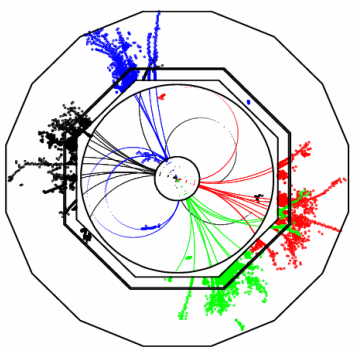


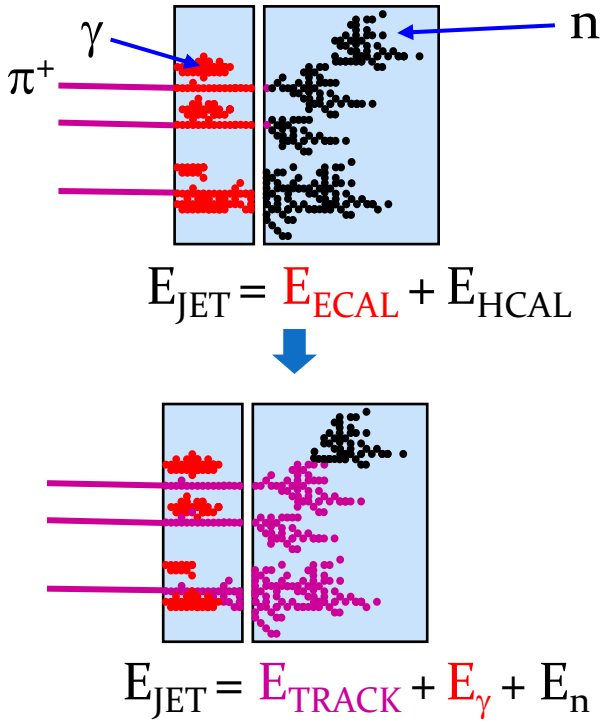
Redesign of Pandora PFA

John Marshall,
University of Cambridge

LCWS, Beijing, March 2010

Pandora PFA

- In a typical jet:
  - 60% of jet energy is in the form of charged hadrons
  - 30% is in photons (mainly from $\pi^0 \rightarrow \gamma\gamma$)
  - 10% is in neutral hadrons (mainly *n* and K_L)
- Particle flow calorimetry aims to improve jet energy resolution by:
  - Measuring charged particles in detector tracker (essentially perfectly)
  - Measuring photon energies in the ECAL $\sigma_E/E < 20\% / \sqrt{E(GeV)}$,
  - Only measuring neutral hadron energies in the HCAL, largely avoiding the intrinsically poor HCAL resolution.



E_{JET}	σ_E/E (rms_{90})
45 GeV	3.7 %
100 GeV	2.9 %
180 GeV	3.0 %
250 GeV	3.1 %

- The Pandora Particle Flow Algorithm:
  - Initially developed for the ILD detector concept.
  - The most mature PFA, giving the best performance.
  - Its algorithms are now well tested and understood.
  - Fully documented, NIMA 611 (2009) 25-40
  - Meets ILC jet energy goal of ~3.5 % at all relevant jet energies.

Pandora Redesign

- Whilst Pandora works well, current code has reached a point where it is extremely difficult to extend. It is not flexible enough to try out new ideas and improvements…
- ILD Letter of Intent version of Pandora has been frozen and a new version is being written from scratch.
- This is much more than just a re-implementation; Pandora is now a framework for running decoupled particle flow algorithms:
  - Increased flexibility, designed to make it easy to try out new ideas
  - Independent of any specific software framework and any specific detector details
  - Properly designed code, taking findings from previous PFAs into account, makes it easier to maintain
  - Easier for other people to get involved; simple for users to create and run their own algorithms
  - Pandora framework helps separate physics in particle flow algorithms from C++ memory management
- The new Pandora is a separate library, with no dependencies. A user application, in any framework, accesses the library via a simple C++ API (application programming interface).

Will now give an overview of new Pandora framework, summarise current status and present first results…

New Pandora Structure

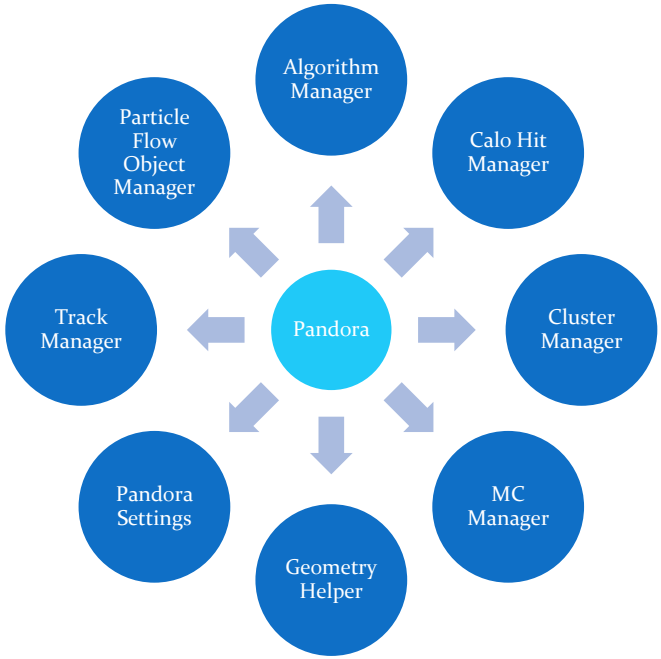
User Application:

- Specify Geometry
- Create Calo Hits
- Create Tracks
- Create MC Particles
- Register User Algorithms

Get Particle Flow Objects

Pandora API

Pandora Framework, can treat as "black box":



Pandora Content API

Pandora Algorithms:

- Clustering Algorithm
- Topological Associations Algorithm
- Statistical Reclustering Algorithm
- Photon ID Algorithm
- Fragment Removal Algorithm
- Track-cluster Association Algorithms
- PFO Construction Algorithm

