Particle flow networks for kinematic reconstruction in semi-inclusive deep-inelastic scattering at the ePIC experiment

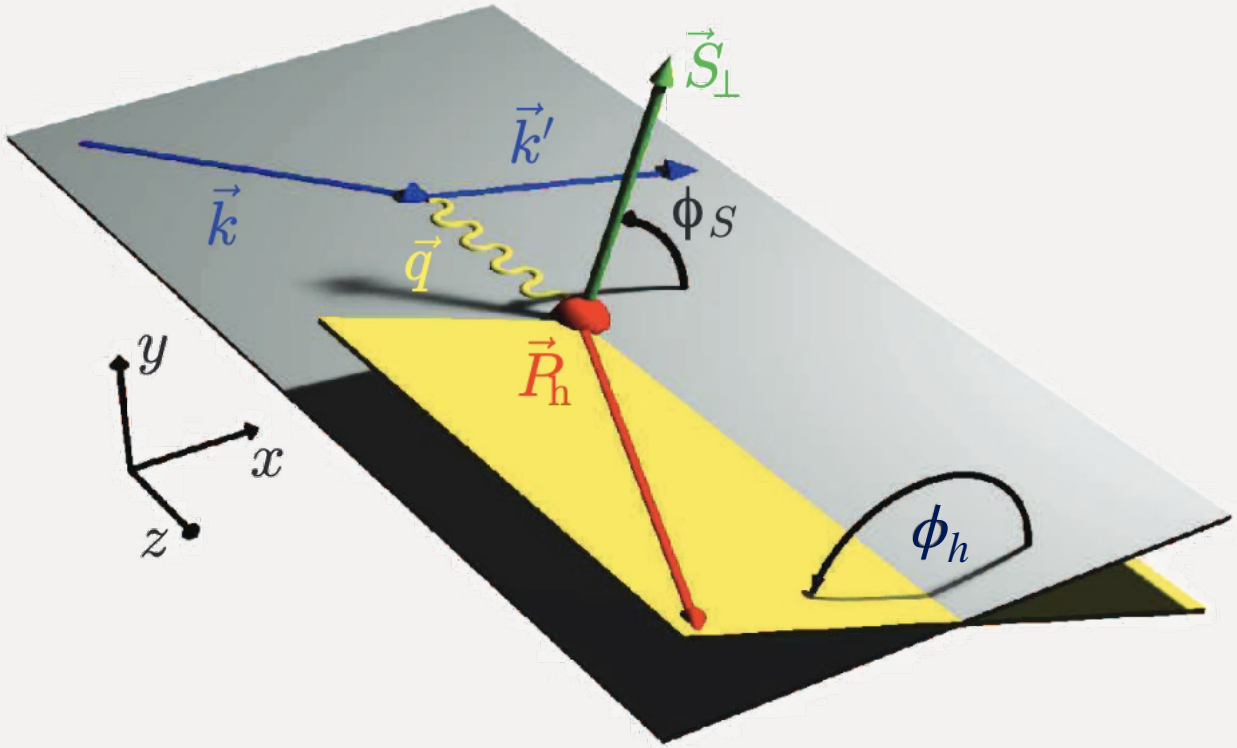
Daniel Almonte[1], Anselm Vossen[2]

[1] Stony Brook University
[2] Duke University

Duke

Semi-Inclusive Deep Inelastic Scattering (SIDIS)

$$e + N \rightarrow e' + h + X$$



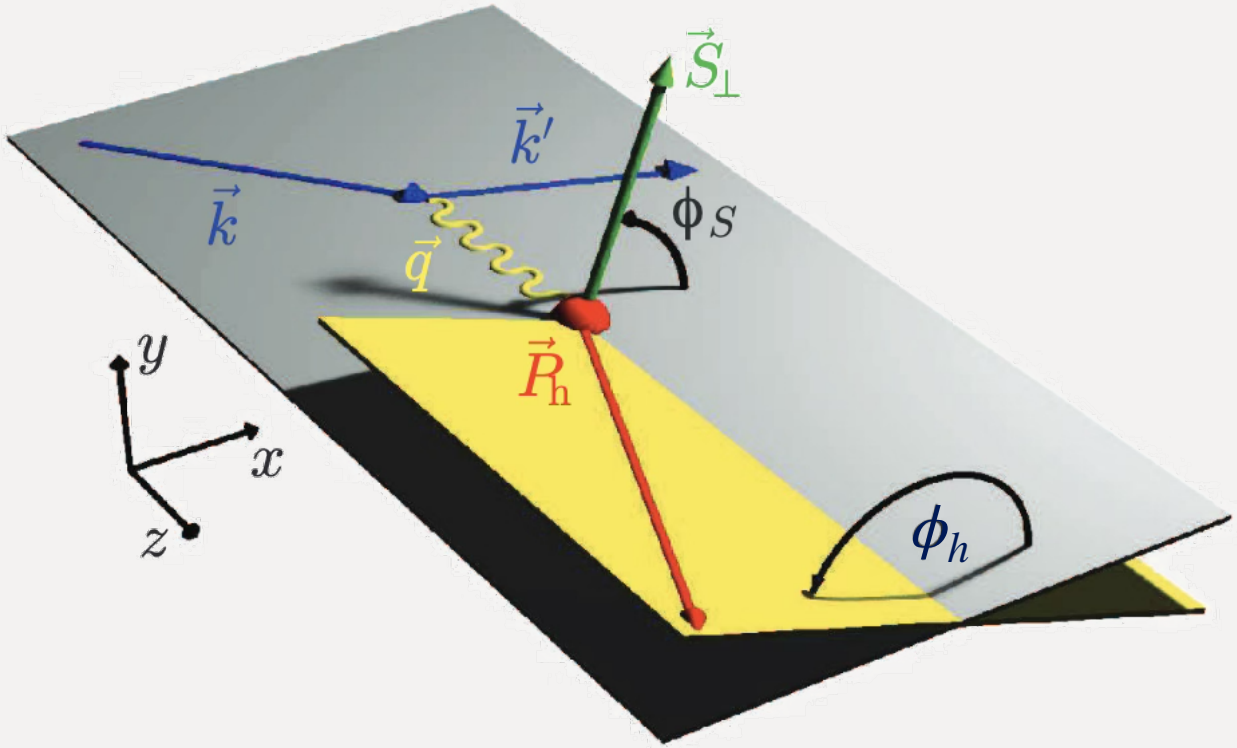
Differential Cross Section

$$\frac{d^5\sigma}{dx\ dQ^2\ dz\ dP_{hT}^2\ d\phi_h}$$

Duke

Semi-Inclusive Deep Inelastic Scattering (SIDIS)

$$e + N \rightarrow e' + h + X$$



Differential Cross Section

$$\frac{d^5\sigma}{dx\ dQ^2\ dz\ dP_{hT}^2\ d\phi_h}$$

DNNs have already been shown to improve the reconstruction of the inclusive variables …

Reconstructing the Kinematics of Deep Inelastic Scattering with Deep Learning

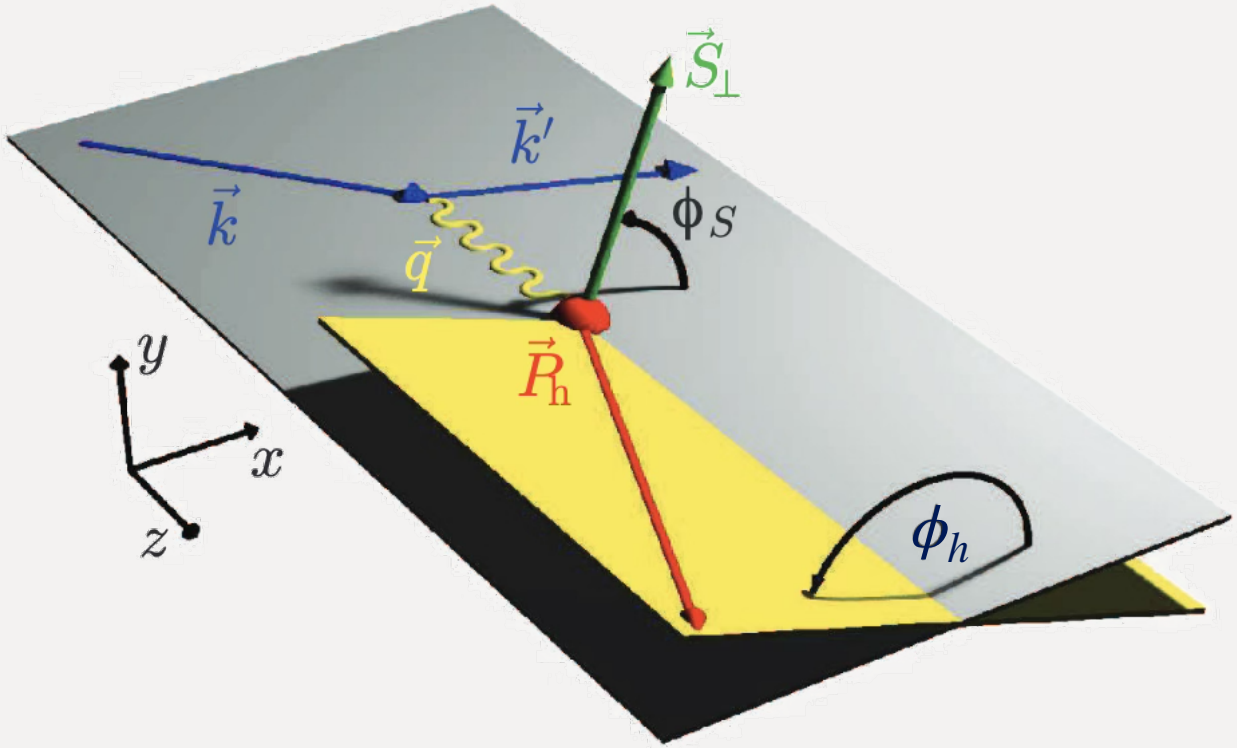
Miguel Arratia,[a,b] Daniel Britzger,[c] Owen Long,[*a] and Benjamin Nachman[d,e]

[a] *Department of Physics and Astronomy, University of California, Riverside, CA 92521, USA*
[b] *Thomas Jefferson National Accelerator Facility, Newport News, VA 23606, USA*
[c] *Max-Planck-Institut für Physik, Föhringer Ring 6, 80805 München, Germany*
[d] *Physics Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA*
[e] *Berkeley Institute for Data Science, University of California, Berkeley, CA 94720, USA*

