Studies of Beam and Cosmic events at ProtoDUNE with PandoraPFA

Pantelis Melas, Niki Saoulidou

17/10/2019

National and Kapodistrian
University of Athens

Outline

- Preliminary results on analyzing beam and cosmic events:
  - Selection and analysis of "good" beam events (PandoraPFA tracks and showers)
  - Comparisons of characteristics beam with cosmic particles.
- Summary and on going work

Summary of Event Information

For the run 5387 which has 126.900 events

(samweb.fnal.gov:8480/sam/dune/api/definitions/name/runset-5387-reco-unified-hv-180kV-beam-1GeV-v0/files/list)

(I took only 123.377 events (97,2% or 918 files) of the run)

I found **27.401 (22%) events** passed the beam quality check (**match=1**) and had reconstructed PFparticles associated with the beam. (**BEAM_EVENT=1**),

I found **9.030 (7%) events** which had reconstructed PFparticles associated with beam in the detector (**BEAM_EVENT=1**),but they failed to pass the beam quality check. (**match =0**)

I found **12.056 (10%) events** which had beam information (**match=1)** ,but they didn't have reconstructed PFparticles (tracks or showers) in the detector. (**BEAM_EVENT=0**)

I found **74.890 (61%) events** which didn't have reconstructed PFparticles associated with the the beam (**BEAM_EVENT=0**), and they failed to pass the quality check. (**match =0**)

Beam related Information

Code for Obtaining Beam ID

/dune/app/users/pmelas/larsoft_mydev/srcs/dunetpc/dune/Protodune/Analysis/MyProtoDUNETestAnalyzer_module.cc

```
  //Lets find beam info


      art::Handle< std::vector<beam::ProtoDUNEBeamEvent> > beamHandle;
      std::vector<art::Ptr<beam::ProtoDUNEBeamEvent>> beaminfo;
    if(e.getByLabel(fBeamModuleLabel,beamHandle)){
      art::fill_ptr_vector(beaminfo,beamHandle);
      }

    for(unsigned int i = 0; i < beaminfo.size(); ++i){

  std::map<unsigned int, std::vector<const recob::PFParticle*>> sliceMap;
          fbeamtrigger = beaminfo[i]->GetTimingTrigger();

  if(!beaminfo[i]->CheckIsMatched()) {
           std::cout << "Failed quality check" << std::endl;
           match = 0;  // IT MEANS THAT I DIDNT TAKE THE BEAM INFO
           }

  if(beaminfo[i]->CheckIsMatched()) {
           std::cout << "Successed quality check" << std::endl;
           match = 1;  // IT MEANS THAT I TOOK THE BEAM INFO

     //Access PID

    const beam::ProtoDUNEBeamEvent & beamEvent = *(beaminfo.at(0)); //Should just have one
    std::vector< int > pids = beamlineUtil.GetPID( beamEvent, 1. );

    for( size_t i = 0; i < pids.size(); ++i ){
       std::cout << pids[i] << std::endl;
       beamID.push_back(pids[i]);
       }

std::cout << "electron " << beamlineUtil.GetPIDCandidates( beamEvent, 1. ).electron << std::endl;
std::cout << "muon "     << beamlineUtil.GetPIDCandidates( beamEvent, 1. ).muon     << std::endl;
std::cout << "pion "     << beamlineUtil.GetPIDCandidates( beamEvent, 1. ).pion     << std::endl;
std::cout << "kaon "     << beamlineUtil.GetPIDCandidates( beamEvent, 1. ).kaon     << std::endl;
std::cout << "proton "   << beamlineUtil.GetPIDCandidates( beamEvent, 1. ).proton   << std::endl;
std::cout << "proton "   << beamlineUtil.GetPIDCandidates( beamEvent, 1. ).deuteron   << std::endl;

std::cout << std::endl;
//////////////////////////////////////////////////////////
```

Selection of “quality” beam events

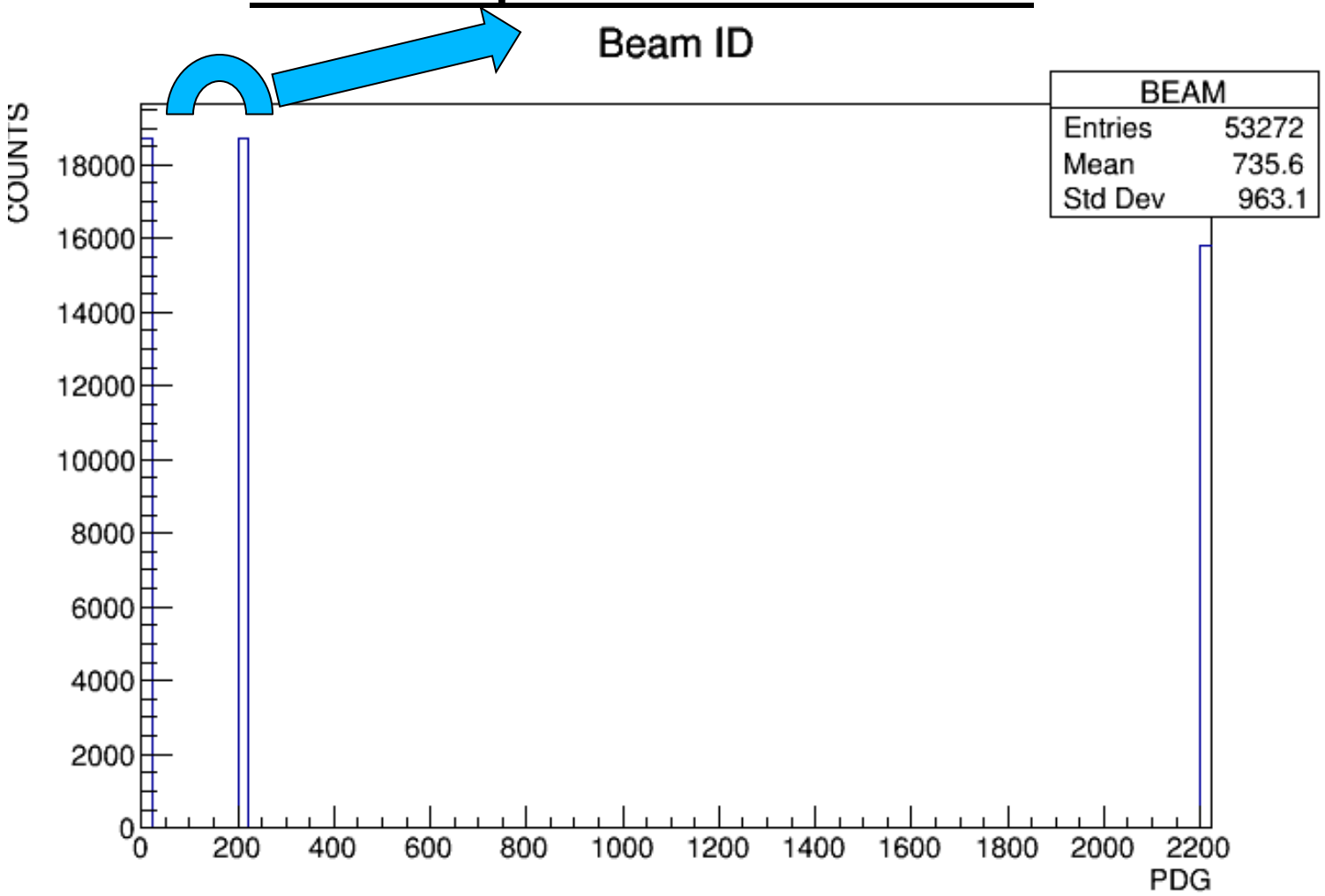
- In order to access beamline information, we use the **beam::ProtoDUNEBeamEvent class**. This is filled by BeamEvent_module.cc (@ dunetpc/dune/Protodune/singlephase), which is included in RunRawDecoder.fcl.

- **One should select for events which were successfully matched to the database information.** Information is generally saved to the database on a per-spill basis, and it is up to the module to match between the timestamp of the event coming from the timing system to the General Triggers coming within a spill. **In the case that the art event was not matched to the beamline data, the TOF, Profiler, and Cerenkov info is not saved to the event. Events with useful information can be selected for using :**

  CheckIsMatched()

BEAM ID

This corresponds to same events !!!!

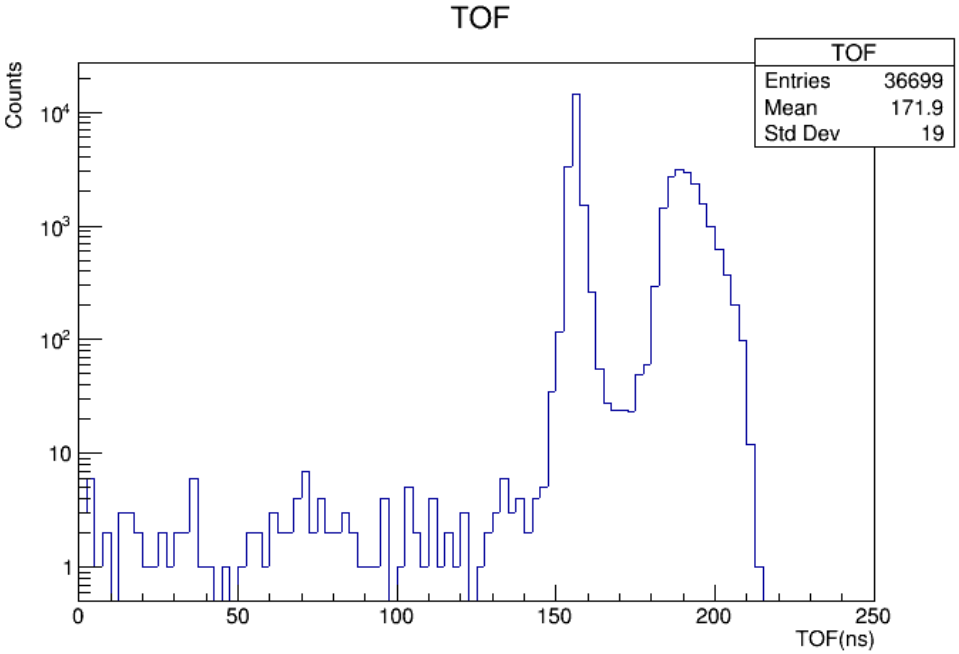


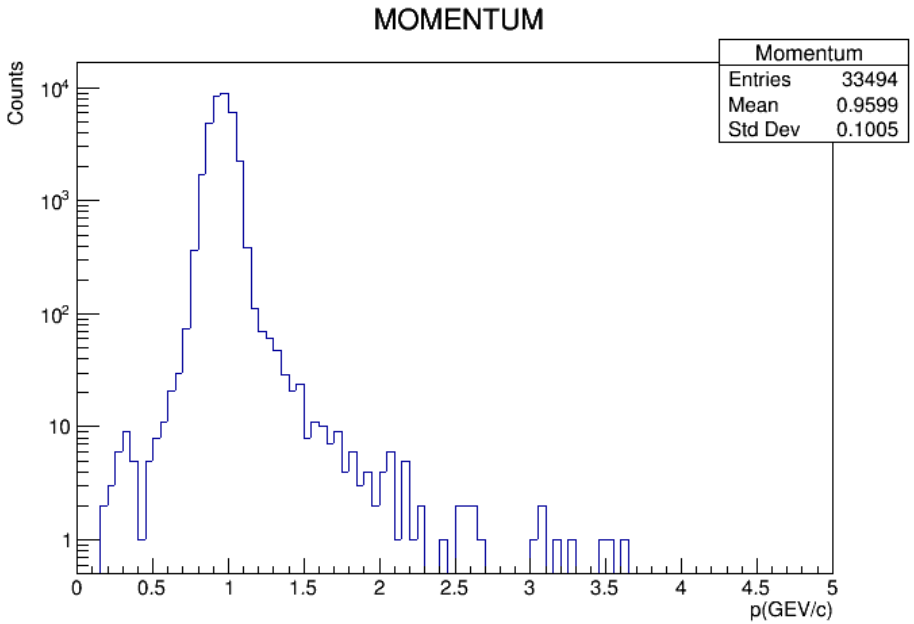
From the **39.457 events with "match==1"**, **18.726 events** are pions OR muons and **15.820 events** are protons. The rest, **4.911 events (12,4%)** didn't have the beam ID information (see Back Up for example of such event)

