

The source routine

Advanced course – ANL, June 2023

Why do we need s source routine

 The source routine is used to define complex sources, when the options provided in the BEAM, BEAMPOS and BEAMAXES cards are not enough.

- Most common use cases:
 - Mixed field
 - Beam with an energy spectrum
 - Complex beam shape
 - Second step of a two-step simulation



The "old" source routine

- Scary for beginners, limited documentation
- Use of **IMPLICIT** and **FORTRAN77** naming convention (see later)

63 * Push one source particle to the stack. Note that you could as well

	63 * Push one source particle to the stack. Note that you could as well
1 *	64 * push many but this way we reserve a maximum amount of space in the
2 * source*	65 * stack for the secondaries to be generated
3 *	66 * Npflka is the stack counter: of course any time source is called it
4 SUBROUTINE SOURCE (NOMORE)	67 * must be =0
5	68 NPFLKA = NPFLKA + 1
	69 * Wt is the weight of the particle
6 INCLUDE 'dblprc.inc'	70 WTFLK (NPFLKA) = ONEONE
7 INCLUDE 'dimpar.inc'	
8 INCLUDE 'iounit.inc'	71 WEIPRI = WEIPRI + WTFLK (NPFLKA)
9 *	72 * Particle type (1=proton). Ijbeam is the type set by the BEAM
10 **	73 * card
11 * *	74 * +
12 * Copyright (C) 2003-2019: CERN & INFN *	75 * (Radioactive) isotope:
13 * All Rights Reserved. *	76 IF (IJBEAM .EQ2 .AND. LRDBEA) THEN
14 * *	77 IARES = IPROA
	78 IZRES = IPROZ
15 * New source for FLUKA9x-FLUKA20xy: *	
16 * *	
17 * Created on 07 January 1990 by Alfredo Ferrari & Paola Sala *	80 CALL STISBM (IARES, IZRES, IISRES)
18 * Infn - Milan *	81 IJHION = IPROM * 100000 + MOD (IPROZ, 100) * 1000 + IPROA
19 *	82 IJHION = IJHION * 100 + KXHEAV
20 * This is just an example of a possible user written source routine. *	83 IONID = IJHION
21 * note that the beam card still has some meaning - in the scoring the *	84 CALL DCDION (IONID)
22 * maximum momentum used in deciding the binning is taken from the *	85 CALL SETION (IONID)
22 * maximum momentum used in deciding the binning is taken from the *	86 LFRPHN (NPFLKA) = .FALSE.
23 * beam momentum. Other beam card parameters are obsolete. *	
24 * *	87 *
25 * Output variables: *	88 *
26 *	89 * Heavy ion:
27 * Nomore = if > 0 the run will be terminated *	90 ELSE IF (IJBEAM .EQ2) THEN
28 * *	91 IJHION = IPROM * 100000 + MOD (IPROZ, 100) * 1000 + IPROA
*	92 IJHION = IJHION * 100 + KXHEAV
	93 IONID = IJHION
	94 CALL DCDION (IONID)
31 INCLUDE 'beamcm.inc'	
32 INCLUDE 'fheavy.inc'	95 CALL SETION (IONID)
33 INCLUDE 'flkstk.inc'	96 ILOFLK (NPFLKA) = IJHION
34 INCLUDE 'ioiocm.inc'	97 * Flag this is prompt radiation
35 INCLUDE 'ltclcm.inc'	98 LRADDC (NPFLKA) = .FALSE.
36 INCLUDE 'paprop.inc'	99 * Group number for "low" energy neutrons, set to 0 anyway
37 INCLUDE 'sourcm.inc'	100 IGROUP (NPFLKA) = 0
	101 * Parent radioactive isotope:
	102 IRDAZM (NPFLKA) = 0
39 *	
40 LOGICAL LFIRST, LISNUT	103 * Particle age (s)
41 *	104 AGESTK (NPFLKA) = +ZERZER
42 SAVE LFIRST	105 * Kinetic energy of the particle (GeV)
43 DATA LFIRST / .TRUE. /	106 TKEFLK (NPFLKA) = SQRT (PBEAM**2 + AM (IONID)**2)
44 * Statement function:	107 & - AM (IONID)
45 LISNUT (IJ) = INDEX (PRNAME (IJ), 'NEUTRI') .GT. 0	108 * Particle momentum
46 *	109 PMOFLK (NPFLKA) = PBEAM
	110 * PMOFLK (NPFLKA) = SQRT (TKEFLK (NPFLKA) * (TKEFLK (NPFLKA)
48 * BASIC VERSION *	
49 *	112 LFRPHN (NPFLKA) = .FALSE.
50 *====================================	113 *
51 NOMORE = 0	114 * +
52 * +*	115 * Normal hadron:
53 * First call initializations:	116 ELSE
54 IF (LFIRST) THEN	117 IONID = IJBEAM
55 * *** The following 3 cards are mandatory ***	118 ILOFLK (NPFLKA) = IJBEAM
	119 * Flag this is prompt radiation
56 TKESUM = ZERZER	
57 LFIRST = .FALSE.	120 LRADDC (NPFLKA) = .FALSE.
58 LUSSRC = .TRUE.	121 * Group number for "low" energy neutrons, set to 0 anyway
59 * *** User initialization ***	122 IGROUP (NPFLKA) = 0
60 END IF	123 * Parent radioactive isotope:
61 *	124 IRDAZM (NPFLKA) = 0
62 * +*	125 * Particle age (s)
V4	

```
TYPOL (NPFLKA) = +ZERZER
126
             AGESTK (NPFLKA) = +ZERZER
                                                                               190
                                                                                          TZPOL (NPFLKA) = +ZERZER
127 *
          Kinetic energy of the particle (GeV)
                                                                               191 * Particle coordinates
128
             TKEFLK (NPFLKA) = SQRT ( PBEAM**2 + AM (IONID)**2 )
                                                                                          XFLK (NPFLKA) = XBEAM
129
                            - AM (IONID)
                                                                               192
                                                                                          YELK (NPELKA) = YREAM
130
          Particle momentum
                                                                               193
131
             PMOFLK (NPFLKA) = PBEAM
                                                                               194
                                                                                          ZFLK (NPFLKA) = ZBEAM
                                                                               195 * Calculate the total kinetic energy of the primaries: don't change
132
             PMOFLK (NPFLKA) = SQRT ( TKEFLK (NPFLKA) * ( TKEFLK (NPFLKA)
133
    *
                                + TWOTWO * AM (IONID) ) )
                                                                               196 * +-----
                                                                                          (Radioactive) isotope:
                                                                               197 *
134
    *
135
                                                                               198
                                                                                          IF ( IJBEAM .EQ. -2 .AND. LRDBEA ) THEN
    *
             Check if it is a neutrino, if so force the interaction
136
                                                                               199 *
             (unless the relevant flag has been disabled)
    *
137
              IF ( LISNUT (IJBEAM) .AND. LNUFIN ) THEN
                                                                               200 *
138
                LFRPHN (NPFLKA) = .TRUE.
                                                                               201 *
                                                                                          Heavy ion:
                                                                                          ELSE IF ( ILOFLK (NPFLKA) .EQ. -2 .OR.
                                                                                202
139
140
                                                                                203
