GNN for tracking at BESIII and STCF

Xiaoqian Jia[1] , Xiaoshuai Qin[1] , Teng Li[1] , Xingtao Huang[1] , Xueyao Zhang[1] ,
Yao Zhang[2] and Ye Yuan[2]

1. Shandong University, Qingdao

2. Institute of High Energy Physics, Beijing

Workshop of Tracking in Particle Physics Experiments
May 18, 2024

Outline

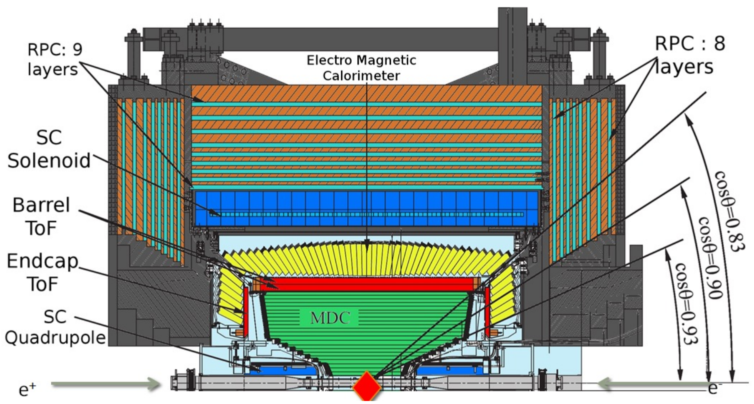
01 MDC at BESIII and STCF

Beijing electron-positron collider (BEPCII)

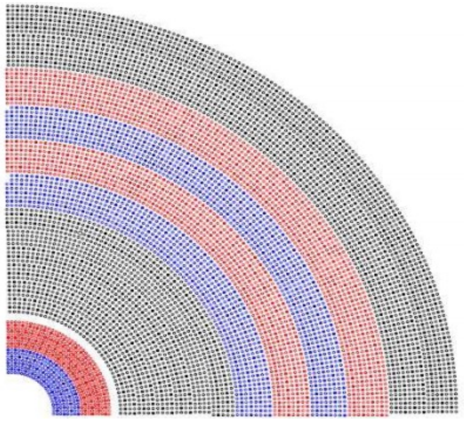
- Peak luminosity : 10^{33} cm^{-2} s^{-1}
- CMS: 2.0 - 4.95 GeV, τ -charm region
- World's largest J/ψ dataset : 10 billion

◆ Main Drift Chamber (MDC) at BESIII

- 43 sense wire layers
- 5 axial wire super-layers,6 stereo wire super-layers
- dE/dx resolution : 6%
- Momentum resolution : 0.5%@1GeV/c



BESIII detector

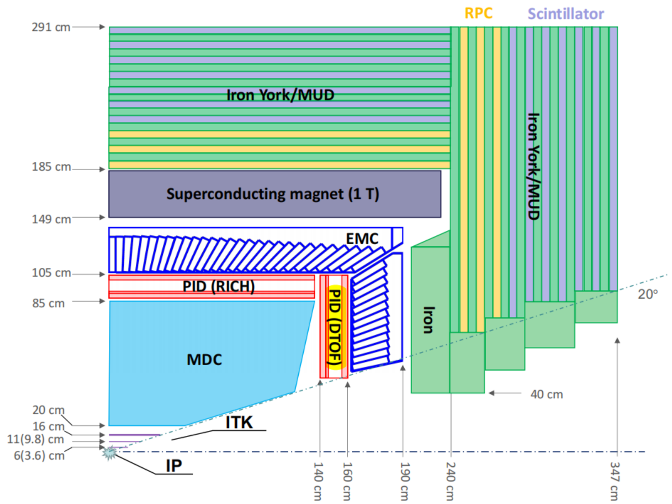

BESIII MDC

Super Tau-Charm Facility (STCF)

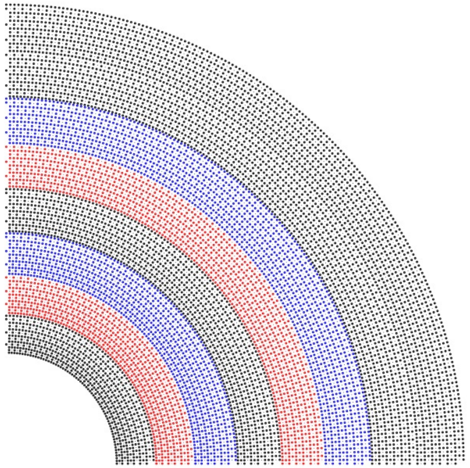
- High Luminosity: > 0.5×10^{35} cm^{-2} s^{-1}@4GeV
- CMS: 2.0 - 7 GeV

◆ Main Drift Chamber (MDC) at STCF

- 48 sense wire layers
- 4 axial wire super-layers,4 stereo wire super-layers
- dE/dx resolution : ~6%
- Momentum resolution : 0.5%@1GeV/c



