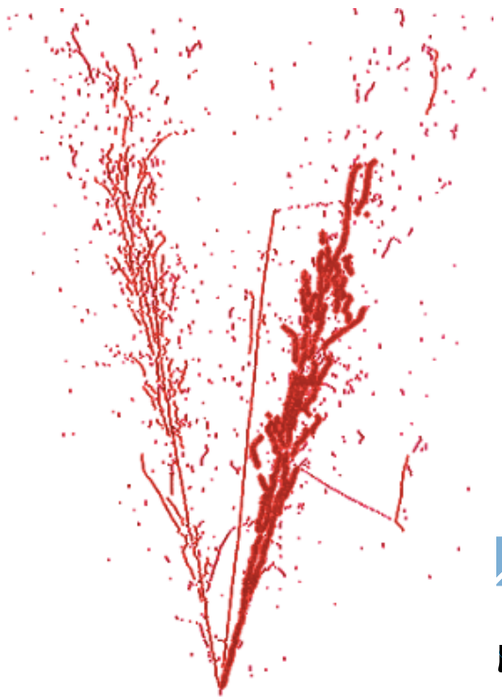
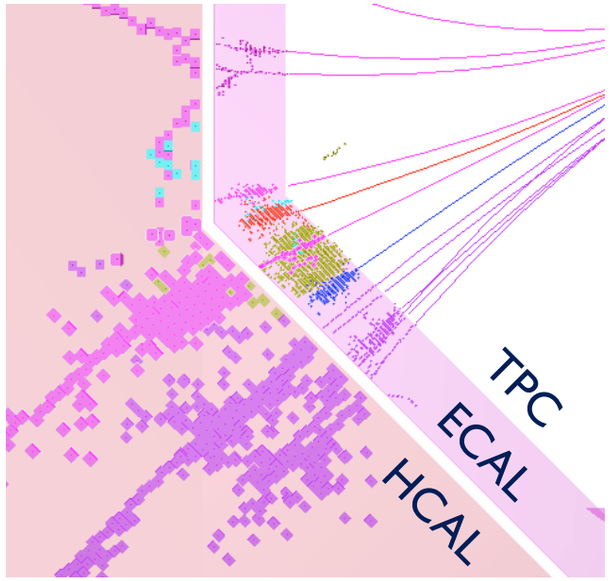


Working With Pandora

J. S. Marshall

High Level Reconstruction Week

6 July 2015

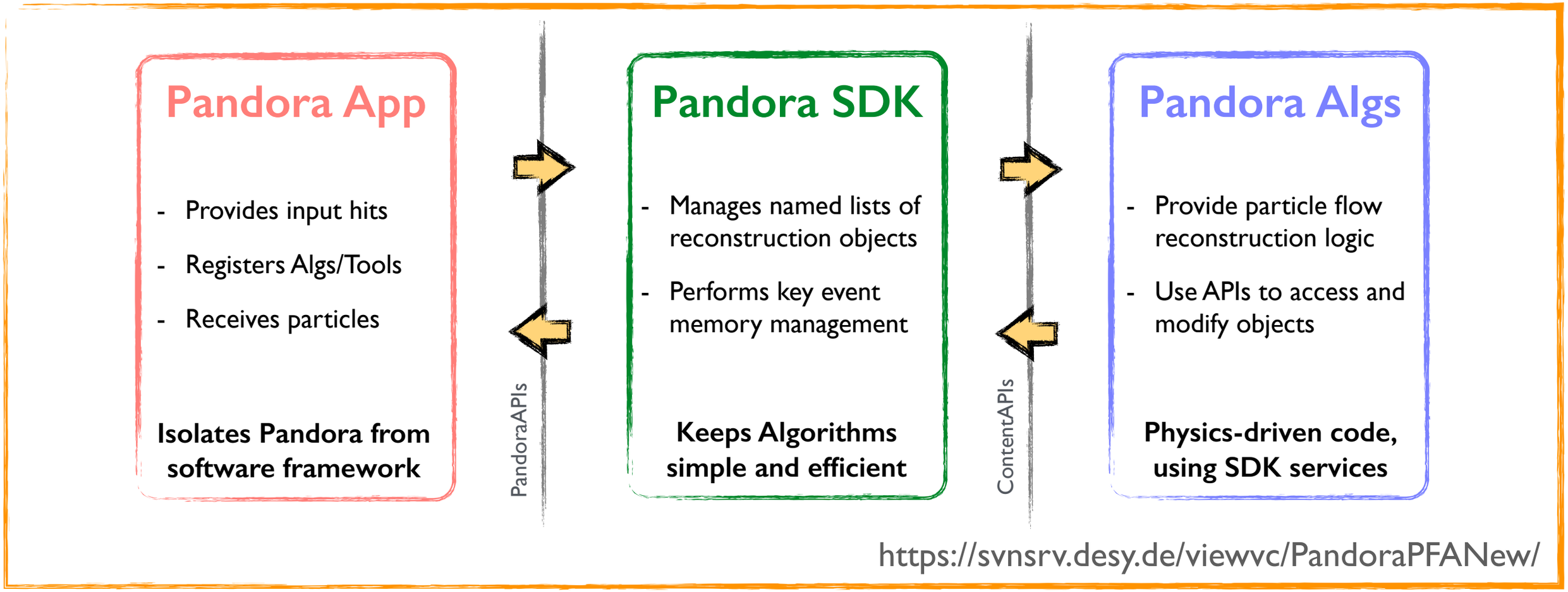


Pandora Approach

The Pandora Software Development Kit provides a software environment in which:

1. It is easy for users to provide the building-blocks that define a pattern recognition problem.
2. Logic required to solve pattern recognition problems is cleanly implemented in algorithms.
3. Operations to access or modify building-blocks, or to create new structures, are requested by algorithms and performed by the Pandora framework.

This promotes the use of large numbers of algorithms, each addressing specific event topologies.

