

UPDATE ON FLASH FINDING, TRACK MATCHING, COSMIC REJECTION

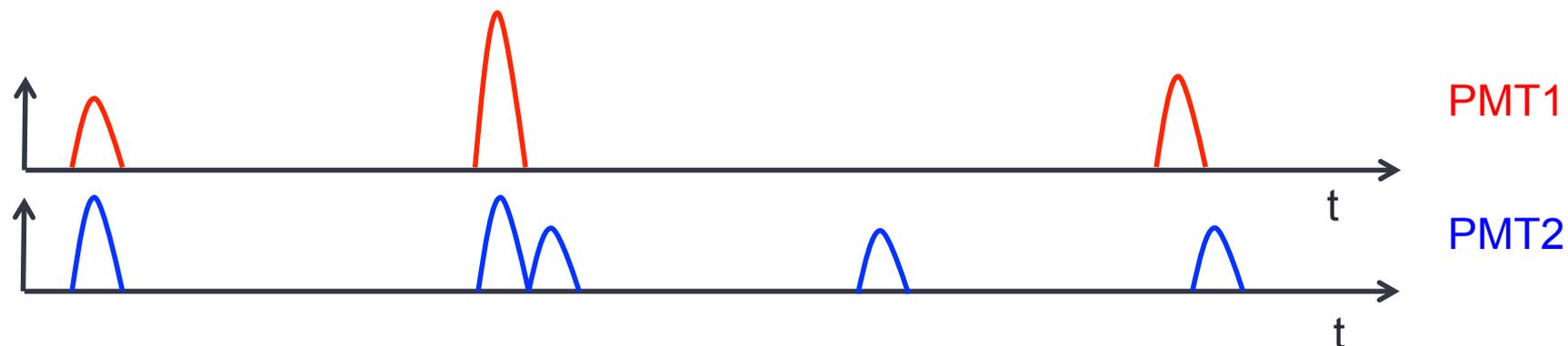
Ben jones, MIT

This Talk

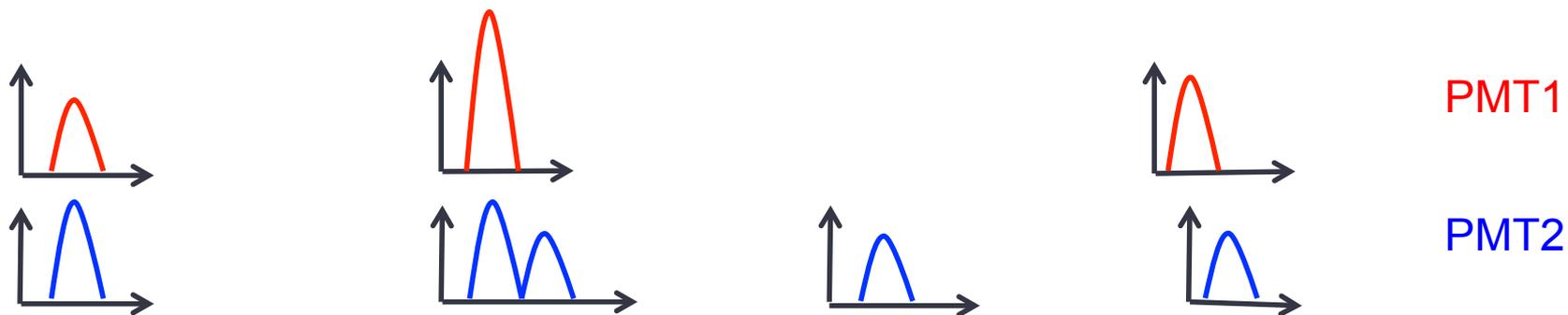
- **Part 1: New Flash Finder**
- Part 2: Cosmic Tagging

Flash finding

Old paradigm : OpDigi objects (toy digitization, BJPJ)



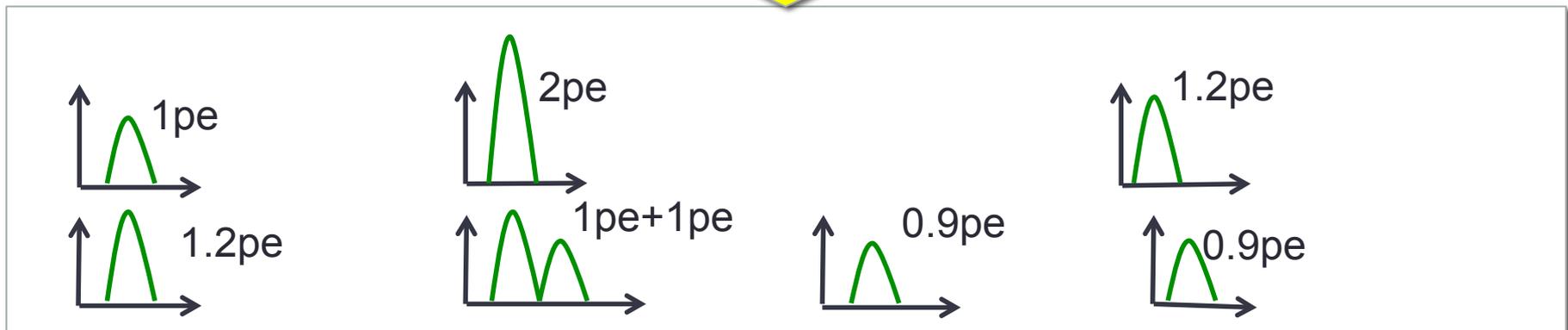
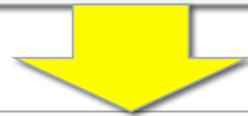
New paradigm : OpFIFOChannel objects (Seligman, Terao)



Flash finder

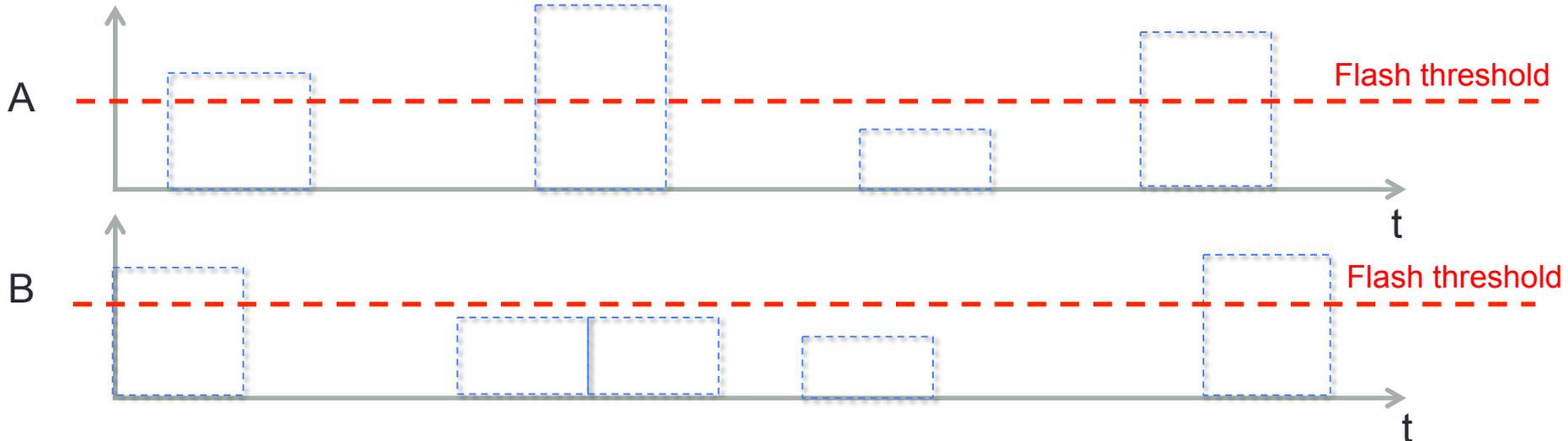
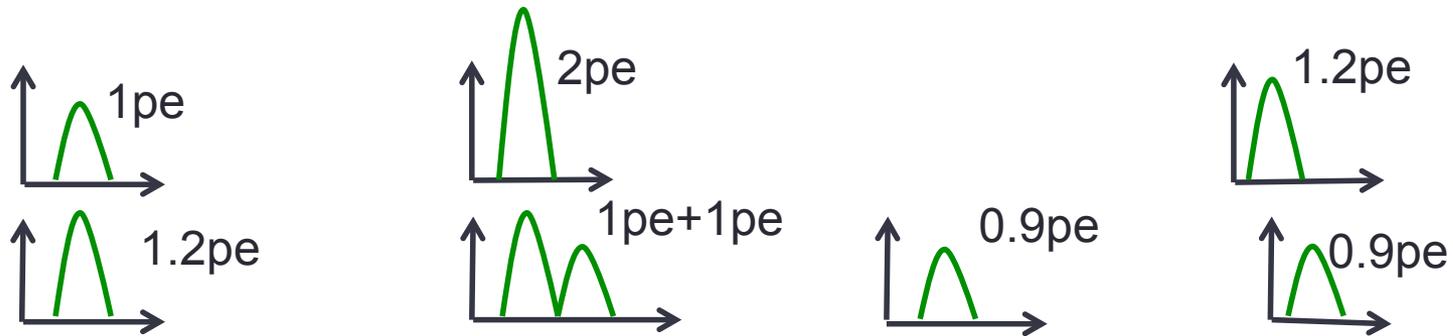
Step 1 : Find pulses using Kazu's peak finding framework

- New objects and new way of handling the data mandated an almost-complete rewrite of the flash finder. This now complete.



