Re-Alignment of 2016 with KF-based Tracks

Tom Eichlersmith

he/him/his

University of Minnesota

eichl008@umn.edu

May 2, 2023

Why?

Continued Validation of KF-based Alignment

MC-based studies look very promising, but a study with real data can help shed some light on previously hidden areas.

Potential for Improvement

Plan to re-reconstruct 2016 data with updates to tracking for newer analyses, potentially include an improved detector.

How?

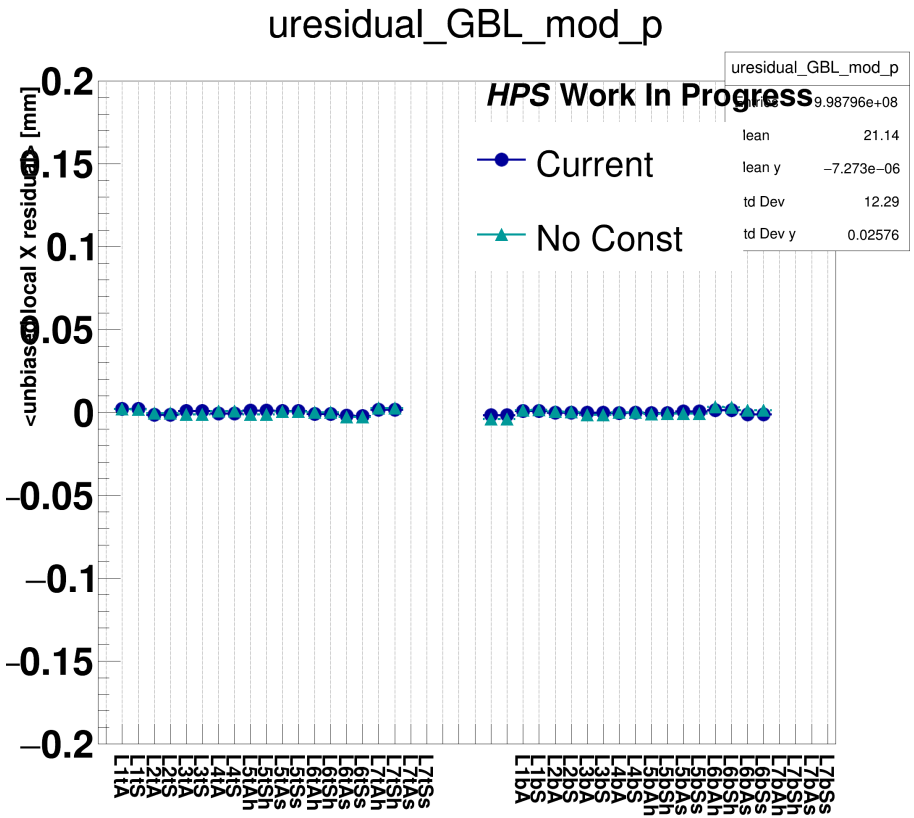
Been keeping notes on our GitHub repositories.

1. hps-java and GBL setup
2. hps-java-5.1-74-g4f599a3 The last alignment-related update to hps-java
3. hps-mc:2016-param-patch A few patches to hps-mc to support 2016 parameters
4. hps-align:22-to-pip Ongoing developments of plotting library

Vocab

- **Current** – current best alignment done using ST-based tracks but now using KF-based tracks to calculate residuals and track parameters (detector: HPS-PhysicsRun2016-Pass2)
- **No Const** – kept detector definition but dropped all millepede constants

In General, Alignment Constants don't do much...



- Seemingly no trends in residuals across layers
- Pretty tight around zero

Let's look in more detail.

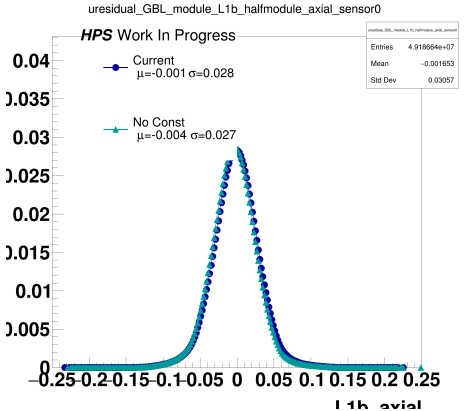
Plain Unbiased Residuals

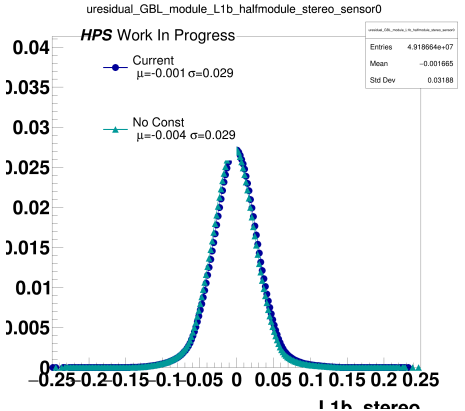
Looking for translation along sensitive direction (tu) misalignment

