# genfit2:
# a general track fitting tool

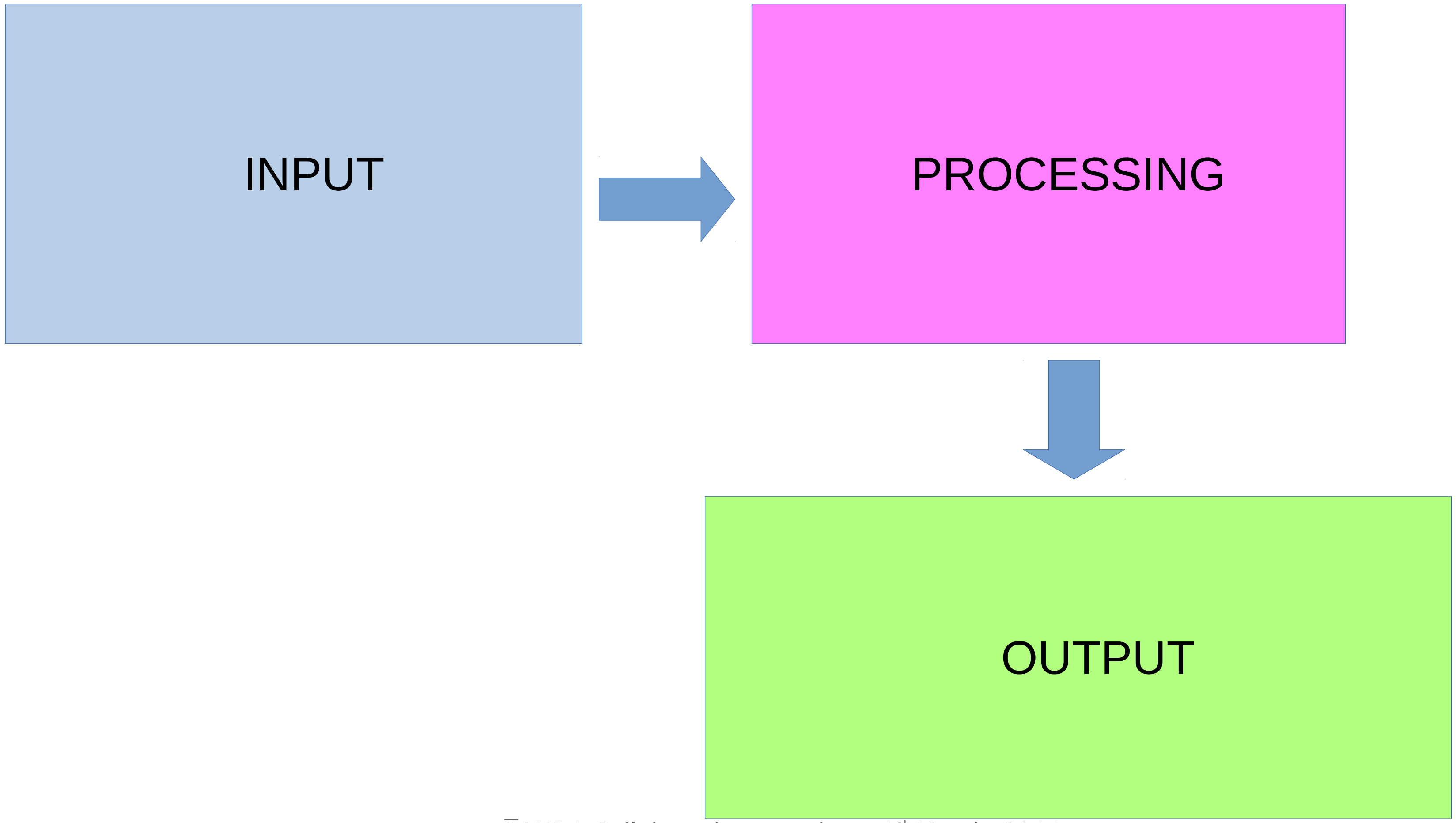
Elisabetta Prencipe, Forschungszentrum Jülich
XVI $\overline{\text{P}}$ANDA Collaboration Meeting
1st March 2016

- Main feature of genfit2 are given
- Comparison with old genfit features
- Why shall we use genfit2?

I am going to show the talk presented at the
Common Tracking software forum +
a couple of useful cases from PandaRoot tests

- genfit2:
general tracking fitting tool
Authors: J. Rauch, T. Schlüter, arXiV:1410.3698[physic.ins-det]

- genfit2 handles all aspects of track fitting....or is something still missing?



- genfit2:

general tracking fitting tool

Authors: J. Rauch, T. Schlüter, arXiV:1410.3698[physic.ins-det]

- genfit2 handles all aspects of track fitting....or is something still missing?

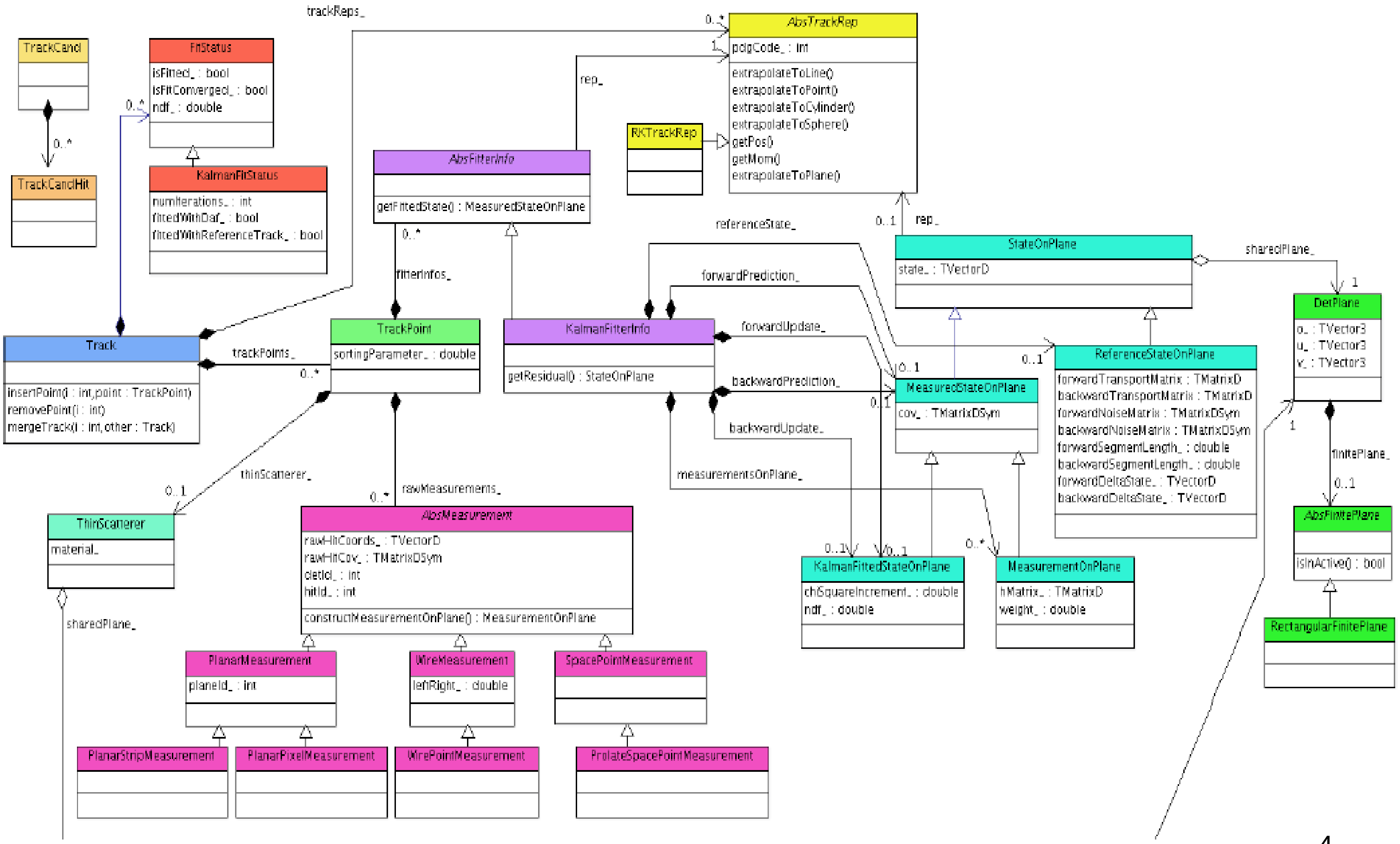
- Detector geometry
- Flexible classes
- Track candidate handling
- Interfaces between tracks, hits, alignment information

- Extrapolation code
- Fitting algorithms: KF, DAF, GBL
- Flexible convergence criteria

- Storage fitted tracks (ROOT)
- Interface with Millepede II and RAVE
- Visualization (eventDisplay)

# genfit2 structure

arXiV:1410.3698



## genfit2 design

- Measurements
  - PlanarMeasurement
  - WireMeasurement
  - SpacePointMeasurement
  - Virtual detector plane, measurement coordinates, covariance
- Track Representations
  - Track parameterization
  - Extrapolation through material and magnetic field
  - Propagate time of flight
  - Particle hypothesis
- Track fitting algorithms
  - Use measurements and TrackReps to calculate fit results
  - Start value for fit needed, e.g. from pattern recognition

## Basic class definitions

- Track:
  contain measurements;
  measurements can be from different detectors;

  *Track* can be fitted with different trackReps
  (e.g., can be fitted with different particle hypothesis)

- Measurements:
  serve as objects containing measured coordinates from a detector.
  Base class: *AbsMeasurement*

- *TrackPoints* can contain *measurements*, *FitInfo objects*,....