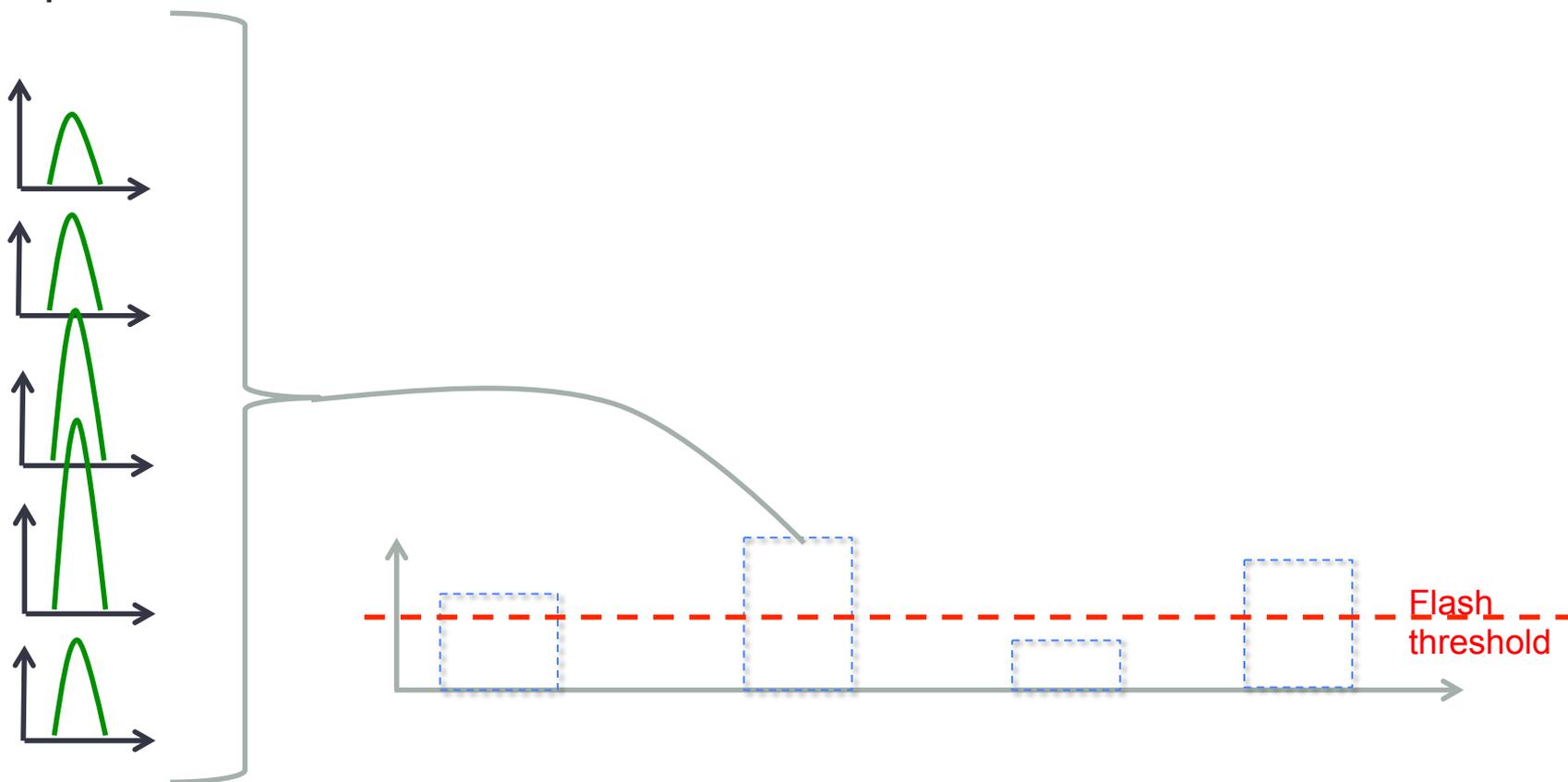
Flash finder

Step 2 : Fill 2 broadly binned accumulators, tracking which channels contributed to each bin



Flash Finder

If we cross the threshold, this bin and its associated pulses get tagged as a protoflash.



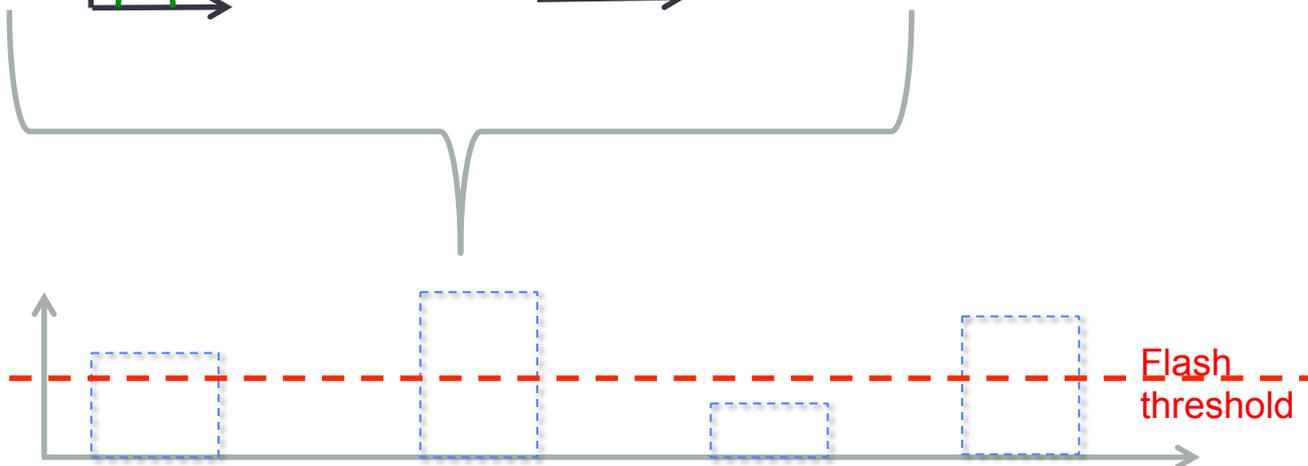
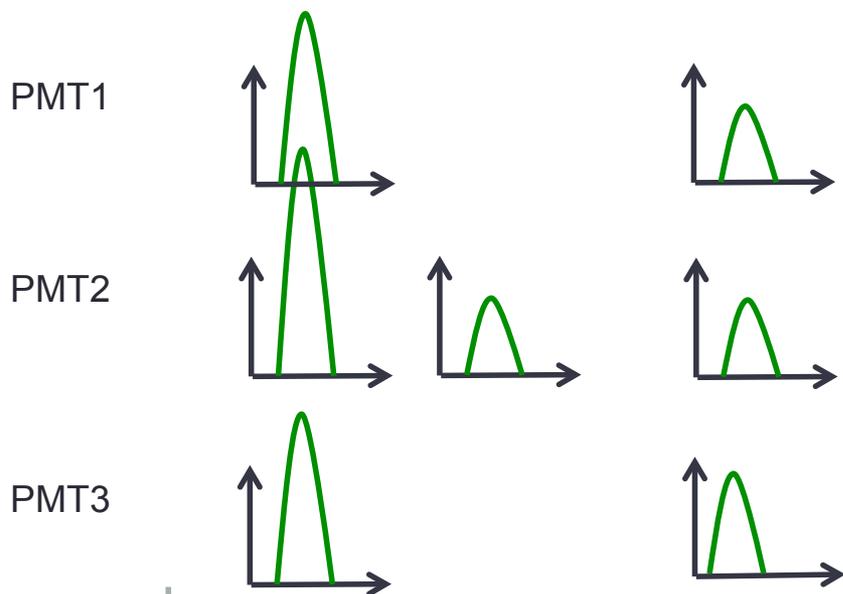
Removing OpHit Duplicates / Overlaps

