Smearing of Parameters in the Truth Seeding

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

Old Slides

https://indico.bnl.gov/event/18084/contributions/72076/attachments/45516/76791/EPIC_ACTS_Meeting_Shyam12Jan23.pdf

epic_brycecanyon.xml with calorimeters and far forward detectors removed

```
shyam@shyam:~/eic/epic$ git tag -l
22.10.0
22.10_rc1
22.11.0
shyam@shyam:~/eic/epic$ git checkout 22.10.0
shyam@shyam:~/eic/epic$ git pull origin main
```

Truth Parameters in ACTS (Old)

```
// 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;

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 / (pmag * GeV);
params(Acts::eBoundTime)   = part->getTime() * ns;
```

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

↓

$(d_0, z_0, \theta, \phi, q/p)$

↓ Extrapolate to Vertex

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

Truth parameters are not smeared !!!

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

Truth Parameters in ACTS (New)

```
// modify initial momentum to avoid bleeding truth to results when fit fails
// this picks uniformly between [1-eps,1,1+eps] times true momentum, then smeared
const auto pinit = pmag
                 * (1.0 + m_cfg.m_momentumSplit * m_uniformIntDist(generator))
                 * (1.0 + m_cfg.m_momentumSmear * m_normDist(generator));
```

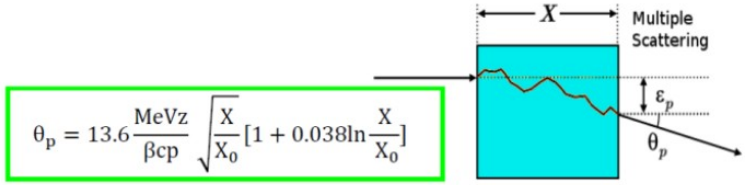
```
struct TrackParamTruthInitConfig {

    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_momentumSplit    = 0.0;
    double m_momentumSmear    = 0.0;
```

m_momentumSplit = 0.01, 0.1, 0.2, 0.90
m_momentumSmear = 0.01, 0.1, 0.2, 0.90

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

Track Fitting

Multiple Scattering (M.S.)



$$\theta_p = 13.6\frac{MeVz}{\beta cp}\sqrt{\frac{X}{X_0}}\left[1 + 0.038\ln\frac{X}{X_0}\right]$$

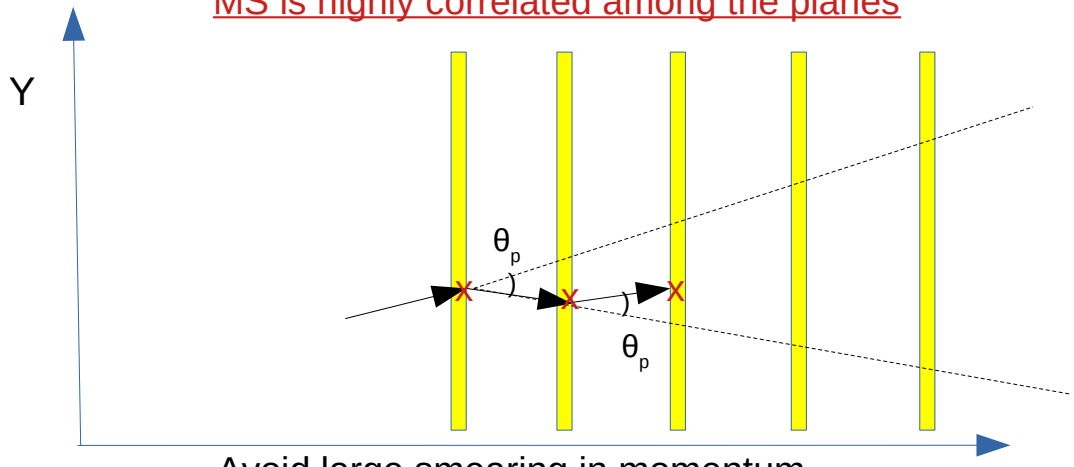
$$\epsilon_p \approx \frac{1}{\sqrt{3}}\theta_p X$$

Ref: "Particle Detectors", C. Grupen and B. A. Shwartz

At low p: m.s. dominated by $\propto \frac{1}{p}$

If we underestimate/overestimate assumption of p then we underestimate/overestimate θ_p and energyloss effect in the covariance at each plane (smearing of p will play role here)

MS is highly correlated among the planes



Avoid large smearing in momentum

Parameter Covariance

	l_0	l_1	ϕ	θ	$1/p$
l_0	σ^2_{l0}	σ_{l0l1}	$\sigma_{l0\phi}$	$\sigma_{l0\theta}$	$\sigma_{l0.1/p}$
l_1	σ_{l1l0}	σ^2_{l1}	$\sigma_{l1\phi}$	$\sigma_{l1\theta}$	$\sigma_{l1.1/p}$
ϕ	$\sigma_{\phi l0}$	$\sigma_{\phi l1}$	σ^2_{ϕ}	$\sigma_{\phi\theta}$	$\sigma_{\phi.1/p}$
θ	$\sigma_{\theta l0}$	$\sigma_{\theta l1}$	$\sigma_{\theta\phi}$	σ^2_{θ}	$\sigma_{\theta.1/p}$
$1/p$	$\sigma_{1/p.l0}$	$\sigma_{1/p.l1}$	$\sigma_{1/p.\phi}$	$\sigma_{1/p.\theta}$	$\sigma^2_{1/p}$

Symmteric matrix = 5(5+1)/2 = 15 independent entries

Multiple Scattering

	l_0	l_1	ϕ	θ	$1/p$
l_0	$\frac{\langle\theta_p^2\rangle dl^2}{3}$	0	$\frac{\langle\theta_p^2\rangle dl}{2\cos\lambda}$	0	0
l_1	0	$\frac{\langle\theta_p^2\rangle dl^2}{3}$	0	$\frac{-\langle\theta_p^2\rangle dl}{2}$	0
ϕ	$\frac{\langle\theta_p^2\rangle dl}{2\cos\lambda}$	0	$\frac{\langle\theta_p^2\rangle}{\cos\lambda}$	0	0
θ	0	$\frac{-\langle\theta_p^2\rangle dl}{2}$	0	$\langle\theta_p^2\rangle$	0
$1/p$	0	0	0	0	0

https://agenda.infn.it/event/1096/contributions/6159/attachments/4504/4980/Rotondi_3.pdf

Based on my experience in ALICE

$$\lambda = \pi/2 - \theta$$

Spatial Resolution and Multiple Scattering

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

From Fast simulation

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

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

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

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

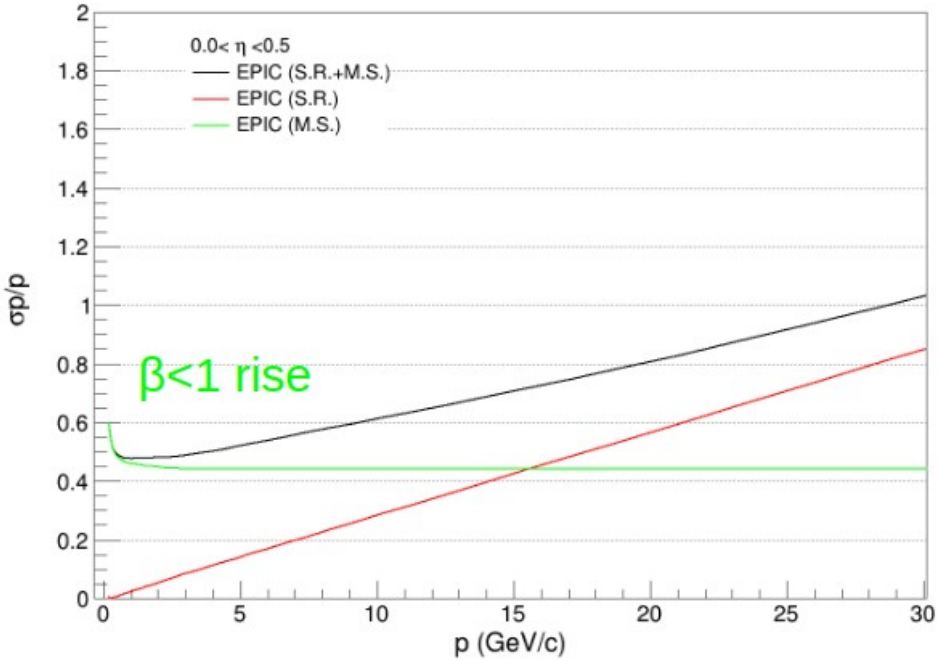
Curvature

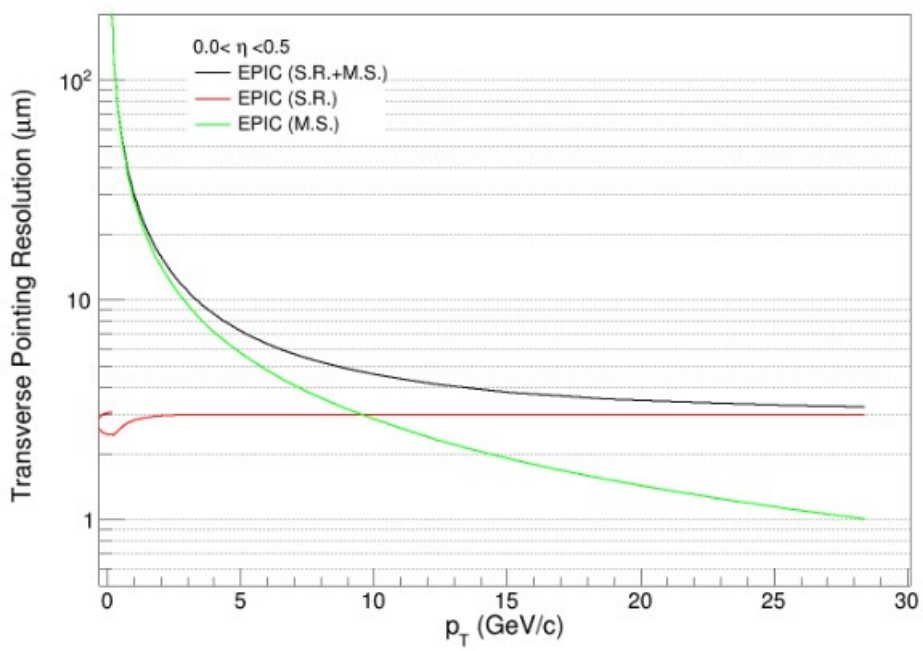
$\sigma_{pT_{MS}} \propto \frac{1}{\beta p} p = \text{const}/\beta$

Curvature

Momentum hypothesis → Realistic seeding → Curvature

Momentum hypothesis → Realistic seeding → Curvature





https://indico.bnl.gov/event/17750/contributions/71187/attachments/44843/75637/EPIC_Tracking_Meeting_Shyam1Dec2022.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

My Idea on Oct 2022

https://github.com/eic/EICrecon/issues/215

```
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);          Initialization default

  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)                     If there is a truth information
  {
    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)                Option for 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));
    }
  }
```

