# STUDYING THE CMS ECAL TRIGGER AT THE CERN LHC

Kirsten Randle
Prof. Toyoko Orimoto

# THE ELECTROMAGNETIC CALORIMETER

- The Compact Muon Solenoid (CMS) detects proton-proton collisions at the CERN LHC
- CMS is made up of several layers that study different properties of particles
- The Electromagnetic Calorimeter (ECAL) is the layer that is sensitive to photons and electrons.

layers: -silicon tracker, -e cal, -hcal, -solenoid (not a detector, produces B-field) , - muon chambers

# THE ELECTROMAGNETIC CALORIMETER

- Particles scintillate in lead tungstate ($PbWO_4$) crystals
- Scintillation light is collected by photodetectors
- Signal is amplified, shaped, and digitized

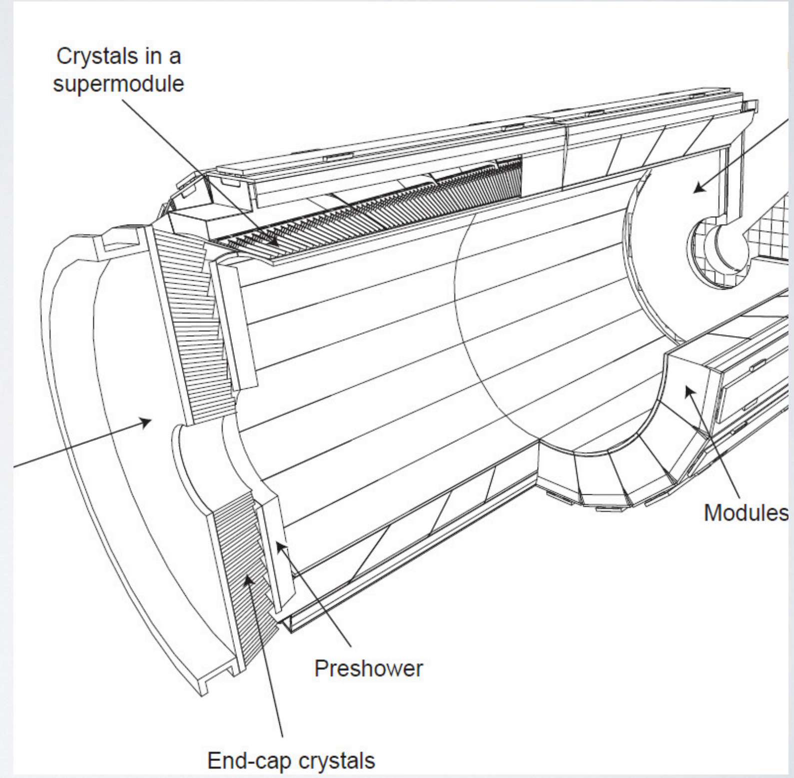



Image courtesy CMS Collaboration

CMS is split into 2 parts -barrel -endcaps

Crystals are first in groups of 5 crystals for one board (VFE -very front-end) and then in larger groups that depend on the angle from the beam-axis ( eta = $-\ln(\tan(\theta)/2)$ )

# ENERGY RECONSTRUCTION OF SIGNALS

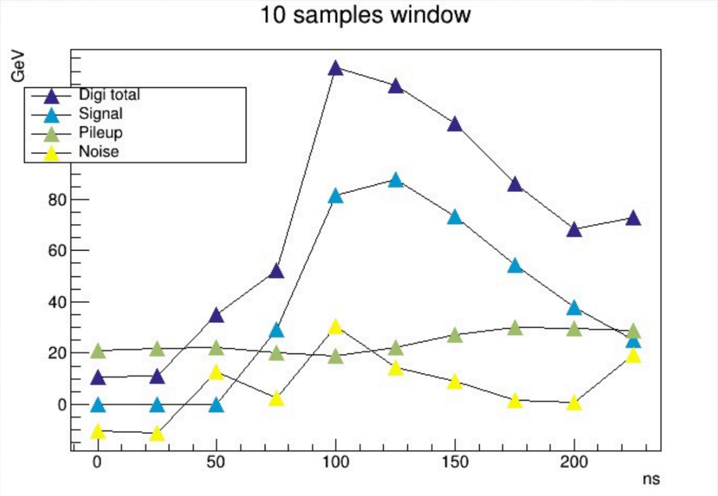



Image courtesy A. Tishelman-Charny

- Accurate & precise energy reconstruction is necessary for triggering
- For every window of 10 samples (taken every 25ns by onboard electronics), each sample is multiplied by an assigned weight for energy reconstruction
- Initial Run2 studies have shown that weights are not ideal and can be improved

Weights are designed to preferentially amplify the peak amplitude over the background to get more accurate energy reconstruction of the event

