

Secondary Track Finder status

Lia Lavezzi

INFN Pavia

Tracking Session

XLVI PANDA Collaboration Meeting - via SeeVogh - Bochum, 11 September 2013



Summary

≈ Presentation of the method

≈ Brief recall of the last results shown @ 22 july 2013 Computing SeeVogh
for more details, please look at

http://panda-wiki.gsi.de/pub/Computing/Minutes22July2013/SecTrkFinder_lug13.pdf

≈ Update on the new features

≈ Outlook



The method

General introduction

To reconstruct neutral particles decaying far from the IP it is necessary to write a pattern recognition **without any constraint on the position of the vertex**

- ≈ The main idea is that the primary track finder finds all the tracks coming from the IP and leaves unassigned only the secondary track (and the time overlap) hits
- ≈ at this point, the secondary track finder is applied and finds the remaining secondary tracks

- ≈ Many algorithm solutions have been tested in these months:
 - ≈ clustering (different solutions)
 - ≈ Hough transform in real plane
 - ≈ Legendre transform in conformal plane
 - ≈ association of the MVD/STT hits with different distance criteria

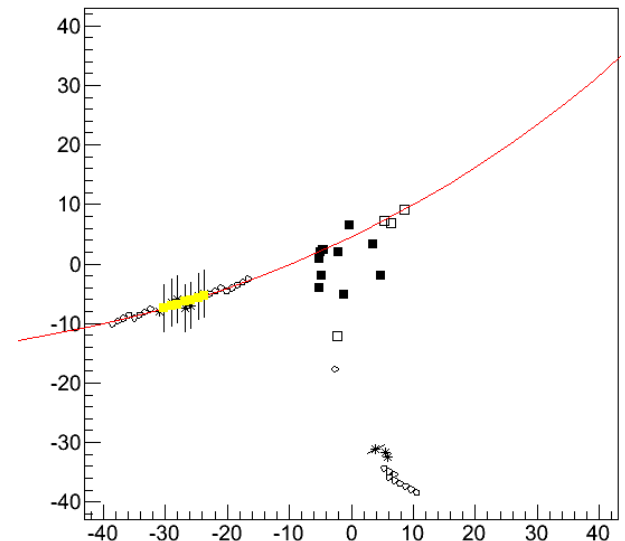
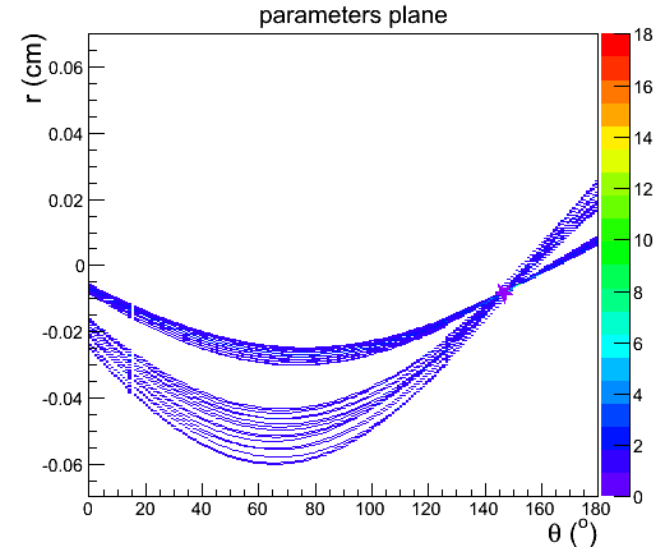
- ≈ The chosen procedure is based on the Legendre transformation in the conformal plane, with the possibility of a preliminary clustering under study

Outline of the procedure

- ≈ The problem is divided in xy projection and $z\phi$ projection
- ≈ xy projection:
 - ≈ **Conformal** & **Legendre** transformation
 - ≈ the CT transforms circular tracks into straight lines
 - ≈ the LT finds the straight line tangent to the hits belonging to the track
 - ≈ the hits are associated with distance criterion
 - ≈ the found track is refitted with a Least Square fit
- ≈ $z\phi$ projection:
 - ≈ the projections of the skewed wires are drawn in the xy plane
 - ≈ if they intercept the track they are associated to it
 - ≈ a straight line fit is performed in the $z\phi$ plane

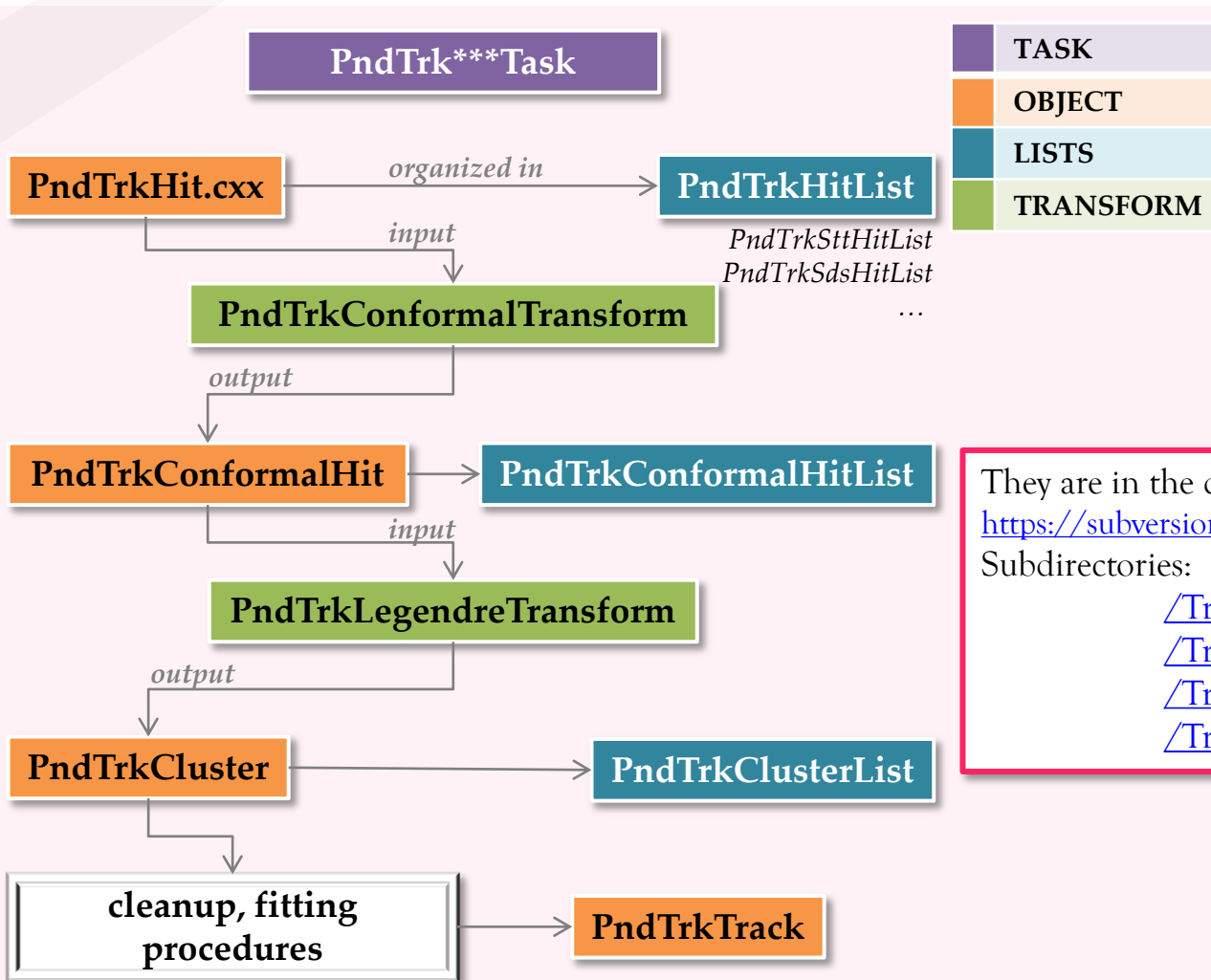
The CT needs a **translation** to a point **on** the track:

- ≈ for primaries it is the IP;
- ≈ for secondaries it is the minimum isochrone tube center



Structure of the code

Structure classes have been prepared to have a more readable, modular and OO code, with respect to the very first version used in TPC/STT decision & STT TDR studies



They are in the directory:

<https://subversion.gsi.de/fairroot/pandaroot/trunk/tracking>

Subdirectories:

[/TrkAlgo](#)

[/TrkData](#)

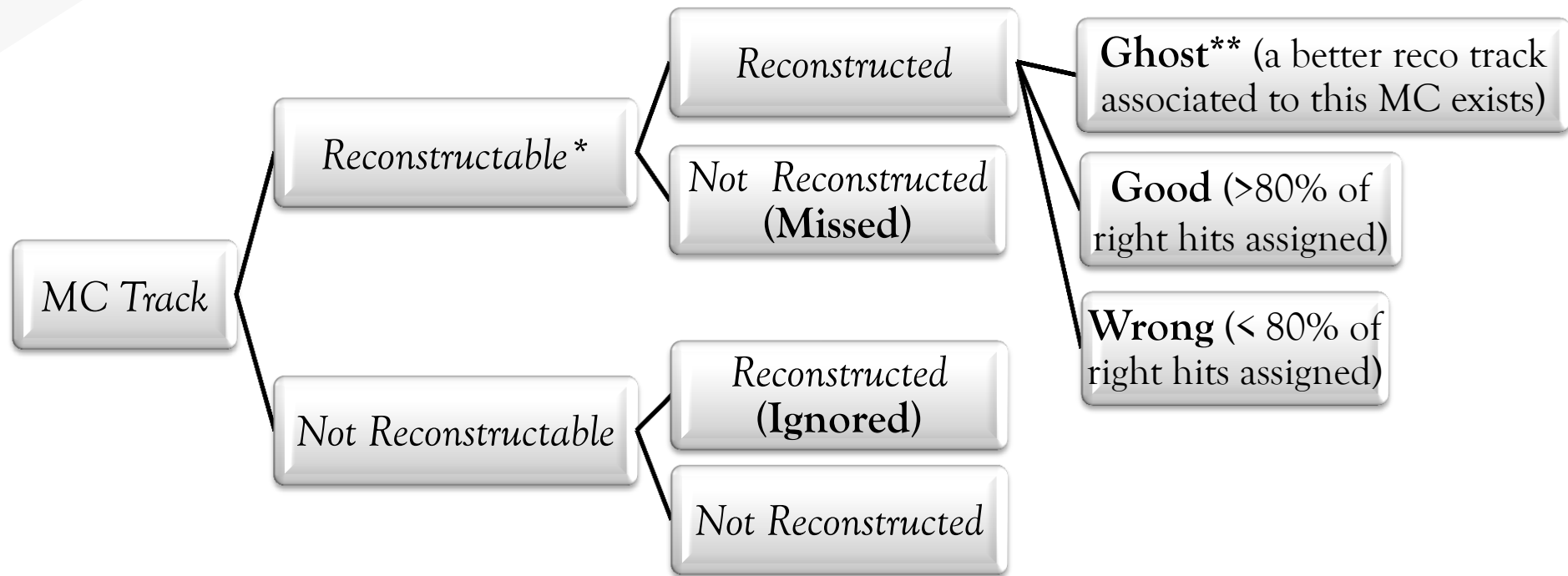
[/TrkSecondary](#)

[/TrkStructure](#)

Performance evaluation task

For the evaluation of the results, the PndTrkQATask was used:

<https://subversion.gsi.de/fairroot/pandaroot/trunk/tracking/TrkStructure/PndTrkQATask.cxx>



* «Reconstructable» means: the MC track has at least 3 parallel hits + at least 3 skew hits

** it might be better to call this **Clone**



Results @ 22 july 2013

Tests @ Computing SeeVogh Meeting

22 July 2013

Given: 1000 events, $p\bar{p}_{\text{bar}} \rightarrow \Lambda\Lambda_{\text{bar}}$, like this:

noPhotos

Decay pbarpSystem

1.0 anti-Lambda0 Lambda0 LambdaLambdaBar 1.643;

Enddecay

Decay Lambda0

1.0 p+ pi- PHSP;

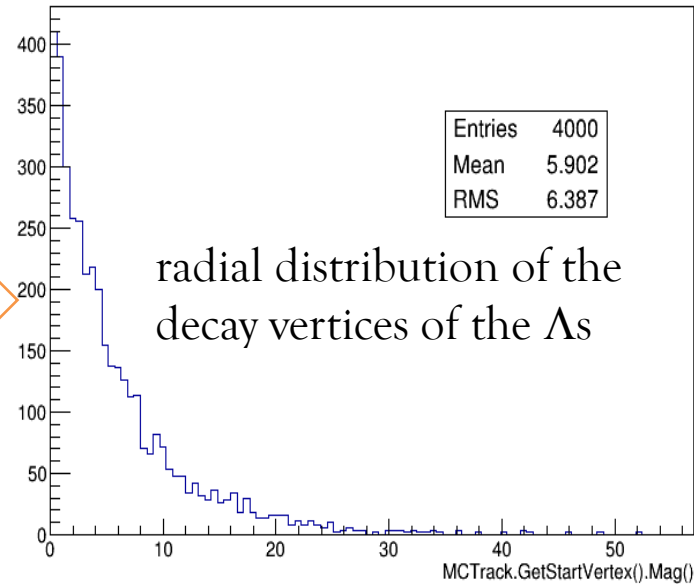
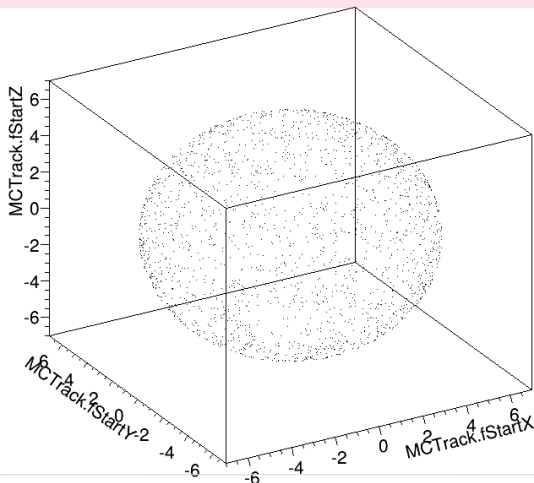
Enddecay

Decay anti-Lambda0

1.0 anti-p- pi+ PHSP;

Enddecay

End



Tests were made with 1000 μ r events generated over a sphere centered in the IP with radius = 6 cm - multiplicity = 2 - momentum = 1 GeV/c

