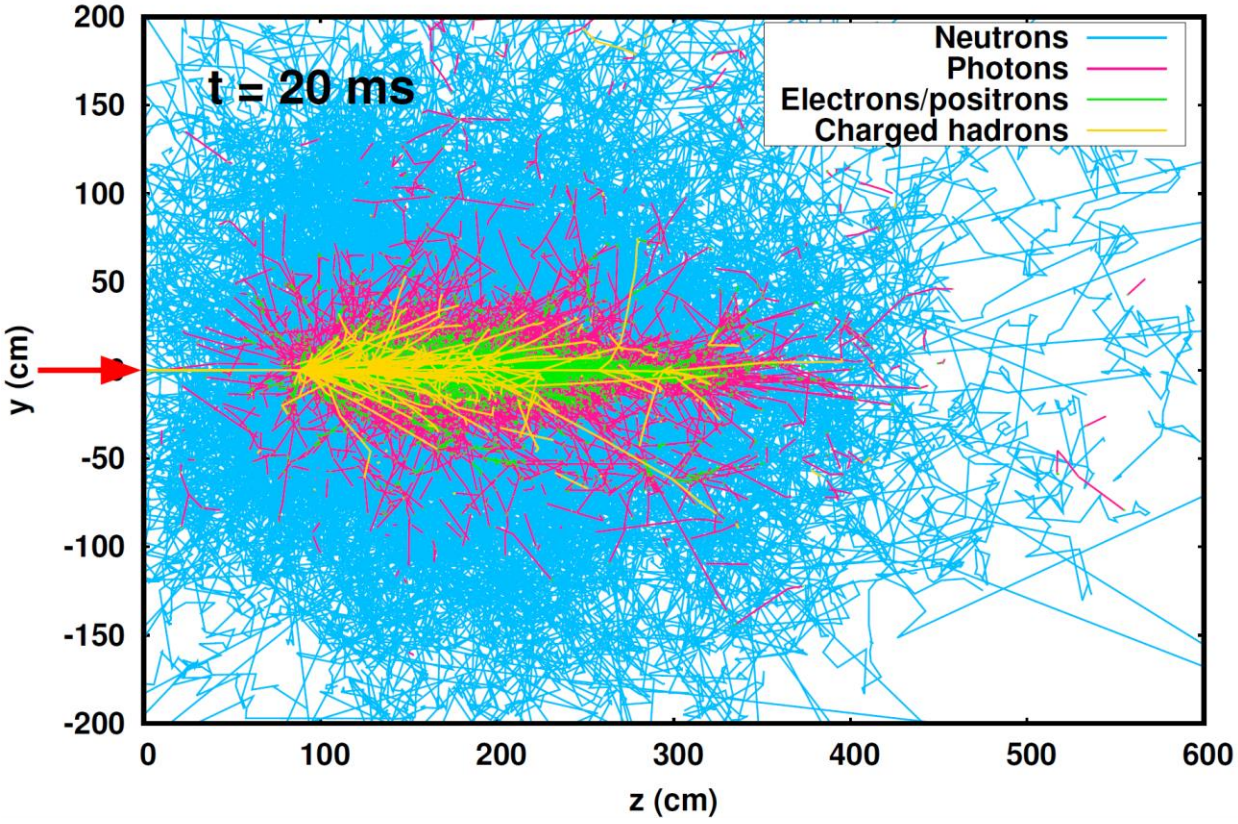
MGDRAW routine

Exercises



Exercise 1: Create a phase-space file

We are interested in the **antiprotons at the ADTrgTi - ADTrgDs border.**

Simulating antiprotons production through the CERN Antiproton Decelerator Target takes a very **significant amount of time**: in the order of CPU-days for e.g. 25 millions primary protons.

Instead of simulating the time expensive antiproton production **every time**, we can do a **two-step simulation**:

- Record the exiting antiprotons (today's exercise).
- Replay the recorded antiprotons as phase-space source (source routine exercise).

