Practical guide to getting started in LArSoft

Tingjun Yang

LArSoft Workshop 2019

Jun 24, 2019

Fermilab

Introduction

- What will be discussed in this talk
  - Data products saved in a reconstructed larsoft file.
  - Practical guide to getting started in LArSoft
- What will not be discussed
  - Detailed information on simulation or reconstruction algorithms.
- I will use many ProtoDUNE examples. They apply to most LArTPC experiments.
- A lot can be learnt from existing code, talking to people and asking for help on SLACK.
- https://cdcvs.fnal.gov/redmine/projects/larsoft/wiki

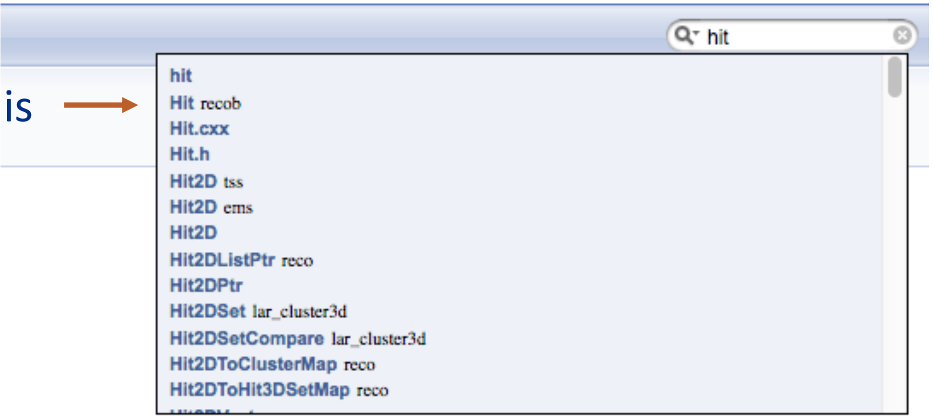
Data products

- raw::* - raw data
  - raw::RawDigit, raw::AuxDetDigit, raw::OpDetPulse, raw::OpDetWaveform, raw::Trigger, raw::BeamInfo, etc.
- recob::* - reconstructed information
  - recob::Wire, recob::Hit, recob::Cluster, recob::EndPoint2D, recob::Vertex, recob::PFParticle, recob::Track, recob::Shower, recob::OpHit, recob::OpFlash, etc.
- anab::* - information that is derived from reconstruction information
  - anab::Calorimetry, anab::ParticleID, anab::CosmicTag, anab::T0, etc.
- simb::* - simulation information
  - simb::MCTruth, simb::MCParticle, simb::MCFlux, etc.
- Associations - links between different data products
  - https://cdcvs.fnal.gov/redmine/projects/larsoft/wiki/Use_associations

Get information from Doxygen

- http://nusoft.fnal.gov/larsoft/doxsvn/html/index.html

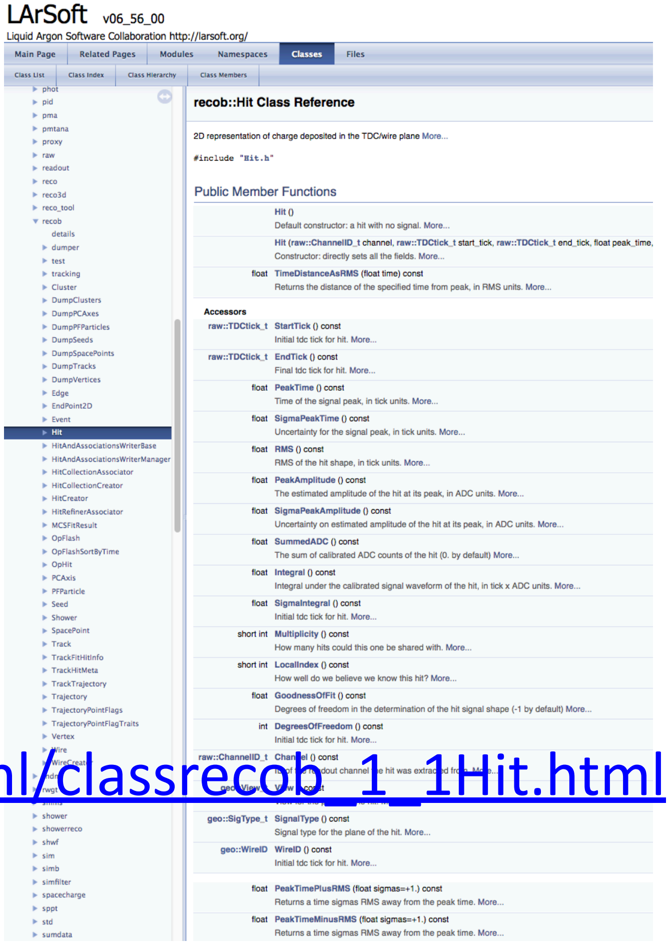
If you want to get information for **recob::Hit**, type hit here



hit

hit
Hit recob
Hit.cxx
Hit.h
Hit2D tss
Hit2D ems
Hit2D
Hit2DListPtr reco
Hit2DPtr
Hit2DSet lar_cluster3d
Hit2DSetCompare lar_cluster3d
Hit2DToClusterMap reco
Hit2DToHit3DSetMap reco

Click this



LArSoft v06_56_00
Liquid Argon Software Collaboration http://larsoft.org/

phot
pid
pma
pmtana
proxy
raw
readout
reco
reco3d
reco_tool
recob
details
dumper
test
tracking
Cluster
DumpClusters
DumpPCAxes
DumpPFParticles
DumpSeeds
DumpSpacePoints
DumpTracks
DumpVertices
Edge
EndPoint2D
Event
Hit
HitAndAssociationsWriterBase
HitAndAssociationsWriterManager
HitCollectionAssociator
HitCollectionCreator
HitCreator
HitRefinerAssociator
MCSFitResult
OpFlash
OpFlashSortByTime
OpHit
PCAxis
PFParticle
Seed
Shower
SpacePoint
Track
TrackFitHitInfo
TrackHitMeta
TrackTrajectory
Trajectory
TrajectoryPointFlags
TrajectoryPointFlagTraits
Vertex
Wire
WireCreator
rndm
rwgt
[illegible]
shower
showerreco
shwf
sim
simb
simfilter
spacecharge
sppt
std
sumdata

recob::Hit Class Reference

2D representation of charge deposited in the TDC/wire plane More...

```
#include "Hit.h"
```

Public Member Functions

Hit ()
Default constructor: a hit with no signal. More...

Hit (raw::ChannelID_t channel, raw::TDCtick_t start_tick, raw::TDCtick_t end_tick, float peak_time,
Constructor: directly sets all the fields. More...

float TimeDistanceAsRMS (float time) const
Returns the distance of the specified time from peak, in RMS units. More...

