Scoring physical quantities I

Introduction to built-in estimators

3D distributions (**USRBIN**) & 1D-2D plots

FLUKA scoring

- It is said that Monte Carlo (MC) is a **“mathematical experiment”**; the MC equivalent of the result of a real experiment (*i.e.*, of a **measurement**) is called an **estimator**

- Just as a real measurement, an estimator is obtained by sampling from a statistical distribution and has a **statistical error** (and in general also a **systematic** one)

- There are often several different techniques to measure the same physical quantity: in the same way, **the same quantity can also be calculated using different kinds of estimators**

- FLUKA offers **numerous different estimators**, *i.e.* **scoring** for various quantities of interest can be requested directly from the input file

Built-in and user scoring

- Several **pre-defined estimators** can be activated in FLUKA; one usually refers to these estimators as **built-in scoring** capabilities
- Users may build their own custom scoring through **user routines**
- However, **built-in scoring**:
  - covers most **common needs**
  - has been **extensively tested**
  - takes **biasing automatically into account**
  - has **refined algorithms** for track subdivision (apportioning)
  - comes with **utility programs** that allow to evaluate statistical errors
- Therefore users are strongly encouraged to **prefer built-in scorings** wherever possible
- Standard scoring can be adapted by means of simple user routines (**`fluscw.f`**, **`comscw.f`**), activated via **`USERWEIG`** card

Definitions

- N : **number of identical particles**

- N_0 : **number of atoms per unit volume**

- λ : **mean free path**, *i.e.* average distance travelled by a particle in a material before an interaction. It depends on the material, particle type and energy

- l : **total distance travelled**

- v : **average particle velocity**

Cross-section

- $\Sigma\ [cm^{-1}] = 1/\lambda\ [cm]$: **macroscopic cross-section**, i.e. probability of interaction per unit distance. It depends on the material, particle type and energy.

- $\sigma = \frac{\Sigma}{N_0} = atom\ effective\ area\ , [barn = 10^{-24} cm^2]$: **microscopic cross-section**, *i.e.*
  - the **area of an atom weighted with the probability of interaction** (hence the name “cross-section”)
  - or the **probability of interaction per unit length, with the length measured in atoms/cm^2**

- The microscopic and macroscopic cross-section have a similar physical meaning of “probability of interaction per unit length”, with length measured in different units. Thus, **the number of interactions** can be obtained by multiplying them by the corresponding particle **track-length**

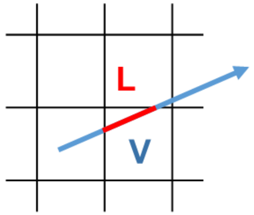
Reaction rate and fluence

- $R = \Sigma\Phi V$: number of reactions in a given time interval inside the volume V (where Φ is the fluence and the product $\Sigma\Phi$ is integrated over energy or velocity)

- $\dot{R} = N\frac{dl}{dt}\Sigma = Nv\Sigma$: **reaction rate**

- $\frac{d\dot{R}}{dV} = \frac{dN}{dV}v\,\Sigma = n(\boldsymbol{r},v)v\Sigma$: reaction rate inside the volume element dV

- $\Phi(\boldsymbol{r},v) = n(\boldsymbol{r},v)dl\,, [cm^{-2}]$: **fluence**, *i.e.* time integral of the flux density
  - Fluence is expressed in **"particles" per cm²** but in reality represents the **density of particle tracks [cm / cm³] !**



Path length **L** [cm] inside the bin divided by the bin volume **V** [cm³] → cm / cm³ = 1/cm²

- $\dot{\Phi}(\boldsymbol{r},v) = n(\boldsymbol{r},v)v\,, [cm^{-3}cm\,s^{-1}] = [cm^{-2}s^{-1}]$: **fluence rate** or **flux density**

FLUKA scoring

What?

Energy deposition and derivatives (dose), fluence or current versus energy, angle or other kinematic variables, time, DPA, residual activity…

Where?

In regions, across boundaries, on region-independent grids

When?

At the end of each cycle or at each event

Output?

Saved in `[inputname]nnn_fort.##` files, where `nnn` is the cycle number & `##` is the logical unit number chosen by the user

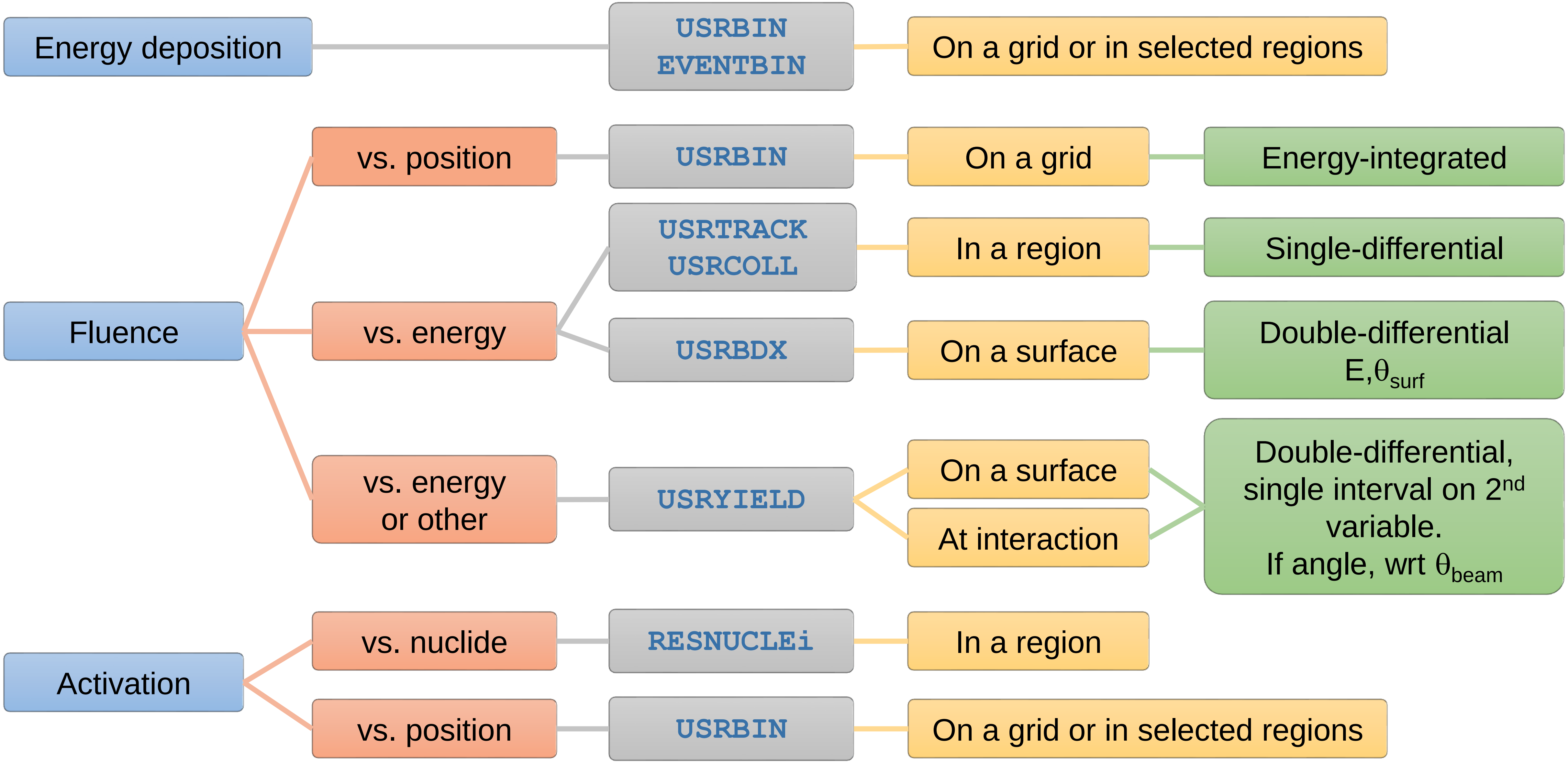
Results?

