Data analysis at CMS
-- Exotic hadron study as an example

Jingqing Zhang

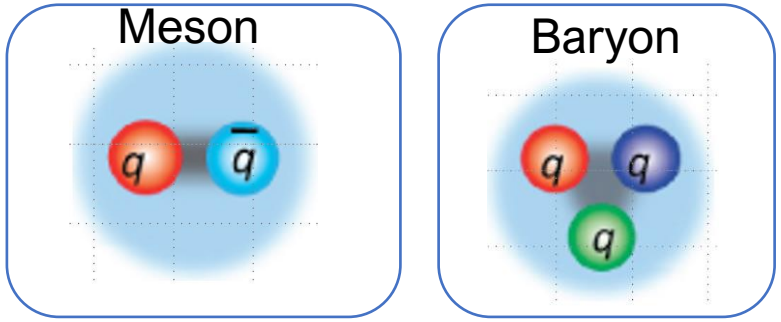
Nanjing Normal University

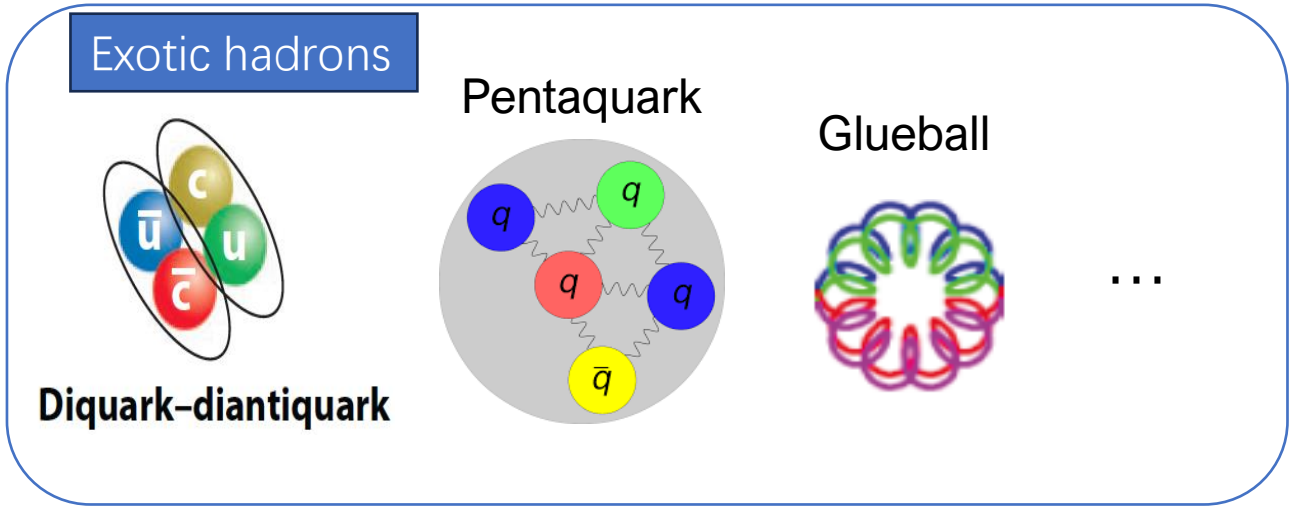
Outline

- Motivation of the analysis
- Steps of an analysis: from scratch to the end
- Dataset to ntuple
  - Analyzer to make ntuple
- Analyze ntuple
  - Background
  - Fit the mass spectrum
  - Significance

Motivation

- What is the exotic hadron





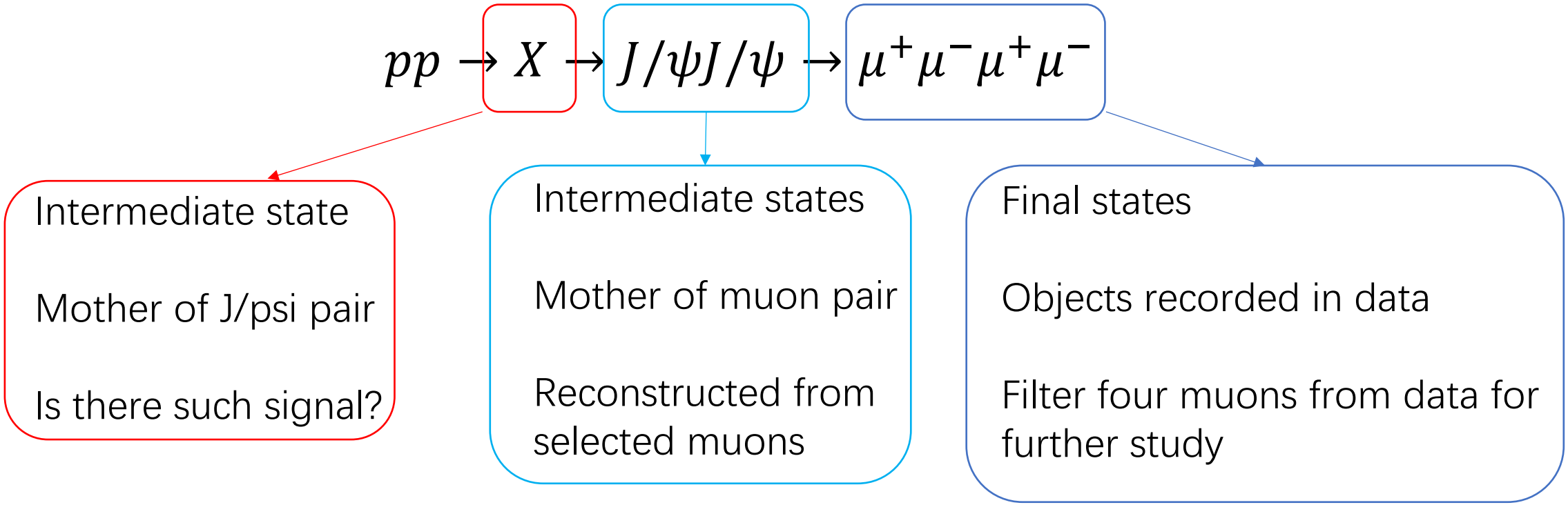
- Why we want to study exotic hadron
- Why at CMS
  - CMS can provide data with high production energy
  - And can provide more spin possibility than ee collider

What channel we want to study

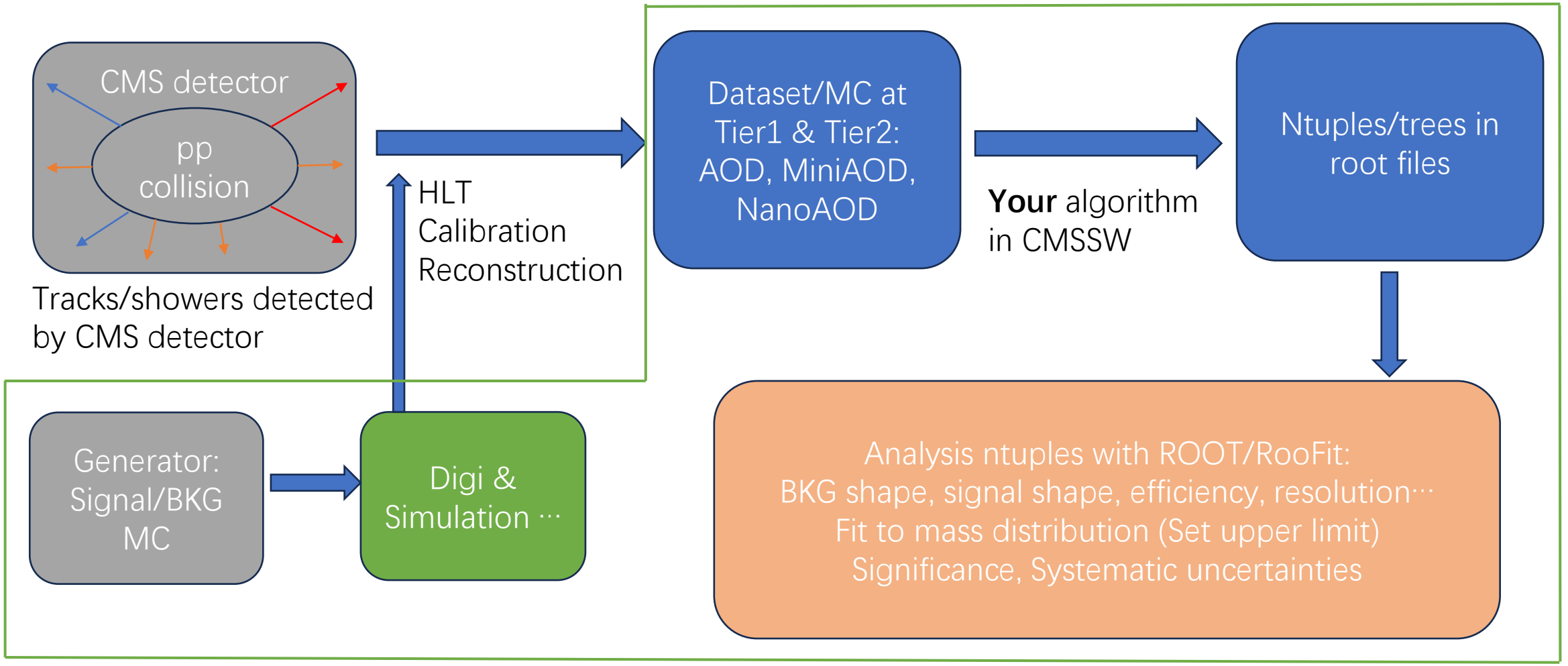
- J/ψ can be easily tagged/reconstructed
- Resonances in $J/\psi J/\psi$ most likely contains at least 4 charm quarks
  - Otherwise decays to $J/\psi J/\psi$ will be suppressed
- Study the channel: $J/\psi J/\psi \rightarrow \mu^+\mu^-\mu^+\mu^-$