Accessors

raw::TDCtick_t StartTick () const
Initial tdc tick for hit. More...

raw::TDCtick_t EndTick () const
Final tdc tick for hit. More...

float PeakTime () const
Time of the signal peak, in tick units. More...

float SigmaPeakTime () const
Uncertainty for the signal peak, in tick units. More...

float RMS () const
RMS of the hit shape, in tick units. More...

float PeakAmplitude () const
The estimated amplitude of the hit at its peak, in ADC units. More...

float SigmaPeakAmplitude () const
Uncertainty on estimated amplitude of the hit at its peak, in ADC units. More...

float SummedADC () const
The sum of calibrated ADC counts of the hit (0. by default) More...

float Integral () const
Integral under the calibrated signal waveform of the hit, in tick x ADC units. More...

float SigmaIntegral () const
Initial tdc tick for hit. More...

short int Multiplicity () const
How many hits could this one be shared with. More...

short int LocalIndex () const
How well do we believe we know this hit? More...

float GoodnessOfFit () const
Degrees of freedom in the determination of the hit signal shape (-1 by default) More...

int DegreesOfFreedom () const
Initial tdc tick for hit. More...

raw::ChannelID_t Channel () const
ID of the readout channel the hit was extracted from. More...

geo::View_t View () const
View for the plane of the hit. More...

geo::SigType_t SignalType () const
Signal type for the plane of the hit. More...

geo::WireID WireID () const
Initial tdc tick for hit. More...

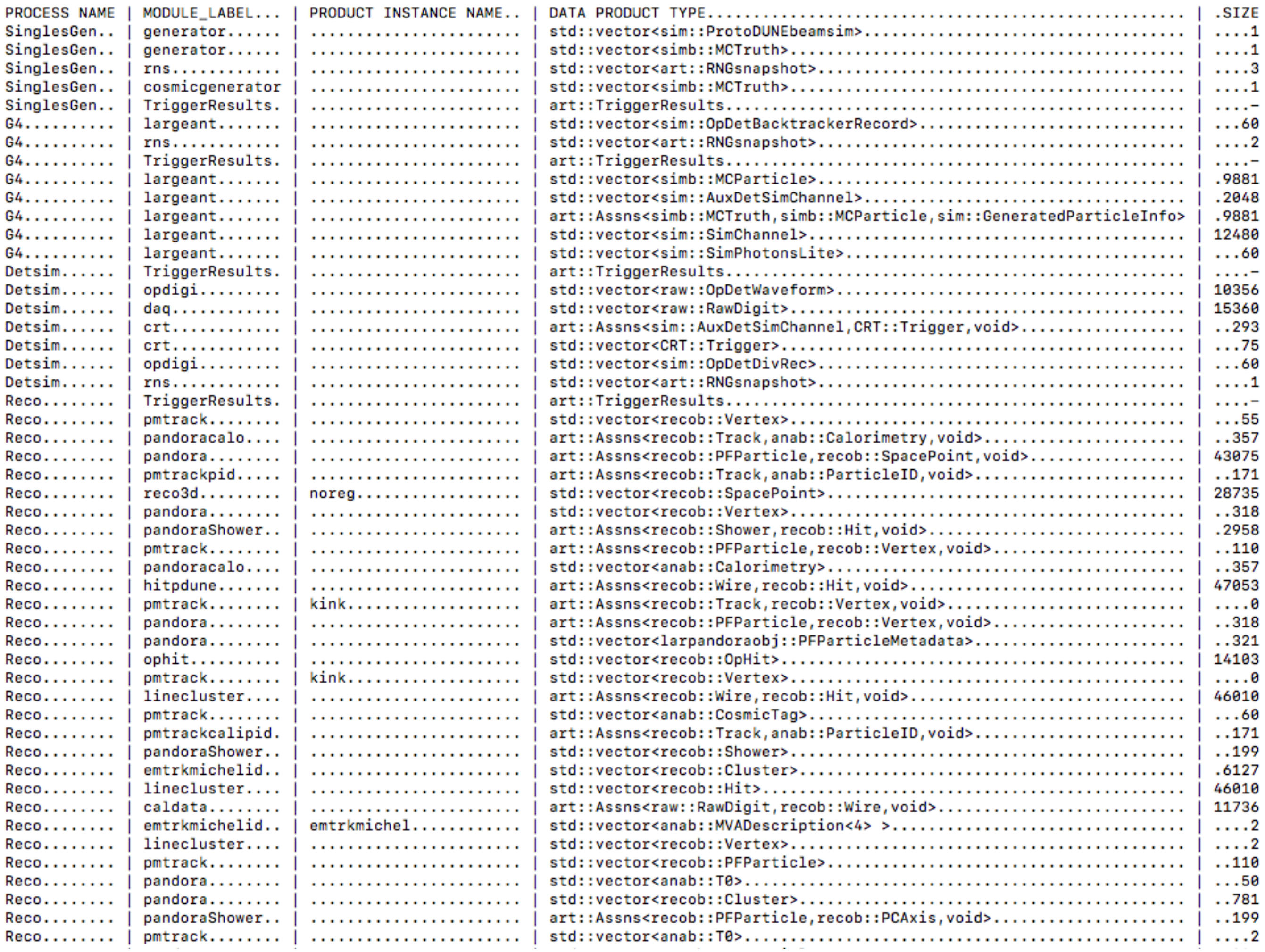
float PeakTimePlusRMS (float sigmas=+1.) const
Returns a time sigmas RMS away from the peak time. More...

float PeakTimeMinusRMS (float sigmas=+1.) const
Returns a time sigmas RMS away from the peak time. More...

