

Database discussion

- Who's using the database? Success/failure reports...
- Requirements for database use.
 - * What have people done/tried to do with the DB?
(Andre:DAQ, Nick: ACNET/IFIX data)
 - * What have people wanted to do that was not supported?
- Connection Maps: what's the plan to get them (all) into the database?

Bad channel masks

- Before the reconstruction looks at “raw” data I think known “bad” channels should be masked. Bad channels would be channels that are *too hot* to be trusted, or *too quiet* that when they are on, they should not be trusted

- I think this applies to the TPC, wire chambers, RICH PMT's etc. etc.

- Basic scheme:
 - [1] A "quasi" offline job runs on the raw data and accumulates statistics on a subrun-by-subrun basis. This job could be onmon...

 - [2] Channels that fall within some normal range of activity are called Channels that fall outside this range are called bad

 - [3] List of bad channels are put into DB indexed by run/subrun

 - [4] RawData package provides simple filter subroutines to remove bad channels from a list

What should the table look like?

I'm not sure, but I think its simple. For RICH this could be as simple as:

`run# | subrun# | insertion date | pmt_number[]`

Then given a run/subrun you'd grab the list of channels inserted into the table most recently

Questions/Plan:

[a] Is this reasonable?

[b] Can we identify *one* person to

1. write code to identify bad channels in one detector system
2. create a database table
3. insert data into table
4. extract data and apply filter
5. report back

[c] Once one detector works full circle, we'll need volunteers for other detectors

[d] Time scale: after shutdown? (Not that far away...)

RICH Reconstruction

Currently there are 3 modules:

RICHReco – S.Seun's code. Relies on having reliable track and momentum fits. For each track looks where rings are expected and calculates:

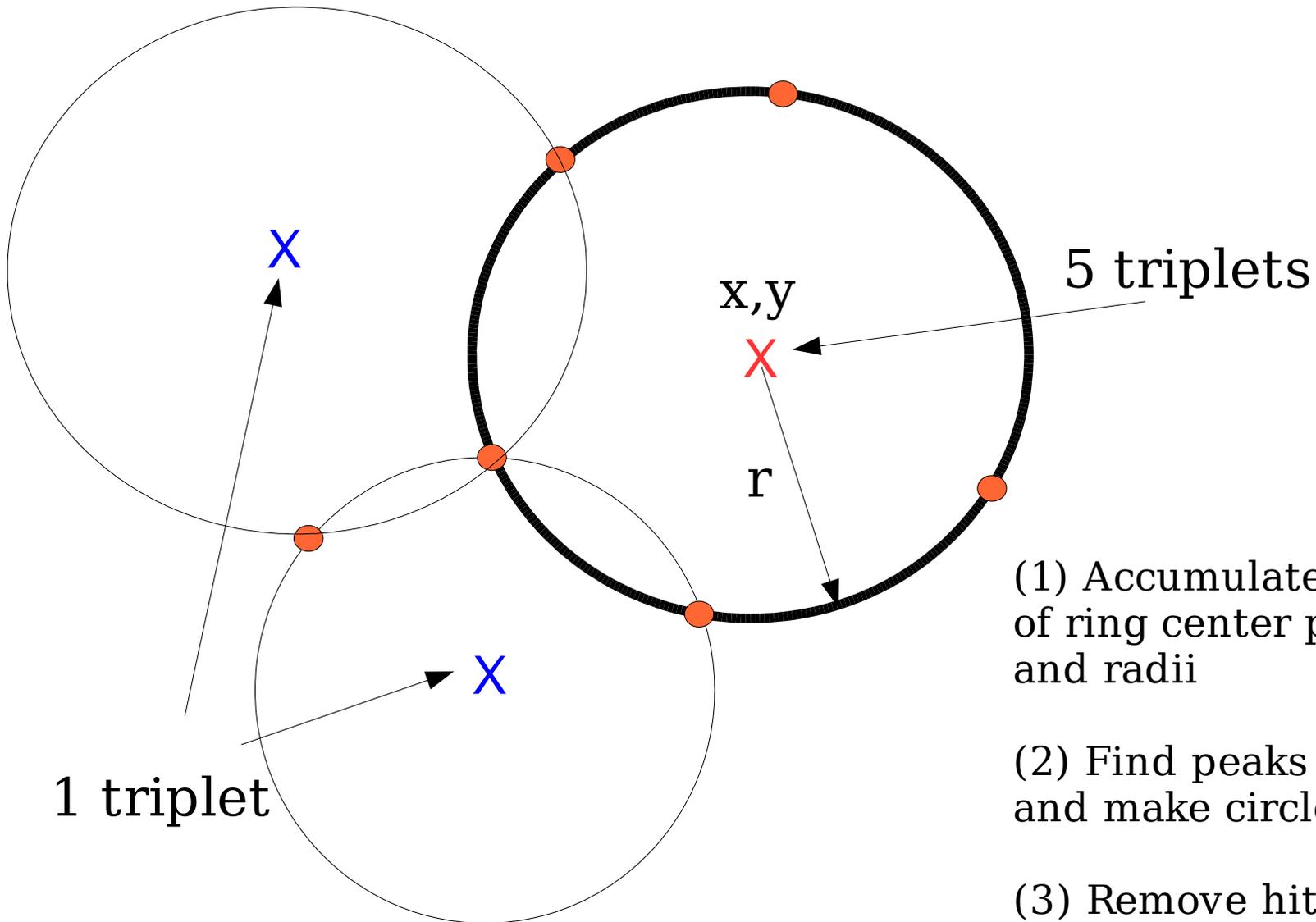
$$\mathcal{L}(\pi) - \mathcal{L}(K), \mathcal{L}(\pi) - \mathcal{L}(p), \mathcal{L}(K) - \mathcal{L}(p)$$

