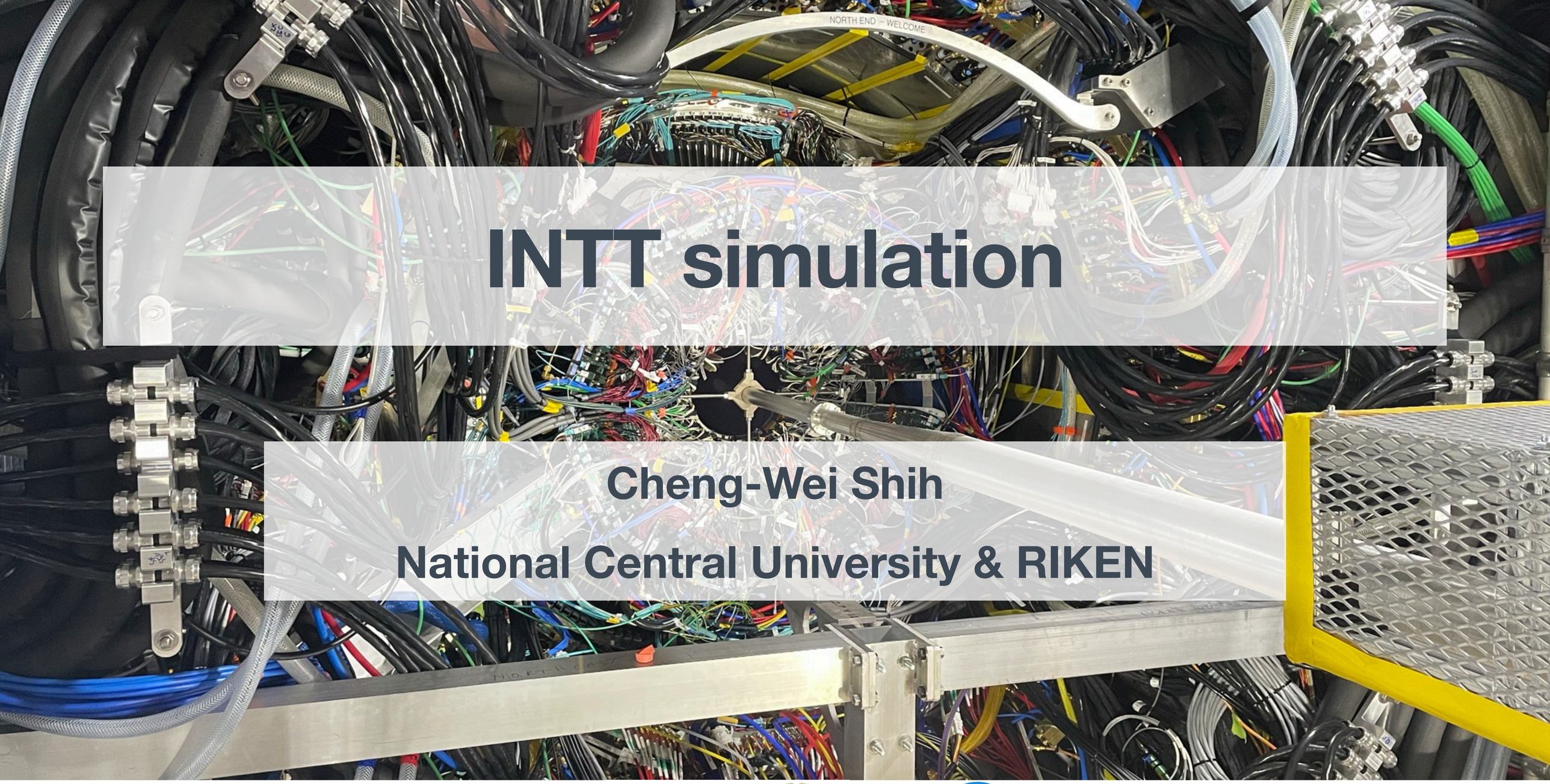


INTT simulation

Cheng-Wei Shih

National Central University & RIKEN

Nov 7th, 2024
INTT meeting

INTT Geant4 issue spot - 1

Spotted by Cameron

```
242         // Si-sensor inactive area
243         G4VSolid *siinactive_box = new G4SubtractionSolid((boost::format("siinactive_box_%d_%d") % inttlayer % itype).str(), sifull_box, siactive_
244         G4LogicalVolume *siinactive_volume = new G4LogicalVolume(siinactive_box, GetDetectorMaterial("G4_Si"), (boost::format("siinactive_volume_
245
246         if ((m_IsAbsorberActiveMap.find(inttlayer))->second > 0)
247         {
248           m_PassiveVolumeTuple.insert(std::make_pair(siinactive_volume, std::make_tuple(inttlayer, PHG4InttDefs::SI_INACTIVE)));
249         }
250         m_DisplayAction->AddVolume(siinactive_volume, "SiInActive");
251
252         // Glue for Si-sensor full area
253         G4VSolid *si_glue_box = new G4Box((boost::format("si_glue_box_%d_%d") % inttlayer % itype).str(), si_glue_x / 2., sifull_y / 2.0, sifull_
254
255         G4LogicalVolume *si_glue_volume = new G4LogicalVolume(si_glue_box, GetDetectorMaterial("SilverEpoxyGlue_INTT"), (boost::format("si_glue_v
256
257         if ((m_IsAbsorberActiveMap.find(inttlayer))->second > 0)
258         {
259           m_PassiveVolumeTuple.insert(std::make_pair(siinactive_volume, std::make_tuple(inttlayer, PHG4InttDefs::SI_GLUE)));
260         }
261         m_DisplayAction->AddVolume(si_glue_volume, "SiGlue");
```

Incorrect assignment of key of the “passive volume map”
Not yet fixed, seems not to be a urgent problem

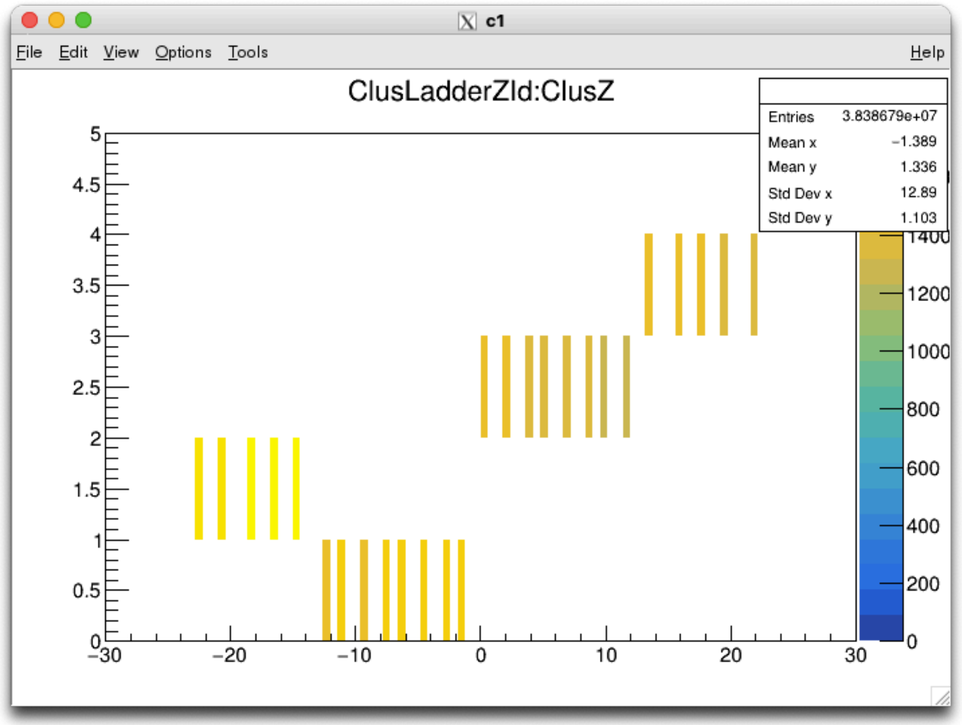
INTT Geant4 issue spot - 2

The cluster Z index

zID 1 (B)	zID 0 (A)	zID 2 (A)	zID 3 (B)

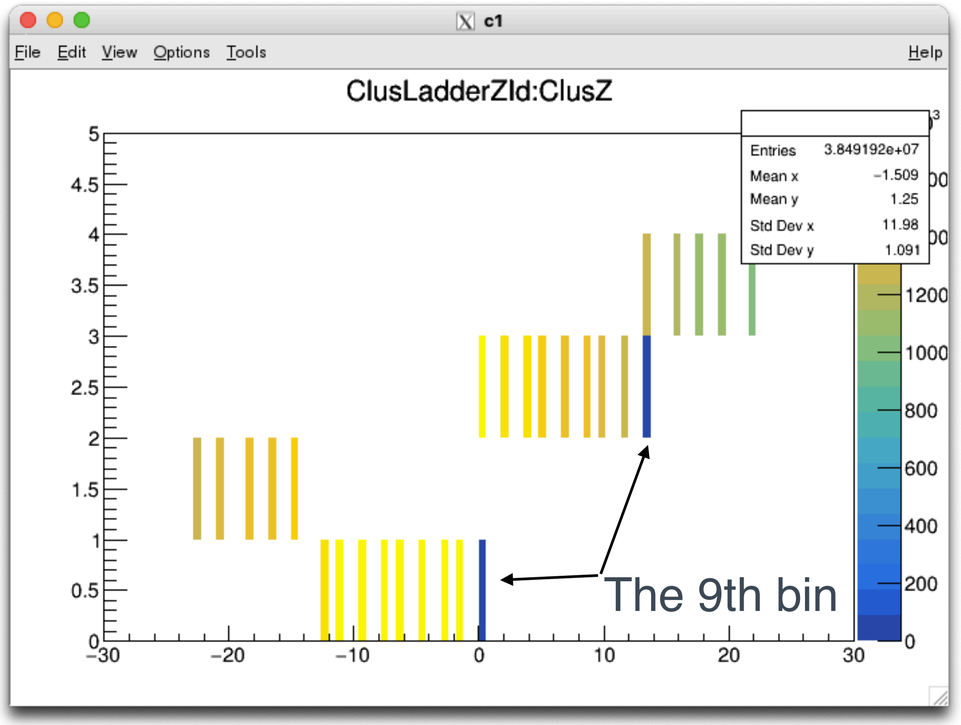
data

Run 54280 (AuAu run in Zero field)