In the source routine exercise, we used an external phase-space file as an antiprotons source.

Today, we are going to study how to produce such a phase-space file.

Exercise 1: Create a phase-space file

- We are interested in the dump of produced **antiprotons at the ADTrgTi - ADTrgDs border *in a specific format***: we hence need to customize **BXDRAW** to our needs.

- The format which is expected by the source routine as a phase space file, is documented at: fluka_manual_phase_space_file
  It is reminded below for your convenience.

```
The external phase space file has to contain the following columns
in this order, with the specified variable type:
•Particle code, (integer)
•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)
•Particle weight, (double precision)
```

Exercise 1: Create a phase-space file

- We will focus on the **BXDRAW entry** in the provided `source_routine_mgdraw.f`.

- For your convenience, the **regions names -> integers** conversions are already added.
  Note the use of **geon2r** subroutine.

- Modify the **conditional statement**, in order to **filter antiprotons at the ADTrgTi - ADTrgDs border:**
  - Origin region selection: we want to select 'ADTrgTi '. Use **MREG** and **from_reg** variables.
  - Destination region selection: we want to select 'ADTrgDs '. Use **NEWREG** and **to_reg** variables.
  - Particle selection: we want to select antiprotons. Use the **JTRACK** variable.

- Add the variables of interest to **dump the particles in a format compatible with the read_phase_space_file** source subroutine.
  NB: You need to use arguments of BXDRAW, as well as variables from `trackr.inc`.

Exercise 1: Create a phase-space file

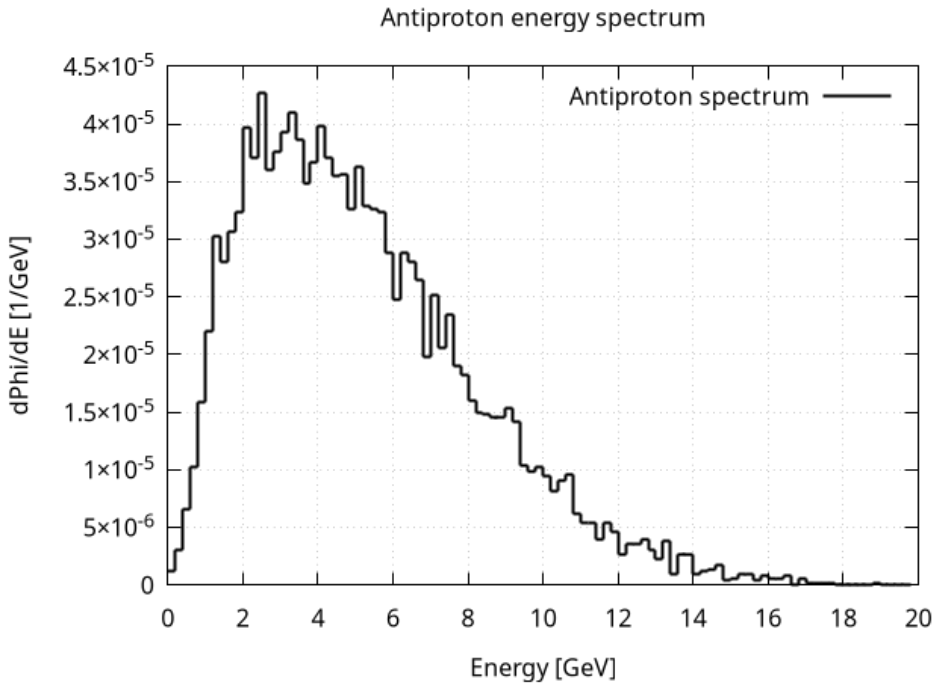
- Use the provided `antiprotons_decelerator_target_mgdraw.flair` (CERN Antiproton Decelerator Target).

