Running the reconstruction

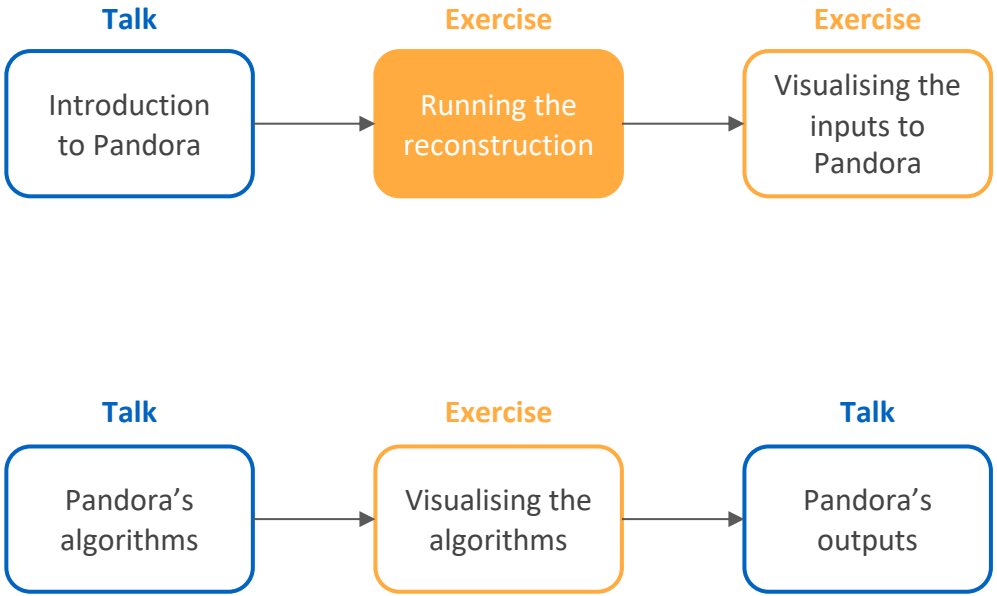
2nd November 2021

6th LArTPC Software Analysis Workshop - Edinburgh

Ryan Cross - for the Pandora team

Reconstruction session

Session structure



Credit to Lorena Escudero and Andrew Smith
These slides are heavily based on their slides from previous years!

Goals

- **This session scheduled for 1 hour**

- **Main goal 1** - Find and get to grips with the SBND reconstruction FHiCL files
  - Find the `standard_reco1_sbnd.fcl` and `standard_reco2_sbnd.fcl` configuration files
  - Look at the different reconstruction steps that we will run
  - Understand what each of them do

- **Main goal 2** - Run the reconstruction
  - Run the reconstruction on the files we simulated yesterday
  - This includes running Pandora
  - Dump out the new output products to confirm we produced what we wanted

Before we get started...

- Later today we'll be running the event display
- ssh on the machine you are working on and follow the steps from yesterday to get everything set up. To check, make sure MRB_TOP points to where you expect. Let us know if you have any issues here

```
$ echo $MRB_TOP # This should print the path to your development area
```

- In order for this to work, after you `ssh` into the machines you are working on make sure that you start a VNC session - this is required to do anything graphical

Main goal 1

Understanding the SBND reconstruction FHiCL files

SBND reconstruction FHiCL file

- Open `standard_reco1_sbnd.fcl`, `standard_reco2_sbnd.fcl`, and `reco_sbnd.fcl`.
  We'll use the first two to run the reconstruction, and they use sequences of paths defined in the last.

```
$ less $MRB_SOURCE/sbndcode/sbndcode/JobConfigurations/standard/standard_reco1_sbnd.fcl
$ less $MRB_SOURCE/sbndcode/sbndcode/JobConfigurations/standard/standard_reco2_sbnd.fcl
$ less $MRB_SOURCE/sbndcode/sbndcode/JobConfigurations/base/reco_sbnd.fcl
```

include

- Find the `trigger_paths: [...]`. **Q**: which producers are we going to run?
- **A**: It's the ones in the "`reco1`" and "`reco2`" paths:

```
reco1: [ rns
       , ophitpmt
       , ophitarapuca
       , opflashtpc0
       , opflashtpc1
       , caldata
       , gaushit
       , fasthit
       , gaushitTruthMatch
]
```

```
reco2: [ rns
       , pandora
       , pandoraTrack
       , pandoraShower
       , pandoraShowerSBN
       , pandoraCalo
       , pandoraPid
       , crthit
       , crttrack
       , crthitt0
       , crttrackt0
       , fmatch
       , caloskimCalorimetry
 ]
```

First time using less? Use the ↑ / ↓ arrow keys to navigate the file, and press q to quit

File looks empty? Make sure you setup your working area again, otherwise $MRB_SOURCE won't point to anywhere!

