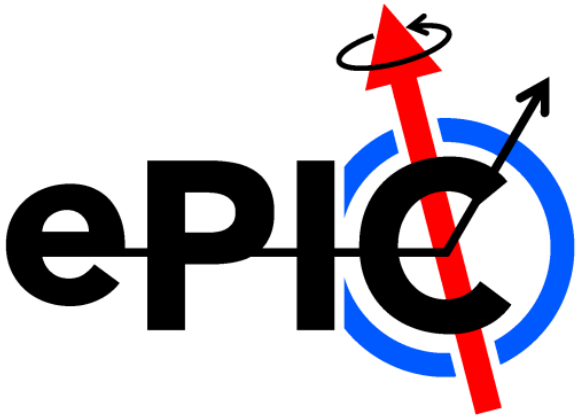


dRICH simulation – Noise

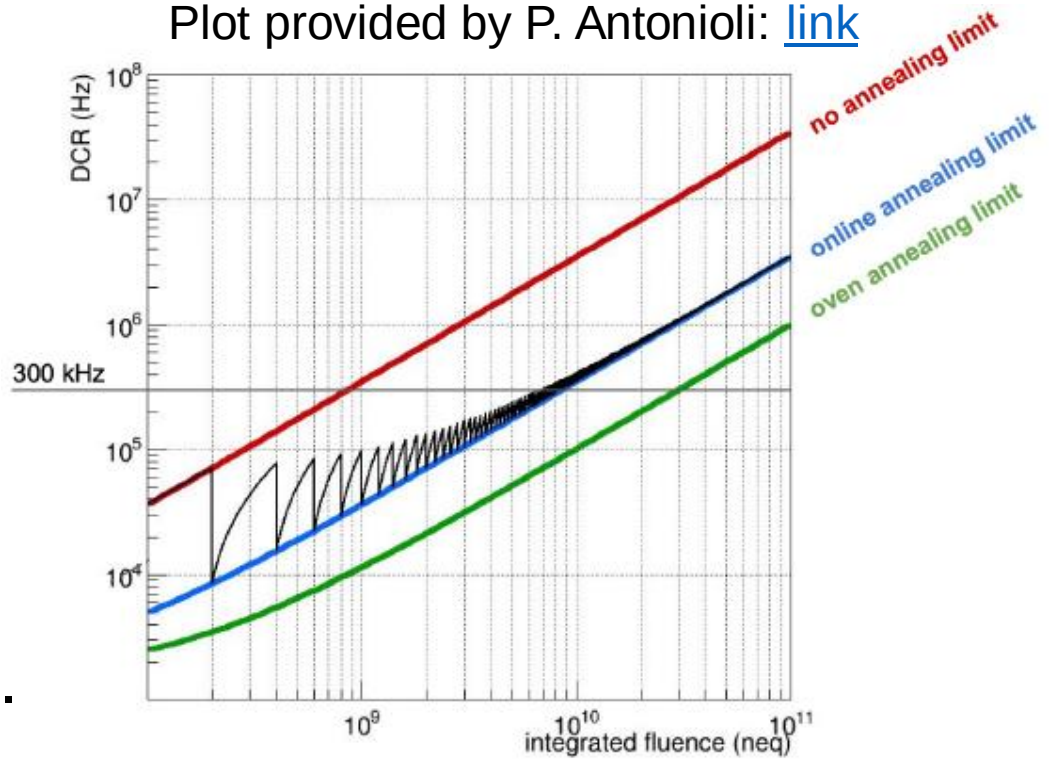
L. Dello Stritto (University and INFN Salerno)

dRICH Simulation meeting, 01/06/2023

Outline

- Implementation of the dRICH noise simulation.
- Performance study as a function of the DCR.

Plot provided by P. Antonioli: link



- Noise hits added in the PhotoMultiplierHitDigi.cc .
- Noise configurable: DRICH.cc

- Pull request: https://github.com/eic/EICrecon/pull/555

Algorithm

```
int k = p*m_num_sec*m_num_mod*m_num_px*m_num_px;

for (int i = 0; i < k; i++) {
  int isec = m_random.Uniform(0., m_num_sec);
  int imod = m_random.Uniform(0., m_num_mod);
  int x = m_random.Uniform(0., m_num_px);
  int y = m_random.Uniform(0., m_num_px);

  auto cellID = cellIDEncoding(isec, imod, x, y);
```

