Study of Theta/Phi resolutions with the Smearing of Truth Parameters

Shyam Kumar, Annalisa Mastroserio, Domenico Elia
INFN Bari, Italy

https://indico.bnl.gov/event/17621/contributions/70622/subcontributions/2129/attachments/45442/76674/ePIC_collaboration_meeting_tracking_011023.pdf

Truth Parameters in Fun4All

int PHG4TrackFastSim::process_event(PHCompositeNode* /*topNode*/)

int PseudoPatternRecognition(const PHG4Particle* particle,

std::vector<PHGenFit::Measurement*>& meas_out, SvtxTrack* track_out,

TVector3& seed_pos,

TVector3& seed_mom, TMatrixDSym& seed_cov, const bool do_smearing = true);

https://github.com/sPHENIX-Collaboration/coresoftware/blob/master/simulation/g4simulation/g4trackfastsim/PHG4TrackFastSim.cc#L734

10% smearing in p_{true}

root [0] 3.0/180.*3.1416
(double) 0.052360000
0.05236 radian smearing in theta and Phi

```
int PHG4TrackFastSim::PseudoPatternRecognition(const PHG4Particle* particle,
                                               std::vector<PHGenFit::Measurement*>& meas_out,
                                               SvtxTrack* track_out,
                                               TVector3& seed_pos,
                                               TVector3& seed_mom, TMatrixDSym& seed_cov, const bool do_sm
{
  assert(track_out);

  seed_cov.ResizeTo(6, 6);

  seed_pos.SetXYZ(0, 0, 0);
  // reset the seed resolution to the approximate position resolution of the last detector
  seed_cov[0][0] = .1 * .1;
  seed_cov[1][1] = .1 * .1;
  seed_cov[2][2] = 30 * 30;
  //  for (int i = 0; i < 3; i++)
  //  {
  //    seed_cov[i][i] = _phi_resolution * _phi_resolution;
  //  }

  seed_mom.SetXYZ(0, 0, 10);
  for (int i = 3; i < 6; i++)
  {
    seed_cov[i][i] = 10;
  }

  if (particle)
  {
    TVector3 True_mom(particle->get_px(), particle->get_py(),
                      particle->get_pz());

    seed_mom.SetXYZ(particle->get_px(), particle->get_py(),
                    particle->get_pz());
    if (do_smearing)
    {
      const double momSmear = 3. / 180. * M_PI;  // rad
      const double momMagSmear = 0.1;            // relative

      seed_mom.SetMag(
          True_mom.Mag() + gsl_ran_gaussian(m_RandomGenerator,
                                            momMagSmear * True_mom.Mag()));
      seed_mom.SetTheta(True_mom.Theta() + gsl_ran_gaussian(m_RandomGenerator, momSmear));
      seed_mom.SetPhi(True_mom.Phi() + gsl_ran_gaussian(m_RandomGenerator, momSmear));
    }
  }
```

Initialization default

If there is a truth information

Option for smearing

Truth Parameters in ACTS

```
#pragma once

#include <Acts/Definitions/Units.hpp>

struct TrackParamTruthInitConfig {

    double m_maxVertexX       = 80  * Acts::UnitConstants::mm;
    double m_maxVertexY       = 80  * Acts::UnitConstants::mm;
    double m_maxVertexZ       = 200 * Acts::UnitConstants::mm;
    double m_minMomentum      = 100 * Acts::UnitConstants::MeV;
    double m_maxEtaForward    = 4.0;
    double m_maxEtaBackward   = 4.1;
    double m_momentumSmear    = 0.1;
```

Smearing momentum by 10% (Gaussian)

Track Parameters

$(l_0, l_1, \theta, \phi, q/p)$

↓ Extrapolate to Vertex

$(DCA_{xy}, DCA_z, \theta, \phi, q/p)$

Introducing Theta/Phi smearing

```
// modify initial momentum to avoid bleeding truth to results when fit fails
const auto pinit = pmag*(1.0 + m_cfg.m_momentumSmear * m_normDist(generator));
```

```
// build some track cov matrix
Acts::BoundSymMatrix cov                    = Acts::BoundSymMatrix::Zero();
cov(Acts::eBoundLoc0, Acts::eBoundLoc0)     = 1000*um*1000*um;
cov(Acts::eBoundLoc1, Acts::eBoundLoc1)     = 1000*um*1000*um;
cov(Acts::eBoundPhi, Acts::eBoundPhi)       = 0.05*0.05;
cov(Acts::eBoundTheta, Acts::eBoundTheta)   = 0.01*0.01;
cov(Acts::eBoundQOverP, Acts::eBoundQOverP) = (0.1*0.1) / (GeV*GeV);
cov(Acts::eBoundTime, Acts::eBoundTime)     = 10.0e9*ns*10.0e9*ns;
```

Covariance

```
Acts::BoundVector  params;
params(Acts::eBoundLoc0)   = 0.0 * mm ;  // cylinder radius
params(Acts::eBoundLoc1)   = 0.0 * mm ;  // cylinder length
params(Acts::eBoundPhi)    = phi;
params(Acts::eBoundTheta)  = theta;
params(Acts::eBoundQOverP) = charge / (pinit * GeV);
params(Acts::eBoundTime)   = part->getTime() * ns;
```

```
        double m_momentumSmear    = 0.1;
        double m_angleSmear       = 1.0/180.*M_PI; // radian
    }
    return nullptr;
}

// modify initial momentum to avoid bleeding truth to results when fit fails
const auto pinit = pmag*(1.0 + m_cfg.m_momentumSmear * m_normDist(generator));
const auto thetainit = theta + m_cfg.m_angleSmear * m_normDist(generator);
const auto phiinit = phi + m_cfg.m_angleSmear * m_normDist(generator);

Acts::BoundVector  params;
params(Acts::eBoundLoc0)   = 0.0 * mm ;  // cylinder radius
params(Acts::eBoundLoc1)   = 0.0 * mm ;  // cylinder length
params(Acts::eBoundPhi)    = phiinit;
params(Acts::eBoundTheta)  = thetainit;
params(Acts::eBoundQOverP) = charge / (pinit * GeV);
params(Acts::eBoundTime)   = part->getTime() * ns;
```

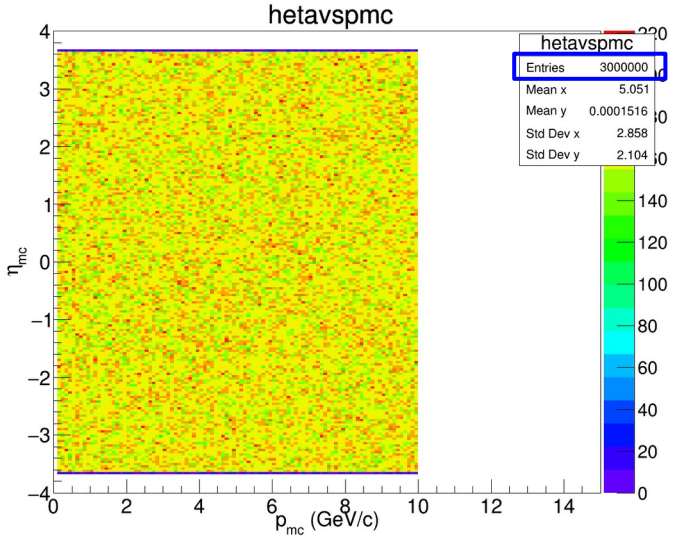
Parameter Initialization

https://github.com/eic/EICrecon/blob/main/src/algorithms/tracking/TrackParamTruthInitConfig.h

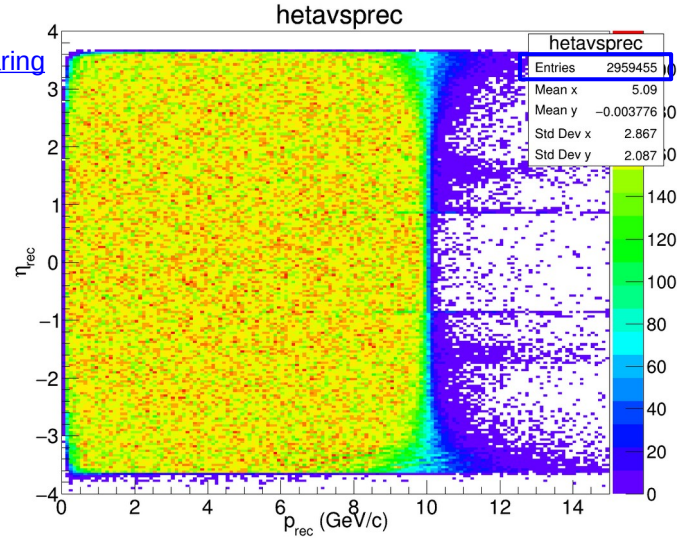
https://github.com/eic/EICrecon/blob/main/src/algorithms/tracking/TrackParamTruthInit.cc

Simulation Details

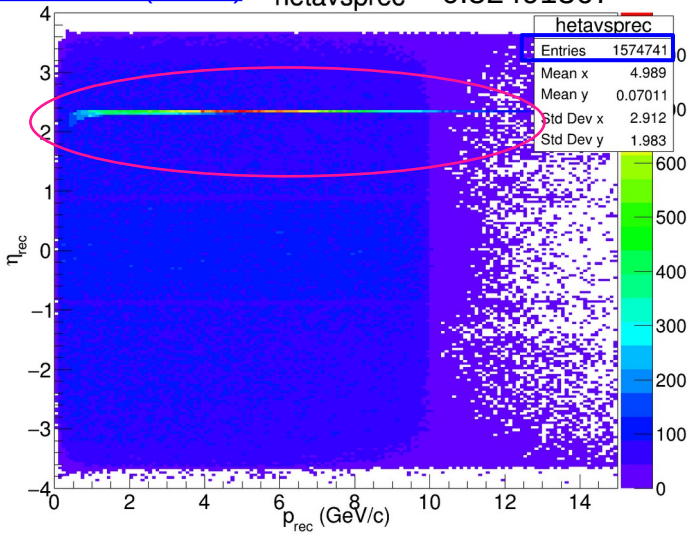
3M Pi+ Simulation with momentum [0.1,10.] GeV/c with epic_brycecanyon geometry with latest epic software



No Theta/Phi Smearing



Smearing = 3.0*Pi/180. (fun4All)