Duke

Semi-Inclusive Deep Inelastic Scattering (SIDIS)

$$e + N \rightarrow e' + h + X$$



Differential Cross Section

$$\frac{d^5\sigma}{dx\ dQ^2\ dz\ dP_{hT}^2\ d\phi_h}$$

but ML has not yet been applied to the reconstruction of the semi-inclusive variables…

z : hadron momentum fraction

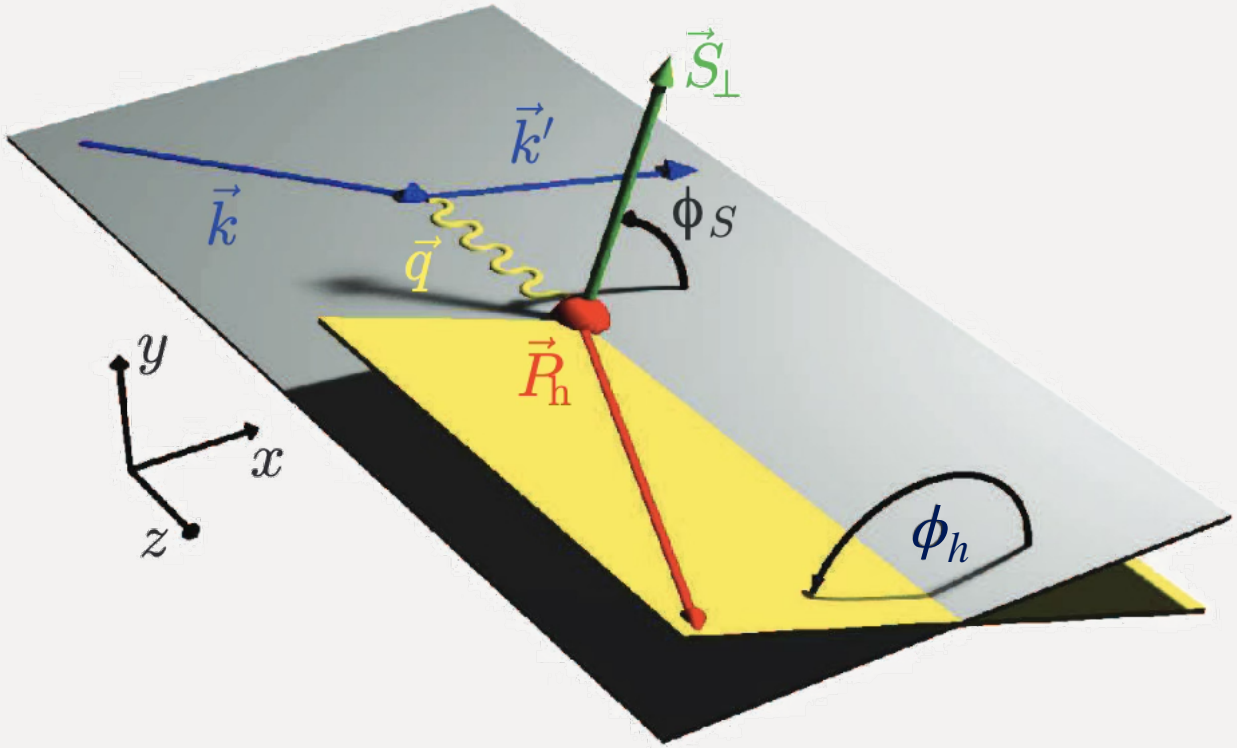
P_{hT} : hadron transverse momentum

ϕ_h : hadron plane angle

Duke

Kinematic Reconstruction

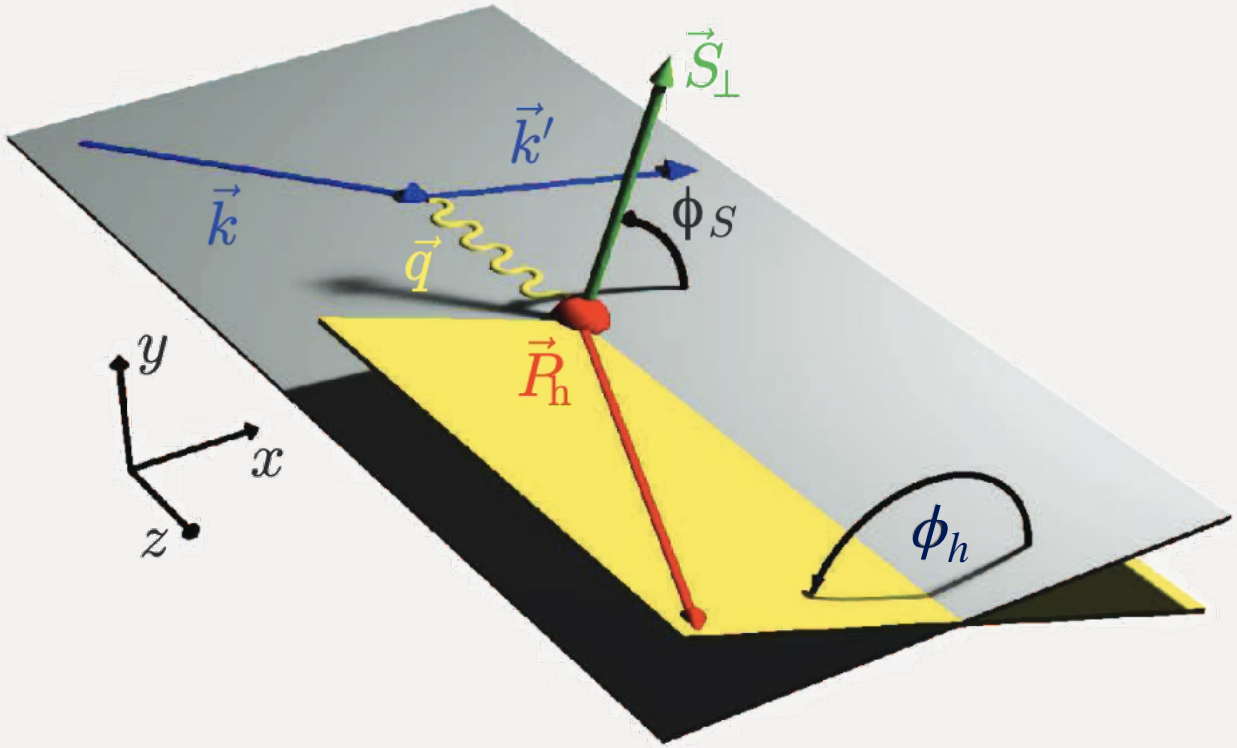
$$e + N \rightarrow e' + h + X$$



Duke

Kinematic Reconstruction

$$e + N \rightarrow e' + h + X$$



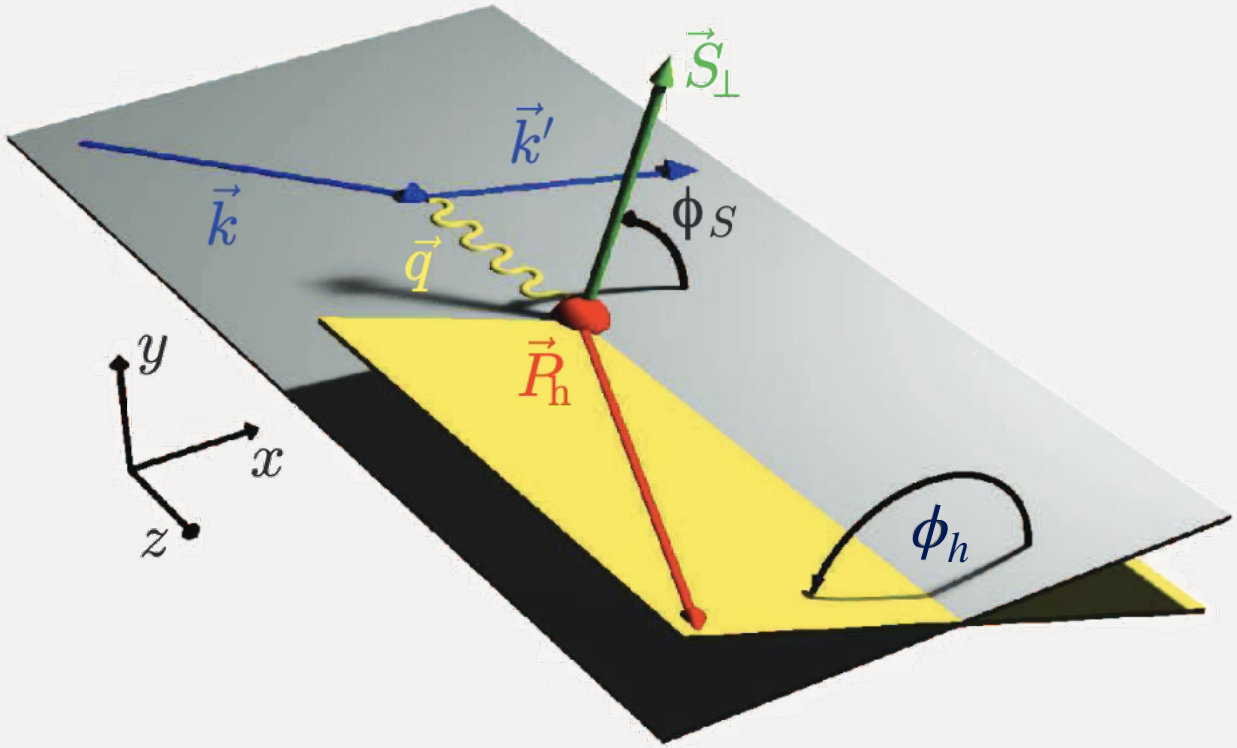
To calculate P_{hT} and ϕ_h we first need to **define our coordinate system** …

Our ability to reconstruct $\vec{q}$ will directly impact the accuracy of P_{hT} and ϕ_h

Duke

Kinematic Reconstruction

$$e + N \rightarrow e' + h + X$$



Methods for reconstructing $\vec{q}$

Electron Method
Only using scattered electron kinematics.

Jacquet-Blondel (JB) Method
Using the HFS kinematics.

Double Angle (DA) Method
Using both the scattered electron and HFS kinematics.

Machine Learning (ML) Method….