                                                                                                    ILOFLK (NPFLKA) .GT. 100000 ) THEN
    *
                                                                                204
                                                                                             TKESUM = TKESUM + TKEFLK (NPFLKA) * WTFLK (NPFLKA)
141
             Not a neutrino
    *
142
                                                                               205 *
             ELSE
143
                LFRPHN (NPFLKA) = .FALSE.
                                                                                206 *
                                                                               207 *
                                                                                          Standard particle:
144
             END IF
                                                                                208
                                                                                          ELSE IF ( ILOFLK (NPFLKA) .NE. 0 ) THEN
145
    .
                                                                                209
                                                                                              TKESUM = TKESUM + ( TKEFLK (NPFLKA) + AMDISC (ILOFLK(NPFLKA))
146
    *
147
                                                                                210
                                                                                                    * WTFLK (NPFLKA)
          END IF
                                                                                211 *
148 +
                                                                               212 *
149 * +-----
150 * From this point ....
                                                                               213 *
                                                                               214
151 * Particle generation (1 for primaries)
                                                                               215
                                                                                             TKESUM = TKESUM + TKEFLK (NPFLKA) * WTFLK (NPFLKA)
          LOFLK (NPFLKA) = 1
152
                                                                               216
153
    * User dependent flag:
                                                                                          END IF
                                                                               217 *
154
          LOUSE (NPFLKA) = 0
                                                                               218 * +-
155
       No channeling:
                                                                                          RADDLY (NPFLKA) = ZERZER
          KCHFLK (NPFLKA) = 0
                                                                               219
156
                                                                               220 * Here we ask for the region number of the hitting point.
157
          ECRFLK (NPFLKA) = ZERZER
                                                                                          NREG (NPFLKA) = ...
    * Extra infos:
                                                                               221 *
158
                                                                               222 * The following line makes the starting region search much more
159
          INFSTK (NPFLKA) = 0
                                                                               223 * robust if particles are starting very close to a boundary:
224 CALL GEOCRS ( TXFLK (NPFLKA), TYFLK (NPFLKA), TZFLK (NPFLKA) )
160
          LNFSTK (NPFLKA) = 0
161
          ANFSTK (NPFLKA) = ZERZER
                                                                               225
                                                                                          CALL GEOREG ( XFLK (NPFLKA), YFLK (NPFLKA), ZFLK (NPFLKA),
162
    * Parent variables:
                                                                                                       NRGFLK(NPFLKA), IDISC )
                                                                               226
                                                                                         8
163
          IPRSTK (NPFLKA) = 0
                                                                               227 * Do not change these cards:
164
          EKPSTK (NPFLKA) = ZERZER
                                                                                          CALL GEOHSM ( NHSPNT (NPFLKA), 1, -11, MLATTC )
165
    * User dependent spare variables:
                                                                               228
                                                                                          NLATTC (NPFLKA) = MLATTC
166
        DO 100 ISPR = 1, MKBMX1
                                                                               229
                                                                               230
                                                                                          CMPATH (NPFLKA) = ZERZER
167
             SPAREK (ISPR, NPFLKA) = ZERZER
     100 CONTINUE
                                                                               231
                                                                                          CALL SOEVSV
168
                                                                               232
                                                                                          RETURN
169
    * User dependent spare flags:
170
          DO 200 ISPR = 1, MKBMX2
                                                                               233 *== End of subroutine Source ===============================
             ISPARK (ISPR,NPFLKA) = 0
                                                                               234
                                                                                          END
171
                                                                               235
172
    200 CONTINUE
173 * Save the track number of the stack particle:
                                                                               236
          ISPARK (MKBMX2,NPFLKA) = NPFLKA
174
175
           NPARMA = NPARMA + 1
          NUMPAR (NPELKA) = NPARMA
176
177
          NEVENT (NPFLKA) = 0
          DENEAR (NPELKA) = +ZERZER
178
179
        ... to this point: don't change anything
180
          AKNSHR (NPFLKA) = -TWOTWO
181
    Cosines (tx,ty,tz)
182
          TXFLK (NPFLKA) = UBEAM
183
          TYFIK (NPFIKA) = VRFAM
184
          TZFLK (NPFLKA) = WBEAM
185
          TZFLK (NPFLKA) = SQRT ( ONEONE - TXFLK (NPFLKA)**2
186
                                 - TYFLK (NPFLKA)**2 )
    +
         5
187 * Polarization cosines:
```