- Because we used the 2 accumulators, some flashes appear in A, some in B, some in both
- To overcome this:
 - a) Sort the flashes from largest to smallest
 - b) Let flashes start claiming hits. Each hit is in only one flash.
 - c) Largest flash always gets dibbs.
 - d) Flashes which end up under threshold after this procedure are thrown out

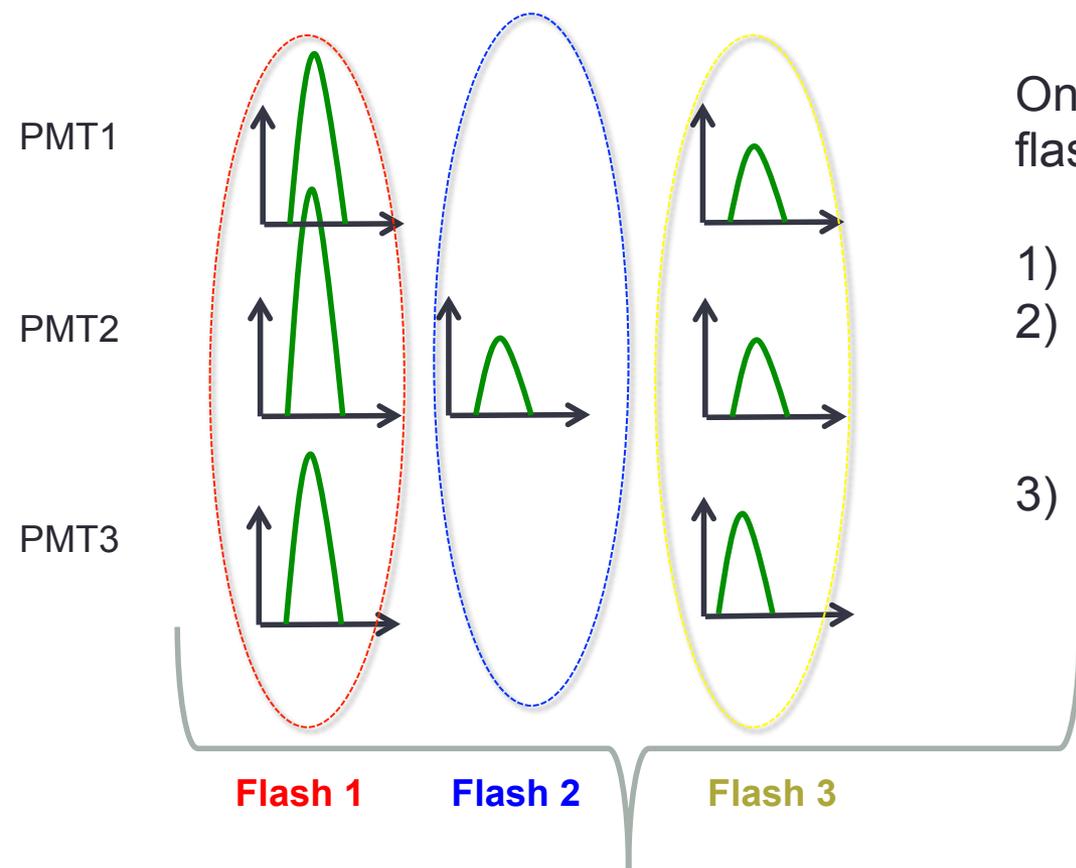
Breaking Up Protoflashes

One accumulator bin can hold several flashes. To separate:

- 1) Sort collected OpHits by size
- 2) Starting with largest, collect those on other channels within n reconstructed widths (presently $n=2$)
- 3) Repeat for next largest remaining OpHit



Breaking Up Protoflashes



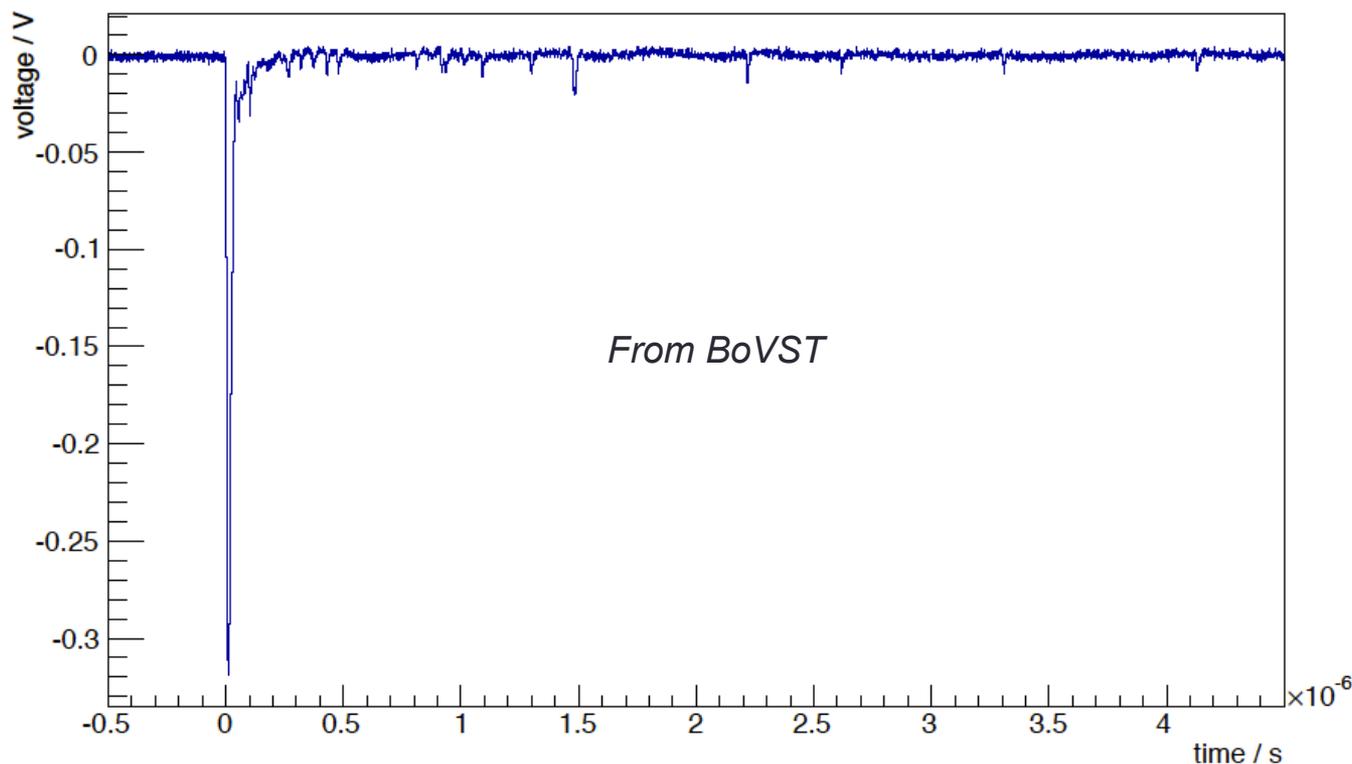
One accumulator bin can hold several flashes. To separate:

- 1) Sort collected OpHits by size
- 2) Starting with largest, collect those on other channels within n reconstructed widths (presently $n=2$)
- 3) Repeat for next largest remaining OpHit



