

# Quality Assurance for Trackers: a proposal

Lia Lavezzi, Stefano Spataro  
University of Torino & INFN - Torino

LII PANDA Collaboration Meeting – Giessen, 16 – 20 March 2015  
Pattern Recognition Day



# QA procedure

## OUTPUT

0. REAL TRACK FINDER/PR TASK

RecoTrack  
RecoTrackCand

*PndTrack*  
*PndTrackCand*



1. IDEAL TRACK FINDER TASK

IdealTrack  
IdealTrackCand

*PndTrack*  
*PndTrackCand*

2. MC TRACK ASSOCIATOR TASK

RecoTrackID  
IdealTrackID

*PndTrackID*  
*PndTrackID*

3. QUALITY ASSURANCE TASK

MCTrackInfo  
RecoTrackInfo

*PndTrkMCTrackInfo*  
*PndTrkRecoTrackInfo*



4. QA MACRO

fills  $n$ -tuples/histograms w/ different cuts by MCTrackInfo/RecoTrackInfo

# Ideal Track Finder

# Ideal Track Finder

source: [pandaroot](#) / [trunk](#) / [sttmvdtracking](#) @ 27138

Name ▲
../
CMakeLists.txt
PndLambdaIM.cxx
PndLambdaIM.h
PndMixBackgroundEvents.cxx
PndMixBackgroundEvents.h
PndMvdSttGemRiemannTrackFinder.cxx
PndMvdSttGemRiemannTrackFinder.h
PndSecondaryTrackFinder.cxx
PndSecondaryTrackFinder.h
PndSttMvdGemTracking.cxx
PndSttMvdGemTracking.h
PndSttMvdGemTrackingIdeal.cxx
PndSttMvdGemTrackingIdeal.h
SttMvdTrackingLinkDef.h



## PndSttMvdGemTrackFinderIdeal

- ❖ It can be used if the cuts present in it are parametrized and possibly removed
- ❖ The only request is that the track must be **charged** and must leave **at least one signal** in one of the trackers
- ❖ The OUTPUT is a TCA of Ideal PndTrack & PndTrackCand corresponding to the PndMCTrack TCA read directly from the simulation

MC Track Associator

# MC Track Associator

source: [pandaroot](#) / [trunk](#) / [PndMCMATCH](#) @ 27138

**PndMCTrackAssociator**

Name ▲
../
examples
CMakeLists.txt
PndMCDataCrawler.cxx
PndMCDataCrawler.h
PndMCEntry.cxx
PndMCEntry.h
PndMCList.cxx
PndMCList.h
PndMCMatch.cxx
PndMCMatch.h
PndMCMatchCreatorTask.cxx
PndMCMatchCreatorTask.h
PndMCMatchExamplesLinkDef.h

PndMCMatchLinkDef.h
PndMCMatchLoaderTask.cxx
PndMCMatchLoaderTask.h
PndMCMatchSelectorTask.cxx
PndMCMatchSelectorTask.h
PndMCObject.cxx
PndMCObject.h
PndMCResult.cxx
PndMCResult.h
PndMCStage.cxx
PndMCStage.h
PndMCTrackAssociator.cxx
PndMCTrackAssociator.h
PndMCTrackEnumAssociator.cxx
PndMCTrackEnumAssociator.h

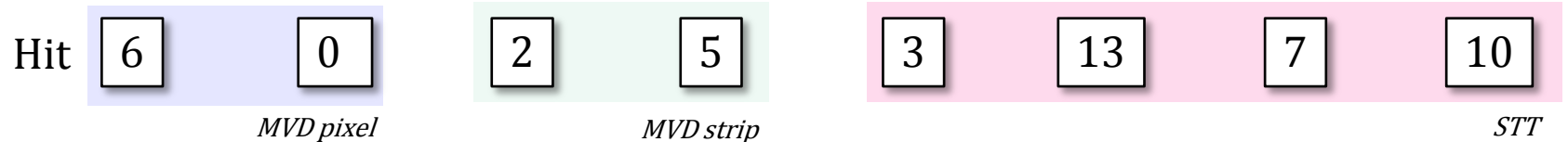


- ❖ The hits coming from the different detectors and associated to the different MC tracks are counted and the track is associated to the  $n$  MC tracks with a given *multiplicity* (i.e. how many hits voted for this MC track)
- ❖ The reco track is associated to the MC track which the **majority of the hits** belong to.