How to do this analysis from scratch

- We now have chosen the channel:

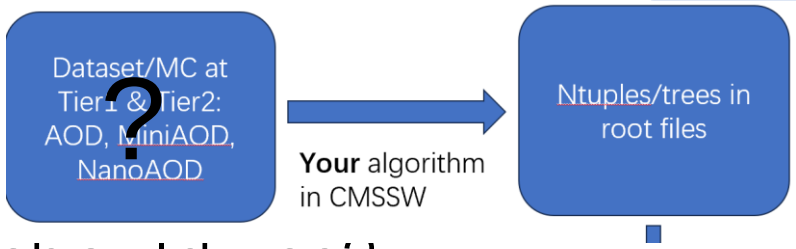


How to do the analysis from scratch

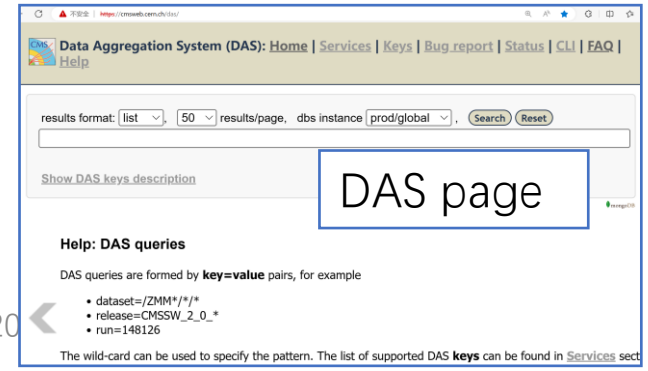


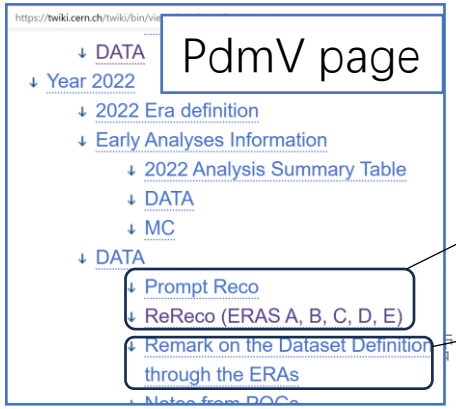
What/where is the data

- Our channel: $pp \rightarrow X \rightarrow J/\psi J/\psi \rightarrow \mu^+\mu^-\mu^+\mu^-$



- What data we should use?
  - Unfortunately, no webpage which contains all datasets and their triggers
  - Ask supervisor, colleagues, convenors
  - Lookup in DAS: CMS Data Aggregation Service (cern.ch)
  - Explore in PdmV: PdmVRun3Analysis < CMS < TWiki (cern.ch)





Links to dataset in DAS

Applied triggers
(≈ what events in datasets)

What/where is the data

- From PdmV:

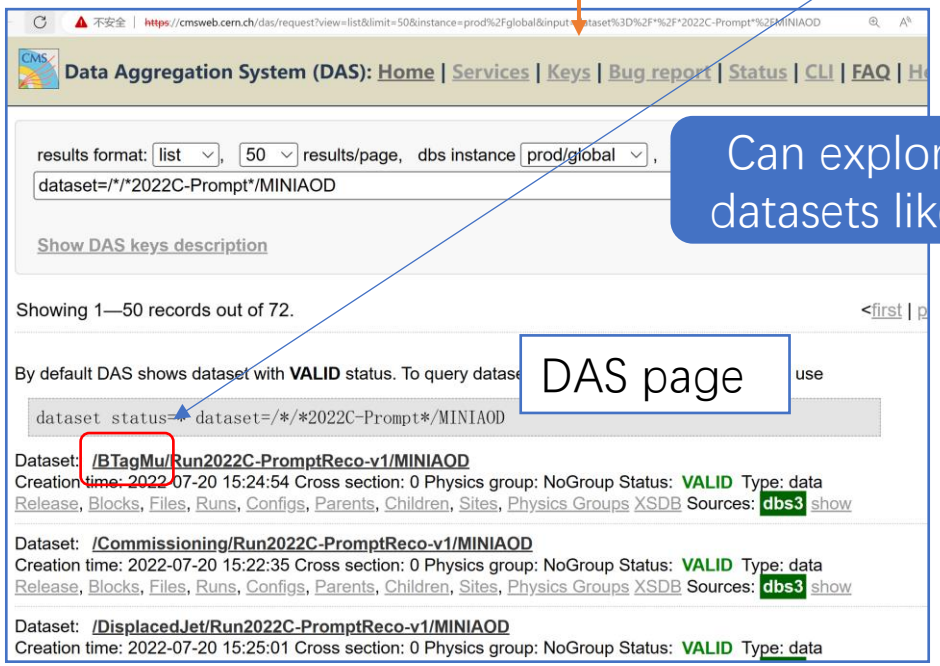
Prompt Reco

ERA	MINIAOD	NANOAOD
C	DAS	DAS
D	DAS	DAS
E	DAS	DAS
F	DAS	DAS
G	DAS	DAS

ERAs	!HLT menu	PDs set
A		
B		
C	v1.2.0, and v1.2.5	ZeroBias, MinimumBias, EGamma, BTagMu, DisplacedJet, JetHT, MET, Tau, DoubleMuon, MuonEG, SingleMuon, ScoutingPFMonitor, ScoutingPFRun3, ParkingBPH [1-5], ParkingDoubleElectronLowMass [0-5], ParkingDoubleMuonLowMass [0-7]
D	v1.2.5	ZeroBias, MinimumBias, EGamma, BTagMu