Simulation

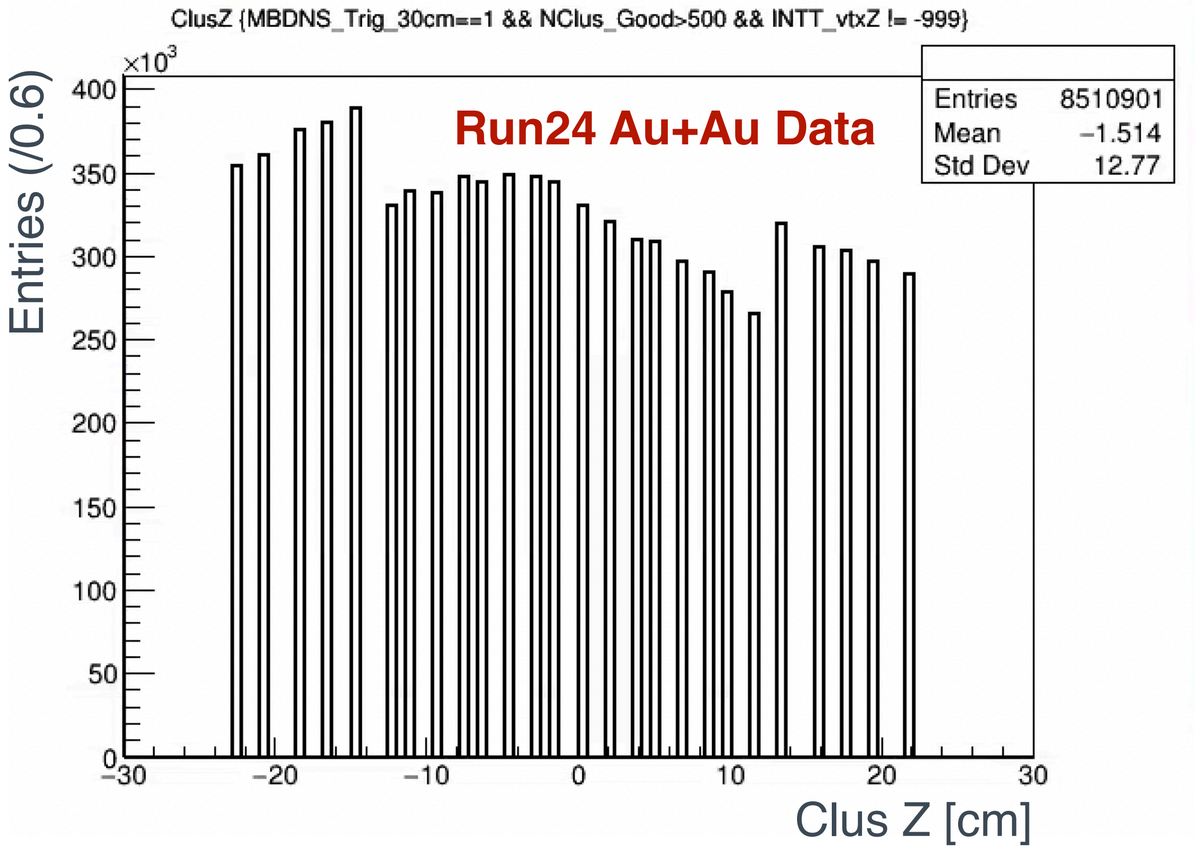
Sim_Ntuple_HIJING_ana443_20241030



It’s a problem, but seems to be minor, at least at this moment

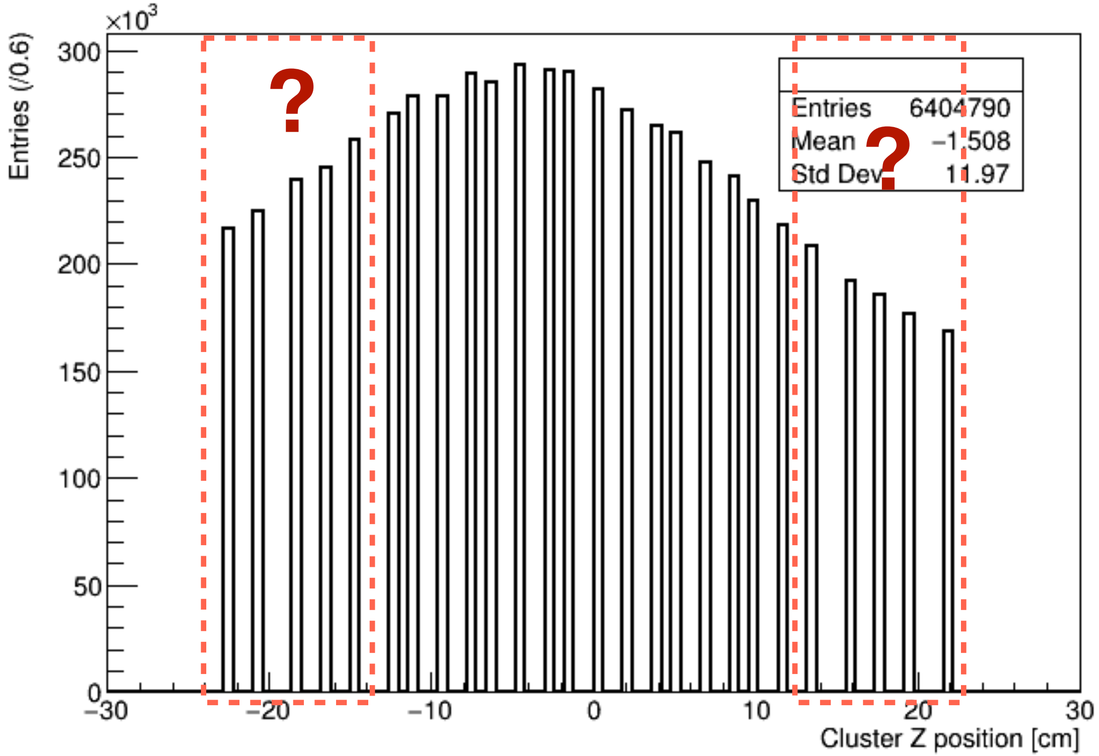
INTT Geant4 issue spot - 3

First 10k events



Simulation

Sim_Ntuple_HIJING_ana443_20241030



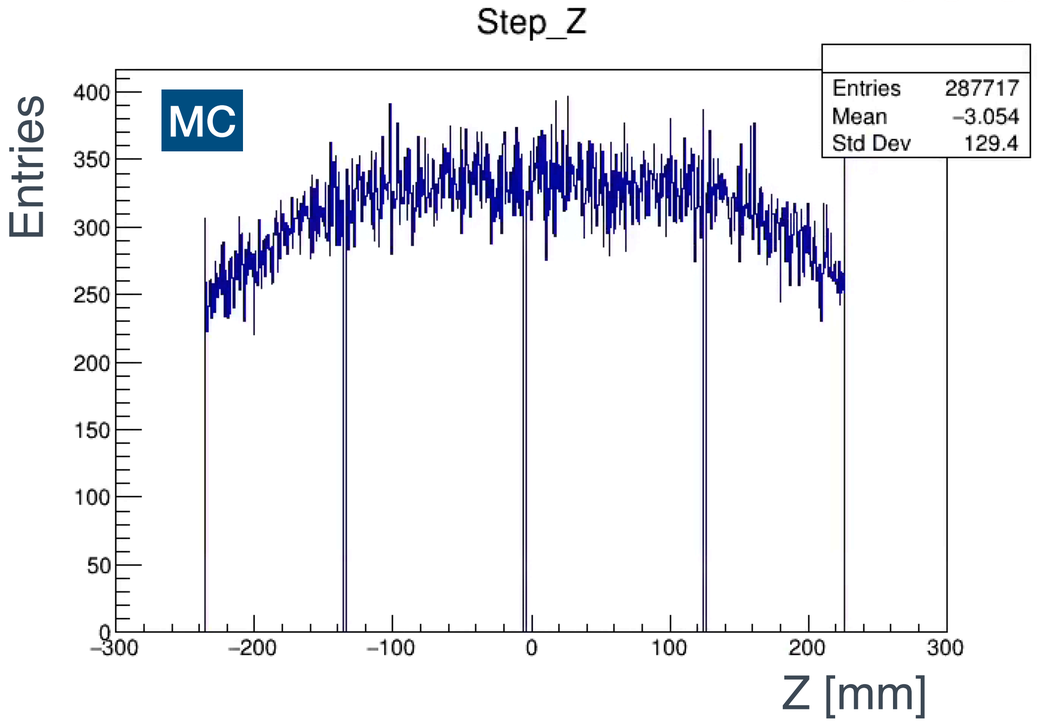
Some INTT hits in the type-B sensors are randomly omitted