# MC Track Associator

## PndMCTrackAssociator

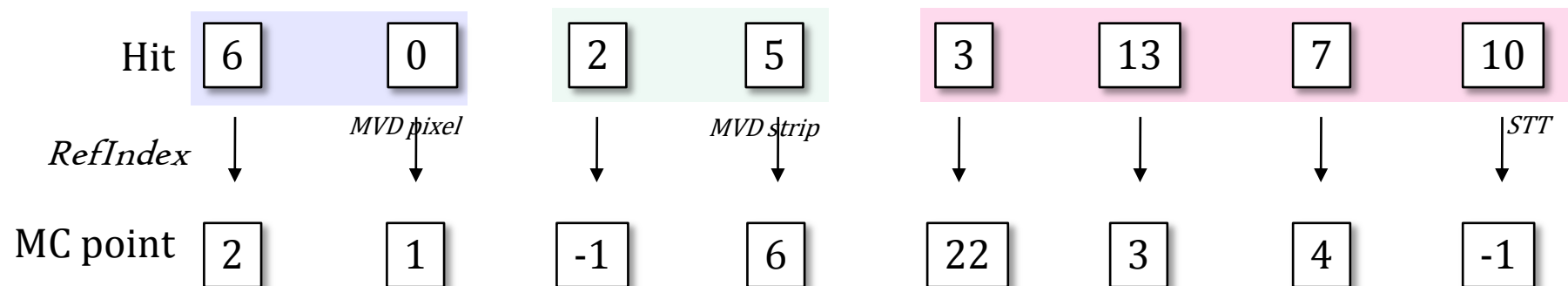
- ❖ The hits coming from the different detectors and associated to the different MC tracks are counted and the track is associated to the  $n$  MC tracks with a given *multiplicity* (i.e. how many hits voted for this MC track)
- ❖ The reco track is associated to the MC track which the **majority of the hits** belong to.



# MC Track Associator

## PndMCTrackAssociator

- ❖ The hits coming from the different detectors and associated to the different MC tracks are counted and the track is associated to the n MC tracks with a given multiplicity (i.e. how many hits voted for this MC track)
- ❖ The reco track is associated to the MC track which the majority of the hits belong to.