HLT Menu

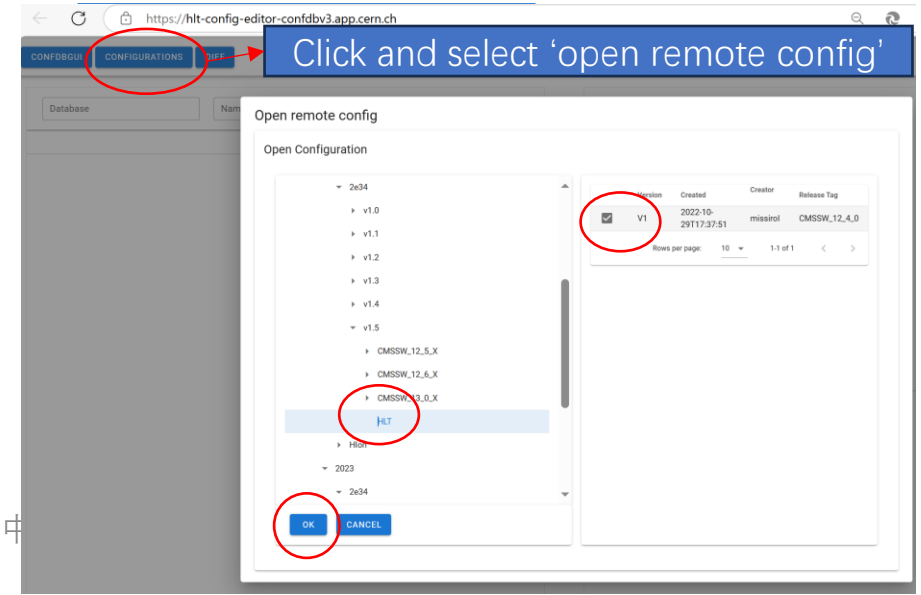
Primary name of dataset



Can explore all datasets like this

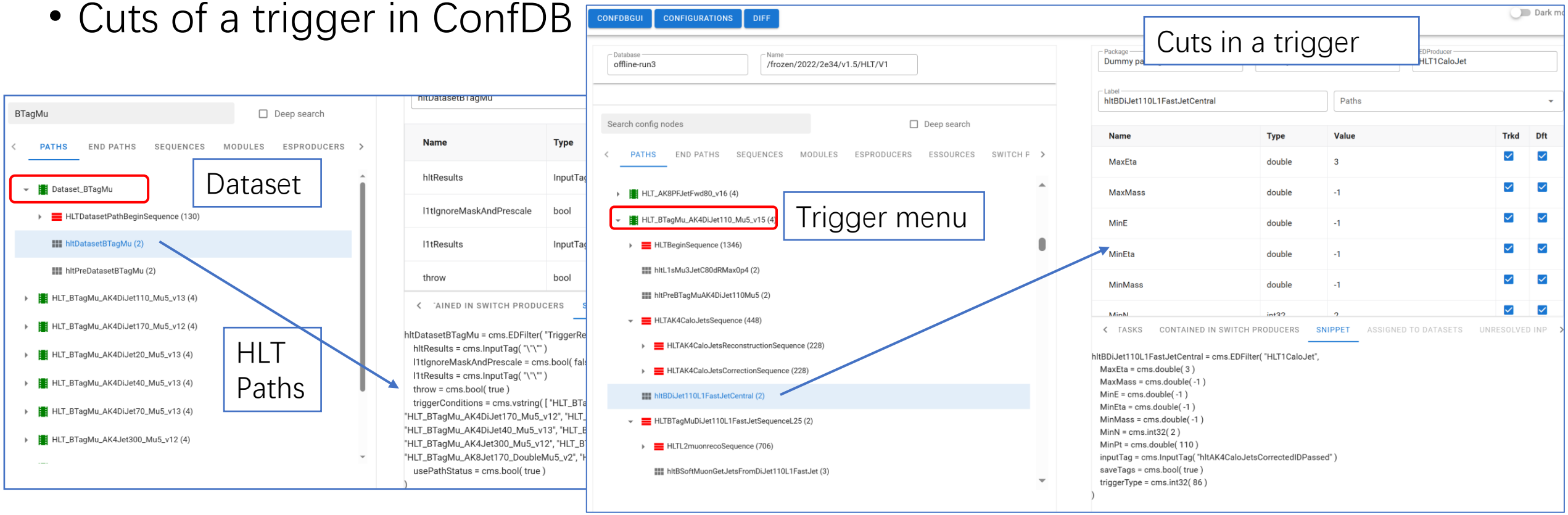
DAS page

Then check the triggers/cuts of a primary dataset in confDB: ConfDB < CMS < TWiki (cern.ch)
→ ConfDB Web interface



What/Where is the data

- Trigger menus of a dataset in ConfDB
- Cuts of a trigger in ConfDB



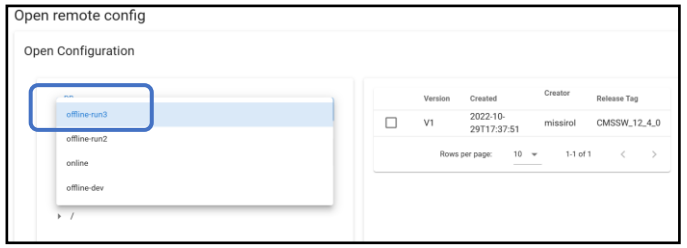
- A dataset containing J/psi:

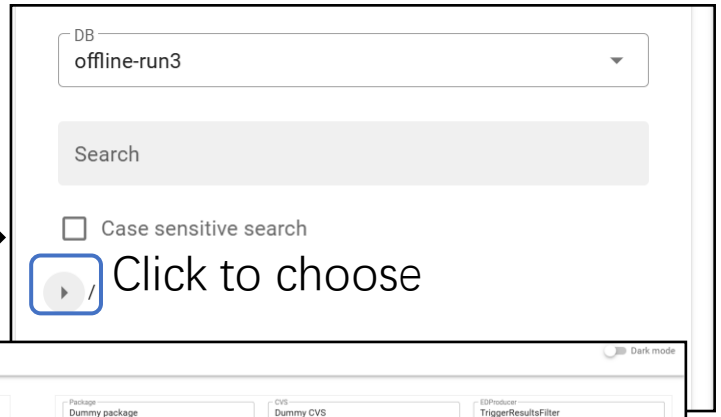
/ParkingDoubleMuonLowMass0/Run2022C-PromptReco-v1/MINIAOD **Exercise: how many trigger paths containing J/psi for this dataset?**

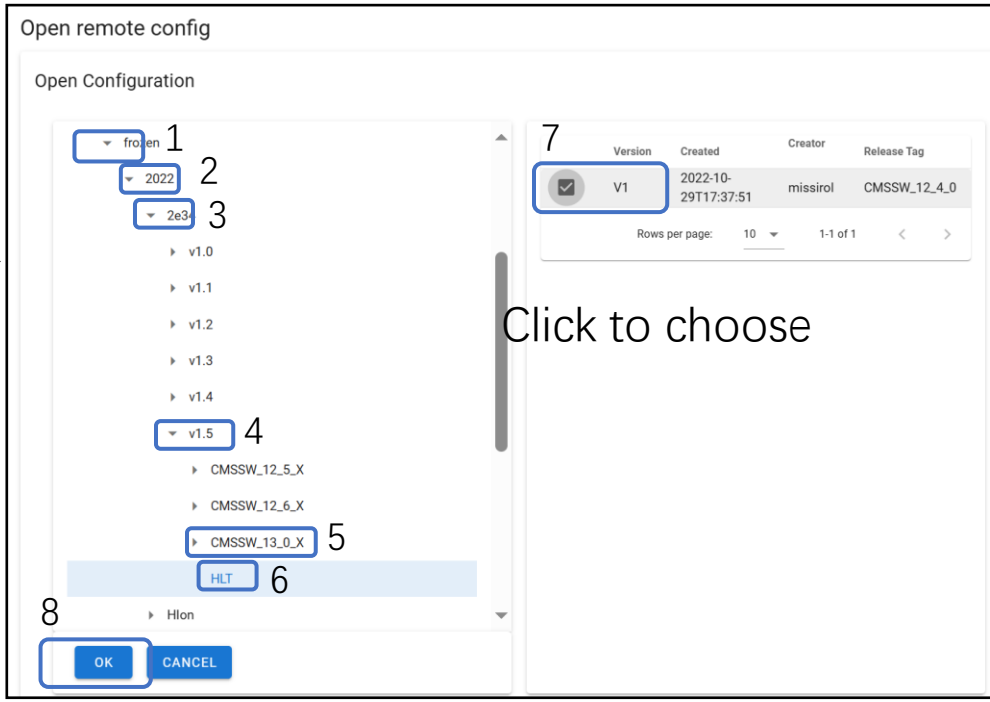
Exercise: how many trigger paths containing J/psi for this dataset?