Post-processing utilities merge cycles, calculate average and rms, provide data files for plotting. Available via **Flair**

Results normalised **per primary**

User code needed for processing of custom scoring!

The FLUKA estimator zoo



Main FLUKA estimators

- **USRBIN** scores the **spatial distribution** of **energy density** or **fluence** (or star* density) in a **selection of regions** or in a **regular mesh** (cylindrical, cartesian) described by the user

in this lecture

- **USRTRACK** (**USRCOLL**) scores average **$d\Phi/dE$ (differential fluence)** of a given type or family of particles in a **given region**
- **USRBDX** scores average **$d^2\Phi/dEd\Omega$ (double-differential fluence or current)** of a given type or family of particles on a **given surface**
- **USRYIELD** scores a **double differential yield** of particles on a **given surface**
  - The distribution can be with respect to energy and angle, but also other more "exotic" quantities

in Scoring II lecture

- All scorings write their results into **logical output units assigned by the user**
  - Unit numbers must be >20 (1-20 reserved by the code)
  - The only exception is SCORE: its output is printed in the **standard output**

***** *A star is a hadronic inelastic interaction*

More “special” scoring cards

- **DETECT** scores energy deposition for each event (primary history) in coincidence or anti-coincidence with a trigger

in Scoring II lecture

- **EVENTBIN** is like **USRBIN**, but prints the binning output after each event instead of an average over histories
- **USERDUMP** allows the user to dump selected information within each primary history
- **TCQUENCH** sets scoring time cut-offs and/or Birks quenching parameters for binnings (**USRBIN** or **EVENTBIN**) indicated by the user
- **ROTPRBIN** assigns rotations/translations for a given user-defined binning (**USRBIN** or **EVENTBIN**) (and sets the storage precision, single or double). Useful with **LATTICE**s

in Geometry II lecture

- **AUXSCORE** defines filters and conversion coefficients
- **RESNUCLEi** scores stopping nuclei in a given region
- **DCYSCORE** assigns cooling times

in Radiation protection lecture

Standard post-processing programs

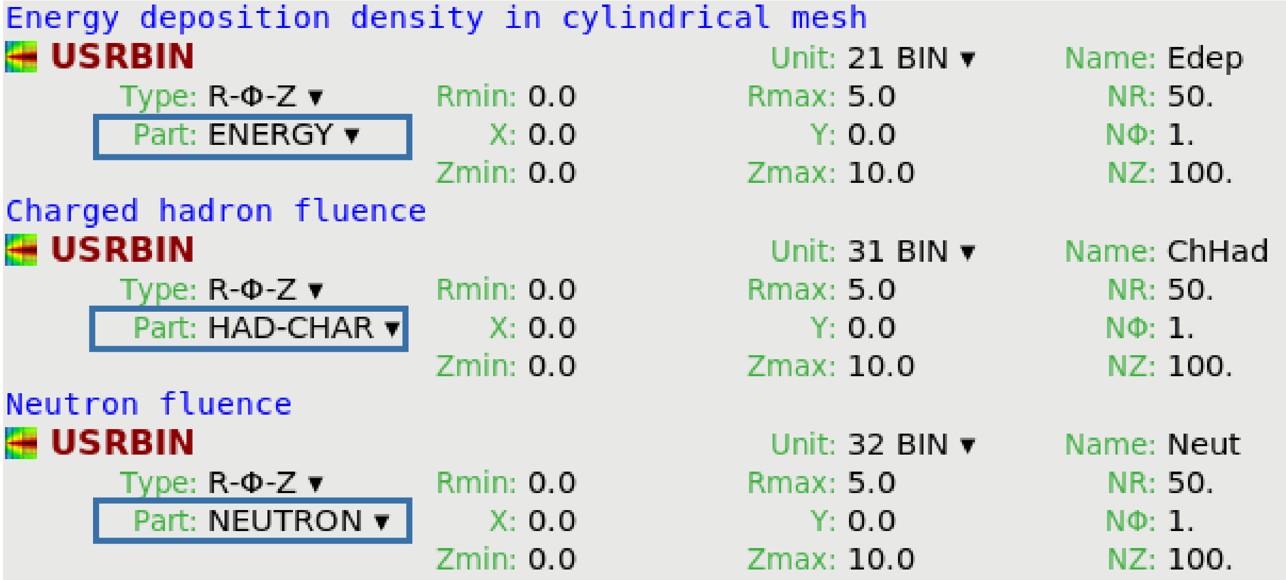
- To analyse the results of the different scoring options, several programs are made available
- **Behind the scenes, Flair uses these programs**
- The executables are in **`/pathtofluka/bin`**, while the sources are available in **`/pathtofluka/src/tools`** in case modifications are needed
- They assume that the estimator files are **unformatted**, and can calculate standard deviations and average values over many cycles:
  - **`ustsuw.f`** to analyse **`USRTRACK`** and **`USRCOLL`** outputs
  - **`usxsuw.f`** to analyse **`USRBDX`** outputs
  - **`usysuw.f`** to analyse **`USRYIELD`** outputs
  - **`usbsuw.f`** to analyse **`USRBIN`** outputs
  - **`usrsuw.f`** to analyse **`RESNUCLEi`** outputs
  - **`usbrea.f`** to convert **`USRBIN`** outputs to ASCII file
- The new fluprocess.py script in **`/pathtofluka/bin`** has been added for easier manual processing