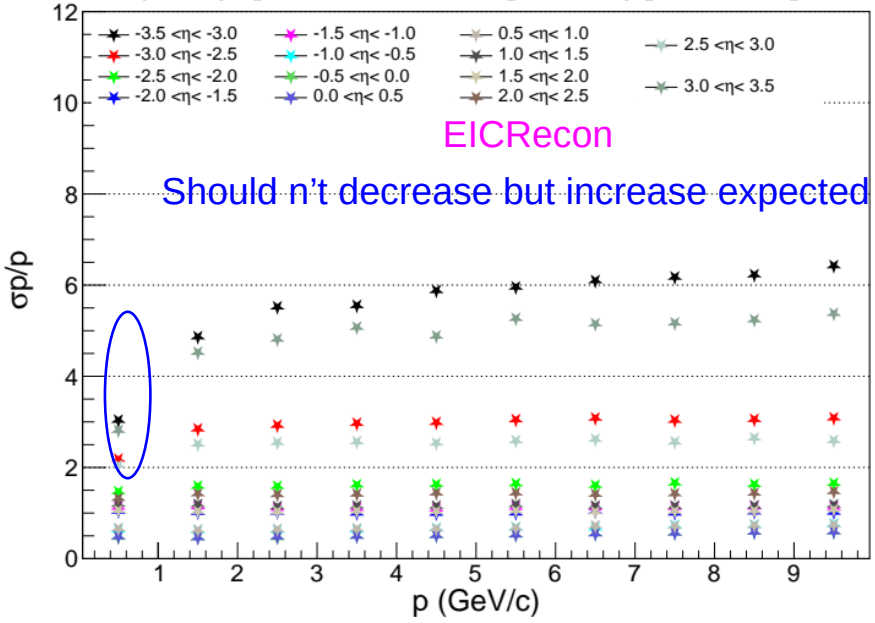
10% smearing in p_{true}

Fun4All also avoid large smearing in momentum but smears theta and phi

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

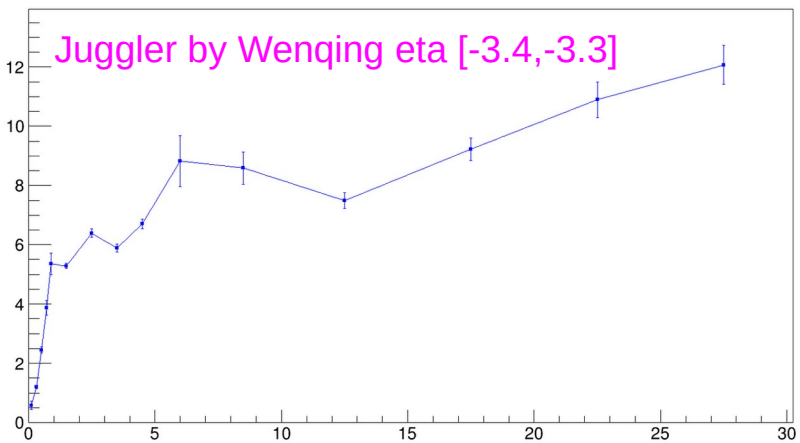
Results

3M pi+ p [0.1,10. GeV/c] and η [-3.5,3.5]

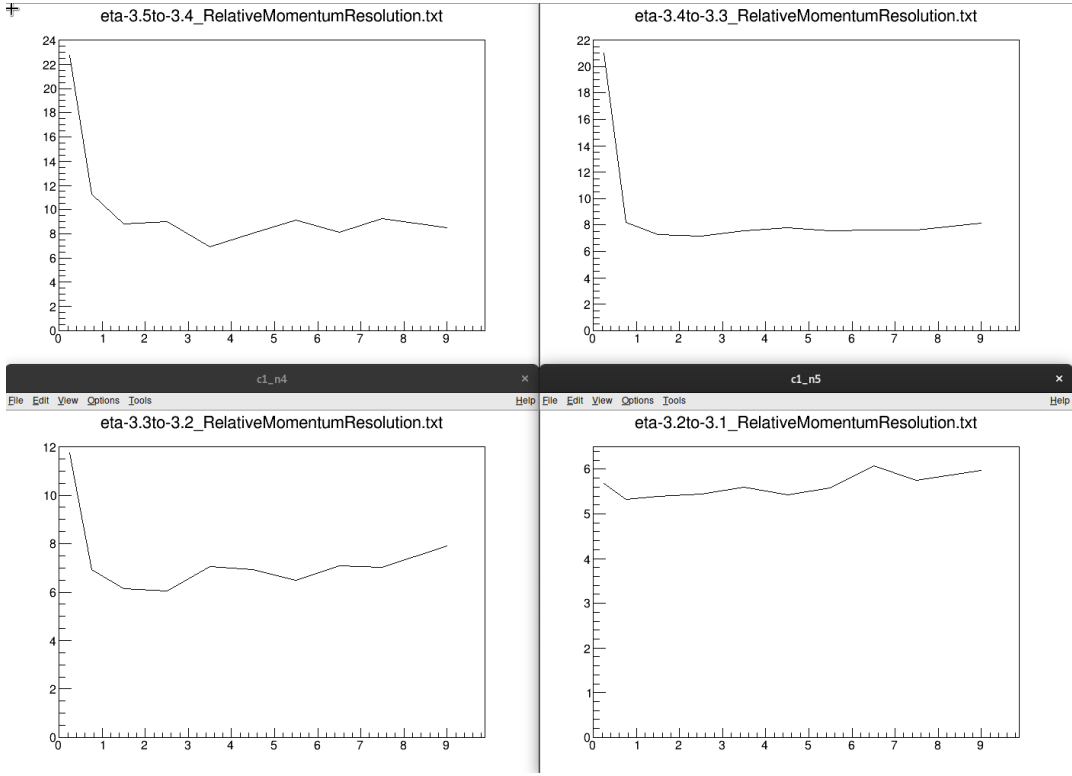


EICRecon

Should n't decrease but increase expected

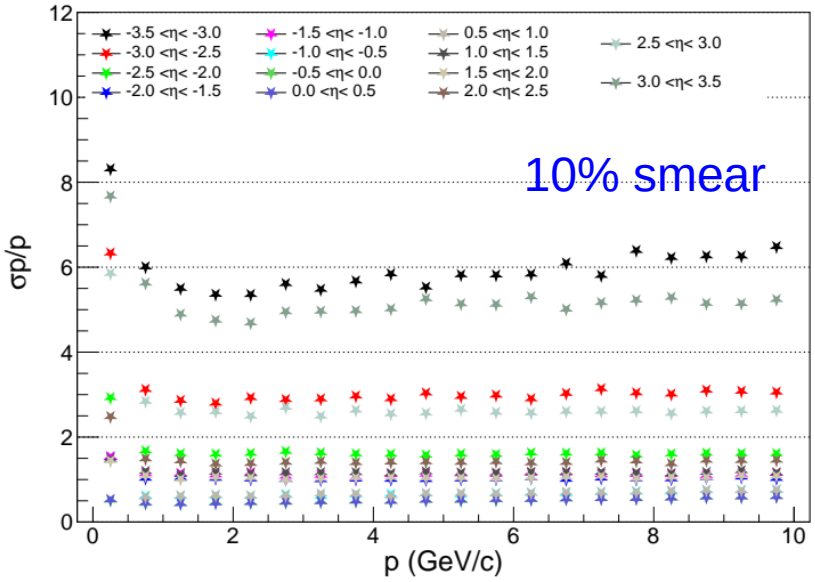


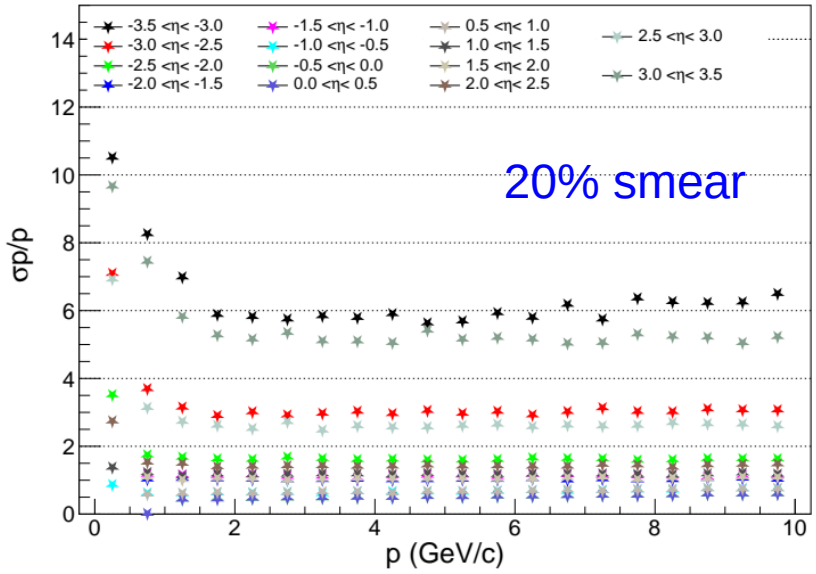
From Stephen using Fun4All (geometry is same)



Fun4All is giving correct trend at low momentum !!

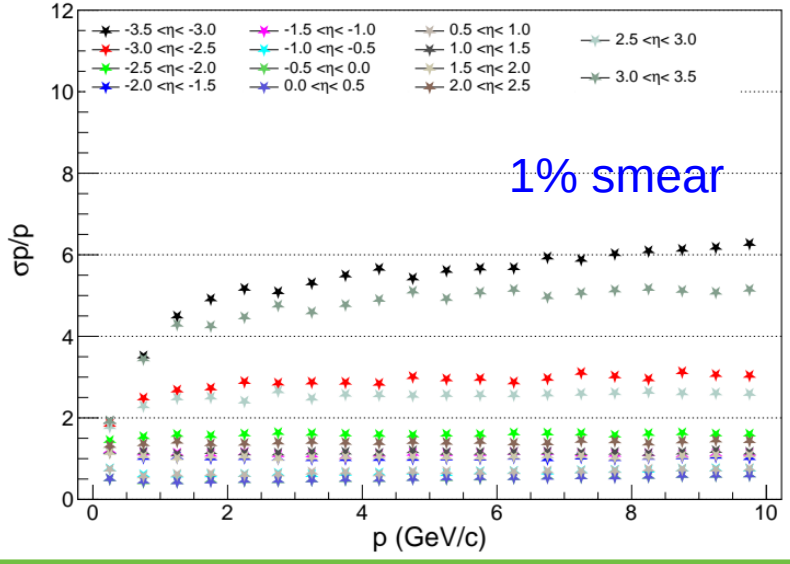
Results (Fine η Bins)