Duke

Kinematic Reconstruction Methods

Electron Method

$$q = k - k'$$

Jacquet-Blondel (JB) Method

$$\delta_h = \sum_i^{N_{HFS}} E_i - p_{z,i}$$

$$p_{t,h}^2 = \left(\sum_i^{N_{HFS}} p_{x,i} \right)^2 + \left(\sum_i^{N_{HFS}} p_{y,i} \right)^2$$

$$Q_{JB}^2 = \frac{p_{t,h}^2}{1 - y_{JB}}$$

$$y_{JB} = \frac{\delta_h}{2E_0}$$

$$q_x = \sum_i^{N_{HFS}} p_{x,i} \qquad q_y = \sum_i^{N_{HFS}} p_{y,i}$$

$$q_z, q_t \leftarrow \begin{cases} Q^2 = -\left(q_x^2 + q_y^2 + q_z^2 - q_t^2\right) \\ y = \dfrac{P_x q_x + P_y q_y + P_z q_z - P_t q_t}{P \cdot k} \end{cases}$$

Double Angle (DA) Method

$$\alpha_e = \tan \frac{\theta_e}{2}$$

$$\alpha_h = \frac{\delta_h}{p_{t,h}}$$

$$Q_{DA}^2 = \frac{4E_0^2}{\alpha_e(\alpha_e + \alpha_h)}$$

$$y_{DA} = \frac{\alpha_h}{\alpha_e + \alpha_h}$$

Kinematic Reconstruction Methods

Electron Method

$$q = k - k'$$

Jacquet-Blondel (JB) Method

$$\delta_h = \sum_i^{N_{HFS}} E_i - p_{z,i}$$

$$p_{t,h}^2 = \left(\sum_i^{N_{HFS}} p_{x,i}\right)^2 + \left(\sum_i^{N_{HFS}} p_{y,i}\right)^2$$

$$Q_{JB}^2 = \frac{p_{t,h}^2}{1 - y_{JB}}$$

$$y_{JB} = \frac{\delta_h}{2E_0}$$

$$q_x = \sum_i^{N_{HFS}} p_{x,i} \qquad q_y = \sum_i^{N_{HFS}} p_{y,i}$$

$$q_z, q_t \leftarrow \begin{cases} Q^2 = -(q_x^2 + q_y^2 + q_z^2 - q_t^2) \\ y = \dfrac{P_x q_x + P_y q_y + P_z q_z - P_t q_t}{P \cdot k} \end{cases}$$

Double Angle (DA) Method

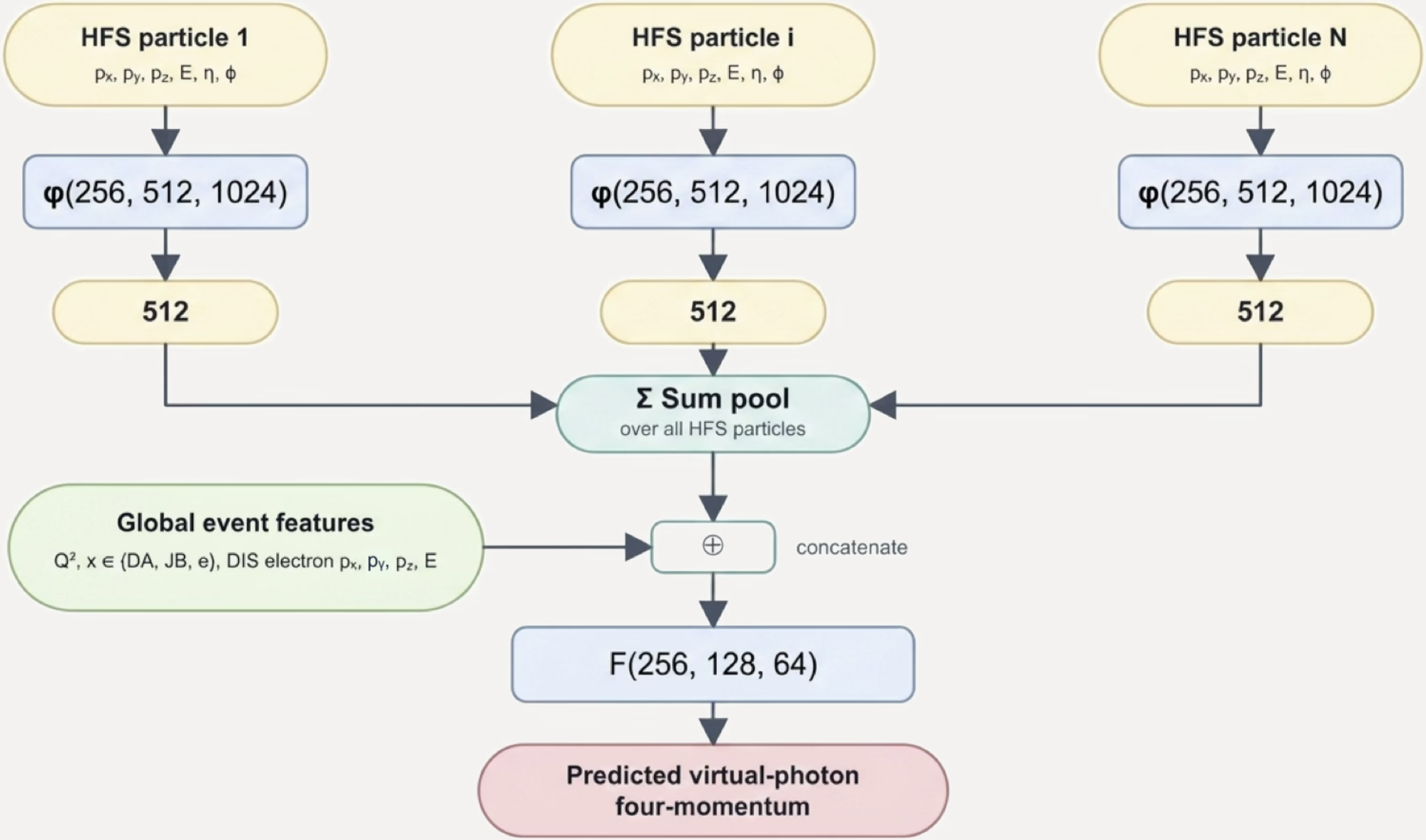
$$\alpha_e = \tan\frac{\theta_e}{2}$$

$$\alpha_h = \frac{\delta_h}{p_{t,h}}$$

$$Q_{DA}^2 = \frac{4E_0^2}{\alpha_e(\alpha_e + \alpha_h)}$$

$$y_{DA} = \frac{\alpha_h}{\alpha_e + \alpha_h}$$

Machine Learning Method: Particle Flow Network



Duke

Machine Learning Method: Particle Flow Network

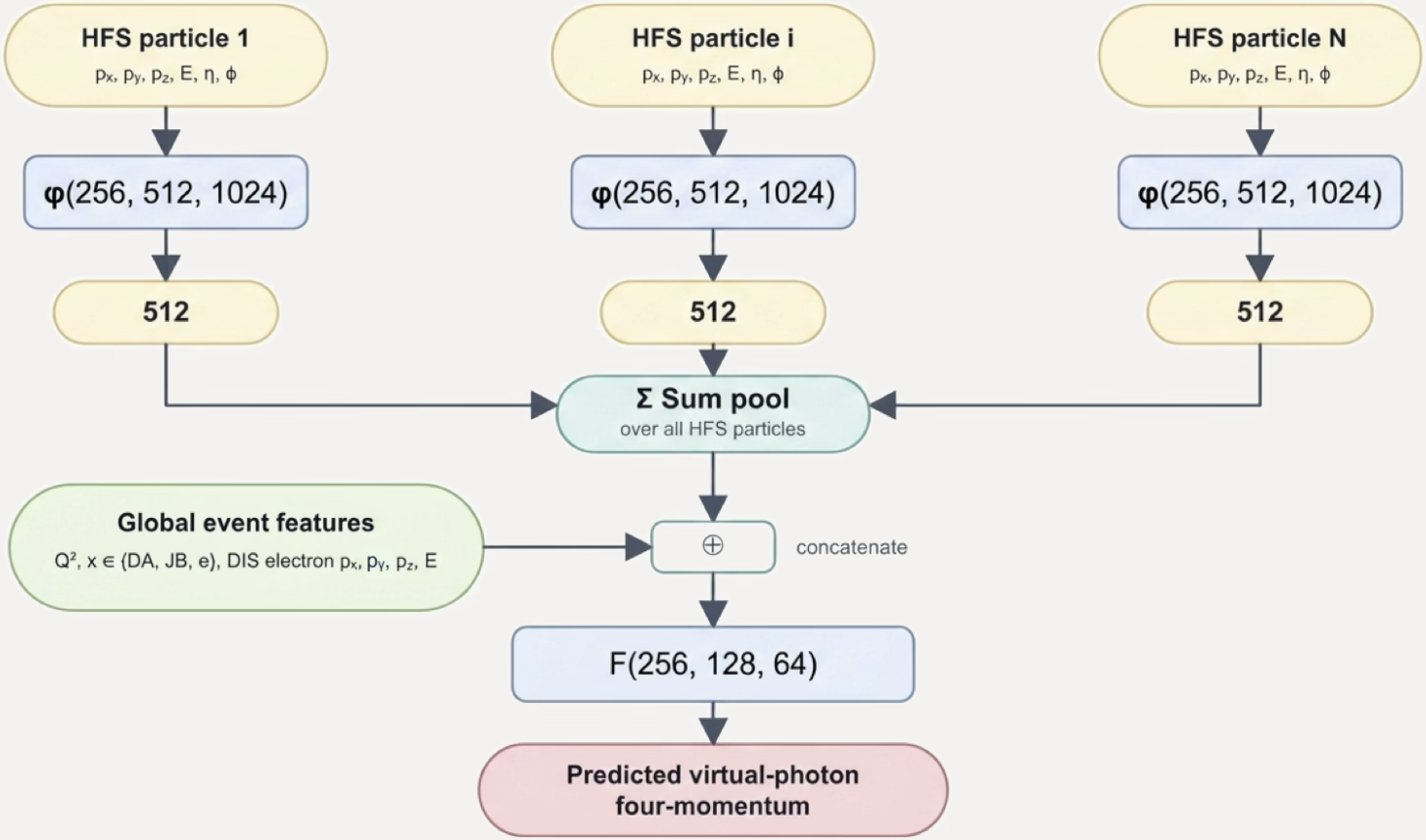
PFNs are permutation invariant.

