

# PANDA-Meeting in Bochum

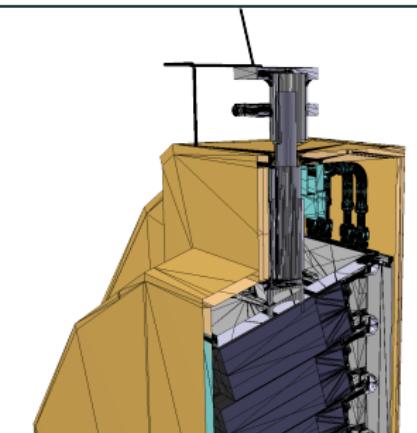
PandaROOT-EMC-Software: Restructuring - An update

---

Ben William Salisbury  
[salisbury@hiskp.uni-bonn.de](mailto:salisbury@hiskp.uni-bonn.de)

07.03.2023

HISKP – University Bonn





- Revert to a more FairRoot approach: One geometry, one FairDetector: → EmcDetector
- Encapsulate responsibilities (logic to decode crystal names) into sub objects: → SensorNameIdMap

```
BSEmcDetector *barrel = new BSEmcDetector("EmcBarrel", kTRUE);
barrel->SetGeometryFileName("emc_module12_2018v1_EmcDetector.root");
BSEmcBarrelSensorNameIdMap *barrelMap = new BSEmcBarrelSensorNameIdMap();
barrel->SetIdMap(barrelMap);
barrel->SetSensitiveNames({"Crystal-"});
barrel->SetBranchOutName(BSEmcDataBranchNames::fgMCPointBranchName + "Barrel");
barrel->SetFolderName("EmcBarrel");
barrel->SetPersistency(kTRUE);
fRun->AddModule(barrel);
```

| You have Uncommitted changes

- Very verbose, loose coupling between geometry and SensorNameIdMap-class, sensitive name, etc.
- We can wrap it for user-friendliness

```
BSEmcBarrel::BSEmcBarrel() : BSEmcDetector("EmcBarrel", kTRUE)
{
    this->SetGeometryFileName("emc_module12_2018v1_EmcDetector.root");
    BSEmcBarrelSensorNameIdMap *barrelMap = new BSEmcBarrelSensorNameIdMap();
    this->SetIdMap(barrelMap);
    this->SetSensitiveNames({{"Crystal-"}});
    this->SetBranchOutName(BSEmcDataBranchNames::fgMCPointBranchName + "Barrel");
    this->SetFolderName("EmcBarrel");
    this->SetPersistency(kTRUE);
}
```

You, 2 weeks ago • Fix/Style: Split header-only emc\_restructured cla...

- and use it

```
fRun->AddModule([new BSEmcBarrel()]);
```

- We can do the same for the other detectors and create an EMC setup

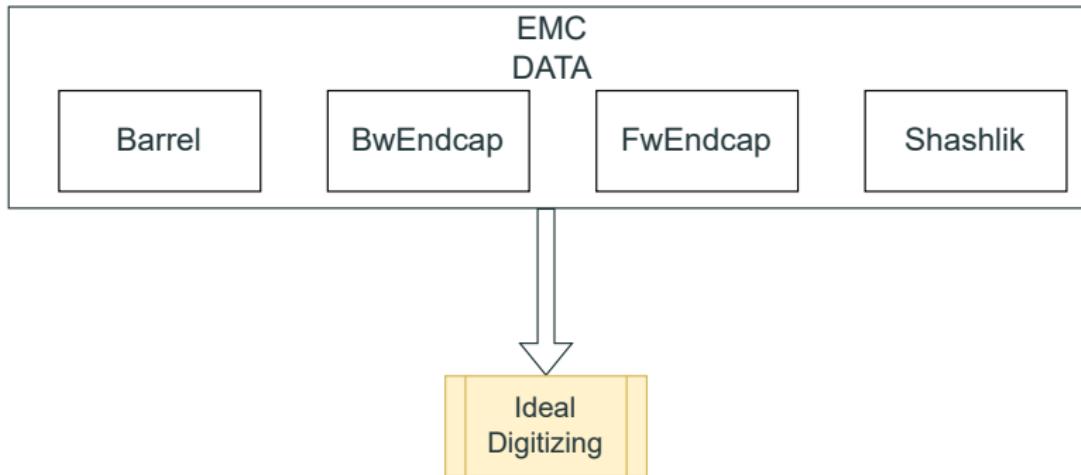
```
void PndEmcDetectors::AddEmcSetup(FairRunSim *t_run)
{
    t_run->AddModule(new BSEmcBarrel());
    t_run->AddModule(new BSEmcBwEndcap());
    t_run->AddModule(new BSEmcFwEndcap());
    t_run->AddModule(new BSEmcShashlik());
}
```

which in turn allows us to include the EMC system with just:

```
PndEmcDetectors::AddEmcSetup(fRun);|
```

So we can reproduce what PndEmc did, but in clean. It is easy to swap a geometry out, and we do not touch running code/logic.

- The EMC System is also treated as a single entity during the data processing, rather than four distinct Subdetectors

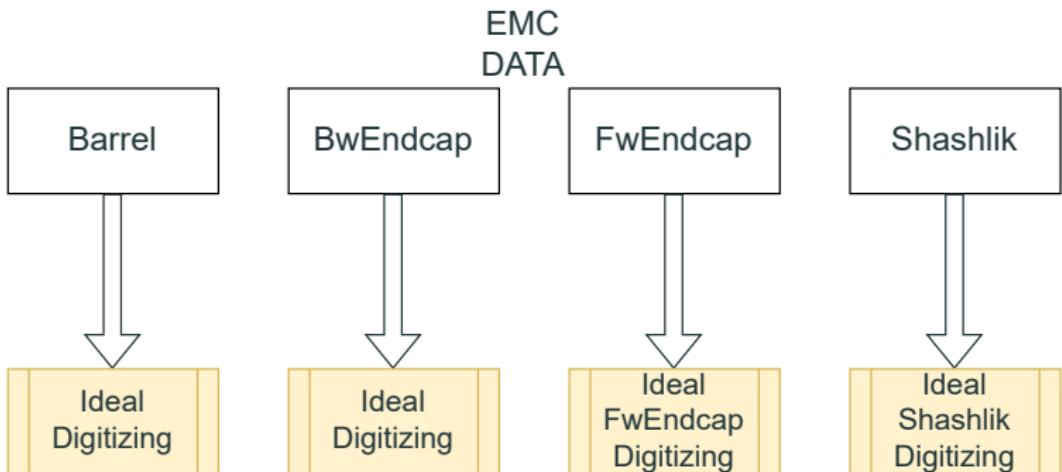


- The EMC System is treated as a single entity, rather than four distinct Subdetectors

```
if (fUseDigitEffectiveSmearing==1){  
    int module = theHit->GetModule();  
    Double_t a,sigma_E;  
    switch (module){  
        case 1: // Barrel  
            a=sqrt(fExcessNoiseFactorAPD/(fnPhotoElectronsPerMeVAPDBarrel*1e3)); // 1e3 is conversion from MeV to GeV  
            sigma_E=sqrt(pow(a/sqrt(energy),2)+pow(fIncoherent_elec_noise_width_GeV_APD/energy,2));  
            break;  
        case 2: // Barrel  
            a=sqrt(fExcessNoiseFactorAPD/(fnPhotoElectronsPerMeVAPDBarrel*1e3)); // 1e3 is conversion from MeV to GeV  
            sigma_E=sqrt(pow(a/sqrt(energy),2)+pow(fIncoherent_elec_noise_width_GeV_APD/energy,2));  
            break;  
        case 3: // ProtoBO  
            a=sqrt(fExcessNoiseFactorAPD/(fnPhotoElectronsPerMeVAPDBarrel*1e3)); // 1e3 is conversion from MeV to GeV  
            sigma_E=sqrt(pow(a/sqrt(energy),2)+pow(fIncoherent_elec_noise_width_GeV_APD/energy,2));  
            break;  
        case 4: // FWD endcap  
            a=sqrt(fExcessNoiseFactorVPT/(fnPhotoElectronsPerMeVVPT*1e3)); // 1e3 is conversion from MeV to GeV  
            sigma_E=sqrt(pow(a/sqrt(energy),2)+pow(fIncoherent_elec_noise_width_GeV_VPT/energy,2));  
            break;  
        case 5: // BWD endcap  
            a=sqrt(fExcessNoiseFactorAPD/(fnPhotoElectronsPerMeVAPDBWD*1e3)); // 1e3 is conversion from MeV to GeV  
            sigma_E=sqrt(pow(a/sqrt(energy),2)+pow(fIncoherent_elec_noise_width_GeV_APD/energy,2));  
            break;  
        case 6: // shashlyk  
            // sigma_E=5.6/E+2.4/sqrt(E)+1.3 (%)  
            sigma_E=sqrt(pow(0.056/energy,2)+pow(0.024/sqrt(energy),2)+0.013);  
            break;  
        default:  
            std::cout<<"PndEmcMakeDigi::Exec - Unknown module number in EMC digitization"<<std::endl;  
            abort();  
    }  
}
```

- It forces us to specify specific Subdetector behaviour on low level. Messy!

