Kalman Fitter Updates

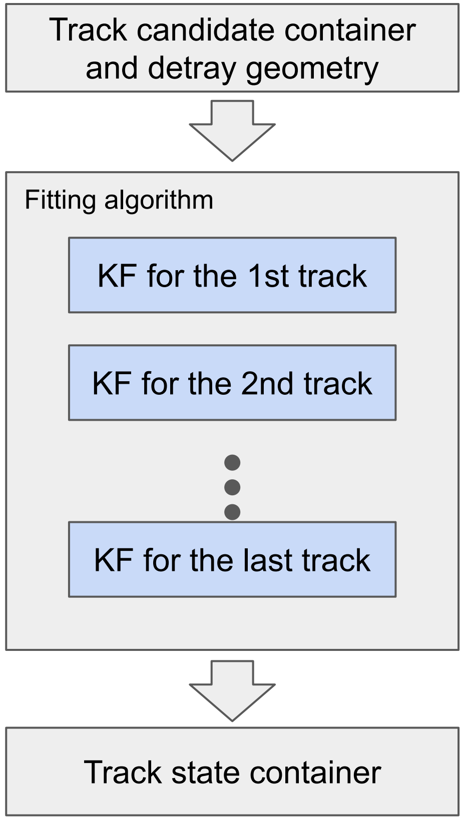
Beomki Yeo

Major updates

- CPU Kalman fitter in traccc (acts-project/traccc#264)
- SYCL support in detray (acts-project/detray#369)
- GPU Kalman fitter (CUDA & SYCL) in traccc (acts-project/traccc#296)

Recap on Kalman Fitter

- Input is a track candidate which is a set of measurements associated with a track
  - Potentially the output of combinatorial KF
- Fitting algorithms runs the KF
  - CPU: iterates over tracks
  - GPU: one thread per track
- Output is a track state which contains the fitted track parameter information



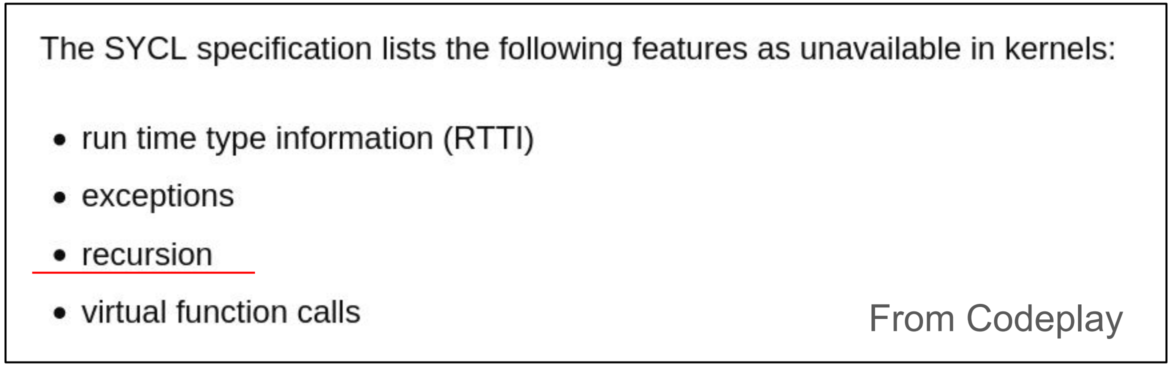
Some issues

- Detray had not been fully tested with SYCL device code

- Pull value distributions were wrong in certain conditions
  - Single precision
  - Nonzero incidence angle (Non-linear effect?)

