A quick intro to the ALICE Software triggers

Maximiliano Puccio (CERN)

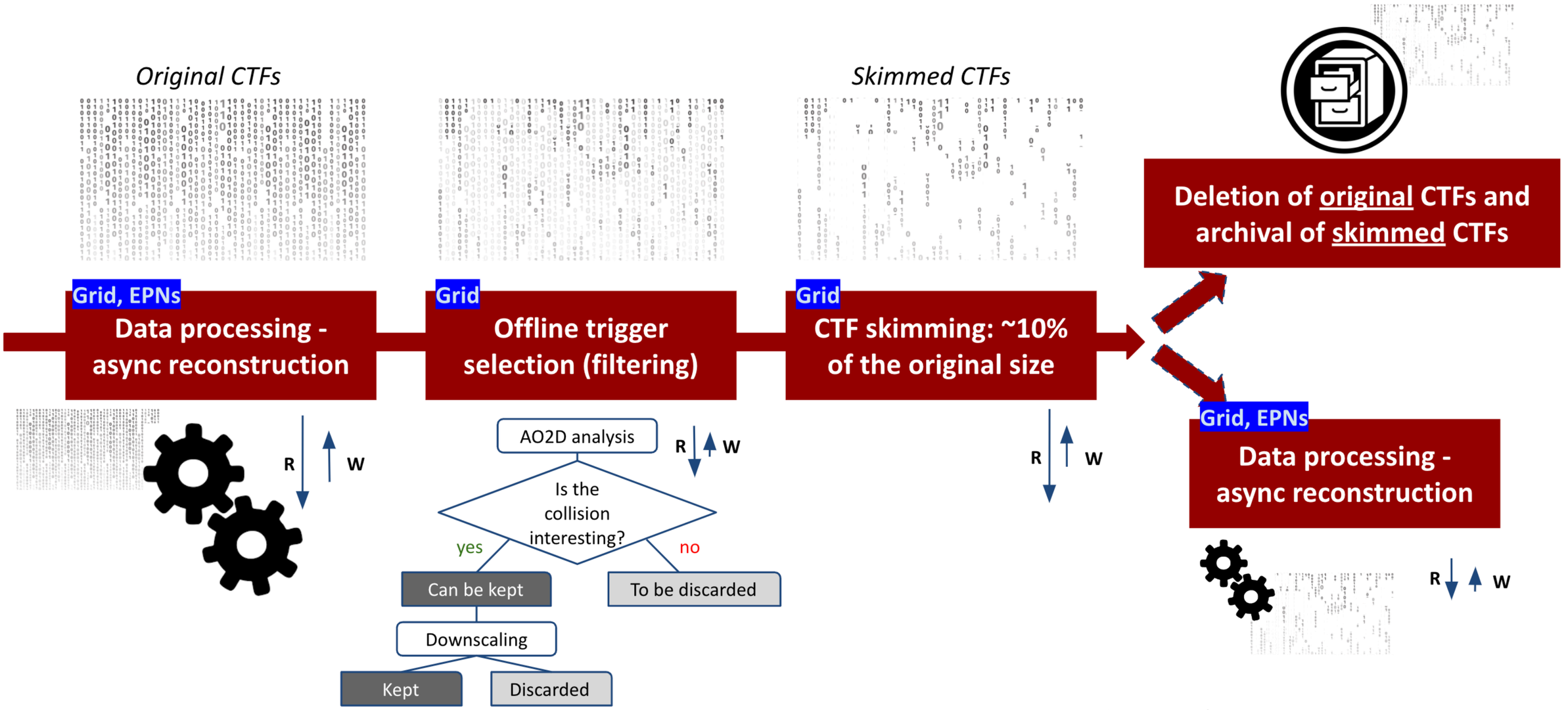
Analysis Tutorial 15th October 2024

Introduction

What should be the take away of this talk:

- Know what a software trigger selection is in ALICE
- Know which dataset you should use for your pp analysis
- Know how to analyse and normalise your analysis on triggered data

The data flow



NB: sizes of symbols/images only for illustration purposes

For more details on the data flow, see the DPG presentation.

