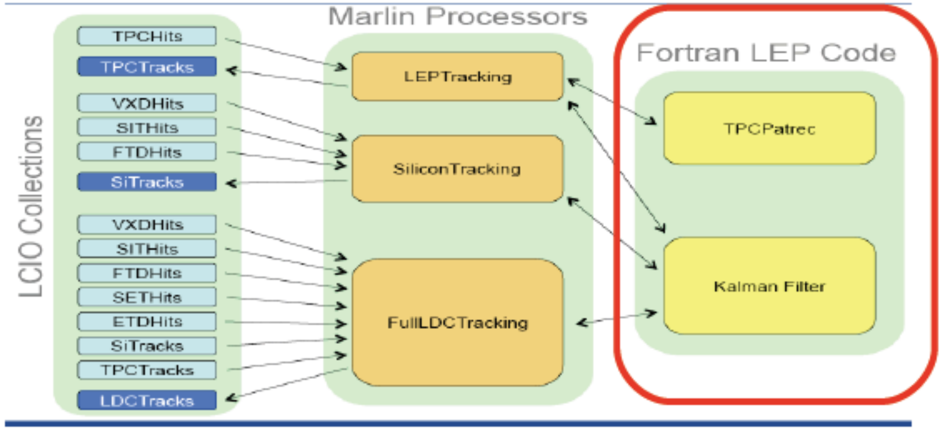
Progress in ILD Tracking SW for the DBD

Steve Aplin and Frank Gaede

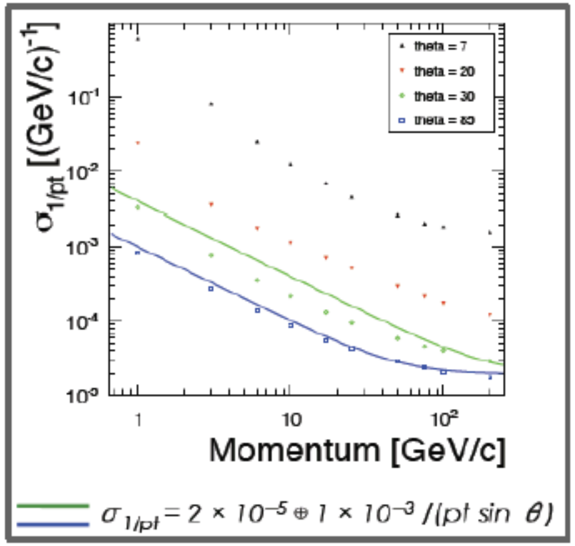
ILD-SW Meeting
2nd February 2011

ILD Tracking software



- standalone tracking in TPC – LEP code (f77)
- standalone patrec in VXD/SIT/FTD – in Marlin (C++)
- merging of Track segments and refit w/ f77 Kalman fitter

- current tracking used for LOI process
  - required p_t resolution reached
  - also in presence of backgrounds (even bg*3)
- issues:
  - f77: maintenance 'nightmare' !
  - homogeneous B-field only !
  - difficult to use with backgrounds
  - only single BX reconstruction
  - issues at 1-3 TeV
  - no strip tracking (ghost hits)



$\sigma_{1/pt} = 2 \times 10^{-5} \oplus 1 \times 10^{-3} / (pt \sin \theta)$

- -> need for a new tracking package

Towards Tracking-SW for the DBD

- had a look into ATLAS tracking code (S.Aplin)
- full featured modern PatRec:
  - (combinatorical) Kalman Filter
  - Gaussian Sum Filter, DAF,...
  - modular design
- hoped for simple integration into Marlin - however
  - rather tight coupling to GAUDI and Athena frameworks
    - algtools, DataVec, logging,...
- too involved for now

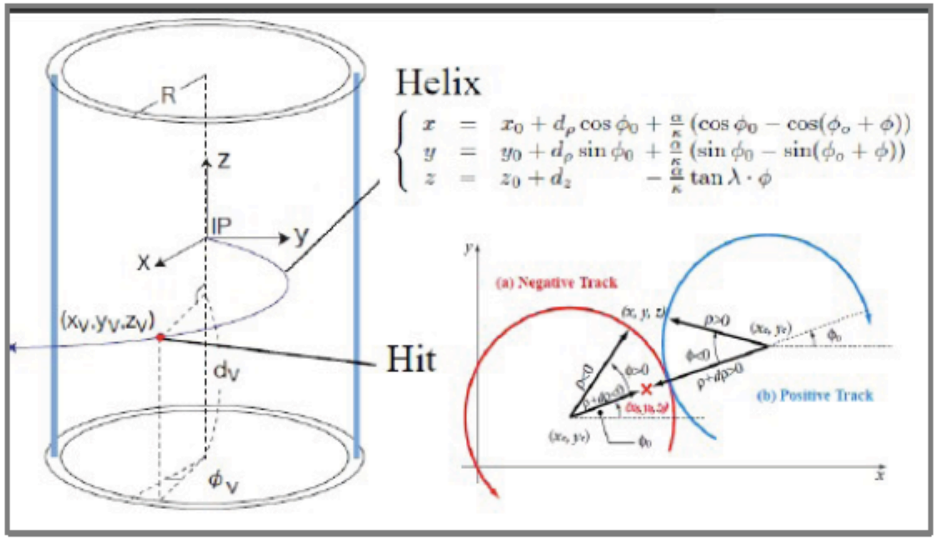
- also checked other Tracking/Fitting packages:
  - KalTest
    - developed @ KEK, used by LCTPC, based on ROOT
  - GenFit
    - developed @ TU Munich, to be used for SuperBelle, ROOT based
  - both seem to be good candidates for developing a new iLCSoft tracking package
  - started incorporating KalTest:
    - develop independent TPC patrec
    - use KalTest for track fit

KalTest Kalman fitter package

- KalTest
  - Kalman Fitting library (Keisuke Fujii et al)
    - recently migrated code base to SVN
    - added cmake build scripts
- KalDet
  - detector description (geometry and material) for KalTest
    - migrated to SVN
    - currently writing the geometry build up from GEAR
- released in iLCSoft release v01-10 !
- both packages are used by LCTPC (MarlinTPC) and ILD / iLCSoft
  - -> try to share as much common code as possible, i.e. is reasonable given the slightly different requirements for testbeam and global detector optimization

