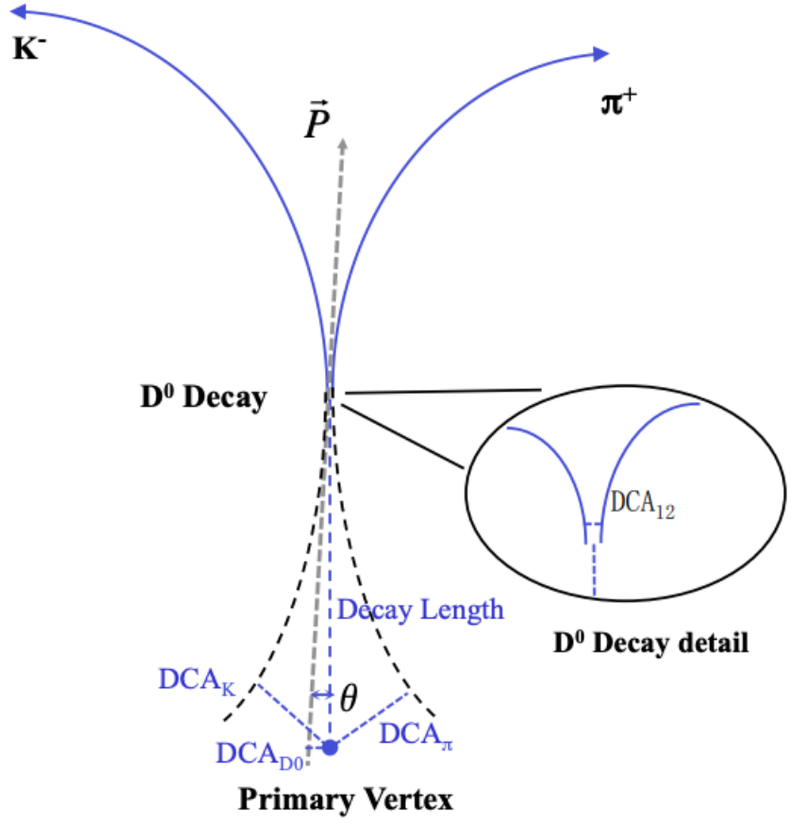
Helix and SecondaryVertices Factory in ElCrecon

Xin Dong



Thanks to:

Bishoy Dongwi, Lokesh Kumar, Shyam Kumar, Rongrong Ma, Joe Osborn, Ashish Pandav, *Harsimran Singh*, *Khushi Singla*, Deepa Thomas, Connie Yang etc.

Previous presentations:
https://indico.bnl.gov/event/29424/contributions/112551/attachments/64322/110448/20250825_ePIC_Helix.pdf

Vertex Finders

ACTS Track Objects

ReconstructedParticles

Vertex Finder(s) (Primary)

(Secondary)

Acts-based AMVF Finder
- PR # 1915 - *Bishoy*

CentralTrackVertices

ACTS objs
PODIO output
EICrecon ready
Under development

2025/08/25 Proposal

User Analysis Vertex Finders (Helix/KFParticle)

a. Helix swimming (adopted from STAR)
   - helix utils to edm4eic - PR # 115
b. KFParticle
   - used by STAR/sPHENIX/ALICE/CBM etc.
   - under testing now, to be deployed later

Implementation Change

2025/10/20 Proposal

EICrecon

reco/SecondaryVertices (Helix/KFParticle)

a. reco/Helix.*
b. KFParticle standalone package working on integration

SecondaryVerticesXXX

EICrecon: PR #2144

- src/algorithms/reco/Helix.cc
- src/algorithms/reco/Helix.h

adopted from STAR code, adjusted for EICrecon

```
/// ReconstructParticle, b field
Helix(const edm4eic::ReconstructedParticle& p, const double b_field);
```

- src/algorithms/reco/SecondaryVerticesHelix.cc
- src/algorithms/reco/SecondaryVerticesHelix.h
- src/algorithms/reco/SecondaryVerticesHelixConfig.h
- src/factories/reco/SecondaryVerticesHelix_factory.h

1. new SecondaryVertices factory, now based on Helix, can be expanded for others (e.g. KFParticle)

