ACTS seed resolutions

Barak Schmookler

with help from Emma and Rey

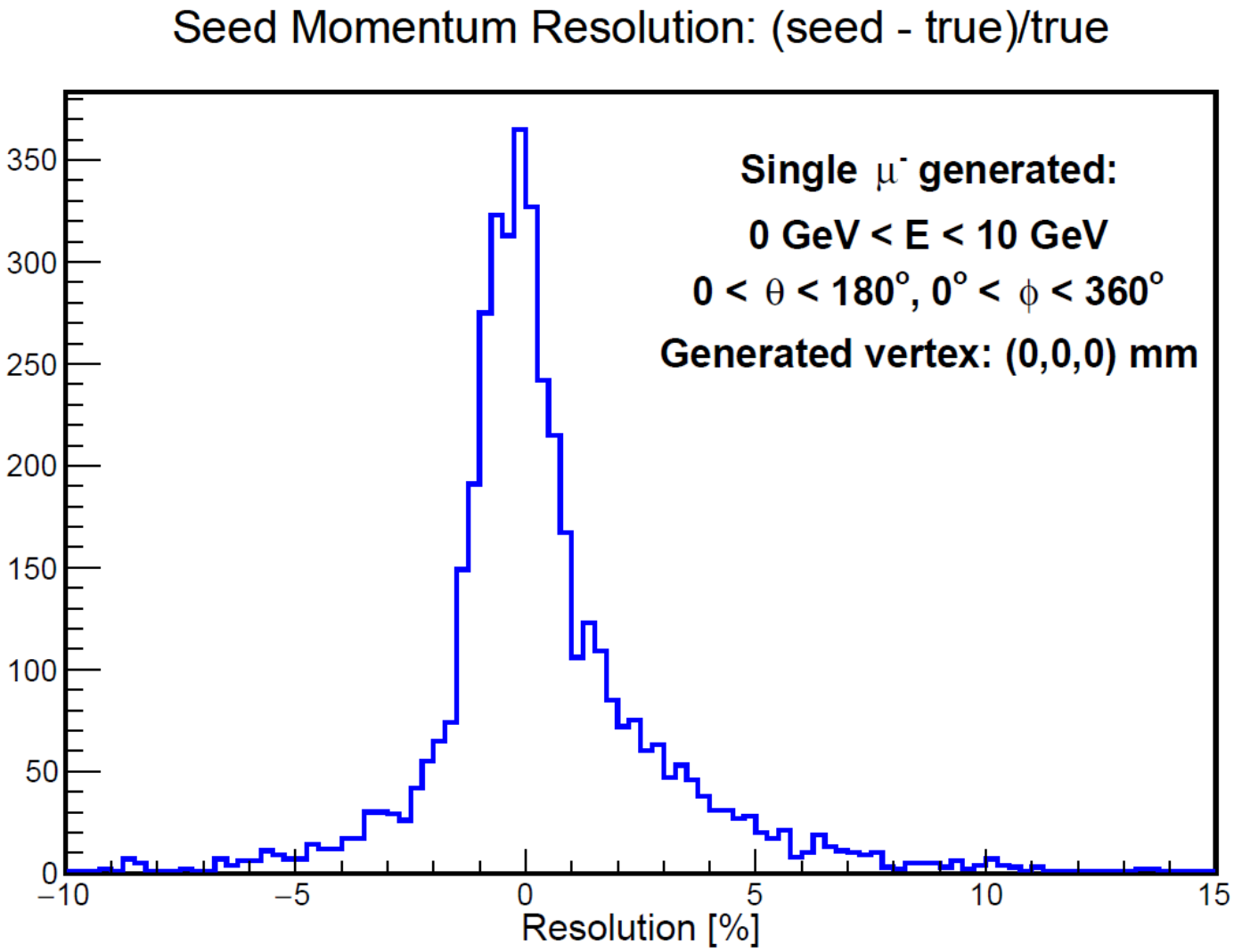
Motivation

- Reconstruction of the seed parameters is done in this file:

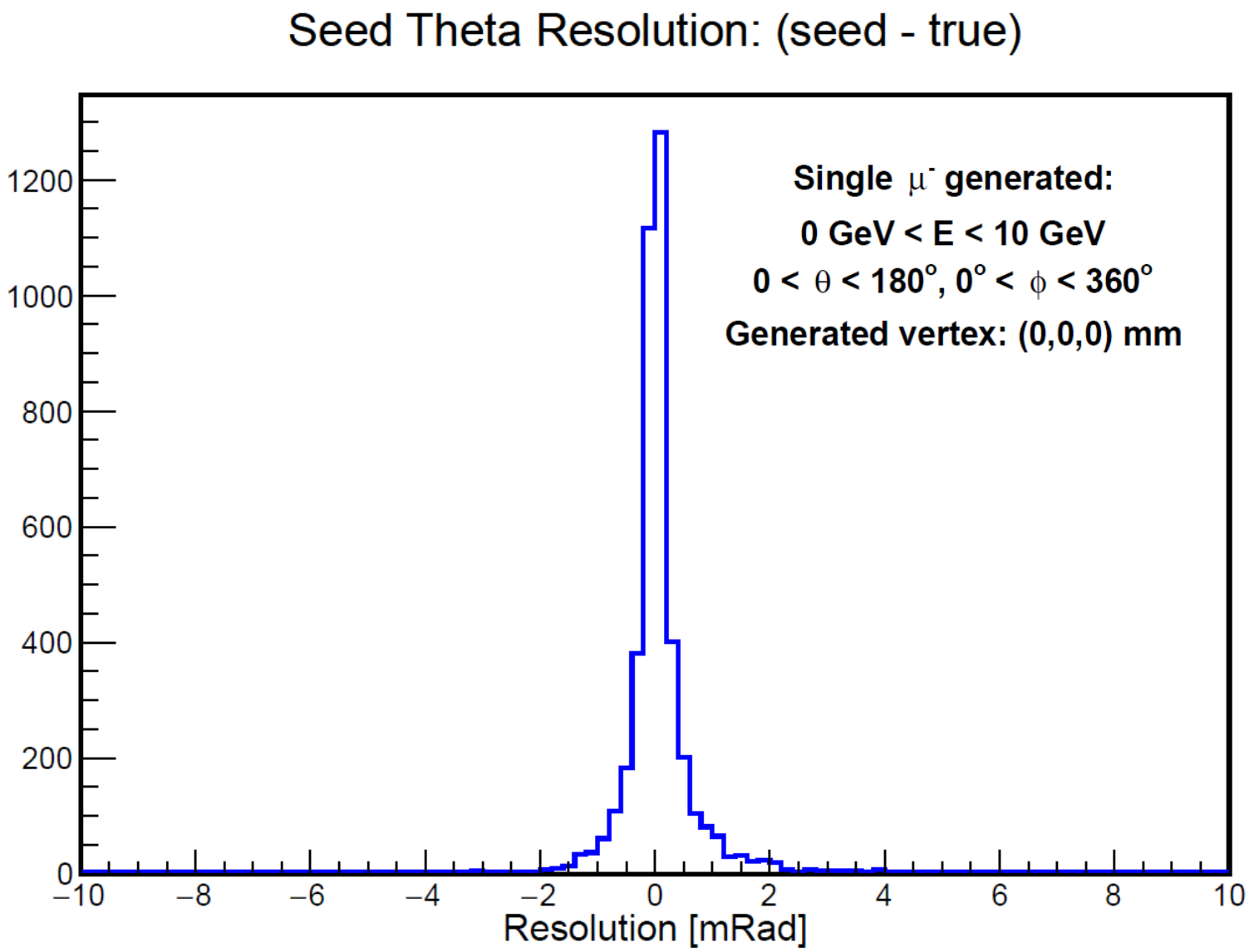
  https://github.com/eic/EICrecon/blob/main/src/algorithms/tracking/TrackSeeding.cc

