Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

CLUSTER OF EXCELLENCE
QUANTUM UNIVERSE

FSP CMS
Erforschung von Universum und Materie

gefördert durch das
Bundesministerium für Bildung und Forschung

CMS

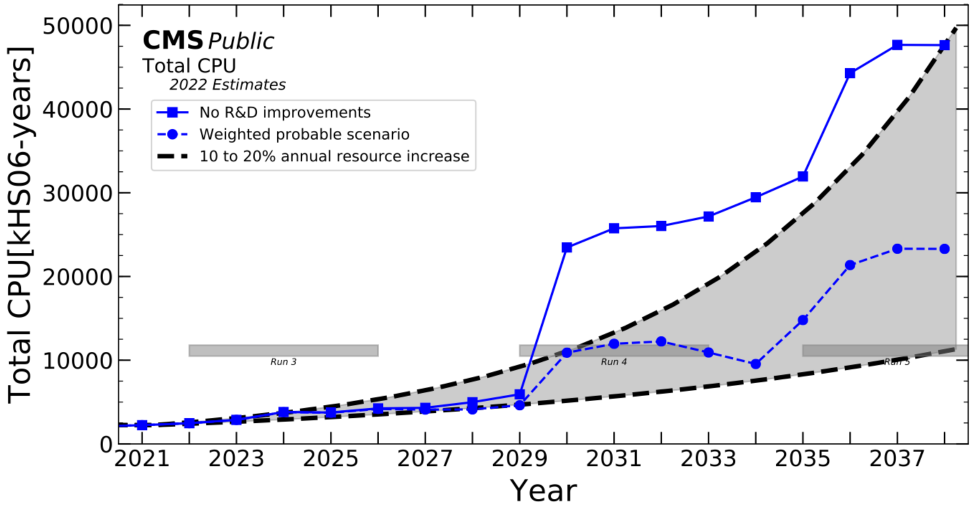
NOVEMBER 6, 2023 — ML4JETS

REGRESSION-BASED REFINEMENT OF FAST SIMULATION

SAMUEL BEIN[1], PATRICK CONNOR[1,2], KEVIN PEDRO[3], PETER SCHLEPER[1], MORITZ WOLF[1]

[1]UNIVERSITÄT HAMBURG
[2]CENTER FOR DATA AND COMPUTING IN NATURAL SCIENCES
[3]FERMI NATIONAL ACCELERATOR LABORATORY

CMS-NOTE-2022-008



INTRODUCTION

- At the LHC, **fast simulation** techniques are crucial to cope with the computing challenges of the HighLumi phase
- In **CMS**, two simulation chains (FullSim/FastSim) are used which produce output of same dimensionality/structure:

	GEN → Event generation	SIM → Detector simulation	DIGI → Digitization	RECO → Reconstruction	(Mini/Nano)AOD Analysis format
FullSim	same e.g., MadGraph	GEANT4	same	analyze as if data	same
FastSim ≈ 15% of sim. events		parametrized energy loss → 100x faster		use GEN info → 5-50x faster	

➢ In total, FastSim ≈ **10x faster** than FullSim and generally in good agreement

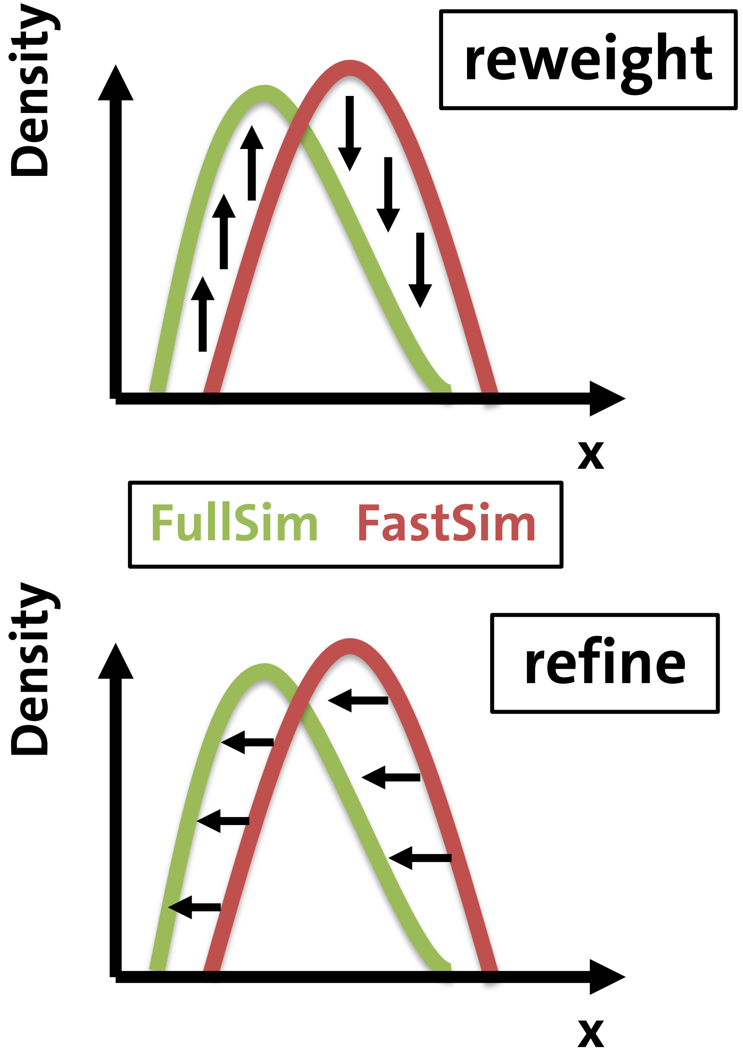
➢ But: FastSim/FullSim **discrepancies up to 20%** in some analysis observables

