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

MAY 11, 2023 – 26TH INTERNATIONAL CONFERENCE ON COMPUTING IN HIGH ENERGY & NUCLEAR PHYSICS

REFINING FAST SIMULATION USING MACHINE LEARNING

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

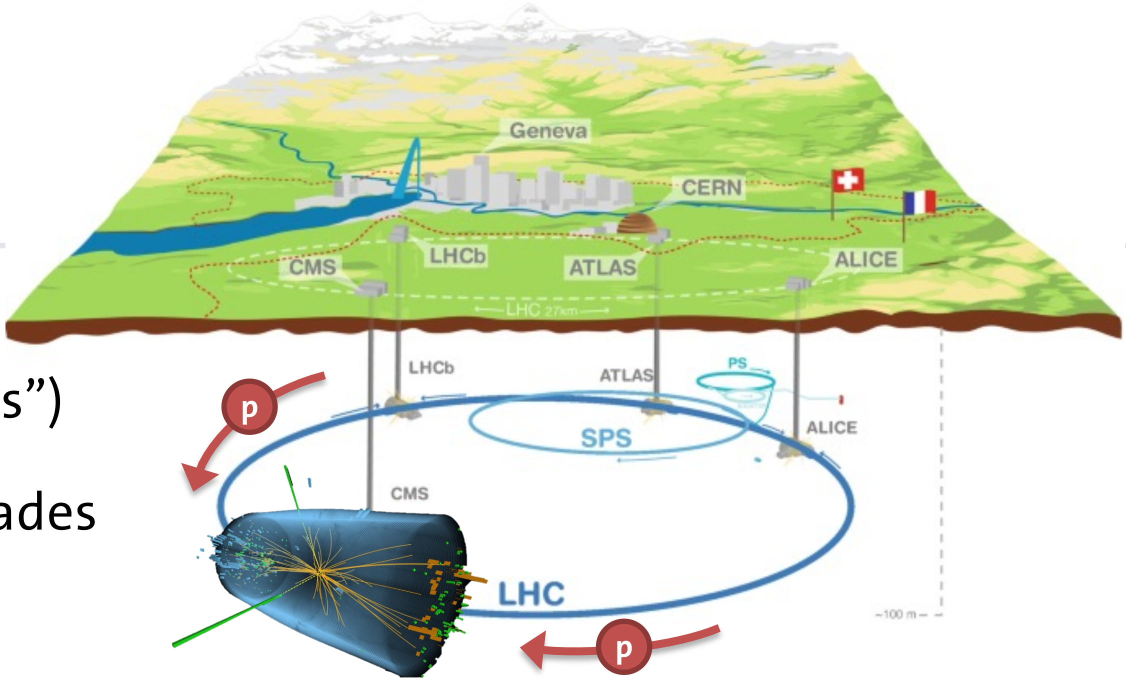
[1]UNIVERSITÄT HAMBURG, [2]FERMI NATIONAL ACCELERATOR LABORATORY

INTRODUCTION

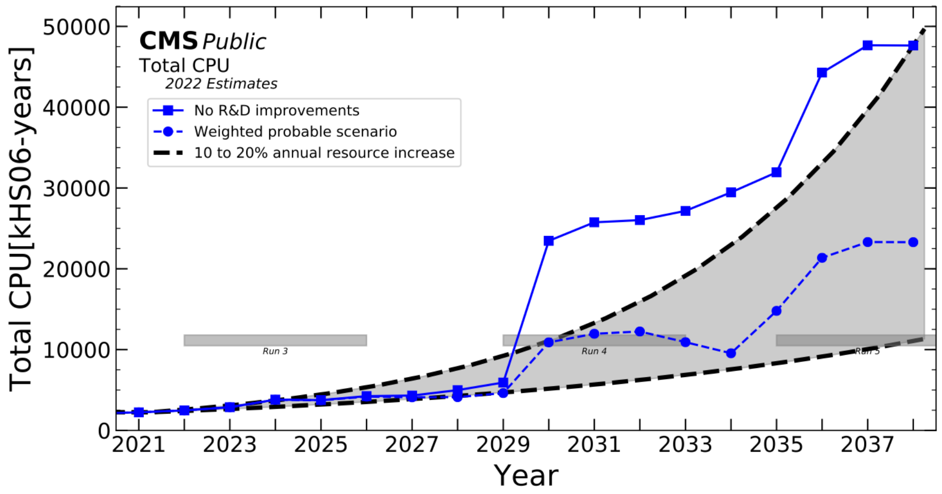
- Almost all high energy physics analyses rely on a large number of **simulated proton-proton collisions** (= "events")
- Higher LHC luminosity (= more events) & detector upgrades (= more complex data) → **fast simulation** techniques needed to stay within computing budget
- In **CMS**, two simulation chains (FullSim/FastSim) are used that produce output of same dimensionality/structure:

	GEN → Event generation	**SIM** → Detector simulation	**DIGI** → Digitization	**RECO** Reconstruction
FullSim	same e.g., MadGraph	GEANT4	same	analyze as if data
FastSim ≈ 15% of sim. events	same e.g., MadGraph	parametrized energy loss 100x faster	same	use GEN info 2.5x faster

➢ In total: FastSim ≈ 10x faster than FullSim



CMS-NOTE-2022-008



twiki.cern.ch/twiki/bin/view/CMSPublic/CMSOfflineComputingResults

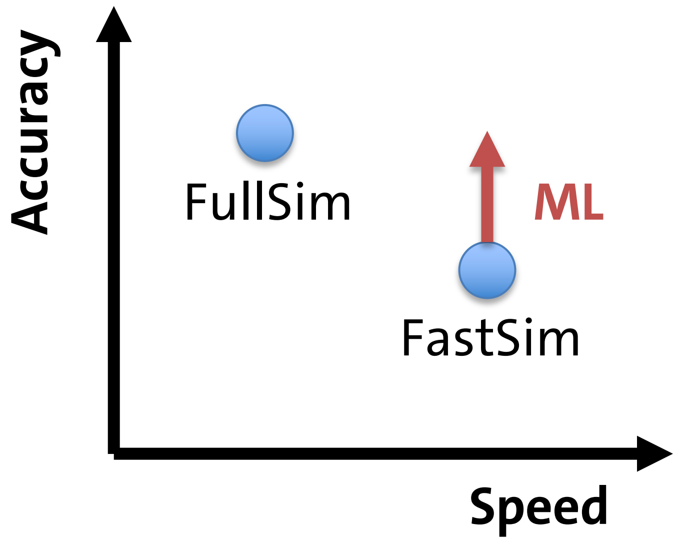
INTRODUCTION

- FastSim **advantage in speed** comes at the price of **decreased accuracy** in some of the final observables
  (N.B.: standard analyses use further processed data formats e.g., AOD or NanoAOD)

