Machine-Learning based Particle-Flow algorithm in CMS

Farouk Mokhtar (for the CMS Collaboration)

EPS-HEP 2025, Marseille
July 7th, 2025

What is Particle-flow Reconstruction?

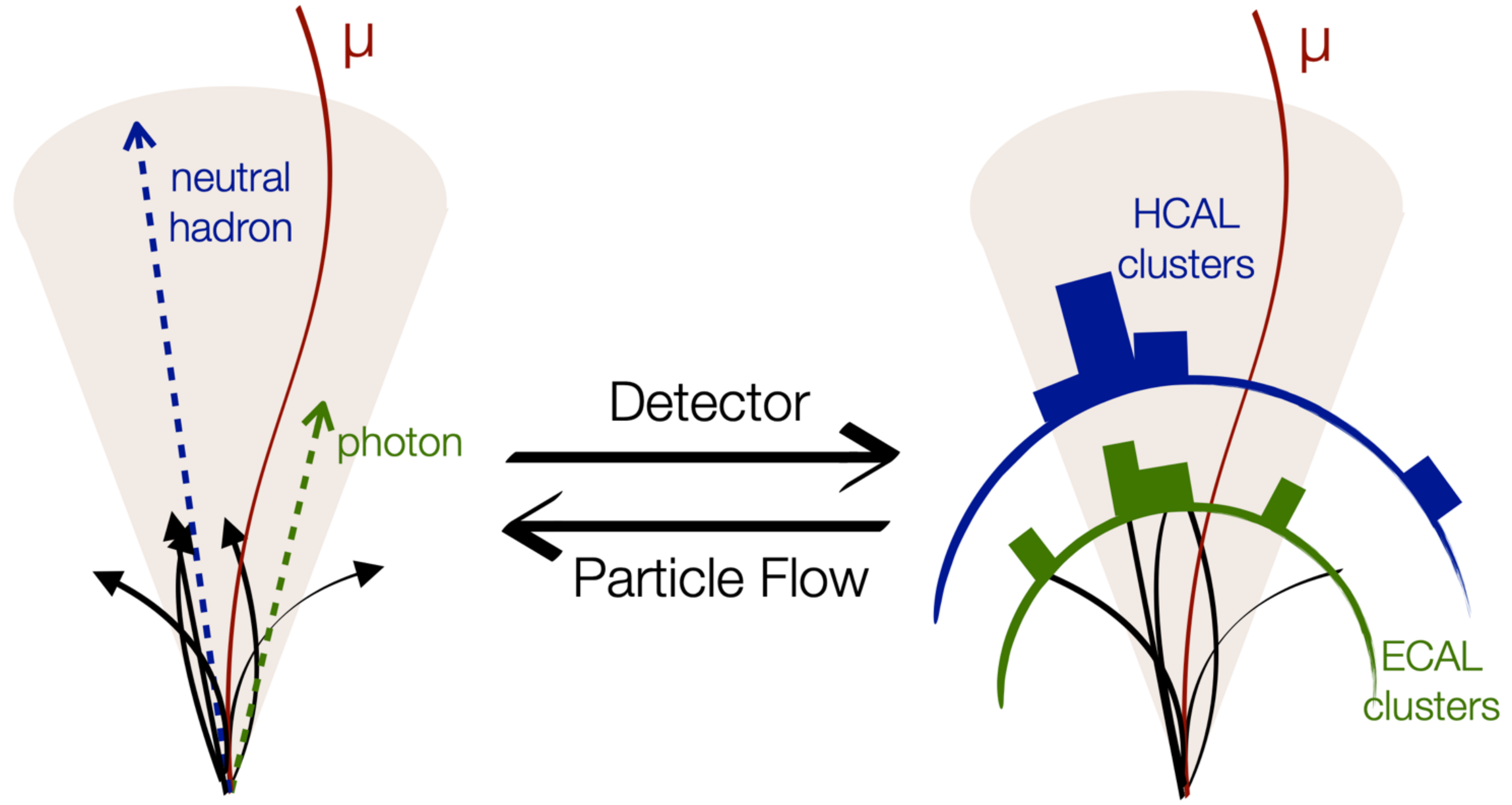
Relies on individual detector subsystems to reconstruct particles

- Reconstruction algorithms at the LHC fall under two categories: <u>local</u> and <u>global</u>

Combines information from multiple subsystems

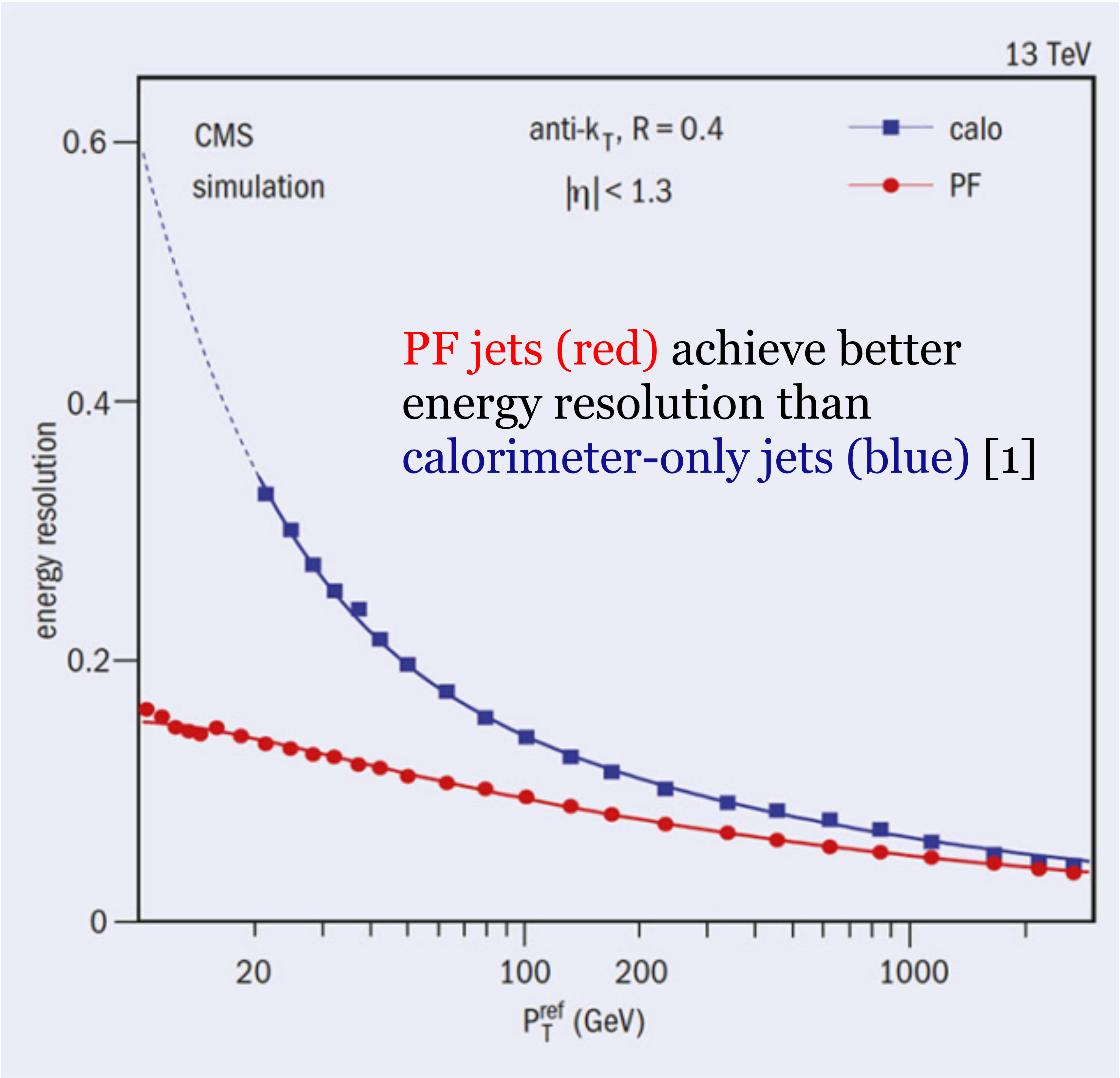
- Particle-flow (PF) is a **global** reconstruction algorithm that combines detector-level elements (e.g. tracks and clusters) to identify and reconstruct all stable particles in the event

- PF solves the inverse problem of detector simulation
  → A <u>complex</u> task with no simple algorithmic solution



[Ref: CERN-CMS-DP-2021-030](CERN-CMS-DP-2021-030)

Why Particle-flow?

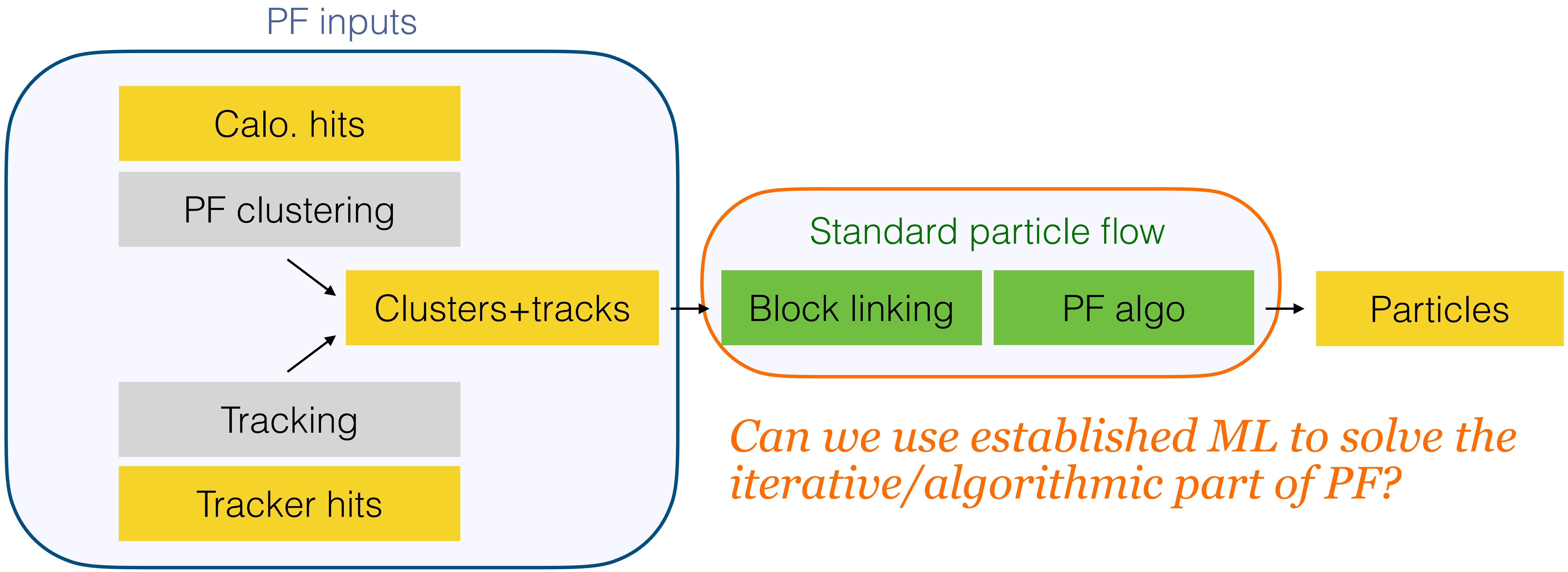


In CMS, PF algorithm is crucial for physics measurements—significantly improving jet energy resolution over local reconstruction algorithms

PF algorithm has been central to CMS analyses since LHC Run 1 era (2009–2013)

How does Particle-flow work?

