UNIVERSITY OF SUSSEX

DEEP UNDERGROUND NEUTRINO EXPERIMENT

DAQ Simulations

Pierre Lasorak

Outline

- Aim and scope of DAQ simulations
- Simulation chain:
  - Geometry
  - Physics
  - Background
  - Particle propagation (GEANT4)
  - Detector simulation (electronic response etc.)
  - Online hit finder
- Some of the references:
  - https://docs.google.com/document/d/1Av1mVvrzqvSCyZvDsk79aAc0AounFvFL_3r7vZEer8Y/edit

Disclaimer:
Simulation is a big effort, and it's very hard to understand everything. I'm putting here the stuff that I think would have helped me at the beginning (and which I could find information about).

A lot of content is from Diego Garcia-Gomez and Andrzej Szelc, thanks to them!!

Aim

- The aim is to help designing the DAQ and to be able to quantify its impact on the physics we are trying to extract. For example:
  - I change the noise on the wire, how does the trigger react?
  - Does my trigger cope with this many neutrons entering my detector?
  - How much data do I need to store if I want to do solar neutrinos?
- DUNE is a very big experiment in terms of data:

Parameter	single-phase	dual-phase
TPC unit	APA	CRO crate
Unit multiplicity	150	240
Channels per unit	2560 (800 collection)	640 (all collection)
ADC sampling	2 MHz	2.5 MHz
ADC resolution	12 bit	12 bit

  - Rate = 150 x 2560 x 2MHz x 12bit > 1 TB / s / 10kT module!!
  - Requirement:
    - Write on disk at 100GB/s
    - Total data / year < 30 PB
  - How to get such reduction of data?
  - What is the impact of this reduction of data on the physics?

Scope

- We are fairly confident we can trigger on beam events:
  - Big events that are easily identifiable.
  - Represent a “small” proportion of the total data.
- Challenge is to get the SN / lowE events
  - We don’t want to miss one!! → Efficient trigger
  - Close to the radiological backgrounds in energy. → Good background rejection
  - Requires to read the DUNE detector for 30 sec without loss. → Capacity to record a lot of data
- That’s why a lot of the DAQ stuff is about SN.
- Now going to describe the simulation chain for SN (but that could be changed to beam event / cosmics)

Simulation chain TPC

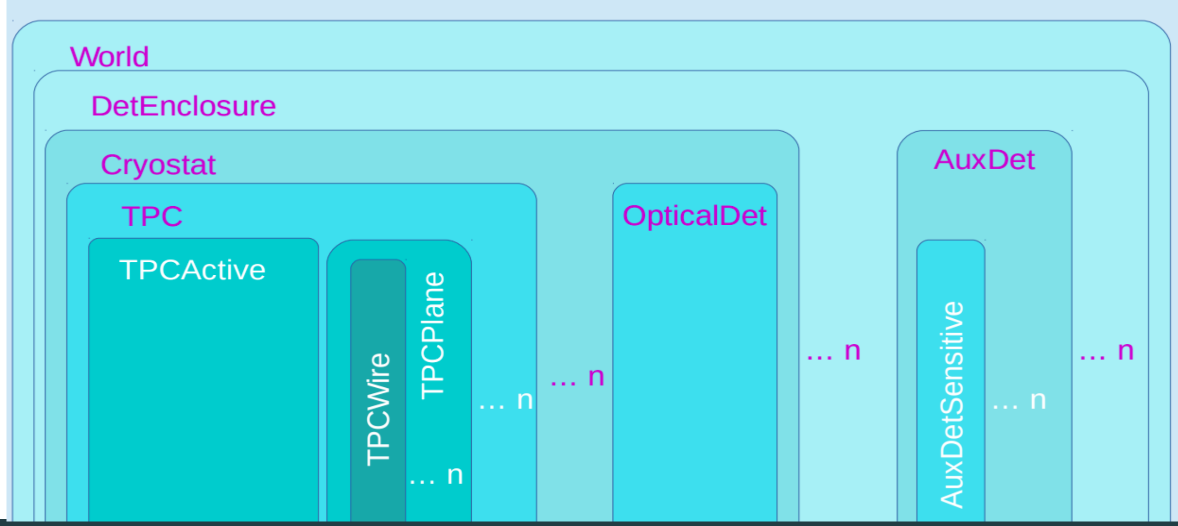
What happens in reality:	What happens in LArSoft:
• SN explodes • A large amount of νs arrive DUNE • A few of them interact in the detector • Mostly through $\nu_e + Ar \rightarrow e^- + K^*$	• MARLEY
• Some radiological background happen	• Radiological generator
• The particles move in the detectors and create the drift electrons + scintillation light • These electrons are drifted and reach the wires	• GEANT4
• The wires record the noise + whatever fluctuation the electrons created	• DetSim
• Some algorithm is ran on a very fast electronic to recognise the electron pulses on top of the noise	• Trigger primitive algorithm
• Another algorithm uses these electron pulses and decides whether to issue a trigger	• Clustering and burst trigger (next talk)

Simulation chain PDS

What happens in reality:	What happens in LArSoft:
• SN explodes • A large amount of νs arrive DUNE • A few of them interact in the detector • Mostly through $\nu_e + Ar \rightarrow e^- + K^*$	• MARLEY
• Some radiological background happen	• Radiological generator
• The particles move in the detectors and create the drift electrons + scintillation light	• GEANT4
• The photons reach the PDS	• SimPhotonCounter
• Some algorithm is ran on a very fast electronic to recognise the photo-electron pulses on top of the noise	• OpDetDigitizer
• Another algorithm uses the PDS pulse info and decide to issue a trigger.	• Clustering and burst trigger (next talk)

