Unleashing the power of generative models: Anomalies, Simulations, and other Surrogates

Gregor Kasieczka
Email: gregor.kasieczka@uni-hamburg.de
Twitter/X: @GregorKasieczka
CERN IML Workshop — 1.2.2024

CLUSTER OF EXCELLENCE
QUANTUM UNIVERSE

KISS

FSP CMS

PUNCH 4NFDI

DASHH

GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung

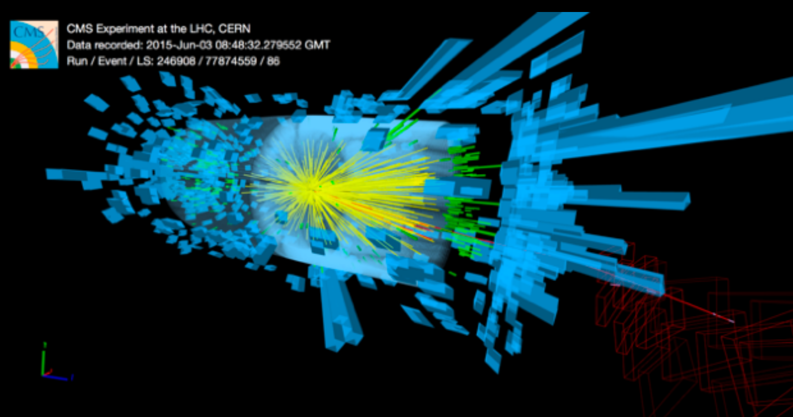
Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

CDCS
CENTER FOR DATA AND COMPUTING IN NATURAL SCIENCES

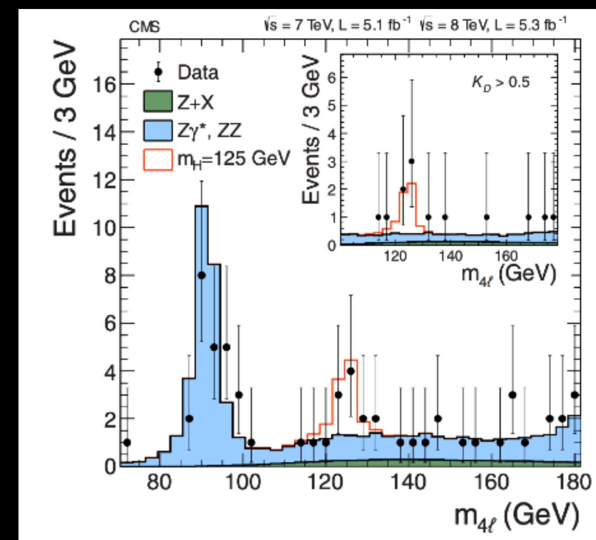
PIER
Partnership of Universität Hamburg and DESY

Emmy Noether-Programm
Deutsche Forschungsgemeinschaft
DFG

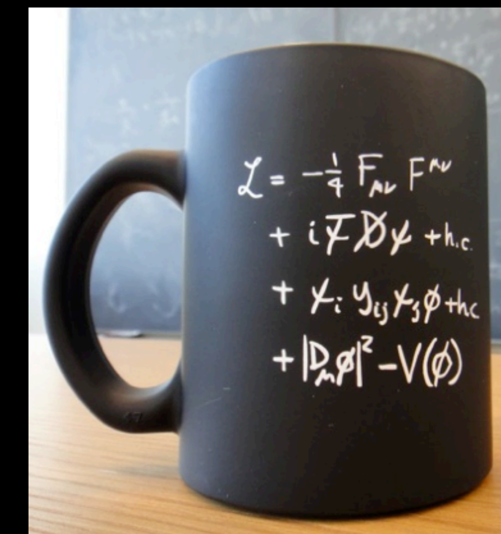
Data at CERN



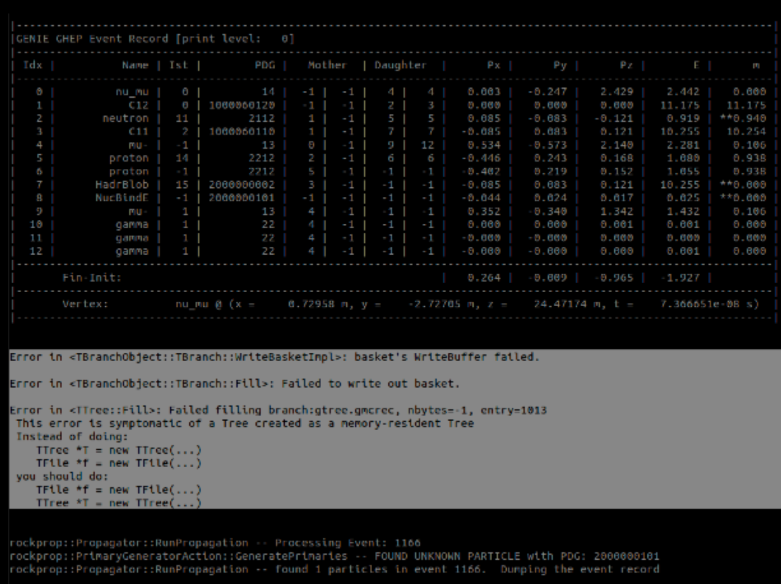
What the media sees



What experimentalists see

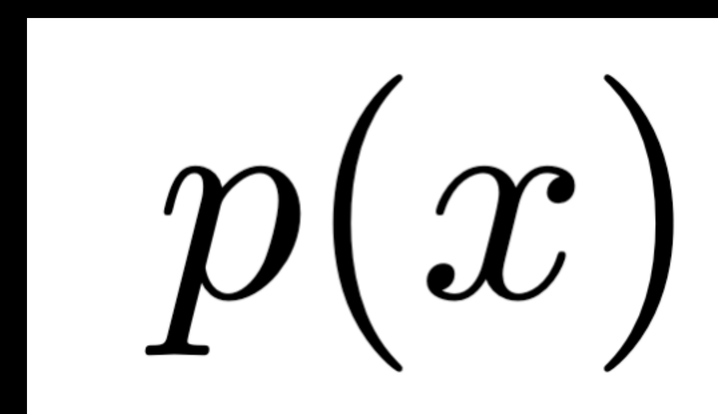


What theorists see



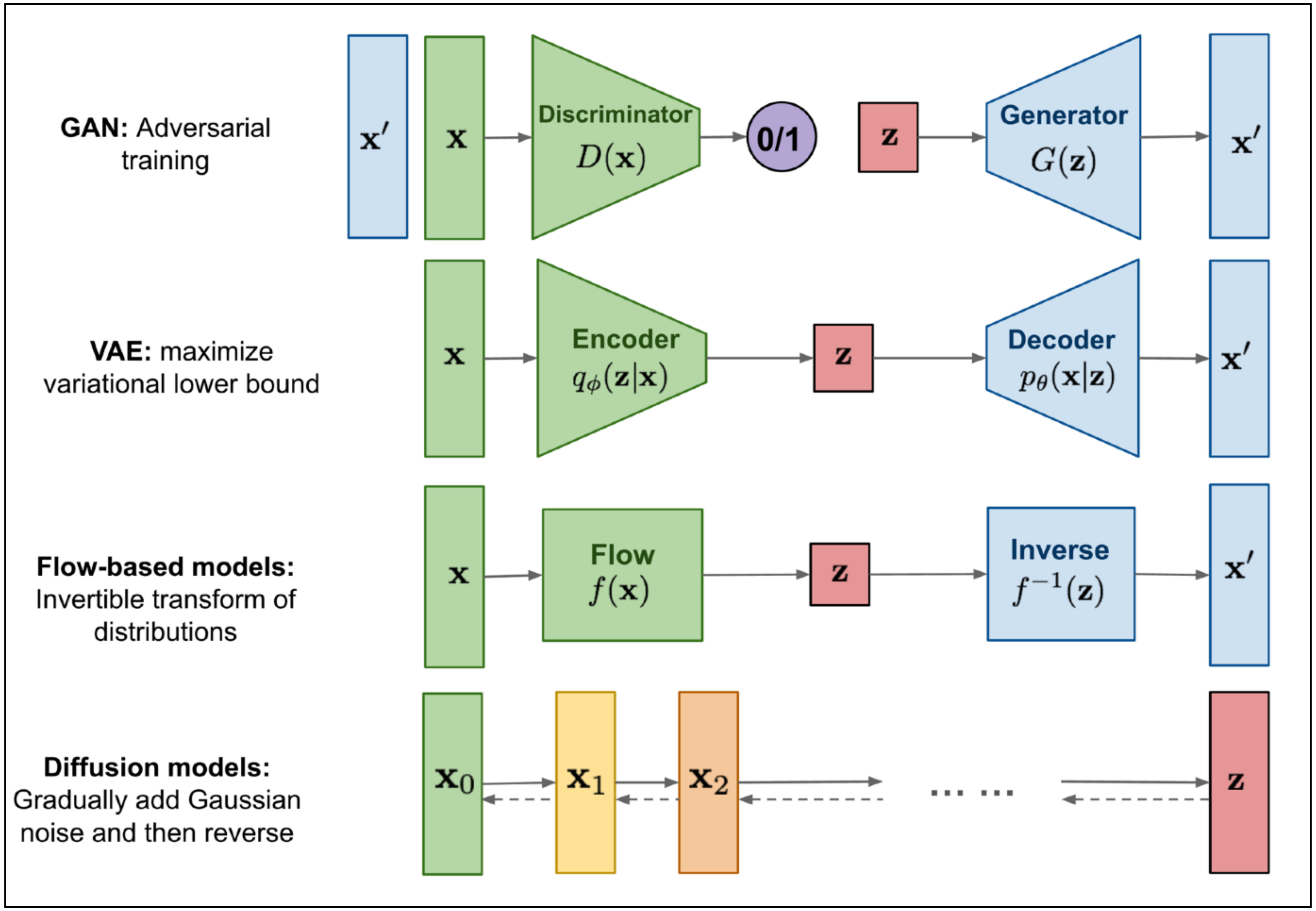
What grad students see

What tourists see



What generative models see

Generative Models



Either implicitly or explicitly learn (an approximation) to

$$p(x)$$

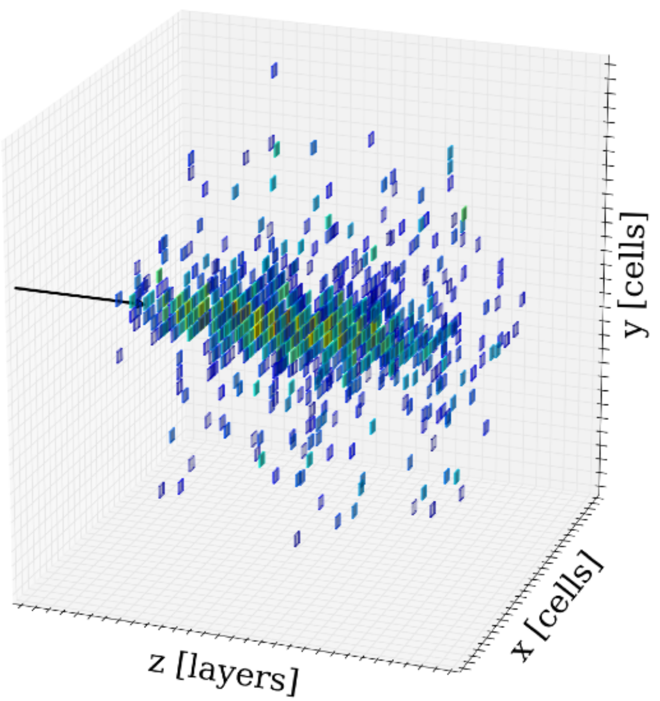
(the probability density of simulation or data)

Why generative models?

$$p(x)$$

Sample $X_i \sim p(x)$
to generate datapoints

Why generative models?



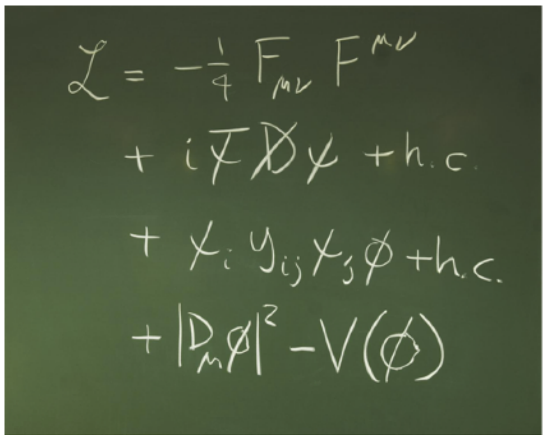
Showers in complex high-resolution calorimeters

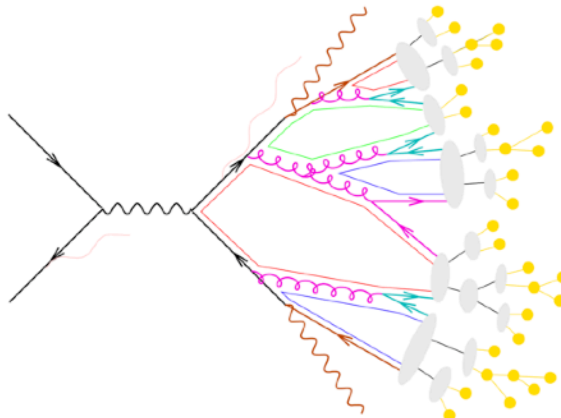
$$p(x)$$

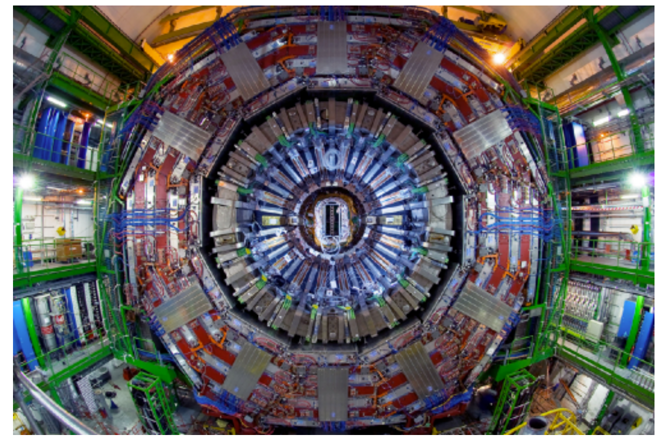
Sample $X_i \sim p(x)$
to generate datapoints

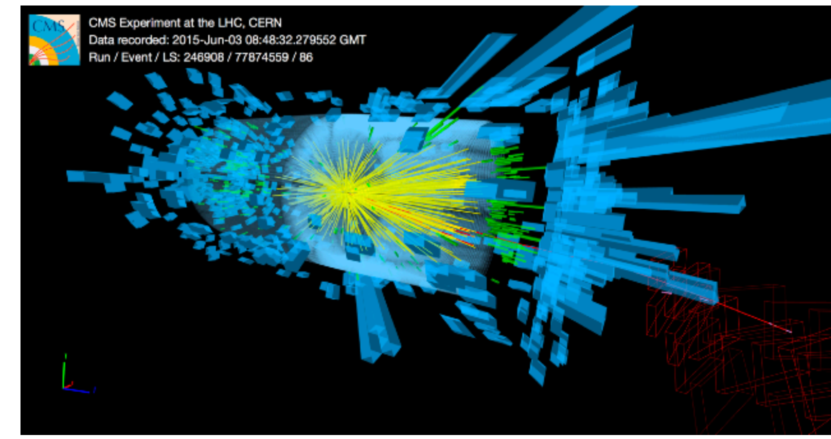
Motivation

This happens in the experiment







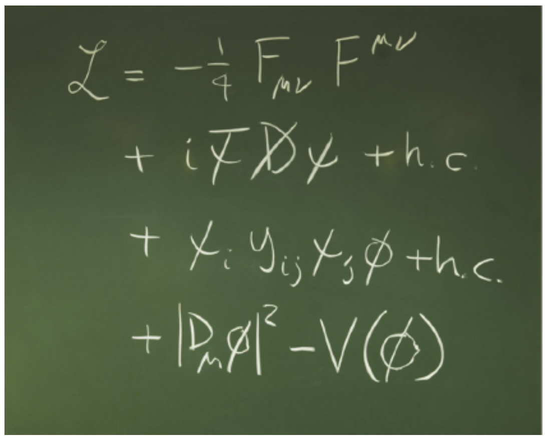


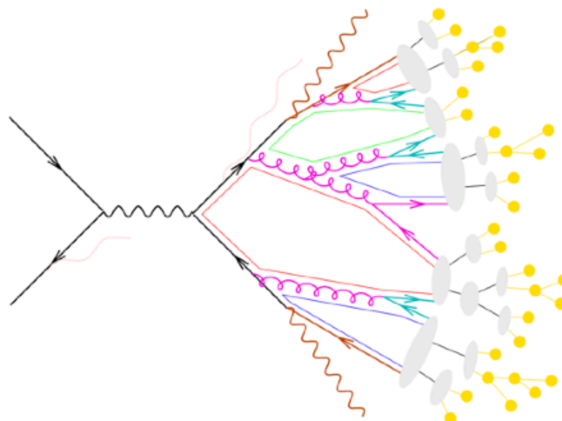
This is what we want to know

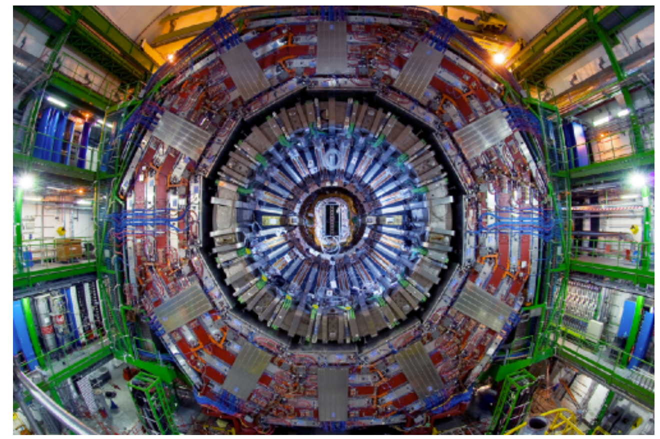
Simulation is crucial to connect experimental data with theory predictions

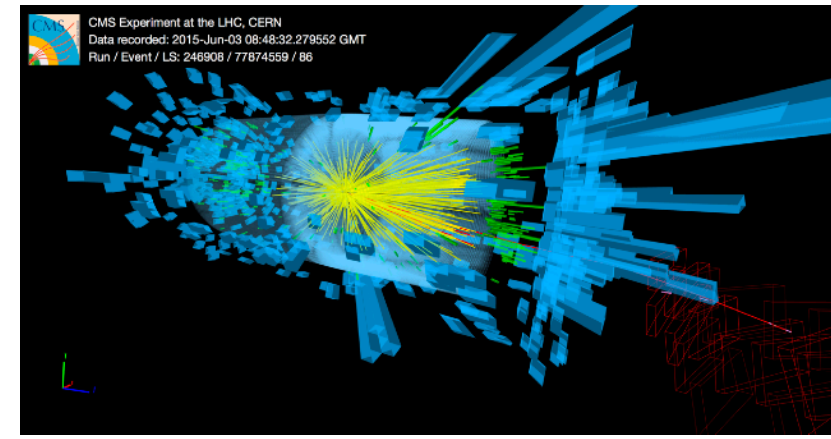
Motivation

This happens in the experiment



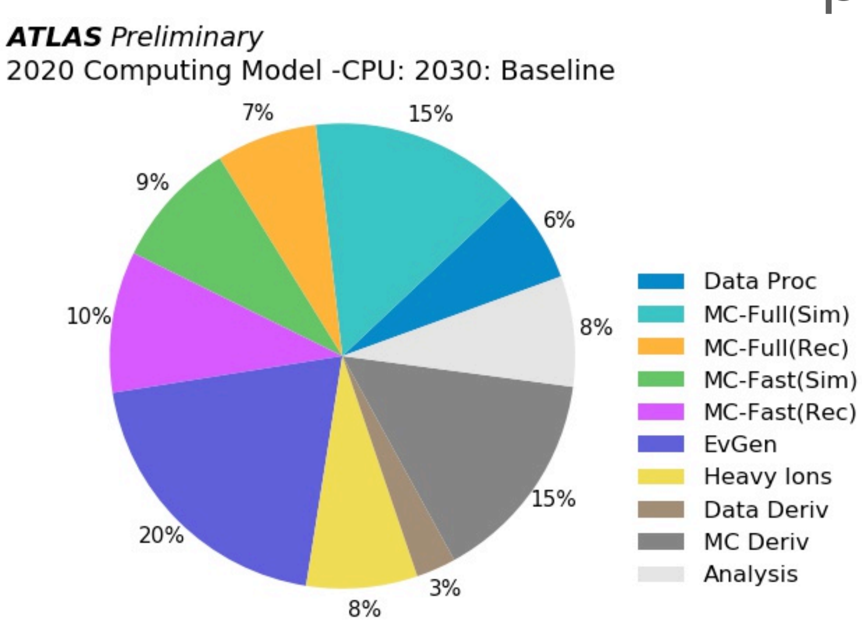


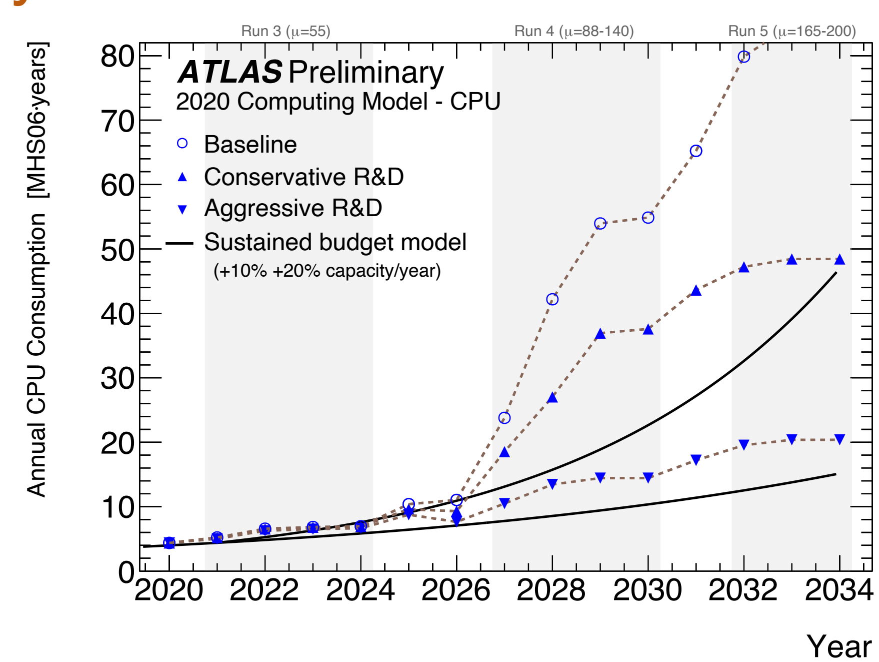




This is what we want to know

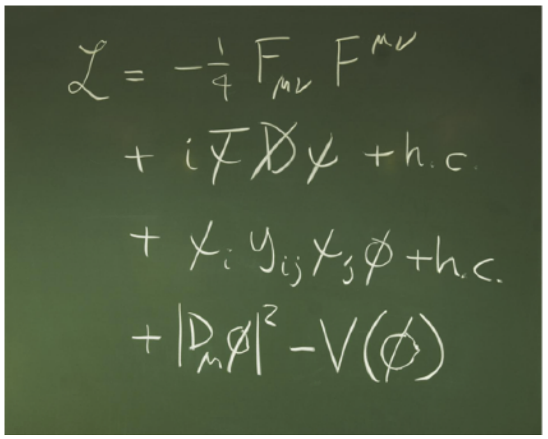
Simulation is crucial to connect experimental data with theory predictions, but computationally very costly

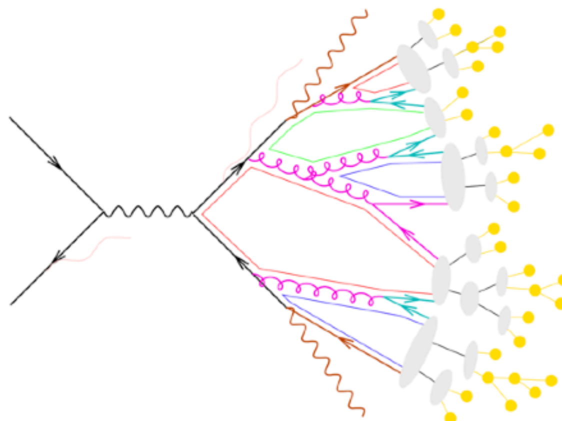


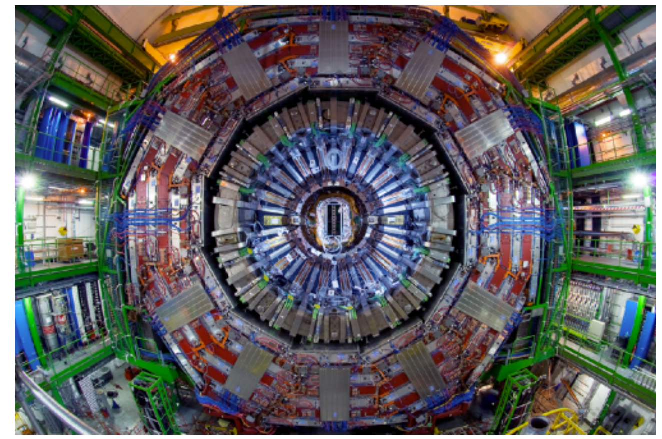


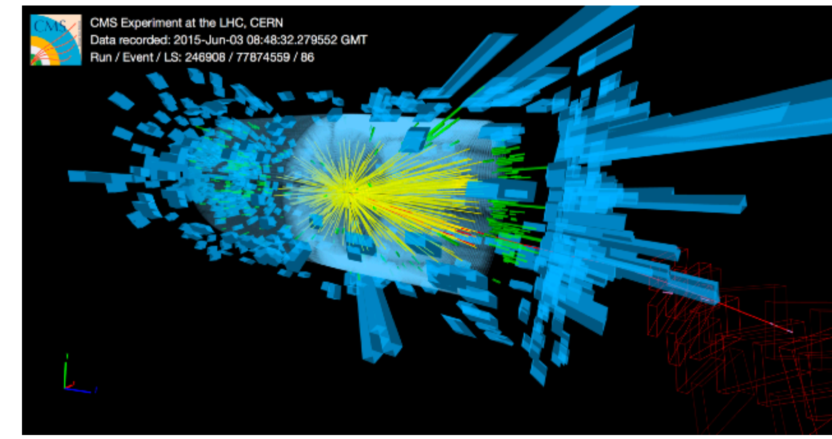
Motivation

This happens in the experiment







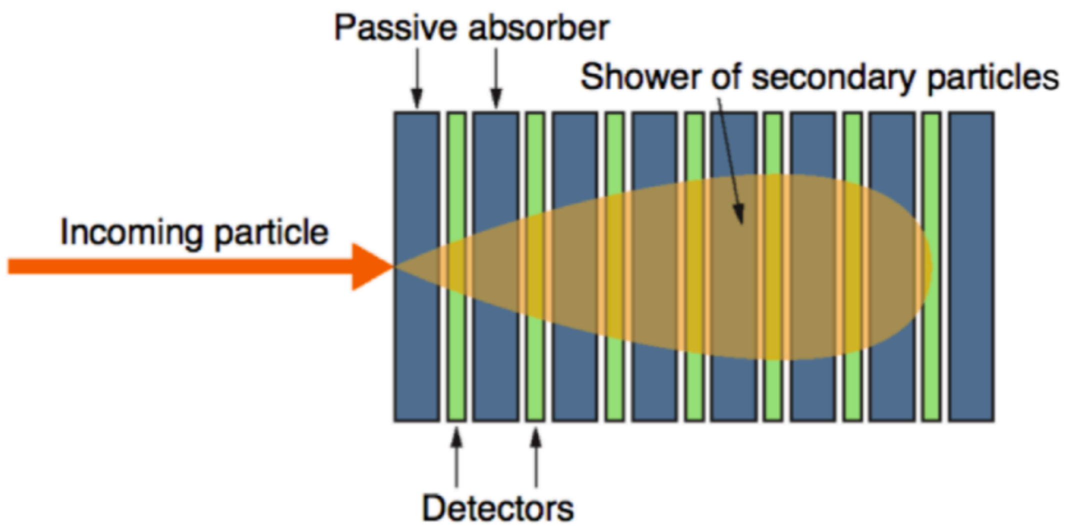


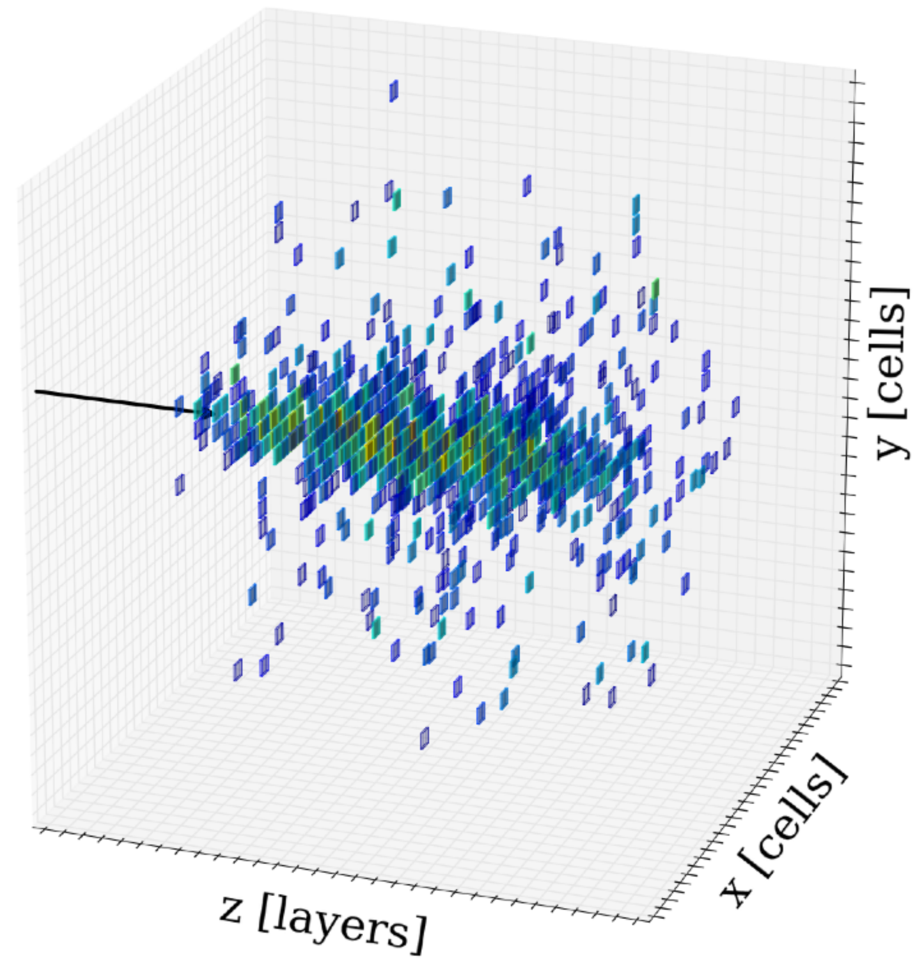
This is what we want to know

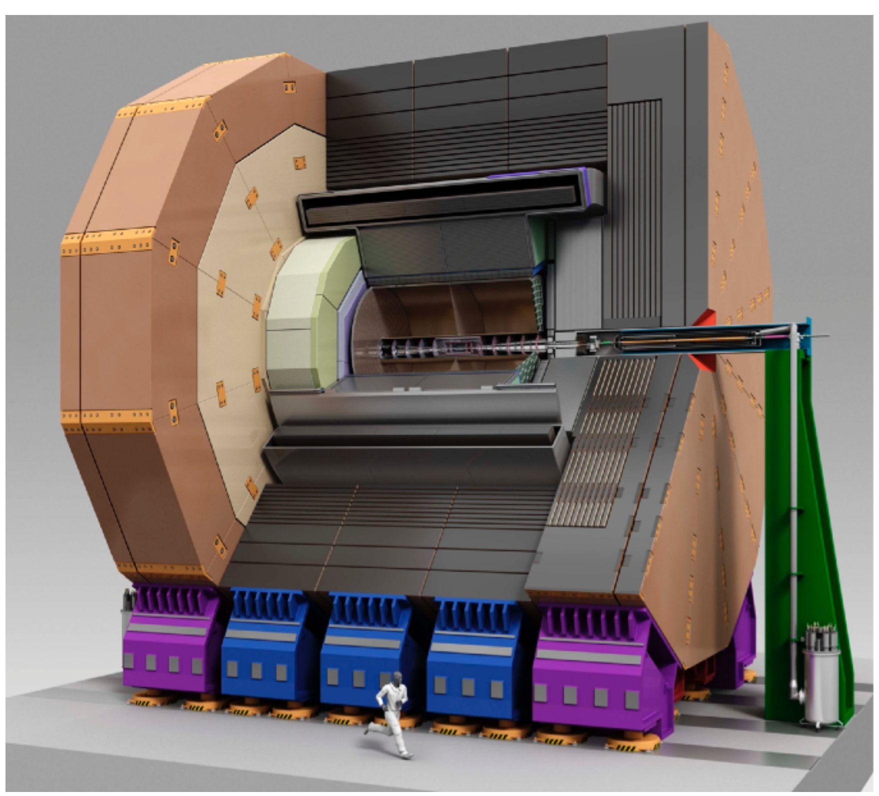
Simulation is crucial to connect experimental data with theory predictions, but computationally very costly

→Use generative models trained on simulation or data to augment simulations

Simulation target



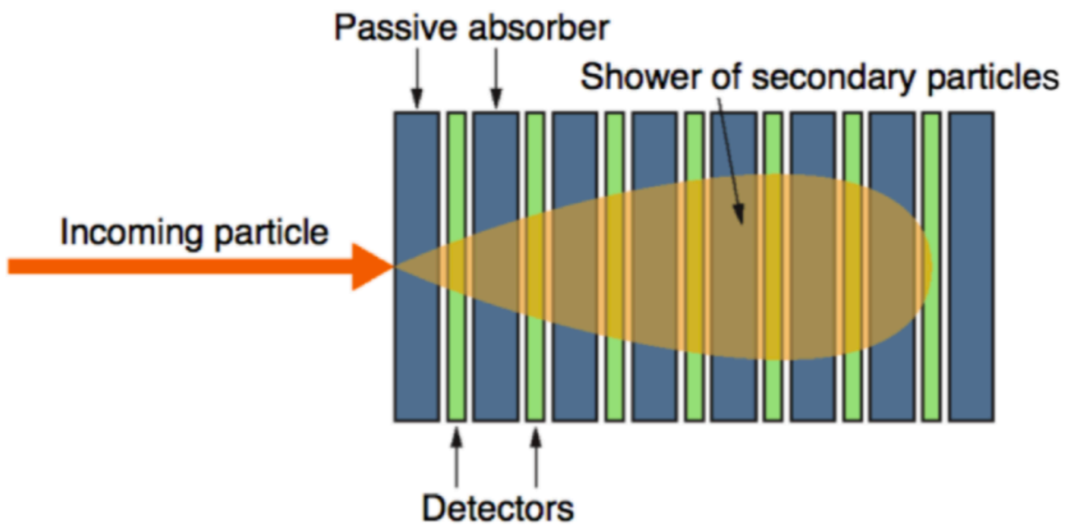


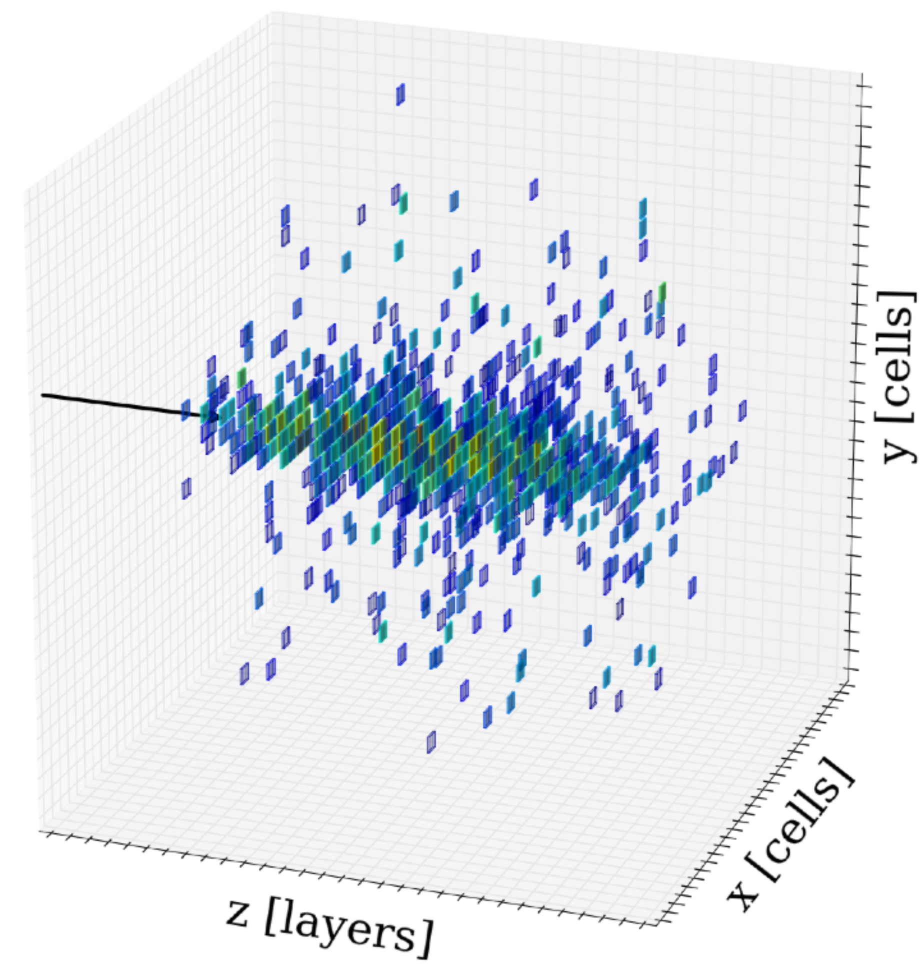


ILD Detector

- Shower in ILD Electromagnetic Calorimeter
- 30x30x30 cells (Si-W)
- Photon energies from 10 to 100 GeV
- Use 950k examples (uniform in energy) created with GEANT4 to train

Simulation target

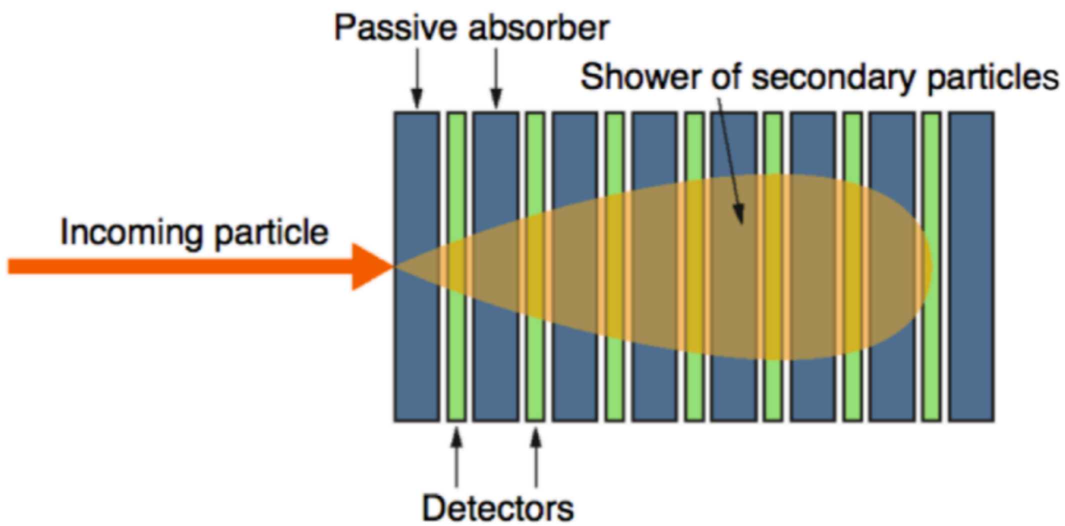


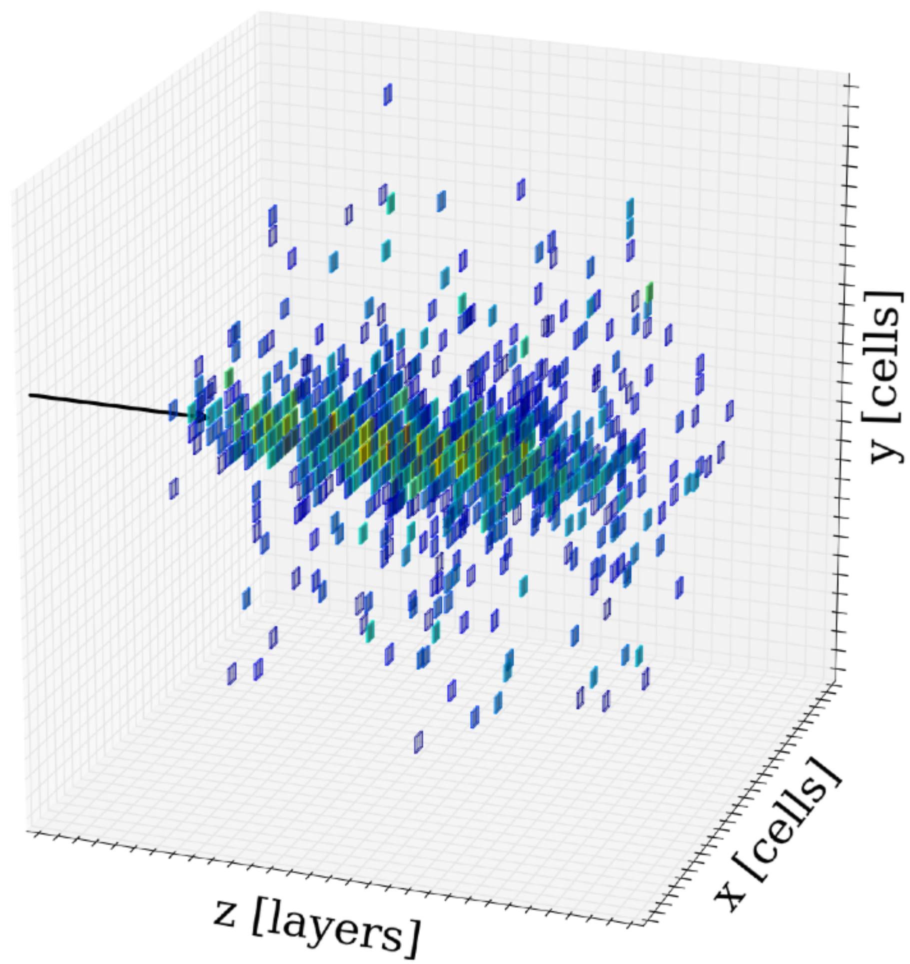


How to represent?