The software trigger selection tasks

The software trigger selection tasks

The offline trigger selection triggers are simple analysis tasks producing a table containing one column for each interesting signal being looked for

- Example: the nuclei trigger task produces a table “NucleiFilter” with a boolean column for events containing deuterons, one for events containing helions, and so on…

The software trigger selection tasks

The offline trigger selection triggers are simple analysis tasks producing a table containing one column for each interesting signal being looked for

- Example: the nuclei trigger task produces a table “NucleiFilter” with a boolean column for events containing deuterons, one for events containing helions, and so on…

The selection table produced by the trigger task must be joinable with the collision table (i.e. it must contain one row for each collision. Practical consequence:

- One practical consequence: you can’t use `Filter` directives acting on collisions as the process function must process all of them

The software trigger selection tasks

The offline trigger selection triggers are simple analysis tasks producing a table containing one column for each interesting signal being looked for

- Example: the nuclei trigger task produces a table “NucleiFilter” with a boolean column for events containing deuterons, one for events containing helions, and so on…

The selection table produced by the trigger task must be joinable with the collision table (i.e. it must contain one row for each collision. Practical consequence:

- One practical consequence: you can’t use `Filter` directives acting on collisions as the process function must process all of them

Example task: EventFiltering/PWGLF/nucleiFilter.cxx

The software trigger selection tasks

The offline trigger selection triggers are simple analysis tasks producing a table containing one column for each interesting signal being looked for

- Example: the nuclei trigger task produces a table “NucleiFilter” with a boolean column for events containing deuterons, one for events containing helions, and so on…

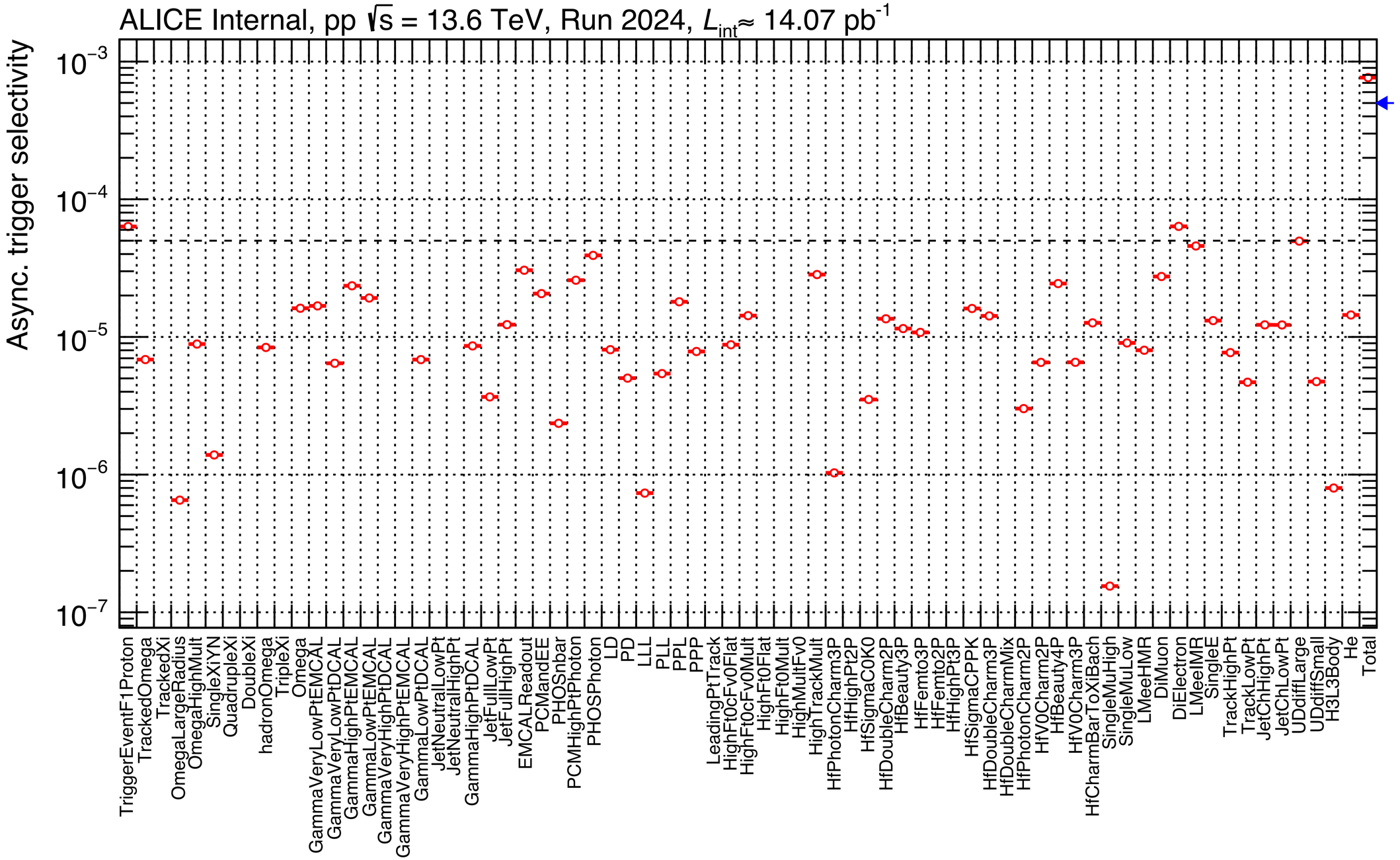
The selection table produced by the trigger task must be joinable with the collision table (i.e. it must contain one row for each collision. Practical consequence:

- One practical consequence: you can’t use `Filter` directives acting on collisions as the process function must process all of them

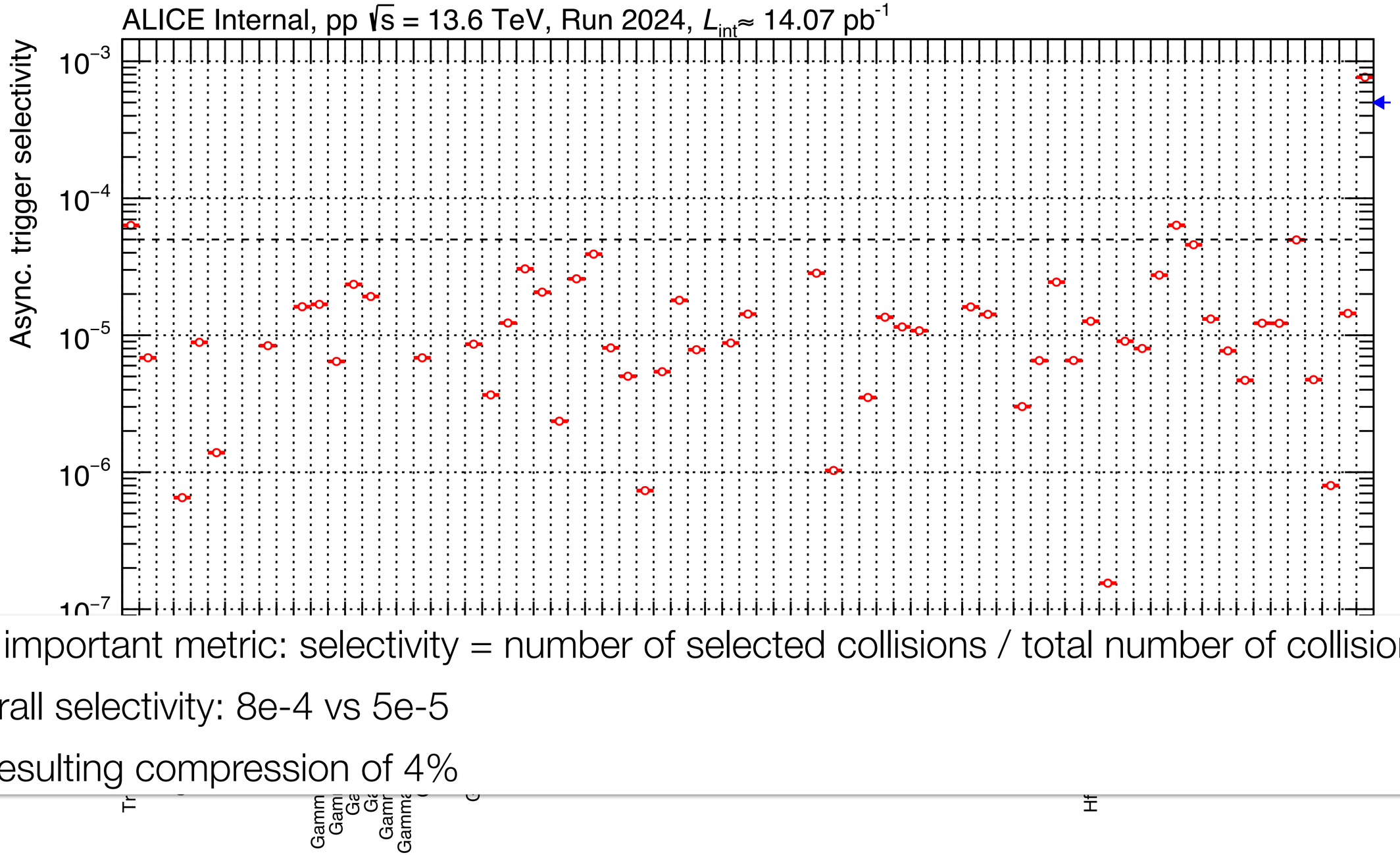
Example task: EventFiltering/PWGLF/nucleiFilter.cxx

