含むディレクトリーは、egs5run を実行しているディレクトリーにあることを前提としている。入力する核種名は、第 2 又は 3 表にある表記方法である。例えば、^{90}Sr の場合には、Sr-90 と入力する。

RADAR データを使用することに関連した箇所は、以下の箇所である。

1. 変数の定義

```
      real*8                                                 ! Local variables
     * availke,tnum,wtin,wtsum,xi0,yi0,zi0,esbin(MXEBIN),
     * spe(MXEBIN),ebeta(21),pbeta(21),cbeta(21),
     * emax

      integer
     * i,icases,idin,ie,ifti,ifto,ii,j,k,n,ner,nbtype,nbnum

      character*10 atom
      character*72 soinf
      character*72 filename
```

ebeta は、データ区分の上限運動エネルギー、pbeta は、データベースの区分ごとの崩壊当たりの放出数で、cbeta は、累積分布関数である。nbtype は、電子 (-1) と陽電子 (1) かを識別する変数、nbnum は、分点数 21 である。また、atom は、使用する核種名、soinf は、データベースの 1 行目に書かれている線源情報、filename は、入力した核種名を使って作成するデータファイル名である。

2. データファイルの open

```
      write(6,'(A/A,A)')
     *   ' Key in atomic number and mass number like Sr-90'

      read(5,*) atom

      filename='RADAR/'//trim(adjustl(atom))//'.dat'

      open(3,file=filename,STATUS='old')
```

3. データの読み込み

```
! Read beta-ray spectrum data fron RADAR data-base
      read(3,'(A72)') soinf
      read(3,*) nbtype
      nbnum=21
      ebeta(1)=0.d0
      do i=2,nbnum
        read(3,*) ebeta(i),pbeta(i)
      end do
```

エネルギー分点数を 21 とし、エネルギーと確率分布関数を 2 から nbnum までのデータとして読み込んでいるのは、ファイルのエネルギー値はエネルギービンの上限であるためである。

4. 確率密度関数と累積分布関数の算出

```
!-----------------------------------------------
!     Calculate CDF and PDF from emission rate
!-----------------------------------------------
      tnum=0.d0
      do ie=2,nbnum
```

```
      tnum=tnum+pbeta(ie)
    end do
    cbeta(1)=0.d0
    do ie=2,nbnum
      cbeta(ie)=cbeta(ie-1)+pbeta(ie)/tnum
    end do

    iqin=nbtype          ! Incident charge - electrons
    ekein=ebeta(nbnum)   ! Maximum kinetic energy
```

ビン毎の放出率から、確率密度関数と累積分布関数求める。その後、読み込んだデータに基づき入射粒子の電荷と最大運動エネルギーを設定する。

5. エネルギーのサンプリング

```
!       ------------------------
!       Determine source energy
!       ------------------------
        call randomset(rnnow)
        do ie=2,nbnum
          if(rnnow.le.cbeta(ie)) go to 1000
        end do
1000    if(ie.gt.nbnum) ie=nbnum
        ekein=ebeta(ie-1)+(rnnow-cbeta(ie-1))*(ebeta(ie)-ebeta(ie-1))
     *      /(cbeta(ie)-cbeta(ie-1))
```

確率分布関数は、合計すると 1 になるはずであるが、数値表現の精度のため、厳密に 1 にならない場合がある。ie が 21 を超えないようにするために、超えた場合には、nbnum (=21) に制限している。

β 線のエネルギーサンプリングに直接関係するのは以上である。結果を入射 β 線当たりとする場合には、ヒストリー全体での平均を求めればよい。一方、線源強度が $\mathrm{Bq/cm^2}$ 又は $\mathrm{Bq/cm^3}$ の場合で、単位放射能当たり ($\mathrm{Bq/cm^2}$ 又は $\mathrm{Bq/cm^3}$) の量を計算する場合には、入射 β 線当たりの結果に tnum を掛ける必要がある。

`subroutine ausgab` 以降は、`ucicrp56.f` と同じであるので、Appendix に、`ucradar.f` のメインプログラムを示す。

7 データベース及びユーザーコードのダウンロード

上記に紹介した egs5 用の ICRU-56 と RADAR データベース及びそれぞれのサンプルユーザーコードは、`rcwww.kek.jp/research/egs/kek/egs5/beta_ray/` からダウンロードできる。

References

[1] "Dosimetry of External Beta Rays for Radiation Protection", ICRU Report 56.

[2] <http://www.doseinfo-radar.com/RADARDecay.html> (2016.4.1 final confirmation)

[3] K. F. Eckerman, R. J. Westfall, J. C. Ryman, and M. Cristy, "Availability of Nuclear Decay Data in Electronic Form, Including Beta Spectra not Previously Published", Health Phys. **67(4)**(1994)338-345.

[4] T.W. Burrows, "The Program RADLST", Brookhaven National Laboratory Report BNL-NCS-52142 (1988).

[5] W. R. Nelson and J. Liu, "SAMPLING THE FERMI DISTRIBUTION FOR β-DECAY ENERGY INPUT TO EGS4",Stanford Linear Accelerator Center Report SLAC-TN92-1, June 1992, December 1997(Rev).

[6] "Nuclear Decay Data for Dosimetric Calculations",ICRP Publication 107, Annals of ICRP, 38(2008).

[7] A. Endo, Y. Yamaguchi and K. F. Eckerman, "Nuclear Decay Data for Dosimetry Calculation Revised Data of ICRP Publication 38", JAERI 1347, Japan Atomic Energy Research Institute (2004).

β-ray Spectrum Data for egs5 based on ICRU-56 or RADAR

(English Parts)

1 Introduction

β-sources have a different continuous spectrum than γ-source as mentioned in the "Lecture note of Practices on How to Write Source Routine".

In general, it is difficult to apply a direct sampling method for continuous distribution. The approximation method appllicable to any case is to use a robability distribution function for a segmented interval in β-ray energy if a spectrum is known . The energy interval can be sampled using the cumulative distribution function with a random number. The β-ray energy in each energy bin is sampled assuming a uniform distribution inside the energy bin. The spectrum data is necessary for each radionuclide to apply this method. In the above-mentioned lecture note, spectrum data for ^{90}Sr in ICRU Report 56[1] (call "ICRU-56 data") is used. For other radionuclides, it is necessary to find the spectra data for each time.

In this lecture note, β-ray spectra data for egs5 and an explanation of how to use these in egs5 are presented. The β-ray spectra data, ICRU-56 data, and "RADAR - The Decay Data[2]" have been made open to public by the Brookhaven National Laboratory (BNL) National Nuclear Data Center (the latter includes more radionuclides than the ICRU-56 data).

2 ICRU-56 data

In ICRU-56, β-ray spectrum data for 36 radionuclides are presented. The energy bin width is $E_{max}/40$ and is expressed in E/E_{max}, where E/E_{max} is a maximum β-ray energy. Each spectrum data includes E_{max} and β-ray emission rate per decay per unit E/E_{max}. From this expression, the number of β-rays per decay per bin can be calculated from the data provided by dividing by 40.

3 RADAR data

From the BNL National Nuclear Data Center, β-ray spectrum data for many radionuclides are presented as "RADAR - The Decay Data[2]". β-ray spectra data for 429 radionuclides were cited from the paper by K. F. Eckerman et al.[3] and those for 34 others were cited from the BNL report[4] by T. W. Burrown. Data based on [3] are presented in the β-ray emission rate per energy bin, which is divided in equal 20 bins. The number of energy bins based on [4] is not fixed and the widths are not equal.

4 Comparison between ICRU-56 data and RADAR data

Comparisons between ICRU-56 data and RADAR data of 36 radionuclides, which are included in both data were performed and presented in Figure 1. In this figure, calculated spectra using a BetaCDF code as presented in SLAC-TN-92-1[5] are also shown as reference data.

Except for ^{210}Bi, ICRU-56 data and RADAR data agrees well within a few percentages. For ^{210}Bi, the RADAR data agrees well with the result calculated by the BetaCDF code. The data in JAERI 1347[6], which present β-ray spectra data for ICRP-107[7], also agree well with the RADAR data. From theses facts, the RADAR data will be assumed correct in favor of ICRU-56 data, for ^{210}Bi.

Owing to the fact that the BetaCDF code ignores multi energy β-ray emissions, the spectra of ^{56}Mn, ^{95}Zr, ^{99}Mo, ^{124}Sb, ^{131}I, ^{134}Cs, ^{137}Cs, ^{140}La, ^{141}Ce and ^{143}Ce are different than those provided by the ICRU-56 and RADAR data.

5 β-ray data base for egs5

5.1 ICRU-56 data

A data file for egs5 based on ICRU-56 data was made for each raionuclide.

