ClusterID Tutorial

Based on code developed by
Gary Bower, SLAC
Ron Cassell, SLAC
Ayanah George, SLAC summer student
Ayodele Onibokun, SLAC summer student
Suarav Pathak, U Penn

All the instructions will be given for Windows. Linux folk, please forgive us and do what you always do in this situation and translate to your own frame of reference

To install the big .zip file right click and choose open and when WinZip comes up extract to C:\, the top level of the C disk. You should find a directory called Tutorial on the top level of the C disk with two folders inside and files inside of those as shown on the web page.

The jas.jar file should also appear on the top level of your C disk. Move the jas.jar file to Program Files/Java Analysis Studio/lib to replace the version that came with the JAS installation.

Now startup JAS by going the Start menu> Java Analysis Studio > the debug version. The debug version just means that it starts up with a visible command console so we will later be able to see features of the neural net in operation.

Now go to View > Plugin Manager > Install a plugin and select “lcd” from the list and install it.

Then kill JAS and restart again in debug mode. Ignore the console message about unable to load Register plugin.

Set class path C:\Tutorial\ClusterID

Set class path C:\Tutorial\ClusterAnalysis

ClusterID - Overview

ClusterID is an approach to reconstructing calorimeter data where reconstruct means beginning with calorimeter cell hits then creating clusters of hits with a cluster builder and then identifying the type of particle that created the cluster and finally producing a set of reconstructed particles.

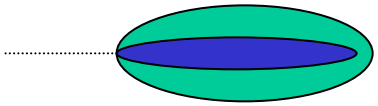
ClusterID: A Modular Framework

Cluster builders can be interchanged and the technique for identifying particles can be varied. However, in this tutorial demonstration we will generally use the same cluster builder and the same ID algorithm.

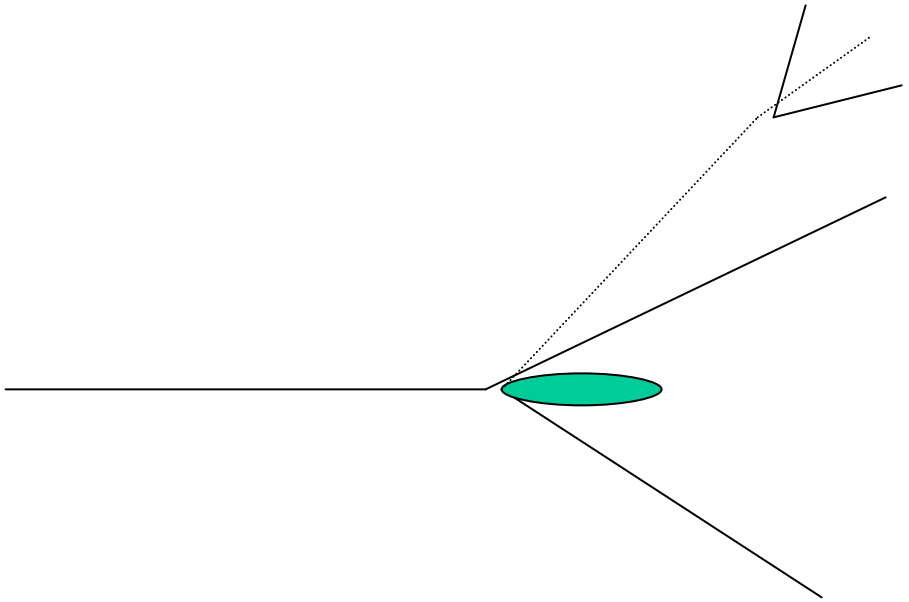
General method to identify cluster producing particle type.

The key concept of ClusterID is that each particle type produces clusters with properties, called discriminators, that differ from the clusters produced by another type of particle. For each cluster a set of discriminants are measured and cuts or a neural net are used to identify cluster types.