If you want to add another trigger you can present your physics case to the trigger meeting where we can see how this will fit with the current trigger menu

The result of the filtering

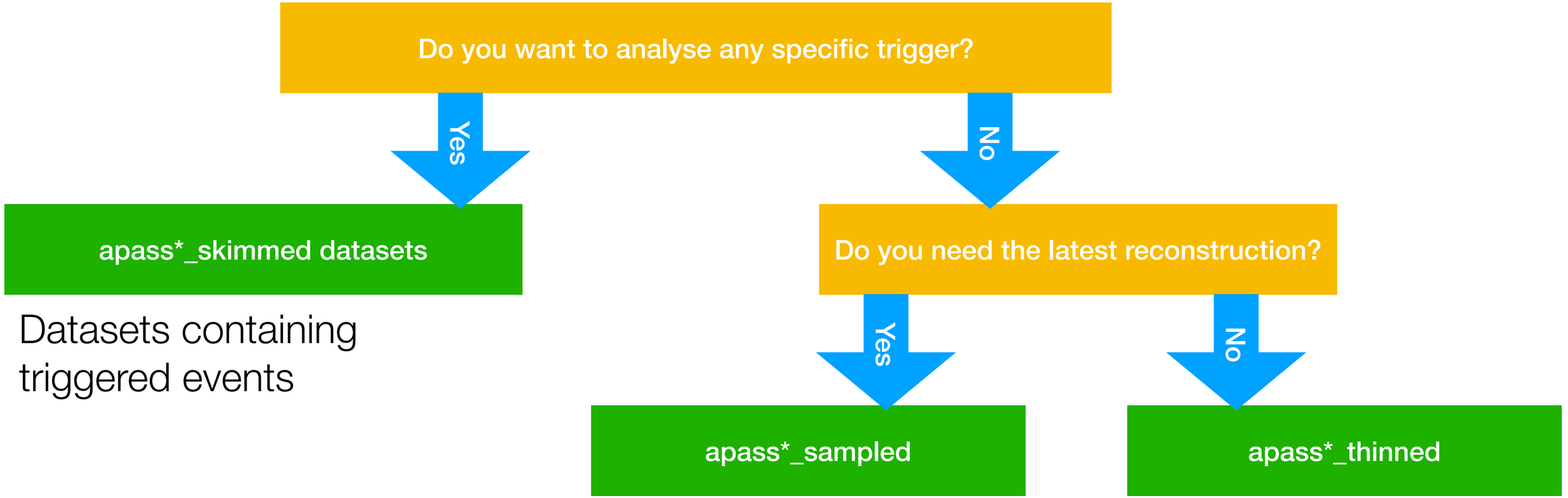


The result of the filtering



- The important metric: selectivity = number of selected collisions / total number of collisions
- Overall selectivity: 8e-4 vs 5e-5
  - Resulting compression of 4%

What datasets should you use for your analysis?



Datasets containing triggered events

Datasets containing a sampled MB events (0.5% in 2024, 0.5/pb in 2023, and 1/pb in 2022).

AO2Ds without TPC standalone tracks, only the reco available at the time of triggering is/will be available but full MB statistics. Datasets for 2022 and 2023 available

Analysing the skimmed dataset

All the skimmed datasets are already available on hyperloop to be analysed

- From P2 to your laptop in less than 2 months
- Currently, from 2024, we have an equivalent statistics that is larger than the 2022 dataset

Two different use cases:

- Analyses with no event normalisation (e.g. multi-particle correlations, jet-substructure …): you can just run on these datasets with no modifications to your code
- Analyses with event normalisation: you will need to account for the fact that you analyse only a subset of the integrated T0VTX triggers corresponding to interesting events