1. First line : Explanation of radionuclide etc. within 20 characters
2. Second line : Type of β-ray, electron (-1) and positron (1) and a maximum energy E_{max}
3. From third line to the 42-nd line : β-ray emission rate per unit E/E_{max} interval and decay

The following is an example for ^{90}Sr.

```
ICRU5-56 Sr-90 Beta-
-1, 0.546
1.597
1.538
1.532
1.526
1.518
1.509
1.500
1.490
1.479
1.466
1.453
1.439
1.422
1.404
1.384
1.361
1.335
1.306
1.274
1.238
1.198
1.154
1.106
1.053
0.997
0.935
0.870
0.801
0.729
0.654
0.577
0.498
0.420
0.343
0.268
0.198
0.135
0.081
0.038
0.010
```

The ICRU-56 data for egs5 includes radionuclides, as shown in Table 1.

5.2 RADAR-56 data

It is difficult to find an energy corresponding to the energy bin in the case of data based on [4] because it presents the data for central enery and not for equal interval. Considering that most of the data in RADAR data is based on [3], a database for egs5 was made using the data in [3]. Some radionuclides have a decay mode in which they emit both electron and positron. These radionuclides are excluded to avoid confusion in egs5 calculation.

According to the condition mentioned above, a data file with the following properties was made for each radionuclide.

1. First line : Explanation of radionuclide etc. within 20 characters
2. Second line : Type of β-ray, electron (-1) and positron (1)
3. From third lines to 22-nd line : Energy corresponding to the upper bin and β-ray emission rate per bin per decay

Following is an example for ^{90}Sr.

```
RADAR Sr-90 Beta-
-1
0.0273,7.79E-02
0.0546,7.60E-02
0.0819,7.50E-02
0.1092,7.40E-02
0.1365,7.30E-02
0.1638,7.17E-02
0.1911,7.01E-02
0.2184,6.80E-02
0.2457,6.53E-02
0.2730,6.19E-02
0.3003,5.78E-02
0.3276,5.27E-02
0.3549,4.68E-02
0.3822,4.01E-02
0.4095,3.27E-02
0.4368,2.48E-02
0.4641,1.71E-02
0.4914,9.75E-03
0.5187,4.28E-03
0.5460,1.01E-03
```

The RADAR data for egs5 includes radionuclides that are shown in Table 2 and 3.

6 Sample user code

6.1 ucicru56.f

`ucicru56.f` is the user code to use ICRU-56 data in the same framework with `ucsource.f`. It is assumed that a directory including ICRU-56 data is existing in the directory of the running egs5run. The radionuclide that is to be used is chosen with the keyboard from an expression in Table 1. For example, type in `Sr-90` for ^{90}Sr.

The following are statements used for the ICRU-56 data.

1. Definition of variables

```
 real*8                                                ! Local variables
* availke,tnum,wtin,wtsum,xi0,yi0,zi0,ebmax,
* spe(MXEBIN),ebeta(41),pbeta(41),cbeta(41),
* emax

 integer
* i,icases,idin,ie,ifti,ifto,ii,j,k,n,ner,nbtype,nbnum

 character*10 atom
 character*72 soinf
 character*72 filename
```

`ebmax` is the maximum kinetic energy of β-ray, `pbeta` is an emission rate per unit bin per decay, `cbeta` is the cumulative distribution function. `nbtype` is the charge of β-ray; -1 for electron and 1 for positron. `nbnum` is a bin number (=41). `atom` is the name of radionuclide and `soinf` is the explanation of radionuclide etc. within 20 characters. `filename` is the full name of the data file of radionuclide to use,

2. Open data file

```
      write(6,'(A/A,A)')
     *   ' Type in atomic number and mass number like Sr-90'

      read(5,*) atom

      filename='ICRU56/'//trim(adjustl(atom))//'.dat'

      open(3,file=filename,STATUS='old')
```

3. Read data

```
! Read beta-ray spectrum data from ICRU-56 data-base
      read(3,'(A72)') soinf
      read(3,*) nbtype,ebmax
      nbnum=41
      ebeta(1)=0.d0
      do i=2,nbnum
        read(3,*) pbeta(i)
      end do
```

The normalized energy bin corresponds to the upper bin energy. Energy and emission rate are assigned from 2 to nbnum for this procedure.

4. Calculate the probability density function and cumulative distribution function

```
!-----------------------------------------------
!     Calculate CDF and PDF from emission rate
!-----------------------------------------------
      tnum=0.d0
      ebeta(1)=0.d0
      do ie=2,nbnum
        tnum=tnum+pbeta(ie)
        ebeta(ie)=ebmax*(ie-1)/40.d0
      end do

      cbeta(1)=0.d0
      pbeta(1)=0.d0
      do ie=2,nbnum
        cbeta(ie)=cbeta(ie-1)+pbeta(ie)/tnum
        pbeta(ie)=pbeta(ie)/tnum  ! pdf
      end do
      tnum=tnum/40.d0  ! Number of beta-rays per decay

      iqin=nbtype          ! Incident charge - electrons
      ekein=ebeta(nbnum)   ! Maximum kinetic energy
```

Calculate the probability density function and cumulative distribution function from the emission rate. Set a charge of the source particle and a maximum kinetic energy from the read data.

5. Sampling the kinetic energy of of a β-ray

```
!       ------------------------
!       Determine source energy
!       ------------------------
        call randomset(rnnow)
        do ie=2,nbnum
          if(rnnow.le.cbeta(ie)) go to 1000
        end do
1000    if(ie.gt.nbnum) ie=nbnum
        ekein=ebeta(ie-1)+(rnnow-cbeta(ie-1))*(ebeta(ie)-ebeta(ie-1))
     *      /(cbeta(ie)-cbeta(ie-1))
```

The aboves are directly related to the sampling of β-ray kinetic energy. If a source strength is given in Bq/cm^2 or Bq/cm^3 and it is necessary to calculate values per Bq/cm^2 or Bq/cm^3, tnum must be multiplied with the results per β-ray.

A full listings of `ucicru56.f` is shown in the Appendix.

6.2 `ucradar.f`

`ucradar.f` is the user code to use RADAR data in the same framework with `ucsource.f`. It is assumed that a directory including the RADAR data is existing in the directory of the running egs5run. The radionuclide that is to be used is chosen with the keyboard from an expression in Table 2 or 3. For example, type in `Sr-90` for ^{90}Sr.

The following are statements are used for the ICRU-56 data.

1. Definition of variables

```
 real*8                                                  ! Local variables
* availke,tnum,wtin,wtsum,xi0,yi0,zi0,esbin(MXEBIN),
* spe(MXEBIN),ebeta(21),pbeta(21),cbeta(21),
* emax

 integer
* i,icases,idin,ie,ifti,ifto,ii,j,k,n,ner,nbtype,nbnum

 character*10 atom
 character*72 soinf
 character*72 filename
```

   `ebeta` is the upper bin kinetic energy, `pbeta` is the emission rate per unit bin per decay, `cbeta` is the cumulative distribution function. `nbtype` is the charge of β-ray; -1 for electron and 1 for positron. `nbnum` is an bin number and 21. `atom` is the name of radionuclide and `soinf` is the explanation of radionuclide etc. within 20 characters. `filename` is the full name of the data file of the radionuclide to use,

2. Open data file

```
 write(6,'(A/A,A)')
*   ' Type in atomic number and mass number like Sr-90'

 read(5,*) atom

 filename='RADAR/'//trim(adjustl(atom))//'.dat'

 open(3,file=filename,STATUS='old')
```

3. Read data

```
! Read beta-ray spectrum data from RADAR data-base
      read(3,'(A72)') soinf
      read(3,*) nbtype
      nbnum=21
      ebeta(1)=0.d0
      do i=2,nbnum
        read(3,*) ebeta(i),pbeta(i)
      end do
```

   The energy bin corresponds to the upper bin energy. Energy and emission rate are assigned from 2 to nbnum for this procedure.

4. Calculate the probability density function and cumulative distribution function

```
!-------------------------------------------
!     Calculate CDF and PDF from emission rate
!-------------------------------------------
      tnum=0.d0
      do ie=2,nbnum
        tnum=tnum+pbeta(ie)
      end do
      cbeta(1)=0.d0
      do ie=2,nbnum
        cbeta(ie)=cbeta(ie-1)+pbeta(ie)/tnum
      end do

      iqin=nbtype          ! Incident charge - electrons
      ekein=ebeta(nbnum)   ! Maximum kinetic energy
```

Calculate probability density function and cumulative distribution function from the emission rate. Set the charge of source particle and a maximum kinetic energy from the read data.