STCF detector


STCF MDC

01 Traditional tracking of drift chamber

MDC hits produced by charged particles

↓

Track finding

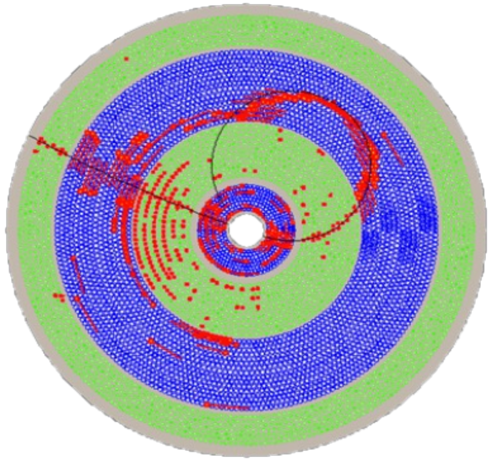
↓

Track fitting

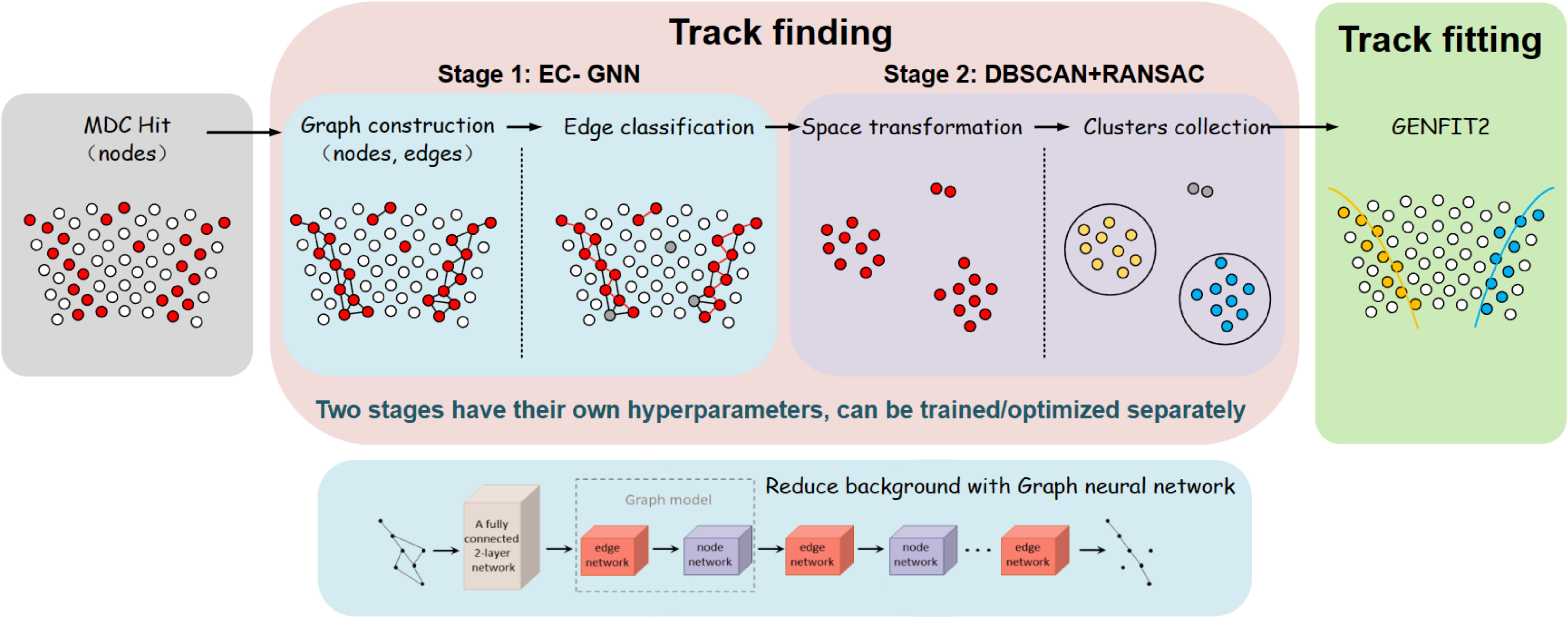
↓

Vertex and physics object reconstruction

- Build candidate tracks and perform hits assignment
  - Global approach : Hough Transform (HOUGH)
  - Local approach : Template Matching (PAT) Track Segment Finding (TSF)
    Combinatorial Kalman Filter (CKF)
- Estimate the track parameters
  - Global fit : Least Square Method, Runge-Kutta Method
  - Recursive fit : Kalman filter

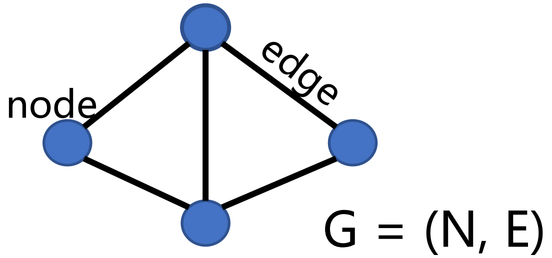


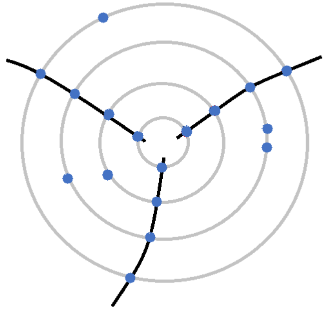
02 Methodology: GNN based tracking pipeline

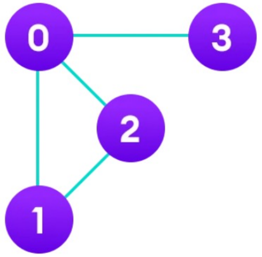


02 Graph and Graph Neural Network

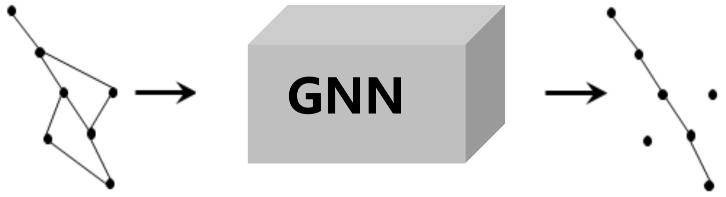
- A type of neural network that are specifically designed to operate on graph-structured data
- Graph: nodes, edges
- Graph → Track
  - Nodes → Hits
  - edges → track segments
- The storage structure of graphs
  - Adjacency matrix √
  - Adjacency table
  - Orthogonal list
  - Adjacency multiple table
  - Edge set array
  - ……



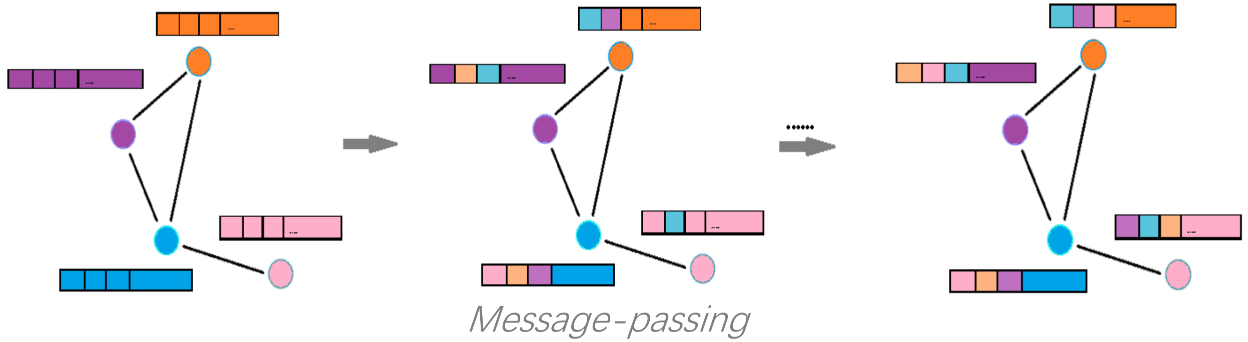




	0	1	2	3
0	0	1	1	1
1	1	0	1	0
2	1	1	0	0
3	1	0	0	0



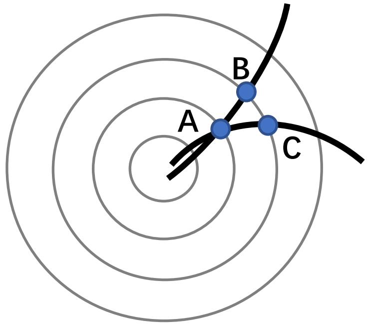
- GNN key idea: propagate information across the graph using a set of learnable functions that operate on node and edge features
- Graph Neural Network edge classifier
  - High classification score
    → *the edge belongs to a true particle track*
  - Low classification score
    → *it is a spurious or noise edge*



