

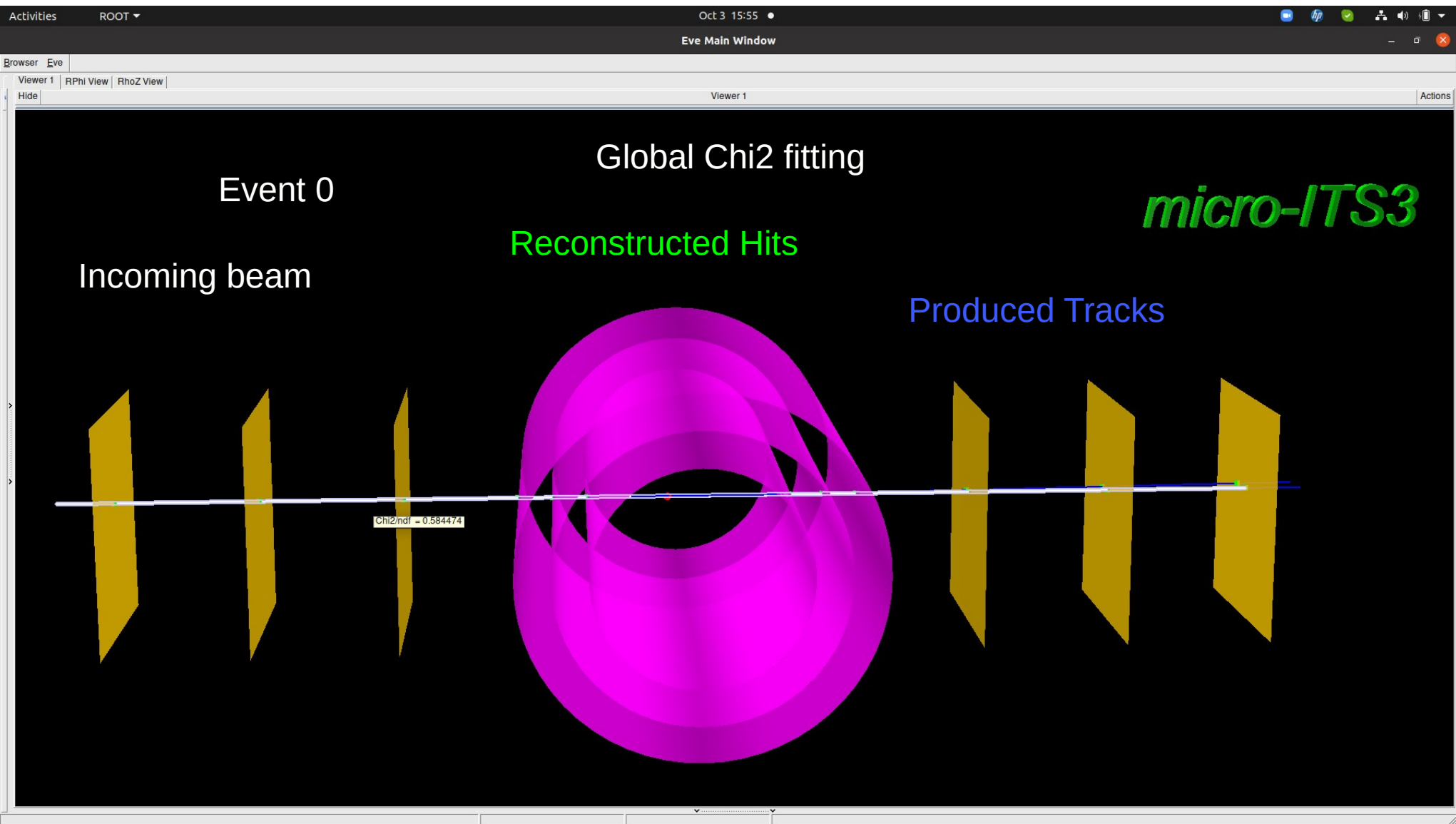
Tracking using Kalman filter in ALICE Framework

Shyam Kumar
INFN Bari, Italy

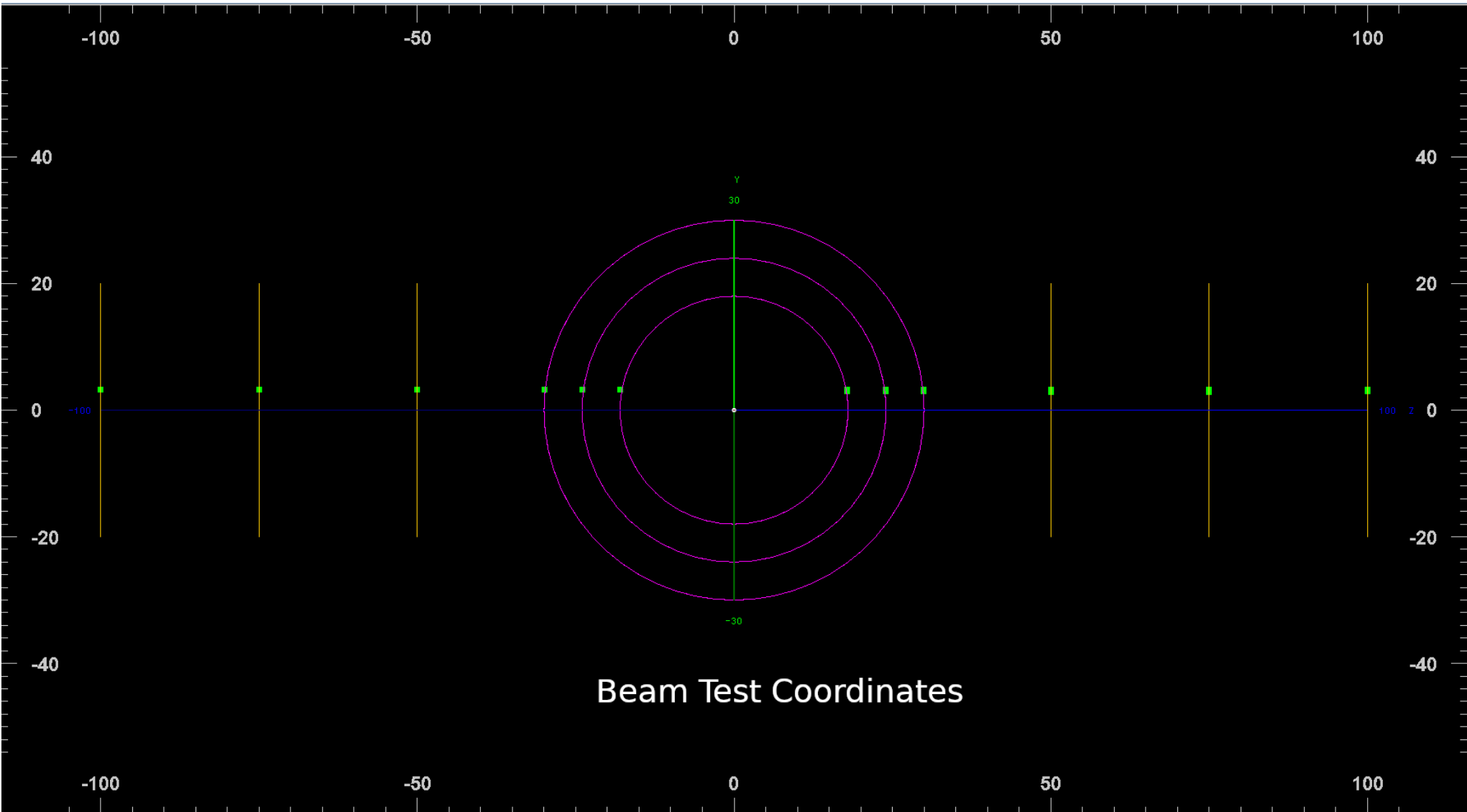
https://agenda.infn.it/event/1096/contributions/6159/attachments/4504/4980/Rotondi_3.pdf

<http://www.le.infn.it/lhcschool/talks/Ragusa.pdf>

Event display



Beam Test



Fitting Tracks with Straight Lines (Chi2 Minimization)

Let's understand lines in 3D for fitting them

Vector equation of a line in 3D:

$$\vec{r} = \vec{r}_0 + \vec{a}$$

If u is the unit vector along the line and t is parameter:

$$\vec{r} = \vec{r}_0 + t\vec{u}$$

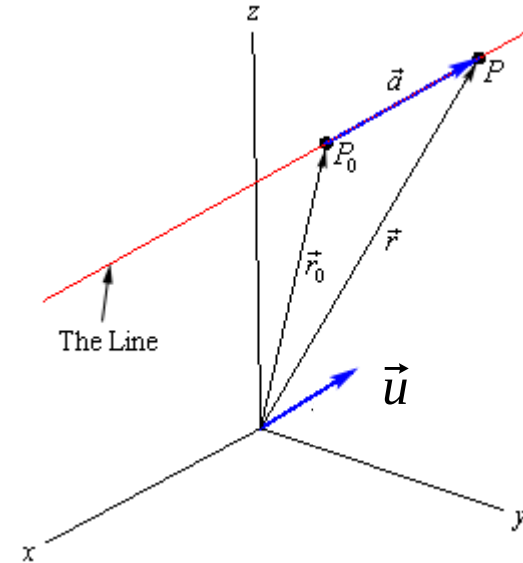
$$(x, y, z) = (x_0, y_0, z_0) + t(a, b, c)$$

$$x = x_0 + ta$$

$$y = y_0 + tb$$

$$z = z_0 + tc$$

$$t = \frac{(x - x_0)}{a} = \frac{(y - y_0)}{b} = \frac{(z - z_0)}{c}$$



To define a line in 3D, we should know a point (x_0, y_0, z_0) on the line and unit vector (a, b, c) in the direction of line
Therefore 6 parameters to minimize!!!

<https://tutorial.math.lamar.edu/classes/calciiii/eqnsoline.aspx>

Fitting Lines in 3D (General)

General line in 3D with 6 parameters:

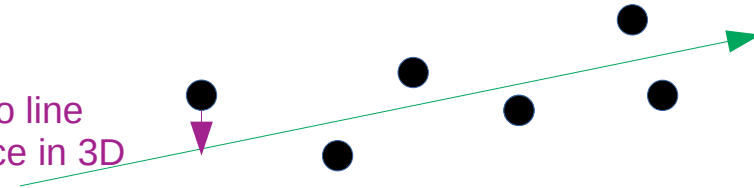
$$x = p_0 + t p_1$$

$$y = p_2 + t p_3$$

$$z = p_4 + t p_5$$

We have given hit points $(x_0, y_0, z_0), (x_1, y_1, z_1), \dots$
Minimize of Chi2 to best estimate the parameters

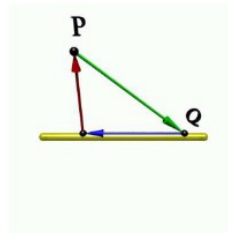
Point to line
distance in 3D



DISTANCE POINT-LINE (3D). If P is a point in space and L is the line $\vec{r}(t) = Q + t\vec{u}$, then

$$d(P, L) = \frac{|(\vec{PQ}) \times \vec{u}|}{|\vec{u}|}$$

is the distance between P and the line L . Proof: the area divided by base length is height of parallelogram.



Here we are minimizing distance
correspond to a general point

```
// define the parametric line equation
void line(double t, const double *p, double &x, double &y, double &z) {
    // a parametric line is define from 6 parameters but 4 are independent
    // x0,y0,z0,z1,y1,z1 which are the coordinates of two points on the line
    x = p[0] + p[1]*t;
    y = p[2] + p[3]*t;
    z = p[4] + p[5]*t;
}
// where ux is direction of line and x0 is a point in the line (like t = 0)
XYZVector xp(x,y,z);
XYZVector x0(p[0], p[2], p[4])
// distance line point is D= | (xp-x0) cross ux |
```

<https://github.com/Simple-Shyam/Phd-work/blob/master/HFPPT/distance.pdf>

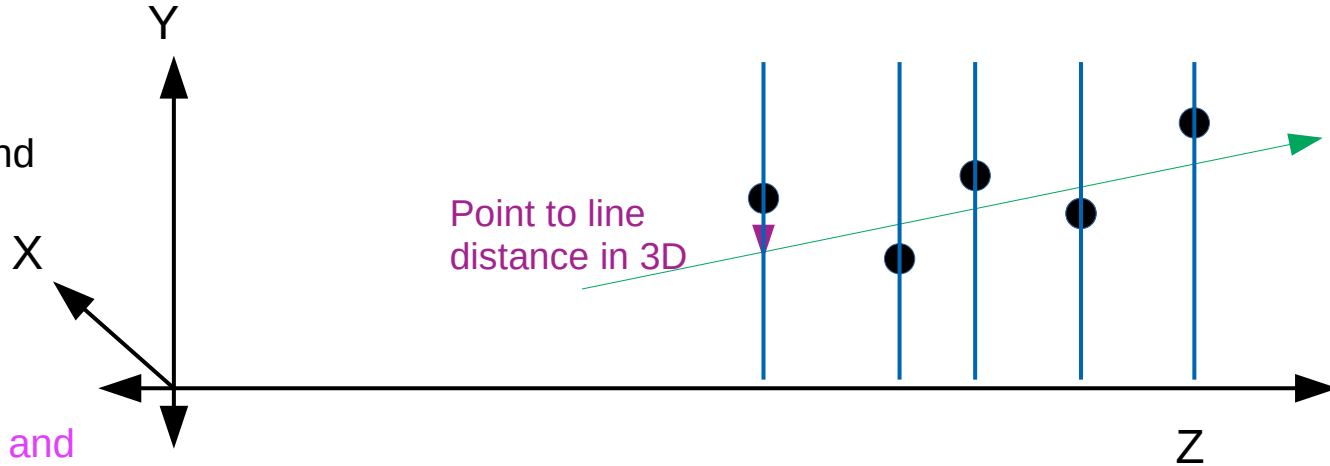
https://root.cern.ch/doc/master/line3Dfit_8C.html

