SEPTEMBER 15, 2021

BARREL ECAL TUTORIAL

JIHEE KIM
Argonne National Laboratory
jihee.kim@anl.gov

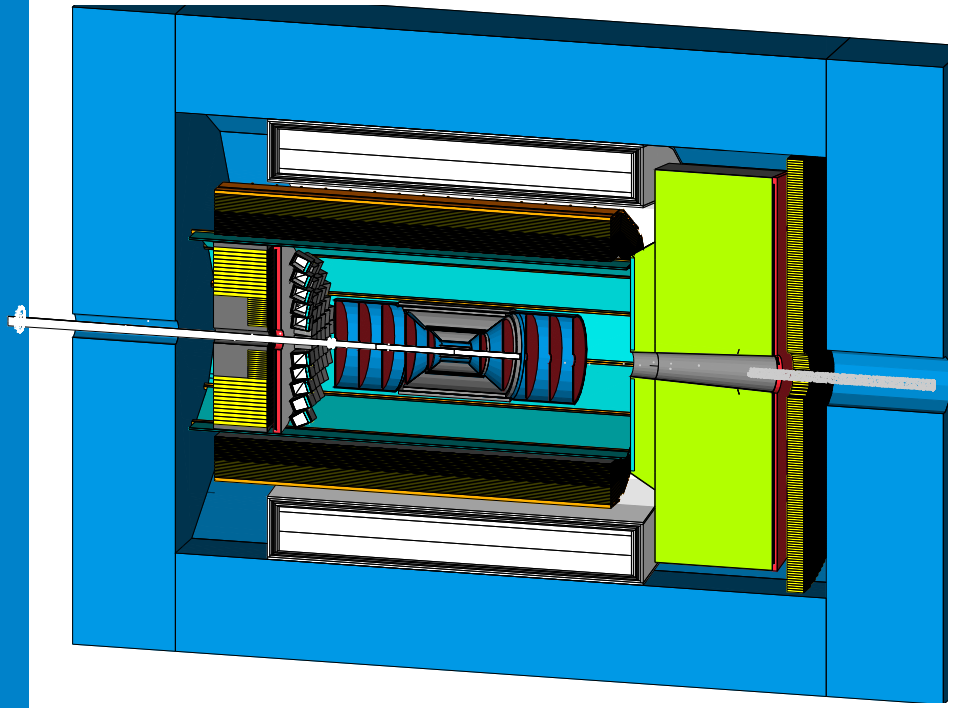
OUTLINE

Barrel Emcal Analysis Tutorial

- Step by Step
  - Step 1: Detector geometry
  - Step 2: Generate input particles
  - Step 3: Run Geant4 simulation
  - Step 4: Info from Geant4 simulation
  - Step 5: Run reconstruction
  - Step 6: Analyze output

STEP 1: CHECK DETECTOR GEOMETRY

ATHENA Central Detector



The Compact detector description files:

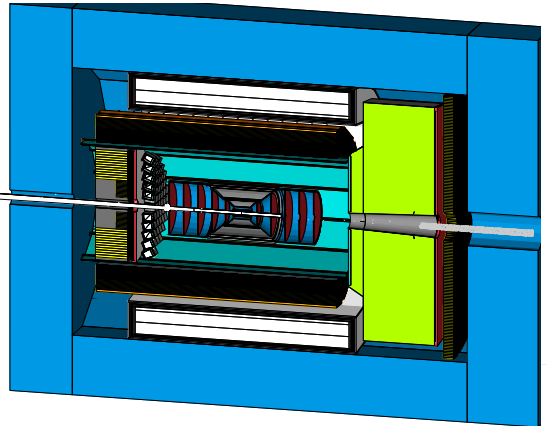
- athena.xml
  - definition.xml
  - beampipe.xml
  - solenoid.xml
  - tracker.xml
  - ecal.xml: endcaps and barrel
  - hcal.xml
  - far_foward_detectors.xml
- material.xml
- elements.xml
- display.xml