GEN →	SIM →	DIGI →	RECO →	AOD →	NanoAOD
Event generation	Detector simulation	Digitization	Reconstruction	“Analysis Object Data”	further processed analysis format

➢ **Aim**: increase FastSim accuracy to promote its wider usage

- Possible FastSim **tuning** approaches:
  - Internal tuning of functions/parameters (within SIM/RECO)
  - Post-hoc tuning (after NanoAOD)
    - **Reweighting** = defining weights for individual events/objects/…
      e.g., DCTR introduced in [arXiv:1907.08209](https://arxiv.org/abs/1907.08209)
    - **Refining** = changing (high-level) observables
      e.g., Wasserstein-GAN for air showers in [arXiv:1802.03325](https://arxiv.org/abs/1802.03325)



INTRODUCTION

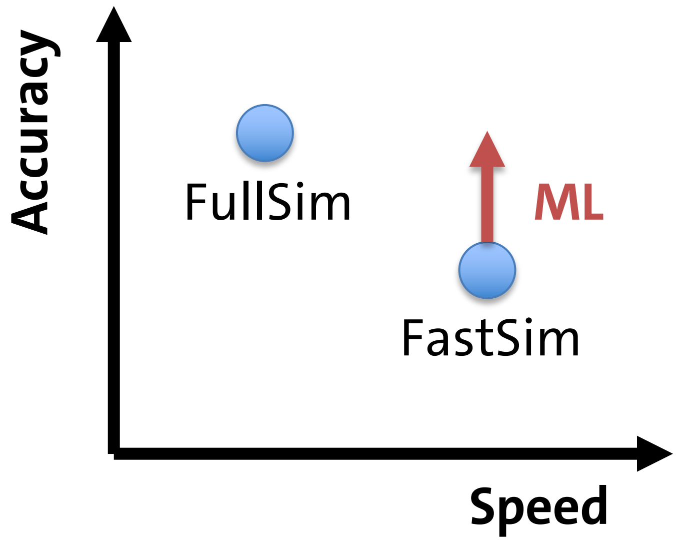
- FastSim **advantage in speed** comes at the price of **decreased accuracy** in some of the final observables

  (N.B.: standard analyses use further processed data formats e.g., AOD or NanoAOD)

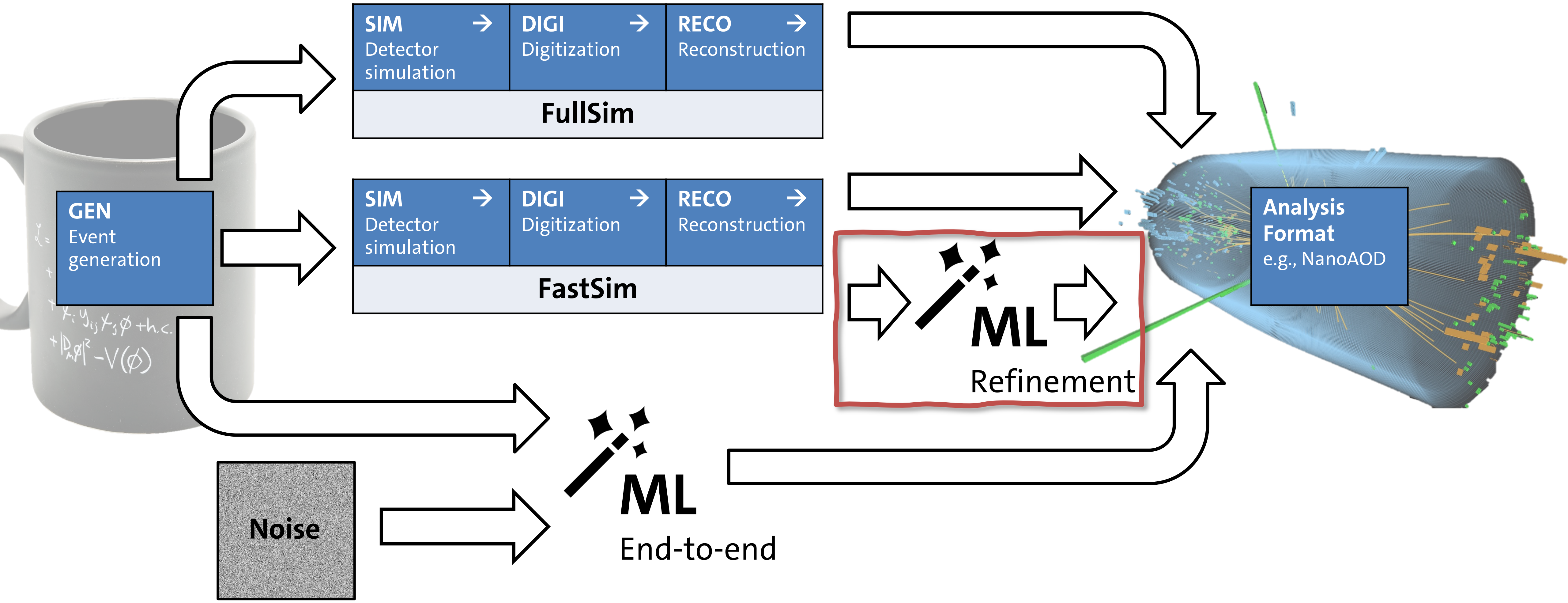
GEN → Event generation	**SIM** → Detector simulation	**DIGI** → Digitization	**RECO** → Reconstruction	**AOD** → "Analysis Object Data"	**NanoAOD** further processed analysis format

➢ **Aim**: increase FastSim accuracy to promote its wider usage

- Possible FastSim **tuning** approaches:
  - Internal tuning of functions/parameters (within SIM/RECO)
  - Post-hoc tuning (after NanoAOD)
    - **Reweighting** = defining weights for individual events/objects/…
      e.g., DCTR introduced in arXiv:1907.08209
    - **Refining** = changing (high-level) observables
      e.g., Wasserstein-GAN for air showers in arXiv:1802.03325

