Track-Based Particle Flow

Outline:

- Status of Particle Flow Algorithms in Marlin
- Details on Track-Based Particle Flow
- Performance of Track-Based Particle Flow
- Conclusions / Future Plans

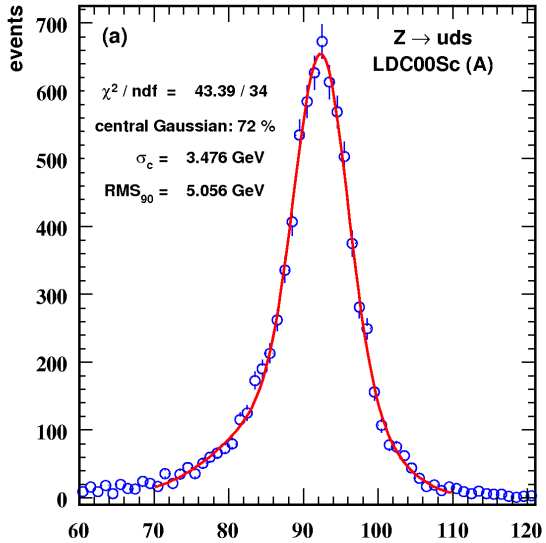
Status of Particle Flow Algorithms in Marlin

Wolf and **PandoraPFA**:

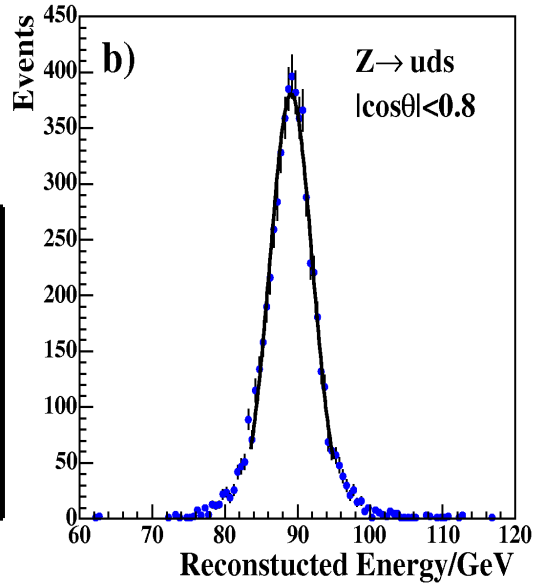
- Wolf: $\Delta E/E \approx \mathbf{0.53}/\sqrt{E}$ ($RMS_{90\%}$) for $Z^0 \rightarrow$ uds @ 91.2 GeV
- PandoraPFA: $\Delta E/E \approx \mathbf{0.30}/\sqrt{E}$ ($RMS_{90\%}$) for $Z^0 \rightarrow$ uds @ 91.2 GeV
- ✔ PandoraPFA reaches design goal at 91.2 GeV and reaches 37% for 100 GeV jets
- ➔ good enough for physics studies with $E_{jet} < 100$ GeV
- ✗ performance of both **degrade** with increasing jet energy
- ➔ study the limits of PFlow in general (quantitatively)
- both are **'cluster-based'** algorithms (PandoraPFA: tracks ↔ cluster-ass.)
- **'track-based'** algorithm offer more potential
- ➔ started to develop a **modular** track-based PFlow in Marlin (summer/autumn 2006)

E_{JET}	$\sigma_E/E = \alpha\sqrt{(E/GeV)}$ $\lvert\cos\theta\rvert<0.8$
45 GeV	0.30
100 GeV	0.37
180 GeV	0.57
250 GeV	0.75

PandoraPFA

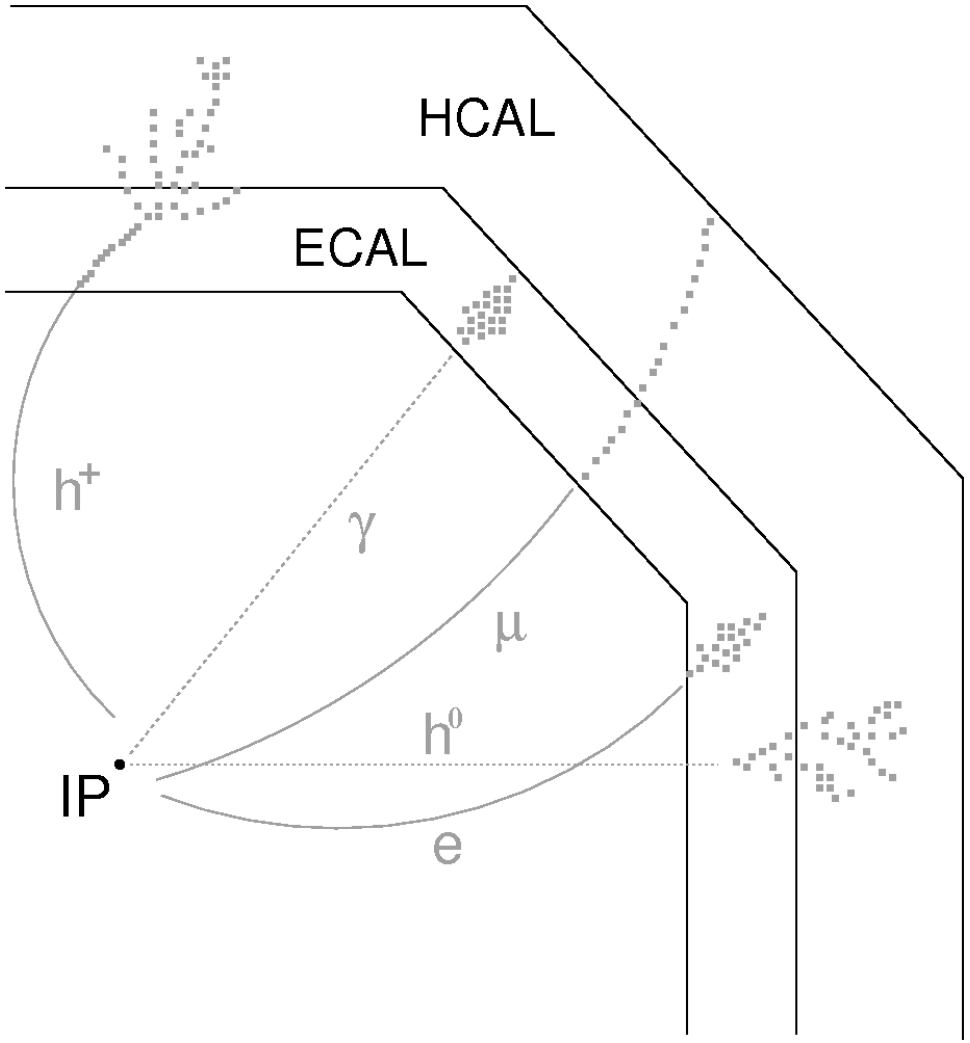


Wolf



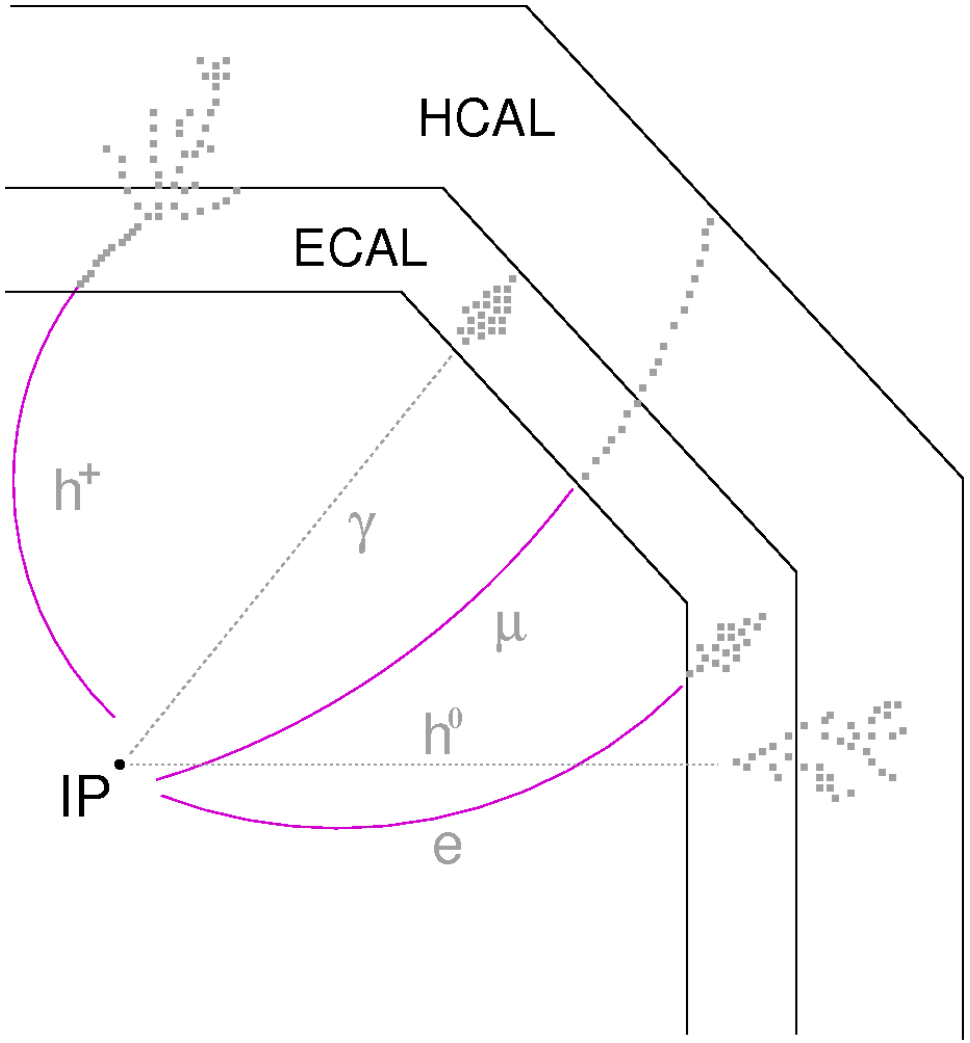
PandoraPFA

Track-Based Particle Flow



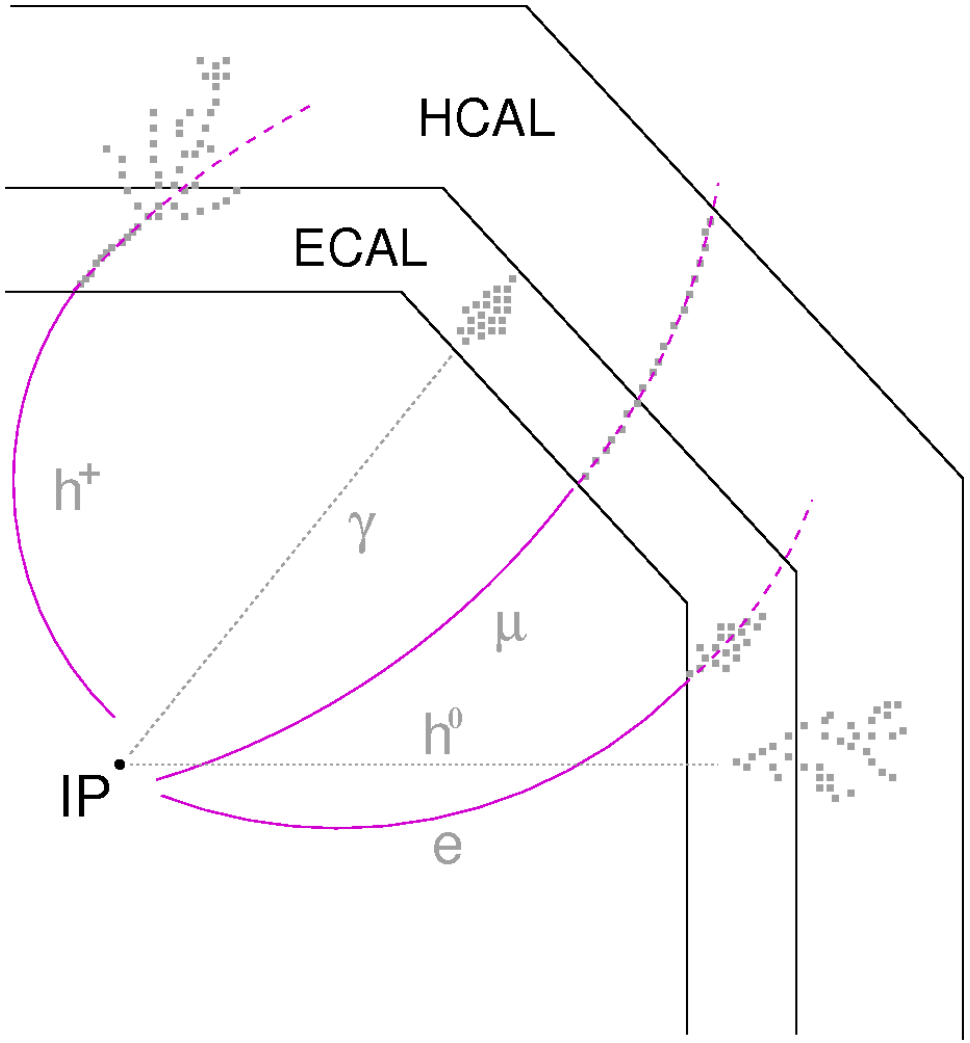
Track-Based Particle Flow

1. tracking (VTX, SIT, TPC...)



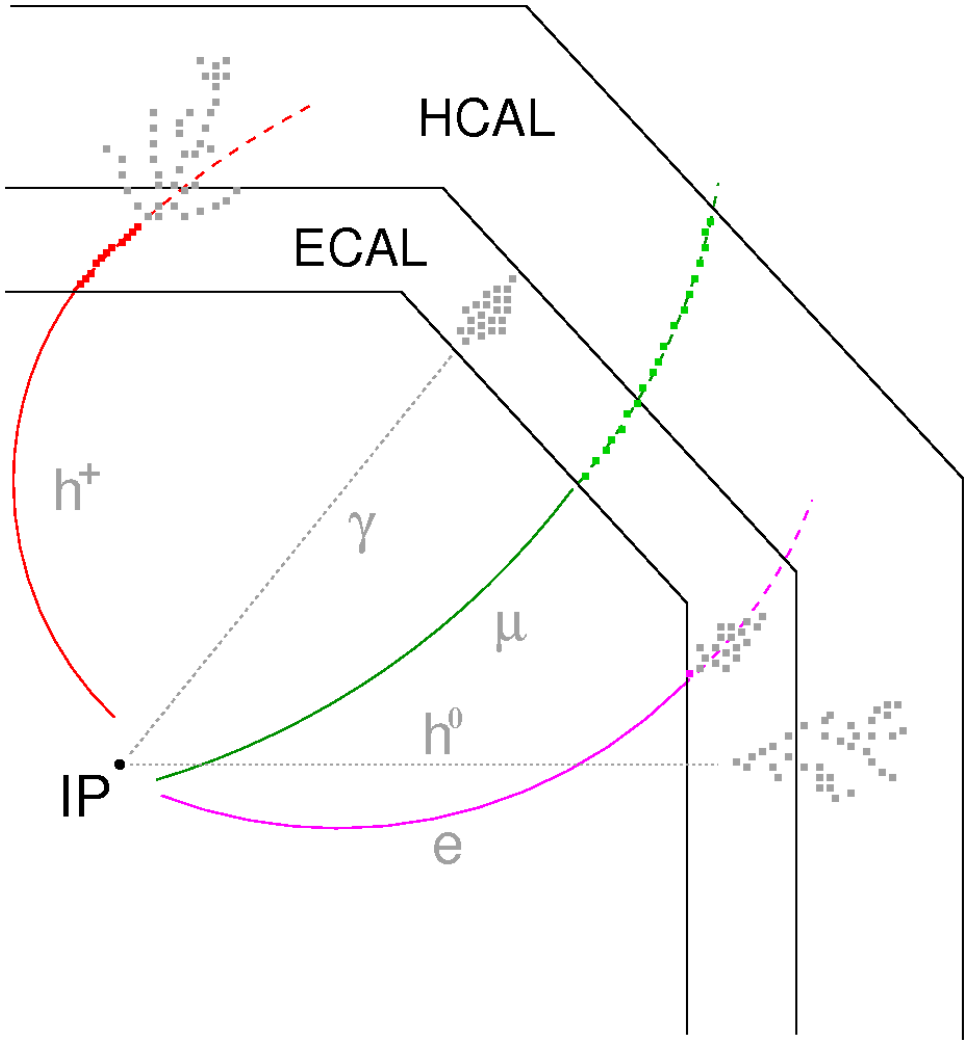
Track-Based Particle Flow

1. tracking (VTX, SIT, TPC...)
2. extrapolate tracks into Calorimeter



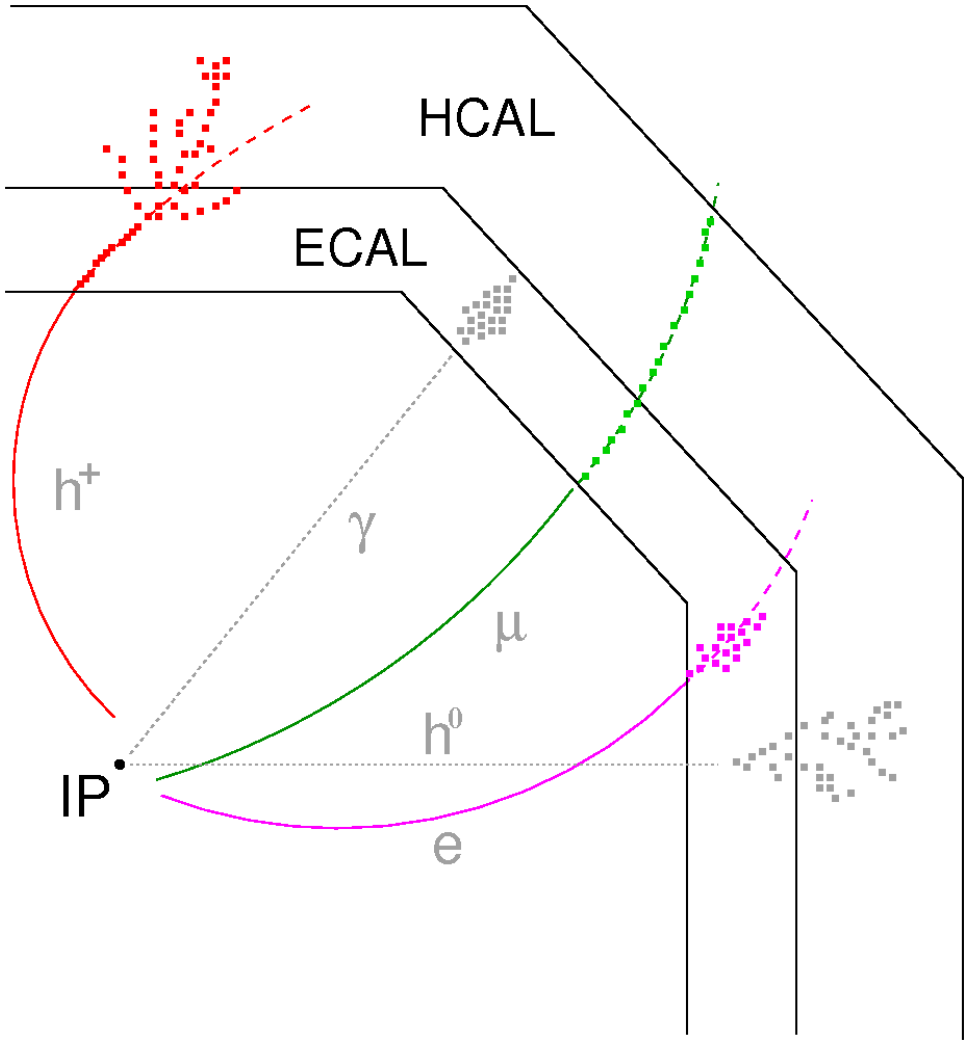
Track-Based Particle Flow

1. tracking (VTX, SIT, TPC...)
2. extrapolate tracks into Calorimeter
3. assign MIP stub to track
   - → get $\mu^{+/-}$ as well



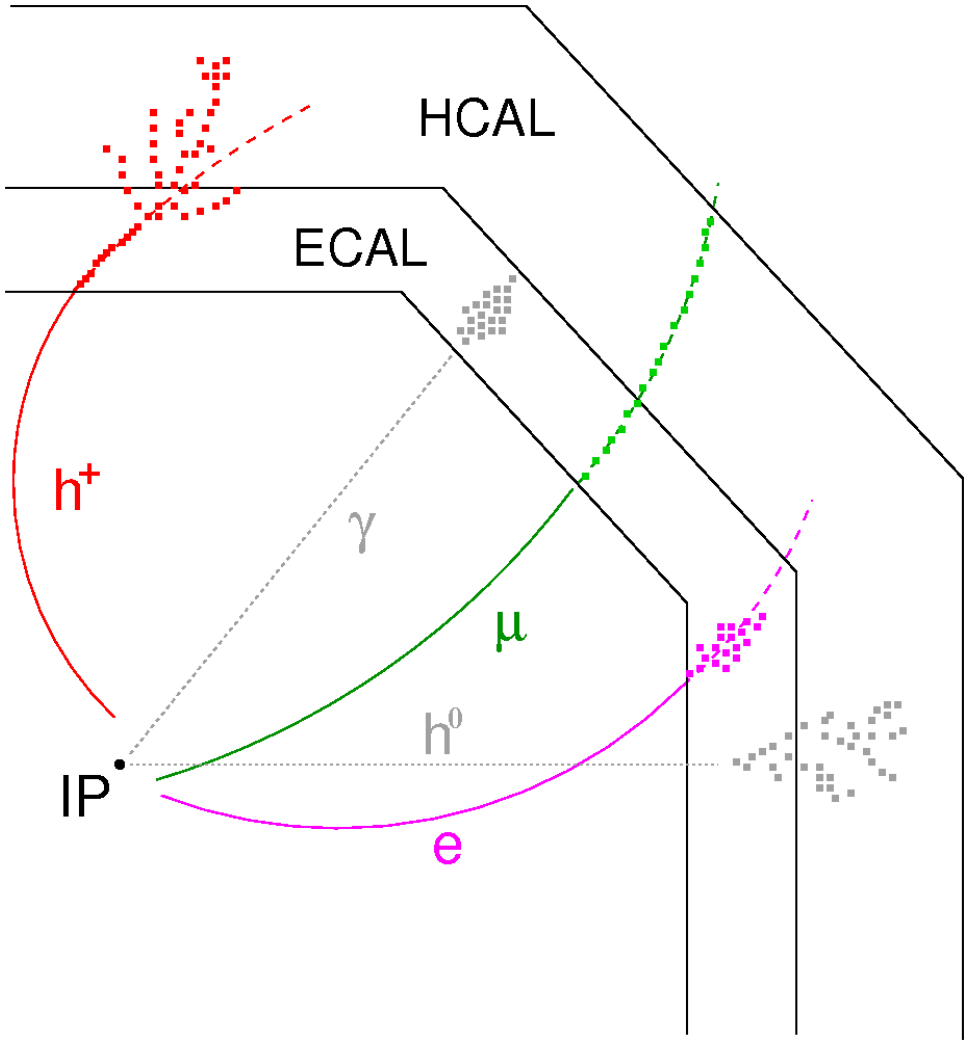
Track-Based Particle Flow

1. tracking (VTX, SIT, TPC...)
2. extrapolate tracks into Calorimeter
3. assign MIP stub to track
   - get $\mu^{+/-}$ as well
4. clustering (ECAL and HCAL)
   - variable, depending on track
   - different algorithms



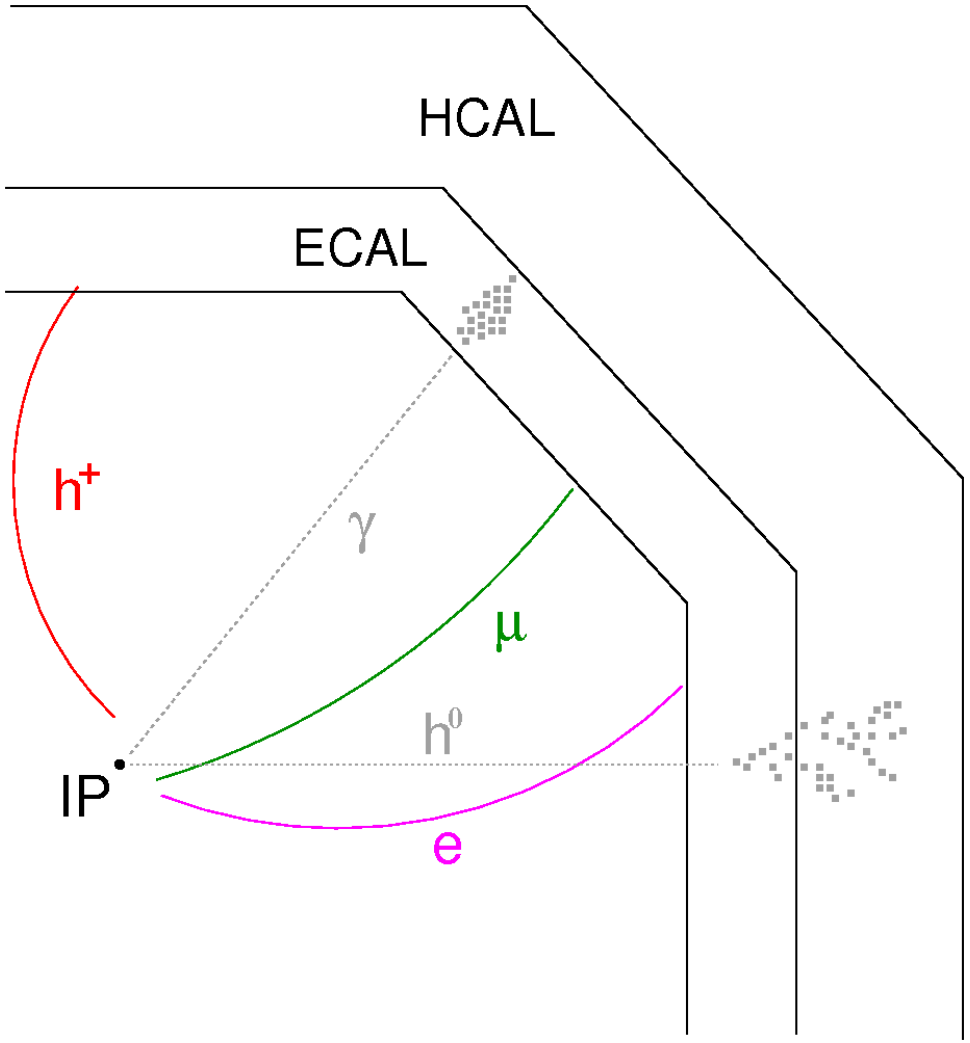
Track-Based Particle Flow

1. tracking (VTX, SIT, TPC...)
2. extrapolate tracks into Calorimeter
3. assign MIP stub to track
   - get $\mu^{+/-}$ as well
4. clustering (ECAL and HCAL)
   - variable, depending on track
   - different algorithms
5. particle ID for $e^{+/-}$, $\mu^{+/-}$, $h^{+/-}$
   - fraction of energy in ECAL/HCAL



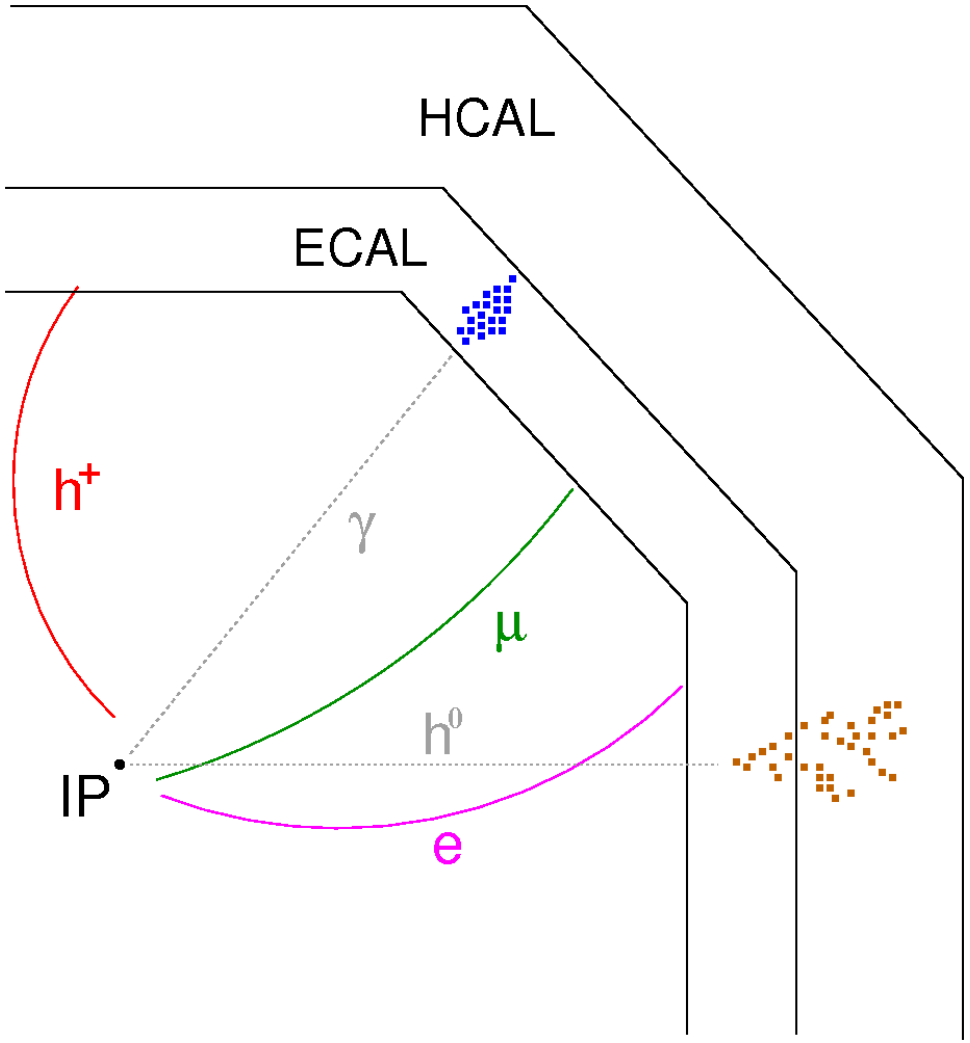
Track-Based Particle Flow

1. tracking (VTX, SIT, TPC...)
2. extrapolate tracks into Calorimeter
3. assign MIP stub to track
   - → get $\mu^{+/-}$ as well
4. clustering (ECAL and HCAL)
   - → variable, depending on track
   - → different algorithms
5. particle ID for $e^{+/-}$, $\mu^{+/-}$, $h^{+/-}$
   - → fraction of energy in ECAL/HCAL
6. remove 'charged' Calorimeter hits



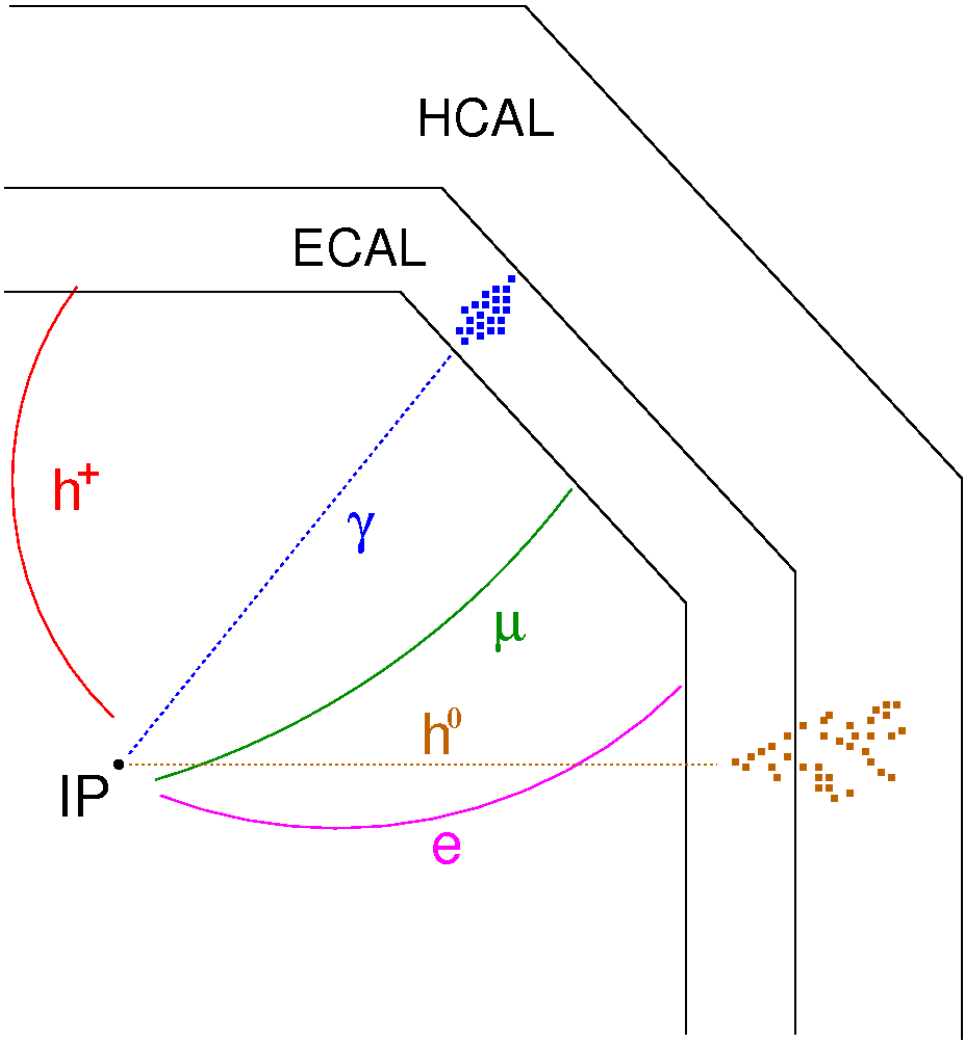
Track-Based Particle Flow

1. tracking (VTX, SIT, TPC...)
2. extrapolate tracks into Calorimeter
3. assign MIP stub to track
   - get $\mu^{+/-}$ as well
4. clustering (ECAL and HCAL)
   - variable, depending on track
   - different algorithms
5. particle ID for $e^{+/-}$, $\mu^{+/-}$, $h^{+/-}$
   - fraction of energy in ECAL/HCAL
6. remove 'charged' Calorimeter hits
7. clustering on 'neutral' hits