- We want to compare the reconstructed seed parameters – momentum (q/p), theta, phi, ACTS positions (a,b) – to the generated particle.
- This will allow us to check if our seed reconstruction is reasonable, and it provide guidance for initial values for the CKF covariance matrix.

Momentum resolution



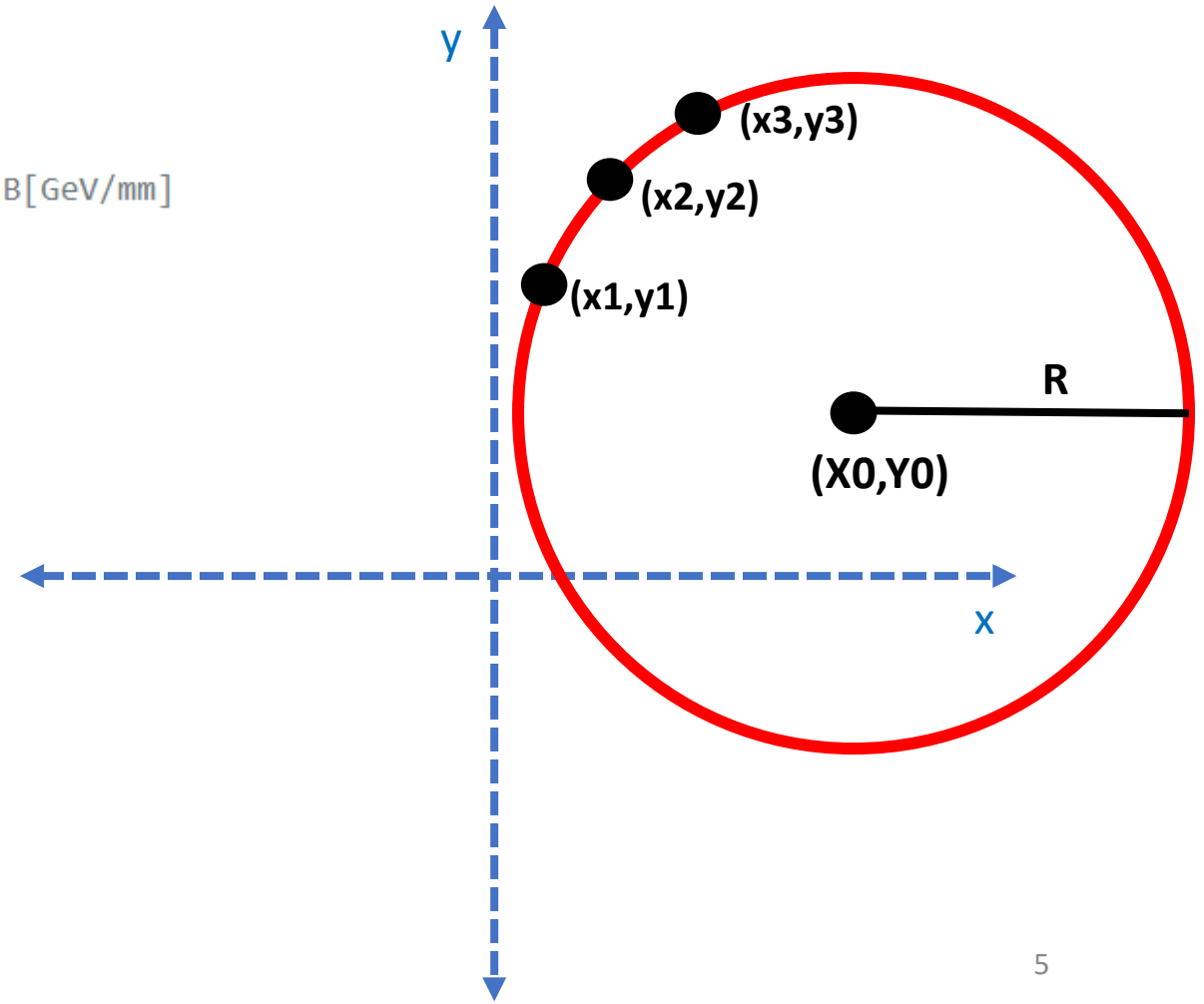
Theta resolution



How the seed (transverse) momentum is reconstructed

```
85        auto RX0Y0 = circleFit(xyHitPositions);
86        float R = std::get<0>(RX0Y0);
87        float X0 = std::get<1>(RX0Y0);
88        float Y0 = std::get<2>(RX0Y0);
97        float pt = R * m_cfg.m_bFieldInZ; // pt[GeV] = R[mm] * B[GeV/mm]
```

