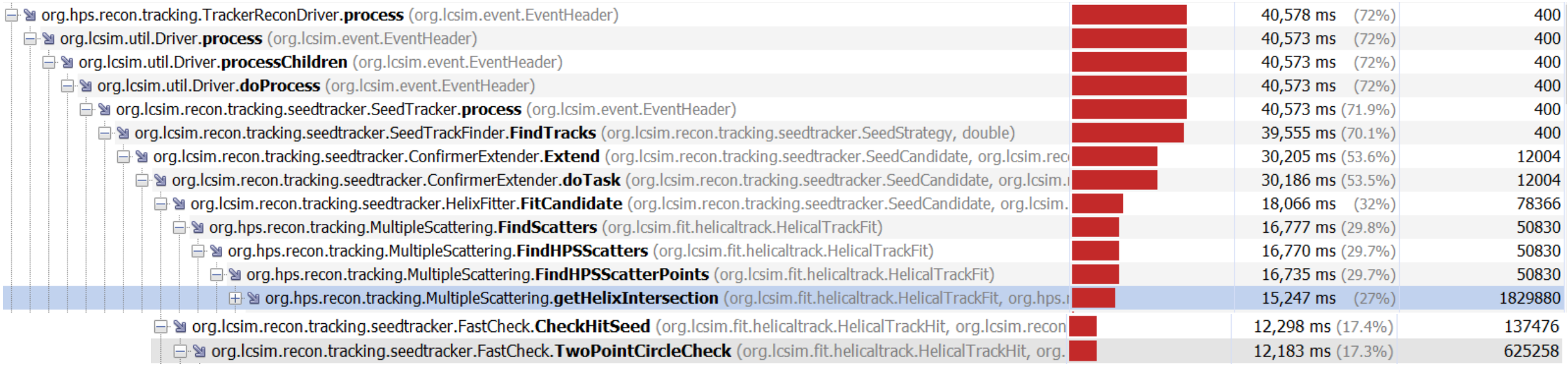
SPEEDING UP THE TRACKING CODE

MIRIAM DIAMOND

JUNE 19 2017

github issue 117

EVIO TO LCIO PROFILING: TRACKER RECON DRIVER



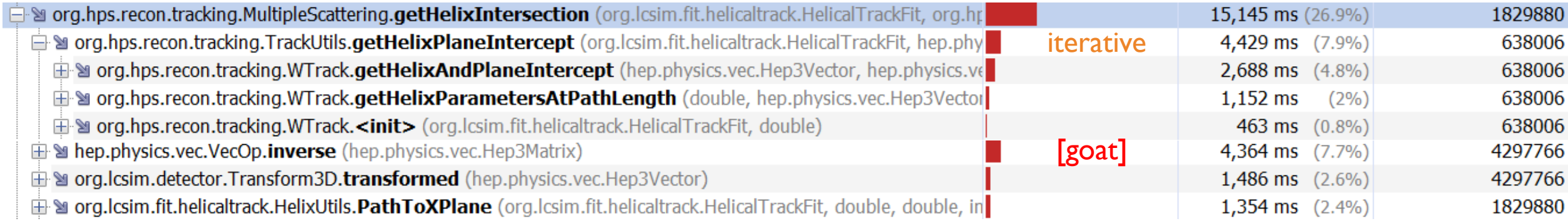
I investigated speeding up:

- getHelixIntersection
- TwoPointCircleCheck

HELIX INTERSECTION: CURRENT CODE

- For each track seed (starting with hits triplet), Extend calls findHPSScatterPoints for each possible track extension into each subsequent layer
- For each possible track extension, findHPSScatterPoints calls getHelixIntersection for each sensor
- getHelixIntersection steps:
  - Approximate calculation of helix intersection pt with sensor plane
  - Determine whether intersection pt falls within boundaries of sensor, +/- isInside tolerance (1 mm)
  - If so, proceed to iterative calculation (convergence precision $\varepsilon = 10^{-4}$ mm, typically requires 2-3 iterations)
  - Determine whether more precise intersection pt is within boundaries of sensor