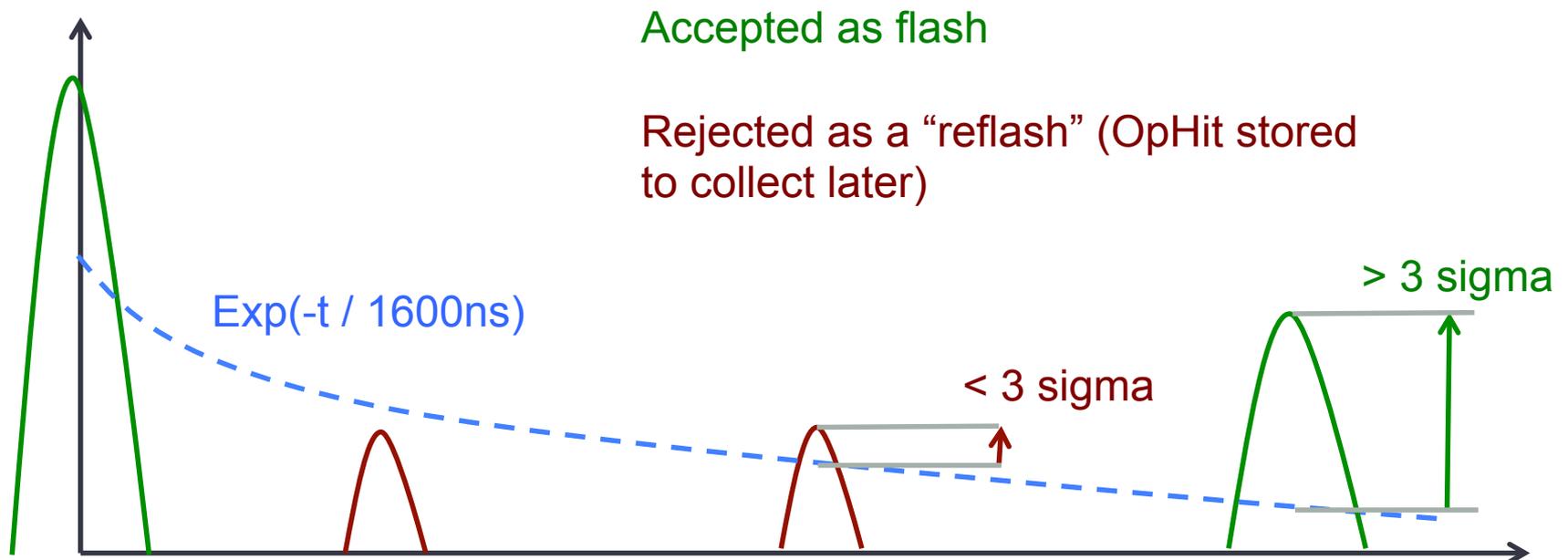
Reflash Filtering

- Scintillation in LAr has a prompt component and a slow tail
- We want to pick out only the prompt light in the first pass – this is important for good timing resolution.
- On the first pass we want to “throw out” fake flashes from coincidences in the late light tail of another flash
- In a second step we will collect the late light with the prompt in flash (tbd).



Reflash Filtering

- Scintillation in LAr has a prompt component and a slow tail
- We want to pick out only the prompt light in the first pass – this is important for good timing resolution.
- On the first pass we want to “throw out” fake flashes from coincidences in the late light tail of another flash
- In a second step we will collect the late light with the prompt (/todo).



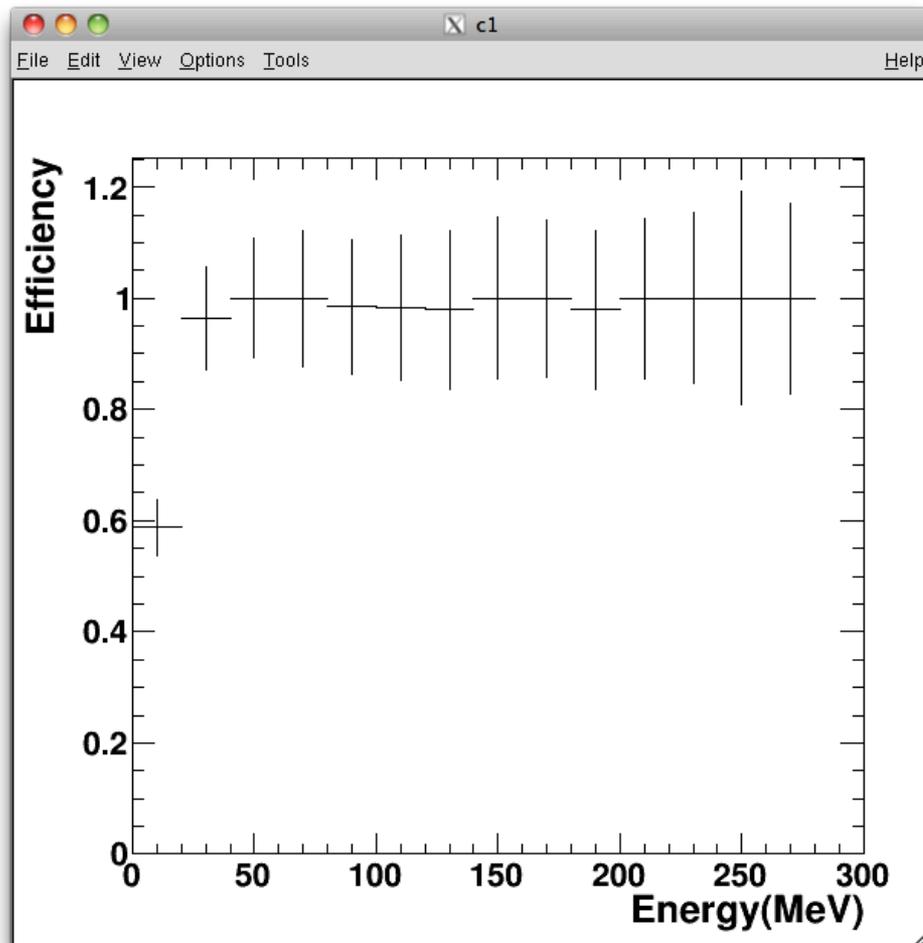
Then store in event

Summary:

- 1) Read in FIFOChannelData and apply Kazus pulse finding
- 2) Accumulate PEs per time in 2 broad accumulators, check for flash threshold during accumulation
- 3) Make OpHits and collect into protoflashes
- 4) Remove duplicate hits with bigger flashes taking priority
- 5) Break up protoflashes into smaller units based on hit widths
- 6) Filter reflashes consistent with late light from another flash
- 7) Store OpHits, OpFlashes, Associations in event

Performance Check

- Quick check : 1000 low energy single proton events

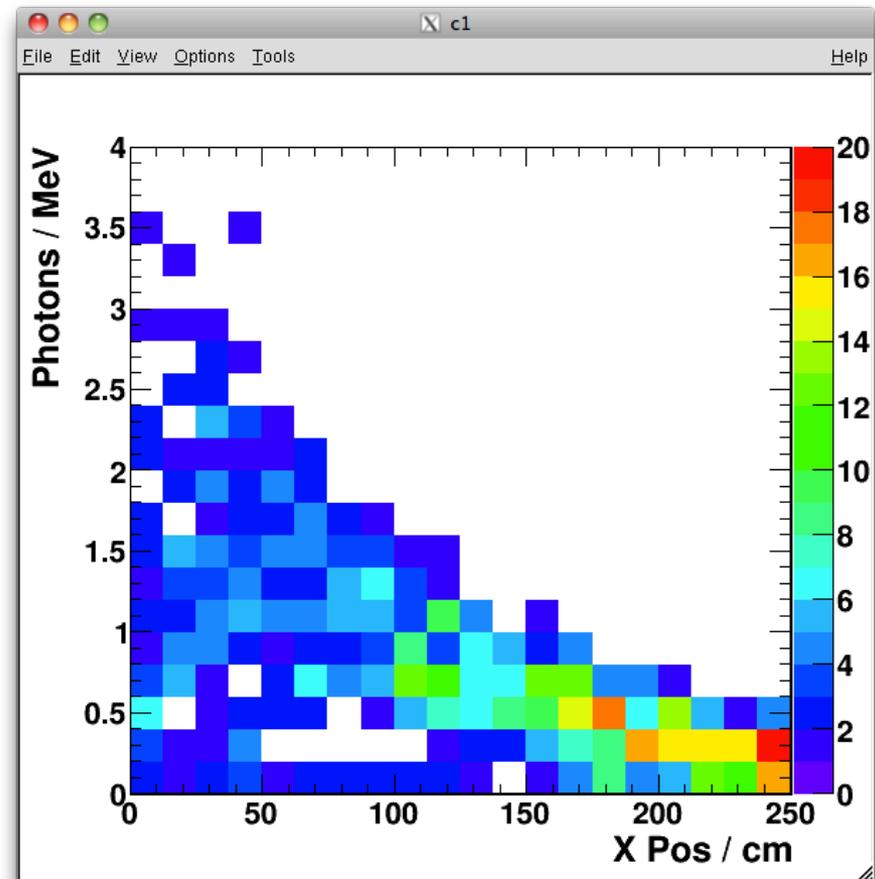
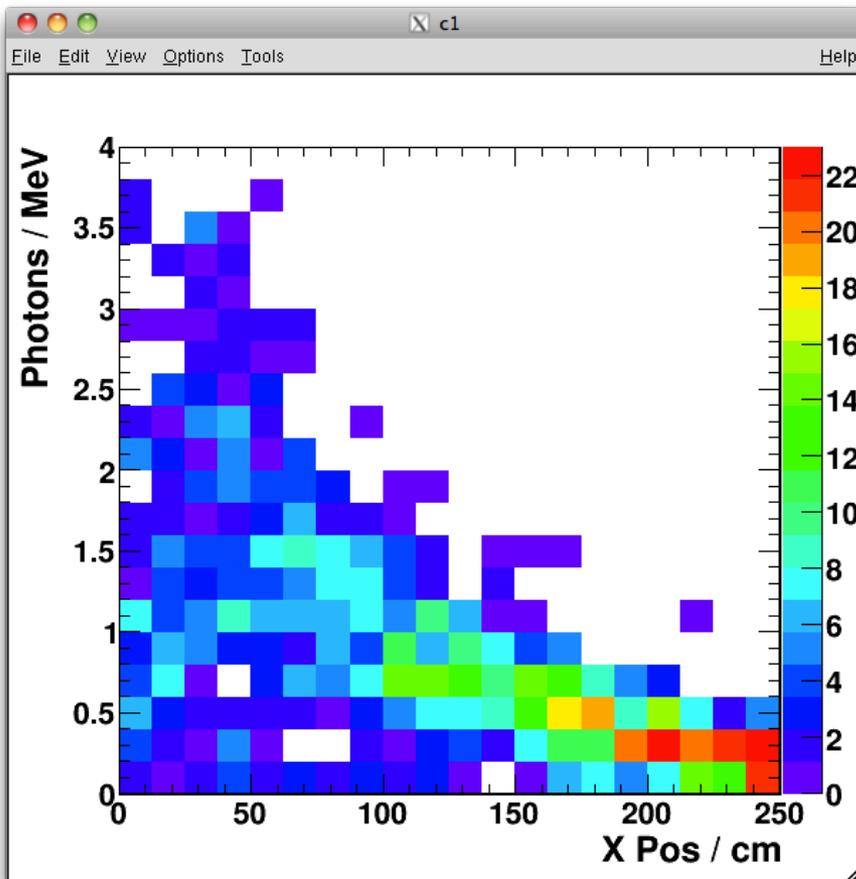


