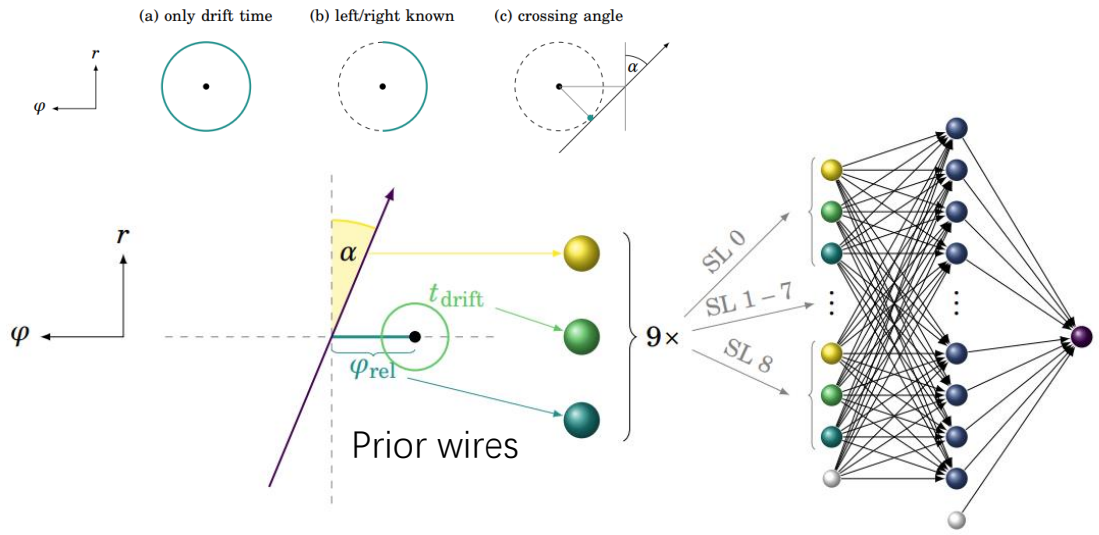
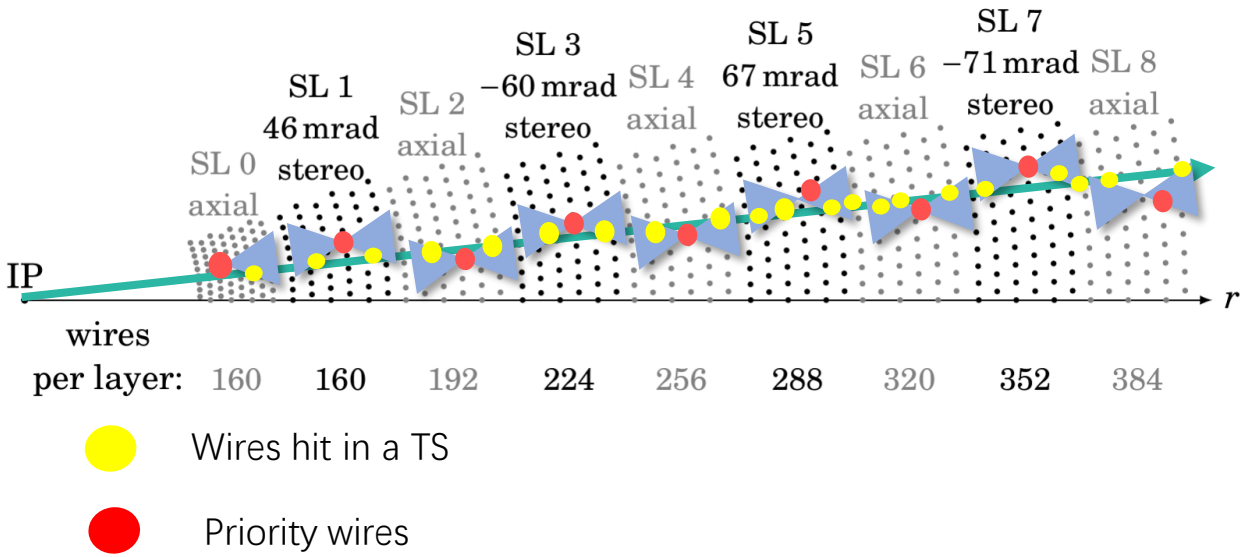
CDCTRG NN with enriched input information

Yuxin Liu

2022/11/30

Motivation





Present 3D NN use only one prior wire per every Track Segment.

With UT4 Module, more input and larger NN is possible for CDCTRG NN

For extra wire even with $\boldsymbol{\sigma_{t_{drift}} \sim 32ns}$

$$z_0 = z_{cross} - \cot\theta_0 \frac{2\alpha}{\omega}$$

$$\Delta z_{cross} = \frac{r_{wire}}{\sin\psi}\sqrt{(\Delta\phi_{cross})^2 + (\Delta\phi_B)^2}$$

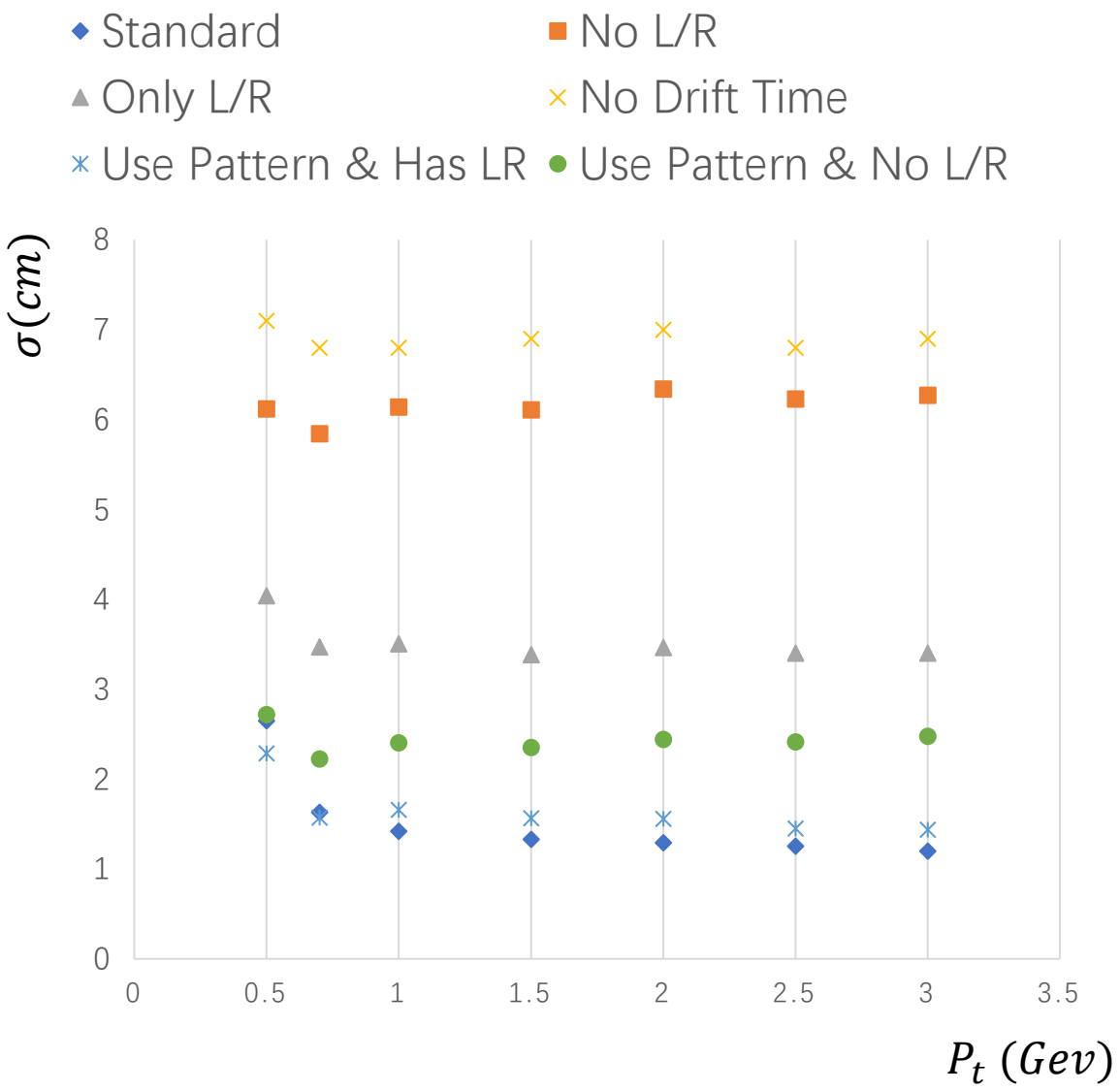
The Δz_{cross} calculated by a single wire is $(\boldsymbol{P_t > 0.4GeV})$

$$\Delta z_{cross} \sim 2.0 \text{ cm to } 3.4 \text{ cm}$$

In the same order of prior wire (0.4cm ~ 1.4cm)

Can be used to improve the resolution of NN.

Check the input for drift time

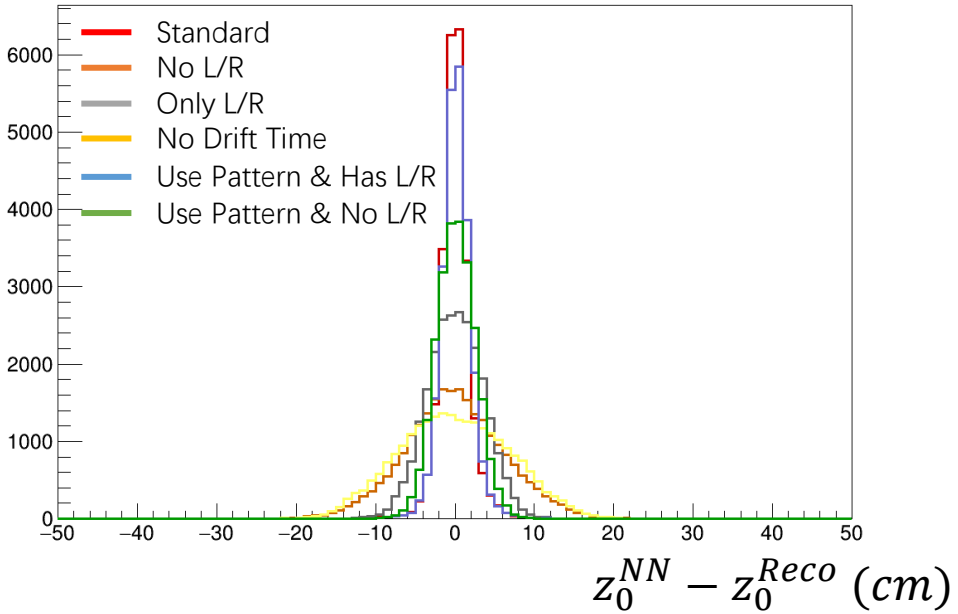


Trained NN based on single track MC w/ fann

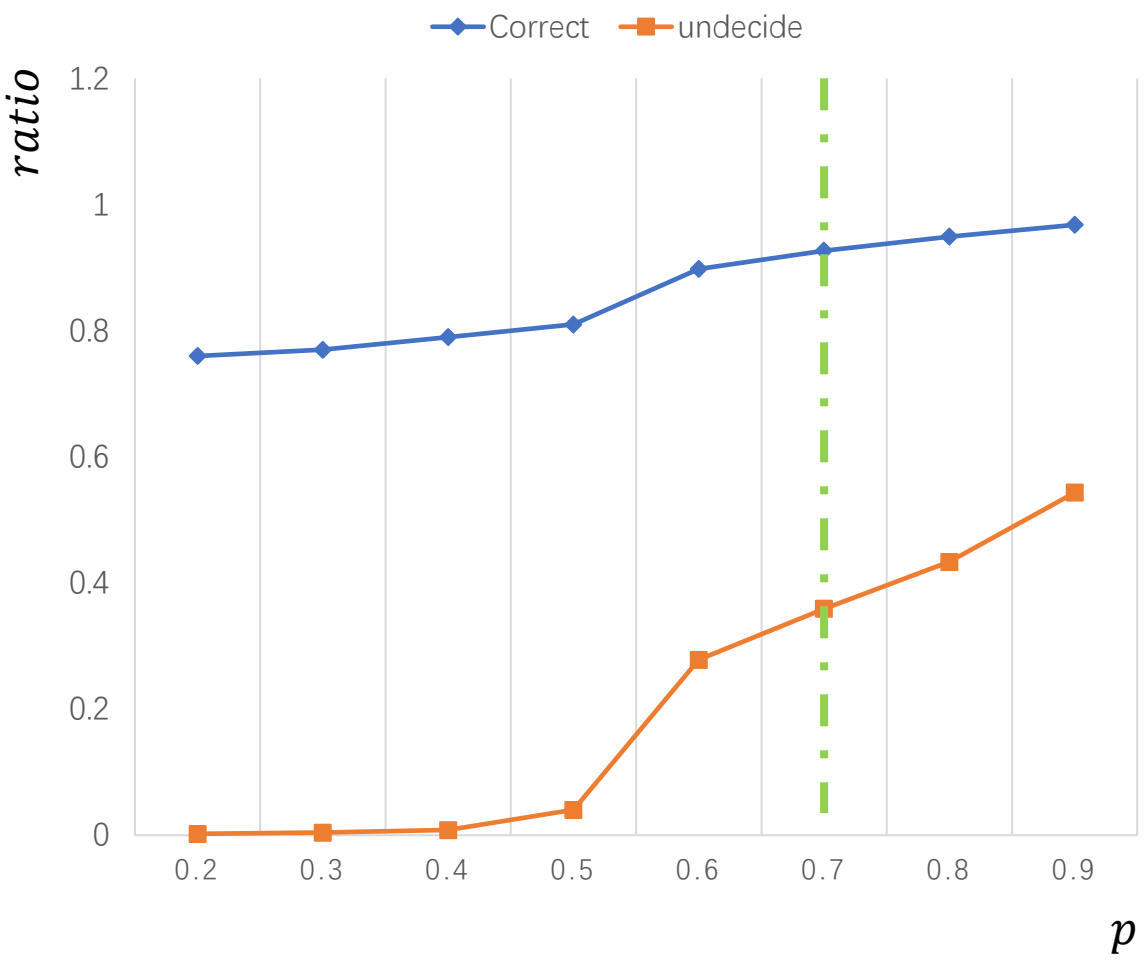
ETF : Set Event T0 as zero for precise t_{drift}

L/R is extremely important for currently NN

Pattern input can not fully replace L/R. Even with both pattern and L/R, no improvement for the standard one



Build L/R LUT table for every wires in TS



Following the old way to build up a LUT for **every wires in TS**

Use MC without Bkg first

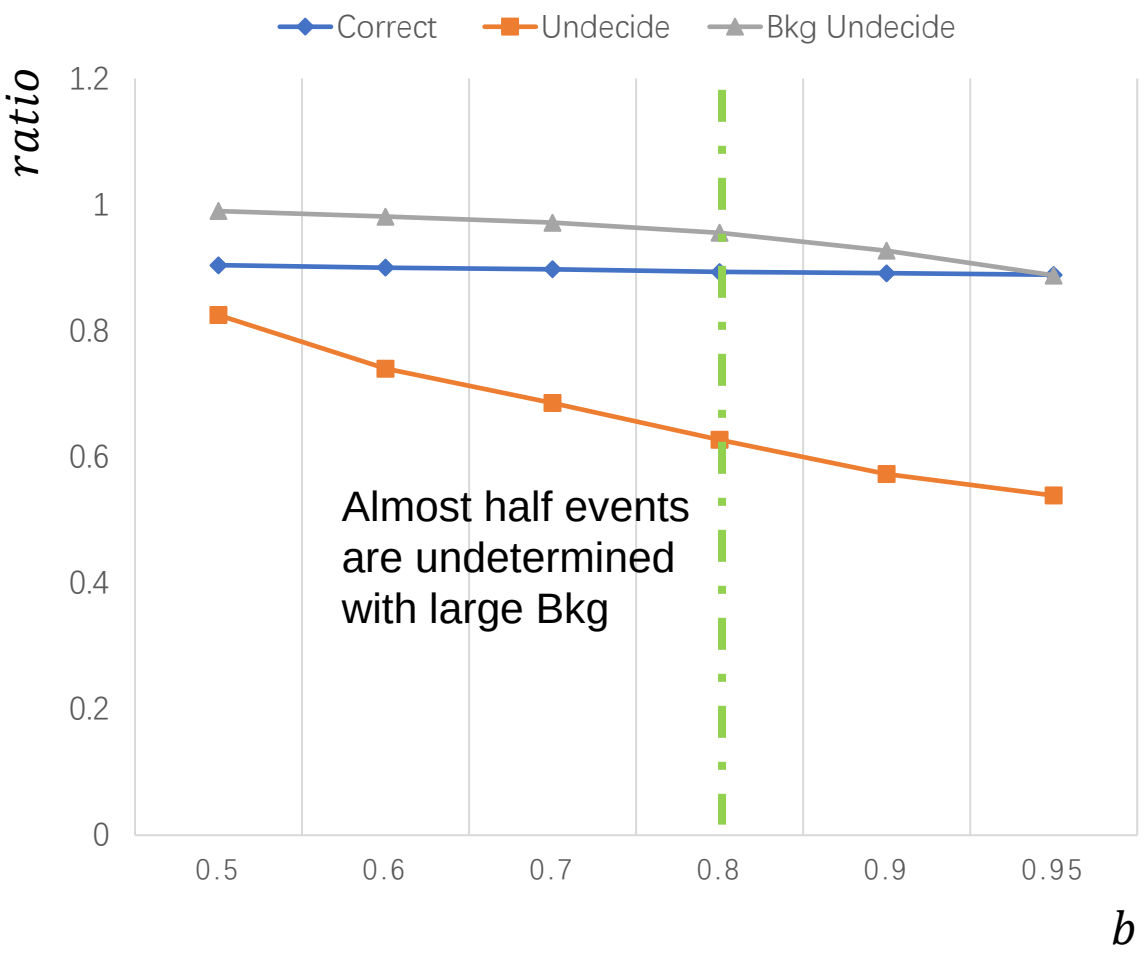
$$L/R\ state = \begin{cases} left & if\ n_L > p(n_L + n_R) + 3\sigma \\ right & if\ n_R > p(n_L + n_R) + 3\sigma \\ undecide & otherwise \end{cases}$$

$$\sigma = \sqrt{(n_L + n_R)p(1-P)}$$

Choose P = 0.7 for LUT.

Since undetermined rate is high, for more wires (>1) case, undeterminded event increases

Build L/R LUT table for every wires in TS



Following the old way to build up a LUT for **every wires in TS**

Use MC with Phase III Bkg (Coulomb, Touschek, RBB, two photon, BHWide)

$$L/R\ state = \begin{cases} left & if\ n_L > p(n_L + n_R) + 3\sigma \\ right & if\ n_R > p(n_L + n_R) + 3\sigma \\ undecide & otherwise \end{cases}$$

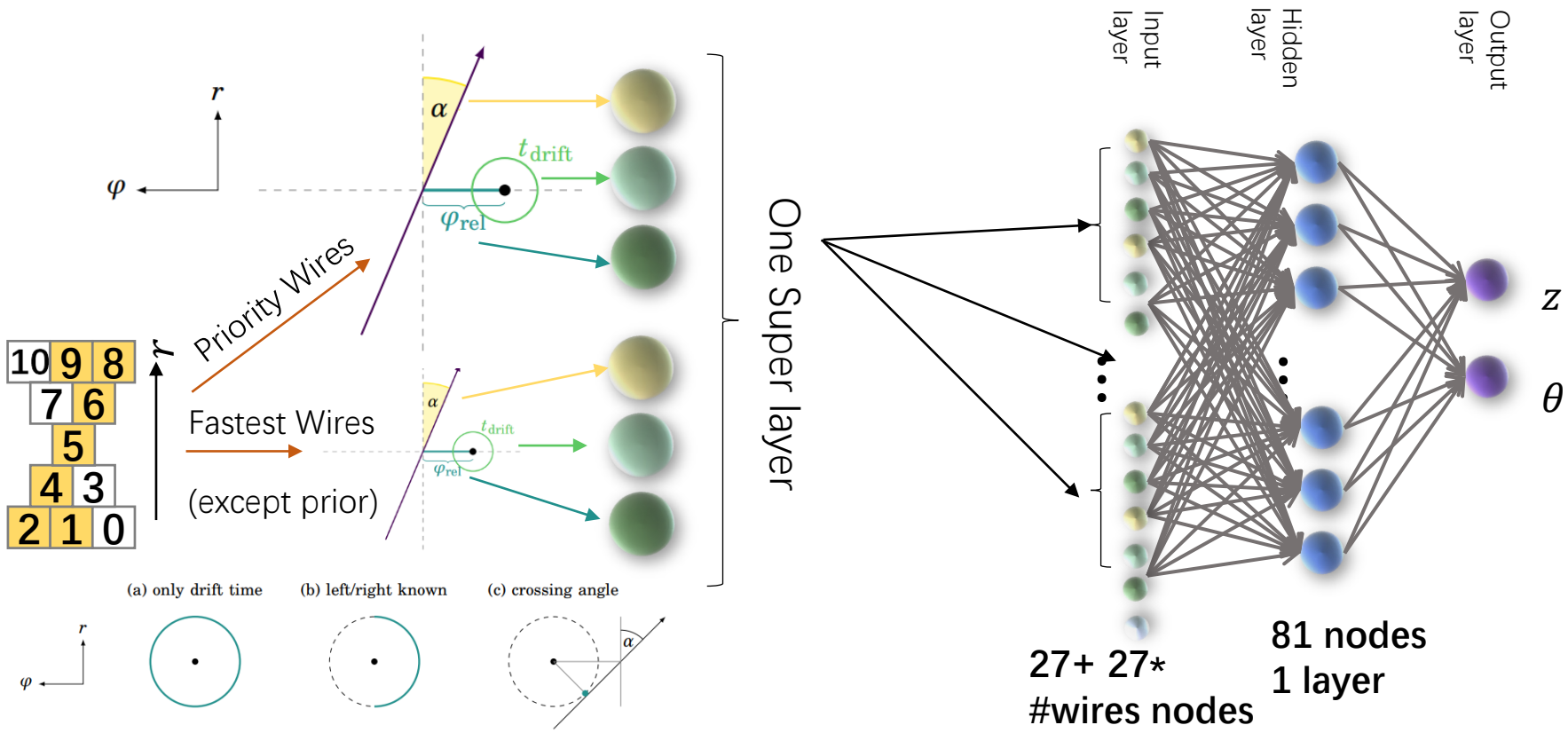
$$\sigma = \sqrt{(n_L + n_R)p(1-P)}$$

$$L/R\ state(Bkg) = \begin{cases} signal: & otherwise \\ Bkg: & if\ n_b > b(n_{Total}) \end{cases}$$

Choose b = 0.8 for LUT.

Will generated LUT with Recotrack later

First attempt: Use extra wire(s) with full information



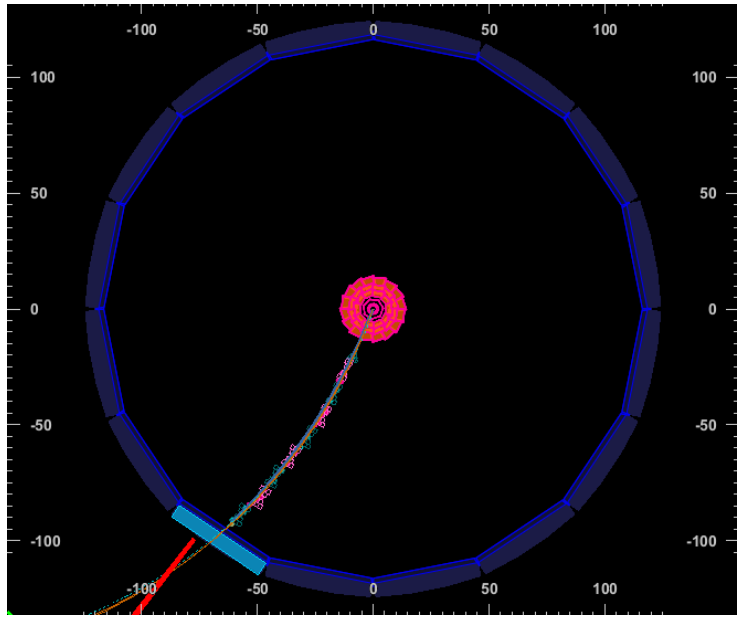
Using wires with no hit as input would decrease resolution significantly

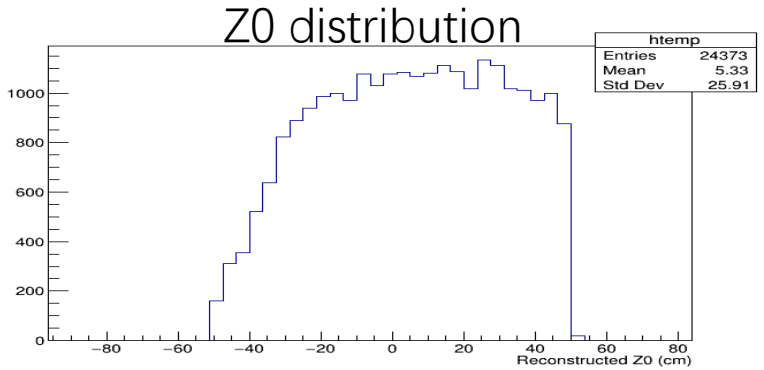
Build up L/R look up table for every wires in TS

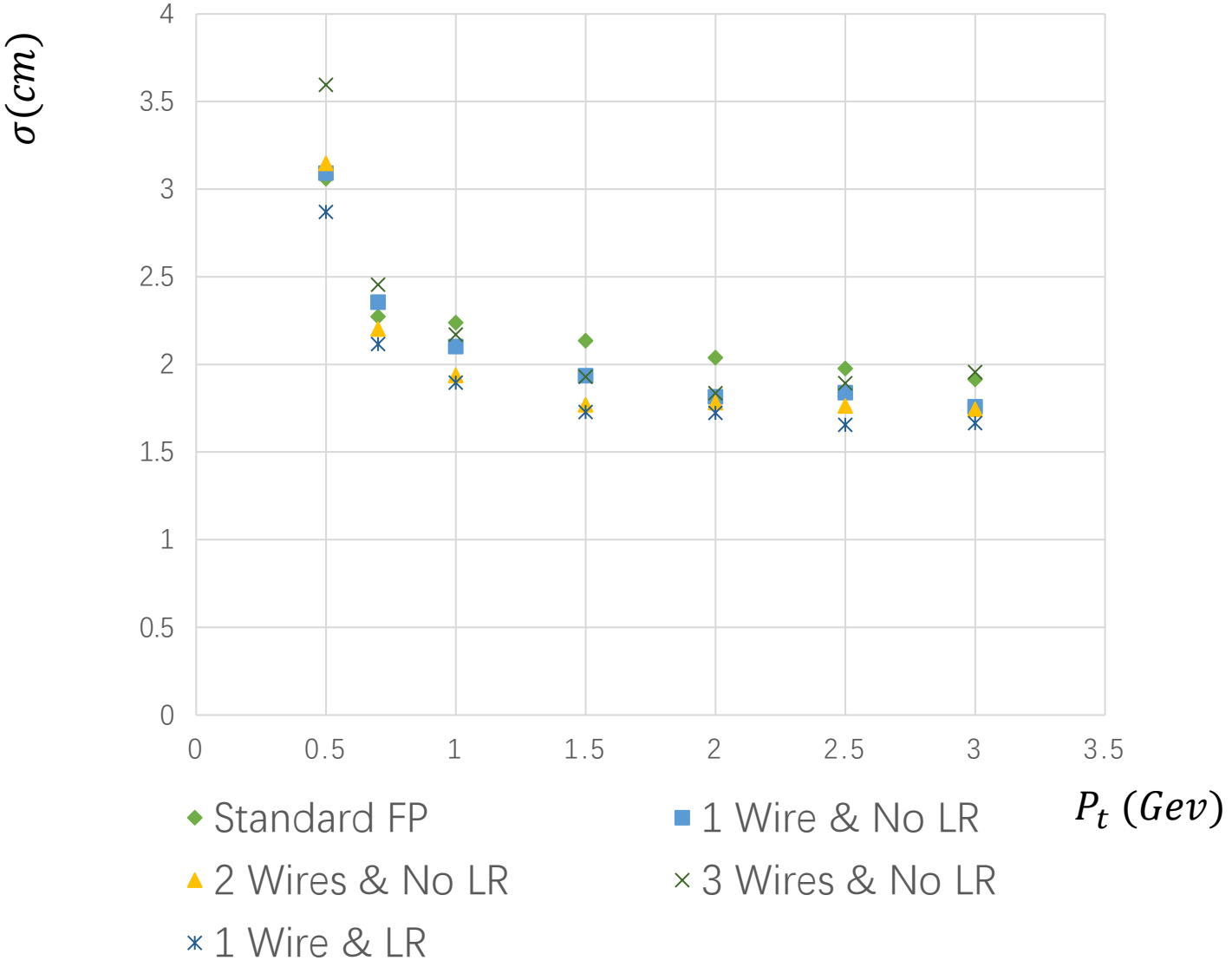
Choose the 1(2,3) wire(s) w**/ L/R know first** (if applied) and fastest t_{drift}