RICH CircFit and **RICH RadFit** – My code. Works without tracking. CircFit finds rings using Hough transform. RadFit performs a χ^2 fit to the ring radius and center.

*RICH*CircFit

Basic idea:

Every triplet of PMT's defines a circle w/ center x, y , and radius r

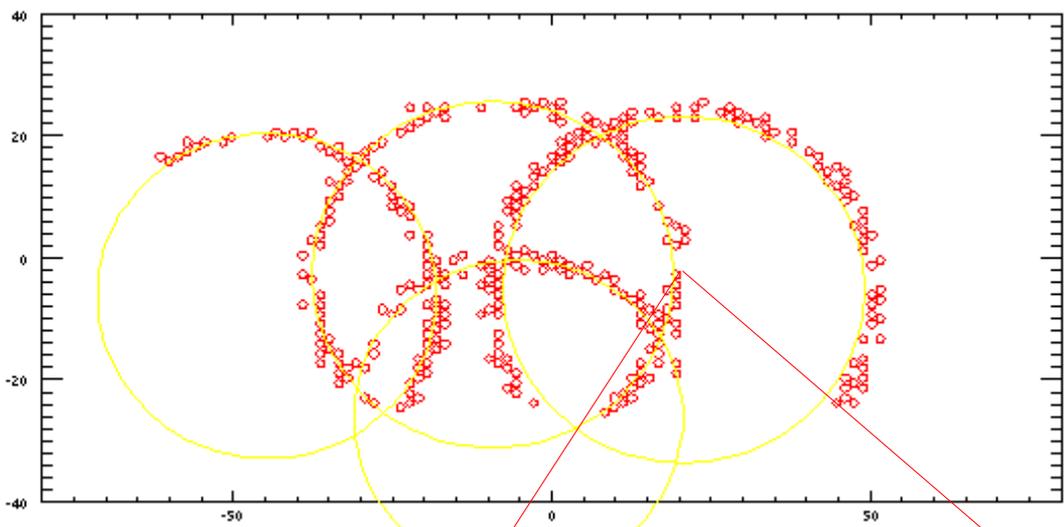


(1) Accumulate histograms of ring center positions and radii

(2) Find peaks in histograms and make circle (x, y, r)