Geometries

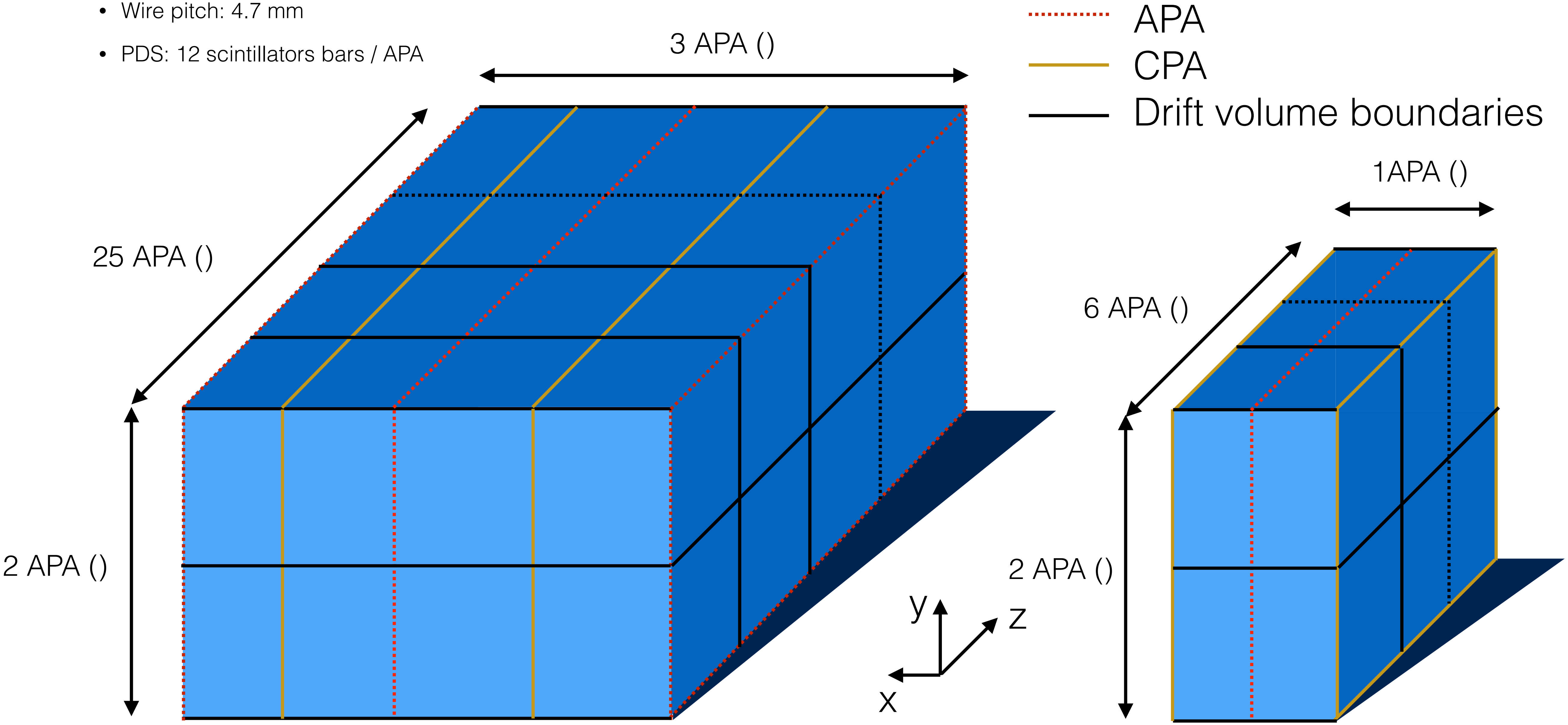
- Geometry description for each LAr detector (μBooNE, LArIAT, ICARUS, DUNE, etc.)
- Reconstruction doesn't depend on the geometry.
- Uses GDML (Geometry Description Markup Language).
- For the simulation 2 geometries:
  - One with TPC wires → determine the place of the wire etc.
  - One without wire (_nowires) → what is used in GEANT4 for generating particle (save time and RAM).
- Hierarchical organisation of the volumes
- Coordinate systems
  - Z direction: beam (same direction)
  - Y direction: height (positive is higher)
  - X direction: width (positive direction to make a direct coordinate system).



Geometries

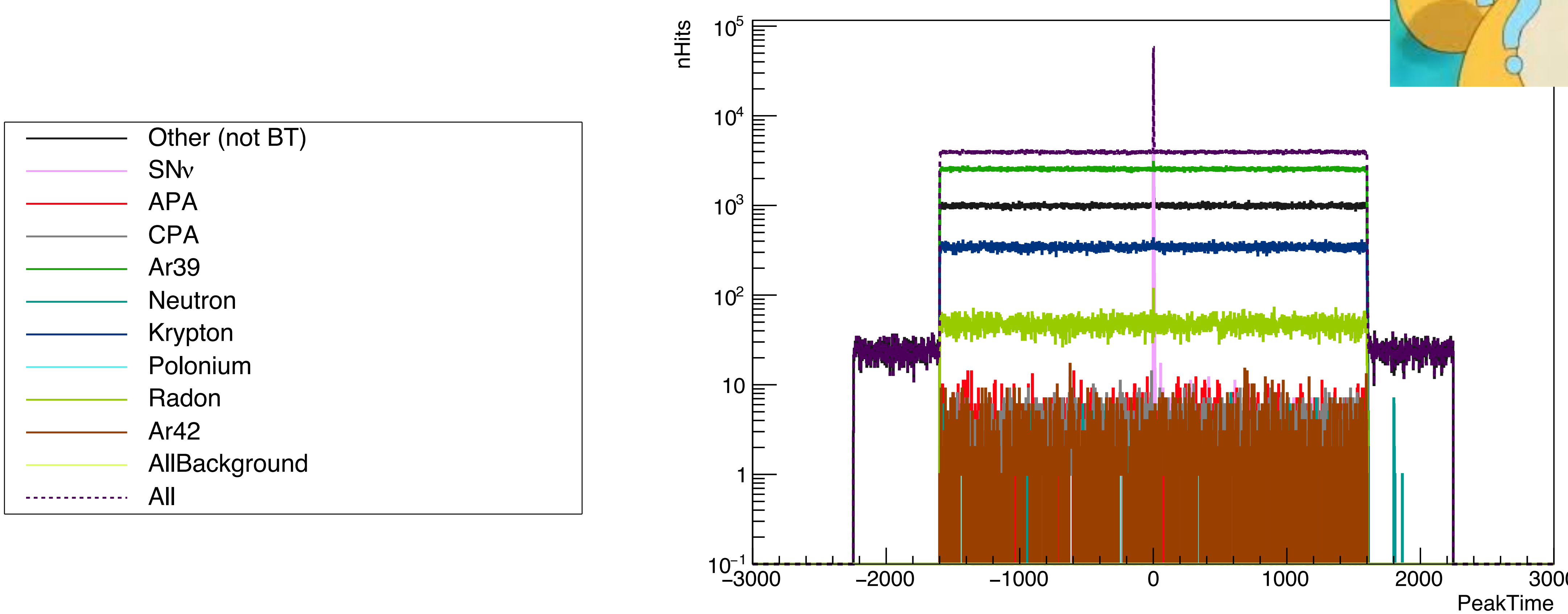
Volume ratio: 0.12
Area ratio: 0.12
Wire ratio: 0.125