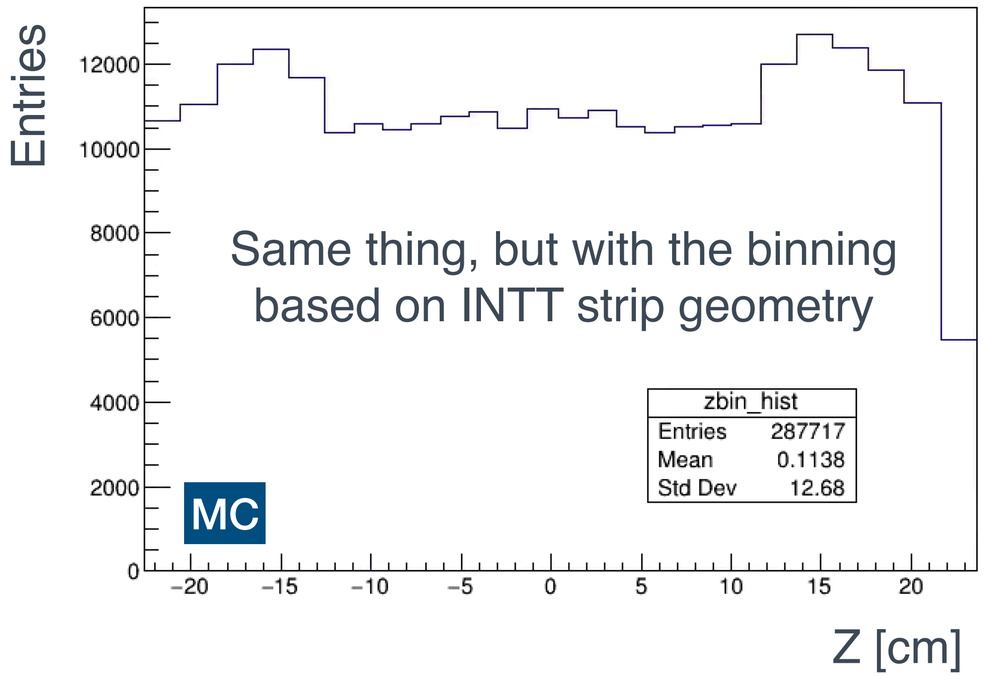
A big problem in our INTT simulation!! → It affects the cluster η distribution!

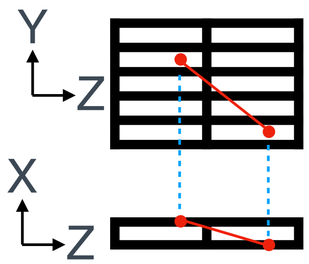
INTT Geant4 issue spot - 3

In PHG4InttSteppingAction.cc/.h

The G4step Post_Z position







G4step: the distance of the particle interacts with the active volume (the lowest level info. you can get from G4)

The distribution of G4step Z position seems to be reasonable → The INTT geometry in the simulation is correct

The next object post G4step → G4Hit

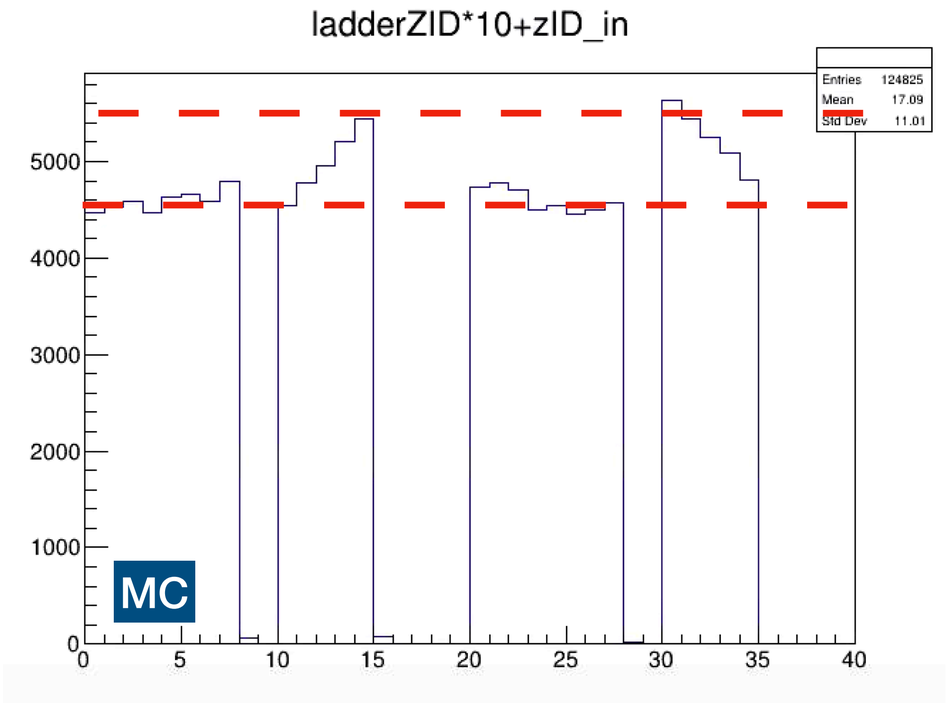
INTT Geant4 issue spot - 3

In PHG4InttHitReco.cc/.h

The G4hit Z index

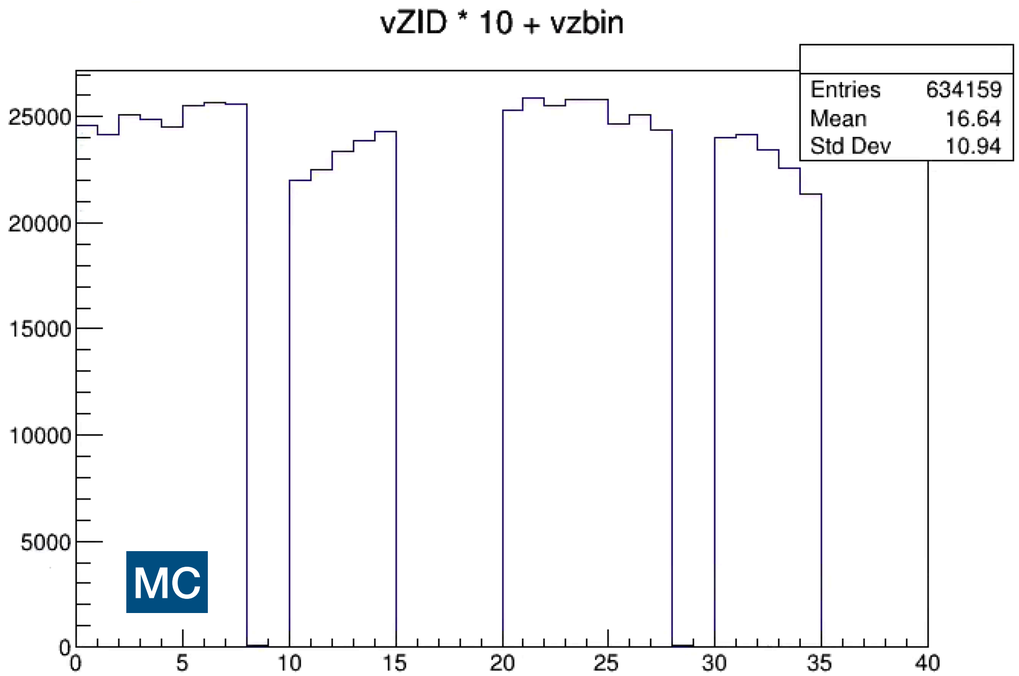
zID 1 (B)	zID 0 (A)	zID 2 (A)	zID 3 (B)

Before charge diffusion



Ladder Z ID x 10 + strip_ZID

After charge diffusion



Ladder Z ID x 10 + strip_ZID

Apparently that the bug is in the step of charge diffusion

INTT Geant4 issue spot - 3

In PHG4InttHitReco.cc/.h

```
457         // Now find the area of overlap of the diffusion circle with each pixel and apportion the energy
458         for (int iz = minstrip_z; iz <= maxstrip_z; iz++)
459         {
460           for (int iy = minstrip_y; iy <= maxstrip_y; iy++)
461           {
462             // Find the pixel corners for this pixel number
463             double location[3] = {-1, -1, -1};
464             layergeom->find_strip_center_localcoords(ladder_z_index, iy, iz, location);
465             // note that (y1,z1) is the top left corner, (y2,z2) is the bottom right corner of the pixel - circle_rectangle_intersectio
466             double y1 = location[1] - layergeom->get_strip_y_spacing() / 2.0;
467             double y2 = location[1] + layergeom->get_strip_y_spacing() / 2.0;
468             double z1 = location[2] + layergeom->get_strip_z_spacing() / 2.0;
469             double z2 = location[2] - layergeom->get_strip_z_spacing() / 2.0;
470
471             // here m_SegmentVec.1 (Y) and m_SegmentVec.2 (Z) are the center of the circle, and diffusion_radius is the circle radius
472             // circle_rectangle_intersection returns the overlap area of the circle and the pixel. It is very fast if there is no overl
473             double striparea_frac = PHG4Utils::circle_rectangle_intersection(y1, z1, y2, z2, gsl_vector_get(m_SegmentVec, 1), gsl_vecto
474             // assume that the energy is deposited uniformly along the tracklet length, so that this segment gets the fraction 1/nsegme
475             stripenergy[iy - minstrip_y][iz - minstrip_z] += striparea_frac * hiter->second->get_edep() / (float) nsegments;
476             if (hiter->second->has_property(PHG4Hit::prop_eion))
477             {
478               stripeion[iy - minstrip_y][iz - minstrip_z] += striparea_frac * hiter->second->get_eion() / (float) nsegments;
479             }
480             if (Verbosity() > 5)
481             {
482               std::cout << "    strip y index " << iy << " strip z index  " << iz
483                         << " strip area fraction of circle " << striparea_frac << " accumulated pixel energy " << stripenergy[iy - mins
484                         << std::endl;
485             }
486           }
487         }
488       }  // end loop over segments
```

In CylinderGeomIntt.h

```
double get_strip_z_spacing() const override
{
  return m_StripZ[0];
}
```

!