# Instead - one reconstruction chain per Subdetector



- Instead: Parameterize! And keep Subdetector data separate  
→ each Subdetector has its own data-processing chain
- New/different functionality: new Processing-Class and only add it, where it is needed

# Instead - one reconstruction chain per Subdetector

```
auto *barreldigitizerChain = new BSEmcDigitizerTask("Barrel");
barreldigitizerChain->SetDigiBranchName(BSEmcDataBranchNames::fgDigiBranchName + "Barrel");
barreldigitizerChain->AddProcess(new BSEmcIdealDigitizationProcess());
barreldigitizerChain->AddProcess(new BSEmcApplyCalibrationProcess());
barreldigitizerChain->AddProcess(new BSEmcDigiDCSetterProcess());
fRun->AddTask(barreldigitizerChain);
```

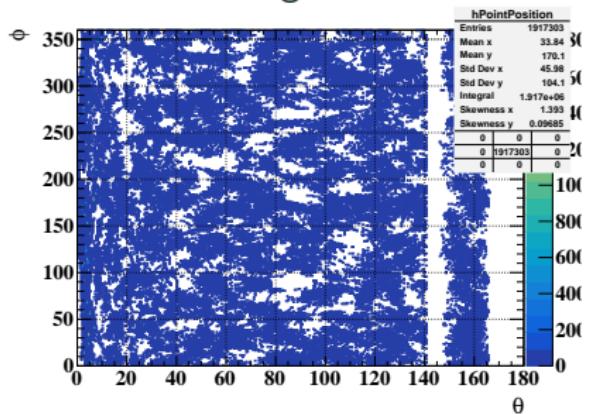
```
auto *fwendcapdigitizerChain = new BSEmcDigitizerTask("FwEndcap");
fwendcapdigitizerChain->SetDigiBranchName(BSEmcDataBranchNames::fgDigiBranchName + "FwEndcap");
auto *fwendcapdigitizer = new BSEmcFwEndcapIdealDigitizerProcess();
fwendcapdigitizer->SetVpttIds(BSEmcFwEndcapVpttIds::GetVpttIds());
fwendcapdigitizerChain->AddProcess(fwendcapdigitizer);
fwendcapdigitizerChain->AddProcess(new BSEmcApplyCalibrationProcess());
fwendcapdigitizerChain->AddProcess(new BSEmcDigiDCSetterProcess());
fRun->AddTask(fwendcapdigitizerChain);
```

- Instead: Parameterize! And keep Subdetector data separate  
→ each Subdetector has its own data-processing chain
- New/different functionality: new Processing-Class and only add it, where it is needed

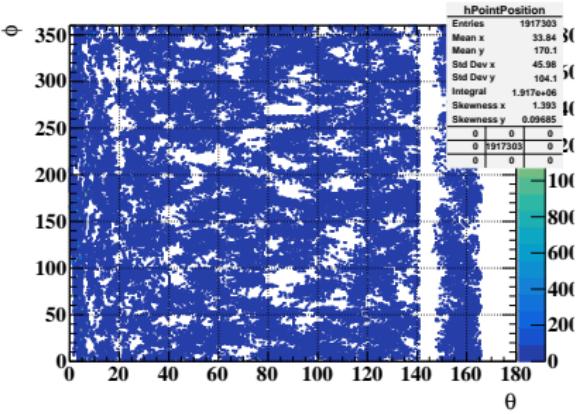
- Today I want to briefly show the differences of the original (one processing chain) and restructured (split processing chains) versions by comparing (mostly just) the energy distributions at different/distinct points along the simulation/reconstruction-chain for 1000 1 GeV/c photons.

# Simulation - MCPPoints Position distribution

Original



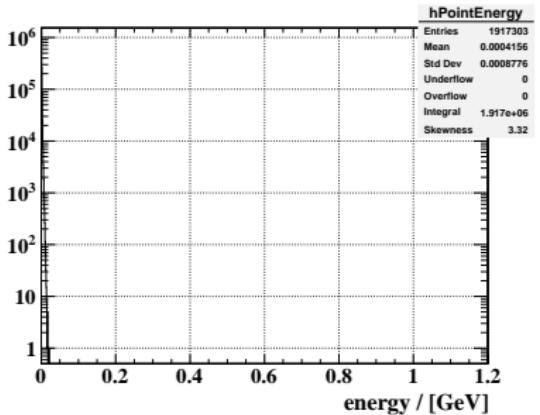
Restructured



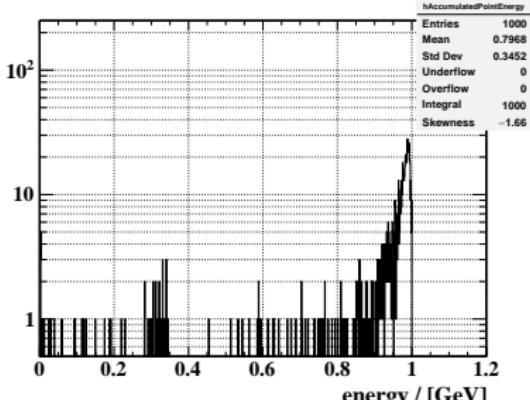
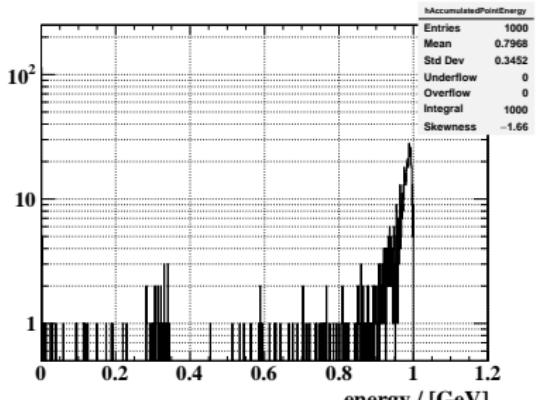
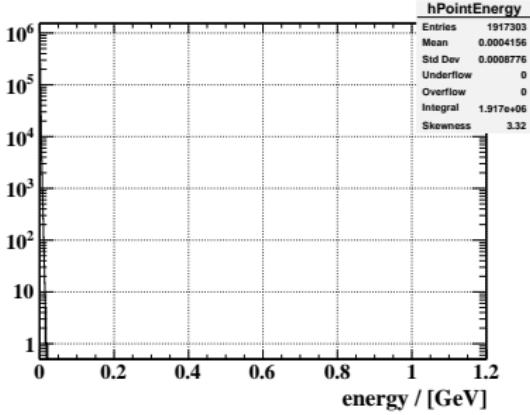
Identical! We reproduced the same positions!

# Simulation - MCPPoints Energy distributions

Original

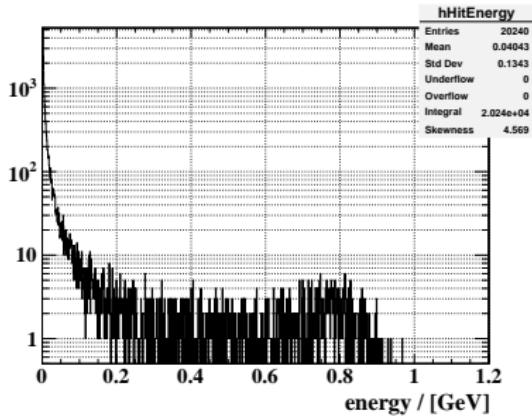


Restructured

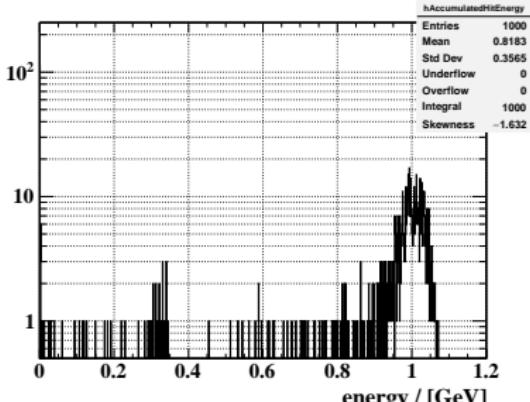
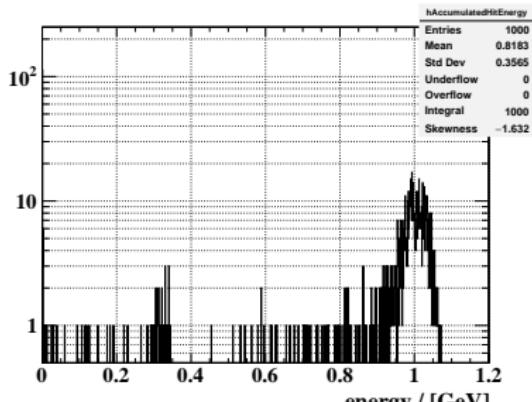
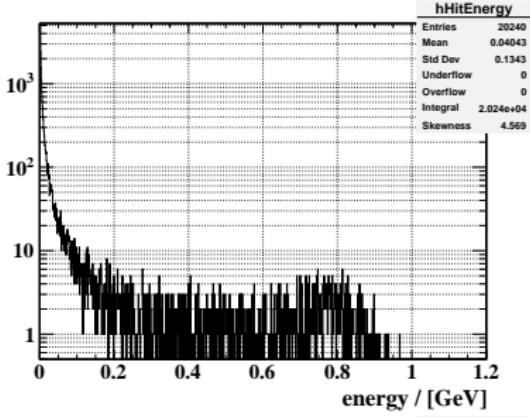


# Simulation - MCHits Energy distribution

Original



Restructured



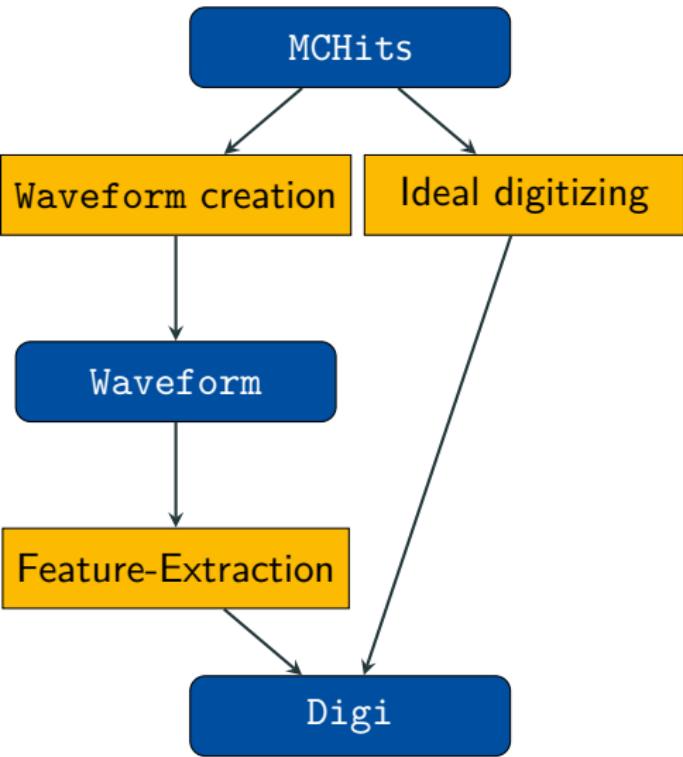
- There are no differences between the outcome of the original and the restructured versions of the EMC!

Performance “indicators” for the 1000 1 GeV photon simulation:

measure	original	restructured
init time	10 s	90 s
total time	236 s	320 s
max memory	1000 MB	1200 MB

The Shashlik-geometry-version is the cause for the init time increase! It needs fixing, not the restructured EMC code.