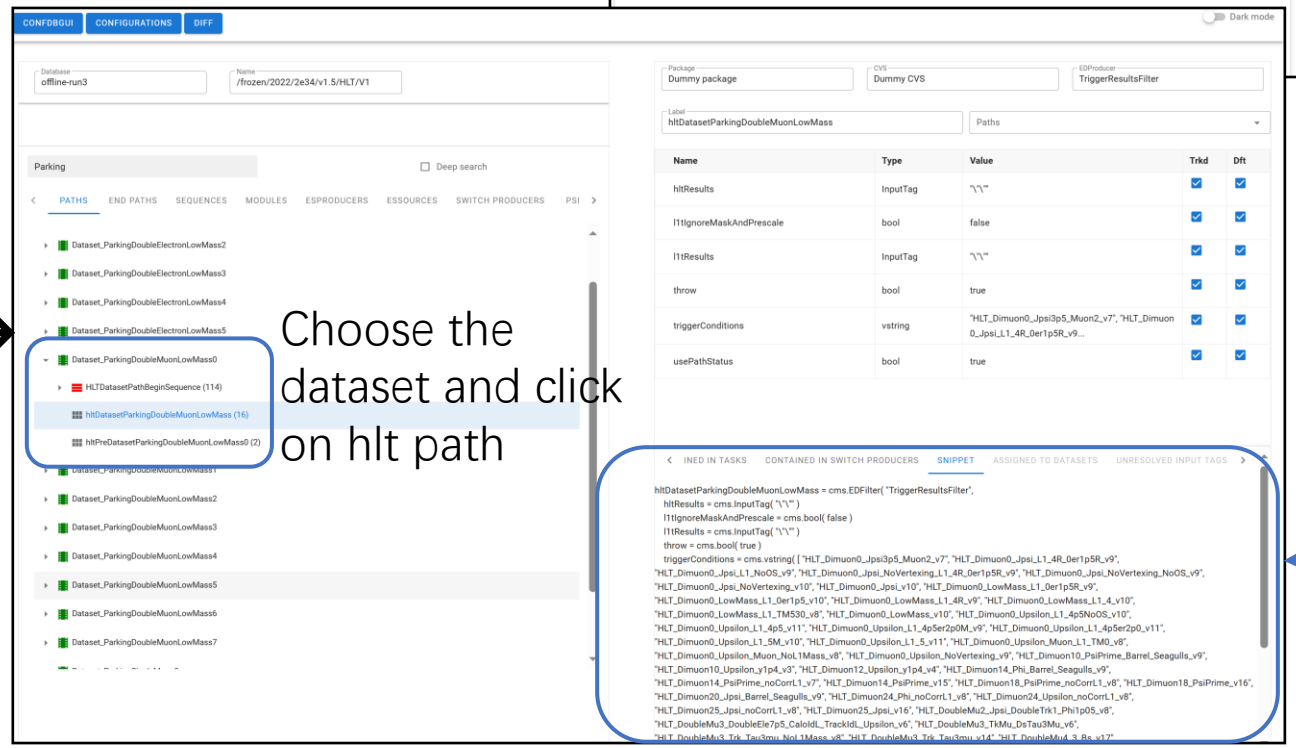
- ConfDB Web interface → Configurations → Open remote →

Make a selection









CMS HLT Info tool

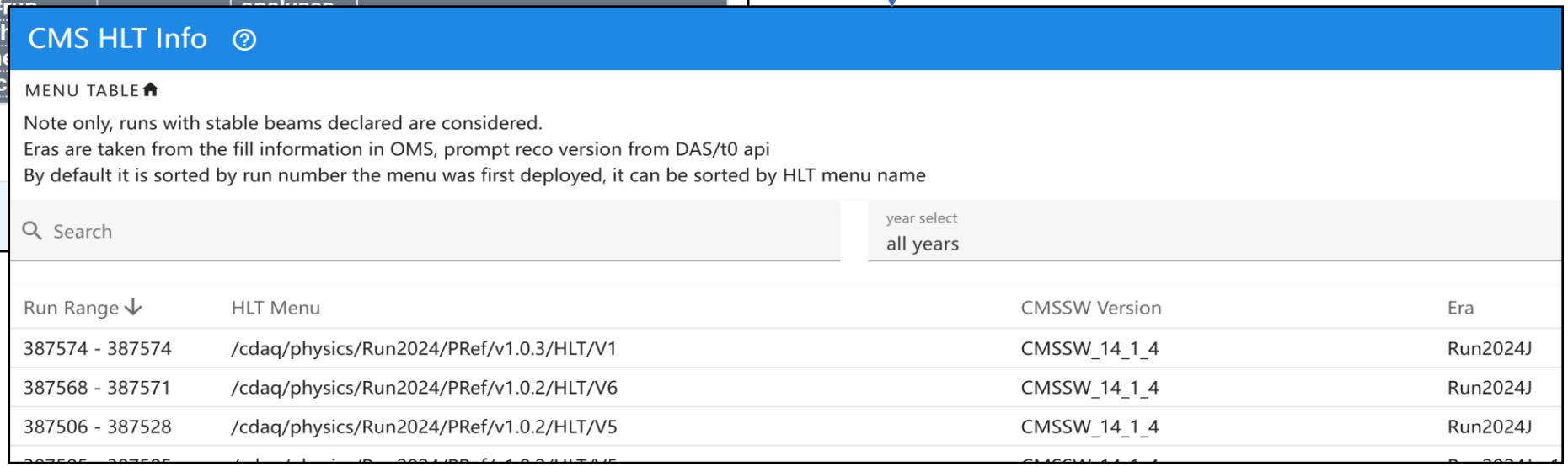
2023 Era definition

DISCLAIMER: Run ranges are based on the Era/procVersion change in Tier0 processing, hence on the PromptReco datasets.

Tier0 has an API for accessing historical information on Eras, Tier0 configurations, PDs and more: T0 API Entrypoints

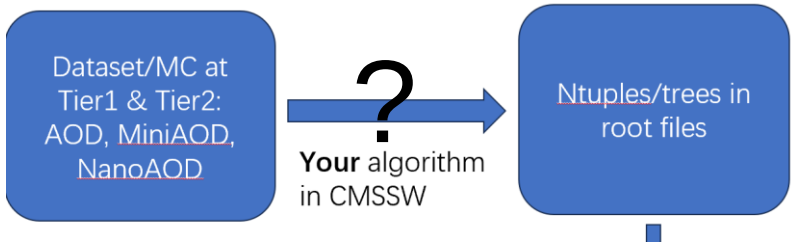
You can find out the HLT Menu related to a given Era, CMSSW version and run range, by using the CMS HLT Info tool .

Era	First Run	Last Run	HLT Menu (# wi me ac	Reference	dataset for	Comments
Commissioning2023 -v1	363380	364666				
Commissioning2023 -v2	364667	365738				



From dataset to ntuple

- Signal channel: $pp \rightarrow X \rightarrow J/\psi J/\psi \rightarrow \mu^+\mu^-\mu^+\mu^-$
- Dataset: /ParkingDoubleMuonLowMass0/Run2022C-PromptReco-v1/MINIAOD



- Write an analyzer, analyze data and produce ntuples: WorkBookWriteFrameworkModule < CMSPublic < TWiki (cern.ch)
- Setup env

```
csh or tcsh users:
  source /cvmfs/cms.cern.ch/cmsset_default.csh
bash users:
  source /cvmfs/cms.cern.ch/cmsset_default.sh
```

- Setup CMSSW

```
cmsrel CMSSW_13_0_13
cd CMSSW_13_0_13/src
cmsenv
```

Write an analyzer

- Create an analyzer package

Write a Framework Module

First, create a subsystem area. The actual name used for the directory is not important, we'll use `Demo`. From the `src` directory, make and change to the `Demo` area:

```
mkdir Demo
cd Demo
```

Note that if you do not create the subsystem area and create you module directly under the `src` directory, your code will not compile. Create the "skeleton" of an EDAnalyzer module (see SWGuideSkeletonCodeGenerator for more information):

```
mkedanlzr DemoAnalyzer
```

Compile the code:

```
cd DemoAnalyzer
scram b
```

- Create a ConfFile_cfg.py in DemoAnalyzer/python, example link
- Run the job (at present it does not work, due to invalid input AOD file; we will come back)

```
cd into the DemoAnalyzer/python directory and do:
cmsRun ConfFile_cfg.py
```

Write analyzer

- Analyzer module is executed to filter data and make ntuple

cmsRun ConfFile_cfg.py

```
import FWCore.ParameterSet.Config as cms

process = cms.Process("Demo")

process.load("FWCore.MessageService.MessageLogger_cfi")

process.maxEvents = cms.untracked.PSet( input = cms.untracked.int32(-1) )

process.source = cms.Source("PoolSource",
                            # replace 'myfile.root' with the source file you want to use
                            fileNames = cms.untracked.vstring(
        'file:/afs/cern.ch/cms/Tutorials/workbook_twiki2021/MinBias_pythia8_14TeV_100events.root'
            )

process.demo = cms.EDAnalyzer('DemoAnalyzer',
    tracks    = cms.untracked.InputTag('generalTracks')
                            )

process.p = cms.Path(process.demo)
```

Input data files

The analyzer module/package

Input file should contain contents of 'generalTracks',
Which is in AOD, but not in MiniAOD

To run the whole dataset, you need run crab job instead of running it locally.