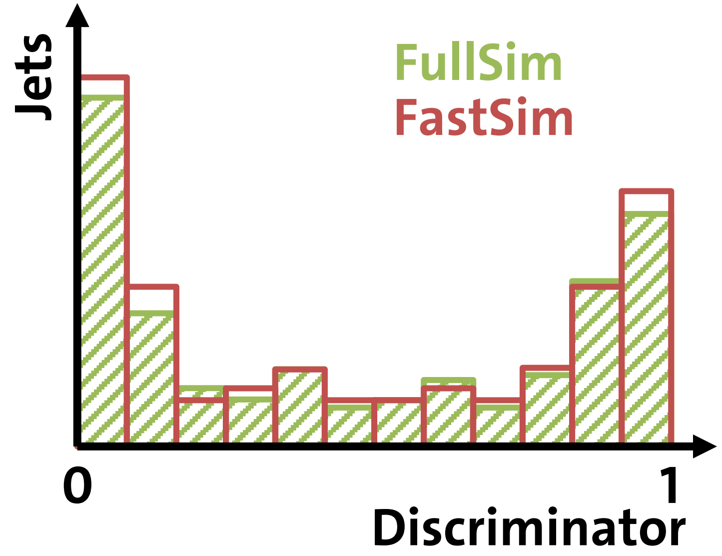


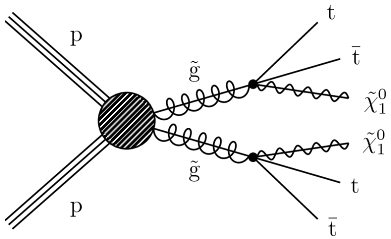
INTRODUCTION



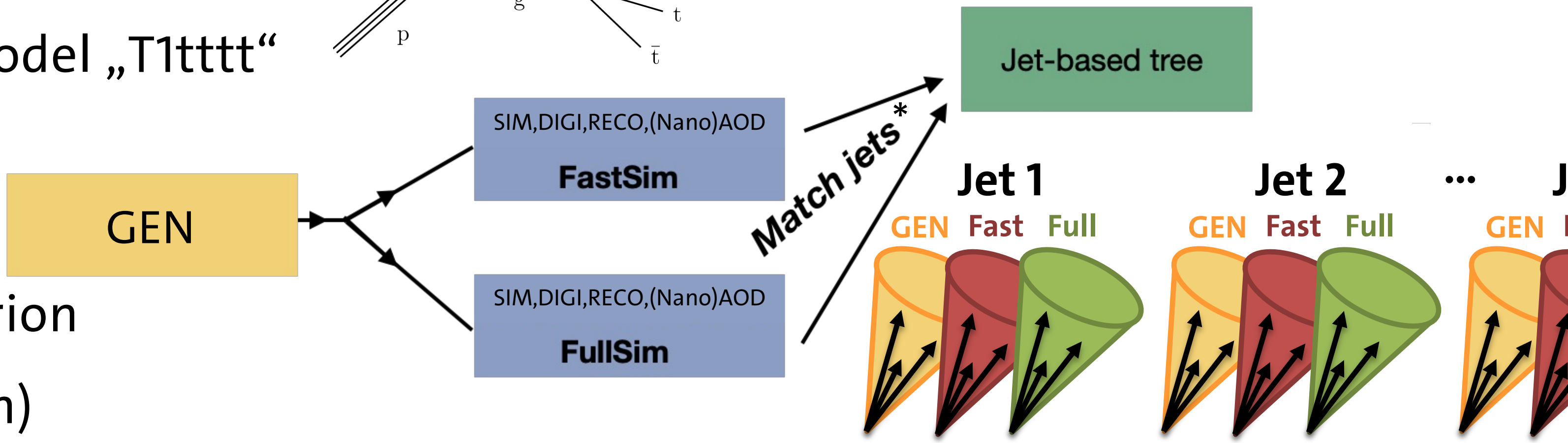
INTRODUCTION

- Focus on jet flavour tagging: 4 NanoAOD **DeepJet discriminators**:
  b+bb+lepb, c vs b+bb+lepb, c vs uds+g, g vs uds
  „B“ „CvB“ „CvL“ „QG“
  (DeepJet NN softmax output nodes: b, bb, lepb, c, uds, g; [arXiv:2008.10519](https://arxiv.org/abs/2008.10519))
  ➢ FastSim/FullSim discrepancies O(10 %)



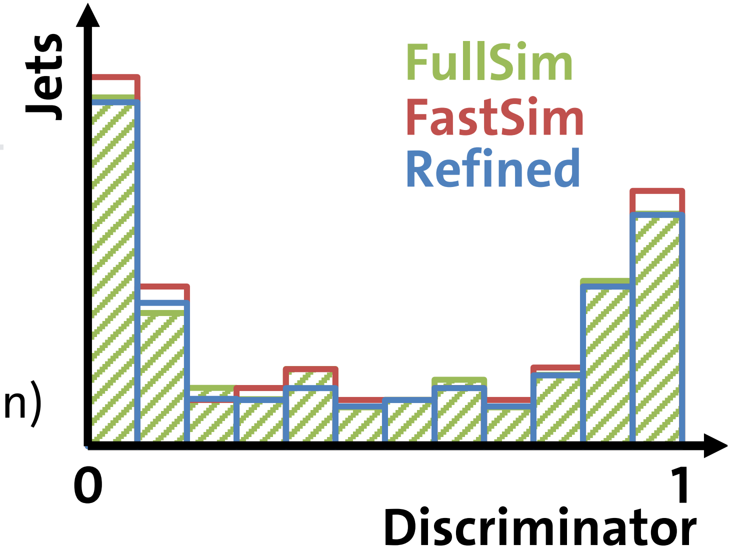


- **Training sample**: SUSY simplified model „T1tttt“ simulated with FastSim and FullSim
  (same GEN events, no pile-up)
  ➢ Match jets using ΔR angular criterion
  ➢ **Jet triplets**: (GEN, FastSim, FullSim)



