Fast Data-Driven simulation of Cherenkov Detectors Using Generative Adversarial Networks

Artem Maevskiy, Denis Derkach, Nikita Kazeev, Andrey Ustyuzhanin, Maksim Artemev, Lucio Anderlini
On behalf of the LHCb collaboration

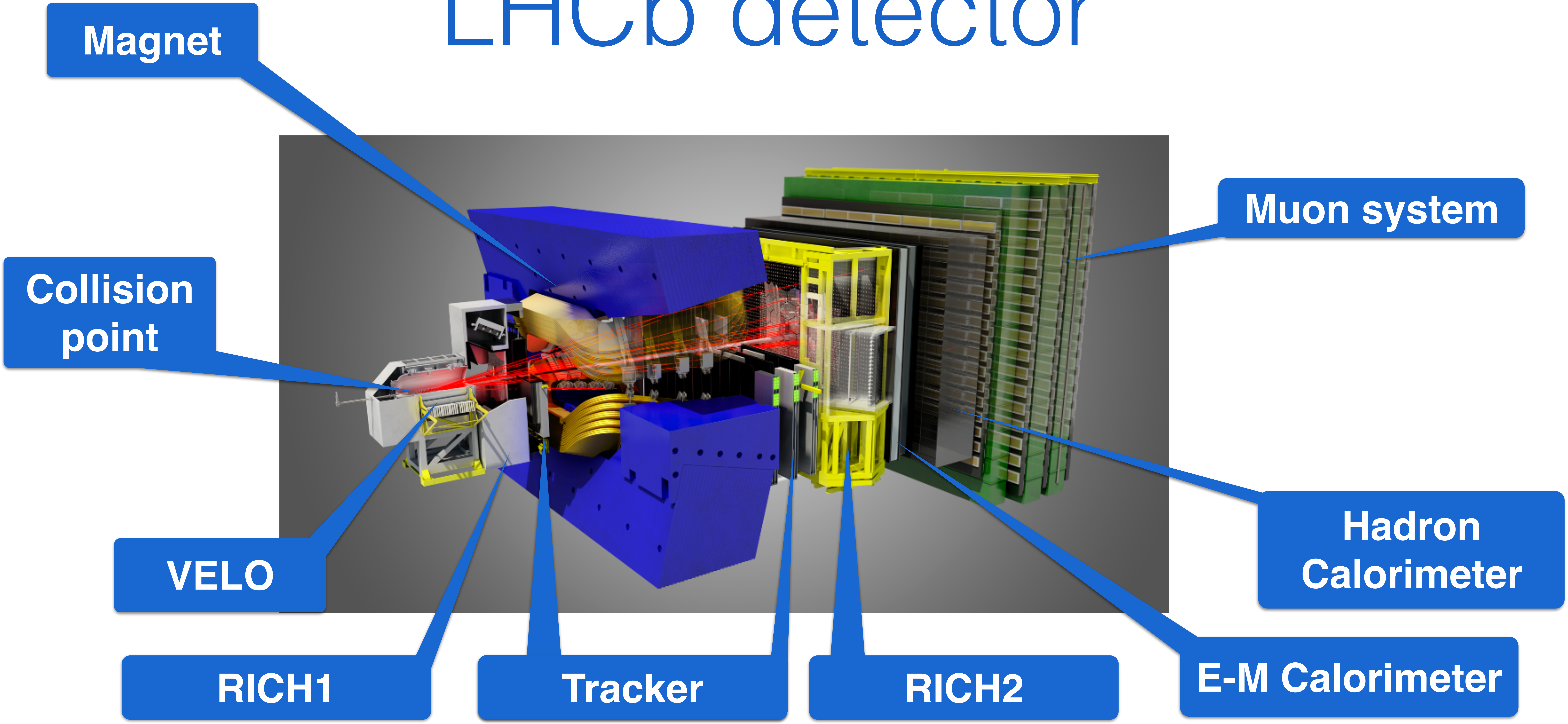
19th International Workshop on Advanced Computing and Analysis Techniques in Physics Research
ACAT 2019
11-15 March 2019
Saas Fee, Switzerland

NATIONAL RESEARCH UNIVERSITY
SCHOOL OF DATA ANALYSIS
SAPIENZA UNIVERSITÀ DI ROMA
Istituto Nazionale di Fisica Nucleare

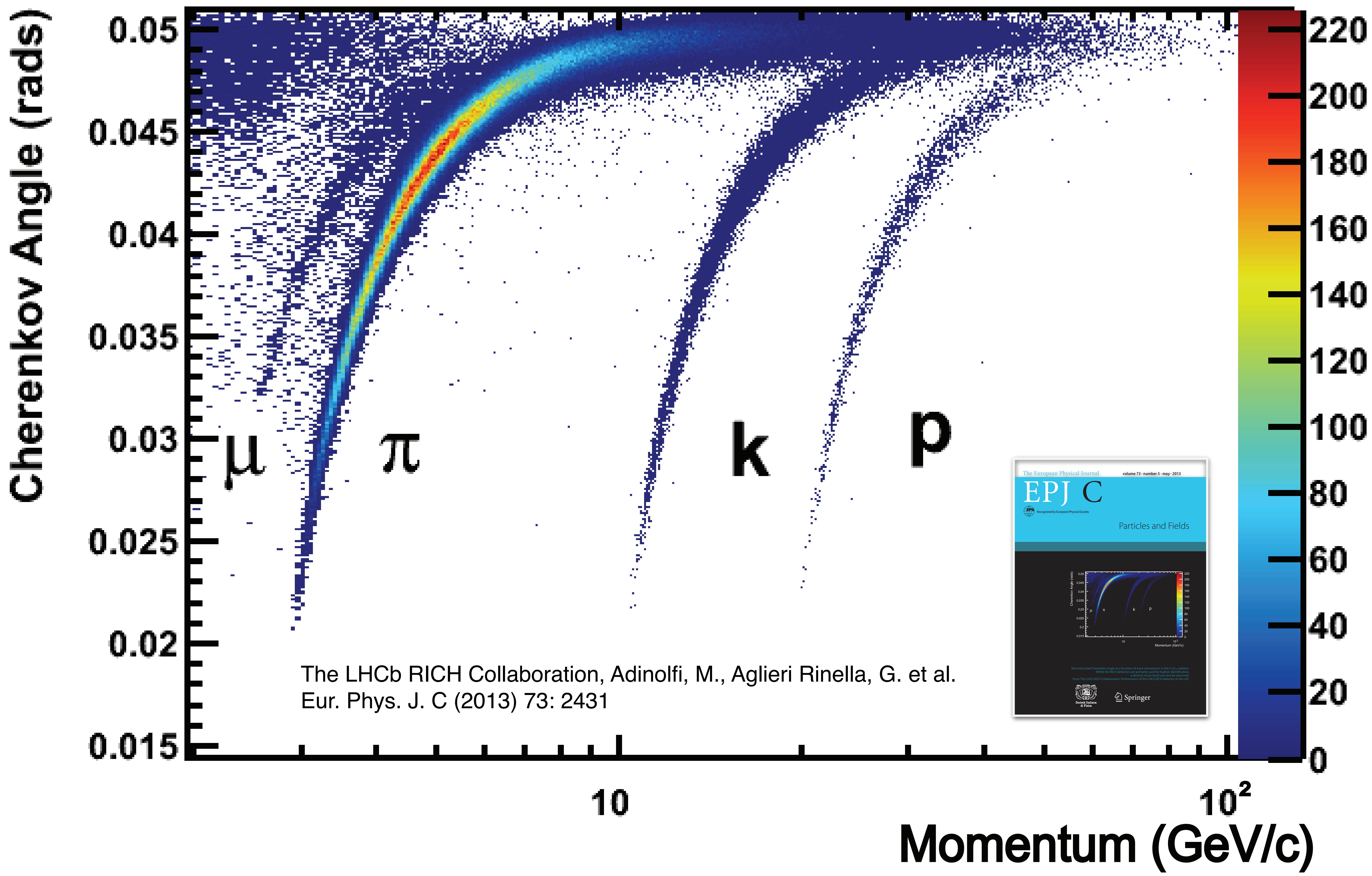
Outline

- Particle identification with Cherenkov detectors at LHCb
  - Fast simulation set-up
- Generative Adversarial Networks (GANs)
  - Basic architecture
  - JS and Wasserstein metrics and problems with them
  - Cramér and energy distances
- Results and discussion

