Sources

Advanced FLUKA Course

Overview

1. Built-in sources

- Beam definition
- Extended sources
- Sources for collider experiments

2. User-defined sources

- User routine SOURCE
- Useful auxiliary routines
- Sampling techniques
- Two-step methods

3. Example: point vs. extended source

Built-in sources

Beam definition – 1

Input card: BEAM

defines several *beam characteristics*:
type of particle, energy, divergence, profile

Example

```
*...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+...
BEAM             3.5 -0.082425       -1.7       0.0       0.0       0.0PROTON
```

- 3.5 GeV/c [**WHAT(1)**] proton beam [**SDUM**] with weight 1 [**WHAT(6)**]
- Gaussian momentum distribution: 0.082425 GeV/c FWHM [**WHAT(2)**]
- Gaussian angular distribution: 1.7 mrad FWHM [**WHAT(3)**]
- no beam width along x (point-like source) [**WHAT(4)**]
- no beam width along y (point-like source) [**WHAT(5)**]

Beam definition - 2

Input card: BEAMPOS

If **SDUM** = blank:

defines the coordinates of the centre of the beam spot (*i.e.*, the point from which transport starts) and the beam direction

Example

```
*...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+...
BEAMPOS          0.0       0.0      -0.1       0.0       0.0       0.0
```

- x-coordinate: 0.0 [**WHAT(1)**]
- y-coordinate: 0.0 [**WHAT(2)**]
- z-coordinate: -0.1 cm [**WHAT(3)**]
- direction cosine with respect to the x-axis: 0.0 [**WHAT(4)**]
- direction cosine with respect to the y-axis: 0.0 [**WHAT(5)**]
- **WHAT(6)** is not used !

→ beam points in the positive z-direction starting at (0.,0.,-0.1)

Beam definition - 3

Input card: BEAMAXES

defines the beam reference frame which all parameters defined with BEAM and BEAMPOS refer to (angular divergence, transverse profile, polarization, extended sources)

Example

```
*...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+...
BEAMAXES               1.0        0.0        0.0        0.0 0.7071068 0.7071068
```

- cosine of angle between x-axis of beam and x-axis of geometry frame [WHAT(1)]
- cosine of angle between x-axis of beam and y-axis of geometry frame [WHAT(2)]
- cosine of angle between x-axis of beam and z-axis of geometry frame [WHAT(3)]

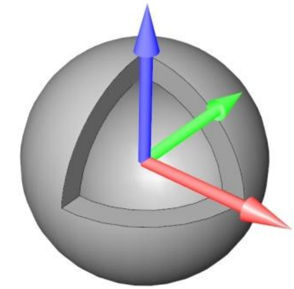
  (1.,0,0) → x-axes of beam and geometry frames are parallel

- cosine of angle between z-axis of beam and x-axis of geometry frame [WHAT(4)]
- cosine of angle between z-axis of beam and y-axis of geometry frame [WHAT(5)]
- cosine of angle between z-axis of beam and z-axis of geometry frame [WHAT(6)]

  (0.,0.7071068,0.7071068) → z-axes of beam frame is at 45deg to both y- and z-axes of geometry frame

Extended sources - *Spherical shell source*

Input card: BEAMPOS



If **SDUM** = **SPHE-VOL**:

defines a spatially extended source in a spherical shell

Example

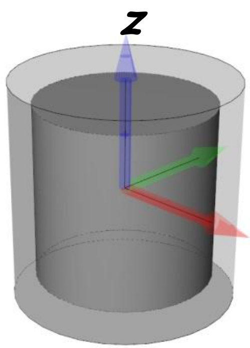
```
*...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+...
BEAMPOS          0.0       0.0       0.0       0.0       0.0       0.0
BEAMPOS          0.0       1.0       0.0       0.0       0.0       0.0SPHE-VOL
```

- radius (in cm) of the inner sphere shell: 0.0 cm [**WHAT(1)**]
- radius (in cm) of the outer sphere shell: 1.0 cm [**WHAT(2)**]
- **WHAT(3)** - **WHAT(6)** are not used !

The shell is centred at the (x,y,z) point defined by another BEAMPOS card with **SDUM** = blank (or = **NEGATIVE**). The particle direction or angular distribution are those defined by BEAM, BEAMAXES and another BEAMPOS cards.

Extended sources - *Cylindrical shell source*

Input card: BEAMPOS



If **SDUM** = **CYLI-VOL**:

defines a spatially extended source in a cylindrical shell with the height parallel to the z-axis of the beam frame

Example

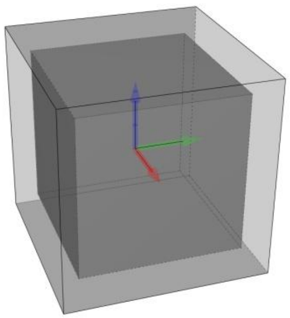
```
*...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+...
BEAMPOS          0.0       0.0       0.0       0.0       0.0       0.0
BEAMPOS          0.0       1.0       0.0       1.0       0.0       0.0CYLI-VOL
```

- radius (in cm) of the inner cylinder defining the shell: 0.0 cm [**WHAT(1)**]
- radius (in cm) of the outer cylinder defining the shell: 1.0 cm [**WHAT(2)**]
- height (in cm) of the inner cylinder defining the shell: 0.0 cm [**WHAT(3)**]
- height (in cm) of the outer cylinder defining the shell: 1.0 cm [**WHAT(4)**]
- **WHAT(5)** - **WHAT(6)** are not used !

The shell is centred at the (x,y,z) point defined by another BEAMPOS card with **SDUM** = blank (or = **NEGATIVE**). The particle direction or angular distribution are those defined by BEAM, BEAMAXES and another BEAMPOS cards.

Extended sources - *Cartesian shell source*

Input card: BEAMPOS



If **SDUM** = **CART-VOL**:

defines a spatially extended source in a Cartesian shell with the sides parallel to the beam frame axes

Example

```
*...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+...
BEAMPOS                0.0       0.0       0.0       0.0       0.0       0.0
BEAMPOS                0.0       1.0       0.0       1.0       0.0       1.0CART-VOL
```

- length (in cm) of the x-side of the inner parallelepiped defining the shell: 0.0 cm [**WHAT(1)**]
- length (in cm) of the x-side of the outer parallelepiped defining the shell: 1.0 cm [**WHAT(2)**]
- length (in cm) of the y-side of the inner parallelepiped defining the shell: 0.0 cm [**WHAT(3)**]
- length (in cm) of the y-side of the outer parallelepiped defining the shell: 1.0 cm [**WHAT(4)**]
- length (in cm) of the z-side of the inner parallelepiped defining the shell: 0.0 cm [**WHAT(5)**]
- length (in cm) of the z-side of the outer parallelepiped defining the shell: 1.0 cm [**WHAT(6)**]

The shell is centred at the (x,y,z) point defined by another BEAMPOS card with **SDUM** = blank (or = **NEGATIVE**). The particle direction or angular distribution are those defined by BEAM, BEAMAXES and another BEAMPOS cards.

Extended sources – *Spherical surface source*

Input card: BEAMPOS

If **SDUM** = **FLOOD**:

defines a source distribution on a spherical surface

Example

```
*...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+...
BEAMPOS          0.0       0.0       0.0       0.0       0.0       0.0
BEAMPOS          0.0       1.0       0.0       0.0       0.0       0.0FLOOD
```

- radius (in cm) of the sphere: 1.0 cm [**WHAT(1)**]
- **WHAT(2)** - **WHAT(6)** are not used !