2. take ReconstructedParticle as input, run secondary finder (Ks included so far, can be expanded to include others)

3. output -> SecondaryVerticesHelix (edm4eic::Vertex) in PODIO
   - *proposed to add edm4eic::SecondaryVertex container (next slide)*

4. geometric cut variables controlled by SecondaryVerticesHelixConfig.h

- src/global/reco/reco.cc
- src/services/io/podio/JEventProcessorPODIO.cc

SecondaryVerticesHelix Factory

initial config file

```
struct SecondaryVerticesHelixConfig {

  float b_field     = -1.7 * dd4hep::tesla;
  bool unlikesign   = true;
  float minDca      = 0.03 * edm4eic::unit::mm; // mm, daughter to pVtx
  float maxDca12    = 1. * edm4eic::unit::mm;   // mm, dca between daughter 1 and 2
  float maxDca      = 1. * edm4eic::unit::mm;   // mm, dca of V0 to pVtx
  float minCostheta = 0.8; // costheta, theta: angle of V0 decay direction and momentum
};
```

```
if ((*rcvtx).size() == 0) {
  info(" No primary vertex in this event! Skip secondary vertex finder!");
  return;
}
const auto pVtxPos4f = (*rcvtx)[0].getPosition();
// convert to cm
edm4hep::Vector3f pVtxPos(pVtxPos4f.x * edm4eic::unit::mm / edm4eic::unit::cm,
                          pVtxPos4f.y * edm4eic::unit::mm / edm4eic::unit::cm,
                          pVtxPos4f.z * edm4eic::unit::mm / edm4eic::unit::cm);
info("\t Primary vertex = ({},{},{})cm \t b field = {} tesla", pVtxPos.x, pVtxPos.y, pVtxPos.z,
     m_cfg.b_field / dd4hep::tesla);

std::vector<Helix> hVec;
hVec.clear();
std::vector<unsigned int> indexVec;
indexVec.clear();
for (unsigned int i = 0; const auto& p : *rcparts
  if (p.getCharge() == 0)
    continue;
  Helix h(p, m_cfg.b_field);
  double dca = h.distance(pVtxPos) * edm4eic::uni
  if (dca < m_cfg.minDca)
    continue;

  hVec.push_back(h);
  indexVec.push_back(i);
  ++i;
}

if (hVec.size() != indexVec.size())
  return;

debug("\t Vector size {}, {}", hVec.size(), index
```

```
for (unsigned int i1 = 0; i1 < hVec.size(); ++i1) {
  for (unsigned int i2 = i1 + 1; i2 < hVec.size(); ++i2) {
    const auto& p1 = (*rcparts)[indexVec[i1]];
    const auto& p2 = (*rcparts)[indexVec[i2]];

    if (!(m_cfg.unlikesign && p1.getCharge() + p2.getCharge
      continue;

    const auto& h1 = hVec[i1];
    const auto& h2 = hVec[i2];

// Helix function uses cm unit
double dca1 = h1.distance(pVtxPos) * edm4eic::unit::cm;
double dca2 = h2.distance(pVtxPos) * edm4eic::unit::cm;
if (dca1 < m_cfg.minDca || dca2 < m_cfg.minDca)
  continue;

std::pair<double, double> const ss = h1.pathLengths(h2)
edm4hep::Vector3f h1AtDcaTo2       = h1.at(ss.first);
edm4hep::Vector3f h2AtDcaTo1       = h2.at(ss.second);

double dca12 = edm4hep::utils::magnitude(h1AtDcaTo2 - h
if (std::isnan(dca12))
  continue;
if (dca12 > m_cfg.maxDca12)
  continue;
edm4hep::Vector3f pairPos = 0.5 * (h1AtDcaTo2 + h2AtDca

edm4hep::Vector3f h1MomAtDca = h1.momentumAt(ss.first,
edm4hep::Vector3f h2MomAtDca = h2.momentumAt(ss.second,
edm4hep::Vector3f pairMom    = h1MomAtDca + h2MomAtDca;
```

```
double e1 =
    std::hypot(edm4hep::utils::magnitude(h1MomAtDca), particleSvc.particle(p1.getPDG()).mass);
double e2 =
    std::hypot(edm4hep::utils::magnitude(h2MomAtDca), particleSvc.particle(p2.getPDG()).mass);
double pairE = e1 + e2;
//      double pairP = edm4hep::utils::magnitude(pairMom);

//      double m_inv2 = pairE * pairE - pairP * pairP;
//      double m_inv  = (m_inv2 > 0) ? sqrt(m_inv2) : 0.;
double angle = edm4hep::utils::angleBetween(pairMom, pairPos - pVtxPos);
if (cos(angle) < m_cfg.minCostheta)
  continue;

double beta = edm4hep::utils::magnitude(pairMom) / pairE;
double time = edm4hep::utils::magnitude(pairPos - pVtxPos) / (beta * dd4hep::c_light);
edm4hep::Vector3f dL = pairPos - pVtxPos; // in cm
edm4hep::Vector3f decayL(dL.x * edm4eic::unit::cm, dL.y * edm4eic::unit::cm,
                         dL.z * edm4eic::unit::cm);
double dca2pv = edm4hep::utils::magnitude(decayL) * sin(angle);
if (dca2pv > m_cfg.maxDca)
  continue;

auto v0 = out_secondary_vertices->create();
v0.setType(2); // 2 for secondary
v0.setPosition({(float)(pairPos.x * edm4eic::unit::cm / edm4eic::unit::mm),
                (float)(pairPos.y * edm4eic::unit::cm / edm4eic::unit::mm),
                (float)(pairPos.z * edm4eic::unit::cm / edm4eic::unit::mm), (float)time});
v0.addToAssociatedParticles(p1);
v0.addToAssociatedParticles(p2);

info("One secondary vertex found at (x,y,z) = ({}, {}, {}) mm.",
     pairPos.x * edm4eic::unit::cm / edm4eic::unit::mm,
     pairPos.y * edm4eic::unit::cm / edm4eic::unit::mm,
     pairPos.x * edm4eic::unit::cm / edm4eic::unit::mm);
```