- In the last session, we were introduced to `pandora`, but there are many other steps in the reconstruction chain too!
- Next is a single-slide overview of these steps - not nearly enough to do them justice - but today we will mainly be focusing on `pandora`

SBND reconstruction chain on one slide

rns

"Random number saver" saves the state of art's random number generator

caldata, gaushit, ...

These modules process the wire signals (e.g. to remove noise), and then find peaks to make hits

linecluster

Runs a clustering algorithm on the hits, used later by PMA

pandora

Runs Pandora's pattern recognition itself.

Produces reconstructed particles from the input hits which are used by downstream modules and eventual physics analyses

Confusingly, some other modules contain the word "pandora" but this just refers to the fact that their inputs come from Pandora

pandoraShowerSBN

Advanced electromagnetic shower reconstruction

pandoraTrack, pandoraShower

Simple track and shower fitting modules using particles created by Pandora particle as input

crthit, ...

Reconstruction algorithms for the cosmic-ray tagger (CRT) detector subsystem

pandoraCalo, pandoraPid

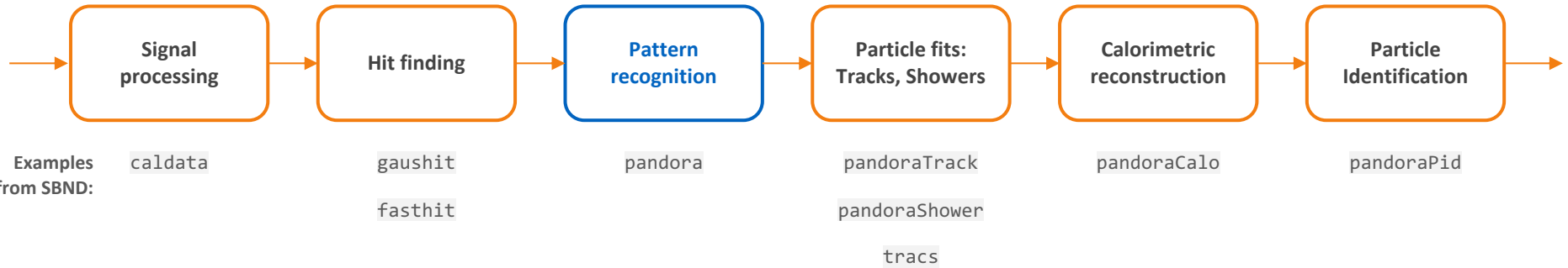
Reconstructs calorimetric information (dE/dx) from Pandora tracks, and runs particle identification (PID) to distinguish between reconstructed particle types

```
reco1: [ rns
       , ophitpmt
       , ophitarapuca
       , opflashtpc0
       , opflashtpc1
       , caldata
       , gaushit
       , fasthit
       , gaushitTruthMatch
]
```

```
reco2: [ rns
       , pandora
       , pandoraTrack
       , pandoraShower
       , pandoraShowerSBN
       , pandoraCalo
       , pandoraPid
       , crthit
       , crttrack
       , crthitt0
       , crttrackt0
       , fmatch
       , caloskimCalorimetry
 ]
```

A note on other experiments

- Remember here we are looking at the configuration for **SBND**
- Each experiment has its own unique needs, so expect to see some differences in the reconstruction chain if you work on MicroBooNE, ProtoDUNE, DUNE, etc.
- As far as Pandora is concerned, we can generalise the reconstruction chain to the following steps:



- Next, let's see how `pandora` is configured for SBND

Pandora's configuration

- `fhicl-dump` follows all `#includes` to get the bottom-line configuration. We can pipe (`|`) its output to `less` and search for one of the producers to learn more. `pandora` runs in the second stage of the reconstruction, so we can search for it by typing:

```
$ fhicl-dump standard_reco2_sbnd.fcl | less -p "pandora:"
```