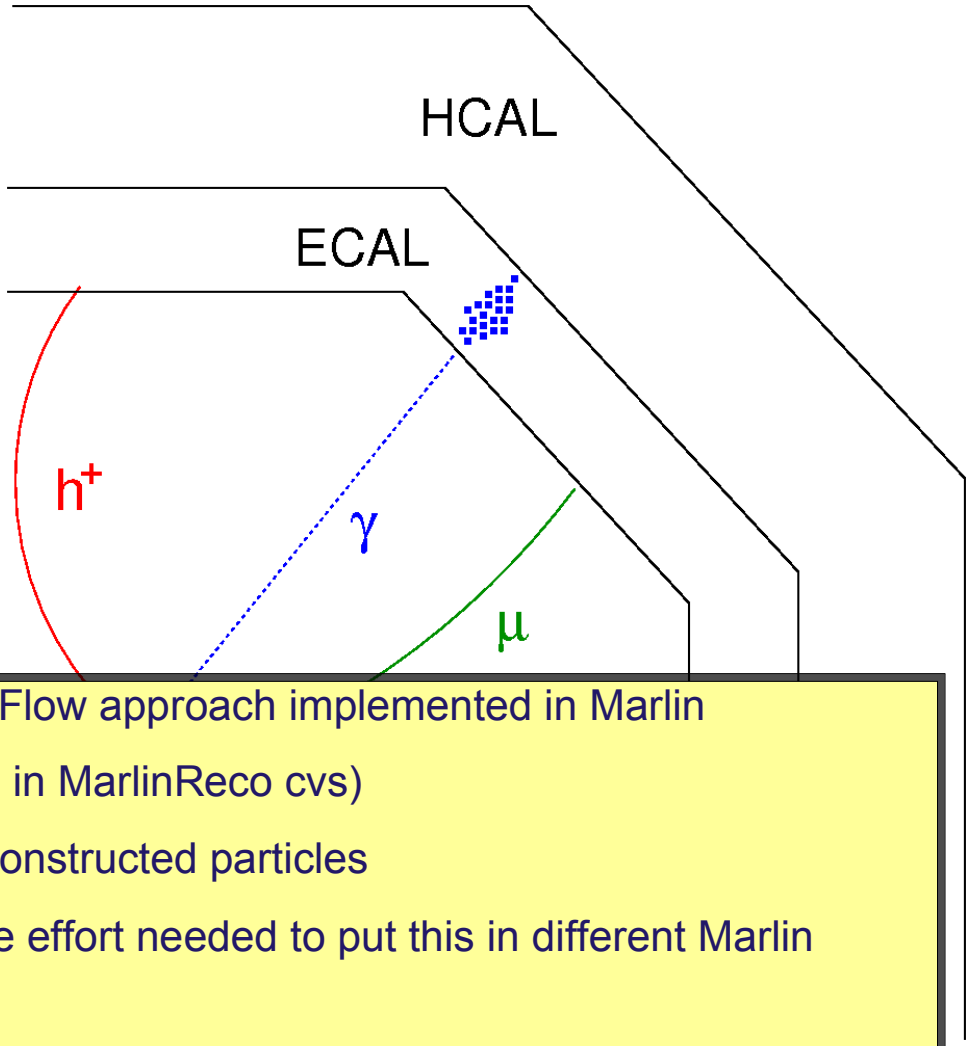
Track-Based Particle Flow

1. tracking (VTX, SIT, TPC...)
2. extrapolate tracks into Calorimeter
3. assign MIP stub to track
   - get $\mu^{+/-}$ as well
4. clustering (ECAL and HCAL)
   - variable, depending on track
   - different algorithms
5. particle ID for $e^{+/-}$, $\mu^{+/-}$, $h^{+/-}$
   - fraction of energy in ECAL/HCAL
6. remove 'charged' Calorimeter hits
7. clustering on 'neutral' hits
8. particle ID for γ, h^0



Track-Based Particle Flow

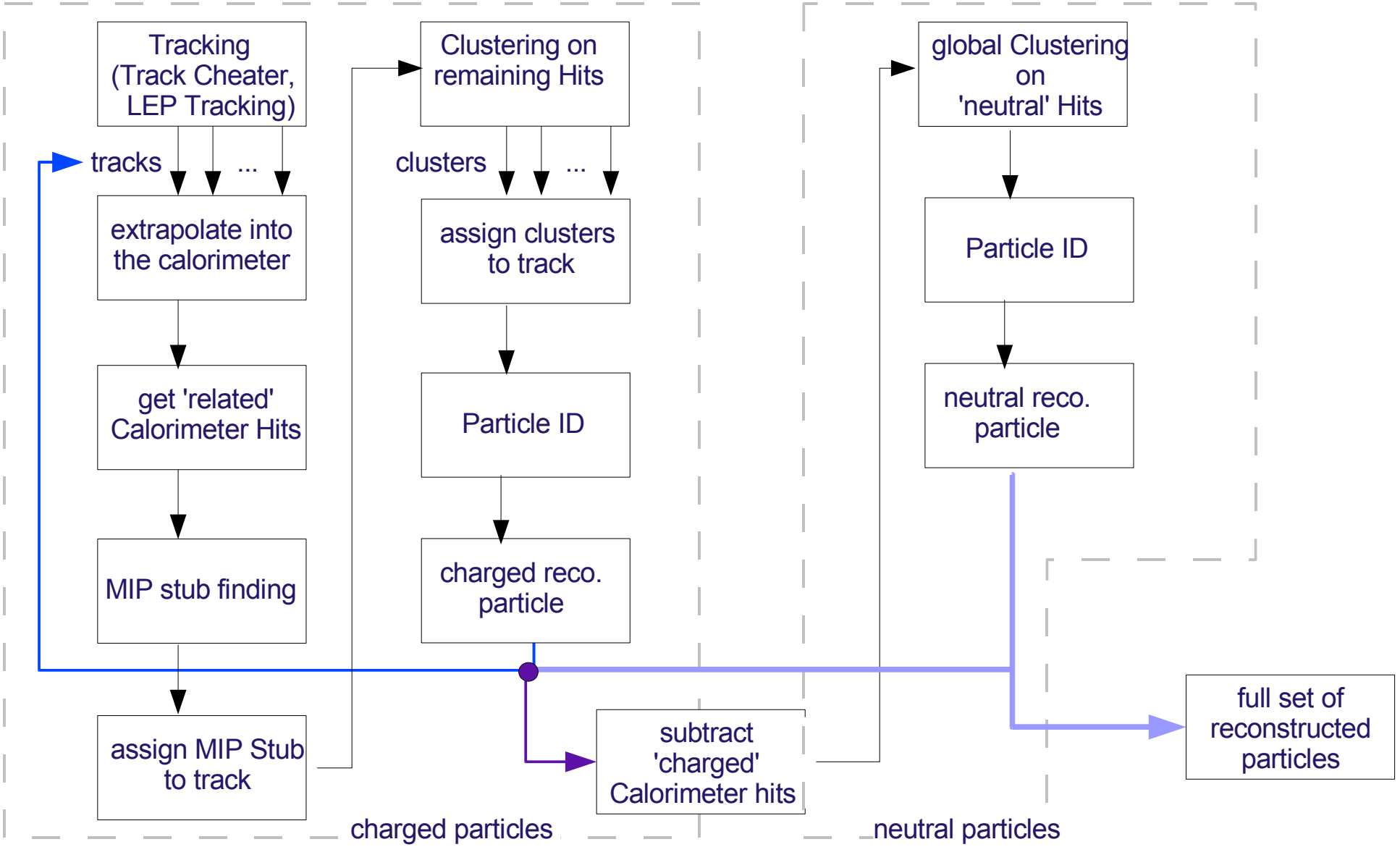
1. tracking (VTX, SIT, TPC...)
2. extrapolate tracks into Calorimeter
3. assign MIP stub to track
   - → get $\mu^{+/-}$ as well
4. clustering (ECAL and HCAL)
   - → variable, depending on track
   - → different algorithms
5. particle ID for $e^{+/-}$, $\mu^{+/-}$, $h^{+/-}$
   - → fraction of energy in ECAL/HCAL



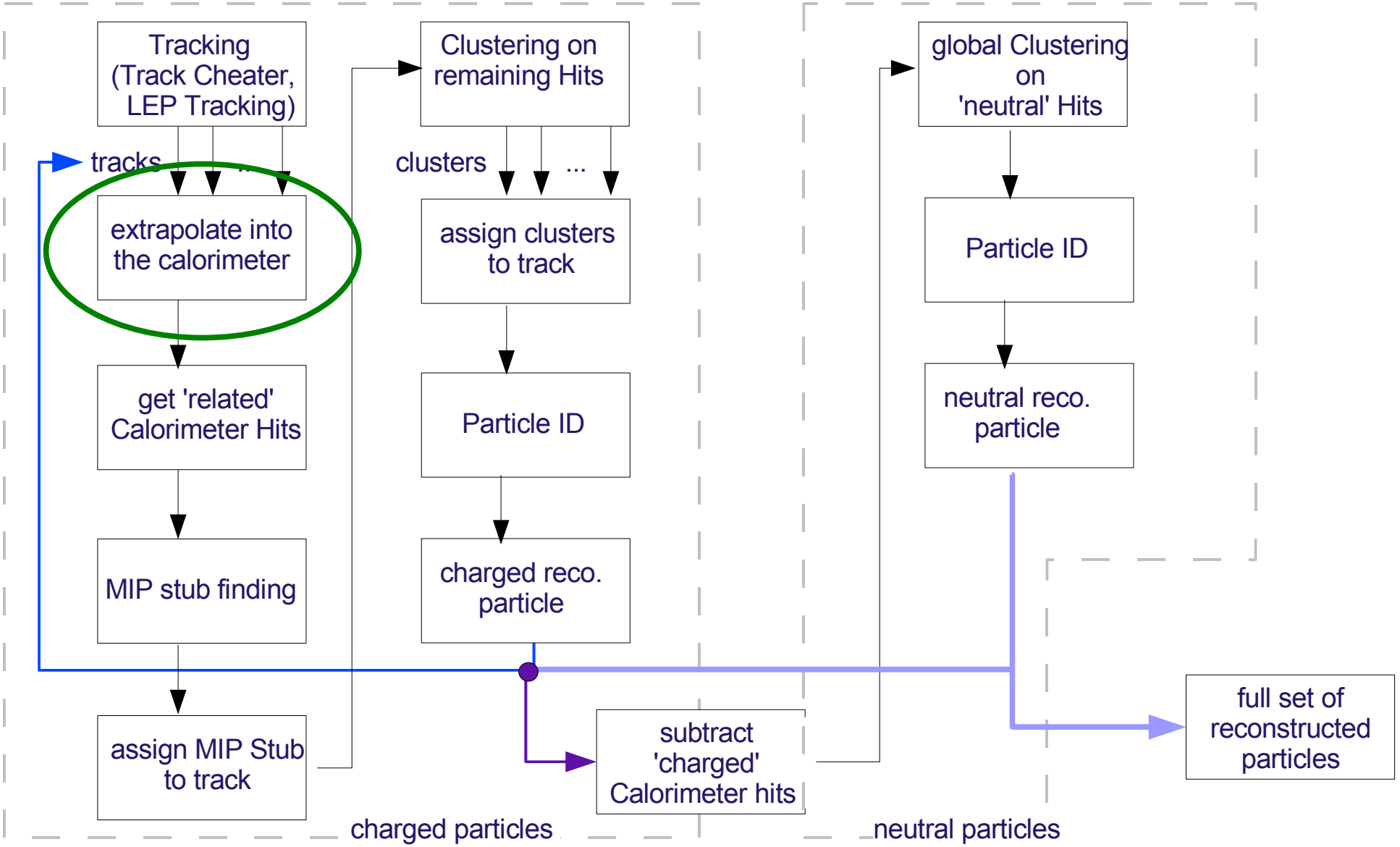
first full version of such a **'track-based'** Particle Flow approach implemented in Marlin

- full software chain established (initial version in MarlinReco cvs)
- → put in tracks and calorimeter hits, get out reconstructed particles
- → more or less modular approach, but still more effort needed to put this in different Marlin processors
- first results **only** for $Z^0 \rightarrow$ uds @ 91.2 GeV

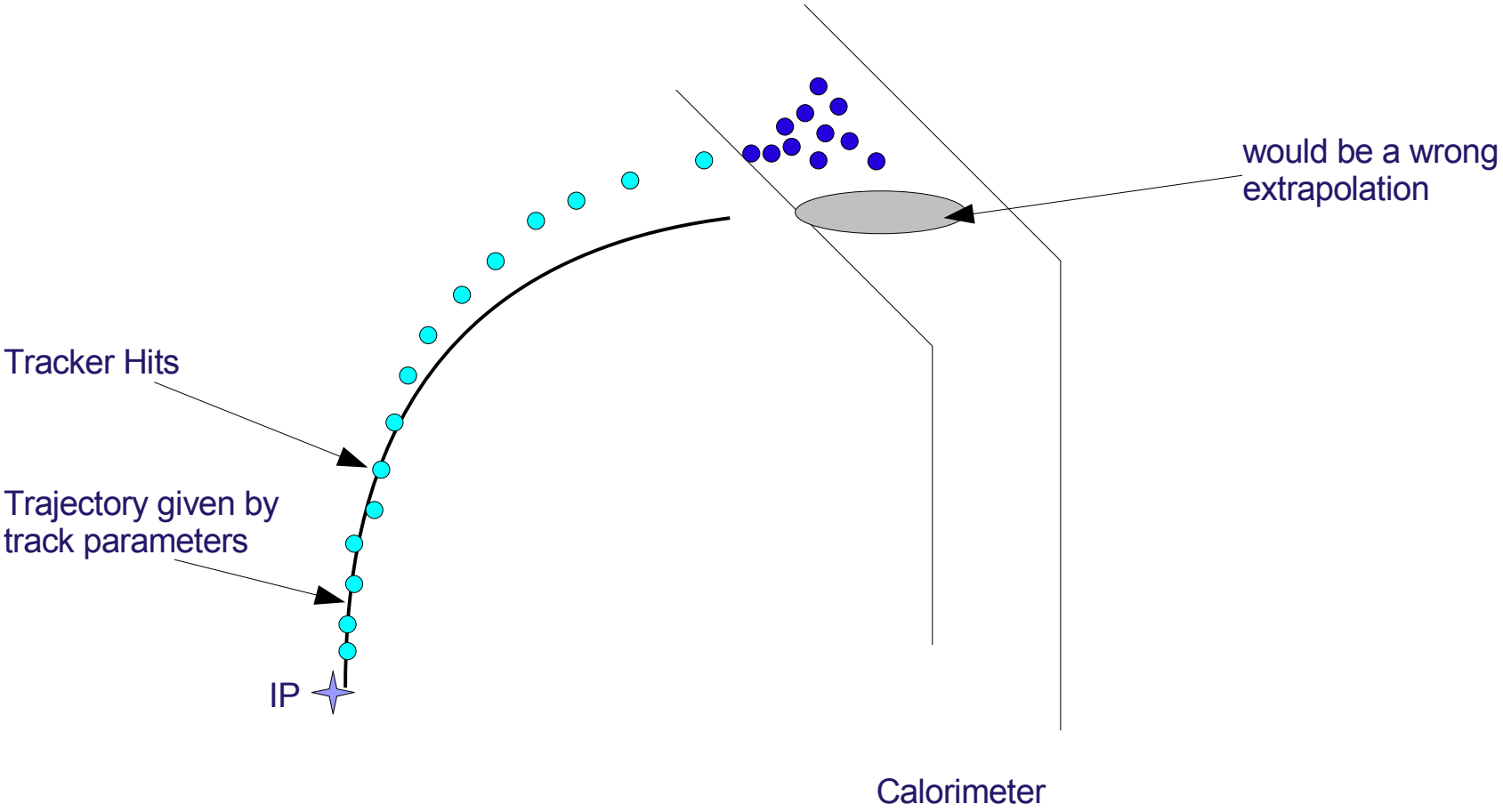
Track-Based Particle Flow



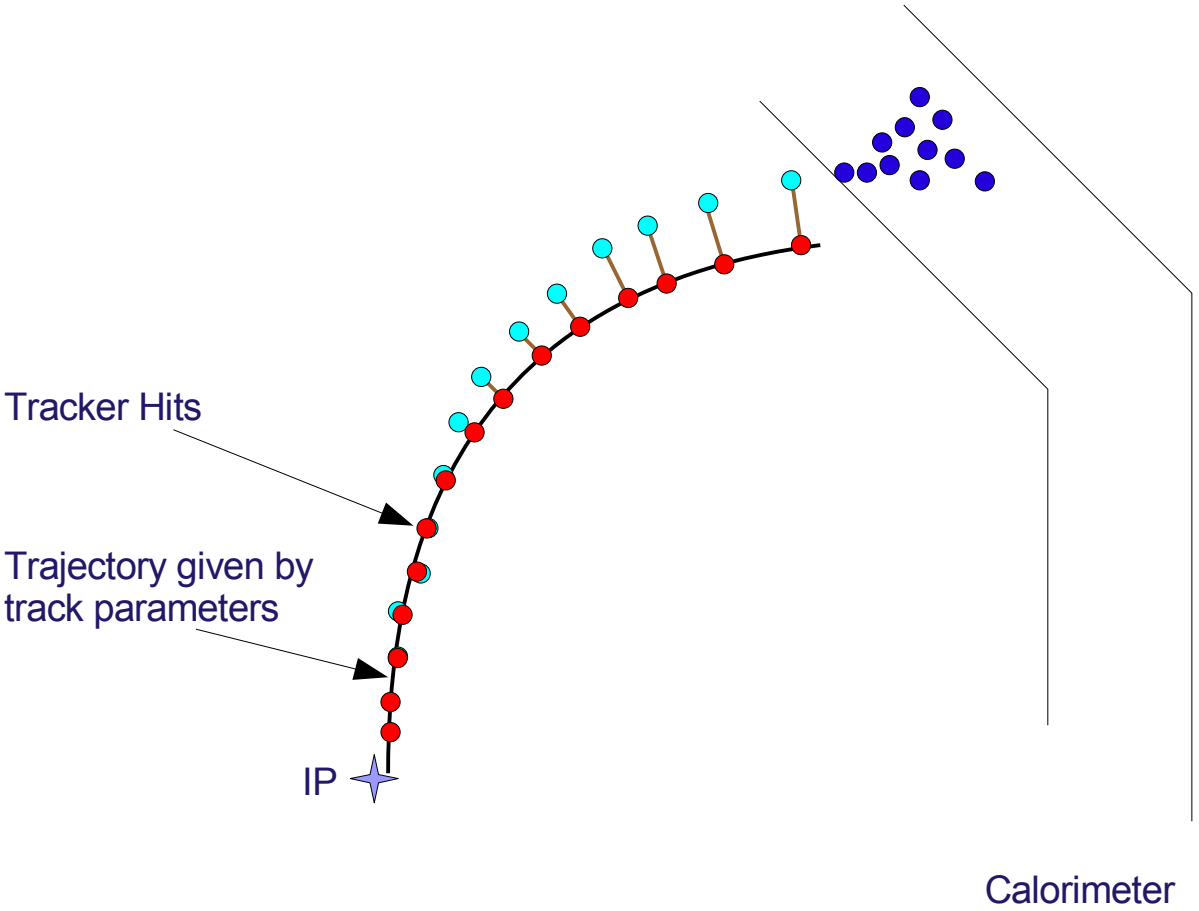
Track Extrapolation



Track Extrapolation

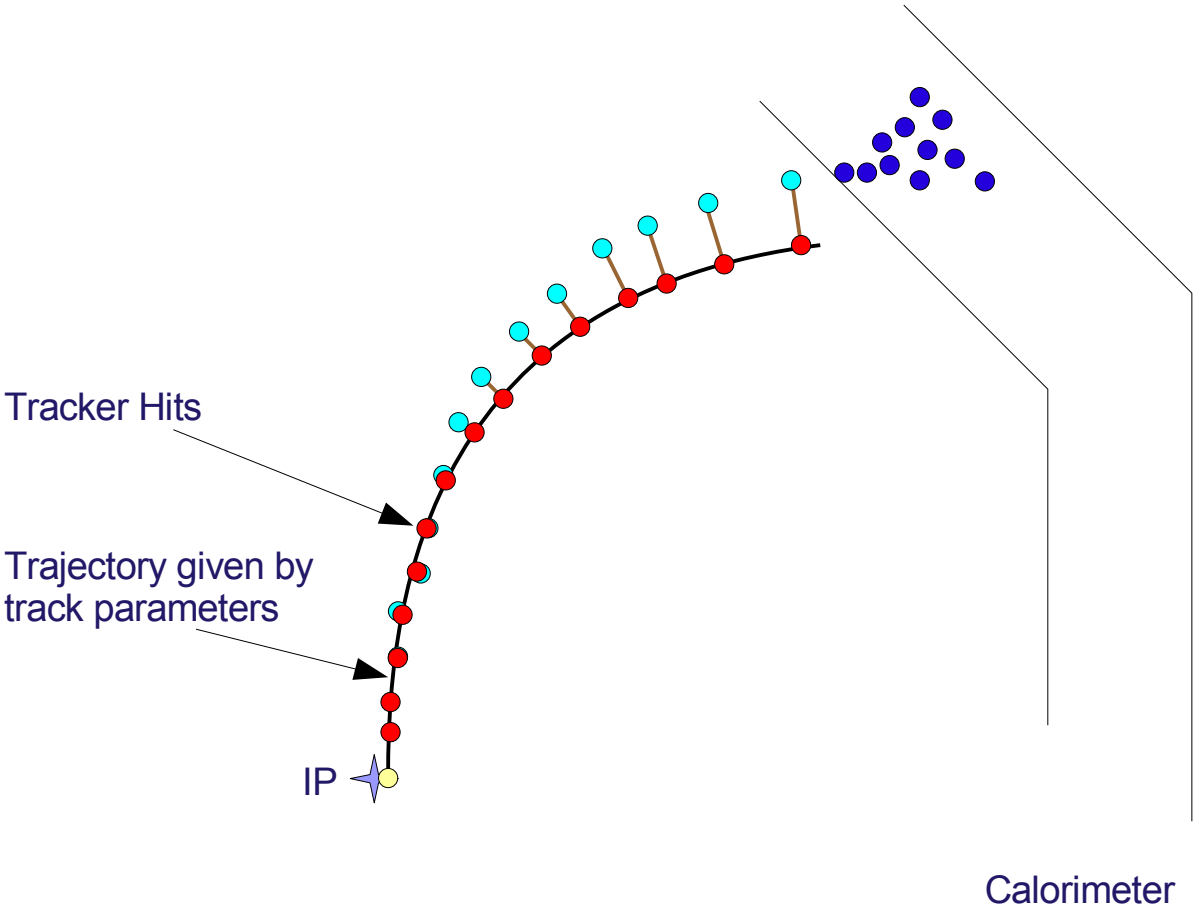


Track Extrapolation



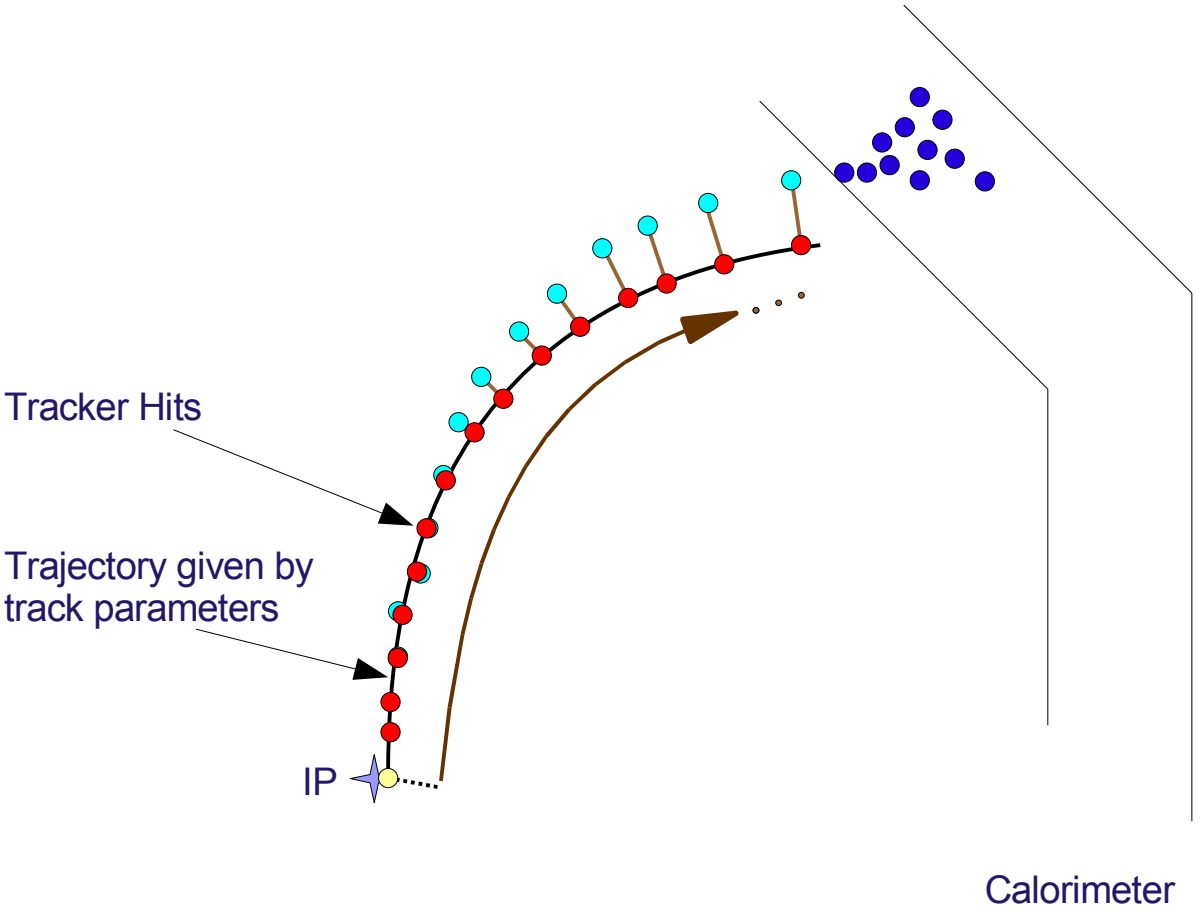
- project hits on trajectory

Track Extrapolation



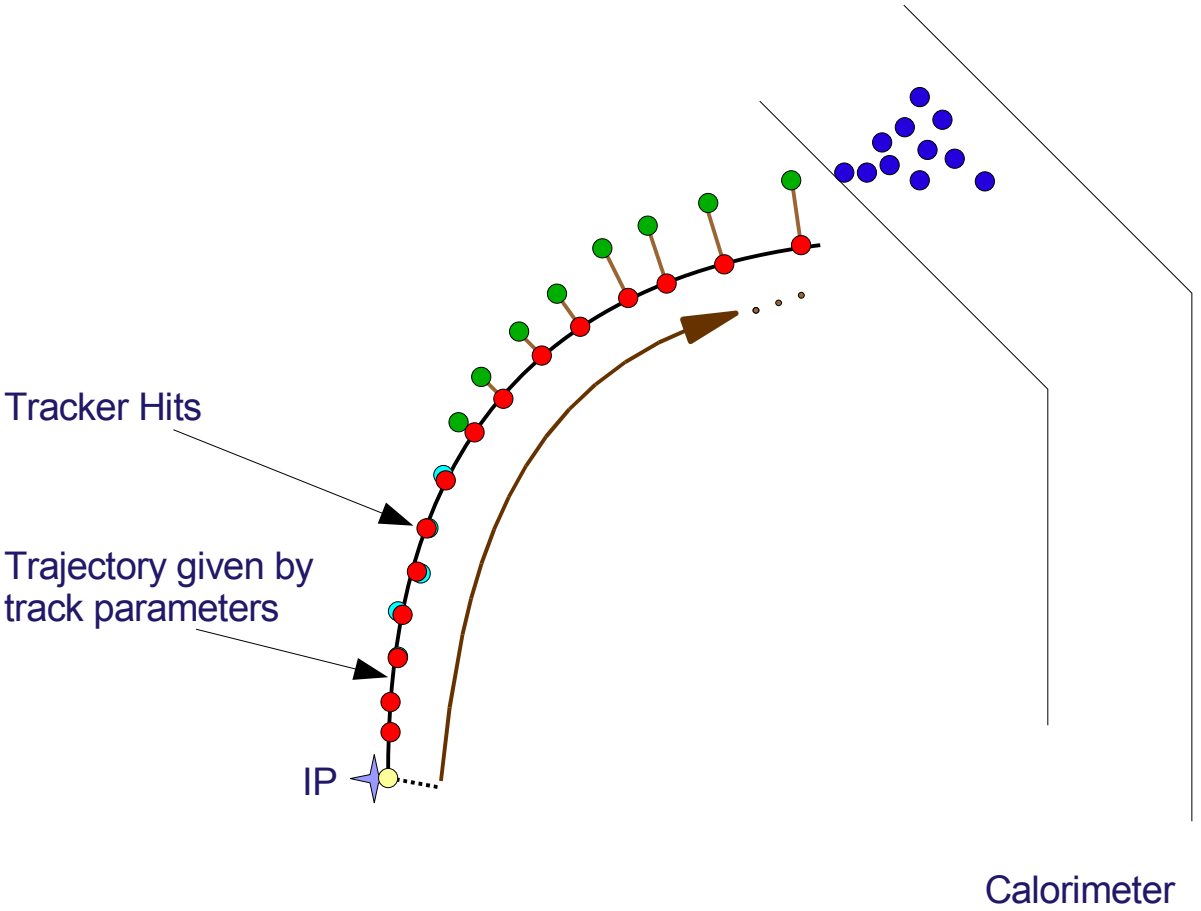
- project hits on trajectory
- take PCA as reference point

Track Extrapolation



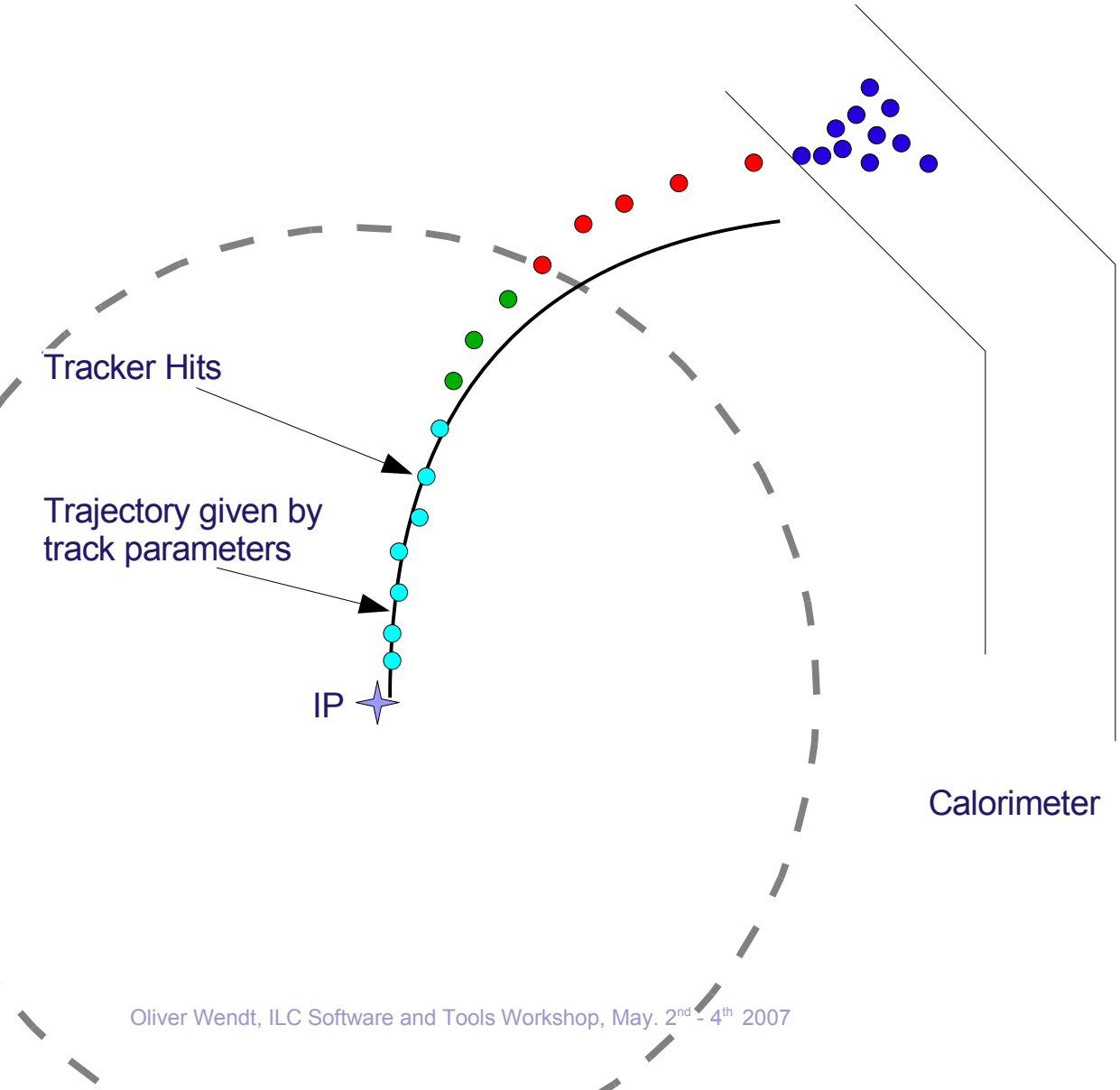
- project hits on trajectory
- take PCA as reference point
- calculate path length of all projected hits

Track Extrapolation



- project hits on trajectory
- take PCA as reference point
- calculate path length of all projected hits
- ➔ take the **n** hits with the largest path length as outermost hits

Track Extrapolation



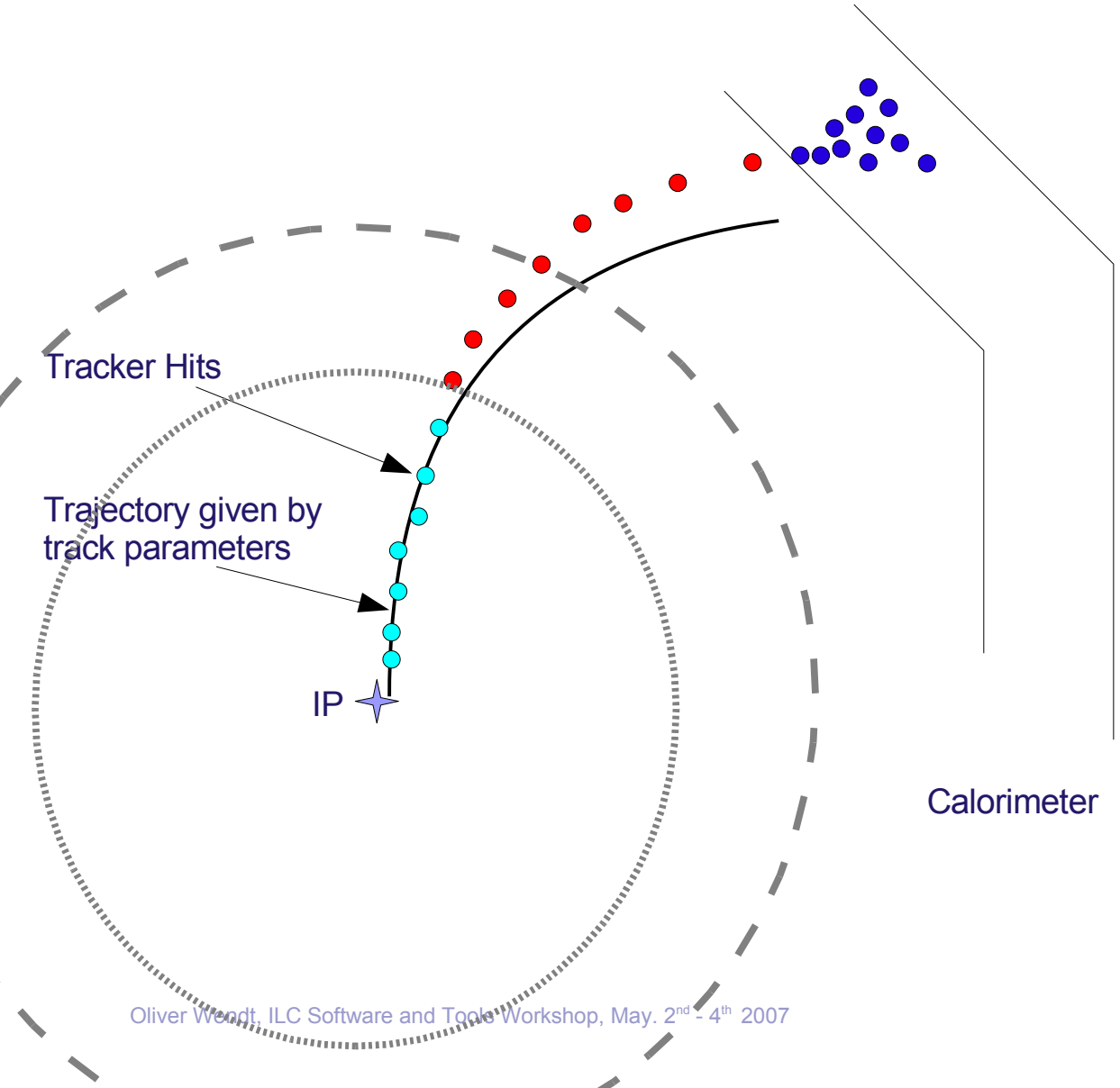
- project hits on trajectory
- take PCA as reference point
- calculate path length of all projected hits