INTRODUCTION

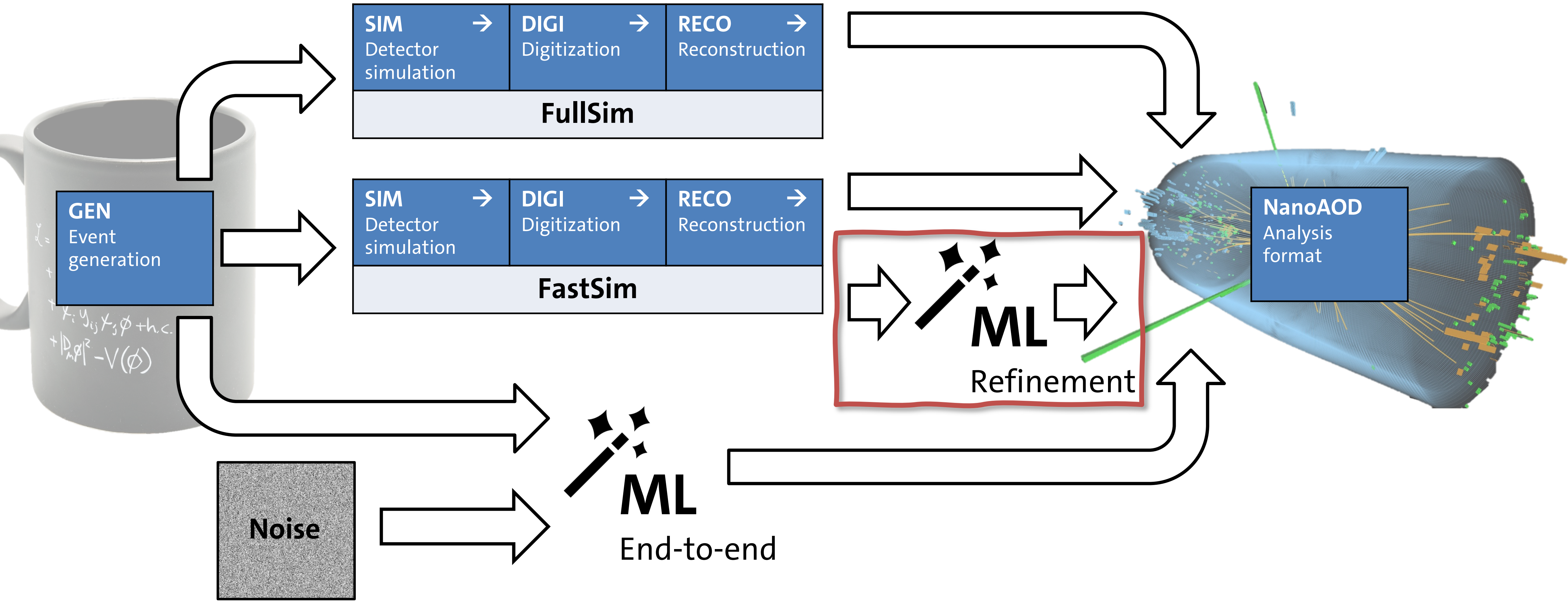
GEN →	SIM →	DIGI →	RECO →	NanoAOD
Event generation	Detector simulation	Digitization	Reconstruction	Analysis format

How to **improve FastSim accuracy** (i.e. agreement with FullSim)?

- Internal tuning of functions/parameters (within SIM/RECO)
- Post-hoc tuning (e.g., on top of NanoAOD)
  - **Reweighting** = defining weights for individual events/objects/… e.g., DCTR 1907.08209
  - **Refining / morphing** = changing (high-level) observables, multiple ML approaches e.g.,
    - Wasserstein-GAN 1802.03325
    - Diffusion model 2308.03876
    - Normalizing flows 2309.15912 2309.06472
    - Regression (fully-connected NN)