STEP 1: CHECK DETECTOR GEOMETRY

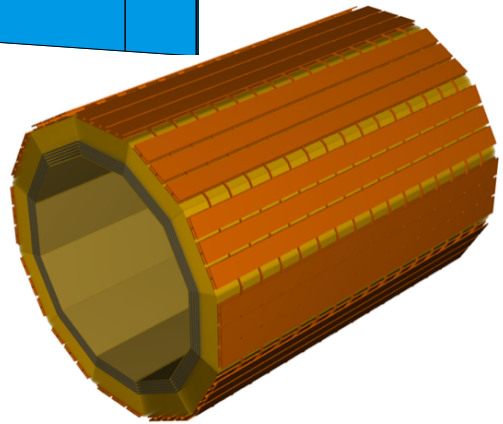
ATHENA Central Detector

- In athena.xml, either you can use whole ATHENA detector or just one subdetector

```
<documentation level="10">
### PID detectors
</documentation>
<include ref="compact/pid_config_acadia.xml" />

<documentation level="10">
## Central calorimetry
</documentation>
<include ref="compact/ecal.xml"/>
<include ref="compact/hcal.xml"/>


<documentation level="11">
## Far foward detectors
</documentation>
<include ref="ip6/B0_tracker.xml"/>
<include ref="ip6/B0_preshower.xml"/>
<include ref="ip6/far_forward_offM_tracker.xml"/>
<include ref="ip6/far_forward_detectors.xml"/>
<include ref="ip6/roman_pots_eRD24_design.xml"/>
```



Whole Central detector



Barrel EM calorimeter

STEP 1: CHECK DETECTOR GEOMETRY

ATHENA Central Detector

- In definition.xml and ecal_barrel_interlayers.xml, detector parameters are parameterized. For example, given space and layer thicknesses, the number of layers is computed.

```
<constant name="EcalBarrel_ImagingLayerThickness"
  value="EcalBarrel_SiliconThickness
  + EcalBarrel_ElectronicsThickness
  + EcalBarrel_CopperThickness
  + EcalBarrel_KaptonThickness
  + EcalBarrel_EpoxyThickness
  + EcalBarrel_CarbonThickness"/>
<constant name="EcalBarrel_ScFiLayerThickness"
  value="EcalBarrel_RadiatorThickness
  + EcalBarrel_CarbonThickness
  + EcalBarrel_LayerSpacing"/>

<constant name="EcalBarrelImagingLayers_nMax"  value="9"/>
<constant name="EcalBarrelImagingLayers_num"
    value="min(EcalBarrelImagingLayers_nMax,
          floor((EcalBarrel_AvailThickness-EcalBarrel_ImagingLayerThickness)/
                (EcalBarrel_ImagingLayerThickness + EcalBarrel_ScFiLayerThickness + EcalBarrel_SpaceBetween)))"/>

<constant name="EcalBarrel_ImagingPartThickness"
    value="(EcalBarrelImagingLayers_num-1)*(EcalBarrel_ImagingLayerThickness + EcalBarrel_ScFiLayerThickness + EcalBarrel_SpaceBetween)
          + EcalBarrel_ImagingLayerThickness + EcalBarrel_SpaceBetween"/>
<constant name="EcalBarrel_ScFiPartThickness_max"
    value="max(0, EcalBarrel_AvailThickness-EcalBarrel_ImagingPartThickness)"/>
<constant name="EcalBarrel_ScFiPartThickness"
    value="min(EcalBarrel_ScFiPartThickness_max, EcalBarrel_FiberZSpacing*13*12)"/>
<constant name="EcalBarrel_SensitiveLayers_rmax"
    value="EcalBarrel_rmin + EcalBarrel_ImagingPartThickness + EcalBarrel_ScFiPartThickness"/>
```

```
<constant name="EcalBarrel_Support_thickness"    value="5*cm"/>
<constant name="EcalBarrel_SiliconThickness"     value="500*um"/>
<constant name="EcalBarrel_ElectronicsThickness" value="150*um"/>
<constant name="EcalBarrel_CopperThickness"      value="100*um"/>
<constant name="EcalBarrel_KaptonThickness"      value="200*um"/>
<constant name="EcalBarrel_EpoxyThickness"       value="100*um"/>
<constant name="EcalBarrel_CarbonThickness"      value="0.5*mm"/>
<constant name="EcalBarrel_CarbonSpacerWidth"    value="4*mm"/>
<constant name="EcalBarrel_LayerSpacing"         value="10.0*mm"/>
<constant name="EcalBarrel_FiberRadius"          value="0.5*mm"/>
<constant name="EcalBarrel_FiberXSpacing"        value="1.34*mm"/>
<constant name="EcalBarrel_FiberZSpacing"        value="1.22*mm"/>
<constant name="EcalBarrel_SpaceBetween"         value="0.1*mm"/>
<constant name="EcalBarrel_FiberChunkLayers_num" value = "12"/>
```