- PF reconstructs particles by iteratively linking detector elements—tracks and calorimeter clusters—in a process known as block linking

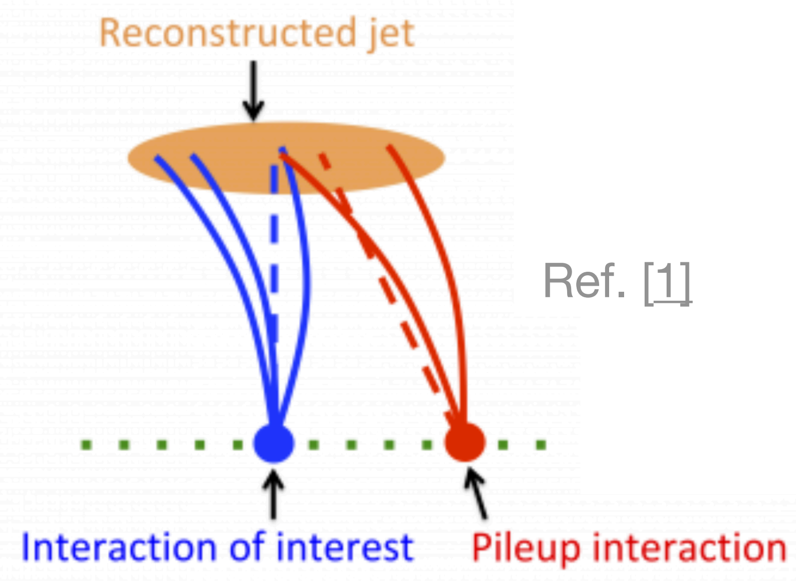


Can we use established ML to solve the iterative/algorithmic part of PF?

How about ML-based particle-flow?

- CMS is in a unique position to test ML for full event reconstruction

- We present **MLPF**, an end-to-end ML approach to PF block linking and particle reconstruction—**including ML-based per-particle pileup rejection**



Calo. hits

PF clustering

Tracking

Tracker hits

Clusters+tracks →

This work: MLPF

Block linking | PF algo | PUPPI

Standard particle flow

Pileup subtraction

→ Particles

MLPF replaces multiple hand-crafted steps with a single neural network

MLPF Summary and Goals

- 📈 Demonstrates **realistic event-level performance**
- 🛠 **Integrated in CMS software framework**
- 🛡 Includes **per-particle pileup (PU) mitigation**
- 🌐 **Generalizable** to new detector inputs or outputs → [1] MLPF for CLICdet [2] Fine-tuning MLPF for FCC
- ⚡ **Runs on GPUs** at ~40 ms/event
- 🧠 **Tested on data** for full event reconstruction

Datasets and training

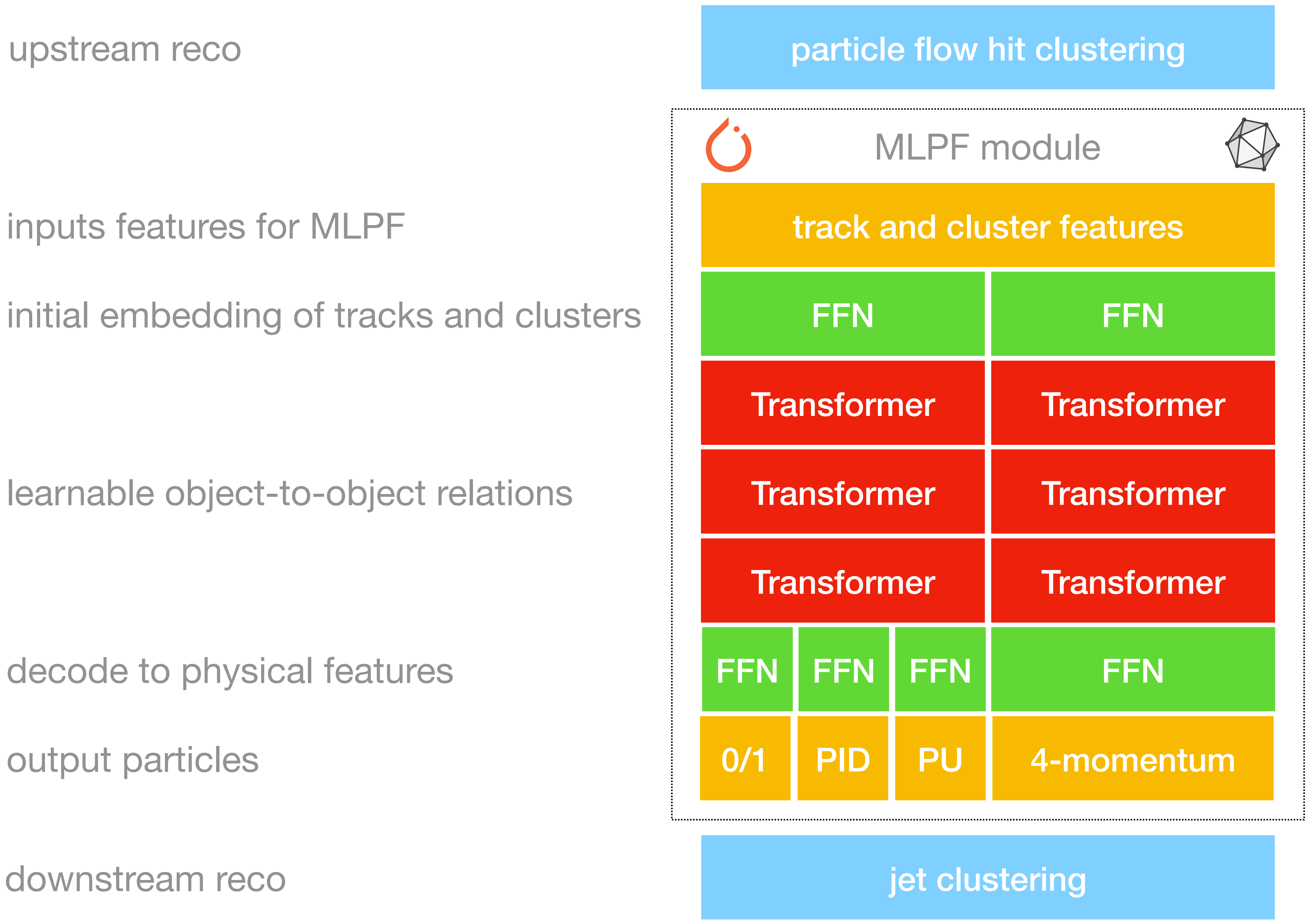
- We train the model in a **fully supervised** fashion using a standard PyTorch setup, and export the static compute graph to ONNX

- The model is small: **~4M parameters / 20 MB** and is trained on MC samples simulated under Run 3 (2022–2026) conditions

physics process	PU configuration	MC events
top quark-antiquark pairs	flat 55–75	500 k
QCD $\hat{p}_T \in [15, 3000]$ GeV	flat 55-75	500 k
$Z \to \tau\tau$ all-hadronic	flat 55–75	500 k
top quark-antiquark pairs	no PU	5 M
QCD $\hat{p}_T \in [15, 3000]$ GeV	no PU	5 M
$Z \to \tau\tau$ all-hadronic	no PU	5 M

Table 1: MC simulation samples used for optimizing the MLPF model.

MLPF architecture: end-to-end particle reconstruction



Transformer model processes embedded track and cluster features

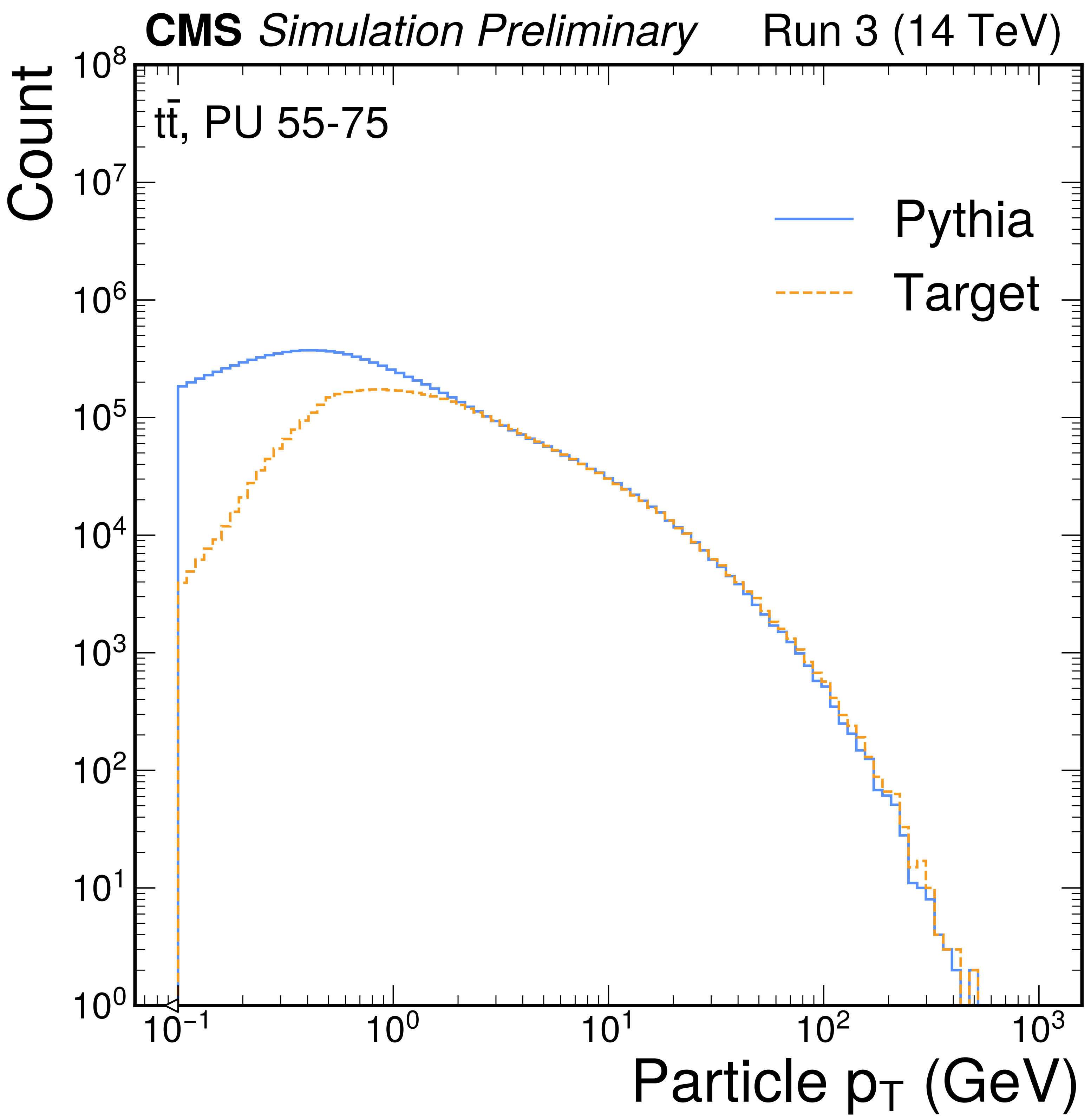
Outputs include: particle ID, 4-momentum, and per-particle pileup score

We compare the model output to a **particle-level target** using a per-particle loss function (more on this in the next slides)