5. Sampling the kinetic energy of a β-ray

```
!       ------------------------
!       Determine source energy
!       ------------------------
        call randomset(rnnow)
        do ie=2,nbnum
          if(rnnow.le.cbeta(ie)) go to 1000
        end do
1000    if(ie.gt.nbnum) ie=nbnum
        ekein=ebeta(ie-1)+(rnnow-cbeta(ie-1))*(ebeta(ie)-ebeta(ie-1))
     *      /(cbeta(ie)-cbeta(ie-1))
```

The above is directly related to the sampling of β-ray kinetic energy. If a source strength is given in Bq/cm^2 or Bq/cm^3 and it is necessary to calculate values per Bq/cm^2 or Bq/cm^3, tnum must be multiplied with the results per β-ray.

Statements after the `subroutine ausgab` are the same as `ucicrp56.f`. The main program of `ucradar.f` is shown in the Appendix.

7 Download of the data base and sample user codes

The β-ray data base based on ICRU-56 or RADAR for egs5 and sample user codes mentioned above can be downloaded from the following address.

`rcwww.kek.jp/research/egs/kek/egs5/beta_ray/`

References

[1] "Dosimetry of External Beta Rays for Radiation Protection", ICRU Report 56.

[2] <http://www.doseinfo-radar.com/RADARDecay.html> (2016.4.1 final confirmation)

[3] K. F. Eckerman, R. J. Westfall, J. C. Ryman, and M. Cristy, "Availability of Nuclear Decay Data in Electronic Form, Including Beta Spectra not Previously Published", Health Phys. **67(4)**(1994)338-345.

[4] T.W. Burrows, "The Program RADLST", Brookhaven National Laboratory Report BNL-NCS-52142 (1988).

[5] W. R. Nelson and J. Liu, "SAMPLING THE FERMI DISTRIBUTION FOR β-DECAY ENERGY INPUT TO EGS4",Stanford Linear Accelerator Center Report SLAC-TN92-1, June 1992, December 1997(Rev).

[6] A. Endo, Y. Yamaguchi and K. F. Eckerman, "Nuclear Decay Data for Dosimetry Calculation Revised Data of ICRP Publication 38", JAERI 1347, Japan Atomic Energy Research Institute (2004).

[7] "Nuclear Decay Data for Dosimetric Calculations",ICRP Publication 107, Annals of ICRP, 38(2008).

Tables

Table 1: Nuclide included for egs5 based on ICRU-56

C-14	Na-24	P-32	S-35	Mn-52	Mn-56
Co-56	Fe-59	Cu-62	Sr-89	Sr-90	Y-90
Y-91	Zr-95	Nb-95	Mo-99	Ru-106	Rh-106
Sb-106	I-131	Cs-134	Cs-137	Ba-140	La-140
Ce-141	CE-143	Pr-143	Ce-144-Pr	Pm-147	Tm-170
Au-198	Tl-204	Bi-210	Pa-234m		

Table 2: Nuclide included for egs5 based on RADAR data base (1)

C-11	**C-14**						
N-13							
O-14	**O-15**	**O-19**					
F-18	**F-19**						
Ne-19							
Na-22	**Na-24**						
Mg-28							
Al-26	**Al-28**						
Si-31	**Si-32**						
P-30	**P-32**	**P-33**					
S-35							
Cl-36	**Cl-38**	**Cl-39**					
Ar-39	**Ar-41**						
K-38	**K-40**	**K-42**	**K-43**	**K-44**	**K-45**		
Ca-45	**Ca-47**	**Ca-49**					
Sc-43	**Sc-44**	**Sc-46**	**Sc-47**	**Sc-48**	**Sc-49**		
Ti-45							
V-47	**V-48**						
Cr-48	**Cr-49**						
Mn-51	**Mn-52**	**Mn-52m**	**Mn-56**				
Fe-52	**Fe-59**						
Co-55	**Co-56**	**Co-58**	**Co-60**	**Co-60m**	**Co-61**	**Co-62m**	
Ni-57	**Ni-63**	**Ni-65**	**Ni-66**				
Cu-60	**Cu-61**	**Cu-62**	**Cu-66**	**Cu-67**			
Zn-62	**Zn-65**	**Zn-71m**	**Zn-72**				
Ga-66	**Ga-68**	**Ga-70**	**Ga-72**	**Ga-73**			
Ge-66	**Ge-69**	**Ge-75**	**Ge-77**	**Ge-78**			
As-69	**As-71**	**As-72**	**As-77**	**As-78**			
Se-70	**Se-73m**	**Se-79**	**Se-81**	**Se-83**			
Br-82	**Br-83**	**Br-84**					
Kr-74	**Kr-77**	**Kr-79**	**Kr-85**	**Kr-85m**	**Kr-87**		
Rb-77	**Rb-79**	**Rb-80**	**Rb-81**	**Rb-82**	**Rb-86**	**Rb-86m**	**Rb-87**
Rb-88	**Rb-89**						
Sr-81	**Sr-89**	**Sr-90**	**Sr-91**	**Sr-92**			
Y-86	**Y-86m**	**Y-87**	**Y-88**	**Y-90**	**Y-91**	**Y-92**	**Y-93**
Y-94	**Y-95**						
Zr-89	**Zr-93**	**Zr-95**					
Nb-89	**Nb-89m**	**Nb-90**	**Nb-94**	**Nb-95**	**Nb-97**	**Nb-98**	
Mo-90	**Mo-99**	**Mo-101**					
Tc-94	**Tc-94m**	**Tc-95m**	**Tc-98**	**Tc-99**	**Tc-101**	**Tc-104**	
Ru-105	**Ru-106**						
Rh-99	**Rh-99m**	**Rh-100**	**Rh-106m**	**Rh-107**			
Pd-101	**Pd-107**	**Pd-109**					
Ag-102	**Ag-103**	**Ag-104**	**Ag-104m**	**Ag-106**	**Ag-108**	**Ag-110**	**Ag-110m**
Ag-112							
Cd-107	**Cd-113**	**Cd-113m**	**Cd-115**	**Cd-115m**	**Cd-117**	**Cd-117m**	
In-109	**In-110(69m)**	**In-110(5h)**	**In-110m**	**In-115**	**In-116**	**In-117**	**In-117m**
In-119	**In-119m**						
Sn-111	**Sn-121**	**Sn-123**	**Sn-123m**	**Sn-125**	**Sn-126**	**Sn-127**	**Sn-128**
Sb-116	**Sb-116m**	**Sb-117**	**Sb-118**	**Sb-118m**	**Sb-120**	**Sb-124**	**Sb-124m**
Sb-125	**Sb-126**	**Sb-126m**	**Sb-127**	**Sb-129**	**Sb-130**	**Sb-131**	
Te-127	**Te-127m**	**Te-129**	**Te-131**	**Te-131m**	**Te-132**	**Te-133m**	**Te-134**
I-120	**I-120m**	**I-121**	**I-122**	**I-128**	**I-129**	**I-130**	**I-131**
I-132	**I-132m**	**I-133**	**I-134**	**I-135**			

Table 3: Nuclide included for egs5 based on RADAR data base (2)

Xe-120	**Xe-121**	**Xe-123**	**Xe-125**	**Xe-133**	**Xe-135m**	**Xe-138**	
Cs-127	**Cs-130**	**Cs-134**	**Cs-135**	**Cs-136**	**Cs-137**	**Cs-138**	
Ba-139	**Ba-140**	**Ba-141**	**Ba-142**				
La-131	**La-134**	**La-138**	**La-140**	**La-141**	**La-142**	**La-143**	
Ce-143	**Ce-144**						
Pr-138	**Pr-138m**	**Pr-139**	**Pr-143**	**Pr-144**	**Pr-144m**	**Pr-145**	**Pr-147**
Nd-139	**Nd-141**	**Nd-147**	**Nd-149**	**Nd-151**			
Pm-141	**Pm-146**	**Pm-147**	**Pm-148**	**Pm-148m**	**Pm-149**	**Pm-150**	**Pm-151**
Sm-142	**Sm-151**	**Sm-153**	**Sm-155**	**Sm-156**			
Eu-145	**Eu-146**	**Eu-152m**	**Eu-154**	**Eu-155**	**Eu-156**	**Eu-157**	**Eu-158**
Gd-147							
Tb-147	**Tb-149**	**Tb-151**	**Tb-153**	**Tb-158**	**Tb-160**	**Tb-161**	
Dy-155	**Dy-165**	**Dy-166**					
Ho-155	**Ho-157**	**Ho-159**	**Ho-164**	**Ho-166**	**Ho-166m**	**Ho-167**	
Er-171	**Er-172**						
Tm-166	**Tm-170**	**Tm-171**	**Tm-172**	**Tm-173**	**Tm-175**		
Yb-167	**Yb-175**						
Lu-169	**Lu-170**	**Lu-172**	**Lu-174**	**Lu-176**	**Lu-176m**	**Lu-177**	**Lu-177m**
Lu-178	**Lu-178m**	**Lu-179**					
Hf-181	**Hf-182**	**Hf-184**					
Ta-173	**Ta-174**	**Ta-175**	**Ta-176**	**Ta-180m**	**Ta-182**	**Ta-183**	**Ta-184**
Ta-185	**Ta-186**						
W-177	**W-185**	**W-188**					
Re-177	**Re-178**	**Re-180**	**Re-182m**	**Re-186**	**Re-188**	**Re-189**	
Os-181	**Os-191**	**Os-193**	**Os-194**				
Ir-186	**Ir-194**	**Ir-195**					
Pt-197m	**Pt-199**						
Au-194	**Au-198**	**Au-199**	**Au-200**	**Au-201**			
Hg-193	**Hg-193m**	**Hg-203**	**Hg-206**				
Tl-195	**Tl-197**	**Tl-198**	**Tl-199m**	**Tl-200**	**Tl-204**	**Tl-206**	**Tl-208**
Tl-209	**Tl-210**						
Pb-209	**Pb-210**	**Pb-211**	**Pb-212**	**Pb-214**			
Bi-202	**Bi-203**	**Bi-204**	**Bi-205**	**Bi-210**	**Bi-212**	**Bi-213**	**Bi-214**
Po-203	**Po-207**						
Fr-222	**Fr-223**						
Ra-225	**Ra-227**	**Ra-228**	**Ra-232**				
Ac-228							
Th-234							
Pa-232	**Pa-233**	**Pa-234**	**Pa-234m**				
U-237	**U-239**						
Np-232	**Np-234**	**Np-236**	**Np-238**	**Np-239**	**Np-240**	**Np-240m**	
Pu-243	**Pu-246**						
Am-242	**Am-244m**	**Am-245**	**Am-246**				
Cm-249							
Bk-249	**Bk-250**						
Cf-253							