Straight Line Fit in 3D (Improvement)

$$\begin{aligned}x &= p[0] + p[1]*t; \\y &= p[2] + p[3]*t; \\z &= p[4] + p[5]*t;\end{aligned}$$

(Chi2 Minimization)

Here we are minimizing distance correspond to point $t=0$ for which z_{hit} and z_{line} both are same



Ndf: 2 (number of planes)-4

Each plane has 2 degrees of freedom and lines has 4 independent parameters (constraint)

```
x = p[0] + p[1]*t;
y = p[2] + p[3]*t;
z = t;
```

```
double ti = z;
```

Improvement after discussion with Annalisa
 $p4 = 0$ and $p5 = 1$ reduces two parameters

```
XYZVector xp(x,y,z);
double ti = (z-p[4])/p[5];
XYZVector u(p[0]+p[1]*ti,p[2]+p[3]*ti,z);
double d2 = ((xp-u).Mag2());
double dx = (xp-u).X(); double dy = (xp-u).Y();
double chi2 = dx*dx/(xerr*xerr)+dy*dy/(yerr*yerr);
return chi2;
```

Corryvrecken also using the similar method as given the link of the class

<https://gitlab.cern.ch/corryvrecken/corryvrecken/-/blob/master/src/objects/StraightLineTrack.cpp>

ALPIDE Layers (See Geometry)

ALPIDE Planes: For Planes

$$\chi^2 = \frac{dx^2}{\sigma_x^2} + \frac{dy^2}{\sigma_y^2}$$

$$\sigma_x = \sigma_y = 5 \mu m$$

ALPIDE Cylindrical:

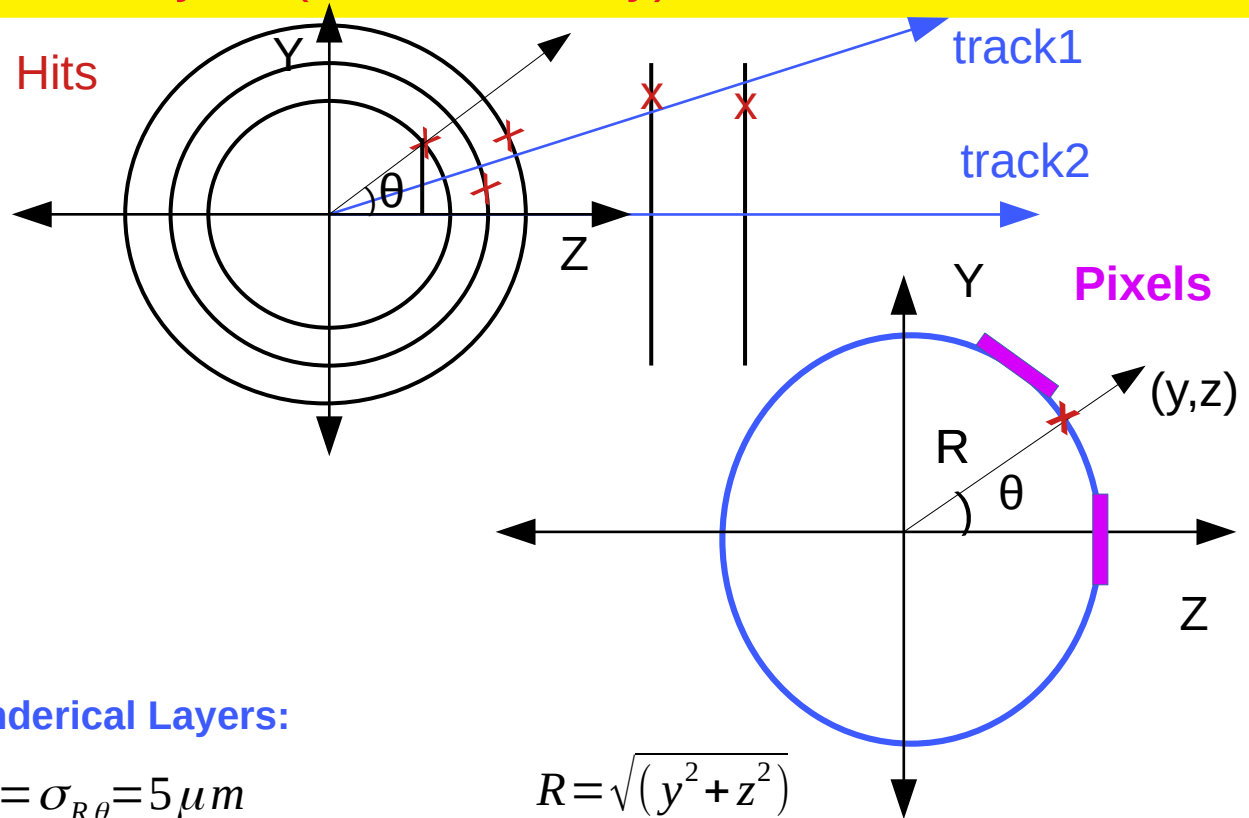
σ_x will be the same

$$\chi^2 = \frac{dx^2}{\sigma_x^2} + \frac{dy^2}{\sigma_y^2}$$

$$\theta = \tan^{-1} \left(\frac{y}{z} \right)$$

$$\sigma_y' = \sigma_{R\theta} \cos \theta$$

Sensor frame will be helpful



For Cylindrical Layers:

$$\sigma_x = \sigma_{R\theta} = 5 \mu m$$

$$R = \sqrt{(y^2 + z^2)}$$

$$z = R \cos \theta$$

$$y = R \sin \theta$$

$$dy = R \cos \theta d\theta$$

$$dy = R \cos \theta d\theta = d(R\theta) \cos \theta$$

Straight Line Fit (With Multiple Scattering)

$$y = a + bz$$

If points on planes are uncorrelated

Minimize the quantity below (works for Spatial resolutions):

$$\chi^2 = \sum_{i=0}^N \frac{(y_m - y_i)^2}{\sigma_i^2} = \sum_{i=0}^N \frac{(y_m - a - bz_i)^2}{\sigma_i^2}$$

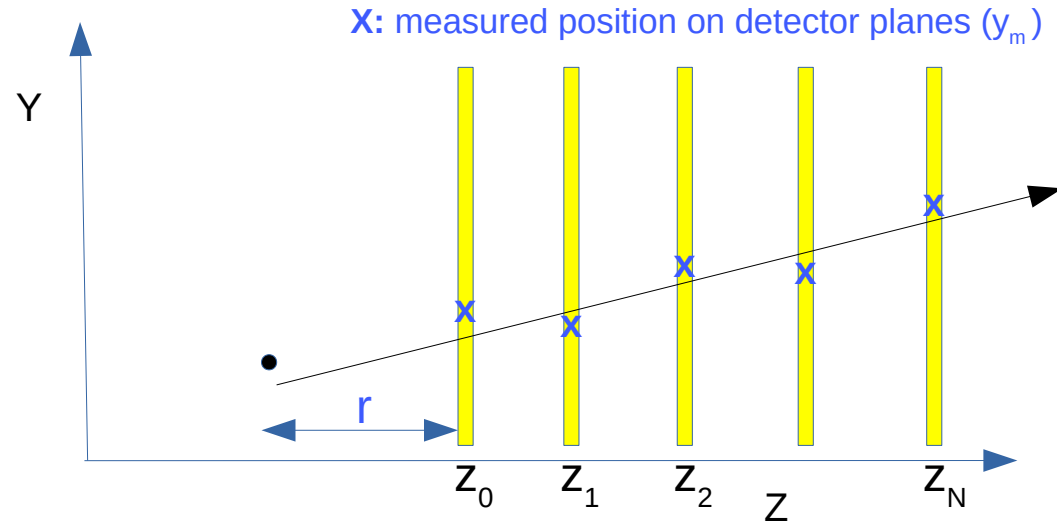
Multiple Scattering between planes are highly correlated, then quantity to be Minimized:

$$\chi^2 = \sum_{i,j=0}^N \frac{(y_{m_i} - y_i)(y_{m_j} - y_j)}{\sigma_{ij}}$$

For 100 points:

100x100 matrix difficult to Inverse (Chi2 fitting)

For Kalman filter 100 matrix of 5x5 dimensions



$$\chi^2 = (Y - Ap)^T (V_{SR} + V_{MS})^{-1} (Y - Ap)$$

$$V_{SR} = \begin{pmatrix} \sigma_0^2 & \dots & 0 & \dots & 0 \\ 0 & \sigma_1^2 & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \sigma_N^2 & \dots \end{pmatrix} \quad V_{MS} = \begin{pmatrix} \sigma_0^2 & \sigma_{01\dots} \\ 0 & \sigma_1^2 \dots \\ \dots & \dots \\ 0 & 0 \dots \sigma_N^2 \end{pmatrix}$$

MS matrix is non-diagonal

Local Coordinate ALICE

Tracking frame is Sensor local frame

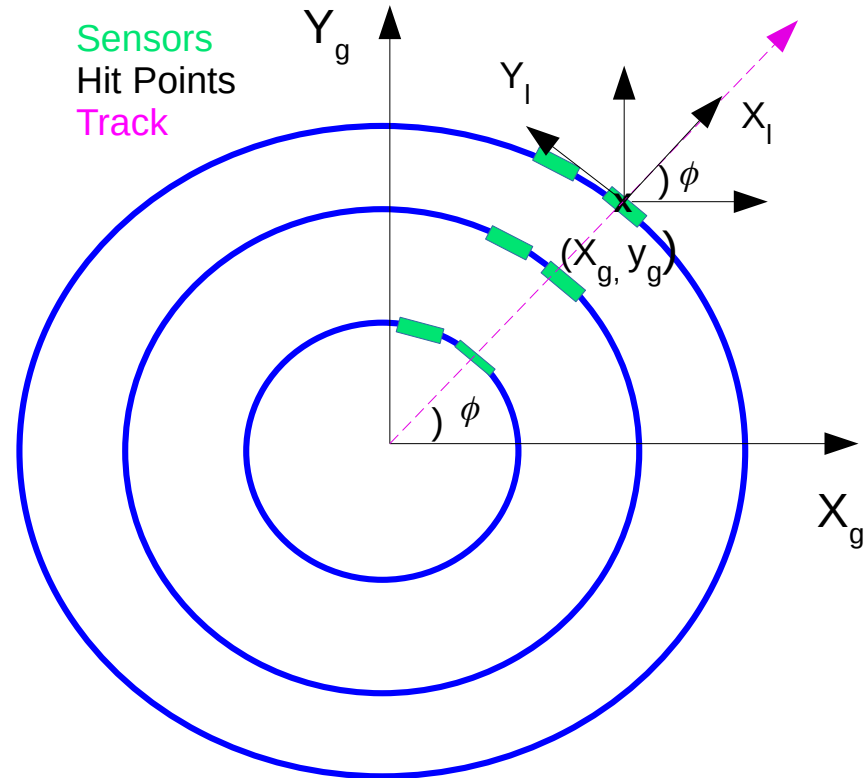
X local is normal to the sensor

Y local is on the sensor

In this coordinate system: ylocal and zlocal will describes the sensor

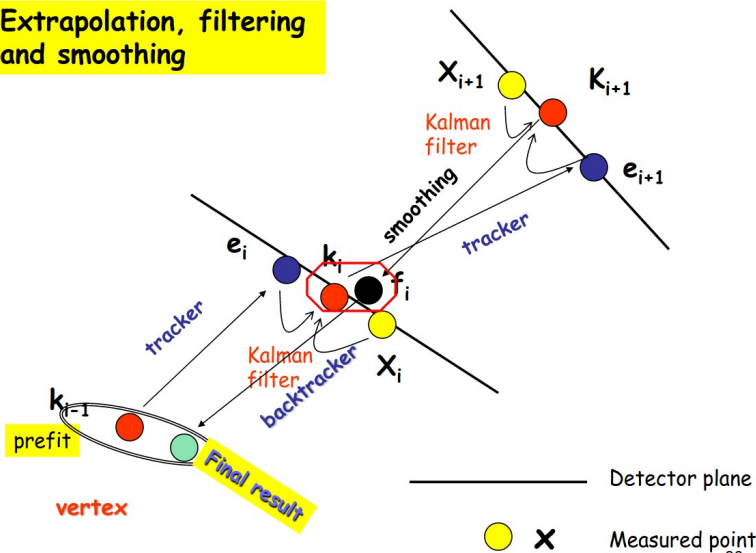
Uncertainties in ylocal is $\sigma(r\phi)$ and on zlocal is $\sigma(z)$
 x_1 is describing the radius

$$\phi = \tan^{-1}\left(\frac{y_g}{x_g}\right)$$



Track Parameters (Local): $(y_l, z_l, \sin \phi, \tan \lambda, q/p_T)$

Extrapolation, filtering and smoothing



Kalman Filter Method:

1. **Extrapolation:** Extrapolation (e_i) on the next plane with M.S. effect
2. **Filtering:** Weighted average of extrapolated value (e_i) and measured value (x_i), known as Kalman filtered value (k_i)
3. **Smoothing:** Estimation of parameters to extrapolate

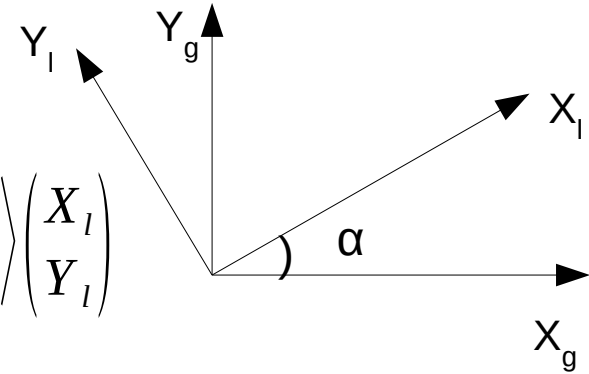
ALICE Track Parameters

Track Parameters (Local): $(y, z, \sin \phi, \tan \lambda, q/p_T)$

On cylindrical surface:

$$x^2 + y^2 = R^2$$

$$\begin{pmatrix} X_g \\ Y_g \end{pmatrix} = \begin{pmatrix} \cos \alpha & -\sin \alpha \\ \sin \alpha & \cos \alpha \end{pmatrix} \begin{pmatrix} X_l \\ Y_l \end{pmatrix}$$