# Tracks	p (GeV/c)	Good	Wrong	Missed	Ghost	Ignored
2	1	0.46	0.43	0.12	0.41	0.31

Improvements @ Computing SeeVogh Meeting

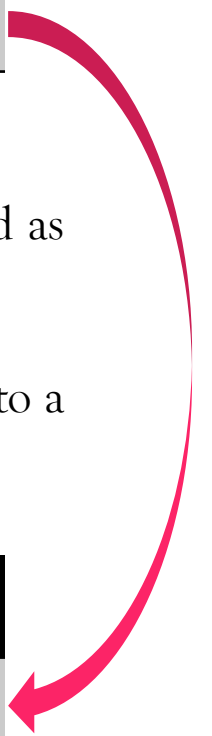
22 July 2013

	Good	Wrong	Missed	Ghost	Ignored
Very first results	0.46	0.43	0.12	0.41	0.31

It has been shown that the results are highly influenced by:

- ≈ a fine cut tuning on the hit-to-track distance (i.e. the distance considered as limit to add a hit to a track)
- ≈ preliminary clusterization (group hits in preliminary tracklet hypothesis)
- ≈ use or not of the pixel hits (due to the noise hit: it is better to add them to a track when it is well defined instead of starting from this sector)

	Good	Wrong	Missed	Ghost	Ignored
Best accomplished results	0.77	0.13	0.10	0.02	0.17





News

New procedure idea

I focused on *clusterization*, which I added as starting point of the procedure:

≈ I - Create full clusterization

≈ II - Legendre

≈ III - Refinement

The clusterization aim is to clean the Legendre accumulation plot to enhance the peak height and suppress the probability of fake peaks

In an ideal case, where with the clusterization I gather in one cluster all and only the hits coming from one track, the Legendre procedure serves only as a fit

The task with this code can be found on *svn* at

<https://subversion.gsi.de/fairroot/pandaroot/trunk/tracking/TrkSecondary/PndTrkLegendreNew.cxx>

New procedure idea: clusterization

≈ I - **Create full clusterization:** all the hits belonging to *neighboring* tubes are grouped together (in the full STT, no matter if the hits belong to skewed or parallel tubes)

New procedure idea: clusterization

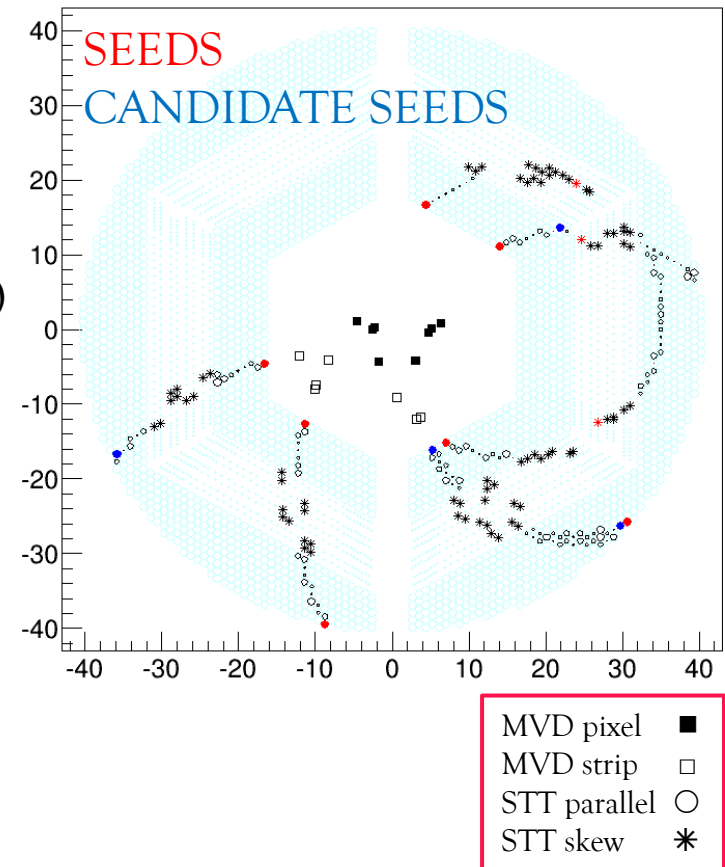
≈ **I - Create full clusterization:** all the hits belonging to *neighboring* tubes are grouped together (in the full STT, no matter if the hits belong to skewed or parallel tubes)

I.1 Hit map creation

Create a map of hits (PndTrkNeighboringMap):

≈ loop over all hits and find:

- ≈ tubes with no neighborings (will not use them)
- ≈ tubes with only 1 neighboring (will serve as **seeds**)
- ≈ tubes with only 2 neighborings, one of which is on the same layer and has no neighboring (will serve as **candidates** to be seeds, if needed)
- ≈ fill the map of hit ↔ existing neighboring in the STT hit list



New procedure idea: clusterization

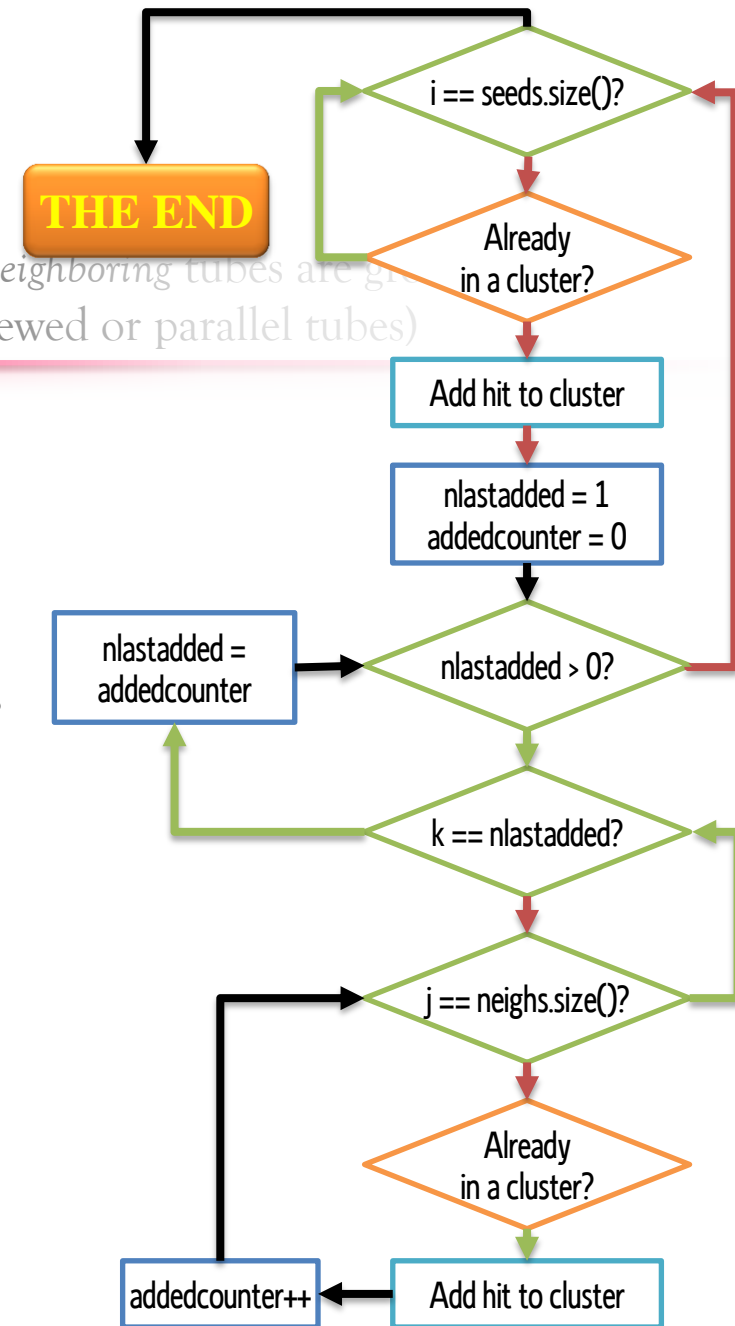
≈ **I - Create full clusterization:** all the hits belonging to *neighboring tubes* are grouped together (in the full STT, no matter if the hits belong to *skewed or parallel tubes*)