Tabular data:
Easy, insufficient for high-dimensions

Simulation target



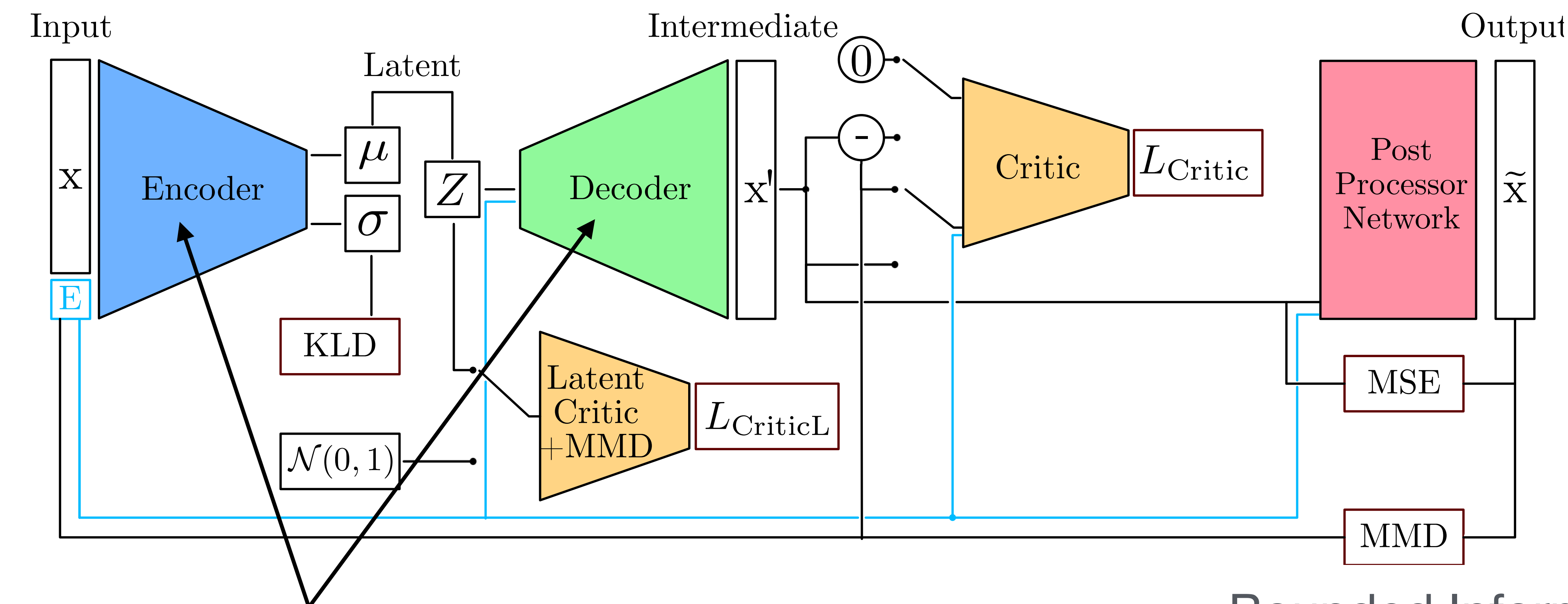


How to represent?

Tabular data

Fixed grid: Voxel image
(allows using e.g. convolutional networks)

Generative Architecture



(Transposed) Convolution

Bounded Information
Bottleneck AE
BIB-AE (GAN + VAE)

$$
\begin{aligned}
L_{\text{BIB-AE}} = & -\beta_{C_L} \cdot \mathbb{E}[C_L(N_E(x))] \\
& -\beta_C \cdot \mathbb{E}[C_E(D_E(N_E(x)))] \\
& -\beta_{C_D} \cdot \mathbb{E}[C_{D,E}(D_E(N_E(x)) - x)] \\
& +\beta_{\text{KLD}} \cdot \text{KLD}(N_E(x)) \\
& +\beta_{\text{MMD}} \cdot \text{MMD}(N_E(x), \mathcal{N}(0,1)).
\end{aligned}
$$

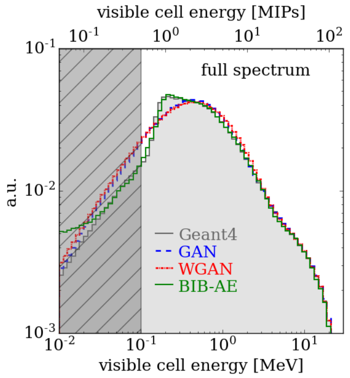
Latent Critic

Critic

Difference
Critic

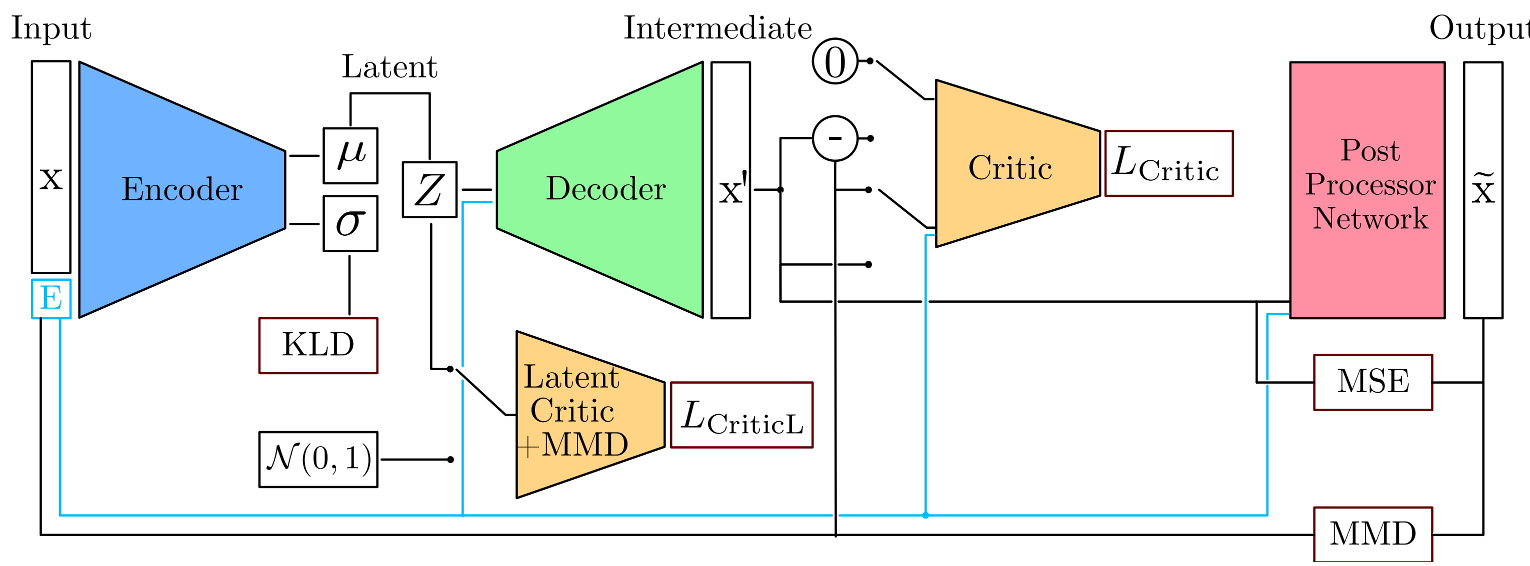
Latent Regularisation

1912.00830, 2005.05334



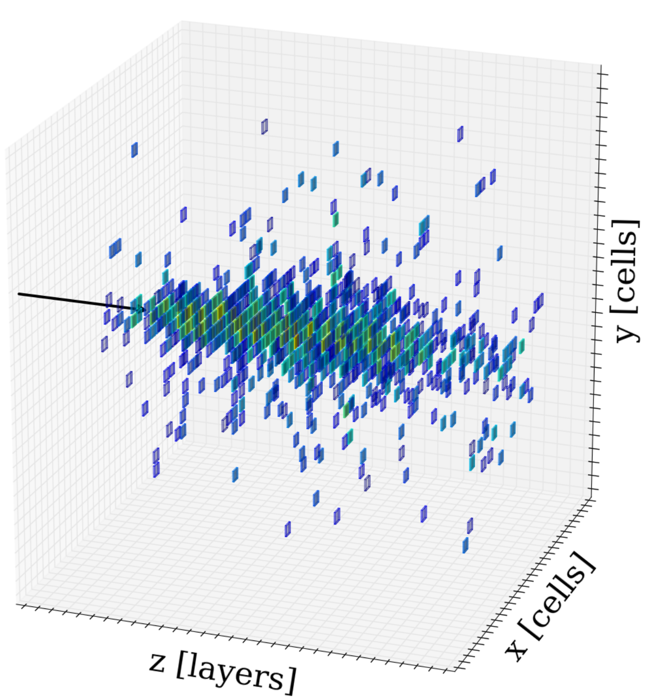
Generative progress

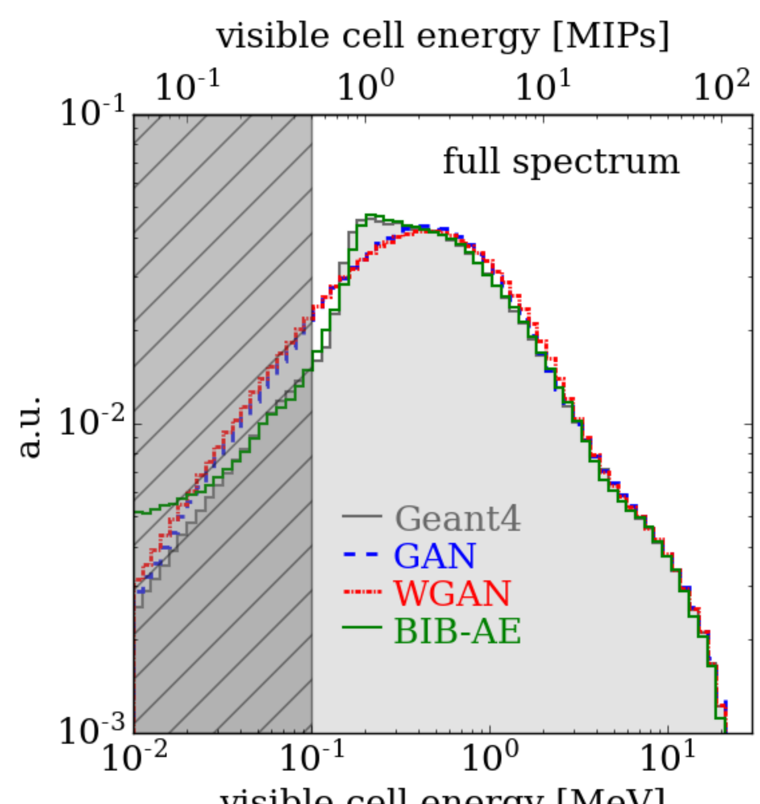
Progress



BIB-AE (GAN + VAE):

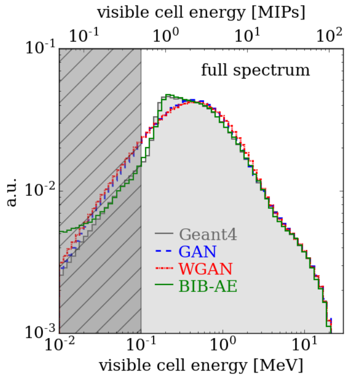
1st simulation of Photon shower in 27k cell calorimeter

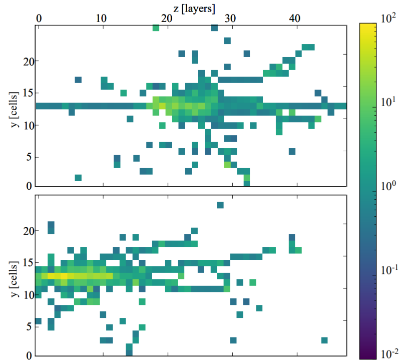




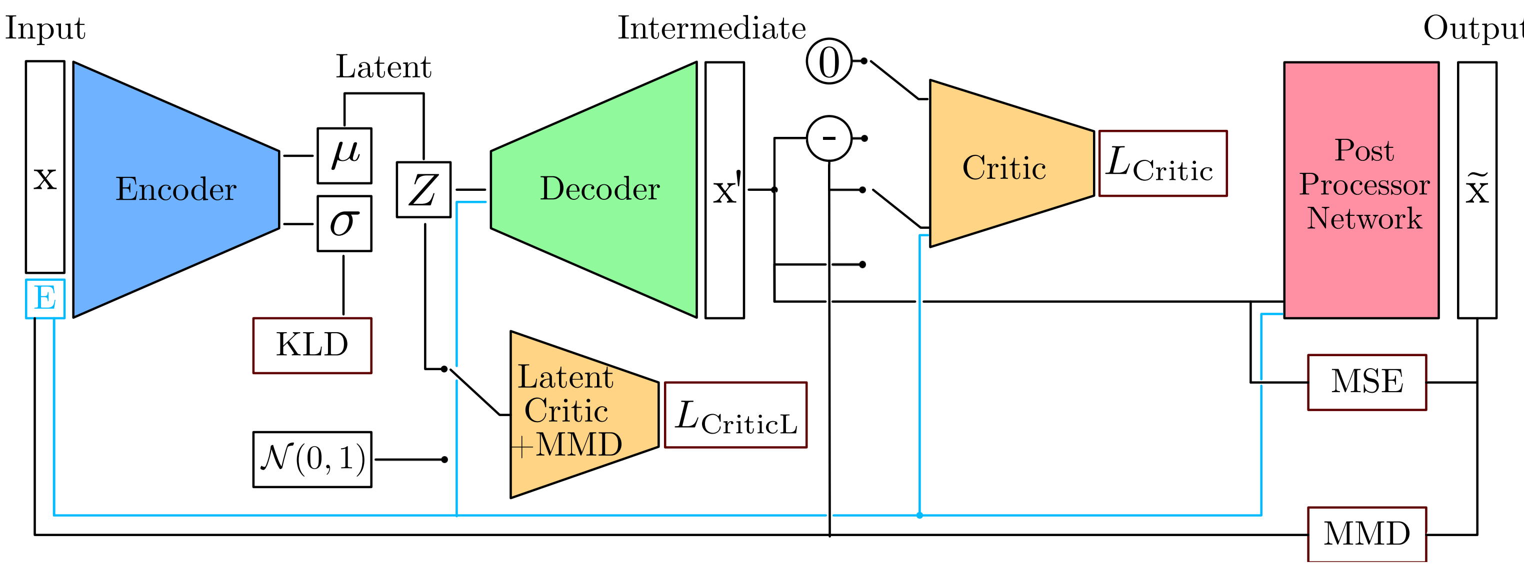
Buhmann, .., **GK** et al 2005.05334

erative progress

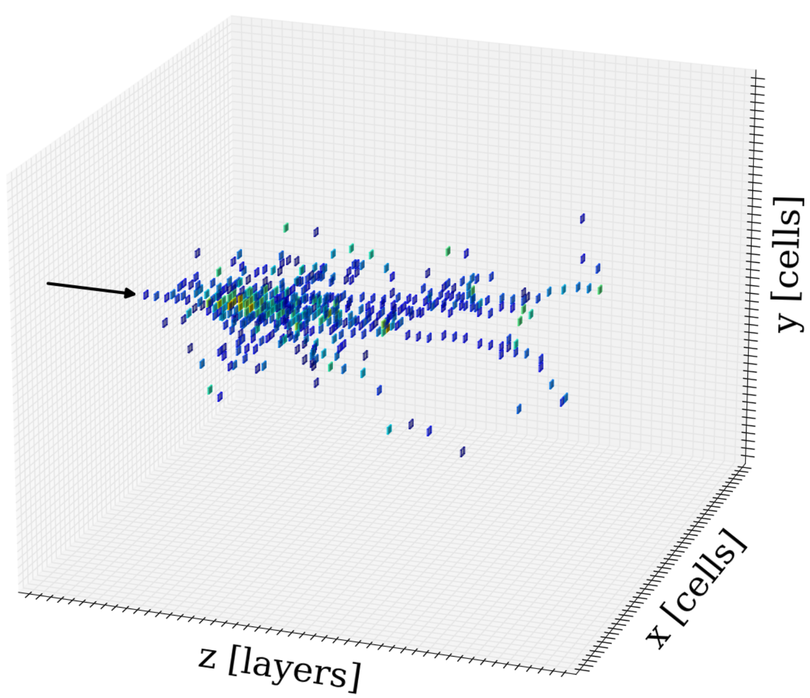


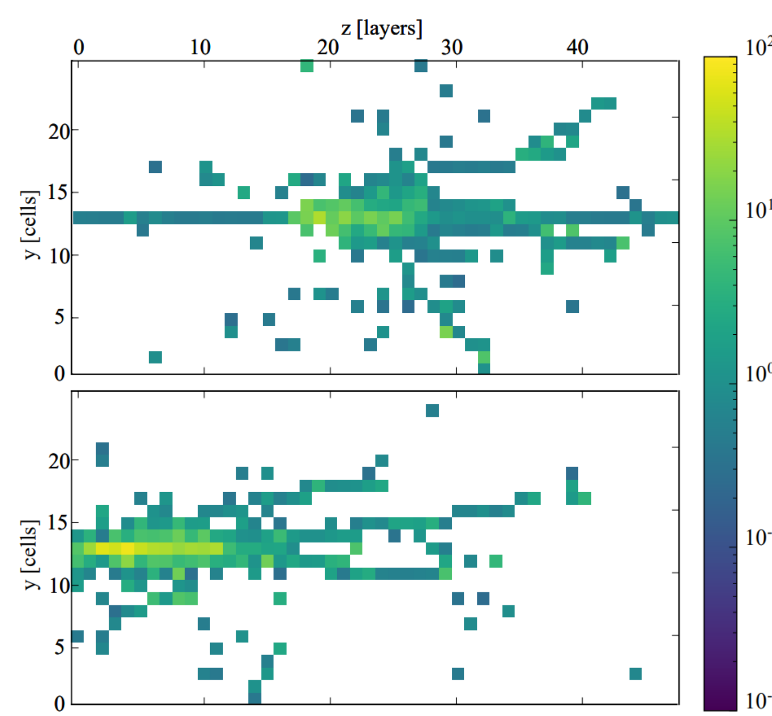


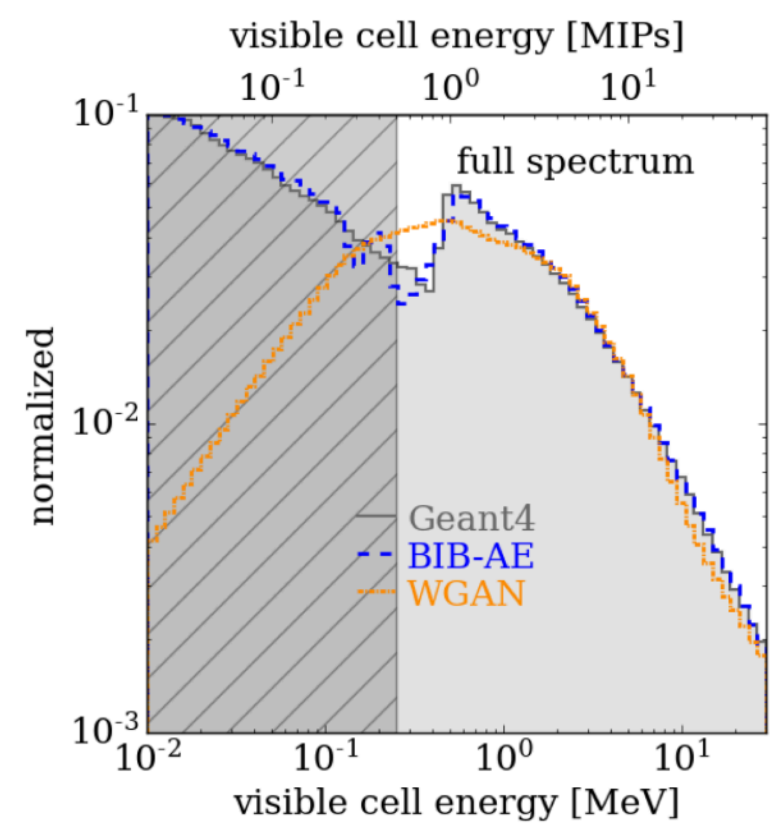
Progress



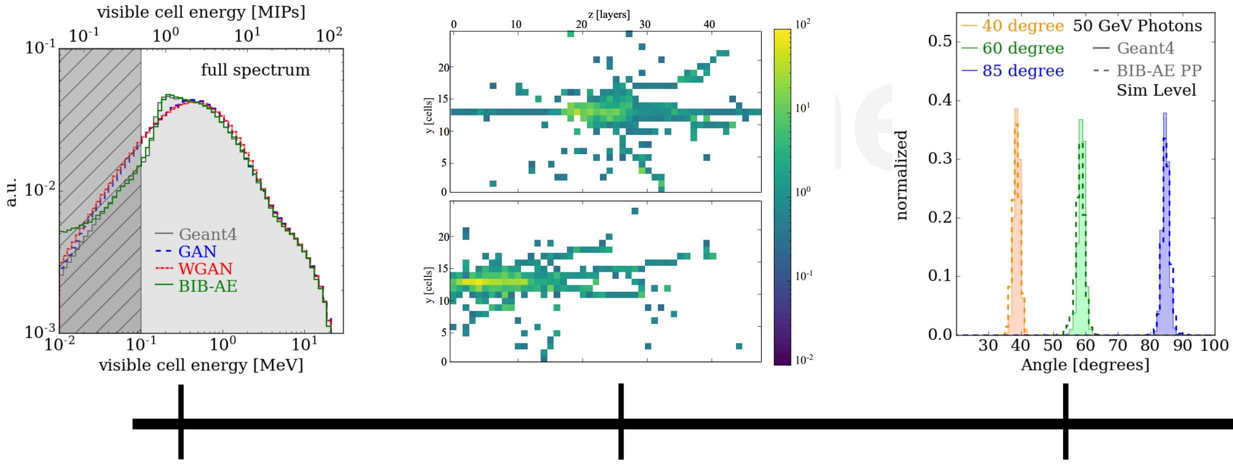
Handle more complex pion showers in hadronic calorimeter



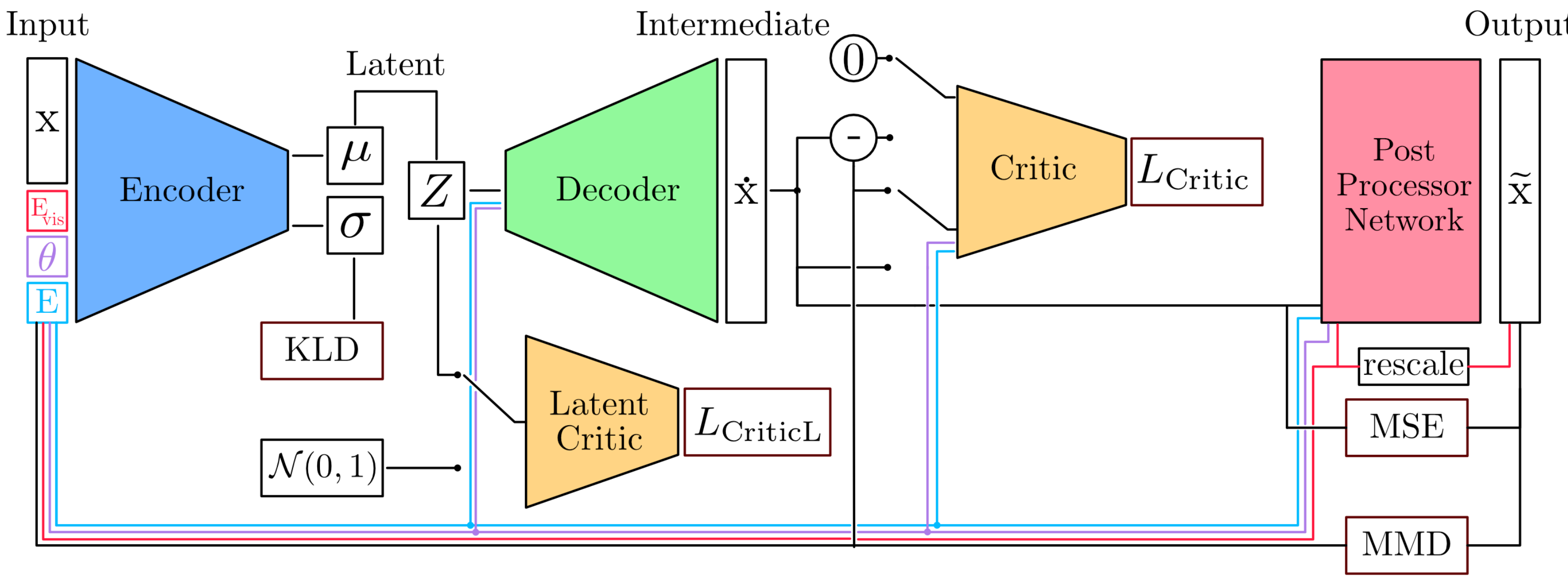


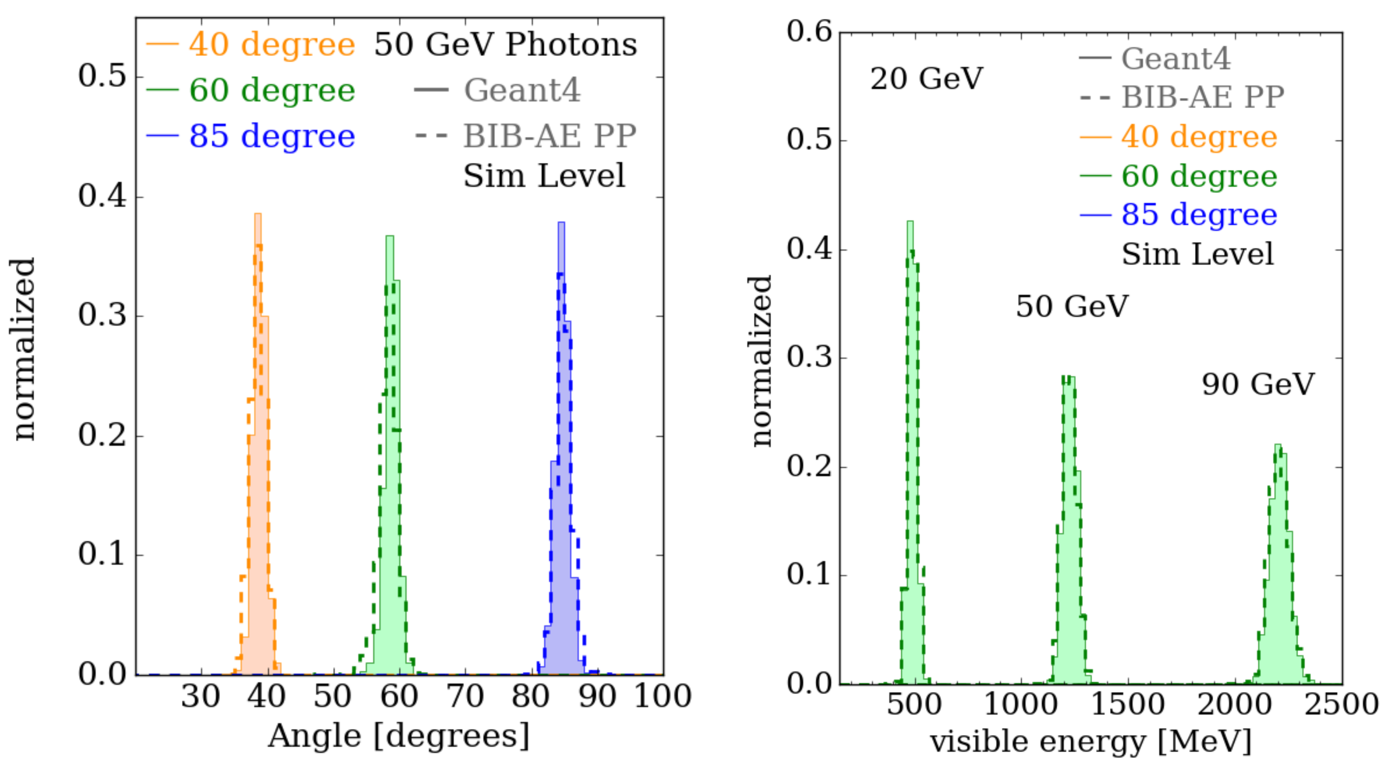


Buhmann, .., **GK** et al 2112.09709;



Progress

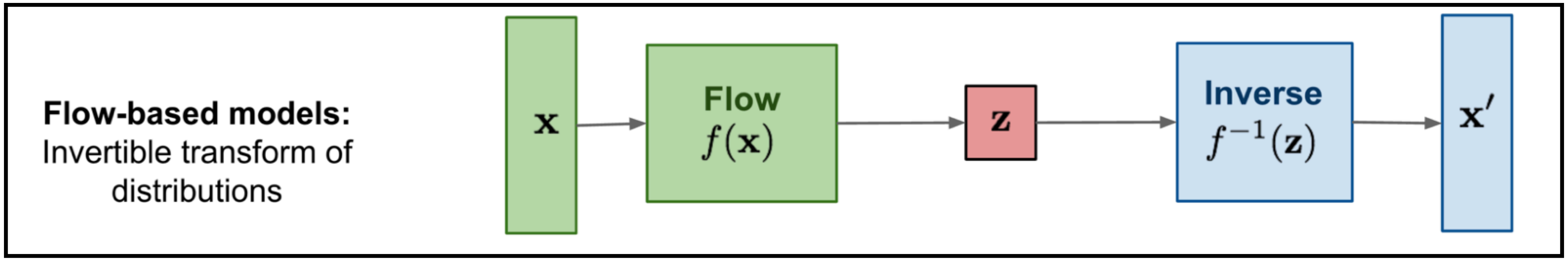




Extend to condition on angles

Diefenbacher, .., **GK** et al 2303.18150

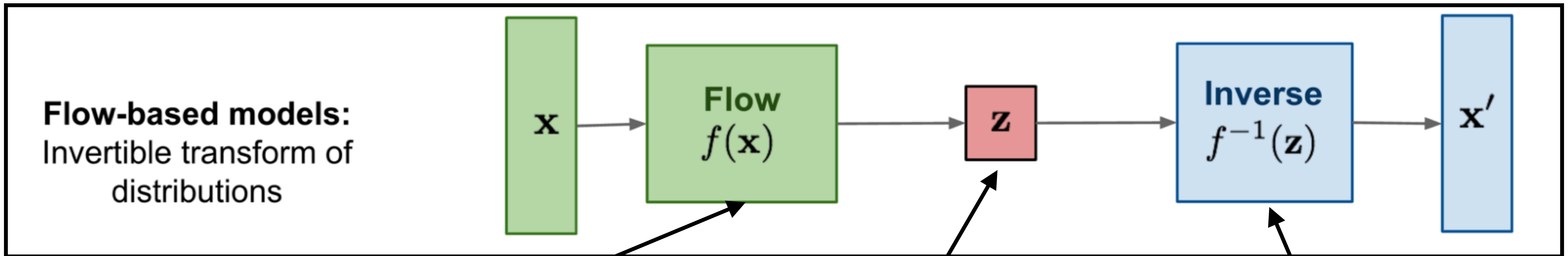
Normalising Flows



In auto-encoders, the decoder learns to ‘undo’ the encoder

Can we make this exact and directly learn the likelihood?

Normalising Flows



Take into account Jacobian determinant to evaluate probability density

Choose latent space, e.g. standard normal distribution (normalising flow!) Same dimension as data!

f^{-1} is not a learned inversion, but exact inverse by construction

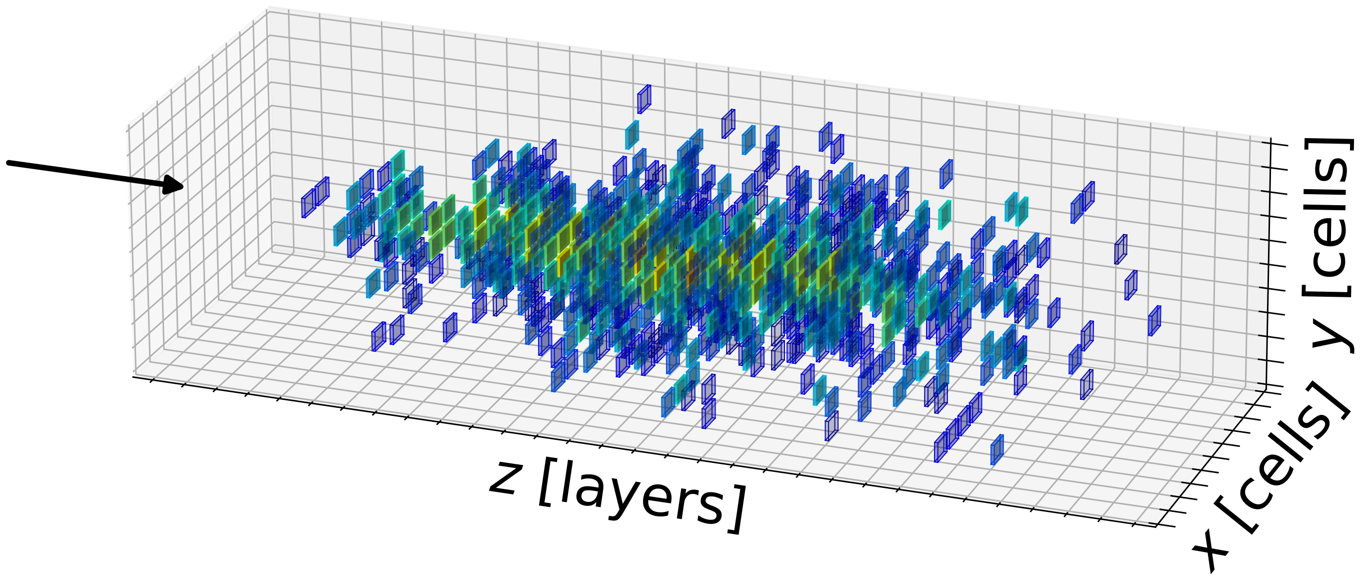
Learn a diffeomorphism between data and latent-space

Bijective, invertable

Learn likelihood of data

1505.05770, 1908.09257

Flows for detector simulation



10x10 cells / layer
30 layers

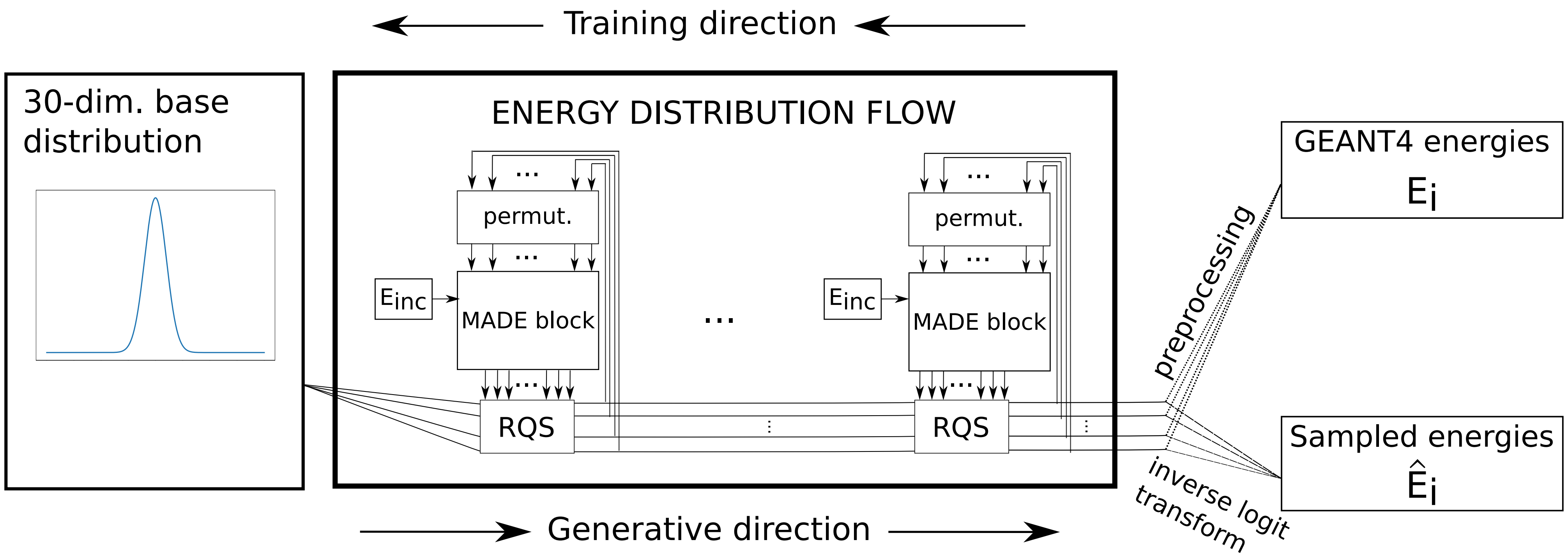
By directly learning the likelihood, flows should be of higher fidelity than GAN/VAE.

But inefficient scaling with data dimension.

How to do flows for high-dimensional data?

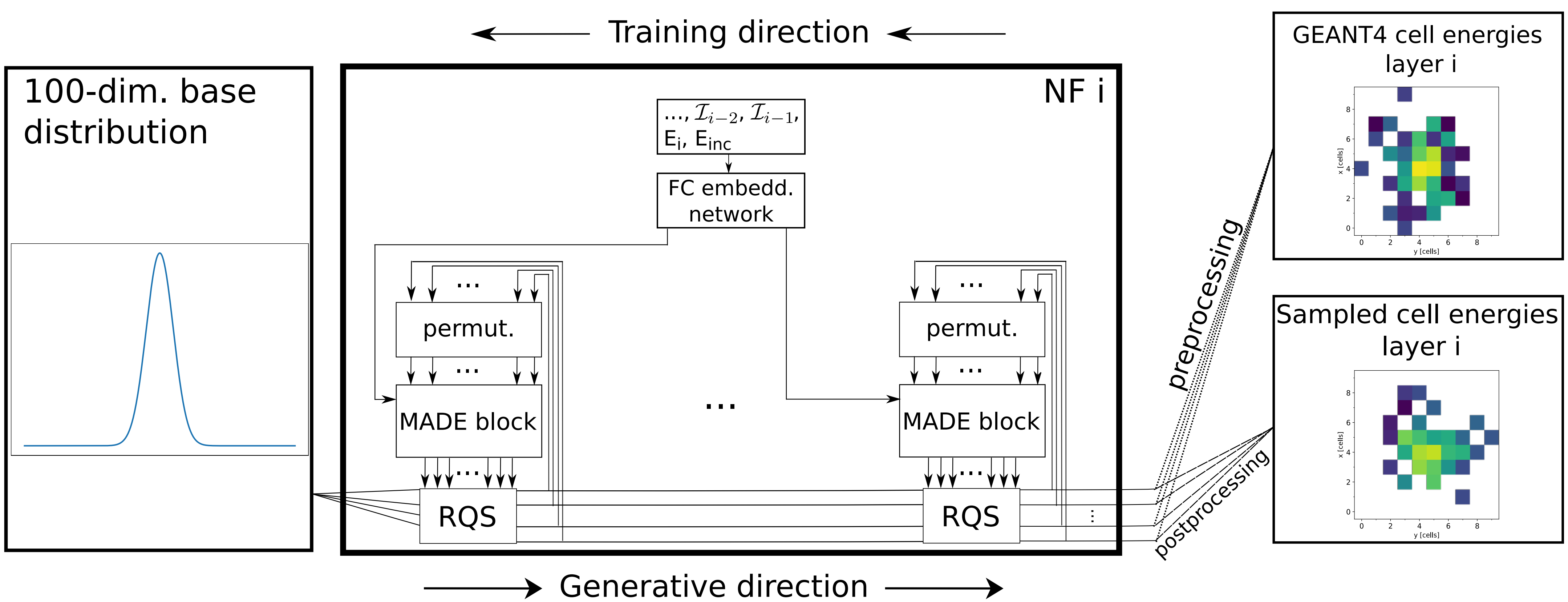
2302.11594

Flows for detector simulation

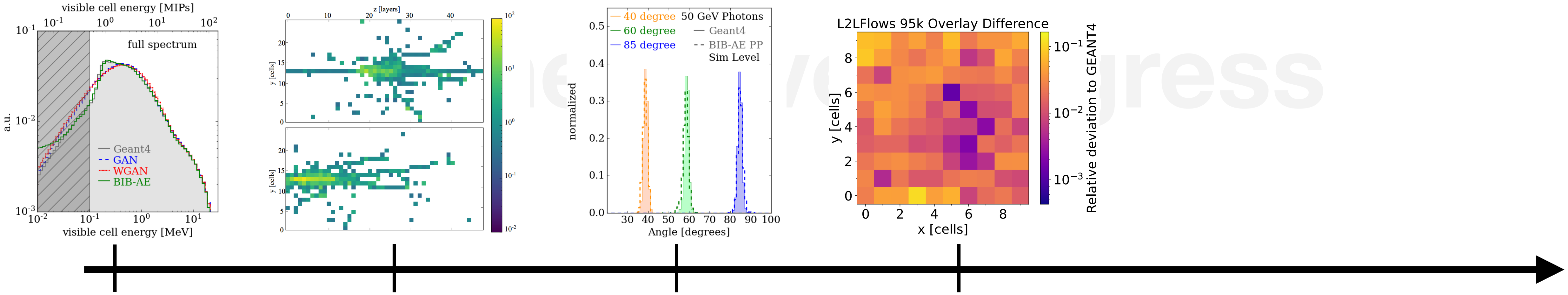


How to flows for high-dimensional data?

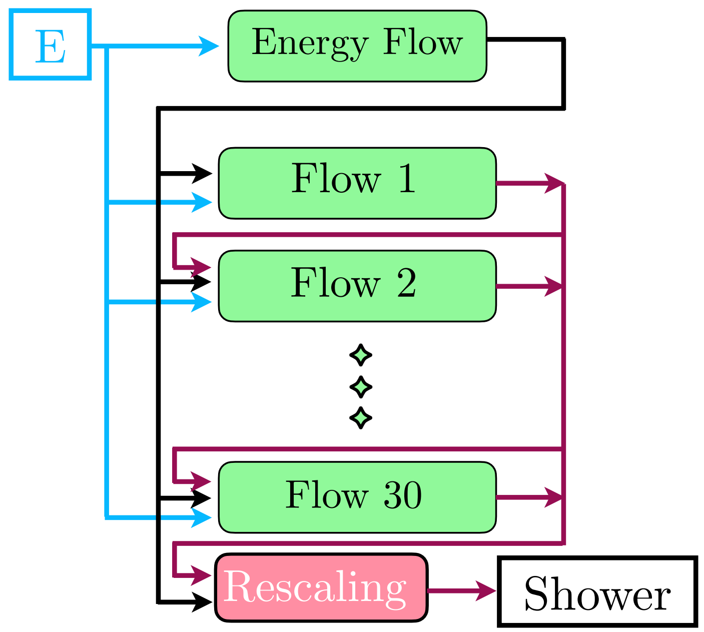
Split!