SecondaryVertex to edm4eic: add-secondaryvertex branch

1. Current edm4eic::vertex object doesn't contain many topological variables for secondary vertices.
2. Though one can in principle re-calculate all these based on the associated ReconstructedParticle, it will be much more convenient to save them during reconstruction and the down-stream analysis can directly use them for physics analysis and also this will avoid repeated calculations.

```
edm4eic::Vertex:
  Description: "EIC vertex"
  Author: "J. Osborn"
  Members:
    - int32_t             type          // Type flag, to ide
    - float               chi2          // Chi-squared of th
    - int                 ndf           // NDF of the vertex
    - edm4hep::Vector4f   position      // position [mm] + t
    ## this is named "covMatrix" in EDM4hep, renamed for con
    - edm4eic::Cov4f      positionError // Covariance matrix
  OneToManyRelations:
    - edm4eic::ReconstructedParticle associatedParticles //
```

```
edm4hep::Vertex:
  Description: "Vertex"
  Author: "EDM4hep authors"
  Members:
    - uint32_t            type          // flagword that defines t
    - float               chi2          // chi-squared of the vert
    - int32_t             ndf           // number of degrees of fr
    - edm4hep::Vector3f   position [mm]     //  position of the ve
    - edm4hep::CovMatrix3f covMatrix [mm^2]  // covariance matrix
    - int32_t                 algorithmType // type code for the a
  VectorMembers:
    - float               parameters    // additional parameters r
  OneToManyRelations:
    - edm4hep::ReconstructedParticle particles // particles that h
```