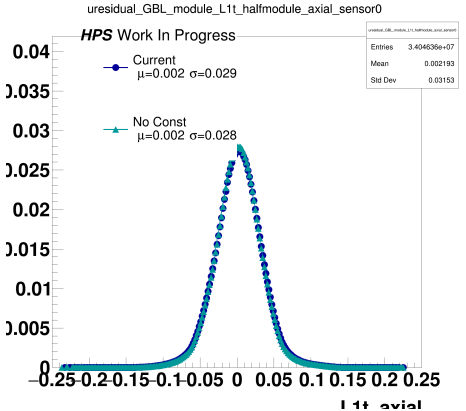
Long Story Short

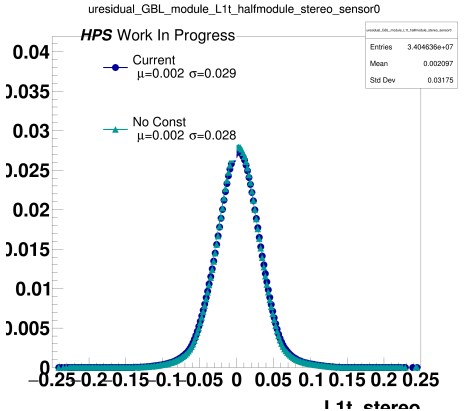
Unbiased residuals look Gaussian and are close to 0.

Layer 1 sensors shown to the right, other layers are included in tar-ball uploaded on the agenda.









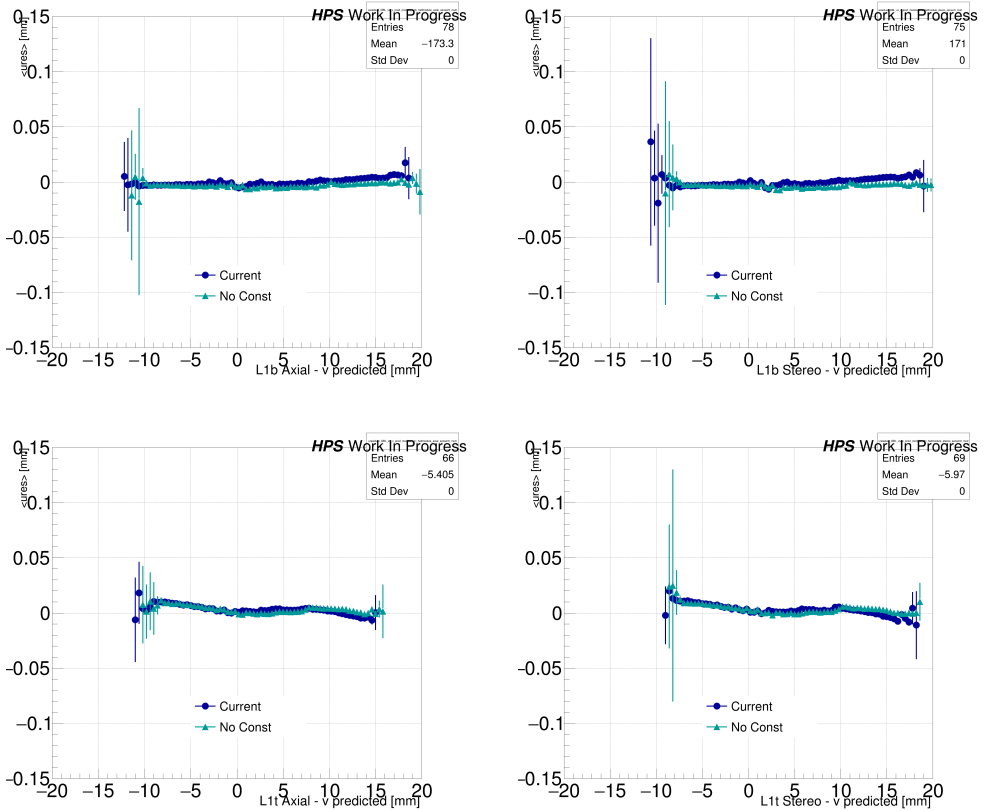
Unbiased Residuals against v

Looking for rotation about normal to sensor plane (rw) misalignment

Long Story Short

Residuals appear to show a marginally smaller trend without the alignment constants?!?

