A survival guide to UPC o2 simulations

Simone Ragoni

Creighton University, USA

Requirements

- Working O2 installation
- Have the DPG tools ready (e.g. git clone git@github.com:AliceO2Group/O2DPG.git or aliBuild init O2DPG@master followed by aliBuild build O2DPG)
- If you plan on doing stuff on local, try to also have QC (brew install glfw on macOS followed by aliBuild init QualityControl@master --defaults o2 and aliBuild build QualityControl --defaults o2 works for me, the first is needed due to missing libraries on macOS…)
- Certificates loaded on the browser (if you have issues loading your certificates on your own browsers you can have them loaded on google chrome on the cernts remote windows service)

Requirements 2

- If you plan on doing stuff locally, e.g. unanchored MC productions, or trying to submit jobs on local, your laptop needs to have sufficient resources (they recommend 16GB of memory but it didn't work for me despite using the tricks to force the commands)

- If you don't have them then the only solution might really be to submit jobs for anchored productions on GRID, commands later

Common to all procedures

- For us of UPCs we need to use STARlight
- Now Michal Bros is developing the interface for direct STARlight input

STARlight

- Now part of ALICE O2
- An example can be found on https://github.com/AliceO2Group/AliceO2/tree/dev/run/SimExamples/HepMC_STARlight
- To start with, build STARlight on your laptop (cannot be found through cvmfs, alienv q | grep -i starlight returns nothing)
- This can be done with aliBuild build STARlight --defaults o2
  - Or through O2sim
- This builds r313 of STARlight on your laptop

STARlight: compilation issues

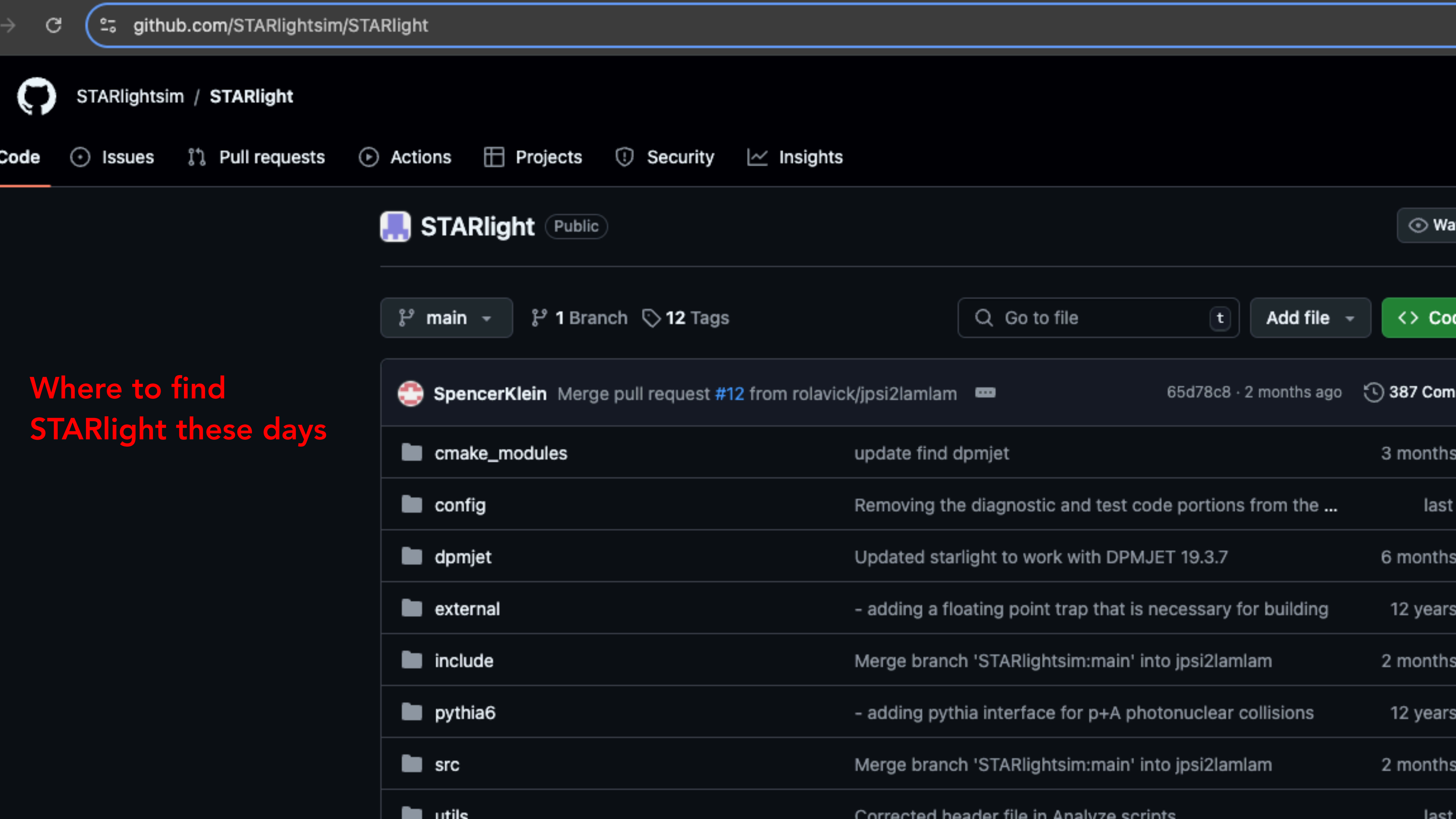
- If you can't build STARlight out of the box you will likely need to manually edit the source files
- They can usually be found in: /PATH/TO/O2/sw/SOURCES/STARlight/r313-c/r313-c/src/
- And then you can try to build again

- The only big advantage of doing it is having some starlight2hepMC converter available for you, but it can be found in the sw/SOURCES/STARlight/r313-c/r313-c/HepMC/ directory

STARlight: compilation issues

- If you still can't install it for some reason (e.g. the build command reverts the changes that you have made), the only option is to download STARlight and build it externally
- STARlight is only needed to generate the events
- You will still need to do the procedure in the previous slide just to have access to the most recent STARlight to HepMC converter by ALICE

Where to find
STARlight these days



github.com/STARlightsim/STARlight

STARlightsim / STARlight

STARlight Public

main · 1 Branch · 12 Tags

SpencerKlein Merge pull request #12 from rolavick/jpsi2lamlam · 65d78c8 · 2 months ago · 387 Com

cmake_modules	update find dpmjet	3 months
config	Removing the diagnostic and test code portions from the ...	last
dpmjet	Updated starlight to work with DPMJET 19.3.7	6 months
external	- adding a floating point trap that is necessary for building	12 years
include	Merge branch 'STARlightsim:main' into jpsi2lamlam	2 months
pythia6	- adding pythia interface for p+A photonuclear collisions	12 years
src	Merge branch 'STARlightsim:main' into jpsi2lamlam	2 months
utils	Corrected header file in Analyze scripts	last

STARlight

- Feed in a reasonable slight.in as usual
- Prepare two files:
  - pdgMass.awk
  - starlight2hepmc.awk
  - Both can be found in the source directory of STARlight (sw/SOURCES/STARlight/r313-c/r313-c/HepMC/)
- Run STARlight with ./starlight slight.in | tee slight.log
- Convert to HepMc format with cat slight.log slight.out | \ awk -f pdgMass.awk -f starlight2hepmc.awk > starlight.hepmc

Unanchored productions

Strategy

- Simulate STARlight events
- Convert to HepMc format
- Pass it to o2-sim
- Digitisation
- Reconstruction
- AO2D production

In principle straightforward, but it never works (with my laptop) just from outside the box