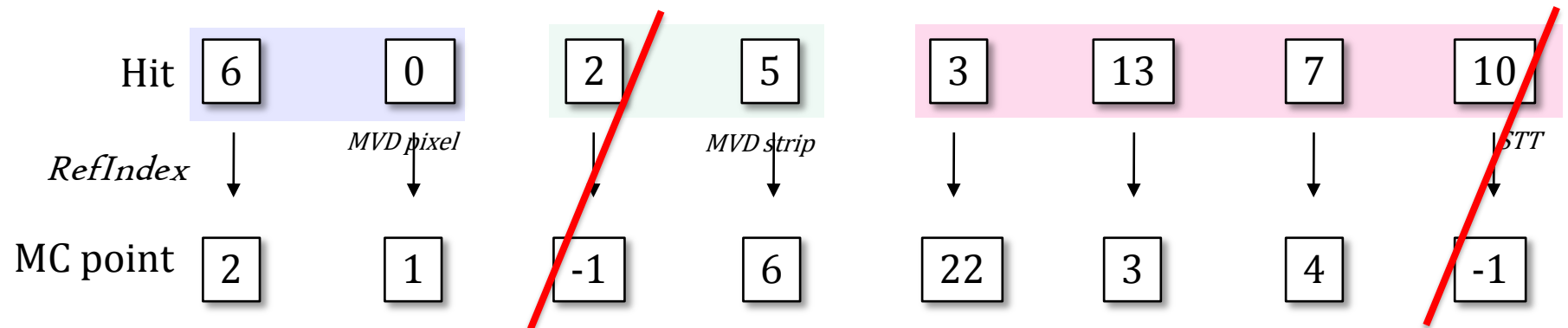




# MC Track Associator

## PndMCTrackAssociator

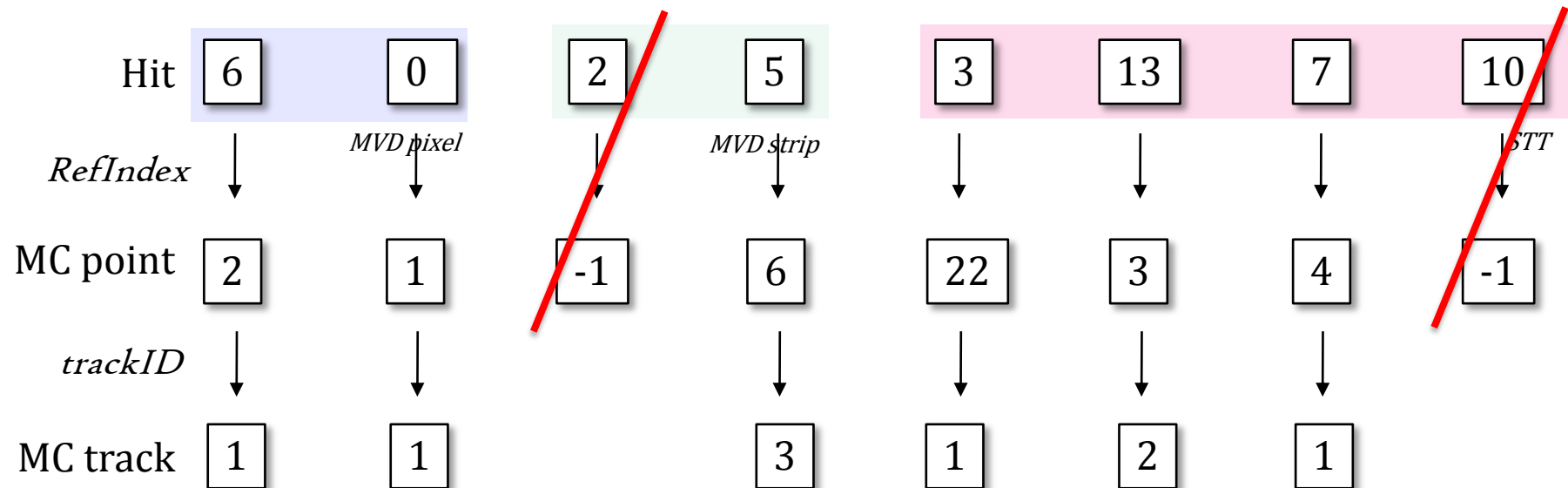
- ❖ The hits coming from the different detectors and associated to the different MC tracks are counted and the track is associated to the n MC tracks with a given multiplicity (i.e. how many hits voted for this MC track)
- ❖ The reco track is associated to the MC track which the majority of the hits belong to.



# MC Track Associator

## PndMCTrackAssociator

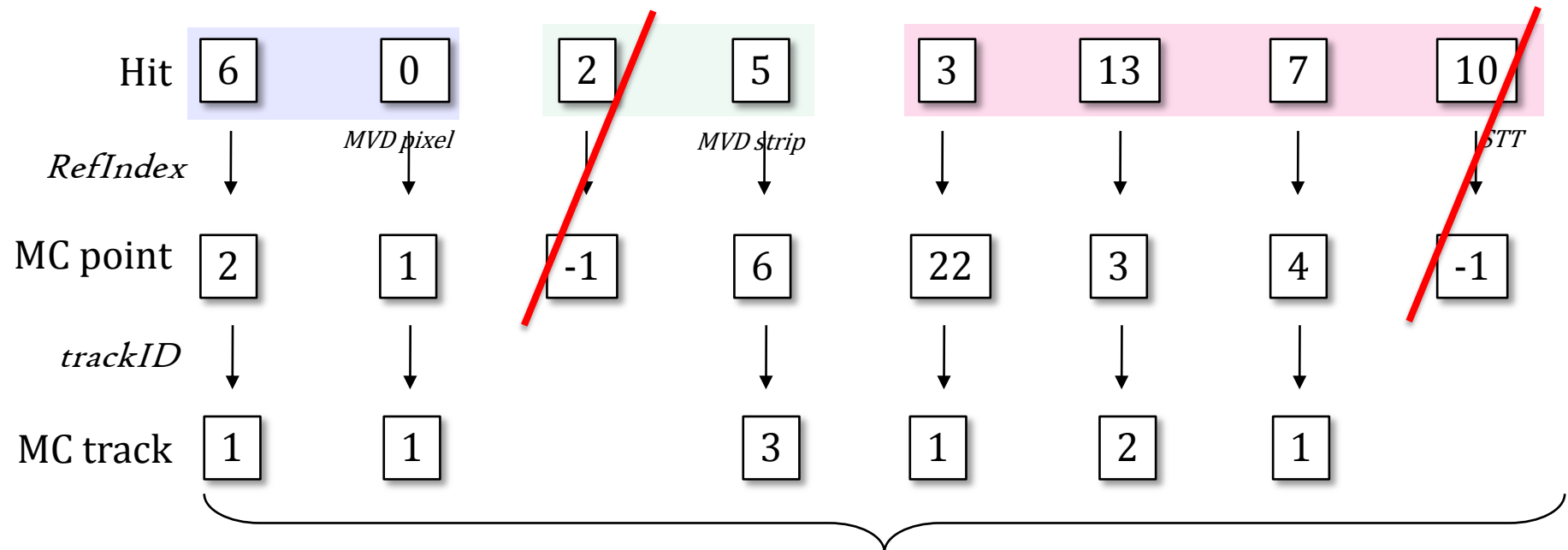
- ❖ The hits coming from the different detectors and associated to the different MC tracks are counted and the track is associated to the n MC tracks with a given multiplicity (i.e. how many hits voted for this MC track)
- ❖ The reco track is associated to the MC track which the majority of the hits belong to.



