

# Reconstruction of CERN runs

V.L. Morgunov and A. Zhelezov

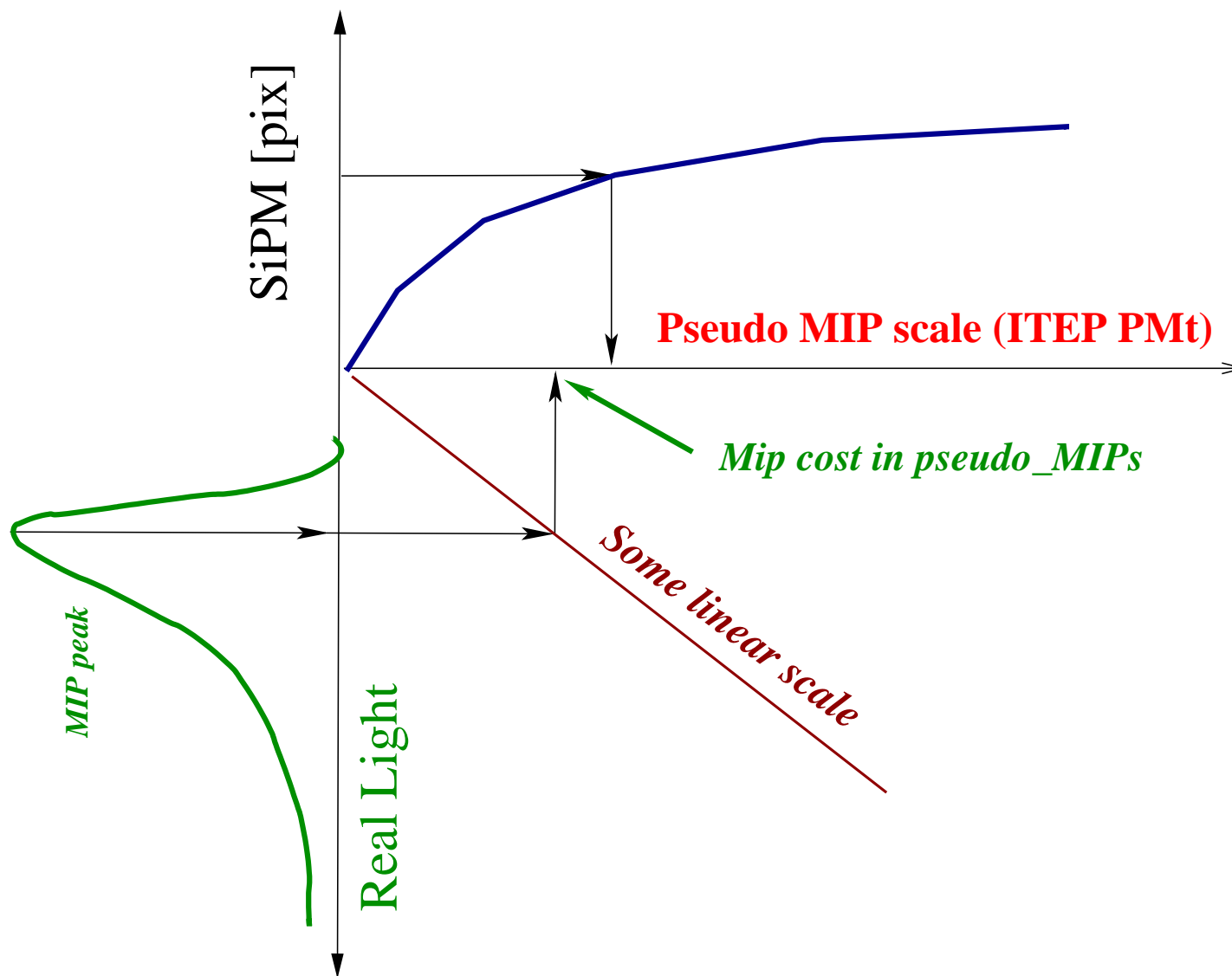
DESY – ITEP



DESY, March 2007

The copy of this talk one can find at the <http://www.desy.de/~morgunov>

# Individual Amplitude Calibration with saturation correction



## How to get it

Technically it is one Class `CallibrationCurve`; (of one screen size)

Easy to use:

One include

```
#include "CalibrationCurve.hh"
```

and one function call

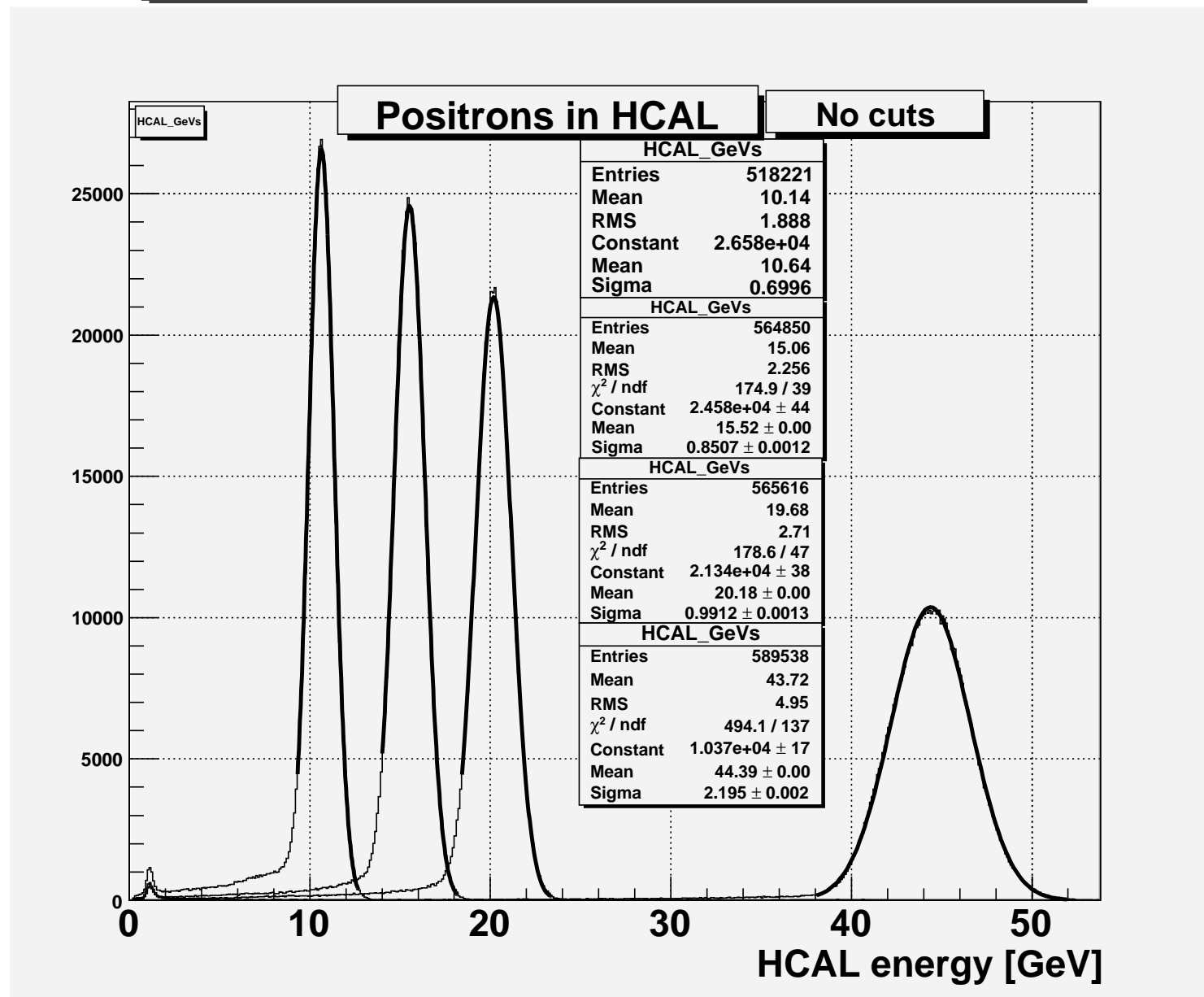
```
double a_mip = curve.correct_amp(pos,lay,Ampl_in_ACD_ch);
```

It creates singleton.

All other calibration databases unneeded more for at least October IIb period.

Gain and InterCalibration (taken from Niels and Beny) are in .

# Positron runs HCAL only (EM scale)



## Deep Analysis

Technically it is also written as one Class `DeepAnalysis`

```
#include "DeepAnalysis.hh"
```

```
#include "cernlib_utils.hh"
```

# BUT:

What does it doing? **It splits of one hadronic shower into 4 different types of clusters.**

The types are:

**Electromagnetic-like;** **Track-like;** **"Hadron-like"** **"Neutron-like"**

All these types are represented as set of clusters. Each cluster has many parameters.

Deep Analysis does not changes amplitude of any hit.

## Using of Deep Analysis class

```
DeepAnalysis deep_analysis; // Create class
deep_analysis.detector.normal_thresh = threshold_1;
deep_analysis.detector.neut_thresh  = threshold_1 + 0.1;
// Fill it with hits
for(int i=ehit_count;i<hhit_count;i++){
    DeepAnalysis::KIND type = DeepAnalysis::KIND_COUNT;
    if(ampl_m[i] < threshold_0)
        continue;
    else if(ampl_m[i] < threshold_2)
        type = DeepAnalysis::TRK;
    else if(ampl_m[i] < threshold_3)
        type = DeepAnalysis::HAD;
    else
        type = DeepAnalysis::EM;
// Occasionaly: it is helpfull to convert hits  HAD->TRK
    if(type == DeepAnalysis::HAD)
        type = DeepAnalysis::TRK;
    deep_analysis.add_hit(xx0[i]/10.,yy0[i]/10.,zz0[i]/10.,
ampl_g[i],ampl_m[i],lay0[i],type);
}
deep_analysis.reconstruction();
```