# WEIGHTS

- Amplitude weights
  - active, uniform for whole detector
  - Current study: update weights and vary weights for different parts of the detector
- Timing weights
  - VFE boards have unused capacity for second set of weights
  - Current study: optimize timing weights to identify out-of-time pileup in a signal

Reminder: VFE -very front end

Crystals in high-eta regions, like the endcaps are especially susceptible to radiation damage and may need unique weights, while weights are currently uniform for the whole detector

# PILEUP

- Bunches of protons meet inside the detector every 25 ns, this is called a Bunch Crossing (BX)
- Scintillation in the detector takes much longer than this (10 samples, 250 ns)
- Pieces of signals from other bunch crossings can add to overall amplitude
- Out-of-time pileup not only changes the amplitude but also the pulse shape

# SCOPE OF PROJECT

- Get ROOT and pyROOT running
- Write a flexible plotter that uses ROOT
- Produce interesting of plots of various parameters used to study the amplitude and timing weights from simulated data

ROOT is C++ based particle physics analysis framework
pyROOT — root libraries can be imported into python
do two layers of code: first use a layer

# PLOTTER

- Plotter is split into 2 layers to separate time-heavy processes for ease of use
- ROOT libraries are used for power to deal with large datasets
- First layer takes data and produces histogram objects and saves them to a file that can be accessed for the second layer
- Second layer takes histogram objects and plots them, since second layer runs quickly it can be tweaked easily
- Flexibility makes changes to cuts, studying different parameters, or repeating the same studies on different data sets easy

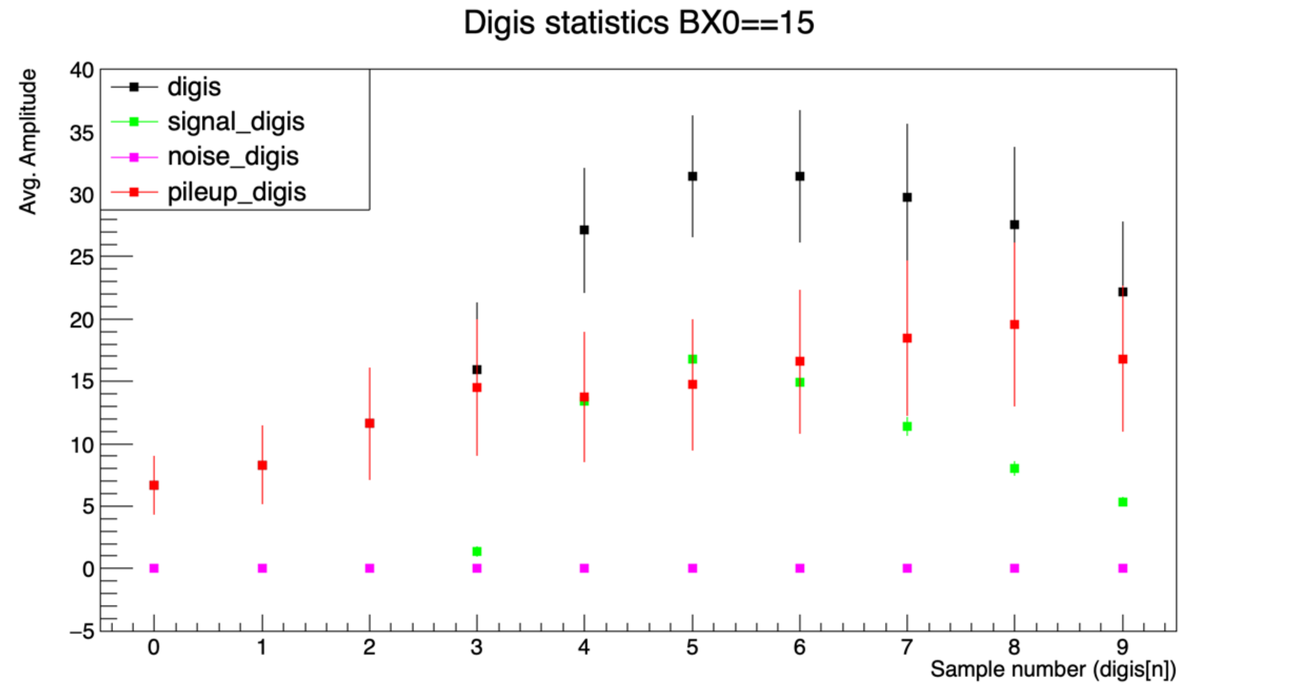
```python
def create_1Dhisto(bias_tree,histo_name,binparams,parameter,cuts):
    h = TH1F(histo_name,parameter,binparams[0],binparams[1],binparams[2])
    h.GetXaxis().SetTitle(parameter)
    h.GetYaxis().SetTitle('Entries')
    drawstatement = parameter + ' >> ' + histo_name
    bias_tree.Draw(drawstatement,cuts,'hist')
    h.SetDirectory(0)
    return h

def create_2Dhisto(bias_tree,histo_name,binparams,parameters,cuts):
    h = TH2F(histo_name,histo_name,binparams[0][0],binparams[0][1],binparams[0][2],binparams[1][0],binparams[1][1],binparams[1][2])
    h.GetXaxis().SetTitle(parameters[0])
    h.GetYaxis().SetTitle(parameters[1])
    drawstatement= parameters[1] + ':' + parameters[0] + ' >> ' + histo_name
    bias_tree.Draw(drawstatement,cuts,'COLZ1')
    h.SetDirectory(0)
    return h

def slicefity(histo,func,slicebins,options):
    fitparams = TObjArray()
    histo.FitSlicesY(func,slicebins[0],slicebins[1],slicebins[2],options,fitparams)
    return fitparams

def iterate_curves(tree,curvelist,type):
    list = [0 for x in range(len(curvelist))]
    for i,p in enumerate(curvelist):
        if type == 'TH1F':
            list[i] = create_1Dhisto(tree,p[0],p[2],p[3],p[4])
        if type == 'TH2F':
            list[i] = create_2Dhisto(tree,p[0],p[2],p[3],p[4])
    return list
```