HELIX INTERSECTION: MODIFYING THE CODE

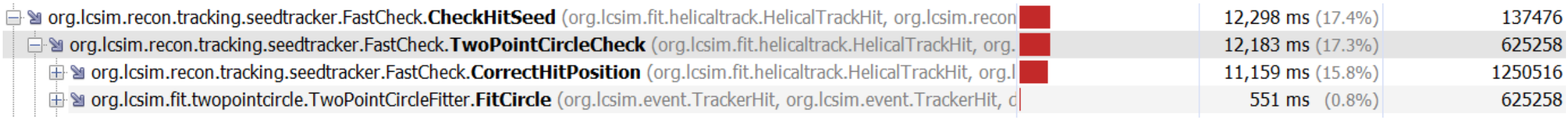


- I reduced #calls to getHelixIntersection by skipping sensors in layers >3 we know the track won't hit
  - Assume the track hits *top* or *bottom* but not both
  - Assume track cannot hit both *hole* and *slot* in same half-module
- I added doIterative switch to getHelixIntersection: when off, only performs approximate calculation
  - Turned it off for Extend steps, but back on for final track fits

TWO POINT CIRCLE CHECK: CURRENT CODE

- For each valid pair of sectors (determined by Sectoring) in seed layers (dictated by Strategy), algorithm goes through every possible hit pair
  - constructs circle through the two points + max helix DCA
  - then examines arc length and (r, z) of points
- Fundamentally different from standard ATLAS / CMS pair-finding that limits initial combinatorics
  - for each hit in outer sector, uses [ϕ + max DCA] to set a limited ϕ range of hits in inner sector to examine
  - then examines z of points

TWO POINT CIRCLE CHECK: MODIFYING THE CODE

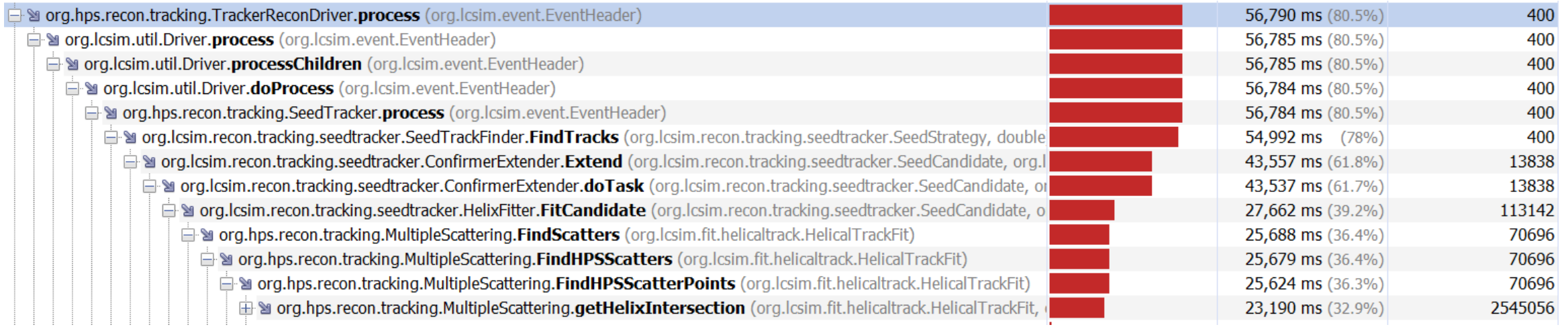


- I considered replacing current algorithm with ATLAS / CMS pair-finding, but current algorithm performs well despite combinatorics disadvantage
- Most TwoPointCircleCheck time is taken by CorrectHitPosition
  - CheckHitSeed checks a hit (to maybe add to track seed) against every existing hit in seed, calling TwoPointCircleCheck(hitToMaybeAdd, hitExisting) in loop over all hitExisting
  - TwoPointCircleCheck corrects positions of hitToMaybeAdd and hitExisting, independently… calculating same correction on hitToMaybeAdd for each hitExisting
  - Calculating correction on hitToMaybeAdd only once per seed cuts execution time in ~half