# MC Track Associator

## PndMCTrackAssociator

- ❖ The hits coming from the different detectors and associated to the different MC tracks are counted and the track is associated to the n MC tracks with a given multiplicity (i.e. how many hits voted for this MC track)
- ❖ The reco track is associated to the MC track which the majority of the hits belong to.



MC track 1 with multiplicity 4  
3 with multiplicity 1  
2 with multiplicity 1

# MC Track Associator

PndMCTrackAssociator

*more reco tracks can be associated to the same MC track*

*If more reco tracks are associated to the same MC track,  
then only one is **true** and the others are **clones***

- ❖ **True track** – the reco track with highest # *true* hits.
  - If two reco tracks have the same # *true* hits
    - the reco track with lowest number of *fake* hits is chosen.
  - If two reco tracks have the same # *true* and # *fake* hits
    - the reco track with lowest number of *missing* hits is picked
- ❖ **Clone track** – the other reco tracks associated to the MC track which are not the *true* track

# True, Fake, Missing Hits

- ❖ **True hit** - a hit associated to my reco track, which has a *RefIndex* pointing to a MC point belonging to the right MC track
- ❖ **Fake hit** - a hit associated to the reco track, which has a *RefIndex* pointing to a MC point belonging to the wrong MC Track *or* has *RefIndex* = -1 (→ background)
- ❖ **Missing hit** - a hit *not* associated to the reco track, which has a *RefIndex* pointing to a MC point belonging to the right track, i.e. a hit not associated to the reco track when it should be!

---

RECO Track associated to MC Track = 1

Hit	6	0	2	5	3	13	7	10
-----	---	---	---	---	---	----	---	----

# True, Fake, Missing Hits

- ❖ **True hit** - a hit associated to my reco track, which has a *RefIndex* pointing to a MC point belonging to the right MC track
- ❖ **Fake hit** - a hit associated to the reco track, which has a *RefIndex* pointing to a MC point belonging to the wrong MC Track *or* has *RefIndex* = -1 (→ background)
- ❖ **Missing hit** - a hit *not* associated to the reco track, which has a *RefIndex* pointing to a MC point belonging to the right track, i.e. a hit not associated to the reco track when it should be!

RECO Track associated to MC Track = 1

Hit	6	0	2	5	3	13	7	10
<i>RefIndex</i>	↓	↓	↓	↓	↓	↓	↓	↓
MC point	2	1	-1	6	22	3	4	-1

# True, Fake, Missing Hits

- ❖ **True hit** - a hit associated to my reco track, which has a *RefIndex* pointing to a MC point belonging to the right MC track
- ❖ **Fake hit** - a hit associated to the reco track, which has a *RefIndex* pointing to a MC point belonging to the wrong MC Track *or* has *RefIndex* = -1 (→ background)
- ❖ **Missing hit** - a hit *not* associated to the reco track, which has a *RefIndex* pointing to a MC point belonging to the right track, i.e. a hit not associated to the reco track when it should be!

RECO Track associated to MC Track = 1

Hit	6	0	2	5	3	13	7	10
<i>RefIndex</i>	↓	↓	↓	↓	↓	↓	↓	↓
MC point	2	1	-1	6	22	3	4	-1



*fake*



*true*

# True, Fake, Missing Hits

- ❖ **True hit** - a hit associated to my reco track, which has a *RefIndex* pointing to a MC point belonging to the right MC track
- ❖ **Fake hit** - a hit associated to the reco track, which has a *RefIndex* pointing to a MC point belonging to the wrong MC Track *or* has *RefIndex* = -1 (→ background)
- ❖ **Missing hit** - a hit *not* associated to the reco track, which has a *RefIndex* pointing to a MC point belonging to the right track, i.e. a hit not associated to the reco track when it should be!

RECO Track associated to MC Track = 1

Hit	6	0	2	5	3	13	7	10
<i>RefIndex</i>	↓	↓	↓	↓	↓	↓	↓	↓
MC point	2	1	-1	6	22	3	4	-1
<i>trackID</i>	↓	↓		↓	↓	↓	↓	
MC track	1	1		3	1	2	1	



