Fast, accurate, and precise neural networks for high-dimensional calorimeters

Luigi Favaro

from 2305.16774, 2312.09290, 2405.09629

10.09.2024 - Louvain-la-Neuve

UNIVERSITÄT HEIDELBERG ZUKUNFT SEIT 1386

Collaborative Research Center TRR 257

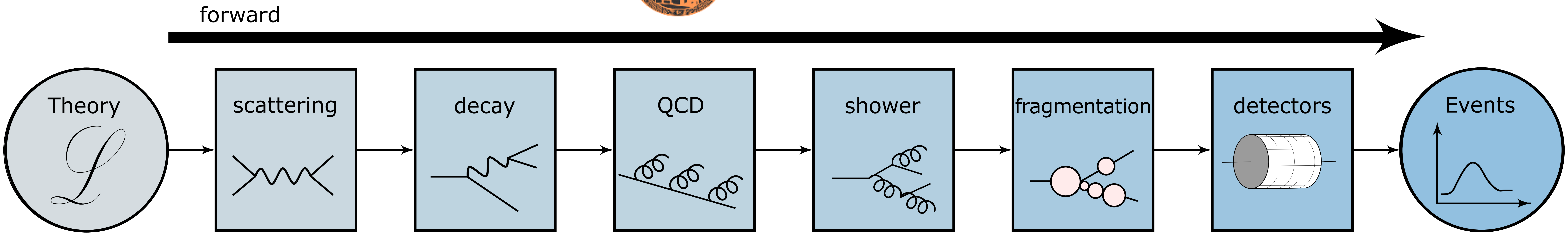
Particle Physics Phenomenology after the Higgs Discovery

Deutsche Forschungsgemeinschaft

German Research Foundation

Simulation Chain

Pythia/Sherpa/Herwig

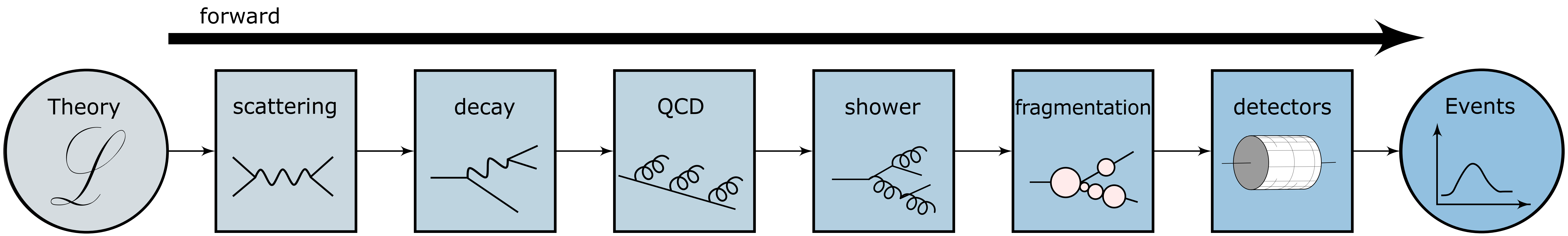


Madgraph

Geant4/Delphes

- First-principled simulations, from QFT to events

Simulation Chain

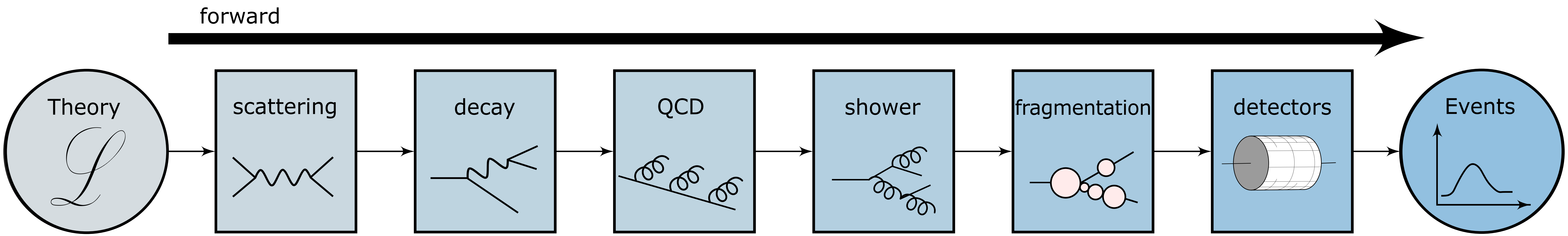


# of particles: $\mathcal{O}(10)$ $\mathcal{O}(10^2 - 10^3)$ $\mathcal{O}(10^4)$

- In the last step we reconstruct $\mathcal{O}(10)$ objects from the detector readouts;
- High-dimensional spaces which pose different questions.

Simulation Chain

end-to-end forward surrogate model



Efficient importance sampling
from complex distribution

Slow forward model simulation

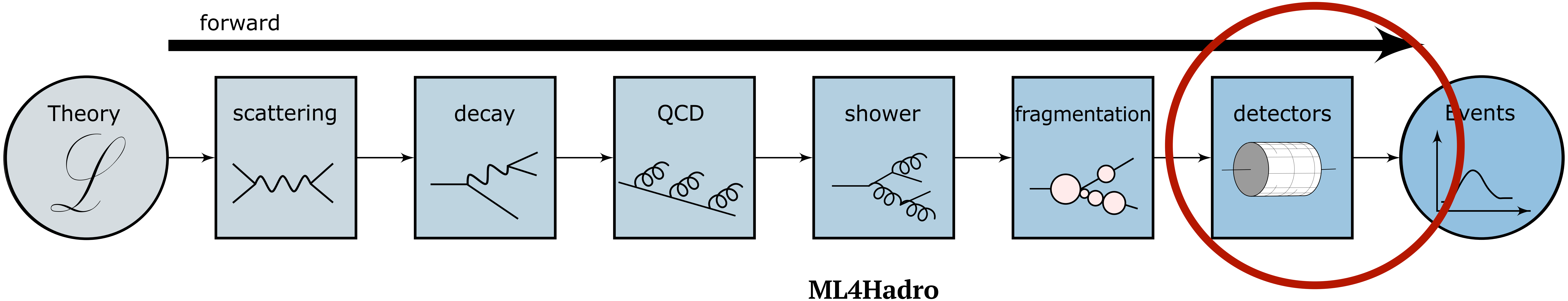
inverse problems: from events back to partons

Simulation Chain

Event generation

- Data amplification;
- Efficient storage of events via the network weights.

this talk



ML4Hadro

- Model-dependence of the hadronization step;
- data-driven approaches to hadronization.

MadNIS

- Neural importance sampling;
- differentiable programming.

Detector simulation

- Speed-up full sim, e.g. AtlFast;
- surrogate models for future calorimeters
- ML-enhanced reconstruction.

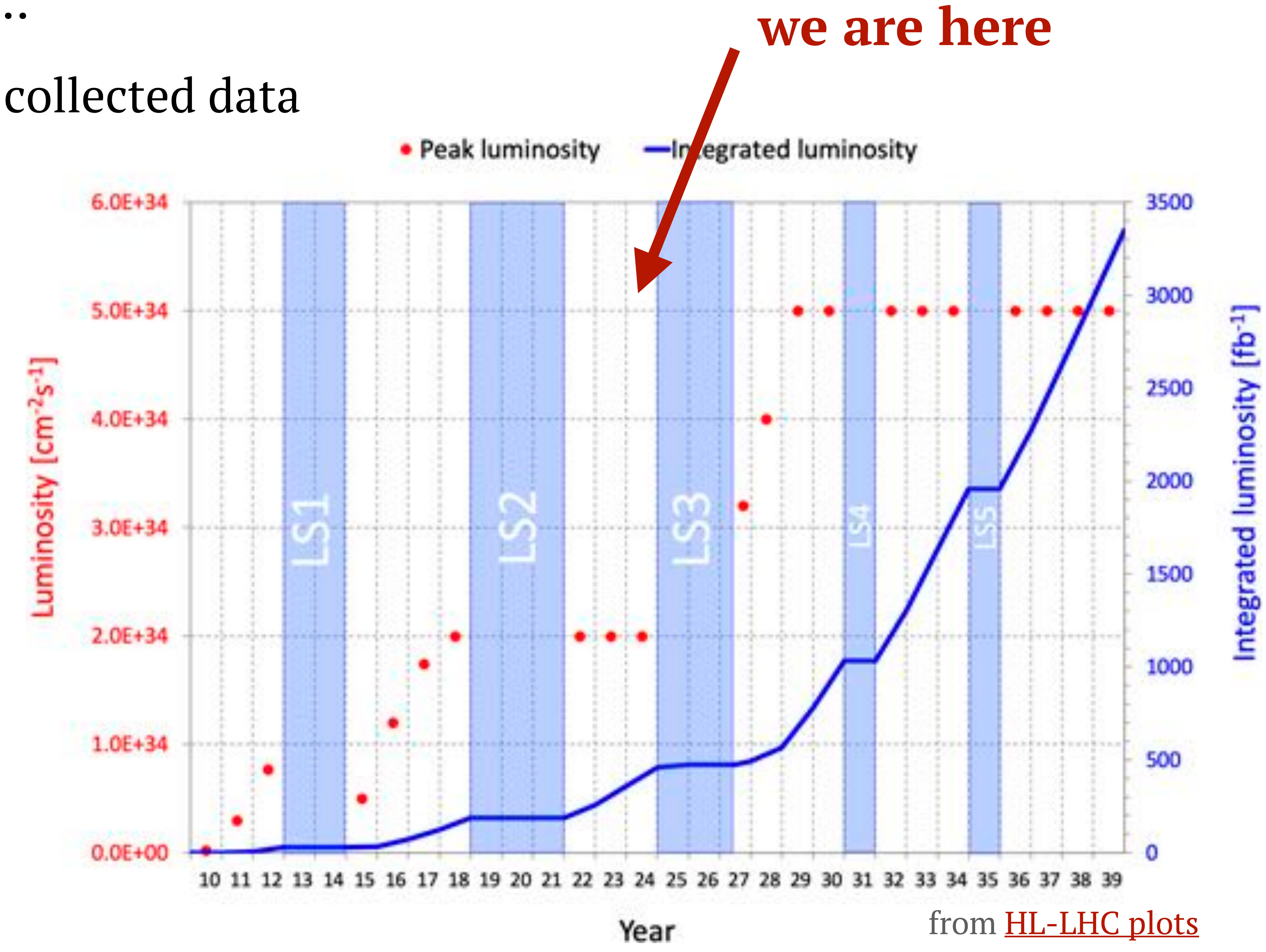
LHC future plan

- The high-luminosity data taking phase is close…
- Simulations will have to match the statistics of collected data

Need for fast generators…

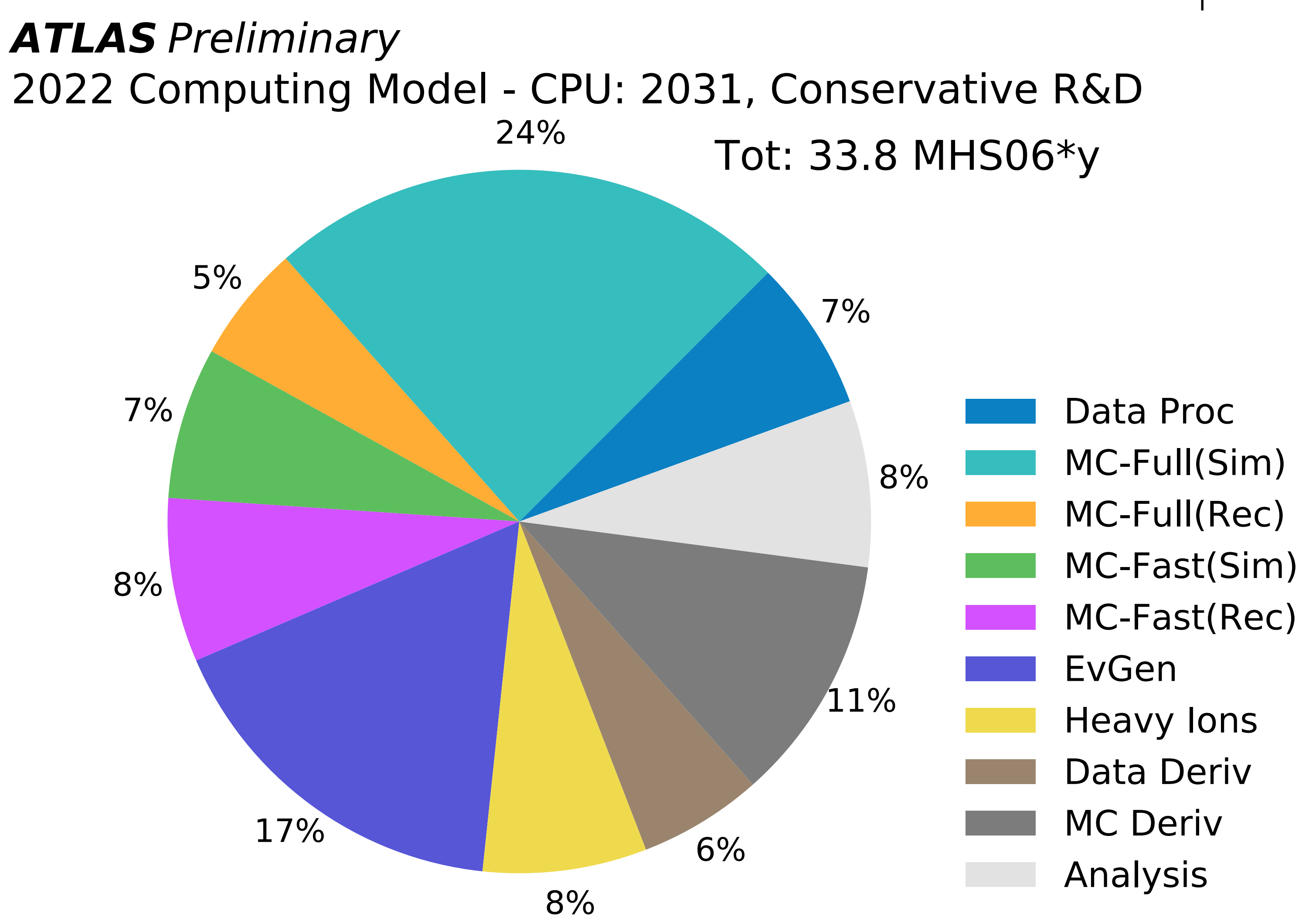
… which are still (more) accurate and precise