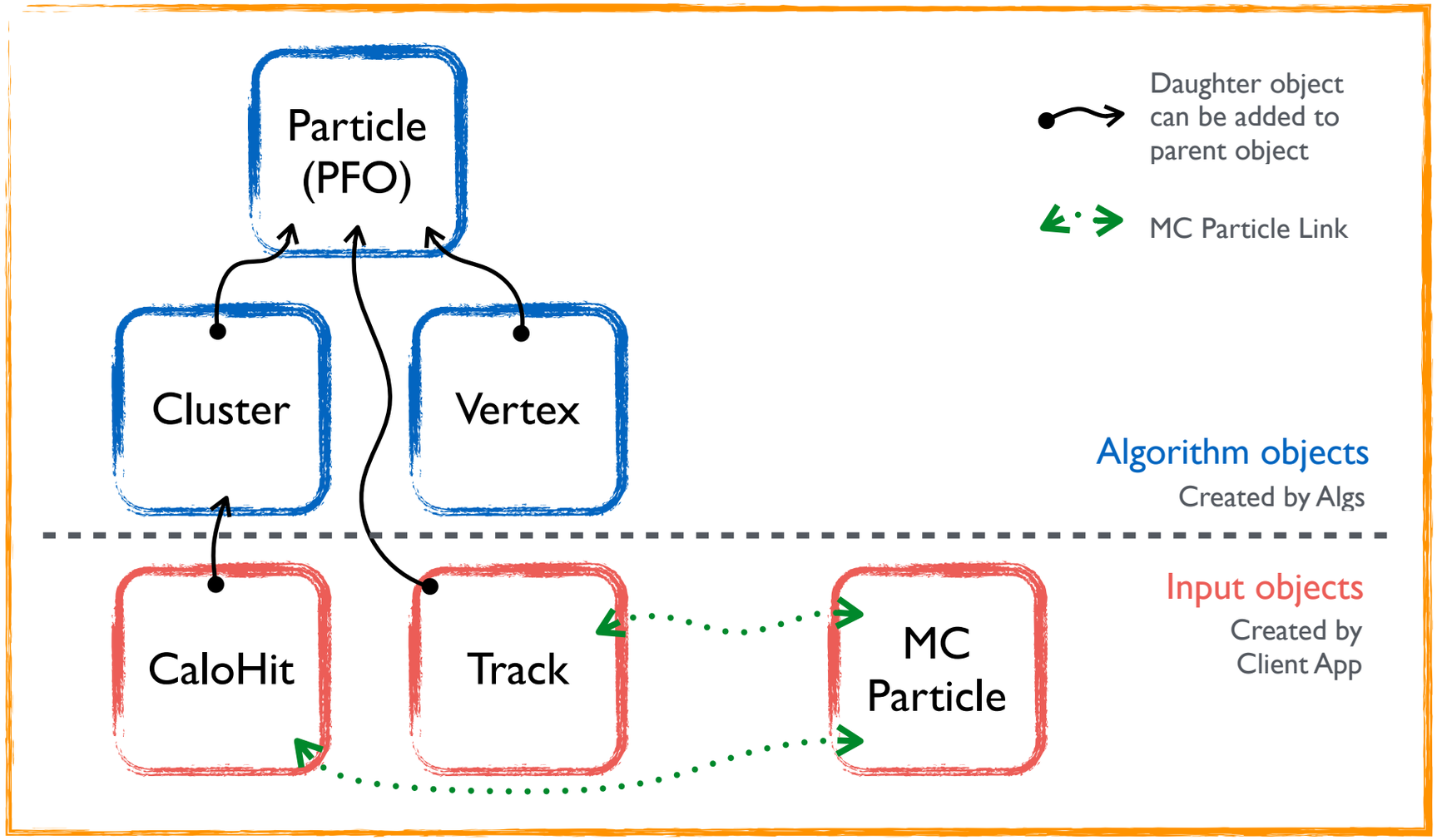


https://svnsrv.desy.de/viewvc/PandoraPFANew/

Pandora SDK

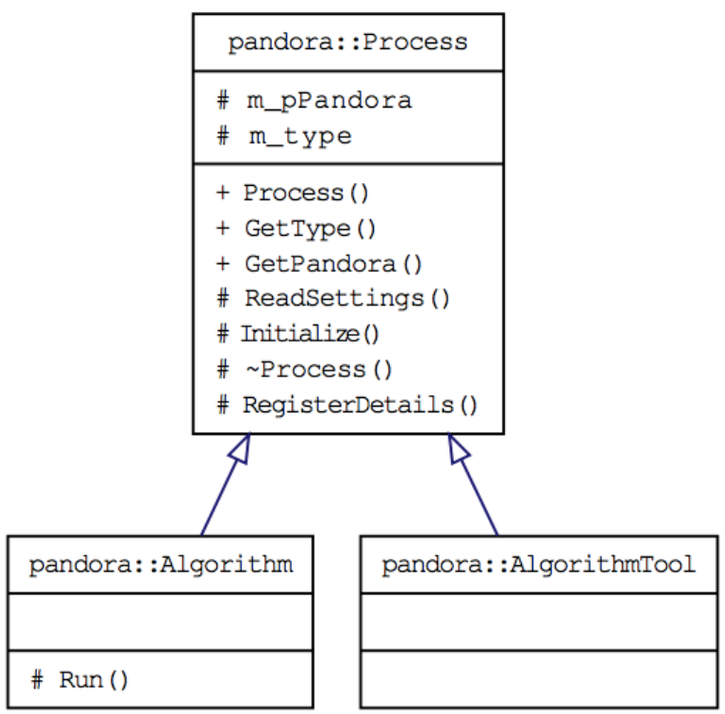
- The Pandora SDK provides a comprehensive **Event Data Model** (EDM) for managing pattern recognition problems. Instances of objects in the EDM are owned by **Pandora Managers**.
- The object instances are stored in named lists and the Managers are able to create new objects, delete objects, create and save new lists and move objects between lists.
- The Managers provide a complete set of low-level operations that allow the high-level operations requested by pattern recognition algorithms to be satisfied.

```
pandora::Pandora
- m_pAlgorithmManager
- m_pCaloHitManager
- m_pClusterManager
- m_pGeometryManager
- m_pMCManager
- m_pPfoManager
- m_pPluginManager
- m_pTrackManager
- m_pVertexManager
- m_pPandoraSettings
- m_pPandoraApiImpl
- m_pPandoraContentApiImpl
- m_pPandoraImpl
+ Pandora()
+ ~Pandora()
+ GetPandoraApiImpl()
+ GetPandoraContentApiImpl()
+ GetSettings()
+ GetGeometry()
+ GetPlugins()
- PrepareEvent()
- ProcessEvent()
- ResetEvent()
- ReadSettings()
```



Pandora Algorithms

- Pandora algorithms contain the step-by-step instructions for finding patterns in the provided data.
- They use the APIs to access objects and to request the Managers to make new objects or modify existing objects.
- They inherit from the **Process** class, which provides functionality for handshaking with Pandora, XML config and function callbacks.



Algorithm 1 Cluster creation pseudocode. The logic determining when to create new Clusters and when to extend existing Clusters will vary between algorithms.

```
procedure CLUSTER CREATION
    Create temporary Cluster list
    Get current CaloHit list
    for all CaloHits do
        if CaloHit available then
            for all newly-created Clusters do
                Find best host Cluster
            if Suitable host Cluster found then
                Add CaloHit to host Cluster
            else
                Add CaloHit to a new Cluster
    Save new Clusters in a named list
```

Algorithm 2 Cluster merging pseudocode. The logic governing the identification of suitable parent Clusters and daughter Clusters will vary between algorithms.

```
procedure CLUSTER MERGING
    Get current Cluster list
    for all Clusters do
        if Cluster is suitable parent then
            for all Clusters do
                Find best daughter Cluster
            if Suitable daughter Cluster found then
                Merge daughter Cluster into Parent
```

Pandora Client App

- To use the Pandora SDK, a user must create a Pandora client application. This provides the input building-blocks to describe the pattern recognition problem and receives the final output.
- The client application is responsible for controlling the pattern recognition reconstruction. It creates the Pandora instance(s) and uses Pandora APIs to send requests to these instances.

- For ILD, the Pandora client application is **MarlinPandora**
- All Algorithms and Plugins are built as part of the LCContent library.
- The Pandora geometry information is provided using GEAR.
- Pandora CaloHits, Tracks, etc. are extracted from lcio collections.
- The output consists of a lcio ReconstructedParticle collection.