*fake*



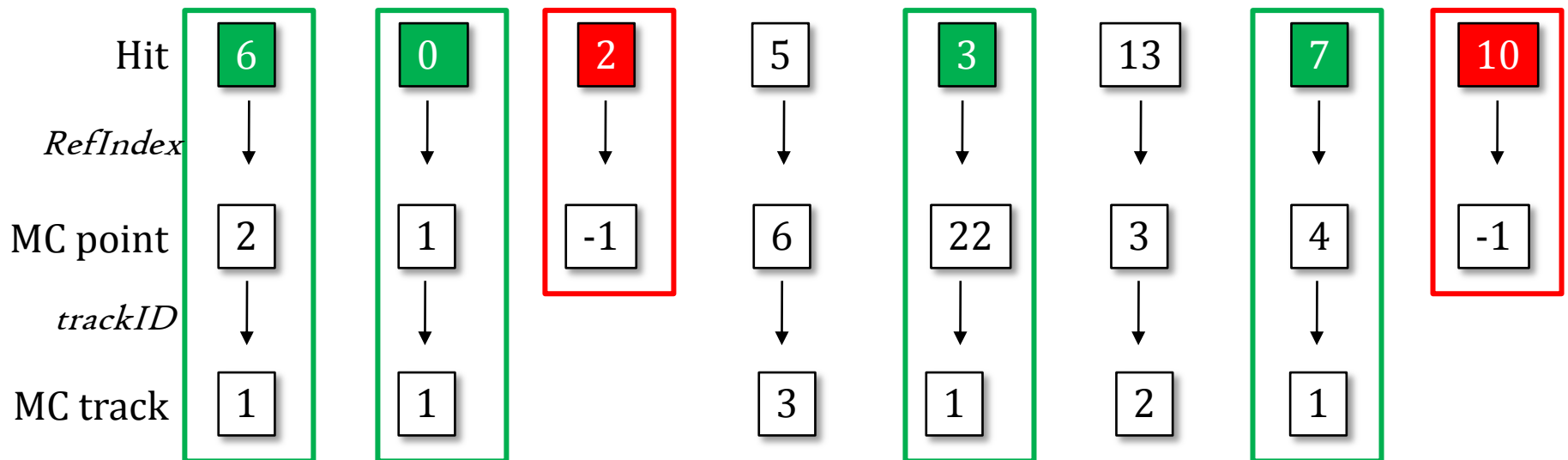
*true*



# True, Fake, Missing Hits

- ❖ **True hit** - a hit associated to my reco track, which has a *RefIndex* pointing to a MC point belonging to the right MC track
- ❖ **Fake hit** - a hit associated to the reco track, which has a *RefIndex* pointing to a MC point belonging to the wrong MC Track *or* has *RefIndex* = -1 (→ background)
- ❖ **Missing hit** - a hit *not* associated to the reco track, which has a *RefIndex* pointing to a MC point belonging to the right track, i.e. a hit not associated to the reco track when it should be!

RECO Track associated to MC Track = 1

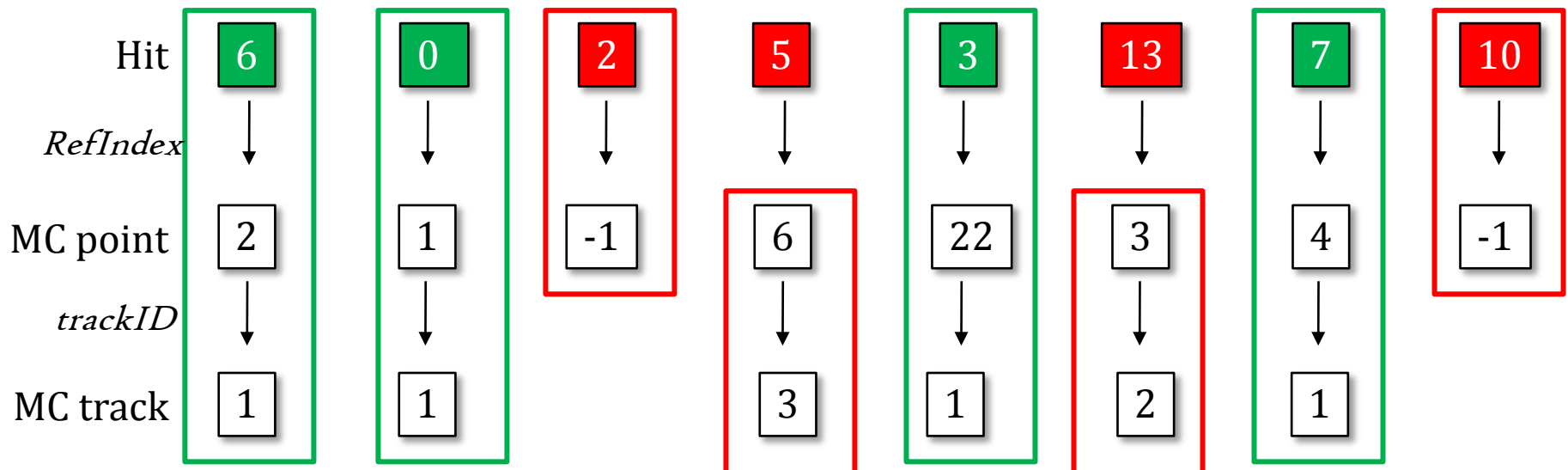


 *fake*  
 *true*

# True, Fake, Missing Hits

- ❖ **True hit** - a hit associated to my reco track, which has a *RefIndex* pointing to a MC point belonging to the right MC track
- ❖ **Fake hit** - a hit associated to the reco track, which has a *RefIndex* pointing to a MC point belonging to the wrong MC Track *or* has *RefIndex* = -1 (→ background)
- ❖ **Missing hit** - a hit *not* associated to the reco track, which has a *RefIndex* pointing to a MC point belonging to the right track, i.e. a hit not associated to the reco track when it should be!

