SoLID Tracking Reconstruction

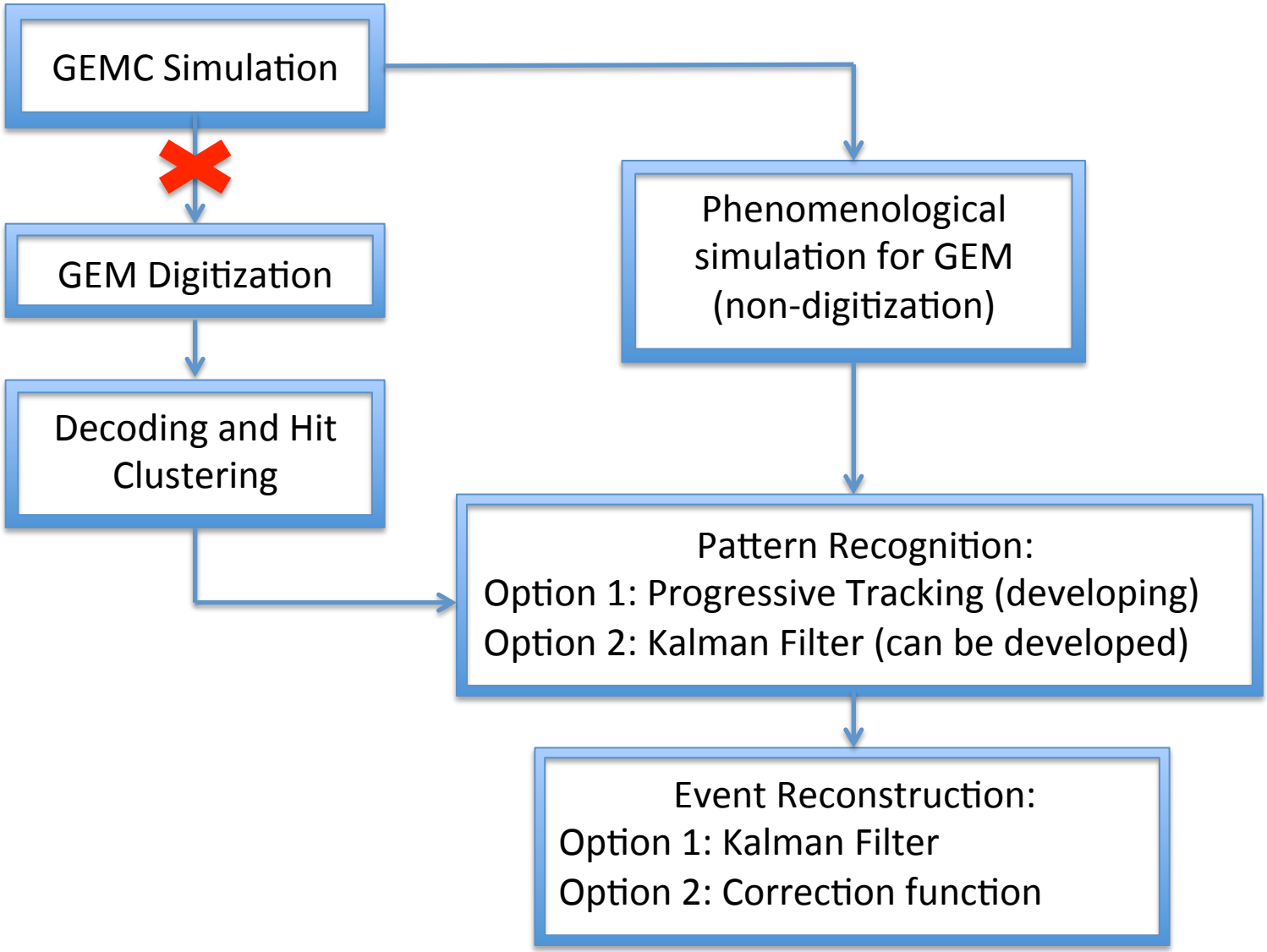
Weizhi Xiong

08/31/2015

Outline

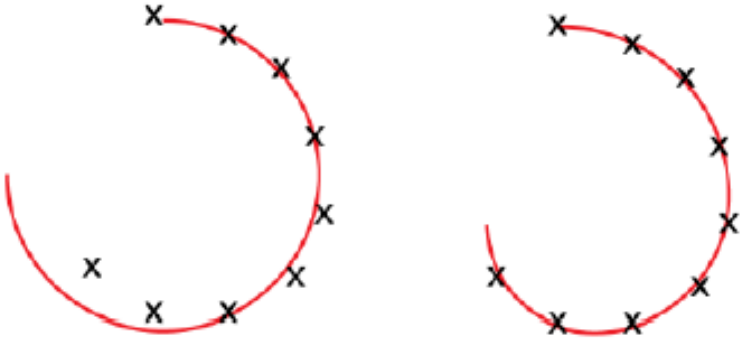
- Introduction to Kalman filter
- Issues with GEMC simulation
- Tracking reconstruction results using Kalman filter
- Conclusion and plan

Tracking Framework Overview



Introduction on Kalman Filter

- Kalman filter vs Least square fit:
  - Both are χ^2 minimizing method
  - Kalman filter is a recursive fitting algorithm
  - Kalman filter allows the parameters (state vector) to change along the trajectory (easy to make correction due to energy loss and inhomogeneous field)



Least Squares Fit

Kalman Filter

Introduction on Kalman Filter

- Basic steps for Kalman Filter[1]:
  - The optimized state vector a_{k-1} on detector k-1 is extrapolated to detector k by means of a propagation method

$$a_k^{k-1} = f_{k-1}(a_{k-1})$$

  - The covariance matrix of the predicted state vector, and also the covariance matrix of the process noise between detector k-1 and k are computed by error propagation

$$C_k^{k-1} = F_{k-1} C_{k-1} F_{k-1}^T + Q_{k-1}$$

$$F_{k-1} = \frac{\partial f_{k-1}(a_{k-1})}{\partial a_{k-1}}$$

Introduction on Kalman Filter

- Basic steps for Kalman Filter:
  - The predicted state vector and its covariance matrix are projected into the measurement space by means of a projector H, thus obtain the predicted measurement vector
  - The weighted mean of the extrapolated and the actual measurement vector m_k of detector k is computed, yielding an optimal estimate of the state vector at k

$$C_k = (I - K_k H_k) C_k^{k-1}$$

$$a_k = a_k^{k-1} + K_k (m_k - H_k a_k^{k-1})$$

$$K_k = C_k^{k-1} (H_k)^T (V_k + H_k C_k^{k-1} (H_k)^T)^{-1}$$

In the simplest case, assume 1d state vector x, and we directly measure this quality, then the Kalman filter formulae can be reduced to a simple form

$$\frac{x_o}{\sigma_o^2} = \frac{x_p}{\sigma_p^2} + \frac{x_m}{\sigma_m^2}$$

Introduction on Kalman Filter

- The state vector used in the current program[2]:
  - $(x, y, t_x, t_y, q/p)$
- The propagation of state vector and its covariance matrix is done based on 4^{th} order classical Runge-Kutta method (similar to RKClassicalRK4 class in Geant4)
- Process noise and corrections such as Coulomb multiple scattering, energy loss due to ionization and Bremsstrahlung can be done step by step along the propagation
- This approach is designed to work for both SIDIS and PVDIS. Changing from one configuration to the other requires nothing but redefining detector locations

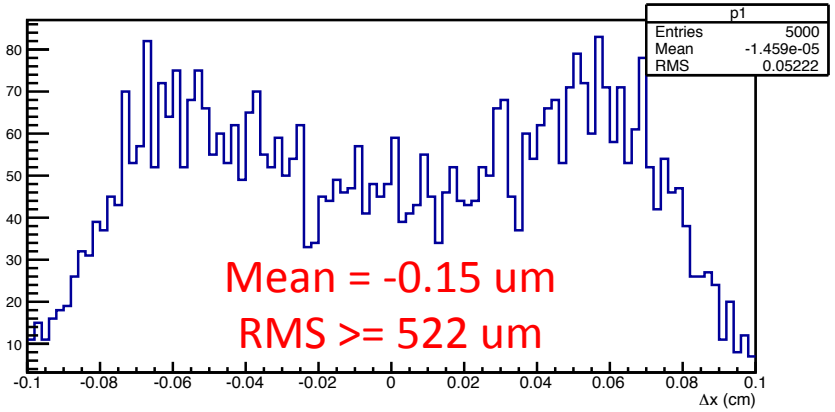
Issues with Current GEMC Simulation

- Biases and large errors are observed if I run the program on GEMC simulation
- Develop two propagation methods in my program (core component the same as RKClassicalRK4 and RKNystromRK4 in Geant4)
  - Using linear interpolation for the field map
  - Fixed step propagation method (step fixed at minimal step used in GEMC)
  - Accurate calculation for the intersection between trajectory and detector surface (propagation step decreases as particle approaching the detector, make sure the particle “land” on the detector surface. This is a time consuming process, but a typical way used in Kalman filter reconstruction algorithm to ensure high accuracy)

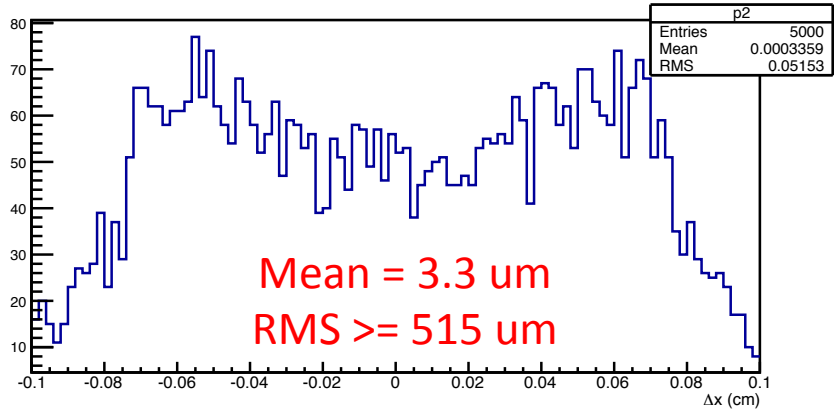
Issues with Current GEMC Simulation

Compare the coordinates of intersection between GEMC and my simulation on the last SIDIS GEM

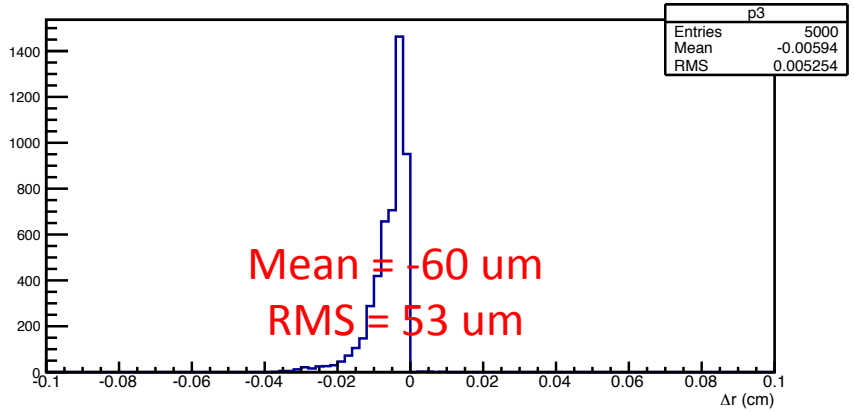
Δx on the last SIDIS GEM



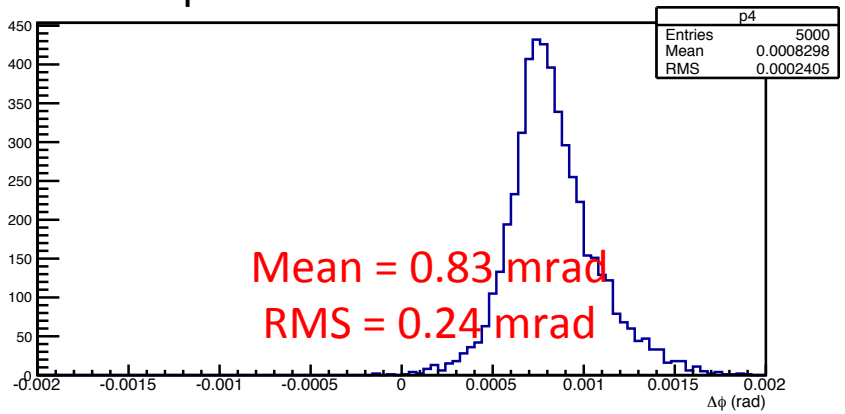
Δy on the last SIDIS GEM



Δr on the last SIDIS GEM



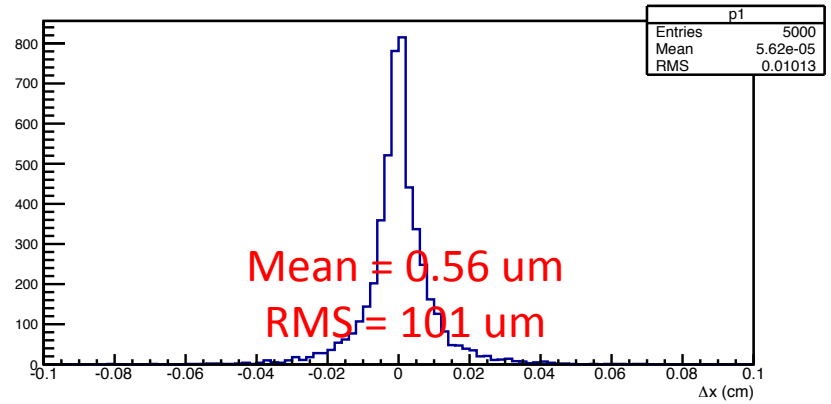
Δφ on the last SIDIS GEM



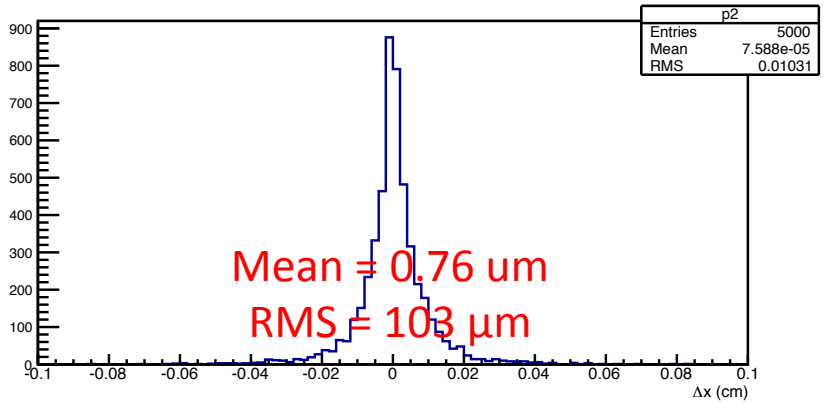
Issues with Current GEMC Simulation

DeltaIntersection (1mm) -> DeltaIntersection(0.001mm)
DeltaOneStep(1mm) -> DeltaOneStep(0.01mm)
G4CachedMagField(12cm)->G4CacheMagField(1cm)

