Software Developments for Hyperon Physics with PANDA@HADES

Jenny Regina

PANDA CM Computing Session

March 07, 2023

Outline and Purpose of this Presentation

- Show software work PANDA members are doing also outside of PandaRoot
- Identify further connections to PandaRoot and collaborations

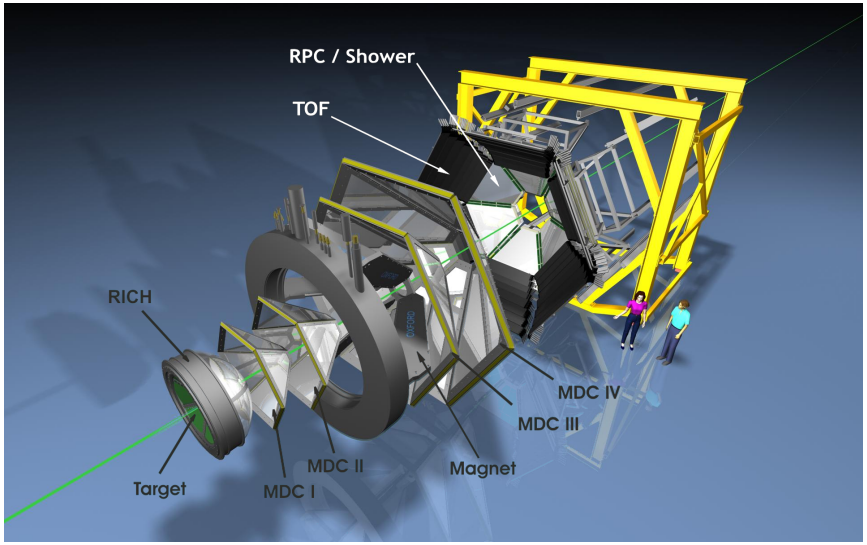
Outline

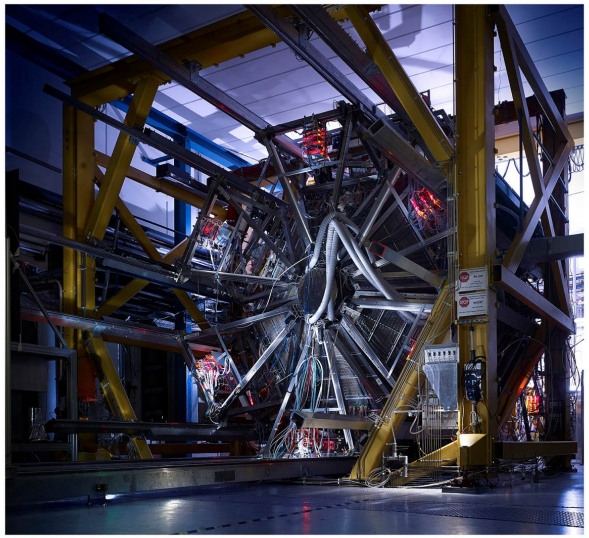
- Straw Tracking Stations
  - Tracking
  - Calibration
  - Alignment
- KinFit - Experiment independent fitting library

General HADES

High-Acceptance Di-Electron Spectrometer

- Operating at GSI at SIS18 since 2001
- Precise spectroscopy of e^+e^- pairs and charged hadrons
- pp and heavy ion (*e.g.* Ag-Ag, Au-Au) collisions
- Main purpose: Dense nuclear matter properties via in-medium hadron properties
- Hyperon physics a hot topic lately
- Acceptance of detector: $\sim$15-85°





Hyperons in the forward direction

- HADES missing low angular region where many hyperon decay products go
- Good idea to put a forward detector at low angles?