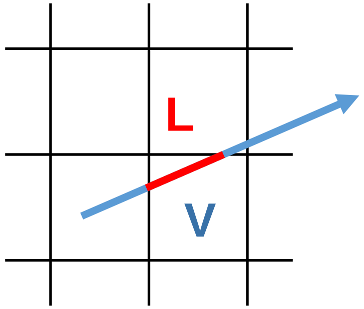
`USRBIN`: Input, visualisation & plotting

USRBIN scoring definition

Energy deposition density
(GeV/cm^3 per primary)

Particle fluence
(1/cm^2 per primary)

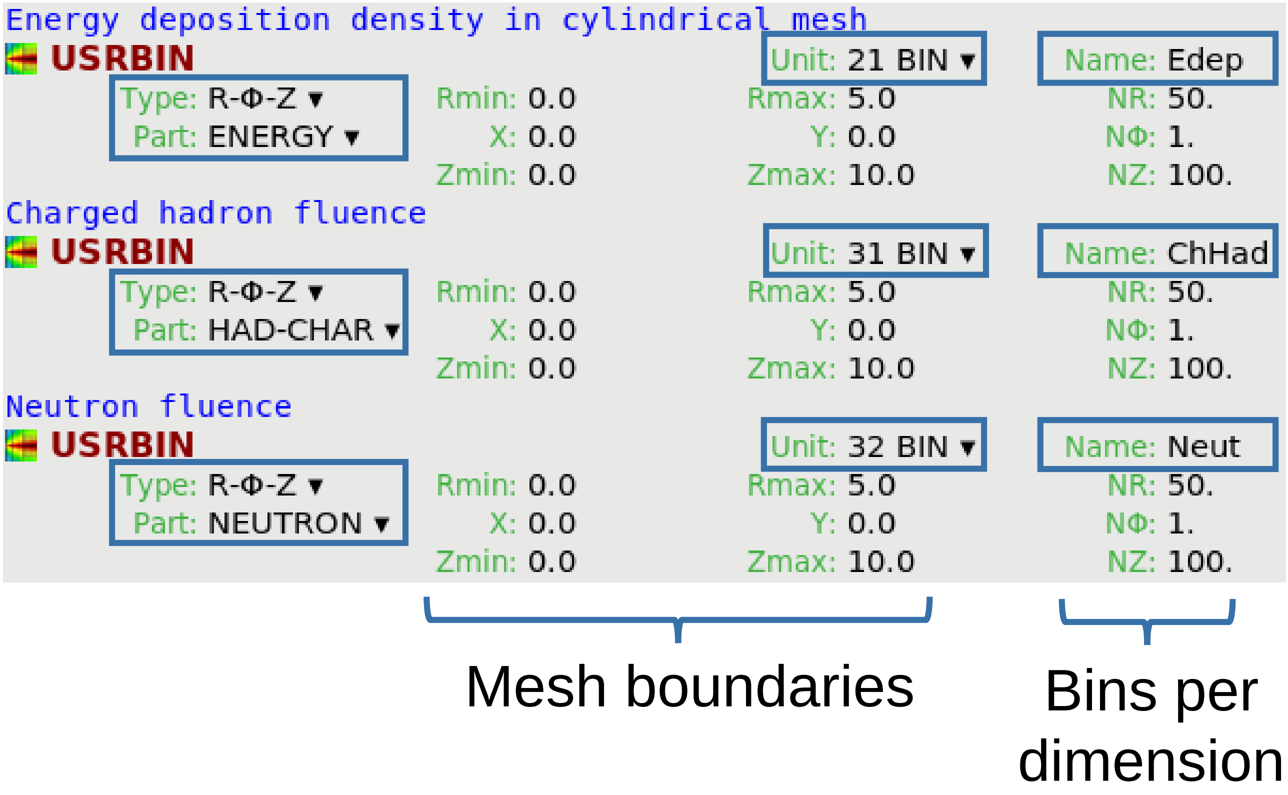




Path length **L** [cm] inside the bin divided by the bin volume **V** [cm^3] → cm / cm^3 = 1/cm^2

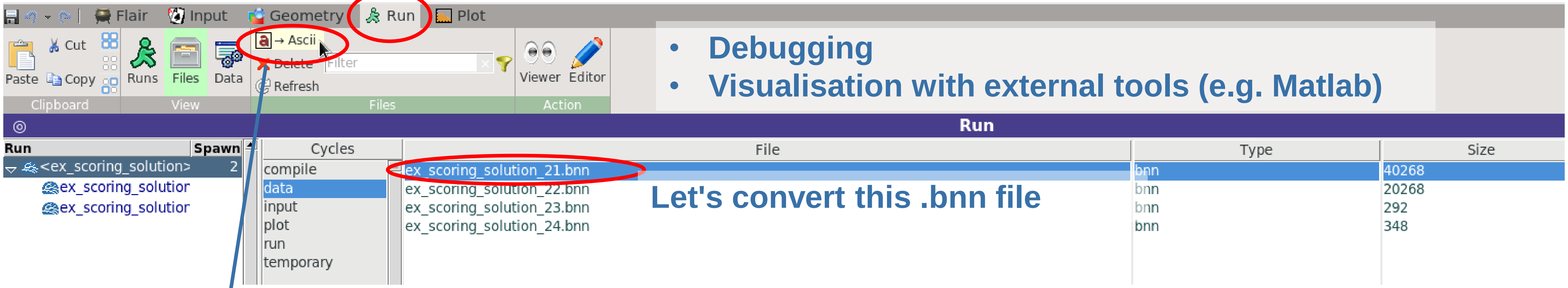
USRBIN input card

- **Type**:
  - X-Y-Z: cartesian mesh
  - R-Φ-Z: cylindrical mesh
  - per region
  - …
- **Part**: generalised particle
- **Unit**: logical output unit
  - **BIN** (binary): unformatted output
    Can be converted to ASCII
    or directly post-processed via Flair.
  - **ASC** (ASCII): formatted output
    Cannot be post-processed via Flair.
- **Name**: 8-character limit



Note on "Type":
Do not choose the **"point"** versions,
which do not return e.g. fluence when a particle is selected,
but density of stars produced by the particles.