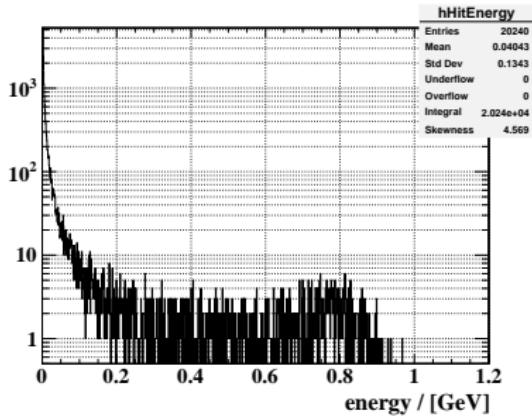
# Digitizing - Approaches



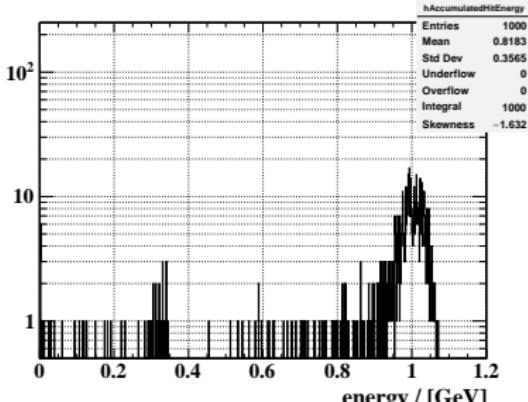
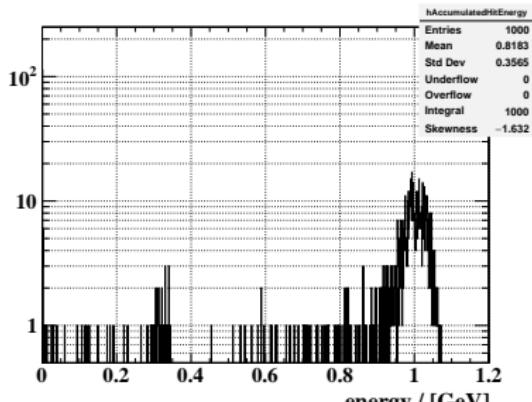
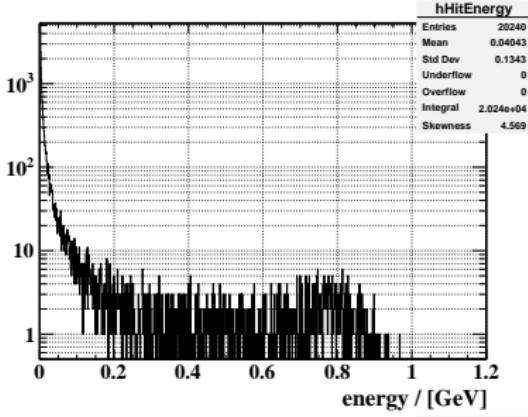
- we will use the (Ideal digitizing) naive smearing approach
- separation of EMC Subdetectors – processing order of MCHits will change
- smearing of the MCHits using random numbers will cause differences

# Digitizing - input MCHits

Original

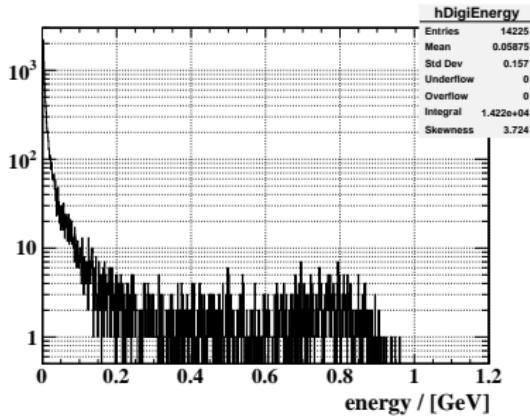


Restructured

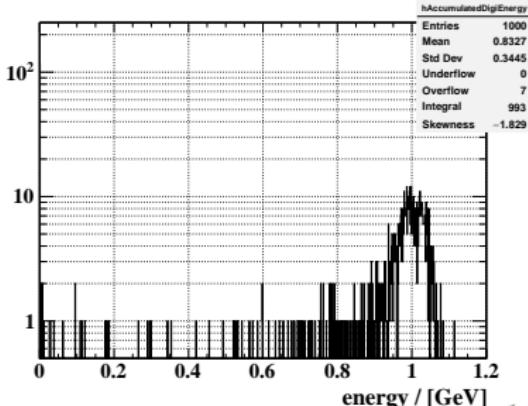
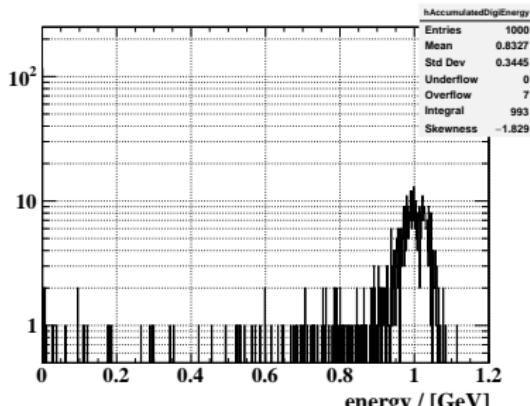
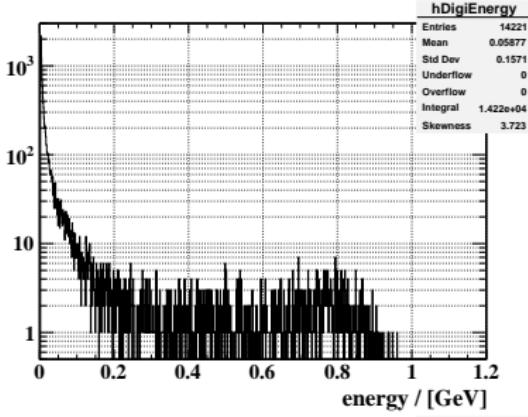