MC Test

MC :
Single track w/o Bkg;
uniform Pt,Φ,θ and vertex z







Not significant but could see improvement with L/R LUT

Pytorch training with real data

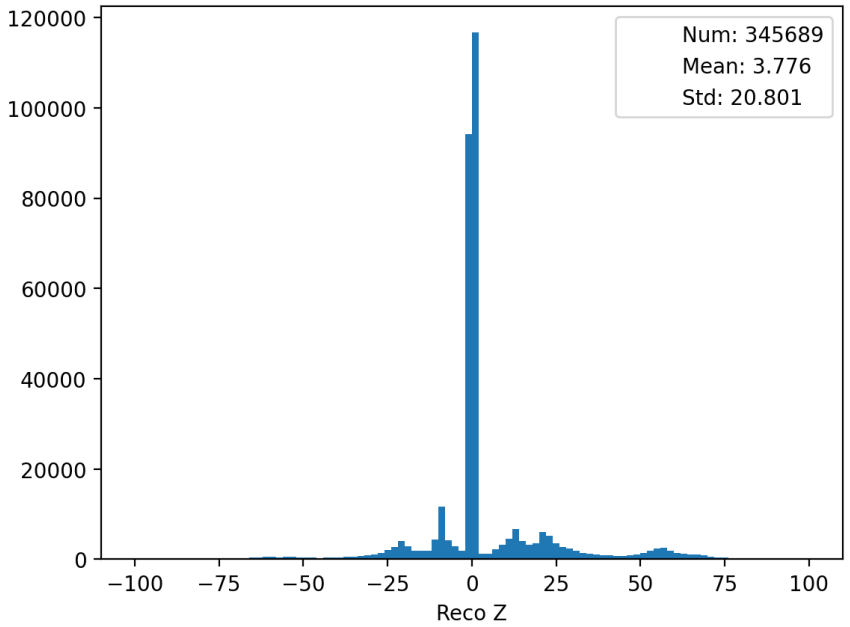
Data: exp26run1756-1780 (w/ beam reco monitor) (random separated to two set)

Generate training data with Extra 3 wires with LUT

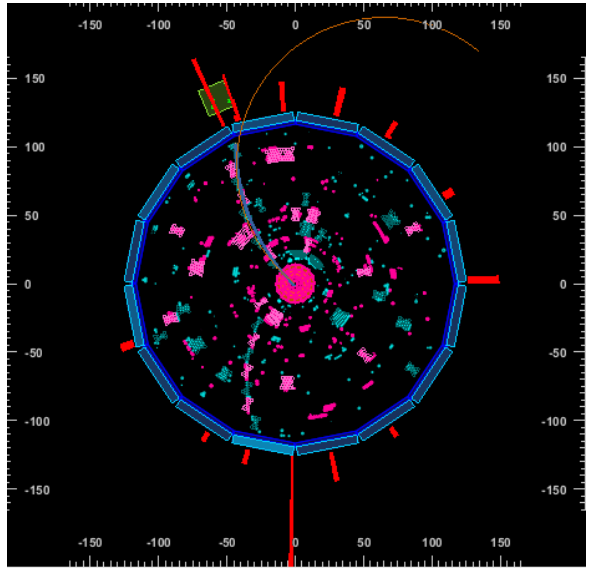
Change training method to pytorch → faster convergence and better optimization

Using simulated ETFHough

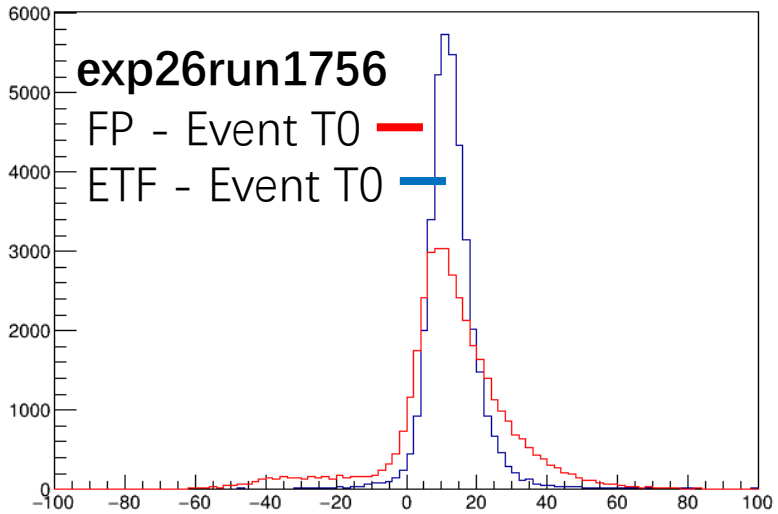
Reco Z distribution for data



Event display



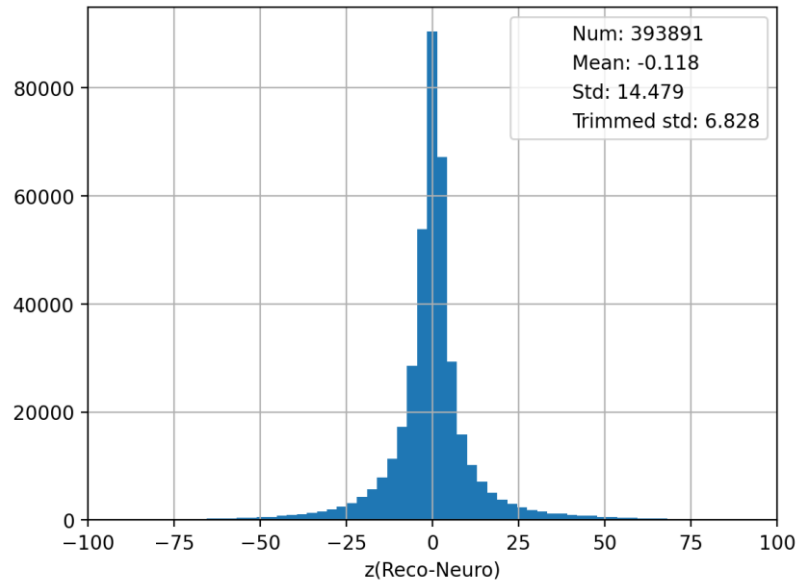
ETF compare with FP

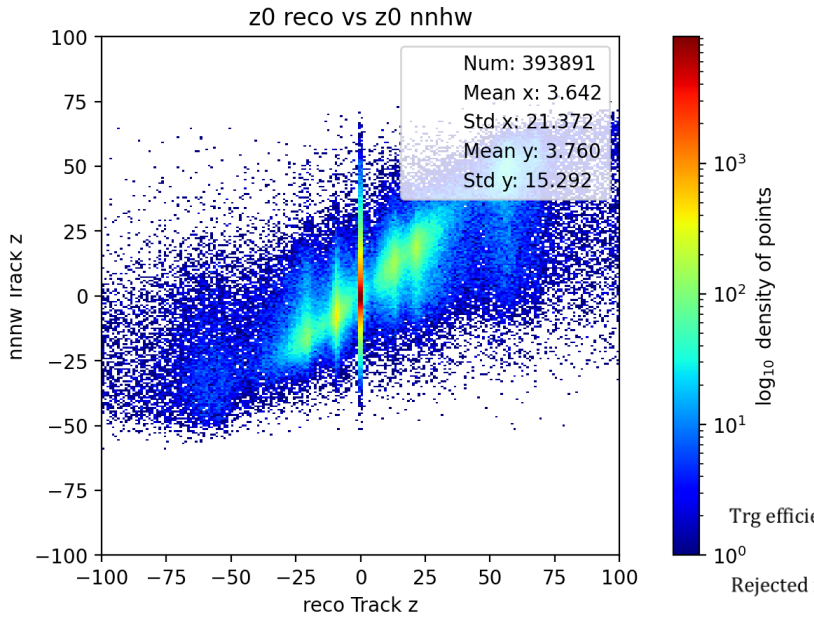


Pytorch training with real data

(Sum over all fives experts)

Standard method





Regarding a needed events as |z0|<1 and z0 cut at 15 cm

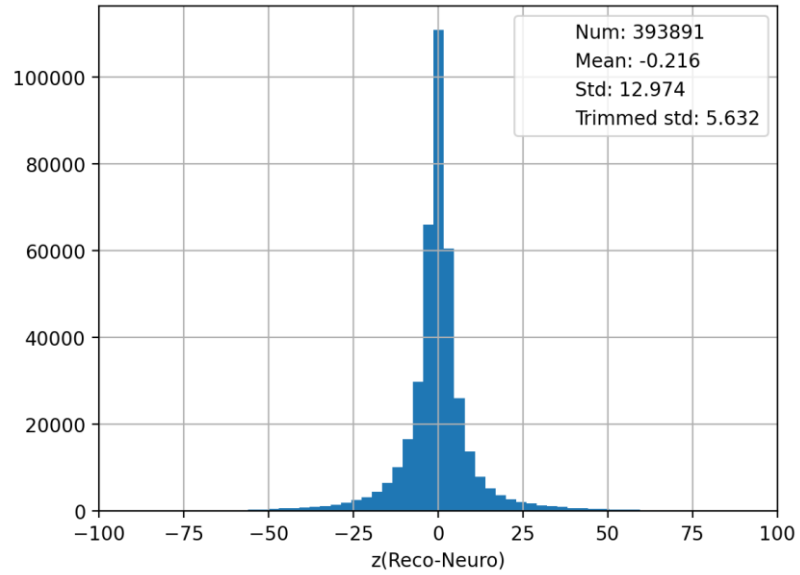
TRG Efficiency: 94.6%

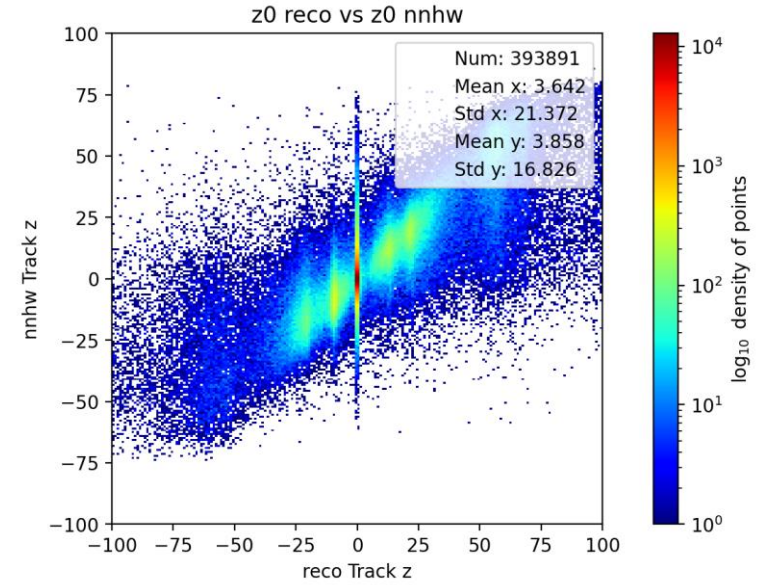
Rejected rate: 50.8%

$$\text{Trg efficiency} = \frac{\#|z_0| \text{ from RecoTracks} < 1 \;\&\&\# \; |z_0| \text{ from CDCNNTrack} < \text{cut}}{\#|z_0| \text{ from RecoTracks} < 1}$$

$$\text{Rejected rate} = \frac{\#|z_0| \text{ from RecoTracks} > 1 \;\&\&\# \; |z_0| \text{ from CDCNNTrack} > \text{cut}}{\#|z_0| \text{ from RecoTracks} > 1}$$

Extra one wire





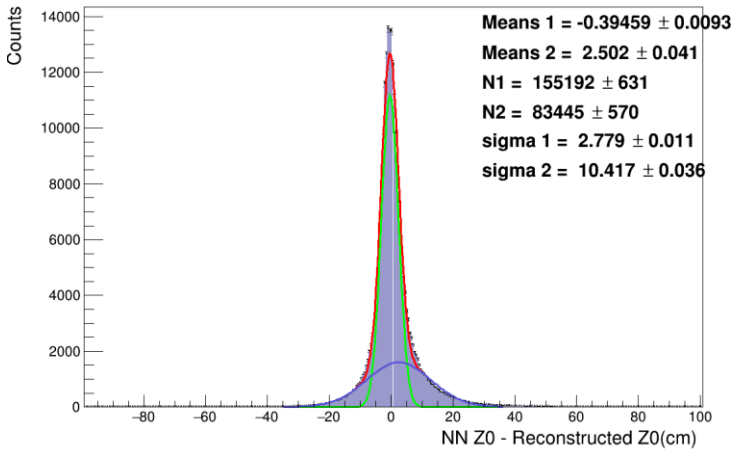
TRG Efficiency: 96.2%

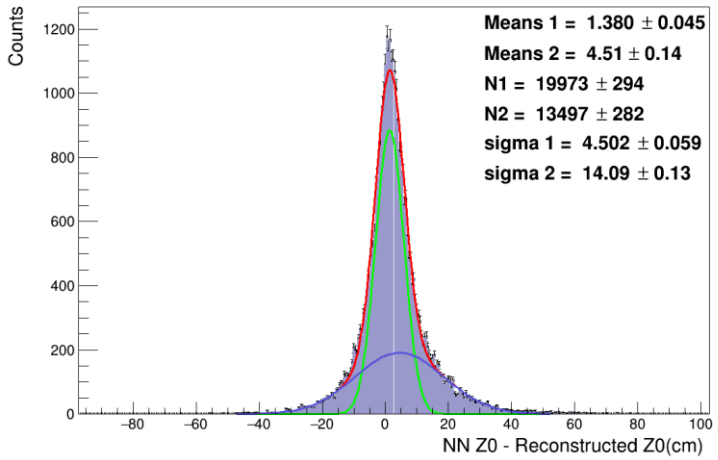
Rejected rate: 53.5%

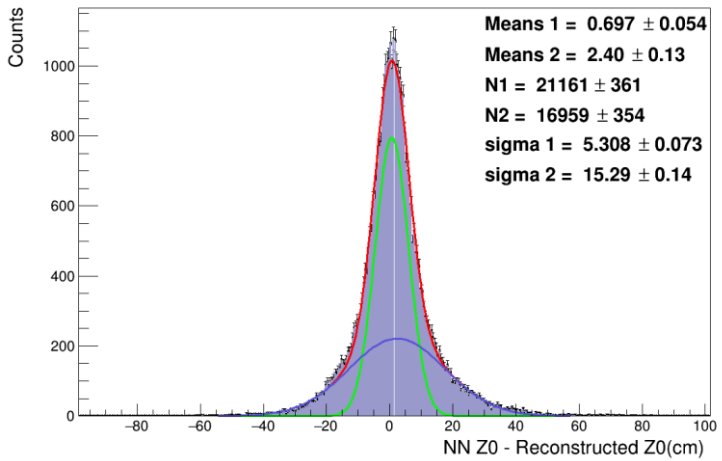
Better performance comparing with standard one, especially for large z0

Details performance at different z0

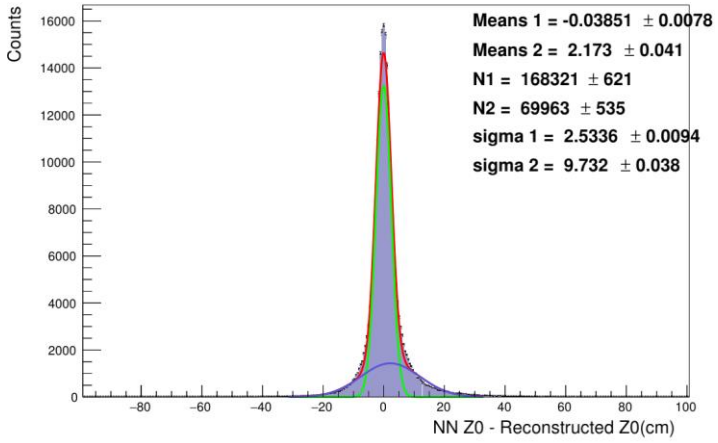
Standard method

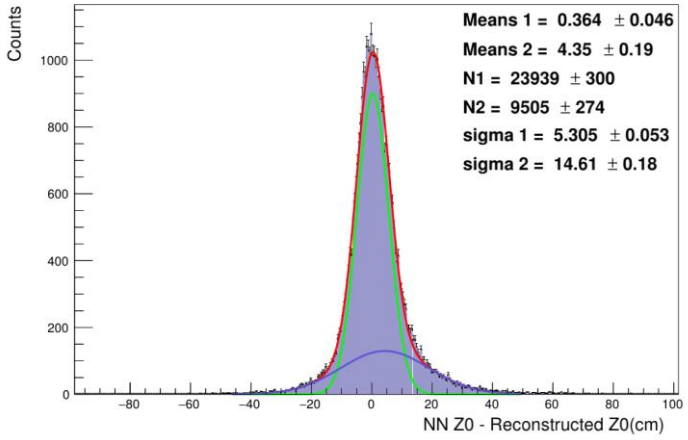


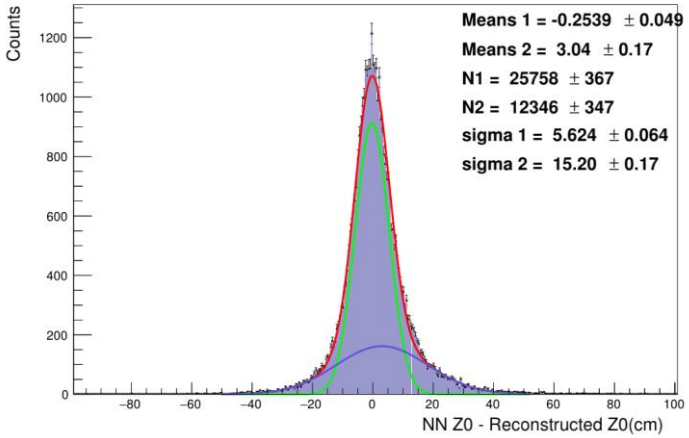




Extra one wire

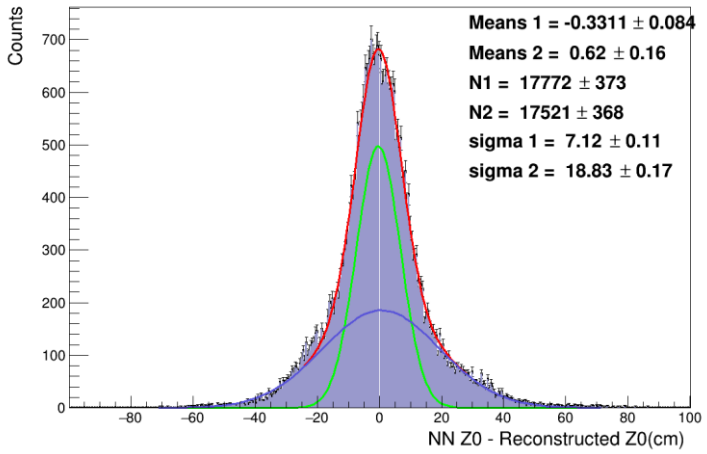


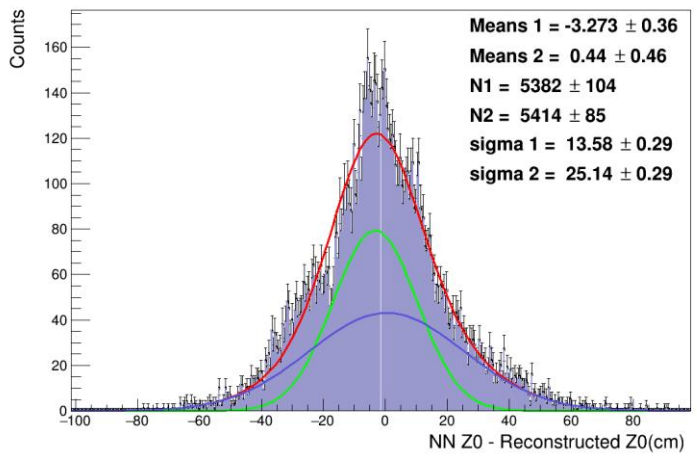


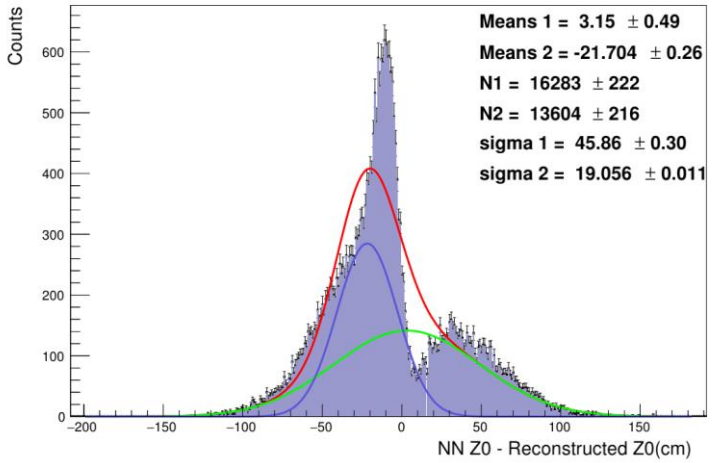


Details performance at different z0

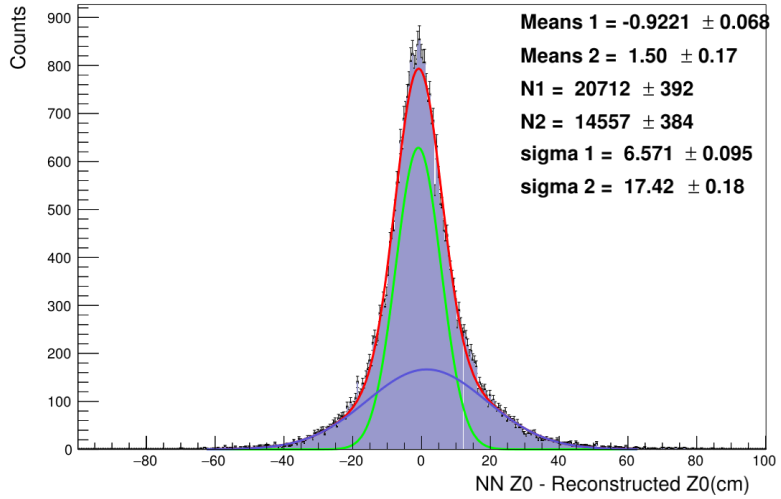
Standard method

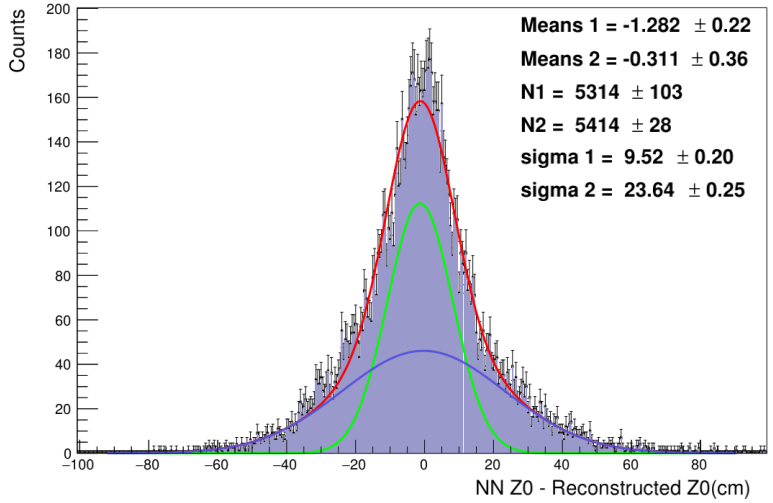


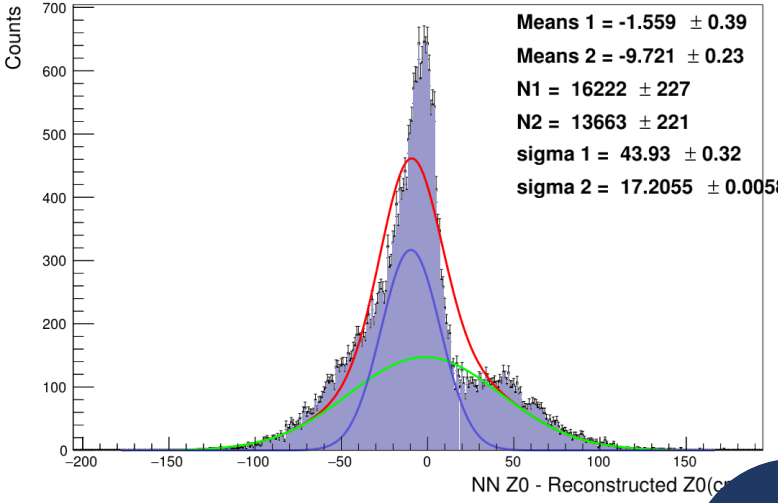




Extra one wire



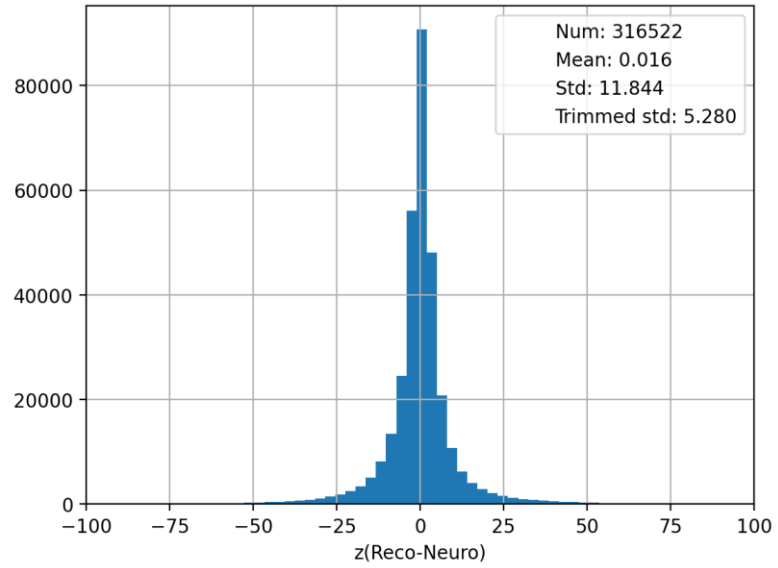




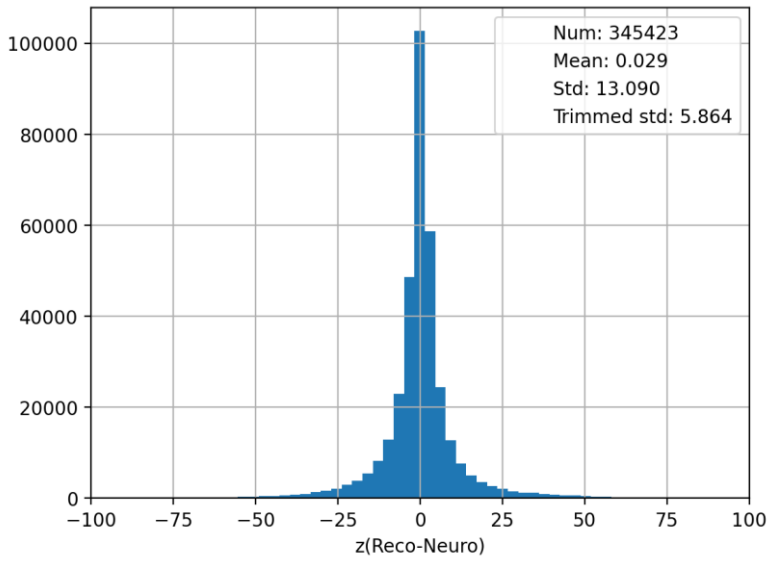
Still some “feed down” and feed up → leakage of training data?