- What a FD will look like:
  - 10kT = 150 APA = 2 APA high x 3 APA wide x 25 APA long
  - Wire pitch: 4.7 mm
  - PDS: 12 scintillators bars / APA
- What we are using in the simulation (MCC10, MCC11, and probably upcoming productions too):
  - “1 x 2 x 6” = 2 APA high x 1 APA wide x 6 APA long
  - Wire pitch: 4.7 mm
  - PDS: 12 scintillators bars / APA



Before we go any further

- An event in LArSoft is a full drift window: 2.244ms
  - Photon hits timing → goes negative!!
  - That's because it takes some time for the electron to arrive to the APA → you need to generate events before to take into account the number of electrons from the previous events!! (don't let this trick you)



Event generation

- First step of any particle physics experiment always involves a generator that have an assumption of the cross section, the luminosity (or flux).
- Wide variety of generators:
  - For beam/atmospheric events: GENIE, NuWro
  - For SN neutrinos: Marley (we will discuss more about this one this afternoon)
  - For cosmic events: CRY
  - Nucleon decay: NDk
  - Particle gun: Single, TextFileGen
  - Radiological backgrounds
  - … and all the ones that I forgot

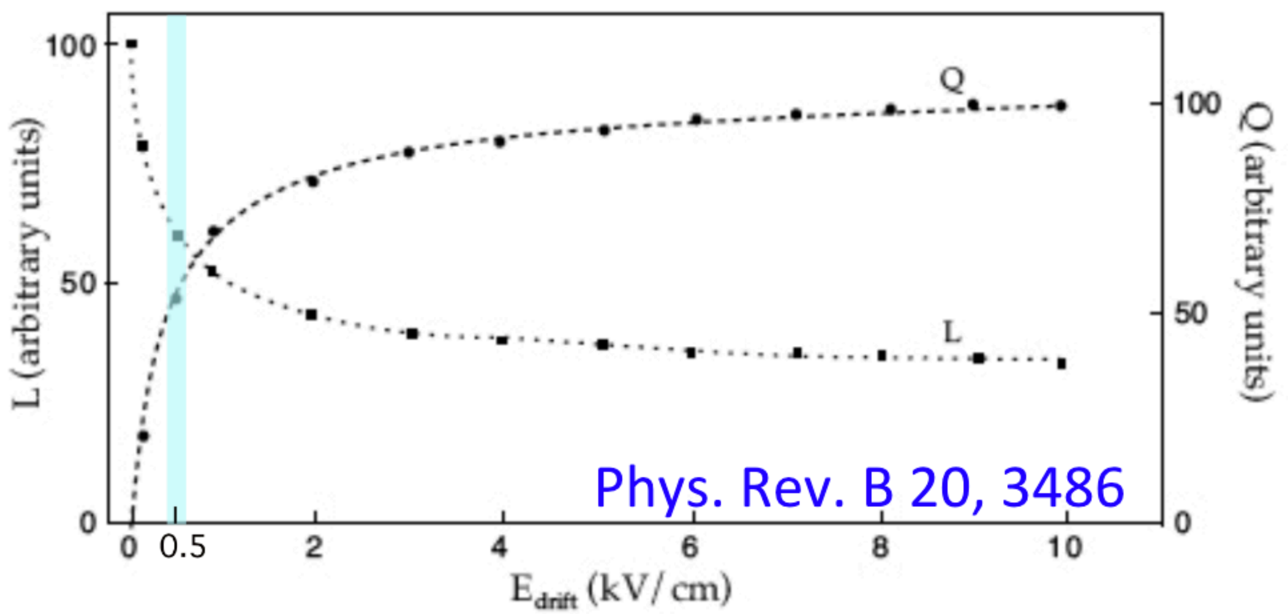
All these live in `LArSim/EventGenerator`

- Relies on very different physics:
  - Nuclear physics + beam exposure → GENIE
  - SN core + Nuclear physics → MARLEY
  - SUSY + Nuclear physics → NDk (right?)

- All of these we know are more or less **wrong**, and they have systematics included to the modelling etc.
  - In DAQ, **we basically don't care**, we want to trigger, have a data flow etc. As long as the estimate resembles roughly what they will be in the detector. We also want DAQ to be **stable against the systematic errors** there are in these generators.
  - … as long as it's deemed to be good enough, of course.
  - … and as long as you are not doing Solar (which depends heavily on the background rate).

(LAr)GEANT4

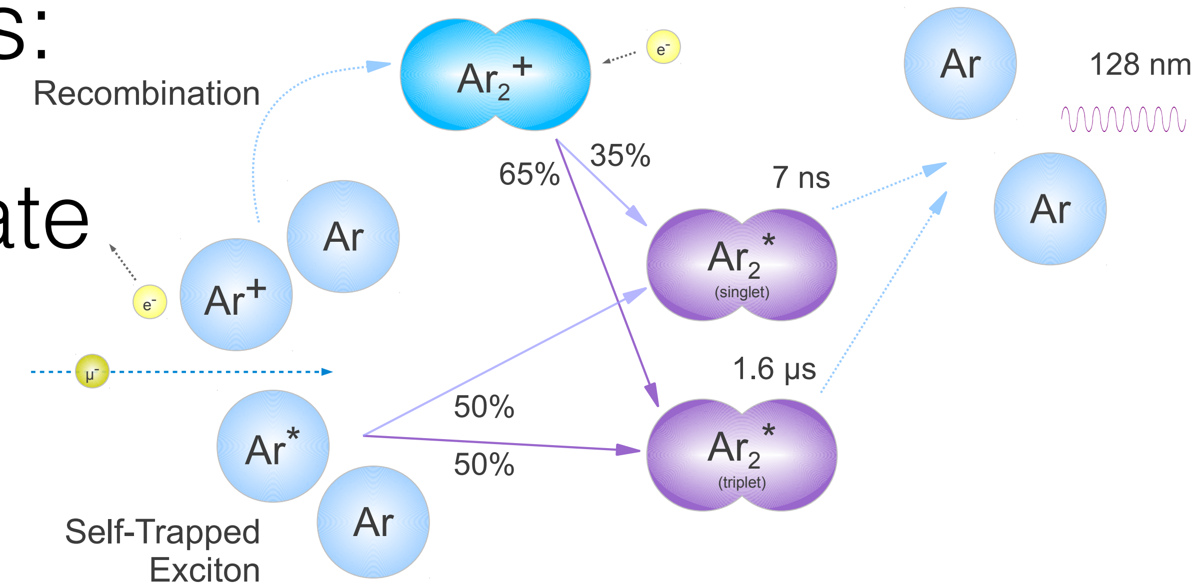
- Detector simulation:
  - GEANT → developed for all particles physics experiments
  - LArGeant → wrapper of GEANT4 in LArSoft.
- Propagates the particles in the detector
  - Divide all detector geometry into voxels
  - Step the particles, one after the other
  - If a particle looses energy → create drift electrons and scintillation photons