Pandora API

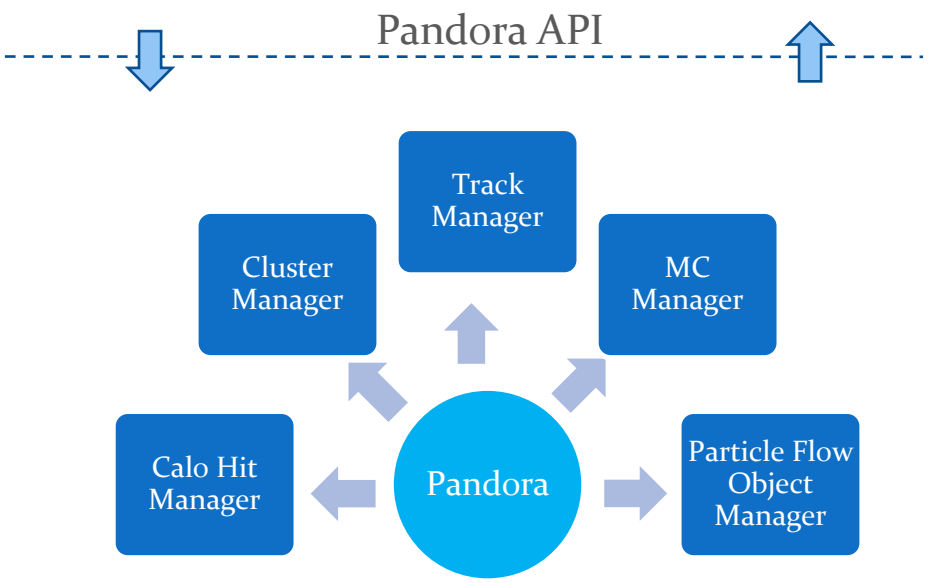
- To run Pandora, a user needs to write a small application in their chosen software framework.
- This application uses the PandoraAPI to supply Pandora with details of the detector geometry and of the calo hits and tracks in each event.
- Pandora then builds its own simple objects.

- Construction of these objects is simple; the user makes a Parameters class, fills the member variables and then calls the API Create function.
- Example member variables for a track:
  d0, z0, track state at start, track state at ECal, etc.
- All member variables must be specified, or an exception will be thrown when Create is called.

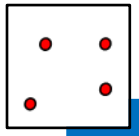
- The user can provide this information in any order, then call the API ProcessEvent function.
- Finally, user calls the API GetParticleFlowObjects function.

User Application, e.g. ILDPandora

```
PandoraApi::Track::Parameters parameters;
parameters.m_d0 = ...;
 ...
PandoraApi::Track::Create(pandora, parameters);
```

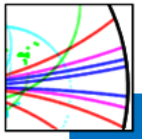


Pandora Objects



Calo Hit

- Position + normal vectors
- Calorimeter cell size
- Absorber material in front of cell
- Time of first energy deposition
- Calibrated energy (mip equivalent, EM, Had)
- Layer + pseudolayer
- Hit type + detector region
- Density weight
- Surrounding energy
- IsDigital, IsIsolated + IsPossibleMip flags
- Associated MC particle
- Associated user object

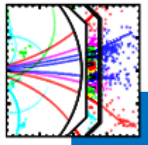


Track

- 2D impact parameters
- Momentum at d.c.a
- Particle mass
- Charge sign
- Start track state
- End track state
- ECal track state
- ReachesECal flag
- List of track state projections to calorimeter surfaces
- Associated cluster
- Associated MC particle
- Associated user object
- PFO formation flag
- "Clusterless" PFO formation flag

Cluster

- List of constituent calo hits, ordered by pseudolayer
- Mip fraction
- EM energy measure
- Had energy measure
- Initial direction
- Current direction
- Result of linear fit to all hits in cluster
- Energy-weighted centroid
- ShowerStart layer
- Shower profile properties
- List of associated tracks



Particle Flow Object

- PDG Code
- Charge
- Mass
- Energy
- Momentum
- List of tracks
- List of clusters

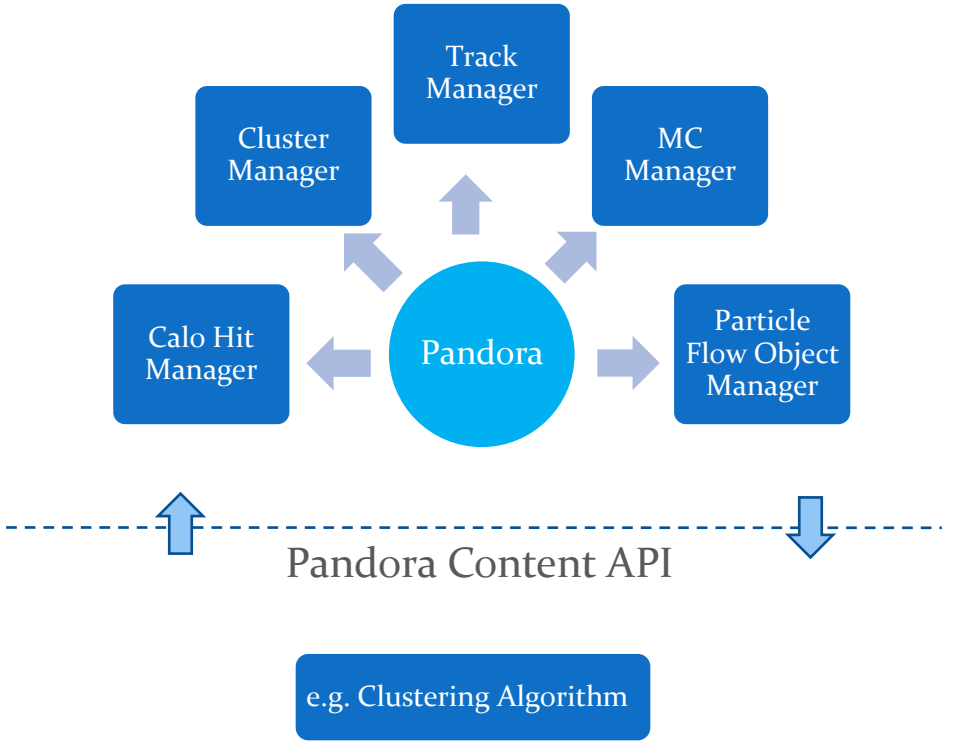
Mixture of properties specified by user and value-added properties, but all simple and well defined physics quantities for use in particle flow algorithms.