- *TrackCand* serves as helper class:
  it stores indices of raw detector hits in *TrackCandHits* objects

- *WireTrackCandHits* objects can store the left/right ambiguity

## Fitters and representations

- Tracking is the core of physics analysis.
- PandaRoot is the official framework of $\overline{\text{P}}$ANDA @ FAIR.
- PandaRoot has made use of the Kalman Filter for track finding/ fitting procedures.
- Kalman equations in PandaRoot have been provided by an external package: ***genfit.*** Update: ***genfit2***

- ***Kalman Filter*** = set of mathematical equations
  – It uses a series of measurements observed over time, containing **noise** (random variables) and other inaccuracies, and produces estimates of unknown variables.
  – Iterative least square method in 2 steps: prediction, and update.
  – Many applications: used in HEP since  more than 10 years.

- Track representation in genfit2: Runge-Kutta.
- ***Runge-Kutta method***: family of iterative methods, used in temporal discretization for the approximation of solutions of ordinary differential equations.

- Kalman is the fitter: needs state prediction and measurements to calculate the state update;
- task of track representation: to determine the state prediction by extrapolating the previous update to the next measurement.

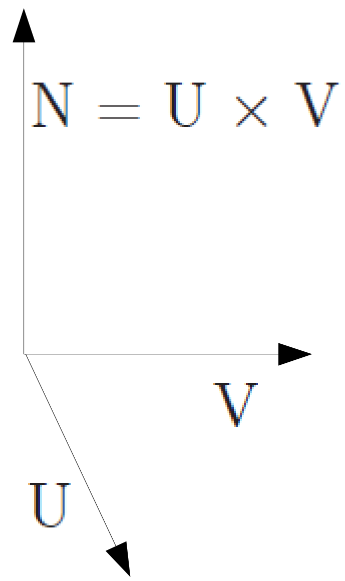
# Coordinate system

- 3 different sets of variables to characterize a trajectory:

6D-coordinates $(\mathbf{x},\ \mathbf{p})$ + q define the helix

Local coordinates $(q/p, u', v', u, v)$

Global coordinates $(\mathbf{x},\ \mathbf{T},\ \mathrm{q/p})$



- Representation by helix [BaBar]: $(d_0, \phi, \omega, z_0, tan\lambda)$

u', v' = direction cosines relative to the plane;
u, v = local coordinates to the plane;
q = charge;
**X** = position coordinates;
**P** = momentum cordinates;
*p* = momentum module;
T = direction unit vector.

# Testing a PandaRoot trunk-revision with genfit2

- Basic variables checked: px, py, pz, e, x, y, z
- Need to test:
  - ▸reconstructed variables
  - ▸true values
  - ▸error distributions
  - ▸reconstruction efficiency vs $p_T$
  - ▸pull of the variable distributions

- Main parameters checked in this talk:

z0, d0 = $\mathrm{Sqrt}(x^2 + y^2)$, curvature ($\propto Q/p_t$), $\tan\lambda$ ( $p \cdot \cos\lambda = pt$), $\phi$

Residual = $\mathrm{var}_{\mathrm{reco}} - \mathrm{var}_{\mathrm{gen}}$

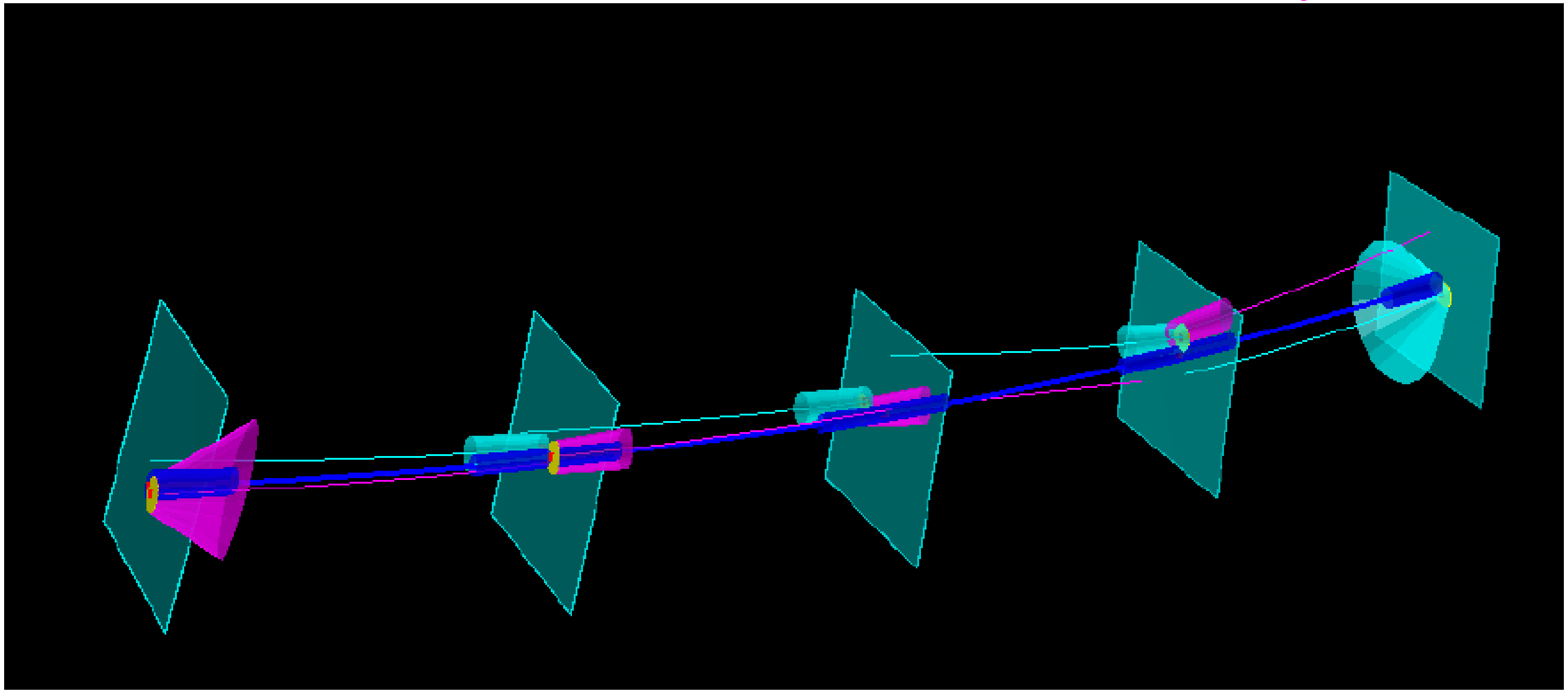
Pull = $(\mathrm{var}_{\mathrm{reco}} - \mathrm{var}_{\mathrm{gen}}) / \mathrm{err}_{\mathrm{reco}}$

## Fitting algorithms: Kalman Filter

- Prediction step: extrapolate state and covariance to next measurement
- Update step: calculate a weighted average between prediction and measurement
- Prediction + update: iterate over measurements, forth and back, until convergence

- Linearization. Kalman filter is a linear estimator: need to linearize the transport

Courtesy of J. Rauch



Smoothed track: weighted average between forward fit and backward fit.

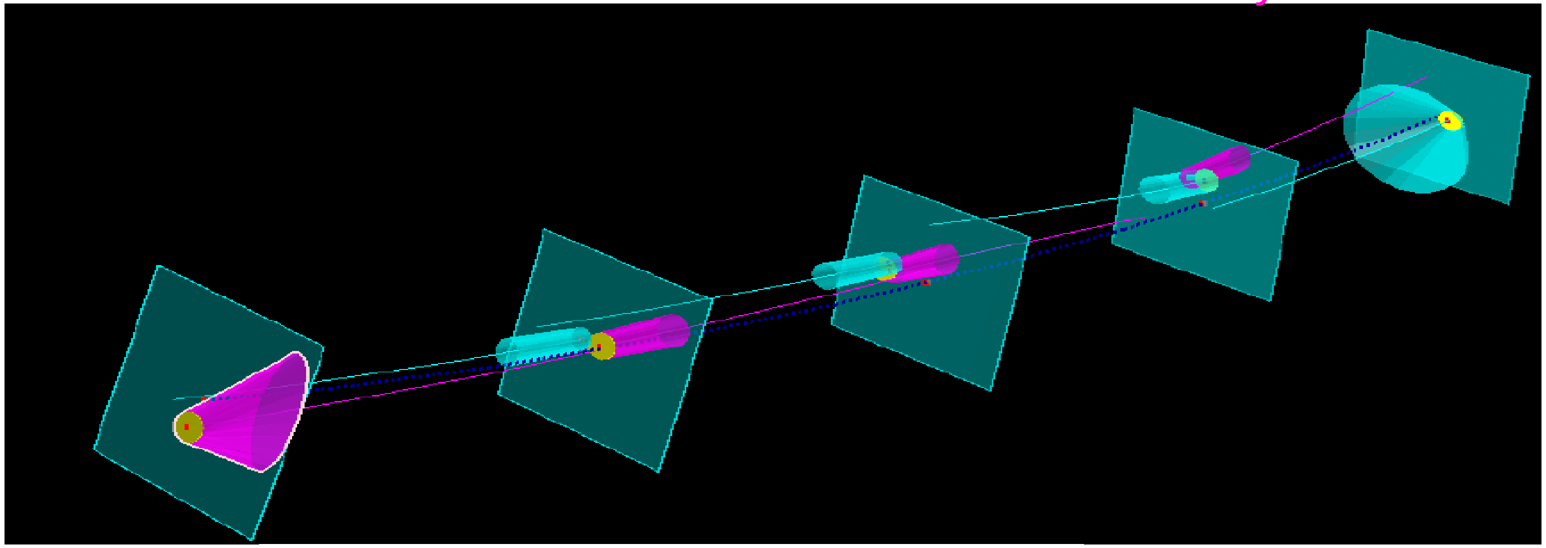
## Fitting algorithms: Kalman Filter with reference track

- Problems when linearizing around predictions:
  state predictions can be far off the real trajectory (e.g., first few hits); outliers can bend the prediction away.

