Linear Collider Software at CLIC

André Sailer
On behalf of the CLICdp collaboration

CERN-EP-LCD

LCWS 2017
Strasbourg
October 24, 2017

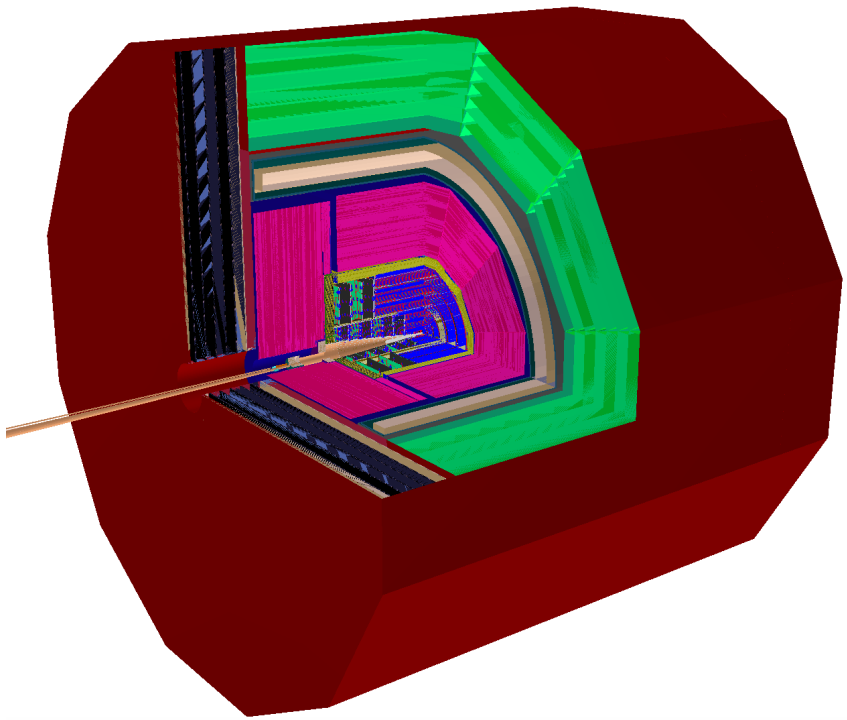
Table of Contents

Section 1:

1 Simulation and Reconstruction

Simulation and Detector Model

- Detector Model frozen, documented in note
- Thirteen is our lucky number
- Described in **CLICdp-Note-2017-001**



Reconstruction Workflow

Reconstruction workflow

1. Overlay
2. Digitisation
3. Track Pattern recognition (TruthTracking or ConformalTracking)
4. Track Fit (Refit for better track parameter estimate)
5. Particle Flow Reconstruction (PandoraPFA)
6. Forward calorimeter reconstruction (LumiCal/BeamCal)
7. PFO selection

- Implemented in one unified steering file for the reconstruction that can be configured on the fly: avoid duplicating parameter settings in different files that will diverge as much as possible
  `CLICPerformance/examples/clicReconstruction.xml`
- `Marlin --Config.Tracking=Conformal --global.LCIOIn...`

Validation

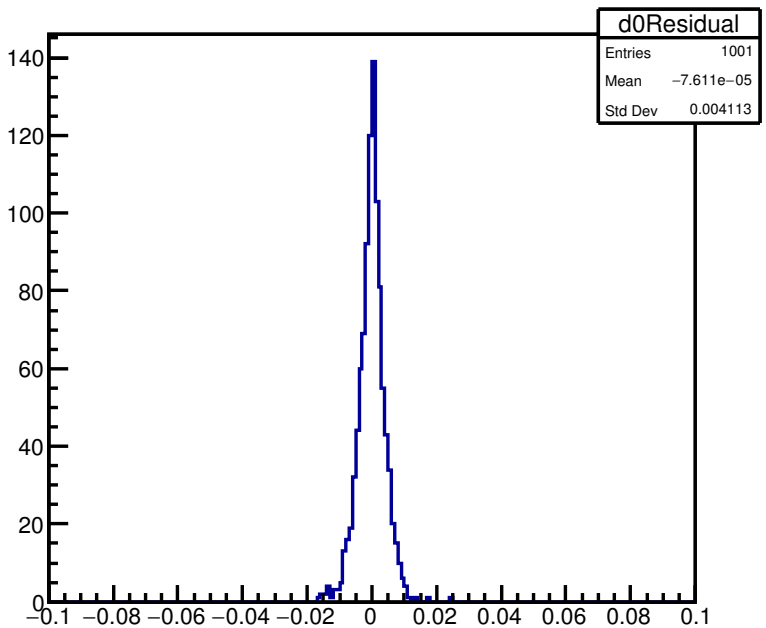
- The reconstruction is running from overlay to flavour tagging, etc.
  - Centrally creating larger sized samples: $\texttt{Z} \rightarrow \texttt{uds}$, $\texttt{qqqq}$, $\texttt{t}\bar{\texttt{t}}$ in `iLCDirac`
  - with and without overlay of $\gamma\gamma \rightarrow$ hadron
- Performances are being studied to identify issues that still need to be addressed, can, or should be improved
  - Improve pattern recognition for displaced tracks, tune parameter settings
  - Improve track parameter resolutions, parameter uncertainty
  - Improve particle ID efficiencies
  - Tune particle flow reconstruction
  - Add PFO Selector for lower energy