In ecal.xml, endcaps and barrel

```
<documentation level="10">
  ### Ecal configuration
</documentation>
<include ref="ci_ecal.xml"/>
<include ref="hybrid_ecal.xml"/>
<include ref="ecal_barrel_interlayers.xml"/>
```

STEP 2: GENERATE INPUT PARTICLES

Simple Particle Gun

- Use gen_particles.py code and generate hepmc file

(https://eicweb.phy.anl.gov/EIC/benchmarks/reconstruction_benchmarks/-/blob/master/benchmarks/imaging_ecal/scripts/gen_particles.py)

for example,

```
python gen_particles.py embarrel_pion0 –n 1000 --angmin 45 --angmax 135 --pmin 0.1 –pmax 5.0 --particles=“pion0”
```

List of arguments
-n: number of events to generate
--angmin: minimum angle in degree
--angmax: maximum angle in degree
--pmin: minimum momentum in GeV
--pmax: maximum momentum in GeV
--particles: particle pdg code

```
PARTICLES = {
    "pion0": (111, 0.1349766),        # pi0
    "pion+": (211, 0.13957018),       # pi+
    "pion-": (-211, 0.13957018),      # pi-
    "kaon0": (311, 0.497648),         # K0
    "kaon+": (321, 0.493677),         # K+
    "kaon-": (-321, 0.493677),        # K-
    "proton": (2212, 0.938272),       # proton
    "neutron": (2112, 0.939565),      # neutron
    "electron": (11, 0.51099895e-3), # electron
    "positron": (-11, 0.51099895e-3),# positron
    "photon": (22, 0),                # photon
}
```

STEP 2: GENERATE INPUT PARTICLES

Simple Particle Gun

- Use gen_particles.py code and generate hepmc file

  for example, (px, py, pz, e, pdg code, status) info is created

```
E 0 1 3
U GEV MM
P 1 0 11 0.0000000000000000e+00 0.0000000000000000e+00 1.0000000000000000e+01 1.0000000000000000e+01 0.0000000000000000e+00 4
P 2 0 2212 0.0000000000000000e+00 0.0000000000000000e+00 0.0000000000000000e+00 9.3799999999999994e-01 9.3799999999999994e-01 4
V -1 0 [1,2]
P 3 -1 111 -4.0383620994744556e+00 2.1140543164035481e+00 -2.0548493620759158e+00 5.0018215472494623e+00 1.3497700000000531e-01 1
E 0 1 3
U GEV MM
P 1 0 11 0.0000000000000000e+00 0.0000000000000000e+00 1.0000000000000000e+01 1.0000000000000000e+01 0.0000000000000000e+00 4
P 2 0 2212 0.0000000000000000e+00 0.0000000000000000e+00 0.0000000000000000e+00 9.3799999999999994e-01 9.3799999999999994e-01 4
V -1 0 [1,2]
P 3 -1 111 1.5308045578993650e+00 4.1460881342644917e+00 2.3380741195320818e+00 5.0018215472494623e+00 1.3497700000000531e-01 1
```

beam particles info

final state particle info

Above embarrel_pion0.hepmc file will be an input file to Geant4 simulation

STEP 3: RUN GEANT4 SIMULATION

Geant4 simulation

- Get events through the detector geometry
- For example,

```
npsim --runType batch
        -v WARNING
        --part.minimalKineticEnergy “0.5*MeV”
        --numberOfEvents 1000
        --compactFile athena.xml
        --inputFiles embarrel_pion0.hepmc
        --outputFile sim_embarrel_pion0.root
```