This config file then is an input of the crab job config file, which uses the dataset name to identify data

- **Exercise: create a DemoAnalyzer following above instruction**

Write an analyzer

- Write codes in the DemoAnalyzer then compile and cmsRun
- Save a histogram and a tree in output
- DemoAnalyzer/plugins/DemAnalyzer.cc

1 header files:

```
34 ////added to use service and save ntuple/histogram
35 #include "FWCore/ServiceRegistry/interface/Service.h"
36 #include "CommonTools/UtilAlgos/interface/TFileService.h"
37 #include "DataFormats/PatCandidates/interface/Muon.h"
38 #include "TH1.h"
39 #include "TTree.h"
40 #include <vector>
41 ////
```

1

2 Member data in class definition:

```
66   // ----------member data ---------------------------
67   //edm::EDGetTokenT<TrackCollection> tracksToken_;  //u
   to read from configuration file
68   edm::EDGetTokenT<std::vector<pat::Muon>> muonsToken_;
69   TH1F *h_pt;//to save in file
70   TTree *tree1;
71   unsigned int runNum;
72   unsigned int lumiNum;
73   unsigned int eventNum;
74   int nTrack;
75   std::vector<double> *vec_pt;
76   std::vector<double> *vec_px;
77   std::vector<double> *vec_py;
78   std::vector<double> *vec_pz;
```

2

3 Constructor

```
96 DemoAnalyzer::DemoAnalyzer(const edm::ParameterSet& iConfig)
97     //: tracksToken_(consumes<TrackCollection>(iConfig.getUntrackedParameter<edm::
   InputTag>("tracks"))),
98     : muonsToken_(consumes<std::vector<pat::Muon>>(edm::InputTag("slimmedMuons")))
   ,
99     runNum(0), lumiNum(0), eventNum(0), nTrack(0), vec_pt(0), vec_px(0), vec_py(0)
   , vec_pz(0) {
100   //to save histograms/ntuples
101   edm::Service<TFileService> fs;
102   h_pt = fs->make<TH1F>("h_pt", "title of h_pt", 100, 0, 100);
103   tree1 = fs->make<TTree>("tree1", "my tree 1");
104   tree1->Branch("runNum", &runNum, "runNum/i");
105   tree1->Branch("lumiNum", &lumiNum, "lumiNum/i");
106   tree1->Branch("eventNum", &eventNum, "eventNum/i");
107   tree1->Branch("nTrack", &nTrack, "nTrack/I");
108   tree1->Branch("vec_pt", &vec_pt);
109   tree1->Branch("vec_px", &vec_px);
110   tree1->Branch("vec_py", &vec_py);
111   tree1->Branch("vec_pz", &vec_pz);
```

3

```
root [2] demo->cd()
(bool) true
root [3] .ls
TDirectoryFile*         demo    demo
 KEY: TH1F      h_pt;1  h_pt
 KEY: TTree     tree1;1 my tree 1
root [4]
```

'generalTracks' is a label of AOD contents.
'slimmedMuons' is a label of MiniAOD
We make this analyzer to read MiniAOD file.

Example: /publicfs/cms/user/zhangjq/cms2025beihang/CMSSW_13_0_13/src/Demo/DemoAnalyzer/plugins/DemoAnalyzer.cc

Write an analyzer

4 fill histograms and trees in analyze()

5 Modify DemoAnayzer/plugins/BuildFile.xml

```
133   ////event level
134   runNum = iEvent.id().run();
135   lumiNum = iEvent.id().luminosityBlock();
136   eventNum = iEvent.id().event();
137   nTrack = 0;
138   // for (const auto& track : iEvent.get(tracksToken_)) {
139   for (const auto & track : iEvent.get(muonsToken_)) {
140     // do something with track parameters, e.g, plot the charge.
141     // int charge = track.charge();
142     if (track.charge() < 0) continue;
143     nTrack++;
144     ////track/candidate level
145     h_pt->Fill(track.pt());
146     vec_pt->push_back(track.pt());
147     vec_px->push_back(track.px());
148     vec_py->push_back(track.py());
149     vec_pz->push_back(track.pz());
150   }
151   tree1->Fill();
152   //need to clear the vector after this event/candidate is fill to the tree
153   runNum = 0;
154   lumiNum = 0;
155   eventNum = 0;
156   nTrack = 0;
157   vec_pt->clear();
158   vec_px->clear();
159   vec_py->clear();
160   vec_pz->clear();
```

4

```
1 <use name="DataFormats/TrackReco"/>
2 <use name="FWCore/Framework"/>
3 <use name="FWCore/ParameterSet"/>
4 <use name="FWCore/PluginManager"/>
5 <use name="CommonTools/UtilAlgos"/>
6 <use name="FWCore/ServiceRegistry"/>
7 <flags EDM_PLUGIN="1"/>
```

5

We comment out ‘generalTracks’ (to read reco::Track of AOD) related contents, and add ‘slimmedMuons’ to read pat::Muon of MiniAOD

Example: /publicfs/cms/user/zhangjq/cms2025beihang/CMSSW_13_0_13/src/Demo/DemoAnalyzer/plugins/DemoAnalyzer.cc

Run the analyzer

- Add output part in DemoAnalyzer/python/ConfFile_cfg.py
- cd DemoAnalyzer/; scram b; cd python; cmsRun ConfFile_cfg.py

```python
import FWCore.ParameterSet.Config as cms

process = cms.Process("Demo")

process.load("FWCore.MessageService.MessageLogger_cfi")

process.maxEvents = cms.untracked.PSet( input = cms.untracked.int32(100) )

process.source = cms.Source("PoolSource",
                            # replace 'myfile.root' with the source file you want to use
                            fileNames = cms.untracked.vstring(
        'file:/publicfs/cms/user/zhangjq/cms2025beihang/data/418cda25-b45c-4e9a-b740-6e35e6dd57db.root'
#         '/store/data/Run2023B/ParkingDoubleMuonLowMass0/MINIAOD/22Sep2023-v1/2560000/418cda25-b45c-4e9a-b740-6e35e6dd57db.root'
                )
                        )

process.TFileService = cms.Service("TFileService",
                                   fileName = cms.string('demo_out.root')
                                  )

process.demo = cms.EDAnalyzer('DemoAnalyzer',
#    tracks    = cms.untracked.InputTag('generalTracks'),
    tracks    = cms.untracked.InputTag('slimmedMuons'),
                              )

process.p = cms.Path(process.demo)
```

Input file format should contain contents with the InputTag

```
[zhangjq@lxlogin003 python]$root -l demo_out.root
root [0]
Attaching file demo_out.root as _file0...
(TFile *) 0x4191a30
root [1] .ls
TFile**         demo_out.root
 TFile*         demo_out.root
  KEY: TDirectoryFile   demo;1  demo
root [2] demo->cd()
(bool) true
root [3] .ls
TDirectoryFile*         demo    demo
 KEY: TH1F      h_pt;1  title of h_pt
 KEY: TTree     tree1;1 my tree 1
root [4] tree1->Print()
******************************************************************************
*Tree    :tree1     : my tree 1                                              *
*Entries :      100 : Total =           22991 bytes  File  Size =      13818 *
*        :          : Tree compression factor =   1.42                       *
******************************************************************************
*Br    0 :runNum    : runNum/i                                               *
*Entries :      100 : Total  Size=        963 bytes  File Size  =        101 *
*Baskets :        1 : Basket Size=      32000 bytes  Compression=   4.69     *
*.............................................................................*
*Br    1 :lumiNum   : lumiNum/i                                              *
*Entries :      100 : Total  Size=        968 bytes  File Size  =        101 *
*Baskets :        1 : Basket Size=      32000 bytes  Compression=   4.70     *
*.............................................................................*
*Br    2 :eventNum  : eventNum/i                                             *
*Entries :      100 : Total  Size=        973 bytes  File Size  =        476 *
*Baskets :        1 : Basket Size=      32000 bytes  Compression=   1.00     *
*.............................................................................*
*Br    3 :nTrack    : nTrack/I                                               *
*Entries :      100 : Total  Size=        963 bytes  File Size  =        192 *
*Baskets :        1 : Basket Size=      32000 bytes  Compression=   2.47     *
*.............................................................................*
*Br    4 :vec_pt    : vector<double>                                         *
*Entries :      100 : Total  Size=       4631 bytes  File Size  =       2225 *
```

Output root file

- **Exercise: add above codes to save a tree and a histogram and run it**

Example: /publicfs/cms/user/zhangjq/cms2025beihang/CMSSW_13_0_13/src/Demo/DemoAnalyzer/python/ConfFile_cfg.py

