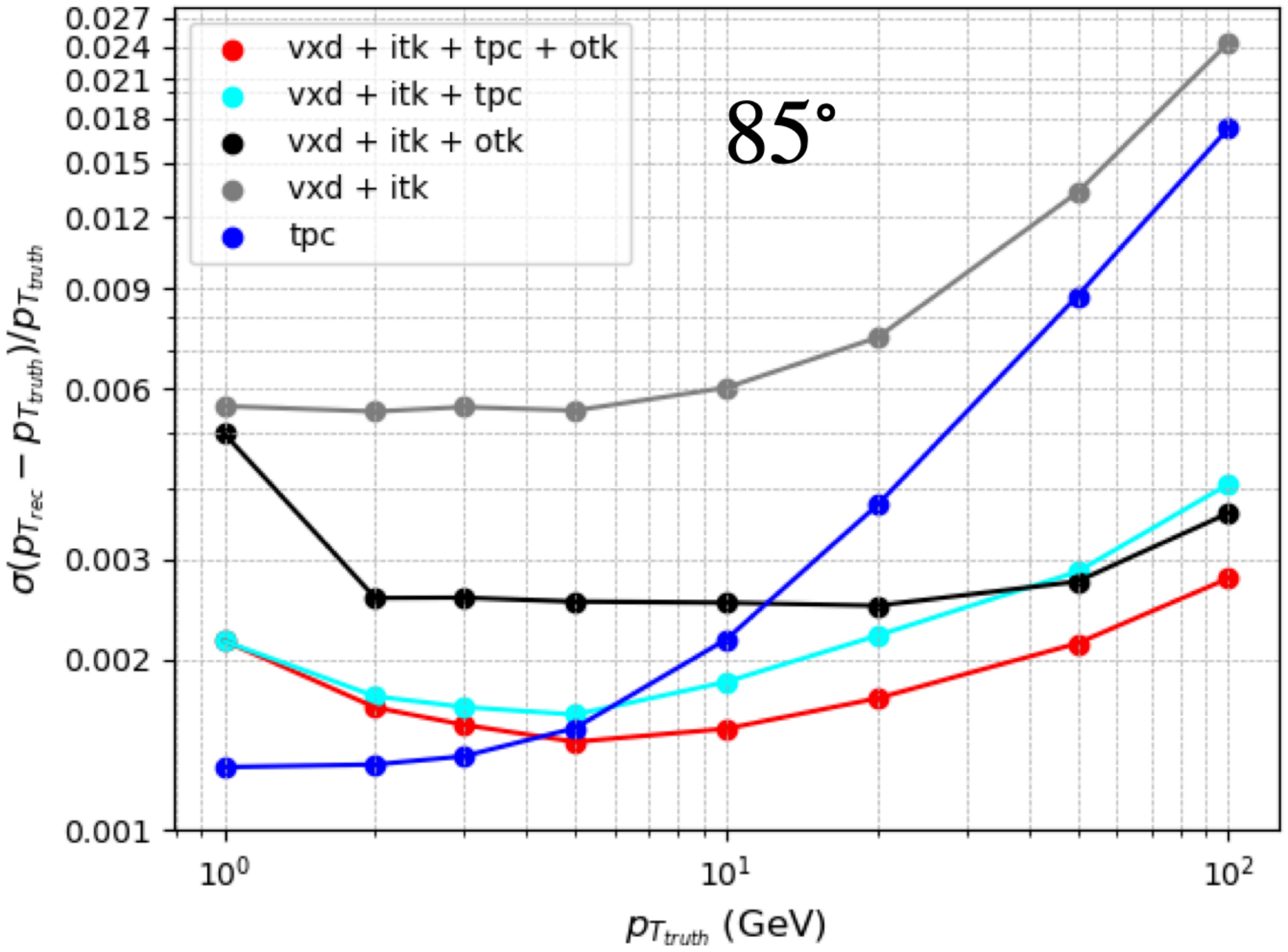