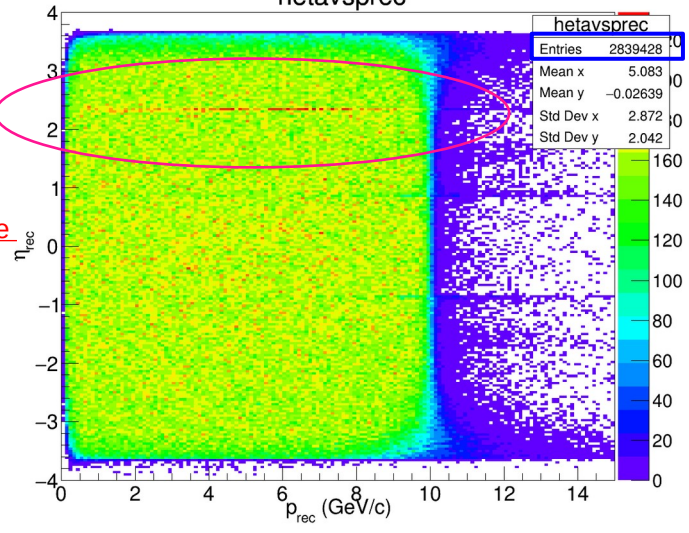
hetavsprec 0.52491367

Structures

Smearing Theta/Phi implies losing efficiency because of track finding

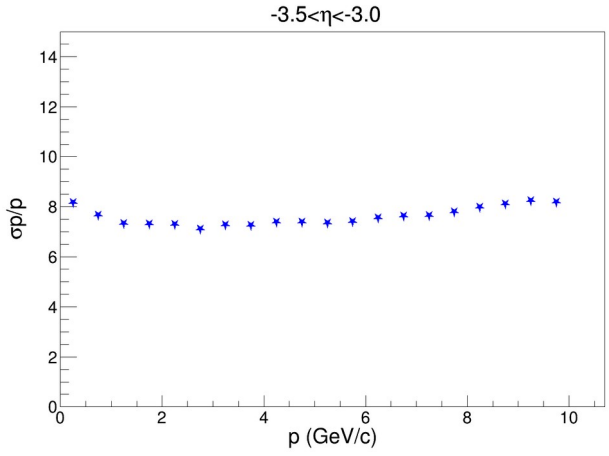
Warning No Tracks found !!

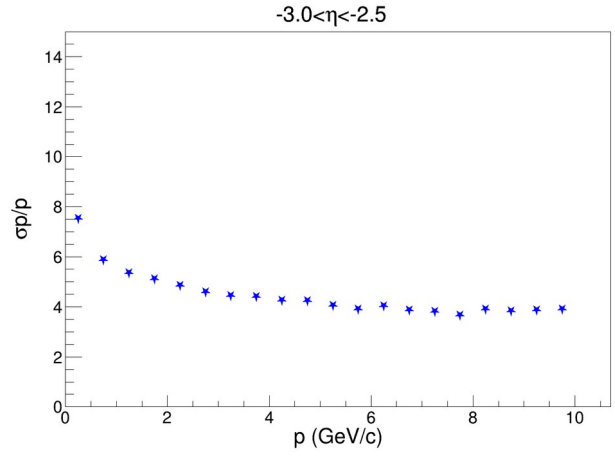
Smearing = 1.0*Pi/180.

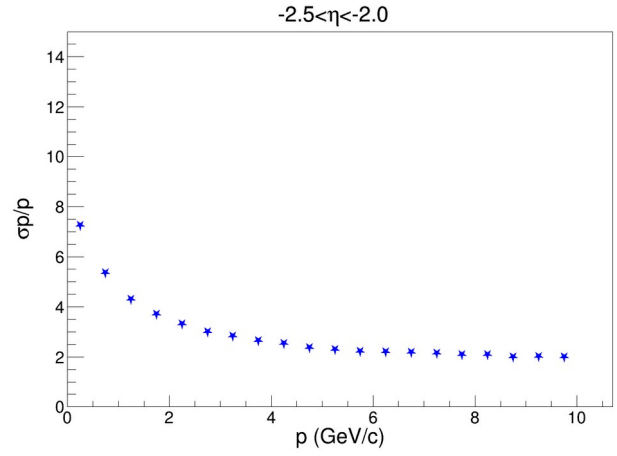


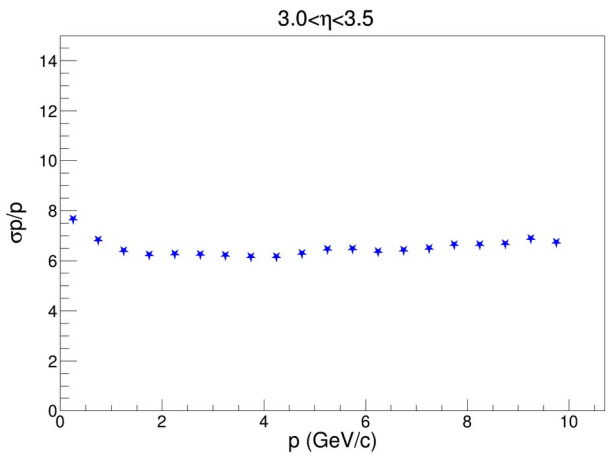
hetavsprec 0.94647600

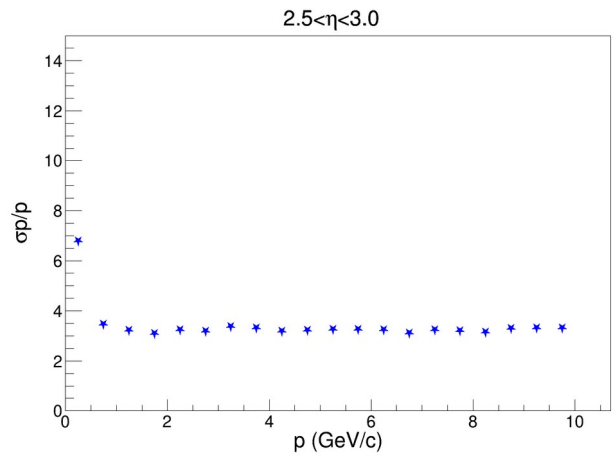
Momentum Resolution (Only Momentum Smearing)

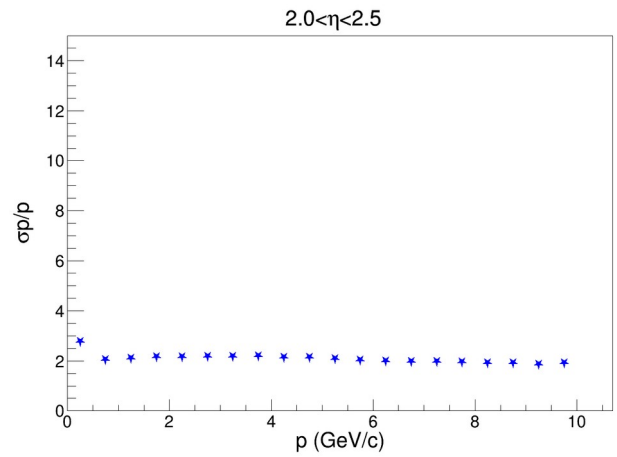




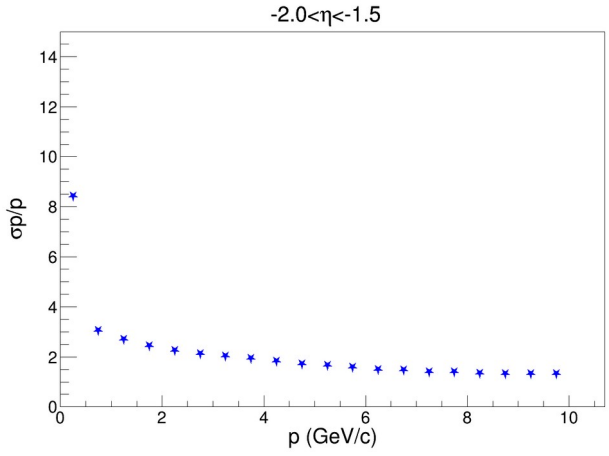


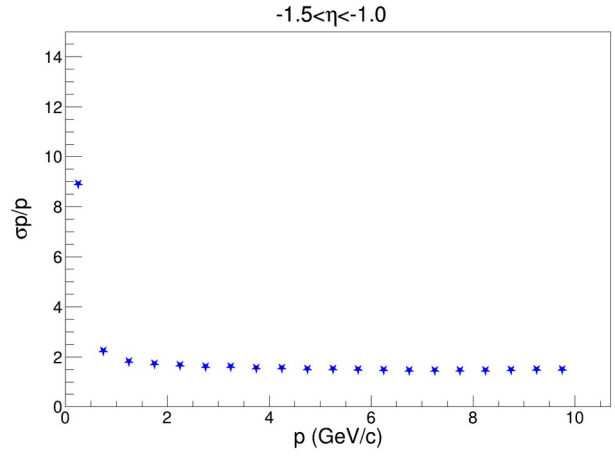


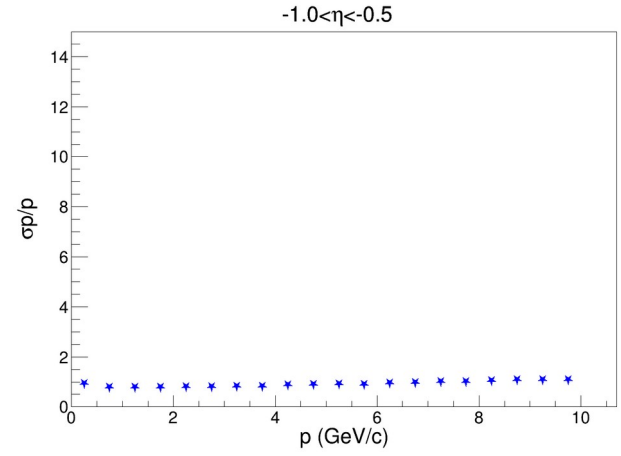


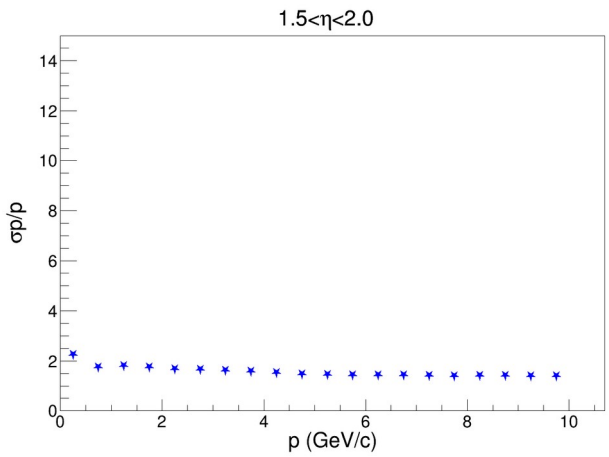


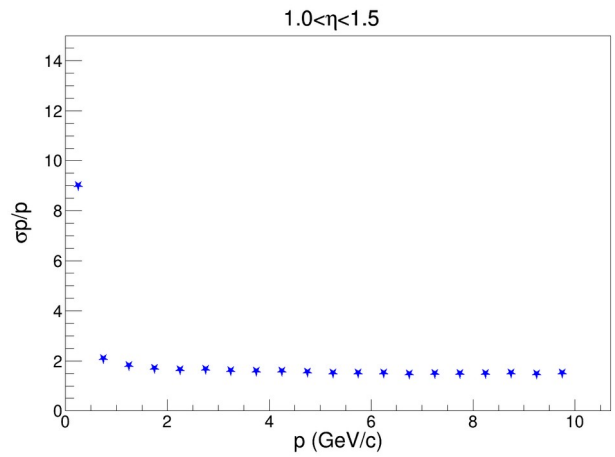
Momentum Resolution (Only Momentum Smearing)