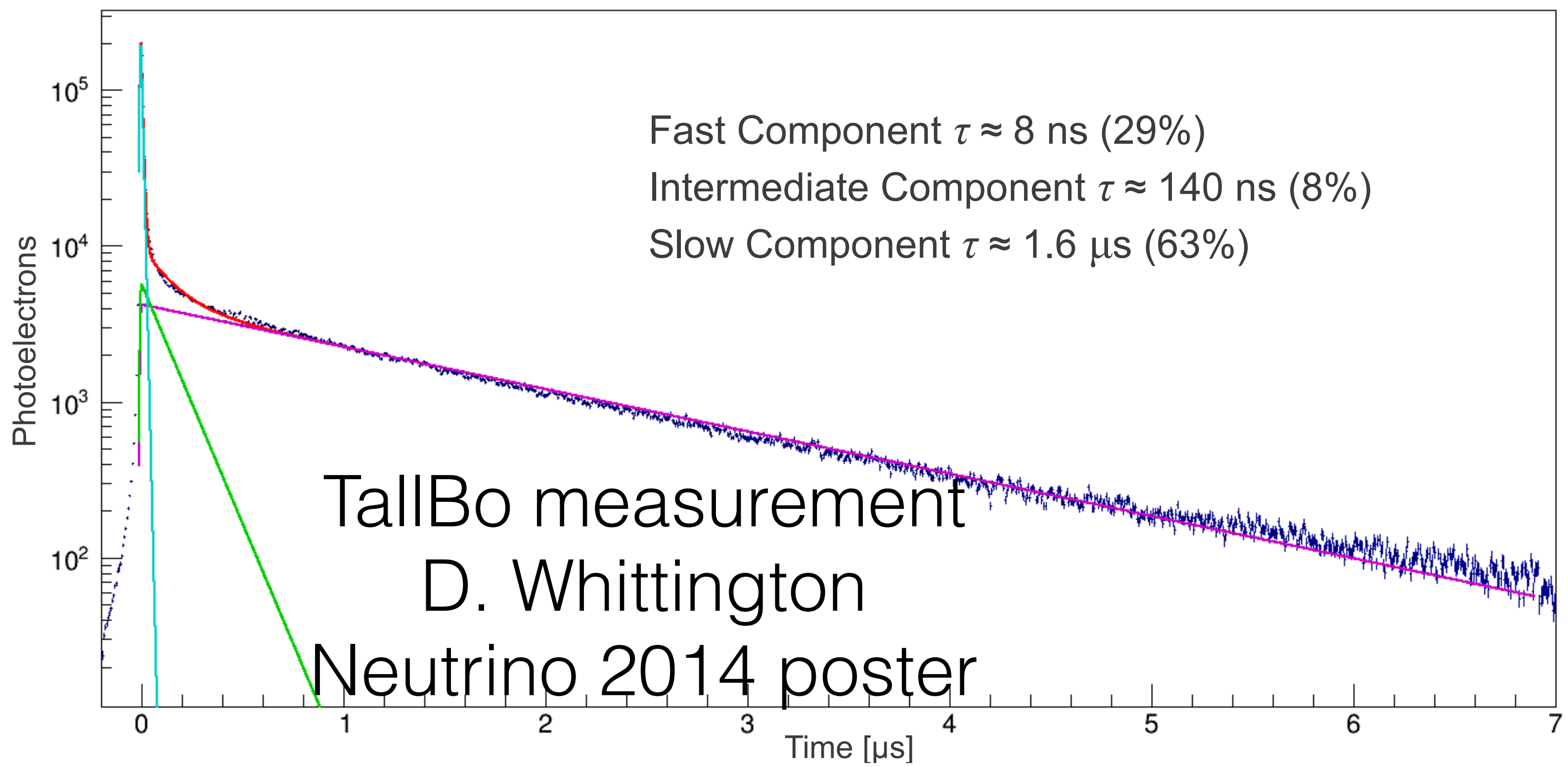
dEdX → $n_{\text{electrons to drift}} + n_{\text{scintillation photons}}$

Scintillation light in LArTPC

- LAr scintillation, 2 components:
  - Fast light from singlet Ar2* state
  - Slow light from triplet state



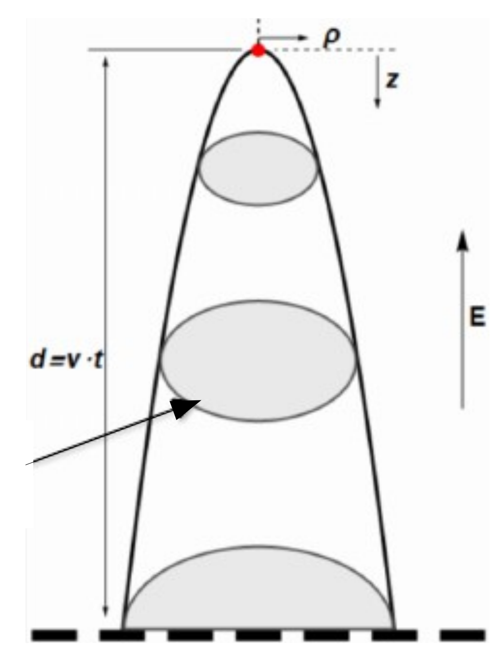
D. Whittington
Neutrino 2014 poster



Effect of HV, impurities can change the timing of the different components.

