ProtoDUNE Beam Simulation Interface

(also some cosmic studies)

Leigh Whitehead

ProtoDUNE Reconstruction Meeting

28/11/16

Introduction

- We have had some beam simulation files for a while now
  - We need an interface to these files in larsoft
- I have an implementation for the current format of these files
  - This will have to follow any updates to the files such as the inclusion of the beam halo once the beam design has been finalised
- I outline the input files and the module I have written in these slides

H4 Beam Simulation Files

- The H4 beam simulation files from Nikos and Yannis contain a TNTuple called NTuples/GoodParticle

- There are a lot of variables in there, but the ones of interest are:

Lag_ENTRY_x	Entry x position (cm)
Lag_ENTRY_y	Entry y position (cm)
Lag_ENTRY_z	Entry z position (cm)
Lag_ENTRY_t	Entry time (ns)
Lag_ENTRY_Px	Entry x momentum (MeV/c)
Lag_ENTRY_Py	Entry y momentum (MeV/c)
Lag_ENTRY_Pz	Entry z momentum (MeV/c)
Lag_ENTRY_PDGid	Particle PDG code
TRIG2_t	Second trigger time (ns) (Use for T0?)

This is just to define a T0, will check if I should be using a different time

ProtoDUNEBeam Module

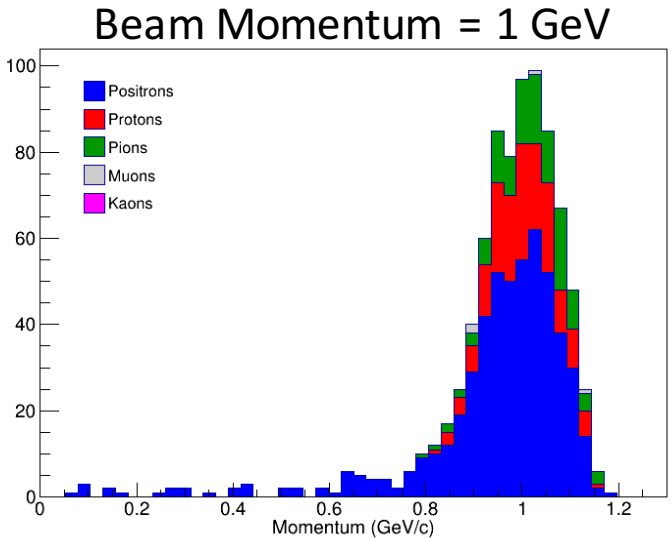
- I have added a file to larsim/EventGenerator called ProtoDUNEBeam_module.cc
- Contains an art::EDProducer inherited class called ProtoDUNEBeam.
- Reads in the H4 Beam file using a TTree, taking each entry as a separate art::Event.
  - This might well change in the future, but it is true for the current files
- Currently lives in the branch features/lhw_protoDUNE_evgen
  - Up-to-date with v06_15_01