I.2 Actual clusterization

Loop over seeds in the map of hits:

≈ if the seed has not been used yet → start a new cluster
≈ start iterations and, at each iteration, loop over the hits added to the cluster at the previous one and add all their neighborings for next iteration

≈ if necessary also the candidate seeds can be used as starting points



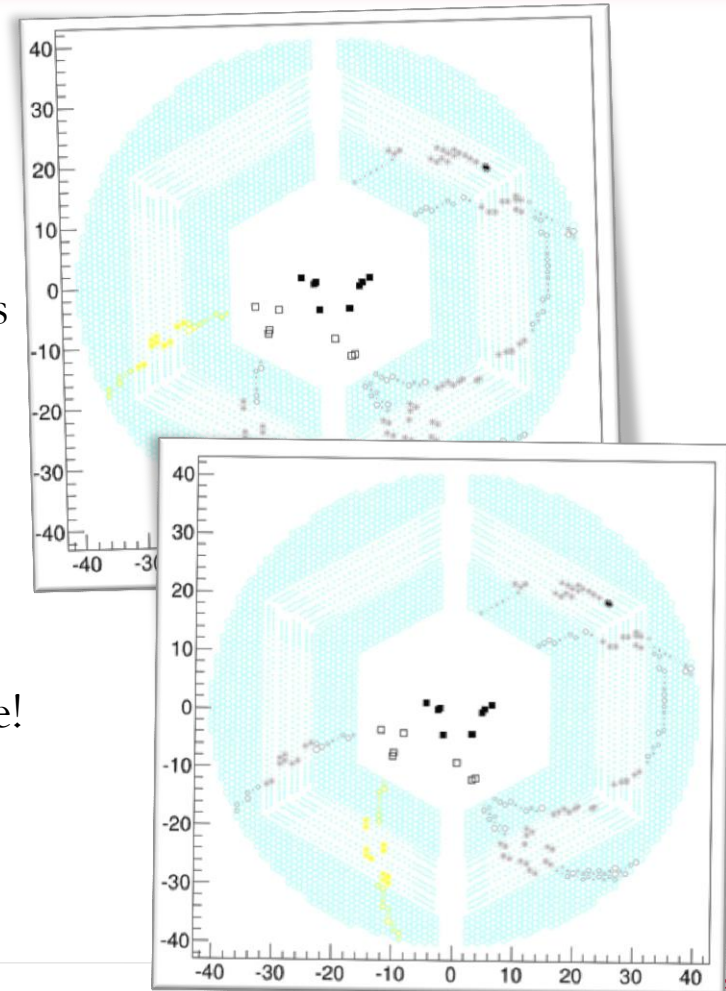
New procedure idea: clusterization

≈ **I - Create full clusterization:** all the hits belonging to *neighboring* tubes are grouped together (in the full STT, no matter if the hits belong to skewed or parallel tubes)

I.2 Actual clusterization

Loop over seeds in the map of hits:

- ≈ if the seed has not been used yet → start a new cluster
- ≈ start iterations and, at each iteration, loop over the hits added to the cluster at the previous one and add all their neighborings for next iteration
- ≈ if necessary also the candidate seeds can be used as starting points
- ≈ if there are clean tracks, the clustering procedure is fine!



New procedure idea: clusterization

≈ **I - Create full clusterization:** all the hits belonging to *neighboring* tubes are grouped together (in the full STT, no matter if the hits belong to skewed or parallel tubes)

I.2 Actual clusterization

Loop over seeds in the map of hits:

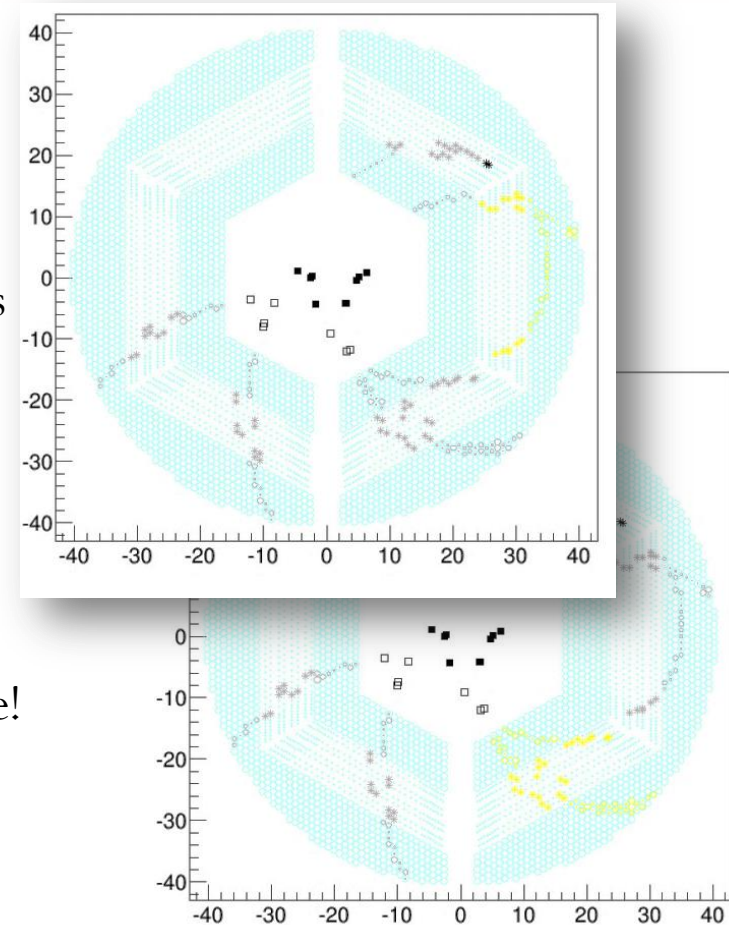
≈ if the seed has not been used yet → start a new cluster

≈ start iterations and, at each iteration, loop over the hits added to the cluster at the previous one and add all their neighborings for next iteration

≈ if necessary also the candidate seeds can be used as starting points

≈ if there are clean tracks, the clustering procedure is fine!

≈ if two tracks are too close, one cluster can be formed out of two or more tracks



New procedure idea: Legendre

≈ **I - Create full clusterization:** all the hits belonging to *neighboring* tubes are grouped together (in the full STT, no matter if the hits belong to skewed or parallel tubes)

≈ **II - Legendre:**

≈ infer the number of effective tracks in each cluster \rightarrow *nof_hypothetic_tracks*

New procedure idea: Legendre

≈ I - Create full clusterization: all the hits belonging to *neighboring* tubes are grouped together (in the full STT, no matter if the hits belong to skewed or parallel tubes)

≈ II - Legendre:

≈ infer the number of effective tracks in each cluster → *nof_hypothetic_tracks*

II.1 Count track in cluster

≈ We want to infer how many tracks are actually there in the cluster and we use the **skew sector** layers