- Poissonian distribution of the noise. **k** noise hits expected:

$$k = noise\ rate * time\ window\ * n\ pixels$$

- Generation of ***k*** random cell IDs.

```
//build noise raw hits
if (m_cfg.enableNoise) {
  m_log->trace("{:=^70}"," BEGIN NOISE INJECTION ");
  float p = m_cfg.noiseRate*m_cfg.noiseTimeWindow;
  auto cellID_action = [this,&hit_groups] (auto id) {

    // cell time, signal amplitude
    double   amp  = m_cfg.speMean + m_rngNorm()*m_cfg.speError;
    TimeType time = m_cfg.noiseTimeWindow*m_rngUni();
    dd4hep::Position pos_hit_global = m_cellid_converter->position(id);
```

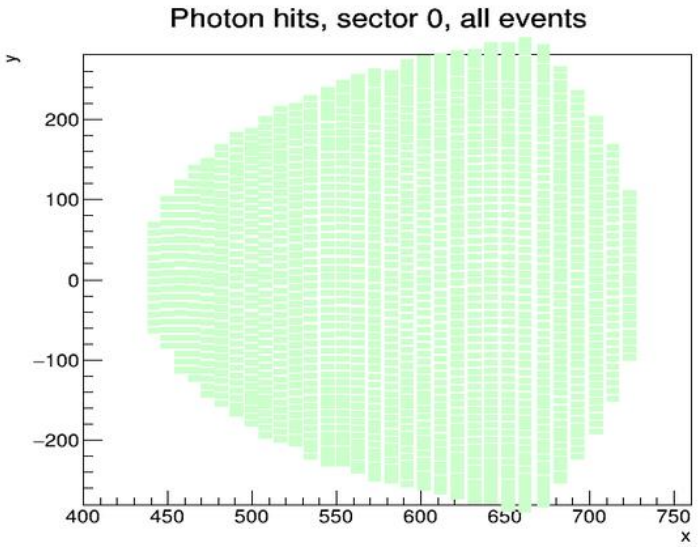
- Fill these cells with a noise hit.

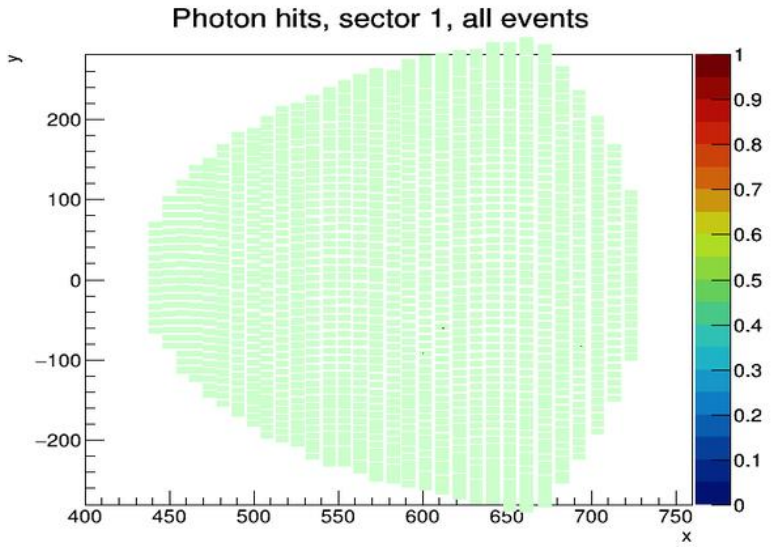
- **Time** associated to the noise hit randomly generated within the time widow.

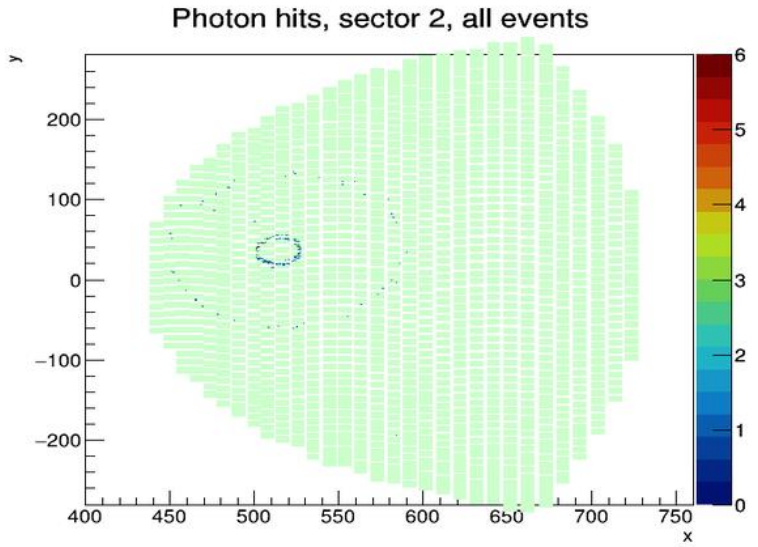
```
digi_cfg.enableNoise     = false;
digi_cfg.noiseRate       = 20000; // [Hz]
digi_cfg.noiseTimeWindow = 20.0 * dd4hep::ns; // [ns]
```