http://nusoft.fnal.gov/larsoft/doxsvn/html/classrecob_1_1Hit.html

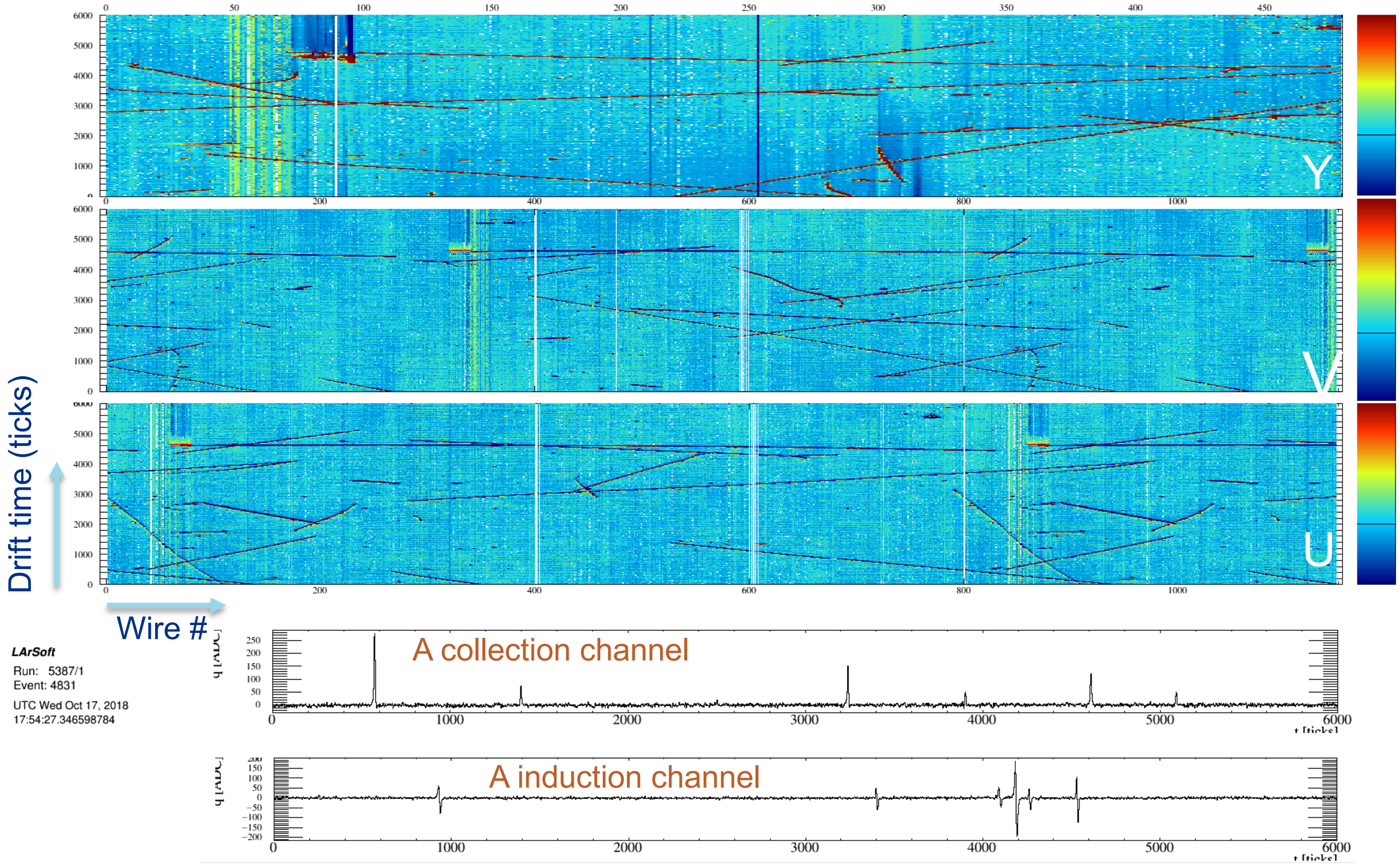
Get information from an art file

- lar -c **eventdump.fcl** /pnfs/dune/tape_backed/dunepro/mcc11/protodune/mc/full-reconstructed/06/67/65/21/mcc11_protoDUNE_sp_reco_12231114_0_4a73bdae-00a8-428a-9269-fe18d6cb6db4.root -n 1