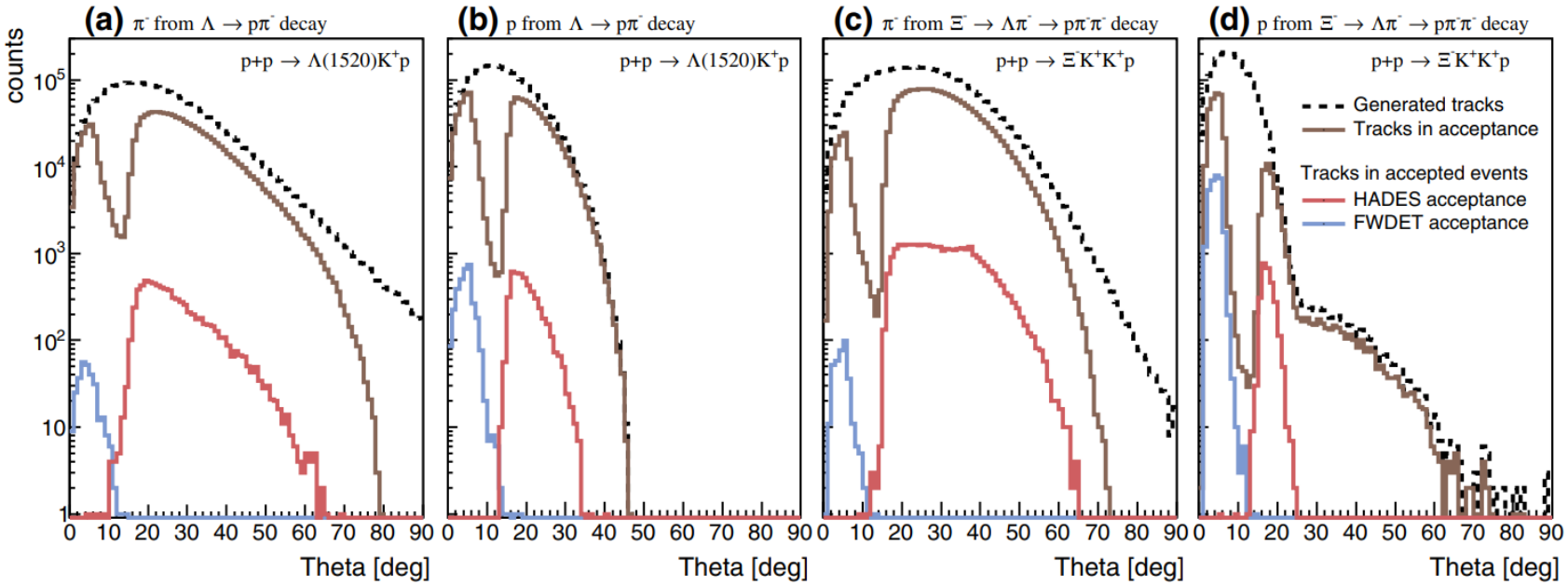
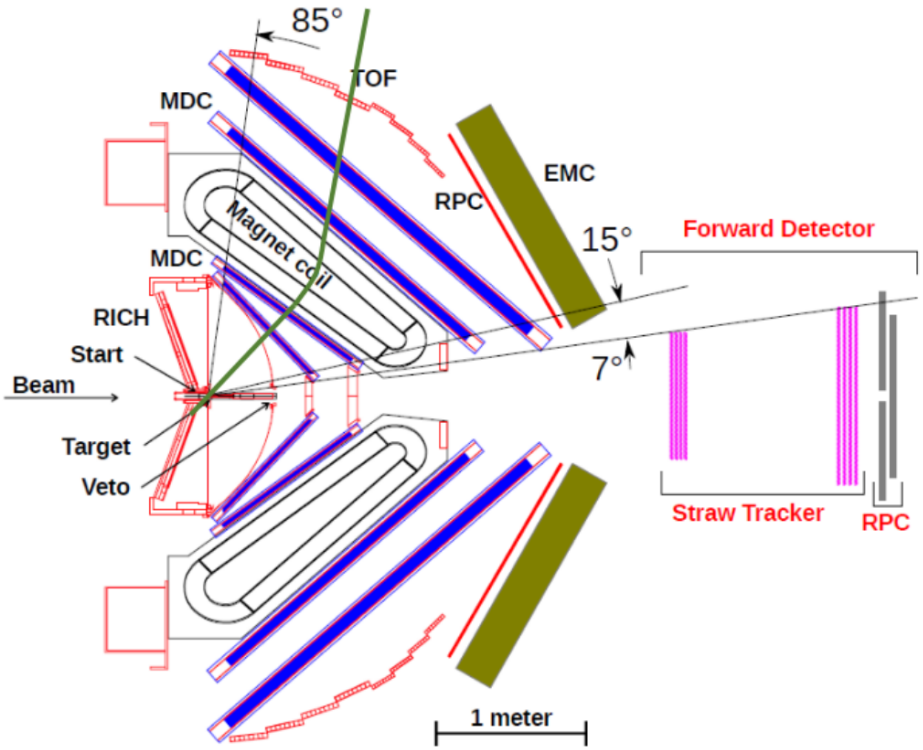


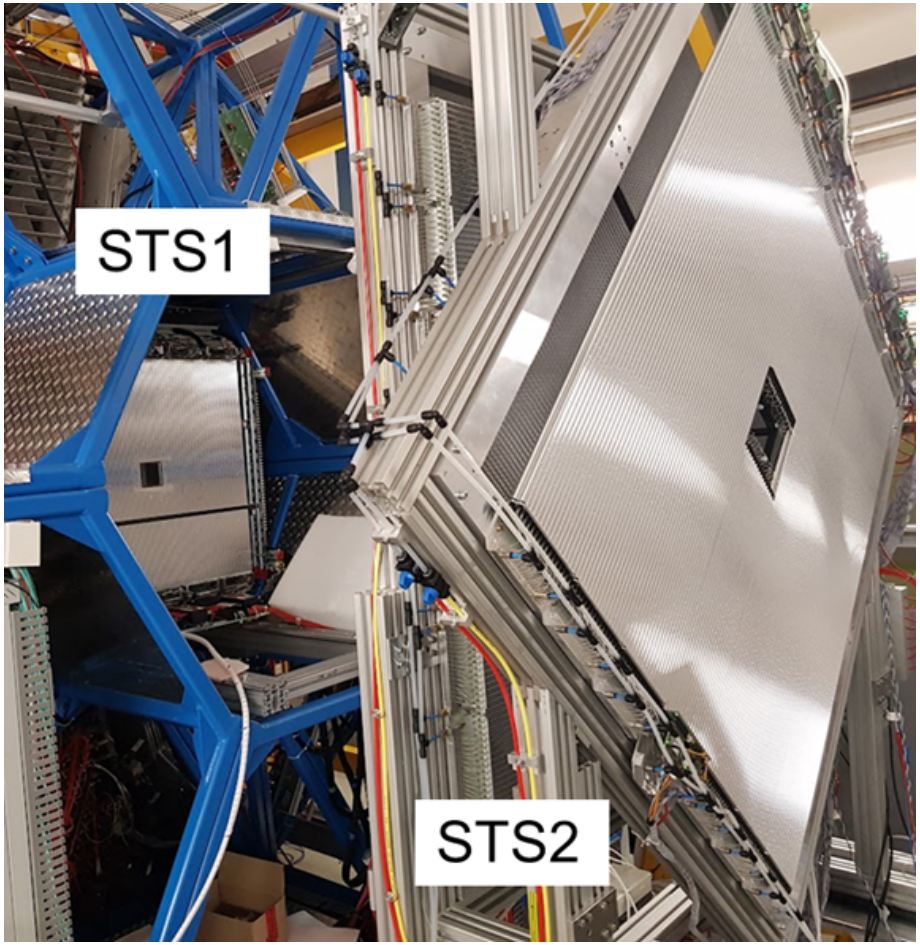
Figure from: Eur. Phys. J. A (2021) 57 :138

Forward Detector Upgrades

- Covers angles between ∼1-7°
- Straw Tracking Stations (STS)
  - Based on PANDA design
  - Geometrical track Reconstruction
  - 8 double layers of straws
- Forward Resistive Plate Chamber (fRPC) timing detector
  - Momentum estimation
  - Magnetic field free region, mass hypothesis of protons currently assumed
- Used in feb21 proton test beam data taking and feb22 proton beam physics run



STS



Plane orientations:

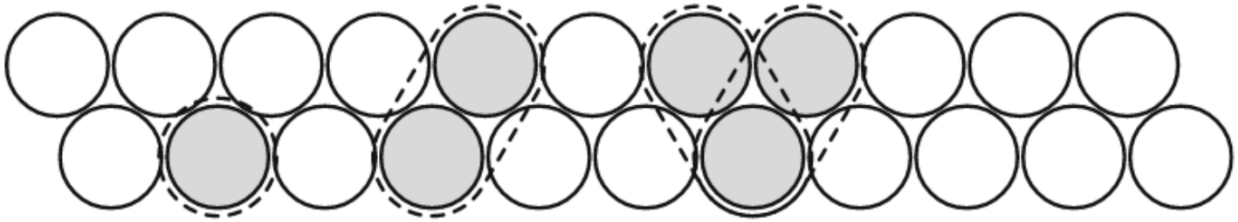
STS1: $0°$, $90°$, $90°$, $0°$

STS2: $0°$, $90°$, $+45°$, $-45°$

Allows for 3D reconstruction

Tracking, Clustering in Double Layers

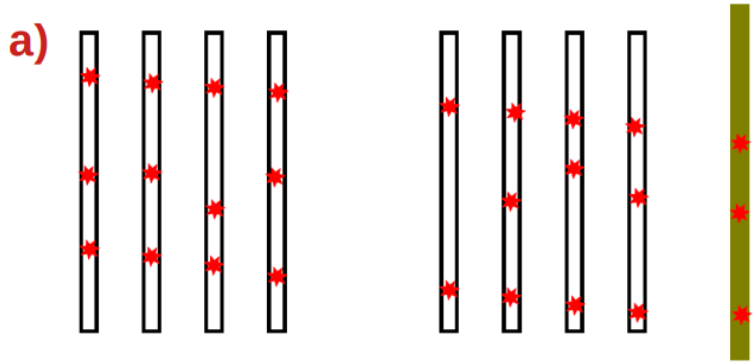
Current tracking implemented by Rafal Lalik, based on CBM-MUCH and COSY-TOF code



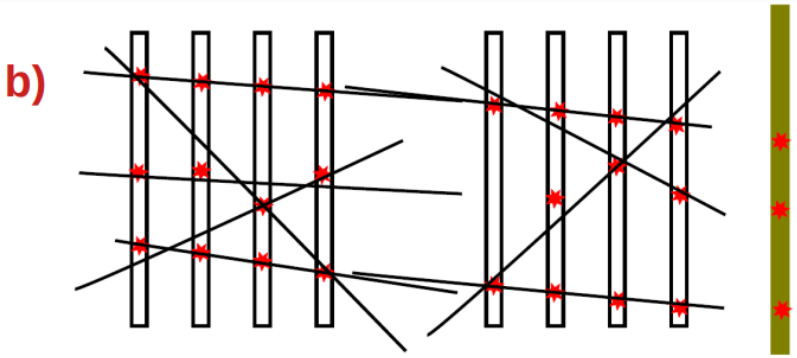
- Pre-clustering in double layers
- Hits in configurations that are unlikely to have been created by different particles
- Reduce combinatorics and runtime

Figure from: Eur. Phys. J. A (2021) 57 :138

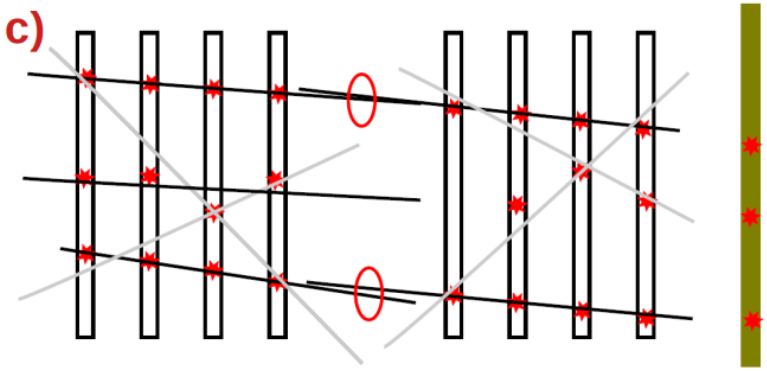
Tracking



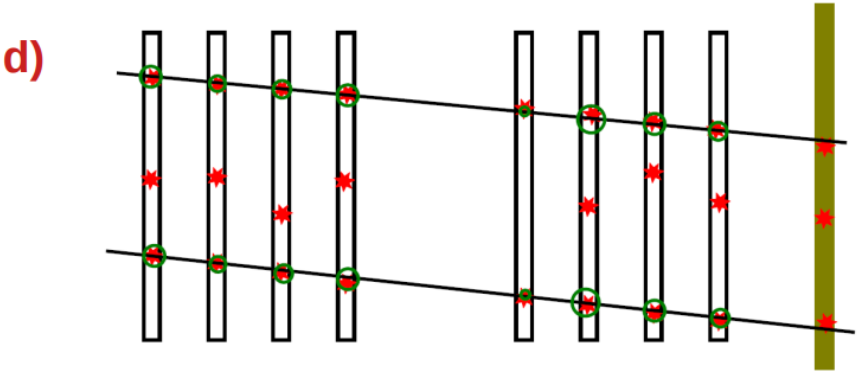
Take all hits in the STS + fRPC



Fit a line to hits in STS1
Fit a line to hits in STS2



Combine best matching tracklets from STS1 and STS2

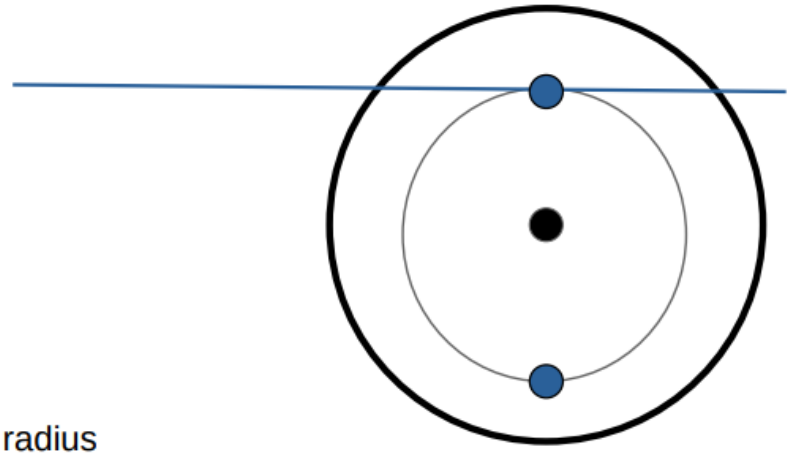


Closest STS and fRPC hits are assigned to track

Picture credit R. Lalik.