- Consequence: linearization makes not sense, and fit can fail

- Solution: use reference track ⇒ Take estimated track parameters from pattern recognition or previous fit as expansion point for linear approximation (i.e., linearize around reference track, instead of state prediction)

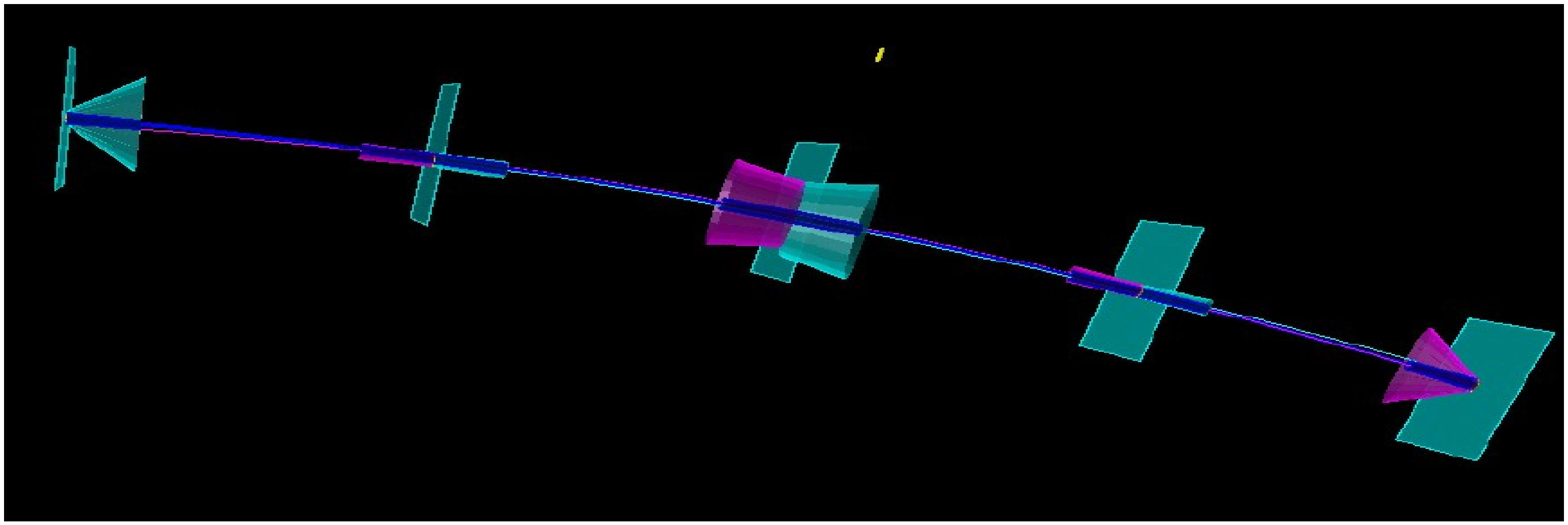
Courtesy of J. Rauch


Reference track, forward- and backward fit.

## Fitting algorithms: Deterministic annealing fitter (DAF)

- Robust track fitter
- Produces assignment probabilities of measurement (e.g., weights)
- Iterative Kalman filter with weighting and annealing to find the best fit
- Can reject outliers or resolve left/right ambiguities of WireMeasurements.

Courtesy of J. Rauch


after 6 iterations

## DAF: How does the weighting procedure work?

- Useful to solve left/right ambiguities
- Weight of the *MeasurementOnPlane* must be initialized
- Basic solution: to assign both left and right measurements a weight = 0.5

- Wire positions are taken as measurements in the first iteration: covariance is twice the mean of the individual covariances ⇨
  all wire positions have same covariace ⇨
  systematic false estimate of the covariance biases the fit.

- Novel technique implemented in genfit2:
  measurements with larger drift radii are assigned smaller weights ⇨ larger cov.;
  measurements with smaller drift radii are assigned larger weights ⇨ smaller cov.

efficiency improvement

## Fitting algorithms: Generalized broken line (GBL)

- GBL is implemented for using Millepede II
- Mathematically equivalent to the Kalman fitter

- Feature mainly introduced for Belle II

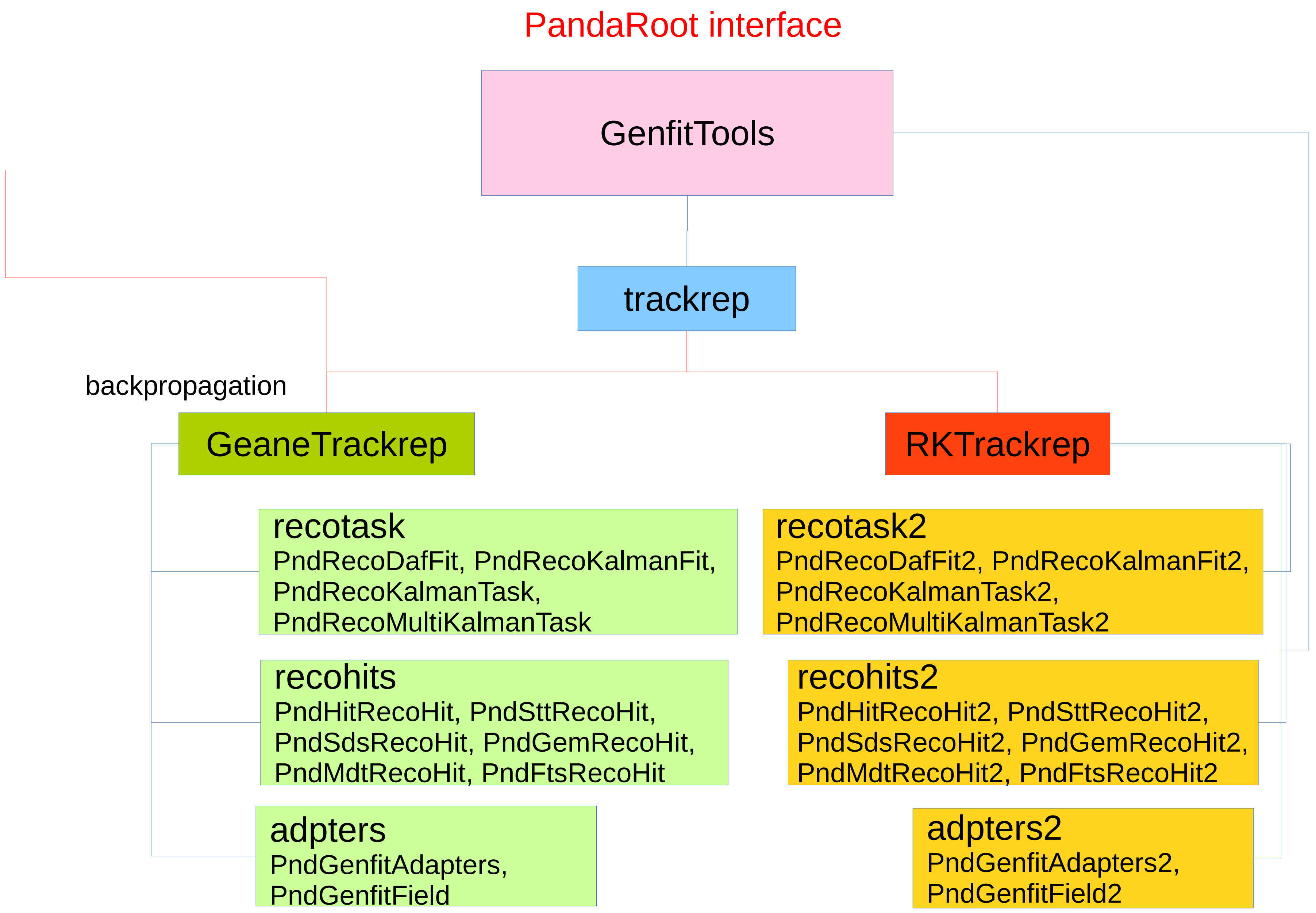
## The Runge-Kutta representation

- *RKTrackRep* based on extrapolator from Geant3
- An abstract interface class interacts with detector geometry
- During fitting, material properties are used to calculate:
  - Energy loss
  - Energy loss straggling according to Bethe-Block formula
      (code ported from Geant3)
  - Multiple scattering (Highland-Lynch-Dahl) formula
  - Bremsstrahlung energy loss and energy loss straggling for electrons

- Field inhomogeneity and curvature taken into account

- Provide different methods to find the POCA of the tracks to non planar measurements