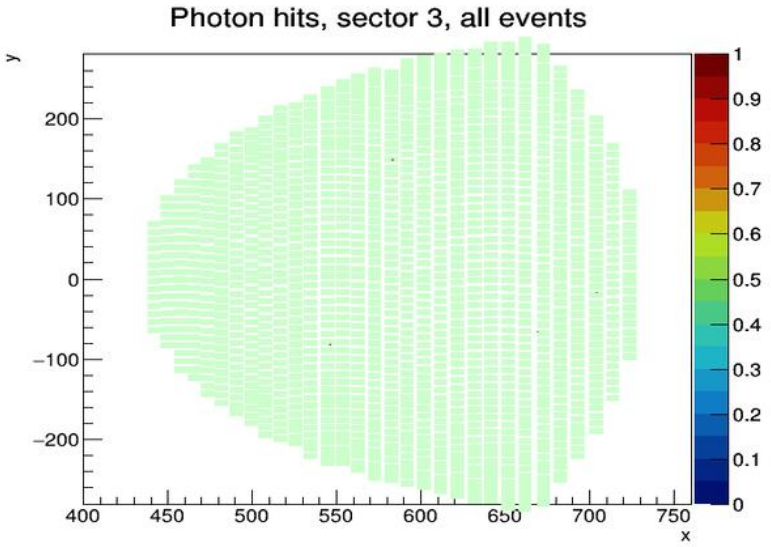
- Configurables in DRICH.cc.