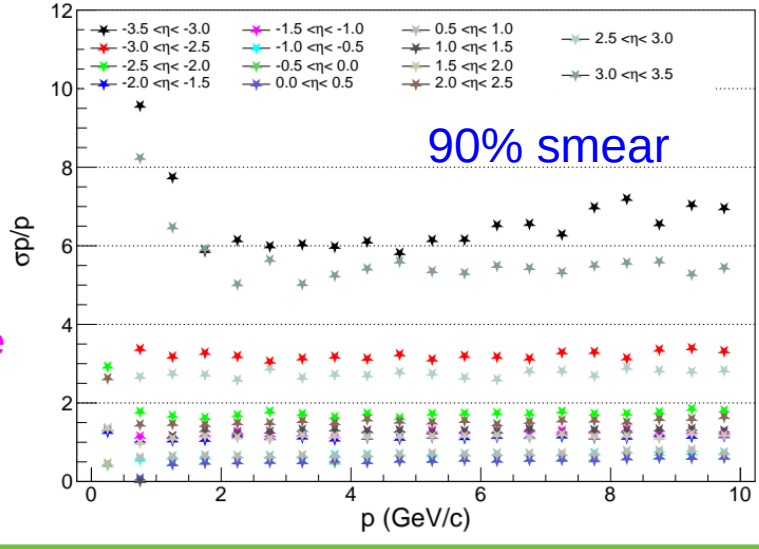
m_momentumSplit
= 0.01, 0.1, 0.2, 0.90

m_momentumSmear
= 0.01, 0.1, 0.2, 0.90

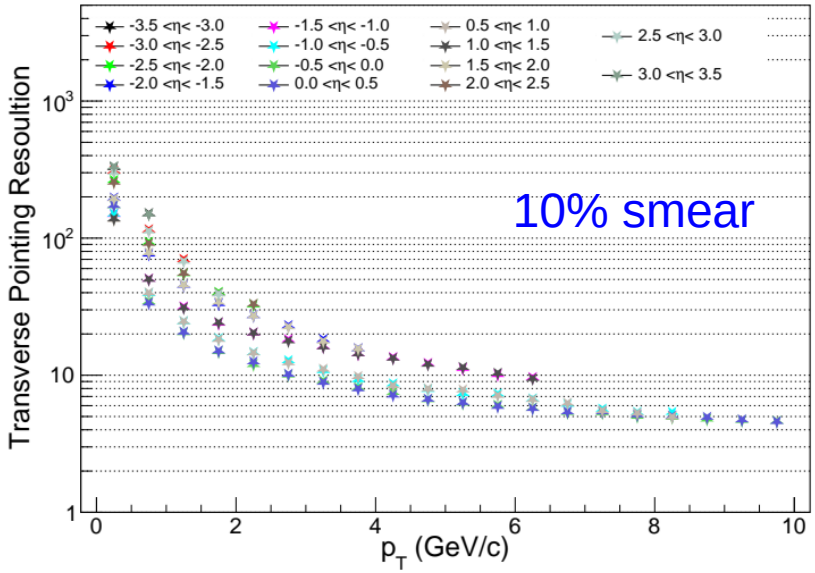


If you smear very small fitter is biased

90 % is really bias the result as can be seen in the plot

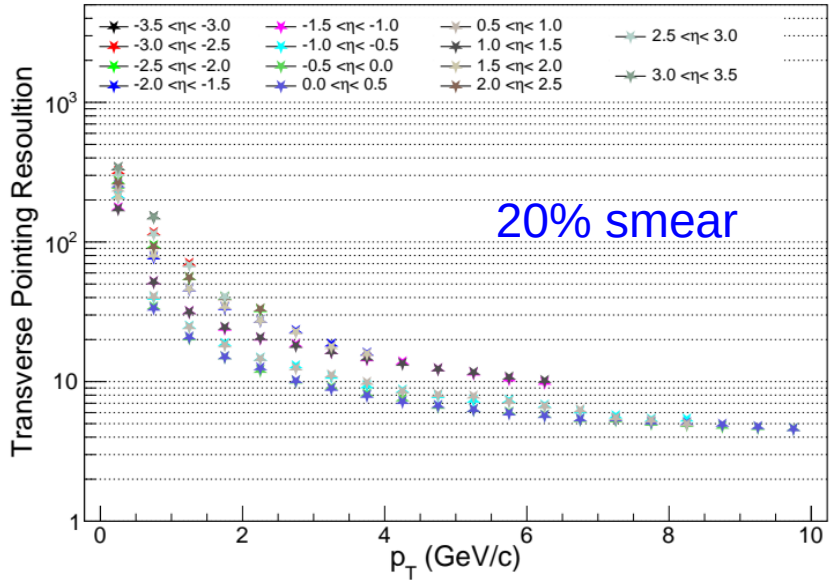


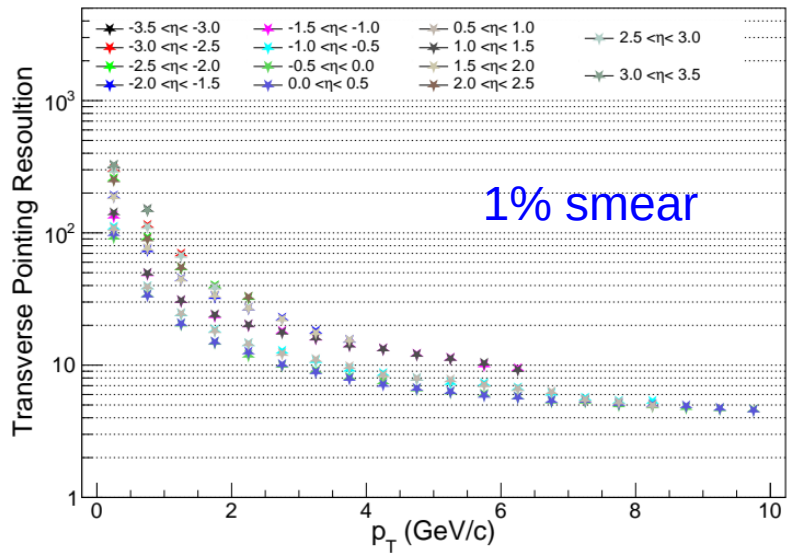
Results (Fine η Bins)



m_momentumSplit
= 0.01, 0.1, 0.2, 0.90

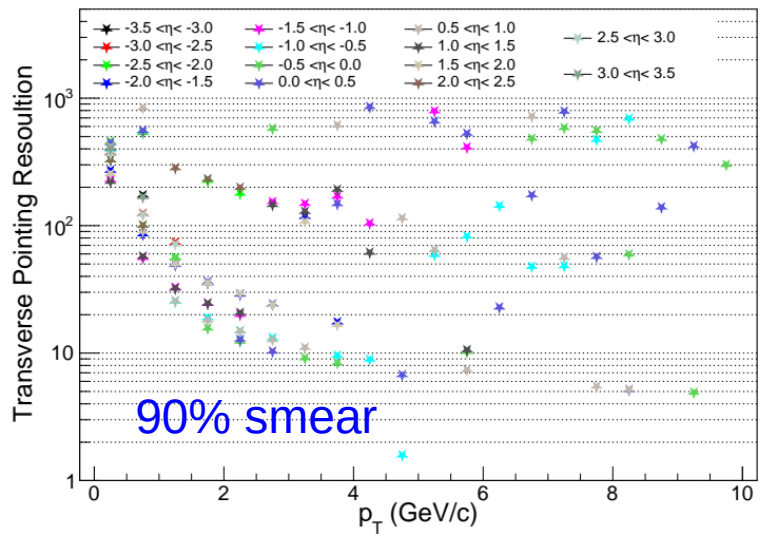
m_momentumSmear
= 0.01, 0.1, 0.2, 0.90



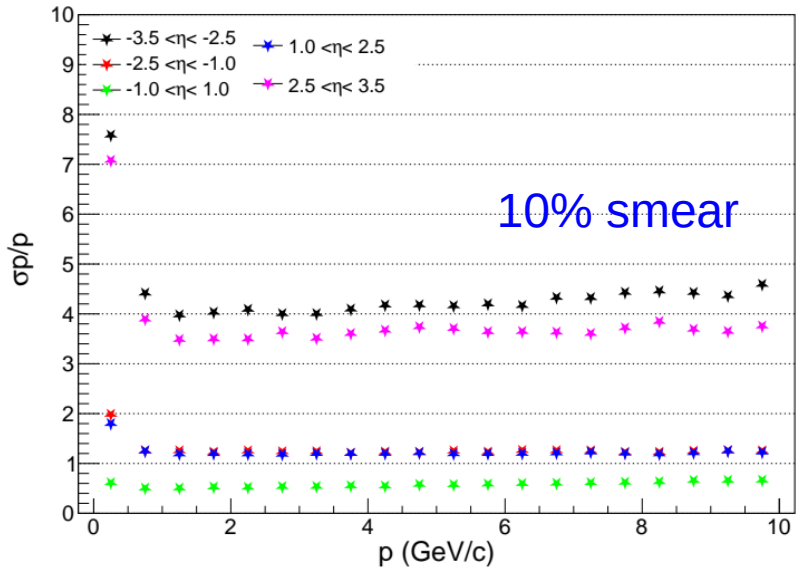


If you smear very small fitter is biased

90 % is really bias the result as can be seen in the plot

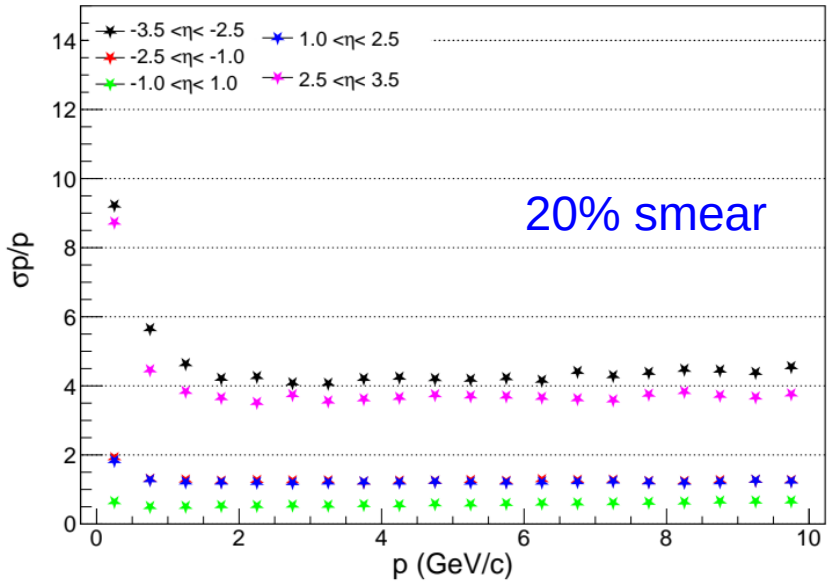


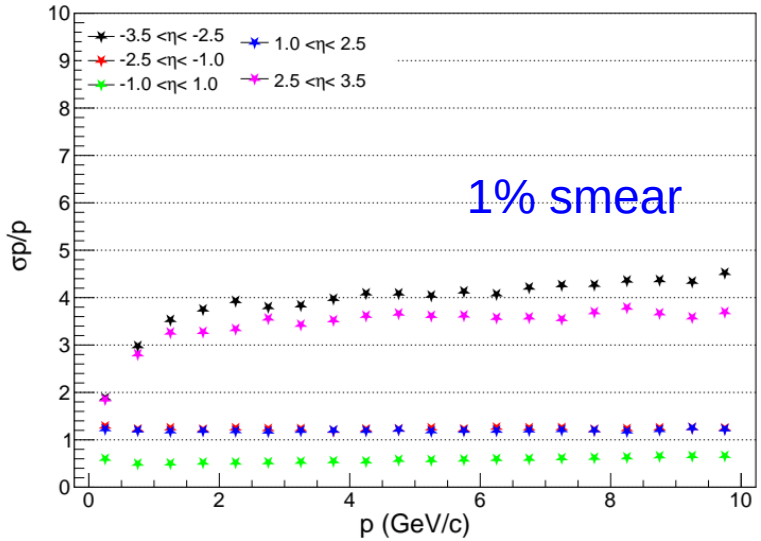
Results (Wide η Bins)