Algorithm 3 Pseudocode description of a client application used for Linear Collider event reconstruction.

```
procedure MAIN
    Create a Pandora instance
    Register Algorithms and Plugins
    Provide detector geometry description
    Ask Pandora to parse XML settings file
    for all Events do
        Create Track instances
        Create CaloHit instances
        Create MCParticle instances
        Specify Track-Track relationships
        Specify MCParticle-Track relationships
        Specify MCParticle-CaloHit relationships
        Ask Pandora to process the event
        Get output PFOs and write to file
        Reset Pandora before next event
```

Pandora SDK Paper

http://arxiv.org/abs/1506.05348

The Pandora Software Development Kit for Pattern Recognition

J. S. Marshall*, M. A. Thomson

Cavendish Laboratory, University of Cambridge, Cambridge, United Kingdom

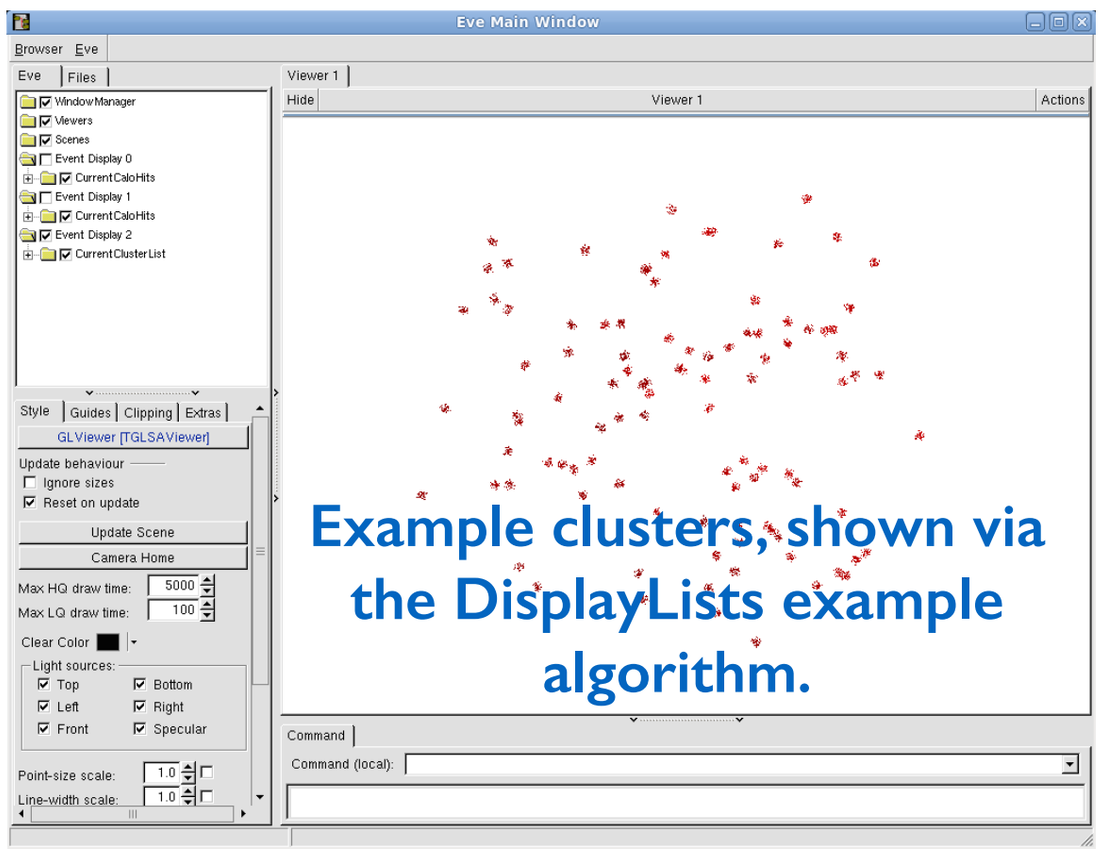
Abstract

The development of automated solutions to pattern recognition problems is important in many areas of scientific research and human endeavour. This paper describes the implementation of the Pandora Software Development Kit, which aids the process of designing, implementing and running pattern recognition algorithms. The Pandora Application Programming Interfaces ensure simple specification of the building-blocks defining a pattern recognition problem. The logic required to solve the problem is implemented in algorithms, with all operations to create or modify event data structures requested by algorithms and performed by the Pandora framework. This design promotes an approach using many decoupled algorithms, each addressing specific topologies. Details of algorithms addressing two pattern recognition problems in High Energy Physics are presented: reconstruction of events at a high-energy e^+e^- linear collider and reconstruction of cosmic ray or neutrino events in a liquid argon time projection chamber.

Keywords: Software Development Kit, Pattern recognition, High Energy Physics

Pandora Learning Library

- Pandora algorithms create and/or modify clusters, vertices and PFOs. Their decisions (the algorithm logic) whether to proceed with operations can be complex and use-case specific.
- The aim of the Pandora ExampleContent library and test application is to demonstrate the key Pandora functionality in a very simple testing and learning environment.
- The ExampleContent library is structured in exactly the same manner as the LCContent and LArContent libraries, currently in use for Linear Collider and LAr TPC reconstruction.



Example clusters, shown via the DisplayLists example algorithm.

- The library consists of example Algorithms, AlgorithmTools, Plugins and Helper functions:
  - Example list access and display
  - Example Cluster, Vertex and PFO creation
  - Cluster manipulation, including merging, deletion, fragmentation and reclustering
  - Creating and saving new lists of objects
  - Using Algorithm Tools and Plugins
  - Writing a tree using PandoraMonitoring.

http://www.hep.phy.cam.ac.uk/~marshall/PandoraExample.pdf

Pandora Client App Steering

```xml
<processor name="MyMarlinPandoraDefault" type="PandoraPFANewProcessor">
  <parameter name="PandoraSettingsXmlFile" type="String">PandoraSettingsDefault.xml</parameter>
  <!-- Collection names -->
  <parameter name="TrackCollections" type="StringVec">MarlinTrkTracks</parameter>
  <parameter name="ECalCaloHitCollections" type="StringVec">ECALBarrel ECALEndcap ECALOther</parameter>
  <parameter name="HCalCaloHitCollections" type="StringVec">HCALBarrel HCALEndcap HCALOther</parameter>
  <parameter name="LCalCaloHitCollections" type="StringVec">LCAL</parameter>
  <parameter name="LHCalCaloHitCollections" type="StringVec">LHCAL</parameter>
  <parameter name="MuonCaloHitCollections" type="StringVec">MUON</parameter>
  <parameter name="MCParticleCollections" type="StringVec">MCParticle</parameter>
  <parameter name="RelCaloHitCollections" type="StringVec">RelationCaloHit RelationMuonHit</parameter>
  <parameter name="RelTrackCollections" type="StringVec">MarlinTrkTracksMCTruthLink</parameter>
  <parameter name="KinkVertexCollections" type="StringVec">KinkVertices</parameter>
  <parameter name="ProngVertexCollections" type="StringVec">ProngVertices</parameter>
  <parameter name="SplitVertexCollections" type="StringVec">SplitVertices</parameter>
  <parameter name="V0VertexCollections" type="StringVec">V0Vertices</parameter>
  <parameter name="ClusterCollectionName" type="String">PandoraClustersDefault</parameter>
  <parameter name="PFOCollectionName" type="String">PandoraPFOsDefault</parameter>
  <!-- Calibration constants -->
  <parameter name="ECalToMipCalibration" type="float">160.0</parameter>
  <parameter name="HCalToMipCalibration" type="float">34.8</parameter>
  <parameter name="ECalMipThreshold" type="float">0.5</parameter>
  <parameter name="HCalMipThreshold" type="float">0.3</parameter>
  <parameter name="ECalToEMGeVCalibration" type="float">1.007</parameter>
  <parameter name="HCalToEMGeVCalibration" type="float">1.007</parameter>
  <parameter name="ECalToHadGeVCalibrationBarrel" type="float">1.075</parameter>
  <parameter name="ECalToHadGeVCalibrationEndCap" type="float">1.075</parameter>
  <parameter name="HCalToHadGeVCalibration" type="float">1.027</parameter>
  <parameter name="MuonToMipCalibration" type="float">10.0</parameter>
  <parameter name="DigitalMuonHits" type="int">0</parameter>
  <parameter name="MaxHCalHitHadronicEnergy" type="float">1.</parameter>
  <!-- Absorber properties -->
  <parameter name="AbsorberRadLengthECal" type="float">0.2854</parameter>
  <parameter name="AbsorberIntLengthECal" type="float">0.0101</parameter>
  <parameter name="AbsorberRadLengthHCal" type="float">0.0569</parameter>
  <parameter name="AbsorberIntLengthHCal" type="float">0.0060</parameter>
  <parameter name="AbsorberRadLengthOther" type="float">0.0569</parameter>
  <parameter name="AbsorberIntLengthOther" type="float">0.0060</parameter>
  <!--Whether to calculate track states manually, rather than copy stored fitter values-->
  <parameter name="UseOldTrackStateCalculation" type="int">0</parameter>
</processor>
```

Pandora alg steering

Input and output collection names

Pandora calibration constants

Additional geometry information

Support for old tracking software

Pandora Calibration

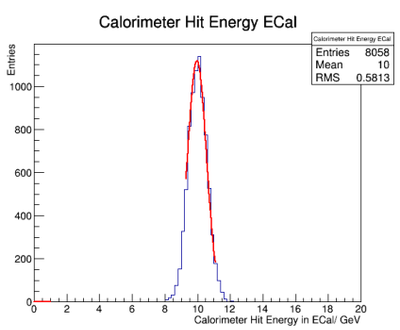
S. Green.