--part.minimalKineticEnergy: minimum energy of particles which being saved/tracking into output file

The output will be a root file containing the generated events and calorimeter hits

STEP 4: INFO FROM GEANT4 SIMULATION

Output ROOT file from Geant4 simulation sim_embarrel_pion0.root

- Events – Trees: main data structure

  (https://eic.phy.anl.gov/eicd/)

- mcparticles
  - Geant4Particle class
  - Access thrown particle information such pdgID, vertex, momentum, mass, etc

- EcalBarrelHits
  - CalorimeterHit class
  - Access information such energy deposit, position, etc

STEP 4: INFO FROM GEANT4 SIMULATION

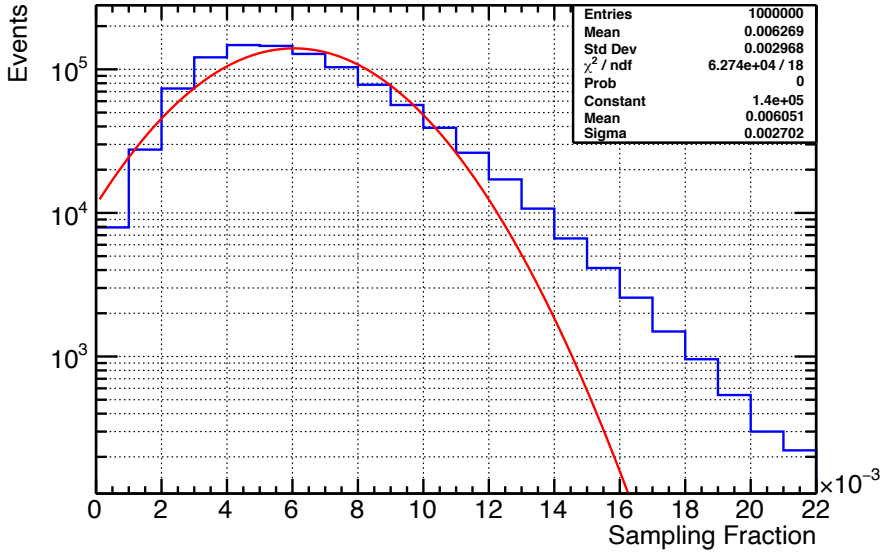
Using output ROOT file from Geant4 simulation

- For example,

```
// Thrown Energy [GeV]
auto Ethr = [](std::vector<dd4pod::Geant4ParticleData> const& input) {
  std::vector<double> result;
  auto p = input[2];
  auto energy = TMath::Sqrt(p.ps.x * p.ps.x + p.ps.y * p.ps.y +
                p.ps.z * p.ps.z + p.mass * p.mass);
  result.push_back(energy);
  return result;
};

// Energy deposition [GeV]
auto Esim = [](const std::vector<dd4pod::CalorimeterHitData>& evt) {
  std::vector<double> result;
  auto total_edep = 0.0;
  for (const auto& i: evt)
    total_edep += i.energyDeposit;
  result.push_back(total_edep);
  return result;
};
```