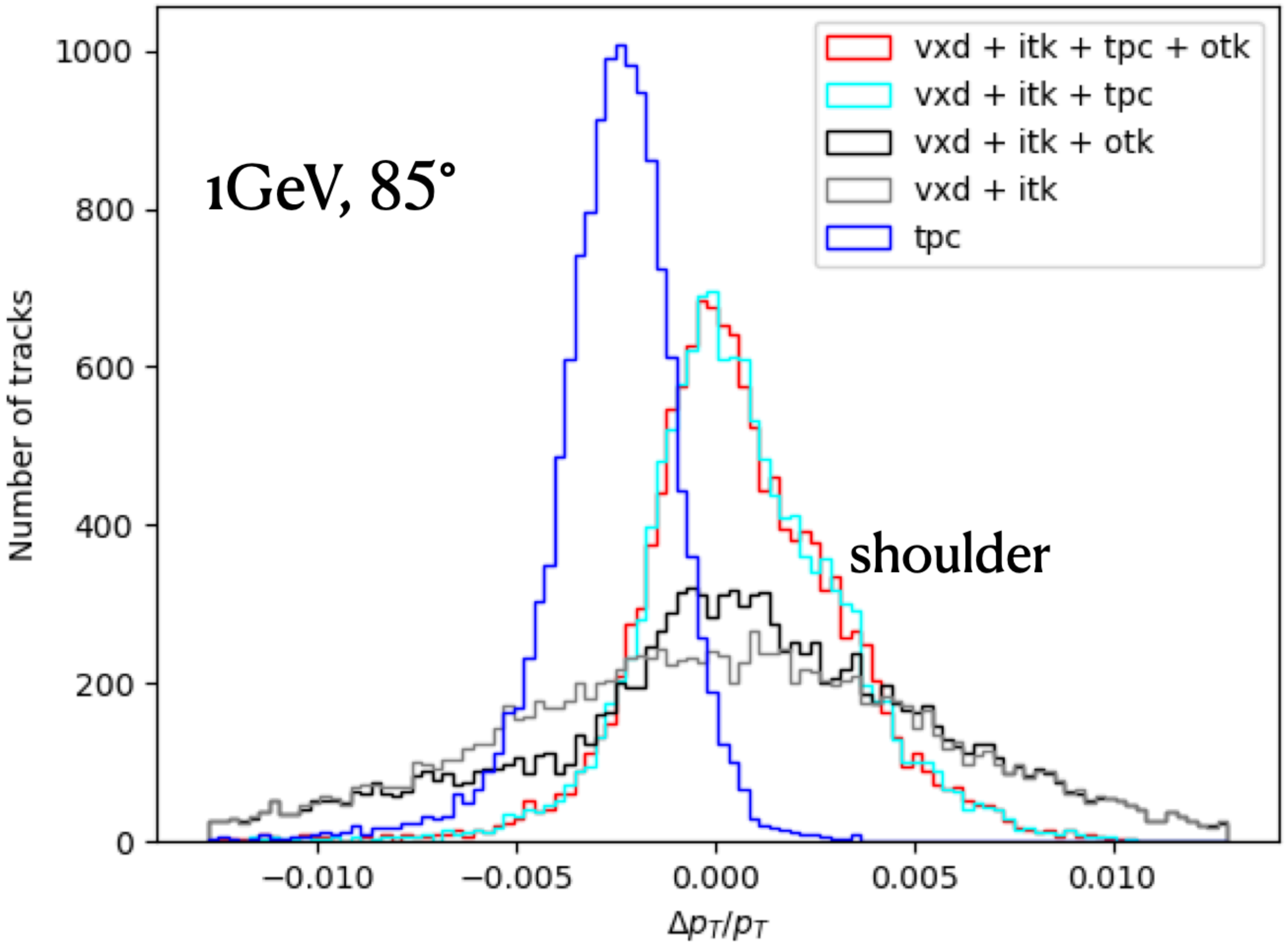
Trk/Vtx & PID

