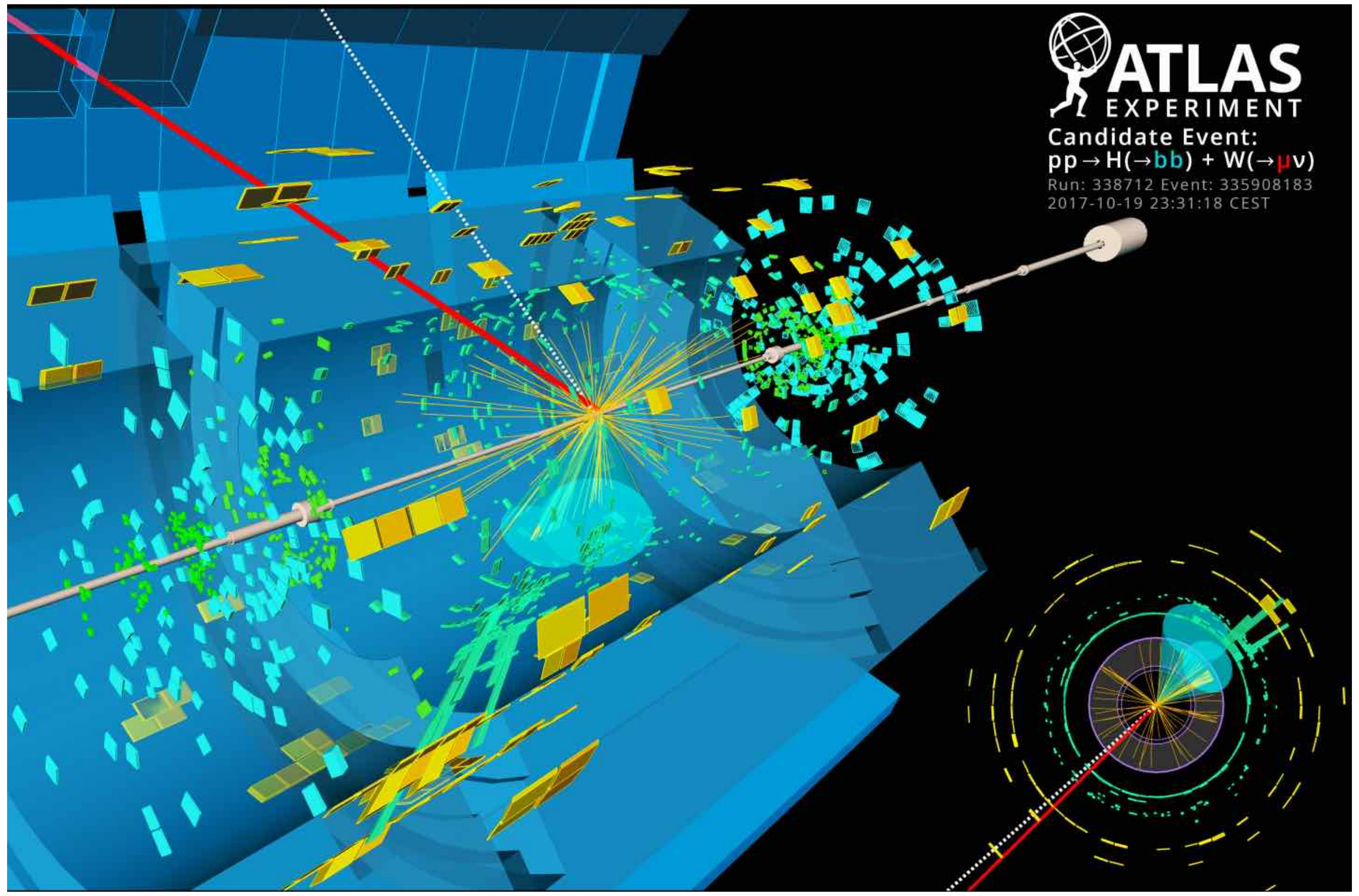
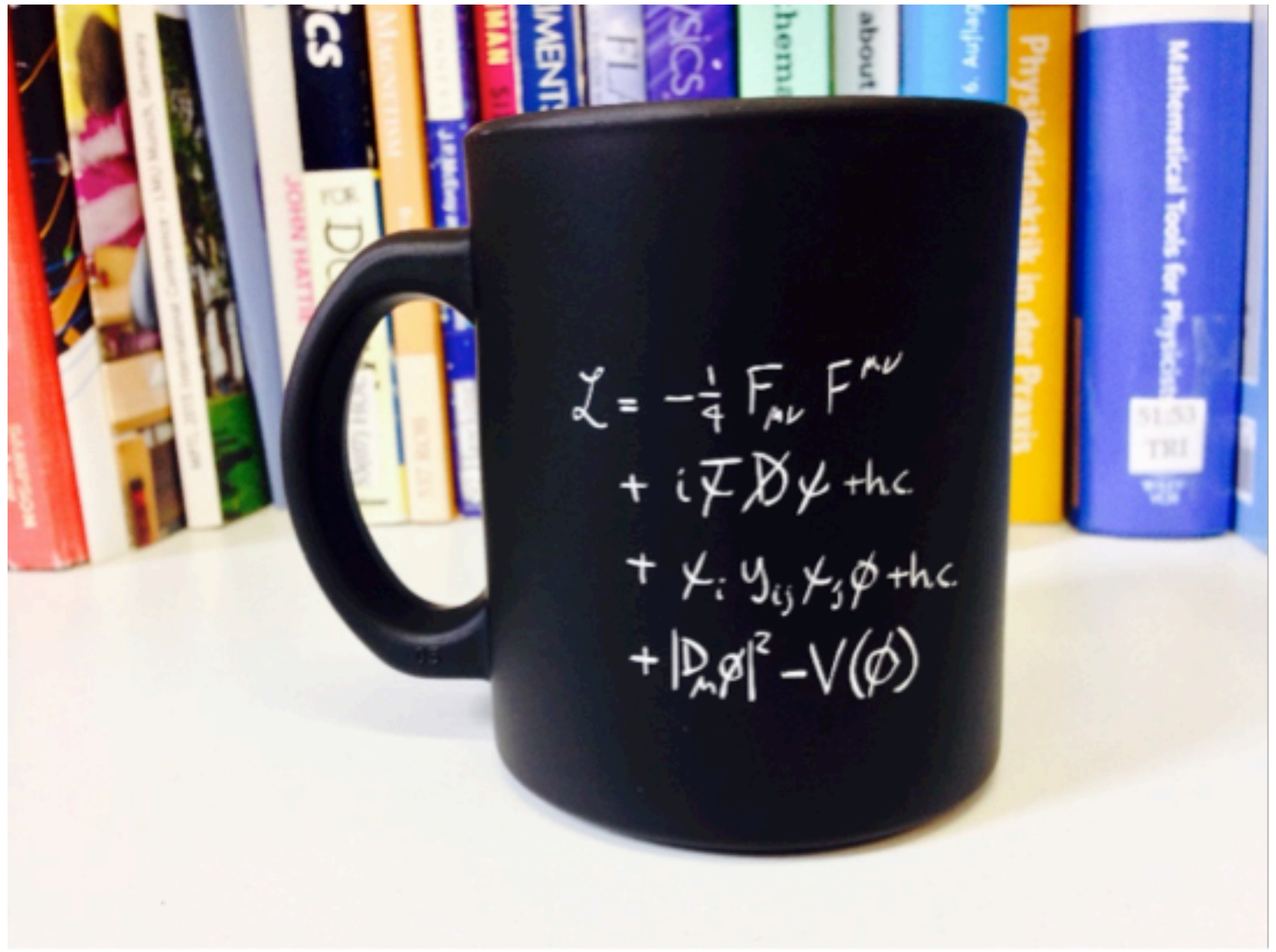
Machine Learning & LHC Event Generation

Mathematics, Physics and ML seminar series in Lisbon

Anja Butter, ITP Heidelberg

A short overview on LHC physics





Setting

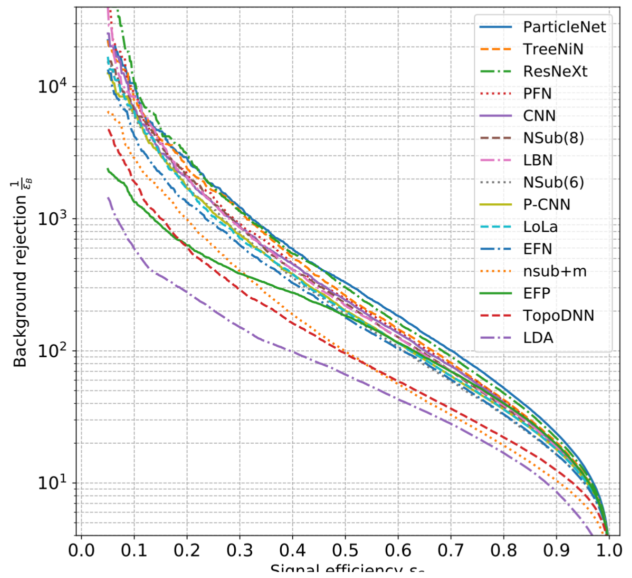
- Large Hadron Collider at CERN
- Proton collisions at 13 TeV
- **Huge** dataset ~1Pb/s before trigger selection

Goal

- Understand full dataset from **1st principles**
- Precision measurements of the SM
- Find signs of new physics (eg dark matter)

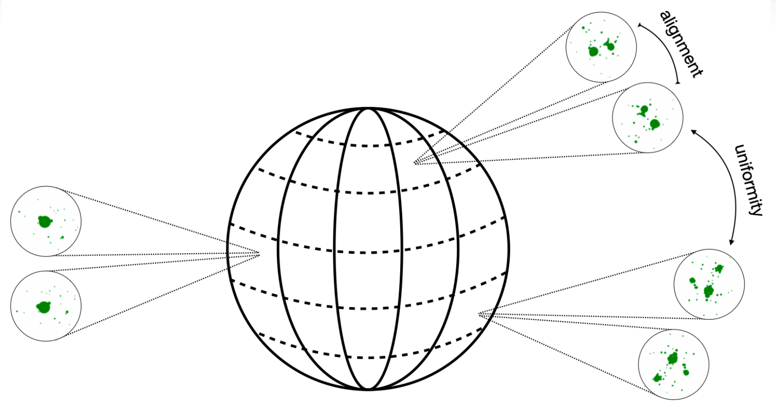
ML for big data in particle physics

Top tagging



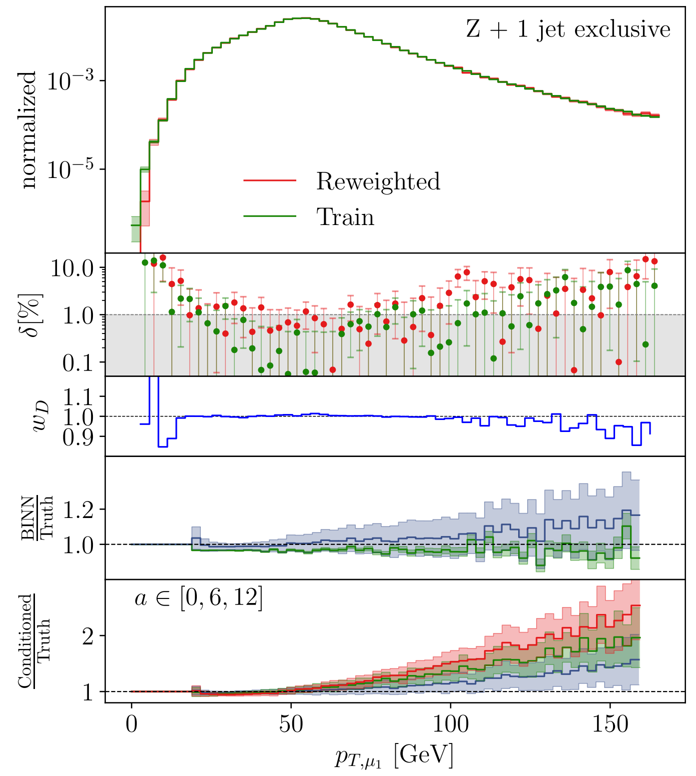
G. Kasieczka et al. [1902.09914]