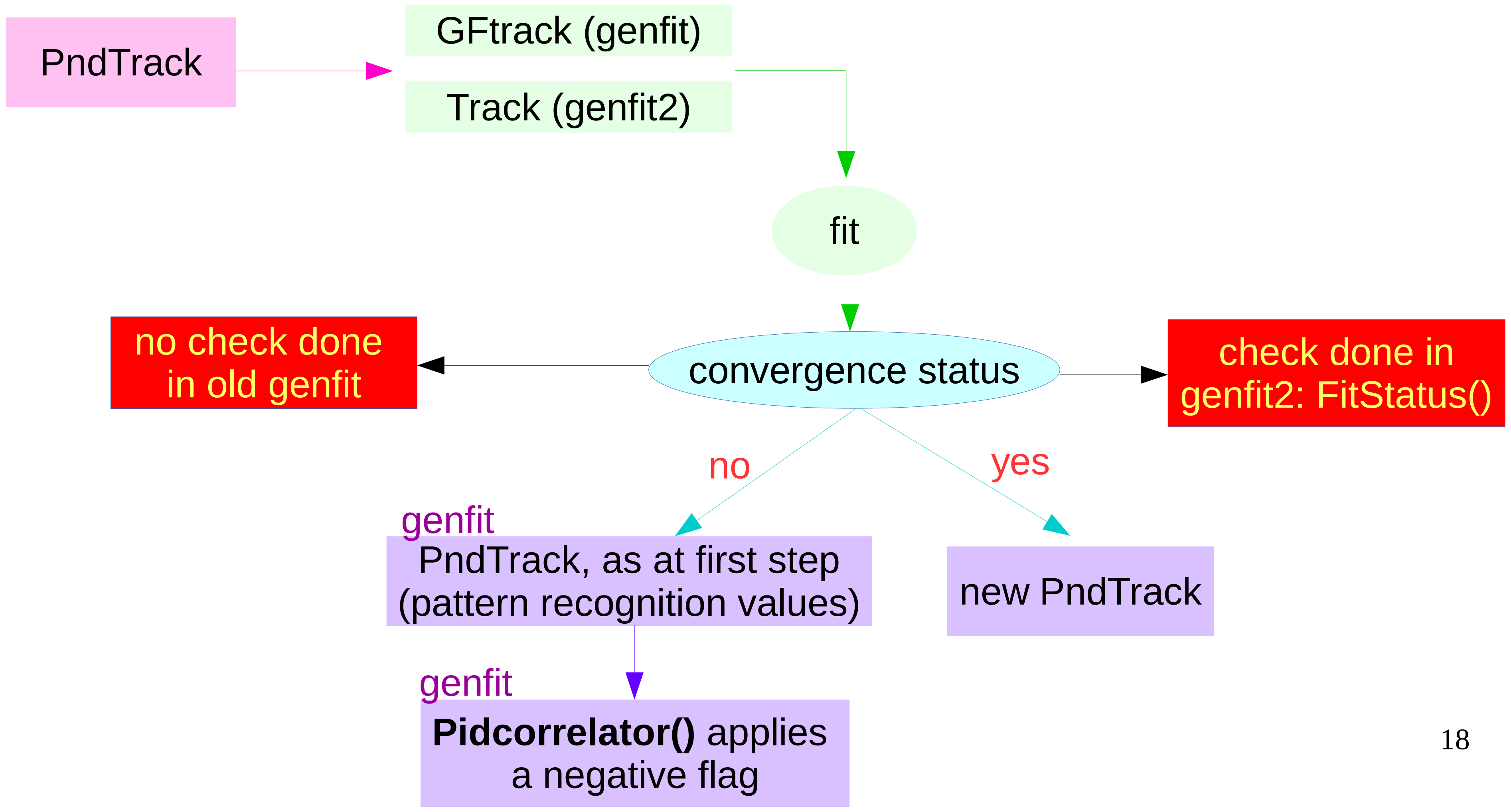
# Structure of  GenfitTool in PandaRoot



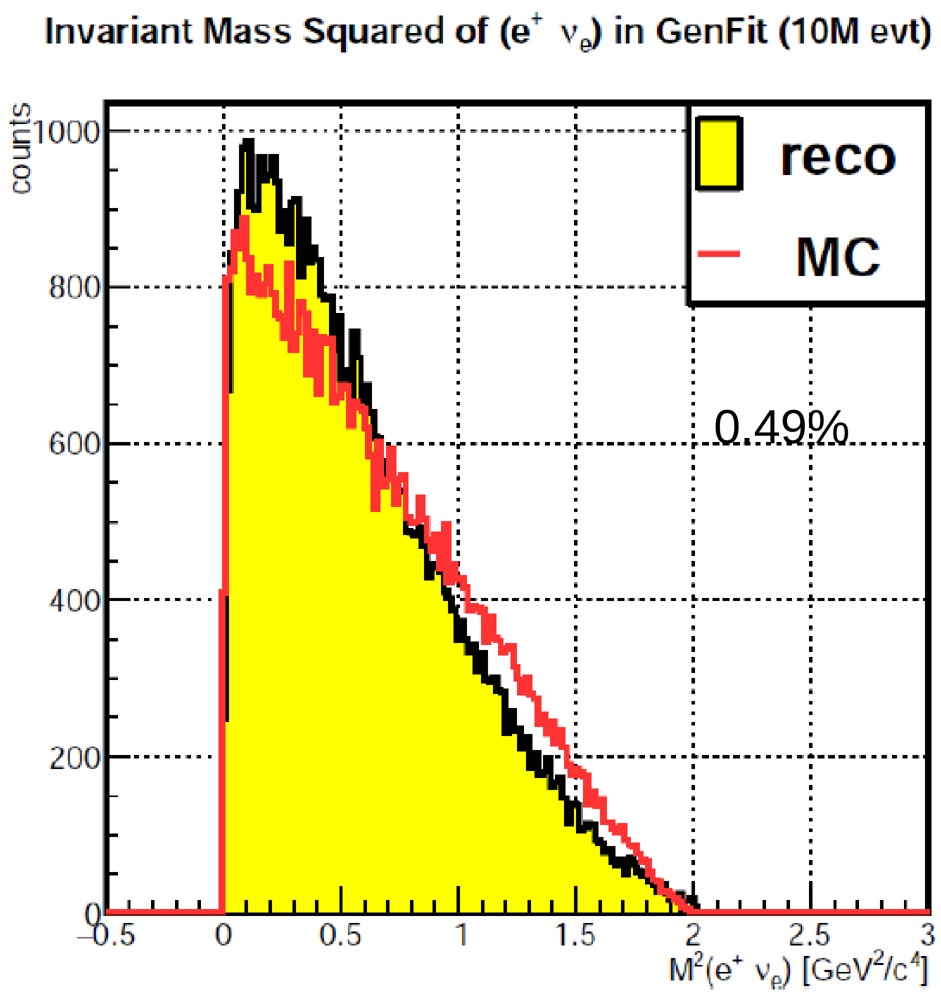
PandaRoot interface
GenfitTools
trackrep
backpropagation
GeaneTrackrep
RKTrackrep
recotask
PndRecoDafFit, PndRecoKalmanFit, PndRecoKalmanTask, PndRecoMultiKalmanTask
recotask2
PndRecoDafFit2, PndRecoKalmanFit2, PndRecoKalmanTask2, PndRecoMultiKalmanTask2
recohits
PndHitRecoHit, PndSttRecoHit, PndSdsRecoHit, PndGemRecoHit, PndMdtRecoHit, PndFtsRecoHit
recohits2
PndHitRecoHit2, PndSttRecoHit2, PndSdsRecoHit2, PndGemRecoHit2, PndMdtRecoHit2, PndFtsRecoHit2
adpters
PndGenfitAdapters, PndGenfitField
adpters2
PndGenfitAdapters2, PndGenfitField2

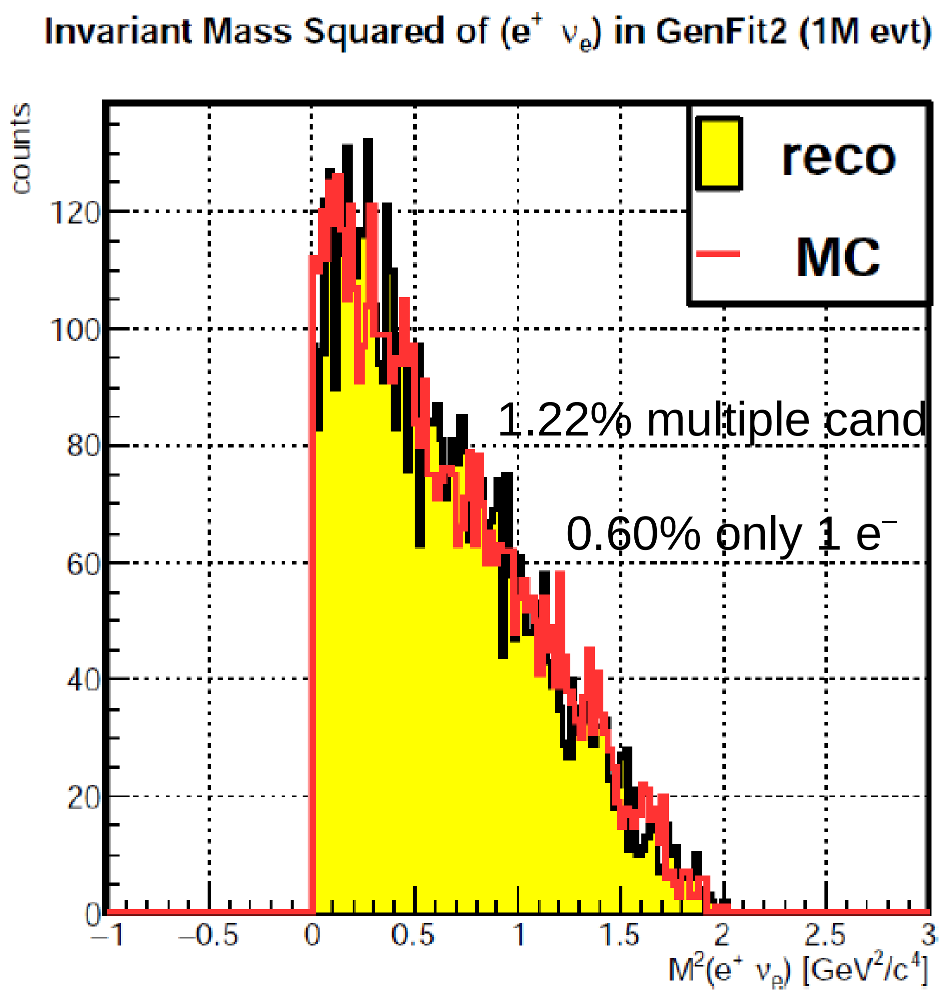
# Fitting tracks in PandaRoot

- Process in 3 steps:
  PndTrack converted in genfit(2) track; genfit(2) track fitted; then, transformed to PndTrack