C.Zhang/22Nov2024

Trk

- Low pT performance needs to be understood; working in progress...





PID

- K/Pi separation @ $\mathrm{p} = 12\mathrm{GeV}, \cos(\theta) = 0.3$; Investigation for a better understanding of the differences is ongoing…
- Switch attention to development of new algorithm

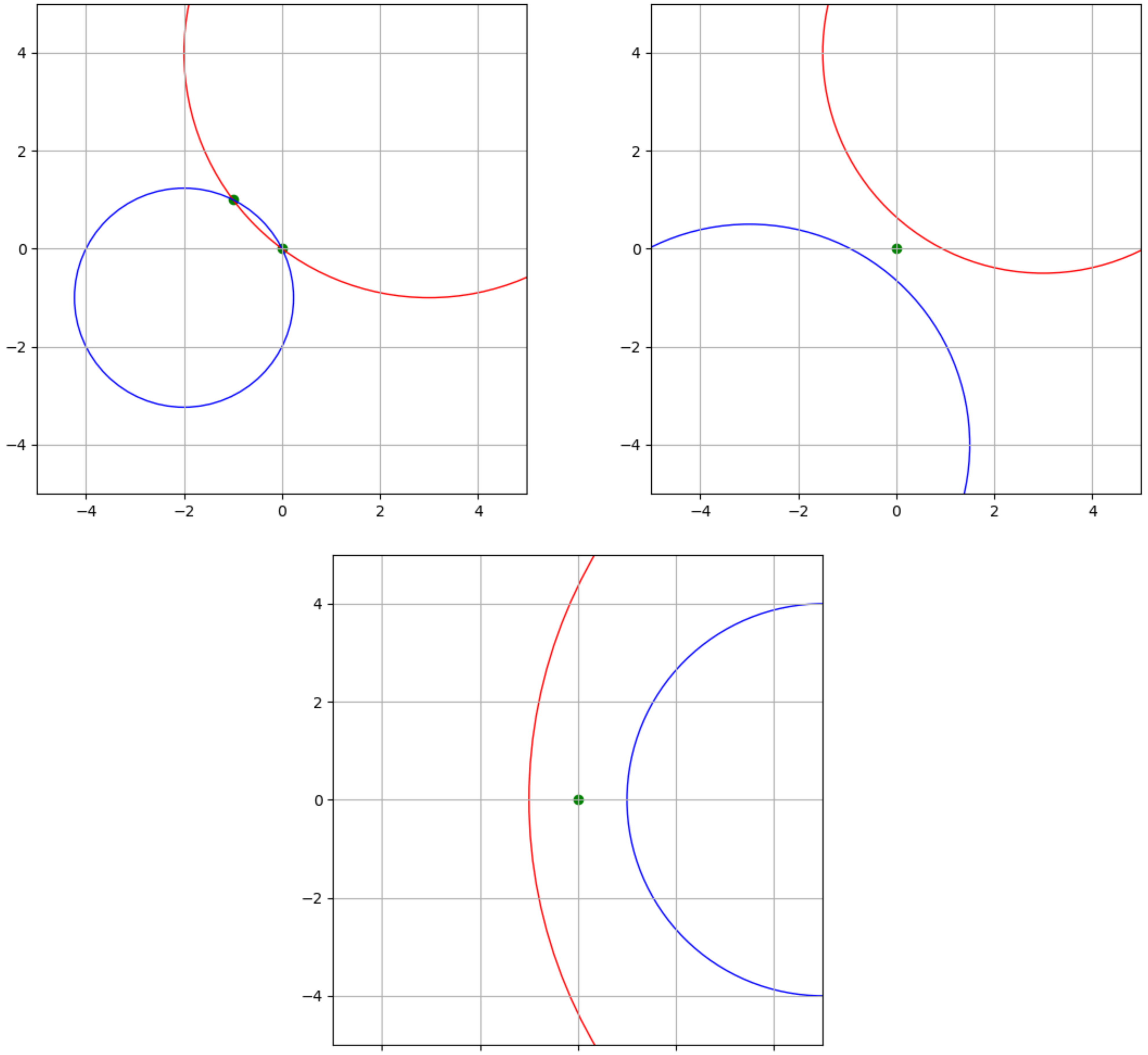
	TDR Truth (Garfield, dNdx)		**Reference (Geant4, dEdx[** $\mathrm{MeV^{-1}g^{-1}cm^{2}}$ **])**	
	Pi	**K**	**Pi**	**K**
mean	36.0	32.2	1.169	1.003
sigma	0.52	0.50	0.029	0.025
sigma/mean	0.014	0.015	0.025	0.025
separation	~ 5 sigma		~ 4.3 sigma	
	TDR Rec (Garfield, dNdx)		**Reference Rec**	
	Pi	K	Pi	K
mean	58.1	53.6		
sigam	1.37	1.39	Assum 0, 20, 50, 100% worse than truth	
sigma/mean	0.024	0.025		
separation	~ 2.3 sigma		~ 4.3, 3.6, 2.8, 2.1 sigma	