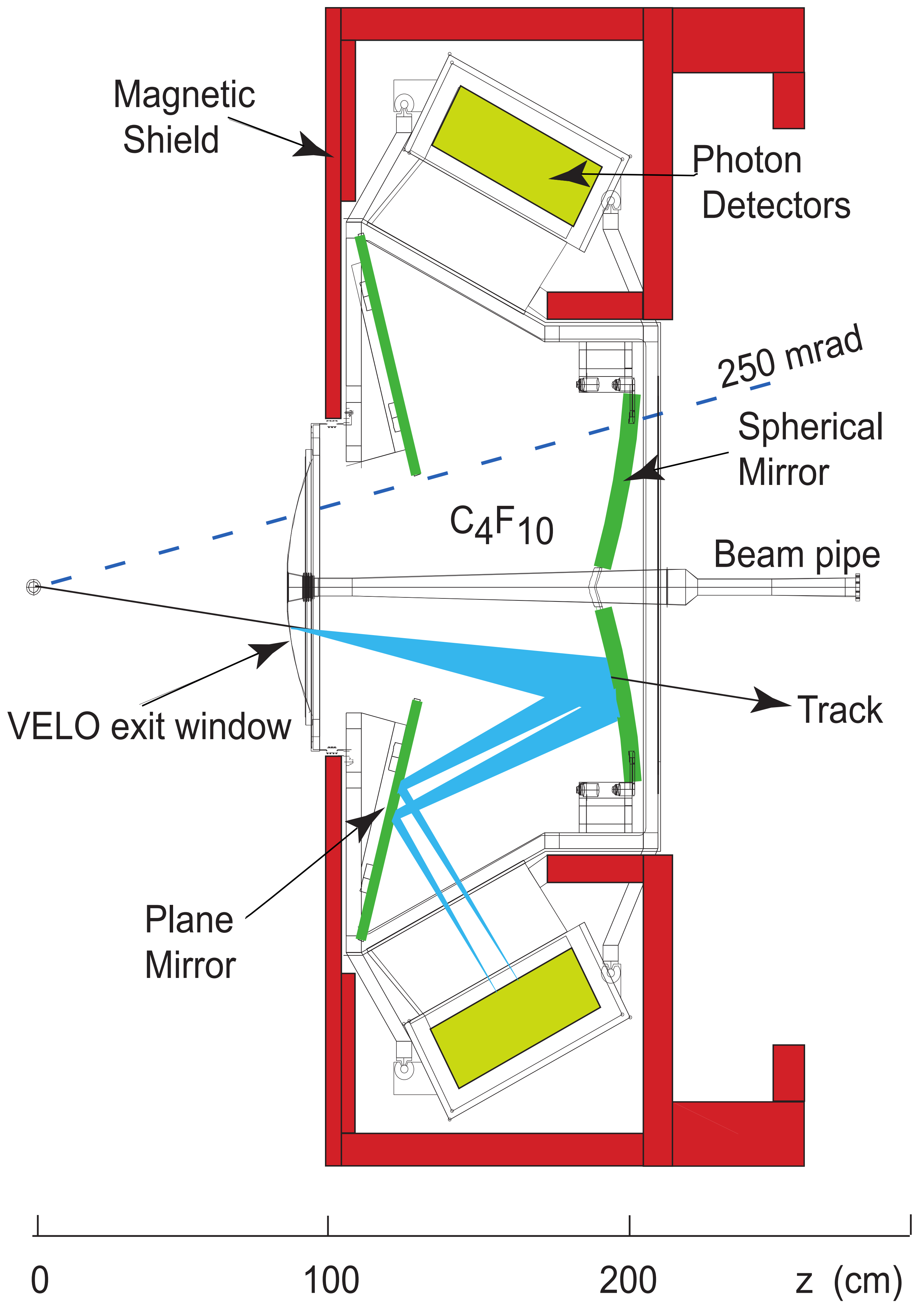
LHCb detector



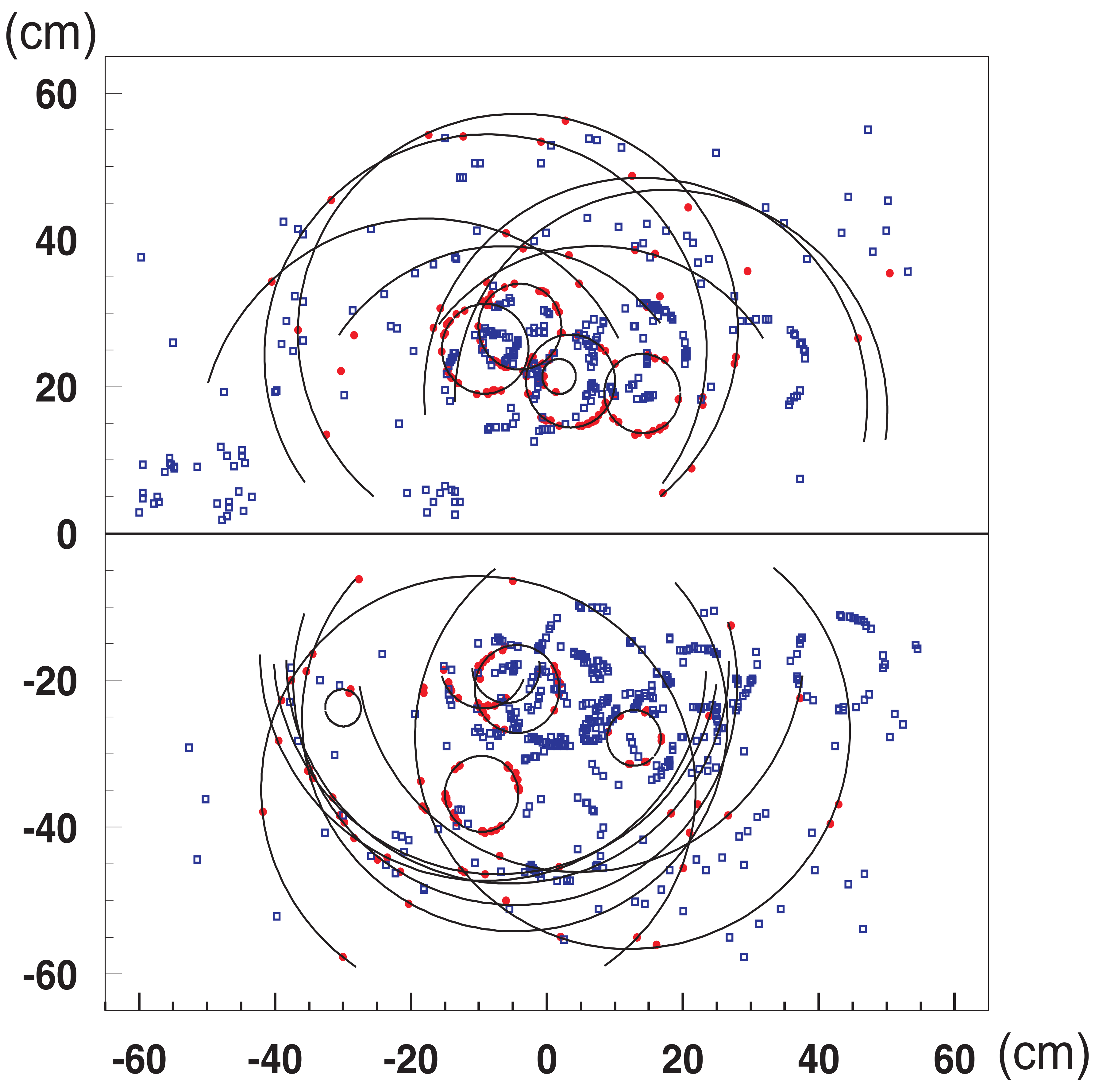
Ring Imaging Cherenkov Detectors (RICH)