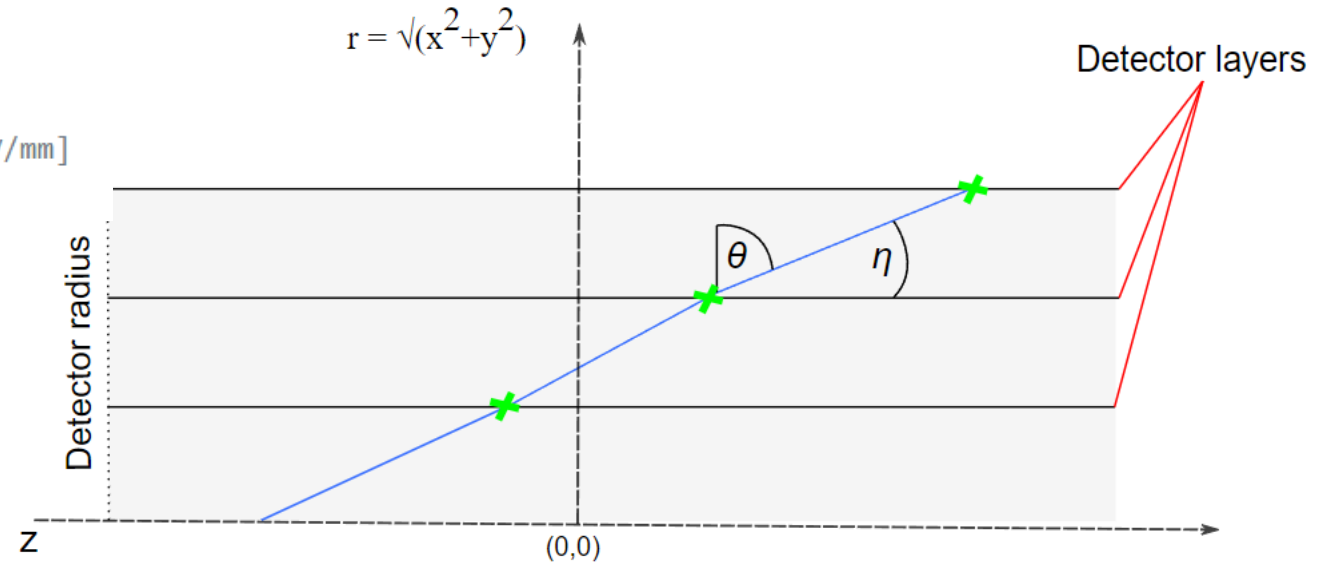
circleFit determines the circle radius and center in the (x,y) plane. The function can fit more than 3 points – so it may be more complex than necessary – but it seems to work well.



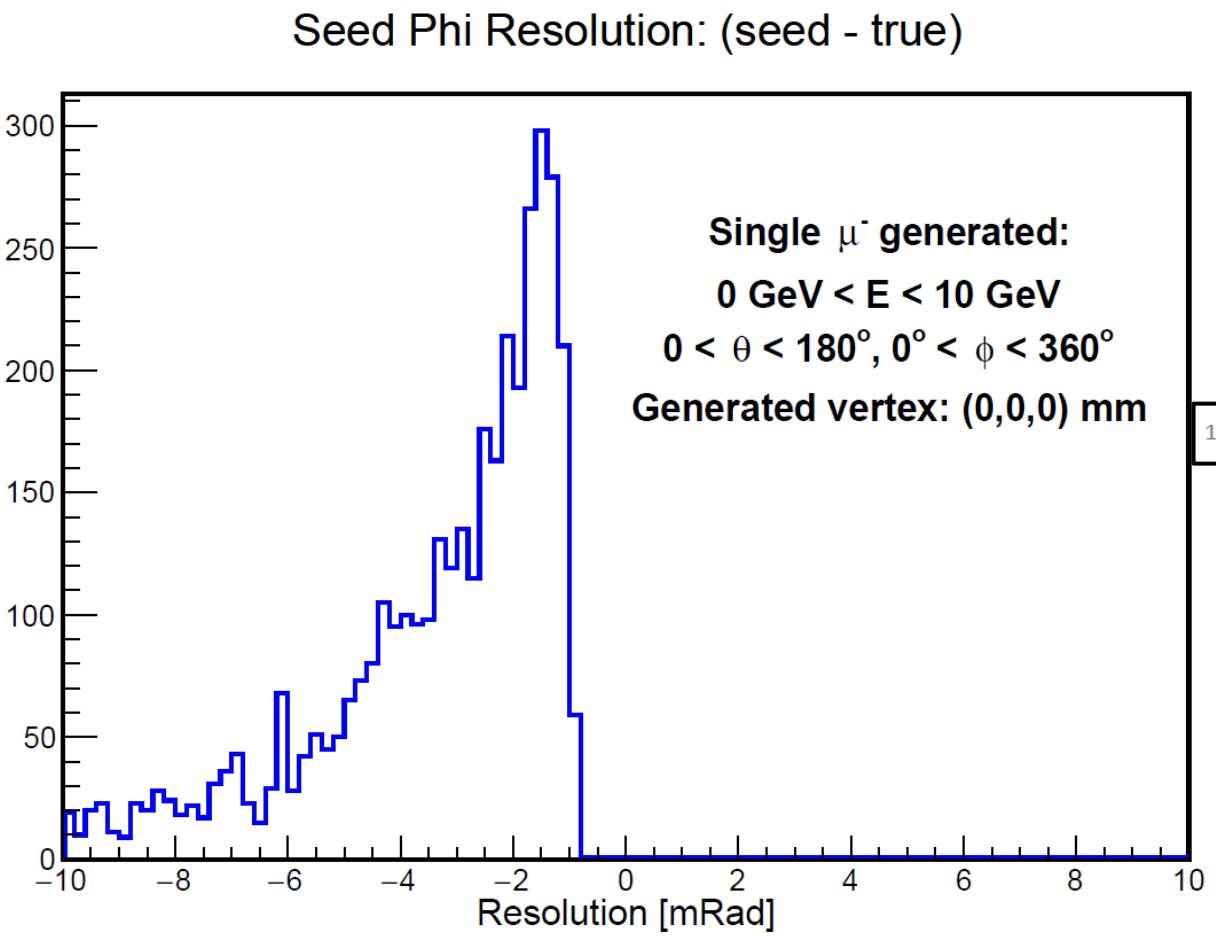
How the seed theta is reconstructed

```
89      auto slopeZ0 = lineFit(rzHitPositions);
92     float theta = atan(1./std::get<0>(slopeZ0));
93     // normalize to 0<theta<pi
94     if(theta < 0)
95       { theta += M_PI; }
96     float eta = -log(tan(theta/2.));
97     float pt = R * m_cfg.m_bFieldInZ; // pt[GeV] = R[mm] * B[GeV/mm]
98     float p = pt * cosh(eta);
```

lineFit makes a straight line fit of the three seed points in the r-z plane to determine theta.