INTRODUCTION

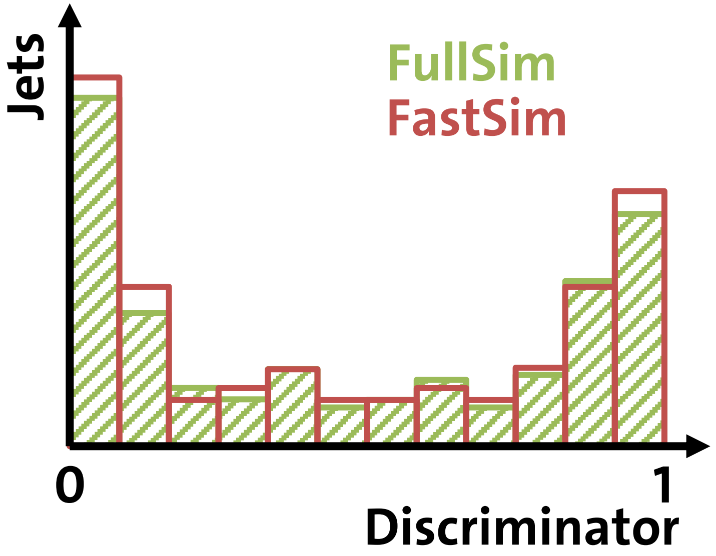


DATASET

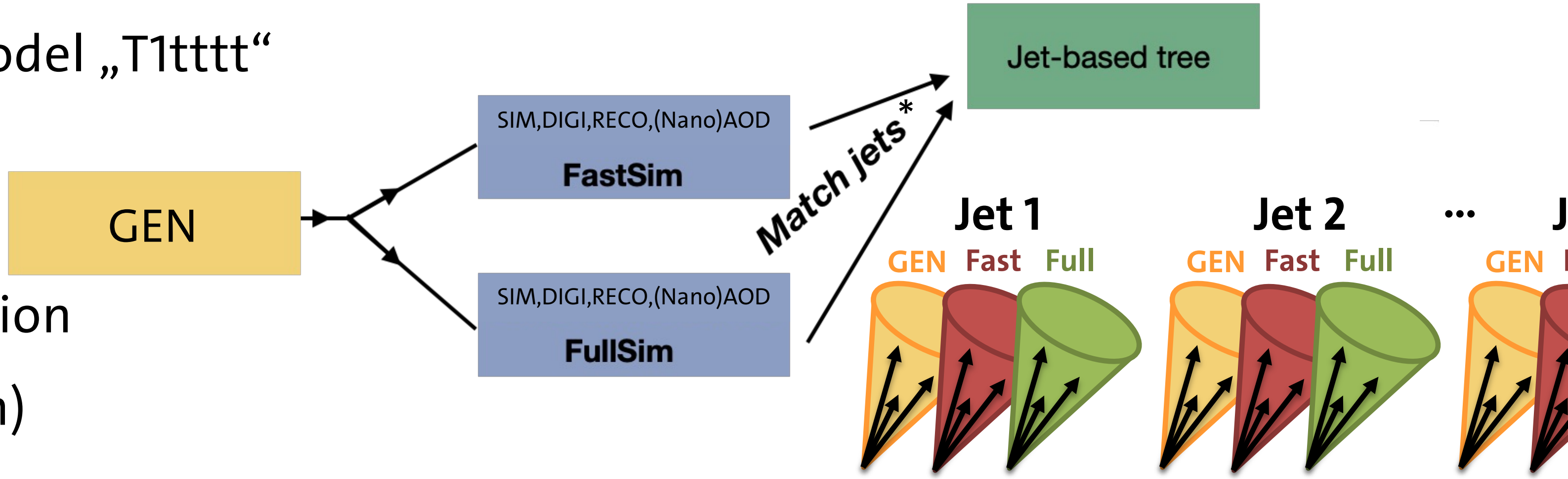
- Known shortcoming of FastSim due to the lack of fake tracks: **jet flavour tagging** observables too optimistic
  - ➢ Focus on refinement of 4 NanoAOD **DeepJet discriminators**:

$$\mathrm{B} = \mathrm{b} + \mathrm{bb} + \mathrm{lepb}\,,\ \mathrm{CvB} = \frac{\mathrm{c}}{\mathrm{c} + \mathrm{b} + \mathrm{bb} + \mathrm{lepb}}\,,\ \mathrm{CvL} = \frac{\mathrm{c}}{\mathrm{c} + \mathrm{uds} + \mathrm{g}}\,,\ \mathrm{QG} = \frac{\mathrm{g}}{\mathrm{g} + \mathrm{uds}}$$

(with DeepJet softmax output nodes: b, bb, lepb, c, uds, g; arXiv:2008.10519)



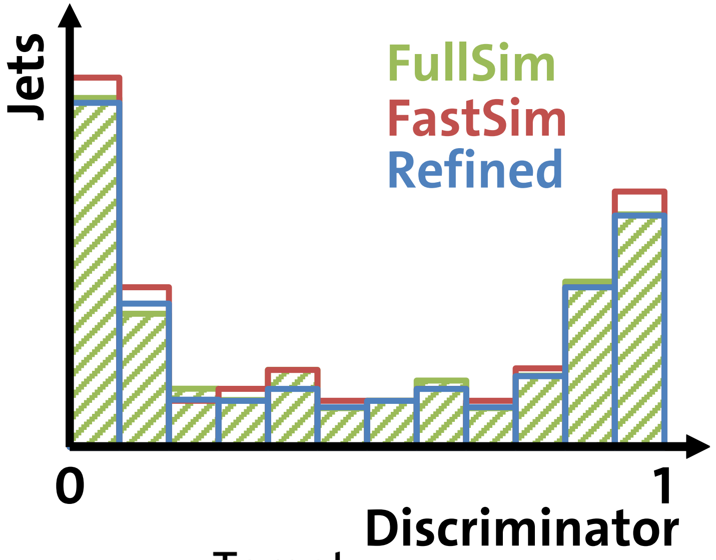
- **Training sample**: SUSY simplified model „T1tttt" simulated with FastSim and FullSim (same GEN events, Run 2 UL, no pile-up)
  - ➢ Match jets using ΔR angular criterion
  - ➢ **Jet triplets**: (GEN, FastSim, FullSim)

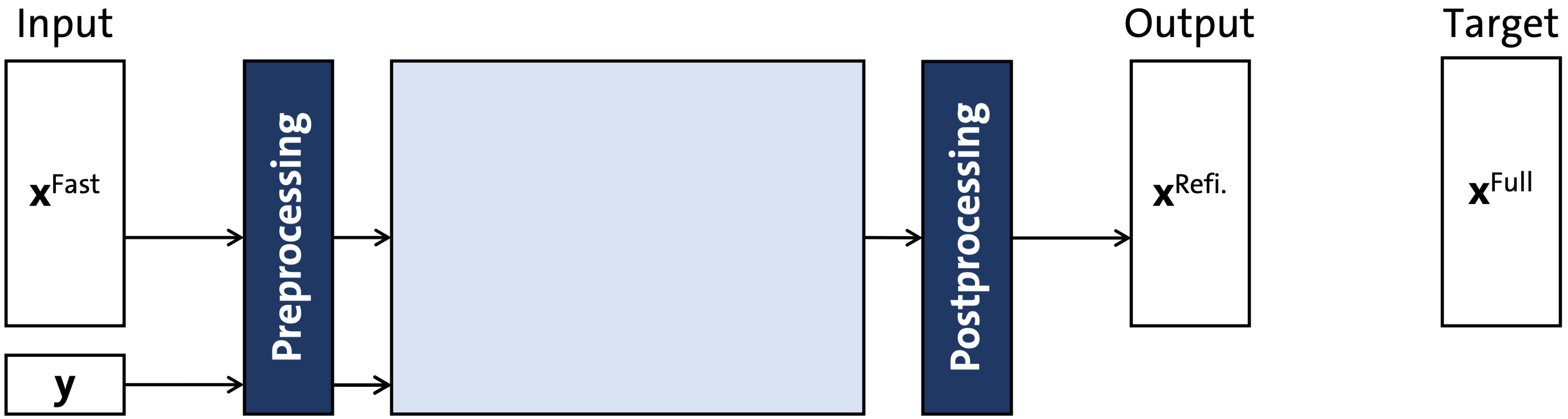


$^{*}\Delta R = \sqrt{\Delta\eta^2 + \Delta\varphi^2} < 0.2$

METHOD

- **Aim**: construct $\{\mathbf{x}_i^{\text{Refined}}\}$ from $\{\mathbf{x}_i^{\text{Fast}}\}$ that is more similar to $\{\mathbf{x}_i^{\text{Full}}\}$
- Network input/output:
  - Variables $\mathbf{x}^{\text{Fast/Full/Refined}}$ = (B, CvB, CvL, QG) [DeepJet discriminators]
  - Parameters $\mathbf{y}$ = (p_T^{GEN}, η^{GEN}, true hadron flavor)

