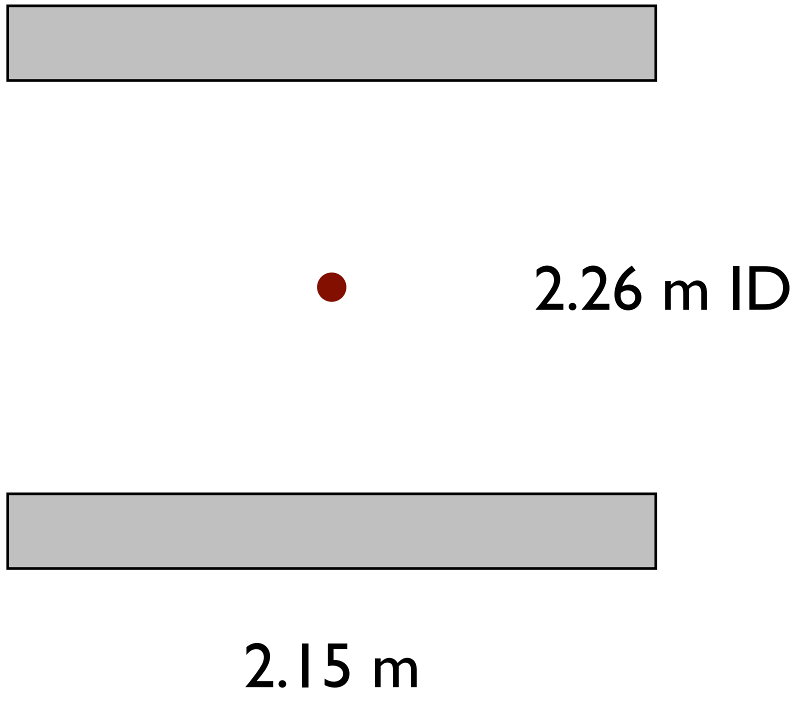
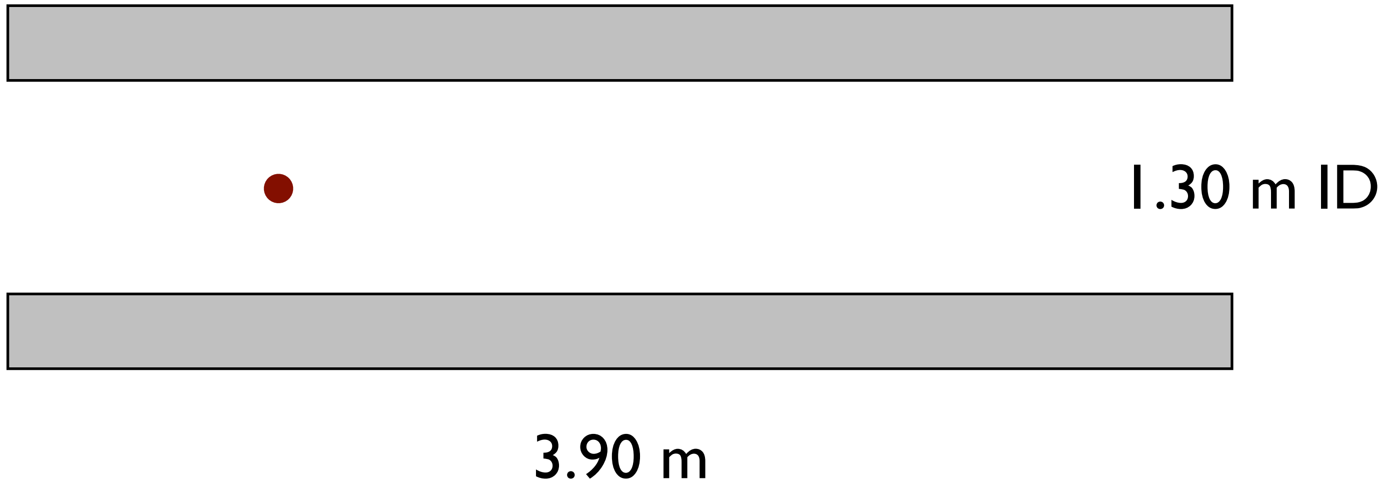
Motivation for New BCAL Software