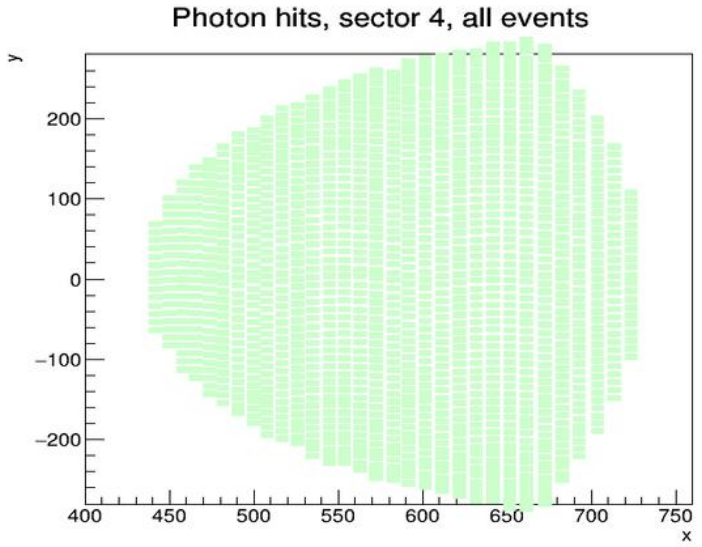
DRICH ring without noise

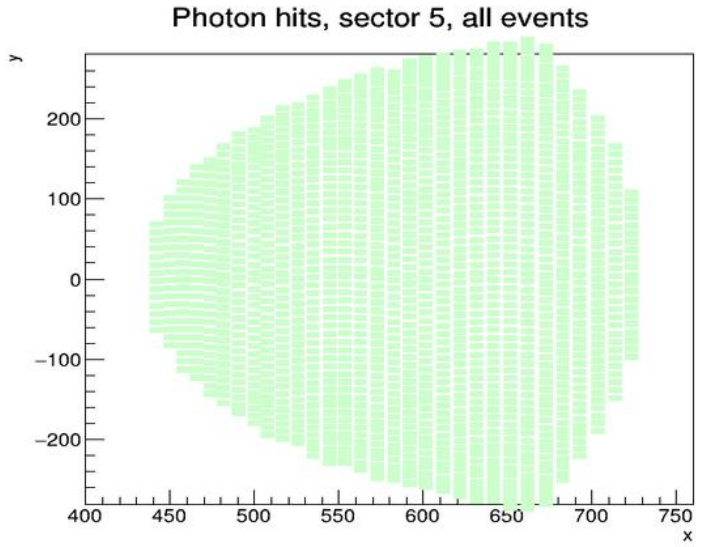






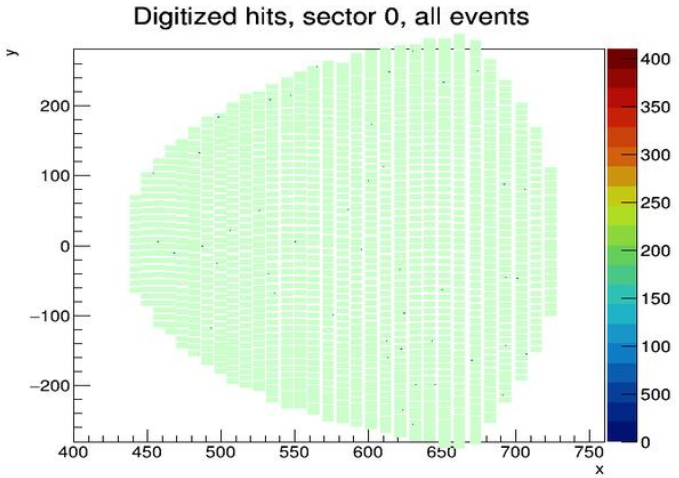


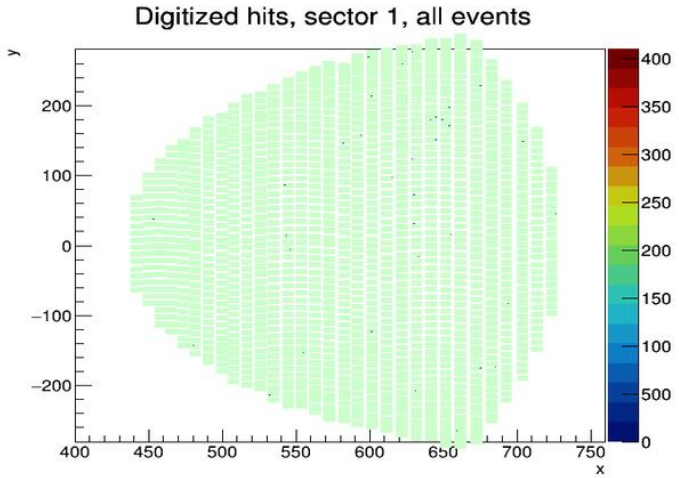