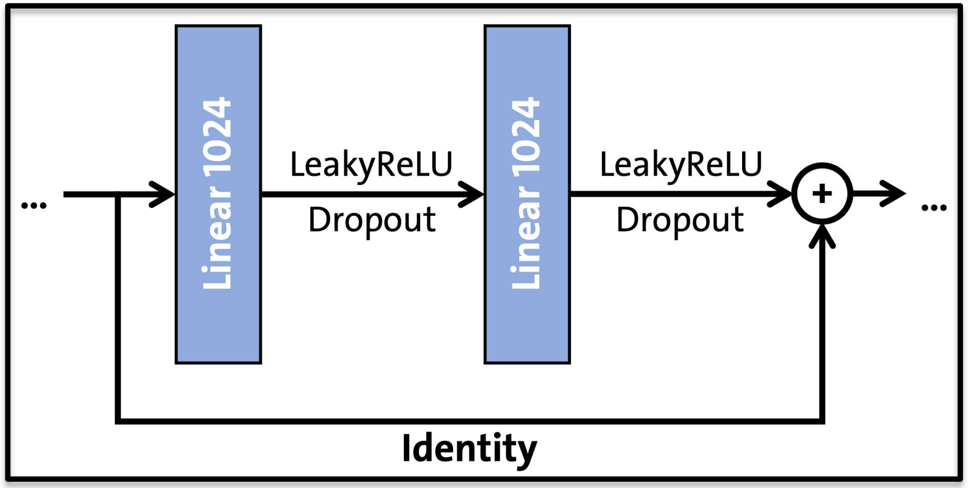


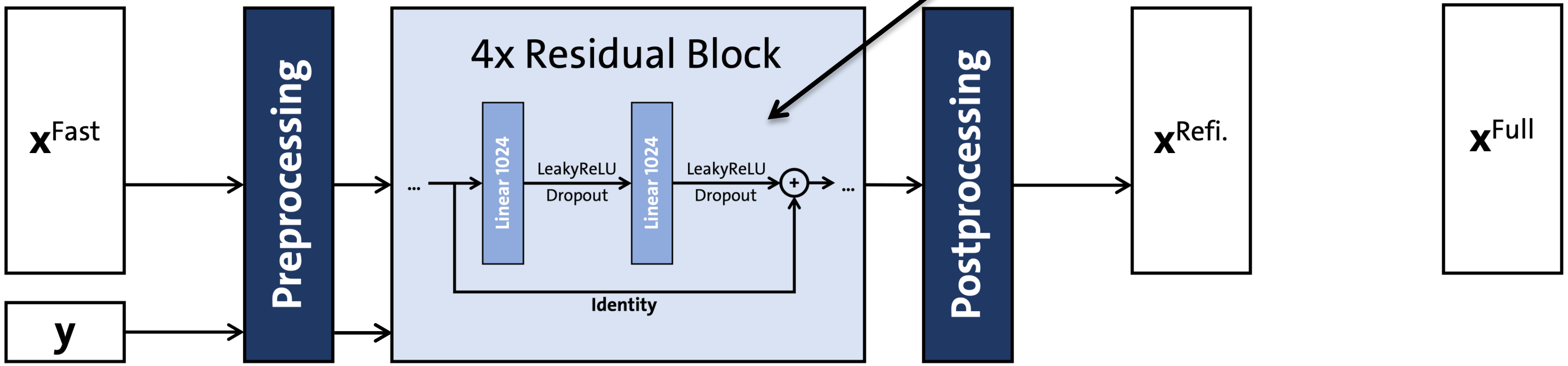


METHOD – NN ARCHITECTURE

ResNet paper arXiv:1512.03385

- ResNet-like **skip connections** → learn only residual corrections
- **Preprocessing**: transform input variables/parameters
  e.g., $\mathrm{logit}(x) = \ln\left(\frac{x}{1-x}\right)$
- **Postprocessing**: transform back & enforce DeepJet constraint
  (original DeepJet softmax output nodes need to sum to 1)





METHOD – LOSS TERMS

- Primary loss: **MMD** (ensemble-based) → cope with independent **stochasticity** in both simulation chains
  - Two-sample test: given two samples from P(X) and Q(Y):

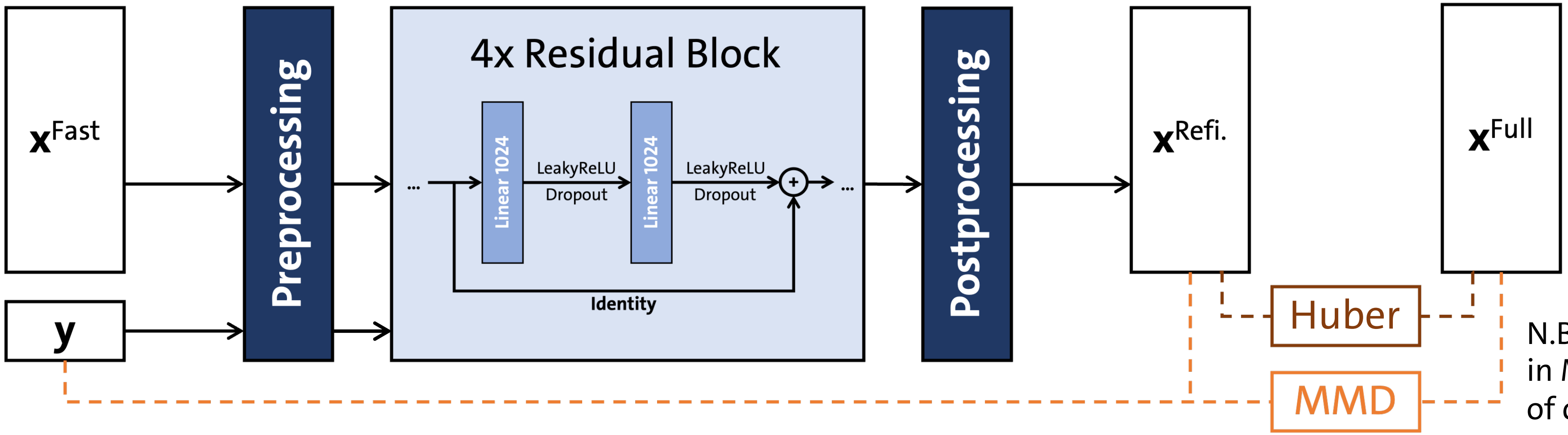
$$\widehat{\mathrm{MMD}}(P,Q) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} k(x_i,x_j) + \frac{1}{m^2}\sum_{i=1}^{m}\sum_{j=1}^{m} k(y_i,y_j) - \frac{2}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m} k(x_i,y_j)$$