Simulation-based *target*

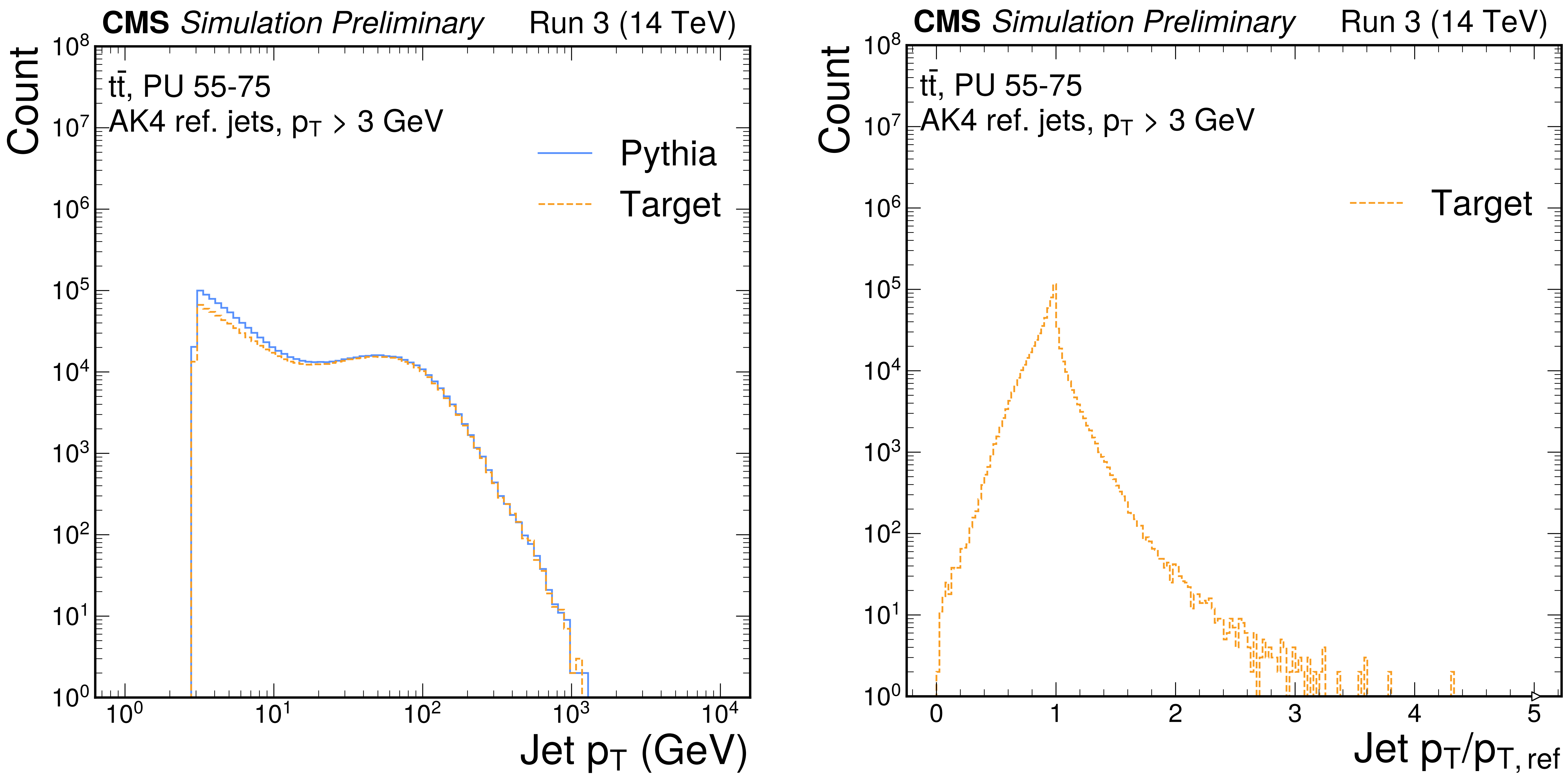
What set of particles should a particle-flow algorithm aim to reconstruct?

- Not all Pythia stable particles are **directly reconstructible—that is, particles that leave detectable signals in the detector**
- We use generator + simulation information to define a set of *target* particles that interact with the detector either directly or through their descendants
- We cross-check the *target* against stable Pythia particles
- We also define an energy-weighted PU fraction per particle (typically 0 or 1) which can be used to subtract pileup contributions



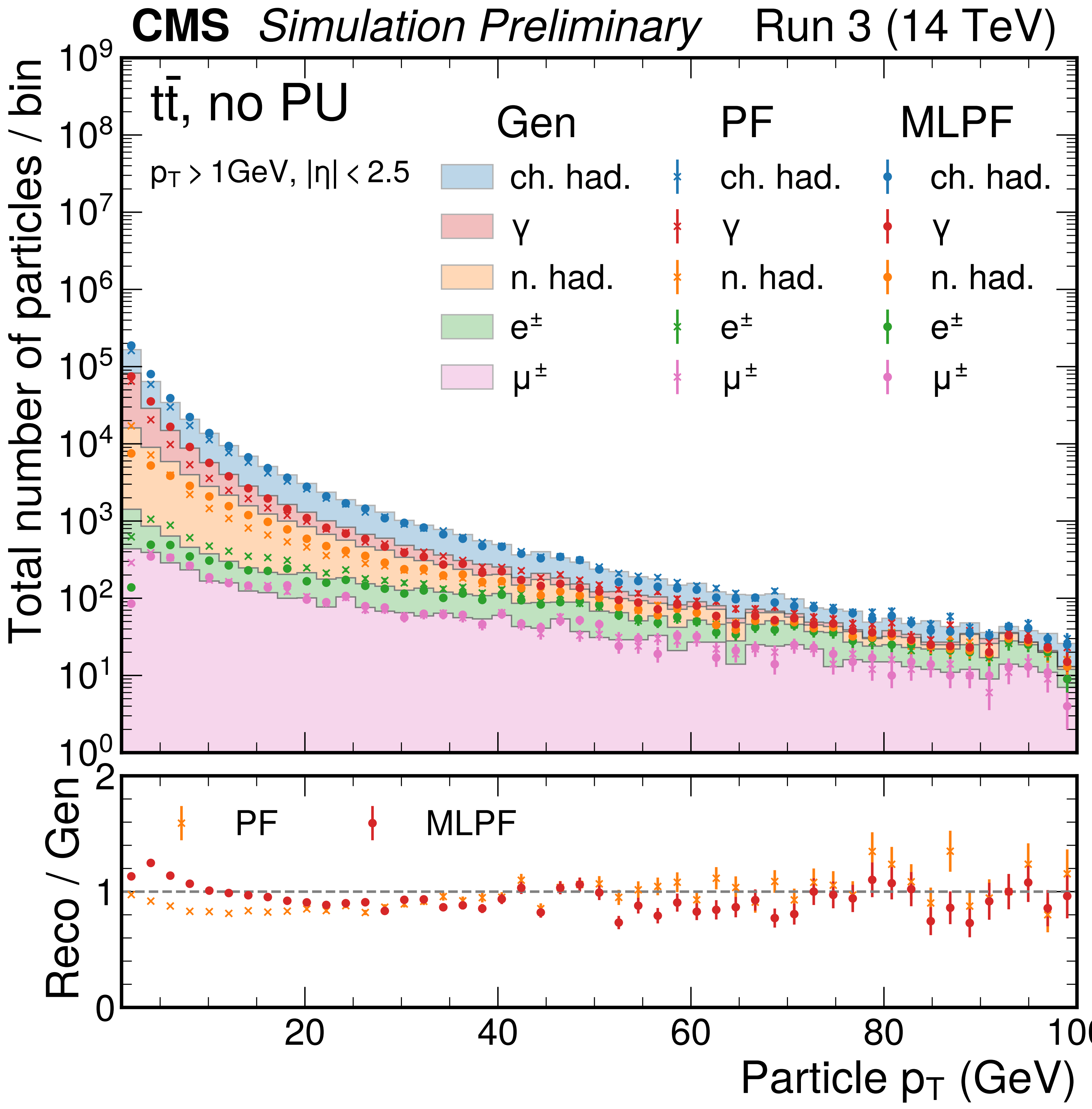
Residual low p_T disagreement is driven by reconstruction and simulation acceptance

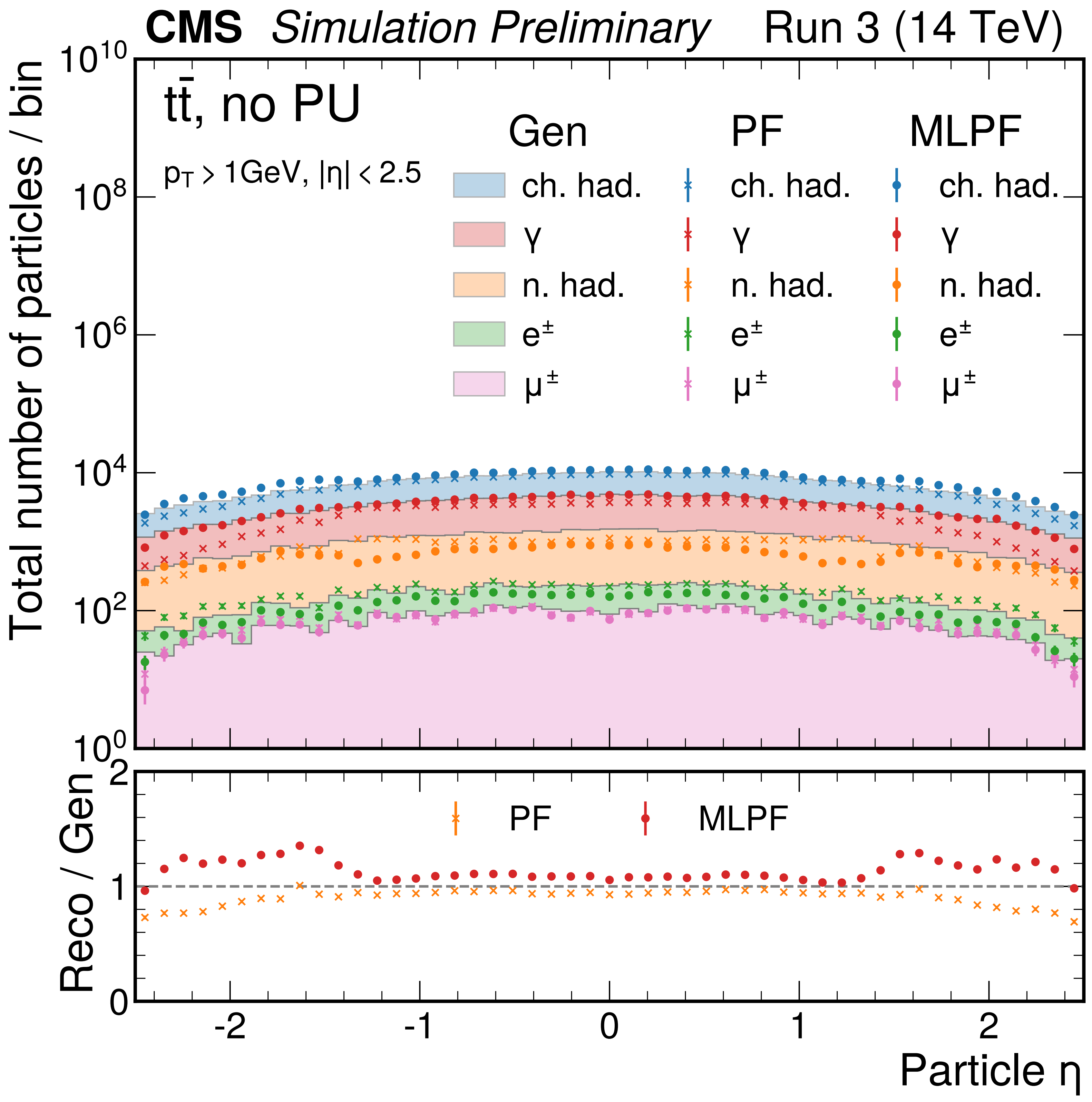
Next, we cluster jets from the simulation-level *target* and validate them against generator-level reference jets



Jets clustered from *target* particles closely match reference generator-level jets