Figures

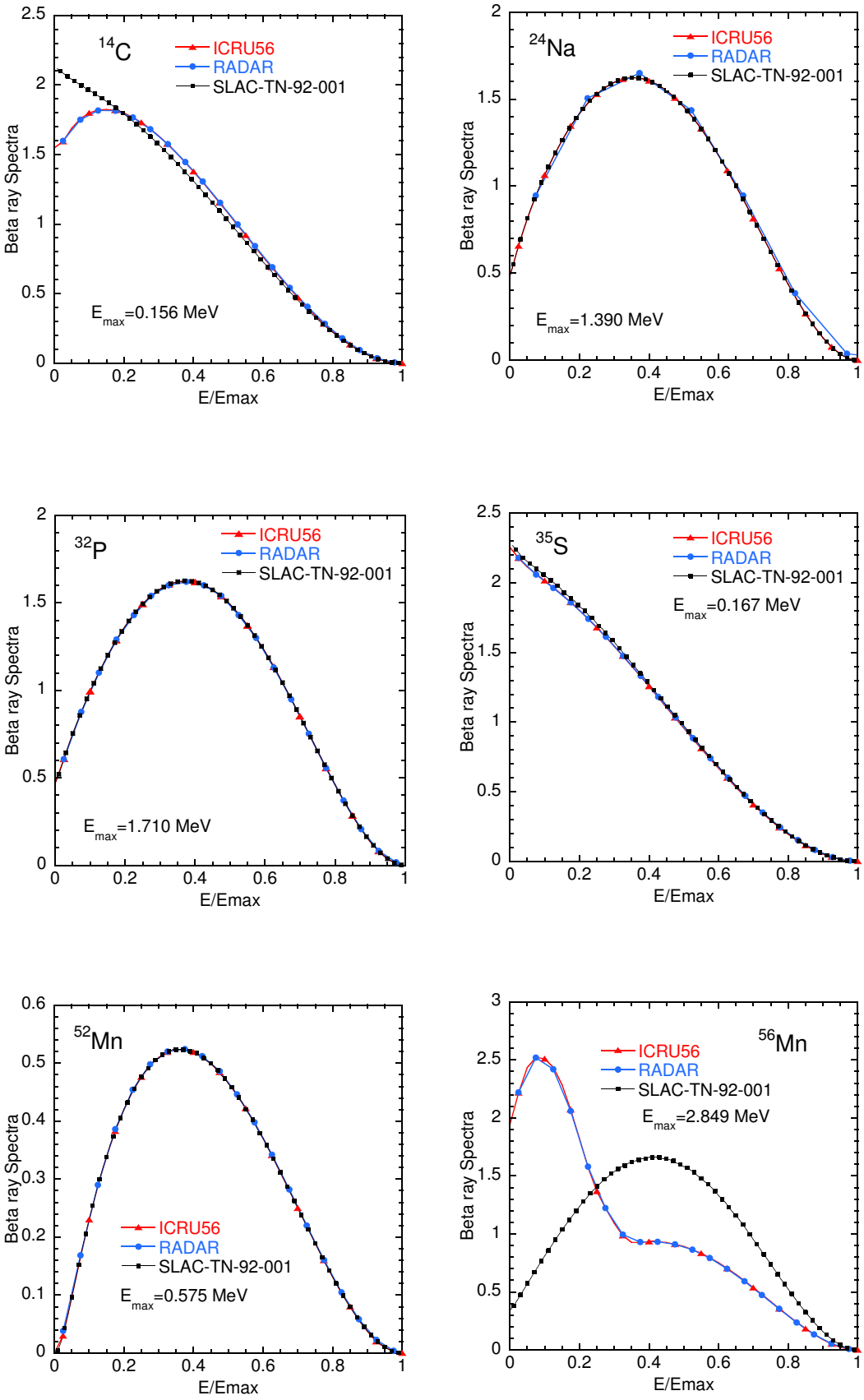

Figure 1: Comparison of β-ray spectrum between ICRU Report 56, RADAR and SLAC-TN-92-1(1).

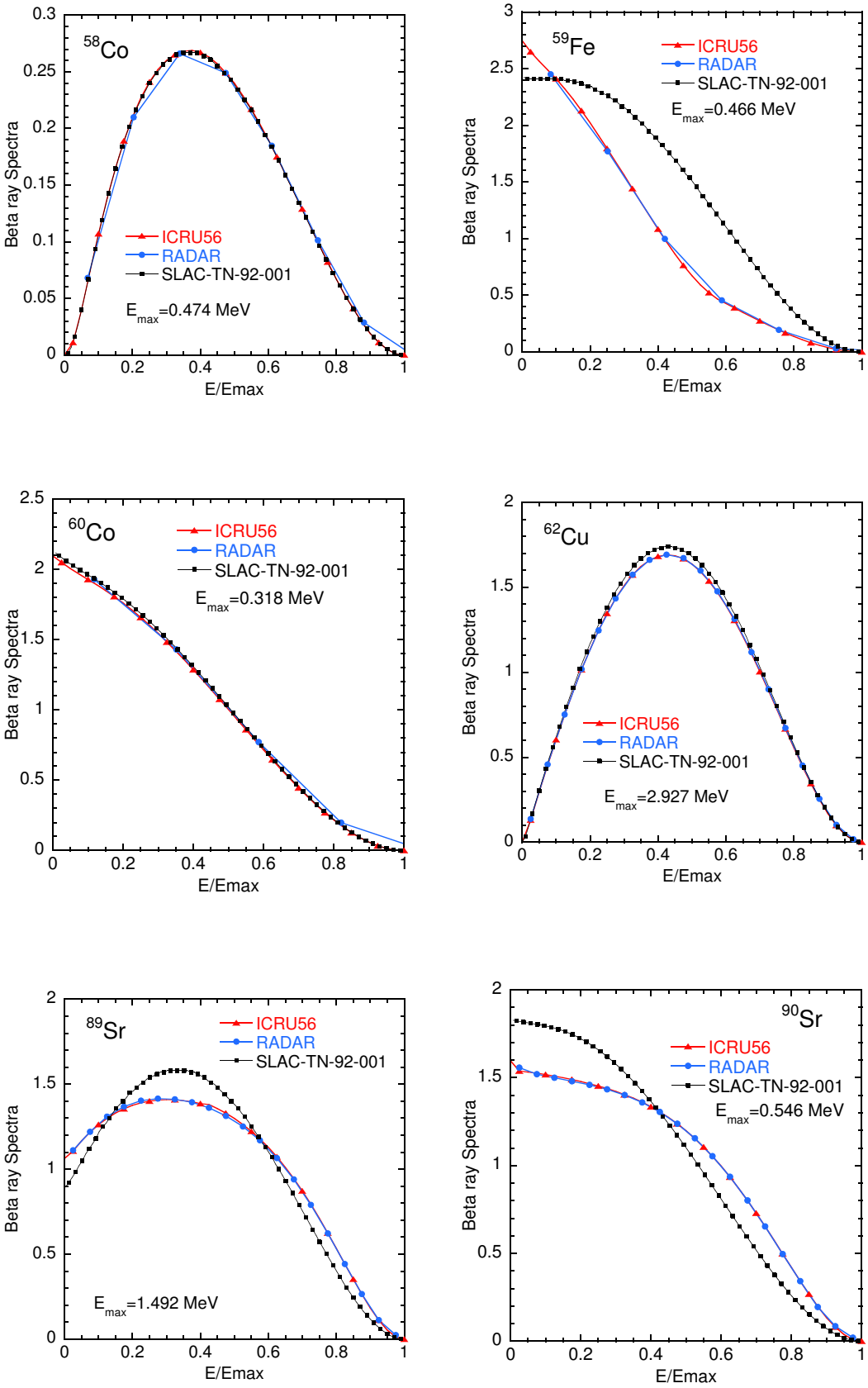

Figure 2: Comparison of β-ray spectrum between ICRU Report 56, RADAR and SLAC-TN-92-1(2).