02 Graph construction at BESIII

To reduce the number of fake edges during graph construction

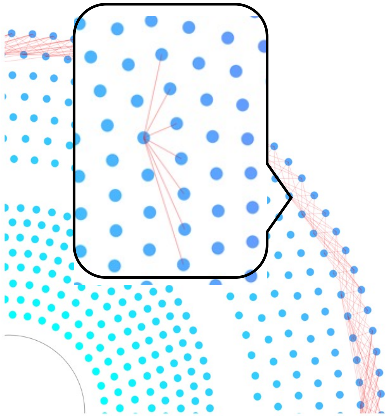
Pattern Map based on MC simulation at BESIII



- Definition of valid neighbors
  - Hits on the same layer

    Two adjacent sense wires on the left and right
  - Hits on the next layer

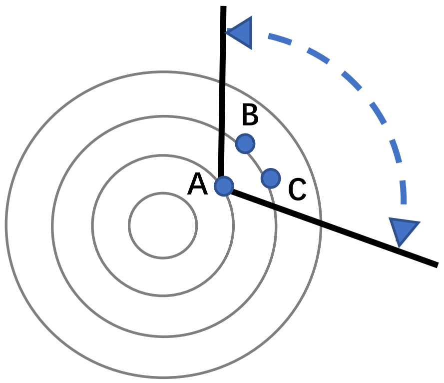
    The collection of sense wires that could potentially represent two successive hits on a track
- MC sample used to build pattern map
  - Two million single tracks produced with BESIII offline software (BOSS)
  - 5 types of charged particles ($e^{\pm}$, $K^{\pm}$, $\mu^{\pm}$, $p^{\pm}$, $\pi^{\pm}$)
  - 0.05 GeV/c < P < 3 GeV/c
- Edge assignment based on Pattern Map
  - Hit with its neighbors on the same layer and next layer
  - Hit with its neighbors' neighbors on one layer apart
- To reduce the size of the graphs, the Pattern Map is further reduced based on a probability cut
- Graph representation
  - Node features (raw time, position coordinates r, ϕ of the sense wires), adjacency matrices, edge labels



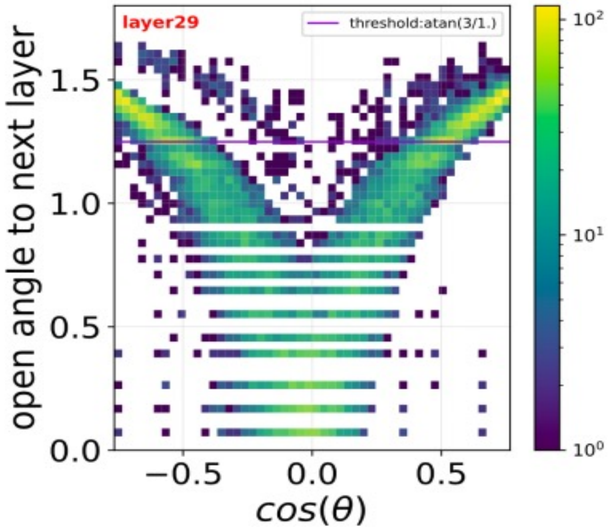
A wire on layer13 and tits neighbors on layer14

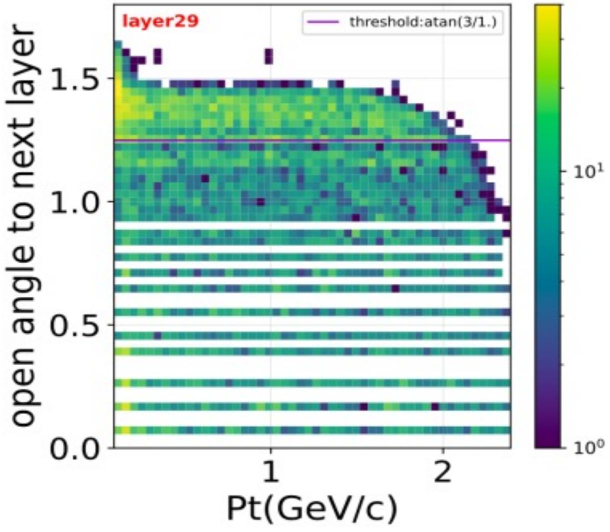
02 Graph construction at STCF

Geometric cut at STCF



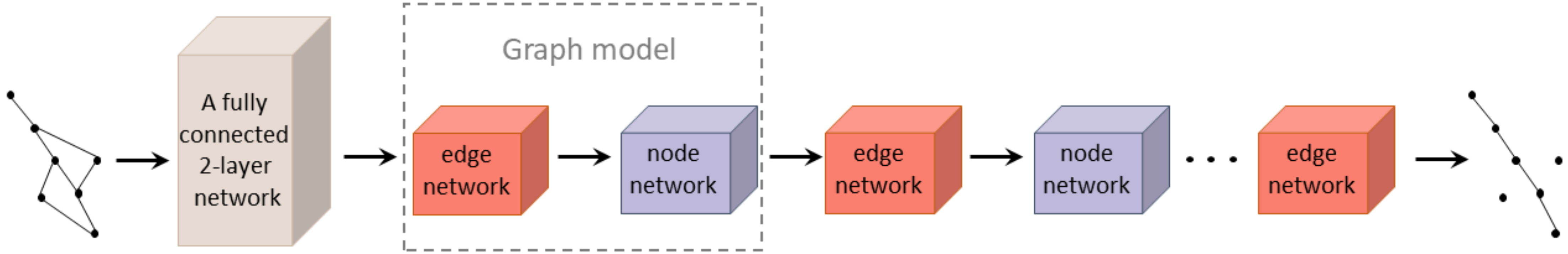
- Edge assignment
  - Hit and two adjacent hits on the left and right sides (same layer)
  - Within a certain opening angle (the next layer and one layer apart)
- Angle range
  - No sense wire efficiency
  - The junction of U-V superlayers (layers 11 and 29) appropriately amplify the threshold
- Graph representation
  - Node features (raw time, position coordinates r, ϕ of the sense wires), adjacency matrices, edge labels



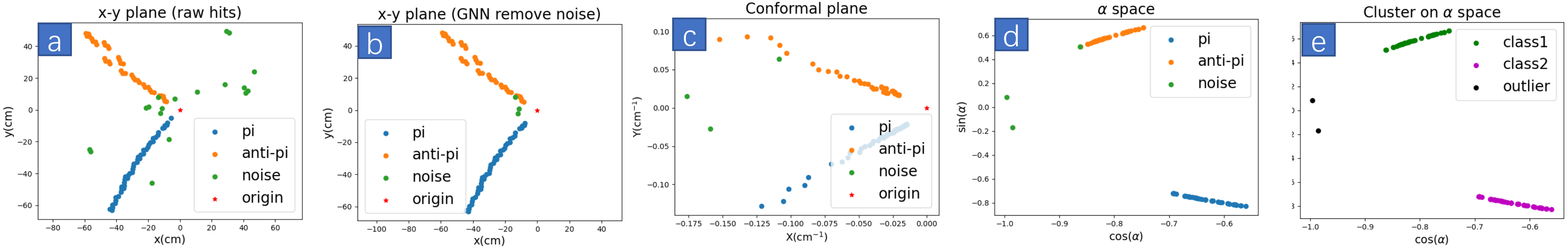


02 GNN edge classifier based on PyTorch

- Input network
  - Node features embedded in latent space
- Graph model
  - Edge network computes weights for edges using the features of the start and end nodes
  - Node network computes new node features using the edge weight aggregated features s of the connected nodes and the nodes' current features
  - MLPs
  - 8 graph iterations
- Strengthen important connections and weaken useless or spurious ones



