- ECAL calibration
- Analysis in train

D.Peresunko

NRC “Kurchatov institute”

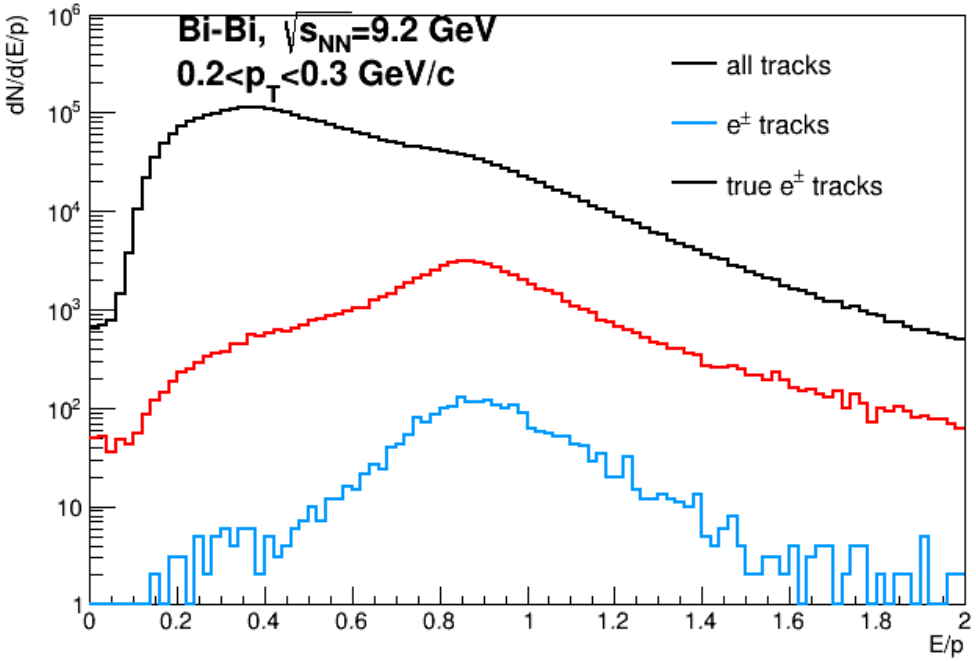
What should be calibrated

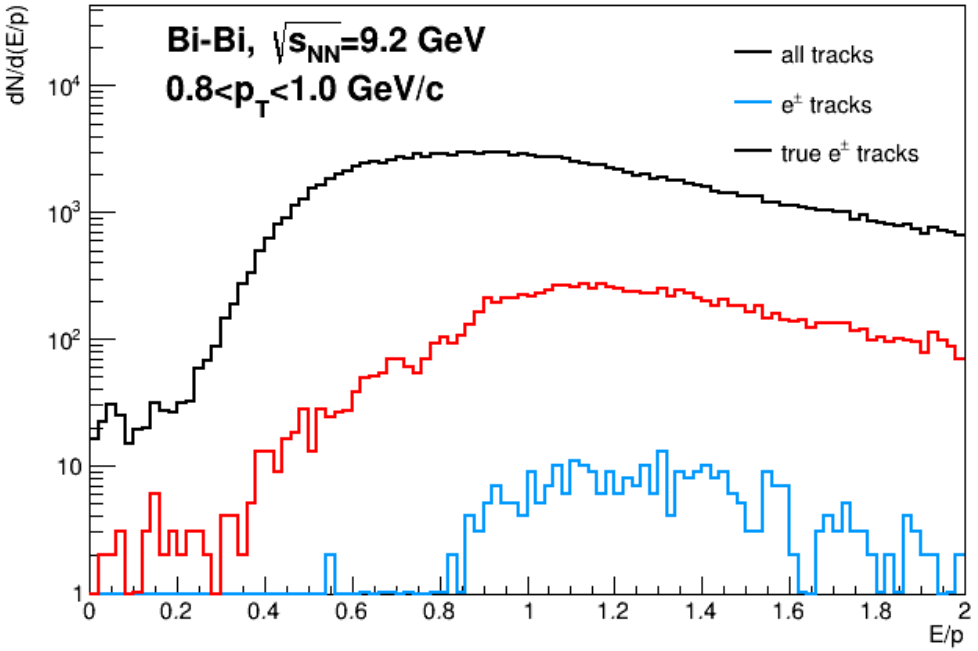
- Energy calibration
  - Relative cell-by-cell calibration
  - Absolute energy scale and alignment
- Time calibration

Relative cell-by-cell calibration

- Pre-calibration using beam-test results
  - Huge and expensive procedure
- Calibration using cosmic muons
- Calibration using slopes
- Calibration using electron E/p peak
- Calibration using π^0 peak

Electron E/p ratio

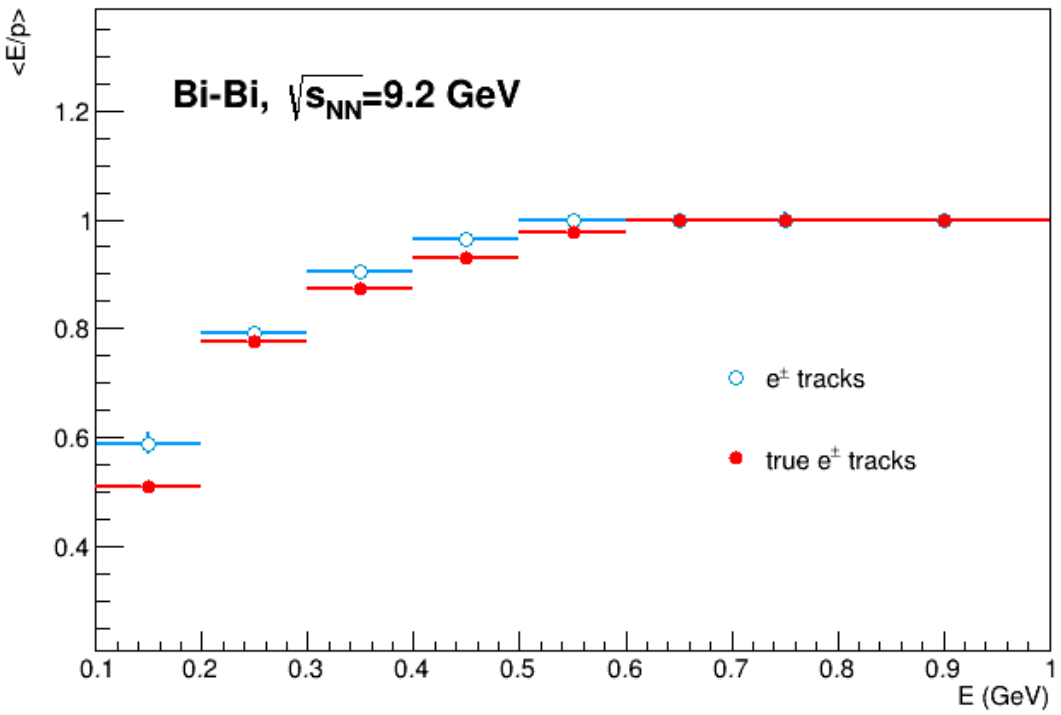




Calculate E/p ratio for all and TPC-identified electron tracks in Bi-Bi collisions