## Extracting results

```
//    Get results of Deep Analysis
unsigned ec_n_cl=ec_hits=0;
double ec_sum=tc_sum=hc_sum=0.;
deep_analysis.color_clusters_stat(DeepAnalysis::EM,ec_n_cl,ec_hits,ec_sum);

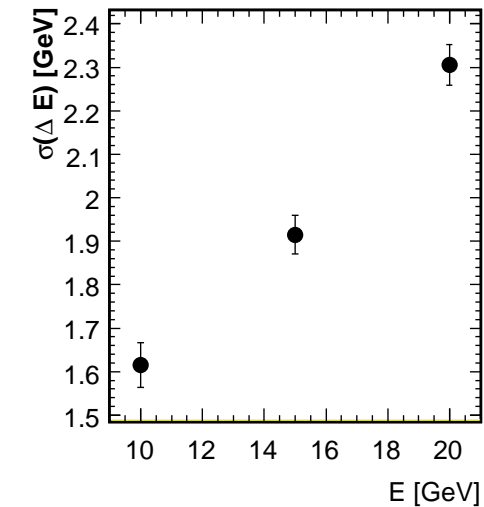
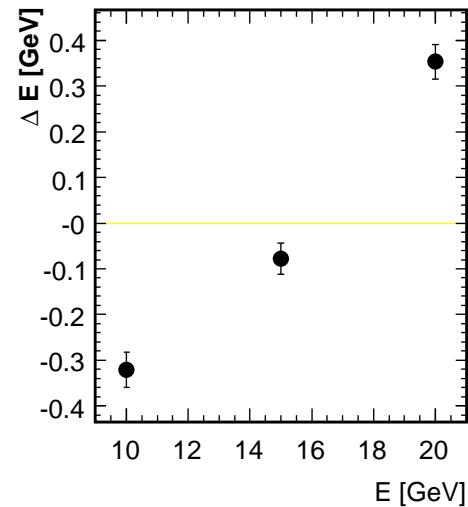
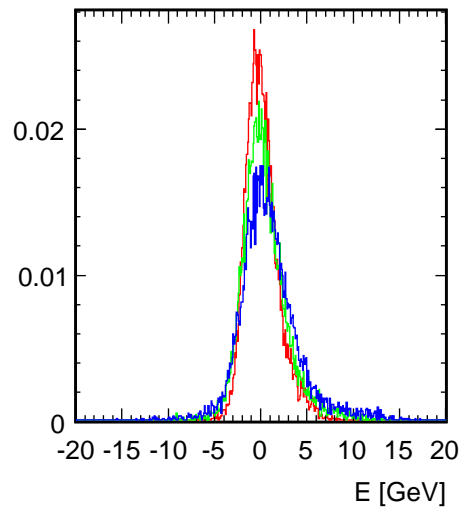
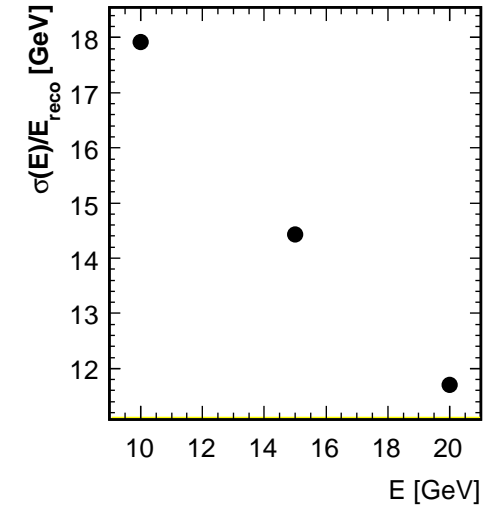
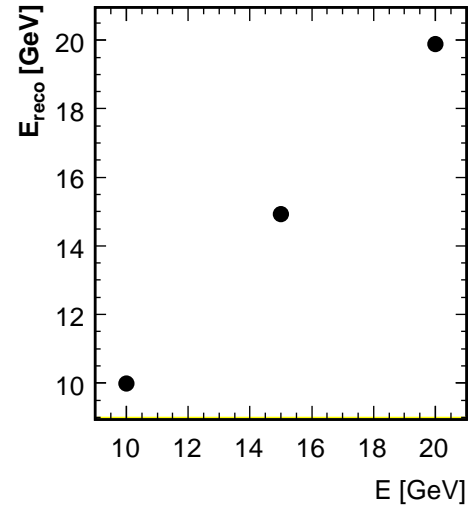
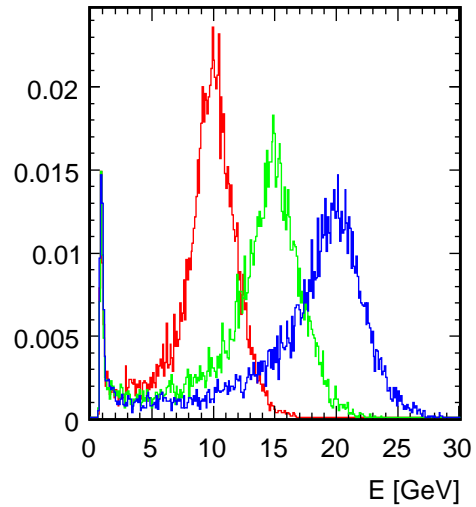
unsigned tc_n_cl=tc_hits=0;
deep_analysis.color_clusters_stat(DeepAnalysis::TRK,tc_n_cl,tc_hits,tc_sum);