`less`'s `-p` option allows us to jump straight to the part of the file we are interested in

We searched for this

```
pandora: {
                    ConfigFile:
"PandoraSettings_Master_SBND.xml"
          EnableLineGaps: true
          EnableMCParticles: false
          EnableProduction: true
          GeantModuleLabel: "largeant"
          HitFinderModuleLabel: "gaushit"
          PrintOverallRecoStatus: false
          ShouldPerformSliceId: true
          ShouldRunAllHitsCosmicReco: true
          ShouldRunCosmicHitRemoval: true
          ShouldRunCosmicRecoOption: true
          ShouldRunNeutrinoRecoOption: true
          ShouldRunSlicing: true
          ShouldRunStitching: true
          UseGlobalCoordinates: true
          UseHitWidths: true
          module_type: "StandardPandora"
      }
```

The settings file that contains the list of algorithms that Pandora will run

The producer module that created the hits that we are going to feed into Pandora

The steering parameters that tell Pandora which of it's high level reconstruction steps it should execute

The type of the LArSoft module to use

Configuring Pandora steps

(For reference! Don't try this now…)

- Pandora's full reconstruction chain is designed to handle neutrino interactions in dense cosmic environments. As you will hear later, there are two main algorithm chains optimised for **cosmic rays**, and **neutrinos** respectively
- For SBND, an experiment that will have neutrinos and cosmics - we normally want to run all of the steps
- For cosmic events, we only need to run the **cosmic** algorithm chain. For neutrino events, we only need to run the **neutrino** algorithm chain. We can configure Pandora to run one, many or all of the steps in its full reconstruction chain by modifying the FHiCL steering parameters
- Make a new directory to work in for this session, and add a new FHiCL file with the following lines, then save and close the file:

```
$ mkdir -p $MRB_TOP/reco/config
$ cd $MRB_TOP/reco/config          # Put your new .fcl file here
$ vim my_reco_sbnd_basic.fcl
```

Please use your favourite text editor, here we use `vim`. To save in vim type `Esc, :w, Return ↵`
To close it's `Esc, :q, Return ↵`
To save and close in the same command, it's `Esc, :wq, Return ↵`

```
#include "standard_reco2_sbnd.fcl"

physics.producers.pandora.ShouldRunAllHitsCosmicReco: true
physics.producers.pandora.ShouldRunStitching: true
physics.producers.pandora.ShouldRunCosmicHitRemoval: false
physics.producers.pandora.ShouldRunSlicing: false
physics.producers.pandora.ShouldRunCosmicRecoOption: false
physics.producers.pandora.ShouldRunNeutrinoRecoOption: true
physics.producers.pandora.ShouldPerformSliceId: false
```

Include the standard configuration

Example of modified configuration

Pointing to a new configuration (For reference! Don't try this now…)

- We want to make sure that LArSoft will know where to look for our new FHiCL file, to do this we add it to the `FHICL_FILE_PATH` environment variable. Start by printing it to the terminal:

```
$ echo $FHICL_FILE_PATH
```

- You will see many many directories, all separated by a '`:`'. To add our `reco/config` folder to this list run the following command:

```
$ export FHICL_FILE_PATH=$MRB_TOP/reco/config:$FHICL_FILE_PATH
```

- Echo the `FHICL_FILE_PATH` again to check that everything worked (it should be the first in the list)
- Now run `fhicl-dump` again to make sure our new configuration file is set up as we want

```
$ fhicl-dump my_reco_sbnd_basic.fcl | less -p "pandora:"
```

Main goal 2

Running the reconstruction

Running the reconstruction

- We are now poised to run the reconstruction! Make a directory to work in, and run it:

```
$ mkdir -p $MRB_TOP/reco/work
$ cd $MRB_TOP/reco/work
$ lar -c standard_reco1_sbnd.fcl -n -1 -s /path/to/my/detsim/file.root -o reco1_events.root
$ lar -c standard_reco2_sbnd.fcl -n -1 -s reco1_events.root -o reco2_events.root
Can also run on pre-made gen+g4+detsim files in /home/LArsoft/shared/files/simulation
prodsingle_1mu1p_10events_g4_detsim.root , prodgenie_nue_AV_10events_g4_detsim.root
```