Gamma shower cartoon



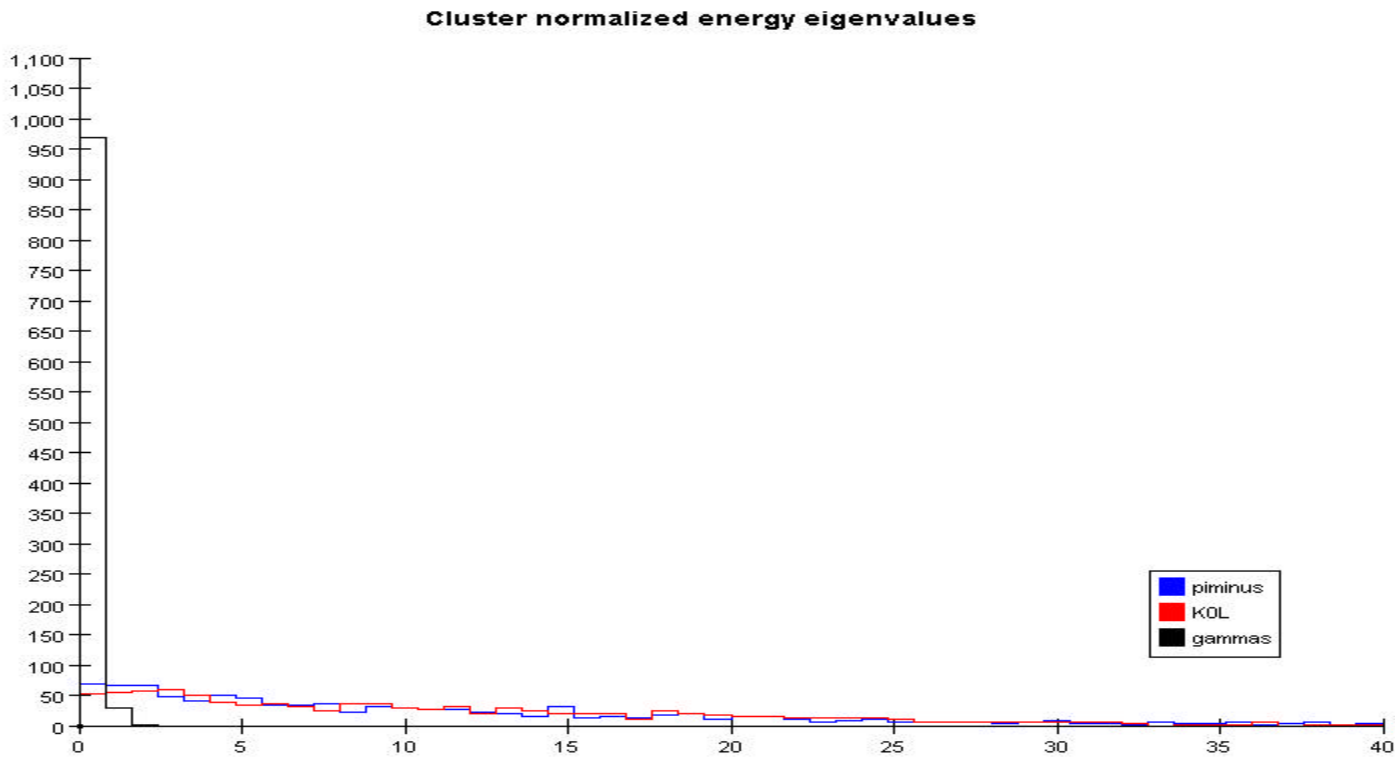
Charged hadron shower cartoon



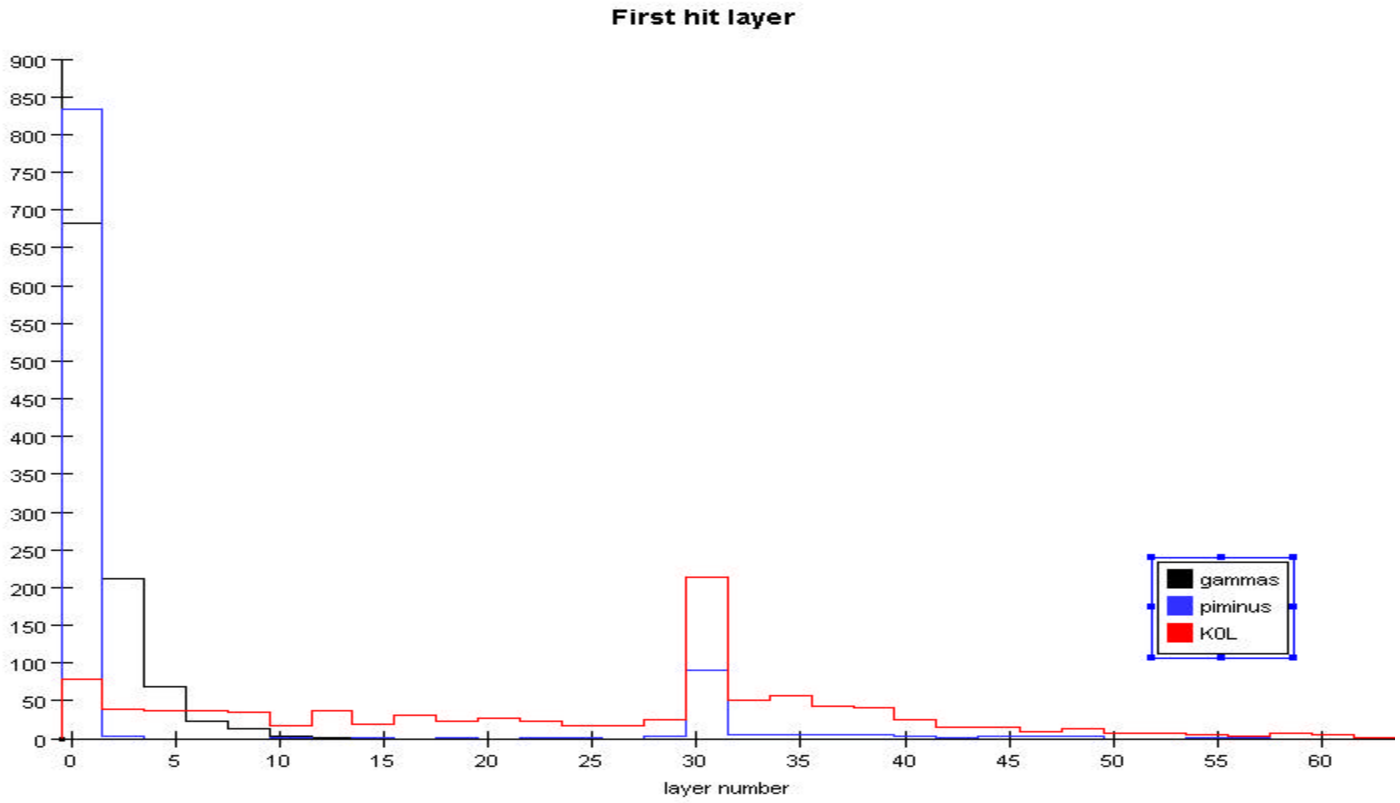
15 Discriminators

- 3 normalized energy tensor eigenvalues, ne1,ne2,ne3.
- ne1/ne2, ne2/ne3.
- First layer hit, last layer hit, length of cluster, (firstL+1)/length.
- Angular separation between e1-axis and IP.
- Energy in first 5 layers.
- Nhits in first 2 layers.
- z-coordinate of center of energy.
- Nhits
- Measured cluster energy.