- Activate the calls to BXDRAW.
  To do so, add a **USERDUMP** card:
  - Type: Dump
  - File: dump
  - What: Complete
  - Score: Traj&Cont losses (or All)

- Build a **custom executable** with your implementation of `mgdraw.f` (`mgdraw_ex1.f`).

- Launch a test run, then **run** 2 cycles with 5000 primaries each.

- Open one output file. Where is the output file unit set?
  Check that the antiprotons are dumped in the same format
  as in the phase-space file you used in the first source routine exercise.



Exercise 2: Plotting trajectories with Flair

We want to visualize the particles trajectories within an event.
We are going to do so in the following use case: 450 GeV protons on an Al target.

MGDRAW allows the **dump of trajectories information**, in a format **compatible with Flair**.

We are going to rely on the **default version of MGDRAW**: no need to modify any code within `mgdraw.f`.

- Open the provided input file (`trajectories.flair`).

- **Activate MGDRAW calls**.
  To do so, add a `USERDUMP` card to your project, with trajectories dump enabled ("Traj&Cont losses").

  NB: It is advised to use a file name containing the word "dump", so that Flair can detect it easily later on (when the dump file is used in the Geometry tab).

Exercise 2: Plotting trajectories with Flair

- Note that **we do not need a custom build**, because we are relying on the **default version** of `mgdraw.f`.

- **Run 1 cycle with the default FLUKA executable**.
  WARNING: Try with 2 primaries first (output file size)!

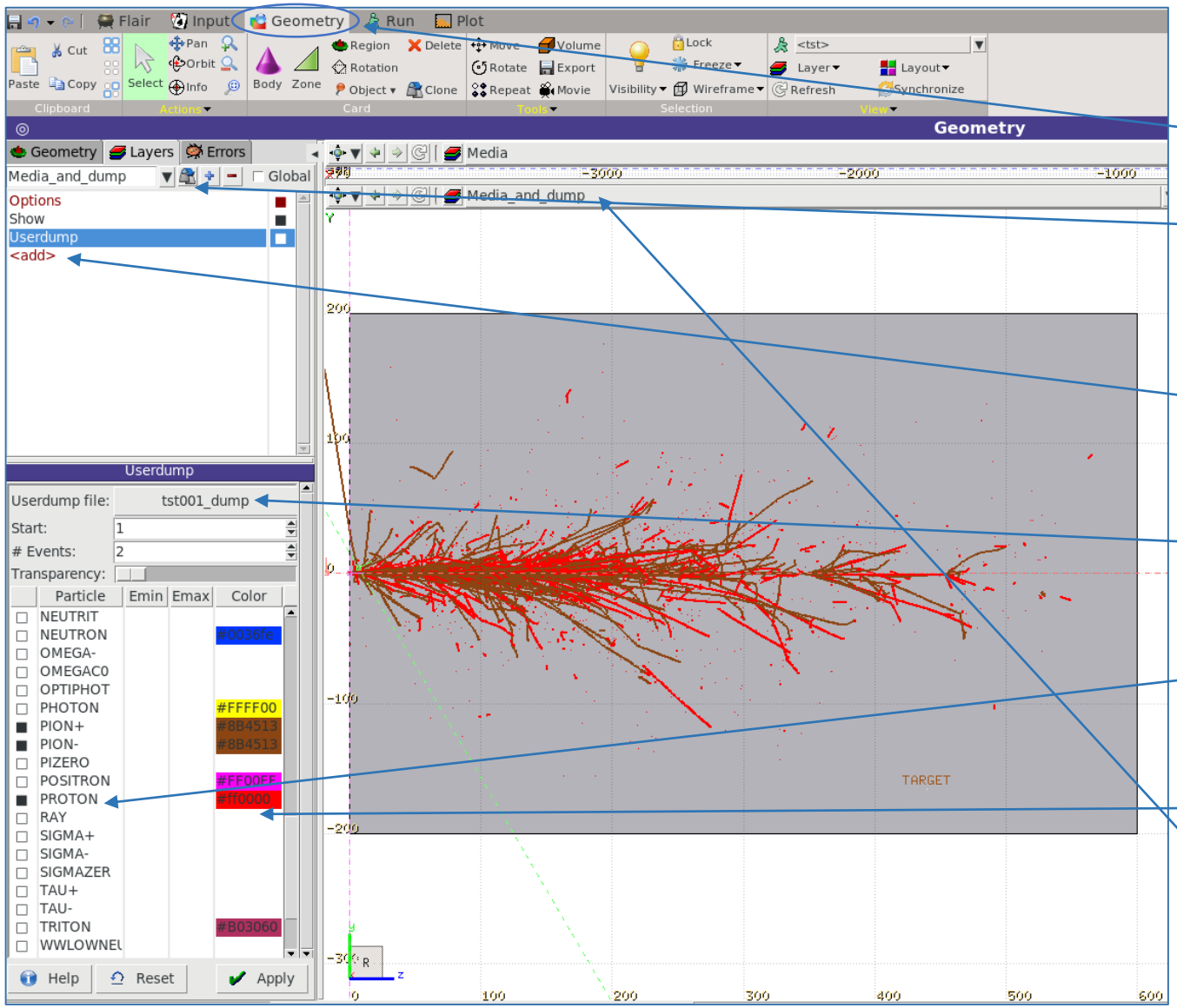
- Spot your dump file among the output files. What is its **size in MB**? What is its **format**?
  NB: You cannot "Process" dump files.