TXPOL (NPFLKA) = -TWOTWO

188

The "new" source routine

- Distributed since FLUKA4-1.0 release
- Simplified appearance
- Long & meaningful names for variables and routines
- Use of implicit none (see later)
- Well documented by comments and in the manual
- Variables for user's usage clearly indicated
- Lines not to be edited are "hidden" in routines in the source_library.inc library file
- Old source routines can still be used



The "new" source routine

subroutine SOURCE (momore)

Copyright (C) 2020: CERN Source routine or FLUKA 4: Created on 24 September 2020 by David Hervath & Roberts Versac FLI Bearlines Modified on 17 November 2020 by David Horvath 0 Roberto Versaci ELI Beamlines this is a simplified user written source routine utilizing a separate source voutine library. It is intended as an alternative new-user-friendly version of the swarte.f routime. Existing FLMAA 4 source routines remain commative. Note that the beam card still has some meaning - in the scoring maximum momentum used in deciding the binning is taken from the beam momentum. Other beam card parameters are obsolete.

Output variables: nomore - if > 0 the run will be terminated

Quick start guide:

This user source routine template mins to modernize the legacy routine by implementing modern Fortram conventions and to provide built in sampling functions. The users only need to change / add code between the BEGISWEMG and END marks, one soction for doclaration of user variables, and one for assigning values to the beam parameters.

By default there is no user variable defined, and all code lines for parameter assignment are communication. These communic start with the symbol: " $^{++}$. To enable one, the $^{++}$ needs to be deleted.

(Note: In Fortran each code line should start in column 7 or further in.) Every beam parameter has a default value based on the FLURA input file. A parameter assignment should only be used if the default value has to be

There are three ways to assign a value to a parameter: 1. Direct assignment: A parameter is equal to a value. For example: scentum, energy = 8.100

If the parameter defined as a double precision, then the assigned value should be represented as double precision as well so as to not loose numerical precision. To do this a "D" exponential mark must be used.

Using a sampling function: A parameter is assigned to a value, which is calculated by a separate function. For example:

coordinate x = sample flat distribution([min], [max]) The parameters between the '[' and ']' brackets need to be replaced with numbers, or user variables containing the desired values.

 Using a sampling subroutine: They are similar to function, but they are not returning values directly; instead they modify the variables in their argument list. For example: call sample_annular_distribution(Inmin], [rmax], coordinate_x, con

The example above has two input parameters between brackets, and two output parameters (without bracket). The input parameters have to be provided, as for functions. The output parameter names usually don't need to be changed, but there are cases, where a subset of possible output parameters has to be selected.

for further details see the flink manual.

module source_variables implicit none

integer, save :: particle_code integer, save :: heavyion_stomic_number, heavyion_mass_number, heavyion_isomer logical, save :: radioactive_isotope

double precision, save :: nonestum_energy, particle_weight legical, save :: energy_logical_flag

double precision, save :: divergence_x, divergence_y
logical, save :: gaussian_divergence_logical_flag

double precision, save :: coordinate_s, coordinate_y, coordinate_r

integer, save :: direction_flag double precision, save :: direction_cosx, direction_cosy, direction_cosx double precision, save :: polarization_cosy, polarization_cosy, polarization_cosy 223 + radioactive_isotope - .trve.