Convert USRBIN .bnn file to ASCII (3D mesh data)



- Debugging
- Visualisation with external tools (e.g. Matlab)

Let's convert this .bnn file

Result: bnn.lis file
Always 10 columns per line

```
1
   R - Z     binning n.   1  "EneSmall   " , generalized particle n.  208
      R coordinate: from  0.0000E+00 to  5.0000E+00 cm,    50 bins ( 1.0000E-01 cm wide)
      Z coordinate: from  0.0000E+00 to  1.0000E+01 cm,   100 bins ( 1.0000E-01 cm wide)
      axis coordinates: X = 0.0000E+00, Y =  0.0000E+00 cm
      Data follow in a matrix A(ir,iz), format (1(5x,1p,10(1x,e11.4)))

      accurate deposition along the tracks requested
       2.3867E-01  1.8430E-04  1.6286E-05  9.9585E-06  3.3918E-07  2.1157E-06  7.4462E-06  6.5266E-06  6.3637E-06  4.9691E-06
       0.0000E+00  0.0000E+00  1.6428E-06  3.8209E-06  4.4210E-06  3.7473E-06  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00
       0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  1.1102E-06
       1.8253E-08  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00
       0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00
       2.3794E-01  3.4083E-04  5.7607E-05  2.7670E-05  2.7495E-05  2.0694E-05  0.0000E+00  0.0000E+00  0.0000E+00  2.5324E-06
       3.5606E-06  3.5531E-06  1.8965E-06  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00
       0.0000E+00  0.0000E+00  2.7077E-09  0.0000E+00  0.0000E+00  0.0000E+00  6.8063E-08  0.0000E+00  0.0000E+00  0.0000E+00
       0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  6.1228E-08  0.0000E+00  0.0000E+00  0.0000E+00  2.7059E-07  0.0000E+00
       0.0000E+00  0.0000E+00  4.4389E-11  0.0000E+00  1.6188E-10  2.2633E-14  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00
```

R-Z binning: (ir, iz)
First loop on ir,
[then on iphi (if exists),]
then on iz

Cartesian binning: (ix, iy, iz)
First loop on ix,
then on iy,
then on iz

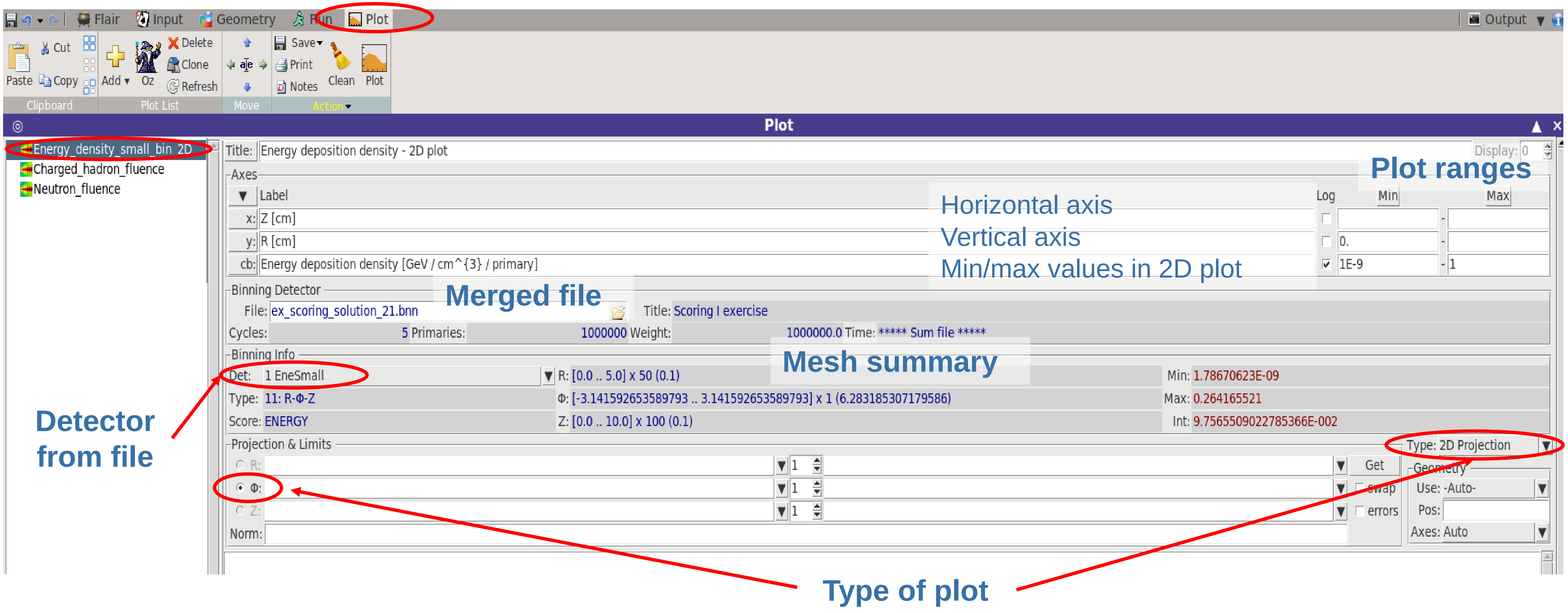
First 50 values (first 5 lines) are $\{val_{(ir,\ iz=1)}\}$, with **ir ∈ ⟦1, 50⟧**. **Next 50 values** are $\{val_{(ir,\ iz=2)}\}$, with **ir ∈ ⟦1, 50⟧**. And so on, **for each iz** ∈ ⟦1, 100⟧.