The LHCb RICH Collaboration, Adinolfi, M., Aglieri Rinella, G. et al.
Eur. Phys. J. C (2013) 73: 2431



Information from RICH detectors



- PID with RICH is done with the maximum likelihood method
  - $\mathcal{L}(t_1, \ldots, t_N)$ – likelihood to observe a given picture, as a function of **all track PIDs**
    t_i – hypothesized particle type for track i
  - A hypothesis $(\underline{t_1}, \ldots, \underline{t_N})$ maximizing $\mathcal{L}$ is searched for
- For each track i, for each x ∈ {K, μ, e, p, below threshold}, quantities **DLLx** are then calculated as:

$$\log\mathcal{L}(t_k=\underline{t_k},\ k\neq i;\ t_i=\mathrm{x}) - \log\mathcal{L}(t_k=\underline{t_k},\ k\neq i;\ t_i=\pi)$$

RICH simulation

- Accurate RICH simulation involves:
  - Tracing the particles through the radiators and delta-electron generation
    - Delta-electrons contribute to Cherenkov light emissions
  - Cherenkov light generation
  - Photon propagation, reflection, refraction and scattering
  - Hybrid Photon Detector (photo-cathode + silicon pixel) simulation
- All this takes time & resources
- Given the growing demand on the number of simulated events, accurate simulation becomes unfeasible

RICH fast simulation

- A possible solution:
  - Bypass all those steps from Cherenkov light generation up to the high-level likelihood parameters (DLLs)
  - Learn the distribution of DLLs for given track parameters and sample from it, P(DLLs | <track params>)

Derkach et al, NIMA 2019 (01) 031

Generative Adversarial Networks (GANs)

x - input variables
y - target variables

P(DLLs | <track params>)

y **x**

- Random variable z
- Two deterministic functions (neural nets)
  - generator G(x, z)
  - discriminator D(x, y)
- Generator maps (x, z) to y^{gen}
- Discriminator distinguishes between (x, y^{gen}) and (x, y^{real})
- Training step («competition» between the two nets):
  - train discriminator to improve (x, y^{gen}) and (x, y^{real}) separation
  - train generator to increase the discriminator's error rate

Discriminator metric

- Some of the options for the discriminator metric:
  - Binary cross-entropy between the real and generated samples
    - Equilibrium when Jensen–Shannon divergence is minimized
    - Problematic for distributions with different support; mode collapse problems
  - Wasserstein (aka Earth Mover's) distance
    - Discriminator => «Critic» (evaluates the metric)
    - Naturally solves the non-equal support and mode collapse problems
    - Suffers from biased gradients

arXiv:1406.2661
arXiv:1701.07875

GAN (Cramér / energy distance)

- Cramér distance between distributions P and Q:

$$l_2^2(P,Q) := \int_{-\infty}^{\infty} (F_P(x) - F_Q(x))^2 \mathbf{d}x$$

- F_P and F_Q are CDFs
- This is (1/2 times) the 1-dimensional case of the Energy distance:

$$\mathcal{E}(X,Y) := 2\,\mathbb{E}\,\|X-Y\|_2 - \mathbb{E}\,\|X-X'\|_2 - \mathbb{E}\,\|Y-Y'\|_2$$

X,X'~P and Y, Y'~Q

- A GAN using this metric preserves all the nice properties of Wasserstein GAN, while solving the biased gradients problem

arXiv:1705.10743

RICH fast simulation

- Dimensionality of P(DLLs | <track params>)
  - 5 output DLLs
  - input: track momentum and pseudorapidity (+2)
  - input: omitting ϕ for simplicity (+0)

RICH fast simulation

- Dimensionality of P(DLLs | <track params>)
  - 5 output DLLs
  - input: track momentum and pseudorapidity (+2)
  - input: omitting ϕ for simplicity (+0)
- DLLs depend not only on the given track's parameters
  - overlapping circles at higher occupancies
  - **input: total number of tracks in that event (+1)**