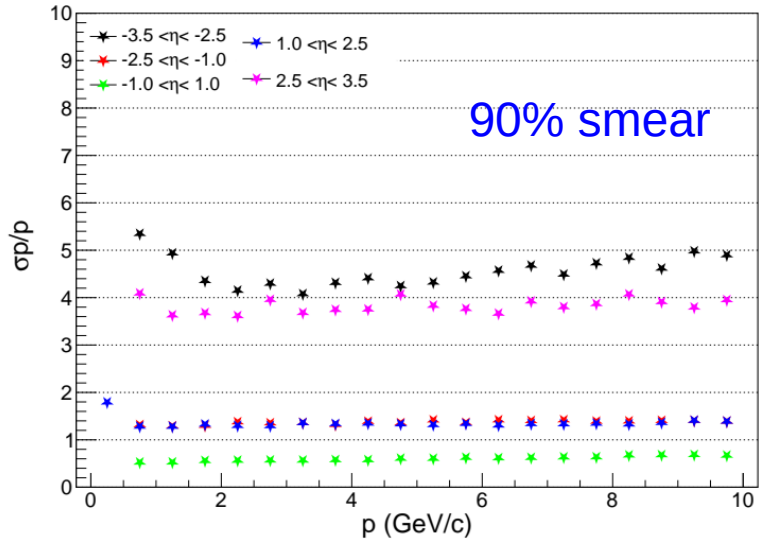


m_momentumSplit
= 0.01, 0.1, 0.2, 0.90

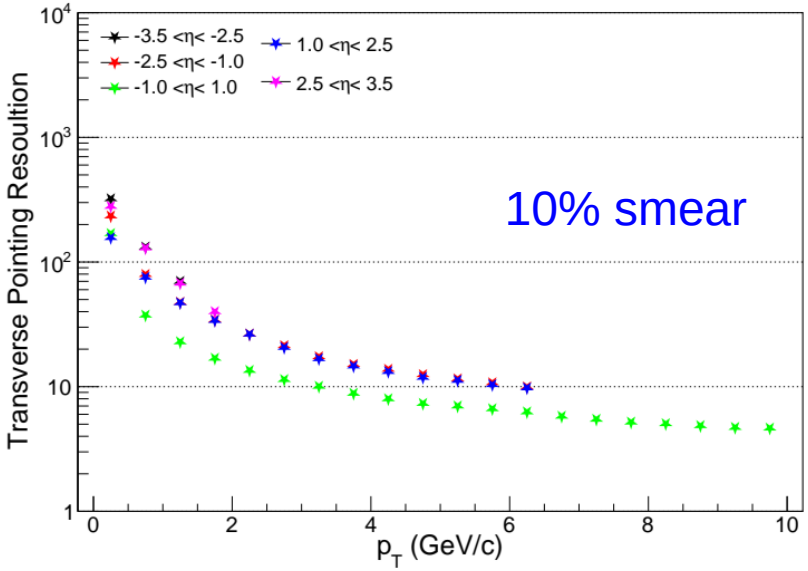
m_momentumSmear
= 0.01, 0.1, 0.2, 0.90





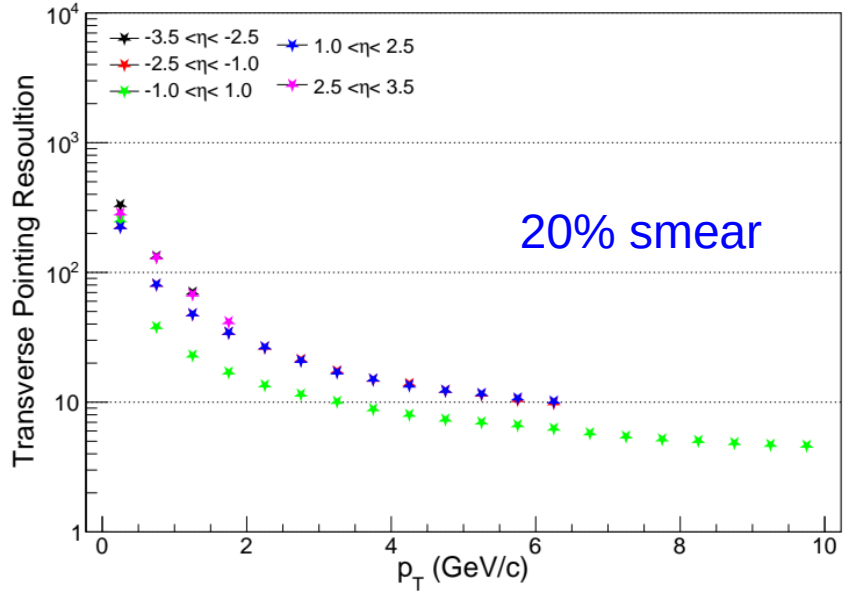


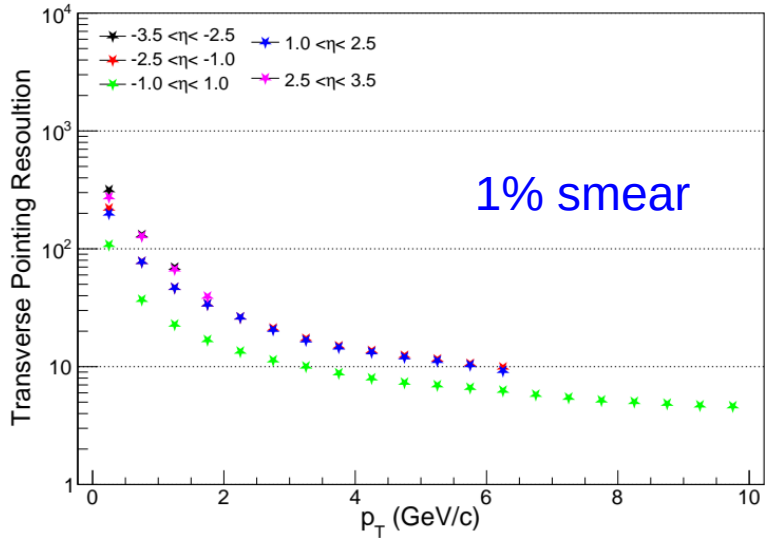
Results (Wide η Bins)

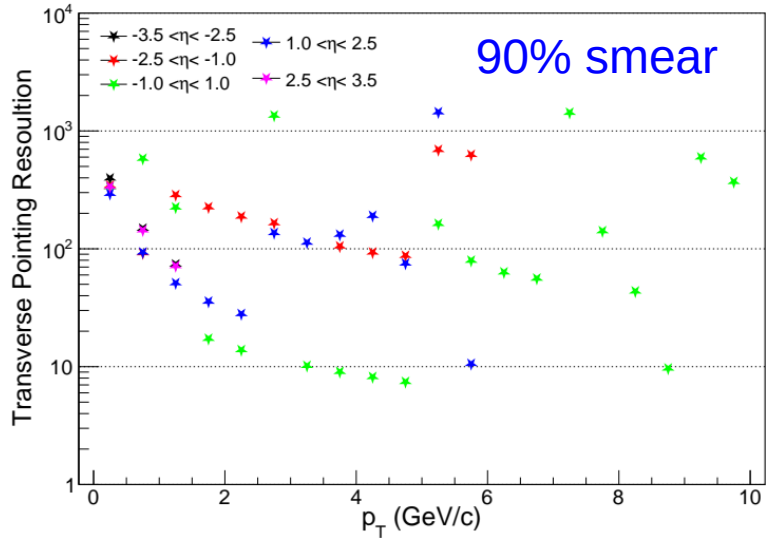


m_momentumSplit
= 0.01, 0.1, 0.2, 0.90

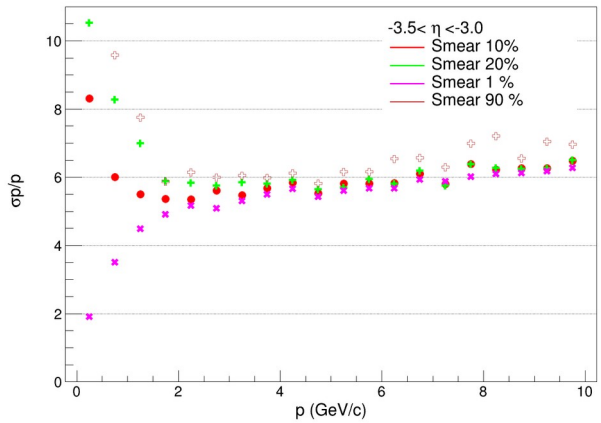
m_momentumSmear
= 0.01, 0.1, 0.2, 0.90

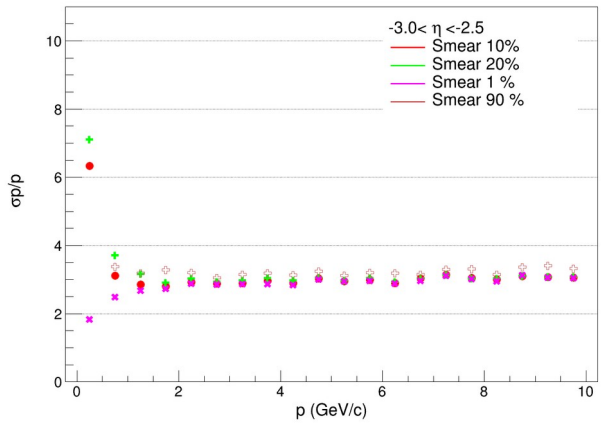


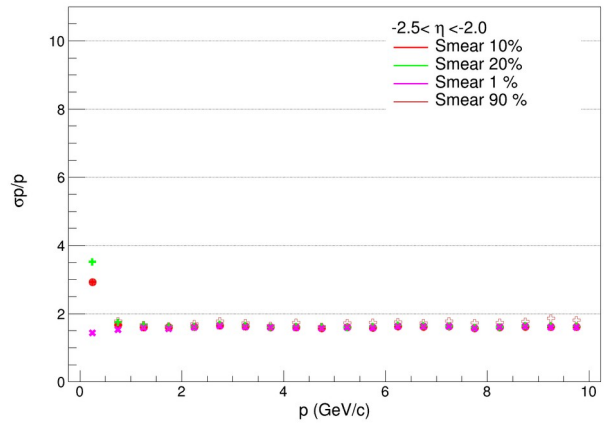


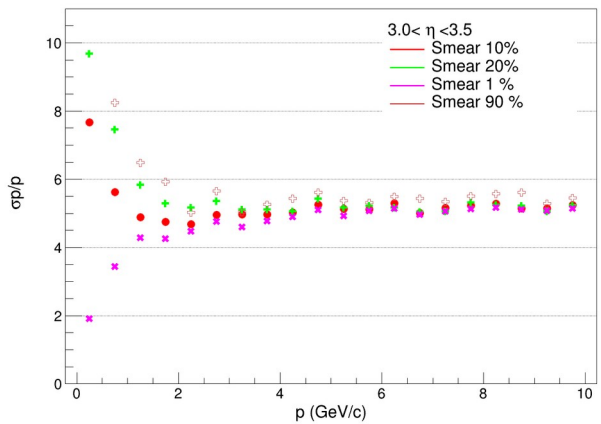


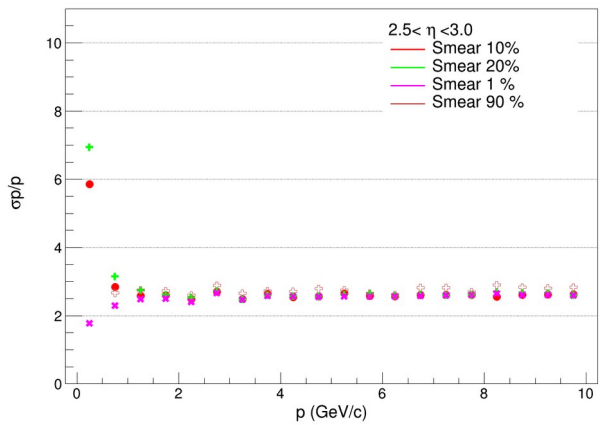
Comparison Momentum Resolution [Fine eta bins]

