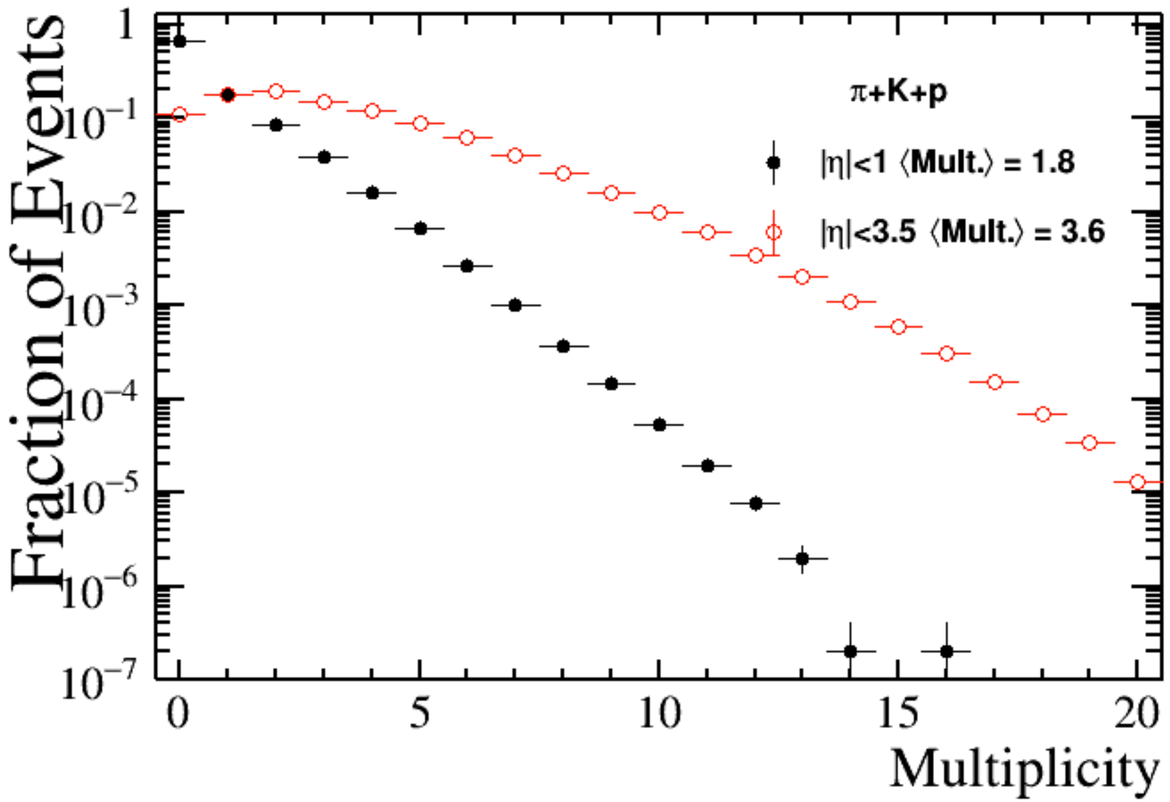
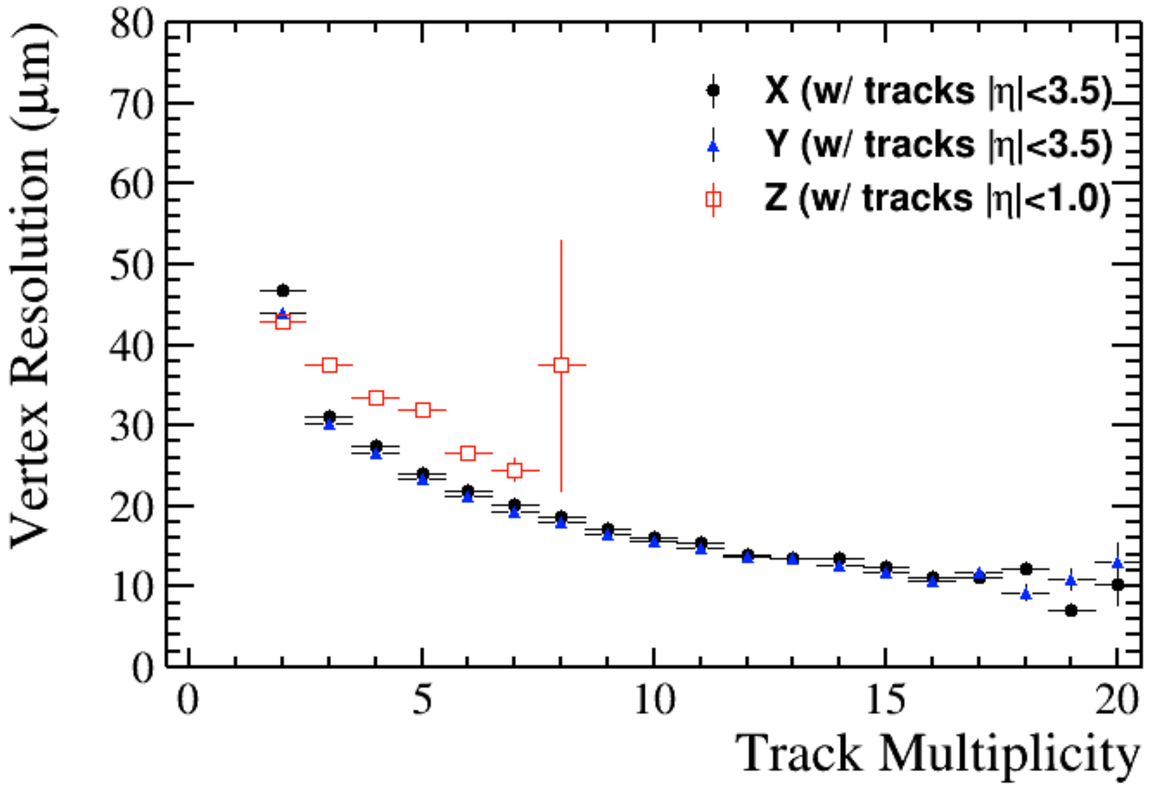
Charm F2 Studies with baseline-2

Matthew Kelsey
Wayne State University

PV Resolution with ATHENA Geom.