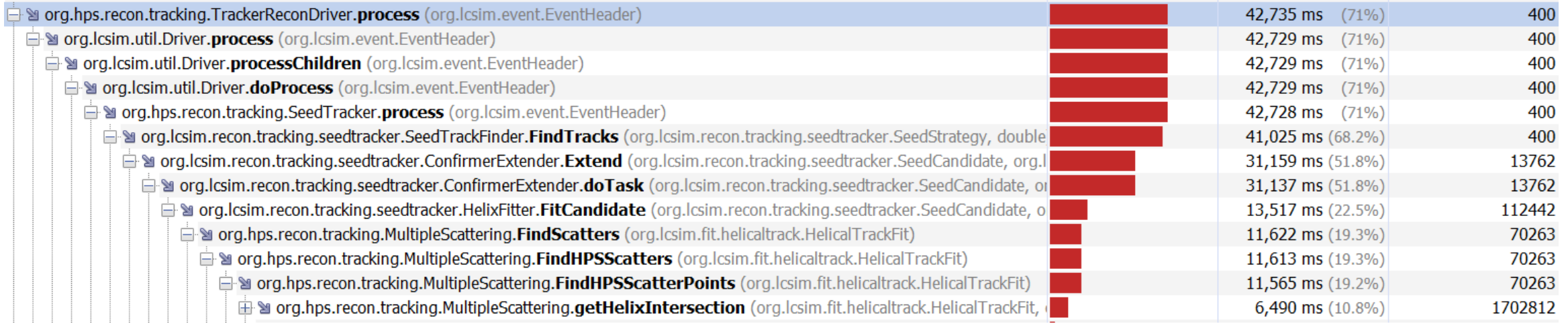
ASSESSMENT

- Total time savings according to profiler
- Approximate vs iterative results for individual helix intersection points
- Performance studies: to discuss
  - What quantities (impact parameters? residuals?)
  - Designating "standard" files (data? MC?) for benchmarking