Anomaly detection



B. Dillon et al. [2108.04253]

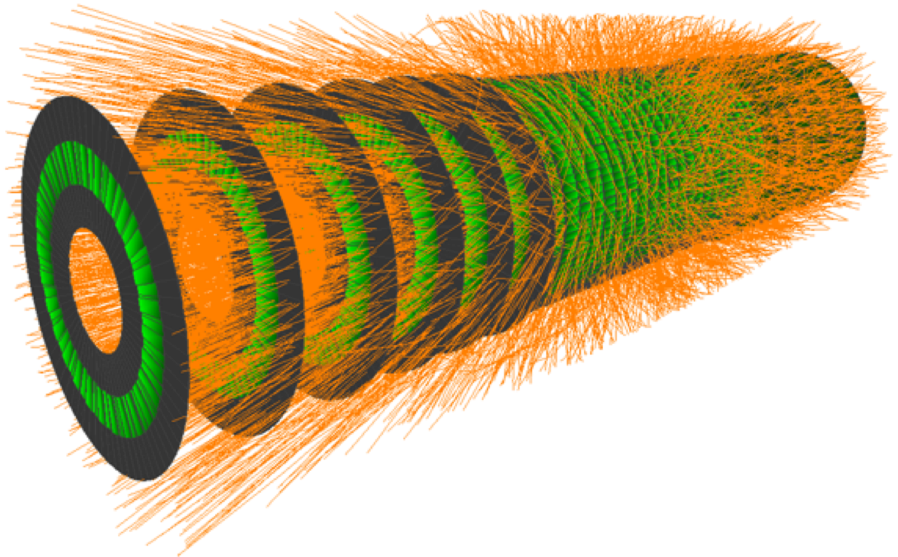
Event generation



A. Butter et al. [2110.13632]

Track reconstruction

Kaggle challenge



Classification

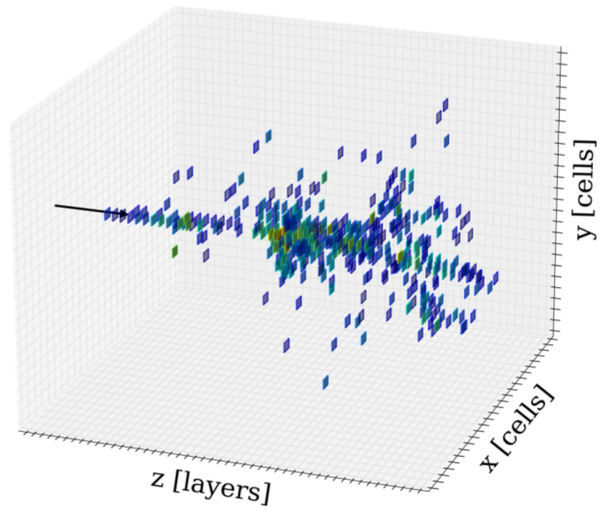
Generative models

Graph networks

Bayesian networks

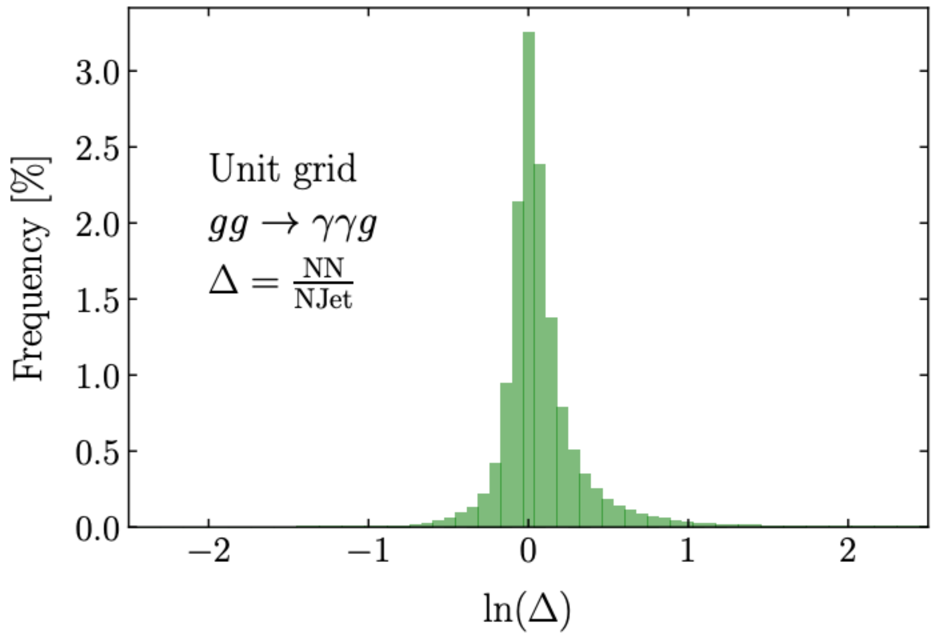
Regression

Detector simulation



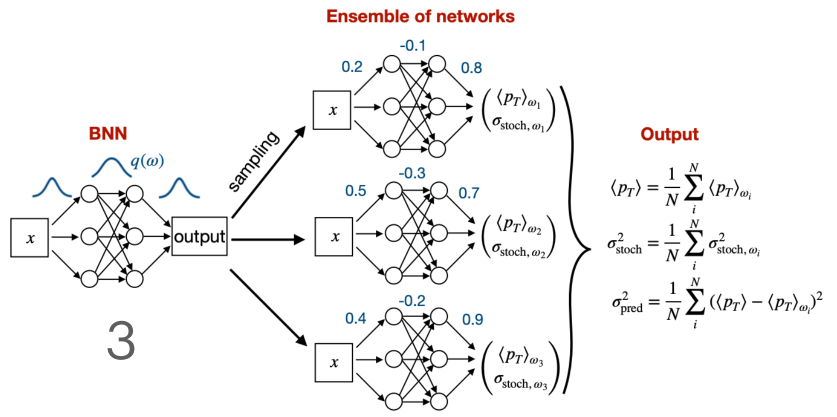
E. Buhmann et al. [2112.09709]

Amplitude estimation



J. Aylett-Bullock, et al. [2106.09474]

Jet calibration & uncertainties

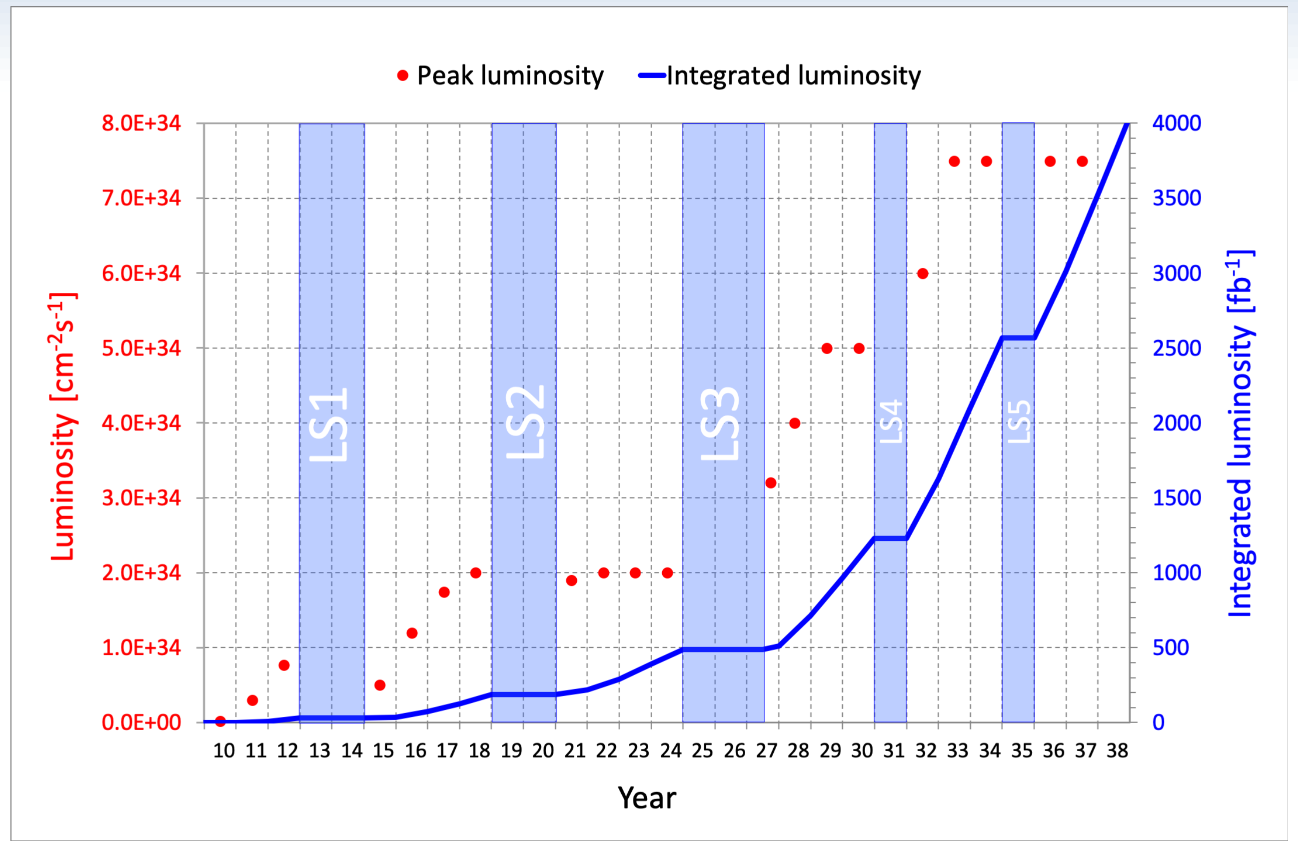


G. Kasieczka et al. [2003.11099]

Complete citations $\mathcal{O}(800)$
https://iml-wg.github.io/HEPML-LivingReview/