Code for Obtaining Beam Momentum and TOF Info

```
    // Beam momentum

  auto & beammom = beaminfo[i]->GetRecoBeamMomenta();
if(beaminfo[i]->GetTOFChan() >= 0)
  ftof =  beaminfo[i]->GetTOF();


    if(!beammom.empty()){
      fbeamtrackMomentum = beammom[0];
      ftof_expElec =  beamlineUtil.ComputeTOF(11,         beammom[0]);
      ftof_expMuon =  beamlineUtil.ComputeTOF(13,         beammom[0]);
      ftof_expPion =  beamlineUtil.ComputeTOF(211,        beammom[0]);
      ftof_expKaon =  beamlineUtil.ComputeTOF(321,        beammom[0]);
     ftof_expProt =  beamlineUtil.ComputeTOF(2212,       beammom[0]);
      ftof_expDeut =  beamlineUtil.ComputeTOF(1000010020, beammom[0]);
      }

      std::cout<<" beam momentum  "<<  fbeamtrackMomentum << std::endl;
      std::cout<<" ftof  "<<  ftof << std::endl;
      std::cout<<" ftof_expElec  "<<  ftof_expElec << std::endl;
      std::cout<<" ftof_expMuon  "<<  ftof_expMuon << std::endl;
      std::cout<<" ftof_expPion  "<<  ftof_expPion << std::endl;
      std::cout<<" ftof_expKaon  "<<  ftof_expKaon << std::endl;
      std::cout<<" ftof_expProt  "<<  ftof_expProt << std::endl;
      std::cout<<" ftof_expDeut  "<<  ftof_expDeut << std::endl;


    } // FOR MATCH=1
  }    // beaminfo.size loop
```

TOF AND BEAM MOMENTUM





- **Some particles don't have TOF info**

- There are 2 clear peaks in the TOF distribution none of which matches to the information printed by the BeamLineUtil (see previous slide for code) : is there a known reason for this?

- **Some particles don't have momentum info**

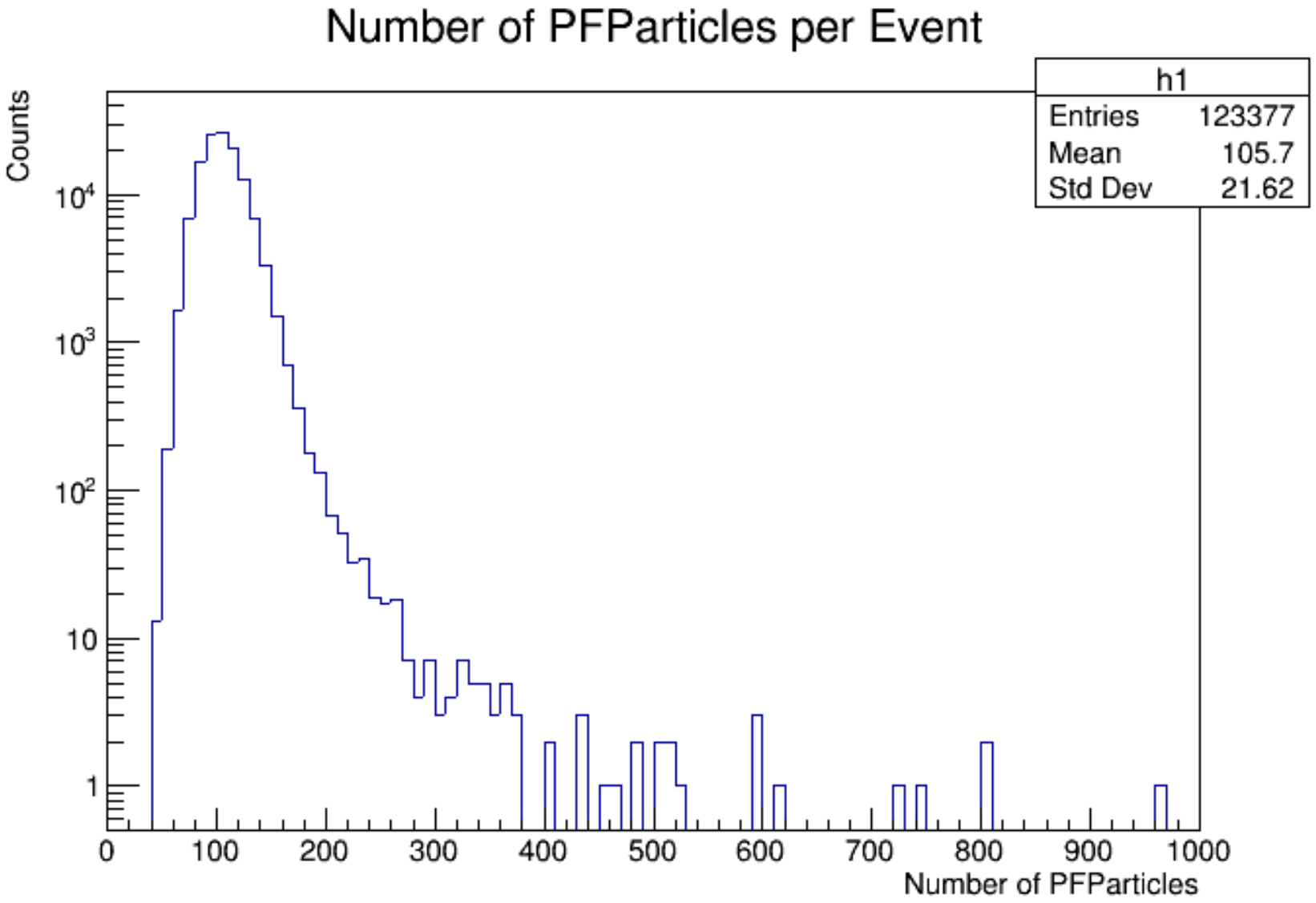
```
ftof  196.805
ftof_expElec  95.316
ftof_expMuon  95.9468
ftof_expPion  96.414
ftof_expKaon  108.255
ftof_expProt  136.38
ftof_expDeut  230.007
```