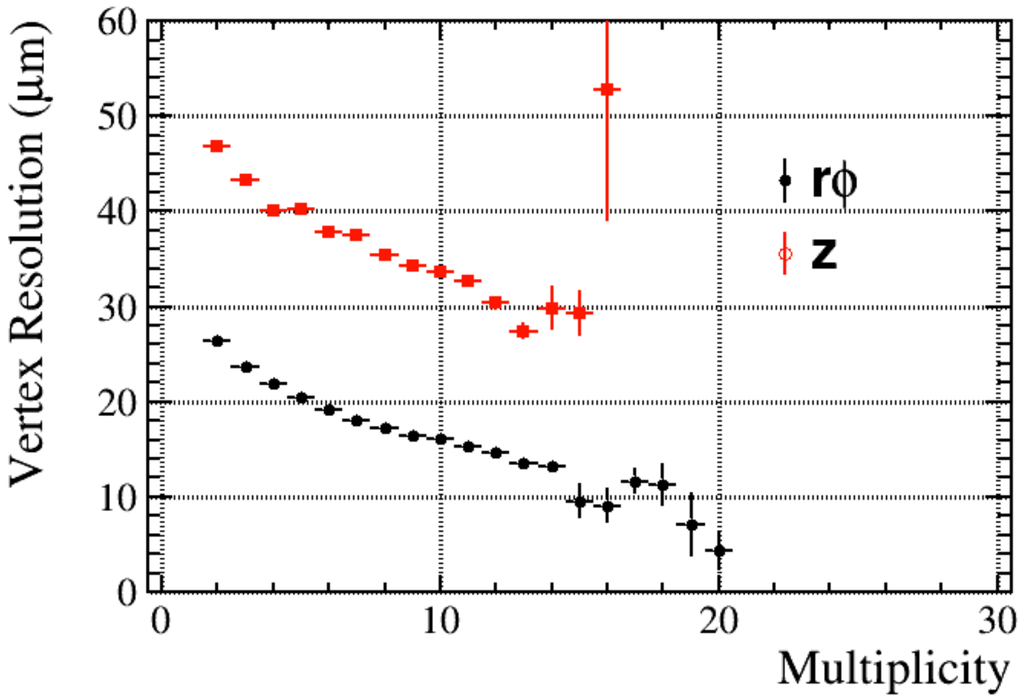


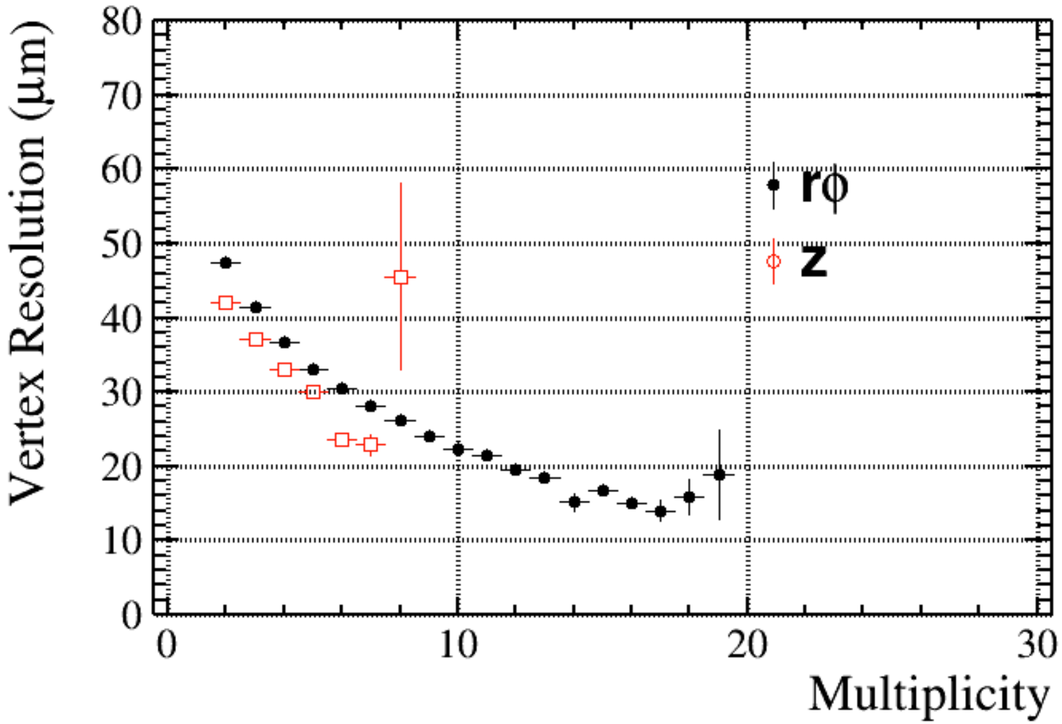
Note, truth seeding starting vertex position

Z resolution improved if we just considered tracks within barrel only

- With all tracks, increased by factor of 2. In any case z resolution here should not be trusted