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

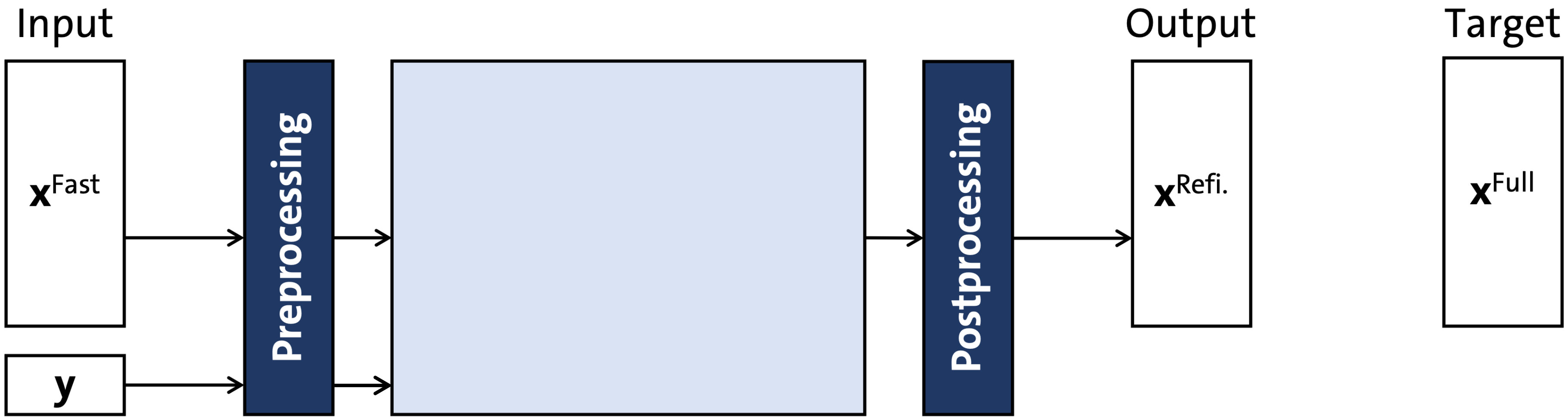
REFINING (REGRESSION) – GENERAL IDEA



Employ **regression neural network** to refine FastSim:

- Input: **FastSim** variables $\mathbf{x}^{\text{Fast}}$ = 4 DeepJet discriminators

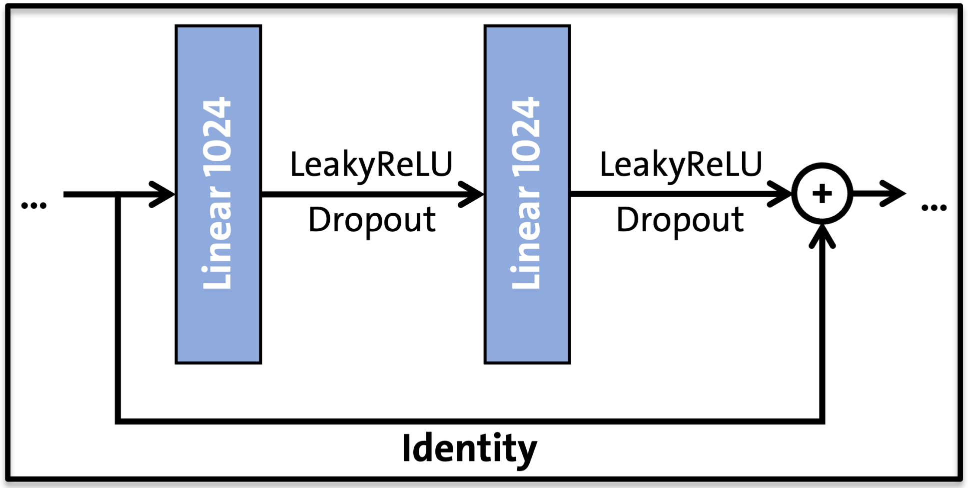
  Parameters $\mathbf{y}$ = p_T^{GEN}, η^{GEN}, true hadron flavor (b, c, or light quark/gluon)
- Output: **Refined** variables $\mathbf{x}^{\text{Refined}}$ = 4 DeepJet discriminators
- Target: **FullSim** variables $\mathbf{x}^{\text{Full}}$ = 4 DeepJet discriminators

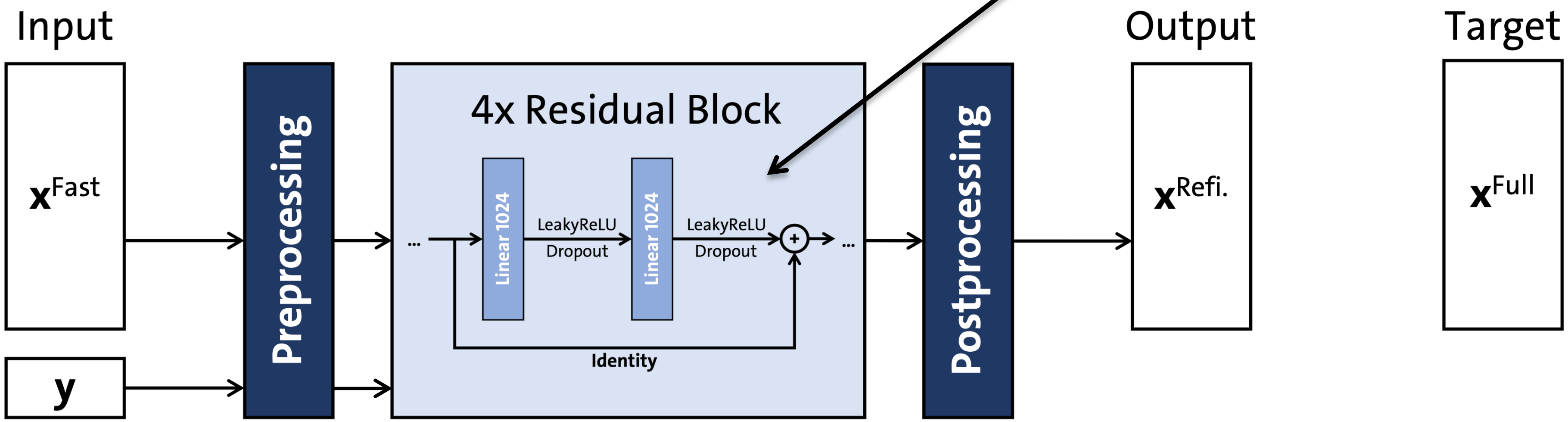


REFINING (REGRESSION) – NN ARCHITECTURE

ResNet paper arXiv:1512.03385

- ResNet-like **skip connections** → learn only residual corrections
- **Preprocessing**: transform input variables/parameters
  e.g., logit-transform $\mathrm{logit}(x) = \ln\left(\frac{x}{1-x}\right)$ to map from (0, 1) to (-∞, ∞)
- **Postprocessing**: transform back & enforce DeepJet constraint
  (original DeepJet softmax output nodes need to sum to 1)





