Simple sources and preprocessor

Outline

1. Simple source definition

- Definition of simple beams
  - Beam parameters
  - Beam visualisation
  - Beam rotation
- Volumetric sources
- Further possibilities

2. The FLUKA preprocessor

1. Simple sources

BEAM & **BEAMPOS** cards | Visualisation | Volumetric sources

Required cards

- In the *Basic input & Flair introduction* lecture we already saw two cards related to defining a beam in FLUKA, namely the **`BEAM`** and **`BEAMPOS`** cards:

BEAM	Beam: Momentum ▾	p:	Part: ▾
Δp: Flat ▾	Δp:	Δϕ: Flat ▾	Δϕ:
Shape(X): Rectangular ▾	Δx:	Shape(Y): Rectangular ▾	Δy:

BEAMPOS	x:	y:	z:
	cosx:	cosy:	Type: POSITIVE ▾

The BEAM card

BEAM		Beam:	Momentum ▾	p:		Part:	▾
Δp:	Flat ▾	Δp:		Δϕ:	Flat ▾	Δϕ:	
Shape(X):	Rectangular ▾	Δx:		Shape(Y):	Rectangular ▾	Δy:	

- The **BEAM** card allows to specify the following parameters:
  - Particle type
  - Momentum or kinetic energy
  - Momentum distribution
  - Angular distribution
  - Shape in the X-Y plane

The **BEAM** card: particle type

BEAM | Beam: Momentum ▾ | p: | Part: ▾
Δp: Flat ▾ | Δp: | Δϕ: Flat ▾ | Δϕ:
Shape(X): Rectangular ▾ | Δx: | Shape(Y): Rectangular ▾ | Δy:

Select particle type from the dropdown menu

Default particle: **PROTON**

Non-standard particles:

- **HEAVYION**: Ion beams heavier than ^{4}He – Requires a **HI-PROPE** card.
- **ISOTOPE**: Radioactive isotope sources – Requires the **HI-PROPE** and **RADDECAY** cards. See the *Activation* lecture

The BEAM card: momentum/energy definition

BEAM	Beam: Momentum ▾	p:	Part: ▾
Δp: Flat ▾	Δp:	Δϕ: Flat ▾	Δϕ:
Shape(X): Rectangular ▾	Δx:	Shape(Y): Rectangular ▾	Δy:

Select **Momentum** or **Energy** from the dropdown menu

Enter the **value** in the input field next to it

Default value: **200 [GeV/c]**

Note: In the case of advanced sources, setting the momentum slightly higher than the maximum momentum used in those sources is **crucial**, since this value is used to initialise the cross section data tables.

(See the *Source routine* and *Advanced sources* lectures)

The BEAM card: momentum and angular distributions

BEAM	Beam: Momentum ▾	p:	Part: ▾
Δp: Flat ▾	Δp:	Δϕ: Flat ▾	Δϕ:
Shape(X): Rectangular ▾	Δx:	Shape(Y): Rectangular ▾	Δy:

Momentum distribution types:

- **Flat**: Full width of a rectangular **momentum** distribution centred at beam momentum [GeV/c]
- **Gaussian**: FWHM of a Gaussian **momentum** distribution [GeV/c]

IMPORTANT: This is always momentum distribution, even if **Energy** was selected

Angular distribution types:

- **Flat**: Full width of a rectangular angular distribution centred at the beam axis [mrad]
- **Isotropic**: Isotropic distribution
- **Gaussian**: FWHM of a Gaussian angular distribution centred at the beam axis [mrad]

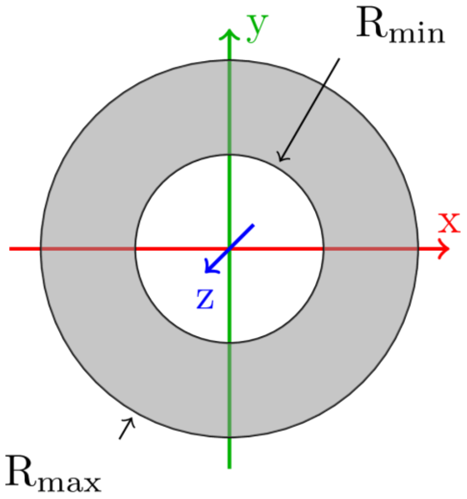
The **BEAM** card: beam shape in the X-Y plane

BEAM		Beam:	Momentum ▾	p:		Part:	▾
Δp:	Flat ▾	Δp:		Δϕ:	Flat ▾	Δϕ:	
Shape(X):	Rectangular ▾	Δx:		Shape(Y):	Rectangular ▾	Δy:	

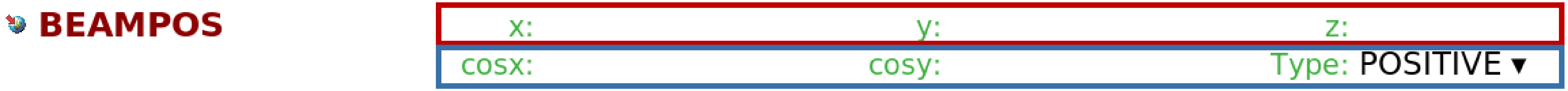
Distribution type:

- **Rectangular**: Full beam width in **x/y** direction centred at the beam axis [cm]
- **Gaussian**: FWHM of a Gaussian distribution in **x/y** direction centred at the beam axis [cm]

  Default: **Δx** = 0.0, **Δy** = **Δx**
- **Annular** distribution can be selected in the dropdown menu of **Shape(X)**
  - **Rmin** and **Rmax** are the radii of the distribution
  - The beam particle position is uniformly sampled on the **x-y** plane between **Rmin** and **Rmax**
  - For circular beam use **Rmin = 0.0**



The BEAMPOS card: beam position and direction



Position:

The beam position is defined with its **x**, **y** and **z** coordinates [cm]

Default: Origin of the coordinate system

Direction:

The beam axis is defined via direction cosines with respect to the x and y axes

The third direction cosine (**cosz**) is automatically calculated by FLUKA

Note that this is not enough for an unequivocal direction definition; the sign of **cosz** has to be provided as well. Select **POSITIVE** or **NEGATIVE** from the *Type* dropdown

Default: Positive z direction

Default beam

- What happens if the **BEAM** and **BEAMPOS** cards are not filled in or are missing?
- FLUKA will use the built-in default (***note****: may change in the future*):
  - Protons at 200 GeV/c momentum
  - Pencil beam: No divergence, zero radius
  - Starting from the origin of the coordinate system
  - Directed along the positive z axis

- This is almost never what you want!
- Always complete the relevant information in the **BEAM** and **BEAMPOS** card
- It is good practice to confirm what source you have defined by checking the FLUKA output (see the *Standard output* lecture)

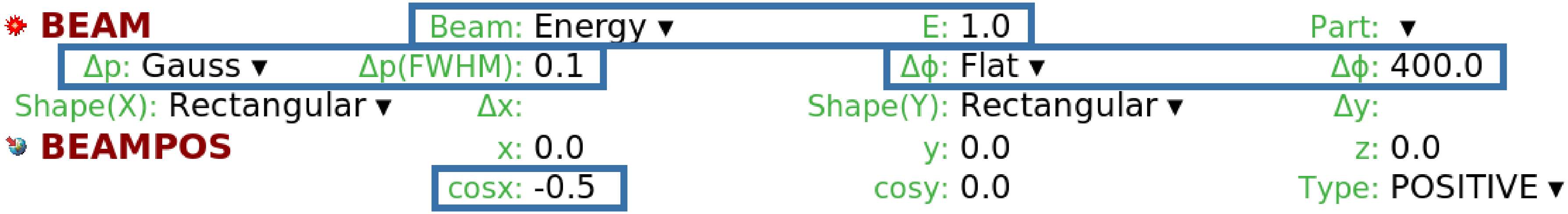
Beam visualisation

- The easiest way to check whether the beam parameters are set correctly is to visualise the beam

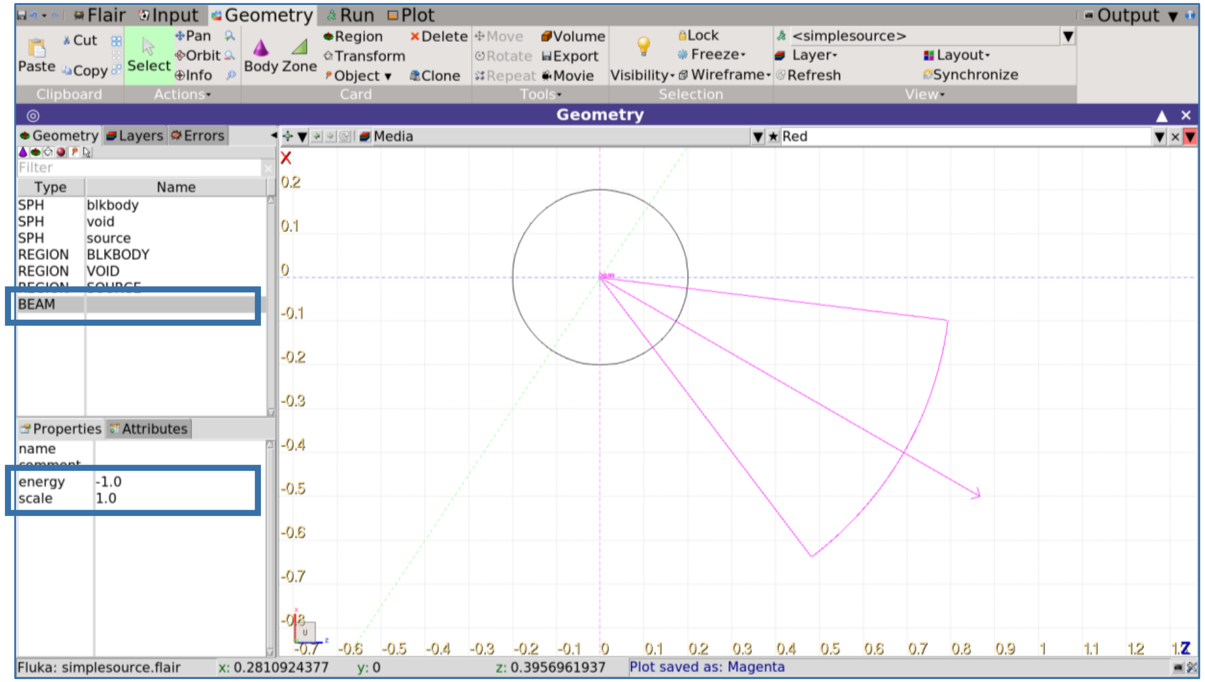
- There are two ways to do this:
  - Use the Geoviewer’s BEAM object
  - Use standard FLUKA scorings (See the *Scoring* lectures) with **BEAMPART** as particle type
    - USRBIN for particle location and direction
    - USRBDX for energy spectrum (with a closed surface surrounding the source location)