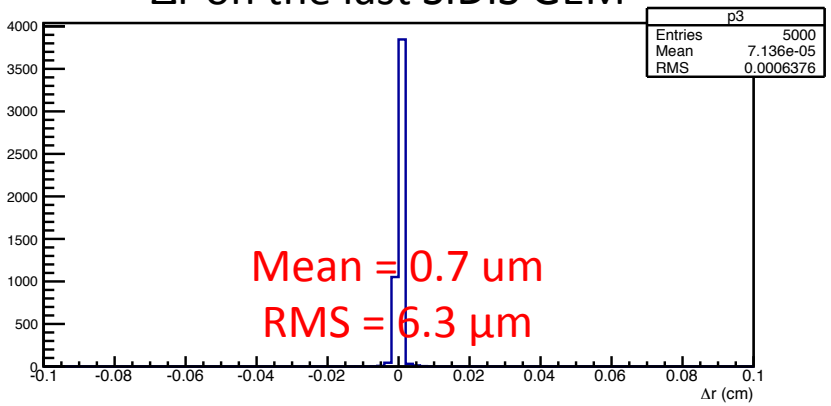
Δx on the last SIDIS GEM



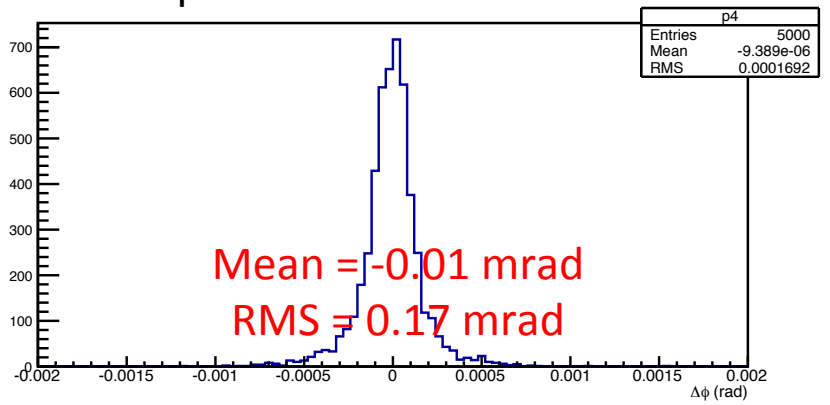
Δy on the last SIDIS GEM



Δr on the last SIDIS GEM



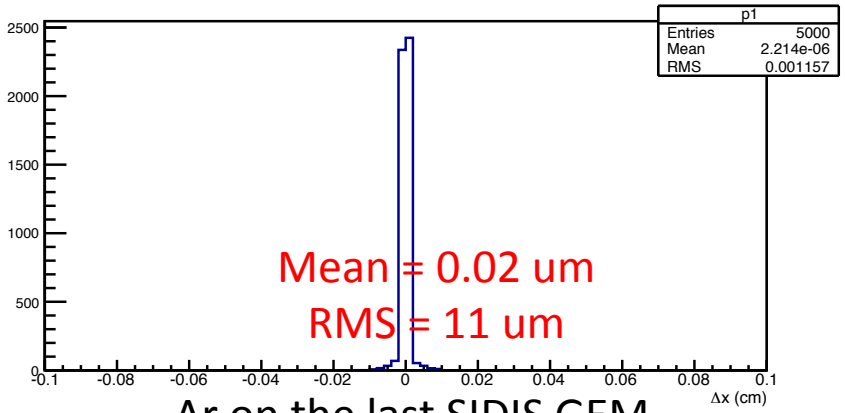
Δφ on the last SIDIS GEM



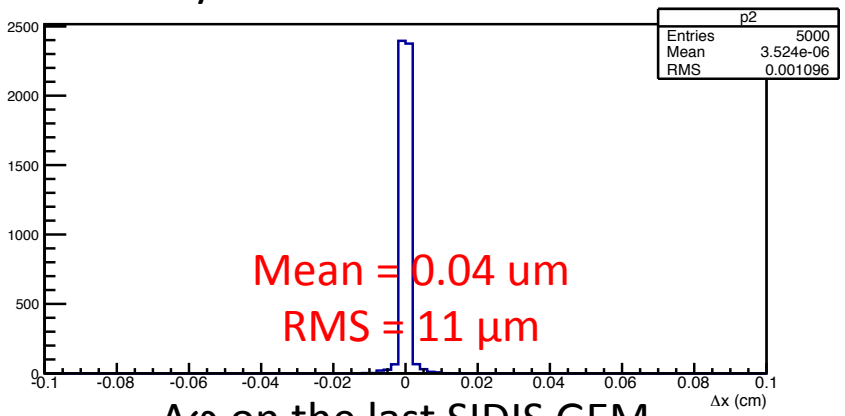
Issues with Current GEMC Simulation

Using in addition, linear interpolation of the field map

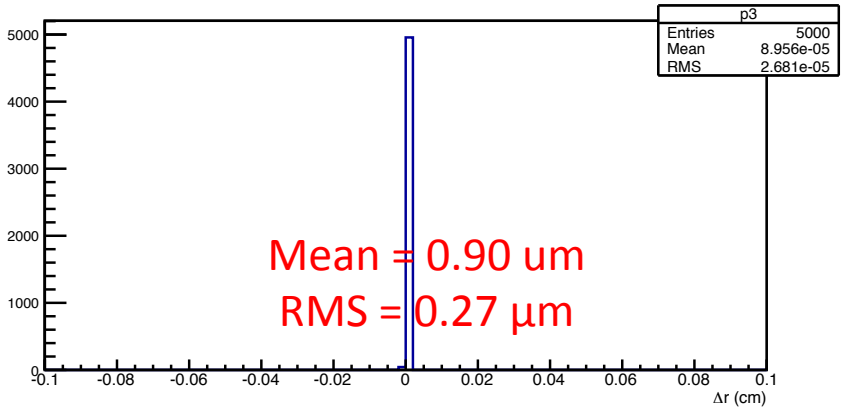
Δx on the last SIDIS GEM



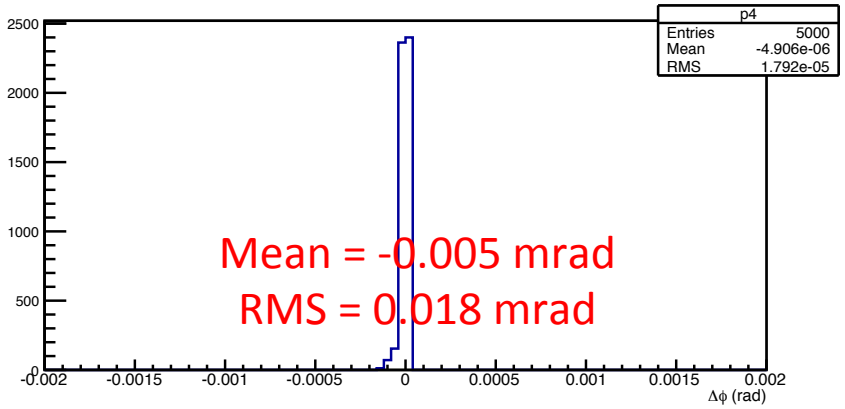
Δy on the last SIDIS GEM



Δr on the last SIDIS GEM



Δφ on the last SIDIS GEM

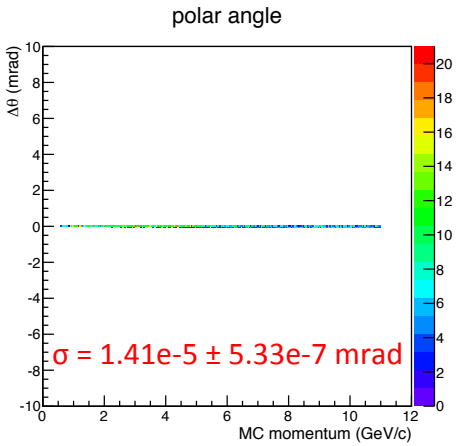


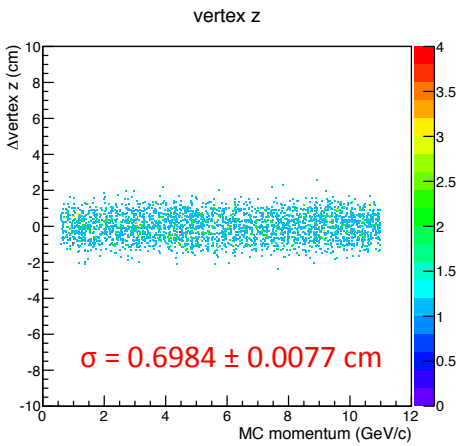
Vertex Reconstruction General Info

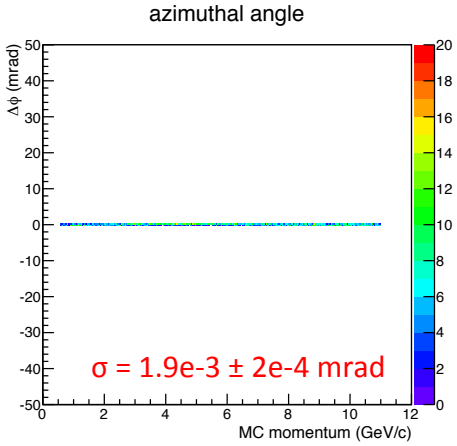
- Initializing the Kalman filter:
  - For SIDIS, choose three hits to compute the initial helix, assuming uniform field, helical parameters are used to initialize the state vector
  - For PVDIS, the last two hits, and energy measurement from the calorimeter (10% error) is used for state vector initialization
  - For J/ψ, start the first fit like SIDIS, take the result of first fit and initialize a second fit
- Final Fitting starts from the last GEM and moves towards the first one
- Propagate the state vector on the first GEM toward the target
- Find the interaction vertex (find the closest approach between the trajectory and z-axis, may not the best way)
- Add the interaction vertex to the fit