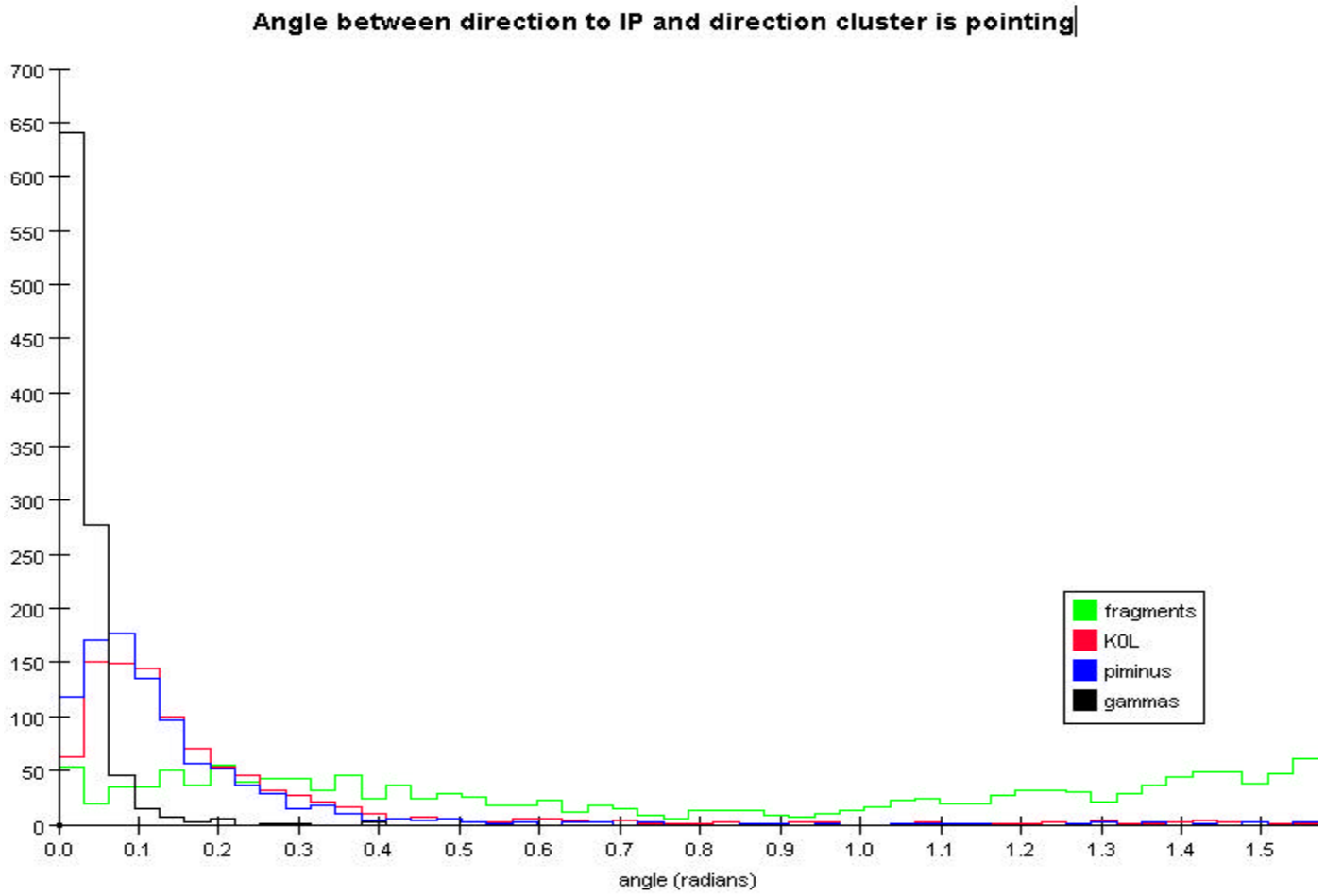
Separating gammas



Separating piminuses

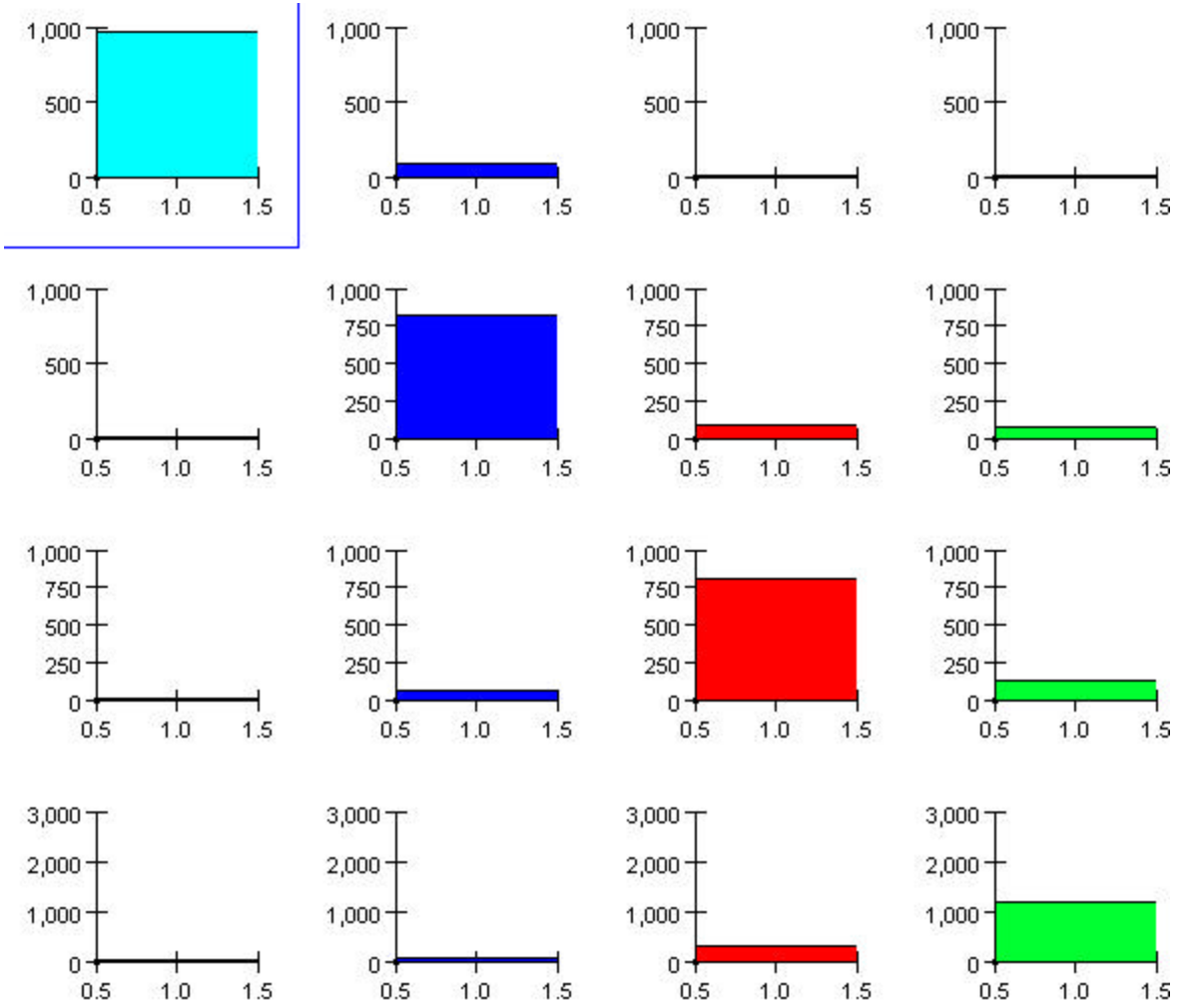


Separating K0Ls



ID result:

Input:



Testing tools -1

Since ClusterID allows various cluster builders and the use of various discriminators, it is essential to have tools that can measure the efficacy of the different algorithms. ClusterID includes a set of tools for making these comparisons.

Testing tools -2

As we will show later in the tutorial these testing tools are an essential feature of ClusterID because they allow a detailed analysis of cases where improvement is possible. ClusterID has many areas where improvements are possible with additional study.

ClusterID - Implementation

ClusterID is implemented as an extension to JAS/LCD. JAS/LCD is based on the event processor/event loop concept used in SLD. An object called an event is extended by a series of independent modular processors.

Event loop - 1

Each process in the event loop in sequence adds information to the event object. Beginning with a random number as the sole element of the event an event generator creates a set of particles according to a chosen physics process.

Event loop - 2

The event is passed to a detector simulator which adds hits to the event. From there it goes to a multistage reconstruction process. For calorimeter data building clusters is a first stage, followed by identifying the cluster's origin, followed by a user analysis using the reconstructed particles.