Beam visualisation

- **Example 1:** 1 GeV Gaussian beam | 0.1 GeV/c FWHM momentum distribution | 0.4 rad flat angular distribution | rotated around the y axis by -30°

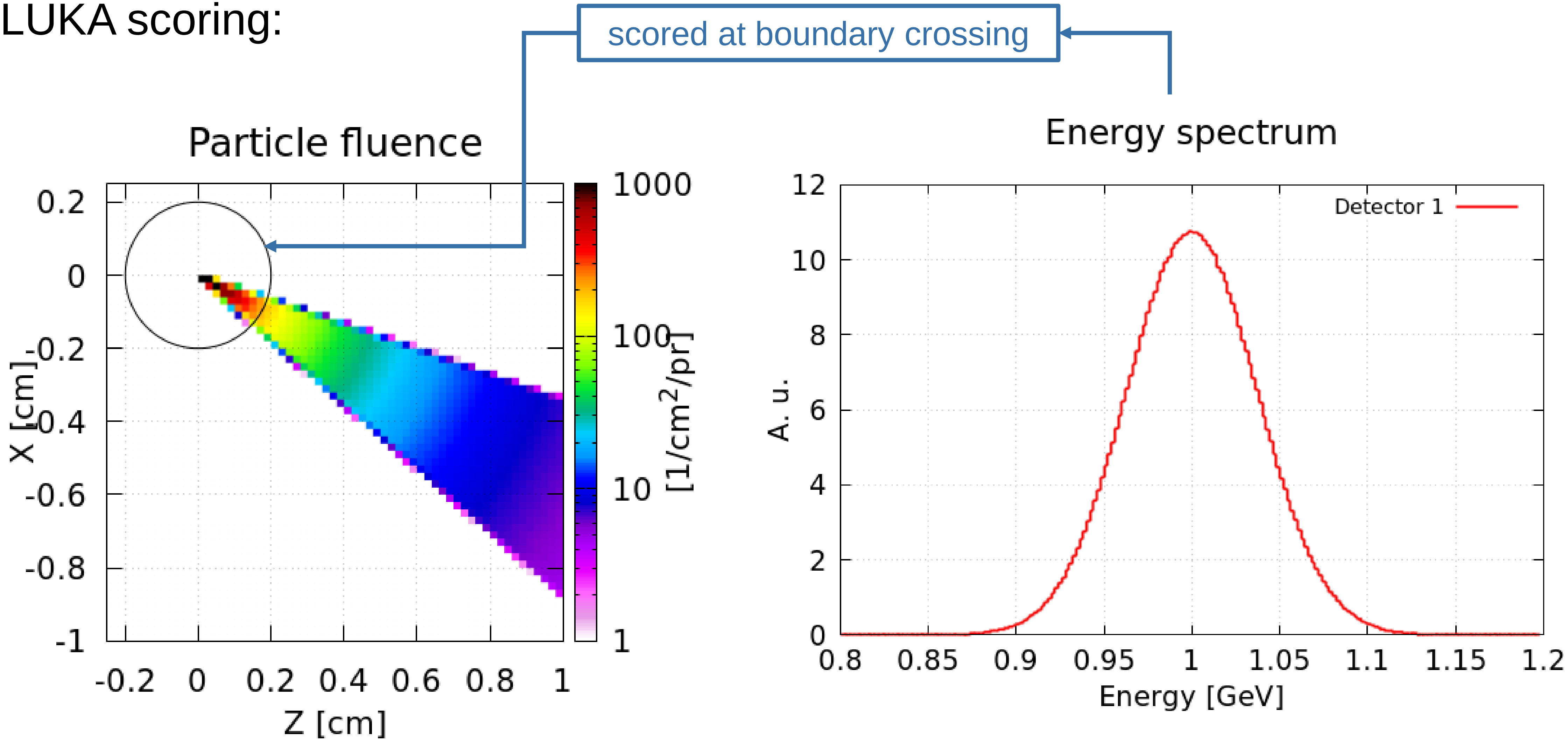


- Geoviewer BEAM object:
  - Starting point
  - Direction
  - Angular distribution
  - Beam mean energy
  - Default scale: 1 GeV(/c) = 1 cm
    Can be changed with the scale parameter