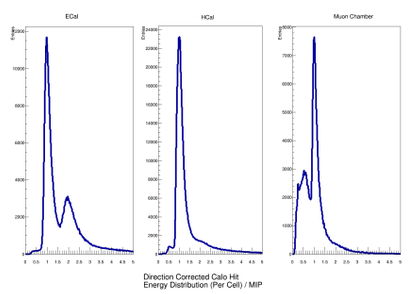
Simulation Output
Energy Deposited in Active Material of Calorimeters

Calorimeter Energy
Energy Deposited in the Total Calorimeter Volume

Particle Energies
Energy of individual reconstructed particles

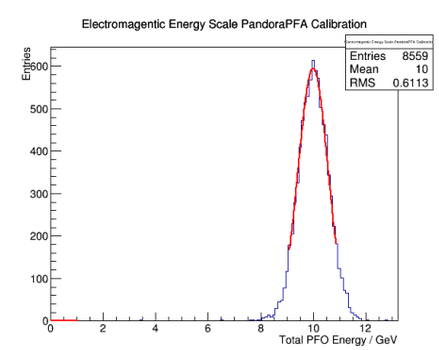
Digitisation
Set by looking at contained kaonL and γ events throughout the calorimeters.

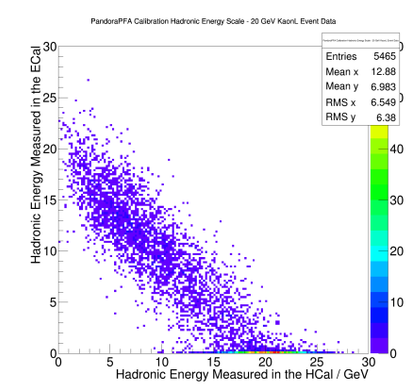




Minimum Ionising Particle (MIP) Detector Response
Determine the response of each part of the detector to a MIP, μ^- events. MIP scale used in PandoraPFA.

Electromagnetic and Hadronic Scale
Electromagnetic/Hadronic scale set using PFO energy of contained γ and kaonL events.





The **PandoraAnalysis** toolkit has scripts designed for setting the digitisation and calibration constants. The user has to provide samples of kaonL, γ and μ^-.
These scripts make semi-automated calibration possible. Procedure is fully documented.

Pandora Algorithm Steering

- **Pandora is configured via an XML file, provided by the client application.**
- It looks for algorithm XML tags within the top level Pandora tags, creating instances of any algorithms found. It will run these algorithms, in order, for each event.
- Each algorithm receives a ReadSettings callback, with a provided XML handle. Algorithms can have mandatory or optional parameters (override default values).
- Algorithms can use the ReadSettings callback to control the creation of daughter Algorithms or AlgorithmTools. Allows for use of (multiple) alternative approaches to solving a problem.

```xml
<!-- Pandora settings xml file -->
<pandora>
    <!-- GLOBAL SETTINGS -->
    <IsMonitoringEnabled>true</IsMonitoringEnabled>
    <ShouldDisplayAlgorithmInfo>false</ShouldDisplayAlgorithmInfo>
    <ShouldCollapseMCParticlesToPfoTarget>true</ShouldCollapseMCParticlesToPfoTarget>

    <!-- PLUGIN SETTINGS -->
    <HadronicEnergyCorrectionPlugins>CleanClusters ScaleHotHadrons</HadronicEnergyCorrectionPlugins>
    <EmShowerPlugin>LCEmShowerId</EmShowerPlugin>
    <PhotonPlugin>LCPhotonId</PhotonPlugin>
    <ElectronPlugin>LCElectronId</ElectronPlugin>
    <MuonPlugin>LCMuonId</MuonPlugin>

    <!-- ALGORITHM SETTINGS -->

    <!-- Set calo hit properties, then select tracks and hits to use for clustering -->
    <algorithm type = "CaloHitPreparation"/>
    <algorithm type = "EventPreparation">
        <OutputTrackListName>Tracks</OutputTrackListName>
        <OutputCaloHitListName>CaloHits</OutputCaloHitListName>
        <OutputMuonCaloHitListName>MuonYokeHits</OutputMuonCaloHitListName>
        <ReplacementTrackListName>Tracks</ReplacementTrackListName>
        <ReplacementCaloHitListName>CaloHits</ReplacementCaloHitListName>
    </algorithm>
    …SNIP…
```

Pandora XML tag opened here

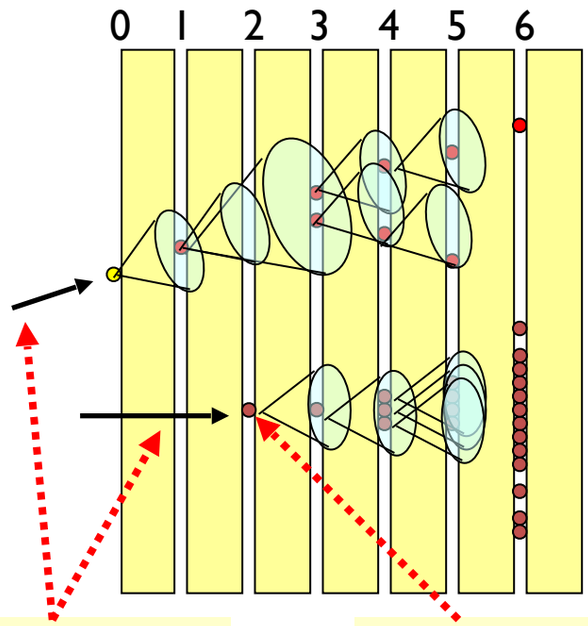
Non-default global parameters

Particle Id and Energy correction Plugins

First two algorithms, with required parameters

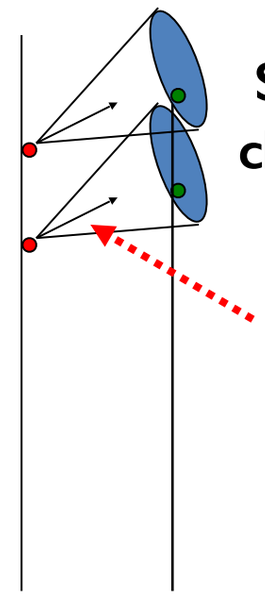
Pandora Clustering Example

- **Philosophy:** "It's easier to put clusters together, than to split them up again."
- Clustering algorithm very careful to avoid accidentally merging energy deposits from separate particles.