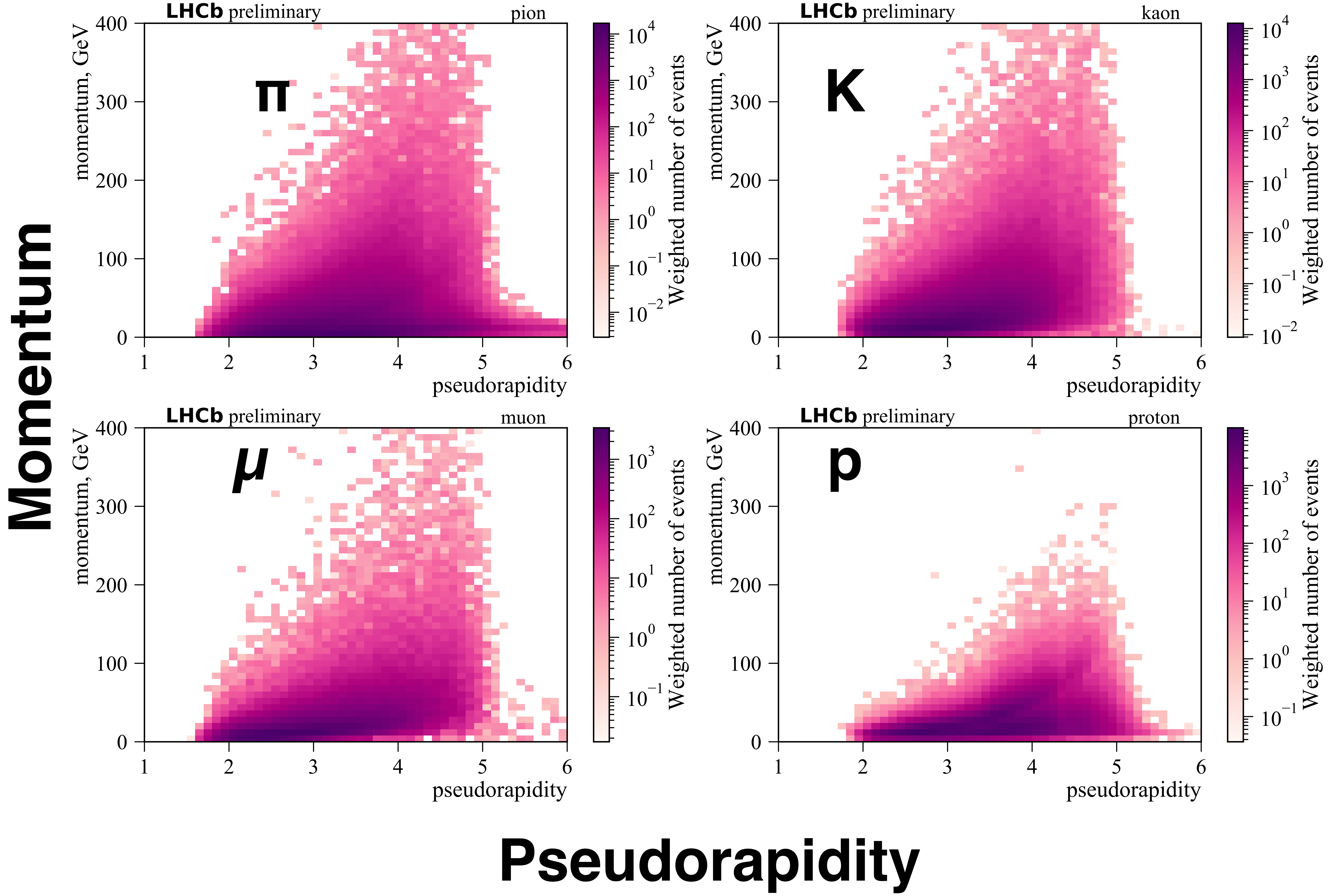
RICH fast simulation

- Dimensionality of P(DLLs | <track params>)
  - 5 output DLLs
  - input: track momentum and pseudorapidity (+2)
  - input: omitting ϕ for simplicity (+0)
- DLLs depend not only on the given track's parameters
  - overlapping circles at higher occupancies
  - **input: total number of tracks in that event (+1)**
- Training on real data (calibration channels)
  - using sPlot technique[1] to extract signal distributions
  - loss function is weighted (with possibly negative weights)

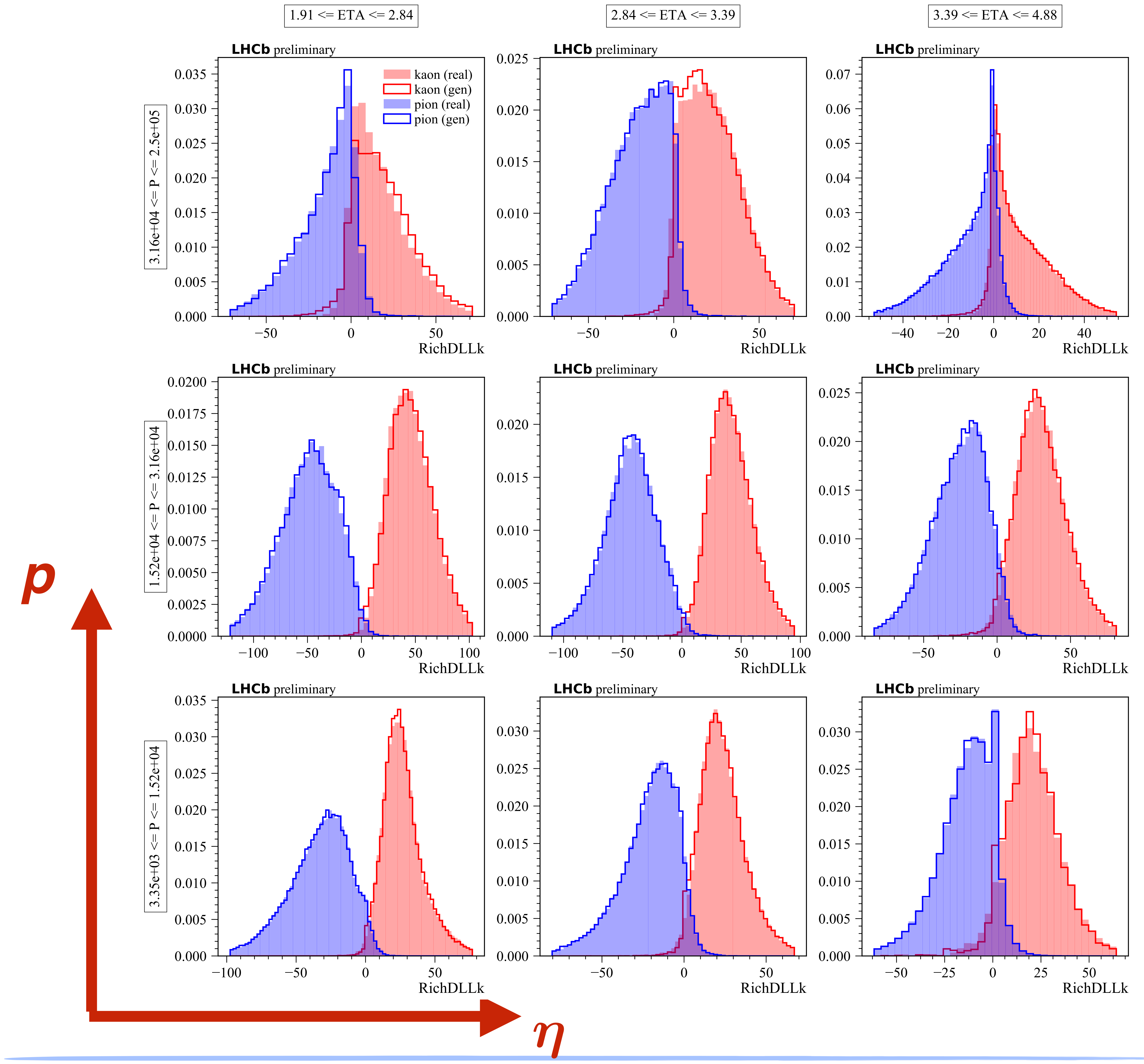
[1]Pivk, Muriel et al. Nucl.Instrum.Meth.A 555 (2005) 356-369

Implementation details and input parameter distributions

- 10 hidden fully-connected layers for both generator and discriminator
  - 128 neurons each
  - ReLU activation
- 64-dimensional latent space (noise shape)
- 256-dimensional discriminator output
- 15 discriminator updates per 1 generator update
- RMSProp optimizer, exp decaying learning rate