PFParticle Distribution

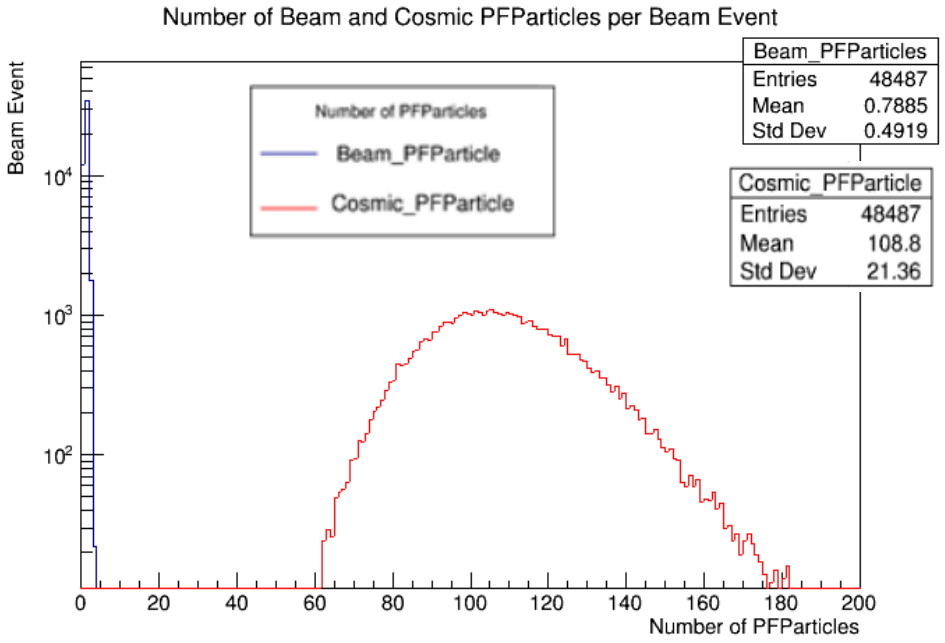
Code for Finding Beam PFparticles

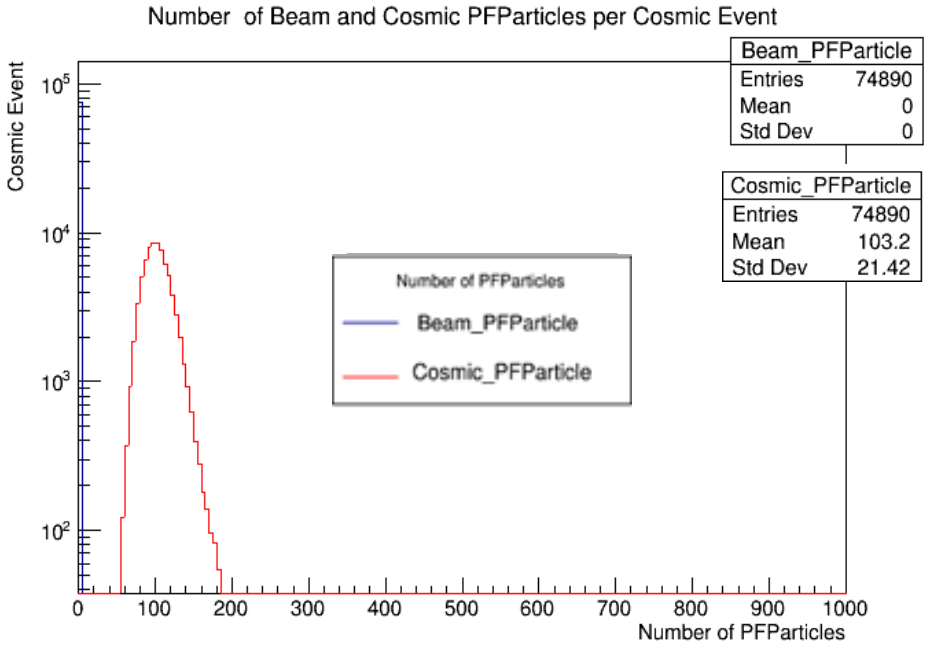
```
auto recoParticles = e.getValidHandle<std::vector<recob::PFParticle>>(fPFParticleTag);
if(recoParticles.isValid()) {

const art::FindManyP<recob::Track> findTracks(recoParticles,e,fTrackerTag);
const art::FindManyP<recob::Shower> findShowers(recoParticles,e,fShowerTag);

for(unsigned int p = 0; p < recoParticles->size(); ++p) {
const recob::PFParticle particle = recoParticles->at(p);
   if(particle.IsPrimary()) {
// Only consider primaries so we don't include deltas etc

//const recob::Track* thisTrack   = pfpUtil.GetPFParticleTrack(*particle, e,fPFParticleTag,fTrackerTag);
//const recob::Shower* thisShower = pfpUtil.GetPFParticleShower(*particle,e,fPFParticleTag,fShowerTag);
 const std::vector<art::Ptr<recob::Track>> pfpTrack = findTracks.at(p);
 const std::vector<art::Ptr<recob::Shower>> pfpShower = findShowers.at(p);


        BEAMSLICE = 0;
        IDofEverything = -999;
        AngleOfEverything = -999.0;
        LengthOfEverything = -999.0;
        PhiOfEverything    = -999.0;
        ShowerIDofEverything = -999;
        ShowerLengthOfEverything = -999.0;
        ShowerOpenAngleOfEverything = -999.0;

        unsigned short beamParticle = pfpUtil.IsBeamParticle(particle,e,fPFParticleTag);
      if(beamParticle != 0){
        std::cout << "YEAH,WE FOUND BEAM PARTICLES" << std::endl;
        BEAMSLICE = 1;
        has_beam_particle = 1;
        }// NOW BEAMSLICE=1

       VECTOR BEAMSLICE.push back(BEAMSLICE);
```

Number of PFParticles per Event



Number of Beam and Cosmic PFParticles per Event





- We define as Beam Events : BEAM_EVENT = 1 && match = 1
  BEAM_EVENT = 0 && match = 1
  BEAM_EVENT = 1 && match = 0

- We define as Cosmic Events : BEAM_EVENT = 0 && match = 0

The number of Beam PFParticles takes the value zero for the Beam Events histogram due to BEAM_EVENT = 0 && match = 1

TRACKS

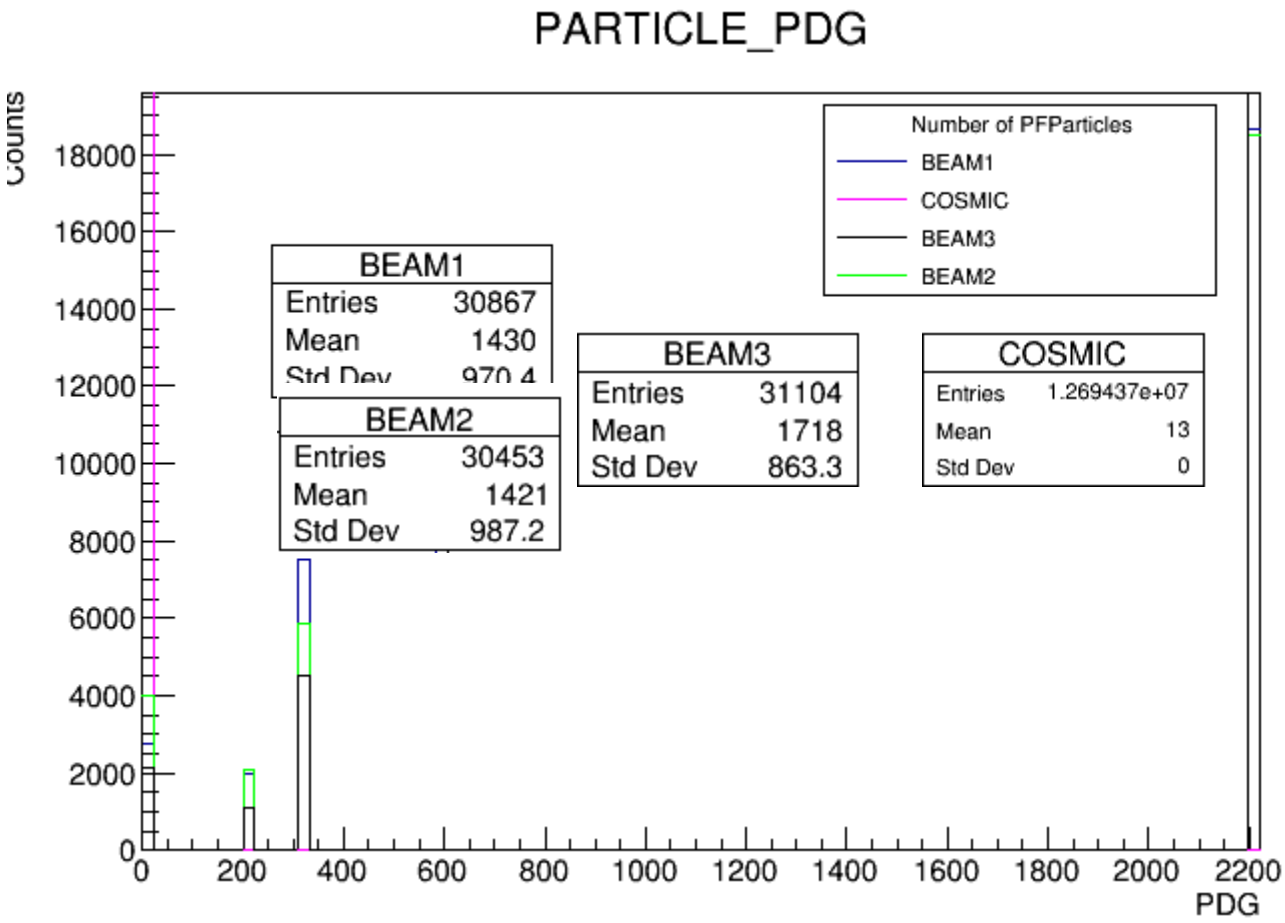
Code for Finding Tracks

```
if(pfpTrack.size()>0){

  art::Ptr<recob::Track> ThisTracks = pfpTrack.at(0);
  IDofEverything = particle.PdgCode();
  LengthOfEverything = ThisTracks->Length();
  AngleOfEverything = ThisTracks->Theta()*180/3.14;
  PhiOfEverything = ThisTracks->Phi()*180/3.14;

  EVERYSHOWER_ID.push_back(ShowerIDofEverything);
  EVERYSHOWER_LENGTH.push_back(ShowerLengthOfEverything);
  EVERYSHOWER_OPENANGLE.push_back(ShowerOpenAngleOfEverything);
  EVERYTRACK_LENGTH.push_back(LengthOfEverything);
  EVERYTRACK_ID.push_back(IDofEverything);
  EVERYTRACK_ANGLE.push_back(AngleOfEverything);
  EVERYTRACK_PHI.push_back(PhiOfEverything);

  } // is pfpTrack size >0 ?
```

Testing PID from anab::ID

```
        std::vector<const recob::PFParticle*> beamSlicePrimaries = pfpUtil.GetPFParticlesFromBeamSlice(e,fPFParticleTag);
     // std::cout << " We found  " << beamSlicePrimaries.size() << " beam particles in slice " << beamSlice << std::endl;
    // Loop over the particles in the beam slice
 for (const recob::PFParticle* particle : beamSlicePrimaries){
    const recob::Track* thisTrack    = pfpUtil.GetPFParticleTrack(*particle, e,fPFParticleTag,fTrackerTag);

  if(thisTrack != 0x0){

    // PID
    std::vector<anab::ParticleID> pids = trackUtil.GetRecoTrackPID(*thisTrack, e, fTrackerTag, fParticleIDTag);
    if(pids.size() != 3 && fVerbose > 0)
      std::cerr << "WARNING::PID vector size for primary is = " << pids.size() << std::endl;

    for(size_t k = 0; k < pids.size() && k<3; k++){
      int plane = pids[k].PlaneID().Plane;
      if(plane < 0) continue;
      if(plane > 2) continue;

for(int k=0; k < 3; k++){

   fprimaryPID_Pdg[k] = -999;
   fprimaryPID_Ndf[k] = -999;
   fprimaryPID_MinChi2[k] = -999.0;
   fprimaryPID_DeltaChi2[k] = -999.0;
   fprimaryPID_Chi2Proton[k] = -999.0;
   fprimaryPID_Chi2Kaon[k] = -999.0;
   fprimaryPID_Chi2Pion[k] = -999.0;
   fprimaryPID_Chi2Muon[k] = -999.0;
   fprimaryPID_MissingE[k] = -999.0;
   fprimaryPID_MissingEavg[k] = -999.0;
   fprimaryPID_PIDA[k] = -999.0;
   }

     fprimaryPID_Pdg[plane]             = pids[plane].Pdg();
     fprimaryPID_Ndf[plane]             = pids[plane].Ndf();
     fprimaryPID_MinChi2[plane]         = pids[plane].MinChi2();
     fprimaryPID_DeltaChi2[plane]       = pids[plane].DeltaChi2();
     fprimaryPID_Chi2Proton[plane]      = pids[plane].Chi2Proton();
     fprimaryPID_Chi2Kaon[plane]        = pids[plane].Chi2Kaon();
     fprimaryPID_Chi2Pion[plane]        = pids[plane].Chi2Pion();
     fprimaryPID_Chi2Muon[plane]        = pids[plane].Chi2Muon();
     fprimaryPID_MissingE[plane]        = pids[plane].MissingE();
     fprimaryPID_MissingEavg[plane]     = pids[plane].MissingEavg();
     fprimaryPID_PIDA[plane]            = pids[plane].PIDA();
```