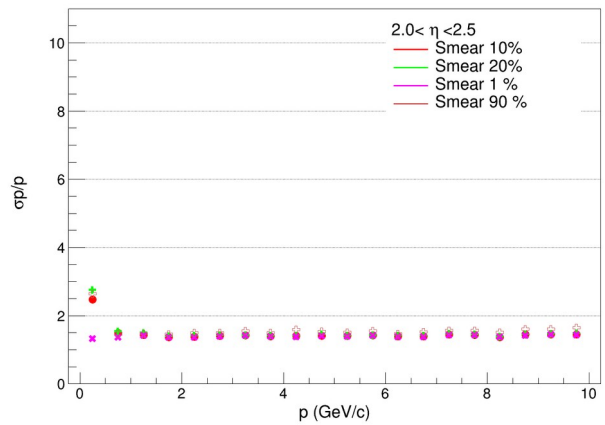




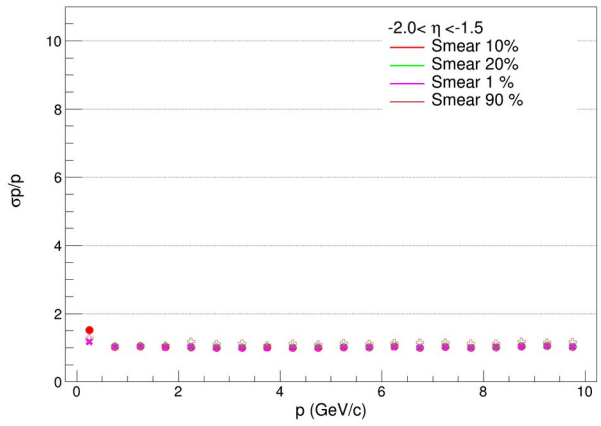


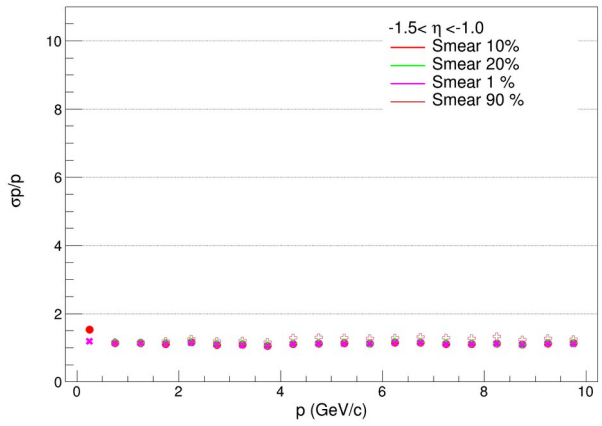


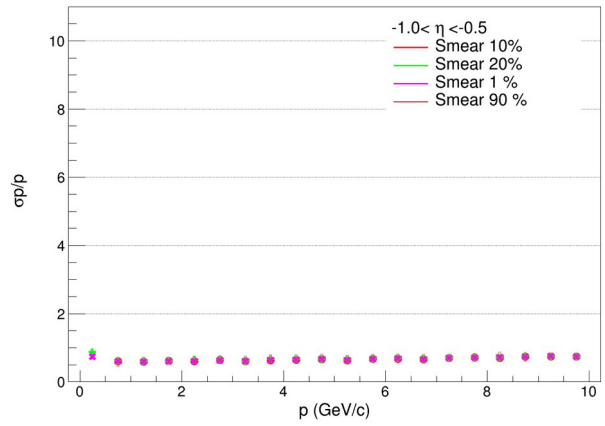


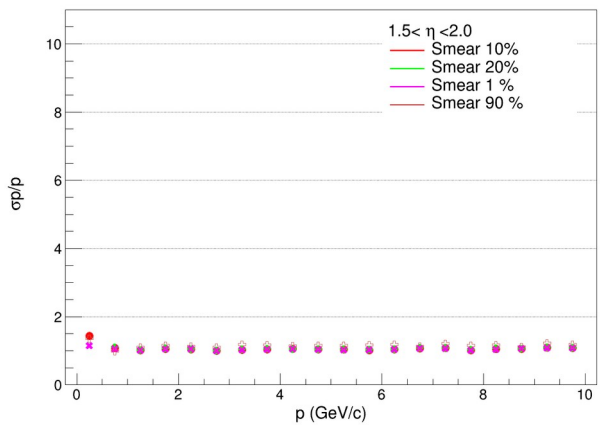


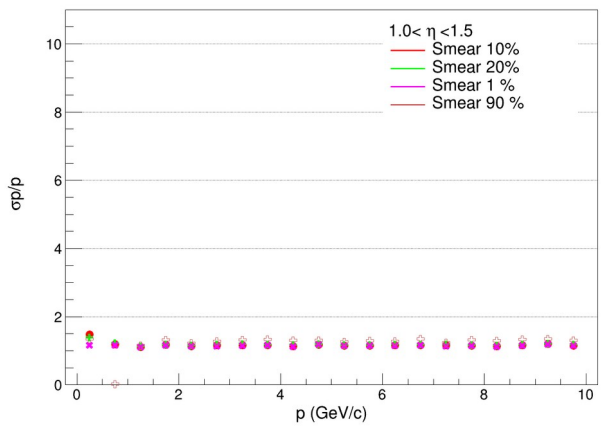
Comparison Momentum Resolution [Fine eta bins]

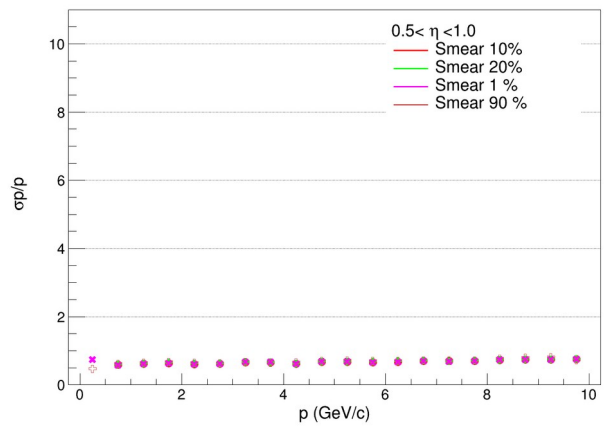




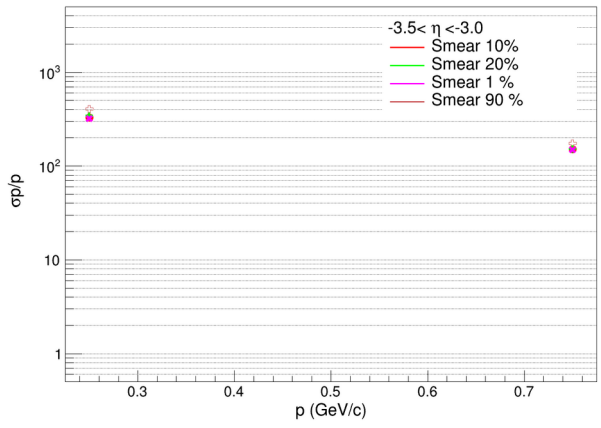


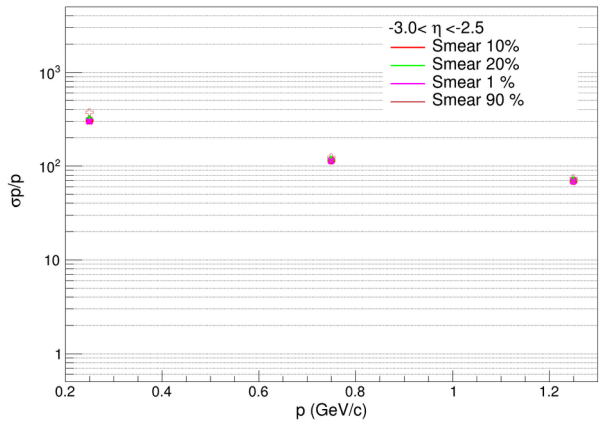


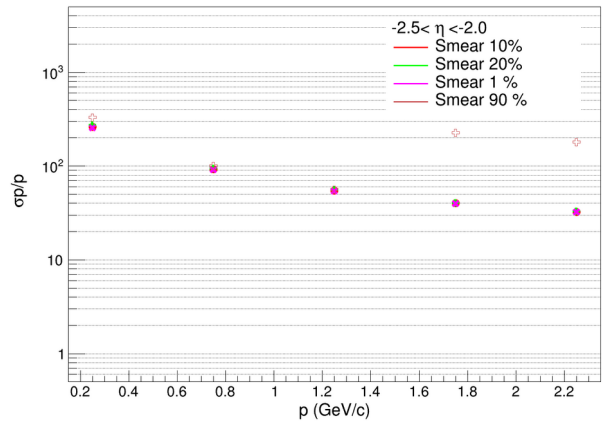


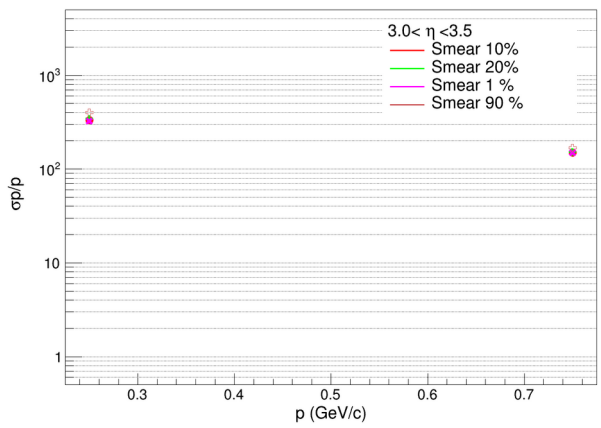


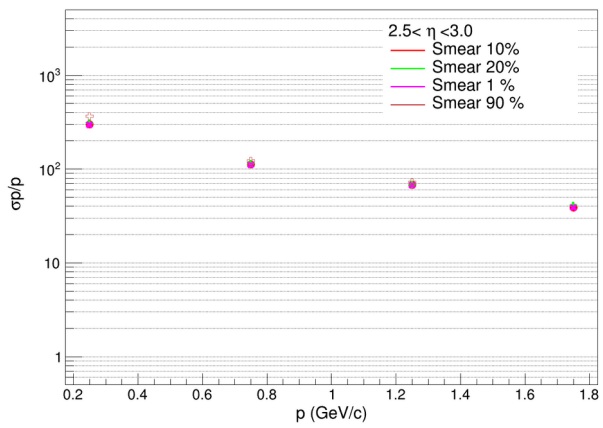
Comparison DCA_{xy} Resolution [Fine eta bins]