n = m = batch size = 4096
k: Gaussian kernel (adaptive σ)

- Additional loss: **Huber** (output-target pair-based) → correct for **deterministic** FastSim biases
  - Distances of pairs:

$$l_n = \begin{cases} 0.5(x_n - y_n)^2, & \text{if } |x_n - y_n| < delta \\ delta * (|x_n - y_n| - 0.5 * delta), & \text{otherwise} \end{cases}$$

combination of MSE with MAE, less sensitive to outliers



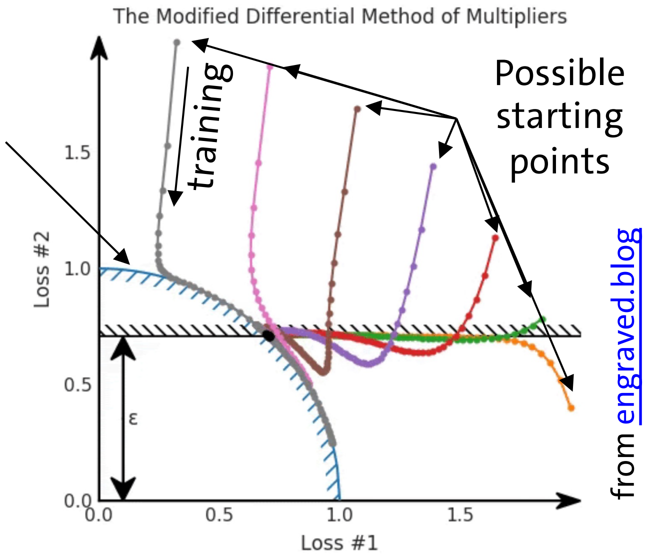
N.B.: include parameters in MMD to take account of correlations

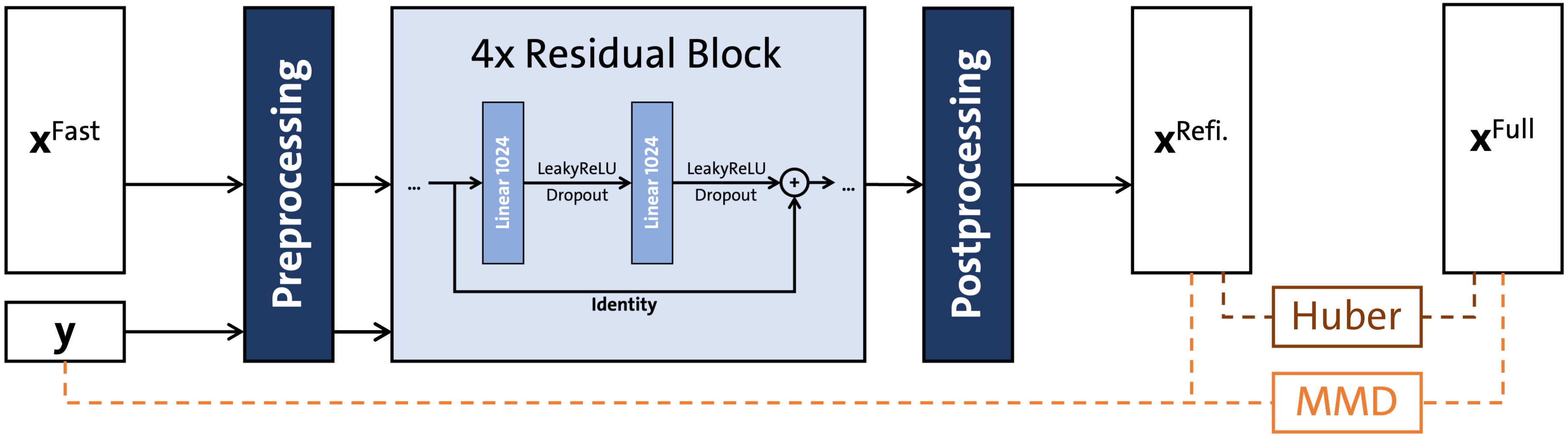
METHOD – MDMM

Combine loss terms via **MDMM algorithm** (see original paper)

- Reframe problem as **constrained optimization** using **Lagrangian**:
  $\mathcal{L} = f(\theta) - \lambda * (\varepsilon - g(\theta))$ → convergence mathematically formalized
- Minimize $f(\theta)$ (primary loss, „Loss #1") subject to $g(\theta) = \varepsilon$ (additional loss, „Loss #2")
- Gradient descent for NN parameters θ, gradient ascent for Lagrange multiplier λ
- Damping term to ensure convergence

Pareto front
(set of all optimal solutions, shape unknown)