Propose to add edm4eic::SecondaryVertex container

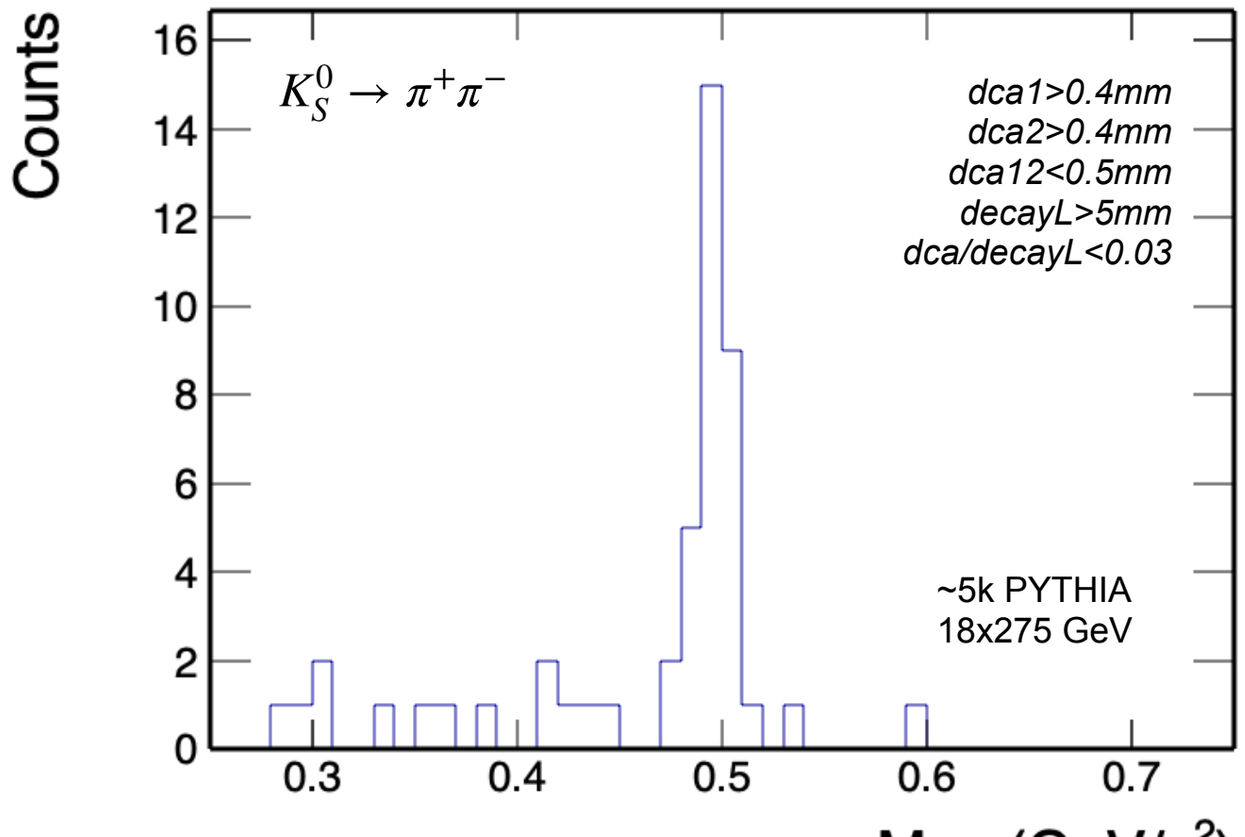
```
edm4eic::SecondaryVertex:
  Description: "EIC secondary vertex"
  Author: "X. Dong"
  Members:
    - int32_t             type          // Type flag, to identify what type of vertex it is (e.g.
    - float               chi2          // Chi-squared of the vertex fit
    - int                 ndf           // NDF of the vertex fit
    - edm4hep::Vector4f   position      // position [mm] + time t0 [ns] of the vertex. Time is 4th
    - edm4eic::Cov4f      positionError // Covariance matrix of the position+time. Time is 4th cor
    - edm4hep::Vector3f   parentMomentum        // parent momentum
    - edm4hep::Vector3f   parentDecayLength     // parent decay length L
    - float               parentInvariantMass   // parent invariant mass
    - float               parentDecayLengthChi2 // parent L/dL
    - float               parentDca2PV          // parent dca to primary vertex
    - float               parentDca2PVChi2      // parent dca/sigma to primary vertex
  VectorMembers:
    - edm4hep::Vector3f   daughterMomentum   // daughter track momentum
    - float               daughterMass       // daughter mass
    - float               daughterDca2PV     // daughter dca to primary vertex
    - float               daughterDca2PVChi2 // daughter dca/sigma to primary vertex
    - int                 daughterPairIndices    // pair track indices
    - float               daughterPairDca        // pair dca to primary vertex
    - float               daughterPairDcaChi2    // pair dca/sigma to primary vertex
  OneToOneRelations:
    - edm4eic::Vertex     primaryVertex      // associated primary vertex
  OneToManyRelations:
    - edm4eic::ReconstructedParticle associatedParticles // particles associated to this vertex.
```