Initial cluster direction

Unmatched hit seeds new cluster

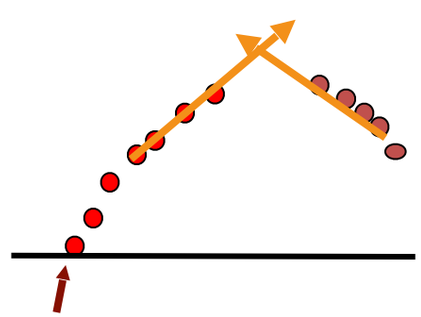


Simple cone-based clustering algorithm

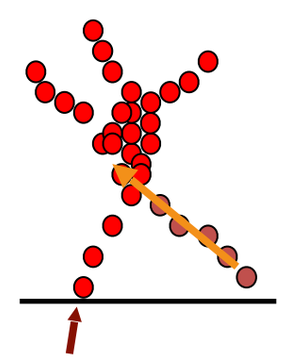
Cones based on either: initial direction or current direction

Topological Associations

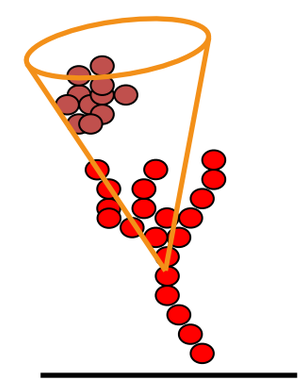
- Fine granularity of the calorimeters exploited to merge cluster fragments that are clearly associated.
- Very few mistakes made.



Looping tracks



Back-scattered tracks



Cone associations

Pandora Clustering Config

Parent clustering algorithm

Cluster formation daughter algorithm

```
<algorithm type = "ClusteringParent">
    <algorithm type = "ConeClusteringFast" description = "ClusterFormation"/>
    <algorithm type = "TopologicalAssociationParent" description = "ClusterAssociation">
        <associationAlgorithms>
            <algorithm type = "LoopingTracks"/>
            <algorithm type = "BrokenTracks"/>
            <algorithm type = "ShowerMipMerging"/>
            <algorithm type = "ShowerMipMerging2"/>
            <algorithm type = "BackscatteredTracks"/>
            <algorithm type = "BackscatteredTracks2"/>
            <algorithm type = "ShowerMipMerging3"/>
            <algorithm type = "ShowerMipMerging4"/>
            <algorithm type = "ProximityBasedMerging">
                <algorithm type = "TrackClusterAssociationFast"/>
            </algorithm>
            <algorithm type = "ConeBasedMerging">
                <algorithm type = "TrackClusterAssociationFast"/>
            </algorithm>
            <algorithm type = "MipPhotonSeparation">
                <algorithm type = "TrackClusterAssociationFast"/>
            </algorithm>
            <algorithm type = "SoftClusterMergingFast">
                <algorithm type = "TrackClusterAssociationFast"/>
                <AdditionalClusterListNames>PhotonClusters</AdditionalClusterListNames>
            </algorithm>
            <algorithm type = "IsolatedHitMerging">
                <AdditionalClusterListNames>PhotonClusters</AdditionalClusterListNames>
            </algorithm>
        </associationAlgorithms>
    </algorithm>
    <ClusterListName>PrimaryClusters</ClusterListName>
    <ReplaceCurrentClusterList>true</ReplaceCurrentClusterList>
</algorithm>
```

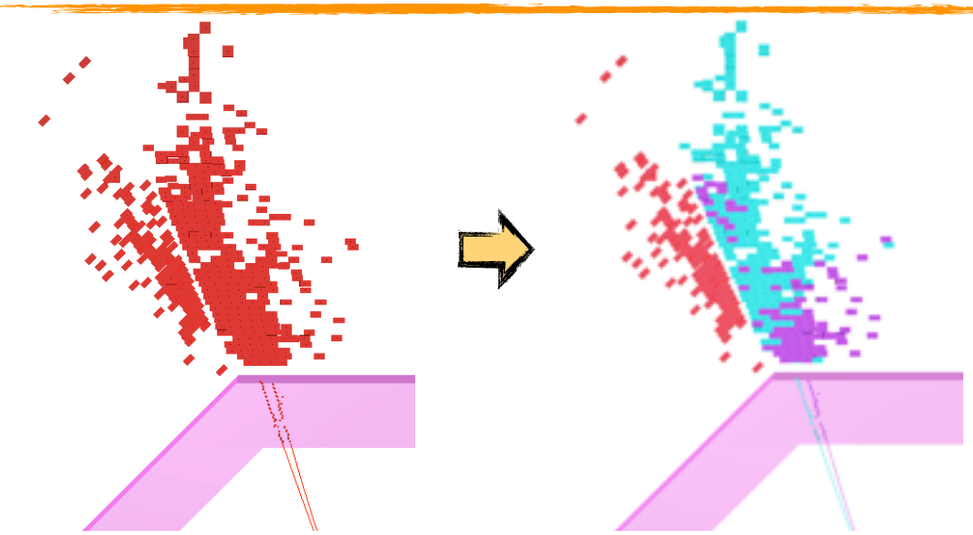
List of topological association daughter algorithms

Some topological association algs use daughter algs

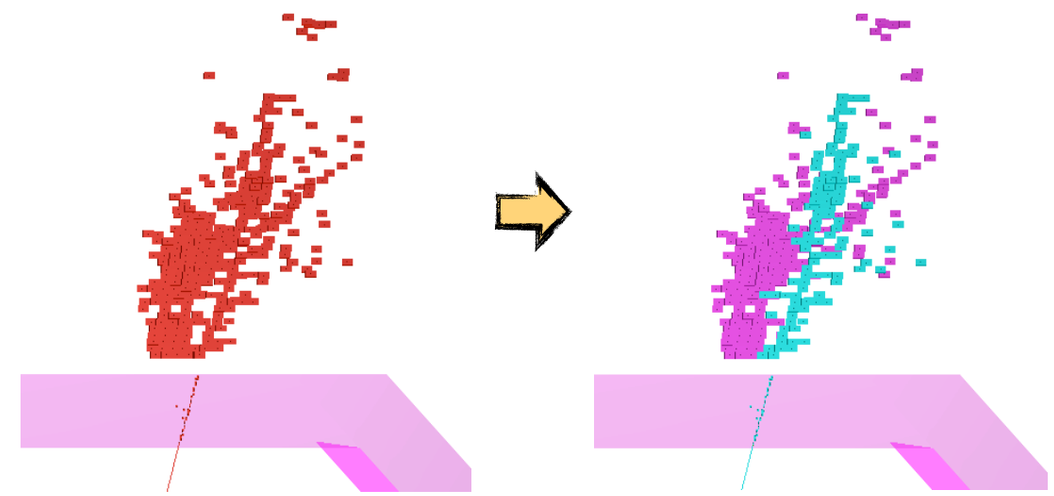
Configuration of output Cluster list

Pandora Reclustering Example

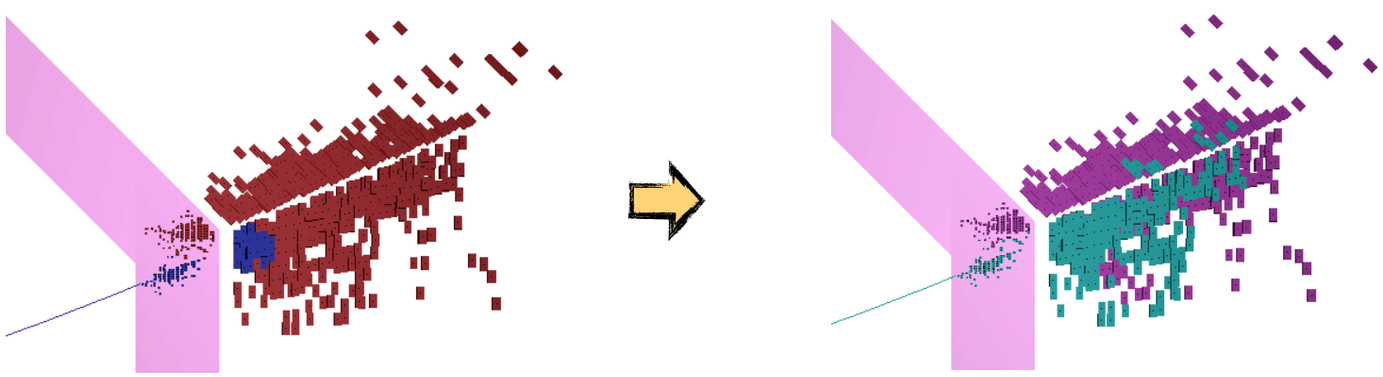
If identify significant discrepancy between cluster energy and associated track momentum, choose to recluster. Alter clustering parameters until cluster splits to obtain track-cluster consistency.