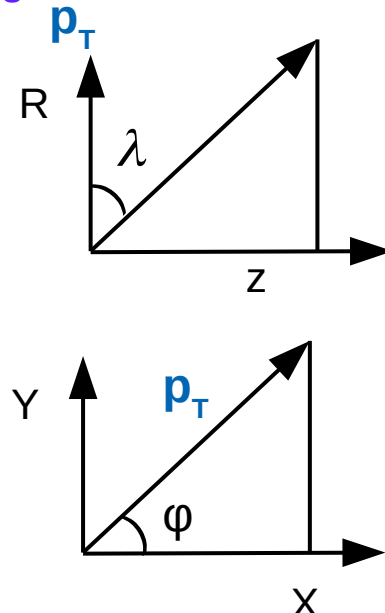
Local coordinates rotated by α w.r.t. global coordinates

$$\frac{\sigma_{1/p_t}}{(1/p_t)} = \frac{1/p_t^2 * \sigma_{p_t}}{(1/p_t)} = \frac{\sigma_{p_t}}{p_t} \quad \sigma_{DCA_{r\phi}}^2 = \sigma_y^2 \quad \sigma_{DCA_z}^2 = \sigma_z^2$$

Parameter Covariance

Track Parameters $\begin{pmatrix} y \\ z \\ \sin \phi \\ \tan \lambda \\ 1/p_T \end{pmatrix}$

$$W = \begin{pmatrix} \sigma_y^2 & \sigma_{yz} & \sigma_{y \sin \phi} & \sigma_{y \tan \lambda} & \sigma_{y \cdot 1/p_T} \\ \sigma_{zy} & \sigma_z^2 & \sigma_{z \sin \phi} & \sigma_{z \tan \lambda} & \sigma_{z \cdot 1/p_T} \\ \sigma_{\sin \phi y} & \sigma_{\sin \phi z} & \sigma_{\sin \phi}^2 & \sigma_{\sin \phi \tan \lambda} & \sigma_{\sin \phi \cdot 1/p_T} \\ \sigma_{\tan \lambda y} & \sigma_{\tan \lambda z} & \sigma_{\tan \lambda \sin \phi} & \sigma_{\tan \lambda}^2 & \sigma_{\tan \lambda \cdot 1/p_T} \\ \sigma_{1/p_T \cdot y} & \sigma_{1/p_T \cdot z} & \sigma_{1/p_T \cdot \sin \phi} & \sigma_{1/p_T \cdot \tan \lambda} & \sigma_{1/p_T}^2 \end{pmatrix}$$



Parameter Covariance matrix = 5X5 matrix; $5(5+1)/2 = 15$ independent entries

$$(\sigma_y^2, \sigma_{yz}, \sigma_{y \sin \phi}, \sigma_{y \tan \lambda}, \sigma_{y \cdot 1/p_T}, \sigma_z^2, \sigma_{z \sin \phi}, \sigma_{z \tan \lambda}, \sigma_{z \cdot 1/p_T}, \sigma_{\sin \phi}^2, \sigma_{\sin \phi \tan \lambda}, \sigma_{\sin \phi \cdot 1/p_T}, \sigma_{\tan \lambda}^2, \sigma_{\tan \lambda \cdot 1/p_T}, \sigma_{1/p_T}^2)$$

Definition of Chi2

$$\chi^2 = (Y_{meas} - Y_{fit})^T W^{-1} (Y_{meas} - Y_{fit})$$

In our case

Chi2 with 5 parameters

```
Double_t
AliExternalTrackParam::GetPredictedChi2(const Double_t p[2], const Double_t cov[3]) const {
//-----
// Estimate the chi2 of the space point "p" with the cov. matrix "cov"
//-----
Double_t sdd = fC[0] + cov[0];
Double_t sdz = fC[1] + cov[1];
Double_t szz = fC[2] + cov[2];
Double_t det = sdd*szz - sdz*sdz;

if (TMath::Abs(det) < kAlmost0) return kVeryBig;

Double_t d = fP[0] - p[0];
Double_t z = fP[1] - p[1];

return (d*szz*d - 2*d*sdz*z + z*sdd*z)/det;
}
```

```
TMatrixDSym c(5);
c(0,0)=GetSigmaY2();
c(1,0)=GetSigmaZY(); c(1,1)=GetSigmaZ2();
c(2,0)=GetSigmaSnPY(); c(2,1)=GetSigmaSnPZ(); c(2,2)=GetSigmaSnP2();
c(3,0)=GetSigmaTg1Y(); c(3,1)=GetSigmaTg1Z(); c(3,2)=GetSigmaTg1SnP(); c(3,3)=GetSigmaTg12();
c(4,0)=GetSigma1PtY(); c(4,1)=GetSigma1PtZ(); c(4,2)=GetSigma1PtSnP(); c(4,3)=GetSigma1PtTg1(); c(4,4)=GetSigma1Pt2();

c(0,0)+=t->GetSigmaY2();
c(1,0)+=t->GetSigmaZY(); c(1,1)+=t->GetSigmaZ2();
c(2,0)+=t->GetSigmaSnPY(); c(2,1)+=t->GetSigmaSnPZ(); c(2,2)+=t->GetSigmaSnP2();
c(3,0)+=t->GetSigmaTg1Y(); c(3,1)+=t->GetSigmaTg1Z(); c(3,2)+=t->GetSigmaTg1SnP(); c(3,3)+=t->GetSigmaTg12();
c(4,0)+=t->GetSigma1PtY(); c(4,1)+=t->GetSigma1PtZ(); c(4,2)+=t->GetSigma1PtSnP(); c(4,3)+=t->GetSigma1PtTg1(); c(4,4)+=t->GetSigma1Pt2();
c(0,1)=c(1,0);
c(0,2)=c(2,0); c(1,2)=c(2,1);
c(0,3)=c(3,0); c(1,3)=c(3,1); c(2,3)=c(3,2);
c(0,4)=c(4,0); c(1,4)=c(4,1); c(2,4)=c(4,2); c(3,4)=c(4,3);

c.Invert();
if (!c.IsValid()) return kVeryBig;

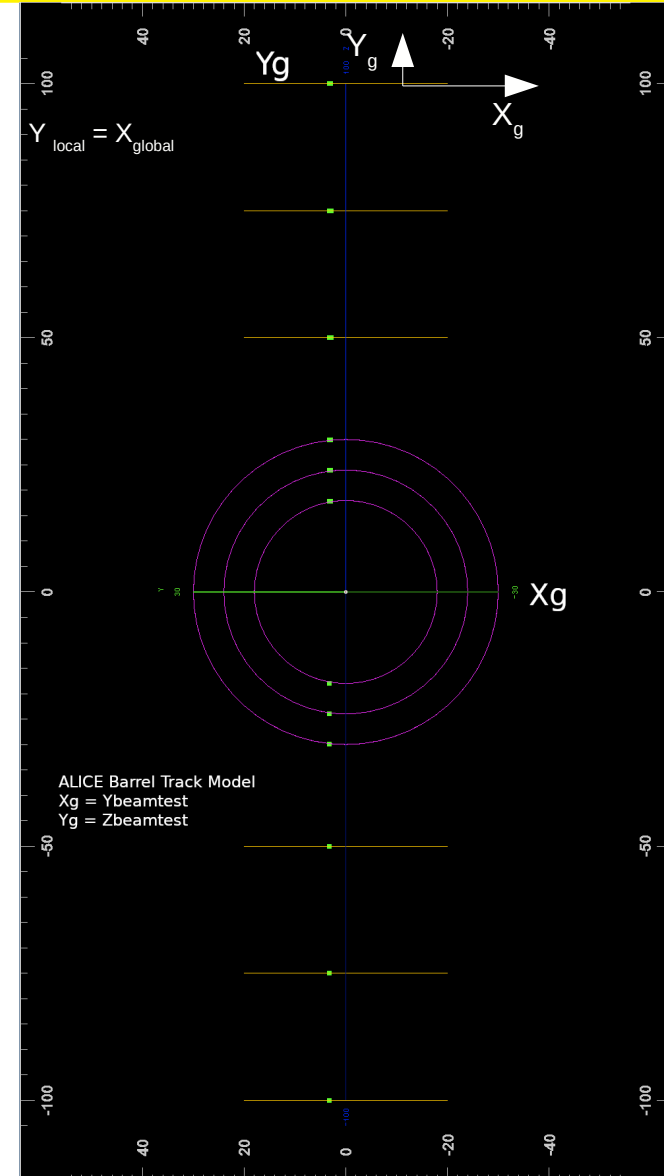
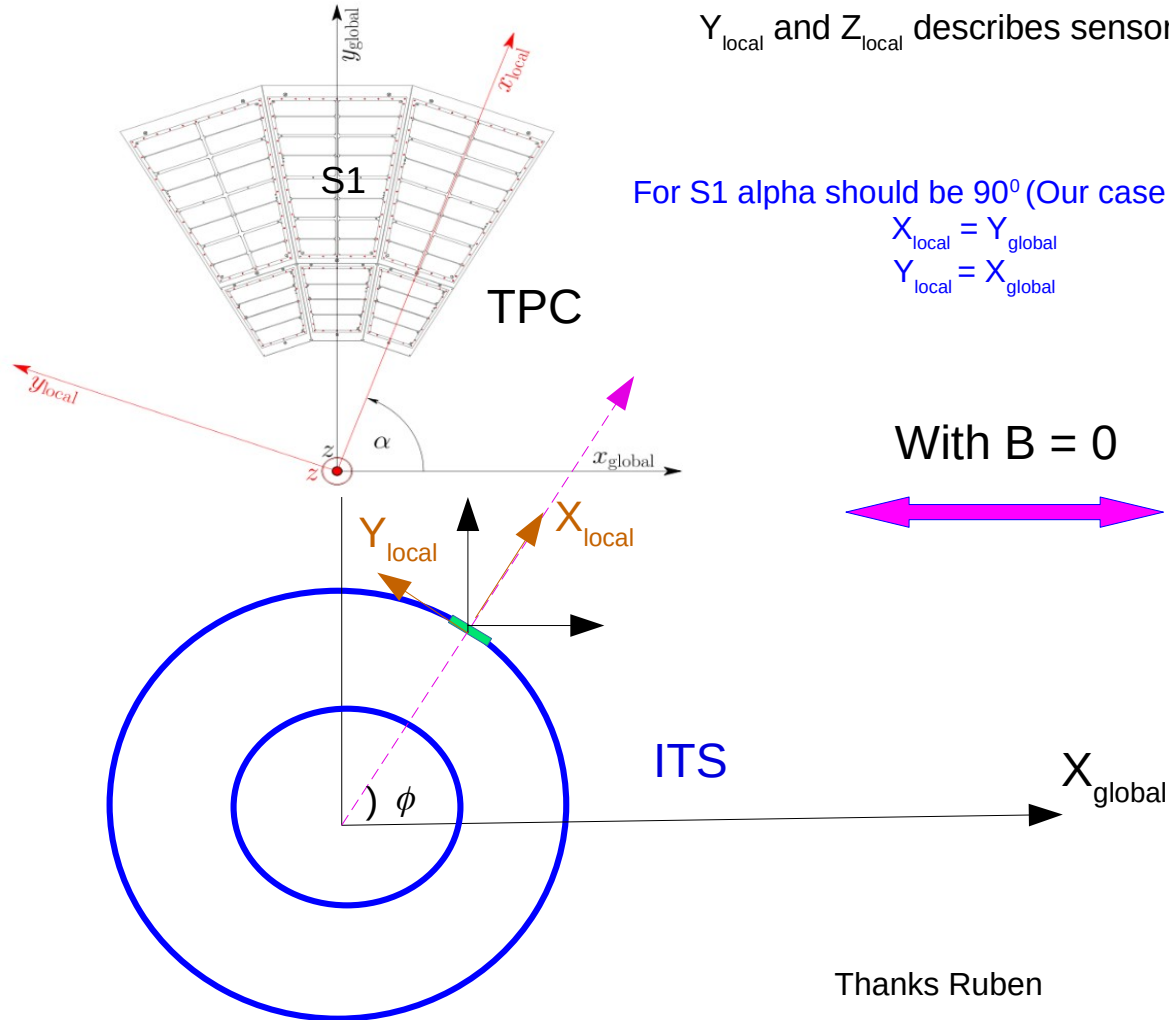
Double_t res[5] = {
GetY() - t->GetY(),
GetZ() - t->GetZ(),
GetSnP() - t->GetSnP(),
GetTg1() - t->GetTg1(),
GetSigned1Pt() - t->GetSigned1Pt()
};

Double_t chi2=0.;
for (Int_t i = 0; i < 5; i++)
for (Int_t j = 0; j < 5; j++) chi2 += res[i]*res[j]*c(1,i,j);

return chi2;
```

TPC + ITS, Beam Test

Sensors TPC single sector is a planar detector
Hit Points so only one rotation is required
Track