# Digitizing - resulting Digits

Original



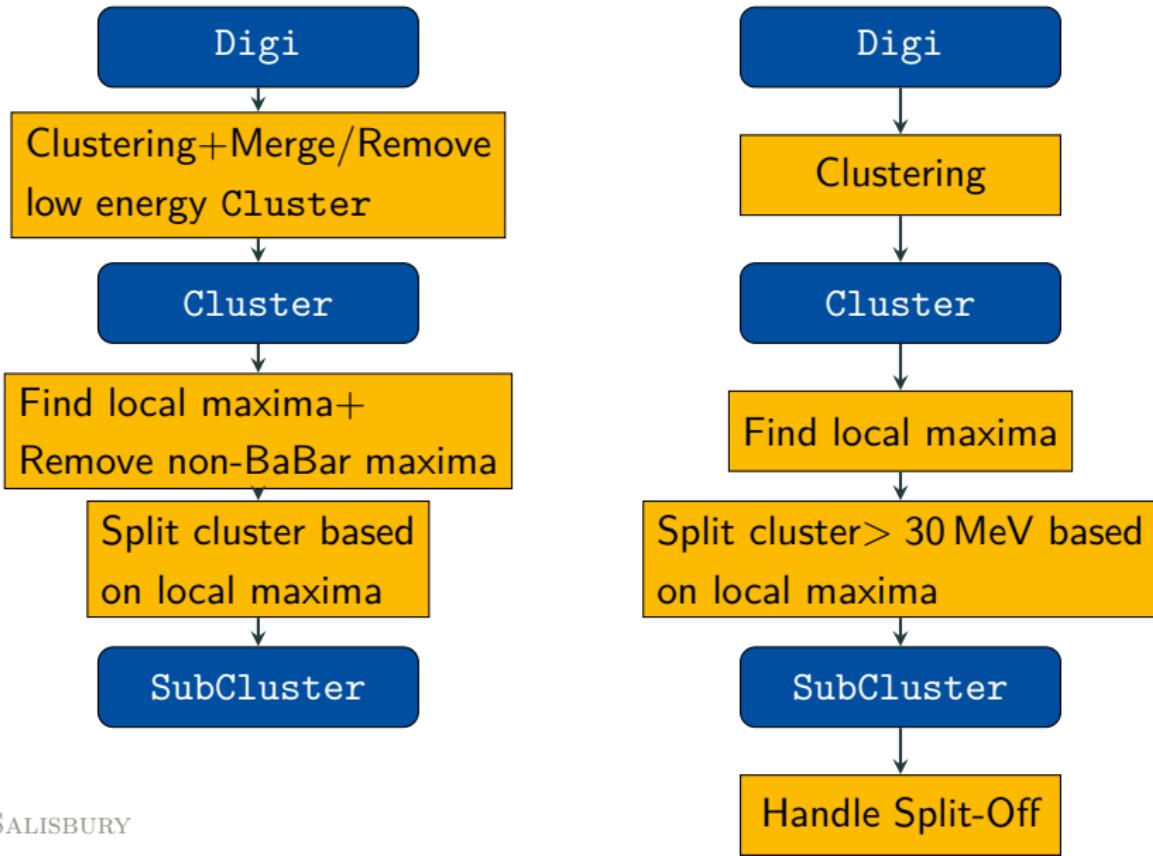
Restructured



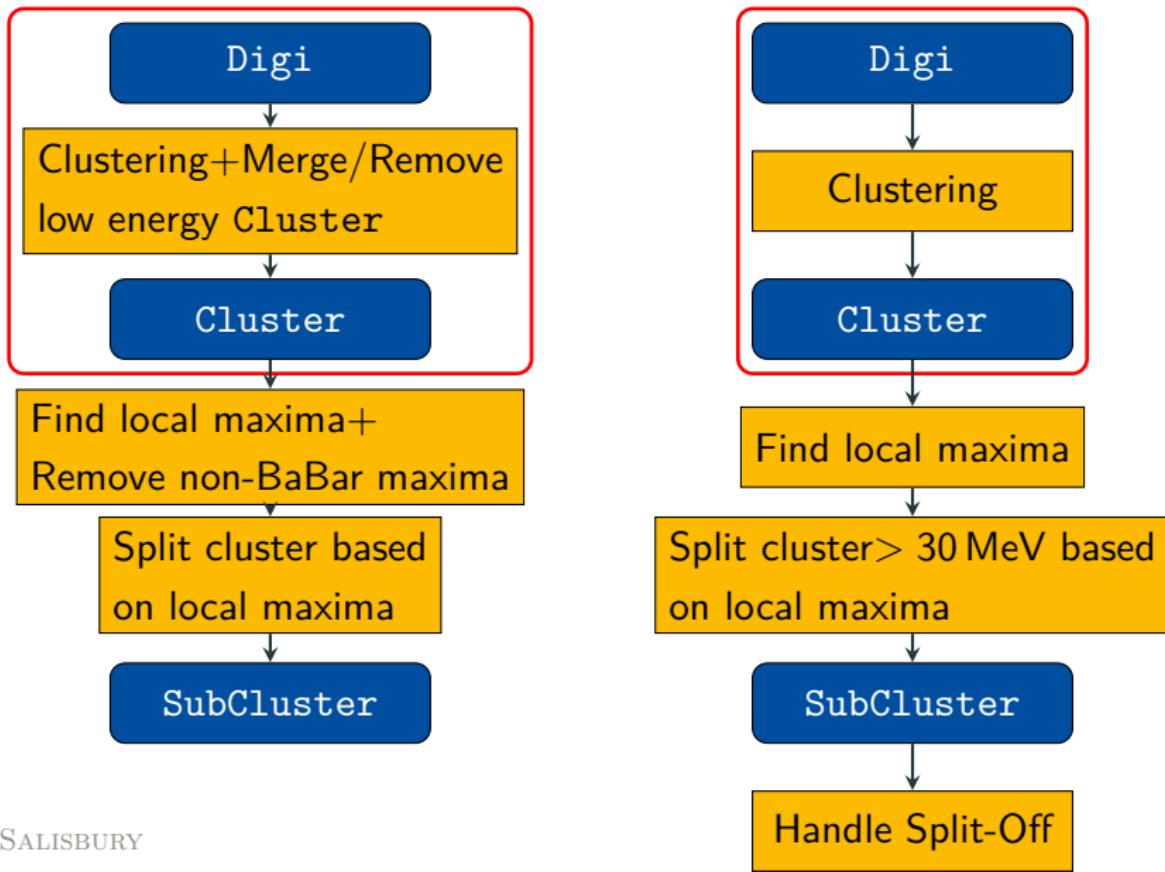
- Differences negligible and merely due to different data sequence in combination with the random number usage (for the ideal case)
- Goal is to use the P. Mahlberg's Waveform-Digitizing-“Package” in the future.
- Even for the Waveform-Approach we do not expect any other differences than those due to different processing sequence.  
The code changes will be boilerplate-code-changes!

- So far, the processing did not differ between the two EMC code versions
- From now on, we introduce changes of not just technical nature, but that actually impact the reconstruction

# Reconstruction-chain



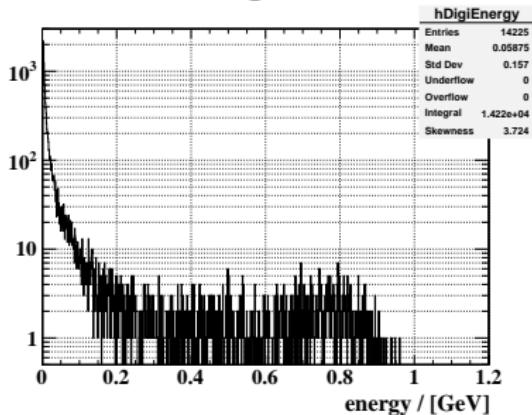
# Reconstruction-chain



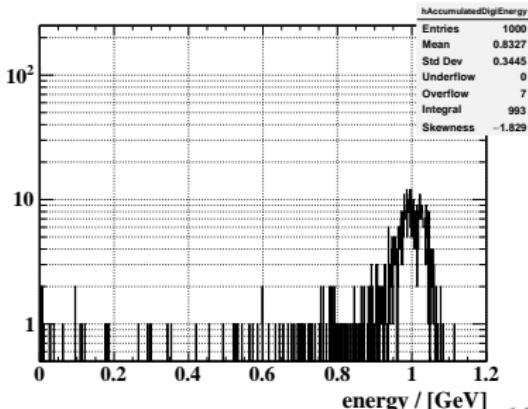
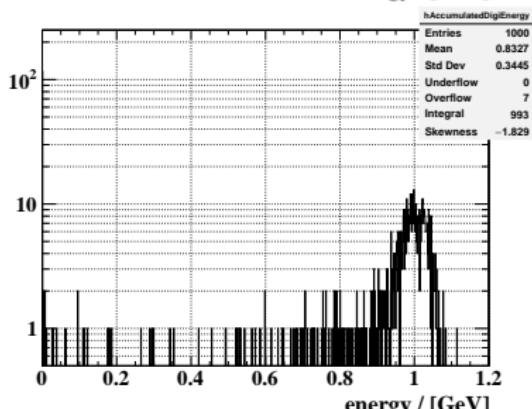
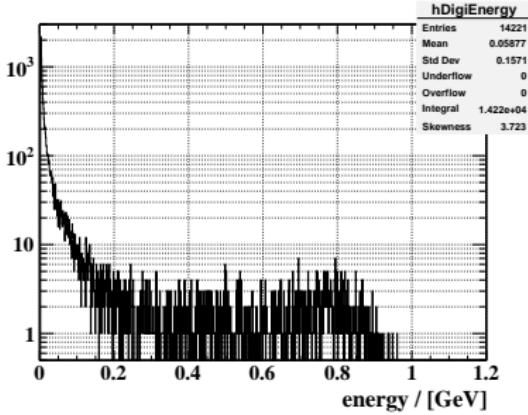
- The original version groups Digits with more energy than 3 MeV into Cluster, but then merges/removes low-energetic Cluster into other Cluster. This is a form of Split-Off handling. However, it introduces problems with the position reconstruction.
- The restructured version also groups Digits with more energy than 3 MeV into Cluster, but keeps every created Cluster at this point. Split-Off handling is done at a later stage. Low energy Cluster are dropped during the SubClustering.

# Clustering - input Digits

Original

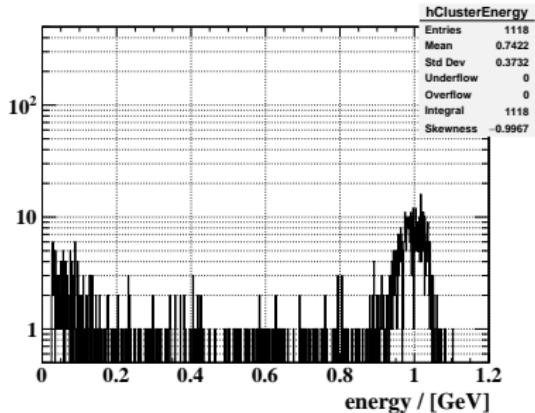


Restructured

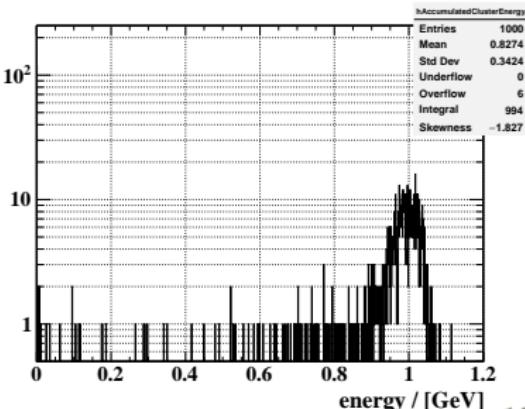
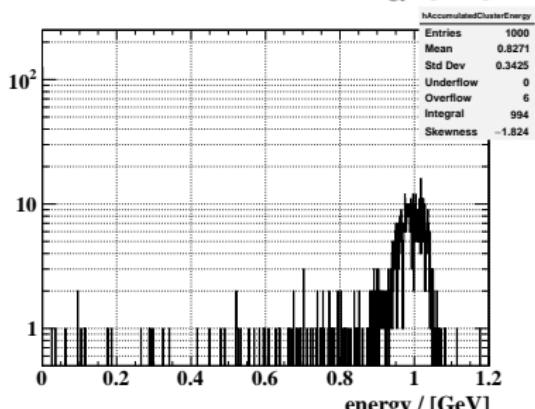
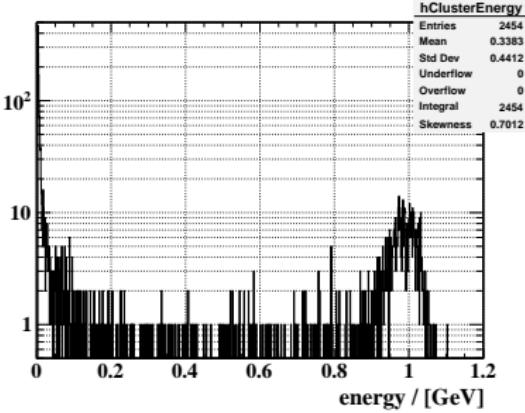