(3) Remove hits on circle

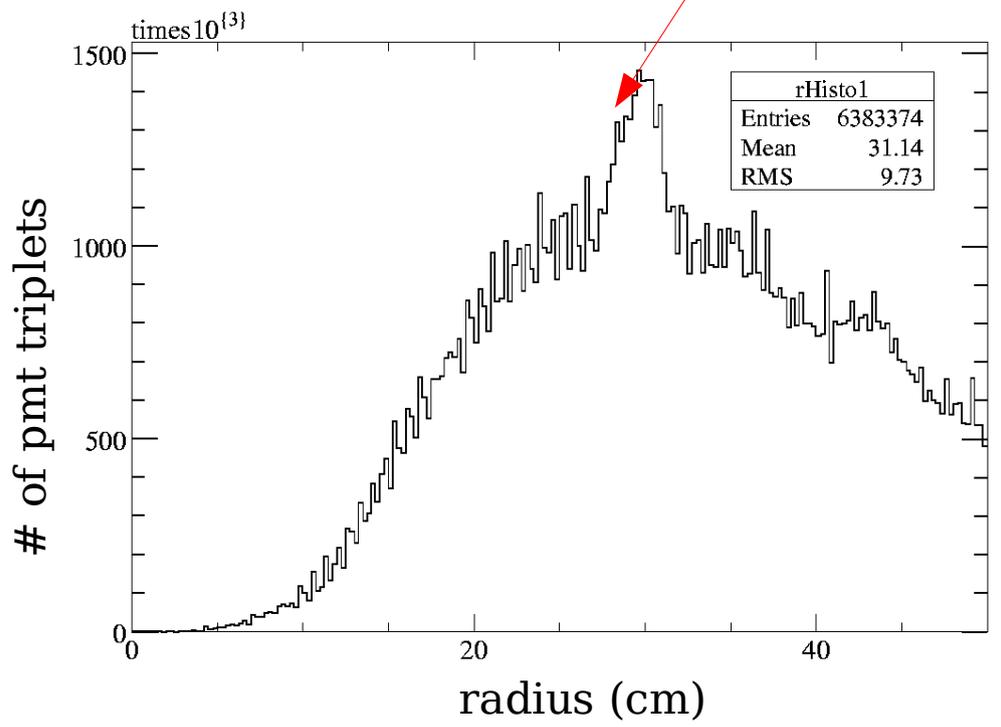
(4) Repeat until no circle is found



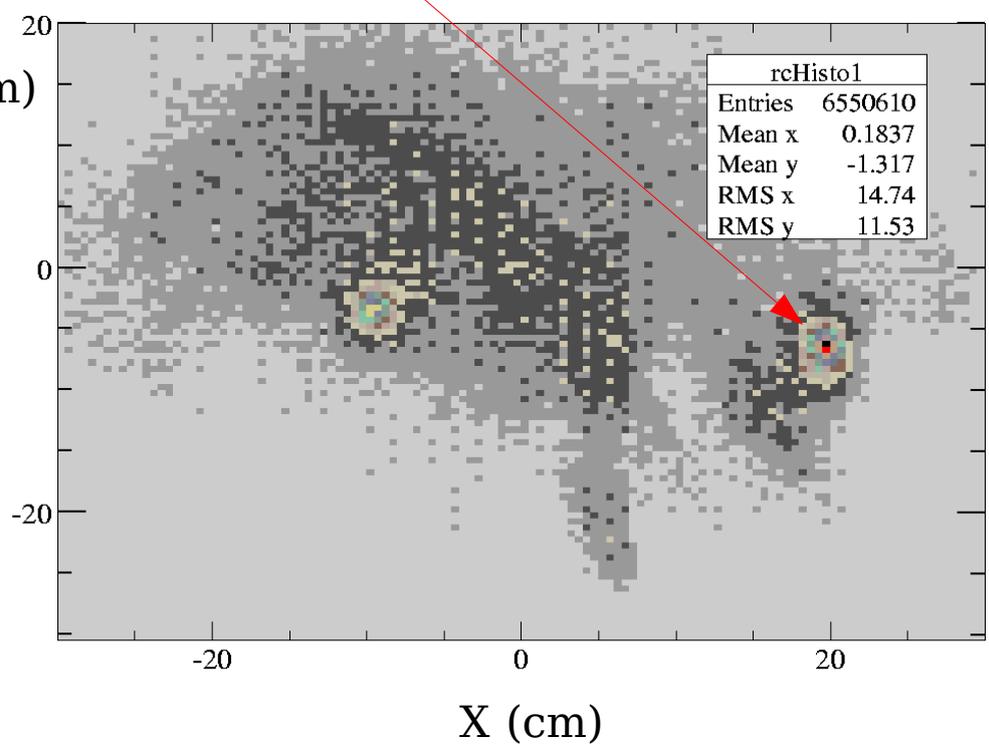
MC Event

yellow rings were found

Results saved to
evt.Reco("./rich")
as RICH Circ objects



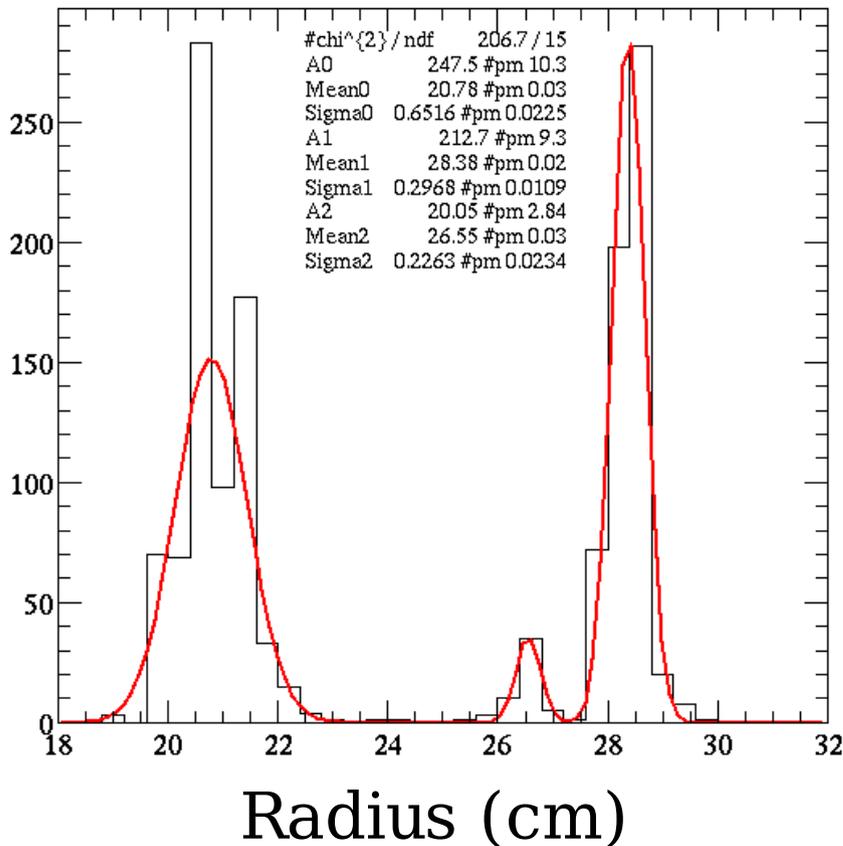
Y (cm)



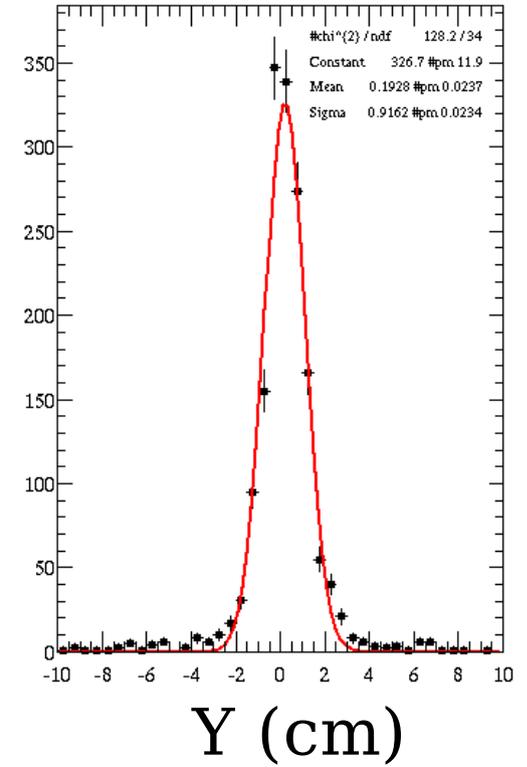
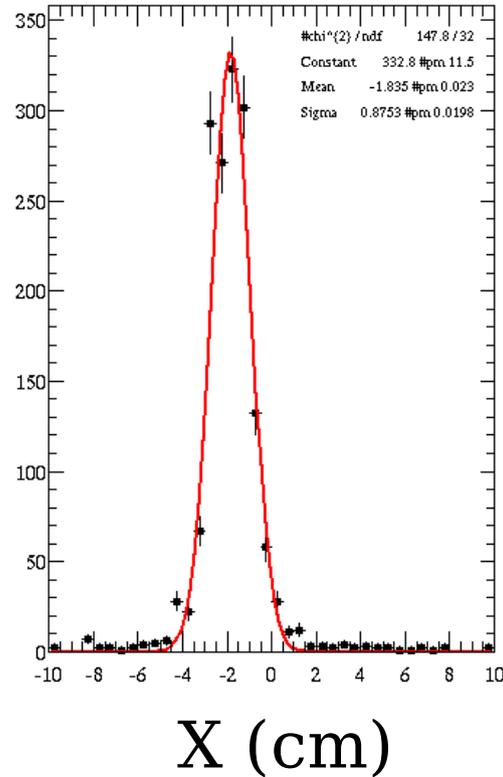
Tuning and results:

Parameters are:

- * maximum number of iteration
- * distance from pmt hit to ring <
- * minimum number of hits on a
- * ring radius > 10 cm
- * distance between ring centers



RING Centers



Results for run 8835 (target out)

Clear aliasing of PMT array seen in proton peak – work on better fit to center and radius:

[RICHRadFit](#) module

RICHRadFit

- Uses rings found by RICHCircFit
- Assumes likelihood PMT it hit is proportional to photon density at PMT face.
- Models spread in photons as gaussian of width (PMT diameter)/sqrt(12) (guess...)