Pandora Managers

- Pandora Managers are designed to store named lists of their respective objects.
- These objects can be accessed by the Pandora Algorithms, which perform the reconstruction.
- The algorithms interact with the Managers in a controlled way, via PandoraContentAPI, and the Managers perform the memory management.

- At any instant each Manager has a "current" list, which can be accessed by an algorithm.
- Parent algorithms can manipulate the current list in order to control scope and behaviour of daughter algorithms.

- The Managers store information about currently running algorithms so they can keep track of lists.
- Algorithms can use the PandoraContentAPI to modify lists and/or save new lists.



Algorithms can use the API without worrying about how the managers work – separation of physics and C++ memory management!

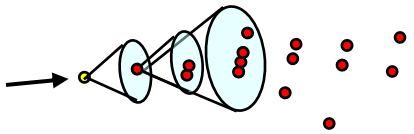
Pandora Algorithms

- In the new Pandora framework, the algorithms contain almost exclusively physics-driven code, alongside the following typical usages of the PandoraContentAPI:
  - Create new clusters and particle flow objects
  - Modify clusters, by adding hits, merging or deleting
  - Access the current lists of Pandora objects
  - Save new lists of clusters, calo hits or tracks
  - Run a daughter algorithm, etc…
- Static helper functions are provided to perform tasks that are useful to multiple algorithms, such as functions to evaluate the overlap between two clusters or to perform a linear fit to (layers of) a cluster.
- The Pandora algorithms are configured via xml and can be swapped in/out without recompiling. The algorithms required to reproduce old Pandora performance are:
  - Clustering
  - Topological associations
  - Fragment Removal
  - Photon Id
  - Statistical reclustering
  - Track-cluster association
  - Particle flow object formation

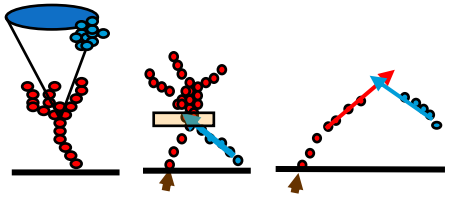
Example Algorithm: Clustering

- To reproduce original Pandora clustering algorithm in new framework, require:
  i. A parent algorithm to control operations,
  ii. A cluster formation algorithm,
  iii. A topological association algorithm (which may itself run multiple daughter algorithms).

- Parent algorithm asks to run a clustering algorithm; cluster manager then creates a new temporary cluster list, associated with the parent algorithm, sets this as "current", and allows new clusters to be formed.



- Daughter clustering algorithm gets current calo hit and track lists, uses its logic to populate temporary cluster list and returns control to parent algorithm.



- Parent algorithm then calls topological association algorithm, which cannot form new clusters, but can modify or merge existing clusters.

- Parent algorithm can then save (a subset of) the temporary clusters as a new named cluster list. Can set new list to be the current list for future algorithms, if desired. Any remaining temporary cluster s will be tidied automatically.

Example Algorithm: Reclustering

- An important part of the Pandora reconstruction is "statistical reclustering", in which attempts are made to redistribute hits between clusters in order to improve consistency between cluster energies and associated track momenta.
- The new algorithm framework and idea of parent algorithms controlling daughter algorithms is designed to make reclustering simple and flexible. A parent reclustering algorithm needs only to perform following operations:

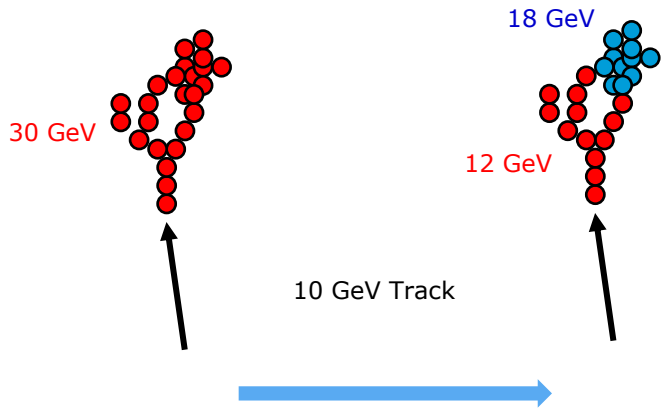
i. Identify inconsistent pairing of track and cluster(s) and ask to recluster these.
   - Relevant clusters will be moved to a new temporary cluster list, associated with the parent algorithm. Current calo hit/track lists changed.

ii. Ask to run a clustering algorithm.
   - This will create another uniquely named temporary cluster list, which will be filled by the daughter clustering algorithm.

iii. Calculate a figure of merit for the consistency of the track and new cluster(s).

iv. Repeat stages ii. and iii. as required.
   - Can run copies of the same algorithm, with different clustering parameters, or use entirely different approaches.

v. Choose most appropriate cluster(s).
   - Cluster lists will be reorganised and tidied accordingly.



Change clustering parameters and/or clustering algorithm until cluster splits and get sensible track-cluster match

Algorithm Configuration

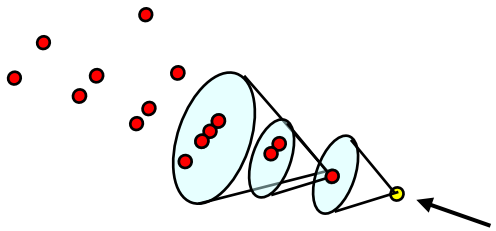
- Pandora algorithms are configured via xml, a very natural way to configure nested algorithms:
  - Ideal for quickly experimenting with running new algorithms, or exploring new methods to address problems such as leakage. Easy to mix “real” and “cheating” algorithms.
  - The complicated process of reclustering is reduced to the following simple configuration:

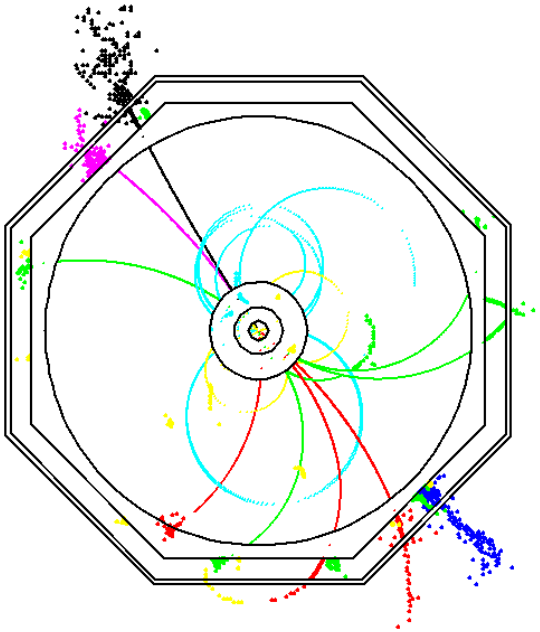
```
<!-- Parent reclustering algorithm runs multiple clustering algorithms -->
<algorithm type = "Reclustering”>
    <!-- List of daughter clustering algorithms -->
    <clusteringAlgorithmList>
        <algorithm type = "ClusteringType1”> ClusteringType1 parameters ... </algorithm>
        <algorithm type = "ClusteringType2”> ClusteringType2 parameters ... </algorithm>
        <algorithm type = "ClusteringTypeN”> ClusteringTypeN parameters ... </algorithm>
    </clusteringAlgorithmList >

    <!-- Other parent reclustering algorithm properties ... -->
</algorithm>
```

- Pandora will provide a comprehensive library of built-in algorithms. However, want other people to get involved:
  - Quick and easy for a user to create their own algorithm. We provide a template; just need to inherit from the Pandora Algorithm base class. User then calls functions available via API and implements their algorithm logic.
  - The new type of algorithm is registered with Pandora via an API. User then just needs to add the algorithm settings to the xml file (there is an XmlHelper to make reading these settings very easy).
  - The algorithm will then run in the Pandora framework, without needing to recompile Pandora.

Algorithm Status





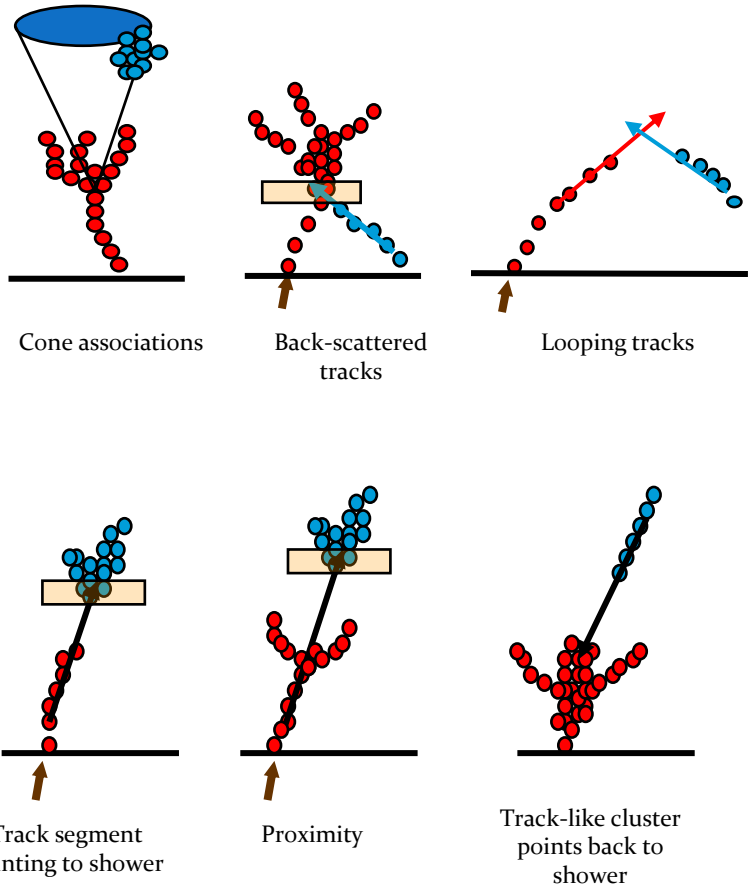
Clustering

- The main Pandora clustering algorithm is a cone-based forward projective method.
- Working from innermost to outermost pseudolayer, the algorithm either adds hits to existing clusters or uses them to seed new clusters.
- This algorithm has been fully implemented in the new framework and tested extensively; the new code exactly reproduces the old Pandora clusters.
- This re-implementation was an opportunity to 'clean up' an algorithm that had changed many times during development; now more efficient.
- Code is clean and readable and have now fully separated the framework/objects from the actual algorithm.
- Configuration options have been tweaked to identify independent and physically motivated parameters.

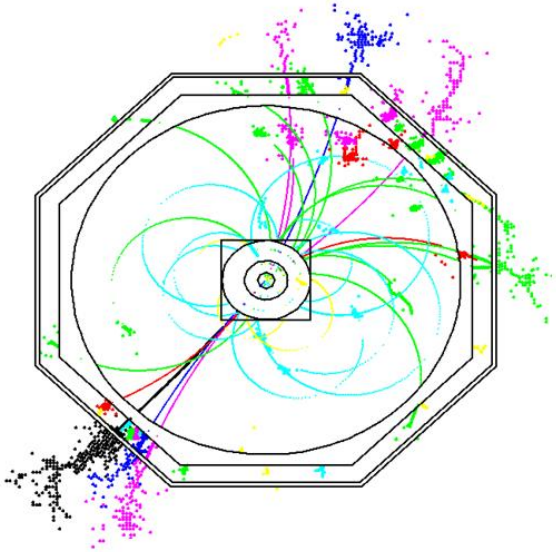
Algorithm Status

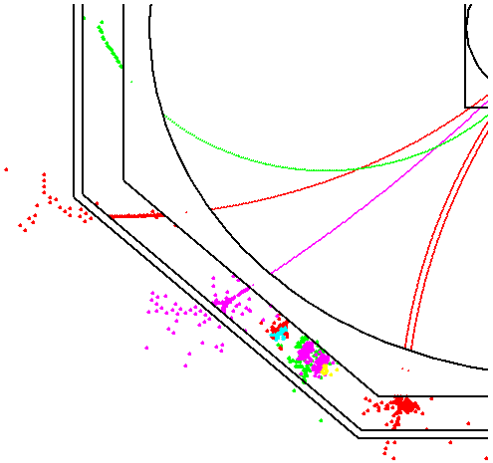
Topological Associations

- The approach in Pandora is to err on side of splitting up true clusters, then merge the clusters following a number of topological rules.
- Each of these rules has been implemented via an algorithm in the new Pandora framework.
- The algorithms essentially compare pairs of clusters, applying a series of cuts to identify whether the clusters should be merged.
- Many of the quantities, upon which cuts are placed, are useful properties characterising cluster interactions. As such, they are calculated by re-usable helper functions.
- The algorithms have been validated and they fully reproduce the performance of the old Pandora code.
- During this process, some improvements were identified and these have been implemented. The improvements were cross-checked using old Pandora .