# Clustering - resulting Cluster

Original

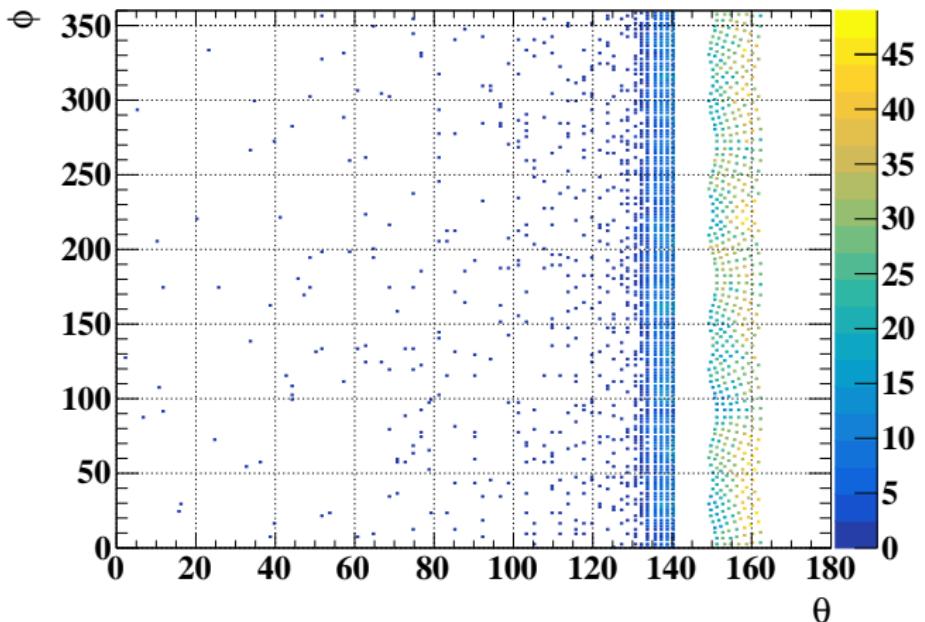


Restructured



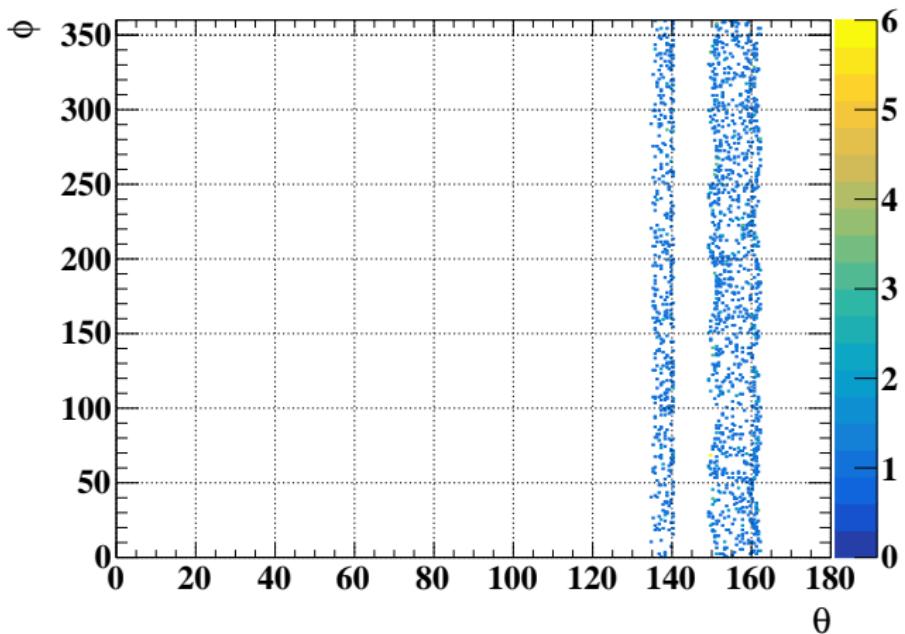
# Illustration of Cluster-Merging artefacts

Digi positions



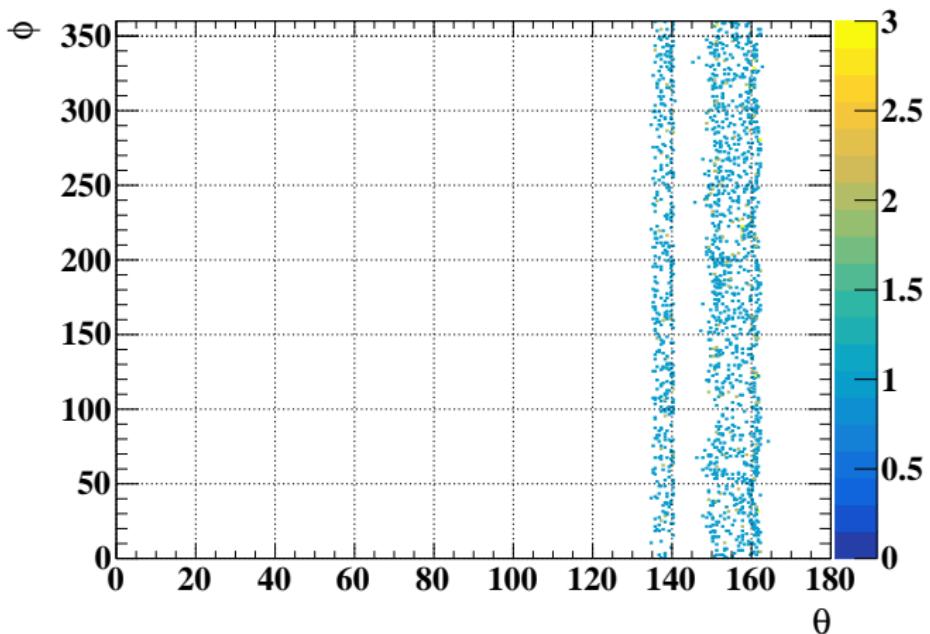
# Illustration of Cluster-Merging artefacts

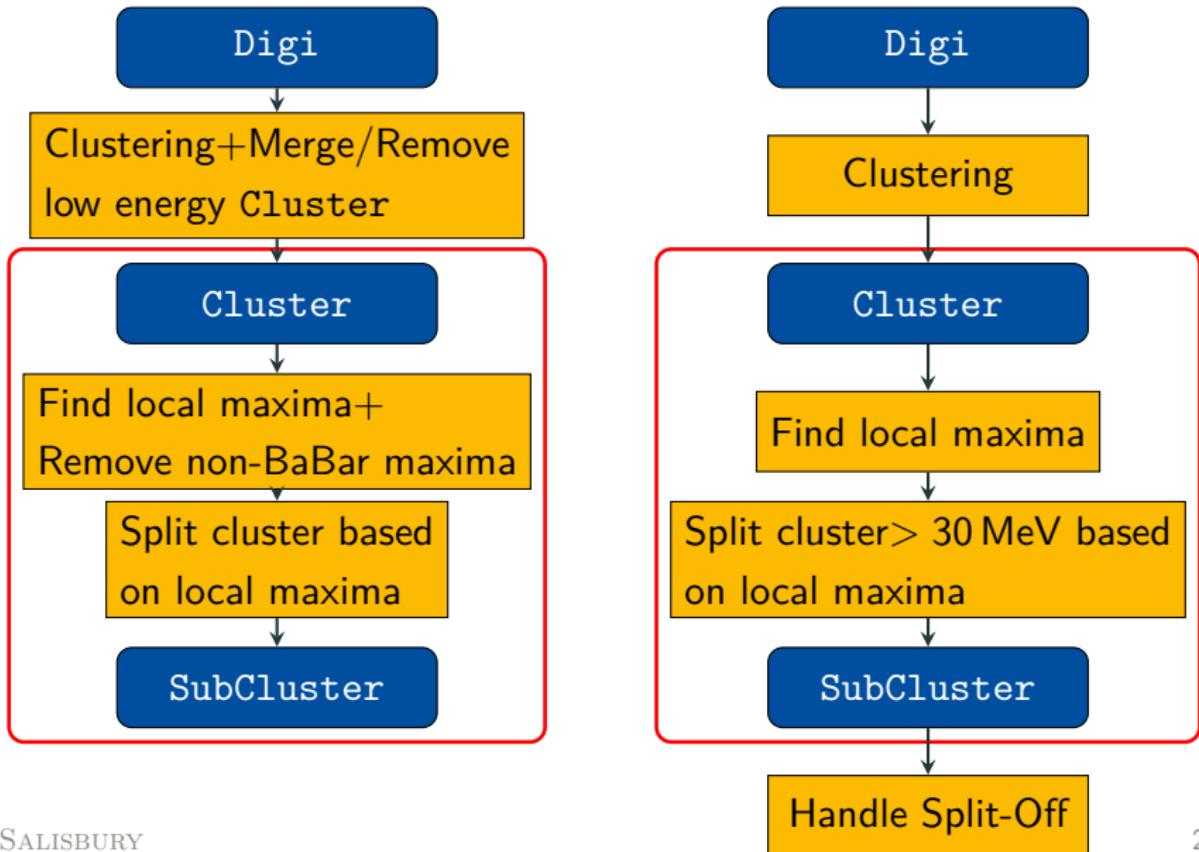
Cluster positions without Cluster-Merging



# Illustration of Cluster-Merging artefacts

Cluster positions with Cluster-Merging





```
// Environment:  
// Software developed for the BaBar Detector at the SLAC B-Factory.  
// Adapted for the PANDA experiment at GSI  
//
```

```
// Author List:  
//      Phil Strother  
//      Helmut Schmuecker  
//
```

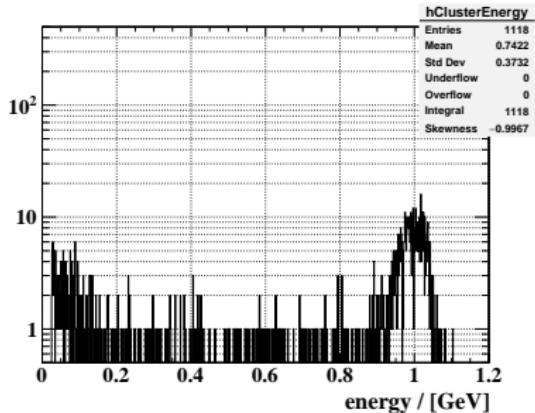
```
// Copyright Information:  
// Copyright (C) 1997           Imperial College  
// Modified:  
// M. Babai  
//-----
```



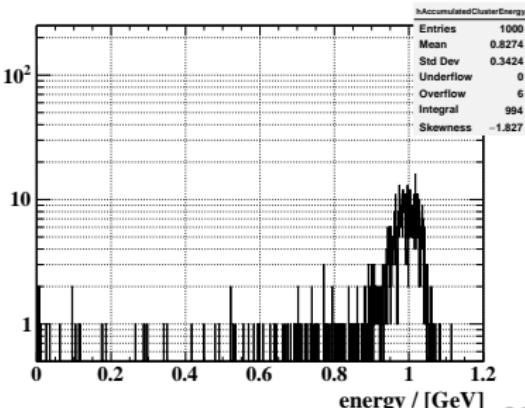
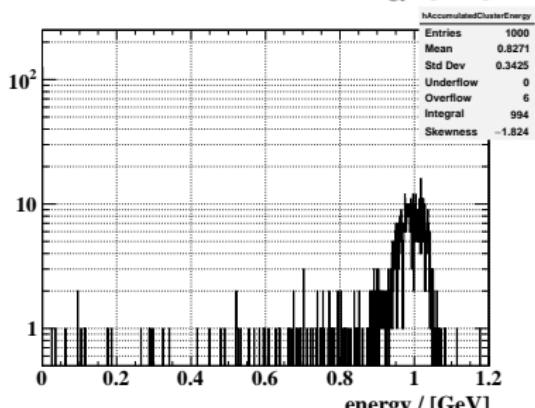
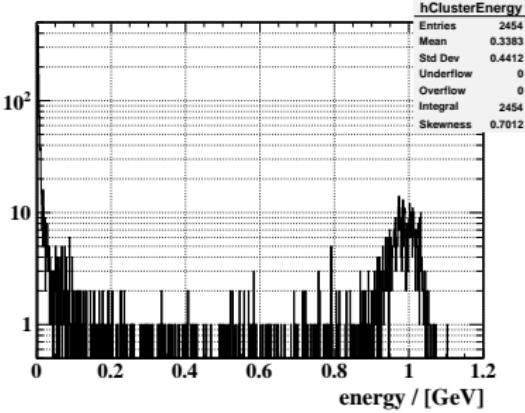
- The original version uses a Split-Off rejection in its Finding of local Maxima. I think it may come from the BaBar framework (this code has hardly been touched or tuned since the beginning of the git-history (as far as I can tell)).
- Every Cluster is, however, turned into at least one SubCluster.
- The restructured version removes the “BaBar”-part. The Subclustering now applies a cut (to reject noise energy) on the Cluster energy before creating SubCluster.