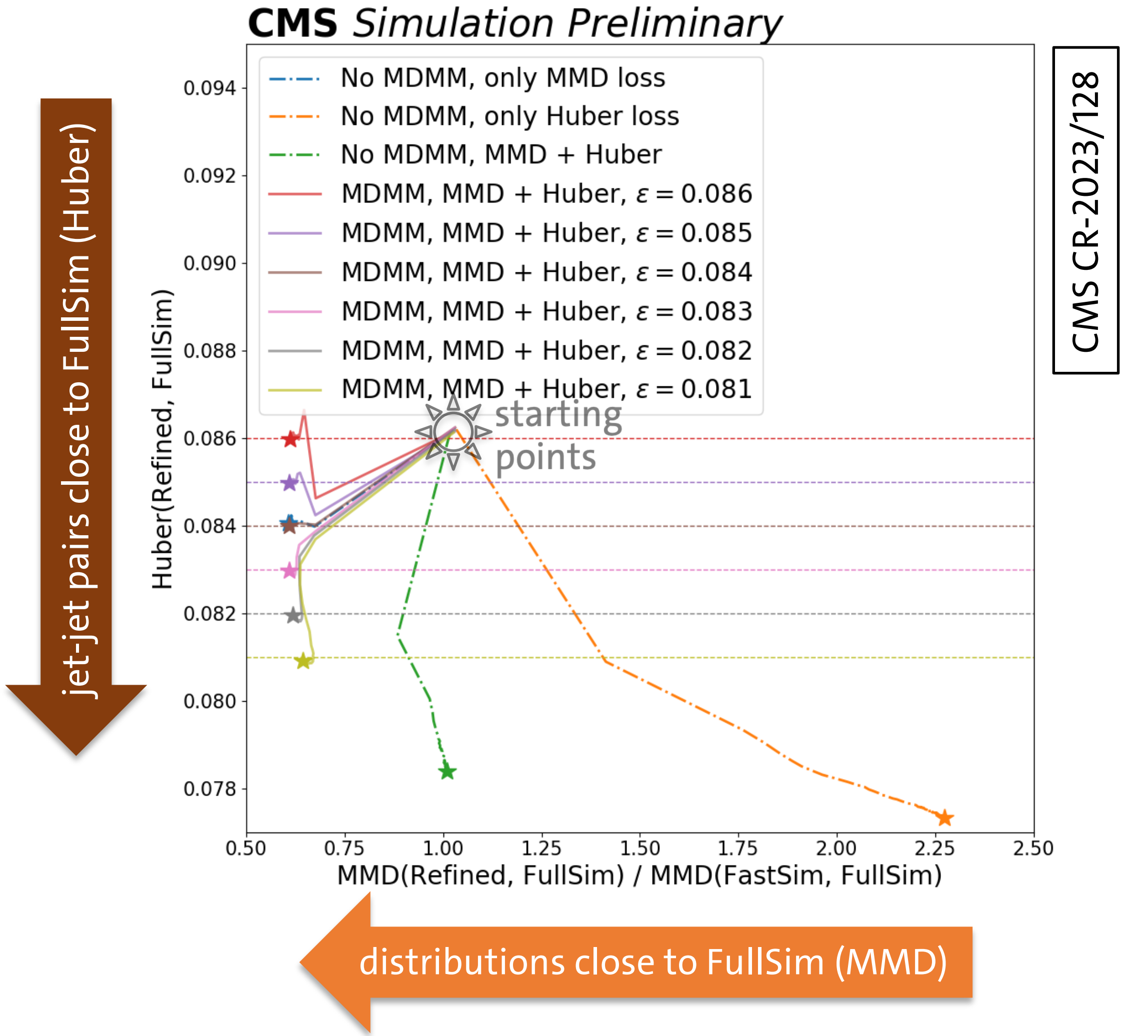




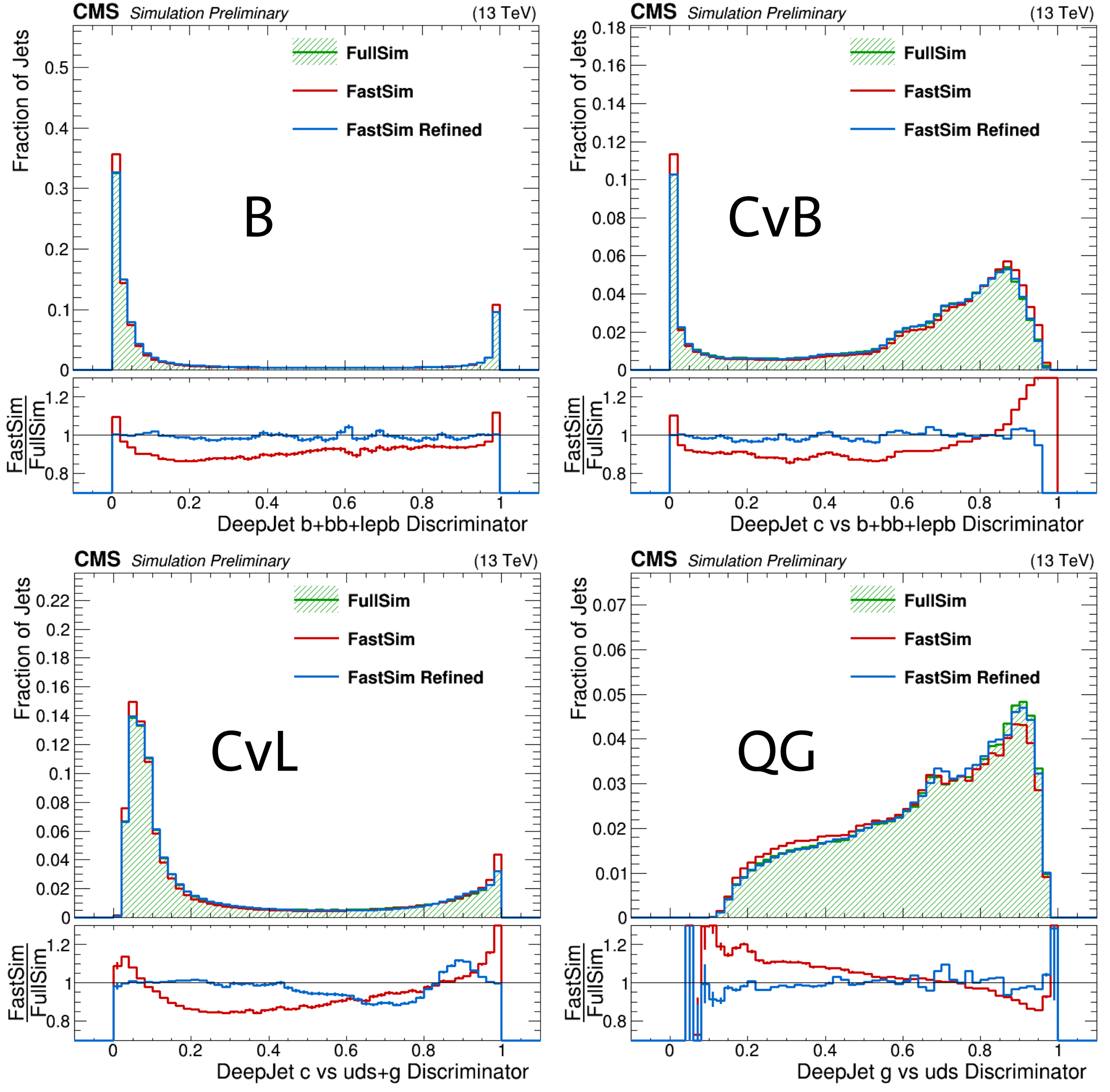
METHOD – TRAINING

Mapping out the Pareto front with different training versions:

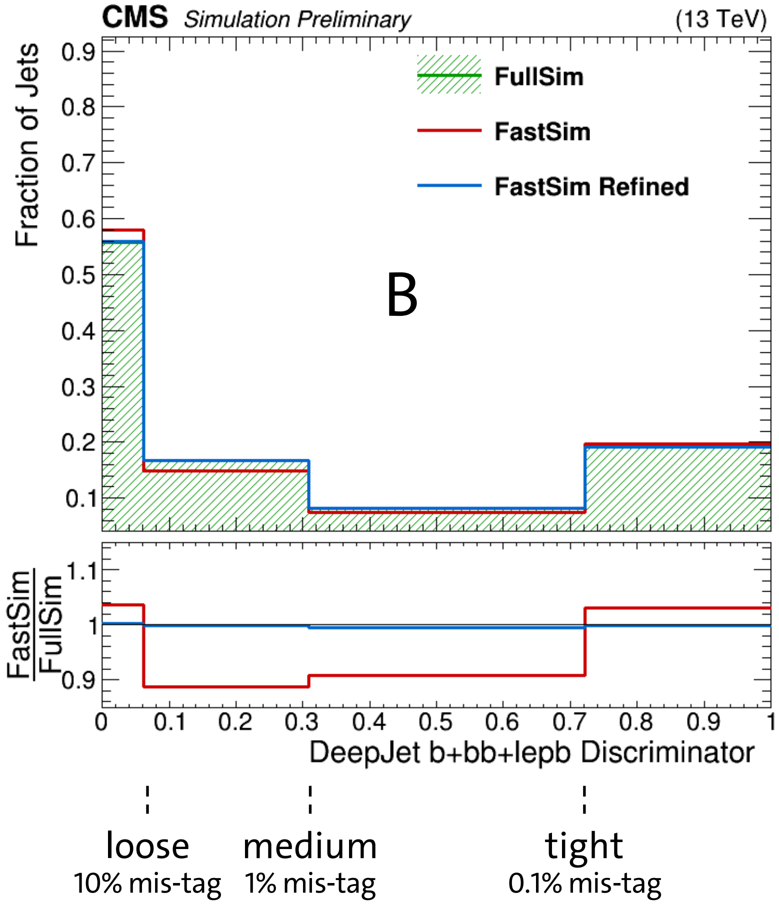
- **No MDMM** (dash-dotted lines)
  - ➢ Only one loss or constant weighted addition
  - ➢ Convergence might not be optimal (esp. Huber-only training)
- **With MDMM** (solid lines)
  - ➢ Scan of different ε values (horizontal dashed lines)
  - ➢ Convergence to desired point on Pareto front
  - ➢ Choose **ε = 0.084**