Open questions towards HL-LHC

A biased selection



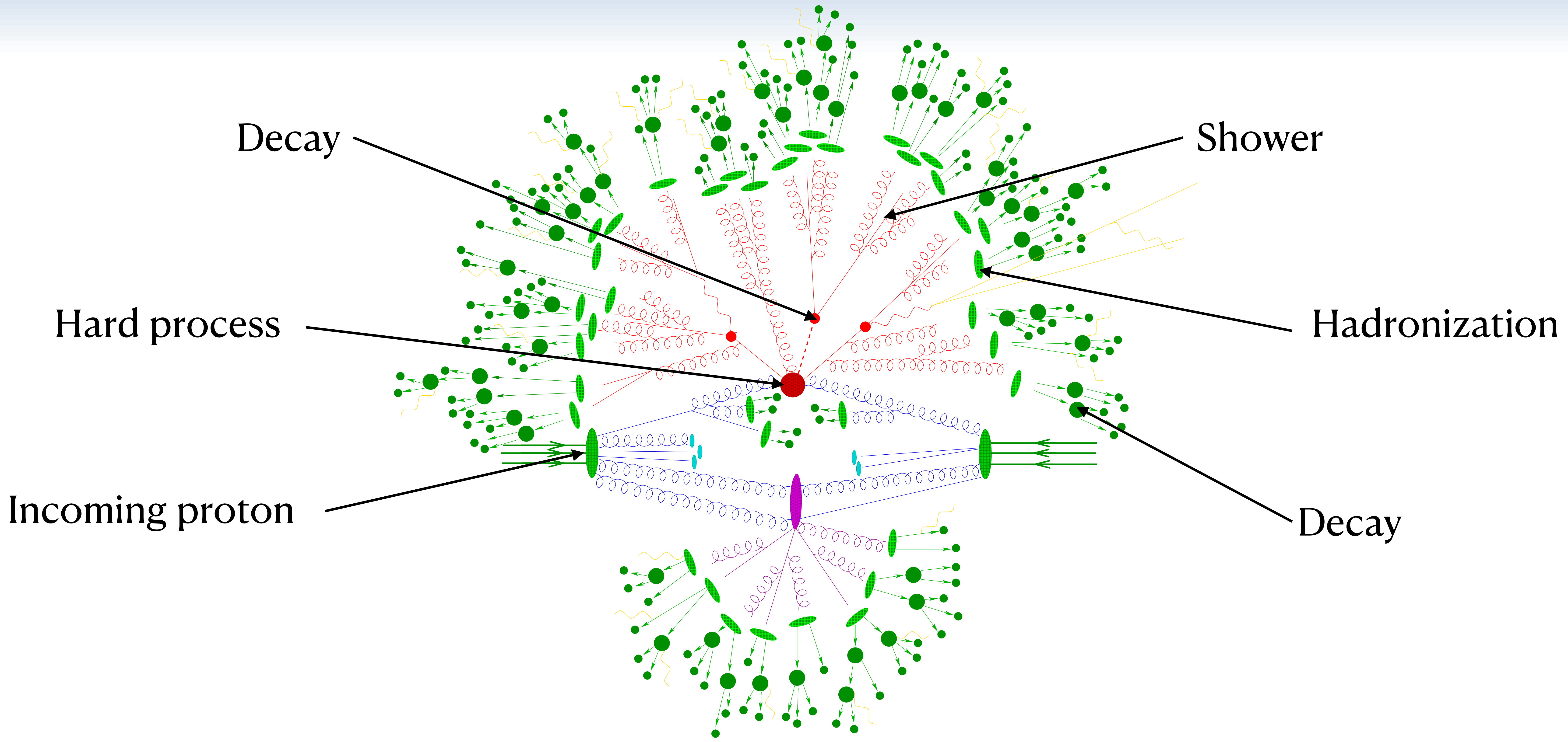
- Facing **25 times** the amount of data
- What do we need to understand the data? (*read:* find new physics)

- **Precision predictions**
  - Higher order amplitudes
  - Event generation
  - Shower
  - Detector simulation

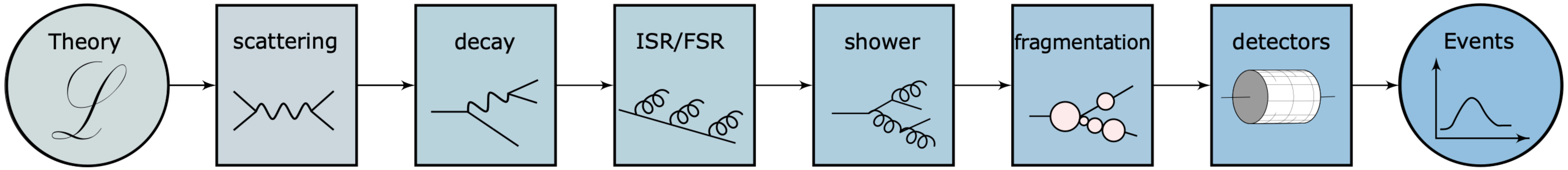
- **Optimized analysis for high-dimensional data**
  - Likelihood free inference
    - Optimal Observables, Unfolding
  - Anomaly detection
  - Uncertainty treatment

Problems beyond supervised classification/regression → How can machine learning help?

Event generation at the LHC



Simulation tool chain



Monte carlo event generation

1. Generate phase space points

$\rightarrow$ set of four-momenta p_i

2. Calculate event weight

$$w_{\text{event}} = f(x_1, Q^2) f(x_1, Q^2) \times \mathcal{M}(x_1, x_2, p_1, \ldots, p_n) \times J(p_i(r))$$

PDF

Matrix element †

Phase space mapping

3. Unweighting †

keep events with $\frac{w_i}{w_{\max}} > r \in [0,1]$

† Bottlenecks

1. Slow **matrix element** calculation
   - Complexity grows exponentially with
     - # final state particles
     - Precision (LO, NLO, NNLO, ...)
2. Low **unweighting** efficiency
   - Discard most events if $w_i \ll w_{\max}$
   - Optimize phase space mapping

     ➡ $J(p_i(r)) = (f \times \mathcal{M})^{-1}$

Approximating Amplitudes

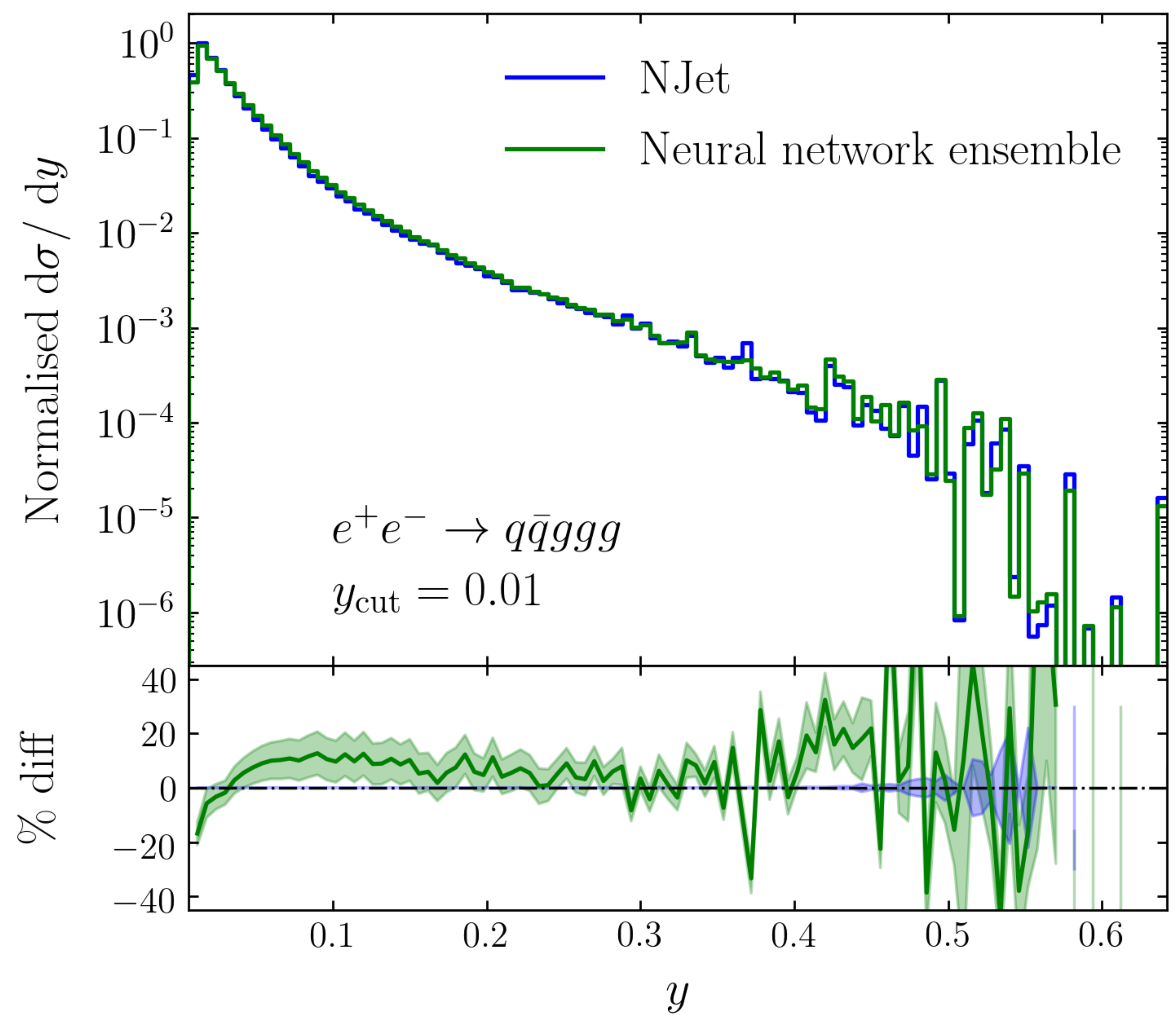
- Approximate matrix element with NN

- Regression problem
- Minimize distance between prediction and truth

$\Rightarrow \mathscr{L} = \left(NN(p_i) - \mathscr{M}(p_i)\right)^2$

+ Generalization of interpolation
+ Better scaling than grids for large dimensions

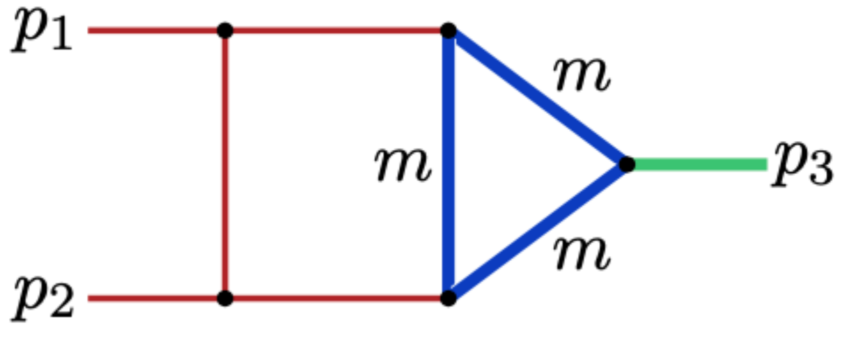
- Open questions
  - Limited precision
  - Overtraining vs interpolation



Badger, Bullock [2002.07516]

Multi-loop calculations with NNs

Precision predictions based on loop diagrams



Analytic expression for loop amplitude

$$G = \int_{-\infty}^{\infty} \left(\prod_{l=1}^{L} \frac{\mathrm{d}^D k_l}{i\pi^{\frac{D}{2}}} \right) \prod_{j=1}^{N} \frac{1}{(q_j^2 - m_j^2 + i\delta)^{\nu_j}}$$

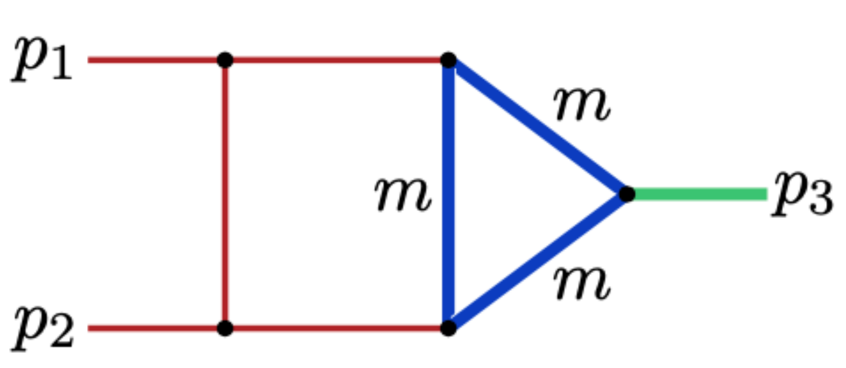
$$= \int_0^1 \prod_{j=1}^{N-1} \mathrm{d}x_j \, x_j^{\nu_j - 1} \frac{U^{\nu-(L+1)D/2}}{F^{\nu-LD/2}} = \int_0^1 \prod_{j=1}^{N-1} \mathrm{d}x_j \, I(\overrightarrow{x})$$

Rewrite with
Feynman parameters

Still contains singularities

Multi-loop calculations with NNs

Precision predictions based on loop diagrams



Analytic expression for loop amplitude

$$G = \int_{-\infty}^{\infty} \left(\prod_{l=1}^{L} \frac{\mathrm{d}^D k_l}{i\pi^{\frac{D}{2}}} \right) \prod_{j=1}^{N} \frac{1}{(q_j^2 - m_j^2 + i\delta)^{\nu_j}}$$