Beam visualisation

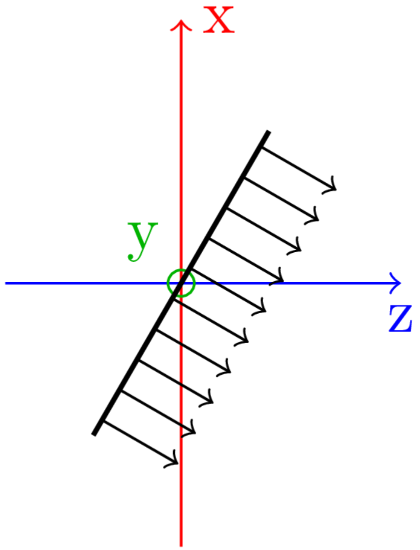
- FLUKA scoring:



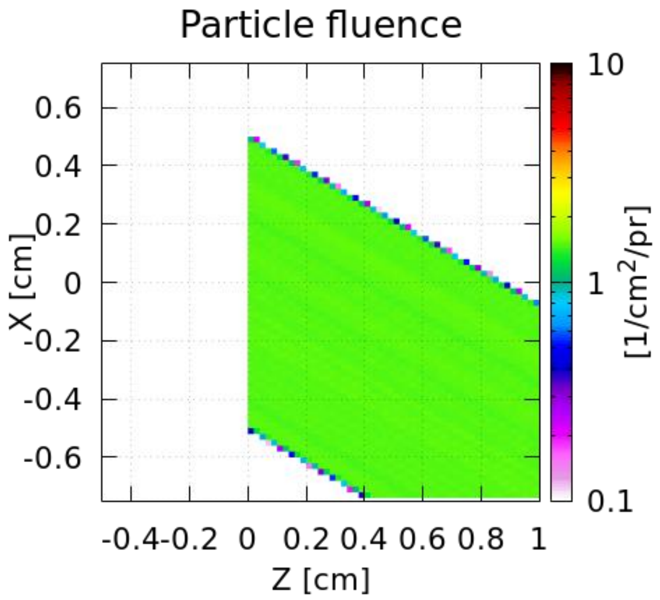
Beam rotation

- **Example 2:** R = 0.5 cm cylindrical beam | zero divergence | rotated around y axis by -30°

Expectation:



Reality:



- *Remember:* the **BEAM** card sets the X-Y shape of the beam, which is not influenced by the beam direction set in the **BEAMPOS** card... so how can we rotate the beam?

Beam rotation

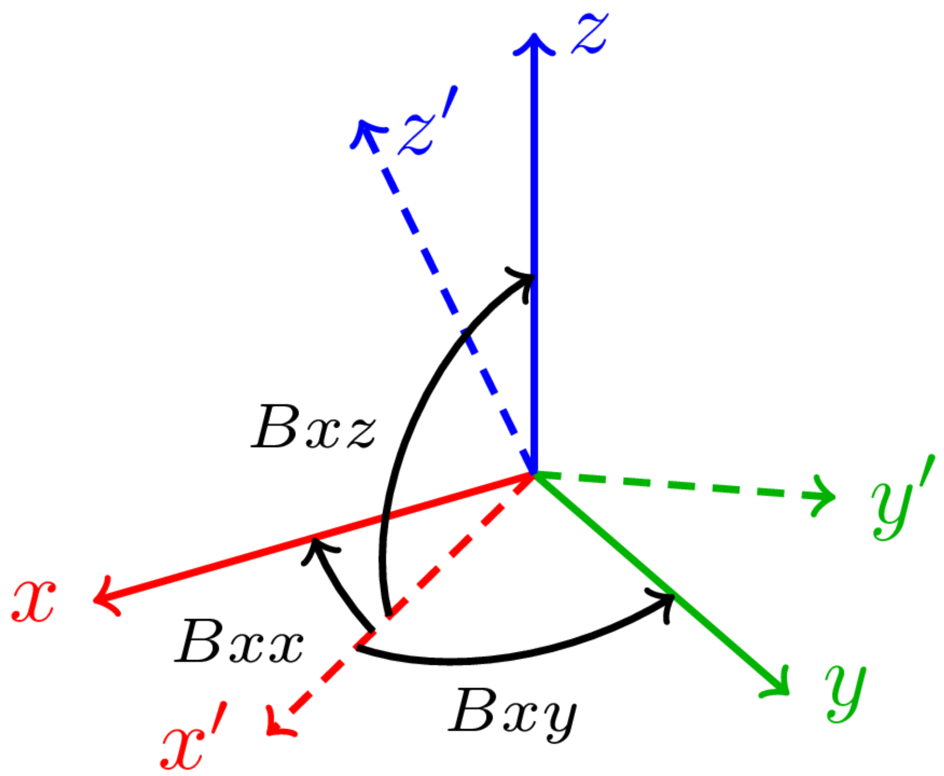
- Input card: **BEAMAXES**

BEAMAXES cosBxx: cosBxy: cosBxz:
cosBzx: cosBzy: cosBzz:

Defines the beam coordinate system (**x′, y′, z′**) with respect to the geometry one (**x, y, z**)

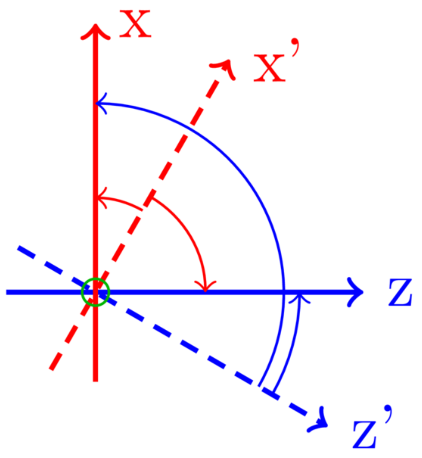
Input fields:

- **cosBxx**: cosine of the angle between **x′** and **x**
- **cosBxy**: cosine of the angle between **x′** and **y**
- **cosBxz**: cosine of the angle between **x′** and **z**
- **cosBzx**: cosine of the angle between **z′** and **x**
- **cosBzy**: cosine of the angle between **z′** and **y**
- **cosBzz**: cosine of the angle between **z′** and **z**

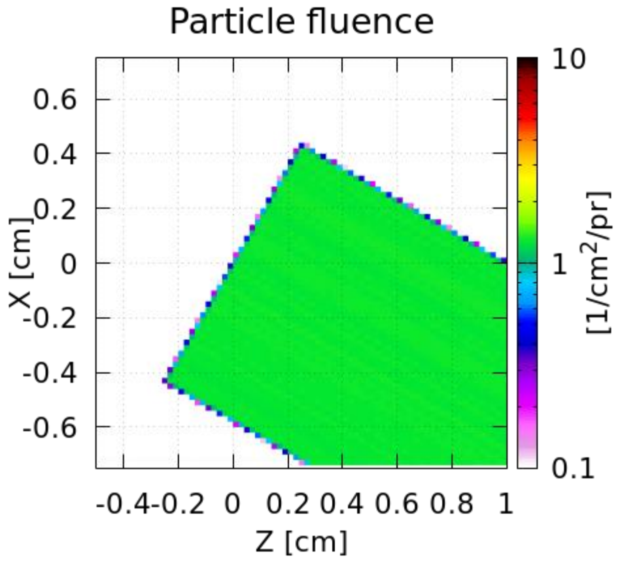


Beam rotation

- **Example 2:** R = 0.5 cm cylindrical beam | zero divergence | rotated around y axis by -30°



WHAT	Beam axis	Geo axis	Angle [°]	Cos(Angle)
cosBxx	x'	x	30	~0.866
cosBxy	x'	y	90	0
cosBxz	x'	z	-60	0.5
cosBzx	z'	x	120	-0.5
cosBzy	z'	y	90	0
cosBzz	z'	z	30	~0.866



BEAMAXES cosBxx: 0.86602540378 cosBxy: 0.0 cosBxz: 0.5
cosBzx: -0.5 cosBzy: 0.0 cosBzz: 0.86602540378

Volumetric sources

Volumetric sources can be defined with a second **BEAMPOS** card:

- Available types:
  - Spherical shell (**SPHE-VOL**)
  - Cylindrical shell (**CYLI-VOL**)
  - Cartesian shell (**CART-VOL**)
  - Spherical surface (**FLOOD**)

- Volumetric sources are centred around the position defined in the first **BEAMPOS** card
- The location inside the volume is sampled uniformly
- The particle direction and angular distribution set in the first **BEAMPOS** card and the **BEAM** card are still applied
- **Warning:** The spatial distributions specified in the **BEAM** card will be disregarded