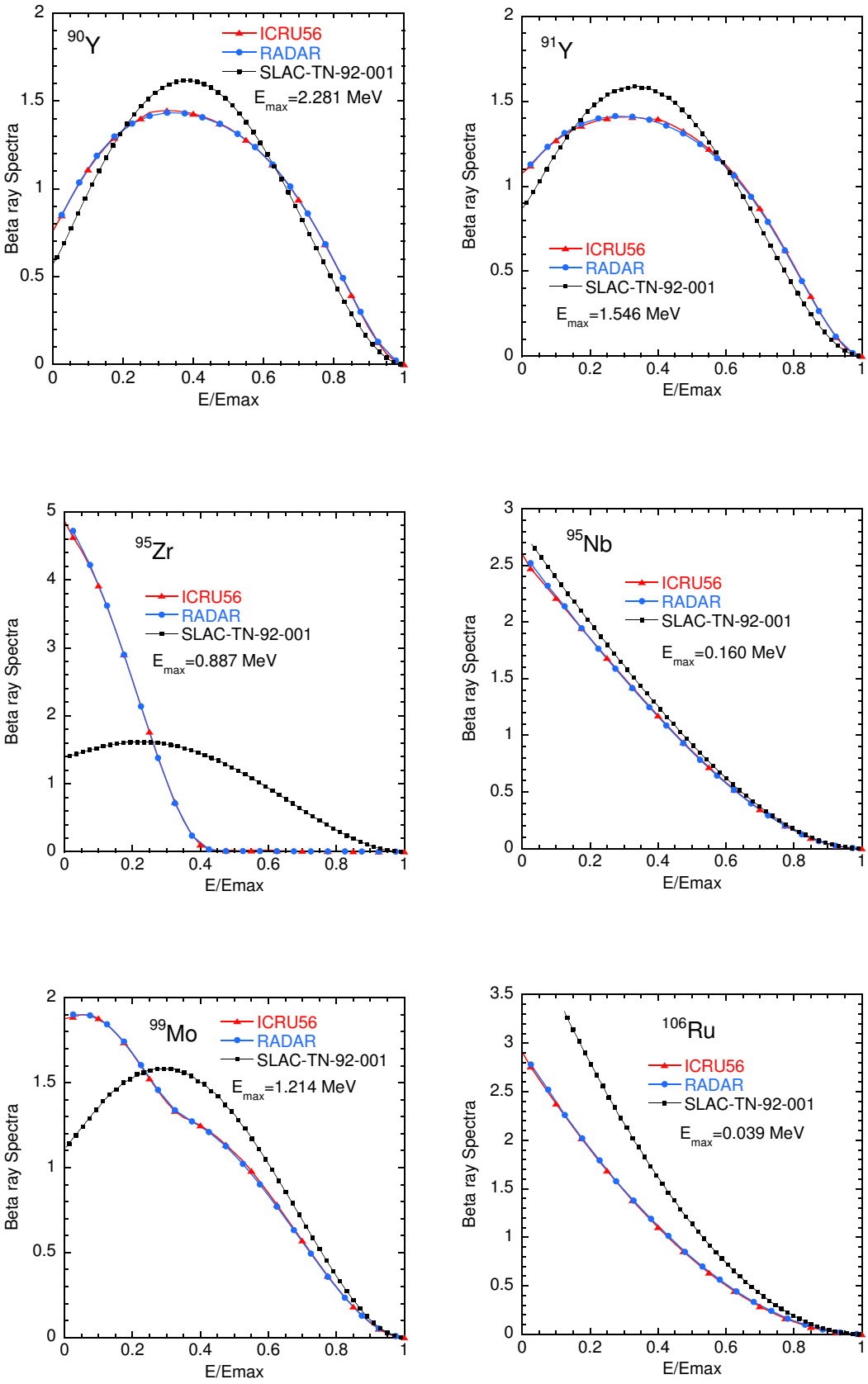

Figure 3: Comparison of β-ray spectrum between ICRU Report 56, RADAR and SLAC-TN-92-1(3).

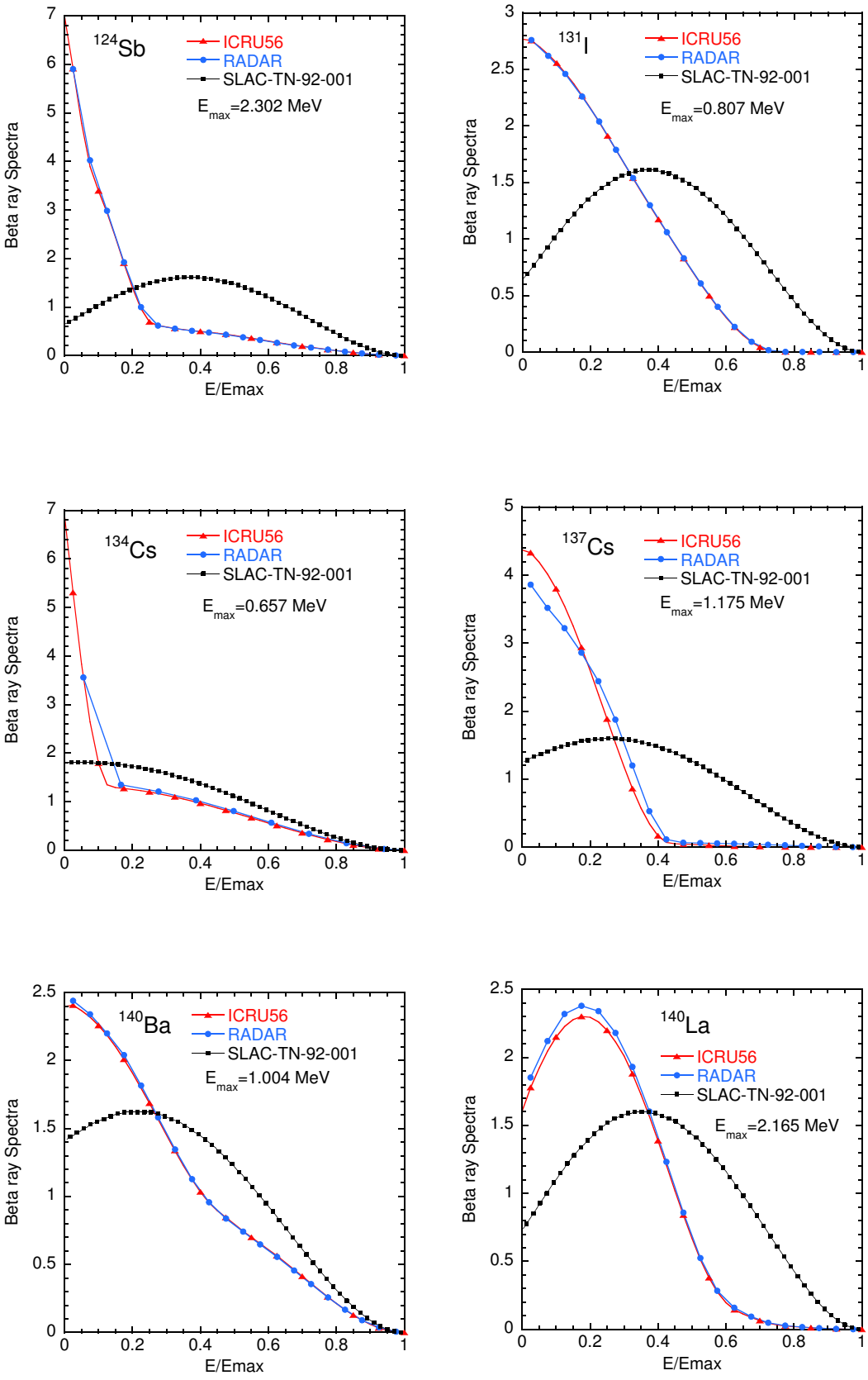

Figure 4: Comparison of β-ray spectrum between ICRU Report 56, RADAR and SLAC-TN-92-1(4).