Training dataset: ~500,000 events
Validation dataset: ~60,000 events
Test dataset: ~60,000 events

All layers of ϕ and F use a ReLU activation function.

The final output layer uses a linear activation function.

Batch size: 1000



Initial learning rate is $3.0 \cdot 10^{-4}$.
Learning rate is cut in half whenever validation loss
Has not improved after five epochs.

Results

Simulated Dataset Info:

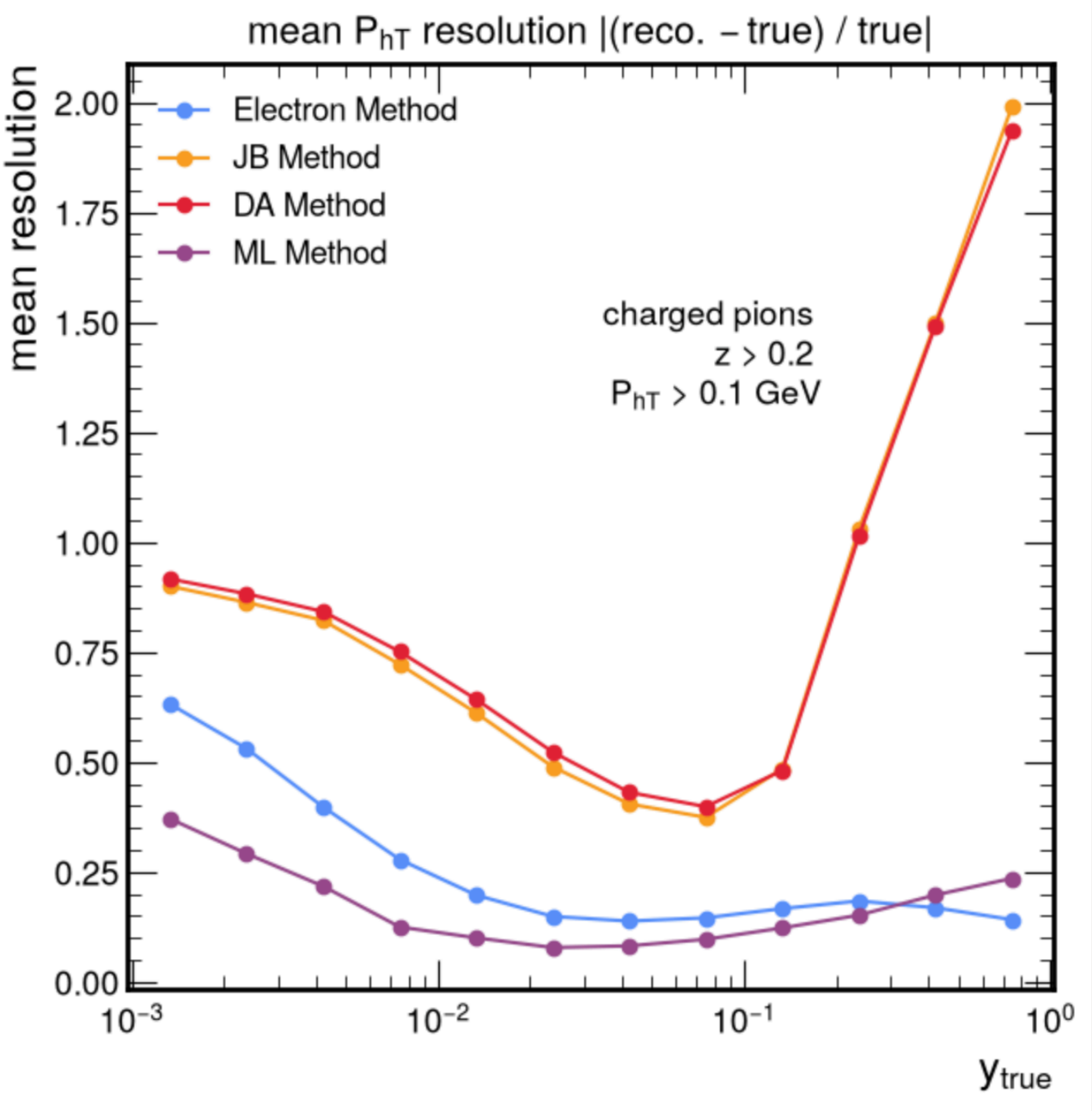
- 25.10.4 Campaign
  - DJANGOH version 4.6.21-1.0
- Neutral Current
- 18 GeV e- beam
- 275 GeV proton beam
- $1\ GeV^2 \leq Q^2 \leq 10^4\ GeV^2$
- 60,000 events

Duke

Results

$$y = \frac{E_\gamma}{E_e}$$

(in the target rest frame)

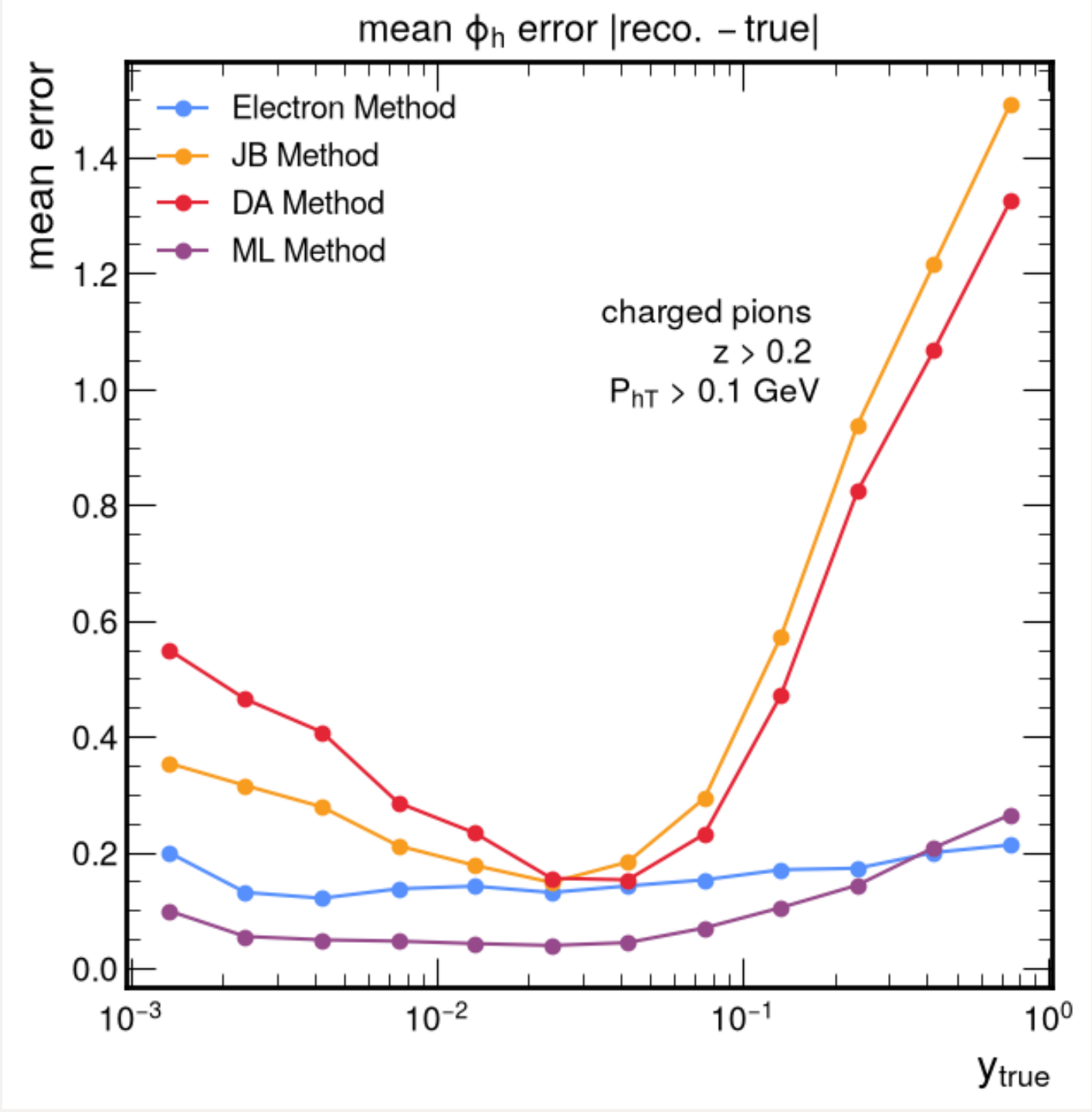


Duke

Results

$$y = \frac{E_\gamma}{E_e}$$

(in the target rest frame)