Comparison with Last Fast Sim

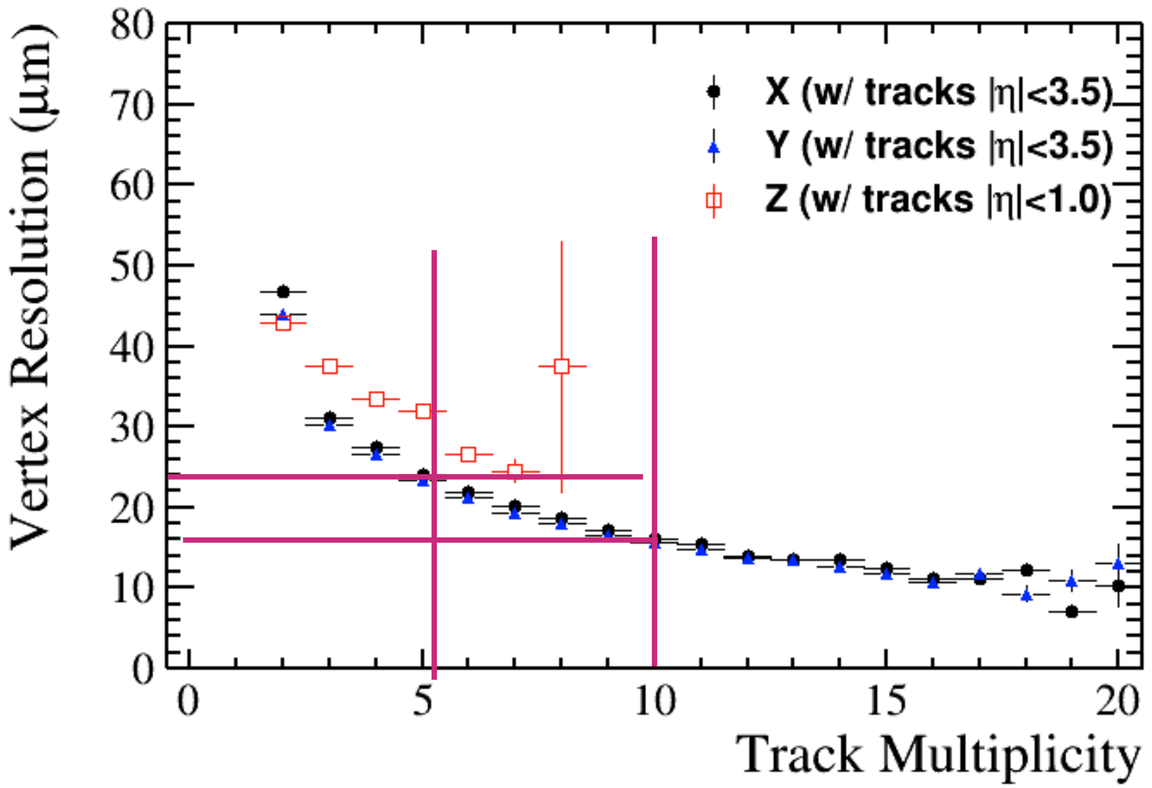


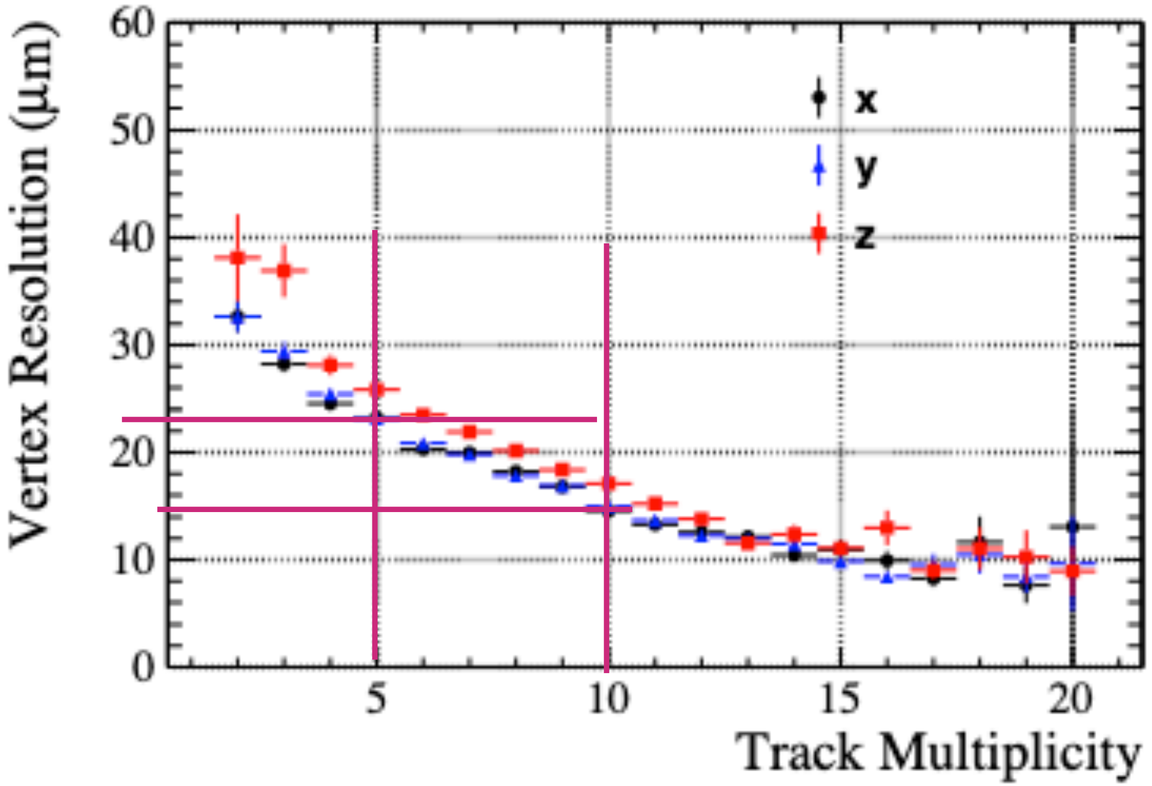


Z resolution not direct comparison (ignore)