≈ # skewed on a layer - # neighboring couples = # tracks

Example: 1 cluster with 3 tracks

≈ *** * ** are tubes no.: 0 1 2 3 4

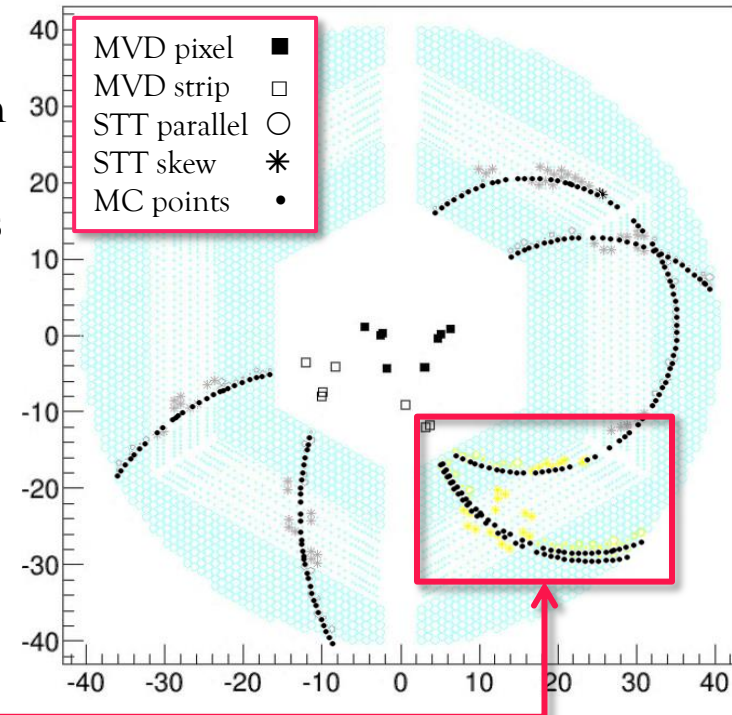
≈ *nof tubes on the layer* = 5

≈ calculation of the couples of neighborings:

0/1 | 3/4

≈ so: *nof neighboring couples* = 2

≈ → *noftubes* (5) - *nofcouples* (2) = *noftracks* (3)



New procedure idea: Legendre & Refinement

≈ I - **Create full clusterization:** all the hits belonging to *neighboring* tubes are grouped together (in the full STT, no matter if the hits belong to skewed or parallel tubes)

≈ II - **Legendre:**

≈ infer the number of effective tracks in each cluster → *nof_hypothetic_tracks*

≈ iterate the following procedure *nof_hypothetic_tracks* times (at least) :

≈ port the hits of the cluster to the conformal plane

≈ fill the Legendre parameter plane and extract the track parameters (fit)

≈ create a cluster around the found track

≈ if more than 15 hit remain unassigned → increase *nof_hypothetic_tracks* by 1 and repeat the procedure

≈ III - **Refinement:**

≈ port the hits of the cluster to the conformal plane

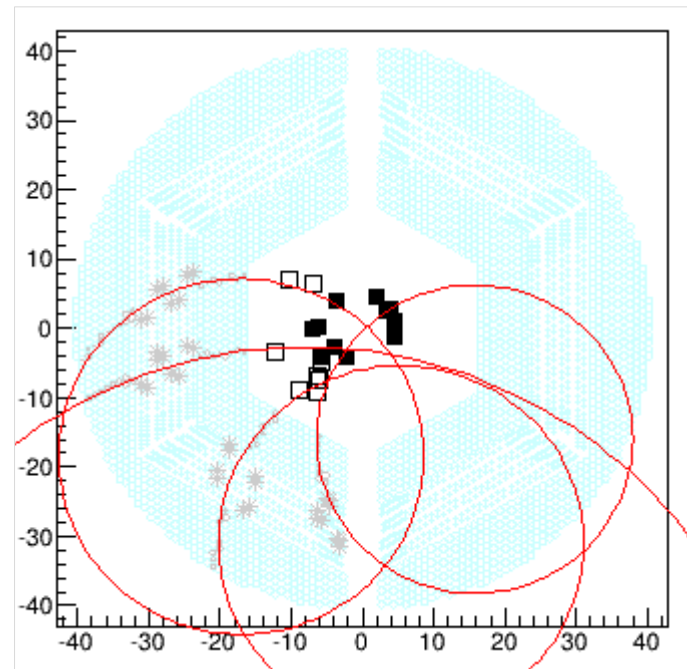
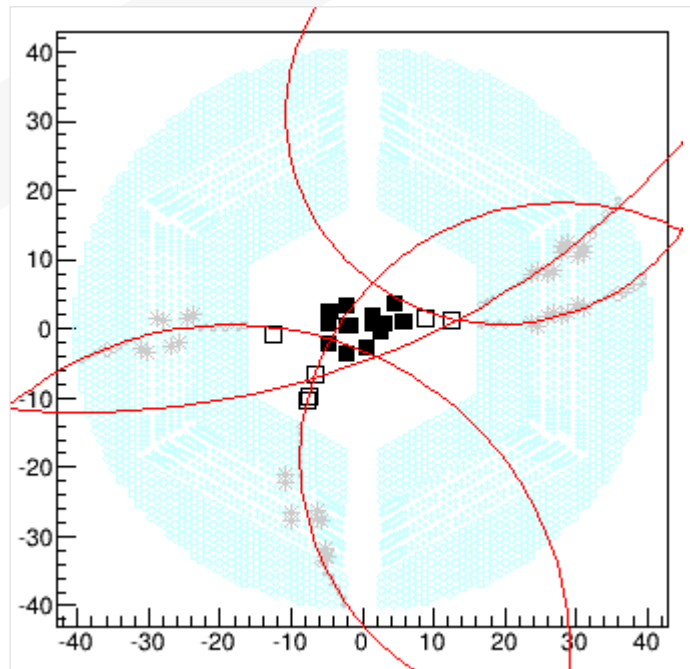
≈ apply Legendre procedure again Legendre fit

≈ create a cluster around the found track

≈ finalize it with an analytical LS track fit

...this is the procedure so far

Some pictures: the good ones first



MVD pixel	■
MVD strip	□
STT parallel	○
STT skew	*
Found tracks	—

Sample:

4 μ tracks

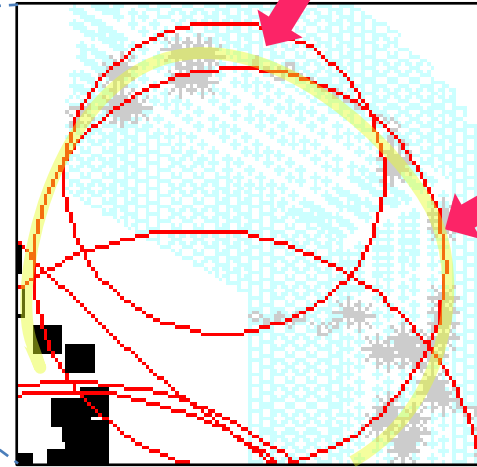
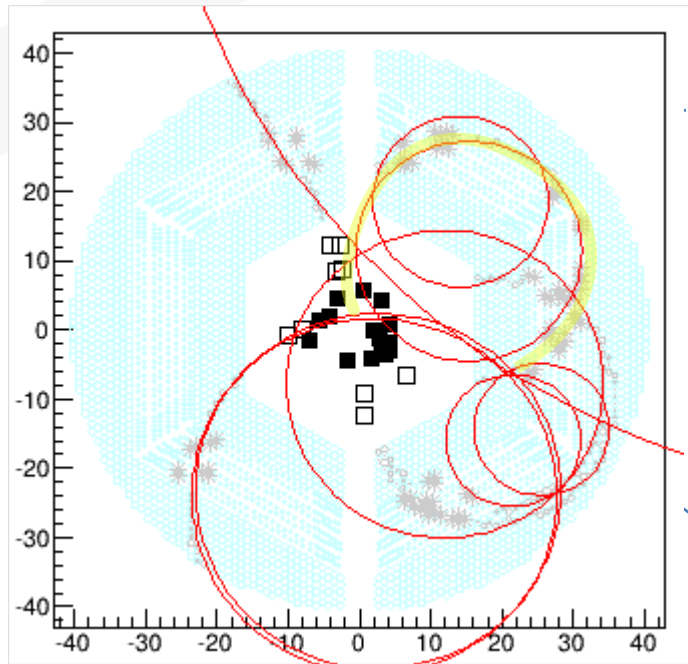
vtx @ 6 cm from IP