Tiny peak in all tracks E/p ratio
Visible peak in identified electron tracks

Electron E/p ratio



Number of electron tracks is not sufficient to calibrate each channel.

However, one can fix absolute energy scale with electron E/p peak.

Warning! Electrons are sensitive to material budget and this estimate includes uncertainties in description of material budget in front of calorimeter.

Strong ~50% E/p decrease at low energies => large material budget in front of ECAL => large uncertainties.

Calibration with π^0: algorithm

- For each cell in ECAL fill inv. mass distributions for pairs of photons hit this cell and any other
- Find peak position in each cell and calculate correction for calibration for each cell

$$c_i = f(m_i)$$

- Re-clusterize and repeat iterations until no improvement is observed in next iteration

What is correct functional form $f(m_i)$?

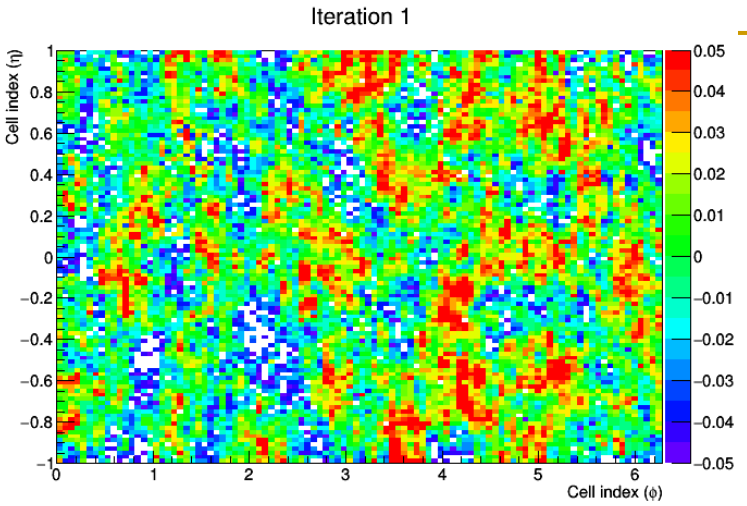
$$c_i \sim E_i, \quad m_{ij} = \sqrt{E_i E_j (1 - \cos\theta_{ij})} \quad \Rightarrow \quad c_i = \left(\frac{m_i}{\langle m \rangle} \right)^2$$

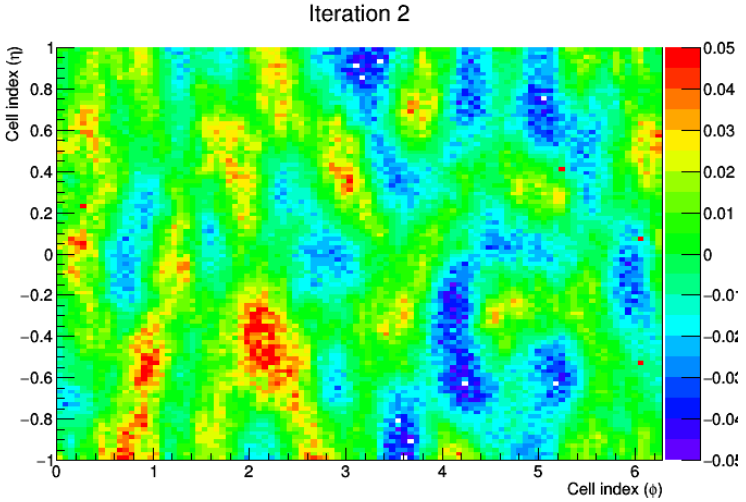
m_i: peak position, measured in given cell i