SYCL support for detray

- Recursion is not allowed in the SYCL device code :/



The SYCL specification lists the following features as unavailable in kernels:

- run time type information (RTTI)
- exceptions
- recursion
- virtual function calls

From Codeplay

- Detray uses both run time and compile time recursion
  - **Run time:** std::sort to sort the surface candidates at navigation
  - **Compile time:** used everywhere to iterate tuple containers

- Fortunately, it turned out that *only runtime recursion* is unavailable in kernels
  - Just replacing std::sort with a custom sort function did the job

An Issue in single precision matrix inversion

- In the first implementation of KF, the pull value distributions were distorted with **single precision** and custom matrix algebra (cmath)
  - Imprecise 6X6 matrix inversion in the smoothing algorithm
  - Rounding error in *the cofactor method* was the main problem

- Following is an example of 6x6 matrix that appears during the smoothing, and its determinants calculated from different math plugins

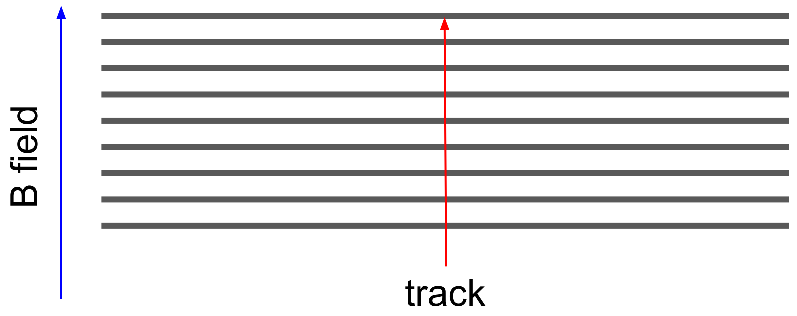
$$\begin{pmatrix} 10.792386 & 0.216181 & 0.057650 & -0.002764 & 0.000001 & 0. \\ 43.909368 & 10.372997 & 0.231496 & -0.065972 & -0.000002 & 0. \\ 0.045474 & -0.001730 & 0.000246 & 0.000004 & 0. & 0. \\ -0.255134 & -0.059846 & -0.001345 & 0.000383 & 0. & 0. \\ -0.001490 & -0.000057 & -0.000008 & 0.000001 & 0.000001 & 0. \\ 0. & 0. & 0. & 0. & 0. & 89875.517874 \end{pmatrix}$$

	Determinant
cmath (float)	**0.0000000004327766964**
cmath (double)	0.0000000004306355415
Eigen (float)	0.0000000004306401069
Eigen (double)	0.0000000004306355415

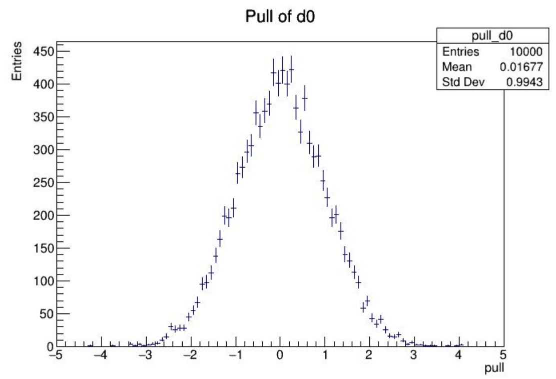
- The problem was solved after adding another matrix inversion algorithm (LU decomposition)