Tracking High Resolution

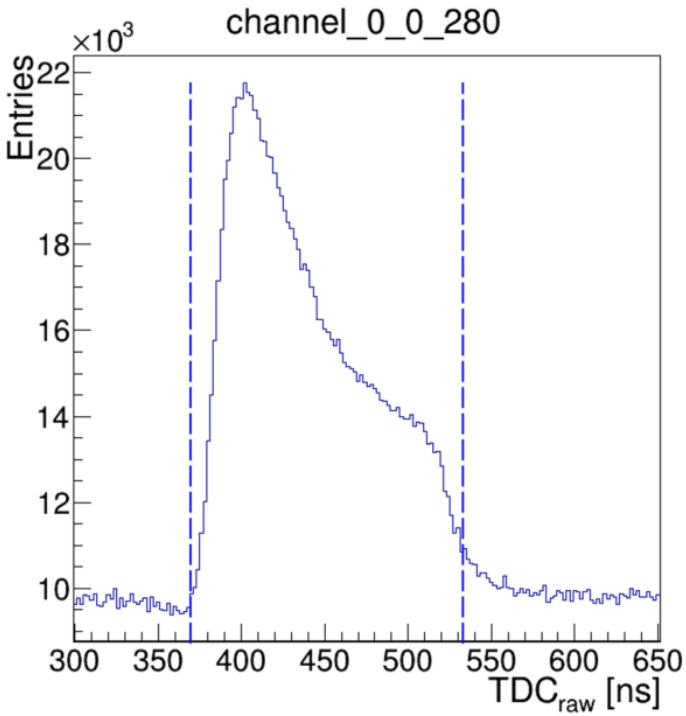
- Every combination of wire center ± drift radius tested due to ambiguity
- Gives up to 2^{16} combinations

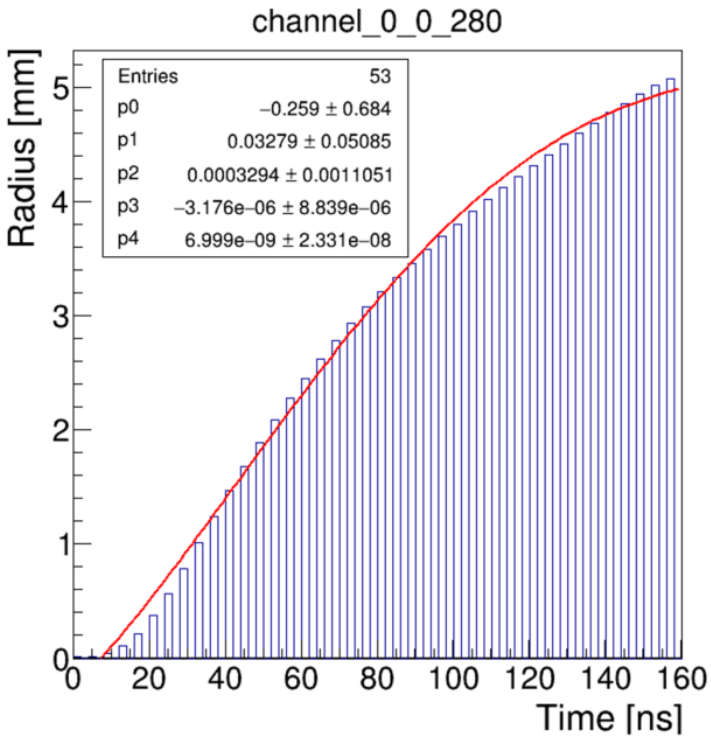


$$\chi^2 = \frac{1}{N}\sum_{i=0}^{N}\left(\frac{\Delta r_i(P)}{\sigma_{r_i}}\right)^2$$

- Δr_i (P) = distance between the i wire and the track – drift radius
- σr_i = uncertainty

Calibration

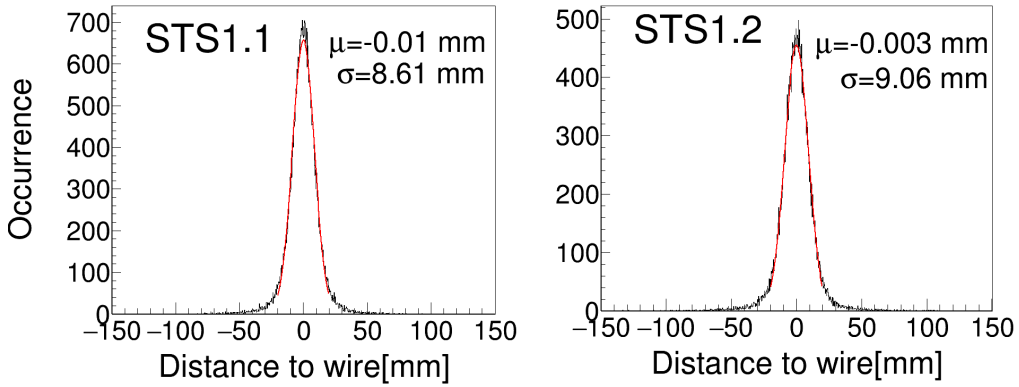




- Work by Gabriela Pérez Andrade
- Minimum time and t_{max} for integration found from signal shape
- Fit to drift-time spectrum used as input to tracking

Alignment

- pp-elastic scattering
- Measure one proton in main HADES where alignment is fairly well under control
- Use elastic scattering conditions to find the other proton in Forward Detector
- Re-calculate the angles of the forward going proton from the angles of the Main HADES proton
- Plot residuals between track and wire
- Shift the planes according to mean value