Single-particle reconstruction performance in $t\bar{t}$ events without pileup

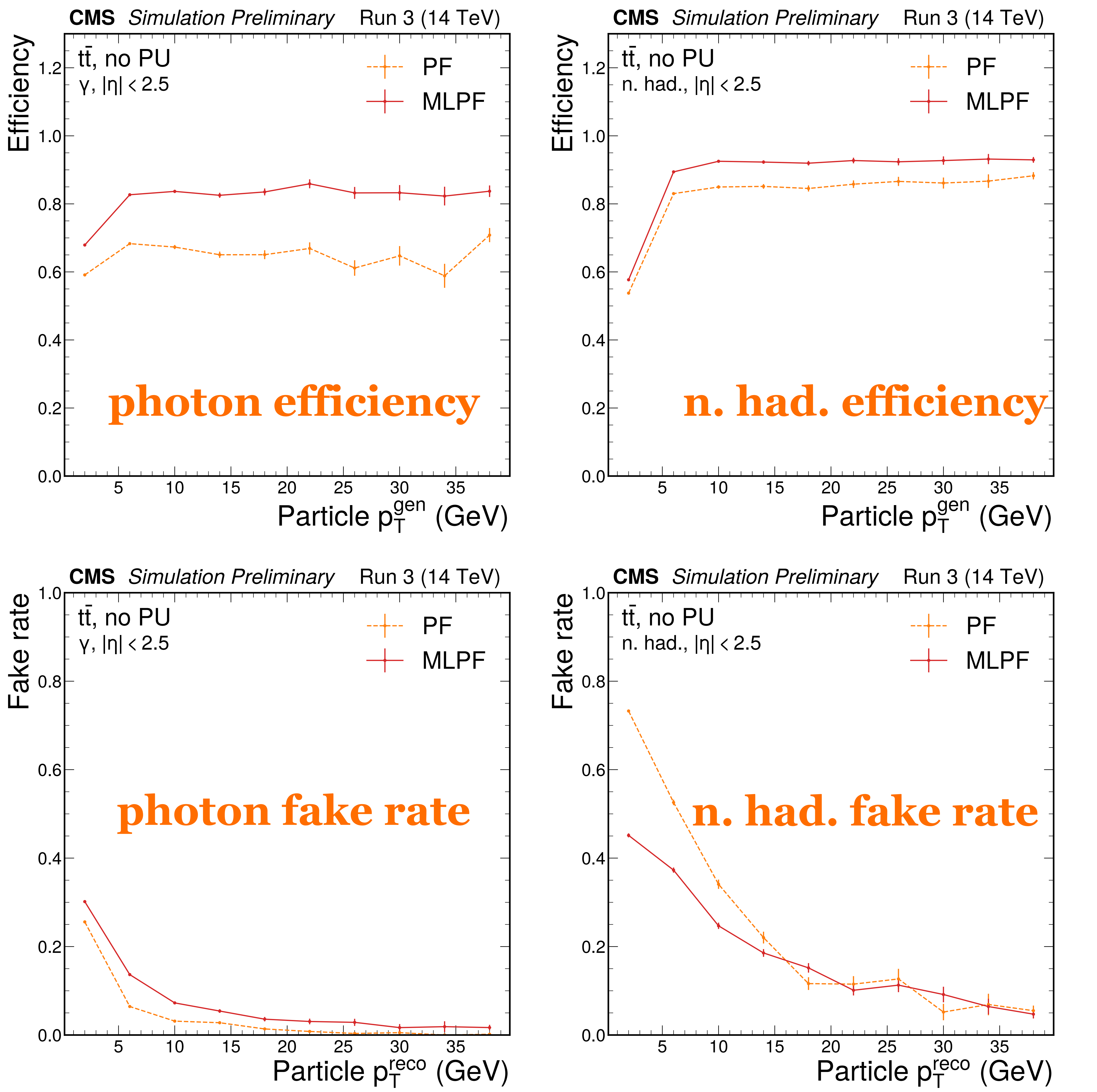




MLPF shows realistic particle-level performance

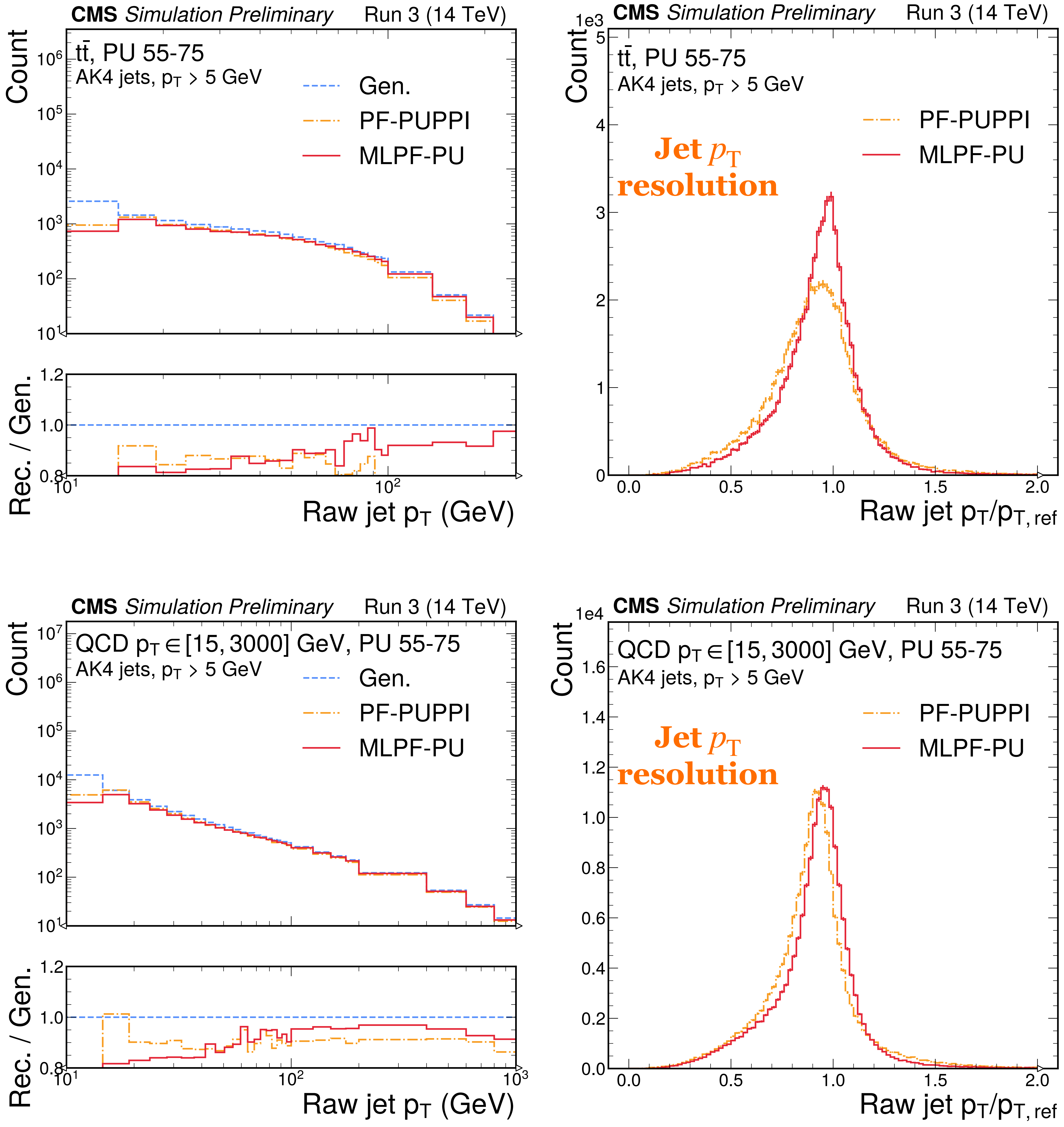
Particle efficiency and fake rate

- We define efficiency and fake rate by first associating generator-level particles to reco particles using $\Delta R < 0.15$ matching
- MLPF improves photon efficiency, but also slightly increases the fake rate
- **MLPF achieves higher reconstruction efficiency for neutral hadrons while maintaining the same fake rate**