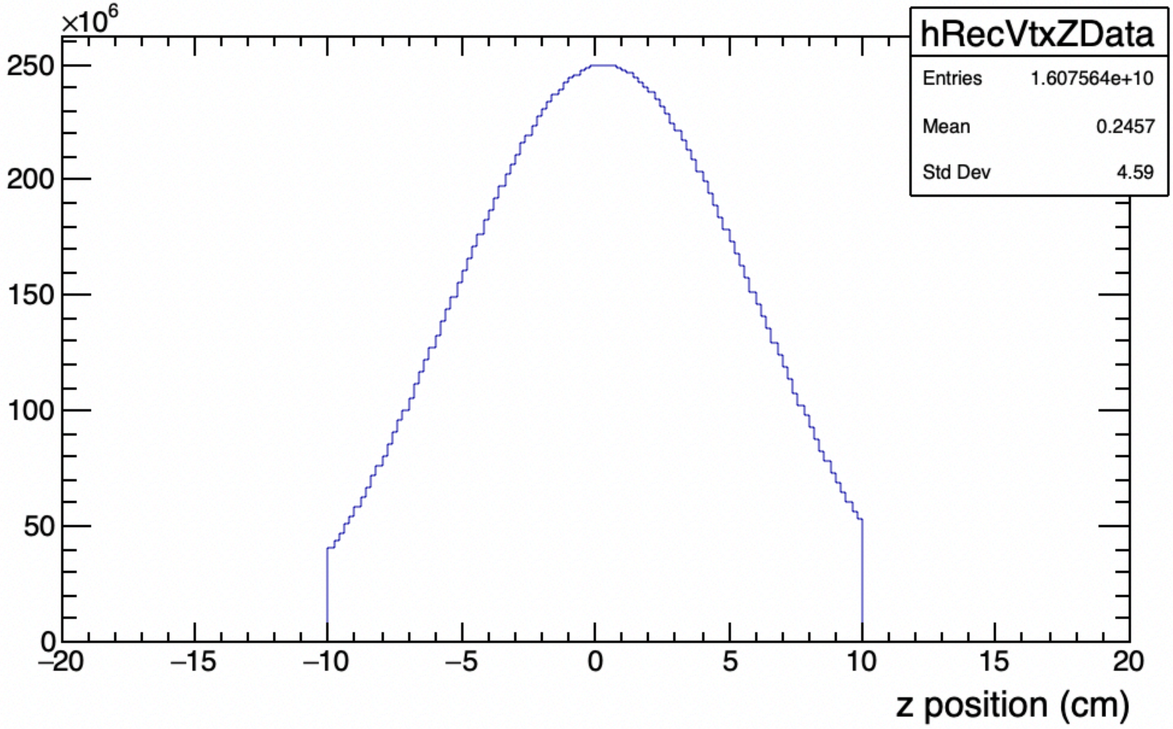
Analysing the skimmed dataset

All the skimmed datasets are already available on hyperloop to be analysed

- From P2 to your laptop in less than 2 months
- Currently, from 2024, we have an equivalent statistics that is larger than the 2022 dataset

Two different use cases:

- Analyses with no event normalisation (e.g. multi-particle correlations, jet-substructure …): you can just run on these datasets with no modifications to your code
- Analyses with event normalisation: you will need to account for the fact that you analyse only a subset of the integrated T0VTX triggers corresponding to interesting events



Example: analysing the 2024 skimmed dataset, we see ~16 billion events, but the equivalent number of T0TVX is ~1000 billion events.

Introducing Zorro

The information stored in the CCDB allows us to analyse the skimmed data by selecting BC that were triggered for specific channels

- New tool (*zorro*) for analysis tasks to select interesting events

Easy setup, in your task add the object *zorro* and initialise it in your `initCCDB` function

```
if (cfgSkimmedProcessing) {
  zorro.initCCDB(ccdb.service, bc.runNumber(), bc.timestamp(), "fHe3");
}
```

Trigger of interest (comma separated list if you are interested in more than one)

Then in your process function you can check if the collision/BC (example, one could use the `foundBC` as well) you are analysing was selected with:

```
zorro.isSelected(collision.bc_as<aod::BCsWithTimestamps>().globalBC());
```

You can also add a control plots to your histogram registry that can be used for the normalisation with the function

```
void populateHistRegistry(o2::framework::HistogramRegistry& histRegistry, int runNumber, std::string folderName = "Zorro");
```