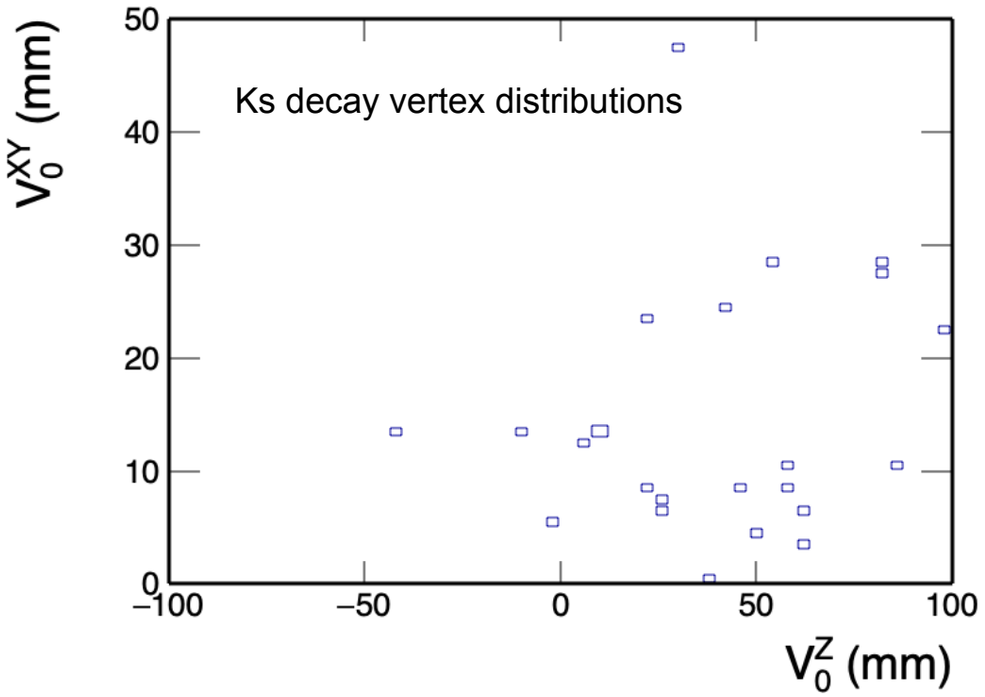
New structure has been integrated to EICrecon locally and tested to work well!
A new brunch add-secondaryvertex on edm4eic repo.

Test Output

PODIO output

- ScatteredElectronsTruth_objIdx
- SecondaryVerticesHelix
  - SecondaryVerticesHelix.type
  - SecondaryVerticesHelix.chi2
  - SecondaryVerticesHelix.ndf
  - SecondaryVerticesHelix.position.x
  - SecondaryVerticesHelix.position.y
  - SecondaryVerticesHelix.position.z
  - SecondaryVerticesHelix.position.t
  - SecondaryVerticesHelix.positionError.xx
  - SecondaryVerticesHelix.positionError.yy
  - SecondaryVerticesHelix.positionError.zz
  - SecondaryVerticesHelix.positionError.tt
  - SecondaryVerticesHelix.positionError.xy
  - SecondaryVerticesHelix.positionError.xz
  - SecondaryVerticesHelix.positionError.xt
  - SecondaryVerticesHelix.positionError.yz
  - SecondaryVerticesHelix.positionError.yt
  - SecondaryVerticesHelix.positionError.zt
  - SecondaryVerticesHelix.parentMomentum.x
  - SecondaryVerticesHelix.parentMomentum.y
  - SecondaryVerticesHelix.parentMomentum.z
  - SecondaryVerticesHelix.parentDecayLength.x
  - SecondaryVerticesHelix.parentDecayLength.y
  - SecondaryVerticesHelix.parentDecayLength.z
  - SecondaryVerticesHelix.parentInvariantMass
  - SecondaryVerticesHelix.parentDecayLengthChi2
  - SecondaryVerticesHelix.parentDca2PV
  - SecondaryVerticesHelix.parentDca2PVChi2
  - SecondaryVerticesHelix.daughterMomentum_begin
  - SecondaryVerticesHelix.daughterMomentum_end
  - SecondaryVerticesHelix.daughterMass_begin
  - SecondaryVerticesHelix.daughterMass_end
  - SecondaryVerticesHelix.daughterDca2PV_begin
  - SecondaryVerticesHelix.daughterDca2PV_end
  - SecondaryVerticesHelix.daughterDca2PVChi2_begin
  - SecondaryVerticesHelix.daughterDca2PVChi2_end
  - SecondaryVerticesHelix.daughterPairIndices_begin
  - SecondaryVerticesHelix.daughterPairIndices_end