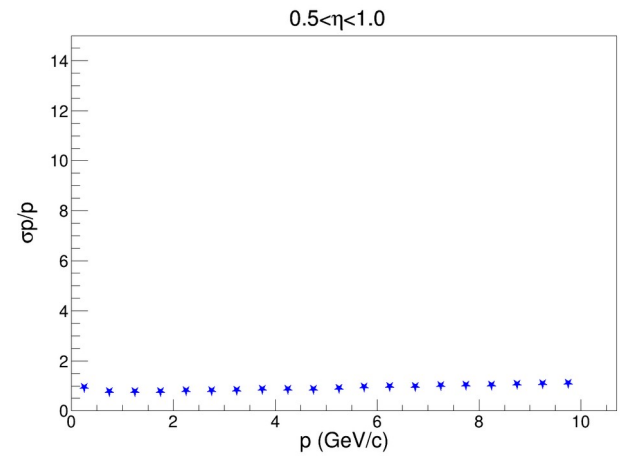






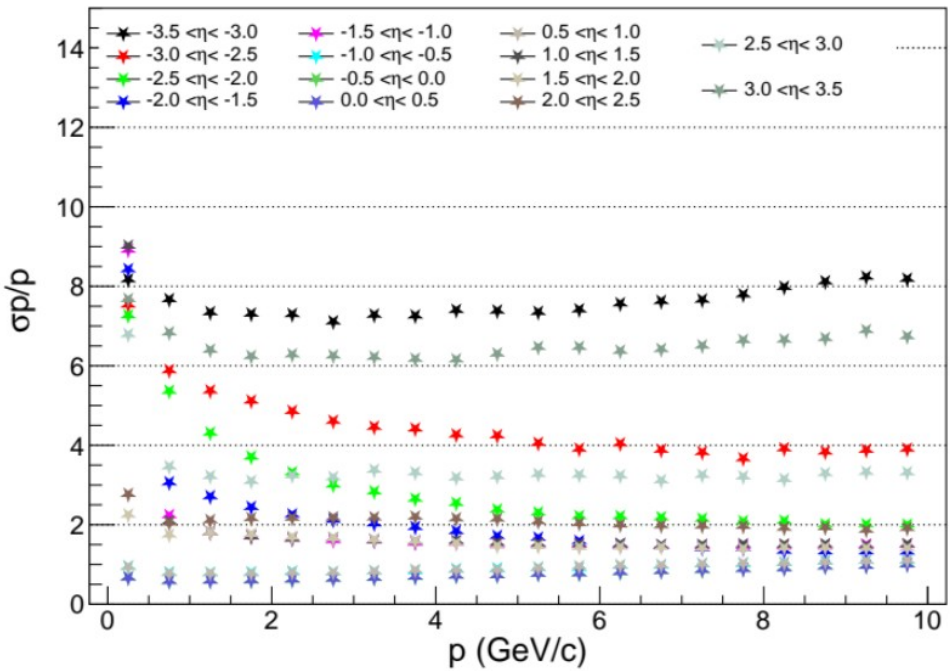


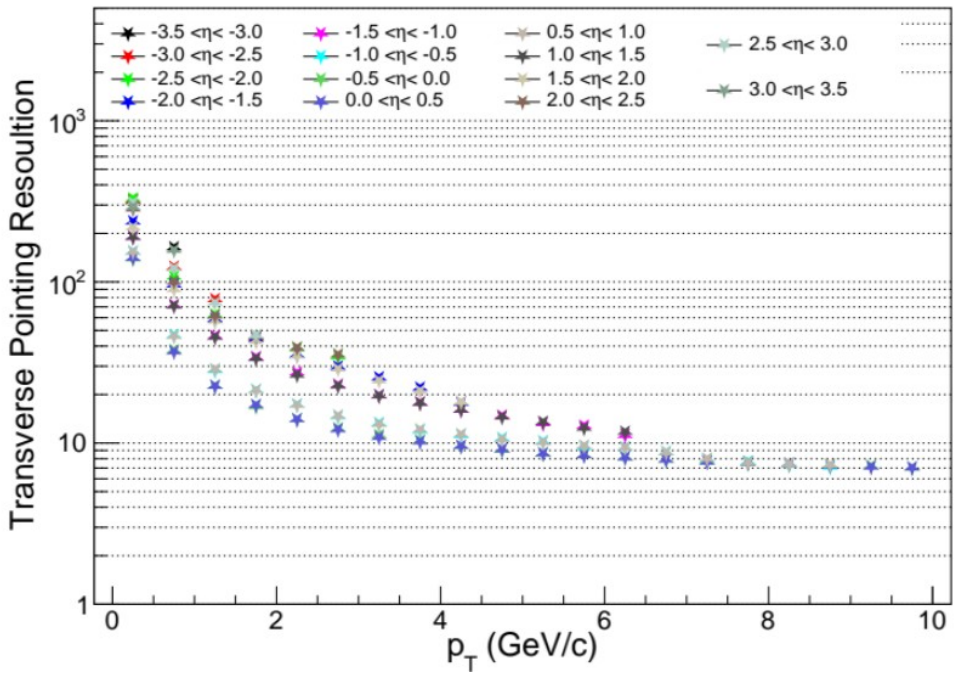




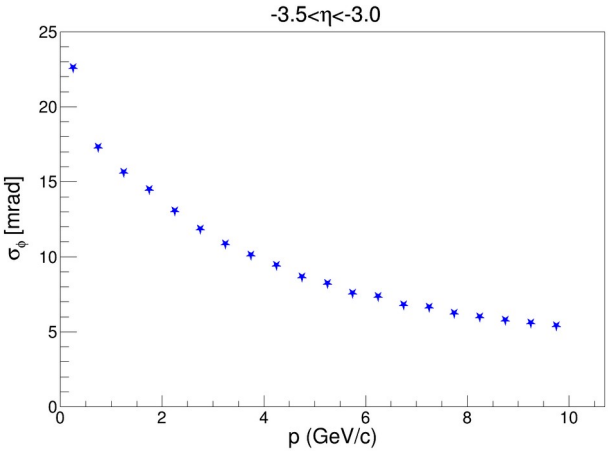
Momentum Resolution (Only Momentum Smearing)

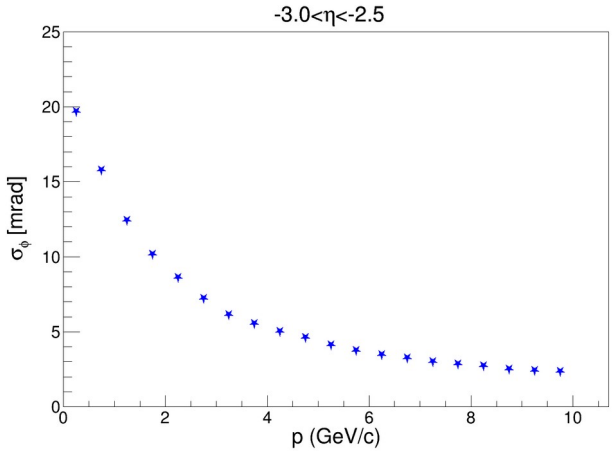
https://indico.bnl.gov/event/18272/contributions/72753/attachments/45921/79986/EPIC_Meeting_Shyam9Feb23.pdf

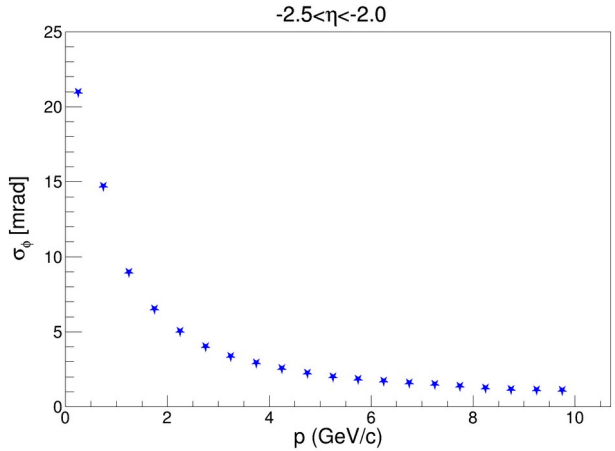


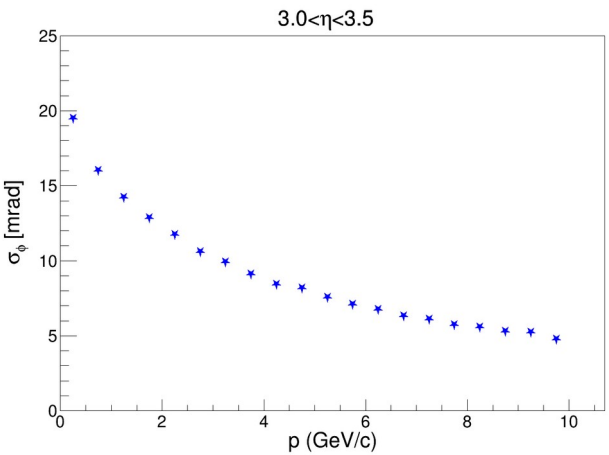


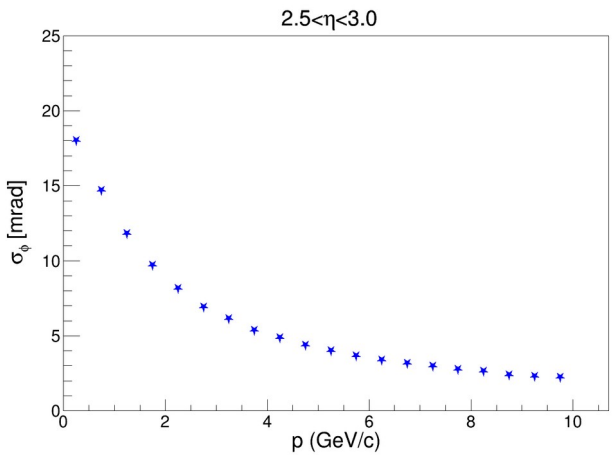
Phi Resolution (Only Momentum Smearing)