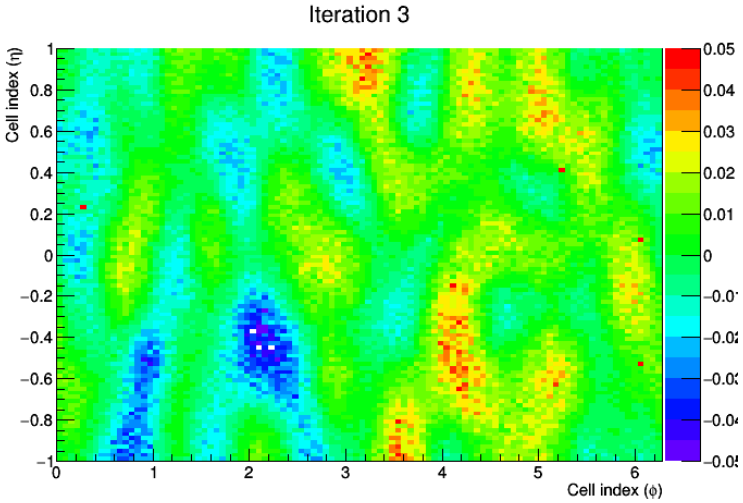
Toy simulation

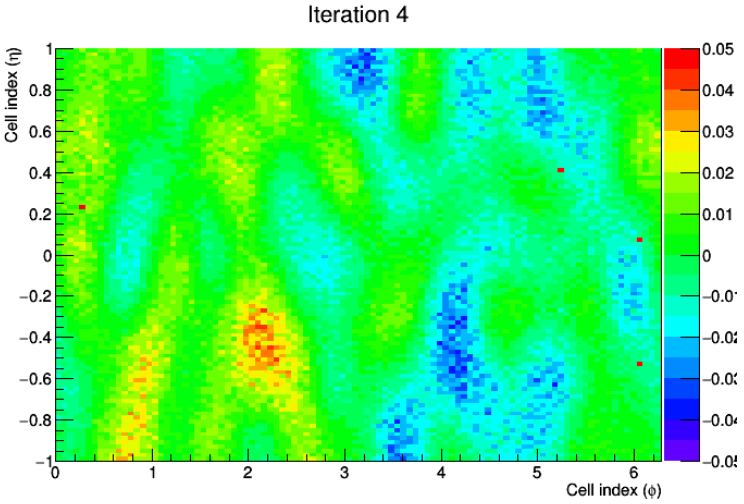
- Simulate calorimeter by introducing grid in (φ,y)
  - 100*100 cells, full phi, |y|<1
- Generate random de-calibrations for each cells
  - gRandom->Gaus(1,resolution)
- Add random energy smearing for each photon
  - $E_{meas}=c_i$*gRandom->Gaus(E_{true},σ_E), $\sigma_E^2=0.02^2+0.04^2/E$
- Generate π^0s, decay to photons and decalibrate (and smear) photon energies according to de-calibration coefficients
  - Realistic p_T distribution, flat in rapidity and phi
- Look at correlation between estimated and true calibration coefficients for all cells.

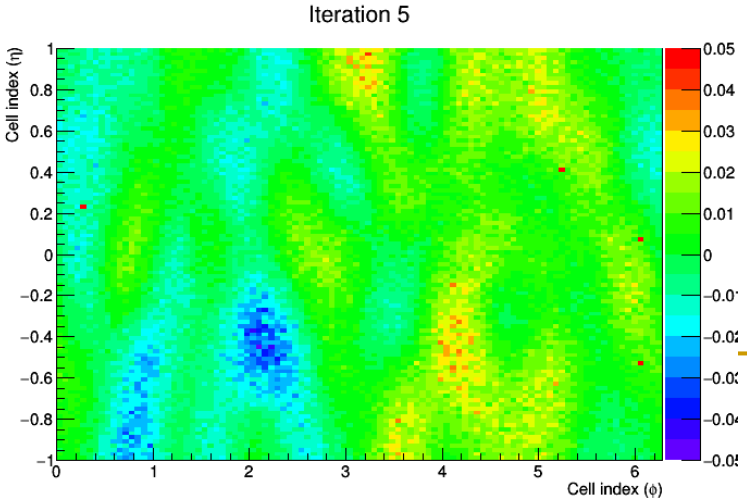
Iterations: c~m^2







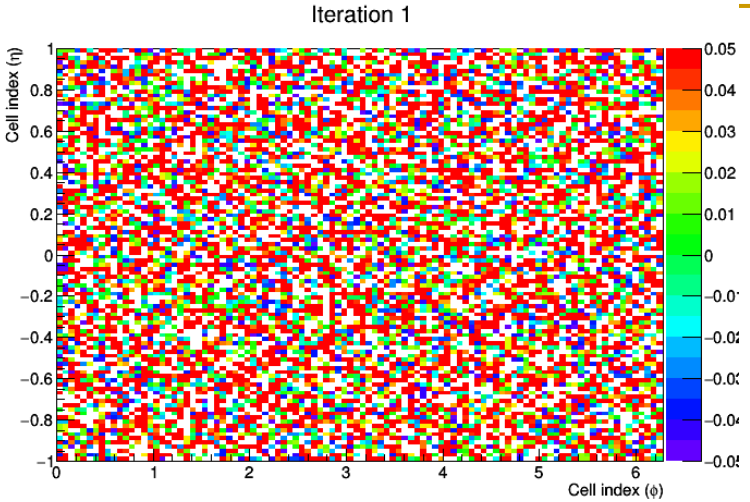


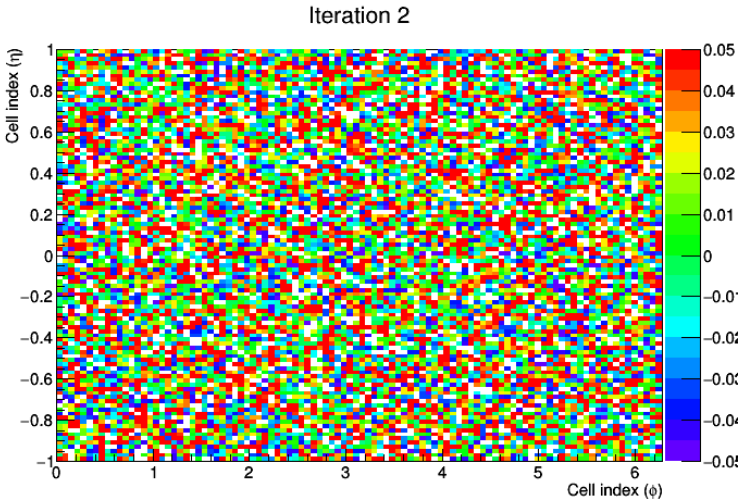


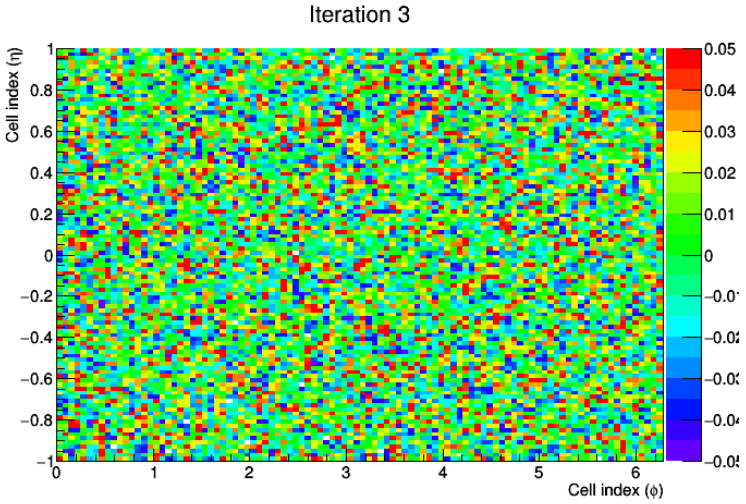
Plot difference between true and estimated calibration $C_{estimated} - C_{true}$ for each cell.