The surface is centred at the (x,y,z) point defined by another BEAMPOS card with **SDUM** = blank (or = **NEGATIVE**). The particle direction or is distributed according to a diffusive distribution, so that to generate a uniform and isotropic fluence equal to $1/\pi R^2$ everywhere in the sphere (in absence of materials)

Extended sources - *Example*

Radioactive source of ^{60}Co (two main γ-emissions: 1332.5 keV and 1173.2 keV)

cylindrical shape, 2cm diameter, 2mm height along z, centre of base of cylinder at origin

```
*...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+...
BEAM             0.0                                                  ISOTOPE
HI-PROPE        27.0      60.0
BEAMPOS          0.0       0.0       0.1       0.0       0.0       0.0
BEAMPOS          0.0       1.0       0.0       0.2       0.0       0.0CYLI-VOL
```

or

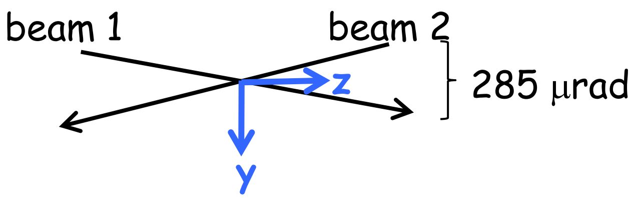
```
*...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+...
BEAM       1252.8E-6            10000.                                PHOTON
BEAMPOS          0.0       0.0       0.1       0.0       0.0       0.0
BEAMPOS          0.0       1.0       0.0       0.2       0.0       0.0CYLI-VOL
```

If height along x (instead of z) add

```
*...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+...
BEAMAXES         0.0       0.0      -1.0       1.0       0.0       0.0
```

Special sources – hadron-nucleus *collision*

Input card: SPECSOUR



Example: LHC

7 TeV/c, full crossing angle of 285 μrad in yz-plane

Momentum vectors of colliding proton beams: *three possibilities*

1) If **SDUM** = **PPSOURCE**:

```
SPECSOUR          0.    0.9975 6999.9999       0.0    0.9975-6999.9999PPSOURCE
```

- x, y, z-components of lab momentum for proton beam 1 [**WHAT(1-3)**]
- x, y, z-components of lab momentum for proton beam 2 [**WHAT(4-6)**]

2) If **SDUM** = **CROSSASY**:

```
SPECSOUR       7000.  142.5E-6      90.0     7000.  142.5E-6       0.0CROSSASY
```

- lab momentum for proton beam 1 [**WHAT(1)**]
- polar angle (rad) between proton beam 1 and positive z-direction [**WHAT(2)**]
- azimuth angle (deg!) defining crossing plane [**WHAT(3)**]
- lab momentum for proton beam 2 [**WHAT(4)**]
- polar angle (rad) between proton beam 2 and positive z-direction [**WHAT(5)**]

Special sources – hadron-nucleus *collision*

3) If SDUM = CROSSSYM:

```
*...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+...
SPECSOUR       7000.   142.5E-6       90.0         0.0         0.0         0.0CROSSSYM
```

- lab momentum for proton beams 1 and 2 [WHAT(1)]
- half crossing angle (rad) [WHAT(2)]
- azimuth angle (deg!) defining crossing plane [WHAT(3)]
- **WHAT(4) - WHAT(6)** are not used !

Interaction point of colliding proton beams (continuation card):

```
SPECSOUR       7000.   142.5E-6       90.0         0.0         0.0         0.0CROSSSYM
SPECSOUR      12.E-5     12.E-5        5.0                                    &
```

- sigma_x in cm for Gaussian sampling around XBEAM: 12 mm [WHAT(7)]
- sigma_y in cm for Gaussian sampling around YBEAM: 12 mm [WHAT(8)]
- sigma_z in cm for Gaussian sampling around ZBEAM: 5 cm [WHAT(9)]

(XBEAM,YBEAM,ZBEAM) defined with BEAMPOS card

Special sources – hadron-nucleus *collision*

```
BEAM          7000.0                                                  HEAVYION
HI-PROPE        82.0      208.0
...
SPECSOUR       7000.   142.5E-6       90.0        0.0        0.0        0.0CROSSSYM
SPECSOUR      12.E-5     12.E-5        5.0       -2.0      208.0       82.0&
```

- id of beam particle 1 (default the one of BEAM) [WHAT(10)]
- mass of beam particle 2 (default 1) [WHAT(11)]
- charge of beam particle 2 (default 1) [WHAT(12)]

For collisions in the DPMJET energy range, don't forget the following cards

```
*...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+...
EVENTYPE                                 2.0                          DPMJET
PHYSICS       8000.0                                                  LIMITS
```

Where "8000.0" should be larger than ½ of the centre-of-mass energy (the energy of whichever beam for a head-on collision in a symmetric collider)

WARNING: In the present release, EM-dissociation is not implemented at source collision level (e.g. Dominant contribution for LHC Pb-Pb)

BEAM Visualization

Within geometry viewer

For the moment the beam position and direction can be plotted with arrows inside the flair geometry editor.

- Add a #define to define the beam length
  #define bl 50.0

- Add an !arrow card and set as whats the following functions:

```
x:    =c(BEAMPOS,0,1)
y:    =c(BEAMPOS,0,2)
z:    =c(BEAMPOS,0,3)
dx:   =bl*c(BEAMPOS,0,4)
dy:   =bl*c(BEAMPOS,0,5)
dz:   =bl*sqrt(1.0-c(BEAMPOS,0,4)**2-c(BEAMPOS,0,5)**2)
```

c(BEAMPOS,n,m) is a function that returns from the n^{th} (zero based) BEAMPOS card the m^{th} argument

USRBIN

- Create a usrbin covering the beam position (preferentially X-Y-Z)
- Set as scoring particle the BEAMPART
- Setting all materials to vacuum (to speed up calculation)
- Make one run of 1 cycle
- Visualize the results:
  - in flair as USRBIN plot
  - in the geometry editor as a custom USRBIN layer (don’t forget to set properly the colorband)

With USERDUMP

- Add a USERDUMP card selecting ONLY Source particles
- Create a USERDUMP plot in flair
- Select the “Source” tab
- You have the ability to make
  - 1D histogram plots of any of the source quantities
  - 2D scattered plots for any of the quantities with even the possibility to overlay on an geometry image

User-defined sources

Source routine - 1

- Allows the definition of primary particle properties (in space, energy, time, direction or mixture of particles) which cannot be described with built-in sources

- Activated with input card SOURCE. The parameter list of that card (two continuation cards possible!) allows the user to pass on up to 18 numerical values `WHASOU(1-18)` and one 8-character string `SDUSOU` via `COMMON /SOURCM/`

- At each call, one (or more) particle(s) must be loaded onto `COMMON /FLKSTK/` (particle bank) before returning control. These values can be read from a file, generated by some sampling algorithm, or just assigned.

- Argument list: if `NOMORE=1` (output variable) the run will be terminated after exhausting the primary particles loaded onto the stack in the present call. The history number limit set with card START will be overridden.