double precision, save :: particle_age double precision, save :: kshort_component double precision, save :: delayed_radioactive_decay

end module source_variables

Include 'source_library.isc'

FLUKA

use source_library use source_variables implicit none logical lfirst save lfirst data lfirst / .true. / integer nomore logical debug_logical_flag
data debug_logical_flag / .false. / integer debug_lines data debug_lines / 100 / double precision adamny twoe(chase space) phase space entry double precision sample_flat_momentum_energy double precision sample_gaussian_momentum_energy double precision sample_inisogram_momentum_energy double precision sample_inisogram_momentum_energy

double precision sample_gaussian_distribution double precision sample_flat_distribution double precision FLRMON

END of user declared variables nomore = 0

if (lfirst) them
 call initialization()
 lfirst = .false.
end if

BEGINNING of customizable code

1 1. Accessing variables from the SOURCE card tvalues set on the SOURCE card can be accessed with the following variables: mumerical values (souble precision): muHSOU(), wwFSOU(), ... wHSOU(18) SOUR text (& character): fontoor

1 2. Primary particle

1 2.1. Particle code FILME particle code of the primary See section 5.1 of the FILMA menual for the list of particle code ! Defaulti # Farticle code of the primary defined on the BEAM card if present, otherwise 1 (proton) 195 196 • particle_code = ...

1 2.2. Heavy ion 1F the HLAWYON particle type (particle_code - -2) has been selected on the BLAW card, the ion can be specified with the following parameters: 1 Specified on HL-MONE card (if present), otherwise 2-6, A-12, 1-8 (12C)

206 • heavyion_atomic_number = ... 207 • heavyion_mass_number = ... 208 • heavyion_isomer = ... 1 2.3. Radioactive isotope

i Selects if the specified ion is to be used as a radioactive source I Default: remail() .true. : if ISOTOPE particle type is selected on the BEAM card, .false. : otherwise

! Note: ! Requires RADDECAY (semi-analogue) and DCYSCDRE cards ! Direction sampling is always isotropic ! Romentum / emergy settings are disregarded

3. Particle momentum / energy and weight

1 3.1. Momentum & Energy Set the momentum [GeV/c] or the kinetic energy [GeV] of the primary particle for heavy ions values are per nuclear mass unit. 233 | Default: 234 | Momentum calculate 235 236 * momentum_energy = ... ! Default: ! Romentum calculated from values set on the BEAM card (if present), 200 GeV/c otherwise

1 3.2. Energy flag

I Solici becomendia and energy I fibe energy flg is: - faits: The monoton_energy worlable contains the momentum of the particle I fibe monoton_energy worlable contains the kisetic energy of the particle I effective I faits:

237 238 239 240 241 242 243 244 245 246 245 246 247 248 * 249 energy_logical_flag = .true 1 3.3. Particle weight

> f Sets the initial weight of the primary Default: 1.000 particle_weight = ...

3.4. Sampling functions and subroutines

3.4.1. Flat distribution

nomentum_energy = sample_flat_momentum_energy([min], [max]) 3.4.2. Gaussian distribution

271 | ------272
273 * nomentum_energy = sample_gaussian_momentum_energy([nean], [fwhm])
274 3.4.3. Maxwell-Boltzmann distribution

Tengerature is given in GeV, energy flag must be .true. momentum energy = sample maxwell boltzmann energy([temperature])

1 3.4.4. Sampling from histogram Possible [usit]s: "ter", "Gev", "Mev", "kev" "ev" "ter/_", "Gev", "Gev(", "Mev/c", "kev/c", "kev/c", Mistogram file in ter bin - Thmac (of the bin) - GW/GE (bin beight; NOTE: doesn't need to be normalized)

momentum energy = sample histogram momentum energy([filename], [unit]) 1 3.4.5. Exponential distribution

------Ingut variables: - e_min [GeV] - e_max (GeV] - intensity_ratio = (int_e_max / int_e_min) Outgut variables: - maxetian_energy - particle_mergy

call sample exponential energy weight([e min], [e max], [intensity ratio], momentum energy, particle weight) 4. Beam angular divergence

1 4.1. Divergence value 1 Sets the beam divergence in the X-2 (divergence_x) and Y-2 (divergence_y) planes [rad] Divergences are applied before beam direction Defaults: Set on the BEAM card if present (converted to radians), 0.0 otherwise

 315
 1 Divergences are app

 316
 1 Defaults:

 317
 1 Set on the BLAN

 318
 4 divergence_x = ...

 320
 4 divergence_y = ...