KalTest library

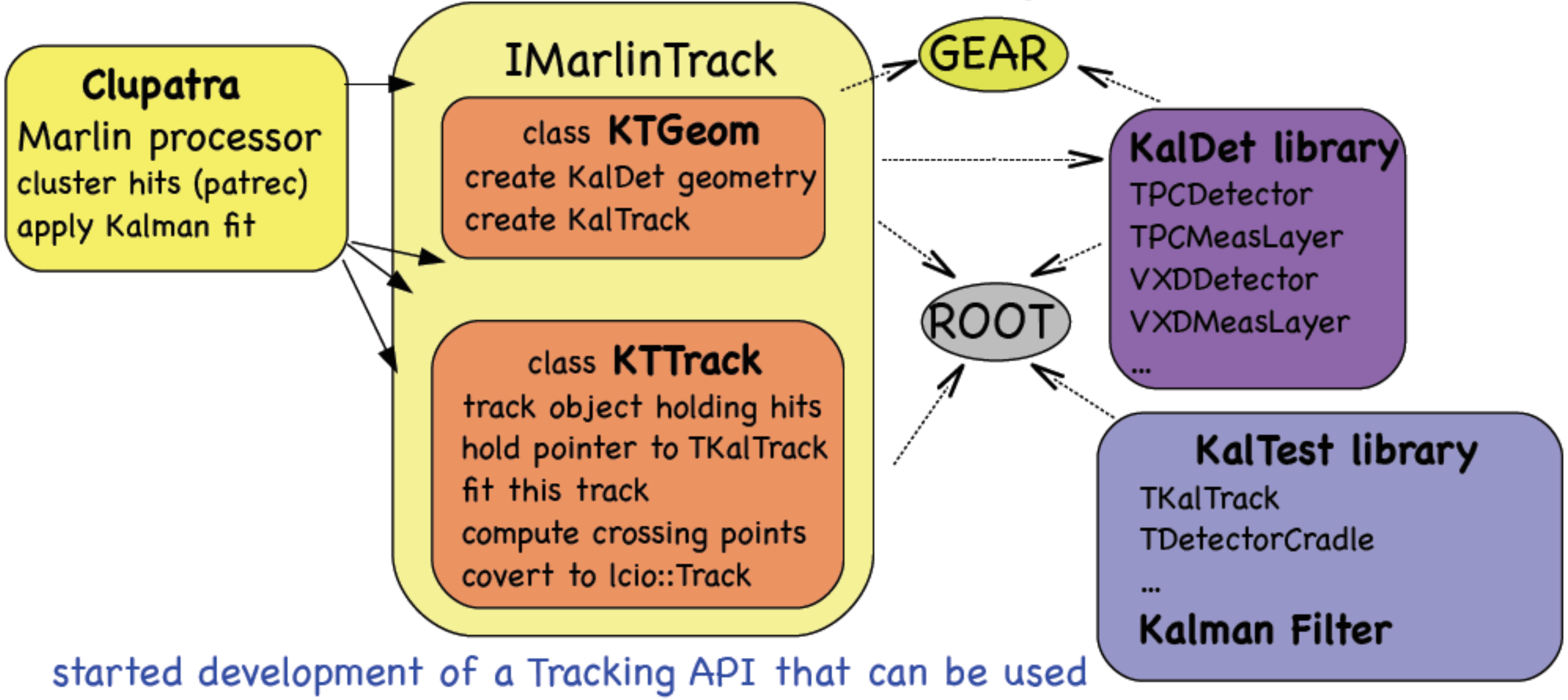
- based on ROOT
  - TGeo, TMath, TObjArray
- structured in sub-libraries
  - geomlib – geometry
  - kallib – Kalman filter
  - kaltracklib – Kalman tracking
  - utils – utilities
- built into one libKalTest.so
- users need to define their detector classes (KalDet):
  - TVMeasLayer
    - meas. layer, coordinate to track state transform. …
  - TVDetector
    - position of meas. layer and material properties


R
z
IP
y
X
(x_V, y_V, z_V)
d_V
ϕ_V
Hit
Helix
$x = x_0 + d_\rho \cos\phi_0 + \frac{\alpha}{\kappa}(\cos\phi_0 - \cos(\phi_o + \phi))$
$y = y_0 + d_\rho \sin\phi_0 + \frac{\alpha}{\kappa}(\sin\phi_0 - \sin(\phi_o + \phi))$
$z = z_0 + d_z - \frac{\alpha}{\kappa}\tan\lambda \cdot \phi$
(a) Negative Track
(b) Positive Track
(x, y, z)
$\rho>0$
$\rho<0$
$\phi>0$
$\phi<0$
$\rho+d\rho>0$
$\rho+d\rho<0$
(x_c, y_c)
(x_0, y_0, z_0)
ϕ_0
y
x

- track parameters correspond to LCIO, except:
  - d0_lcio = – drho_kaltest
  - omega_lcio = a(cB) * kappa_kaltest
  - phi_lcio = phi_kaltest + PI/2
- Kaltest adopted to use LCIO units: mm, Tesla, GeV

interface to KalTest

- need interface to KalTest fitter
- would like to have loose coupling between patrec and fitting
- need several iterations between patrec and fitting
  - LCIO::Track class not optimal for that (not designed to be)



Clupatra
Marlin processor
cluster hits (patrec)
apply Kalman fit

IMarlinTrack

class **KTGeom**
create KalDet geometry
create KalTrack

class **KTTrack**
track object holding hits
hold pointer to TKalTrack
fit this track
compute crossing points
covert to lcio::Track

GEAR

ROOT

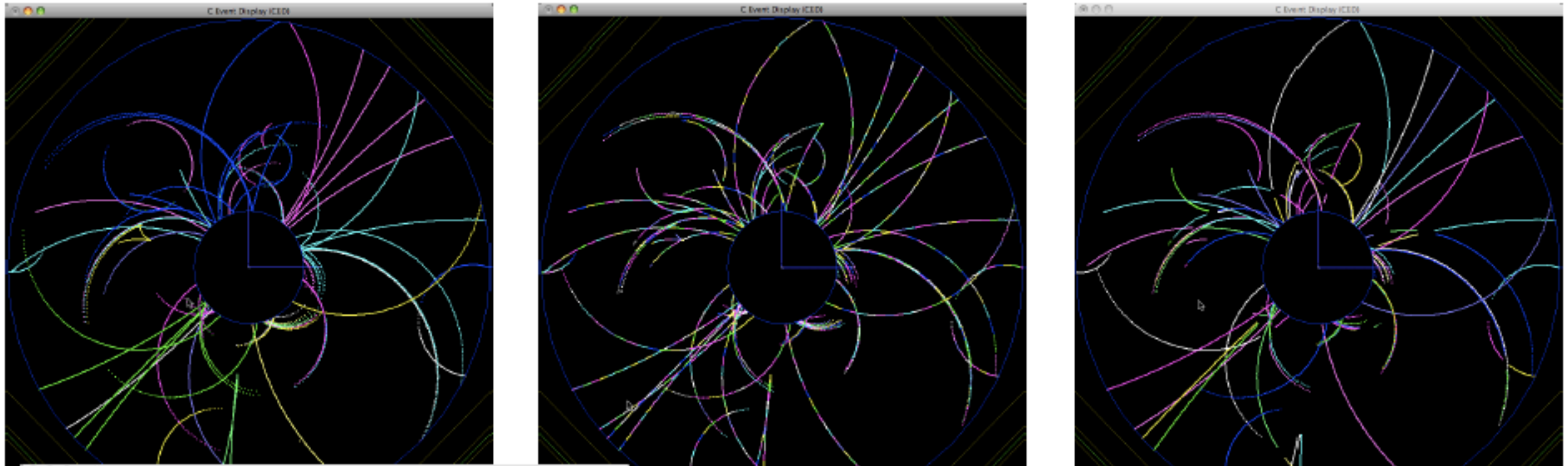
KalDet library
TPCDetector
TPCMeasLayer
VXDDetector
VXDMeasLayer
...