→ take the **n** hits with the largest path length as outermost hits

1. cut: at least **k** hits (out of **n**) outside a cylindrical volume

→ ensure that the track reaches the calorimeter

Track Extrapolation



- project hits on trajectory
- take PCA as reference point
- calculate path length of all projected hits

→ take the **n** hits with the largest path length as outermost hits

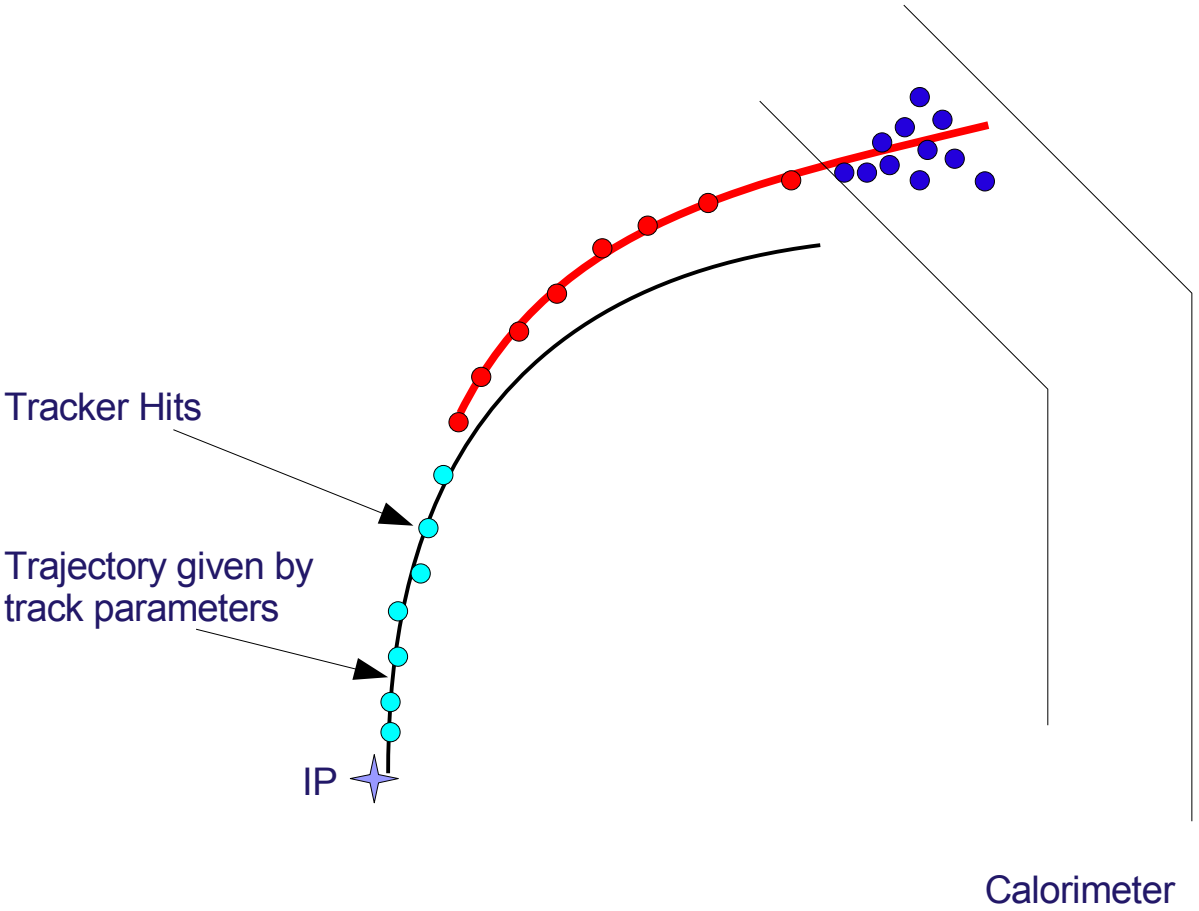
1. cut: at least **k** hits (out of **n**) outside a cylindrical volume

→ ensure that the track reaches the calorimeter

2. cut: at least **l** hits outside a second cylindrical volume

→ ensure that there are enough hits to make a helix/trajectory fit

Track Extrapolation



- project hits on trajectory
- take PCA as reference point
- calculate path length of all projected hits

→ take the **n** hits with the largest path length as outermost hits

1. cut: at least **k** hits (out of **n**) outside a cylindrical volume

→ ensure that the track reaches the calorimeter

2. cut: at least **l** hits outside a second cylindrical volume

→ ensure that there are enough hits to make a helix/trajectory fit

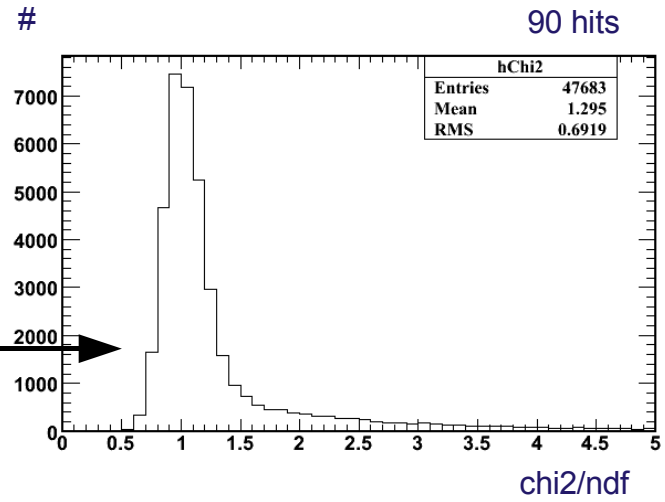
- perform a helix/trajectory fit on 'outermost' hits

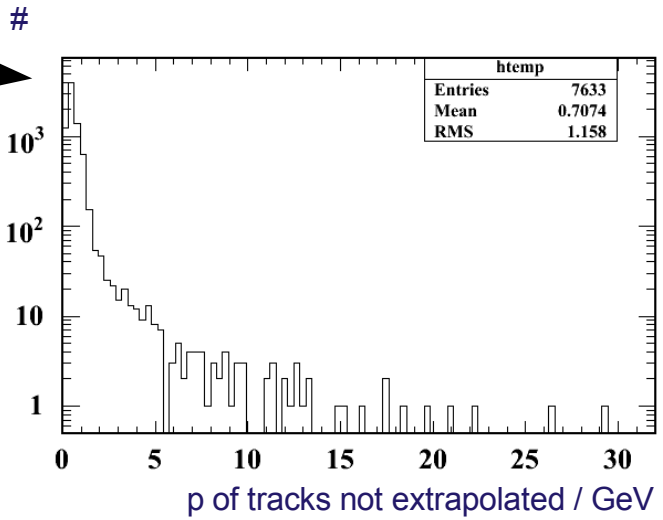
→ points towards corresponding shower

Track Extrapolation

Trajectory class serves as an interface

- ➔ first implementation available in MarlinUtil
- ➔ helix model used as Trajectory at the moment
- **fitting** based on GSL using canonical track parametrisation
- works quite well for tracks with > 50 hits
- but problems with nearly **half** of the tracks, which are:
  - tracks with < 10 to 15 hits and / or
  - tracks with small momenta → curler
- need more sophisticated / general fitting module 'in the spirit of' the Trajectory class to improve fit





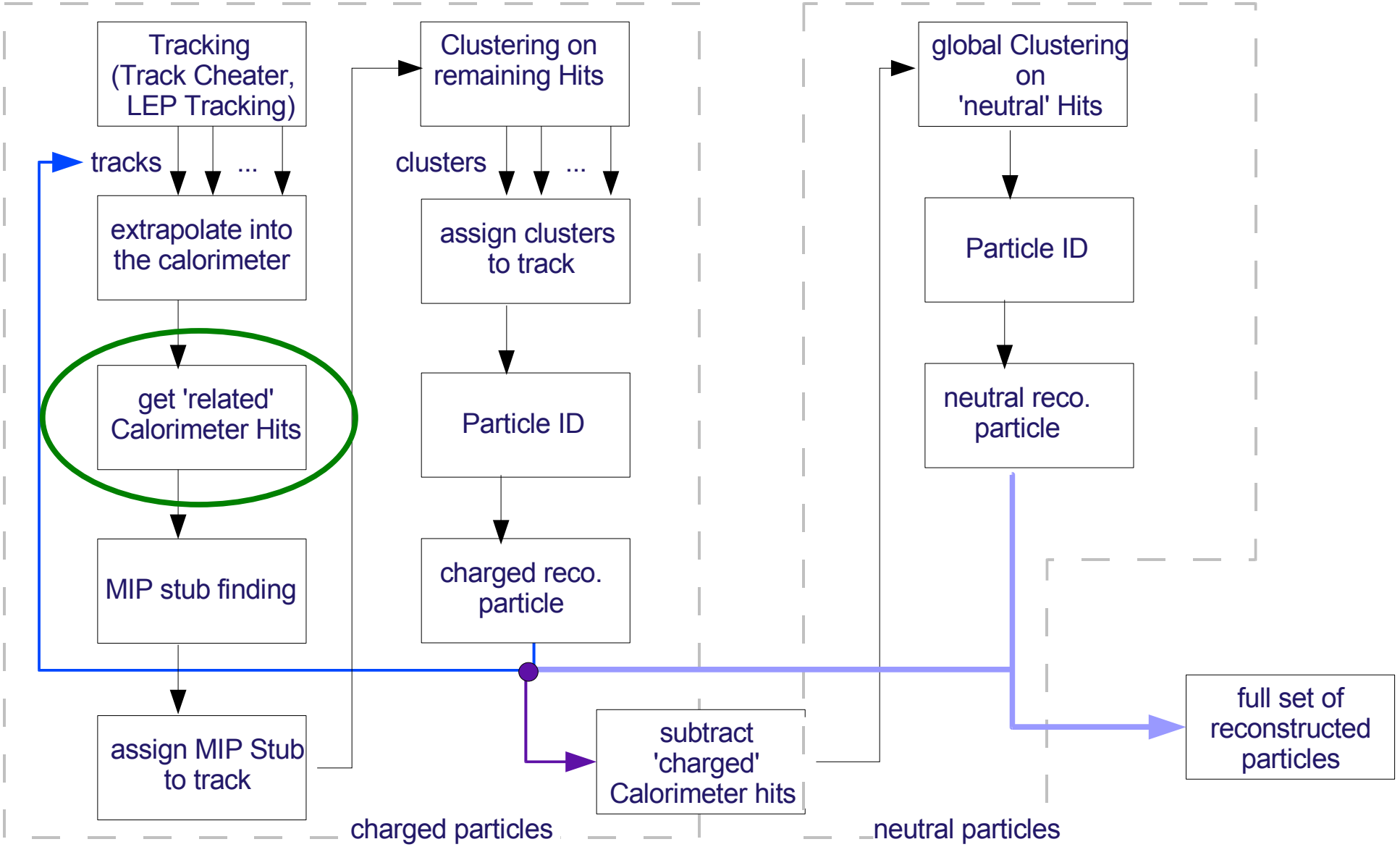
For tracks **failed** to extrapolate:

- if such track reach calorimeter → reconstructed with calorimeter resolution
- if not, and if it has no daughters already reconstructed
  → e.g. particle hit beam pipe
  → measured as particle with track only

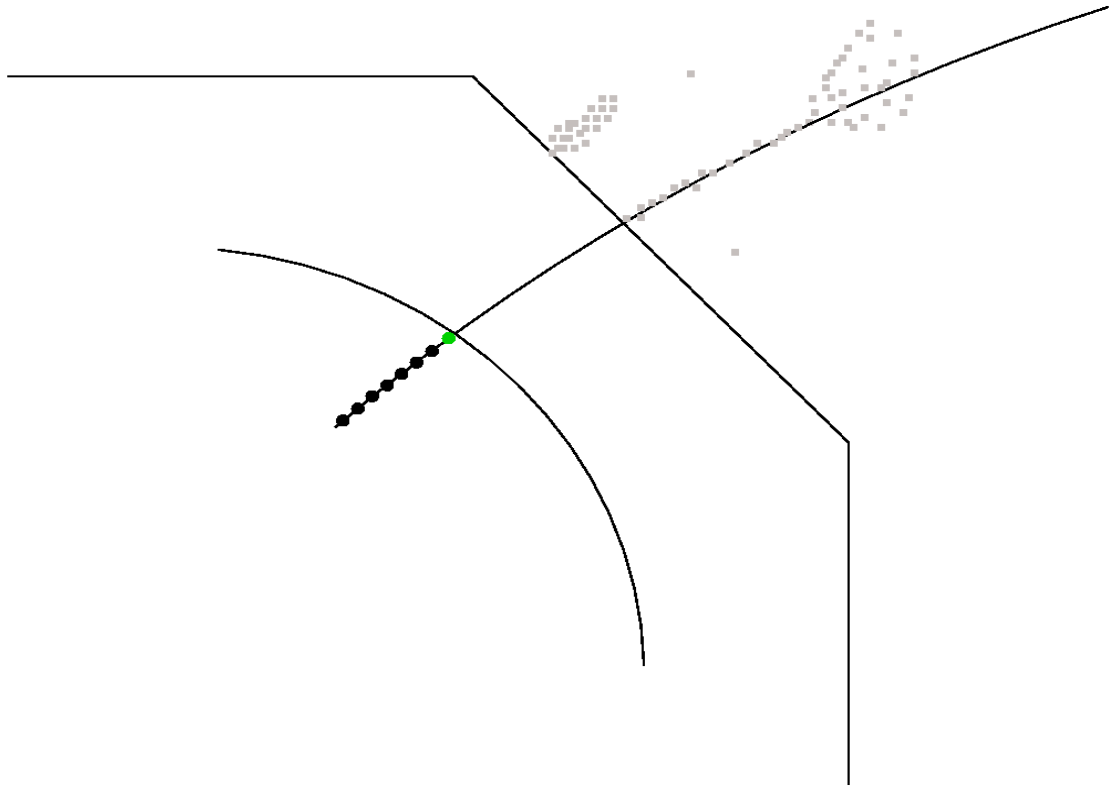
➔ at the moment done by MC information

➔ need dedicated V0-, kink-finding module

Assign related Calorimeter Hits

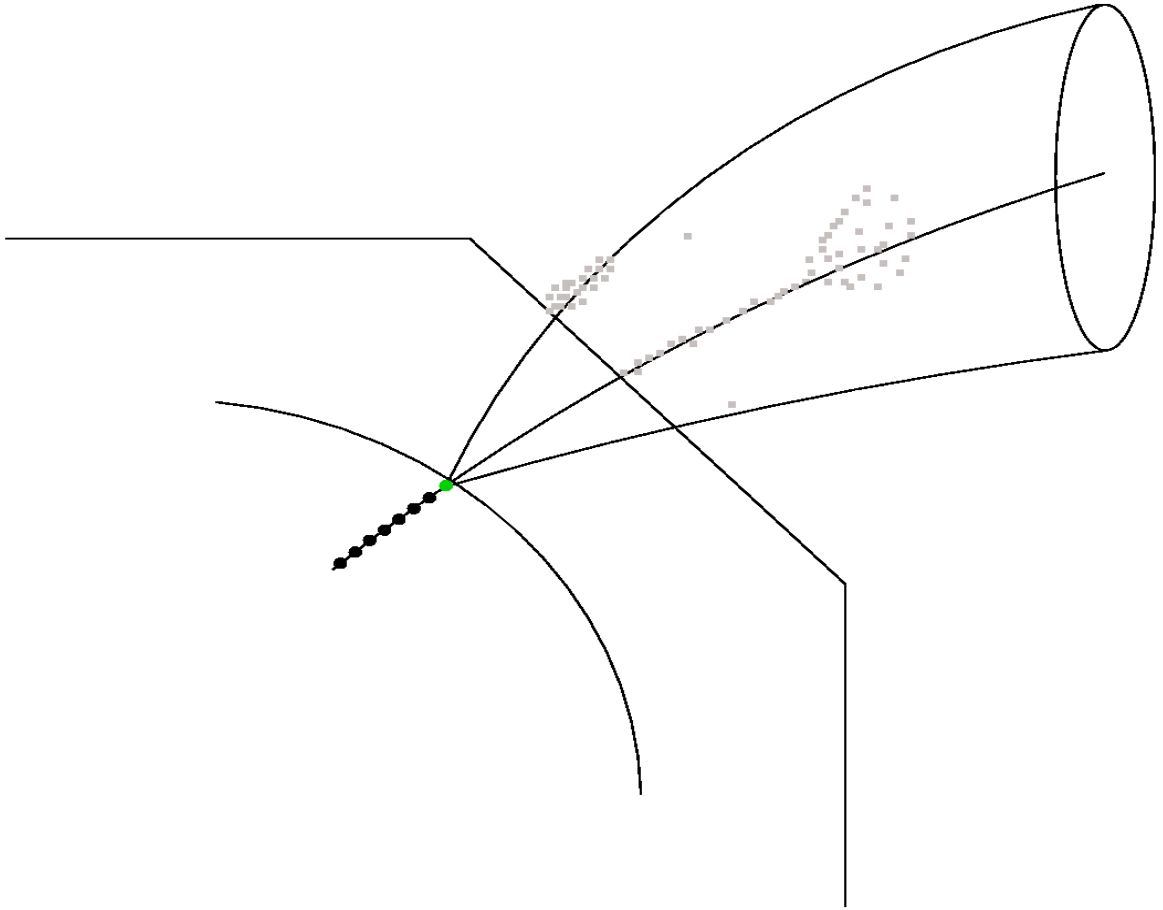


Assign related Calorimeter Hits



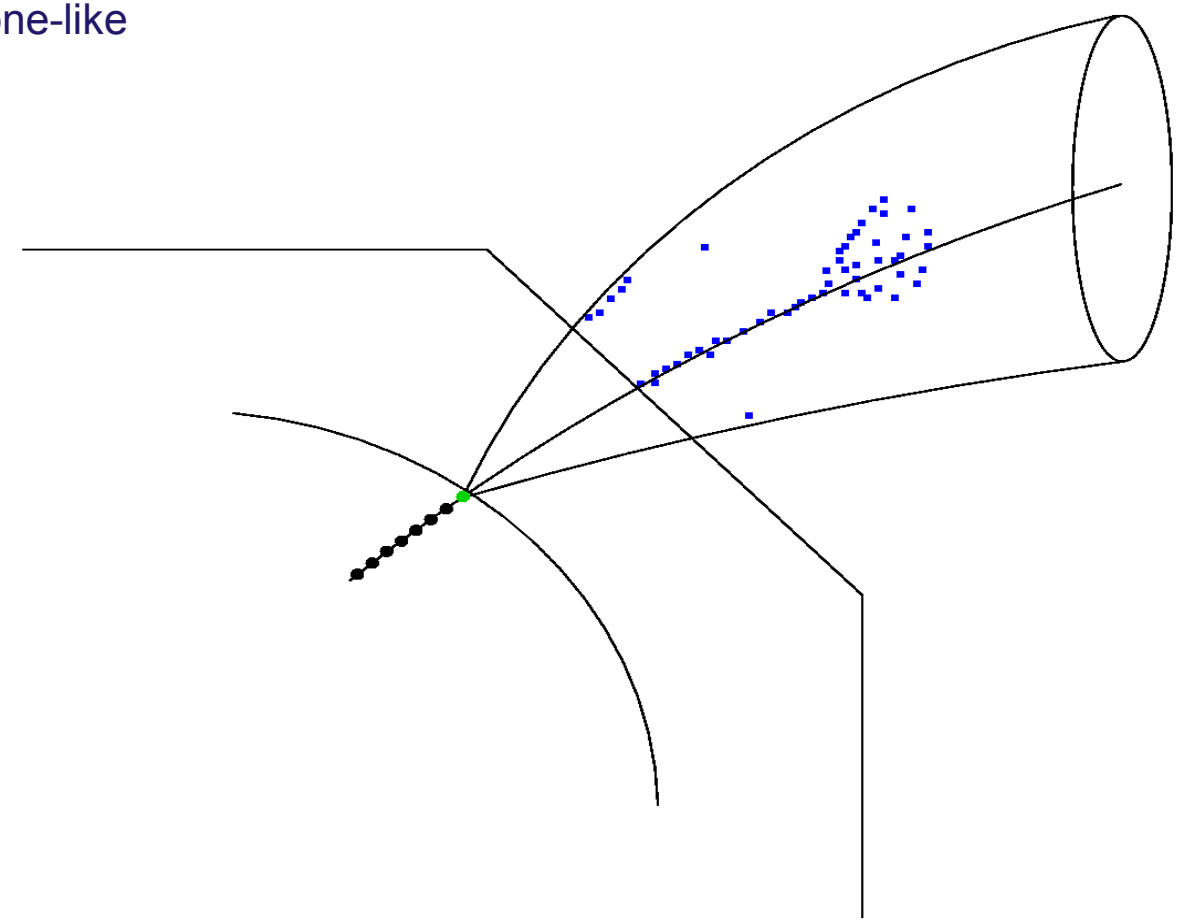
Assign related Calorimeter Hits

- put cone-like tube around extrapolated trajectory



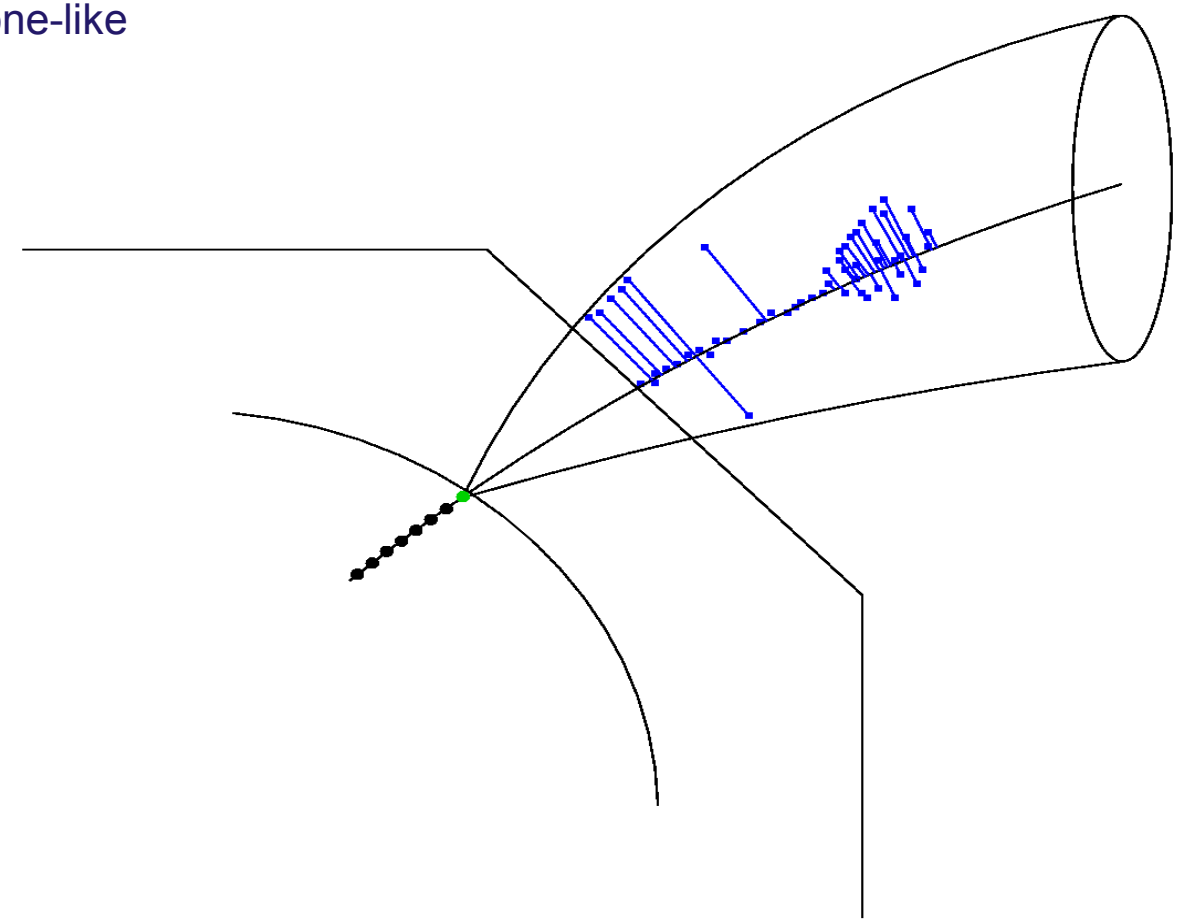
Assign related Calorimeter Hits

- put cone-like tube around extrapolated trajectory
- cut calorimeter hits outside cone-like tube



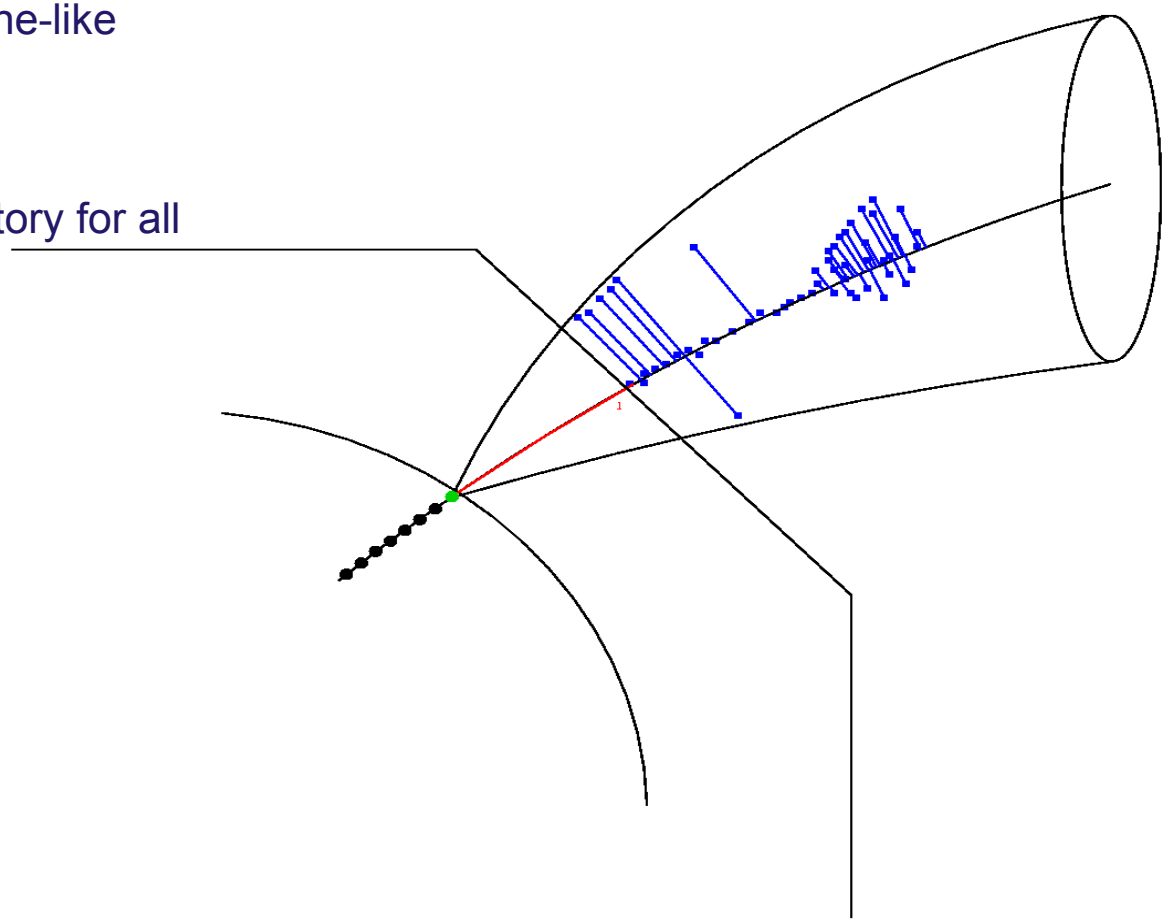
Assign related Calorimeter Hits

- put cone-like tube around extrapolated trajectory
- cut calorimeter hits outside cone-like tube
- project all hits on trajectory



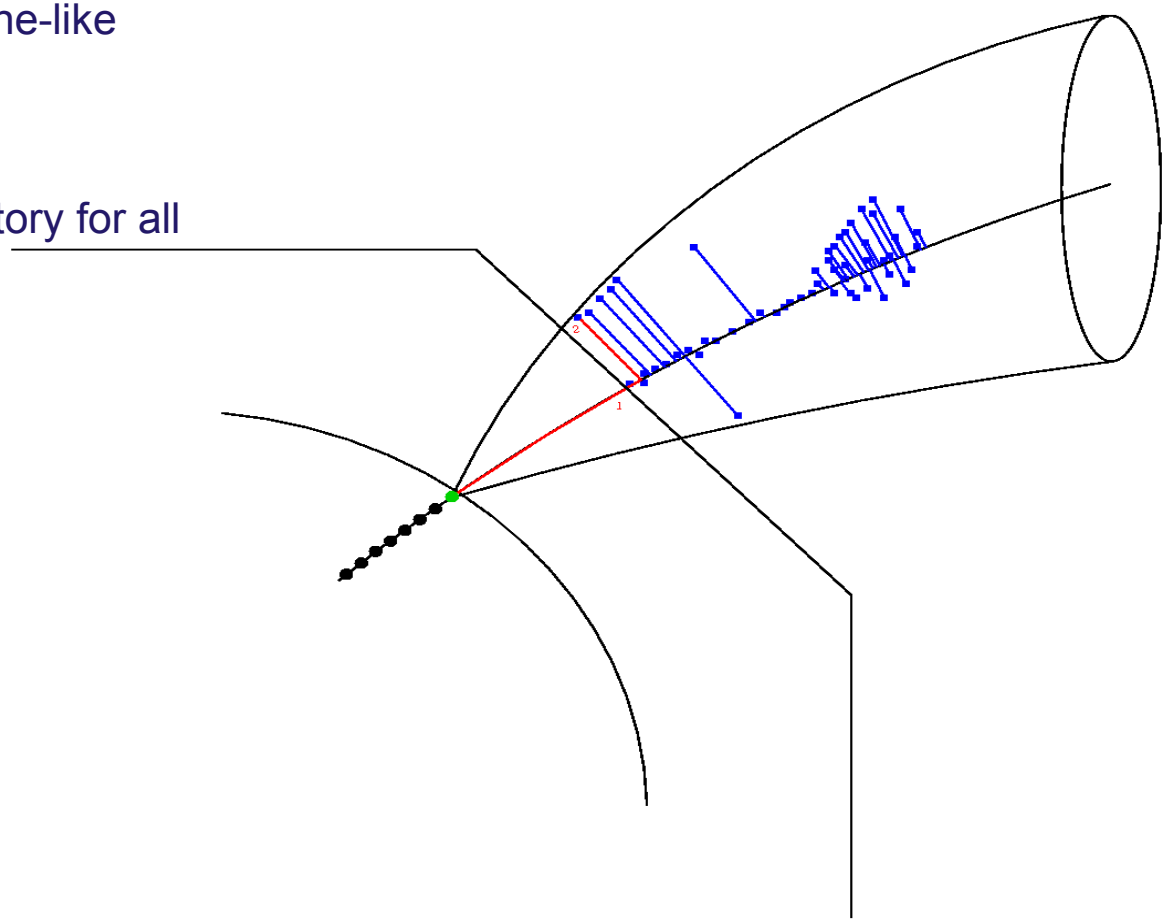
Assign related Calorimeter Hits

- put cone-like tube around extrapolated trajectory
- cut calorimeter hits outside cone-like tube
- project all hits on trajectory
- calculate path length on trajectory for all hits



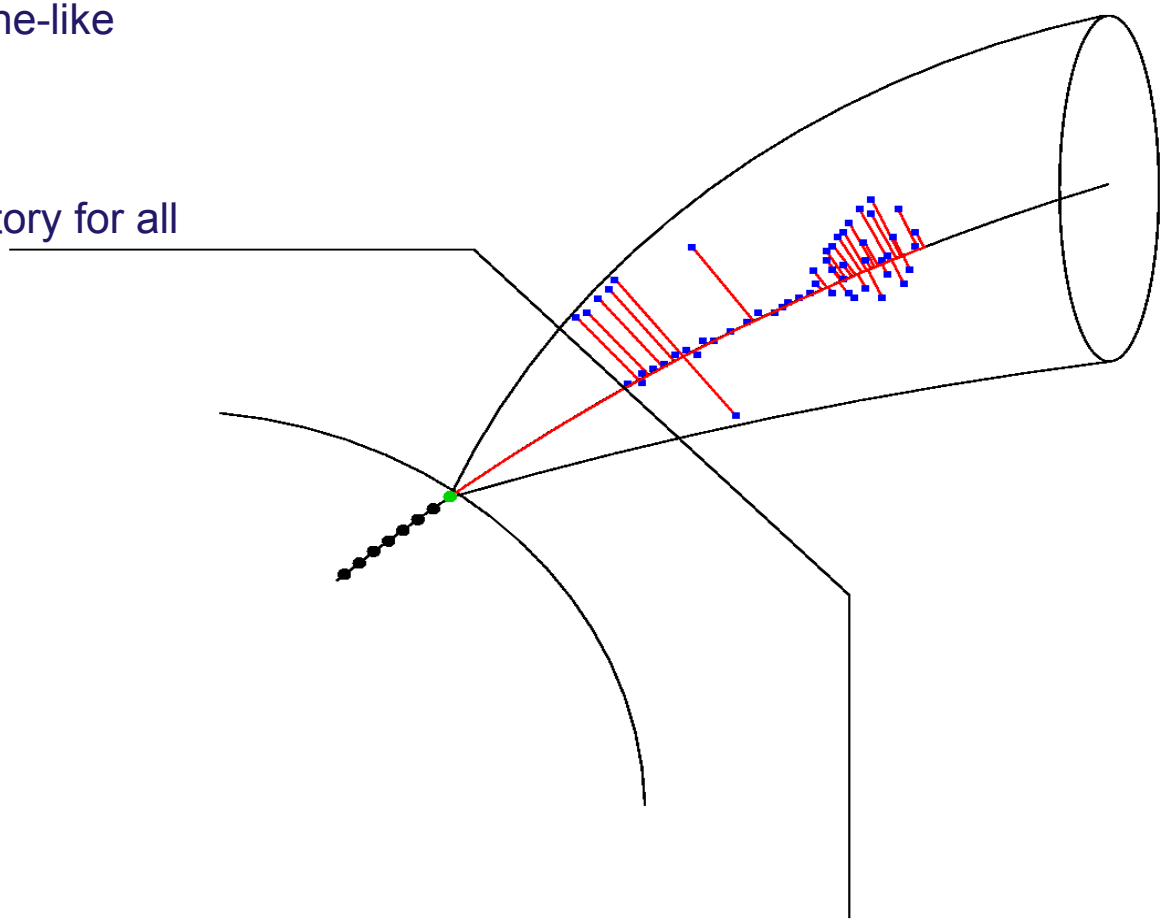
Assign related Calorimeter Hits

- put cone-like tube around extrapolated trajectory
- cut calorimeter hits outside cone-like tube
- project all hits on trajectory
- calculate path length on trajectory for all hits