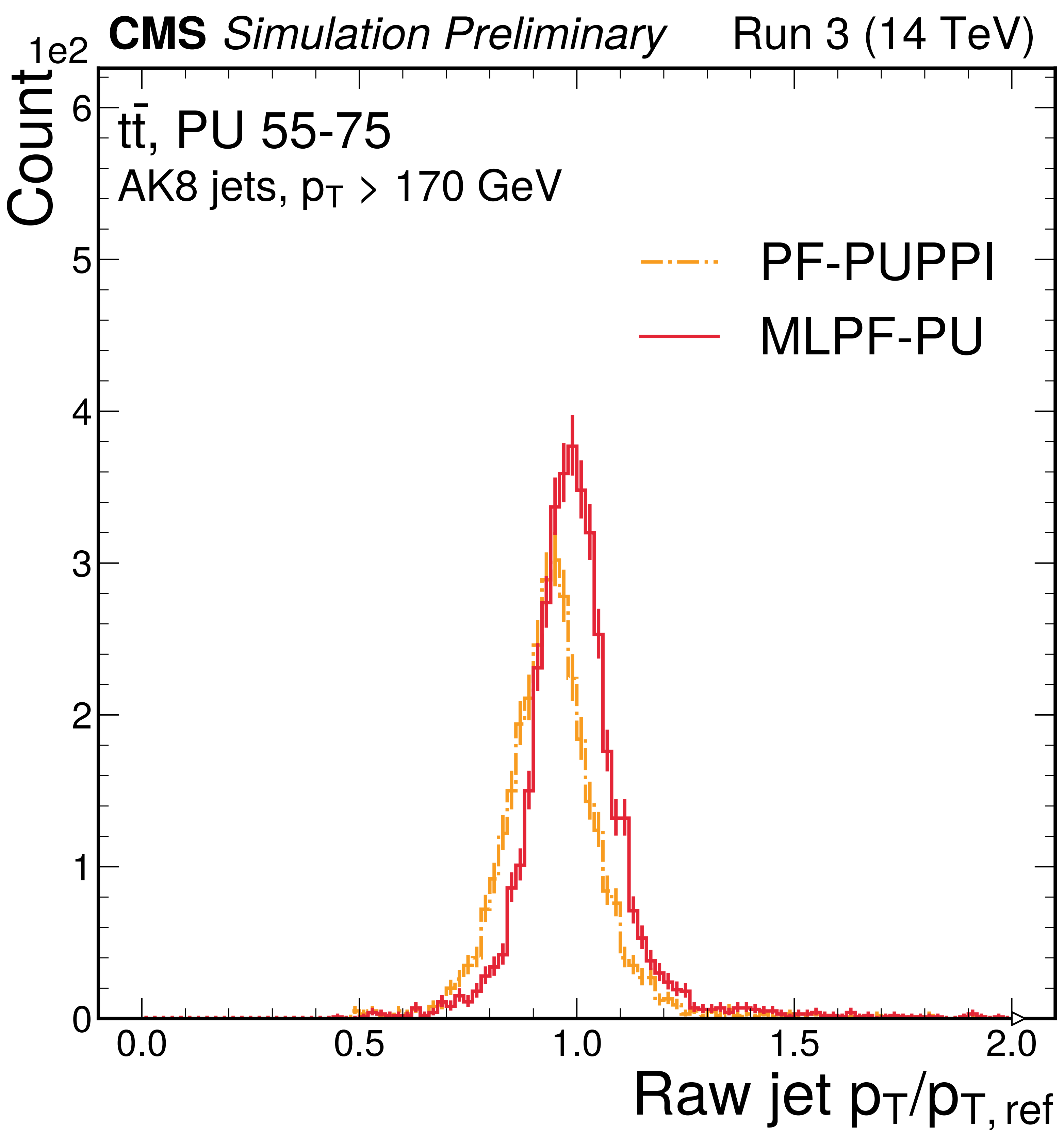


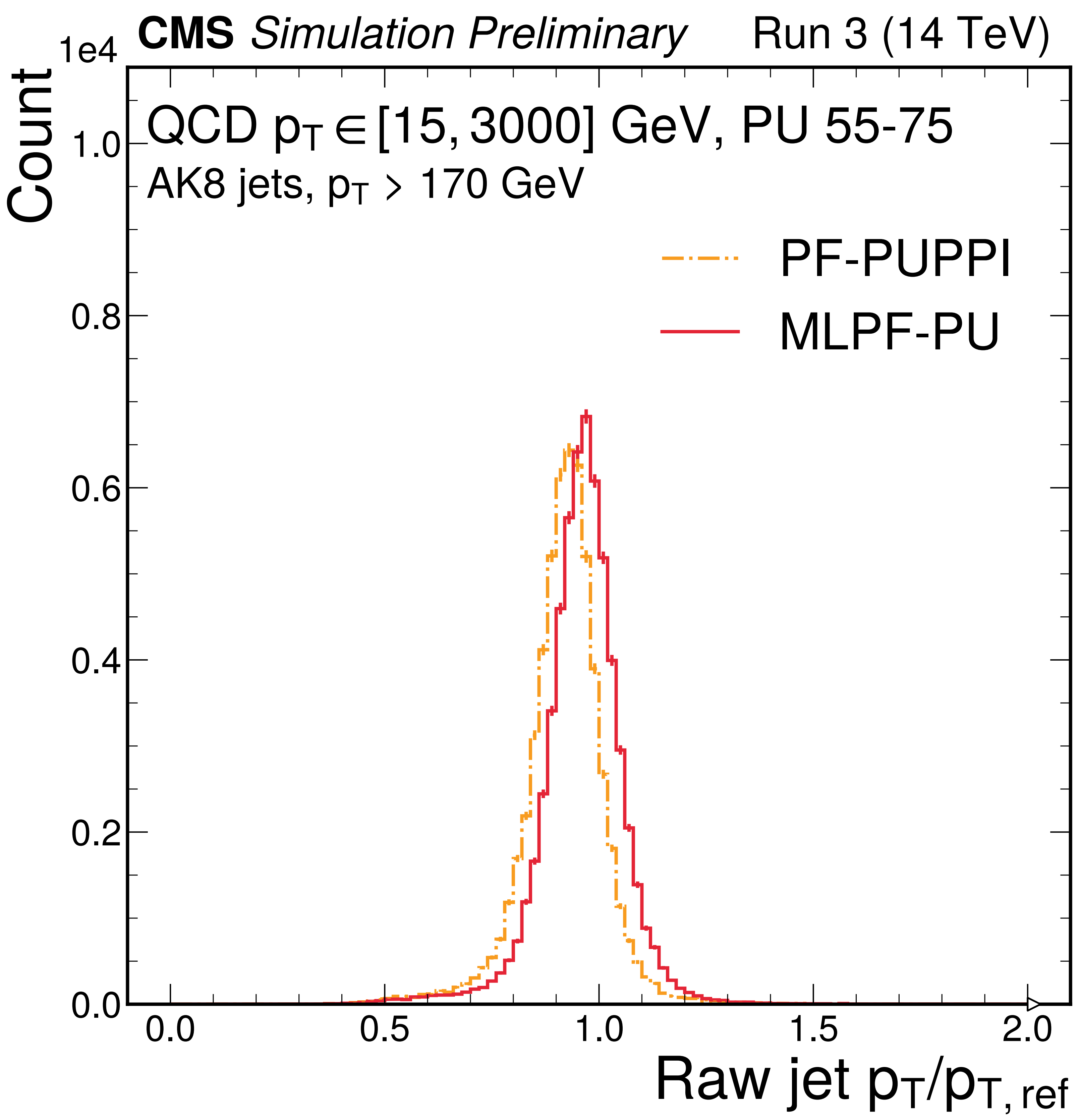
Anti-kT R=0.4 jets

- **We evaluate jet reconstruction performance in $t\bar{t}$ (top) and QCD (bottom) samples with pileup**
- We show the raw jet p_T before any corrections
- **In PF + PUPPI:** jets are built from PF candidates with PUPPI applied for pileup mitigation
- **In MLPF:** pileup subtraction uses the per-particle pileup predictions from the MLPF model
- *Note that jet reconstruction was never explicitly trained against with MLPF*



Anti-kT R=0.8 (a.k.a. large-radius jets)



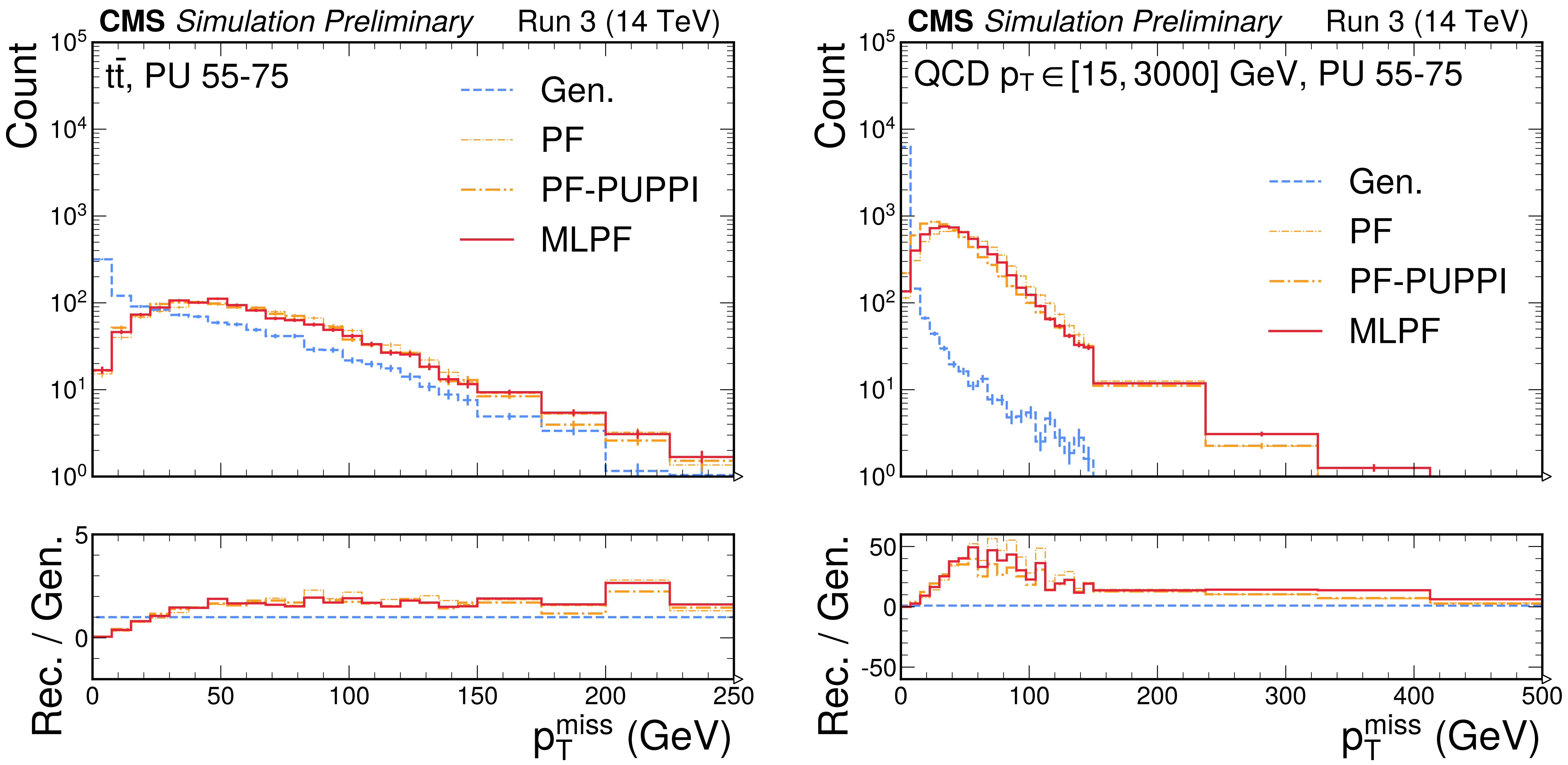


MLPF also provides excellent jet p_T resolution in the boosted regime across both $t\bar{t}$ (left) and QCD (right) samples

No corrections applied on the reconstructed distributions

Missing transverse momentum

- We define p_T^{miss} as the negative vectorial sum of reconstructed particle p_T
- Generator-level p_T^{miss} differs from reconstructable p_T^{miss} due to fiducial cuts in the simulation and reconstruction, and pileup contamination in the samples



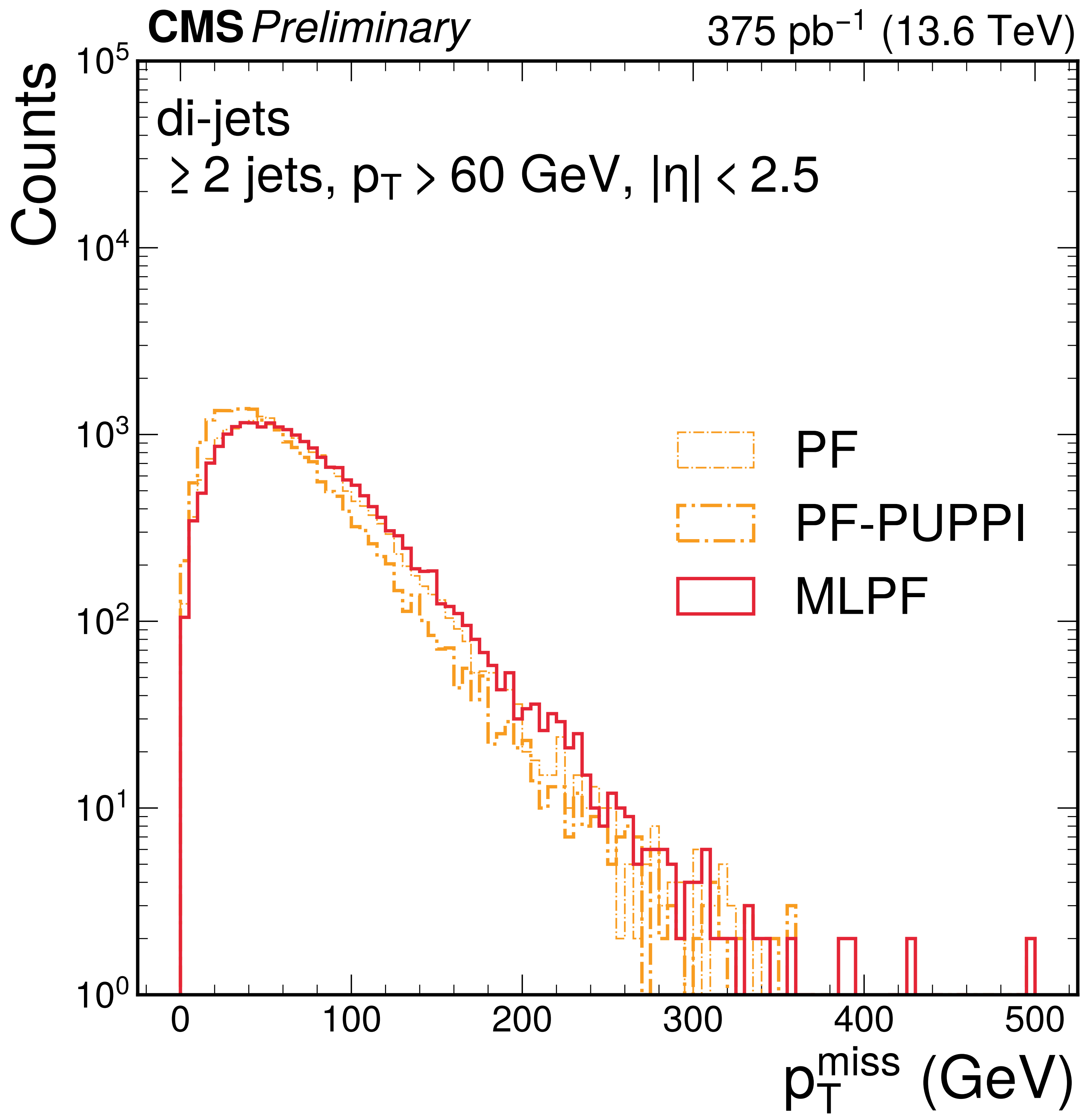
In both $t\bar{t}$ and QCD samples, PF and MLPF are consistent