Algorithm Status





Track-Cluster Associations

- The basic method for associating tracks to clusters is to examine the distance of closest approach between a cluster and the projected track state at the ECal.
- Algorithms have also been implemented that consider the following:
  - The distance between clusters and the projection of a track helix fit.
  - The consistency of projected track directions and initial cluster directions.
  - The consistency of the track momentum and the cluster energy.
- The newly implemented algorithms exactly reproduce the performance of the old Pandora track-cluster association code.
- In the new framework, it is simple to experiment with new association methods or to alter the point(s) in the reconstruction at which associations are made.

Algorithm Status

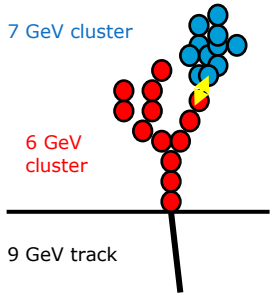
Fragment Removal

- After the topological associations, there are still a significant number of neutral clusters, which are really fragments of charged particle hadronic showers.
- The fragment removal algorithms attempt to identify these clusters and merge them with the appropriate parent charged cluster.

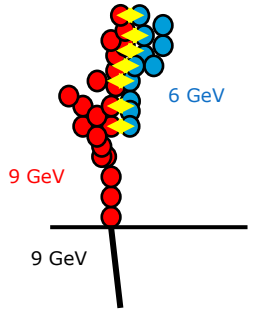
- Associations are made by assessing cluster contact and proximity, together with track association information.
- Many of the quantities used to identify fragments have been implemented as re-usable helper functions.

- An algorithm has also been implemented to collect together the fragments of neutral hadron clusters.

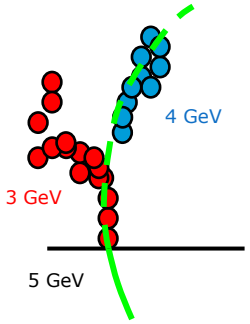
- The fragment removal algorithms have been fully implemented and validated. They exactly reproduce the performance of the old Pandora code.



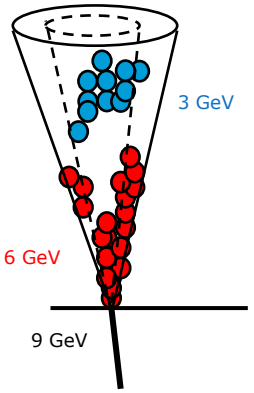
Distance of closest approach



Layers in close contact

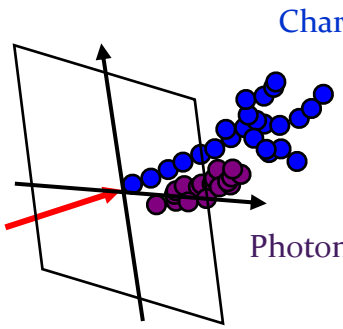


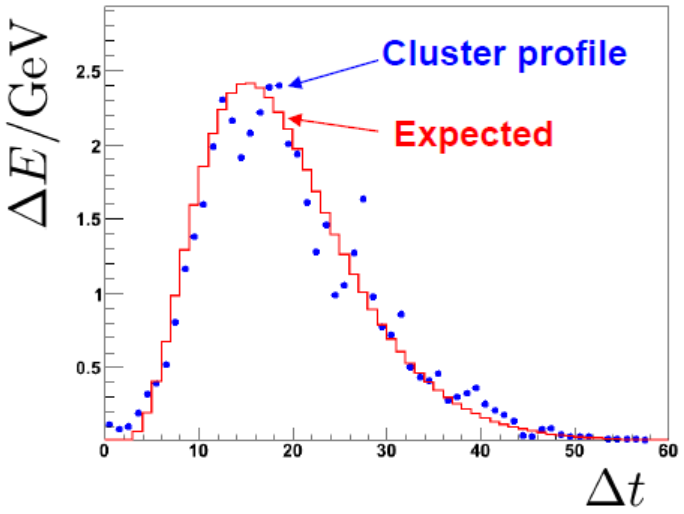
Distance to track extrap.



Fraction of energy in cone

Algorithm Status





Photon ID

- The old version of Pandora offered several approaches to photon cluster identification.
- In the new Pandora, a fast photon id helper function has been provided, which reproduces the basic photon id from the old code.
- Shower profile id helper functions have also been provided.

- Peter Speckmayer has been working on a separate algorithm, which takes the output from the clustering algorithm, applies a shower-profile based selection and saves the photon clusters as a separate named cluster list.
- The removal of these clusters allows for improved identification of hadronic showers when the clustering algorithm is called again.
- Can simply "plug-in" this algorithm to provide an alternative photon ID, without changing any other code.

Note: performance plots shown later use only fast photon id.