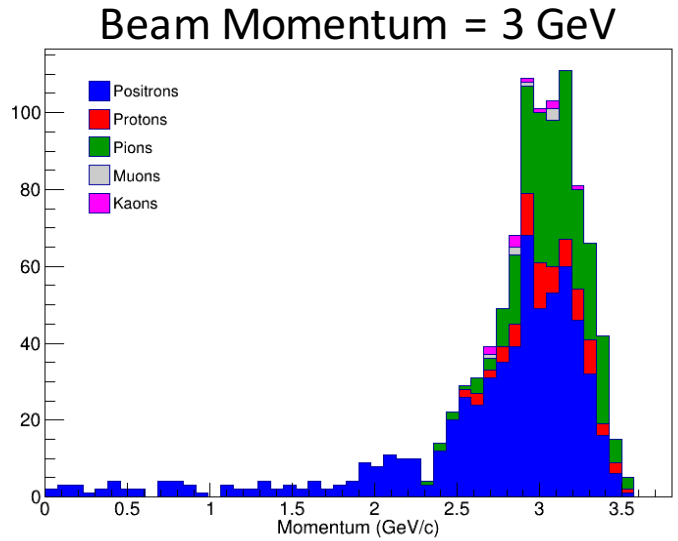
ProtoDUNEBeam Module

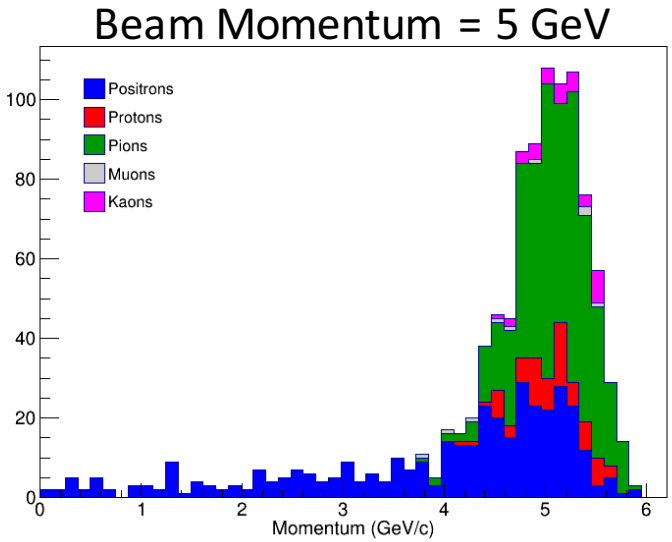
- I have a .fcl file to run the code too, please ask if you want it
- The beam entry point defaults to the values used in protoDUNE_gensingle
- Rather, I shift the central value to keep and x,y spread

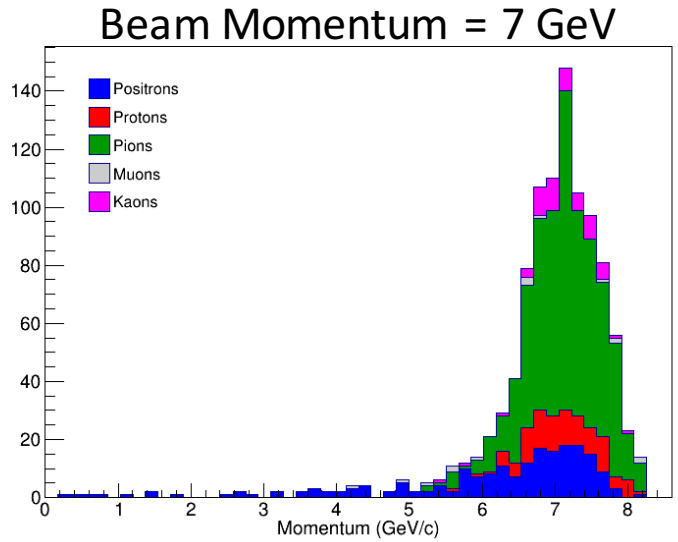
```
#include "services_dune.fcl"
#include "protodunebeam.fcl"

process_name: H4BeamGen

services:
{
  # Load the service that manages root files for histograms.
#  TFileService: { fileName: "gensingle_beam_protoDUNE_hist.root" }
  TimeTracker:       {}
  RandomNumberGenerator: {} #ART native random number generator
  FileCatalogMetadata:  @local::art_file_catalog_mc
  @table::protodune_simulation_services
}

#Start each new event with an empty event.
source:
{
  module_type: EmptyEvent
  timestampPlugin: { plugin_type: "GeneratedEventTimestamp" }
  maxEvents:   100
  firstRun:    1           # Run number to use for this file
  firstEvent:  1           # number of first event in the file
}

physics:
{
 producers:
 {
   generator: @local::protodune_beam
   rns:       { module_type: "RandomNumberSaver" }
 }
 simulate: [ rns, generator ]
 stream1:  [ out1 ]
 trigger_paths: [simulate]
 end_paths:     [stream1]
}

outputs:
{
 out1:
 {
   module_type: RootOutput
   fileName:    "gensingle_protoDUNE.root" #default file name, can override from command line with -o or --output
   dataTier: "generated"
   compressionLevel: 1
 }
}

physics.producers.generator.FileName: "/mnt/nas00/scratch/h4/H4_TILT22_APR_FTFP_BERT_6GeV_6M.root"
physics.producers.generator.TreeName: "NTuples/GoodParticle"
physics.producers.generator.StartEvent: 0
physics.producers.generator.BeamX: 118.106 # In cm, taken from protoDUNE_gensingle.fcl
physics.producers.generator.BeamY: 395.649
physics.producers.generator.BeamZ: -196.113
physics.producers.generator.RotateXZ: -8.189 # In degrees, taken from protoDUNE_gensingle.fcl
physics.producers.generator.RotateYZ: 11.229
```

GEANT True Momentum Distributions









Beam Interface Summary

- I have written an interface to the protoDUNE H4 beam files
  - Generates a single event for each beam particle
  - Distributions look sensible
- I will continue to develop the interface as the beam simulation is updated
  - The next major change foreseen is the addition of the halo muons
- At some point the simulation will hopefully provide the beam PID detector response
  - Will have to think how best to incorporate this information into LarSoft when we have it

Bonus Cosmic Slides

- I've also been looking at the files Elizabeth made.