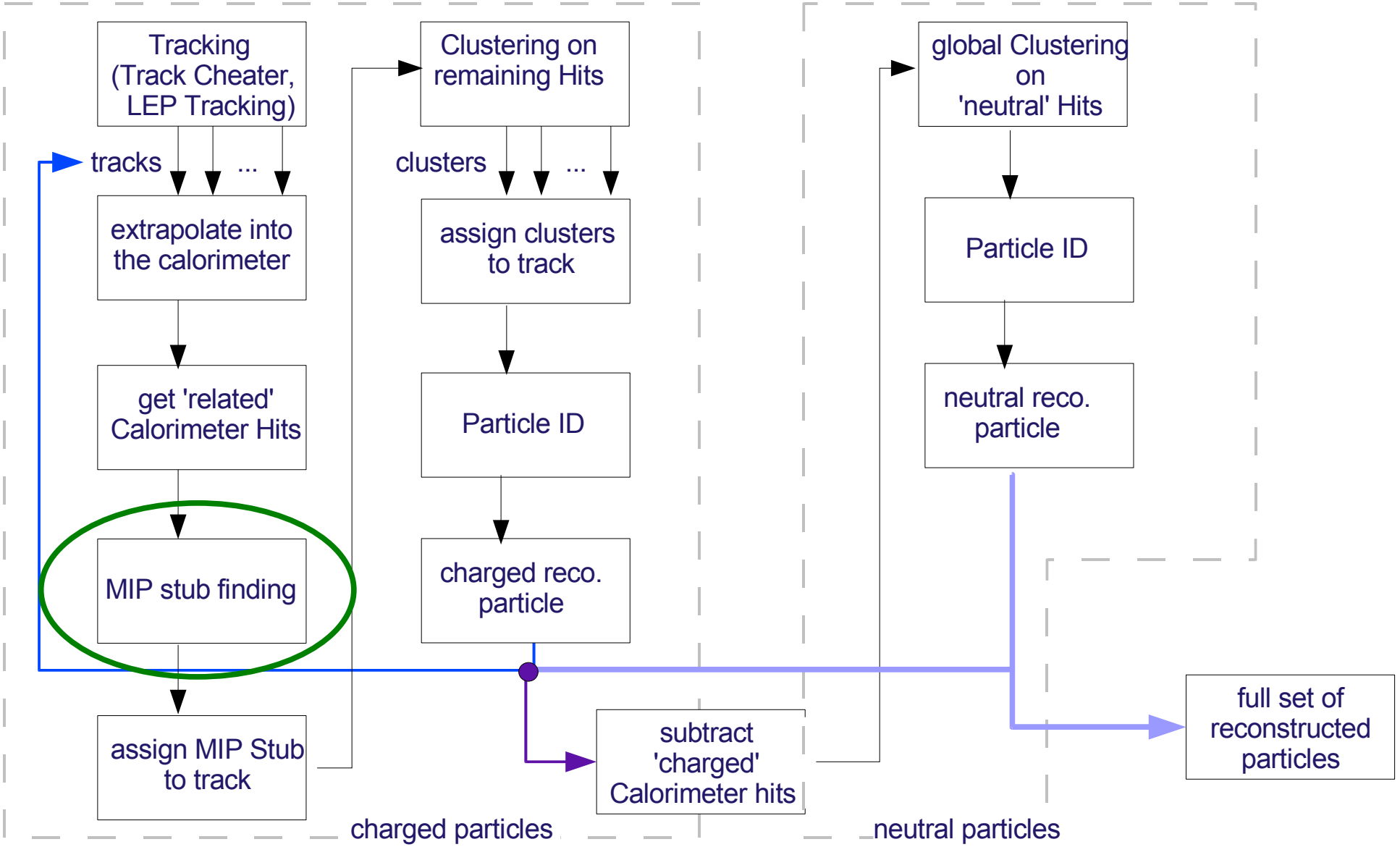


Assign related Calorimeter Hits

- put cone-like tube around extrapolated trajectory
- cut calorimeter hits outside cone-like tube
- project all hits on trajectory
- calculate path length on trajectory for all hits
- sort hits by their path lengths

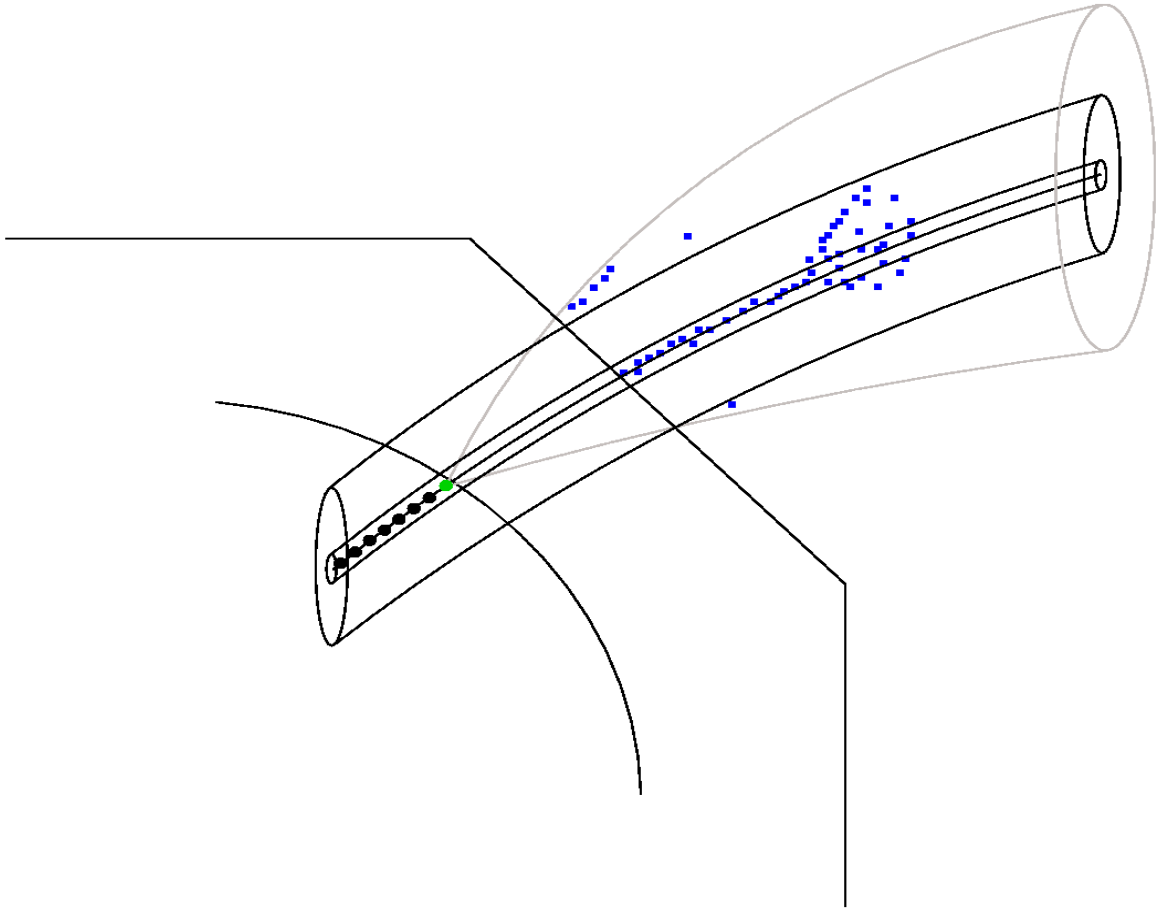


Details on MIP stub finding



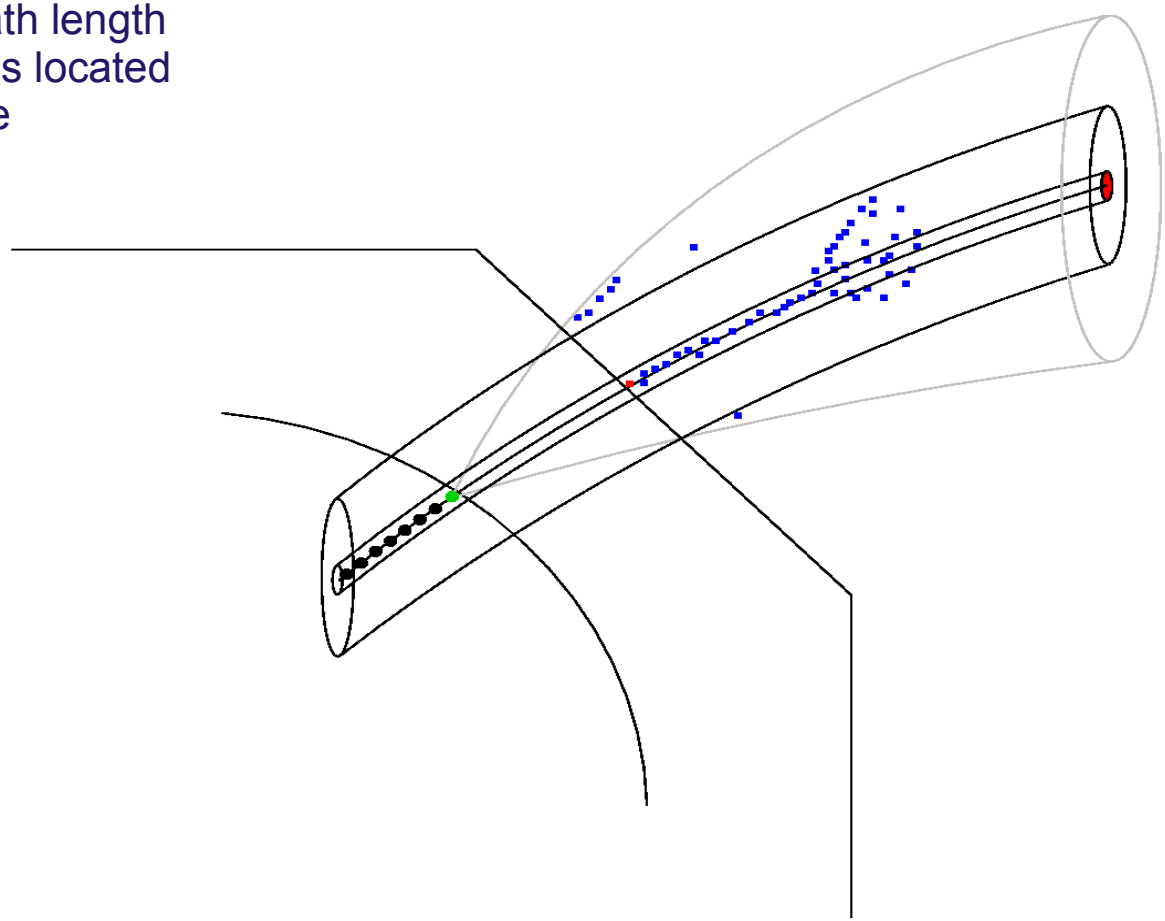
Details on MIP stub finding

- put two cylindrical tubes around extrapolated trajectory



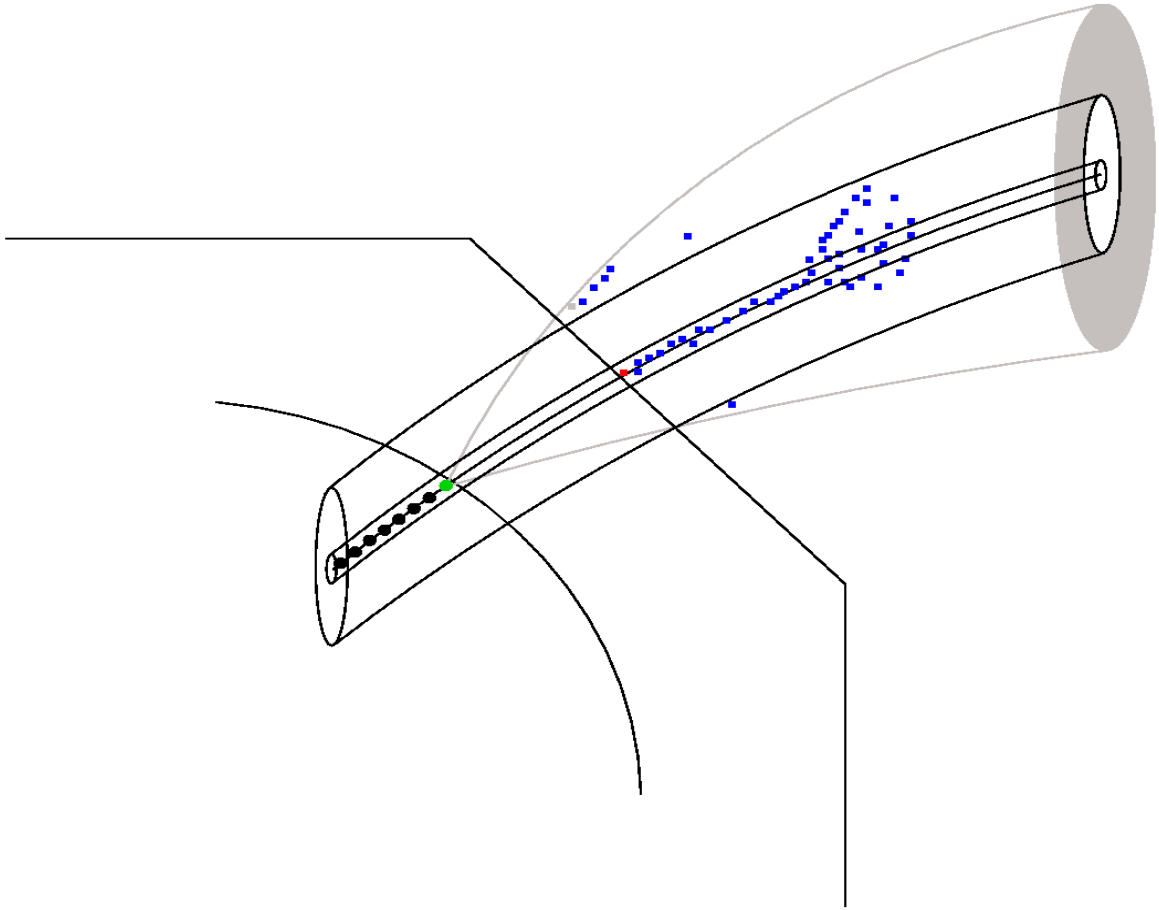
Details on MIP stub finding

- put two cylindrical tubes around extrapolated trajectory
- take first hit according to its path length and add it to the MIP stub if it is located inside the inner cylindrical tube



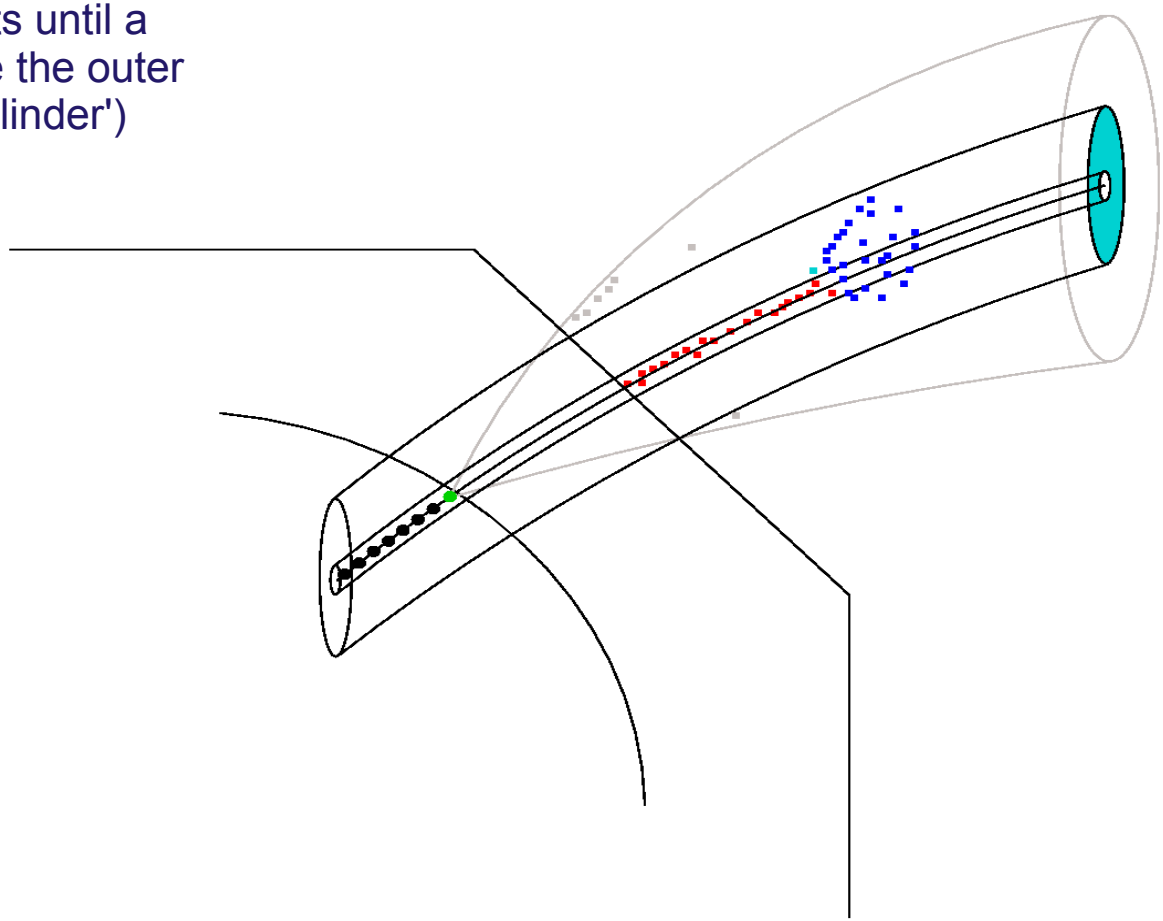
Details on MIP stub finding

- take the next hit and discard it if it is located outside the outer tube



Details on MIP stub finding

- take the next hit and discard it if it is located outside the outer tube
- repeat this procedure for all hits until a hit outside the inner and inside the outer cylinder tube is found ('veto-cylinder')

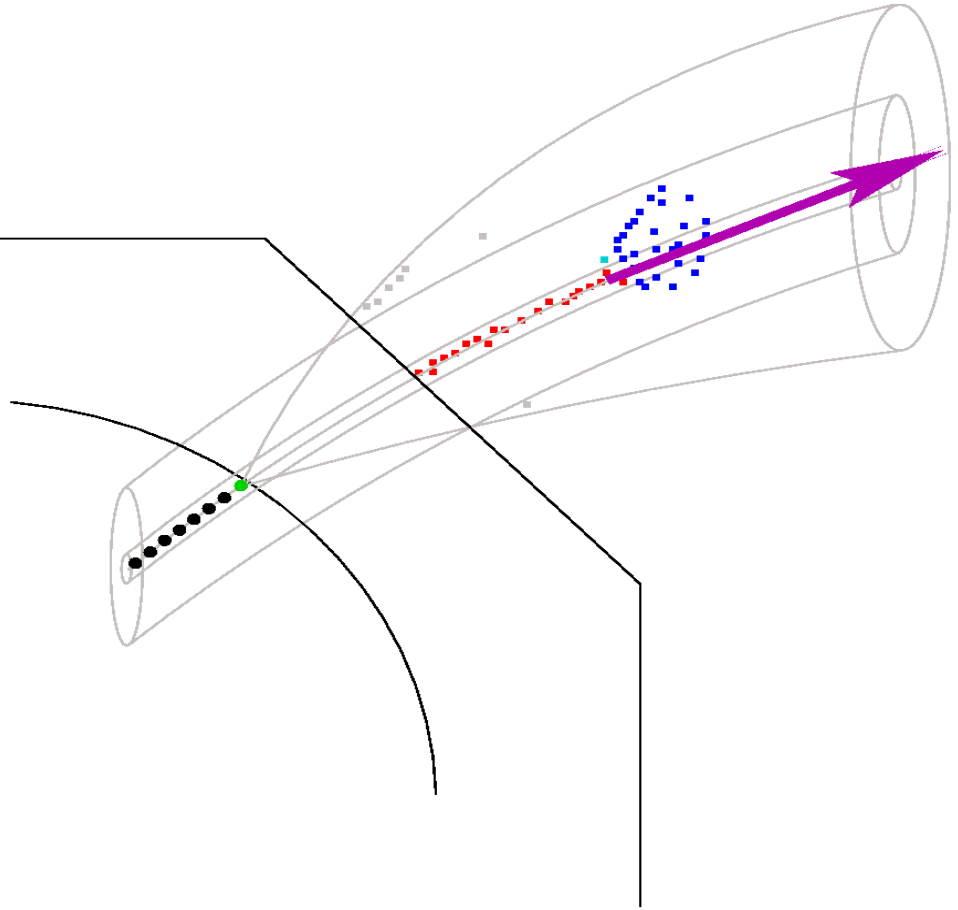


Details on MIP stub finding

- take the next hit and discard it if it is located outside the outer tube
- repeat this procedure for all hits until a hit outside the inner and inside the outer cylinder tube is found ('veto-cylinder')
- stop the MIP stub finding
- take the projection of the last hit collected for the MIP stub as a start point for clustering
- take the direction (tangent) of this point as a start direction for clustering

→ also done with amplitude information (MIP like) and a cut on maximal energy in 'veto-cylinder'

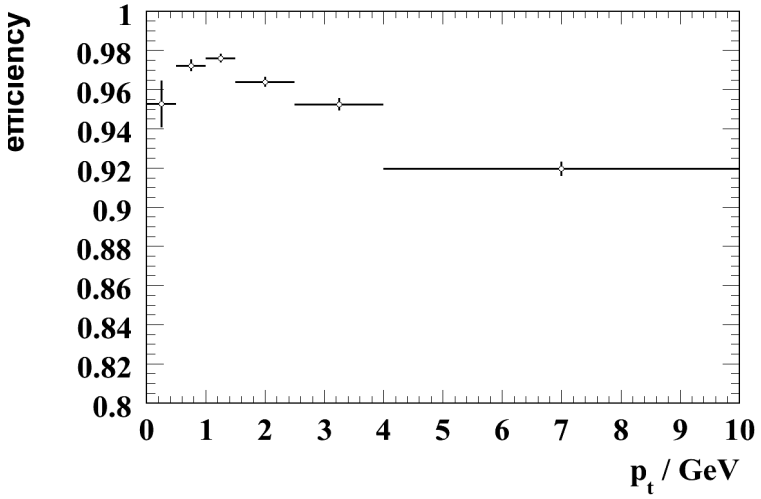
→ no big influence on efficiencies

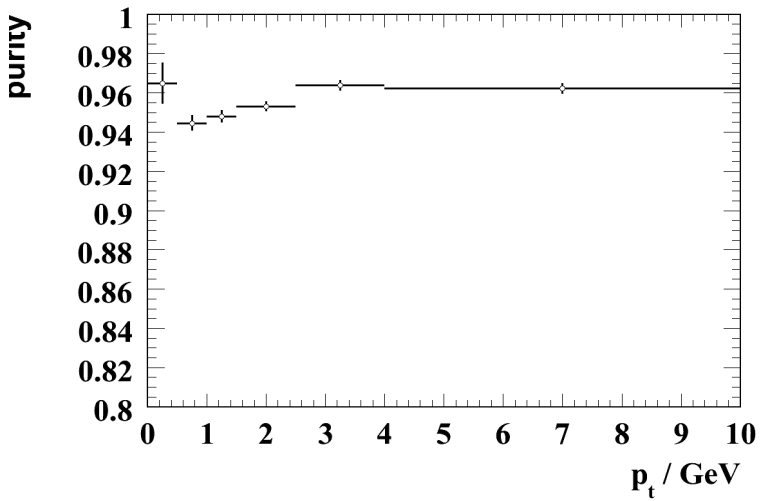


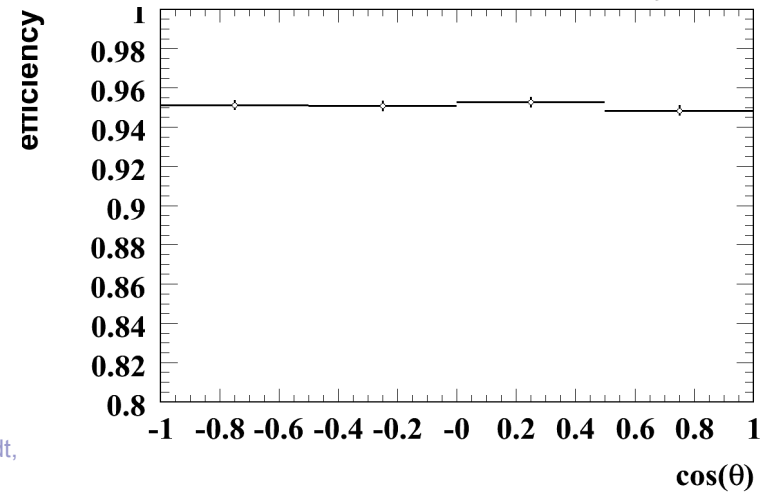
Details on MIP stub finding

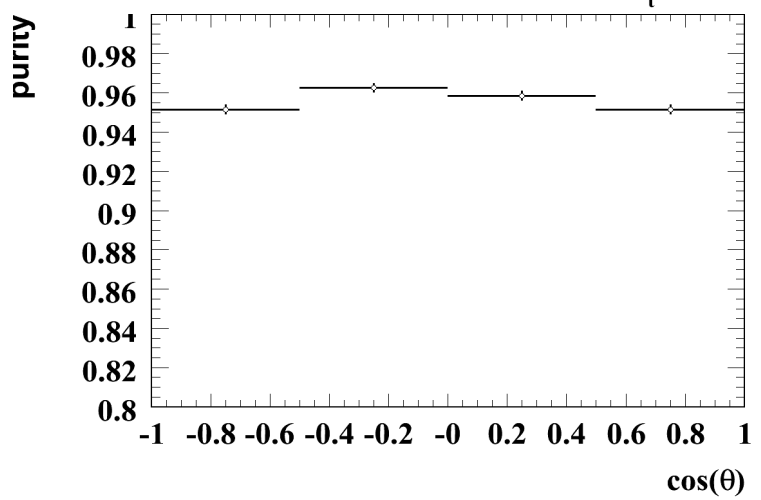
Comparison with MC:

- efficiency and purity vs. p_t and cos(θ):
- overall efficiency >90%, overall purity >90%