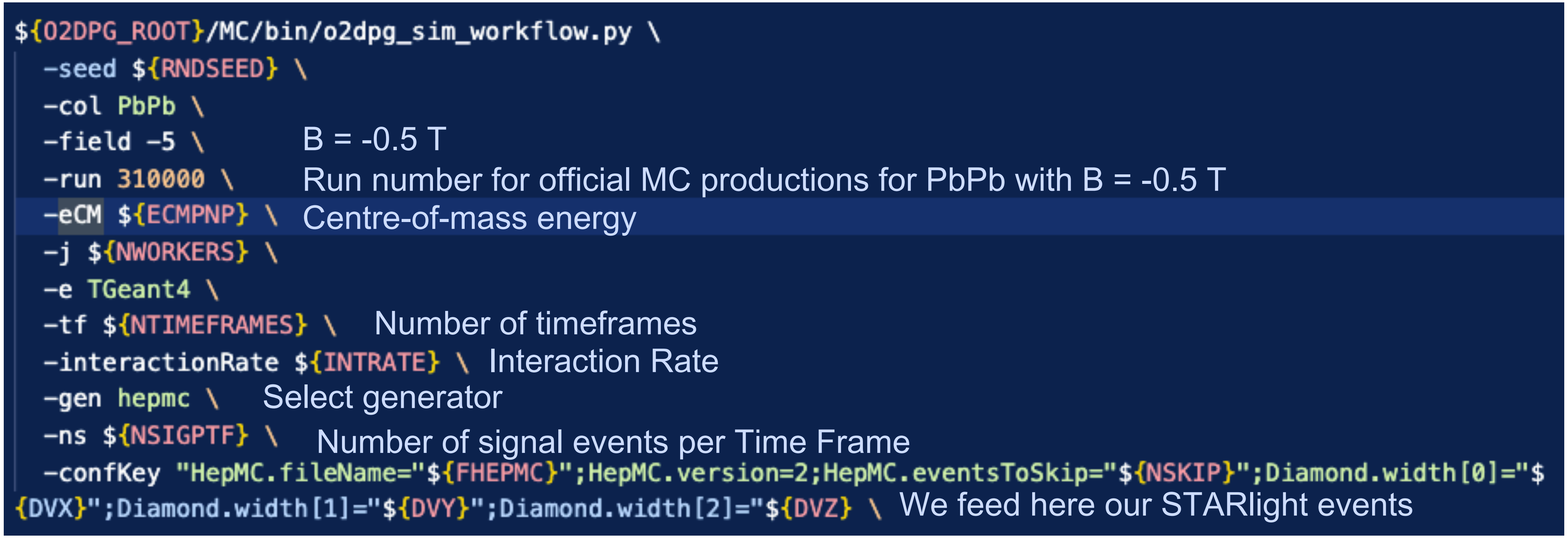
DPG

- Use DPG tools to obtain a workflow
- You first need to build them
- Do it with aliBuild build O2DPG
- These are also needed to run digitisation and reconstruction

- In a nutshell the two steps which are needed are:
  - Create a workflow which includes all steps from event generation to AO2D creation. The output of this process is workflow.json
  - Execute the workflow with the workflow.json as input (note that the workflow contains a “list” of commands to be run, useful for debugging in case of issues)

Workflow

- Use DPG tools to obtain a workflow
- Thanks Paul Buhler for helping me set up the workflow!
- Something like this can work:



```
${O2DPG_ROOT}/MC/bin/o2dpg_sim_workflow.py \
  -seed ${RNDSEED} \
  -col PbPb \
  -field -5 \
  -run 310000 \
  -eCM ${ECMPNP} \
  -j ${NWORKERS} \
  -e TGeant4 \
  -tf ${NTIMEFRAMES} \
  -interactionRate ${INTRATE} \
  -gen hepmc \
  -ns ${NSIGPTF} \
  -confKey "HepMC.fileName="${FHEPMC}";HepMC.version=2;HepMC.eventsToSkip="${NSKIP}";Diamond.width[0]="$
{DVX}";Diamond.width[1]="${DVY}";Diamond.width[2]="${DVZ} \
```

B = -0.5 T

Run number for official MC productions for PbPb with B = -0.5 T

Centre-of-mass energy

Number of timeframes

Interaction Rate

Select generator

Number of signal events per Time Frame

We feed here our STARlight events

Runner

- Now ready to run digitisation and AOD production
- Note: it might crush due to qedsim, solution: build AEGIS, since it might be that there is a missing library libTEPEMGEN with aliBuild build AEGIS —defaults O2
- There are two options here, running as part of a script, or in parallel mode (without script)
- It is mostly a choice based on resources of the machine, both options work on lxplus, I could only make the parallel mode half work on my laptop

Runner

- Parallel mode:
  - ${O2DPG_ROOT}/MC/bin/o2_dpg_workflow_runner.py -f workflow.json -tt dici -j 6 --stdout-on-failure --mem-limit 20000
  - ${O2DPG_ROOT}/MC/bin/o2_dpg_workflow_runner.py -f workflow.json -tt aod -j 6 --stdout-on-failure --mem-limit 20000
  - The —mem-limit option is to try and make it work on machines with few resources (such as my laptop)
- Serial/script mode:
  - ${O2DPG_ROOT}/MC/bin/o2_dpg_workflow_runner.py -f workflow.json -tt aod -j 6 --stdout-on-failure --mem-limit 20000 --produce-script myscript.sh
  - Source myscript.sh

Running on lxplus

- Has to be run on lxplus7 as far as I know
- Put your .globus folder in your home
- Load AEGIS, O2, and O2Physics in your environment, e.g.: /cvmfs/alice.cern.ch/bin/alienv enter VO_ALICE@O2DPG::daily-20231212-0100-1,VO_ALICE@AEGIS::v1.5-8,VO_ALICE@O2Physics::daily-20231212-0100-1
- Recreate the workflow (otherwise it will cause issues with paths)

Results of unanchored productions

- If everything goes fine you should have a AO2D.root after all of this with the correct output
    - There is a directory created per TF, each one with its own AODs
    - They are merged afterwards
    - If something went wrong consult the log files of this directory, which correspond to the jobs run by the workflow
- However, things might still go wrong even if the AO2D was produced, it is important to ascertain the status of all the digitisation and reconstruction on ALL the logs
- Example: all the logs were fine and just four ERROR messages on the TPC
- If the AO2D was produced, and only the BC-related branches are filled, it is very likely that something was broken at the level of the inputs - most likely input HepMC was somehow not fed

Anchored productions

Strategy

- Much easier here, let's fully use the DPG tools to get around!

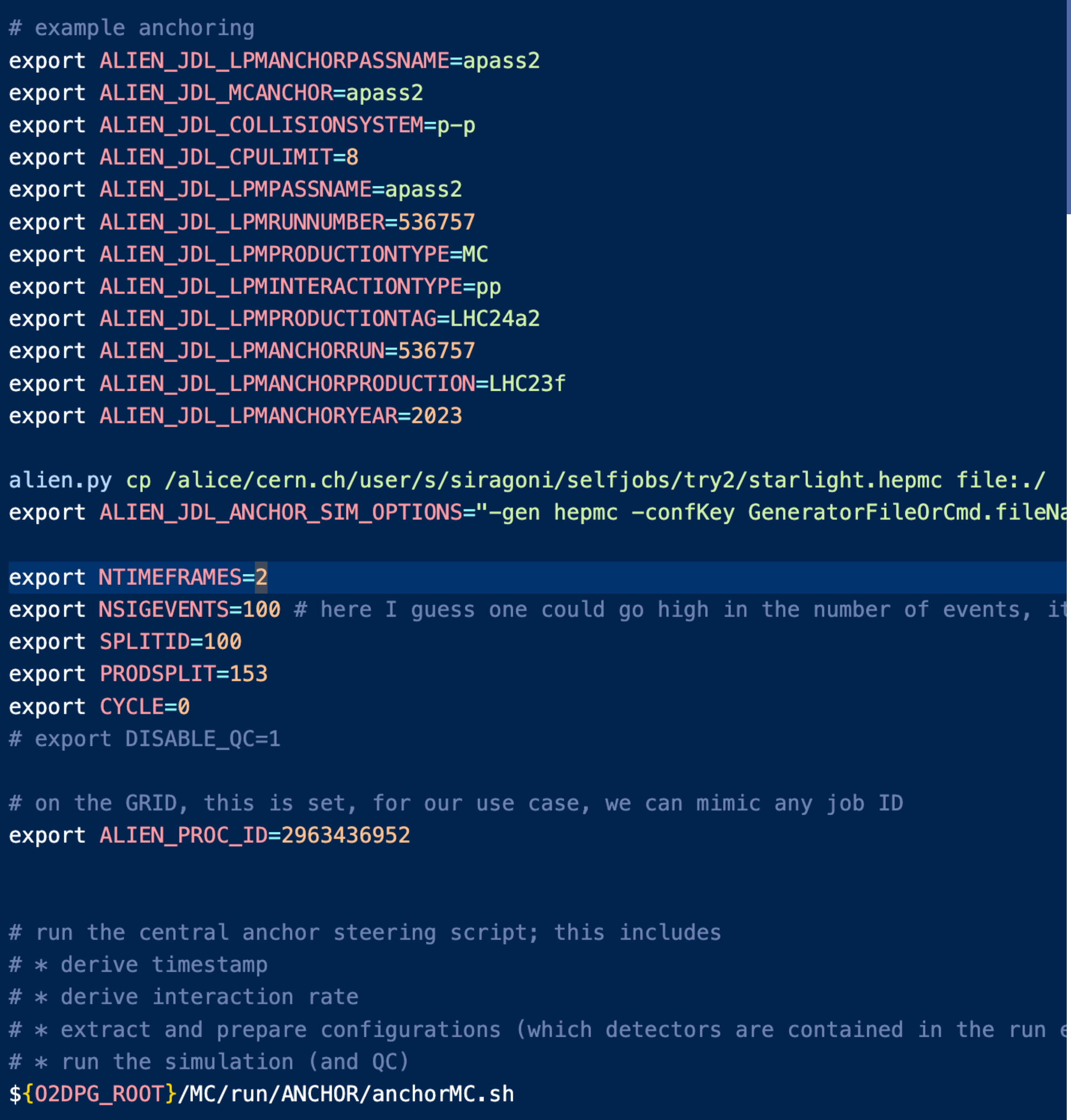
- We completely rely on ${O2DPG_ROOT}/GRID/utils/grid_submit.sh to generate all the submission logic
- This script can produce both local and GRID jobs
- My advice: try NOT to submit local jobs, it might have issues with QC stuff and settings, with only solution manually disabling features by editing the submission script