Summary

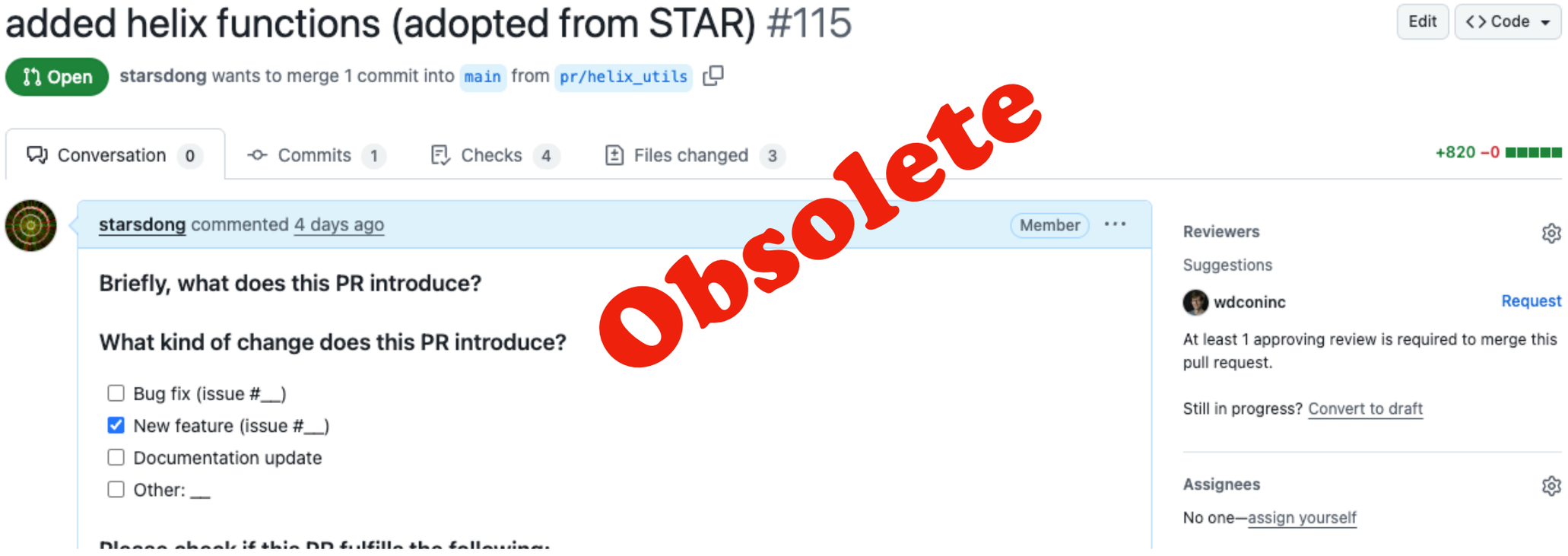
1. The proposal was presented and endorsed at the Reconstruction WG meeting 10/20.
   - current version is working with edm4eic::Vertex container
   - PR #2144 submitted under review

2. We will propose a new edm4eic container: SecondaryVertex (add-secondaryvertex brunch)
   - once this is added to edm4eic, will update SecondaryVertexHelix_factory then.

3. Continue the development of secondary VFs (Helix with more particles included, KFParticle etc.)

Backup

Adding Helix Functions in EDM4eic (obsolete!)