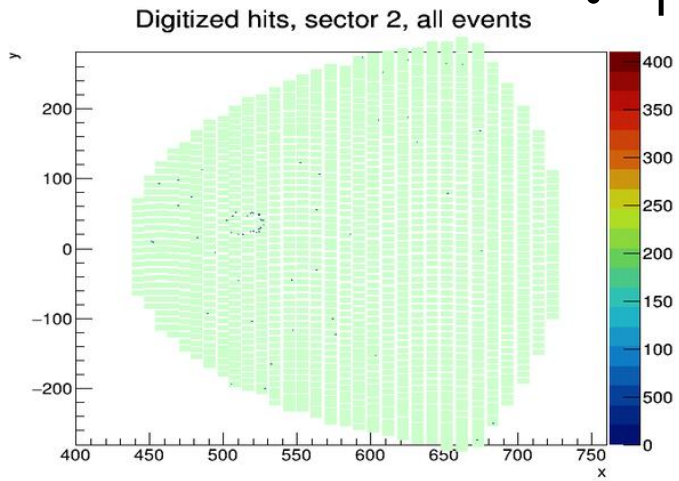


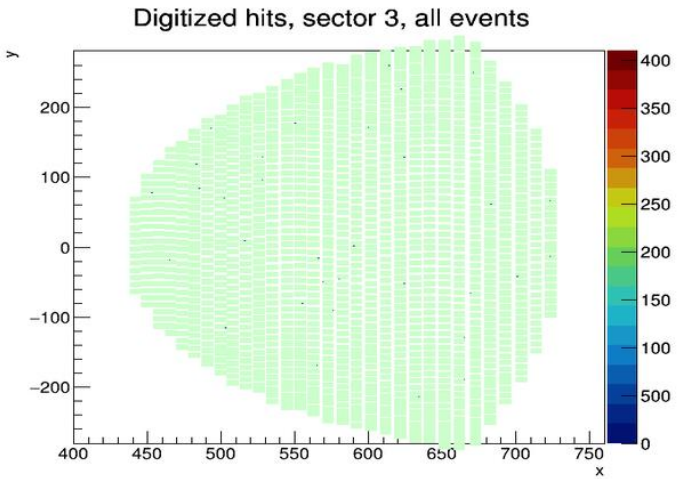
DRICH ring with noise

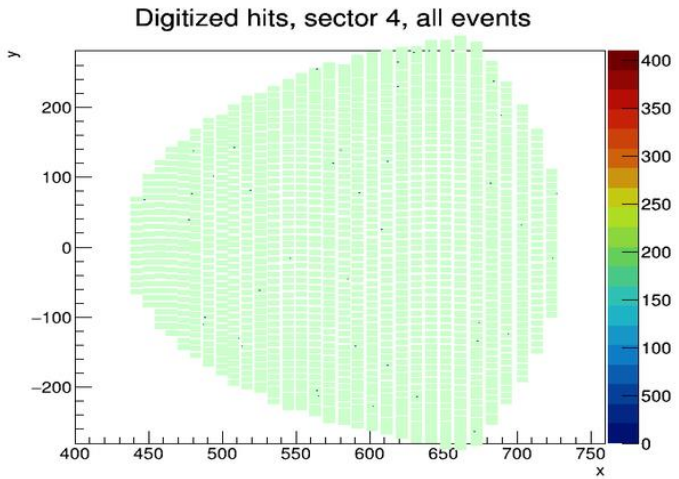
- Noise rate: 20 kHz*
- Time window: 20 ns