Common issues

- Most likely you will have CCDB connection issues
- Before anything happens, try doing alien.py (the successor of aliensh)
- If you can’t see anything, remember to: create a token e.g. alien-token-init siragoni, check that the tokens are in the tmp, e.g. ls /tmp, you should find files like tokenkey_$UID.pem and tokencert_$UID.pem, then you have to source new environment variables (there is an ongoing open issue with alien credentials) as export X509_USER_PROXY=/tmp/x509up_$UID, export JALIEN_TOKEN_CERT=/tmp/tokencert_$UID.pem, export JALIEN_TOKEN_KEY=/tmp/tokenkey_$UID.pem
- If you still have issues connecting with the grid_submit.sh file due to misspelt username, there are two solutions, one with editing the file itself (never touch it if you don’t fully know what you are doing), the other by adding to the .zshrc export ALIEN_USER="siragoni"

The command

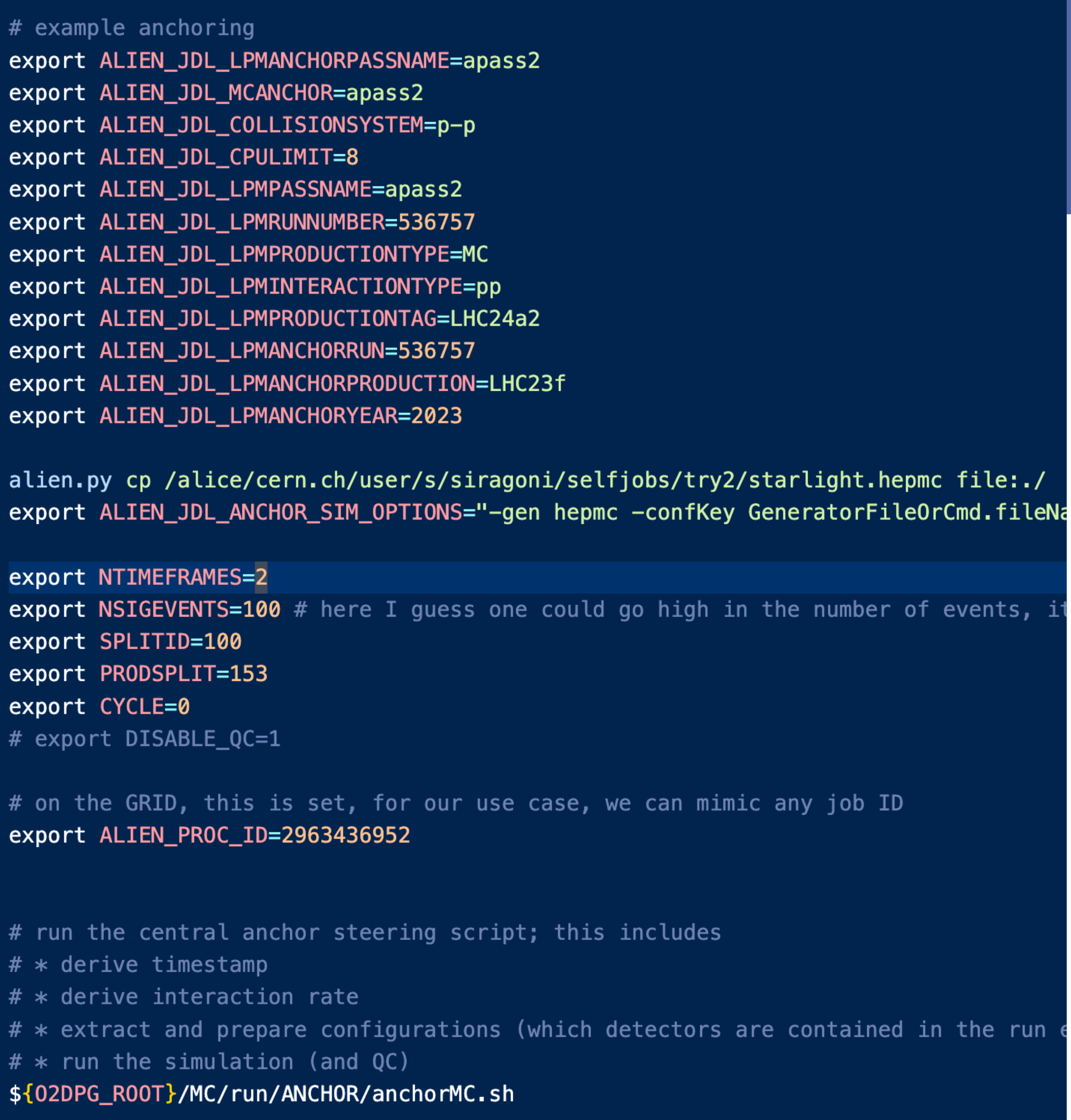
- The full command should look like: ${O2DPG_ROOT}/GRID/utils/grid_submit.sh --script run_anchored_prod_phi.sh --jobname testPhi --outputspec "*.log@disk=1","*.root@disk=2" --packagespec "VO_ALICE@O2sim::v20240303-1" --wait --fetch-output
- Try not to touch the output spec, and keep a recent version of the O2sim (e.g. use grep from lxplus for that, this version should be ok for now)
- In principle all of your anchoring logic and all the stuff that you CAN touch should go inside the script!



```
# example anchoring
export ALIEN_JDL_LPMANCHORPASSNAME=apass2
export ALIEN_JDL_MCANCHOR=apass2
export ALIEN_JDL_COLLISIONSYSTEM=p-p
export ALIEN_JDL_CPULIMIT=8
export ALIEN_JDL_LPMPASSNAME=apass2
export ALIEN_JDL_LPMRUNNUMBER=536757
export ALIEN_JDL_LPMPRODUCTIONTYPE=MC
export ALIEN_JDL_LPMINTERACTIONTYPE=pp
export ALIEN_JDL_LPMPRODUCTIONTAG=LHC24a2
export ALIEN_JDL_LPMANCHORRUN=536757
export ALIEN_JDL_LPMANCHORPRODUCTION=LHC23f
export ALIEN_JDL_LPMANCHORYEAR=2023

alien.py cp /alice/cern.ch/user/s/siragoni/selfjobs/try2/starlight.hepmc file:./
export ALIEN_JDL_ANCHOR_SIM_OPTIONS="-gen hepmc -confKey GeneratorFileOrCmd.fileNa

export NTIMEFRAMES=2
export NSIGEVENTS=100 # here I guess one could go high in the number of events, it
export SPLITID=100
export PRODSPLIT=153
export CYCLE=0
# export DISABLE_QC=1

# on the GRID, this is set, for our use case, we can mimic any job ID
export ALIEN_PROC_ID=2963436952


# run the central anchor steering script; this includes
# * derive timestamp
# * derive interaction rate
# * extract and prepare configurations (which detectors are contained in the run e
# * run the simulation (and QC)
${O2DPG_ROOT}/MC/run/ANCHOR/anchorMC.sh
```

The anchoring script (for pp?)