Learning through pictures

- We begin by illustrating the concepts of ClusterID with some pictures.

Using a neural net

- Using a neural net involves three stages:

1) Making a training data set.
2) Training the net.
3) Applying the net.

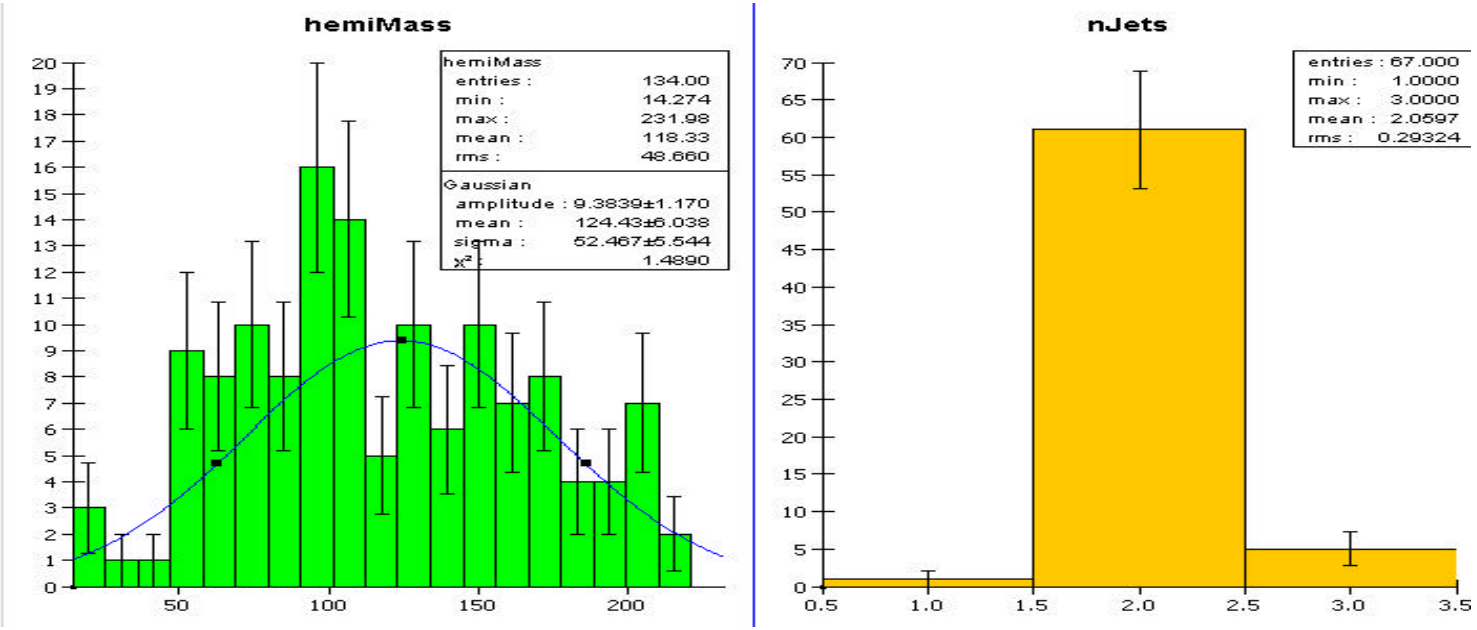
Implementation tour overview

- We begin by showing results from applying a previously trained net.
- Then we will demonstrate how to use ClusterID to make training data sets and to train a net.

Important notice

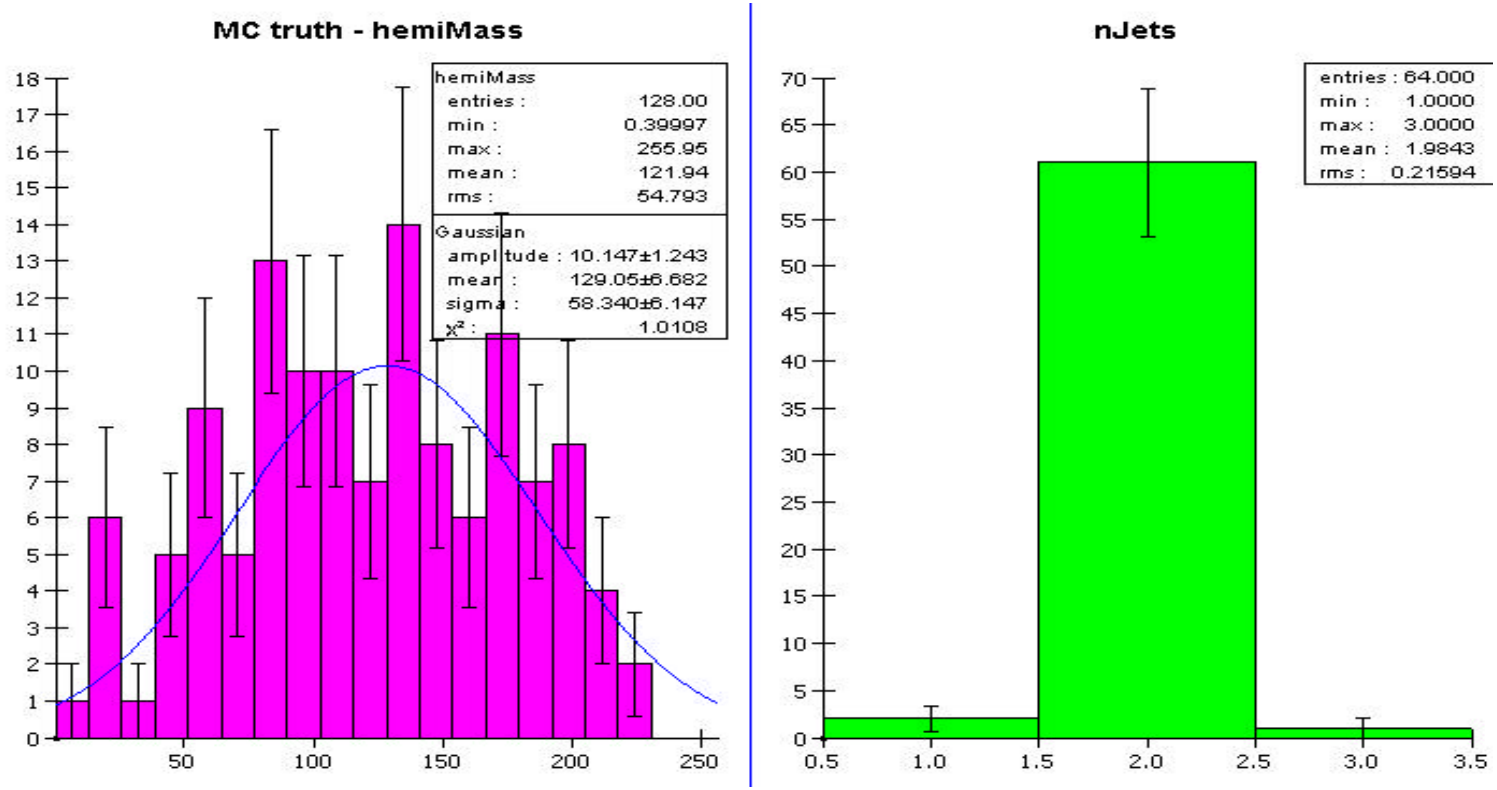
- ClusterID is fully functional, all the essential elements are available for use.
- However, there is much that can be done to improve the resolution currently available by further developing the discriminants used, attaching fragments, finding pi-zeros, etc. (more on this later)