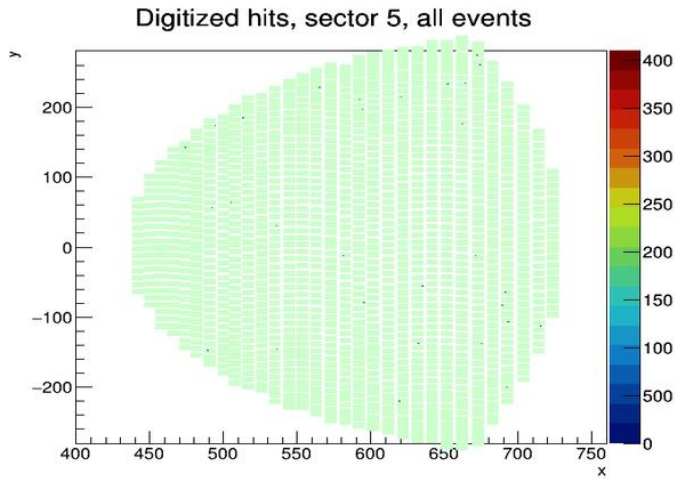








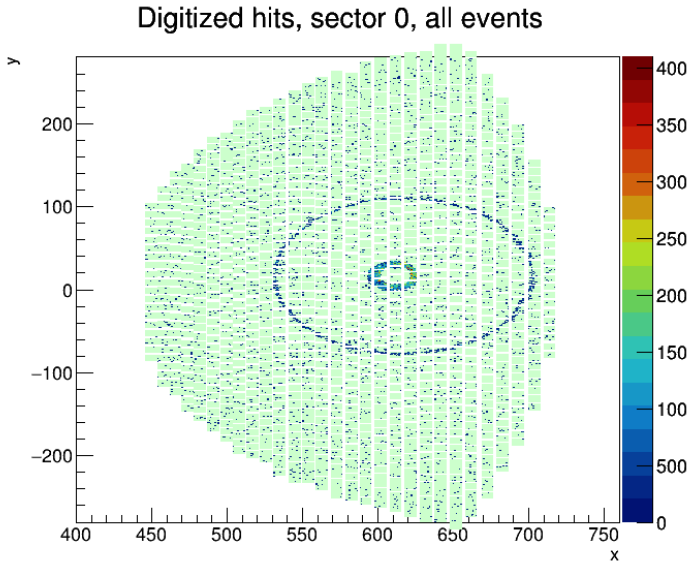


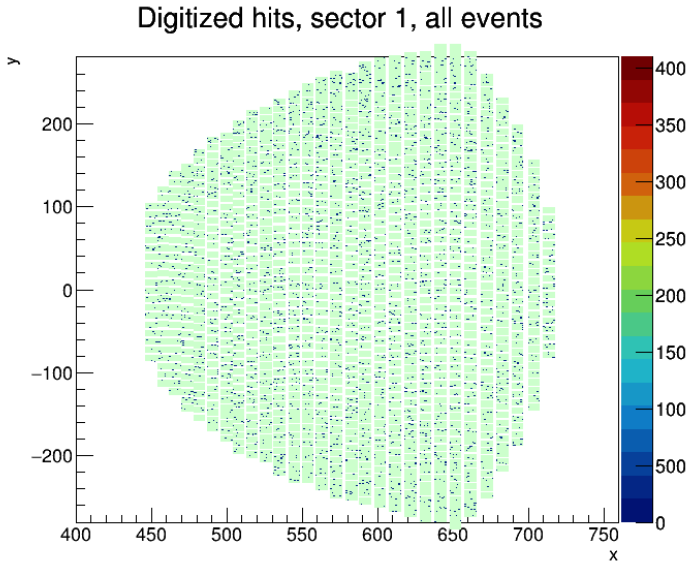


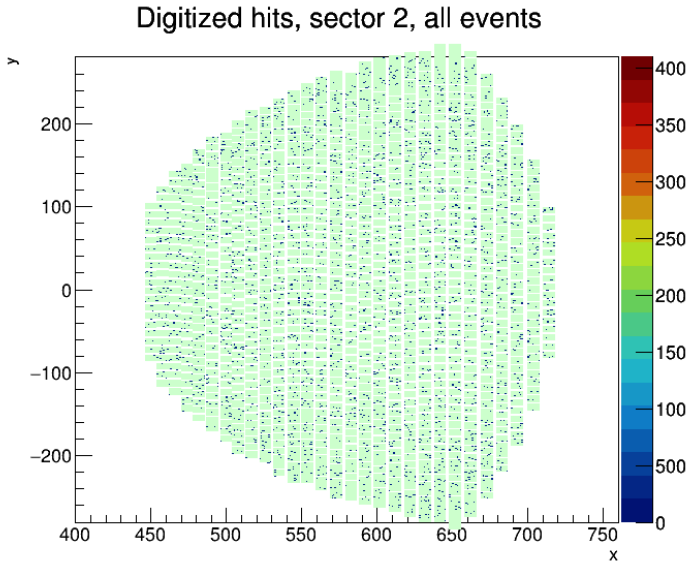
*Numbers provided by P. Antonioli: link

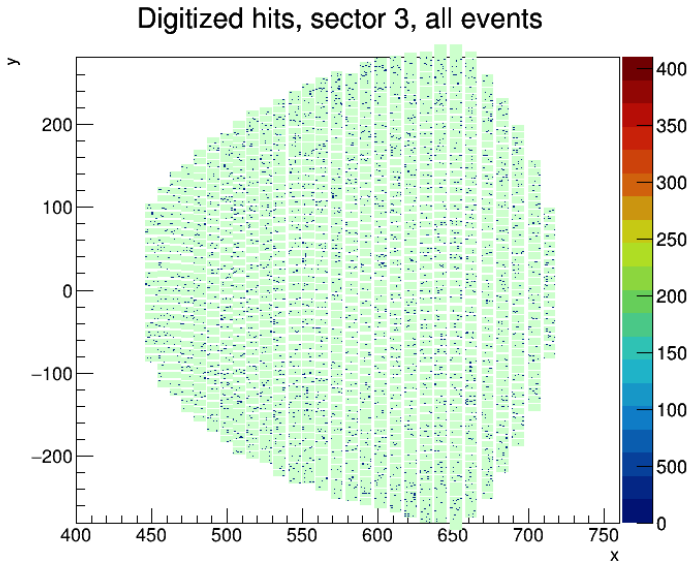
DRICH ring with noise

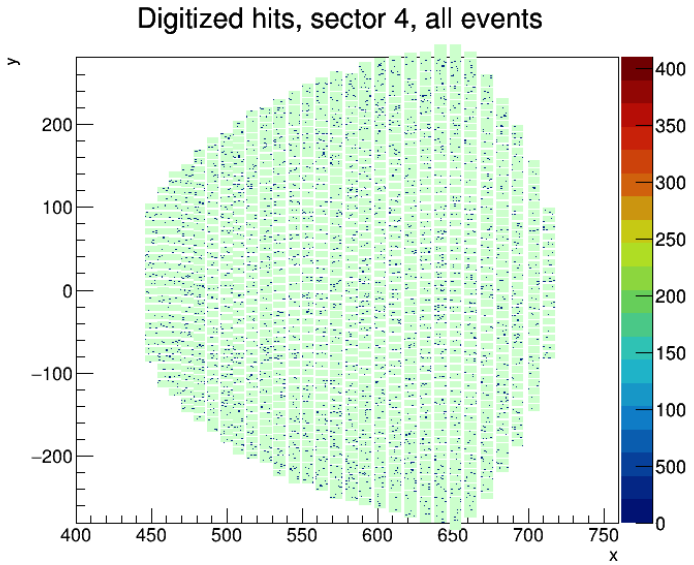
Gun with
100 pions.