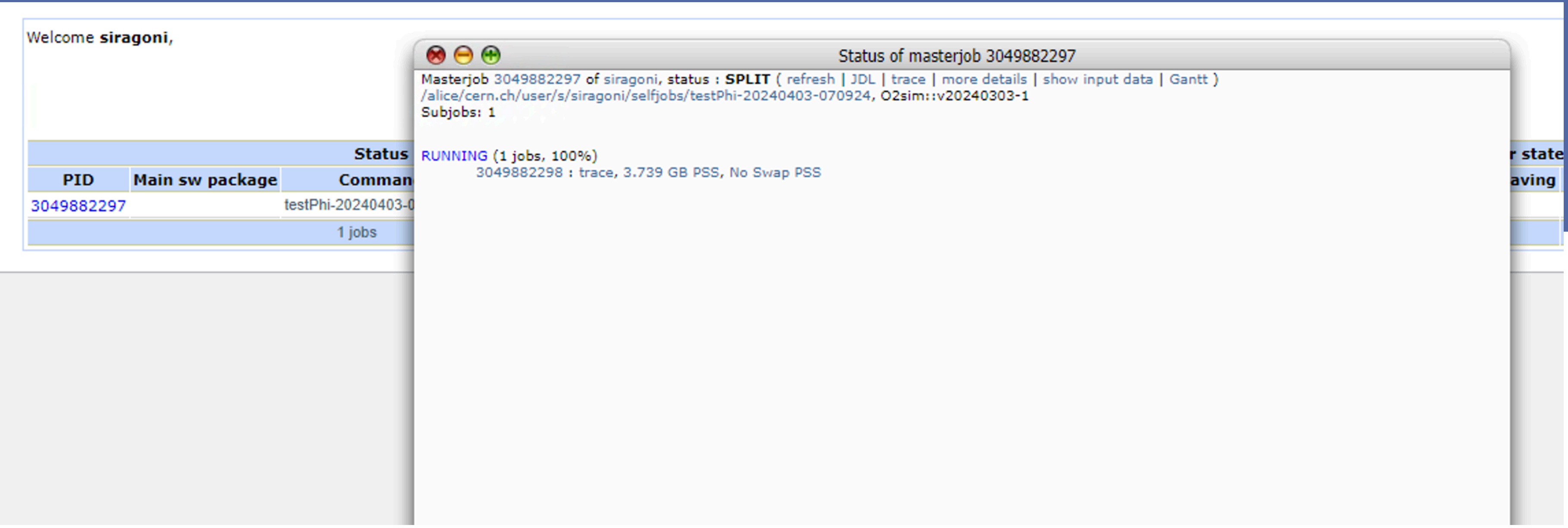
- You just need to change the settings to effect the anchoring you need
- The important stuff is that you have to upload your HepMC file to alien beforehand, e.g. alien.py cp -f file:starlight-phi.hepmc /alice/cern.ch/user/s/siragoni/selfjobs/try2/
- Then the file is copied at runtime from alien to the working dir with the command alien.py cp /alice/cern.ch/user/s/siragoni/selfjobs/try2/starlight.hepmc file:./
- You need to refer to the file with:



```
# example anchoring
export ALIEN_JDL_LPMANCHORPASSNAME=apass2
export ALIEN_JDL_MCANCHOR=apass2
export ALIEN_JDL_COLLISIONSYSTEM=p-p
export ALIEN_JDL_CPULIMIT=8
export ALIEN_JDL_LPMPASSNAME=apass2
export ALIEN_JDL_LPMRUNNUMBER=536757
export ALIEN_JDL_LPMPRODUCTIONTYPE=MC
export ALIEN_JDL_LPMINTERACTIONTYPE=pp
export ALIEN_JDL_LPMPRODUCTIONTAG=LHC24a2
export ALIEN_JDL_LPMANCHORRUN=536757
export ALIEN_JDL_LPMANCHORPRODUCTION=LHC23f
export ALIEN_JDL_LPMANCHORYEAR=2023

alien.py cp /alice/cern.ch/user/s/siragoni/selfjobs/try2/starlight.hepmc file:./
export ALIEN_JDL_ANCHOR_SIM_OPTIONS="-gen hepmc -confKey GeneratorFileOrCmd.fileNa

export NTIMEFRAMES=2
export NSIGEVENTS=100 # here I guess one could go high in the number of events, it
export SPLITID=100
export PRODSPLIT=153
export CYCLE=0
# export DISABLE_QC=1

# on the GRID, this is set, for our use case, we can mimic any job ID
export ALIEN_PROC_ID=2963436952


# run the central anchor steering script; this includes
# * derive timestamp
# * derive interaction rate
# * extract and prepare configurations (which detectors are contained in the run e
# * run the simulation (and QC)
${O2DPG_ROOT}/MC/run/ANCHOR/anchorMC.sh
```

The anchoring script (for pp?)

- You need to refer to the file with the confKey, e.g. export ALIEN_JDL_ANCHOR_SIM_OPTIONS="-gen hepmc -confKey GeneratorFileOrCmd.fileNames=${PWD}/starlight.hepmc"
- AnchorMC.sh will finish the job for you

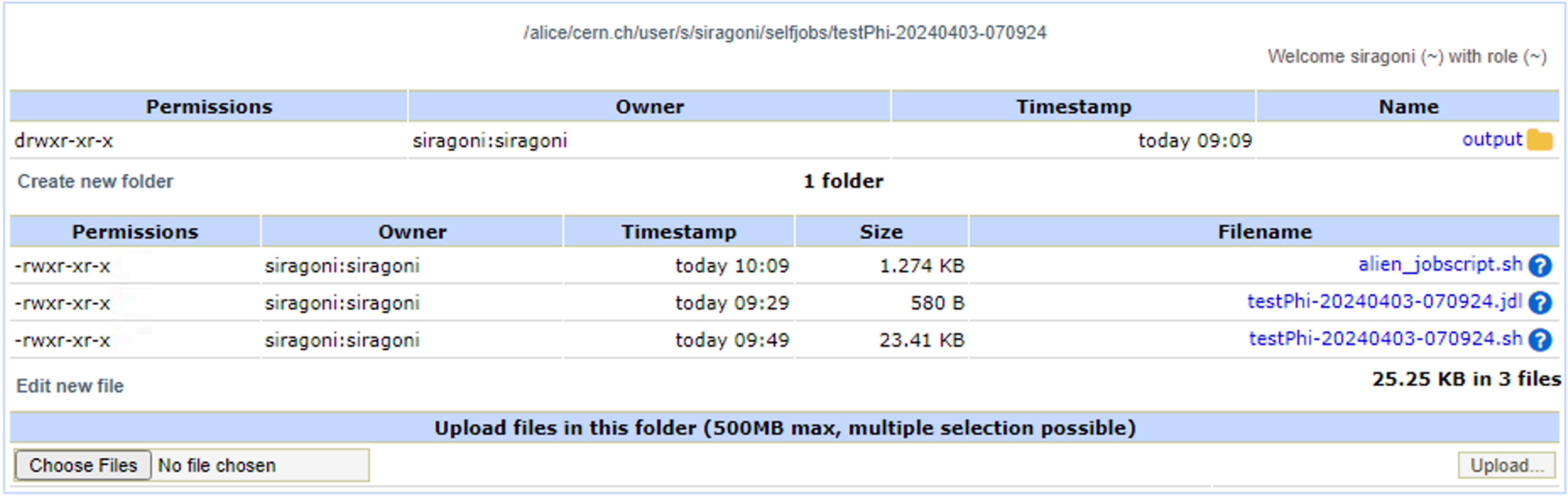


Welcome **siragoni**,

		Status	
PID	**Main sw package**	**Comman**	**aving**
3049882297		testPhi-20240403-0	
		1 jobs	

Status of masterjob 3049882297

```
Masterjob 3049882297 of siragoni, status : SPLIT ( refresh | JDL | trace | more details | show input data | Gantt )
/alice/cern.ch/user/s/siragoni/selfjobs/testPhi-20240403-070924, O2sim::v20240303-1
Subjobs: 1

RUNNING (1 jobs, 100%)
        3049882298 : trace, 3.739 GB PSS, No Swap PSS
```



/alice/cern.ch/user/s/siragoni/selfjobs/testPhi-20240403-070924

Welcome siragoni (~) with role (~)