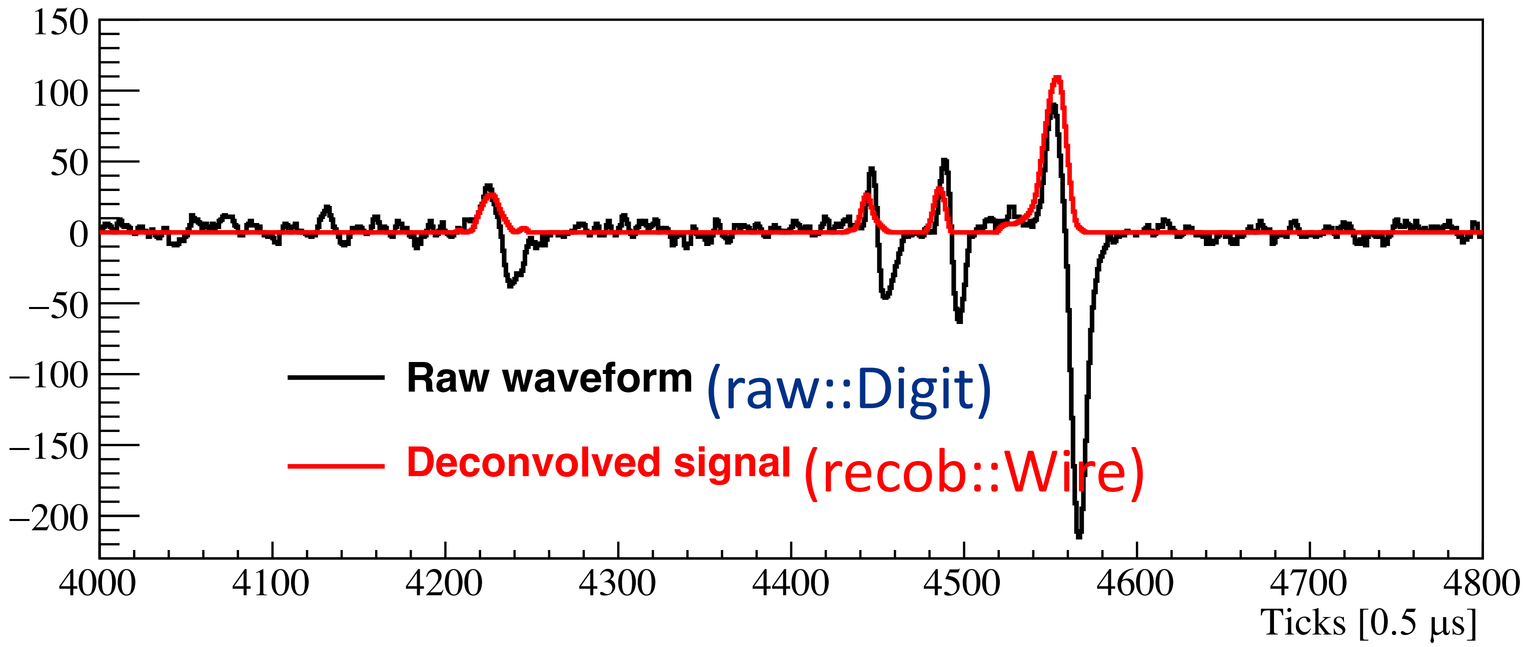
```
PROCESS NAME | MODULE_LABEL... | PRODUCT INSTANCE NAME.. | DATA PRODUCT TYPE.................................................... | .SIZE
SinglesGen.. | generator...... | ....................... | std::vector<sim::ProtoDUNEbeamsim>................................... | ....1
SinglesGen.. | generator...... | ....................... | std::vector<simb::MCTruth>........................................... | ....1
SinglesGen.. | rns............ | ....................... | std::vector<art::RNGsnapshot>........................................ | ....3
SinglesGen.. | cosmicgenerator | ....................... | std::vector<simb::MCTruth>........................................... | ....1
SinglesGen.. | TriggerResults. | ....................... | art::TriggerResults.................................................. | ....-
G4.......... | largeant....... | ....................... | std::vector<sim::OpDetBacktrackerRecord>............................. | ...60
G4.......... | rns............ | ....................... | std::vector<art::RNGsnapshot>........................................ | ....2
G4.......... | TriggerResults. | ....................... | art::TriggerResults.................................................. | ....-
G4.......... | largeant....... | ....................... | std::vector<simb::MCParticle>........................................ | .9881
G4.......... | largeant....... | ....................... | std::vector<sim::AuxDetSimChannel>................................... | .2048
G4.......... | largeant....... | ....................... | art::Assns<simb::MCTruth,simb::MCParticle,sim::GeneratedParticleInfo> | .9881
G4.......... | largeant....... | ....................... | std::vector<sim::SimChannel>......................................... | 12480
G4.......... | largeant....... | ....................... | std::vector<sim::SimPhotonsLite>..................................... | ...60
Detsim...... | TriggerResults. | ....................... | art::TriggerResults.................................................. | ....-
Detsim...... | opdigi......... | ....................... | std::vector<raw::OpDetWaveform>...................................... | 10356
Detsim...... | daq............ | ....................... | std::vector<raw::RawDigit>........................................... | 15360
Detsim...... | crt............ | ....................... | art::Assns<sim::AuxDetSimChannel,CRT::Trigger,void>.................. | ..293
Detsim...... | crt............ | ....................... | std::vector<CRT::Trigger>............................................ | ...75
Detsim...... | opdigi......... | ....................... | std::vector<sim::OpDetDivRec>........................................ | ...60
Detsim...... | rns............ | ....................... | std::vector<art::RNGsnapshot>........................................ | ....1
Reco........ | TriggerResults. | ....................... | art::TriggerResults.................................................. | ....-
Reco........ | pmtrack........ | ....................... | std::vector<recob::Vertex>........................................... | ...55
Reco........ | pandoracalo.... | ....................... | art::Assns<recob::Track,anab::Calorimetry,void>...................... | ..357
Reco........ | pandora........ | ....................... | art::Assns<recob::PFParticle,recob::SpacePoint,void>................. | 43075
Reco........ | pmtrackpid..... | ....................... | art::Assns<recob::Track,anab::ParticleID,void>....................... | ..171
Reco........ | reco3d......... | noreg.................. | std::vector<recob::SpacePoint>....................................... | 28735
Reco........ | pandora........ | ....................... | std::vector<recob::Vertex>........................................... | ..318
Reco........ | pandoraShower.. | ....................... | art::Assns<recob::Shower,recob::Hit,void>............................ | .2958
Reco........ | pmtrack........ | ....................... | art::Assns<recob::PFParticle,recob::Vertex,void>..................... | ..110
Reco........ | pandoracalo.... | ....................... | std::vector<anab::Calorimetry>....................................... | ..357
Reco........ | hitpdune....... | ....................... | art::Assns<recob::Wire,recob::Hit,void>.............................. | 47053
Reco........ | pmtrack........ | kink................... | art::Assns<recob::Track,recob::Vertex,void>.......................... | ....0
Reco........ | pandora........ | ....................... | art::Assns<recob::PFParticle,recob::Vertex,void>..................... | ..318
Reco........ | pandora........ | ....................... | std::vector<larpandoraobj::PFParticleMetadata>....................... | ..321
Reco........ | ophit.......... | ....................... | std::vector<recob::OpHit>............................................ | 14103
Reco........ | pmtrack........ | kink................... | std::vector<recob::Vertex>........................................... | ....0
Reco........ | linecluster.... | ....................... | art::Assns<recob::Wire,recob::Hit,void>.............................. | 46010
Reco........ | pmtrack........ | ....................... | std::vector<anab::CosmicTag>......................................... | ...60
Reco........ | pmtrackcalipid. | ....................... | art::Assns<recob::Track,anab::ParticleID,void>....................... | ..171
Reco........ | pandoraShower.. | ....................... | std::vector<recob::Shower>........................................... | ..199
Reco........ | emtrkmichelid.. | ....................... | std::vector<recob::Cluster>.......................................... | .6127
Reco........ | linecluster.... | ....................... | std::vector<recob::Hit>.............................................. | 46010
Reco........ | caldata........ | ....................... | art::Assns<raw::RawDigit,recob::Wire,void>........................... | 11736
Reco........ | emtrkmichelid.. | emtrkmichel............ | std::vector<anab::MVADescription<4> >................................ | ....2
Reco........ | linecluster.... | ....................... | std::vector<recob::Vertex>........................................... | ....2
Reco........ | pmtrack........ | ....................... | std::vector<recob::PFParticle>....................................... | ..110
Reco........ | pandora........ | ....................... | std::vector<anab::T0>................................................ | ...50
Reco........ | pandora........ | ....................... | std::vector<recob::Cluster>.......................................... | ..781
Reco........ | pandoraShower.. | ....................... | art::Assns<recob::PFParticle,recob::PCAxis,void>..................... | ..199
Reco........ | pmtrack........ | ....................... | std::vector<anab::T0>................................................ | ....2
```