R-phi about 50% worse at moderate to high multiplicity

From Charm F2 Paper





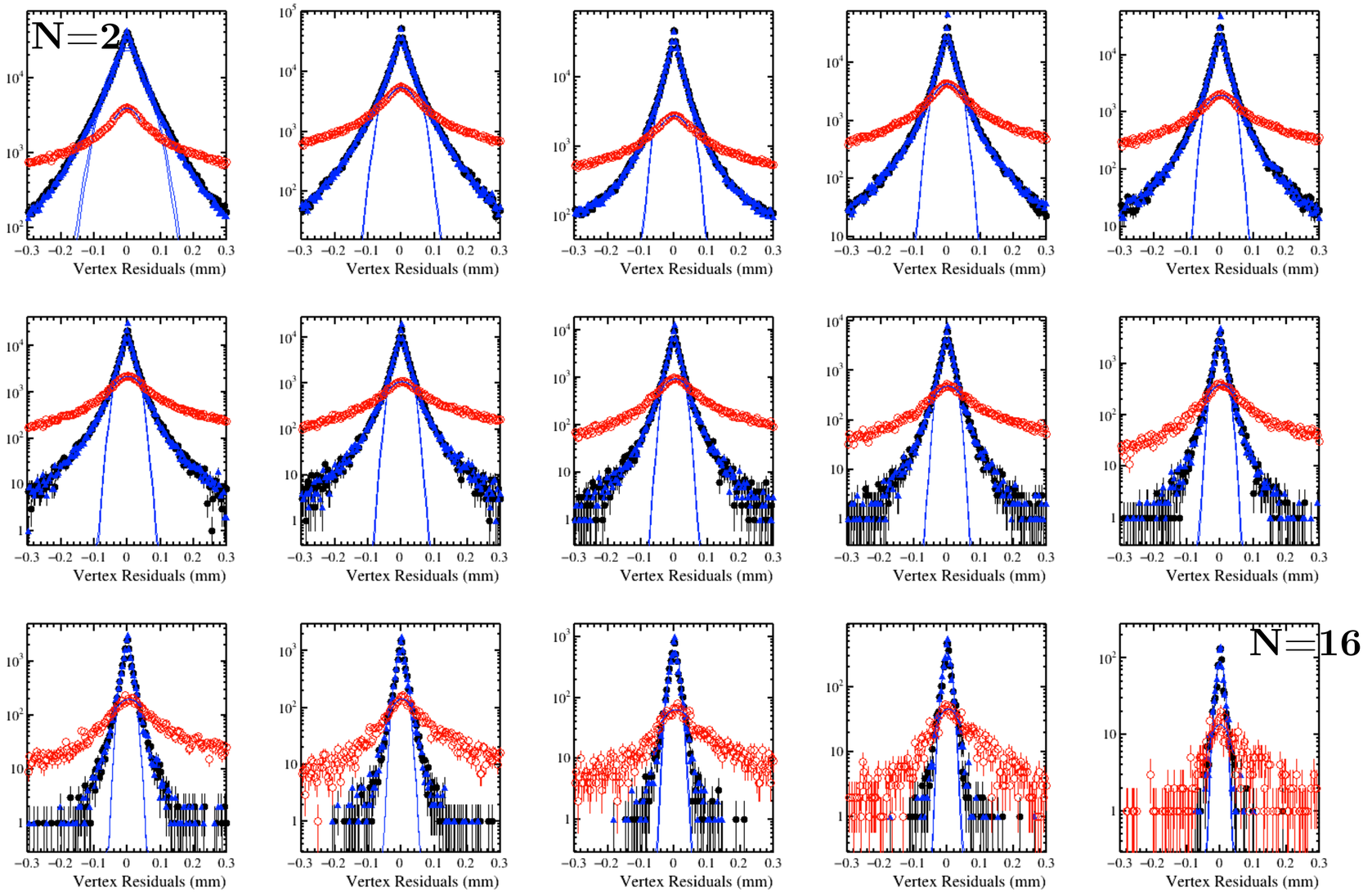
Very comparable with all-Silicon full simulation