PndTrack
GFtrack (genfit)
Track (genfit2)
fit
convergence status
no check done in old genfit
check done in genfit2: FitStatus()
no
yes
genfit
PndTrack, as at first step (pattern recognition values)
new PndTrack
genfit
**Pidcorrelator()** applies a negative flag

# Improvement in analysis with genfit2



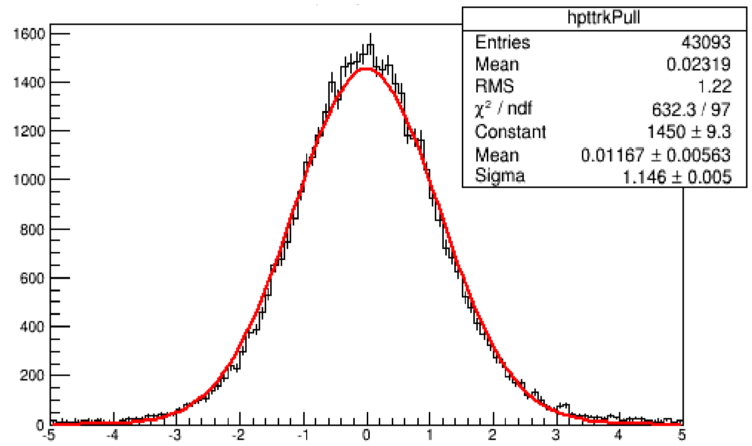


$$\bar{p}p \rightarrow D_s^+ D_s^-$$

$$D_s^+ \rightarrow \eta e^+ \nu_e,\ \eta \rightarrow \pi^+\pi^-\pi^0,\ \text{and}\ D_s^- \rightarrow K^+K^-\pi^-$$

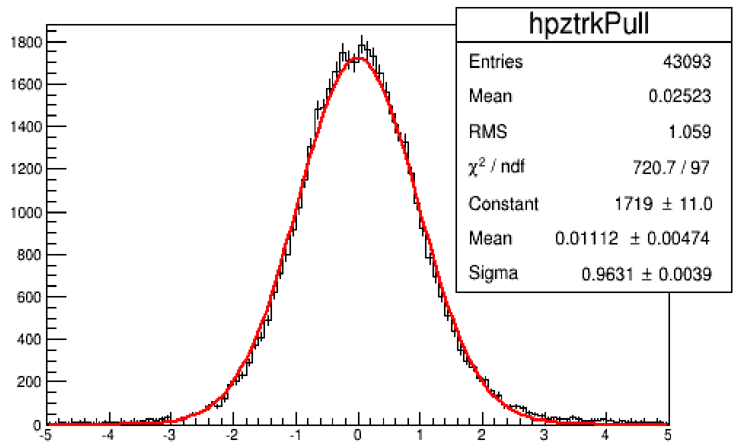
# Testing genfit2: Pull

$P_{beam}$ = 15 GeV/c; N = 50 000 $\pi^+$, **$p_t$ = 1 GeV/c**; reconstruction efficiency = (86.18 ±0.15)%



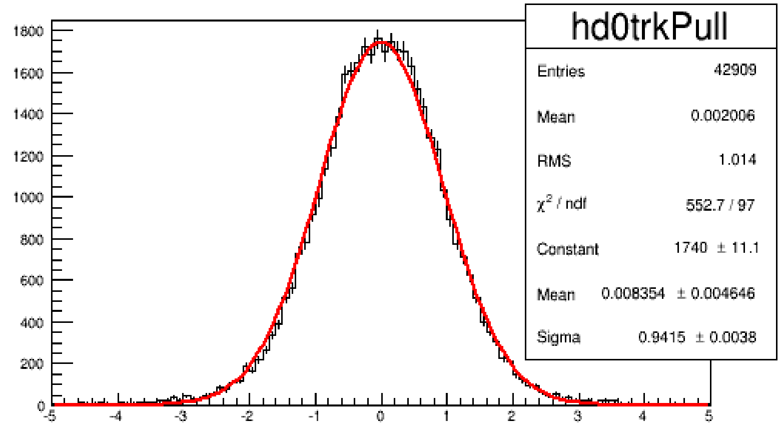
| hpttrkPull | |
|---|---|
| Entries | 43093 |
| Mean | 0.02319 |
| RMS | 1.22 |
| $\chi^2$ / ndf | 632.3 / 97 |
| Constant | 1450 ± 9.3 |
| Mean | 0.01167 ± 0.00563 |
| Sigma | 1.146 ± 0.005 |

$P_t$ pull



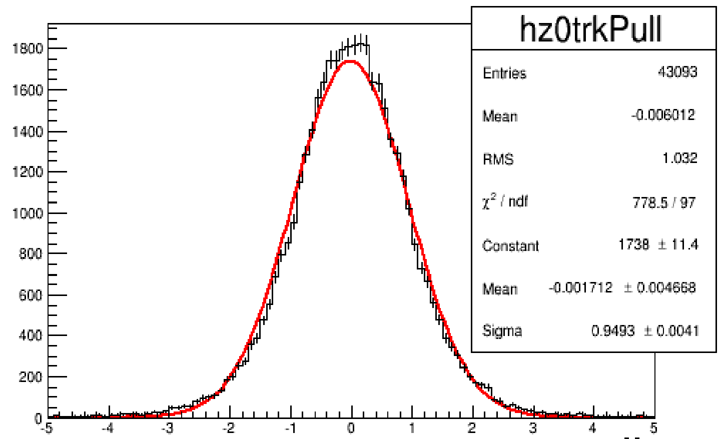
| hpztrkPull | |
|---|---|
| Entries | 43093 |
| Mean | 0.02523 |
| RMS | 1.059 |
| $\chi^2$ / ndf | 720.7 / 97 |
| Constant | 1719 ± 11.0 |
| Mean | 0.01112 ± 0.00474 |
| Sigma | 0.9631 ± 0.0039 |

$P_z$ pull



| hd0trkPull | |
|---|---|
| Entries | 42909 |
| Mean | 0.002006 |
| RMS | 1.014 |
| $\chi^2$ / ndf | 552.7 / 97 |
| Constant | 1740 ± 11.1 |
| Mean | 0.008354 ± 0.004646 |
| Sigma | 0.9415 ± 0.0038 |

$d_0$ pull



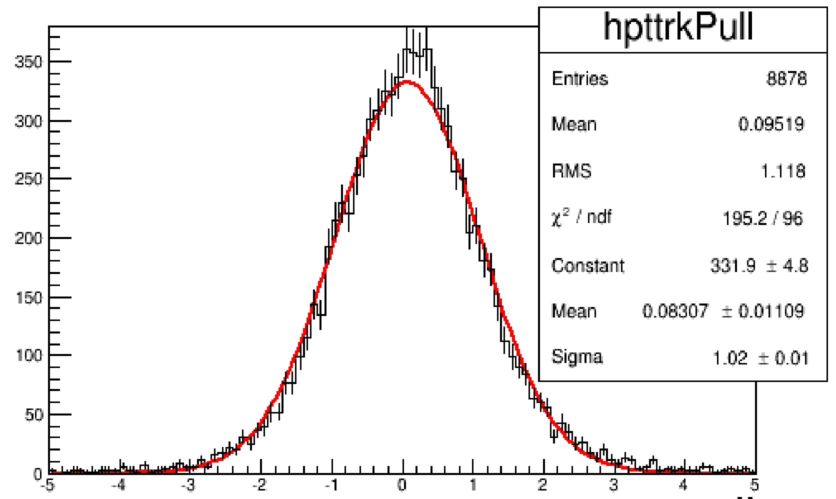
| hz0trkPull | |
|---|---|
| Entries | 43093 |
| Mean | -0.006012 |
| RMS | 1.032 |
| $\chi^2$ / ndf | 778.5 / 97 |
| Constant | 1738 ± 11.4 |
| Mean | -0.001712 ± 0.004668 |
| Sigma | 0.9493 ± 0.0041 |