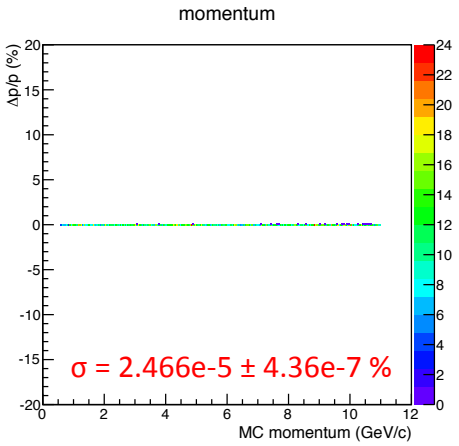
Vertex Reconstruction SIDIS FA

GEM Resolution = 0 um

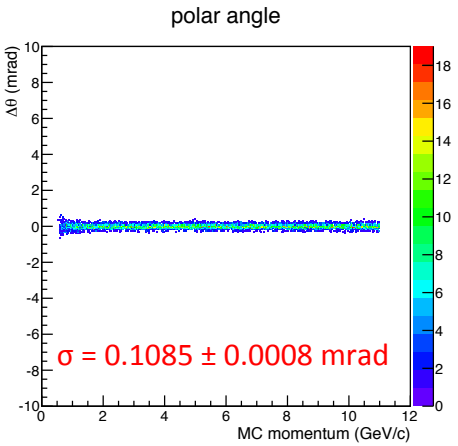


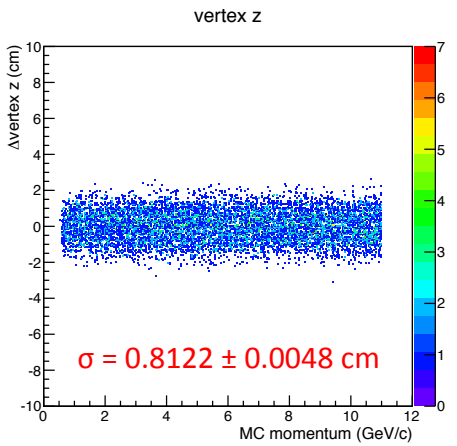


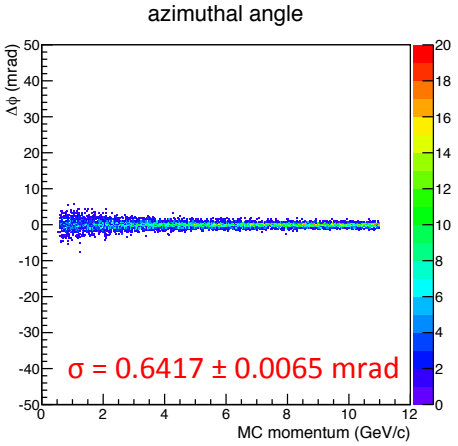


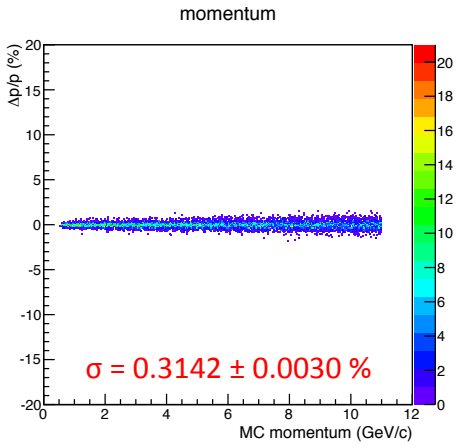


GEM Resolution = 30 um





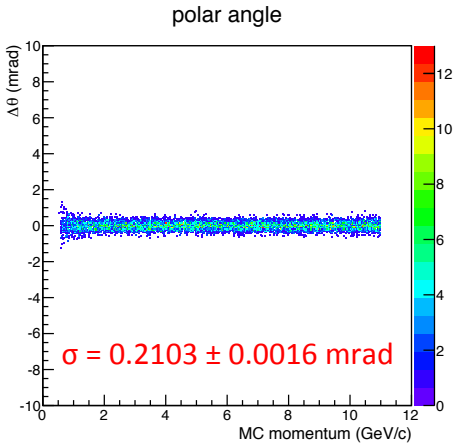


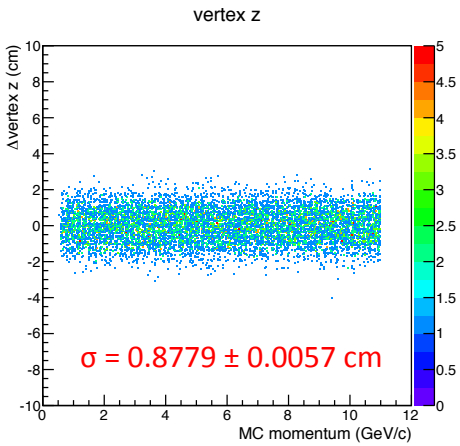


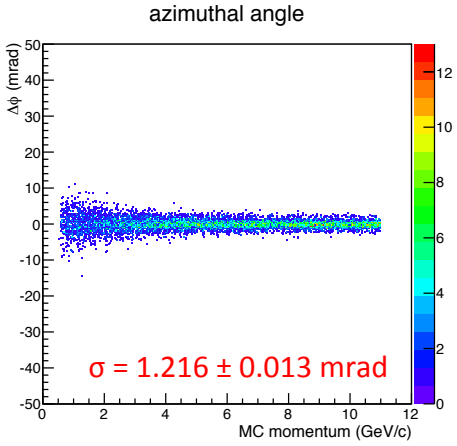
No process noise is simulated in this simulation. Generator is uniform.

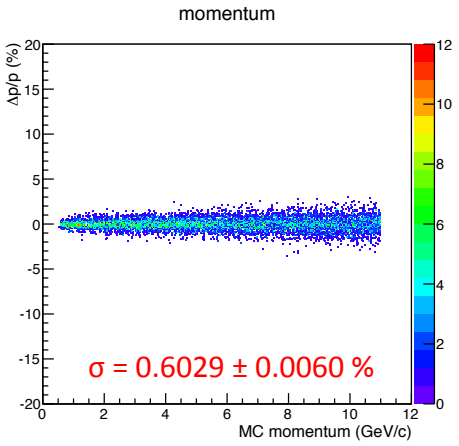
Vertex Reconstruction SIDIS FA

GEM Resolution = 60 um

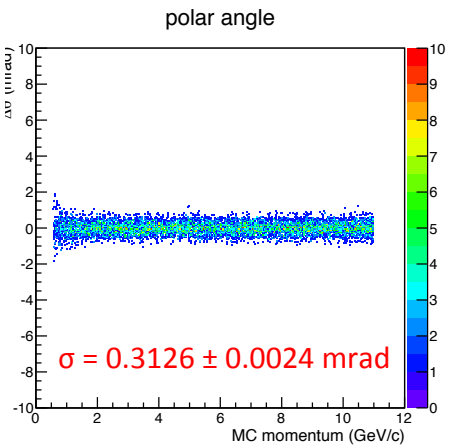


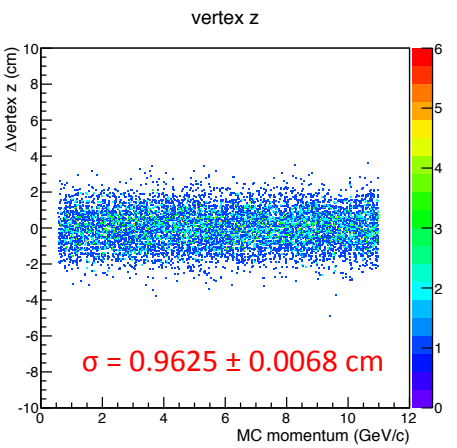


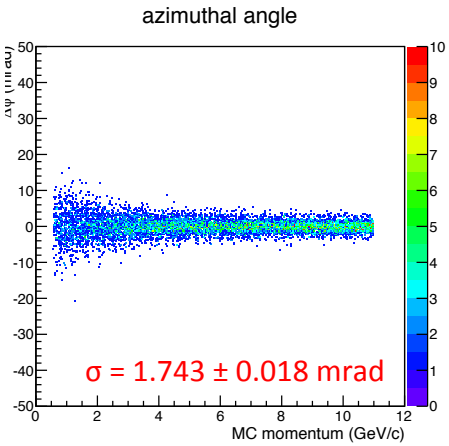


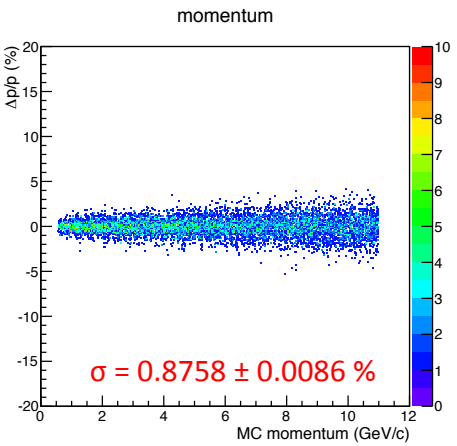


GEM Resolution = 90 um





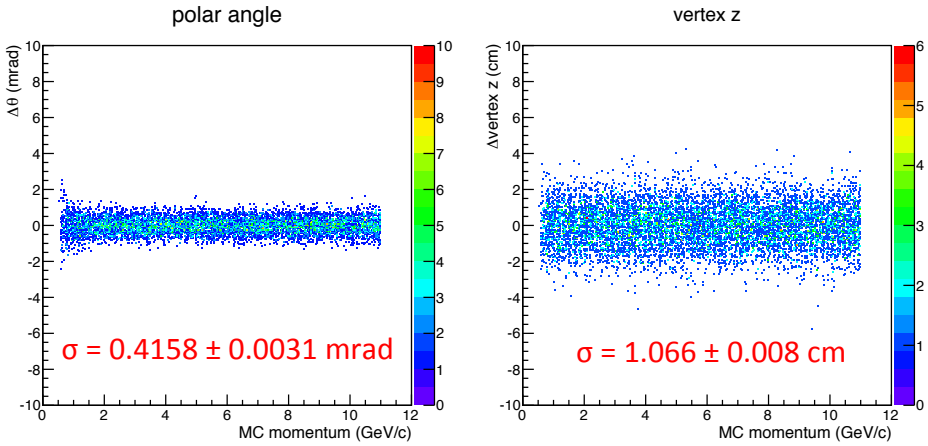


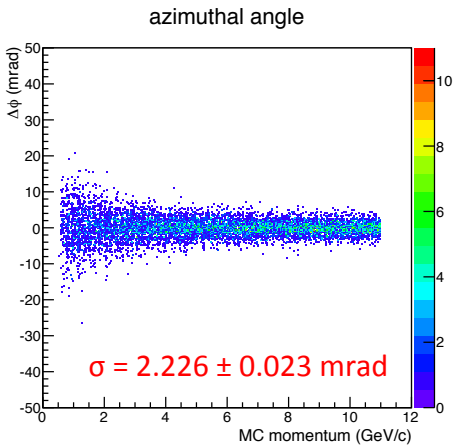


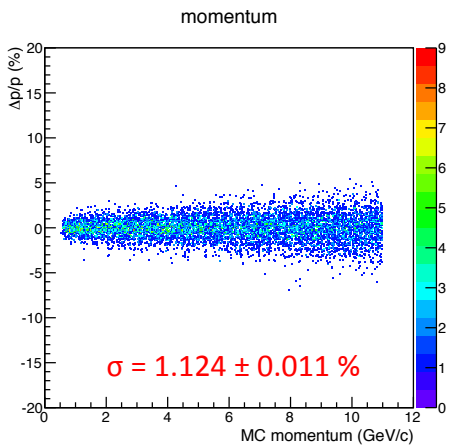
No process noise is simulated in this simulation. Generator is uniform.