Volume normalisation

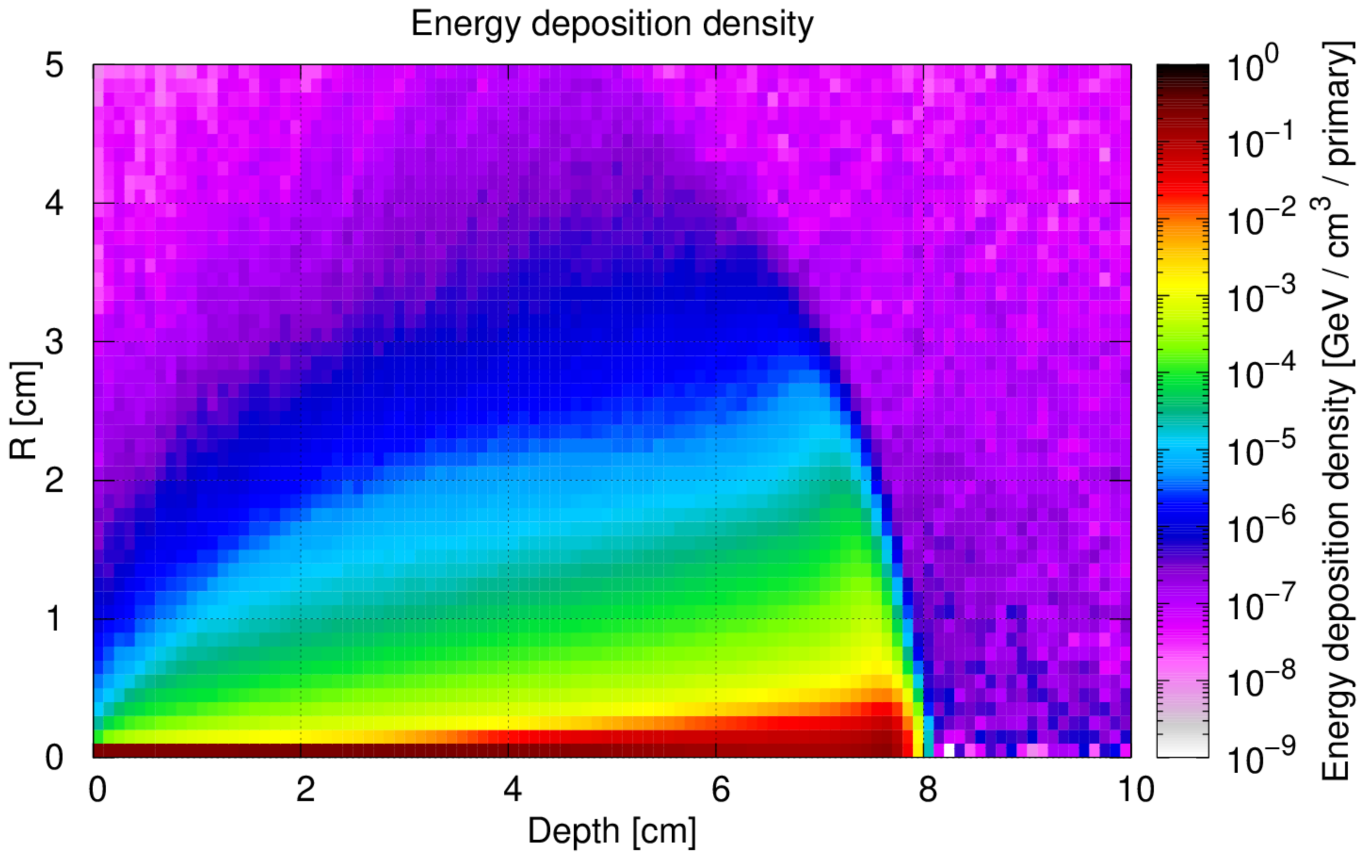
- **USRBIN per region: results are NOT normalised by volume.**
  When scoring particle *fluence* (e.g. NEUTRON) *or energy density* (ENERGY) with USRBIN per region, FLUKA results are *total track-length and energy deposition*, respectively.
  - Indeed, FLUKA does not calculate regions volumes (which can be arbitrarily complex!).
  - Results are provided assuming the region has a volume equal to 1 cm^3 .
  - The user needs to divide by the region volume to obtain the intended quantities.

- **USRBIN meshes: results are normalised by volume.**
  Conversely, when USRBIN scoring on regular (Cartesian, cylindrical) meshes is requested, *particle fluence and energy density* will be automatically provided.
  - Results are already normalised by the bins volume(s).
  - Indeed, FLUKA can easily compute the mesh bins volume(s). In addition, it would not make sense for the user to calculate each cylindrical mesh bin volume (varying radially)!

Plotting – Energy deposition density (USRBIN mesh)

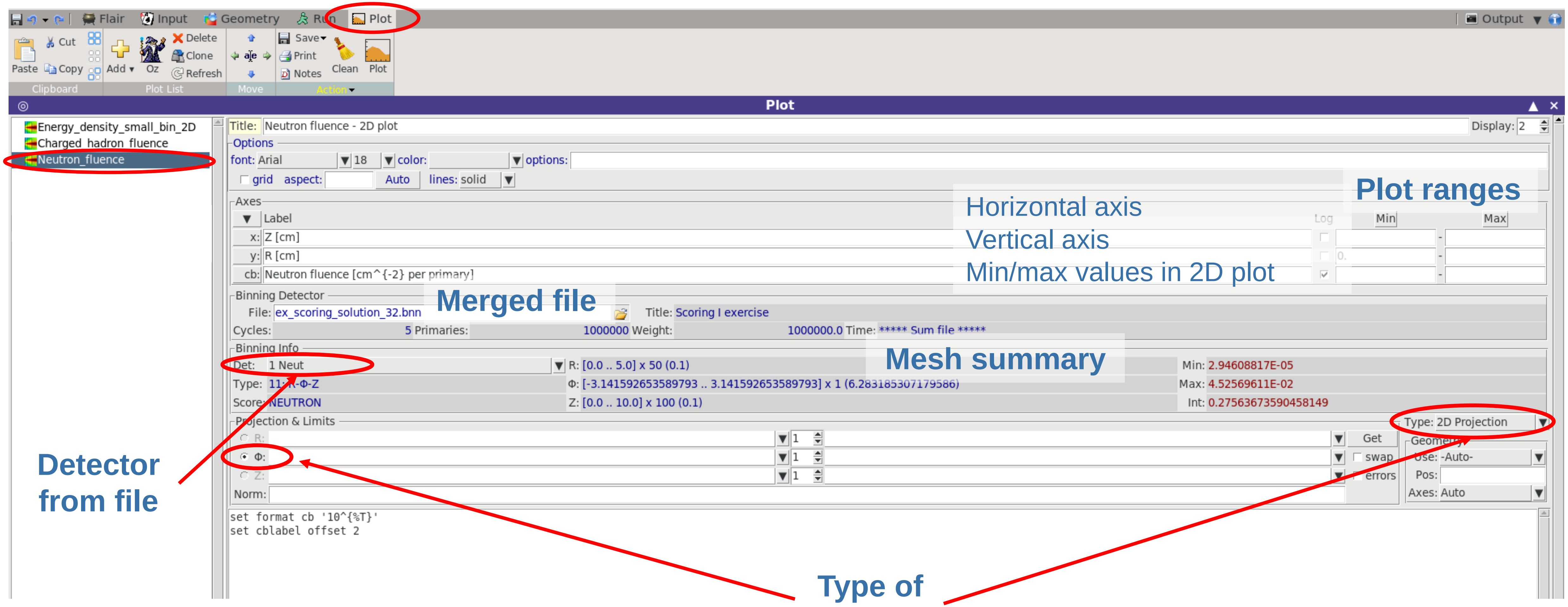


Plot result – Energy deposition density (`USRBIN` mesh)