- No GlueX collaborator completely understands the inner workings of the KLOE reconstruction code; this makes it hard to
  - incorporate new features (e.g., single ended readout)
  - optimizing functionality
  - provide long term maintenance
- Physics motivation is different:
  - KLOE really needed to vertex photons for $K_s \rightarrow \pi^0\pi^0$ -- an impressive feat, but unnecessary for GlueX
  - Eliminating unnecessary requirements streamlines code, makes it easy to maintain, and may improve reconstruction
- Shower properties in the z direction are drastically different between the two calorimeters
  - KLOE algorithm is likely not optimal for GlueX -- can it be tuned? (see the first major bullet)
- Apparent problem of having a large number of hadronic splitoffs in BCAL and no good idea of how to control them

KLOE and GlueX Comparisons





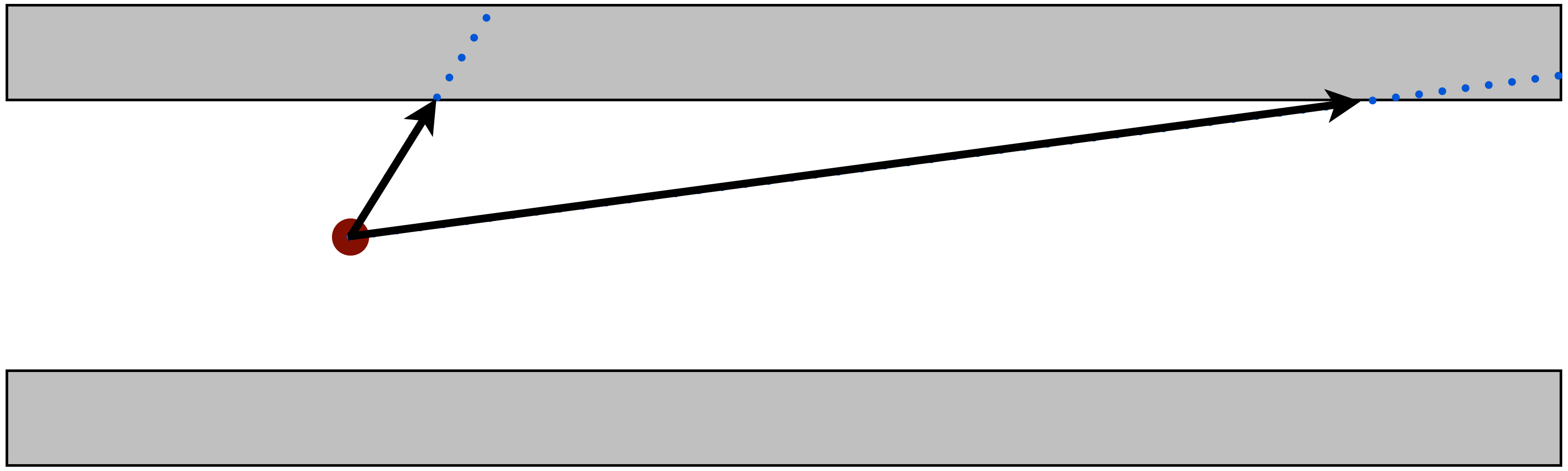
Maximum incident angle:
44 degrees

colliding beam qq production
typically distributed like $\sin^2\theta$

Maximum incident angle:
79 degrees

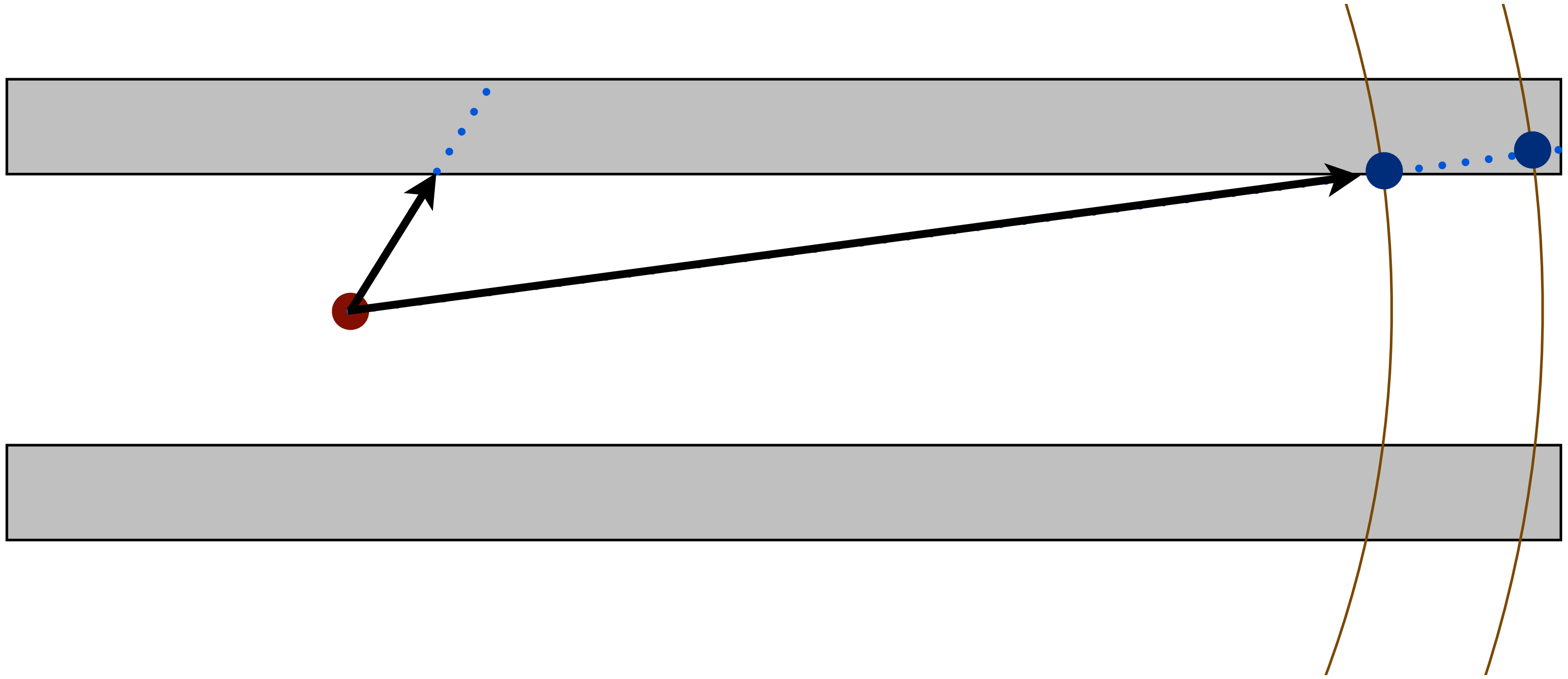
fixed target production strongly
populates high incident angle

KLOE Clustering in z Dimension



- By the name of the variable, the KLOE code uses the RMS of the individual cells in the shower to determine if a shower should be split in the z dimension
  - BREAK_THRESH_TRMS = 5.0
  - single parameter: can't be optimal for both of the showers pictured above