# SubClustering - input Cluster

Original

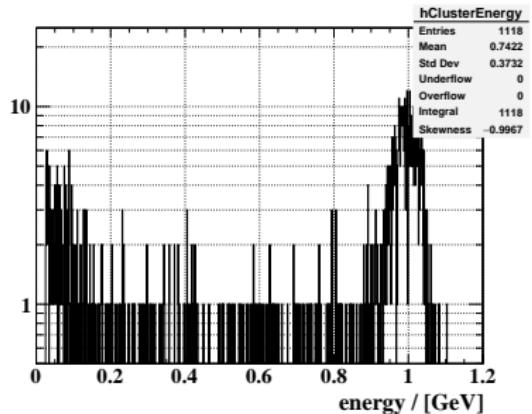


Restructured

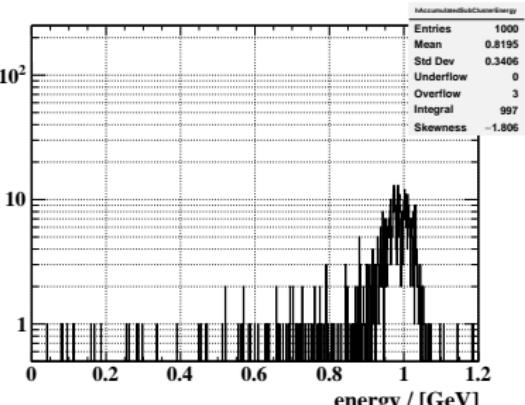
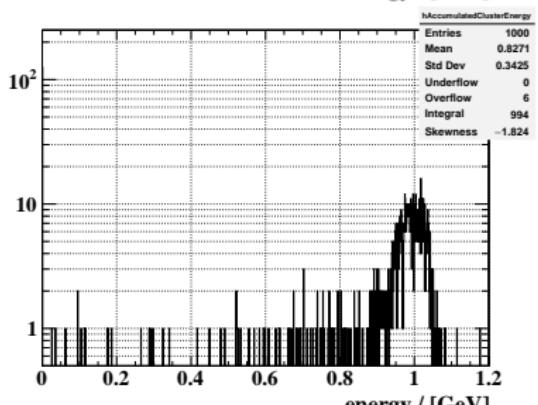
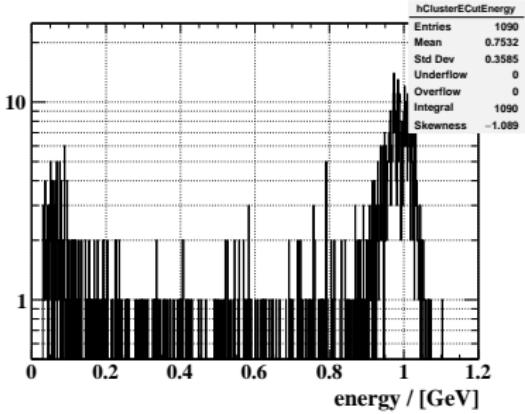


# SubClustering - input Cluster (after E-Cut)

Original

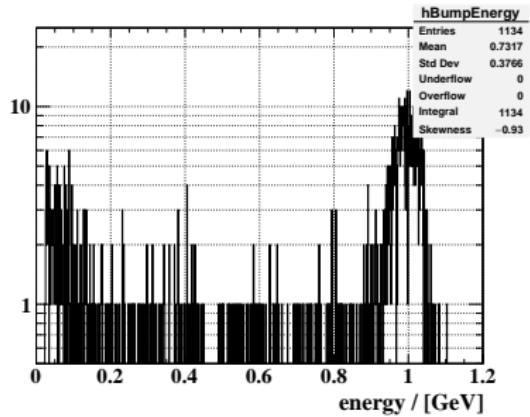


Restructured

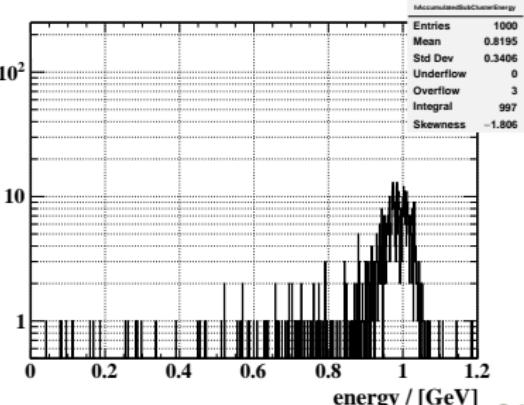
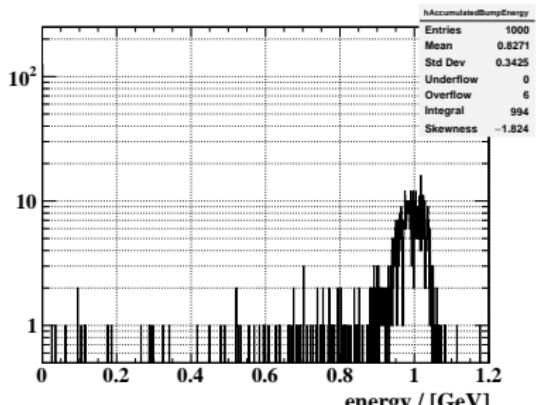
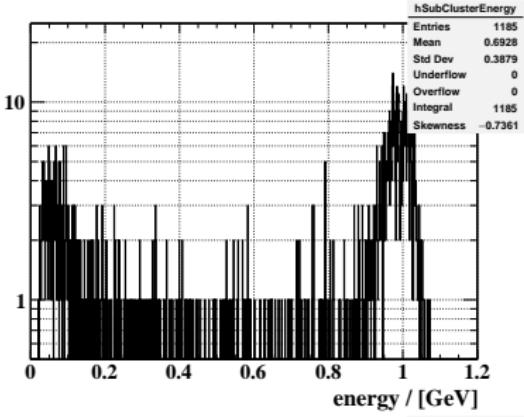


# SubClustering - resulting SubCluster

Original

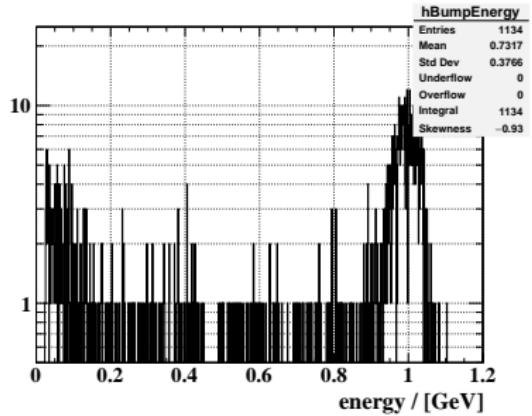


Restructured

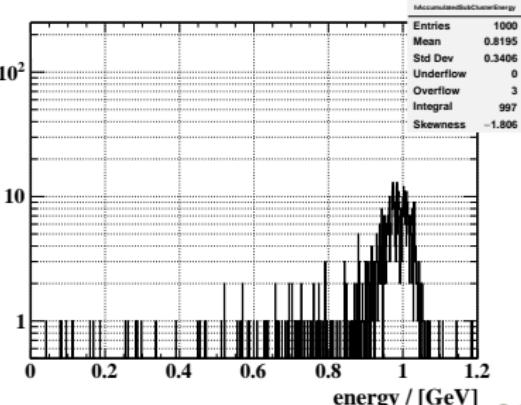
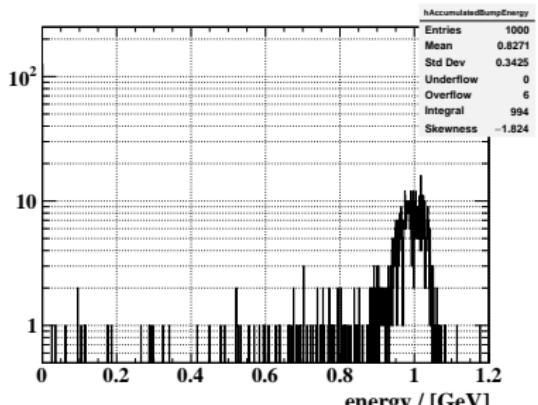
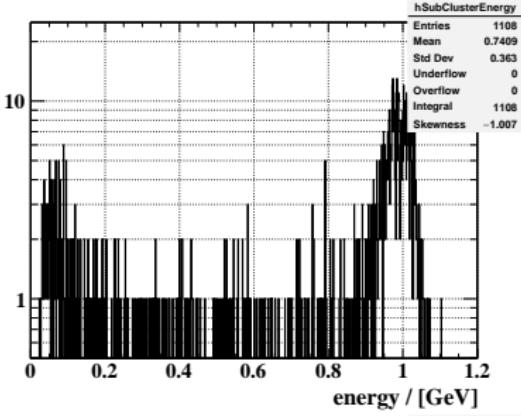


# SubClustering - resulting SubCluster (with BaBar)

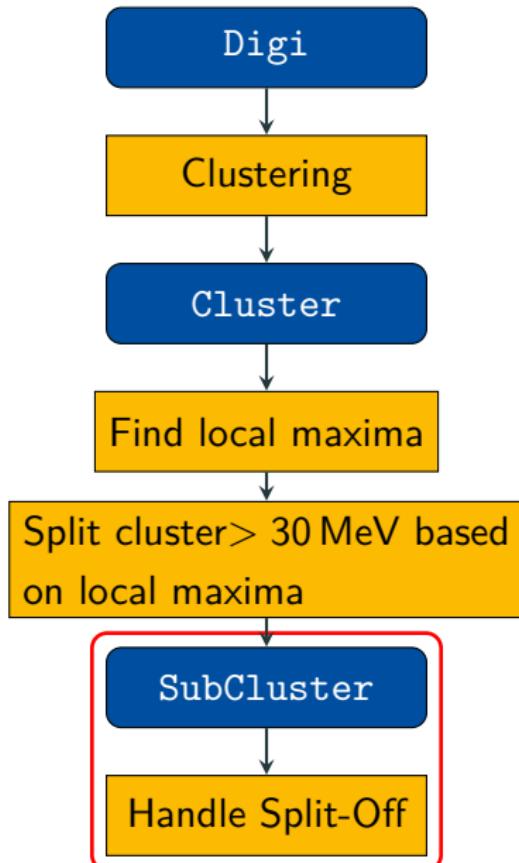
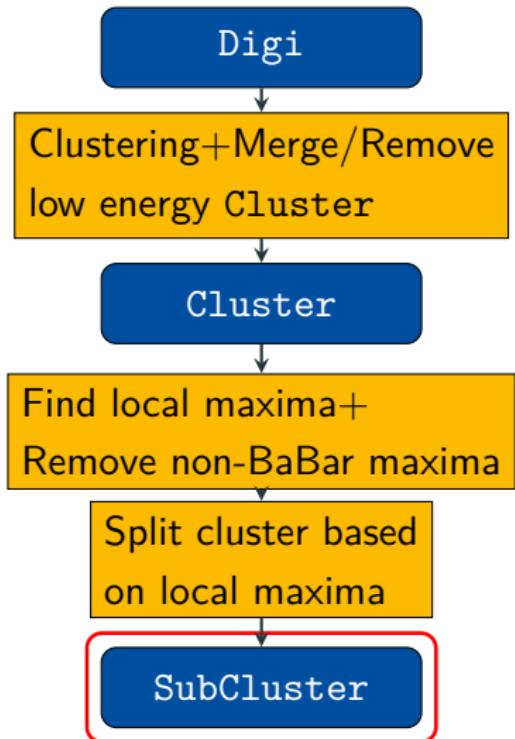
Original