Vertex Reconstruction SIDIS FA

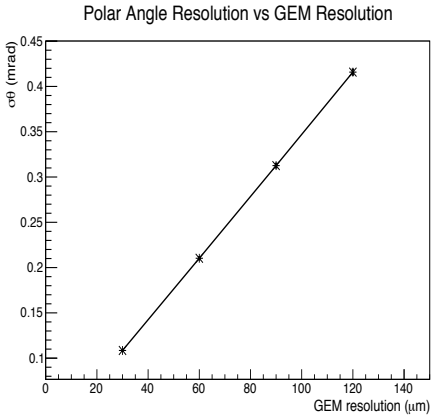
GEM Resolution = 120 um

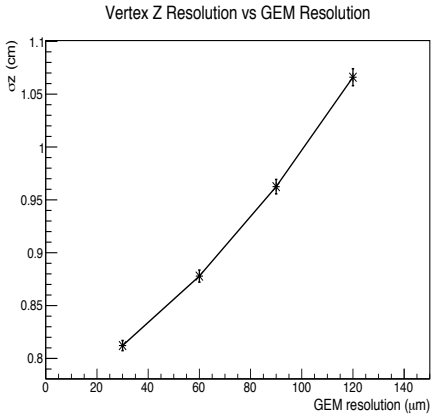


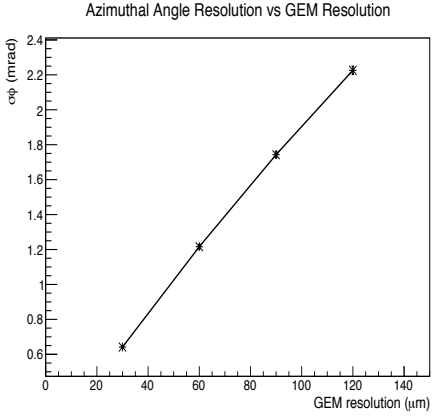


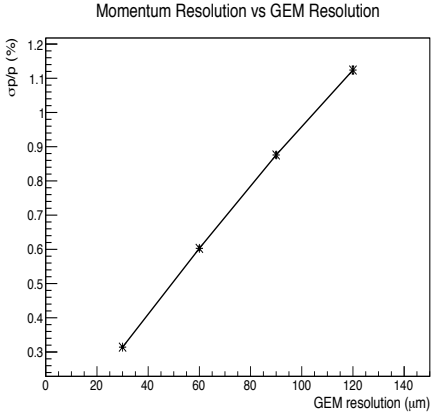


Resolution of Vertex Variables
Vs
GEM Resolution



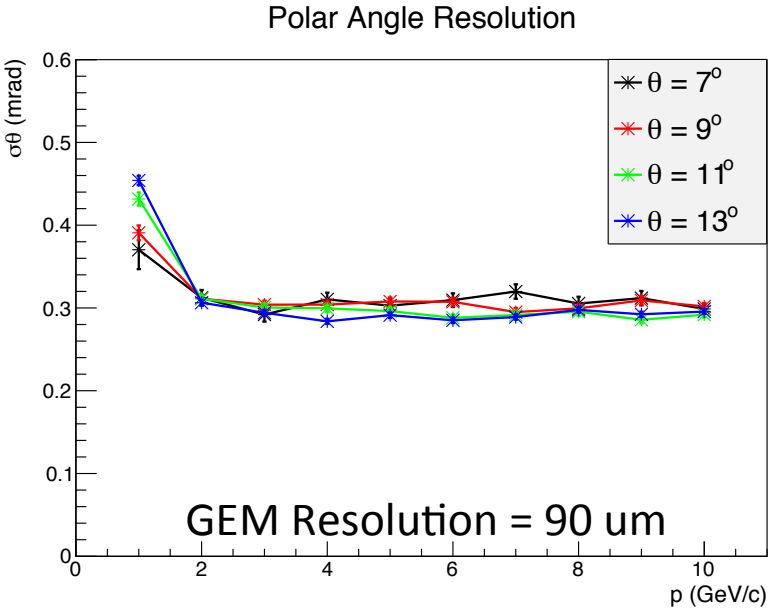


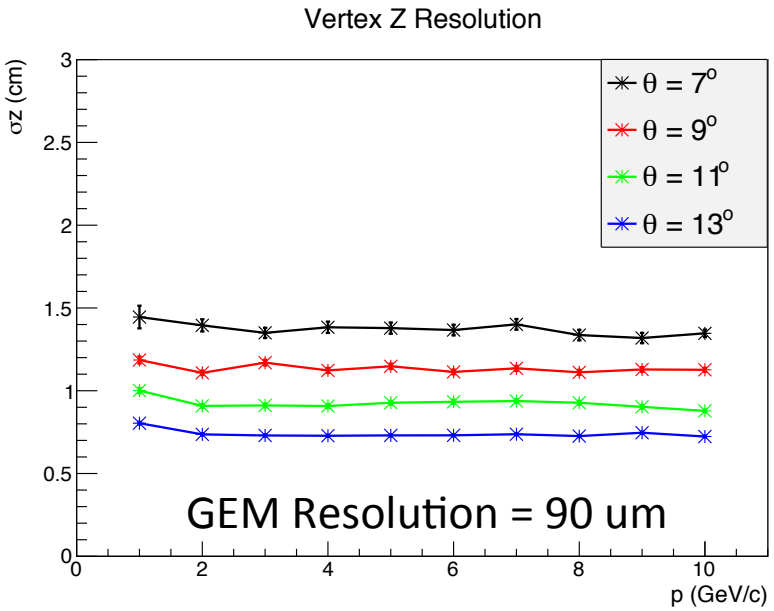


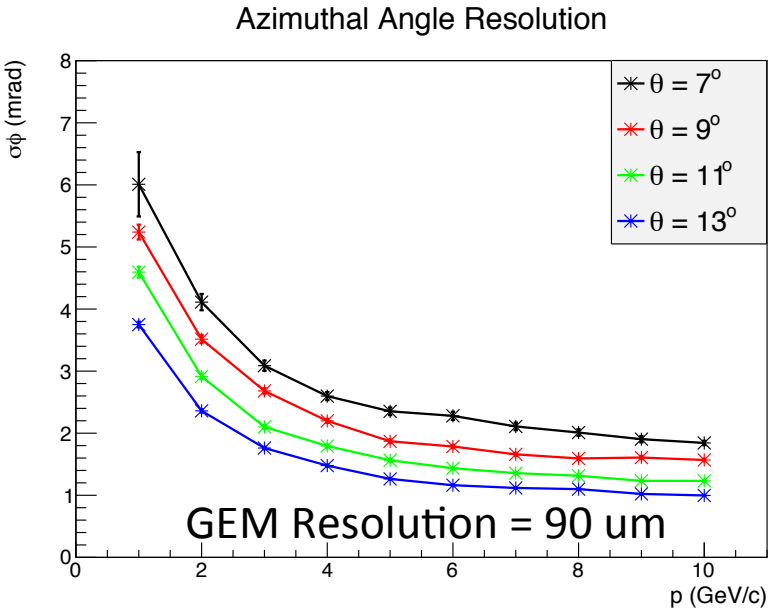


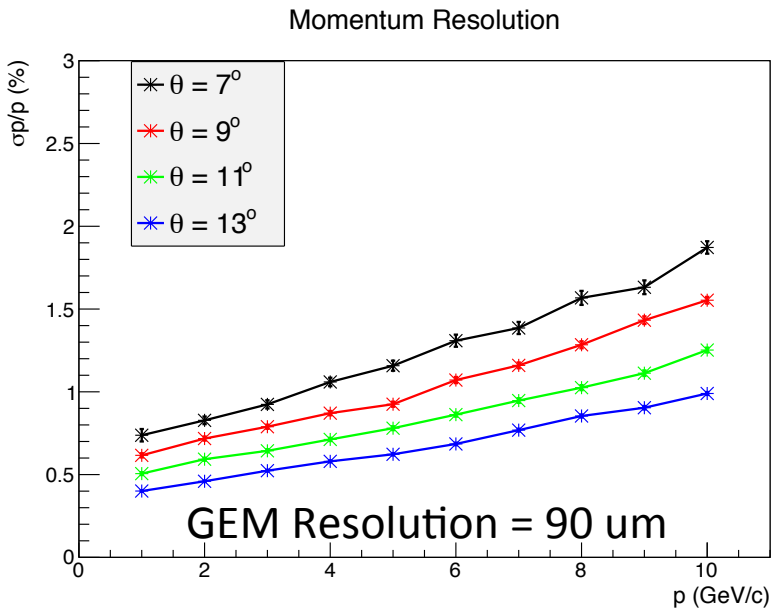
No process noise is simulated in this simulation. Generator is uniform.

Vertex Reconstruction SIDIS FA



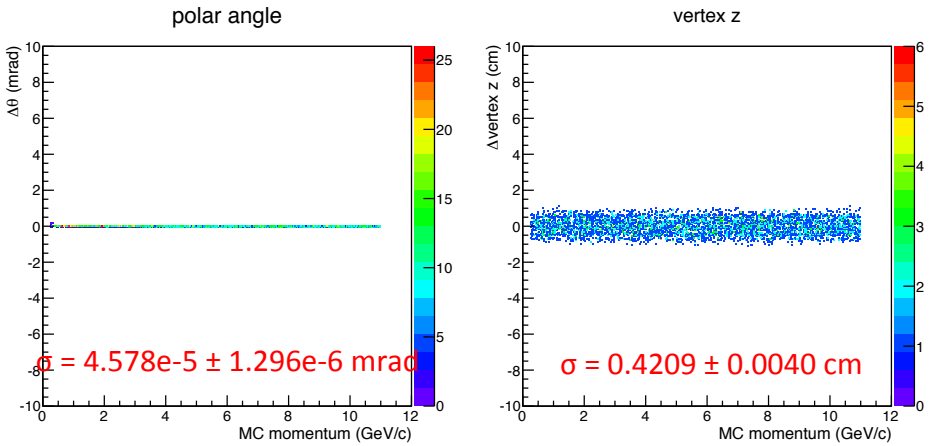


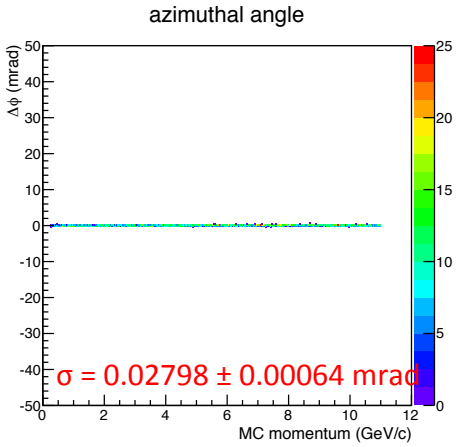


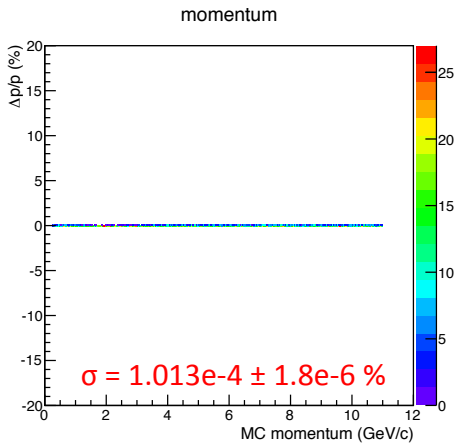


Vertex Reconstruction SIDIS LA

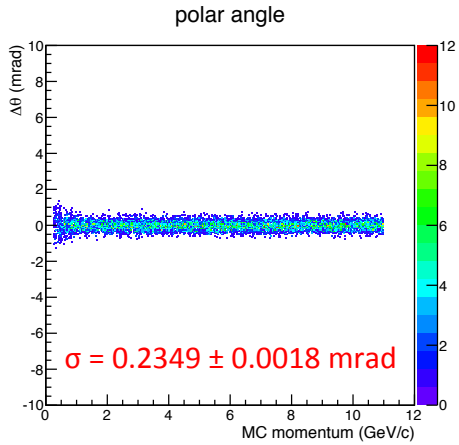
GEM Resolution = 0 um

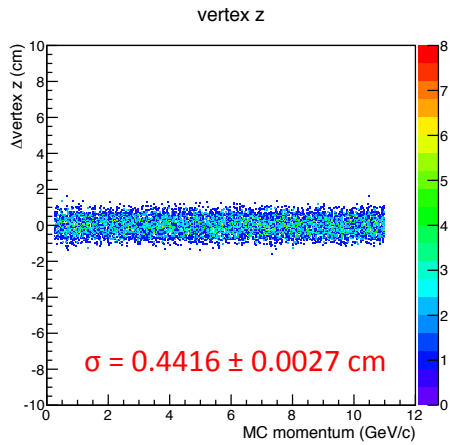


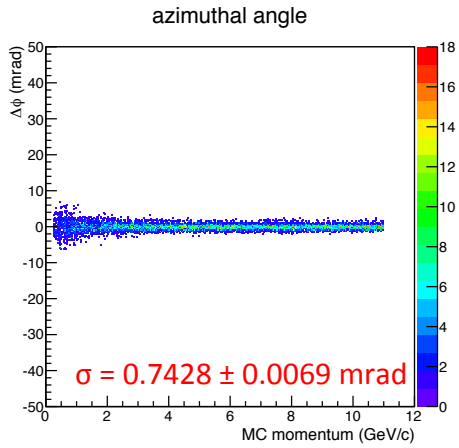


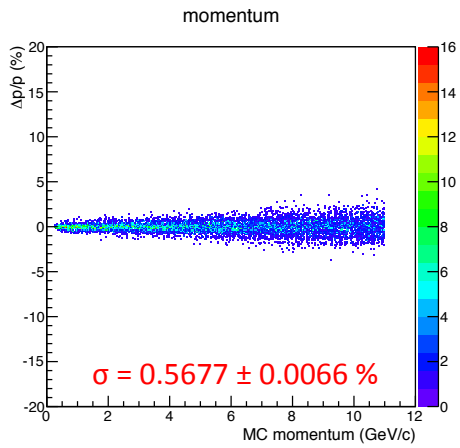


GEM Resolution = 30 um





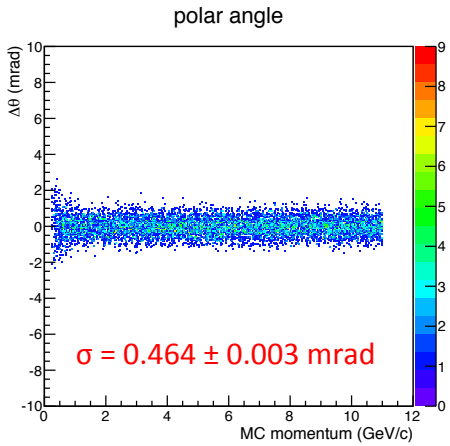


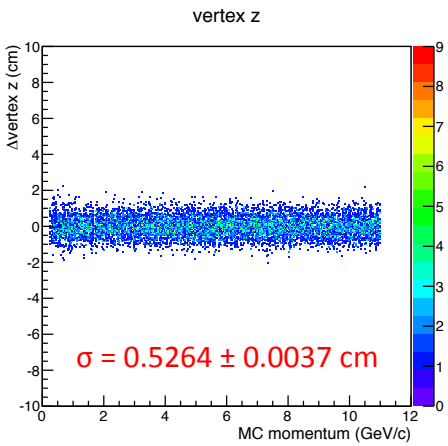


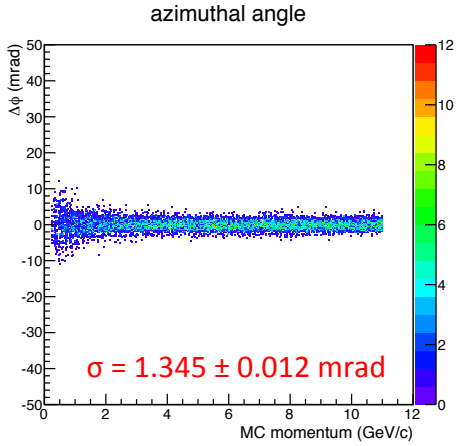
No process noise is simulated in this simulation. Generator is uniform.