Imaging Layers

Sampling Fraction = E_{sim}/E_{thr} ~ 0.6%



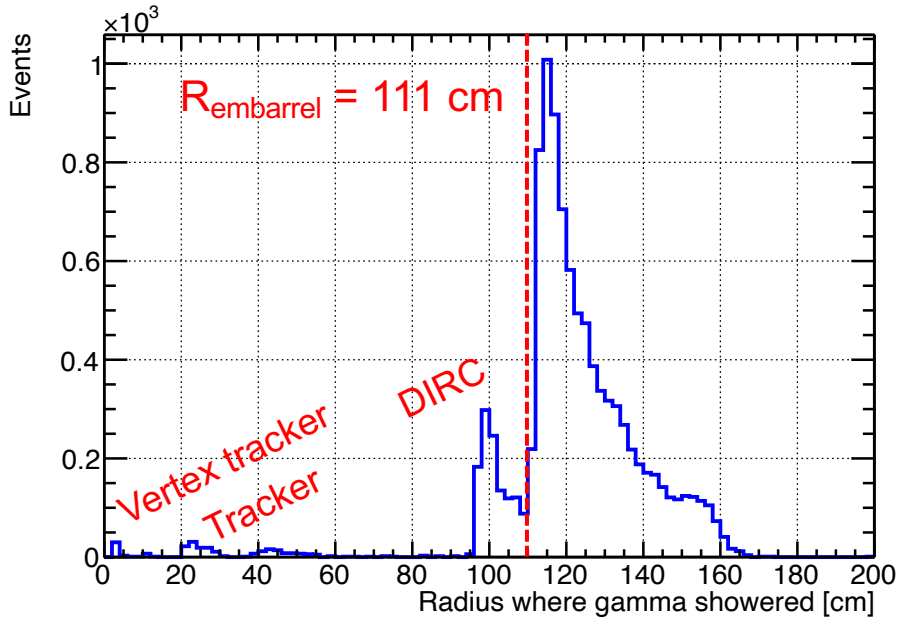
STEP 4: INFO FROM GEANT4 SIMULATION

Using output ROOT file from Geant4 simulation

- For example,

```
// Distance where decaying gammas shower
auto r_end_gamma = [](std::vector<dd4pod::Geant4ParticleData> const& evt) {
  std::vector<double> result;
  auto dist_r_end = 0.0;
  for (const auto& i: evt) {
    if (i.g4Parent == 1 && i.pdgID == 22) {
      dist_r_end = TMath::Sqrt(i.ve.x*i.ve.x + i.ve.y*i.ve.y + i.ve.z*i.ve.z);
      result.push_back(dist_r_end/10.); // unit cm
    }
  }
return result;
};
```

Radius where decaying γs shower



STEP 5: RUN RECONSTRUCTION

Run Juggler

- Juggler is a Gaudi based event processing framework using eicd data model

```
xenv –x Juggler.xenv gaudirun.py options/imaging_topocluster.py
```

- Option file includes a set of algorithms, input data, and output data assembled.
  - Digitization: take the output hits from the geant4 simulation and make them look like real experimental data
  - Reconstruction: take out mock experimental data and reconstruct the events including clustering

```
from Configurables import PodioInput
from Configurables import Jug__Base__InputCopier_dd4pod__Geant4ParticleCollection_dd4pod__Geant4ParticleCollection_ as MCCopier
from Configurables import Jug__Base__InputCopier_dd4pod__CalorimeterHitCollection_dd4pod__CalorimeterHitCollection_ as CalCopier
from Configurables import Jug__Digi__CalorimeterHitDigi as CalorimeterHitDigi
from Configurables import Jug__Reco__ImagingPixelReco as ImagingPixelReco
from Configurables import Jug__Reco__ImagingTopoCluster as ImagingTopoCluster
from Configurables import Jug__Reco__ImagingClusterReco as ImagingClusterReco
```

STEP 5: RUN RECONSTRUCTION

Run Juggler

- Tweak parameters - for example in clustering algorithm, minimum number of hits to form a cluster
- The output will be a root file (rec_embarrel_pion0.root containing the generated events and calorimeter hits from Geant4 simulation and reconstructed events and cluster information
- EcalBarrelImagingClusters
  - Access Cluster info such cluster position, cluster energy, etc

```
imcaldigi = CalorimeterHitDigi("imcal_digi",
        inputHitCollection="EcalBarrelHits",
        outputHitCollection="DigiEcalBarrelImagingHits",
        energyResolutions=[0., 0.02, 0.],
        **daq_setting)
imcalreco = ImagingPixelReco("imcal_reco",
        inputHitCollection=imcaldigi.outputHitCollection,
        outputHitCollection="RecoEcalBarrelImagingHits",
        readoutClass="EcalBarrelHits",
        layerField="layer",
        sectorField="module",
        thresholdFactor=5.0,
        **daq_setting)

imcalcluster = ImagingTopoCluster("imcal_cluster",
        inputHitCollection=imcalreco.outputHitCollection,
        outputProtoClusterCollection="EcalBarrelImagingProtoClusters",
        localDistXY=[2.*mm, 2*mm],
        layerDistEtaPhi=[0.01*rad, 0.01*rad],
        neighbourLayersRange=2,
        sectorDist=30.*mm,
        minClusterNhits=15,
        minClusterEdep=0.5*MeV)

clusterreco = ImagingClusterReco("imcal_clreco",
        inputHitCollection=imcalcluster.inputHitCollection,
        inputProtoClusterCollection=imcalcluster.outputProtoClusterCollection,
        outputLayerCollection="EcalBarrelImagingClustersLayers",
        outputClusterCollection="EcalBarrelImagingClusters",
        outputInfoCollection="EcalBarrelImagingClustersInfo",
        samplingFraction=kwargs['img_sf'])
```