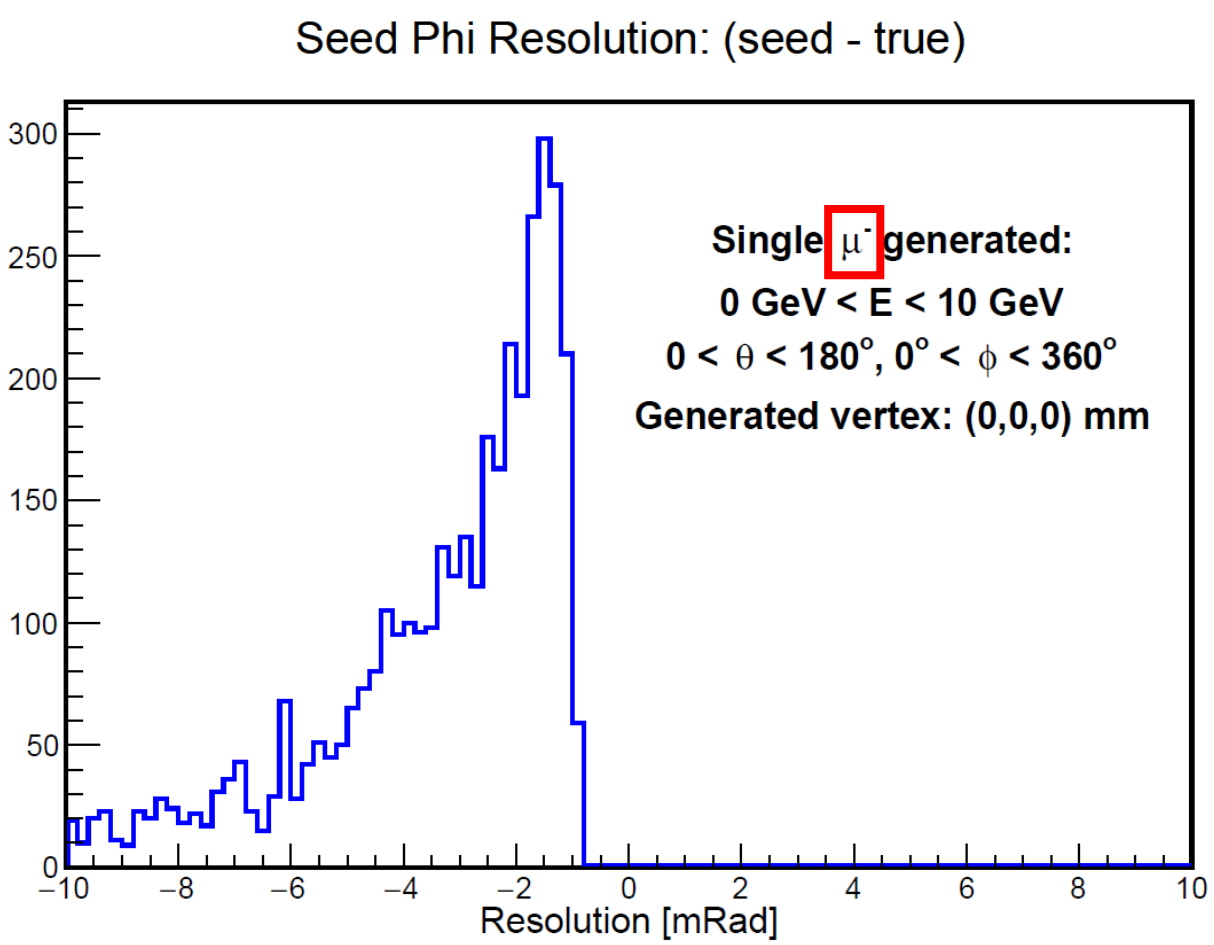
Seed phi reconstruction

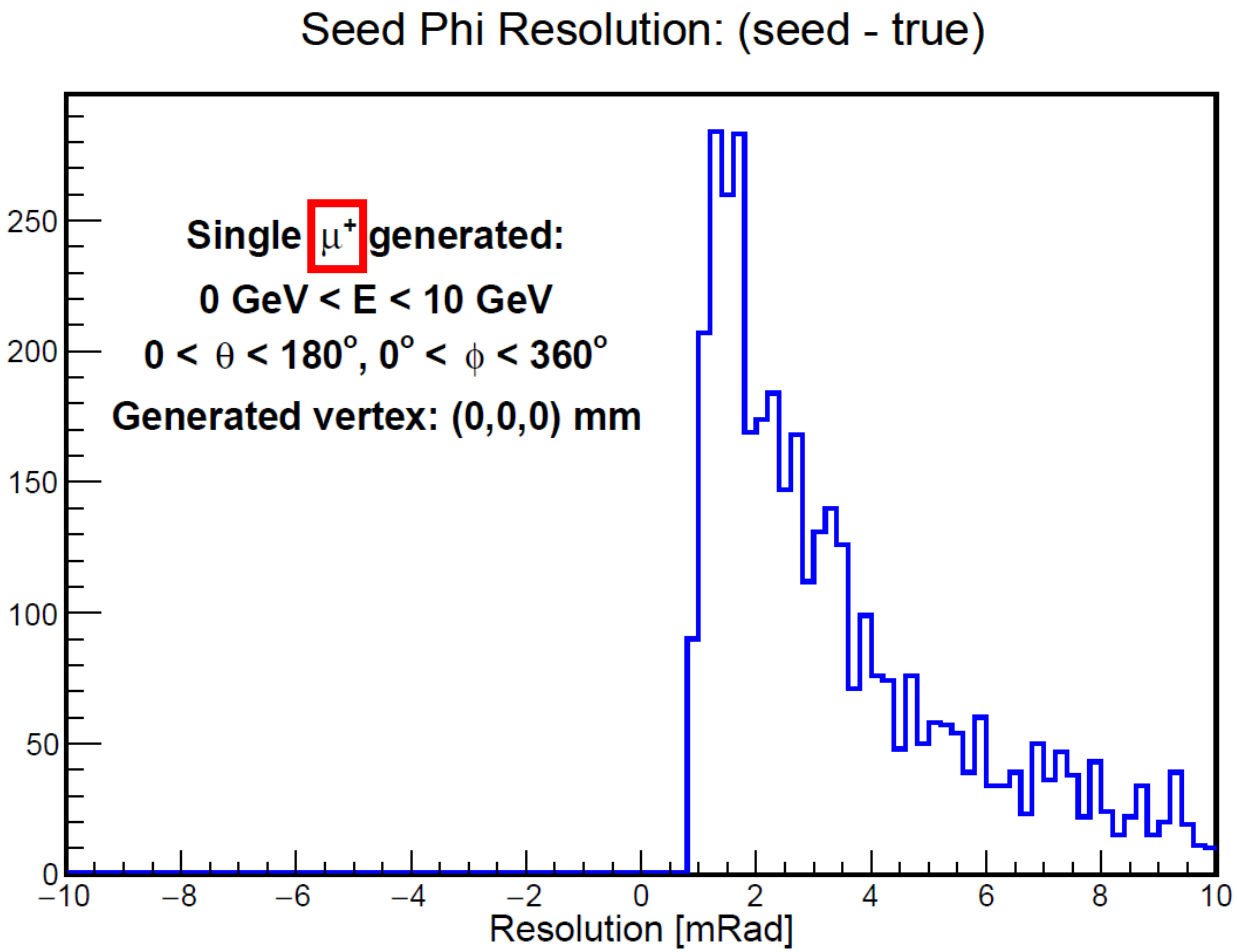


The seed phi is calculated based on position of first seed point.

```
120         atan2(xyHitPositions.at(0).second, xyHitPositions.at(0).first), // phi of first hit (rad)
```

Seed phi reconstruction





See 'reversed' effect for positive muons, as expected.

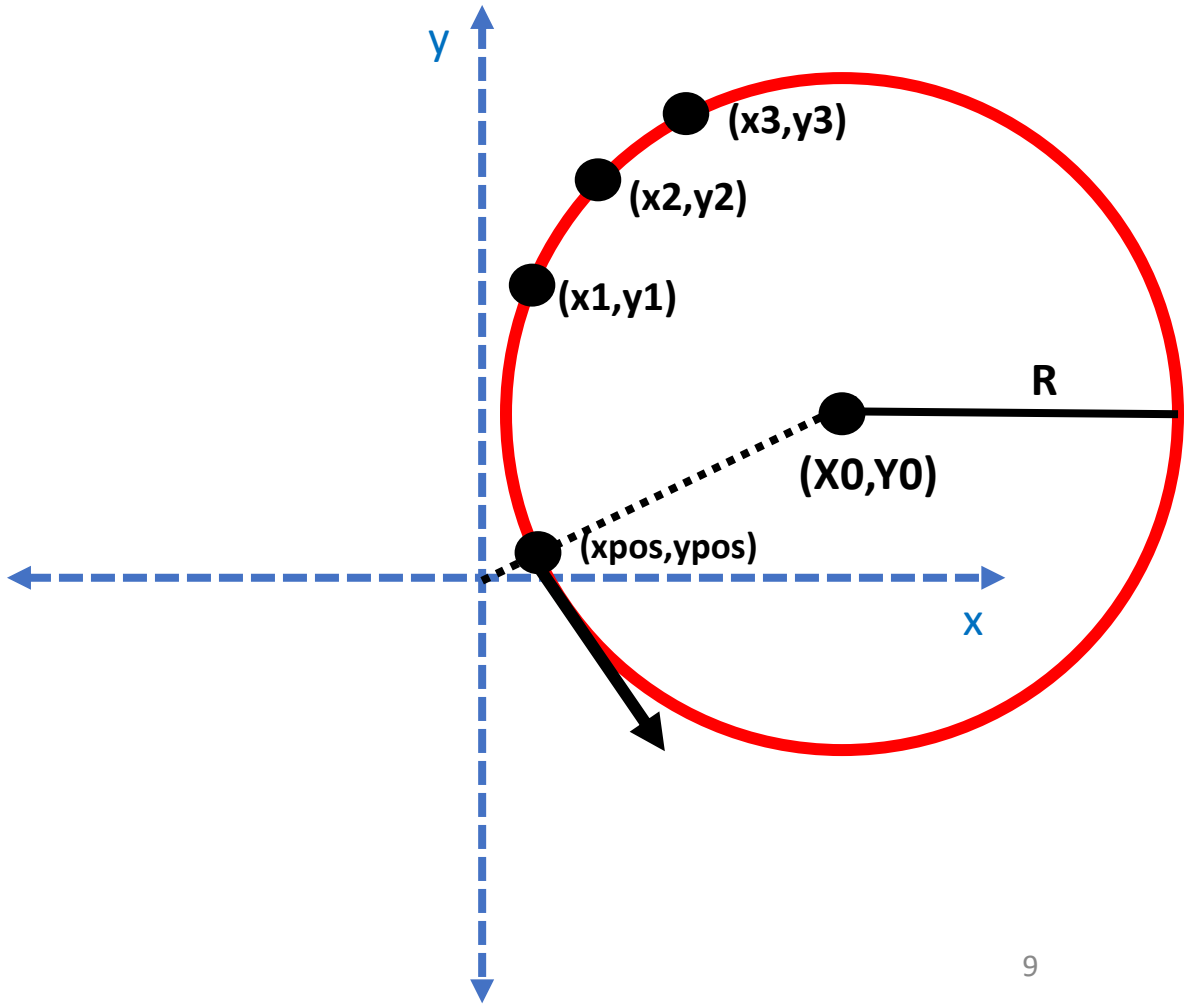
How to reconstruct the correct seed phi

```
85          auto RX0Y0 = circleFit(xyHitPositions);
86          float R = std::get<0>(RX0Y0);
87          float X0 = std::get<1>(RX0Y0);
88          float Y0 = std::get<2>(RX0Y0);
101         const auto xypos = findRoot(RX0Y0);
```

findRoot finds the point of closest approach on the circle to the origin – (xpos,ypos).

Geometry ‘fun fact’: the line from the origin to (xpos,ypos) also passes through the center of the circle.

So, we need to find the angle of the vector going through (xpos,ypos) that is tangential to the circle.