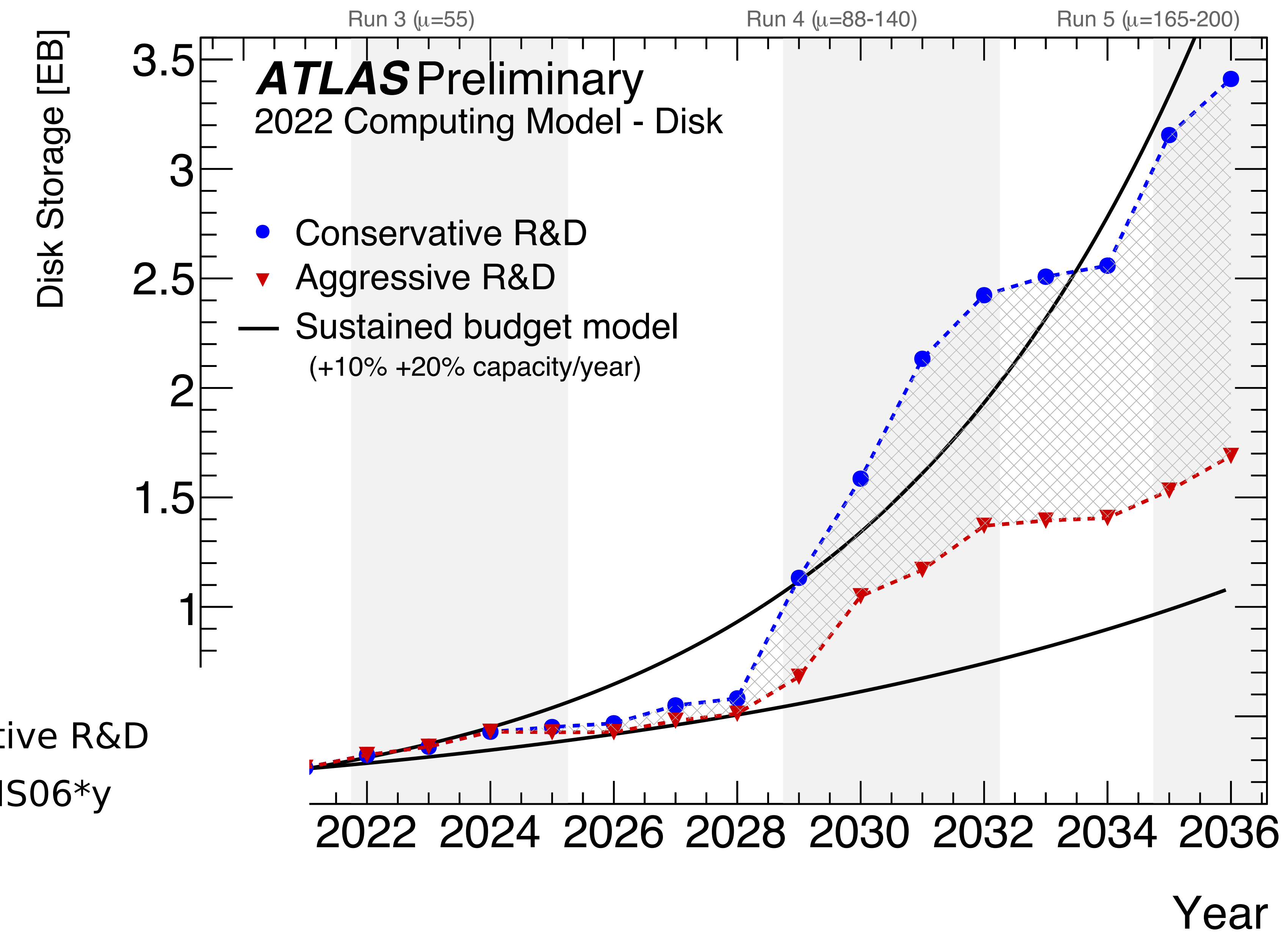
find new physics!
(or rather understand LHC data)

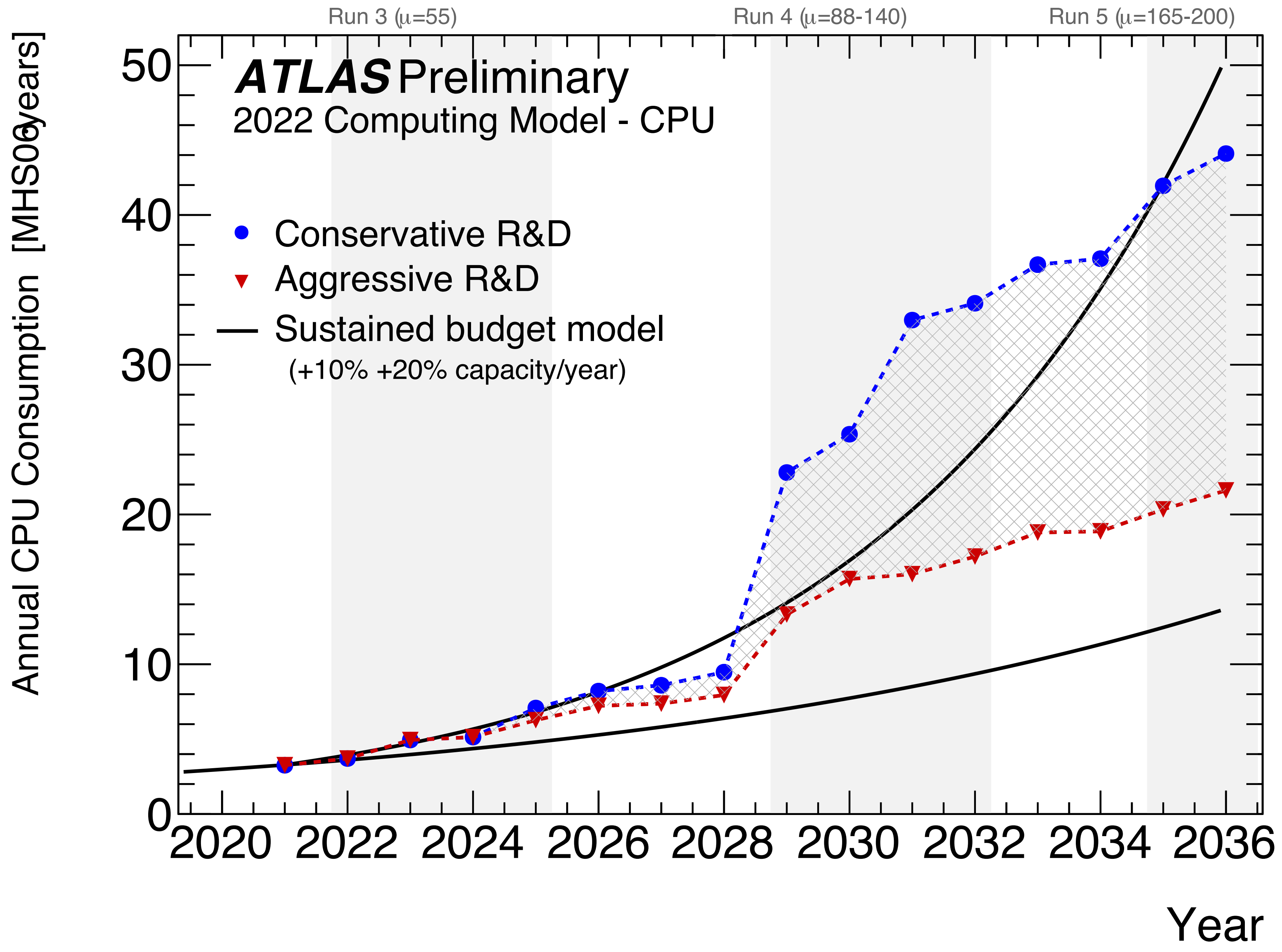


from HL-LHC plots

Detector simulation







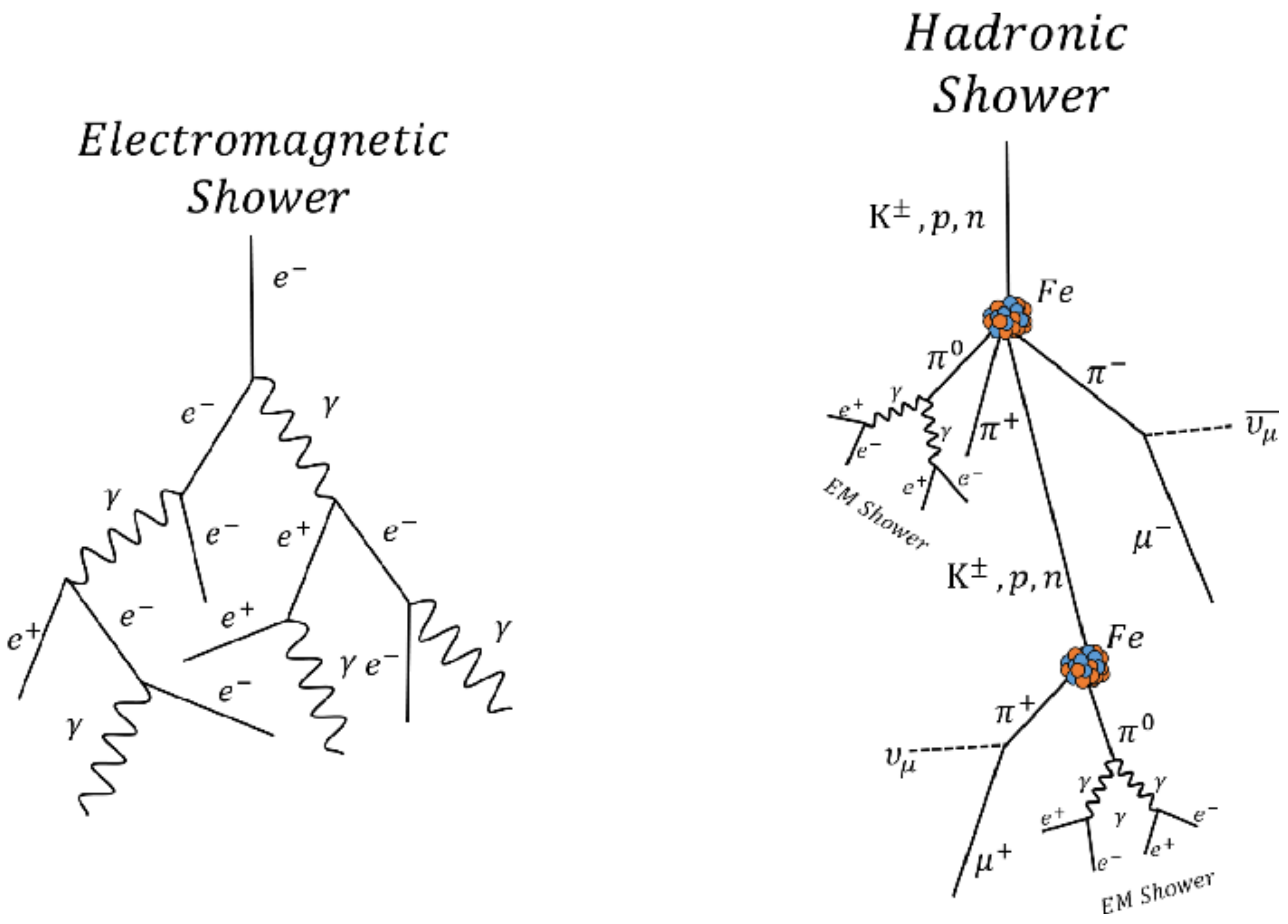
- A conservative R&D approach will not be sustainable
- $\sim 44\,\%$ of computing budget goes into MC full/fast simulation

from ATLAS computing roadmap

What are we simulating?

The leading speed bottleneck is the simulation of calorimeter showers

from here



Incident particle drastically changes the shower:

- $\gamma/e^{+/-}$: electromagnetic showers
  $\longrightarrow$ only Bremsstrahlung and pair-production
- hadrons: hadronic showers
  $\longrightarrow$ complex, non-perturbative phenomenology

Physics depends on the incident particle

Expensive?

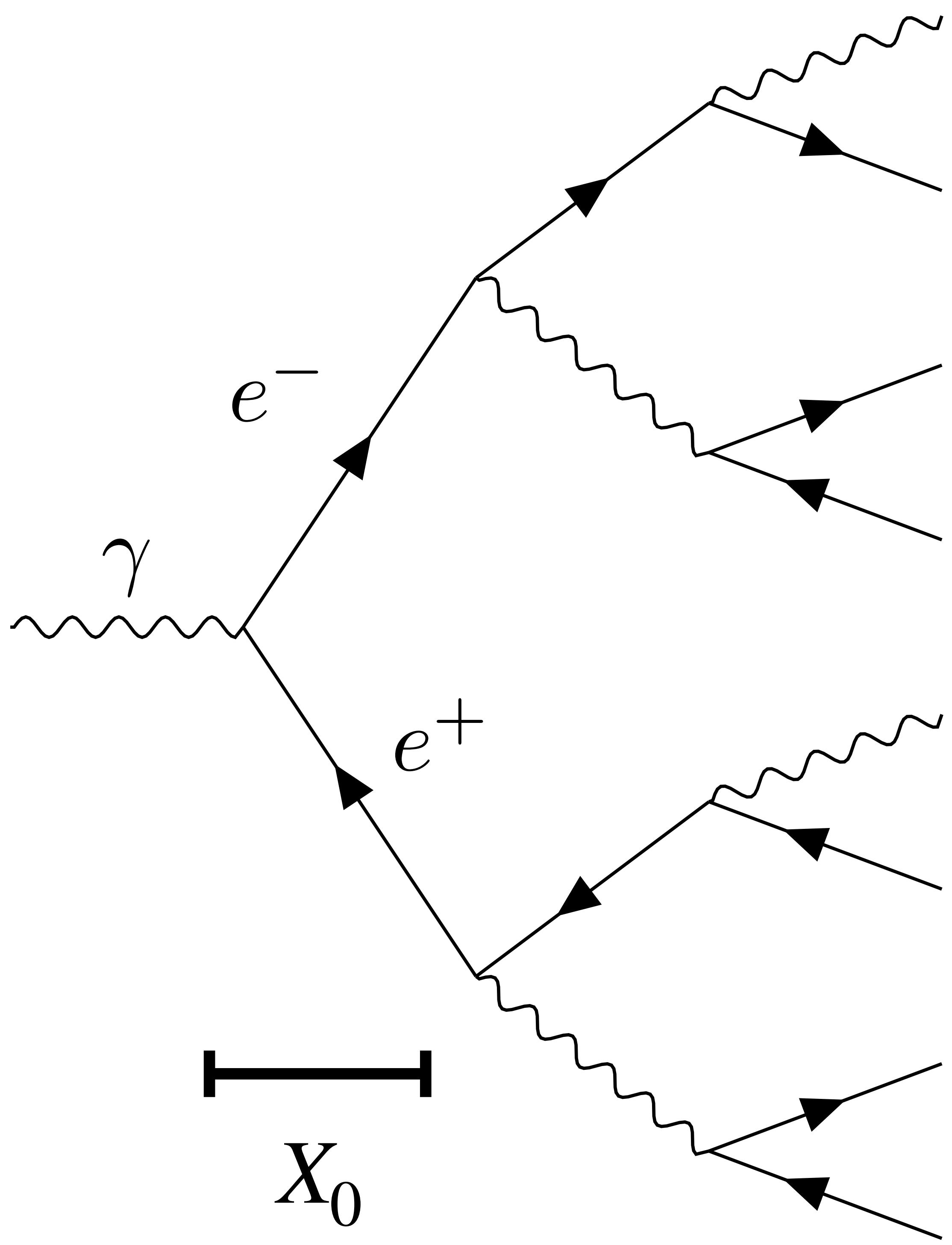
Simple description of an EM shower:

- Assuming one interaction every X_0 with equal energy splitting
- shower stops at the critical energy E_c
- mean energy deposition: $\left\langle \frac{dE}{dx} \right\rangle = -\frac{E}{X_0}$

Features:

- at step $t : N(t) = 2^t$
- $t_{\max} \propto \log E_0$

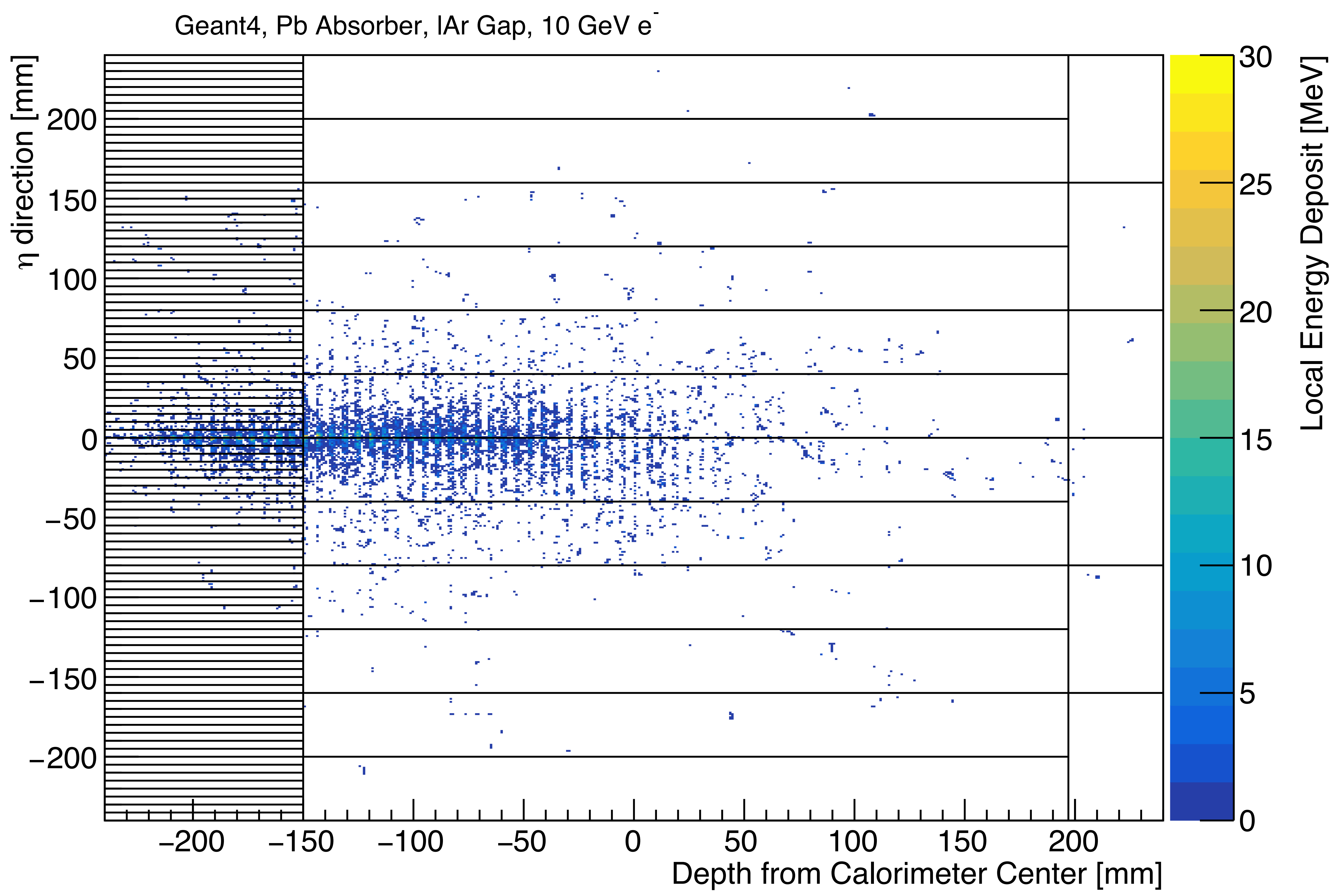
Exponential scaling of particle #



Geant4

Geant4 is the main toolkit for full sims:

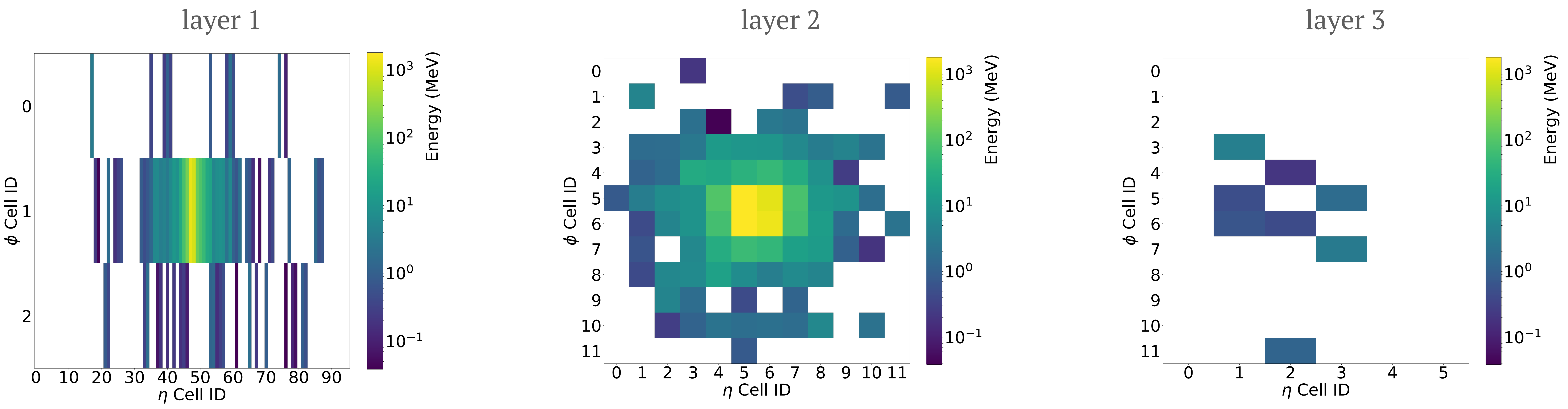
- Defines a custom geometry
- stochastically particles interact with the material
- keeps track of all of them



Energy deposition from hits in one cell are summed together

CaloGAN data

Geant4



CaloGAN data

- The shower is represented as 4-tuple (E, x, y, z)
- There can be no-energy deposition
- visualized as an image

This is known as "voxelised" approach, can we speed up this process?

Difficulties:

- High-dimensional
- non-trivial correlations
- energy conservation

ML to the rescue

Measure for the difference between two distributions: Kullback-Leibler divergence:

$$\mathrm{KL}(p\,|\,q) = \int dx\, p(x)\, \log \frac{p(x)}{q(x)}$$

- positive definite
- zero if $p(x) = q(x)$

Turn this expression into a loss function:

$$\begin{aligned}\mathrm{KL}(p\,|\,q_\theta) &= -\int dx\, p(x)\, \log q_\theta(x) + c \\ &= -\langle q_\theta(x)\rangle_{x\sim p(x)}\end{aligned}$$

ML to the rescue

Assume $x \sim p_{\text{data}}(x)$ and model $p_\theta(x)$:

$$\mathscr{L} \approx -\frac{1}{N}\sum_{i=1}^{N} \log p_\theta(x_i) \qquad x_i \sim p_{data}(x)$$

Generative "Flow" networks:

- transform from input space to a latent space $\quad G_\theta(x)\colon x \to z$
- easy to sample from the latent distribution $\quad \bar{G}_\theta(z)\colon z \to x \qquad x, z \in \mathbb{R}^d,$

ML to the rescue

Change of variable formula:

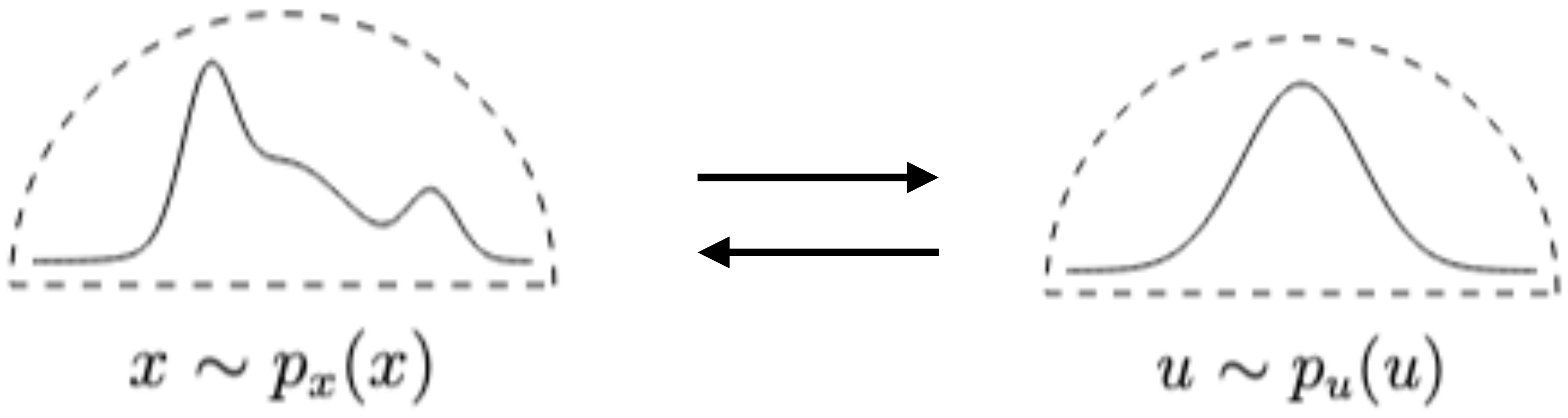
$$p_{lat}(z) = p_\theta(G_\theta(x)) \left| \frac{\partial G_\theta(z)}{\partial z} \right|$$

Typical choice is a standard Gaussian latent space:

$$\log p_{lat}(z) = \log(\sqrt{2\pi}) + \frac{z^2}{2}$$

Aim to a loss function that looks like:

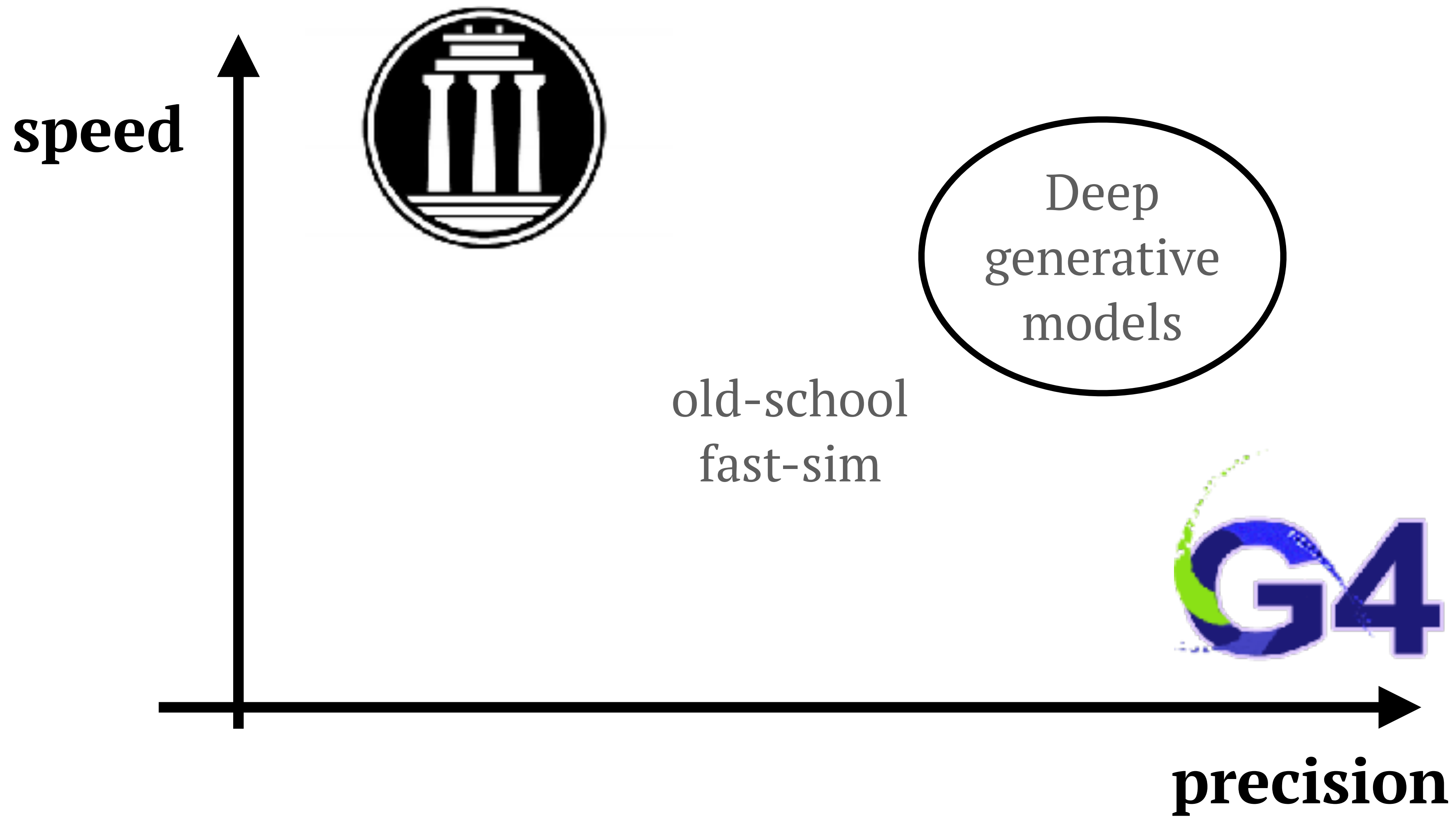
$$\mathscr{L}_F = -\left\langle \log p_{lat}(\bar{G}_\theta(x)) + \log \left| \frac{\partial \bar{G}_\theta}{\partial x} \right| \right\rangle_{p_{data}} .$$