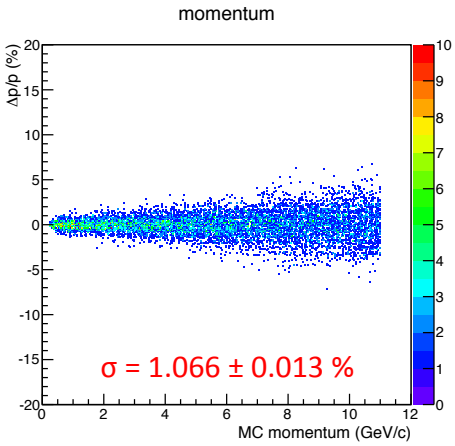
Vertex Reconstruction SIDIS LA

GEM Resolution = 60 um

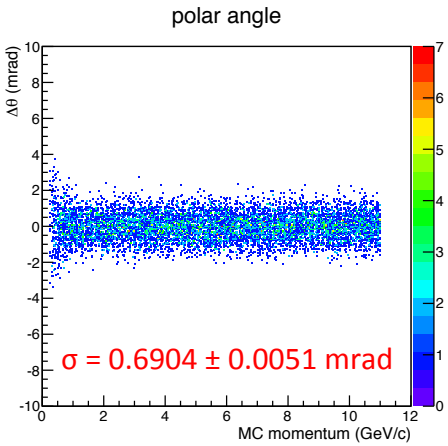


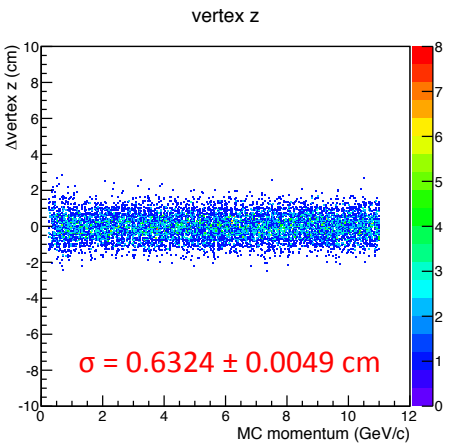


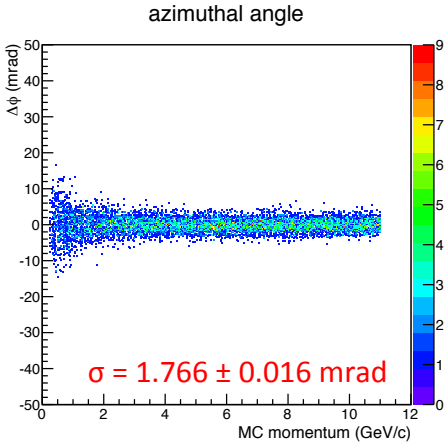


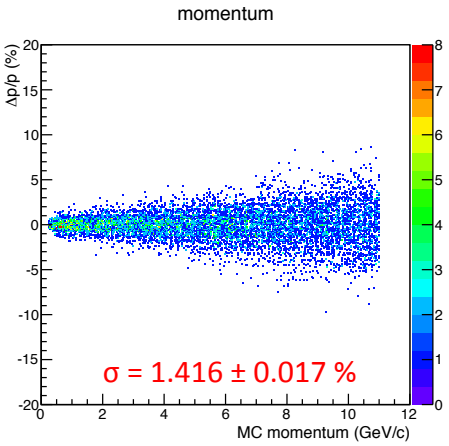


GEM Resolution = 90 um







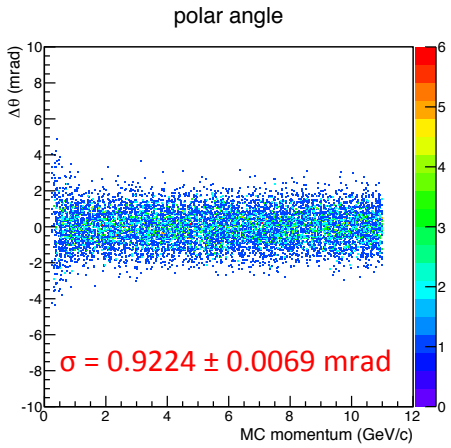


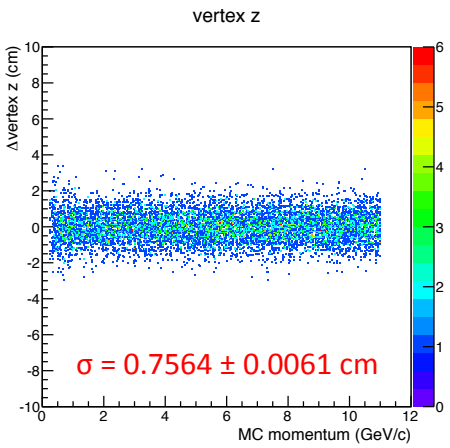
No process noise is simulated in this simulation. Generator is uniform.

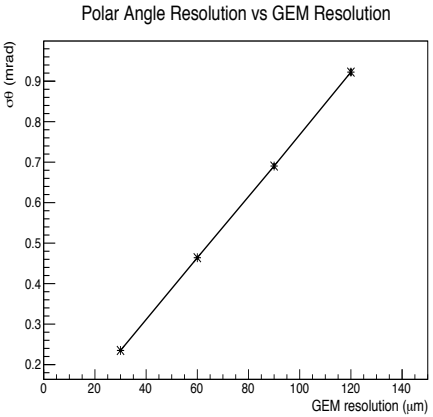
Vertex Reconstruction SIDIS LA

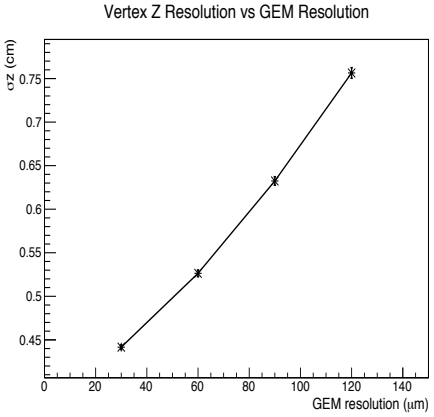
GEM Resolution = 120 um

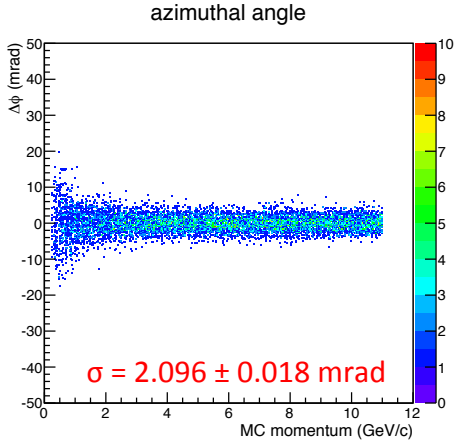
Resolution of Vertex Variables
Vs
GEM Resolution

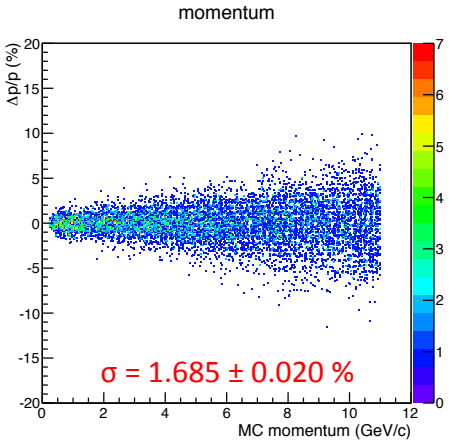


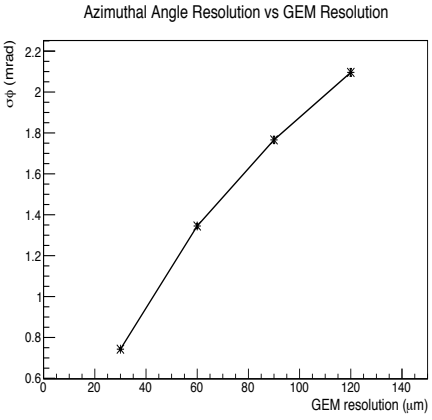


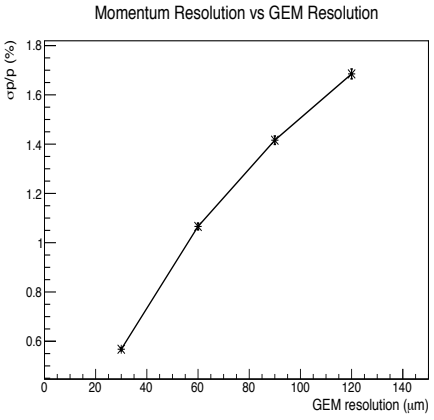






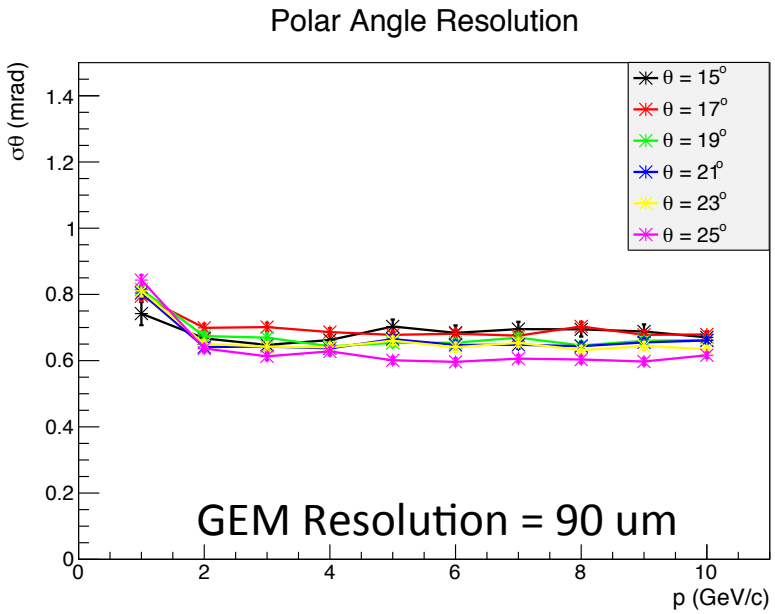


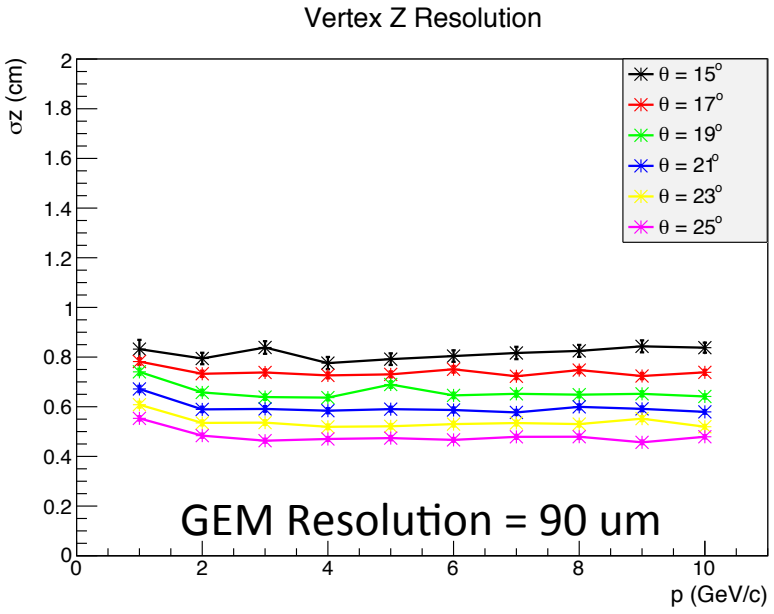


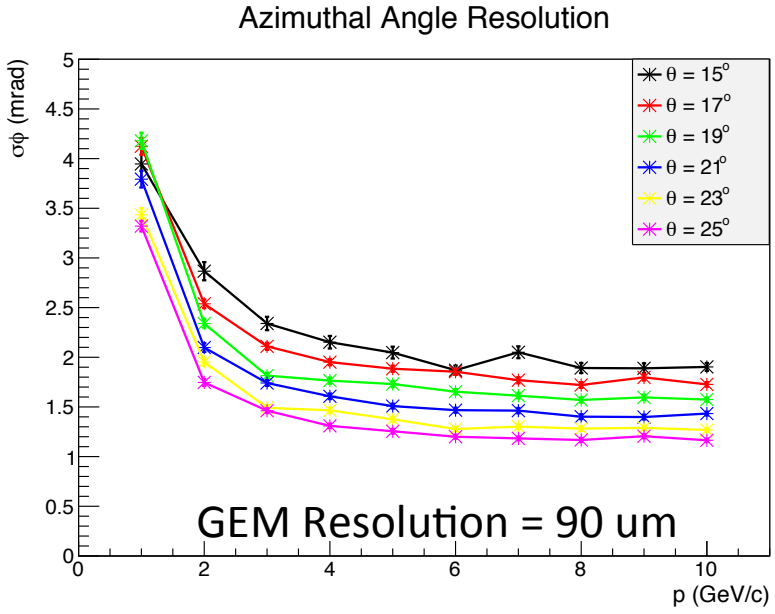


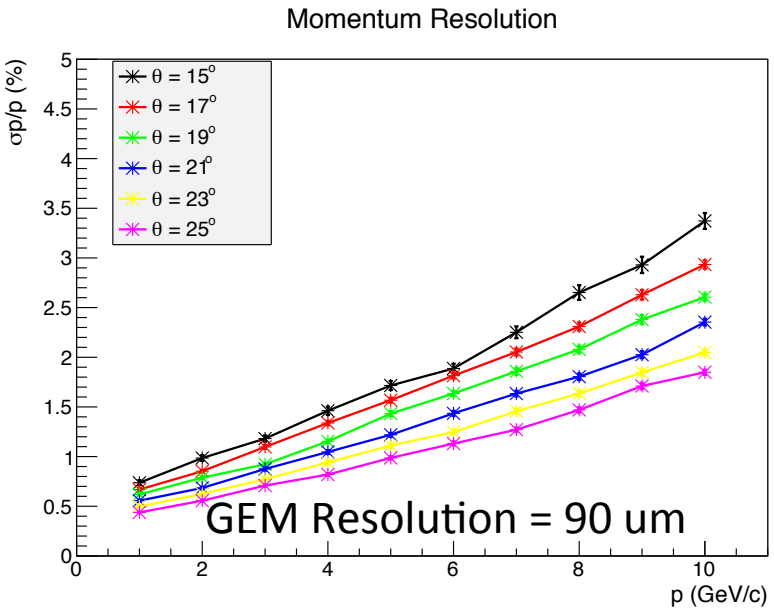
No process noise is simulated in this simulation. Generator is uniform.