RECO Track associated to MC Track = 1



 *fake*  
 *true*

# Quality Assurance

# TrkQA directory

source: [pandaroot](#) / [trunk](#) / [tracking](#) / [TrkQA](#) @ 27134

Name ▲
↑ ../
📄 PndTrkAnaTask.cxx
📄 PndTrkAnaTask.h
📄 PndTrkMCTrackInfo.cxx
📄 PndTrkMCTrackInfo.h
📄 PndTrkQualityAssuranceTask.cxx
📄 PndTrkQualityAssuranceTask.h
📄 PndTrkRecoTrackInfo.cxx
📄 PndTrkRecoTrackInfo.h

## PndTrkQualityAssuranceTask

- ❖ Loop over **Ideal**Track TCA and fill the PndTrk**MC**TrackInfo TCA
- ❖ Loop over **Real**Track TCA and fill PndTrk**Reco**TrackInfo TCA
- ❖ Compare Reco-to-MC Track Info and fill the reco flag clone/true

# Data Objects

## PndTrkMCTrackInfo

- ❖ # MC points in each detector\*
- ❖ index of the associated PndMCTrack
- ❖ array of indices of associated PndTracks
- ❖ MC position/momentum @1<sup>st</sup> /last points
- ❖ MC charge
- ❖ reconstructability flag

## PndTrkRecoTrackInfo

- ❖ # true hits in each detector\*
- ❖ # fake hits in each detector\*
- ❖ # missing hits in each detector\*
- ❖ PndTrkMCTrackInfo object
- ❖ index of the associated PndMCTrack
- ❖ index of the associated PndTrack
- ❖ reco position/momentum @1<sup>st</sup>/last points
- ❖ reco charge
- ❖ true/clone flag

\* *mvd pixel, mvd strip, stt parallel, stt skew, gem, scitil*

# Draw Data Objects

❖ The point in favour of this is that you can draw all the histograms directly from the TCA in the file and from a macro/prompt of ROOT

e.g.

❖ Set a cut

```
TCut cut = ``MCTrackInfo.GetMCTrackID()==0 && MCTrackInfo.IsReconstructable()==1``
```

❖ Draw efficiency

```
cbmsim->Draw( ``RecoTrackInfo[MCTrackInfo.GetRecoTrackID()].GetEfficiency()`` , cut)
```

❖ Draw efficiency vs MC momentum @ vertex

```
cbmsim->Draw( ``RecoTrackInfo[MCTrackInfo.GetRecoTrackID()].GetEfficiency():  
MCTrack[MCTrackInfo.GetMCTrackID()].GetMomentum().Mag()`` , cut, ``colz``)
```

❖ (MC mom – Reco mom)<sub>x</sub> @ last hit

```
cbmsim->Draw( ``MCTrackInfo.GetMomentumLast().X() -  
RecoTrackInfo[MCTrackInfo.GetRecoTrackID()].GetMomentumLast().X()`` , cut)
```

# Quality of the single track

Key factors to set the quality of a track are:

- ❖ the **conformity to the MC track**
- ❖ the **contamination of the reco track**