raw::Digit



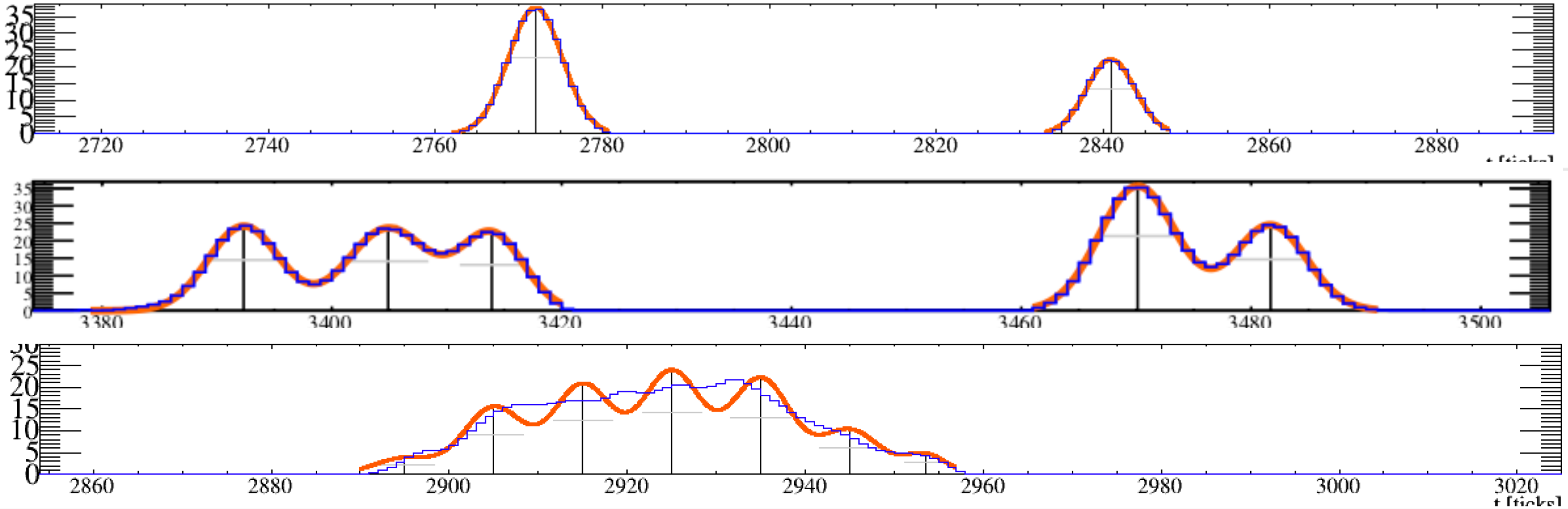
- One **raw::Digit** per channel, 15360 in total for ProtoDUNE.
- A **raw::Digit** has a vector of raw ADC counts. The size is determined by readout window, 6k by default, some runs were taken with 15k.
- Each tick is 0.5 μs.

recob::Wire



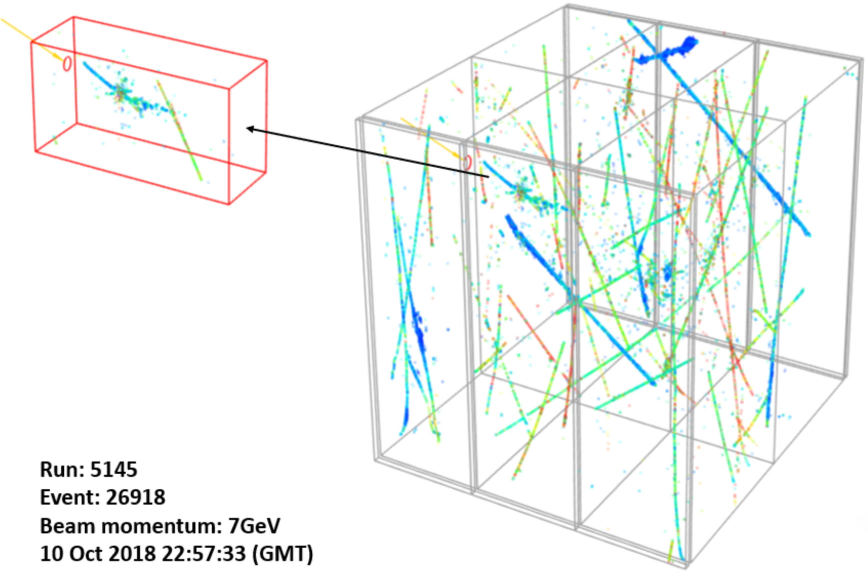
- Output of signal processing after noise filtering and deconvolution and ROI finding.
- A **recob::Wire** has a vector of float point numbers, which are deconvolved ADC counts.
- 2D deconvolution to account for induced charge on neighboring wires.

recob::Hit



- The gaussian hit finder module fits the deconvolved signal to a gaussian.
- Multiple gaussians are used to fit overlapping signal.
- The gaussian fit returns the peak time and the area, as well as wireID, width, peak amplitude etc.
- There can be several copies of hit collections – hits after disambiguation and refined hits after pattern recognition (e.g. trajcluster).
- Fit a very long pulse, the hit finder will return a train of hits with the same width. The maximum number of gaussians to fit and hit width are configurable.

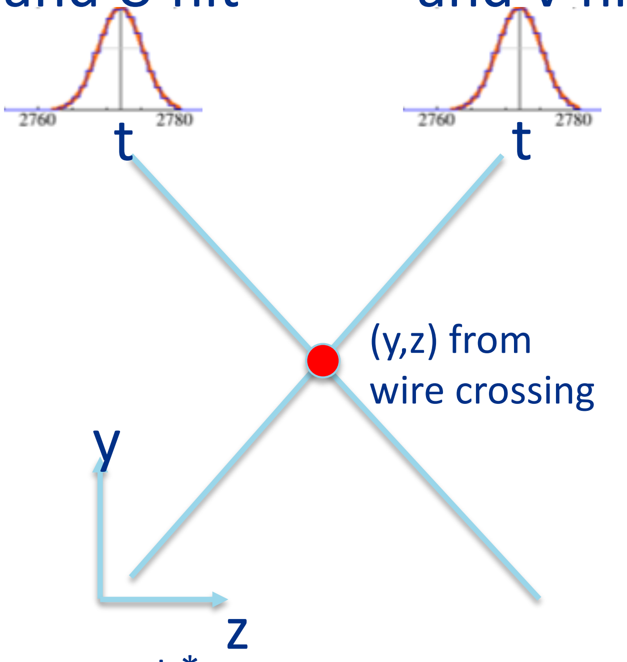
recob::SpacePoint



Run: 5145
Event: 26918
Beam momentum: 7GeV
10 Oct 2018 22:57:33 (GMT)