LArGeant

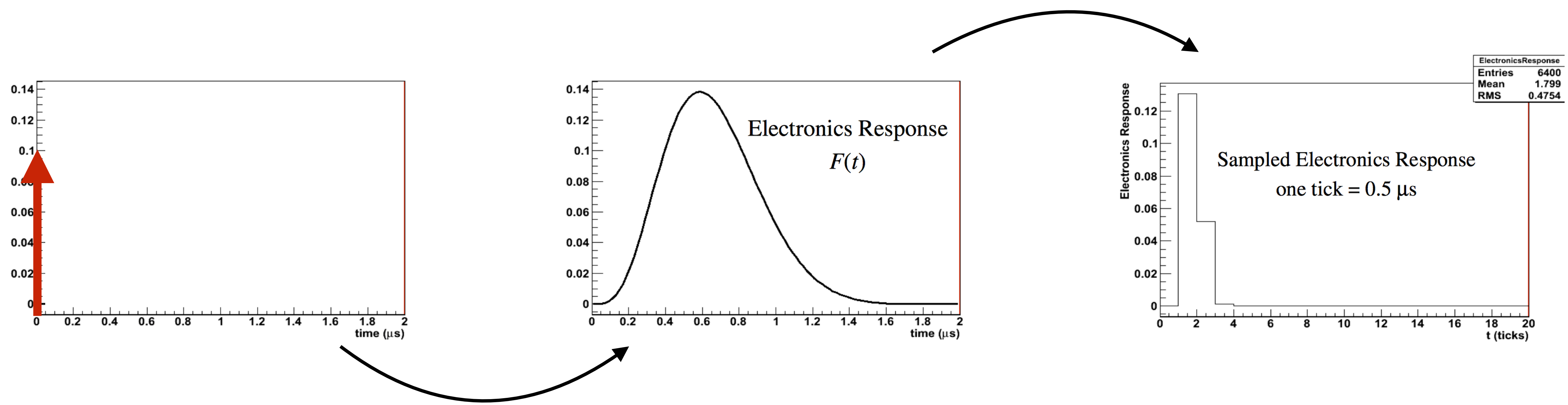
- Creating the drift electrons:
  - Takes too long to keep track of all the electrons generated.
  - Want to get rid of geometric effects from the Geant4 voxels.
- Apply probabilities to get the number of electrons on the wire.
  - Clusters of 600 electrons:
    - That's because a MIP perpendicular to the wire creates around ~30 clusters).
  - DEdx → $n_{electrons}$ + electron life time + recombination + impurities + diffusion → $n_{electrons}$ on the wire.
  - Smear applied "manually"
    - One cluster at the cathode gets a smear of 1.8mm in the longitudinal direction and 2.5mm in transverse direction.
  - For example, diffusion:
    - $\sigma_{L/T} = (2\, D_{L/T}\, X_{drift} / v_x)^{1/2}$



DetSim

- Aim: getting from the electron reaching the wire → waveform sampled at 2 MHz.
    - Simulate the electronic response to a pulse.
        - Usually from a test bench reproducing the hardware in the experiment (doing this at the moment)

Sample the response



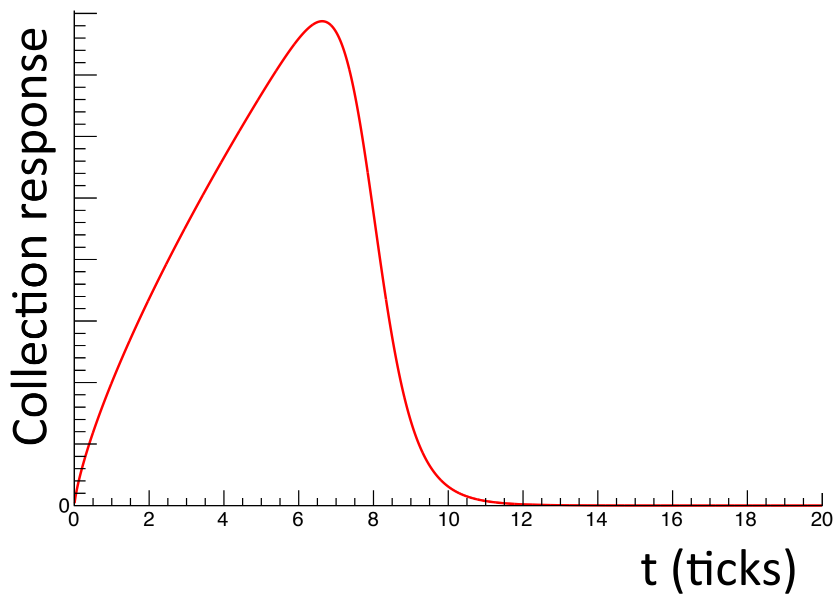
Feed in a Dirac function to the electronics

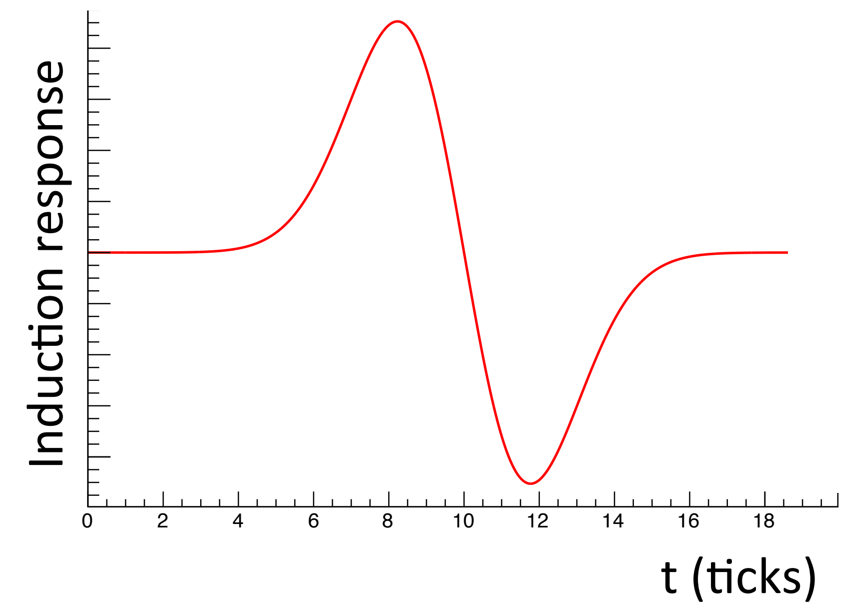
μBooNE pulse response

- Stuff that are relevant for this:
    - Gain and "Shaping"