Kalman Filter

$$\chi^2 = \frac{(x_p - \mu)^2}{\sigma_p^2} + \frac{(x_m - \mu)^2}{\sigma_m^2}$$

$$\frac{\partial \chi^2}{\partial \mu} = 0 \quad \text{Chi2 minimization}$$

$$\mu = \frac{\frac{x_p}{\sigma_p^2} + \frac{x_m}{\sigma_m^2}}{\frac{1}{\sigma_p^2} + \frac{1}{\sigma_m^2}} = x_p \frac{\sigma_m^2}{\sigma_p^2 + \sigma_m^2} + x_m \frac{\sigma_p^2}{\sigma_p^2 + \sigma_m^2} \quad \sigma(\mu)^2 = \frac{1}{\frac{1}{\sigma_p^2} + \frac{1}{\sigma_m^2}} = \frac{\sigma_p^2 \sigma_m^2}{\sigma_p^2 + \sigma_m^2}$$

$$\text{If } (\sigma_m \gg \sigma_p) \quad \mu \approx x_p$$

$$\text{If } (\sigma_p \gg \sigma_m) \quad \mu \approx x_m$$

$$\mu = x_m + \frac{\sigma_m^2}{\sigma_p^2 + \sigma_m^2} (x_p - x_m)$$

$$\mu = x_m + K (x_p - x_m)$$

$$\sigma(\mu)^2 = \frac{\sigma_p^2 \sigma_m^2 + \sigma_m^4 - \sigma_m^4}{\sigma_p^2 + \sigma_m^2} = \sigma_m^2 - \frac{\sigma_m^4}{\sigma_p^2 + \sigma_m^2} = \sigma_m^2 (1 - K)$$

In ALICE

$$\mu_y = \frac{y_p \sigma_m^2 + y_m \sigma_p^2}{\sigma_p^2 + \sigma_m^2}$$

$$\mu_z = \frac{z_p \sigma_m^2 + z_m \sigma_p^2}{\sigma_p^2 + \sigma_m^2}$$

y_p, z_p from the track model

y_m, z_m from the measurement

K- Kalman gain factor

Measurement is corrected by K Factor

Track Fitting Method

Two things required: Extrapolation with MS and Measured Points

Common Steps:

- ✓ Track Model initialize with the last point $x + \text{Margin}$ (0.1)
- ✓ Convert last point to local coordinate system rotated by ϕ .
- ✓ Extrapolate track to the last layer
- ✓ Rotate Track to local coordinate system
- ✓ Update extrapolation and measurement (weight average of position and errors)
- ✓ Multiple scattering correction

Repeat above steps for each layer up to layer 1 and then extrapolate to the vertex

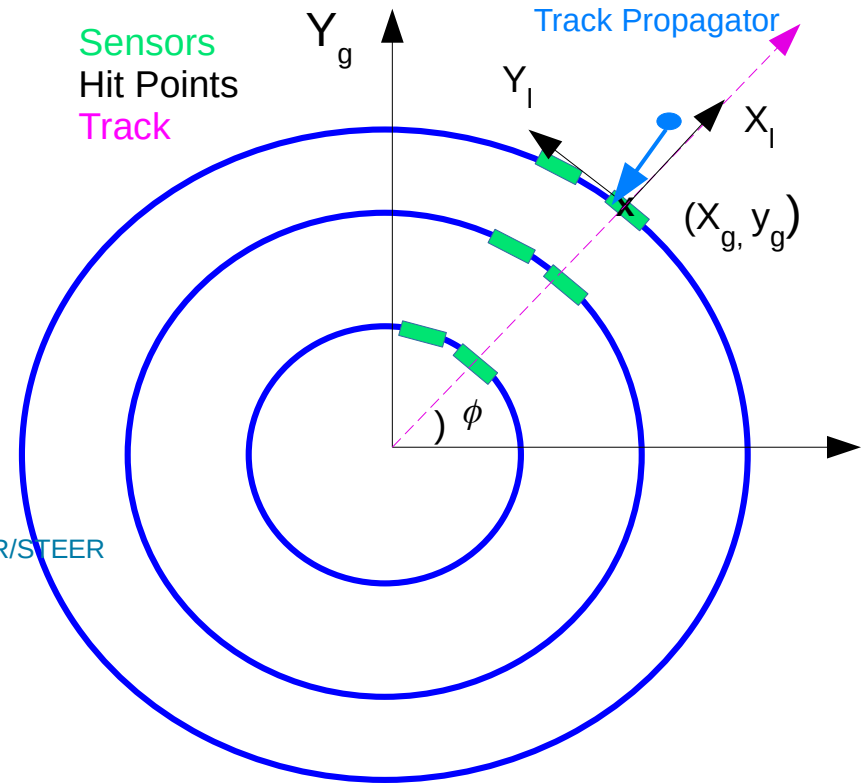
<https://github.com/alisw/AliRoot/blob/master/STEER/STEERBase/AliExternalTrackParam.h>

Track Extrapolation

```
Bool_t PropagateTo(Double_t x, Double_t b);
```

```
Bool_t CorrectForMeanMaterialdEdx(Double_t xOverX0Si, Double_t 0,  
Double_t mPion, Bool_t anglecorr=kTRUE);
```

At the vertex $x_1 = 0$ and y_1 is DCA_{xy}

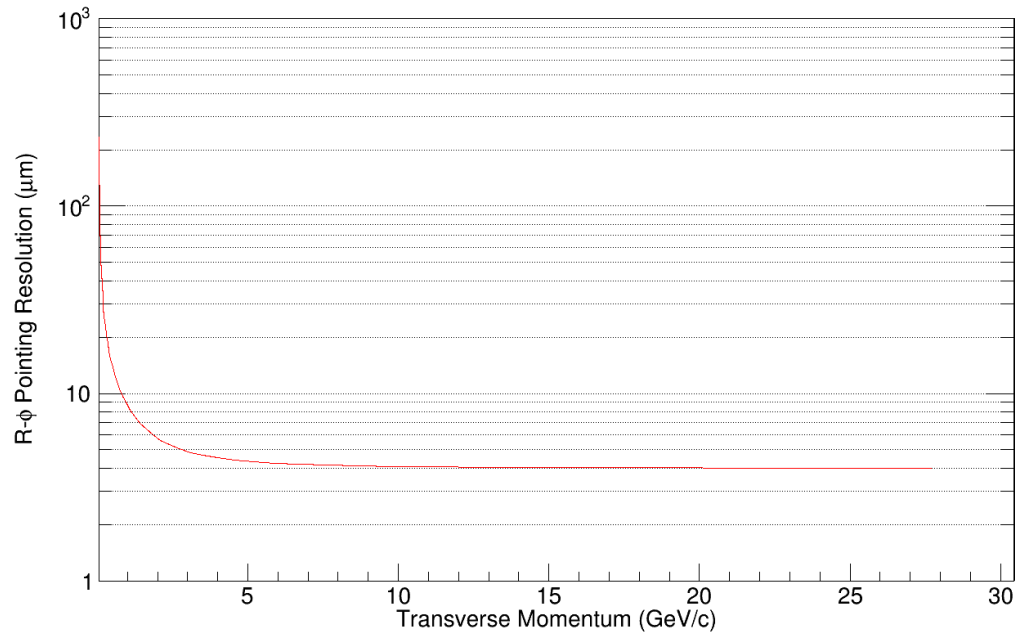


Fast Simulation (Expected DCAxy)

Using Kalman filter

```
Detector BeamTest: "Detector"
Name      r [cm]      X0      phi & z res [um] layerEff
0. vertex  0.00      0.0000  -      -      -
1. VTX0    1.80      0.0005  5      5      1.00
2. VTX1    2.40      0.0005  5      5      1.00
3. VTX2    3.00      0.0005  5      5      1.00
4. VTX3    5.00      0.0005  5      5      1.00
5. VTX4    7.50      0.0005  5      5      1.00
6. VTX5    10.00     0.0005  5      5      1.00
```

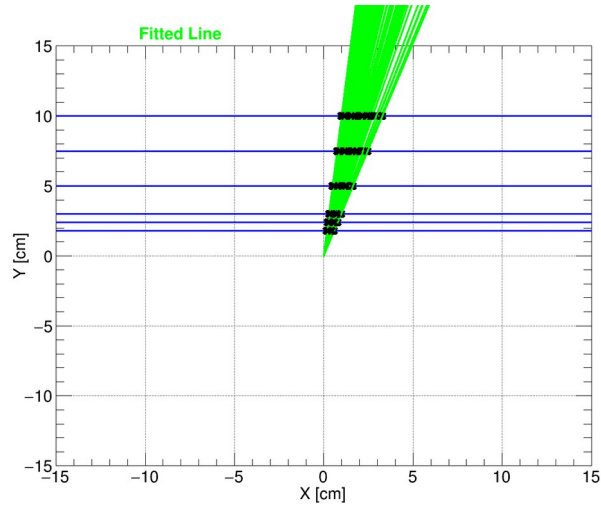
R- ϕ Pointing Resolution .vs. Pt



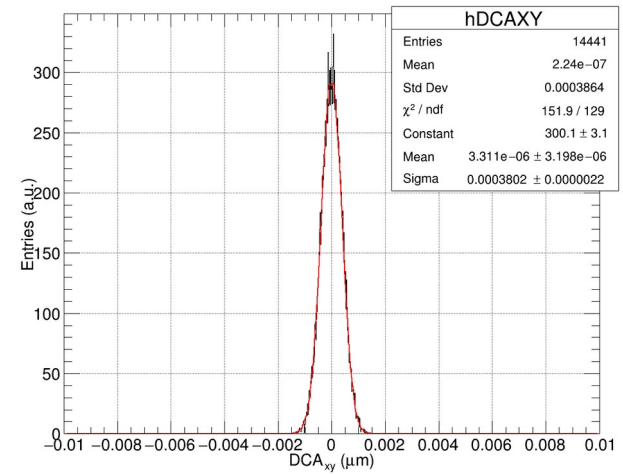
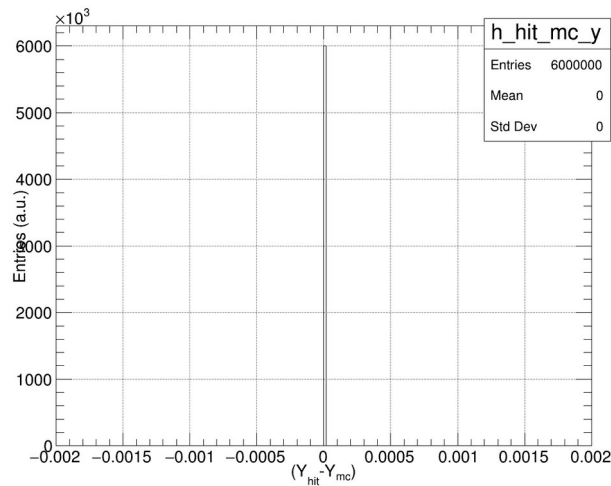
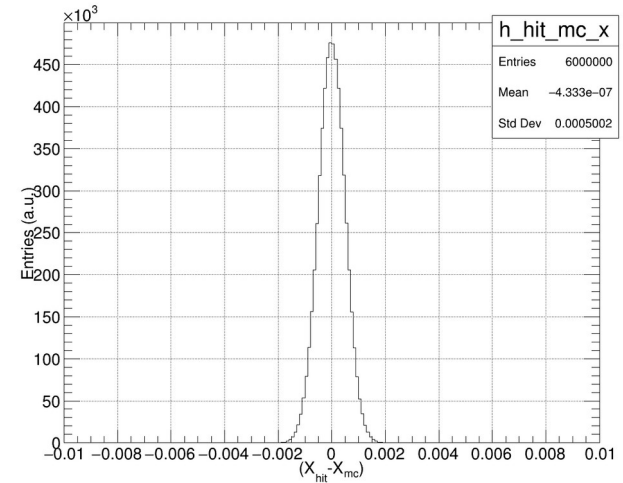
It comes 5 μm if we use equal distance

Fast Simulation (Expected DCAxy)

Chi2 Minimization (Ignoring Multiple Scattering)



$$y = a_0 + a_1 x$$
$$d = \frac{a_0}{\sqrt{1 + a_1^2}}$$



Detector Simulation

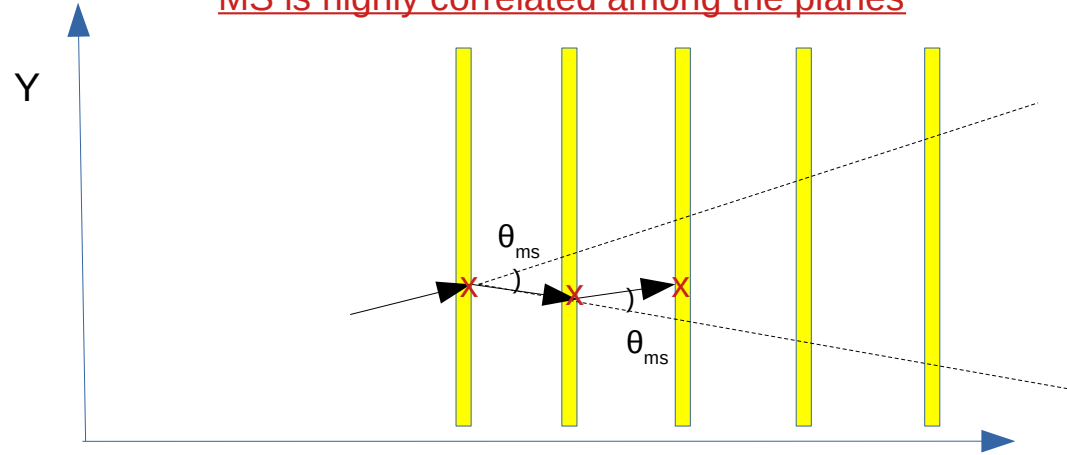
Simulation: Red Points we get during detector simulation with Multiple scattering

Digitization: Smear these red point by pixel resolution (spatial resolution)

Reconstruction: Fit the point after digitization

In the Beam test data, the point are already include both effects so we can directly fit the points

MS is highly correlated among the planes



$$C_{ij} = \langle (y_{hit} - y_{true})_i * (y_{hit} - y_{true})_j \rangle$$

Matrix by me statistically (equal spacing)