1 4.2. Divergence type 1. Select these That and Gaussian divergence. If it is not to: If it is not to: .tensor that the selection of the selection of the selection of the distribution tensor. The divergence is the sourch — divergence values are taken as pawn of the distribution to behavit: . Set is the BLUM card (if present), fails, atherwise
 15
 1 Selects between flat and Gaussian divergence

 16
 17 is is not too.

 17
 1 sets to too.

 18
 1 felster.

 18
 1 true.

 19
 1 felster.

 19
 1 felster.

 10
 1 felster.

 10
 1 felster.

 10
 1 felster.

 11
 1 felster.

 120
 1 felster.

 120
 1 felster.

 121
 1 felster.

 122
 1 felster.

 123
 1 felster.

 124
 1 felster.

 125
 1 felster.

 126
 1 felster.

 127
 1 felster.

 128
 1 felster.

 129
 1 felster.

 120
 1 felster.

 121
 1 felster.

 122
 1 felster.

 1231
 1 felster.

5. Beam starting position

1 5.1. Coordinates Sets the starting coordinates (x,y,z) of the beam [cm] locality: Coordinates set on the BEAMPOS card if present, (0.000, 0.000, 0.000) otherwise

Note: the snapshot is not meant to be read – Detailed view will follow

5.2. Sampling functions and subroutines 1 5.2.1. Flat distribution Replace [a] with "x", "y", or "z".

357
358 * coordinate [a] s sample flat distribution([min], [max]) 1 5.2.2. Gaussian distribution Replace [a] with fst, fyt, or fst,

coordinate [a] + sample gaussian distribution([mean], [fwhm])

1.5.2.3. Annular distribution

 Applies an annular distribution to any two convinctos
 Topol variables:
 Two contributes of the contex of the annular distribution (coordinate_(a/b)) [cn]
 Two contributes of the contex of the annular distribution (coordinate_(a/b)) [cn]
 mapping contributes of the contex of the annular distribution (coordinate_(a/b)) [cn] supplate [a] and [a] match A , , ,
 cutput variables:
 - Modified coordinates of the sampled location (input values have been overwritten) call sample annular distribution([rmin], [rmax], coordinate [a], coordinate [b])

6. Beam direction

1 6.1. Direction cosines 1 Sets the direction cosines of the beam with respect to the X,Y and Z-axis Defaults: Set on the BEAMPOS card if present, otherwise (0.000, 0.000, 1.000)

1 6.2. Direction flag and the second second second second (with a still be sound(ind))
 a (1) since in a state into account (with a still be sound(ind))
 a (1) since into account (with a still be sound(ind))
 a (1) since into account (with a still be sound(ind))
 a (1) since into account (with a still be sound (ind))
 a (1) since into account (with a still be sound (ind))
 a (1) since into account (with a still be sound (ind))
 a (1) since into account (with a still be sound (ind))
 a (1) since into account (ind))
 a (1) since into account (with a still be sound (ind))
 a (1) since into account (ind))

direction_flag = ... 1 6.3. Sampling functions and subroutines

4.3.1. Isotropic distribution Cutput variables: - direction_cosx - direction_cosy - direction_cosz call sample_isotropic_direction(direction_cosx, direction_cosy, direction_cosz)

1 7. Other changeable parameters I for most of the uses none of these parameters should be changed from the defaults 1.7.1. Polarization coninan The three inputs indicate the direction cosines of the particle polarization Defaults: (-2.000, 0.000, 0.000)

polarization_cosx = ... polarization_cosy = ... polarization_cosz = ... 1 7.2. Particle age

Sets the starting age of the primary particle in seconds Default: 0.000 640 s particle are s ...

1 7.3. Kshort compo

Sets The Kshort component of the KO/K8ber Default: -2.40 i55 i56 * kshort_component = .

The source routine

1 7.4. Delayed radioactive decay Sets the delay for the radioactive decay with respect to the standard primary zero time Default: 0.000 465 * delayed_radioactive_decay = ...

1 8. Sampling from phase space file Allows to read particle information from a phase space file and sets the primary accordingly

! 8.1. Input variables

- [filename] - [filename] - [energy_umil]: - Possible [energy unil]s: "TeV", "GeV", "MeV/c", "keV/c", "eV/c" - "TeV/c", "GeV/c", "MeV/c", "keV/c", "eV/c"

PORDer umsg musses "Tere's, "Rev's, "Rev's,

1 8.2. File format

The phase space file has to contain the following columns in this order: ! - Particle code [integer]

1 - Particle momentum / energy [double precision]

! - Starting X coordinate [double precision] ! - Starting Y coordinate [double precision] ! - Starting Z coordinate [double precision]

Starting X direction cosine [double precision Starting Y direction cosine [double precision Starting Z direction cosine [double precision]

I - Particle weight [double precision] 1 8.3. Output variables

phase_space_entry: Variable containing information of a single particle from the phase space file
 nomore: Fleg to indicate that all particles has been read