Left: residuals of original tracks after alignment

Possible outlook to test Millipede II [*] for alignment

[*] $https://www.desy.de/\sim kleinwrt/MP2/doc/html/index.html$

KinFit, general

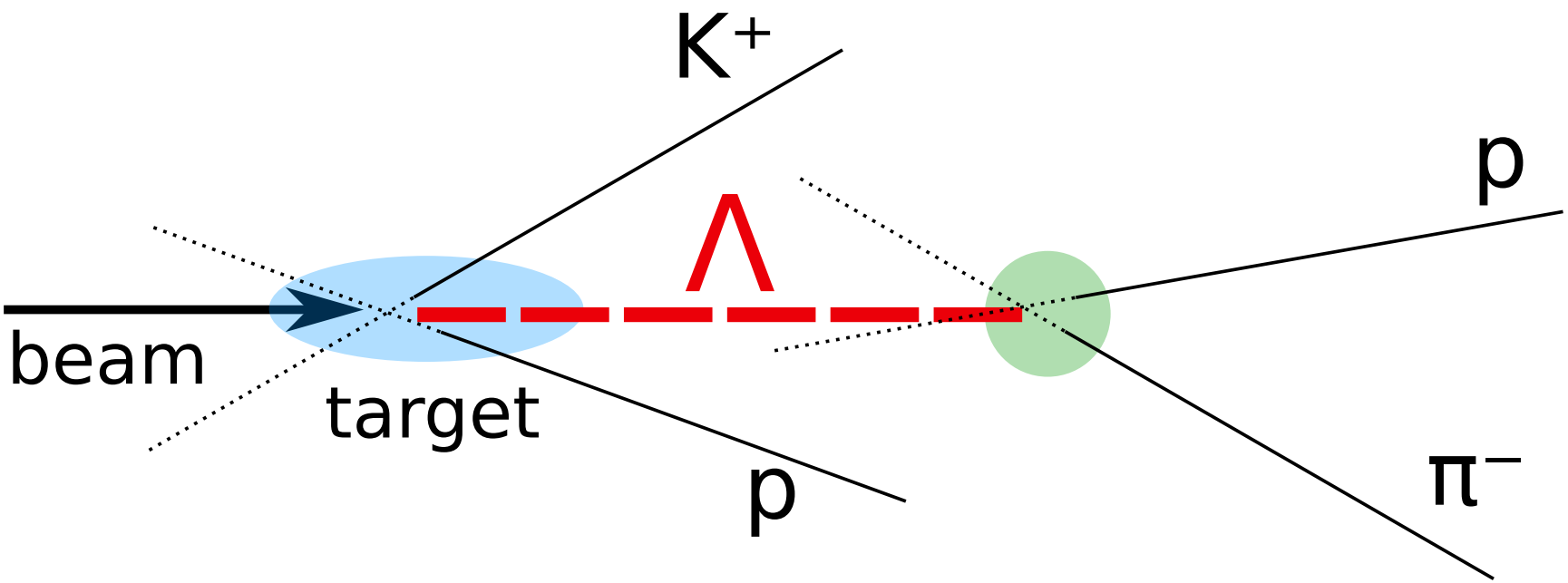
- KinFit: Experiment independent kinematic fitting tool
- Work by Jana Rieger, Waleed Esmail and myself
- Built using cmake, written in C/C++
- Developed within HADES and PANDA@HADES
- Intrinsically using HADES track parametrization (although user can construct their particle objects and pass to the fitter)
- Will be distributed via github
- Kinematic Track fitting: Track parameters are adjusted to fulfill the constraints using Lagrange multiplier method [*]

[*] A. G. Frodesen, O. Skjeggestad, Probability and Statistics in Particle Physics, Universitetsforlaget, Bergen, Norway, 1979

KinFit, constraints

- **Mass Constraint**
- **Geometric vertex Constraint**
  - Optimize track parameters with the constraint that the minimum track distance should be zero
- **3C Constraint**
  - Direction of the Neutral decay particle and that of the final state particle tracks
  - 4-momentum of all the final state particles (*e.g.* p and π^-) at the decay vertex is conserved
- **4C Constraint**
  - 4-momenta of all-final state particles conserved w.r.t. beam-target system
- **Missing particle constraint**

Example Channel



- 3C constraint applied to proton + pion from decay + Λ
- 4C constraint applied to both protons + kaon + pion + beam-target system

Iterative Procedure

$$\chi^2 = \sum_{i=1} \frac{y_i - \eta_i}{\sigma_i^2} \quad \approx minimum$$

- y - vector with measured track parameters
- η - vector with improved track parameters
- σ - uncertainties

$$\mathcal{L} = (y-\eta)^T V^{-1} (y-\eta) + 2\lambda^T f(\eta, \xi) \approx minimum$$

- f - constraint equation(s)
- λ - Lagrange multipliers
- ξ - vector with unmeasured parameters
- V - covariance matrix

$$f_k = f(\eta_1, \eta_2, ..., \eta_N, \xi_1, \xi_2, ..., \xi_J) = 0, \qquad k = 1, 2, ..., K$$

Iterative Procedure

$$r = f^{\nu} + F_{\eta}^{\nu}(y - \eta^{\nu})$$
$$S = F_{\eta}^{\nu} V (F_{\eta}^{\nu})^{T}$$

F_{η} is a $K \times N$ Jacobian matrix ($F_{\eta} = \frac{\partial f}{\partial \eta}$)
ν is the iteration number

$$\xi^{\nu+1} = \xi^{\nu} - (F_{\xi}^{T} S^{-1} F_{\xi})^{-1} F_{\xi}^{T} S^{-1} r$$

$$\lambda^{\nu+1} = S^{-1}\Big(r + F_{\xi}(\xi^{\nu+1} - \xi^{\nu})\Big)$$

$$\eta^{\nu+1} = y - V F_{\eta}^{T} \lambda^{\nu+1}$$

$$V^{Final} = V - V\left[F_{\eta}^{T} S^{-1} F_{\eta} - ((F_{\eta}^{T} S^{-1} F_{\xi})(F_{\xi}^{T} S^{-1} F_{\xi})^{-1}(F_{\eta}^{T} S^{-1} F_{\xi})^{T})\right] V$$

Classes

- KFitDecayBuilder.h
- KFitNeutralCandFinder.h
- KFitParticle.h
- KFitRootAnalyzer.h
- KFitVertexFinder.h
- KinFitter.h