PV Residuals w/ Tracks in |eta|<3.5

Black = x

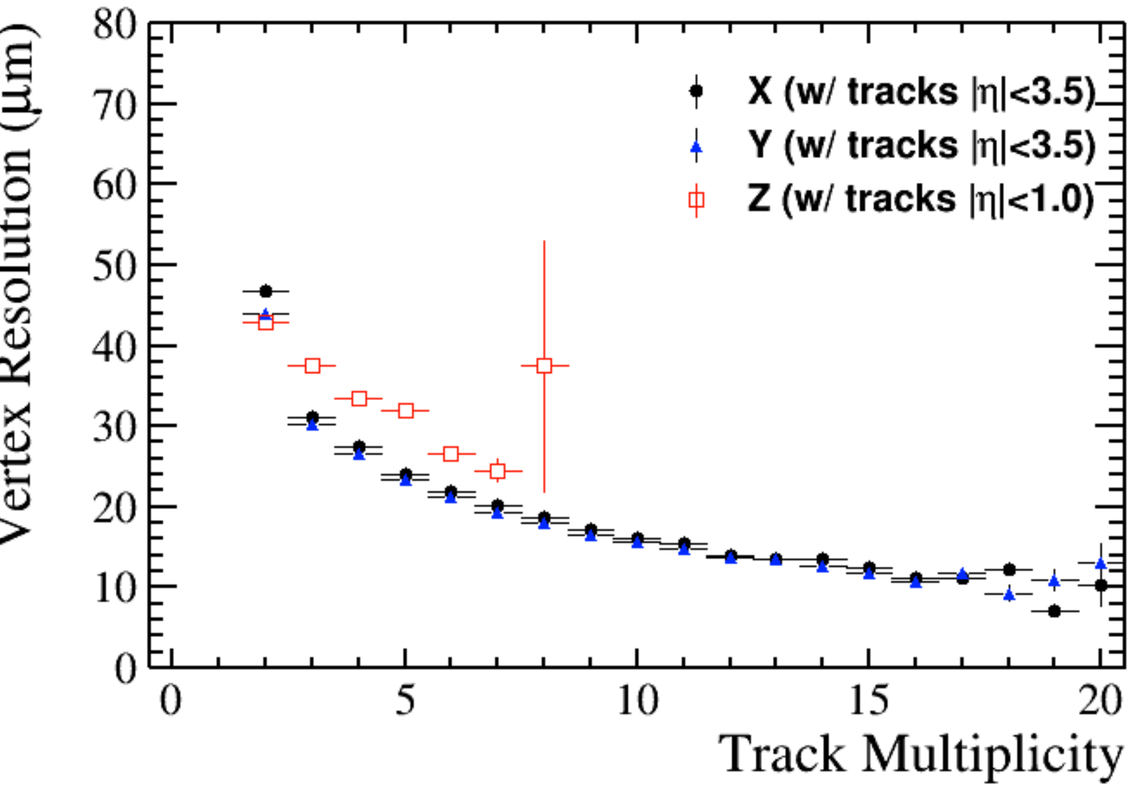
Blue = y

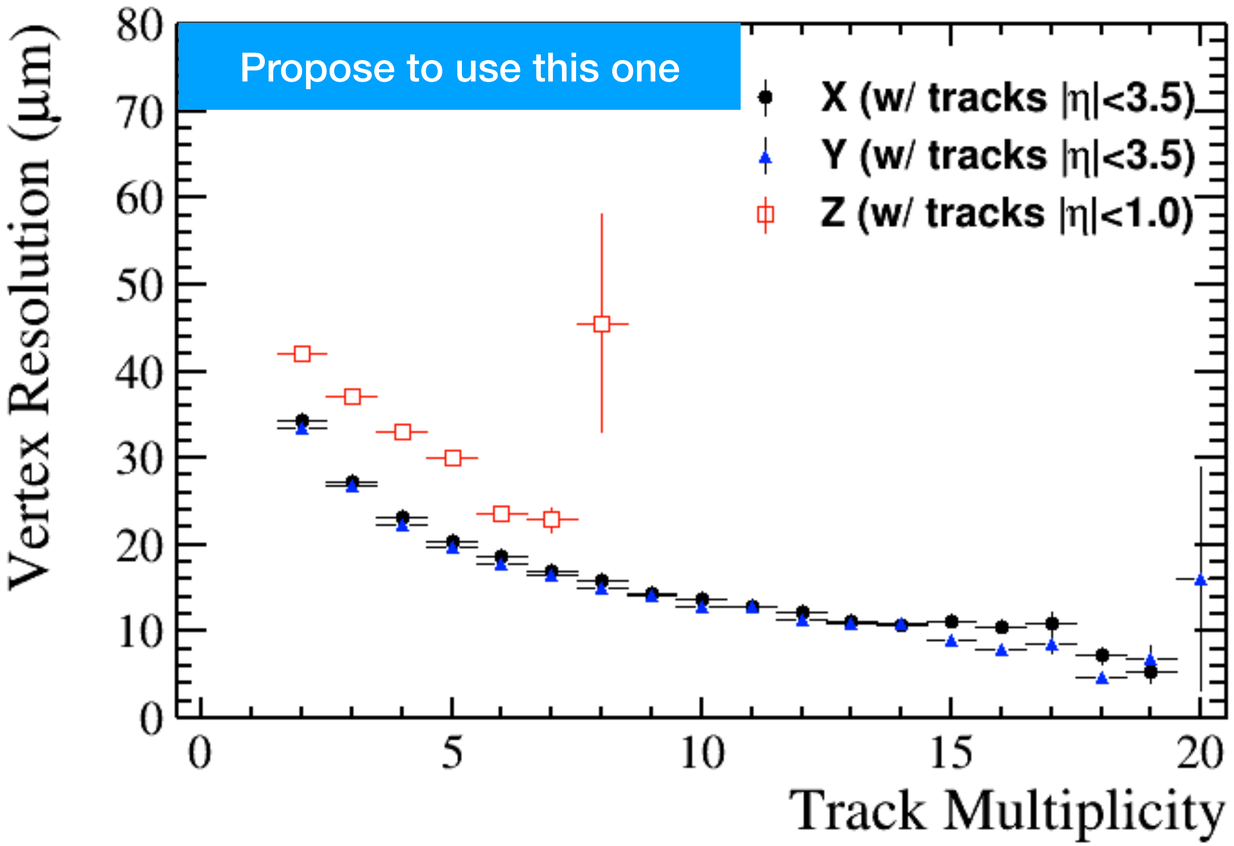
Red = z



Large tails; fit core Gaussian

Improvements in Fit Code

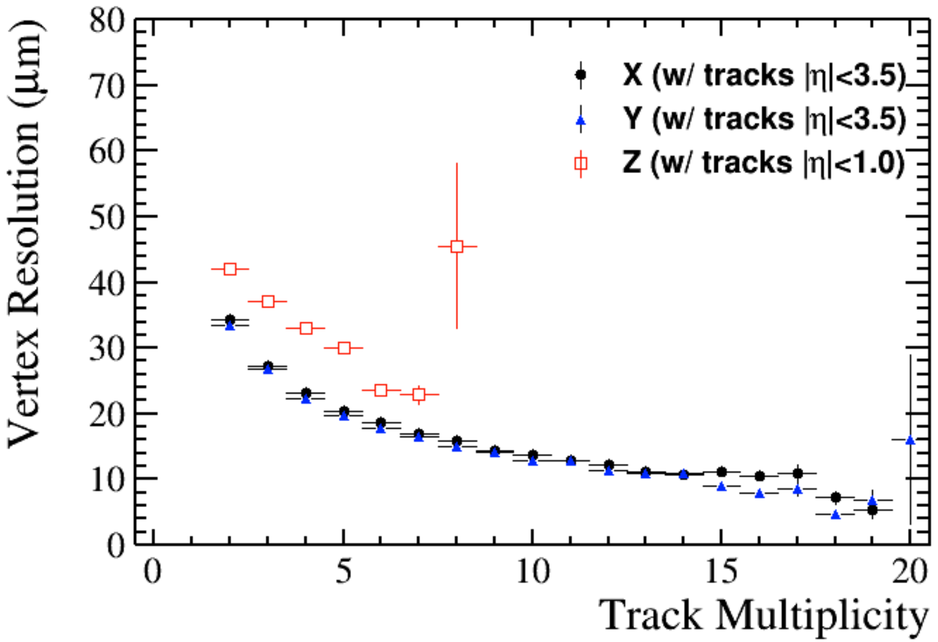