Ttbar event mass plots



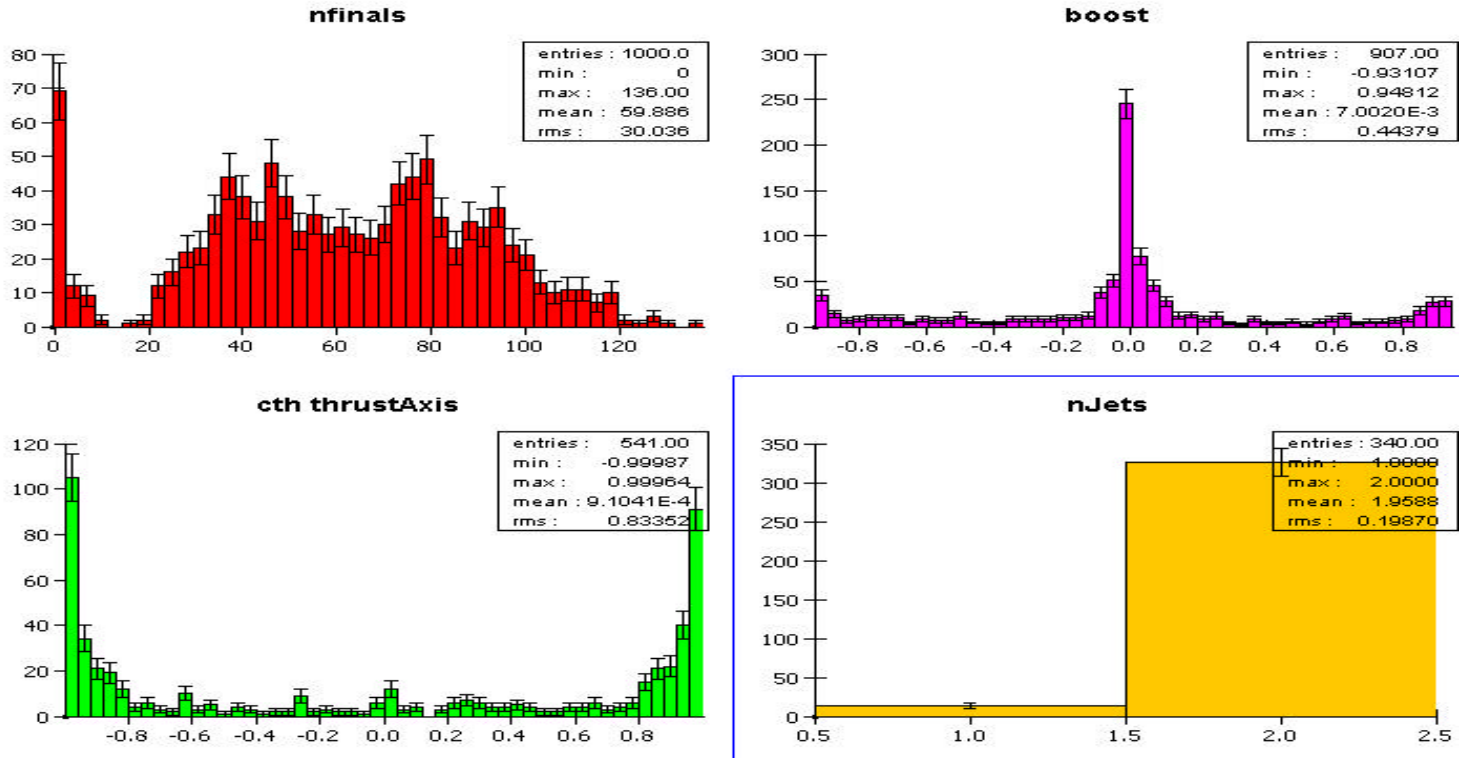
Some work to be done on t jet finding!

Same problem in MC Truth!



Good news: notice MC and recon give very similar mean and sigma.