Source routine - 2

```
...
      LOGICAL LFIRST
*
      SAVE LFIRST
      DATA LFIRST / .TRUE. /
...
      NOMORE = 0
*  +-------------------------------------------------------------------*
*  |  First call initializations:
      IF ( LFIRST ) THEN
*  |  *** The following 3 cards are mandatory ***
         LFIRST = .FALSE.
         TKESUM = ZERZER
         LUSSRC = .TRUE.
*  |  *** User initialization ***
```

Any first-time initialization can be inserted here, for example

- setting up parameters passed on via SOURCE card
- reading spectra from data files

```
      END IF
...
```

Source routine - 3

```
...
      NPFLKA = NPFLKA + 1
* Wt is the weight of the particle
      WTFLK  (NPFLKA) = ONEONE
      WEIPRI = WEIPRI + WTFLK (NPFLKA)
* Particle type (1=proton.....). Ijbeam is the type set by the BEAM
* card
*  +-------------------------------------------------------------------*
*  |  (Radioactive) isotope:
      IF ( IJBEAM .EQ. -2 .AND. LRDBEA ) THEN
         IARES  = IPROA
         IZRES  = IPROZ
         IISRES = IPROM
         CALL STISBM ( IARES, IZRES, IISRES )
         IJHION = IPROZ  * 1000 + IPROA
         IJHION = IJHION * 100 + KXHEAV
         IONID  = IJHION
         CALL DCDION ( IONID )
         CALL SETION ( IONID )
*  |
*  +-------------------------------------------------------------------*
*  |  Heavy ion:
      ELSE IF ( IJBEAM .EQ. -2 ) THEN
         IJHION = IPROZ  * 1000 + IPROA
         IJHION = IJHION * 100 + KXHEAV
         IONID  = IJHION
         CALL DCDION ( IONID )
         CALL SETION ( IONID )
         ILOFLK (NPFLKA) = IJHION
*  |  Flag this is prompt radiation
         LRADDC (NPFLKA) = .FALSE.
*  |  Group number for "low" energy neutrons, set to 0 anyway
         IGROUP (NPFLKA) = 0
*  |
*  +-------------------------------------------------------------------*
*  |  Normal hadron:
      ELSE
         IONID = IJBEAM
         ILOFLK (NPFLKA) = IJBEAM
*  |  Flag this is prompt radiation
         LRADDC (NPFLKA) = .FALSE.
*  |  Group number for "low" energy neutrons, set to 0 anyway
         IGROUP (NPFLKA) = 0
      END IF
*  |
*  +-------------------------------------------------------------------*
...
```

increase pointer in FLKSTK

weight of particle (if ≠ 1 biased source)
total weight of primaries (don't change)

Definition of particle type

- The template sets the type of particle equal to the one defined by the BEAM card (and HI-PROPE, if used).

- Whichever valid particle type can be set inside the source (may be different event by event)

Source routine - 4

```
...
* Particle age (s)
      AGESTK (NPFLKA) = +ZERZER
      AKNSHR (NPFLKA) = -TWOTWO
* Kinetic energy of the particle (GeV)
      TKEFLK (NPFLKA) = SQRT ( PBEAM**2 + AM (IONID)*:
* Particle momentum
      PMOFLK (NPFLKA) = PBEAM
* Cosines (tx,ty,tz)
      TXFLK  (NPFLKA) = UBEAM
      TYFLK  (NPFLKA) = VBEAM
      TZFLK  (NPFLKA) = WBEAM
*     TZFLK  (NPFLKA) = SQRT ( ONEONE - TXFLK (NPFLKA)*
     &                       - TYFLK (NPFLKA)**2 )
* Particle coordinates
      XFLK   (NPFLKA) = XBEAM
      YFLK   (NPFLKA) = YBEAM
      ZFLK   (NPFLKA) = ZBEAM
* Polarization cosines:
      TXPOL  (NPFLKA) = -TWOTWO
      TYPOL  (NPFLKA) = +ZERZER
      TZPOL  (NPFLKA) = +ZERZER
...
```

momentum and energy

- here: taken from BEAM card (PBEAM in COMMON /BEAMCM/)
- the set can set any momentum or energy here (either from file or sampled)
- **NOTE:** BEAM card is still mandatory for initialization purposes. Momentum and energy set here must not be larger than those defined with the BEAM card.

direction cosines and coordinates

- here: taken from BEAMPOS card (PBEAM in COMMON /BEAMCM/)
- ensure proper normalization of cosines!

polarization

- `TXPOL=-2` flag for "no polarization"

Source routine - 5

```
* User dependent flag:
      LOUSE  (NPFLKA) = 0
* User dependent spare variables:
      DO 100 ISPR = 1, MKBMX1
         SPAREK (ISPR,NPFLKA) = ZERZER
 100  CONTINUE
* User dependent spare flags:
      DO 200 ISPR = 1, MKBMX2
         ISPARK (ISPR,NPFLKA) = 0
 200  CONTINUE
```

Variables that allow to store additional information in `COMMON /FLKSTK/`, such as information on ancestors of a certain particle

Auxiliary routines – *Random numbers*

```
… = FLRNDM (XDUMMY)
```

returns a 64-bit random number [0-1)

NOTE: Fundamental for SOURCE! No other external random generators must be used, otherwise the history reproducibility will be lost.

```
CALL FLNRRN (RGAUSS)
```

returns a normally distributed random number RGAUSS

```
CALL FLNRR2 (RGAUS1,RGAUS2)
```

returns an uncorrelated pair of normally distributed random numbers RGAUS1 and RGAUS2

```
CALL SFECFE (SINT,COST)
```

returns SINT and COST, sine and cosine of a random azimuth angle
SINT**2 + COST**2 = 1.D+00

```
CALL RACO (TXX, TYY, TZZ)
```

returns a random 3D direction (TXX, TYY, TZZ) such that:
TXX**2 + TYY**2 + TZZ**2 = 1.D+00

Auxiliary routines – *Name/number conv.*

Conversion of region name to number

```
CALL GEON2R ( REGNAM, NREG, IERR )
```

Input variable:

Regnam = region name (CHAR*8)

Output variables:

Nreg = region number

Ierr = error code (0 on success, 1 on failure)

Conversion of region number to name

```
CALL GEOR2N ( NREG, REGNAM, IERR )
```

Input variable:

Nreg = region number

Output variables:

Regname = region name (CHAR*8)

Ierr = error code (0 on success, 1 on failure)

Auxiliary routines - *Others*

CALL OAUXFI ('file', LUN, 'CHOPT', IERR)

to open an auxiliary file (to read data or parameters) looking automatically for the file in some default locations (temporary directory, working directory)

CALL FLABRT ('name','message')

this allows to force a FLUKA abort on user request: it might be useful to perform a debugging (using gdb for instance)

CALL SFLOOD (XXX, YYY, ZZZ, UXXX, VYYY, WZZZ)

returns in XXX, YYY, ZZZ a random position ON the surface of a sphere of radius 1 and centre 0 (multiply XXX, YYY, ZZZ by the actual radius and add the centre coordinates) and UXXX, VYYY, WZZZ are random cosines distributed so as to generate a uniform and isotropic fluence inside the sphere numerically given by $1/(\pi R^2)$, R being the sphere radius.