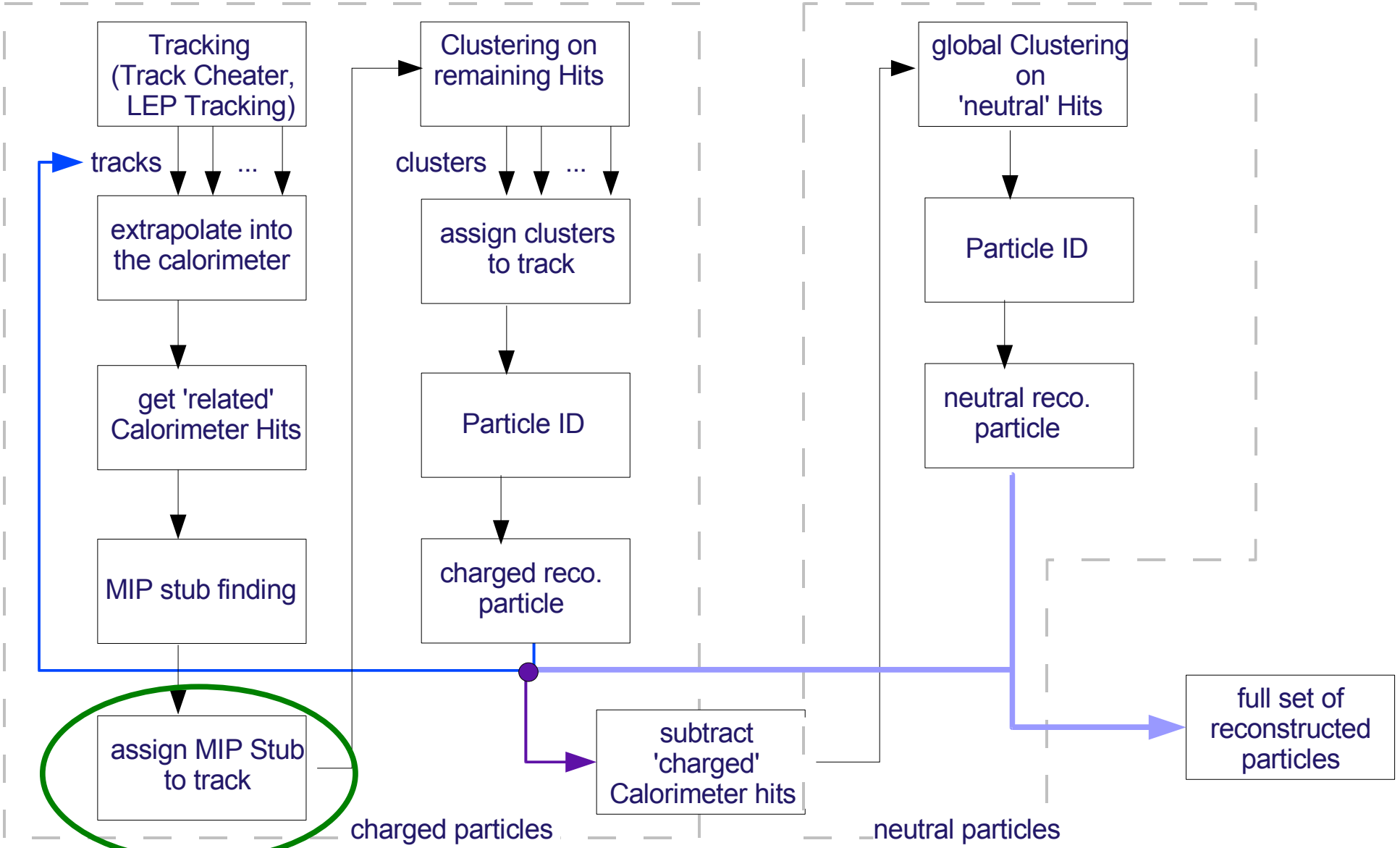




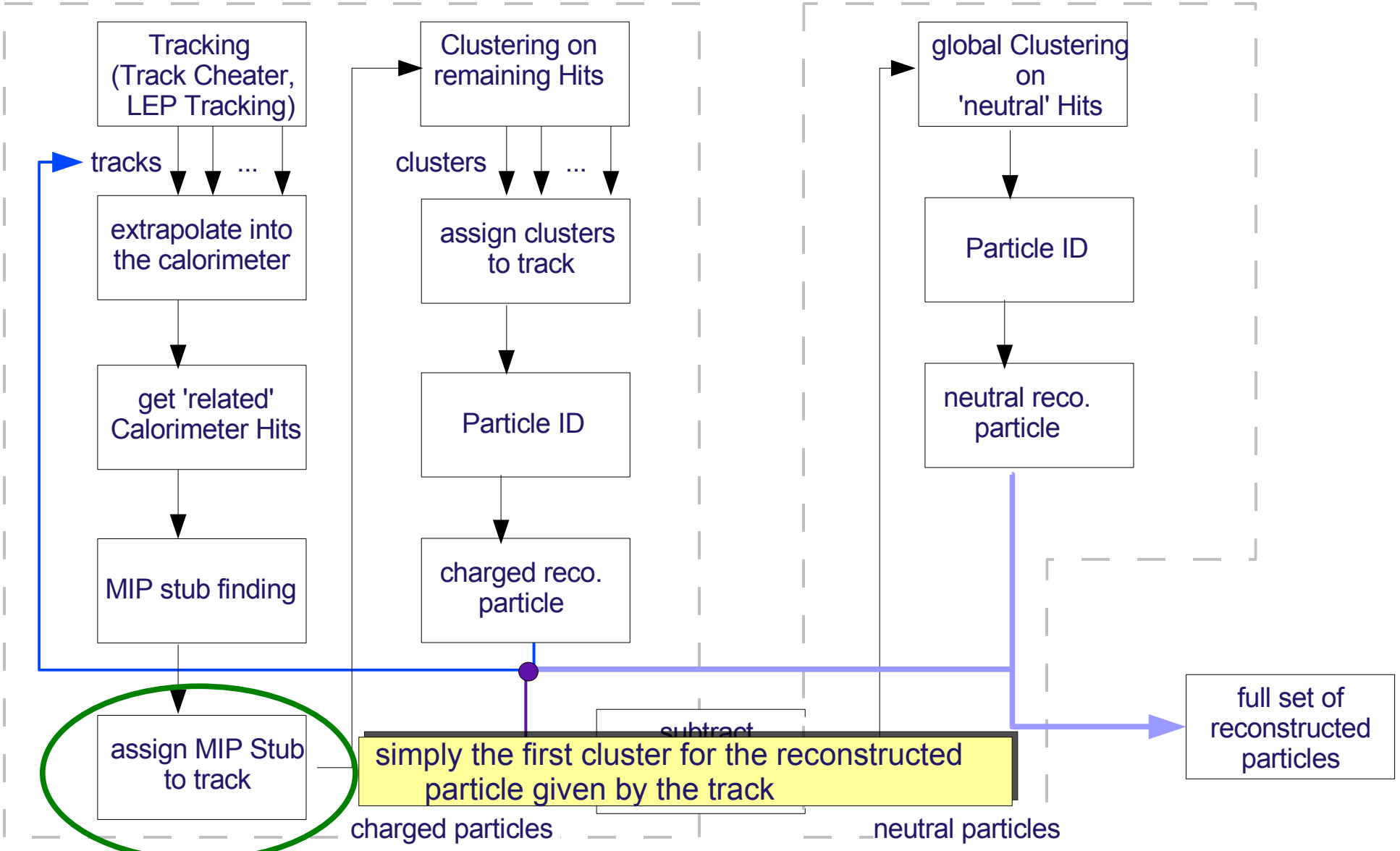




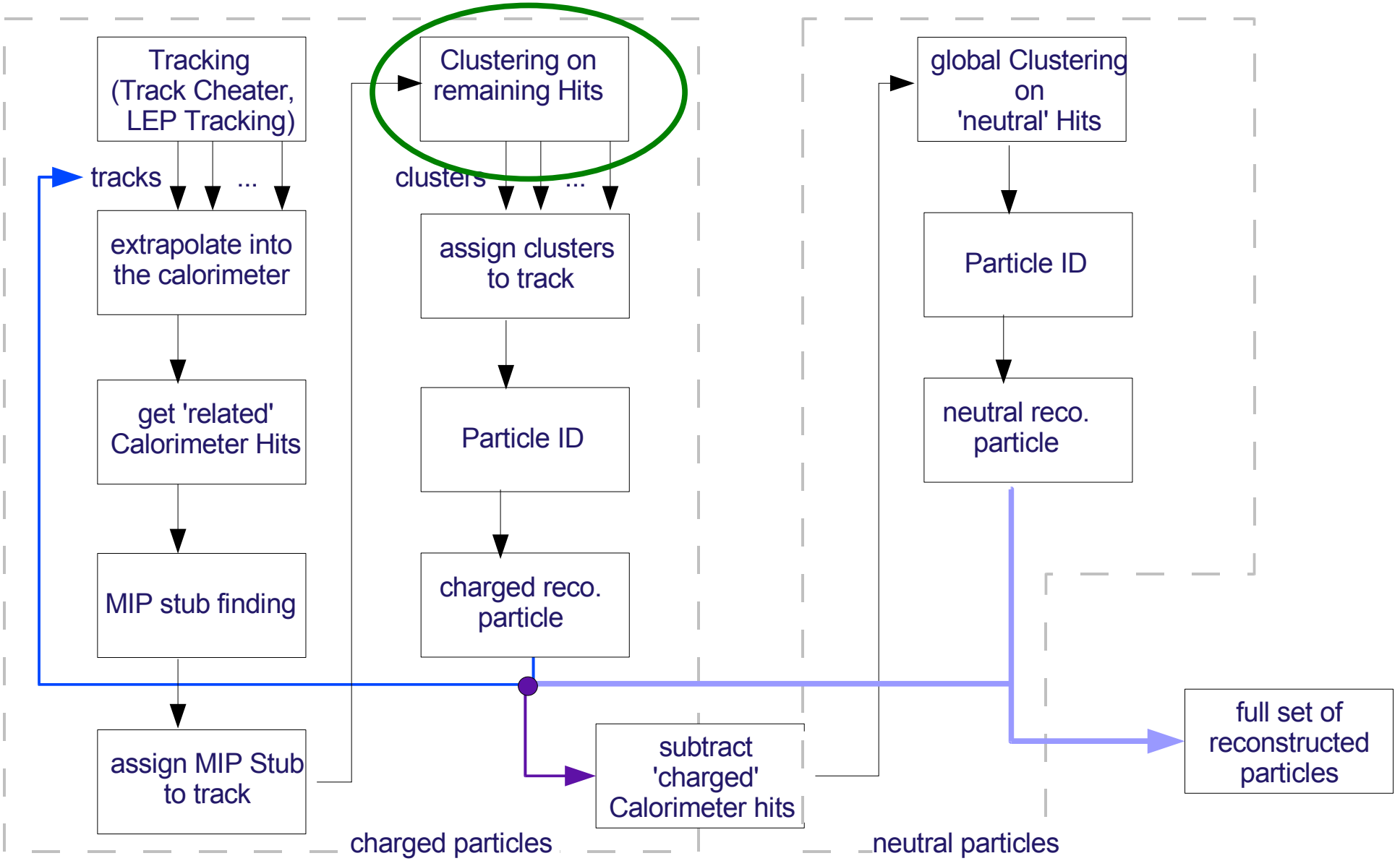
assign MIP stub to track



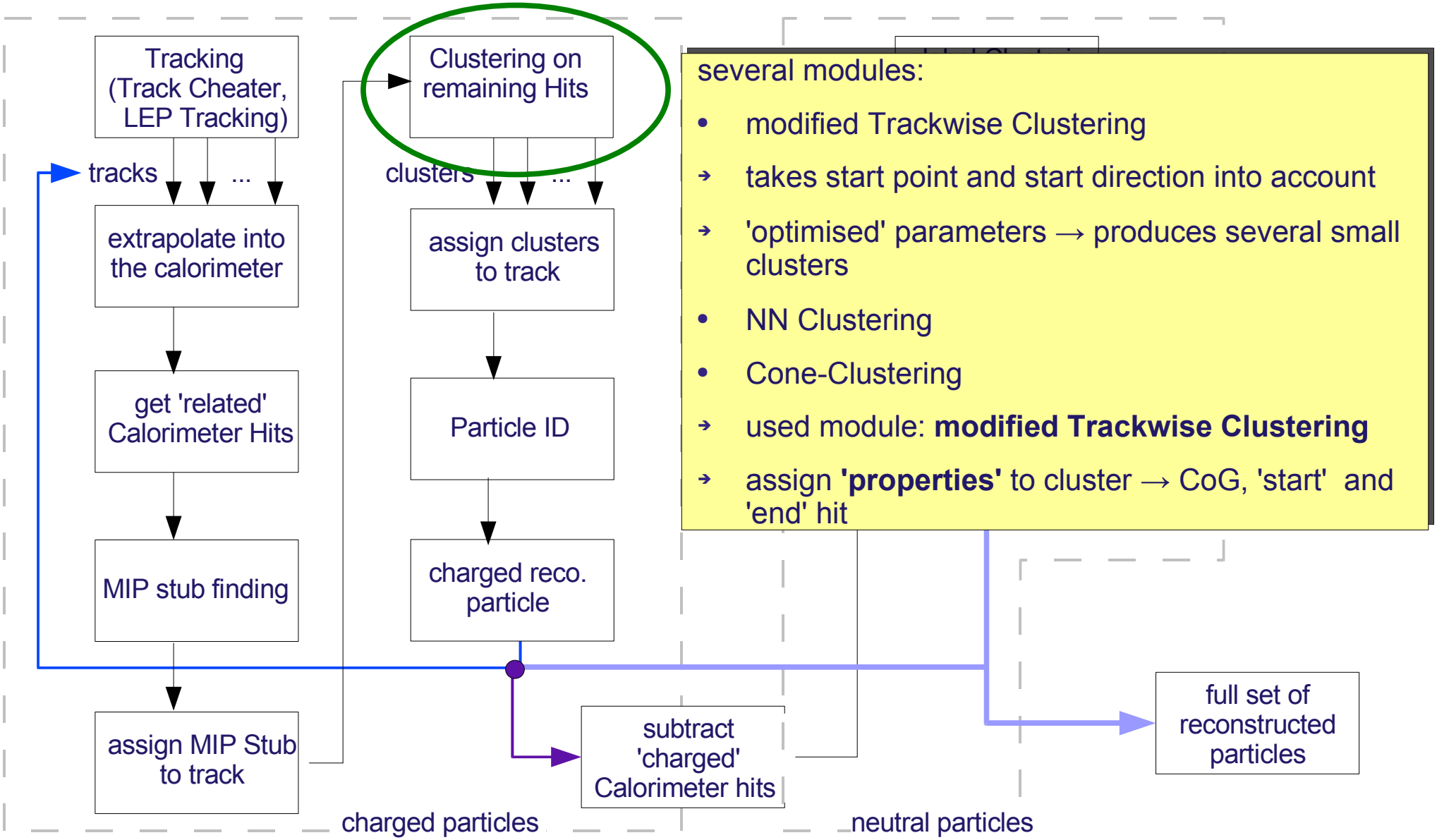
assign MIP stub to track



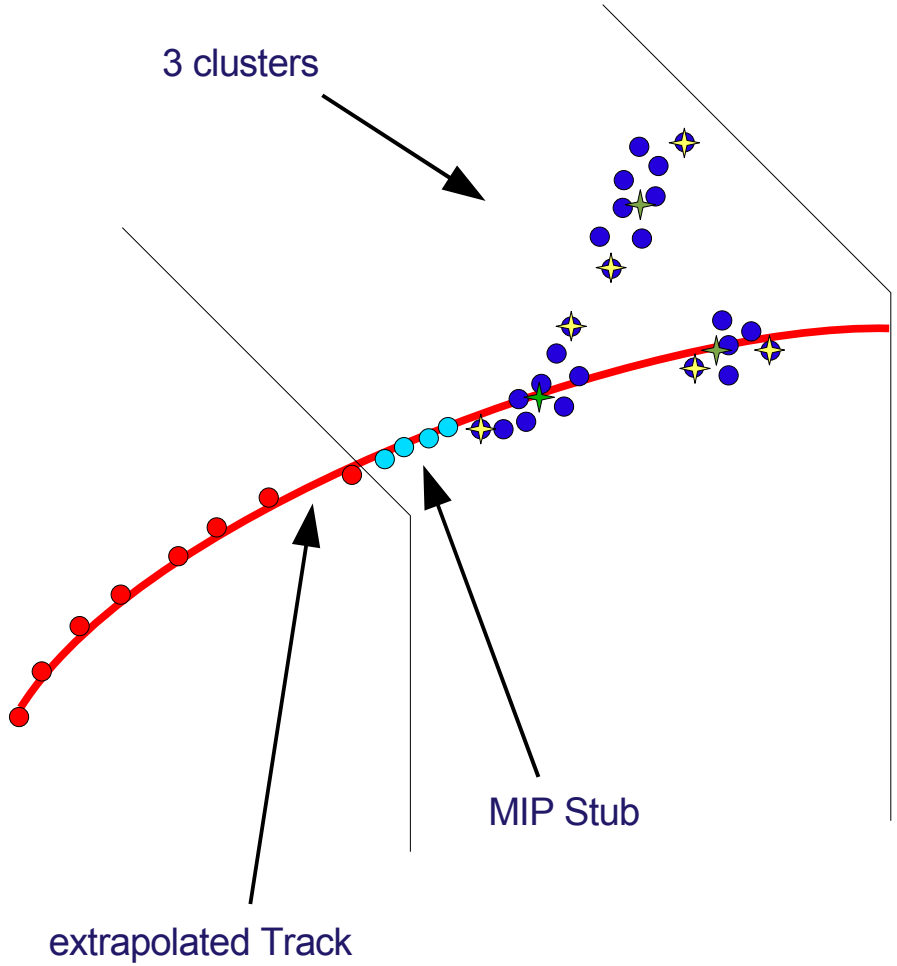
Clustering


Tracking (Track Cheater, LEP Tracking)
tracks
...
extrapolate into the calorimeter
get 'related' Calorimeter Hits
MIP stub finding
assign MIP Stub to track
Clustering on remaining Hits
clusters
...
assign clusters to track
Particle ID
charged reco. particle
subtract 'charged' Calorimeter hits
charged particles
global Clustering on 'neutral' Hits
Particle ID
neutral reco. particle
neutral particles
full set of reconstructed particles

Clustering


Tracking (Track Cheater, LEP Tracking)
tracks
...
extrapolate into the calorimeter
get 'related' Calorimeter Hits
MIP stub finding
assign MIP Stub to track
Clustering on remaining Hits
clusters
...
assign clusters to track
Particle ID
charged reco. particle
subtract 'charged' Calorimeter hits
full set of reconstructed particles
charged particles
neutral particles
several modules:
• modified Trackwise Clustering
→ takes start point and start direction into account
→ 'optimised' parameters → produces several small clusters
• NN Clustering
• Cone-Clustering
→ used module: **modified Trackwise Clustering**
→ assign **'properties'** to cluster → CoG, 'start' and 'end' hit

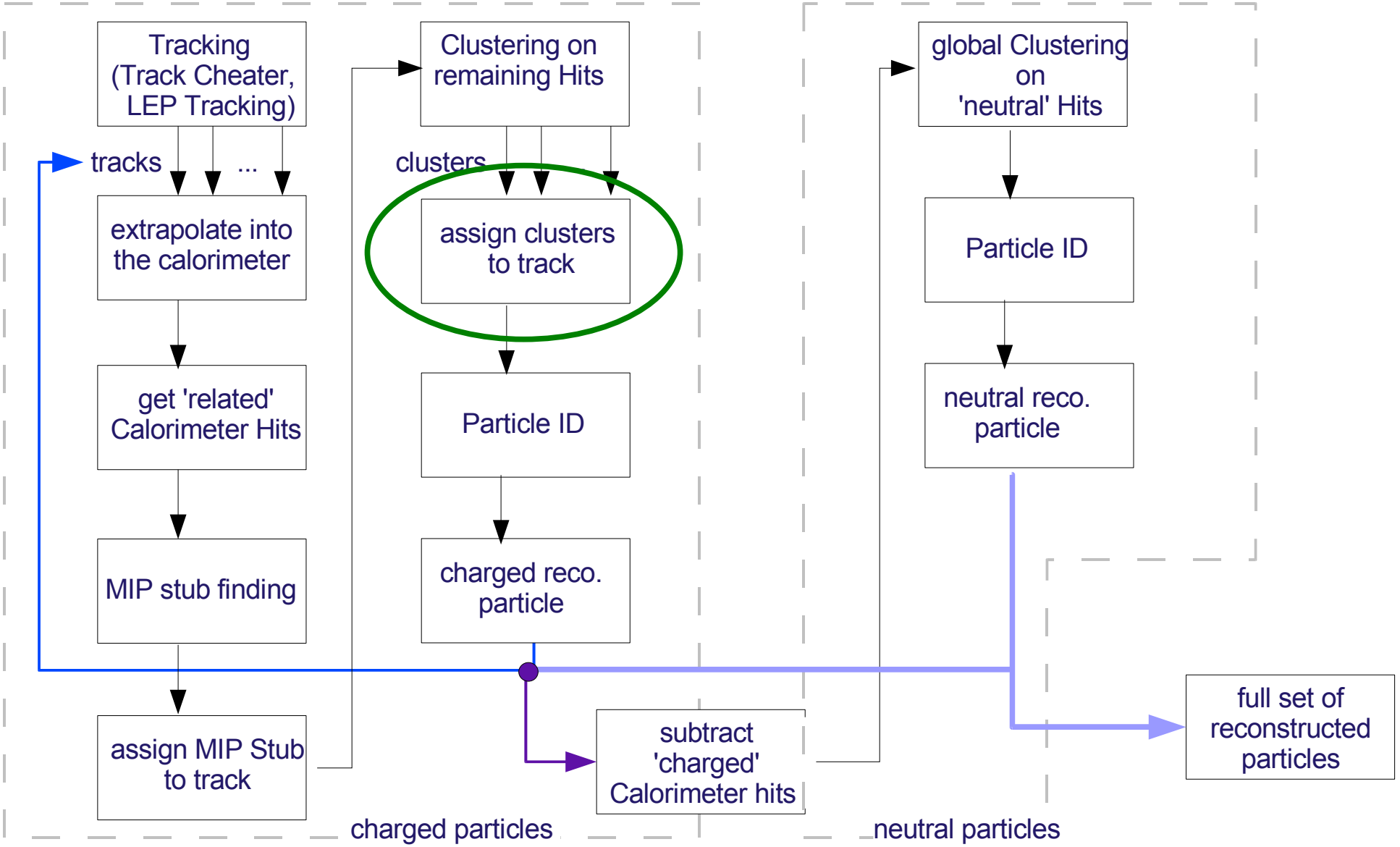
Properties of Clusters



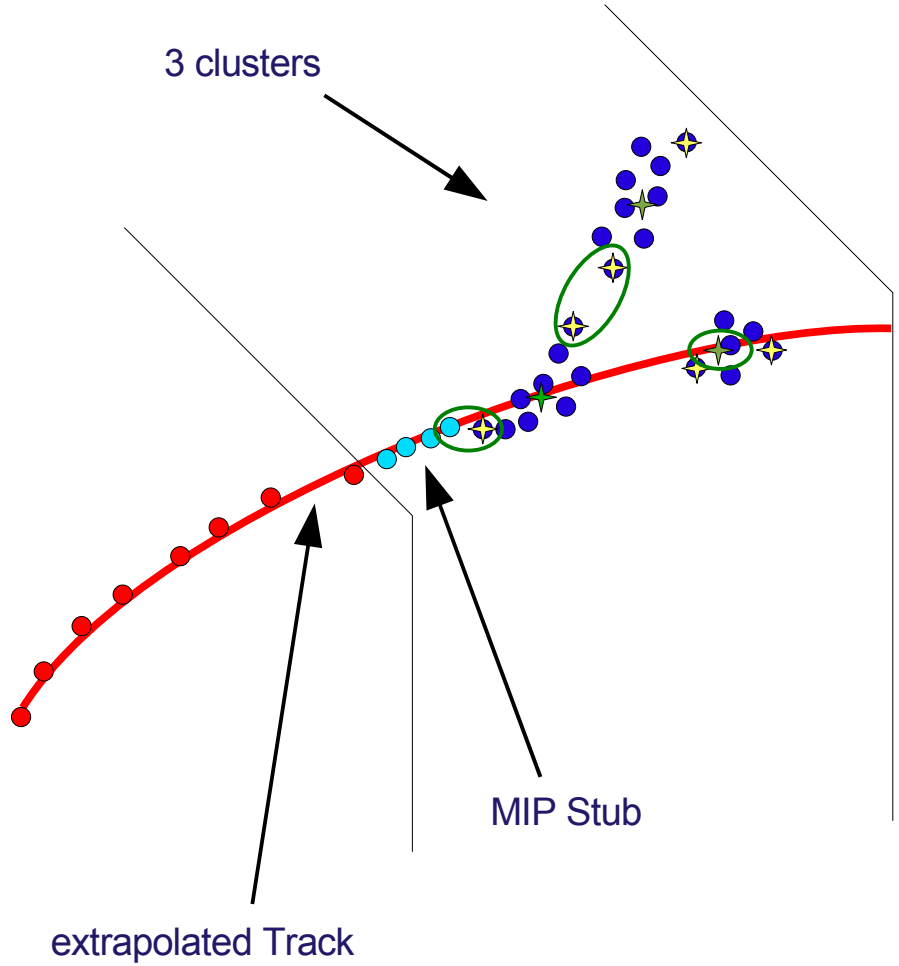
- calculate Center of Gravity (CoG) for each cluster (✦)
- calculate start and end hit for each cluster (✧)

➔ simply smallest and largest path length of Calorimeter Hit on trajectory/helix

Assign Clusters to Track



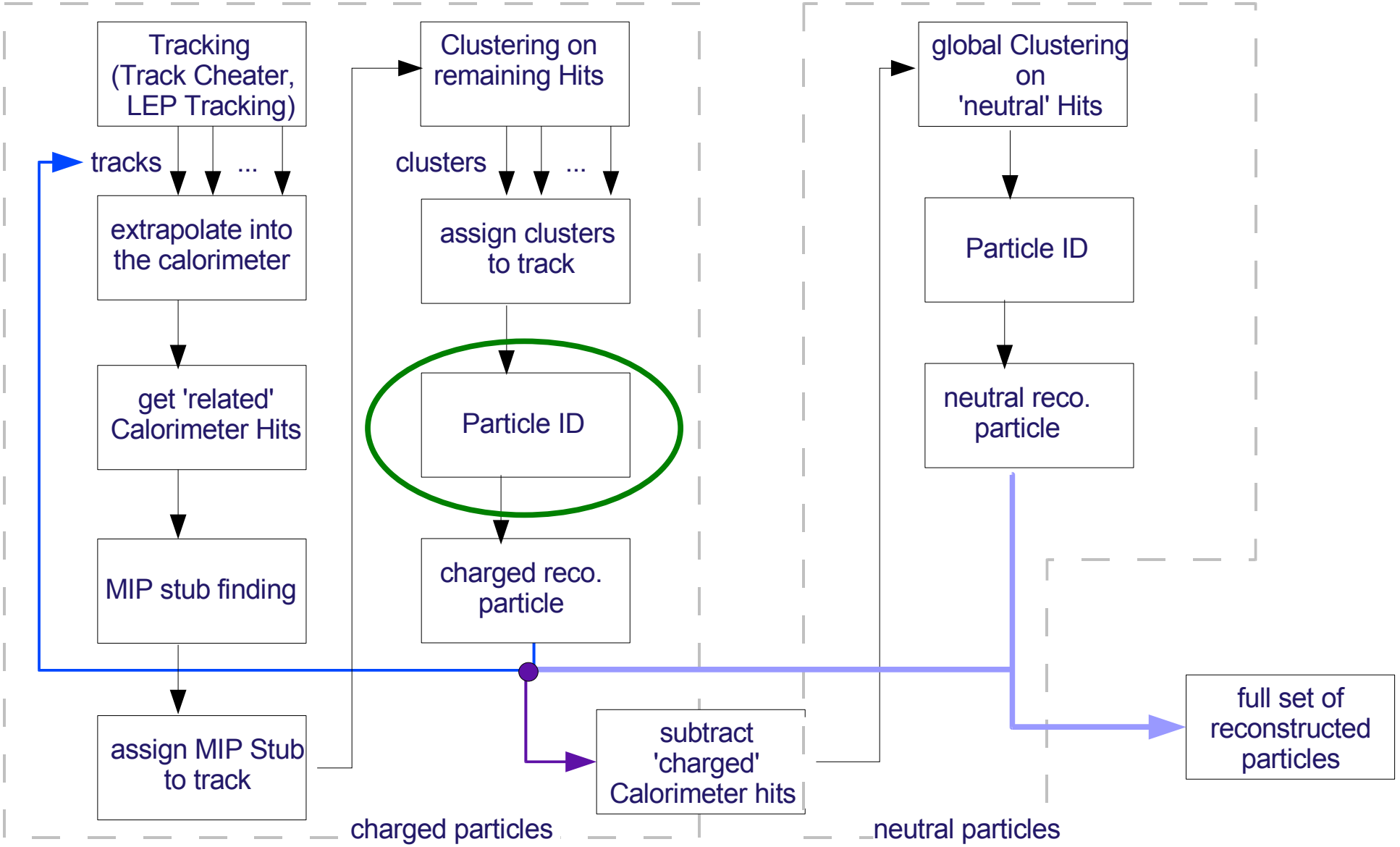
Assign Clusters to Track



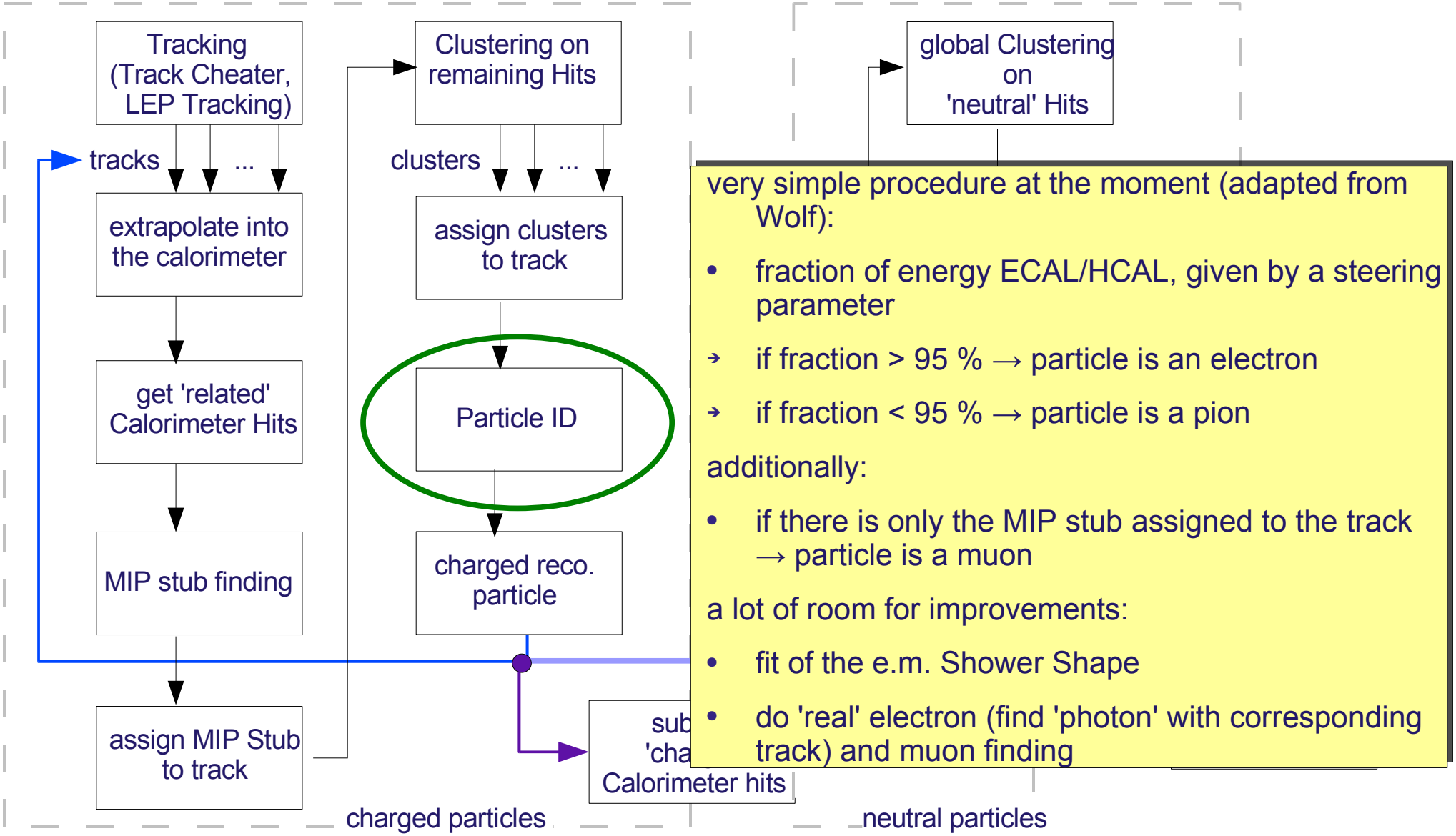
assign Cluster to track if

- distance between end point of cluster i to start point of cluster j is smaller than a given limit
- ➔ limit depends on sampling fraction
- distance of CoG to extrapolated track is smaller than a given limit
- some more geometrical conditions

Particle ID


Tracking (Track Cheater, LEP Tracking)
tracks
...
extrapolate into the calorimeter
get 'related' Calorimeter Hits
MIP stub finding
assign MIP Stub to track
Clustering on remaining Hits
clusters
...
assign clusters to track
Particle ID
charged reco. particle
subtract 'charged' Calorimeter hits
charged particles
global Clustering on 'neutral' Hits
Particle ID
neutral reco. particle
neutral particles
full set of reconstructed particles

Particle ID



very simple procedure at the moment (adapted from Wolf):

- fraction of energy ECAL/HCAL, given by a steering parameter
- → if fraction > 95 % → particle is an electron
- → if fraction < 95 % → particle is a pion

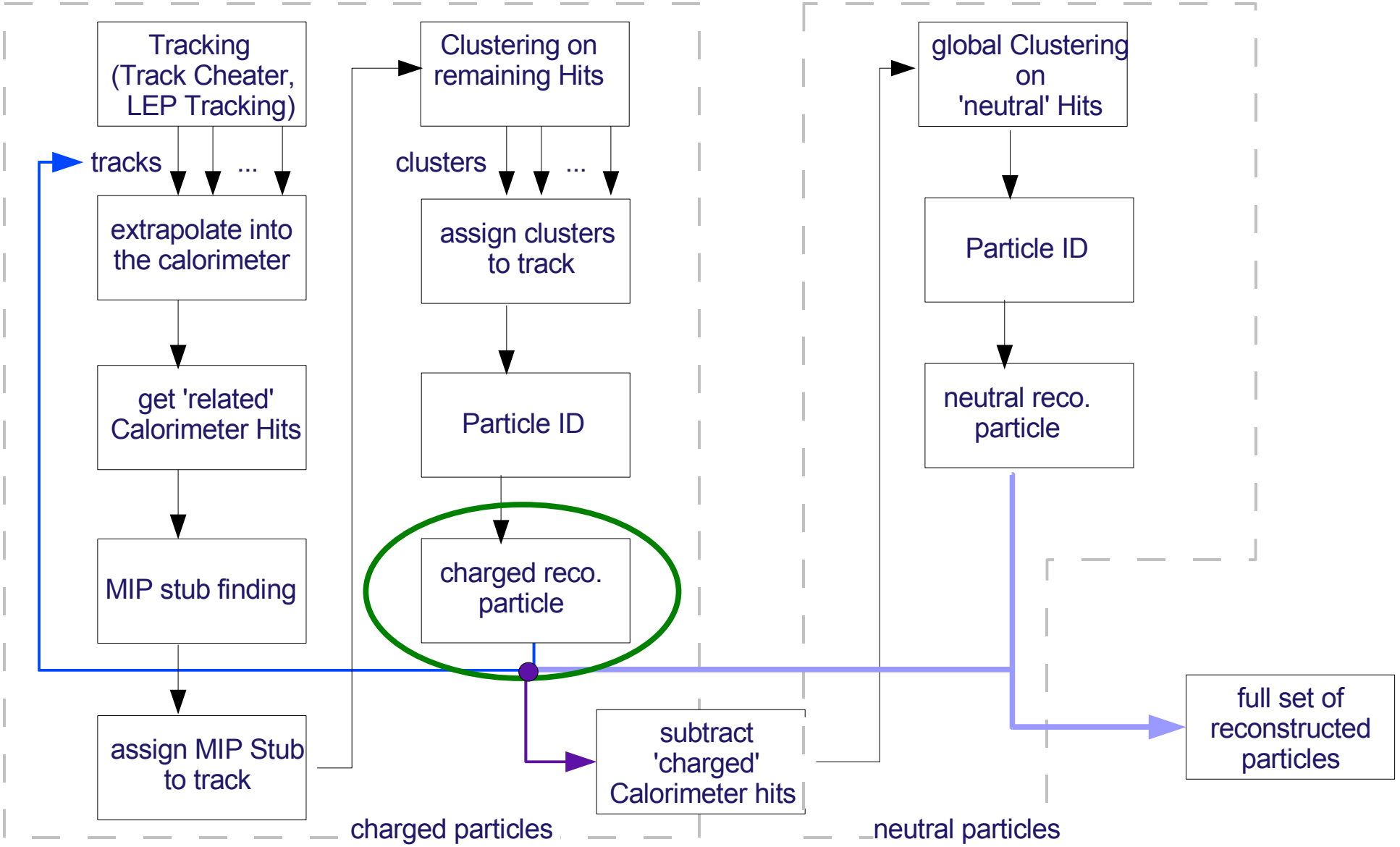
additionally:

- if there is only the MIP stub assigned to the track → particle is a muon

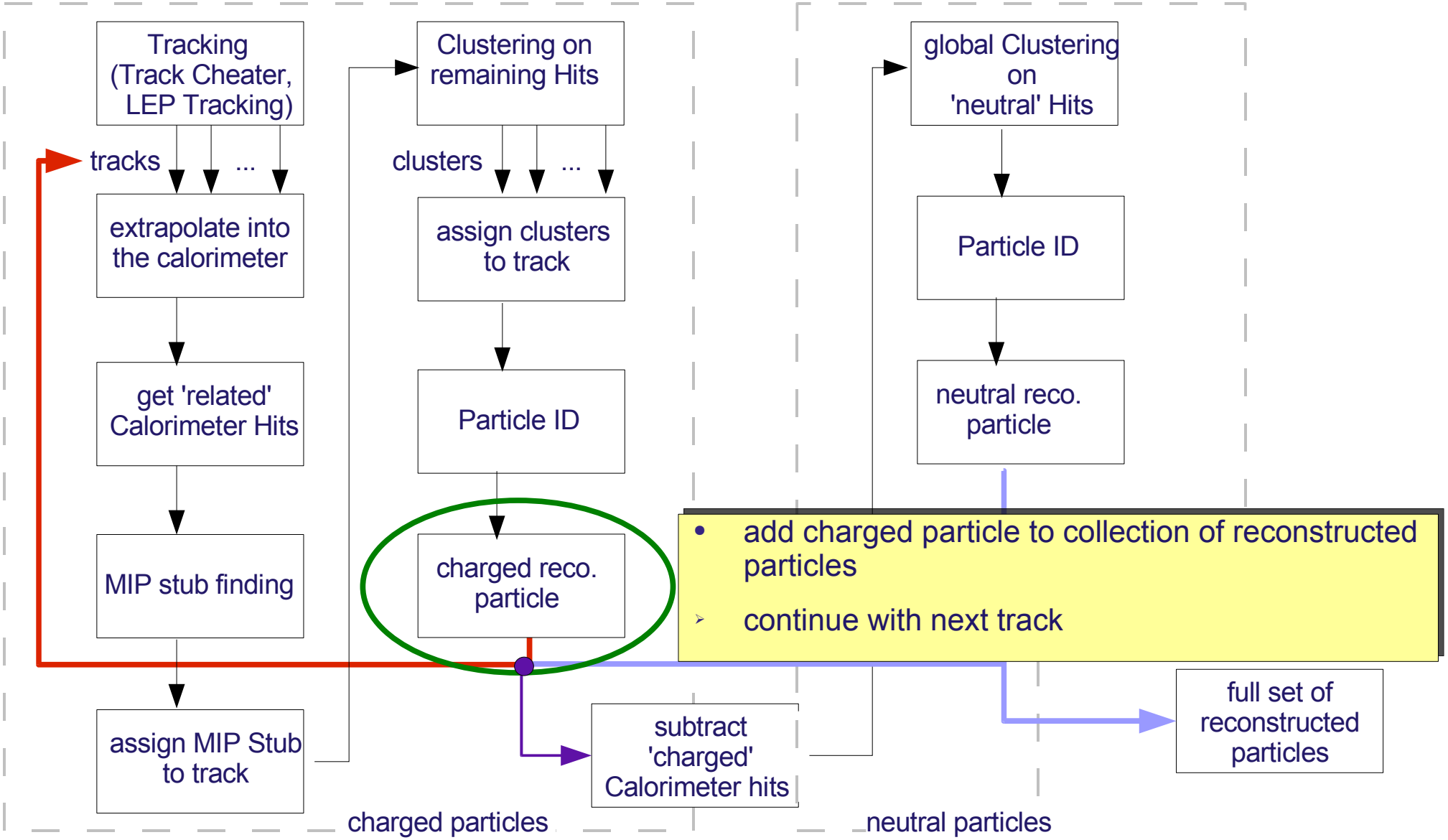
a lot of room for improvements:

- fit of the e.m. Shower Shape
- do 'real' electron (find 'photon' with corresponding track) and muon finding