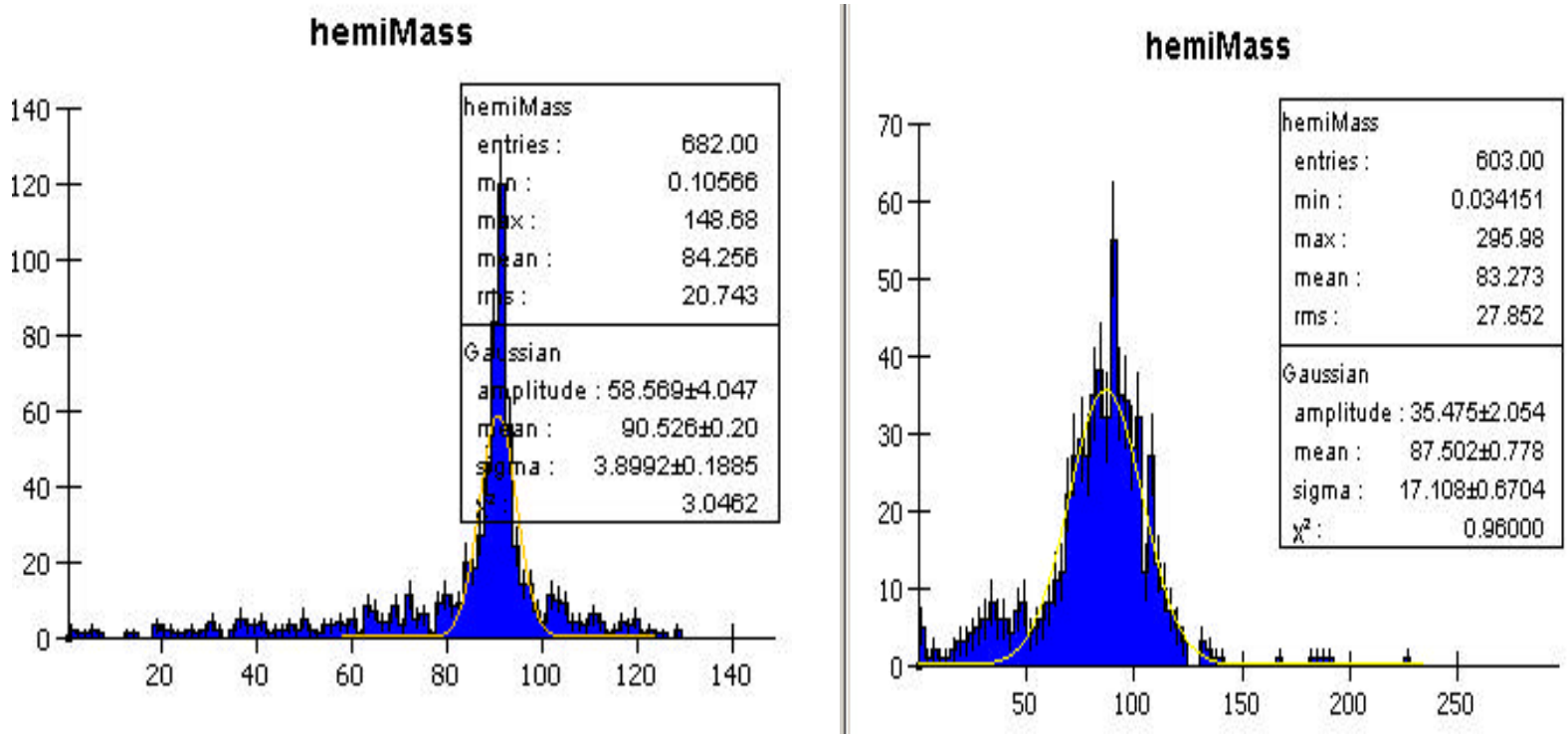
Jet finding is better in ZZ events



But first, some cuts are required:

boost<0.2, cth thrust axis <0.5, #particles>20

Z mass in ZZ events



Left: MCTruth, no neutrinos.

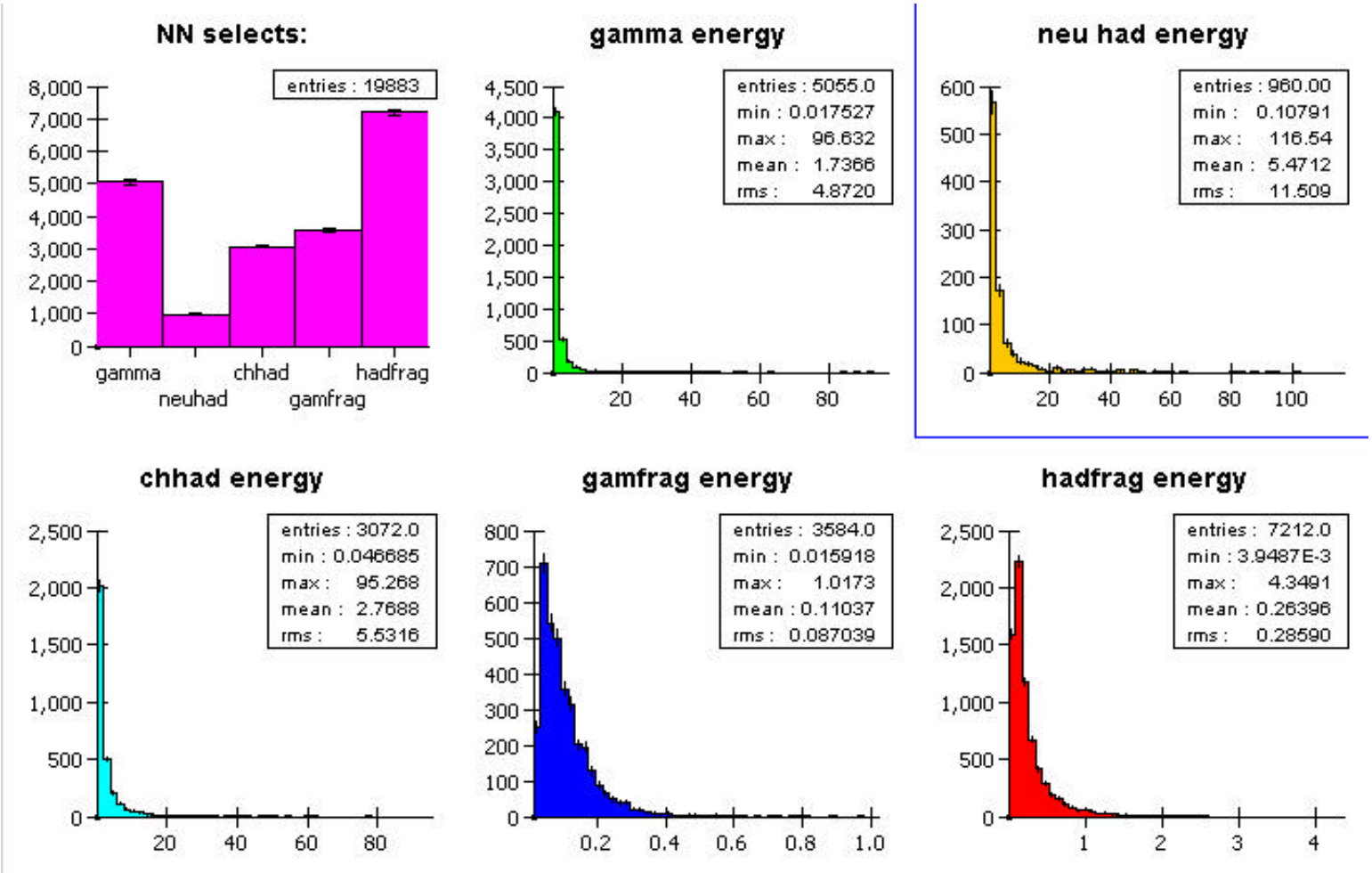
Right: ClusterID neutrals with MCTruth charged.