Diefenbacher, .., **GK** et al 2302.11594

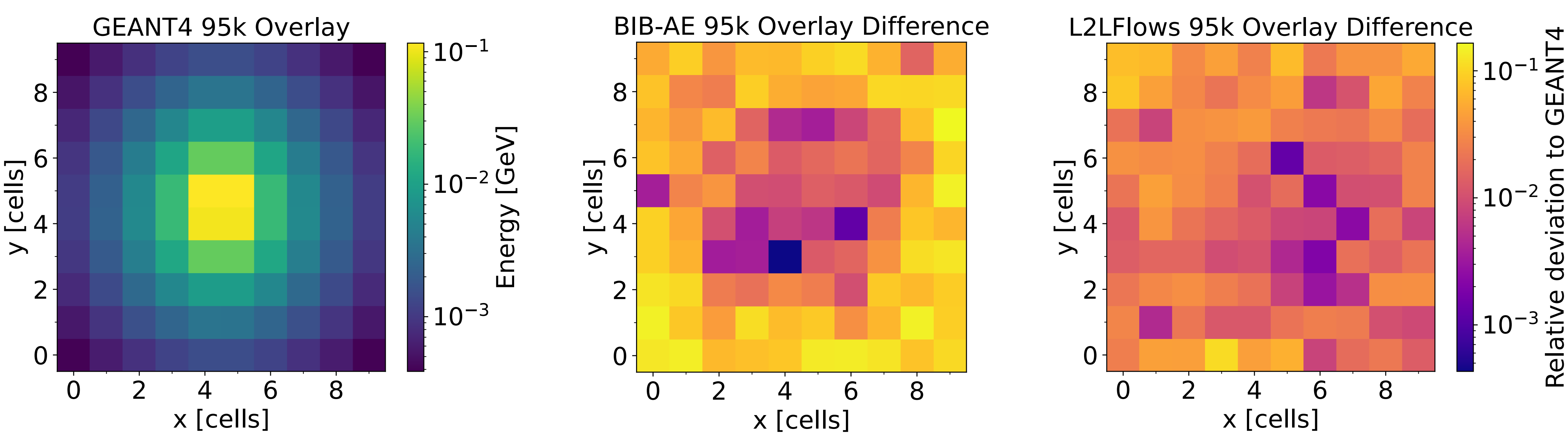


Progress



Better convergence of normalising flows: →better fidelity

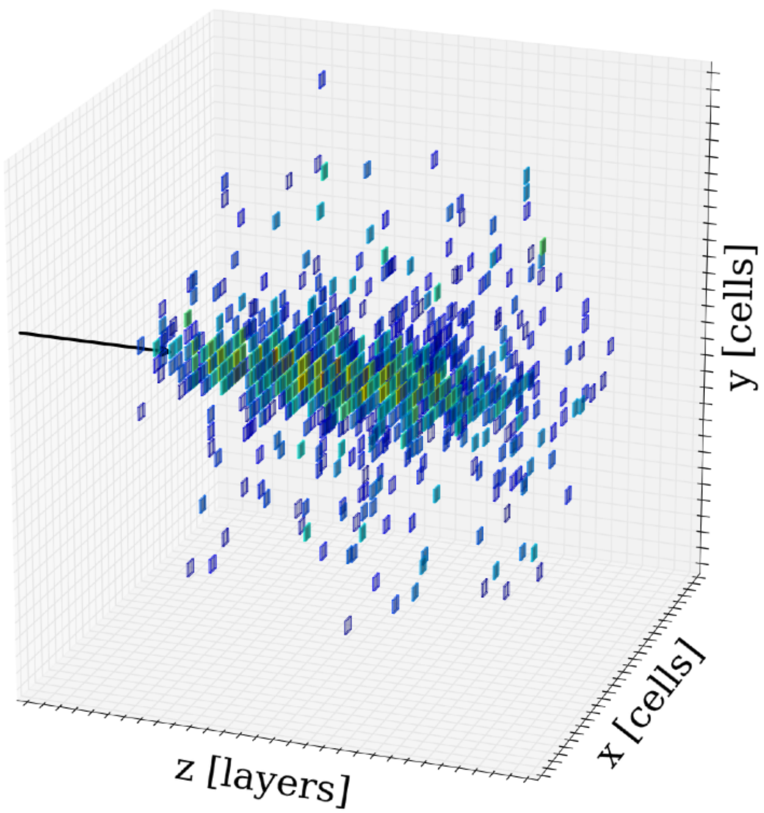
Individual flows per layer for efficiency



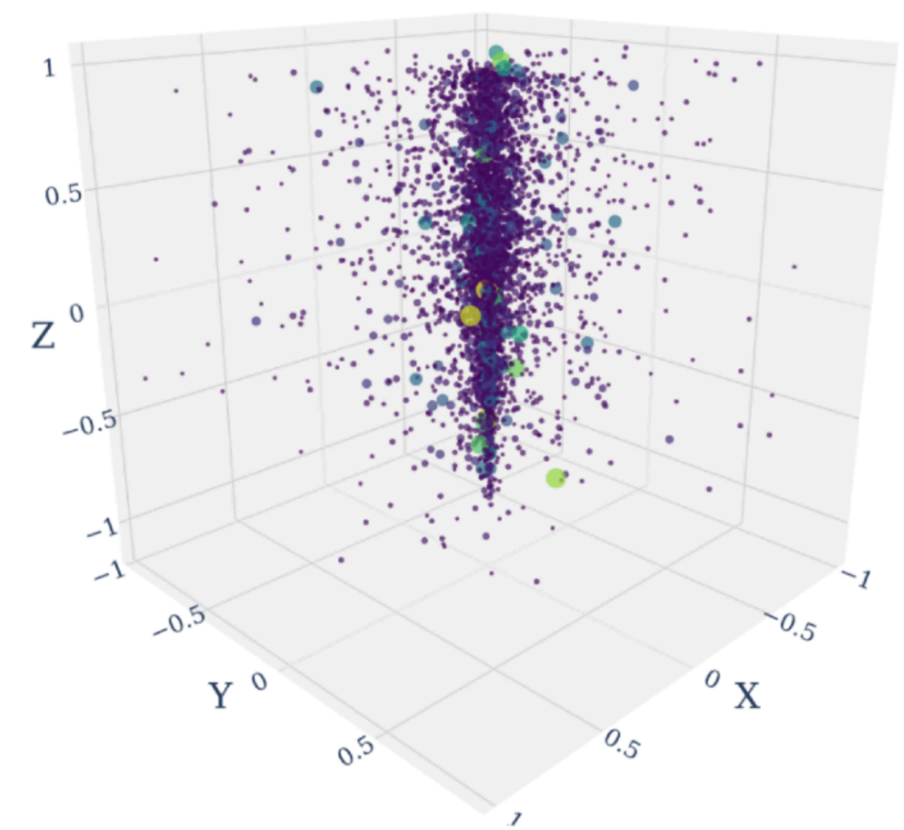
Diefenbacher, .., **GK** et al 2302.11594

CaloClouds

To improve the generative fidelity, move to a point cloud diffusion model



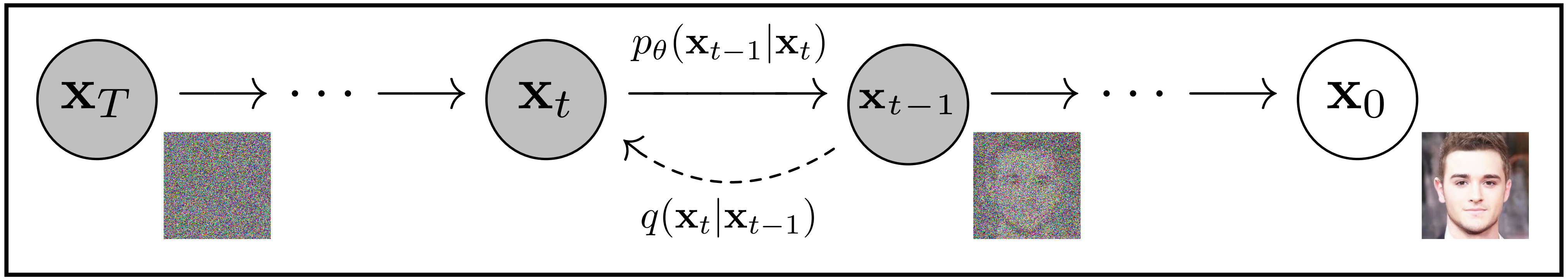
Fixed grid (voxels)
Limiting for high-dimensions (sparse data)



Point cloud:
Only simulate non-zero hits → better scaling

CaloClouds

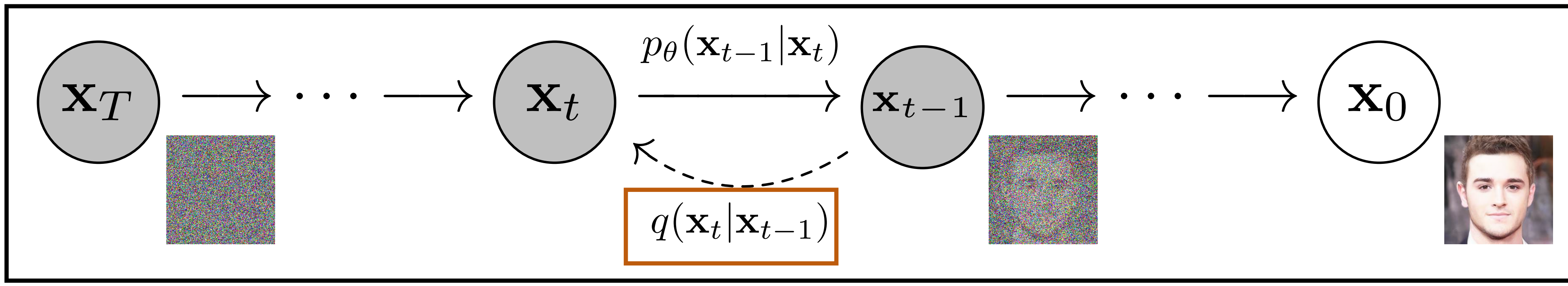
To improve the generative fidelity, move to a point cloud diffusion model



Core idea: Stepwise noising/ denoising

Diffusion

To improve the generative fidelity, move to a point cloud diffusion model



Forward
(Data → Noise)

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) := \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I})$$

Individual step

Noise schedule
(hyper-parameter)

$$\mathbf{x}_t(\mathbf{x}_0, \boldsymbol{\epsilon}) = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon} \text{ for } \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

Rewrite: State at any time

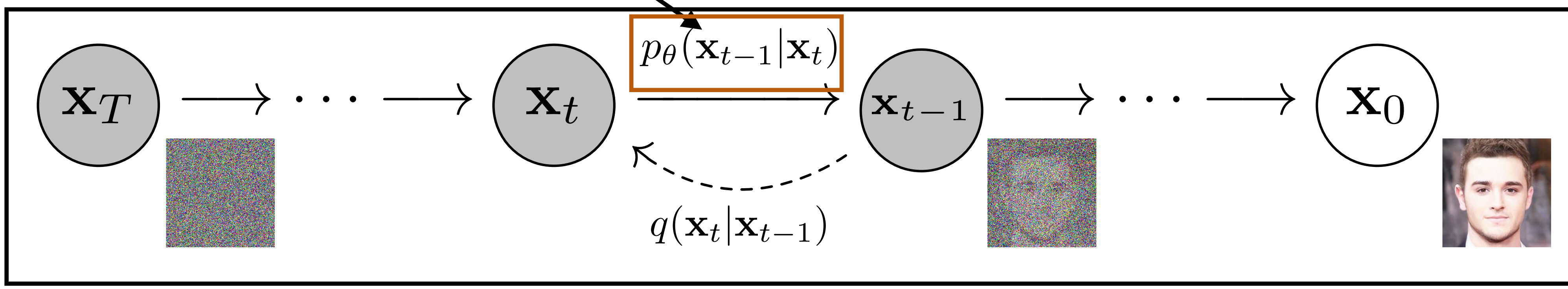
Will try to predict

$$\alpha_t := 1 - \beta_t \qquad \bar{\alpha}_t := \prod_{s=1}^{t} \alpha_s$$

Diffusion

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) := \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$$

To improve the generative fidelity, move to a point cloud diffusion model



Backward
(Noise → Data)

$$L_{\text{simple}}(\theta) := \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2\right]$$

Noisy image

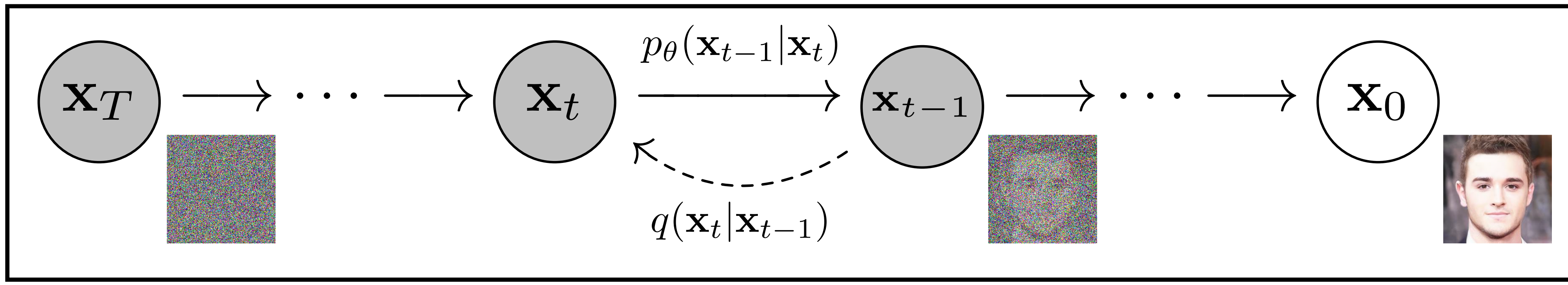
Reminder: Forward diffusion to time t

$$\mathbf{x}_t(\mathbf{x}_0, \boldsymbol{\epsilon}) = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$$

Timestep

Diffusion

To improve the generative fidelity, move to a point cloud diffusion model



Algorithm 1 Training

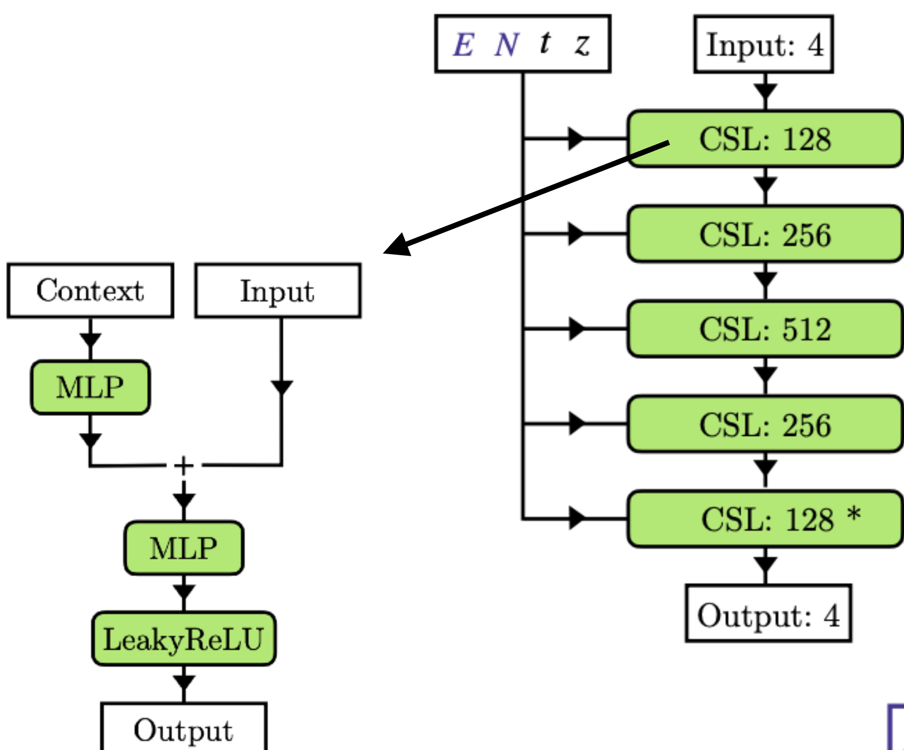
repeat
  $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
  $t \sim \mathrm{Uniform}(\{1, \ldots, T\})$
  $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  Take gradient descent step on
  $$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$
until converged

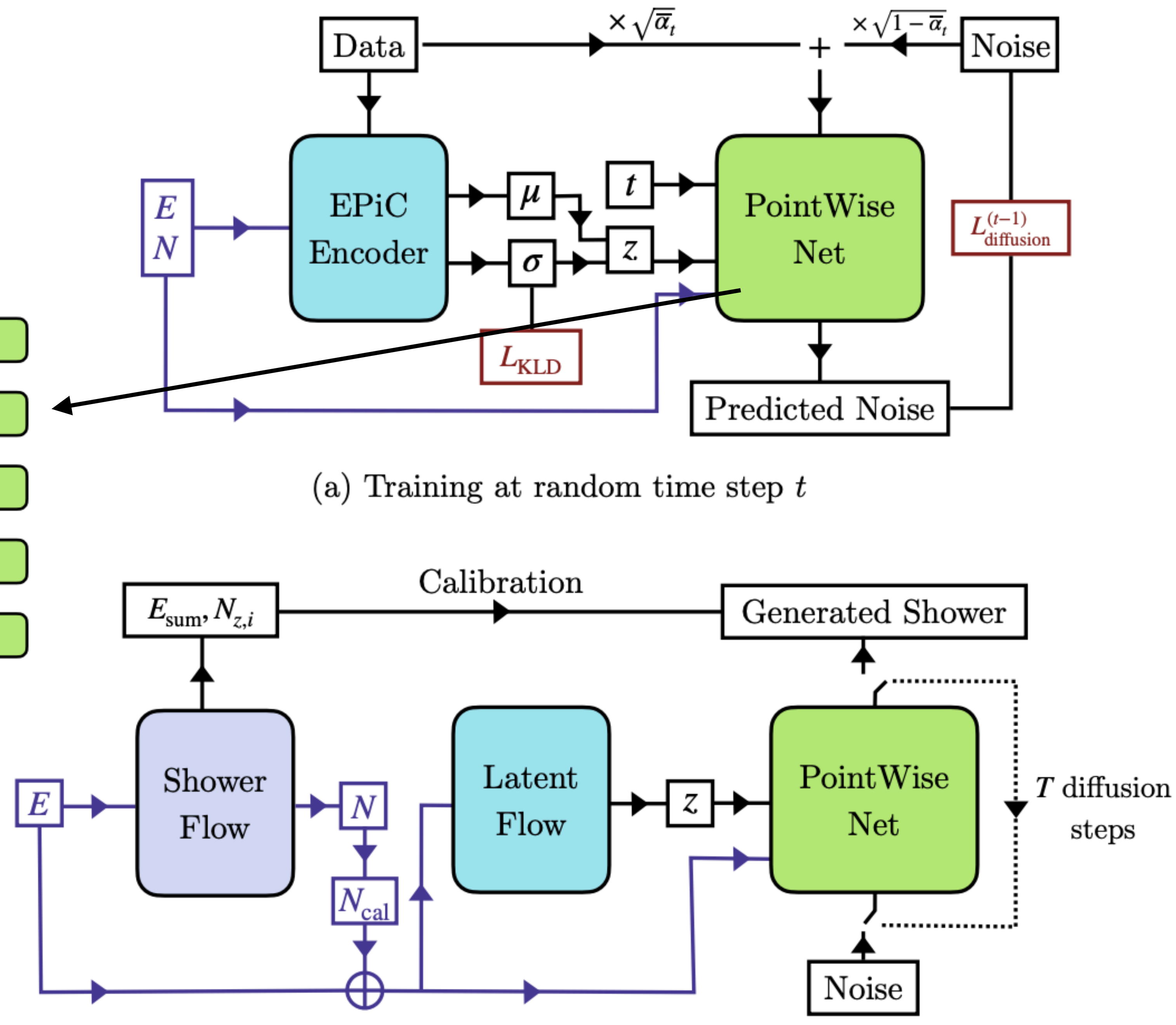
Algorithm 2 Sampling

$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, \ldots, 1$ **do**
  $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
  $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}$
end for
return $\mathbf{x}_0$

CaloClouds

To improve the generative fidelity, move to a point cloud diffusion model





(a) Training at random time step t

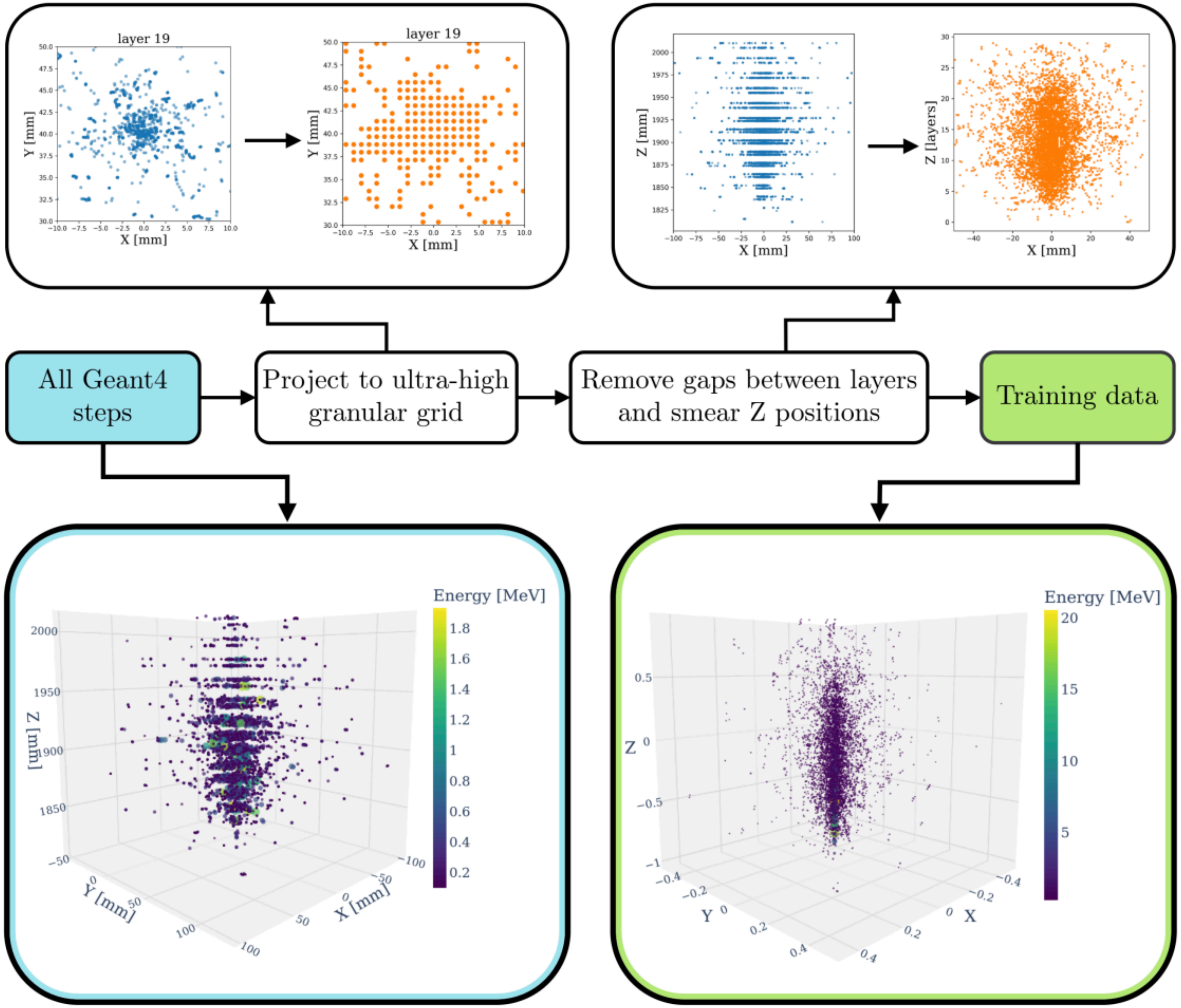
(b) Sampling with reverse diffusion through all time steps T

Buhmann, … GK, et al 2305.04847

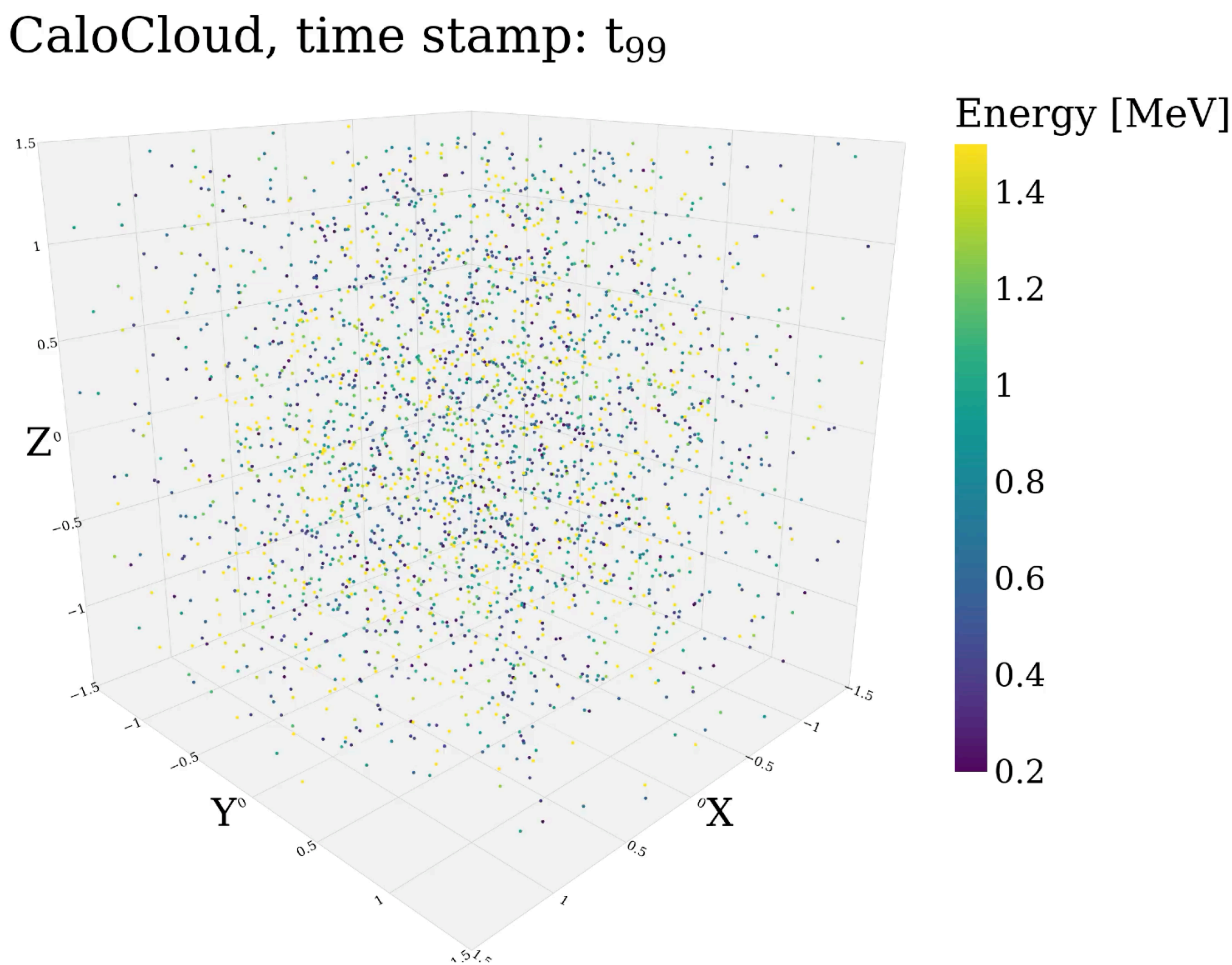
CaloClouds

To improve the generative fidelity, move to a point cloud diffusion model

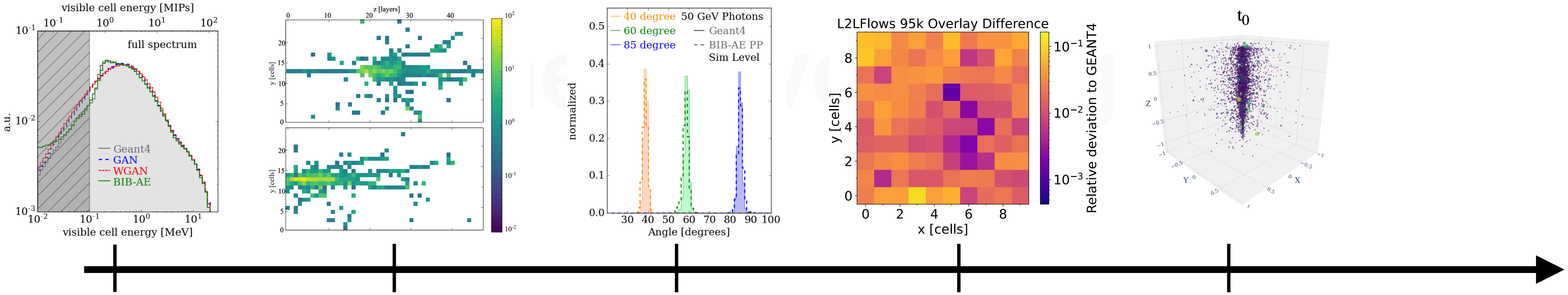
Some input processing needed



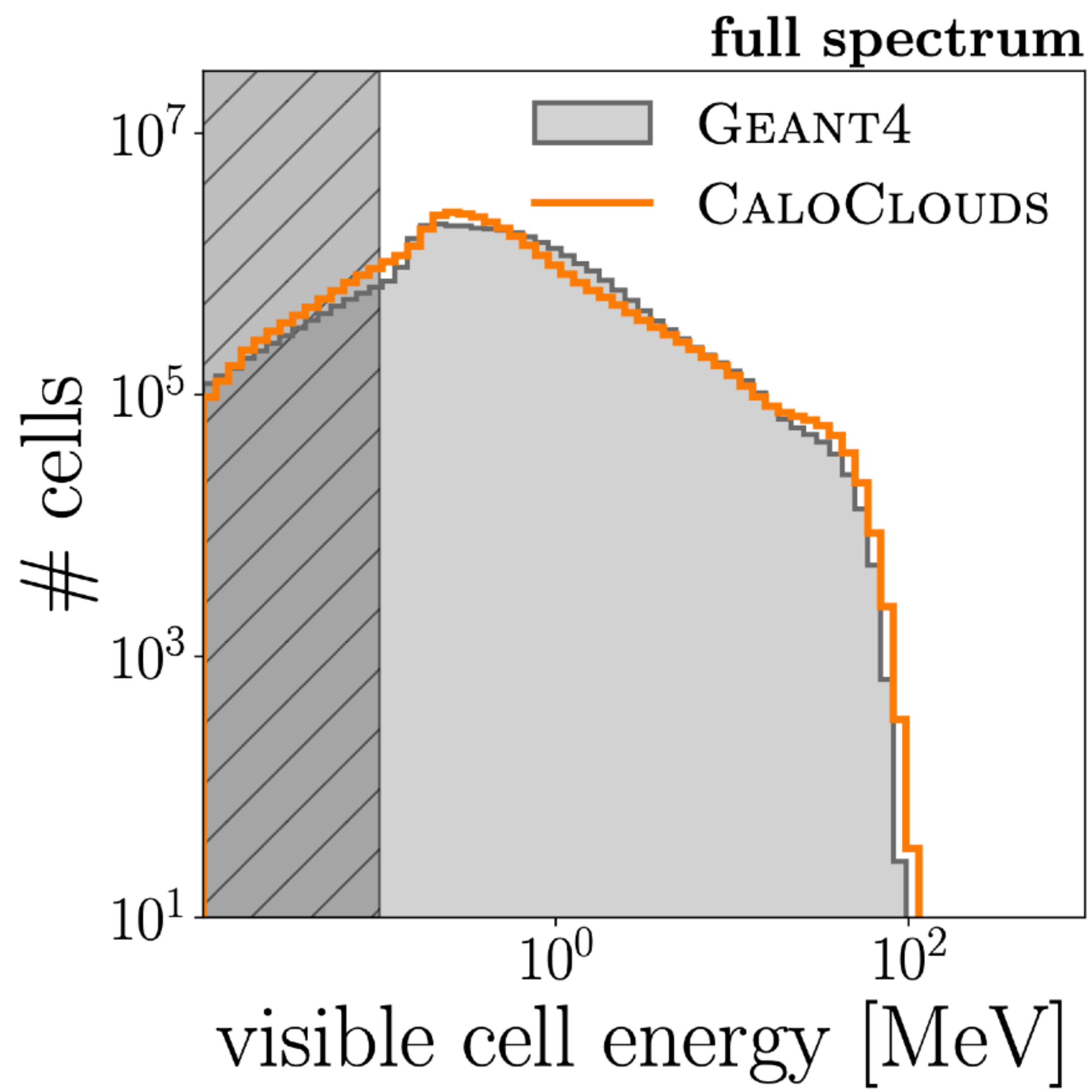
CaloClouds



Buhmann, … GK, et al 2305.04847



To improve the generative fidelity, move to a point cloud diffusion model

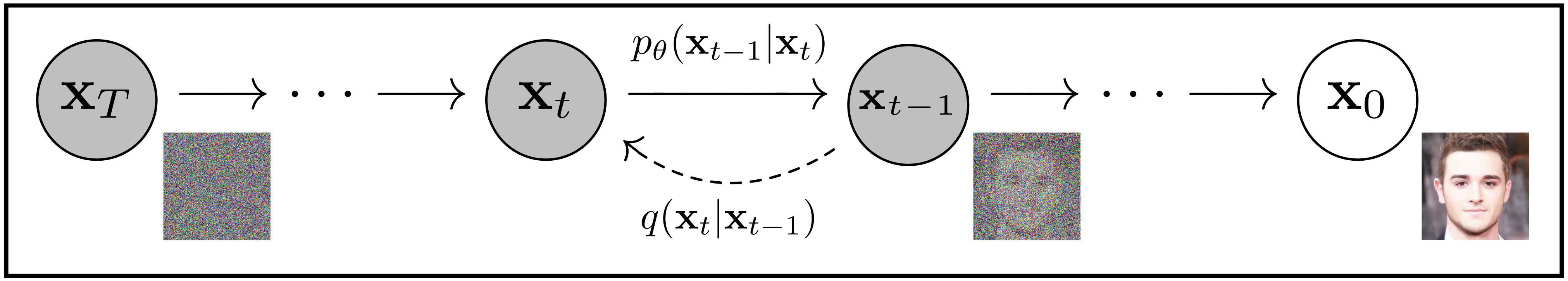


Close in accuracy to fixed grid.
Fairly slow.

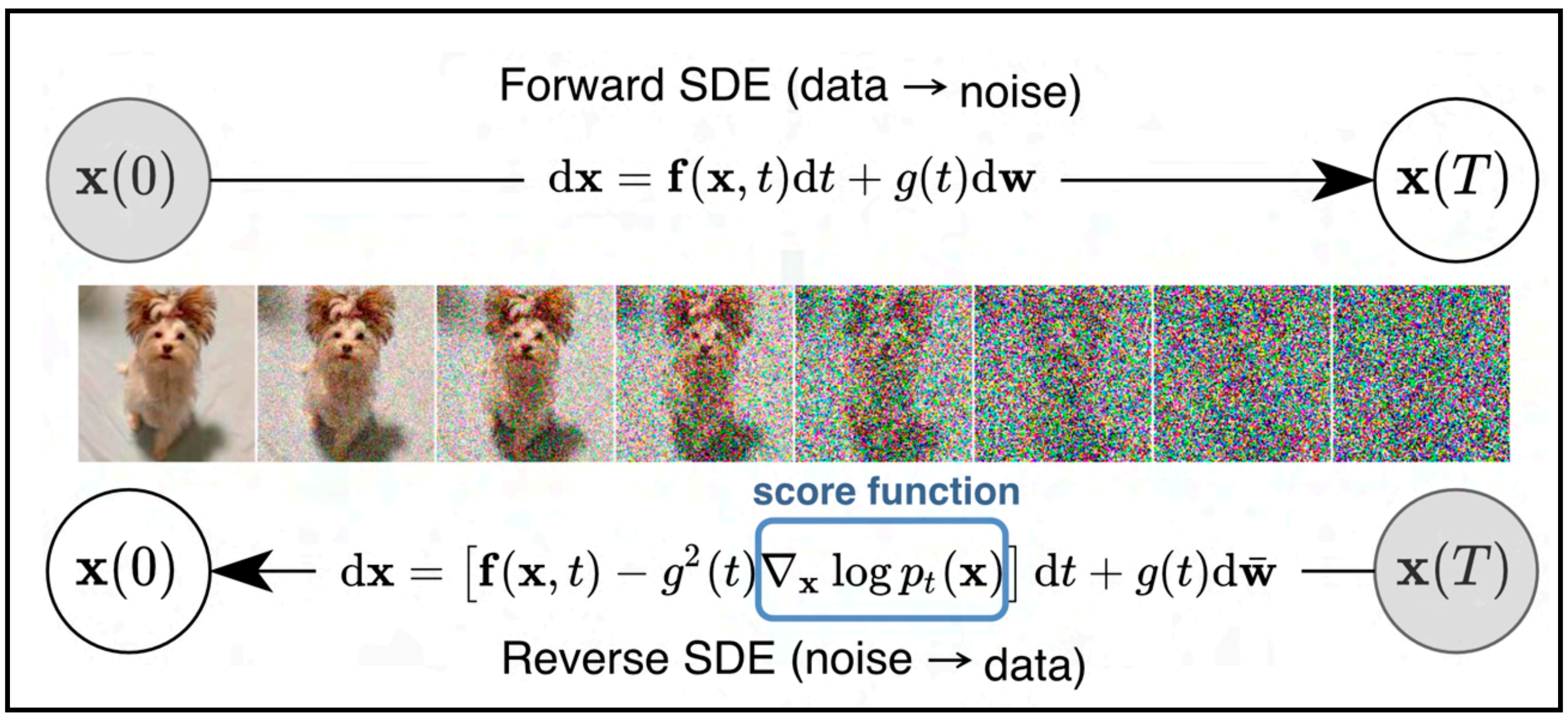
Can we improve further?