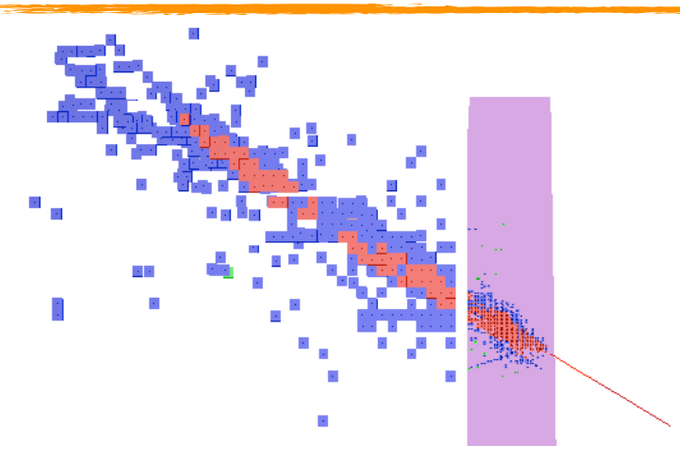
1. Multiple tracks associated to single cluster – split cluster.



2. Cluster energy much greater than track momentum – split cluster.



3. Track momentum much greater than cluster energy – bring in nearby clusters and reconfigure.



4. If, and only if, no E/p match emerges, can force track-cluster consistency ⇒ energy flow.

Pandora Reclustering Config

Parent reclustering algorithm, provides overall logic

```
<algorithm type = "SplitMergedClusters">
    <clusteringAlgorithms>
        <algorithm type = "ConeClusteringFast" instance = "Reclustering1">
            <TanConeAngleFine>0.24</TanConeAngleFine>
            <TanConeAngleCoarse>0.4</TanConeAngleCoarse>
            <AdditionalPadWidthsFine>2</AdditionalPadWidthsFine>
            <AdditionalPadWidthsCoarse>2</AdditionalPadWidthsCoarse>
            <SameLayerPadWidthsFine>2.24</SameLayerPadWidthsFine>
            <SameLayerPadWidthsCoarse>1.44</SameLayerPadWidthsCoarse>
            <MaxTrackSeedSeparation>100</MaxTrackSeedSeparation>
            <MaxLayersToTrackSeed>0</MaxLayersToTrackSeed>
            <MaxLayersToTrackLikeHit>0</MaxLayersToTrackLikeHit>
            <TrackPathWidth>0</TrackPathWidth>
        </algorithm>
        <algorithm type = "ConeClusteringFast" instance = “Reclustering2"/>
        <algorithm type = "ConeClusteringFast" instance = "Reclustering3"/>
        <algorithm type = "ConeClusteringFast" instance = "Reclustering4"/>
        <algorithm type = "ConeClusteringFast" instance = "Reclustering5"/>
        <algorithm type = "ConeClusteringFast" instance = "Reclustering6"/>
        <algorithm type = "ConeClusteringFast" instance = "Reclustering7"/>
        <algorithm type = "ConeClusteringFast" instance = "Reclustering8"/>
        <algorithm type = "ConeClusteringFast" instance = "Reclustering9"/>
        <algorithm type = "ConeClusteringFast" instance = "Reclustering10"/>
        <algorithm type = "ConeClusteringFast" instance = "Reclustering11"/>
        <algorithm type = "ConeClusteringFast" instance = "Reclustering12"/>
    </clusteringAlgorithms>
    <algorithm type = "TopologicalAssociationParent" description = "ClusterAssociation" instance = "reclusterAssociation"/>
    <algorithm type = "TrackClusterAssociationFast" description = "TrackClusterAssociation"/>
    <algorithm type = "ForcedClustering" description = “ForcedClustering"/>
    <UsingOrderedAlgorithms>true</UsingOrderedAlgorithms>
    <ShouldUseForcedClustering>true</ShouldUseForcedClustering>
</algorithm>
```

Alternative clustering daughter algorithm (full config)

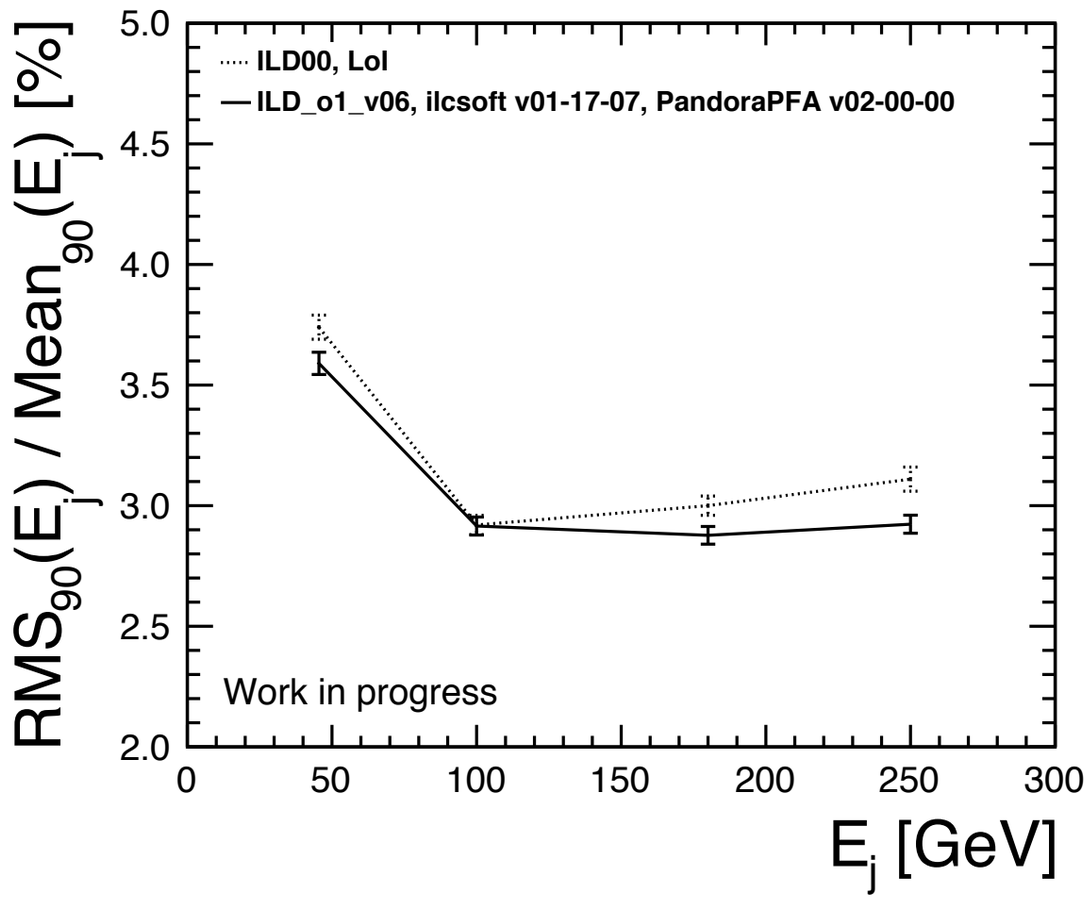
Alternative clustering daughter algorithms (reuse existing instances)

Daughter algorithms providing specific functionality to parent

Steering options for parent algorithm

Pandora Performance

- Pandora particle flow performance is typically characterised by the **Jet Energy Resolution**, measured using Z' particles, which decay into light quarks, producing two mono-energetic jets.
- Performance depends on pattern recognition *and* the estimators used to obtain energy measurements for EM and hadronic showers. The calibration procedure is very important.
- Performance plots shown here use same calibration approach used for ILD LoI. Results show ***consistency*** with and ***progress*** with respect to the LoI results, with clear high E improvement.