$multiplicity = 4$

$p \in [0.1, 0.3]$ GeV/c

- ≈ In some cases, with clean tracks, this procedure finds all and only the true tracks
- ≈ In these cases it should be quite easy to complete the search for the track parameters with the z information, using the skewed tubes (the procedure is ready, but since it depends on the xy track projection, I am trying to improve that first)
- ≈ The MVD hits can be added if the xy projection is fine

Some pictures: the problem of the clones



MVD pixel	■
MVD strip	□
STT parallel	○
STT skew	*
Found tracks	—

Sample:

4 μ tracks

vtx @ 6 cm from IP

multiplicity = 4

$p \in [0.1, 0.3]$ GeV/c

≈ In certain cases (see the highlighted circle in pictures), e.g.:

≈ looping tracks

≈ full circle tracks

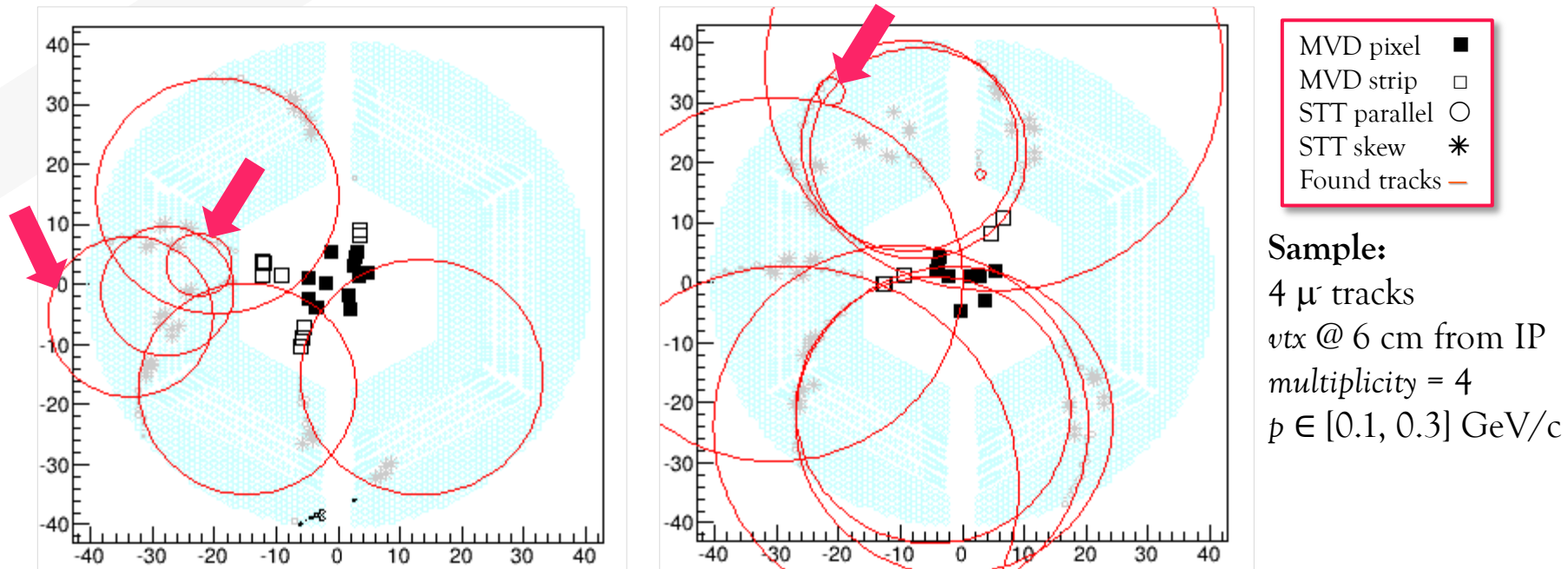
≈ tracks with a missing hit

≈ ...

the initial clustering is not perfect and the track is broken in two pieces, clones may show up

→ it is necessary to recognize them and get rid of them

Some pictures: the problem of the fakes



- ≈ In the Legendre peak finding, some fake peak may show up due to some incidental alignment of hits belonging to different tracks → fake tracks show up
- ≈ A procedure to clean them must be adopted: some clean up functions are already there, but they must be applied in a procedure
- ≈ The fake tracks usually are recognizable due to:
 - ≈ low momentum (very small circles → possible cut off)
 - ≈ holes in the hit list




Outlook

The TODO list

- ≈ preliminary tests with primaries
- ≈ for primary tracks:
 - ≈ cut fine tuning
 - ≈ cleanup
 - ≈ test
- ≈ for secondary tracks:
 - ≈ test
 - ≈ most probably some parts will require changes/adaptations
- ≈ primary goal: the best possible efficiency
- ≈ side goals:
 - ≈ fast computing speed
 - ≈ acceptable momentum resolution
- ≈ an online application can be foreseen, but the code needs changes to do so, since now the procedure is sequential

After 22 july 2013 tests:

- ≈ return to clusterization 
- ≈ lambdas have low momenta → looping particles
- ≈ obviously also time based simulation and time overlap must be added

Next

- ⌘ The **clustering** seems a solution to have less hits in the Legendre accumulation plane
- ⌘ Some **cleanup** functions are already there: a full cleanup must be completed to get the best possible xy projection hypothesis; later the skewed association can be performed (it is already written, but must be added to the last code)
- ⌘ For the moment the idea is to concentrate on **non looping particles** and improve their results