Details on current status of performance in Emilia's talk on tracking and my talk on calorimetry

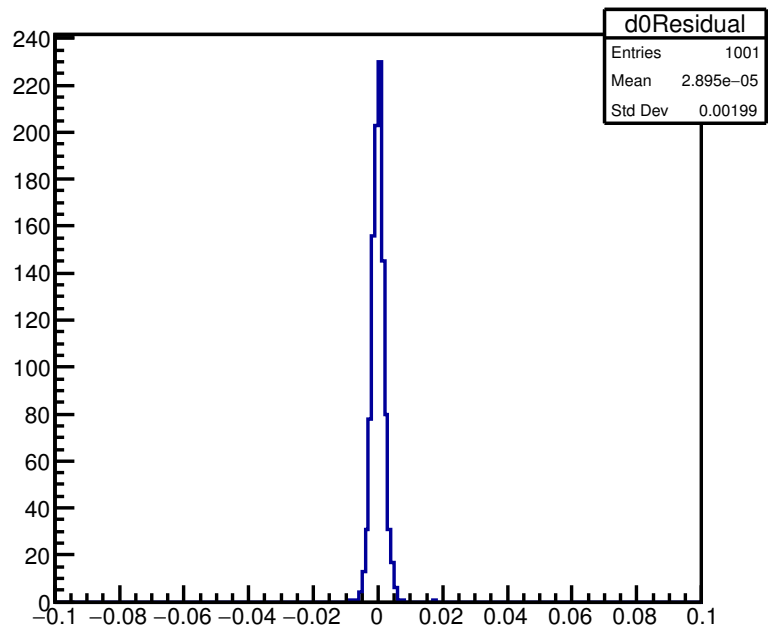
Track Reconstruction

- Pattern Recognition via Conformal Tracking processor for all tracking detectors combined
- Initial TrackFit + Refitting with the output as initial parameters

First Fit



Refit

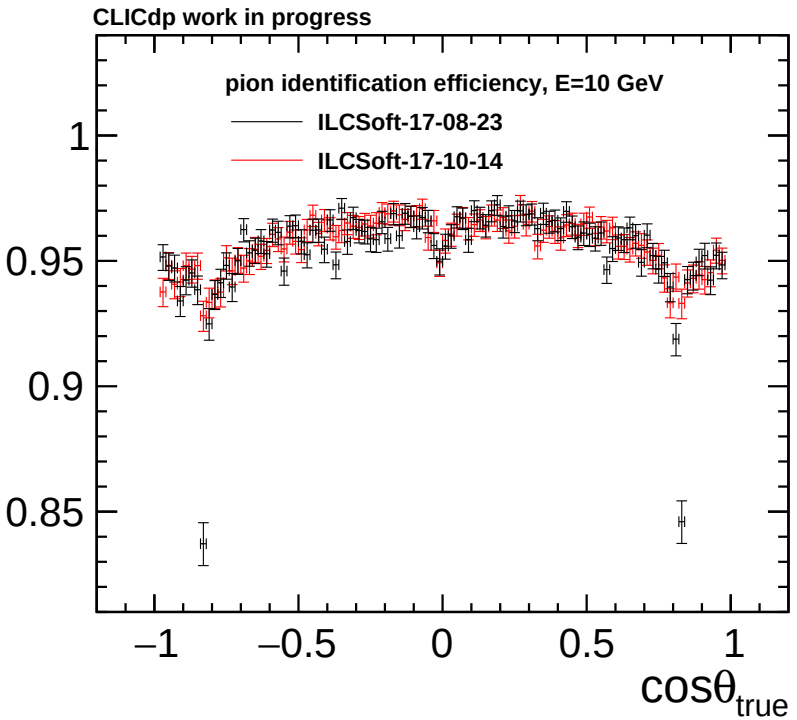


See Emilia Leogrande's presentation Thursday afternoon for the tracking performance