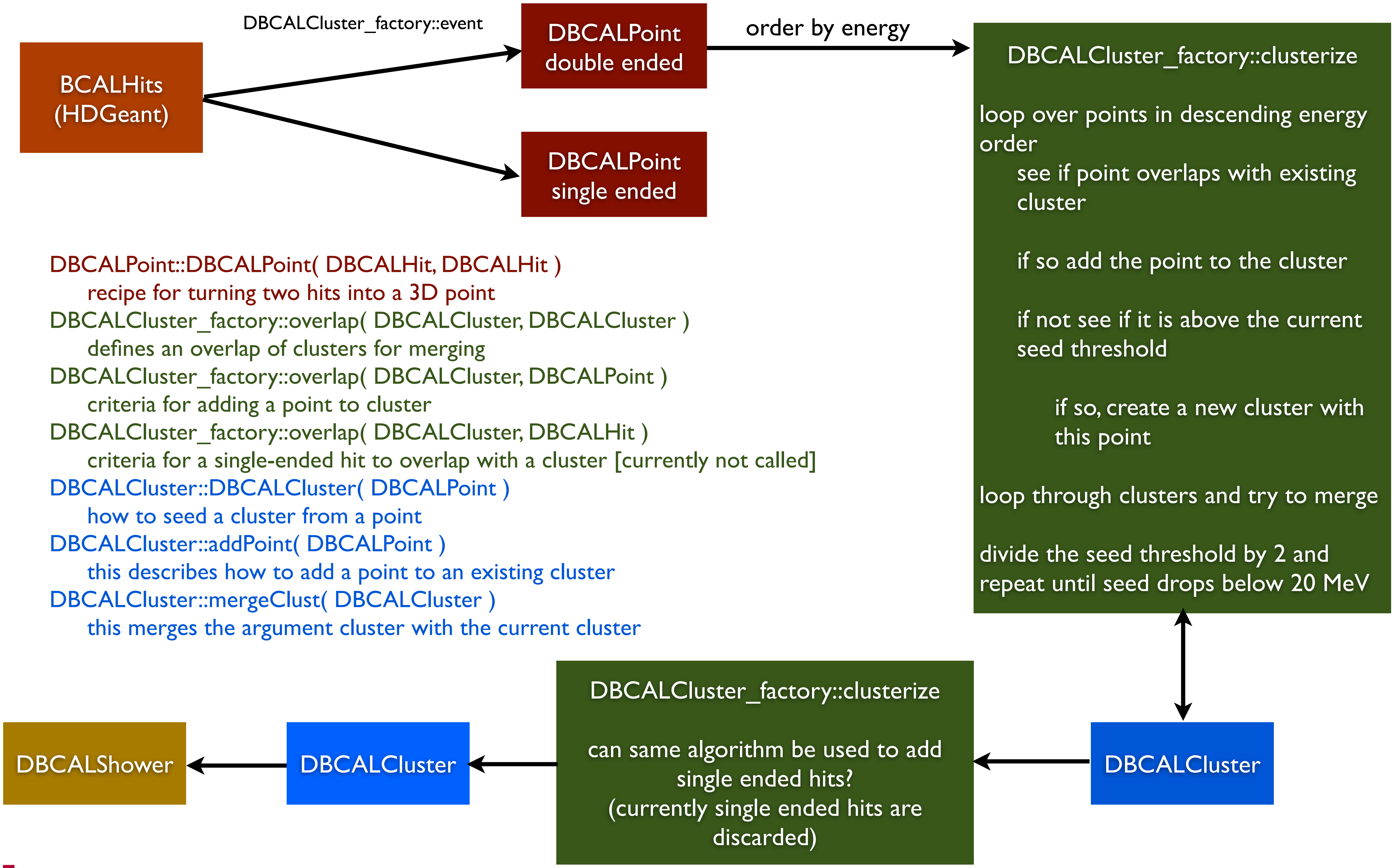
Revised Clustering Approach



This also illustrates why simple merging by distance is not effective. One needs to project to common radius sphere then merge or just consider spanned solid angle.

- Transform cell hits and errors to spherical coordinates (with errors)
- Do clusterizing by examining for overlaps in Φ and θ
- Compare differences in Φ and θ with respect to errors, i.e. “size”, on a cell-by-cell basis (to start cluster) and a cell-by-cluster basis to add cells to the cluster
    - Simple algorithm to check for overlaps -- no “knowledge” about typical EM shower shapes or sizes is currently used

Code Flow Chart



BCALHits (HDGeant)

DBCALCluster_factory::event

DBCALPoint double ended

DBCALPoint single ended

order by energy

DBCALCluster_factory::clusterize

```
loop over points in descending energy order
    see if point overlaps with existing cluster

    if so add the point to the cluster

    if not see if it is above the current seed threshold

        if so, create a new cluster with this point

loop through clusters and try to merge

divide the seed threshold by 2 and repeat until seed drops below 20 MeV
```

DBCALCluster

DBCALCluster_factory::clusterize

can same algorithm be used to add single ended hits?
(currently single ended hits are discarded)

DBCALCluster

DBCALShower

DBCALPoint::DBCALPoint(DBCALHit, DBCALHit)
    recipe for turning two hits into a 3D point

DBCALCluster_factory::overlap(DBCALCluster, DBCALCluster)
    defines an overlap of clusters for merging

DBCALCluster_factory::overlap(DBCALCluster, DBCALPoint)
    criteria for adding a point to cluster

DBCALCluster_factory::overlap(DBCALCluster, DBCALHit)
    criteria for a single-ended hit to overlap with a cluster [currently not called]