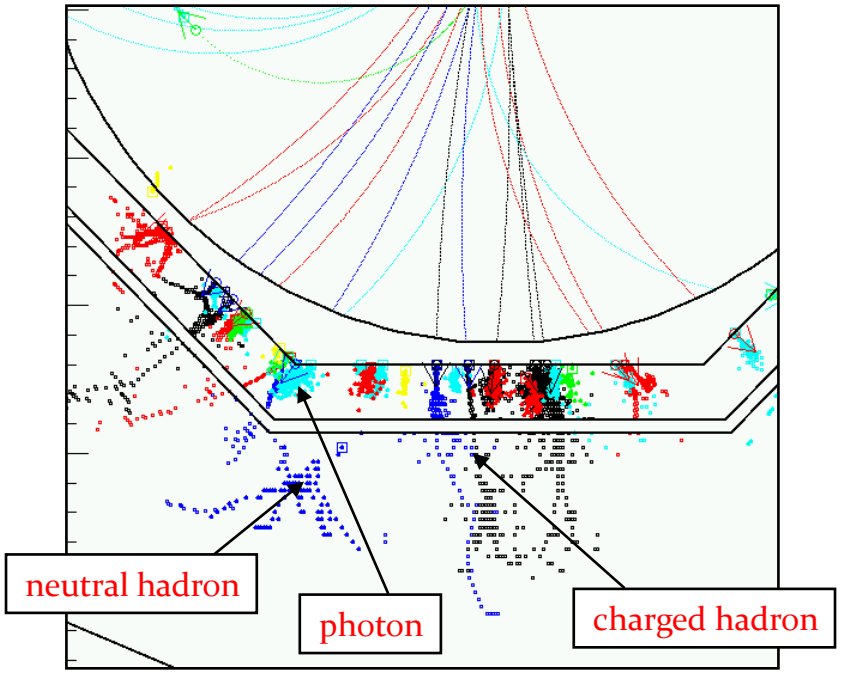
Algorithm Status

PFO Formation

- The final step is to identify the particle flow objects from the tracks and clusters and to write them out for analysis.
- The current PFO formation algorithm is rather simple, with no sophisticated particle identification yet applied.
- In the long term, aim to provide particle identification functions and to allow user to register their own, detector specific, particle id helper functions via PandoraAPI.

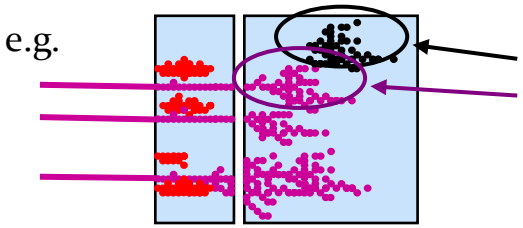
- In the current code, charged PFOs are created from tracks with associated clusters; remaining clusters are used to form neutral PFOs.
- Tracks which have no associated clusters, but which are deemed to be low p_T, are still used to form charged PFOs.
- Fast photon id is used to determine whether neutral PFOs should use electromagnetic or hadronic energy measure.

- The PFO creation is ready to respond to any track relationship information provided to identify track kinks, v^0s, etc.



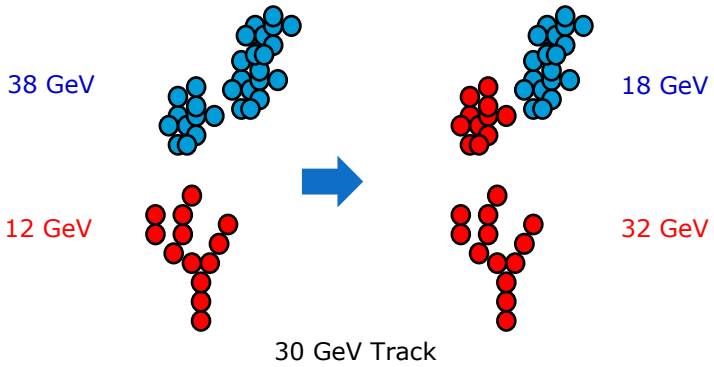
Remaining Work

- At high jet energies, performance degrades due to increasing overlap between hadronic showers from different particles:



If these hits are clustered together with these, lose energy deposit from neutral hadron (now part of track particle) and ruin jet energy measurement.

- Pandora addresses this problem with **statistical reclustering**, illustrated earlier.
  - Clusters that have been incorrectly merged together are identified via consistency of cluster energy and associated track momentum.
  - Attempts are made to redistribute the hits by using different clustering parameters or entirely different clustering algorithms.

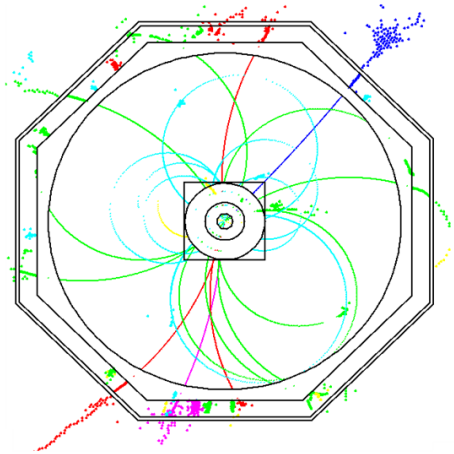


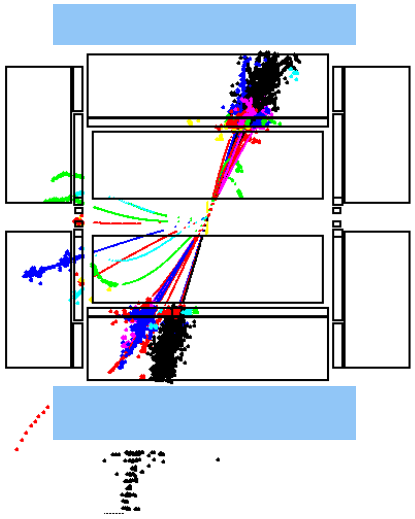
- Algorithms that "steer" the reclustering have been implemented and observed to work. However, no formal validation yet.
- Want to spend some time here, deviating from original Pandora, to tidy these steering algorithms and more clearly define the role of each.
- Have designed some better defined algorithms that should match (or improve on) performance of old code.

Remaining Work

Track relationship information

- New Pandora code deals with specified track parent, daughter and sibling relationships.
- Uses this information when associating tracks to clusters and when forming charged PFOs. However, have yet to input this information.
- Have external v^0 and kink finder code for ILD. Simply need to pass information to Pandora and test thoroughly – should see immediate improvement in jet energy reconstruction.