A U wire and U hit

A V wire and V hit

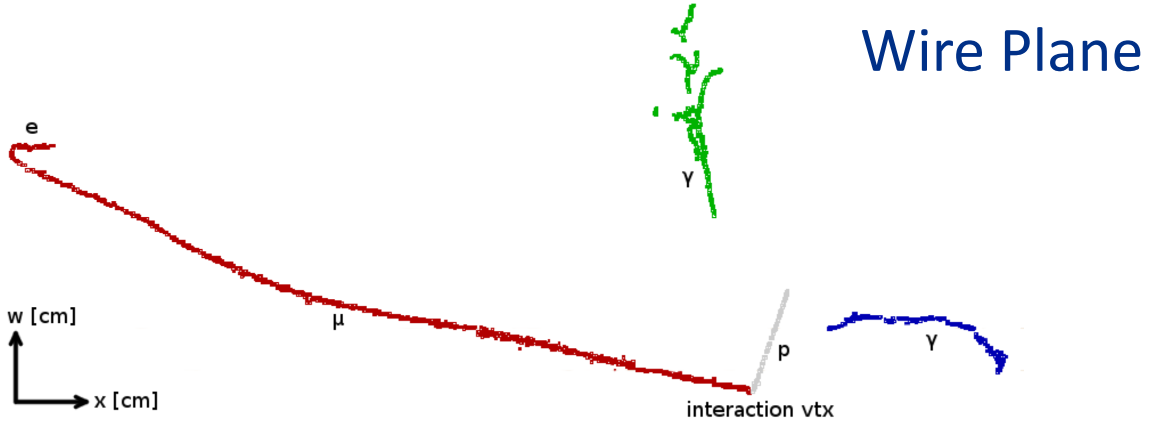


(y,z) from wire crossing

x = t * v
v = drift velocity
1.6 mm/us@500V/cm

- A object to save 3D points.
- It saves the x,y,z coordinates as well as charge information.
- Can be by-product of track fitting.
- SpacePointSolver, wire-cell and Pandora can make space points using hits.
  - Can have associations with hits on 3 planes – help disambiguation and cluster matching.
  - Provide input to 3D pattern recognition – Cluster3D.

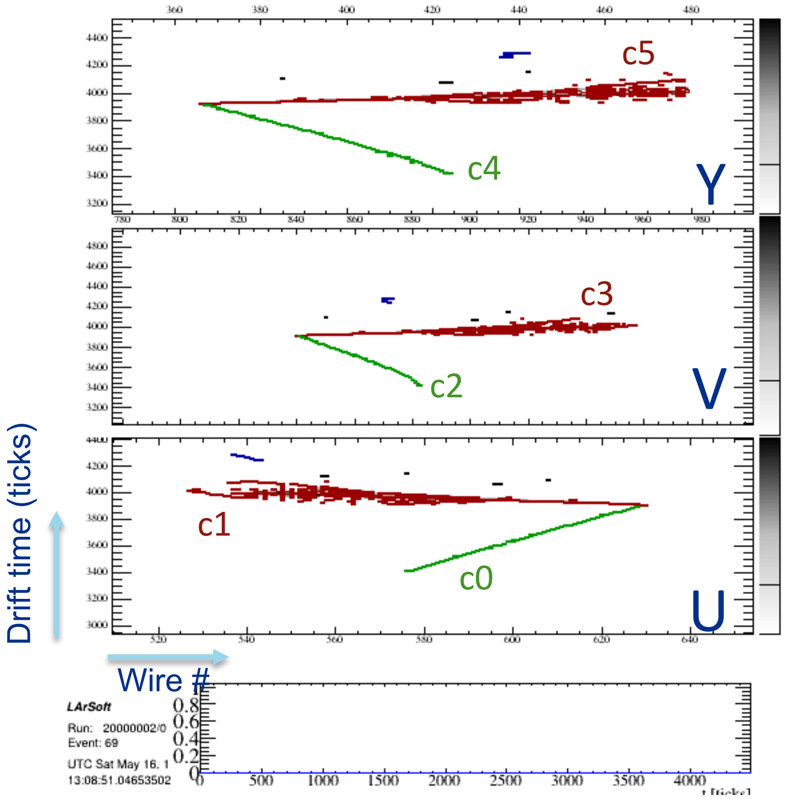
recob::Cluster



- A **recob::Cluster** is a collection of hits produced by the same particle.
- Spatial and charge information is used to cluster hits.
- Several pattern recognition algorithms produces recob::Clusters, the two main ones are Pandora and TrajCluster.

recob::PFParticle

- A **recob::PFParticle** is a collection of matched **recob::Clusters** on all planes
  - It is the main outcome of pattern recognition.
  - It is supposed to include all the hits produced by a single particle on all three planes.
  - Other useful information can be associated with a PFParticle
    - T0
    - Track/shower-like (through pdg)
    - Primary beam particle
    - Hierarchy information (one pfparticle can be the daughter of another)