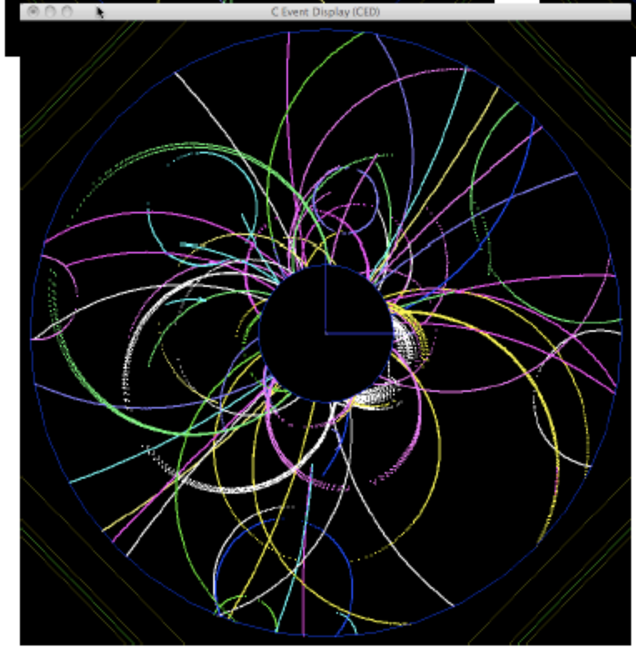
KalTest library
TKalTrack
TDetectorCradle
...
Kalman Filter

started development of a Tracking API that can be used to write patrec in Marlin transparent to underlying track fitting code – now implemented with KalTest

package dependency: ⇢
using class: →

Clustering based patrec in TPC

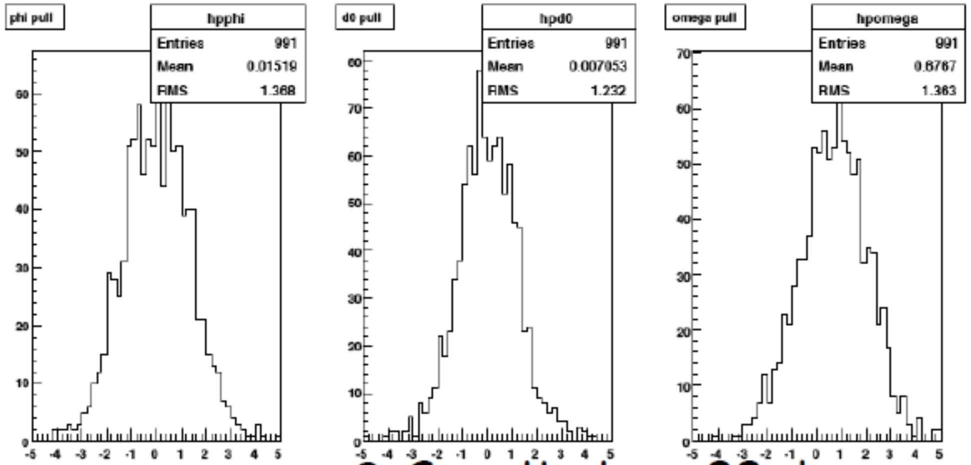




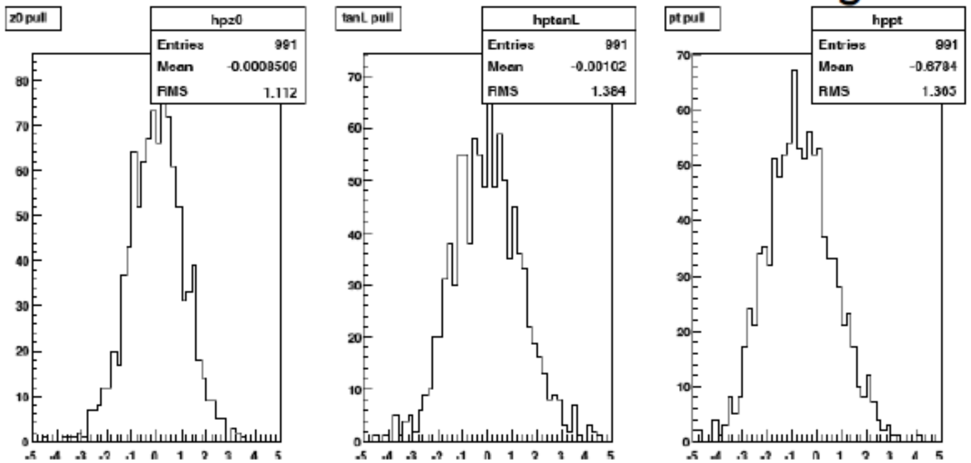
- use NN-Clustering w/ euclidian distance
- cleanup merging regions with pad-row ranges
- assign leftover hits:
  - based on residuals wrt. extrapolated X-ing
  - improved assignment since ILCWS2010
- working on segment merging:
  - based on simple 'circle criterion'
  - implemented chi2 for track state comparison:
    - SUM_i [(P0_i – P1_i)^2 / (sig0^2+sig1^2)]
    - not yet optimal -> work in progress

Improved track fit

- at ILCWS2010 shown that pulls where still wrong
  - fixed some issues in the code
  - moved to new consistent units in KalTest (mm, Tesla, GeV)
  - added errors from LCIO hits
    - parameterization from LCTPC group [F(phi, theta, pt)]
  - introduced dummy material layers (cylinders) for
    - SIT and VXD



mu+ , 3 Gev, theta = 88 deg



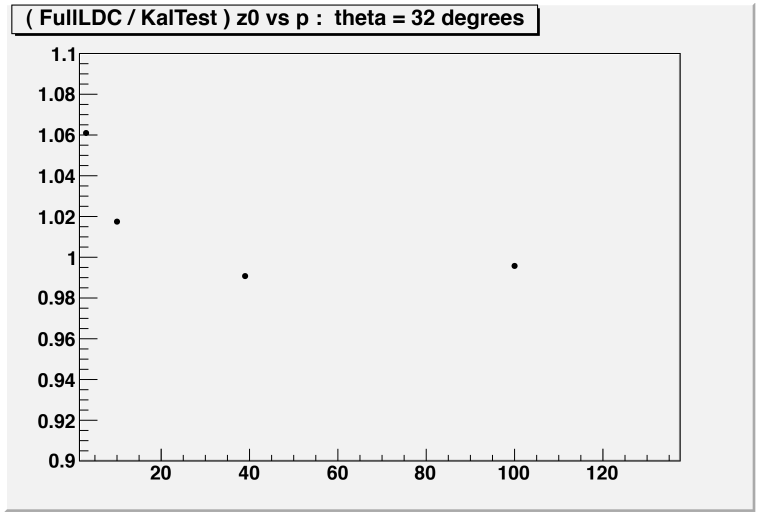
- pulls look more reasonable
- sigma still slightly too large
- bias in pT

-> disentangle material description from fit (code) issues: use TPC only events ...