Calorimeter Reconstruction

Finalising Particle Flow reconstruction

- Apply software compensation at higher shower energies
- Improved track cluster matching

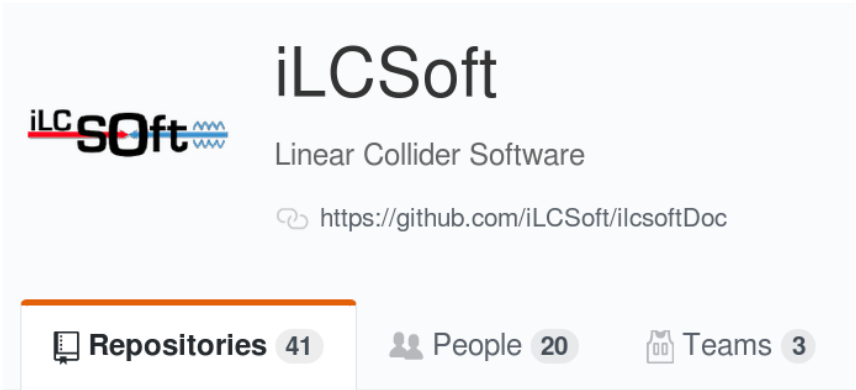


More details on Wednesday morning

Section 2:

2 Infrastructure

iLCSoft on GitHub



`https://github.com/iLCSoft`

- Built in features: Pull Request (PR): **code review**, issue tracker
- Extensions: TravisCI, coverity static analysis

`ilcsoft.github.io`

Nightly	Build GCC			Build LLVM
iLCSoft	build passing			build passing
AIDASoft	**Build Status**	**Issues**	**PRs**	**Coverity Scan**
DD4hep	build passing	issues 6	pull requests 1	coverity passed
aidaTT	build passing	issues 0	pull requests 0	coverity passed
iLCSoft	**Build Status**	**Issues**	**PRs**	**Coverity Scan**
CED	build passing	issues 1	pull requests 0	coverity passed
CEDViewer	build passing	issues 1	pull requests 1	coverity passed
CLICPerformance	build passing	issues 1	pull requests 1	coverity passed

⋮

Continuous Integration

- Run two-hourly compilation of iLCSoft with `gcc` and `clang`
  - currently: gcc 6.2, llvm 3.9
  - Notification when merge breaks dependent package
- Nightly deployment to `/cvmfs/clicdp.cern.ch` used in Continuous Integration system on GitHub
  - CVMFS also includes `git`, compiler, emacs, WHiZARD2, ILCDIRAC

Plan to control nightly builds and releases via GitLab-CI system

- Web interface
- Scheduling for nightly builds part of GitLab-CI, instead of cronjob
- Deploy with single button, as we already do for iLCDirac

Monitoring Code Quality

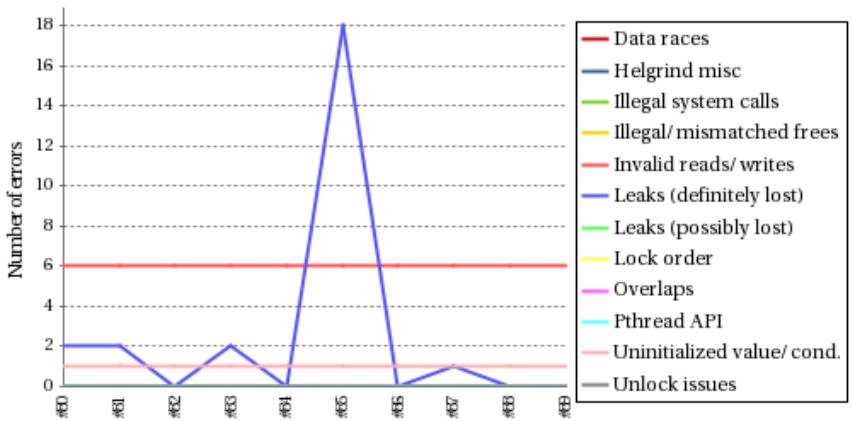
- Individual nightlies to monitor number of compiler warnings
  - Once package without warnings easy to disallow new ones (`-Werror` in CI build)
  - Still large amount of *technical debt*
    - ★ E.g.: `MarlinReco` has 2200 compiler warnings
    - ★ Some of these might be actual bugs, decrease performance