Modern generative networks

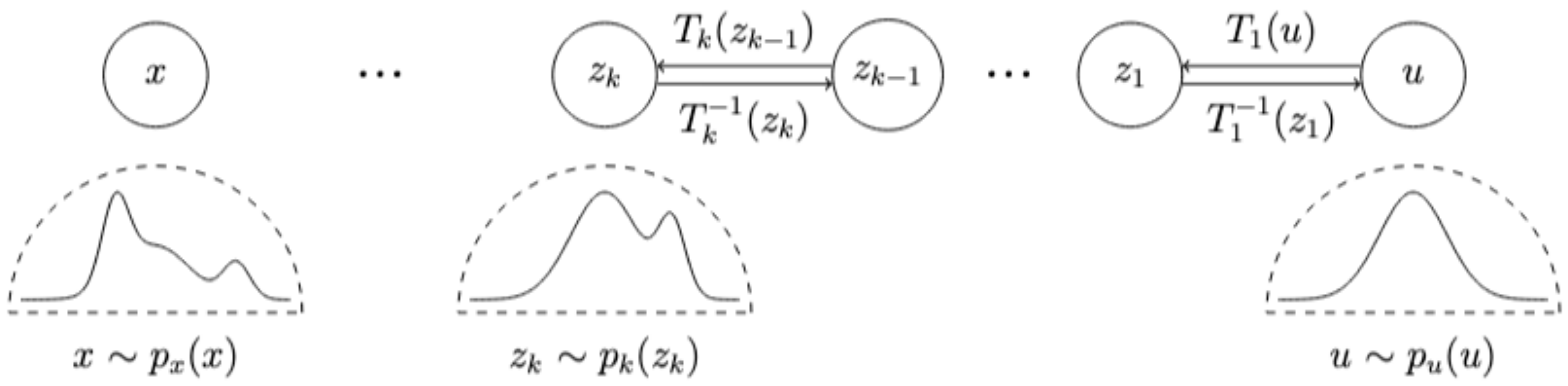
Modern generative networks:

- Complex architectures but still fitting functions
- provided data, approximate Geant4
- speed and precision are key
- tradeoff between speed and precision



Normalizing Flows

figure from future CaloChallenge white paper



Normalizing flows define a discrete number of invertible transformations

Choice of the transformation is crucial $\longrightarrow$ Jacobian has to be tractable

Two popular choices:

- Masked Autoregressive Flows (MAF)
- Invertible Neural Networks (INN) (aka coupling block)

see also CaloFlow

Coupling blocks

Coupling block transformation:

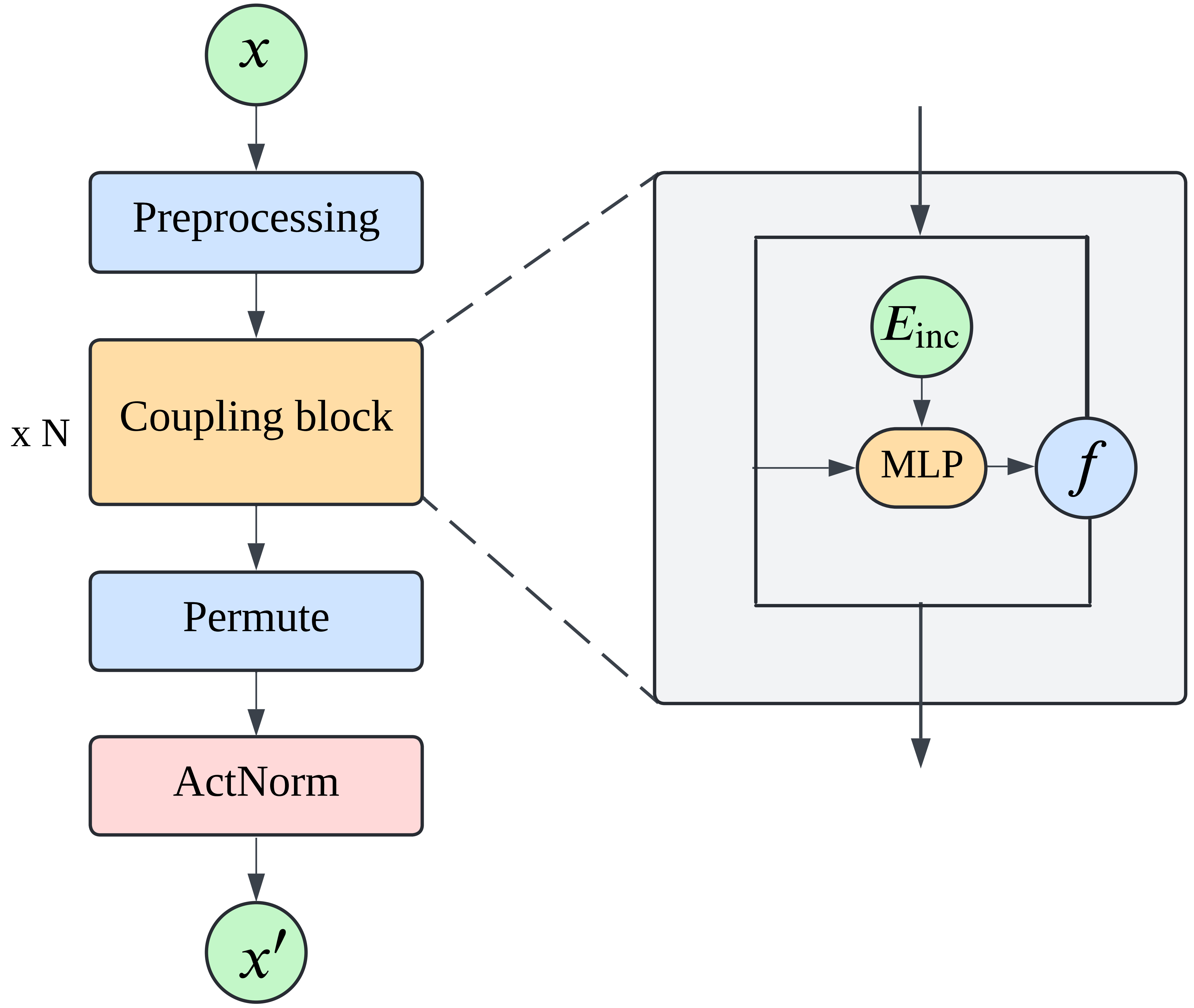
$$\begin{cases} y_i = x_i & i \in 1,\ldots,d \\ y_i = f_\theta(x_i \,|\, x_1, \ldots, x_d) & i \in d+1,\ldots,D, \end{cases}$$

The corresponding Jacobian is

$$\frac{\partial y}{\partial x} = \begin{pmatrix} I_d & (\neq 0) \\ 0 & \frac{\partial y_i}{\partial x_i} \end{pmatrix}$$

Determinant is calculated in $\mathcal{O}(N^2)$ operations

$\longrightarrow$ fast in both sampling and inference direction



Conditional Flow Matching

Promote the discrete transformation to a continuous one:

$$\frac{dx(t)}{dt} = v(x(t), t) \quad \text{with} \quad x \in \mathbb{R}^d \qquad\qquad \frac{\partial p(x,t)}{\partial t} + \nabla_x \big[p(x,t)v(x,t)\big] = 0 \, .$$

We want to impose the boundary conditions for $p(x, t)$:

$$p(x,t) \to \begin{cases} \mathcal{N}(x; 0,1) & t \to 1 \\ p_{data}(x) & t \to 0 \, . \end{cases}$$

Need to define the training trajectories

⟶ linear, simplest choice

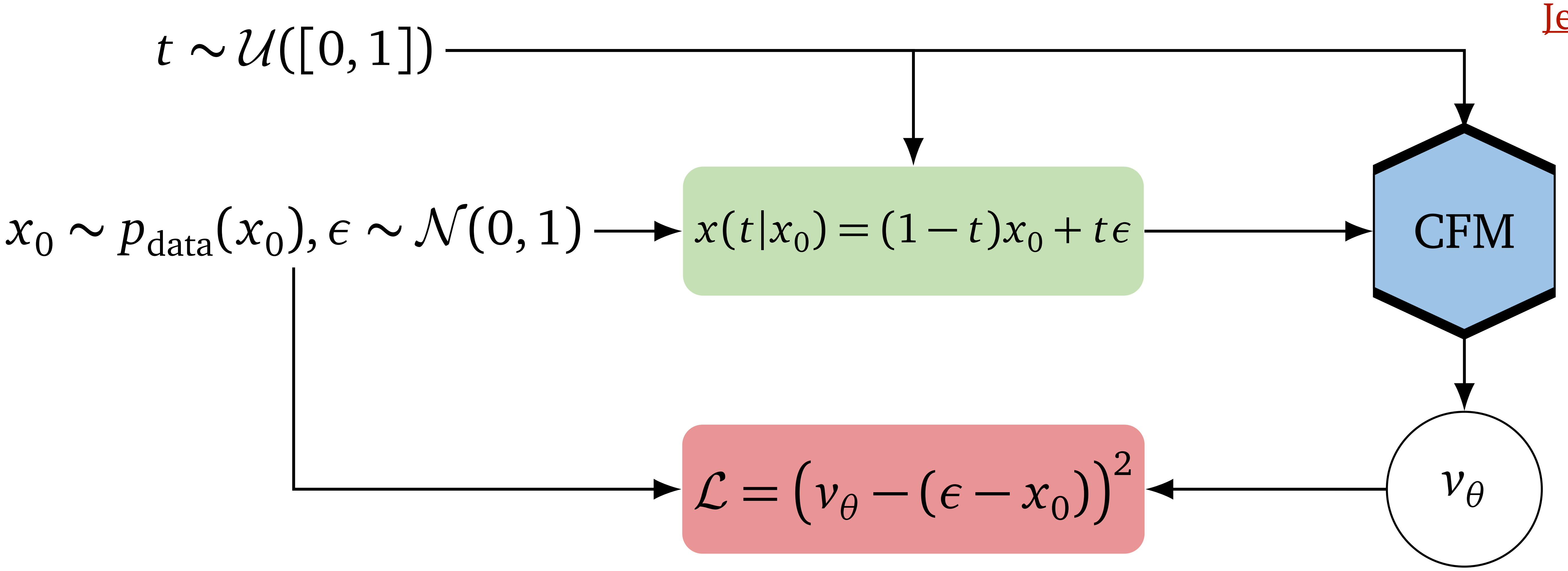
$$x(t\,|\,x_0) = (1-t)x_0 + t\epsilon \qquad \epsilon \sim \mathcal{N}(0,1)$$

Learn this velocity field with a NN:

$$\mathcal{L} = ||\,v(x,t) - v_\phi(x,t)\,||_{L_2}$$

Conditional Flow Matching

Jet diffusion versus JetGPT



$$\mathscr{L}_{\text{CFM}} = \left\langle \left[v_\phi((1-t)x_0 + t\epsilon, t) - (\epsilon - x_0) \right]^2 \right\rangle_{U(0,1), \mathcal{N}, p_{data}} .$$

Sample solving the differential equation numerically: $x(t=0) = x(t=1) - \int_0^1 v_\phi(x,t)dx$

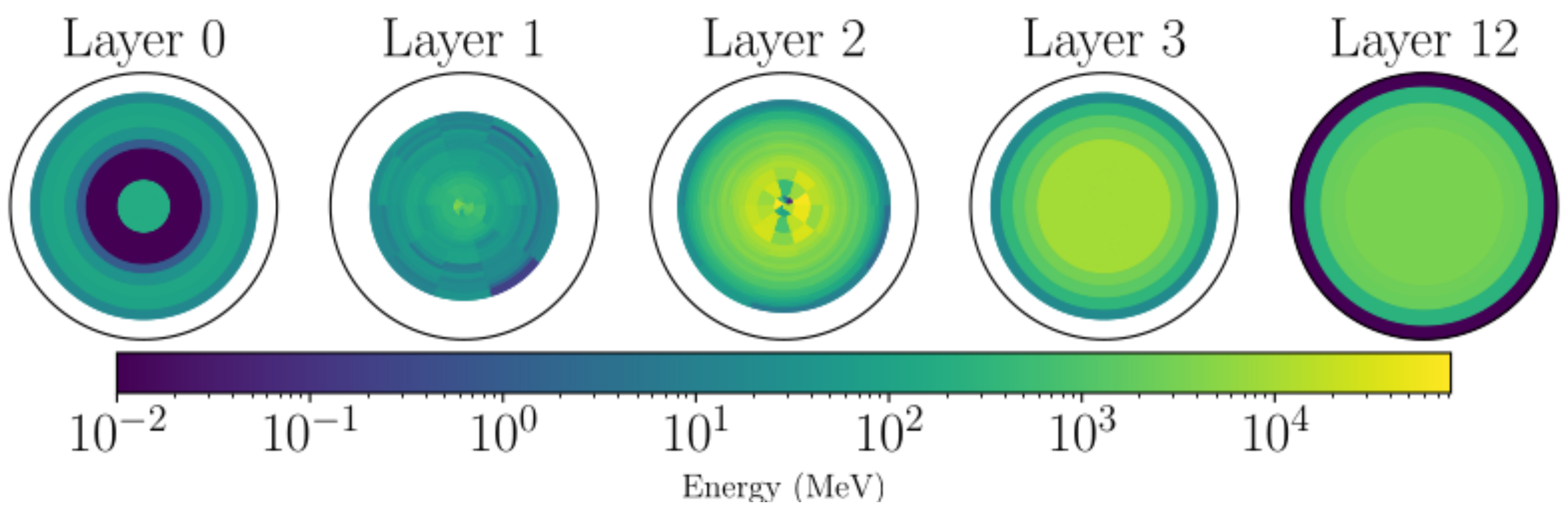
CaloChallenge

Datasets:

[Github](#)

- DS1: Atlas simulation of γ and π^+ showers at $\eta = 0.25$
  - photons have 388 voxels
  - pions have 533 voxels