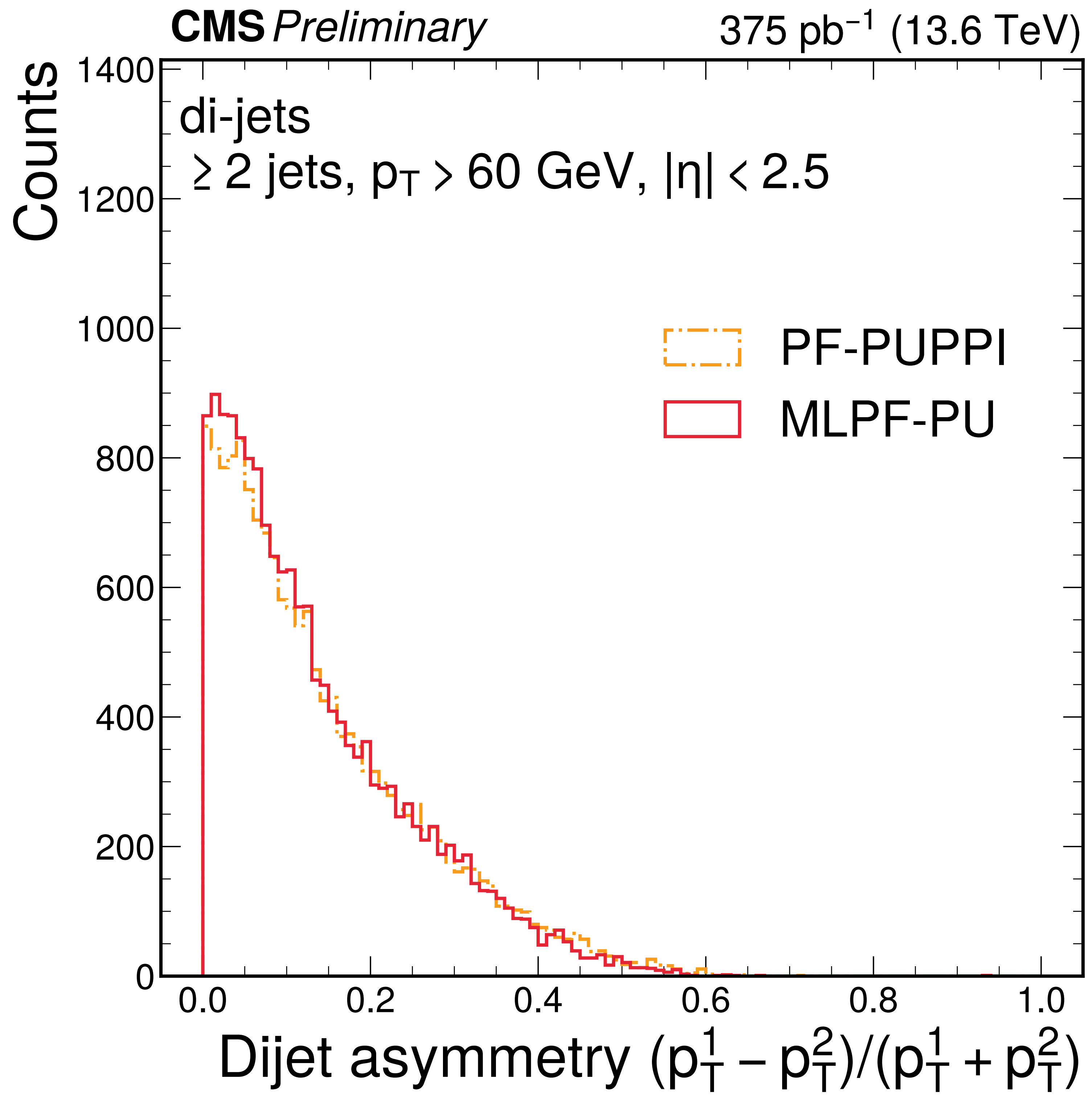
Note that p_T^{miss} was not explicitly included in the loss function when training MLPF

No JECs or other calibrations applied

Commissioning on CMS data

- We study p_T^{miss} and dijet p_T asymmetry in a subset of 2024 CMS data



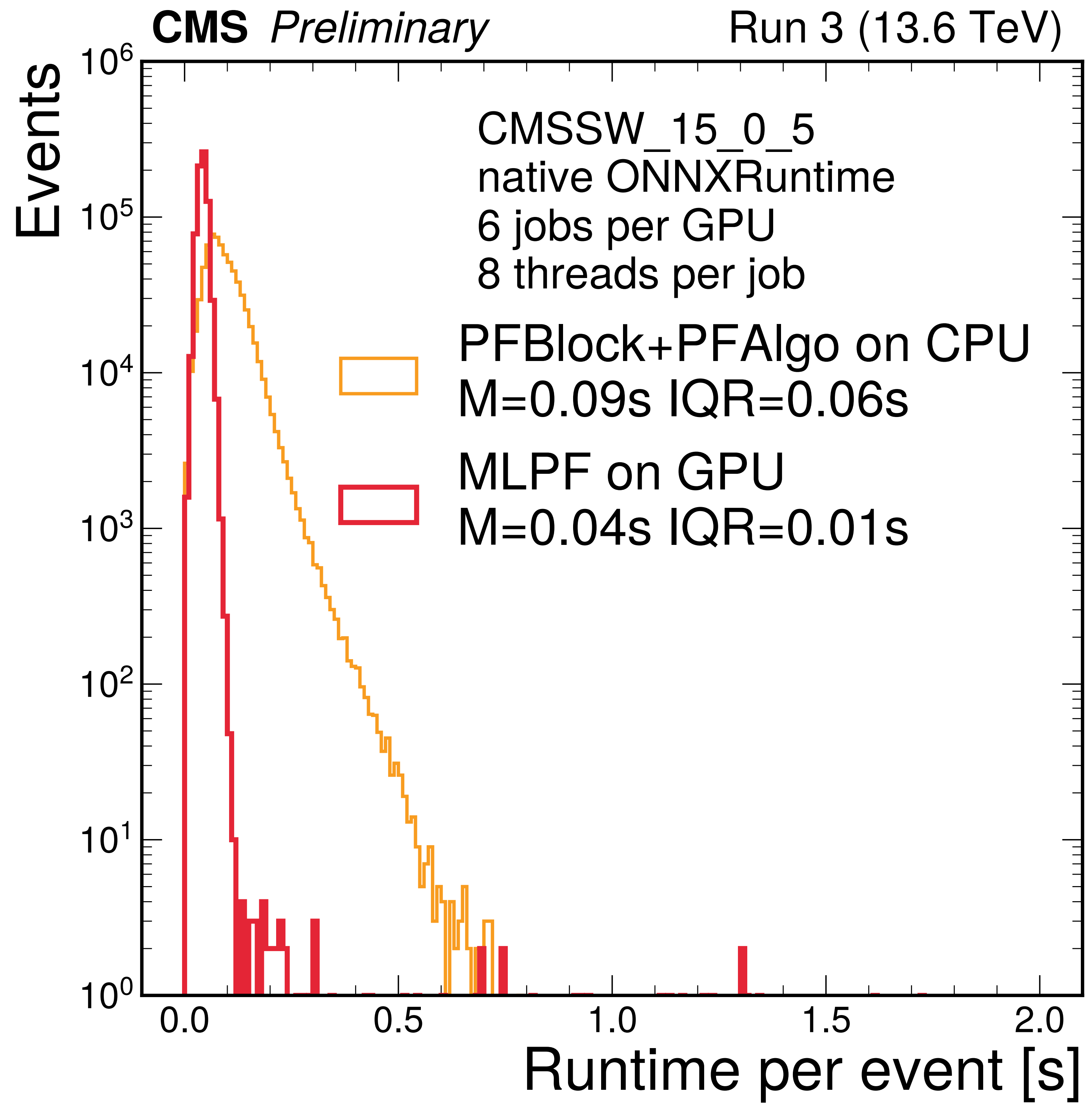


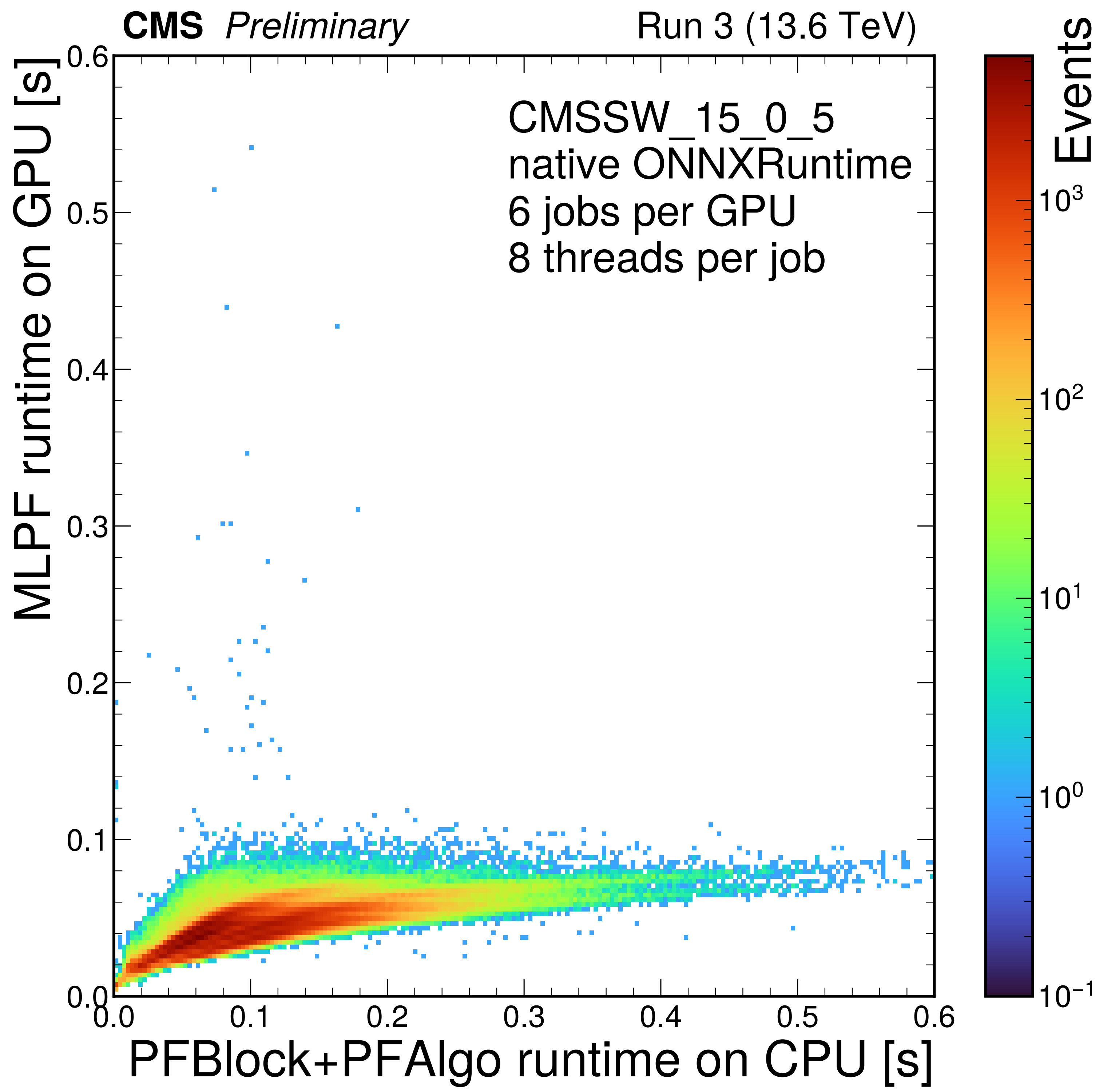
🚀 **First application of MLPF to CMS data**