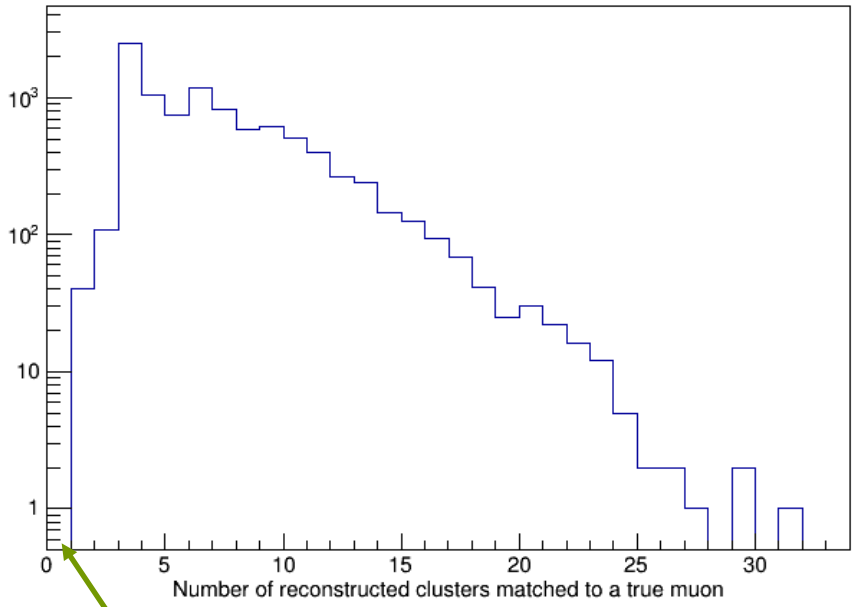
- Overlaid cosmic muons on a 1 GeV muon beam sample.
  - Typically 400 muons/antimuons per event with ~90 crossing the active volume

Based on some AnalysisTree code

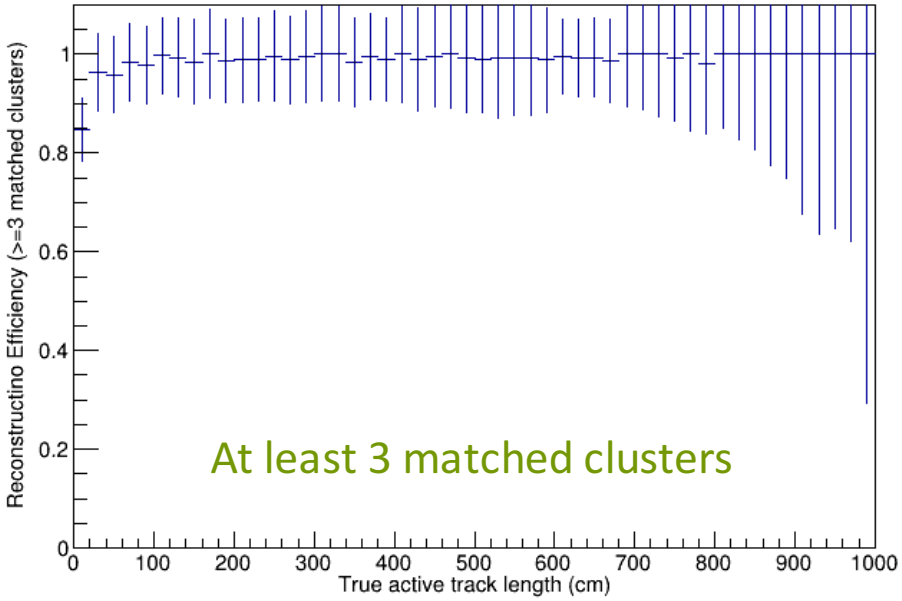
- Procedure:
  - Look for all true muons and antimuons
  - Select only those that traverse the active TPC volume
  - Count the number of matched reconstructed clusters and tracks
  - Define two efficiencies:
    - One with number of matched reco clusters > 2
    - One with number of matched reco tracks > 0 (both pmtrack and pandora)

Clustering Efficiency

- Clustering is independent of pmtrack / pandora, so it is the same for both.