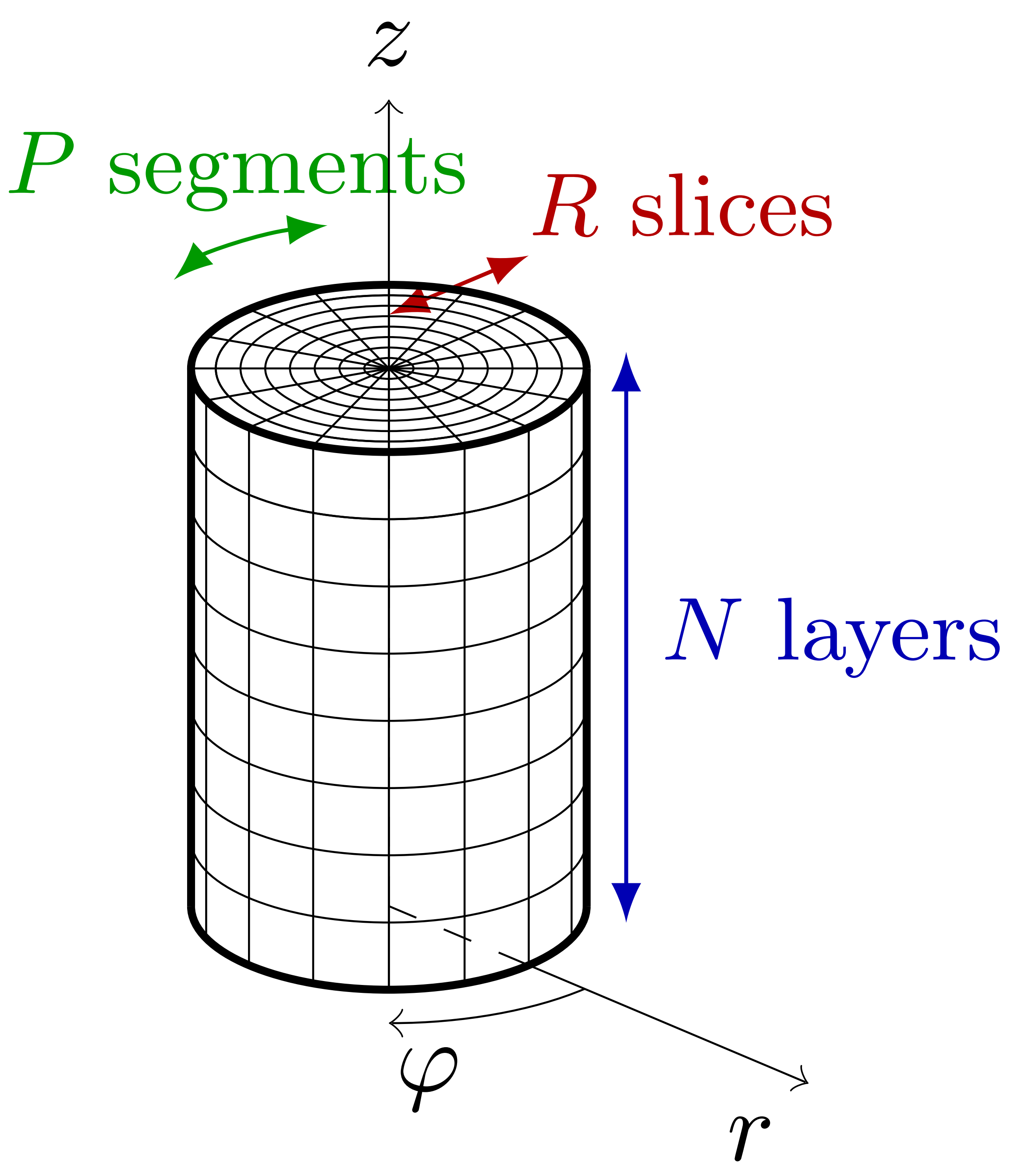
E_{inc}	256 MeV ... 131 GeV	262 GeV	0.524 TeV	1.04 TeV	2.1 TeV	4.2 TeV
photons	10000 per energy	10000	5000	3000	2000	1000
pions	10000 per energy	9800	5000	3000	2000	1000

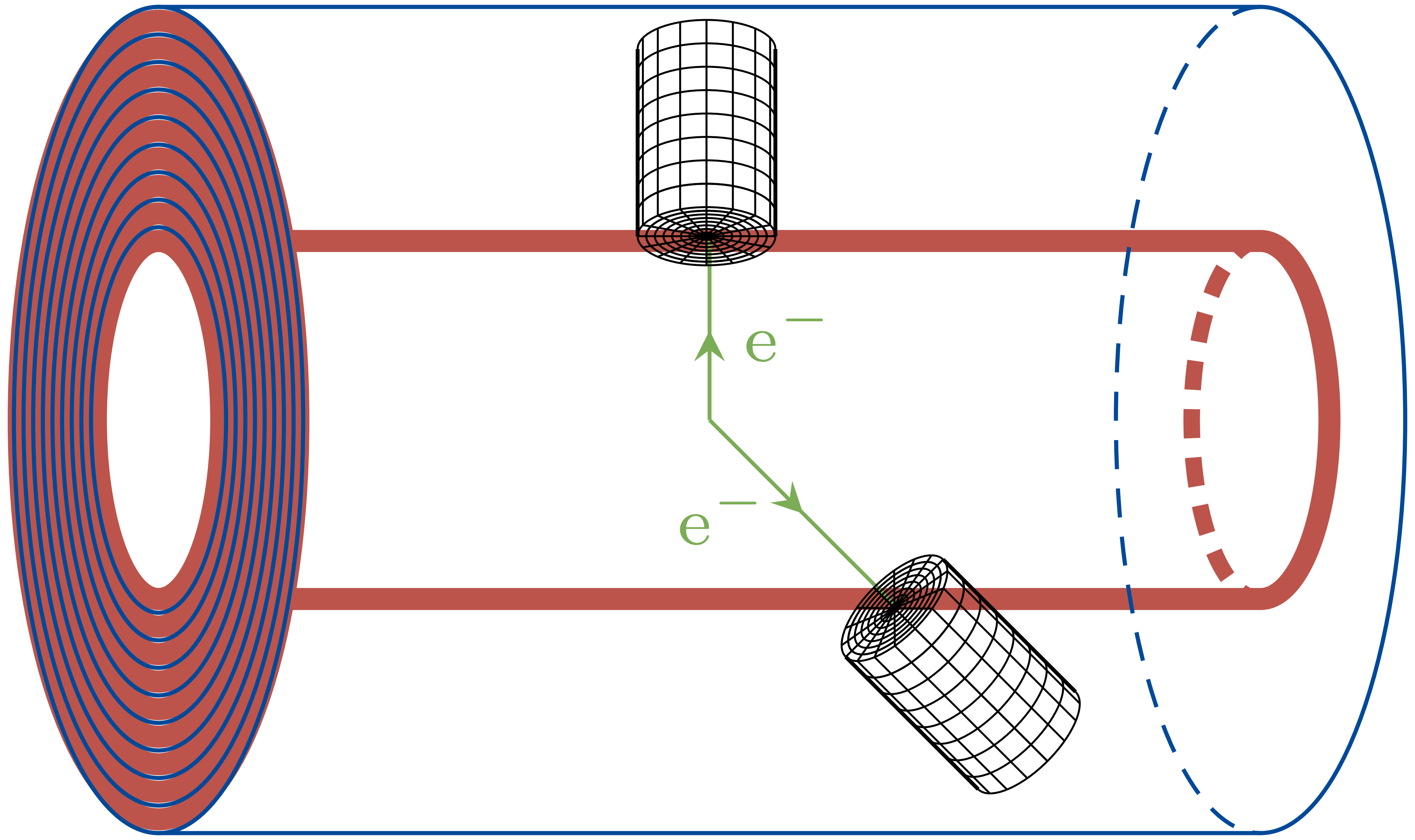


- DS2/3: Geant simulated e^+ with 45 layers of active silicon detector + tungsten absorber
  - DS2 has a total of 6480 voxels
  - DS3 has 45000 voxels
  - log-uniform energy, $E_{inc} \in [1, 10^3]$ GeV

CaloChallenge

[Github](#)





Preprocessing

Clever preprocessing:

normalized showers

$$u_0 = \frac{\sum_i E_i}{E_{inc}} \qquad \text{and} \qquad u_i = \frac{E_i}{\sum_{j\geq i} E_j} \, ,$$

log/logit

$$x_\alpha = (1 - 2\alpha)x + \alpha \in [\alpha, 1 - \alpha] \qquad \text{with} \quad \alpha = 10^{-6}$$

$$x' = \log \frac{x_\alpha}{1 - x_\alpha} \, .$$

We approached the problem in two ways:

Directly learn the full distribution $p(x_i, u_i \,|\, E_{inc})$

CaloINN
arXiv: 2312.09290

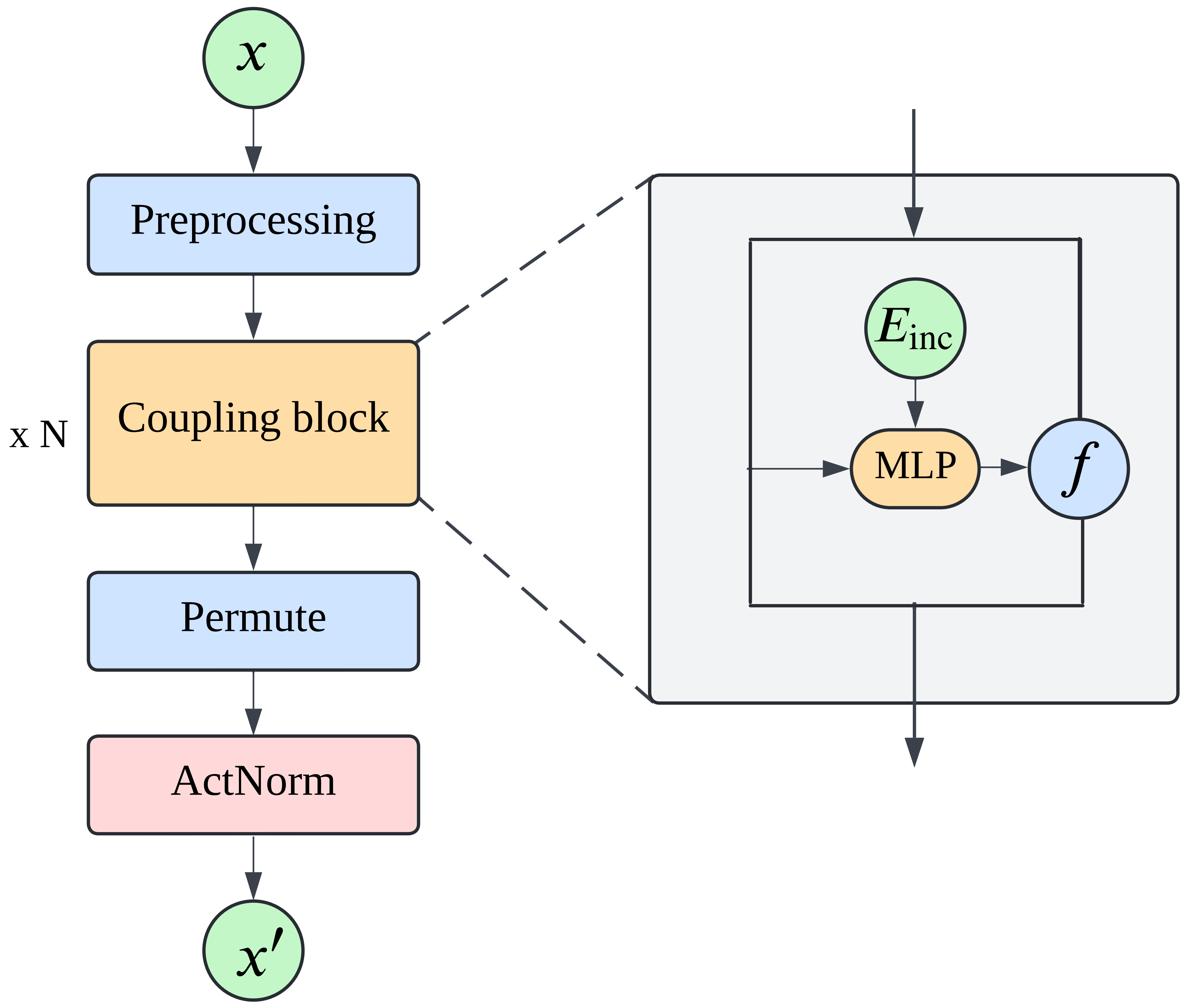
Factorise the problem into:

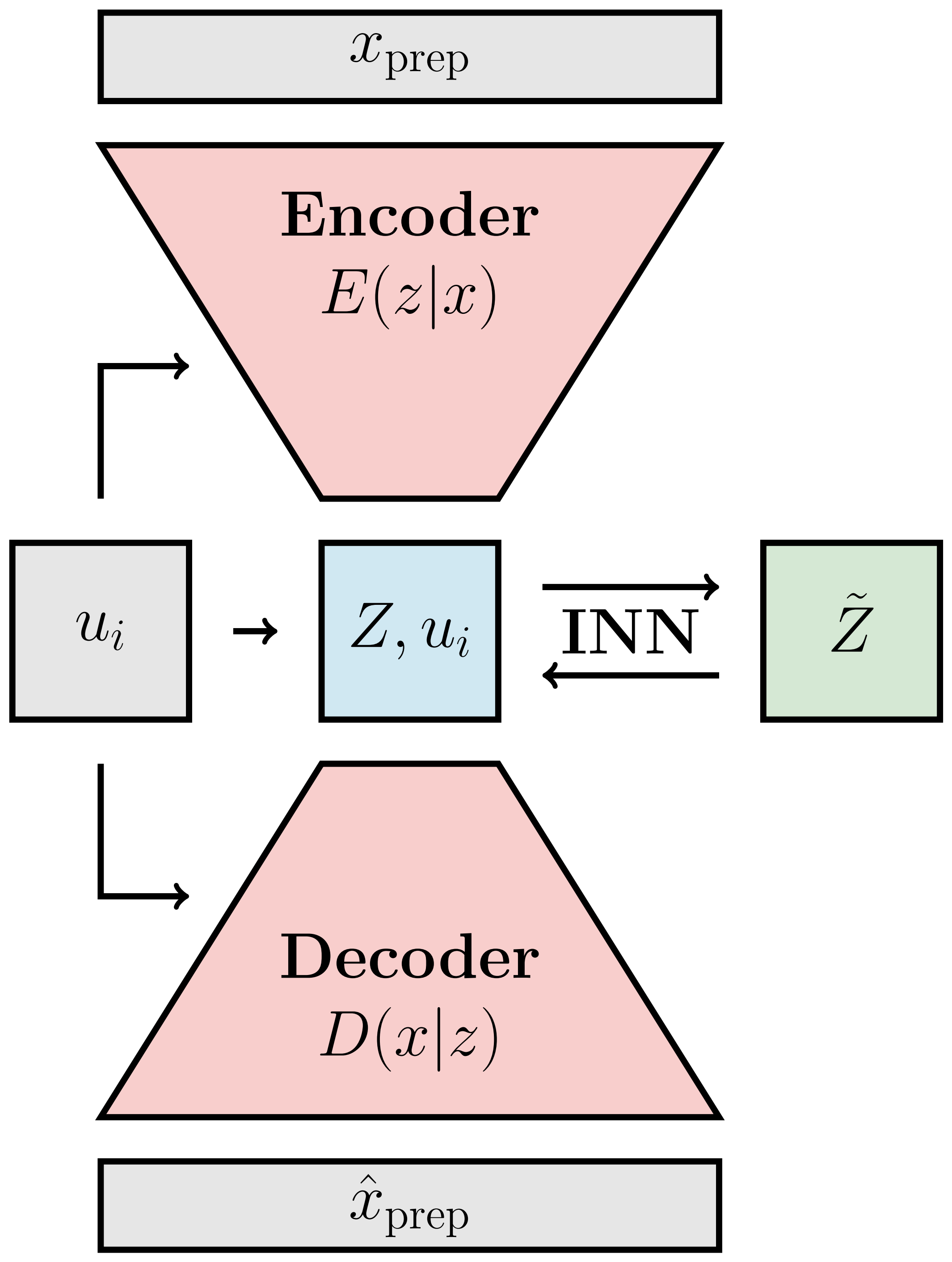
- learn the energy distribution, $p(E_i \,|\, E_{inc})$
- learn the normalised voxels $p(x_i \,|\, u_i, E_{inc})$