Z mass in ZZ events

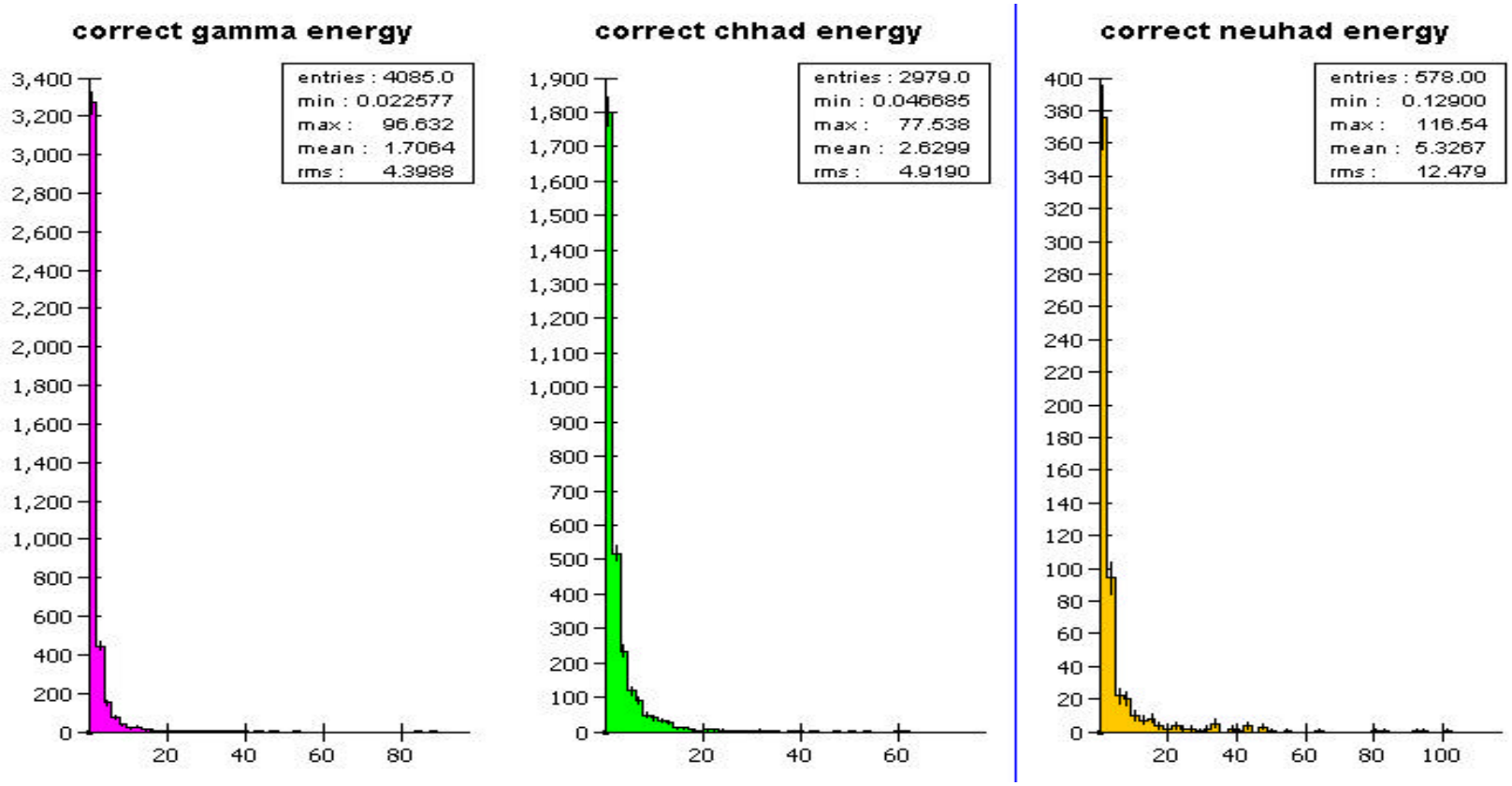
Z mass(fit)	**Z width(fit)**	**Particles Included**
90.5	3.9	All MC except neutrinos
87.5	17.1	MC charged, ClusterID(gamma, neuhad)
51.5	17.0	MC charged, ClusterID(none)
108.5	17.2	MC charged, ClusterID(all)
68.8	21.0	MC charged, ClusterID(ch had, frags)
71.6	21.3	MC charged, ClusterID(gamma*)
63.8	19.1	MC charged, ClusterID(neu had*)
85.0	16.7	FastMC charged, ClusterID(gamma, neuhad)