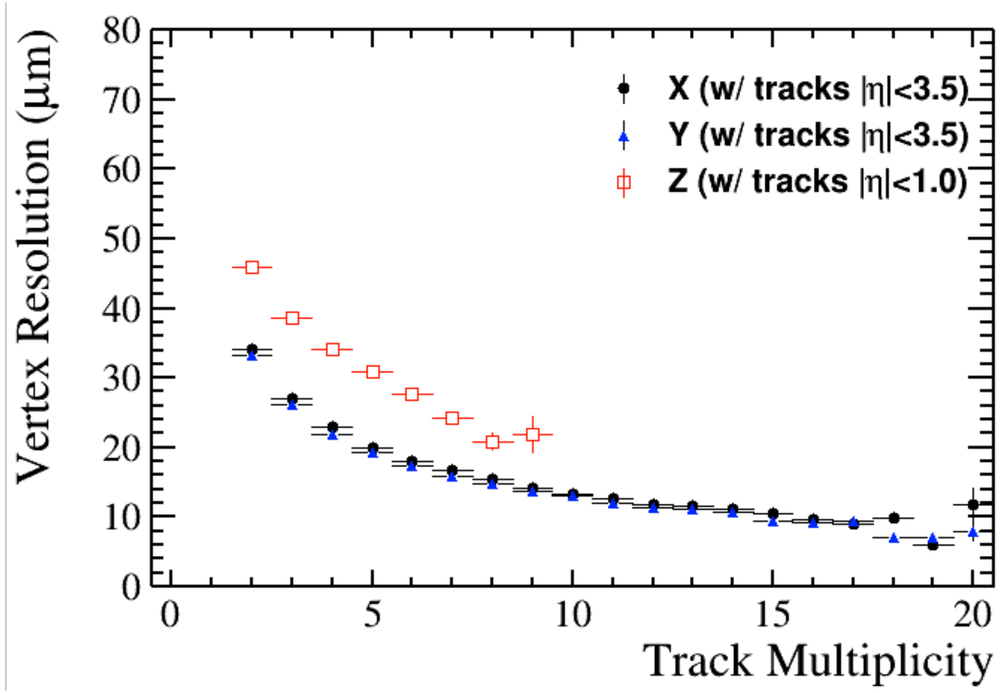


Least squares fit -> Chi2 fit with known DCA resolutions

Low multiplicity bins improved; high bins mainly unchanged

Finer momentum bining for 2D smearing

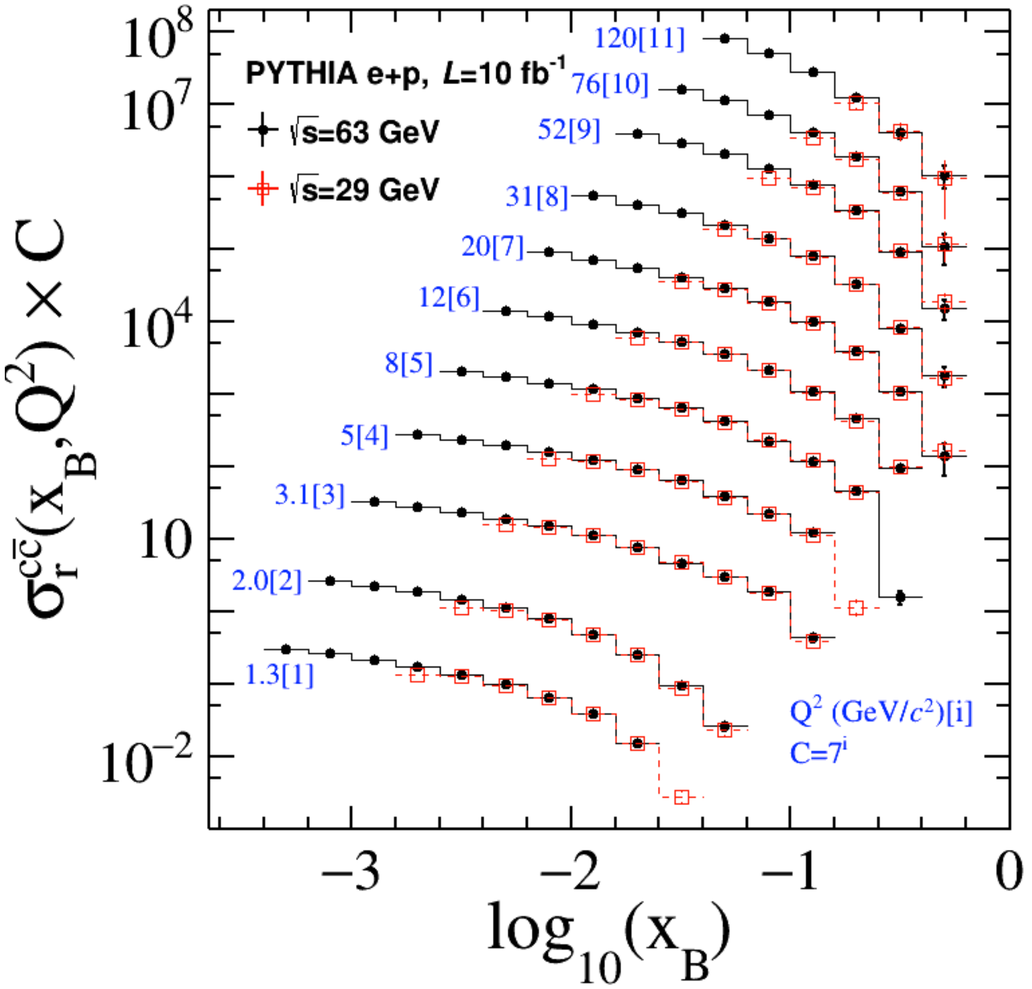