## ❖ Conformity

$$\text{EFFICIENCY} = \frac{\# \text{ true hits}}{\# \text{ MC points}}$$

## ❖ Contamination

$$\text{PURITY} = \frac{\# \text{ true hits}}{\# \text{ reco hits}}$$

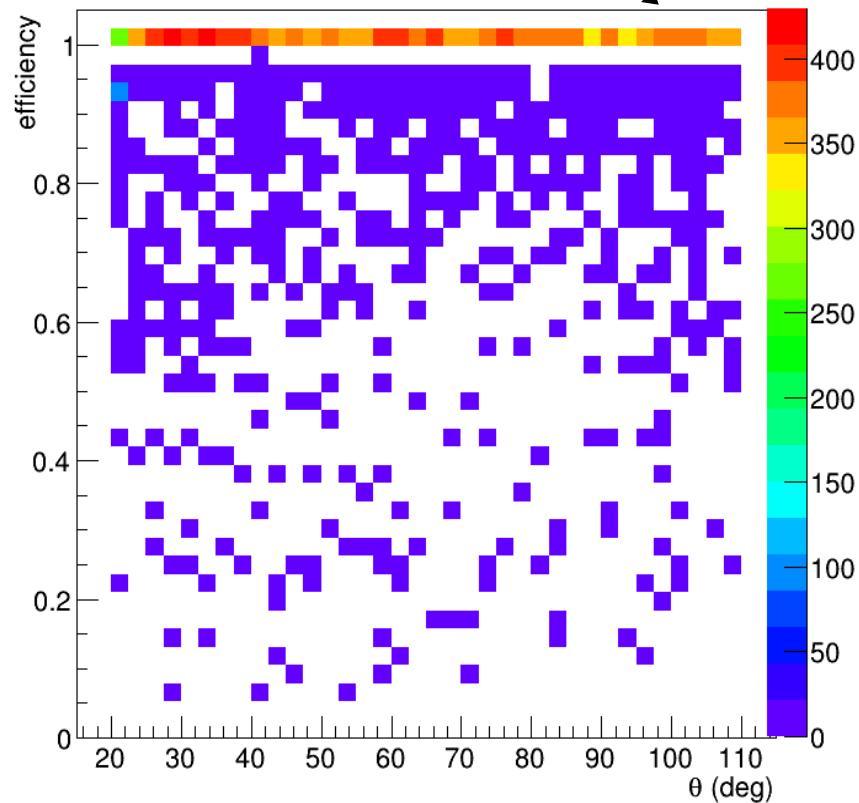
# Example

Draw efficiency vs  $\theta$  @ vertex

```
cbmsim->Draw('`RecoTrackInfo[MCTrackInfo.GetRecoTrackID()].GetEfficiency():  
MCTrack[MCTrackInfo.GetMCTrackID()].GetMomentum().Theta()*TMath::RadToDeg()`',  
RecoTrackInfo.GetMCTrackID() < 3, ``colz``')
```

BoxGenerator,  
3  $\mu^- \forall$  event,  
1 GeV/c,  
 $\theta \in [20^\circ, 110^\circ]$

The line @ 1 is the **best** achievable result





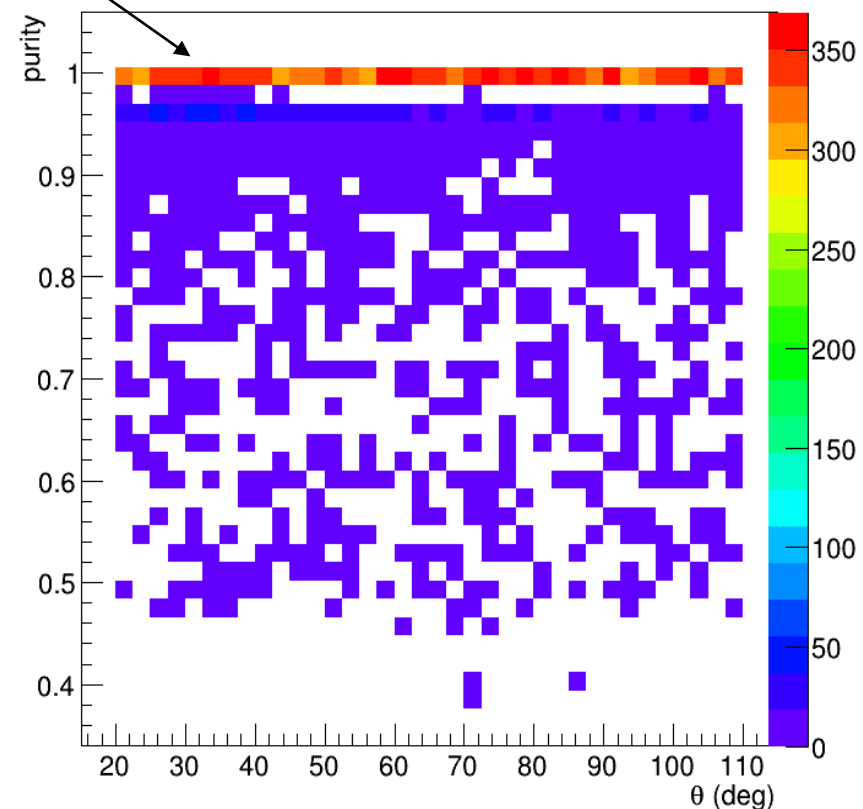
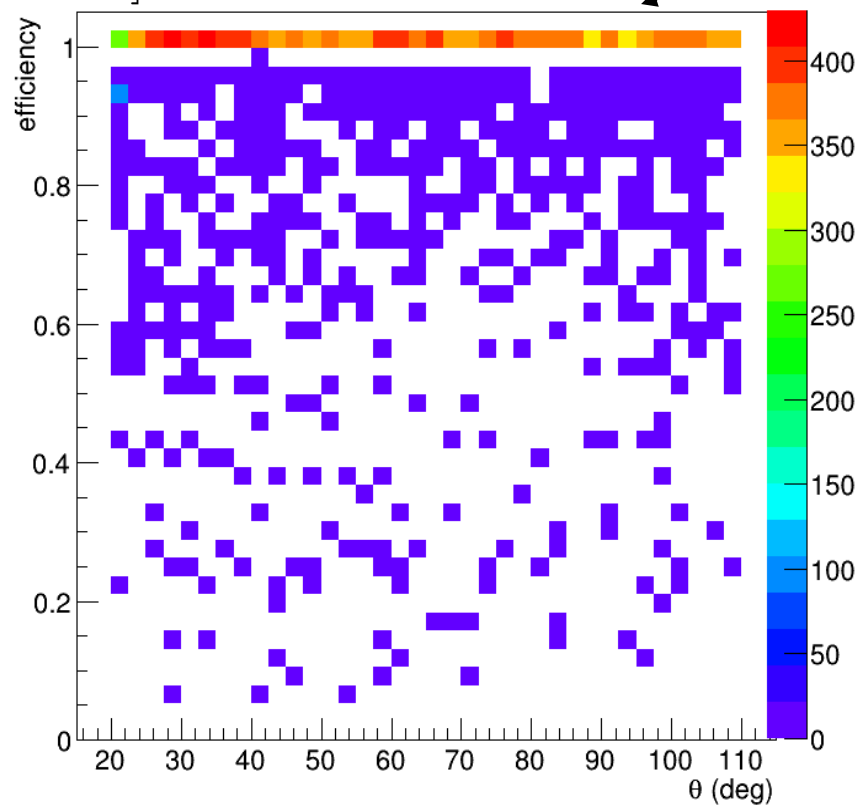
# Example