HADES track paremtrization

HADES track parameters:

- Momentum $1/p$ [a.u.]
- Polar angle θ [radians]
- Azimuthal angle ϕ [radians]
- R [mm] - distance between beam axis and point of closest approach of track to beam axis
- Z [mm] - z coordinate of point of closest approach of track to beam axis

Any point, $\vec{p}$, on straight line can be found from $\vec{p} = \vec{b} + t\vec{d}$ where t is a real number

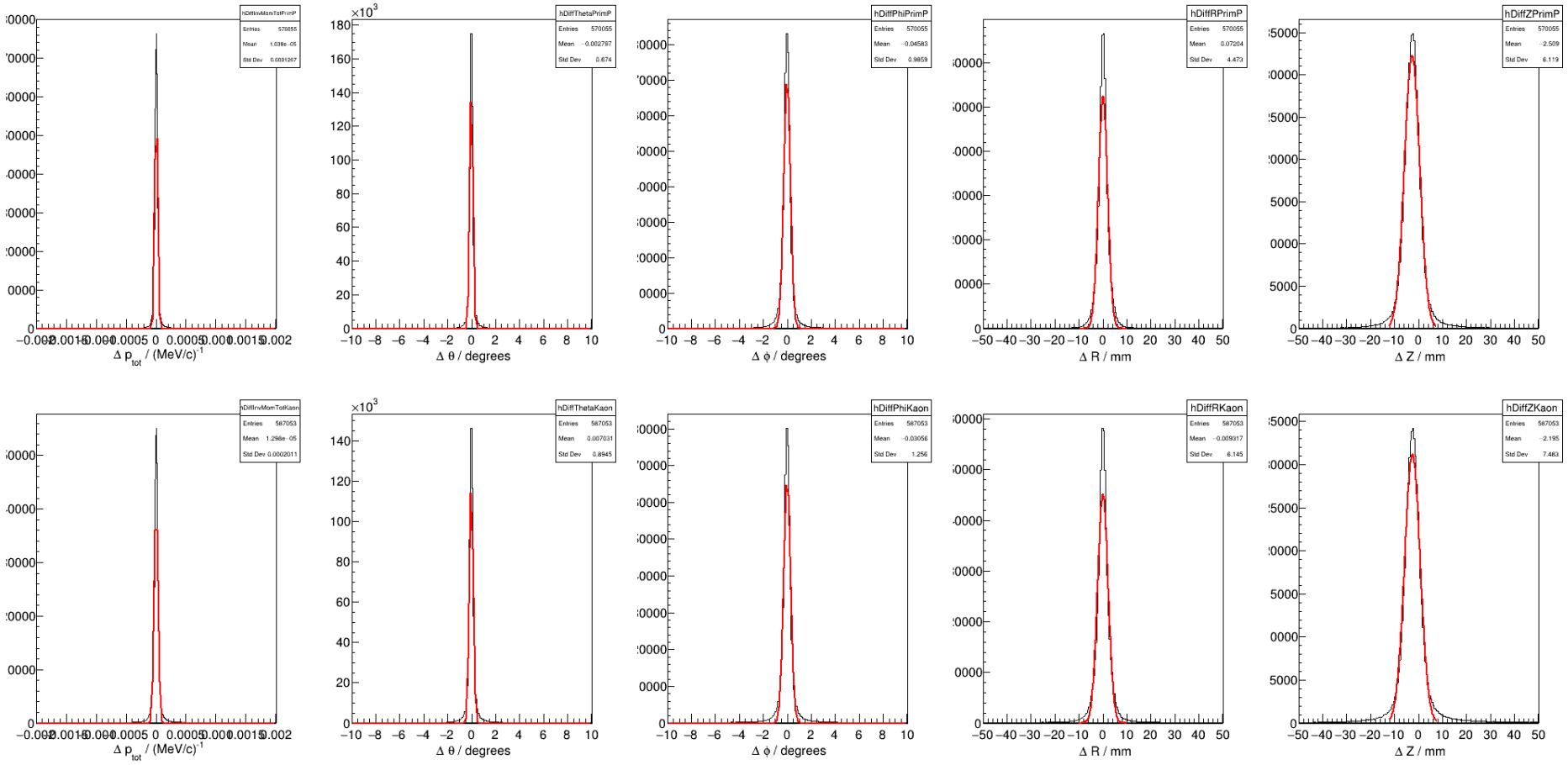
Base:
$\vec{b} = (b_x, b_y, b_z)$
and direction:
$\vec{d} = (d_x, d_y, d_z)$
vectors uniquely define a straight line in 3D

Can be constructed from HADES track parameters:

$$\begin{cases} b_x = R \cdot \cos(\phi + \pi/2) \\ b_y = R \cdot \sin(\phi + \pi/2) \\ b_z = Z \end{cases}$$

$$\begin{cases} d_x = \sin(\theta) \cdot \cos(\phi) \\ d_y = \sin(\theta) \cdot \sin(\phi) \\ d_z = \cos(\theta) \end{cases}$$

Error Estimates



Missing particle, Constraint equations

$$f = \begin{cases} \sum_{i=1}^{N} p_i \sin\theta_i \cos\phi_i + p_{\mathrm{miss},x} - p_{\mathrm{ini},x} = 0 \quad (p_x), \\ \sum_{i=1}^{N} p_i \sin\theta_i \sin\phi_i + p_{\mathrm{miss},y} - p_{\mathrm{ini},y} = 0 \quad (p_y), \\ \sum_{i=1}^{N} p_i \cos\theta_i + p_{\mathrm{miss},z} - p_{\mathrm{ini},z} = 0 \quad (p_z) \\ \sum_{i=1}^{N} \sqrt{p_i^2 + m_i^2} + \sqrt{p_{\mathrm{miss}}^2 + m_{\mathrm{miss}}^2} - E_{\mathrm{ini}} = 0 \quad (E) \end{cases}$$

The sums are over all measured particles and ini is initial system. $miss$ corresponds to missing particle