Difference between experts for extra 1 wire case

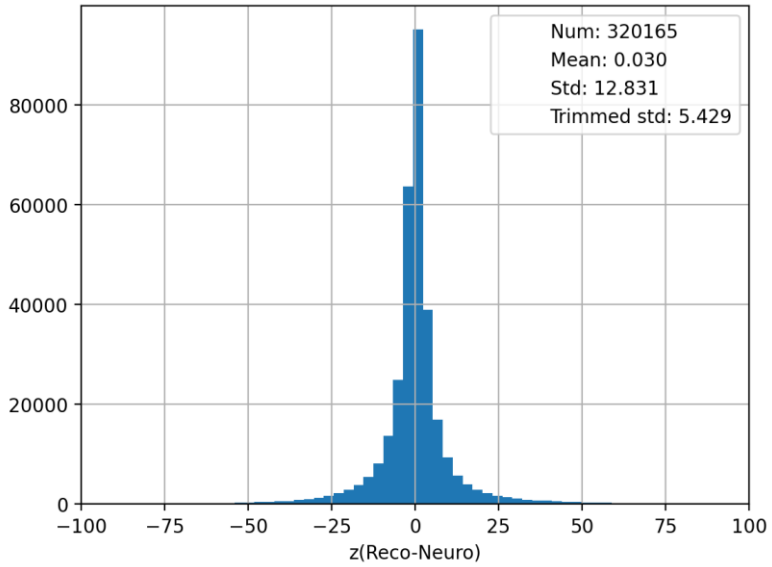
All SL have TS



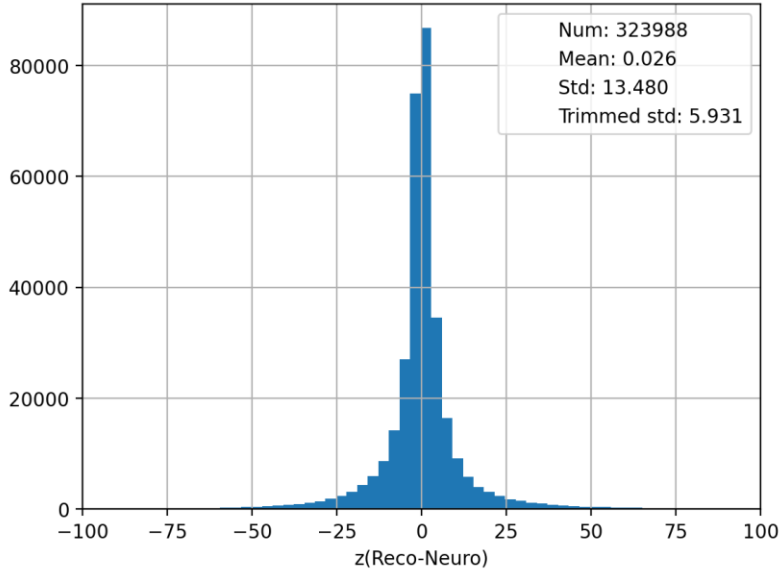
Missing SL 7



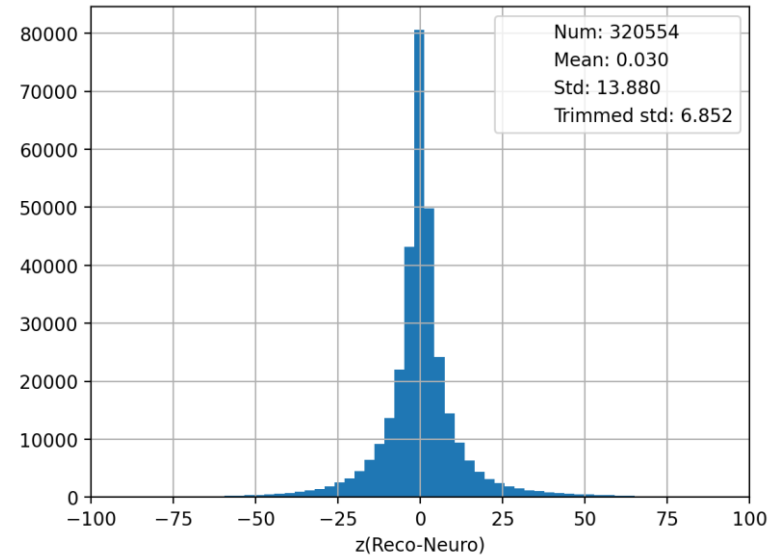
Missing SL 5



Missing SL 3



Missing SL1

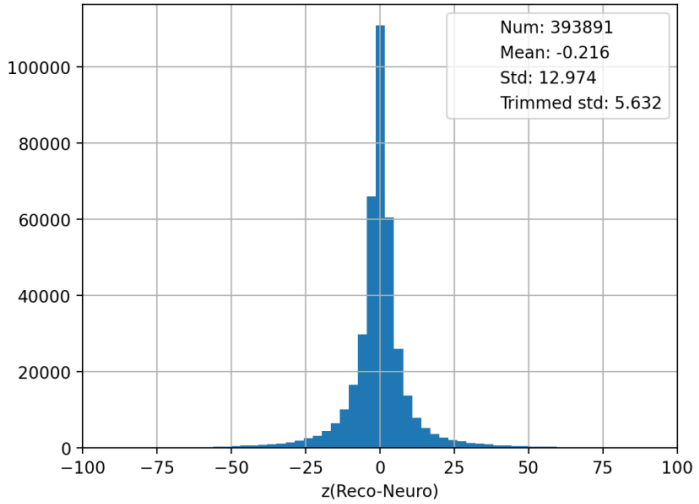


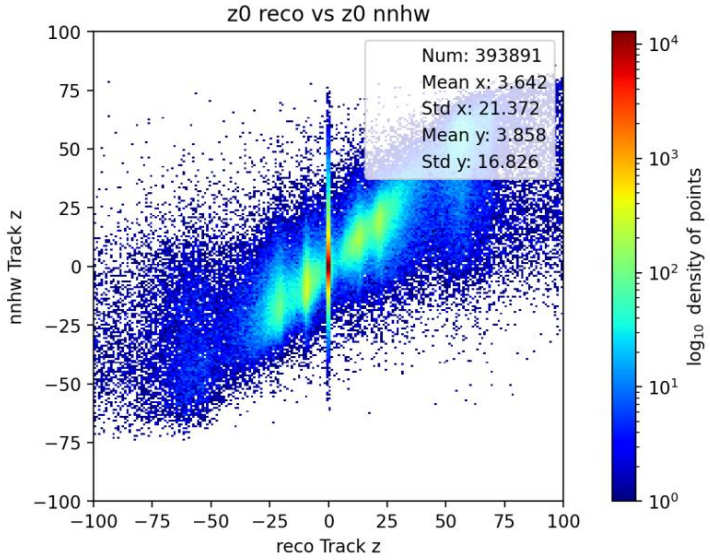
w/ missing SL1 still got worst result.

(However, expert 0 case dominate the events in exp26run1756-1780 >80%)

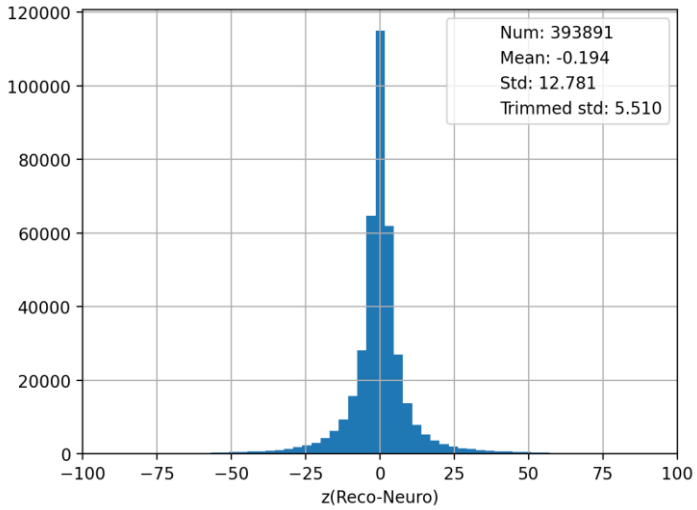
More Extra wires?

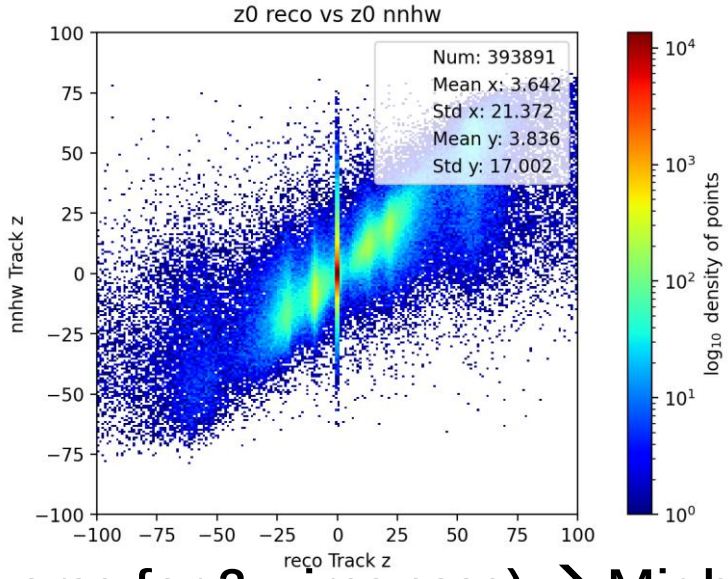
Extra one wire



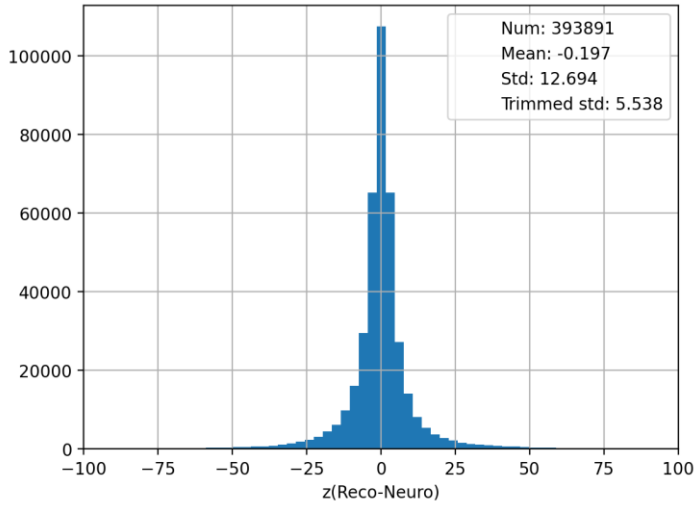


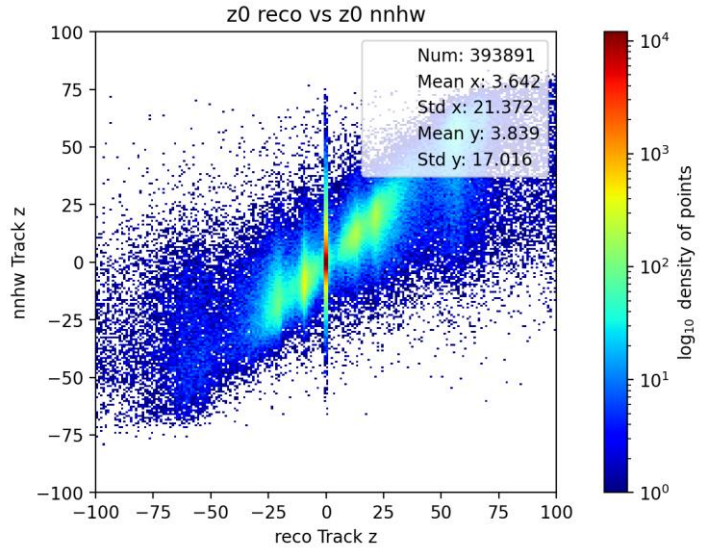
Extra two wires





Extra three wires





No large improvement (even worse for 3 wires case) → Might due to the L/R undetermined wires or not enough hidden layer/ hidden nodes

Summary & Plan

Summary

a)Add 1 extra wire could make improvement for the CDCTRG NN

b) Feed down and feed up still exist –(reshape of dataset needed?)

c) More than one wire make little difference at current NN structure.

Plan

a) Adding ADC into data selection for NN

b) Try different Hidden layer & Hidden nodes for 2(3) extra wires case

c) Reshape dataset may help for fix “feed up” and “feed down”?

d) Directly output prediction for fake TrackSegment with NN?

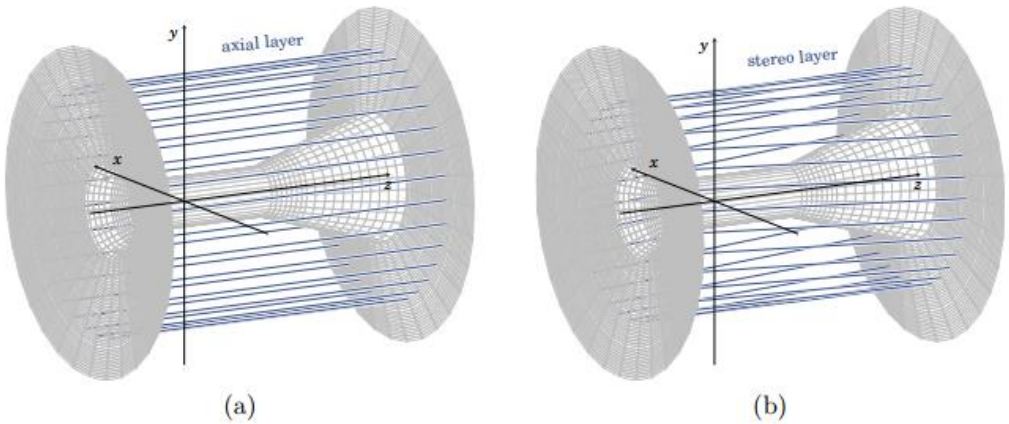
Thanks for your listening and attention!

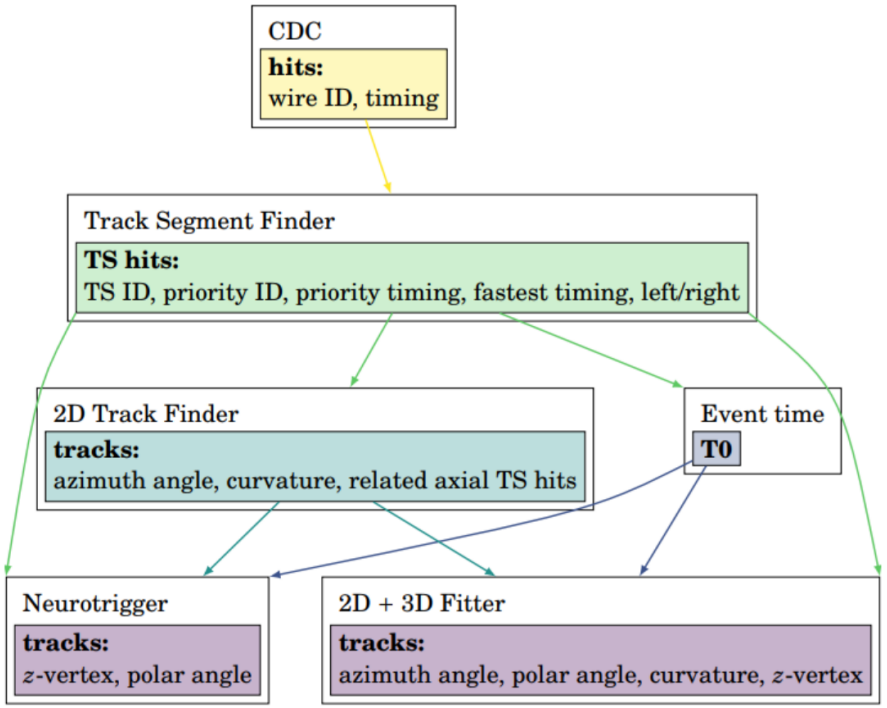
BACK UP

$$\text{Trg efficiency} = \frac{\#|z_0| \text{ from RecoTracks} < 1 \;\&\&\# \; |z_0| \text{ from CDCNNTrack} < \text{cut}}{\#|z_0| \text{ from RecoTracks} < 1}$$

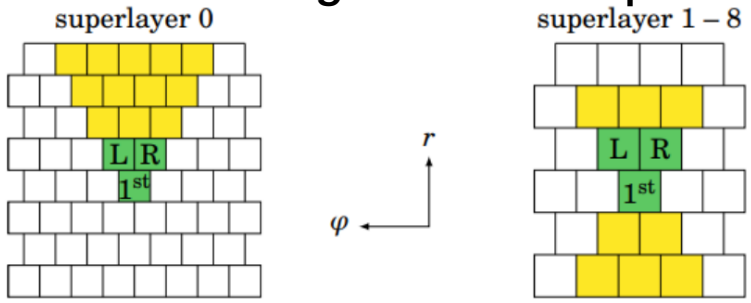
$$\text{Rejected rate} = \frac{\#|z_0| \text{ from RecoTracks} > 1 \;\&\&\# \; |z_0| \text{ from CDCNNTrack} > \text{cut}}{\#|z_0| \text{ from RecoTracks} > 1}$$

Introduction-CDC first level TRG

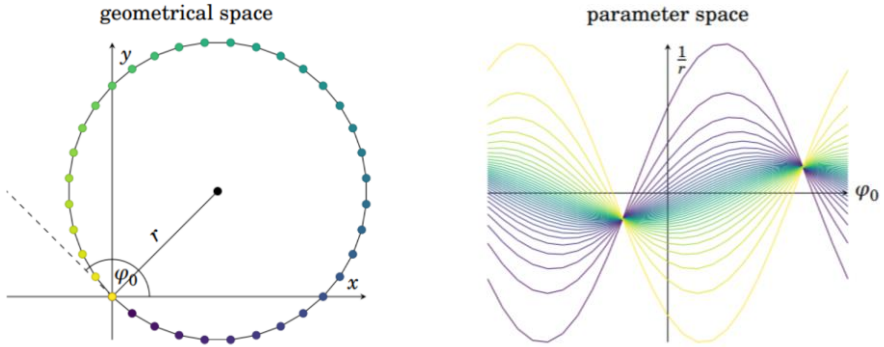




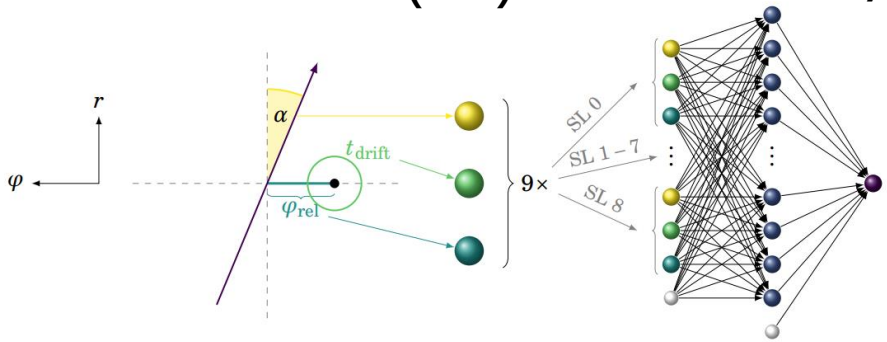
Track Segment build up



2D Track reconstructed (using hough transformation



3D Neural Network (NN) to calculated z / θ

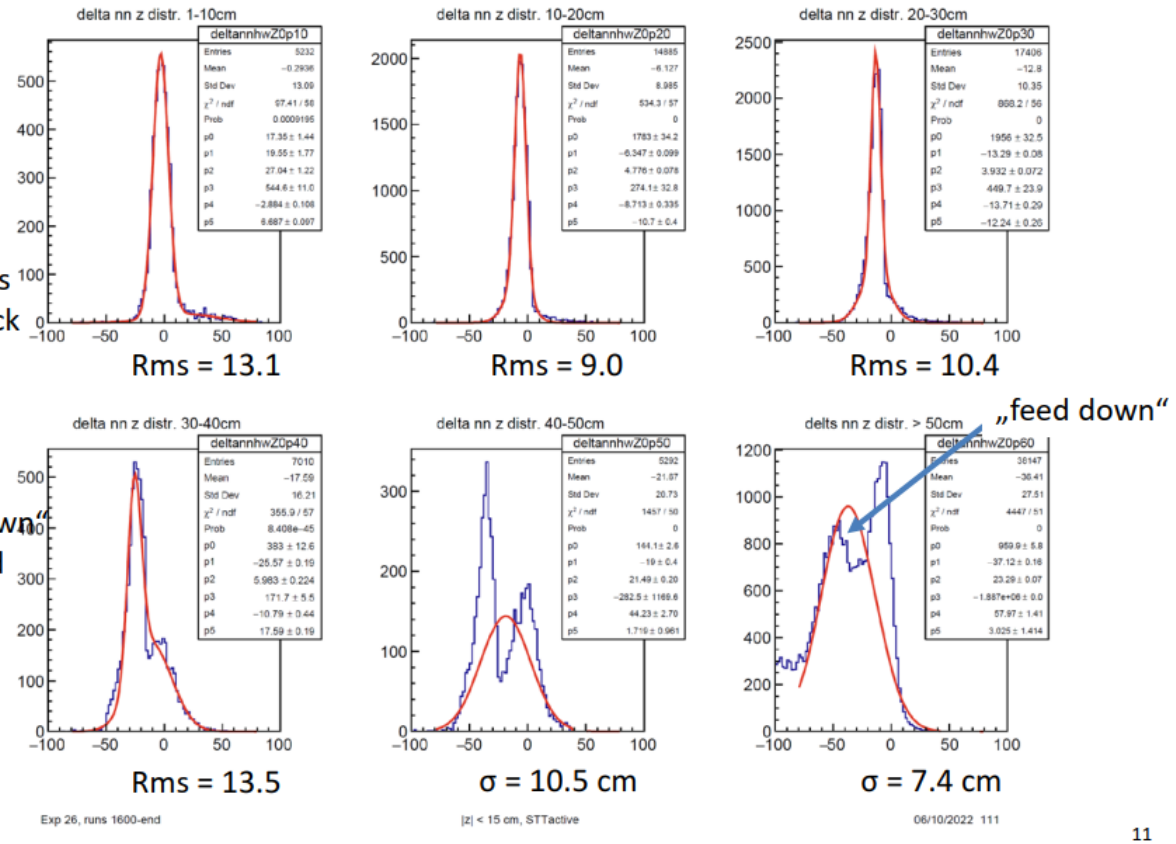


Introduction-Current CDC NN Trigger performance

Δz = z(nn) - z(reco)

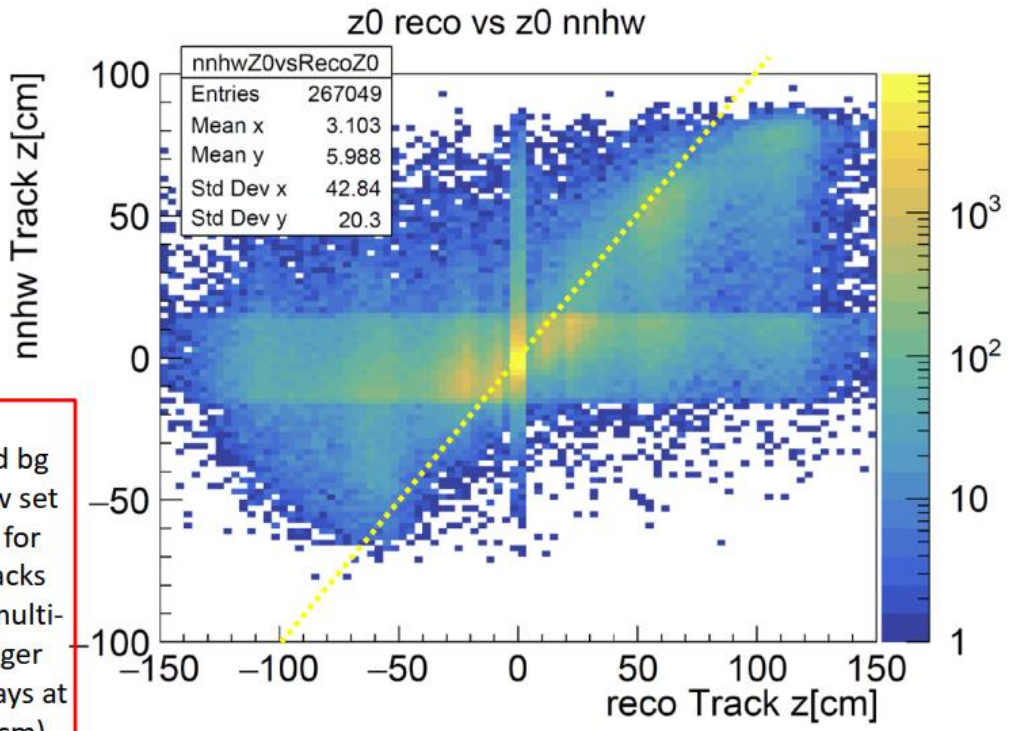
Slight bias of nn track towards smaller values