Sampling from a distribution - *Discrete*

1) From the cumulative distribution

- Suppose to have a *discrete* random variable x, that can assume values $x_1, x_2, \ldots, x_n, \ldots$ with probability $p_1, p_2, \ldots, p_n, \ldots$
- Assume $\sum_i p_i = 1$, or normalize it
- Divide the interval [0,1) in n subintervals, with limits

$$y_0 = 0,\ y_1 = p_1,\ y_2 = p_1 + p_2, \ldots$$

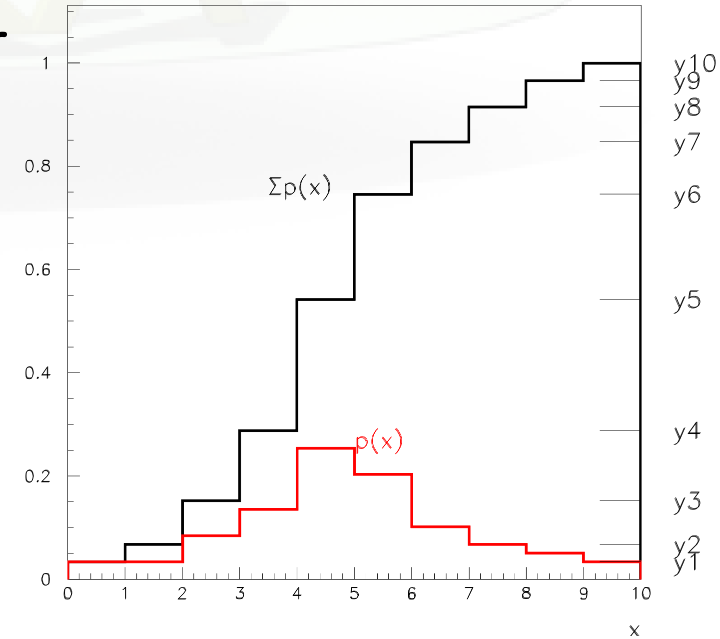
- Generate a uniform pseudo-random number ξ
- Find the interval i^{th} y-interval such that

$$y_{i-1} \le \xi < y_i$$

- Select $X = x_i$ as the sampled value

Since ξ is uniformly random:

$$P(x_i) = P(y_{i-1} \le \xi < y_i) = y_i - y_{i-1} = p_i$$



Sampling from a distribution – *Discrete*

2) By adjusting weights

- Suppose to have an fluence energy spectrum Φ given in N discrete energy bins between E_0 and E_N : $\Phi_1,\ldots,\Phi_N$
- Generate a uniform pseudo-random number ξ
- Find the interval i^{th} energy bin such that

$$E_{i-1} \leq \xi\,(E_N - E_0) < E_i$$

- Generate another uniform pseudo-random number $\xi \in [0,1)$ and sample an energy uniformly within the i^{th} energy bin
- assign a weight Φ_i to that primary particle

Note: This method is often used for spectra steeply decreasing with energy (e.g., $\Phi \sim 1/E$), where the result depends significantly on the particle cascades cause by high energy primaries, as it ensures faster convergence to the mean value.

Example Sampling from a histogram - 1

```
      PARAMETER (NMAX=1000)
      DIMENSION ERG(NMAX), CUM(NMAX)
      CHARACTER*250 LINE
      SAVE  N, ERG, CUM

       IF ( LFIRST ) THEN
       …
           LUNRD = NINT(WHASOU(1))
           N = 0
           SUM   = ZERZER
           EPREV = ZERZER
 10        CONTINUE
             READ (LUNRD, '(A)', ERR=9999, END=20 ) LINE
             READ (LINE, *, ERR=10) E, H
             N = N + 1
             IF (N .GT. NMAX)
     &            CALL FLABRT('SOURCE','Please increase NMAX')
             IF (N .EQ. 1 .AND. ABS(H).GT.AZRZRZ)
     &            CALL FLABRT(
     &                 'SOURCE','ZERO was was expected as first value')
*** Create cumulative sum of events! dN=dE*V
             SUM = SUM + H*(E-EPREV)
             EPREV = E
             ERG(N)  = E
             CUM(N)  = SUM
         GO TO 10
 20      CONTINUE
         CLOSE (LUNRD)
      END IF
```

Logical unit from input file
Use OPEN input card to open the file
Pass the unit with as what1 in SOURCE
The file contains
pairs Energy Value
WARNING first value should have be 0
in order to define the lower energy limit

Build cumulative sum (as a histogram
NOT as flux)

Example: hsource.f

Example Sampling from a histogram - 2

```
*  From this point .....

*** Select a random energy interval
      C = CUM(N) * FLRNDM(C)
*** Find interval (no need to check first interval CUM=0)
      DO I=2,N
        IF (CUM(I) .GT. C) THEN
***         Found interval I, select a random energy inside
            E = ERG(I-1) + (ERG(I)-ERG(I-1))*FLRNDM(C)
            GO TO 90
        END IF
      END DO
```

Select a random position

Find interval in cumulative sum

Select a random Energy inside the interval

Sampling from a distribution – *Continuous*

1) By integration

- Integrate the distribution function ***f(x)***, analytically or numerically, and normalize to 1 to obtain the normalized cumulative distribution

$$F(\xi) = \frac{\int_{x_{\min}}^{\xi} f(x)dx}{\int_{x_{\min}}^{x_{\max}} f(x)dx}$$

- Generate a uniform pseudo-random number $\xi \in [0,1)$
- Get the desired result by finding the inverse value $X = F^{-1}(\xi)$, analytically or most often numerically, i.e. by interpolation (table look-up)

Since ξ is uniformly random:

$$P(a < x < b) = P(F(a) \le \xi < F(b)) = F(b) - F(a) = \int_a^b f(x)dx$$

Sampling from a distribution – *Continuous*

Example

Take $f(x) = e^{-\frac{x}{\lambda}}$, $x \in [0,\infty)$

Cumulative distribution:

$$F(t) = \int_0^t e^{-\frac{x}{\lambda}} dx = \lambda \times \left(1 - e^{-\frac{t}{\lambda}}\right)$$

Normalized:

$$F'(t) = \int_0^t \frac{e^{-\frac{x}{\lambda}}}{\lambda} dx = 1 - e^{-\frac{t}{\lambda}}$$