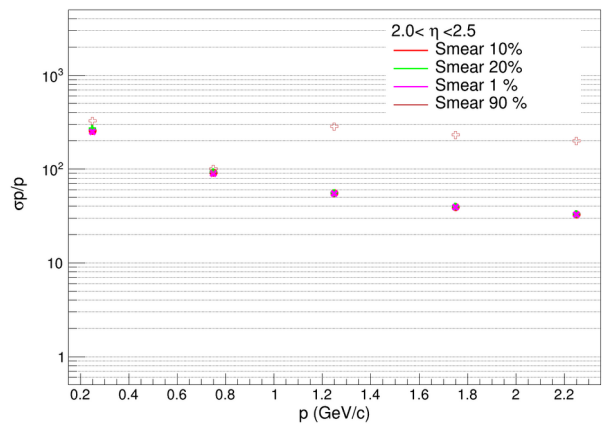




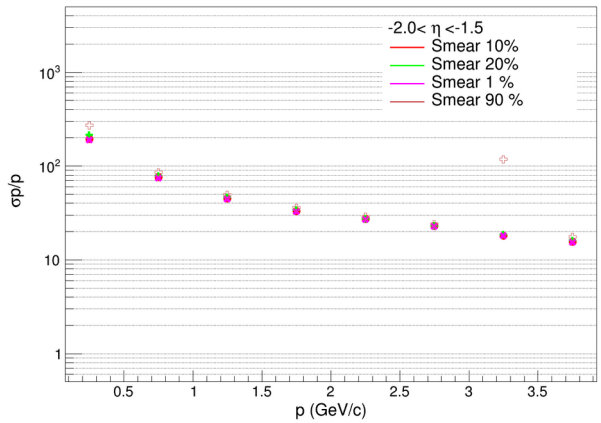


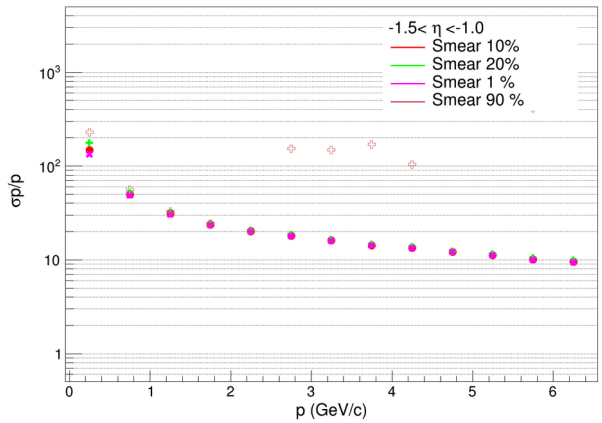


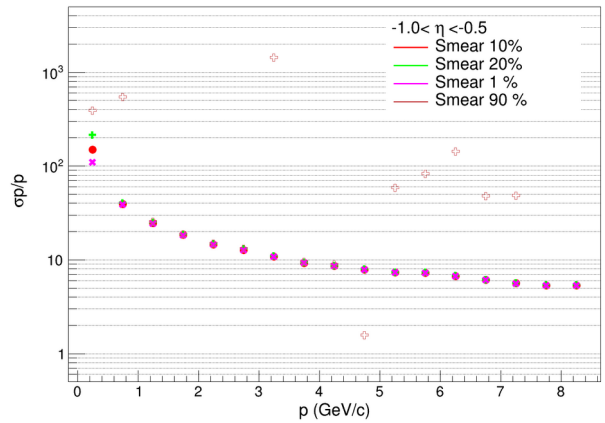


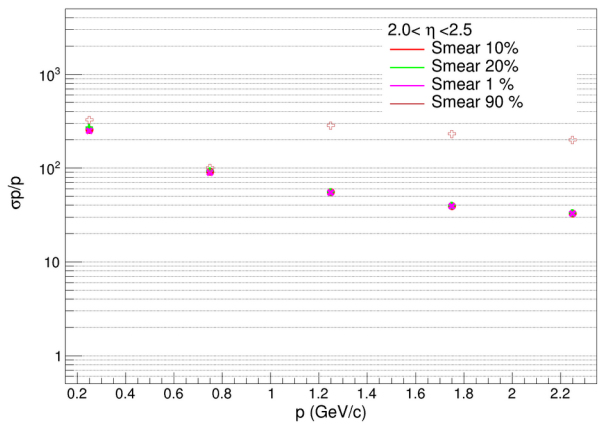


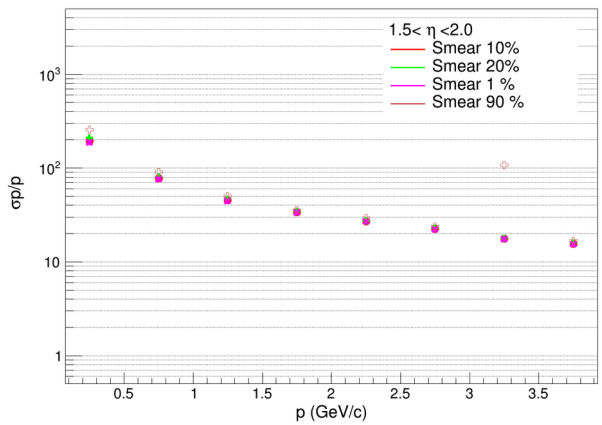
Comparison DCA_{xy} Resolution [Fine eta bins]

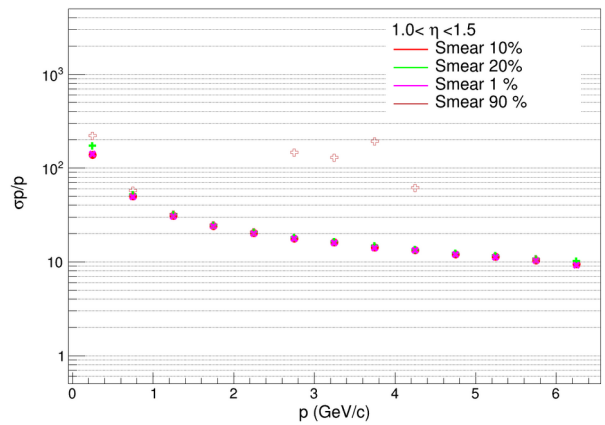




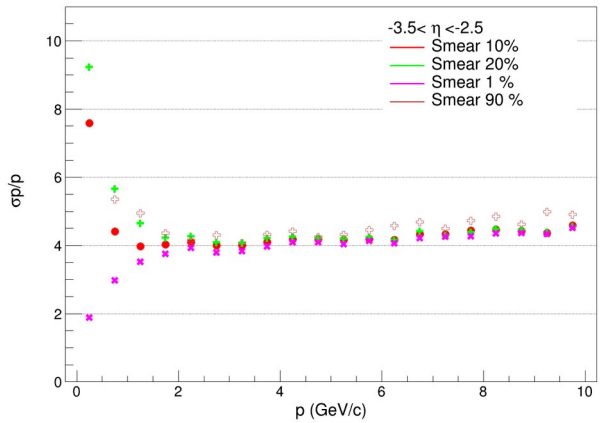


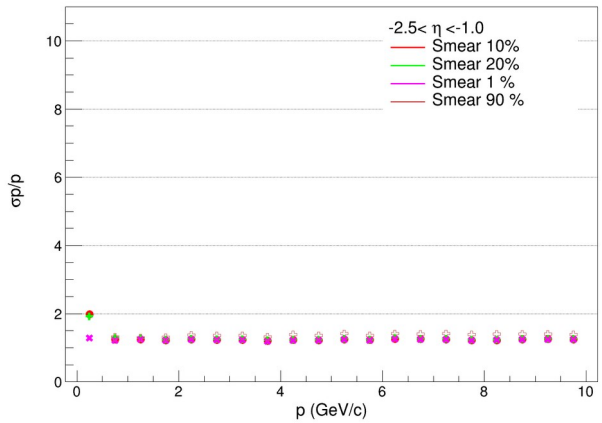


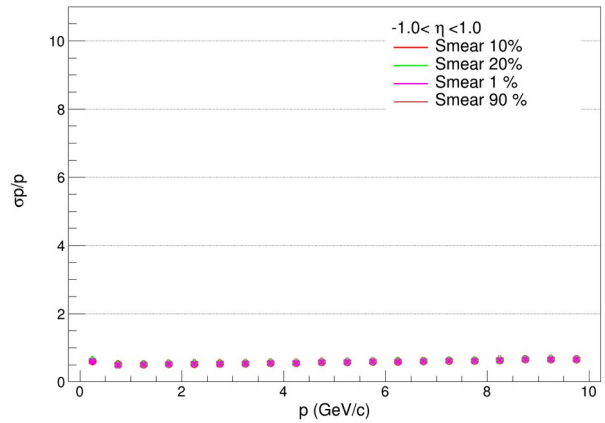


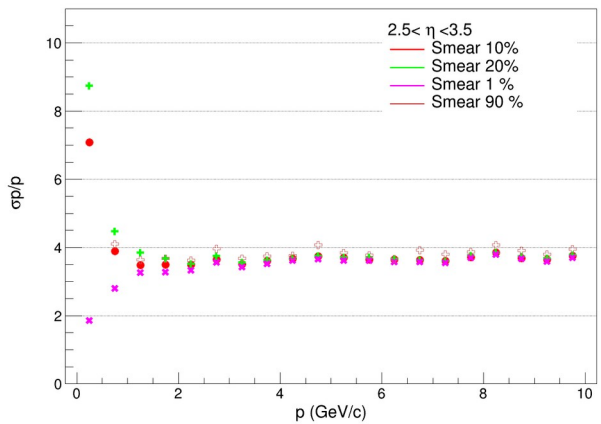


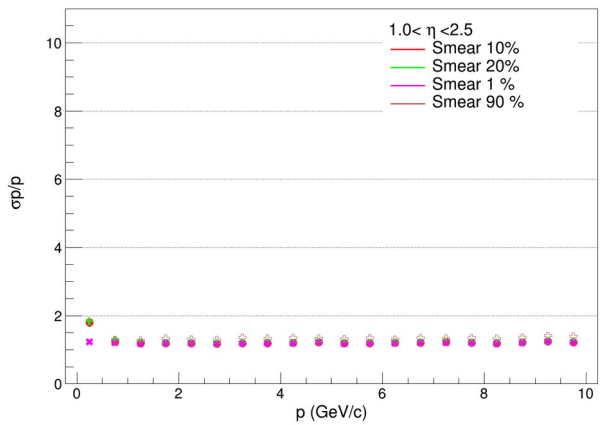
Comparison Momentum Resolution [Wide eta bins]