Restructured



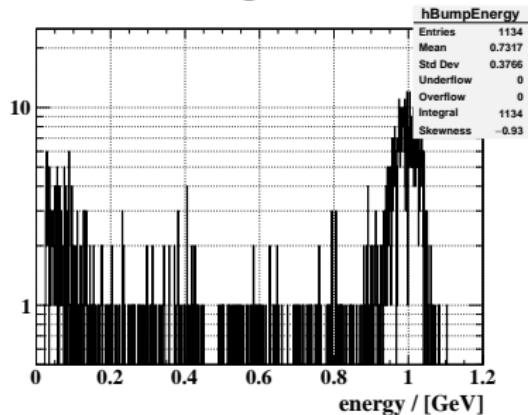
# Reconstruction-chain



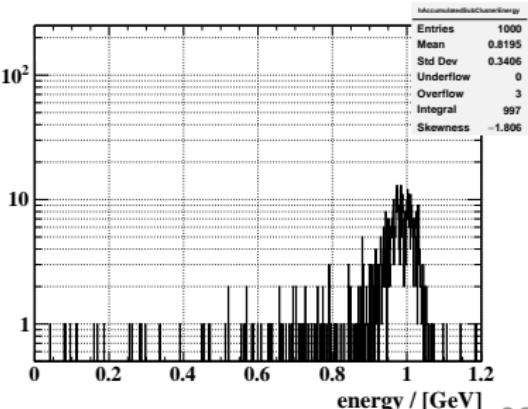
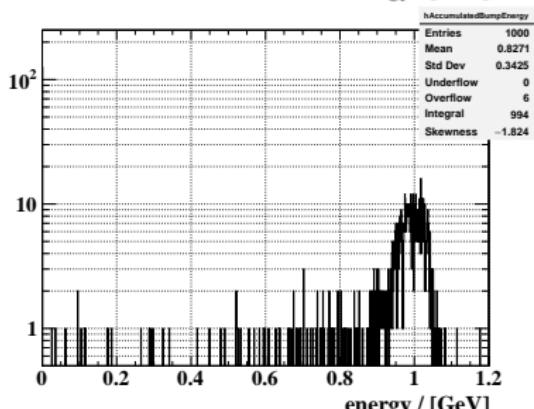
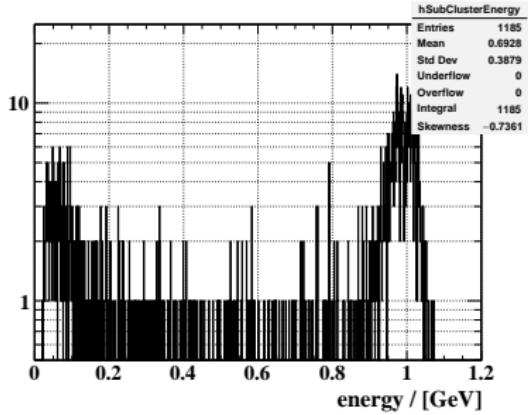
- The original version relies in addition to the Cluster-Merging and Removing on a BaBar-Maximum-Removing during the Maximum finding. This BaBar approach has never been tuned (as far as I know) for any of our EMC Subdetectors.
- The restructured version provides Jonas' Split-Off handling for Barrel, BwEndcap and FwEndcap, but not for the Shashlik.

# Split-Off handling - SubCluster before

Original

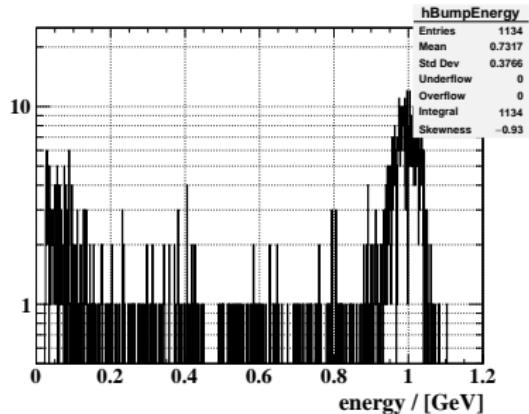


Restructured

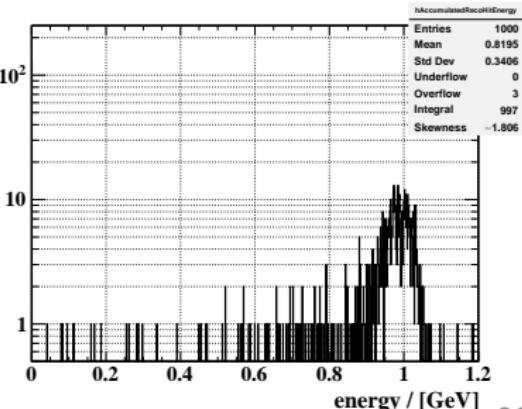
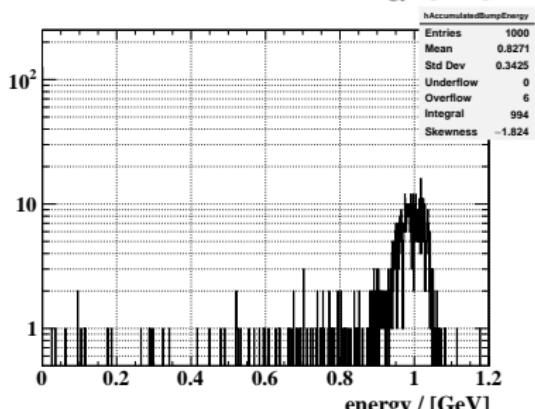
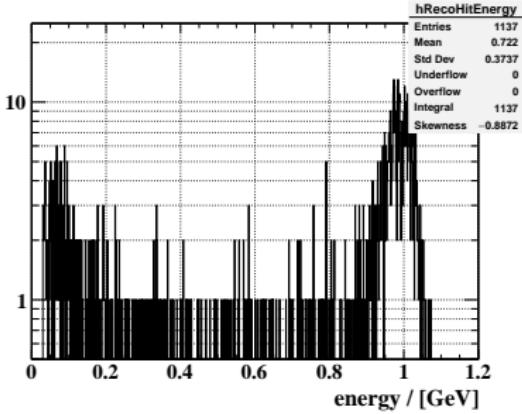


# Split-Off handling - SubCluster after

Original

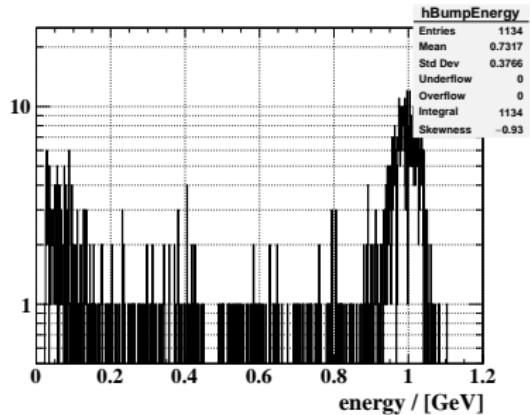


Restructured

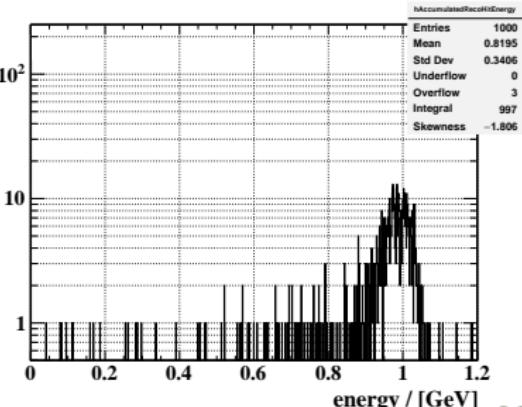
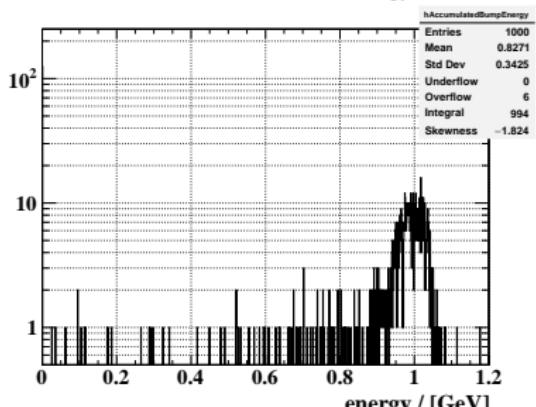
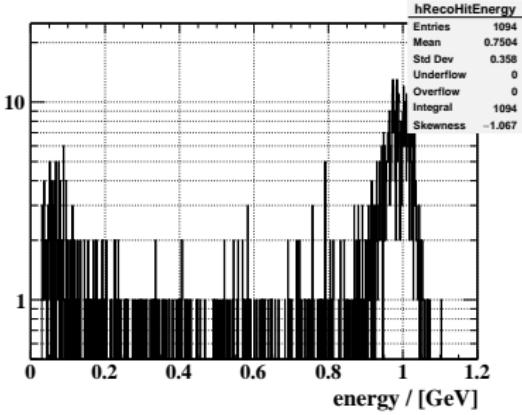


# Split-Off handling and BaBar for Shashlik

Original



Restructured



Performance “indicators” for the 1000 1 GeV photon local reconstruction:

measure	original	restructured
total time	~ 35 s	~ 11 s
max memory	~ 1000 MB	~ 1010 MB

The output of the local Reconstruction of all detectors (Tracking, DIRC, EMC, ...) is then combined to PidCandidates (neutral and charged).