„feed-down“ increased



11

Exp. 26



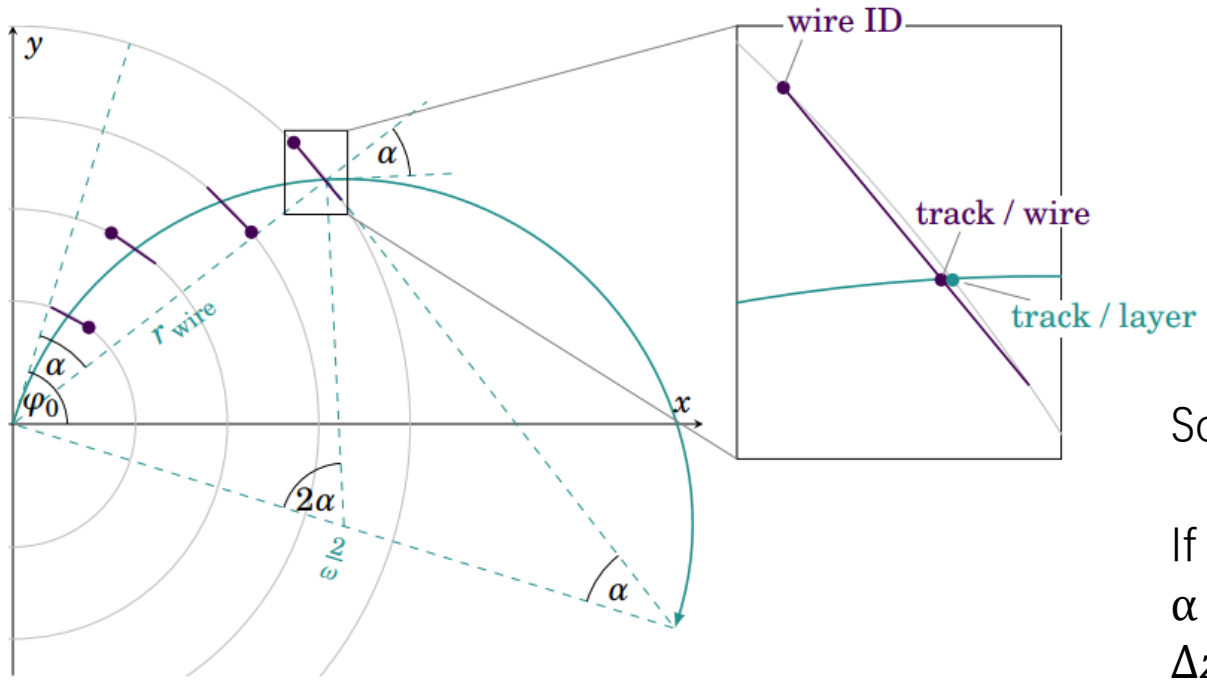
Due to increased bg z-cut now set at 10 cm for neuro tracks used in multi-track trigger (STT always at |z| < 15 cm)

-- from Christian Wessel

“Feed down” and “Feed up” at large z

Not so good for recently data with large Background

How to calculate out z0&z0 uncertainty



With direct cross stereo wire:

$$\phi_{cross} \sim \phi_0 - \arcsin\left(\frac{1}{2} r_{wire}\omega\right) \equiv \phi_0 - \alpha(r,\omega)$$

$$\frac{z_{cross} - z_B}{Z_F - Z_B} = \frac{\phi_{cross} - \phi_B}{\phi_F - \phi_B}$$

$$z_0 = z_{cross} - \cot\theta_0 \frac{2\alpha}{\omega}$$

Drift time would influence:

$$\phi_{hit} = \phi_{wire} \pm \arcsin\left(\frac{v_{drift} t_{drift} cos\alpha}{r_{wire}}\right)$$

$$r_{hit} = r_{wire} \pm v_{drift} t_{drift} sin\alpha$$

So the δt_{drift} would influence ϕ_{cross} and r_{wire}

If we ignore r_{wire} comparing with δt_{drift}, (with small α and large P_t)
Δz_0 could be consist of Δz_{cross} (from 3D Fitter /NN)
And $\Delta(\cot\theta_0 \frac{2}{\omega})$ (From 2D track)

And: $$\Delta z_{cross} = \frac{r_{wire}}{\sin\psi}\sqrt{(\Delta\phi_{cross})^2 + (\Delta\phi_B)^2}$$

Still, ignore r_{wire} comparing with δt_{drift},

$$\Delta\phi_{cross} \times \sim 0.03° - 0.08° \ (varied\ from\ r_{wire})$$

$$\Delta\phi_B \sim \frac{v_{drfit} cos\alpha}{r_{wire}} \Delta t_{drift}$$

MC Test

MC :

Train Sample

Particle gun:

muons; single tracks;

Pt :[0.3 GeV,3 GeV], uniform;

Φ: [0, 360],uniform;

θ: [0,170], uniform;

Vertex z0: [-50, 50], uniform;

N events: 300k

Validation Sample:

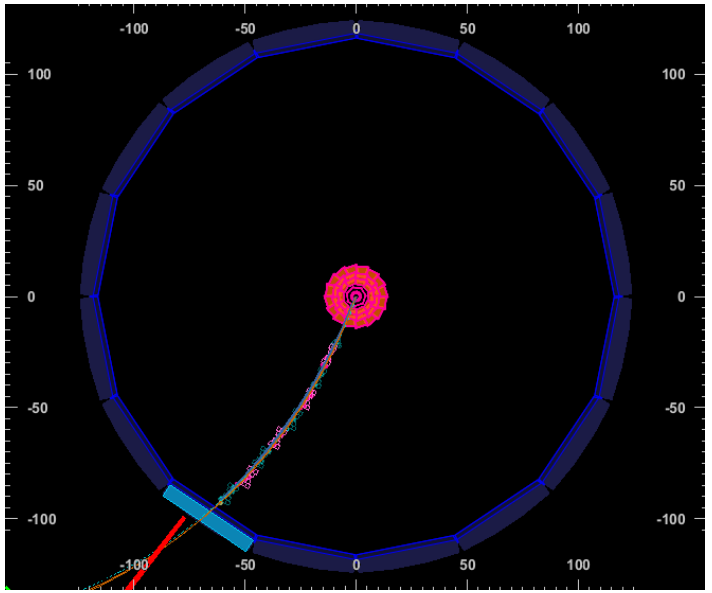
Same config;

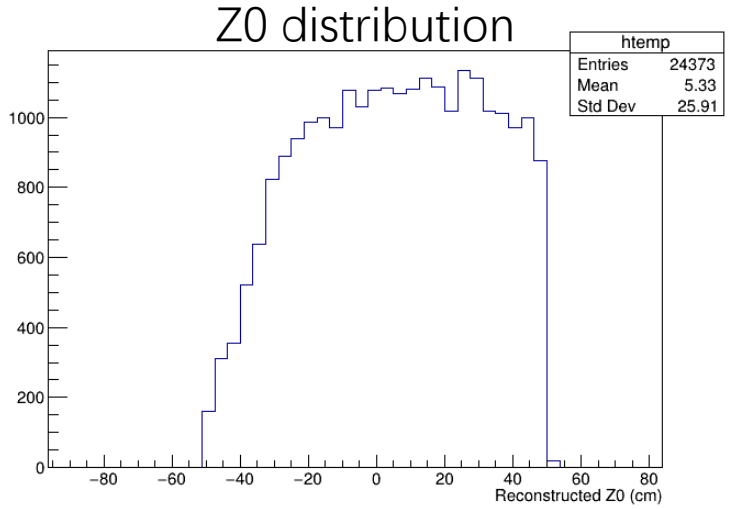
N events: 20k

Test Sample:

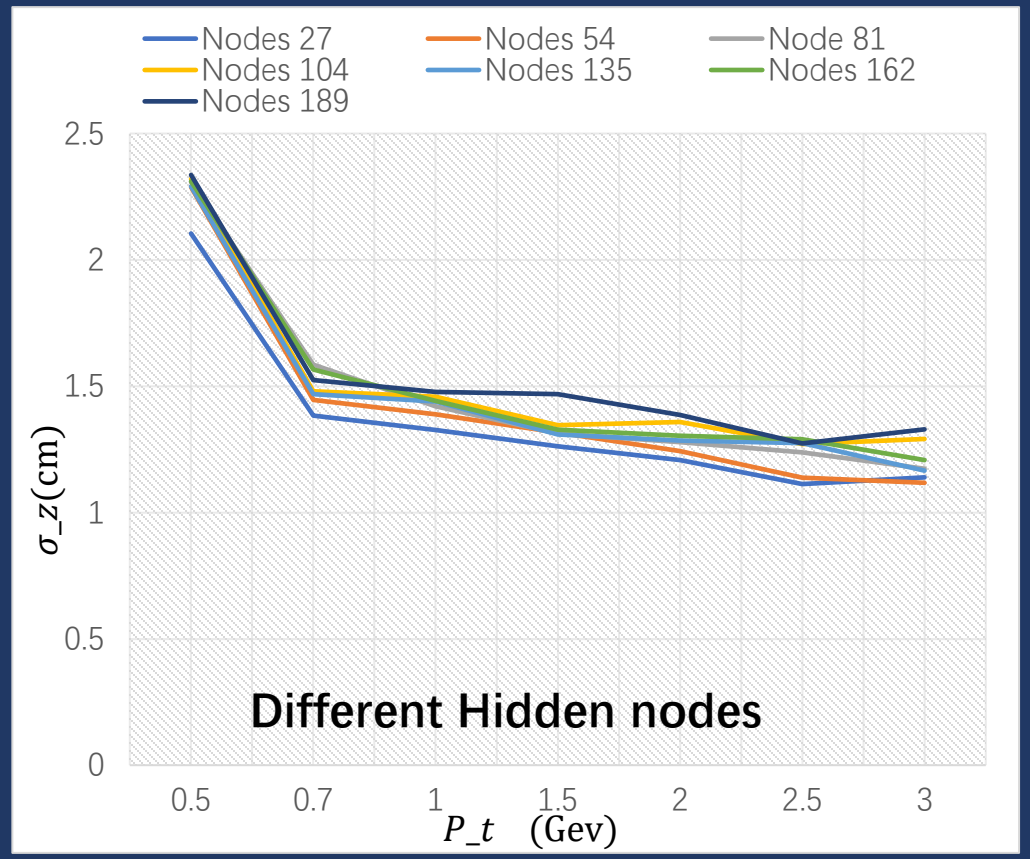
Same config;

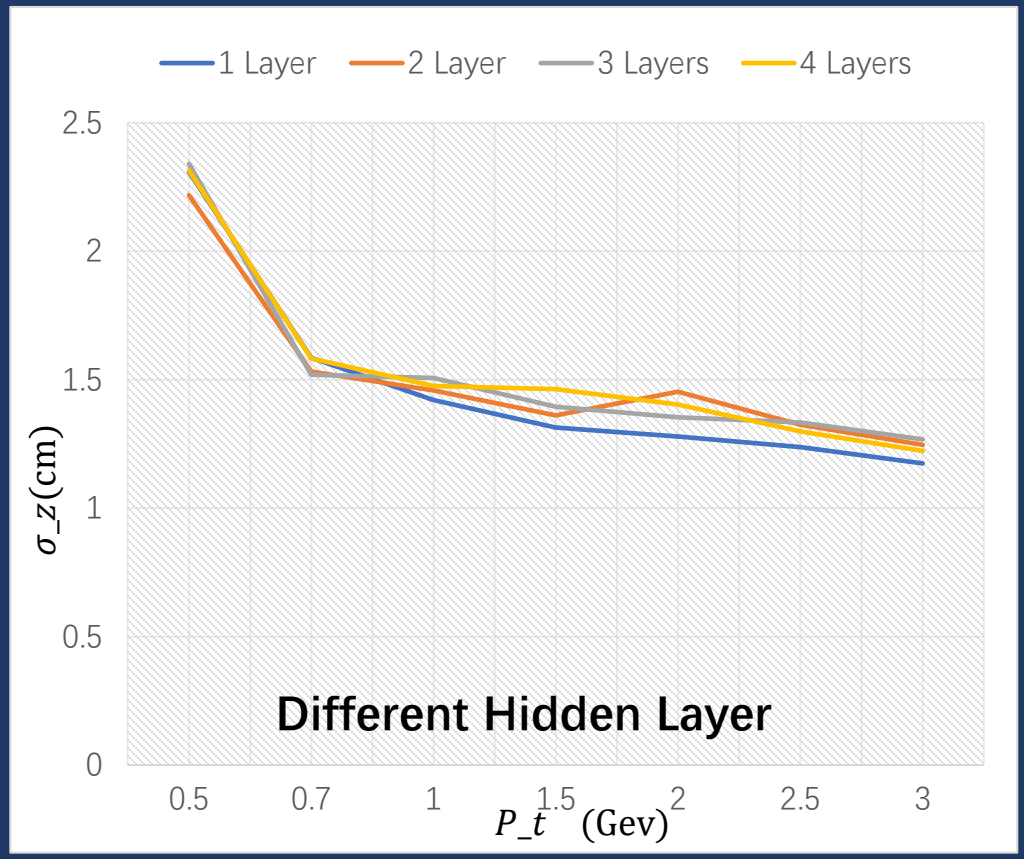
N events: 50k





Hidden Layer



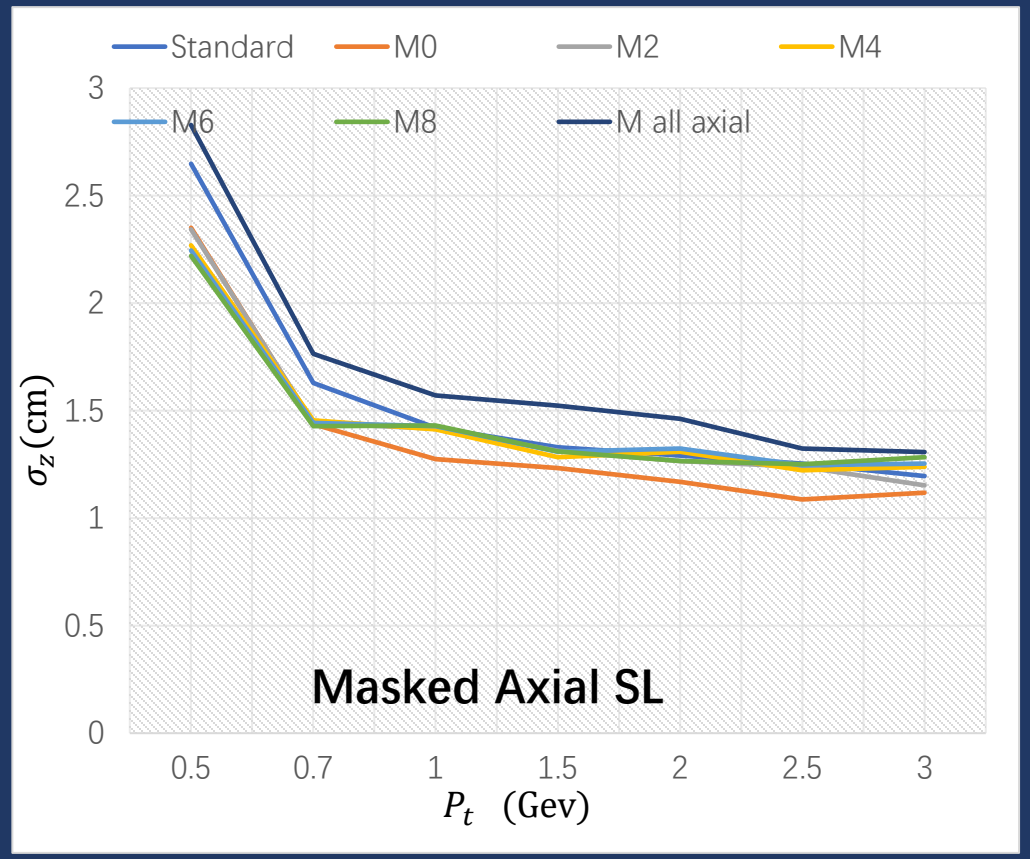


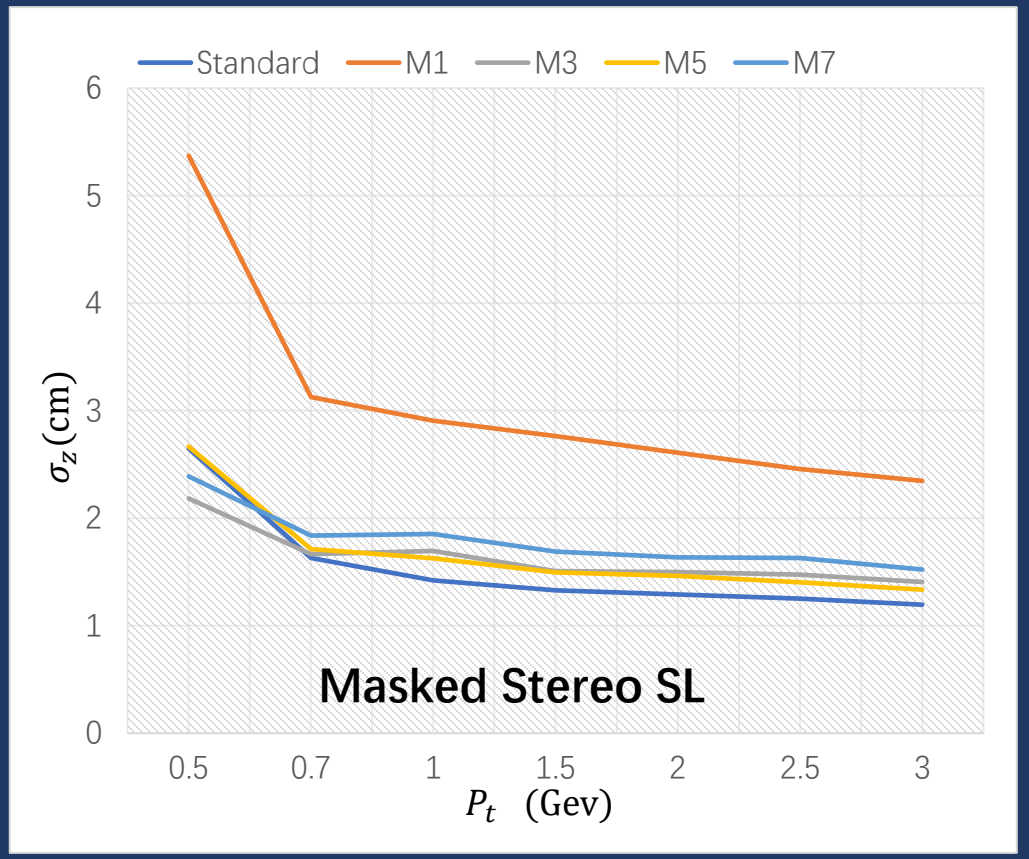
Trained with MC, event t0 =0.

Different hidden Layers / nodes do not make large difference in standard model

Add more wires do not induce other relationship, keep hidden layer as before.

Masked Super Layer

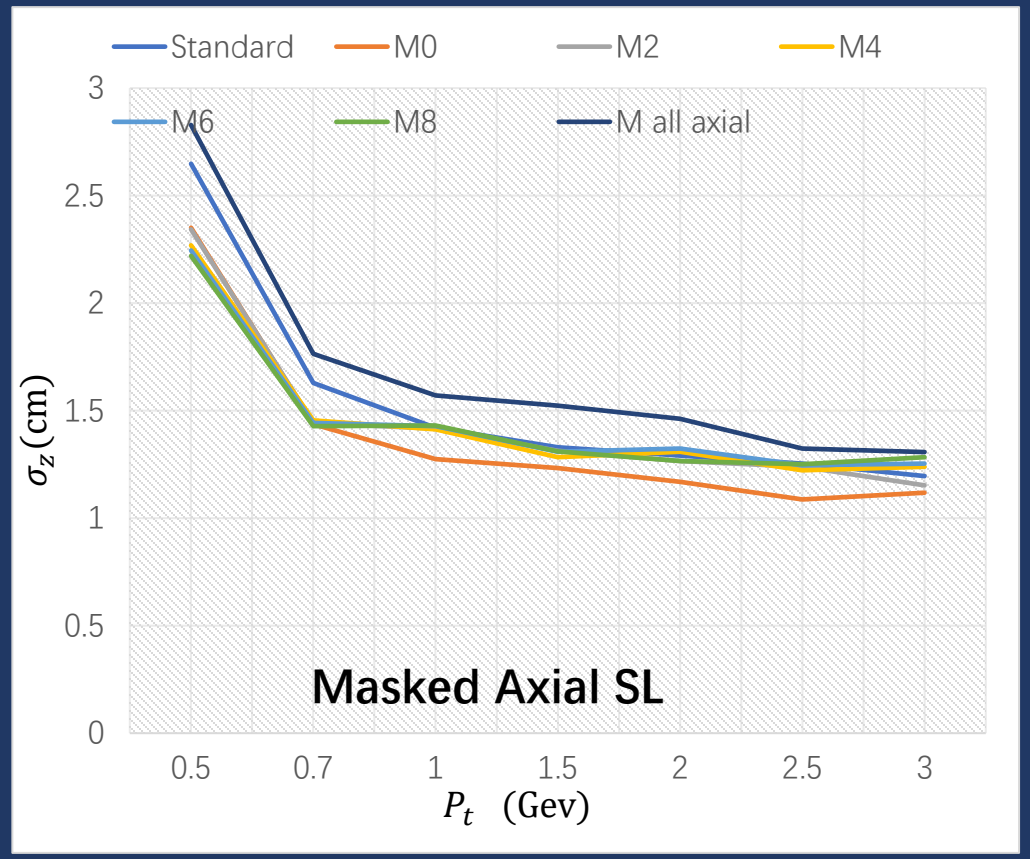


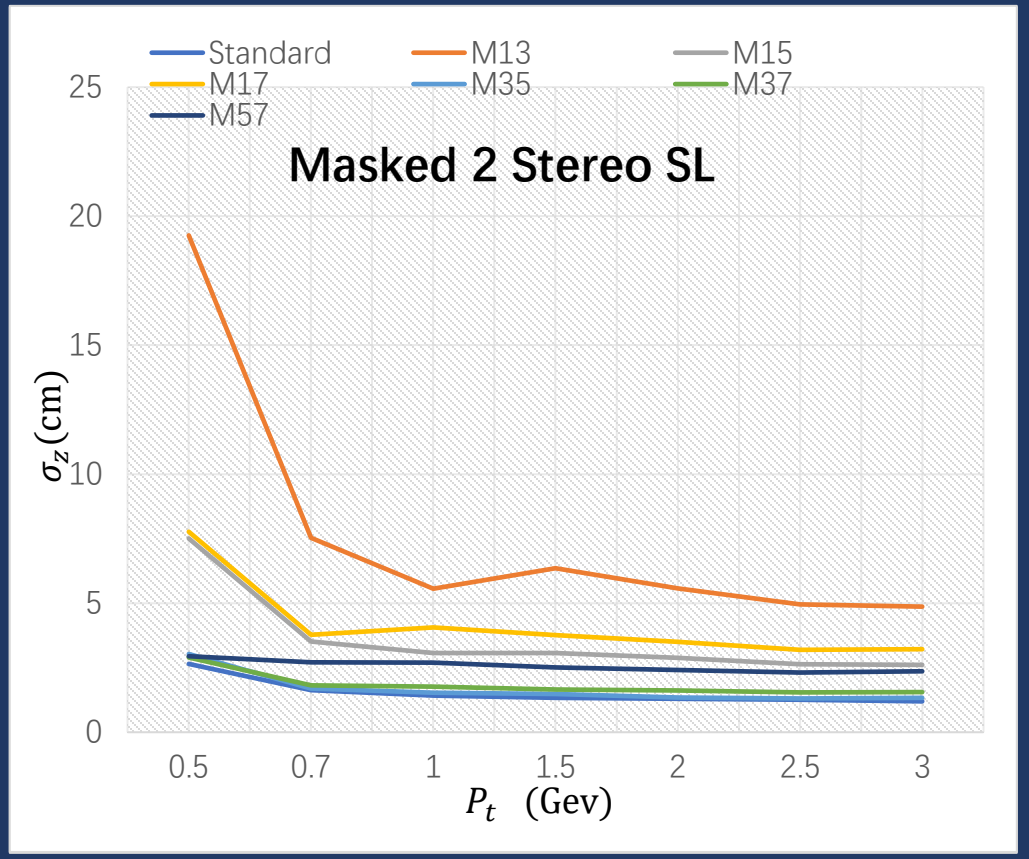


To see the importance of each super layer, masked each one for the training & testing for NN.

Axial layer contribute little to the NN, even masked all, resolution decrease little

Masked Super Layer

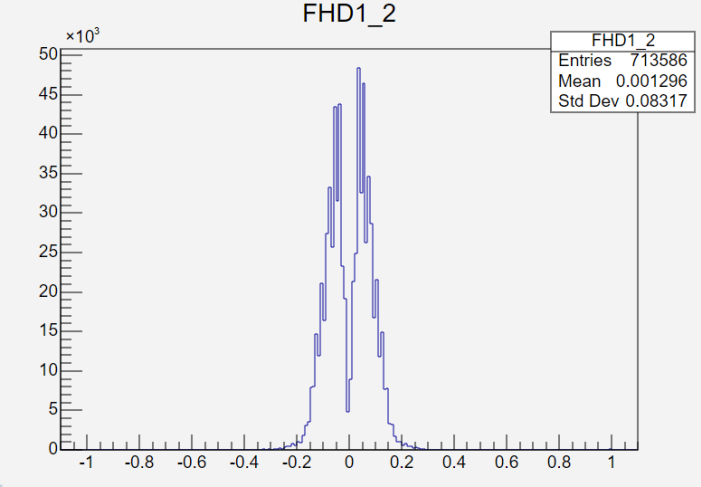




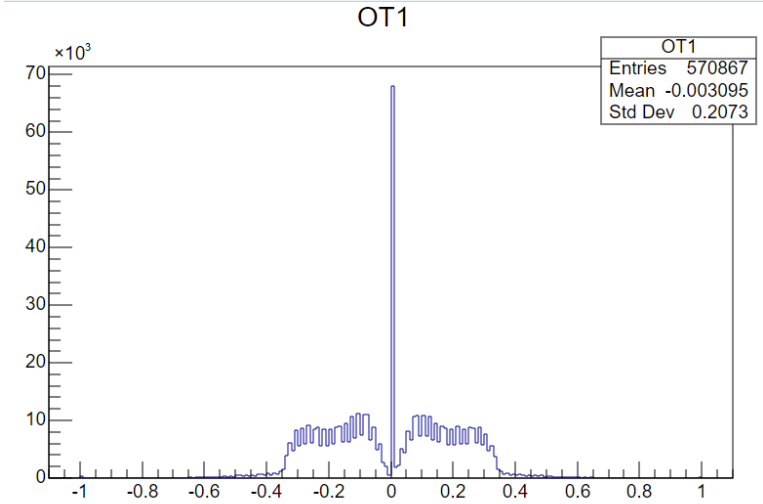
To see the importance of each super layer, masked each one for the training & testing for NN.