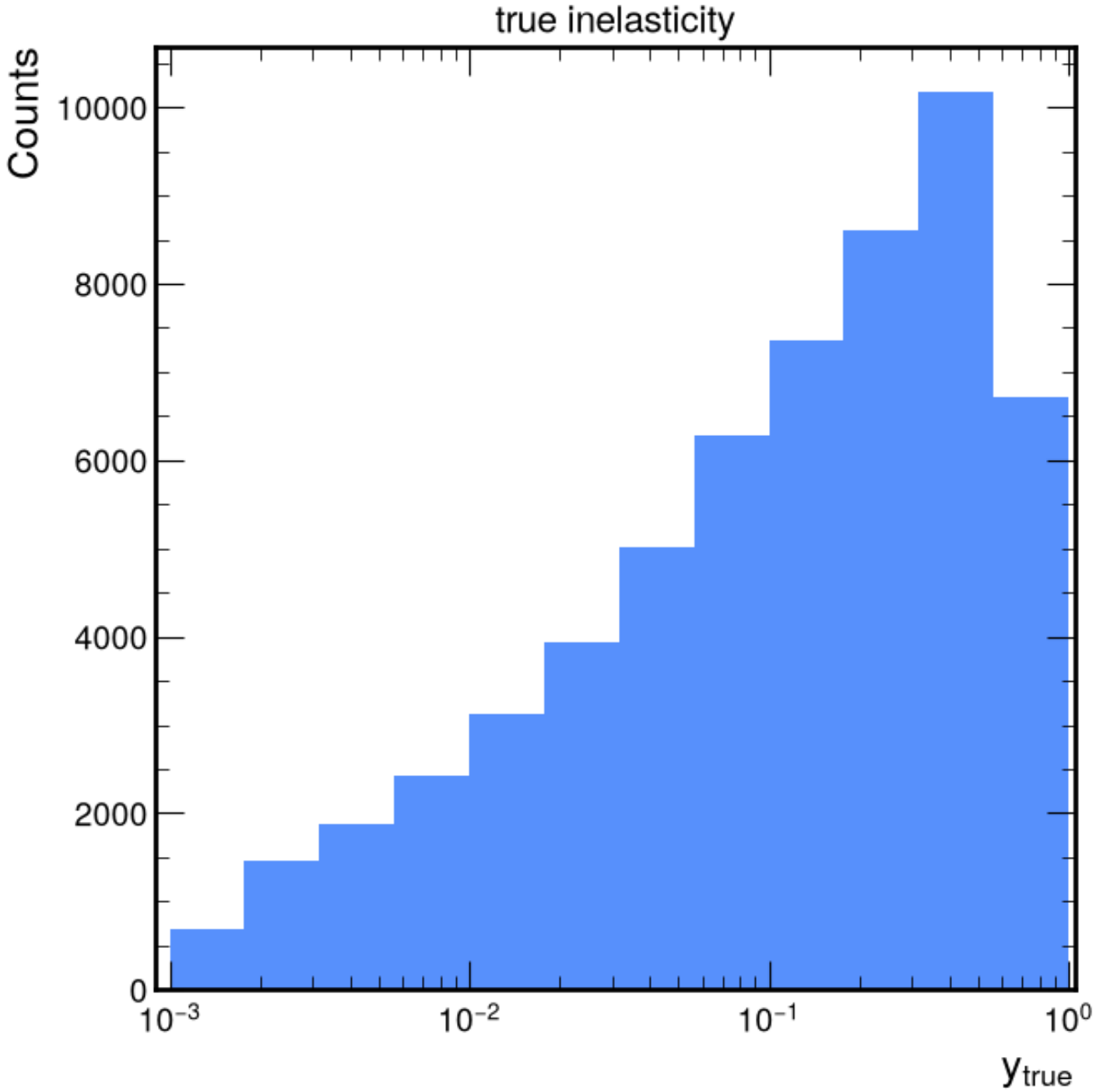


Duke

Results

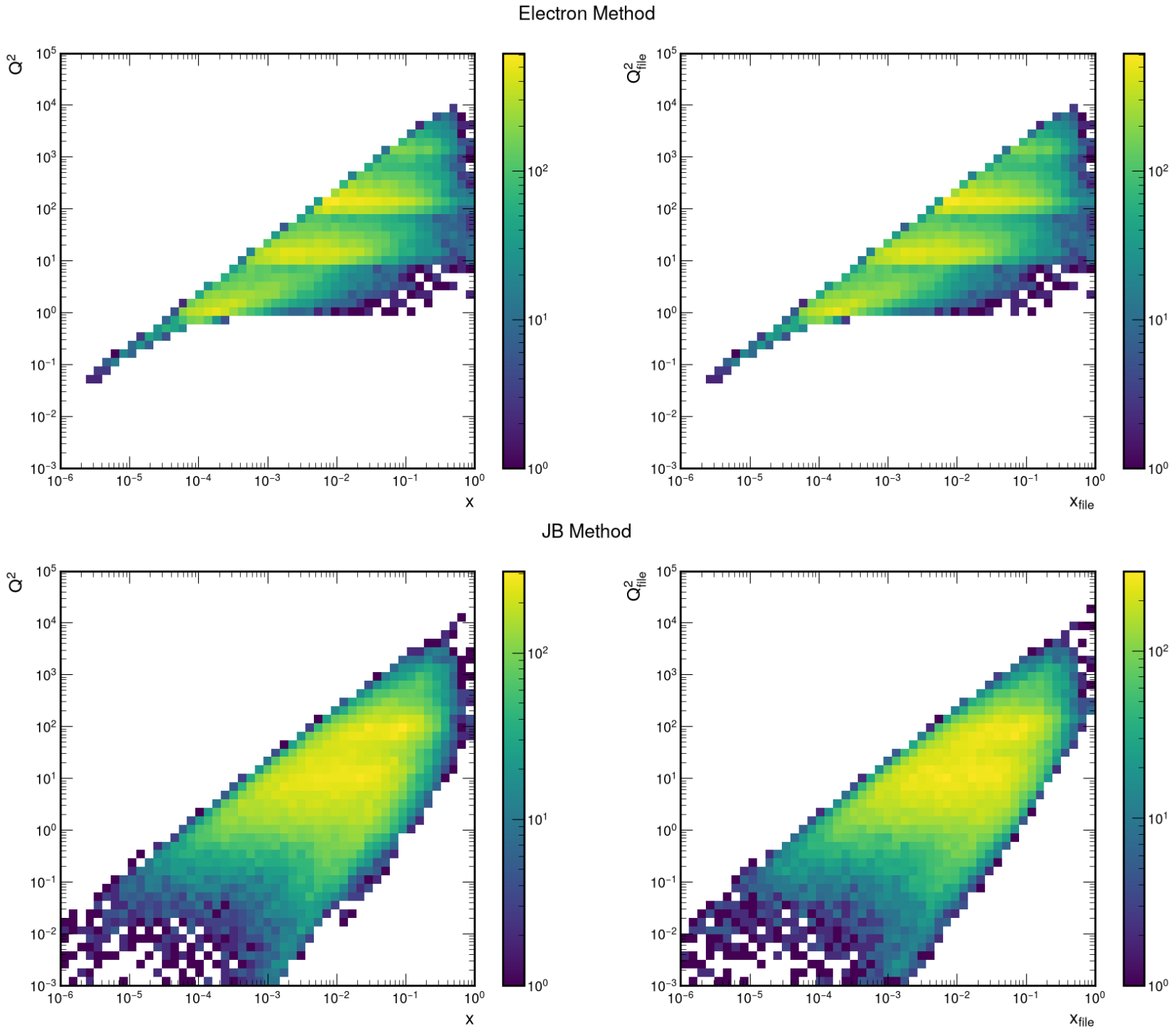
$$y = \frac{E_\gamma}{E_e}$$

(in the target rest frame)