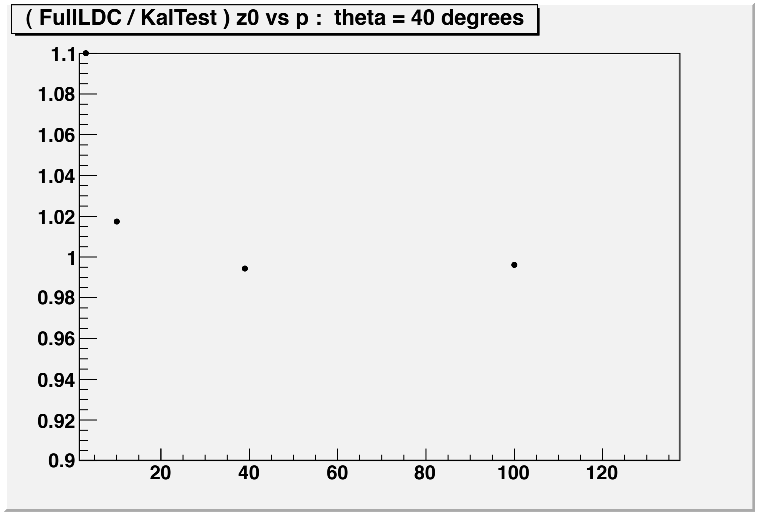
Track Parameter Comparison

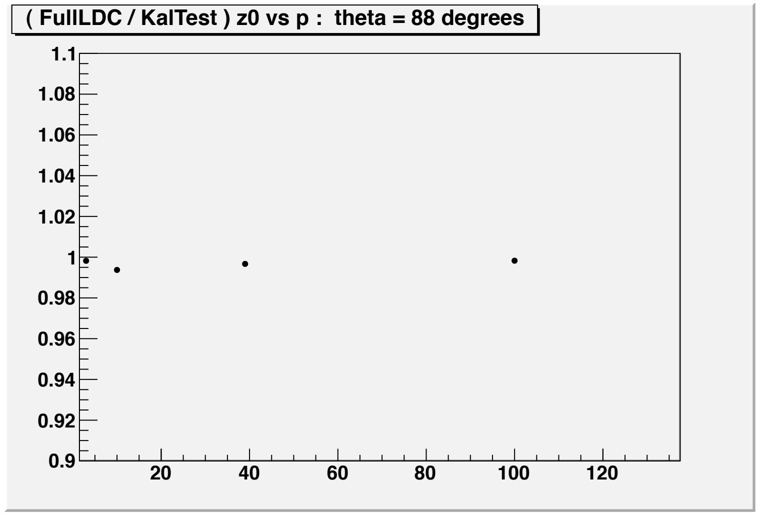
- Created a simple refitting Processor to test development of the Tracking API as well as Track Parameter and error determination in the KalTest implementation
  - Takes lcio Tracks produced by LEPTracking and FullLDCTracking and refits the associated hits using the Kaltest Kalman Filter.
  - Presently fits are only determined at the IP
- Testing performed using ILD_00 TPC with inner detectors removed in Mokka.
- Comparison made with Track Parameters and errors determined by FullLDCTracking using single muons at p = 3, 6, 40, 100 GeV and theta = 88, 40, 32 degrees

Δz0 (KalTest / LDC) vs p GeV

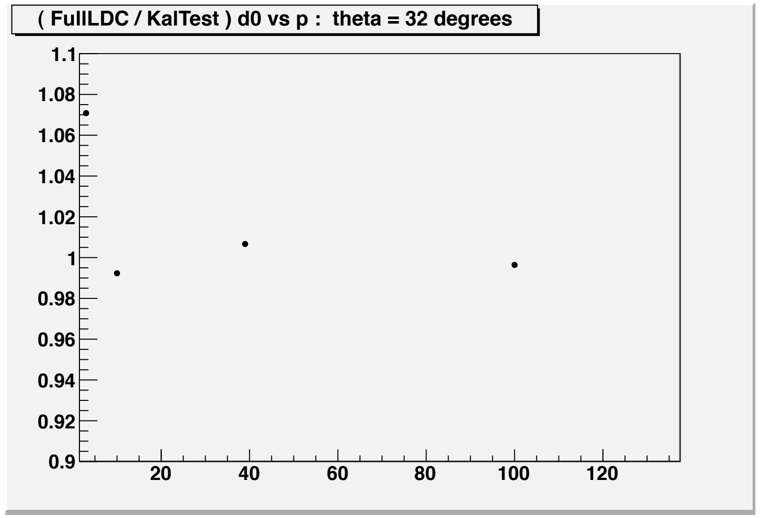


KalTest / LDC
(plots wrongly labelled)

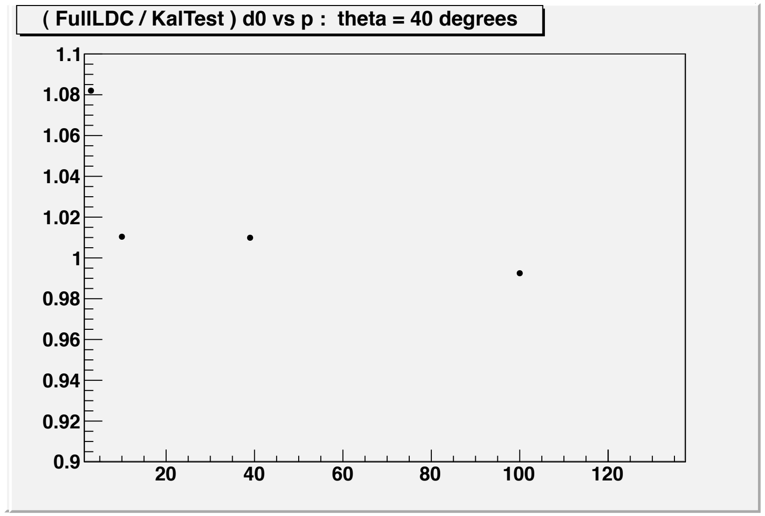


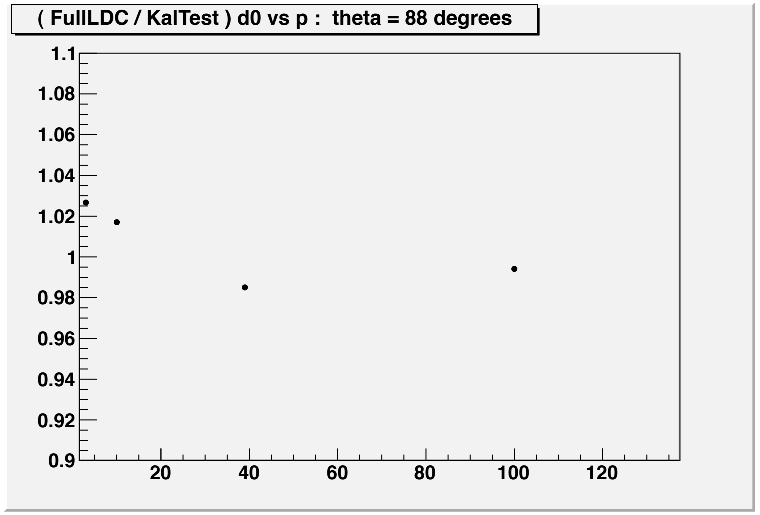


Δd0 (KalTest / LDC) vs p GeV

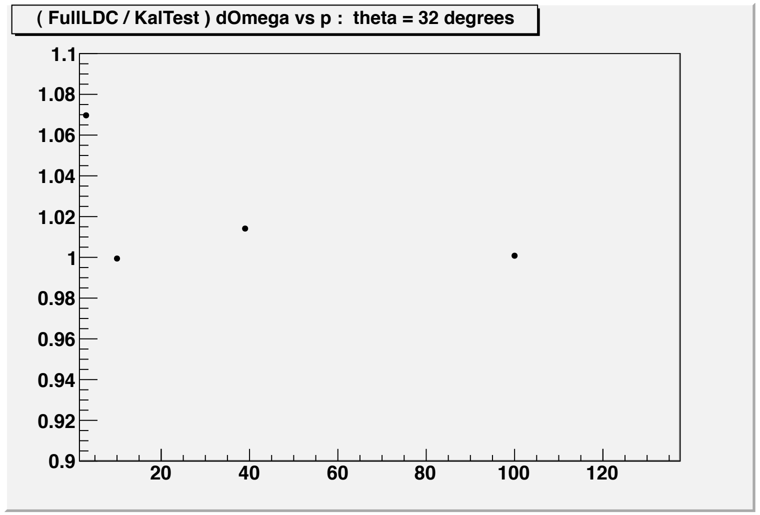


KalTest / LDC
(plots wrongly labelled)