Permissions	Owner	Timestamp	Name
drwxr-xr-x	siragoni:siragoni	today 09:09	output

Create new folder — **1 folder**

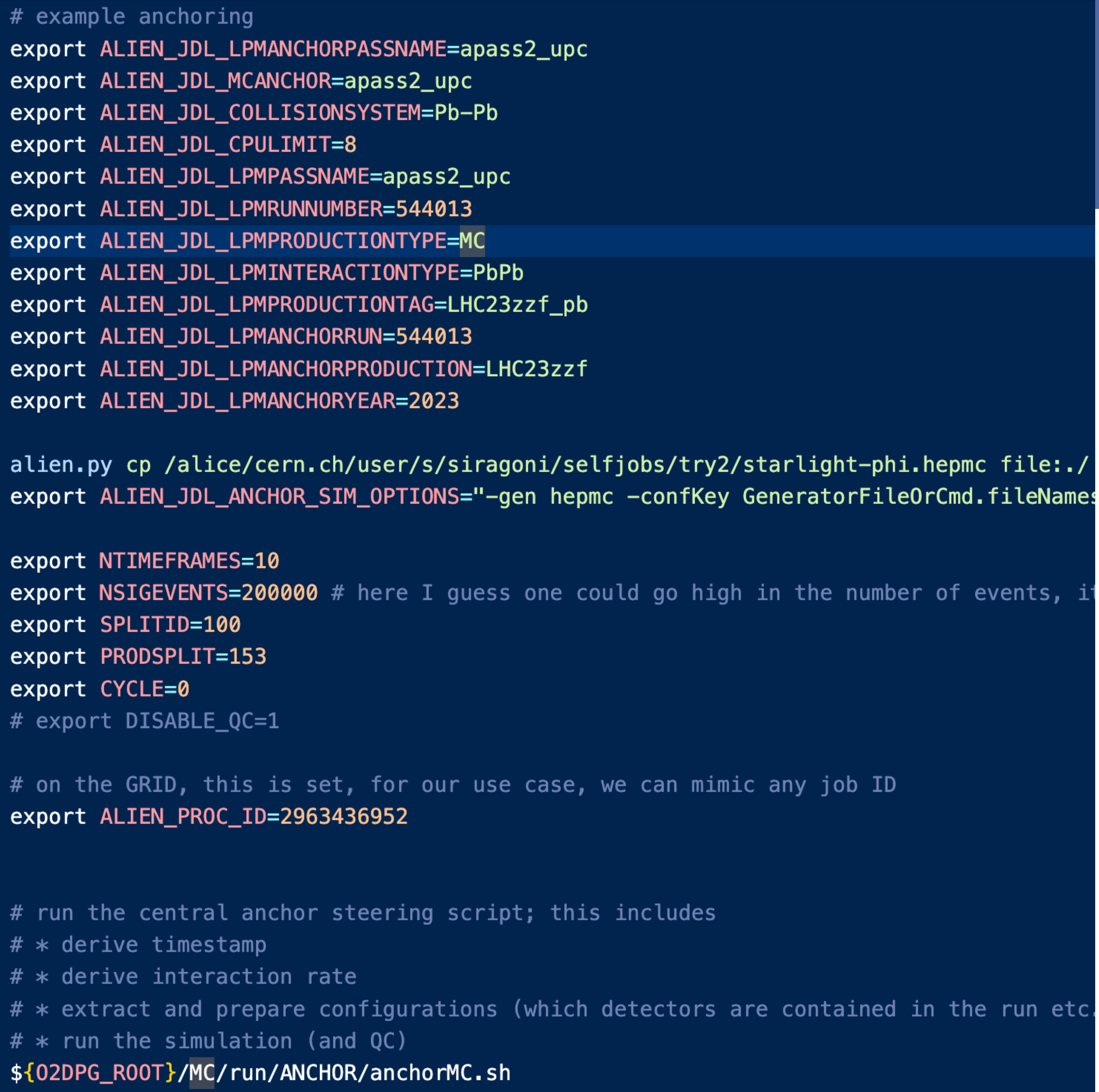
Permissions	Owner	Timestamp	Size	Filename
-rwxr-xr-x	siragoni:siragoni	today 10:09	1.274 KB	alien_jobscript.sh
-rwxr-xr-x	siragoni:siragoni	today 09:29	580 B	testPhi-20240403-070924.jdl
-rwxr-xr-x	siragoni:siragoni	today 09:49	23.41 KB	testPhi-20240403-070924.sh

Edit new file — **25.25 KB in 3 files**

Upload files in this folder (500MB max, multiple selection possible)

Choose Files No file chosen — Upload...

The anchoring script (for PbPb)



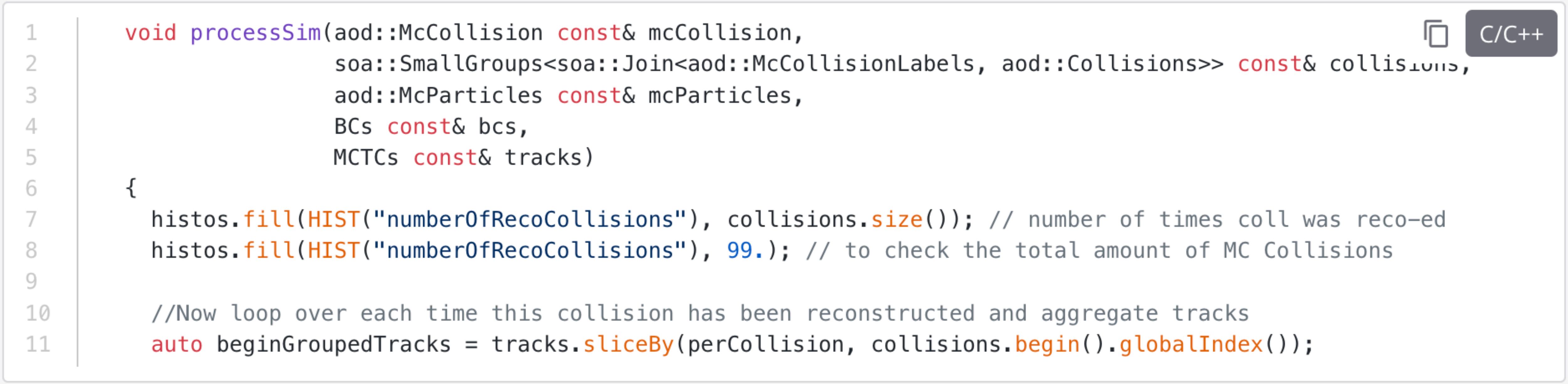
```
# example anchoring
export ALIEN_JDL_LPMANCHORPASSNAME=apass2_upc
export ALIEN_JDL_MCANCHOR=apass2_upc
export ALIEN_JDL_COLLISIONSYSTEM=Pb-Pb
export ALIEN_JDL_CPULIMIT=8
export ALIEN_JDL_LPMPASSNAME=apass2_upc
export ALIEN_JDL_LPMRUNNUMBER=544013
export ALIEN_JDL_LPMPRODUCTIONTYPE=MC
export ALIEN_JDL_LPMINTERACTIONTYPE=PbPb
export ALIEN_JDL_LPMPRODUCTIONTAG=LHC23zzf_pb
export ALIEN_JDL_LPMANCHORRUN=544013
export ALIEN_JDL_LPMANCHORPRODUCTION=LHC23zzf
export ALIEN_JDL_LPMANCHORYEAR=2023

alien.py cp /alice/cern.ch/user/s/siragoni/selfjobs/try2/starlight-phi.hepmc file:./
export ALIEN_JDL_ANCHOR_SIM_OPTIONS="-gen hepmc -confKey GeneratorFileOrCmd.fileNames

export NTIMEFRAMES=10
export NSIGEVENTS=200000 # here I guess one could go high in the number of events, it
export SPLITID=100
export PRODSPLIT=153
export CYCLE=0
# export DISABLE_QC=1

# on the GRID, this is set, for our use case, we can mimic any job ID
export ALIEN_PROC_ID=2963436952


# run the central anchor steering script; this includes
# * derive timestamp
# * derive interaction rate
# * extract and prepare configurations (which detectors are contained in the run etc.
# * run the simulation (and QC)
${O2DPG_ROOT}/MC/run/ANCHOR/anchorMC.sh
```

NOTE: anchoring for Pb-Pb not fully understood yet, issues with the number of TFs and injected signal (still work in progress)

Analysing that AOD

Automatic grouping GEN-RECO

You can do it like this:



```cpp
void processSim(aod::McCollision const& mcCollision,
                soa::SmallGroups<soa::Join<aod::McCollisionLabels, aod::Collisions>> const& collisions,
                aod::McParticles const& mcParticles,
                BCs const& bcs,
                MCTCs const& tracks)
{
  histos.fill(HIST("numberOfRecoCollisions"), collisions.size()); // number of times coll was reco-ed
  histos.fill(HIST("numberOfRecoCollisions"), 99.); // to check the total amount of MC Collisions

  //Now loop over each time this collision has been reconstructed and aggregate tracks
  auto beginGroupedTracks = tracks.sliceBy(perCollision, collisions.begin().globalIndex());
```