Axial layer contribute little to the NN, even masked all, resolution decrease little

Input Parameters

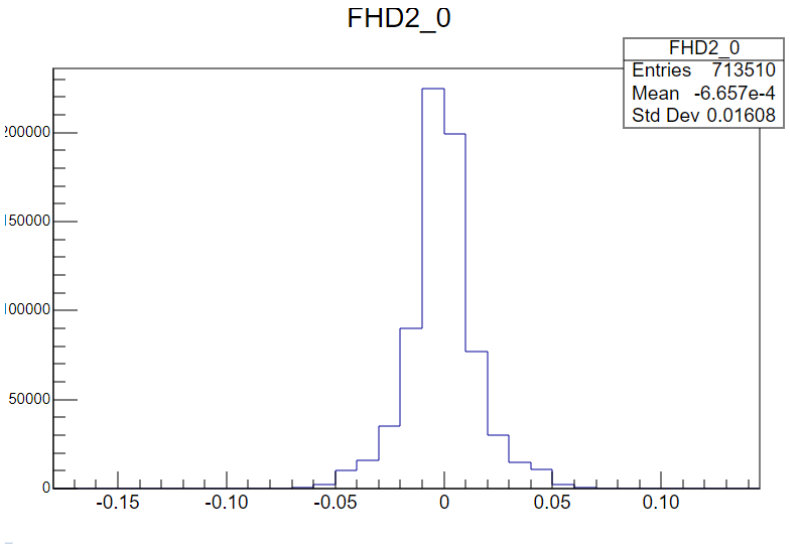


Extra T

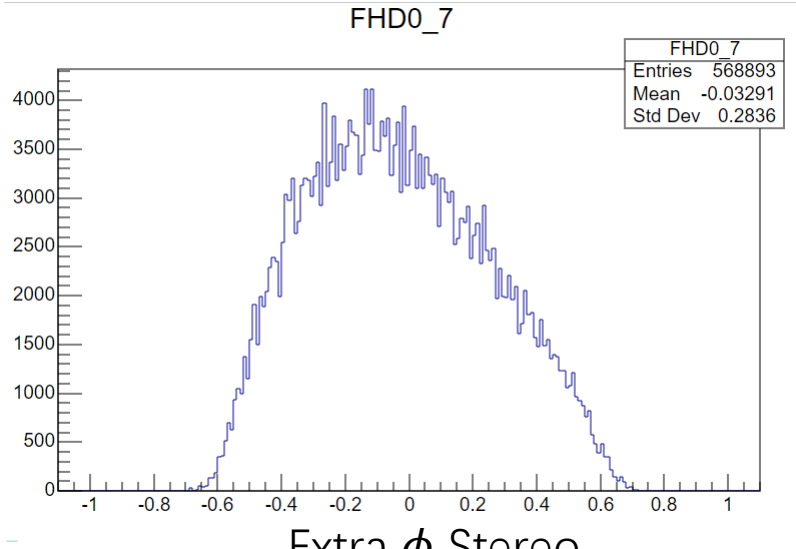


Origin T

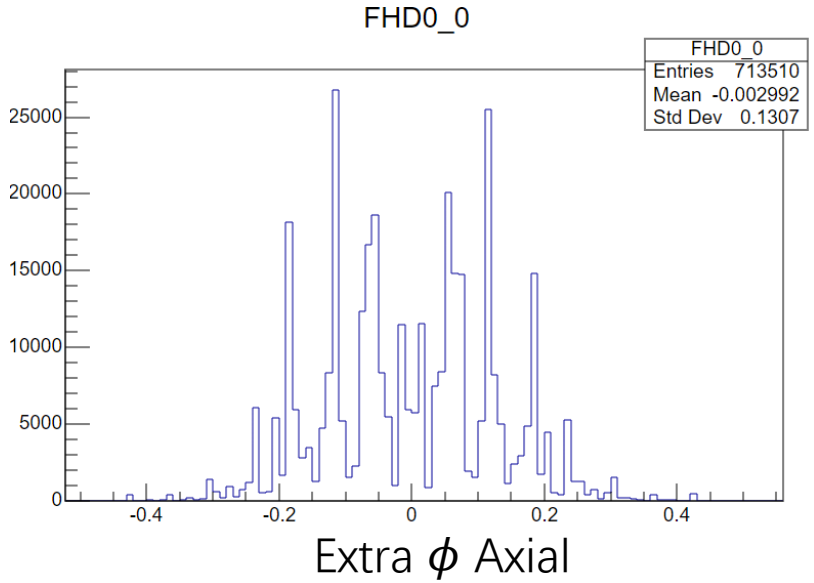
For exp26run1771



Extra α



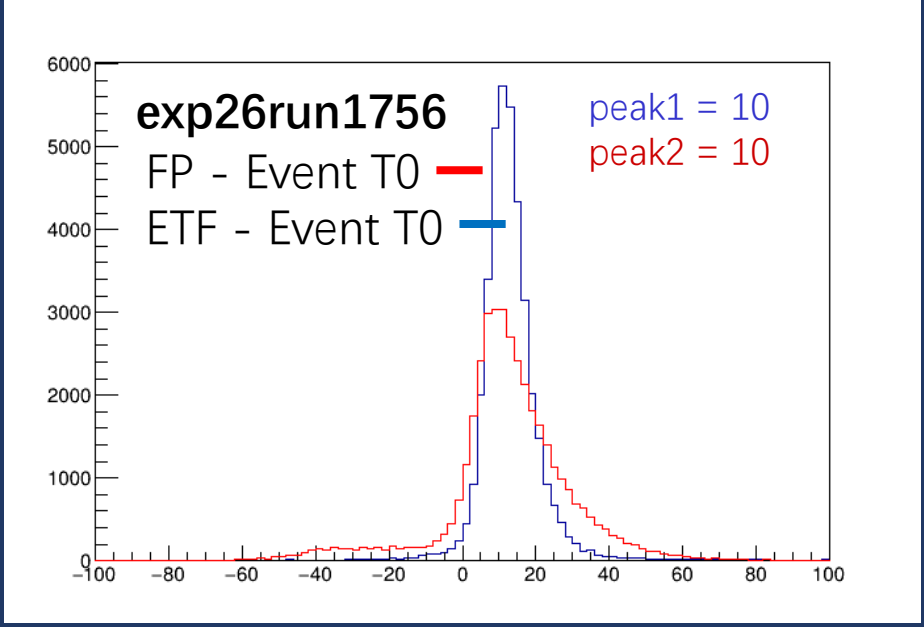
Extra ϕ Stereo



Extra ϕ Axial

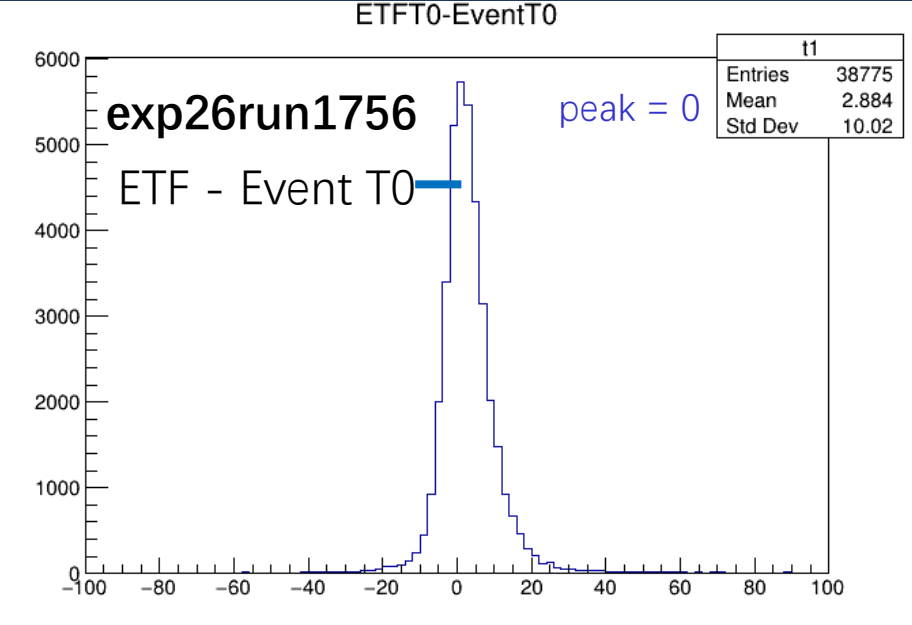
ETF-offset

```
addParam("offset", m_offset,
         "Set certain time offset for ETFHough simulation"
         "Default as 0", 0);
```



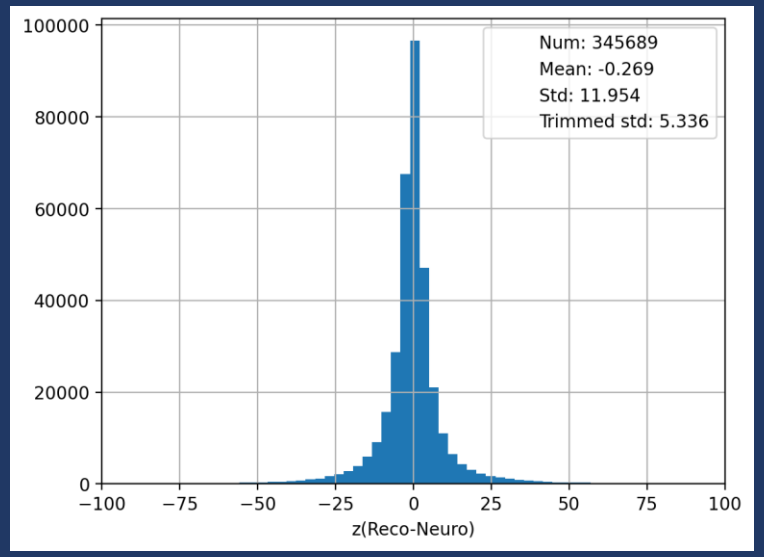
After

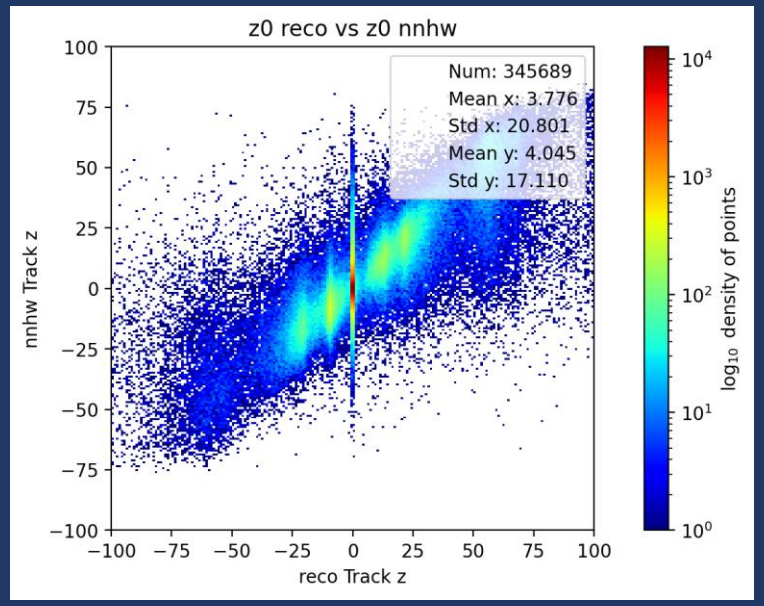
Offset = -10 ns



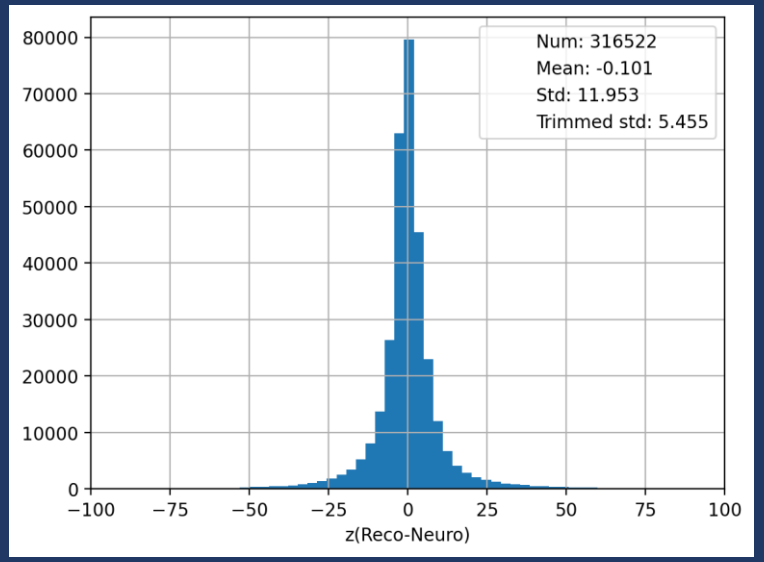
After ETF offset 1 wires expert 0

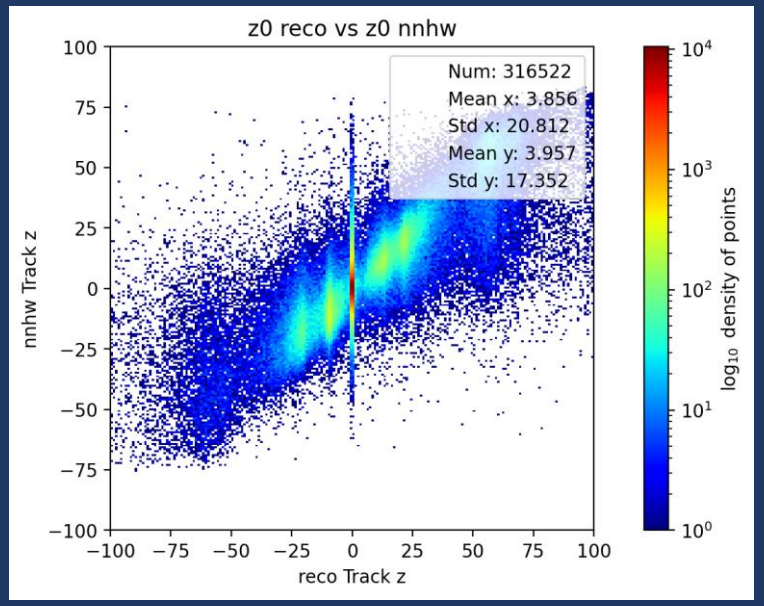
Before offset





After offset





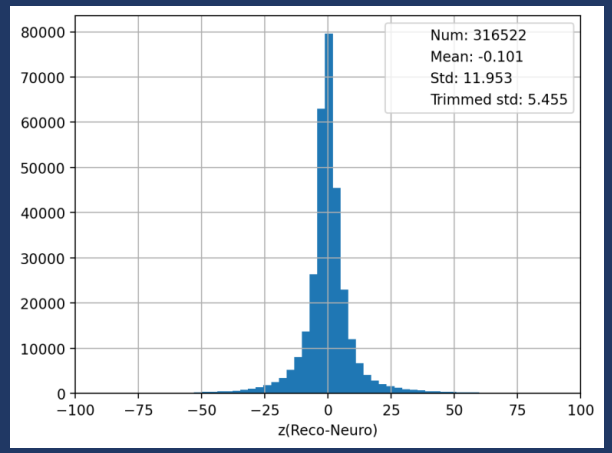
No difference
--As expected: NN could learn the offset

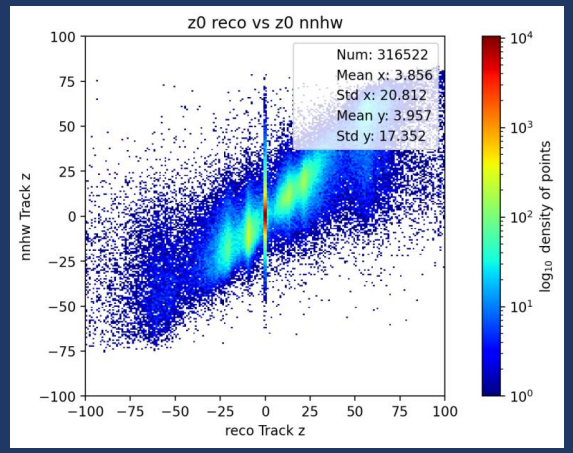
Step learning rate?

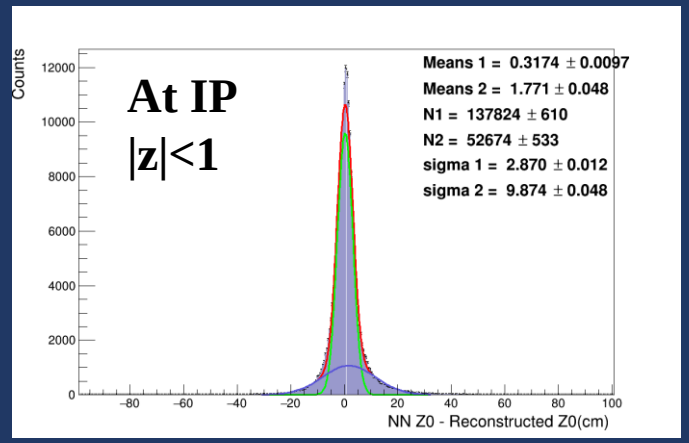
Question: Training error will start to oscillate after a few hundreds epoch → try to adjust learning rate to improve it more deeply.

First attempt: learning rate * 0.2 at every 200 epoch

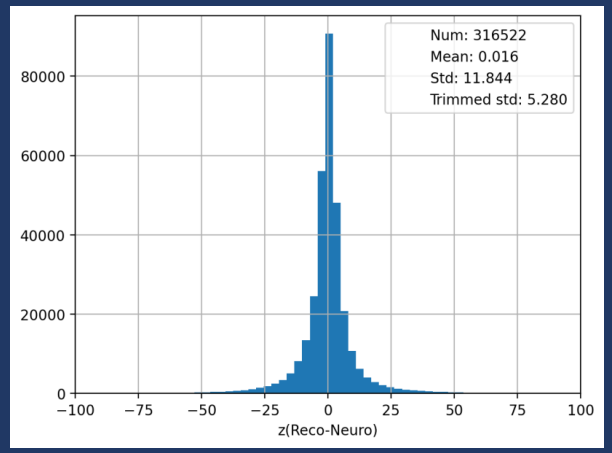
before

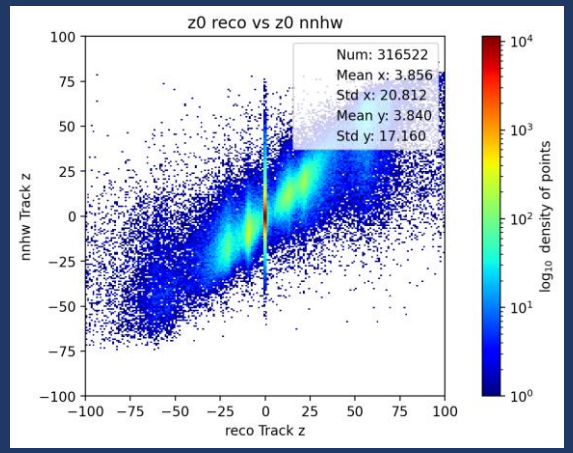


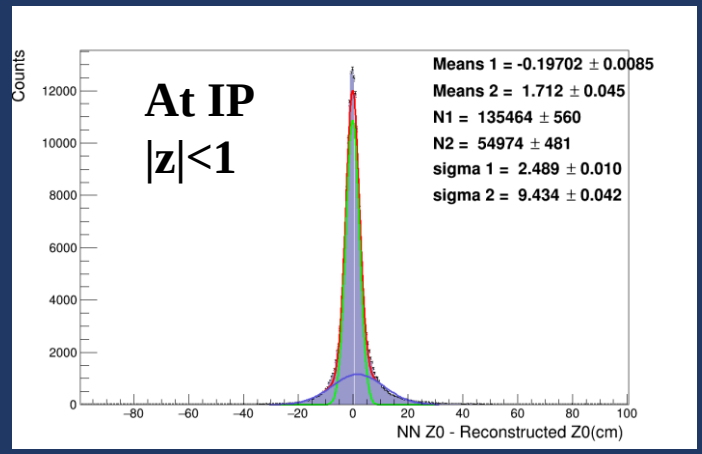




After

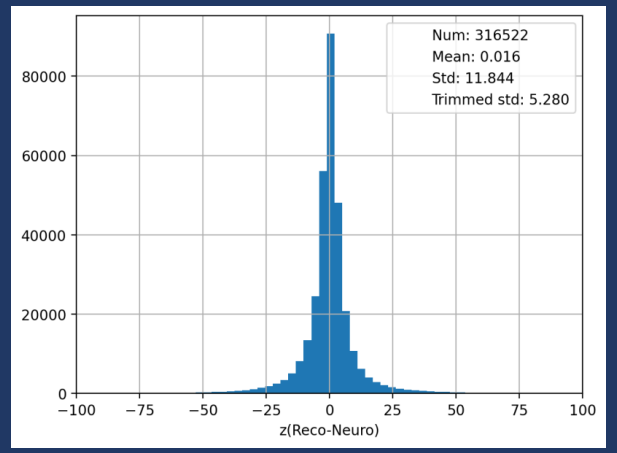


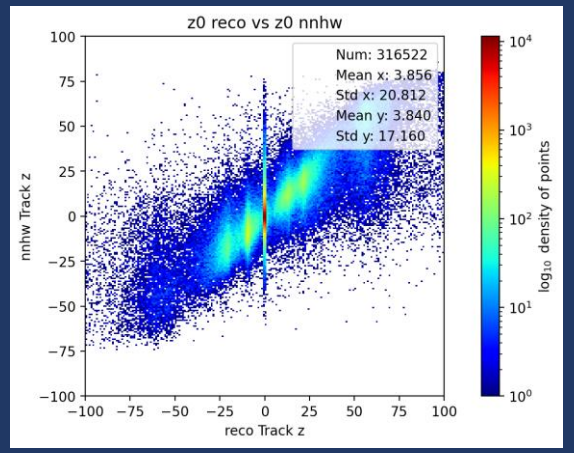


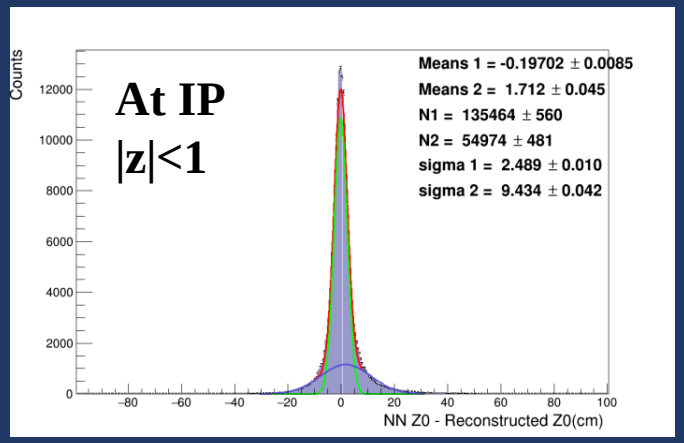


More wires?

