

Cluster/PFA interface thoughts

Clustering

PFA framework conventions

- + Implement as modular Driver
- + Use HitMaps
- + Leave original collections unmodified -- write out a new List/HitMap instead

New clusterers should follow Ron's interface

- + `org.lcsim.recon.cluster.util.Clusterer`

This is more or less stable now.

PFAs

For PFA code (as distinct from clustering), use
`org.lcsim.recon.pfa`

Few basic things in there already:

`org.lcsim.recon.pfa.cheat.PerfectIdentifier`

`org.lcsim.recon.pfa.identifier.SimpleChargedParticleMaker`

`org.lcsim.recon.pfa.identifier.SimpleNeutralParticleMaker`

`org.lcsim.recon.pfa.output.ConfusionPlotter`

`org.lcsim.recon.pfa.output.EnergySumPlotter`

`org.lcsim.recon.pfa.output.CorrectedEnergySumPlotter`

`org.lcsim.recon.pfa.structural.HitBookKeeper`

`contrib.uiowa.template.NonTrivialPFA`

PFAs

For PFA code (as distinct from clustering), use
`org.lcsim.recon.pfa`

Few basic things in there already:

`org.lcsim.recon.pfa.cheat.PerfectIdentifier`

`org.lcsim.recon.pfa.identifier.SimpleChargedParticleMaker`

`org.lcsim.recon.pfa.identifier.SimpleNeutralParticleMaker`

`org.lcsim.recon.pfa.output.ConfusionPlotter`

`org.lcsim.recon.pfa.output.EnergySumPlotter`

`org.lcsim.recon.pfa.output.CorrectedEnergySumPlotter`

`org.lcsim.recon.pfa.structural.HitBookKeeper`

`contrib.uiowa.template.NonTrivialPFA`

**Identifier: Make ReconstructedParticles
out of clusters, tracks, etc.**

PFAs

For PFA code (as distinct from clustering), use
`org.lcsim.recon.pfa`

Few basic things in there already:

`org.lcsim.recon.pfa.cheat.PerfectIdentifier`

`org.lcsim.recon.pfa.identifier.SimpleChargedParticleMaker`

`org.lcsim.recon.pfa.identifier.SimpleNeutralParticleMaker`

`org.lcsim.recon.pfa.output.ConfusionPlotter`

`org.lcsim.recon.pfa.output.EnergySumPlotter`

`org.lcsim.recon.pfa.output.CorrectedEnergySumPlotter`

`org.lcsim.recon.pfa.structural.HitBookKeeper`

`contrib.uiowa.template.NonTrivialPFA`

Common tools for performance plots etc. go in “output”

PFA_s

For PFA code (as distinct from clustering), use
`org.lcsim.recon.pfa`

Few basic things in there already:

`org.lcsim.recon.pfa.cheat.PerfectIdentifier`

`org.lcsim.recon.pfa.identifier.SimpleChargedParticleMaker`

`org.lcsim.recon.pfa.identifier.SimpleNeutralParticleMaker`

`org.lcsim.recon.pfa.output.ConfusionPlotter`

`org.lcsim.recon.pfa.output.EnergySumPlotter`

`org.lcsim.recon.pfa.output.CorrectedEnergySumPlotter`

`org.lcsim.recon.pfa.structural.HitBookKeeper`

`contrib.uiowa.template.NonTrivialPFA`

Implementation-specific code goes in its own sub-package (e.g. the Iowa “structural” algorithm)

Identifiers

Very naive implementations at the moment

SimpleNeutralParticleIdentifier

- + Takes in a list of clusters
- + Makes a neutral ReconstructedParticle from each
- + Global setting: PDG code to assume for calculating energy.
- + Calibration hard-wired for now (ugly linear thing)

I would prefer that the calibration be a PFA-supplied object with a standard interface. (Ron's as default?)

Identifiers

Very naive implementations at the moment

SimpleChargedParticleIdentifier

- + Takes in lists of clusters, MIPs, and tracks
- + Tries to match tracks to clusters
- + Takes momentum from track(s)
- + Takes mass from track truth info (lazy PID)
- + Builds a ReconstructedParticle for each cluster that matched track(s).
- + No E/p sanity check

I am currently refactoring this to

- 1) Put the track extrapolation code in its own class
- 2) Not require MIPs to also be in the cluster list

What else?