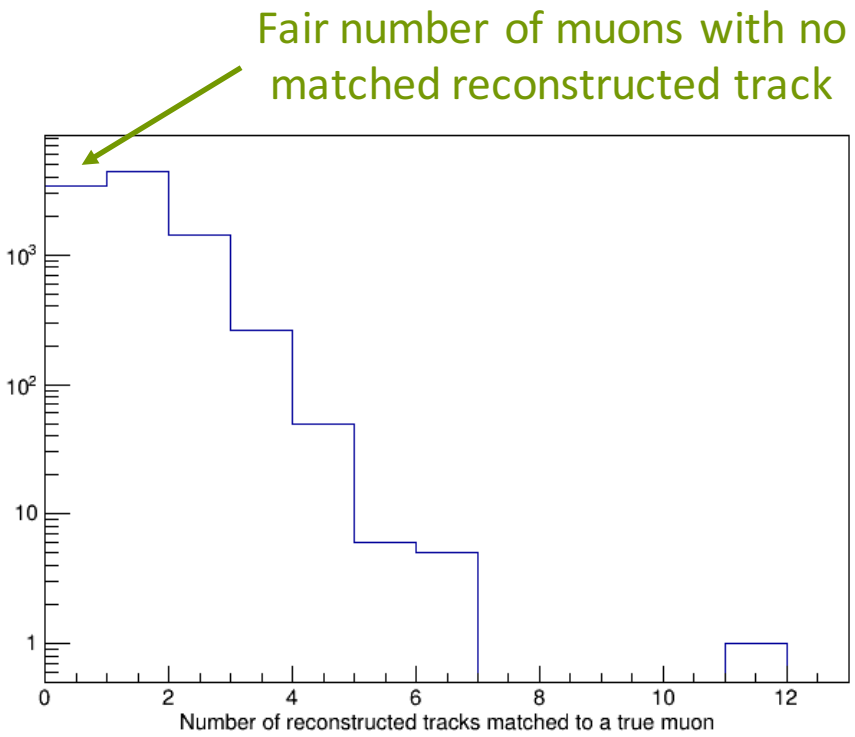


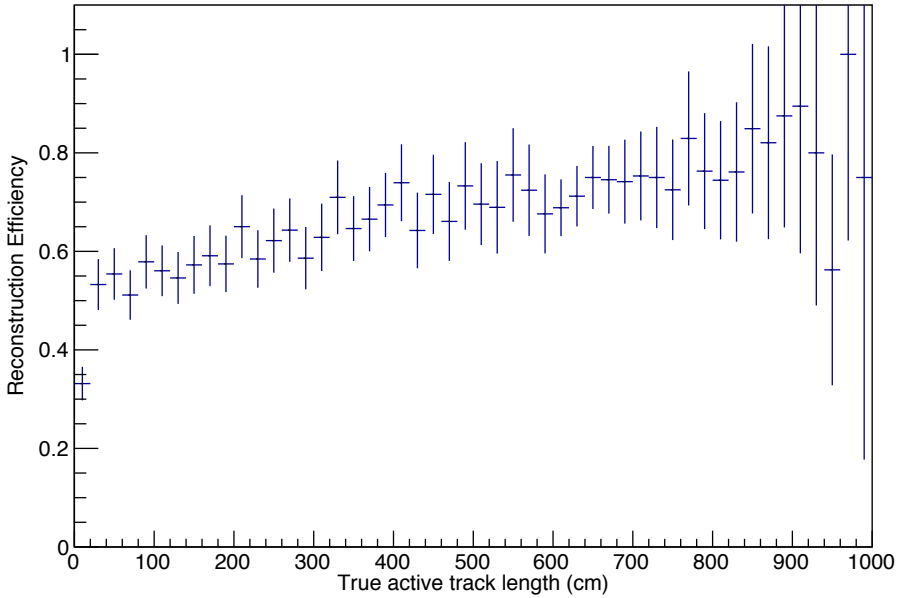
All muons have at least one cluster.