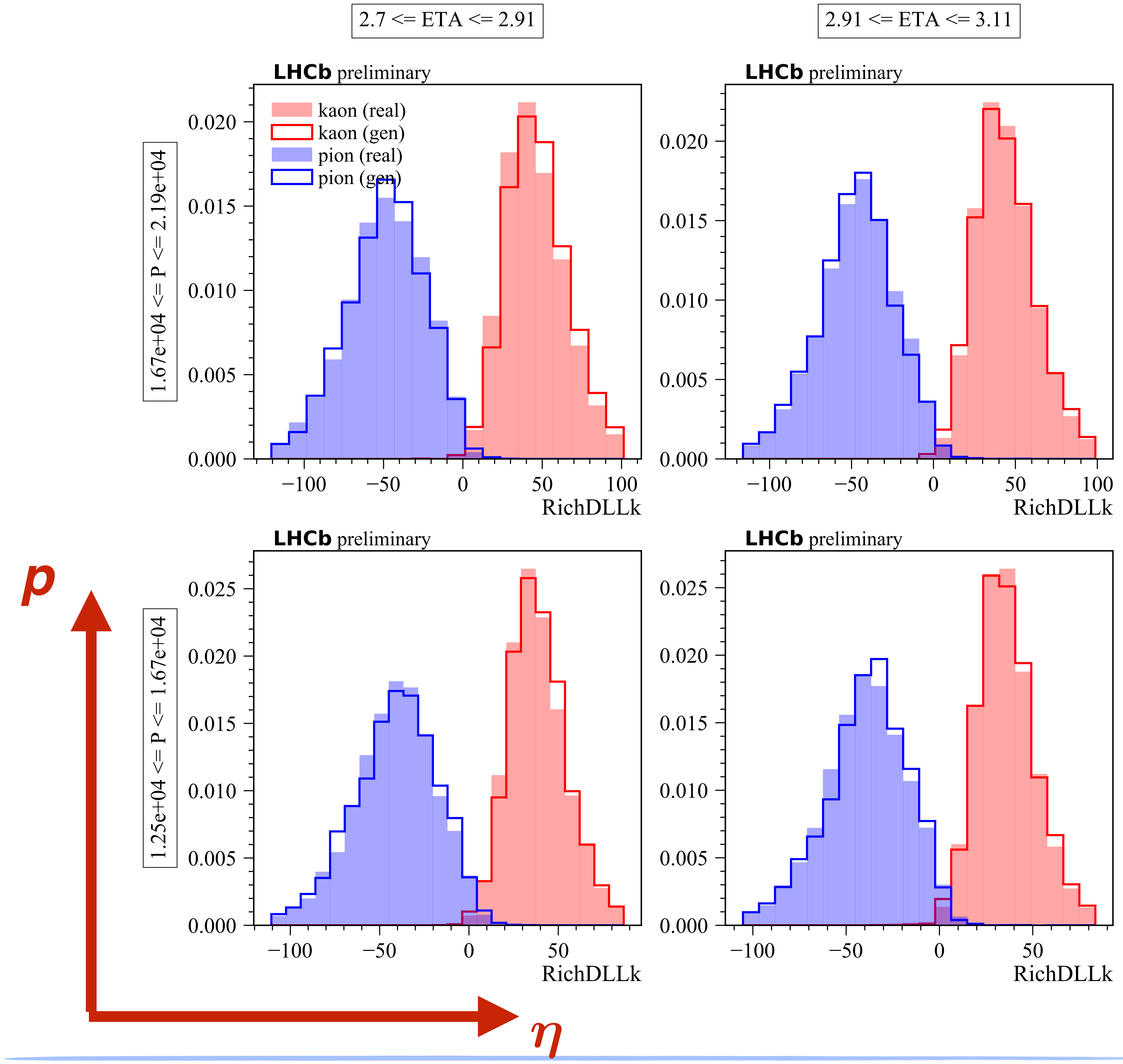


RichDLLk (π vs K)

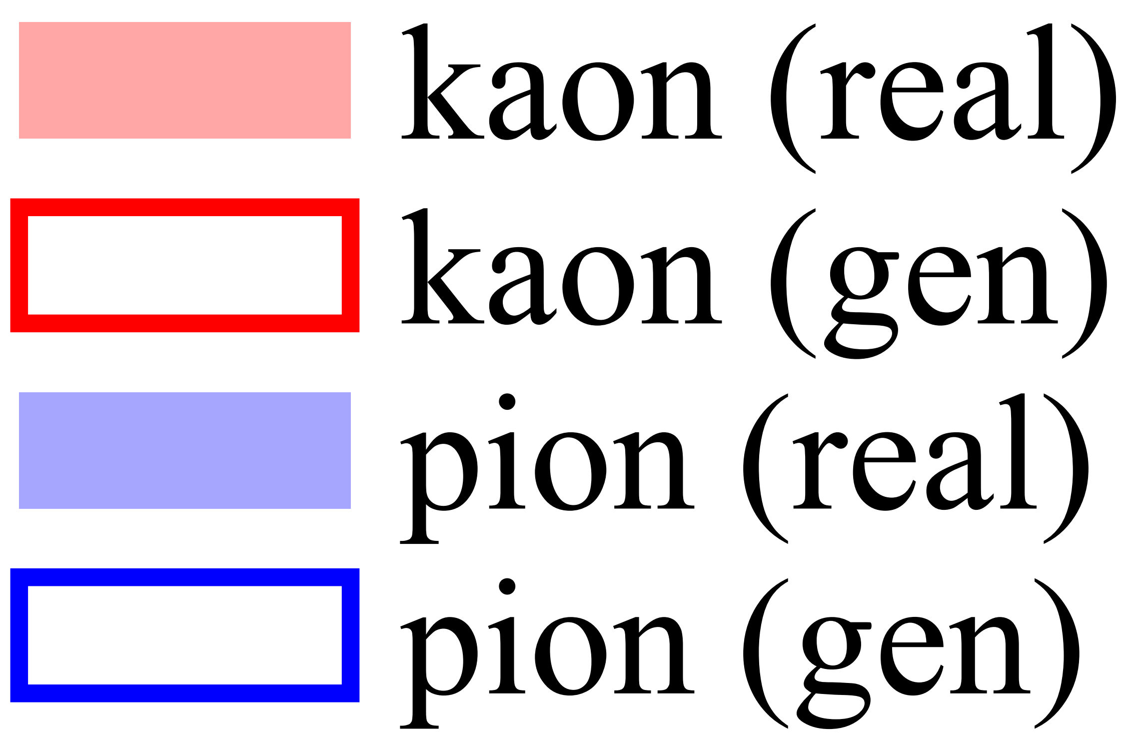


kaon (real)
kaon (gen)
pion (real)
pion (gen)