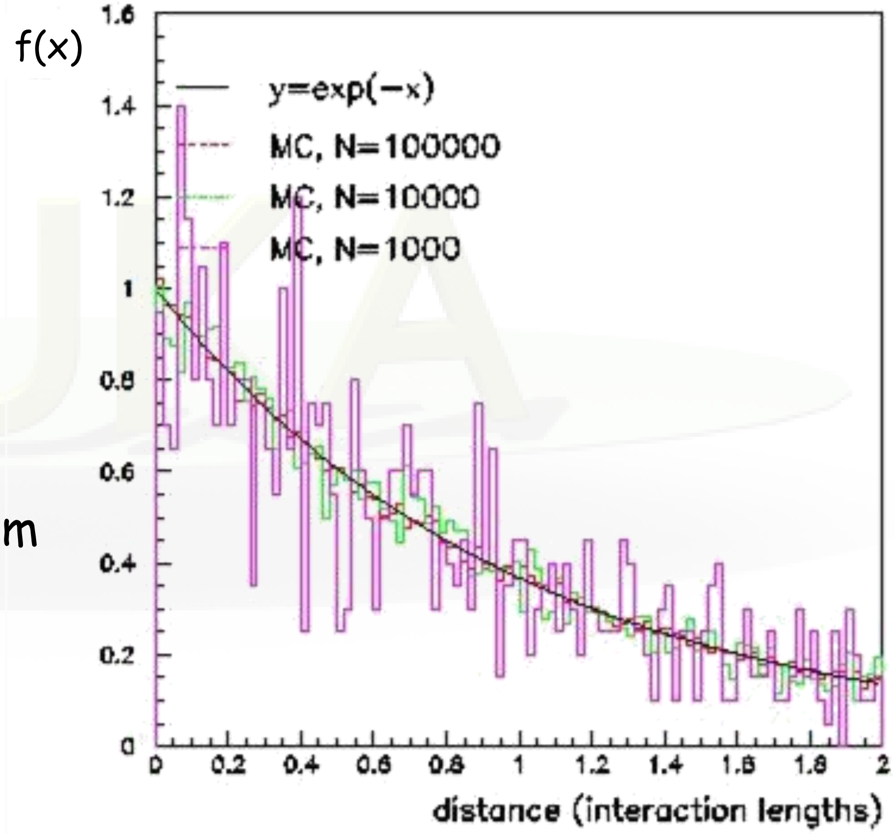
Generate a uniform pseudo-random number $\xi \in [0,1)$

$$1 - e^{-\frac{t}{\lambda}} = \xi$$

Sample t by inverting

$$t = -\lambda \ln(1 - \xi)$$

Repeat N times



Sampling from a distribution – *Continuous*

2) By rejection

- Let be *f′(x)*, a normalized distribution function, which cannot be sampled by integration and inversion
- Let be *g′(x)*, a normalized distribution function, which can be sampled, and such that $Cg'(x) \geq f'(x),\ \forall\ x \in [x_{min}, x_{max}]$
- Sample X from g′(x), and generate a uniform pseudo-random number $\xi \in$ **[0,1)**
- Accept X if $\xi < f'(X)/Cg'(X)$, if not repeat the previous step
- The overall efficiency (accepted/rejected) is given by:

$$R = \int \frac{f'(x)}{Cg'(x)} g'(x)\, dx = \frac{1}{C}$$

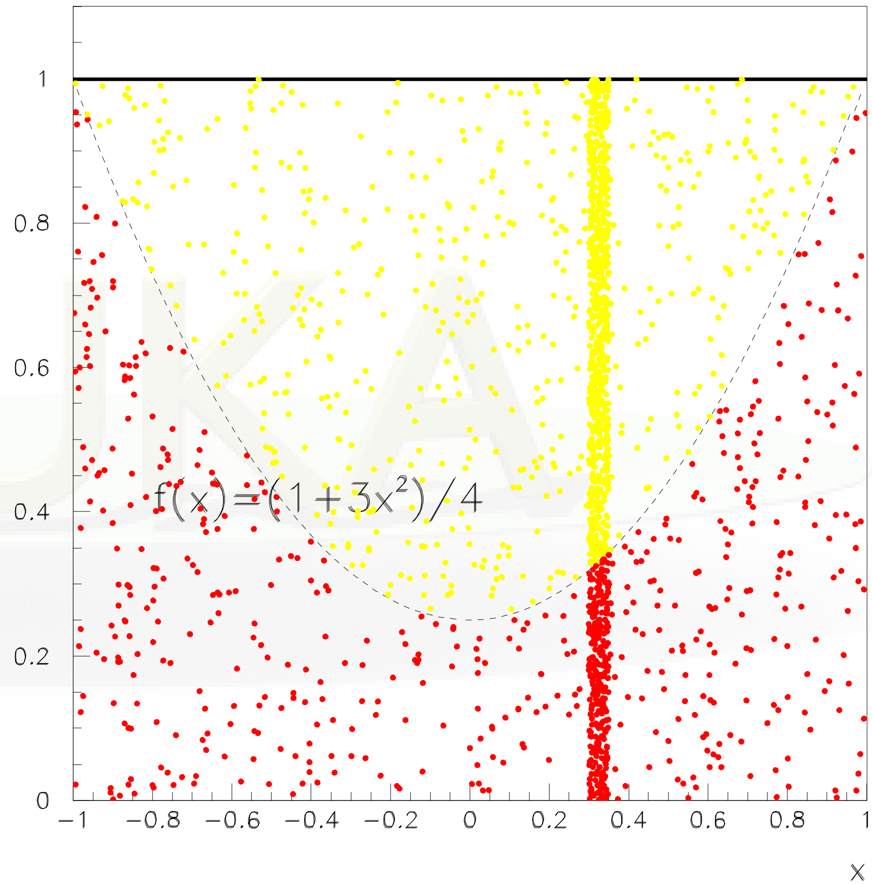
- and the probability that X is accepted is unbiased:

$$P(X)dX = \frac{1}{R} g'(X)dX \times \frac{f'(X)}{Cg'(X)} = f'(X)dX$$

Sampling from a distribution – *Continuous*

Example

- Let be $f'(x) = (1+3x^2)/4$, $x \in [-1,1]$,
- Take $g'(x)=1/2$, $C=2$
- Generate two uniform pseudo-random numbers $\xi_1, \xi_2 \in [0,1)$
- Accept $X=2\xi_1-1$ if $\xi_2 < (1+3X^2)/4$, if not repeat