This function is inherited from PHG4CylinderGeom.h, no passed argument available

Function only returns the strip length of type A sensor

FYI, the charge diffusion part seems to be copied from the MVTX entirely. And MVTX has only one cell type

INTT Geant4 issue spot - 3

In PHG4InttHitReco.cc/.h

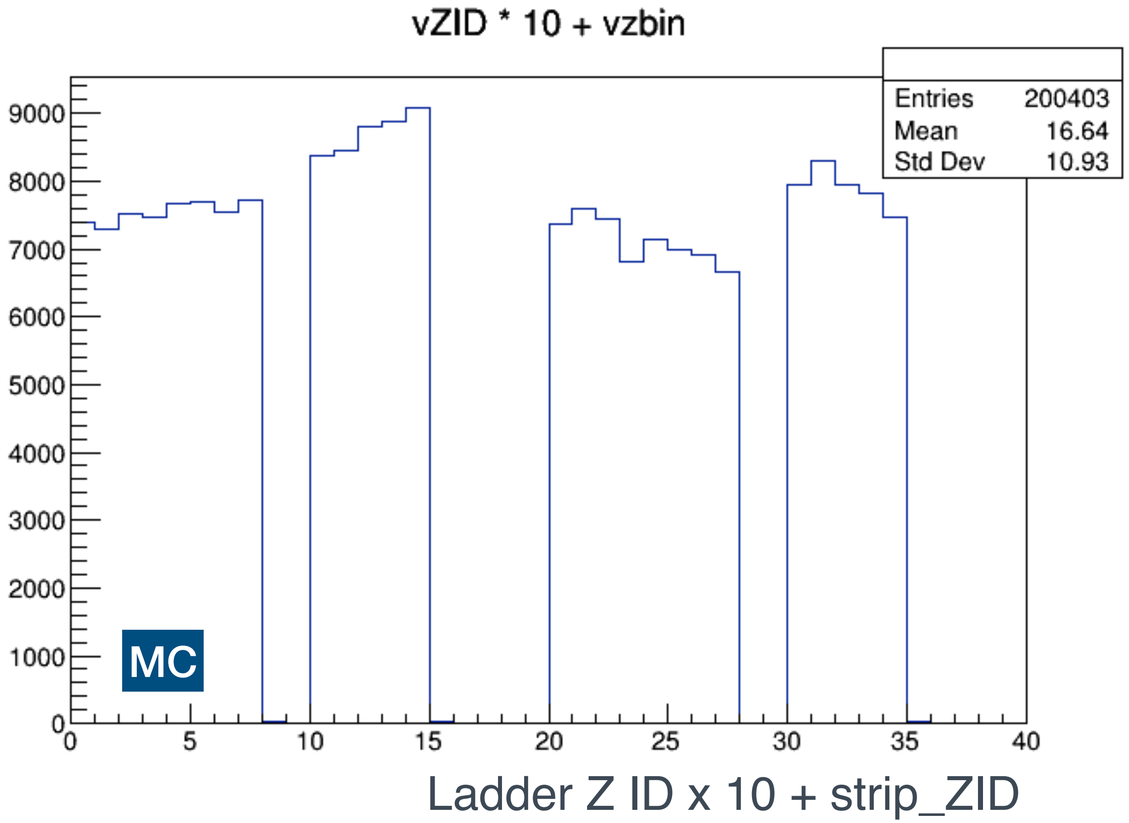
A quick fix in the offline, as a trial

```
// Now find the area of overlap of the diffusion circle with each pixel and apportion the
for (int iz = minstrip_z; iz <= maxstrip_z; iz++)
{
  for (int iy = minstrip_y; iy <= maxstrip_y; iy++)
  {
    // Find the pixel corners for this pixel number
    double location[3] = {-1, -1, -1};
    layergeom->find_strip_center_localcoords(ladder_z_index, iy, iz, location);
    // note that (y1,z1) is the top left corner, (y2,z2) is the bottom right corner of the

    // if (iy == minstrip_y && iz == minstrip_z) {std::cout<<"test test, "<<layergeom->get_

    double strip_z_length = (ladder_z_index == 1 || ladder_z_index == 3) ? 2.0 : 1.6;

    double y1 = location[1] - layergeom->get_strip_y_spacing() / 2.0;
    double y2 = location[1] + layergeom->get_strip_y_spacing() / 2.0;
    double z1 = location[2] + strip_z_length / 2.0;
    double z2 = location[2] - strip_z_length / 2.0;
```



INTT Geant4 issue spot - 3 (the fix)

In CylinderGeomIntt.h

```
double get_strip_z_spacing() const override
{
  return m_StripZ[0];
}
```

```
double get_strip_z_spacing(const int itype = 0) const override
{
  return (itype == 0 || itype == 1) ? m_StripZ[itype] : m_StripZ[0];
}
```

In PHG4InttHitReco.cc

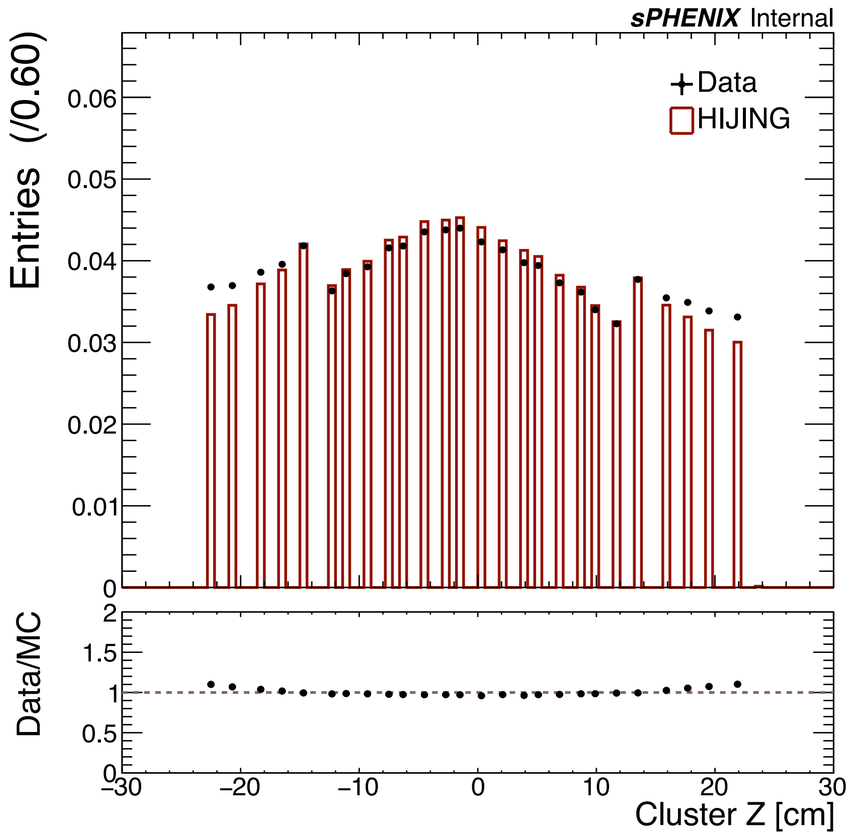
```
// note that (y1,z1) is the top left corner, (y2,z2) is the bottom
double y1 = location[1] - layergeom->get_strip_y_spacing() / 2.0;
double y2 = location[1] + layergeom->get_strip_y_spacing() / 2.0;
double z1 = location[2] + layergeom->get_strip_z_spacing() / 2.0;
double z2 = location[2] - layergeom->get_strip_z_spacing() / 2.0;
```