CaloDREAM
arXiv:2405.09629

From DS1 to DS2

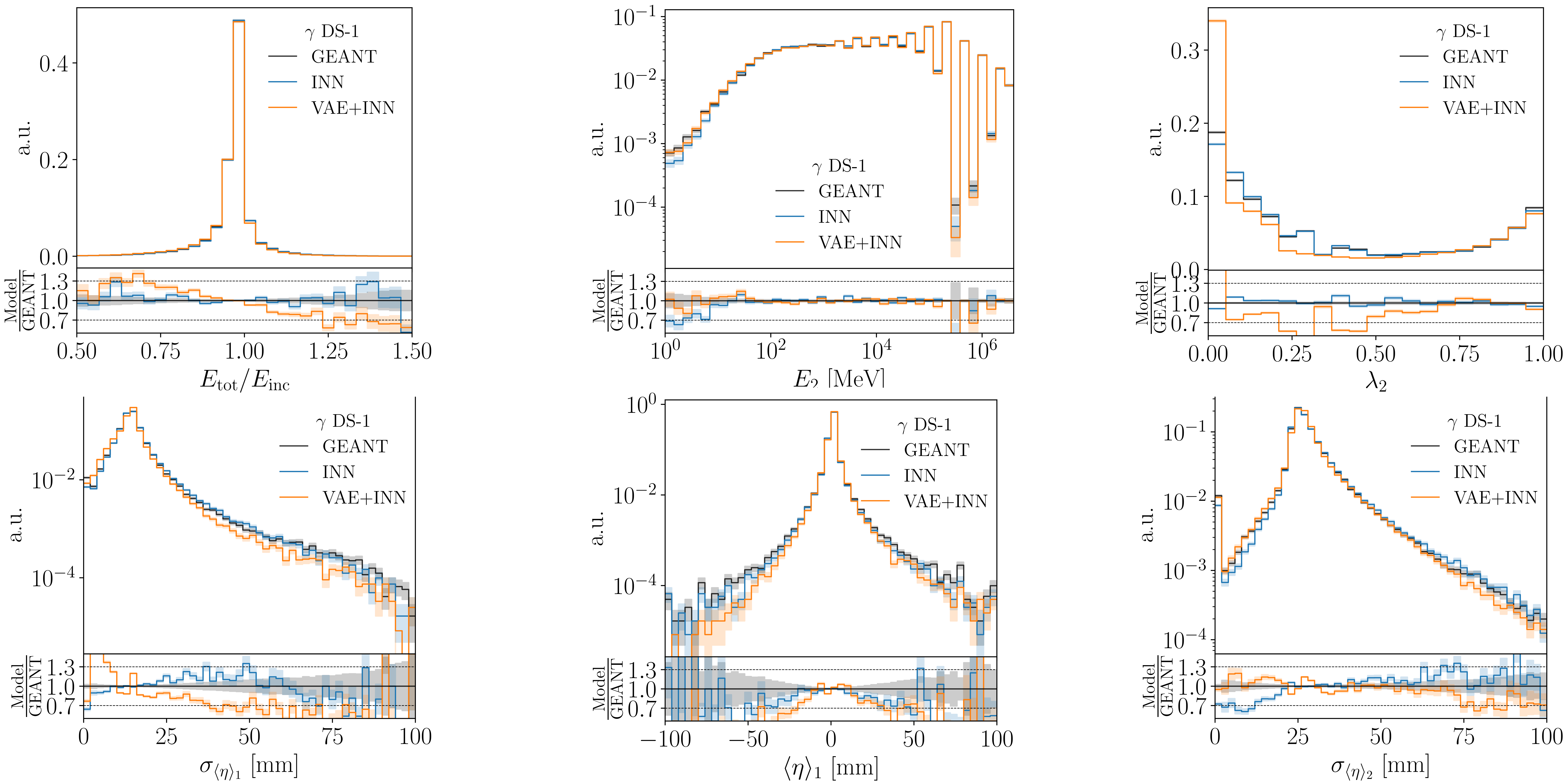
CaloINN



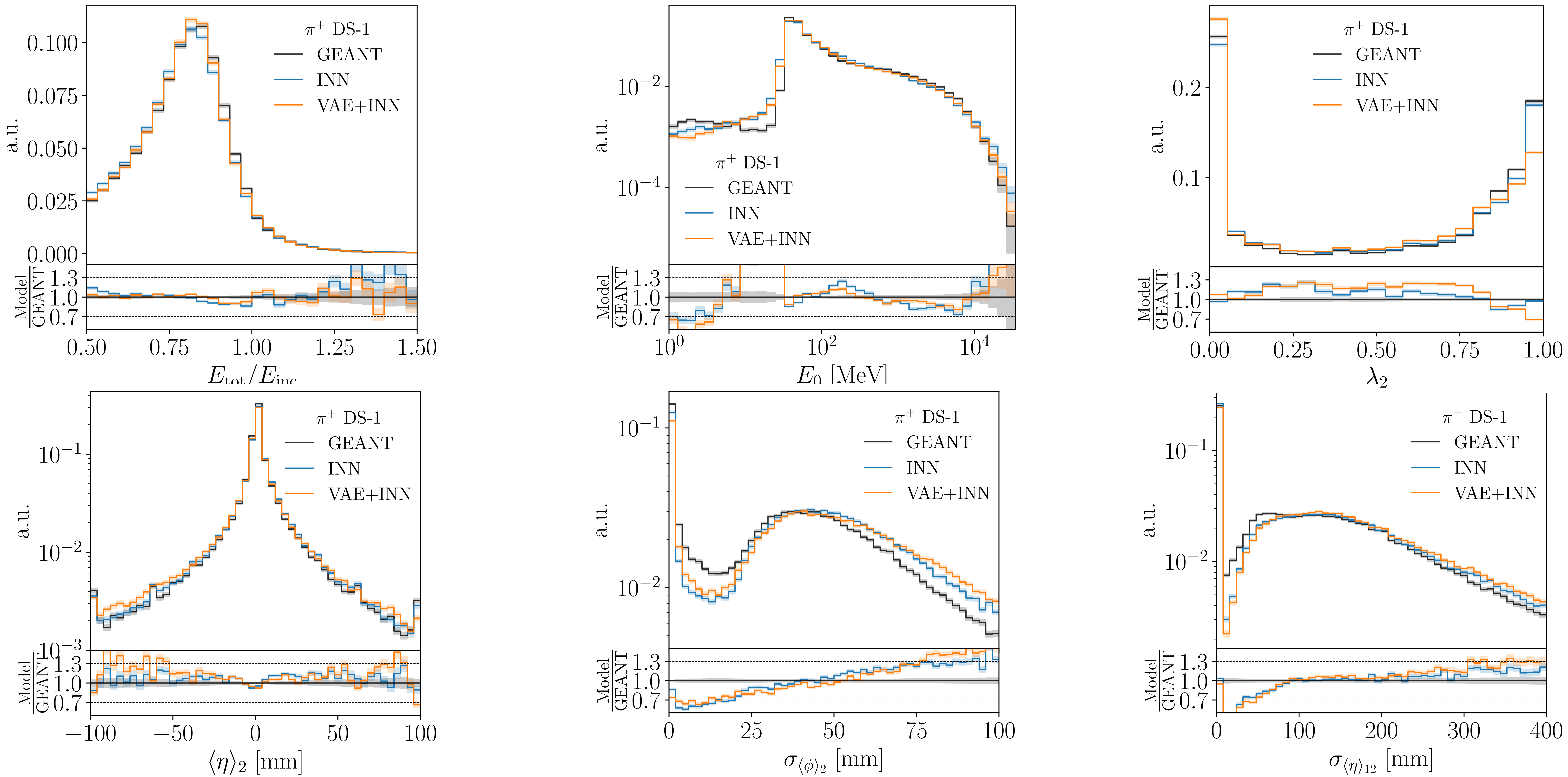


Check out the articles for more info on the architectures and open-source code!