Under the hood

Subfolder
Users/m/mpuccio/EventFiltering/OTS/FilterBitMasks
Users/m/mpuccio/EventFiltering/OTS/FilterCounters
Users/m/mpuccio/EventFiltering/OTS/InspectedTVX
Users/m/mpuccio/EventFiltering/OTS/SelectedBCs
Users/m/mpuccio/EventFiltering/OTS/SelectionBitMasks
Users/m/mpuccio/EventFiltering/OTS/SelectionCounters
Users/m/mpuccio/EventFiltering/OTS/ZorroHelpers

Objects available for 2022, 2023 and 2024 data

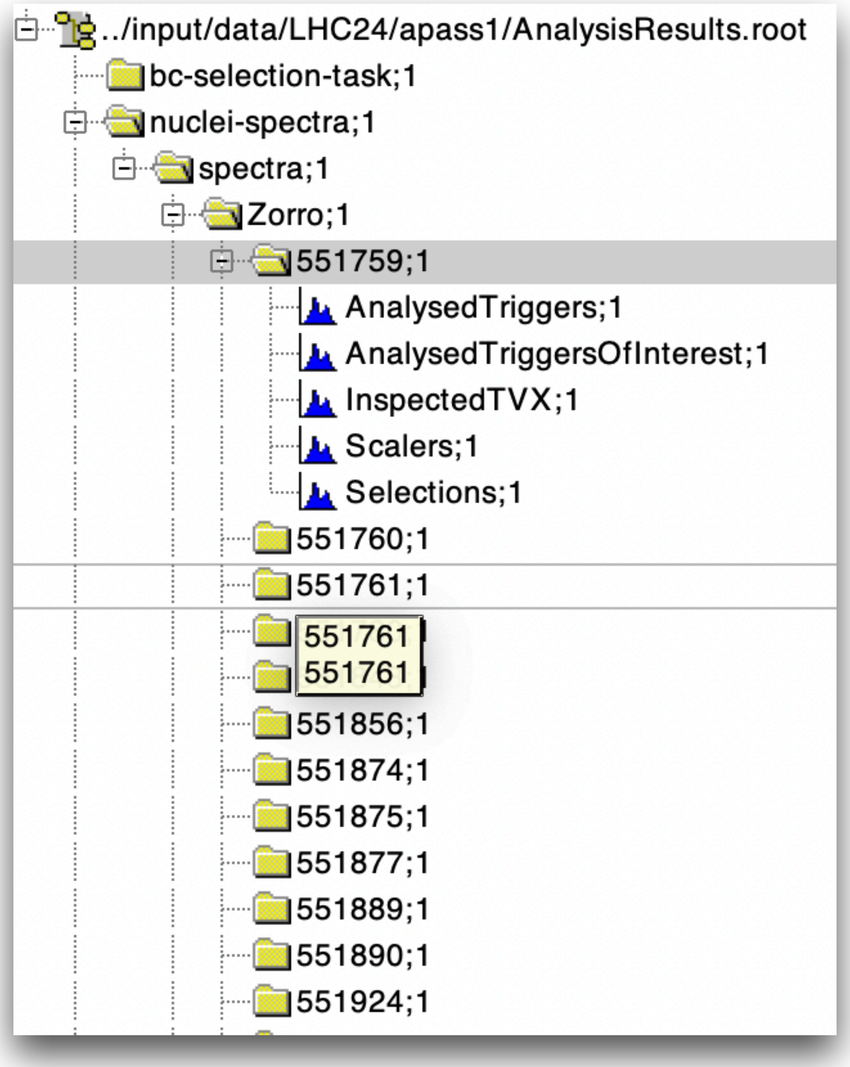
For each analysed run, Zorro reads from the CCDB the following information:

- Total number of T0VTX triggers inspected by the trigger/skimming procedure
- The total number of selected triggers by the trigger/skimming procedure for each of the analysed runs
- The list of all the BCs selected by the trigger/skimming procedure with the corresponding selection mask

When analysing the data, it is important to avoid double counting in the normalisation as more than one collision can be compatible with the triggered BC