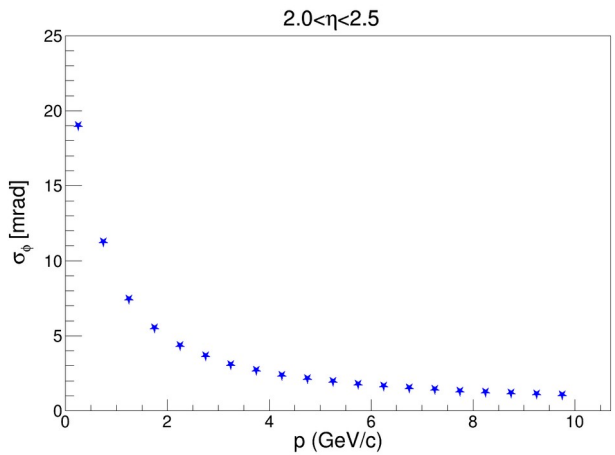




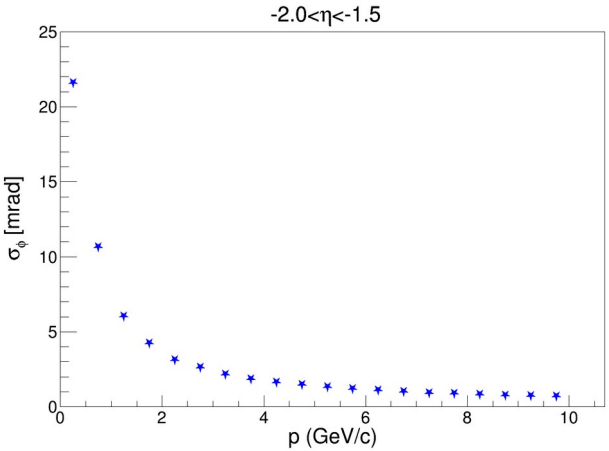


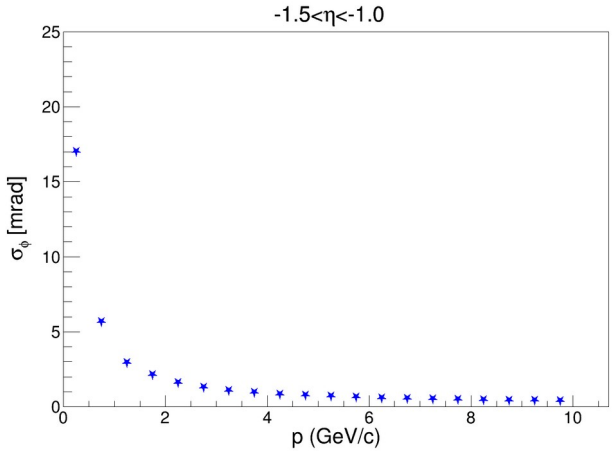


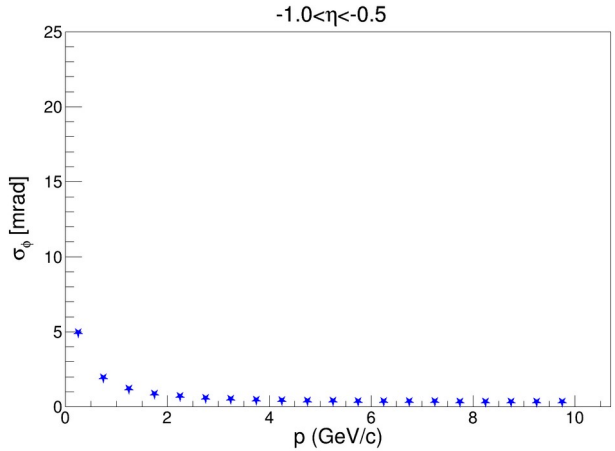


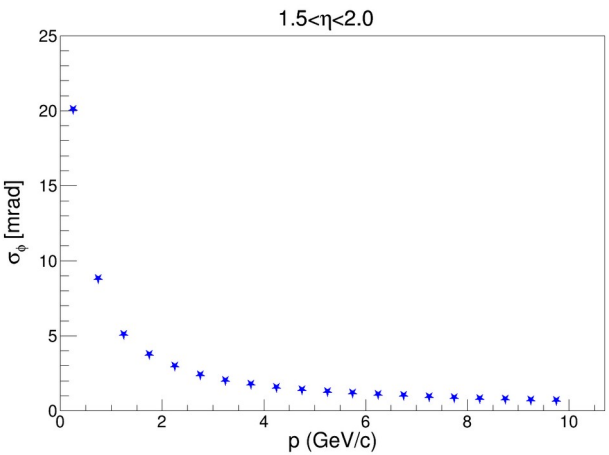


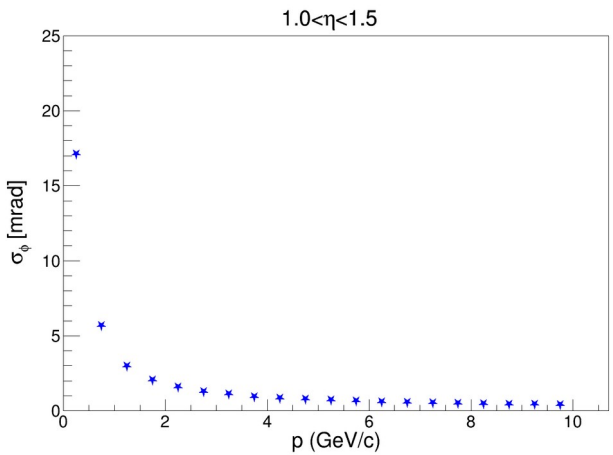
Phi Resolution (Only Momentum Smearing)