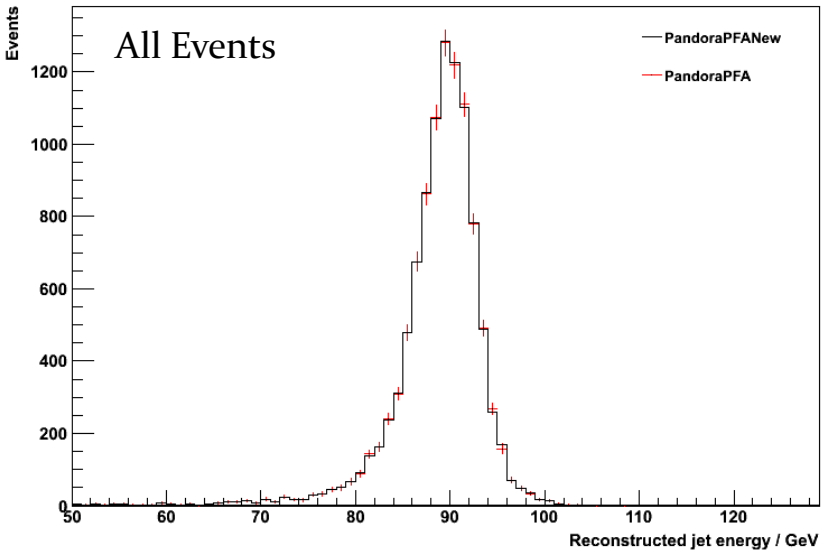


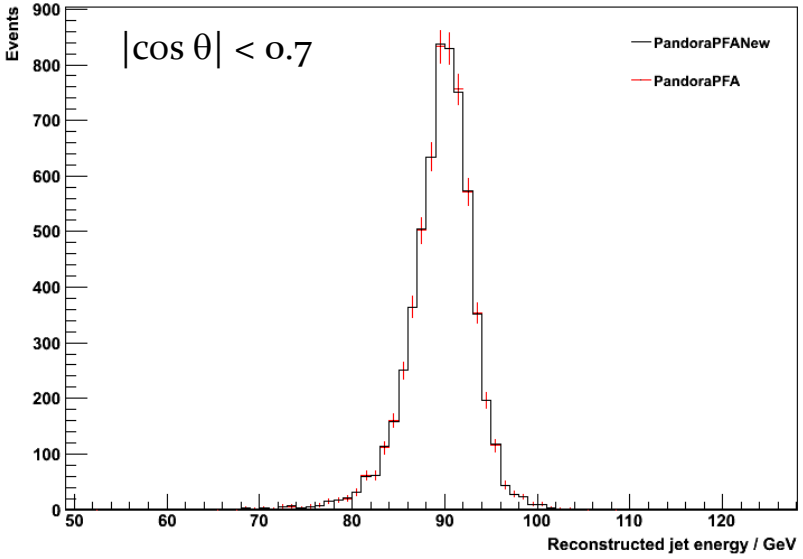


Leakage correction

- For high energy jets, non-containment of showers is significant.
- To address this issue, first need identification of clusters exiting the detector. Can then use this information when e.g. considering track-cluster consistency in reclustering.
- May reproduce old Pandora use of muon chamber information to estimate leakage and energy deposition in the coil, before using flexibility of new framework to investigate new ideas.

First Results 91GeV



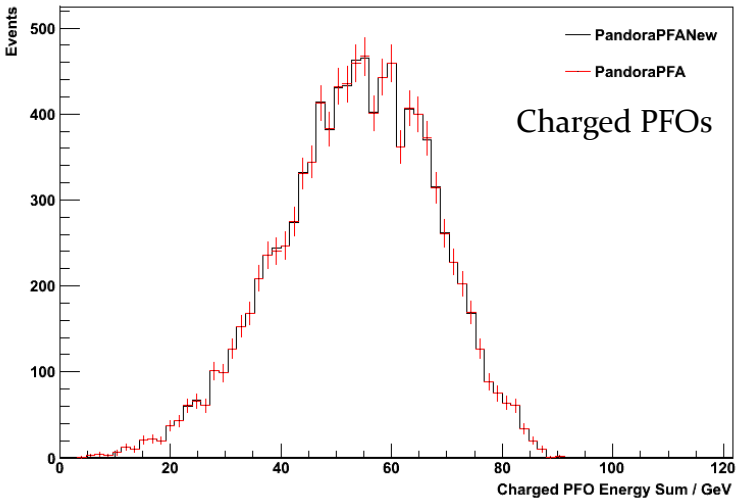


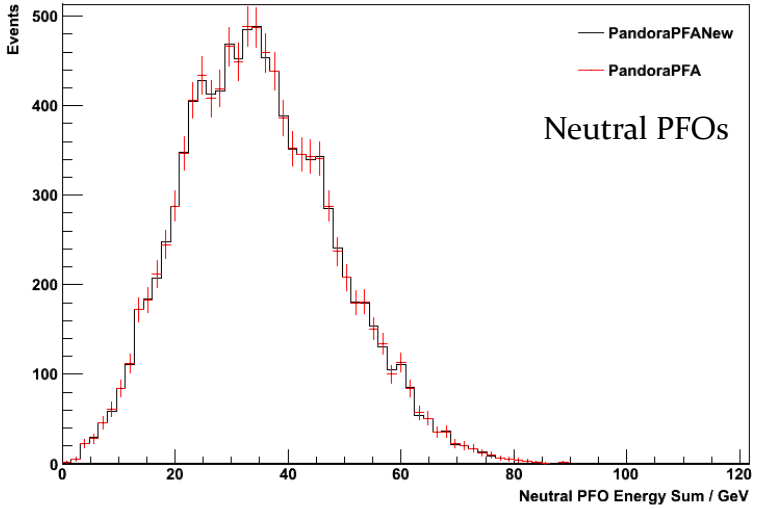
- First tests were performed for the ILD detector concept, using MC samples of approximately 10,000 $Z \rightarrow uds$ generated with the Z decaying at rest with E_z = 91.2GeV.
- At this energy, all newly implemented code and framework is exercised, without (a strong) need for statistical reclustering and/or leakage corrections.
- For fair comparison, reclustering, leakage corrections and all use of track-relationship information was turned off in old Pandora (all still work in progress for new Pandora).
- Excellent agreement observed, by construction (and a lot of hard work!).