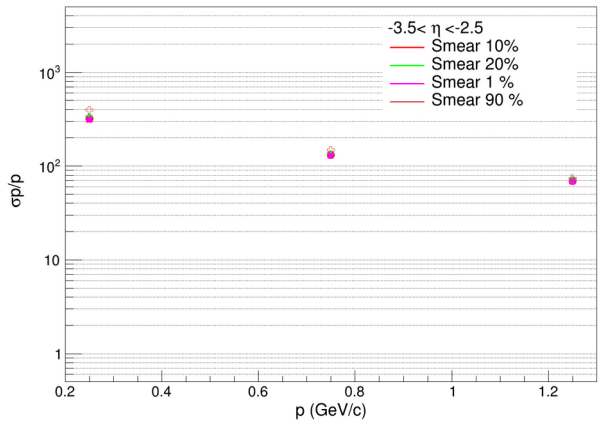


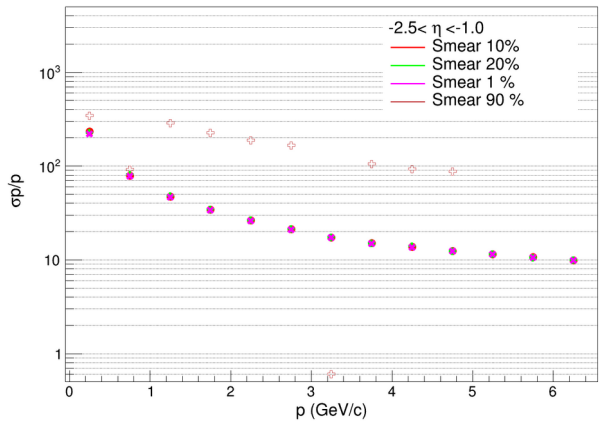


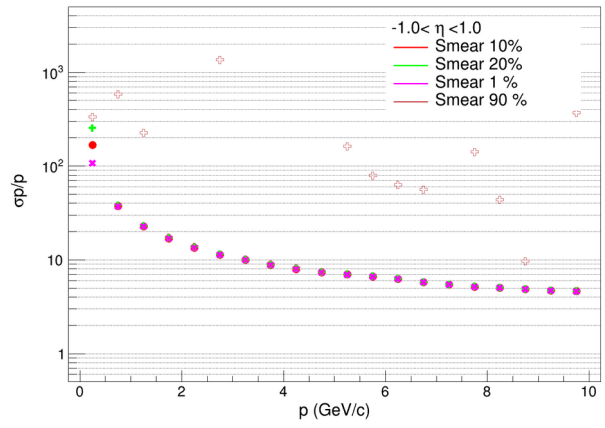


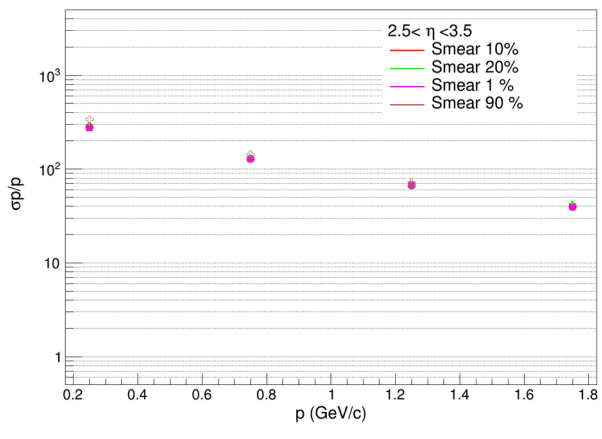


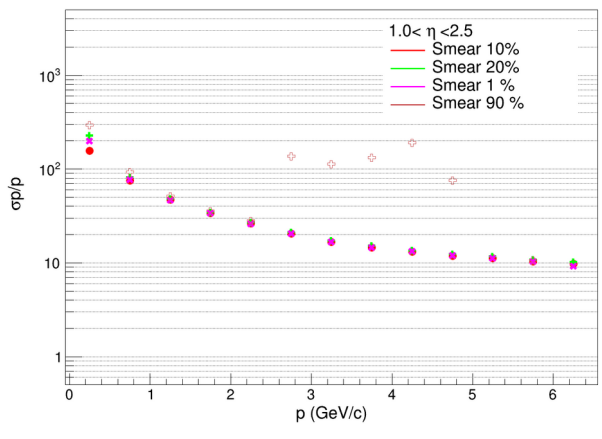
Comparison DCA_{xy} Resolution [Wide eta bins]









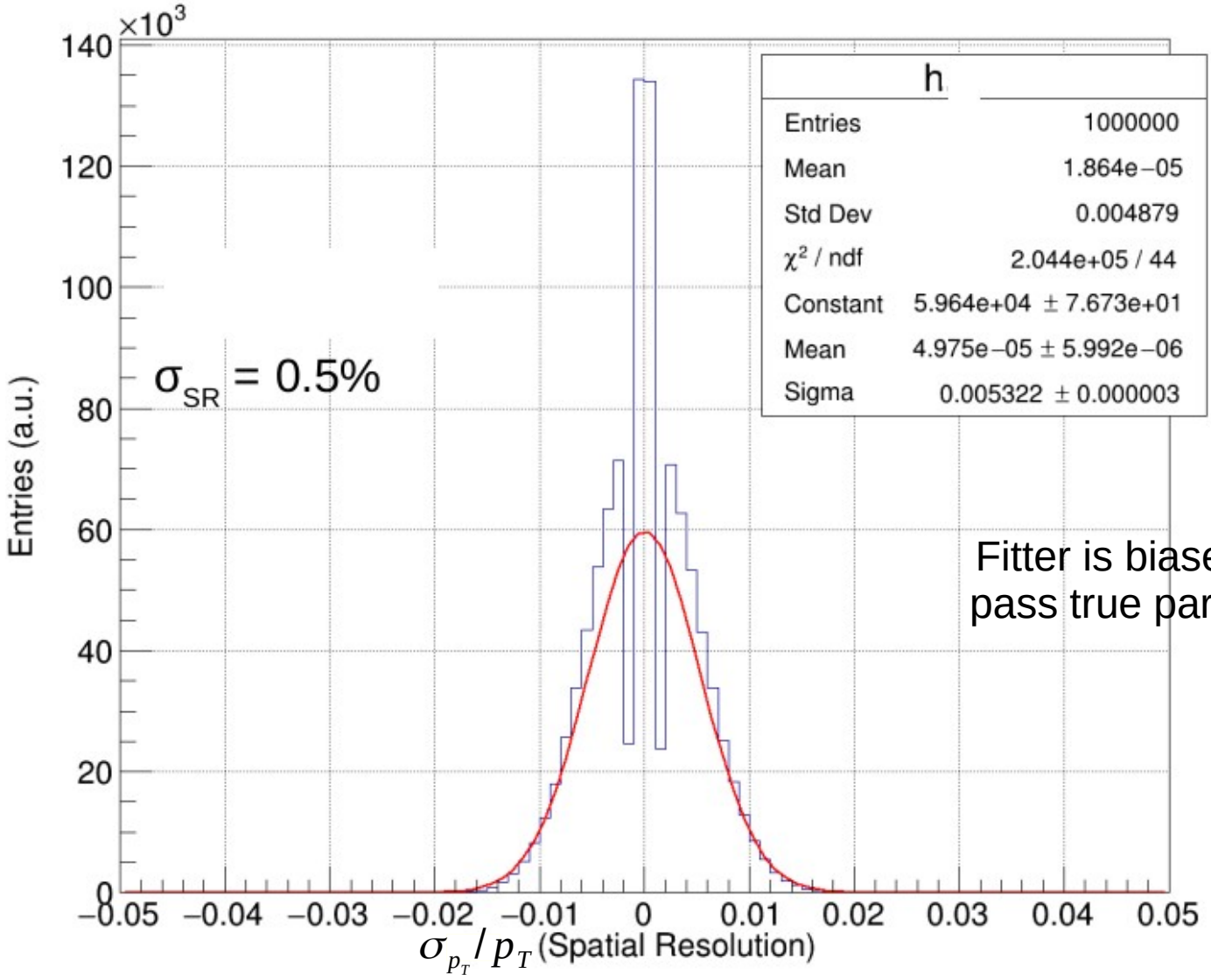


Summary and Future Plan

- Origin of decrease of Momentum resolution at low momentum and higher eta is understood
- Large smearing in momentum can bias results due to underestimation/overestimation of energy loss and multiple scattering
- May be we can try to adopt fun4All solution smear momentum small and also smear other parameters theta, phi which doesn't effect multiple scattering and energy loss
- We need to come with proper smearing solution (using Chi2 information once there)
- Need to cross check with smearing and realistic seeding the momentum resolution should increase at low momentum

Thank You

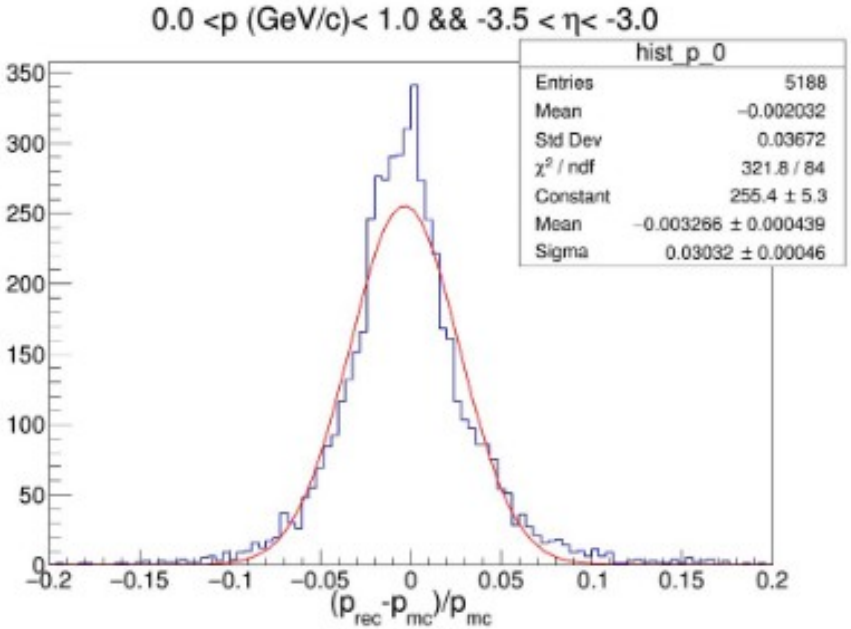
Fitter with true parameters (My Fast Simulation)



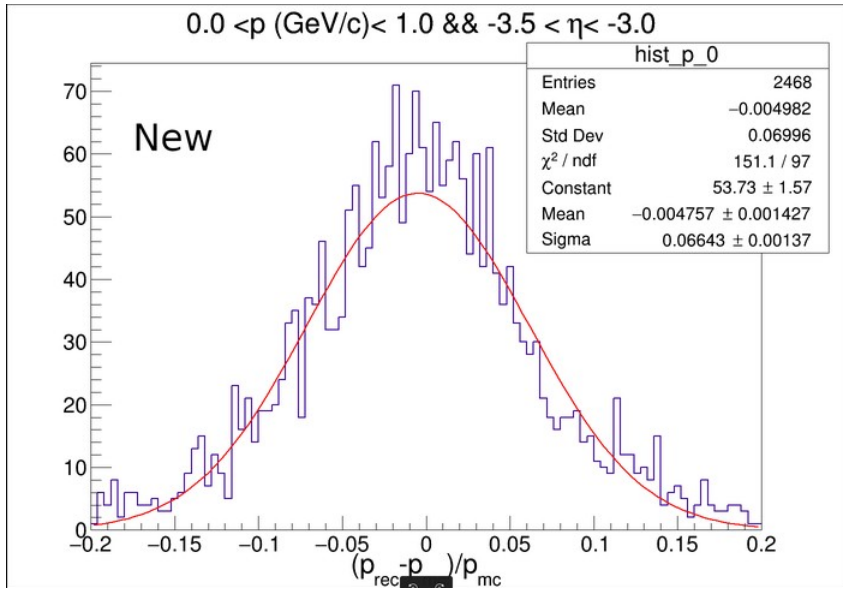
Fitter is biased if you pass true parameters