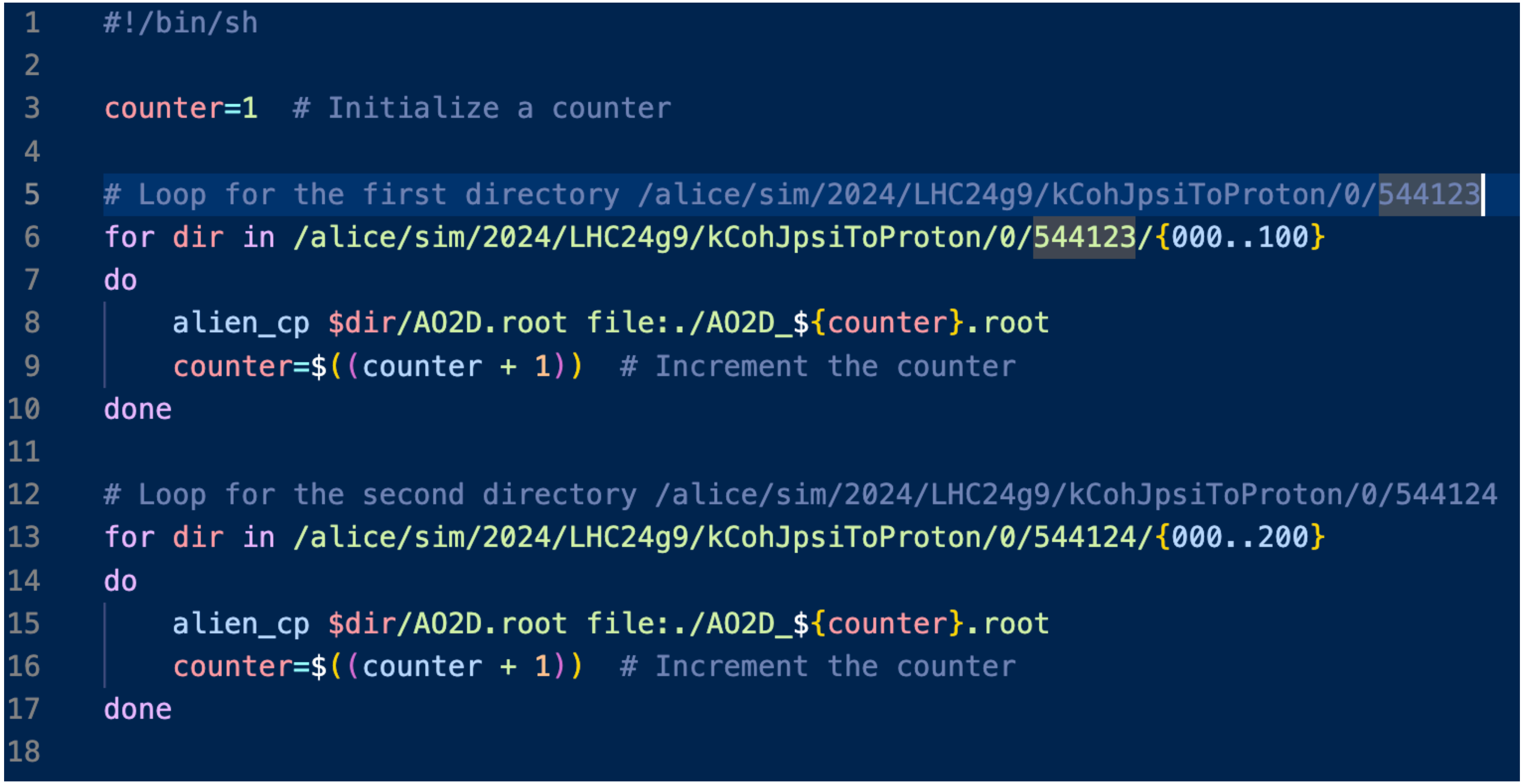
Run 3 O2 production using CENTRAL tools

- Run by Michal with a small production injecting events generated with STARlight DIRECTLY (not hepMC-fed)
- O2 ticket: https://its.cern.ch/jira/browse/O2-5292
- Quite a few productions!
    - Coh $J/\psi \rightarrow p\bar{p}$
    - Coh $J/\psi \rightarrow \mu\mu$
    - Coh $\rho \rightarrow \pi\pi$
    - STARlight+DPMJET

Run 3 O2 production using CENTRAL tools

- STARlight+DPMJET news: production with issues, it is possible that somehow for central tools the automatic splitting of the hepMC files is not enabled (it is the only production that occurred with hepMC injection)
- The next few slides will focus on the coh $J/\psi \to p\bar{p}$
- Two folders:
  - /alice/sim/2024/LHC24g9/kCohJpsiToProton/0/544123/
  - /alice/sim/2024/LHC24g9/kCohJpsiToProton/0/544124/
  - Note that the second link requires to be treated differently using the MCconverter

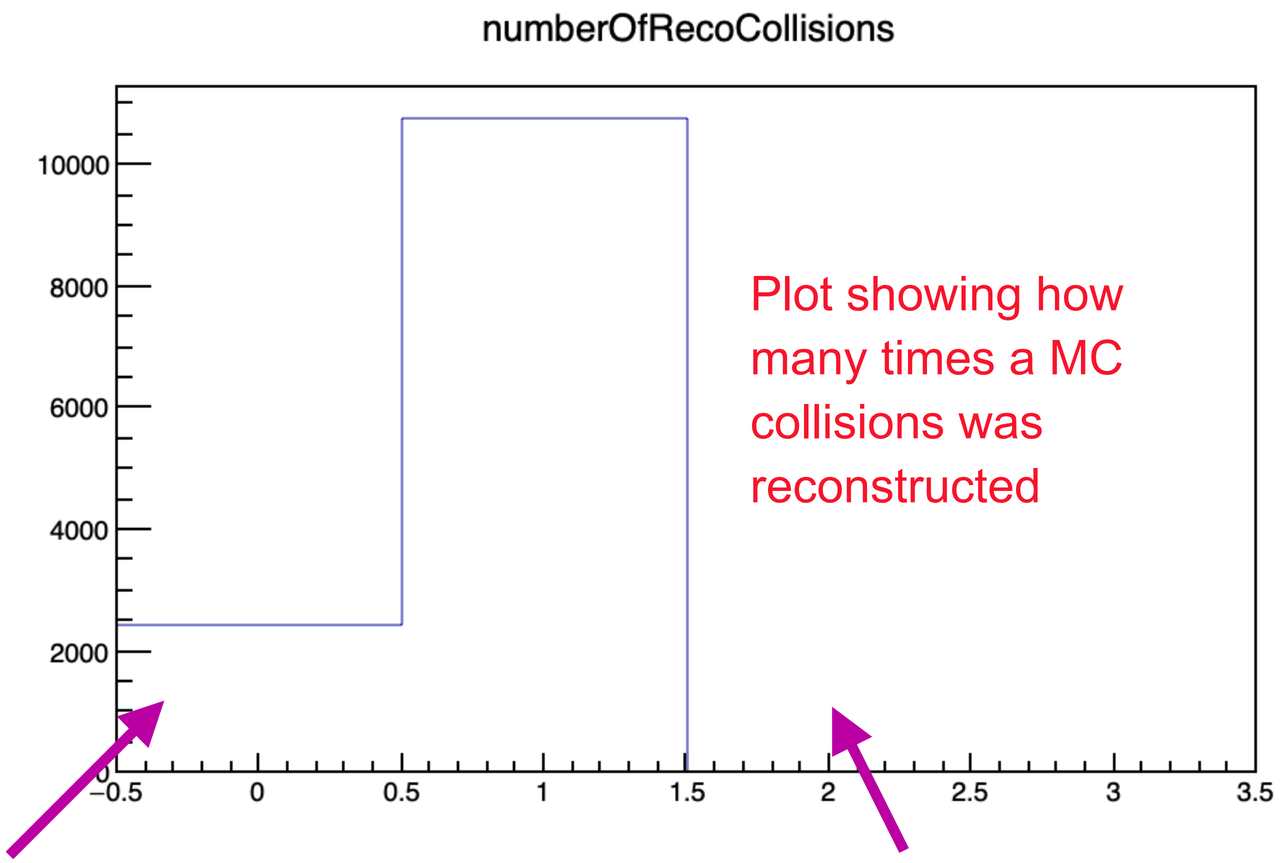
How to download them



```sh
#!/bin/sh

counter=1  # Initialize a counter

# Loop for the first directory /alice/sim/2024/LHC24g9/kCohJpsiToProton/0/544123
for dir in /alice/sim/2024/LHC24g9/kCohJpsiToProton/0/544123/{000..100}
do
    alien_cp $dir/AO2D.root file:./AO2D_${counter}.root
    counter=$((counter + 1))  # Increment the counter
done

# Loop for the second directory /alice/sim/2024/LHC24g9/kCohJpsiToProton/0/544124
for dir in /alice/sim/2024/LHC24g9/kCohJpsiToProton/0/544124/{000..200}
do
    alien_cp $dir/AO2D.root file:./AO2D_${counter}.root
    counter=$((counter + 1))  # Increment the counter
done

```

Run 3 MC coh $J/\psi \to p\bar{p}$

- Strategy: download the files on local, about 500 MB worth of them
- Parse the reco and gen data directly, i.e. without skimmer (ok for MC simulations, i.e. no central or peripheral collisions)