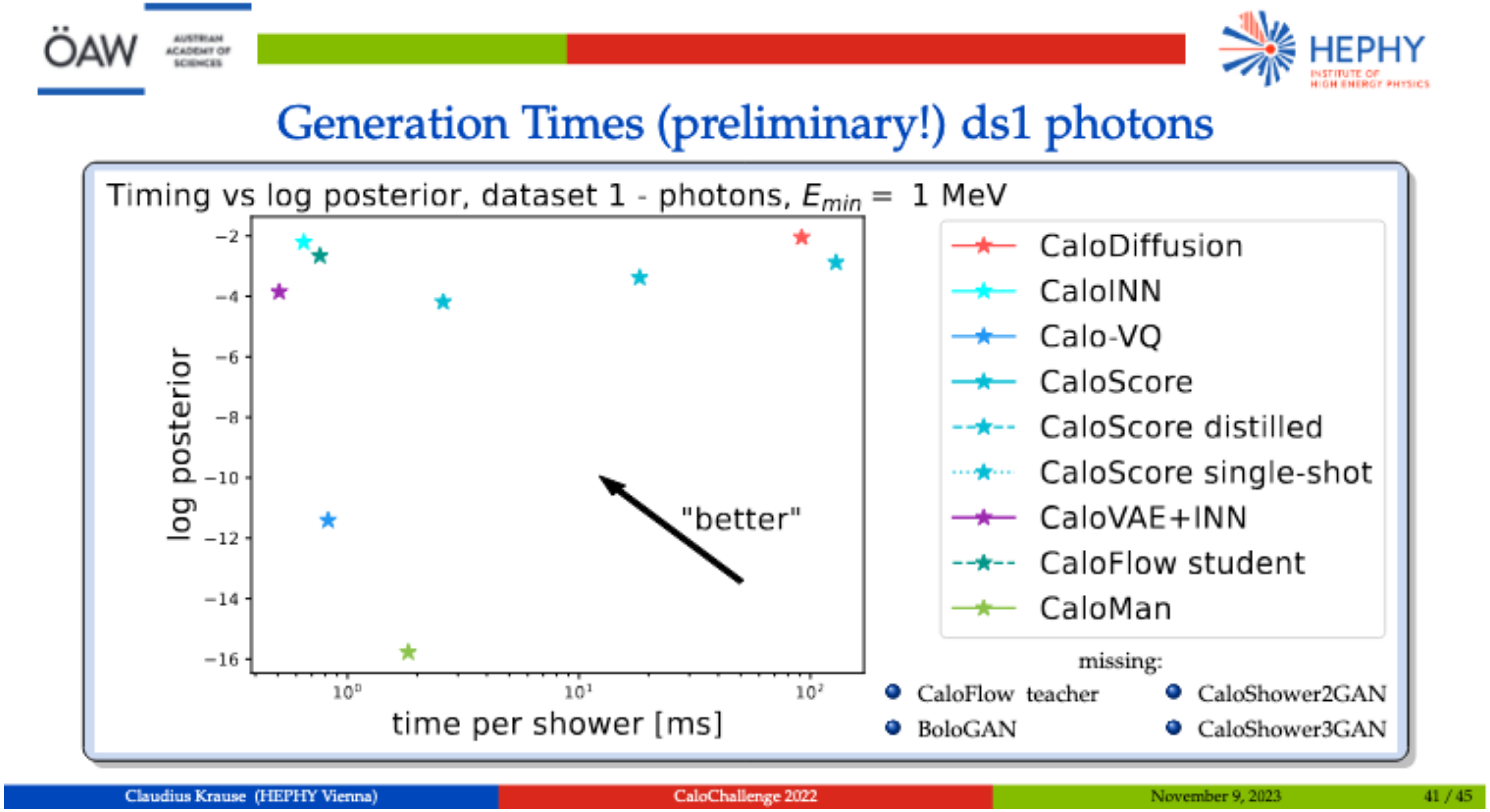
DS1- histograms



DS1- histograms

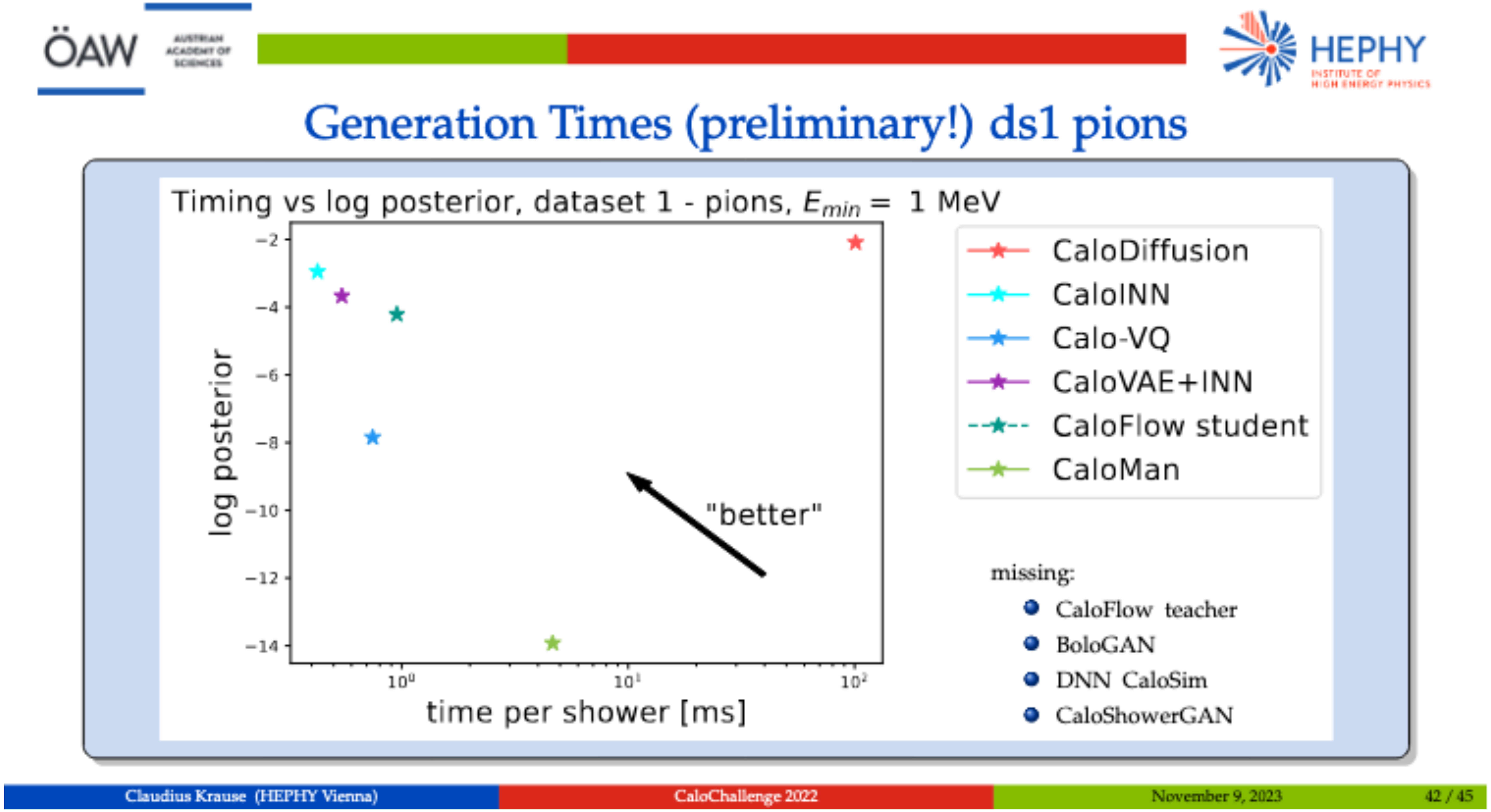


DS1 - timing



from Claudius' talk at ML4jets23

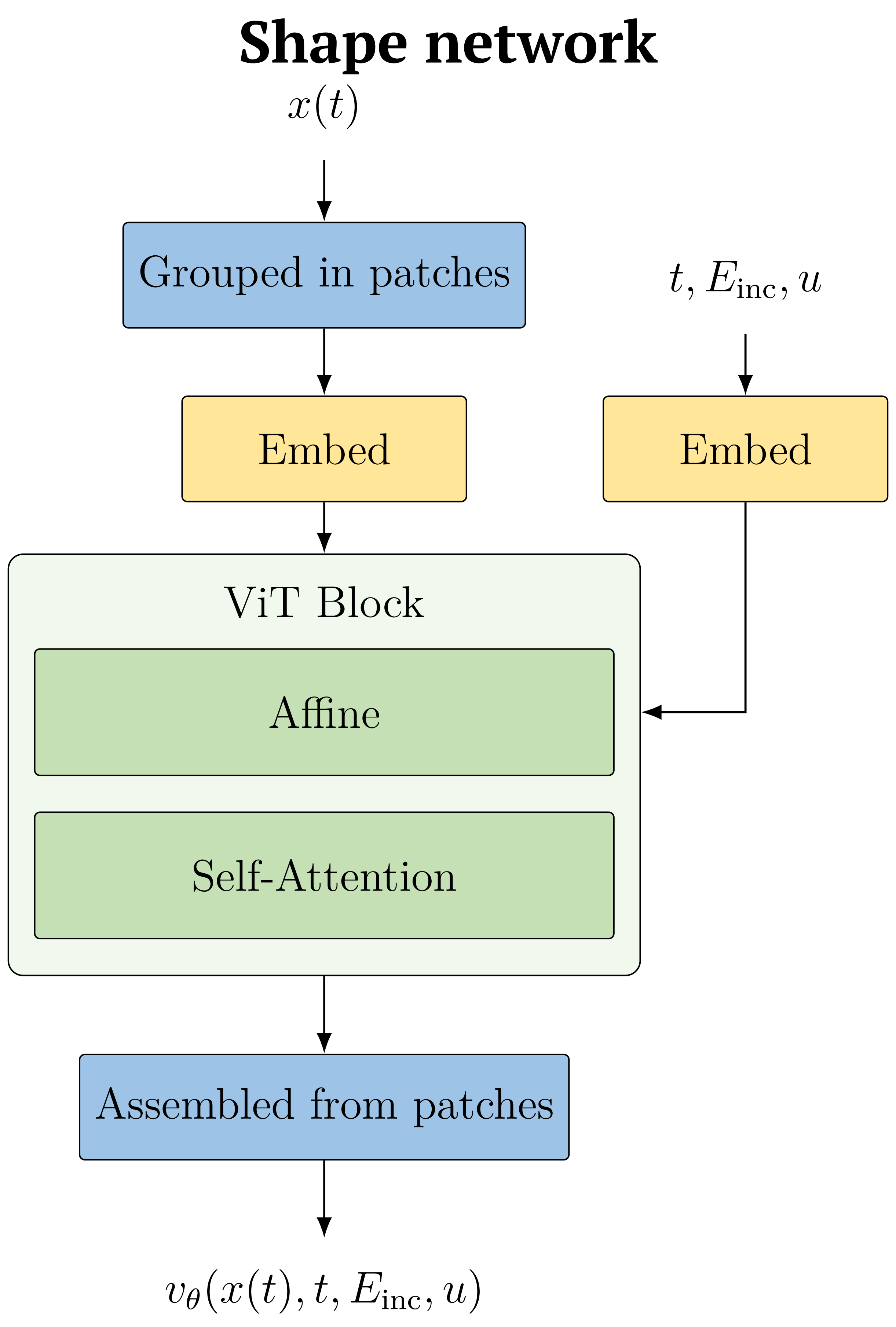
DS1 - timing

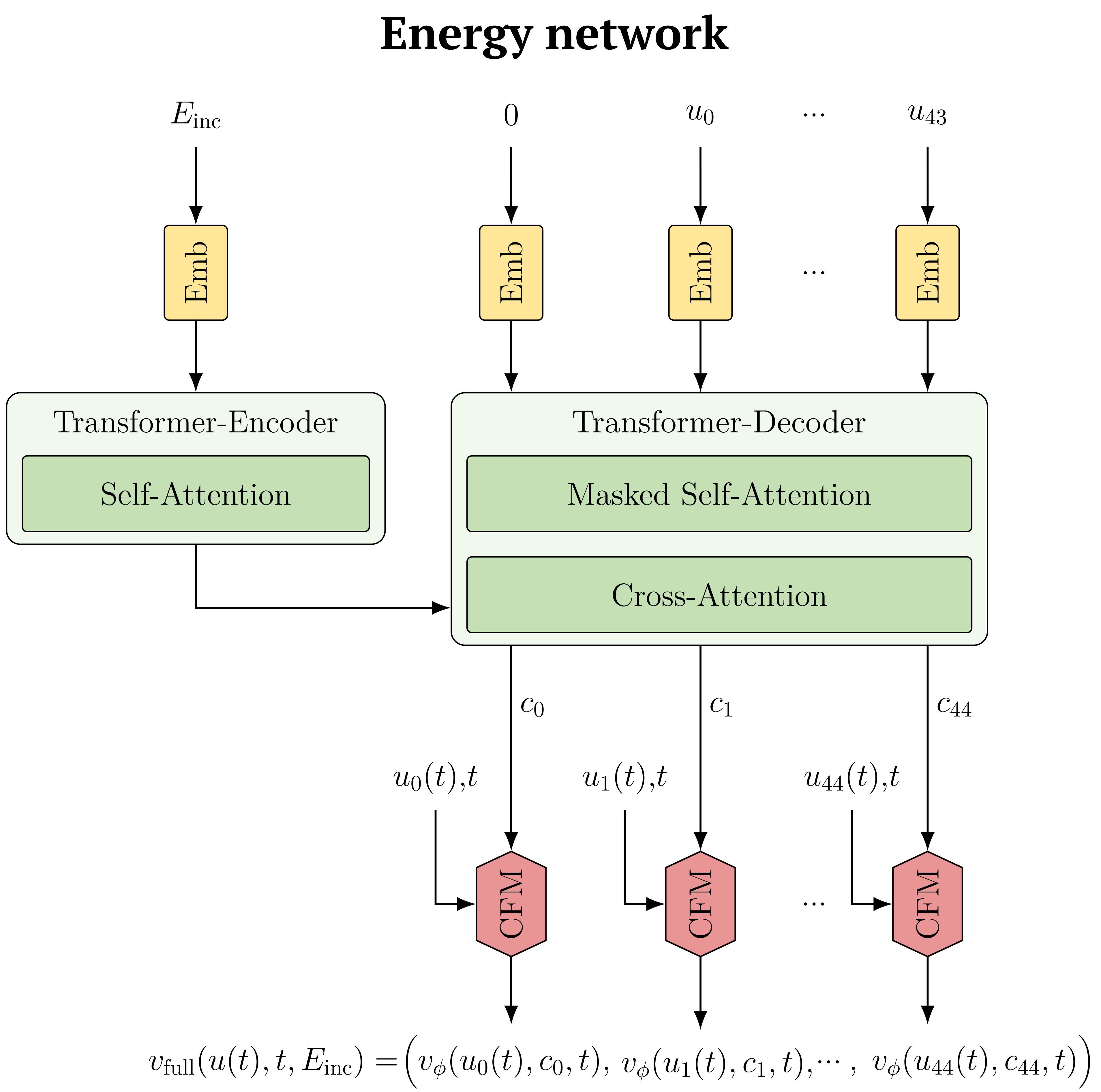


from Claudius' talk at ML4jets23

From DS1 to DS2/3

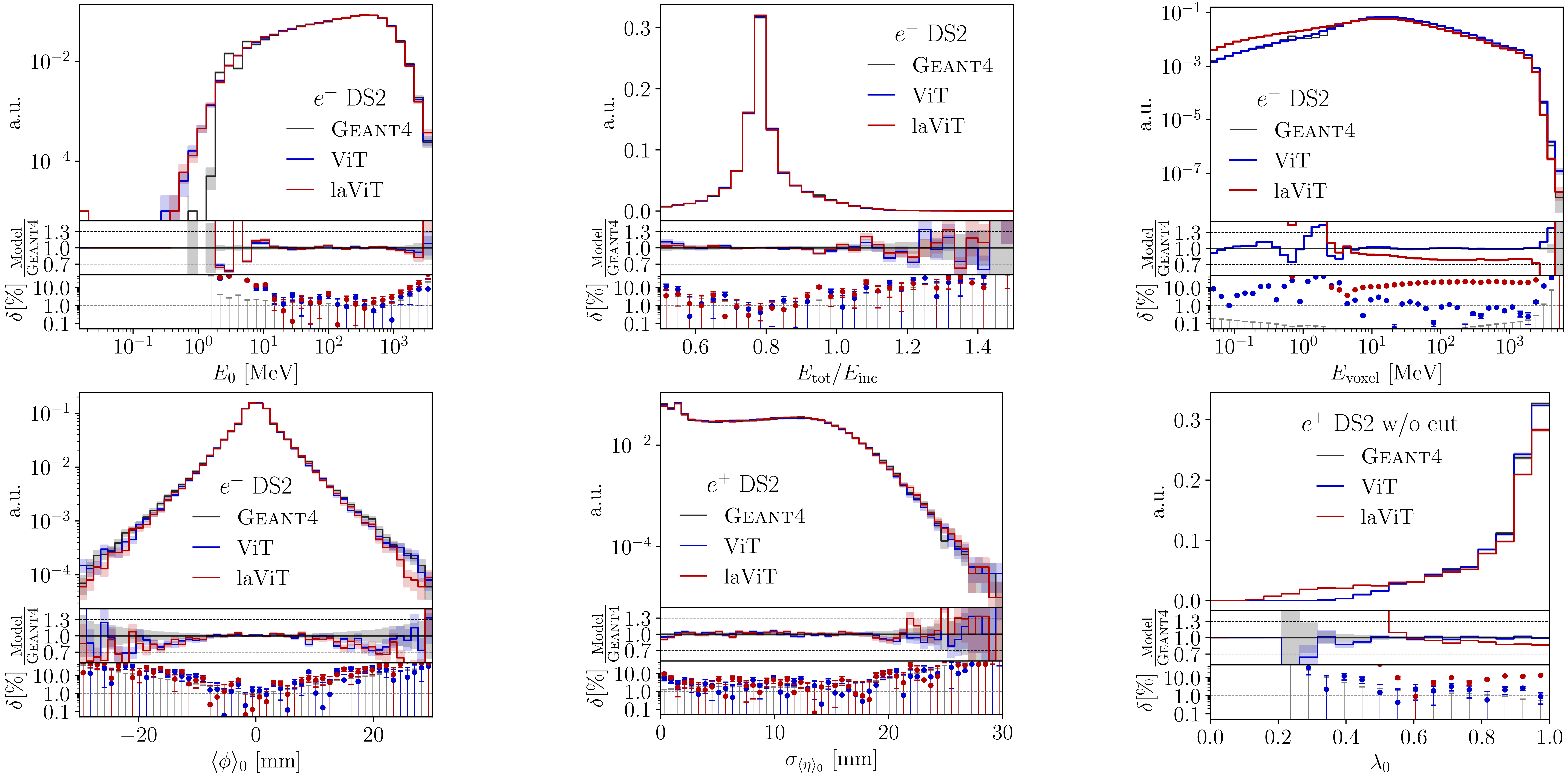
CaloDREAM



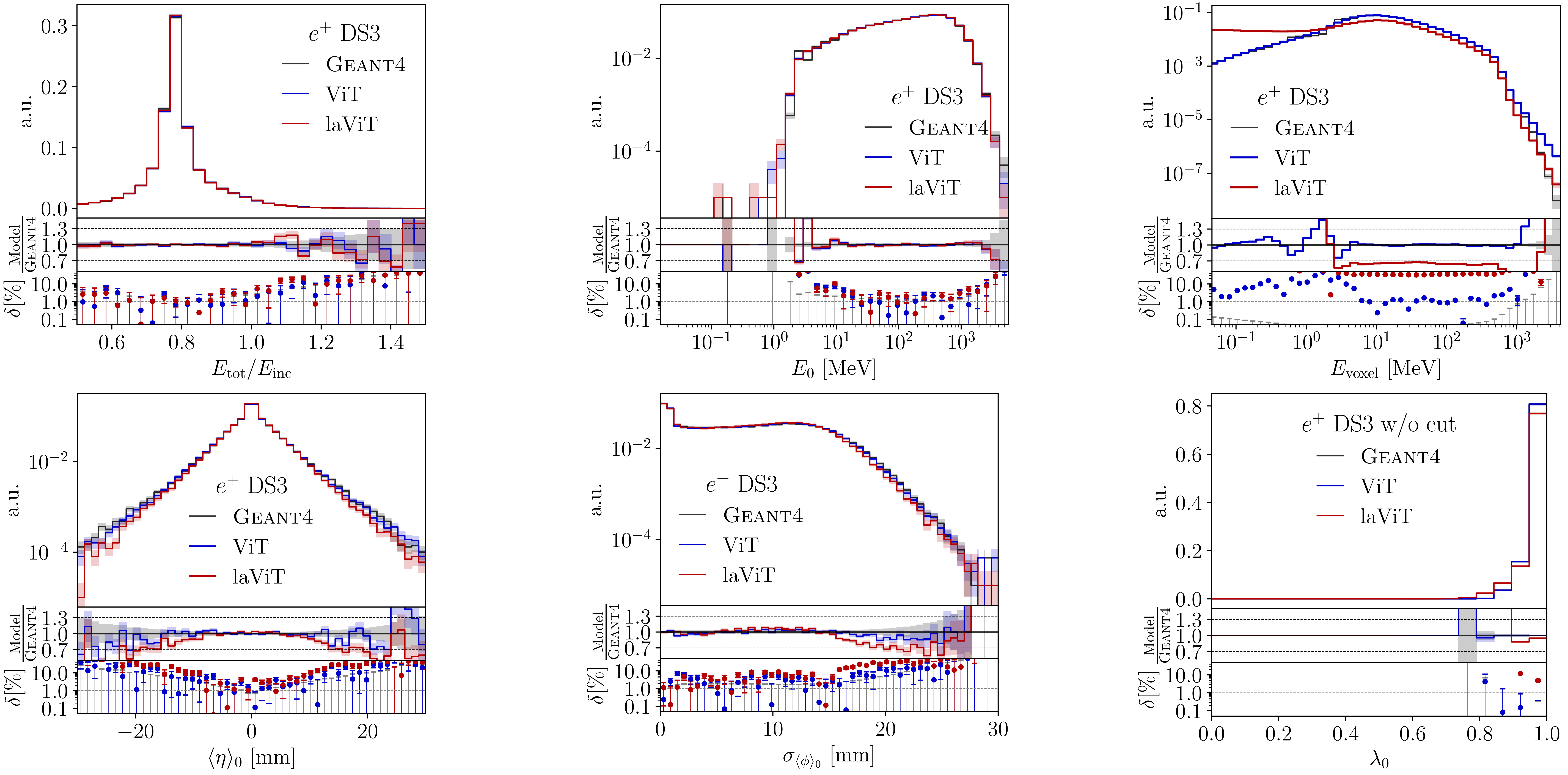


Check out the articles for more info on the architectures and open-source code!