$$S \sim \exp((R_{\text{pmt}}-R)/(2\sigma^2)) / R$$

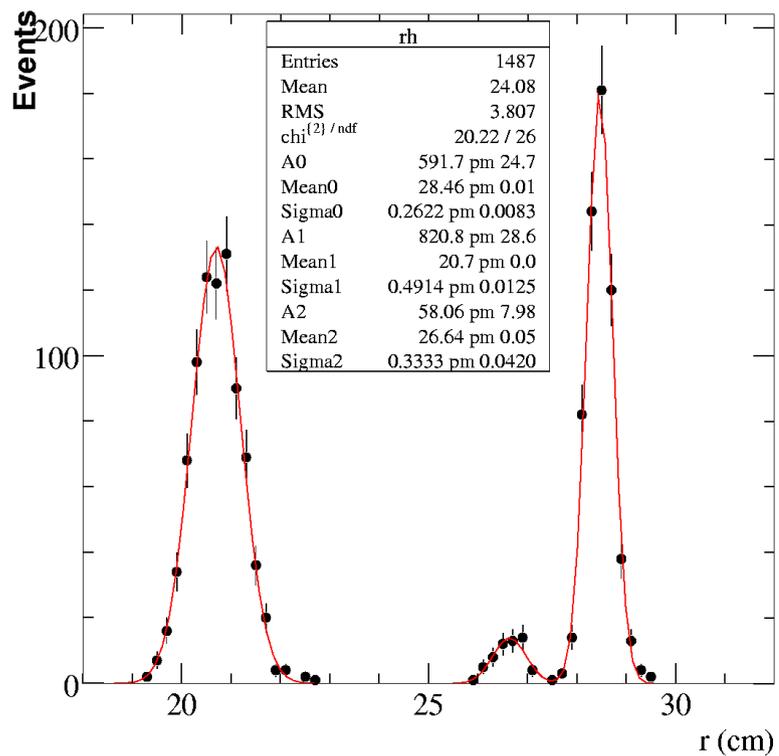
$$\chi^2 = (R_{\text{pmt}} - R)^2/\sigma^2 + \log(R_{\text{pmt}}/R)$$

$$R_{\text{pmt}}^2 = (X_{\text{pmt}} - X)^2 + (Y_{\text{pmt}} - Y)^2$$

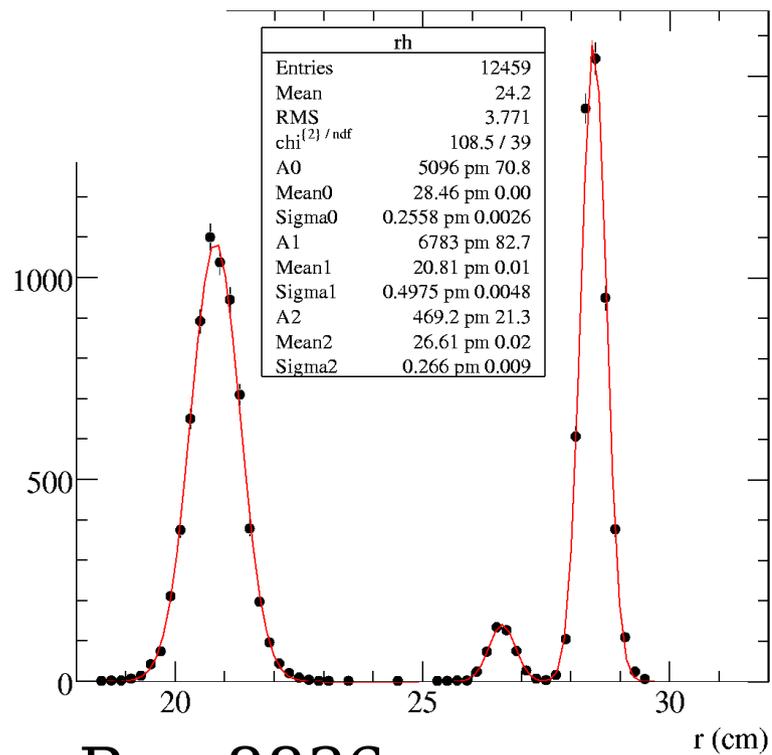
X, Y are ring center, R is radius.

Vary X, Y, R iteratively to minimize χ^2 . Use 3 iterations:

1st iteration include PMT's with $\chi^2 < 9$, 2nd $\chi^2 < 6$, $\chi^2 < 3$. Still tuning...



Run 8835



Run 8836