Buhmann, .., **GK**, et al 2305.04847

CaloClouds II



Replace discrete noise steps



with stochastic differential equations (SDEs)

2011.13456

CaloClouds II

$$L_{\mathrm{simple}}(\theta) := \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2\right]$$

Replace learning added noise

↓

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathbb{E}_t\Big\{\lambda(t)\mathbb{E}_{\mathbf{x}(0)}\mathbb{E}_{\mathbf{x}(t)|\mathbf{x}(0)}\Big[\left\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}(t),t) - \nabla_{\mathbf{x}(t)}\log p_{0t}(\mathbf{x}(t)\mid\mathbf{x}(0))\right\|_2^2\Big]\Big\}$$

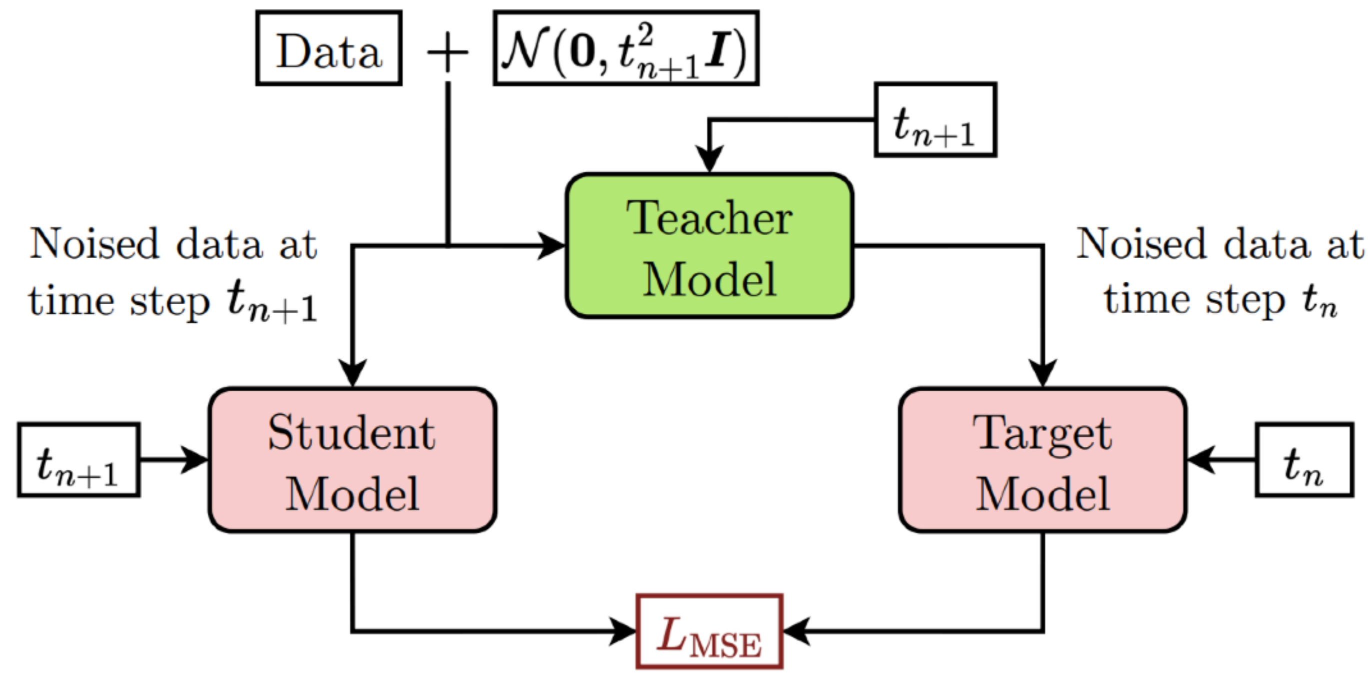
with learning a score function with conditional probability paths

$$\mathrm{d}\mathbf{x} = [\mathbf{f}(\mathbf{x},t) - g(t)^2\nabla_{\mathbf{x}}\log p_t(\mathbf{x})]\mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}}$$

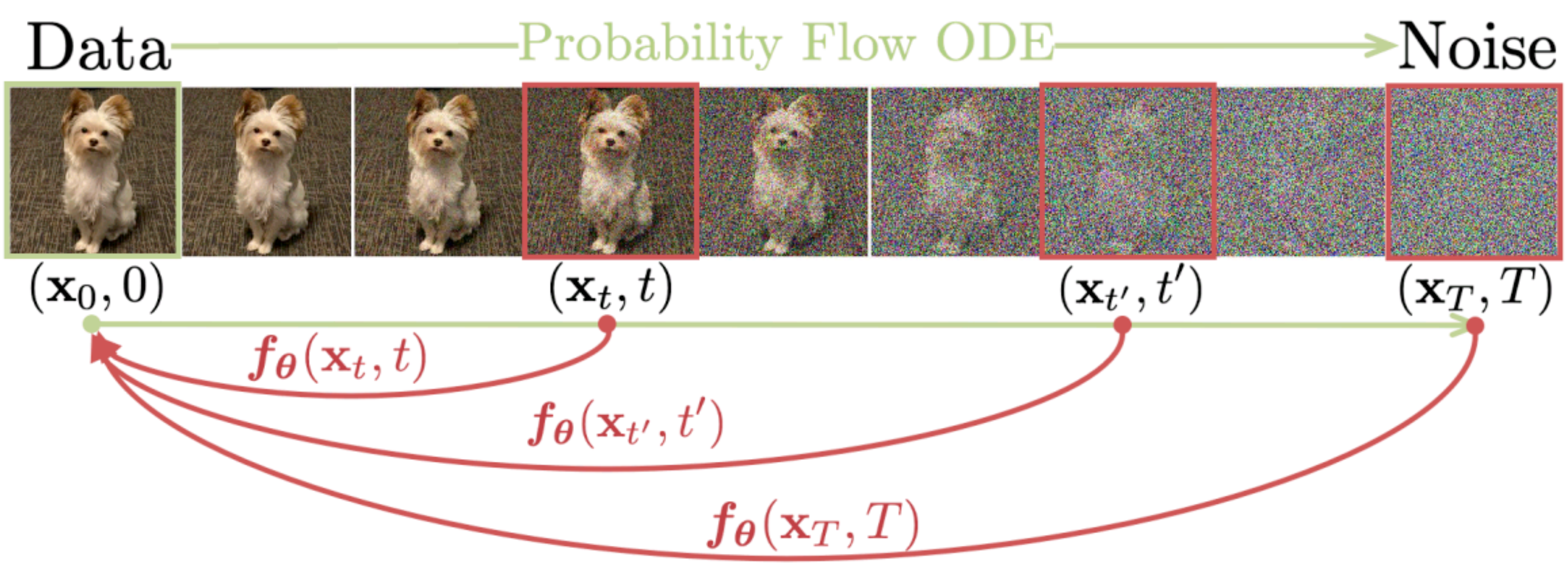
and numerically solve SDE to transport to data space

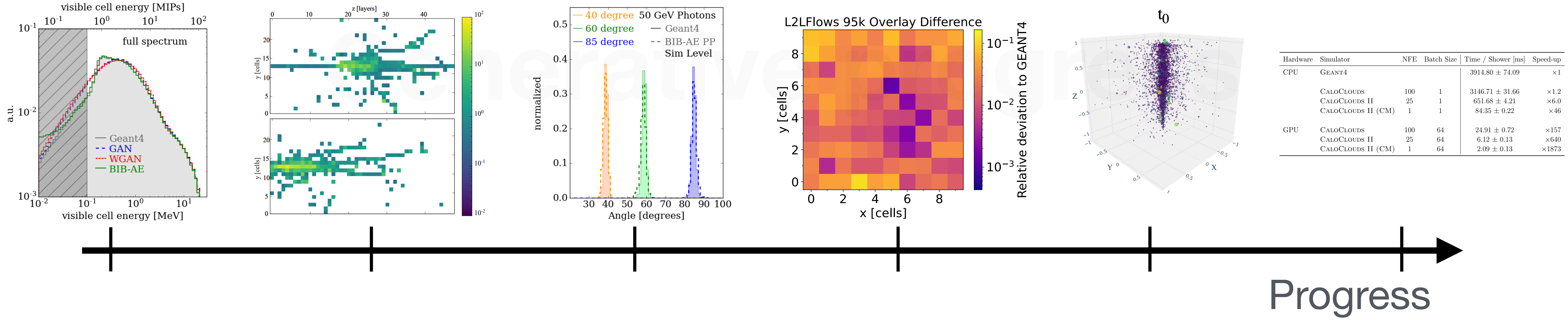
2011.13456

Consistency Distillation



Speed up by training a model to allow single step generation

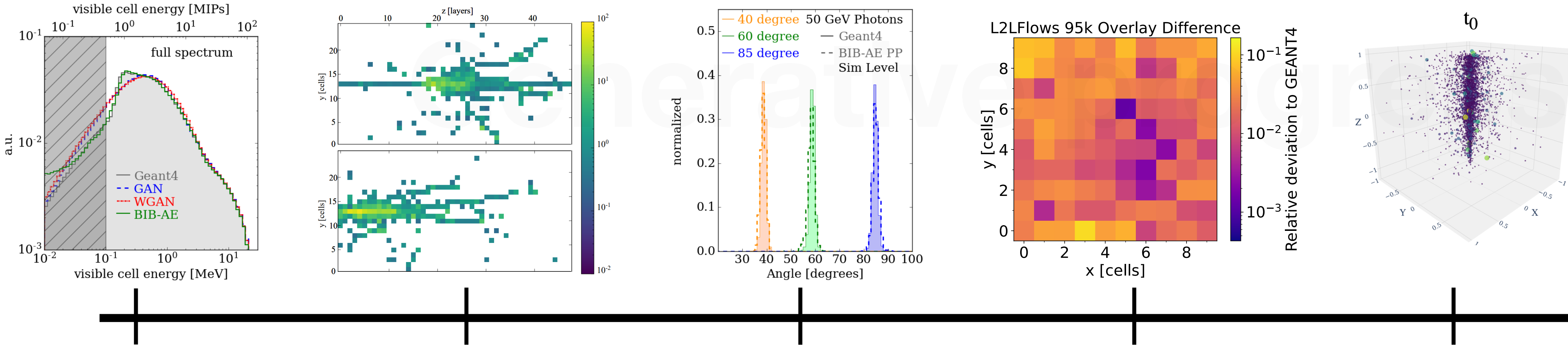




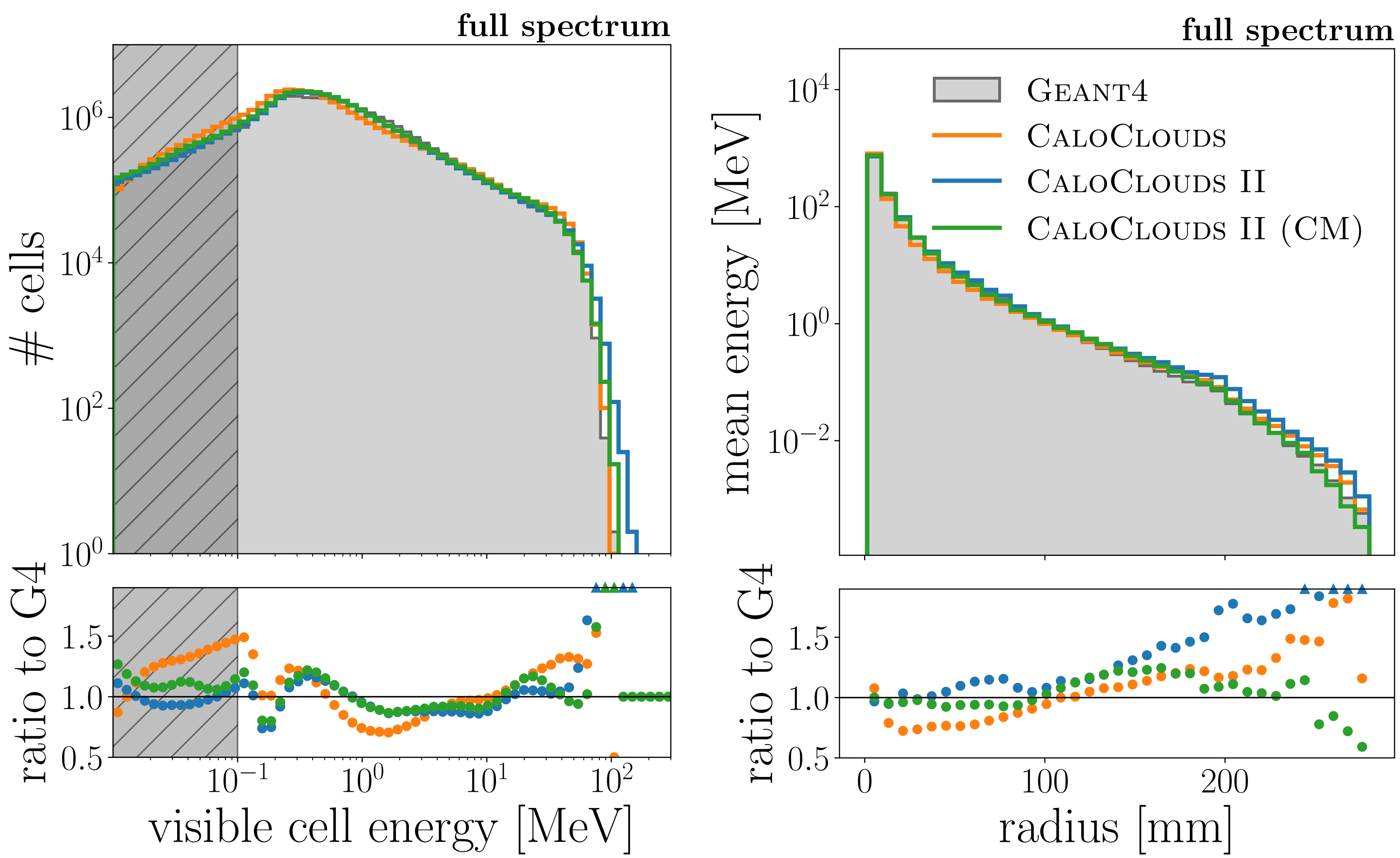
Hardware	Simulator	NFE	Batch Size	Time / Shower [ms]	Speed-up
CPU	GEANT4			3914.80 ± 74.09	×1
	CALOCLOUDS	100	1	3146.71 ± 31.66	×1.2
	CALOCLOUDS II	25	1	651.68 ± 4.21	×6.0
	CALOCLOUDS II (CM)	1	1	84.35 ± 0.22	×46
GPU	CALOCLOUDS	100	64	24.91 ± 0.72	×157
	CALOCLOUDS II	25	64	6.12 ± 0.13	×640
	CALOCLOUDS II (CM)	1	64	2.09 ± 0.13	×1873

Use continuous time diffusion and consistency distillation: Better quality and faster

Buhmann, .., **GK** et al 2309.05704



Hardware	Simulator	NFE	Batch Size	Time / Shower [ms]	Speed-up
CPU	Geant4			3914.80 ± 74.09	×1
	CaloClouds	100	1	3146.71 ± 31.66	×1.2
	CaloClouds II	25	1	651.68 ± 4.21	×6.0
	CaloClouds II (CM)	1	1	84.35 ± 0.22	×46
GPU	CaloClouds	100	64	24.91 ± 0.72	×157
	CaloClouds II	25	64	6.12 ± 0.13	×640
	CaloClouds II (CM)	1	64	2.09 ± 0.13	×1873

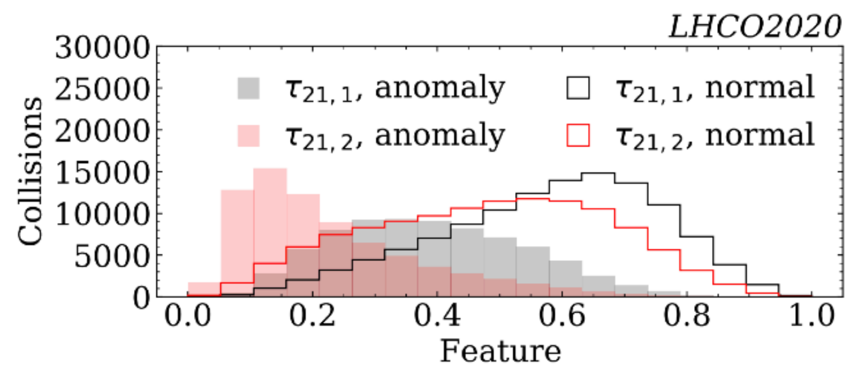


Use continuous time diffusion and consistency distillation: Better quality and faster

Buhmann, .., **GK** et al 2309.05704

Why generative models?

Showers in complex high-resolution calorimeters



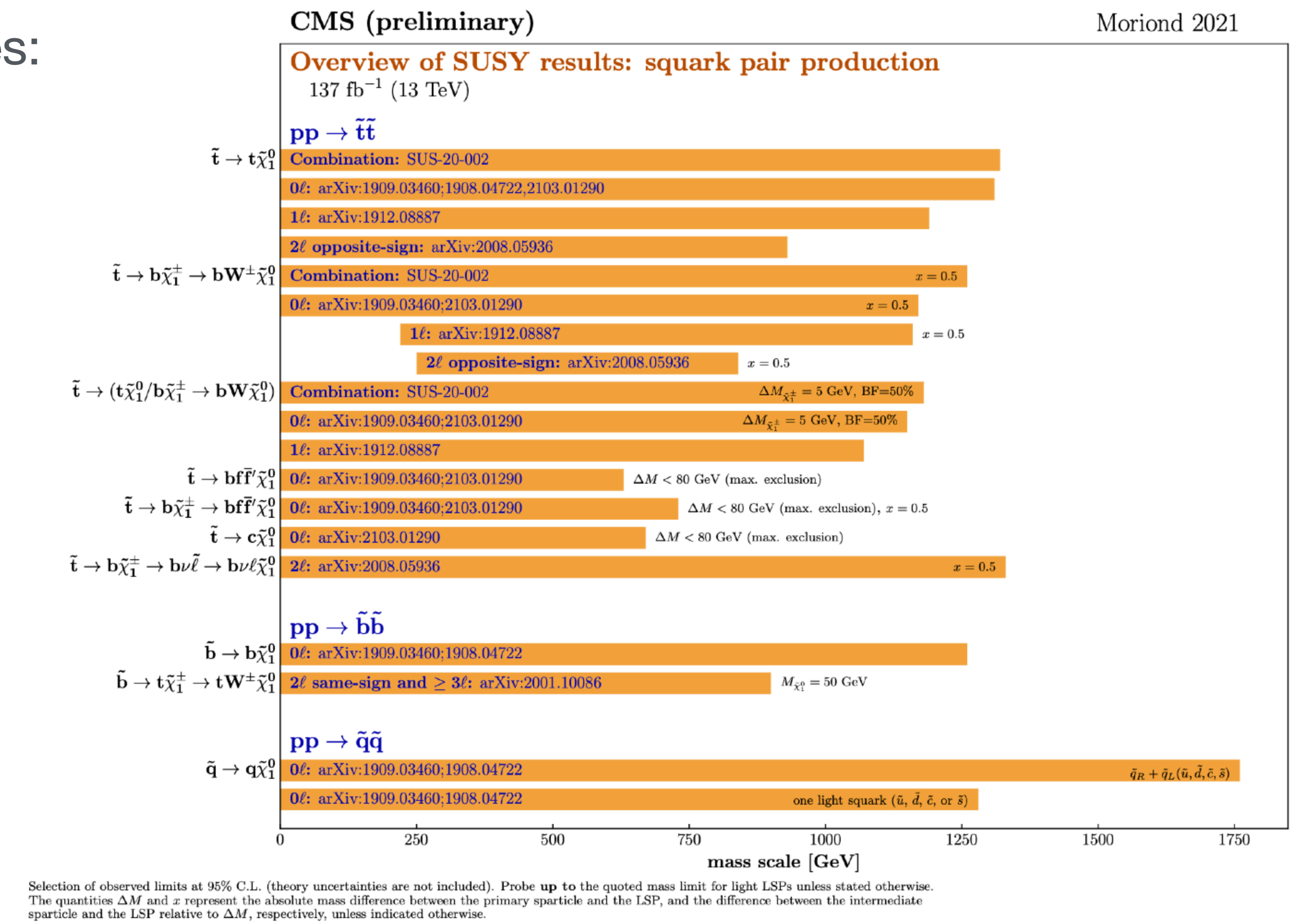
High-level jet features for background estimation

$$p(x)$$

Sample $X_i \sim p(x)$ to generate datapoints

Anomaly detections

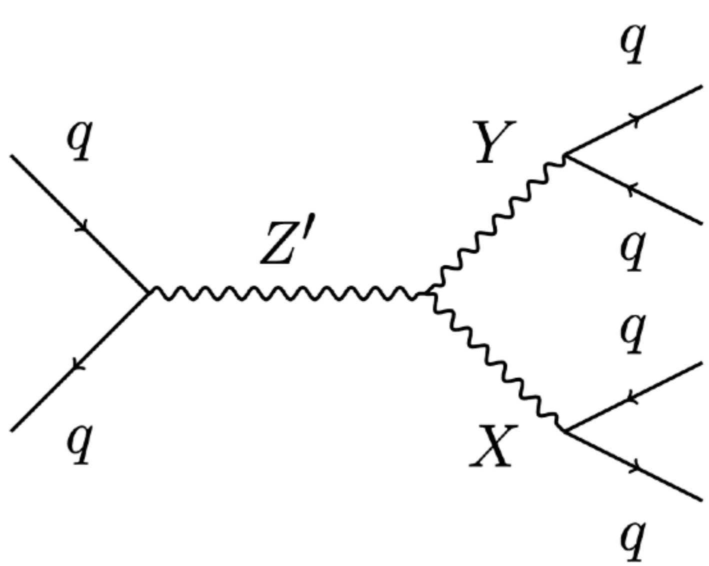
- Expect physics beyond the Standard Model
- Only negative results in searches
- Two discovery strategies:
  - Model-specific
  - Model independent



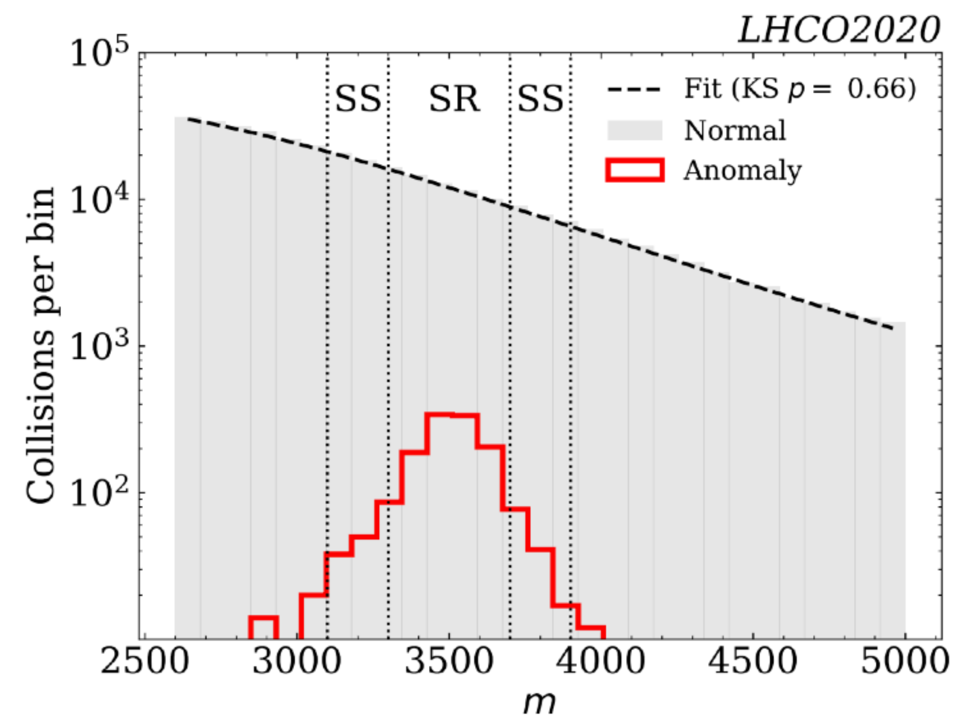
Dataset

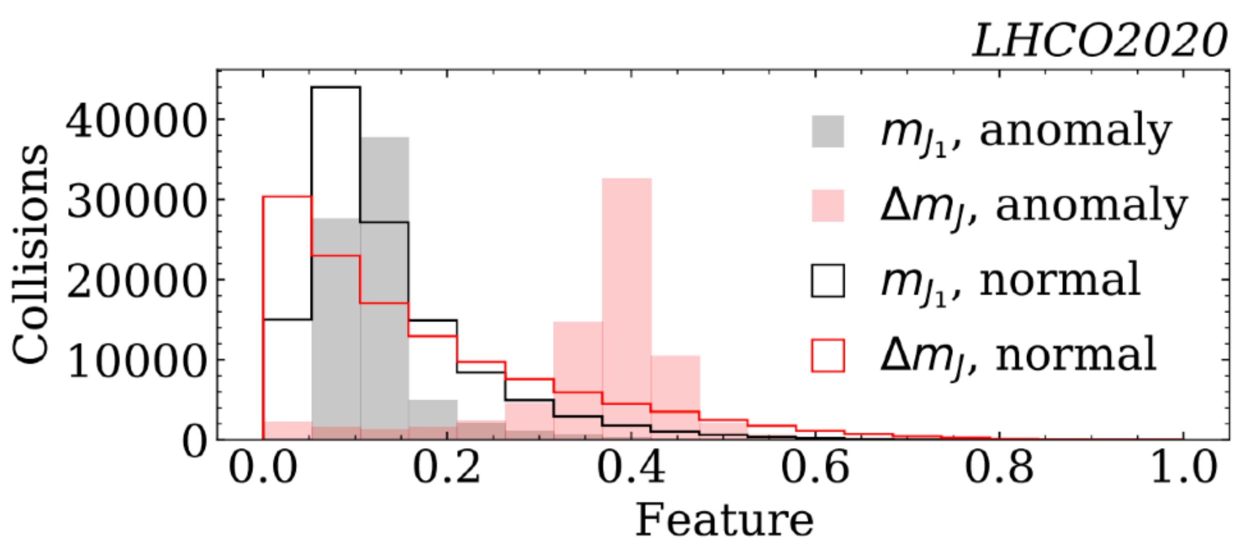
- LHC Olympics (LHCO): Community dataset for anomaly detection development

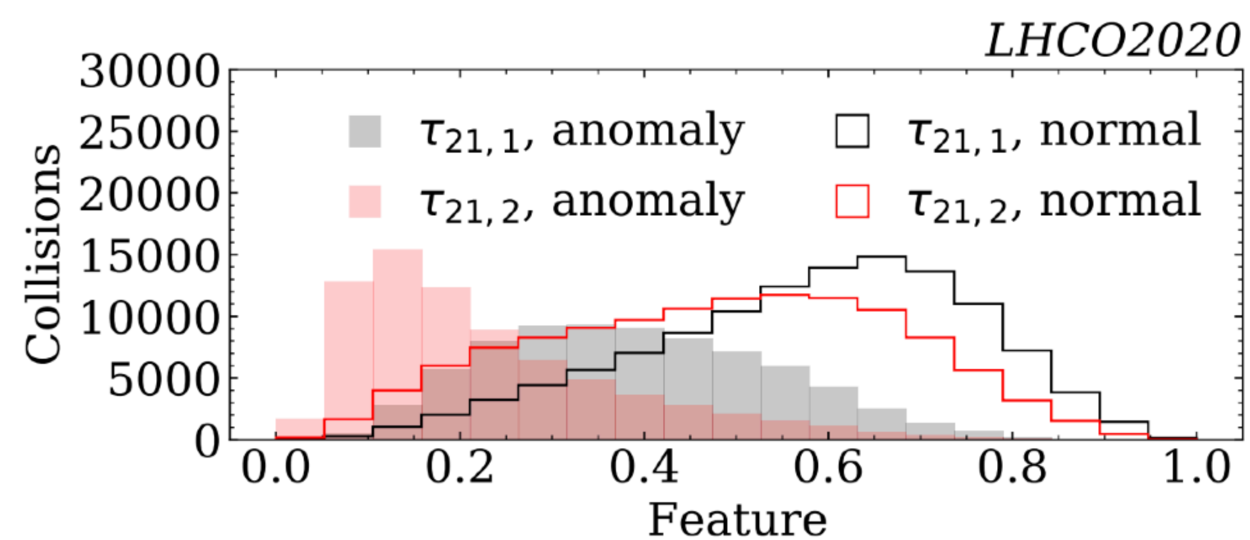
- Z' signal, QCD di-jet backgrounds



m(Z') = 3500 GeV
m(X) = 500 GeV
m(Y) = 100 GeV
(R&D dataset)

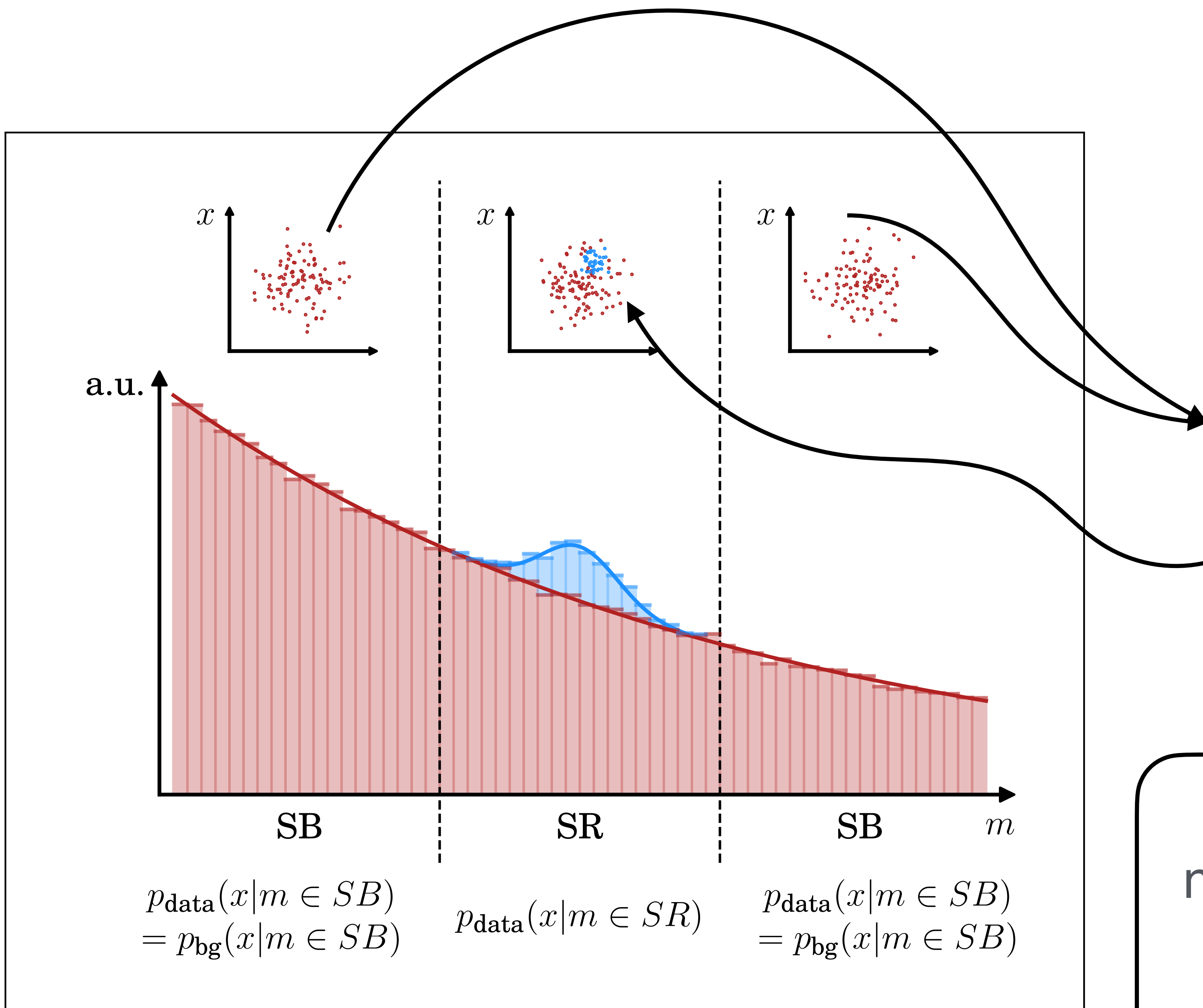






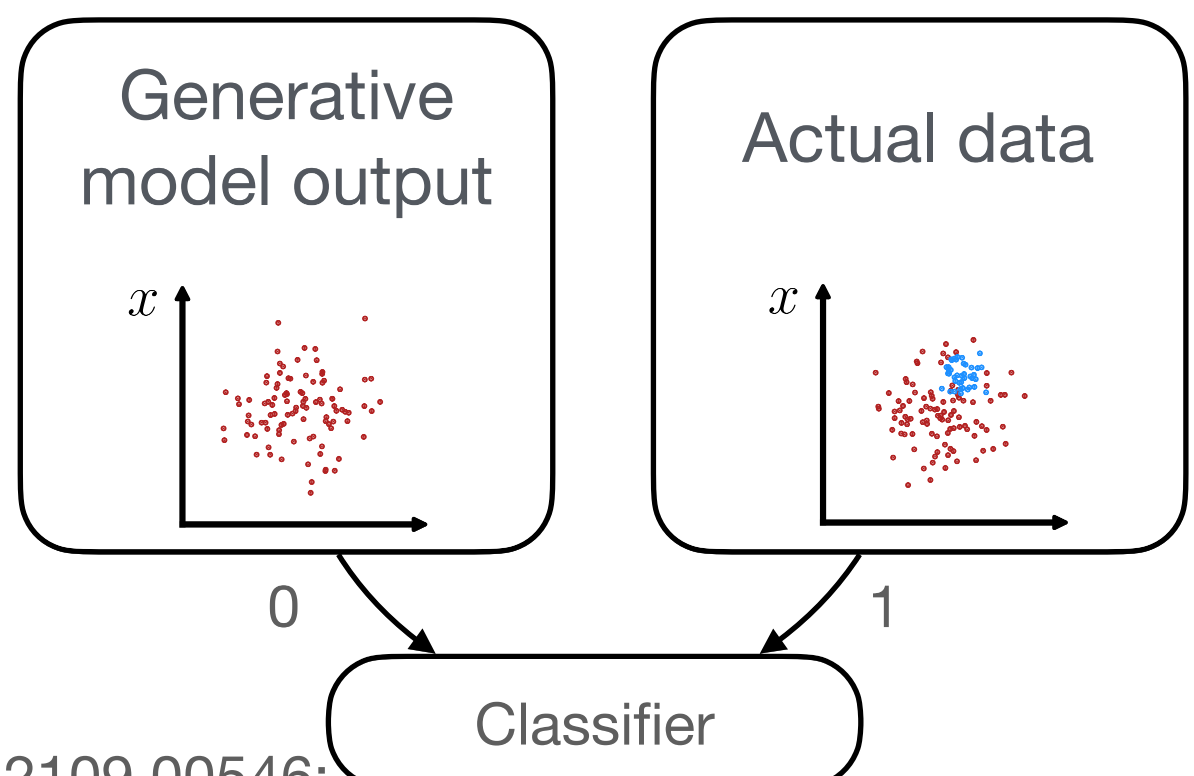
GK, Nachman, Shih, et al 2101.08320; **GK,** Nachman, Shih 2107.02821

CATHODE



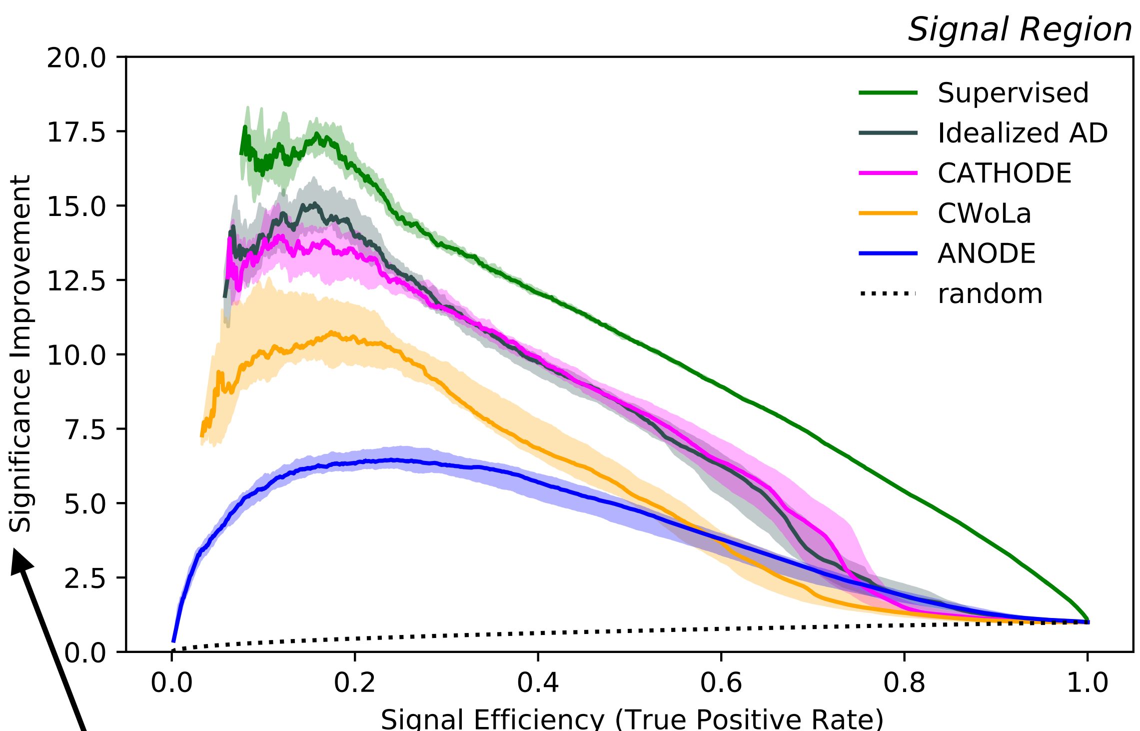
Method for resonant anomaly detection

1. Train generative model (conditional normalising flow)
2. Interpolate and sample
3. Use classifier to identify potential anomaly



GK, Nachmann, Shih et al 2101.08320; Hallin, .., **GK** et al 2109.00546;

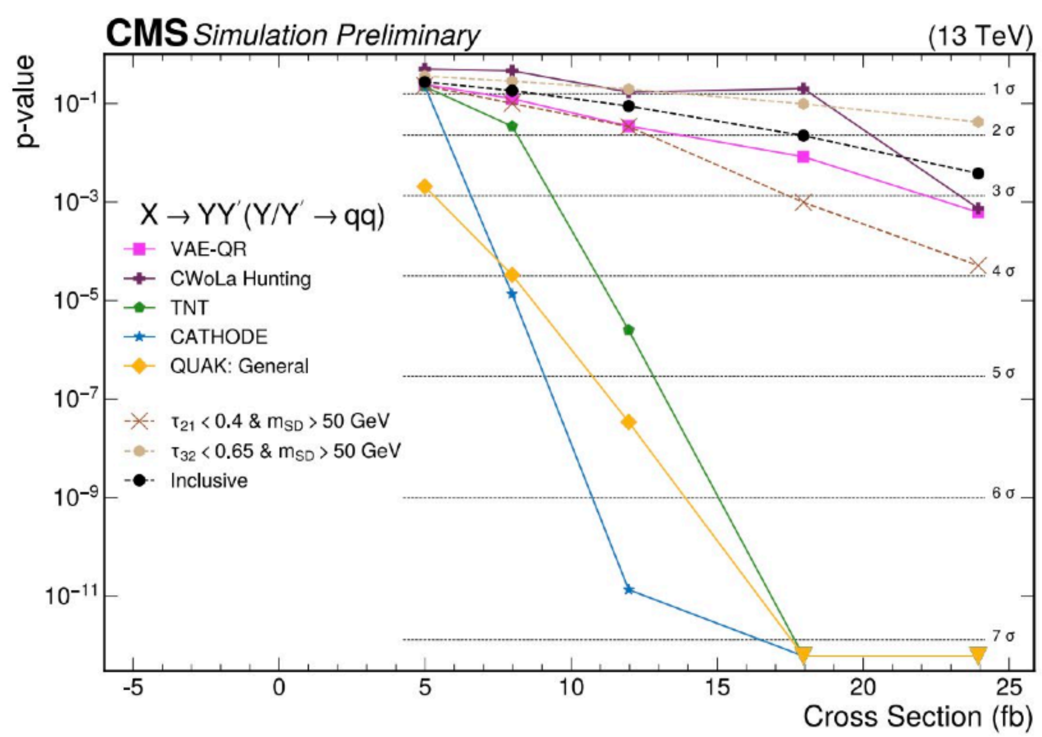
CATHODE



Method for resonant anomaly detection

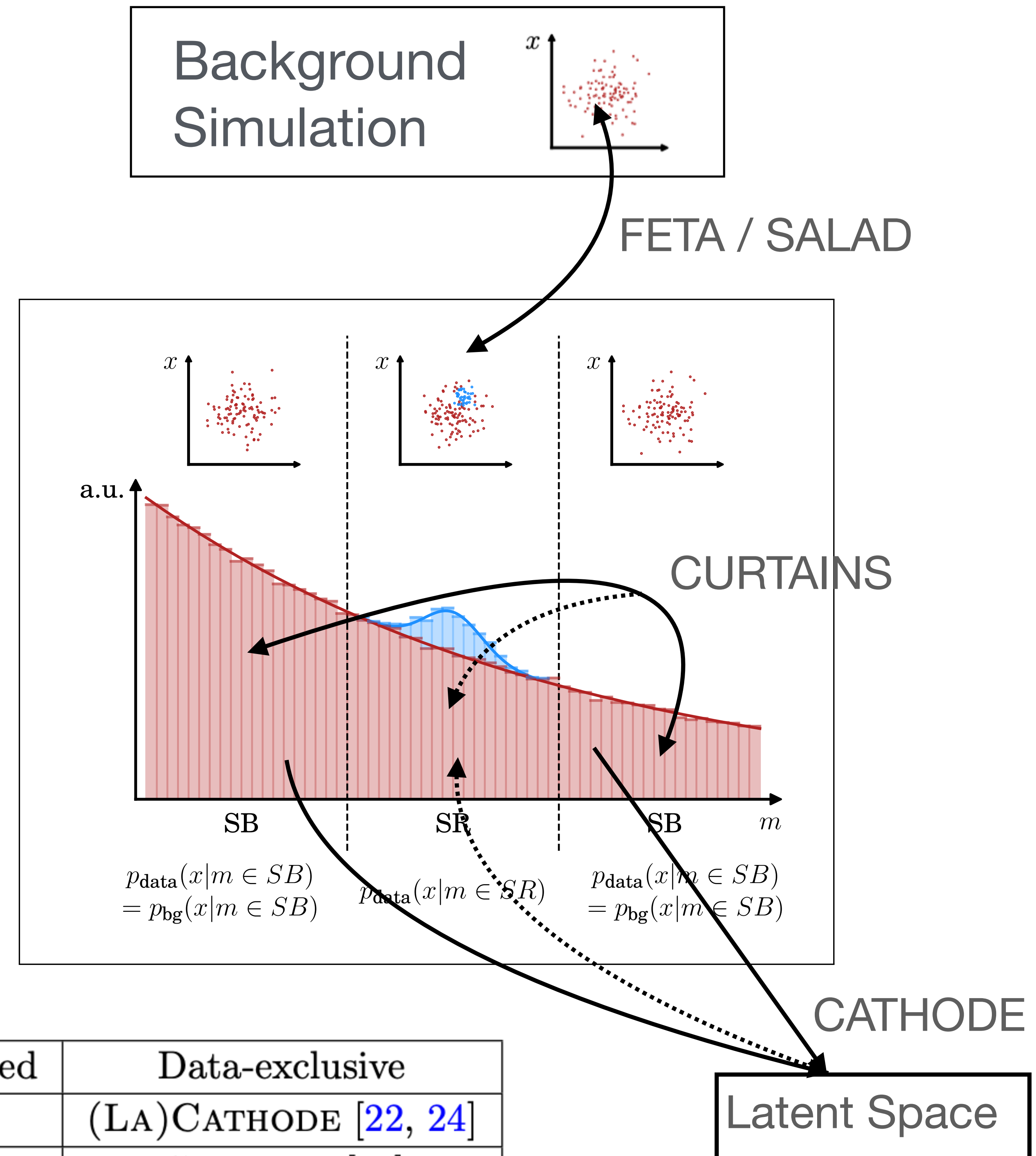
1. Train generative model (conditional normalising flow)
2. Interpolate and sample
3. Use classifier to identify potential anomaly

$$\mathrm{SIC} = \frac{\epsilon_S}{\sqrt{\epsilon_B}}$$



GK, Nachmann, Shih et al 2101.08320; Hallin, .., **GK** et al 2109.00546;
CMS Collaboration CMS-NOTE-2023-013

Alternatives



CATHODE: Conditional generative model interpolates into signal region

CURTAINS: Learns a conditioned morphing function for data

SALAD: Learns weights to for background simulation on the signal region

FETA: Learns a flow to morph background into the signal region

	Simulation-assisted	Data-exclusive
Likelihood learning	SALAD [17]	(LA)CATHODE [22, 24]
Feature morphing	FETA [26]	CURTAINS [27]

Golling, **GK**, .., Mastandrea et al 2307.11157

Alternatives

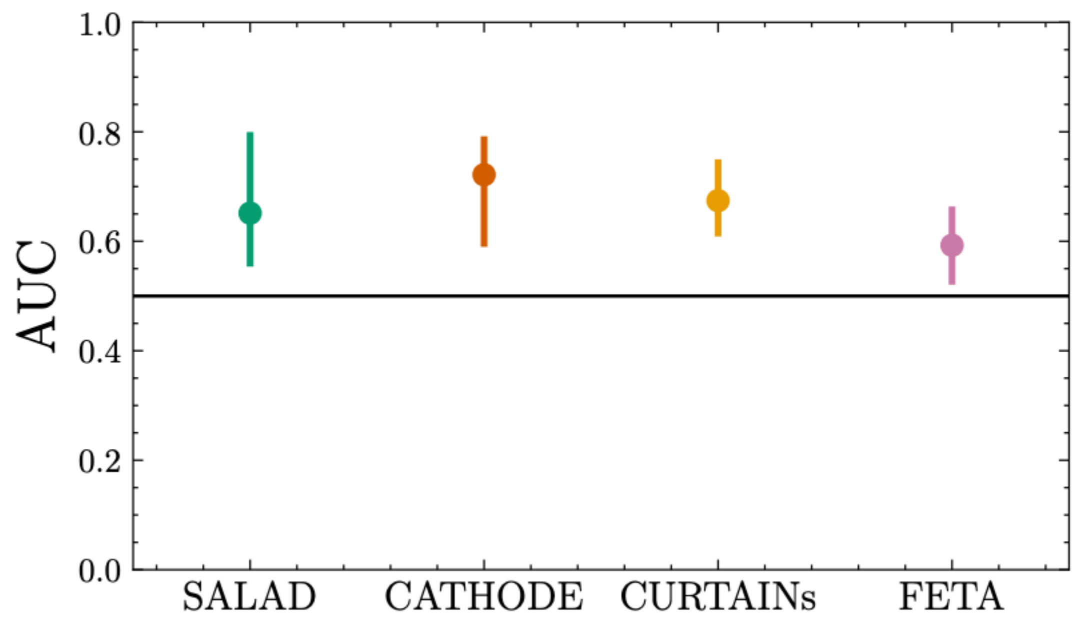
CATHODE: Conditional generative model interpolates into signal region