How to reconstruct the correct seed phi

```
//Calculate phi at xypos
auto xpos = xypos.first;
auto ypos = xypos.second;

auto vxpos = -1.*charge*(ypos-Y0);
auto vypos = charge*(xpos-X0);

auto phi = atan2(vypos,vxpos);
```

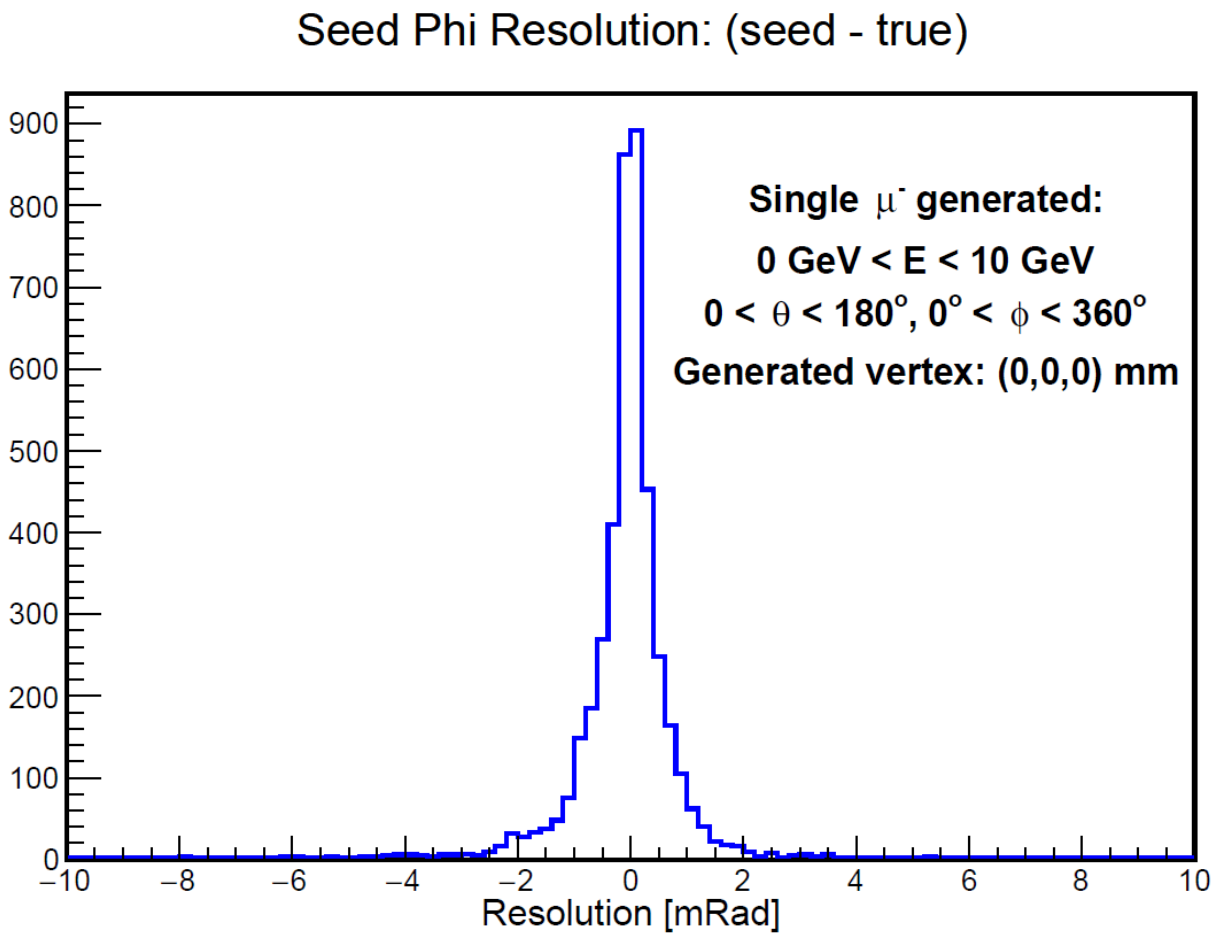
Vector from center of circle to point closest to origin:

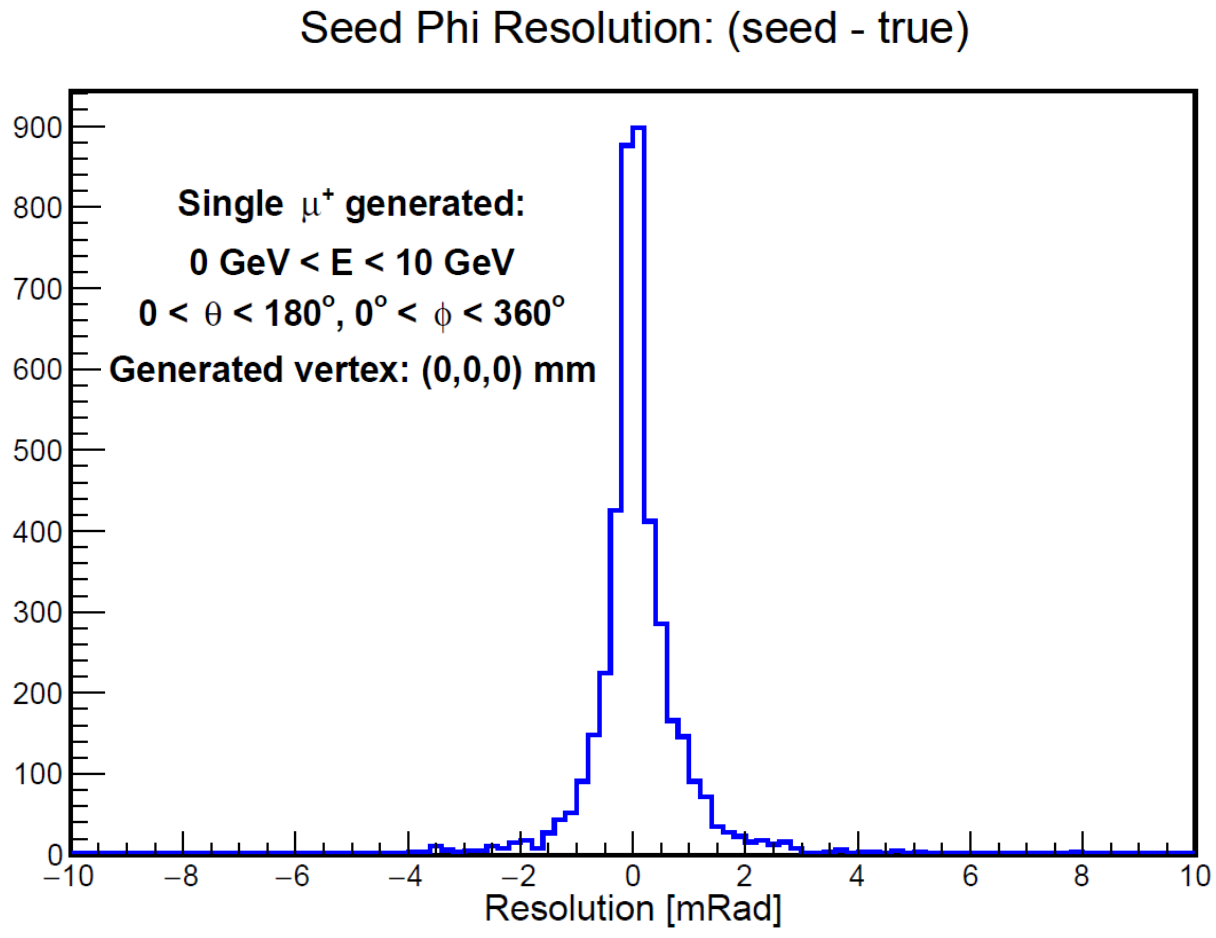
$$< (xpos\ - X0), (ypos\ - YO >$$

Vector tangential to circle at point closest to origin:

$$\pm < (ypos - Y0), -(xpos - X0) >$$

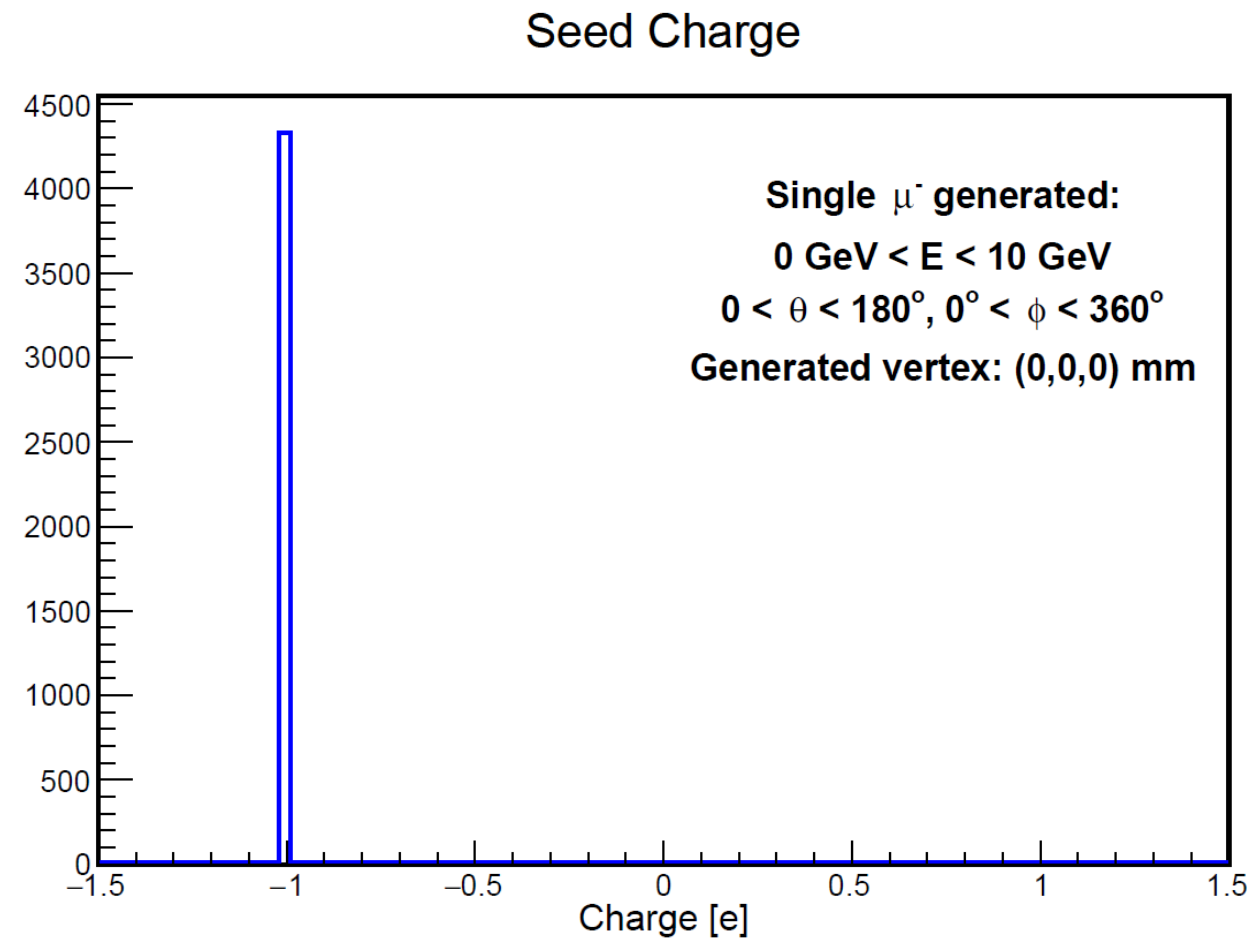
Seed phi reconstruction after fix

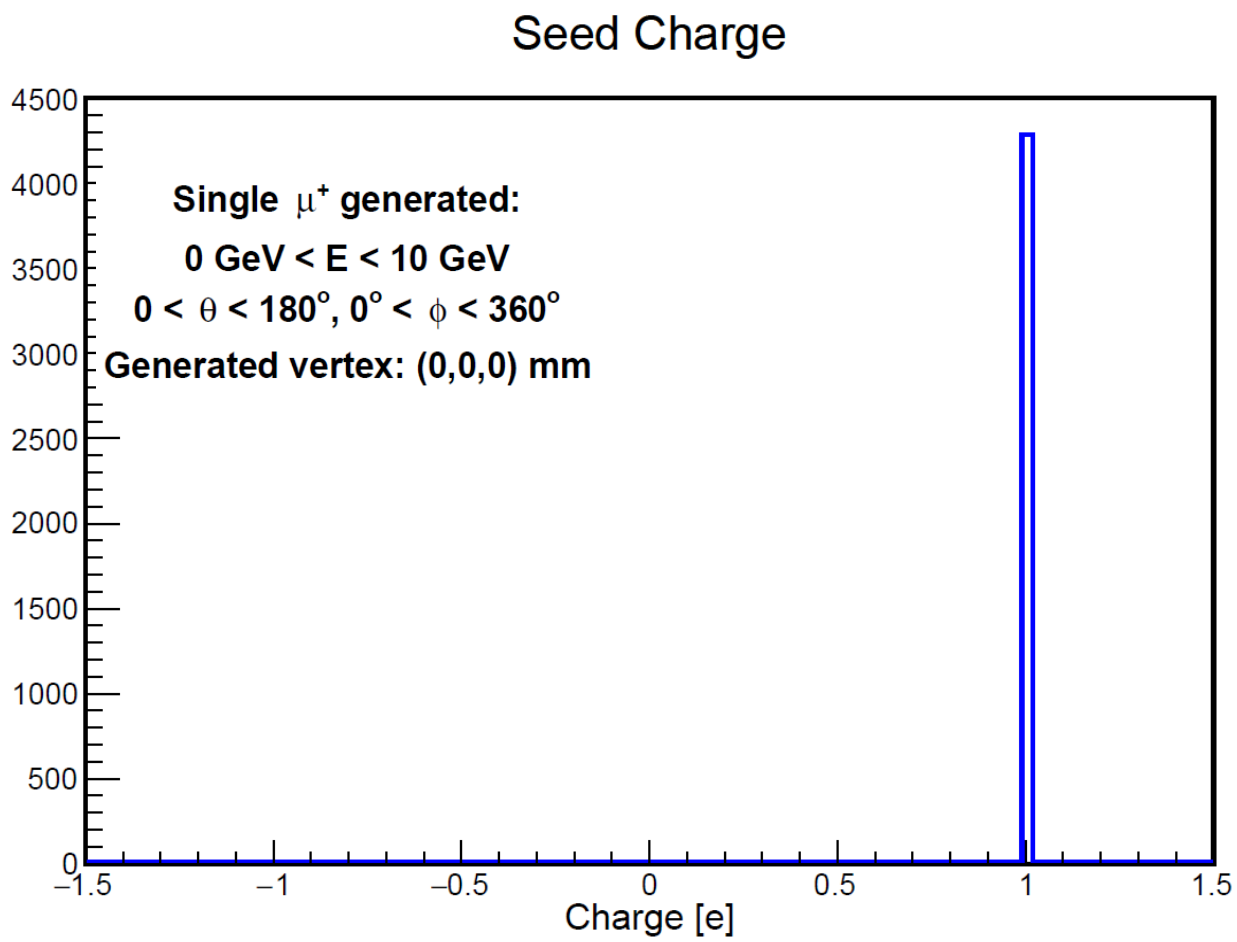




Looks good for both positive and negative particles!

Seed charge reconstruction

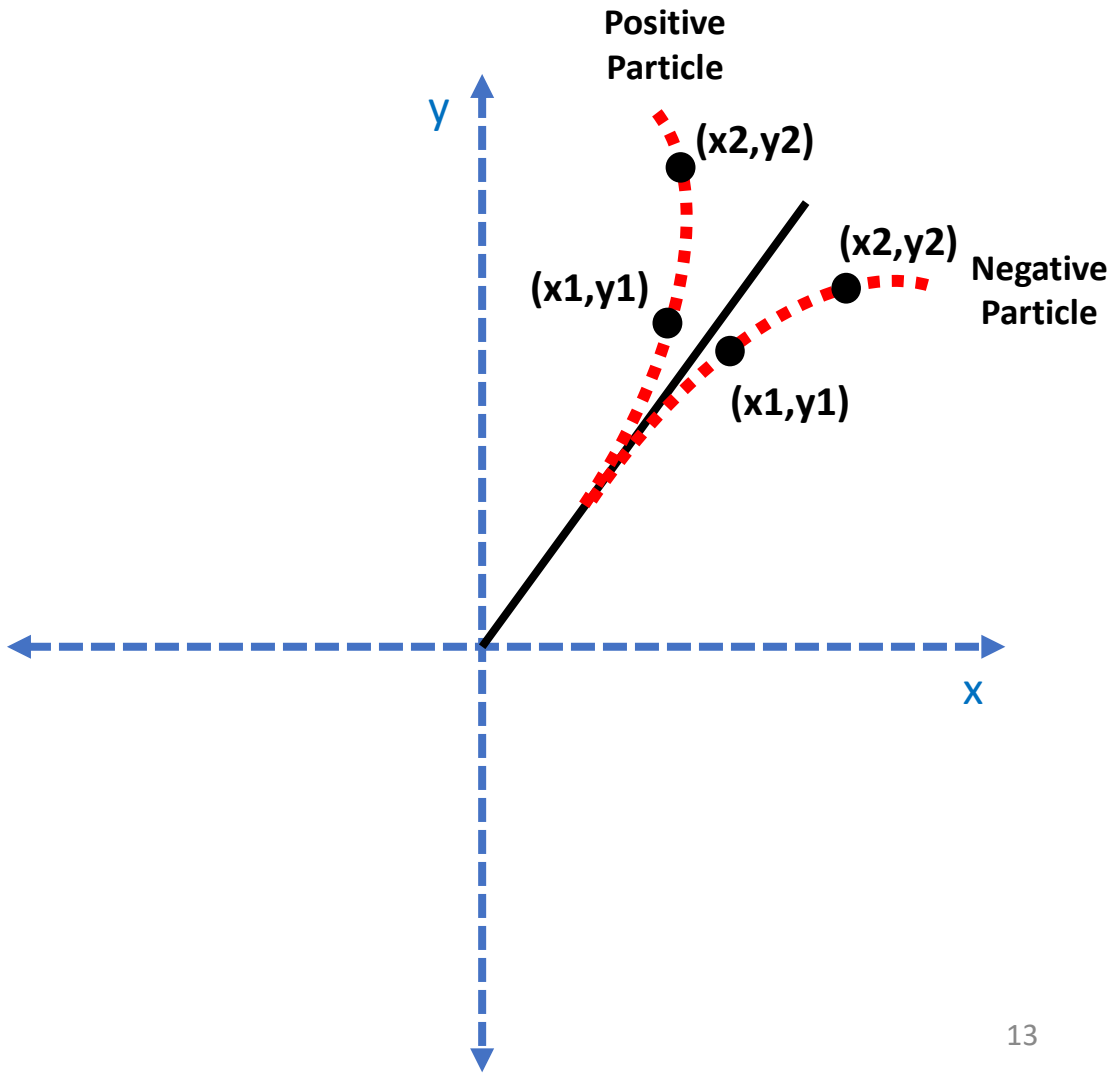




How the seed charge is reconstructed

```
91          int charge = determineCharge(xyHitPositions);
```

determineCharge compares the first 2 seed hits and considers which way they ‘fall off’ a line.



Next steps

- Perform similar analysis for the reconstruction of the seed vertex. This should be done by generating the primary particle both with (x,y,z) = (0,0,0), as was done above, as well as with non-zero z vertex positions. Emma and I can work together on this.
- This study was done with the default set of seed parameters. We should update the configuration file with the most up-to-date seed parameters from Emma and Rey.