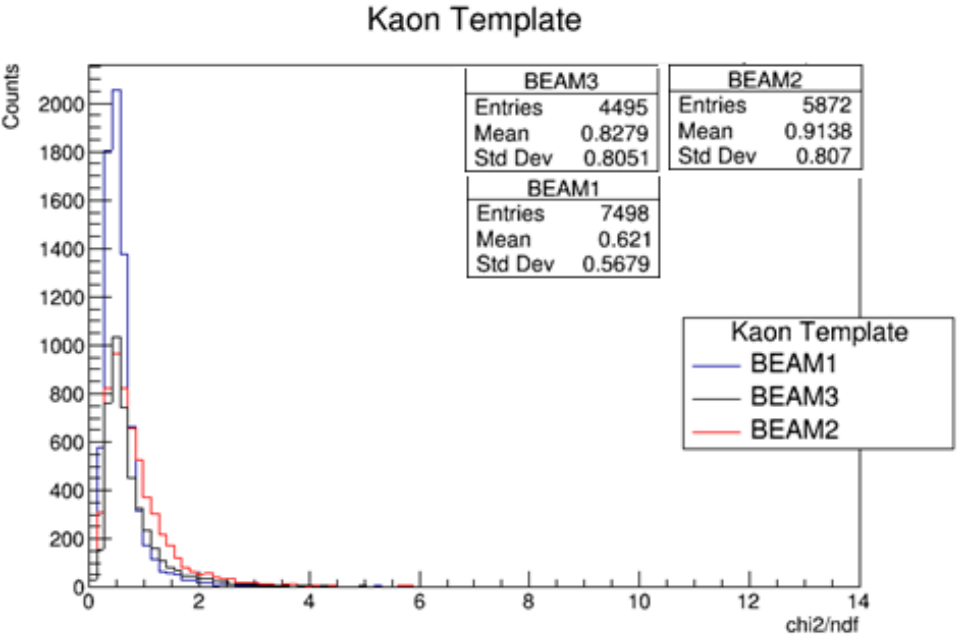
TRACK ID

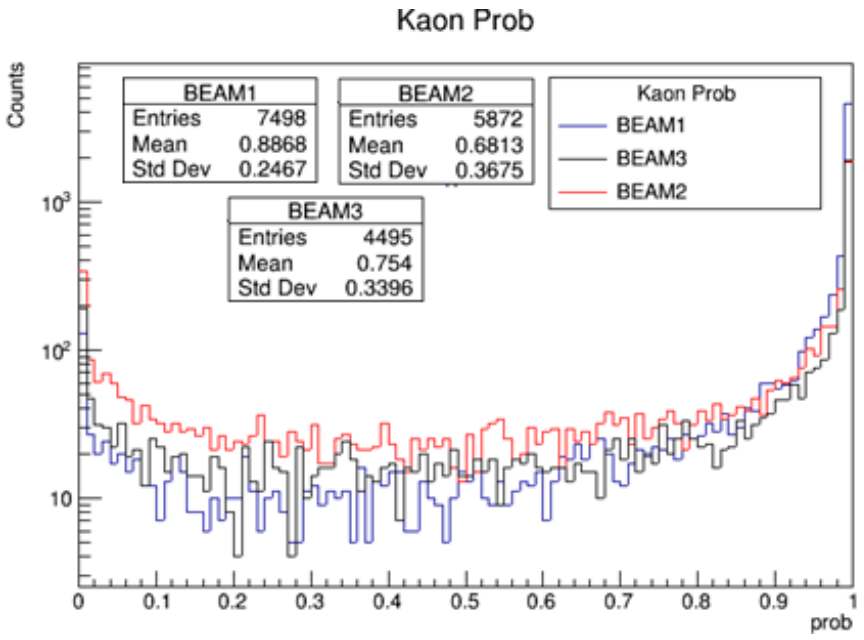


• What does the PID size (which is 3 or sometimes 2) ,that we use to obtain this info (see previous slide) ,corresponds to?

• PDG for cosmic events is always 13 as expected.

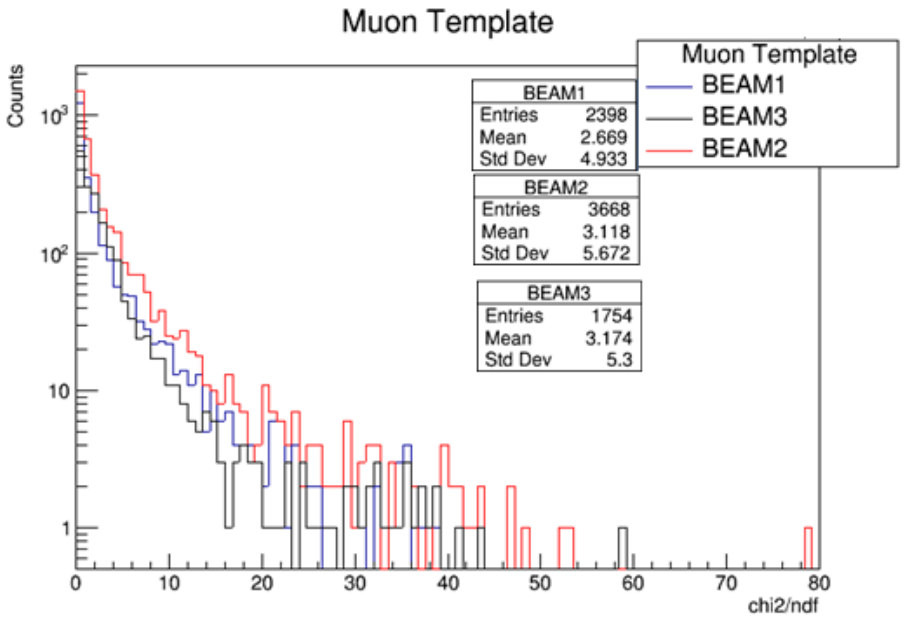
Track probabilities for Kaons (PDG ID ==321)

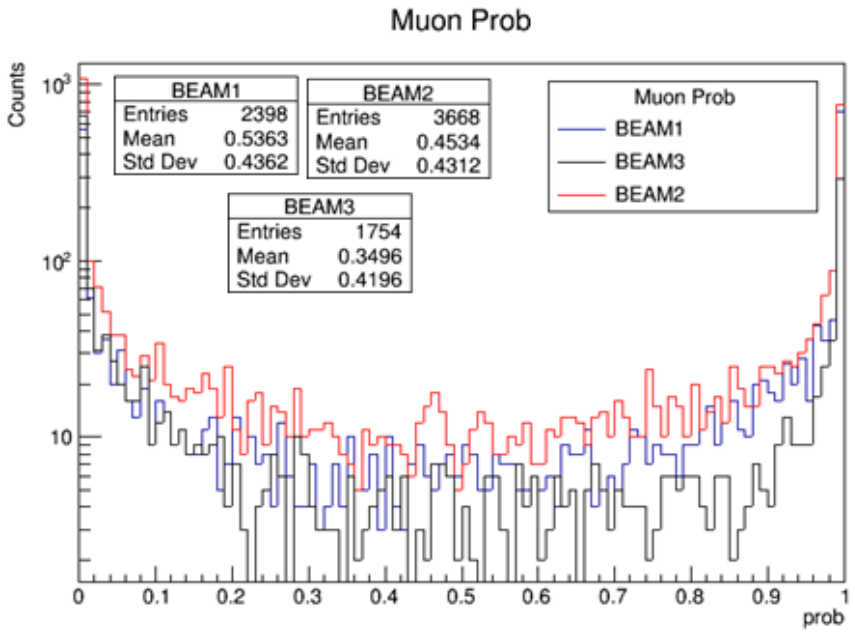




- The probability of the template fit for Kaons is mostly close to one, but there are also tracks with probability very close or at zero. Is this expected?

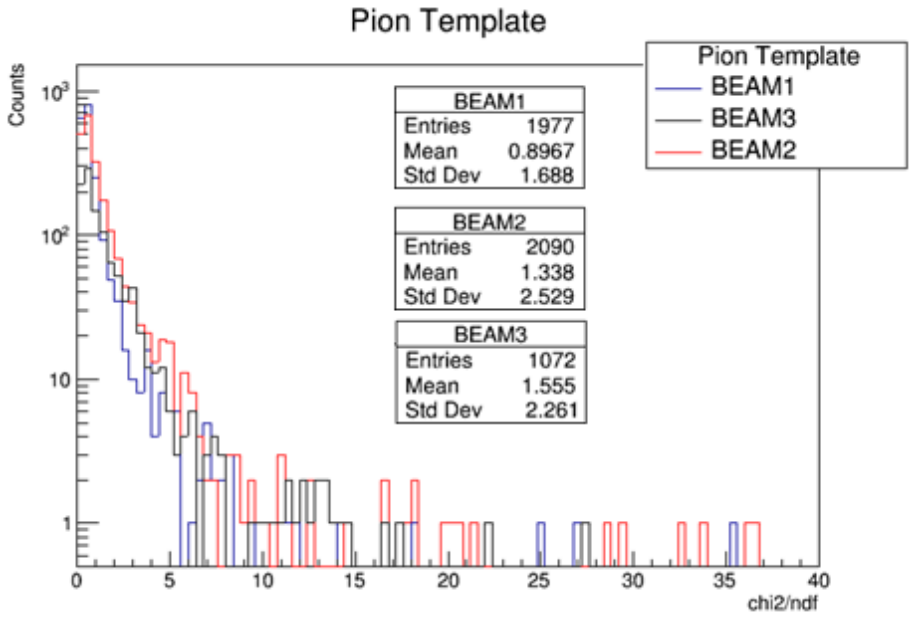
Track probabilities for Muons (PDG ID ==13)

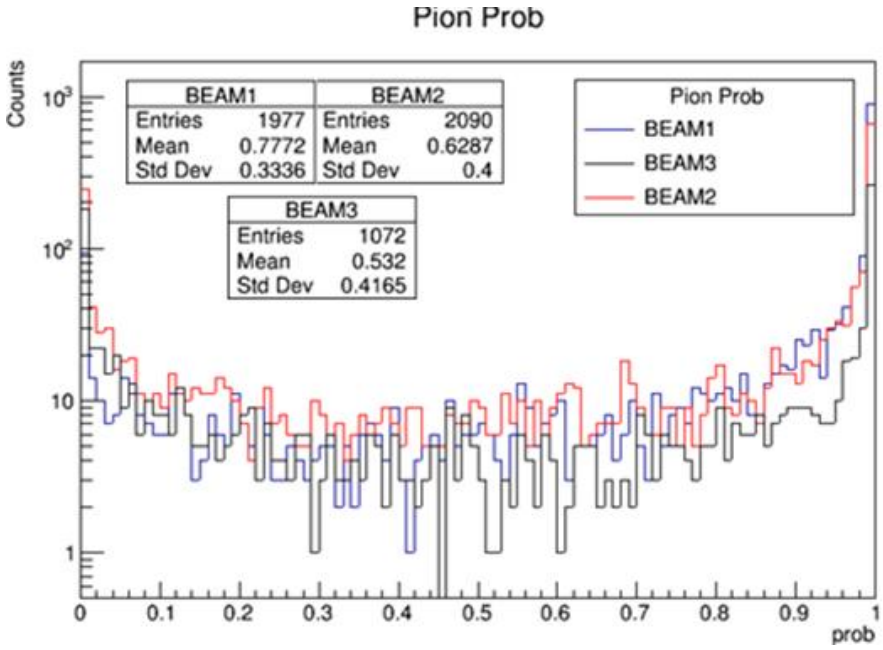




• The probability of the template fit for Muons is close to one for almost half of the tracks with PDG ID ==13 . Is this expected?