Event selection

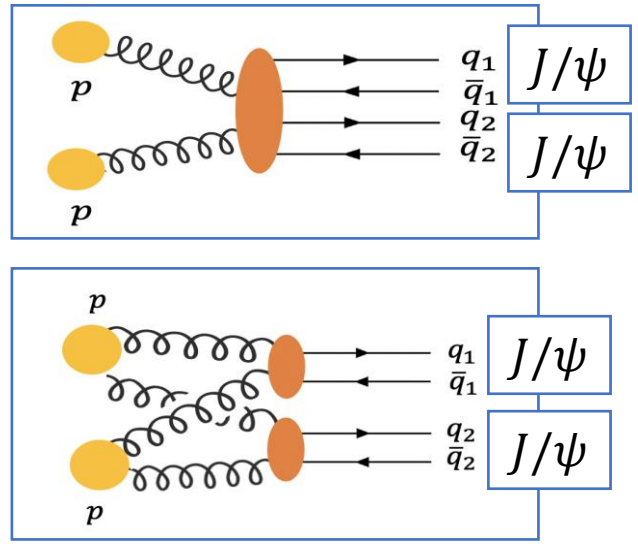
- To suppress bkg:
  - cuts in analyzer step and ntuple analysis step

$$pp \to X \to J/\psi J/\psi \to \mu^+\mu^-\mu^+\mu^-$$

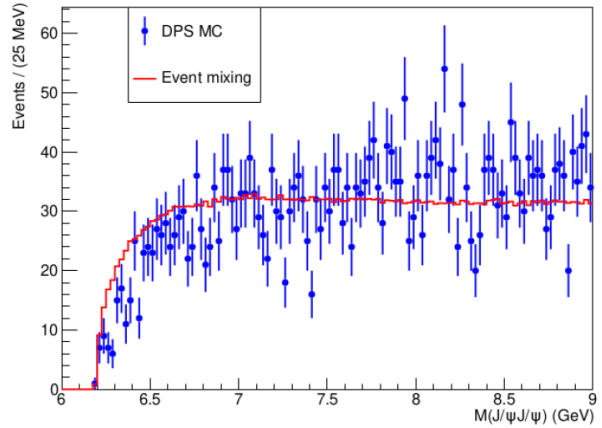
- Example cuts:
  - Detector acceptance/coverage: p_t and η of tracks
  - Tracks should be muon: MuonID
  - J/ψ candidate: mass window of muon pair; vertex fit of muon pair
  - $X \to J/\psi J/\psi \to \mu^+\mu^-\mu^+\mu^-$: vertex fit of two J/ψ and four muons

Background estimation

- Combinatorial background: non-$J/\psi J/\psi$ process
- Background from non-resonant process
  - Non-resonant single parton scattering, NRSPS
  - Double parton scattering, DPS

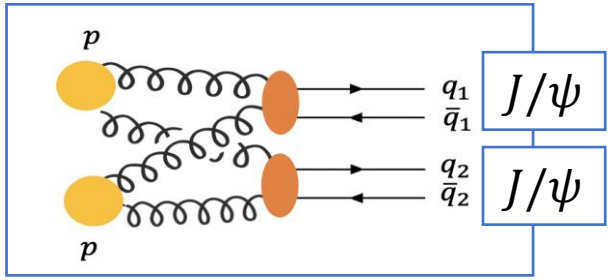


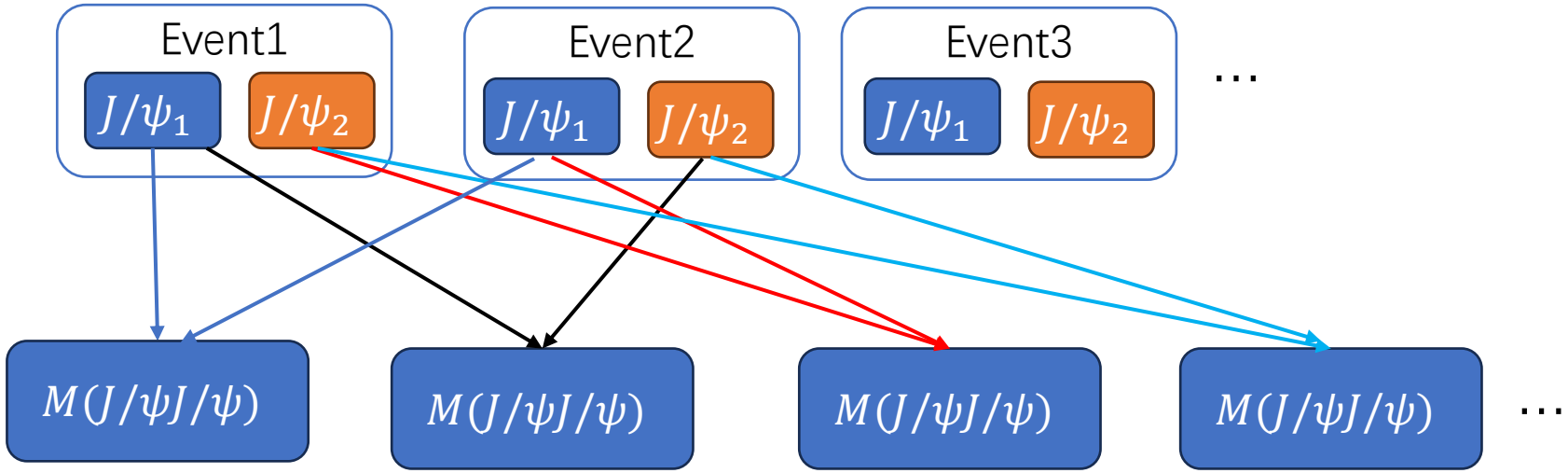
- MC simulation for NRSPS & DPS
- For DPS, another method is 'event mixing'

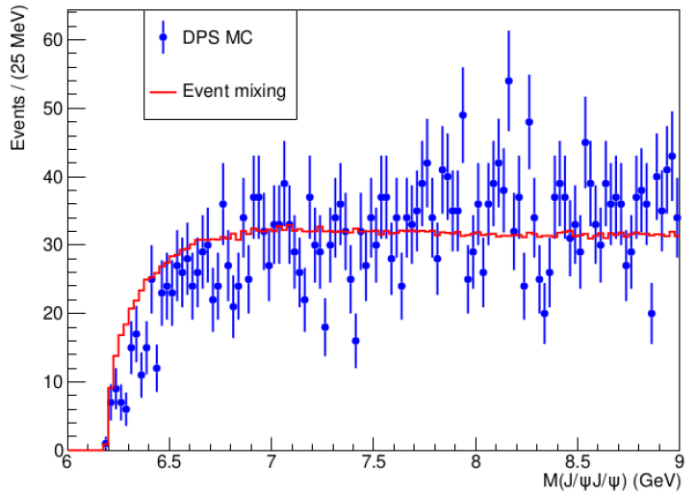


Event mixing for DPS shape

- DPS essentially contains two **independent** J/ψs
- Idea: artificially make events with independent J/ψs







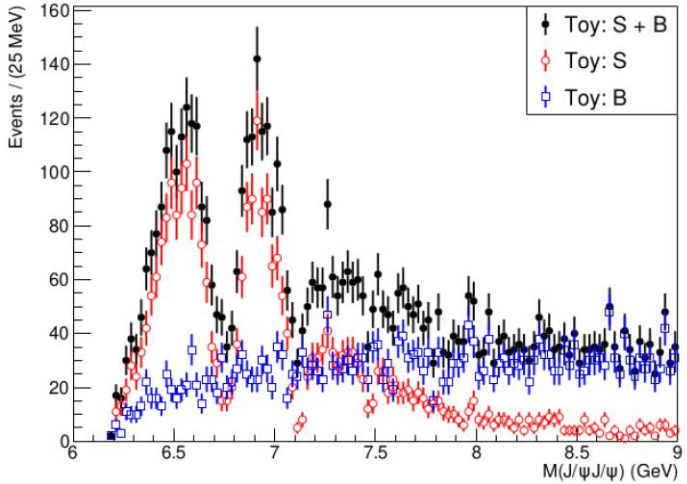
- **Exercise: get $M(J/\psi J/\psi)$ shape from event-mixing using MC in**

  /publicfs/cms/user/zhangjq/cms2025beihang/toy_dps/dps_mc.root

- **Copy mix_event.C in the same path, and add code in line 61-73**
- **Full example is mix_event_done.C in the same path**

Fit to mass distribution

- A toy $M(J/\psi J/\psi)$ distribution
  - Signal: three interfering relativistic BWs
  - Background: DPS



- Signal yields and parameters?
- Significance?