Vertex Reconstruction SIDIS LA



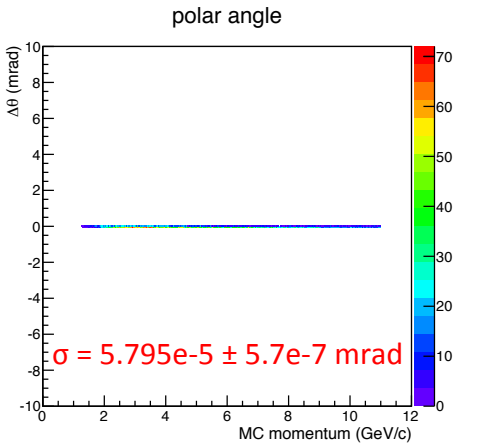


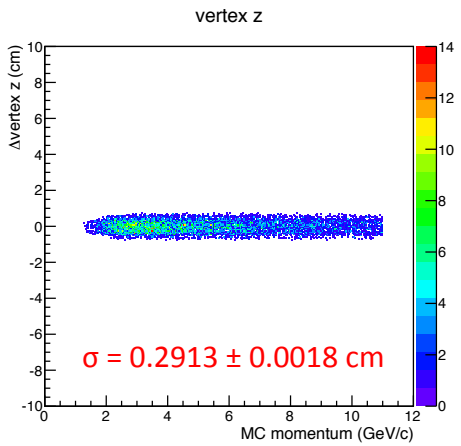


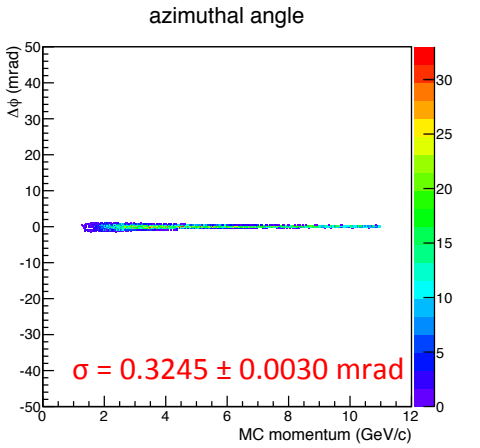


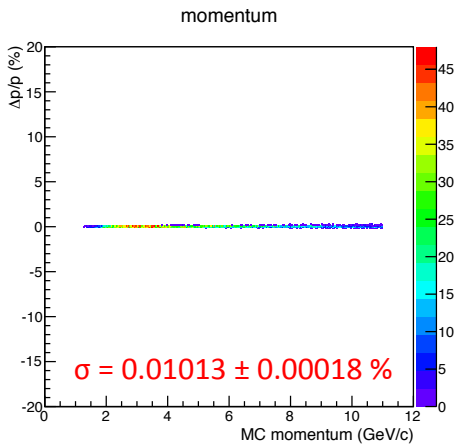
Vertex Reconstruction PVDIS

GEM Resolution = 0 um

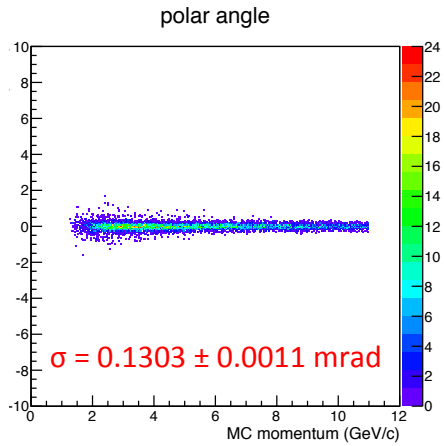


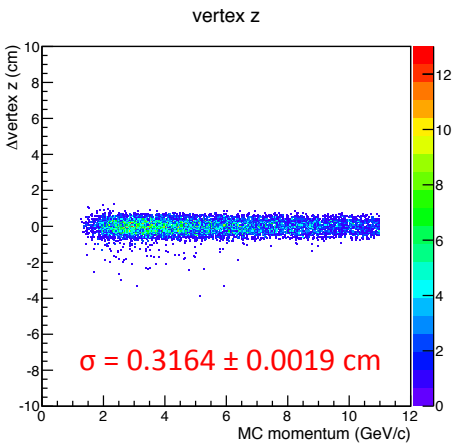


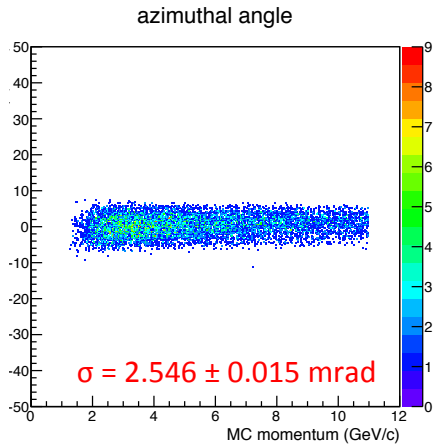


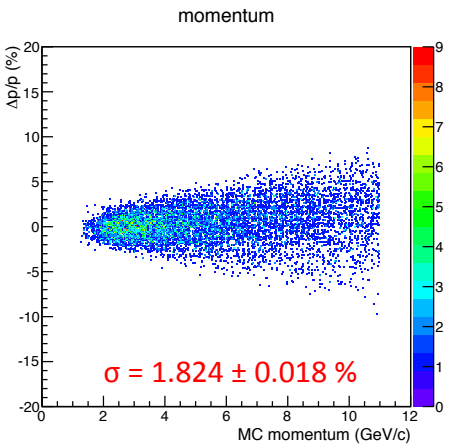


GEM Resolution = 30 um





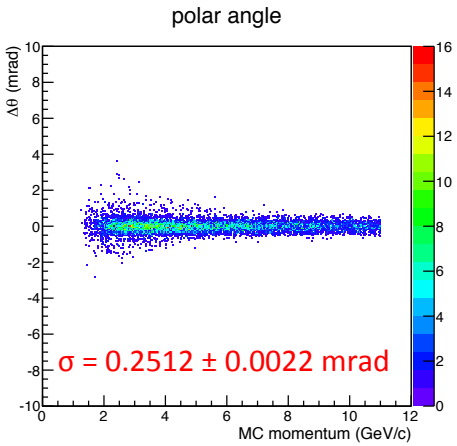


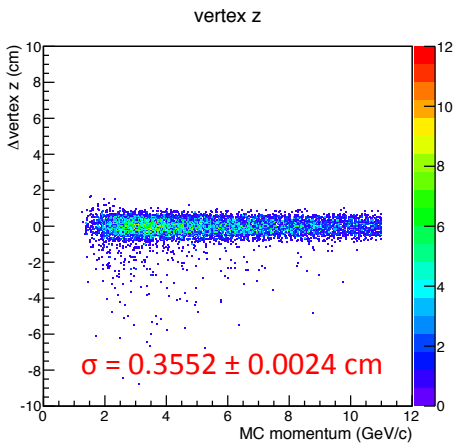


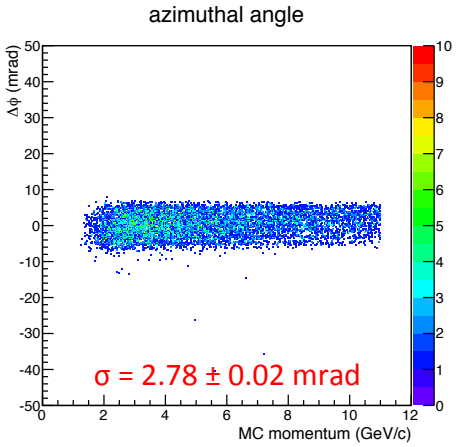
No process noise is simulated in this simulation. Generator is uniform.

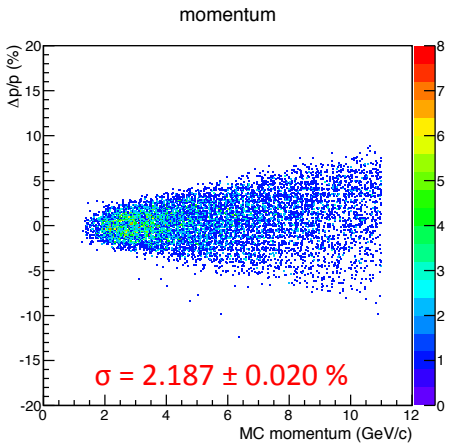
Vertex Reconstruction PVDIS

GEM Resolution = 60 um

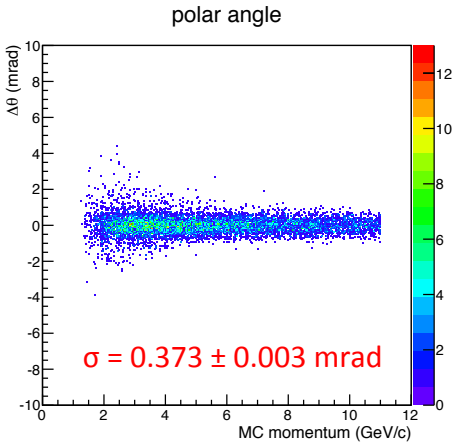


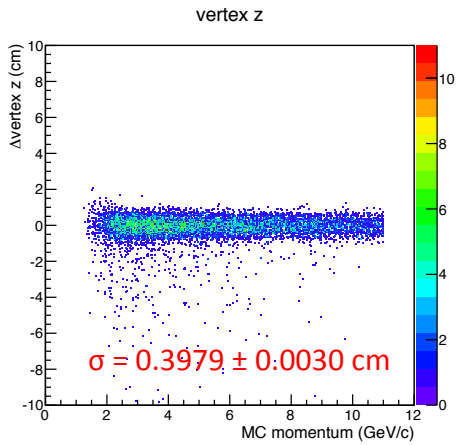


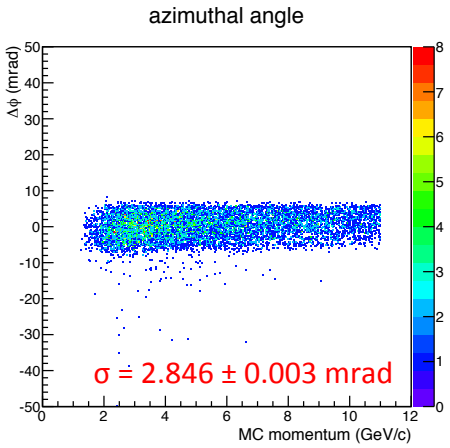


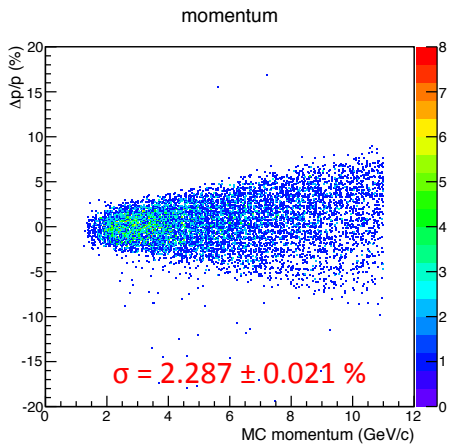


GEM Resolution = 90 um







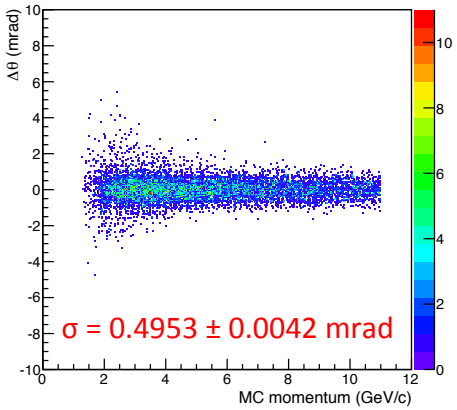


No process noise is simulated in this simulation. Generator is uniform.