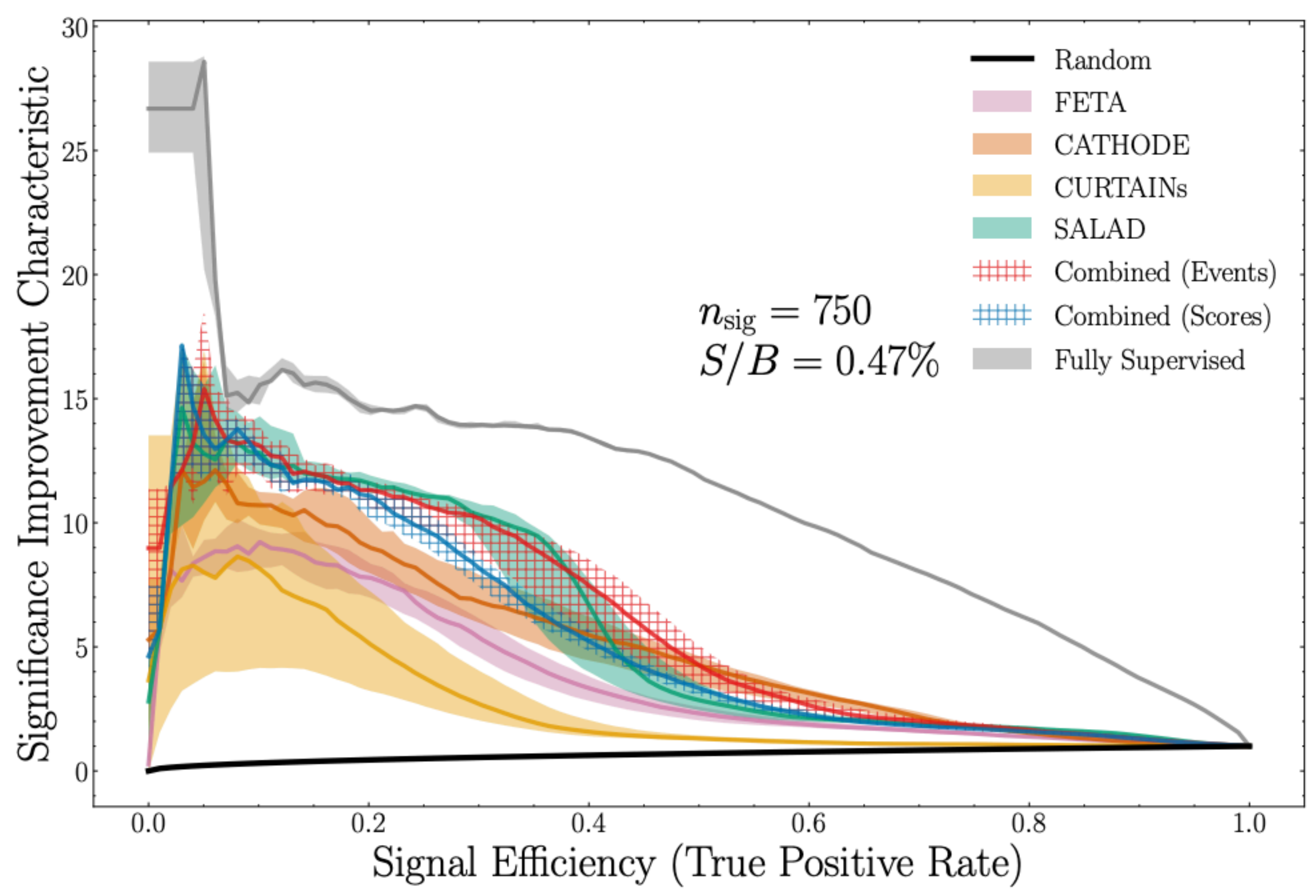
CURTAINS: Learns a conditioned morphing function for data

SALAD: Learns weights to for background simulation on the signal region

FETA: Learns a flow to morph background into the signal region

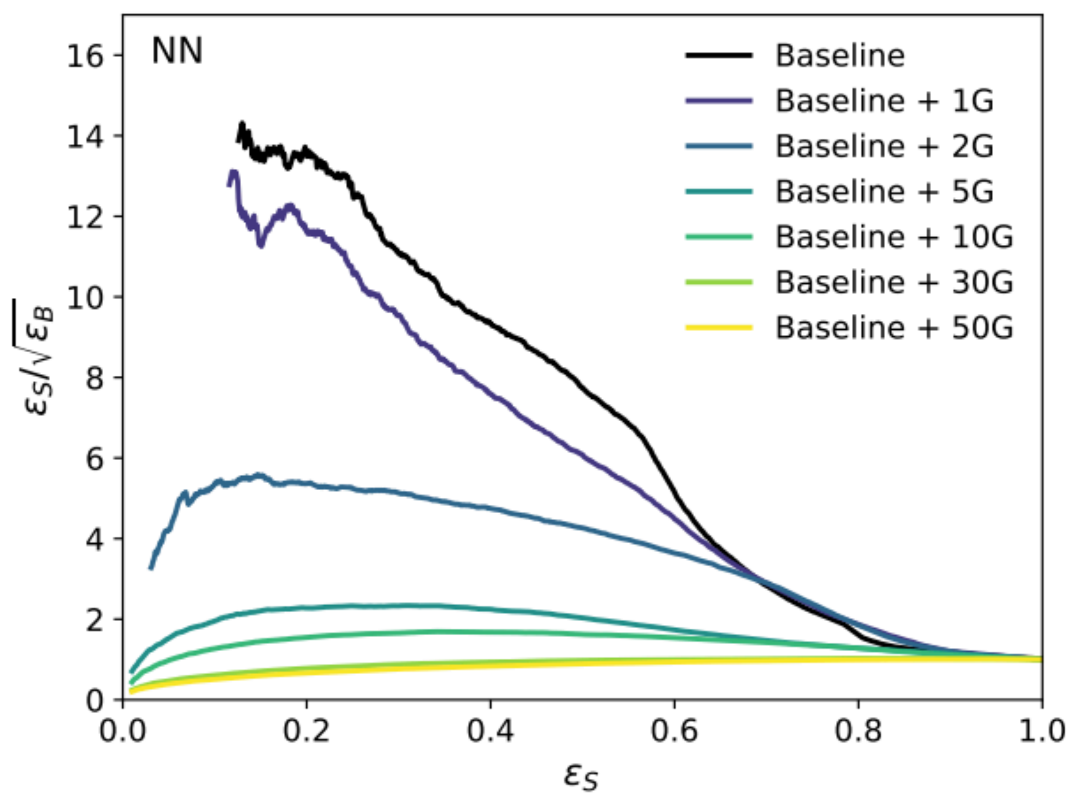
Similar performance, slight gain from combination

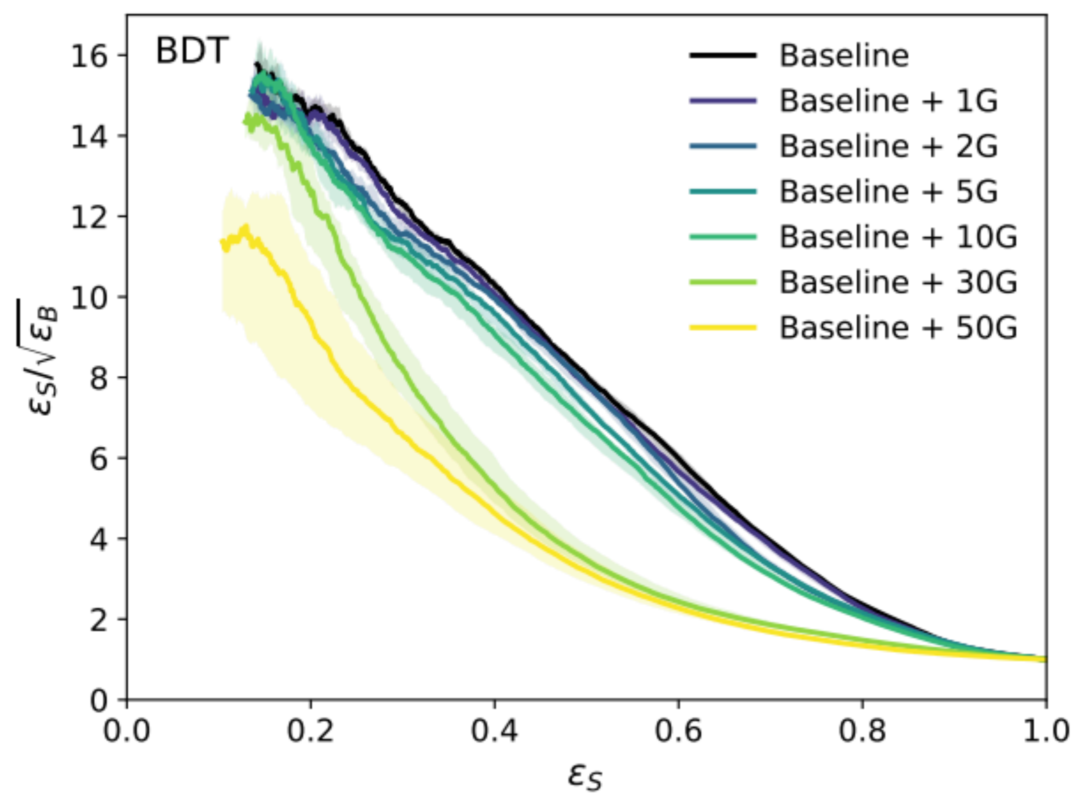




Golling, **GK**, .., Mastandrea et al 2307.11157

Side Note: Noisy Features





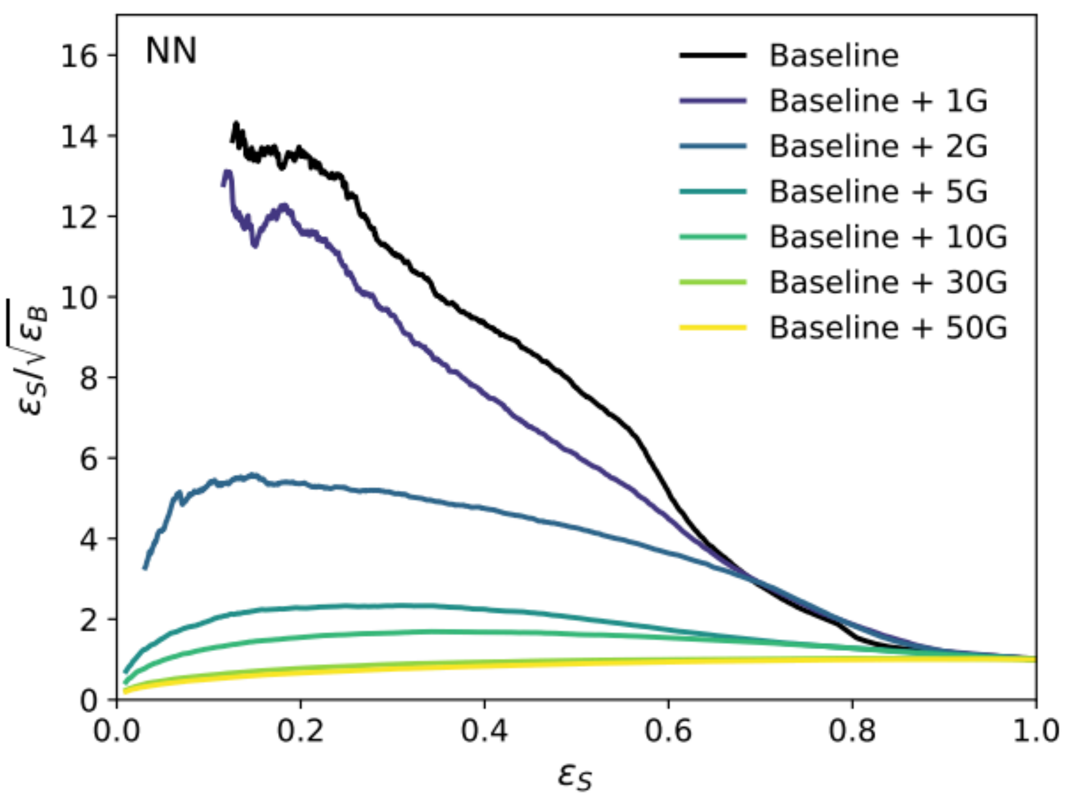
We don’t know in which feature is anomalous: Use more input features

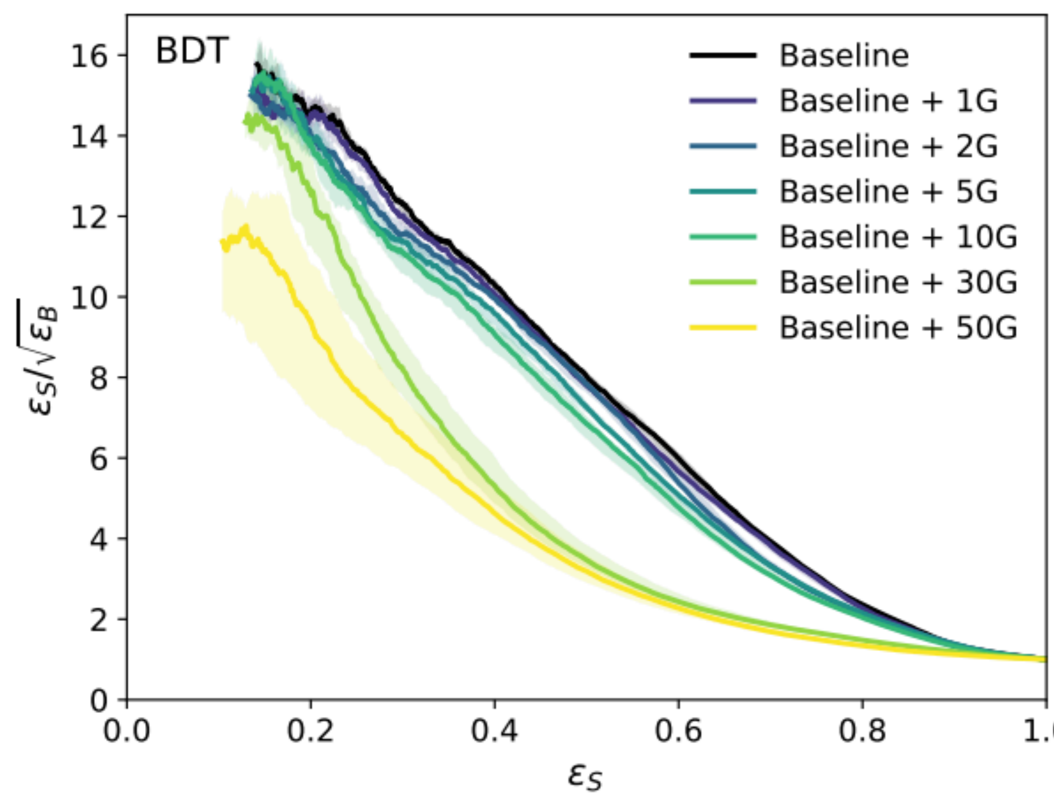
But: degrades performance

BDTs are more robust against noisy features

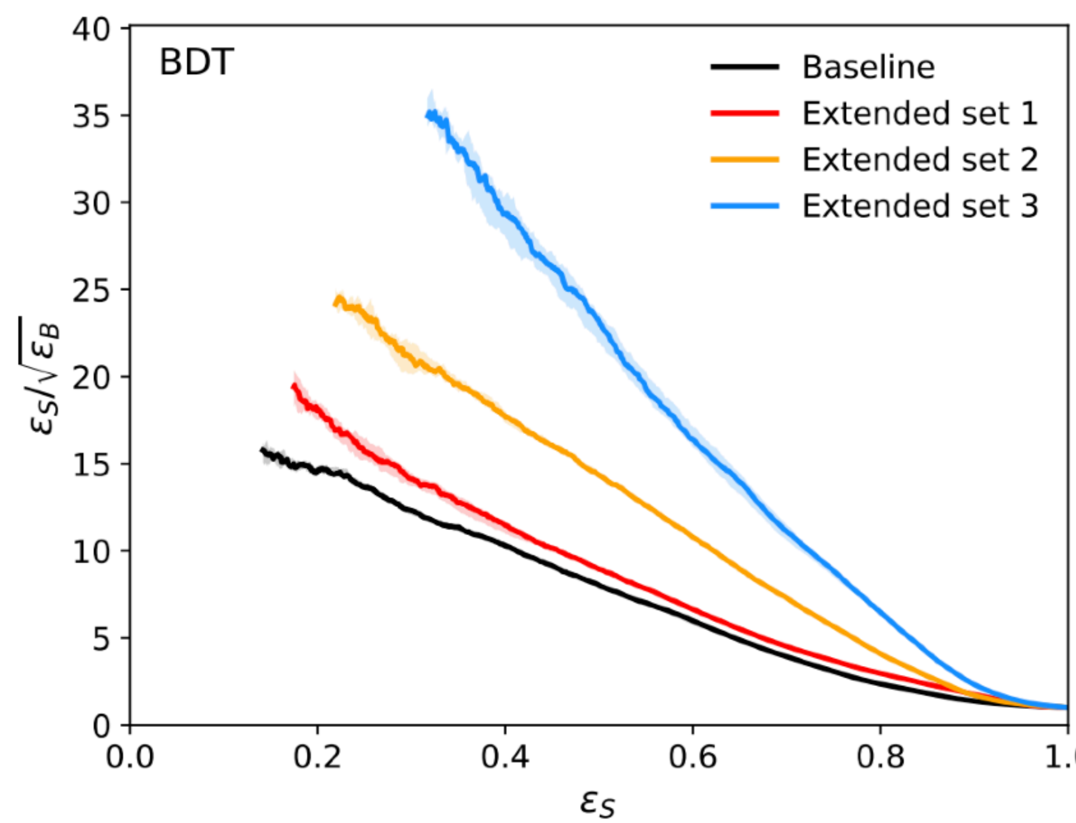
Finke, .., **GK** et al 2309.13111

Side Note: Noisy Features





Allows adding more inputs!

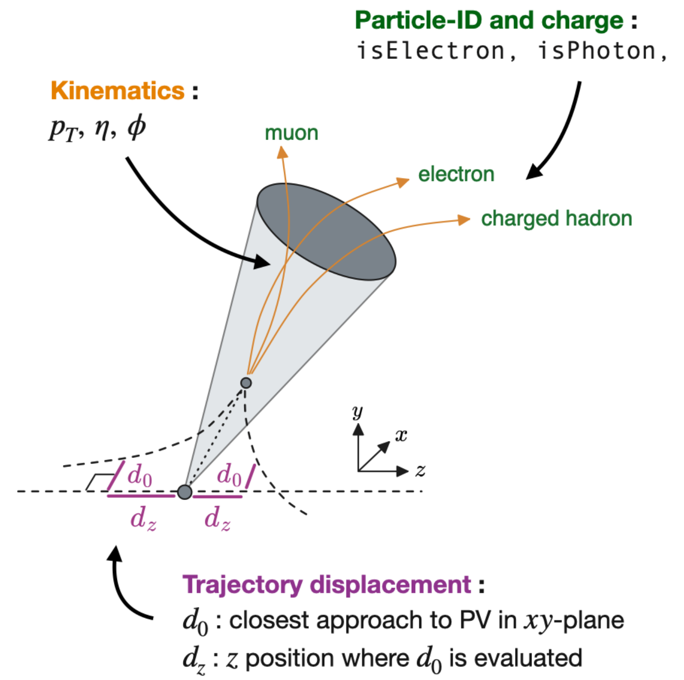


Finke, .., **GK** et al 2309.13111

Why generative models?

Showers in complex high-resolution calorimeters

High-level jet features



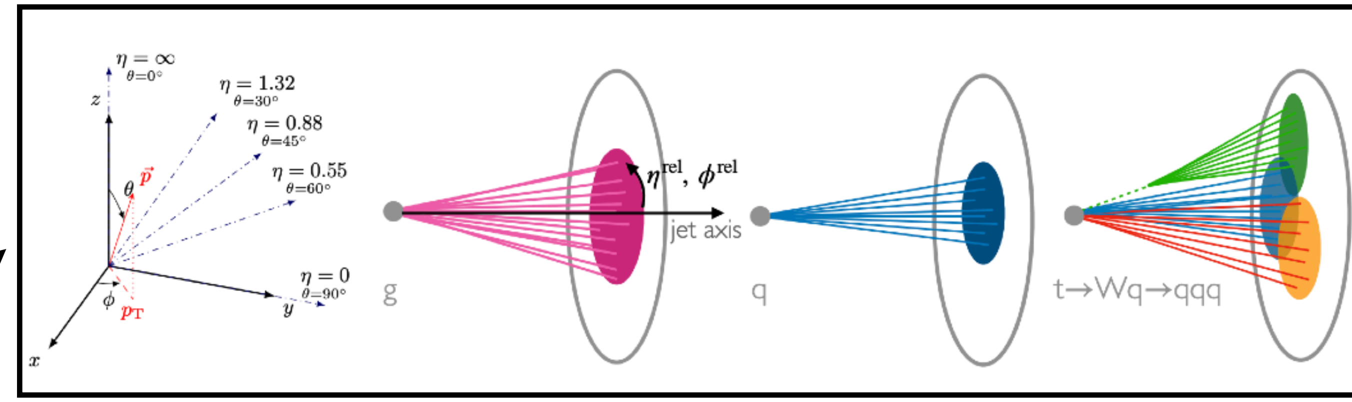
Jet constituents

$$p(x)$$

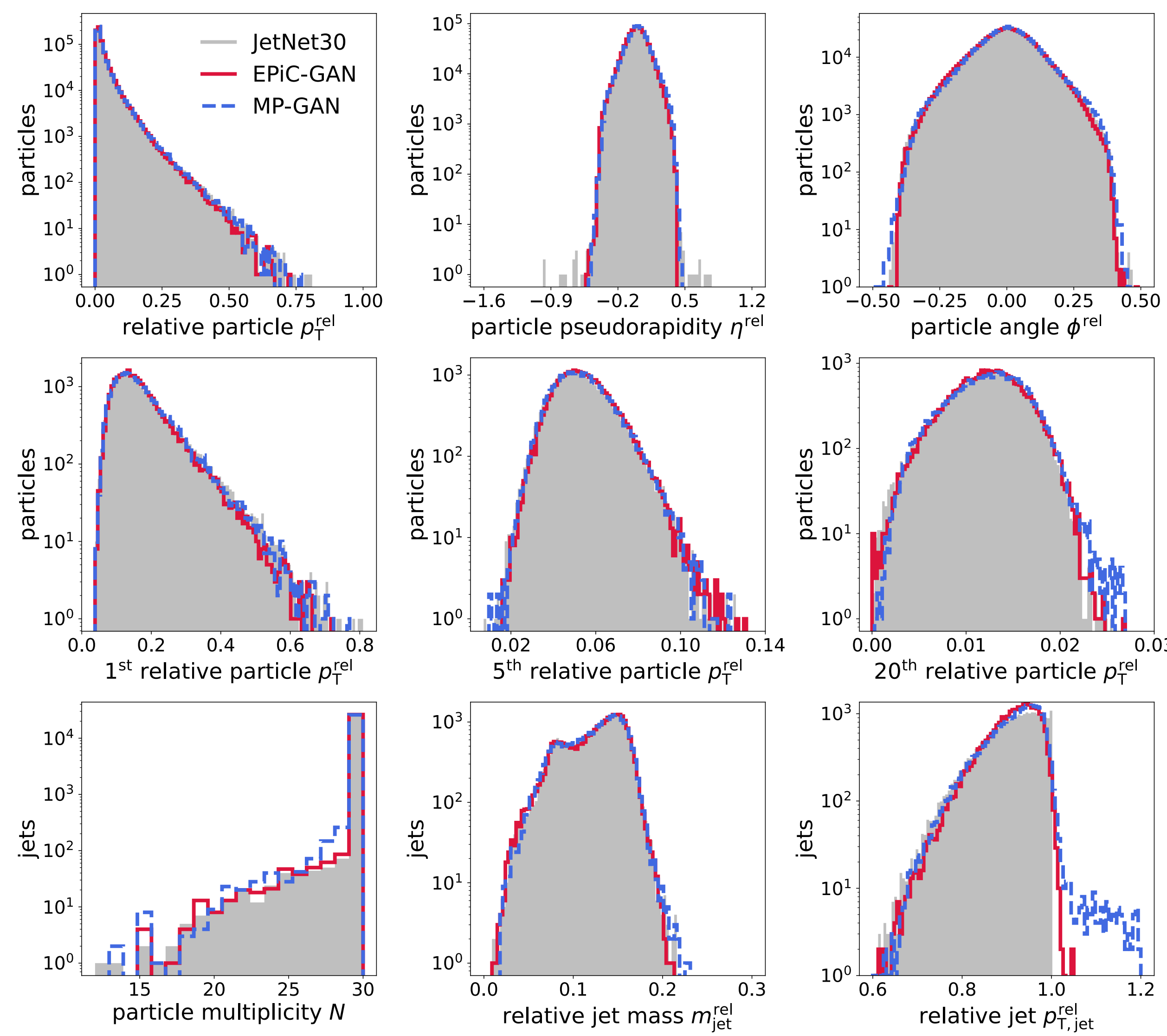
Sample $X_i \sim p(x)$
to generate datapoints

Simulation targets

Improve anomaly detection (and other background estimation tasks) by learning jet constituents instead of high-level features



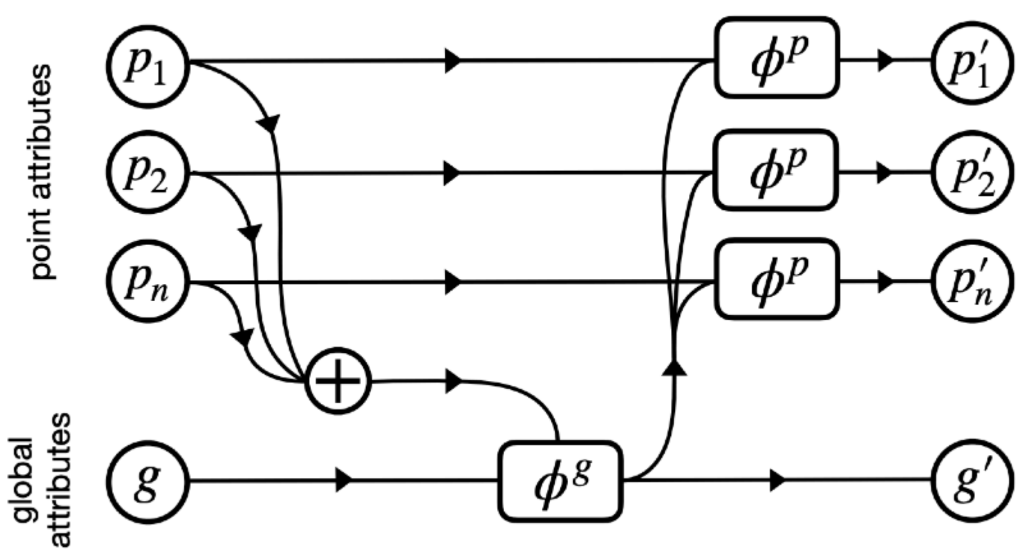
	JetNet [3]
Jet types	5 types
Dataset size	180 thousand jets per class
Features	Kinematics



Simulation targets

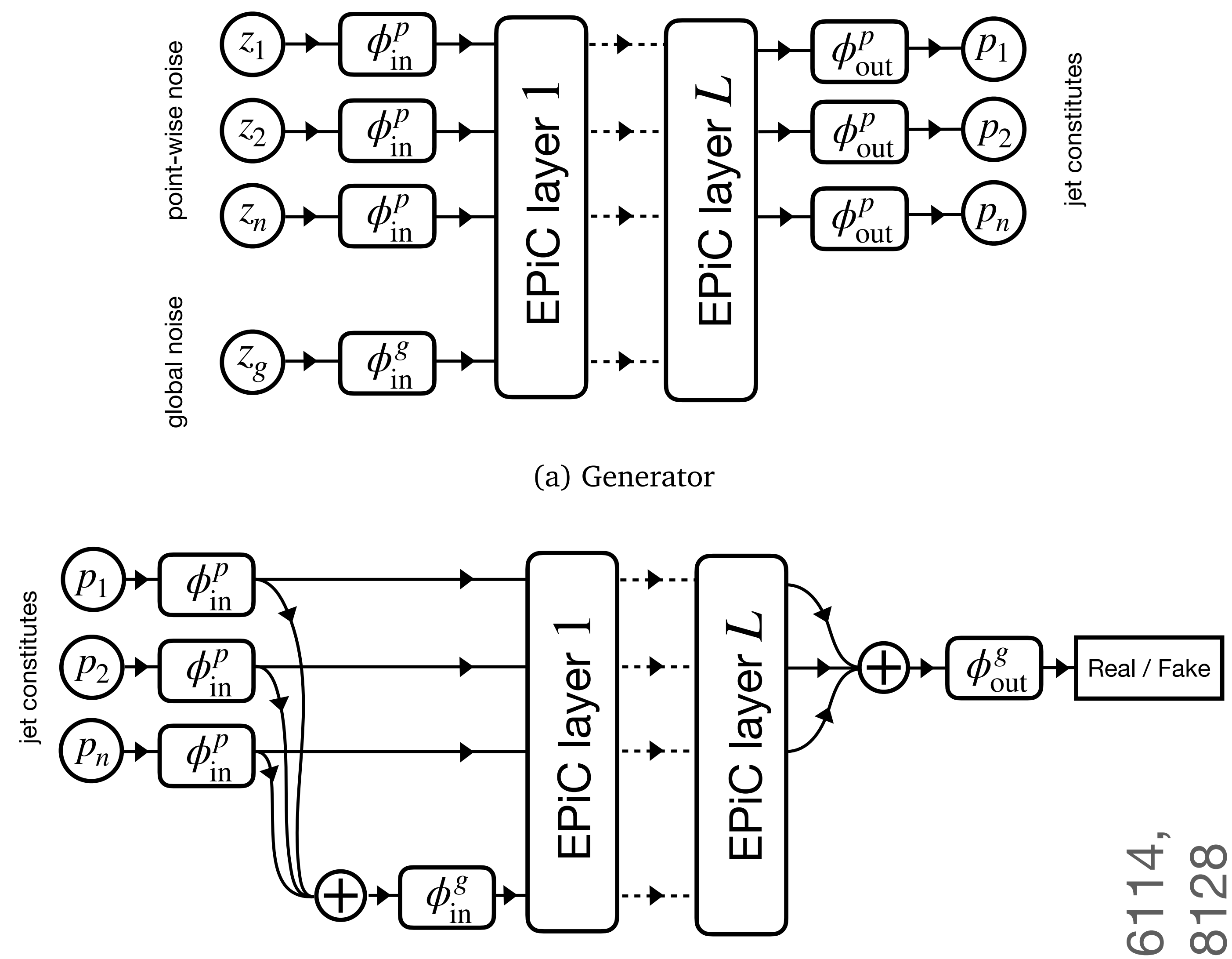
Improve anomaly detection (and other background estimation tasks) by learning jet constituents instead of high-level features.

	JetNet [3]
Jet types	5 types
Dataset size	180 thousand jets per class
Features	Kinematics



EPiC Layer

Treat as point cloud & use a permutation invariant GAN



(a) Generator

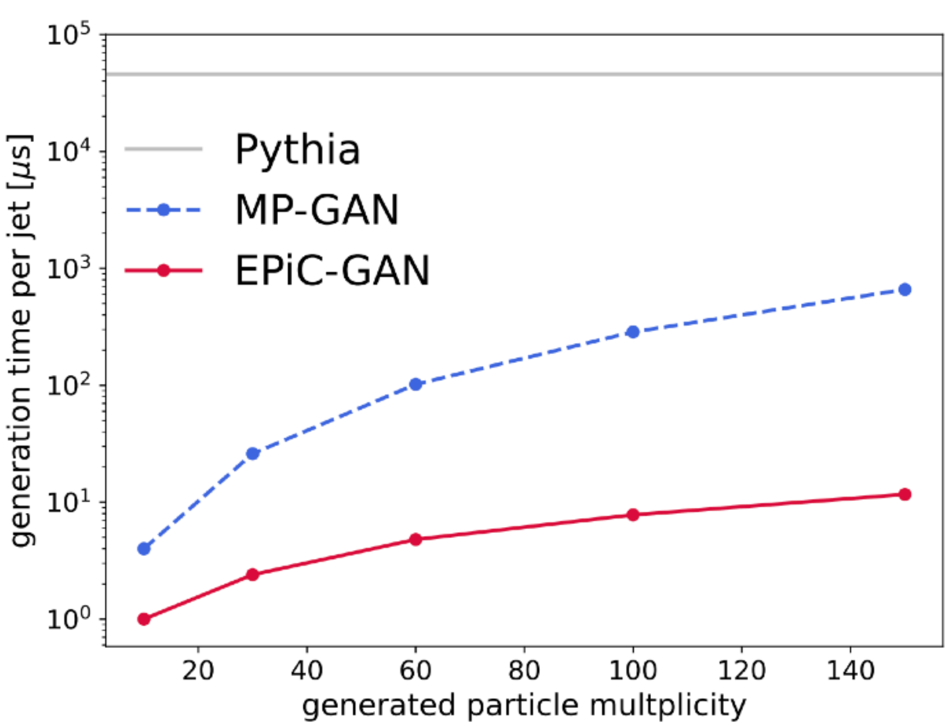
(b) Discriminator

1703.06114, 2301.08128

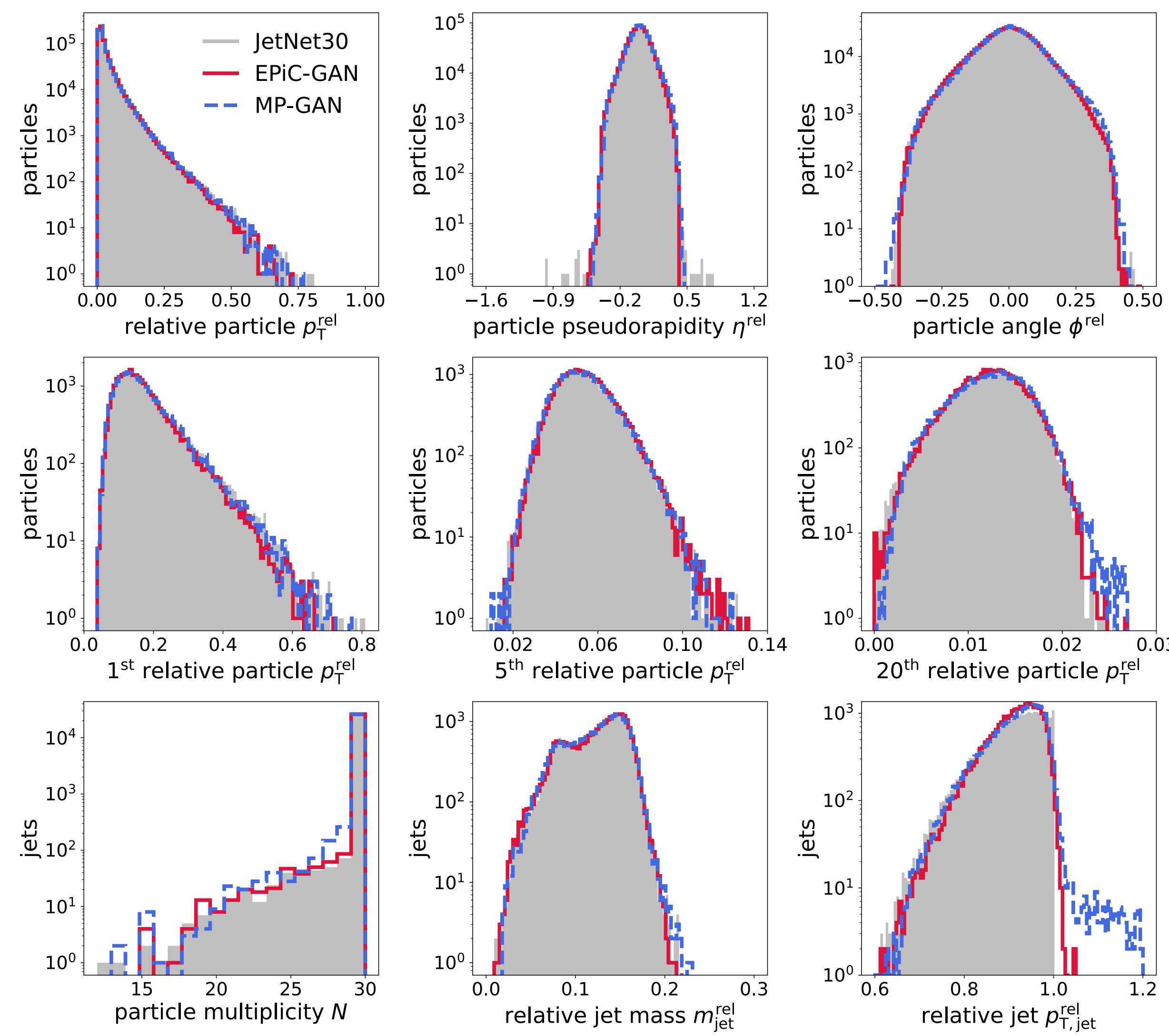
Simulation targets

Improve anomaly detection (and other background estimation tasks) by learning jet constituents instead of high-level features.

	JetNet [3]
Jet types	5 types
Dataset size	180 thousand jets per class
Features	Kinematics



Treat as point cloud & use a permutation invariant GAN



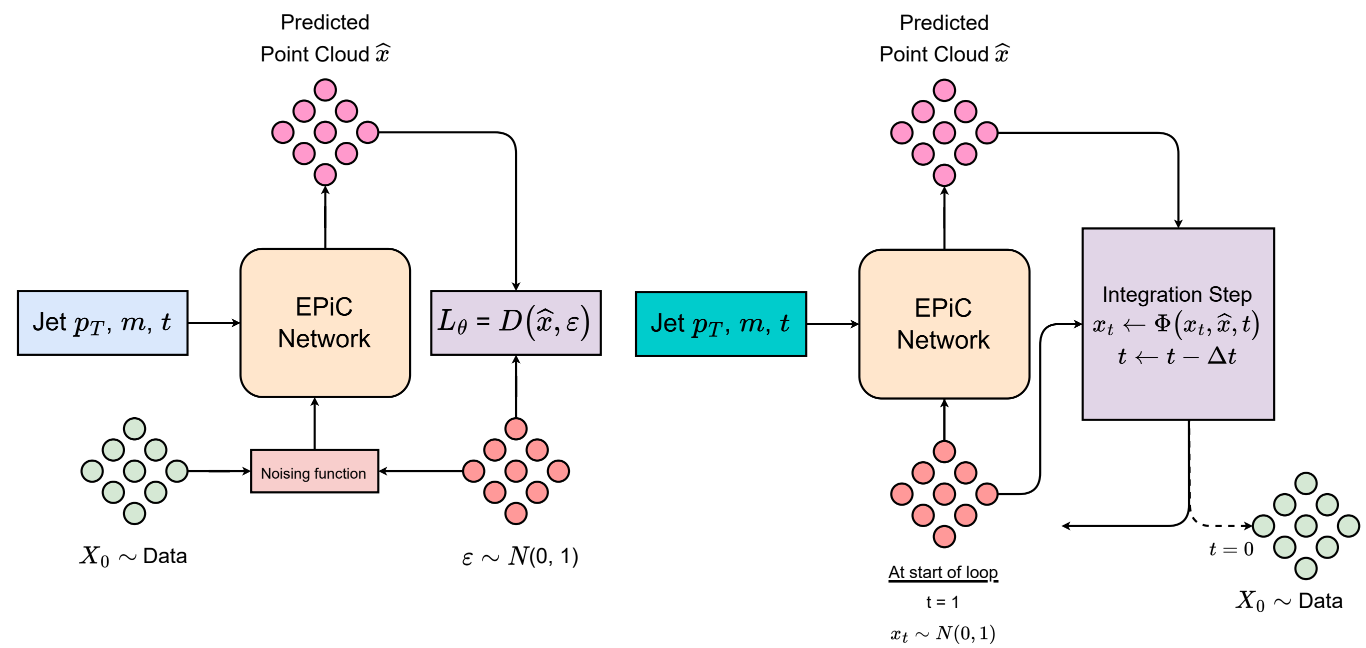
1703.06114, 2301.08128

Simulation targets

Improve anomaly detection (and other background estimation tasks) by learning jet constituents instead of high-level features.