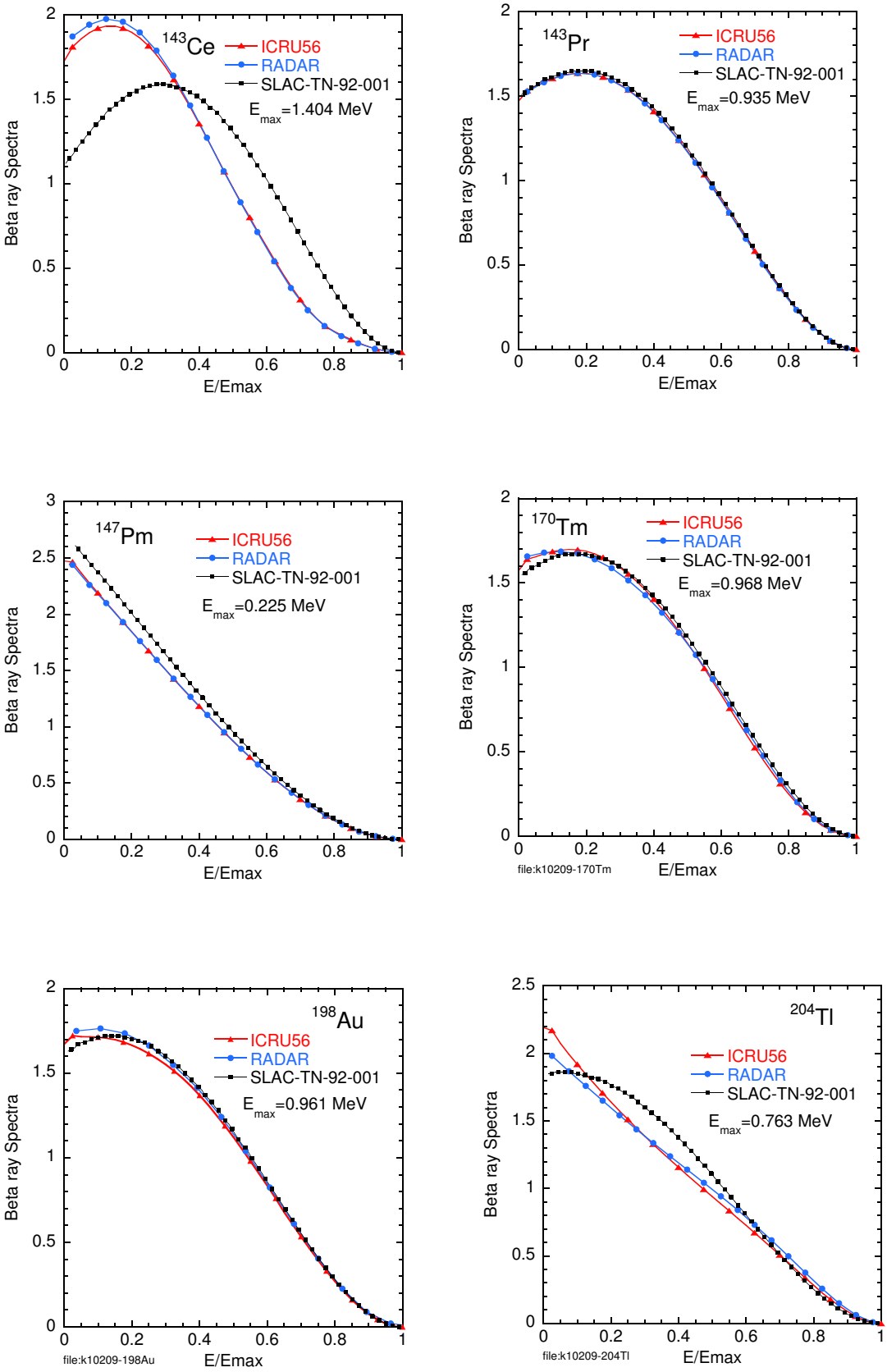

Figure 5: Comparison of β-ray spectrum between ICRU Report 56, RADAR and SLAC-TN-92-1(5).

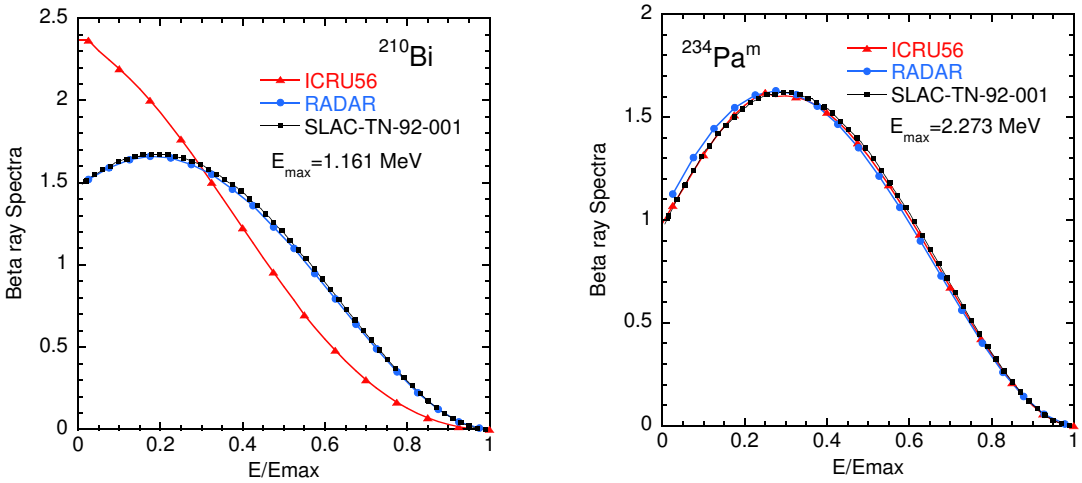

Figure 6: Comparison of β-ray spectrum between ICRU Report 56, RADAR and SLAC-TN-92-1(6).

Appendix: Full listings of `ucicrp56.f`

```
!*********************************************************************
!****************************  KEK, High Energy Accelerator Research  *
!****************************  Organization                           *
!***** u c i c r u 5 6   *****                                        *
!****************************    EGS5.0 USER CODE -  03 Apr 2016      *
!* 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 ucicru56.f User Code requires a cg-input file only              *
! (e.g., ucicru56r.data).                                             *
! The following shows the geometry for ucicru56.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 is sample user code to use RADAR beta-ray data.      *
! Use Ranlux random number generator.                                 *
!*********************************************************************
!                                                                     *
!             ------------------------                                *
!             cg Geometry (ucsource)                                  *
!             ------------------------                                *
!                                                                     *
!                                                                     *
!                      R                                              *
!                      ^                                              *
!                      |                                              *
!           +----+----+----+--------+------+---                       *
!           |         |                                              *
!           |    Outer| vacuum region                                *
!           +    +----+----+--------+------+--- r=6.0 cmm            *
!           |    |         Vacuum          |                         *
!           |    +    +----+--------+------+--- R=4.0 cm             *
!           |    |    |   Vacuum            |                        *
!           +    +    +    +--------+       +--- R=2.0 cm           *
!           |    |    |    |        |       |                        *
!           |    |    |    | Vacuum |Vacuum|                         *
! 1.253 MeV |    |    |    |        |       |                         *
!      =========>+----+----+--------+-------+--------> Z              *
!     photons  -5.0  0.0  3.0      5.0     7.0 cm                     *
!                                                                     *
!                                                                     *
!                                                                     *
!                                                                     *
!*********************************************************************
!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/                          ! Variables to score
     * maxpict
      integer maxpict

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

      real*8                                              ! Local variables
     * availke,tnum,wtin,wtsum,xi0,yi0,zi0,ebmax,
     * spe(MXEBIN),ebeta(41),pbeta(41),cbeta(41),
     * emax

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

      integer
     * i,icases,idin,ie,ifti,ifto,ii,j,k,n,ner,nbtype,nbnum

      character*10 atom
      character*72 soinf
      character*72 filename

      character*24 medarr(1)

!     ----------
!     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 call pegs5. Unit for pict must be 39.
!-----------------------------------------------------------------

      write(6,'(A/A,A)')
     *   ' Key in atomic name and mass number like Sr-90'

      read(5,*) atom

      filename='ICRU56/'//trim(adjustl(atom))//'.dat'

      open(3,file=filename,STATUS='old')

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

!     ---------------------------------
!     Define media before calling PEGS5
!     ---------------------------------

      nmed=1
      medarr(1)='NAI                          '

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

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

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

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

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

!-----------------------------------------------------------------------
! Step 3: Pre-hatch-call-initialization
!-----------------------------------------------------------------------
!-----------------------------------------------
!     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,'(A)') 'CEND'

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

!   Read material for each refion from egs5job.data
      read(4,'(15I5)') (med(i),i=1,nreg)

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

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

!-----------------------------------------------------------------------
! Step 4:  Determination-of-incident-particle-parameters
!-----------------------------------------------------------------------
! Read beta-ray spectrum data fron ICRU-56 data-base
      read(3,'(A72)') soinf
```

```
      read(3,*) nbtype,ebmax
      nbnum=41
      ebeta(1)=0.d0
      do i=2,nbnum
        read(3,*) pbeta(i)
      end do

!-----------------------------------------
!     Calculate CDF and PDF from emission rate
!-----------------------------------------
      tnum=0.d0
      ebeta(1)=0.d0
      do ie=2,nbnum
        tnum=tnum+pbeta(ie)
        ebeta(ie)=ebmax*(ie-1)/40.d0
      end do

      cbeta(1)=0.d0
      pbeta(1)=0.d0
      do ie=2,nbnum
        cbeta(ie)=cbeta(ie-1)+pbeta(ie)/tnum
        pbeta(ie)=pbeta(ie)/tnum  ! pdf
      end do
      tnum=tnum/40.d0  ! Number of beta-rays per decay

      iqin=nbtype          ! Incident charge - electrons
      ekein=ebeta(nbnum)   ! Maximum 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

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

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

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

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

      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.

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

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

!     Zero the variables
      do j=1,nbnum
        spe(j)=0.D0
      end do

!     Set histories and histories to write trajectories
      ncases=100000
!     Set maximum number for pict
      maxpict=500

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

!-----------------------------------------------------------------------
! Step 8:  Shower-call
!-----------------------------------------------------------------------
!     Write batch number
      write(39,fmt="('0    1')")
                                          ! -------------------------
      do i=1,ncases                       ! Start of batch -loop
                                          ! -------------------------

        wtin = 1.0

        wtsum = wtsum + wtin              ! Keep running sum of weights

!       ------------------------
!       Determine source energy
!       ------------------------
        call randomset(rnnow)
        do ie=2,nbnum
          if(rnnow.le.cbeta(ie)) go to 1000
        end do
1000    if(ie.gt.nbnum) ie=nbnum
        ekein=ebeta(ie-1)+(rnnow-cbeta(ie-1))*(ebeta(ie)-ebeta(ie-1))
     *      /(cbeta(ie)-cbeta(ie-1))
        spe(ie)=spe(ie)+1.0

        etot = ekein + 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

!       ---------------------------
!       Determine source  direction
!       ---------------------------

!       ------------------------
!       Determine source position
!       ------------------------

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

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

!       -----------------------------------------------------
!       Verify the normarization of source direction cosines
!       -----------------------------------------------------
```