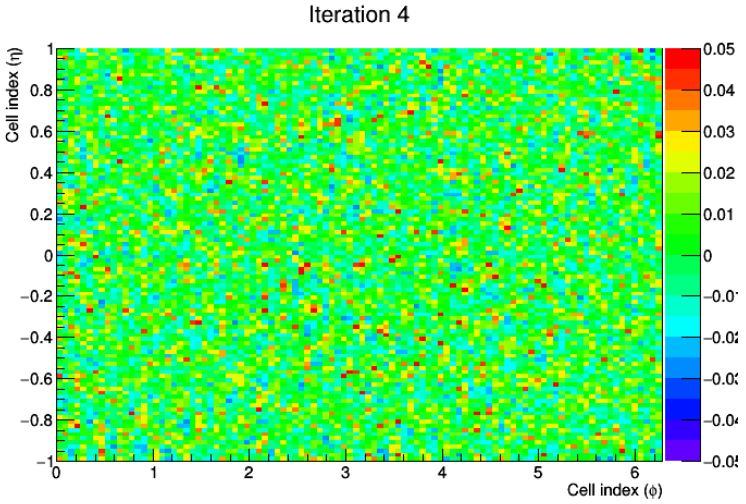
One can see diminishing oscillations in some regions in subsequent iterations.

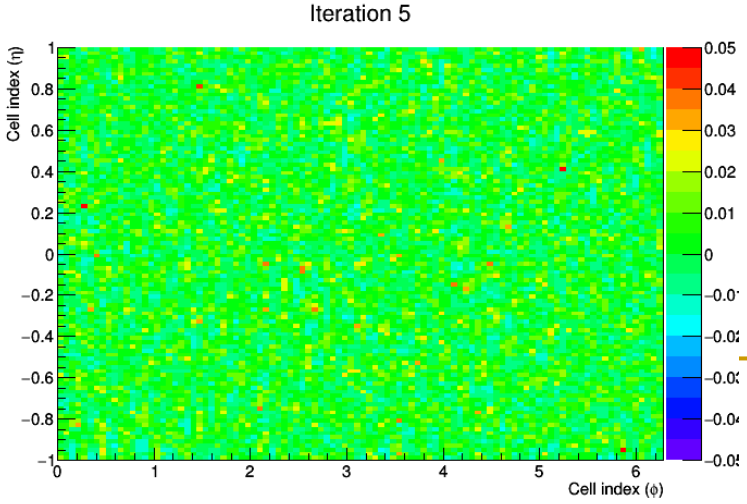
Iterations: c~m[1]











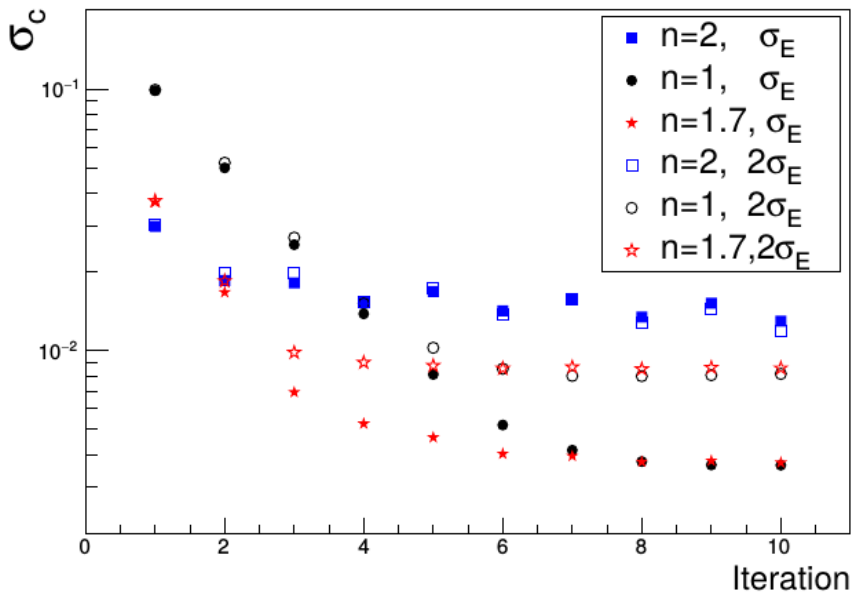
Plot difference between true and estimated calibration $C_{estimated} - C_{true}$ for each cell.

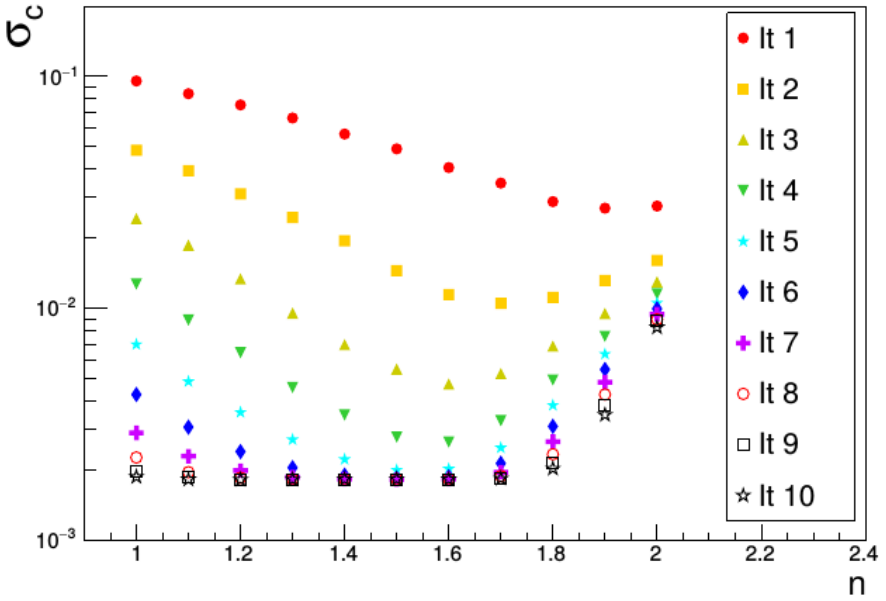
Iteration 0: exactly the same initial de-calibration