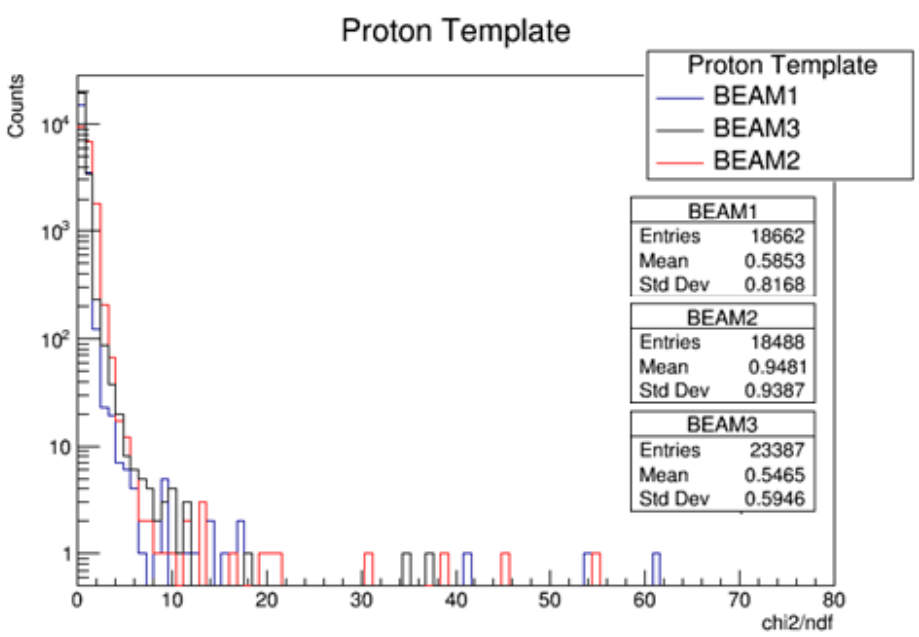
Track probabilities for Pions (PDG ID ==211)

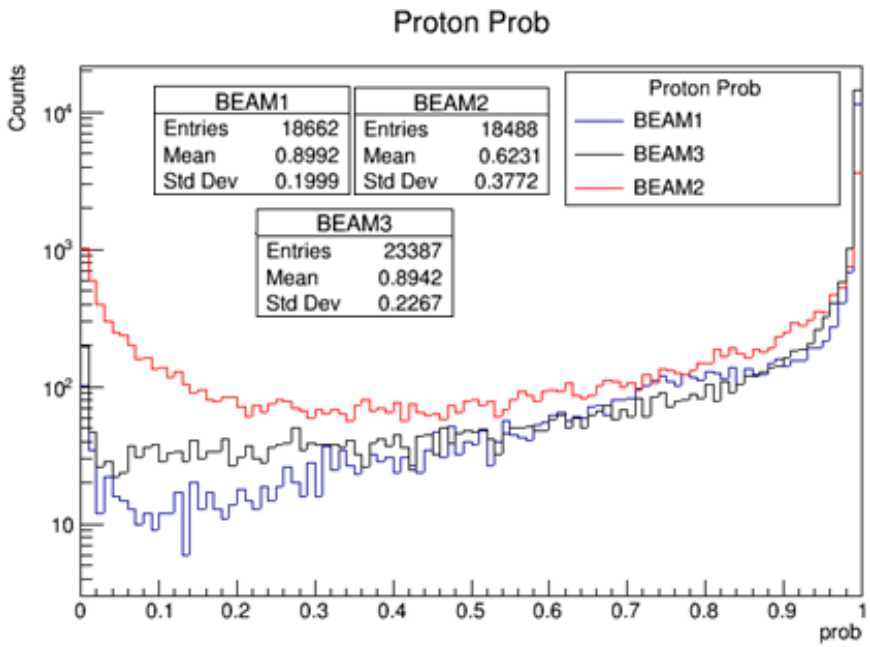




• The probability of the template fit for Pions is mostly close to one, but there are also decent number of tracks with probability very close or at zero. Is this expected ?

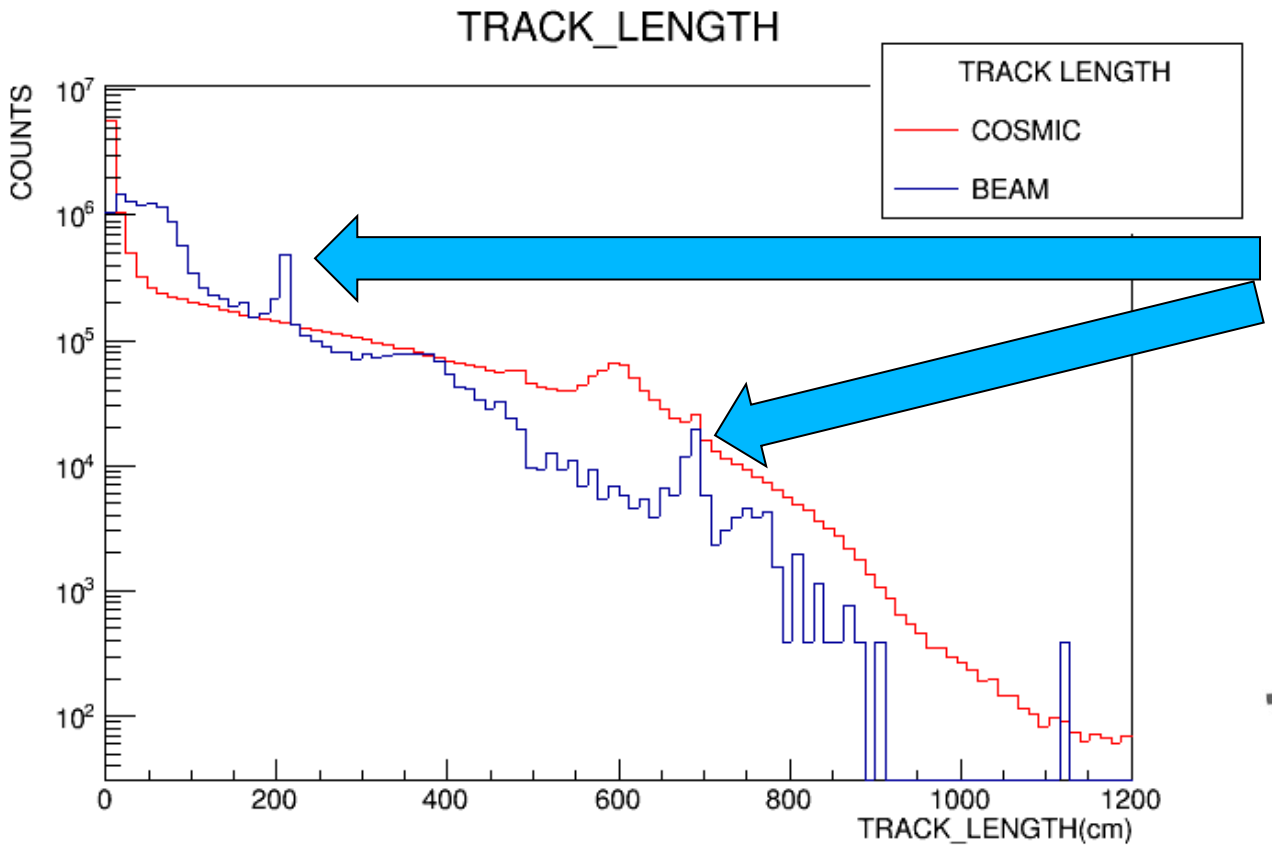
Track probabilities for Protons (PDG ID ==2212)





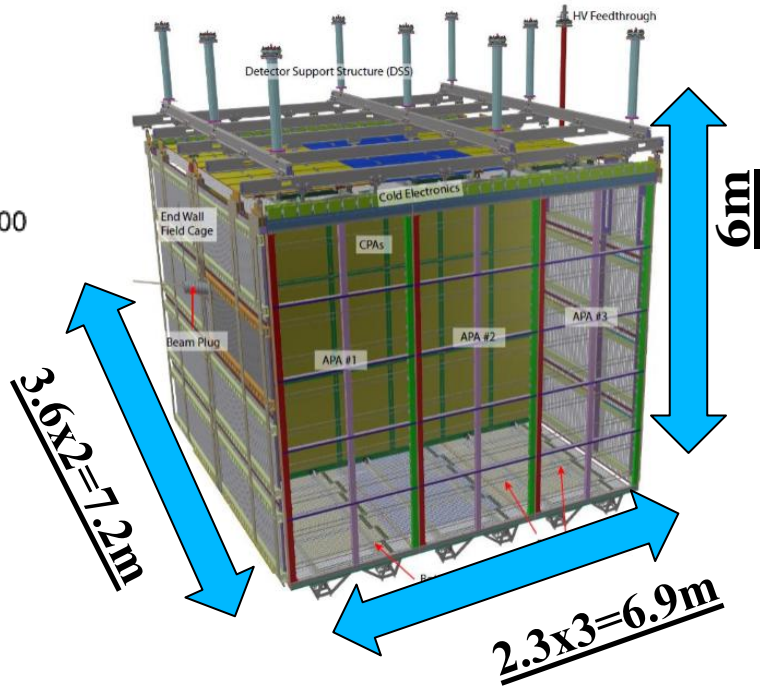
• The probability of the template fit for Protons is close to one but there are also tracks with probability very close or at zero. Is this expected?

TRACK LENGTH

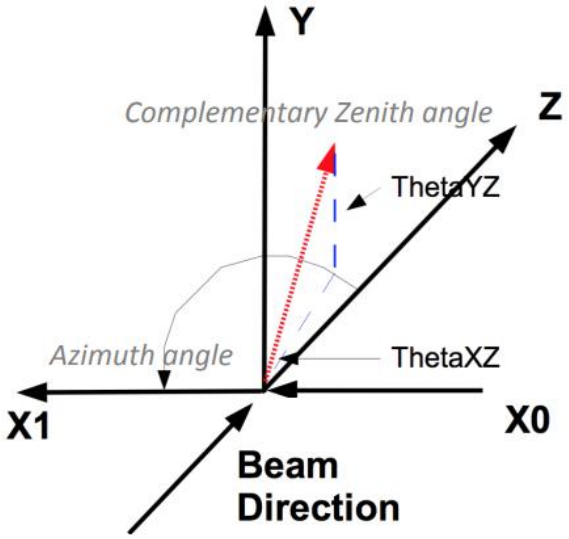


Do the peaks correspond to the boundaries between the different APAs (2.3m) and what are they due to?