DetSim

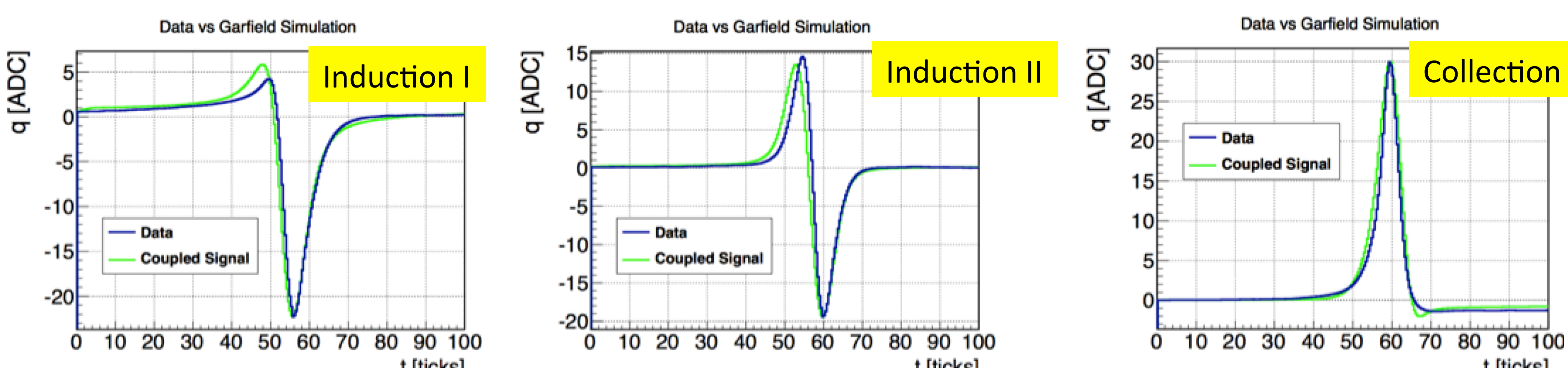
- What do the responses look like for a typical bunch of electrons?





- Very different response!

- Can compare Data/MC, using Garfield simulations

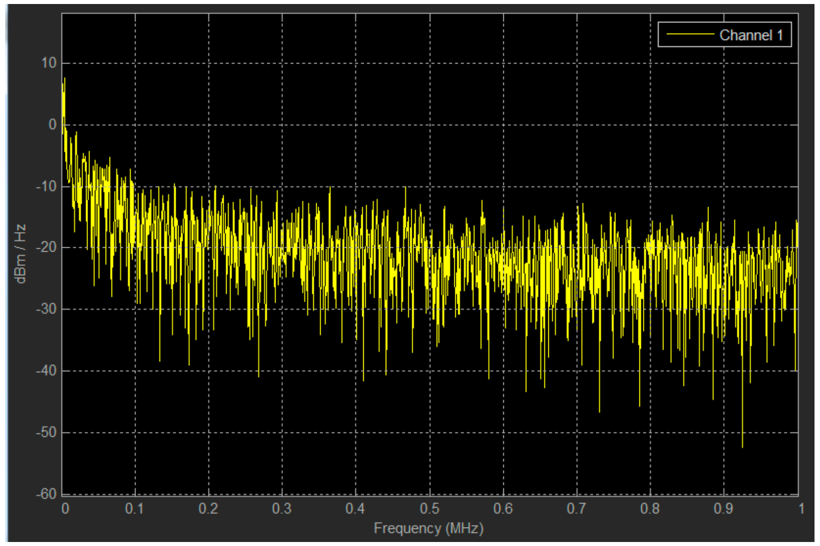


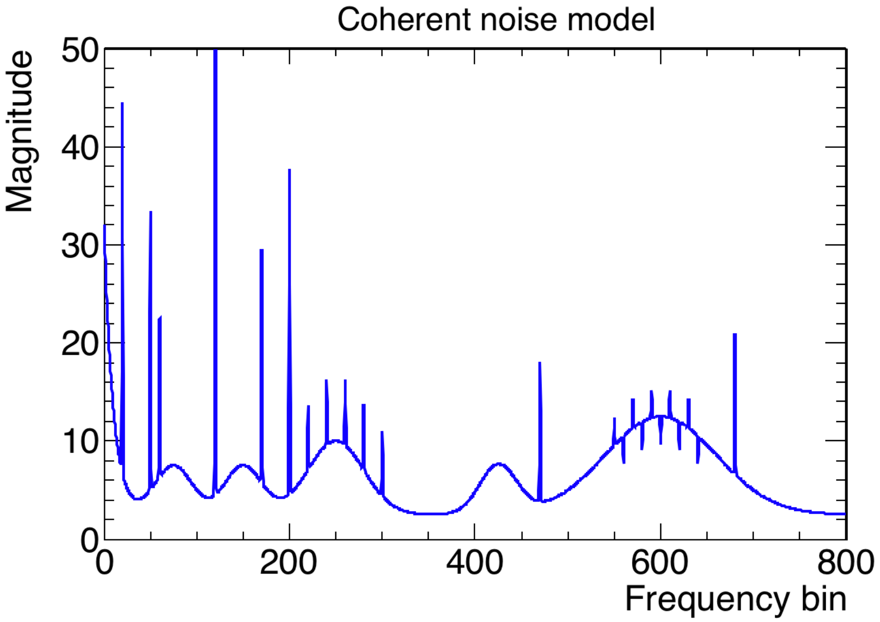
Courtesy of Yun-Tse Tsai

DetSim noise

- Amplitude of the noise (rough estimate):
  - 600 electrons induction
  - 500 electrons collection
- MCC10: parametrisation: $1/F^{\alpha}$ where $\alpha = 0$ (white noise)
  - One could add capacity of the wires etc:
    - $\alpha = 1$ → pink noise
    - $\alpha = 2$ → brown noise (pedestal variation)
- Later, added exponential noise:
  - $\exp(-\alpha F)$ + local peaks
- 2 different ways to handle the noise simulation:
  - Extracting from the ProtoDUNE data.
  - Simulating from electronic and “first principles” … and try to reproduce ProtoDUNE data.
- Coherent noise → variation of the noise coherently across channels (up to here, all the channels vary independently)
  - Pedestal: 16 channels together.
  - 40 (induction) or 48 (collection) channels are wire to the same mother board the mother board.
  - Quite hard to implement, hopefully soon in the simulations.
- Detector noise (microphonic, etc, …) no study yet.