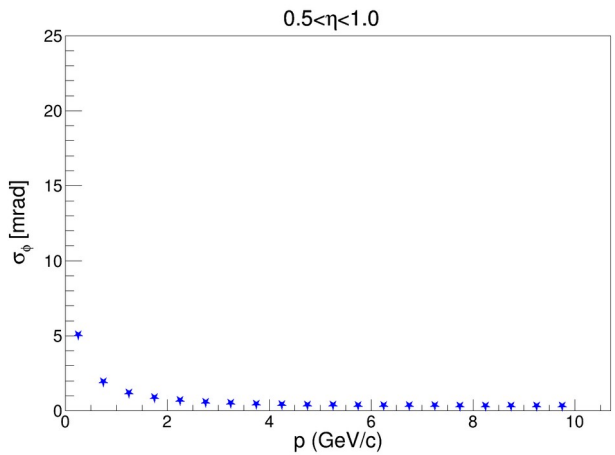




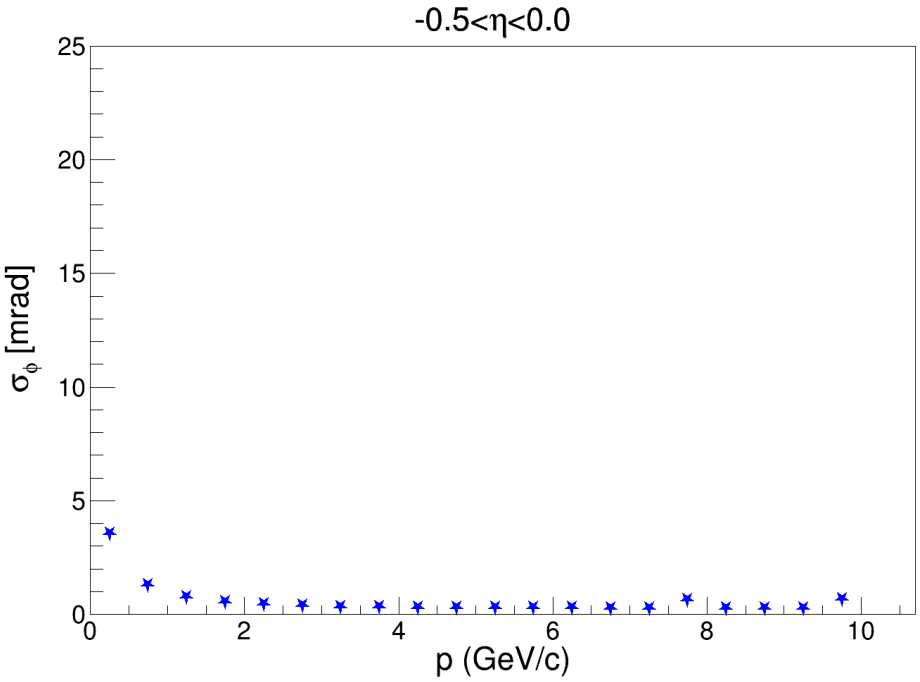


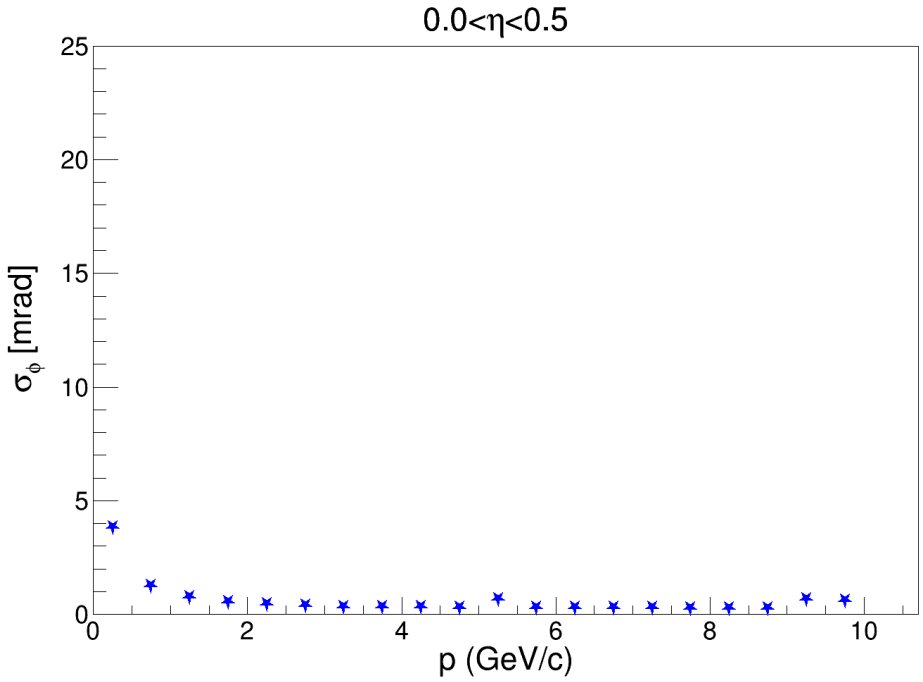




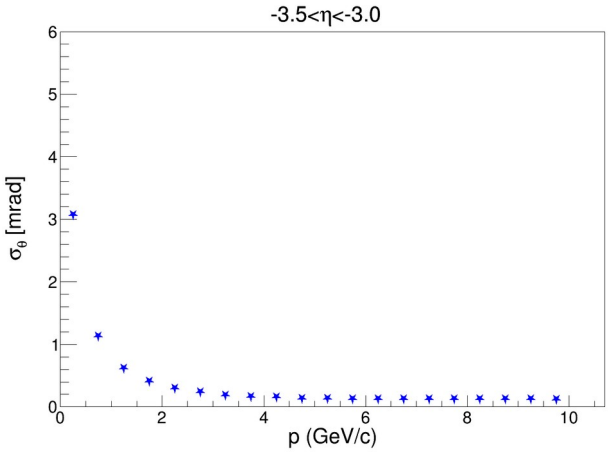


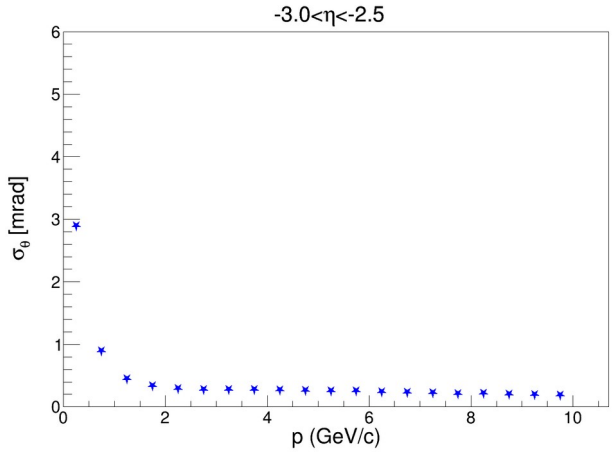
Phi Resolution (Only Momentum Smearing)

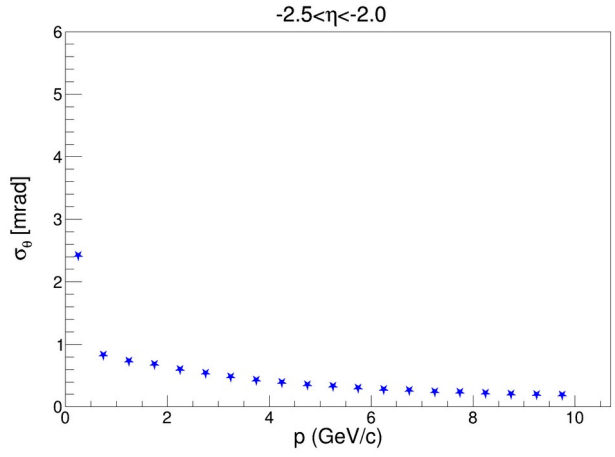


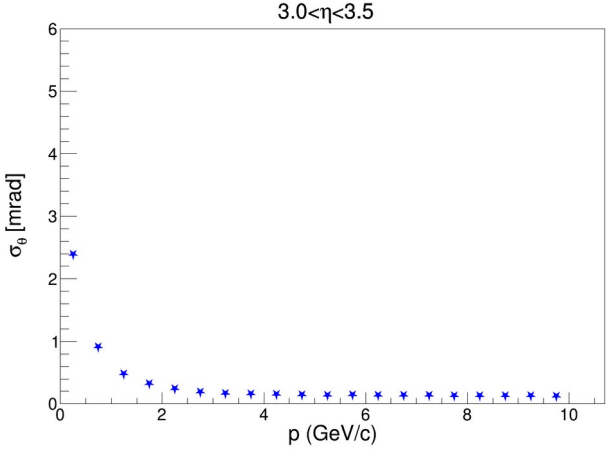


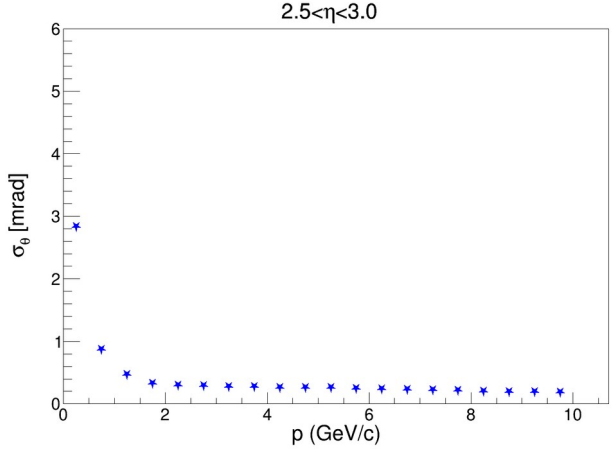
Theta Resolution (Only Momentum Smearing)

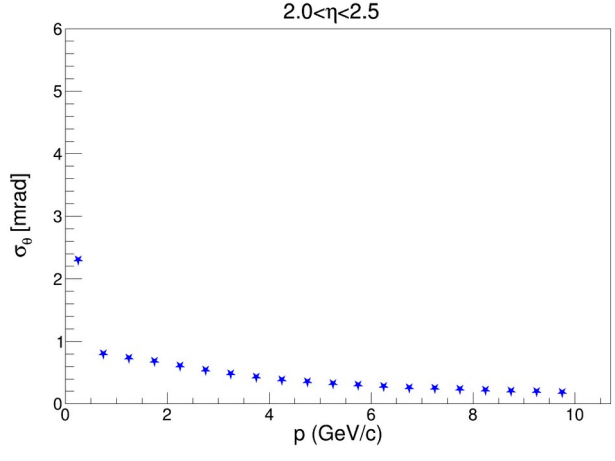




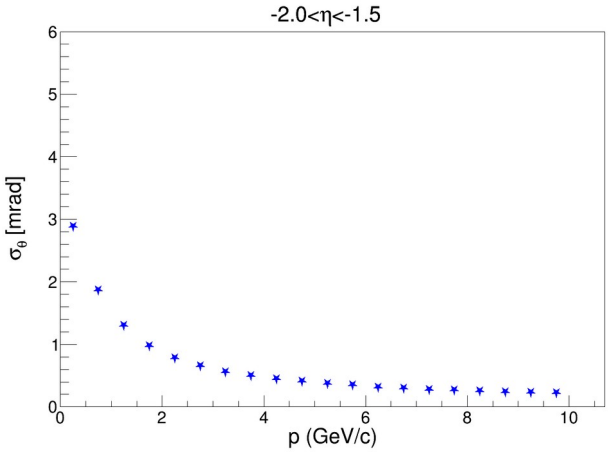


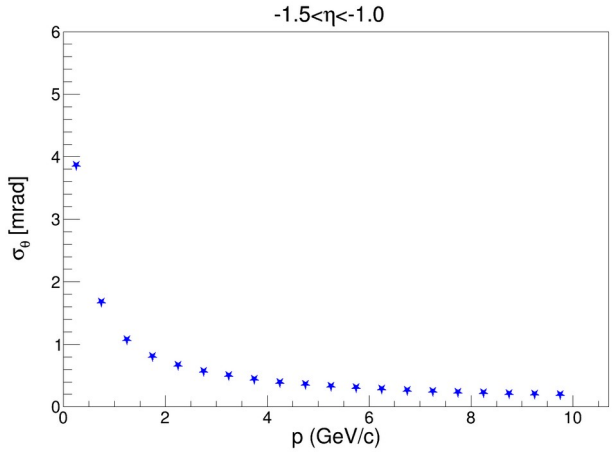


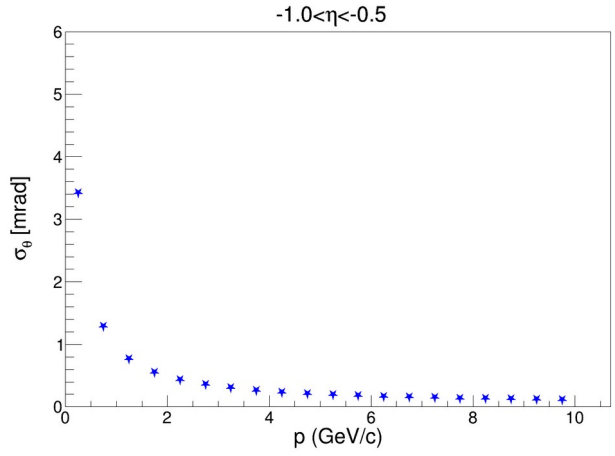


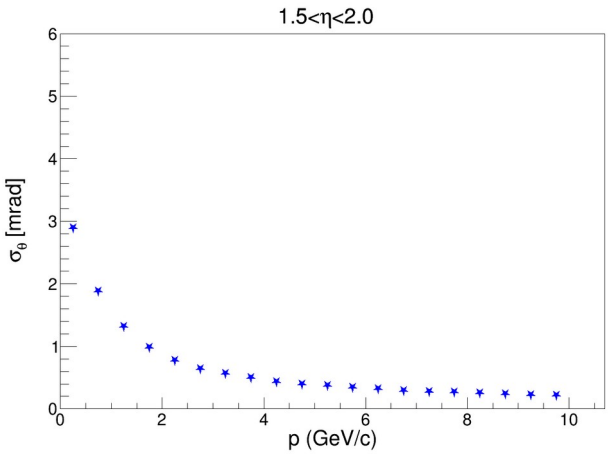


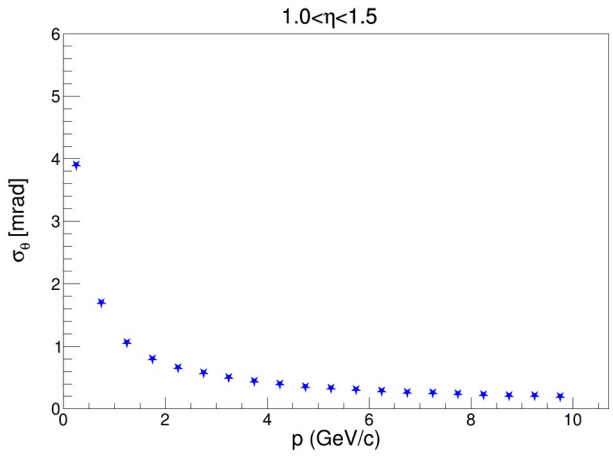
Theta Resolution (Only Momentum Smearing)