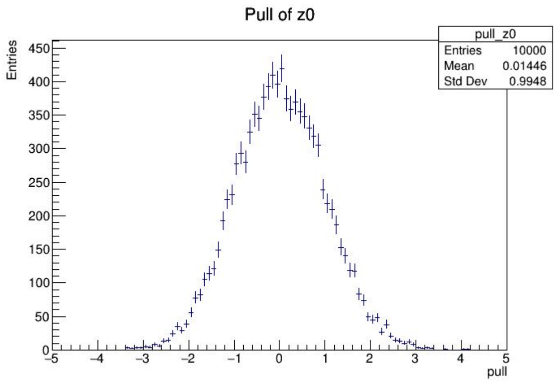
Pull value distribution tests

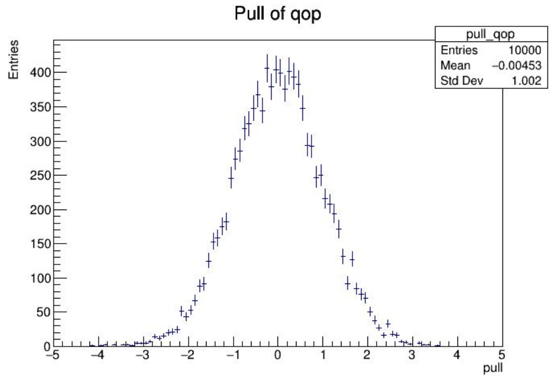
- Simulation setup
  - Single precision
  - 9 silicon planes along the x-axis (2 cm gap)
  - 2 Tesla B field in the x-axis
  - 50 um measurement resolution
  - 1 GeV/c initial momentum