$z_0$ pull

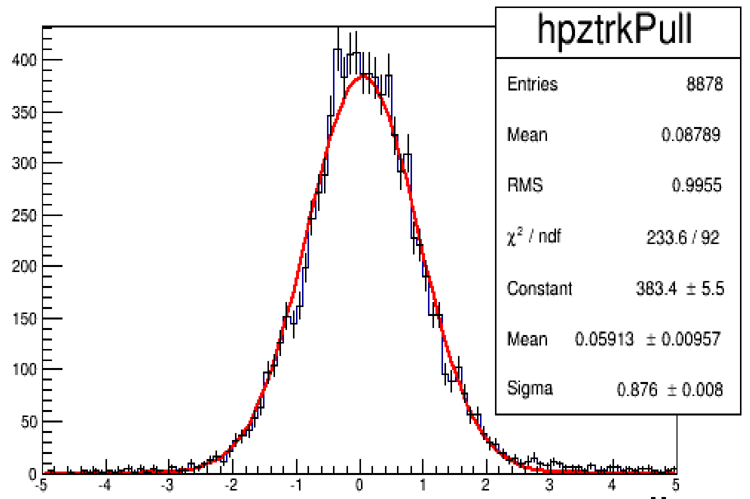
- Pull of tracking parameters is supposed to be have gaussian distribution, with $\sigma$ = 1

# Testing genfit2: Pull

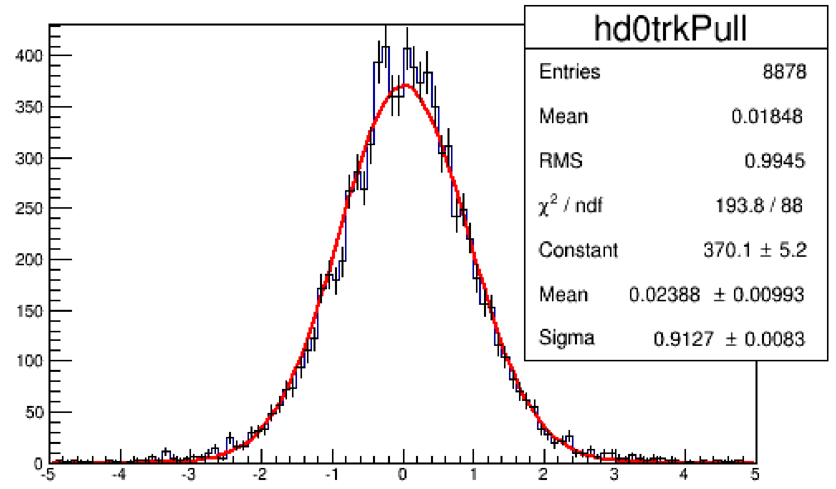
$P_{beam}$ = 15 GeV/c; N = 10 000 $\pi^+$, **$p_t$ = 0.4 GeV/c**; reconstruction efficiency = (85.54 ±0.92)%



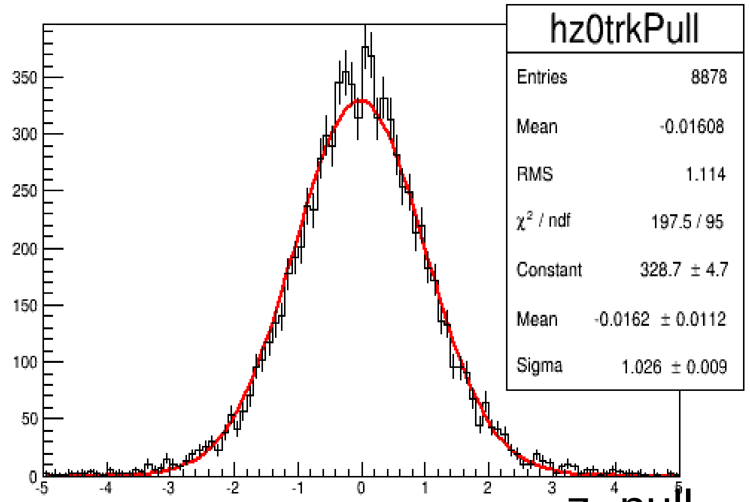
$P_t$ pull



$P_z$ pull



$d_0$ pull

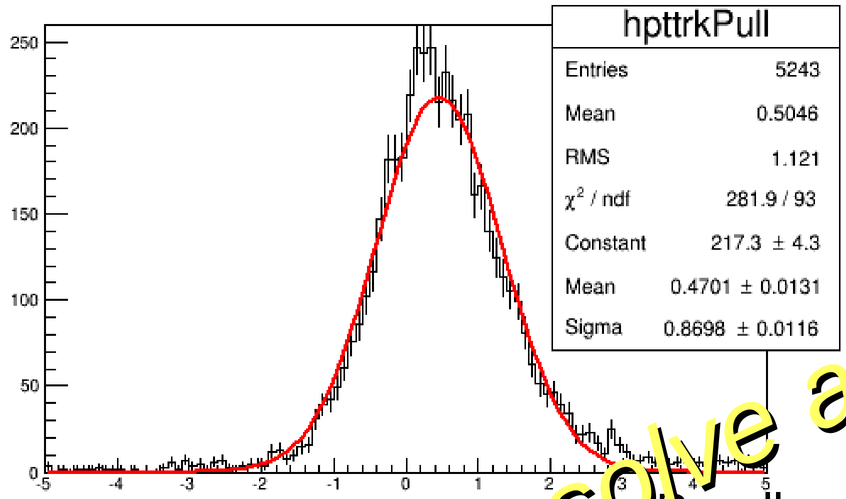


$z_0$ pull

# Testing genfit2: Pull

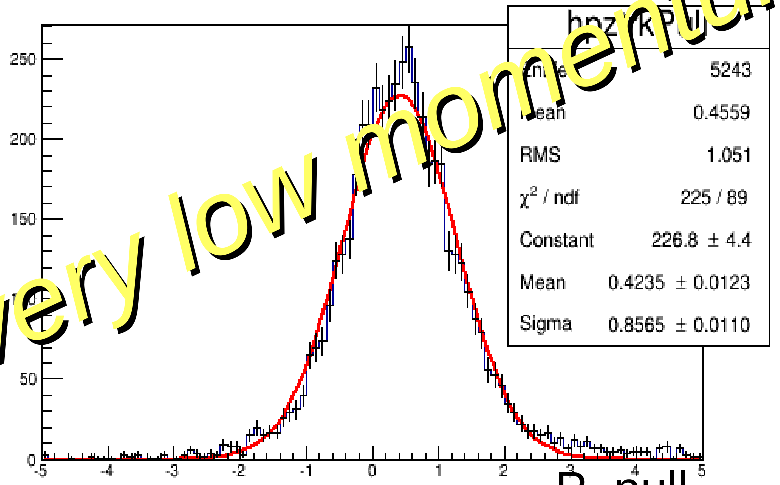
$P_{beam}$ = 15 GeV/c; N = 10 000 $\pi^+$, **$p_t$ = 0.15 GeV/c**; reconstruction efficiency = (50.26 ±0.71)%

Still problems to solve at very low momentum



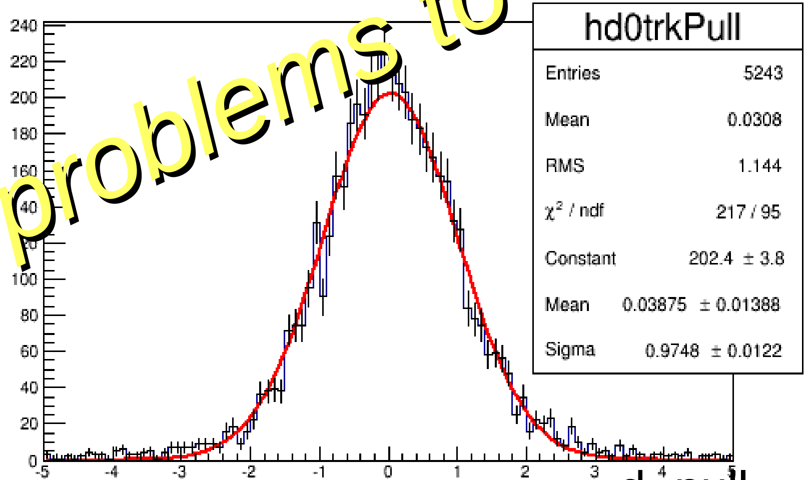
| hpttrkPull | |
|---|---|
| Entries | 5243 |
| Mean | 0.5046 |
| RMS | 1.121 |
| $\chi^2$ / ndf | 281.9 / 93 |
| Constant | 217.3 ± 4.3 |
| Mean | 0.4701 ± 0.0131 |
| Sigma | 0.8698 ± 0.0116 |

$P_t$ pull



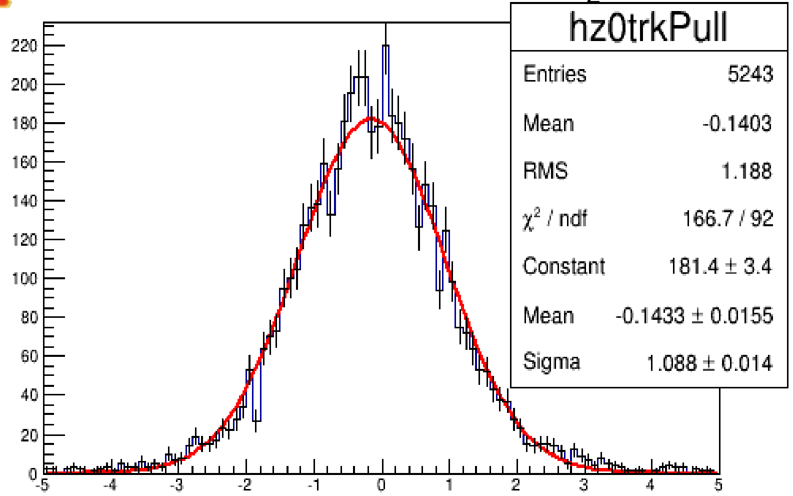
| hpztrkPull | |
|---|---|
| Entries | 5243 |
| Mean | 0.4559 |
| RMS | 1.051 |
| $\chi^2$ / ndf | 225 / 89 |
| Constant | 226.8 ± 4.4 |
| Mean | 0.4235 ± 0.0123 |
| Sigma | 0.8565 ± 0.0110 |

$P_z$ pull



| hd0trkPull | |
|---|---|
| Entries | 5243 |
| Mean | 0.0308 |
| RMS | 1.144 |
| $\chi^2$ / ndf | 217 / 95 |
| Constant | 202.4 ± 3.8 |
| Mean | 0.03875 ± 0.01388 |
| Sigma | 0.9748 ± 0.0122 |

$d_0$ pull



| hz0trkPull | |
|---|---|
| Entries | 5243 |
| Mean | -0.1403 |
| RMS | 1.188 |
| $\chi^2$ / ndf | 166.7 / 92 |
| Constant | 181.4 ± 3.4 |
| Mean | -0.1433 ± 0.0155 |
| Sigma | 1.088 ± 0.014 |

$z_0$ pull

Performance in PandaRoot trunk-rev28747: work in progress....