6x6 matrix is as follows

	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2.001	3	3.999
2	0	2.001	5.003	8.001	11
3	0	3	8.001	14	19.99
4	0	3.999	11	19.99	29.98
5	0	4.997	13.99	25.98	39.97

	5
0	0
1	4.997
2	13.99
3	25.98
4	39.97
5	54.95

Matrix by Werner Reigler (equal spacing)

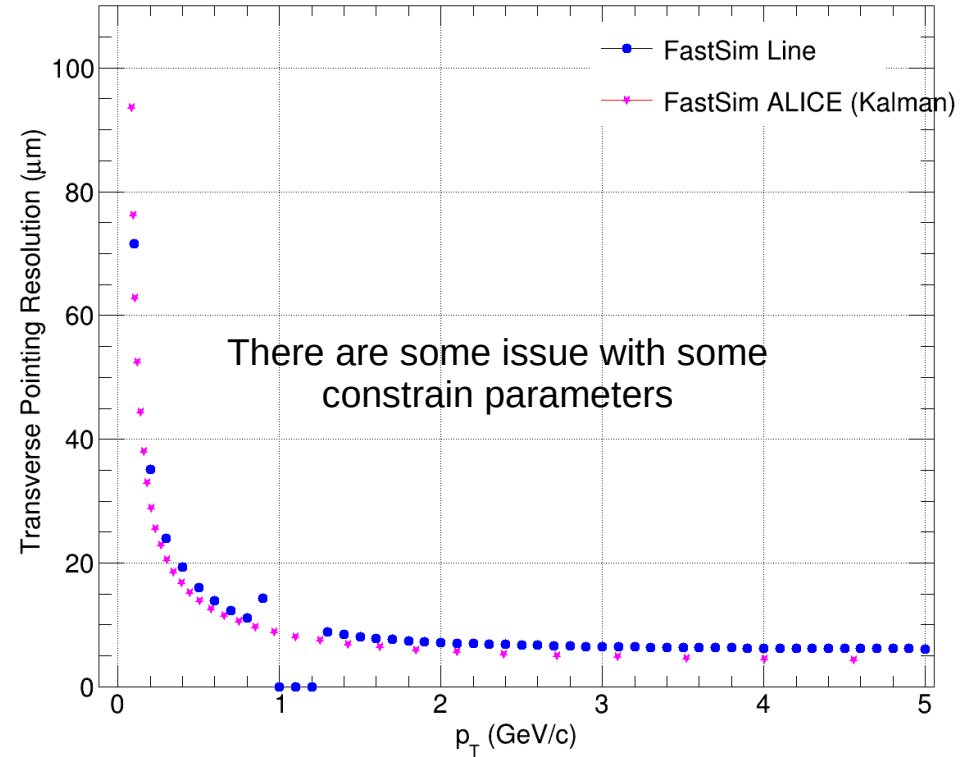
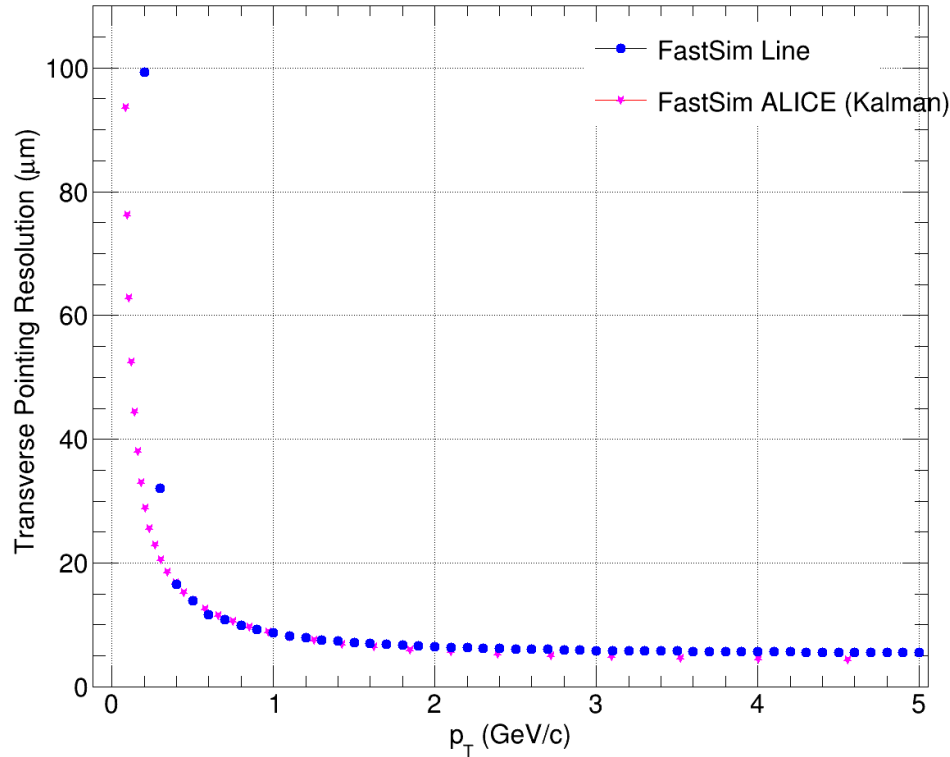
$$C_y = M = \frac{\sigma_\alpha^2 L^2}{N^2} \begin{pmatrix} \frac{N^2 \sigma_0^2}{\sigma_\alpha^2 L^2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & \dots \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & \dots & \dots \\ 0 & 2 & 5 & 8 & 11 & 14 & 17 & 20 & \dots & \dots \\ 0 & 3 & 8 & 14 & 20 & 26 & 32 & 38 & \dots & \dots \\ 0 & 4 & 11 & 20 & 30 & 40 & 50 & 60 & \dots & \dots \\ 0 & 5 & 14 & 26 & 40 & 55 & 70 & 85 & \dots & \dots \\ 0 & 6 & 17 & 32 & 50 & 70 & 91 & 112 & \dots & \dots \\ 0 & 7 & 20 & 38 & 60 & 85 & 112 & 140 & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \ddots \end{pmatrix}$$

Global Chi2 fitting with MS (Line Fit)

$$\chi^2 = \sum_{i=0}^N \frac{(y_{meas} - y_{fit})^2}{\sigma_i^2}$$

Minimization of Full Covariance

$$\chi^2 = \sum_{i=0}^N \sum_{j=0}^N (y_i - y_{fit_i}) * (w_{ij}) * (y_j - y_{fit_j})$$

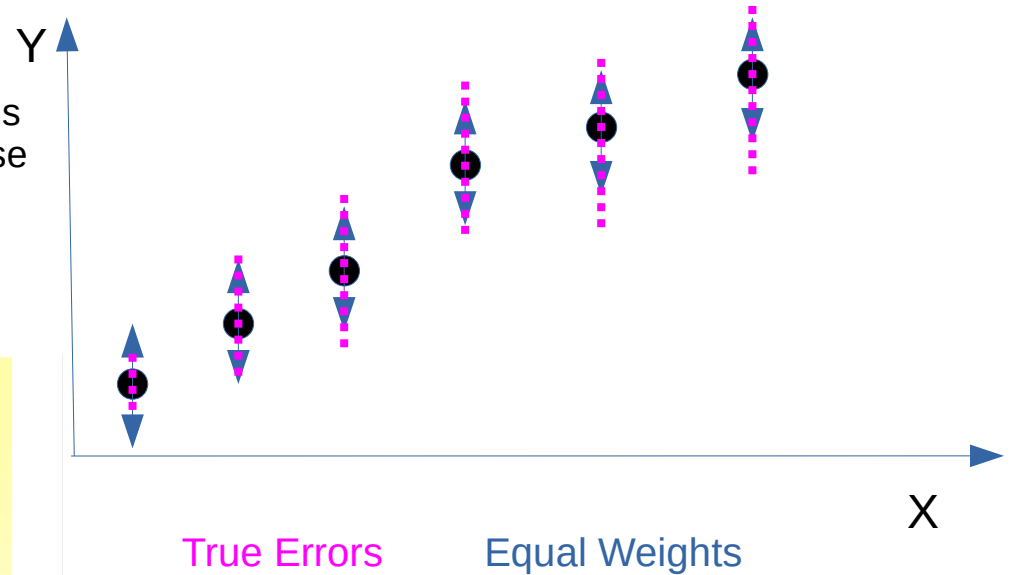


Chi2 minimization gives same results but Chi2 is overestimated (not reliable)

Global Chi2 fitting with MS (Line Fit)

The previous case if we use only spatial resolution means we are giving the equal weights to each points in this case

In reality points have different errors so but the effect on parameters are small: Chi2 is overestimated

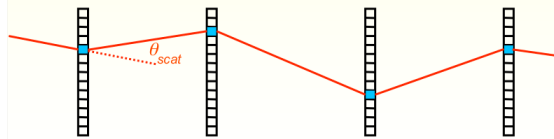
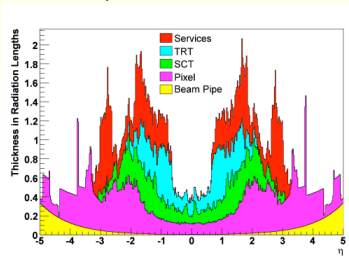


P 10 Scattering Angle Formulation

- The scattering angles and the energy losses become additional fit parameters, along with the track parameters at the vertex (i.e. impact parameter, direction and momentum).
- The χ^2 function to be minimized w.r.t. the fit parameters is:

$$\chi^2 = \sum_{hits} \frac{\Delta y^2}{\sigma_{hit}^2} + \sum_{scatters} \frac{\theta_{scat}^2}{\sigma_{scat}^2} + \sum_{Eloss} \frac{(\Delta E - \overline{\Delta E})^2}{\sigma_{Eloss}^2}$$

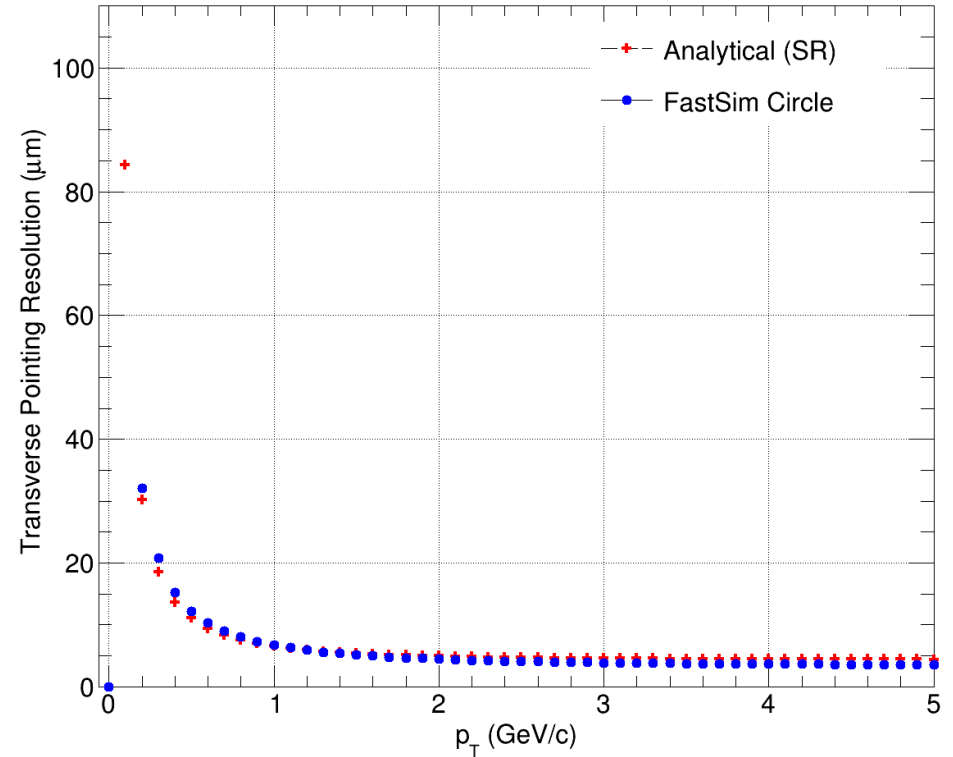
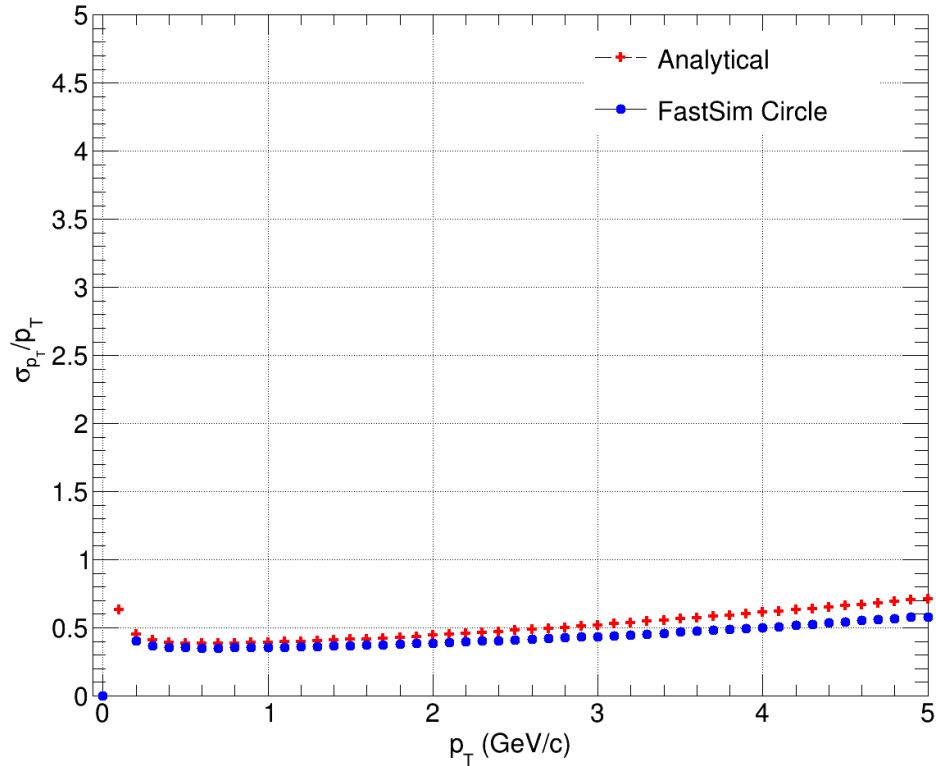
where Δy are the residuals, and σ_{scat} and $\overline{\Delta E}$ can be estimated from the material properties (done by the [MultipleScatteringUpdater](#) and [EnergyLossUpdater](#) tools)