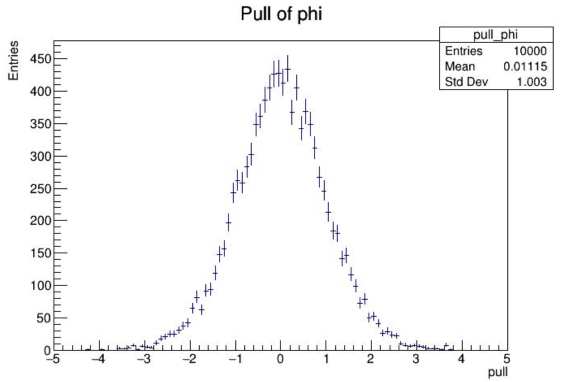


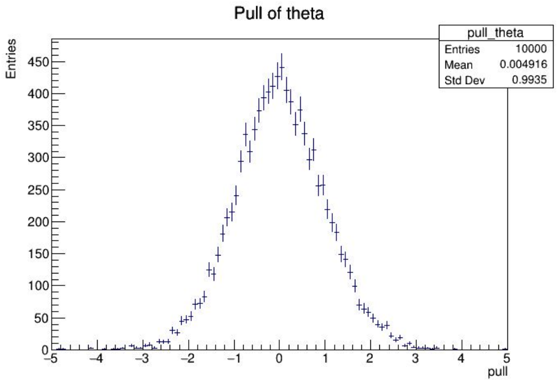
Zero Incidence Angle



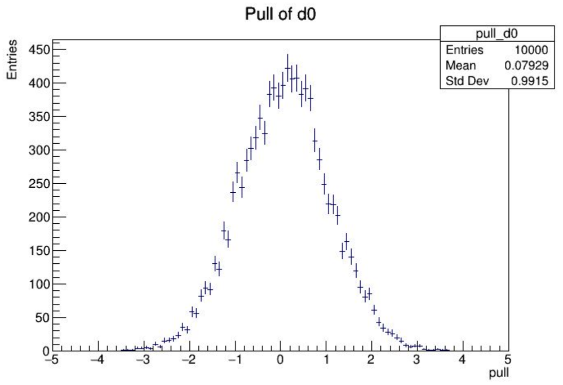


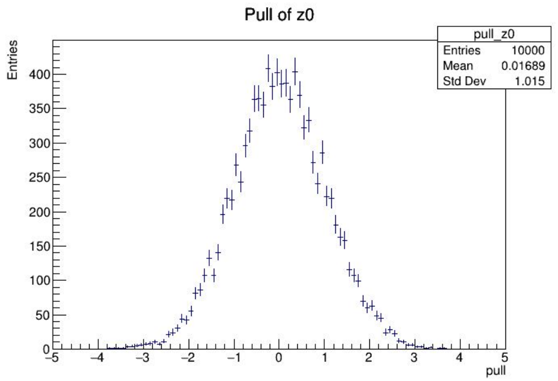


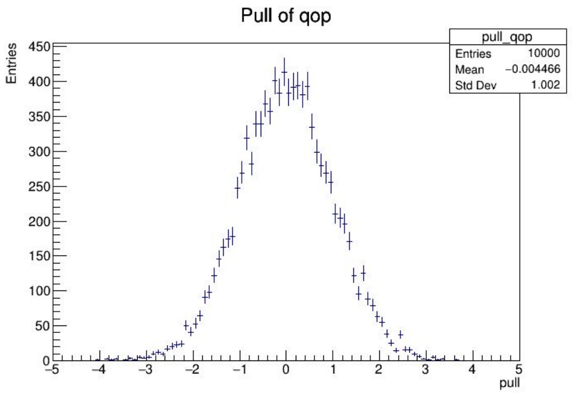


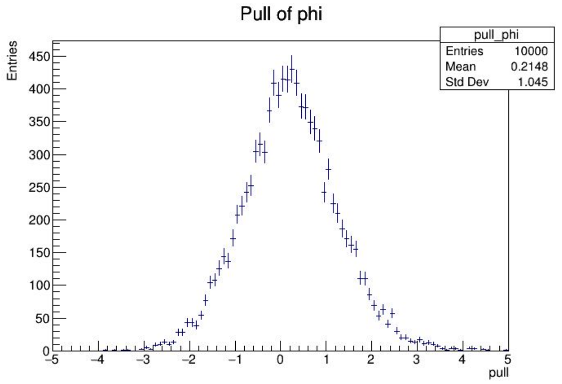


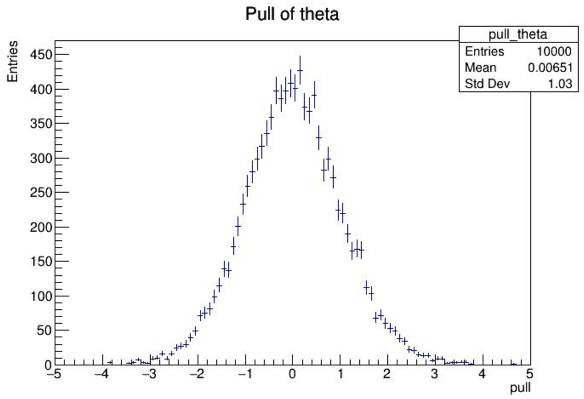
Nonzero Incidence Angle (45 degree)