unsigned hc_n_cl=hc_hits=0;
deep_analysis.color_clusters_stat(DeepAnalysis::HAD,hc_n_cl,hc_hits,hc_sum);

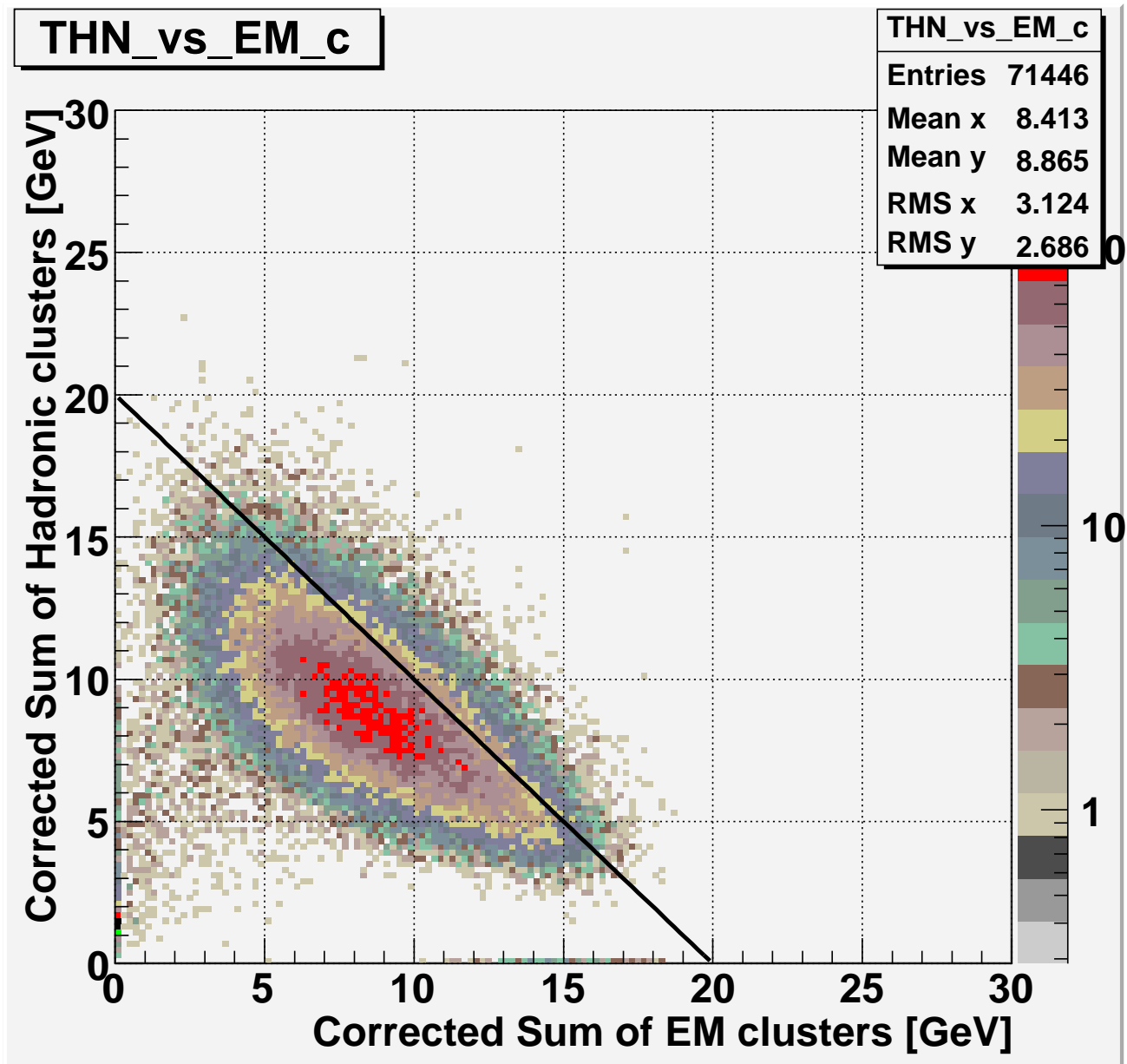
unsigned nc_hits=0;
double nc_sum=0.;
for(RSIterator<DeepAnalysis::Hit> hi(deep_analysis.neutrons);hi.next();)
    nc_sum += hi->am;
if(deep_analysis.neutrons->getNumberOf<DeepAnalysis::Hit>())
    nc_hits=deep_analysis.neutrons->getNumberOf<DeepAnalysis::Hit>();

double ae_sum=0.;
for(RSIterator<DeepAnalysis::Cluster> ci(deep_analysis.clusters);ci.next();)
    ae_sum += ci->ws;
```