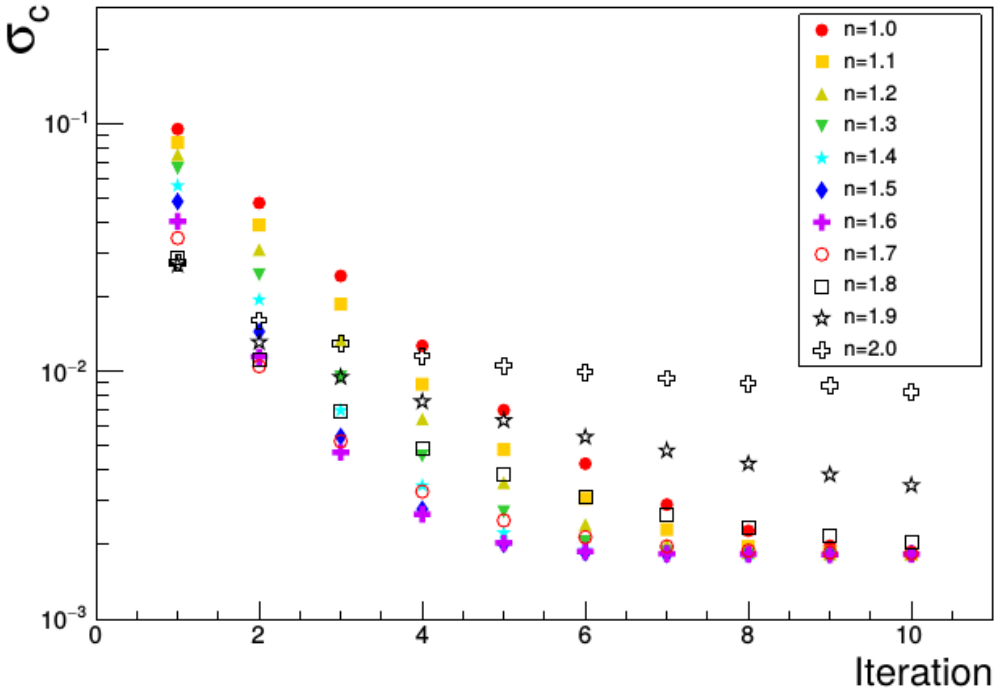
One can see no oscillations in subsequent iterations.

Optimal power of the correction

Quantitative estimate: width of distribution over the difference (true-estimated) calibration coef.





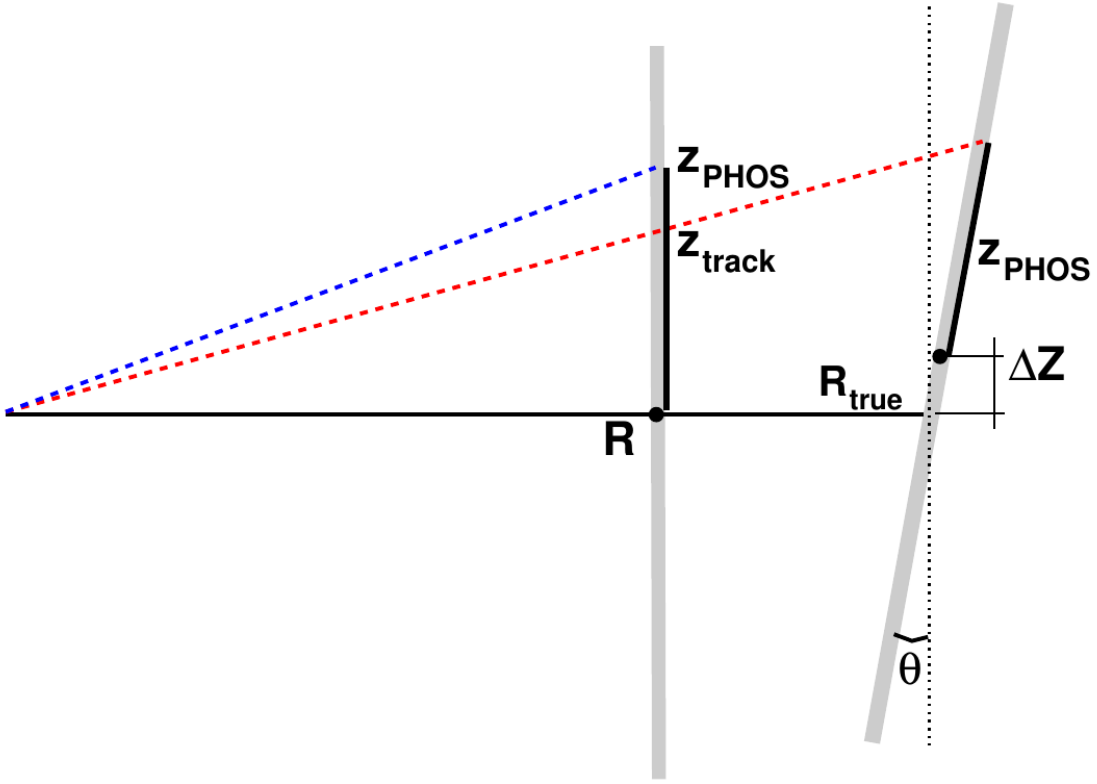


With power parameter n=2 one gets the best 1-2 iterations, then procedure start to fluctuate.
The best value is n~1.6-1.7 for ~5 iterations

Absolute energy scale and alignment

If description of ECAL geometry in MPD root is wrong, then putting π^0 peak at proper position may result in wrong absolute energy calibration:

$$m_{\gamma\gamma} = \sqrt{2E_1E_2(1-\cos\theta_{12})} = 2\sqrt{E_1E_2}\,|\sin(\theta/2)| \approx \sqrt{E_1E_2}\frac{L_{12}}{R}$$