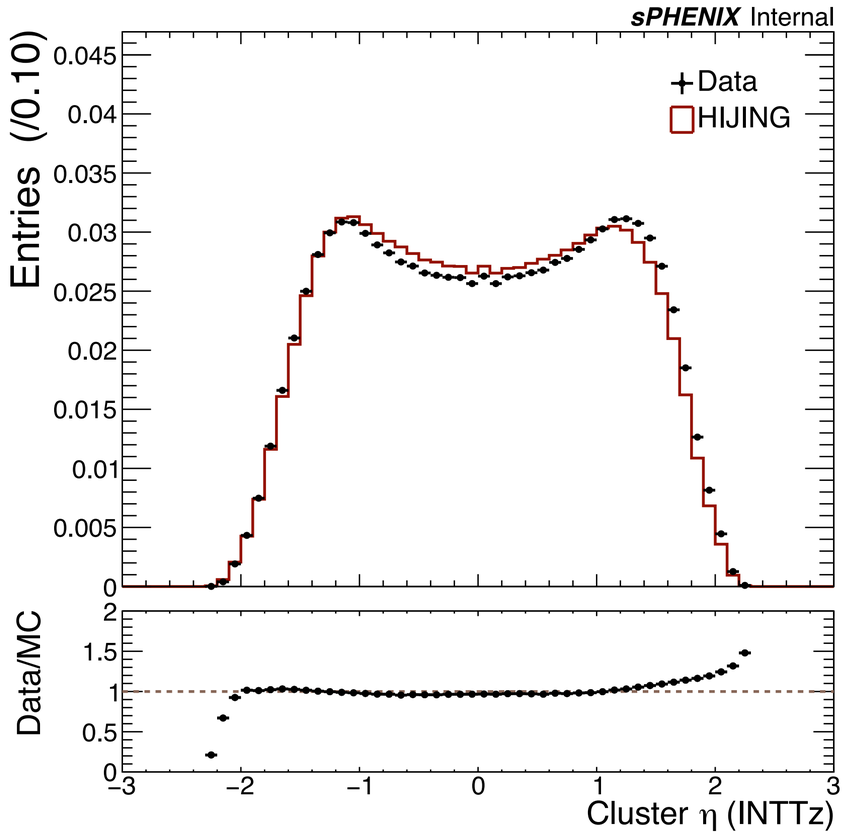
```
double y1 = location[1] - layergeom->get_strip_y_spacing() / 2.0;
double y2 = location[1] + layergeom->get_strip_y_spacing() / 2.0;
double z1 = location[2] + layergeom->get_strip_z_spacing(type) / 2.0;
double z2 = location[2] - layergeom->get_strip_z_spacing(type) / 2.0;
```

- The effect:
  - In simulation, **hit-drop issue** in the charge diffusion step
  - In simulation/data, the **cluster Z error** in the clustering step

```
float length = geom->get_strip_z_spacing(type);
// z error.
const float zerror = zbins.size() * length * invsqrt12;
```

INTT Geant4 issue spot - 3



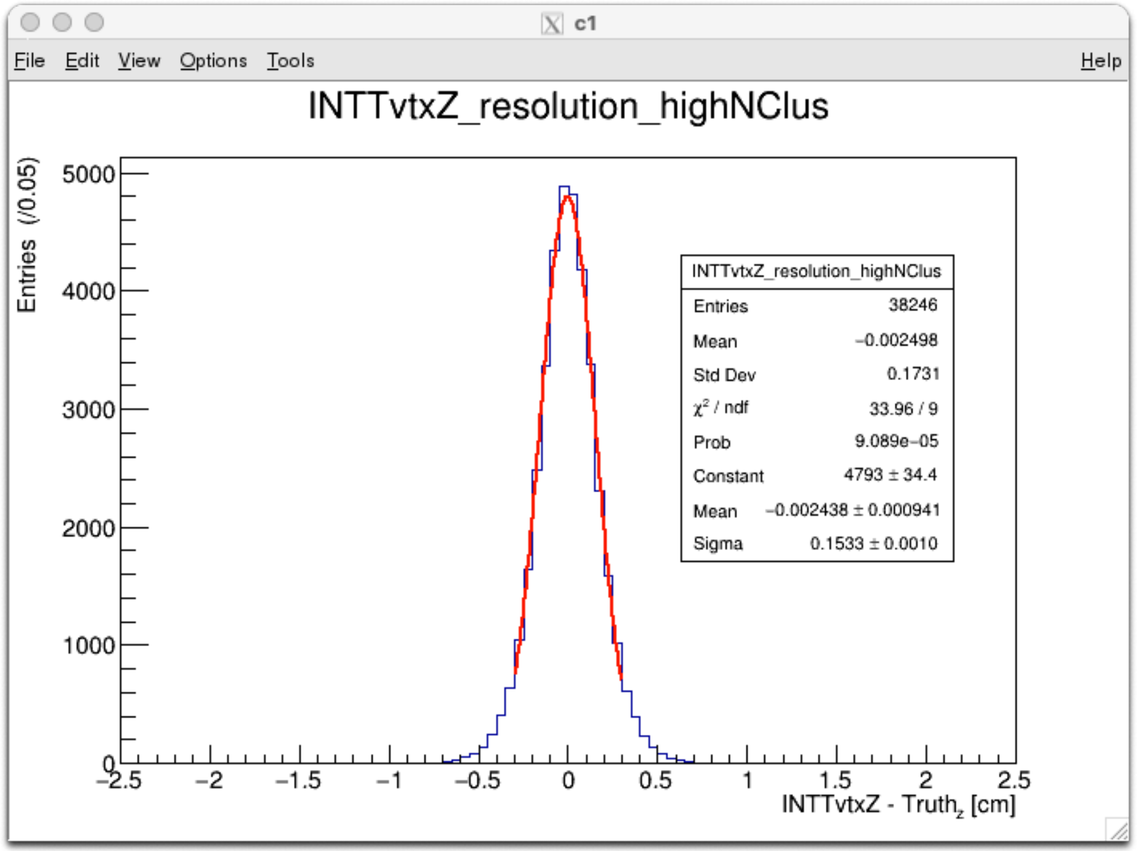


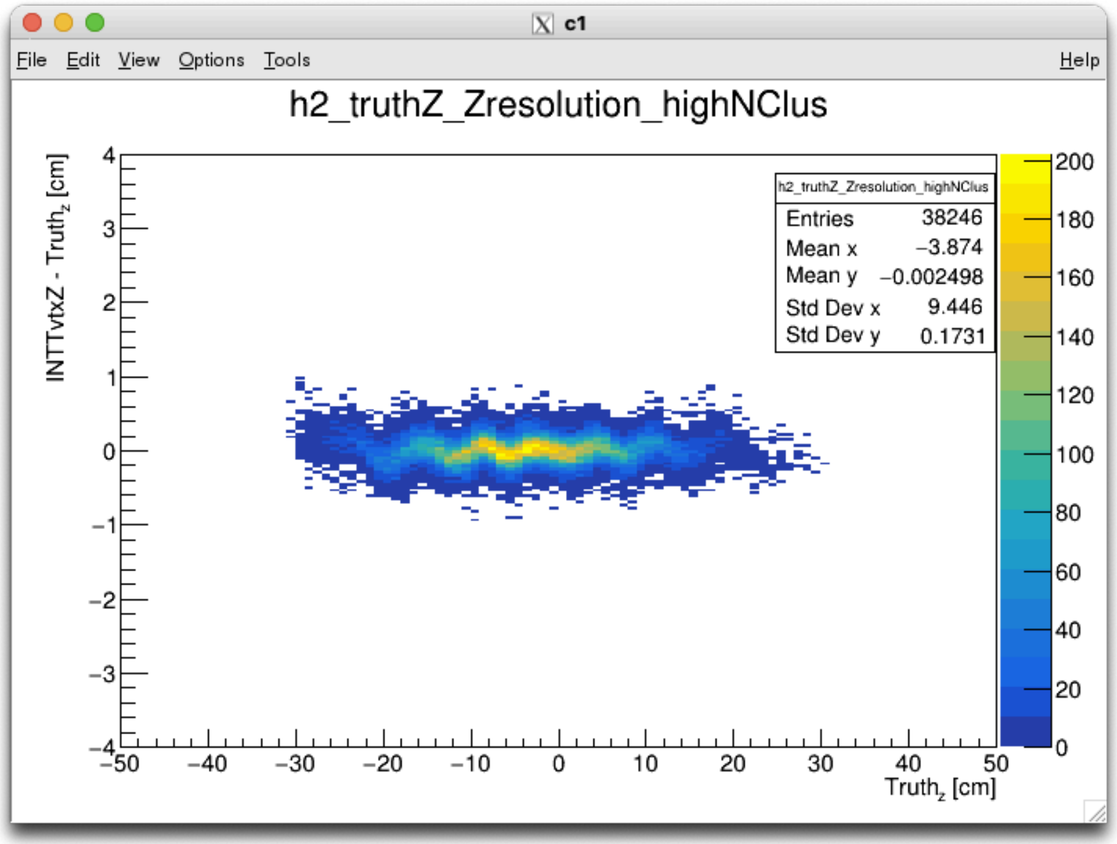
Bug seems to be fixed

INTT vtxZ resolution

Simulation, HIJING, peaked at -4 cm

High multiplicity region selected (to only focus on the best case)