Primary vertex reconstruction

- A package has been developed which takes “CompleteTracks” as its input and produces vertex as its output

1. Feed all tracks into the algorithm for now; selections based on such as chi2, d0/z0, etc. will be applied in here soon.
2. Vertex finder
   - For each track pair, determine its geometric centre and select the one with the highest compatibility with all other tracks as the initial vertex.
3. Vertex fitter
   - Propagate all trajectories to the initial vertex; maximise the compatibility between this vertex and all tracks using Minuit2 (floating x, y, z)

Illustration for vertex finding



Primary vertex reconstruction

- A package has been developed which takes “CompleteTracks” as its input and produces vertex as its output

1. Feed all tracks into the algorithm for now; selections based on such as chi2, d0/z0, etc. will be applied in here soon.

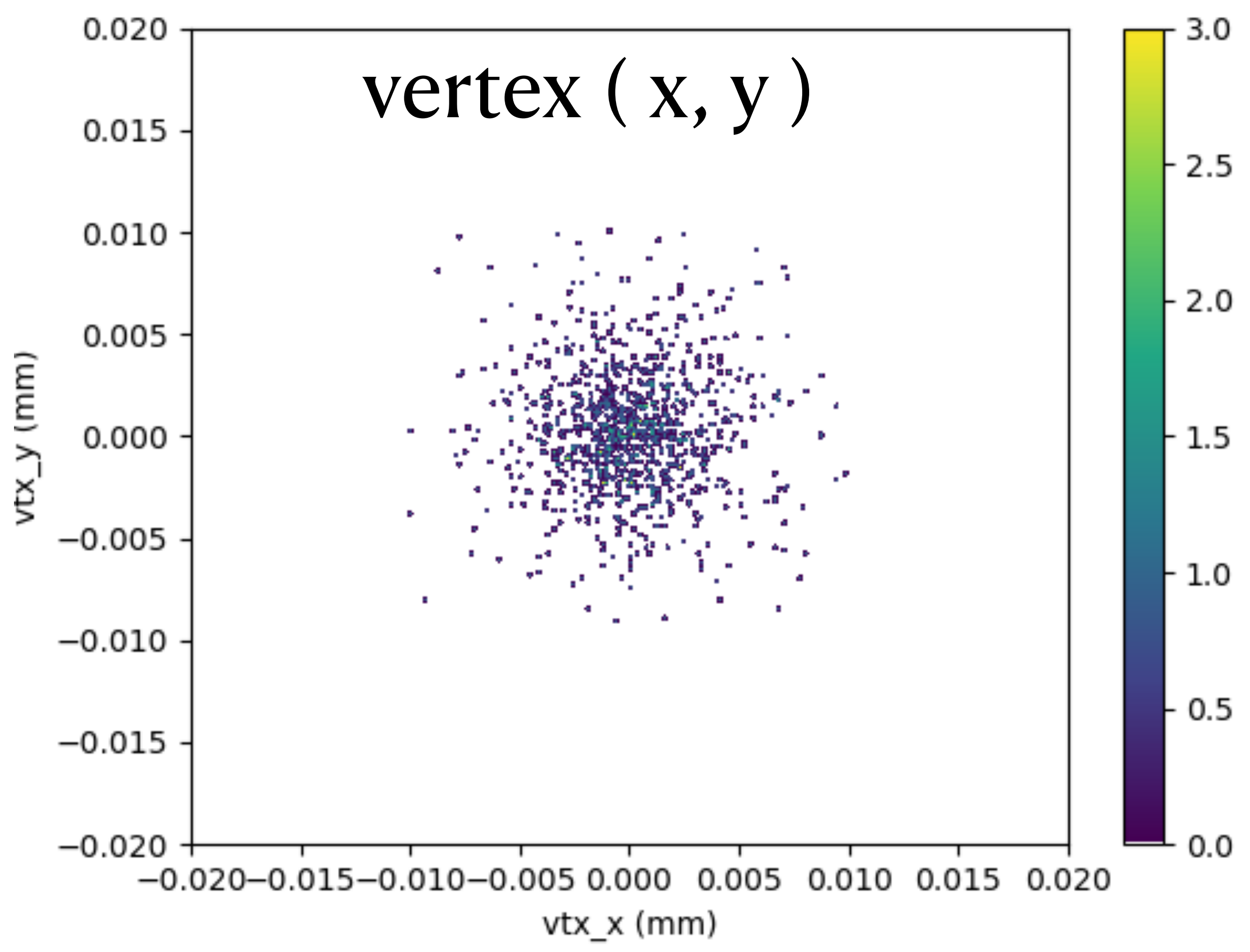
CEPCSW/Reconstruction/Vertexing/src/Vertexing.cpp
can be configured after FullTrackginAlg in SW