CHEP07 conference 5 September 2007, T. Cornelissen

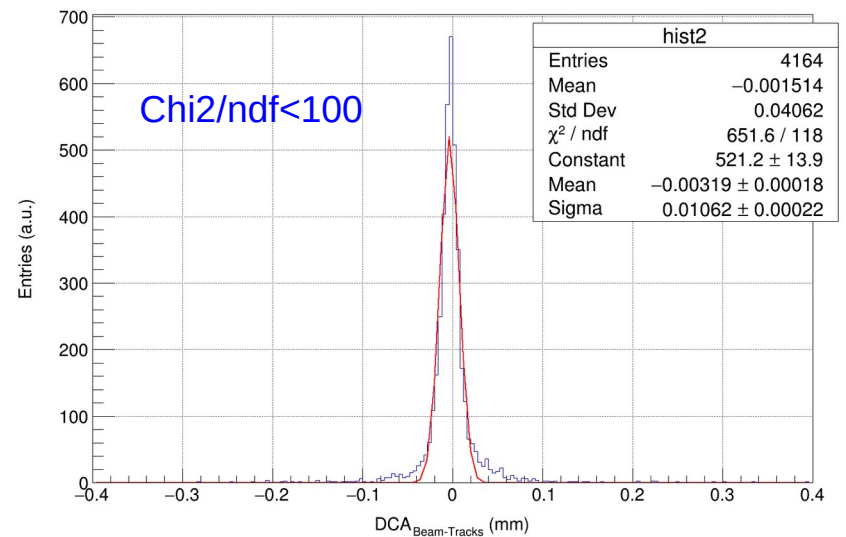
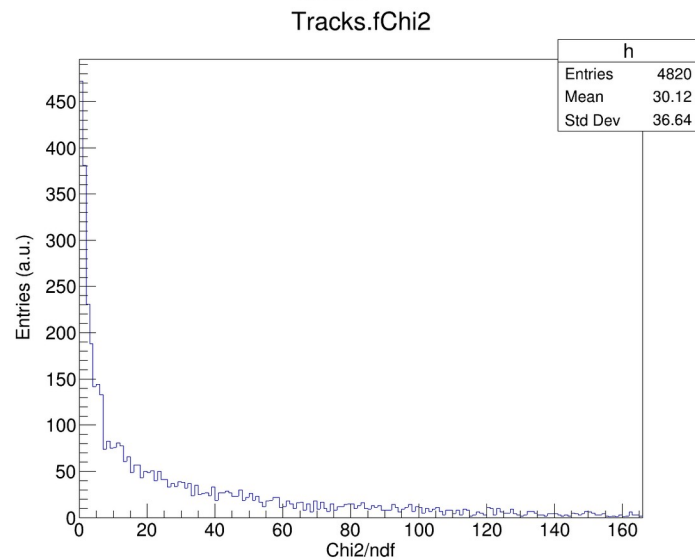
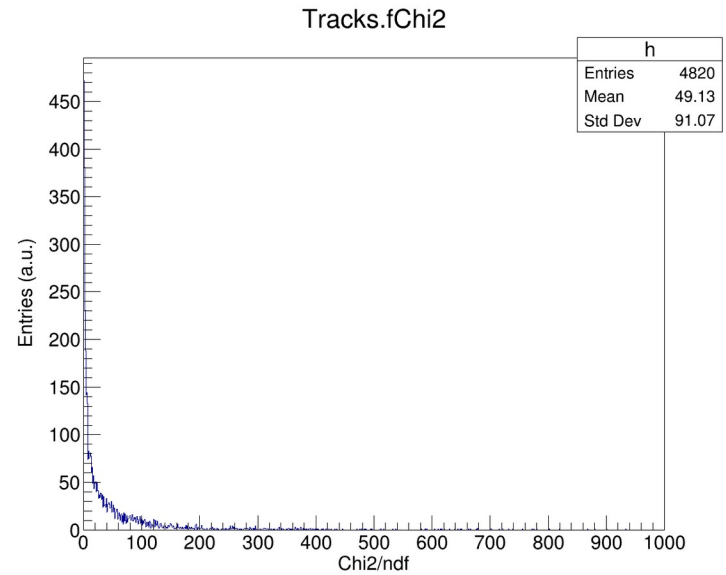
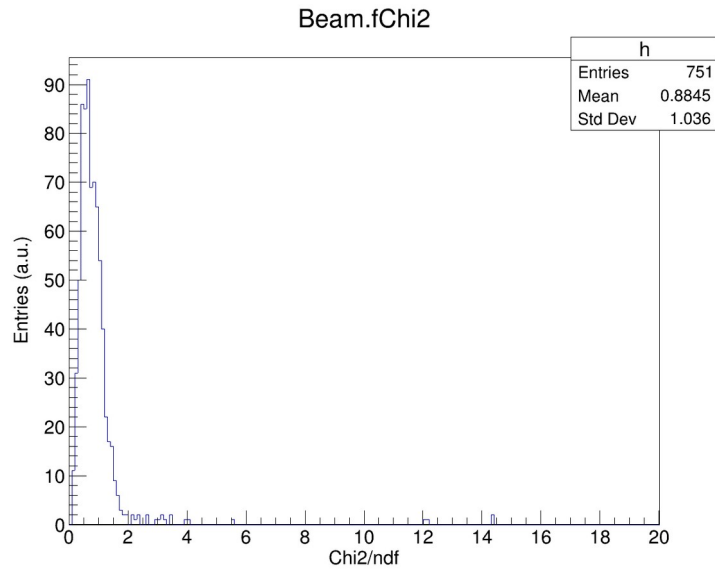
Global Chi2 fitting with MS (Circle Fit)

$$\chi^2 = \sum_{i=0}^N \frac{(x_{meas} - x_{fit})^2}{\sigma_{x_i}^2} + \frac{(y_{meas} - y_{fit})^2}{\sigma_{y_i}^2}$$

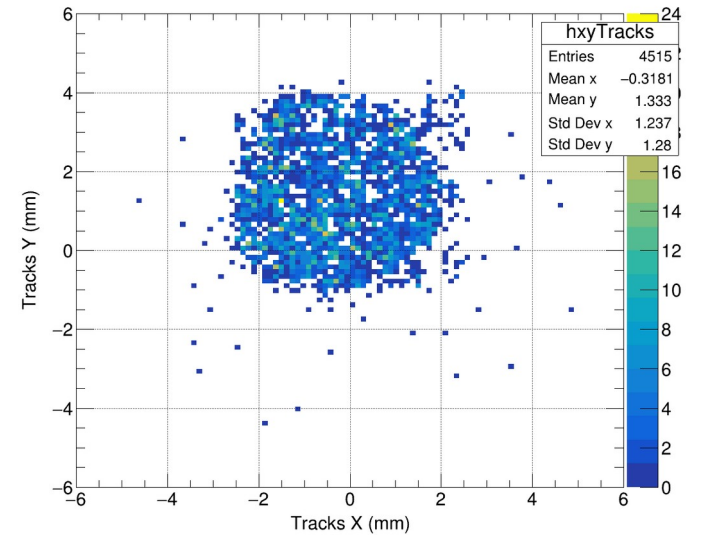
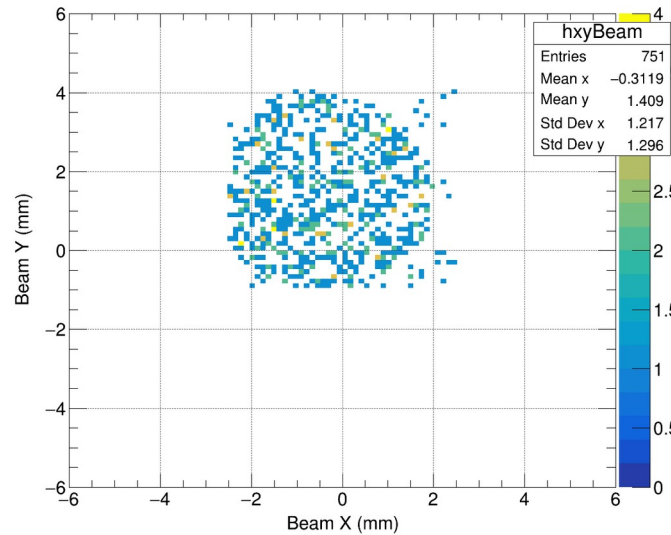
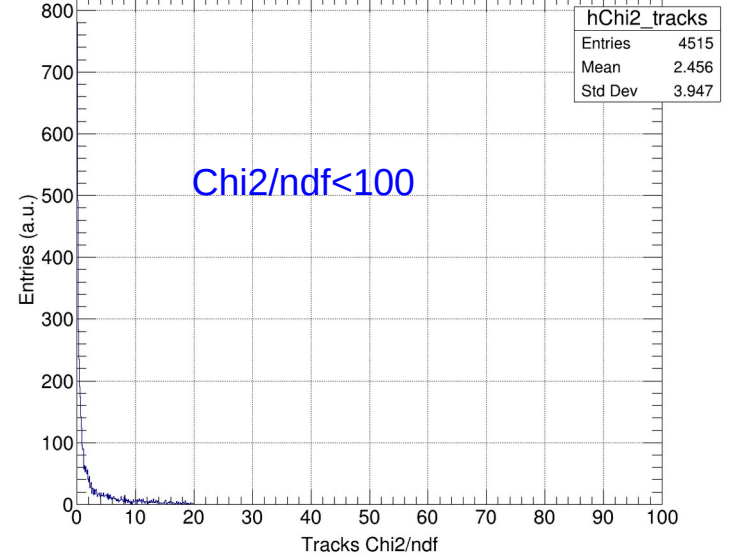
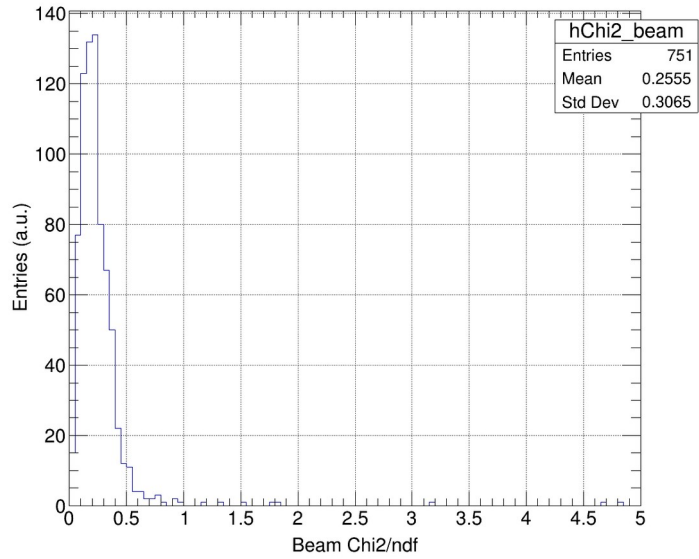


Case of equal spacing of planes where formulas can be used soon will apply for EIC Tracker