MMD: Maximum Mean Discrepancy (ensembles)
MSE: Mean Squared Error (jet-jet pairs)
MAE: Mean Absolute Error (jet-jet pairs)

REFINING (REGRESSION) – LOSS TERMS

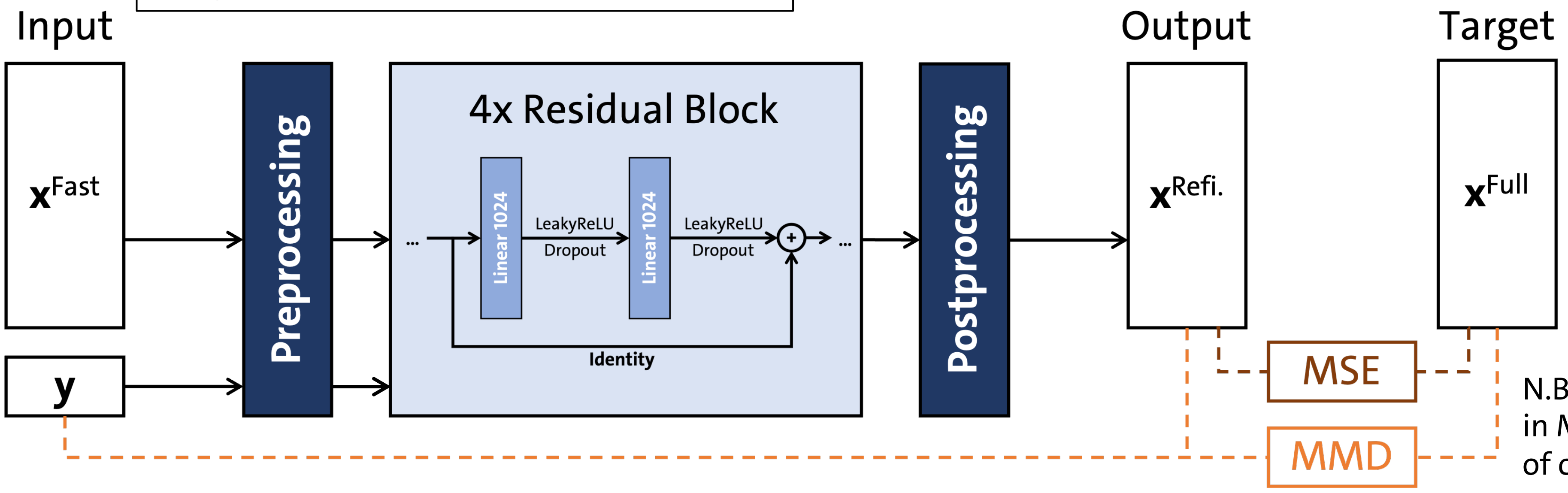
- **Primary loss: MMD** (distribution-based)
  - Comparing ensembles of jets not jet-jet pairs
  - To cope with independent stochasticity in both simulation chains

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: MSE/Huber** (output-target pair-based) → correct for deterministic FastSim biases
  - Use Huber loss: $l_n = \begin{cases} 0.5(x_n - y_n)^2, & \text{if } |x_n - y_n| < delta \\ delta * (|x_n - y_n| - 0.5 * delta), & \text{otherwise} \end{cases}$ (combination of MSE/MAE, less sensitive to outliers)