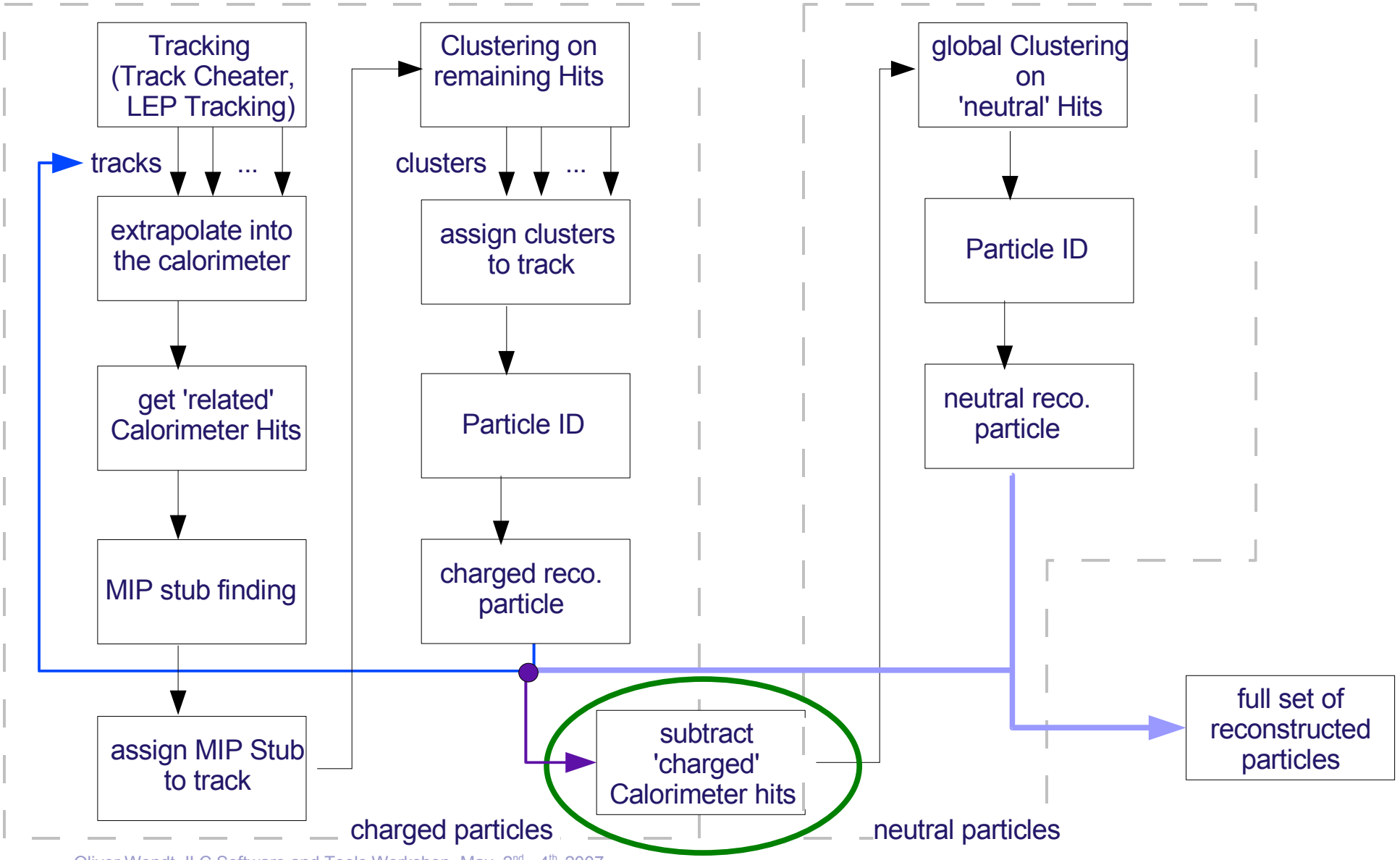
Charged Reconstructed Particle



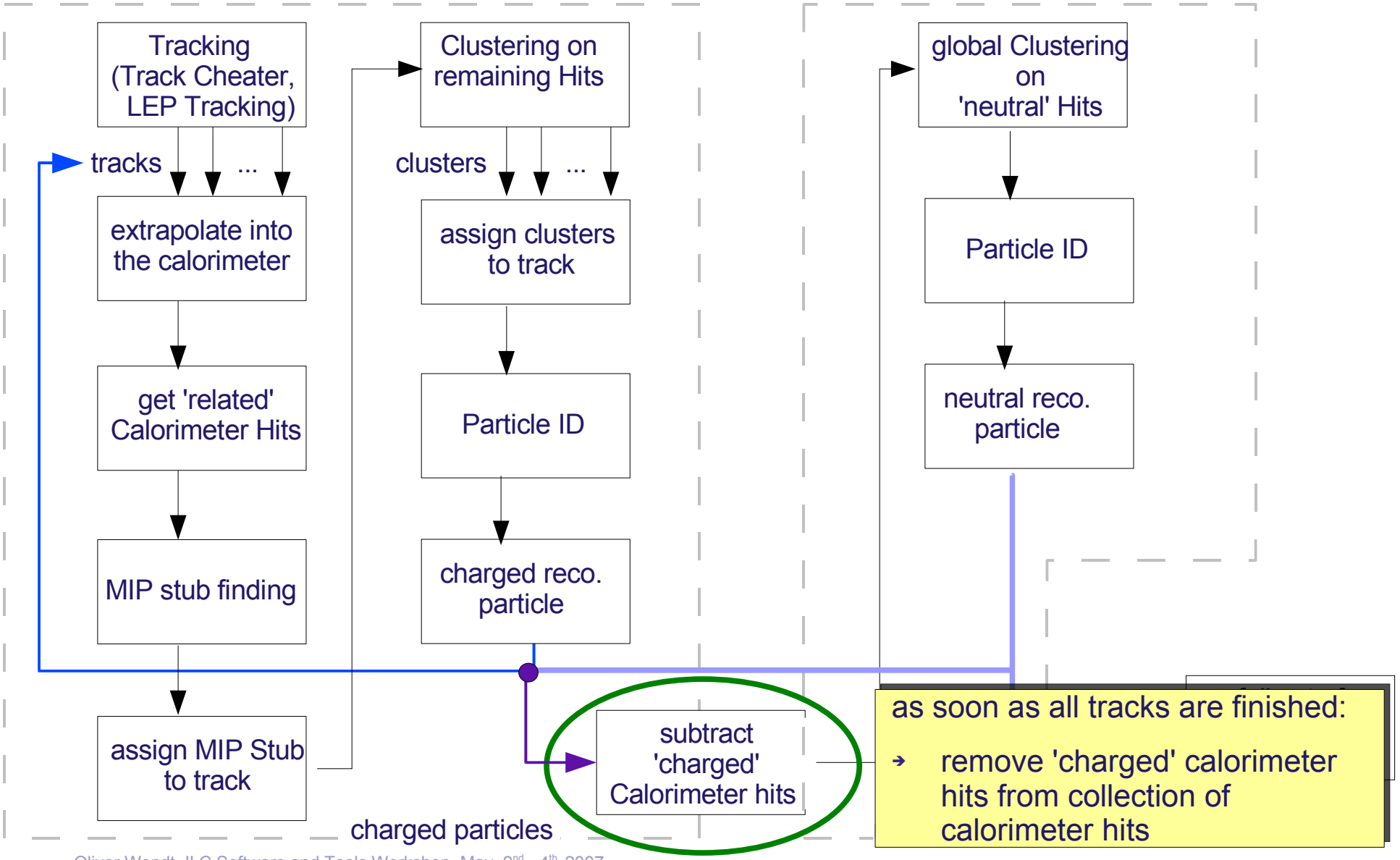
Charged Reconstructed Particle



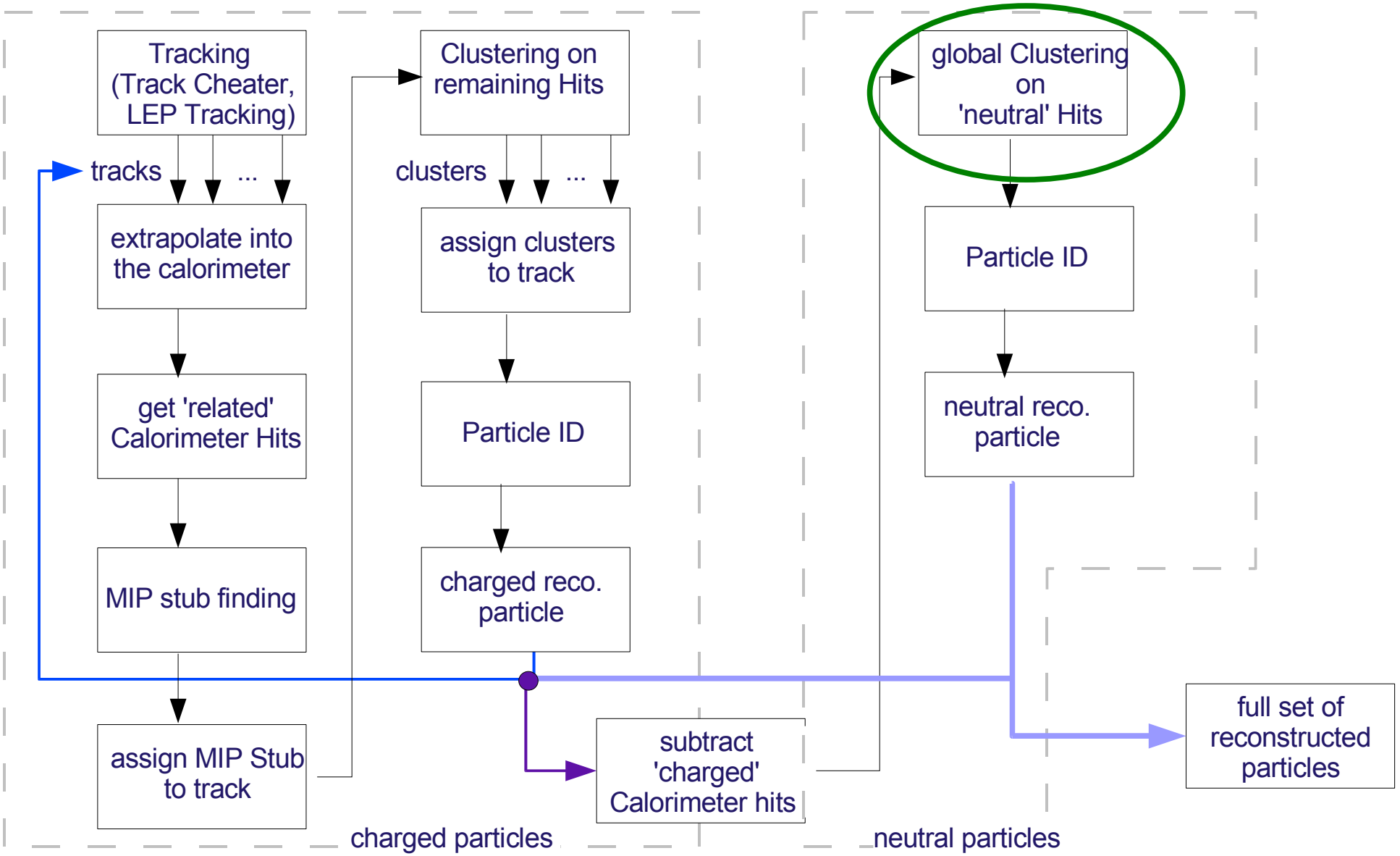
Remove 'Charged' Calorimeter Hits



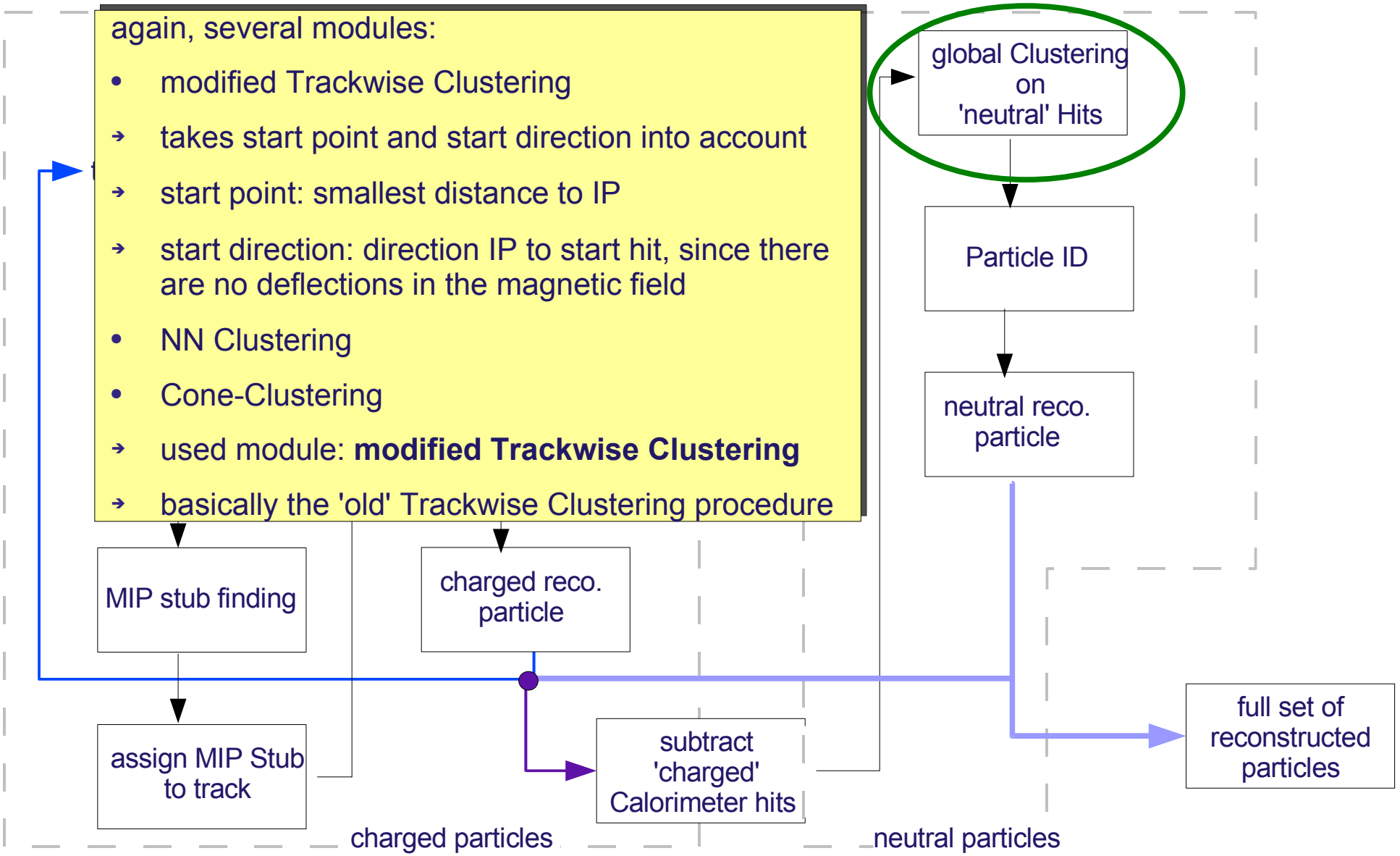
Remove 'Charged' Calorimeter Hits



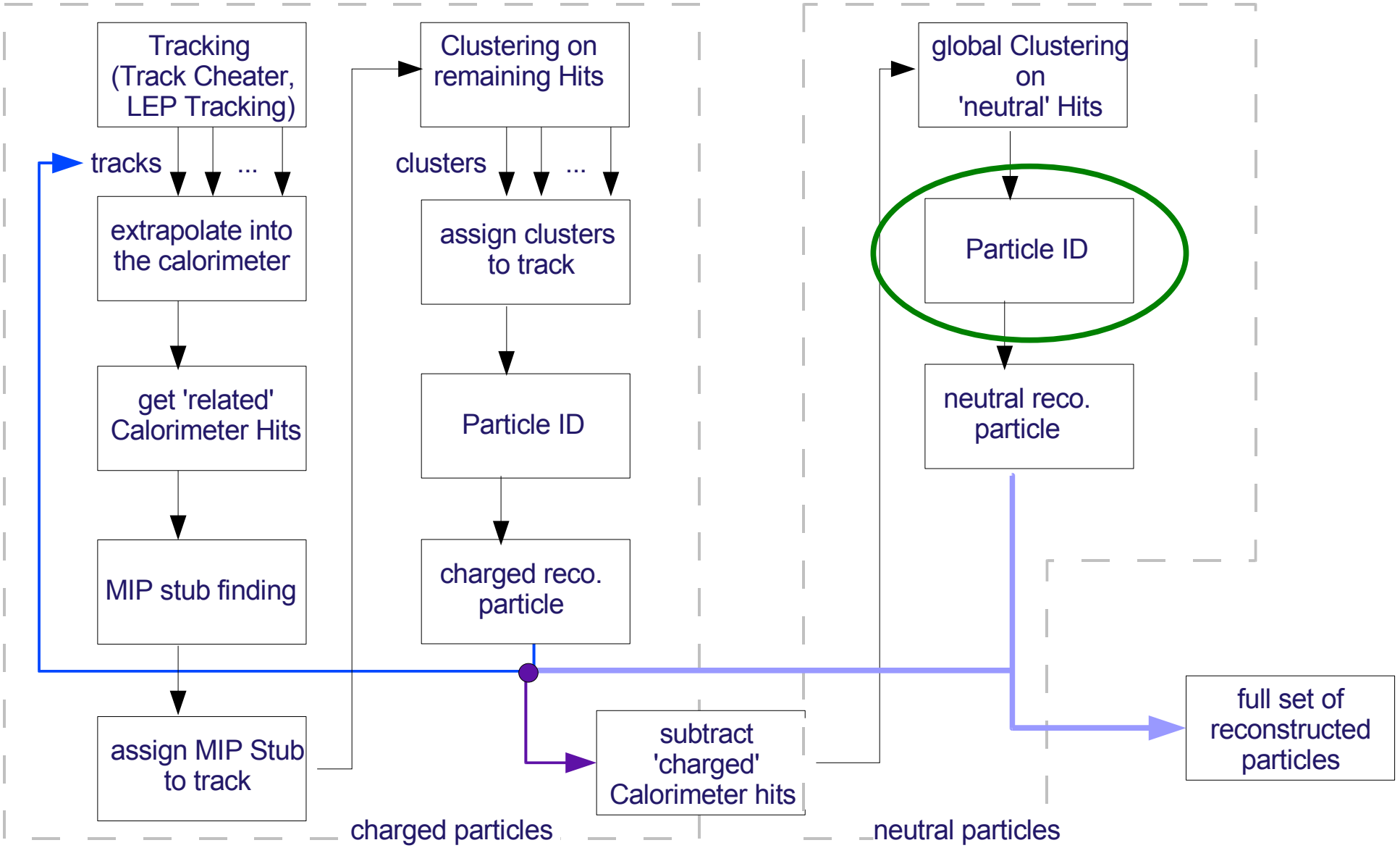
Clustering on Neutral Hits



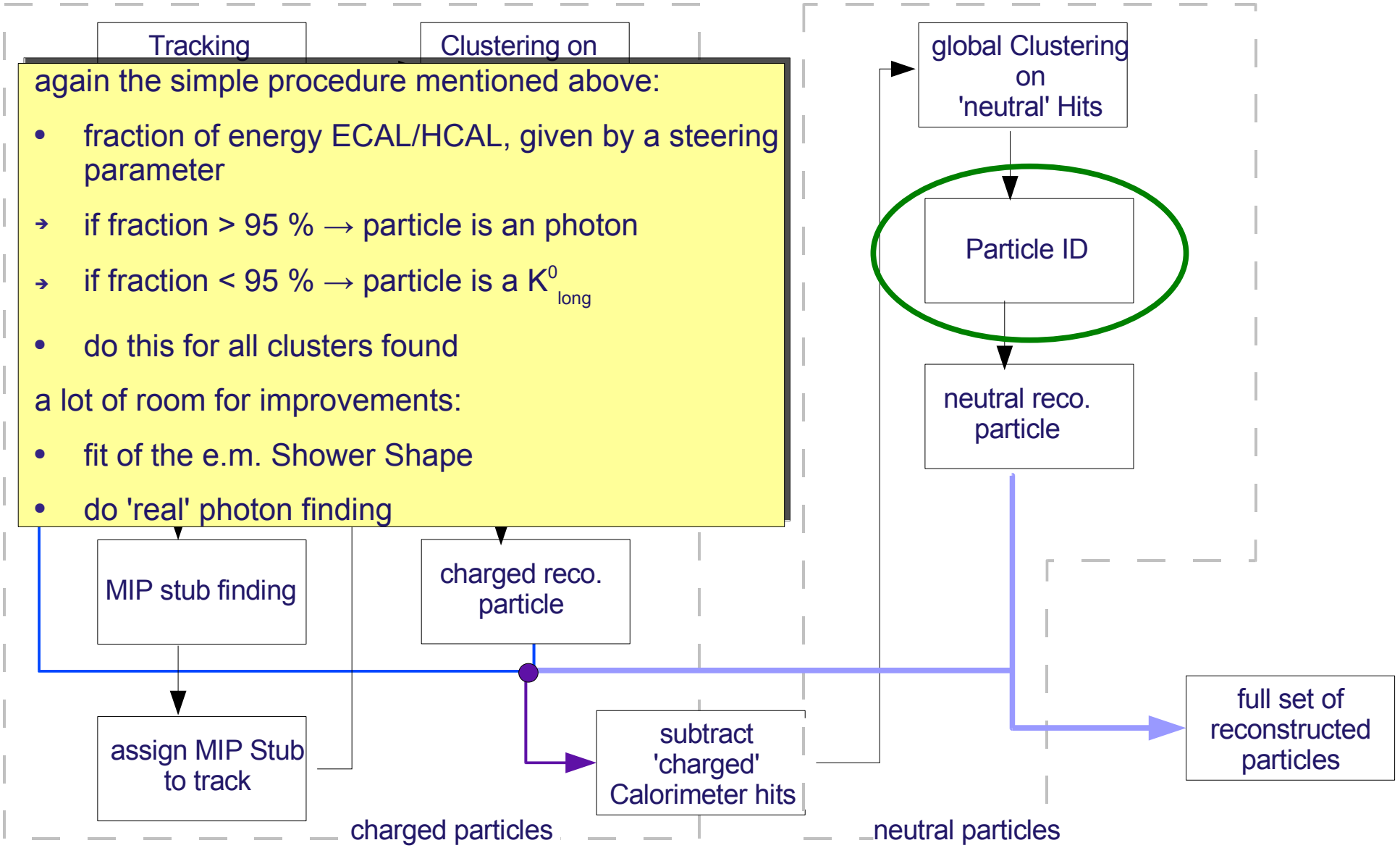
Clustering on Neutral Hits



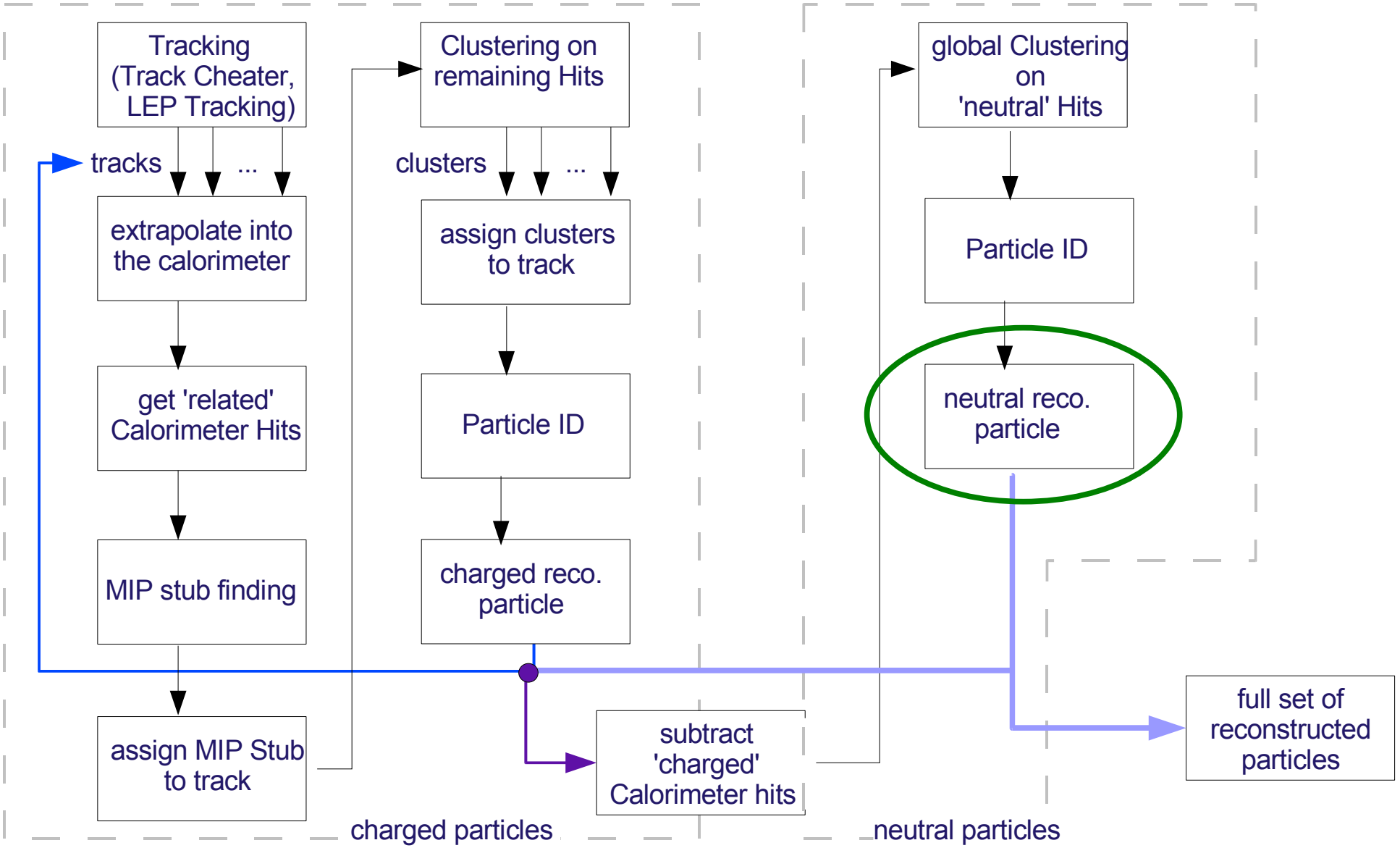
Particle ID for Neutrals



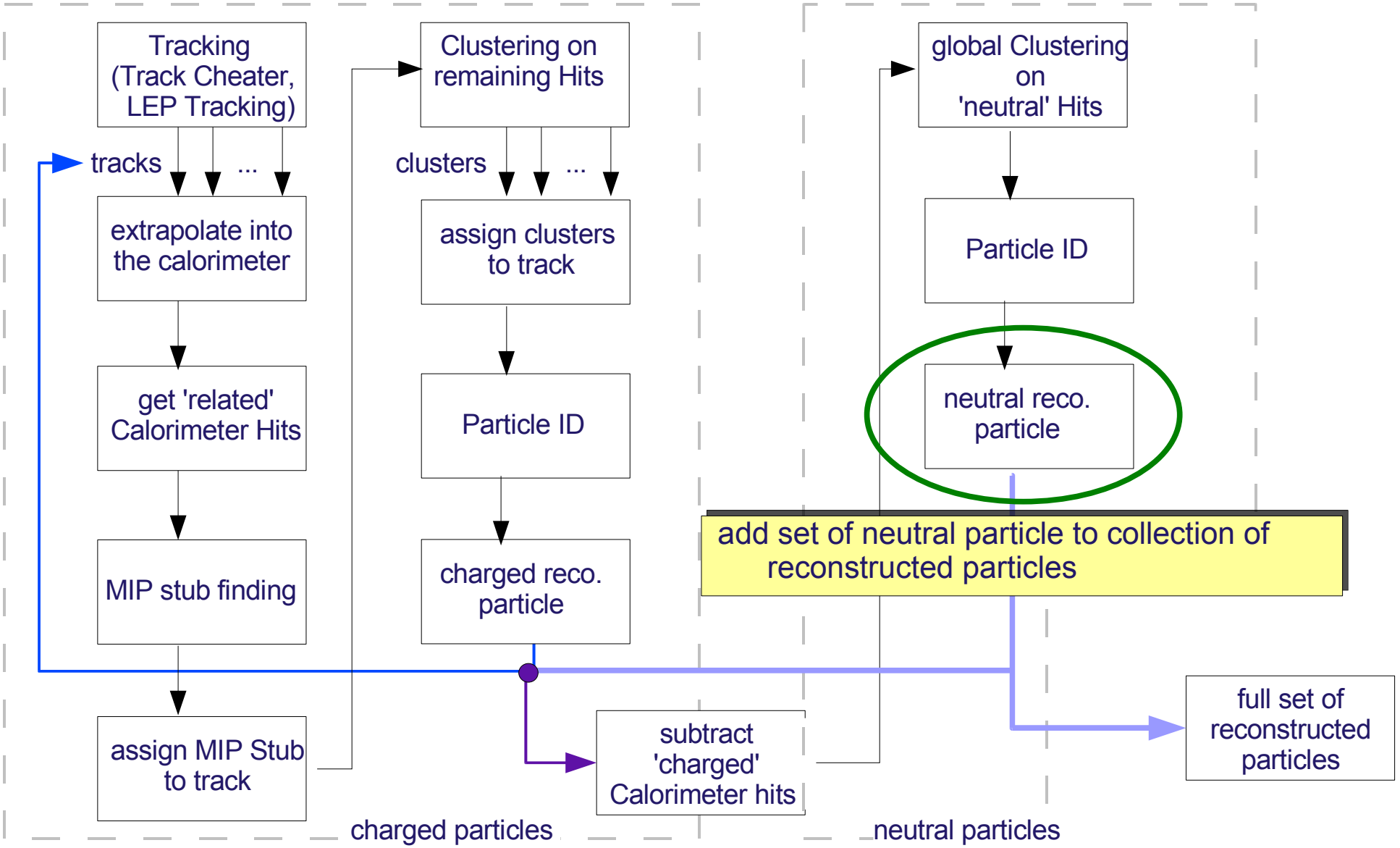
Particle ID for Neutrals



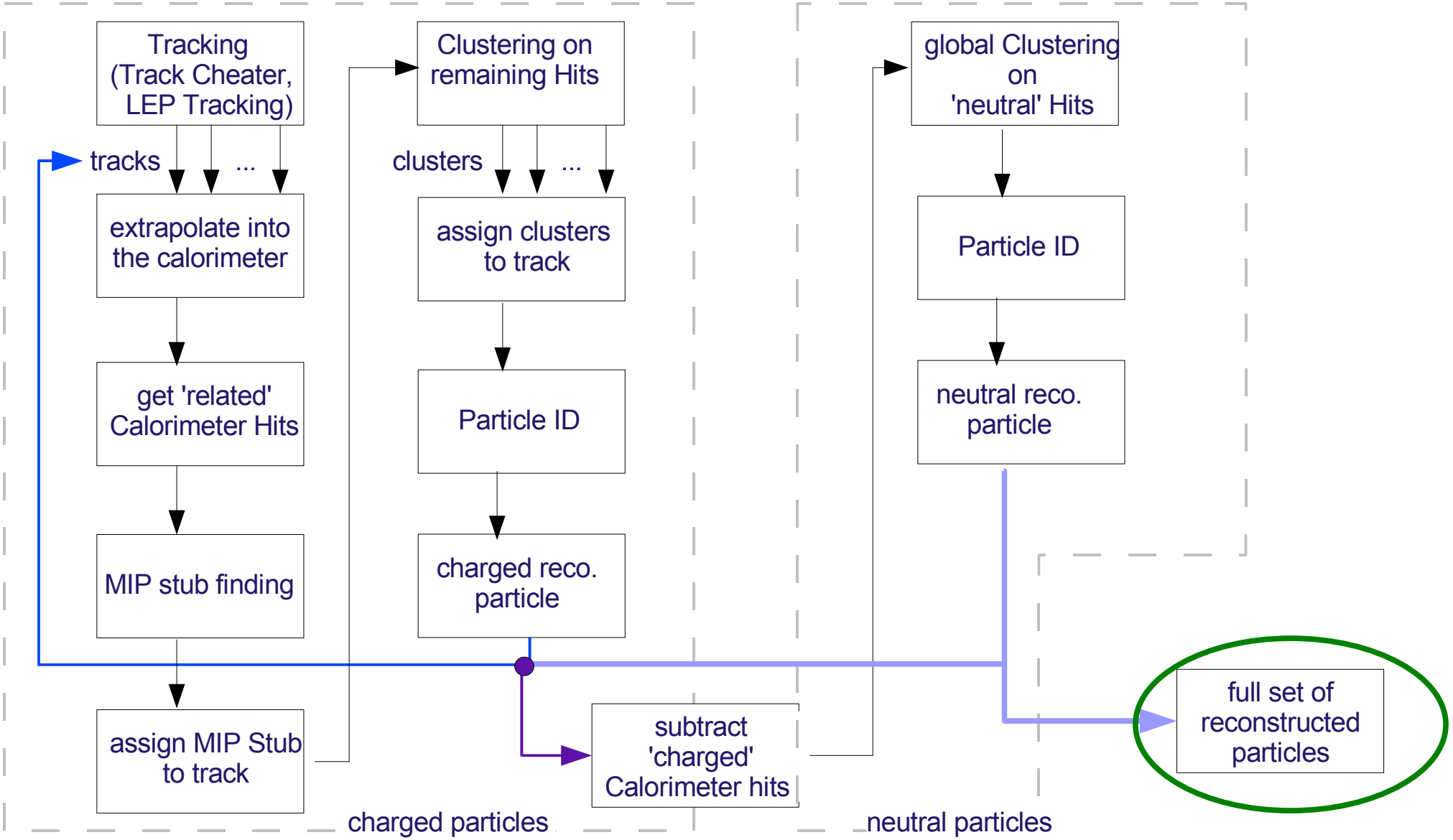
Neutral Reconstructed Particle



Neutral Reconstructed Particle



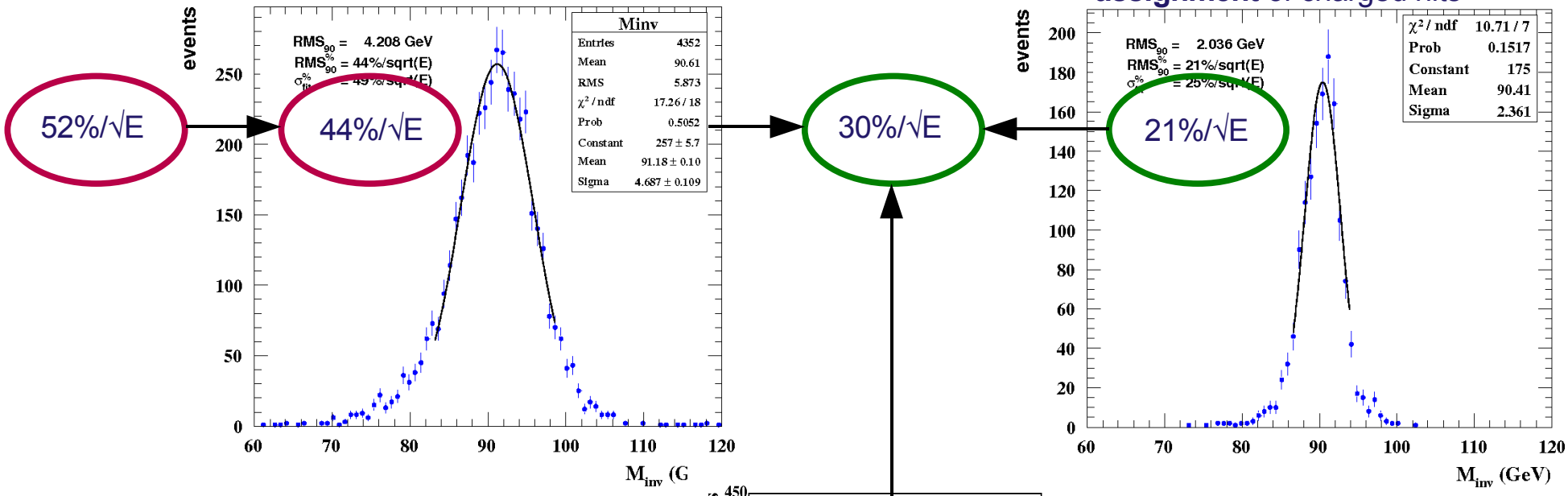
End of PFlow Procedure

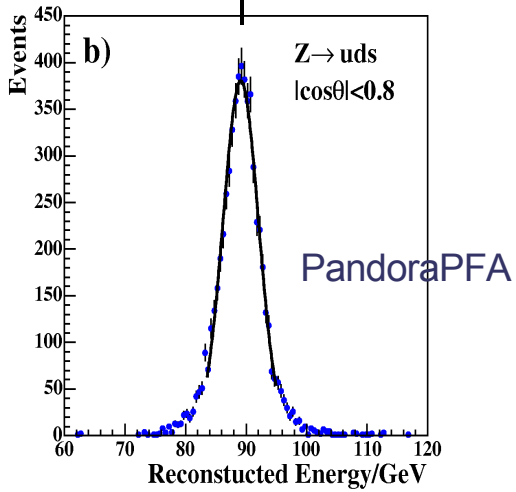


Performance of Track-Based PFlow

some first results for Z → uds, cos(θ) < 0.8, LDC00Sc, R(1690mm), L(2730mm):