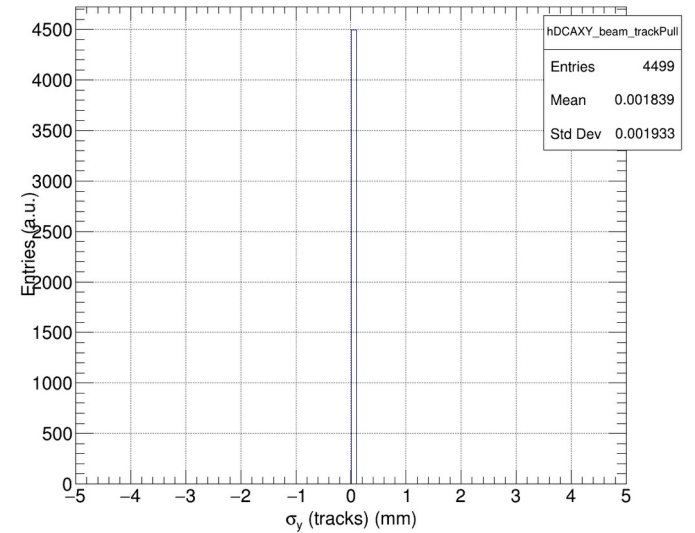
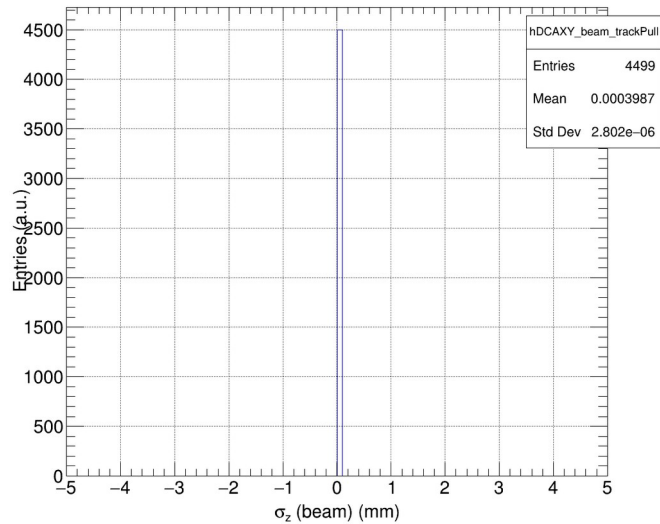
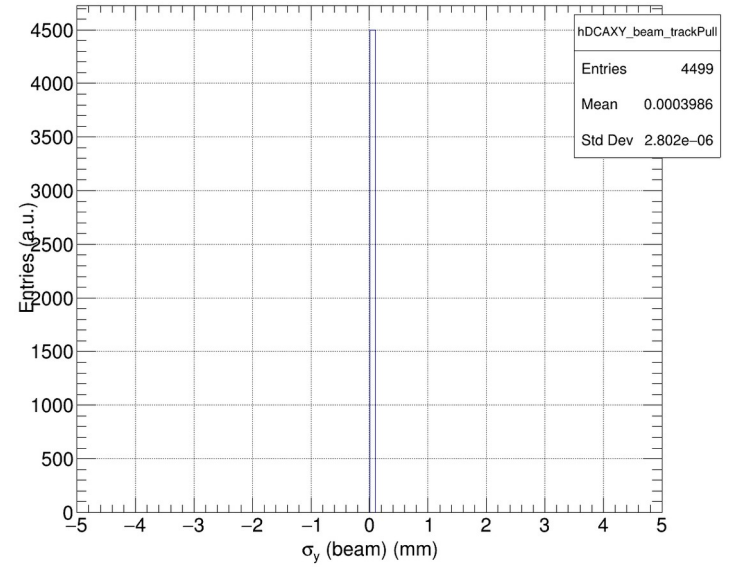
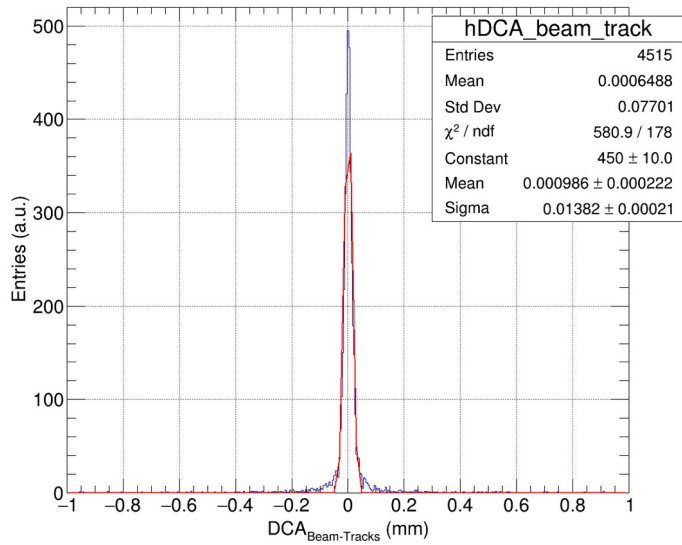
Results (Global Chi2 fitting)



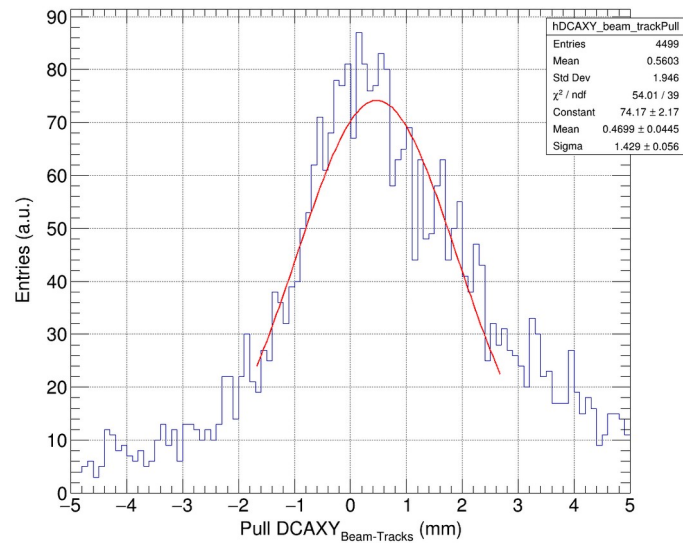
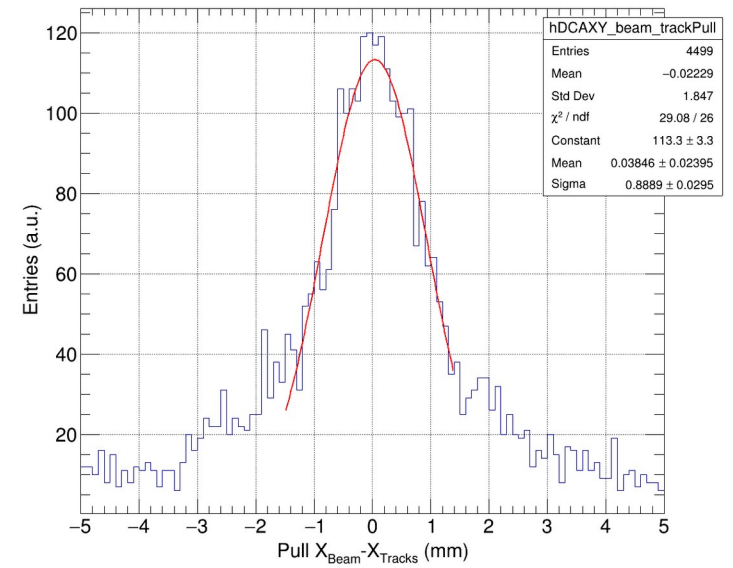
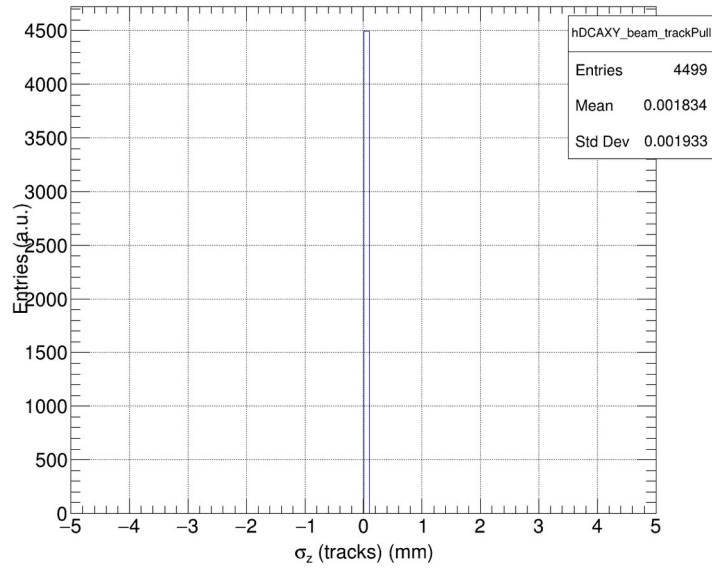
Results (Kalman Filter)



Results (Kalman Filter)

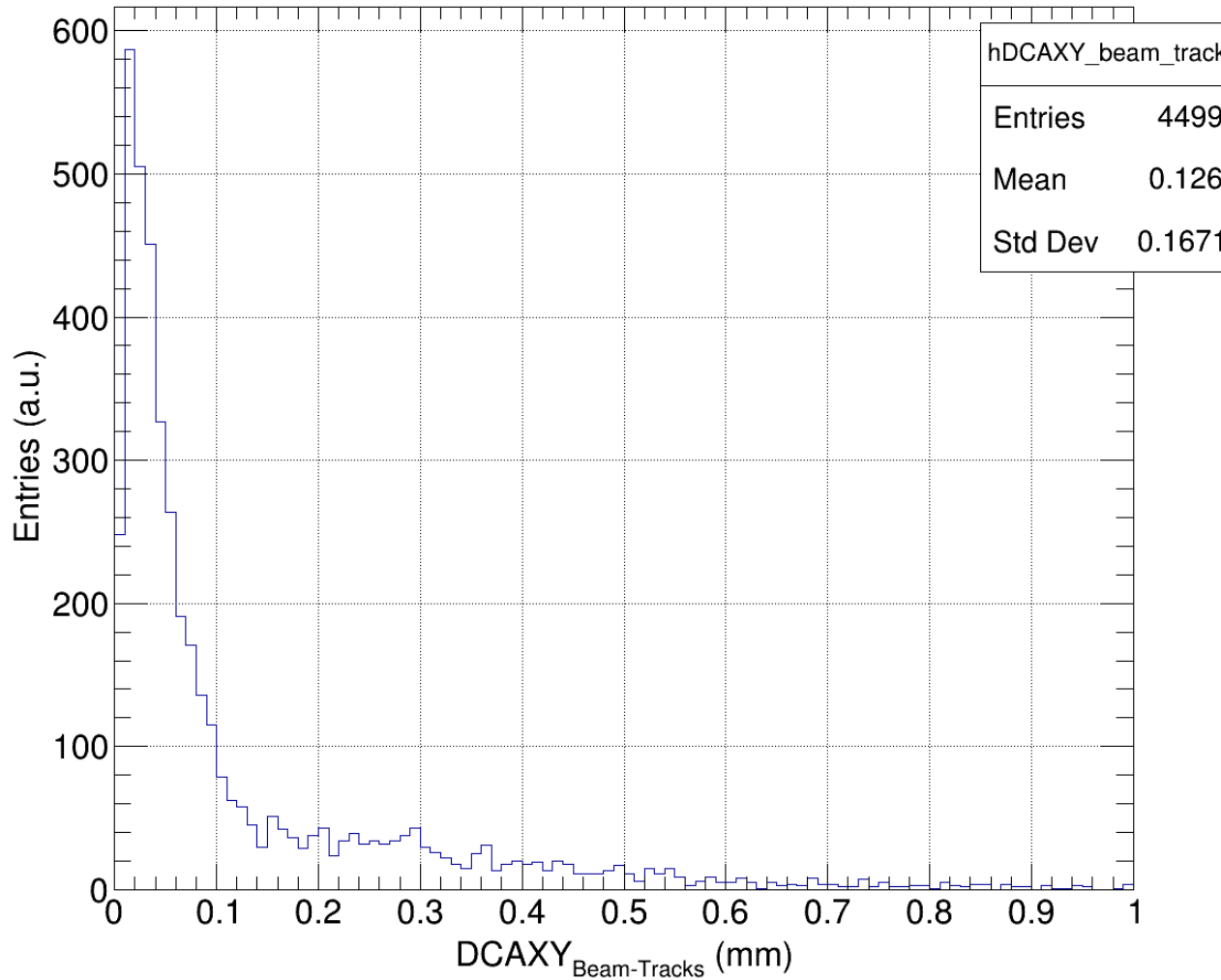


Results (Kalman Filter)