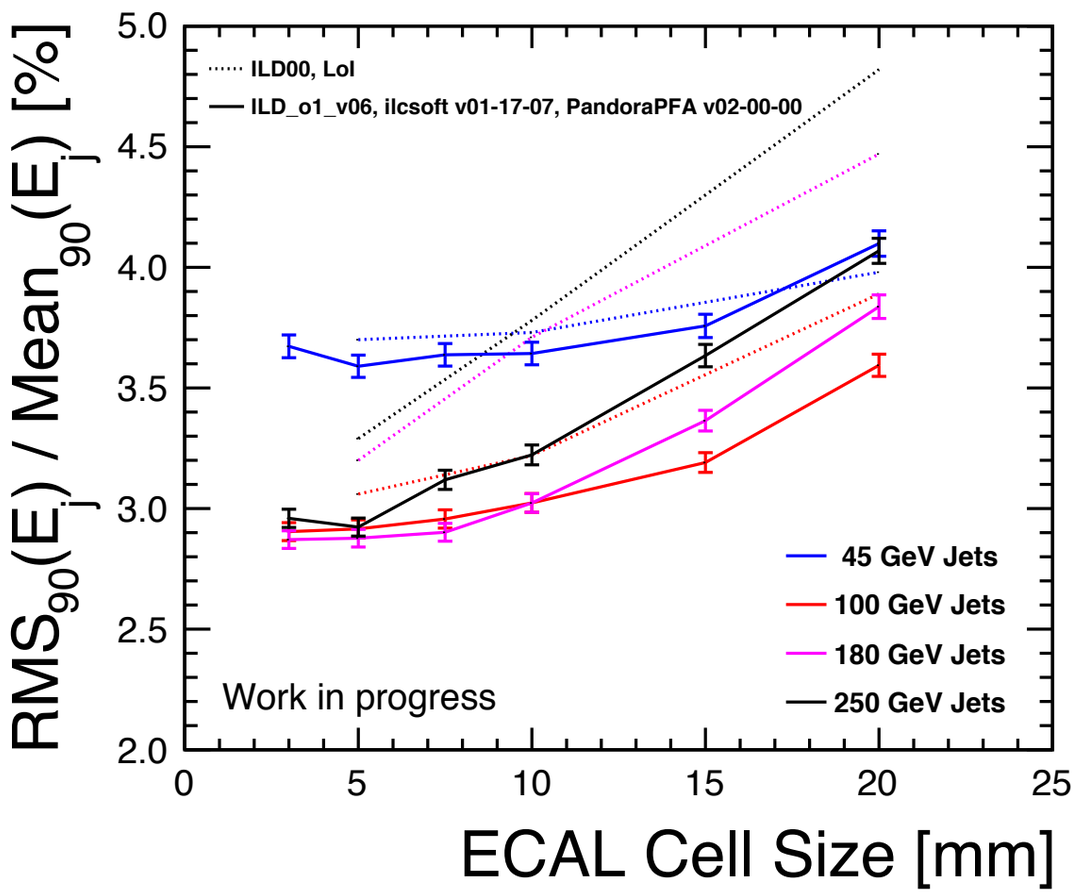


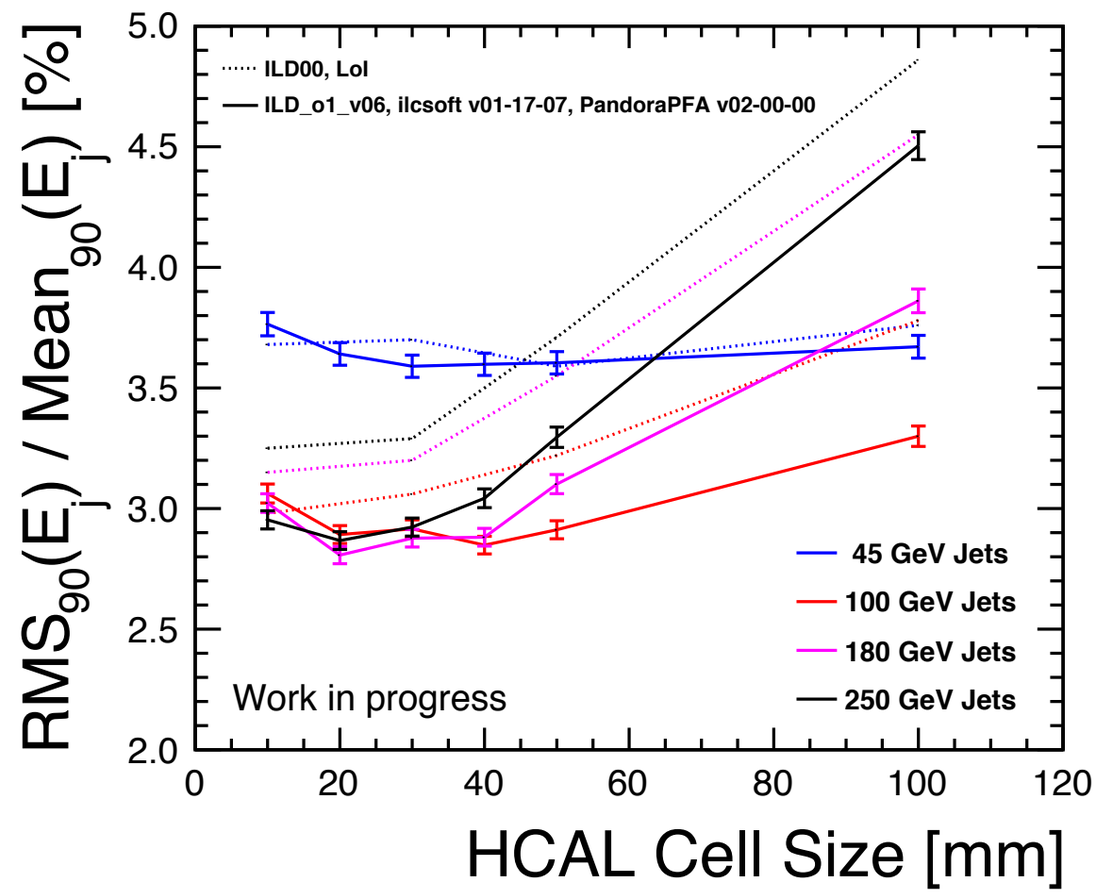
Event type	200 GeV Z'	200 GeV Z'
Background	none	CLIC 3 TeV
Event Time	0.22 ± 0.02 s	12.5 ± 0.4 s
VMem	256 MB	480 MB
RSS	43 MB	266 MB
# Tracks	27 ± 1	650 ± 7
# CaloHits	$4,200 \pm 60$	$39,900 \pm 700$

Table 1: Indicative CPU times and memory footprints for processing 200 GeV Z' events in the CLIC detector, with and without overlaid backgrounds. The mean numbers of Tracks and CaloHits indicate the complexities of the events. The memory footprint is broken down into a virtual memory value and a resident set size value. The event times were recorded using a single socket, quad core, unthreaded Core i5-3570 CPU (clock speed 3.4GHz, SPECint2006 48.5, SpecFP206 62.9).