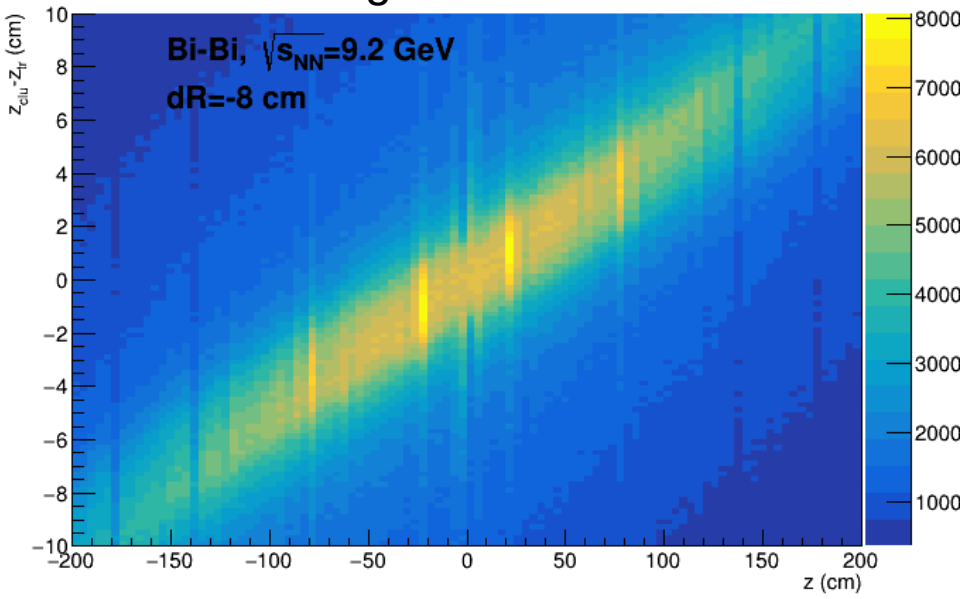


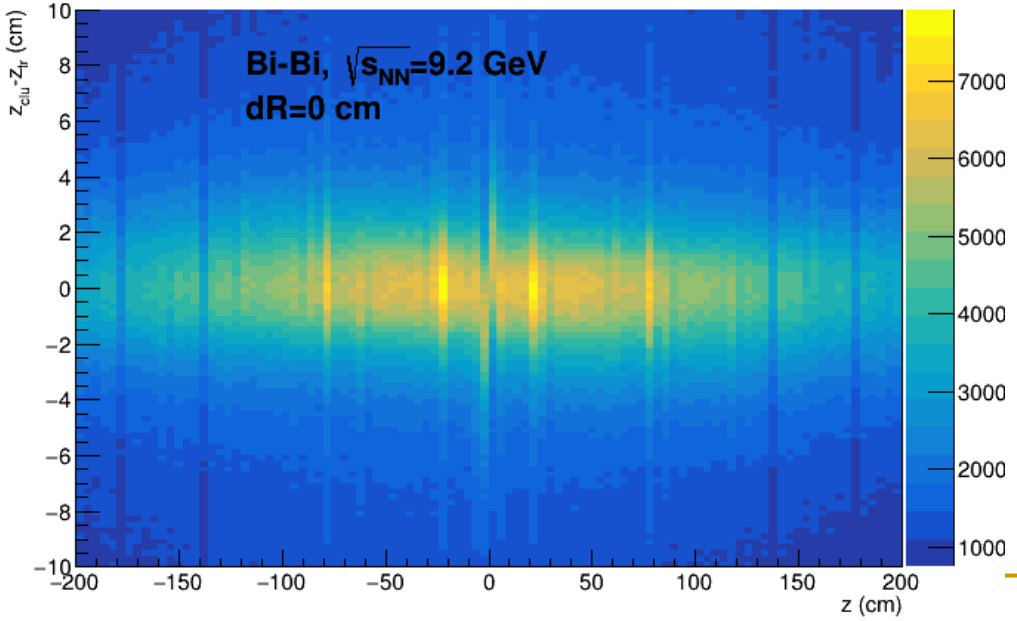
The most dangerous is shift in radial direction. Can be checked by

- Study of systematic dz(z) shift between (electron) tracks and matched clusters in PHOS
- Electron E/p
- PCM-PHOS pion analysis
- …?

dz(z) dependence

All charged

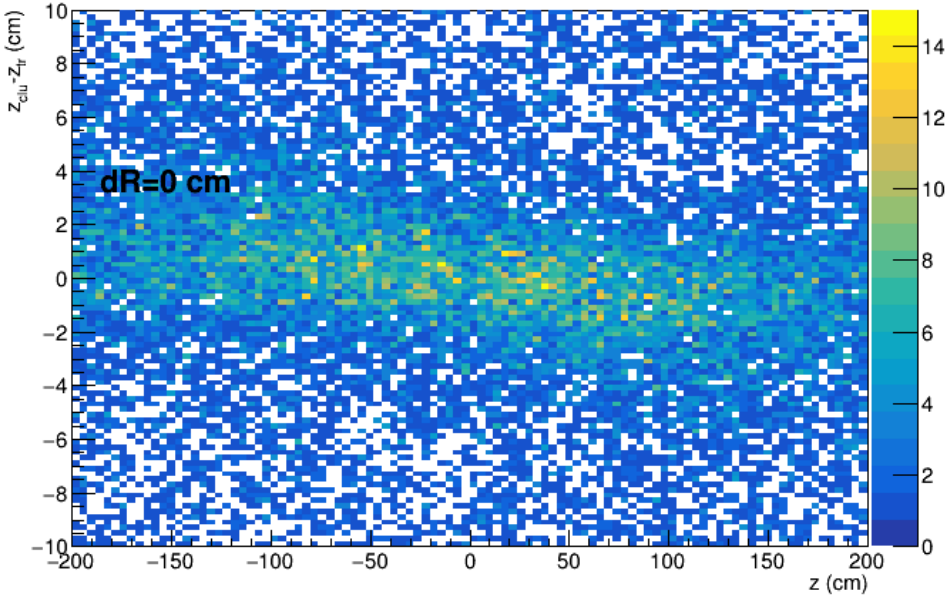




Dependence dz(z) is sensitive enough to find mis-alignment in radial direction with accuracy ~1 cm

Depth of electron and hadron shower slightly differs, should be accounted.

Electrons

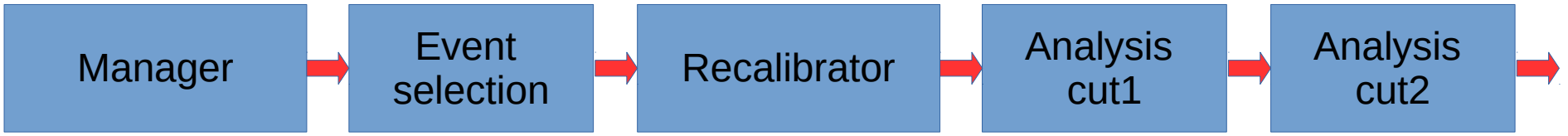


Train analysis

- In simple analysis like spectrum construction, invariant mass calculations etc. most of time is spent in accessing data
  - => Read data and run several analyses on same sample
- Run systematic study in one pass
- Run standard re-calibration and data filtering before analysis

Train analysis

- Analysis manager reads event into memory
- Calls wagons to analyze (modify) data:



Code committed into MPD/physics dir:

https://git.jinr.ru/nica/mpdroot/-/tree/dev/physics

Structure

Class **MpdAnalysisManager** requires list of input files, list of branches to be used for analysis and list of tasks to process these files. Finally MpdAnalysisManager takes care of writing output objects for each task with special list.