Thank you for your attention



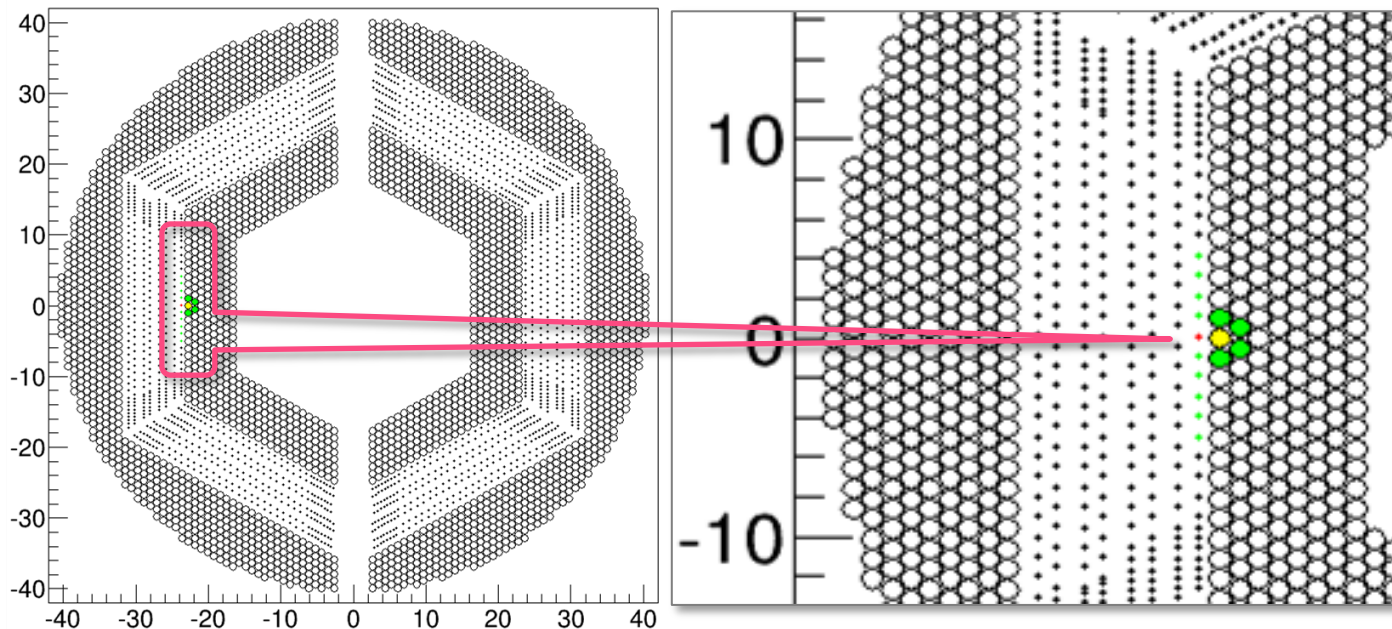
Backup

Neighboring tubes finder

PndSttGeometryMap

- ≈ Integration between PndSttStrawMap class and /development/lia/stt/PndSttMapCreator functions to create a map of the STT and retrieve layerID, sectorID, neighboring tubes.
- ≈ This class is called directly inside PndSttMapCreator and fills the PndSttTube with the relevant info.
- ≈ **The user does not need to call this class but can retrieve all the info in the PndSttTube obj** after having filled the fTubeArray via the PndSttMapCreator.

Authors: L. Lavezzi, M. Mertens 2013

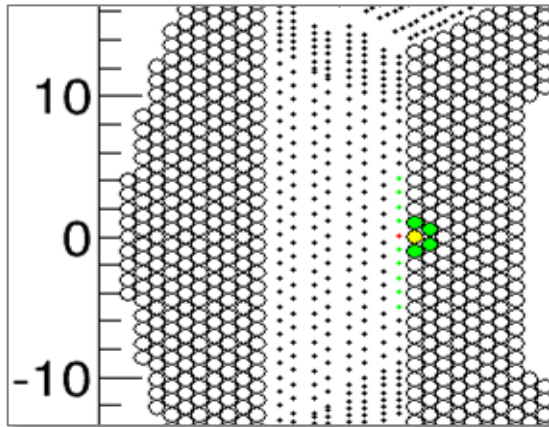


Yellow: tube 891

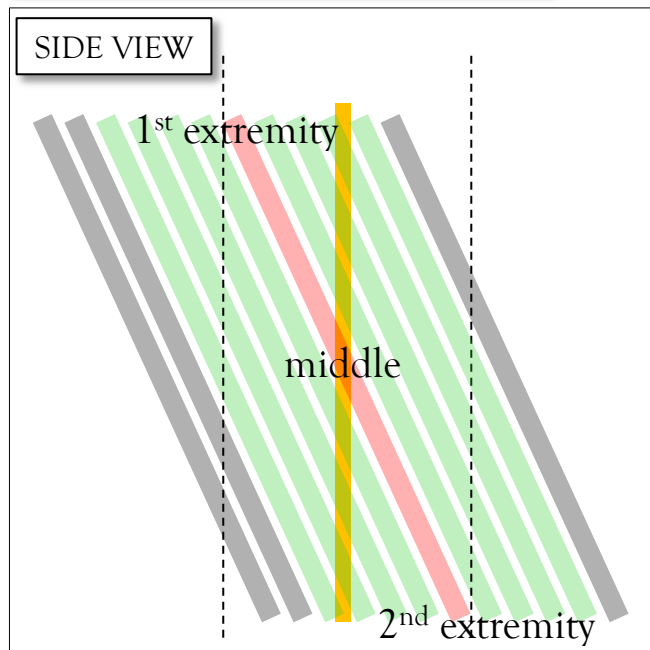
Green: parallel and skew neighboring tubes

Red: let's consider this skew tube (middle of the wire here)

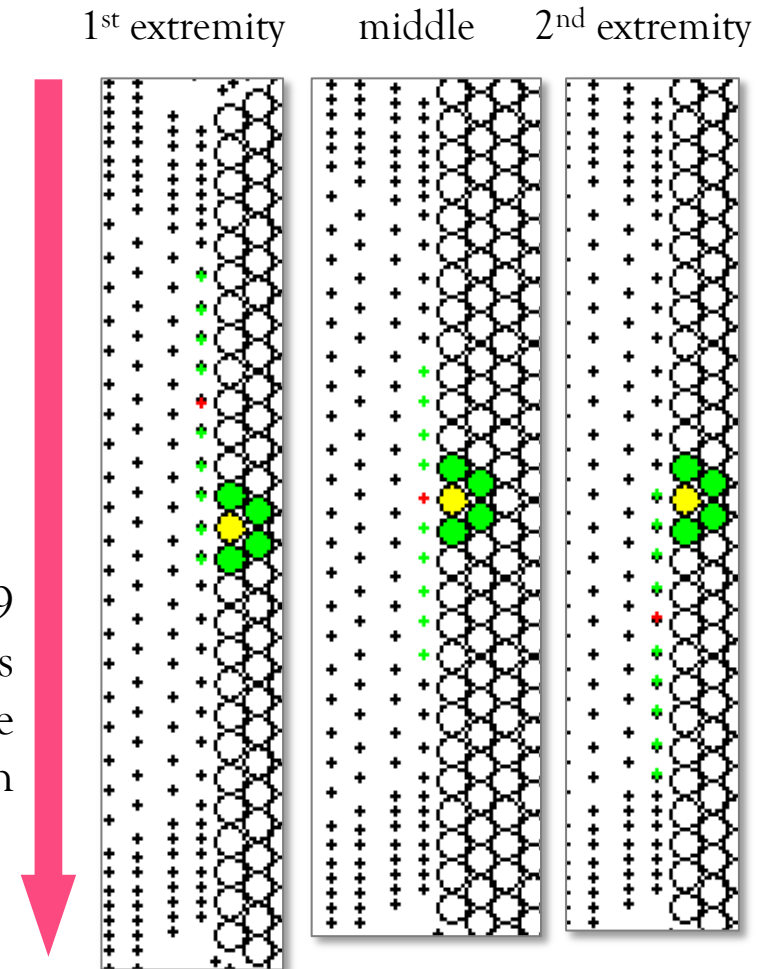
Neighboring tubes finder



The **red** dot is the tube with center in front of the **yellow** tube:
 1° and 2° extremities tell us how many tubes cross the yellow one (**green**) and which do not (black)



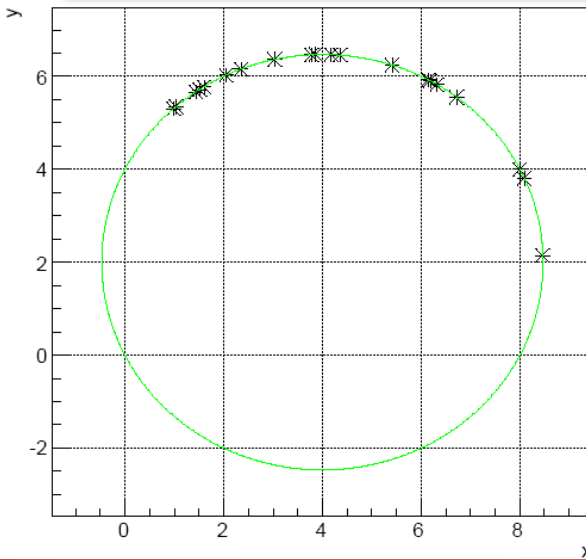
Red dot moves by 9 positions → this means that 9 skew tubes are neighborings of each parallel tube