Fit to mass distribution

- DPS shape:

$$f_{NRDPS}(x) = \sqrt{x_t} \cdot \exp(-a \cdot x_t) \cdot (p_0 + p_1 \cdot x_t + p_2 \cdot x_t^2),$$
$$x_t = x - x_0, \quad x_0 = 2M_{J/\psi}.$$
$$a = 0.24358, p_0 = 0.23137, p_1 = -0.041952, p_2 = 0.012206$$

- Relativistic Breit-Wigner:
  - S-wave: $L = 0, B'_L = 1$

$$BW(m; m_0, \Gamma_0) = \frac{\sqrt{m\Gamma(m)}}{m_0^2 - m^2 - im\Gamma(m)},$$
$$\Gamma(m) = \Gamma_0 \left(\frac{q}{q_0}\right)^{2L+1} \frac{m_0}{m} \left(B'_L(q, q_0, d)\right)^2,$$

- Amplitude with interference:

$$f(m) = |r_1 \cdot \exp(i\phi_1) \cdot BW_1 + BW_2 + r_3 \cdot \exp(i\phi_3) \cdot BW_3|^2$$

- How to implement and use them in RooFit?

Fit mass distribution

- RooClassFactory::makePdf()
- Edit evaluate() function in MyDpsPdf.cxx

```
[zhangjq@lxslc704 test]$ root -l
root [0] RooClassFactory::makePdf("MyDpsPdf", "x,MTH,a,p0,p1,p2")
(bool) false
root [1] .ls
root [2] .q
[zhangjq@lxslc704 test]$ ls
MyDpsPdf.cxx  MyDpsPdf.h
[zhangjq@lxslc704 test]$
```

```
50  Double_t MyDpsPdf::evaluate() const
51  {
52    // ENTER EXPRESSION IN
53    double mx = x - MTH;
54    double fth = 0;
55    if (mx <= 0) return fth;
56    if (mx > 0) fth = TMath::Sqrt(mx);
57    double exponent = TMath::Exp(-1 * a * mx);
58    double poly = p0 + p1 * mx + p2 * mx * mx;
59    double res = fth * exponent * poly;
60    return res;
61  }
```

$$f_{NRDPS}(x) = \sqrt{x_t} \cdot \exp(-a \cdot x_t) \cdot (p_0 + p_1 \cdot x_t + p_2 \cdot x_t^2),$$
$$x_t = x - x_0, \quad x_0 = 2M_{J/\psi}.$$

- root –l –b –q “MyDpsPdf.cxx+”

```
[zhangjq@lxslc704 test]$ root -l -b -q "MyDpsPdf.cxx+"

Processing MyDpsPdf.cxx+...
Info in <TUnixSystem::ACLiC>: creating shared library /publicfs/cms/user/zhangjq/cms2024sysu/test/./MyDpsPdf_cxx.so
(MyDpsPdf) An instance of MyDpsPdf.
[zhangjq@lxslc704 test]$
```

- Then include .h, load .so in a root script, and use it as built-in pdf

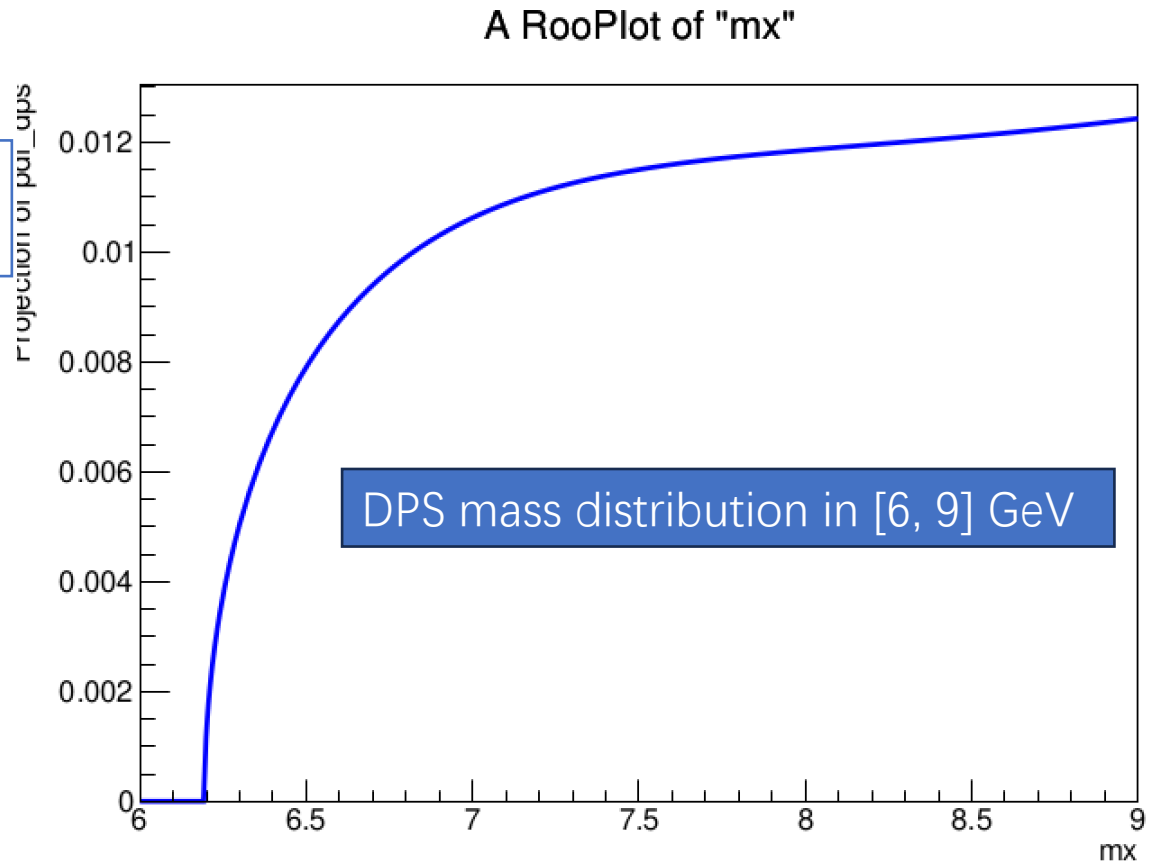
Note: here and after, we use the roofit version in CMSSW_11_1_1
So first: *cd some/path/CMSSW_11_1_1/src;* ***cmssw-el7****; cmsenv;*
Or: *cd some/path;* ***cmssw-el7;*** *cmsrel CMSSW_11_1_1/src; cmsenv;*

Fit mass distribution

- In root script, include .h and load _cxx.so
- Use MyDpsPdf as a built-in pdf

```
#include "RooRealVar.h"
#include "RooPlot.h"

#include "TSystem.h"
#include "MyDpsPdf.h"

using namespace RooFit;

void plotMyDpsPdf() {
  gSystem->Load("MyDpsPdf_cxx.so");

  RooRealVar mx("mx", "mx", 6.0, 9.0);
  RooRealVar R_MTH("R_MTH", "R_MTH", 2 * 3.0969);

  RooRealVar dpsA("dpsA", "dpsA", 0.24358);
  RooRealVar dpsP0("dpsP0", "dpsP0", 0.23137);
  RooRealVar dpsP1("dpsP1", "dpsP1", -0.041952);
  RooRealVar dpsP2("dpsP2", "dpsP2", 0.012206);
  MyDpsPdf pdf_dps("pdf_dps", "pdf_dps", mx, R_MTH, dpsA, dpsP0, dpsP1, dpsP2);

  RooPlot *frame = mx.frame();
  pdf_dps.plotOn(frame);

  TCanvas *c1 = new TCanvas("c1", "c1", 800, 600);
  c1->cd();
  frame->Draw();
  c1->Print("mydpspdf.pdf");

}
```

$a = 0.24358, p_0 = 0.23137,$
$p_1 = -0.041952, p_2 = 0.012206$



- **Exercise: wirte your own MyDpsPdf.cxx and use it to make a plot**
- **Example in the path:** /publicfs/cms/user/zhangjq/cms2025beihang/pdf_dps

Fit mass distribution

- Pdf for interfering BWs

```
root [0] RooClassFactory::makePdf("MyRelBW1BW2BW3Square", "x,mass1,width1,r1,phi1,mass2,width2,r2,phi2,mass3,width3,r3,phi3")
```

```
 9 #include "Riostream.h"
10
11 #include "MyRelBW1BW2BW3Square.h"
12 #include "RooAbsReal.h"
13 #include "RooAbsCategory.h"
14 #include <math.h>
15 #include "TMath.h"
16
17 #include "ComplexRelBWFcn.h"
18
```

```
73 Double_t MyRelBW1BW2BW3Square::evaluate() const
74 {
75   // ENTER EXPRESSION IN TERMS OF VARIABLE ARGUMENTS HERE
76   std::complex<double> bw1 = ComplexRelBW(x, mass1, width1, r1, phi1);
77   std::complex<double> bw2 = ComplexRelBW(x, mass2, width2, r2, phi2);
78   std::complex<double> bw3 = ComplexRelBW(x, mass3, width3, r3, phi3);
79   std::complex<double> sum = bw1 + bw2 + bw3;
80   double res = std::norm(sum);
81   return res;
82 }
```