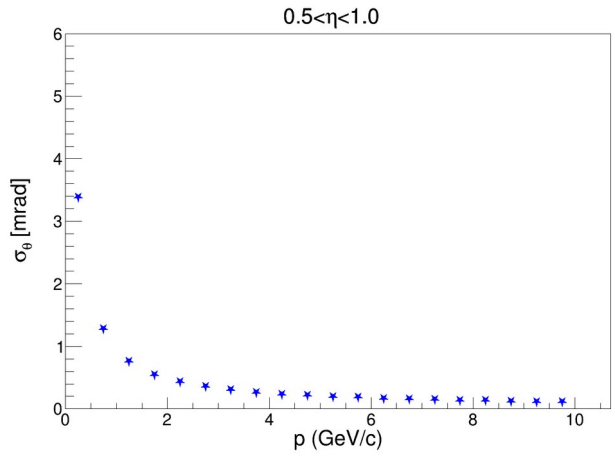




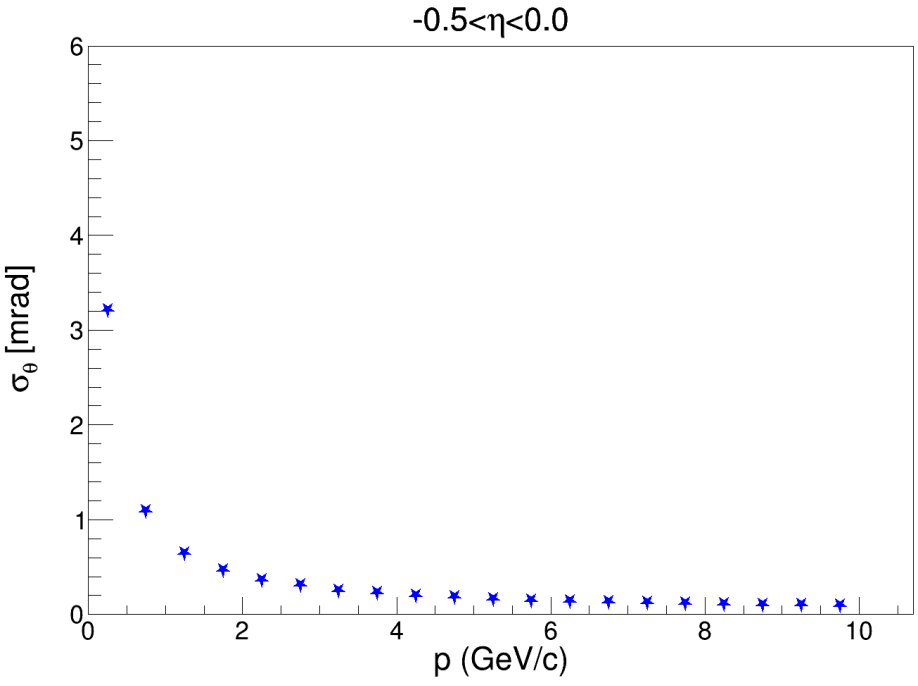


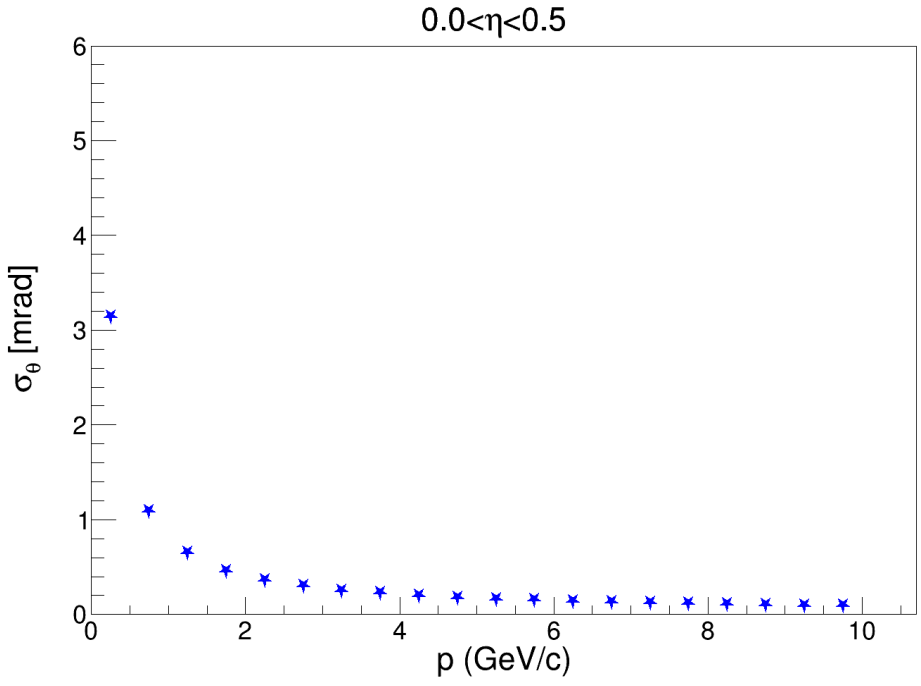






Theta Resolution (Only Momentum Smearing)

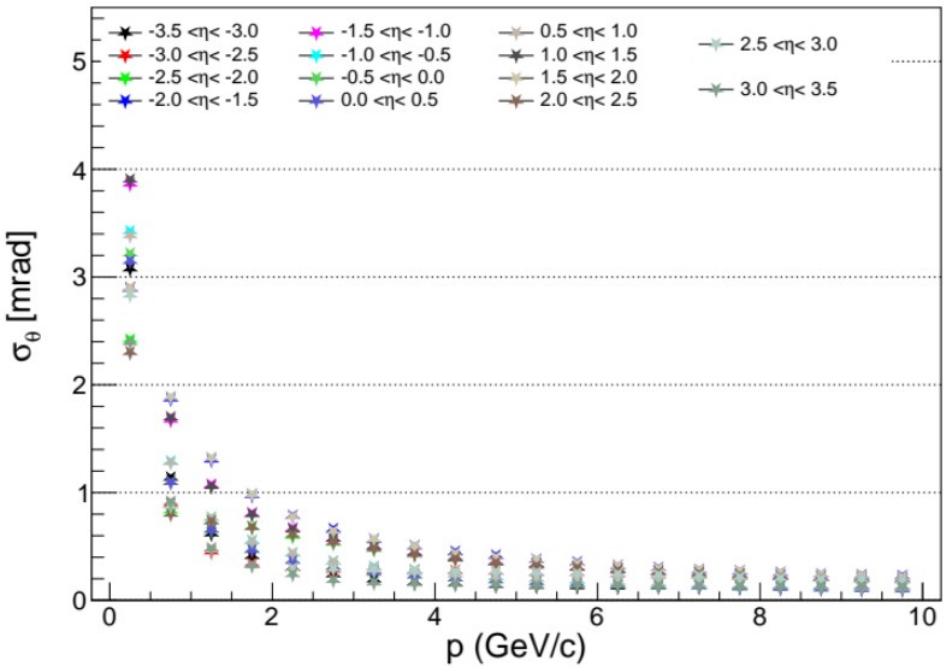


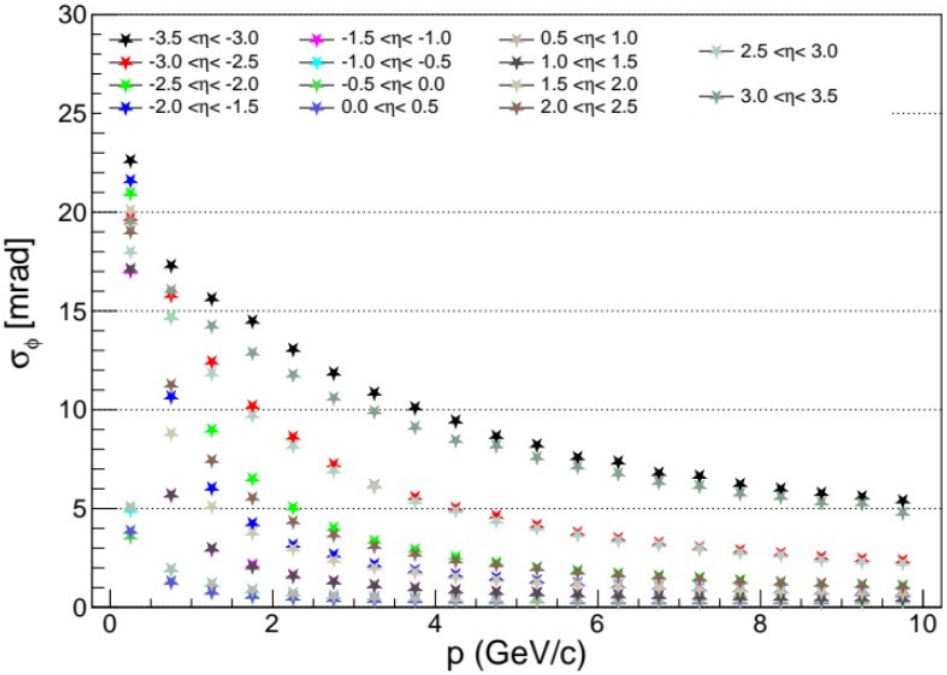


Results (Only Momentum Smearing)

```
mom_Mc.SetXYZ(px_mc[j],py_mc[j],pz_mc[j]);
Double_t theta_mc = mom_Mc.Theta(); Double_t phi_mc = mom_Mc.Phi();
mom_Rec.SetXYZ(px_rec[j],py_rec[j],pz_rec[j]);
Double_t theta_rec = mom_Rec.Theta(); Double_t phi_rec = mom_Rec.Phi();
```

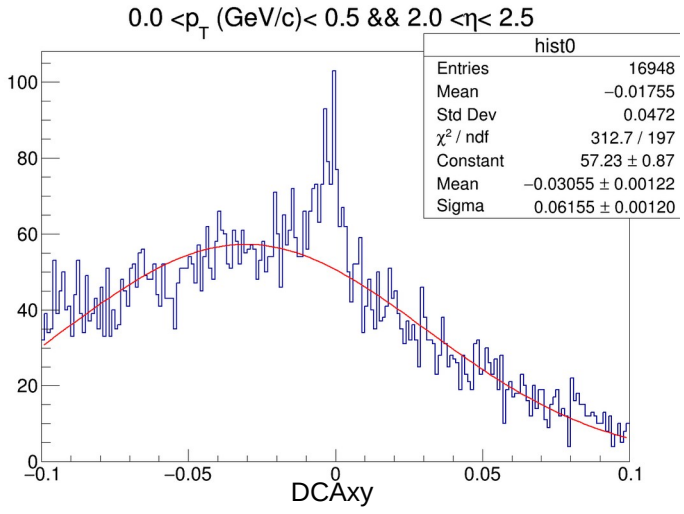
```
Double_t theta_resol = theta_rec-theta_mc;
Double_t phi_resol = phi_rec-phi_mc;
```