Pink noise simulation





Miquel Nebot

CalWire, Deconvolution, advanced hit finding, reconstruction…

- These are beyond the scope of DAQ and triggering.
  - Doesn't run online (very slow)
  - Cannot be used for triggering.
- Basic idea is to recover the "raw" signal:
  - Get rid of the noise → filters
  - Get rid of the shaping → deconvolution
- The APA and DAQ consortia will inform this, but I'm not sure either will be directly responsible for this.
- For more about reconstruction, see previous tutorial: https://indico.hep.manchester.ac.uk/conferenceOtherViews.py?view=standard&confId=5346

Trigger Primitive Finder TPC

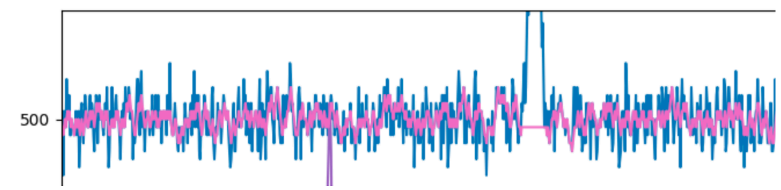
- Online hit finding is central for the triggering → allows us to have simpler (smaller) data objects to manipulate to issue trigger.

> "Hit" is a data object which contains:
> - Time (tick)
> - Time extent (RMS)
> - Amplitude or integral (or both)
> - Channel number

- For triggering, only collection wires will be used:
  - Smaller noise
  - Simpler shape
  - Smaller instantaneous data volume.
- Few "basic steps" that can be ran online:
  - Pedestal subtraction
  - Filtering
  - Hit finding

Trigger Primitive Finder
TPC

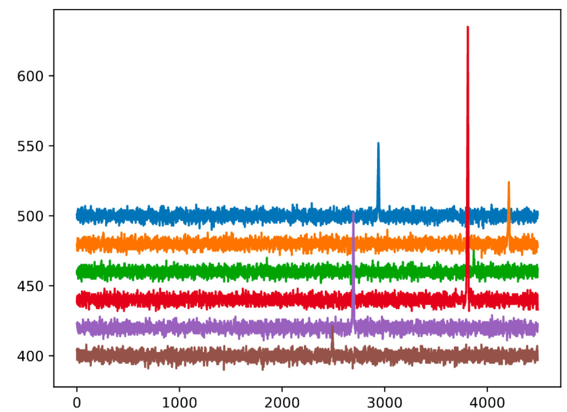
- More details about the hit finder (Phil Rodrigues)
  - Designed to be implemented online with a relatively simple algorithm:
    - FPGA / GPU
    - Simple data type (short int)
  - Pedestal subtraction:
    - "Running median" calculation over the waveform
    - Look a the signal 6 ticks ahead, if the waveform has a variation greater than a number of ADC stop updating.
    - Effectively a high pass filter (will filter low frequency component)
  - Finite Impulse Response filter (FIR):
    - Average the hits in a clever way to get rid of high frequency noise.
    - In the baseline the number of hits history needed is 7 (we say 7 taps FIR filter)
  - Hit finding step:
    - If the resulting waveform goes over the threshold, call this a hit and pass it downstream.



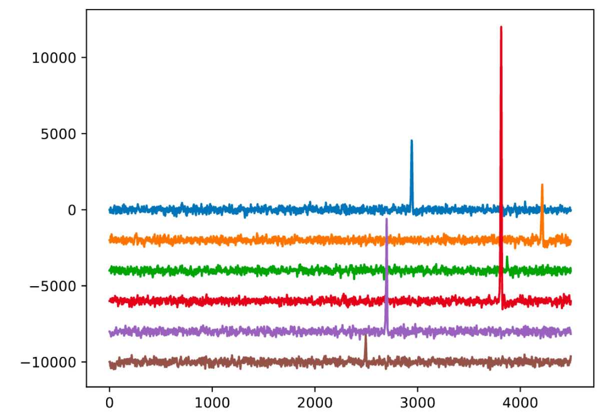
Pedestal estimation (pink)
Waveform (blue)

FIR filter:

$ADC_n = 2ADC_{n-1} + 9ADC_{n-2} + 23ADC_{n-3} + 31ADC_{n-4} + 23ADC_{n-5} + 9ADC_{n-6} + 2ADC_{n-7}$