- Auxiliary files

```
 6 #ifndef COMPLEX_RELBW_FCN_H
 7 #define COMPLEX_RELBW_FCN_H
 8
 9 #include <iostream>
10 #include <complex>
11 #include "TMath.h"
12
13 void ComplexRelBWFcn();
14 double Q2(double sa, double sb, double sc);
15 double BlattWeisskopf(double q2, double d, double L);
16 std::complex<double> ComplexRelBW(double x, double mass, double width, double r, double phi);
17 #endif
```

ComplexRelBWFcn.h

```
 6 #include "ComplexRelBWFcn.h"
 7 #include "TMath.h"
 8
 9 void ComplexRelBWFcn() {
10   return;
11 }
12 double Q2(double sa, double sb, double sc) {
13   double res = (sa + sb - sc) * (sa + sb - sc) / 4.0 / sa - sb;
14   if (res <= 0) {
15     res = 0;
16   }
17   return res;
18 }
```

ComplexRelBWFcn.cxx

- First: root –l –b –q “ComplexRelBWFcn.cxx+”
- Then: root –l –b –q “MyRelBW1BW2BW3Square.cxx+”

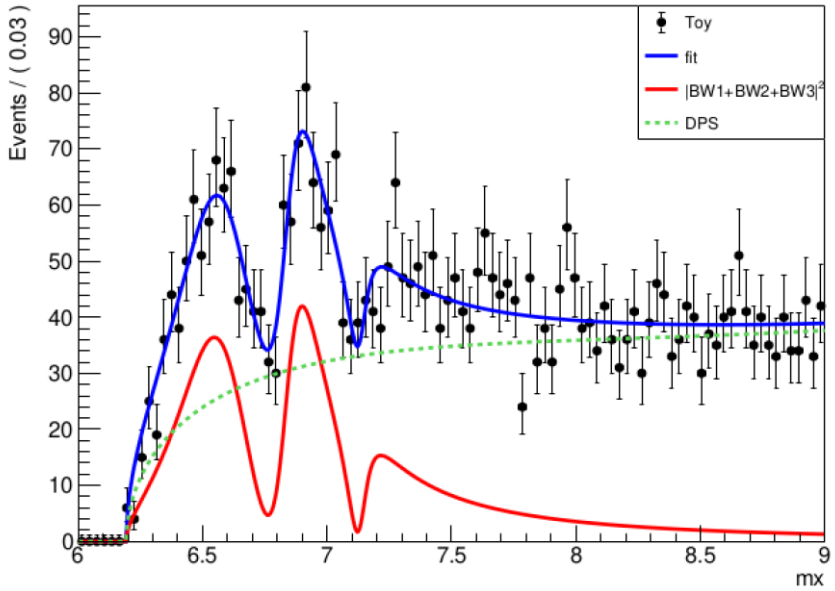
Example at: /publicfs/cms/user/zhangjq/cms2025beihang/fit_test

Fit mass distribution

- Fit to the toy

```
93   MyRelBW1BW2BW3Square pdf_bw123("pdf_bw123", "pdf_bw123", mx,
94       mass_bw1, width_bw1, r_bw1, phi_bw1,
95       mass_bw2, width_bw2, r_bw2, phi_bw2,
96       mass_bw3, width_bw3, r_bw3, phi_bw3);
```

```
113   RooFitResult *fit_res_model = model.fitTo(*data, Save());
114   double nll_model = fit_res_model->minNll();
```



- Significance:
  1. Fit using **hypothesis 0** model, get negative log likelihood, nll0
  2. Fit using **hypothesis 1** model, get nll1
  3. $2\Delta nll$ follows χ^2 distribution
  4.
```
120   double pval = TMath::Prob(2 * delta_nll, 1); //assuming ndof = 1
121   double significance = RooStats::PValueToSignificance(pval / 2.0);
```

- **Question: significance of what, or between which two models?**

Example at: /publicfs/cms/user/zhangjq/cms2024sysu/fit_test/fitToy.C

Fit using condor jobs

- When you need do many independent fits
  - Random sampling in parameter space to find global minimum
  - Closure test
  - Toy-experiment to evaluate significance
- Fit using condor jobs
  - A simple example: gen toys and fit them to do eg. closure test

```
/publicfs/cms/user/zhangjq/cms2025beihang/fit_with_condor
[zhangjq@lxlogin003 fit_with_condor]$ls
condor_err  condor_out  fit_out           run.jdl     scripts
condor_log  fit_fig     make_toy_list.sh  run_job.sh  toy_list_1_7.txt
[zhangjq@lxlogin003 fit_with_condor]$ls scripts/
fit_job.sh  gen_and_fit.C
[zhangjq@lxlogin003 fit_with_condor]$
```

condor_xx for condor jobs output & logs

To submit jobs:
condor_submit run.jdl

Example: /publicfs/cms/user/zhangjq/cms2025beihang/fit_with_condor

Fit using condor jobs

- Condor job description file: run.jdl

```
universe = vanilla
executable = run_job.sh
arguments = $(beg_toy) $(n_toy) $(beg_fit) $(n_fit)

output = condor_out/$(ClusterId).$(ProcId).out
error = condor_err/$(ClusterId).$(ProcId).err
log = condor_log/$(ClusterId).$(ProcId).log

####set max run time by one of the two: +JobFlavor
+JobFlavour = "microcentury"
# +MaRunTime = number of seconds
#+MaxRunTime = 10800

should_transfer_files = yes
transfer_input_files = scripts
#transfer the output root files in the dir job_output to the local job_output dir
#transfer_output_files = job_output
#when_to_transfer_output = ON_EXIT

queue beg_toy, n_toy, beg_fit, n_fit from toy_list_1_7.txt
```

run_job.sh:
executed by condor server.
Will call fit_job.sh

```
#!/bin/bash
cd scripts
#pass the args to the script/command which do the real job
./fit_job.sh $1 $2 $3 $4
```

condor_xx for condor jobs output & logs

Scripts do real jobs are in directory “scripts”, and will be transferred to condor server before running

```
[zhangjq@lxlogin003 fit_with_condor]$ls scripts/
fit_job.sh  gen_and_fit.C
```

Each job has its own argument values, which are listed in a text file

```
[zhangjq@lxlogin003 fit_with_condor]$cat toy_list_1_7.txt
1, 3, 1, 3
3, 5, 1, 3
5, 7, 1, 3
```

Example: /publicfs/cms/user/zhangjq/cms2025beihang/fit_with_condor

Fit using condor jobs

- When job runs in condor server,
  - run_job.sh passes arguments to fit_job.sh and run fit_job.sh
  - fit_jobs.sh edit settings (e.g., random seed) of gen_and_fit.C and then run gen_and_fit.C
  - gen_and_fit.C does real works, and save result directly to disk (if path not accessible by condor job, transfer it back using transfer_out in run.jdl)
- Text file for argument values can be gen by a script and can contain many args for each job
  - see make_toy_list.sh

```
#!/bin/bash
BEG_TOY_IN_JOB=${1}
N_TOY_IN_JOB=${2}
BEG_FIT_IN_TOY=${3}
N_FIT_IN_TOY=${4}

sed -i "s/BEG_TOY = 1/BEG_TOY = ${BEG_TOY_IN_JOB}/g" gen_and_fit.C
sed -i "s/N_TOY = 1/N_TOY = ${N_TOY_IN_JOB}/g" gen_and_fit.C
sed -i "s/BEG_FIT = 1/BEG_FIT = ${BEG_FIT_IN_TOY}/g" gen_and_fit.C
sed -i "s/N_FIT = 1/N_FIT = ${N_FIT_IN_TOY}/g" gen_and_fit.C
```

fit_job.sh

```
  int BEG_TOY = 1;
  int N_TOY = 1;
  int END_TOY = BEG_TOY + N_TOY;
  int BEG_FIT = 1;
  int N_FIT = 1;
  int END_FIT = BEG_FIT + N_FIT;

  double m_min = 1.60, m_max = 2.2;
  int m_bins = 60; //
  TString YTitle("Events / 10 MeV");
```

gen_and_fit.C

Example: /publicfs/cms/user/zhangjq/cms2025beihang/fit_with_condor/

- HTCondor Manual, and a good slides (in same directory): TrainingSlides.pdf