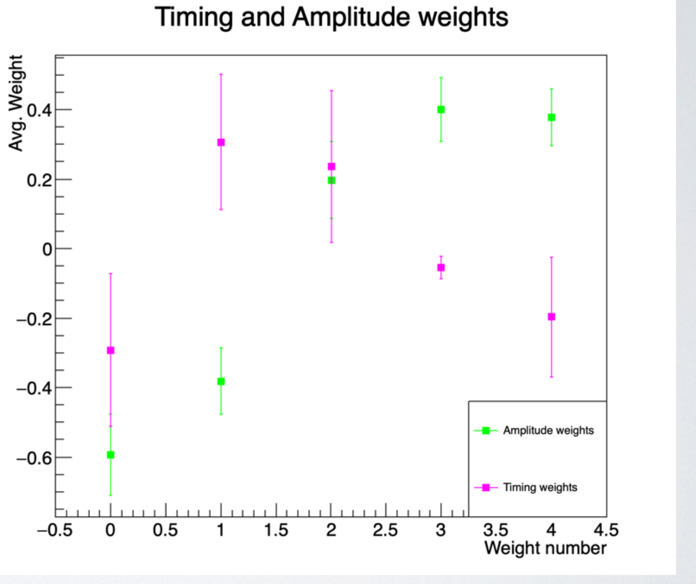
# SIMULATED SIGNALS




The average signal (green)
Average electronic noise (magenta) (can be positive or negative, small amplitude, average is zero)
Pileup (red) is very variable— only 1 BX is studied, which means that different bunch crossings may have different pileup behavior (BXs with more or fewer events preceding/following it will have different distributions)
This plot is at eta ring 28, bunch crossing 15