Pandora Performance

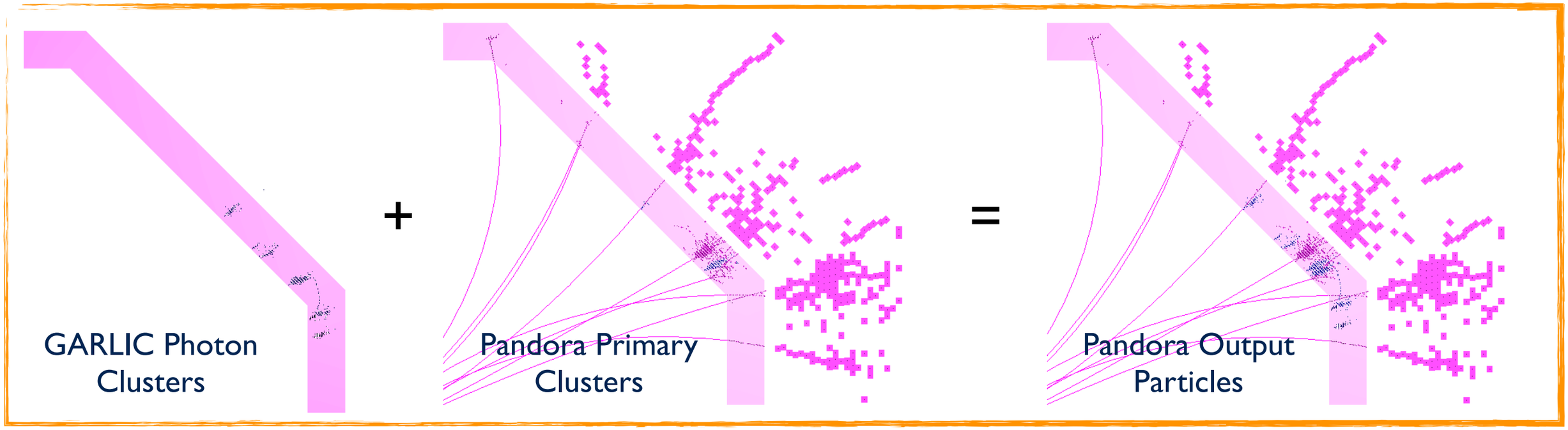
- Detector optimisation plots here compare LoI results with those obtained using ilcsoft v01-17-07 (PandoraPFA v02-00-00). The LoI calibration *approach* was applied (DBD constants).
- The optimum calibration approach is still under investigation, so these plots should not yet be used to draw any conclusions about the most appropriate detector parameters.
- **Take-home message:** PandoraPFA v02-00-00 offers our best ever jet energy resolutions, but users should appreciate the importance of the energy calibration for each detector model.





Pandora External Clusters

- It is possible to import external Cluster collections (e.g. from GARLIC) directly into Pandora. The **ExternalClusteringAlgorithm**, built with MarlinPandora, understands lcio and Pandora.
- The external Clusters are recreated as Pandora Clusters and can then be included, or modified, as required in the Pandora output e.g. can replace the standard Pandora photon Clusters.
- Neither Daniel nor I are sure that GARLIC will aid performance, especially given recent improvements from Bono, but we welcome your feedback (for other external Clusters too).




```
<algorithm type = "ClusteringParent" >
  <algorithm type = "ExternalClustering" description = "ClusterFormation">
        <ExternalClusterCollectionName>GARLICPhotonClustersTightSel</ExternalClusterCollectionName>
        <FlagClustersAsPhotons>true</FlagClustersAsPhotons>
  </algorithm>
  <ClusterListName>PhotonClusters</ClusterListName>
  <ReplaceCurrentClusterList>false</ReplaceCurrentClusterList>
</algorithm>
```

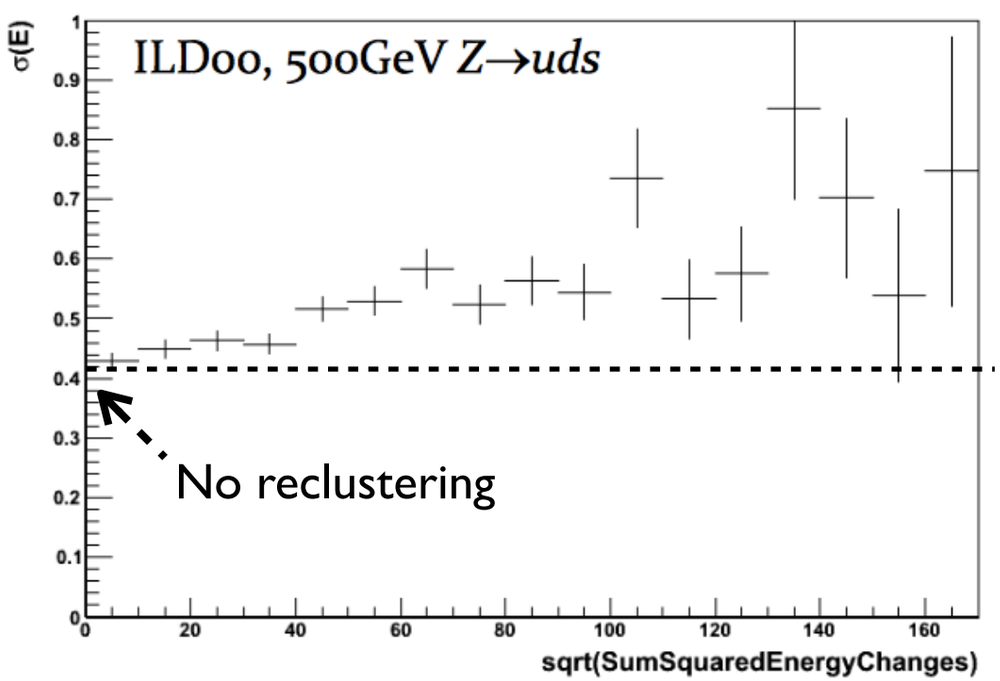
Input lcio collection name

Output Pandora list name

Pandora Uncertainties

- In this workshop, would like to follow-up Jenny's talk regarding the provision of uncertainties by the Pandora reconstruction. Hope to prompt discussion leading to some definitions.
- Neutral hadron fraction in jets varies considerably and jets with a smaller neutral hadron fraction are better measured. Would like to propagate this information for Physics analysis.
- Estimate cluster uncertainties purely from calorimetric measurement. Use track uncertainties for charged PFOs and cluster uncertainties for neutral PFOs. How/where to address confusion?

$$\sigma_{jet} = f_{charged}\ \sigma_{track}\ \text{“+”}\ f_{photon}\ \sigma_{ECal}\ \text{“+”}\ f_{neut.had.}\ \sigma_{HCal}\ \text{“+”}\ \sigma_{confusion}$$



- **Jets are obvious place to add confusion.** Tentative past effort to monitor work done by reclustering.
- For each Track, recorded changes in associated Cluster energy during reclustering processes.
- Stored net energy change, sum of moduli of changes and sum of squared energy changes.
- Some sensitivity, but this code unsupported since v00-16-00 and needs to be reimplemented.

Pandora in ilcsoft v01-17-07

- PandoraPFA v02-00-00: ← Metadata/build package, collects consistent tags
  - Pandora SDK v02-00-00 ← Pandora framework
  - Pandora Monitoring v02-00-00 ← ROOT EVE visualisation, plus tree writing
  - Pandora LCContent v02-00-00 ← Algorithms for use at ILC or CLIC
- MarlinPandora v02-00-00 ← Client App in Marlin framework
- PandoraAnalysis v01-00-01 ← Marlin processors for analysis and calibration

- **It is recommended that you use the DESY ilcsoft v01-17-07 cvmfs or afs installations.**
- The gcc48 installations include C++11-specific functionality, including fast versions of some key algorithms (see LCContentFast.h), which use KD-trees and new unordered containers.
- Recommended for use if there is significant beam-related background in events (x15-20 speed-up). Just append "Fast" to relevant algorithm types in PandoraSettings XML file.