- This plot is a **2D projection** of a 3D structure
  → **the result is the volume-weighted** average **over the 3rd coordinate** (Φ in this case)
- The 2D limits, and the range on which to perform the average, can be set in Flair

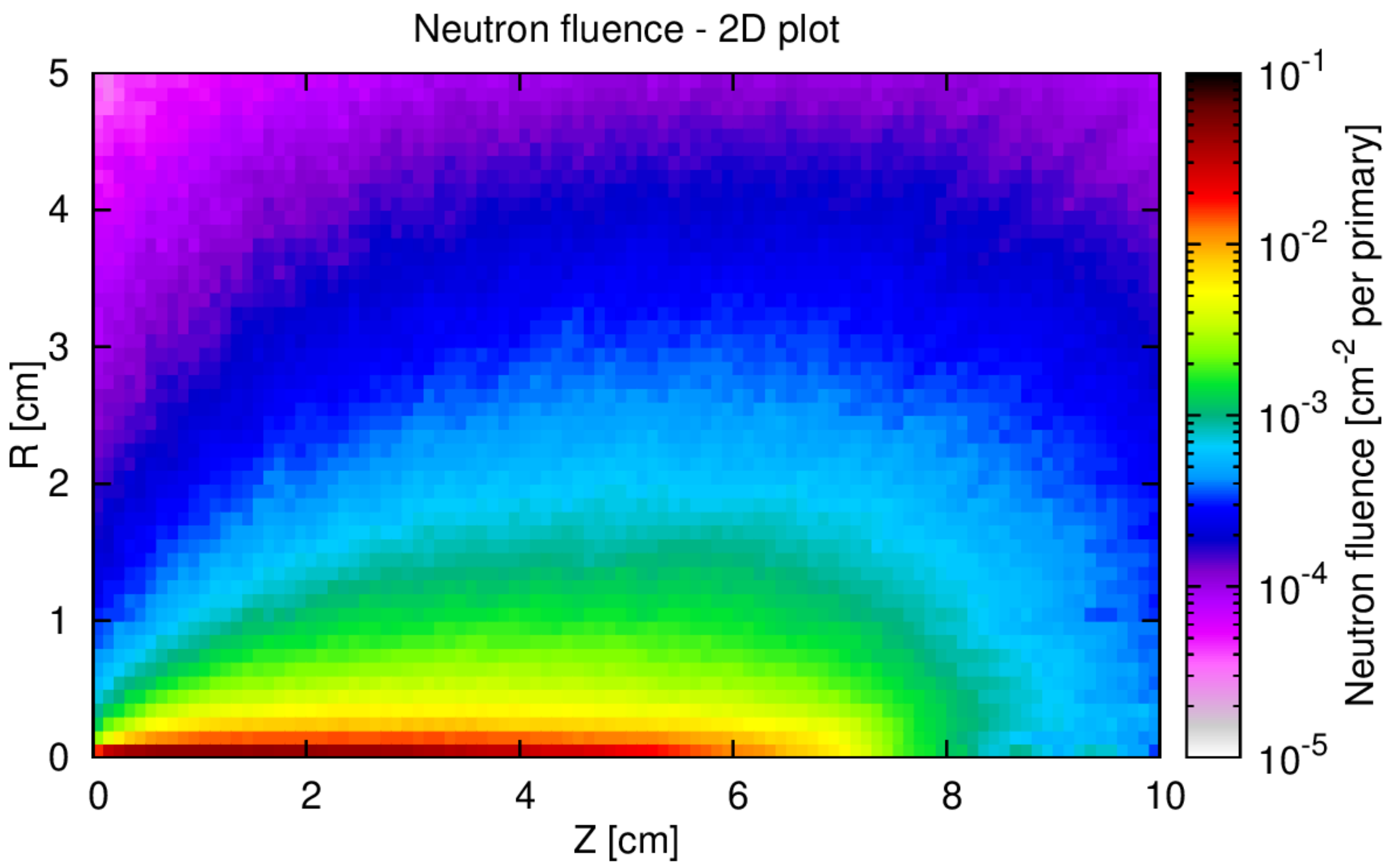
Plot result – Neutron fluence (USRBIN mesh)



Detector from file

Type of plot

Plotting – Neutron fluence (`USRBIN` mesh)



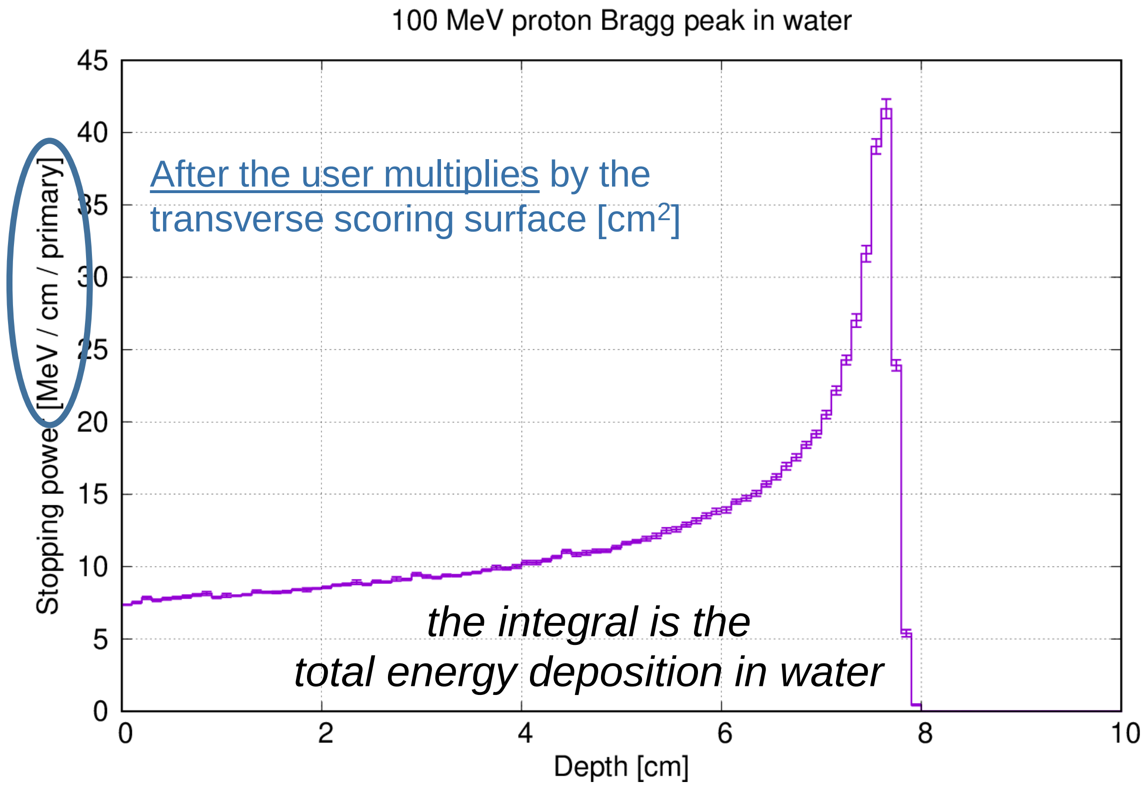
- This plot is a **2D projection** of a 3D structure
  → **the result is the volume-weighted** average **over the 3rd coordinate** (Φ in this case)
- The 2D limits, and the range on which to perform the average, can be set in Flair

