

HF O² Hands-on session

V. Kučera (Inha University),
L. Dello Stritto (CERN)



Introduction to the mini-task

- Two simplified tasks containing a basic structure of the full O2 analysis chain for D^0 mesons:

<https://github.com/AliceO2Group/O2Physics/blob/master/Tutorials/PWGHF/skimCreatorMini.cxx>

{ Skimming

<https://github.com/AliceO2Group/O2Physics/blob/master/Tutorials/PWGHF/taskMini.cxx>

{ Candidate creator
Candidate selector
Analysis task

- In this tutorial:
Modify the taskMini and run it on a provided skimmed derived data file.
 Standard way to perform an HF analysis in O2.

Caveat. Advanced features non included in the mini tasks: MC, CCDB, process function switches, ambiguous tracks, primary-vertex refit, selection cut arrays.

Introduction to the mini-task

- The tutorial code is in the [O2Physics/Tutorial/PWGHF](#) directory:

[skimCreatorMini.cxx](#) - O2 simplified skimming task code

[dpl-config_skim.json](#) - json configuration file of O2 skimming mini task

[run_skim.sh](#) - bash script to execute the skimming mini task workflow

Introduction to the mini-task

- The tutorial code is in the [O2Physics/Tutorial/PWGHF](#) directory:

[skimCreatorMini](#) simplified skim task code

[dpl-config_skim.json](#) file of O2 skimming mini task

[run_skim.sh](#) bash script to execute skimming mini task workflow

Introduction to the mini-task

- The tutorial code is in the [O2Physics/Tutorial/PWGHF](#) directory:

[skimCreatorMini.cxx](#) - O2 simplified task code

[dpl-config_skim.json](#) - json configuration file of O2 skimming mini task

[run_skim.sh](#) - bash script to execute skimming mini task workflow

[taskMini.cxx](#) - O2 simplified task code

[dpl-config_task.json](#) - json configuration file of O2 mini task

[run_task.sh](#) - bash script to execute the mini task workflow

[DataModelMini.h](#) - Mini task data model header.
(it contains the definition of all the HF columns and tables).

Running the mini task

1. Create a working directory.
2. Get the input files (Run 3 MC pp LHC24h1b) and put them in your working directory.

<https://cernbox.cern.ch/s/DXok0HbrEAxVsBp>

[AO2D.root (parent AO2D file)
AnalysisResults_trees.root (skimming derived data file)

3. Load the O2Physics environment

4. Execute the bash script in the working directory:

bash <path>/O2Physics/Tutorials/PWGHF/run_task.sh

(The terminal output will be redirected into **stdout.log**)

Access to the parent AOD

- Processing the derived `AnalysisResults_trees.root` requires the access to the parent `AO2D.root` file.
- The path to the parent `AO2D.root` file is stored in the derived `AnalysisResults_trees.root` in the Tmap `parentFiles`. For the file provided in this tutorial:

```
root [0]
Attaching file AnalysisResults_trees.root as _file0...
(TFile *) 0x27e54b90
root [1] parentFiles->Print()
Collection name='TMap', class='TMap', size=15
Key:   TObjString = DF_2308102743120074
Value: TObjString = /home/vkucera/hfo2-tutorial_5/A02D.root
Key:   TObjString = DF_2308102745602922
Value: TObjString = /home/vkucera/hfo2-tutorial_5/A02D.root
```

Path stored in the derived data file:

`/home/vkucera/hfo2-tutorial_5`

YOU NEED TO OVERWRITE IT!

- The new derived file path can be set with the command line argument “`--aod-parent-base-path-replacement`”. Provide a replacement mask in the format “`old-path-to-parent;new-path-to-parent`”.
- If the parent and the derived files are both in the same directory, “**`new-path-to-parent`**” can be empty.

Example: `run_task.sh --aod-parent-base-path-replacement "/home/vkucera/hfo2-tutorial_5/;"`

Compiling with ninja

- Every time you modify something in the task you need to recompile the O² code following the [rebuilding instructions](#) in the official O² analysis framework documentation.

- To rebuild the full O² analysis framework and load it:

```
aliBuild build O2Physics
alienv enter O2Physics/latest
```

- You can also rebuild only a specific directory (much faster):

```
cd ~/alice/sw/BUILD/O2Physics-latest/O2Physics
direnv allow
ninja <directory>/install
```

 In our specific case: `ninja Tutorials/PWGHF/install`

Exercise 1

- Add a configurable cut on the decay length in the task.
- Define and fill a 2D histogram (vs p_T), using the AxisSpec class for the p_T .

Hints

- Define the configurable cut in the selector struct and add it in the json.
- Apply the cut in the selector struct.
- Define the histogram and fill it in the HfTaskD0 struct.

```
init() {  
    AxisSpec phiAxis = {100, 0., 2. * M_PI};  
    histos.add("phi", " phi", {HistType::kTH1F, {phiAxis}});  
}
```



- Add a histogram with the decayLengthXY.

Hints

- The decay length XY [column](#) is missing in the data model. You need to define it there!
- Add the decay length XY column to the 2-prong candidate [table](#).
- Define the histogram and use the [getter](#) to fill it.



- Change the D^0 selection partition in the HfTaskD0 struct into a filter.

Hints

- Replace the partition declaration with a filter declaration.
- Modify the process function accordingly.



- Add histograms for number of tracks, number of selected D^0 candidates per collision and a collision counter.

Hints

- Group tracks and D^0 candidates by collision modifying the process function.
- Get the number of tracks or D^0 candidates per collision by checking the table size (`.size()`)
- Add and fill new histograms.

N.B.: partitions **do not interact** with grouping (unless you use `sliceBy()`).

Backup
Slides