# Quality control by Monte-Carlo, thanks Marius

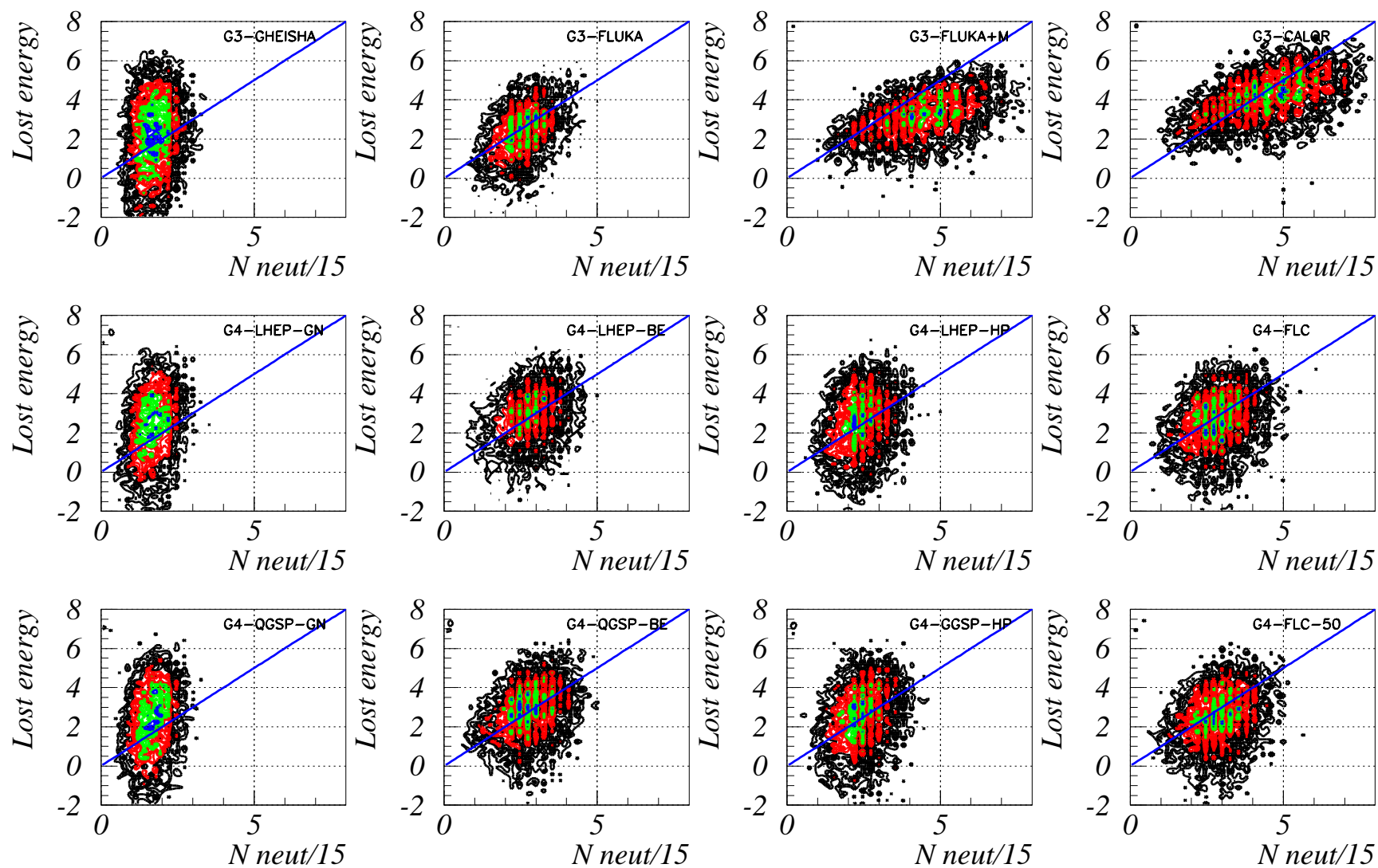


## Correlations inside Hadronic cascade

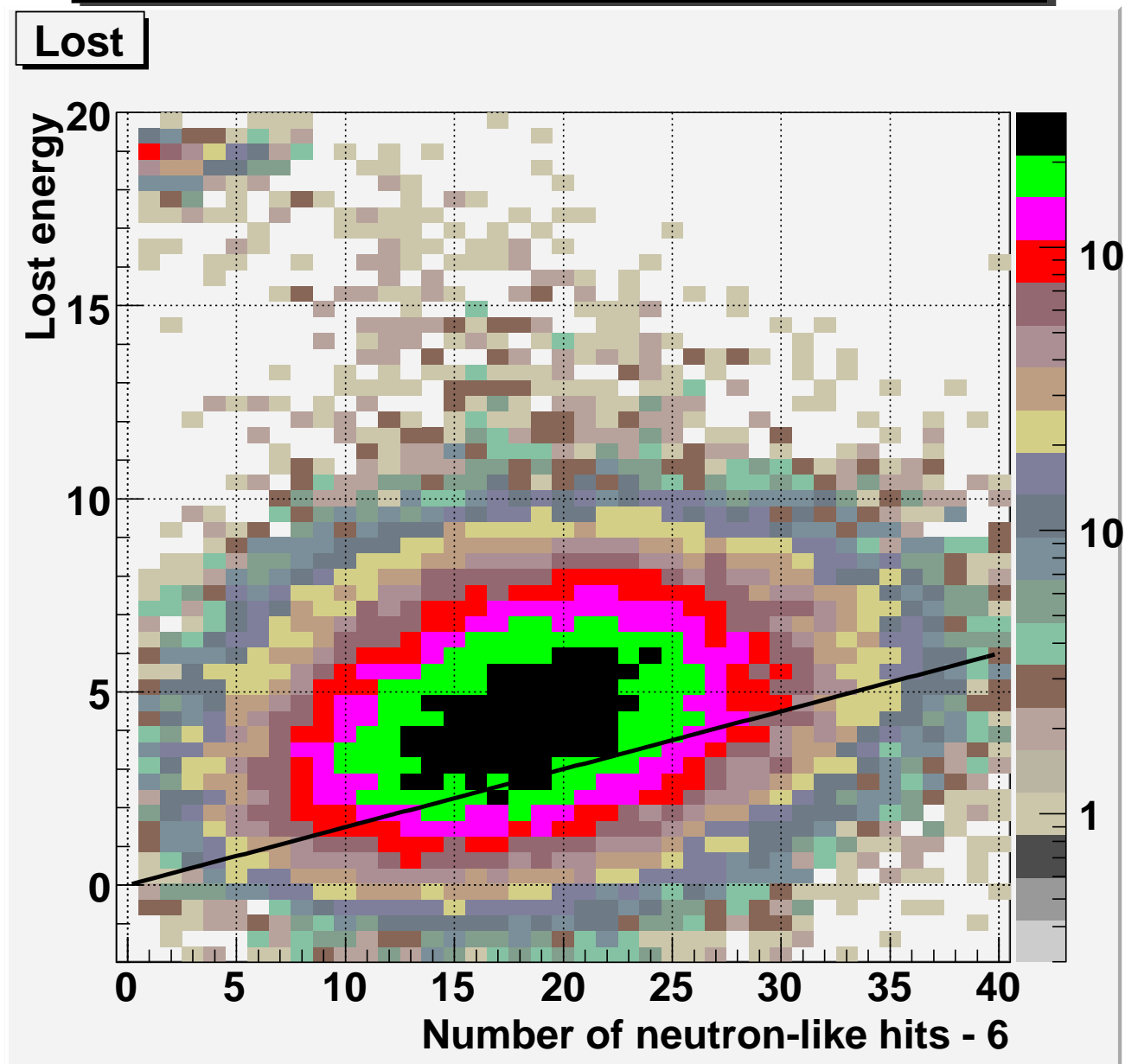


# Old Comparison of Monte-Carlo programs

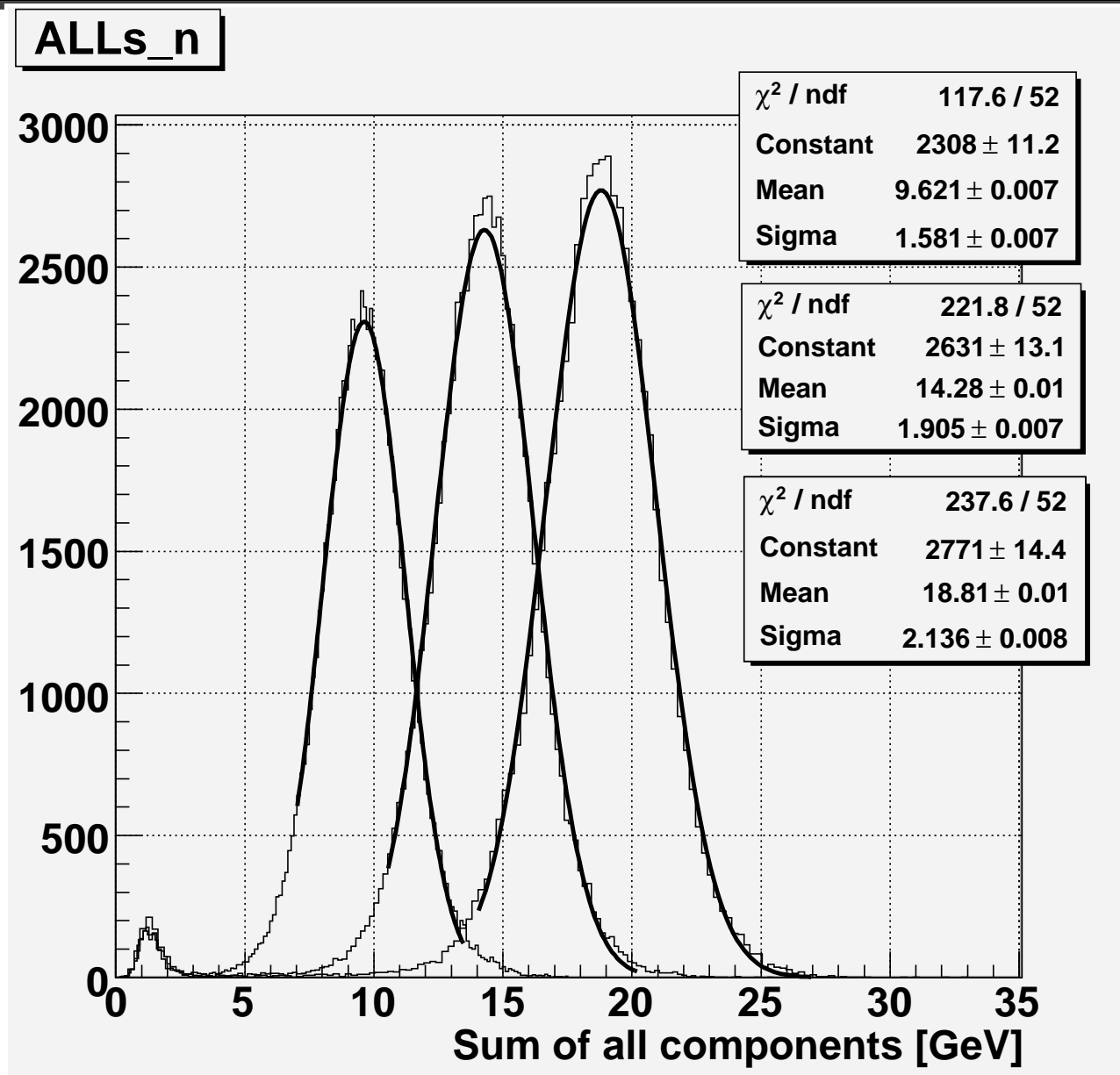
Lost energy vs  $N$  neutron/15



## First Comparison with Monte-Carlo



## Pion runs: Track in ECAL and No Leakage



## **Present for all of You**

**All runs of CERN period October IIb have been treated with new calibration and DeepAnalysis.**

**Root files are at the `/flc/lc2/pool/morgunov/CALICE/CERN/Results`**

**No any selection was done,**

**but all information to make it are in Root-tree**

## Evt ROOT-tree description

```