Taks which will be called by MpdAnalysisManager should be derived from **MpdAnalysisTask** and have several methods implemented:

void UserInit(); // Users should prepare objects to fill (histograms, trees etc)

void ProcessEvent(MpdAnalysisEvent &event) ; // method is called for each event and event data are provided by container MpdAnalysisEvent

void Finish() ; //method is called when scan in finished but class data are not written yet.

Class **MpdAnalysisEvent** contains references to all branched containing data for this event. MpdAnalysisManager can be configured to read only few branches reasllly necessary for analysis.

Example

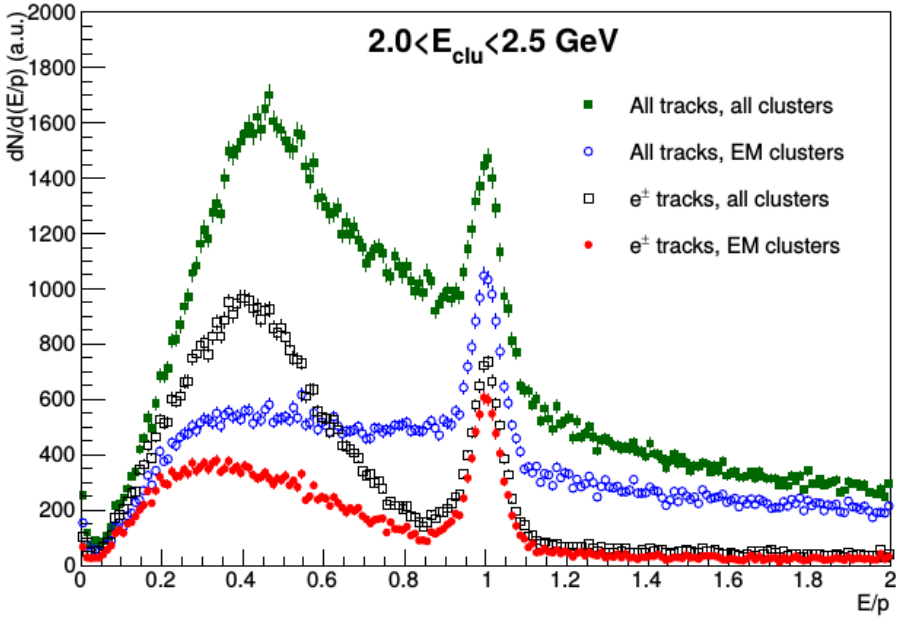
Working example one can find in mpdroot/physics/photons/MpdConvPi0.cxx

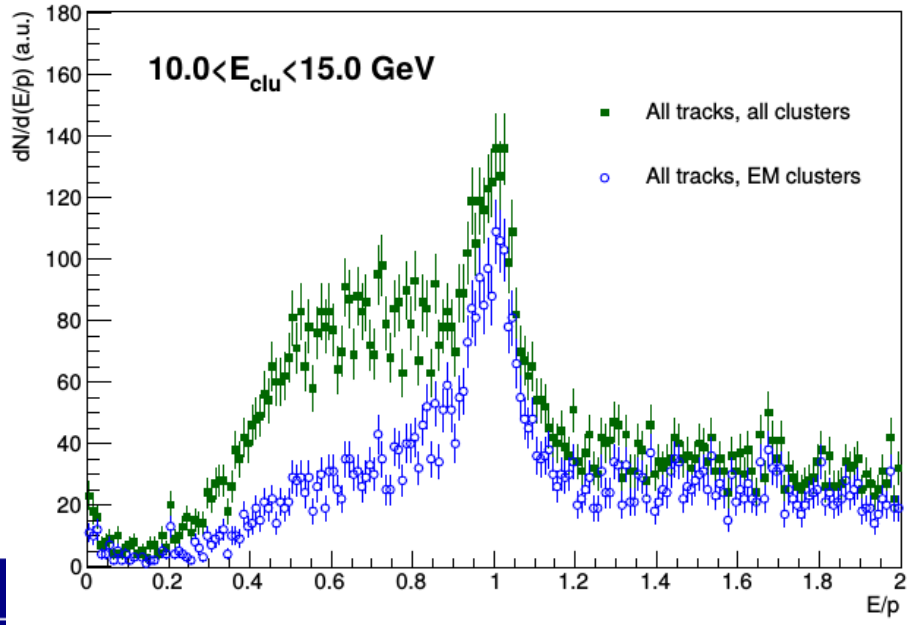
```
void ConversionPi0(){
   gSystem->Load("libEmc.so") ;
   gSystem->Load("libMpdPhysics.so") ;
   gSystem->Load("libMpdPhotons.so") ;
   MpdAnalysisManager man("ManagerPi0") ;
   man.InputFileList("list.txt") ;
   man.ReadBranches("*") ;
   man.SetOutput("histos.root") ;

   MpdConvPi0 pDef("pi0Def","ConvDef") ; //name, param file
   man.AddTask(&pDef) ;
   MpdConvPi0 pMinE("pi0MinE","ConvEmin") ;
   man.AddTask(&pMinE) ;
   man.Process() ;
}
```