All | **Overview** | +

S	W	Name ↓	Last Success	Last Failure	Last Duration	# Compiler Warnings
		aidaTT	37 min - #110	N/A	1 min 19 sec	13
		CED	17 hr - #110	N/A	23 sec	116
		CEDViewer	9 hr 25 min - #110	N/A	1 min 3 sec	192
		ClicPerformance	3 hr 23 min - #137	N/A	31 min	0
		Clupatra	6 hr 24 min - #109	N/A	1 min 12 sec	77
		ConformalTracking	13 hr - #108	N/A	52 sec	331
		DDKalTest	5 hr 3 min - #115	N/A	54 sec	59
		DDMarlinPandora	11 hr - #106	N/A	1 min 40 sec	0
		FCalClusterer	8 hr 23 min - #108	N/A	2 min 22 sec	13
		ForwardTracking	13 hr - #115	N/A	1 min 32 sec	230
		iLCUtil	14 hr - #106	N/A	3 min 16 sec	11
		KalDet	7 hr 15 min - #110	N/A	2 min 14 sec	293
		KalTest	4 hr 0 min - #107	N/A	1 min 50 sec	115
		KiTrack	5 hr 13 min - #109	N/A	1 min 10 sec	105
		KiTrackMarlin	17 hr - #109	N/A	1 min 24 sec	192
		LCFIPlus	7 hr 47 min - #111	N/A	2 min 35 sec	6
		LCFIVertex	14 hr - #110	N/A	4 min 39 sec	4
		lcgeo	22 hr - #117	N/A	7 min 25 sec	0
		LCIO	18 hr - #107	N/A	4 min 11 sec	10

Memory Checking

- Nightly run of `valgrind` to monitor memory leaks in reconstruction
- extensive `valgrind` suppression file for `DD4hep`, `ROOT`, `LCIO` false positives
- Fixed all *definite* and *possible* leaks for processors used in CLIC reconstruction chain
- If there is a new leak, receive email notification. Only Look at PRs of last 24 hours



Code Quality and Modernisation

For example the `ConformalTracking` package

- No more compiler warnings in gcc62 or clang/llvm39
  - Wall, Wextra, Wshadow, Weffc++
- Enforce consistent formatting with `clang-format`: `make format`, or git pre-commit hook
- Use `unique_ptr`, `shared_ptr` for simplified memory management.
  - `unique_ptr` forces efficiency by avoiding copies
  - Automatic cleanup

```
copy_if(make_move_iterator(tracksTemp.begin()),
        make_move_iterator(tracksTemp.end()),
        back_inserter(tracks),
        [this](UcellularTrack const& track) {
          return (int(track->size()) >= (m_minHits-1)); });
tracksTemp.clear();
```

Standard algorithm to move objects from one vector to another using `unique_ptr` and `lambda` function. No `for` loop or `if/else`, `delete`, etc.

Section 3:

3 Beyond the Next Mass Production

Beyond the Next Mass Production

- Modernise software used for linear colliders
  - Get more serious about code quality: let tools take care of mundane tasks
    - ★ Enforce uniform formatting git `(git-)clang-format`: at least be consistent inside each package, file, or pull request
    - ★ no more compiler warnings merged into the repository
  - C++14 (17?)
  - Look into multi threaded running of reconstruction
- Too many packages prohibit complete testing for PRs
  - Would be great to test everything down to running of reconstruction

Let's discuss at the end of this session

Section 4:

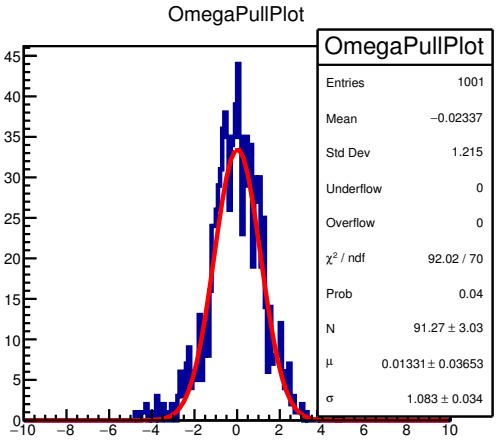
4 Summary

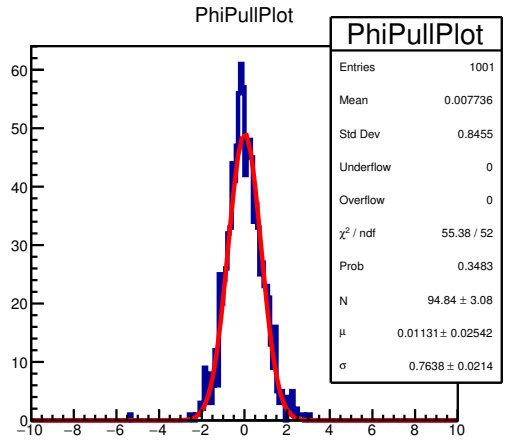
Summary

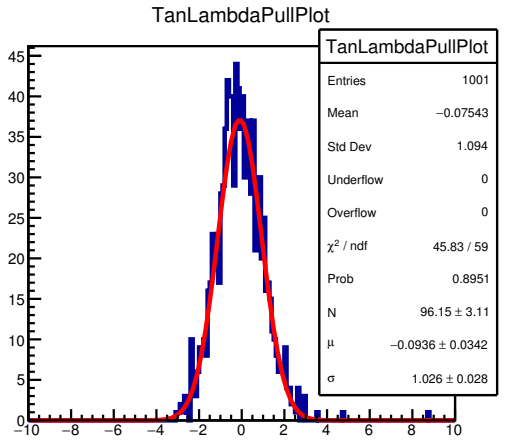
- Pieces for the next large scale production falling into place
  - Finalise detector calibrations and then start mass production
- Automatic testing is absolutely essential, and simplifies one's life, prevents surprises, and let's one do more important and interesting things
- Start thinking about medium term development for software