The `-n -1` option means run over all events in the input file

```
Full event                                                    30.33       36.0671     41.6728     36.4245     3.52474     10
-------------------------------------------------------------------------------------------------------------------------------
source:RootInput(read)                                        0.000763427 0.00177477  0.00313986  0.00184294  0.00080113  10
fullreco:rns:RandomNumberSaver                                4.2575e-05  8.41849e-05 0.000372039 5.36485e-05 9.62063e-05 10
fullreco:ophitpmt:SBNDOpHitFinder                             0.449261    0.622272    1.03332     0.501103    0.206869    10
fullreco:ophitarapuca:SBNDOpHitFinder                         4.21828     5.12789     7.57794     4.49848     1.2325      10
fullreco:opflashtpc0:SBNDFlashFinder                          0.0337186   0.0465044   0.0592497   0.0490617   0.00720987  10
fullreco:opflashtpc1:SBNDFlashFinder                          0.0153911   0.017092    0.019471    0.0170536   0.00111286  10
fullreco:caldata:CalWireSBND                                  4.37116     4.43222     4.51511     4.4182      0.0504974   10
fullreco:gaushit:GausHitFinder                                0.253137    0.404216    0.700568    0.370396    0.111067    10
fullreco:fasthit:RawHitFinder                                 0.307307    0.312703    0.320456    0.31152     0.004768    10
fullreco:linecluster:LineCluster                              0.00983846  0.016409    0.0216492   0.0164625   0.00391641  10
fullreco:pandora:StandardPandora                              0.260475    0.457744    0.774284    0.390011    0.1511      10
fullreco:pandoraTrack:LArPandoraTrackCreation                 0.0084431   0.0110021   0.0141811   0.0107761   0.0014914   10
fullreco:pandoraShower:LArPandoraModularShowerCreation        0.00101441  0.00149197  0.00336328  0.00132518  0.000632474 10
fullreco:pandoraShowerSBN:LArPandoraModularShowerCreation     0.00100304  0.00122369  0.00138096  0.00125789  0.000114251 10
fullreco:pandoraShowerLegacy:LArPandoraShowerCreation         0.000610985 0.000736134 0.000979938 0.000737428 9.38111e-05 10
fullreco:pandoraCalo:Calorimetry                              0.00770421  0.013151    0.0182823   0.0132139   0.00243814  10
fullreco:pandoraPid:Chi2ParticleID                            0.000236185 0.000366934 0.0010507   0.000289568 0.000230409 10
fullreco:pmalgtrackmaker:PMAlgTrackMaker                      0.632458    1.70203     4.64527     1.25026     1.10736     10
fullreco:pmatrackcalo:Calorimetry                             0.00770214  0.0137559   0.0158156   0.0139574   0.00212271  10
fullreco:pmatrackpid:Chi2ParticleID                           0.000248236 0.000315716 0.000472124 0.000295973 6.66861e-05 10
fullreco:emshower:EMShower                                    0.00349526  0.00520842  0.00653785  0.00516831  0.000774087 10
fullreco:crthit:sbndcode/CRT/CRTSimHitProducer                0.00021374  0.000315413 0.00103368  0.000241245 0.000239807 10
fullreco:crttrack:sbndcode/CRT/CRTTrackProducer               9.5663e-05  0.000133222 0.000409461 0.000104356 9.21743e-05 10
fullreco:crthitt0:sbndcode/CRT/CRTTools/CRTT0Matching         0.000911981 0.00114962  0.00152289  0.00105388  0.000209767 10
fullreco:crttrackt0:sbndcode/CRT/CRTTools/CRTTrackMatching    0.000839373 0.00104623  0.00138288  0.000975999 0.000161761 10
fullreco:fmatch:FlashPredict                                  0.00485561  0.00700144  0.00831595  0.0070559   0.00101539  10
fullreco:opt0finder:SBNDOpT0Finder                            17.033      21.9169     25.6382     23.8542     3.07717     10
[art]:TriggerResults:TriggerResultInserter                    2.6827e-05  4.04393e-05 8.9282e-05  3.5893e-05  1.70867e-05 10
end_path:out1:RootOutput                                      5.751e-06   1.01145e-05 2.4017e-05  8.646e-06   5.07523e-06 10
end_path:out1:RootOutput(write)                               0.925388    0.949373    0.969371    0.949746    0.0134437   10
===============================================================================================================================
```

This step can take some time, so please be patient!