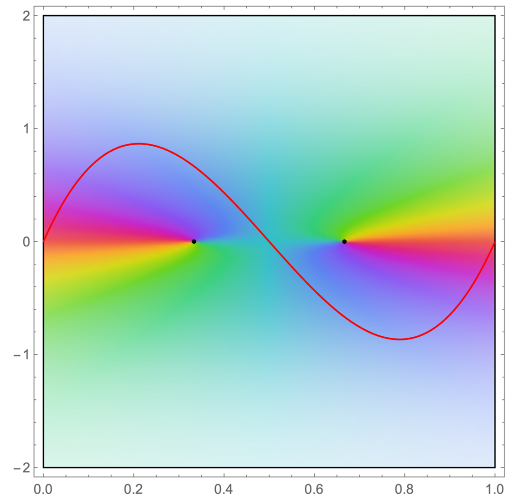
$$= \int_0^1 \prod_{j=1}^{N-1} \mathrm{d}x_j \, x_j^{\nu_j - 1} \frac{U^{\nu-(L+1)D/2}}{F^{\nu-LD/2}} = \int_0^1 \prod_{j=1}^{N-1} \mathrm{d}x_j \; I(\vec{x})$$

Rewrite with
Feynman parameters

Still contains singularities

Solved by contour deformation due to Cauchy's theorem

$$\int_0^1 \prod_{j=1}^{N} \mathrm{d}x_j \, I(\vec{x}) = \int_0^1 \prod_{j=1}^{N} \mathrm{d}x_j \; \det\left(\frac{\partial \vec{z}(\vec{x})}{\partial \vec{x}} \right) I(\vec{z}(\vec{x}))$$



Optimal parametrization = minimal variance

Turn it into an ML Problem

Integration with normalizing flows

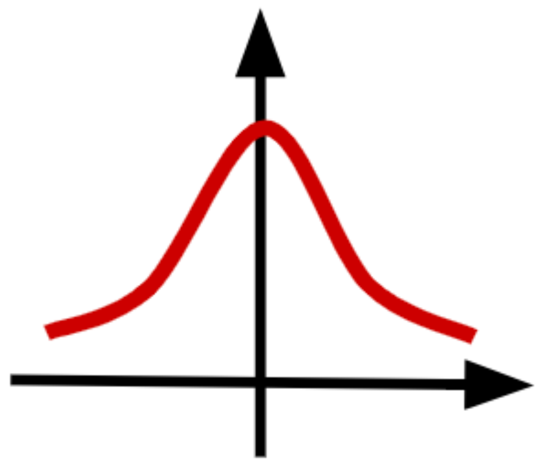
Numeric evaluation of integral $G = \int_0^1 \mathrm{d}x_j \; \det\left(\frac{\partial \vec{z}(\vec{x})}{\partial \vec{x}}\right) I(\vec{z}(\vec{x}))$

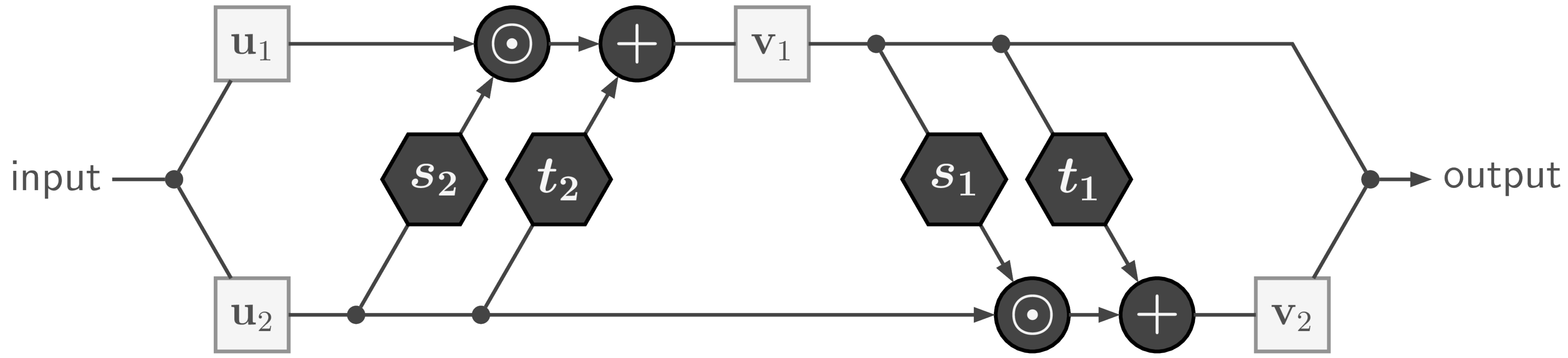
Parametrization $\rightarrow z = \mathrm{INN}(x)$

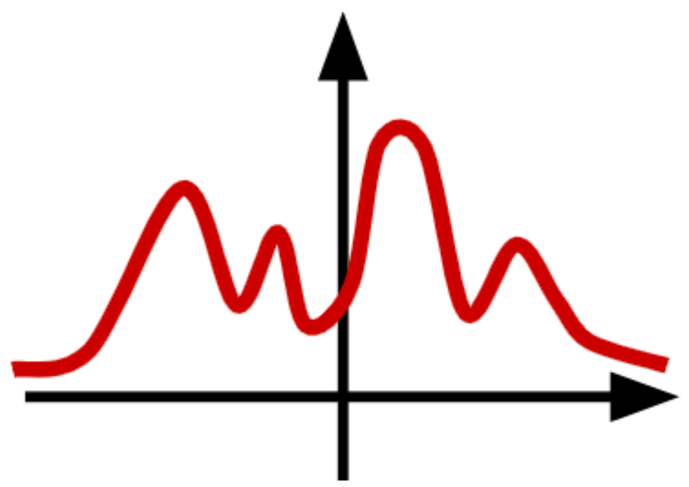
Minimize **variance** $\rightarrow$ loss $\mathscr{L} = \sigma_n^2 = \frac{1}{n-1} \sum_{i=1}^{n} \left| \det\left(\frac{\partial \vec{z}(\vec{x}_{(i)})}{\partial \vec{x}_{(i)}}\right) I(\vec{z}(\vec{x}_{(i)})) - \langle I \rangle \right|^2$

Normalizing flow networks

+ Bijective mapping
+ Tractable Jacobian
+ Combine many blocks



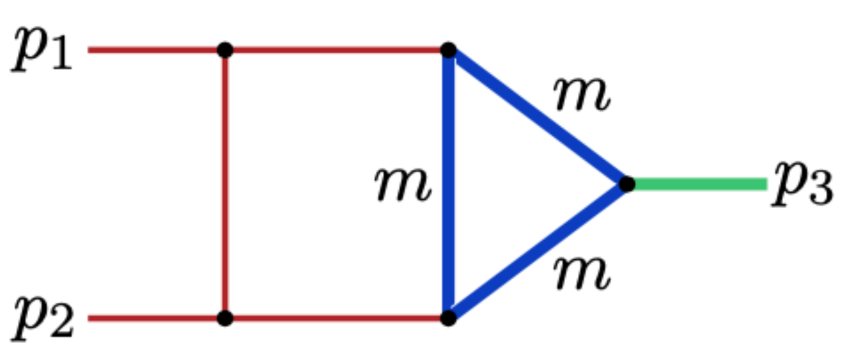




Multi-loop calculations with INNs

Profiting from the Jacobian

Precision predictions based on loop diagrams



Analytic expression for loop amplitude

$$G = \int_{-\infty}^{\infty} \left(\prod_{l=1}^{L} \frac{\mathrm{d}^D k_l}{i\pi^{\frac{D}{2}}} \right) \prod_{j=1}^{N} \frac{1}{(q_j^2 - m_j^2 + i\delta)^{\nu_j}}$$

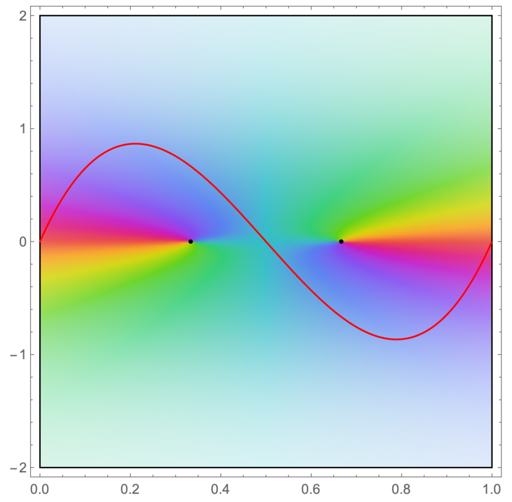
$$= \int_0^1 \prod_{j=1}^{N-1} \mathrm{d}x_j \, x_j^{\nu_j - 1} \frac{U^{\nu-(L+1)D/2}}{F^{\nu-LD/2}} = \int_0^1 \prod_{j=1}^{N-1} \mathrm{d}x_j \, I(\vec{x})$$

Rewrite with Feynman parameters

Still contains singularities

Solved by contour deformation due to Cauchy's theorem

$$\int_0^1 \prod_{j=1}^{N} \mathrm{d}x_j \, I(\vec{x}) = \int_0^1 \prod_{j=1}^{N} \mathrm{d}x_j \, \det\left(\frac{\partial \vec{z}(\vec{x})}{\partial \vec{x}}\right) I(\vec{z}(\vec{x}))$$