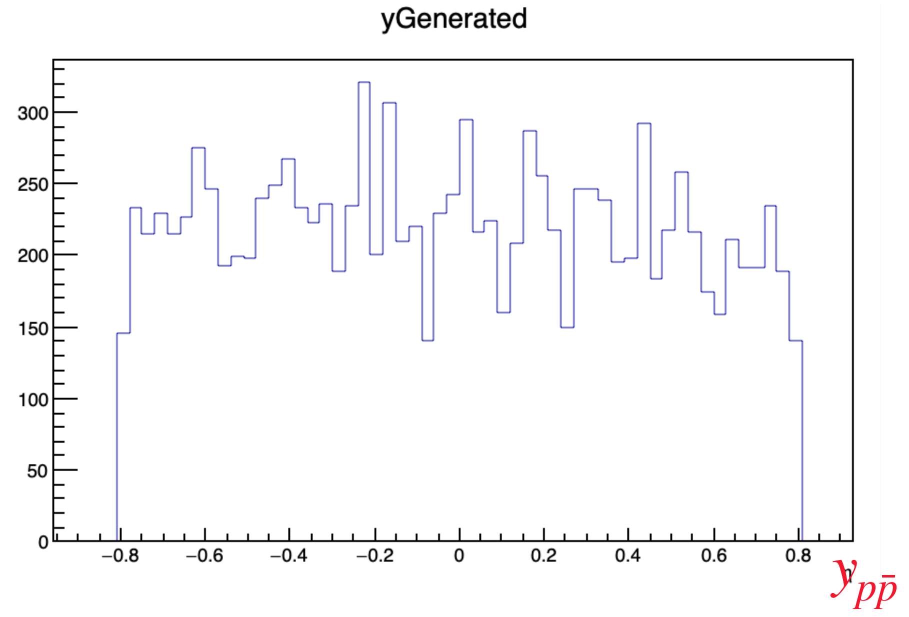
Plot showing how many times a MC collisions was reconstructed

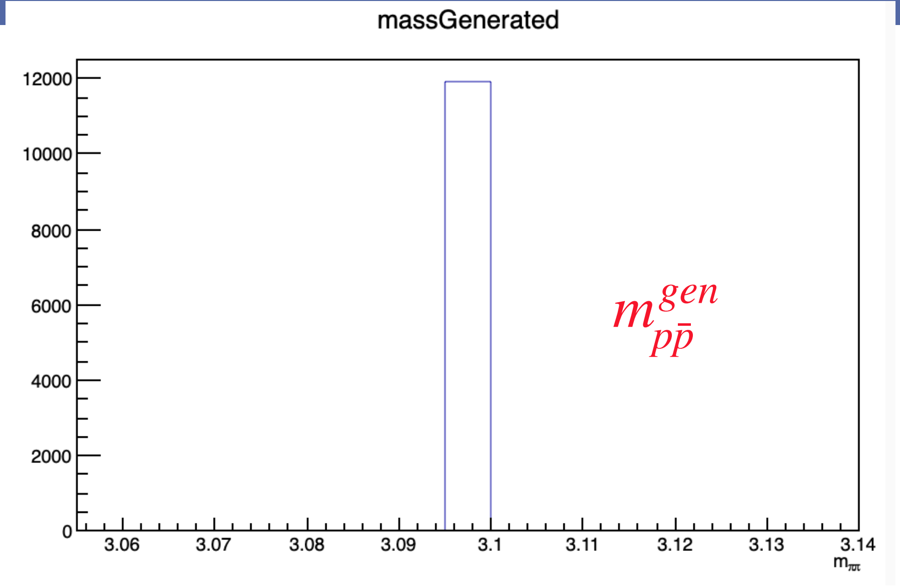
Only 18% of the collisions were never reconstructed (number of reco collisions = 0)!

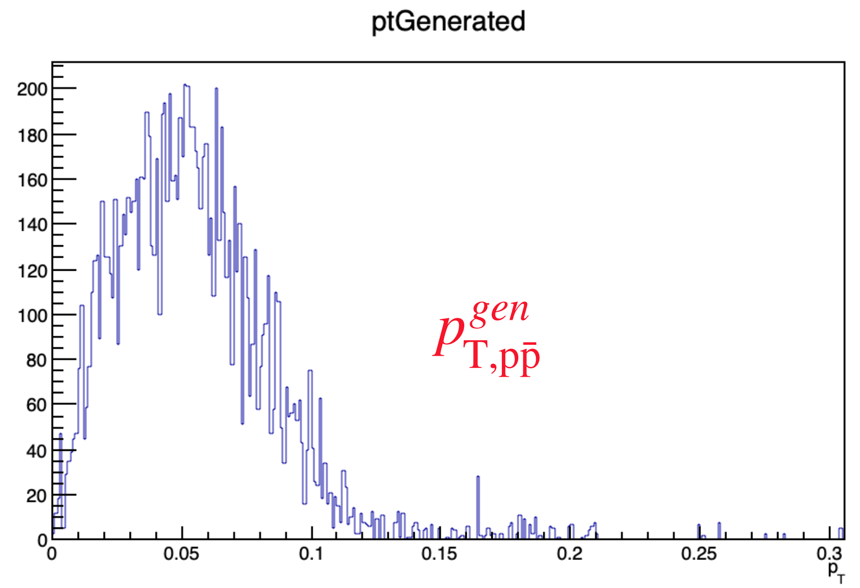
No collisions were reconstructed more than once (number of reco collisions > 1)!