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

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

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

                                                ! ------------------
      end do                                    ! End of batch loop
                                                ! ------------------

      call plotxyz(99,0,0,0.D0,0.D0,0.D0,0.D0,0,0.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,'(A,G15.5)') ' Elapsed Time (sec)=',cputime

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

!     ------------------------------------------------------------
!     Source spectrum.  Incident particle spectrum to detector.
!     ------------------------------------------------------------
      write(6,'(/A,A72)') ' Result for ',soinf

      write(6,'(/A/30X,A/A,11X,A,11X,A)')' Sampled source spectrum',
     *    'particles/source',' Upper energy',' Electron',' pdf'

      do ie=2,nbnum

        spe(ie)=spe(ie)/ncount

        write(6,'(G10.5,A,8X,G12.5,8X,G12.5)')
     *     ebeta(ie),' MeV--',spe(ie),pbeta(ie)
      end do

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

      stop

      end

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

!------------------------------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 AUSGAB to: produce trajectory data for 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/lines.f'        ! Auxiliary-code COMMONs

      common/totals/                           ! Variables to score
     * maxpict
      integer maxpict

      integer                                                ! Arguments
     * iarg

      real*8                                              ! Local variables
     * edepwt

      integer
     * ie,iql,irl

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

!     -----------------------------------
!     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,itvlfg,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 atvaltmp
      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'
         call srzone(xidd,yidd,zidd,iq_np,ir_np,irnext)
c
         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
c
           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'
             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
         itvlfg=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
          go to 410
  420     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
            itvlfg=1
            goto 425
          end if
        end do
  425   continue
        if(itvlfg.eq.0) then
          tval0=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----------------------
```

Appendix: Main programme of `ucradar.f`

```
!*********************************************************************
!****************************  KEK, High Energy Accelerator Research  *
!****************************  Organization                           *
!******* u c r a d a r   *****                                        *
!****************************    EGS5.0 USER CODE -  03 Apr 2016      *
!* 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 ucradar.f User Code requires a cg-input file only               *
! (e.g., ucradar.data).                                               *
! The following shows the geometry for ucradar.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 is sample user code to use RADAR beta-ray data.      *
! Use Ranlux random number generator.                                 *
!*********************************************************************
!                                                                     *
!              ------------------------                               *
!               cg Geometry (ucradar)                                 *
!              ------------------------                               *
!                                                                     *
!                                                                     *
!                      R                                              *
!                      ^                                              *
!                      |                                              *
!            +----+----+----+--------+------+---                      *
!            |         |                                             *
!            |    Outer| vacuum region                               *
!            +    +----+----+--------+------+--- r=6.0 cmm           *
!            |    |         Vacuum          |                        *
!            |    +    +----+--------+------+--- R=4.0 cm            *
!            |    |    |   Vacuum            |                       *
!            +    +    +    +--------+      +--- R=2.0 cm            *
!            |    |    |    |        |      |                        *
!            |    |    |    | Vacuum |Vacuum|                        *
! 1.253 MeV  |    |    |    |        |      |                        *
!      =========>+----+----+--------+------+--------> Z               *
!     photons  -5.0  0.0  3.0      5.0    7.0 cm                     *
!                                                                     *
!                                                                     *
!                                                                     *
!                                                                     *
!*********************************************************************
!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/                          ! Variables to score
     * maxpict
      integer maxpict

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

      real*8                                              ! Local variables
     * availke,tnum,wtin,wtsum,xi0,yi0,zi0,
     * spe(MXEBIN),ebeta(21),pbeta(21),cbeta(21),
     * emax

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

      integer
     * i,icases,idin,ie,ifti,ifto,ii,j,k,n,ner,nbtype,nbnum

      character*10 atom
      character*72 soinf
      character*72 filename

      character*24 medarr(1)

!     ----------
!     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 call pegs5. Unit for pict must be 39.
!-----------------------------------------------------------------

      write(6,'(A/A,A)')
     *   ' Key in atomic name and mass number like Sr-90'

      read(5,*) atom

      filename='RADAR/'//trim(adjustl(atom))//'.dat'

      open(3,file=filename,STATUS='old')

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

!     ---------------------------------
!     Define media before calling PEGS5
!     ---------------------------------

      nmed=1
      medarr(1)='NAI                          '

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

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

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

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

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

!-----------------------------------------------------------------------
! Step 3: Pre-hatch-call-initialization
!-----------------------------------------------------------------------
!-----------------------------------------------
!     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,'(A)') 'CEND'

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

!   Read material for each refion from egs5job.data
      read(4,'(15I5)') (med(i),i=1,nreg)

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

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

!-----------------------------------------------------------------------
! Step 4:  Determination-of-incident-particle-parameters
!-----------------------------------------------------------------------
! Read beta-ray spectrum data fron RADAR data-base
      read(3,'(A72)') soinf
```

```
      read(3,*) nbtype
      nbnum=21
      ebeta(1)=0.d0
      do i=2,nbnum
        read(3,*) ebeta(i),pbeta(i)
      end do

!-------------------------------------------
!     Calculate CDF and PDF from emission rate
!-------------------------------------------
      tnum=0.d0
      do ie=2,nbnum
        tnum=tnum+pbeta(ie)
      end do

      cbeta(1)=0.d0
      pbeta(1)=0.d0
      do ie=2,nbnum
        cbeta(ie)=cbeta(ie-1)+pbeta(ie)/tnum
        pbeta(ie)=pbeta(ie)/tnum
      end do

      iqin=nbtype          ! Incident charge - electrons
      ekein=ebeta(nbnum)   ! Maximum 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

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

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

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

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

      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.