- Use the guide in the next slide to draw protons trajectories in red, pi+ and pi- trajectories in brown in the same plot.

- Observe neutrons, photons, e-/e+ trajectories.

- Until when does the hadronic shower continue?

Example: plotting trajectories with Flair



Follow the steps below in Flair to draw trajectories of interest in your geometry.

Step 1: Select the **Geometry** tab.

Step 2: (optional) **Clone an existing layer** and rename it. E.g.: Media_and_dump cloned from Media layer.

Step 3: **Add** a "Userdump" (active if filled black box).

Step 4: **Select the Userdump file** created by your FLUKA run (here, *_dump).

Step 5: **Select particles** of interest (active ←→ filled black box).

Step 6: Double-click on **colour box** to update colour.

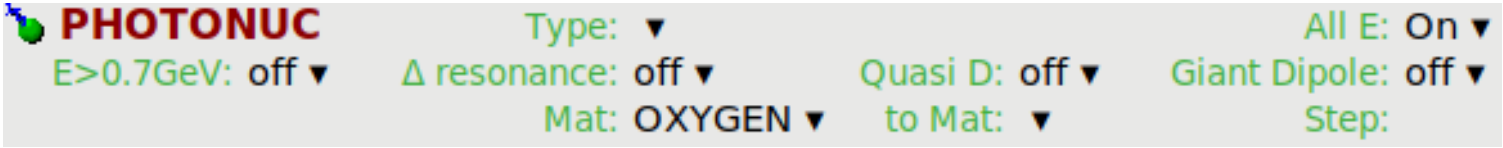
Step 7: **Select the layer** you created in Step 2 in any window of your choice.

Exercise 3: Study reaction final state with USDRAW

The **USDRAW** entry (MGDRAW routine) can provide user-defined dumps after each interaction: we need to **customise it to our needs**.

In this exercise, we are going to study the final state of photonuclear reactions on ^{16}O.