02 Clustering based on DBSCAN



a) Original MC data sample

- $J/\Psi \to \rho^0 \pi^0 \to \gamma\, \gamma\, \pi^+ \pi^-$
- π^+, π^- : Pt (0.2GeV - 1.4GeV)

b) Remove noise via GNN

c) Transform to Conformal plane

- $\boldsymbol{X} = \frac{2x}{x^2+y^2}$ $\boldsymbol{Y} = \frac{2y}{X^2+y^2}$
- Circle passing the origin transform into a straight line

d) Transform to 'α' parameter plane

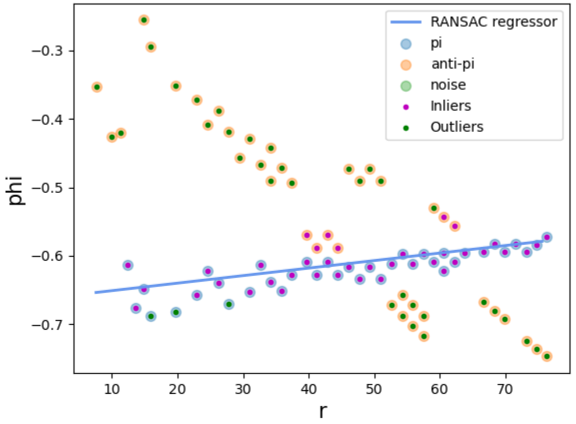
- Hits connected in the X-Y plane in a straight line
- α as the angle between the straight line and X axis
- The parameter space as cosα and sinα

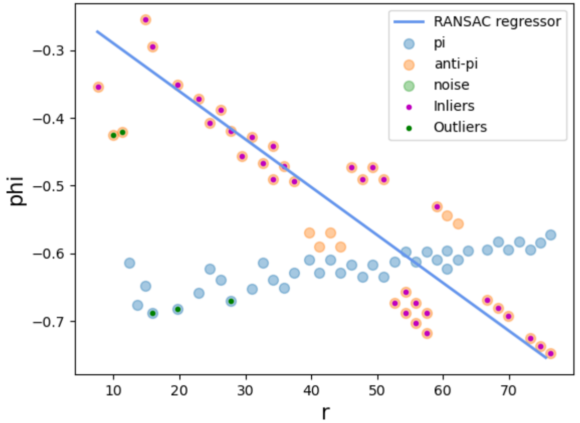
e) DBSCAN clustering in 'α'parameter plane

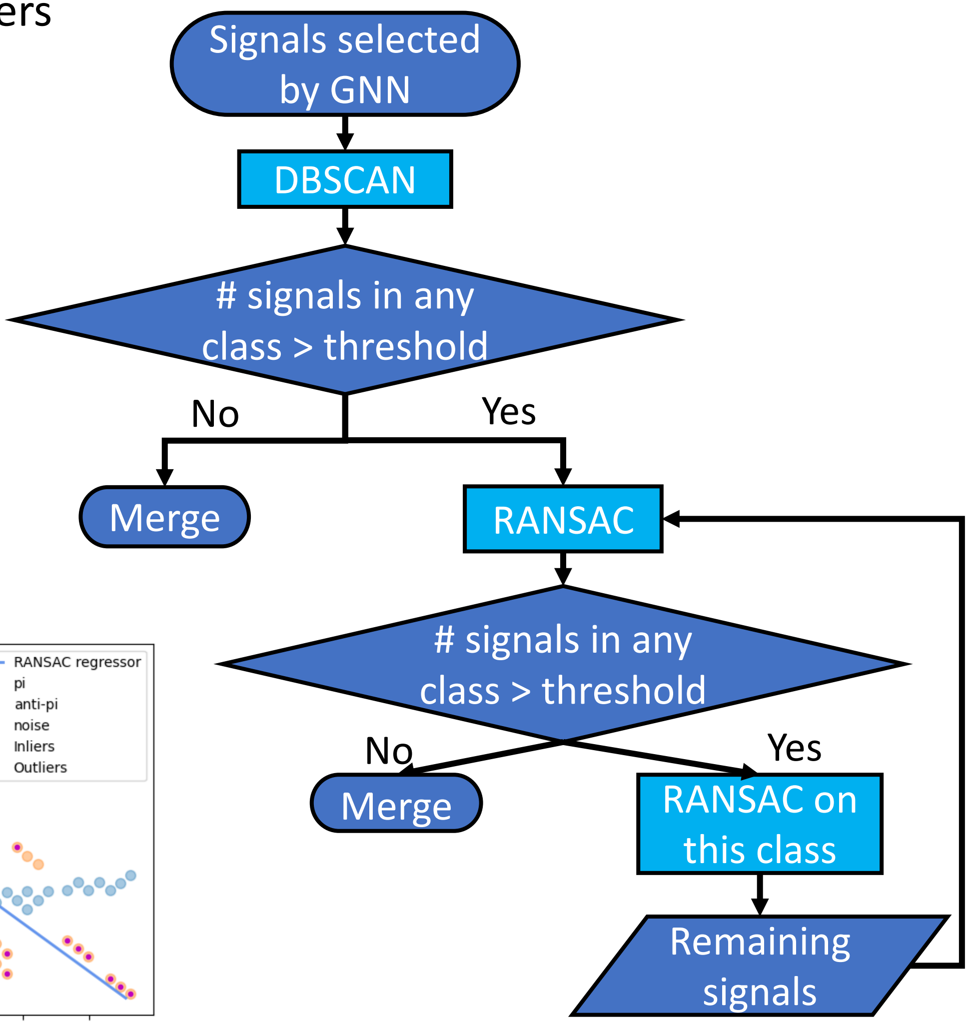
- Density-Based Spatial Clustering of Application with Noise
- Hits in a cluster are considered to be in the same track

02 Clustering salvage algorithm RANSAC

- Random sample consensus (RANCAS)
  - Estimate a mathematical model from the data that contains outliers
  - Its good robustness to noise and outliers
  - Model can be specified
- RANCAS is triggered by the events that DBSCAN processing fails
  - Polar coordinate space
  - linear model
  - Inliers → a track , outliers → other tracks
  - Stop condition: outliers < threshold