STEP 5: RUN RECONSTRUCTION

Bash script: run_emcal_barrel.sh

```
export CB_EMCAL_COMPACT_PATH=${DETECTOR_PATH}/${JUGGLER_DETECTOR}.xml

if [[ ! -n  "${CB_EMCAL_NUMEV}" ]] ; then
  export CB_EMCAL_NUMEV=1000
fi

if [[ ! -n "${CB_EMCAL_IEV}" ]] ; then
  export CB_EMCAL_IEV="0, 1"
fi

if [[ ! -n  "${CB_EMCAL_ENERGY}" ]] ; then
  export CB_EMCAL_ENERGY=5.0
fi

if [[ ! -n "${CB_EMCAL_SAMP_FRAC}" ]] ; then
  export CB_EMCAL_SAMP_FRAC=0.014
fi

export CB_EMCAL_NAME_TAG="${nametag}"
export CB_EMCAL_GEN_FILE="${CB_EMCAL_NAME_TAG}.hepmc"

export CB_EMCAL_SIM_FILE="sim_${CB_EMCAL_NAME_TAG}.root"
export CB_EMCAL_REC_FILE="rec_${CB_EMCAL_NAME_TAG}.root"
```

Set environment variables

```
echo "CB_EMCAL_NUMEV = ${CB_EMCAL_NUMEV}"
echo "CB_EMCAL_COMPACT_PATH = ${CB_EMCAL_COMPACT_PATH}"

# Generate the input events
python benchmarks/imaging_ecal/scripts/gen_particles.py ${CB_EMCAL_GEN_FILE} -n ${CB_EMCAL_NUMEV}\
    --angmin 60 --angmax 120 --parray ${CB_EMCAL_ENERGY} --particles="${particle}"
if [[ "$?" -ne "0" ]] ; then
  echo "ERROR running script: generating input events"
  exit 1
fi
```

Generate input particles

```
# Run geant4 simulations
npsim --runType batch \
      -v WARNING \
      --part.minimalKineticEnergy "0.5*MeV" \
      --numberOfEvents ${CB_EMCAL_NUMEV} \
      --compactFile ${CB_EMCAL_COMPACT_PATH} \
      --inputFiles ${CB_EMCAL_GEN_FILE} \
      --outputFile ${CB_EMCAL_SIM_FILE}

if [[ "$?" -ne "0" ]] ; then
  echo "ERROR running npdet"
  exit 1
fi
```

Run Geant4 simulation

```
rootls -t "${CB_EMCAL_SIM_FILE}"

# Directory for plots
mkdir -p results

CB_EMCAL_OPTION_DIR=benchmarks/imaging_ecal/options
# Run Juggler
xenv -x ${JUGGLER_INSTALL_PREFIX}/Juggler.xenv \
    gaudirun.py ${CB_EMCAL_OPTION_DIR}/hybrid_cluster.py
# gaudirun.py ${CB_EMCAL_OPTION_DIR}/imaging_topocluster.py
if [[ "$?" -ne "0" ]] ; then
  echo "ERROR running juggler"
  exit 1
fi
```

Run Juggler (Reconstruction)