// Decisions were made about this event
Int_t    Empty;          // 1/0 - true/false
Int_t    Trash;          // 1/0 - true/false
Int_t    Muon;           // 1/0 - true/false
Int_t    Electron;      // 1/0 - true/false
Int_t    Hadron;         // 1/0 - true/false
Int_t    Track_in_ECAL; // 1/0 - true/false
Int_t    Track_in_ECAL_and_HCAL; // 1/0 - true/false
Int_t    Leak;           // 1/0 - true/false
Int_t    No_Leak;       // 1/0 - true/false
Int_t    Veto;           // 1/0 - true/false
Int_t    In_spill;      // 1/0 - true/false
Int_t    Select;        // 1/0 - true/false
Int_t    track_exists; // 1/0 - true/false
Float_t  x_trk_e;
Float_t  y_trk_e; // point at the reconstructed track
Float_t  z_trk_e;
Double_t final_cx;
Double_t final_cy; // direct Cosines of reconstructed track
Double_t final_cz;
Float_t  Ampl_veto; // Veto counter
```

## Evt ROOT-tree description

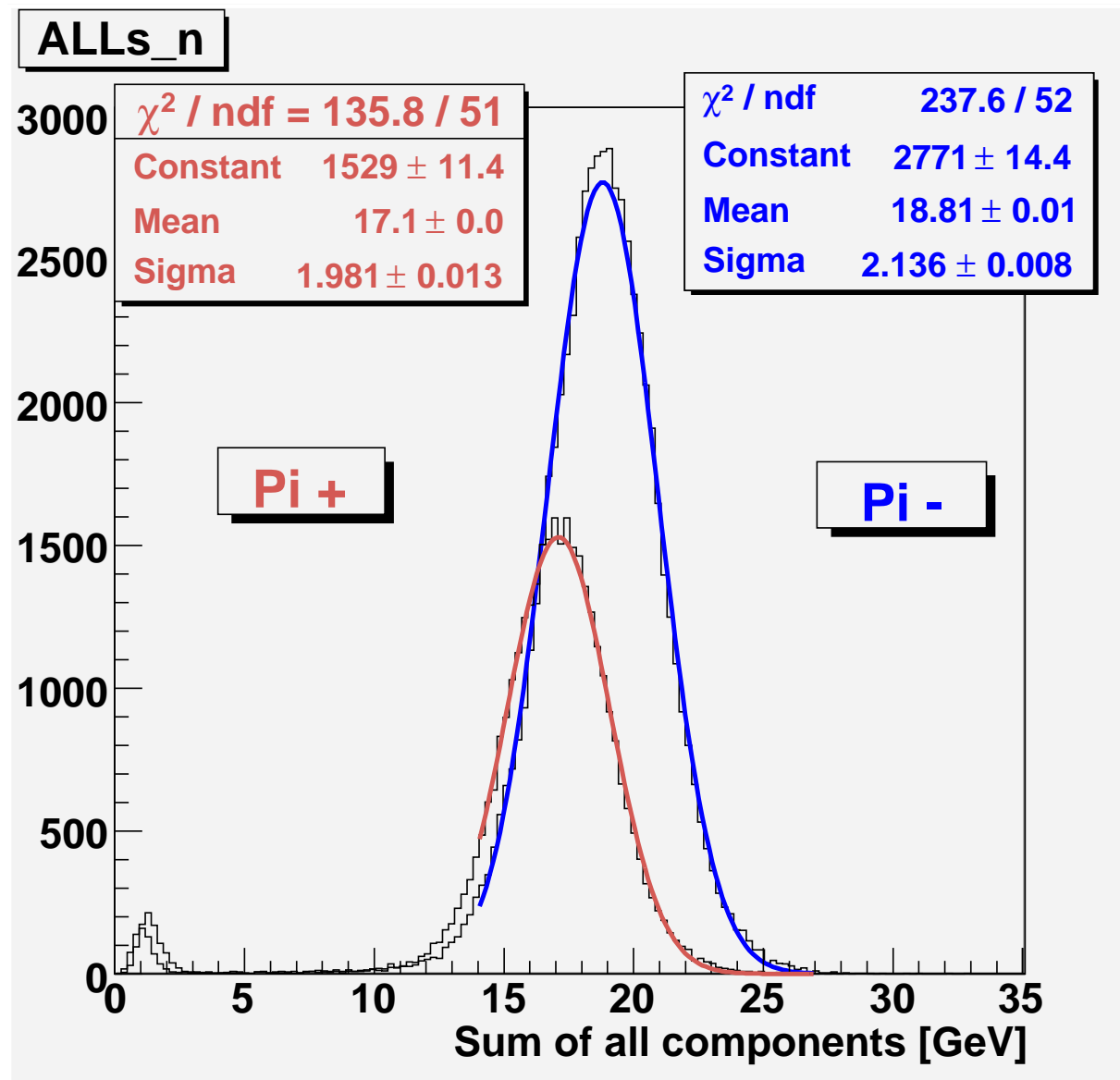
```
// Event as whole
Float_t  Xse;
Float_t  Yse;    // ECAL- Energy center of whole event
Float_t  Zse;
Float_t  Xsh;
Float_t  Ysh;    // HCAL- Energy center of whole event
Float_t  Zsh;
Int_t    E_hit; // ECAL- Number of hit [GeV] over 0.5 MIP
Int_t    H_hit; // HCAL- Number of hit [GeV] over 0.5 MIP
Int_t    T_hit; // TCMT- Number of hit [GeV] over 0.5 MIP
Float_t  E_enr; // ECAL- Sum of hit energies [GeV]
Float_t  H_enr; // HCAL- Sum of hit energies [GeV]
Float_t  T_enr; // TCMT- Sum of hit energies [GeV]
```