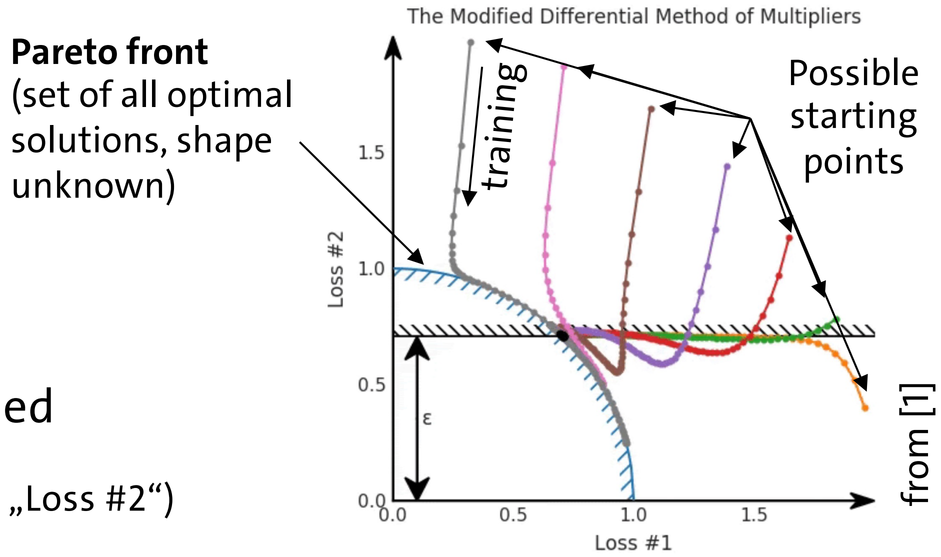


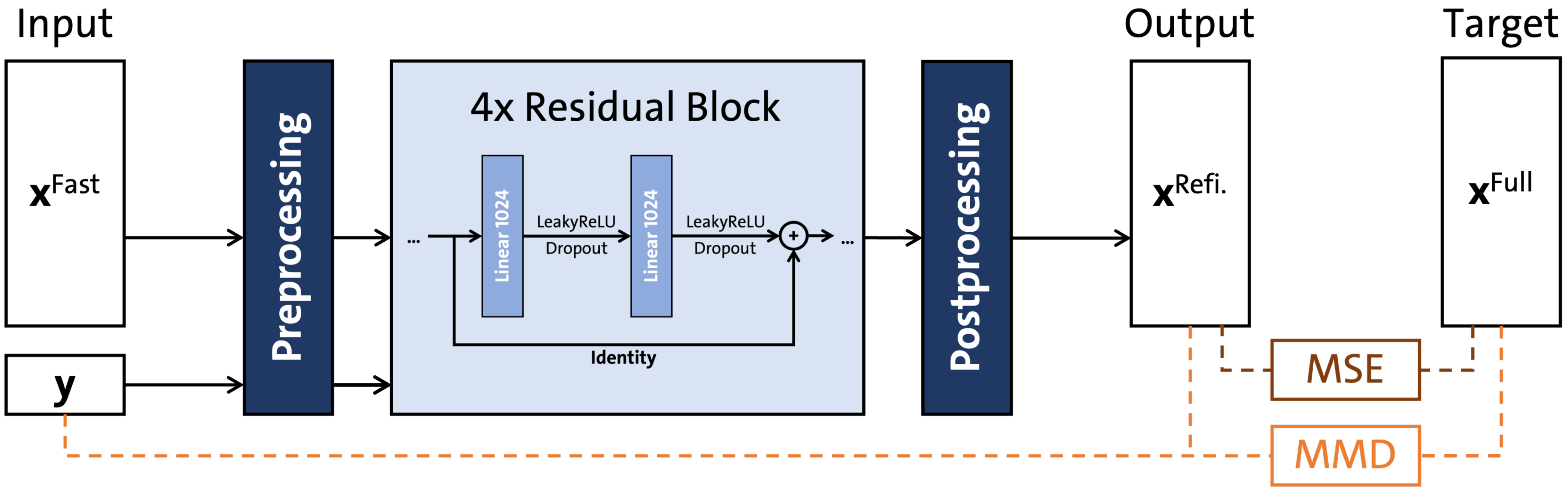
N.B.: include parameters in MMD to take account of correlations

REFINING (REGRESSION) – LOSS TERMS

Combine loss terms via **MDMM algorithm**:

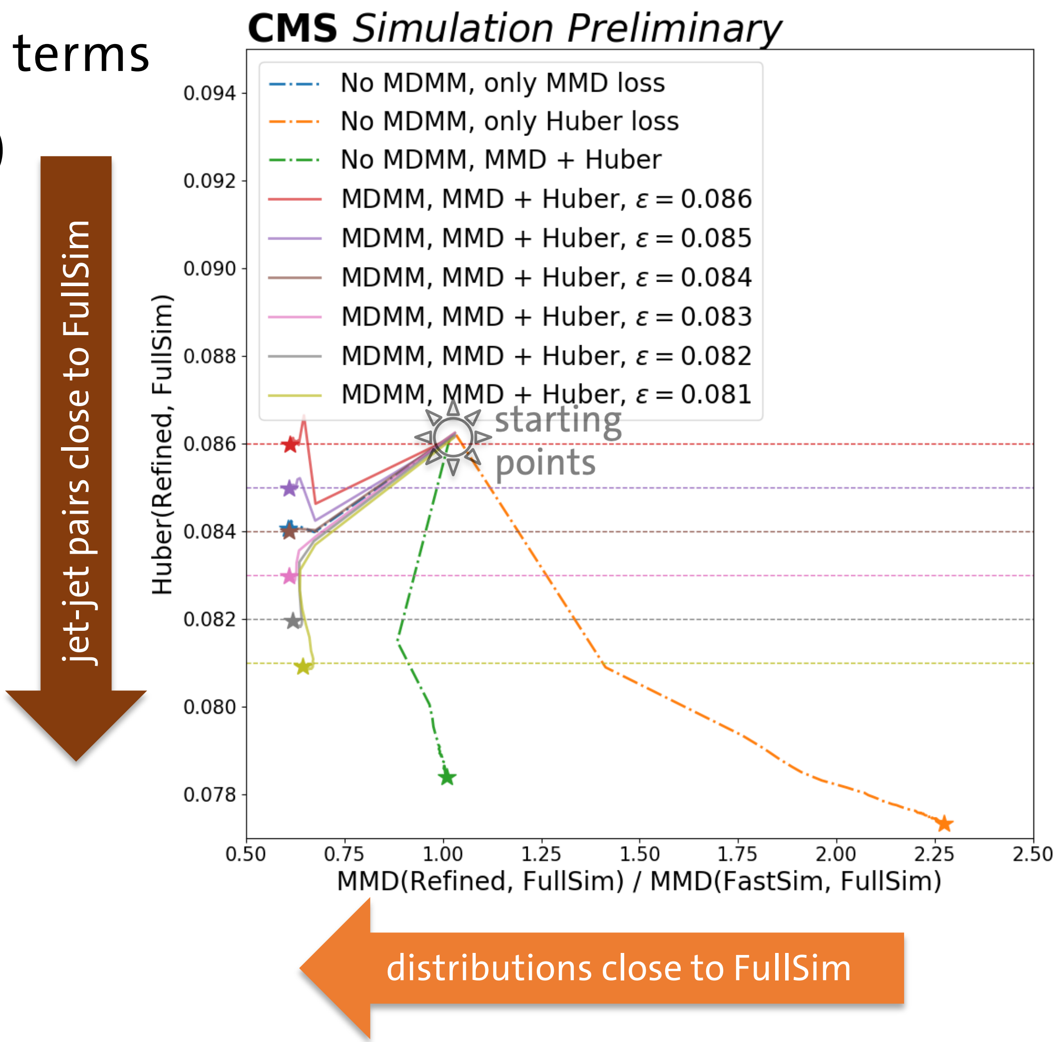
- Reframe problem as **constrained optimization** using **Lagrangian**:
  $\mathcal{L} = f(\theta) - \lambda * \left(\varepsilon - g(\theta)\right)$ → convergence mathematically formalized
- Minimize $f(\theta)$ (primary loss, „Loss #1") subject to $g(\theta) = \varepsilon$ (additional loss, „Loss #2")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