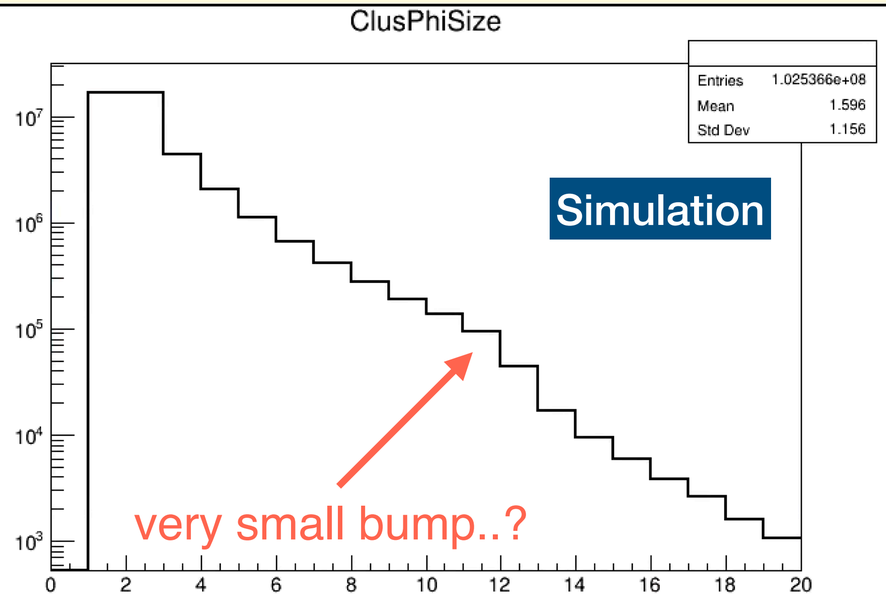


The resolution can be up to **[1.5 mm]**, which is quite nice!
Still see the wiggling structure after the bug fix

INTT Geant4 issue spot - 4

- In PHG4InttHitReco.h/.cc,
  - Only do the charge diffusion if the number of strips involved of this G4step is smaller than 13

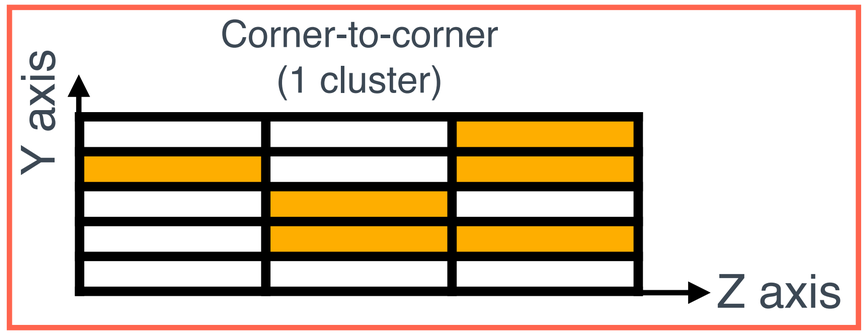
```
405         // skip this hit if it involves an unreasonable  number of pixels
406         // this skips it if either the xbin or ybin range traversed is greater than 8
407         if (maxstrip_y - minstrip_y > 12 || maxstrip_z - minstrip_z > 12)
408         {
409           continue;
410         }
```



The plan of clustering in Z axis?

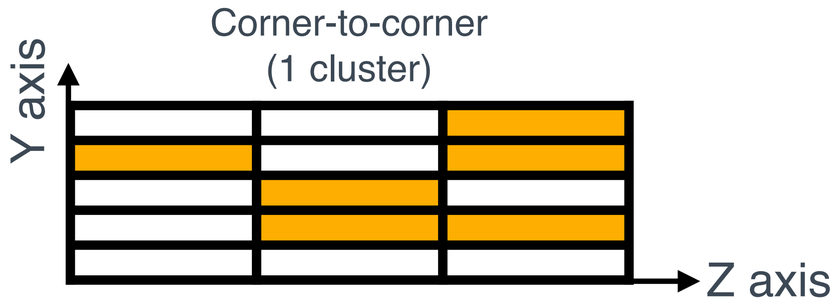
In InttClusterizer.cc

```
54  bool InttClusterizer::ladder_are_adjacent(const std::pair<TrkrDefs::hitkey, TrkrHit*>& lhs, const std::pair<TrkrDefs::hitkey, TrkrHit*>& rhs, const :
55  {
56    if (get_z_clustering(layer))
57    {
58      if (fabs(InttDefs::getCol(lhs.first) - InttDefs::getCol(rhs.first)) <= 1)
59      {
60        if (fabs(InttDefs::getRow(lhs.first) - InttDefs::getRow(rhs.first)) <= 1)
61        {
62          return true;
63        }
64      }
65    }
66    else if (fabs(InttDefs::getCol(lhs.first) - InttDefs::getCol(rhs.first)) == 0)
67    {
68      if (fabs(InttDefs::getRow(lhs.first) - InttDefs::getRow(rhs.first)) <= 1)
69      {
70        return true;
71      }
72    }
73
74    return false;
75  }
```



- The current requirement of the INTT hit clustering in Z axis is corner-by-corner, which might fit to the case of MVTX but not for the INTT for sure
- The INTT clustering in Z axis is disable in default. But shall we fix it?

The plan of clustering in Z axis?



The local quick fix, as a trial

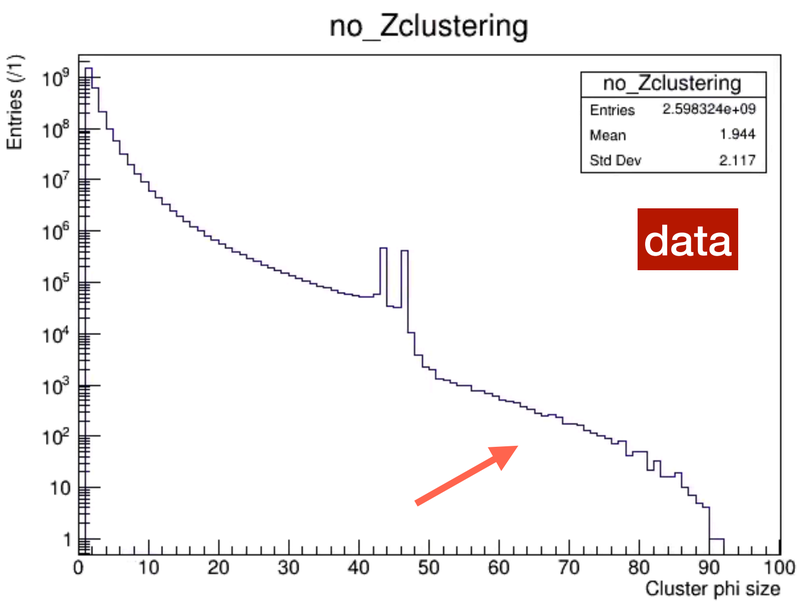
```
if (get_z_clustering(layer))
{
  if ( fabs(InttDefs::getCol(lhs.first) - InttDefs::getCol(rhs.first)) + fabs(InttDefs::getRow(lhs.first) - InttDefs::getRow(rhs.first)) <= 1 )
  {
      return true;
  }
  else
  {
      return false;
  }
}
```

If the distance b/w the two strips are less or equal to 1, they are adjacent

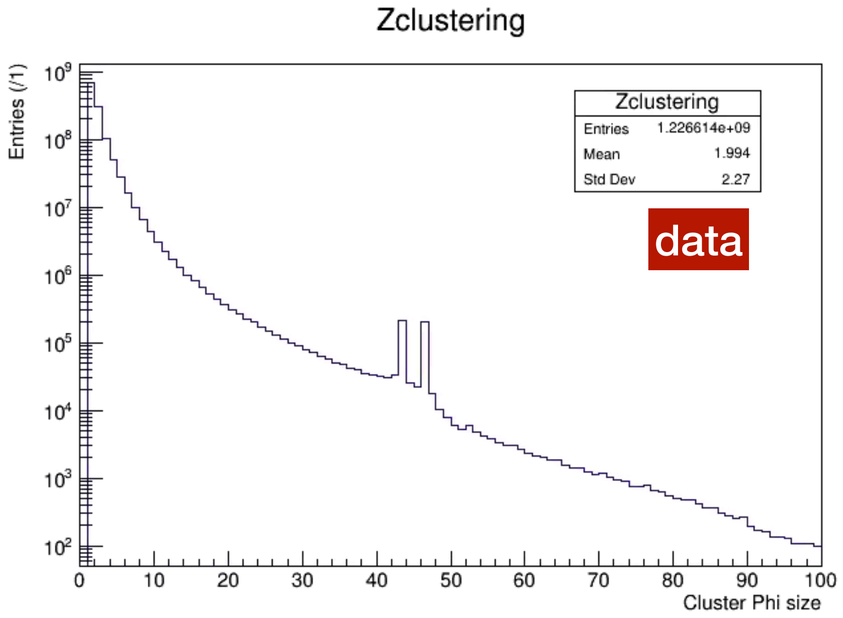
Cluster phi size distributions

Run24 Au+Au Data, run 54280

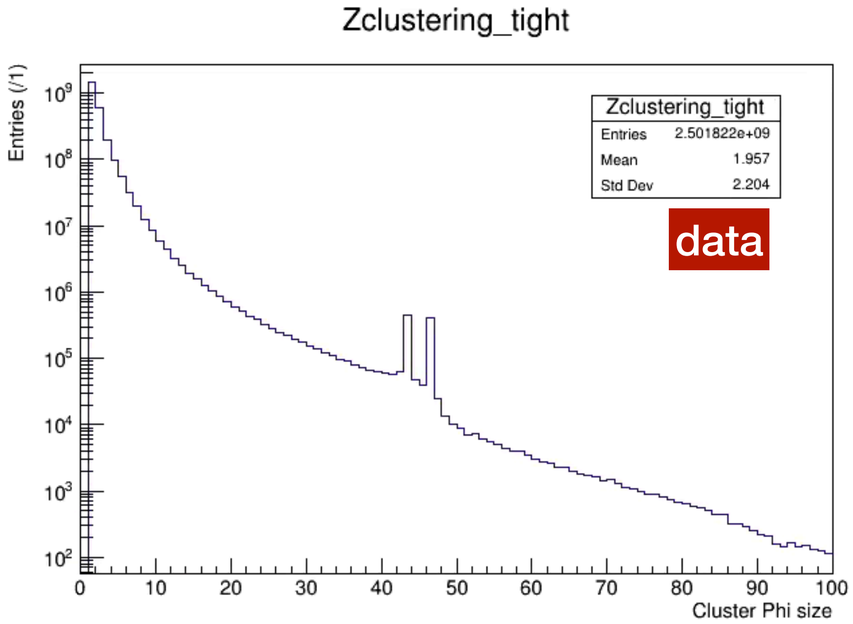
No clustering in Z



Clustering in Z
(corner-to-corner)