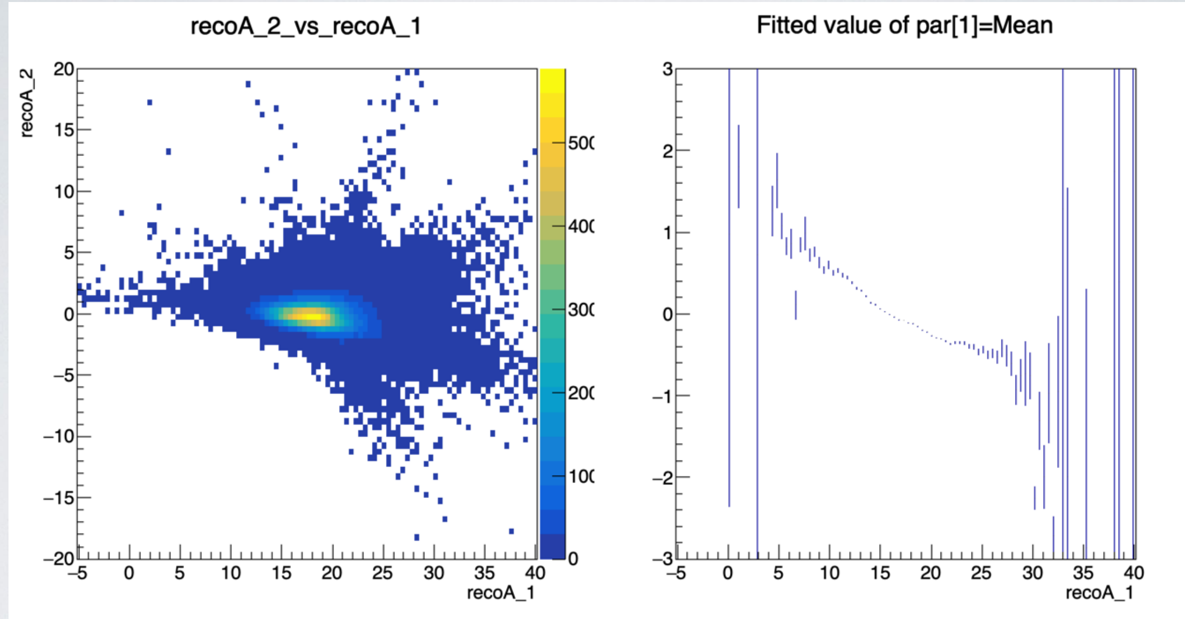
# WEIGHTS

- The model uses the 'true' amplitude to choose the weights that accurately reconstruct the amplitude for each event
- The model similarly chooses the timing weights that reconstruct the amplitude to zero for each event




Check with Abe on how accurate description of timing weights is

# WEIGHTS EFFECTS ON AMPLITUDE

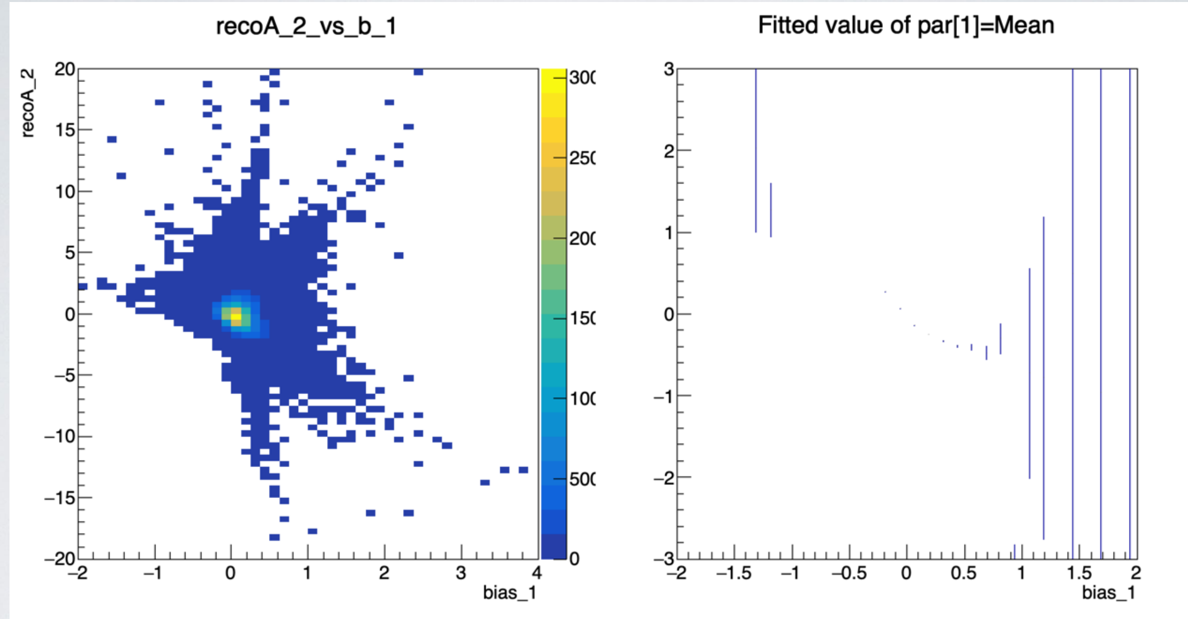



Apparent correlation of reconstruction amplitudes

Right Side uses a ROOT method for 2-D histograms that finds the average value of the y-distribution in each x bin

Examining behavior of recoA_2 vs recoA_1:
we expect the average of recoA_2 to be constant at zero, and for recoA_1 and recoA_2 to be uncorrelated

# WEIGHTS EFFECTS ON AMPLITUDE




bias, a metric for accuracy of energy reconstruction, shows that when recoA_1 fails, recoA_2 also doesn't behave as expected

Examining behavior of recoA_2 vs recoA_1:
we expect the average of recoA_2 to be constant at zero, and for recoA_1 and recoA_2 to be uncorrelated

# FUTURE DIRECTIONS

- Determine what types of events are failing to produce appropriate timing weights (recoA are far from expected values)
- Quantify in which sample high pileup affects amplitude reconstruction the most
- Study failure modes for amplitude weight failure, specifically the pileup dependence
- Examine Timing weight's ability to identify out-of-time pileup

Determine which events are failing — this may require examining single events
Quantify