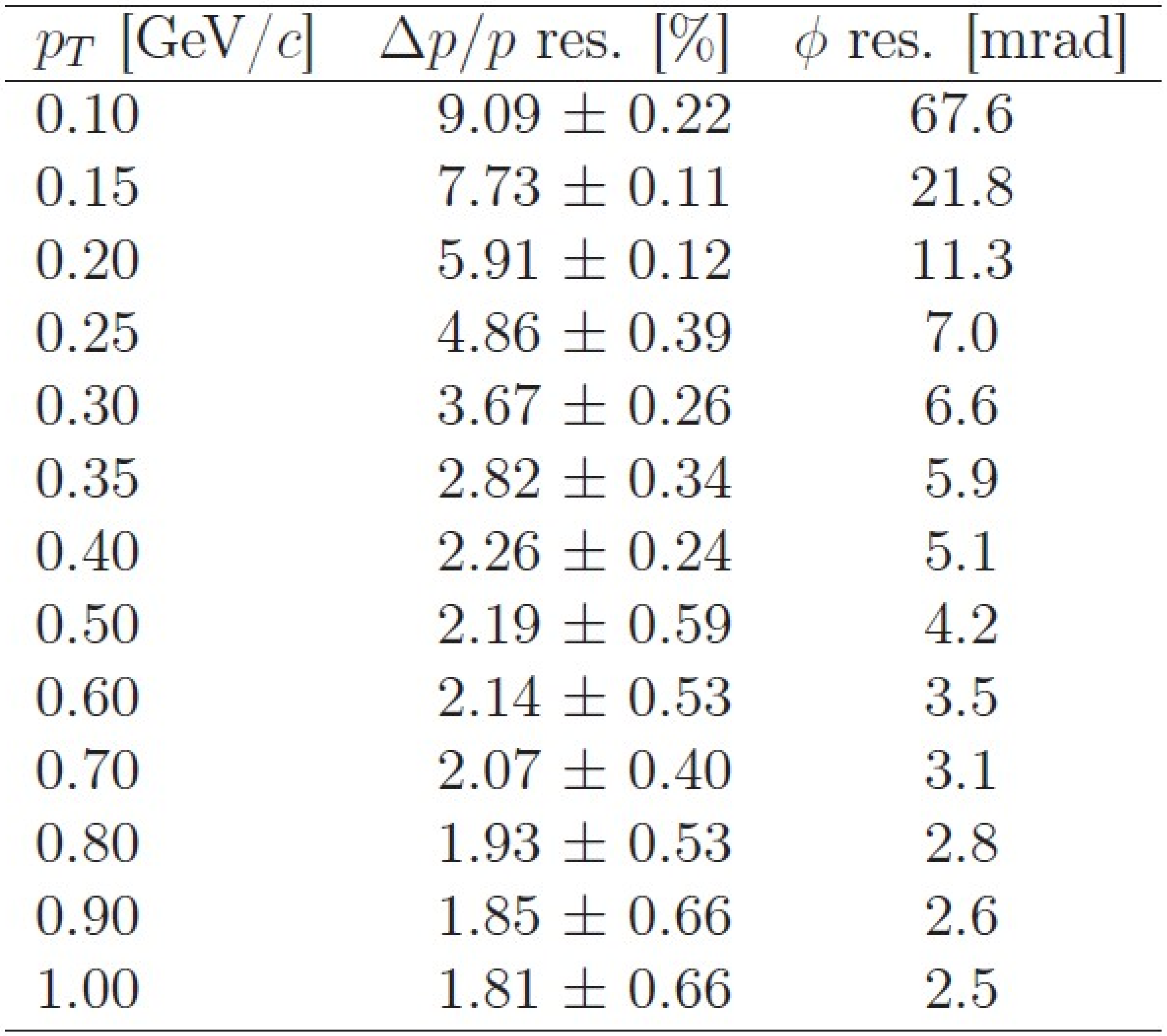
# Summary: resolution and pull fits

| Pion mom. (MeV/c) | P resolution (*) (%) | $\phi$ resolution (mrad) | $\theta$ resolution (mrad) | $z_0$ resolution ($\mu$m) | Efficiency in $\mu\pm3\sigma$ (%) |
|---|---|---|---|---|---|
| 1000 | 1.81±0.66 | 2.419±0.034 | 1.829±0.031 | 96.16±0.44 | 86.18±0.15 |
| 400 | 2.26±0.24 | 5.706±0.049 | 4.546±0.061 | 220.80±0.23 | 85.54±0.92 |
| 150 | 7.73±0.11 | 16.42±0.52 | 13.51±0.95 | 688.90±0.97 | 50.26±0.71 |

(*) p resolution ($\sigma$ gaussian fit) is normalized to the mean value extracted from fit: $\Delta p/p$

| Pion mom. (MeV/c) | $P_t$ pull | $P_z$ pull | $d_0$ pull | $z_0$ pull | Efficiency in $\mu\pm3\sigma$ (%) |
|---|---|---|---|---|---|
| 1000 | 1.146±0.005 | 0.96±0.04 | 0.9415±0.004 | 0.949±0.01 | 86.18±0.15 |
| 400 | 1.02±0.01 | 0.88±0.08 | 0.9127±0.008 | 1.026±0.009 | 85.54±0.92 |
| 150 | 0.87±0.01 | 0.86±0.01 | 0.975±0.012 | 1.088±0.014 | 50.26±0.71 |