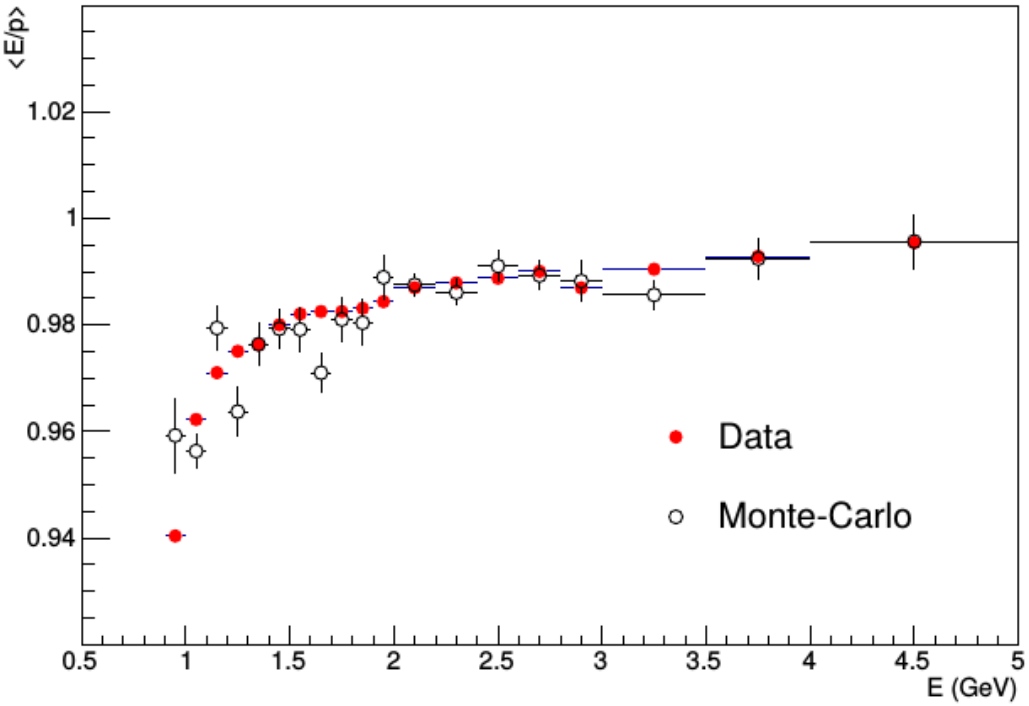
```
//-------------------------------------
void MpdConvPi0::ProcessEvent(MpdAnalysisEvent &event){
  mEMCClusters = event.fEMCCluster;
  int n = mEMCClusters->GetEntriesFast() ;
  for (int i = n; i--; ){
    MpdEmcClusterKI * clu = (MpdEmcClusterKI*) mEMCClusters->At(i);
    float e = Nonlinearity(clu->GetE()) ;
    mhCluCutEff->Fill(0.,e) ;
    if(e < mParams.mCluEmin){
      continue ;
    }
    mhCluCutEff->Fill(1.,e) ;
  }

  mKalmanTracks = event.fTPCKalmanTrack;
  mMCTracks = event.fMCTrack ;
  for(int i=0; i<mMCTracks->GetEntriesFast();i++){
    MpdMCTrack* pr= (static_cast<MpdMCTrack*>(mMCTracks->At(i))) ;
    if(pr->GetPdgCode()==111){
      if(pr->GetStartX()*pr->GetStartX()+pr->GetStartY()*pr->GetStartY()<1.){
        TVector3 momentum;
        pr->GetMomentum(momentum);
        hPrimPi0[mCenBin]->Fill(momentum.Pt(), momentum.Y()) ;
      }
    }
  }
```

Backup

Electron E/p ration in PHOS



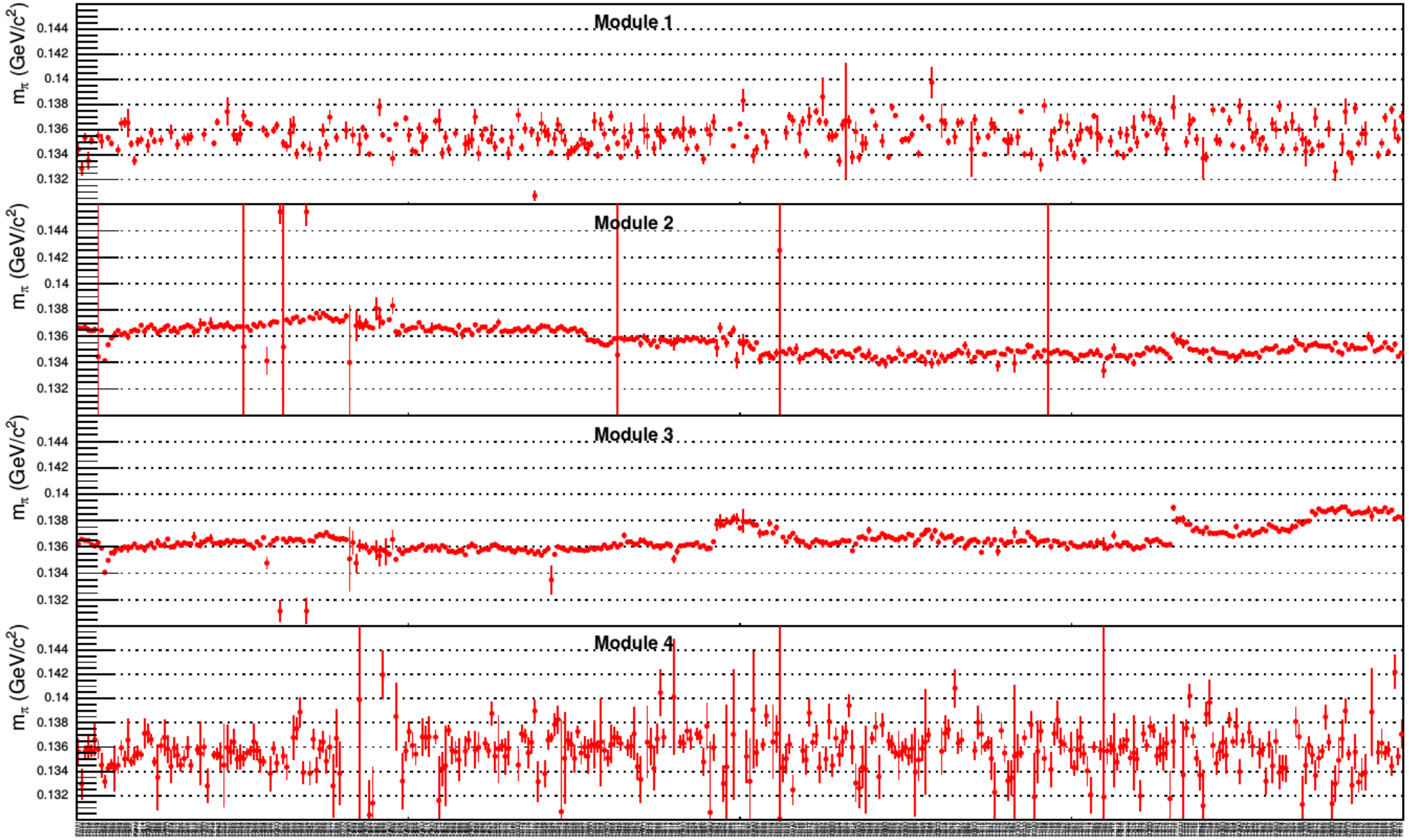




Compare simultaneously in π^0 peak position and electron E/p peak position in data and MC. Difference is the estimate of the systematic uncertainty on absolute energy scale.

Run-by-run correction

LHC17hijkl



Un-correlated drifts in modules are related to FEE cards on/off.
Corrections calculated for runs, jumped no too far from adjacent.
Otherwise mean value for module is used.