Require flash within beam gate region,
with >2PE

Performance comparable to, even
slightly better than, previous flash
finder.

Performance Check

- Energy resolution all
- Energy resolution $>50\text{MeV}$

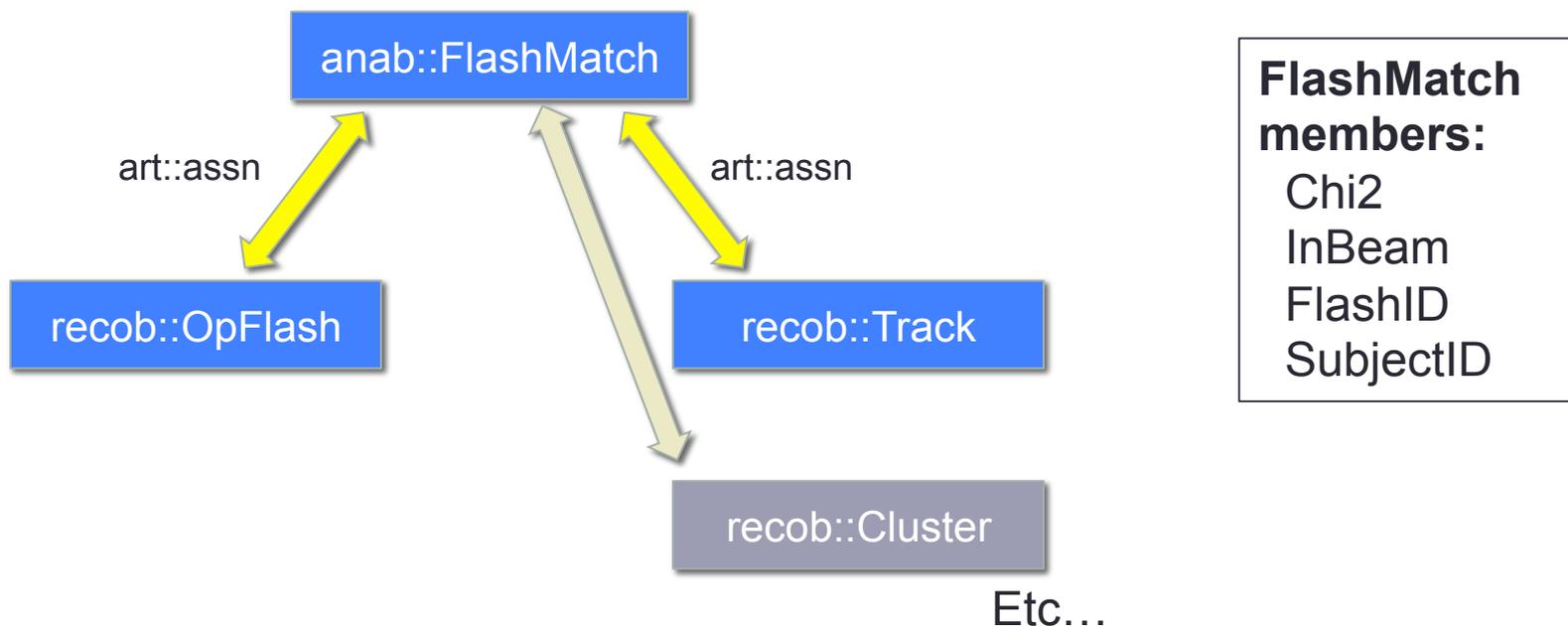


This Talk

- Part 1: New Flash Finder
- **Part 2: Cosmic Tagging**

Track Tagging

- There may be several modules which provide some form of information about TPC objects based on optical information
- This information is stored in the form of an **anab::FlashMatch** object which associates to the flash and the TPC objects



Track Tagging

*I'll only show
this one today*



So far, two approaches attempted:

- 1) Try to rule out tracks which are obviously not consistent with beam-time flash (***BeamFlashCompatabilityCheck_module***)
- 2) Try to match tracks from cosmics to particular flashes out of time with beam (***TrackTimeAssoc_module***)

Performance of both is assessed with new ana module,
TrackTimeAssocAna_module

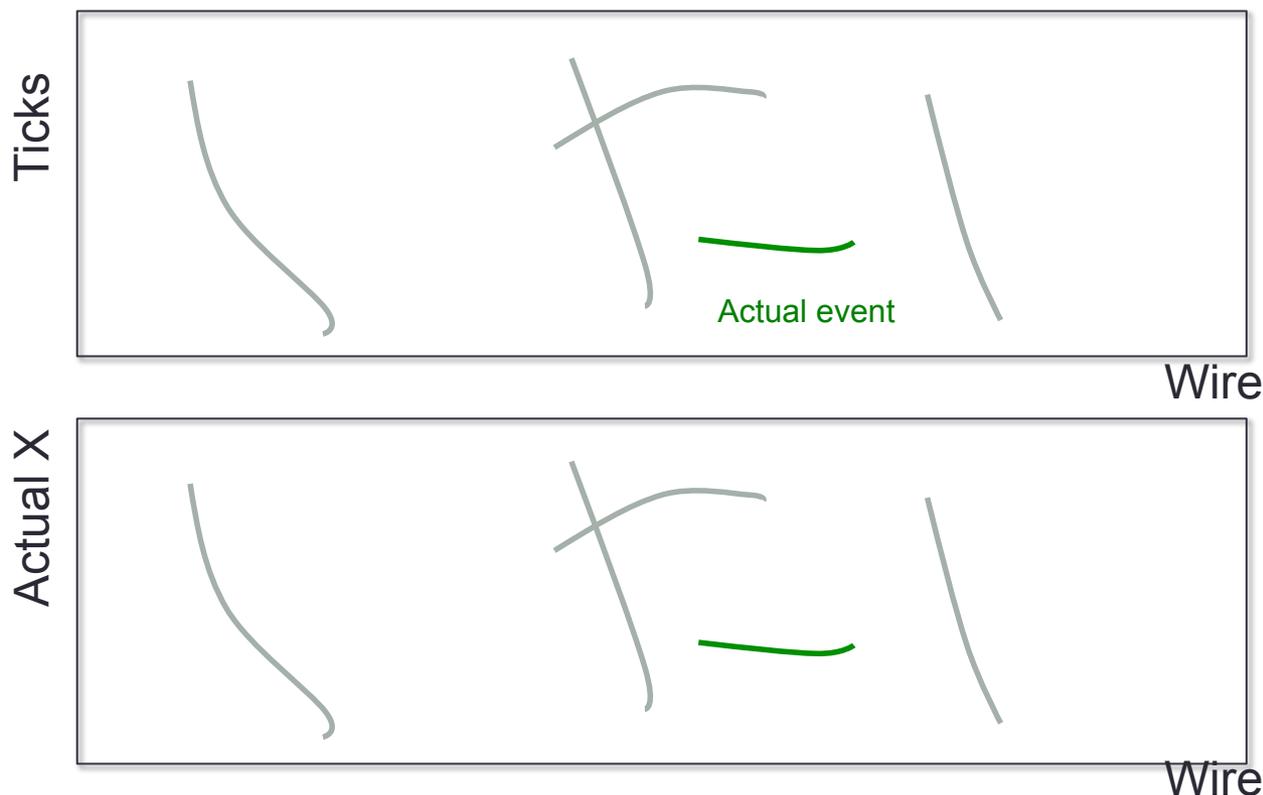
The capabilities of this ana module are limited – in particular no MC truth information.