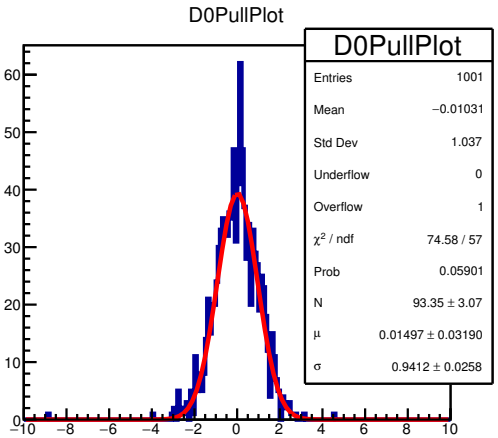
Backup Slides

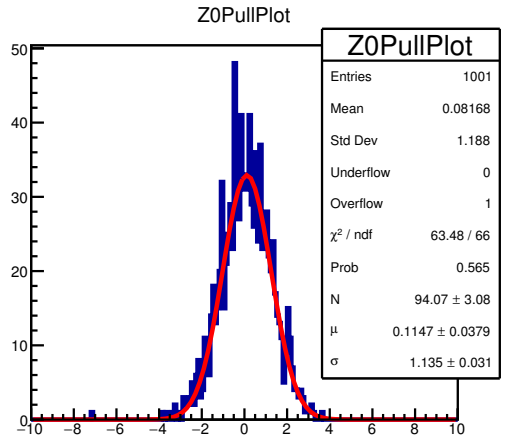
Pulls TruthTrackFinder



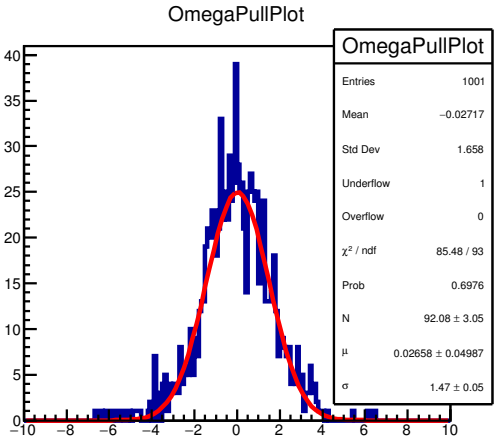


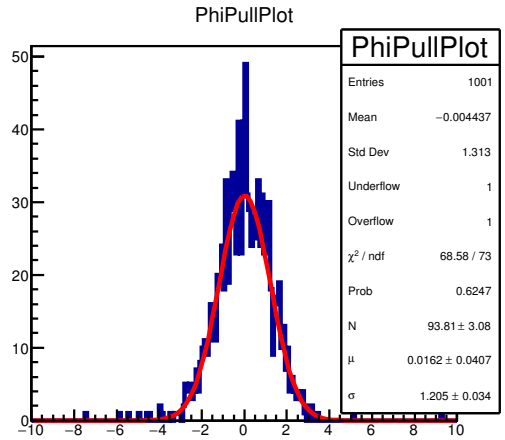


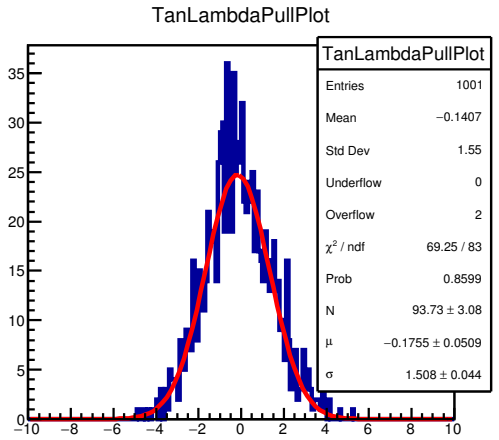


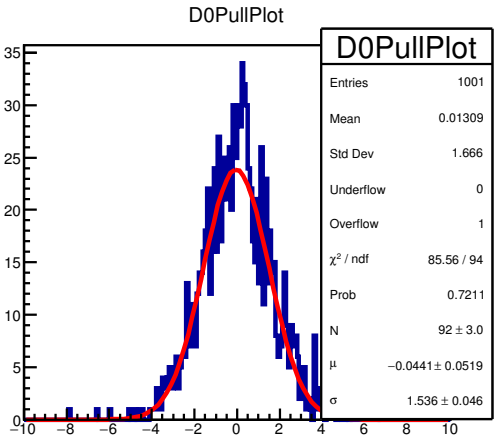


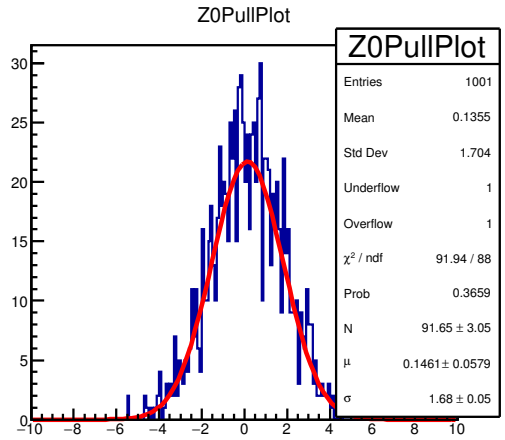
Pulls Refit


OmegaPullPlot
OmegaPullPlot
Entries 1001
Mean −0.02717
Std Dev 1.658
Underflow 1
Overflow 0
χ² / ndf 85.48 / 93
Prob 0.6976
N 92.08 ± 3.05
μ 0.02658 ± 0.04987
σ 1.47 ± 0.05

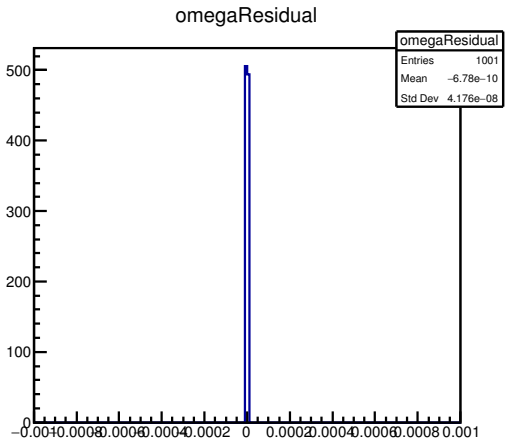

PhiPullPlot
PhiPullPlot
Entries 1001
Mean −0.004437
Std Dev 1.313
Underflow 1
Overflow 1
χ² / ndf 68.58 / 73
Prob 0.6247
N 93.81 ± 3.08
μ 0.0162 ± 0.0407
σ 1.205 ± 0.034