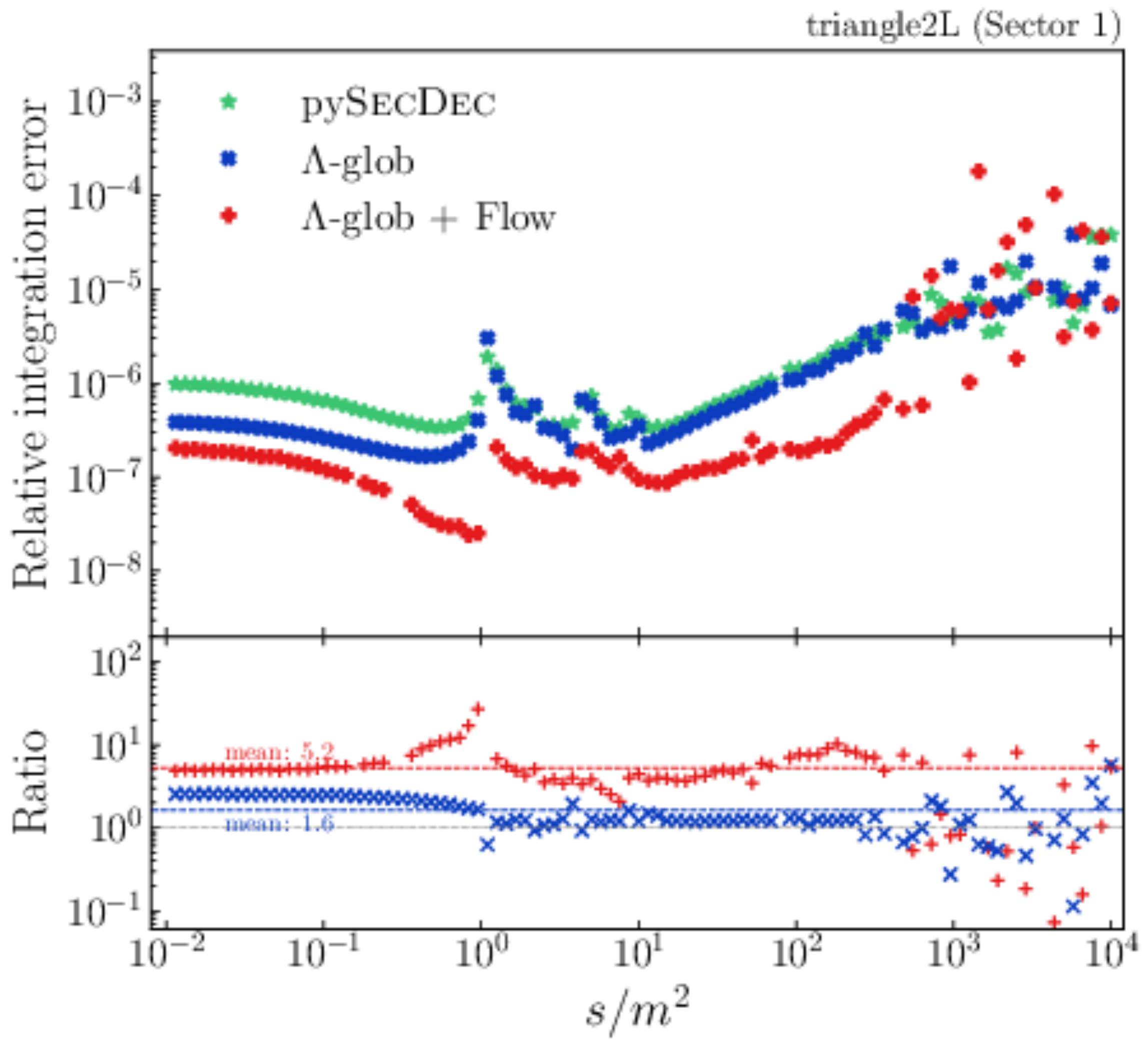


Optimal parametrization = minimal variance

Turn it into an ML Problem

Parametrization $\rightarrow z = \mathrm{INN}(x)$

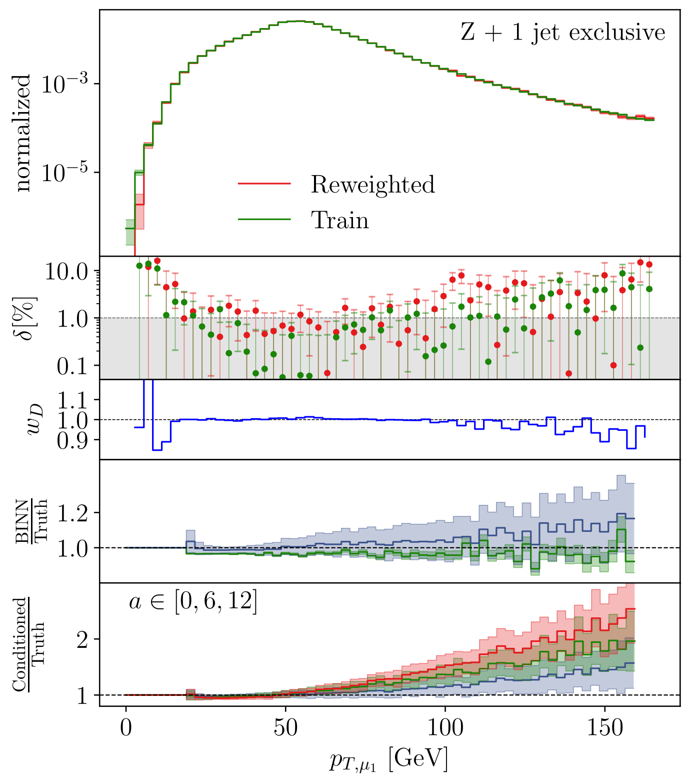
Variance $\rightarrow \mathscr{L}$



R. Winterhalder, et al. [2112.09145]

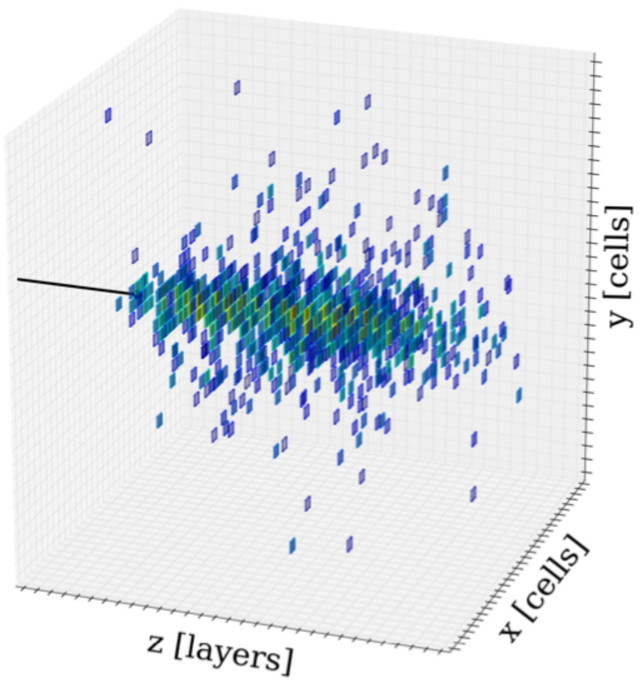
Forward simulations with generative networks

Event generation

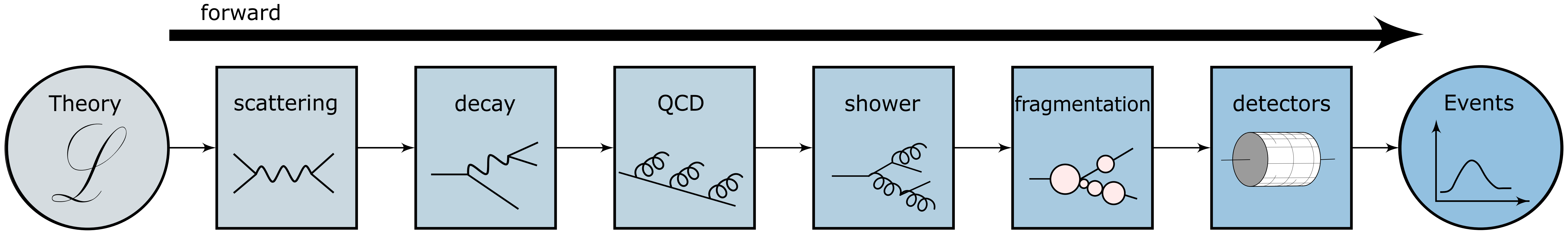


→ Otten et al.
→ Gao et al.
→ Bothmann et al.
→ Stienen et al.
→ AB, et al.
→ and many more

Detector simulation



→ CaloGAN by M. Paganini et al.
→ BIBAE by E. Buhman, S. Diefenbacher et al.
→ CaloFlow by C. Krause , D. Shih
→ and many more



Applications

- Phase space sampling
- End to end learning
- Data compression
- Amplification

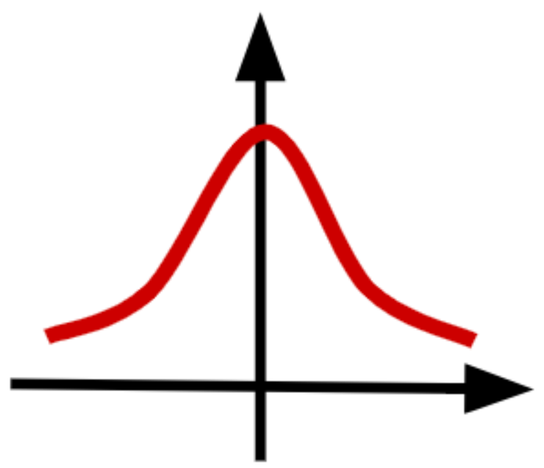
Phase space sampling with generative networks (GAN, VAE, NF)

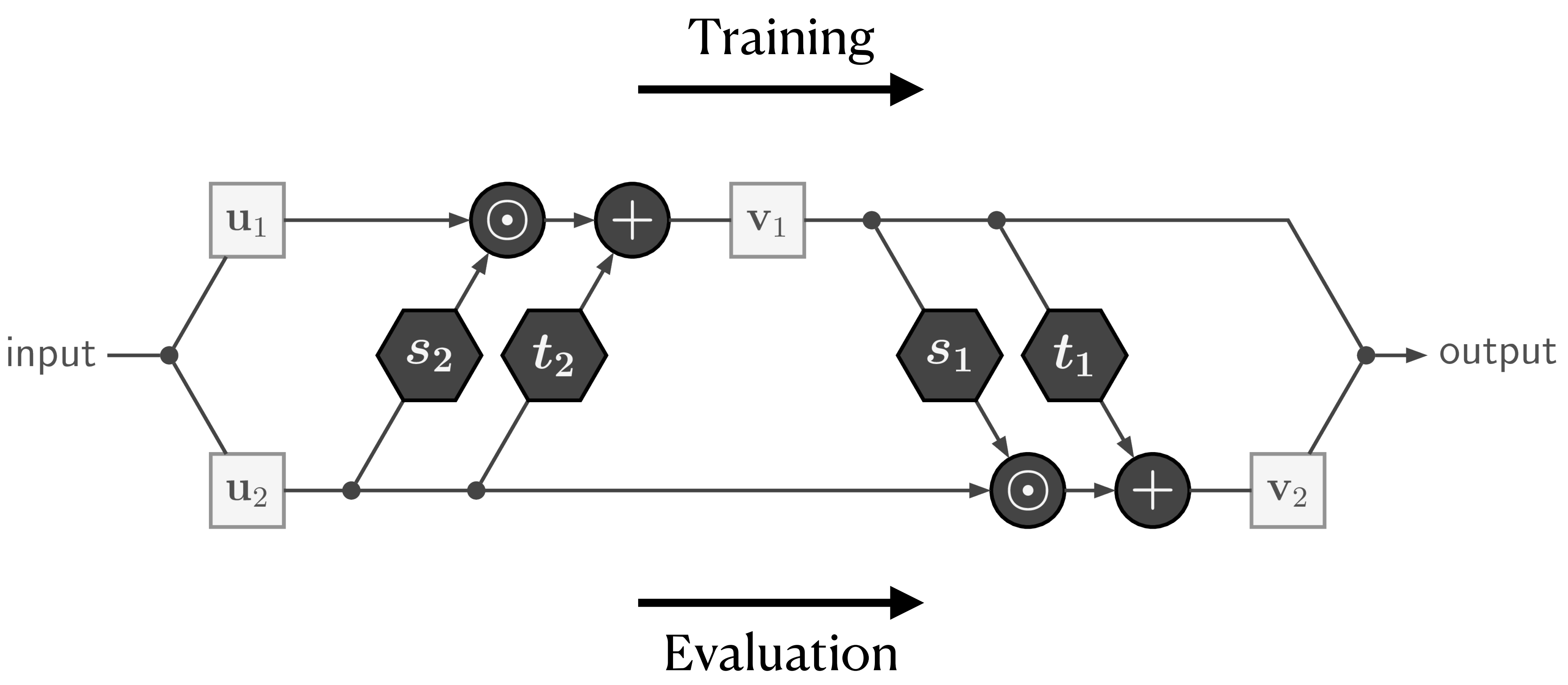
Particularly promising architecture
→ Normalizing flows

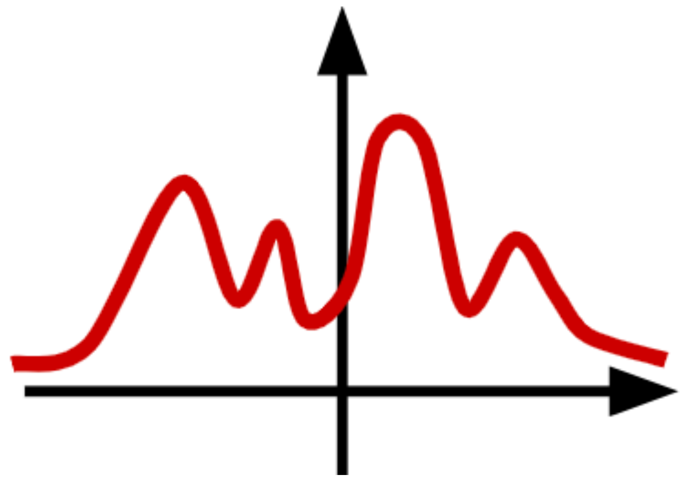
Normalizing flows

Invertible networks for complex transformations

+ Bijective mapping
+ Tractable Jacobian → $p_x(x) = p_z(z) \cdot J_{NN}$
+ Fast evaluation in both direction