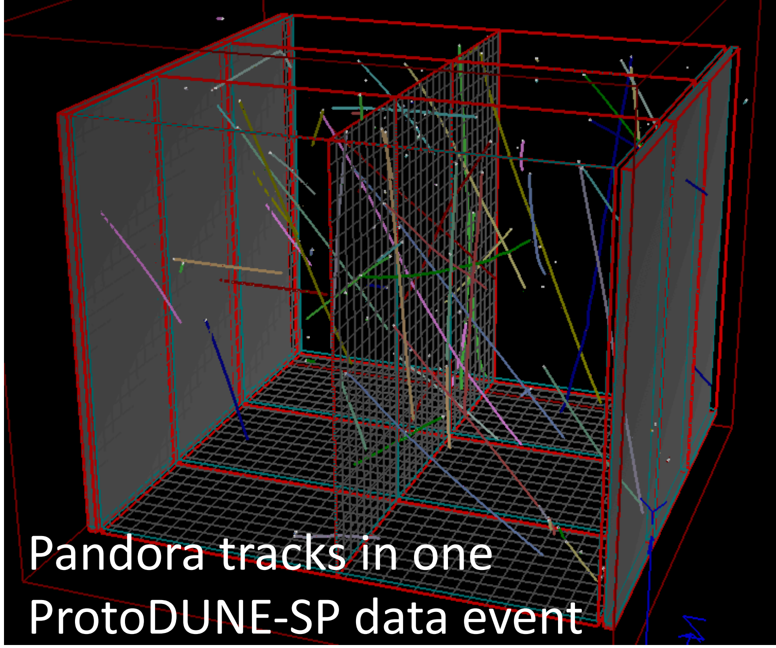


PFP0: c0, c2, c4 track-like

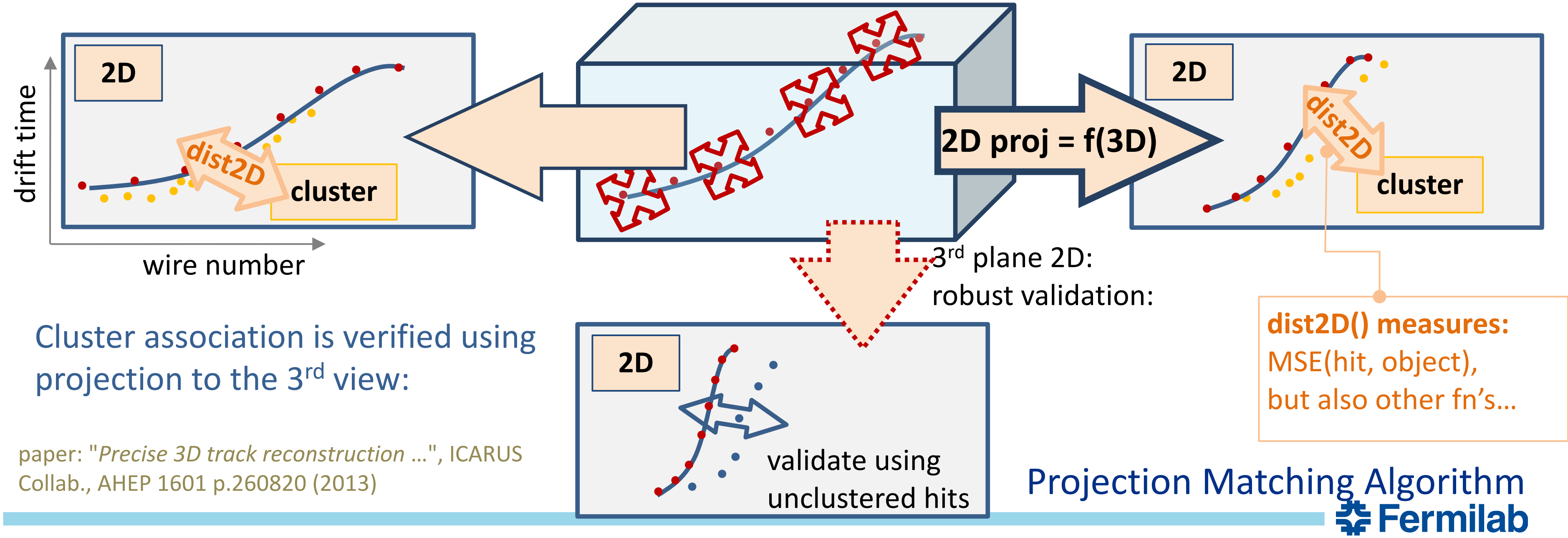
PFP1: c1, c3, c5 shower-like

recob::Track

- Tracks are fitted using track-like PFParticles as input.
  - Tracking may not use all hits
- Three main algorithms
  - Pandora track fitter, PMA, Kalman filter
- Output can include
  - Trajectory points (one trajectory point corresponds to one hit), directions, covariance matrix



Pandora tracks in one ProtoDUNE-SP data event



3rd plane 2D: robust validation:

dist2D() measures: MSE(hit, object), but also other fn's…

Cluster association is verified using projection to the 3rd view:

paper: "*Precise 3D track reconstruction …*", ICARUS Collab., AHEP 1601 p.260820 (2013)

Projection Matching Algorithm

anab::Calorimetry

- Each plane provides an independent calorimetric measurement.
  - 3 anab::Calorimetry objects associated with each recob::Track
- 3 vectors of quantities
  - Residual range – distance with regards to the track end.
  - dQdx – uncorrected dQ/dx values
    - dQ from hit charge
    - dx from track direction and wire pitch
  - dEdx – after correcting for attenuation, SCE, recombination
- Input to calorimetry-based particle ID

recob::Shower

- Reconstruct shower using shower-like PFParticle as input.
- Ideally one recob::Shower for one single electron or one single photon.
- It should provide both geometry and calorimetric information.
  - The Pandora shower maker provides direction and vertex information.
  - dE/dx information is being developed for e/gamma separation.

Other data products

- Photon detector
  - recob::OpHit - regions of the waveforms containing pulses.
  - recob::Flash – higher level object, built from nearby optical hits. Provides spatial, t0 and PE information. The goal is to have one flash for one physics object.
- Experiment specific data products (below are dune examples)
  - CRT
    - CRT::Trigger – module information
    - CRT::Hit – strip information on each module
  - beam::ProtoDUNEBeamEvent – beam information
  - raw::ctb::pdspctb – CTB trigger information

simb::MCTruth

- simb::MCTruth saves the output of any generator: neutrino interaction, nucleon decays, supernova neutrinos, etc.
- Origin: beam neutrino, cosmic interaction, supernova neutrino, single particle, unknown.
- Produce a list of simb::MCParticles before detector simulation.
- Neutrino interaction information saved in simb::MCNeutrino (CCNC, W, X, Y, Q^2, etc.)

simb::MCParticle

- In the Geant4 simulation, simb::MCParticles from all the generators will be copied first (with process name “Primary”) and then propagated through Geant4. Scattered or any new particles will be saved as new simb::MCParticles.
  - Two sets of simb::MCParticles, one from MCTruths, one from Geant4.
- simb::MCParticle saves the particle trajectory, momentum at each trajectory point, pdg, process name, mother/daughter information.
- The energy deposition and timing information are saved in sim::SimChannel.

Backtracker

- Backtracker connects hit information with true energy deposition.
- const std::vector<sim::TrackIDE> cheat::BackTrackerService::HitToTrackIDEs (recob::Hit const &hit)
  - sim::TrackIDE provides Geant4 MCParticle track ID and energy deposition through sim::SimChannel
  - Useful to evaluate efficiency and purity of reconstructed objects.
  - By default, shower daughter particles are not saved as MCParticles. But their energy deposition and parent particle's Geant4 ID (with a minus sign) are saved in sim::SimChannel and can be retrieved through backtracker.
- http://nusoft.fnal.gov/larsoft/doxsvn/html/classcheat_1_1BackTrackerService.html
- Another service particle inventory service connects MCParticle with MCTruth. It can also give MCParticle parent
  - http://nusoft.fnal.gov/larsoft/doxsvn/html/classcheat_1_1ParticleInventoryService.html

Using larsoft

- Set up experiment code
- Create your own module
- CMaleLists.txt
- Code examples
- https://cdcvs.fnal.gov/redmine/projects/larsoft/wiki/Using_art_in_LArSoft

Set up experiment code

- First source the experiment setup script
  - `source /cvmfs/dune.opensciencegrid.org/products/dune/setup_dune.sh`
- List all the available releases
  - `ups list -aK+ dunetpc`
- Setup the release you want to use
  - `setup dunetpc v08_22_00 -q e17:prof`
- You can run any *`lar`* commands now.
- You can also checkout experiment code (e.g. *dunetpc*) or larsoft repository (e.g. *larreco*) if you want to make changes.
- `ups active` shows the list of all active ups products.
- https://cdcvs.fnal.gov/redmine/projects/larsoft/wiki/Quick_Links

Create your own module

- **cetskelgen**
- https://cdcvs.fnal.gov/redmine/projects/cetlib/wiki/Cetskelgen
- `cetskelgen --help`
- art::EDProducer - This is a type of module that makes data products and stores them in the art::Event.
  - `cetskelgen producer myns::MyProducer`
- art::EDAnalyzer - This is a type of module that analyzes data products but cannot write them in an art::Event.
  - `cetskelgen analyzer myns::MyAnalyzer`
- art::EDFilter - The object allows the user to filter events based on information obtained about the event itself.
  - `cetskelgen filter myns::MyFilter`