Tracks from pmtrack

- Now look at tracks from pmtrack



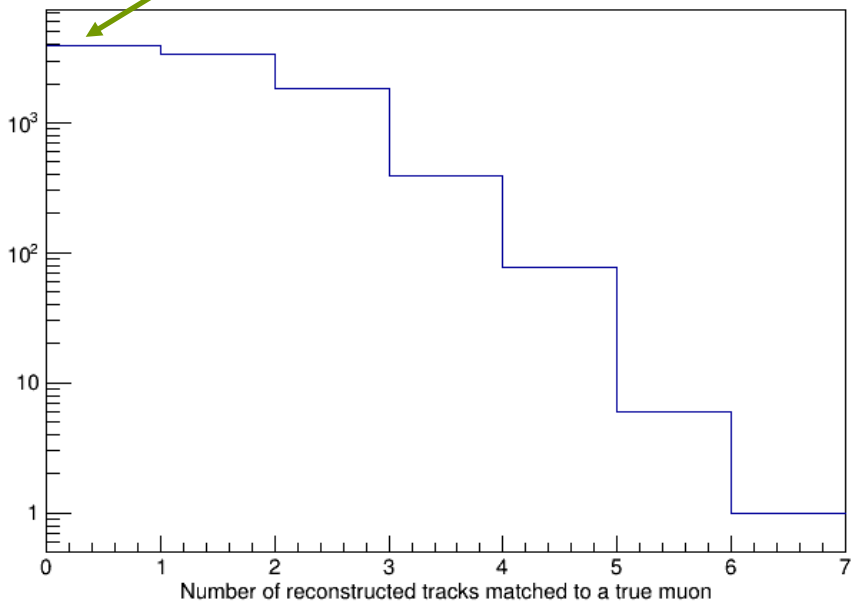


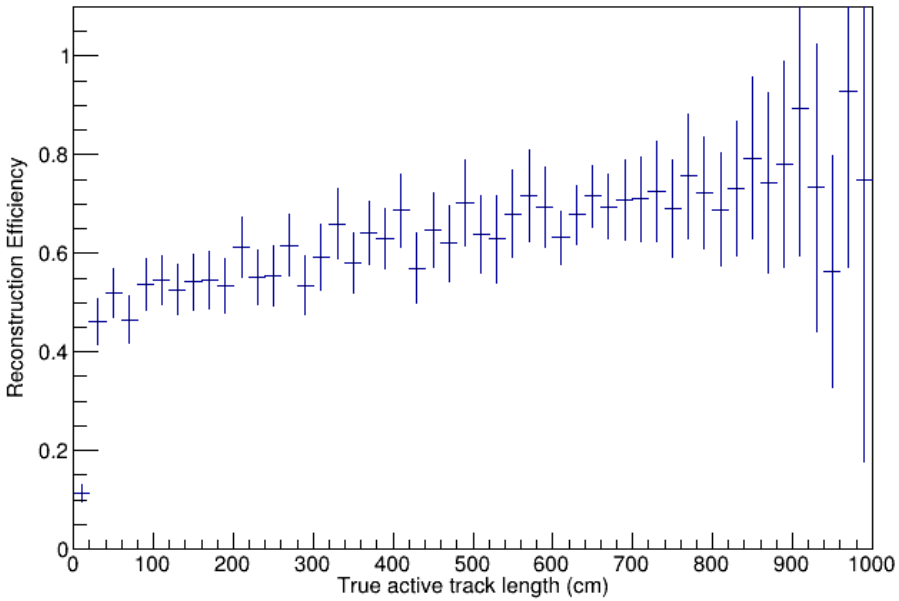
- Integrated efficiency = 64.2%
- Reconstructed 100 / 100 beam muons

Tracks from pandora

- Compare to tracks from pandora

More muons with no matched reconstructed track





- Integrated efficiency = 59.3%
- Reconstructed 97 / 100 beam muons

Cosmics Summary

- See that all true cosmic muons and antimuons have at least one associated reconstructed cluster

- However, the integrated efficiency for having at least one reconstructed track is fairly low:
  - Pmtrack: 64.2%
  - Pandora: 59.3%

- Plot shows the track length spectrum over which the efficiency was integrated