3x3 bin plot over full P-ETA range

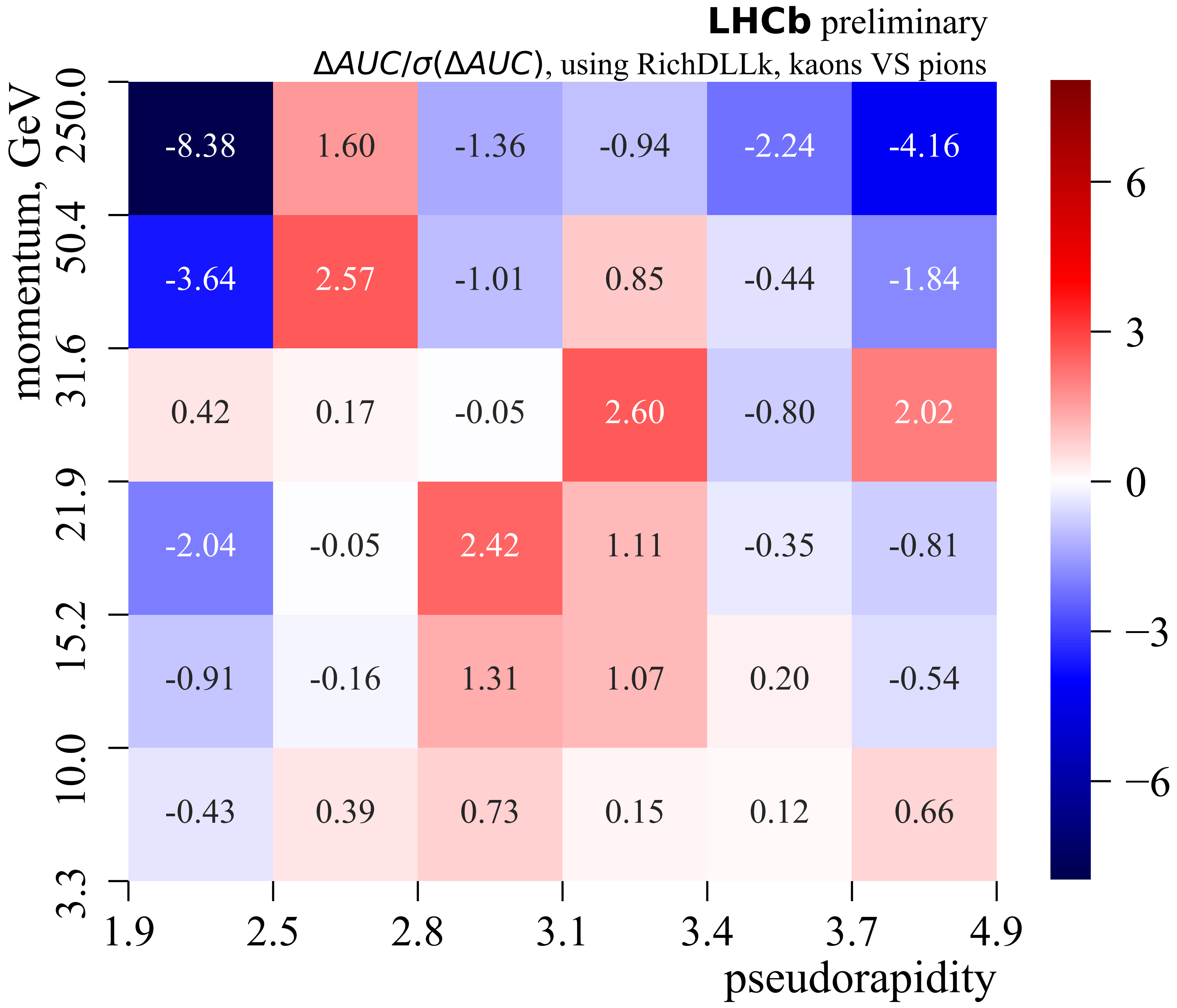


RichDLLk (π vs K)

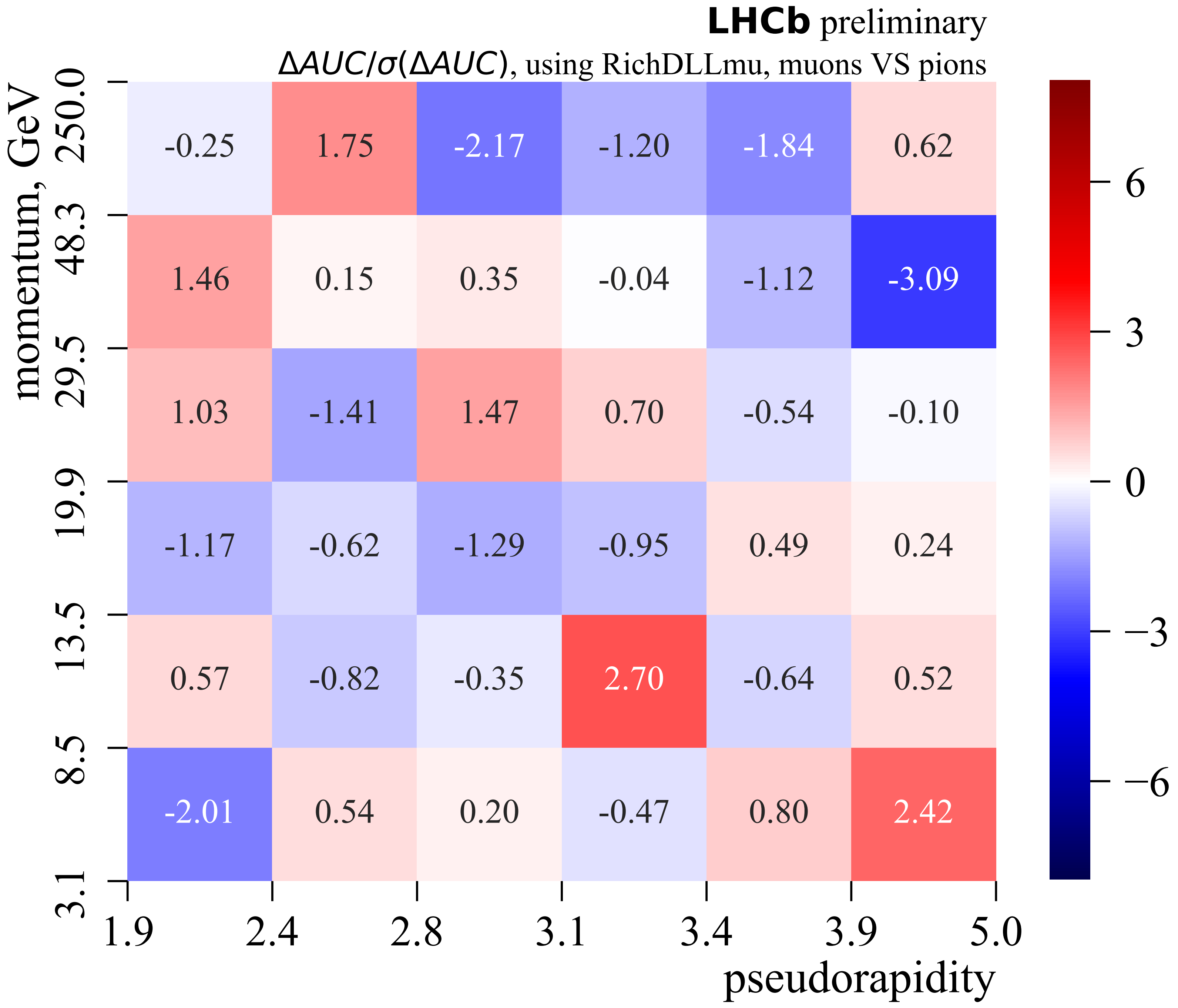


zoomed in to the most populated region

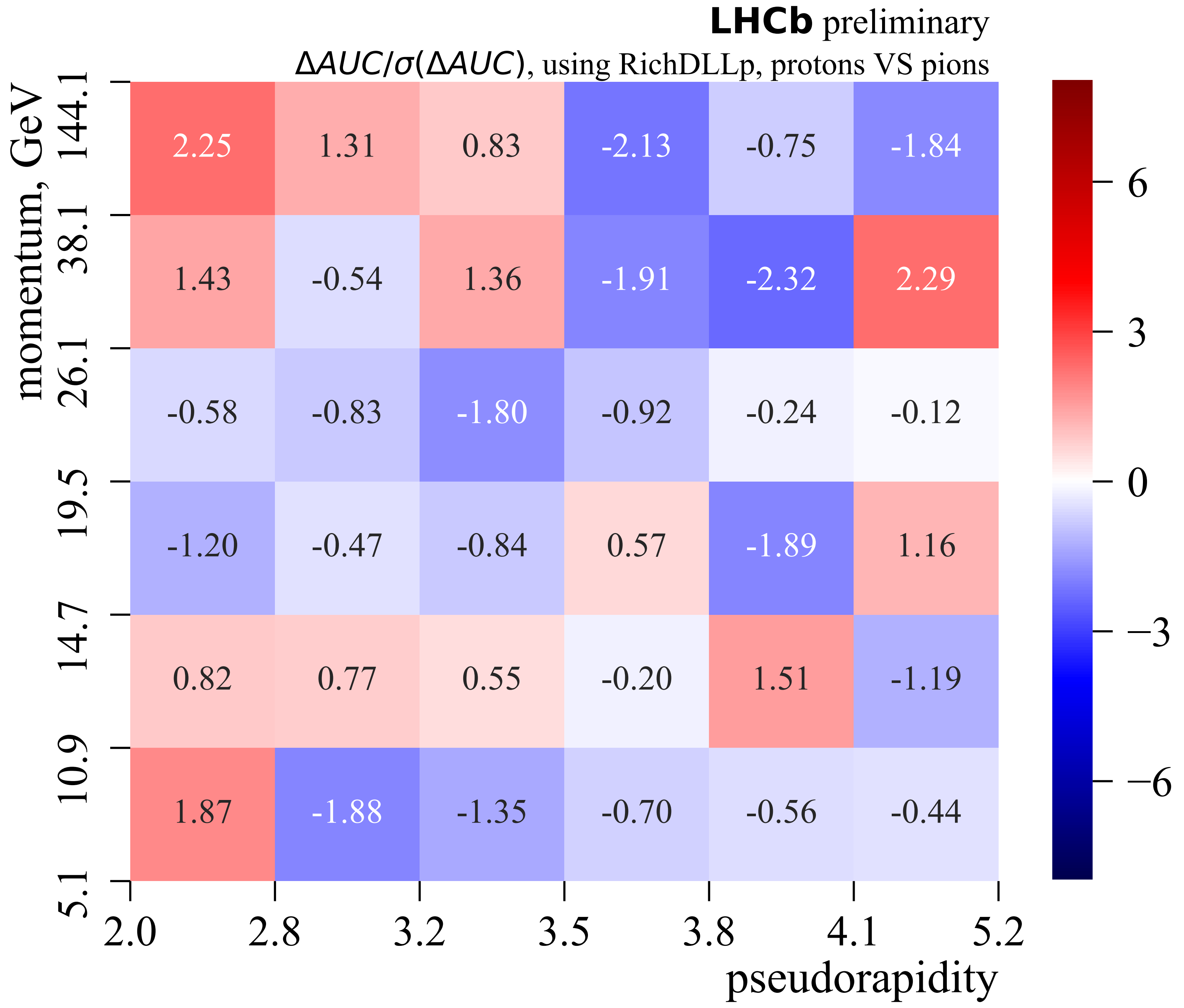
Differences between AUCs for real and generated samples (divided by error)