Again, improve by moving from GAN to diffusion/flow matching

	JetNet [3]
Jet types	5 types
Dataset size	180 thousand jets per class
Features	Kinematics



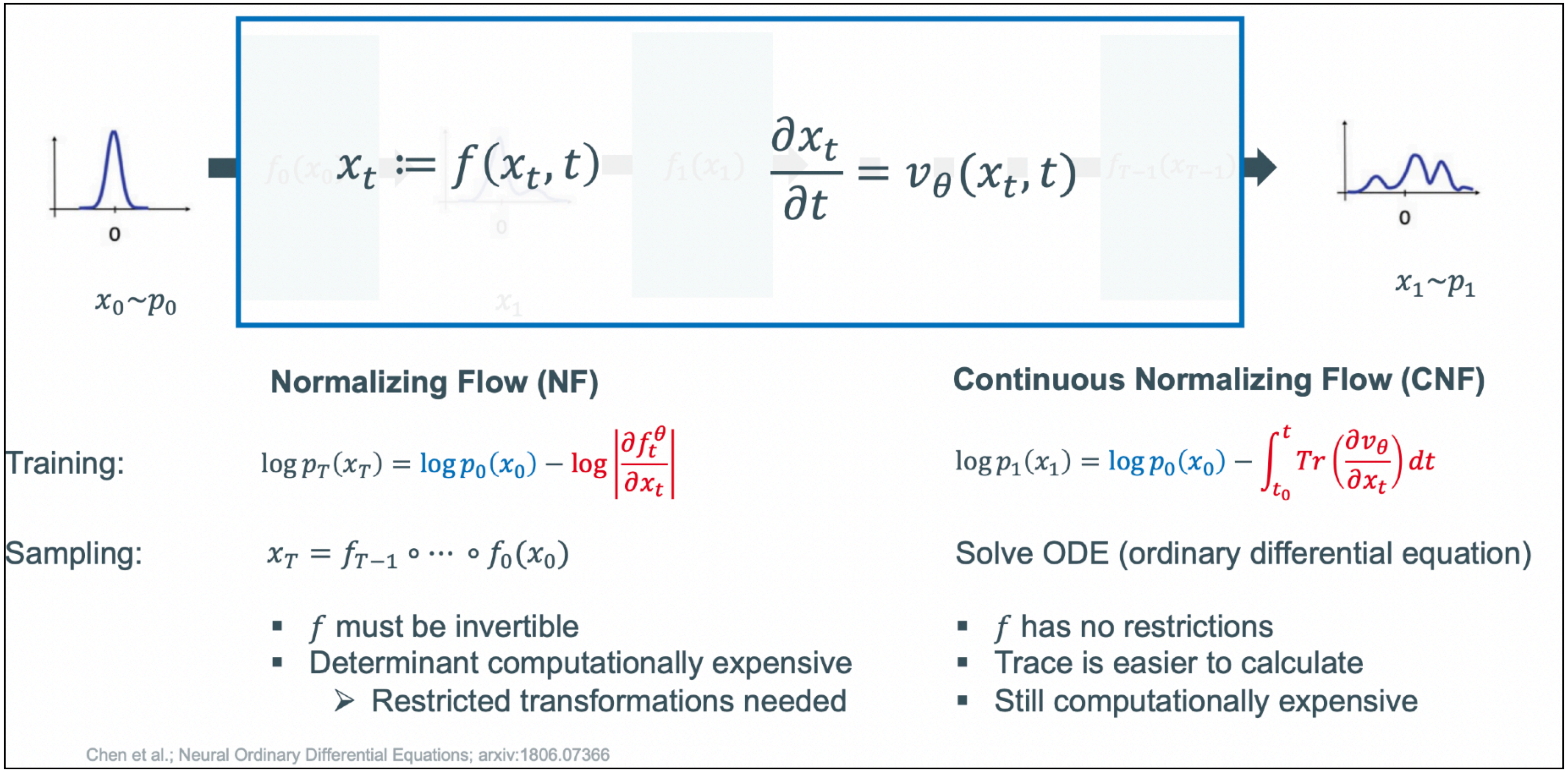
Training

Generation

Generation	Model	FPND	NLP	$\mathrm{KL}^{m}(\times 10^{-3})$	$\mathrm{KL}^{p_T^{\mathrm{const}}}(\times 10^{-3})$	$\mathrm{KL}^{\tau_{21}}(\times 10^{-3})$	$\mathrm{KL}^{\tau_{32}}(\times 10^{-3})$	$\mathrm{KL}^{D_2}(\times 10^{-3})$
Conditional	PC-JeDi	0.40	3.08	8.56 ± 0.75	3.25 ± 0.09	12.82 ± 1.16	27.08 ± 1.40	11.91 ± 0.92
	EPiC-JeDi	0.42	3.1	5.26 ± 0.51	2.99 ± 0.05	$\mathbf{7.81 \pm 0.61}$	17.34 ± 1.08	6.58 ± 0.73
	EPiC-FM	**0.11**	**1.35**	$\mathbf{3.77 \pm 0.50}$	$\mathbf{2.03 \pm 0.02}$	$\mathbf{7.40 \pm 0.64}$	$\mathbf{8.09 \pm 0.93}$	$\mathbf{4.31 \pm 0.46}$
Unconditional	EPiC-GAN	0.34	3.43	$\mathbf{3.71 \pm 0.42}$	3.33 ± 0.03	$\mathbf{8.28 \pm 0.76}$	17.68 ± 0.91	13.18 ± 1.04
	EPiC-JeDi	1.63	3.11	18.42 ± 1.12	3.73 ± 0.08	$\mathbf{8.00 \pm 0.80}$	15.27 ± 1.35	12.33 ± 1.06
	EPiC-FM	0.14	1.38	5.80 ± 0.54	$\mathbf{2.03 \pm 0.01}$	$\mathbf{7.69 \pm 0.71}$	$\mathbf{9.24 \pm 1.00}$	$\mathbf{4.51 \pm 0.58}$

2310.00049

Aside: Flow Matching

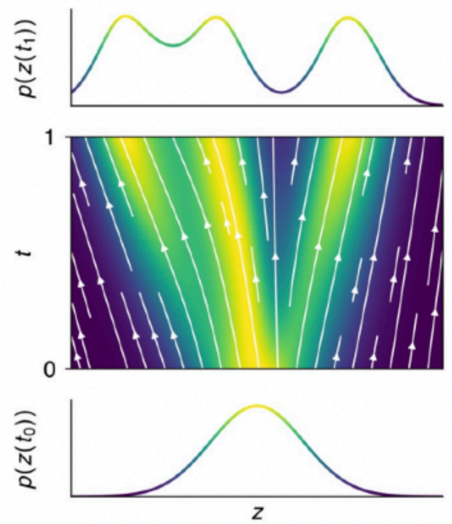


Formally similar to diffusion models

(Material by Cedric Ewen)

2310.00049

Aside: Flow Matching

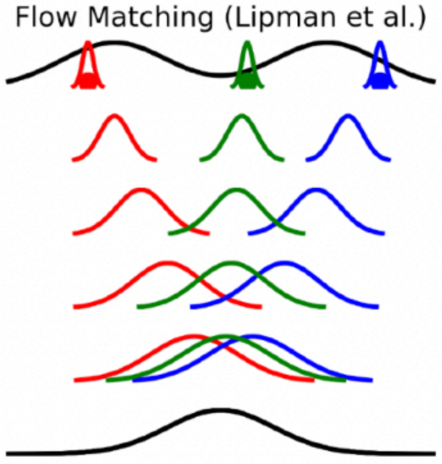


[1810.01367]

Continuous Normalizing Flow (CNF)

$$\log p_1(x_1) = \log p_0(x_0) - \int_{t_0}^{t} Tr\left(\frac{\partial v_\theta}{\partial x_t}\right) dt$$

$$L_{FM} = \left|\left|v_\theta(x_t) - u_t(x_t|x_0)\right|\right|^2$$



$$x_t = \gamma_t x_0 + \sigma_t \epsilon$$

[2302.00482]

Flow Matching (FM)

Training:
- Training is difficult because ODE needs to be solved

$$\frac{\partial x_t}{\partial t} = v_\theta(x_t, t)$$

Training:
- Simulation-free training objective (no ODE solving during training)
- Regressing against conditional flows

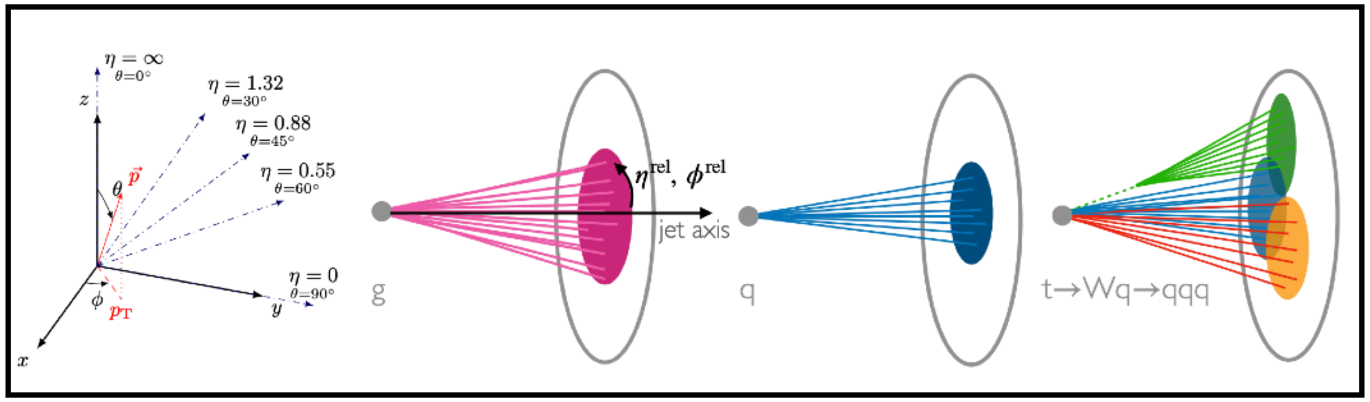
➢ Much faster training

(Material by Cedric Ewen)

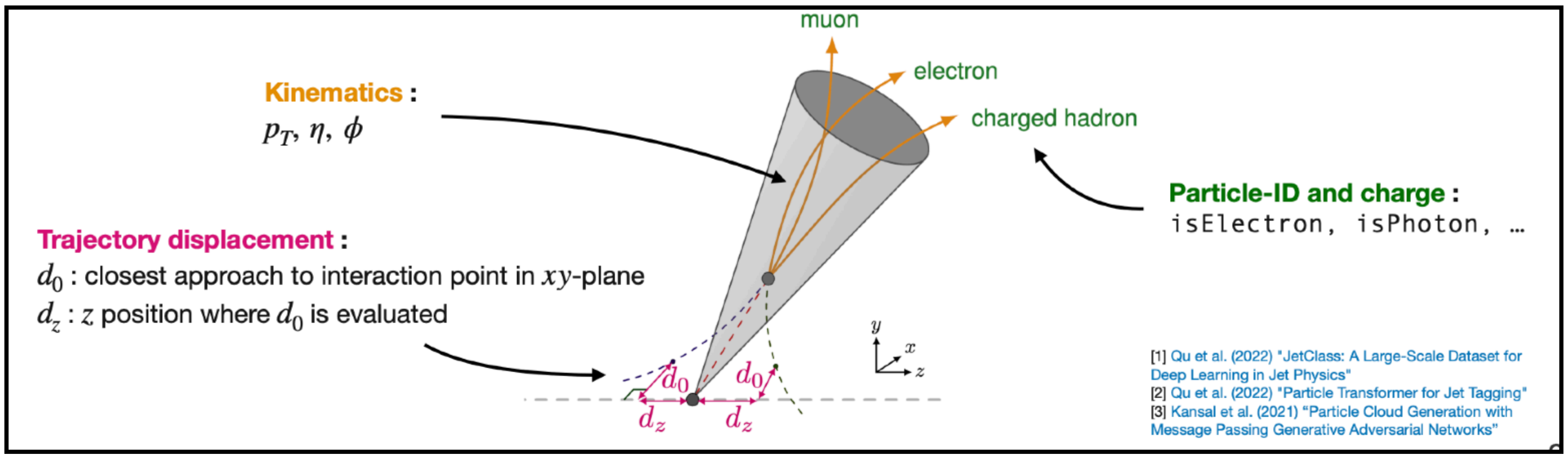
2310.00049

Simulation targets

Improve anomaly detection (and other background estimation tasks) by learning jet constituents instead of high-level features

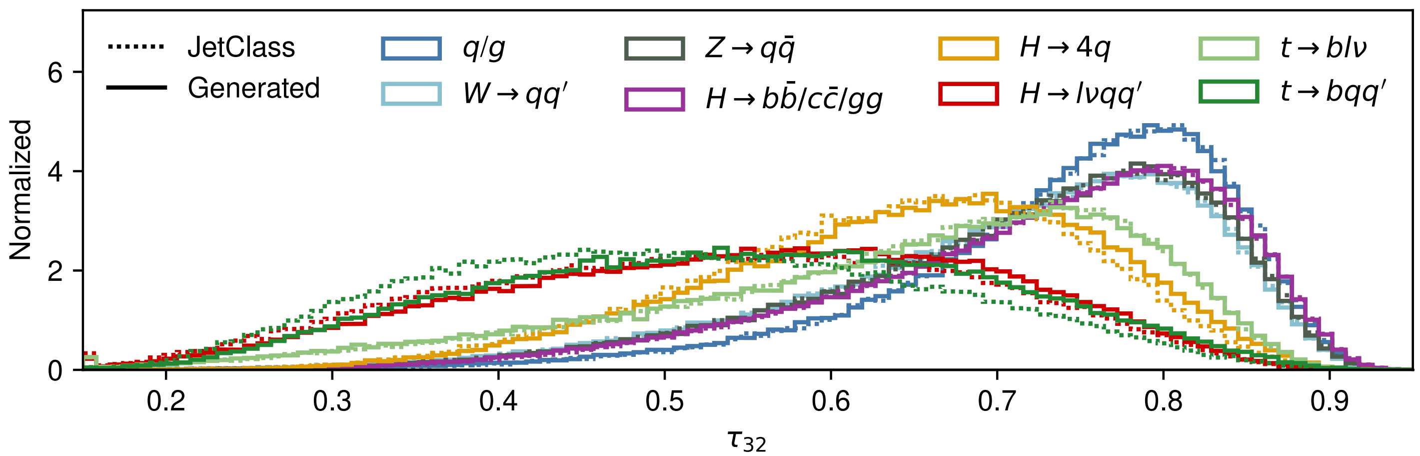


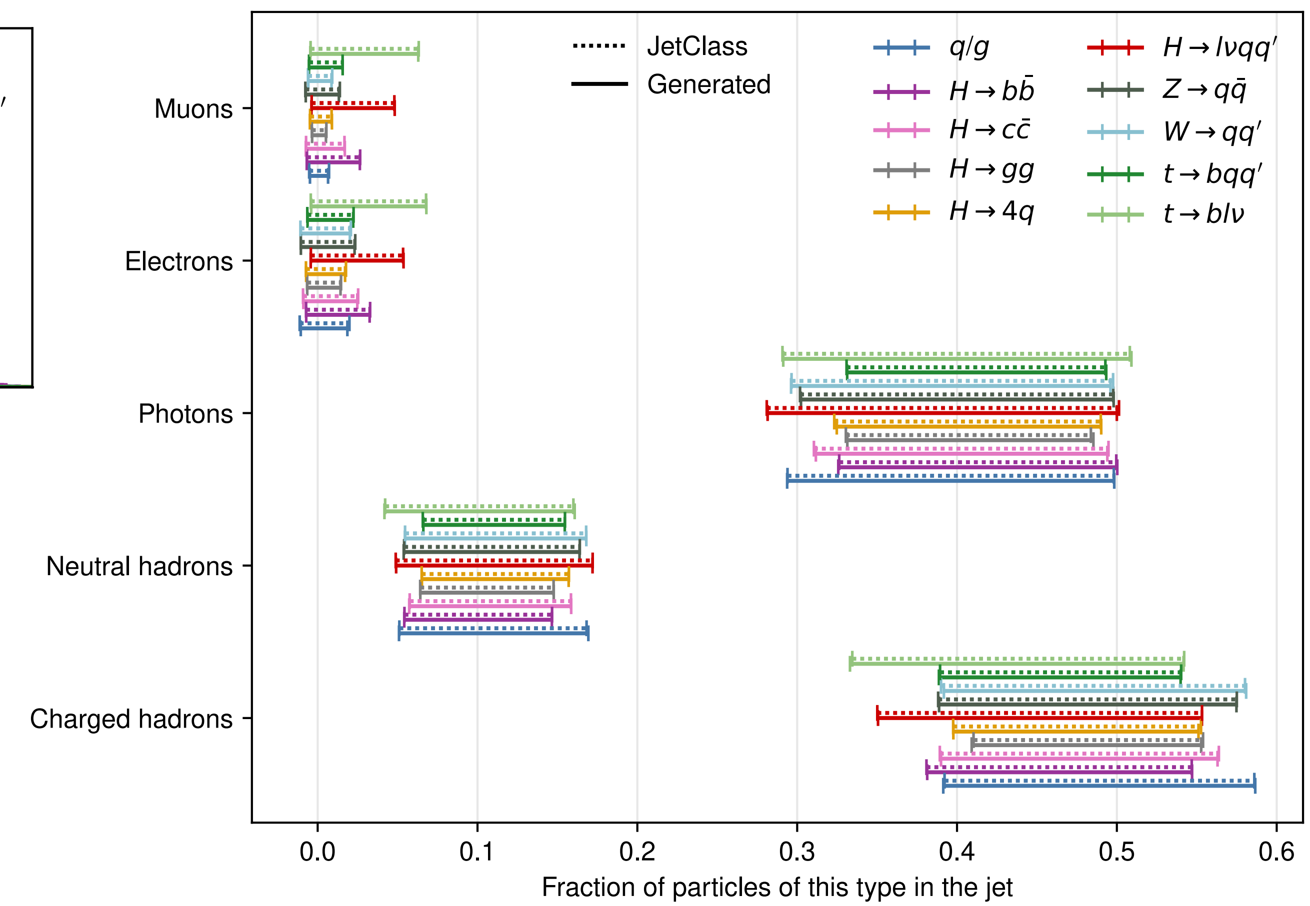
	JetNet [3]	JetClass [1]
Jet types	5 types	10 types (several decay channels for top and H jets)
Dataset size	180 thousand jets per class	12.5 million jets per class (70x more than JetNet)
Features	Kinematics	Kinematics, Particle-ID and charge, trajectory displacement



[1] Qu et al. (2022) "JetClass: A Large-Scale Dataset for Deep Learning in Jet Physics"
[2] Qu et al. (2022) "Particle Transformer for Jet Tagging"
[3] Kansal et al. (2021) "Particle Cloud Generation with Message Passing Generative Adversarial Networks"

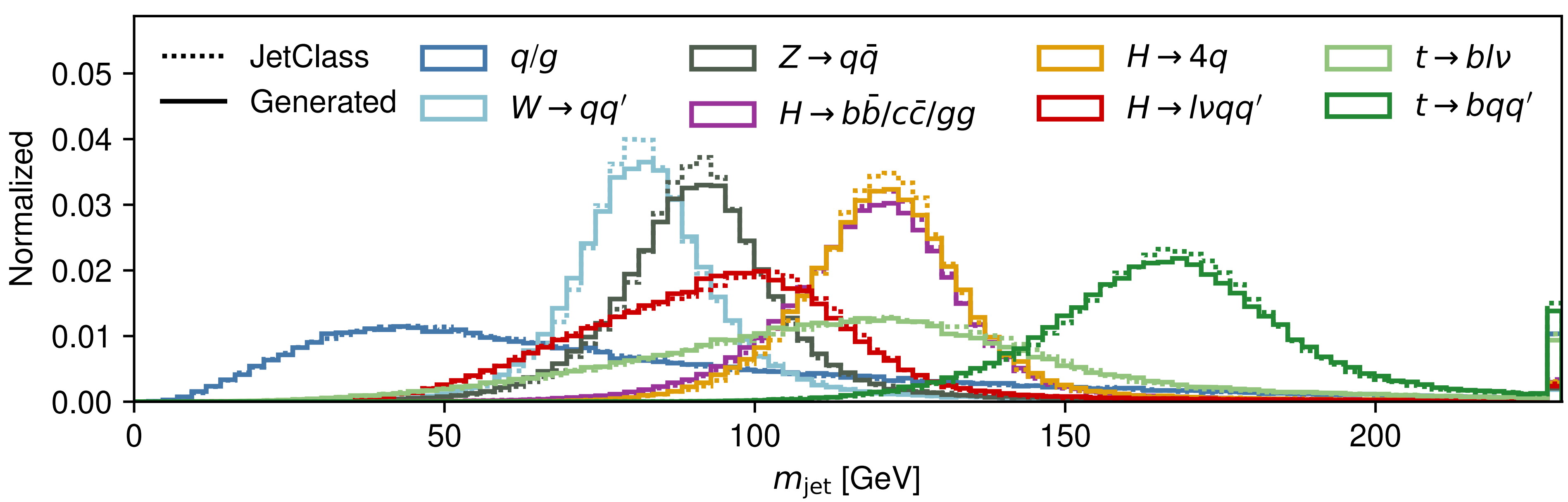
Simulation targets





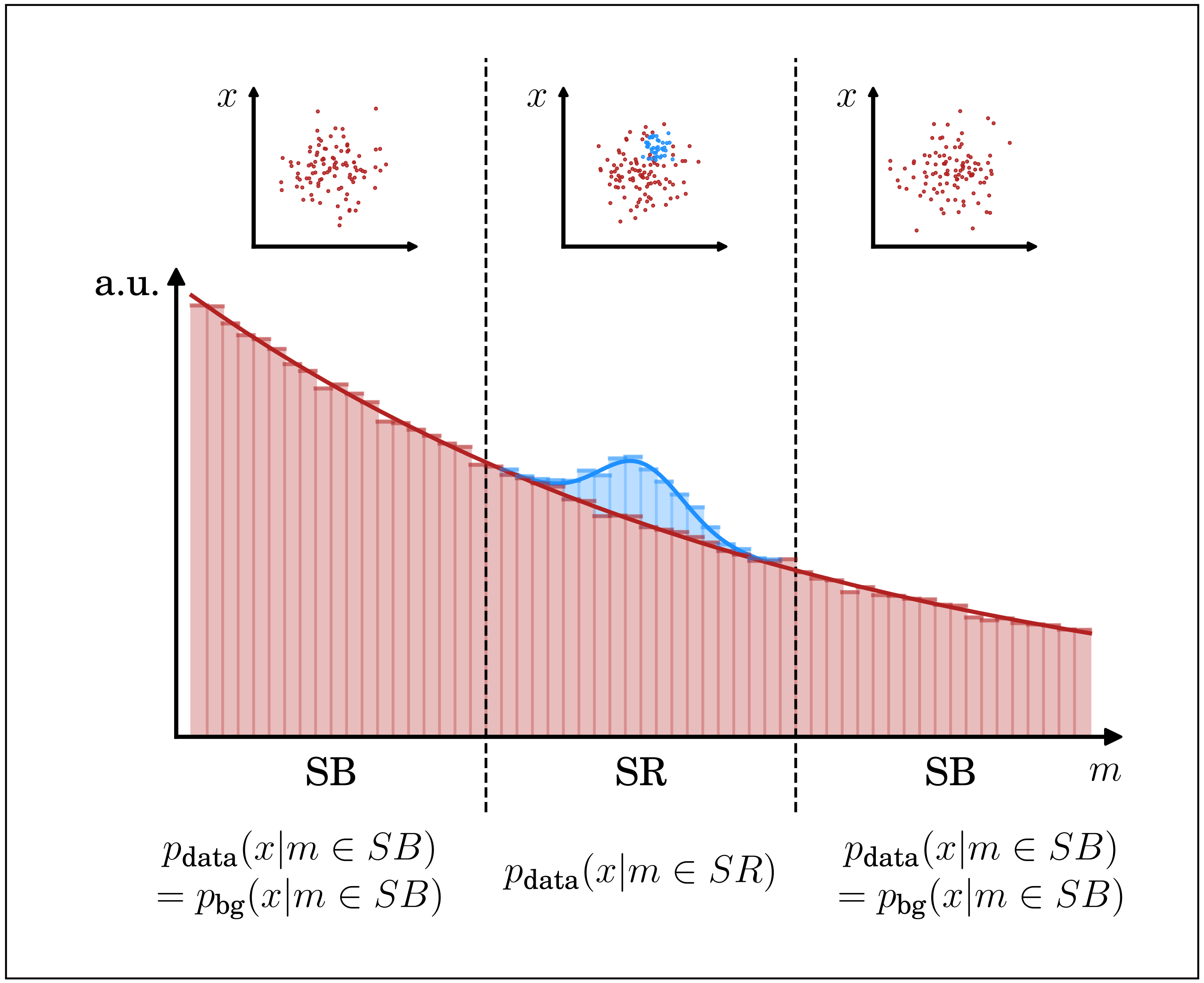
Apply EPiC + Flow matching to additional features

Good agreement across distributions



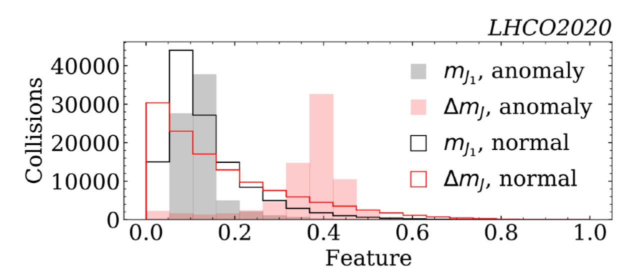
2310.00049, 2312.00123

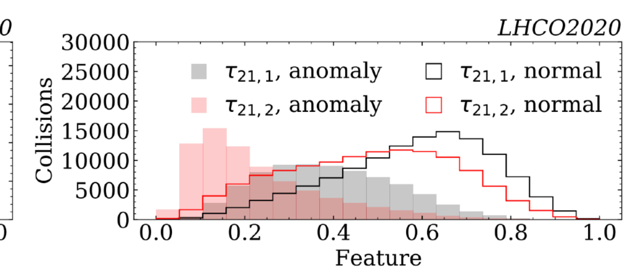
Reminder: CATHODE



1. Train generative model (conditional normalising flow)

Replace 4 high level features:



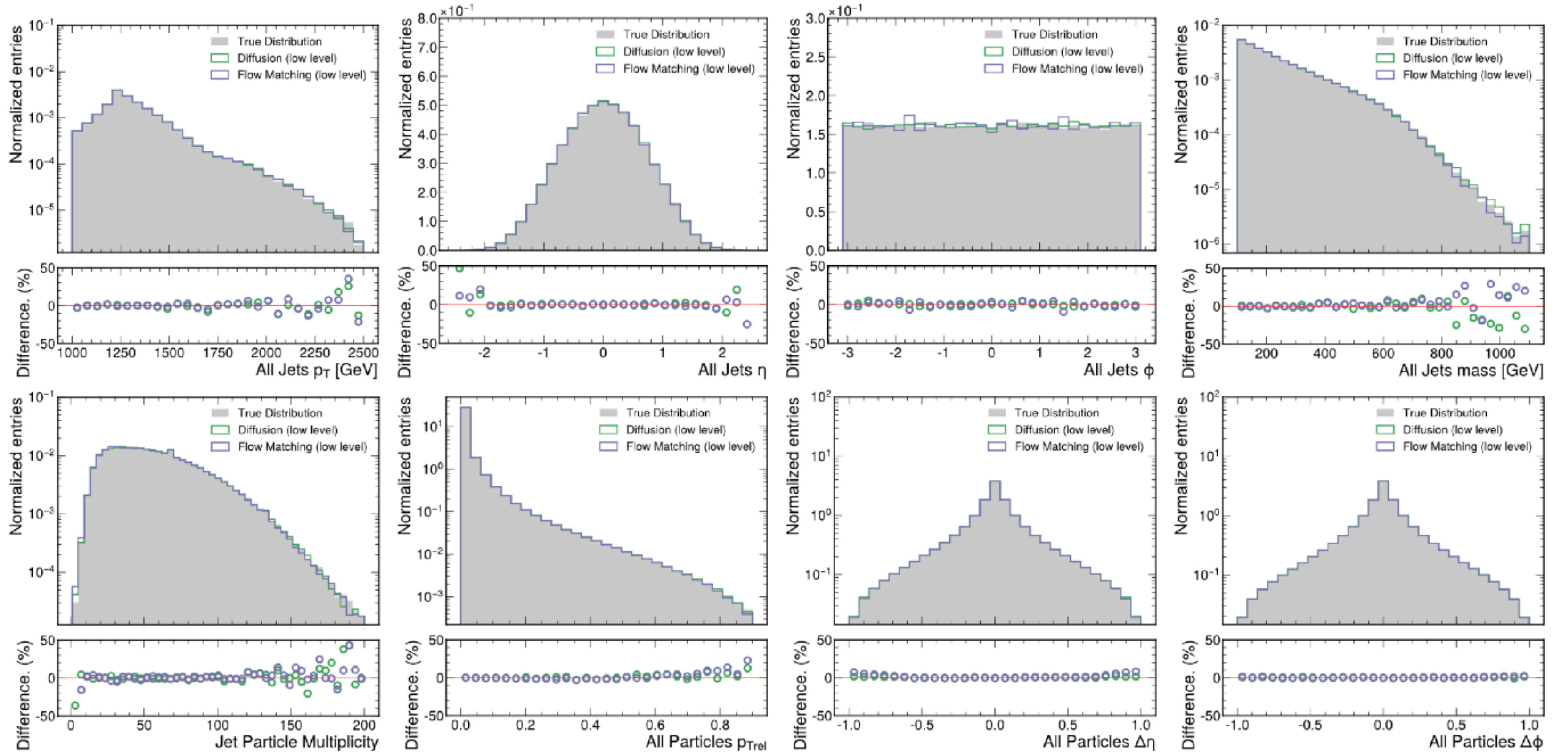


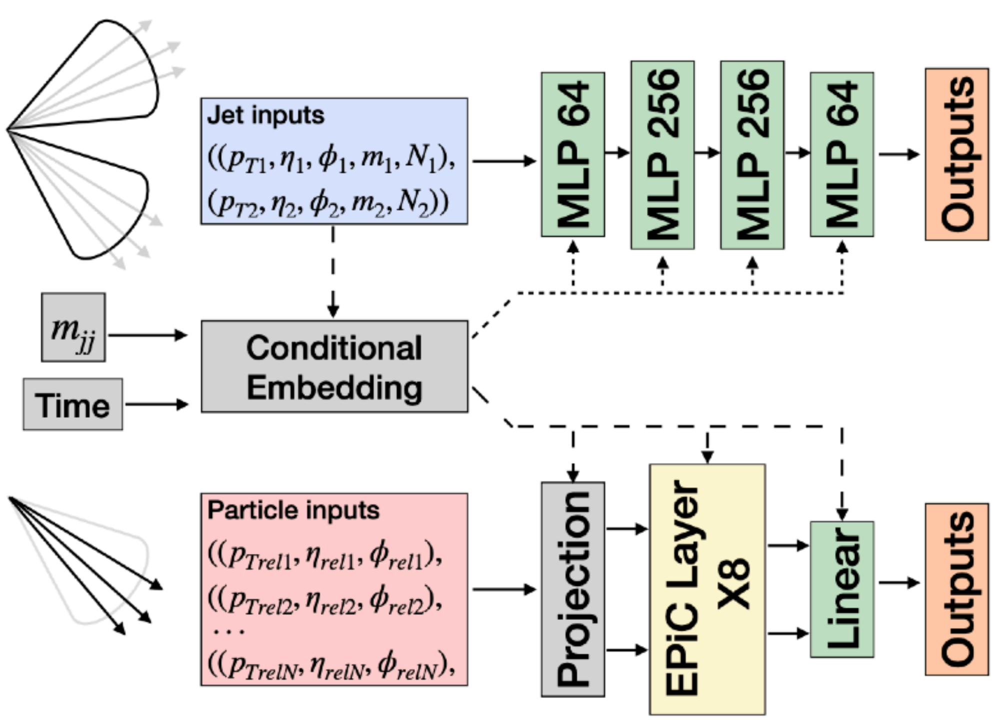
with 1674 low-level features (279 constituent 3-vectors each for 2 jets)

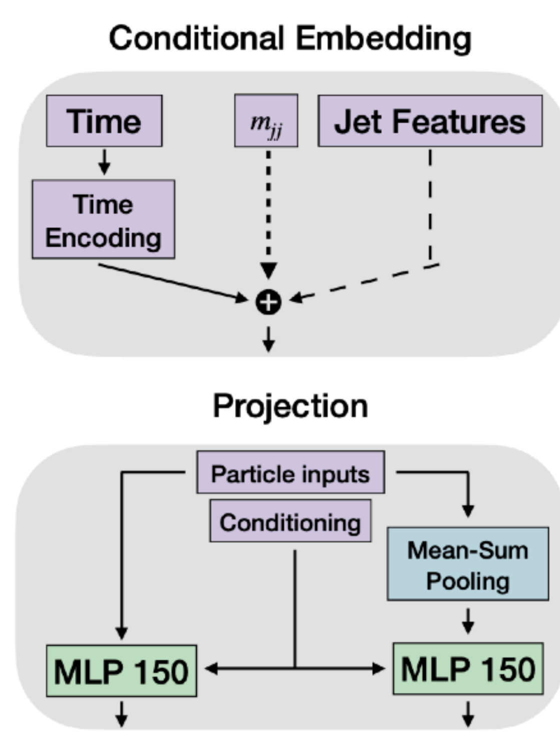
Buhmann, .., **GK**, Mikuni, et al 2310.06897;

Particle Inputs

Using all low-level features in-principle includes all properties





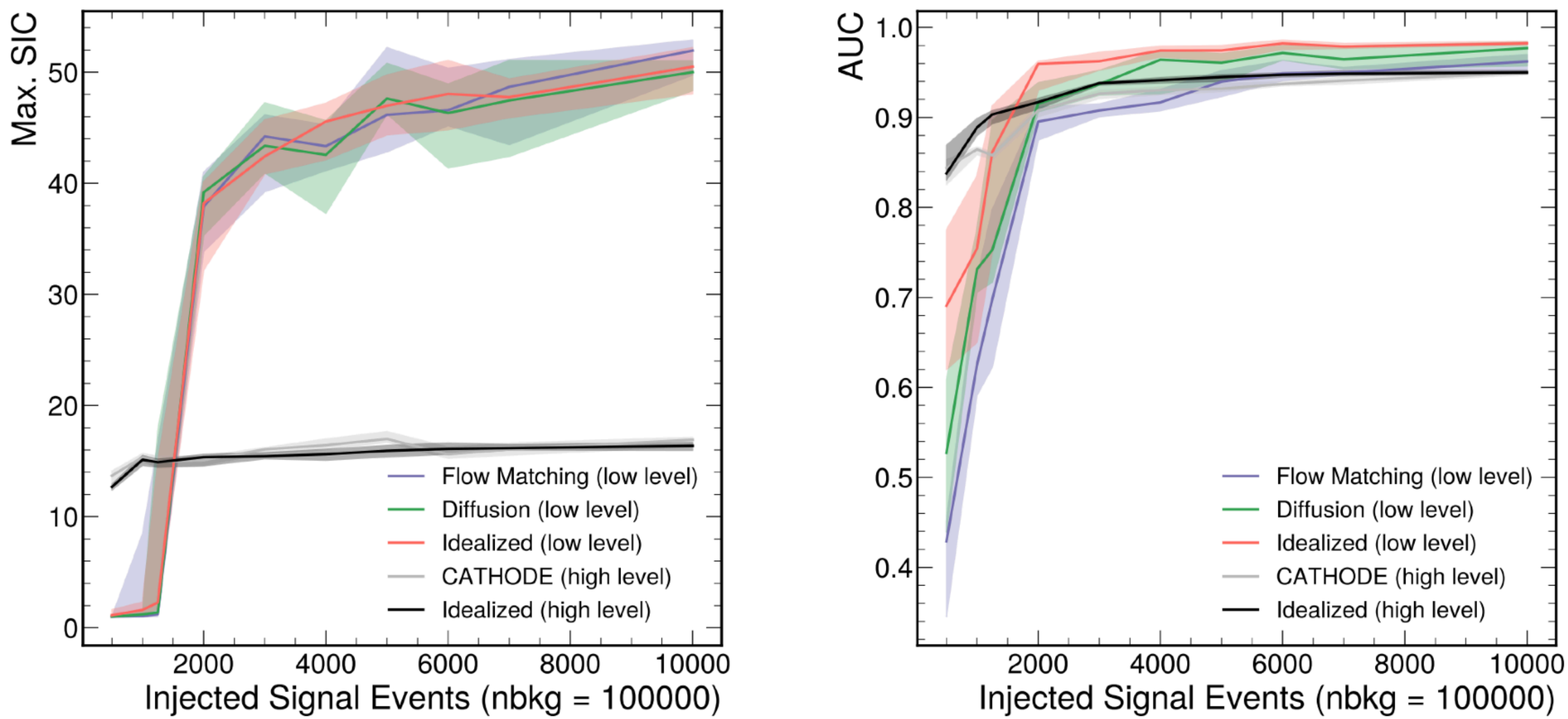


Use point cloud diffusion/ flow matching

CATHODE classifier is a transformer

Buhmann, .., **GK**, Mikuni, et al 2310.06897;

Particle Inputs



Greatly improves maximum SIC, but
currently requires more initial signal

Buhmann, .., **GK**, Mikuni, et al 2310.06897;

Why generative models?

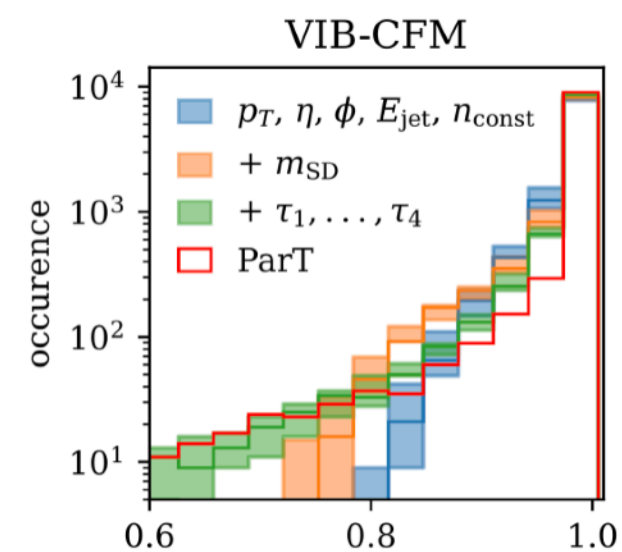
Showers in complex high-resolution calorimeters

High-level jet features

Jet constituents

$$p(x)$$

Sample $X_i \sim p(x)$
to generate datapoints



Classifier surrogates

Classifier Surrogates: Motivation

Les Houches guide to reusable ML models in LHC analyses

Jack Y. Araz[1], *Andy Buckley*[2], *Gregor Kasieczka*[3], *Jan Kieseler*[4], *Sabine Kraml*[5], *Anders Kvellestad*[6], *Andre Lessa*[7], *Tomasz Procter*[2], *Are Raklev*[6], *Humberto Reyes-Gonzalez*[8,9,10], *Krzysztof Rolbiecki*[11], *Sezen Sekmen*[12], *Gokhan Unel*[13]

[1] Jefferson Lab, Newport News, VA 23606, USA
[2] University of Glasgow, Glasgow, UK
[3] Univ. Hamburg, Germany
[4] Karlsruhe Institute for Technology, Karlsruhe, Germany
[5] Univ. Grenoble Alpes, CNRS, Grenoble INP, LPSC-IN2P3, 38000 Grenoble, France
[6] University of Oslo, 0316 Oslo, Norway
[7] Universidade Federal do ABC, Santo André, 09210-580 SP, Brazil
[8] Department of Physics, University of Genova, Via Dodecaneso 33, 16146 Genova, Italy
[9] INFN, Sezione di Genova, Via Dodecaneso 33, I-16146 Genova, Italy
[10] Institut für Theoretische Teilchenphysik und Kosmologie, RWTH Aachen, 52074 Aachen, Germany
[11] Faculty of Physics, University of Warsaw, 02-093 Warsaw, Poland
[12] Department of Physics, Kyungpook National University, Daegu, South Korea
[13] U.C. Irvine, Physics & Astronomy Dept., Irvine, CA, USA

Abstract

With the increasing usage of machine-learning in high-energy physics analyses, the publication of the trained models in a reusable form has become a crucial question for analysis preservation and reuse. The complexity of these models creates practical issues for both reporting them accurately and for ensuring the stability of their behaviours in different environments and over extended timescales. In this note we discuss the current state of affairs, highlighting specific practical issues and focusing on the most promising technical and strategic approaches to ensure trustworthy analysis-preservation. This material originated from discussions in the LHC Reinterpretation Forum and the 2023 PhysTeV workshop at Les Houches.

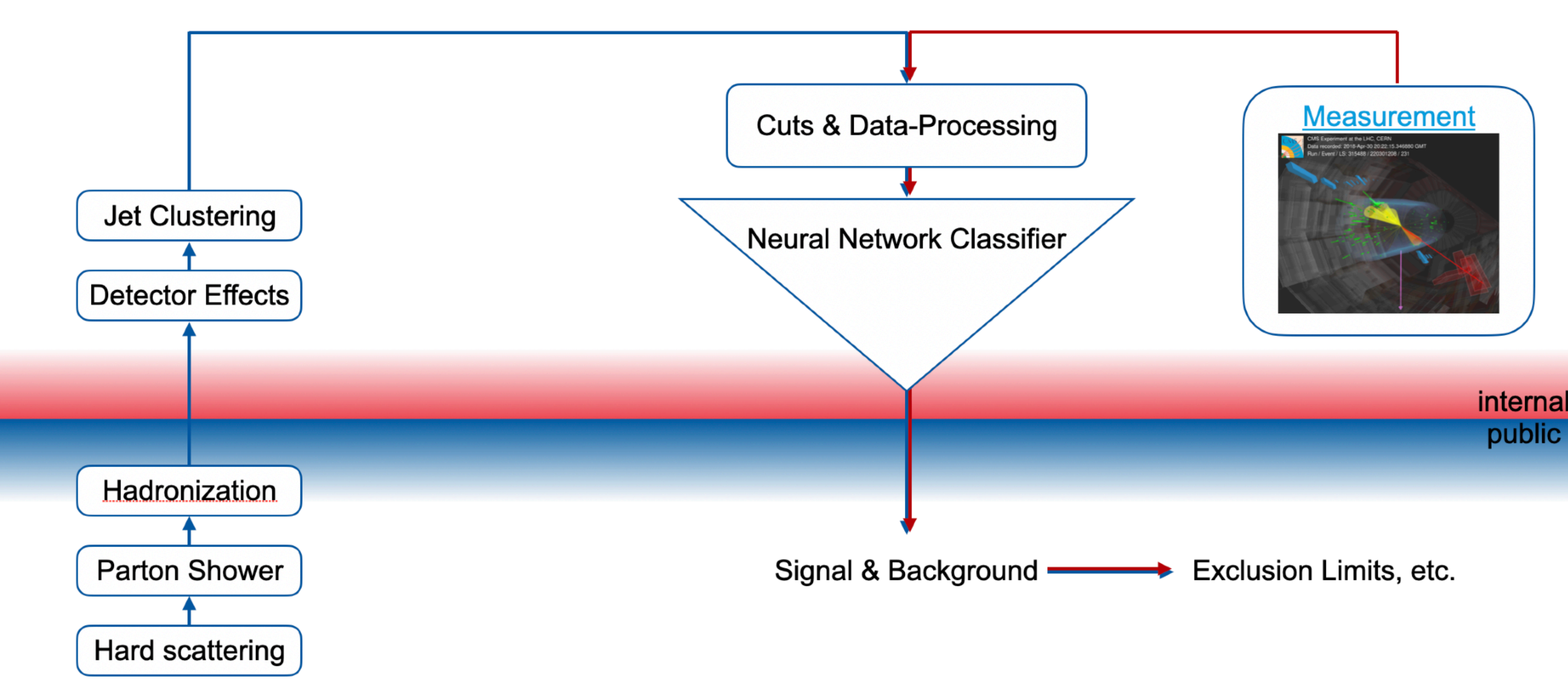
Keywords

BSM; Tools; Machine-learning; Reinterpretation.