Wolf realistic **Track-Based PFlow** **GOAL** **Track-Based PFlow** with **perfect assignment** of charged hits





PandoraPFA

- **exceeds performance** of Wolf but still (significantly) worse than PandoraPFA
- perfect assignment comes close 'theoretical' limit
- ➔ no major bugs in the code

Performance of Track-Based PFlow

same, but less statistics:

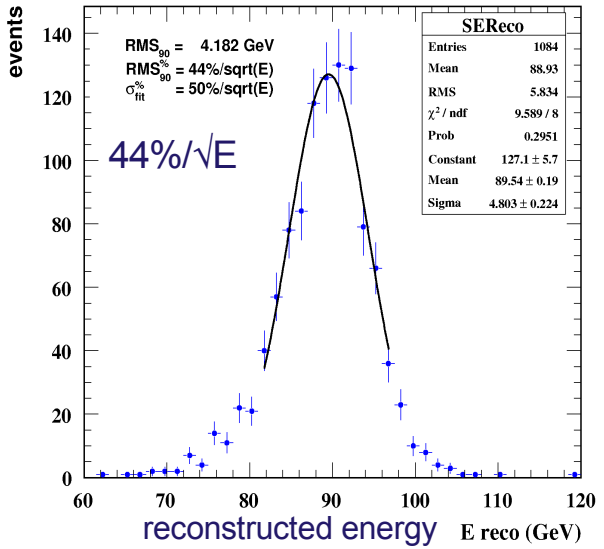
study the performance of this algorithm, define:

- E_{dc}: 'charged energy' counted as neutral
- E_{wa}: 'neutral energy' assigned to charged reconstructed particle
- always integrated numbers over the whole event
  - ➔ need numbers on jet- and single particle basis

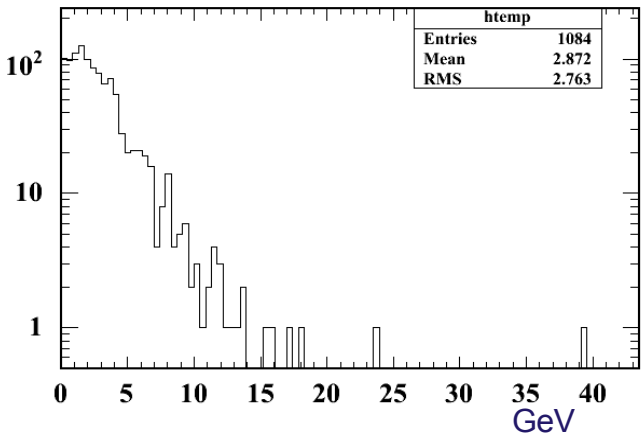
➔ more influence due to double counted energy

➔ optimise parameters of Track-Based PFlow

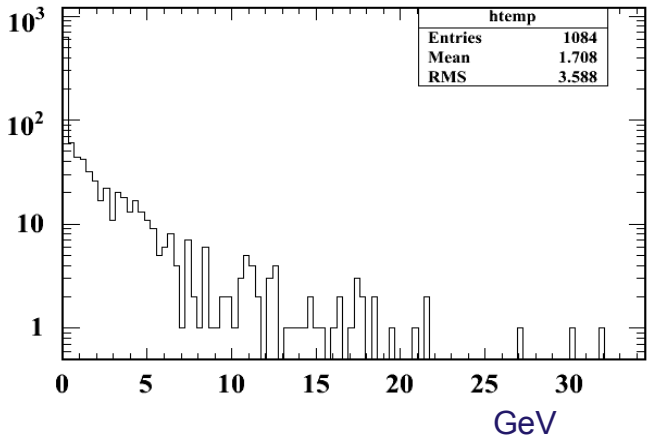
realistic Track-Based PFlow



double counted energy



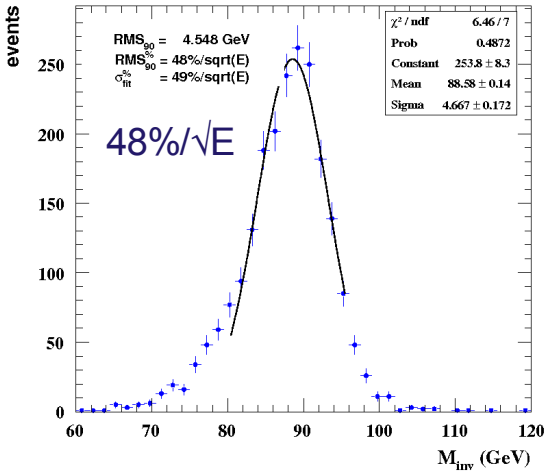
wrong assigned energy



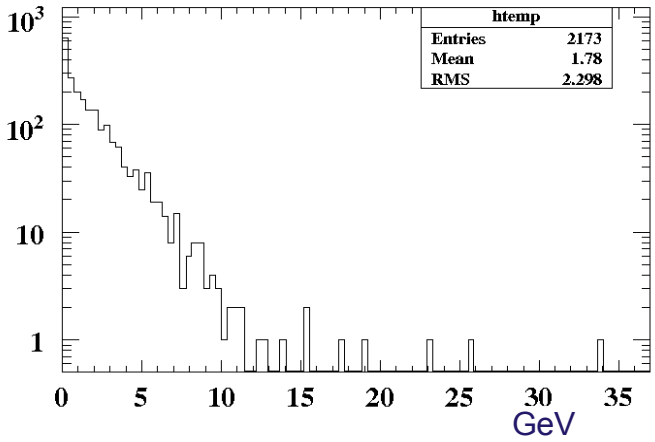
Performance of Track-Based PFlow

first attempt to optimise settings, again different statistics:

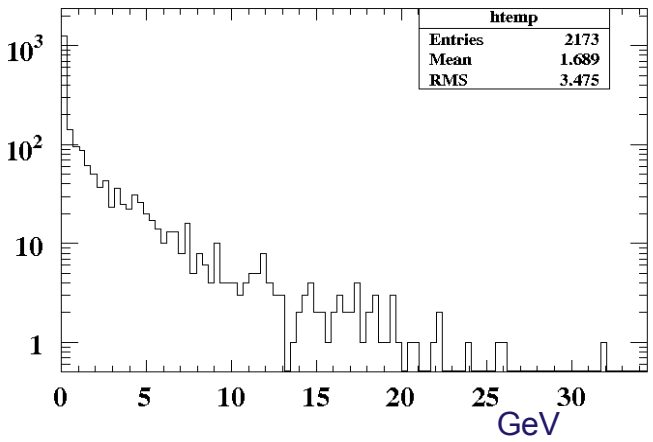
realistic Track-Based PFlow



double counted energy



wrong assigned energy



unfortunately, overall performance is worse

- much less double counting, but enhanced wrong assigned energy
- large left tail of the M_{inv} distribution due to (more) wrong assignment

➔ clustering procedure and (wrong) assignment of clusters to tracks are the main problems

➔ needs more and more detailed studies

Conclusions / Future Plans

- first version of a Track-Based Particle Flow **available** in Marlin (in MarlinReco cvs)
- improvements made since Valencia, performance exceeds Wolf, but is still worse than PandoraPFA → work in progress
- more and more understanding of 'intrinsic' problems / properties of Particle Flow algorithms
- **clustering** and **assignment** of energy (clusters) to tracks seem to be still the main problem
  - leads to wrong assigned and double counted energy
  - needs to be shown in detail (single particle level)
- clustering and cluster assignment are (more or less) stand-alone modules
  - easy to replace by different algorithms
- **clustering** (plans):
  - take amplitude information into account (e.g. RGB clustering of V. Morgunov)
  - study CALICE test beam calorimeter, use this as a prototype for clustering procedures
  - study sub-structure of hadronic showers (e.m. part, hadronic part)
  - study overlapping / neighboured showers
  - apply different procedures on test beam data and simulation as well as on the Particle Flow in the full detector simulation

Conclusions / Future Plans

- **cluster assignment** (plans):
  - assignment of clusters to tracks heavily depend on clustering procedure used
  - use more attributes of a cluster, such as direction (axes of inertia, ...), cluster shapes, ...

<u>additionally a lot more open issues:</u>

- track extrapolation, fitting tracker hits with trajectory model, take geometry and materials into account → Trajectory Class and related modules
- sophisticated V0 and kink finding
- Particle ID → analysis of cluster shapes, muon-, photon finding
  - → implement Photon finder of Predrag
- expand performance of 'Perfect Particle Flow', implement more tools to study performance
  - → Check Plot Processor
- compare different Particle Flow algorithms quantitatively
- compare different detector layouts and different detector models
- ...

→ optimise the detector design

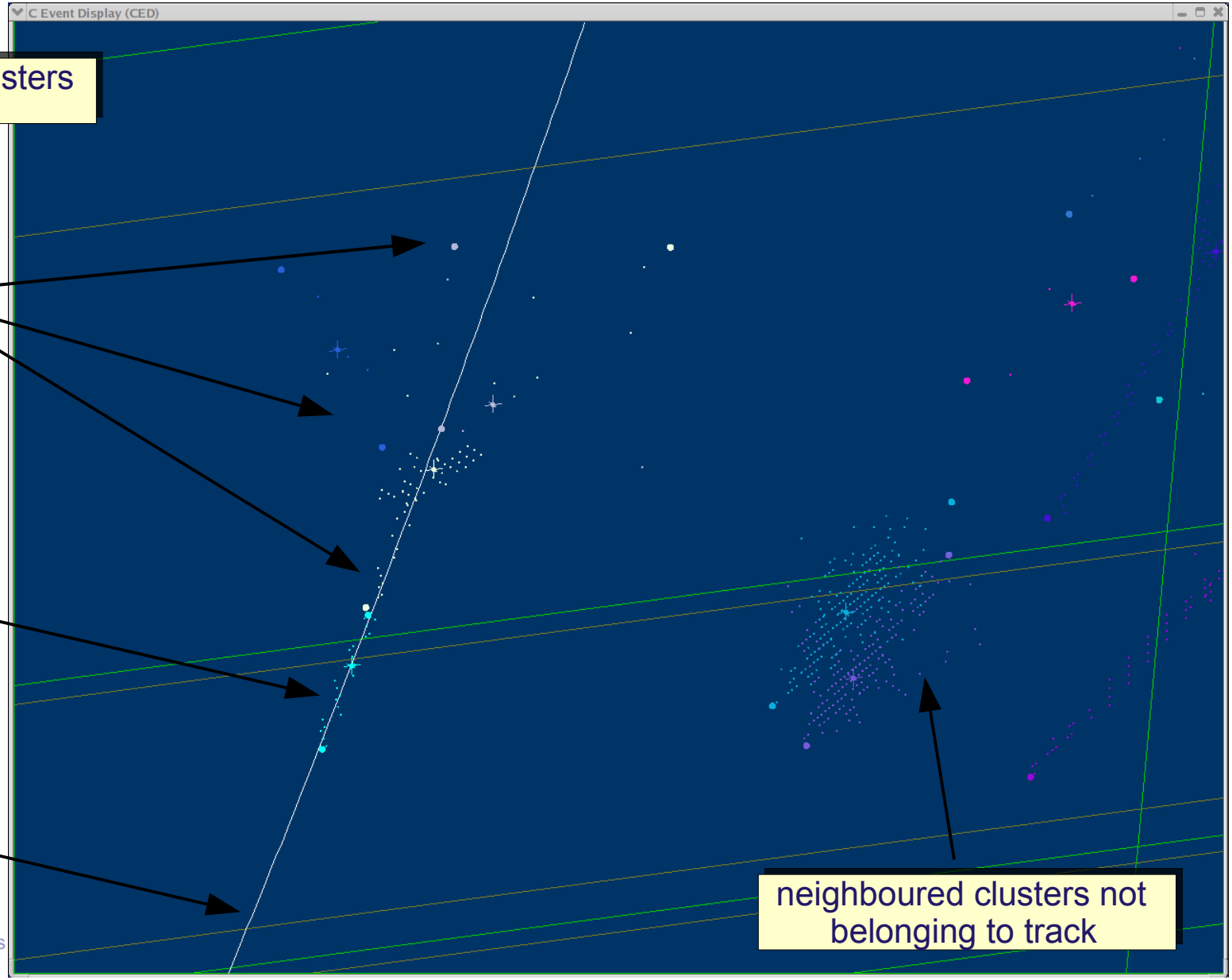
inputs / ideas / discussions are welcome

backup slides ...

Clustering

some examples:

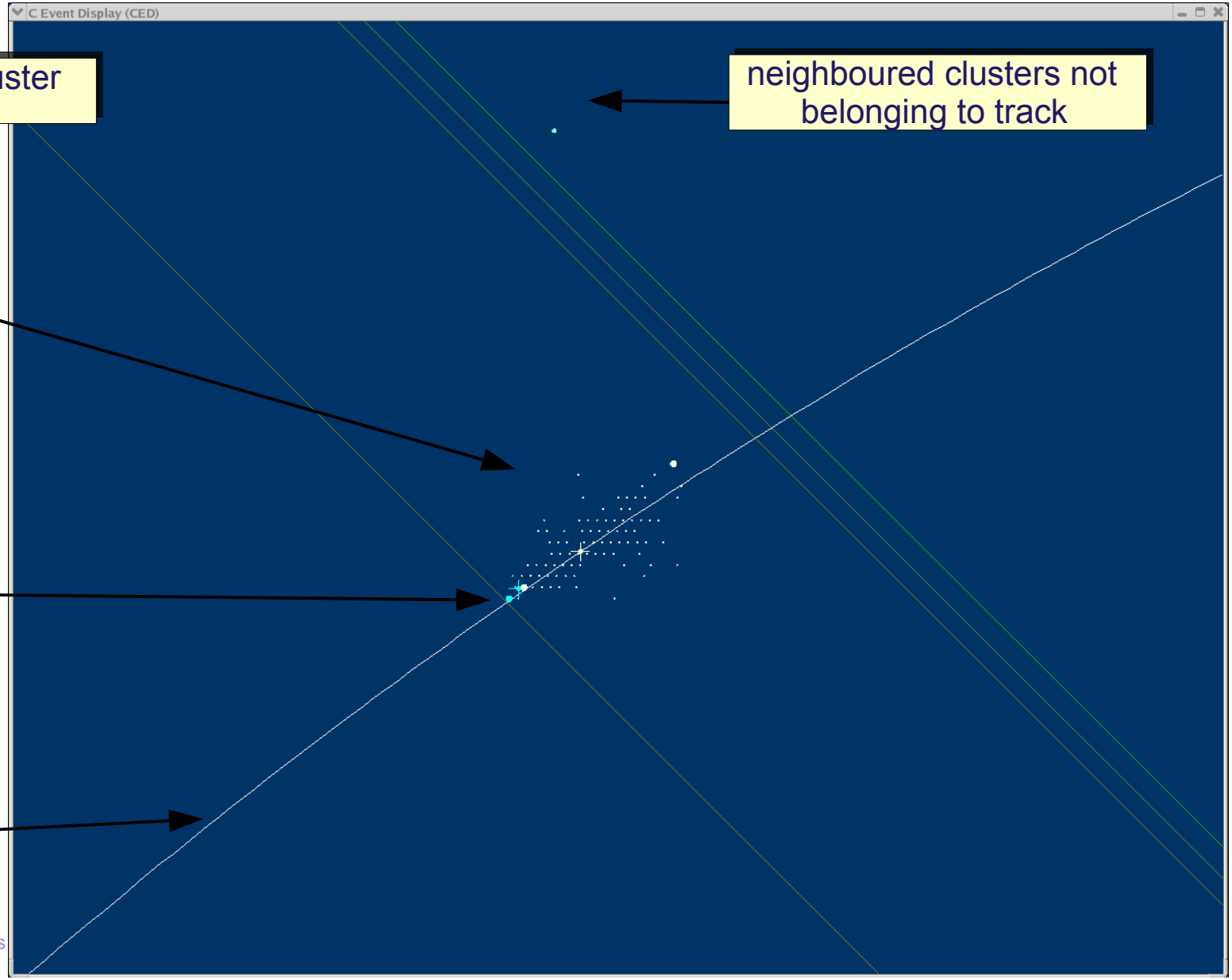
pion with several clusters



Clustering

some examples:

electron with one cluster

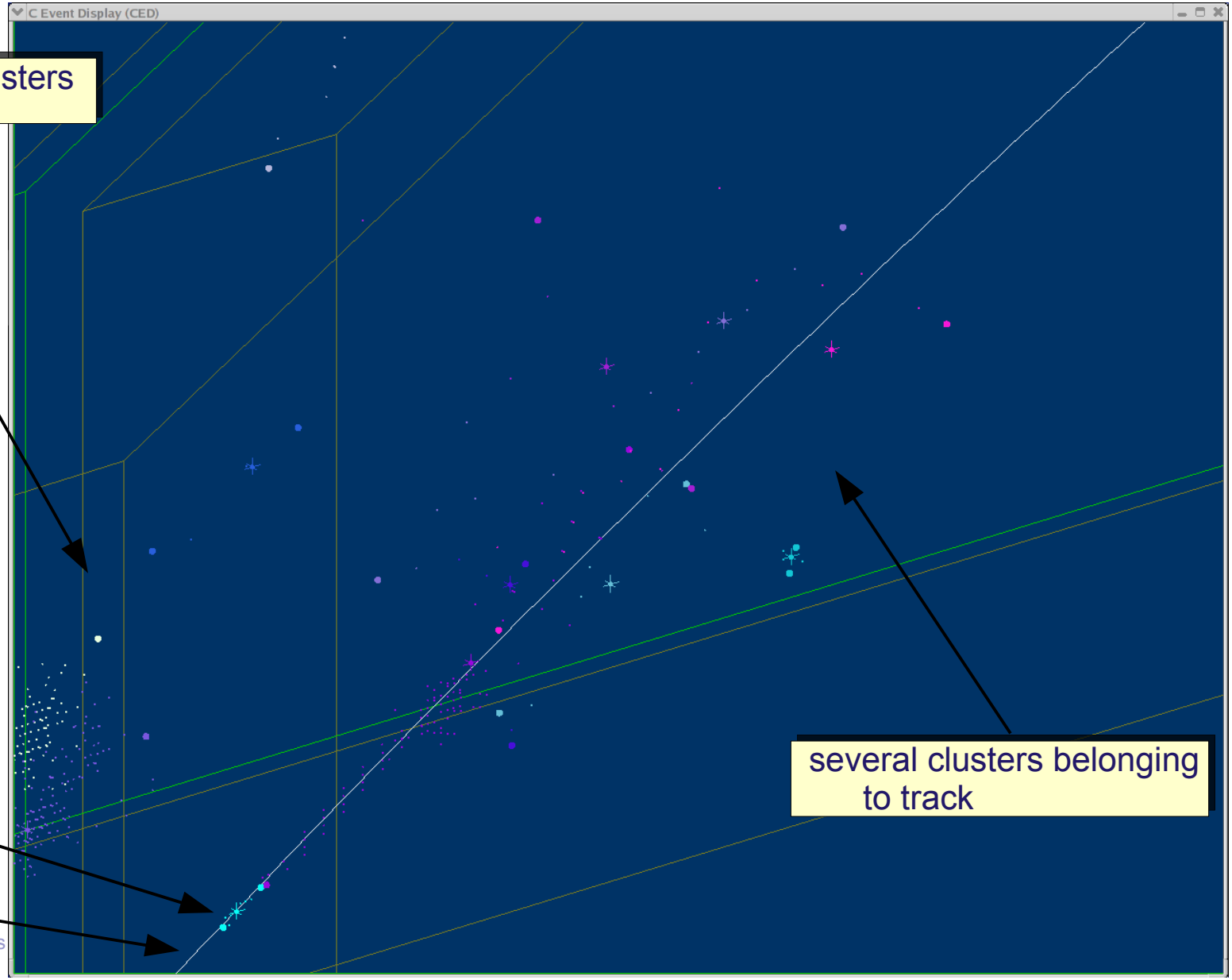


Assign Clusters to Track

some examples:

pion with several clusters

neighboured clusters not belonging to track



several clusters belonging to track

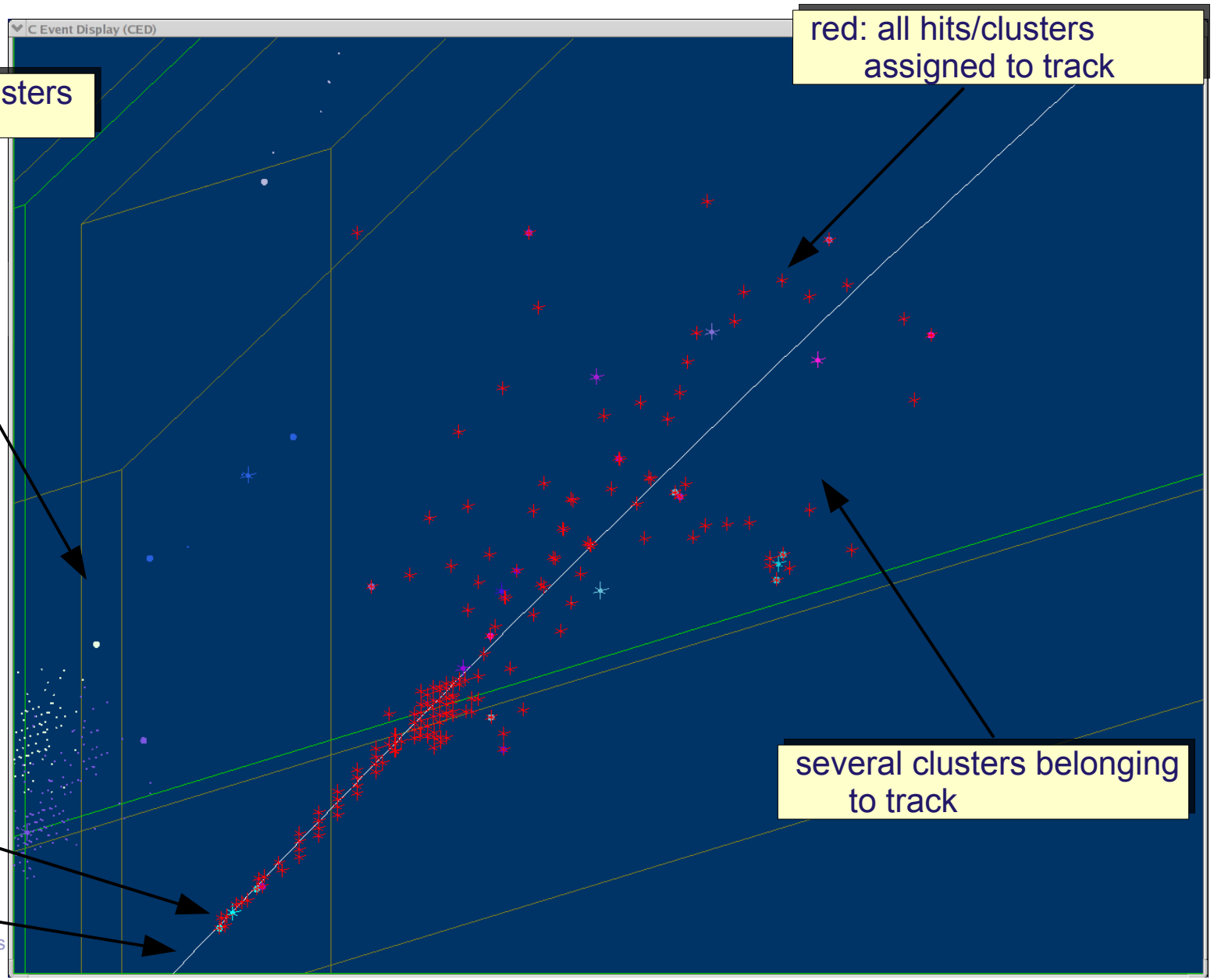
MIP Stub

extrapolated Track

Assign Clusters to Track

some examples:

pion with several clusters



red: all hits/clusters assigned to track

neighboured clusters not belonging to track

several clusters belonging to track

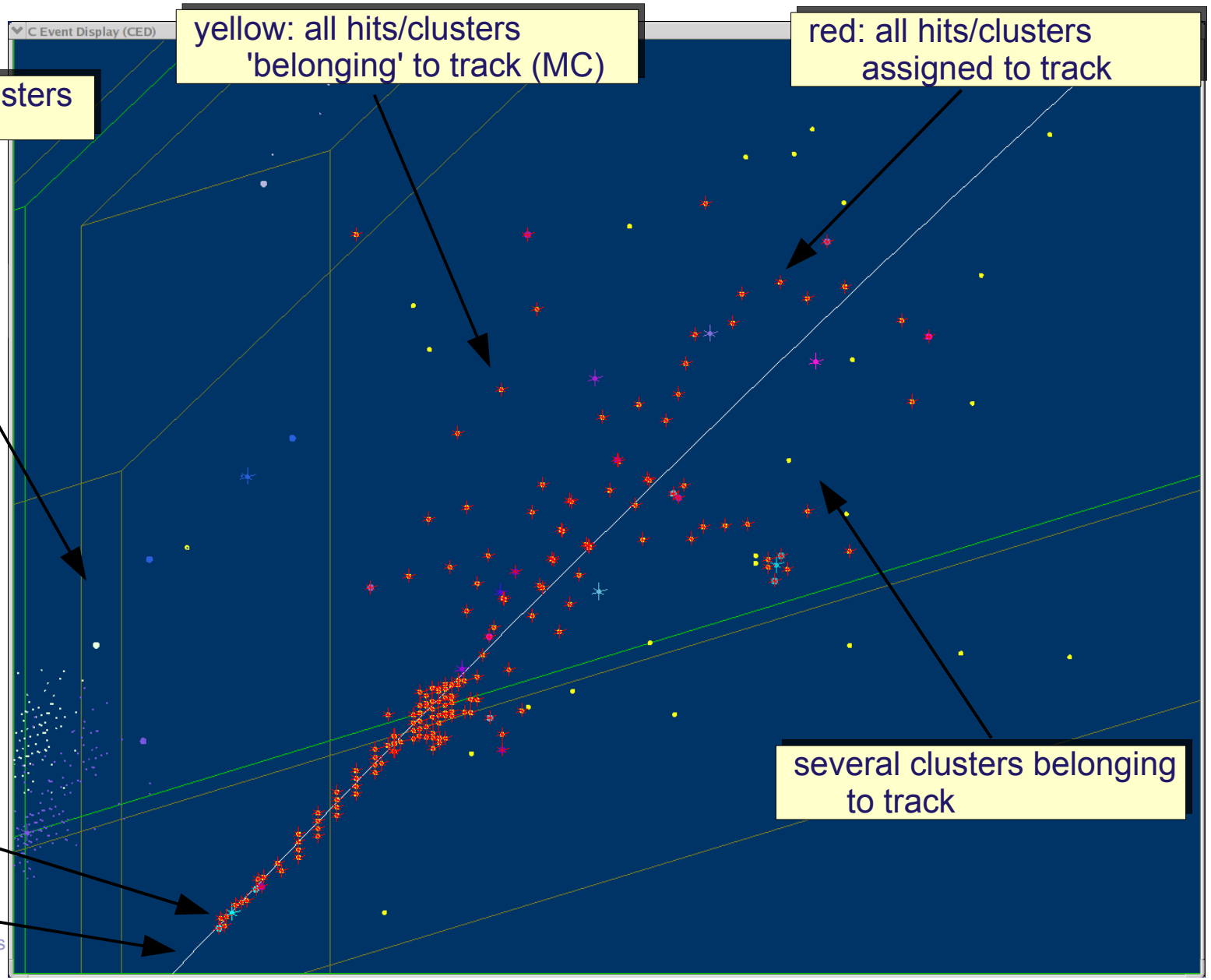
MIP Stub

extrapolated Track

Assign Clusters to Track

some examples:

pion with several clusters



Assign Clusters to Track

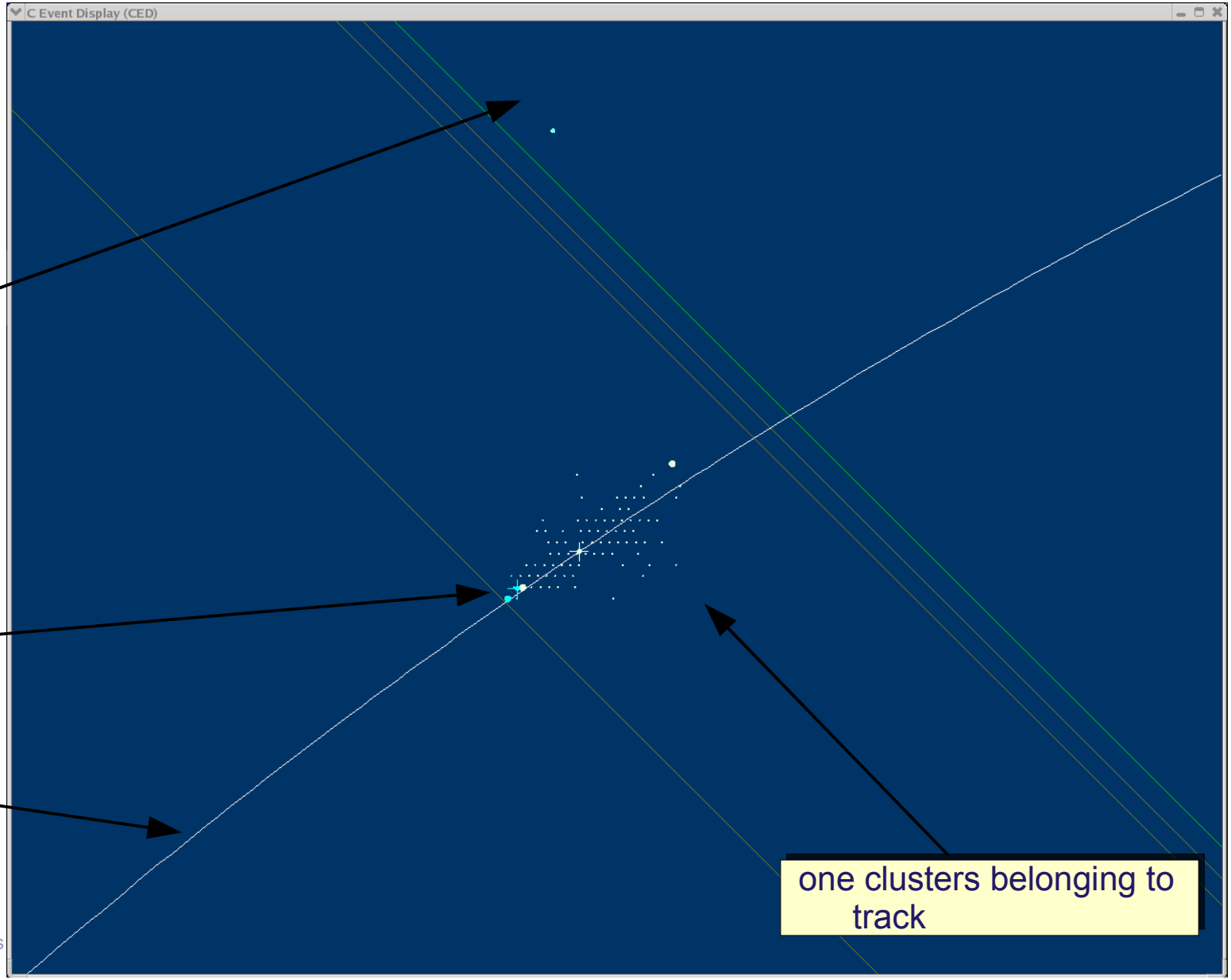
some examples:

electron with one cluster

neighboured clusters not belonging to track

MIP Stub

extrapolated Track

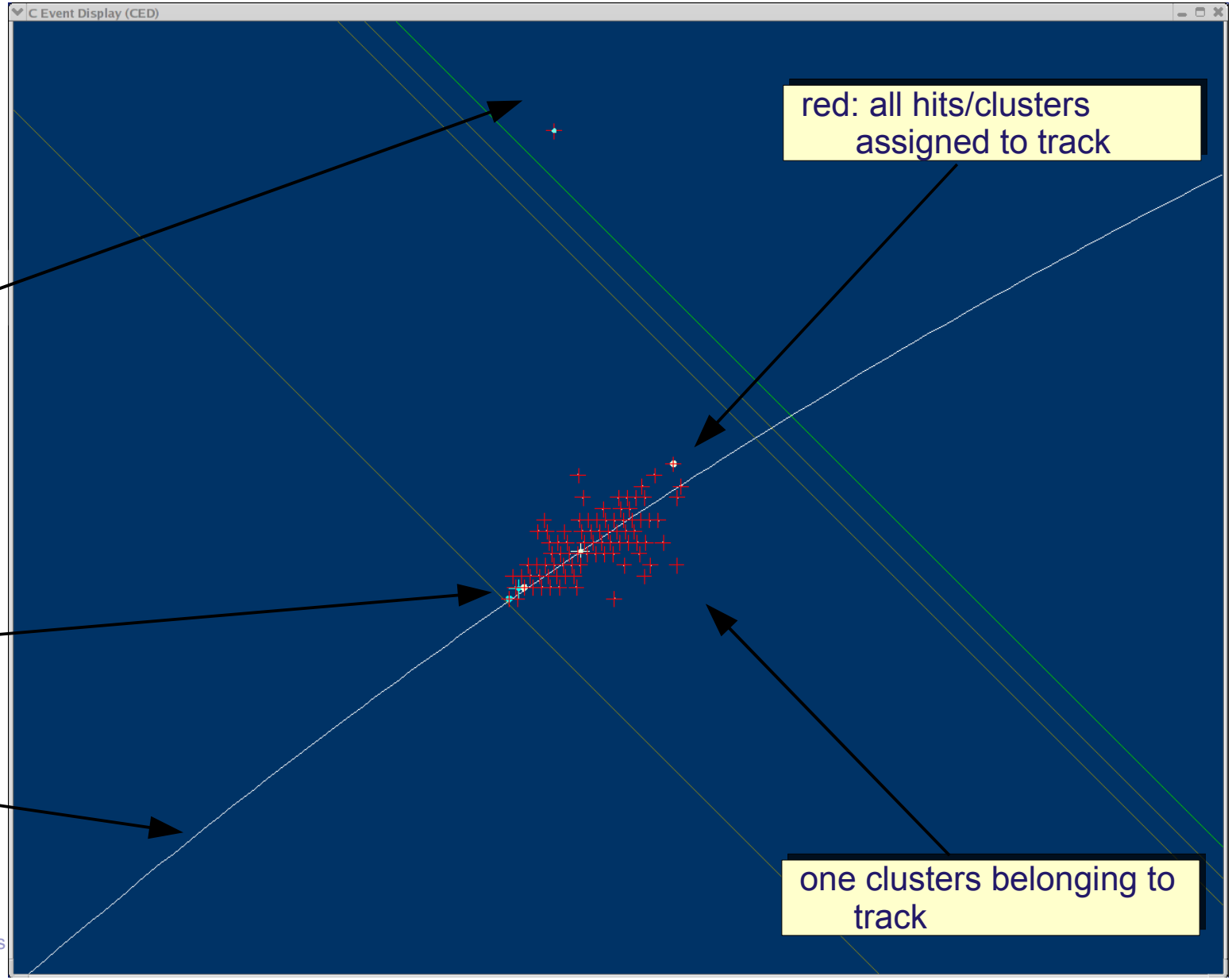


one clusters belonging to track

Assign Clusters to Track

some examples:

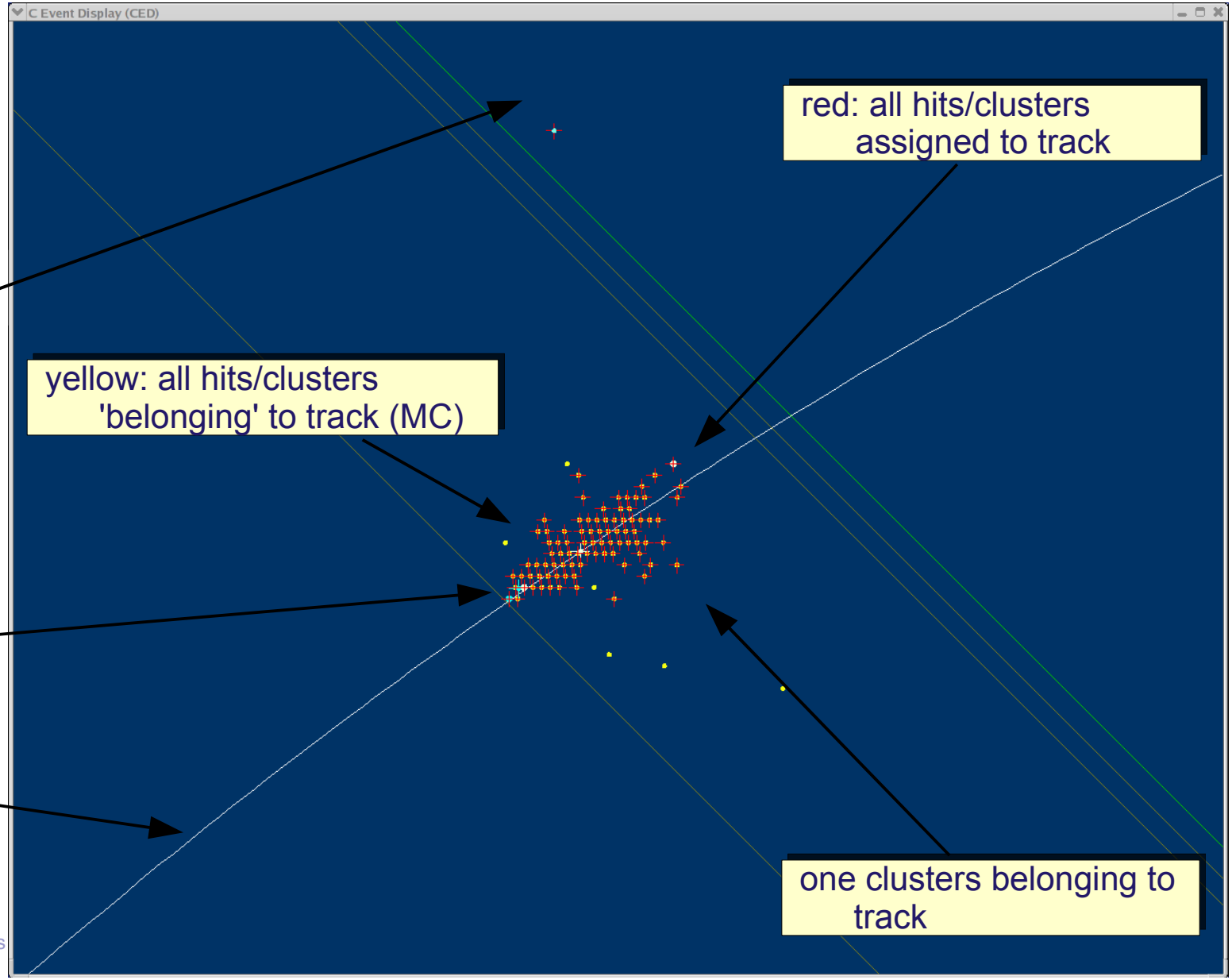
electron with one cluster



Assign Clusters to Track

some examples:

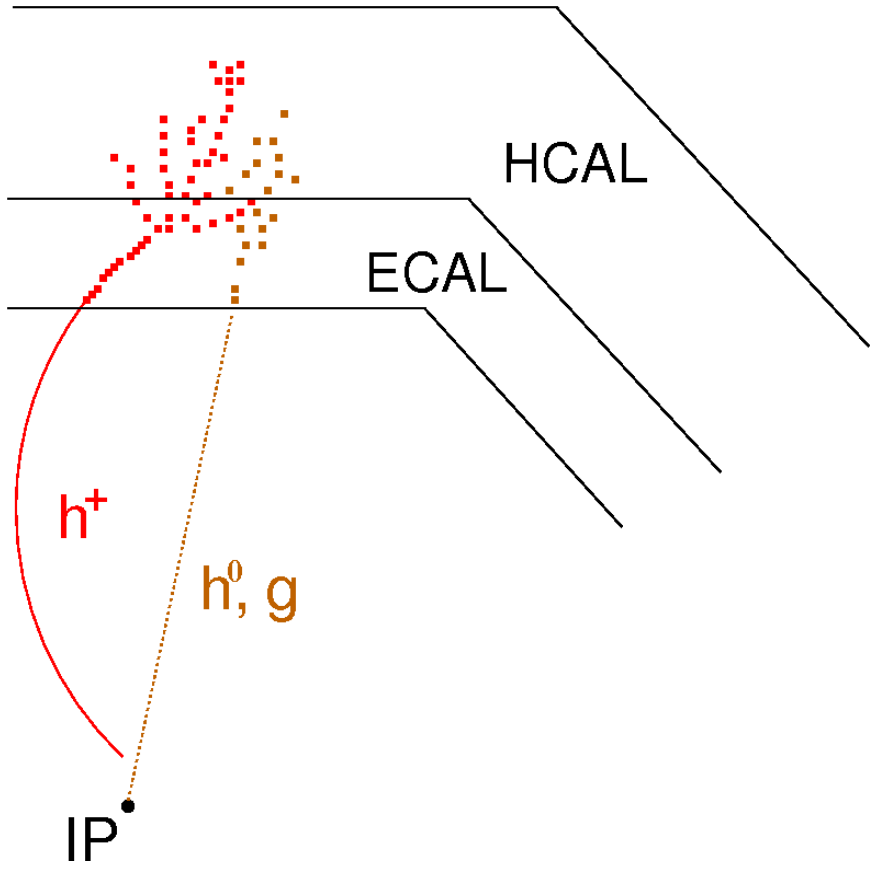
electron with one cluster



Performance of Track-Based PFlow

main reason: **overlaps** → **two** contributing **effects**

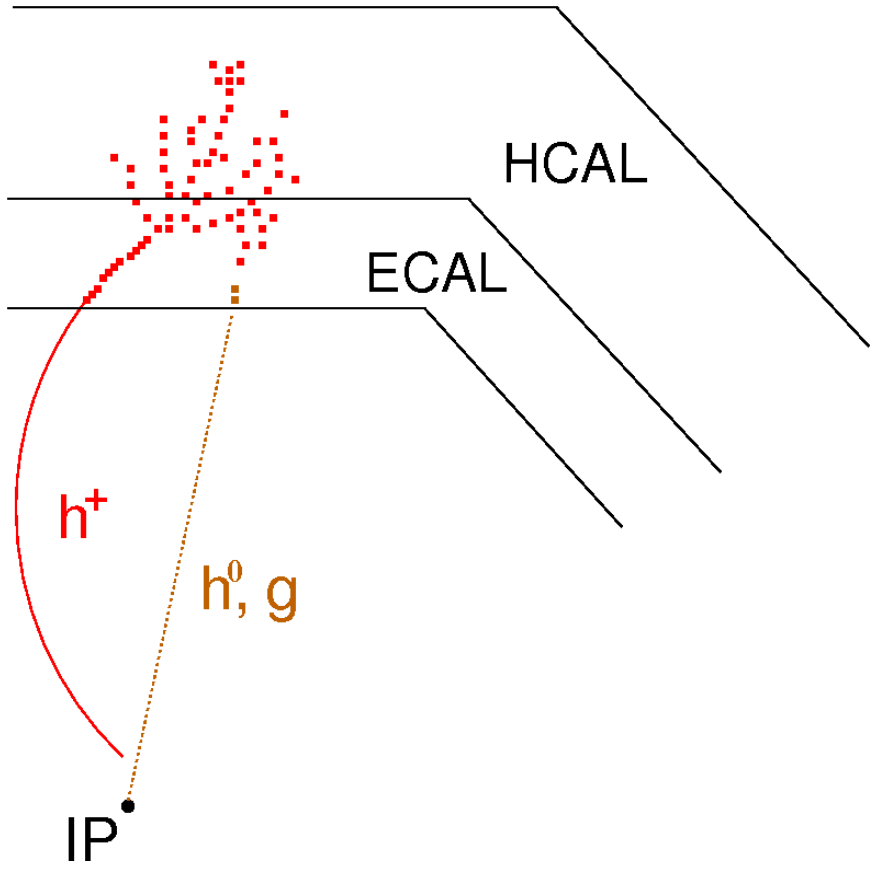
1. missing neutral energy (**wrong assignment**):



Performance of Track-Based PFlow

main reason: **overlaps** → **two** contributing **effects**

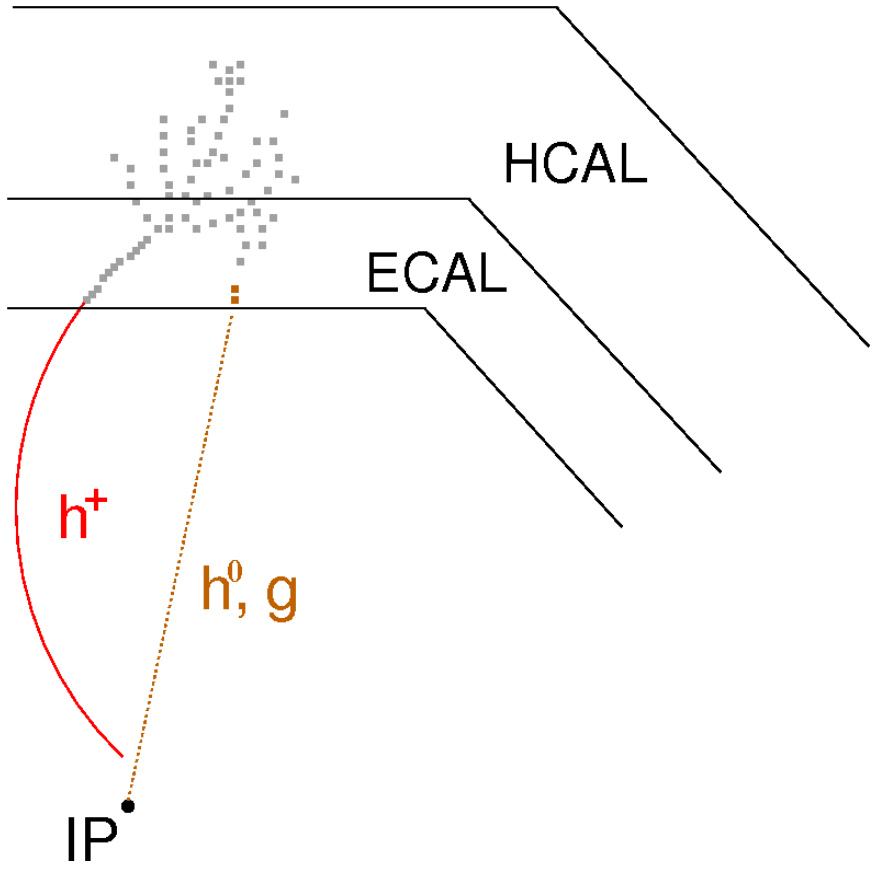
1. missing neutral energy (**wrong assignment**):
   - neighbouring energy depositions assigned to charged particle



Performance of Track-Based PFlow

main reason: **overlaps** → **two** contributing **effects**

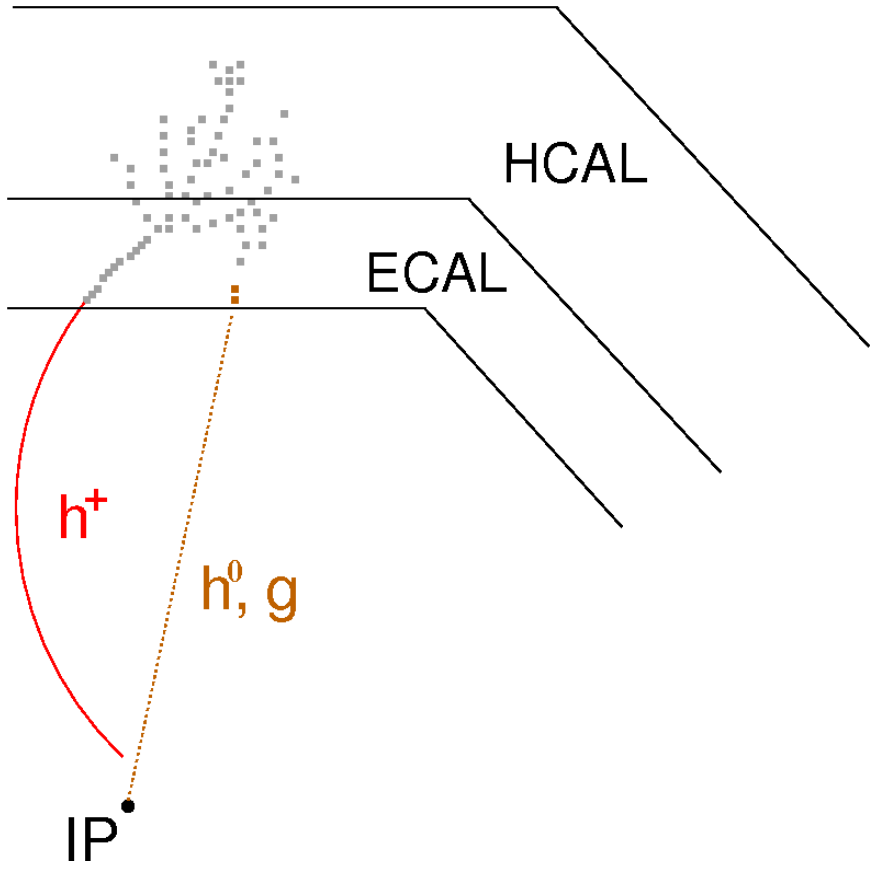
1. missing neutral energy (**wrong assignment**):
   - neighbouring energy depositions assigned to charged particle
   - energy of charged particle is calculated by $E^2 = p^2 + m^2$
   - ➔ additional energy causes 'no' effect
   - ➔ four momentum of charged particle **reconstructed accurately**



Performance of Track-Based PFlow

main reason: **overlaps → two** contributing **effects**

1. missing neutral energy (**wrong assignment**):
   - neighbouring energy depositions assigned to charged particle
   - energy of charged particle is calculated by $E^2 = p^2 + m^2$
   - → additional energy causes 'no' effect
   - → four momentum of charged particle **reconstructed accurately**
   - neutral particle reconstructed with **too small** energy or **not at all**
   - possibly problem to identify particle (h^0 or g → assign mass)

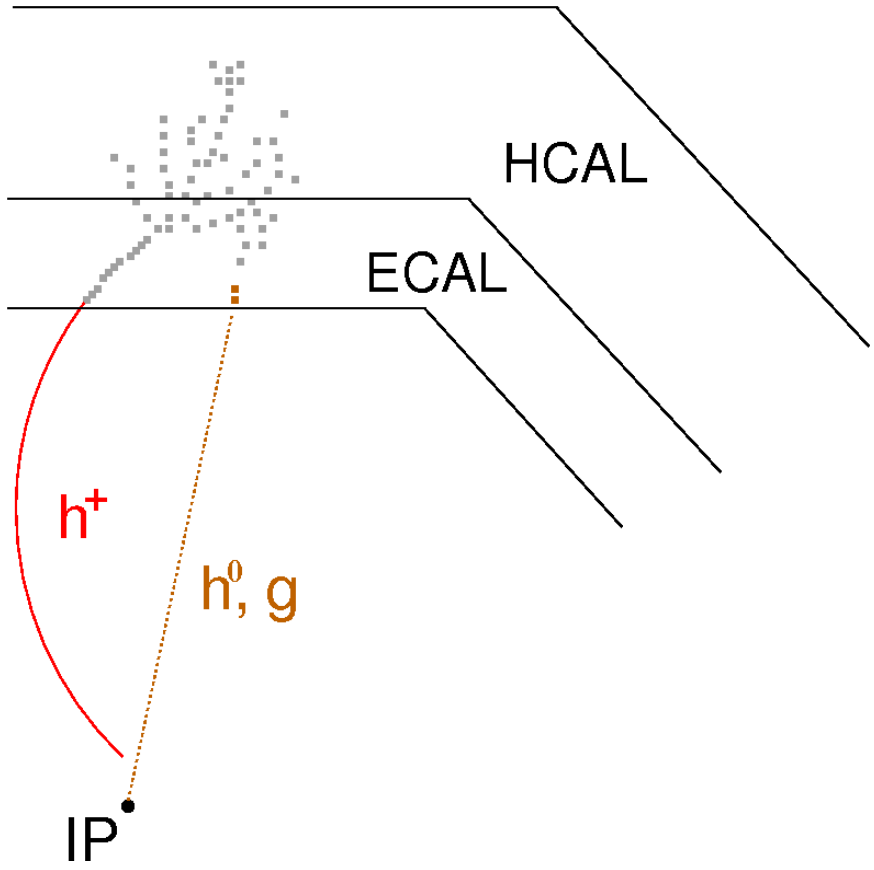


Performance of Track-Based PFlow

main reason: **overlaps → two** contributing **effects**

1. missing neutral energy (**wrong assignment**):
   - neighbouring energy depositions assigned to charged particle
   - energy of charged particle is calculated by $E^2 = p^2 + m^2$
   - → additional energy causes 'no' effect
   - → four momentum of charged particle **reconstructed accurately**
   - neutral particle reconstructed with **too small** energy or **not at all**
   - possibly problem to identify particle (h^0 or g → assign mass)

→ deteriorates (jet) energy to **smaller value**



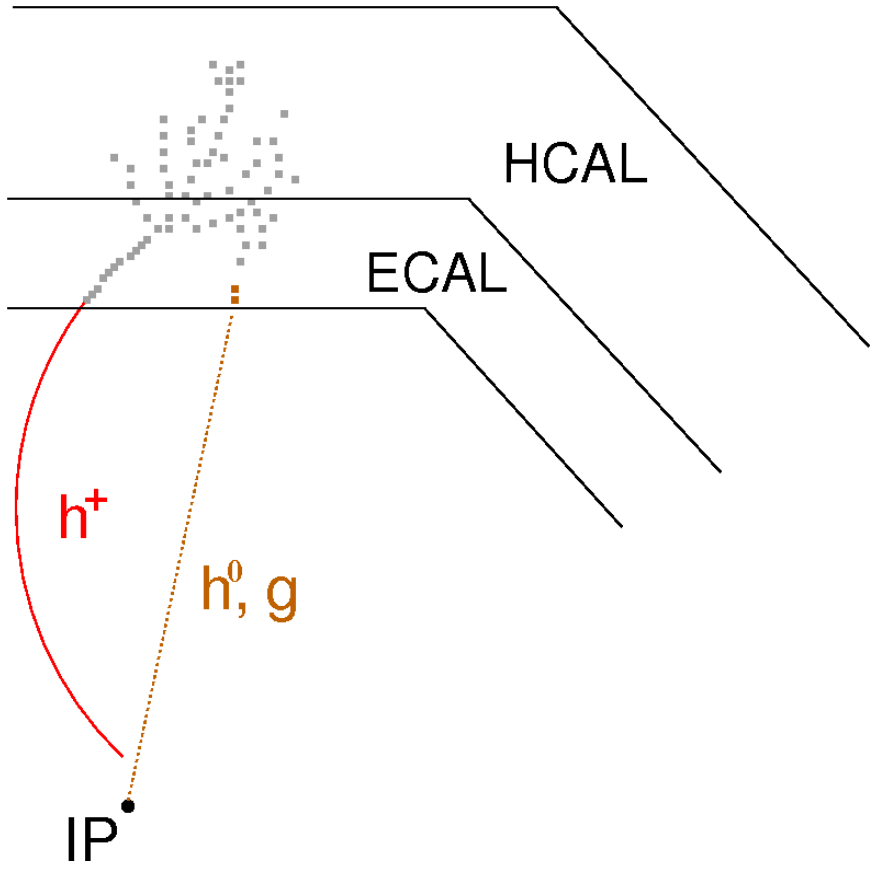
Performance of Track-Based PFlow

main reason: **overlaps** → **two** contributing **effects**

1. missing neutral energy (**wrong assignment**):
   - neighbouring energy depositions assigned to charged particle
   - energy of charged particle is calculated by $E^2 = p^2 + m^2$
   - → additional energy causes 'no' effect
   - → four momentum of charged particle **reconstructed accurately**
   - neutral particle reconstructed with **too small** energy or **not at all**

→ in this definition **only** the misassignment of neutral energy to a track ('charged energy') included

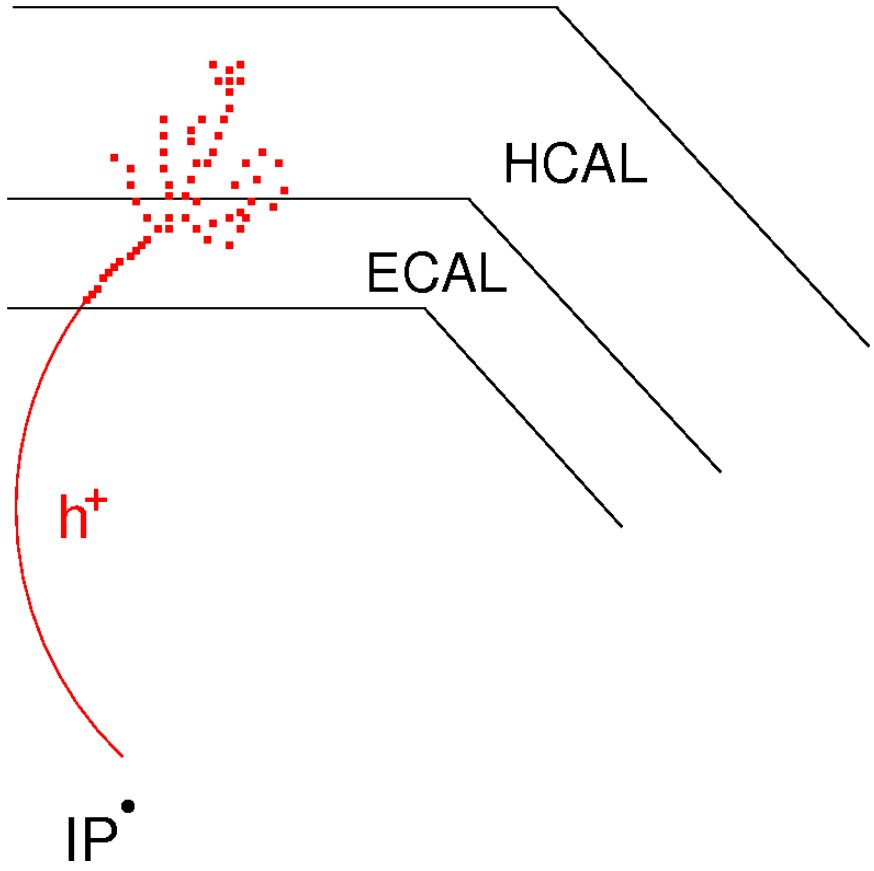
→ **always** negative



Performance of Track-Based PFlow

main reason: **overlaps** → **two** contributing **effects**

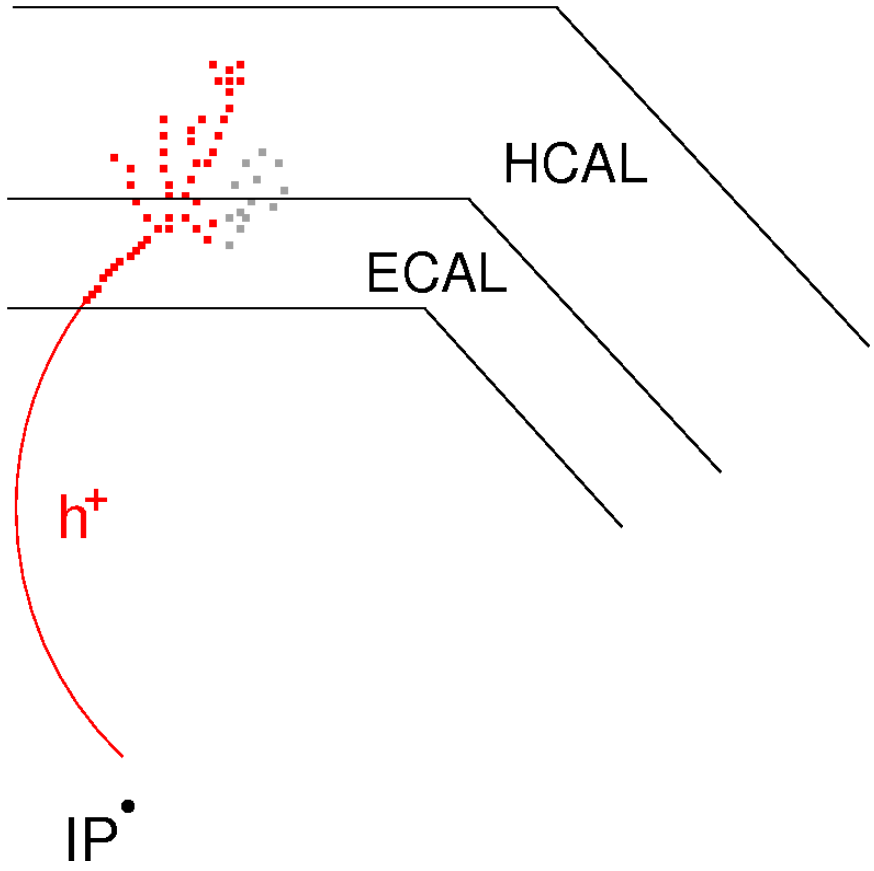
2. additional neutral energy (**double counting**):



Performance of Track-Based PFlow

main reason: **overlaps** → **two** contributing **effects**

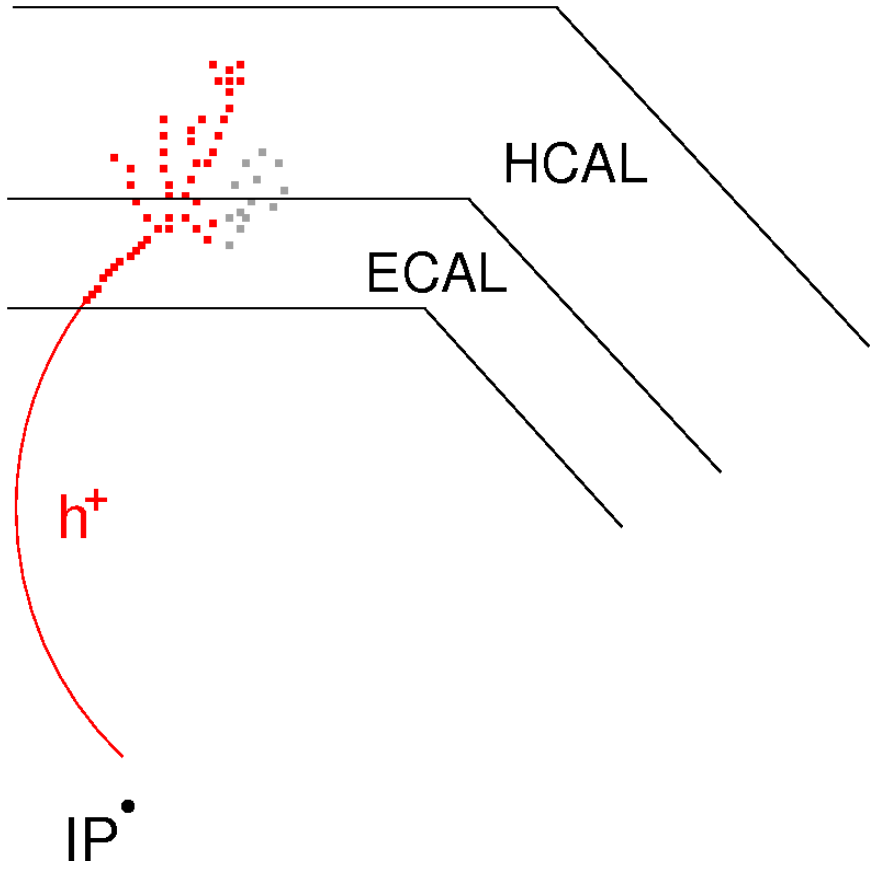
2. additional neutral energy (**double counting**):
   - energy depositions partly **not assigned** to charged particle



Performance of Track-Based PFlow

main reason: **overlaps** → **two** contributing **effects**

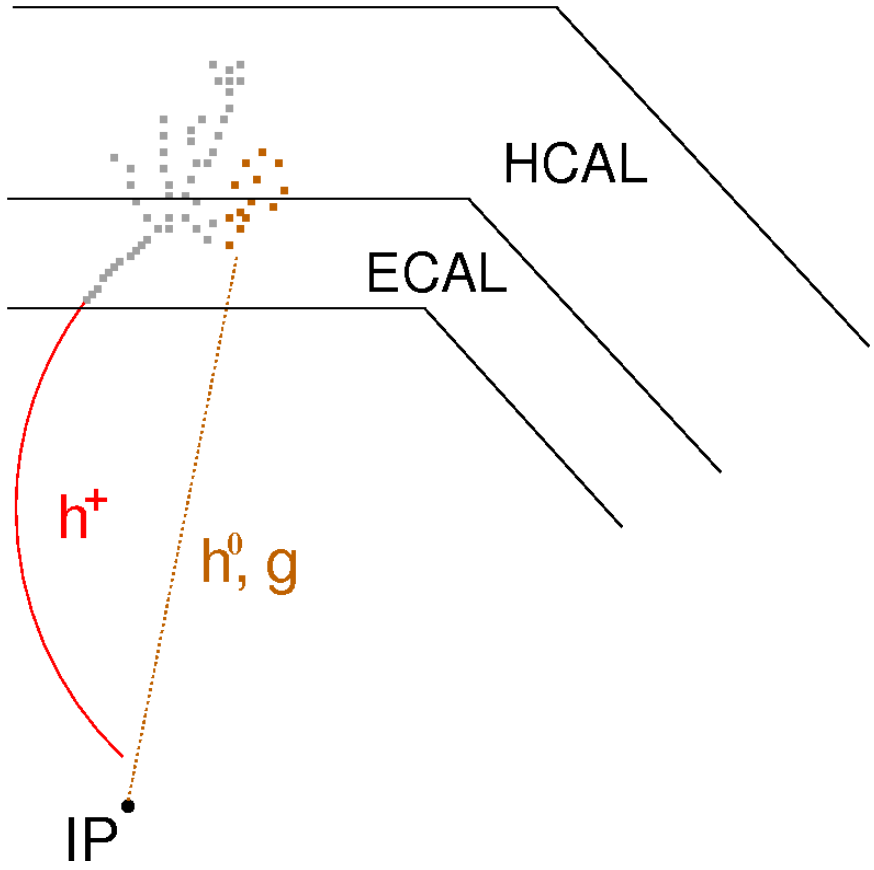
2. additional neutral energy (**double counting**):
   - energy depositions partly **not assigned** to charged particle
   - energy of charged particle is calculated by $E^2 = p^2 + m^2$
   - ➔ missing energy causes 'no' effect
   - ➔ four momentum of charged particle **reconstructed accurately**



Performance of Track-Based PFlow

main reason: **overlaps** → **two** contributing **effects**

2. additional neutral energy (**double counting**):
   - energy depositions partly **not assigned** to charged particle
   - energy of charged particle is calculated by $E^2 = p^2 + m^2$
   - → missing energy causes 'no' effect
   - → four momentum of charged particle **reconstructed accurately**
   - **additional** neutral particle reconstructed

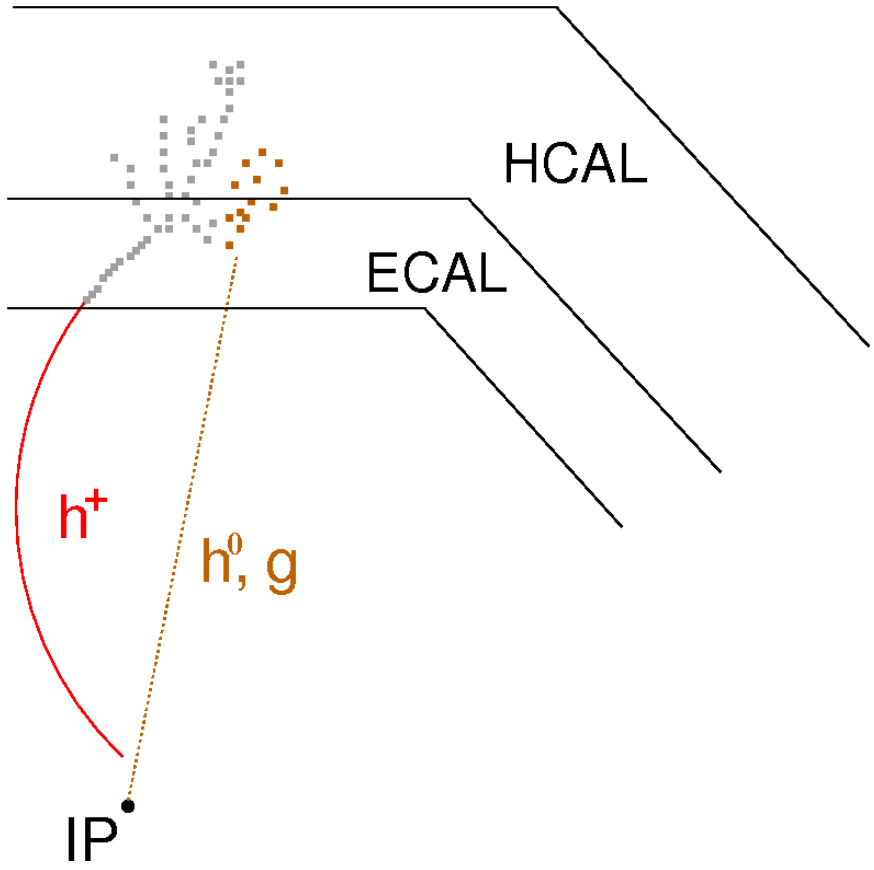


Performance of Track-Based PFlow

main reason: **overlaps** → **two** contributing **effects**

2. additional neutral energy (**double counting**):
   - energy depositions partly **not assigned** to charged particle
   - energy of charged particle is calculated by $E^2 = p^2 + m^2$
   - → missing energy causes 'no' effect
   - → four momentum of charged particle **reconstructed accurately**
   - **additional** neutral particle reconstructed

→ deteriorates (jet) energy to **larger value**



Performance of Track-Based PFlow

main reason: **overlaps → two** contributing **effects**

2. additional neutral energy (**double counting**):
   - energy depositions partly **not assigned** to charged particle
   - energy of charged particle is calculated by $E^2 = p^2 + m^2$
   - → missing energy causes 'no' effect
   - → four momentum of charged particle **reconstructed accurately**
   - **additional** neutral particle reconstructed

→ in this definition **only** the double counting of 'charged energy' is included

→ **always** positive