- Zorro does this accounting for you in the normalisation histograms

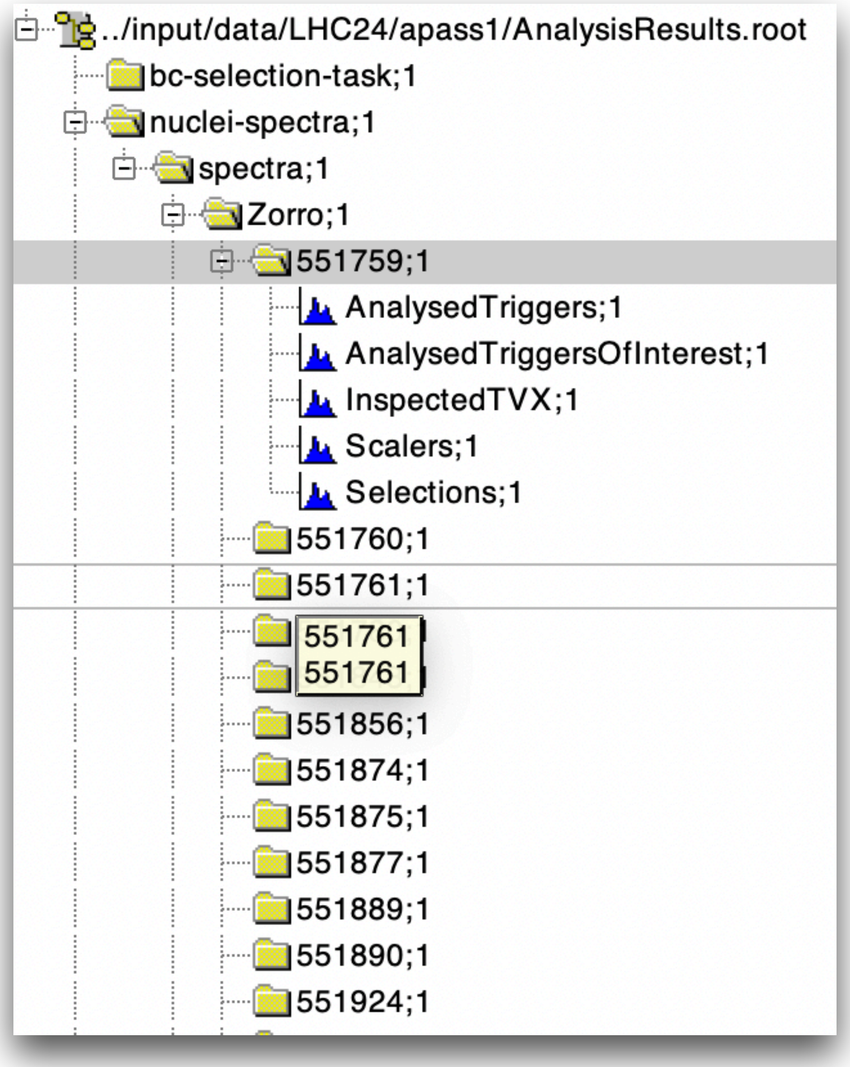
Normalisation histograms of Zorro



For each analysed run you get:

- The number of analysed triggers in your hyperloop train (both trigger of interest and other triggers)
- The T0TVX triggers inspected originally by the skimming
- The originally selected number of events per triggers (the Selections histogram)