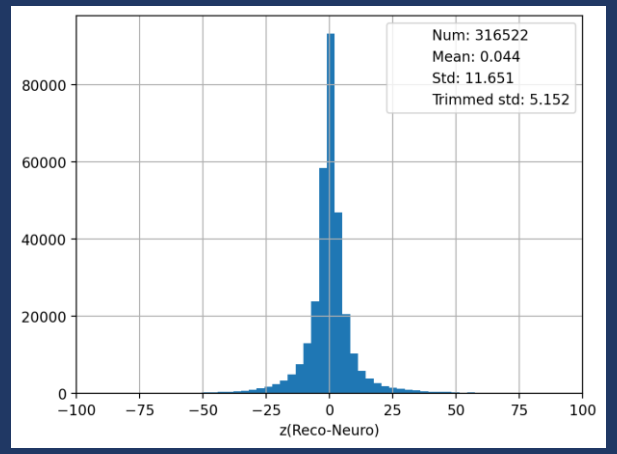
Wire 1

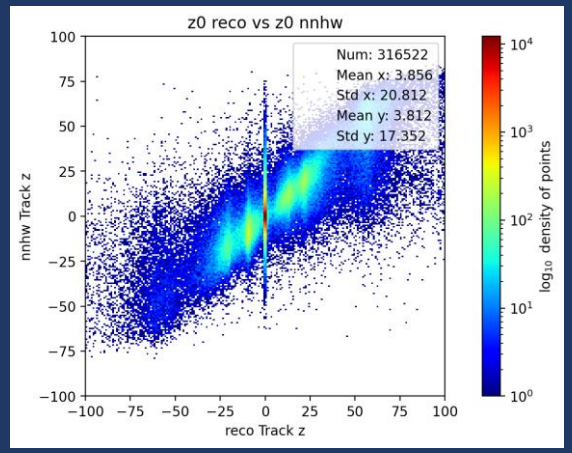


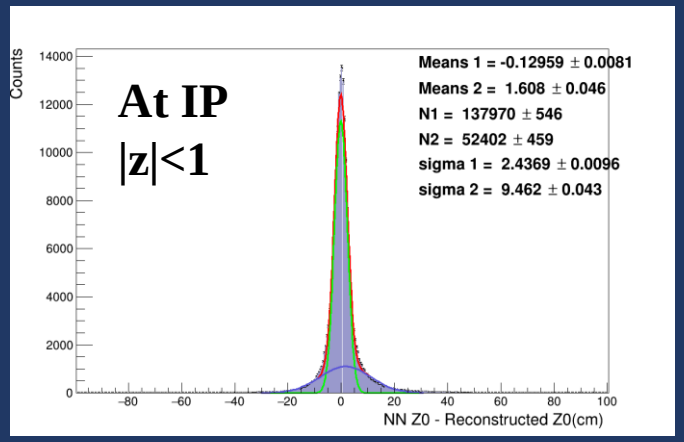




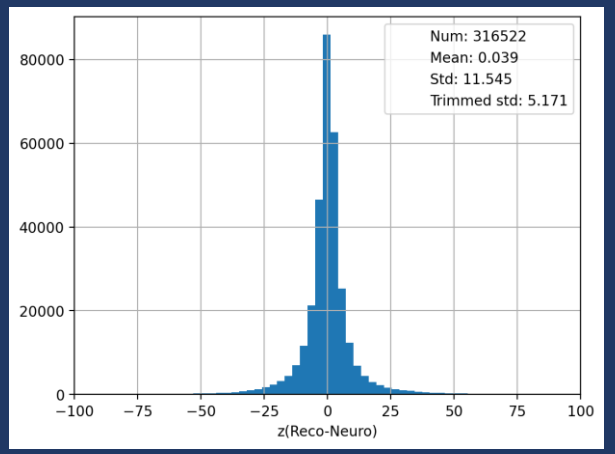
Wire 2

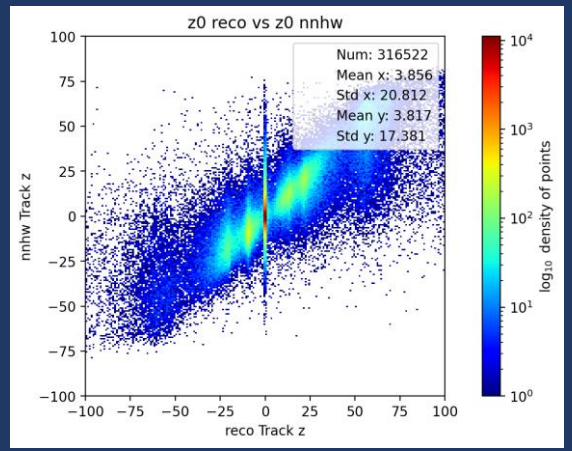


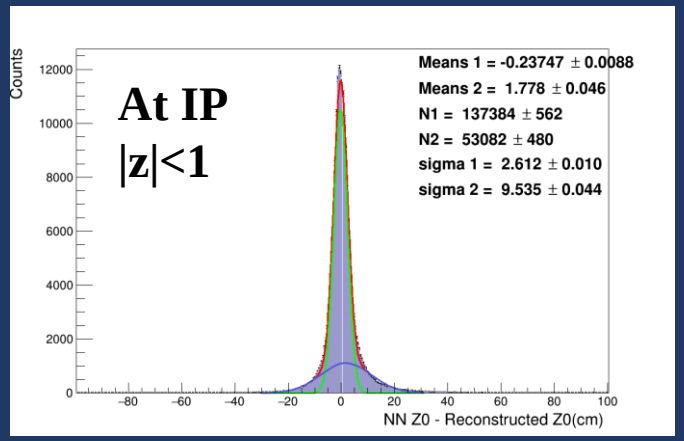




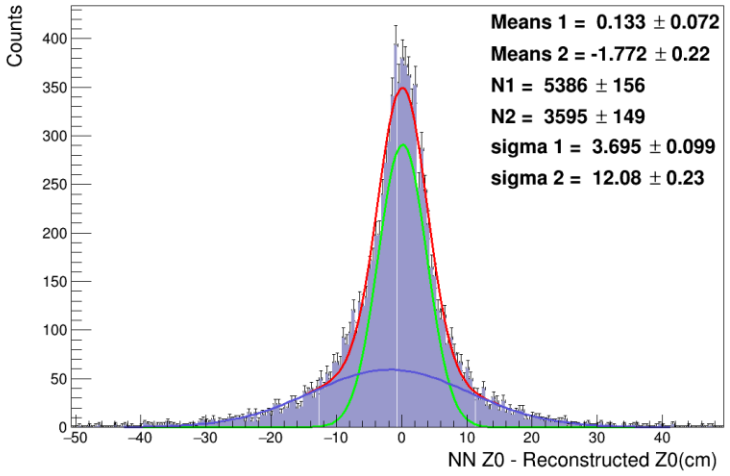
Wire 3

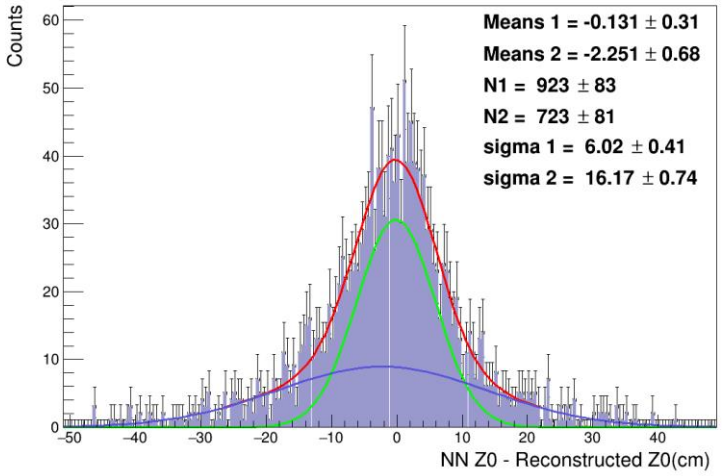


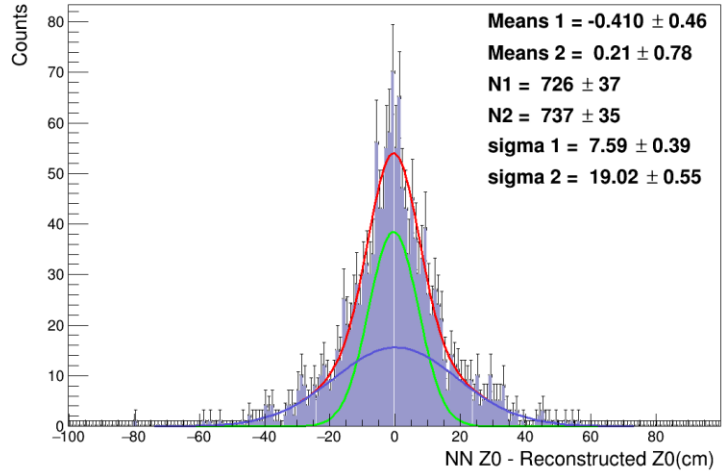




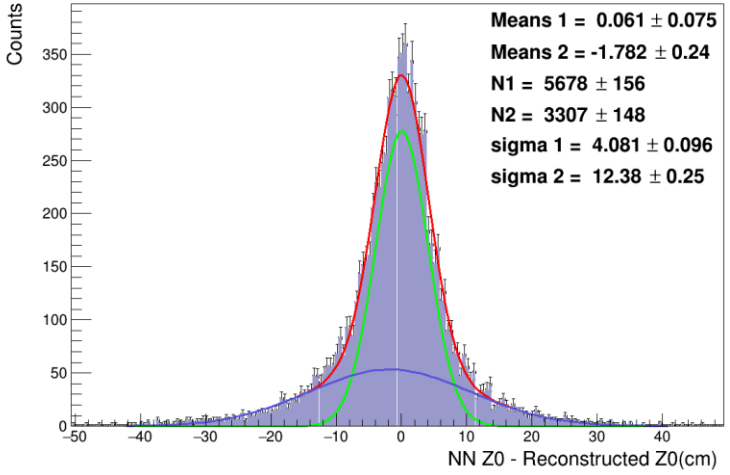
Detail result compare with z0

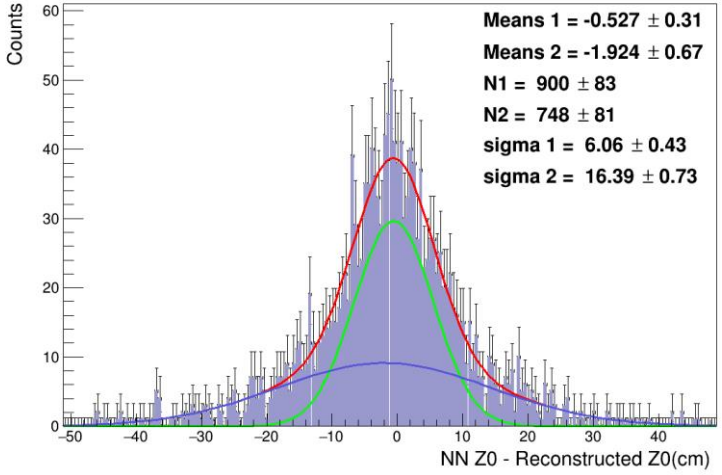


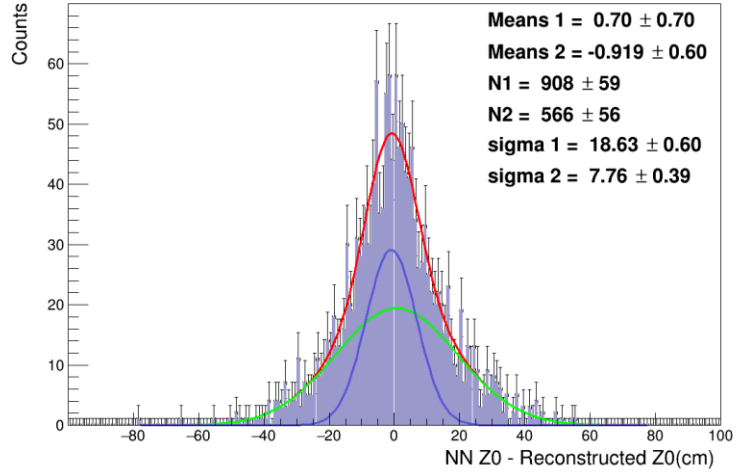




Wire1







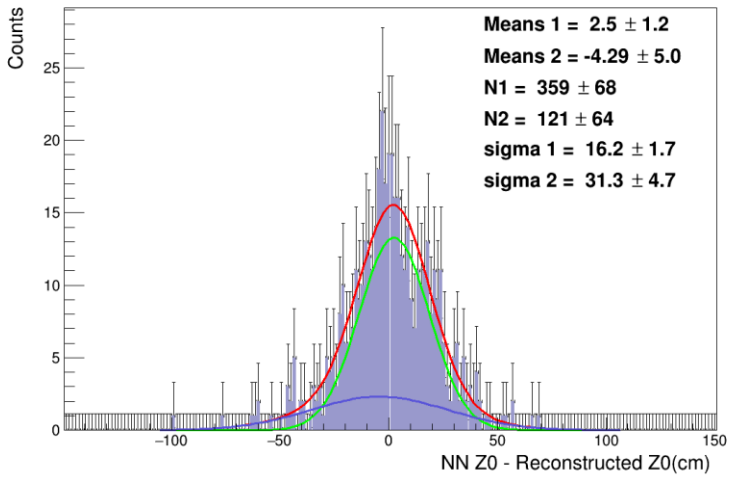
Standard

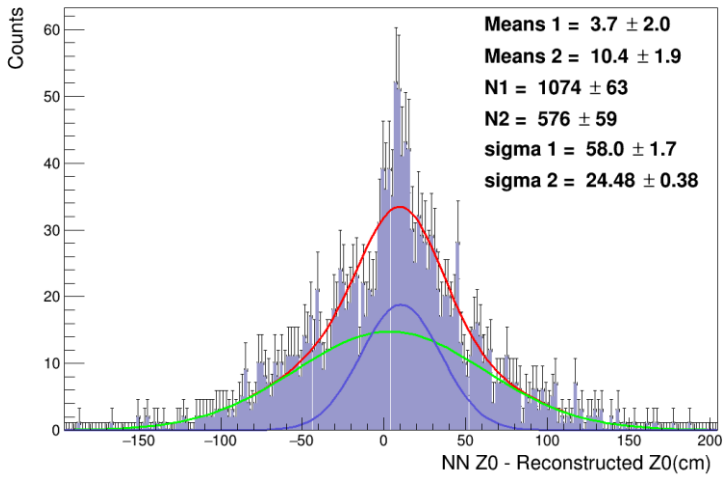
0-10

10-20

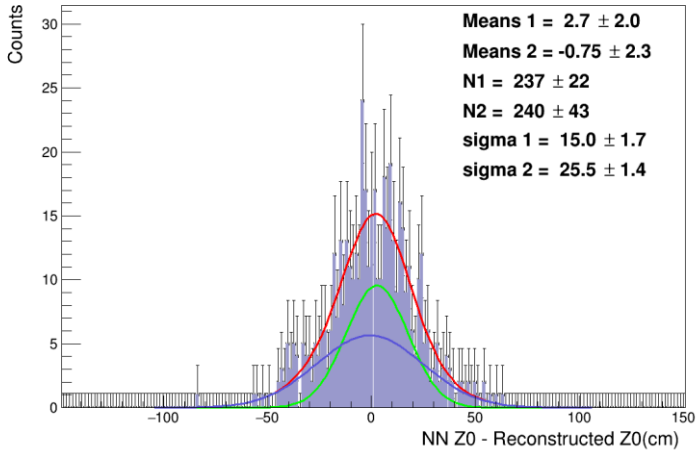
20-30

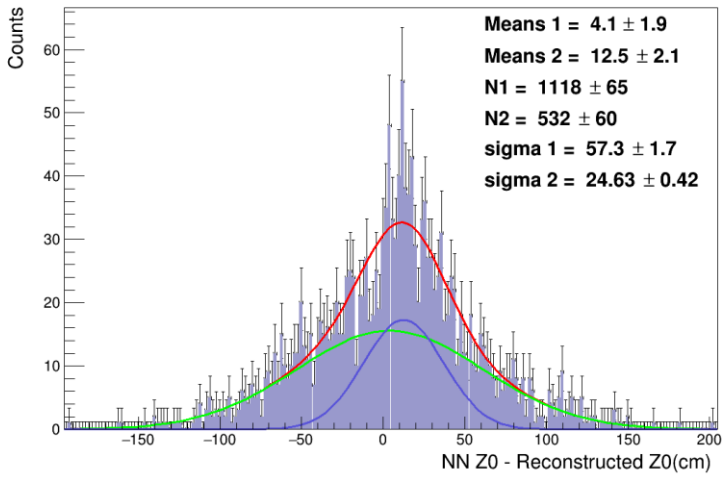
Detail result compare with z0





Wire1





Standard

30-40

40+

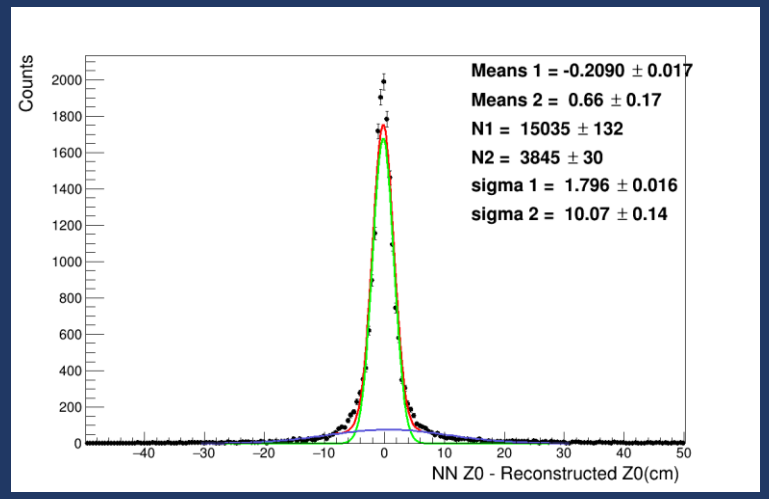
Real data test –Train with exp24 run 2004

Train NN with exp24 run2004 and exp26r1968

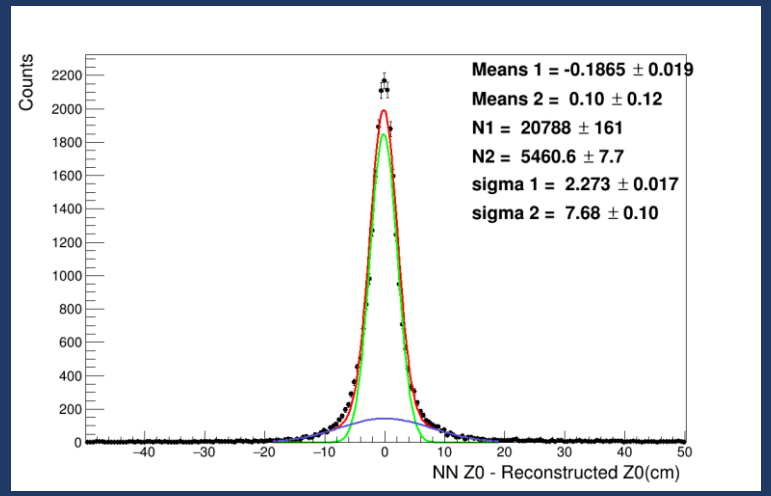
Test with exp24 run2004(sorry not unpacked another one for test)
And exp26 run 1777 (which use for trigger study with z0 in any range)

exp24 run2004

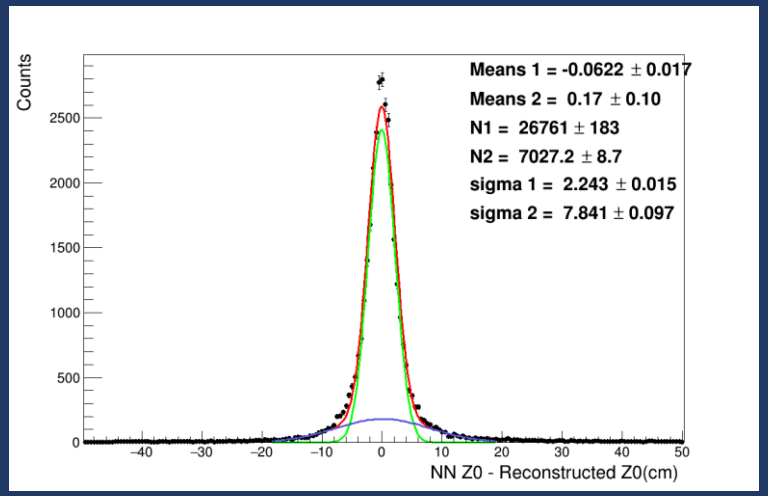
Origin with ETFHough



Wire 1 with ETFHough



Wire 2 with ETFHough



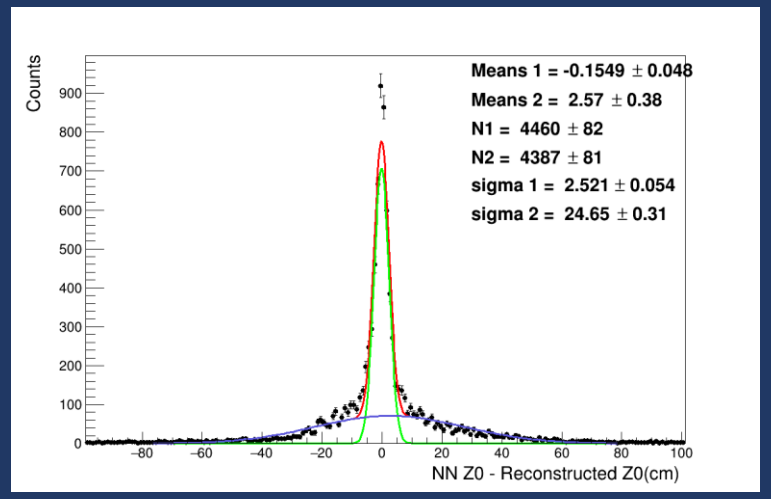
Real data test –Train with exp24 run 2004

Train NN with exp24 run2004 and exp26r1968

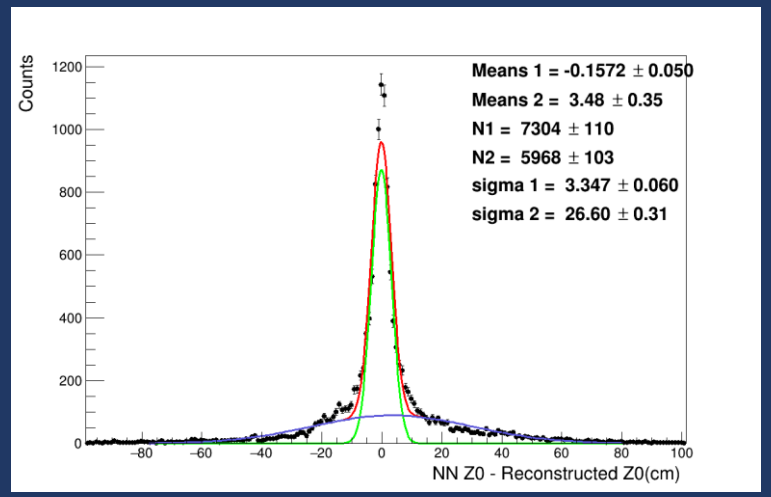
Test with exp24 run2004(sorry not unpacked another one for test)
And exp26 run 1777 (which use for trigger study with z0 in any range)

exp26 run 1777

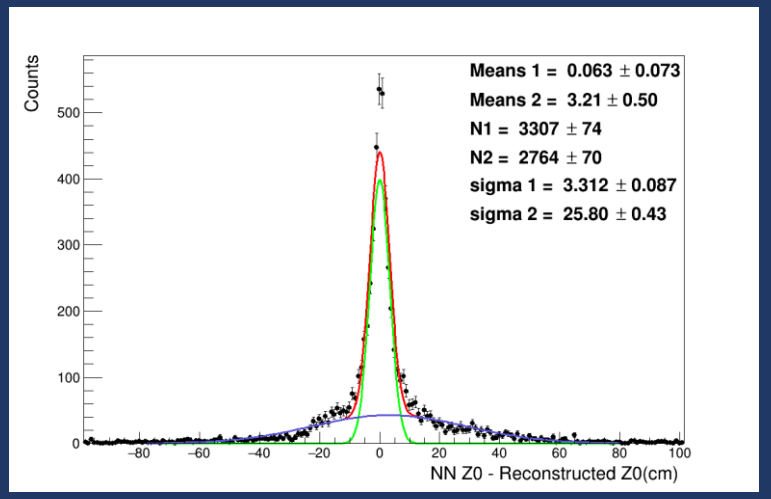
Origin with ETFHough



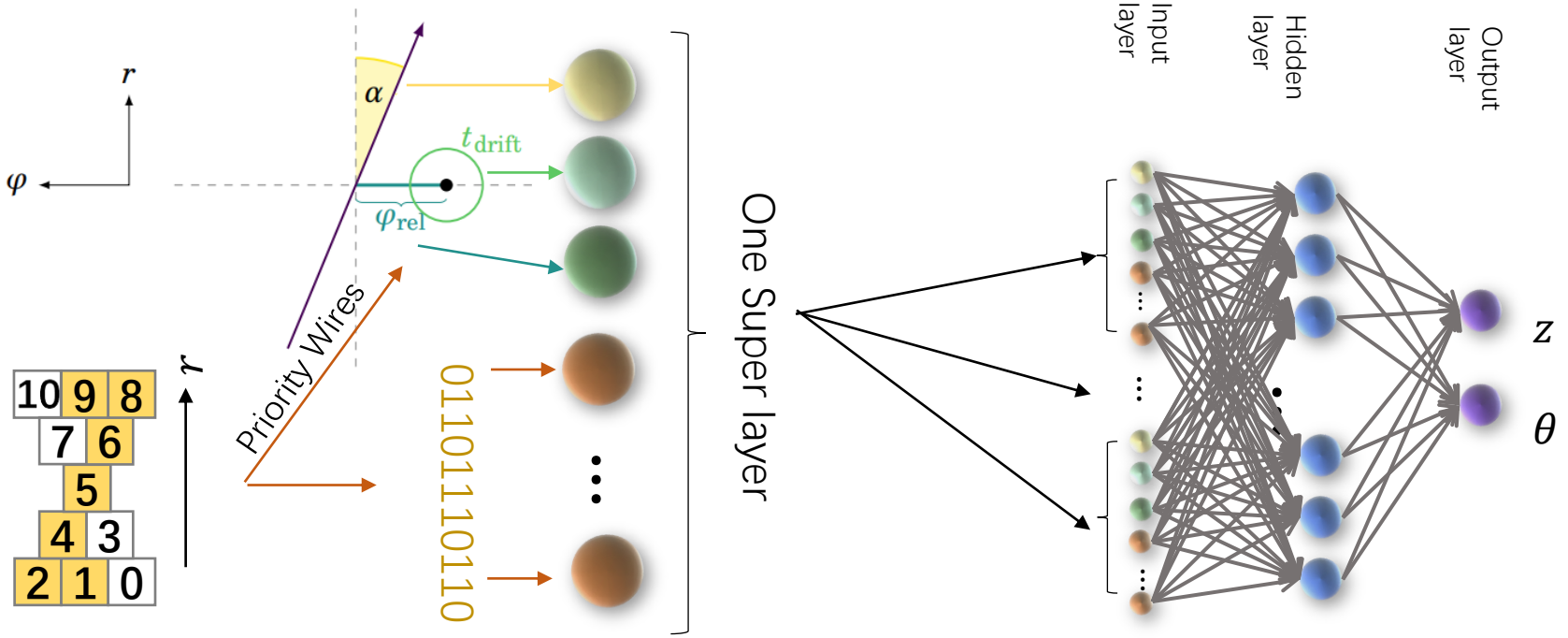
Wire 1 with ETFHough



Wire 2 with ETFHough



First attempt: Directly use TS pattern as input



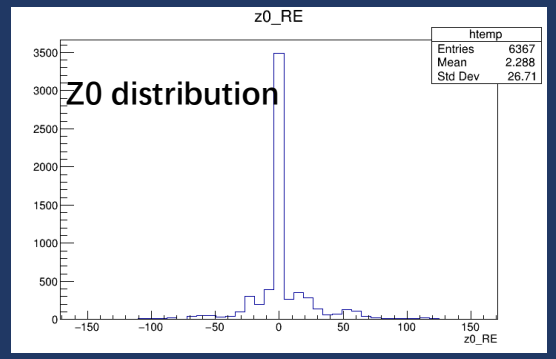
Directly use 11/15 bits pattern as input

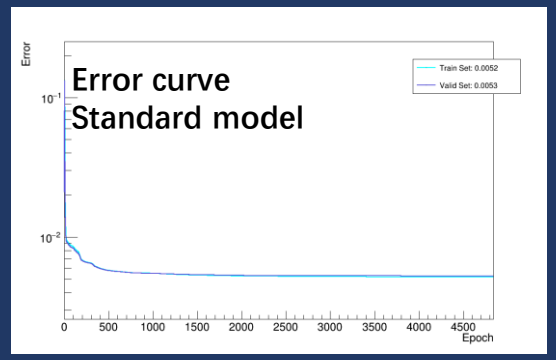
Since L/R information are got from pattern, hoping could replace L/R with it

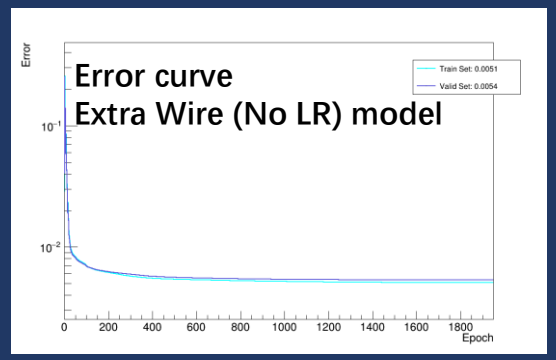
Train with real data

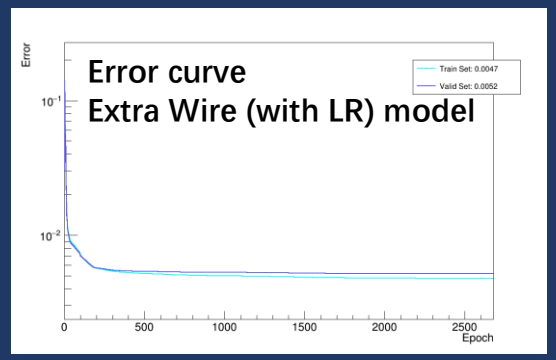
Train Standard model and one extra wire model (with/without LR) with exp26 run1771 & exp26 run1762; beam-reco-monitor.