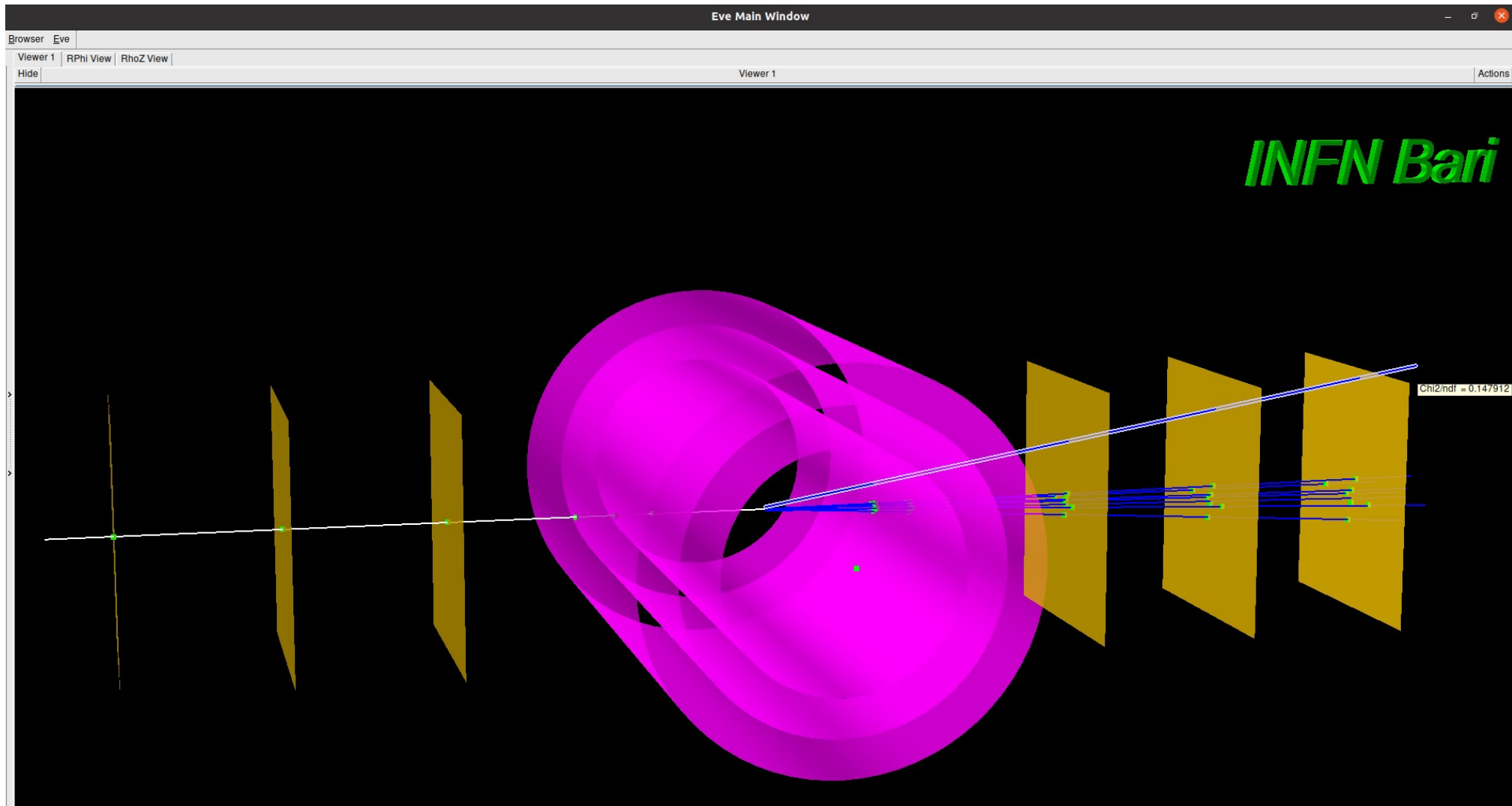


➤ Pull distributions for X and DCAXy are shown

DCAxy beam to tracks



Event display based on Kalman filter



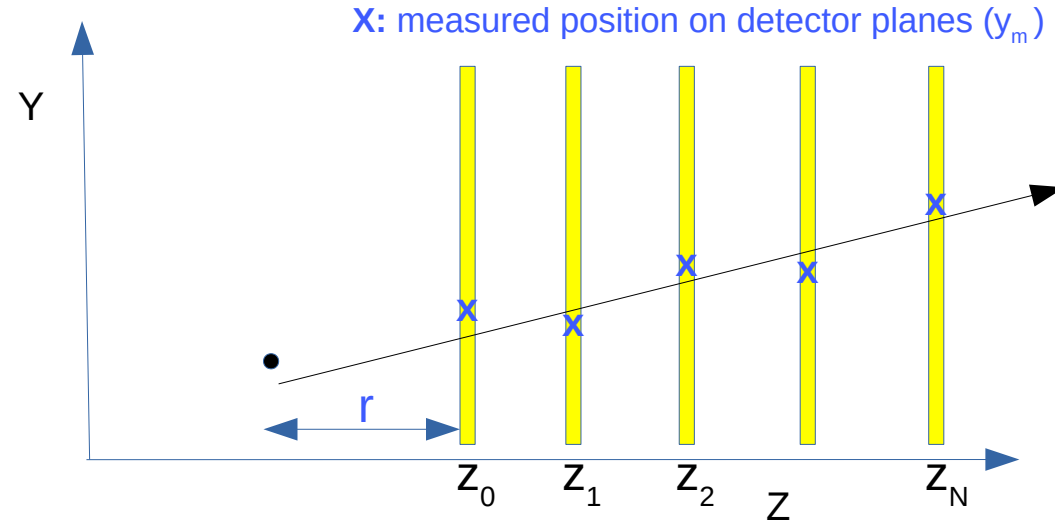
Straight Line Fit

General line

$$y = a + bz$$

$$\chi^2 = \sum_{i=0}^N \frac{(y_m - y_i)^2}{\sigma_i^2} = \sum_{i=0}^N \frac{(y_m - a - bz_i)^2}{\sigma_i^2}$$

$$\partial \chi^2 / \delta a = 0 \quad \partial \chi^2 / \delta b = 0$$



Chi2 minimization: returns best a, b and $\sigma_a, \sigma_b, \sigma_{ab}, \sigma_{ba}$

$$S_1 = \sum_{i=0}^N \frac{1}{\sigma_i^2}$$

$$S_y = \sum_{i=0}^N \frac{y_i}{\sigma_i^2}$$

$$S_z = \sum_{i=0}^N \frac{z_i}{\sigma_i^2}$$

$$S_{yz} = \sum_{i=0}^N \frac{y_i z_i}{\sigma_i^2}$$

$$S_{zz} = \sum_{i=0}^N \frac{z_i^2}{\sigma_i^2}$$

$$\Delta = S_1 S_{zz} - S_z^2$$

$$\sigma_a^2 = S_{zz} / \Delta$$

$$a = (S_y S_{zz} - S_z S_{zy}) / \Delta$$

$$\sigma_b^2 = S_1 / \Delta$$

$$b = (S_1 S_{zy} - S_z S_y) / \Delta$$

$$\sigma_{ab} = \sigma_{ba} = -S_z / \Delta$$

Rewrite



Covariance Matrix of Parameters

$$\begin{pmatrix} \sigma_a^2 & \sigma_{ab} \\ \sigma_{ba} & \sigma_b^2 \end{pmatrix} = \frac{1}{\Delta} \begin{pmatrix} S_{zz} & -S_z \\ -S_z & S_1 \end{pmatrix}$$

<http://www.le.infn.it/lhcschool/talks/Ragusa.pdf>
http://www.foo.be/docs-free/Numerical_Recipe_In_C/c15-2.pdf

Matrix Form

Convention in Matrix Form (Compact)

Matrix of Errors

$$y = (1 \quad z) \begin{pmatrix} a \\ b \end{pmatrix}$$

$$y = A p$$

p: parameter matrix

$$V = \begin{pmatrix} \sigma_0^2 & \dots \\ 0 & \sigma_1^2 \dots \\ \dots & \dots \\ 0 & 0 \dots \sigma_N^2 \end{pmatrix}$$

$$W = V^{-1} = \begin{pmatrix} 1/\sigma_0^2 & \dots \\ 0 & 1/\sigma_1^2 \dots \\ \dots & \dots \\ 0 & 0 \dots 1/\sigma_N^2 \end{pmatrix}$$

$$y = A p$$

Chi2 minimization: Best parameters and Parameter Covariance

$$\tilde{p} = (A^T W A)^{-1} A^T W Y$$

$$C_{\tilde{p}} = (A^T W A)^{-1} = (A^T V^{-1} A)^{-1}$$

Tracking is all about finding track parameters and parameter covariance for any track model

$$\chi^2 = (Y - A p)^T W (Y - A p)$$

Since track is straight, therefore momentum can't be determined

Tracking Performances (DCA Resolution)

Spatial Smearing

$$\chi^2 = (Y - Ap)^T W_{SR} (Y - Ap)$$

$$\tilde{p} = (A^T W_{SR} A)^{-1} A^T W Y$$

$$C_{\tilde{p}} = (A^T W_{SR} A)^{-1} = (A^T V_{SR}^{-1} A)^{-1}$$

$$V_{SR} = \begin{pmatrix} \sigma_{SR0}^2 & \dots & \dots & \dots \\ 0 & \sigma_{SR1}^2 & \dots & \dots \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \sigma_{SRN}^2 \end{pmatrix}$$

Parameters covariance is used for the extrapolation toward the vertex

Multiple Scattering

$$\chi^2 = (Y - Ap)^T W_{MS} (Y - Ap)$$

$$\tilde{p} = (A^T W_{MS} A)^{-1} A^T W Y$$

$$C_{\tilde{p}} = (A^T W_{MS} A)^{-1} = (A^T V_{MS}^{-1} A)^{-1}$$

$$V_{MS} = \begin{pmatrix} \sigma_{MS0}^2 & \dots & \dots & \dots \\ 0 & \sigma_{MS1}^2 & \dots & \dots \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \sigma_{MSN}^2 \end{pmatrix}$$

Extrapolation of line at vertex (MS)

$$\sigma_y^2(MS) = \sigma_a^2 + r^2 \sigma_b^2 + \sigma_{ab} r + \sigma_{ba} r$$

$$\sigma_y^2(MS) = \sigma_a^2 + r^2 \sigma_b^2 + 2 \sigma_{ab} r$$

Extrapolation of line at vertex (SR)

$$\sigma_y^2(SR) = \sigma_a^2 + r^2 \sigma_b^2 + \sigma_{ab} r + \sigma_{ba} r$$

$$\sigma_y^2(SR) = \sigma_a^2 + r^2 \sigma_b^2 + 2 \sigma_{ab} r$$

$$\sigma_{DCA} = \sqrt{\sigma_y^2(SR) + \sigma_y^2(MS)}$$

The main issue is $\sigma_{MS0} = 0$ (V_{MS}) that makes matrix singular and inverse doesn't exist (How to handle?)

Parabolic Fit

A track follows helical trajectory which is a circle in XY plane and straight line in R-Z plane

General equation of a circle

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

$$y = y_0 + \sqrt{R^2 - (x - x_0)^2}$$

$$y = y_0 + R \sqrt{1 - \frac{(x - x_0)^2}{R^2}}$$

$$y = y_0 + R \left(1 - \frac{(x - x_0)^2}{2R^2} \right)$$

$$p_T = 0.3 B R$$

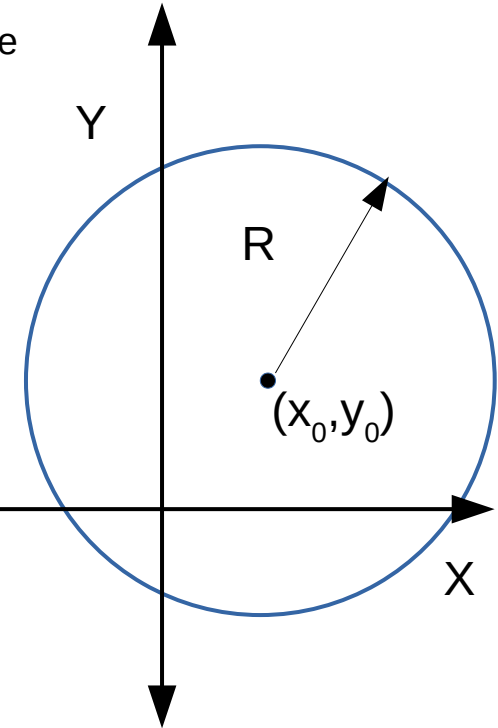
$$\frac{\sigma_{p_T}}{p_T} = \frac{\sigma_R}{R} = \frac{\sigma_c}{c}$$

$$\text{if } (x - x_0) \ll R$$

$$y = y_0 + R - \frac{x_0^2}{2R^2} + x \frac{x_0}{R} - \frac{x^2}{2R}$$

$$y = a + b x + c x^2$$

$$c = (1/2 R)$$



If first layer at $x = r$, then uncertainty in extrapolation:

$$\sigma_y^2(SR) = \sigma_a^2 + r^2 \sigma_b^2 + r^4 \sigma_c^2 + 2 \sigma_{ab} r + 2 \sigma_{bc} r^3 + 2 \sigma_{ac} r^2$$

$$\sigma_y^2(MS) = \sigma_a^2 + r^2 \sigma_b^2 + r^4 \sigma_c^2 + 2 \sigma_{ab} r + 2 \sigma_{bc} r^3 + 2 \sigma_{ac} r^2$$

$$\sigma_{DCA_{xy}} = \sqrt{\sigma_y^2(SR) + \sigma_y^2(MS)}$$

Fit the parabola then you can extract everything

Parabolic Fit (Same analogy to Straight line)

$$\chi^2 = \sum_{i=0}^N \frac{(y_m - y_i)^2}{\sigma_i^2} = \sum_{i=0}^N \frac{(y_m - a - bx_i - cx_i^2)^2}{\sigma_i^2}$$

$$y = A p$$

$$\begin{pmatrix} y_0 \\ y_1 \\ \dots \\ y_N \end{pmatrix} = \begin{pmatrix} 1 & x_0 & x_0^2 \\ 1 & x_1 & x_1^2 \\ \dots & \dots & \dots \\ 1 & x_N & x_N^2 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

Chi2 minimization: Best parameters and Parameter Covariance

$$\chi^2 = (Y - Ap)^T W (Y - Ap)$$

$$\tilde{p} = (A^T W A)^{-1} A^T W Y$$

$$C_{\tilde{p}} = (A^T W A)^{-1} = (A^T V^{-1} A)^{-1}$$

Once we have parameters and covariance: p_T and DCAxy resolution can be extracted

Spatial Smearing

$$\chi^2 = (Y - Ap)^T W_{SR} (Y - Ap)$$

$$\tilde{p} = (A^T W_{SR} A)^{-1} A^T W Y$$

$$C_{\tilde{p}} = (A^T W_{SR} A)^{-1} = (A^T V_{SR}^{-1} A)^{-1}$$

$$\frac{\sigma_{p_T}}{p_T} = \frac{\sigma_R}{R} = \frac{\sigma_c}{c}$$

$$\sigma_{DCA_{xy}} = \sqrt{\sigma_y^2 (SR) + \sigma_y^2 (MS)}$$

Multiple Scattering

$$\chi^2 = (Y - Ap)^T W_{MS} (Y - Ap)$$

$$\tilde{p} = (A^T W_{MS} A)^{-1} A^T W Y$$

$$C_{\tilde{p}} = (A^T W_{MS} A)^{-1} = (A^T V_{MS}^{-1} A)^{-1}$$

The main issue is $\sigma_{MS0} = 0$ (V_{MS}) that makes matrix singular and inverse doesn't exist (How to handle?)

Multiple Scattering

Covariance matrix entries affected by multiple scattering

	$1/p$	λ	ϕ	γ_{\perp}	\mathbf{z}_{\perp}
$1/p$	0	0	0	0	0
λ	0	$\langle \theta_p^2 \rangle$	0	0	$-\frac{\langle \theta_p^2 \rangle dl}{2}$
ϕ	0	0	$\frac{\langle \theta_p^2 \rangle}{\cos^2 \lambda}$	$\frac{\langle \theta_p^2 \rangle dl}{(2 \cos \lambda)}$	0
γ_{\perp}	0	0	$\frac{\langle \theta_p^2 \rangle dl}{(2 \cos \lambda)}$	$\frac{\langle \theta_p^2 \rangle (dl)^2}{3}$	0
\mathbf{z}_{\perp}	0	$-\frac{\langle \theta_p^2 \rangle dl}{2}$	0	0	$\frac{\langle \theta_p^2 \rangle (dl)^2}{3}$

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