Normalisation histograms of Zorro



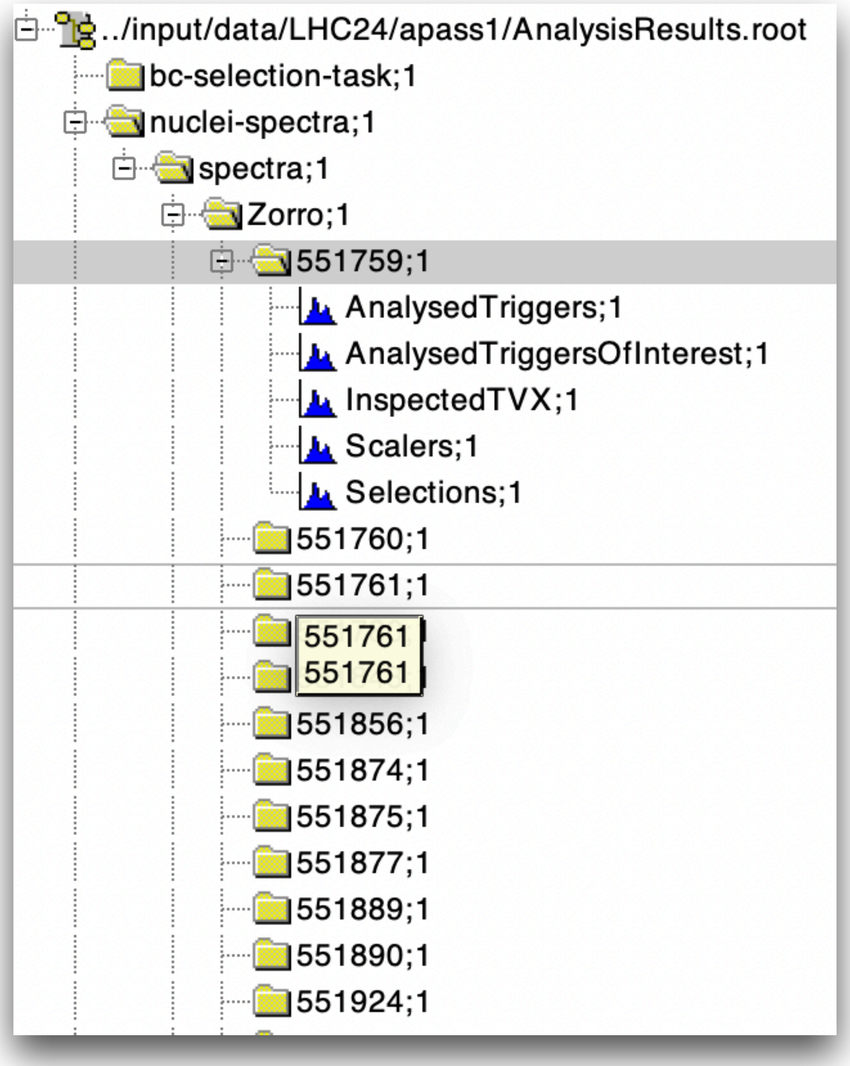
For each analysed run you get:

- The number of analysed triggers in your hyperloop train (both trigger of interest and other triggers)
- The T0TVX triggers inspected originally by the skimming
- The originally selected number of events per triggers (the Selections histogram)

To know the equivalent T0TVX triggers sampled by your hyperloop train run, you will need to do:

- Sum all the originally inspected T0TVX triggers
- Sum all the originally selected events for your trigger of interest
- Sum the number of analysed triggers of interest in the current hyperloop run

Normalisation histograms of Zorro



Example with the nuclei spectra:

For each analysed run you get:

- The number of analysed triggers in your hyperloop train (both trigger of interest and other triggers)
- The T0TVX triggers inspected originally by the skimming
- The originally selected number of events per triggers (the Selections histogram)

To know the equivalent T0TVX triggers sampled by your hyperloop train run, you will need to do:

- Sum all the originally inspected T0TVX triggers
- Sum all the originally selected events for your trigger of interest
- Sum the number of analysed triggers of interest in the current hyperloop run

Analysed triggers: 1.44265e+07

Total TVX: 1.04569e+12

Original triggers: 1.5017e+07

$$N_{T0VTX} = \frac{\textbf{Original T0VTX} \times \textbf{Analysed TOI}}{\textbf{Number of TOI in the skimming}} = 10^{12}$$

Normalisation shortcut using Zorro

Zorro can fill automatically an output object that makes the normalisation easier. 4-steps:

```
  Zorro zorro;
  OutputObj<ZorroSummary> zorroSummary{"zorroSummary"};
```

1) Add the output object ZorroSummary to the data member of you analysis task

2) Init your object in the init function of your task

```
    zorroSummary.setObject(zorro.getZorroSummary());
```

AliHyperloop 

3) Run your hyperloop train

4) Get the normalisation from your AnalysisResults.root

```
[O2Physics/latest] /tmp %> root -l AnalysisResults.root
Starting ROOT version 6.32.02.
root [0]
Attaching file AnalysisResults.root as _file0...
(TFile *) 0x12c776060
root [1] ZorroSummary* zorroSummary = (ZorroSummary*)_file0->Get("nuclei-spectra/zorroSummary");
root [2] std::cout << zorroSummary->getNormalisationFactor(0) << std::endl;
6.80235e+11
```

Conclusions

Most of the data collected by ALICE will be subjected to the Offline Trigger Selection treatment

- Don’t be a stranger to that, you can do a lot of physics with those data
- If you need help / want to have a new trigger reach out to the trigger MM channel