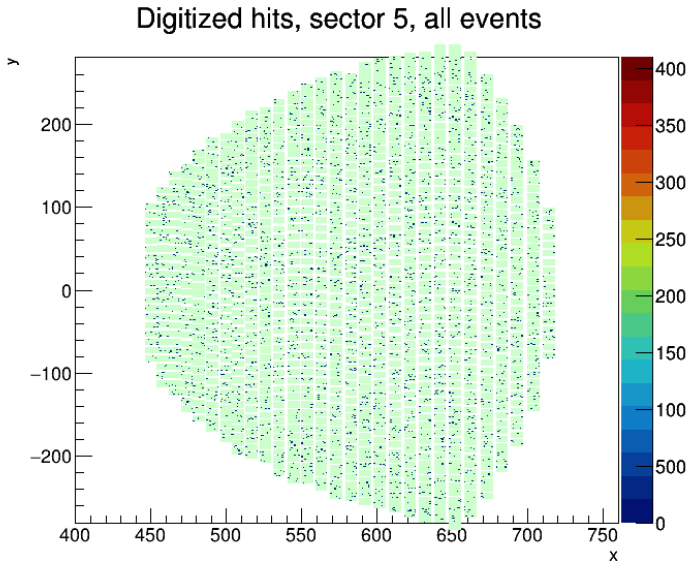






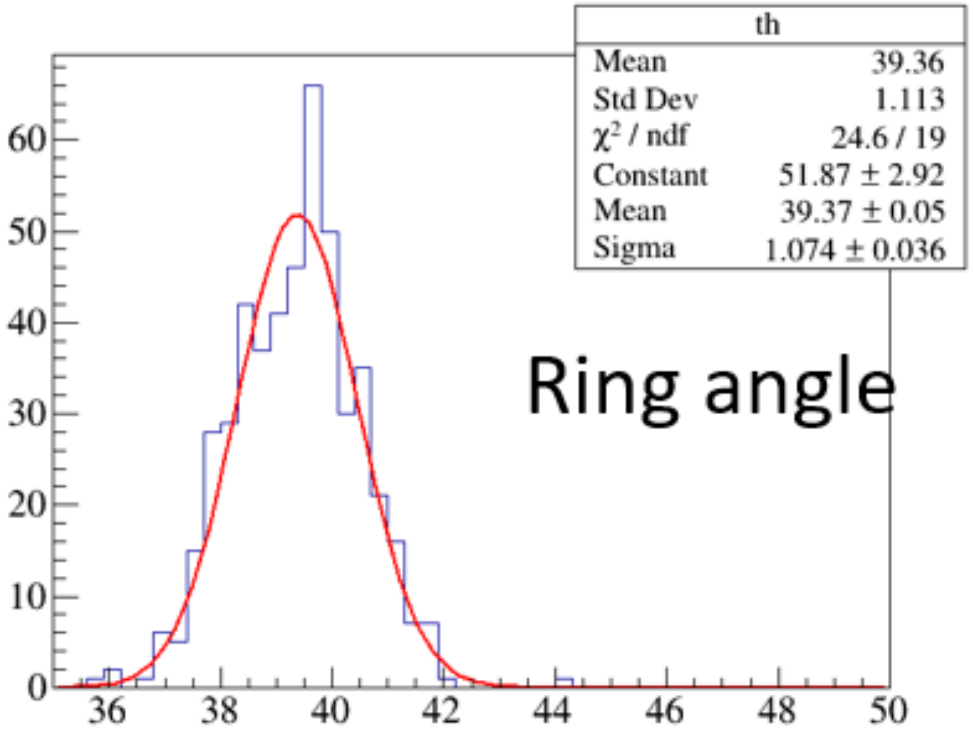






Outlook

- Test the performance at different noise rate, time windows, ...