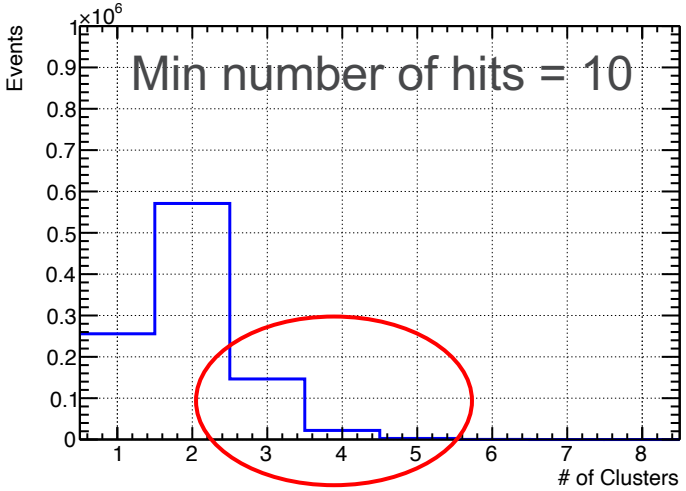
STEP 6: ANALYZE OUTPUT

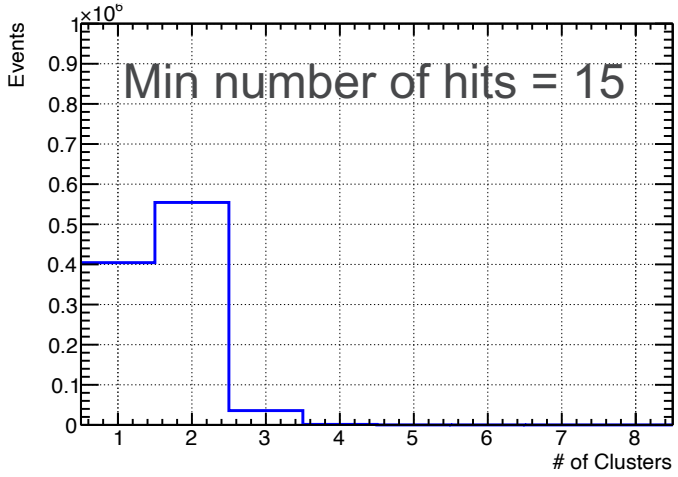
Number of Clusters

- Tweak clustering parameters to find optimal value for algorithm

```
// Number of Clusters
auto ncluster = [](const std::vector<eic::ClusterData>& evt) {
  return (int)evt.size();
};
```

```
imcalcluster = ImagingTopoCluster("imcal_cluster",
        inputHitCollection=imcalreco.outputHitCollection,
        outputProtoClusterCollection="EcalBarrelImagingProtoClusters",
        localDistXY=[2.*mm, 2*mm],
        layerDistEtaPhi=[0.01*rad, 0.01*rad],
        neighbourLayersRange=2,
        sectorDist=30.*mm,
        minClusterNhits=15,
        minClusterEdep=0.5*MeV)
```