We can check to see that everything we expected has been executed, and see how long each took

So… what's new?

- Run `eventdump.fcl` to see all of the new collections we just made

```
$ lar -c eventdump.fcl -s reco2_events.root -n 1
```

```
PROCESS NAME | MODULE LABEL..... | PRODUCT INSTANCE NAME | DATA PRODUCT TYPE..................................................... | .SIZE
SinglesGen.. | generator........ | ..................... | std::vector<simb::MCTruth>............................. | .1
SinglesGen.. | rns.............. | ..................... | std::vector<art::RNGsnapshot>.......................... | .1
SinglesGen.. | TriggerResults... | ..................... | art::TriggerResults.....
G4.......... | largeant......... | LArG4DetectorServicevolFieldCage... | std::vector<sim::SimEnergyDeposit>.....
G4.......... | rns.............. | ..................... | std::vector<art::RNGsnapshot>.........
G4.......... | pdfastsim........ | Reflected............ | std::vector<sim::SimPhotonsLite>.......
G4.......... | TriggerResults... | ..................... | art::TriggerResults.................................... | .1
DetSim...... | opdaq............ | ..................... | std::vector<raw::OpDetWaveform>........................ | .?
DetSim...... | rns.............. | ..................... | std::vector<art::RNGsnapshot>.......................... | .1
DetSim...... | daq.............. | ..................... | std::vector<raw::RawDigit>............................. | .?
DetSim...... | TriggerResults... | ..................... | art::TriggerResults.................................... | .1
Reco1....... | caldata.......... | ..................... | std::vector<recob::Wire>............................... | 11224
Reco1....... | opflashtpc0...... | ..................... | std::vector<recob::OpFlash>............................ | ....1
Reco1....... | opflashtpc0...... | ..................... | art::Assns<recob::OpHit,recob::OpFlash,void>........... | .3252
Reco1....... | gaushit.......... | ..................... | art::Assns<recob::Wire,recob::Hit,void>
Reco2....... | pandora.......... | ..................... | std::vector<recob::PFParticle>.........
Reco2....... | pandoraTrack..... | ..................... | std::vector<recob::Track>..............
Reco2....... | crthit........... | ..................... | art::Assns<sbn::crt::CRTHit,sbnd::crt::C
Reco2....... | pandora.......... | ..................... | art::Assns<recob::PFParticle,larpandoraobj::PFParticleMet.12
Reco2....... | pandoraShowerSBN... | ..................... | art::Assns<recob::Shower,recob::PFParticle,void>.....4
```

These are the existing data products from previous steps

These are the new data products that we have just produced

Got spare time?

Try starting the next tutorial - running the event display

Pandora development team

Pandora is an open project and new contributors would be extremely welcome.
We'd love to hear from you and we will always try to answer your questions.

Pandora Liaisons

Pandora lead	John Marshall	john.marshall@warwick.ac.uk
	Andy Blake	a.blake@lancaster.ac.uk
DUNEFD single phase	Dom Brailsford	d.brailsford@lancaster.ac.uk
	Andy Chappell	andrew.chappell@warwick.ac.uk
ProtoDUNE single phase	Leigh Whitehead	leigh.howard.whitehead@cern.ch
	Steve Dennis	sdennis@hep.phy.cam.ac.uk
ProtoDUNE dual phase & vertical drift	Dom Brailsford	d.brailsford@lancaster.ac.uk
	Maria Brigida Brunetti	Maria.Brunetti@warwick.ac.uk
DUNE NDLAr	Melissa Uchida	mauchida@hep.phy.cam.ac.uk
	Alex Moor	afm67@cam.ac.uk
	John Back	j.j.back@warwick.ac.uk
MicroBooNE	Alex Moor	afm67@cam.ac.uk
SBND	Dom Brailsford	d.brailsford@lancaster.ac.uk
	Henry Lay	h.lay@lancaster.ac.uk
	Ed Tyley	e.tyley@sheffield.ac.uk
ICARUS	Bruce Howard	bruhoward@indiana.edu

Graduate students

Ryan Cross
Henry Lay
Isobel Mawby
Alex Moor
Mousam Rai
Natsumi Taniuchi
Ed Tyley
Karolina Wresilo

github.com/PandoraPFA

PandoraPFA.slack.com