K vs π, using RichDLLk



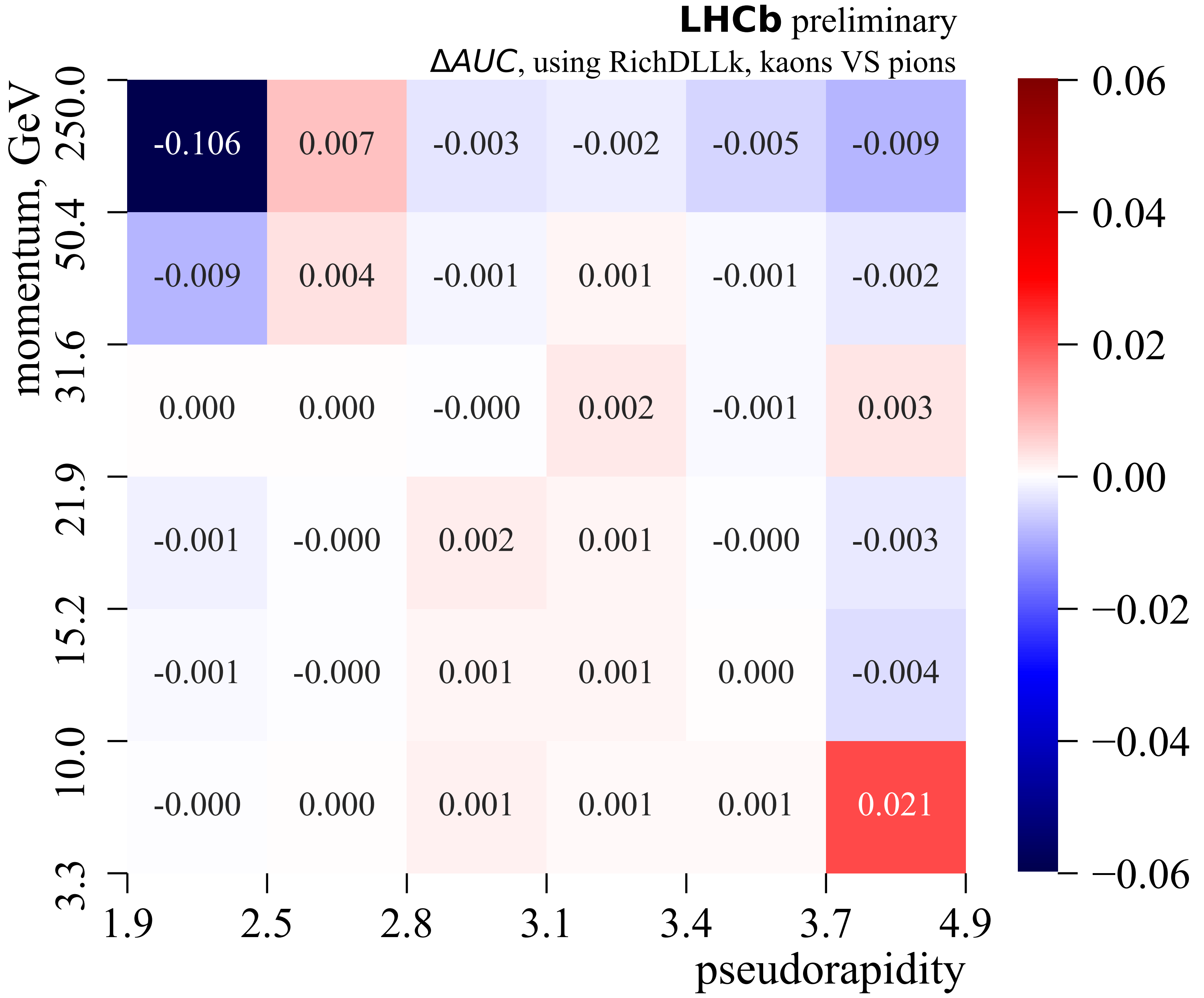
μ vs π, using RichDLLmu



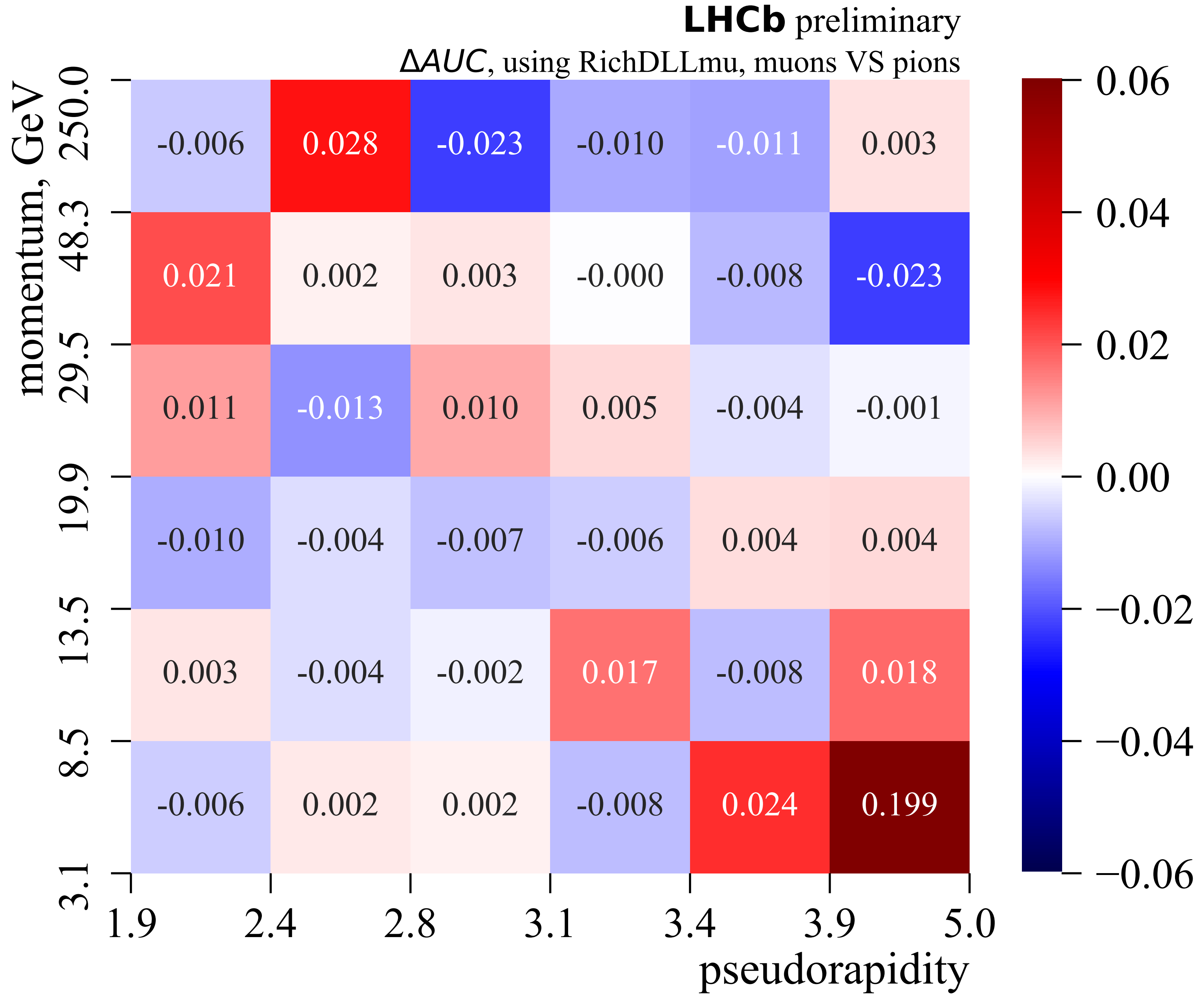
p vs π, using RichDLLp

- Errors estimated using bootstrap technique
- Most differences are within just a few sigmas, larger deviations at low-stat regions

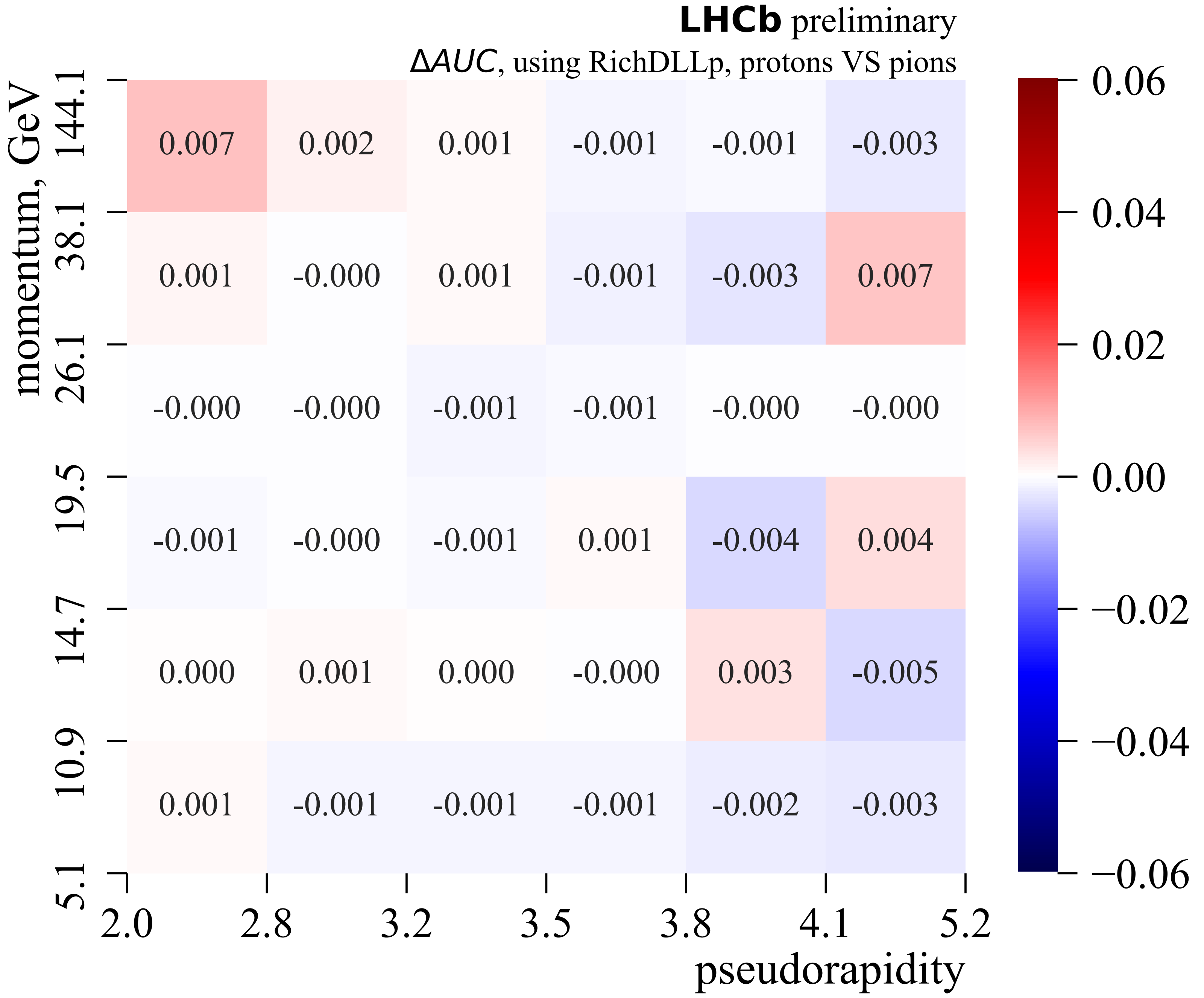
Differences between AUCs for real and generated samples (absolute, generated – real)



K vs π, using RichDLLk



μ vs π, using RichDLLmu



p vs π, using RichDLLp

- Absolute differences between AUCs are mostly in the 0.001-0.01 range

Summary

- GANs are a promising tool for fast simulation models
- Can be trained in a background-contaminated environment, with the help of sPlot technique
  - negative weights don't cause divergencies in our experiments
- Our model is doing well judging by the looks of generated vs real distributions
  - Some imperfections seen in the low-stat regions
  - AUC differences look reasonable
- Ongoing work to implement our model within the fast simulation framework of LHCb
  - Looking forward to evaluate our model in a real analysis environment

Backup

GAN (JS)

- Possible discriminator and generator losses – binary cross-entropy:

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

real samples **discriminator NN** **noise samples** **generator NN**

arXiv:1406.2661

- This leads to equilibrium when Jensen-Shannon divergence between real and generated samples is minimized
- Problems:
  - vanishing gradients when discriminator too powerful
  - mode collapse (generating only a subset of the target distribution)

GAN (Wasserstein)

arXiv:1701.07875

- Another possible metric: Wasserstein distance (Earth Mover's distance)

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma}\big[\, \|x - y\| \,\big]$$

- γ – «optimal transport plan»
- This should solve the mode collapse and vanishing gradients problems
- Solution may not be optimal due to biased gradients (see arXiv: 1705.10743)