Missing Particle Fit, How to use the code

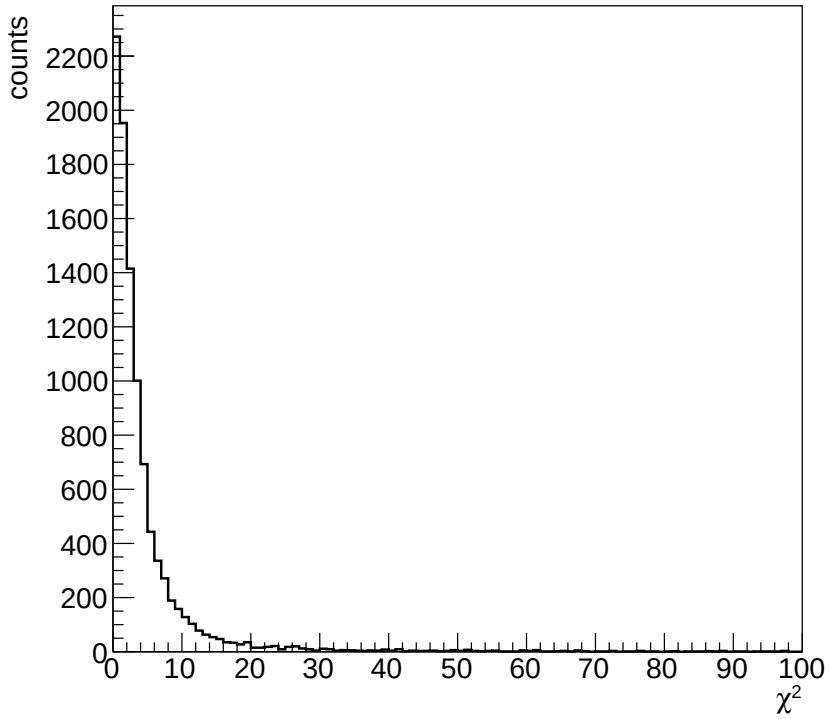
```
#include "TLorentzVector.h"
#include "KinFitter.h"

Double_t mass;
TLorentzVector ppSystem(p1,p2,p3,E);
KinFitter fitter(cand_vector);
void addMomConstraint(ppSystem, mass); // Choose
   momentum constraint for missing particle
fitter.fit();
TLorentzVector getMissingDaughter(); // Retrieve the
    missing daughter after the fit
```

χ^2 and Probability

- 1-to-1 correspondence
- Shape of χ^2 distribution determined by number of degrees of freedom
- Probability distribution should be uniformly distributed between 0 and 1
- Overestimation of errors $\rightarrow$ p-distribution pulled to higher values
- Underestimation of errors $\rightarrow$ p-distribution pulled to lower values

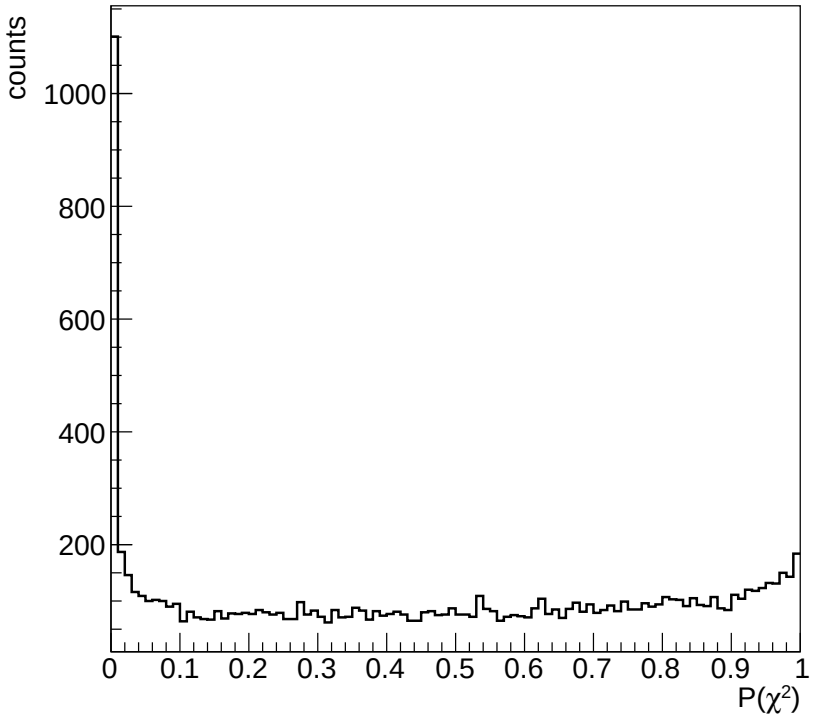
- Example from toy MC
- $pp \rightarrow pK^+\Lambda$, $\Lambda \rightarrow p\pi^-$
- 3C fit



χ^2 and Probability

- 1-to-1 correspondence
- Shape of χ^2 distribution determined by number of degrees of freedom
- Probability distribution should be uniformly distributed between 0 and 1
- Overestimation of errors $\rightarrow$ p-distribution pulled to higher values
- Underestimation of errors $\rightarrow$ p-distribution pulled to lower values

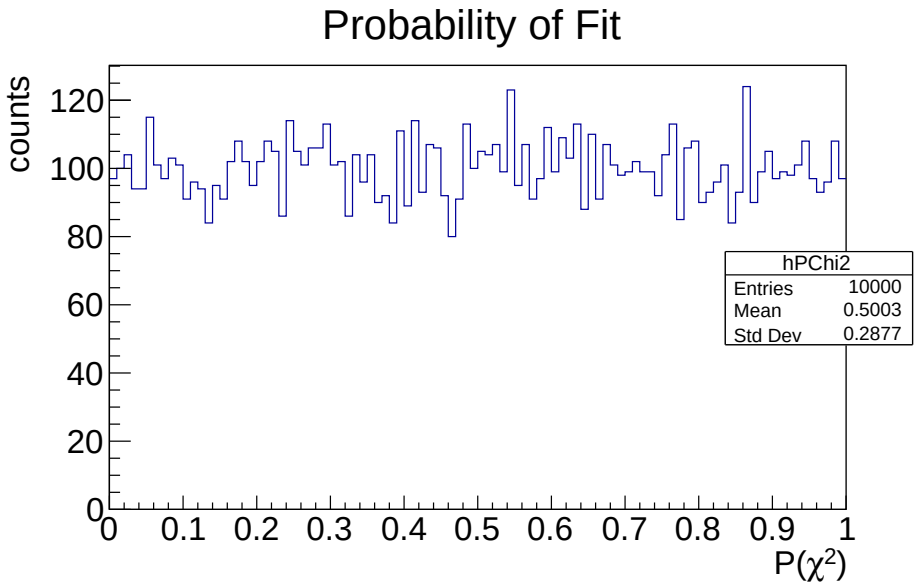
- Example from toy MC
- $pp \rightarrow pK^+\Lambda$, $\Lambda \rightarrow p\pi^-$
- 3C fit