Diagonal ~ 12m

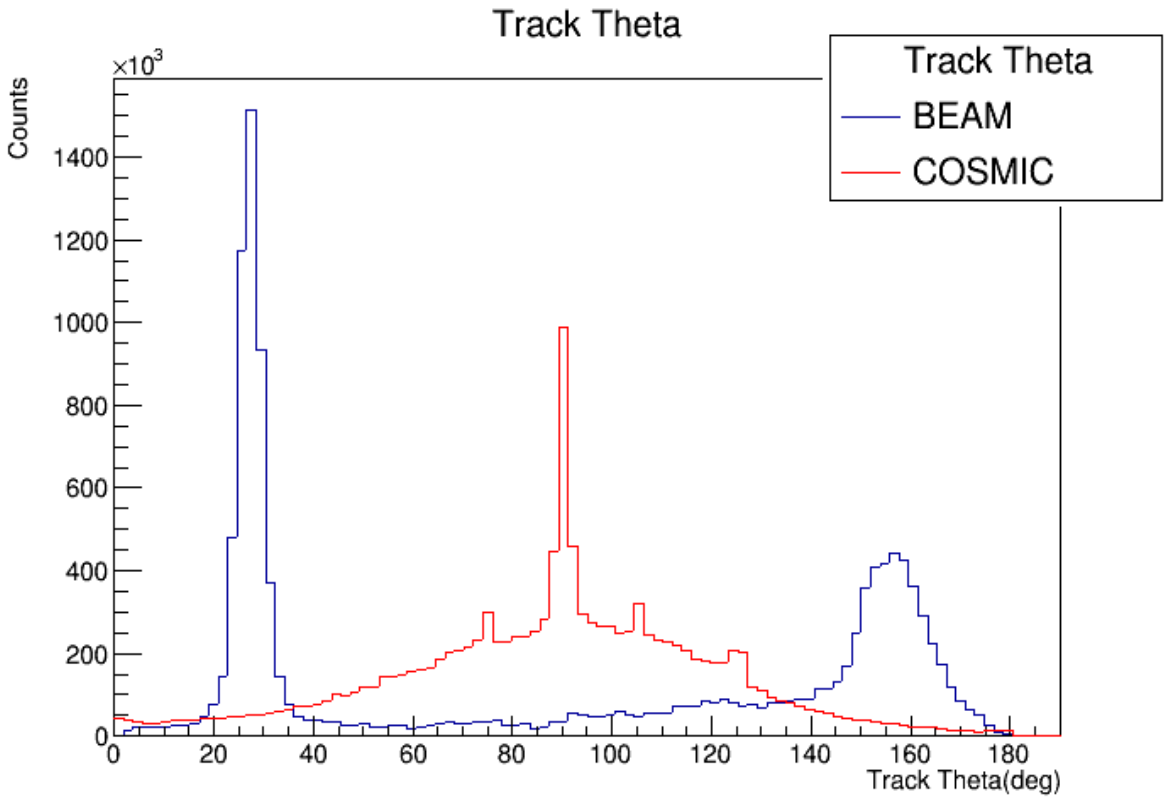


TRACK THETA



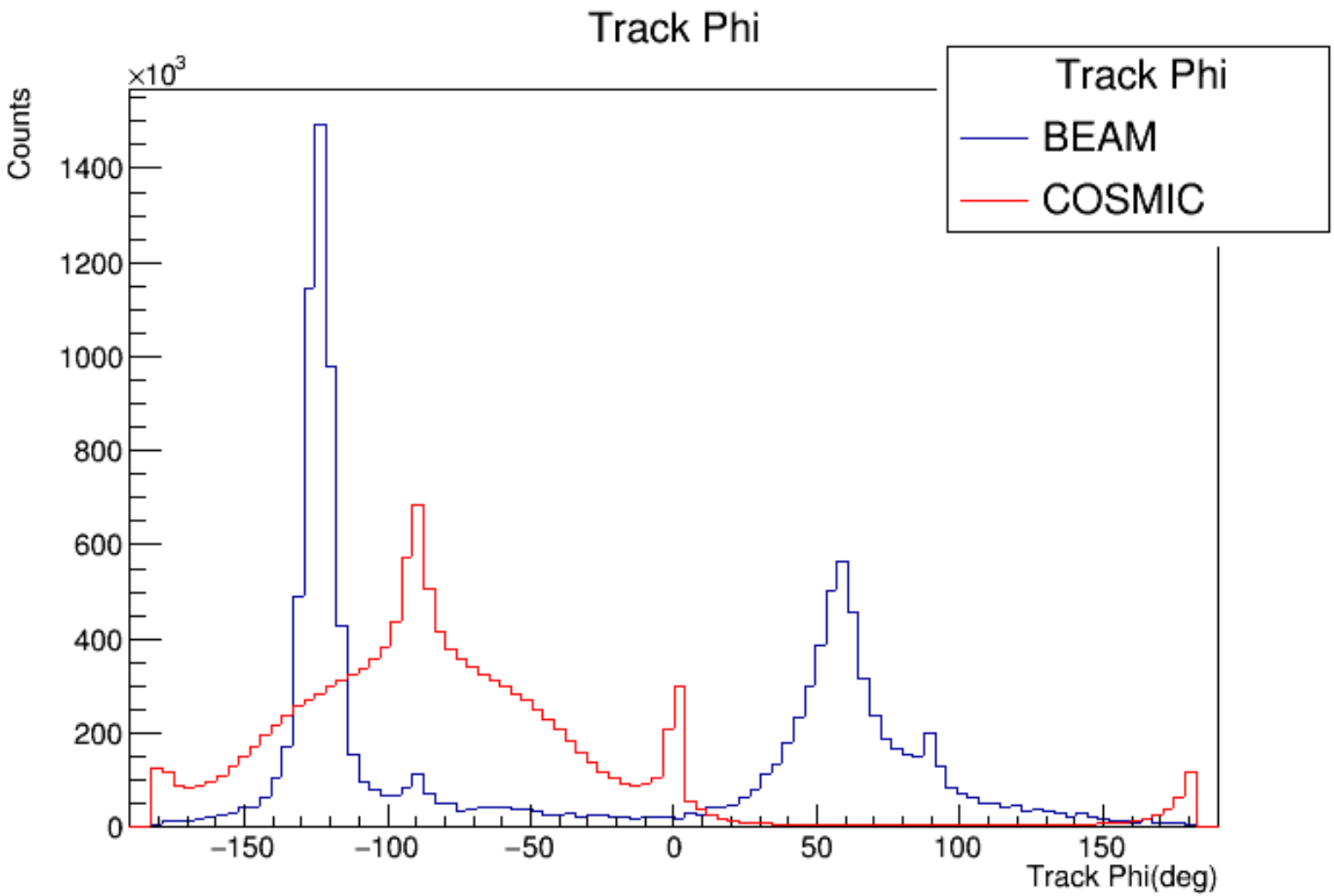
beam angle range: ±15° in XZ, ±10° in YZ

From a webpage discussing simulation to test the effect of beam orientation in the context of observing inefficiencies for wire-plane-parallel



- Is it reasonable to find peak for the beam particles at 30?

TRACK PHI



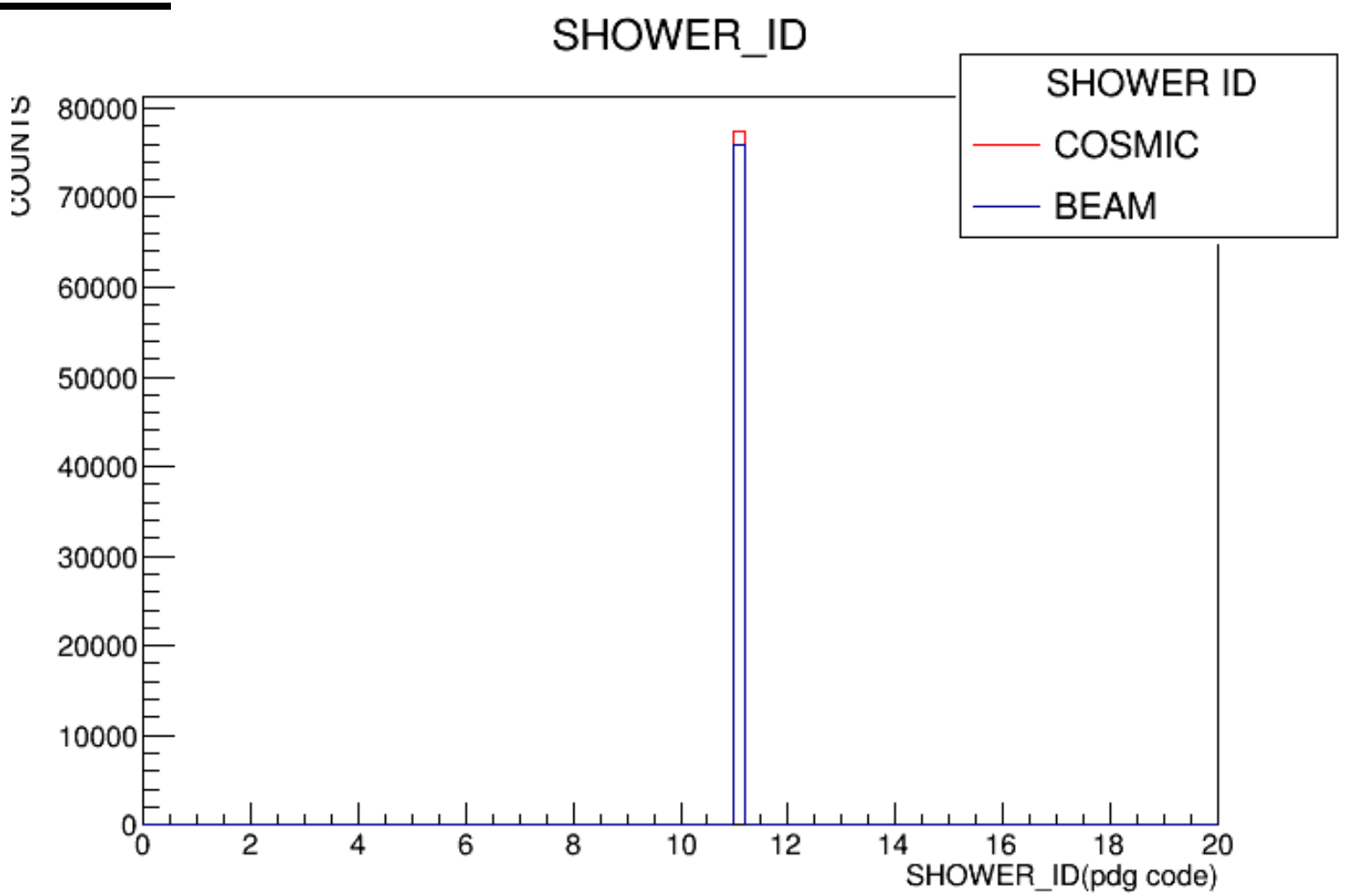
- Is this the phi angle expected for beam events?