Clustering in Z
(edge-to-edge)



The whole phi distribution seems to be more smooth if the clustering in Z axis is enabled

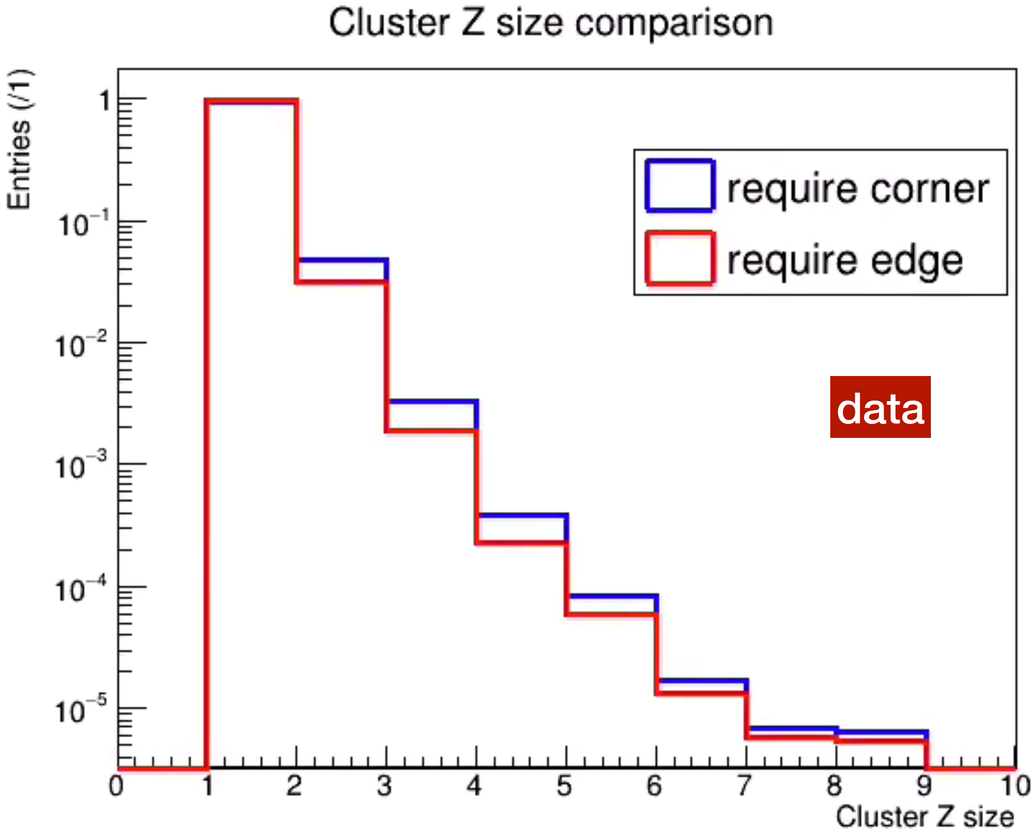
Ntuples available:

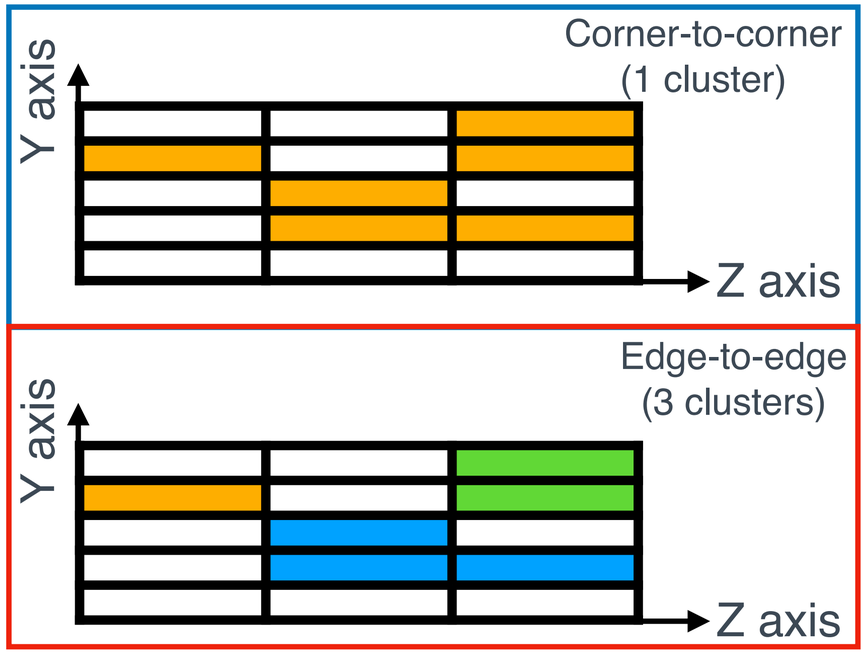
(CtC) /sphenix/tg/tg01/commissioning/INTT/work/cwshih/seflgendata/run_54280/From_official_INTTRAWHIT_DST_clusterZ/completed/*.root

(EtE) /sphenix/tg/tg01/commissioning/INTT/work/cwshih/seflgendata/run_54280/From_official_INTTRAWHIT_DST_clusterZTight/completed/*.root