Wes and I have begun working on a much more complete ana module, to properly assess cosmic tagging performance.

Approach 1:

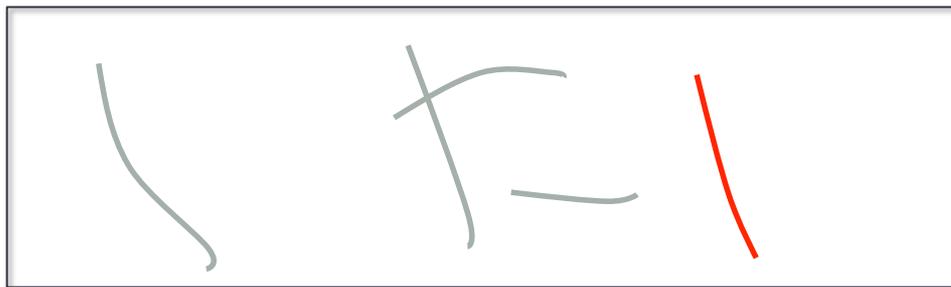
Note : Credit to Wes for helping devise this scheme!

- Assume that every track is a beam-event candidate
- That means its x position is accurate as represented in the TPC

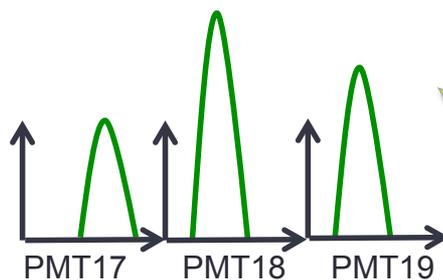
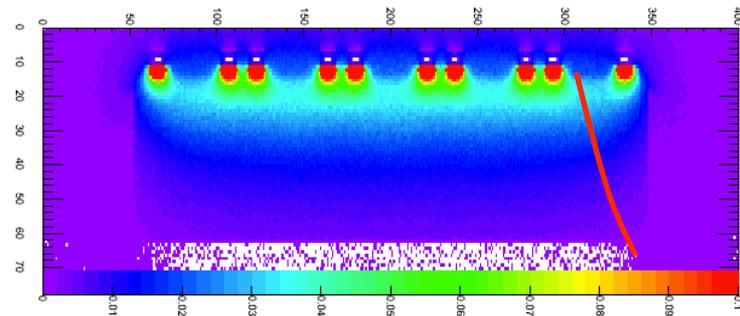


Approach 1

- Make a hypothesis for the light yield expected from each track, using fast optical sim table

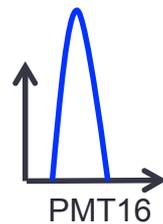


Wire

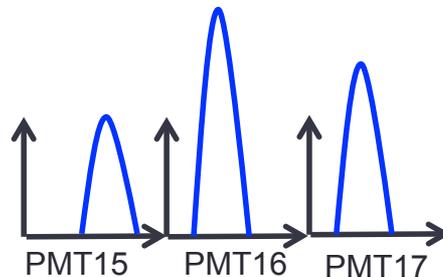


Approach 1

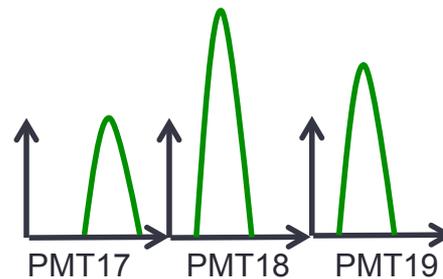
- Compare to each actual flash detected in the beam window



Actual on-beam flash 1

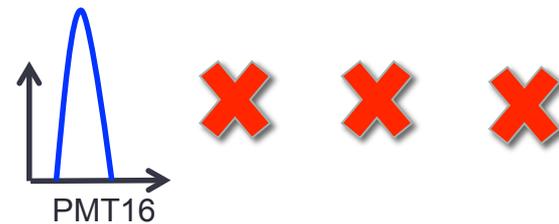


Actual on-beam flash 2

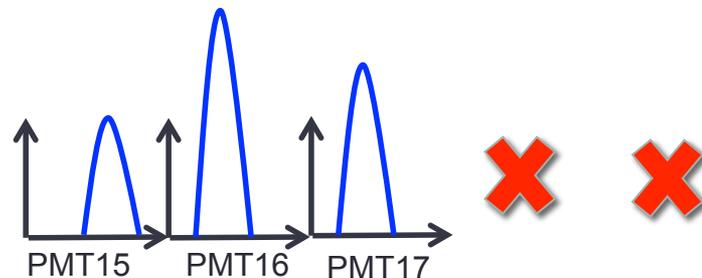


Actual on-beam flash 3

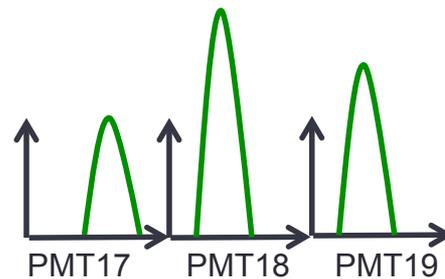
- The hypothesis is deemed inconsistent with a beam event if:
 - A) The hypothesis exceeds the signal by 4 sigma on any PMT
 - B) The hypothesis exceeds the signal by 4 sigma overall
- If inconsistent with all on-beam flashes, we tag this track as off-beam.