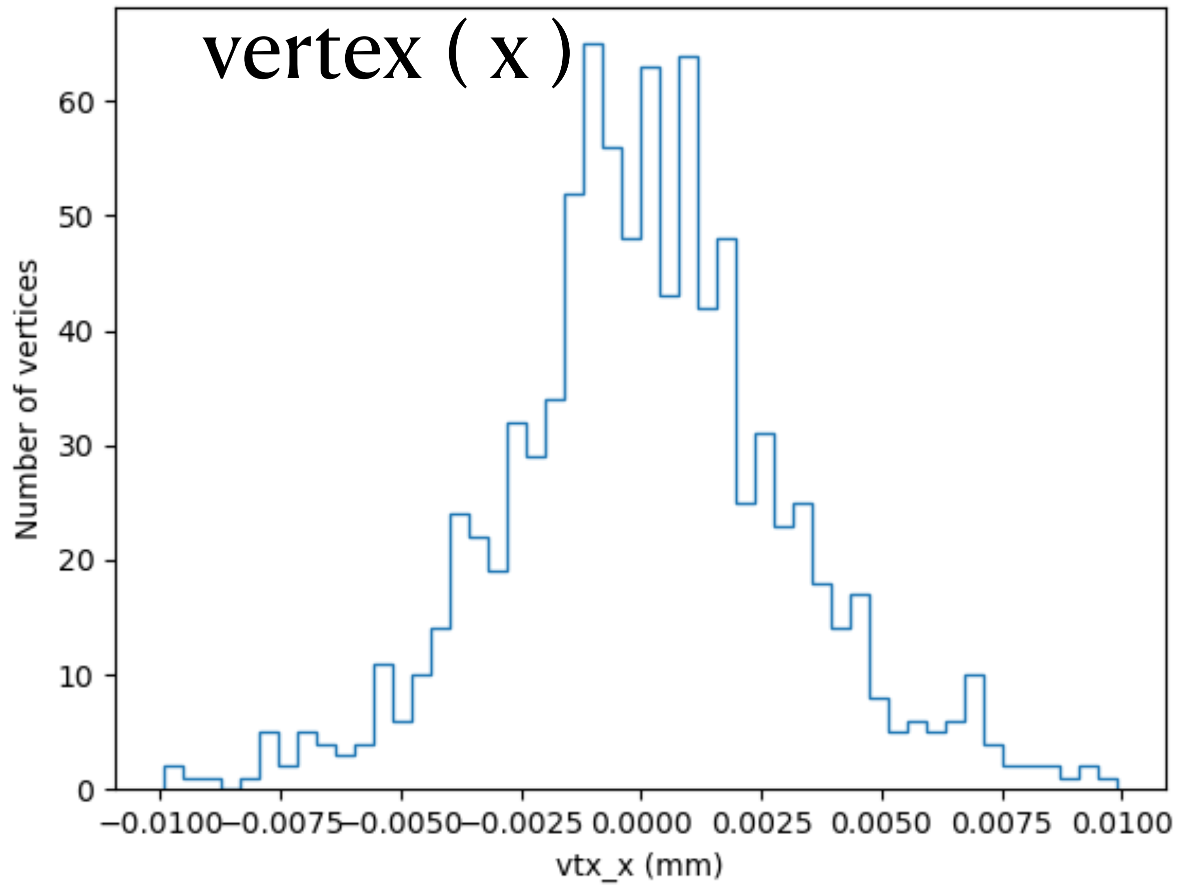
2. Vertex finder
   - For each track pair, determine its geometric centre and select the one with the highest compatibility with all other tracks as the initial vertex.
3. Vertex fitter
   - Propagate all trajectories to the initial vertex; maximise the compatibility between this vertex and all tracks using Minuit2 (floating x, y, z)