Layer 1 sensors shown to the right, other layers are included in tar-ball uploaded on the agenda.



First Attempt

The first attempt to use pede to deduce alignment parameters seems to have failed. (Floating tu for Layers 2, 3, 4, and 5 both axial and stereo).

Pede Res for Top tu

```
  11103    -0.62451E-02      0.0000      -0.62451E-02    0.10039E-04
  11104     -1.2145          0.0000       -1.2145        0.36776E-03
  11105    -0.10358E-01      0.0000      -0.10358E-01    0.15922E-04
  11106     -2.1013          0.0000       -2.1013        0.63620E-03
  11107    -0.36279E-01      0.0000      -0.36279E-01    0.22731E-04
  11108     -1.4004          0.0000       -1.4004        0.42456E-03
  11109     0.17706E-01      0.0000       0.17706E-01    0.37559E-04
  11110      1.4379          0.0000        1.4379        0.43698E-03
  11111    -0.46167E-01      0.0000      -0.46167E-01    0.21059E-04
  11112     -1.0937          0.0000       -1.0937        0.33161E-03
  11113     0.25534E-01      0.0000       0.25534E-01    0.43946E-04
  11114      1.1096          0.0000        1.1096        0.33864E-03
```

Will continue investigating...

Questions

Pede Res for Bottom tu

1	21103	0.34322	0.0000	0.34322	0.31662E-03
2	21104	0.19207E-02	0.0000	0.19207E-02	0.80202E-05
3	21105	0.59237	0.0000	0.59237	0.54726E-03
4	21106	0.34902E-02	0.0000	0.34902E-02	0.12894E-04
5	21107	0.39315	0.0000	0.39315	0.36429E-03
6	21108	0.72776E-02	0.0000	0.72776E-02	0.17732E-04
7	21109	-0.40944	0.0000	-0.40944	0.37972E-03
8	21110	-0.18157E-02	0.0000	-0.18157E-02	0.33369E-04
9	21111	0.31978	0.0000	0.31978	0.29636E-03
10	21112	0.88188E-02	0.0000	0.88188E-02	0.15545E-04
11	21113	-0.32437	0.0000	-0.32437	0.30201E-03
12	21114	-0.40128E-02	0.0000	-0.40128E-02	0.40346E-04

Bugs Patched Since Last Time

1. Second Memory Leak: Tom found a two more edge cases that the GBL-bound objects were allocating and were not getting cleaned up: (a) the GblPoint objects were being copied into the GblTrajectory and then not cleaned and (b) if the trajectory did not pass the χ^2 cut to be written to the output file, it was not being cleaned. ▸ PR 958 in hps-java
2. Including odd-number-hit KF tracks. Previous cut throwing away these tracks to achieve closer parity with ST tracks was removed.
3. Composite Align Bracket Placement. PF noticed the closing bracket on composite align was much too late, moving it up to where it is supposed to be allowed for running on 2016 data where we don't use the "alignable detector elements".

Record of Related Bugs Patched on hps-java:master

1. First Memory Leak: PF deduced that java was not cleaning up the GBL trajectories created by the C++ library and accessed via JNA. This was causing the alignment driver to eventually crash the program on systems with a "small" memory allocation for the JVM.
2. Unintentionally Dynamic Cut: The way "hits" are counted is different between ST and KF tracks. ST tracks use two sensors to construct 3D points. This means whenever we want at least N hits in an ST track, the same cut for KF tracks is $2N$. The alignment driver was mistakenly multiply the cut by 2 *on each track* eventually overflowing and returning to a cut of 0. This meant a lot of poor, low-hit-count tracks were being included in alignment when attempting to use KF.
3. Null Trajectories: Sometimes GBL fails to construct a trajectory from a track. This seems to be happening with some low-quality KF tracks, but I need to investigate more. For now, I simply avoid computing the residuals if the GBL trajectory fails to construct so the entire run doesn't crash.