One dimensional plots

- 1D plots can be obtained from 3D **USRBIN** meshes, e.g. (for an R-Φ-Z mesh):

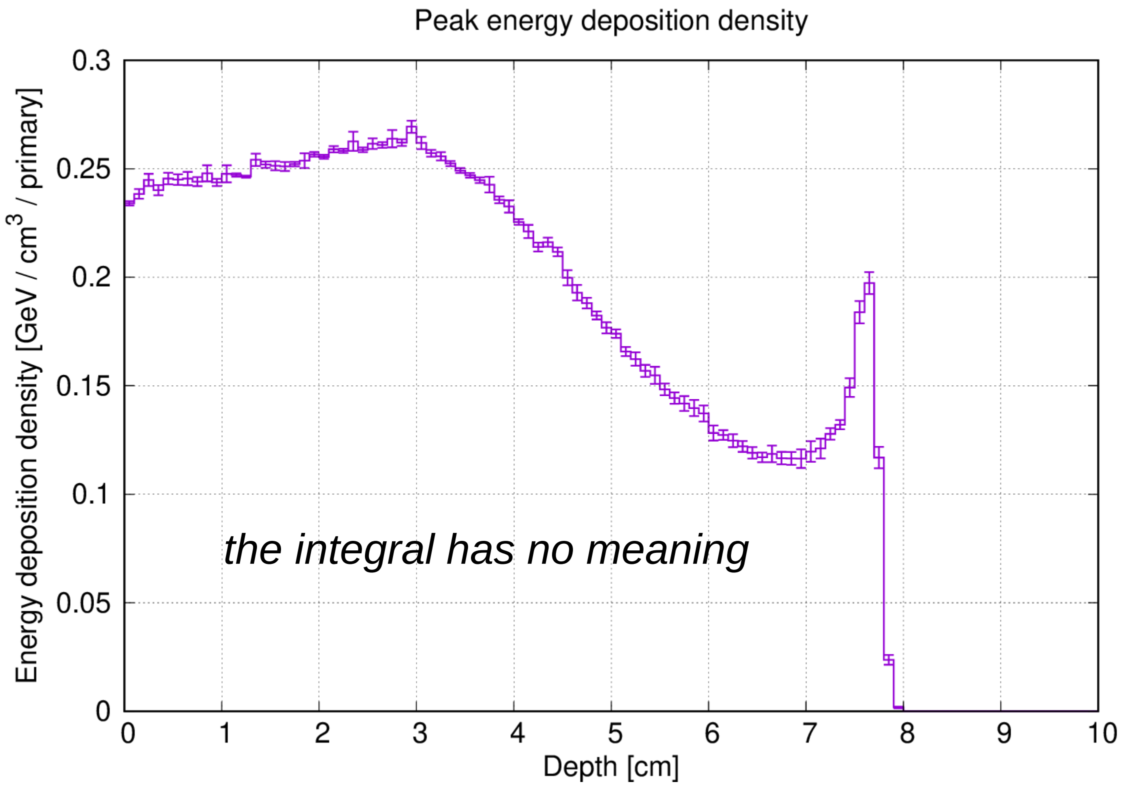
1D Projection

- For each z bin:
  average over all {R, Φ} bins



1D Max

- For each z bin:
  highest over all {R, Φ} bins



USRBIN mesh geometry check

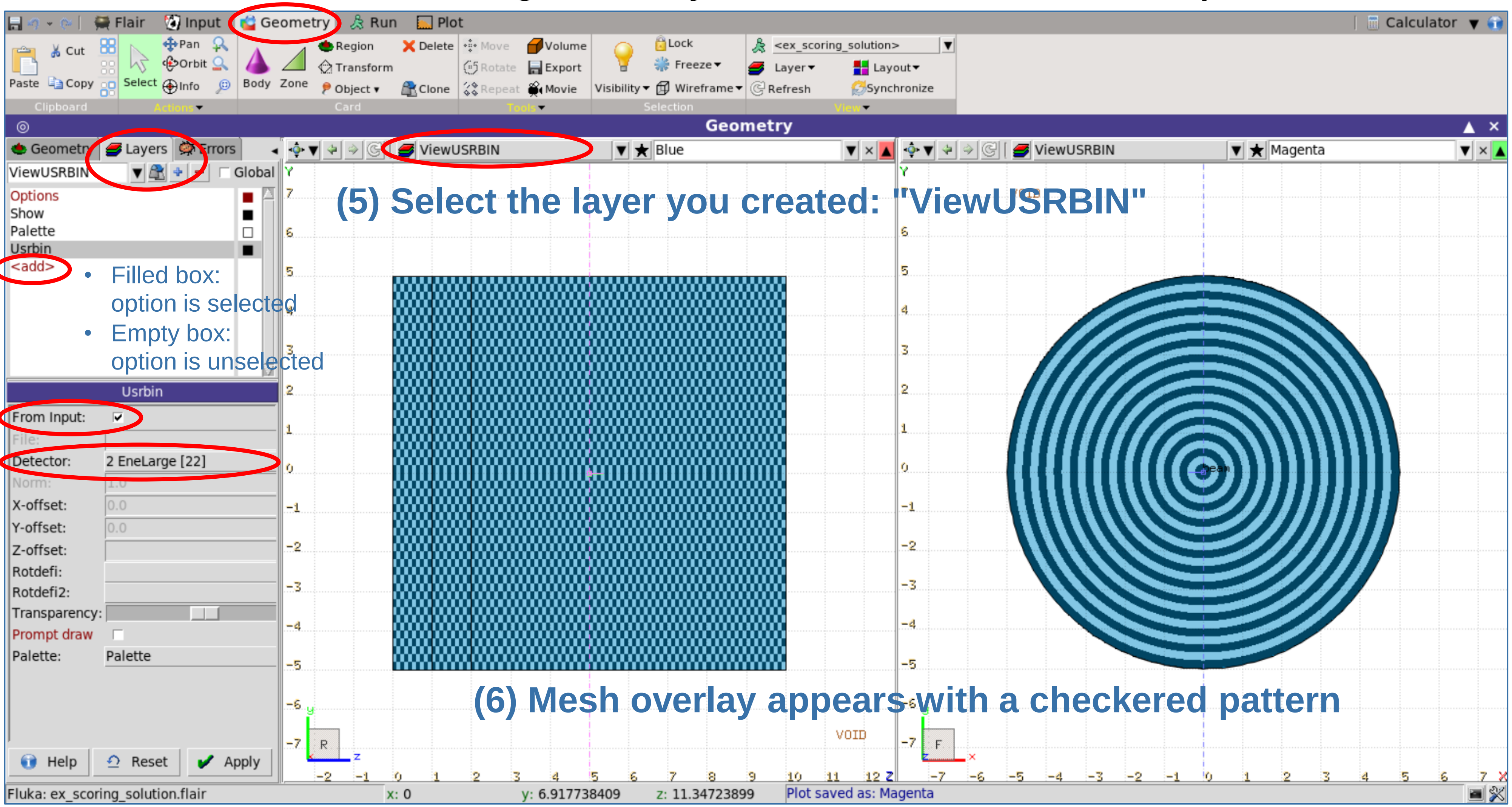
- A defined mesh can be overlaid on the geometry to check that it is well-positioned

(1) Click "+"
to create a new layer.
Here for example,
we name it "ViewUSRBIN".

(2) Click "add"
and select "Usrbin"

(3) Select "From input"