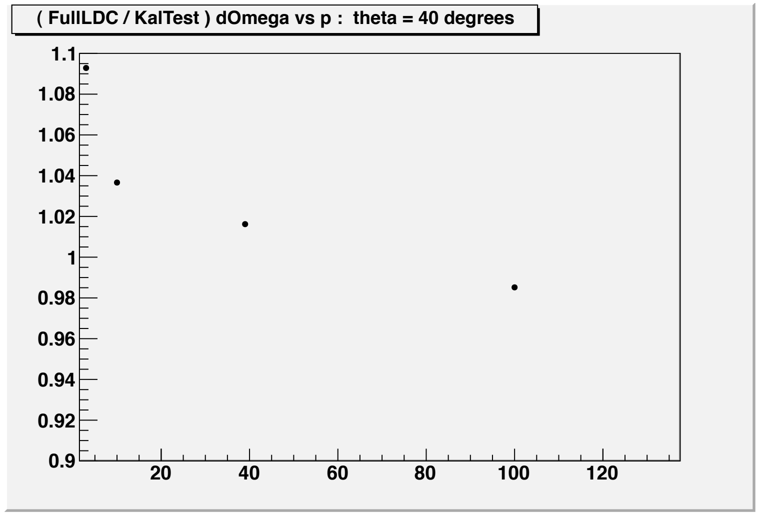


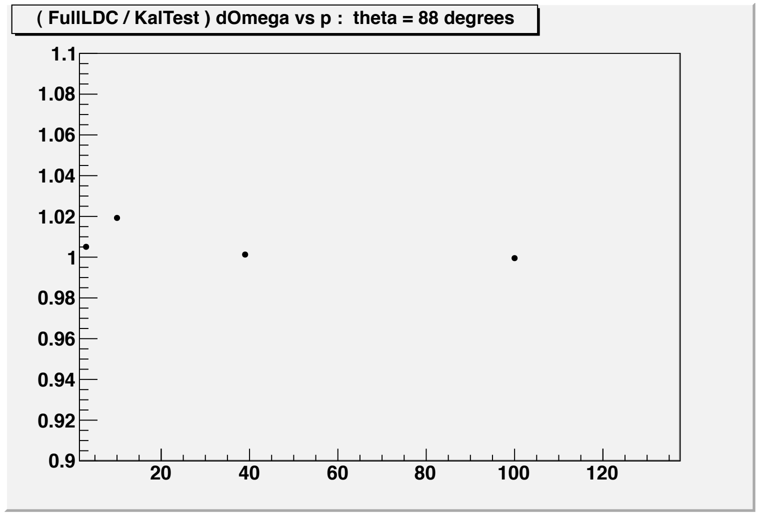


ΔOmega (KalTest / LDC) vs p GeV

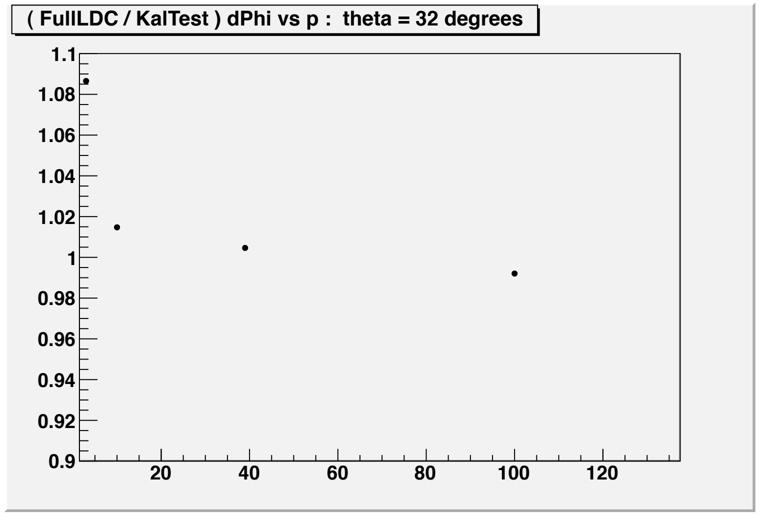


KalTest / LDC
(plots wrongly labelled)