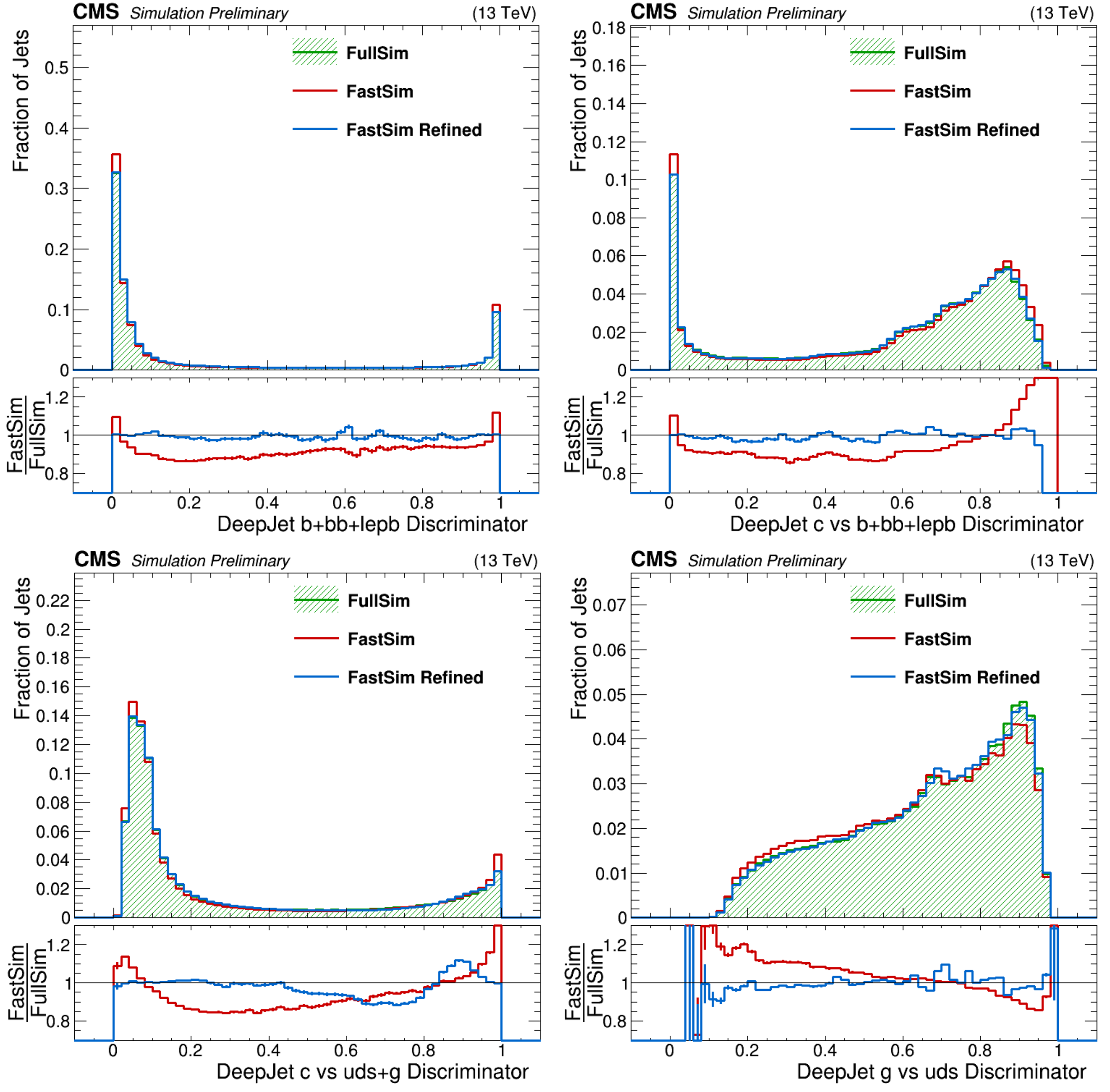


REFINING (REGRESSION) – TRAINING

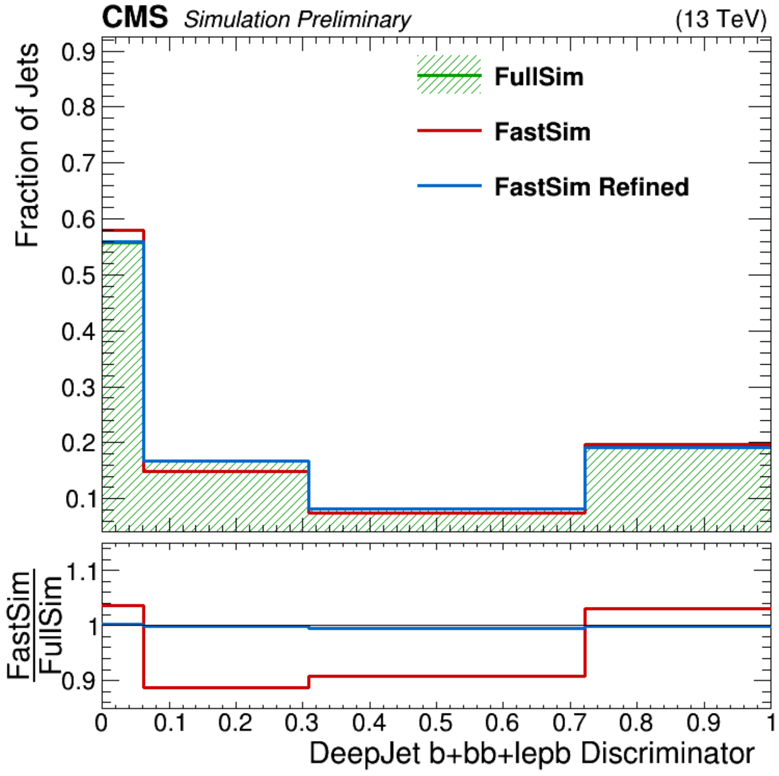
- **Plot**: training convergence in plane of the two loss terms
  - Primary loss: **MMD(Refined, FullSim)** (horizontal axis)
  - Constraint: **Huber(Refined, FullSim)** (vertical axis)
- **No MDMM** (dash-dotted lines)
  - ➢ Only one loss or constant weighted addition
  - ➢ Convergence might not be optimal
- **With MDMM** (solid lines)
  - ➢ Scan of different ε values (horizontal dashed lines)
  - ➢ Convergence to desired point on Pareto front
  - ➢ Choose **ε = 0.084**