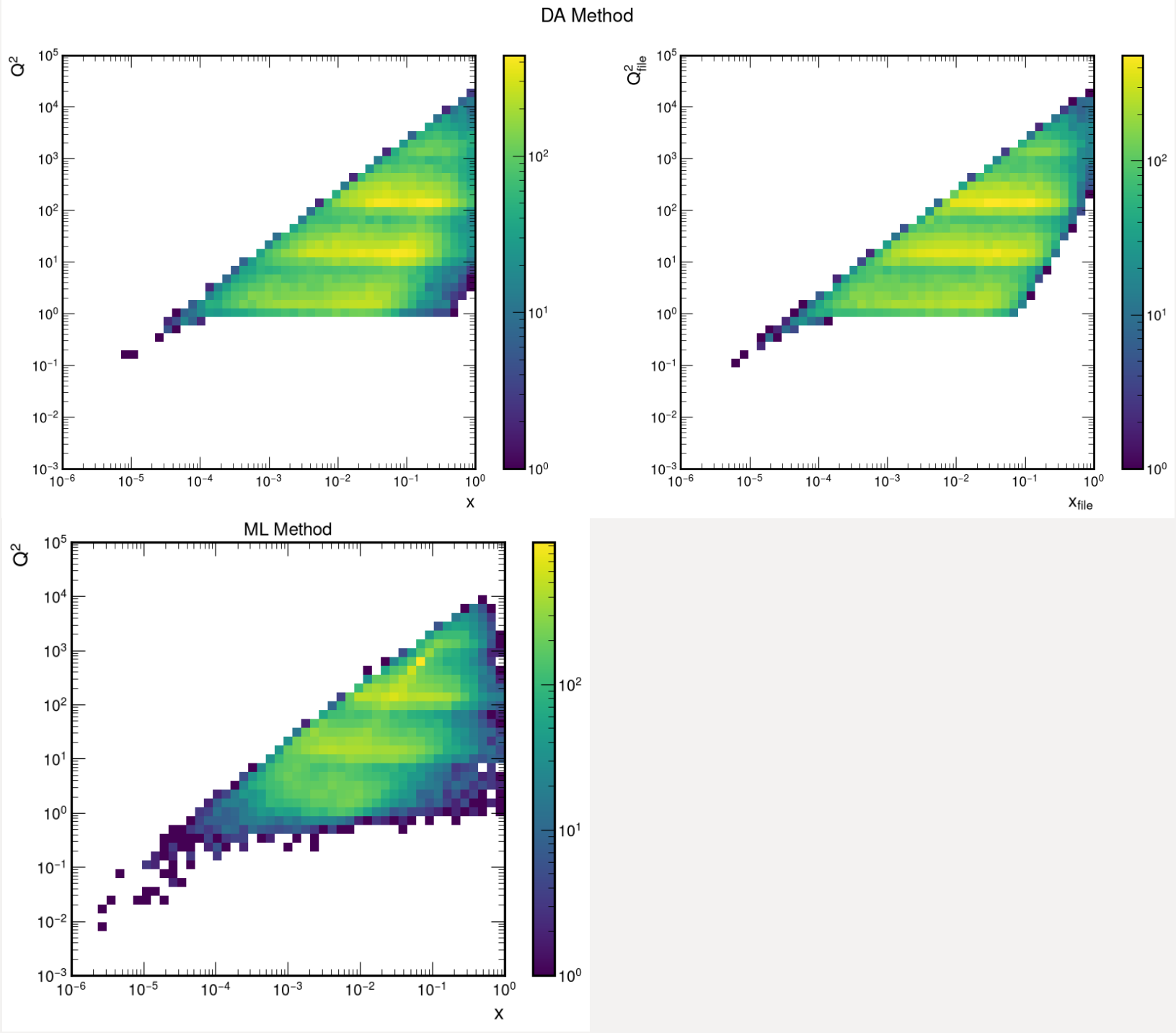
Duke

Results



Duke

Results

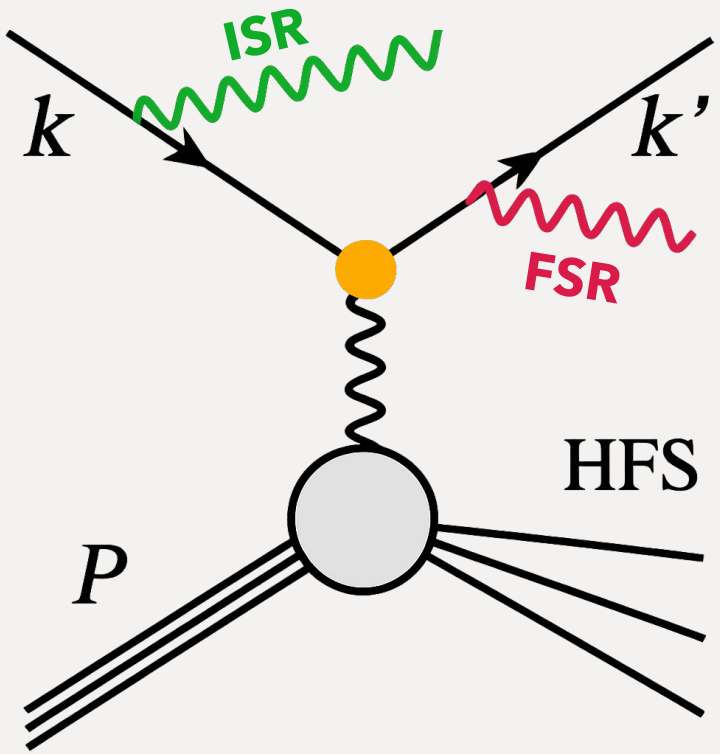


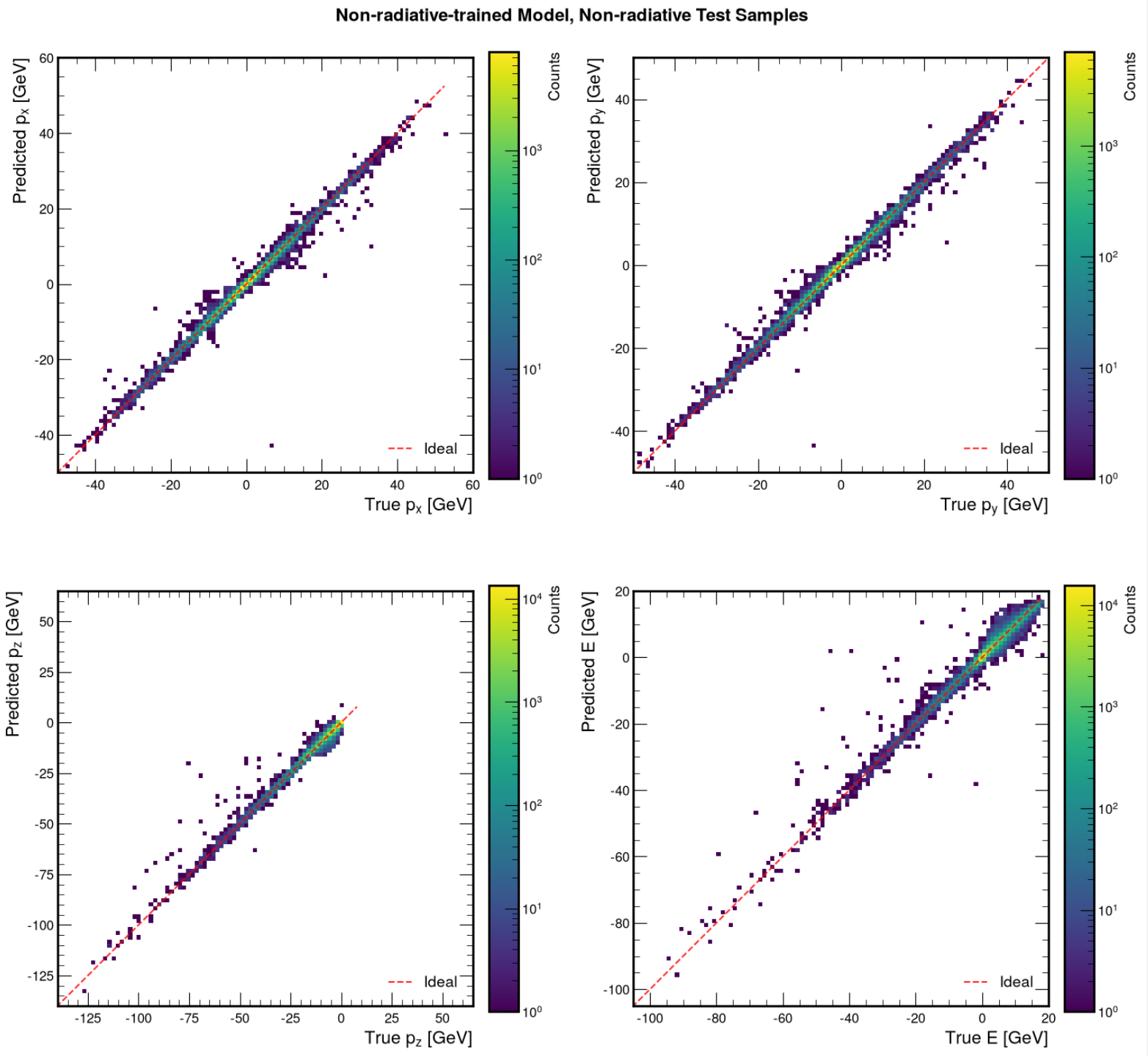
Duke

Results

How well does the PFN account for radiation?

- Initial State Radiation (ISR)
- Final State Radiation (FSR)

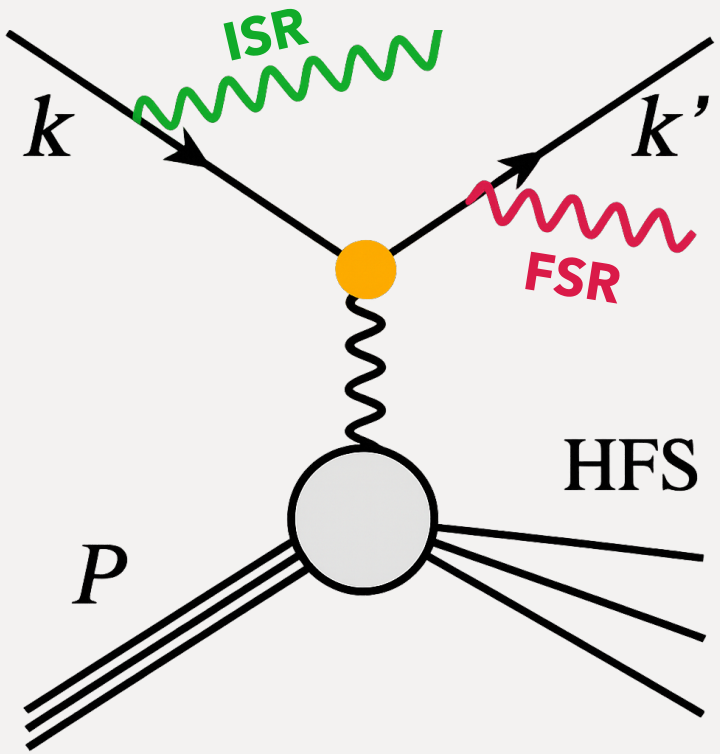


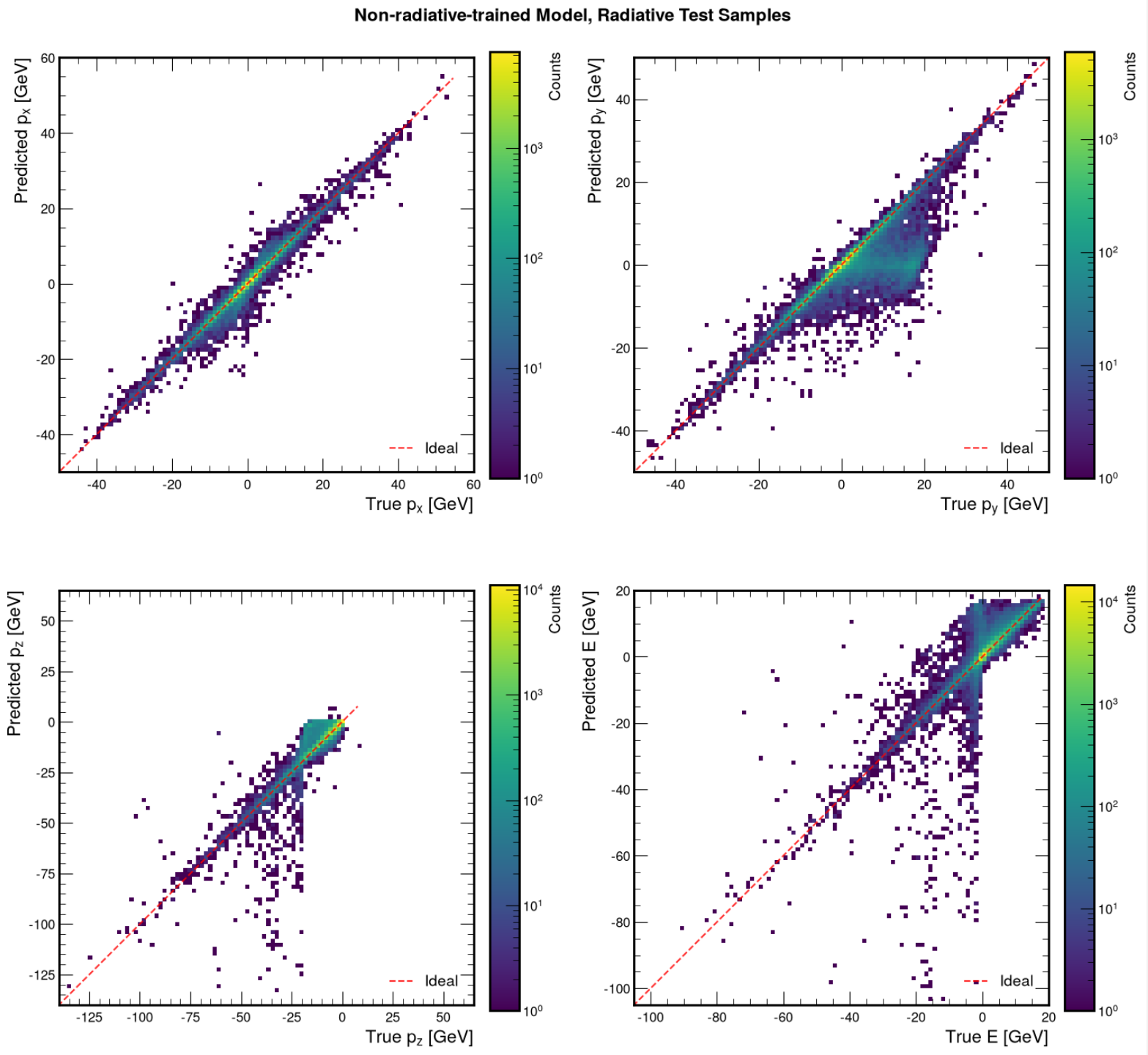


Results

How well does the PFN account for radiation?

