

# **High Performance Computer Architectures Practical Course - Exercise 5 -**

Tutorium 1

David Jordan (6260776)  
Florian Rüffer (7454628)  
Michael Sanjatin (7485765)

May 22, 2023

## Line 324-430

First of all the geometry is set up, containing the number of stations / particles, the distance between those stations and the hit distribution. Those values will result in a linearly spaced distribution of stations along the z-axis. This simulates a set of tracks with random starting positions and slopes.

Next up, we need to get our tracks fitted, to determine the most probable trajectory of a particle. For this we need to define the SIMD-ized version of the fitting process or it will be computed in a scalar way.

```
1  const int NVTracks = NTracks/fvecLen;
2  LFFTrack<fvec,fvec> vTracks[NVTracks];
3
4  CopyTrackHits( tracks, vTracks, NVTracks );
5
6
7  LFFitter<fvec,fvec> fit;
8
9  fit.SetSigma( Sigma );
10
11 #ifdef TIME
12     timer.Start(1);
13 #endif
14     for ( int i = 0; i < NVTracks; ++i ) {
15         LFFTrack<fvec,fvec> &track = vTracks[i];
16         fit.Fit( track );
17     }
18 #ifdef TIME
19     timer.Stop();
20 #endif
21
22
23     CopyTrackParams( vTracks, tracks, NVTracks );
24
25 #endif
```

File 1: KFLineFitter.cpp

In the first line we calculate the number of vectorized tracks, to assign it to our list of vectorized tracks in the next line (as the length). In the following steps and lines of code, we transform our scalar track data into a SIMD vector, fit the tracks parameter to the measure data points and add Sigma (uncertainty) to our simulation. The for-loop then iterates over every vectorized track to fit multiple tracks simultaneously. Last but not least we copy our tracks back to their original format (line 23). Especially the functions CopyTrackHits and CopyTrackParams are of interest. This function iterates over the grouped tracks

and for those over each hit for every track. All tracks have the same number of hits, those correspond to the number of stations, therefore we can treat this parameters as a constant. For CopyTrackParams we need to iterate over the vectorized tracks and their entries, as well as the elements of the covariance matrix.

The last lines of code save our results. If an output file is defined, then the program will output the parameters and covariance matrices for every track. The output will be stored in the defined file and will look like this:

```

1      0
2      90 -9.13228 -0.0736924
3      90 -9.03232 -0.0722956
4      0.0862335 0.00135828 3.00843e-05

```

File 2: Output

Here we will list the line and its corresponding explanation (index number corresponds to line in the output):

1. Track index, which increments by one for every track
2. True values of the parameters (Monte Carlo truth)
3. Reconstructed / fitted values for the same parameters
4. Covariance matrix for the fitted parameters

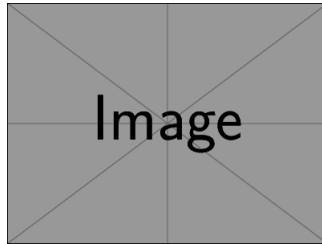


Figure 1: Output