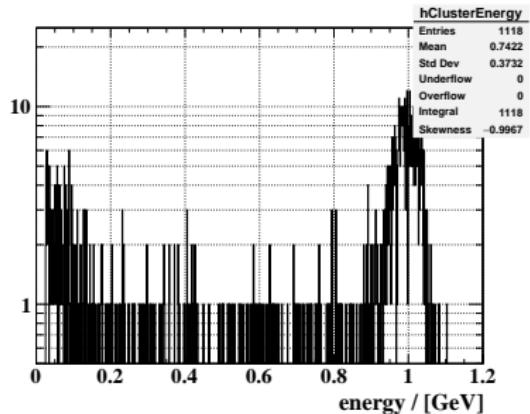
The original and restructured EMC version slightly differ on what they provide:

- Original: Provides Cluster as input to the PID
- Restructured: Provides SubCluster after Split-Off handling as input to the PID

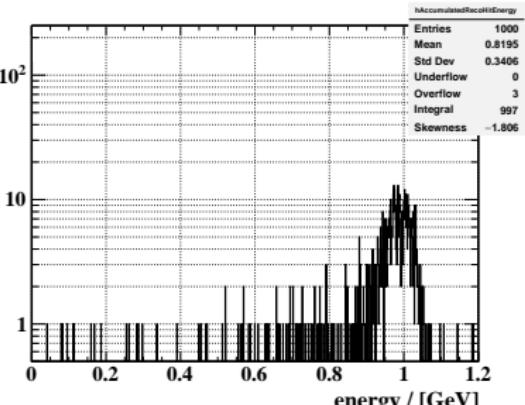
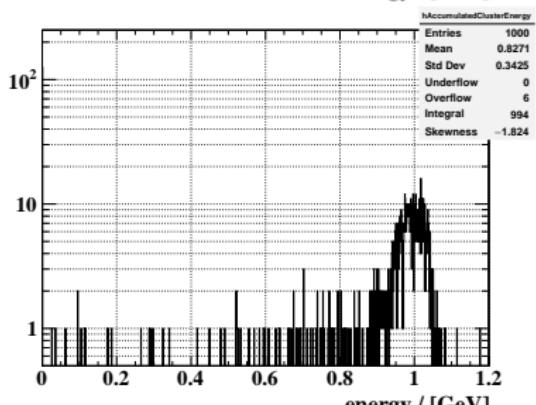
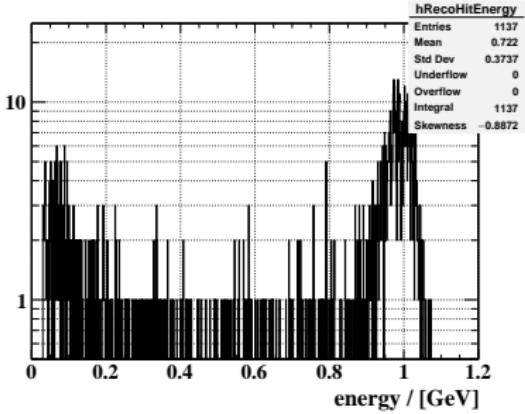
**IMPORTANT!** The original would also have used SubCluster for the neutral PidCandidates in the past, but an error may have “recently” sneaked into the PidCorrelator

# Particle Identification - EMC Input

Original

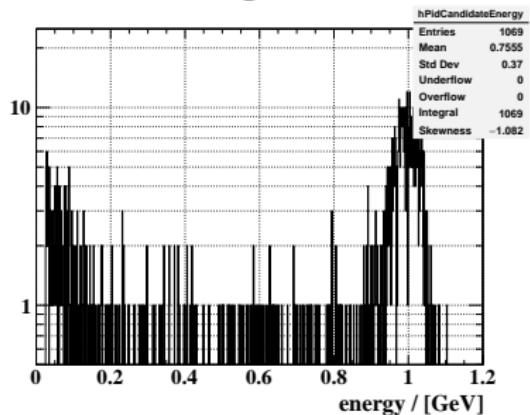


Restructured

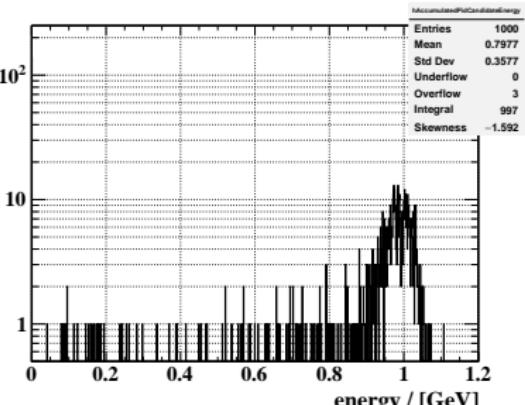
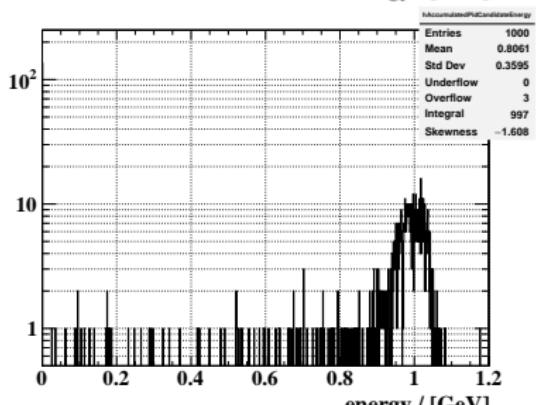
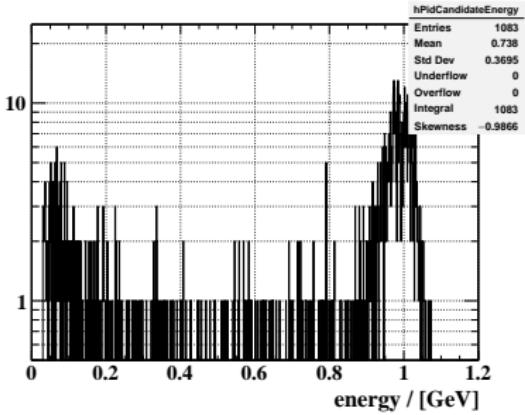


# Particle Identification - Neutral Candidates

Original



Restructured

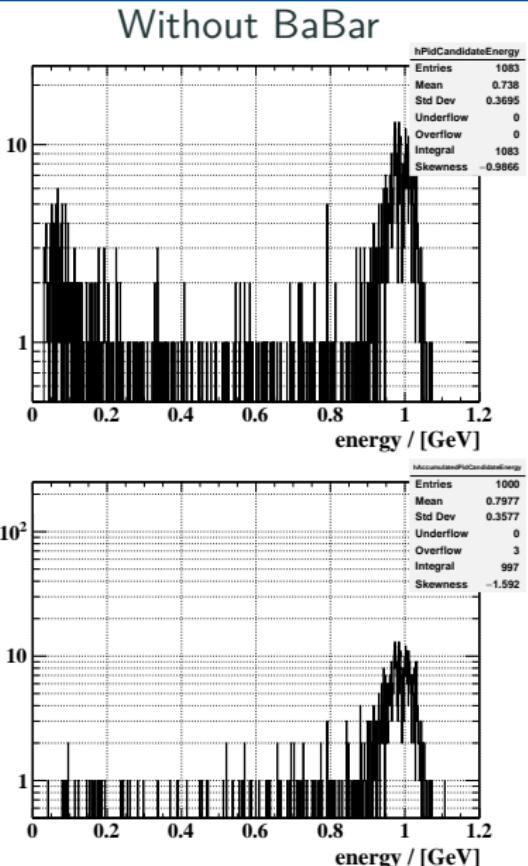


- Simulation results identical! Digitizing results comparable!
- Local reconstruction results differ due to different Split-Off handling (Cluster-Merging/Maximum-Finding) and my assumption of what a valid Particle energy deposition would be (Cluster with more energy than  $x$  MeV + Maximum with more than  $y$  MeV)
- We supply PID with different EMC data (at least at the moment)!

- Problem is: We do not think that we should do the Cluster-Merging. However, this (beyond the already differing Digi-distribution) reproducing identical results impossible.
- Biggest problem is the Shashlik situation: restructured version does not provide Split-Off handling! We can use the BaBar-approach, but we do not know what it is doing.

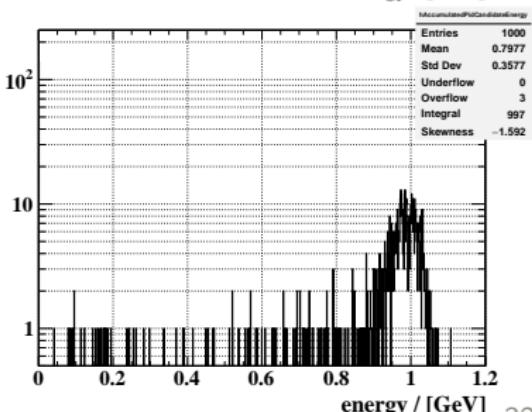
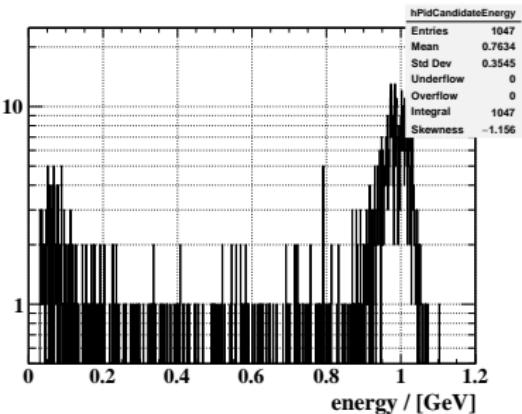
# Removing BaBar-Max-Removing

- neutral PidCandidates final distribution for restructured EMC version



# Removing BaBar-Max-Removing

## Only Shashlik with BaBar-Max-Removing



# Removing BaBar-Max-Removing

## All after BaBar-Max-Removing

- Shashlik has no new Split-Off handling.  
Introducing BaBar-Max-Removing has major effect.
- Using BaBar-Max-Removing on rest has hardly effect → equivalent to Jonas new Split-Off handling (only: we understand Jonas' Split-Off handling, while BaBar-Max-Removing is untuned)