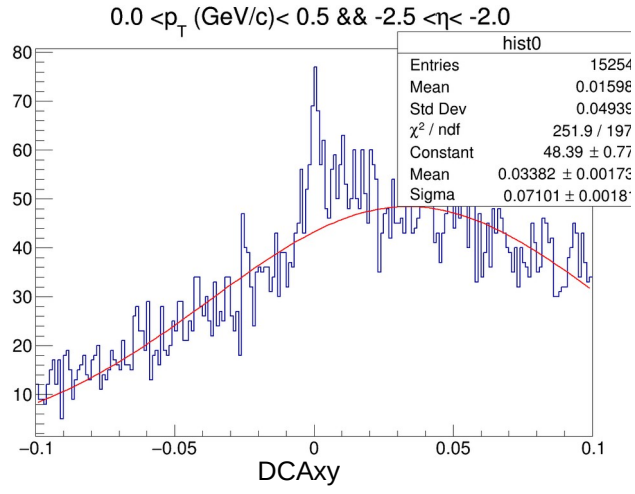


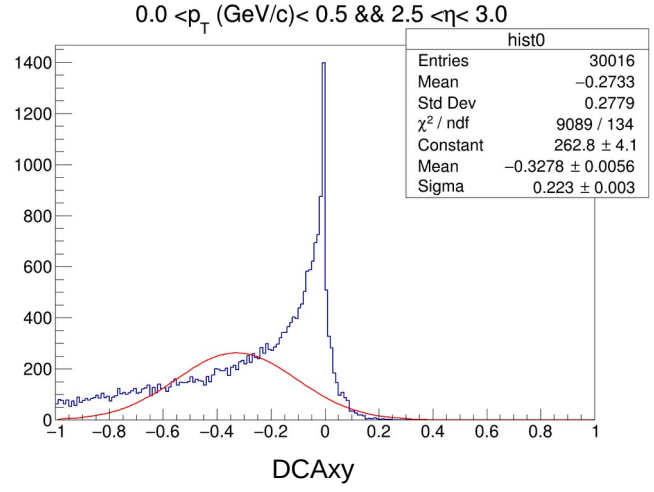


DCAxy Distributions

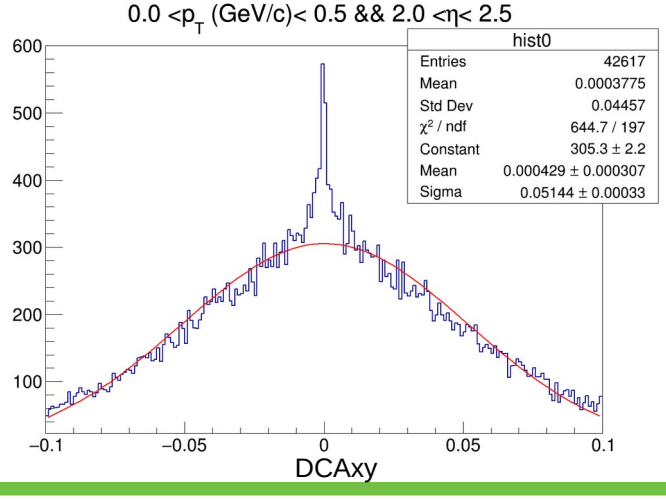
Smearing by 3 deg Theta/Phi (Disturbed DCAxy distribution not a suitable choice)

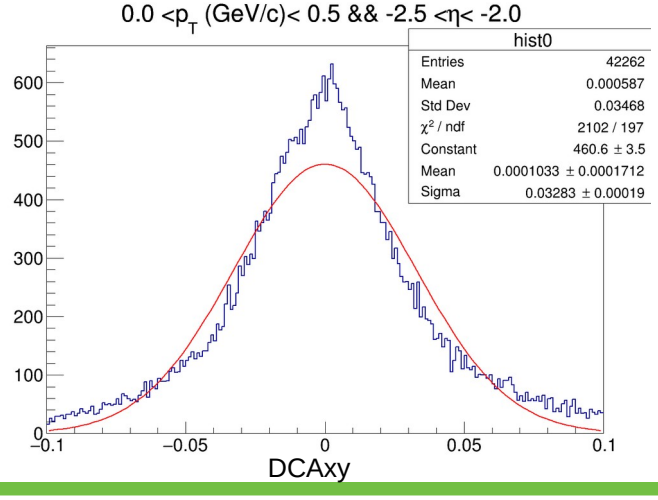


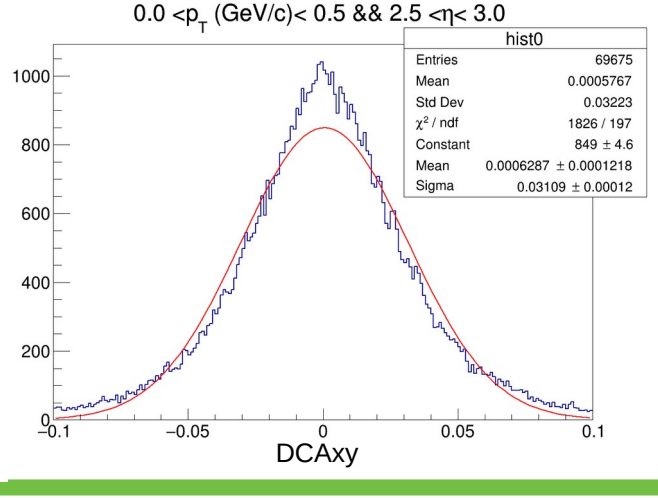




Only Momentum Smearing

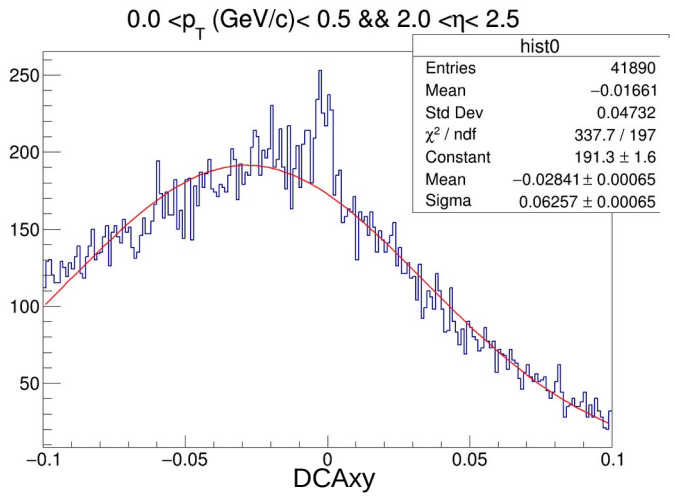


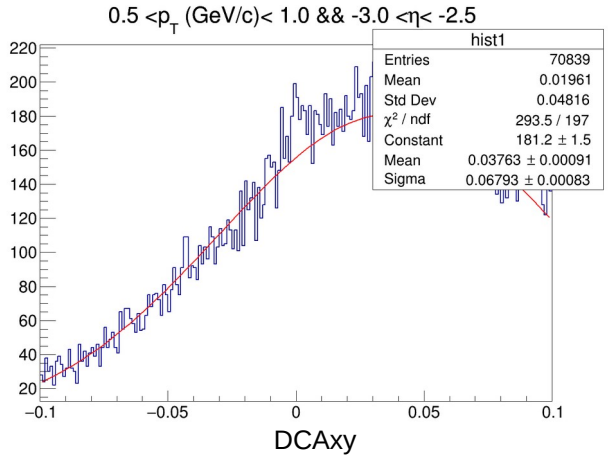


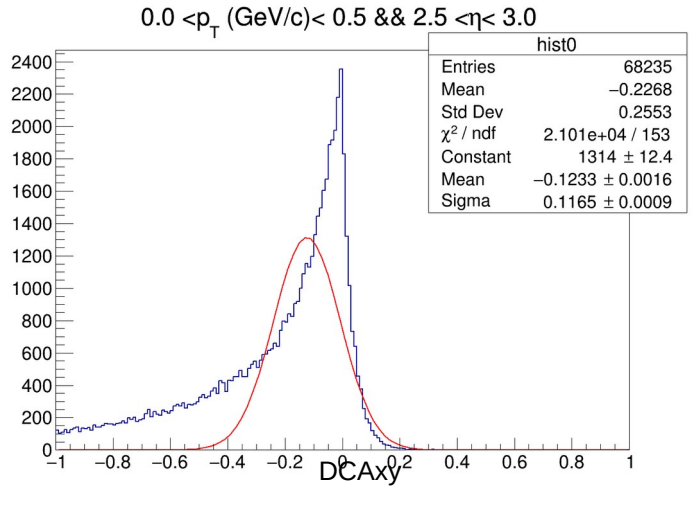


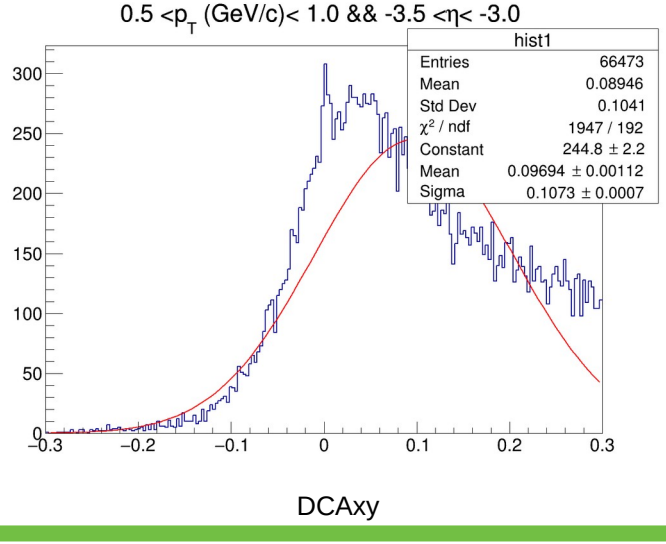
DCAxy Distributions

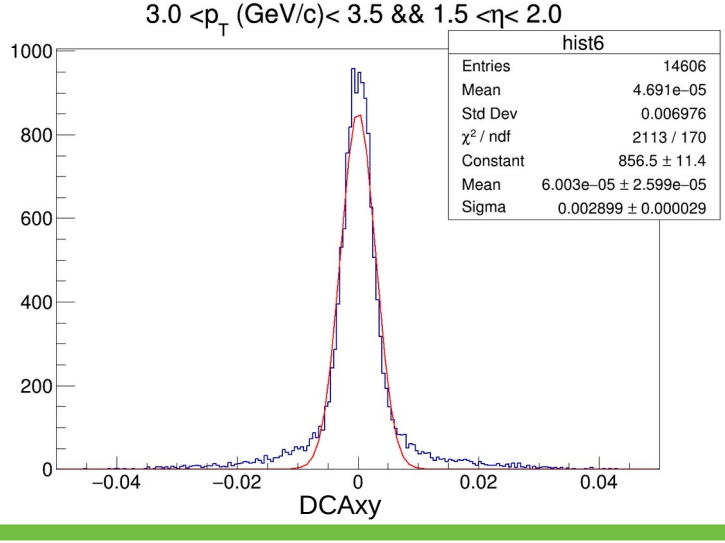
Smearing by 1 deg Theta/Phi (Disturbed DCAxy distribution not a suitable choice)