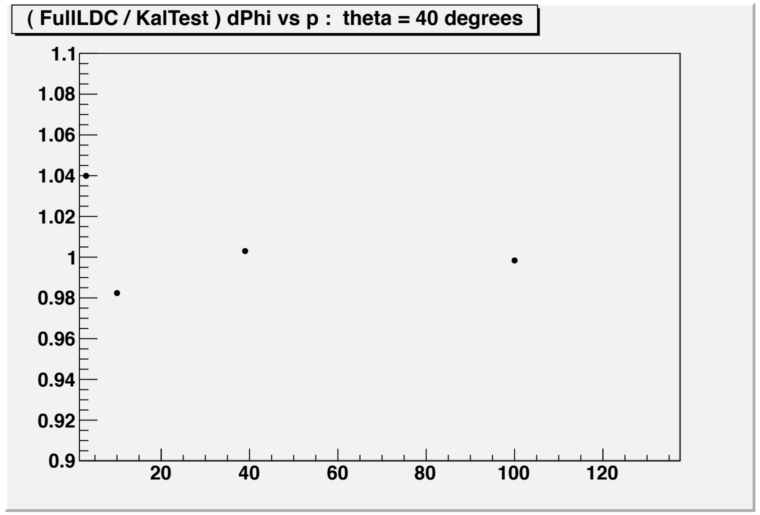


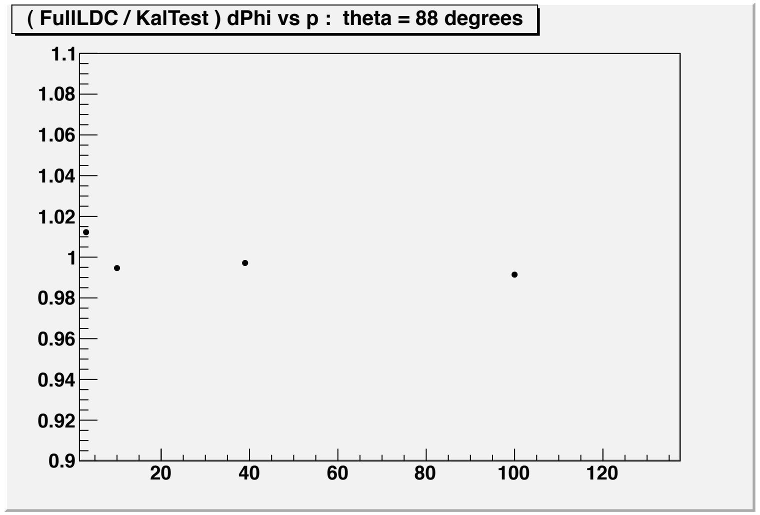


ΔPhi (KalTest / LDC) vs p GeV

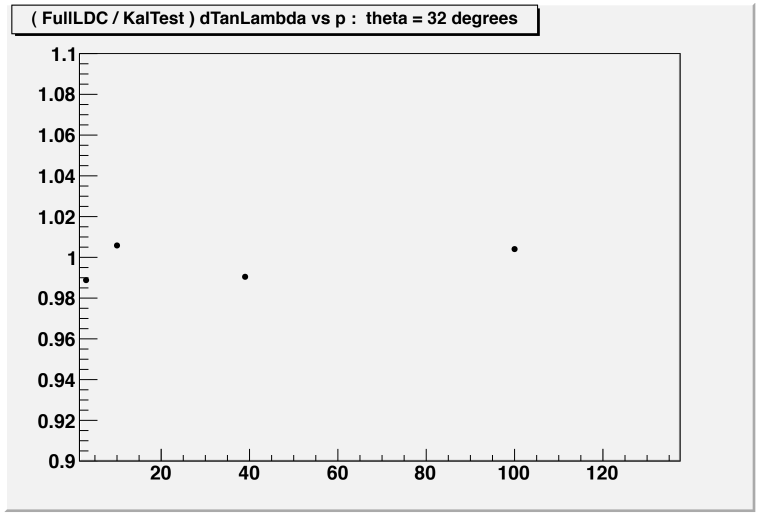


KalTest / LDC
(plots wrongly labelled)

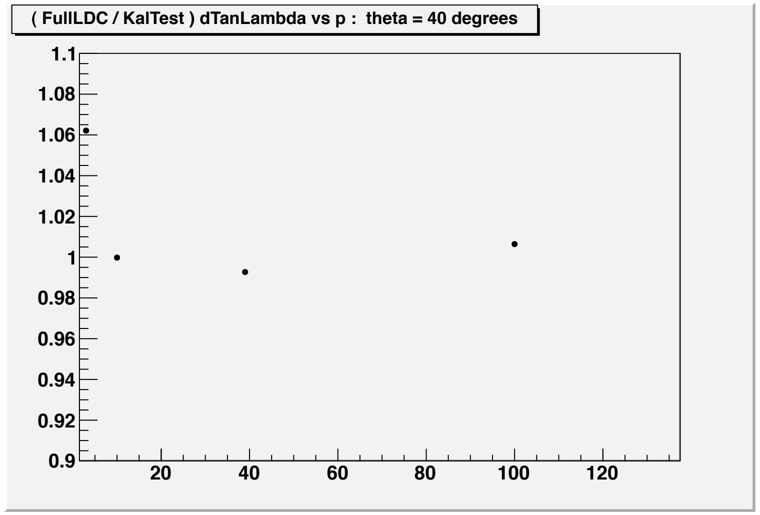


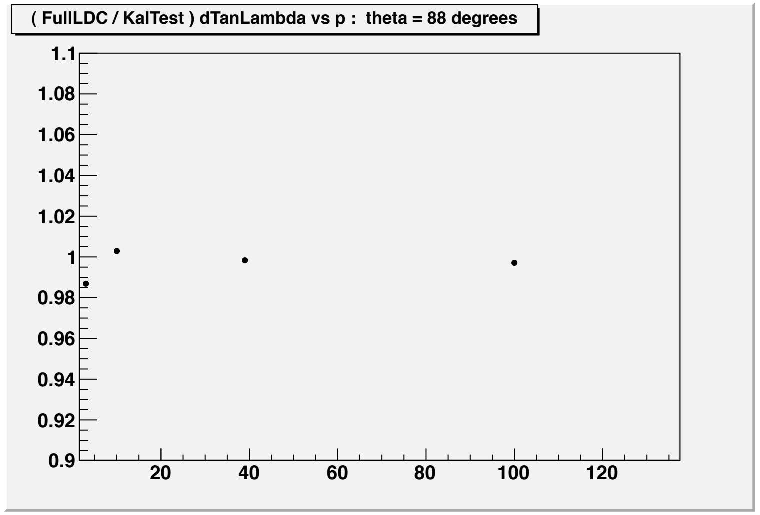


Δtanλ (KalTest / LDC) vs p GeV

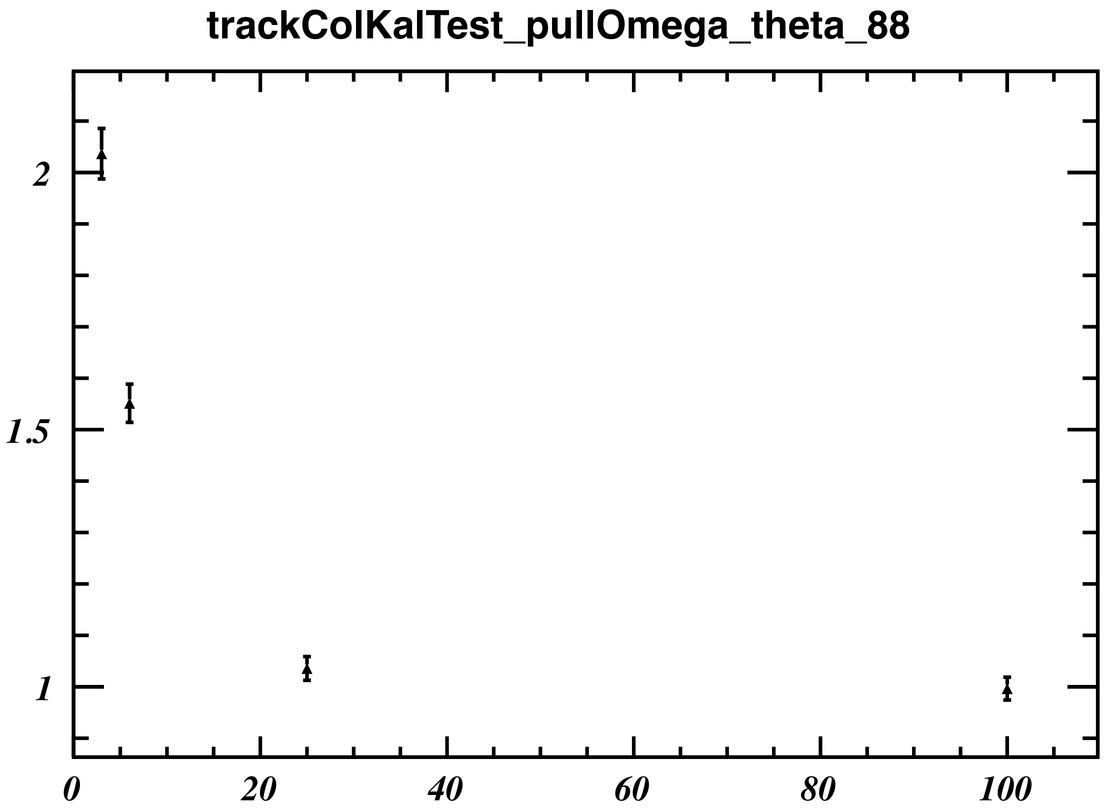


KalTest / LDC
(plots wrongly labelled)