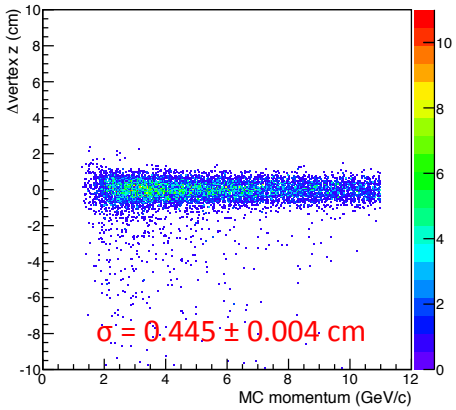
Vertex Reconstruction PVDIS

GEM Resolution = 120 um

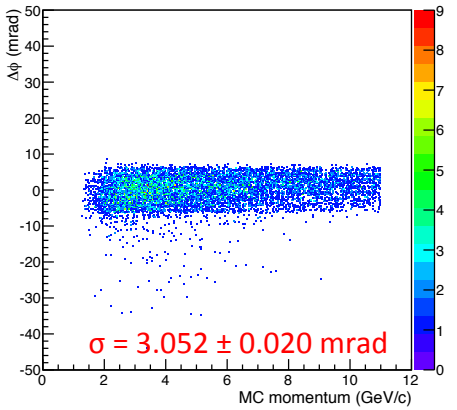
polar angle



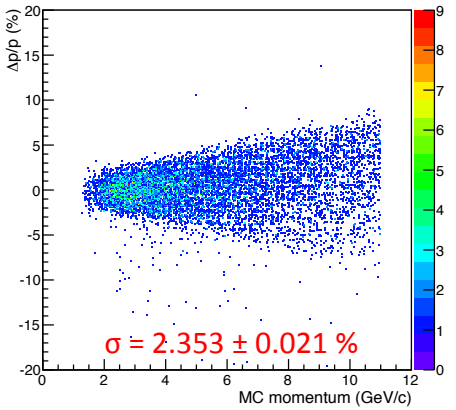
vertex z



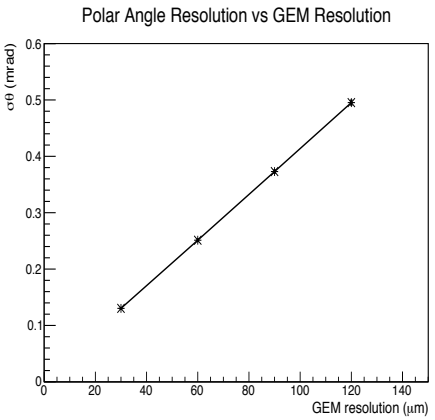
azimuthal angle

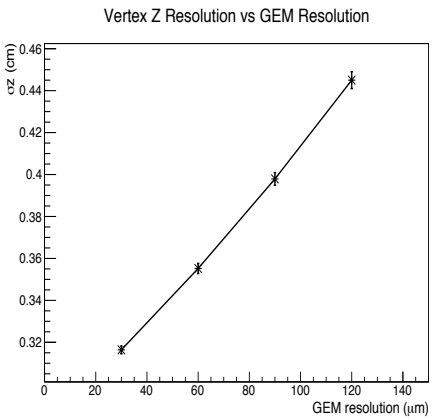


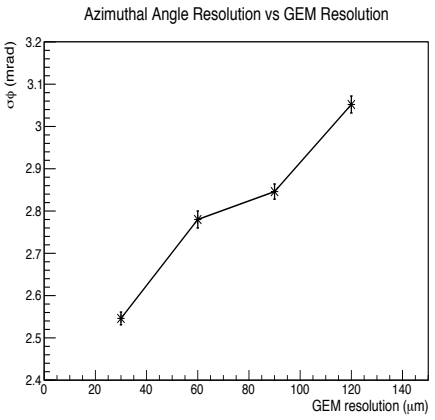
momentum

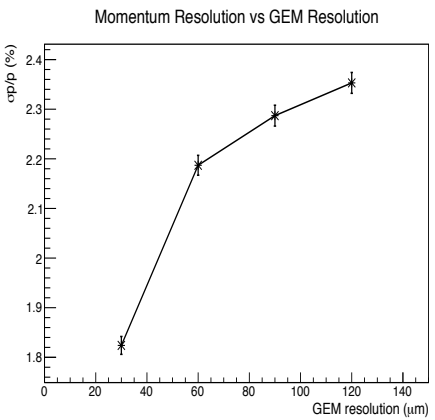


Resolution of Vertex Variables Vs GEM Resolution



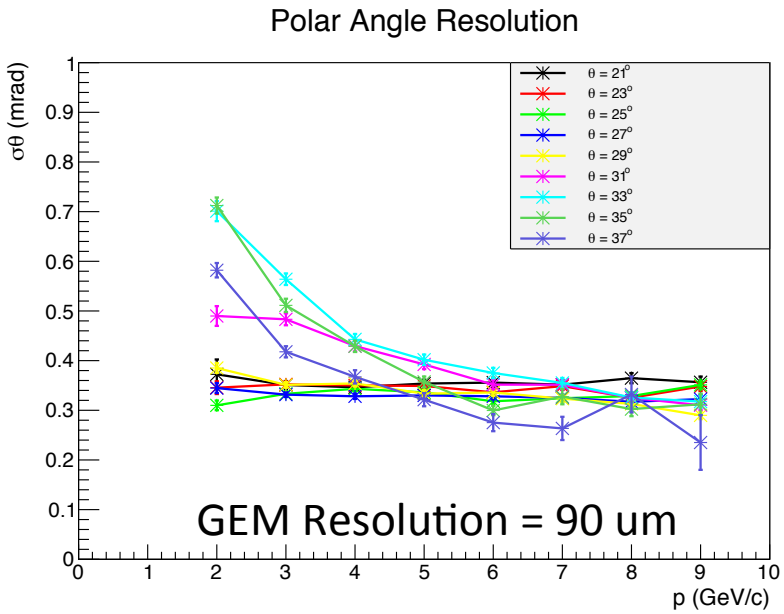


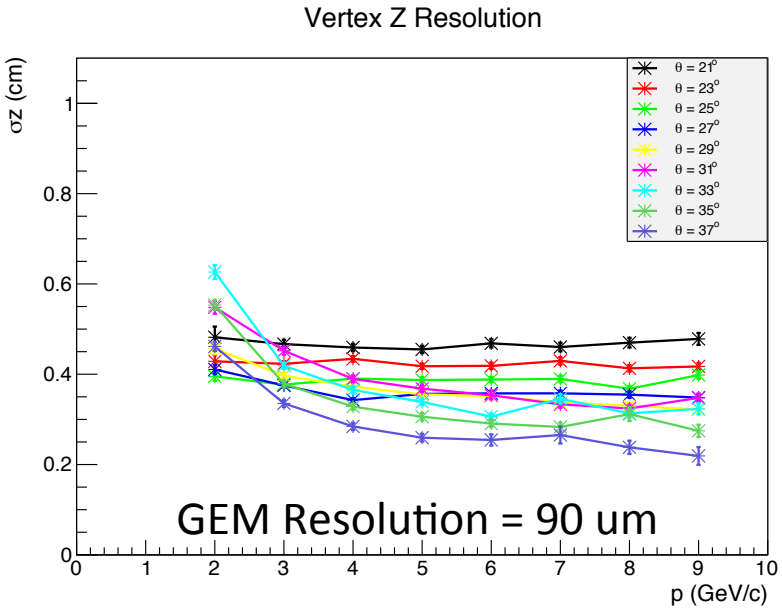


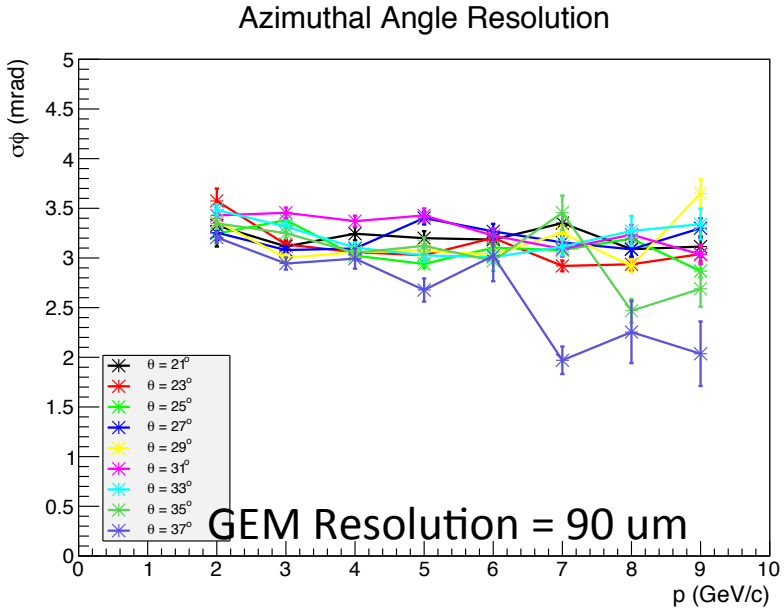


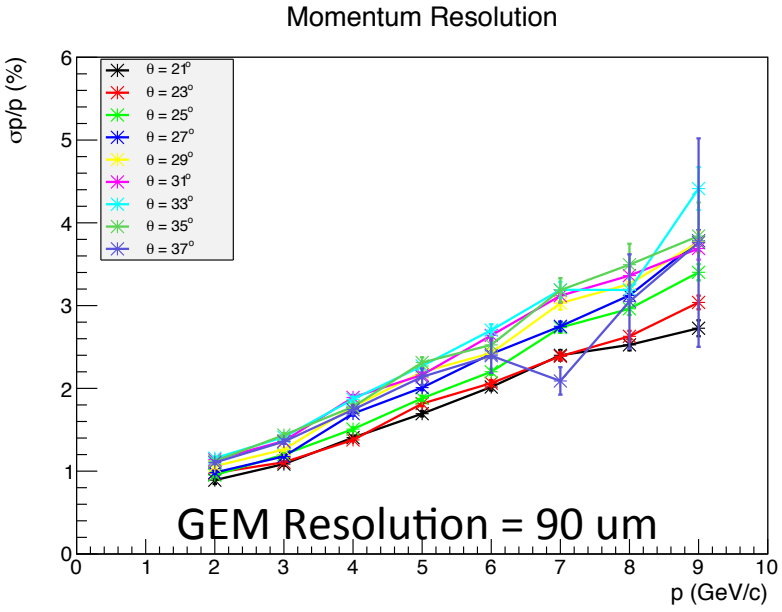
No process noise is simulated in this simulation. Generator is uniform.

Vertex Reconstruction PVDIS









Conclusion

- Kalman filter with current track model and propagation method works for both SIDIS and PVDIS
- Further development quite necessary (better error estimate, better software structure, more functionalities…)
- Kalman filter algorithm can be easily developed into a pattern recognition algorithm, which performs the job of distinguishing signal tracks from background tracks (will consider this later)

Plan

- Further Development for the Kalman filter reconstruction method
- Progressive tracking for PVDIS (curved track) is currently being developed, have a preliminary version, need more tests (digitization)

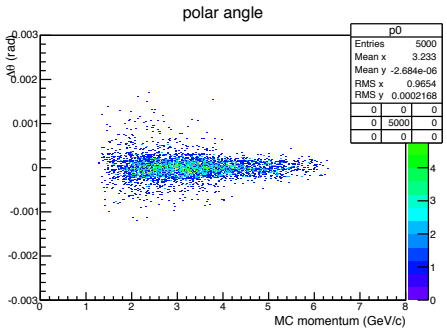
- Reference:

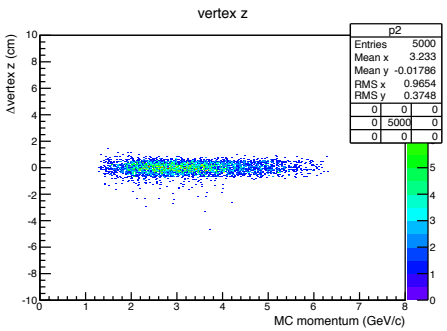
[1] Extended Kalman Filter. http://www-jlc.kek.jp/subg/offl/kaltest/doc/ReferenceManual.pdf

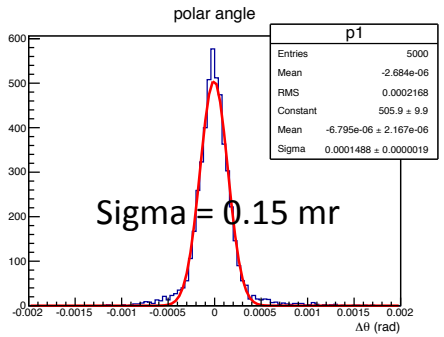
[2] Application of a Kalman Filter and a Deterministic Annealing Filter for Track Reconstruction in the HADES Experiemtn. https://www-alt.gsi.de/documents/DOC-2013-Aug-7-1.pdf