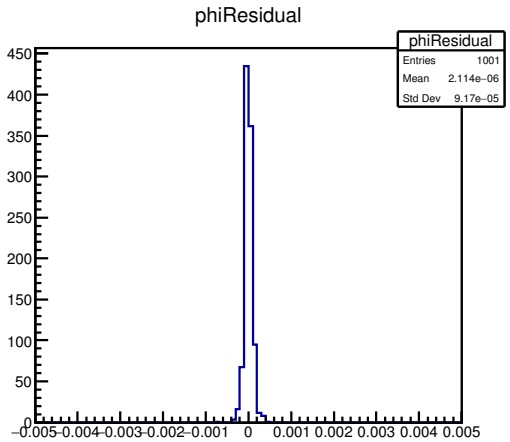

TanLambdaPullPlot
TanLambdaPullPlot
Entries 1001
Mean −0.1407
Std Dev 1.55
Underflow 0
Overflow 2
χ² / ndf 69.25 / 83
Prob 0.8599
N 93.73 ± 3.08
μ −0.1755 ± 0.0509
σ 1.508 ± 0.044

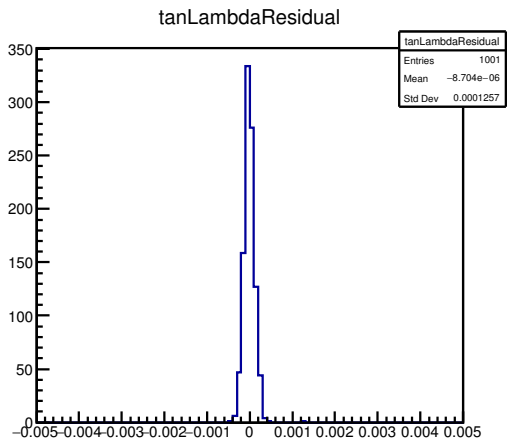

D0PullPlot
D0PullPlot
Entries 1001
Mean 0.01309
Std Dev 1.666
Underflow 0
Overflow 1
χ² / ndf 85.56 / 94
Prob 0.7211
N 92 ± 3.0
μ −0.0441 ± 0.0519
σ 1.536 ± 0.046

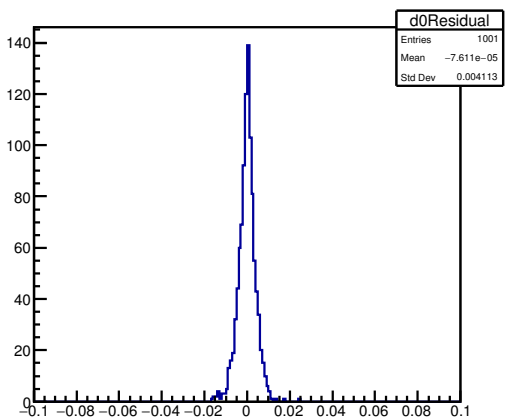

Z0PullPlot
Z0PullPlot
Entries 1001
Mean 0.1355
Std Dev 1.704
Underflow 1
Overflow 1
χ² / ndf 91.94 / 88
Prob 0.3659
N 91.65 ± 3.05
μ 0.1461 ± 0.0579
σ 1.68 ± 0.05

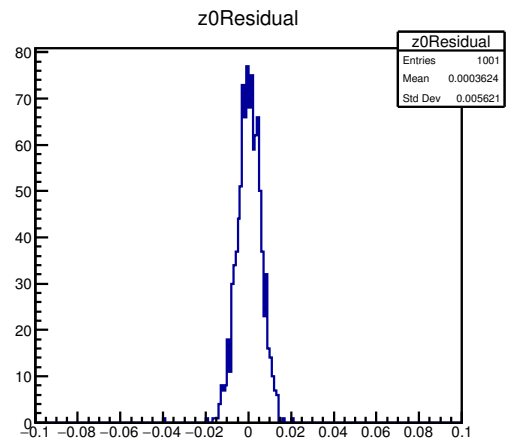
Resolution TruthTrackFinder











Resolution Refit