Jets are required to pass jet ID criteria based on their hadronic, electromagnetic, and muon energy fractions, suppressing jets from noise or muons

Fast and Scalable MLPF Inference

- Baseline PF (CPU): Block linking + PFAlgo vs. MLPF (GPU) using ONNX RUNTIME with 1/7 of an A100 GPU (48 streams total)





MLPF achieves faster and flatter runtime scaling compared to baseline PF

Summary & Outlook

🧠 **ML-based Particle Flow (MLPF)** reconstruction algorithm can be optimized on MC simulation using **supervised learning**

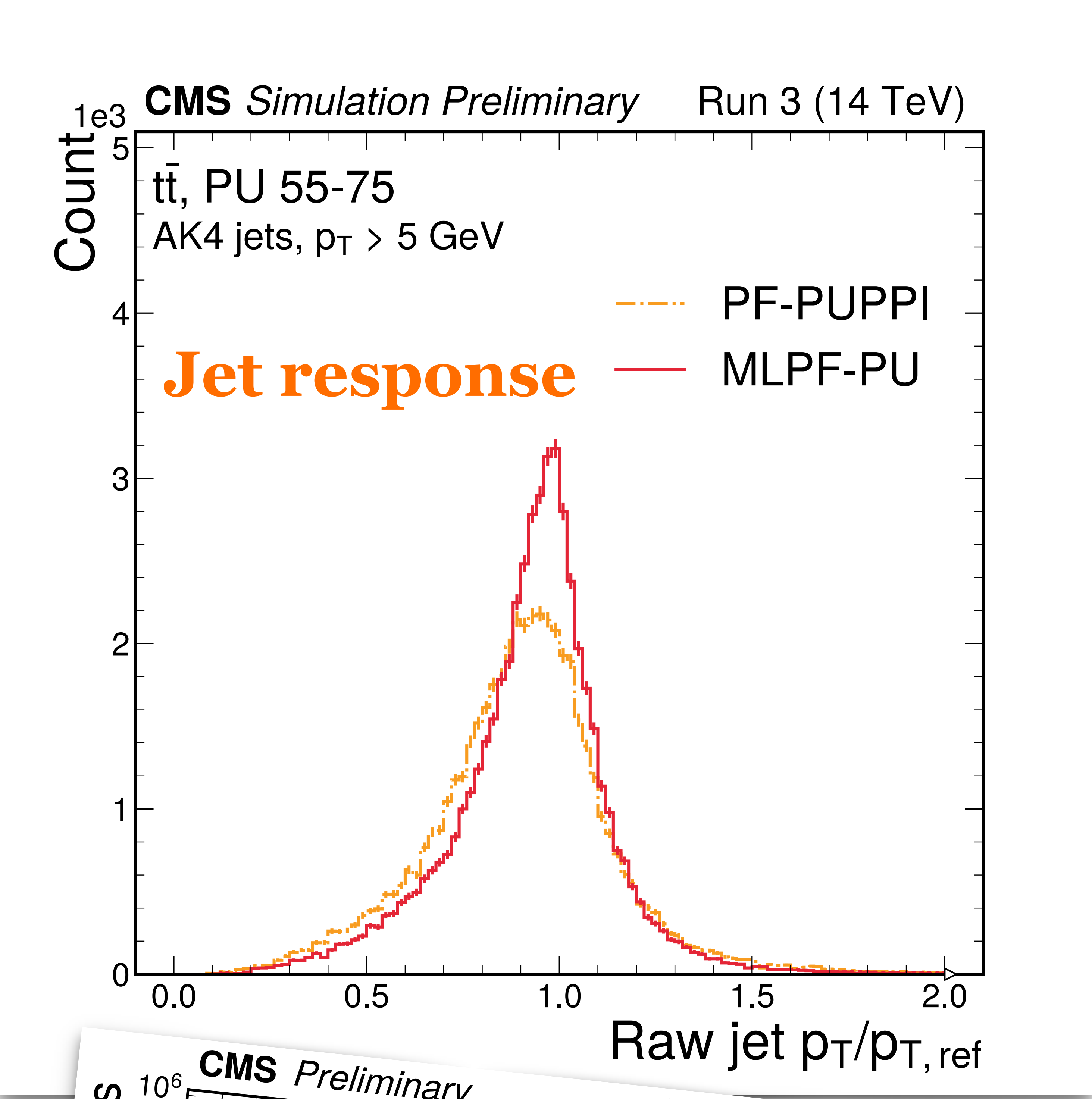
📈 **Comparable physics performance** to standard PF — with improved jet performance when using per-particle pileup rejection

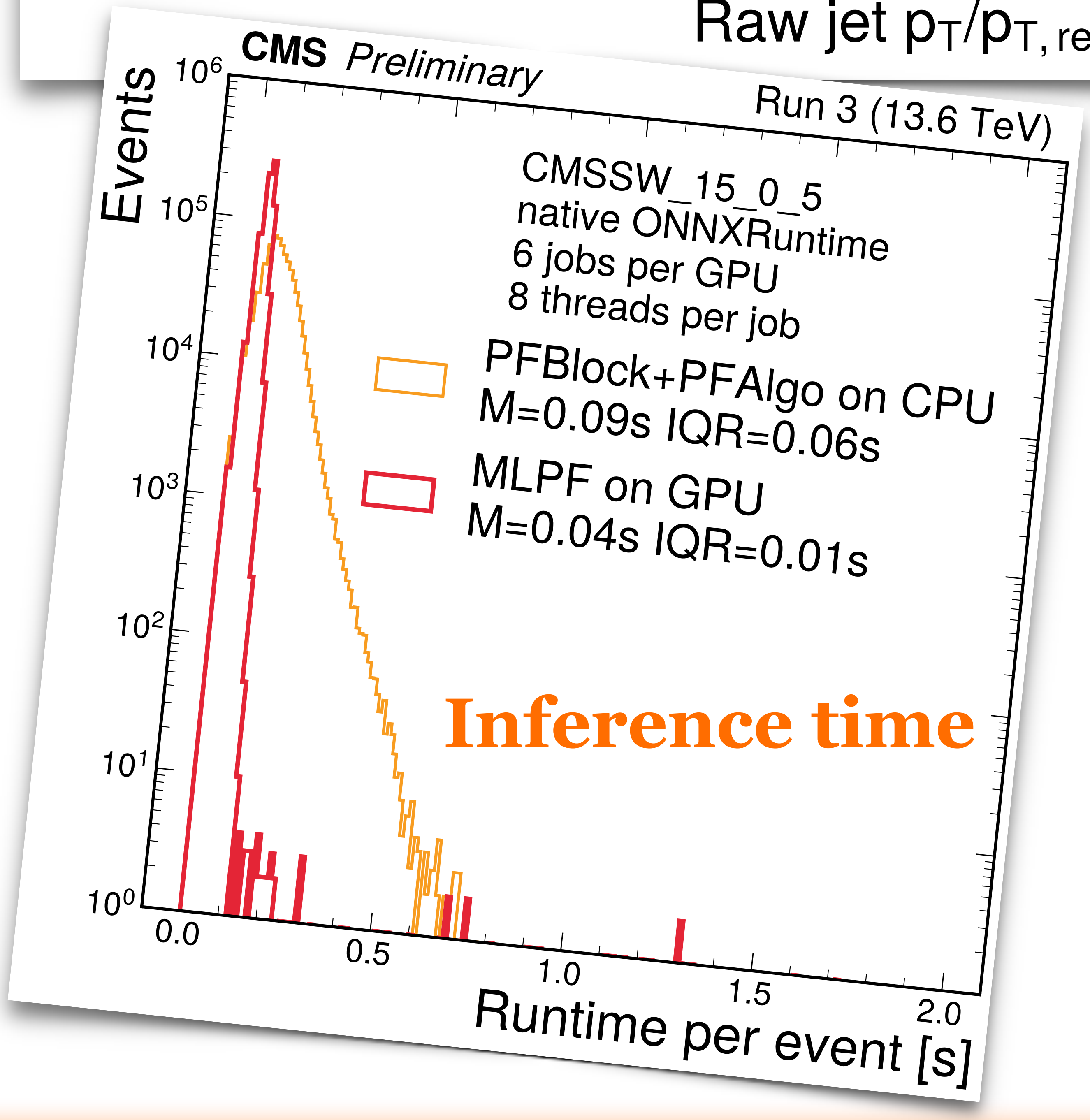
🔍 Initial **commissioning studies on 2024 CMS data** show good agreement in dijet / p_T^{miss} distributions

⚙️ The model **can be integrated in CMS software and runs on GPU** achieving **~40 ms/event** on GPU (A100, 48 streams)

⏳ Further conclusions will require high-statistics data and detailed MC validation

DP note ref: [CMS-DP-2025-033](CMS-DP-2025-033)