Draw efficiency vs  $\theta$  @ vertex

```
cbmsim->Draw(`RecoTrackInfo[MCTrackInfo.GetRecoTrackID()].GetPurity():  
MCTrack[MCTrackInfo.GetMCTrackID()].GetMomentum().Theta()*Tmath::RadToDeg()`,  
RecoTrackInfo.GetMCTrackID() < 3, ``colz``)
```

BoxGenerator,  
3  $\mu^- \nu$  event,  
1 GeV/c,  
 $\theta \in [20^\circ,$   
110 $^\circ]$

The line @ 1 is the **best** achievable result



# Quality of the single track

Key factors to set the quality of a track are:

- ❖ the **conformity to the MC track**
- ❖ the **contamination of the reco track**

## ❖ Conformity

$$\text{EFFICIENCY} = \frac{\# \text{ true hits}}{\# \text{ MC points}}$$

different levels can be defined:

Fully Found:	$\varepsilon = 100\%$
Almost Fully Found:	$90\% < \varepsilon < 100\%$
Partially Found:	$60\% < \varepsilon < 90\%$
Ghost:	$0\% < \varepsilon < 60\%$
Missed:	reconstructable but not reconstructed

## ❖ Contamination

$$\text{PURITY} = \frac{\# \text{ true hits}}{\# \text{ reco hits}}$$

different levels can be defined:

Clean:	$p = 100\%$
Almost Clean:	$90\% < p < 100\%$
Noisy:	$p < 90\%$

The *quality* of the track can be defined as a combination of the two information.

# Reconstructability condition

- ❖ The minimum request for a track to be *reconstructable* is to have:
  - ❖ 3 hits in the  $xy$  plane
  - ❖ 2 hits in the  $z\phi$  plane

...BUT

In the STT case, with only 3 drift circles, you have indeed 8 possible tracks!  
So, maybe, the request must be changed to an higher number...

This has to be discussed!

# In svn repository

source: [pandaroot](#) / [trunk](#) / [tracking](#) @ 27159

Name ▲	Size	Rev	Age
⬆ ../			
▶ TrkAlgo		26602	3
▶ TrkData		27099	13
▶ TrkQA		27121	6
▶ TrkSecondary		27100	13
▶ TrkStructure		26869	7



```
PndSttMvdGemTrackingIdeal* idealpr = new PndSttMvdGemTrackingIdeal();  
idealpr->SetTrackingEfficiency(1.);  
idealpr->SetTrackOutput("IdealTrack");  
idealpr->SetPersistence(kTRUE);  
fRun->AddTask(idealpr);
```

```
PndMCTrackAssociator* trackMC = new PndMCTrackAssociator();  
trackMC->SetTrackInBranchName("IdealTrack");  
trackMC->SetTrackOutBranchName("IdealTrackID");  
trackMC->SetPersistence(kTRUE);  
fRun->AddTask(trackMC);
```

```
PndTrkQualityAssuranceTask *qa = new PndTrkQualityAssuranceTask();  
qa->SetVerbose(1);  
fRun->AddTask(qa);
```

THANK YOU