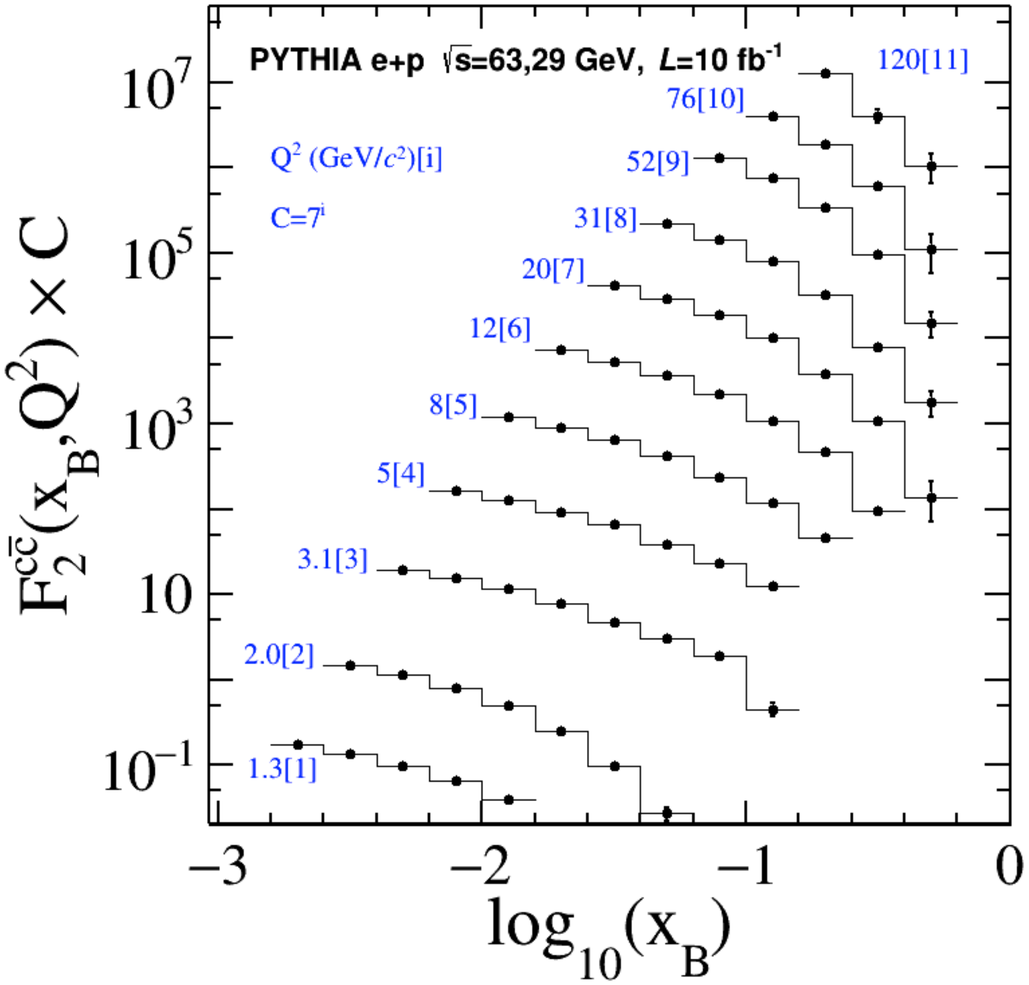


Also higher stats for z vertex resolution

Updated Projections

Reduced Charm X-Sec & F2

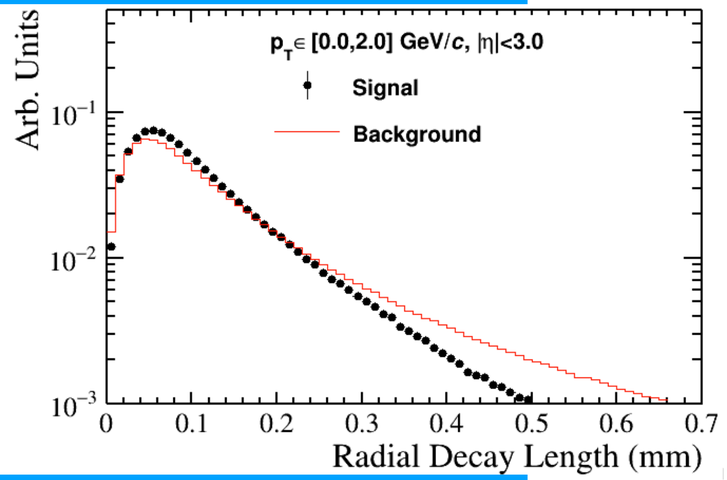


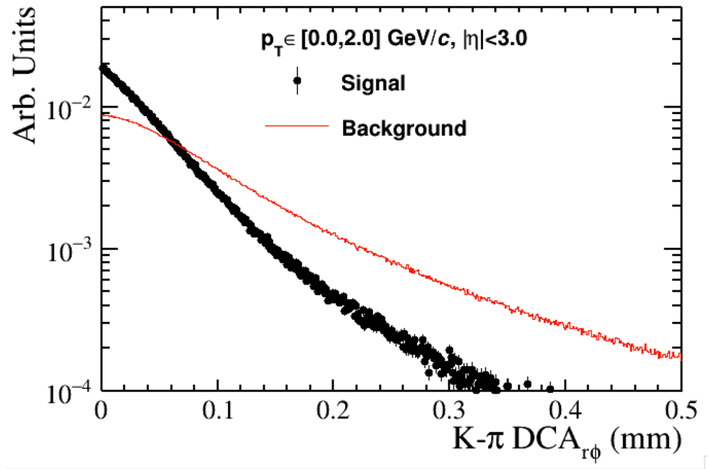


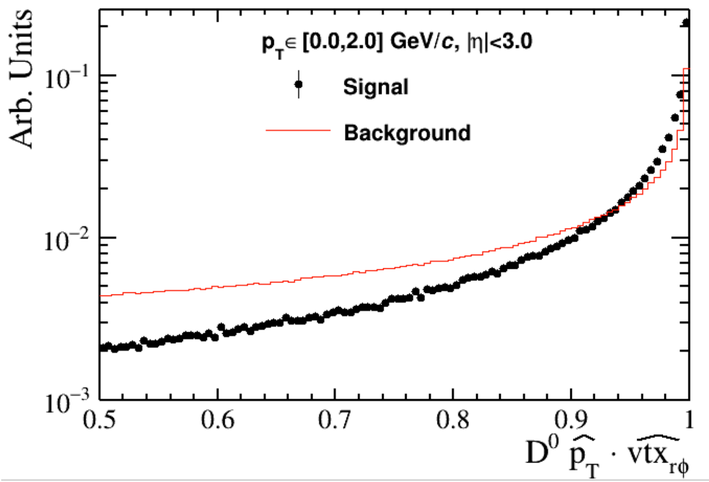
BACKUP

Topo.