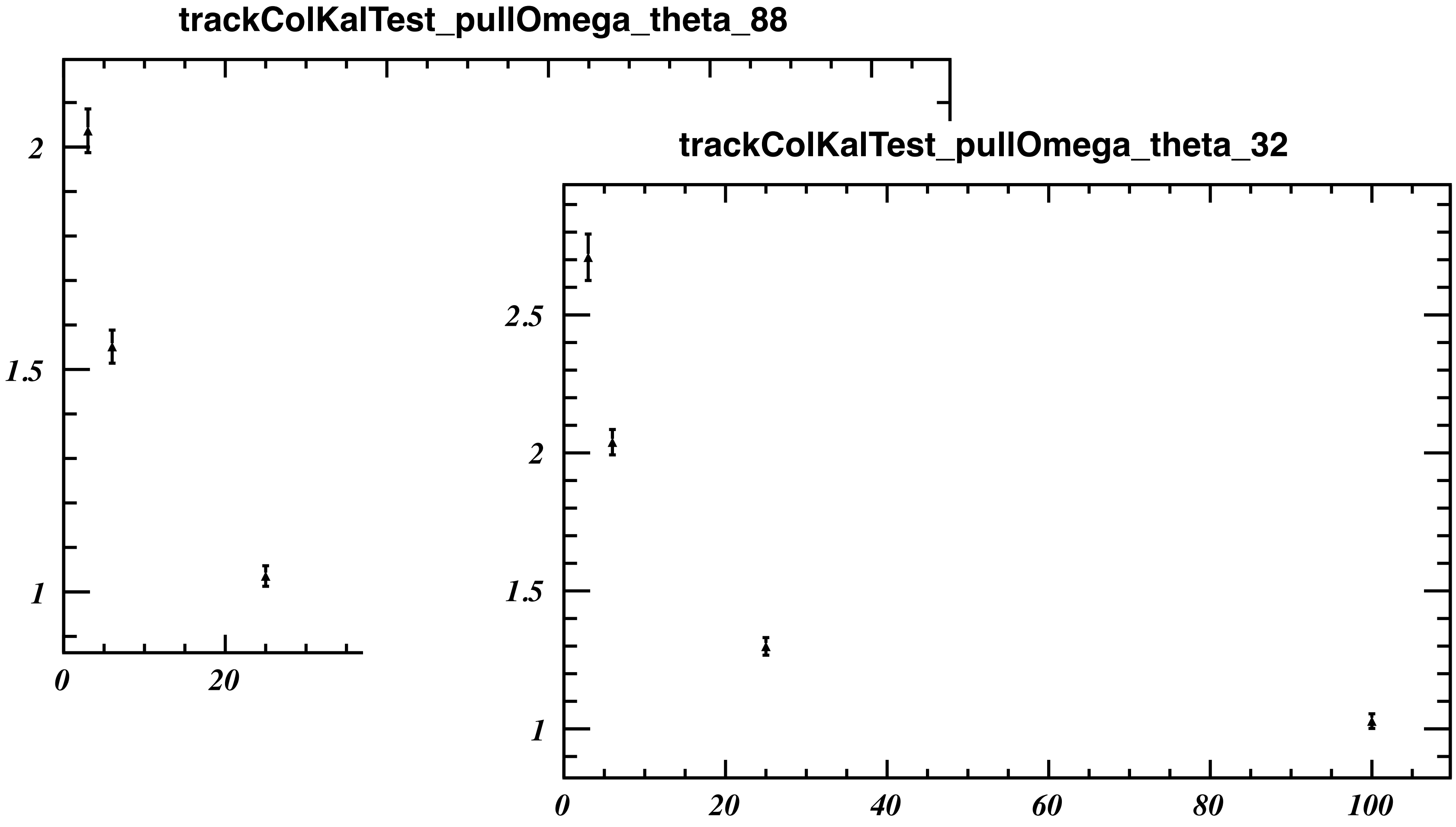




Kaltest Pulls Omega vs p



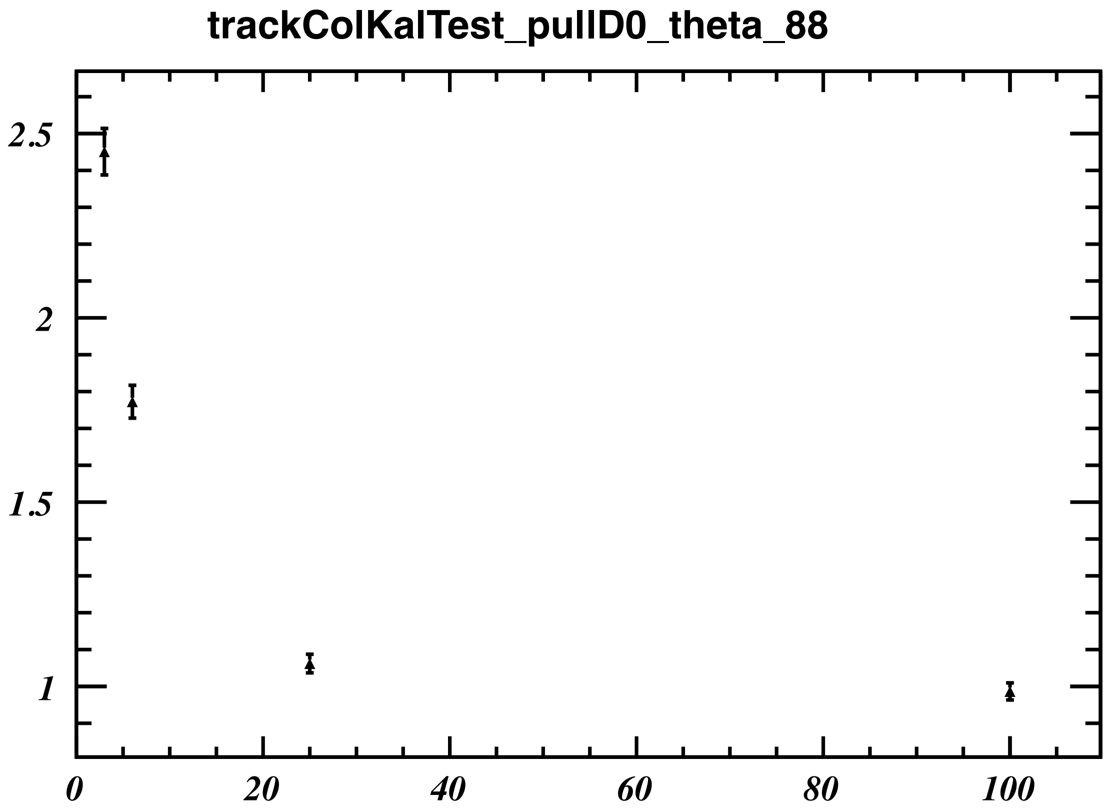
Kaltest Pulls Omega vs p



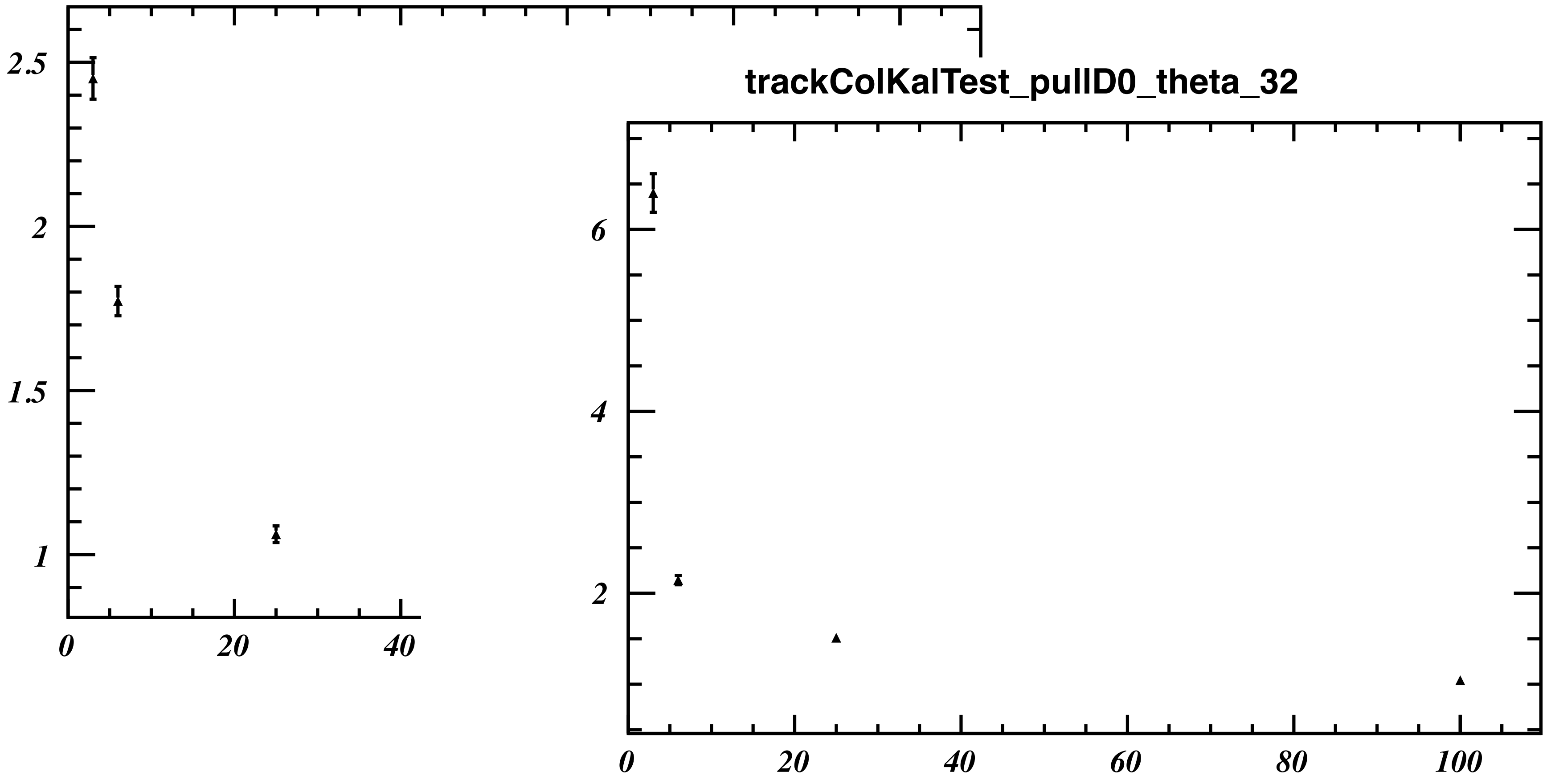
trackColKalTest_pullOmega_theta_88

trackColKalTest_pullOmega_theta_32

Kaltest Pulls d0 vs p



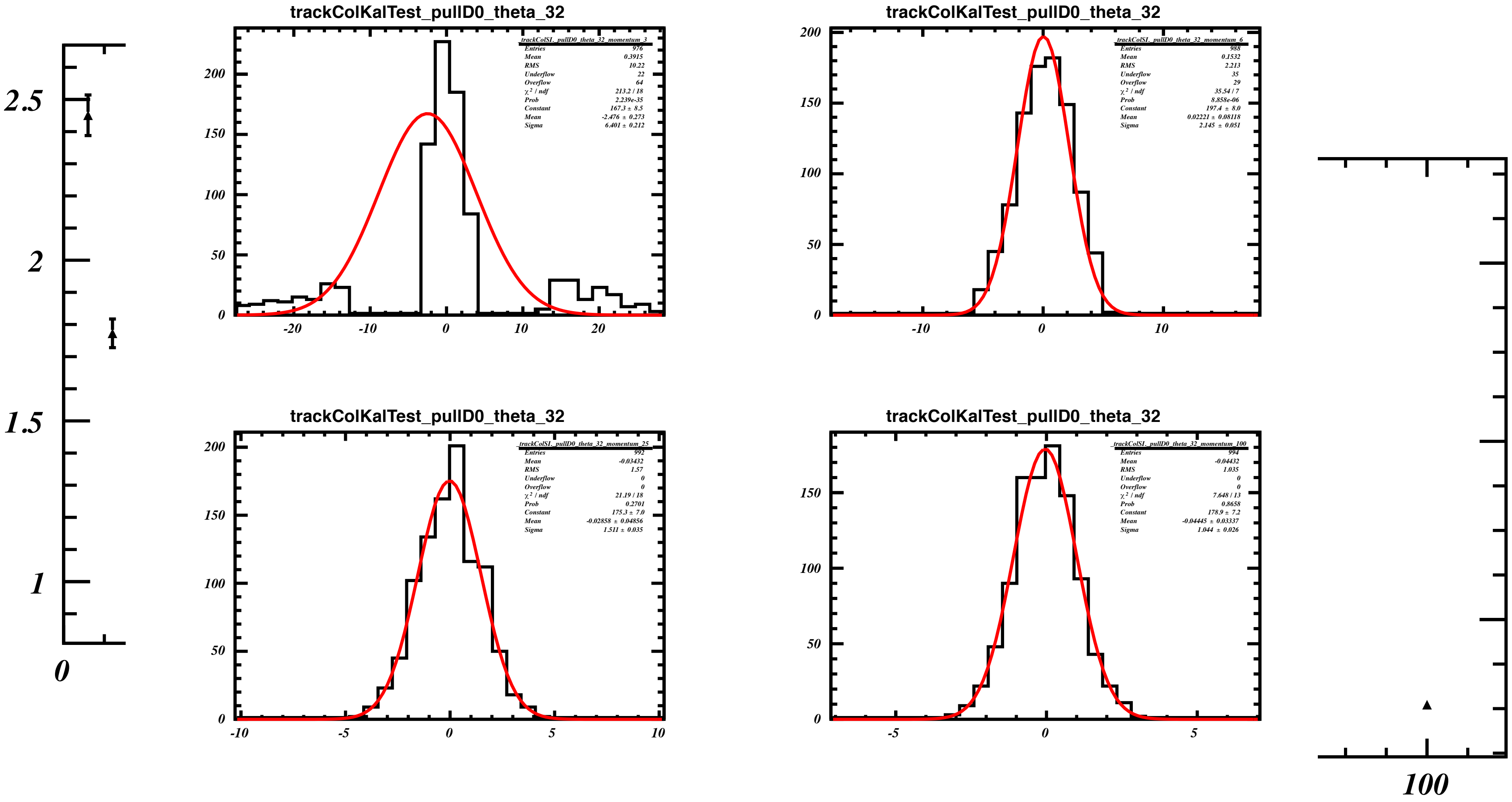
Kaltest Pulls d0 vs p



trackColKalTest_pullD0_theta_88

trackColKalTest_pullD0_theta_32

Kaltest Pulls d0 vs p



Summary

- Started work on a Tracking API for use in Marlin.
- So far minimally implemented using KalTest and TPC.
- The addition of further tracking systems needs the implementation of bounded planar detectors in KalTest.
- Geometry and material budget needs tuning in order to improve Track Parameter errors.
- TPC Pat-Rec currently working on merging of track-segment found.
- Working on improving the diagnostics.