$$\sigma_E / E \ (\mathrm{rms}_{90}) = 3.79\%$$

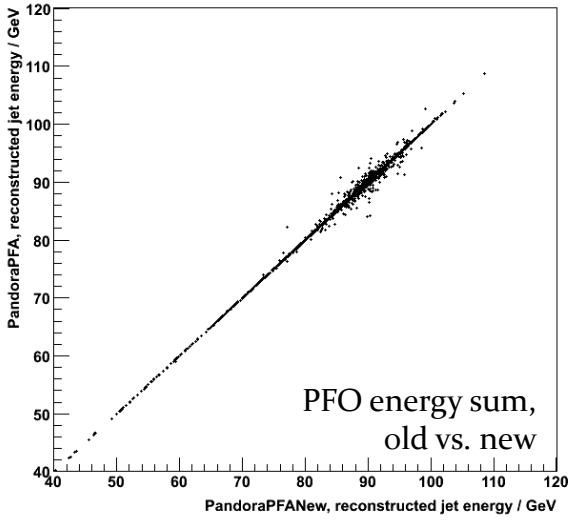
c.f. old Pandora 3.87% without reclustering and track relationship information

First Results 91GeV

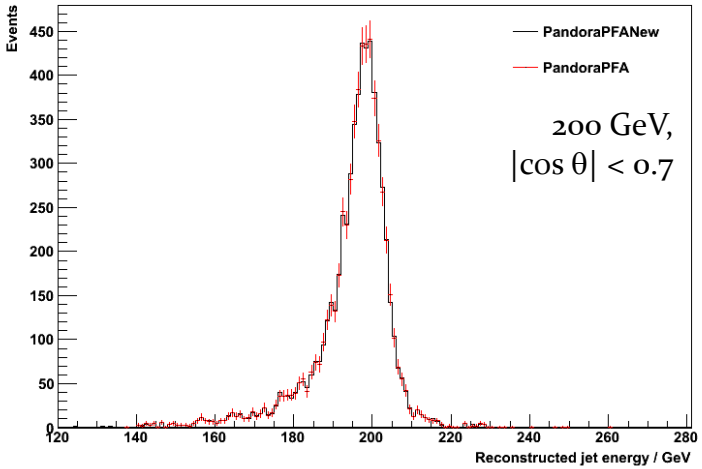


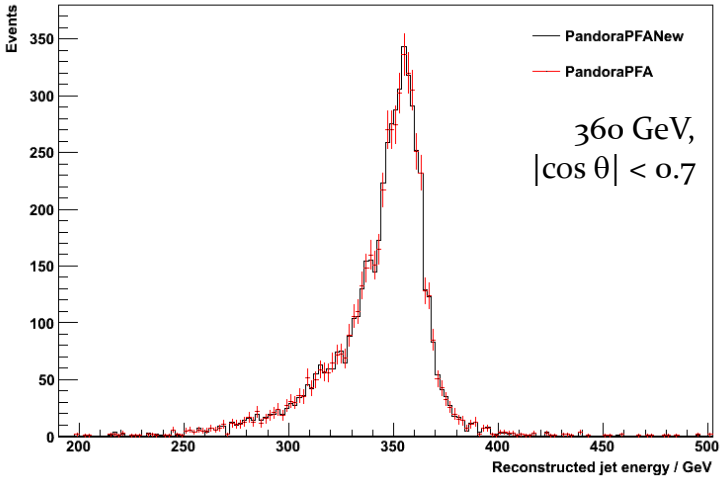


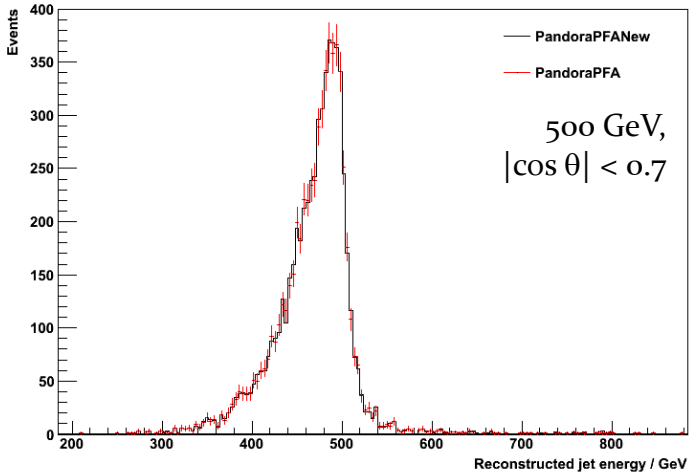
- Looking at the 91GeV results in more details, see that PandoraPFANew reproduces not just the total jet energy, but also the division into charged and neutral PFOS.
- In fact, each PFO is (essentially) identical in its composition and energy measure.
- There are, however, some small known and understood differences, where it has been necessary to use slightly different approach.



First Look at Higher Energies







- These plots give only a first indication of performance at higher energies.
- Close agreement observed between new and old code, for the algorithms that have been implemented.
- Can't use as a measure of performance yet, as important details for high energy missing.
- Remaining work involves reproducing (and improving on) full treatment of high energy events provided by the old Pandora code.

Summary

- The new Pandora framework is complete and has been rigorously tested. It is stable and has been unchanged for several months now.

- The majority of original Pandora algorithms have now been fully implemented and tested, providing a full reconstruction (and exact reproduction of old Pandora results) at low energies.
  - Now need to address high energy issues; work on leakage corrections and statistical reclustering.
  - Will start to deviate from the old Pandora approach in reclustering, as aim for better defined algorithms that match (or improve on) original performance.

- Should also mention progress with the applications that use the Pandora library:
  - ILD Pandora application completed (our default test application),
  - Norman Graf and Jeremy McCormick have started a SlicPandora application,
  - Both applications are Marlin Processors that use the PandoraAPI to access the Pandora library.

- After the high energy reconstruction is complete, have many ideas for moving forwards and improving Pandora within the new framework:
  - New clustering algorithms,
  - Long list of possible topological association improvements,
  - More sophisticated particle identification,
  - Very easy for other people to get involved and write new algorithms or contribute ideas.