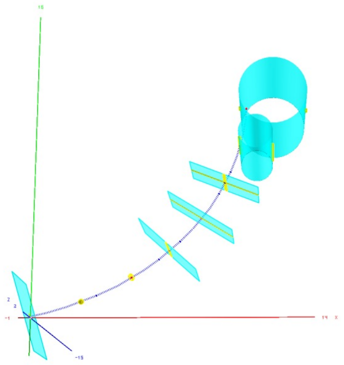






02 Track fitting

Genfit2



- A Generic Track-Fitting Toolkit
- Experiment-independent framework
- PANDA, Belle II, FOPI and other experiments
- Deterministic annealing filter (DAF) to resolving the left-right ambiguities of wire measurements

◆ Configuration: Detector geometry and materials

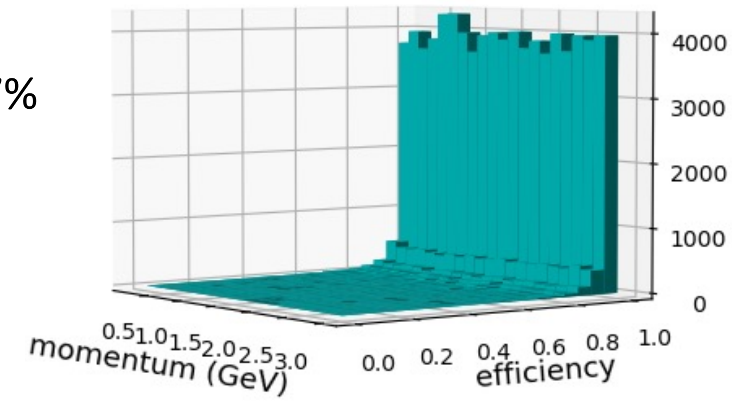
◆ Input : Signal wire position, initial values of position and momentum, particle hypothesis for e, μ, π, k, p

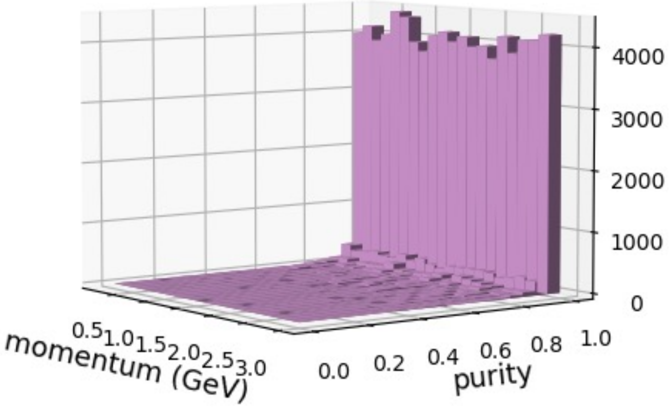
◆ Fitting procedure:

- Start 1st try: drift distance roughly estimated from TDC、ADC of sense wires
- Iteration to update information of drift distance, left-right assignment, hit position on z direction and entrancing angle in the cell et al.

03 Performance of filtering noise at BESIII

- Dataset
  - Single-particle ($e^{\pm}$, $K^{\pm}$, $\mu^{\pm}$, $p^{\pm}$, $\pi^{\pm}$) MC sample
  - 0.2 GeV/c < P < 3.0 GeV/c
  - Mixed with BESIII random trigger data as background (~45% hits)
  - Train: Validation: Test = 4: 1: 1
- Hit selection performance
  - The preliminary results show that GNN provides high efficiency and purity of hits selection
  - *Hit selection Efficiency* : $\frac{N_{signal}^{\text{predicted}}}{N_{signal}^{real}}$ 98.7%
  - *Hit selection Purity* : $\frac{N_{signal}^{\text{predicted}}}{N_{all}^{predicted}}$ 96.5%

