Saved track state information & pathlength error information

Barak Schmookler

Reminder: track states with CKF

- Three track states exist for a given track at any sensitive tracking surface (e.g. layer k):

1. Predicted track state: track state at layer k determined by previous k-1 layers
2. Filtered track state: track state at layer k including measurement at layer k
3. Smoothed track state: track state at layer k taking all measurements into account

What information we currently save to our ROOT file

- In our CentralTrackSegments collection, we save the predicted track state at every layer.
- Since the predicted track states are used for the chi-square determination, this is useful for many studies.
- However, having the final, smoothed track state information is useful for TOF PID estimates, for example.

What information we currently save to our ROOT file

- Current algorithm for CentralTrackSegments:

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

```
92            // get track state bound parameters and their boundCovs
93            const auto& boundParams = trackstate.predicted();
94            const auto& boundCov    = trackstate.predictedCovariance();
95
```

- Should we also add additional filtered and smoothed collections? At the least, we should have a command-line option to switch between the states.

Pathlength error

- Our track propagator algorithm, which is used to project the track to non-tracking layers, uses the final fitted track state:

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

```
241       // Get track state at last measurement surface
242       // For last measurement surface, filtered and smoothed results are equivalent
243       auto trackState        = trackContainer.trackStateContainer().getTrackState(tipIndex);
244       auto initSurface       = trackState.referenceSurface().getSharedPtr();
245       const auto& initParams = trackState.filtered();
246       const auto& initCov    = trackState.filteredCovariance();
```

- However, for both the CentralTrackSegments and the propagator, we don’t calculate path-length error. Is there a way to do that?

```
152       const auto pathLength       = static_cast<float>(trackstate.pathLength());
153       const float pathLengthError = 0;
```

Pathlength calculation with Acts parameterization

We can write the 2D pathlength as

$$s_T = \frac{\Delta\phi}{\kappa_T}$$

$$\kappa_T = |q/p_T| \cdot |B|$$

Then the full 3D pathlength is

$$s_{3D} = \frac{s_T}{\sin\theta} = \frac{\Delta\phi}{\kappa_T \sin\theta}$$

Pathlength calculation with Acts parameterization

Acts parameterizes the track state as

$$\vec{q} = (l_0, l_1, \phi, \theta, q/p, t)$$

So, we have

$$q/p_T = (q/p) \cdot \frac{p}{p_T} = (q/p)/\sin\theta$$

$$s_{3D} = \frac{\Delta\phi}{|q/p| \cdot |B| \cdot \sin^2\theta}$$

Pathlength calculation with Acts parameterization

With

$$s_{3D} = \frac{\Delta\phi}{|q/p| \cdot |B| \cdot \sin^2\theta}$$

We can write this Jacobian:

$$\vec{J} = \frac{\partial s}{\partial \vec{q}} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ -2s\cot\theta \\ -s/(q/p) \\ 0 \end{pmatrix}$$

$$\vec{q} = (l_0, l_1, \phi, \theta, q/p, t)$$

Pathlength calculation with Acts parameterization

With

$$s_{3D} = \frac{\Delta\phi}{|q/p| \cdot |B| \cdot \sin^2\theta}$$

We can write this Jacobian:

$$\vec{J} = \frac{\partial s}{\partial \vec{q}} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ -2s\cot\theta \\ -s/(q/p) \\ 0 \end{pmatrix}$$

For the Jacobian, this is dependent on Δϕ, which is anchored by the position measurement. So, to the extent that the endpoint is fixed, there is limited dependence…but I can think about this more.

$$\vec{q} = (l_0, l_1, \phi, \theta, q/p, t)$$

Pathlength calculation with Acts parameterization

We can then calculate the error on the pathlength:

$$\sigma_s^2 = \vec{J}^T C \vec{J} = \frac{s^2}{(q/p)^2}\sigma_{q/p}^2 + 4s^2\cot^2\theta\,\sigma_\theta^2 + 2\cdot\frac{s}{q/p}\cdot(-2s\cot\theta)\cdot\mathrm{Cov}(q/p,\theta)$$