1 8.4. Subroutine call

call read_phase_space_file([filename], [energy_unit], [length_unit], phase_space_entry, [sequential_logical_flag], nomore 1 8.5. Reading information from 'phase space entry' variable

The information stored in the 'phase space entry' has to be copied to the appropriate variables particle_code = phase_space_entry%pc momentum_energy = phase_space_entry%m_e

emergy_logical_flag = .true.

coordinate_x = phase_space_entry% coordinate_y = phase_space_entry% coordinate_z = phase_space_entry%

direction_cosx = phase_space_entry%
direction_cosy = phase_space_entry%
direction_cosz = phase_space_entry% 535 * particle_weight = phase_space_entry%we

9. Debugging

1 9.1. Debug logical flag Enables or disables the printout of the beam parameters for debugging Possible values: . false. : Debug output disabled (Default) . true. : Debug output enabled

debug_logical_flag = .true. 1 9.2. Debug lines

Sets the maximum number of lines printed in the debug output Default: 100

58 * debug_lines = 100

END of customizable code - Do not change below

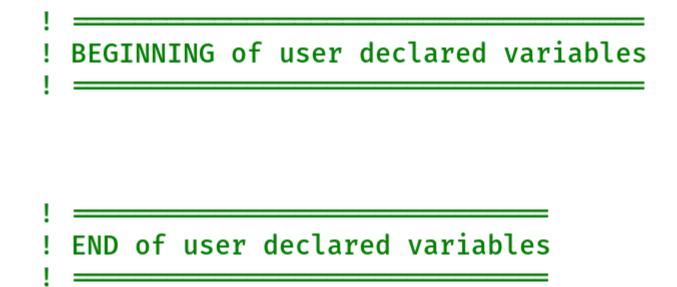
call set primary()

573 return 574 see End of subroutine Source ------

if (debug_logical_flag) call print_primary(debug_lines) end if

5

Source routine – User declaration



• Dedicated space for the declaration of user variables (and functions)



Source routine – Initialization

- Initialization of internal variables Runs every time, resetting their values to the defaults
- Custom initialization block Runs only at the first call

call initialization()

if (first_run) then

BEGINNING of custom initialization

END of custom initialization

first_run = .false.
end if



Source routine – Main section

- For setting the internal variables directly, or using one of the sampling routines
- To enable a line, remove the *
- The variables with '[' ']' brackets and ... are placeholders, they need to be replaced with values or user variables
- Always use double precision format for floating point numbers

```
BEGINNING of customizable code
particle code = ...
heavyion_atomic_number = ...
heavyion_mass_number = ...
heavyion isomer = ...
momentum_energy = ...
energy_logical_flag = .true.
particle_weight = ...
momentum_energy = sample_flat_momentum_energy( [min], [max] )
momentum_energy = sample_gaussian_momentum_energy( [mean], [fwhm] )
momentum_energy = sample_maxwell_boltzmann_energy( [temperature] )
momentum_energy = sample_histogram_momentum_energy( [filename], [unit] )
momentum_energy = sample_spectrum_momentum_energy( [filename], [unit] )
call sample_exponential_energy_weight( [e_min], [e_max], [intensity_ratio], momentum_energy, particle_weight )
divergence_x = ...
divergence_y = ...
gaussian_divergence_logical_flag = .true.
coordinate x = ...
coordinate_y = ...
coordinate_z = ...
coordinate_[a] = sample_flat_distribution( [min], [max] )
coordinate [a] = sample gaussian distribution( [mean], [fwhm] )
call sample_annular_distribution( [rmin], [rmax], coordinate_[a], coordinate_[b] )
direction_cosx = ...
direction_cosy = ...
direction_cosz = ...
direction_flag = ...
call sample_isotropic_direction( direction_cosx, direction_cosy, direction_cosz )
polarization_cosx = ...
polarization_cosy = ...
polarization_cosz = ...
particle_age = ...
kshort component = ...
delayed_radioactive_decay = ...
call read_phase_space_file( [filename], [energy_unit], [length_unit], phase_space_entry, [sequential_logical_flag], nomore )
particle_code = phase_space_entry%pc
momentum_energy = phase_space_entry%m_e
energy_logical_flag = .true.
coordinate_x = phase_space_entry%x
coordinate_y = phase_space_entry%y
coordinate_z = phase_space_entry%z
direction_cosx = phase_space_entry%u
direction_cosy = phase_space_entry%v
direction_cosz = phase_space_entry%w
particle_weight = phase_space_entry%wei
```

```
debug_logical_flag = .true.
```

```
* debug_lines = ...
```



Source routine – Primary particle

*

particle_code = ...