Cluster Z size distribution

Run24 Au+Au Data, run 54280

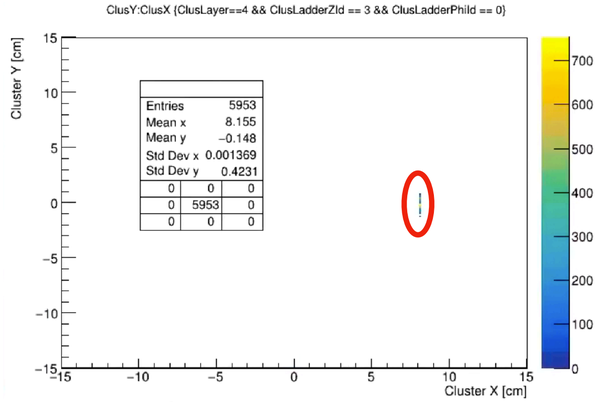




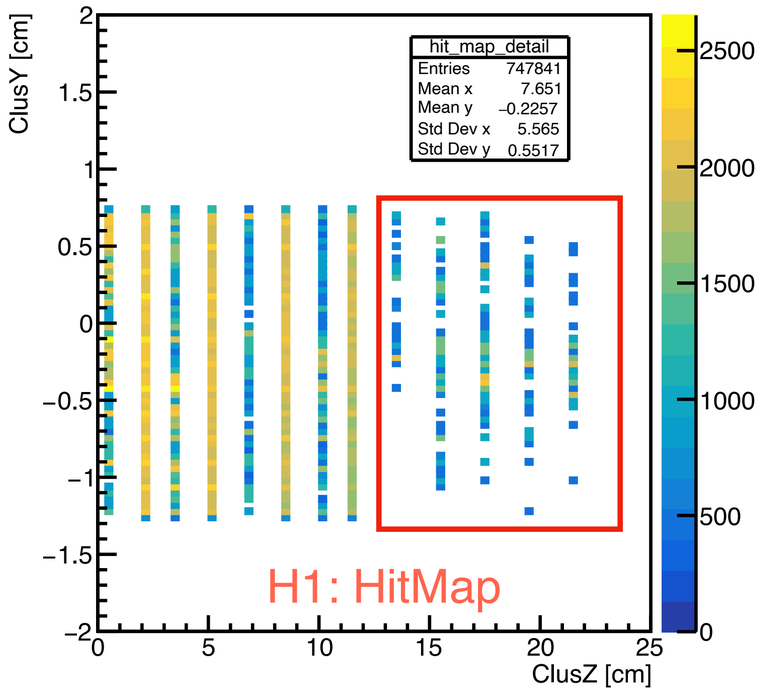
Back up

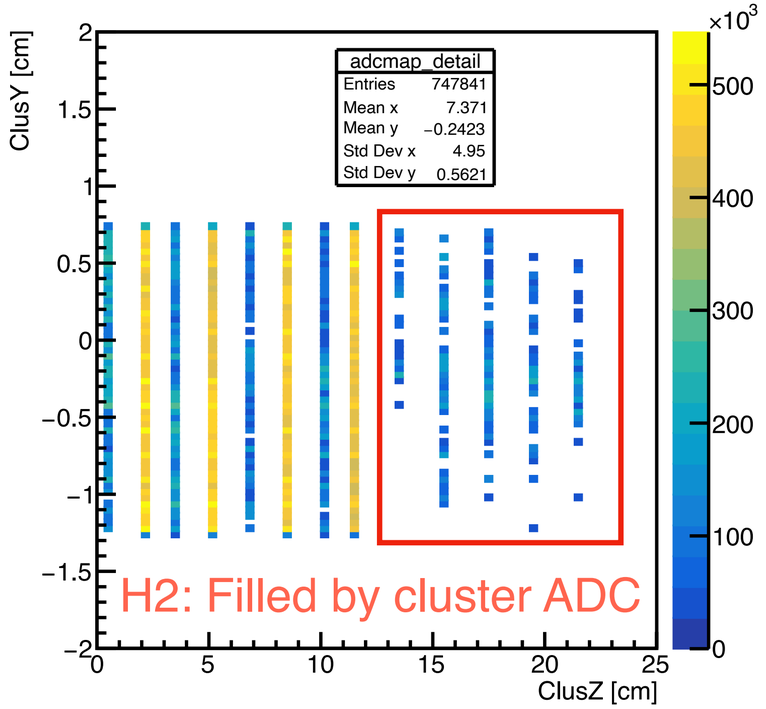
Average cluster ADC

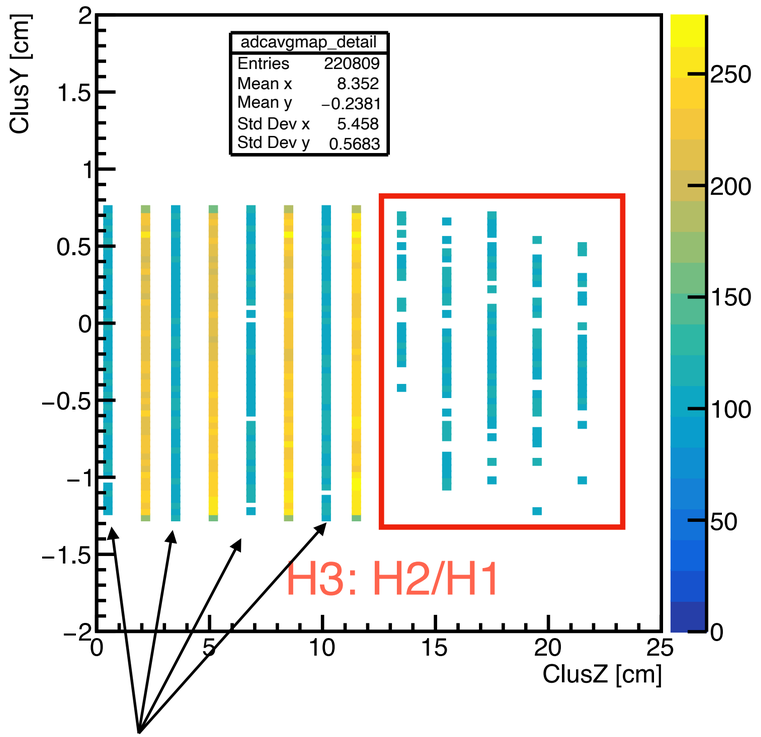
Run 54280 (AuAu run in Zero field)



LayerID 4, **LadderZID (2 II 3)** and **LadderPhiID 0**







Not sure what happened….

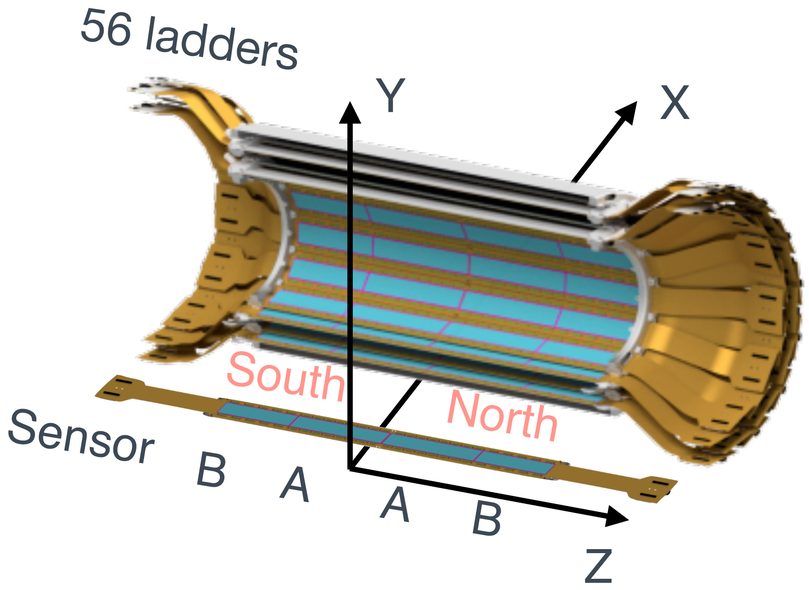
Run description - 54280

- Au+Au collisions at $\sqrt{s} = 200\ GeV$ in zero field
- Data taking time: 1 hour (2024-10-10 05:43:52 → 2024-10-10 06:44:03)
- Number of events: 10,610,255
- Official DST production was not available* → Private production with F4A
  - .evt files → INTTRawHit DST → TrkrHitSet → TrkrCluster
  - Analysis build: ana.439
- 1M production is still ongoing, the first 10k events are analyzed

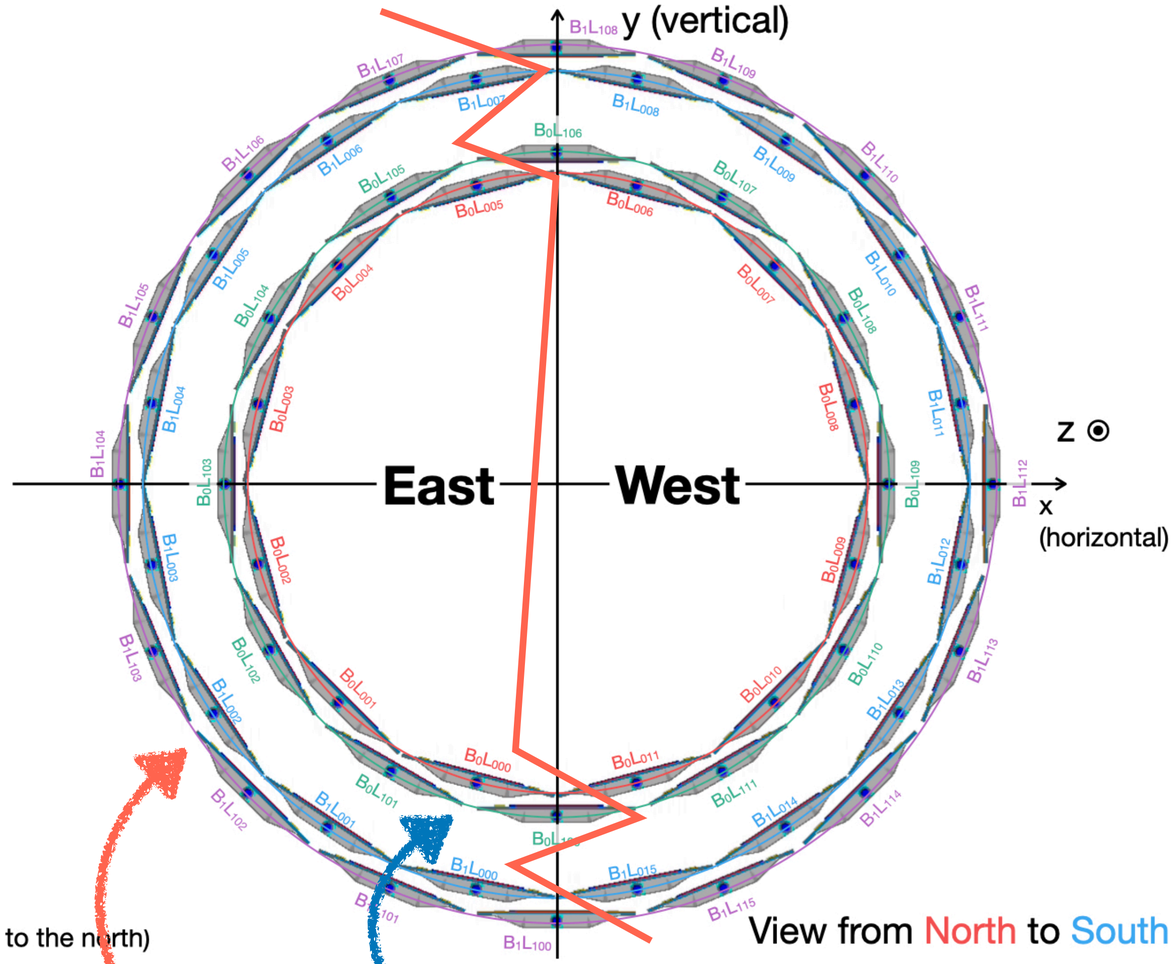
*Now we have 10k INTTRawHit available in the official production directory

INTT geometry

INTT: 2 sensors X 2 sides of half-ladders X 56 ladders = 224 sensors



Notation: B_xL_{yzz}
- x: Barrel ID (0 for inner or 1 for outer)
- y: Layer ID (0 for inner or 1 for outer)
- zz: Ladder ID (from 0 to 15)



Axis (Right-haded coordinate)
- x-axis: $\vec{y} \times \vec{z}$
- y-axis: Vertically upward direction
- z-axis: The blue beam direction (pointing to the north)