Conformal transformation

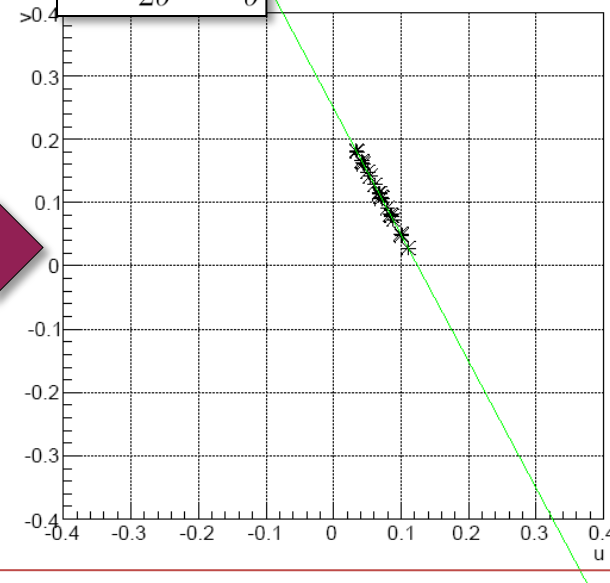
circle

$$(x - a)^2 + (y - b)^2 = r^2 = a^2 + b^2$$



$$v = \frac{1}{2b} - u \frac{a}{b}$$

straight line



points (x, y)

$$u = \frac{x}{x^2 + y^2}, \quad v = \frac{y}{x^2 + y^2},$$

circles (x, y, r_d)

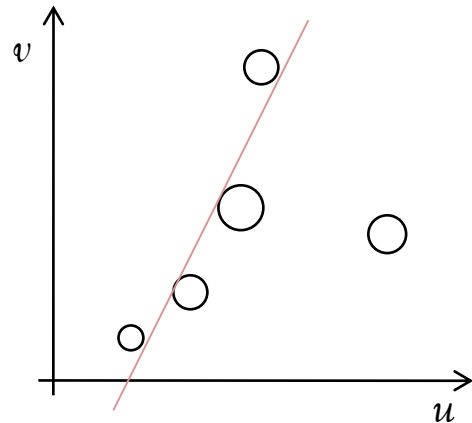
$$u = \frac{x}{x^2 + y^2 - r_d^2}, \quad v = \frac{y}{x^2 + y^2 - r_d^2},$$

$$r_c = \frac{r_d}{x^2 + y^2 - r_d^2}$$

≈ if the track does not stem from the IP → make a translation onto a precise hit, such as an MVD, a SciTil or the center of the STT hit

Legendre transformation

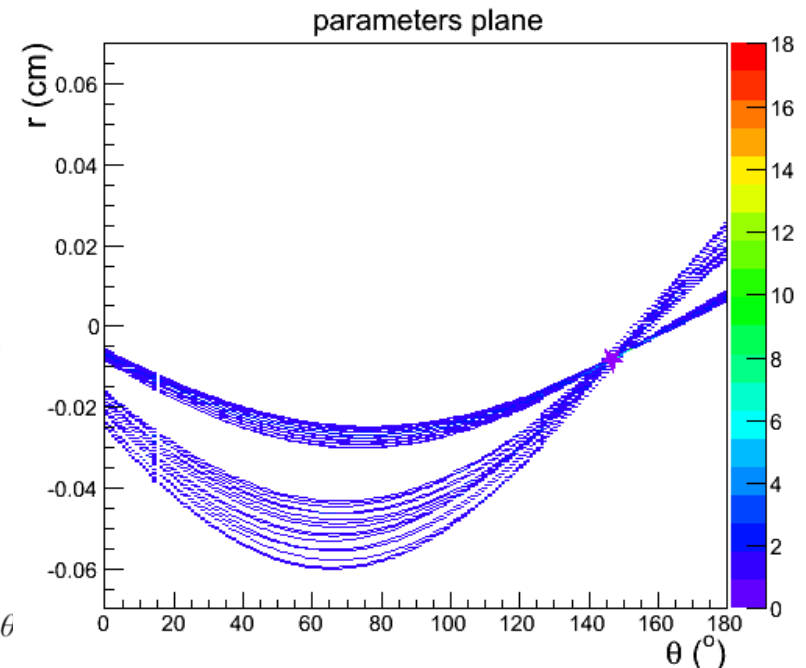
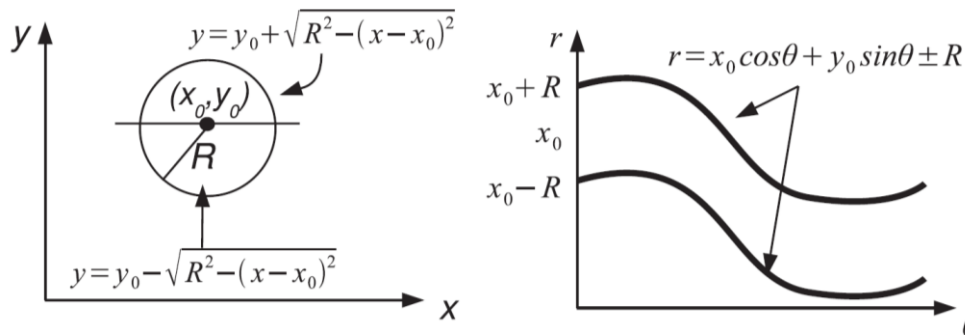
≈ we need to find a straight line tangent to all the circles belonging to a track



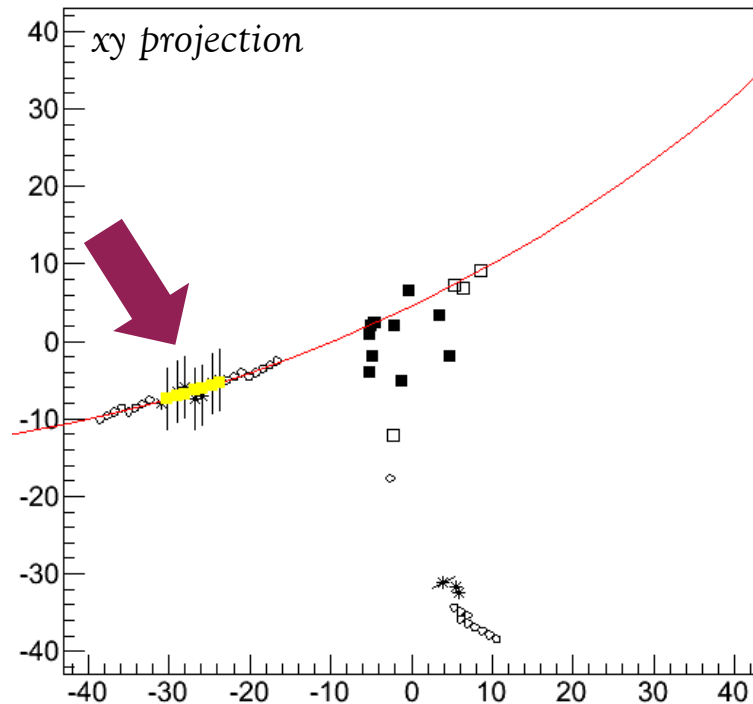
≈ for each hit we get two sinusoidal curve

≈ the accumulation plot is filled and the peak corresponding to a track is found

Alexopoulos T. et al, NIM A 592 (2008) 456– 462



Skewed tubes association



≈ they correspond to two different ϕ and z values