DS2/3 - histograms



DS2/3 - histograms



The ultimate metric

from arXiv:2305.16774

- Classifiers are the best tools we have to test our generative networks;
- the output approximates the quantity:

$$C(x) = \frac{p_{data}}{p_{data} + p_\theta} \qquad\qquad \frac{p_{data}}{p_\theta} = \frac{C(x)}{1 - C(x)}$$

- Optimal observable for a two hypothesis test according to the Neyman-Pearson lemma
- Proper training is essential: architecture, over-fitting, calibration,...
- we can easily extract weights from properly trained classifiers $\longrightarrow w(x) \approx \frac{p_{data}}{p_\theta}(x)$

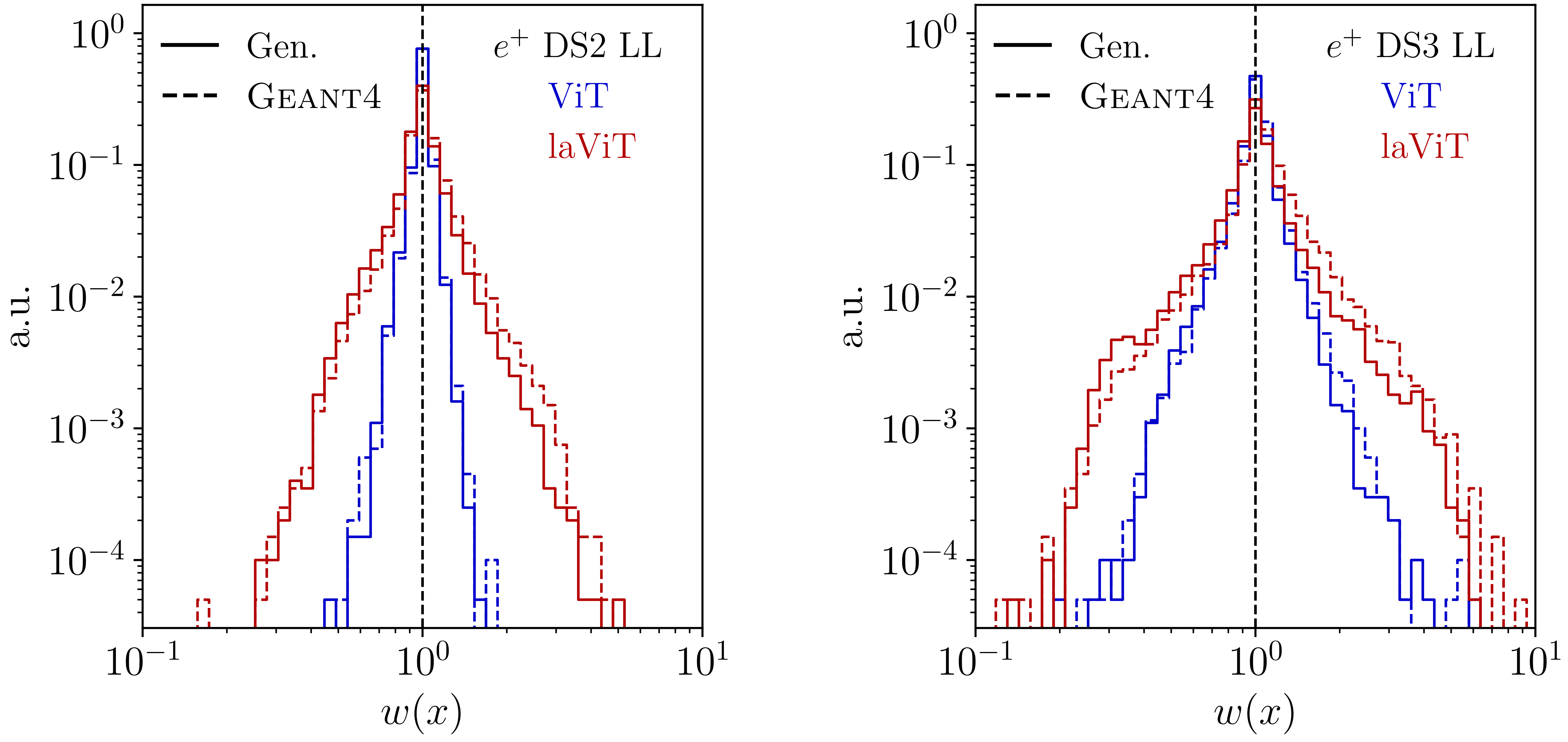
The ultimate metric

- Evaluation done in terms of the area-under-the-ROC (AUC) curve

  ⟶ indistinguishable samples if AUC=0.5

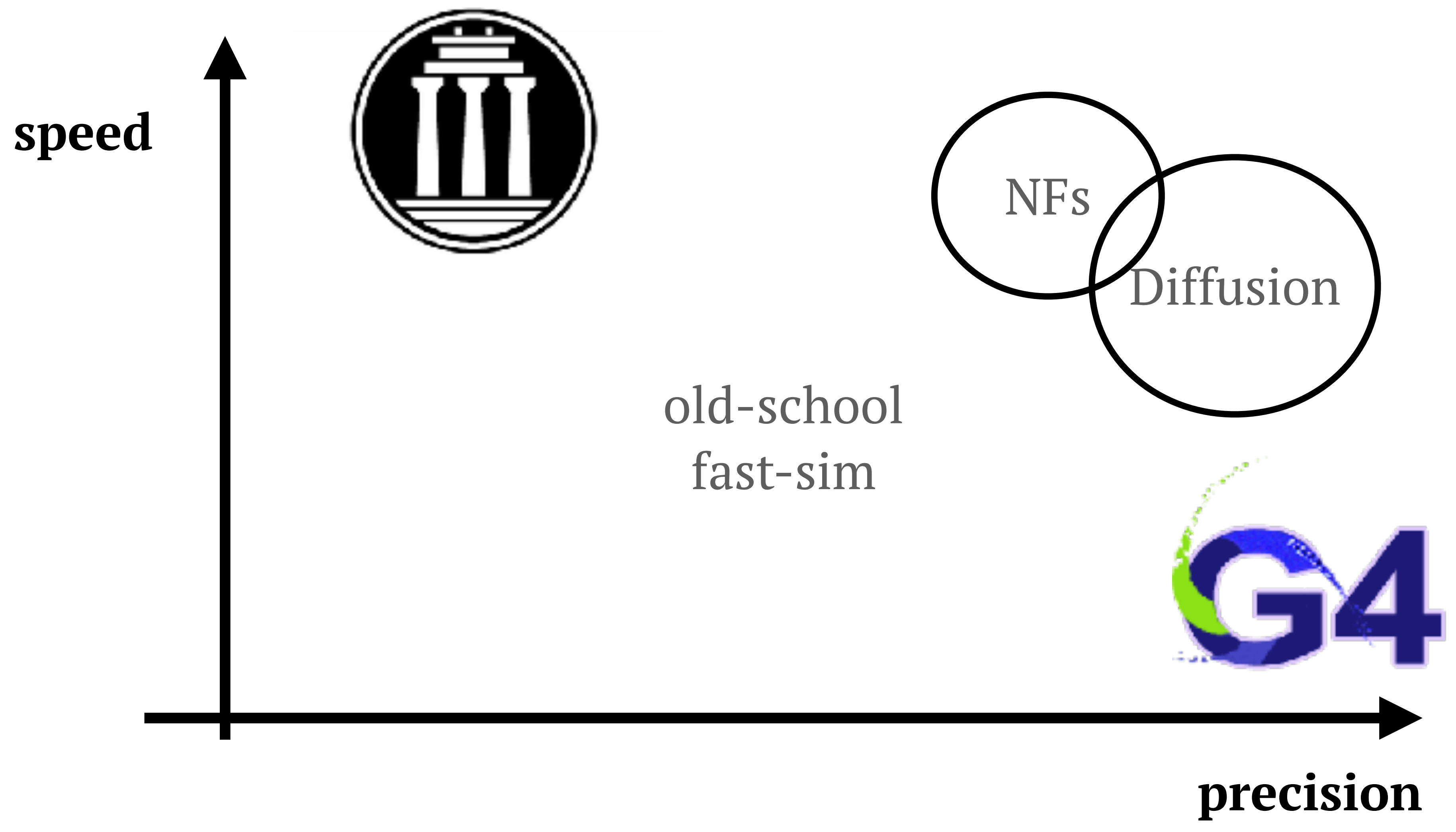
- Better to look at the weight distribution

	AUC (LL/HL)	
	DS2	DS3
ViT	0.54/0.52	0.63/0.53
laViT	0.58/0.53	0.62/0.59



Conclusions

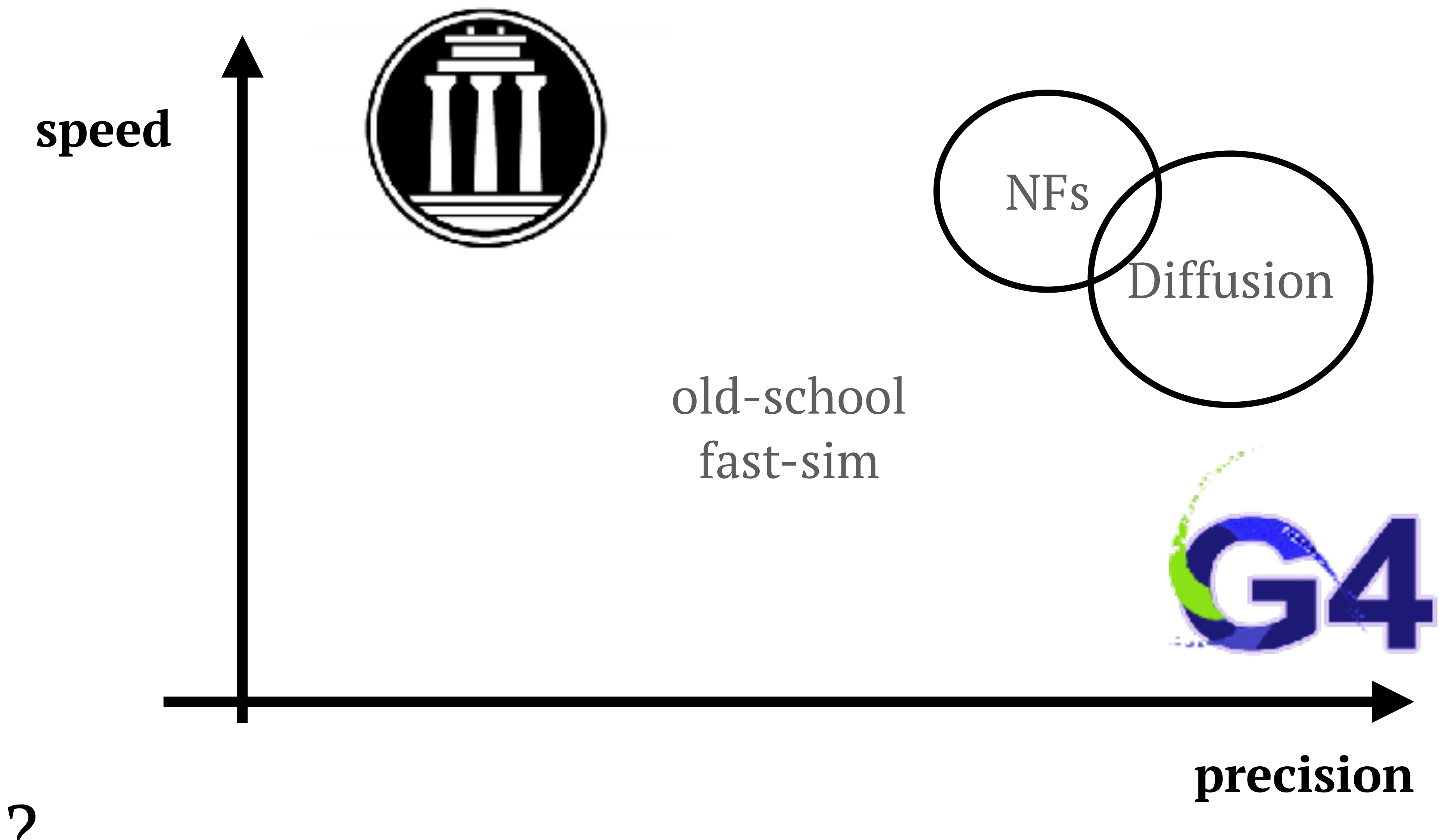
- Very quick introduction on calorimeter simulations;
- even quicker introduction on the ML tools;
- Normalizing flows:
  - fast!;
  - CaloINN has even good performance on DS1;
  - non-trivial to expand to larger calorimeters.
- Conditional Flow Matching:
  - sampling requires more function evaluation;
  - a more involved architecture;
  - awesome performance.



Conclusions

My open questions:

- is there a more efficient way of learning showers?
  - in high-dimensions sparsity is a problem;
  - point-cloud representation might be the answer.
- can we improve speed of diffusion networks?
- foundation models for detector simulation?



Looking at the future:

- how to implement such a network in the simulation chain?

Sidenote: only part of my PhD. I would also love to discuss about learning symmetries, anomaly detection, and unfolding.

Thank you for your attention!