- Initial State Radiation (ISR)
- Final State Radiation (FSR)

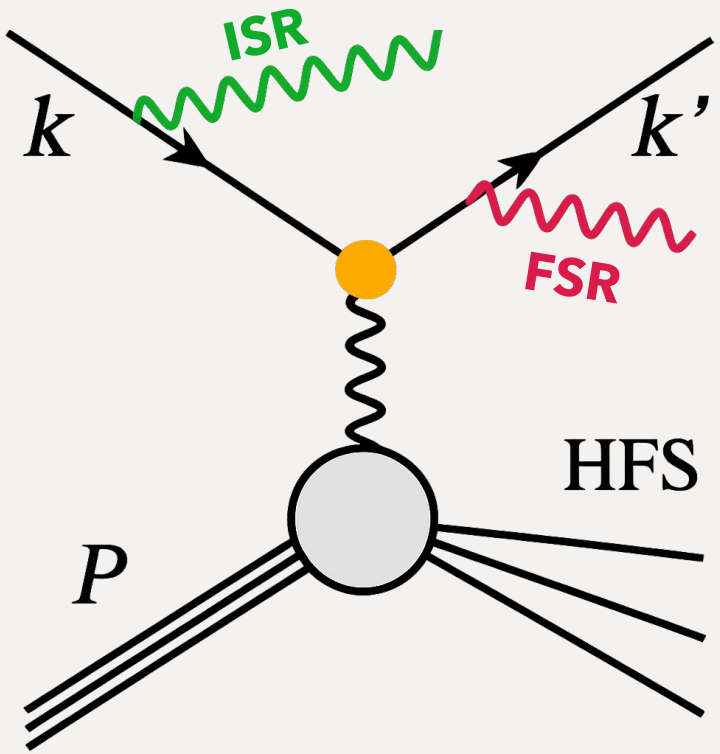


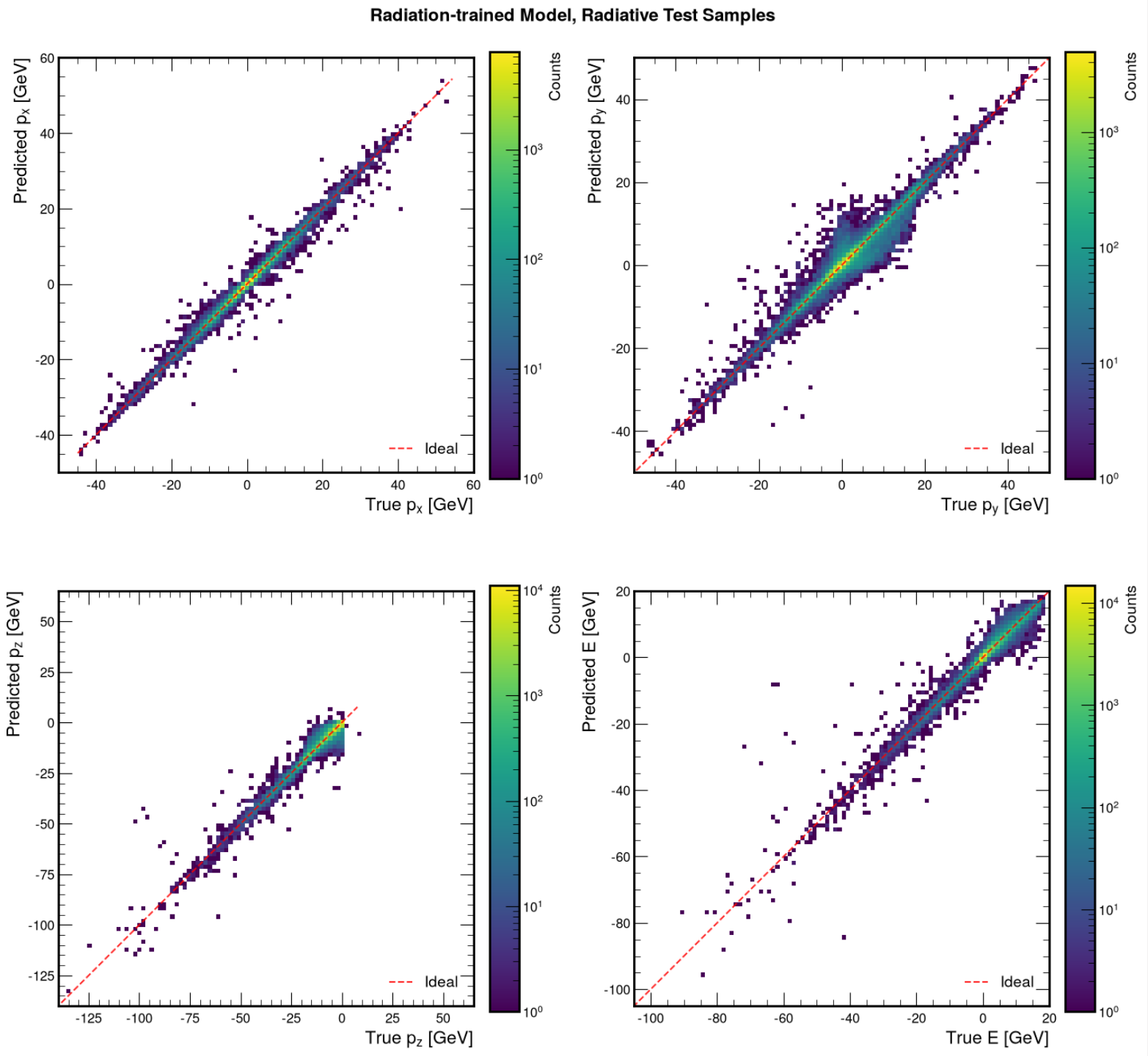


Results

How well does the PFN account for radiation?

- Initial State Radiation (ISR)
- Final State Radiation (FSR)