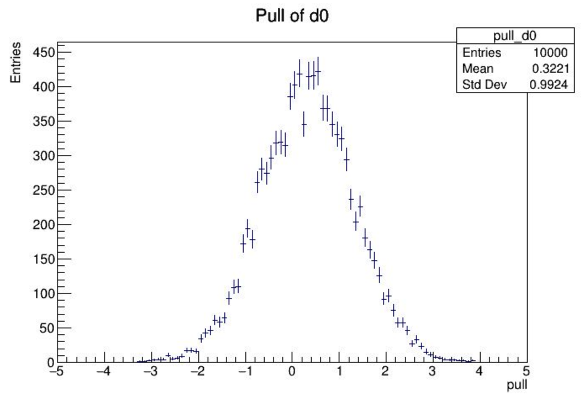


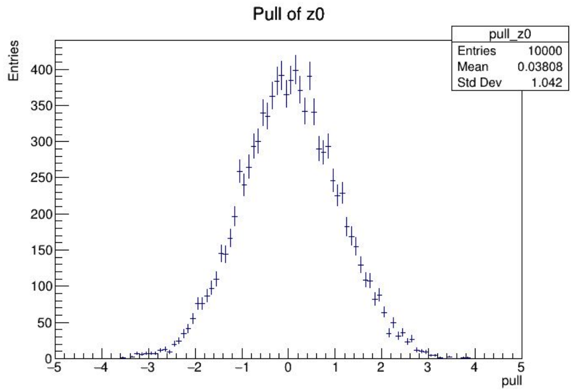


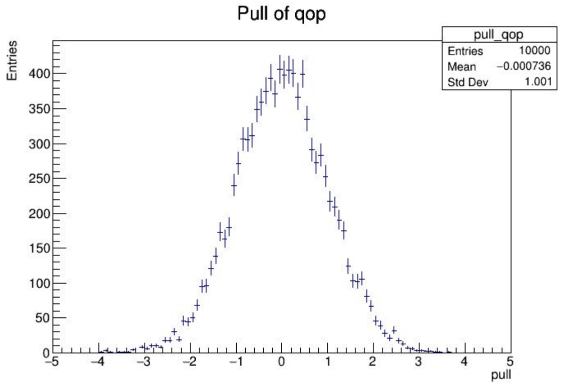


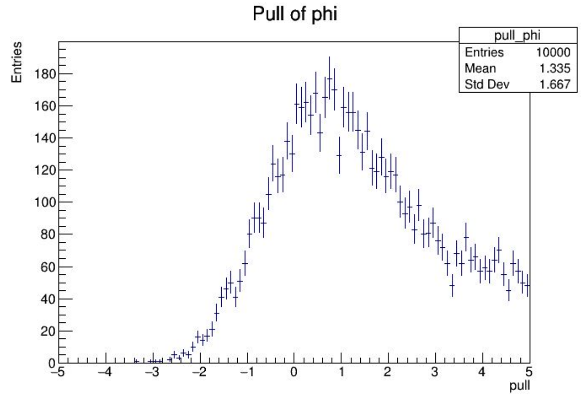


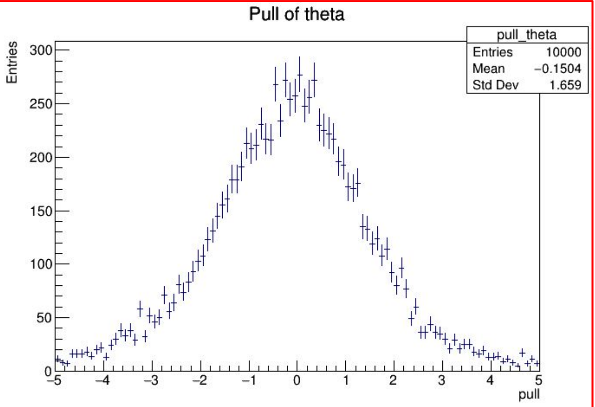
Nonzero Incidence Angle (75 degree)











Nonlinear Effect?

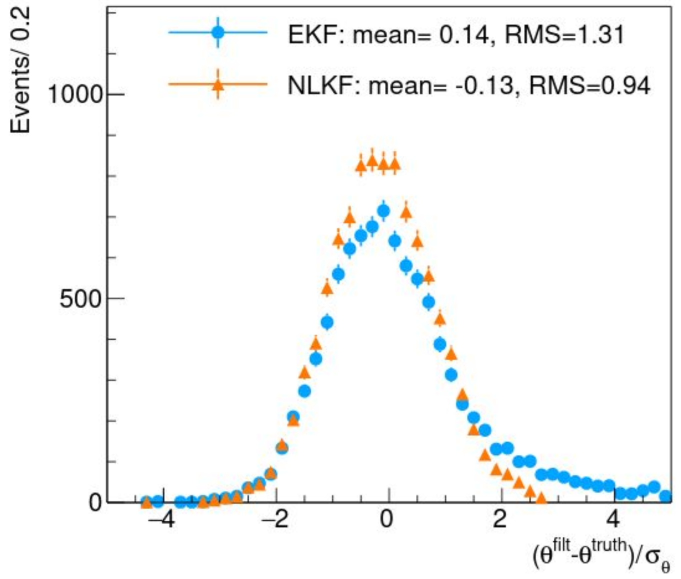


Figure 3: Comparison of pull of the filtered momentum direction polar angle θ using the EKF (blue) and NLKF (orange) with the setup in Fig. 2. Ten thousand tracks and their corresponding measurements are simulated.

- We have observed the same behavior in the nonlinear KF paper

Summary

- There were a couple of issues in implementing Kalman filtering but they are resolved now
  - Runtime recursion is removed from detray to enable SYCL
  - Imprecise single precision of matrix inversion is fixed

- There seems a non-linear effect with nonzero incidence angle after reproducing the paper’s result