* Most high energy photons misIDed as neutral hadrons

ttbar events



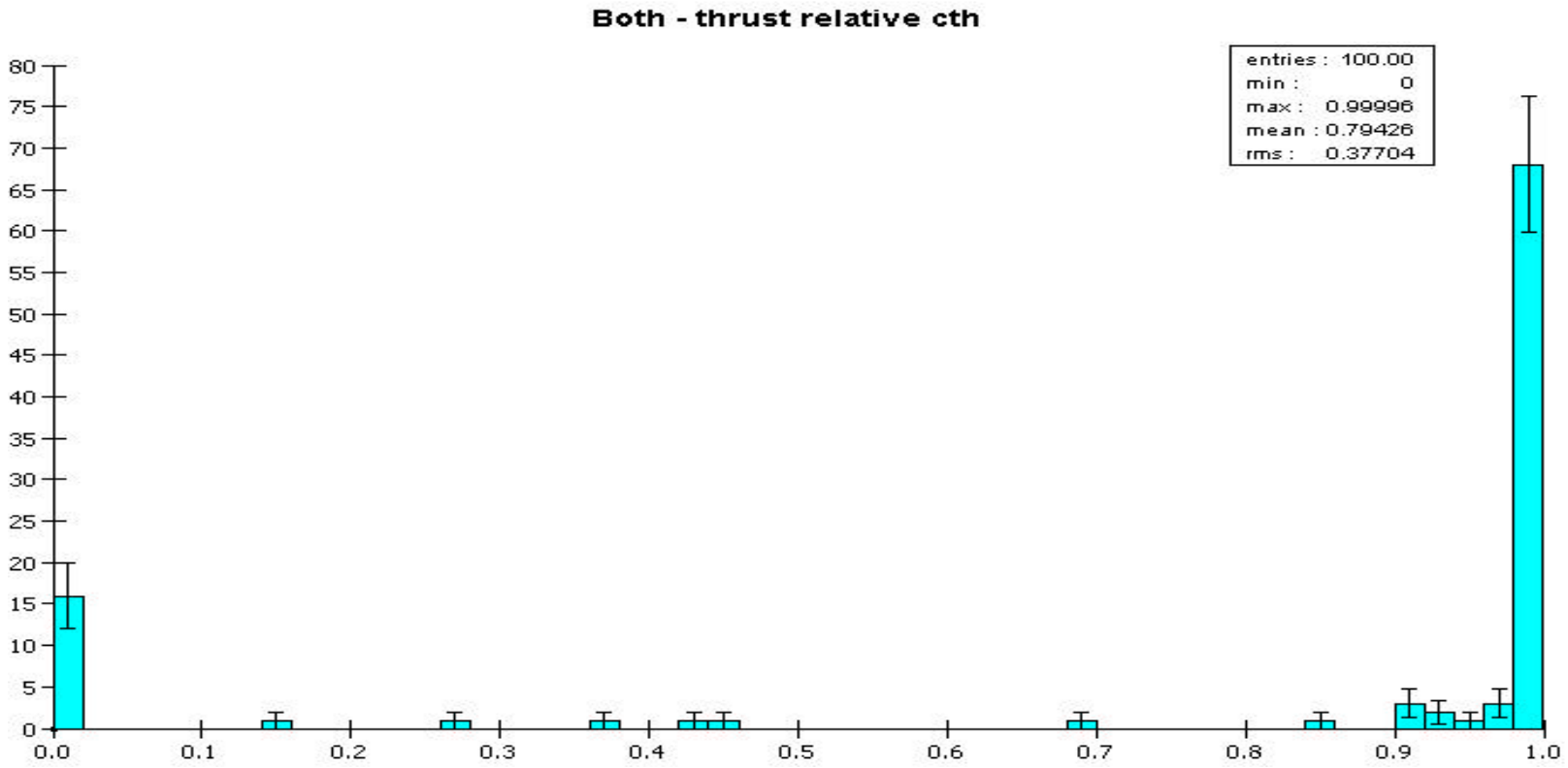
ttbar events



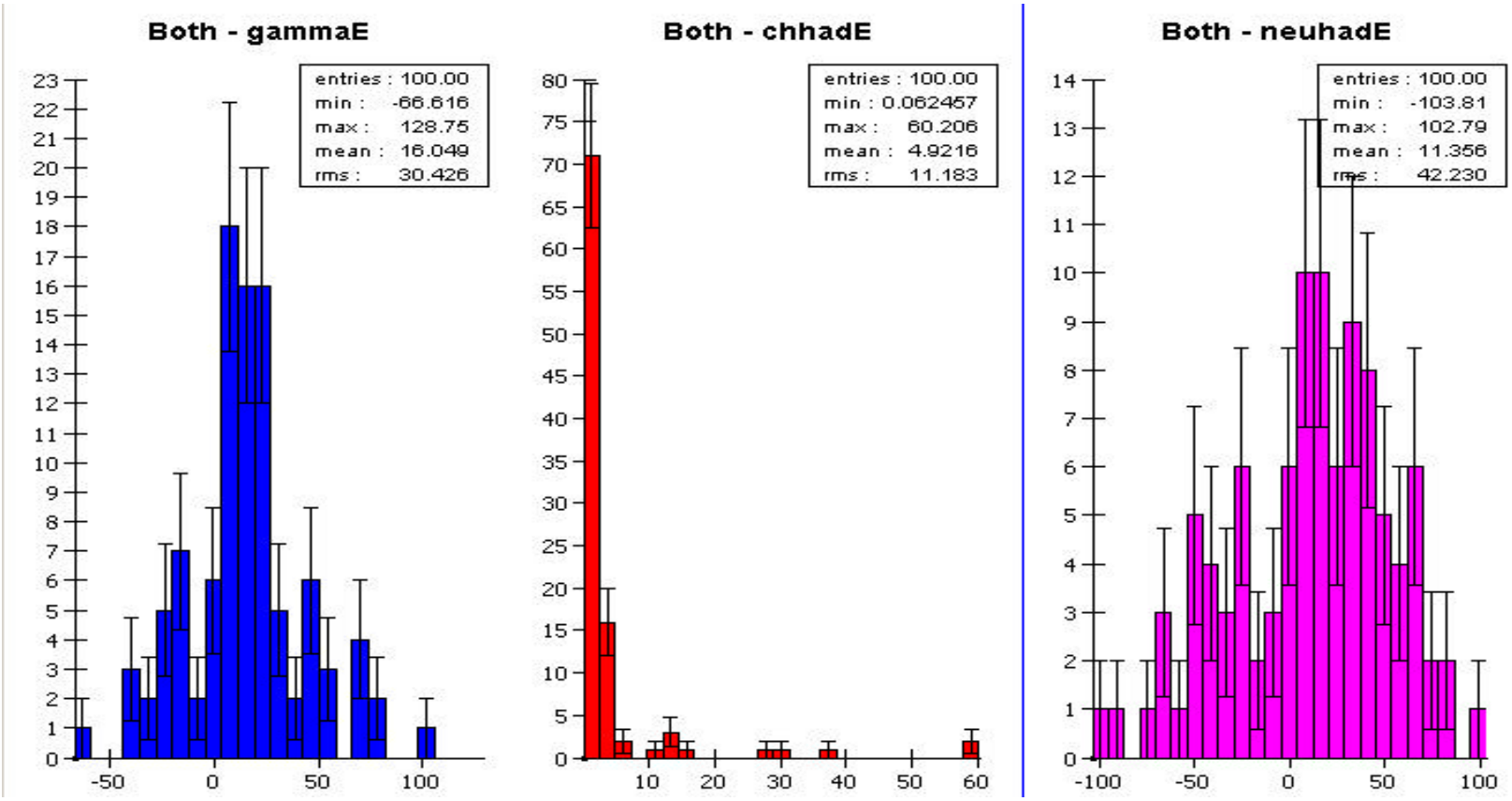
tt bar events – from cluster ID

Cluster type	Ave energy per event	Ave true energy per event
gamma	88	70
neutral hadron	53	31
Charged hadron	85	78
Gamma fragment	4	
Hadron fragment	46	
Total	276	

MC vs Full Recon thrust axis



ttbar events: MCEn - ReconEn



Stage 3 of the future

```
add( new MCFast() );
add( new SimpleClusterBuilder() );
add( new ApplyClusterIDToSioFile(net));
add( new ClusterSplitter() );
add( new FragmentAssociator() );
add( new Dirc&dE/dxParticleID() );
add( new TrackClusterAssociator() );
add( new TPCGammaConversionFinder() );
add( new FinalStateParticleReconstructor() );
add( new Pi-ZeroFinder() );
add( new K0,Rho,etcFinder() );
add( new ZVTop() );
add( new W,Z,t,Higgs,etcFinder() );
add( new JetMassAnalysis() );
```

Sidebar - projects

ClusterID project sub-projects list

Why are high energy photons mis-IDed as neutral hadrons in the single particle samples with tt distributions in SDnocalgaps?

Can we identify and separate overlapped photons?

Pi0 finder

Can we point hadron fragments back at the beginning of a hadron shower?

- For neutral hadron showers the beginning is the first hit layer area.
- For charged hadrons we will have to find the shower beginning point perhaps where the # hits/layer increases or energy/layer increases or both.

Why are some pion showers not connected as it seems they should be from looking at Wired and the hits in single particle events? This is a low percent effect.

Do we need a MakeSignalEventTrainingDataSets?

Why are there Simple clusters with adjacency 1 with significant energy and #hits all lying in one layer?

Add muon and electron ID.

Add overlapped clusters ID.

Why do we get different answers for hemisphere and jet masses each time we run the program over the same data?

Does Ron's CID code output match the ClusterID input type and probs?