## Evt ROOT-tree description

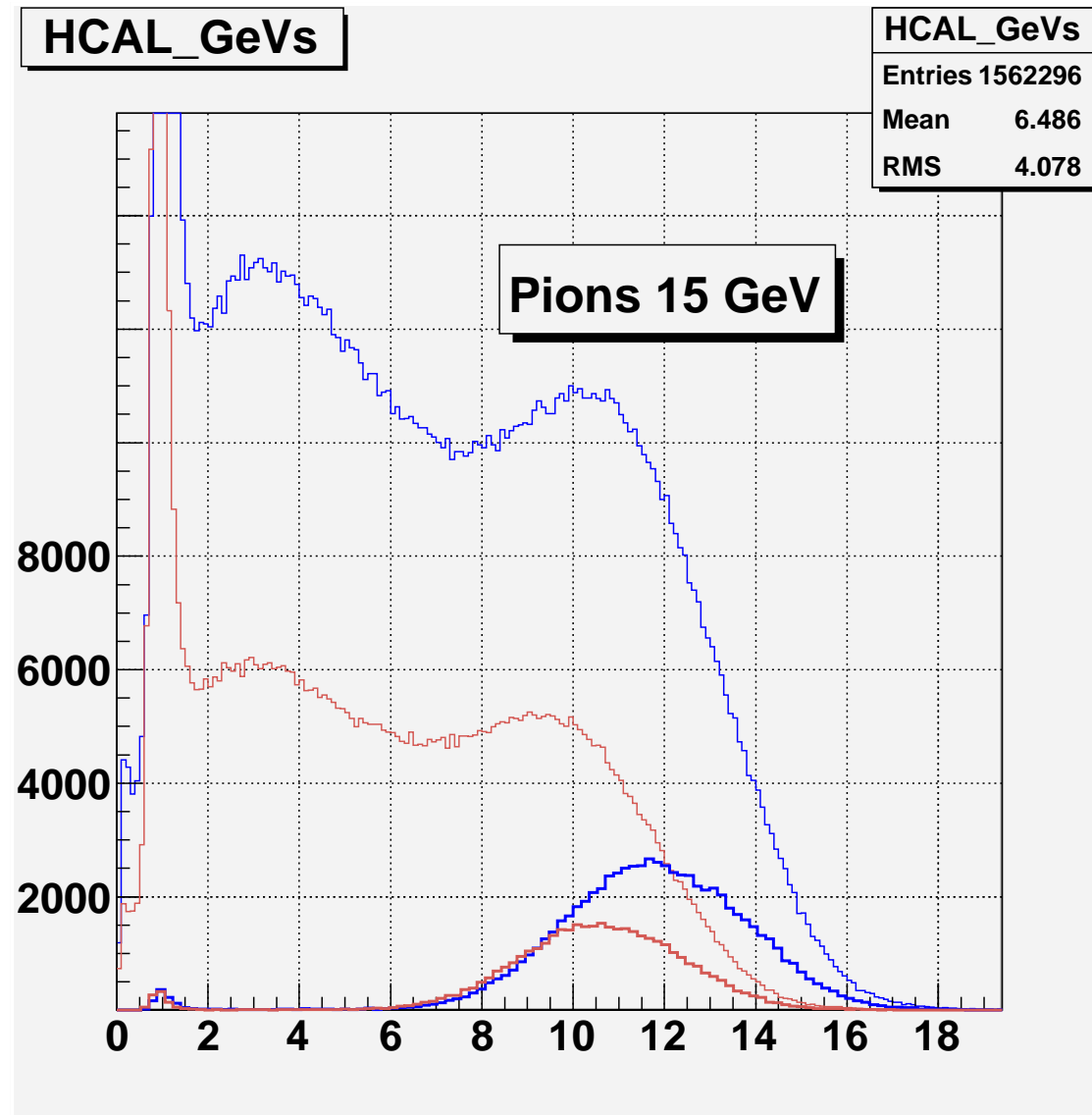
```
// Results of DeepAnalysis
Int_t    E_n_cl;    // Number of EM-like clusters
Int_t    E_n_hit;  // Number of EM-like hits in all clusters
Float_t  E_e_cl;   // Energy sum of EM-like clusters [GeV]
Int_t    T_n_cl;   // Number of TRK-like clusters
Int_t    T_n_hit;  // Number of TRK-like hits in all clusters
Float_t  T_e_cl;   // Energy sum of TRK-like clusters [GeV]
Int_t    H_n_cl;   // Number of HAD-like clusters
Int_t    H_n_hit;  // Number of HAD-like hits in all clusters
Float_t  H_e_cl;   // Energy sum of HAD-like clusters [GeV]
Int_t    N_n_hit;  // Number of NEUT-like hits
Float_t  N_e_cl;   // Energy sum of NEUT-like hits [GeV]

// 0-29 ECAL; 30-52 HCAL; 53-68 TCMT
Float_t  E_m[69];  // Longitudinal profile in MIPs per layer
Int_t    N_m[69];  // Longitudinal profile in number of hits per layer
```

## A lot of questions



# A lot of questions



# A lot of questions