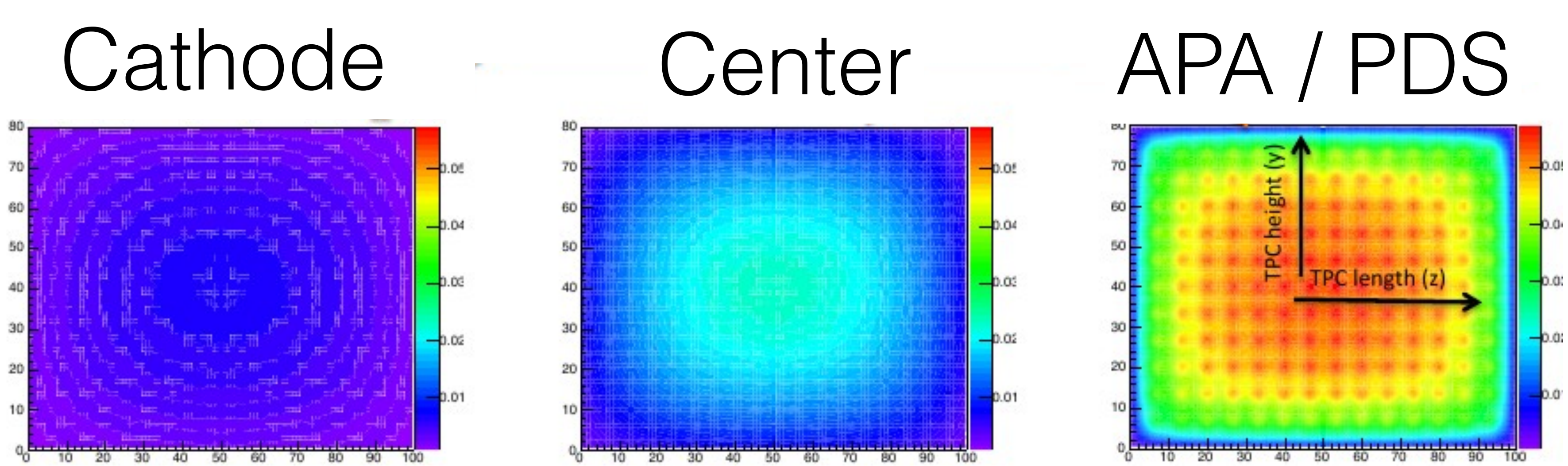


FIR filter



Photon simulations

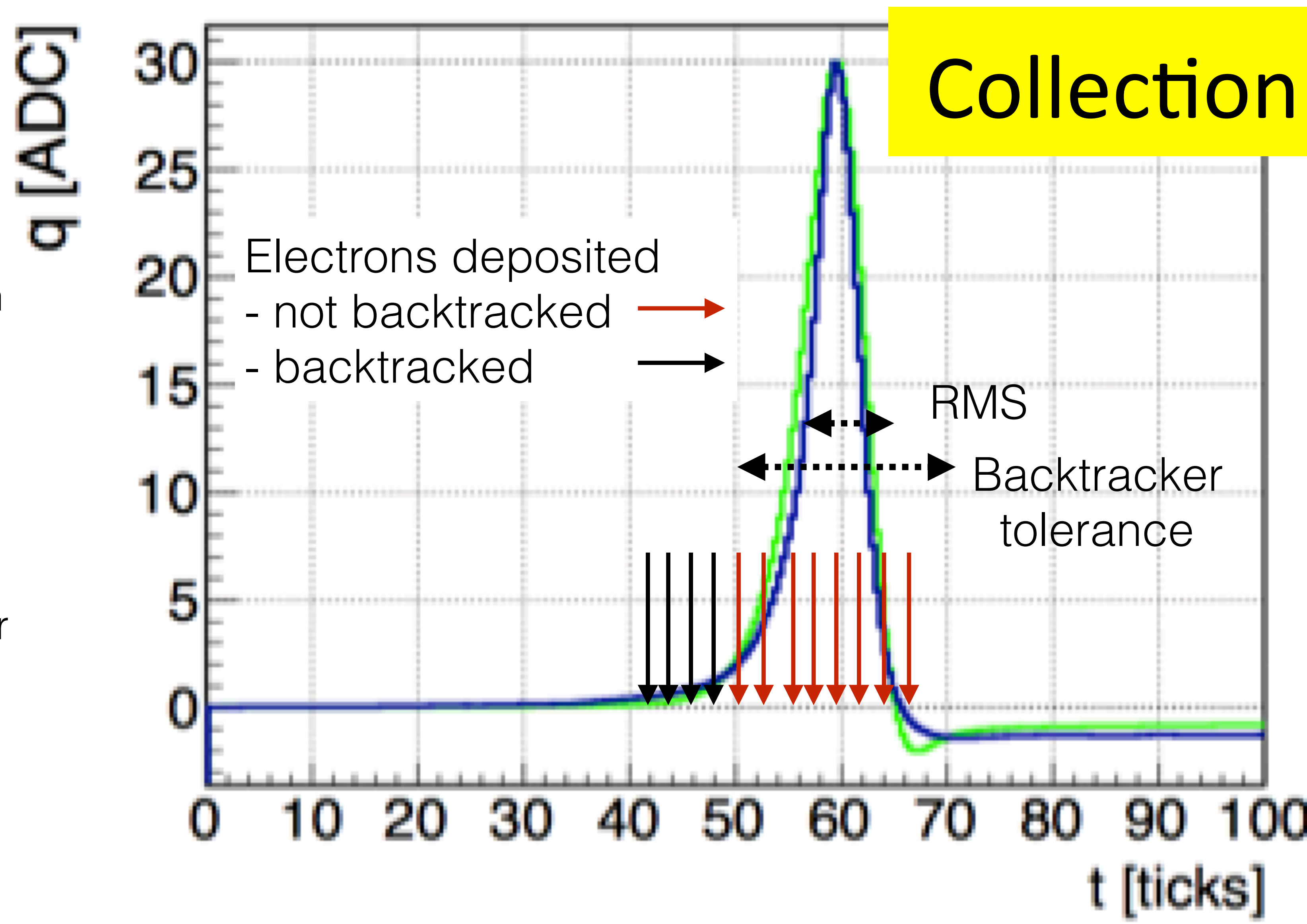
- We are back at the LArGeant step, i.e. we have the number of photons that are created at a position in the detector, from scintillation.
- How does this gets us to the light we see in the detector?
  - Can't simulate each photons (24k photons / MeV of particle lost).
  - → "Fast Optical Simulation" and "Photon Libraries" to avoid having to follow each photons
  - Photon Libraries:
    - Stores the visibility of photons for each voxel: i.e. the probability for a photon to be detected.
    - Advantage: Say you have a better detector, the visibility will be better, you just need to change the library. You can in fact "resimulate" all the photon on top of the one you had before.
- Hit finding… I really don't know, if somebody knows this, intervene now.



Diego Garcia-Gamez

Truth matching Backtracker

- This slide is more about the LArSoft itself.
  - The truth matching is a bit complicated, since we don't follow each electrons
- After doing fancy hit finding, how do you know which hit belong to which events?
  - Trigger primitive knows nothing about the "truth" waveform, only input is the waveform itself (rather than something that says which electrons came from where).
- Truth matching is achieved by the "backtracker":
  - Looks at the hit timing and the hit extent (RMS) to make the truth matching.
  - It looks back in the deposited electron and see which electrons are consistent in time and wire.
  - Sometimes it doesn't work as expected. One has to a bit careful when doing some custom hit finding (essentially if your hits are too short, or you RMS estimation is not accurate).
- PDS → basically the same principle.



Conclusion

- Simulating LArTPC is quite complicated.
  - Requires extensive knowledge of the LAr properties if you don't want to waste unreasonable amount of time and CPU and RAM:
    - Drifting electrons
    - Propagating scintillation light
  - Makes the code quite segmented and with different many steps.
  - Don't worry we have all been there.
- Electronic simulation is also quite complicated, and it's still being developed quite extensively as ProtoDUNE data arrives and decision are made on the hardware.
  - Hopefully it's a bit clearer now.
- In the tutorial, we will try to touch some aspect of the hit finder and some of the backtracker
  - You will run all the simulations that are required for TPC hit finding in the case of SN events, as we have seen this is one of the main challenge for DUNE (let alone solar neutrinos).