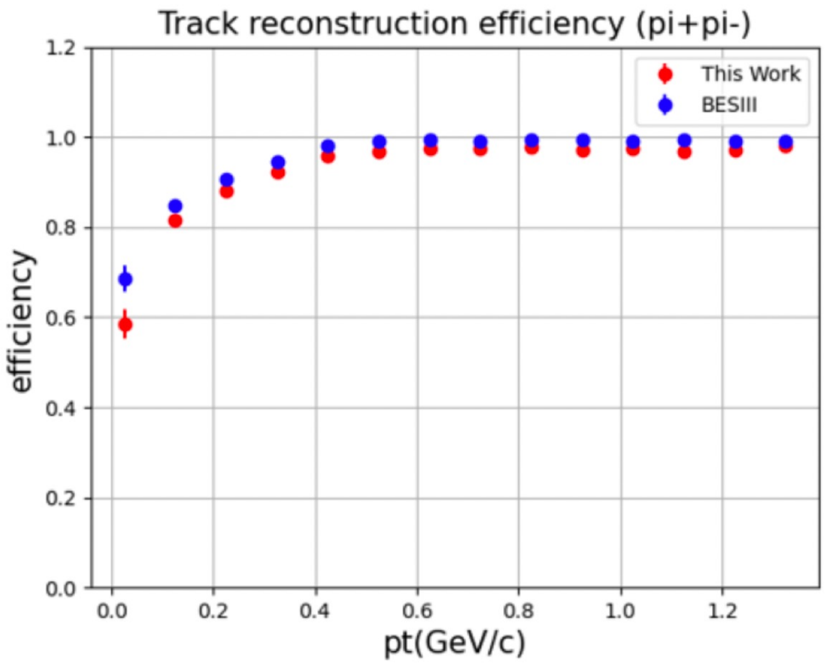


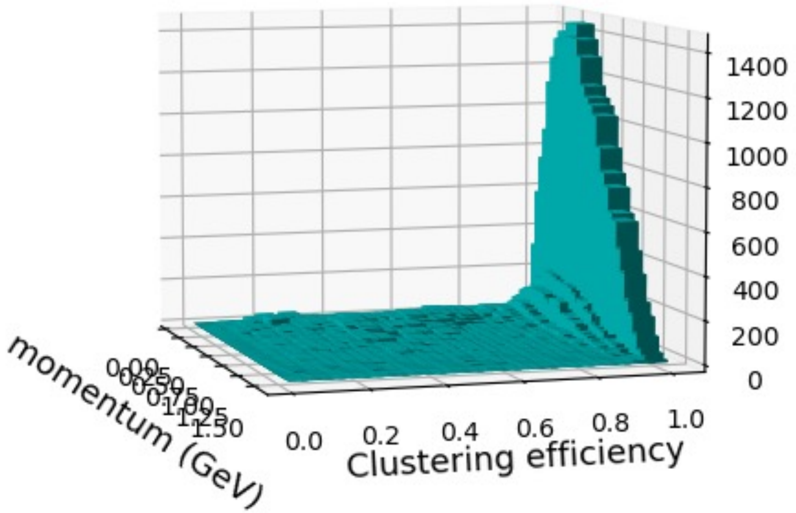


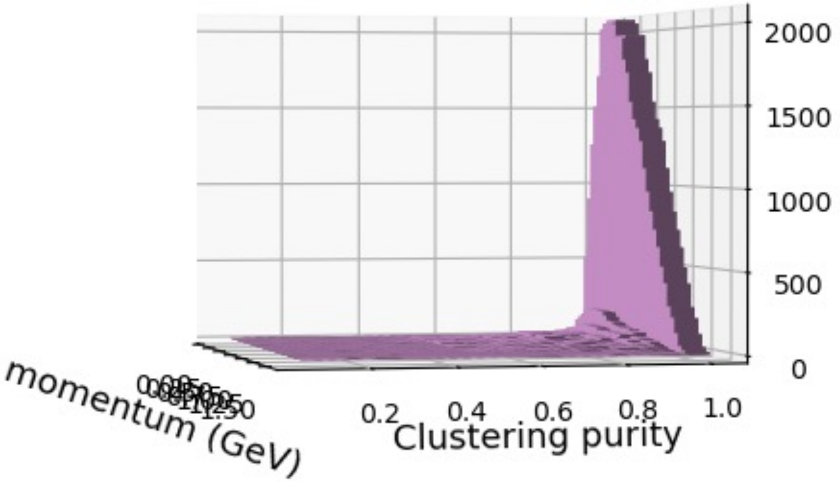
Efficiency and purity can be balanced by adjusting the model parameter

Preliminary tracking performance at BESIII

- Particle reconstructed performance
  - $J/\Psi \to \rho^0 \pi^0 \to \gamma \gamma \pi^+ \pi^-$ from MC simulation
  - *track eff* = $\frac{N_{\text{rec tracks}}}{N_{\text{total tracks}}}$
  - The preliminary results presents promising performance

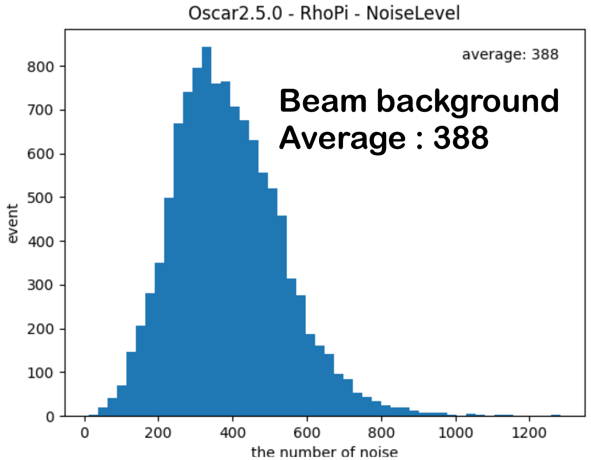


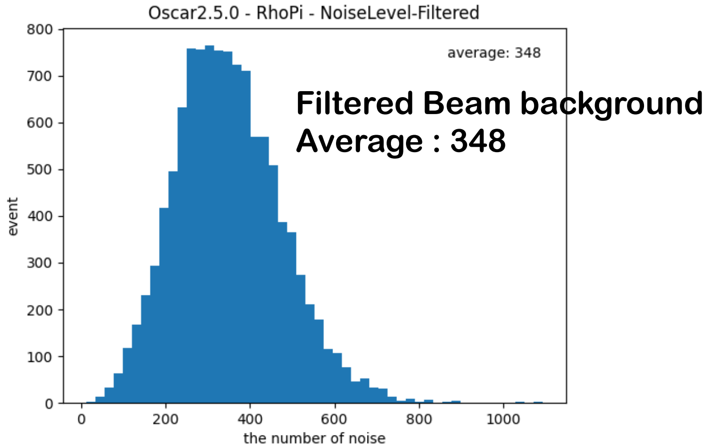


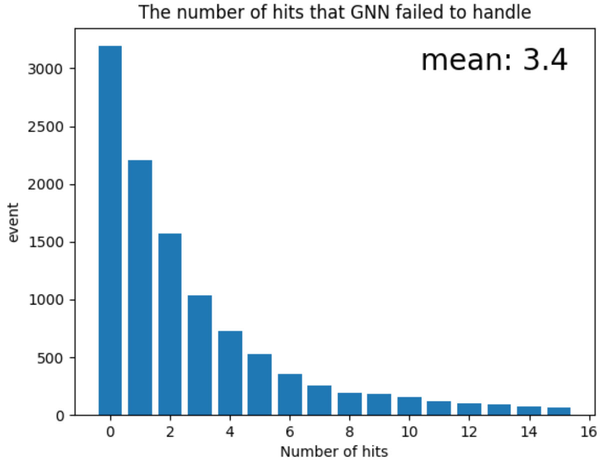


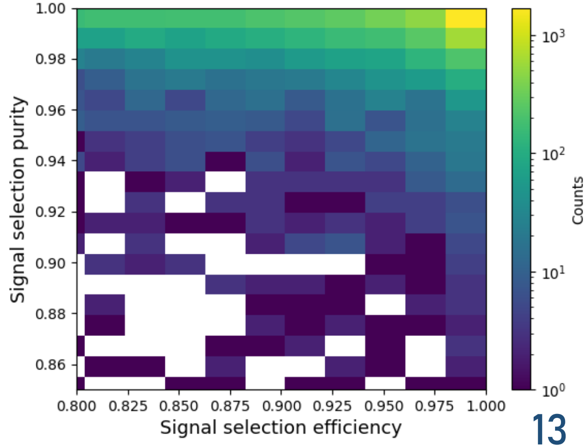
03 Performance of filtering noise at STCF

- Dataset
  - J/Ψ → ρ0 π0 → γ γ π+ π− from MC simulation
  - Mixing background (Luminosity-related, Beam-gas effect, Touschek effect) within the framework
- Hit selection performance
  - The background includes 'track' background , after removal, the noise level is 348
  - *Hit selection Efficiency* : $\frac{N_{signal}^{predicted}}{N_{signal}^{real}}$ 91.7%
  - *Hit selection Purity* : $\frac{N_{signal}^{predicted}}{N_{all}^{predicted}}$ 97.0%
  - *Remove noises rate:* $\frac{N_{noise}^{predicted}}{N_{noise}^{real}}$ 99.0%