Training on density $t(x)$
→ Minimize difference

$$\begin{aligned}\mathscr{L} &= \log p_x(x)/t(x) \\ &= \log p_z(z(x))\, J_{NN} / t(x)\end{aligned}$$

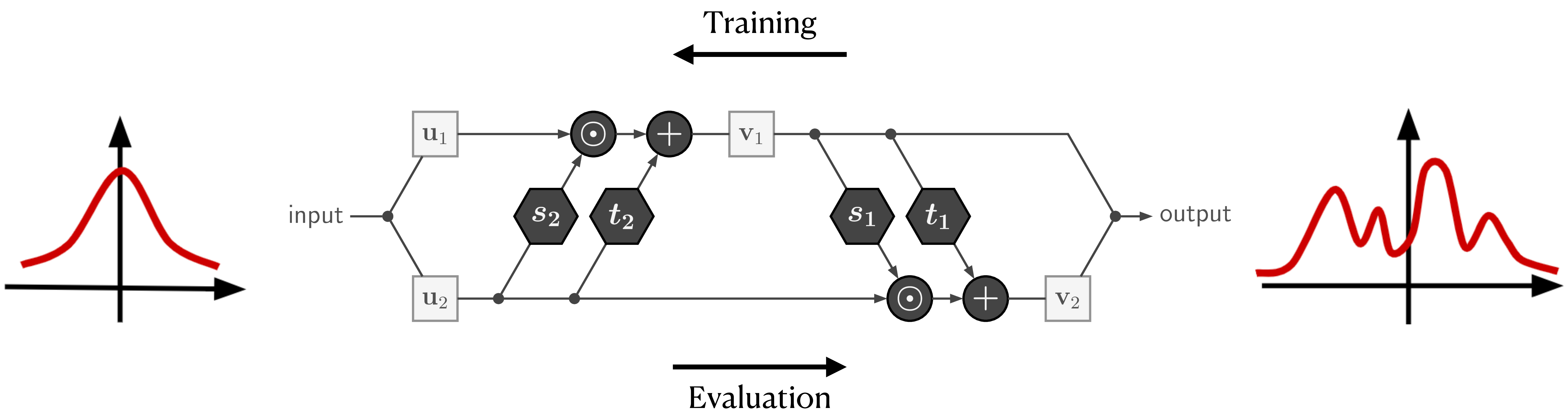
Training on samples x
→ Maximize the log-likelihood

$$\begin{aligned}\mathscr{L} &= \log p(\theta | x) \\ &= \log p(z | \theta) + \log J_{NN} + p(\theta)\end{aligned}$$

Normalizing flows

Invertible networks for complex transformations

+ Bijective mapping
+ Tractable Jacobian → $p_x(x) = p_z(z) \cdot J_{NN}$
+ Fast evaluation in both direction



Training on density $t(x)$

→ Minimize difference

$$\begin{aligned}\mathscr{L} &= \log p_x(x)/t(x) \\ &= \log p_z(z(x))\, J_{NN} \,/\, t(x)\end{aligned}$$

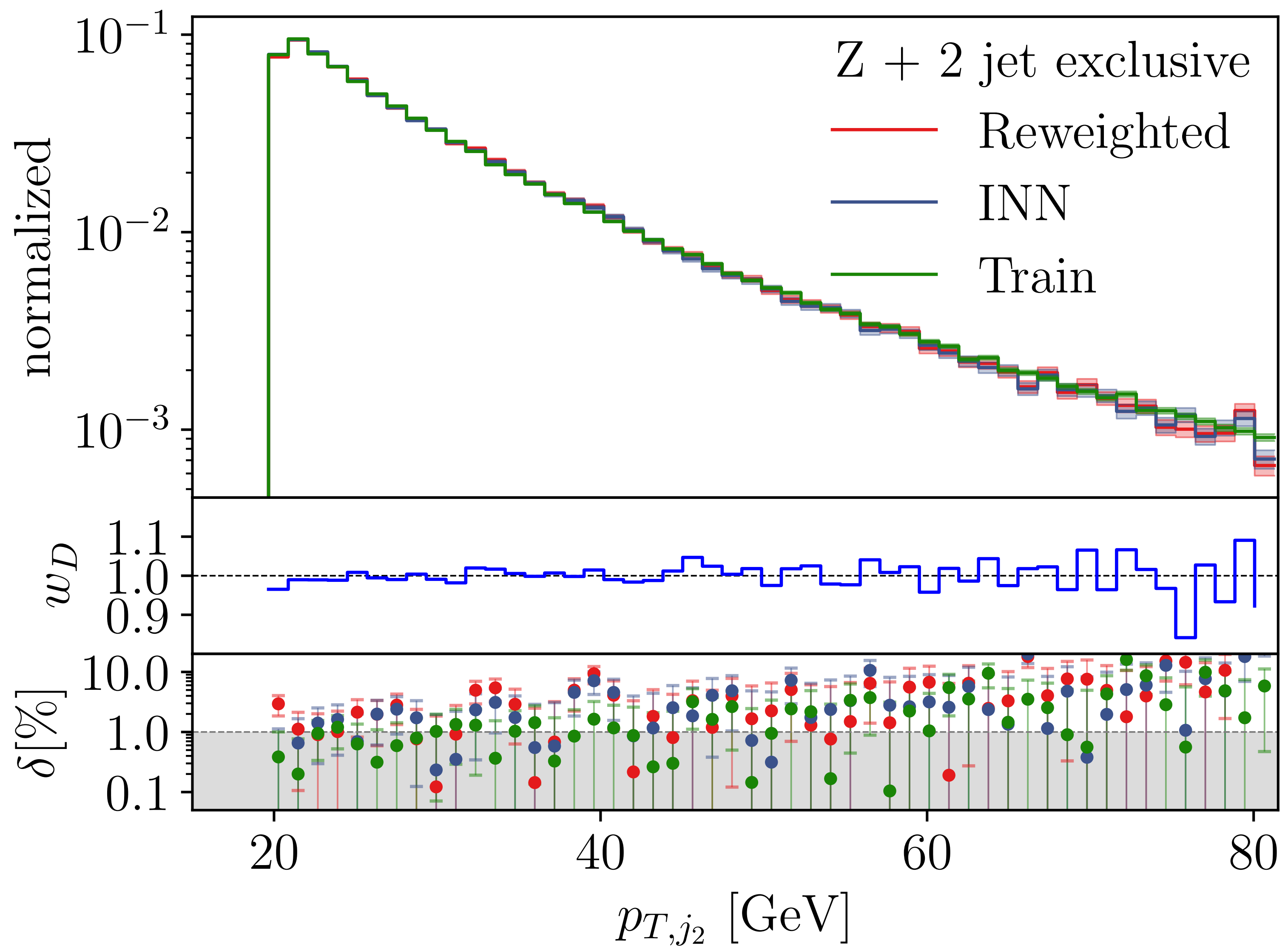
Training on samples x

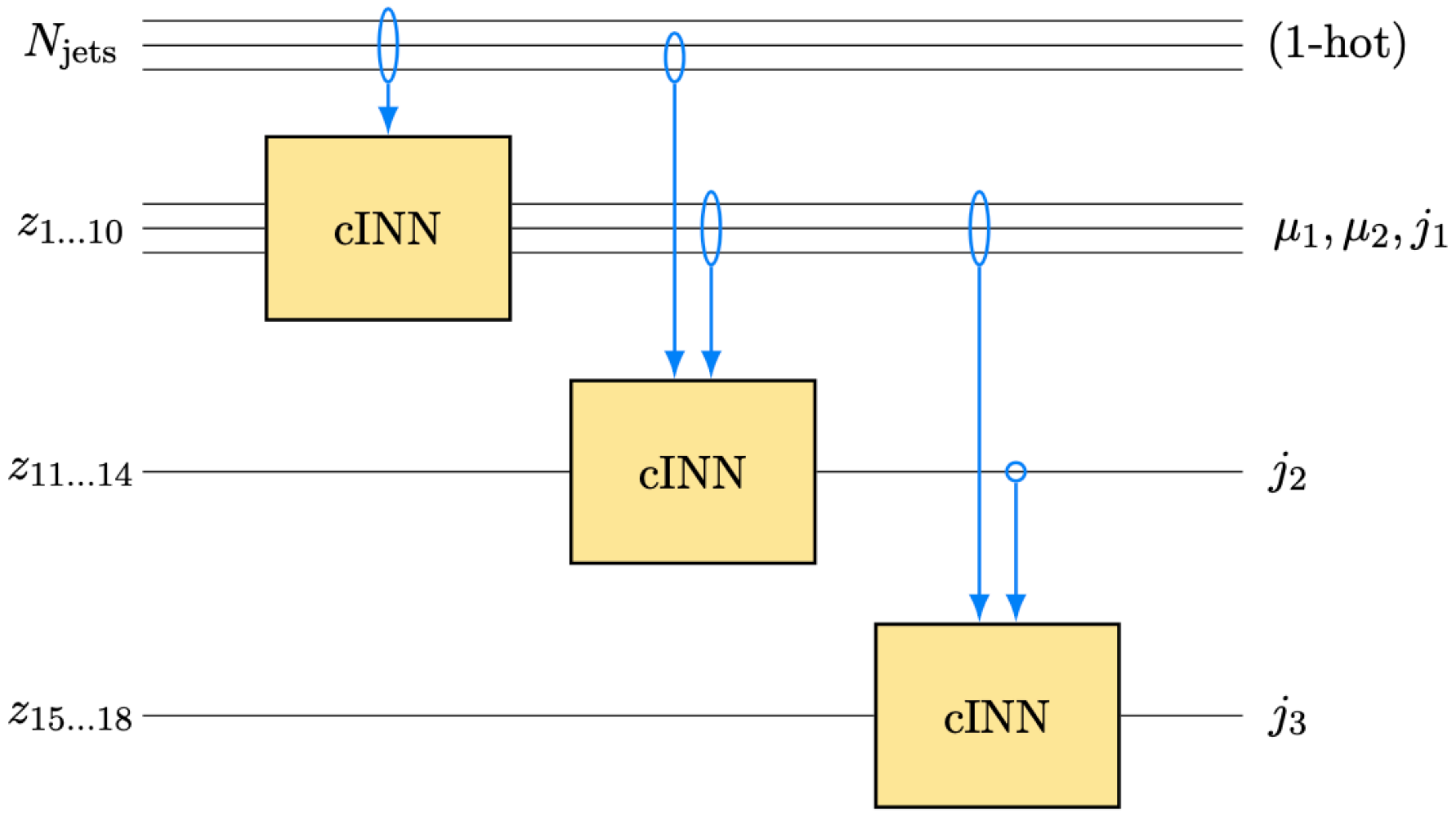
→ Maximize the log-likelihood

$$\begin{aligned}\mathscr{L} &= \log p(\theta \,|\, x) \\ &= \log p(z \,|\, \theta) + \log J_{NN} + p(\theta)\end{aligned}$$