SHOWERS

Code for Finding Showers

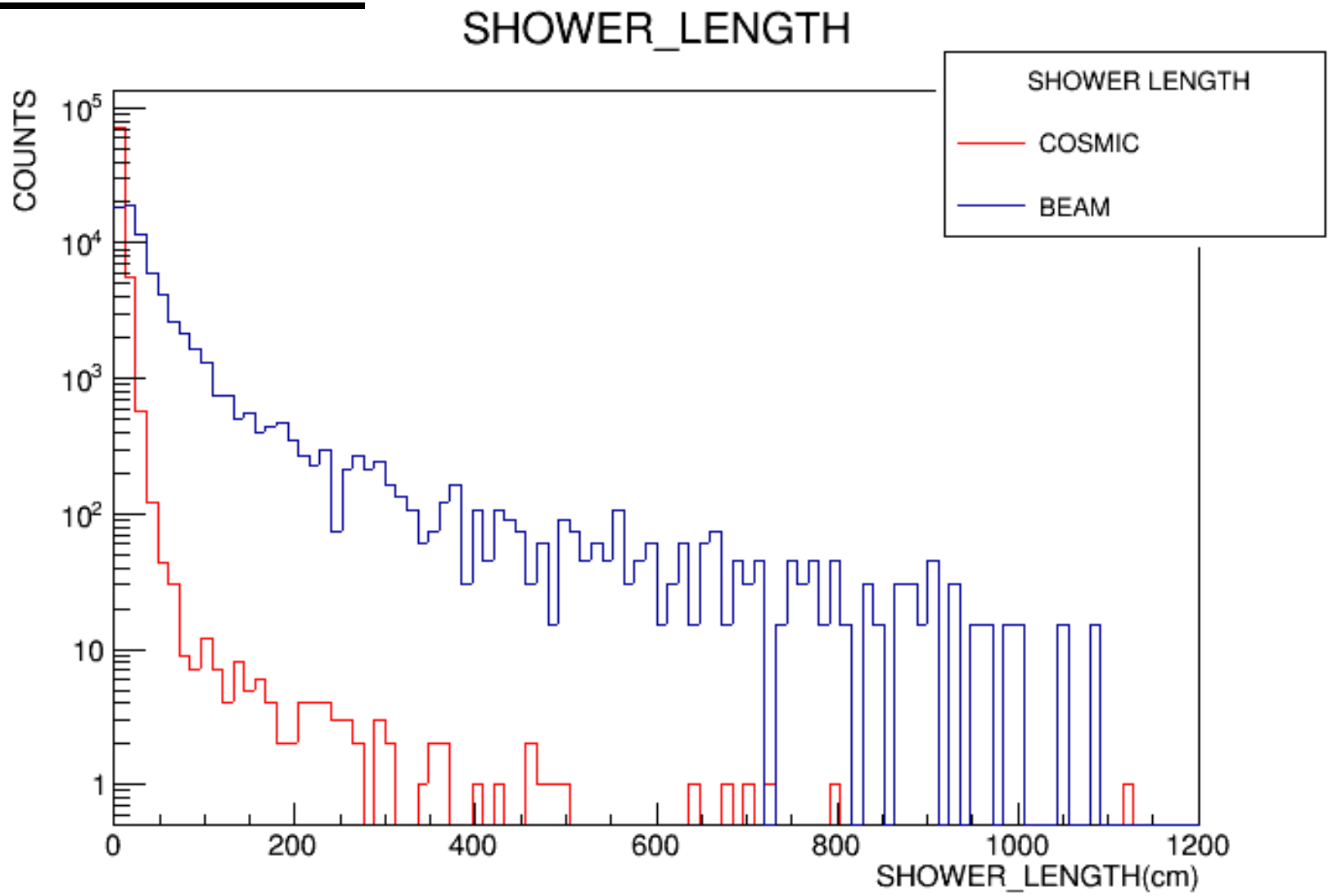
```
    // NOW WE GO TO THE SHOWER

if(pfpShower.size()>0){

  art::Ptr<recob::Shower> ThisShowers = pfpShower.at(0);

  ShowerLengthOfEverything = ThisShowers->Length();
  ShowerIDofEverything = particle.PdgCode();
  ShowerOpenAngleOfEverything = ThisShowers->OpenAngle()*180/3.14;

  EVERYSHOWER_ID.push_back(ShowerIDofEverything);
  EVERYSHOWER_LENGTH.push_back(ShowerLengthOfEverything);
  EVERYSHOWER_OPENANGLE.push_back(ShowerOpenAngleOfEverything);
  EVERYTRACK_LENGTH.push_back(LengthOfEverything);
  EVERYTRACK_ID.push_back(IDofEverything);
  EVERYTRACK_ANGLE.push_back(AngleOfEverything);
  EVERYTRACK_PHI.push_back(PhiOfEverything);
```

SHOWER ID



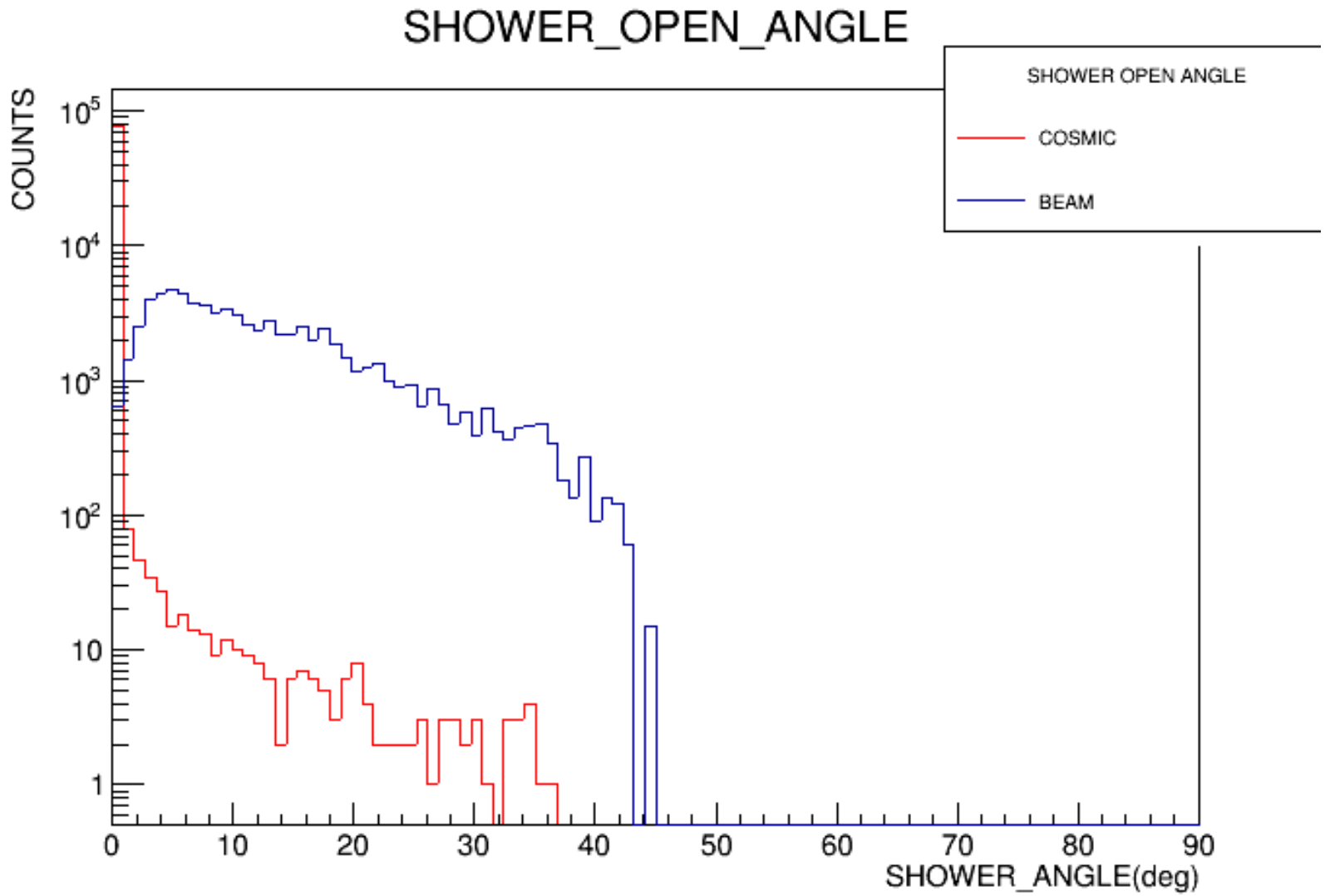
- Shower ID 11 correspond to electron

SHOWER LENGTH



Shower length in cosmic events (from delta rays?) smaller than beam ones as expected.