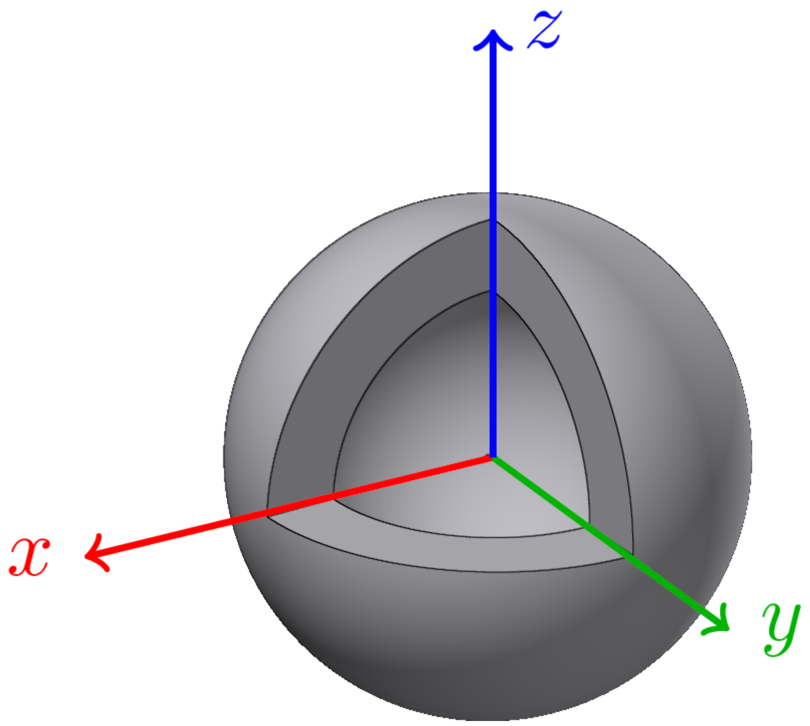
Volumetric sources – *Spherical shell*

BEAMPOS Rin: Rout: Type: SPHE-VOL ▾

Specifies a spherical shell shaped source

Input fields:

- **Rin**: Inner radius [cm]
- **Rout**: Outer radius [cm]



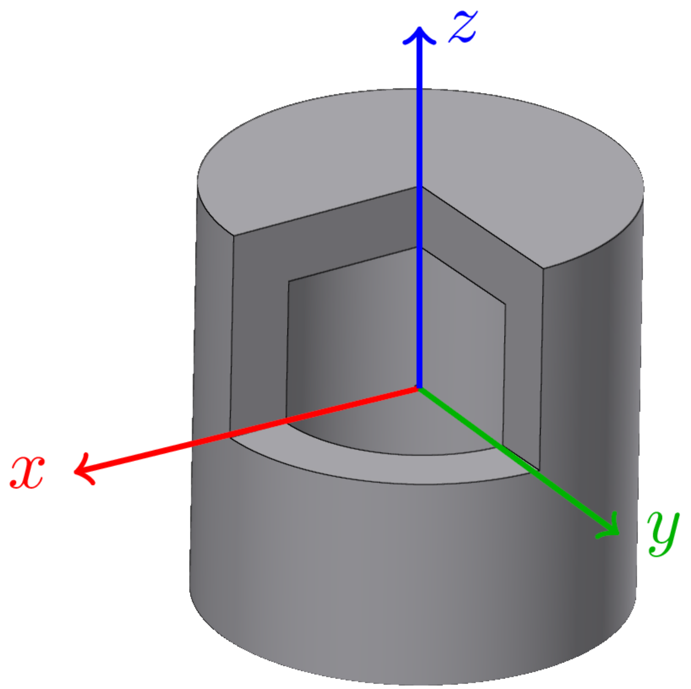
Volumetric sources – *Cylindrical shell*

BEAMPOS Rin: Rout: Type: CYLI-VOL ▾
Hin: Hout:

Specifies a cylindrical shell shaped source around the **z** axis of the geometry

Input fields:

- **Rin**: Inner radius [cm]
- **Rout**: Outer radius [cm]
- **Hin**: Inner height [cm]
- **Hout**: Outer height [cm]



Note: The reference coordinate system can be changed with the **BEAMAXES** card

Volumetric sources – *Cartesian shell*

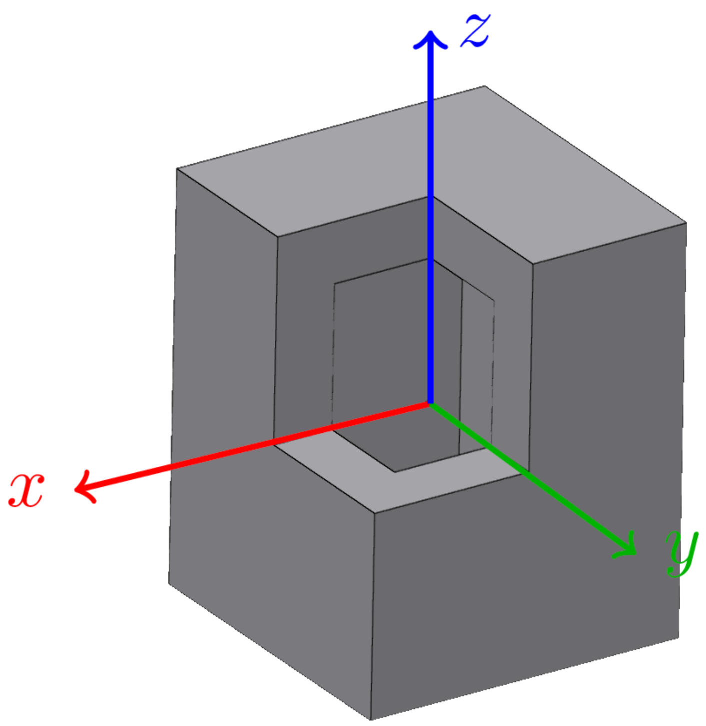
BEAMPOS Xin: Xout: Type: CART-VOL ▾
Yin: Yout:
Zin: Zout:

Specifies a Cartesian shell shaped source along the axes of the geometry

Input fields:

- **Xin & Xout**: Inner & outer length of the **x**-sides
- **Yin & Yout**: Inner & outer length of the **y**-sides
- **Zin & Zout**: Inner & outer length of the **z**-sides