1) Helix afterburner reconstruction is used in D^0 reconstruction (later part), targeted to be used for updated physics projection plots.
2) Constructor includes using EICrecon TrackParameters as input.
3) Handling constant z-magnetic field (or zero field - straight-line)
   - can be extended to handle varying B-field for track projection
4) Iterative varying-step-scan to find DCA positions between helices numerically.

Usage of Helix Method

Helix method has been used in many heavy flavor hadron analysis (Rongrong/Shyam/Connie etc.)

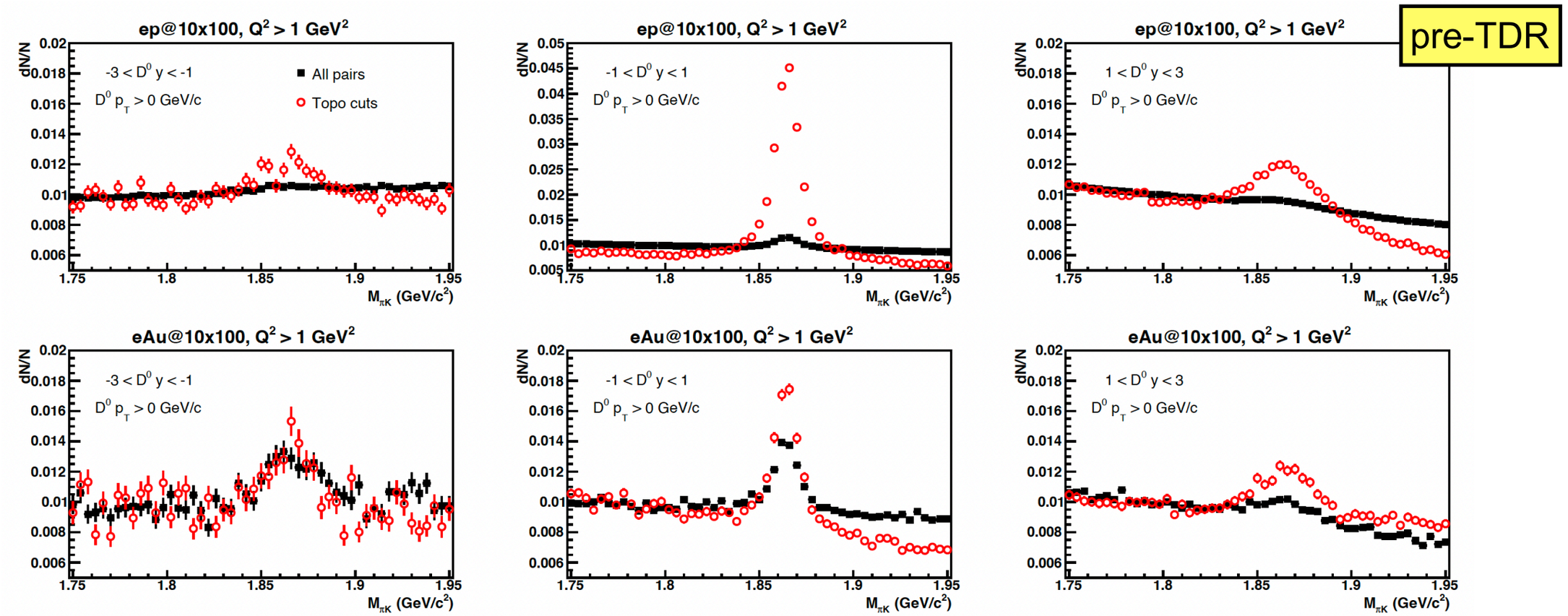


Figure 2.22: Invariant mass distributions of $\pi + K$ pairs with (red circles) and without (black squares) topological selections in 10×100 GeV e+p (top) and e+Au (bottom) collisions with a minimum Q^2 of 1 GeV^2. Different panels from left to right correspond to different D^0 rapidity intervals: $-3 < y < -1$ (left), $-1 < y < 1$ (middle) and $1 < y < 3$ (right).

Example of using Helix method:
https://github.com/marrbnl/ePIC/tree/main/HF_reco/helix