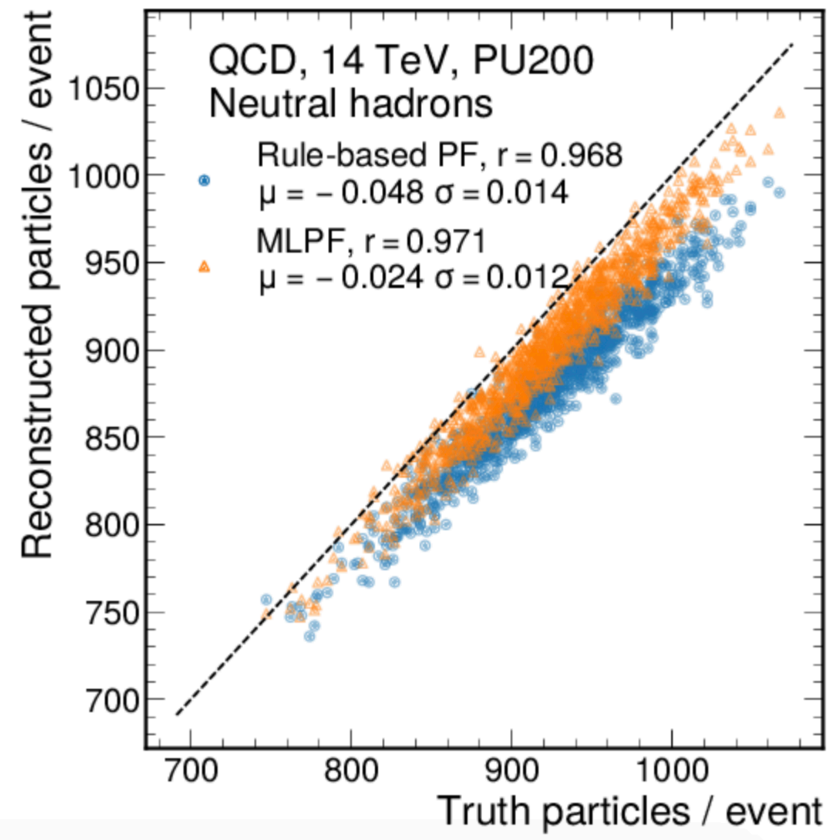


Backup

MLPF History and Timeline

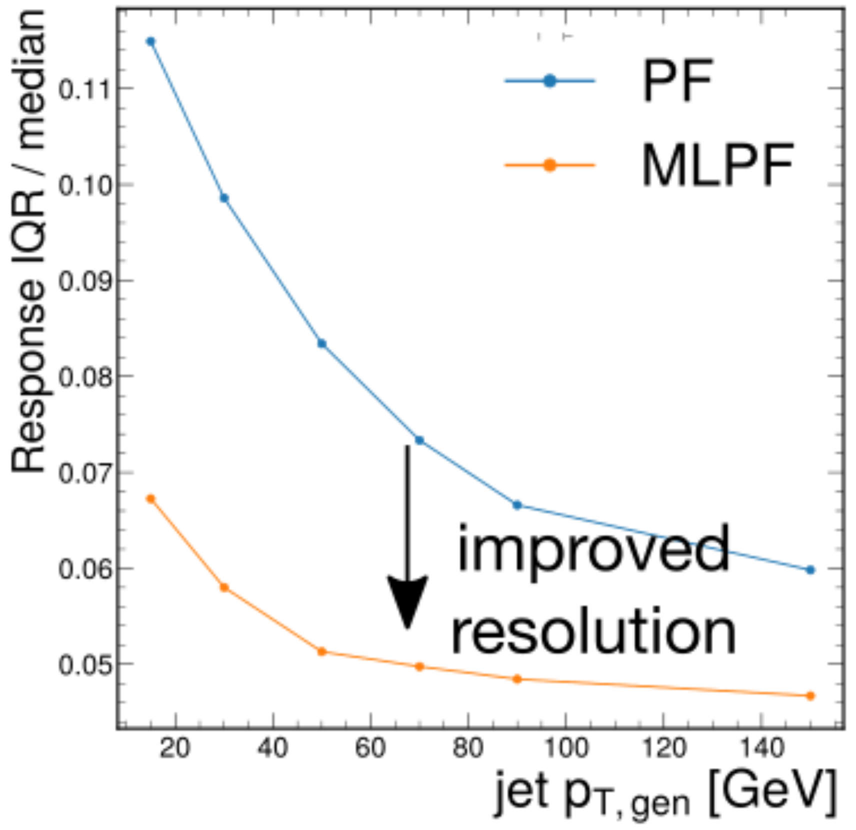
Open Datasets

Proof of concept on DELPHES



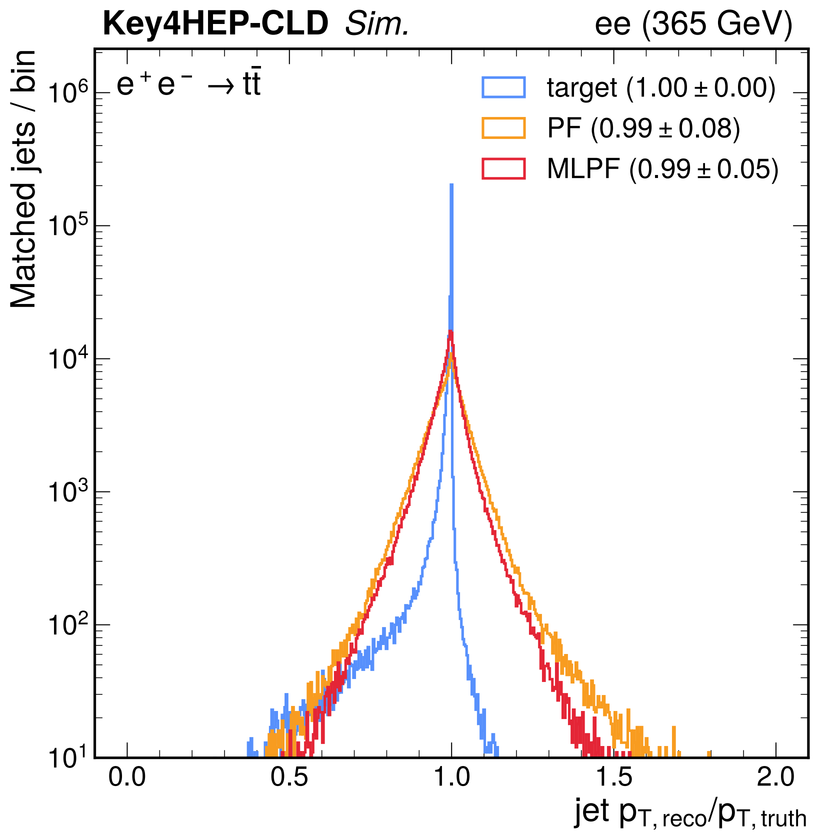
Eur. J. Phys.

CLIC Full-Sim



Commun. Phys.

CLD Full-Sim

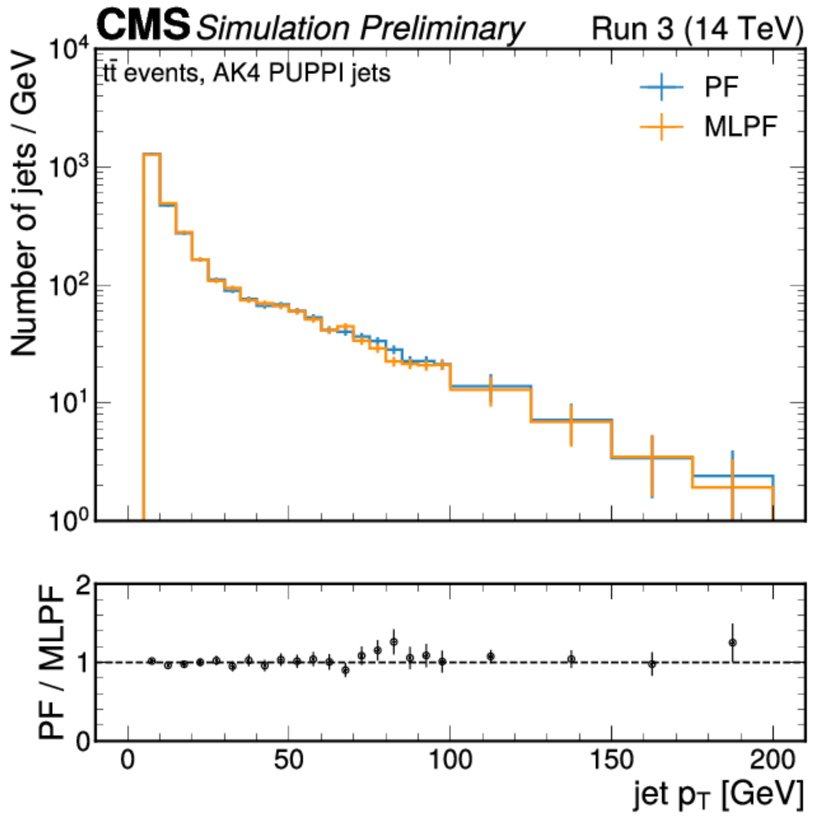


Phys. Rev. D

2020 2021 2022 2023 2024 **2025**

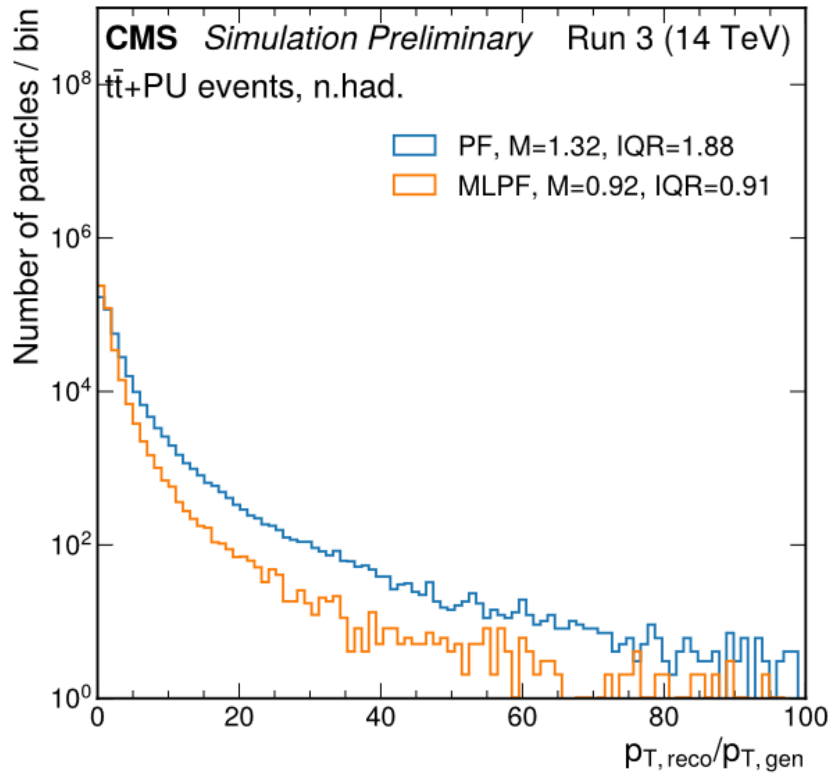
CMS Full-Sim

Target definition = CMS PF



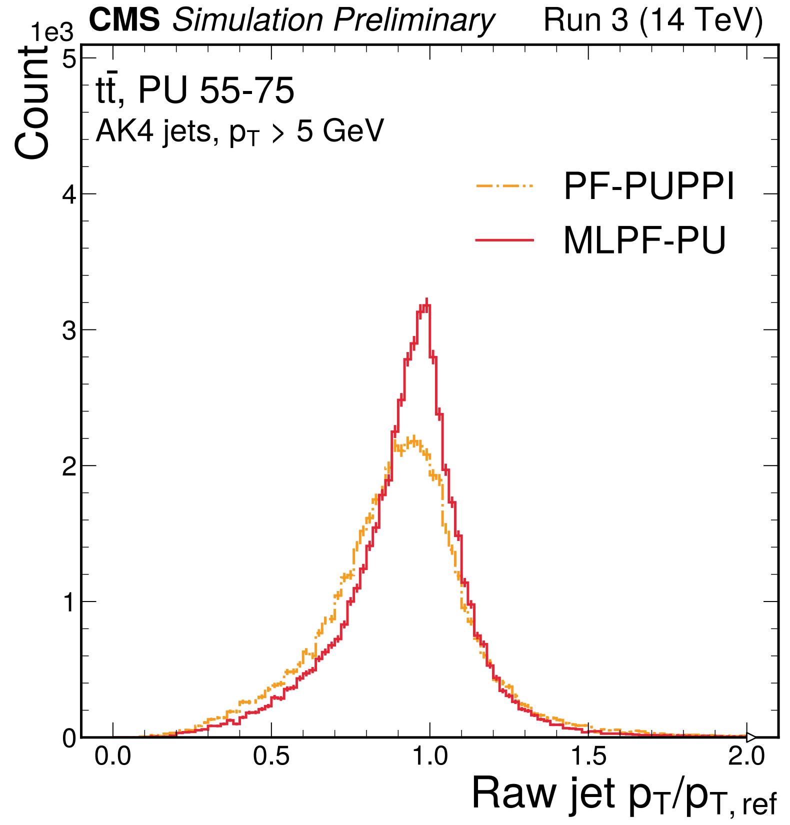
CERN-CMS-DP-2021-030

Adopted sim-based target definition



CERN-CMS-DP-2022-061

⏳ CMS paper



CMS-DP-2025-033

🚀 FlashAttention

- **Memory Efficient**
  - Standard attention computes and stores the full attention matrix in memory, which scales as $O(n^2)$ in sequence length
  - **FlashAttention** avoids storing the full attention matrix as it computes the softmax and the weighted sum in **fused blocks**
- **Speed (up to 2×–4× faster than standard attention on large sequences)**
  - Fused kernels (combine multiple operations into one GPU A100/H100 kernel to reduce memory bandwidth bottlenecks)

Enables training with longer context lengths (e.g. 4K, 8K tokens) that would otherwise cause out-of-memory errors in vanilla attention

Neutral hadron performance

- MLPF achieves higher reconstruction efficiency with a slight increase in fake rate in the forward region due to **its looser working point**

The working point can be optimized in future work!