Momentum Resolution

Without smearing true parameters

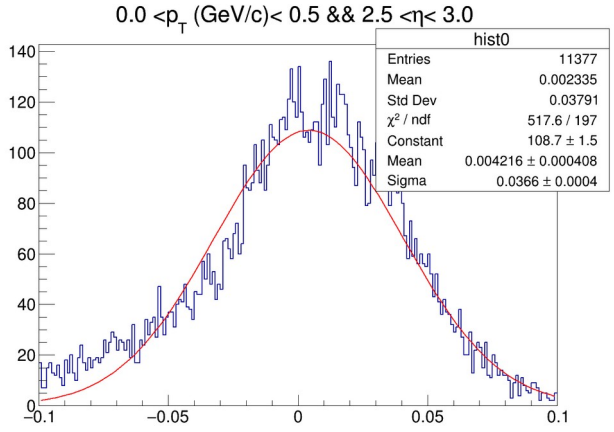


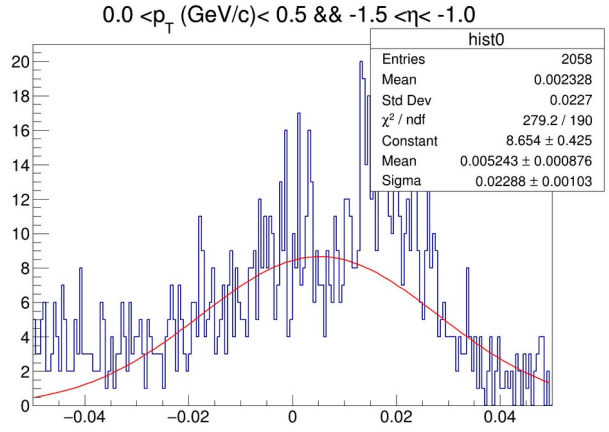
Smearing true parameters by 10%

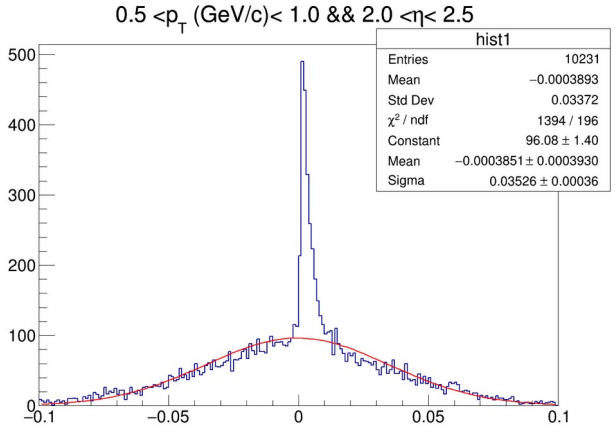


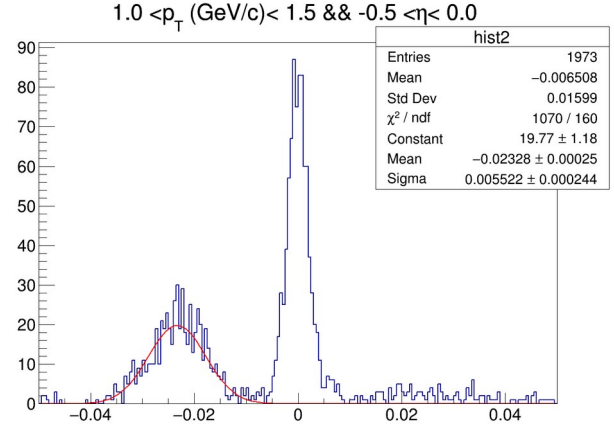
Cases of 90% smearing (DCA_{xy})

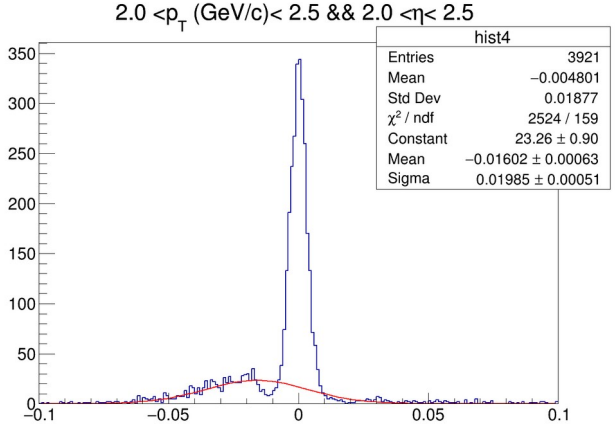
Worst case of smearing

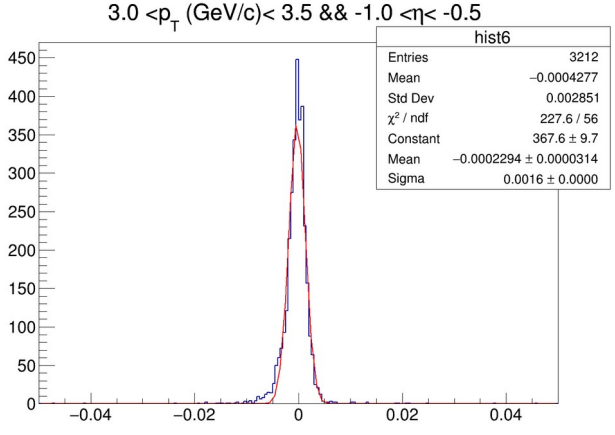






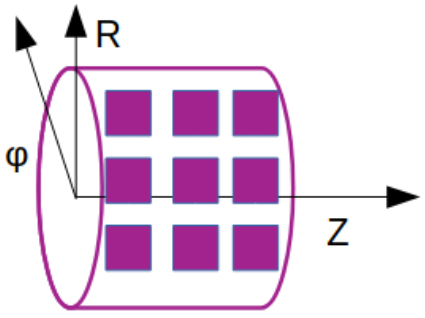






Track Parameters (Global to Local)

$\sigma_\varphi = \sigma_z = \text{pixel size}/\sqrt{(12)}$



Cylinder

Global coordinates: (x,y,z,p_x,p_y,p_z)

Track parameters on a surface (l_1,l_2 are local)

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

https://github.com/NicholasLukow/ATHENA_Tracking_GEM/blob/master/Fun4All_G4_Baseline2.C

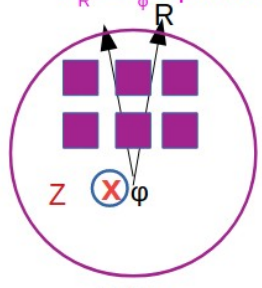
For ACTS see backup

First two local parameters describes the pixel dimensions

Track Parameters in Cylindrical surface:

$$(R\phi, z, \phi, \theta, q/p)$$

$\sigma_R = \sigma_\varphi = \text{pixel size}/\sqrt{(12)}$



Disks

Track Parameters on Disk surface:

$$(R, R\phi, \phi, \theta, q/p)$$

Cylinder (Fun4All)

Class PHG4TrackFastSim

Disks (Fun4All)

```
PHG4TrackFastSim *kalman = new PHG4TrackFastSim("PHG4TrackFastSim");
// add Vertexing Layers
kalman->add_phg4hits(
                "G4HIT_SVTX",
                PHG4TrackFastSim::Cylinder,
                999.,
                pix_size_vtx/10000./sqrt(12.),
                pix_size_vtx/10000./sqrt(12.),
                1,
                0
```

```
// const std::string& phg4hitsNames,

// radial-resolution [cm]
// azimuthal-resolution [cm]
// z-resolution [cm]
// efficiency,
// noise hits
```

```
//  add Disk Layers
kalman->add_phg4hits(
                "G4HIT_FBST",
                PHG4TrackFastSim::Vertical_Plane,
                pix_size_dis/10000./sqrt(12.),
                pix_size_dis/10000./sqrt(12.),
                999.,
                1,
                0
                );
```

```
// const std::string& phg4hitsNames,

// radial-resolution [cm]
// azimuthal-resolution [cm]
// z-resolution [cm]
// efficiency,
// noise hits
```

General equation of a circle

$$(x-x_0)^2+(y-y_0)^2=R^2$$

If circle passed through the vertex (0,0): $x_0^2+y_0^2=R^2$

$$x^2+x_0^2-2xx_0+y^2+y_0^2-2yy_0=R^2$$

using

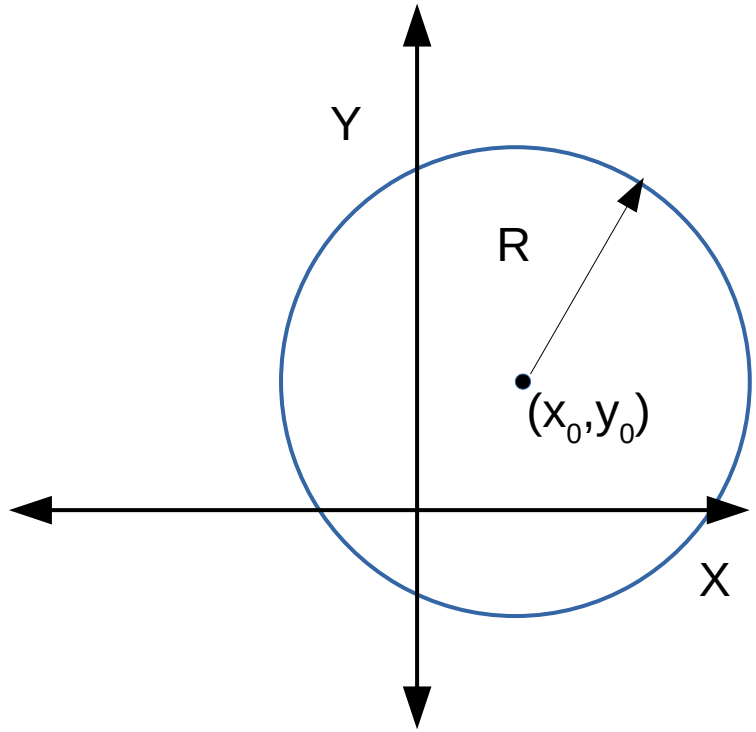
$$u=\frac{x}{x^2+y^2} \qquad v=\frac{y}{x^2+y^2} \qquad x_0^2+y_0^2=R^2$$

$$x^2+y^2-2xx_0-2yy_0=0$$

$$1-2\frac{xx_0}{x^2+y^2}-2\frac{yy_0}{x^2+y^2}=0$$

$$2ux_0+2vy_0=1$$

Line in u-v space



Find three hits belong to straight line in u-v space and use them as the seeding, later use combinatorial track finding and fitting using the Kalman filter