STEP 6: ANALYZE OUTPUT

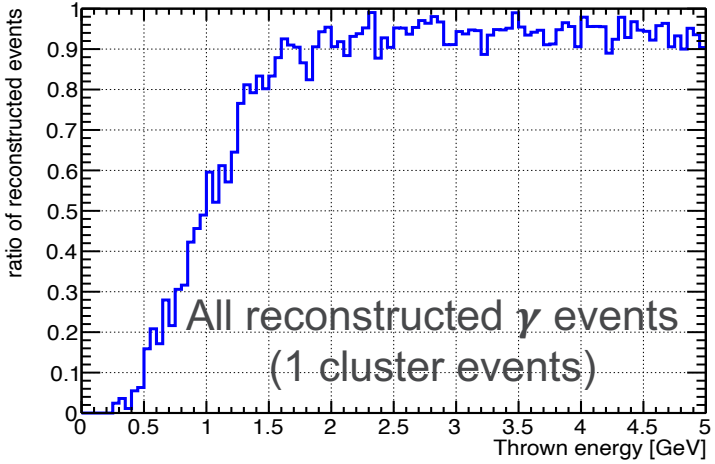
Threshold Energy

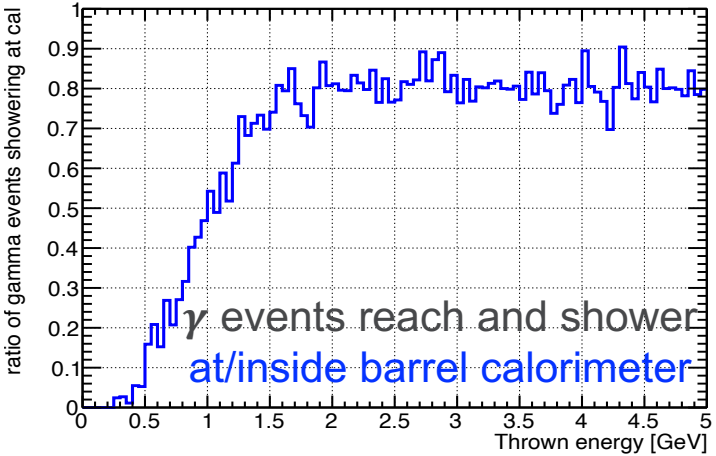
```
// Define variables
auto d1 = d0.Define("Ethr",          Ethr,         {"mcparticles2"})
            .Define("ncluster",      ncluster,     {"EcalBarrelClusters3D"})
            .Define("r_end_gamma",   r_end_gamma,  {"mcparticles2"})
            ;

// Define Histograms
auto hEthr = d1.Histo1D({"hEthr", "Thrown Energy; Thrown Energy [GeV]; Events", 100, 0.0, 5.0}, "Ethr");

auto d2 = d1.Filter("ncluster==1");
auto hEthrNC1 = d2.Histo1D({"hEthrNC1","Thrown Energy; Thrown Energy [GeV]; Events", 100, 0.0, 5.0}, "Ethr");

auto d3 = d2.Filter("r_end_gamma > 111.0");
auto hEthrNC1DistCUT = d3.Histo1D({"hEthrNC1DistCUT", "Thrown Energy; Thrown Energy [GeV]; Events", 100, 0.0, 5.0}, "Ethr");
```

Filter function – apply cuts on events

```
TCanvas* c1 = new TCanvas("c1", "c1");
TH1D *h = (TH1D*)hEthrNC1->Clone("h");
h->SetLineColor(kBlue);
h->SetLineWidth(3);
h->Divide(hEthr.GetPtr());
h->GetYaxis()->SetRangeUser(0.0,1.0);
h->GetYaxis()->SetTitle("ratio of reconstructed events");
h->GetXaxis()->SetTitle("Thrown energy [GeV]");
h->Draw();
c1->SaveAs("results/emcal_barrel_photon_evt_cl_cut.png");
c1->SaveAs("results/emcal_barrel_photon_evt_cl_cut.pdf");
```

```
TCanvas* c2 = new TCanvas("c2", "c2");
TH1D *h = (TH1D*)hEthrNC1Dist->Clone("h");
h->SetLineColor(kBlue);
h->SetLineWidth(3);
h->Divide(hEthr.GetPtr());
h->GetYaxis()->SetRangeUser(0.0,1.0);
h->GetYaxis()->SetTitle("ratio of gamma events showering at cal");
h->GetXaxis()->SetTitle("Thrown energy [GeV]");
h->Draw();
c2->SaveAs("results/emcal_barrel_photon_evt_cl_cut_dist_cut.png");
c2->SaveAs("results/emcal_barrel_photon_evt_cl_cut_dist_cut.pdf");
```





STEP 6: ANALYZE OUTPUT

Using output ROOT file from reconstruction

- Event Display with 5 GeV $\pi^0 \rightarrow \gamma\gamma$ events
- Access mcparticles and RecoEcalBarrelImagingHits/Clusters information (draw_cluster.py)

(https://eicweb.phy.anl.gov/EIC/benchmarks/reconstruction_benchmarks/-/blob/master/benchmarks/imaging_ecal/scripts/draw_cluster.py)

  - Compare cluster and true gamma positions

Cuts on clustering algorithm
- Min nhits = 15
- Min cluster edep = 0.5 MeV

$E_{cluster\ \#1}$ = 3.23 GeV
$E_{cluster\ \#2}$ = 1.94 GeV
Total E_{rec} = 5.17 GeV

Cluster hits and true γ positions