CMakeLists.txt

- List all the libraries needed to build your code.
- You may copy CMakeLists.txt file from another directory and make changes.
- How to find missing libraries
  - For example, compiler gives the following error: undefined reference to `cheat::BackTrackerService::HitToTrackIDEs(art::Ptr<recob::Hit> const&) const’
  - setup the larutils UPS product (> v1_20_08)

  -
    ```
    find_global_symbol.sh -f -d
    "cheat::BackTrackerService::HitToTrackIDEs"
    Found in path
    /cvmfs/larsoft.opensciencegrid.org/products/larsim/v08_08_00/slf6.x86_64.e17.prof/lib/...
       Found in liblarsim_MCCheater_BackTrackerService_service.so
    ```
  - Add larsim_MCCheater_BackTrackerService_service to CMakeLists.txt
  - https://cdcvs.fnal.gov/redmine/projects/larsoft/wiki/Using_find_global_symbolsh_to_find_unresolved_symbols
- You need to specify both LIB_LIBRARIES and MODULE_LIBRARIES if you have module (*_module.cc) and non-module (other files) files in the same directory.

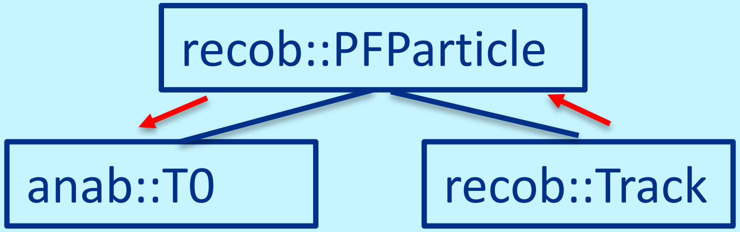
Plot all hit charge

```
//In class definition:
  TH1D *hist_hit_charge;

//In beginJob():
  art::ServiceHandle<art::TFileService> tfs;
  hist_hit_charge = tfs->make<TH1D>("hist_hit_charge","Hit
Charge",100,0,1000);

//In analyze(art::Event const & evt)
  auto hitListHandle = evt.getValidHandle<std::vector<recob::Hit>>("gaushit");
  auto const& hitlist = *hitListHandle;

  for (recob::Hit const& hit: hitlist) {
    hist_hit_charge->Fill(hit.Integral());
  }
```

- TFileService provides a mechanism for making TObject to be stored in the file and managing the memory for those objects.
  - https://cdcvs.fnal.gov/redmine/projects/larsoft/wiki/Using_art_in_LArSoft#artTFileService

Get anab::T0 from recob::Track

```
//Get Tracks
art::Handle< std::vector<recob::Track> > pandoratrkHandle;
std::vector< art::Ptr<recob::Track> > pandoratrks;
if (e.getByLabel("pandoraTrack", pandoratrkHandle))
  art::fill_ptr_vector(pandoratrks, pandoratrkHandle);
//Get PFParticles
art::Handle< std::vector<recob::PFParticle> > pfpListHandle;
e.getByLabel("pandora", pfpListHandle);
//Get PFParticle-Track association
art::FindManyP<recob::PFParticle> fmpfp(pandoratrkHandle, e, "pandoraTrack");
//Get T0-PFParticle association
art::FindManyP<anab::T0> fmt0pandora(pfpListHandle, e, "pandora");

for (size_t i = 0; i<pandoratrks.size(); ++i){
  auto & trk = pandoratrks[i];
  double t0 = 0;
  //Find PFParticle for track i
  //art::Ptr::key() gives the index in the vector
  auto &pfps = fmpfp.at(trk.key());
  if (!pfps.empty()){
    //Find T0 for PFParticle
    auto &t0s = fmt0pandora.at(pfps[0].key());
    if (!t0s.empty()){
      //Get T0
      t0 = t0s[0]->Time();
    }
  }
}
```


recob::PFParticle

anab::T0

recob::Track

https://cdcvs.fnal.gov/redmine/projects/larsoft/wiki/Using_art_in_LArSoft#artAssns

Find MCTruth for a recob::Track

```
  //BackTrackerService and ParticleInventoryService
  art::ServiceHandle<cheat::BackTrackerService>
bt_serv;
  art::ServiceHandle<cheat::ParticleInventoryService>
pi_serv;

  //Get tracks
  art::Handle< std::vector<recob::Track> >
trackListHandle;
  std::vector<art::Ptr<recob::Track> > tracklist;
  if
(evt.getByLabel(fTrackModuleLabel,trackListHandle))
    art::fill_ptr_vector(tracklist, trackListHandle);

  //Get hit-track association
  art::FindManyP<recob::Hit> fmth(trackListHandle,
evt, fTrackModuleLabel);

  //Loop over all tracks
  for(size_t i=0; i<tracklist.size();++i){
    if (fmth.isValid()){
      // Find true track for each reconstructed track
      int TrackID = 0;
      //Get all hits associated with the track
      std::vector< art::Ptr<recob::Hit> > allHits =
fmth.at(i);
      std::map<int,double> trkide;
```

```
 for(size_t h = 0; h < allHits.size(); ++h){
    art::Ptr<recob::Hit> hit = allHits[h];
    //TrackIDE saves the energy deposition for
each Geant particle ID
    std::vector<sim::TrackIDE> TrackIDs = bt_serv-
>HitToEveTrackIDEs(hit);
    for(size_t e = 0; e < TrackIDs.size(); ++e){
     trkide[TrackIDs[e].trackID] +=
TrackIDs[e].energy;
    }
   }
   // Work out which IDE despoited the most charge
in the hit if there was more than one.
   double maxe = -1;
   double tote = 0;
   for (std::map<int,double>::iterator ii =
trkide.begin(); ii!=trkide.end(); ++ii){
    tote += ii->second;
    if ((ii->second)>maxe){
     TrackID = ii->first; //TrackID
     maxe = ii->second;  //Energy
    }
   }
   // Now have trackID, so get PdG code.
   const simb::MCParticle *particle = pi_serv-
>TrackIdToParticle_P(TrackID);
   if (particle){
    std::cout<<"Pdgcode = "<<particle-
>PdgCode()<<std::endl;
   }
  }
 }
```

Gallery

- *gallery* provides lightweight access to event data in art/ROOT files outside the art event processing framework executable.
- It is not an alternative framework; rather, it provides a library that can be used to write programs that need to read (but not write) art/ROOT files.
- Information is available at: http://art.fnal.gov/gallery/
- Can easily access data products in an art file. Need some work to use service.

```
InputTag const track_tag("pandoraTrack");
for (gallery::Event ev(filenames); !ev.atEnd(); ev.next()) {
  for (size_t i = 0; i<tracks->size(); ++i){
    auto const&track = (*tracks)[i];
    std::cout<<"Track length = "<<track.Length()<<std::endl;
  }
}
```