Putting flows to work

Event generation



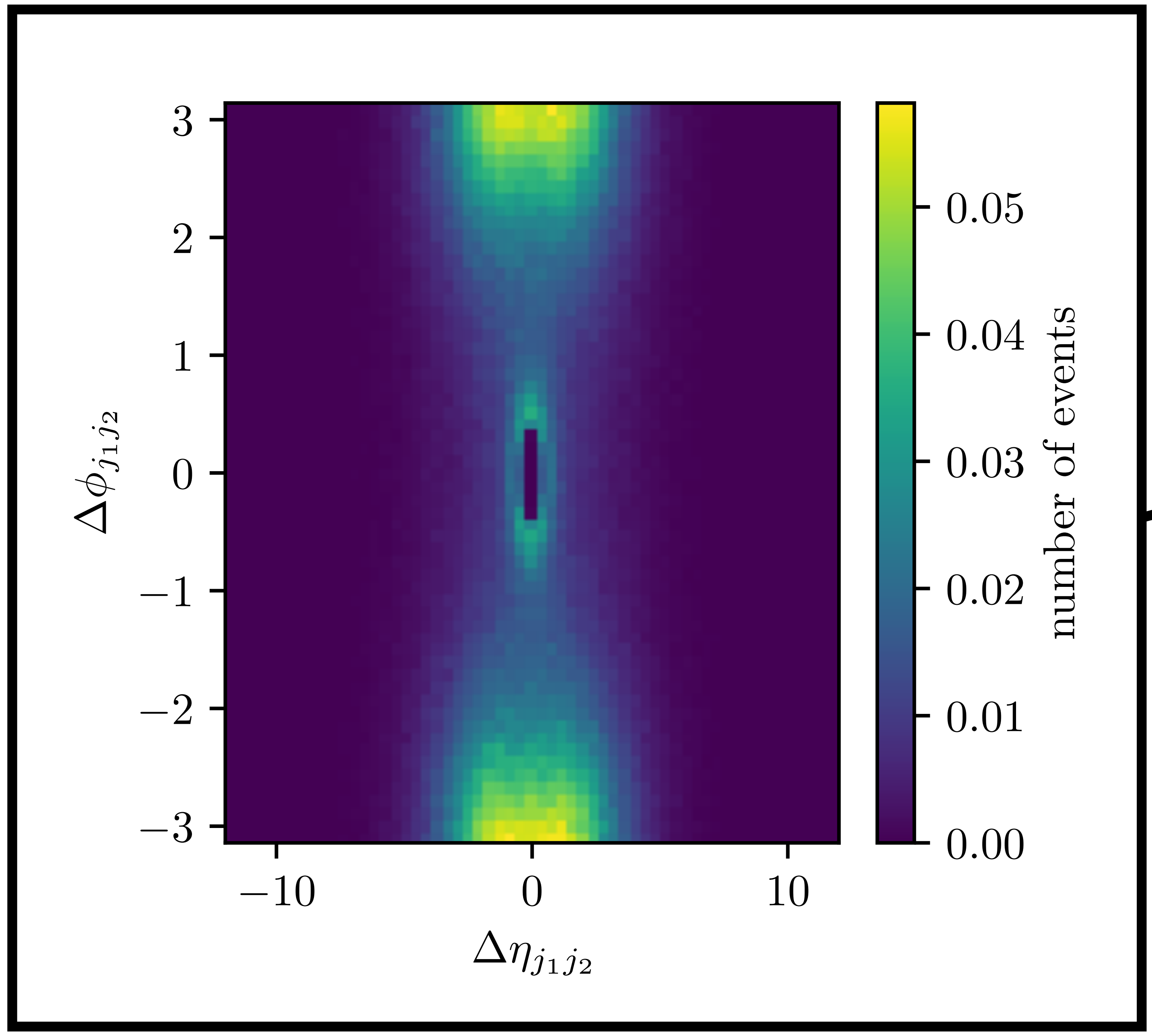


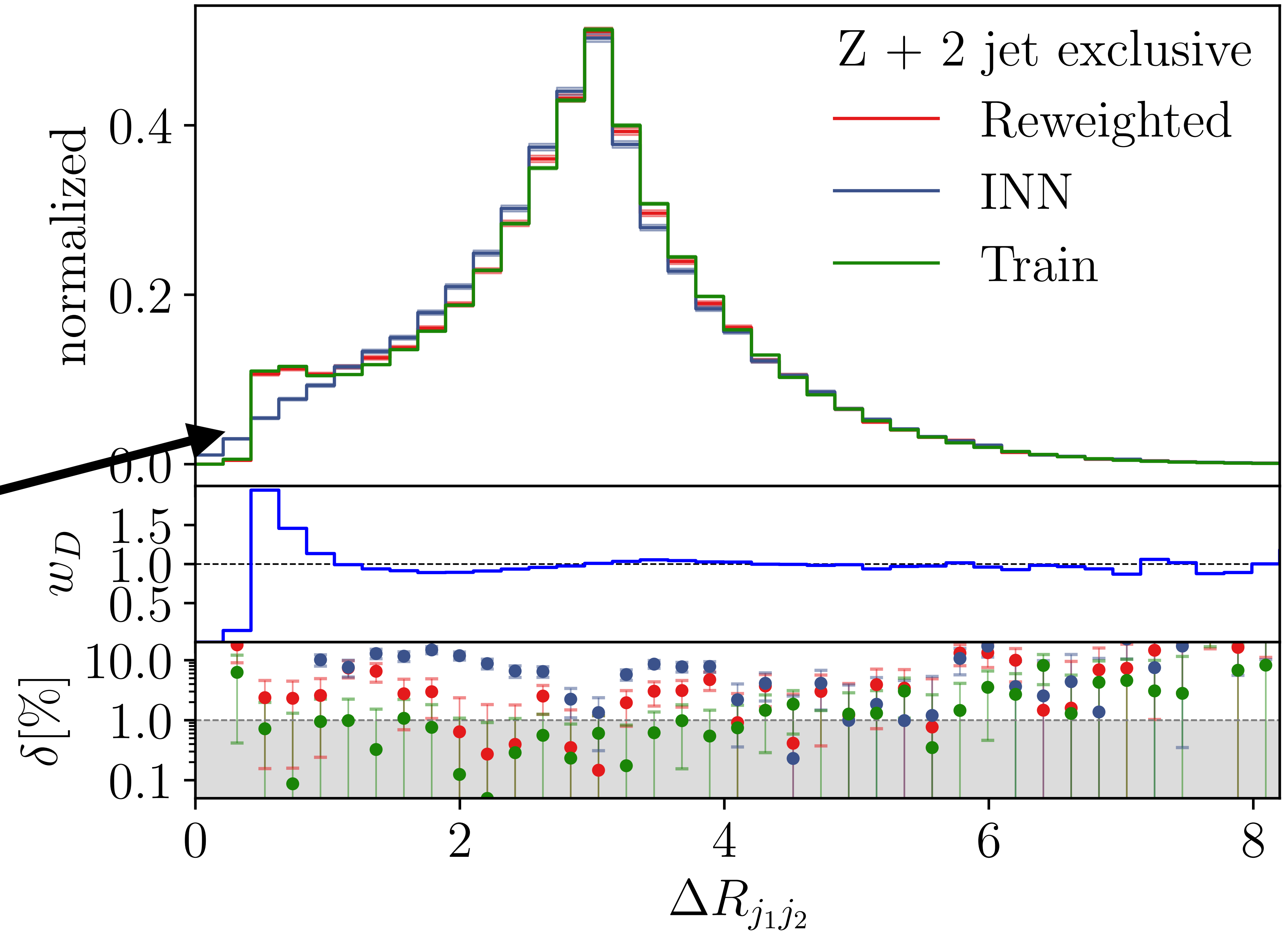
- Train normalizing flow on 4-momenta
- Include symmetries in feature representation
- Excellent performance for direct output

- Extend setup vor variable jet multiplicity

Challenges for normalizing flows

- Narrow features
- Topological holes (eg ΔR cuts)
  - no bijecive mapping possible
  - can only be approximated