Sampling from a distribution – *Continuous*

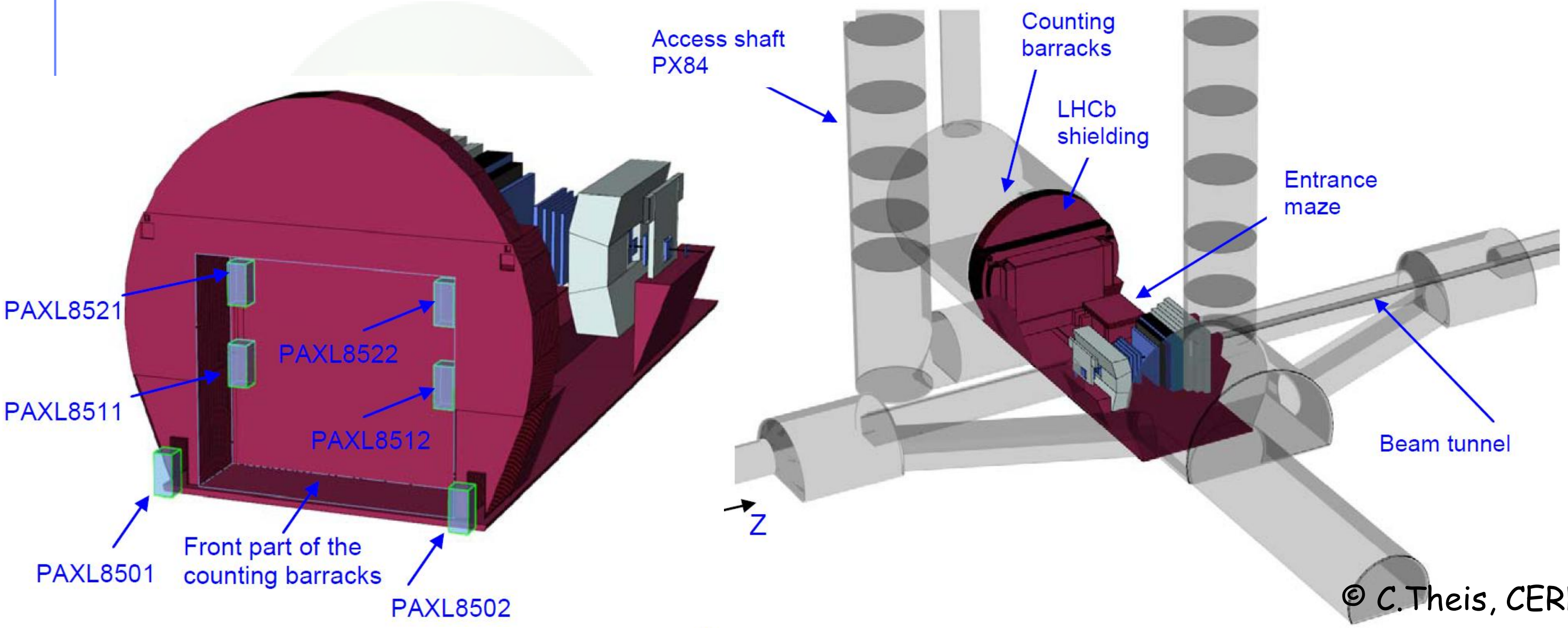
3) By adjusting weights

- Suppose to have a fluence energy spectrum $\Phi(E)$ given in between E_0 and E_1
- Generate a uniform pseudo-random number $\xi \in [0,1)$ and calculate the sampled energy $E = E_0 + \xi (E_1 - E_0)$
- Assign a weight $\Phi(E)$ to that primary particle

Two-step methods

Example:

predict reading of a (small) radiation detector at a remote location in a huge geometry, *e.g.*, LHCb experiment at CERN



© C.Theis, CERN

Problem: direct calculation in one step highly inefficient due to the small affected phase-space

Two-step methods

Solution: split simulation into two steps
1) Calculation of radiation field at detector location
2) Simulation of detector reading

Two options:

Directional dependence of detector reading is negligible

- calculate **average fluence energy spectra**, separately for different particle types, at the detector location
- simulate reading of detector with user-defined source which reads in the calculated spectra and samples particle type, energy and direction (*e.g.,* isotropic incidence)
- ***important:*** results of the second step have to be **normalized to the integrated particle fluence** obtained in the first step

Directional dependence of detector reading is important

- replace detector by 'blackhole' and write all information on particles entering it (type, energy, position, direction) into an **external file**
- simulate reading of detector (if possible with the original geometry now containing the detector) with user-defined source which reads in the particles from the external file
- ***important:*** **pick entries randomly** from external file to avoid going through identical sequence of particles if several runs are performed

Two-step example – Dumping particles

You can dump the particles with several ways e.g:

- mgdraw.f activated with USERDUMP
- fluscw.f activated with USERWEIGHT

The following example is using the fluscw.f activated with USERWEIGHt and coupled with the first USRBDX scoring

```
* Activate with USERWEIGHt Use FLUSCW+ WHAT(3)>2 to
* Couple scoring with the first Boundary crossing estimator
      IF (ISCRNG.EQ.1 .AND. JSCRNG.EQ.1) THEN
               IF (LFIRST) THEN
                   WRITE (99,*)
     &  '# 1.IJ  2.X 3.Y 4.Z  5.TX 6.TY 7.TZ   8.E 9.W'
                   LFIRST = .FALSE.
               END IF
               WRITE (99,'(I3,8(1X,F22.14))')
     &                   IJ,XX,YY,ZZ,TXX,TYY,TZZ,-PLA,WEE
      END IF
```

Example: rdfluscw.f

Two-step example – Sampling particles - 1

```
      PARAMETER (NMAX=1000000)
      SAVE LFIRST
      DATA LFIRST / .TRUE. /
      CHARACTER*250 LINE
      INTEGER    NNN, IJ(NMAX)
      DIMENSION  XXX(NMAX), YYY(NMAX), ZZZ(NMAX)
      DIMENSION  UUU(NMAX), VVV(NMAX), WWW(NMAX)
      DIMENSION  ERG(NMAX), WGT(NMAX)
      SAVE XXX, YYY, ZZZ
      SAVE UUU, VVV, WWW
      SAVE ERG, WGT

      IF ( LFIRST ) THEN
         LUNRD = NINT(WHASOU(1))
          NNN = 0
10        CONTINUE
            READ( LUNRD, '(A)', ERR=9999, END=20 ) LINE
            READ (LINE,*,ERR=10) I, X, Y, Z, U, V, W, E, WG
            NNN = NNN + 1
            IF (NNN.GT.NMAX) CALL FLABRT('SOURCE','Increase NMAX')
            …
```

Logical unit from input file
Use OPEN input card to open the file
Pass the unit with as what1 in SOURCE

Example: rdsource.f

Two-step example - Sampling particles - 2

```
          IJ(NNN)  = I
          XXX(NNN) = X
          YYY(NNN) = Y
          ZZZ(NNN) = Z
*  |  Normalize direction to 1.0
          UVW = SQRT(U**2 + V**2 + W**2)
          UUU(NNN) = U / UVW
          VVV(NNN) = V / UVW
          WWW(NNN) = W / UVW
          ERG(NNN) = E
          WGT(NNN) = WG
        GOTO 10
 20     CONTINUE
        IF (NNN.EQ.0) CALL FLABRT('SOURCE','Error reading file')
        WRITE (LUNOUT,*)
        WRITE (LUNOUT,*) '*** rdsource: ',NNN,' particles loaded'
        WRITE (LUNOUT,*)
      END IF
```

Normalize direction

Example: rdsource.f

Two-step example – Sampling particles - 3

```
      RNDSIG = FLRNDM (RNDSIG)
      N = INT(NNN*RNDSIG)+1
* Wt is the weight of the particle
      WTFLK   (NPFLKA) = WGT(N)
      ILOFLK (NPFLKA) = IJ(N)

* Kinetic energy of the particle (GeV)
*       TKEFLK (NPFLKA) = SQRT ( PBEAM**2 + AM (IONID)**2 ) - AM
(IONID)
      TKEFLK (NPFLKA) = ERG(N)
* Particle momentum
PMOFLK (NPFLKA) = SQRT ( TKEFLK (NPFLKA) * ( TKEFLK (NPFLKA)
     &                      + TWOTWO * AM (IONID) ) )
* Cosines (tx,ty,tz)
      TXFLK   (NPFLKA) = UUU(N)
      TYFLK   (NPFLKA) = VVV(N)
      TZFLK   (NPFLKA) = WWW(N)
* Particle coordinates
      XFLK    (NPFLKA) = XXX(N) !+ XBEAM
      YFLK    (NPFLKA) = YYY(N) !+ YBEAM
      ZFLK    (NPFLKA) = ZZZ(N) !+ ZBEAM
```

Choose a random particle
Has the benefit of reusing the recorded particles and all results will be normalized per recorded particle

Push particle into stack

Example: rdsource.f

Two-step: Normalization and Errors

- The dumped particles represent only a fraction of the full shower → therefore the second step consists only of a subset of the full simulation
- Thus the results of the second step should be multiplied (normalized) with the recorded weight of the first step

$$\text{Normalization} = \frac{\sum \text{weights of recorded particles}}{\sum \text{weights of source particles}}$$

WARNING:

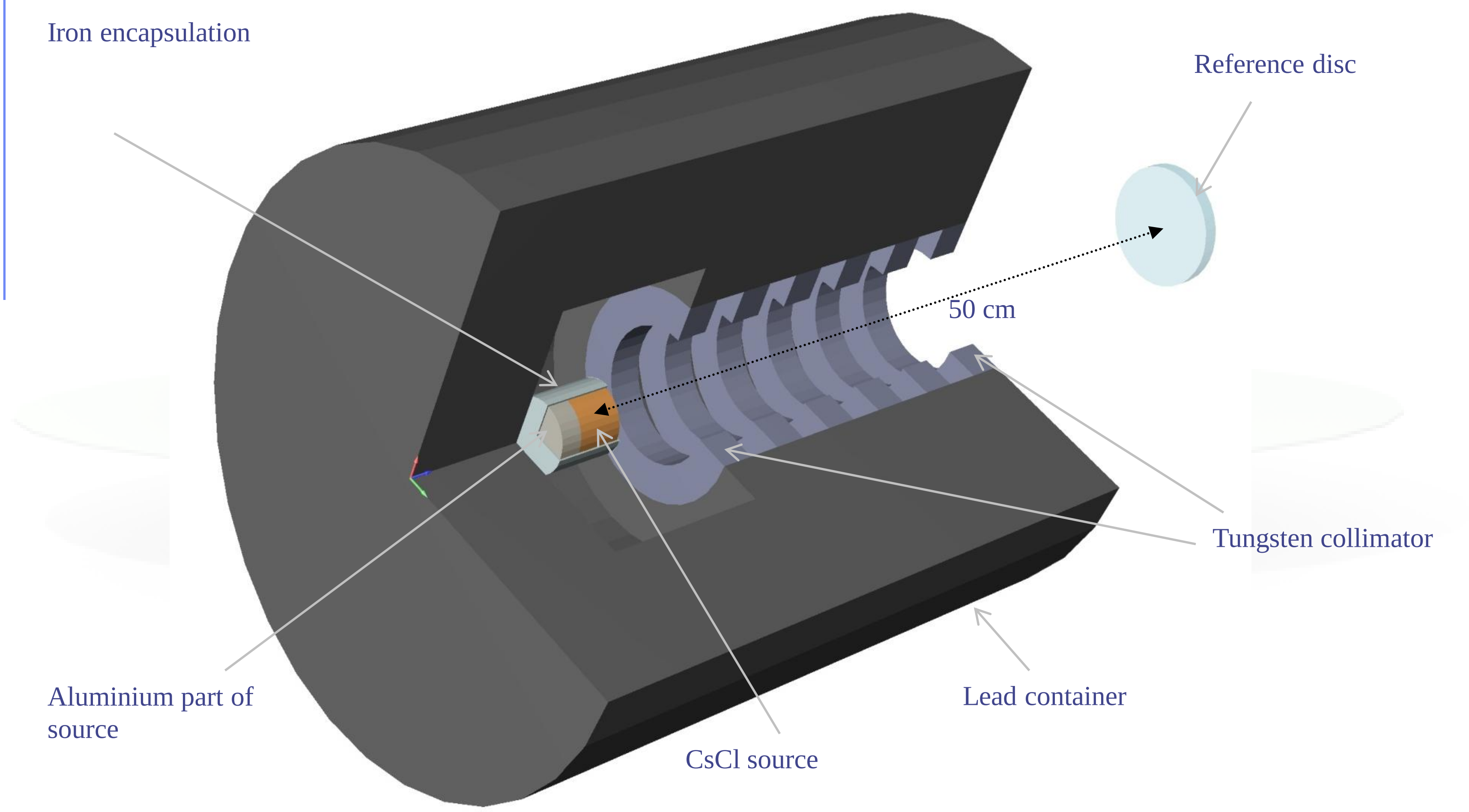
- verify that the recorded particles contains ALL the possible ones that contribute to the effect under study.
  You didn't miss any other that could have an impact on the results