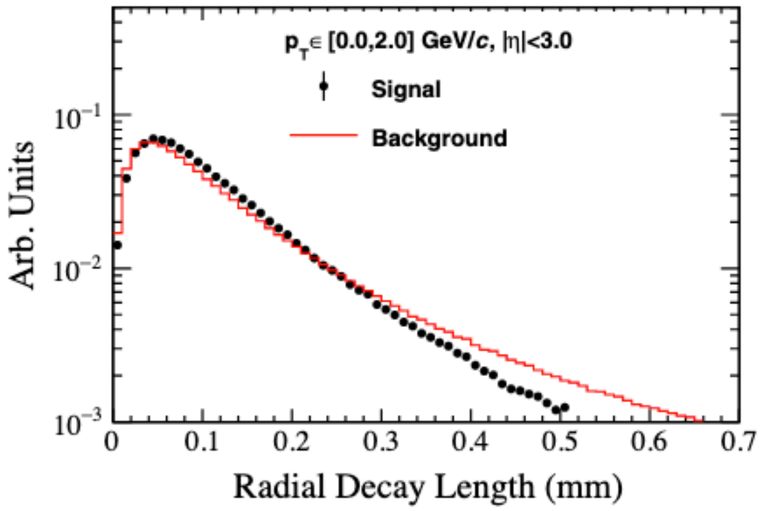
New

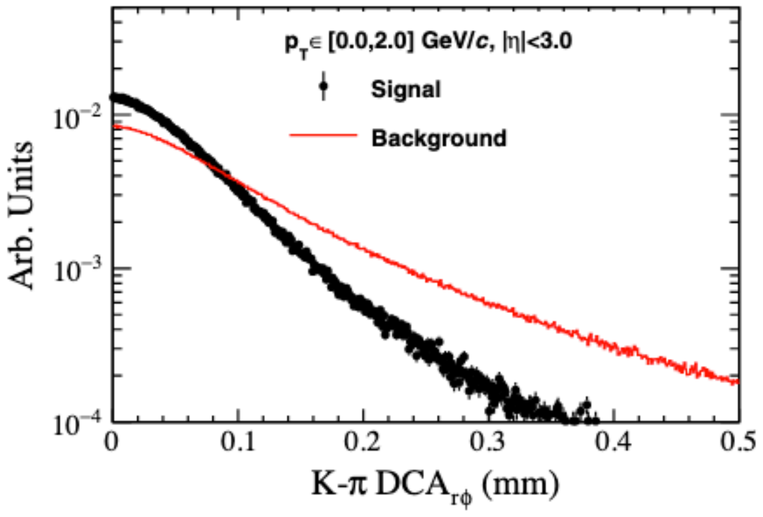


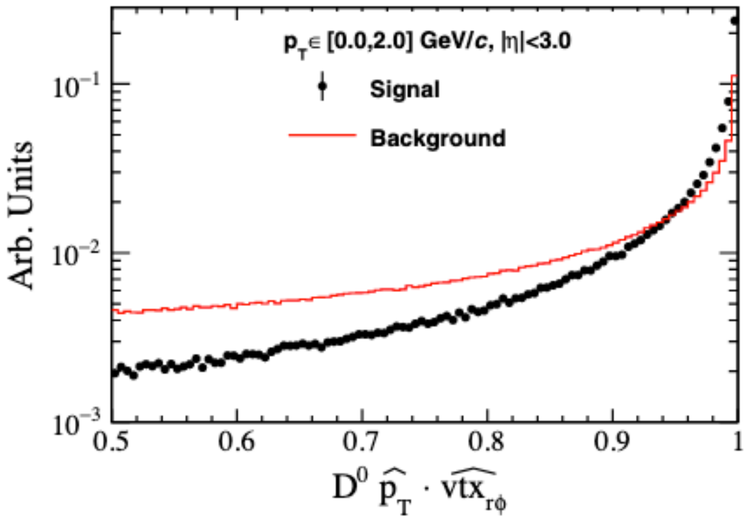




Paper







Get new smearing curves

Arrays of momentum/DCA resolution. Each array object is for one fixed eta bin. Each TGraph has momentum dependence

Res_Handler is needed to keep track of eta bins

```
TFile *track_res = new TFile(“ATHENA_Resolutions_r.root”,"READ");
 TH1F * Res_Handler = (TH1F*)track_res->Get("Res_Handler");
 int const npt = Res_Handler->GetNbinsX();
 TGraph *gmom_res[npt];
 TGraph *gdca_rphi_res[npt];
 TGraph *gdca_z_res[npt];
 for(int i = 0; i<npt; i++){
     gmom_res[i] = (TGraph*)track_res->Get(Form("gmom_res_%i",i));
     gdca_rphi_res[i] = (TGraph*)track_res->Get(Form("gdca_rphi_res_%i",i));
     gdca_z_res[i] = (TGraph*)track_res->Get(Form("gdca_z_res_%i",i));
 }
```

Res_Handler example

Use it to get array index for smearing graphs

```
TLorentzVector *p1 = new TLorentzVector(mpx[tr],mpy[tr],mpz[tr],me[tr]);
TVector3 p1_vtx(mvx[tr],mvy[tr],mvz[tr]);
int bin1 = 1; // Do this first to be agnostic to tracks outside acceptance (i.e., eta cuts come later)
if(fabs(p1->PseudoRapidity())<=3.5)bin1 = Res_Handler->FindBin(p1->PseudoRapidity());
TLorentzVector *ps1 = smearMomATHENA(p1,gmom_res[bin1-1]);
TVector3 pos = smearPosATHENA(ps1,p1_vtx,gdca_rphi_res[bin1-1],gdca_z_res[bin1-1]);
```

Smear Momentum Function

```
TLorentzVector * smearMomATHENA(TLorentzVector const * b,TGraph * g)
{
    float const pt = b->Perp();
    float const p = b->P();
    float sPt = -1;
    float sP = -1;
    float eta = b->PseudoRapidity();
    // resolutions are in relative percent, need to smear with absolute delta p
    sP = gRandom->Gaus(p,p*g->Eval(p));
    sPt = sP*TMath::Sin(b->Theta());
    TLorentzVector *sMom = new TLorentzVector();
    sMom->SetXYZM(sPt * cos(b->Phi()), sPt * sin(b->Phi()), sPt * sinh(eta), b->M());
    return sMom;
}
```

Smear Position Function

```
TVector3 smearPosATHENA(TLorentzVector const * b, TVector3 const& pos, TGraph * g_rphi,  TGraph
* g_z){
    float const pt = b->Perp();
    float const p = b->P();
    float rand_xy = 0;
    float rand_z = 0;
    double eta =  TMath::ATanH(b->Pz()/b->P());//Doing it this way to suppress TVector3
Warnings; depends on any pre-cuts
    // resolutions are in microns, need in mm
    rand_xy=gRandom->Gaus(0,g_rphi->Eval(p))/1000.;
    rand_z=gRandom->Gaus(0,g_z->Eval(p))/1000.;

    TVector3 newPos(pos);
    newPos.SetZ(0);
    TVector3 momPerp(-b->Vect().Y(), b->Vect().X(), 0.0);
    newPos -= momPerp.Unit() * rand_xy;

    return TVector3(newPos.X(), newPos.Y(), pos.Z() + rand_z);
```

Tracking

```
bool passTrackingATHENA(double p, double eta){
    if(fabs(eta)>3.5)return false;
    if(p<0.5)return false;
    return true;
}
```