CEPCSW/Reconstruction/Vertexing/src/HelixPlus.cpp
inherits from made-ready HelixClass; need to import Minuit2 lib. from external

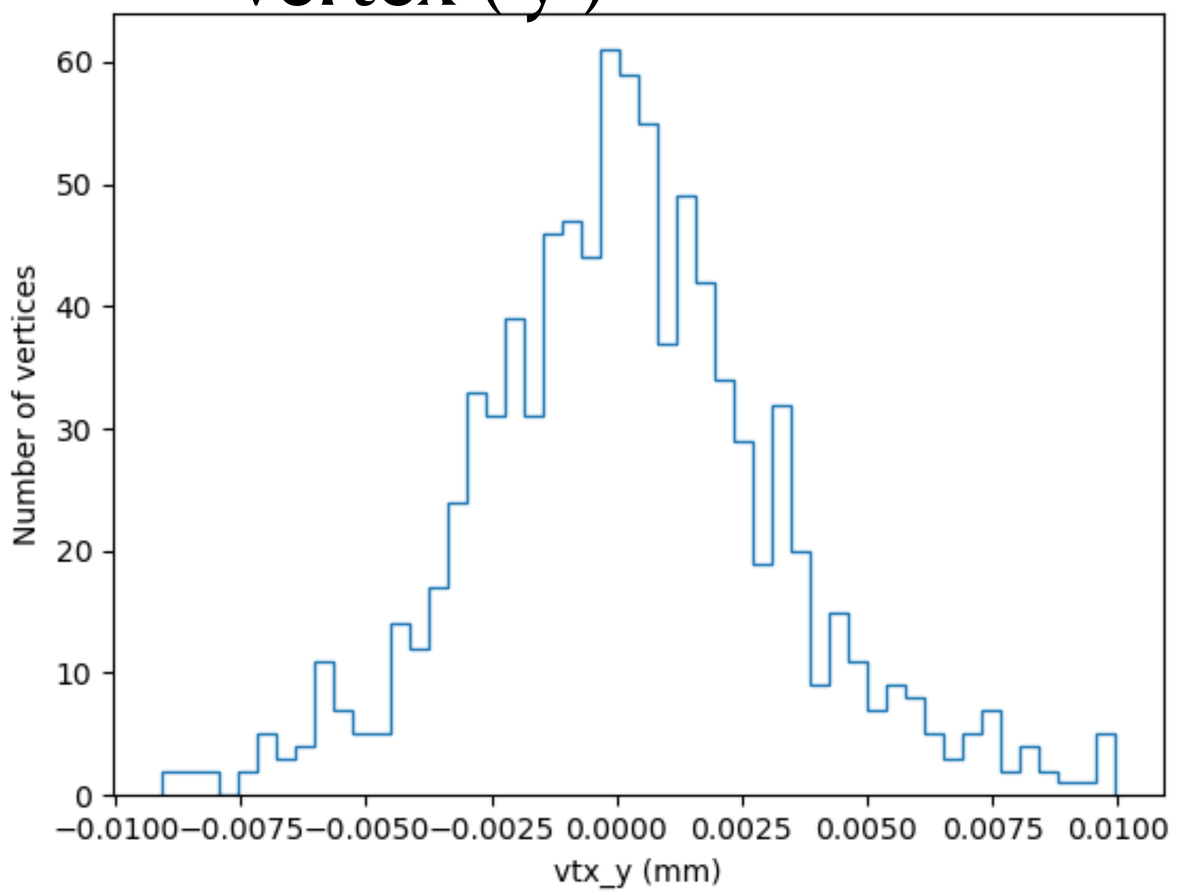
Performance (preliminary)