RESULTS – 1D DISTRIBUTIONS



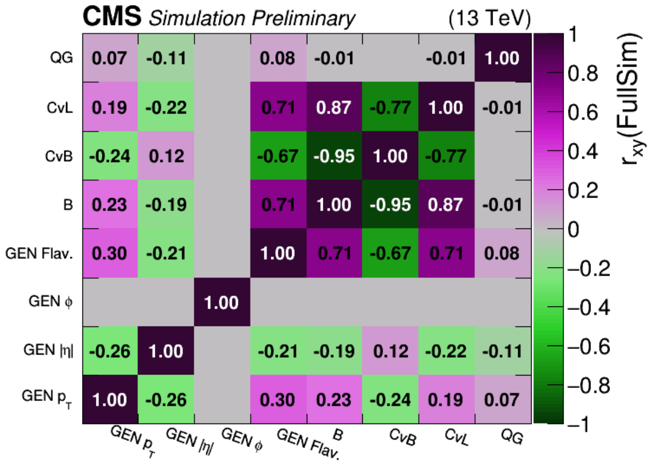
b tag binned by working point

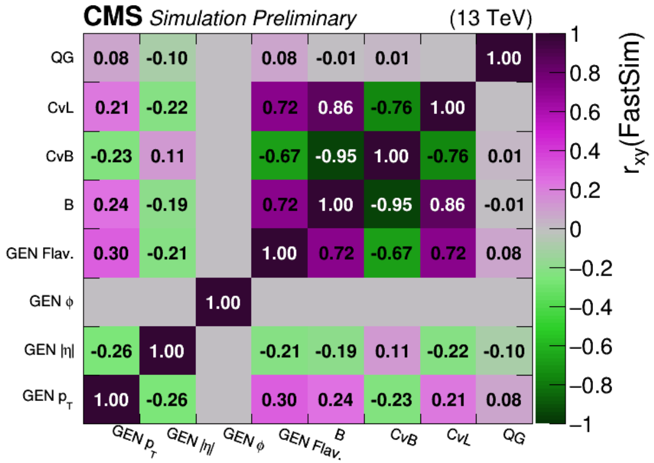


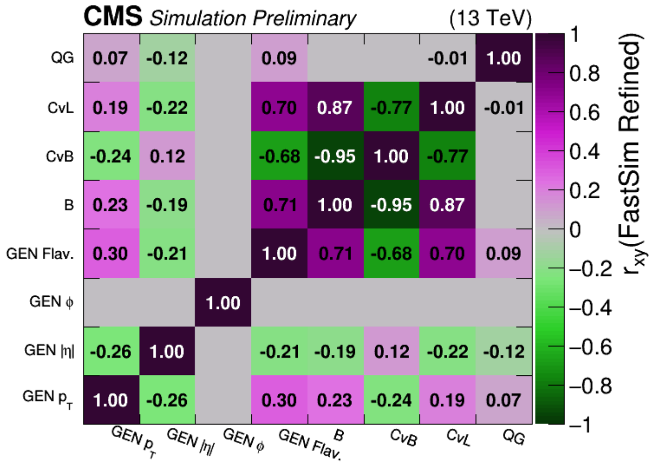
RESULTS – CORRELATIONS

	FullSim	FastSim	FastSim Refined

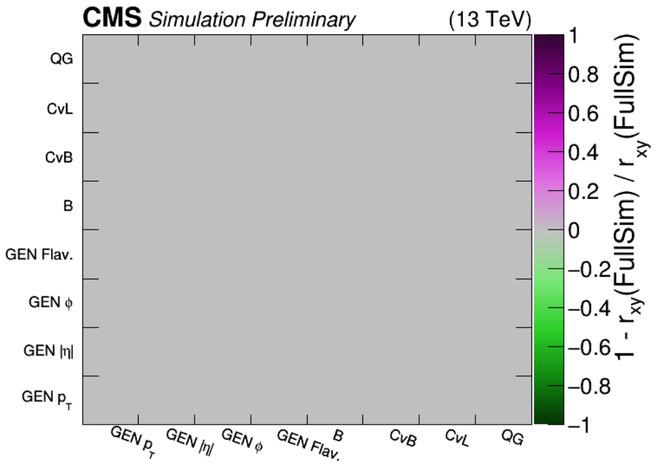
Pearson correlation coefficients

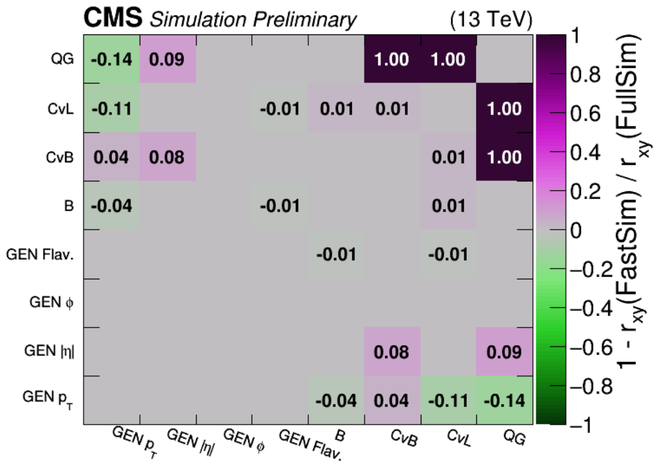


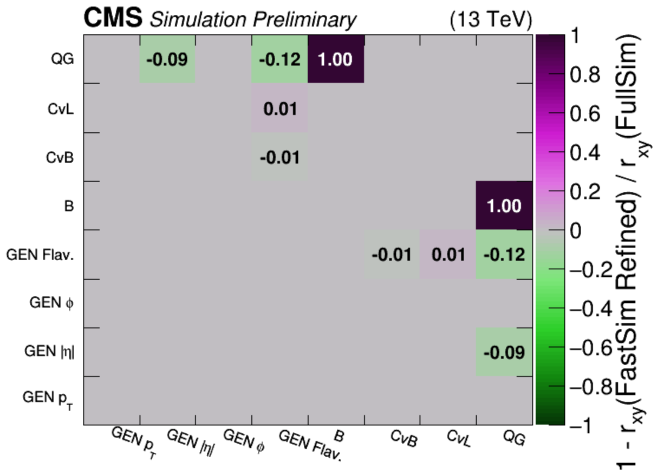




relative difference to FullSim







RESULTS – METRICS

- Quantitative **evaluation metrics** (introduced in 2211.10295)
  - Fréchet Physics Distance (FPD) & Kernel Physics Distance (KPD) (calculated with JetNet package)
  - Measured in 4D space of DeepJet discriminators

➢ Refined FastSim **significantly closer** to FullSim

➢ Similar improvement for **ttbar** (network trained on T1tttt)

	FPD x10^3	KPD x10^3	FPD x10^3 (ttbar)	KPD x10^3 (ttbar)
FullSim vs. FastSim	0.801 ± 0.046	1.07 ± 0.579	0.540 ± 0.036	0.927 ± 0.448
FullSim vs. FastSim Refined	**0.071 ± 0.025**	**0.083 ± 0.418**	**0.065 ± 0.025**	**-0.127 ± 0.164**
FullSim vs. FullSim (truth)	0.061 ± 0.029	-0.024 ± 0.250	0.061 ± 0.024	-0.119 ± 0.167

SUMMARY

- Regression-based refinement of FastSim leads to considerably **improved agreement** with FullSim
- Implemented in **CMSSW**, can be used for Run 2 samples (starting from CMSSW_10_6_35_PATCH1)
- More details in [2309.12919](2309.12919)

➢ **Extend** to other variables/objects (e.g., jet substructure)

➢ Ultimately, **replace** existing FastSim/FullSim corrections?

➢ Tune directly to **data**?