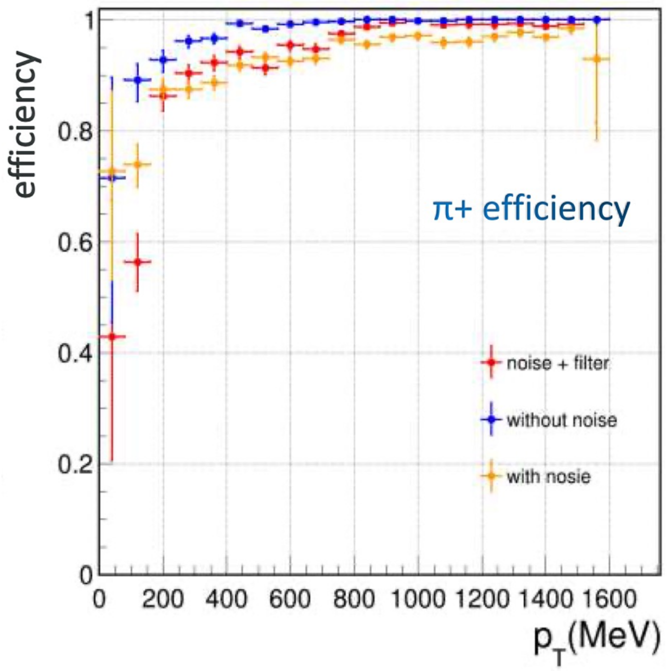


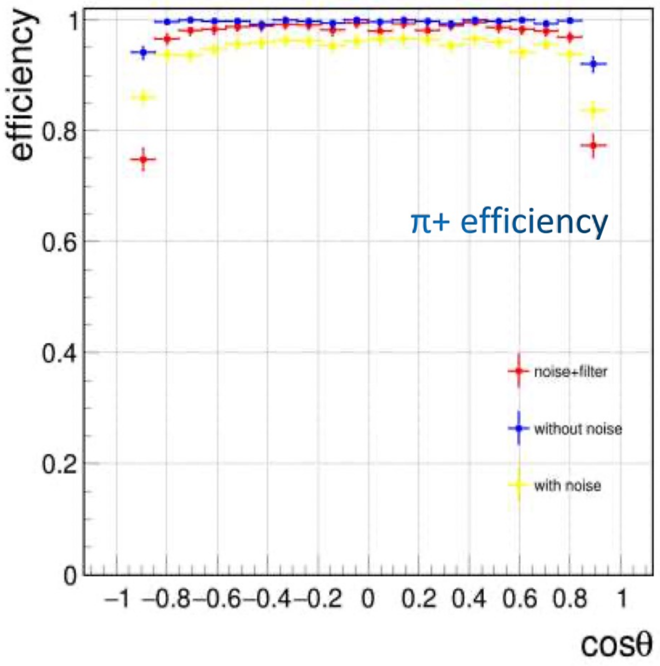




03 Performance of filtering noise at STCF

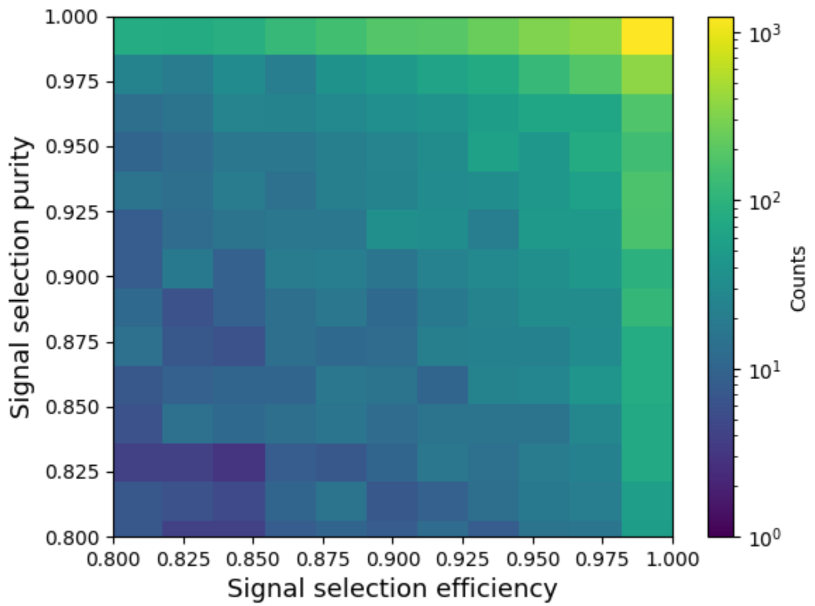
- Dataset
  - J/Ψ → ρ0 π0 → γ γ π+ π− from MC simulation
  - Mixing background (Luminosity-related, Beam-gas effect, Touschek effect) within the framework
- The reconstruction efficiency after GNN filtering noise is significantly improved
- At large $|\cos\theta|$, the tracking efficiency decreases due to fewer signal and more noise

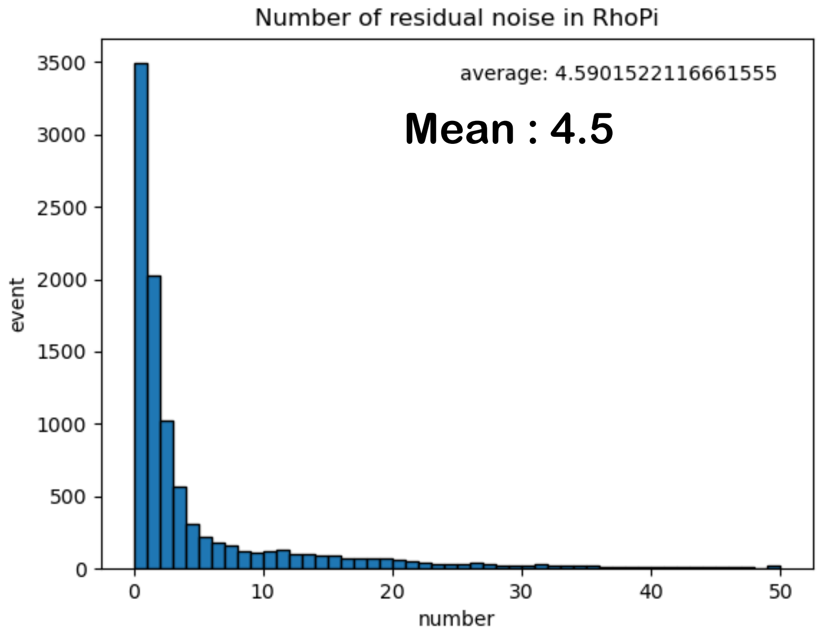




03 Performance of filtering noise at STCF

- Dataset
  - J/Ψ → ρ0 π0 → γ γ π+ π− from MC simulation
  - Mixed with 600 random trigger noises
- Hit selection performance
  - Preliminary results shows promising performance





04 Summary

- A novel tracking algorithm prototype based on machine learning method at BESIII and STCF is under development
  - GNN to distinguish the hit-on-track from noise hits.
  - Clustering method based on DBSCAN and RANSAC to cluster hits from multiple tracks
- Preliminary results on MC data shows promising performance

Outlook

- Further optimization of the cluster model is needed
- Performance verification concerning events with more tracks and long lived particle
- Check the reconstruction time