REFINING (REGRESSION) – RESULTS



b tag binned by working point



loose
10% mis-tag

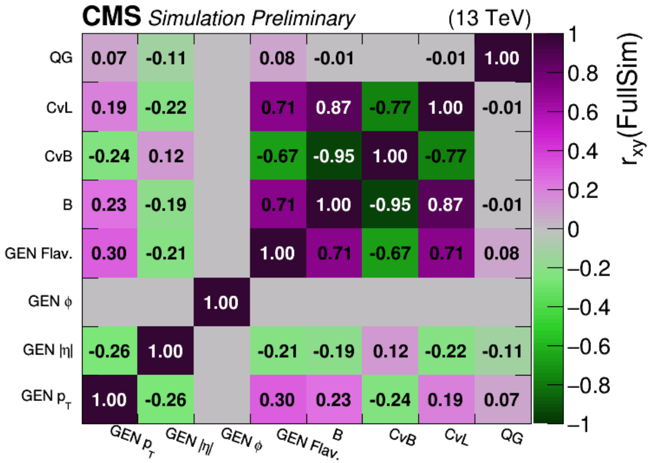
medium
1% mis-tag

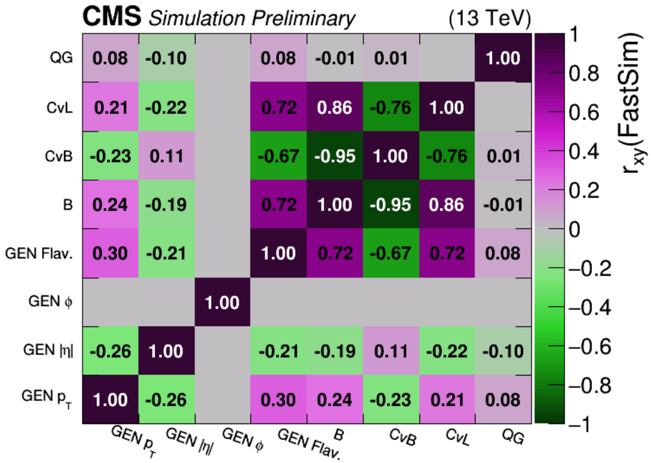
tight
0.1% mis-tag

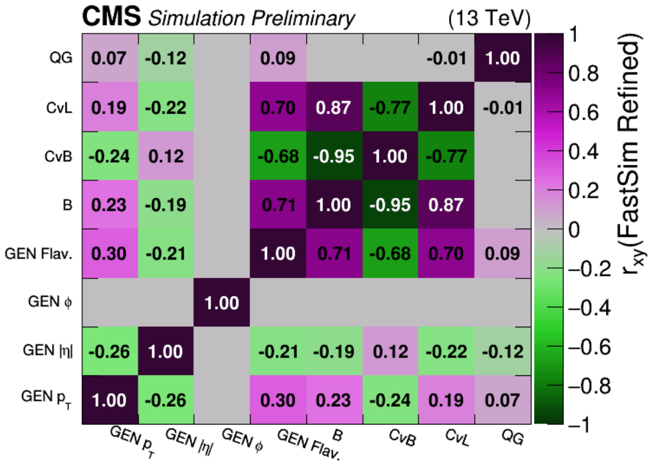
REFINING (REGRESSION) – RESULTS

FullSim | FastSim | FastSim Refined

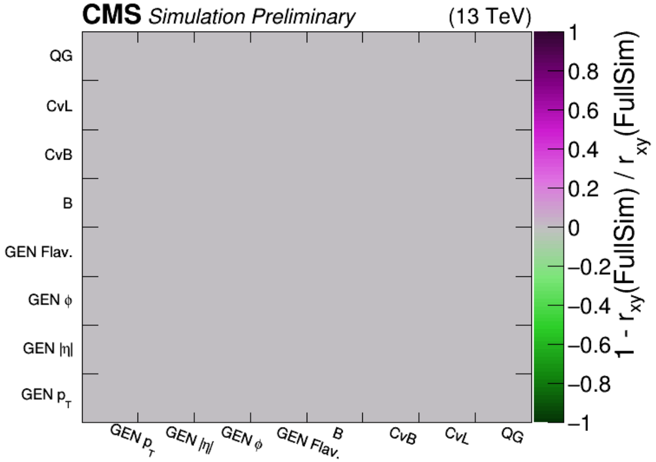
Pearson correlation coefficients

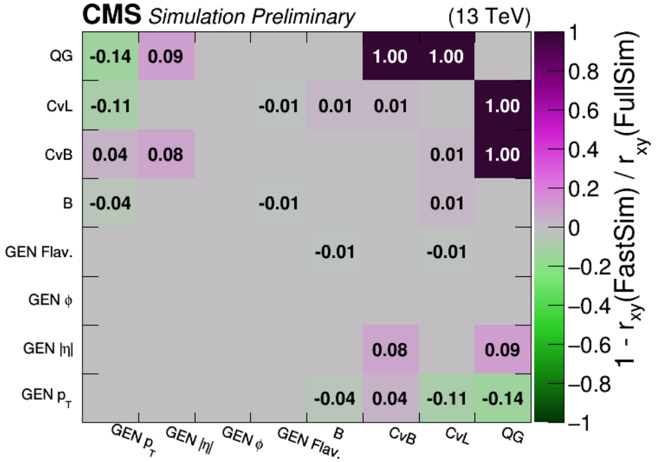


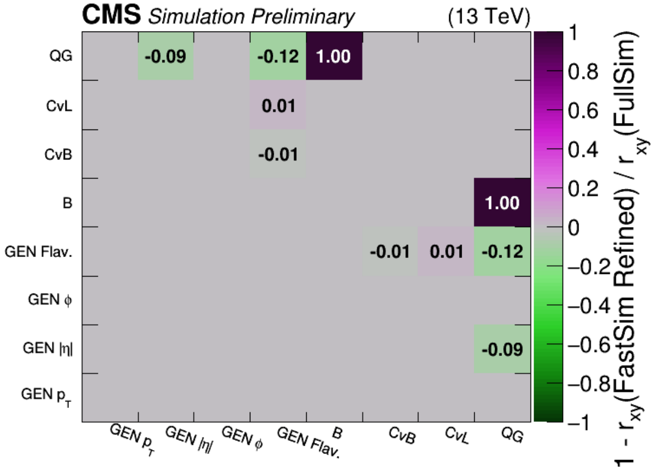




relative difference to FullSim







SUMMARY & OUTLOOK

✓ ResNet-like regression NN can be used as **post-hoc refinement layer** to FastSim output

- Considerably improved agreement with FullSim output
- Improvement in correlations among output observables and external parameters

✓ Validation in TTbar events (different event topology)

✓ Corresponding pull request merged into CMSSW

- Extend to other variables/objects (e.g., jet substructure)
- Tune directly to data?