- By default, the particle type given in the **BEAM** card is taken
- Particle codes explained in FLUKA manual section 5.1
- Possible application: beam made of more than one type of particles

*	<pre>heavyion_atomic_number =</pre>
*	heavyion_mass_number =
*	heavyion_isomer =

- Only used if primary particle is set to HEAVYION or ISOTOPE
- Default values are set on the **HI-PROPE** card, or for ¹²C if the card is missing



Source routine – Energy / momentum

- * momentum_energy = ...
- By default, the particle <u>momentum</u> is expected
- The default value is based on the **BEAM** card (Automatically converted into momentum if energy is given in the **BEAM** card)
- If energy is specified in the source routine, the following logical value must be set .true.

* energy_logical_flag = .true.



Source routine – Energy / momentum

- The momentum divergence set on the **BEAM** card is not retained
- It in necessary to specify it in the source routine
- It is easy with the supplied functions / subroutine

momentum_energy = sample_flat_momentum_energy([min], [max] * momentum_energy = sample_gaussian_momentum_energy([mean], * momentum_energy = sample_maxwell_boltzmann_energy([temperat * momentum_energy = sample_histogram_momentum_energy([filenam * momentum_energy = sample_spectrum_momentum_energy([filename], * call sample_exponential_energy_weight([e_min], [e_max], [in *

Flat spectrum:

Gaussian spectrum:

Maxwell-Boltzmann spectrum:

Histogram sampling:

Spectrum sampling:

Exponential spectrum:



Source routine – Spectrum / histogram sampling

- Both options read external files to determine the probability of different energies
- Histogram sampling requires 3 columns in the input file:
 - Lower energy boundary of the bin
 - Higher energy boundary of the bin
 - Particle intensity per energy unit (dN/dE)
- Spectrum sampling only needs 2 columns:
 - An energy point
 - Particle intensity at the given energy

Between the points the intensity is linearly interpolated.



Source routine – Particle weight

- * particle_weight = ...
- Monte Carlo concept for biased sources
- The default value (particle_weight = 1.0) is usually sufficient
- Note: The exponential spectrum sampling subroutine uses variable particle weight



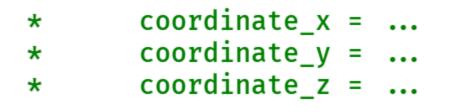
Source routine – Beam divergence

* divergence_x = ...
* divergence_y = ...

- By default:
 - values are taken from the **BEAM** card
 - It is assumed to be a flat angular distribution
- For Gaussian divergence the following logical value must be set .true.
 - * gaussian_divergence_logical_flag = .true.



Source routine – Beam starting position



- By default, values are taken from the **BEAMPOS** card
- Beam shape set in the **BEAM** card, and
- extended sources specified in additional **BEAMPOS** cards are not implemented



Source routine – Beam starting position

Some predefined routines (2 functions and 1 subroutine) are already available:
 Flat distribution:

* coordinate_[a] = sample_flat_distribution([min], [max])

Gaussian distribution:

* coordinate_[a] = sample_gaussian_distribution([mean], [fwhm])

Annular distribution:

* call sample_annular_distribution([rmin], [rmax], coordinate_[a],

Remember the values must be in double precision (1.0D0).

Note: If annular sampling is used, the coordinates have to be selected as well.



Source routine – Beam direction

- * direction_cosx = ...
 - direction_cosy = ...
- * direction_cosz = ...
- By default, values are taken from the **BEAMPOS** card

*

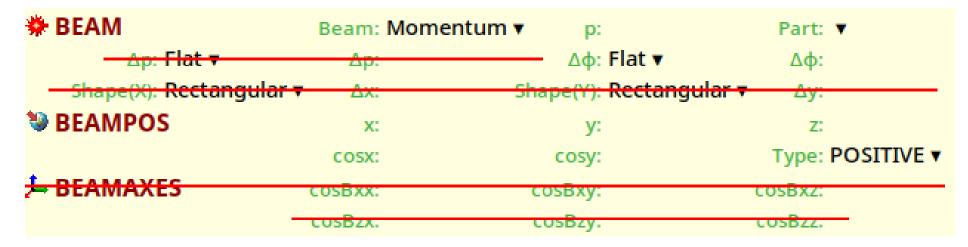
• If the direction_flag is set to: * direction_flag = ...

- 0 : All three values are considered and the they are normalized automatically (Default)
- 1 : The manually set value of the z direction is disregarded. Instead, it is calculated from the x and y direction cosines with a positive sign.
- 2 : As with option 1, but negative sign is used.
- A predefined subroutine is are already available for isotropic direction sampling
 - * call sample_isotropic_direction(direction_cosx, direction_cosy, direction_cosz)



Source routine – Unused values

- It is important to remember, not all values used in the FLUKA input are used in the source routine:
 - The beam momentum distribution
 - The shape of the extended beam / volumetric sources
 - The separate coordinate system set up for the beam



• If one of these features is required, it needs to be programmed in the source routine as well by using the available sampling procedures or by custom code.