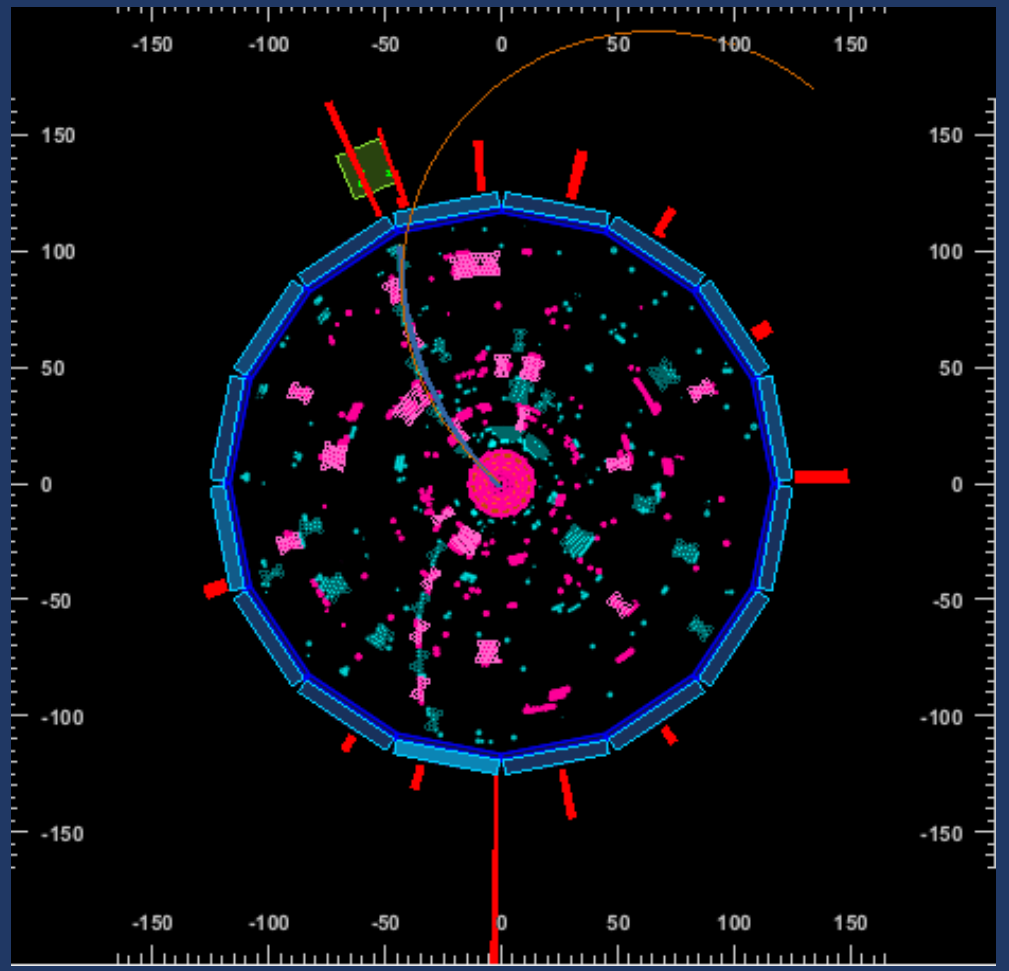
ETF option :fastpriority





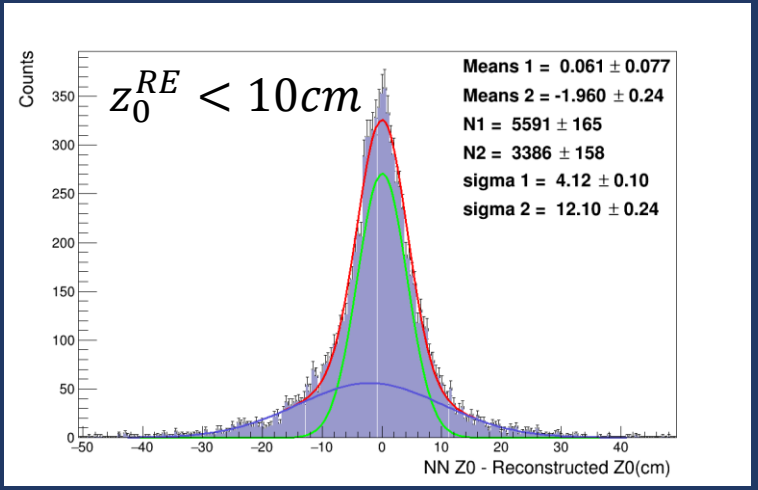


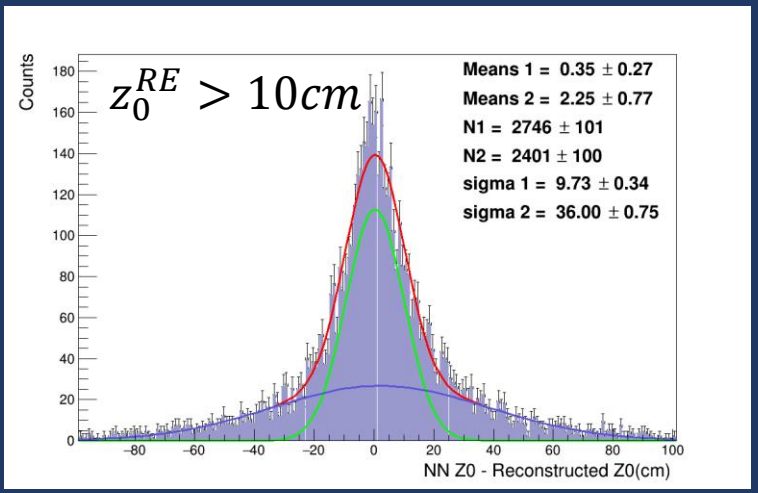




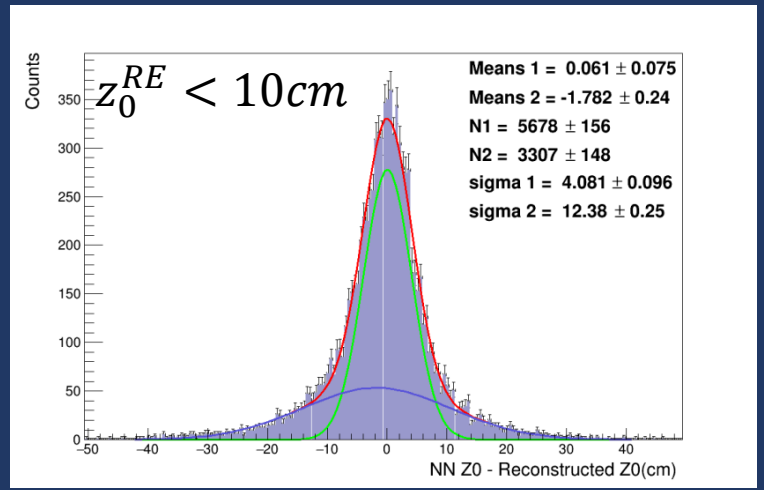
Test with real data exp26 run1771

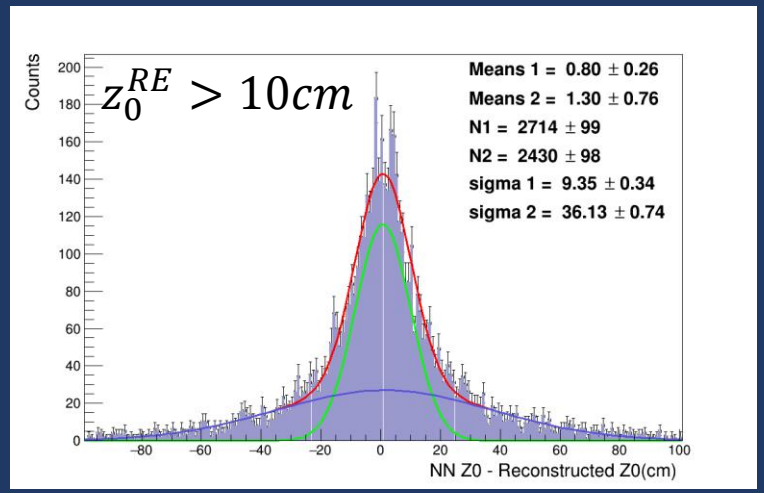
Standard Model



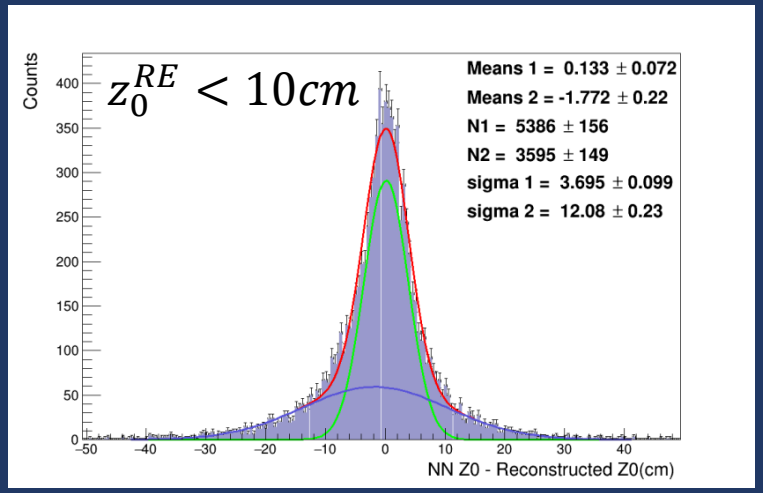


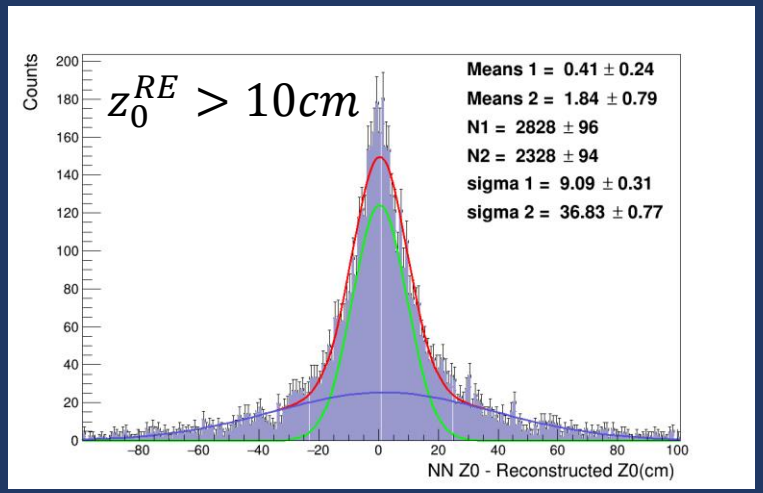
Extra Wire 1 No L/R





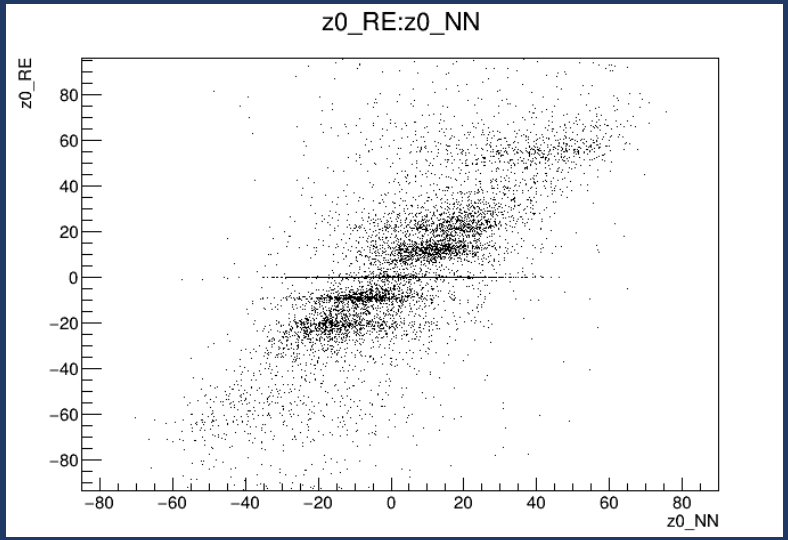
Extra Wire 1 with L/R



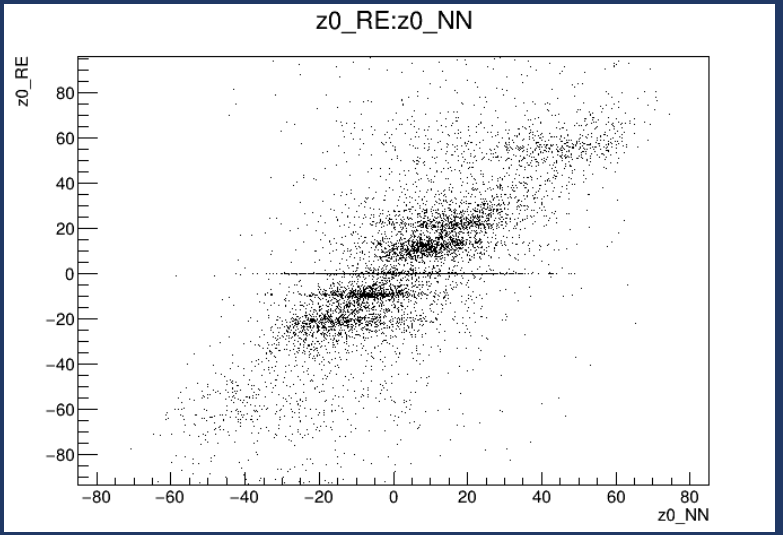


Test with real data exp26 run1771

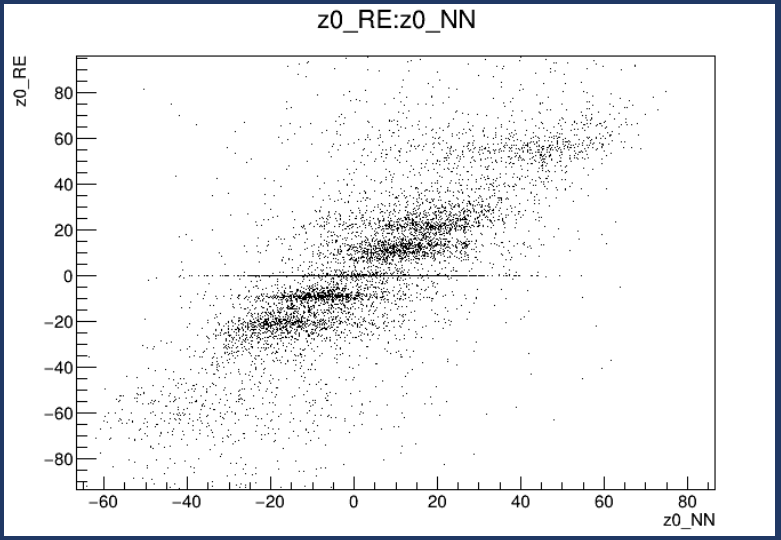
Standard Model



Extra Wire 1 No L/R



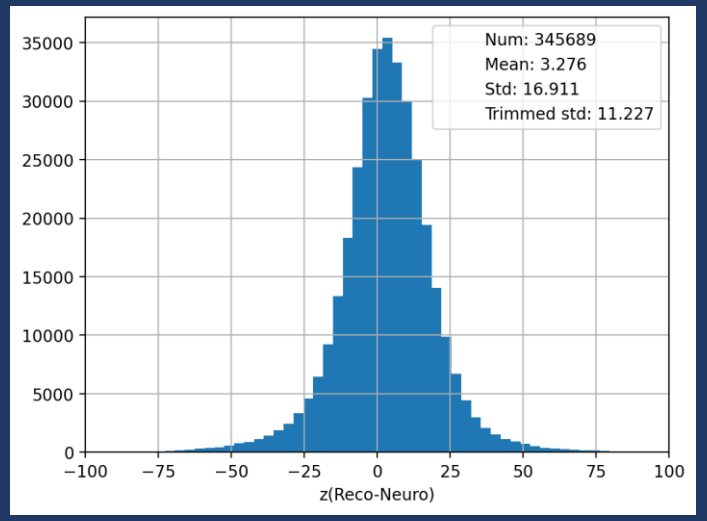
Extra Wire 1 with L/R

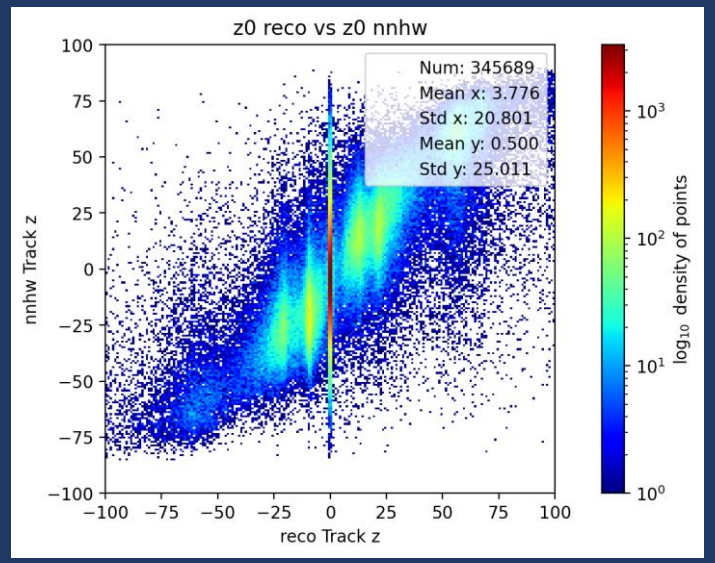


Training still need to be improved.

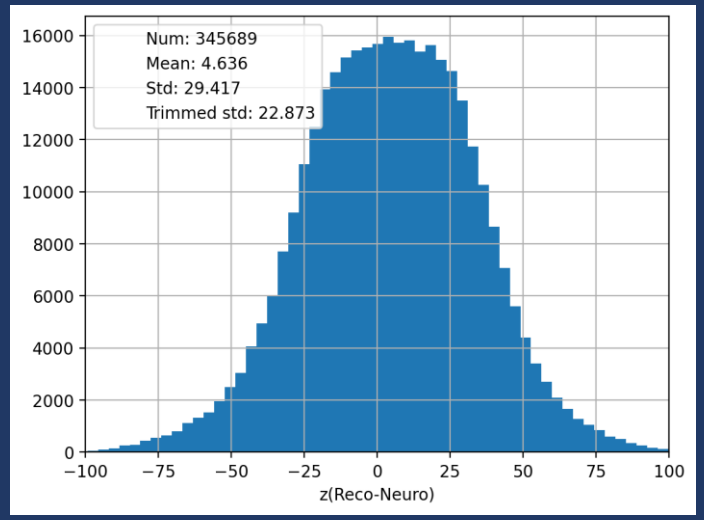
Pytorch training result -Uniform / norm distribution

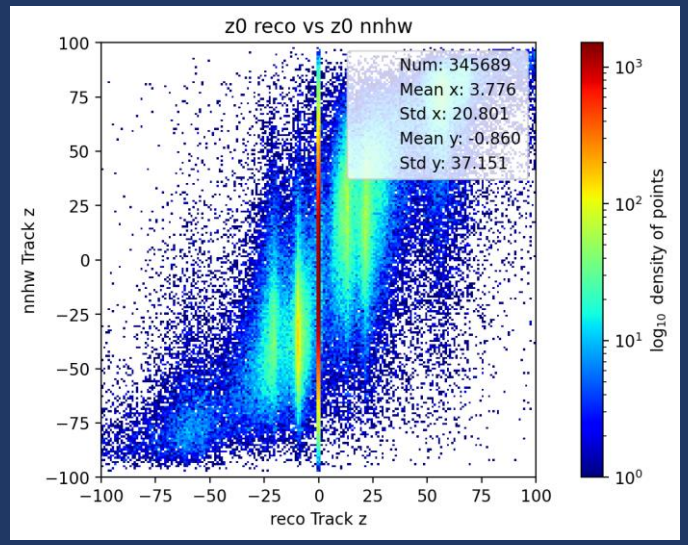
Norm



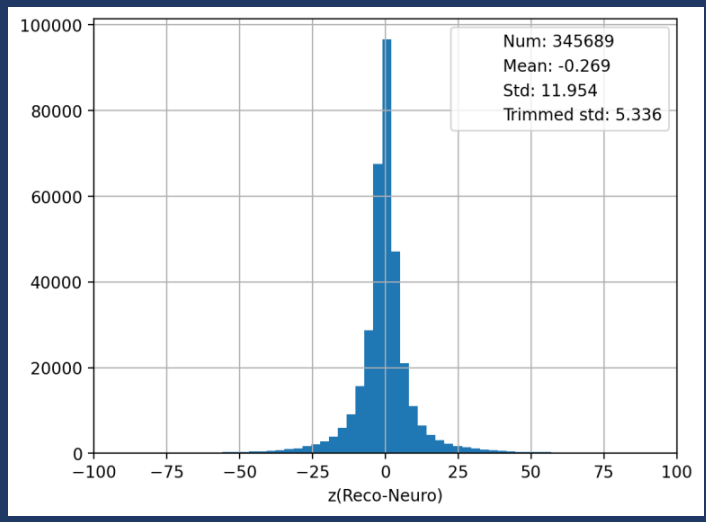


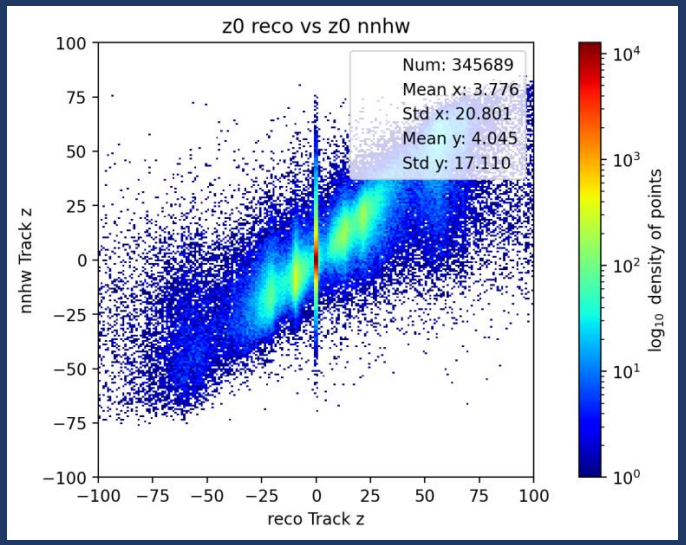
Uniform



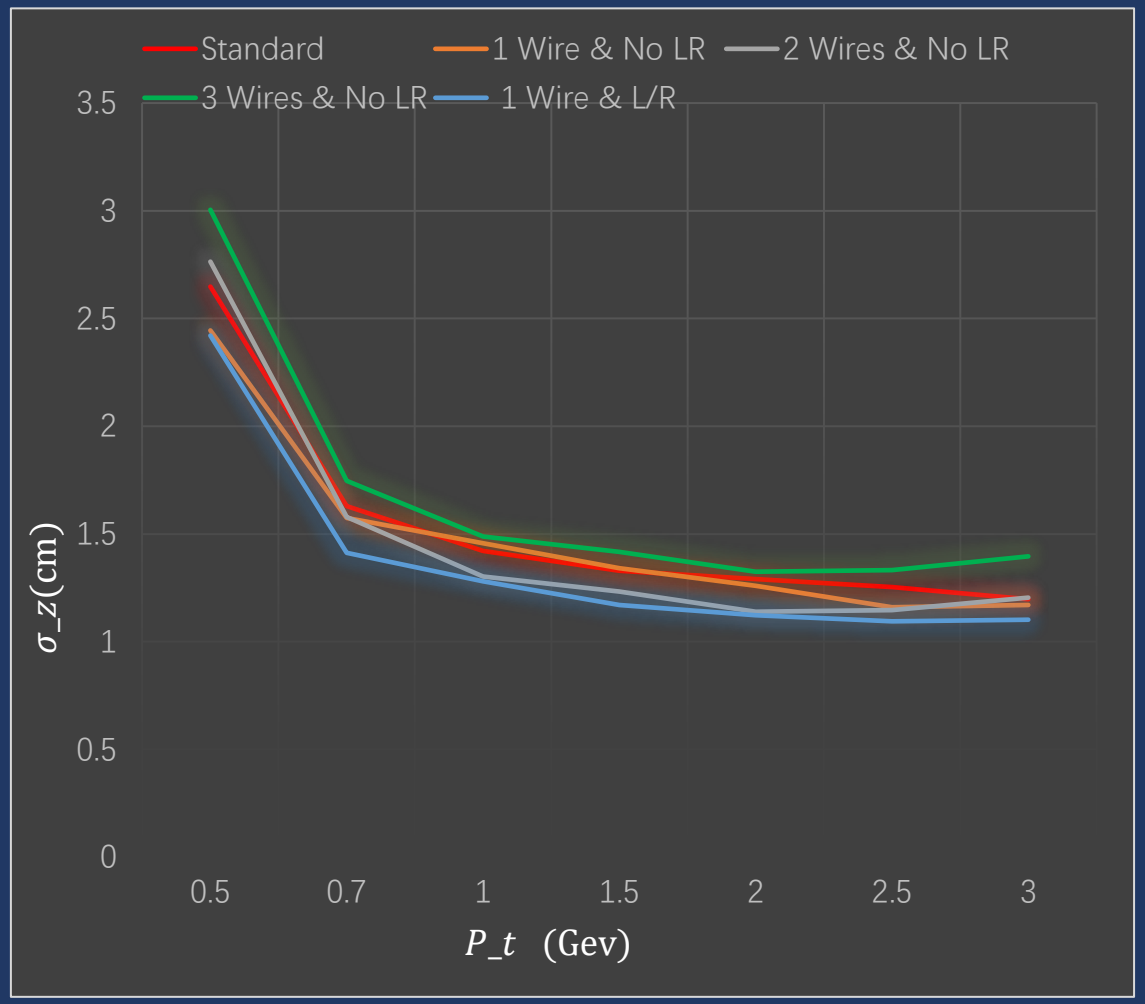


ExtraWires





Extra wire as input

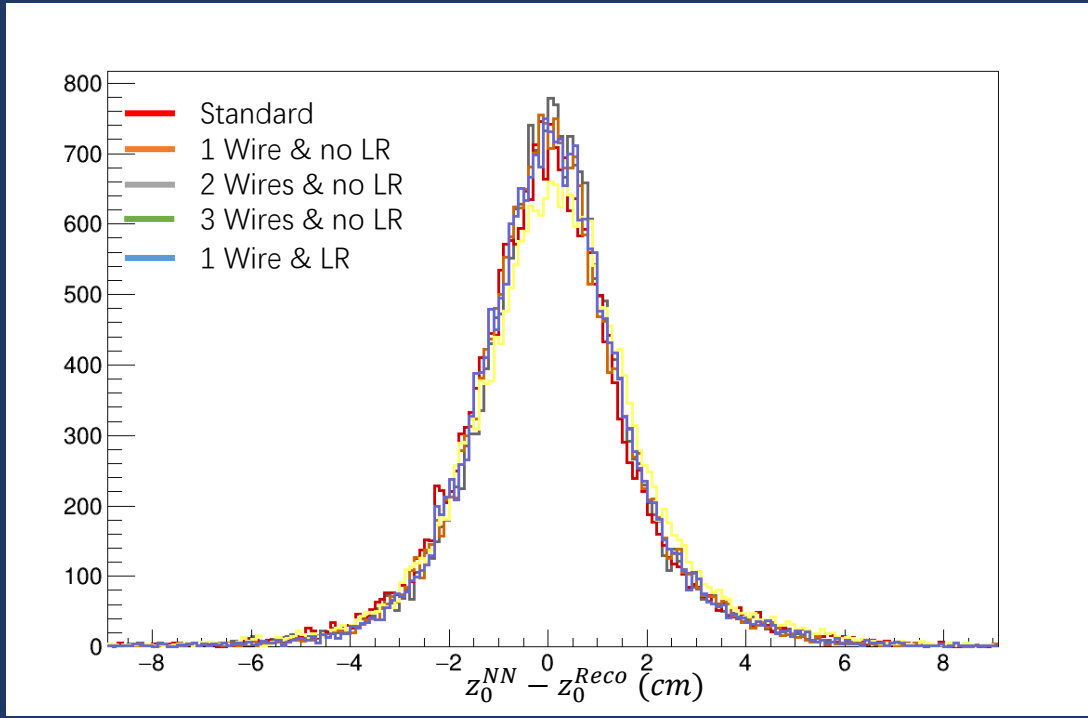


Trained NN based on MC

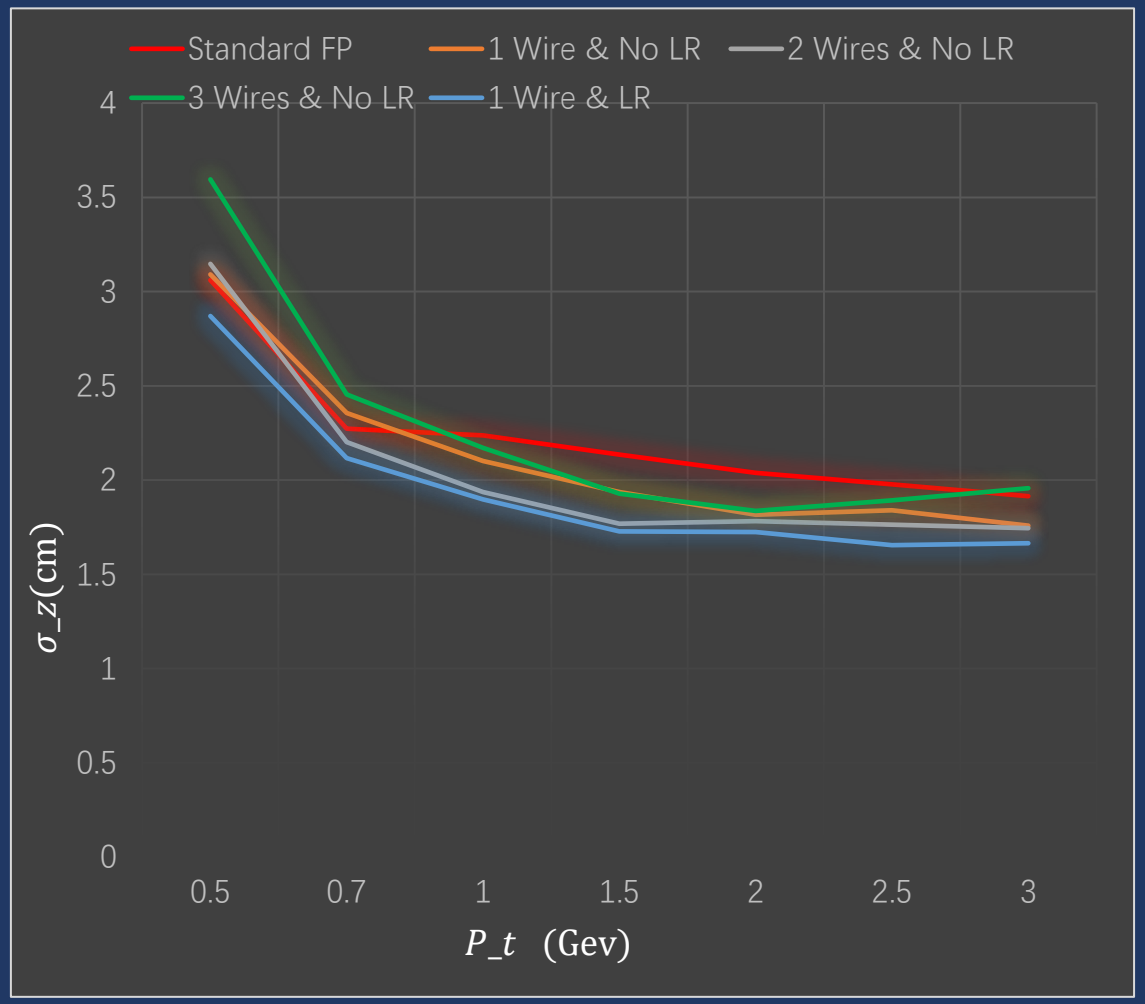
ETF : Set Event T0 as zero for precise t_{drift}