χ^2 and Probability

- 1-to-1 correspondence
- Shape of χ^2 distribution determined by number of degrees of freedom
- Probability distribution should be uniformly distributed between 0 and 1
- Overestimation of errors $\rightarrow$ p-distribution pulled to higher values
- Underestimation of errors $\rightarrow$ p-distribution pulled to lower values

- Example from toy MC
- $pp \rightarrow pK^+\Lambda$, $\Lambda \rightarrow p\pi^-$
- Simple fit



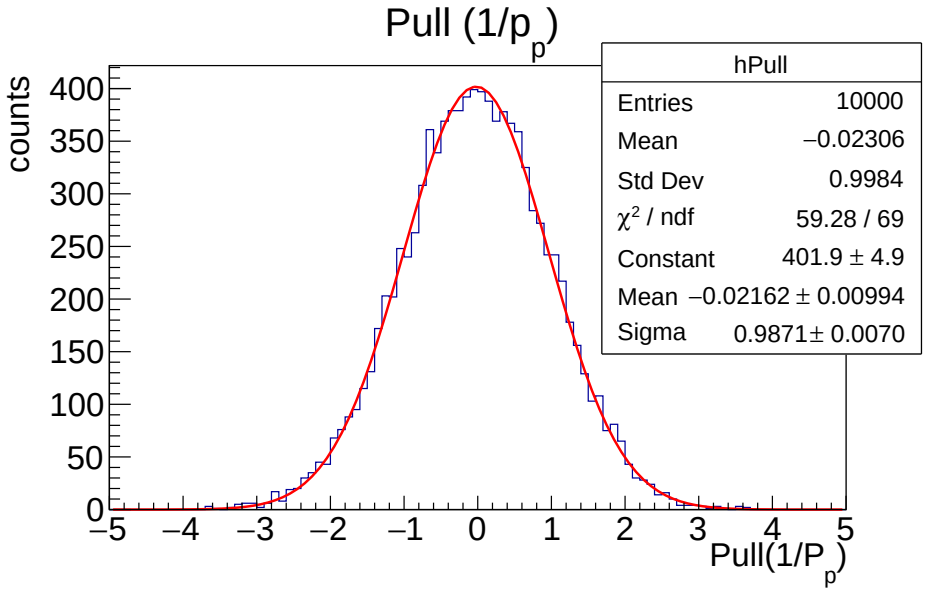
Pull distributions

Ideally N(0,1) distributed

$$pull = \frac{x_{fit} - x_{meas}}{\sqrt{\sigma^2_{fit} - \sigma^2_{meas}}} \quad (1)$$

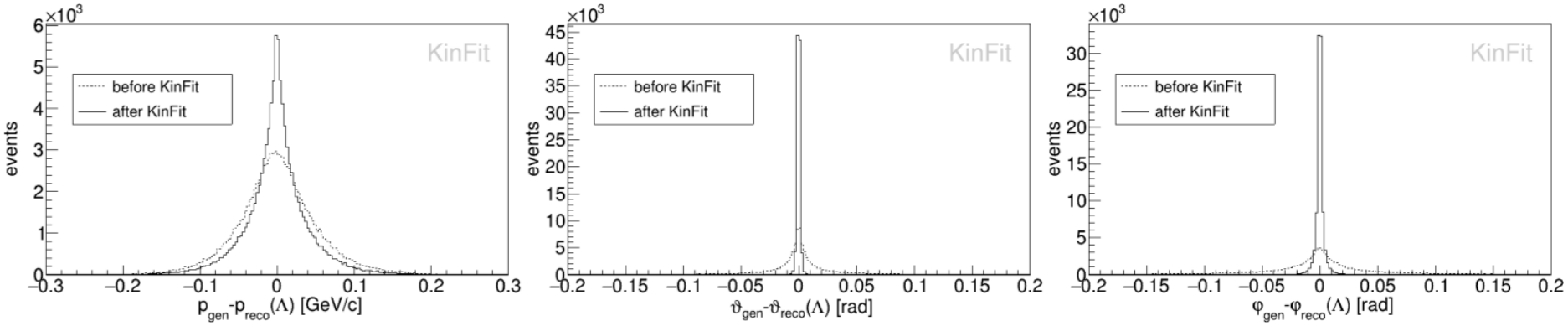
- Overestimation of errors → narrower peak
- Underestimation of errors → broader peak

- Example from toy MC
- $pp \to pK^+\Lambda$, $\Lambda \to p\pi^-$
- 3C fit



Resolutions

- Example from toy MC
- $pp \to pK^+\Lambda$, $\Lambda \to p\pi^-$
- 3C fit
- Resolution greatly improved after fitting



Runtime of code

- stl chrono
- i7-1185G7 processor at 3.00GHz
- Average processing time of 1000 fits
- SImilar performance for all constraints

Stage	Fitting	Initialization	Vertex Finding	Neutral Candidate Finding
Time / event [μs]	10	1	1	1

Table: Runtime of different parts of KinFit.

Possible PandaRoot Connections

- Incorporate functionality from the DecayTreeFitter for sequential decays into KinFit/HYDRA (HADES Software) or make the DecayTreeFitter independent from PandaRoot
- Exporting the Rho package to HYDRA?
- For tracking in the STS, test ACTS when possible to use for gaseous detectors
- High level QA for tracking used at HADES, interesting to also try PANDAs advanced low-level trackingQA?
- Calibration (to some extent also alignment) procedure for STS applicable for PANDA
- Test other PandaRoot (tracking) algorithms with STS?

Thank you!