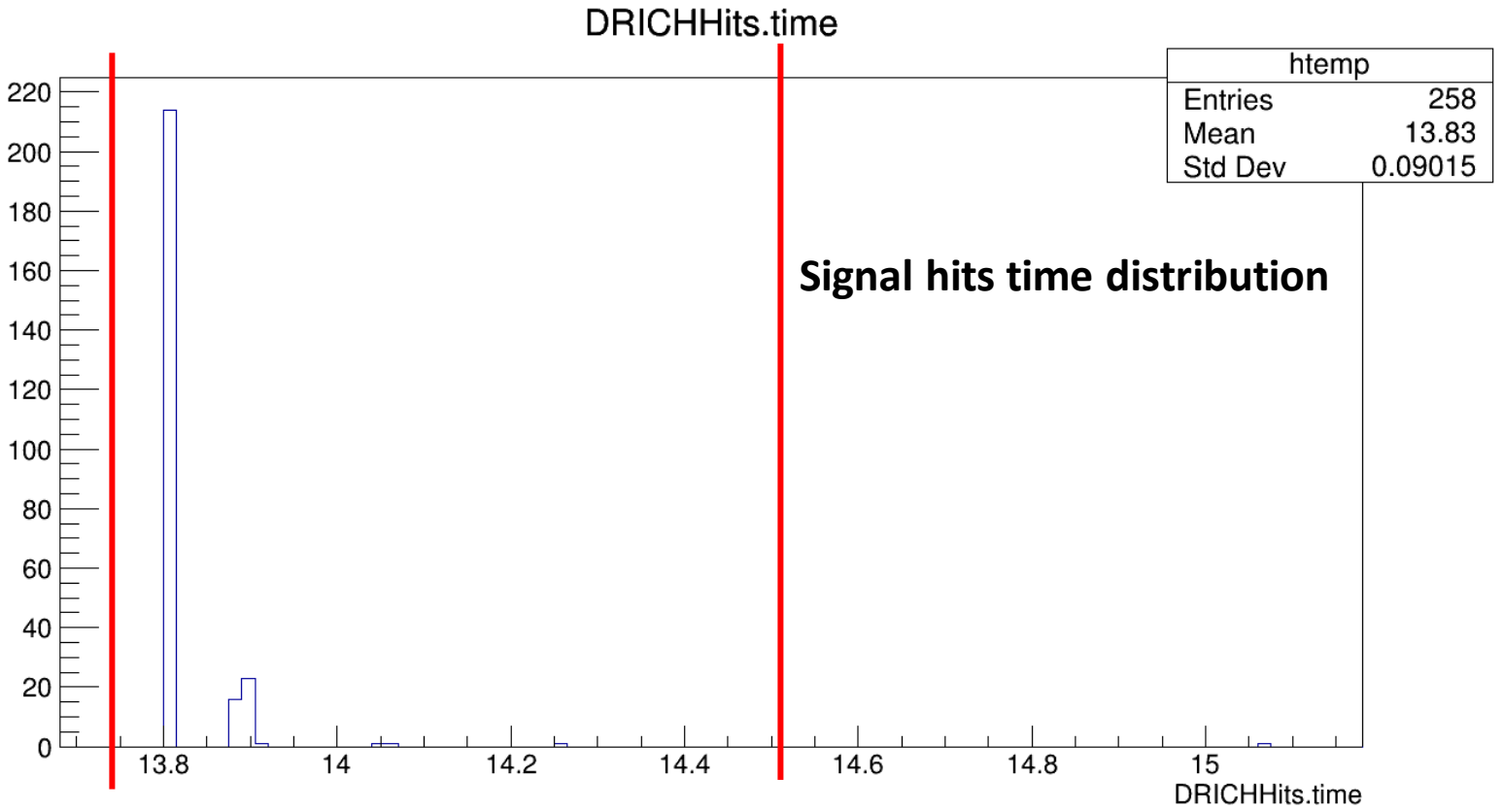


*Plot presented by Chandra: link

- Algorithm to reject noise hits.

Time shutter

- Reduce the noise with a “time shutter”.



Old Algorithm

- Loop over the sensors and generate random noise hits with a configurable rate in a given time window.
- Noise function:

```
bool  eicrecon::PhotoMultiplierHitDigi::Noise_Digits(float noiseRate, int timeWindow) const
{
  return (m_rngUni() < (noiseRate*timeWindow*dd4hep::ns));
}
```

- When the outcome of the Noise function is true, a noise hit is added to the raw hits:

```
if (Noise_Digits(m_cfg.noiseRate, m_cfg.timeWindow)){
  // cell time, signal amplitude
  double amp = m_cfg.speMean + m_rngNorm()*m_cfg.speError;
  double time = m_cfg.timeWindow*m_rngUni();
  auto pos_hit_global = m_cellid_converter->position(cellID);
  hit_groups_noise[cellID] = {HitData{1, amp + m_cfg.pedMean + m_cfg.pedError*m_rngNorm(), time, pos_hit_global}};
}
```