Guidelines for ML model exchange including suggestion of surrogate models

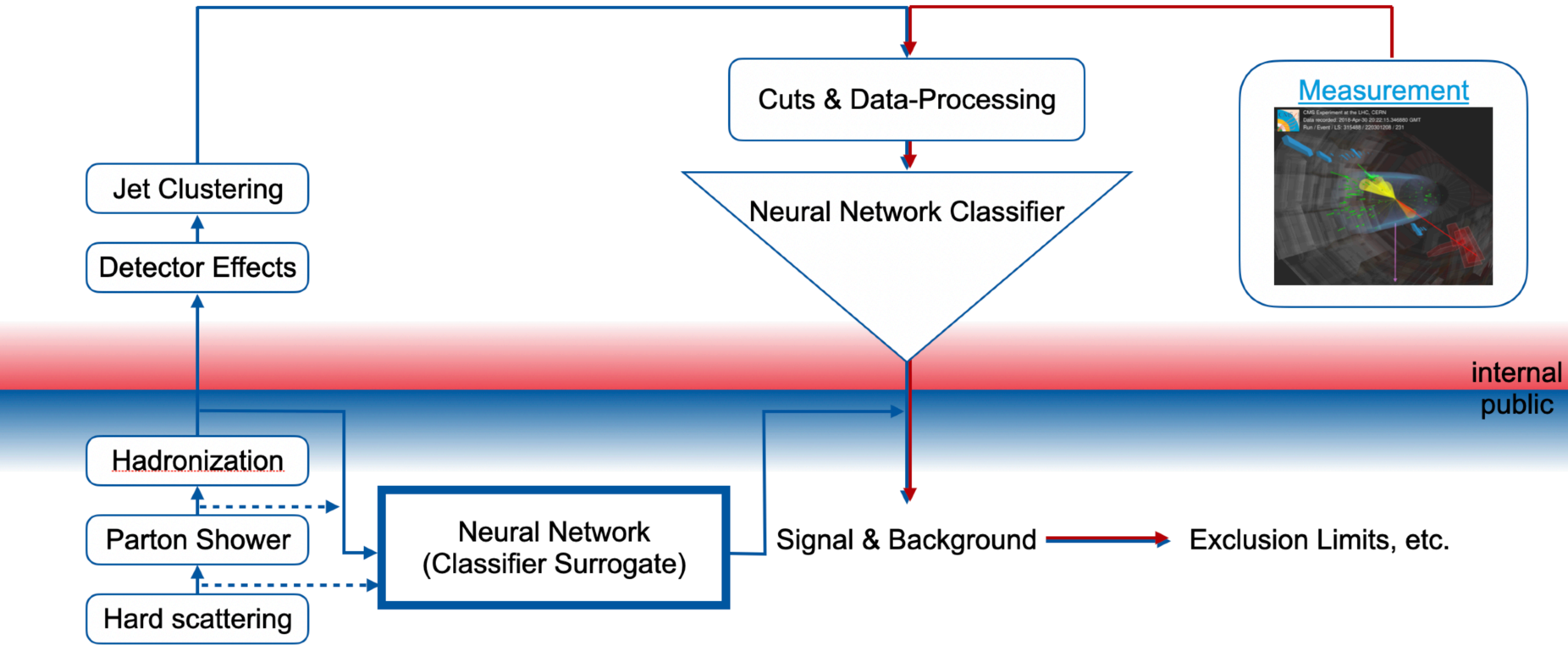
Araz, Buckley, GK, et al 2312.14575

Classifier Surrogates



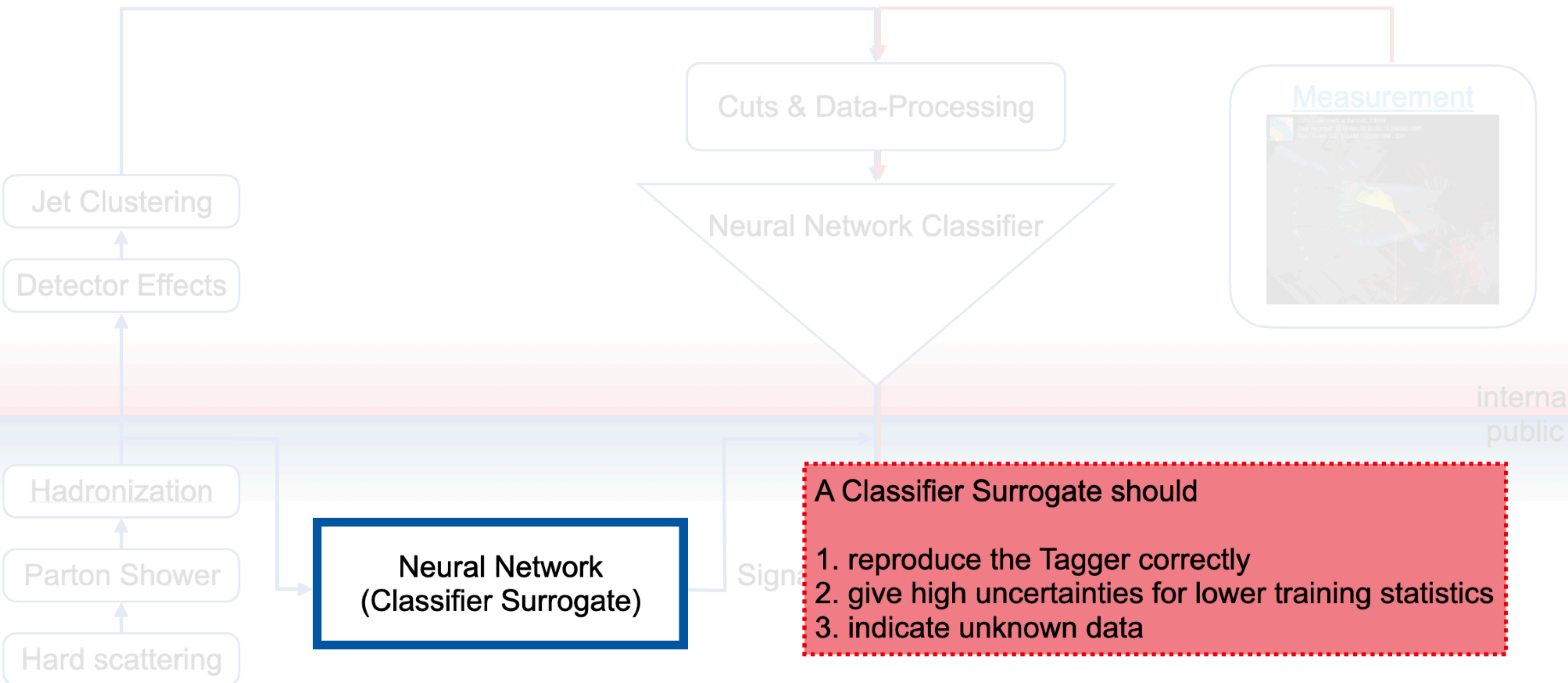
(Material by Sebastian Bieringer)

Classifier Surrogates



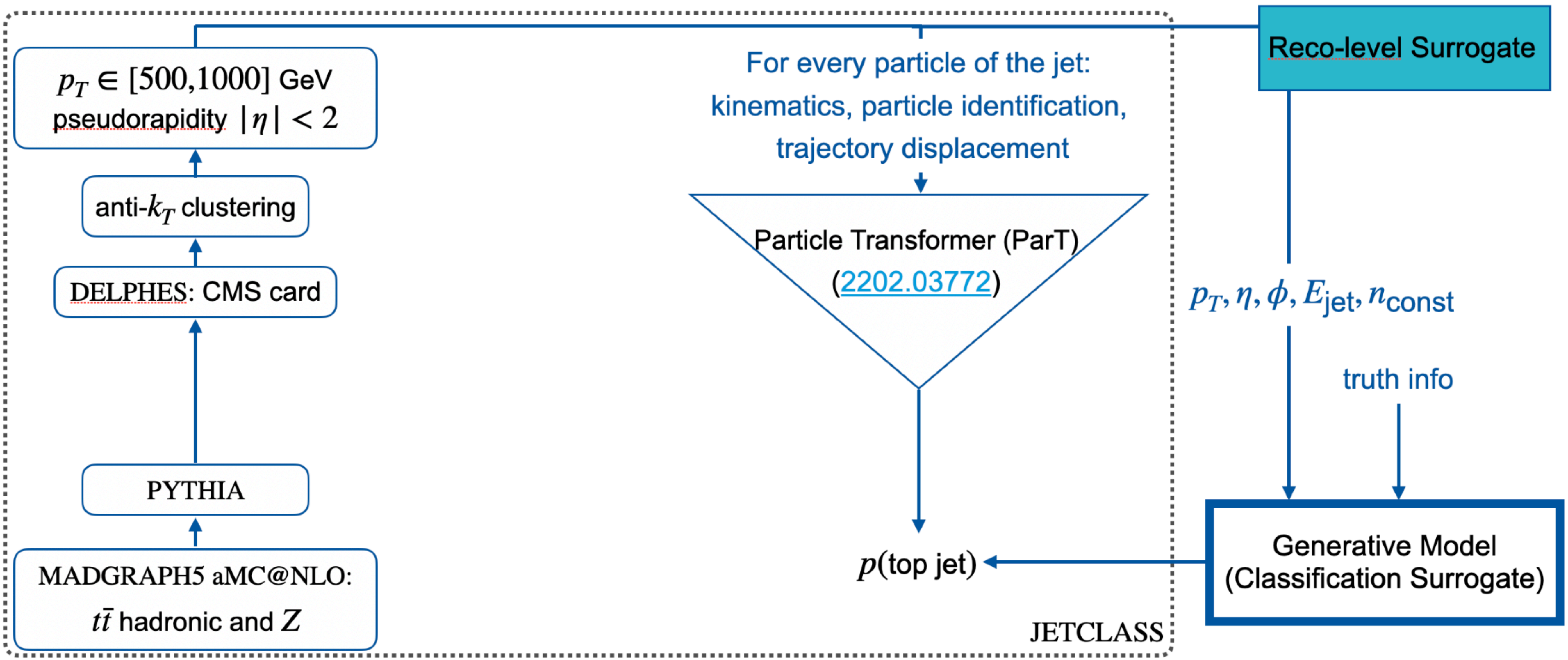
(Material by Sebastian Bieringer)

Classifier Surrogates



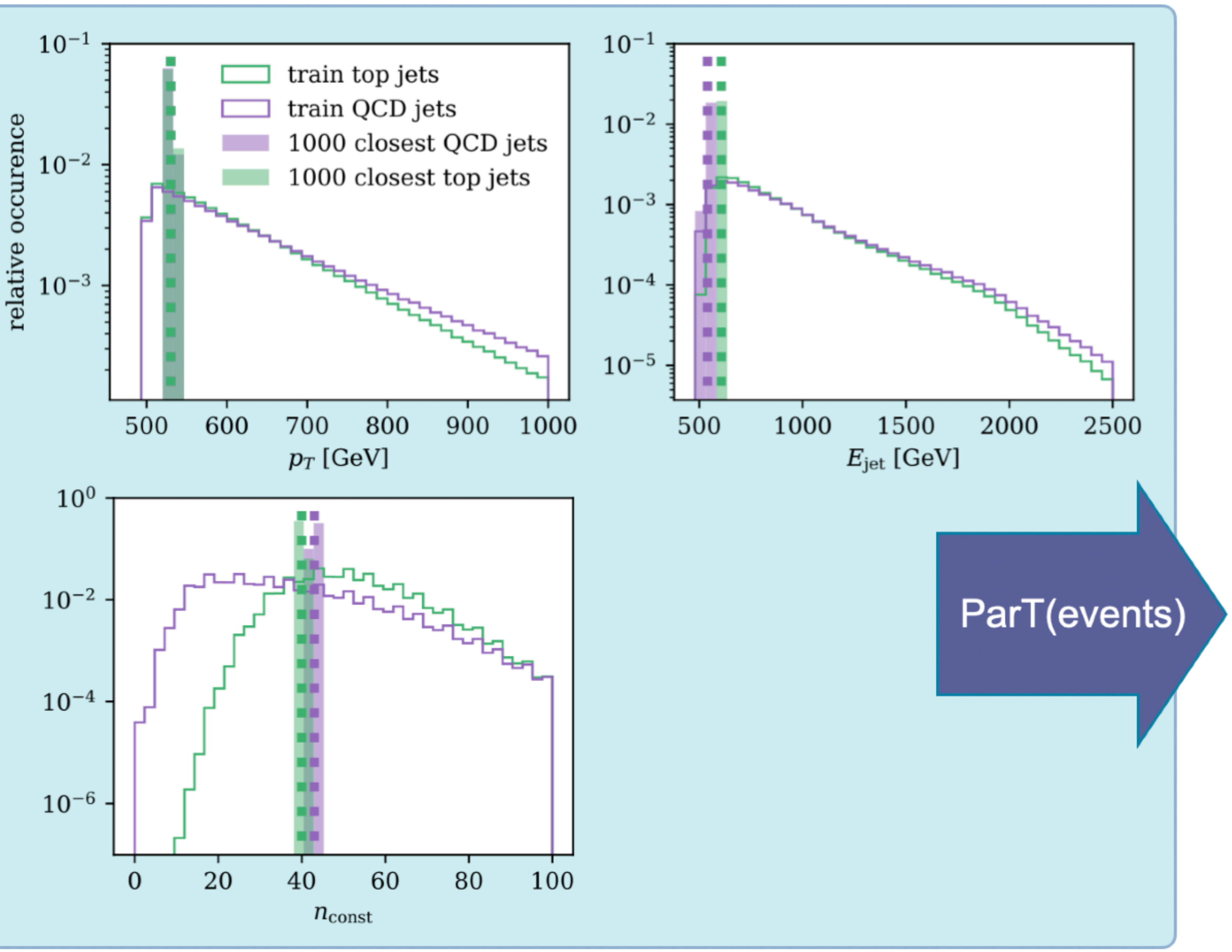
(Material by Sebastian Bieringer)

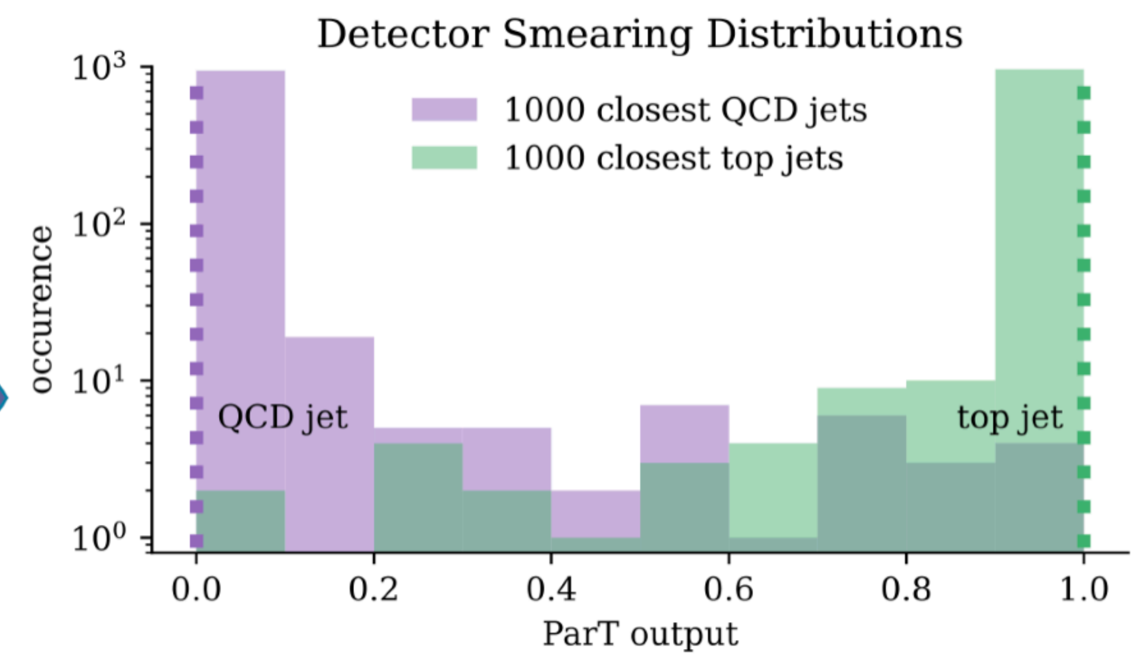
The Toy Setup



(Material by Sebastian Bieringer)

Detector Smearing Distribution





A Classifier Surrogate has to be a **Generative Model**

(Material by Sebastian Bieringer)

The Generative Model

Continuous Normalizing Flow:

- Flow $\phi : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ defined via

$$\frac{\mathrm{d}}{\mathrm{d}t}\phi_t(x) = v_t(\phi_t(x)) = \tilde{v}_t(x,\theta)$$

- solve the ODE to train and sample
- linear trajectory
- transforms probability distributions

$$p_t(x) = p_0\left(\phi_t^{-1}(x)\right) \det\left[\frac{\partial \phi_t^{-1}}{\partial x}(x)\right]$$

Conditional Flow Matching:

- loss that does not ODE solving

$$\mathscr{L}_{\mathrm{FM}}(\theta) = \mathbb{E}_{t,p_t(x)} \left\| v_t(x) - \tilde{v}_t(x,\theta)) \right\|^2$$

- by choice of p_t and v_t

$$\mathscr{L}_{\mathrm{CFM}}(\theta) = \mathbb{E}_{t,p_t(x),\epsilon}\left[\tilde{v}_t\left((1-t)x_0 + t\epsilon, \theta\right) - \left(\epsilon - x_0\right)\right]^2$$

- not a log-Likelihood loss

Variational Inference Bayesian Conditional Flow Matching:

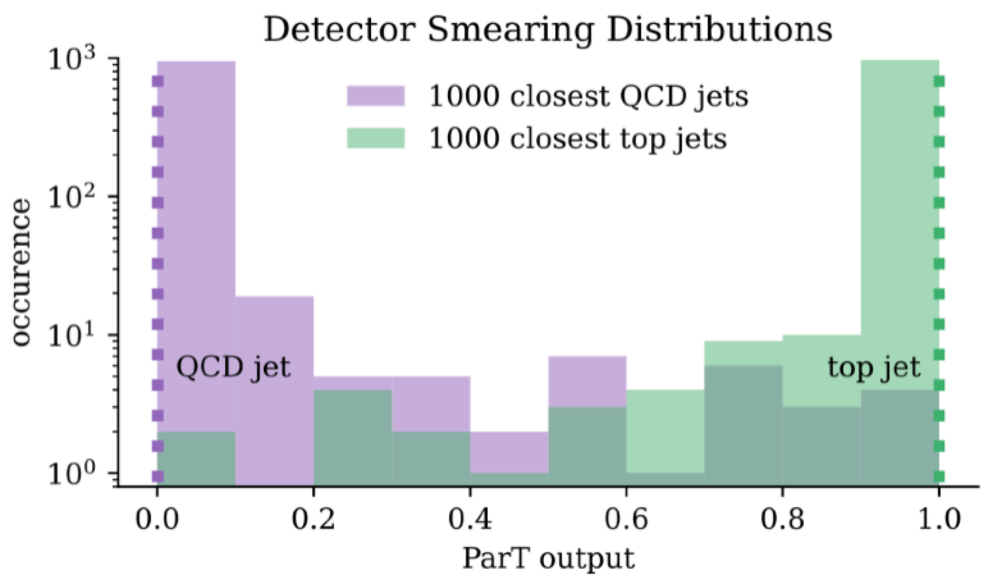
- Bayesian loss $\mathscr{L}_{\mathrm{BNN}} = \mathrm{KL}\left[q(\theta), p\left(\theta \mid x\right)\right] = -\int \mathrm{d}\theta\, q(\theta)\, \log p\left(x \mid \theta\right) + \mathrm{KL}[q(\theta), p(\theta)] +$ const.
- connect both $\mathscr{L}_{\mathrm{B-CFM}} = \left\langle \mathscr{L}_{\mathrm{CFM}} \right\rangle_{\theta \sim q(\theta)} + c\mathrm{KL}[q(\theta), p(\theta)]$, with $q(\theta)$ uncorrelated Gaussian shape

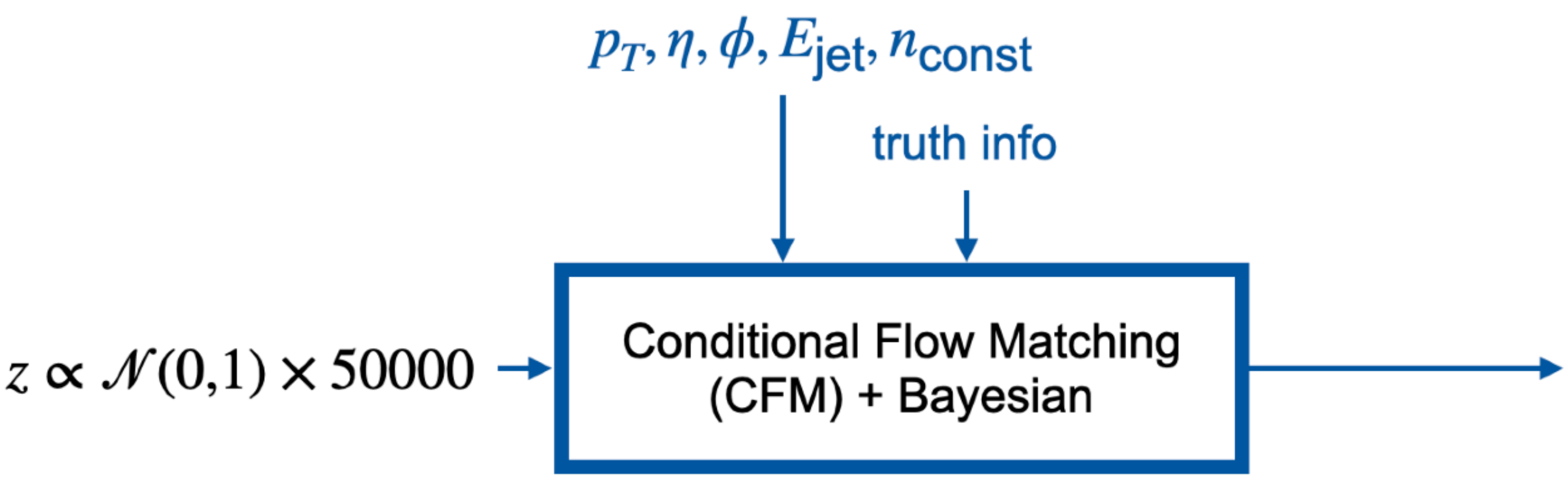
Adam-MCMC Bayesian Conditional Flow Matching:

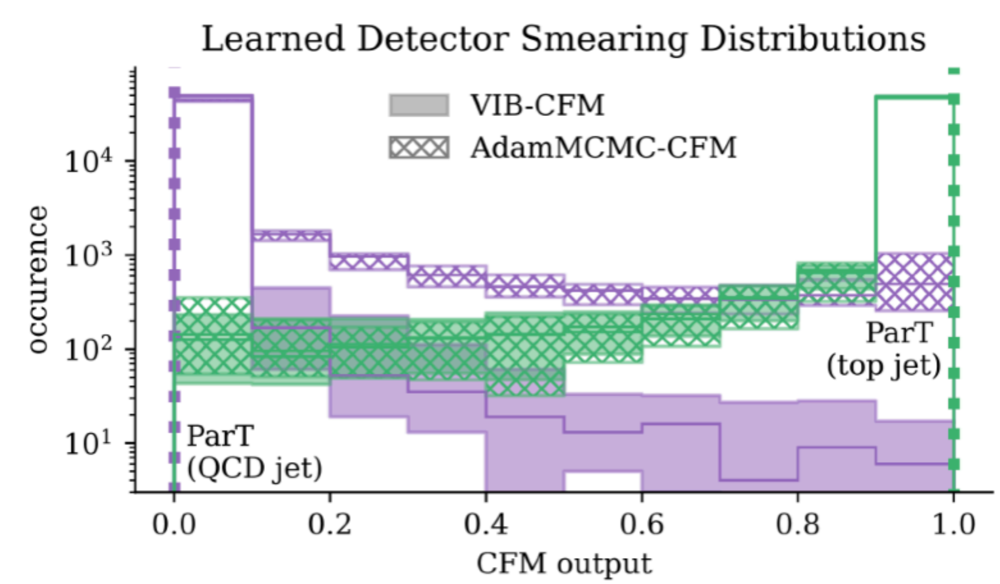
- train the network with CFM
- start Markov Chain from this point (independent of starting point):
  - 1D problem: Solve ODE to get log-Likelihood of batch for update steps and acceptance rates

(Material by Sebastian Bieringer)

Is the Classifier reproduced correctly?

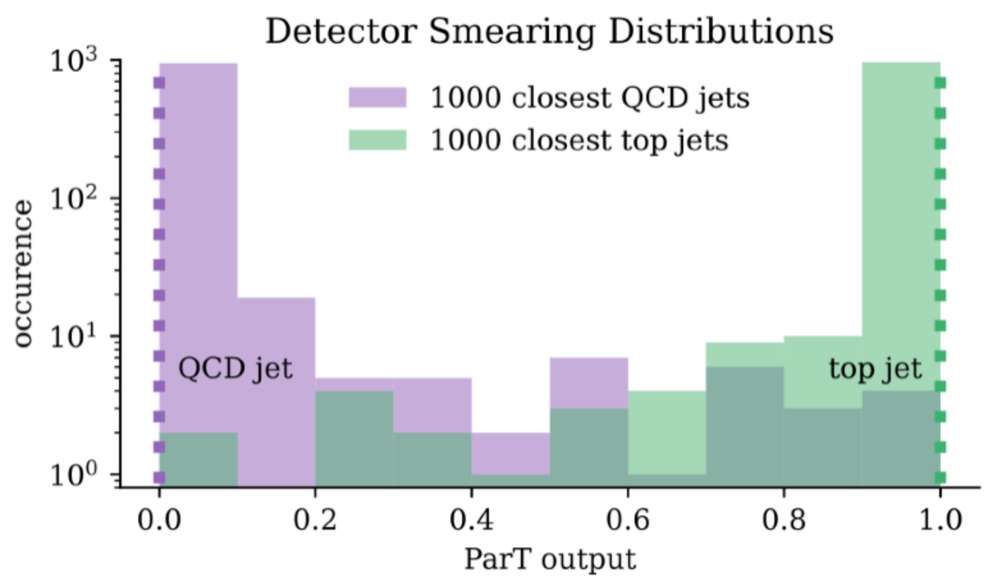






(Material by Sebastian Bieringer)

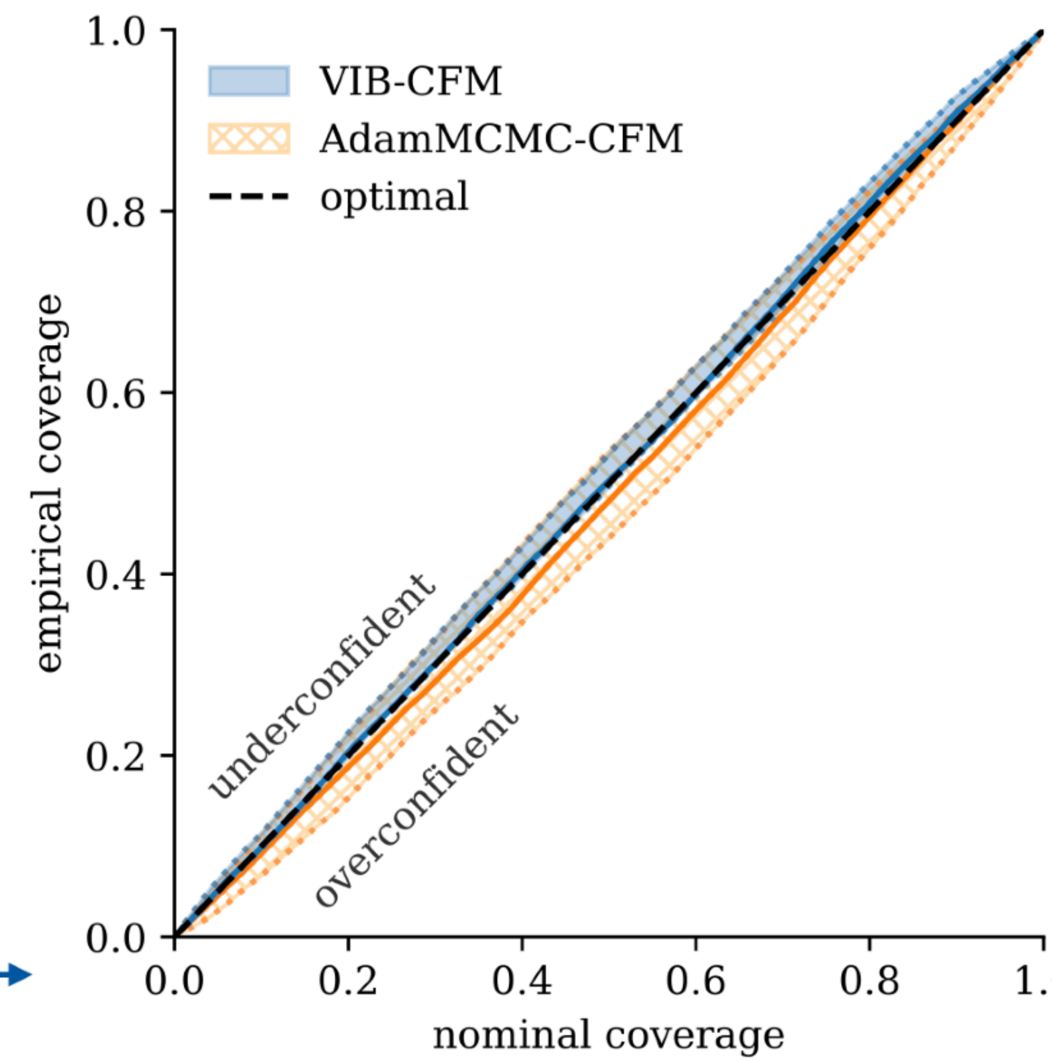
Is the Classifier reproduced correctly?



$10000 \times \{p_T, \eta, \phi, E_{\text{jet}}, n_{\text{const}}\}$

truth info

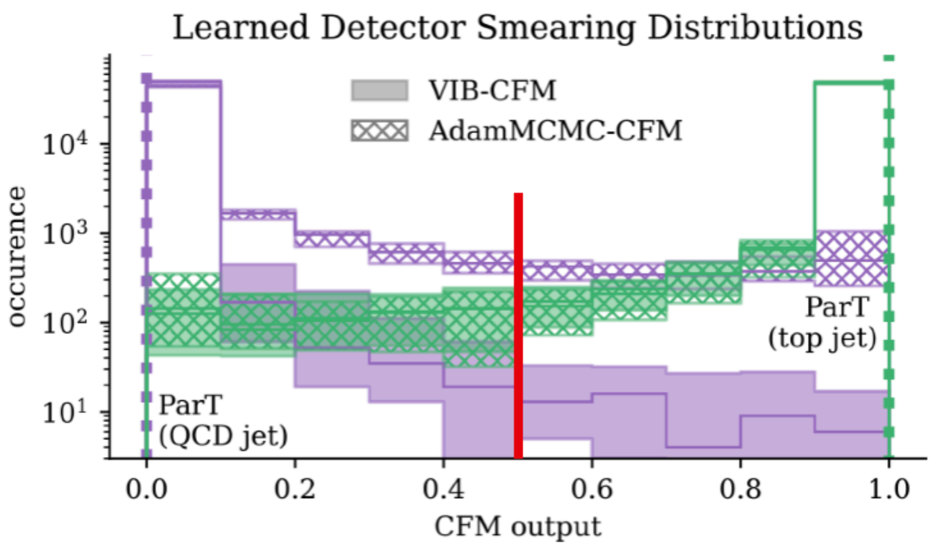
$z \propto \mathcal{N}(0,1) \times 1000$ → **Conditional Flow Matching (CFM) + Bayesian** →



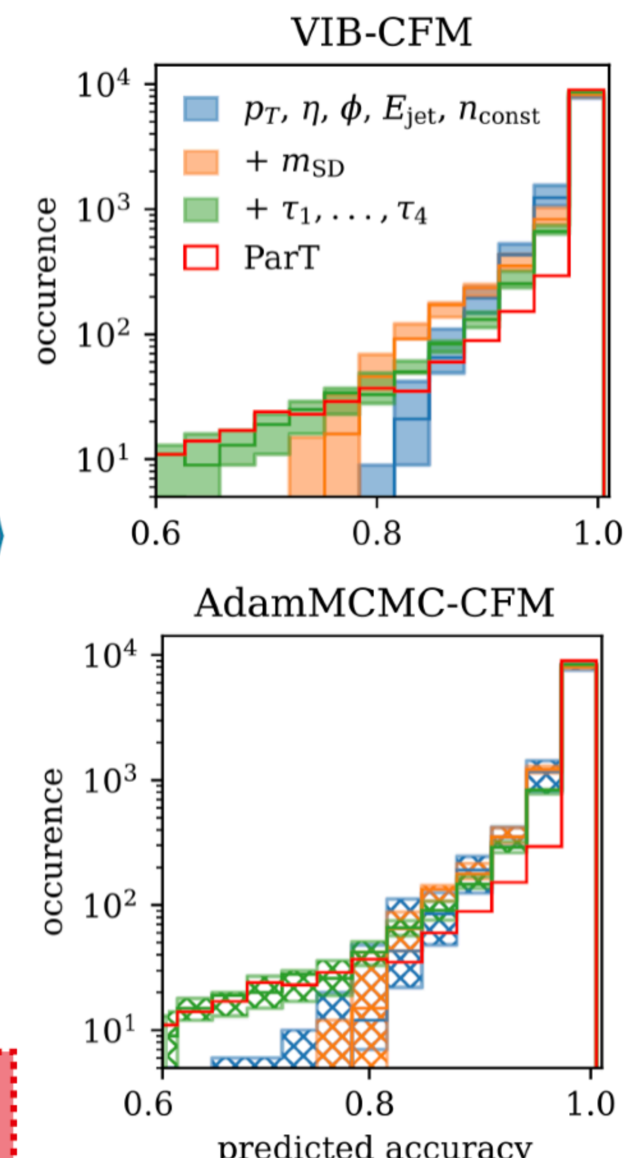
Both architectures deliver **well calibrated** Surrogates

(Material by Sebastian Bieringer)

Is the Classifier reproduced correctly?



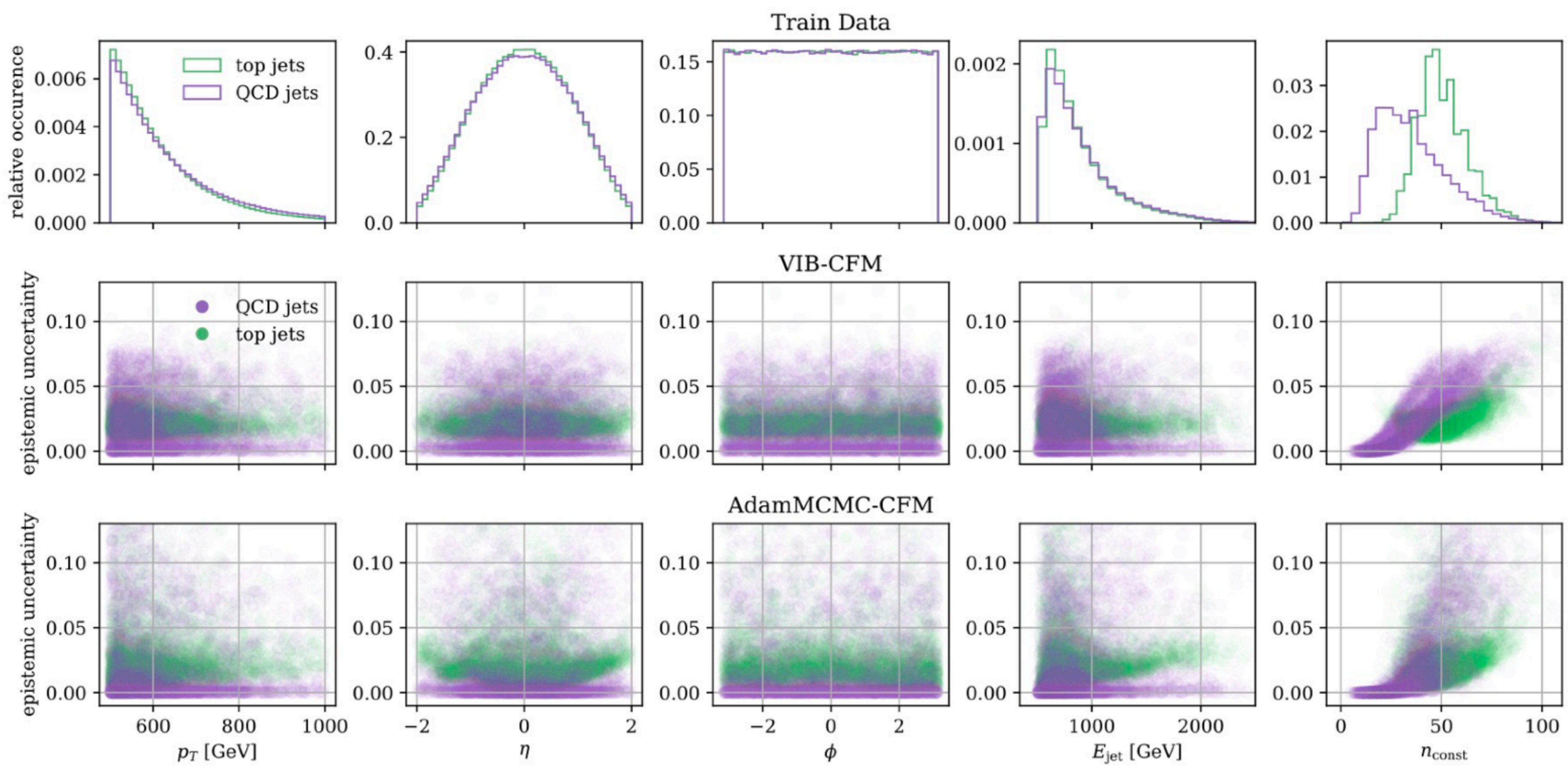
$$\text{accuracy} = \frac{1}{1000}\sum_{i=0}^{1000}\left(\mathbf{1}_{[0.5,1]}\left(\phi_{0,\theta}(x_i)\right)\mathbf{1}(\text{top jet}) + \mathbf{1}_{[0,0.5)}\left(\phi_{0,\theta}(x_i)\right)\mathbf{1}(\text{QCD jet})\right)$$



Quality of the Surrogate depends on input information

(Material by Sebastian Bieringer)

Are high uncertainties produced for lower training statistic?

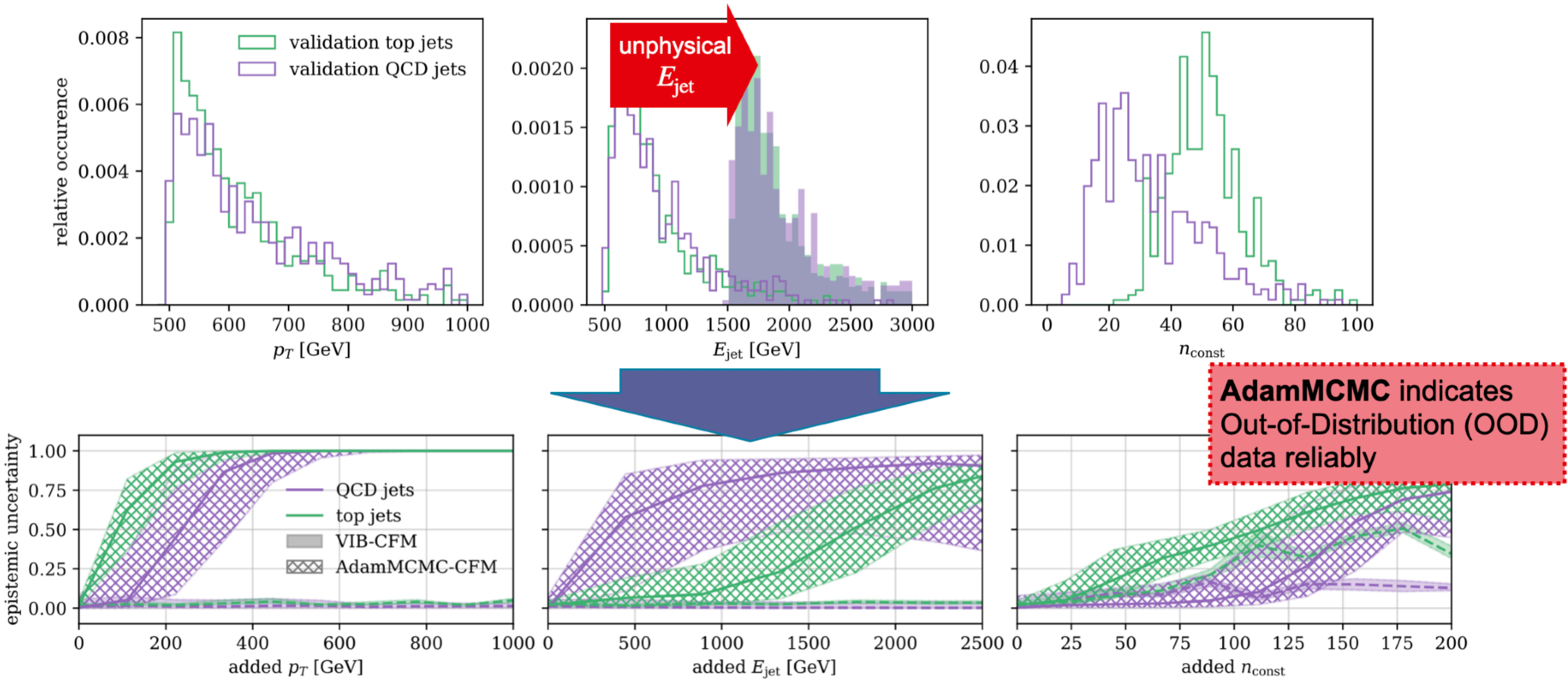


Uncertainties scale towards edges of train distribution

$$\text{epistemic uncertainty} = \frac{1}{n_{\text{stat}}} \sum_{i=0}^{n_{\text{stat}}} \left(\max_{\theta} \phi_{0,\theta}(x_i) - \min_{\theta} \phi_{0,\theta}(x_i) \right)$$

(Material by Sebastian Bieringer)

Is OOD data indicated?



(Material by Sebastian Bieringer)

Comments

A Bayesian - Conditional Flow Matching Surrogate can

- ✓ learn Tagger output well calibrated and to high accuracy
- ✓ indicate data sparse areas
- ✓ report high uncertainties for unknown input if sampled with MCMC

Paper is coming soon!

(Material by Sebastian Bieringer)

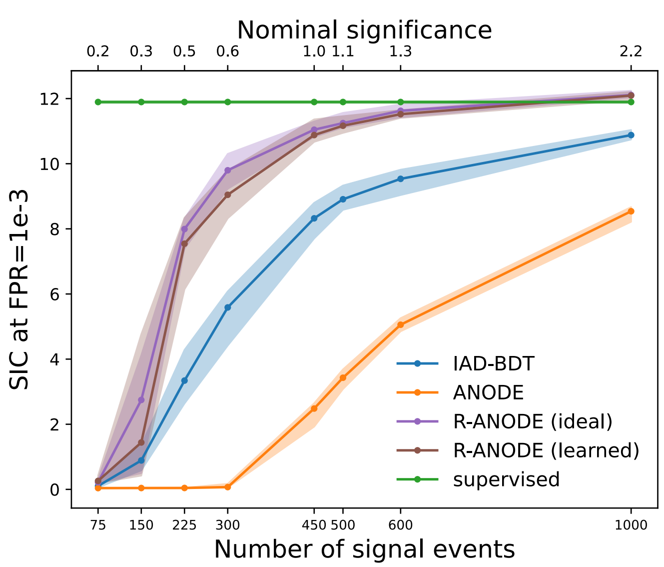
Why generative models?