(4) Select the
USRBIN card name
from your input file



- Filled box: option is selected
- Empty box: option is unselected

(5) Select the layer you created: "ViewUSRBIN"

(6) Mesh overlay appears with a checkered pattern

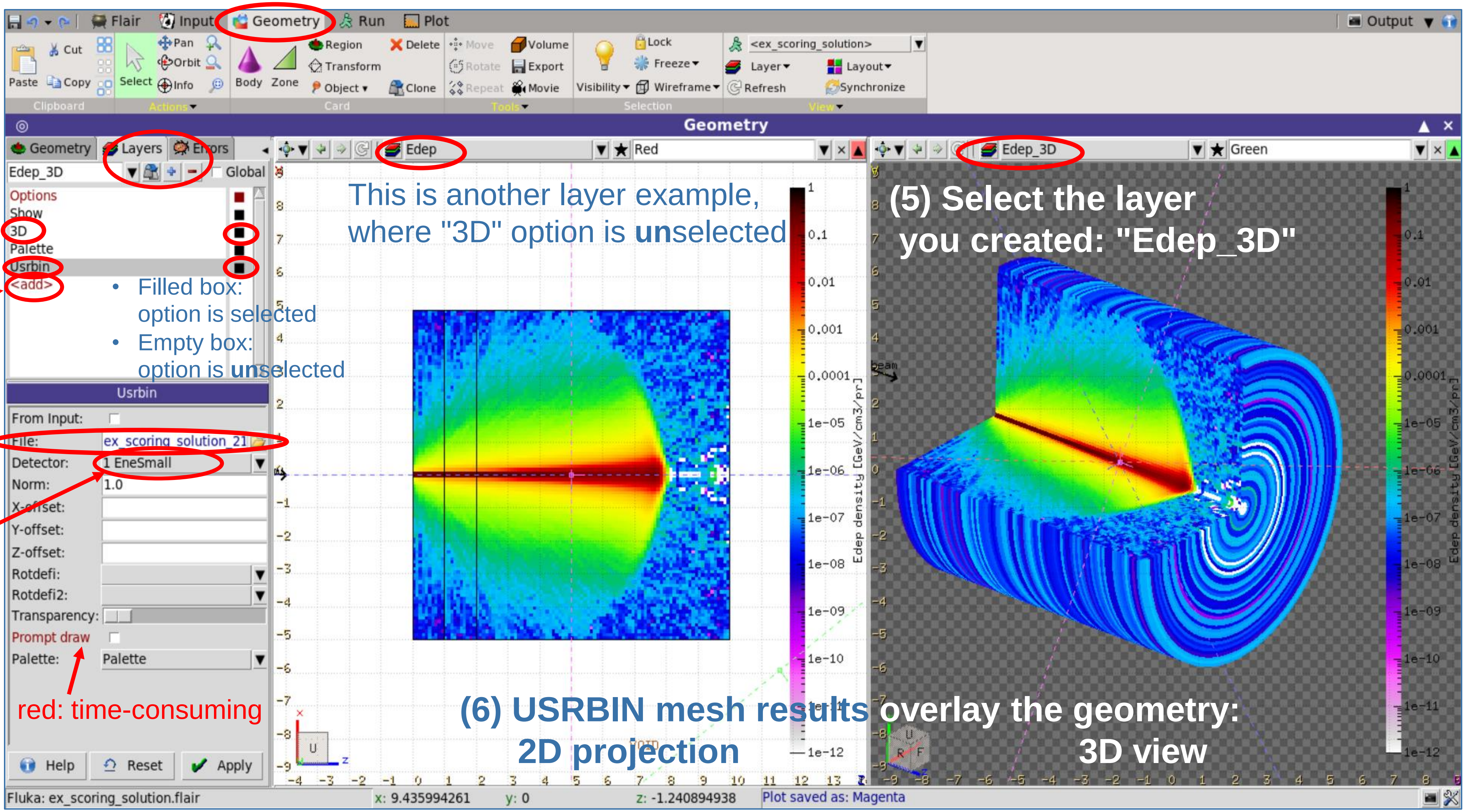
Overlaying USRBIN mesh results on 2D/3D geometry

(1) Click "+" to create a new layer.
Here for example, we name it "Edep_3D".

(2) Click "add" and select "Usrbin". Also add "3D".

(3) Select merged file from run results

(4) Select the USRBIN card name from your input file



FLUKA