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

```
!     Zero the variables
      do j=1,nbnum
        spe(j)=0.D0
      end do

!     Set histories and histories to write trajectories
      ncases=100000
!     Set maximum number for pict
      maxpict=500

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

!-----------------------------------------------------------------------
! Step 8:  Shower-call
!-----------------------------------------------------------------------
!     Write batch number
      write(39,fmt="('0    1')")
                                          ! -------------------------
      do i=1,ncases                       ! Start of batch -loop
                                          ! -------------------------

        wtin = 1.0

        wtsum = wtsum + wtin              ! Keep running sum of weights

!       ------------------------
!       Determine source energy
!       ------------------------
        call randomset(rnnow)
        do ie=2,nbnum
          if(rnnow.le.cbeta(ie)) go to 1000
        end do
1000    if(ie.gt.nbnum) ie=nbnum
        ekein=ebeta(ie-1)+(rnnow-cbeta(ie-1))*(ebeta(ie)-ebeta(ie-1))
     *      /(cbeta(ie)-cbeta(ie-1))
        spe(ie)=spe(ie)+1.0

        etot = ekein + 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

!       ---------------------------
!       Determine source  direction
!       ---------------------------

!       -------------------------
!       Determine source position
!       -------------------------

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

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

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

```
          stop
        end if

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

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

                                                ! ------------------
      end do                                    ! End of batch loop
                                                ! ------------------

      call plotxyz(99,0,0,0.D0,0.D0,0.D0,0.D0,0,0.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,'(A,G15.5)') ' Elapsed Time (sec)=',cputime

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

!     ---------------------------------------------------------
!     Source spectrum.  Incident particle spectrum to detector.
!     ---------------------------------------------------------
      write(6,'(/A,A72)') ' Result for ',soinf

      write(6,'(/A/30X,A/A,11X,A,11X,A)')' Sampled source spectrum',
     *    'particles/source',' Upper energy',' Electron',' pdf'

      do ie=2,nbnum

        spe(ie)=spe(ie)/ncount

        write(6,'(G10.5,A,8X,G12.5,8X,G12.5)')
     *     ebeta(ie),' MeV--',spe(ie),pbeta(ie)
      end do

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

      stop

      end

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

Appendix: Full listings of `ucradar.f`

```
!*********************************************************************
!****************************  KEK, High Energy Accelerator Research  *
!****************************  Organization                           *
!******* u c r a d a r   *****                                        *
!****************************    EGS5.0 USER CODE -  25 Mar 2016      *
!* 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 ucradar.f User Code requires a cg-input file only               *
! (e.g., ucradar.data).                                               *
! The following shows the geometry for usource.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 is sample user code to use RADAR beta-ray data.      *
! Use Ranlux random number generator.                                 *
!*********************************************************************
!                                                                     *
!             ----------------------                                  *
!             cg Geometry (ucsource)                                  *
!             ----------------------                                  *
!                                                                     *
!                                                                     *
!                      R                                              *
!                      ^                                              *
!                      |                                              *
!           +----+----+----+--------+------+---                       *
!           |         |                                              *
!           |    Outer| vacuum region                                *
!           +    +----+----+--------+------+--- r=6.0 cmm            *
!           |    |         Vacuum          |                         *
!           |    +    +----+--------+------+--- R=4.0 cm             *
!           |    |    |   Vacuum          |                         *
!           +    +    +    +--------+      +--- R=2.0 cm             *
!           |    |    |    |        |      |                         *
!           |    |    |    | Vacuum |Vacuum|                         *
! 1.253 MeV |    |    |    |        |      |                         *
!      =========>+----+----+--------+------+--------> Z               *
!     photons  -5.0  0.0  3.0      5.0    7.0 cm                     *
!                                                                     *
!                                                                     *
!                                                                     *
!                                                                     *
!*********************************************************************
!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/                          ! Variables to score
     * maxpict
      integer maxpict

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

      real*8                                              ! Local variables
     * availke,tnum,wtin,wtsum,xi0,yi0,zi0,
     * spe(MXEBIN),ebeta(21),pbeta(21),cbeta(21),
     * emax

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

      integer
     * i,icases,idin,ie,ifti,ifto,ii,j,k,n,ner,nbtype,nbnum

      character*10 atom
      character*72 soinf
      character*72 filename

      character*24 medarr(1)

!     ----------
!     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 call pegs5. Unit for pict must be 39.
!-----------------------------------------------------------------

      write(6,'(A/A,A)')
     *   ' Key in atomic name and mass number like Sr-90'

      read(5,*) atom

      filename='RADAR/'//trim(adjustl(atom))//'.dat'

      open(3,file=filename,STATUS='old')

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

!     ---------------------------------
!     Define media before calling PEGS5
!     ---------------------------------

      nmed=1
      medarr(1)='NAI                       '

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

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

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

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

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

!-----------------------------------------------------------------------
! Step 3: Pre-hatch-call-initialization
!-----------------------------------------------------------------------
!-----------------------------------------------
!     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,'(A)') 'CEND'

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

!   Read material for each refion from egs5job.data
      read(4,'(15I5)') (med(i),i=1,nreg)

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

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

!-----------------------------------------------------------------------
! Step 4:  Determination-of-incident-particle-parameters
!-----------------------------------------------------------------------
! Read beta-ray spectrum data fron RADAR data-base
```

```
      read(3,'(A72)') soinf
      read(3,*) nbtype
      nbnum=21
      ebeta(1)=0.d0
      do i=2,nbnum
        read(3,*) ebeta(i),pbeta(i)
      end do

!--------------------------------------
!     Calculate CDF from PDF
!--------------------------------------
      tnum=0.d0
      do ie=2,nbnum
        tnum=tnum+pbeta(ie)
      end do

      cbeta(1)=0.d0
      pbeta(1)=0.d0
      do ie=2,nbnum
        cbeta(ie)=cbeta(ie-1)+pbeta(ie)/tnum
        pbeta(ie)=pbeta(ie)/tnum
      end do

      iqin=nbtype          ! Incident charge - electrons
      ekein=ebeta(nbnum)   ! Maximum 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

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

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

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

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

      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.

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

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

!     Zero the variables
      do j=1,nbnum
        spe(j)=0.D0
      end do

!     Set histories and histories to write trajectories
      ncases=100000
!     Set maximum number for pict
      maxpict=500

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

!-----------------------------------------------------------------------
! Step 8:  Shower-call
!-----------------------------------------------------------------------
!     Write batch number
      write(39,fmt="('0    1')")
                                          ! -------------------------
      do i=1,ncases                       ! Start of batch -loop
                                          ! -------------------------

        wtin = 1.0

        wtsum = wtsum + wtin              ! Keep running sum of weights

!       ------------------------
!       Determine source energy
!       ------------------------
        call randomset(rnnow)
        do ie=2,nbnum
          if(rnnow.le.cbeta(ie)) go to 1000
        end do
1000    if(ie.gt.nbnum) ie=nbnum
        ekein=ebeta(ie-1)+(rnnow-cbeta(ie-1))*(ebeta(ie)-ebeta(ie-1))
     *      /(cbeta(ie)-cbeta(ie-1))
        spe(ie)=spe(ie)+1.0

        etot = ekein + 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

!       ---------------------------
!       Determine source  direction
!       ---------------------------

!       -------------------------
!       Determine source position
!       -------------------------

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

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

!       -----------------------------------------------------
!       Verify the normarization of source direction cosines
!       -----------------------------------------------------
        if(abs(uin*uin+vin*vin+win*win-1.0).gt.1.e-6) then
          write(6,fmt="(' Following source direction cosines are not',
```

```
     1    ' normarized.',3e12.5)")uin,vin,win
          stop
        end if

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

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

                                                 ! ------------------
      end do                                     ! End of batch loop
                                                 ! ------------------

      call plotxyz(99,0,0,0.D0,0.D0,0.D0,0.D0,0,0.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,'(A,G15.5)') ' Elapsed Time (sec)=',cputime

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

!     ------------------------------------------------------------
!     Source spectrum.  Incident particle spectrum to detector.
!     ------------------------------------------------------------
      write(6,'(/A,A72)') ' Result for ',soinf

      write(6,'(/A/30X,A/A,11X,A,11X,A)')' Sampled source spectrum',
     *    'particles/source',' Upper energy',' Electron',' pdf'

      do ie=2,nbnum

        spe(ie)=spe(ie)/ncount

        write(6,'(G10.5,A,8X,G12.5,8X,G12.5)')
     *     ebeta(ie),' MeV--',spe(ie),pbeta(ie)
      end do

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

      stop

      end

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

!------------------------------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 AUSGAB to: produce trajectory data for 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/lines.f'        ! Auxiliary-code COMMONs

      common/totals/                          ! Variables to score
```

```
     * maxpict
      integer maxpict

      integer                                                  ! Arguments
     * iarg

      real*8                                                ! Local variables
     * edepwt

      integer
     * ie,iql,irl

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

!     -----------------------------------
!     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,itvlfg,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 atvaltmp
      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'
         call srzone(xidd,yidd,zidd,iq_np,ir_np,irnext)
c
         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
c
           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'
             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
         itvlfg=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
          go to 410
  420     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
            itvlfg=1
            goto 425
          end if
        end do
  425   continue
        if(itvlfg.eq.0) then
          tval0=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----------------------
```