Run 3 MC coh $J/\psi \rightarrow p\bar{p}$: generated selections

- isPhysicalPrimary() flag
- PDG code = ±2212
- $|y_{p\bar{p}}| < 0.8$



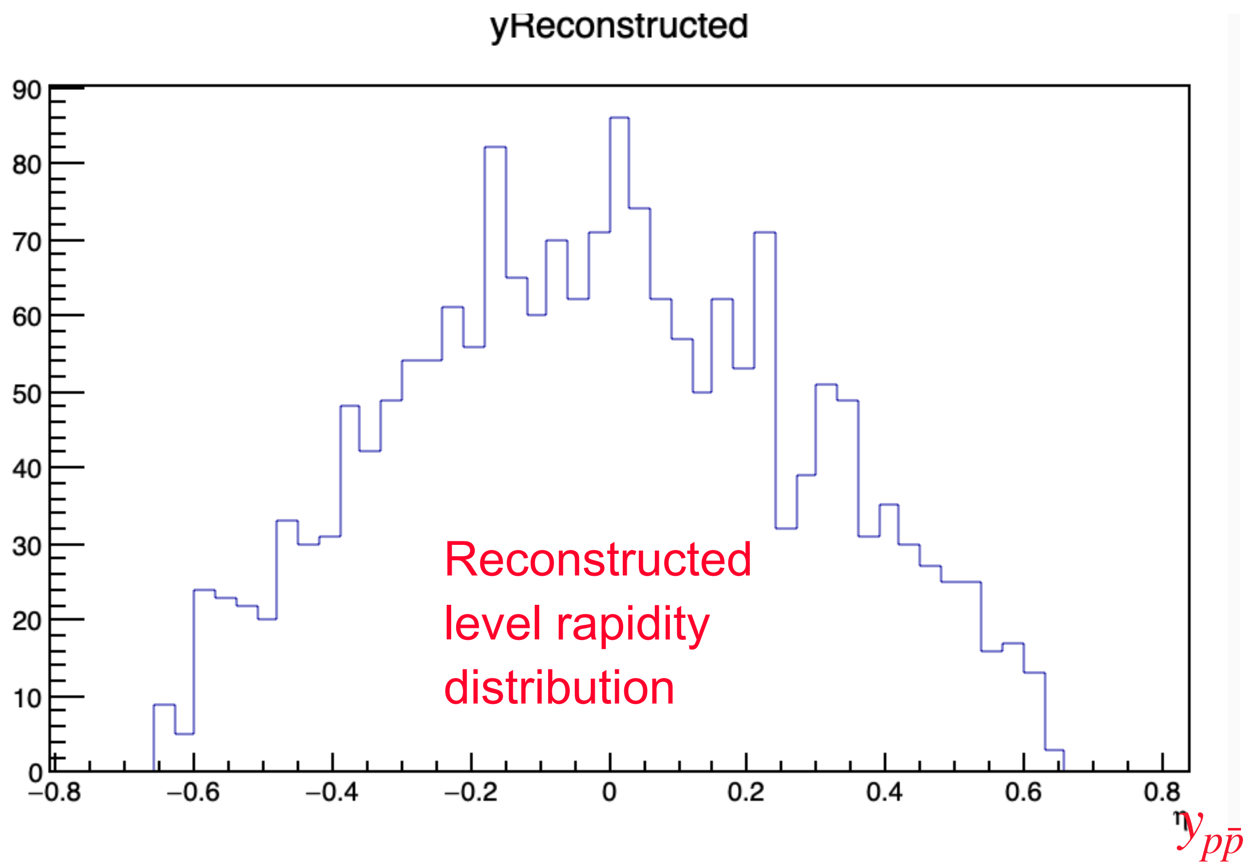




The generated level looks good

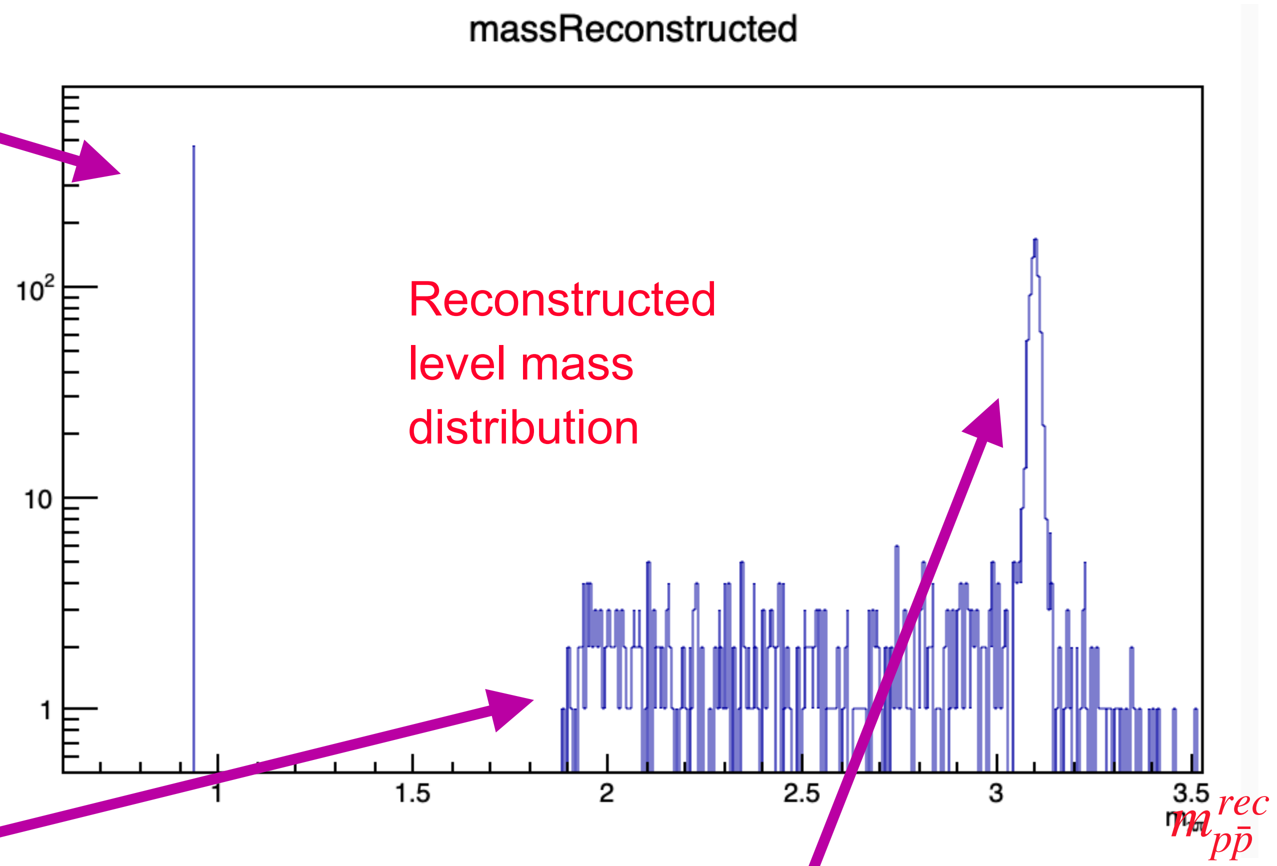
Run 3 MC coh $J/\psi \to p\bar{p}$: reconstructed selections

- Automatic grouping, i.e. automatic association of the reconstructed collision to the generated level
- Track is PV contributor
- $p_{\mathrm{T,track}} > 0.4$ GeV/c
- $N\sigma_p^{TPC} < 3$
- Exactly 2 opposite sign tracks
- $|y_{single\ proton}| < 0.8$
- $|y_{p\bar{p}}| < 0.8$



Run 3 MC coh $J/\psi \rightarrow p\bar{p}$: reconstructed selections

Not sure what is this sharp peak, if it had been that one of the protons was not reconstructed I would have expected it to be smeared…

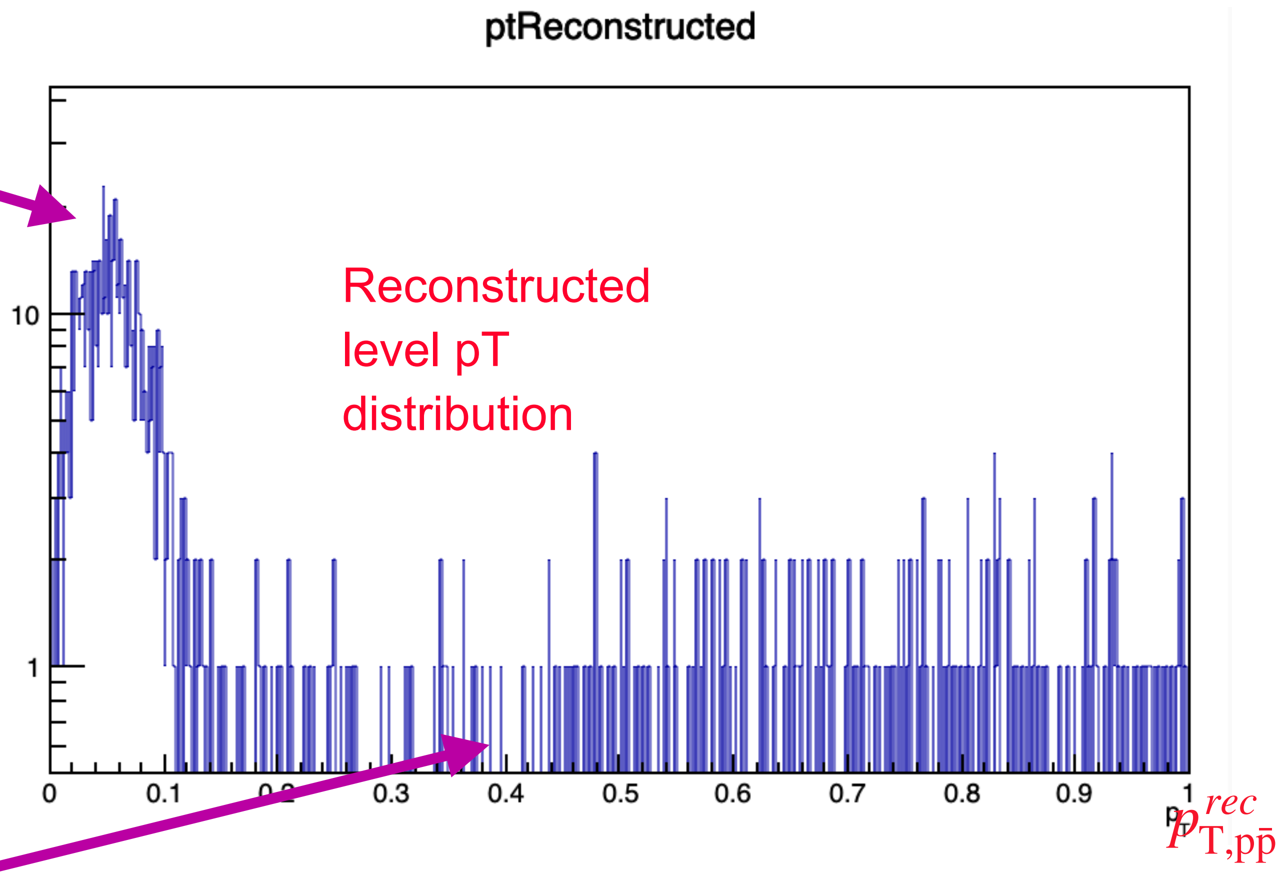


Possibly secondary protons or protons from the beam reconstructed as primary protons? When one of the two protons from the Jpsi don't make it in the acceptance?

Otherwise Jpsi peak ok

Run 3 MC coh $J/\psi \rightarrow p\bar{p}$: reconstructed selections

Jpsi peak ok



The long tail seems to support that these were possibly secondary protons or protons from the beam reconstructed as primary protons

Requesting central MC productions

Procedure

- Make a case at a PAG meeting
- If approved, go to PWG level
- If approved, the PWG conveners will take it to Physics Board
- If PB approves (no reason it shouldn't for UPC productions, i.e. very small productions) you are ready to go

- Create a JIRA ticket
  - https://its.cern.ch/jira/browse/O2-5292 copy from here, or clone it to start with
  - https://its.cern.ch/jira/browse/O2-4591 for a more complete example

Backup slides