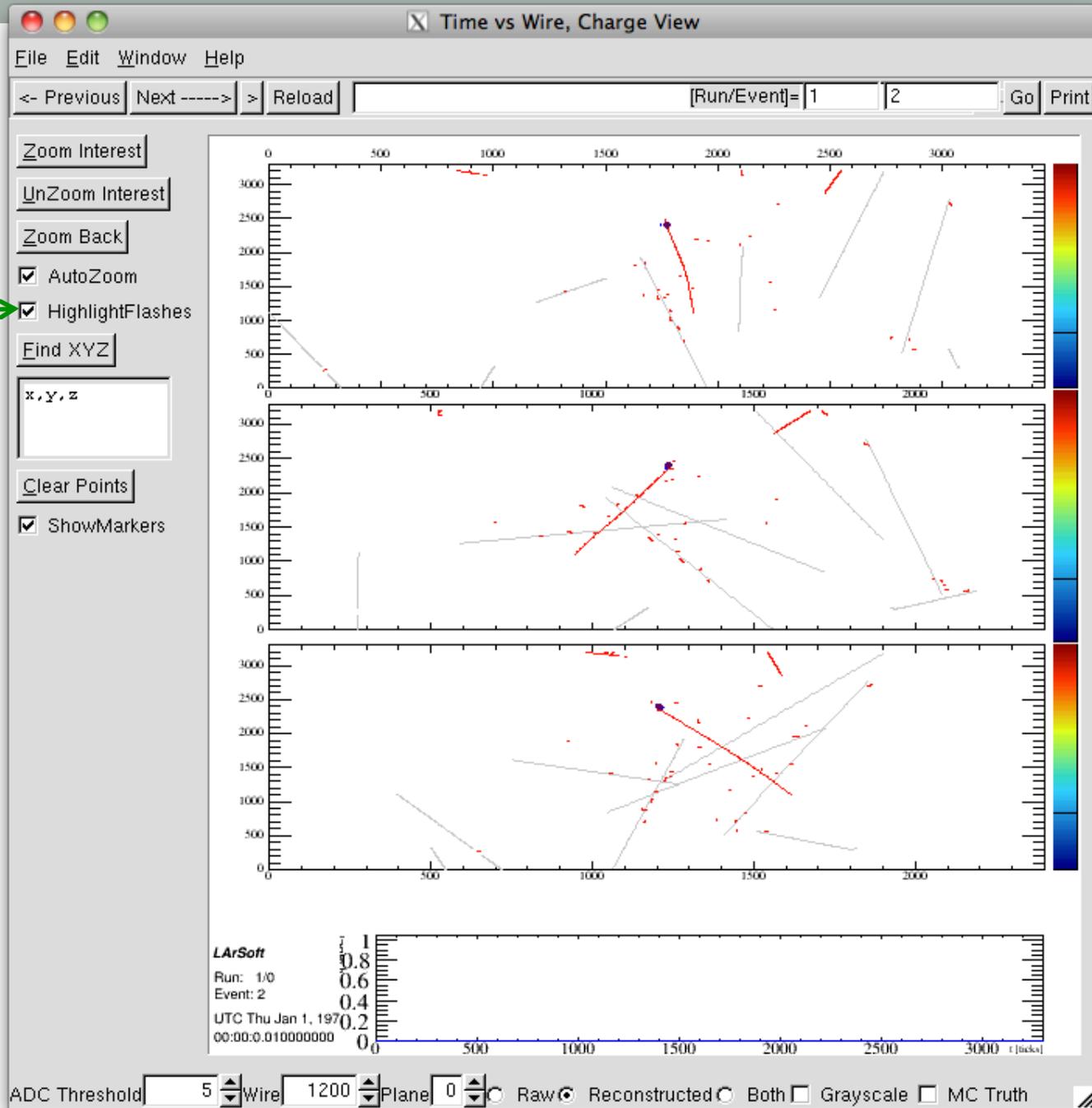
Actual on-beam flash 1



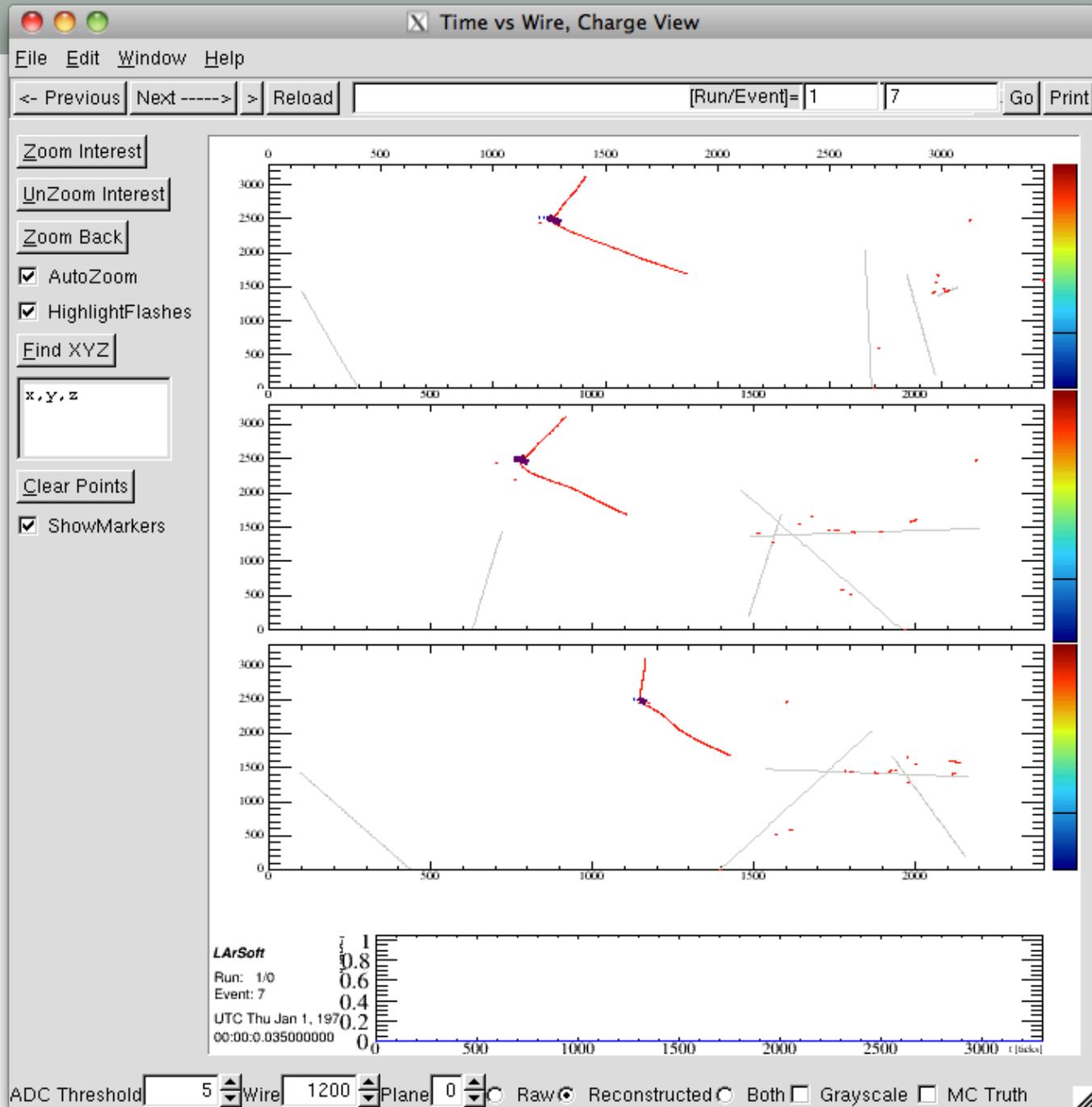
Actual on-beam flash 2



Actual on-beam flash 3



Note:
Request
permission
to commit
this feature?

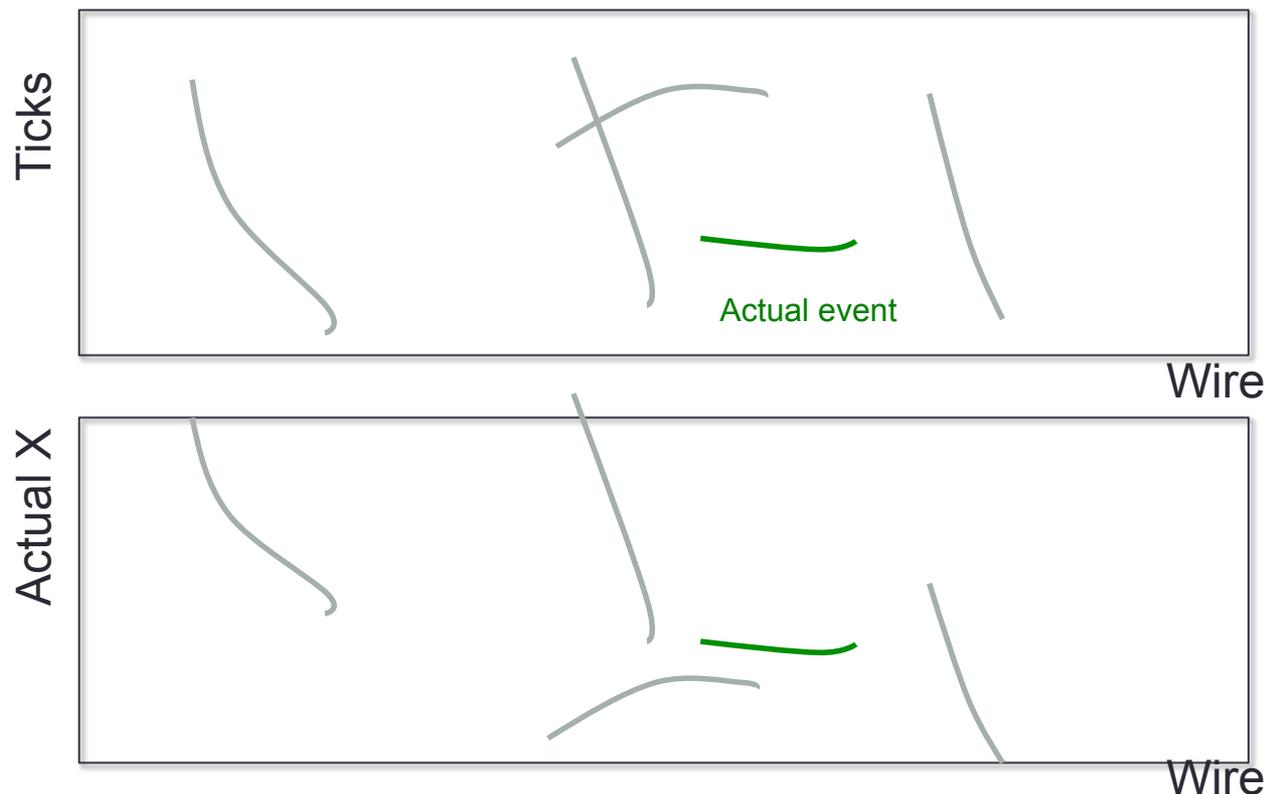


Quantitative Results

- Early results are promising : Based on 10 CCQE events, cosmic rejection at ~80-90% level and no events with neutrino event fully discarded.
- Obviously we need more stats.
- This module is "MCC ready".
- We will learn a lot more once we fully develop a detailed analyzer