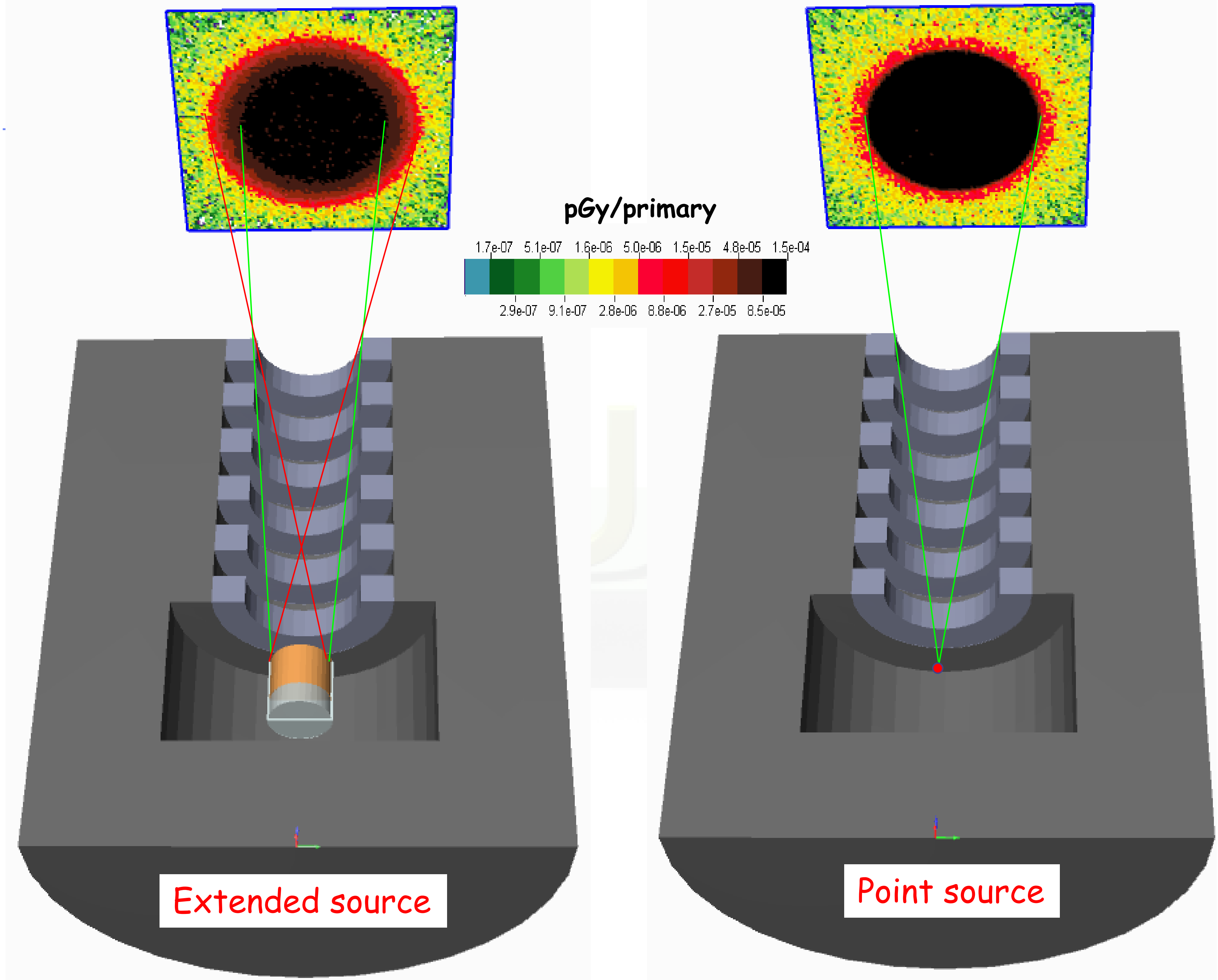
Two-step: Things to remember

1. Sample randomly the recorded particles on the second step! It has many benefits: *i)* you don't have to go through the entire list sequentially (especially if enormous), *ii)* you can reuse particles *iii)* results will be normalized per record particle
2. Verify your NORMALIZATION
   optionally you can make a full run to compare the results between the two step and full run
3. Like in a biasing run the purpose of a two step approach is to keep the mean but to reduce the error or time or study different configurations.
4. A more honest two step approach will be to record several, cycles (e.g. the typical 5 cycles) independently from the 1st step, and run separate 2nd steps one for each cycle.
   Merging the results will provide a more honest estimation of the variance
5. Verify that no other source of particles could contribute to your results (or at least is insignificant)

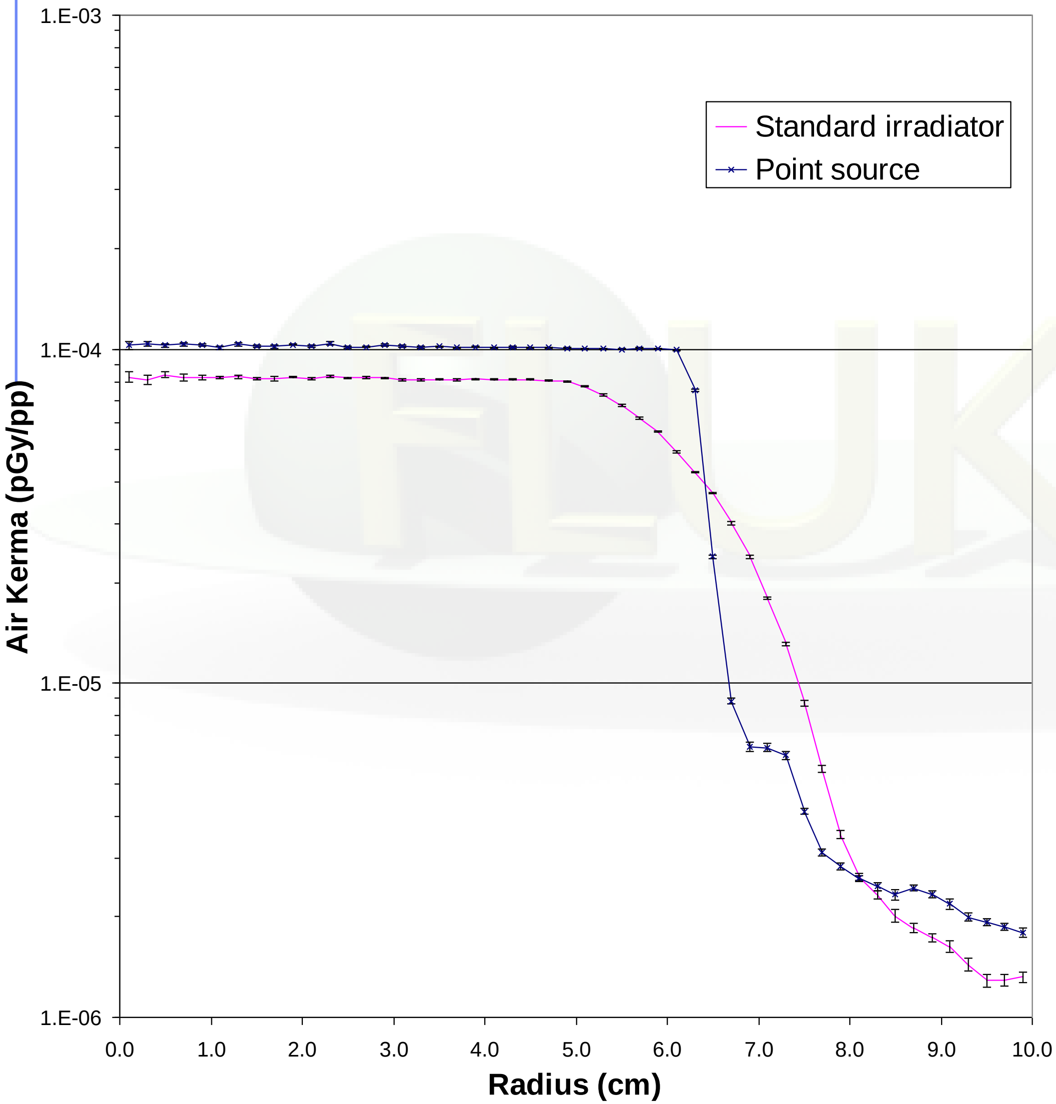
Example:
point vs. extended source

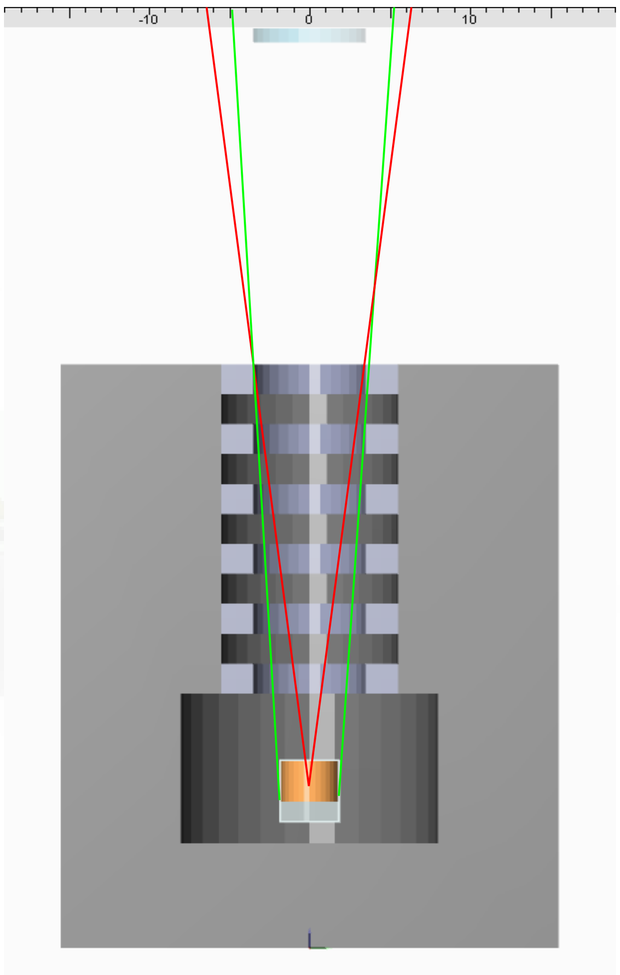
Example - *Cs irradiator*




pGy/primary
1.7e-07
5.1e-07
1.6e-06
5.0e-06
1.5e-05
4.8e-05
1.5e-04
2.9e-07
9.1e-07
2.8e-06
8.8e-06
2.7e-05
8.5e-05
Extended source
Point source

Example - *Cs irradiator*





© H.Vincke, CERN