- Use the provided input file and **set up a beam** with the following parameters:
  - Particle: Photon (particle code: 7)
  - Momentum: 35 MeV/c
- Add a **new material card** to overwrite OXYGEN with ^{16}O.
- **Activate photonuclear reactions** with the `PHOTONUC` card:



- **Bias** the photonuclear reactions to increase their occurrence by a factor 100:

Exercise 3: Study reaction final state with USDRAW

- Activate the calls to USDRAW by adding a **USERDUMP** card.


```
* ..+....1....+....2....+....3....+....4....+....5....+....6....+....7..
USERDUMP              100.         0.           1.
```

- **Read the USDRAW entry** in the provided **mgdraw_ex3.f**
- For your convenience, the dump of the final state (stored in the FLUKA COMMONs) is already provided:
  - secondaries (variables from GENSTCK COMMON).
  - heavy fragments from deuterons onward (variables from FHEAVY COMMON).
  - one heavy residue, unless emitted particles and heavy fragments already exhaust the product list (variables from RESNUC COMMON).
- Note that in the future you will be able to reuse this code, which dumps any reaction final state, for any interaction you would like to study.

- Fill the **conditional statement to select photonuclear reactions** only.

Exercise 3: Study reaction final state with USDRAW

User implementation in mgdraw.f

```
  entry USDRAW(icode, mreg, xsco, ysco, zsco)

  if (icode .eq. 101 .and. JTRACK .eq. 7) then

     write (90, *)
     write (90, *) 'PROJECTILE: id = ', JTRACK,
&         ', kE[GeV] = ', ETRACK, ', dirZ = ', CXTRCK

        write (90, *) 'Interaction id = ', ICODE,
&         ', in region = ', MREG

     write(90,*) 'NUMBER OF GENSTCK SECONDARIES = ', NP-NP0
     do jp = NP0+1, NP
        write(90,*) 'Sec: id = ', KPART(jp), ', kE[GeV] = ', TKI(jp)
     end do

     write(90,*) 'NUMBER OF HEAVY FRAGMENTS = ', NPHEAV
     do jp = 1, NPHEAV
        write(90,*) 'A = ', IBHEAV(KHEAVY(jp)),
&            ', Z = ', ICHEAV(KHEAVY(jp)), ', kE[GeV] = ', TKHEAV(jp)
     end do

     if (IBRES .gt. 0) then
        write(90,*) 'RESIDUAL NUCLEUS A = ', IBRES,
&            ', Z = ', ICRES, ', kE[GeV] = ', EKRES
     end if

  end if
  return
```

Filtering: photonuclear reactions only

- Obviously can adapt to use case:
  - **reaction** code (ICODE)
  - **projectile** (ΞTRACK), **region** (MREG)
  - **primary history index** (NCASE)
  - **generation number** (LTRACK)...

Loop on GENSTCK secondaries

See `genstck.inc` for more properties

- **Start from NP0+1 to skip the primary when present**.
- NB: When the primary is not present, NP0=0.

Loop on heavy fragments `fheavy.inc`

Heavy residuals `resnuc.inc`

End filtering

Exercise 3: Study reaction final state with USDRAW

- **Build a custom executable** with your implementation of `mgdraw.f` (`mgdraw_ex3.f`).
  Run 5 cycles with 10 000 primaries each, using that executable.

- Have a look in the output. In **which files** are the final states dumped?
- The output of the user routine is customized by the user. Hence, **can you process** the output files?
- Open a file containing the final states dumped during one cycle.
  Check **the charge and baryon number conservation** for a few interactions.

- Modify the `mgdraw_ex3.f` routine to be able to evaluate the **number of photonuclear reactions N** occurring in 1 cycle.
  Photonuclear reactions are biased by a factor 100, and there are 10 000 primaries per cycle.
  Explain the relatively low value of N: what are the other reactions the primary photons are undergoing?

- Modify the `mgdraw_ex3.f` routine to be able to evaluate:
  - the $\gamma + {}^{16}O \rightarrow {}^{12}C + \alpha + \gamma$ channel relative probability.
  - the $\gamma + {}^{16}O \rightarrow \alpha + \alpha + \alpha + \alpha$ channel relative probability.
- Once you get first estimates, improve the statistics. Beware of the output files sizes.


FLUKA