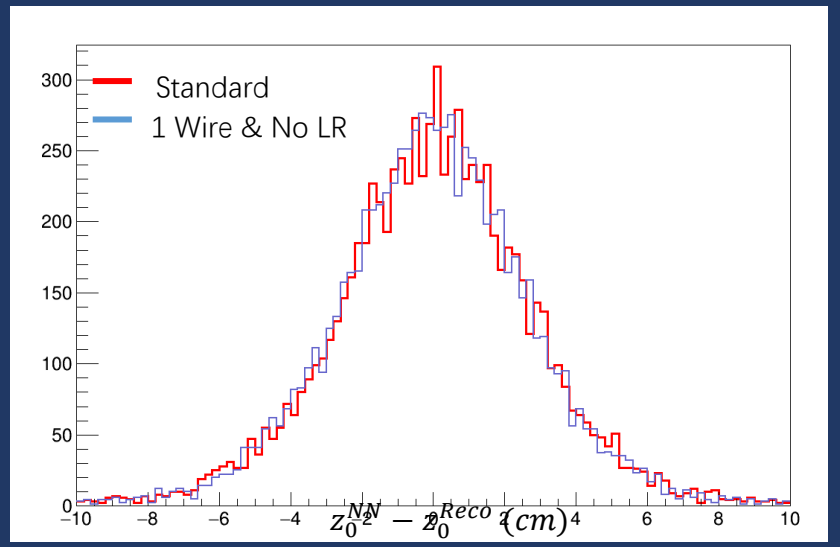
Add More wires without L/R make little improvement

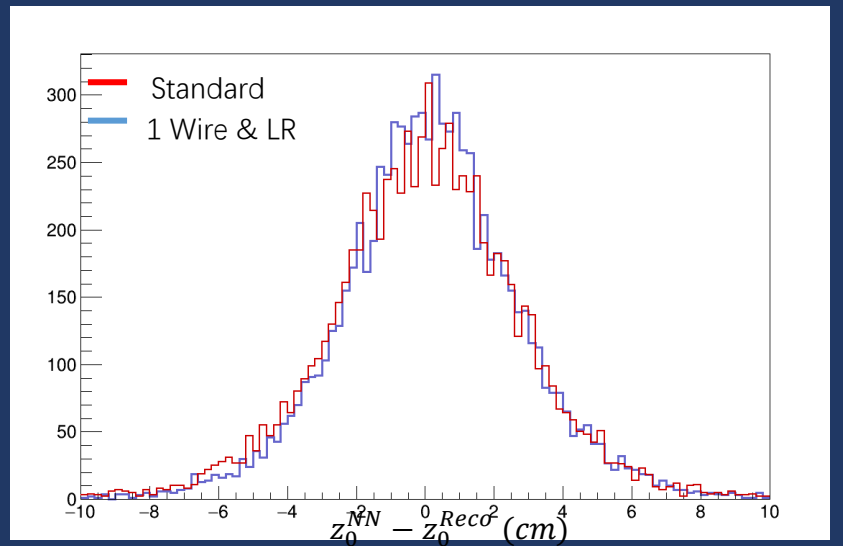
Add wire with L/R could make slightly difference in MC



Extra wire as input -- fastpriority

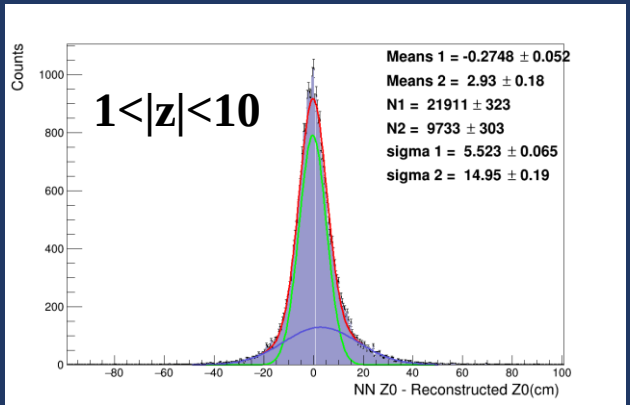


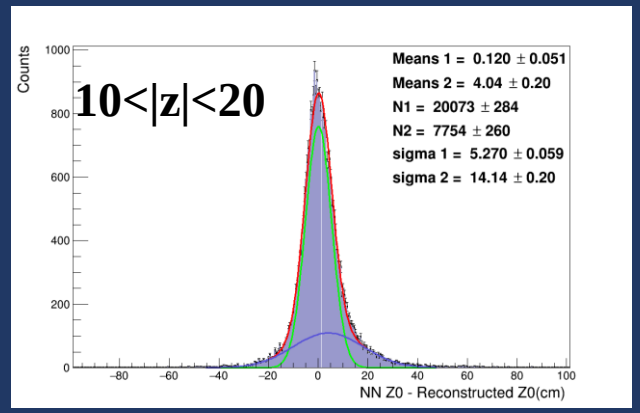


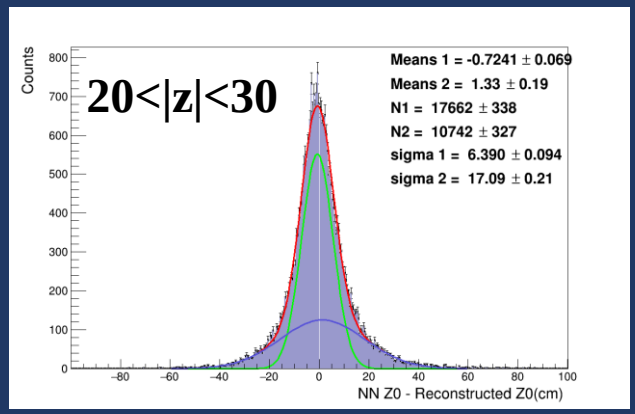


More wires?

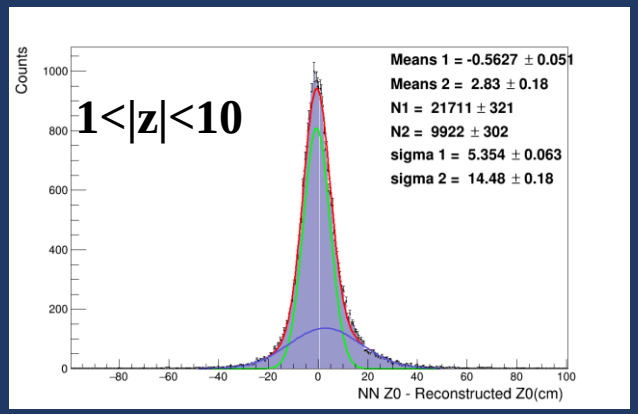
Wire 1

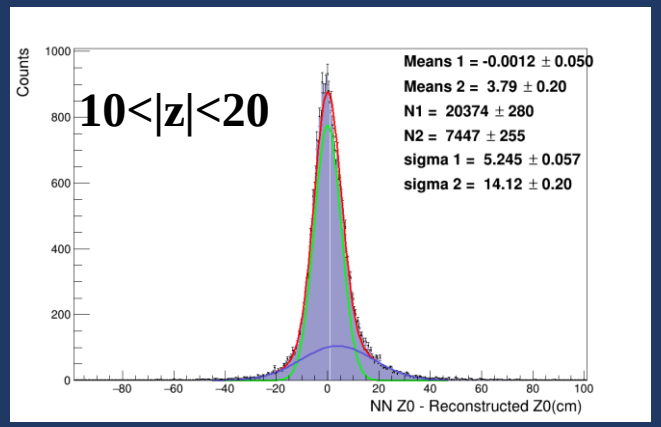


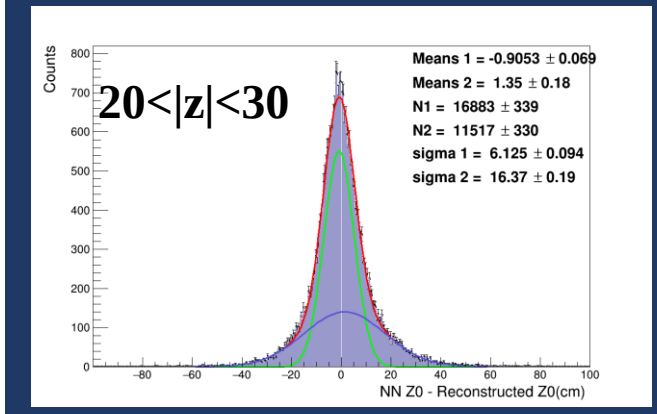




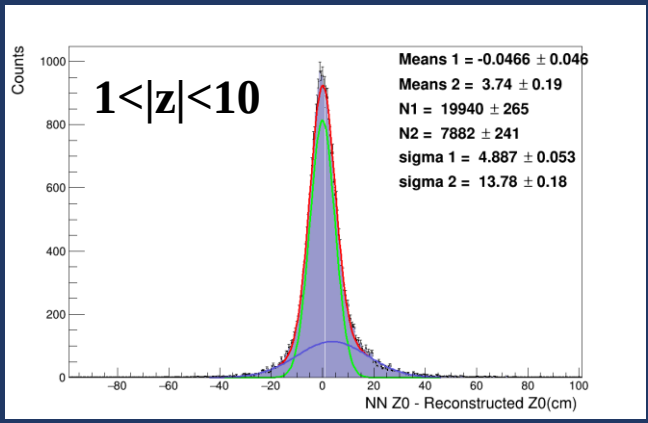
Wire 2

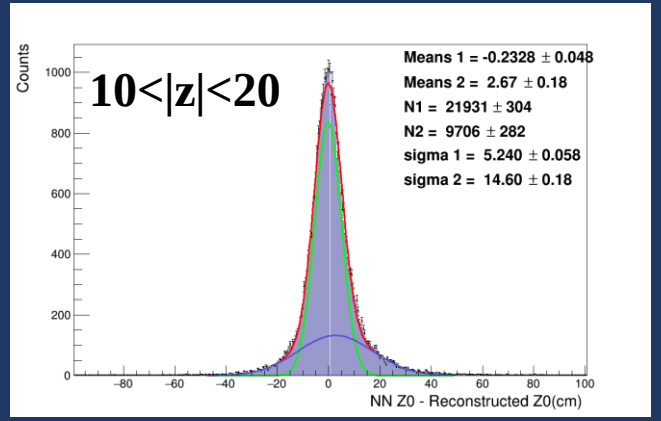


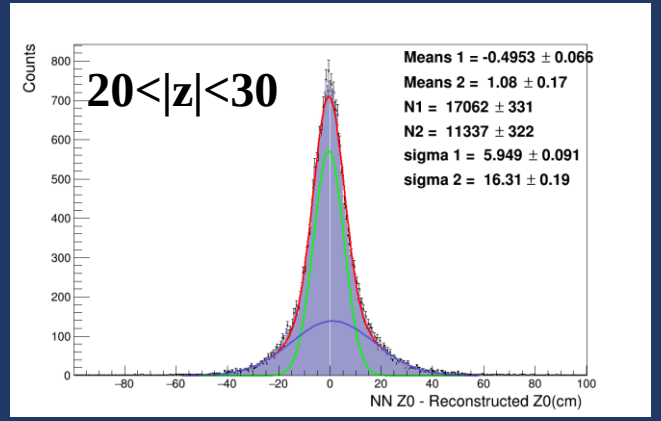




Wire 3

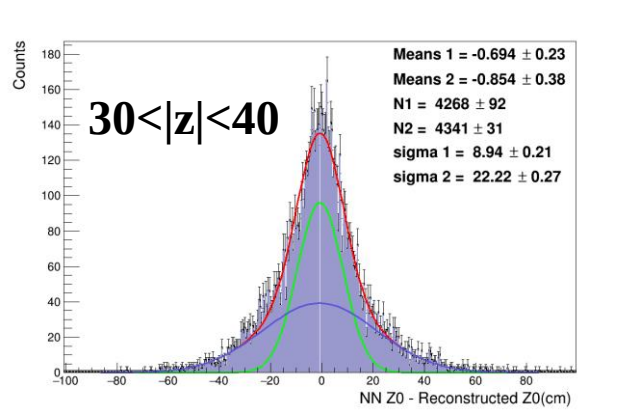


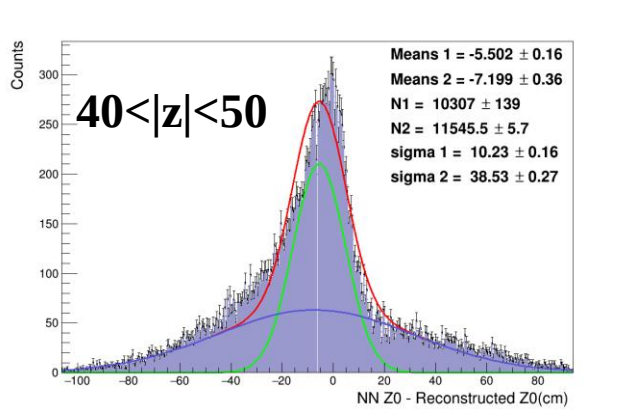




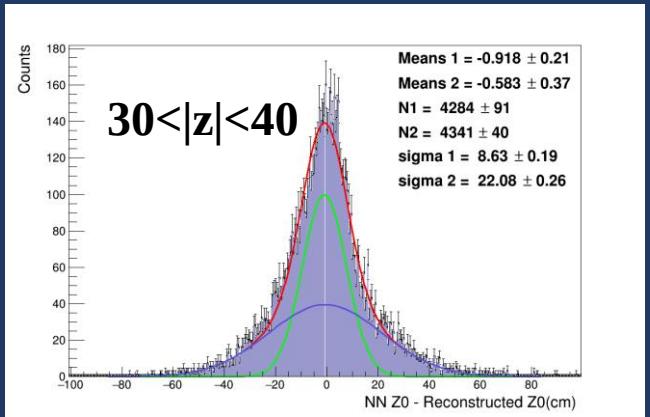
More wires?

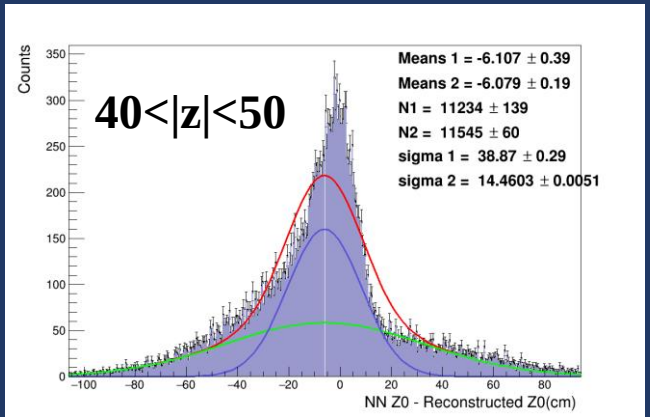
Wire 1



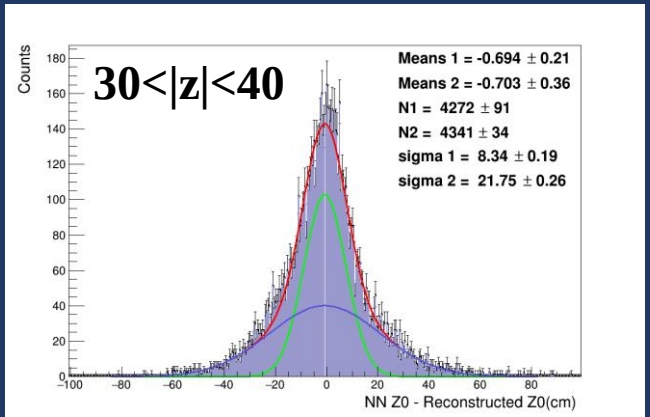


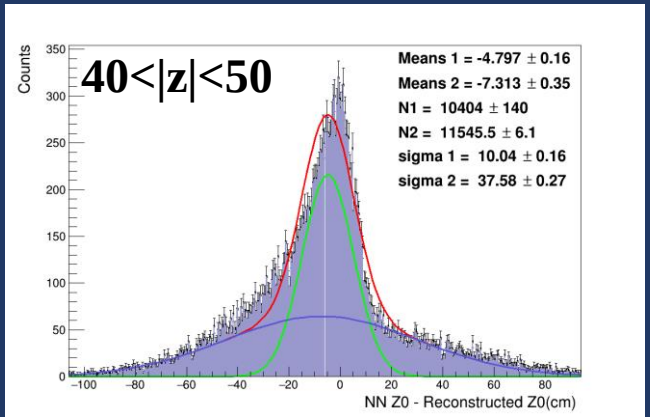
Wire 2





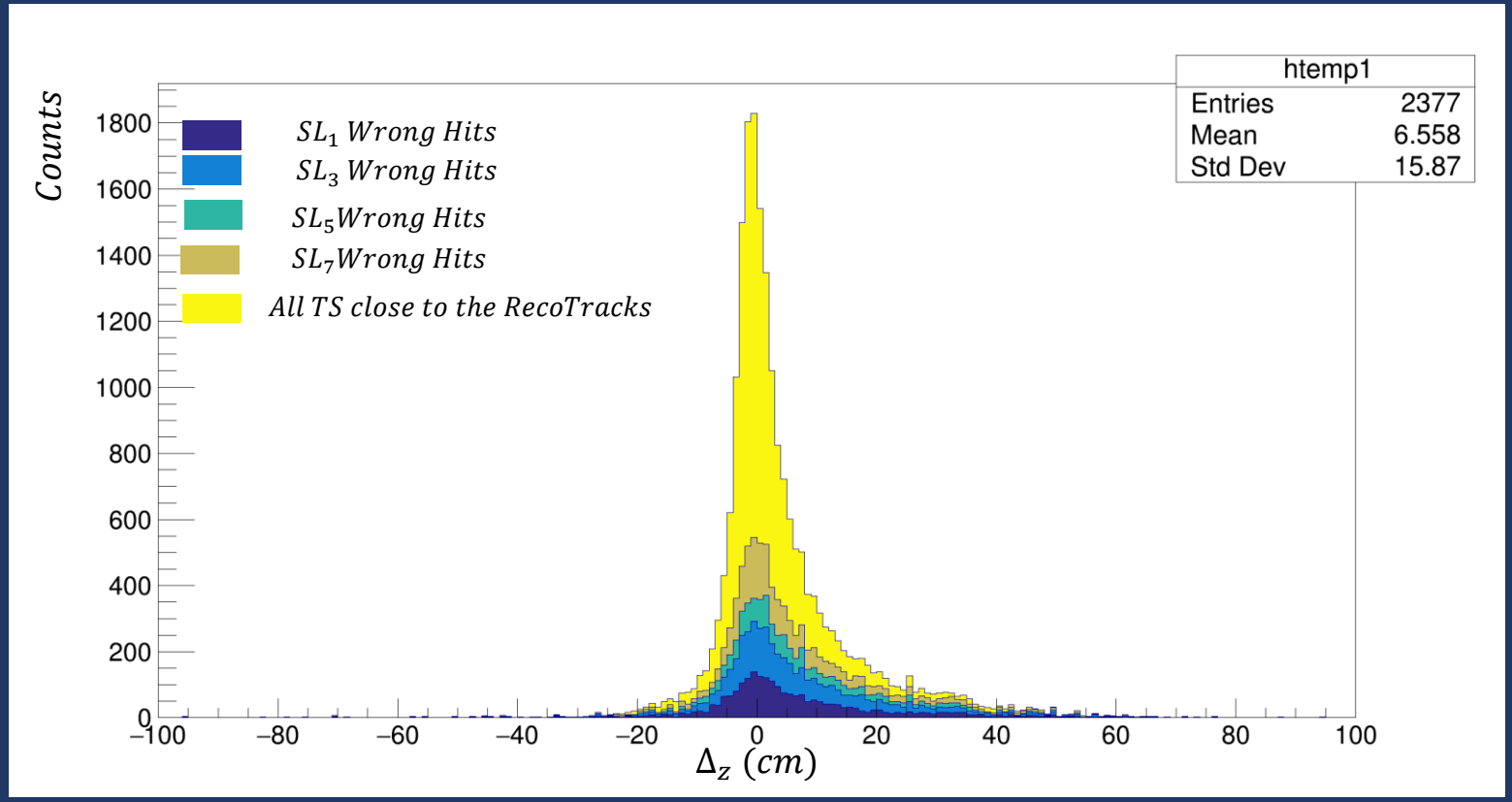
Wire 3





Track Segment ID

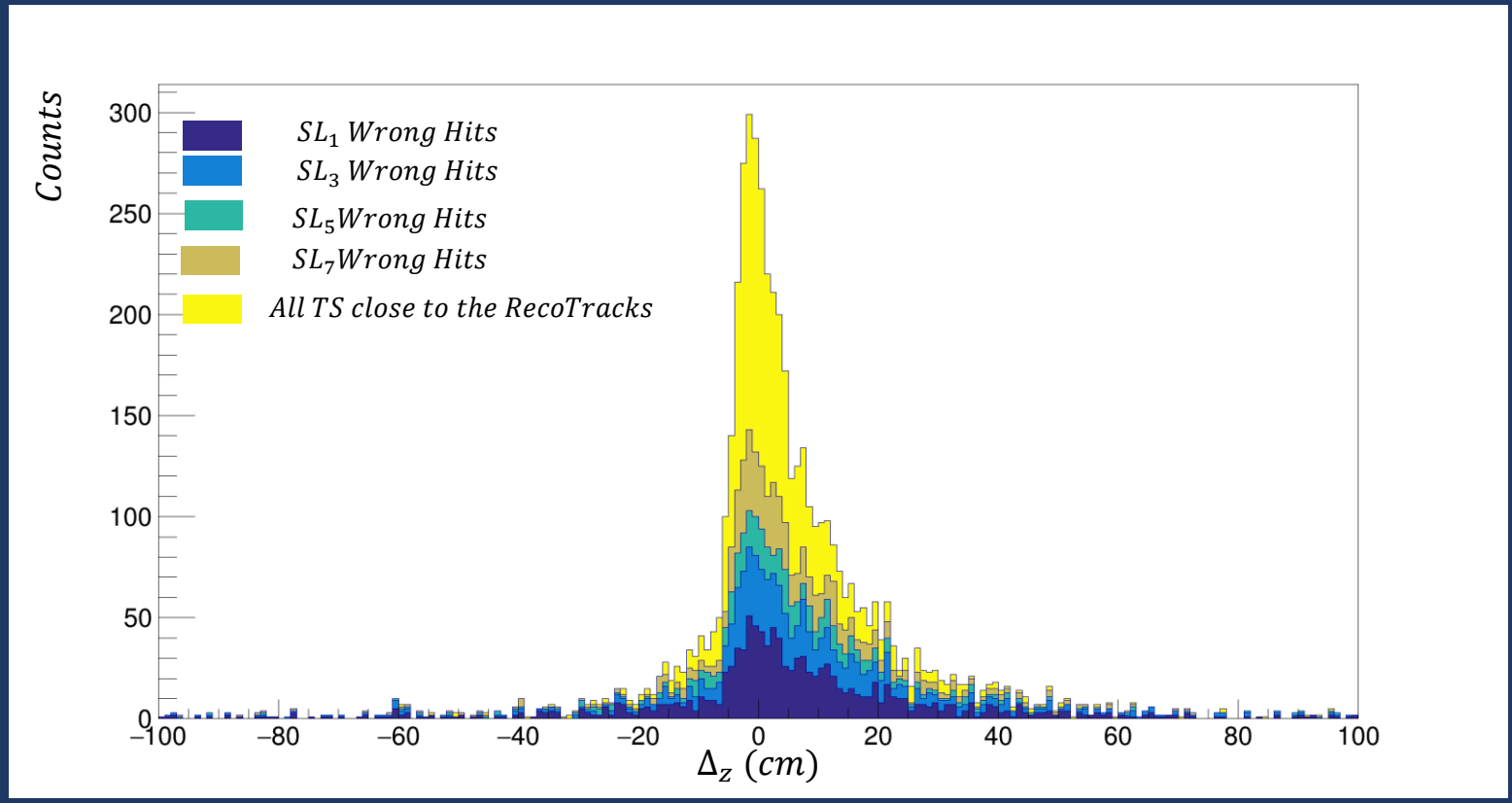
Consider the Delta Z distribution of those NN choose wrong hit



exp024run2004

Track Segment ID

Consider the Delta Z distribution of those NN choose wrong hit



exp026run1777