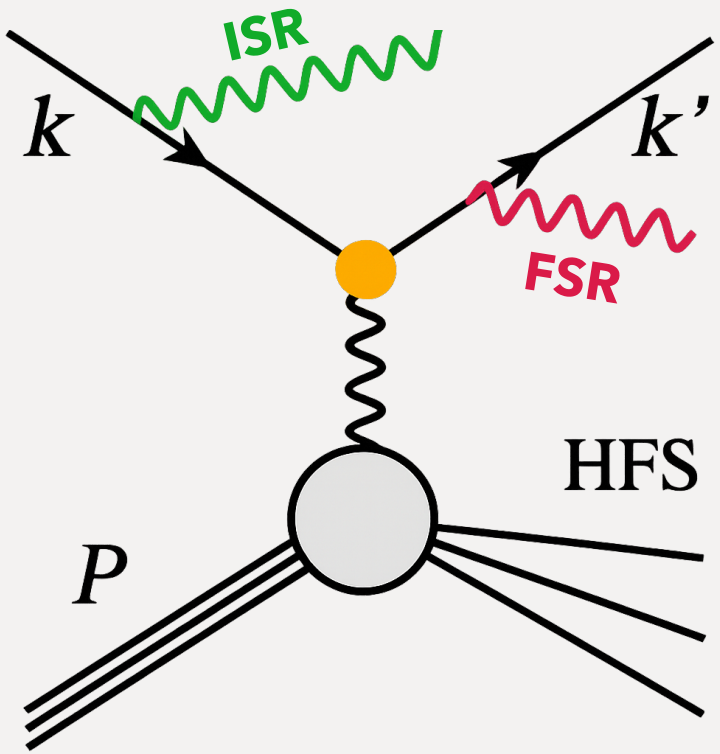


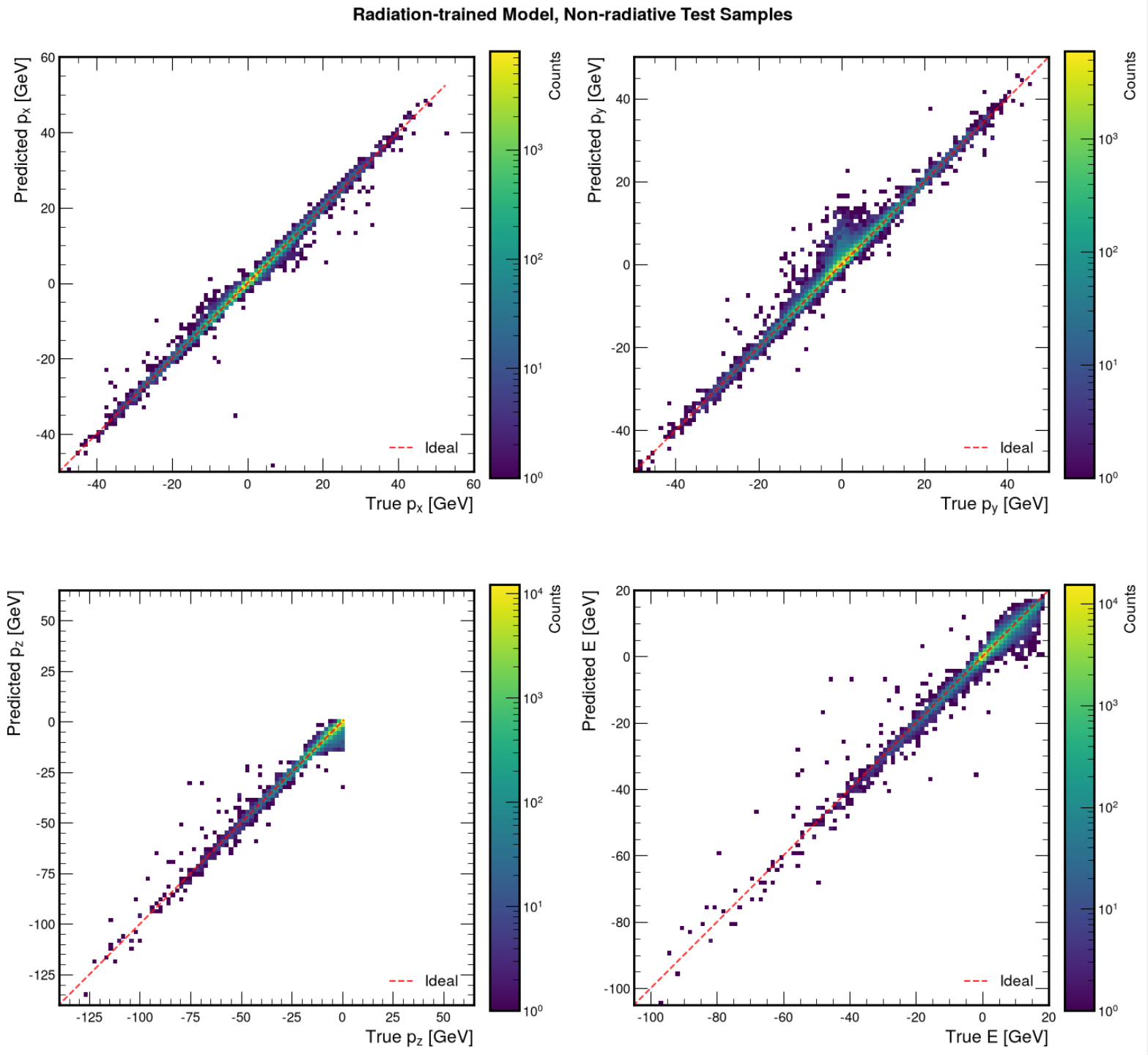


Results

How well does the PFN account for radiation?

- Initial State Radiation (ISR)
- Final State Radiation (FSR)





Conclusions

1. The PFN outperforms the JB and DA methods across the full y range, and outperforms the electron method for $y < 0.4$

2. The PFN improves its reconstruction when trained on events with radiation, suggesting that it learns to account for radiative effects.

Next Steps

1. Explore additional model inputs that may further improve its identification of radiation.
2. Write up a paper

Duke

Backup slides

Duke

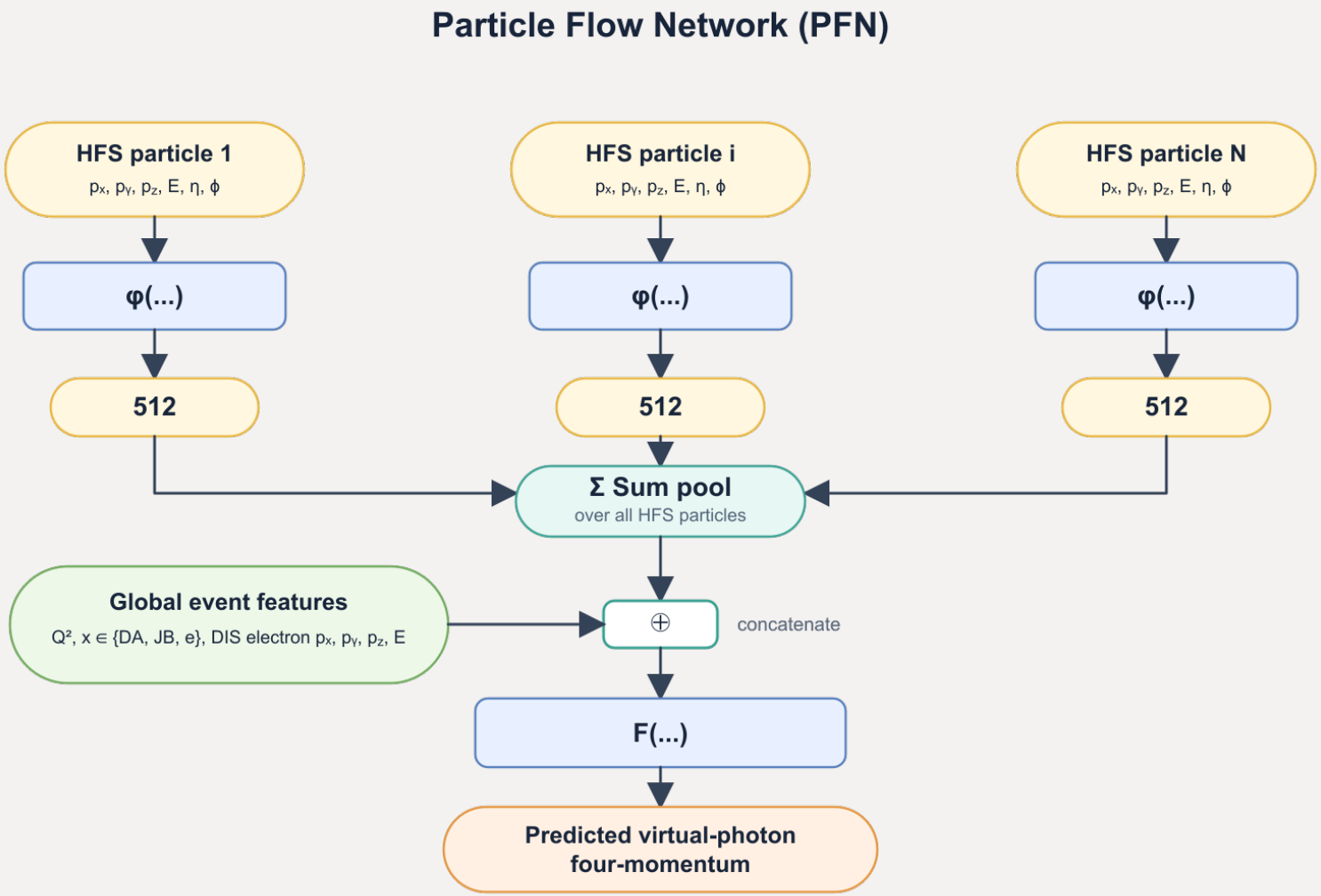
PFN Layer Dimensions

Model 1

$\phi(500, 500, 500)$, $F(200, 200, 200)$

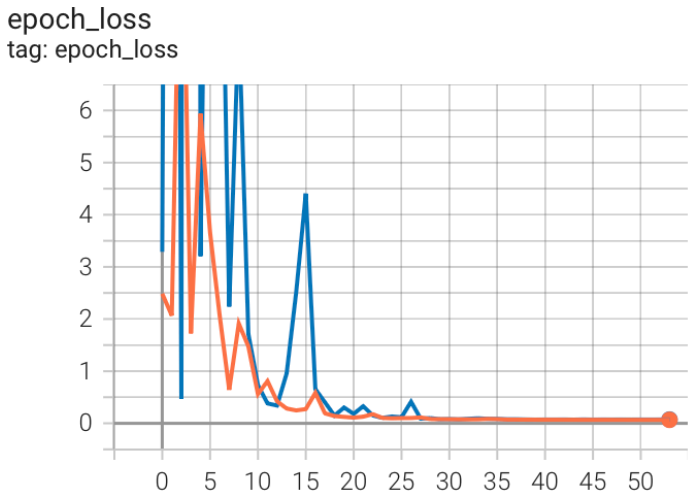
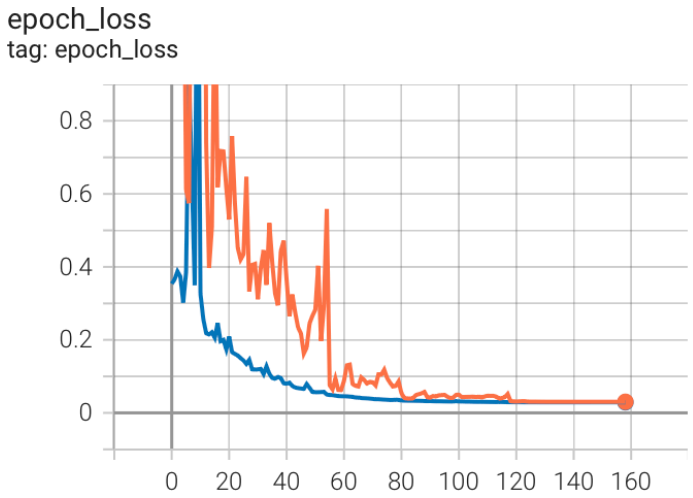
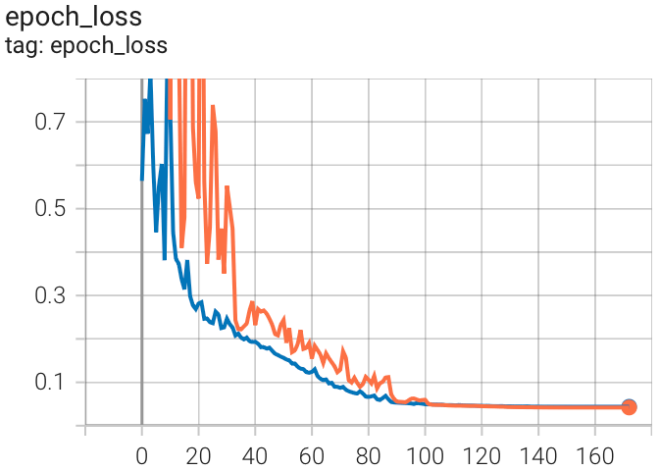
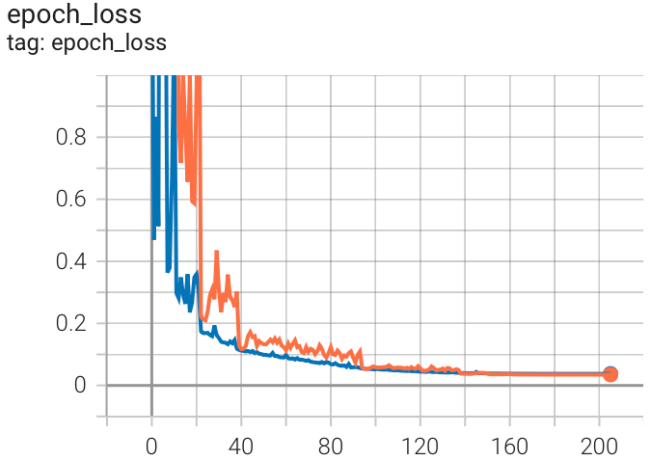
Model 2

$\phi(256, 512, 1024)$, $F(256, 128, 64)$



Duke

PFN Layer Dimensions

	Radiation-less Model	Radiation Model
Model 1		
Model 2		

Duke