- ParticleGun for 1k mu-, mu+ pairs with $p = 1 \sim 100\text{GeV}, \theta = 35^\circ \sim 145^\circ$

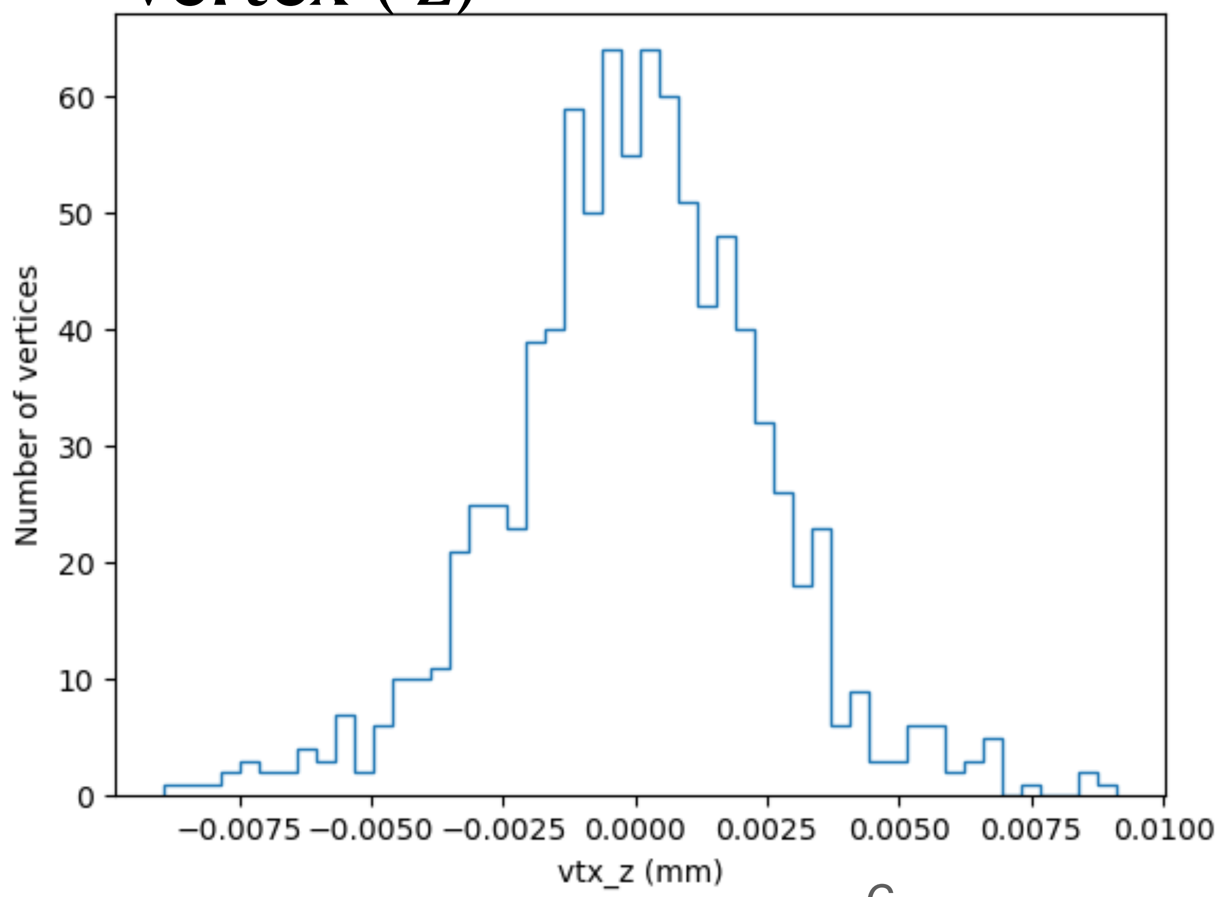

vertex (x, y)


vertex (x)

vertex (y)



vertex (z)



- Summary & To do
  - 1 sigma ~ 3 um seems reachable at first glance
  - Need to check the performance in more details
  - Connect to EDM4hep::vertex
  - Switch to secondary vertex reconstruction
  - A more sophisticated algorithm might be preferable in complex scenarios with jets