Note: The reference coordinate system can be changed with the **BEAMAXES** card



Volumetric sources – *Spherical surface source*

BEAMPOS R: Type: FLOOD ▾

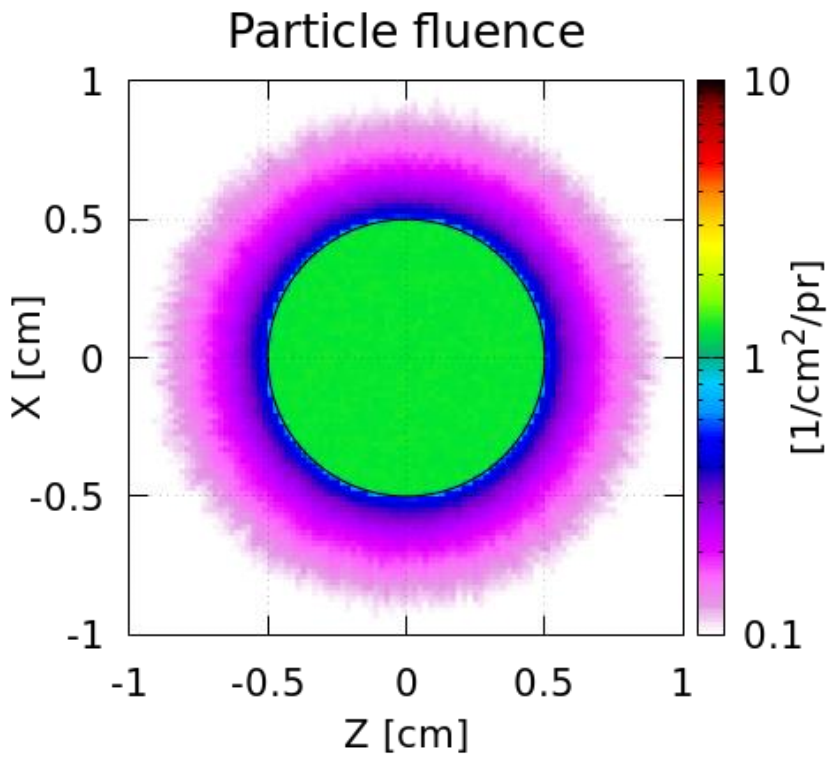
Specifies a spherical surface source in a way that the fluence inside the sphere is **uniform** and **isotropic**

The value of the generated fluence is: $\frac{1}{\pi R^2}$ cm^{-2}

Input fields:

- **R**: Radius of the sphere [cm]

Warning: The particle direction and angular distribution set on the first **BEAMPOS** and the **BEAM** card are disregarded



Further possibilities

Sometimes the **`BEAM`**, **`BEAMPOS`**, and **`BEAMAXES`** cards are not enough

- Special sources available in FLUKA
  - Colliding beams
  - Synchrotron radiation
  - Cosmic rays
  - Multiple beam spots
  - USRBIN source

  (See the *Advanced sources* lecture)

- Program your own custom sources

  (See the *Source routine* lecture)

2. The FLUKA preprocessor

The FLUKA preprocessor

- A limited, “C”-like preprocessor
- Manipulates the input before execution using directives

- 3 type of directives (starting with: **#**):
  - Definition:

    `#define, #undef`
  - Conditional:

    `#if, #elif, #else, #endif`
  - Include:

    `#include`

FLUKA preprocessor - *Definition*

Directive: #define

- Identifiers without numerical or character value:

  #define identifier_name :

  - Used in conjunction with conditional directives (**#if identifier_name ... #endif**)
  - ***identifier_name*** can be up to 40 character long

- Identifiers with numerical or character value:

  #define identifier_name : value

  - The ***value*** can be used in any other input card by referencing **$identifier_name** and can be up to 40 characters long
  - Can also be used in conjunction with conditional directives

Directive: #undef

#undef identifier_name ▾

- Deletes a previously defined identifier

FLUKA preprocessor - *Definition*

- Identifiers can be *defined* and *referenced* anywhere in the input file
- Example:

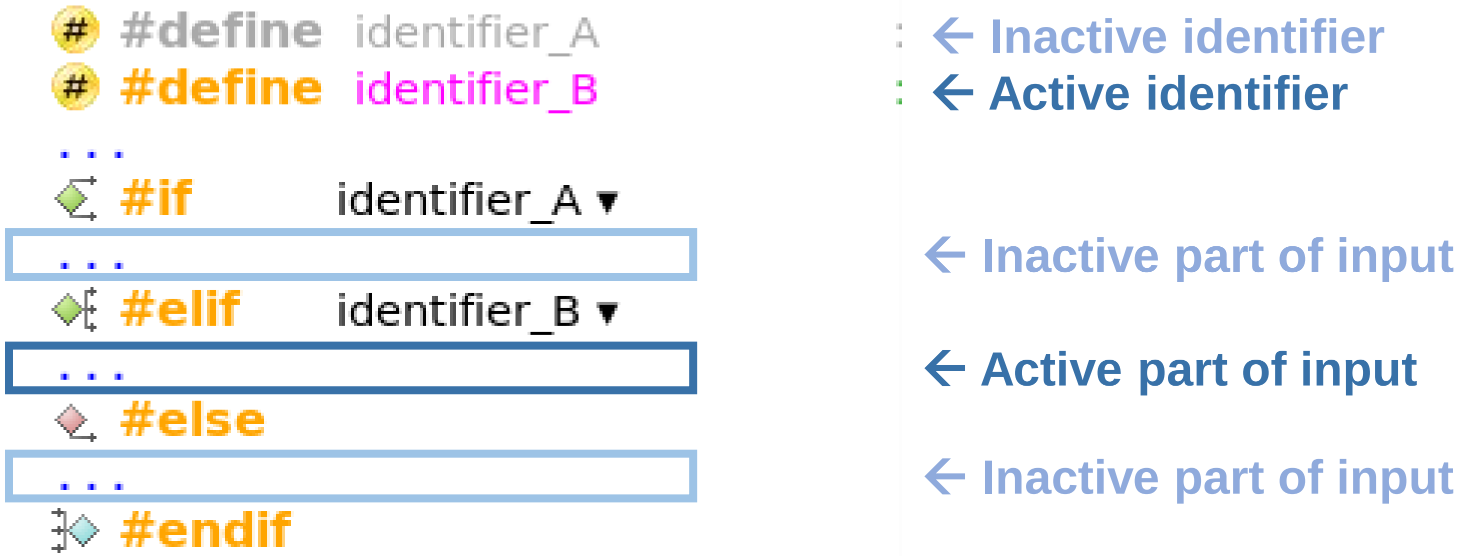
#define Momentum	: 0.1		
#define Particle	: PROTON		
BEAM	Beam: Momentum ▾	p: $Momentum	Part: $Particle ▾
Δp: Flat ▾	Δp:	Δϕ: Flat ▾	Δϕ:
Shape(X): Rectangular ▾	Δx:	Shape(Y): Rectangular ▾	Δy:

- **Note:** It is possible to redefine an identifier with a second `#define` directive
  - This is **NOT RECOMMENDED**
  - If an identifier is redefined, the new value is only applied to cards below
  - The output and error files will contain warning messages

FLUKA preprocessor - *Conditional*

Directives: `#if, #elif, #else, #endif`

- To include or exclude parts of the input in conjunction with the `#define` directives



- They work similarly to any programming language

  Limitation: Cannot compare values, only test if an identifier is defined or not

- Can be nested

  Limitation: Maximum of 10 nesting levels can be used

FLUKA preprocessor - *Conditional*

```
#define  NeutronBeam        :
. . .
#if    NeutronBeam ▾
    BEAM            Beam: Energy ▾            E: 0.1            Part: NEUTRON ▾
    Δp: Flat ▾        Δp:                    Δϕ: Flat ▾          Δϕ:
    Shape(X): Rectangular ▾   Δx:     Shape(Y): Rectangular ▾    Δy:
#else
    BEAM            Beam: Energy ▾            E: 0.05           Part: PHOTON ▾
    Δp: Flat ▾        Δp:                    Δϕ: Flat ▾          Δϕ:
    Shape(X): Rectangular ▾   Δx:     Shape(Y): Rectangular ▾    Δy:
#endif
. . .
#if    NeutronBeam ▾
    ASSIGNMA        Mat: CONCRETE ▾       Reg: SHIELD ▾       to Reg: ▾
               Mat(Decay): ▾              Step:                Field: ▾
#else
    ASSIGNMA        Mat: LEAD ▾           Reg: SHIELD ▾       to Reg: ▾
               Mat(Decay): ▾              Step:                Field: ▾
#endif
```

FLUKA preprocessor - *Conditional*

```
#define  NeutronBeam          :
...
#if    NeutronBeam ▾
    #define  Energy           : -0.1
    #define  Particle         : NEUTRON
    #define  ShieldMa         : CONCRETE
#else
    #define  Energy           : -0.05
    #define  Particle         : PHOTON
    #define  ShieldMa         : LEAD
#endif
...
BEAM                  Beam: Energy ▾         E: $Energy          Part: $Particle ▾
      Δp: Flat ▾        Δp:                  Δϕ: Flat ▾          Δϕ:
Shape(X): Rectangular ▾ Δx:            Shape(Y): Rectangular ▾   Δy:
...
ASSIGNMA              Mat: $ShieldMa ▾       Reg: SHIELD ▾       to Reg: ▾
               Mat(Decay): ▾                 Step:               Field: ▾
```

FLUKA preprocessor - *Include*

Directive: `#include`

`#include <path>/filename.inp`

- Includes the specified file to the input
- Can be nested at multiple levels

- The path can be:
  - Relative to the **main input** file
  - Absolute

- Use cases:
  - Split large input files into multiple smaller ones
  - Reuse same input section (beam definition, scoring, etc.) in multiple input files

FLUKA