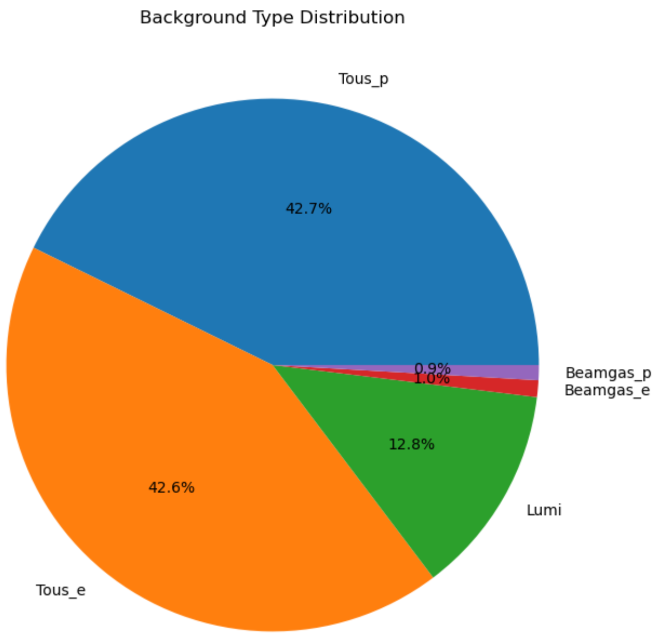
Thank you！

Xiaoqian Jia

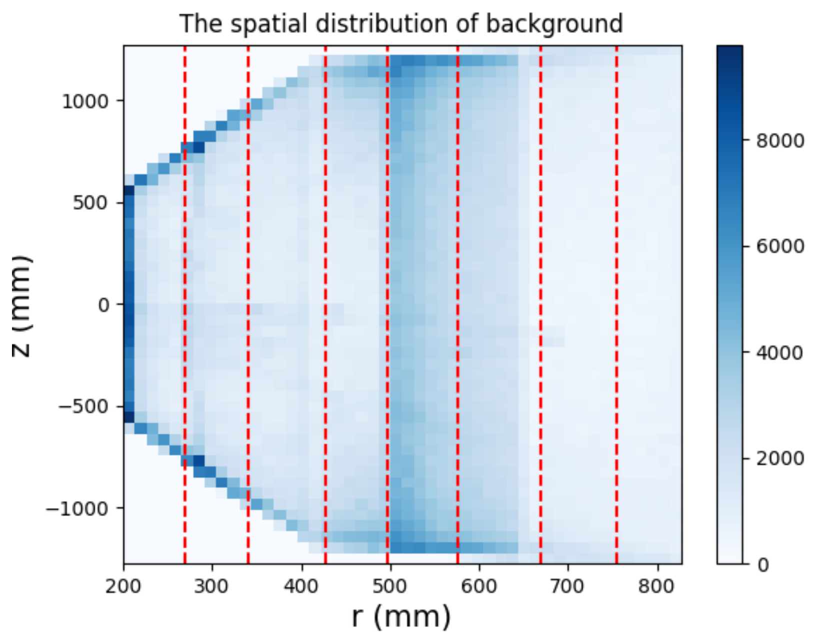
Back up

STCF background

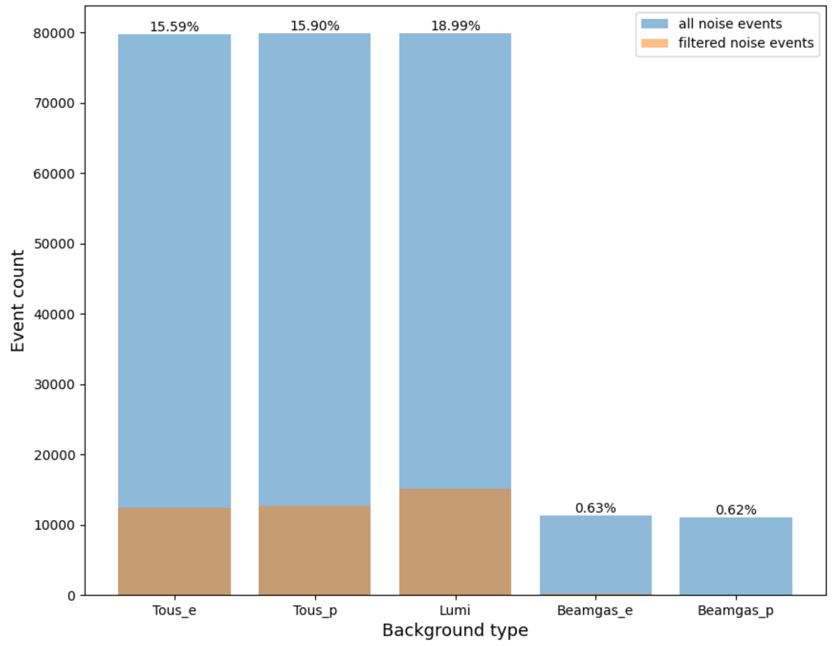
五种类型的噪声占比（hit level）



噪声R –Z空间分布

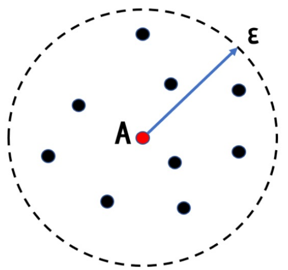


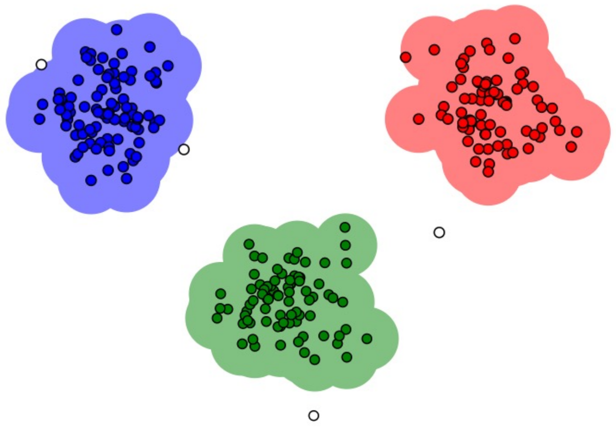
‘Track’ noise 在各类本底中的占比



DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

- A density-based clustering algorithm that can automatically discover clusters of arbitrary shapes and identify noise points
- Robust to outliers
- Not require the number of clusters to be told beforehand
- Parameter
  - Epsilon (radius of the circle to be created around each data point)
  - MinPoints (the minimum number of data points required inside that circle for that data point to be classified as a Core point)
  - Choose MinPoints based on the dimensionality (≥dim+1), and epsilon based on the elbow in the k-distance graph





RANSAC (Random Sample Consensus)

- Basic idea: randomly select a subset of data points, fit a model based on these points, and then judge whether the remaining data points belong to the inlier set by calculating their distances to the model
- Accurately estimate model parameters even in the presence of noise and outliers
- The specific steps
  - Randomly select a small subset of data, called the inlier set
  - Fit a model based on the inlier set
  - Calculate the distances between the remaining data points and the model, and classify these points as inliers or outliers based on a certain threshold
  - If the number of inliers reaches a preset threshold, the algorithm exits and the current model is considered good
  - If the number of inliers is not enough, repeat steps 1-4 until the maximum iteration times are reached
- Parameters such as threshold and iteration times need to be preset