TIME SAVINGS

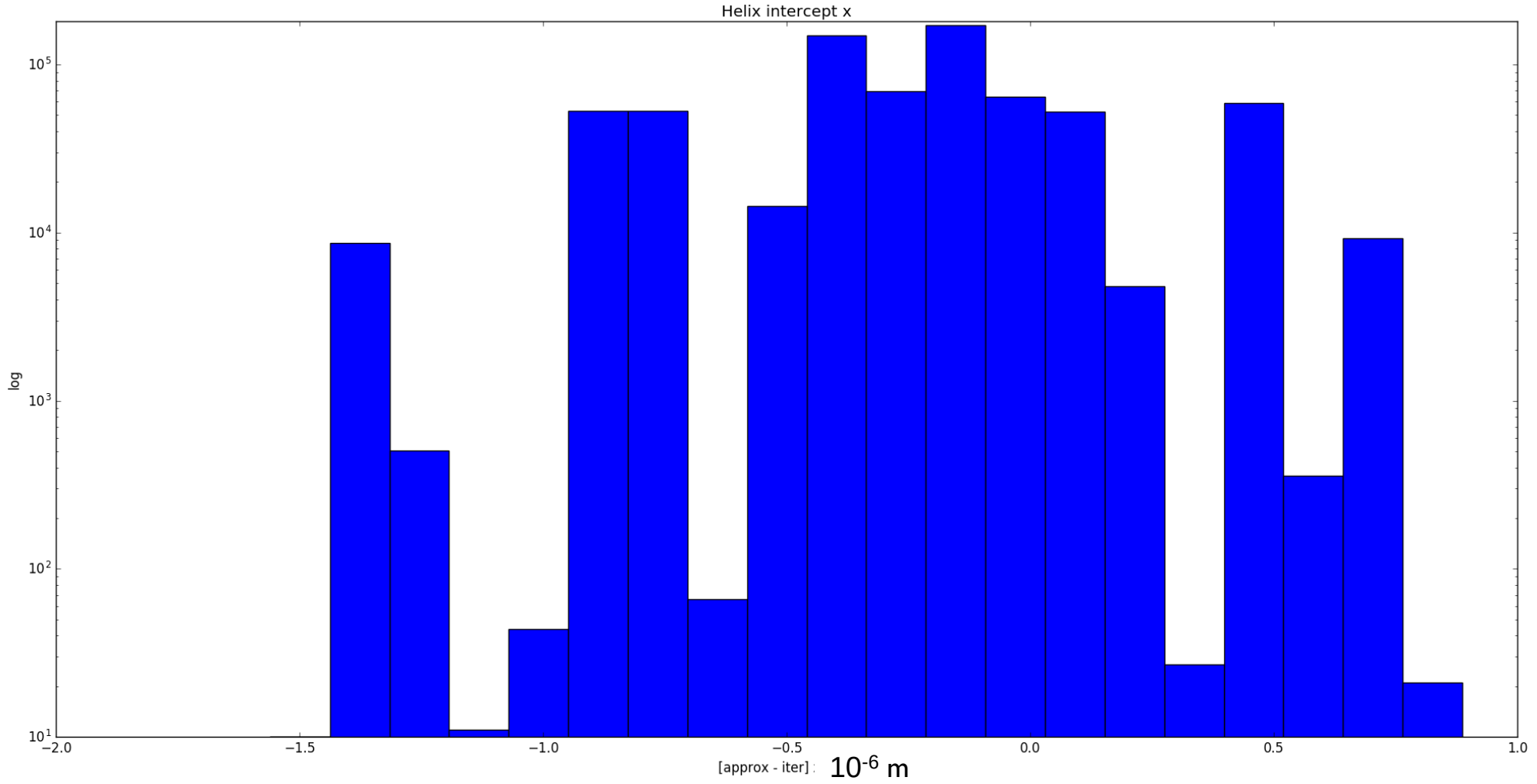


Current

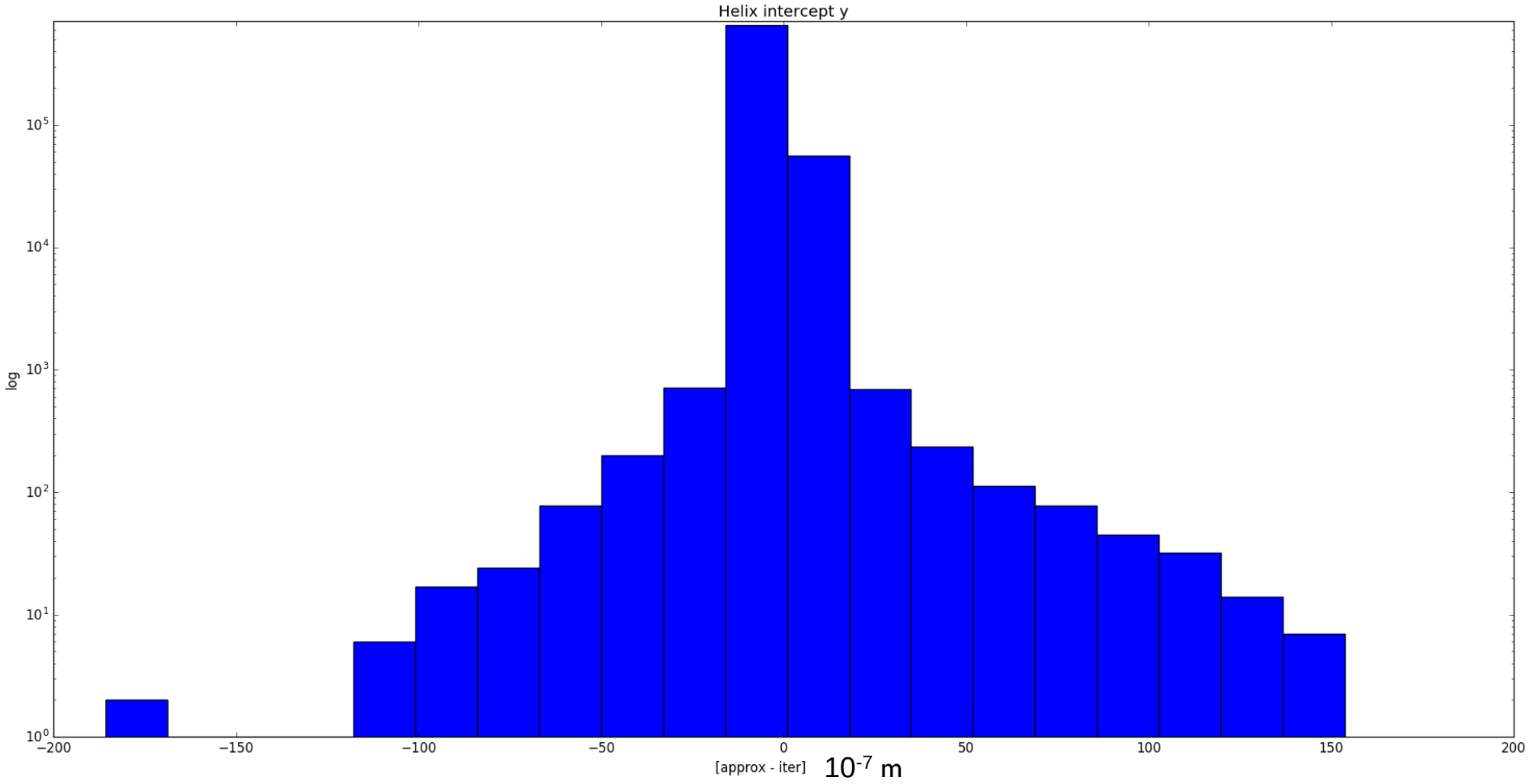


Modified

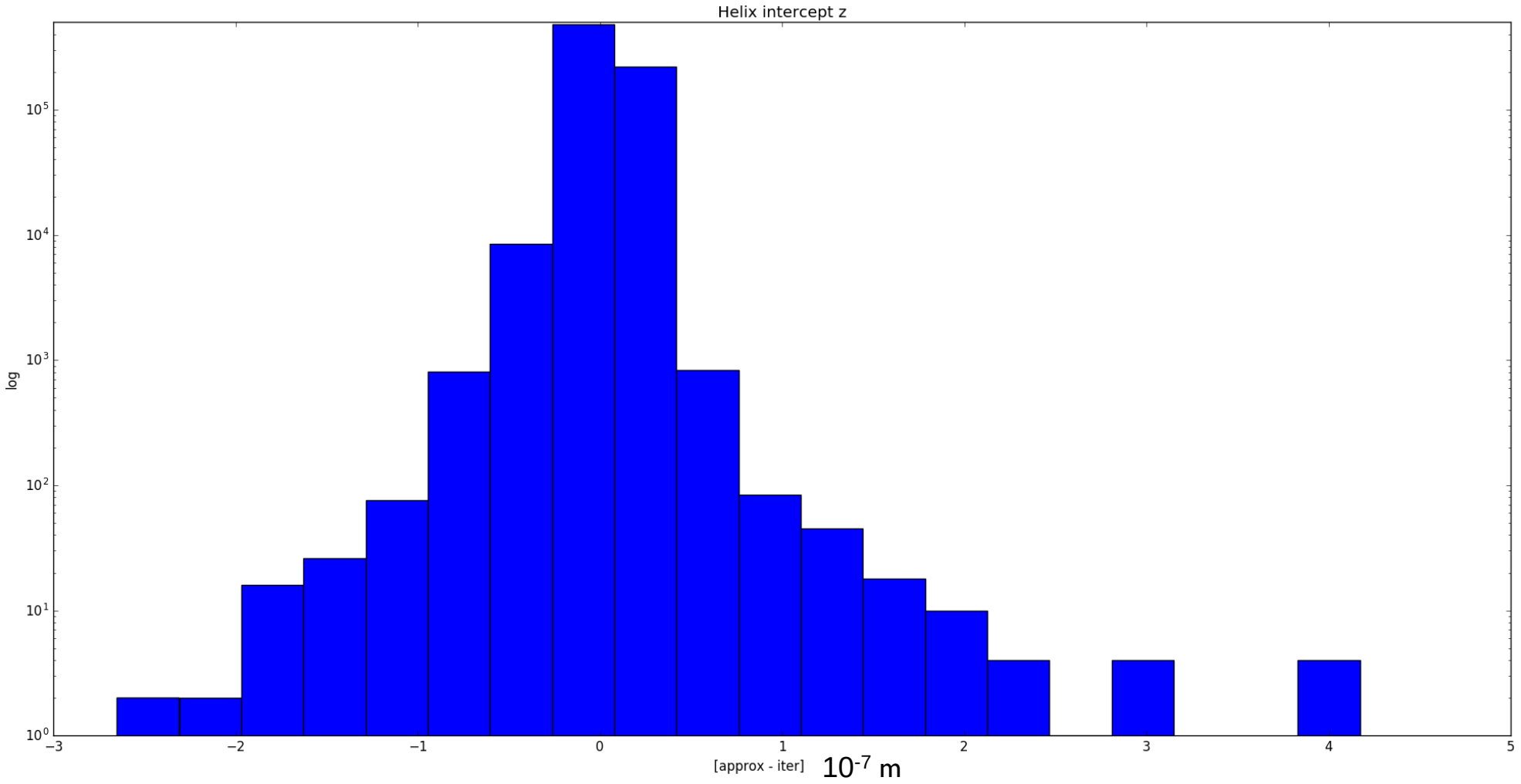
APPROXIMATE VS ITERATIVE : HELIX INTERSECTION POINTS



APPROXIMATE VS ITERATIVE : HELIX INTERSECTION POINTS



APPROXIMATE VS ITERATIVE : HELIX INTERSECTION POINTS



HELIX MULTIPLE SCATTERING ERROR

MIRIAM DIAMOND

JUNE 19 2017

github issue 126

CURRENT CODE

- org.lcsim.recon.tracking.seedtracker.FastCheck ▶ ThreePointHelixCheck
  triplet-finding for track seeds
  - For each of the 3 hits, calculates contribution to *z error*

```
if (hit instanceof HelicalTrack3DHit)
    dztot += _nsig * ((HelicalTrack3DHit) hit).dz();
else {
    zfirst = false;
    if (hit instanceof HelicalTrack2DHit)
        dztot += ((HelicalTrack2DHit) hit).zlen() / 2.;
    else
        dztot += _nsig * Math.sqrt(hit.getCovMatrix()[5]);
}
```

  - Then

```
//  Add multiple scattering error here - for now, just set it to 1 mm
dztot += 1.;
```

  - Compares *total z error* to *(predicted – actual) z position* of middle hit

```
if (Math.abs(zpred - z[1]) > dztot) return false;
```

MODIFYING MULTIPLE SCATTERING ERROR TERM

- Replace 1. with quick approximation based on
  - Momentum estimate
    - Radius of curvature and slope of 3-point helix (already calculated in method)
    - B-field (already required as input to method)
  - Thickness (constant) of an SVT sensor
  - Distances between hits