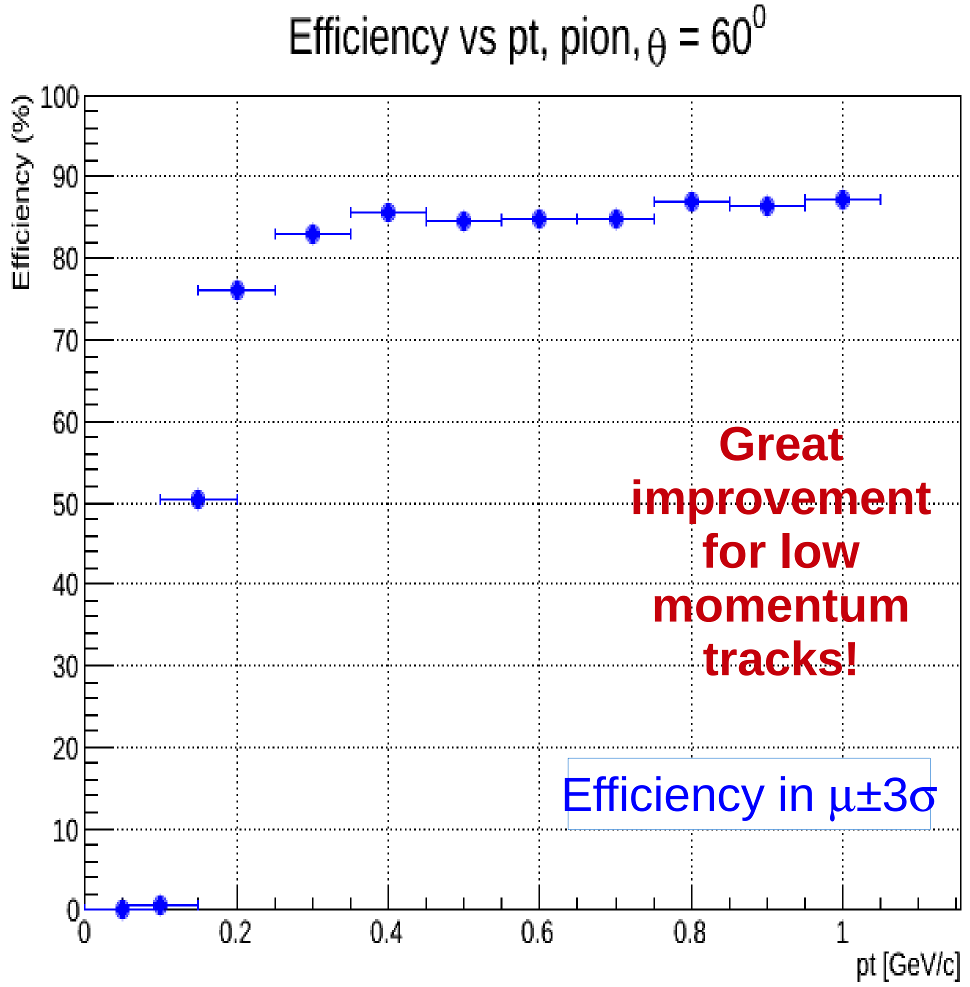
# Summary: momentum resolution



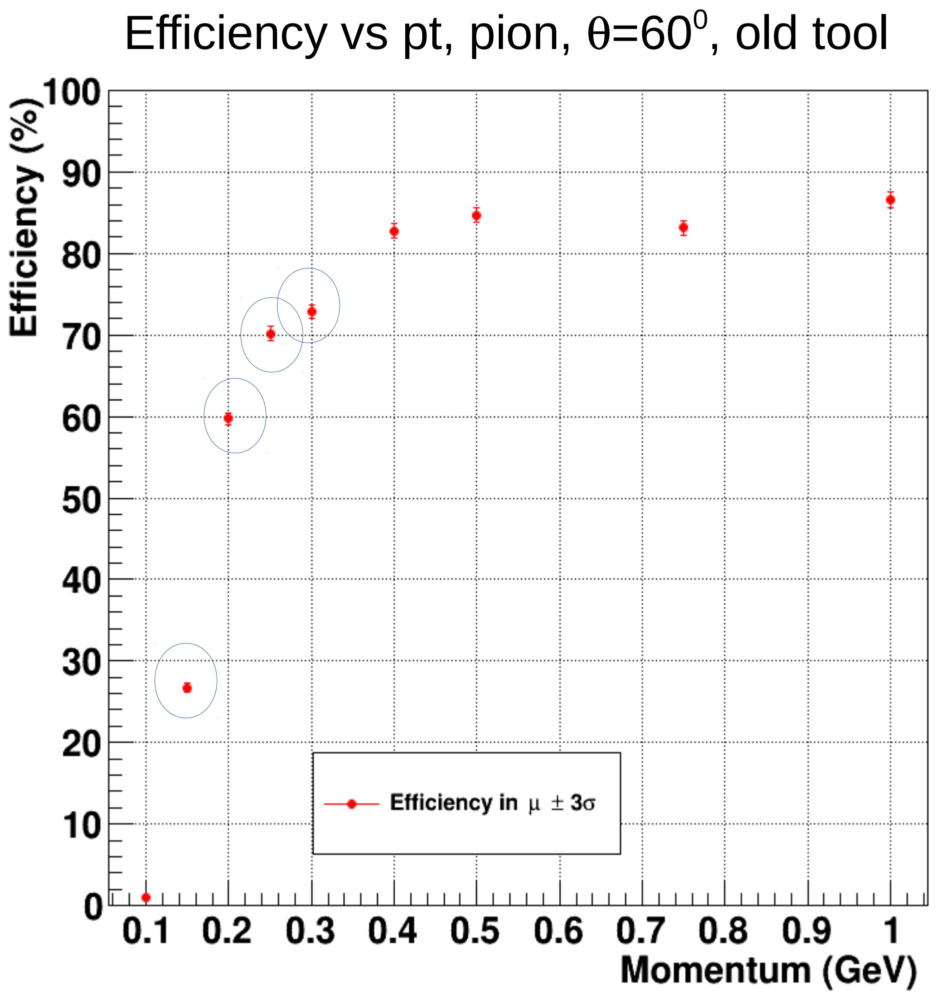
| $p_T$ [GeV/$c$] | $\Delta p/p$ res. [%] | $\phi$ res. [mrad] |
|---|---|---|
| 0.10 | $9.09 \pm 0.22$ | 67.6 |
| 0.15 | $7.73 \pm 0.11$ | 21.8 |
| 0.20 | $5.91 \pm 0.12$ | 11.3 |
| 0.25 | $4.86 \pm 0.39$ | 7.0 |
| 0.30 | $3.67 \pm 0.26$ | 6.6 |
| 0.35 | $2.82 \pm 0.34$ | 5.9 |
| 0.40 | $2.26 \pm 0.24$ | 5.1 |
| 0.50 | $2.19 \pm 0.59$ | 4.2 |
| 0.60 | $2.14 \pm 0.53$ | 3.5 |
| 0.70 | $2.07 \pm 0.40$ | 3.1 |
| 0.80 | $1.93 \pm 0.53$ | 2.8 |
| 0.90 | $1.85 \pm 0.66$ | 2.6 |
| 1.00 | $1.81 \pm 0.66$ | 2.5 |

Performance in PandaRoot trunk-rev28747: work in progress....

# Testing genfit2: Efficiency vs $P_t$



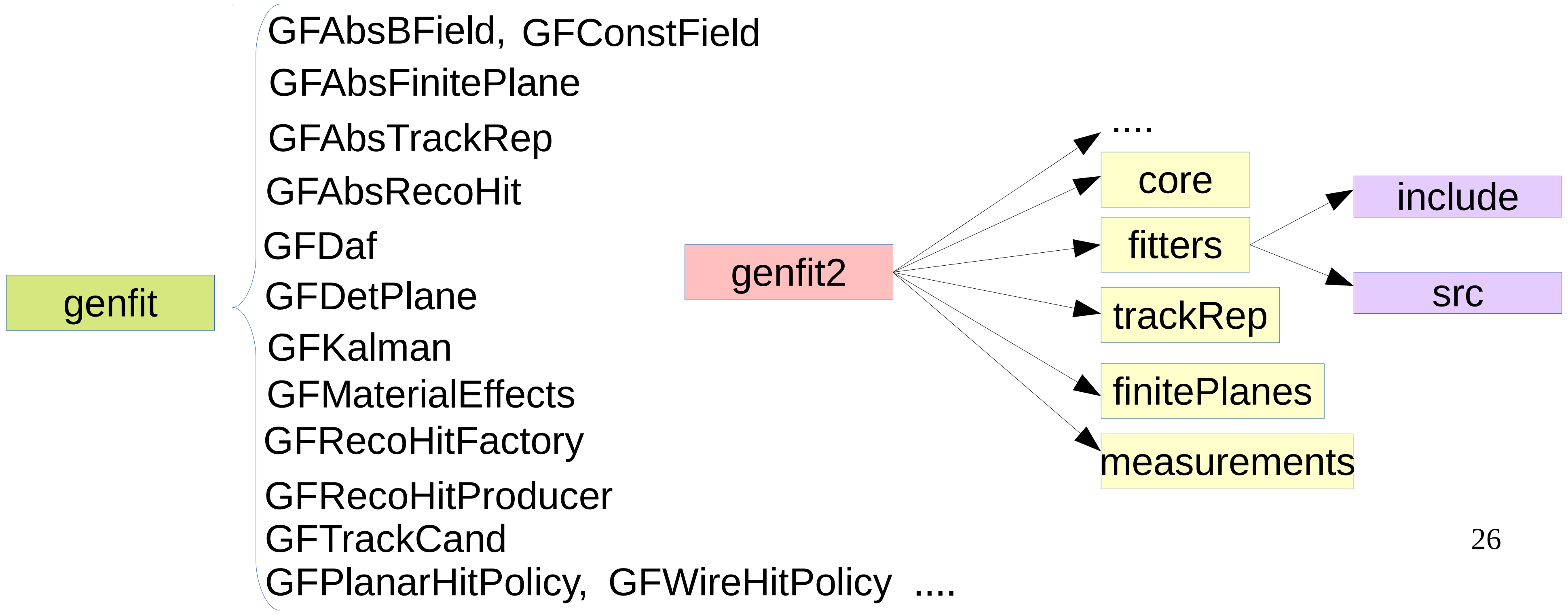
Great improvement for low momentum tracks!



Cut applied: p>50 MeV/c, because at least 1 hit is required in STT ⇨$R_{curv}$>8 cm⇨p>48 MeV/c

## Comparison: genfit vs genfit2

- genfit2 improves over genfit:
  - – genfit2 adds a well-defined, flexible data model that collects hits and fit information
  - – genfit2 allows configurable storage to disk
  - – genfit uses the same algorithm for planes used in the fit
- genfit2 uses of an adaptive step size Runge-Kutta algorithm to follow the track through an arbitrary magnetic field
- genfit2 makes a check on the fit convergence, while it was not done in genfit.
- Kalman with reference track is new in genfit2
- Vertex finder: RAVE is part of genfit2 now



## Why shall we use genfit2?

- General implementation of the Kalman fitter
- Track representation included
- Alignment studies: GBL interfaced
- Vertexing: RAVE interfaced
- Many parameters for fit convergence user-adjustable
- Independent on detector geometry
- Valid tool for every B field
- Suited to track low momentum particles: $\overline{\mathrm{P}}$ANDA and BELLE II: p>50 MeV/c

## How difficult is to interface genfit2 with another framework?

It depends...
My experience with PandaRoot:
~3 months to get the *GenfitTool* interface running inside /development/brunch/;
~3 months for debugging (PidCorrelator, memory leak, ...);
>3 months to perform generalized tests with all mass hypotheses and different $p_{beam}$

- Documentation: common paper with Belle II and $\overline{\mathrm{P}}$ANDA planned.

- Experiments using genfit2: Belle II, $\overline{\text{P}}$ANDA, GEM –TPC, FOPI, SHip, AFIS,...)

The family is growing....

Why?

- It is a fast solution
- It is maintained
- It is versatile
- C++ code

- About possible new missing features: it depends on the community requests...

Please, try and complain!

Thank you a lot for your kind attention