SHOWER OPEN ANGLE

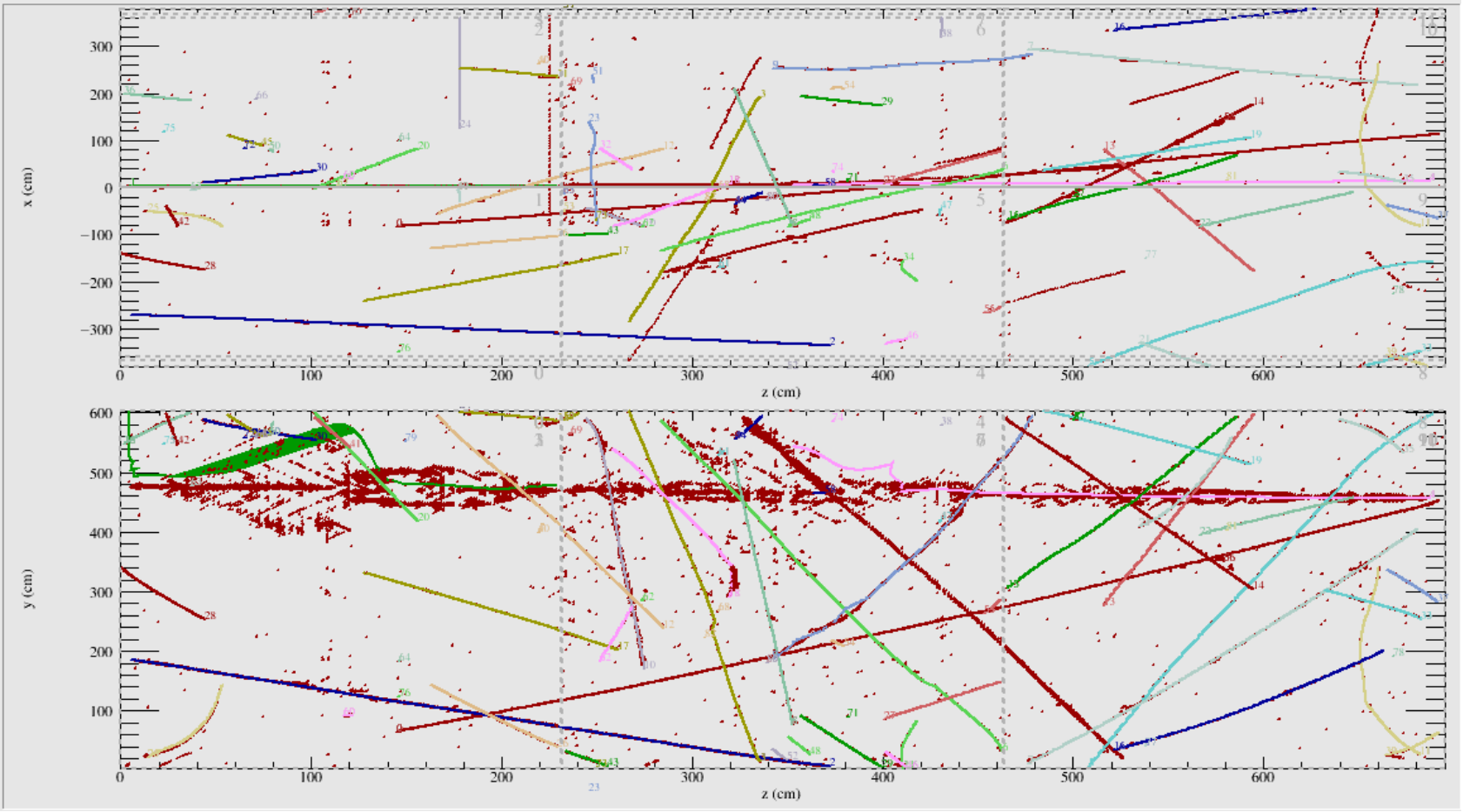


Summary

- Have written an analysis module to study and process beam and cosmic events.
- Have analyzed a single test beam run (5387) and shown preliminary, first results on reconstructed tracks, showers and beam related info.
- Any suggestions, comments, corrections would be more than welcome !

Back UP

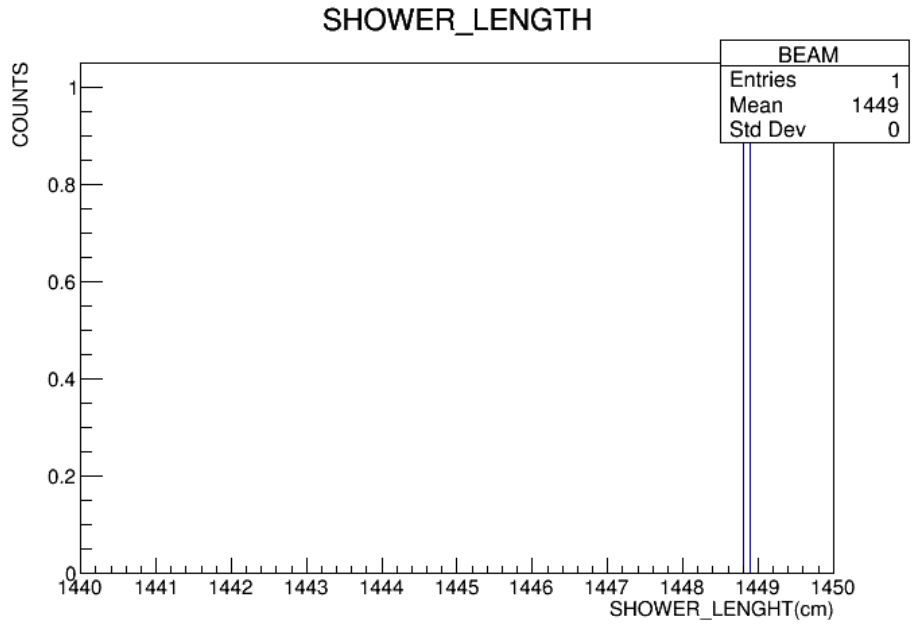
4/9/2019 16/09/19

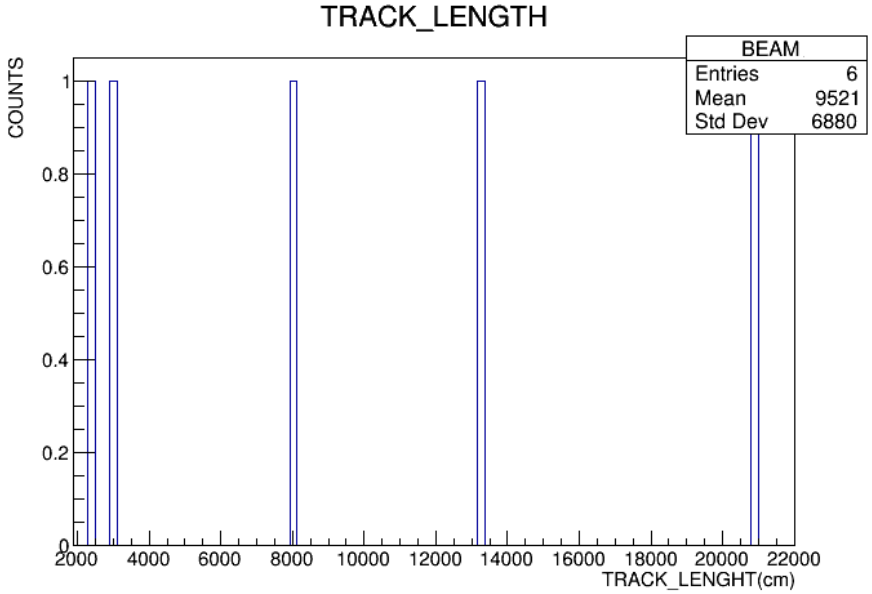


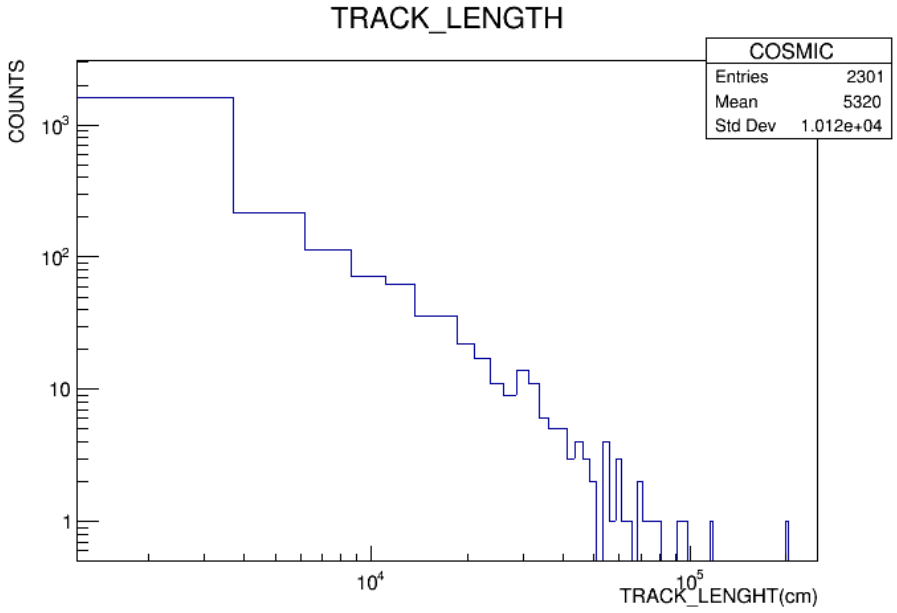
The particle ,which had length 208m, has :

X(start)	4.45759
Y(start)	478.993
Z(start)	228.252
X(end)	4.20206
Y(end)	696.822
Z(end)	5.15406

Back UP


SHOWER_LENGTH
BEAM
Entries 1
Mean 1449
Std Dev 0
COUNTS
SHOWER_LENGHT(cm)


TRACK_LENGTH
BEAM
Entries 6
Mean 9521
Std Dev 6880
COUNTS
TRACK_LENGHT(cm)


TRACK_LENGTH
COSMIC
Entries 2301
Mean 5320
Std Dev 1.012e+04
COUNTS
TRACK_LENGHT(cm)

BACK UP


```
Begin processing the 82nd record. run: 5387 subRun: 1 event: 28658 at 07-Sep-2019 13:41:32 CDT
Successed quality check
Pressures: 0.106813 1.00317
TOF invalid
electron Pressures: 0.106813 1.00317
TOF invalid
0
muon Pressures: 0.106813 1.00317
TOF invalid
0
pion Pressures: 0.106813 1.00317
TOF invalid
0
kaon Pressures: 0.106813 1.00317
TOF invalid
0
proton Pressures: 0.106813 1.00317
TOF invalid
0
proton Pressures: 0.106813 1.00317
TOF invalid
0
```

The reason is that we have events ,which were passed successfully the quality check (so they have match=1), but they had invalid TOF . So, the BeamID wasn't recorded.

np04_raw_run005387_0018_dl8_reco_12900889_0_20181102T023518.root

This is the file, I used for this slide

BACK UP

```
Begin processing the 38th record. run: 5387 subRun: 1 event: 96234 at 18-Sep-2019 06:08:56 CDT
Successed quality check
Pressures: 0.106813 0.996801
High pressure status invalid
electron Pressures: 0.106813 0.996801
High pressure status invalid
0
muon Pressures: 0.106813 0.996801
High pressure status invalid
0
pion Pressures: 0.106813 0.996801
High pressure status invalid
0
kaon Pressures: 0.106813 0.996801
High pressure status invalid
0
proton Pressures: 0.106813 0.996801
High pressure status invalid
0
proton Pressures: 0.106813 0.996801
High pressure status invalid
0

 beam momentum  -999
 ftof  155.738
 ftof_expElec  -999
 ftof_expMuon  -999
 ftof_expPion  -999
 ftof_expKaon  -999
 ftof_expProt  -999
 ftof_expDeut  -999
```

np04_raw_run005387_0060_dl3_reco_12784844_0_20181102T105929.root

This is the file, I used for this slide

BACK UP

```
if(pfpShower.size()==0&&pfpTrack.size()==0){
  std::cout << "YEAH,WE DONT HAVE TRACK OR SHOWER" << std::endl;
  EVERYSHOWER_ID.push_back(ShowerIDofEverything);
  EVERYSHOWER_LENGTH.push_back(ShowerLengthOfEverything);
  EVERYSHOWER_OPENANGLE.push_back(ShowerOpenAngleOfEverything);
  EVERYTRACK_LENGTH.push_back(LengthOfEverything);
  EVERYTRACK_ID.push_back(IDofEverything);
  EVERYTRACK_ANGLE.push_back(AngleOfEverything);
  EVERYTRACK_PHI.push_back(PhiOfEverything);
```

```
np04_raw_run005387_0018_dl8_reco_12900889_0_20181102T023518.root
```

For example, this file has 2 particles with no track or showers in the event 27688

Beam Event and “matching” info