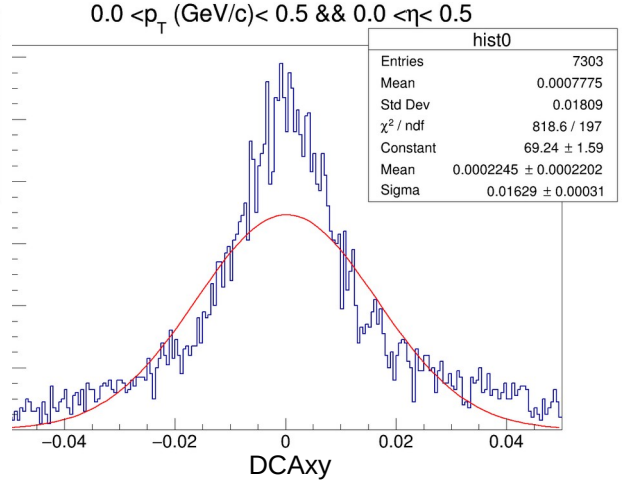




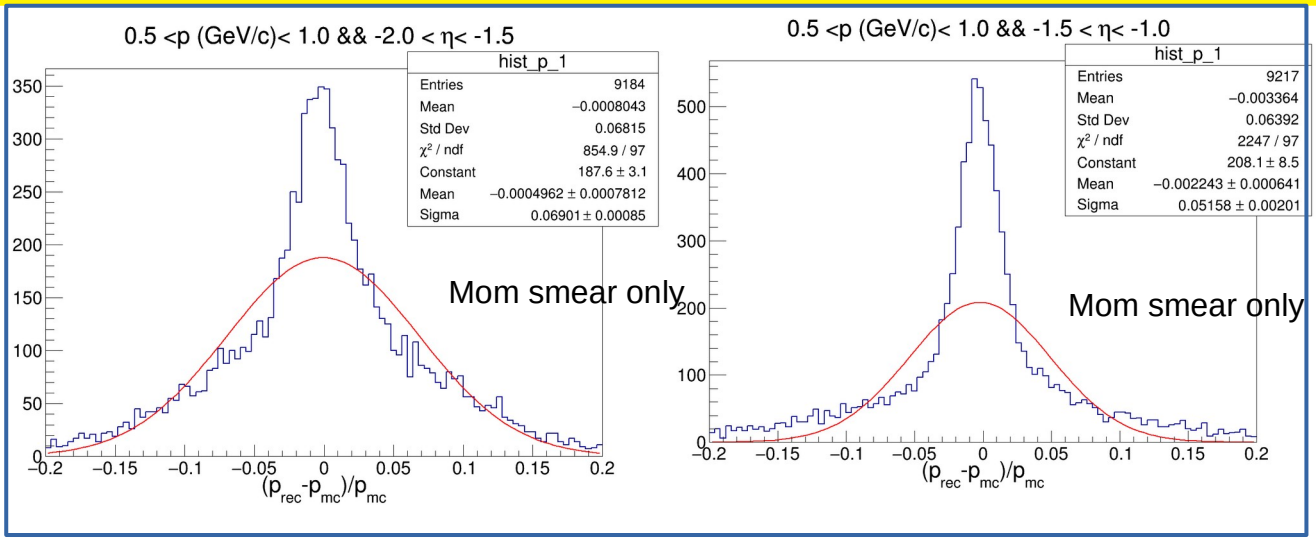


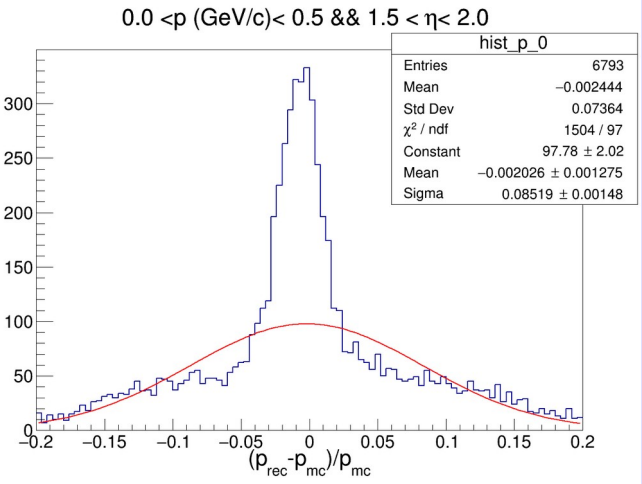


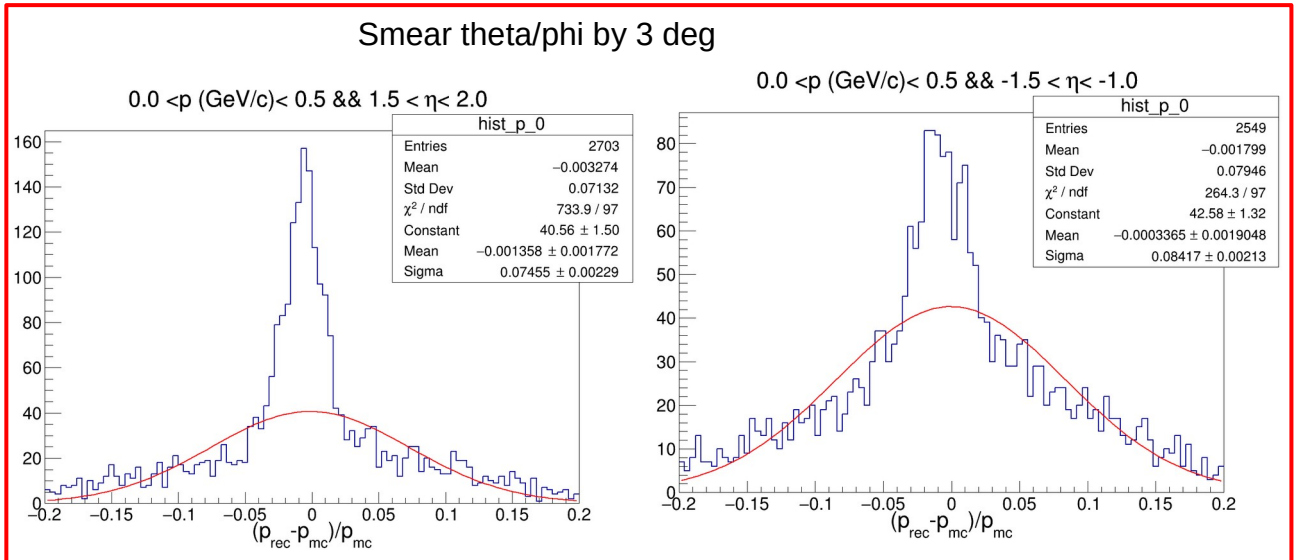


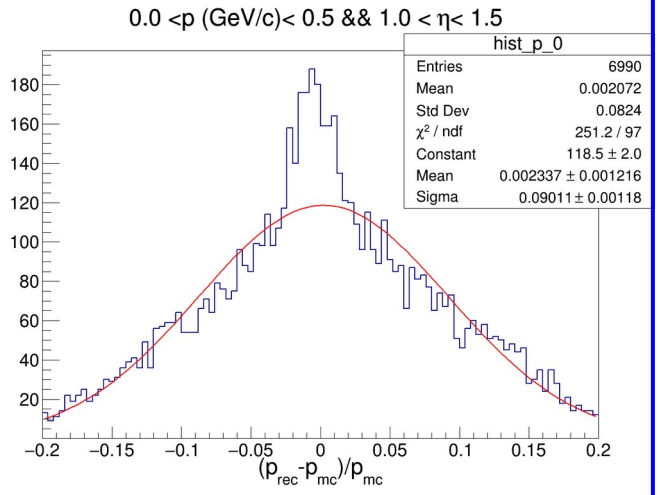


Momentum Resolution (Long tailed distribution)









Summary & Future Plan

- Smearing in Theta and Phi creates the problem in efficiency, also DCAxy distributions are affected
- I don’t see the decreasing trend in Theta/Phi resolution (Momenum smearing already there)
- There is an issue of long tail in the momentum resolution

$$\frac{\sigma_{pT}}{p_T} = \sqrt{\left(\frac{\sigma_{pT_{SR}}}{p_T}\right)^2 + \left(\frac{\sigma_{pT_{MS}}}{p_T}\right)^2}$$

$$\sigma_{d_0} = \sqrt{\sigma_{d0_{SR}}^{\ 2} + \sigma_{d0_{MS}}^{\ 2}}$$

$$\Delta d_0|_{res.} \approx \frac{3\sigma_{r\phi}}{\sqrt{N+5}}\sqrt{1 + \frac{8r_0}{L_0} + \frac{28r_0^2}{L_0^2} + \frac{40r_0^3}{L_0^3} + \frac{20r_0^4}{L_0^4}}$$

$$\sigma_{pT_{SR}} \propto \sigma_{r\phi} p$$

$$\sigma_{pT_{MS}} \propto \frac{1}{\beta p} p = \text{const}/\beta$$

$$\Delta d_0|_{m.s.} \approx \frac{0.0136\,\text{GeV/c}}{\beta p_T} r_0 \sqrt{\frac{d}{X_0 \sin\theta}} \sqrt{1 + \frac{1}{2}\left(\frac{r_0}{L_0}\right) + \frac{N}{4}\left(\frac{r_0}{L_0}\right)^2}$$

Curvature