Showers in complex high-resolution calorimeters

High-level jet features

Jet constituents

$$p(x)$$



Detect anomalies

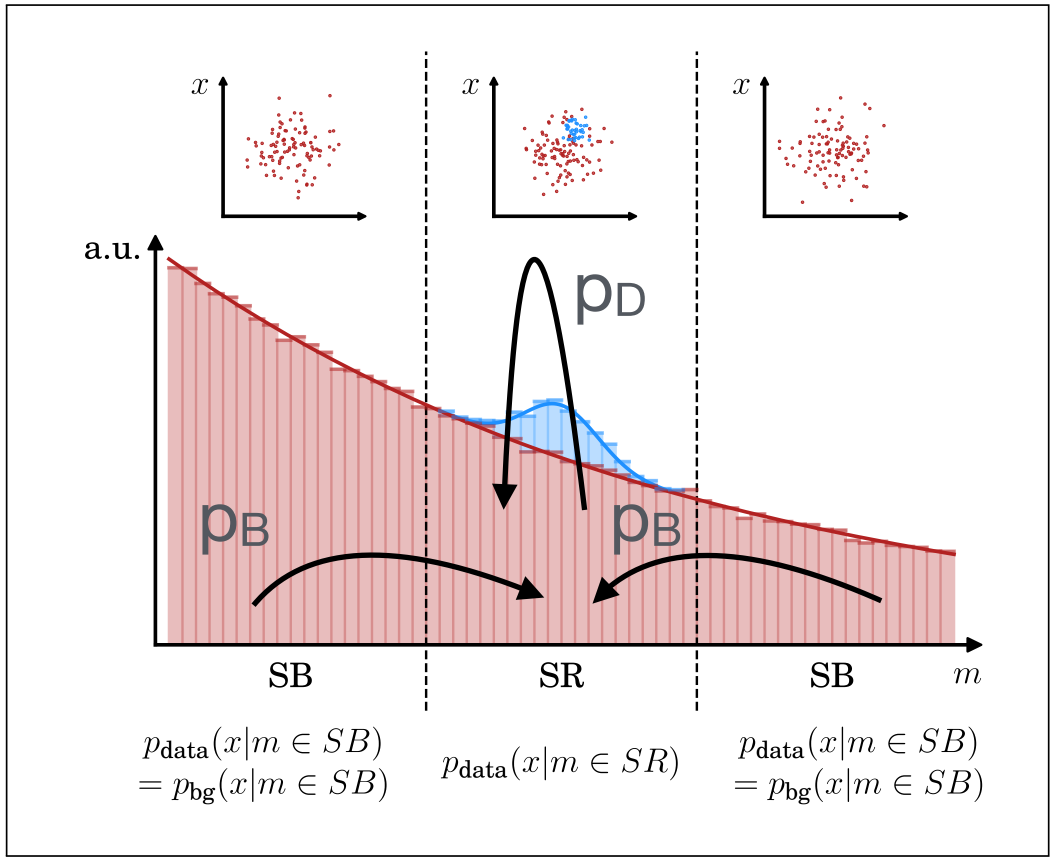
Sample $X_i \sim p(x)$ to generate datapoints

Evaluate p(x) directly as likelihood

Classifier surrogates

ANODE

Before CATHODE,
there was ANODE



ANODE:
Train and interpolate
background flow p_B
(as in CATHODE)

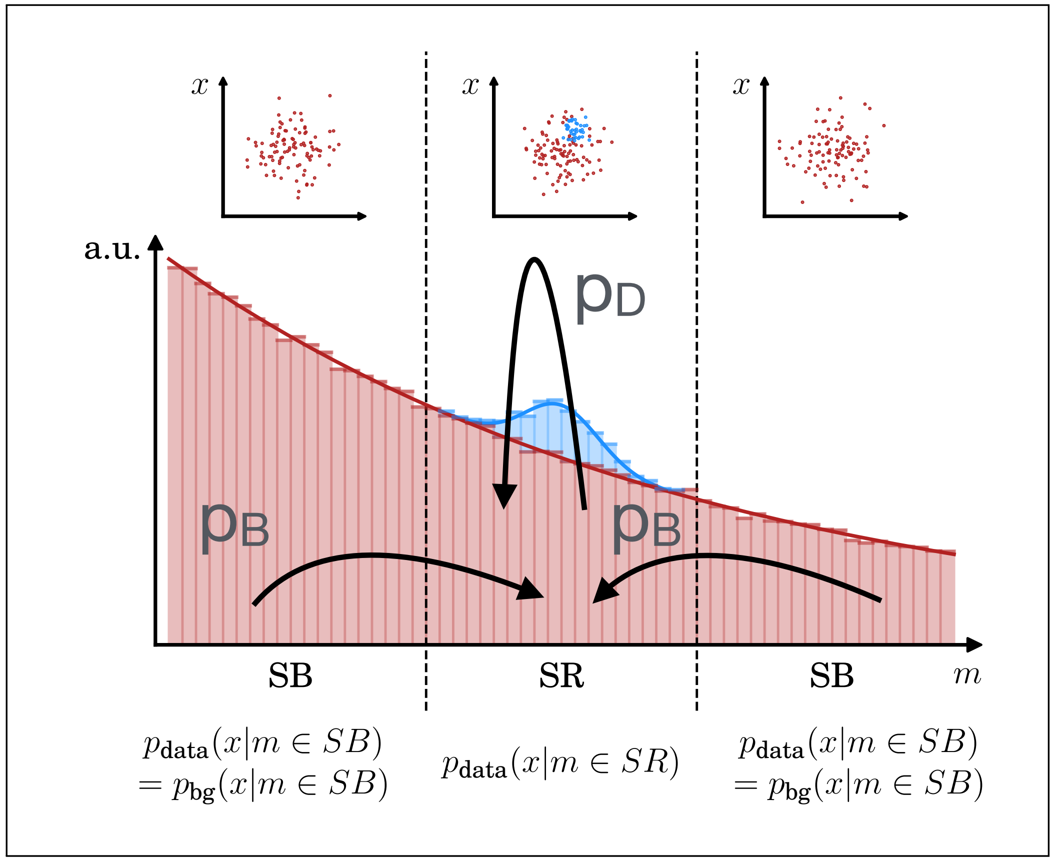
Train signal-region flow p_D

Anomaly score = p_D/p_B

Nachman, Shih, 2001.04990

ANODE

Before CATHODE,
there was ANODE

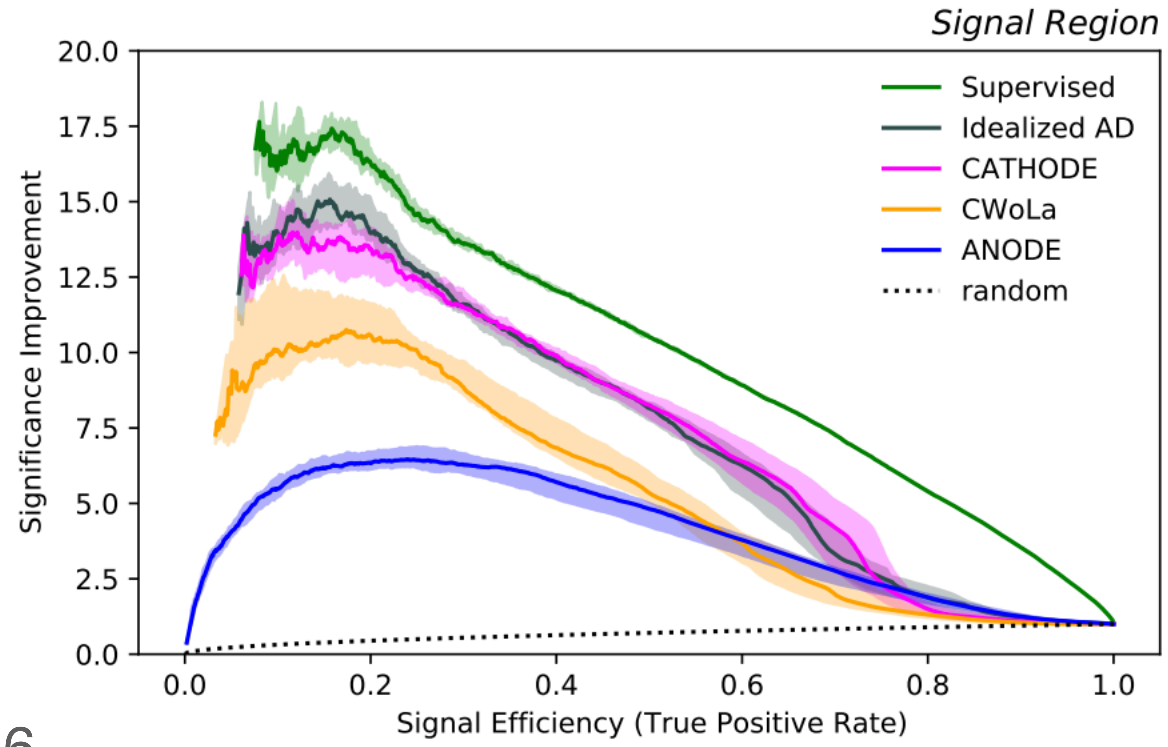


ANODE:
Train and interpolate background flow p_B (as in CATHODE)

Train signal-region flow p_D

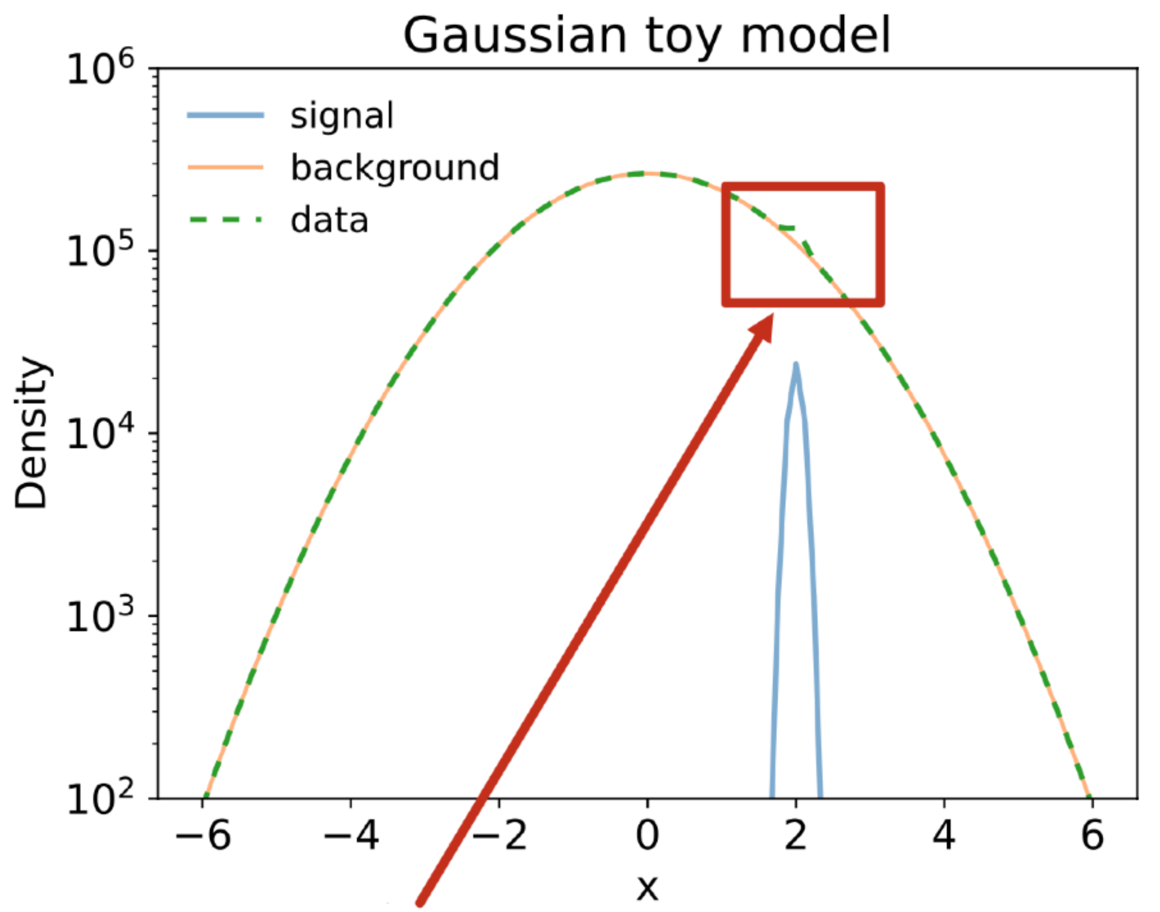
Anomaly score = p_D/p_B

Original ANODE:
Worse than CATHODE



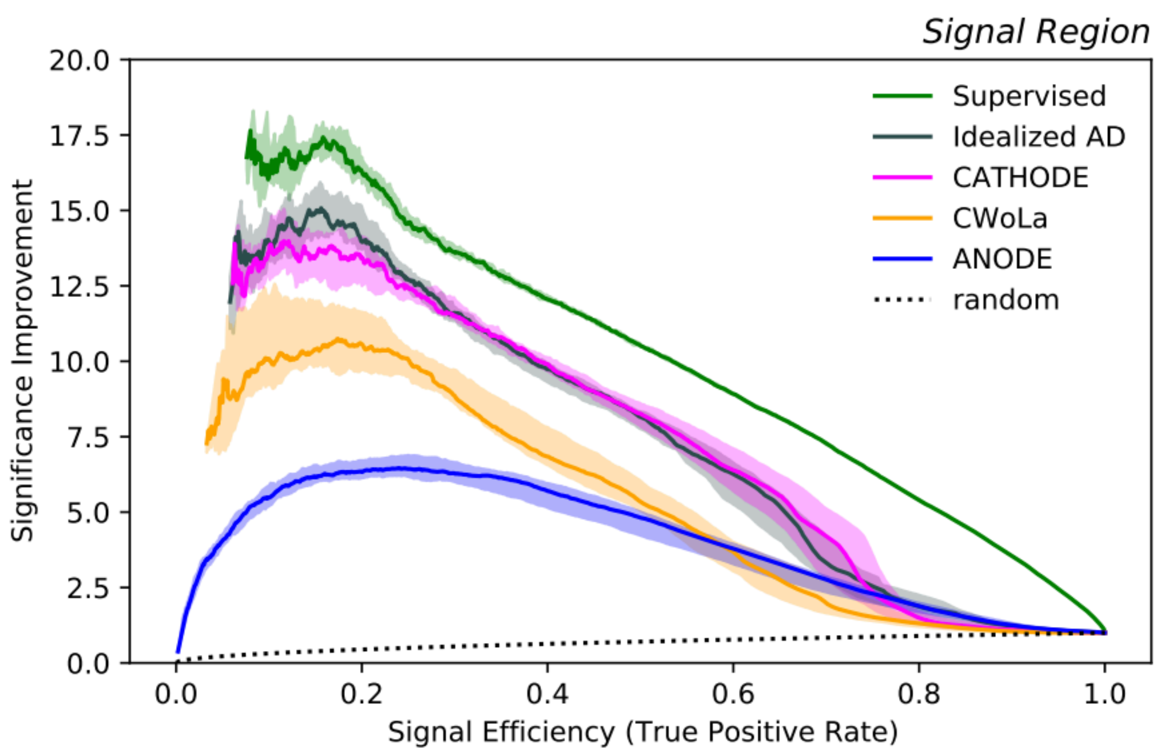
Nachman, Shih, 2001.04990, Hallin,..**GK**, et al 2109.00546

ANODE



ANODE must learn the sharply peaked distributions in x where the signal is localized.

Original ANODE:
Worse than CATHODE



Residual ANODE

R-ANODE:
Train and interpolate
background flow p_B
(as in ANODE, CATHODE)

Train signal contribution
flow using background

(Assumed) Fraction of signal in signal region

$$p_{\rm data}(x,m) = w\, p_{\rm sig}(x,m) + (1-w)\, p_{\rm bg}(x,m)$$

Learn in signal region via

$$L = -\mathbb{E}_{x,m\sim \rm SR\ data} \log p_{\rm data}(x,m)$$

Interpolated background density

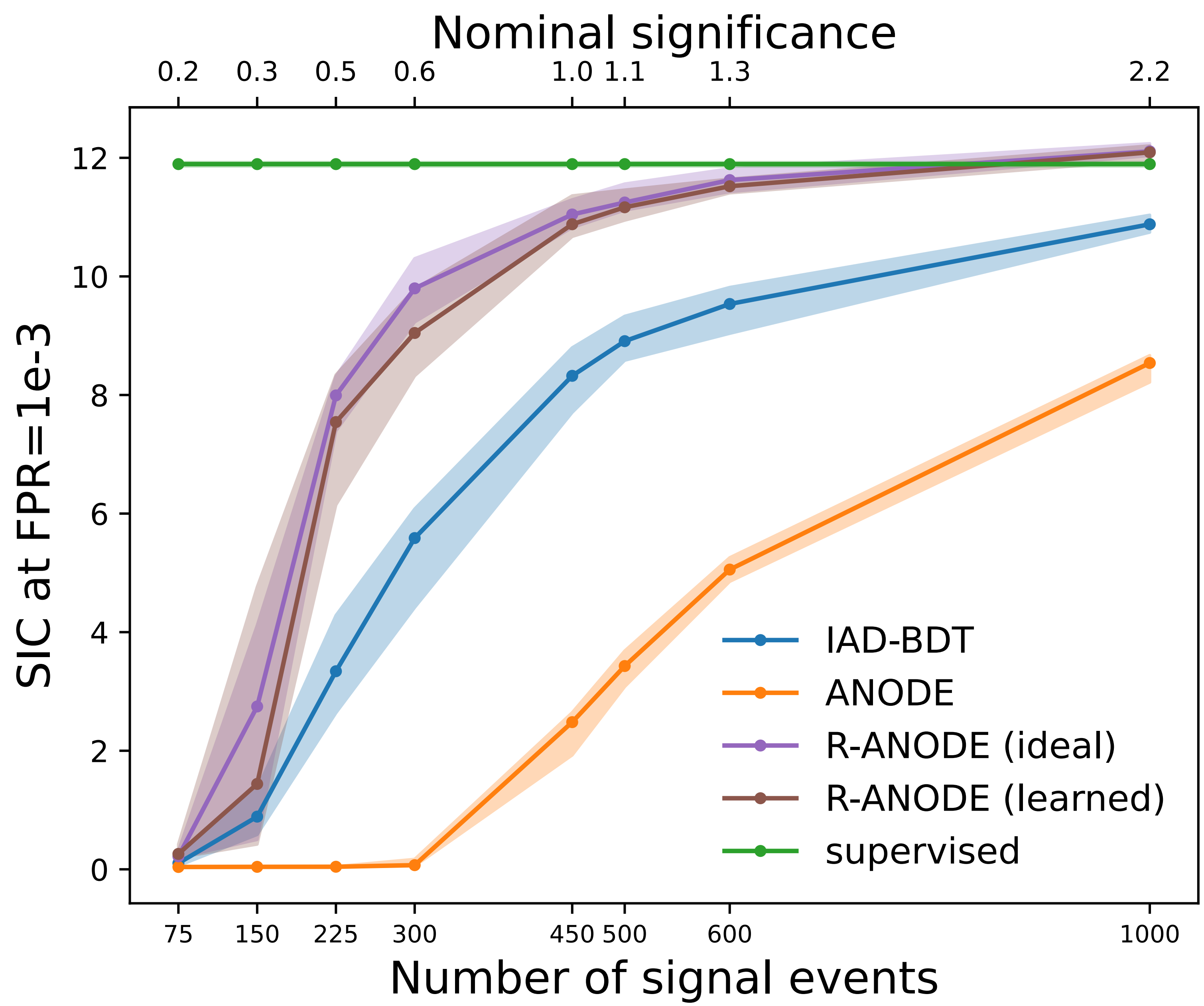
Anomaly score $$R(x,m) = \frac{p_{\rm sig}(x,m)}{p_{\rm bg}(x,m)}$$

Das, **GK**, Shih 2312.11629

R-ANODE

R-ANODE outperforms idealised anomaly detector (upper limit of CATHODE)

Work both with assuming known w, and without

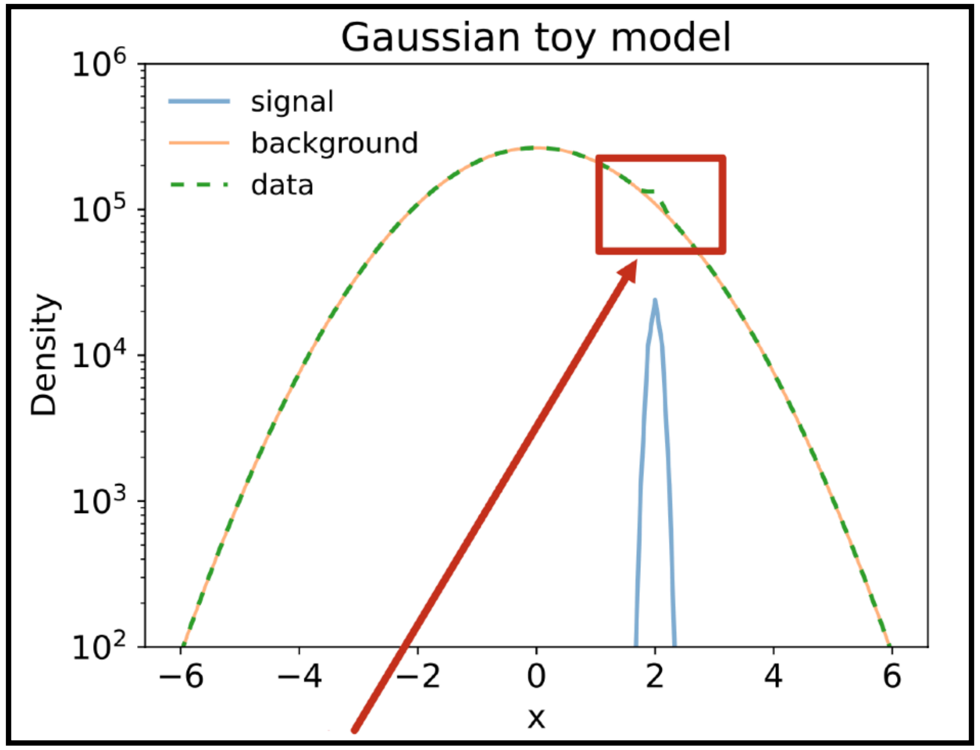


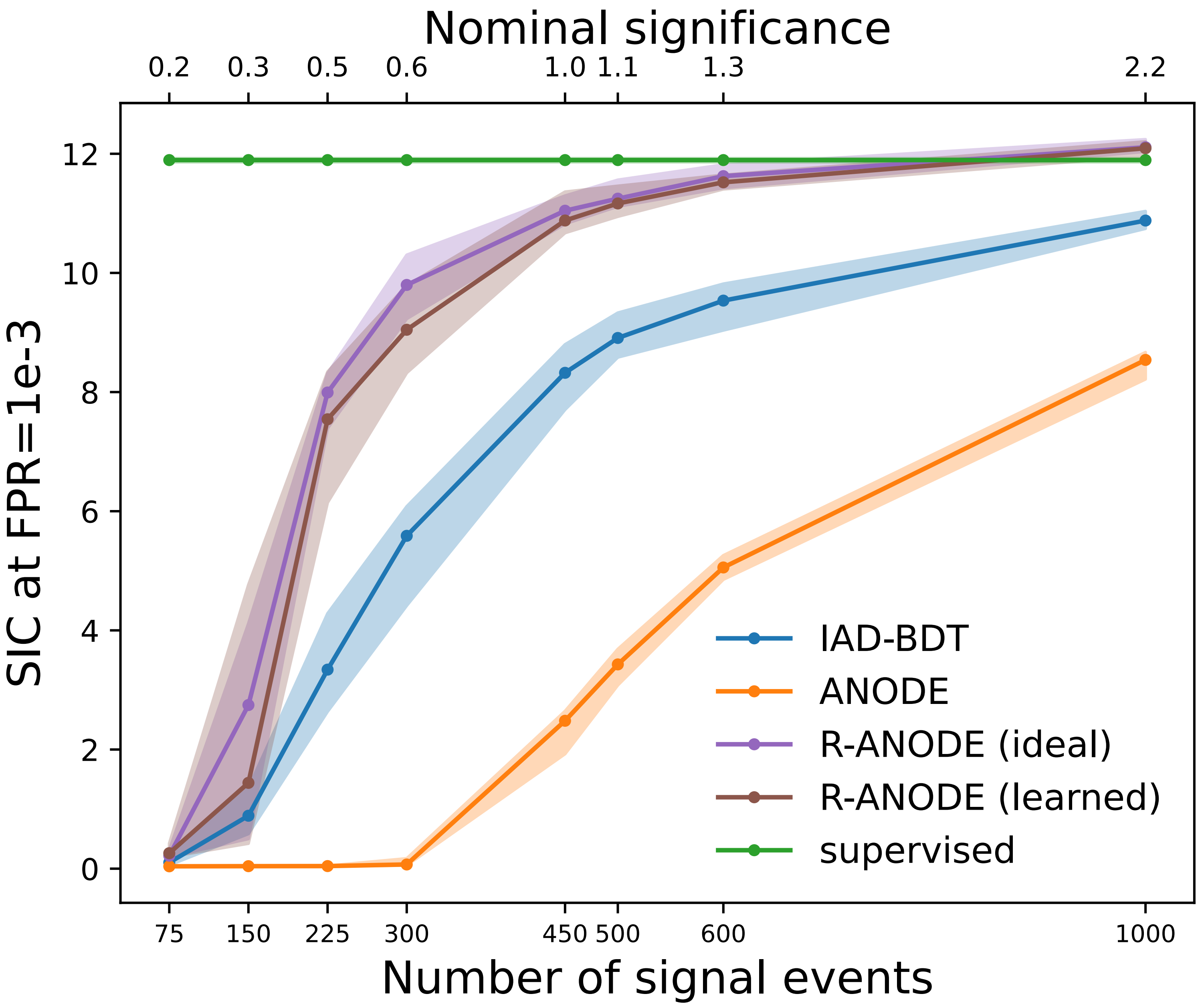
Das, **GK**, Shih, 2312.11629

R-ANODE

R-ANODE outperforms idealised anomaly detector (upper limit of CATHODE)

Work both with assuming known w, and without





Why? Assume fixed background, just need to fit extra peak

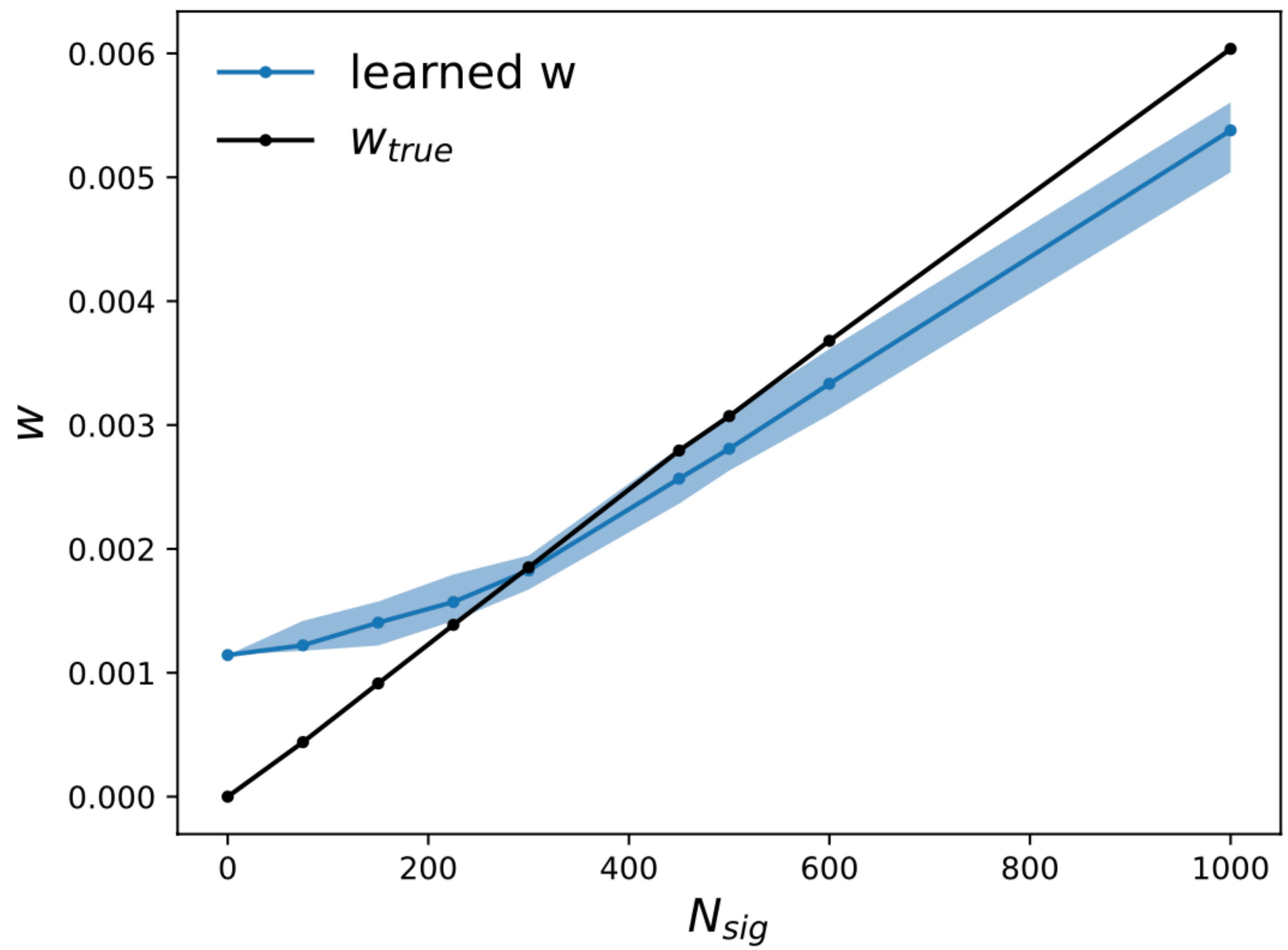
Das, **GK**, Shih, 2312.11629; Das @ ML4Jets '23

R-ANODE

R-ANODE outperforms idealised anomaly detector (upper limit of CATHODE)

Work both with assuming known w, and without

How good is the learned w?



Selection bias at low w, otherwise good.

Das, **GK**, Shih, 2312.11629

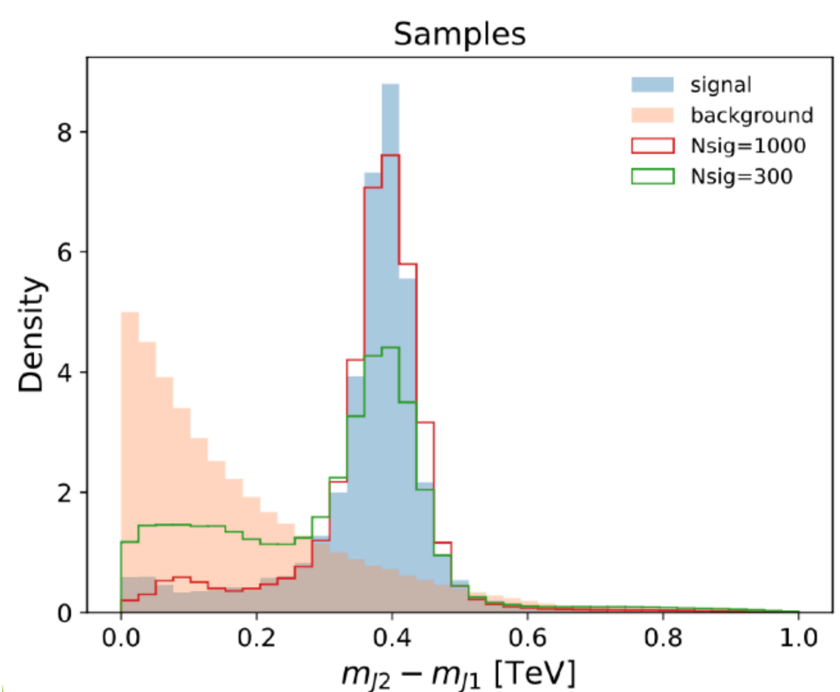
R-ANODE

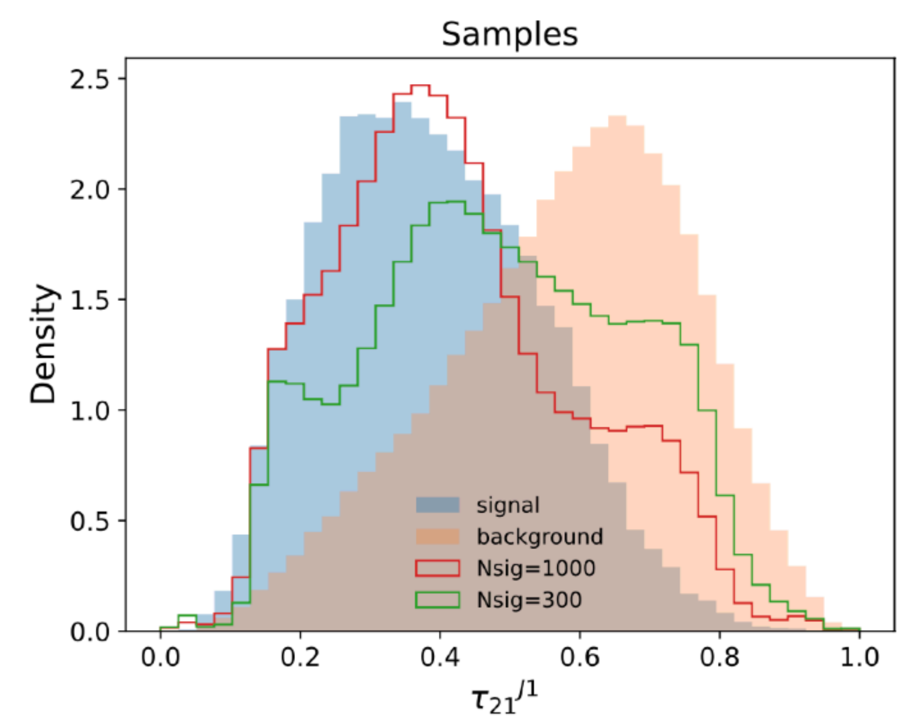
R-ANODE outperforms idealised anomaly detector (upper limit of CATHODE)

Work both with assuming known w, and without

How good is the learned w?

Can we interpret p_{Sig}?

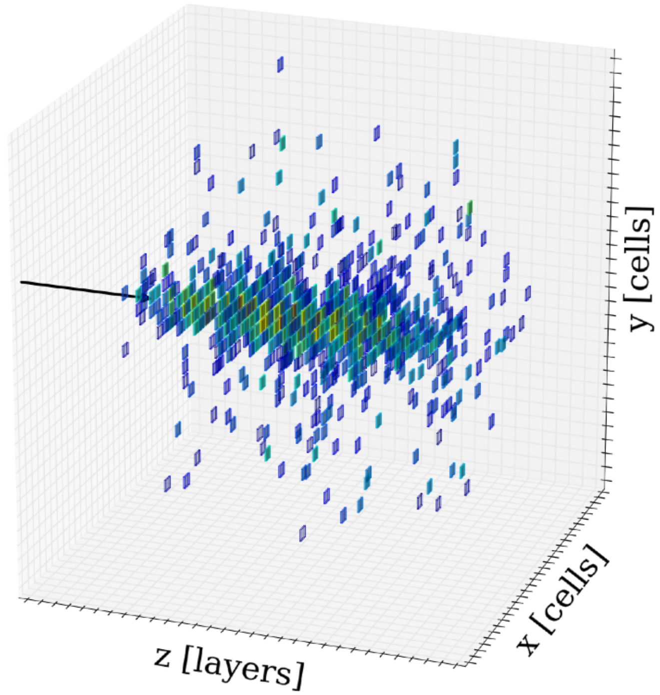




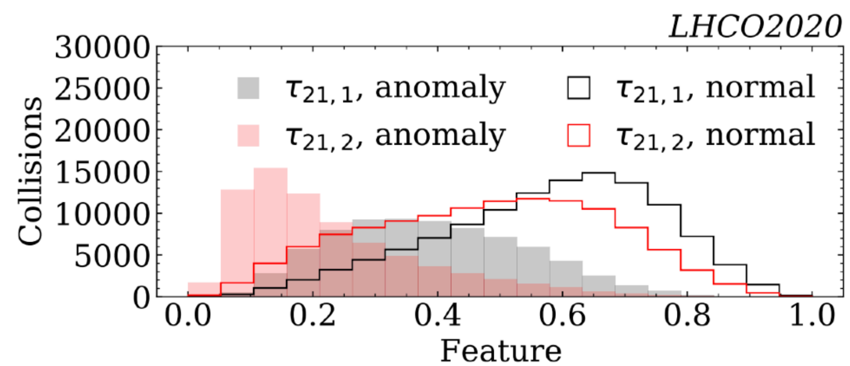
Yes!

Das, **GK**, Shih, 2312.11629

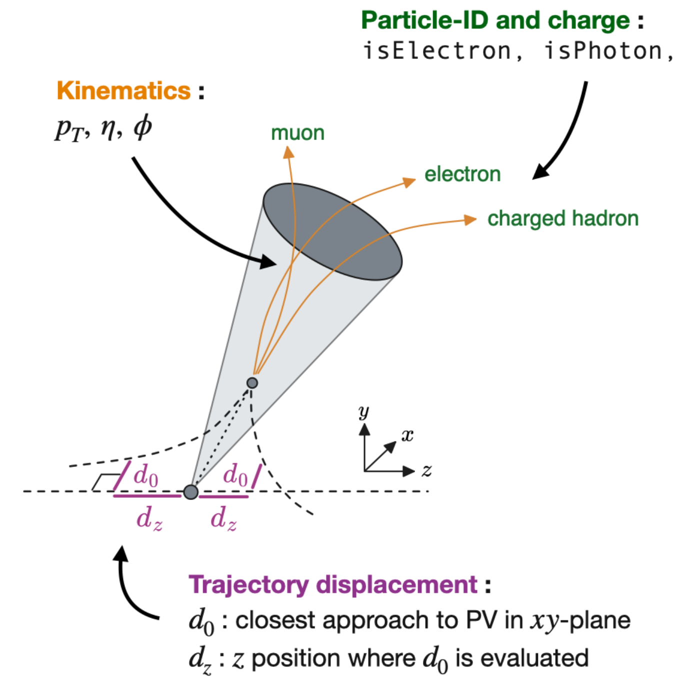
Why generative models?



Showers in complex high-resolution calorimeters

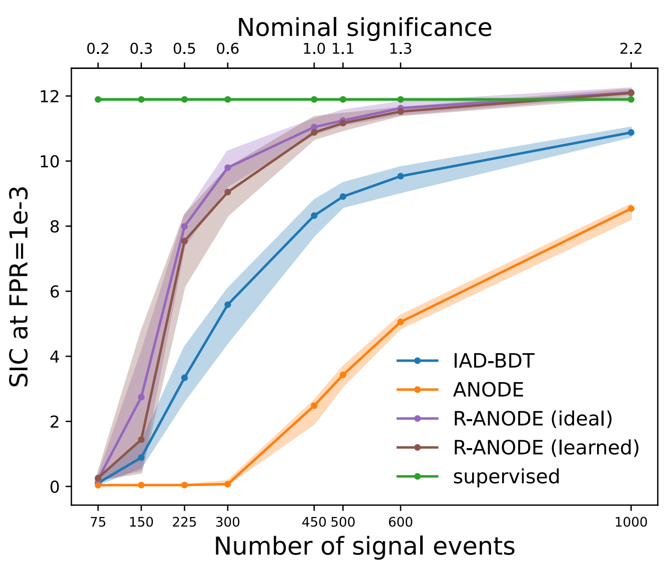


High-level jet features



Jet constituents

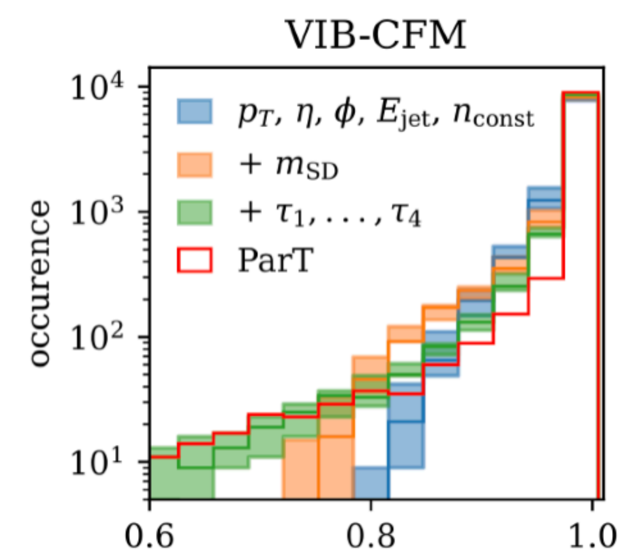
$$p(x)$$



Detect anomalies

Sample $X_i \sim p(x)$ to generate datapoints

Evaluate p(x) directly as likelihood



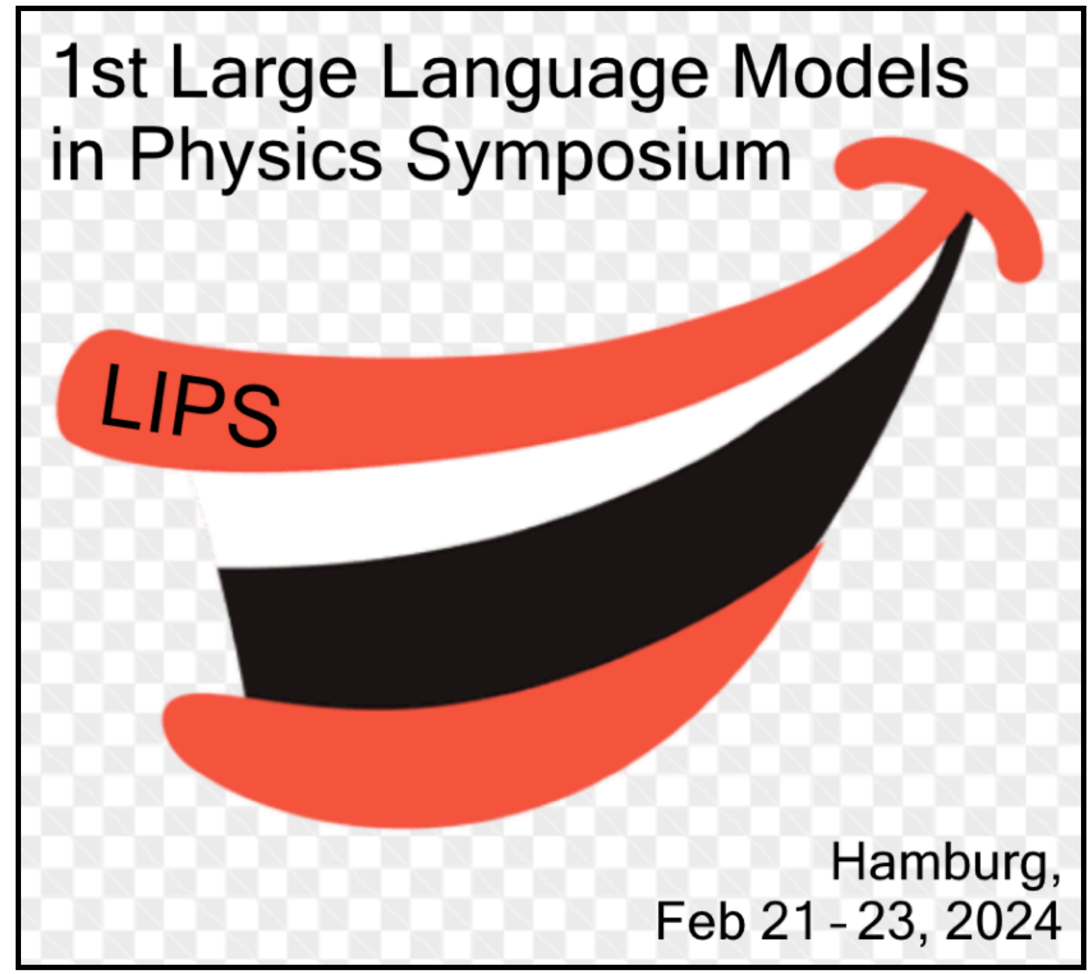
Classifier surrogates

Conclusions

Generative models have wide range of applications for simulation, background, estimation and as other surrogates

Recent progress (diffusion/flow matching + point clouds) allow modelling many high dimensional distributions

Models with tractable likelihood (i.e. normalising flow) enable further new applications



https://indico.desy.de/event/38849/

Thank you!