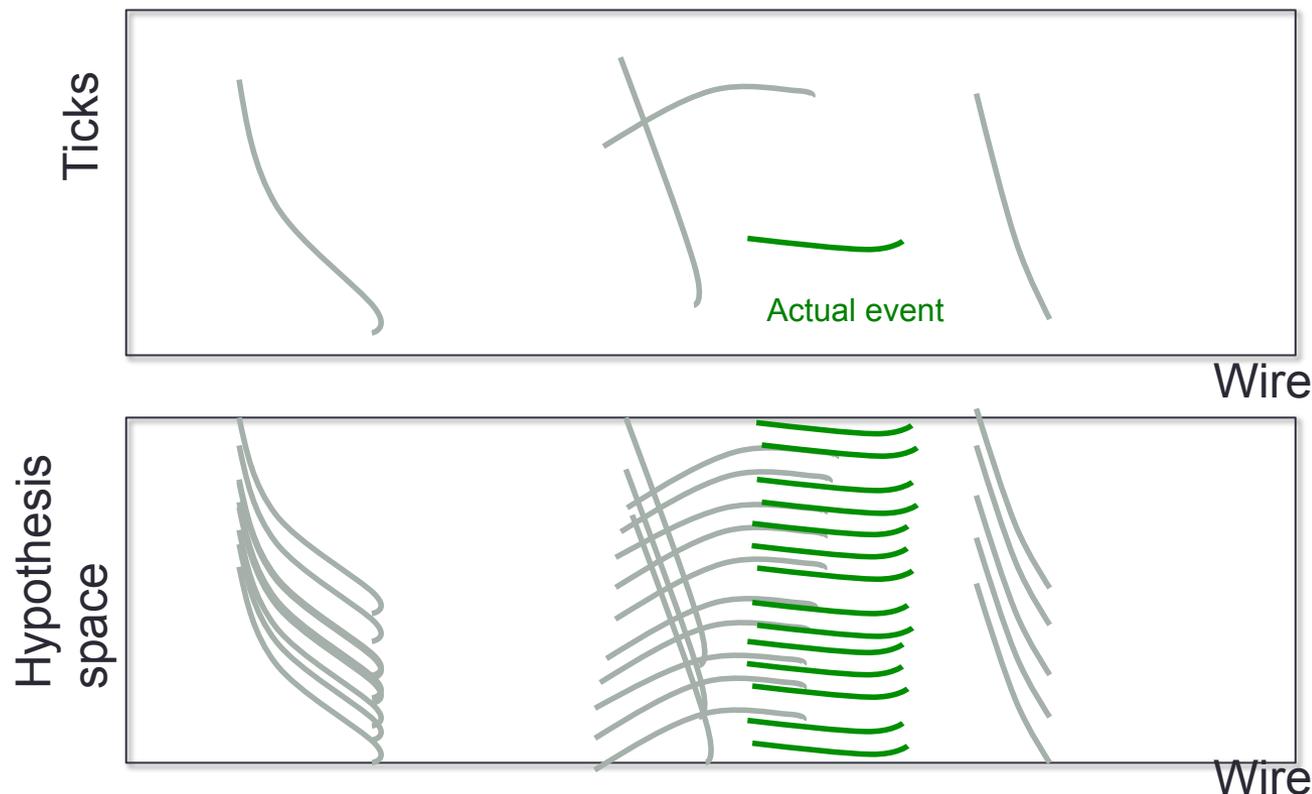
Approach 2:

- Now every track is a beam and a cosmic candidate
- That means its x position is not accurately represented in the TPC



Approach 2:

- We make a hypothesis for each possible X and calculated chi2 for match to each flash.
- Then apply a “stable marriages” algorithm to pair tracks to flashes



Approach 2

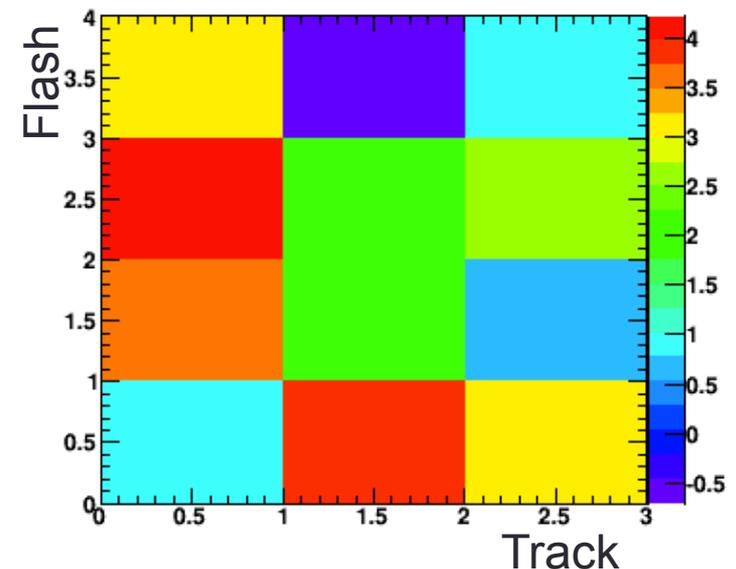
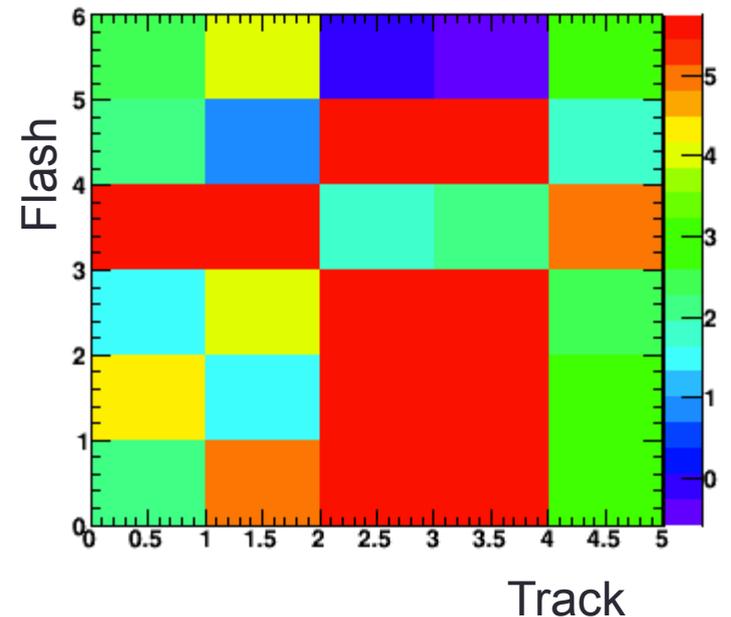
Typical Chi2 Map for an event shown right

At present, this approach is not performing well – the problems are threefold:

- 1) Not every flash corresponds to a track
- 2) Cosmic tracks are in fact longer in the detector than is shown in the TPC
- 3) Often many tracks should match one flash – how to do these combinatorics?

** Your smart idea here! **

Log(chi2) map for 2 events



Summary

- New flash finder is hooked up working
- First pass cosmic rejection module using optical information appears to be performing well
- Both are “MCC ready”
- More advanced cosmic rejection algorithms will require time and cleverness.
- If you have time and / or cleverness, this could be a fun reconstruction task to get involved with!