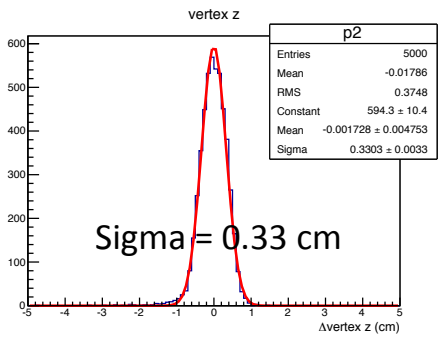
Backup Slides

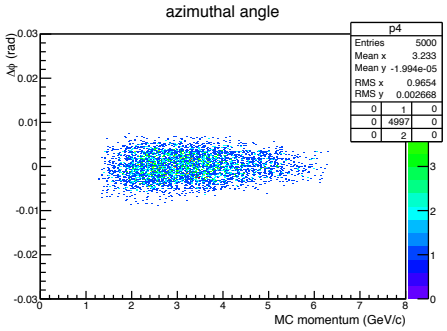
PVDIS Vertex Reconstruction

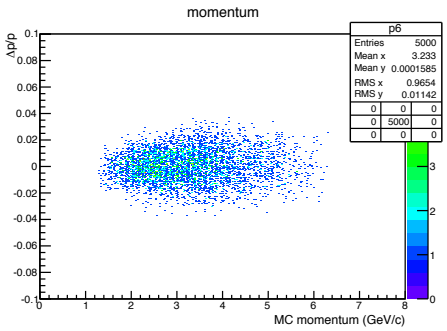


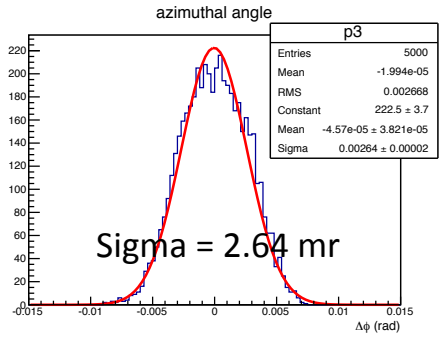


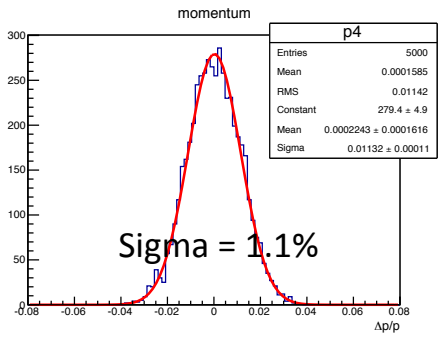












No process noise is simulated in this simulation. Generator is eDIS.

Classical 4th Order Runge-Kutta

First of all, take the derivative of the state vector with respect to z, this is the differential equation that is going to be solved by the Runge-Kutta method:

$$\frac{d\vec{a}}{dz} = \begin{pmatrix} dx/dz \\ dy/dz \\ dt_x/dz \\ dt_y/dz \\ d(q/p)/dz \end{pmatrix} = \begin{pmatrix} t_x \\ t_y \\ t'_x \\ t'_y \\ 0 \end{pmatrix} = f(\vec{a}, z)$$

The solution of the 4th order Runge-Kutta method is:

$$\Delta\vec{a}_1 = h \cdot f(\vec{a}(z_0), z_0)$$

$$\Delta\vec{a}_2 = h \cdot f(\vec{a}(z_0) + \frac{1}{2}\Delta\vec{a}_1, z_0 + \frac{1}{2}h)$$

$$\Delta\vec{a}_3 = h \cdot f(\vec{a}(z_0) + \frac{1}{2}\Delta\vec{a}_2, z_0 + \frac{1}{2}h)$$

$$\Delta\vec{a}_4 = h \cdot f(\vec{a}(z_0) + \Delta\vec{a}_3, z_0 + h)$$

$$\vec{a}_f = \vec{a}_0 + \frac{1}{6}\Delta\vec{a}_1 + \frac{1}{3}\Delta\vec{a}_2 + \frac{1}{3}\Delta\vec{a}_3 + \frac{1}{6}\Delta\vec{a}_4 + \mathrm{O}(h^5)$$

Classical 4th Order Runge-Kutta

- The propagator matrix of this process is:

$$F = \frac{d\vec{a}_f}{d\vec{a}_0} = I + \frac{1}{6}F_1 + \frac{1}{3}F_2 + \frac{1}{3}F_3 + \frac{1}{6}F_4$$

$$F_i = \frac{d\Delta\vec{a}_i}{d\vec{a}_0}$$

- Using this method, there is no need for a pre-defined geometric track. To initialize the fit, x_0 and y_0 can be taken from the last GEM tracker along the beam direction, tx_0 and ty_0 can be calculated using the last two GEM trackers (where field is low and track is almost straight), finally p is given by the calorimeter (we always have a EC hit for PVDIS)

Process Noise Treatment

- In general, there are three important process noises one will need to consider during tracking reconstruction
  - Coulomb multiple scattering:

$$C_{MS} = \left(\frac{13.6\,MeV}{\beta pc}\right)^2 t\,(1 + 0.0038 \ln t)^2$$

  - Ionization:
    - For spinless particle:

$$-\left\langle \frac{dE}{dx} \right\rangle = Kz^2 \frac{Z}{A} \frac{1}{\beta^2} \left(\frac{1}{2} \ln \left(\frac{2m_e c^2 \beta^2 \gamma^2 T_{max}}{I^2} \right) - \beta^2 \right)$$

    - For electron:

$$-\left\langle \frac{dE}{dx} \right\rangle = \frac{1}{2} K \frac{Z}{A} \left(2 \ln \frac{2m_e c^2}{I} + 3 \ln \gamma - 1.95 \right).$$

$$K = 4\pi N_A r_e^2 m_e c^2 \qquad T_{max} = \frac{2m_e c^2 \beta^2 \gamma^2}{1 + 2\gamma \frac{m_e}{M} + \left(\frac{m_e}{M}\right)^2}.$$

  - Bremsstrahlung:

$$\left\langle \frac{dE}{dx} \right\rangle = -\frac{E}{X_0}.$$

More about Process Noise Treatment

- For each Runge-Kutta step that is taken, we make one correction for the process noise
- We ADD the lost energy to the track if the fitting is backward, while SUBTRACT the lost energy if the fitting goes forward
- The process noise matrix Q will mainly affected by Coulomb multiple scattering and Bremsstrahlung radiation. The effect of ionization on Q is neglected because this effect is only strong for low momentum particle, where resolution is dominated by Coulomb multiple scattering

Effect on Q due to Coulomb Multiple scattering

$$Q(t_x, t_x) = (1 + t_x^2) \cdot (1 + t_x^2 + t_y^2) \cdot C_{MS}$$
$$Q(t_y, t_y) = (1 + t_y^2) \cdot (1 + t_x^2 + t_y^2) \cdot C_{MS}$$
$$Q(t_x, t_y) = Q(t_y, t_x) = t_x t_y \cdot (1 + t_x^2 + t_y^2) \cdot C_{MS}.$$

Effect on Q due to Bremsstrahlung radiation

$$Q\left(\frac{q}{p}, \frac{q}{p}\right) = \left(\frac{q}{p}\right)^2 \left(e^{-t\frac{\ln 3}{\ln 2}} - e^{-2t}\right)$$

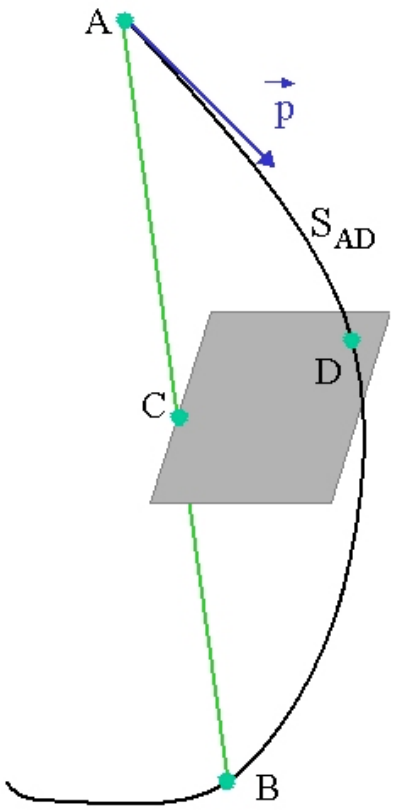
Potential Reasons for This Disagreement

- The performance of Geant4 field propagation is governed by a couple important parameters. They must be tuned carefully in order to achieve high precision and accuracy

One important parameter is the DeltaIntersection, defined in G4FieldManeger. It determines how accurately the intersection between the track and certain detector boundary is calculated. This is done very carefully in my simulation.

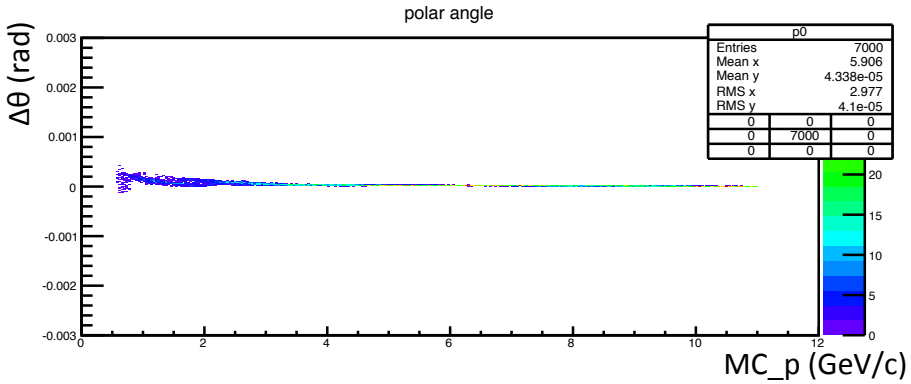
There are many other parameters might need to adjust. I am still reading Geant4 source code to figure them out

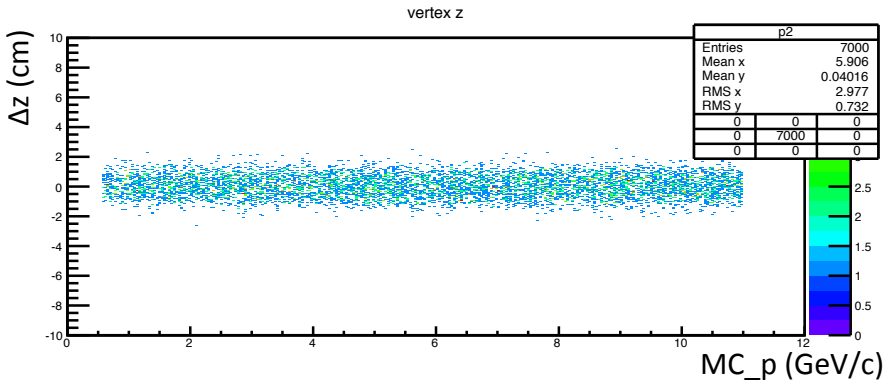
Another reason will probably be the G4CacheMagField, which will use the magnetic field vector at the previously evaluated point, if it is sufficiently close to the currently evaluated point

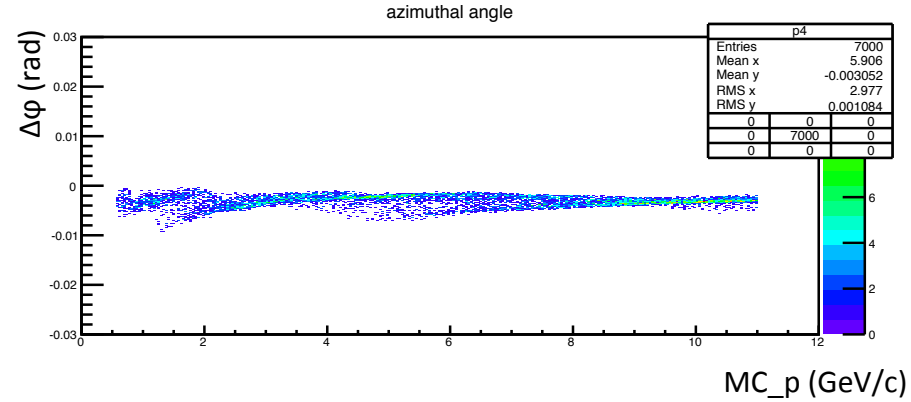


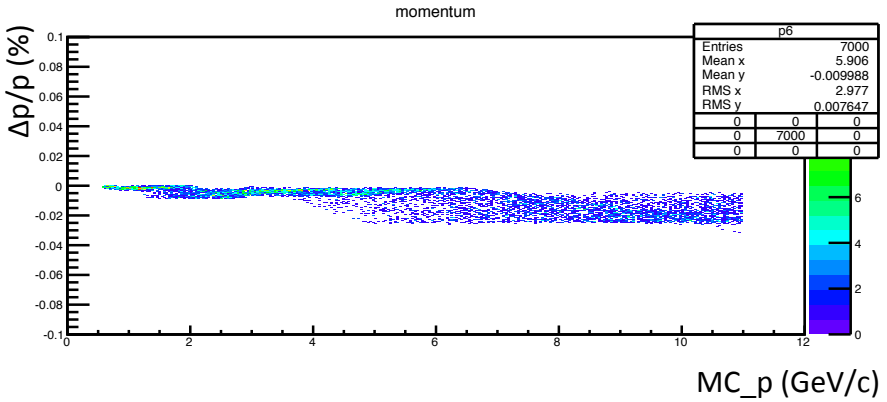
Reconstructed Vertex Variables if Using Old GEMC Simulation

Condition: No GEM smearing, no multiple scattering, no energy loss









Comment: These are the SIDIS forward angle events. The reconstructed vertex variables are much worse for SIDIS large angle event and PVDIS.