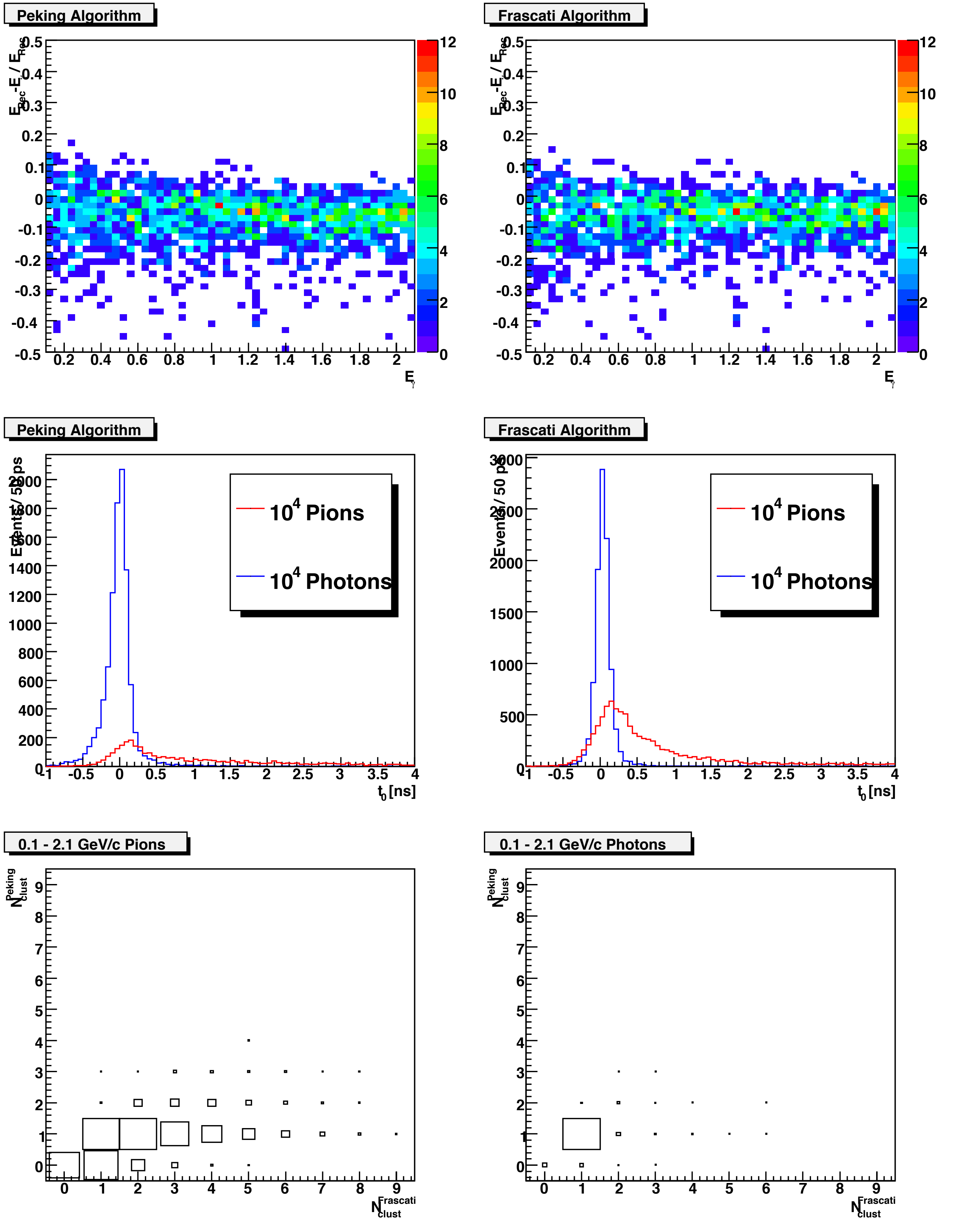
DBCALCluster::DBCALCluster(DBCALPoint)
    how to seed a cluster from a point

DBCALCluster::addPoint(DBCALPoint)
    this describes how to add a point to an existing cluster

DBCALCluster::mergeClust(DBCALCluster)
    this merges the argument cluster with the current cluster

First Plots

- Test with particle gun: pions and photons; averaged over BCAL; 0.1-2.1 GeV
- New algorithm (left column) seems to do something similar to the KLOE algorithm
- Second row shows worse timing resolution -- need to study how time information from points is propagated to cluster
- Bottom left plot shows dramatic reduction in the number of hadron showers
- These plots were made 15-Mar-11 -- last bit of progress from me



Summary and Next Steps

- Assuming the general algorithm is valid, then the structure of a new BCAL reconstruction routine exists in the repository... and it creates clusters
- Need to define some metrics for optimization:
    - single photon efficiency
    - number of showers per hadron
    - photon energy resolution
    - photon timing resolution
- Generate data samples for optimization
    - single photon events
    - single pion events
    - signal MC for a complex event (e.g., $b_1\pi$)
- Systematically fine tune individual methods:
    - How to determine cluster energy, position, and time from individual points?
    - How to merge points and clusters?