Source routine – Phase-space sampling

- Used for the second step in a two-step simulation
- It reads a file containing information on individual particles:
 - Particle code
 - Momentum / energy
 - Starting coordinate
 - Starting direction
 - Weight
- Can replay the particles sequentially, or select from them randomly

*	<pre>call read_phase_space_file([filename], [en</pre>
* *	particle_code
^	momentum_energy - phase_space_entrymm_e
*	<pre>energy_logical_flag = .true.</pre>
*	coordinate_x = phase_space_entry%x
*	<pre>coordinate_y = phase_space_entry%y</pre>
*	<pre>coordinate_z = phase_space_entry%z</pre>
*	direction_cosx = phase_space_entry%u
*	direction_cosy = phase_space_entry%v
*	<pre>direction_cosz = phase_space_entry%w</pre>
*	narticle weight = nhase snace entry%wei



Source routine – Debugging

- To help debug the source routine, the major particle parameters can be printed
- To enable this feature, set

* debug_logical_flag = .true.

- The printed parameters:
 - Energy / momentum
 - Coordinates
 - Direction
 - Weight
- The number of primaries printed can be set with:
 - * debug_lines = 100



SOURCE card and passing parameters

- To invoke a source routine, it is necessary to add a **SOURCE** card
- A **SOURCE** card can be empty or can be used to pass parameters to the routine
 - Max. 18 numerical values (WHASOU (ii)) and 1 string (max. 8 characters) (SDUSOU)

SOURCE	#1:7.	#2: 250.	#3: 12.5
sdum: linksour	#4: 3.75	#5:	#6:
	#7:	#8:	#9:
	#10:	#11:	#12:
	#13:	#14:	#15:
	#16:	#17:	#18:

Good practice:

Even if the beam energy / momentum is defined in the source routine, specify it in the **BEAM** card as it is used for internal initialization. Set a momentum value higher than the maximum possible one.



Sampling from an arbitrary function

- Have the integrable function f(x) as a probability density function
- Calculate the cumulative distribution function (CDF):

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

- Sample a random number (ξ) uniformly between 0 and 1, making $F(x) = \xi$
- Invert the CDF to get $x = F^{-1}(\xi)$



Sampling from an arbitrary function - Example

• Take the following function where $\lambda > 0$:

$$f(x) = \begin{cases} 0, & \text{if } x < 0\\ e^{-x/\lambda}, & \text{if } 0 \le x \end{cases}$$

• The indefinite integral:

$$\int e^{-t/\lambda} dt = -\lambda e^{-t/\lambda} + c$$

• Definite integrals:

$$\int_0^x e^{-t/\lambda} dt = \left[-\lambda e^{-t/\lambda}\right]_0^x = \lambda \left(1 - e^{-x/\lambda}\right)$$
$$\int_0^\infty e^{-t/\lambda} dt = \left[-\lambda e^{-x/\lambda}\right]_0^\infty = \lambda$$



Sampling from an arbitrary function - Example

Cumulative distribution function:

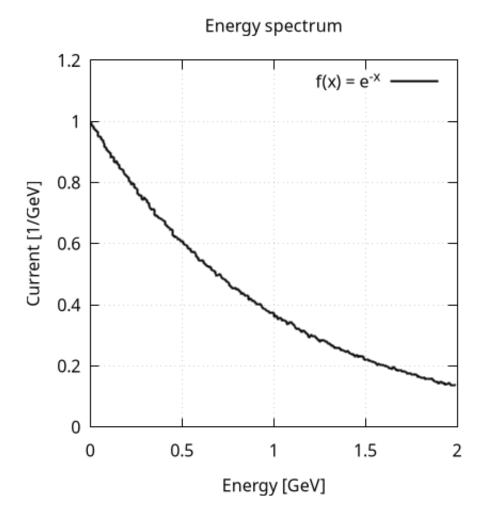
 $F(x) = 1 - e^{-x/\lambda}$

• Sampling a uniformly distributed random number between 0 and 1:

 $1 - e^{-x/\lambda} = \xi$

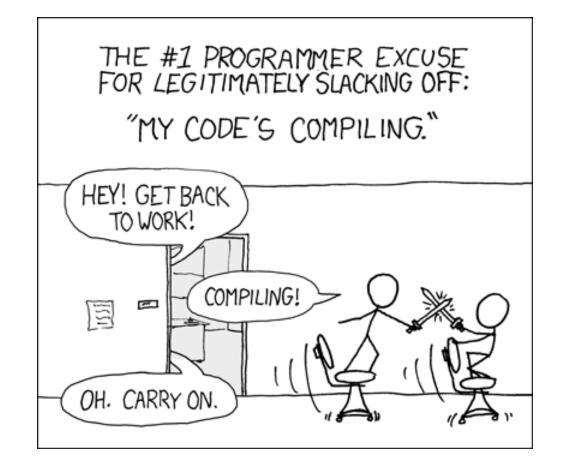
• Inverse function:

 $x = -\lambda \ln \left(1 - \xi \right)$





Time for an exercise!



xkcd.com/303



The source routine