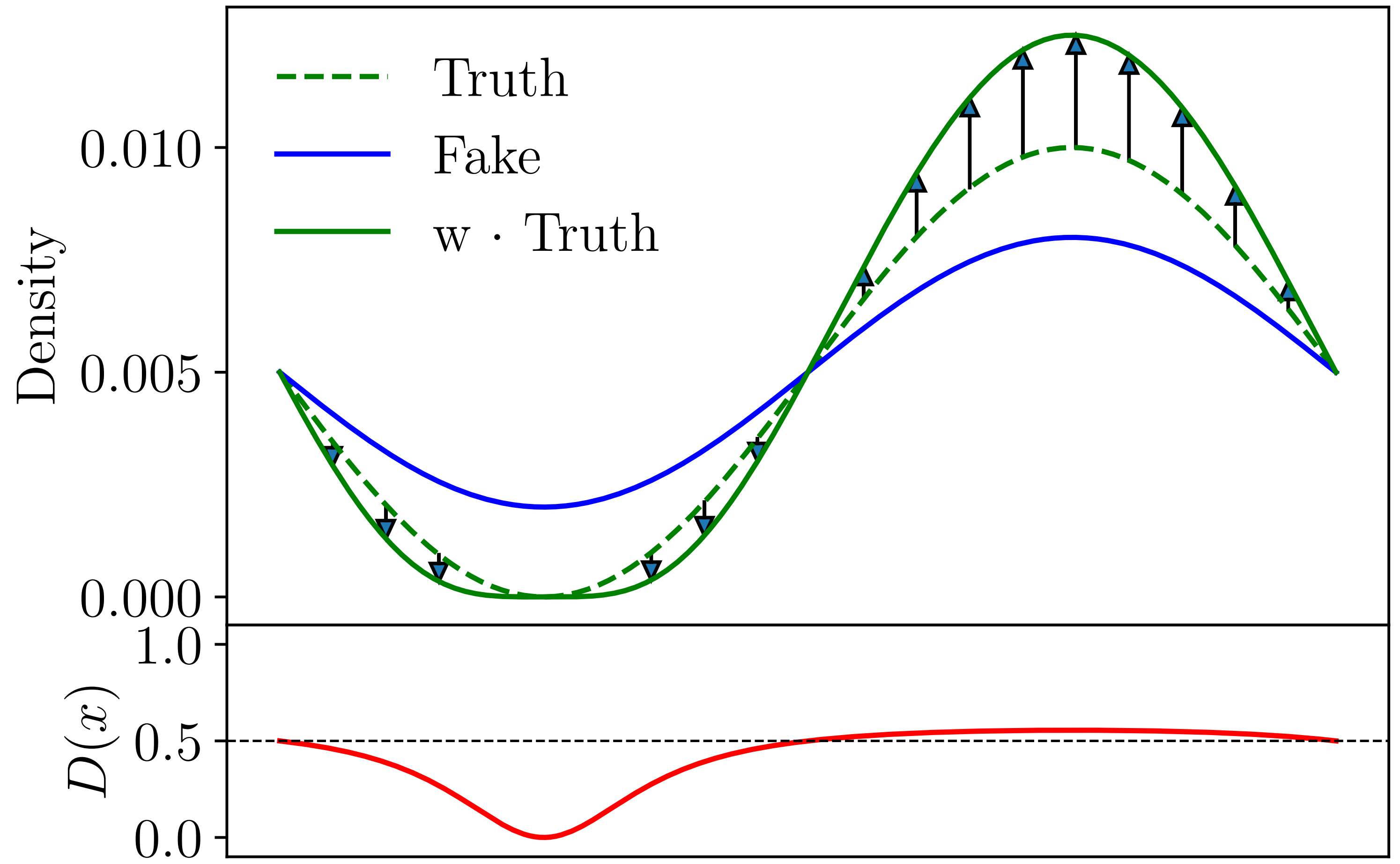


Reweighting for Precision

- Classifier loss

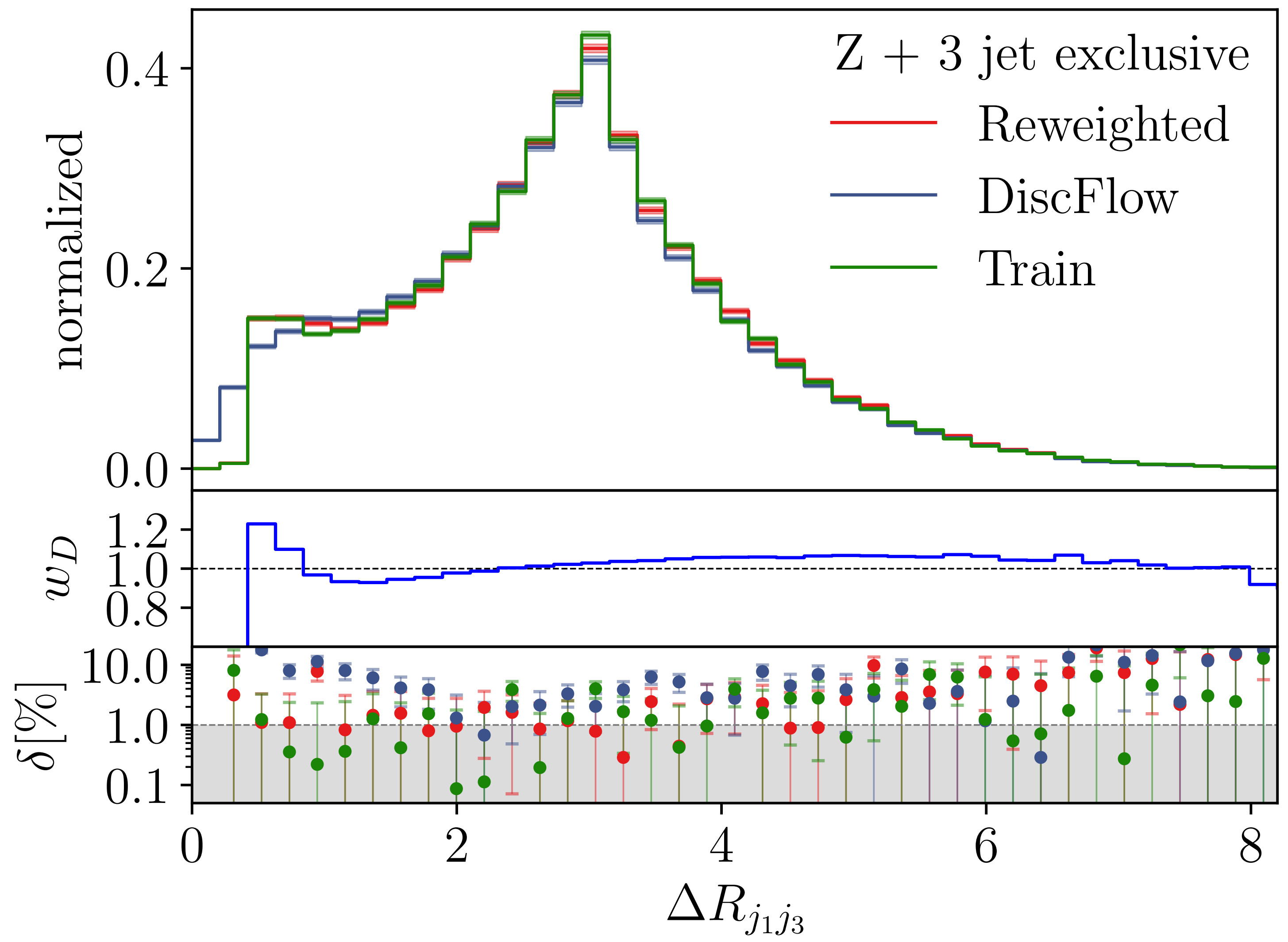
$$\mathscr{L} = -\sum_{x\sim p_{data}} \log(D(x)) - \sum_{x\sim p_{INN}} \log(1 - D(x))$$

$$= -\int \mathrm{d}x\, p_{data}(x)\ \log(D(x)) + p_{INN}(x)\ \log(1 - D(x))$$



- Upon convergence obtain **reweighting factor**

$$\Rightarrow \frac{p_{data}(x)}{p_{INN}(x)} = \frac{D(x)}{1 - D(x)} = w_D$$



- Use classifier feedback to enhance gradients

$$\mathscr{L}_{\text{DiscFlow}} \approx \int \mathrm{d}x \quad \underbrace{w_D(x)^{\alpha} P(x)}_{\text{reweighted truth}} \left(\frac{\psi(x;c)^2}{2} - \log J(x) \right)$$

$\Rightarrow$ Reduces range of reweighting factors

ML Uncertainties

When do we (not) need them?

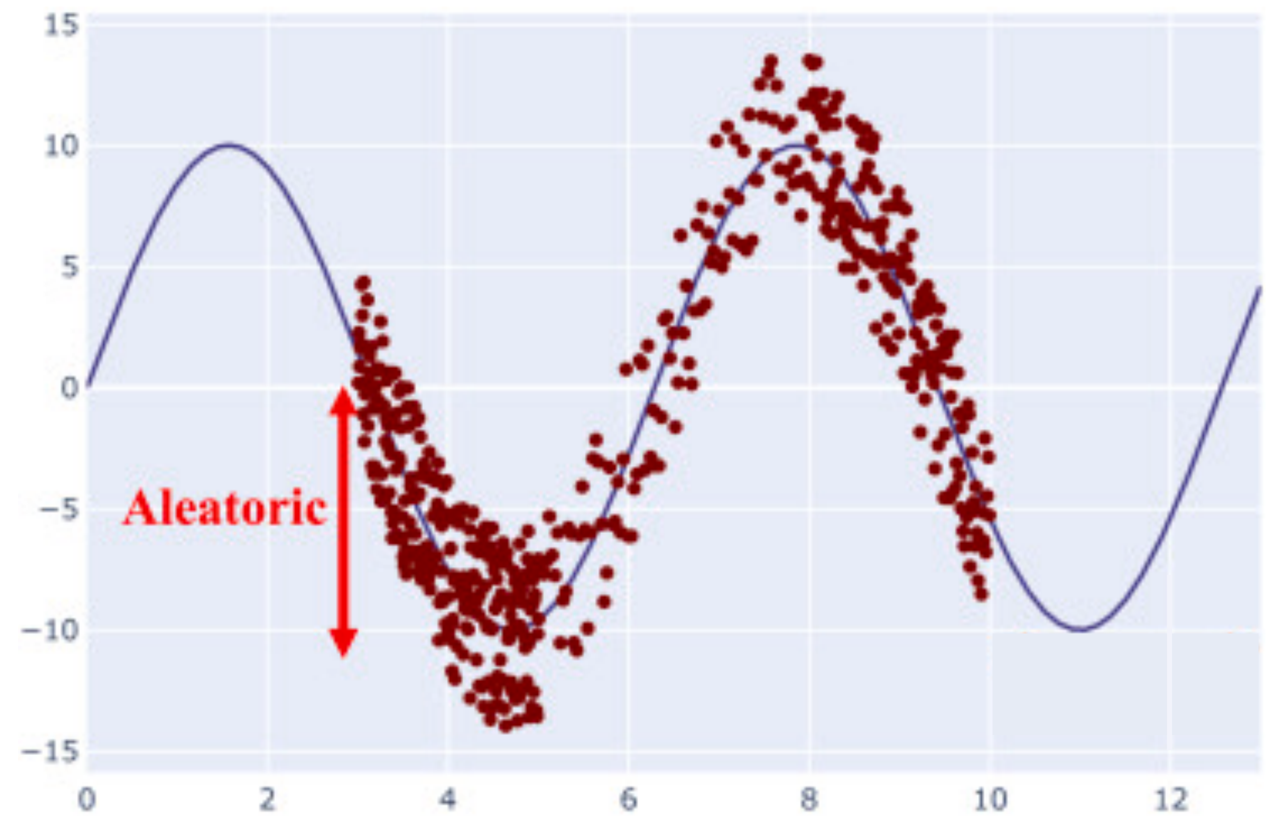
- Predictions with poorly trained NNs are sub-***optimal*** but not ***wrong***

- Example 1: **Phase space sampling**
  - Mapping induces Jacobian
  - Events obtain weight from ME $\times$ J
  - Bad mapping $\rightarrow$ small unweighting efficiency

- Example 2: INN for **integration**
  - Sub-optimal contour deformation
  - **High variance** of integral
  - Not efficient but not wrong

➡ **Control** comes from **simulation** !

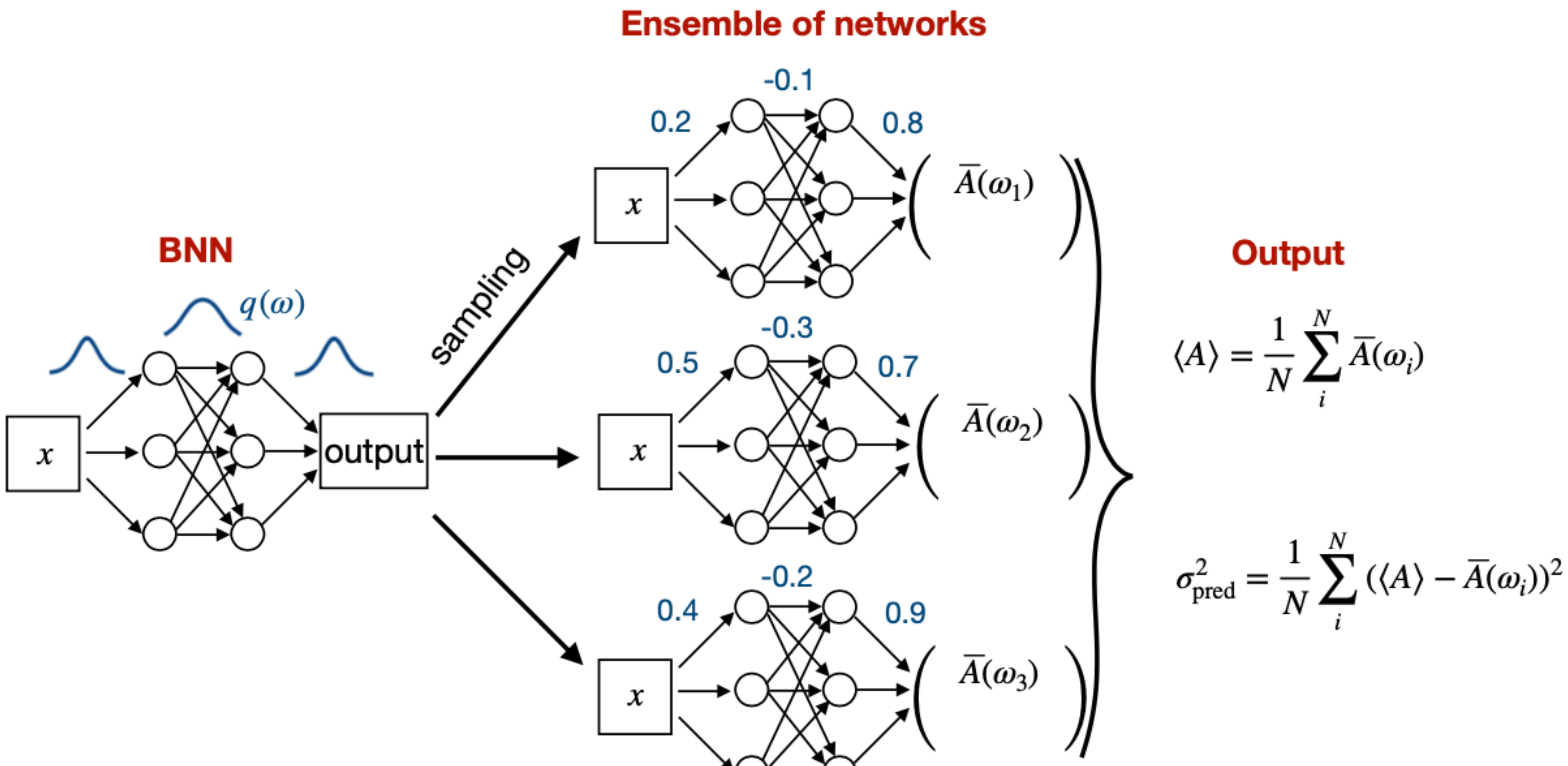
- Example 3: Data compression
  - Direct use of generator output



modified from M. Abdar [doi.org/10.1016/j.inffus.2021.05.008]

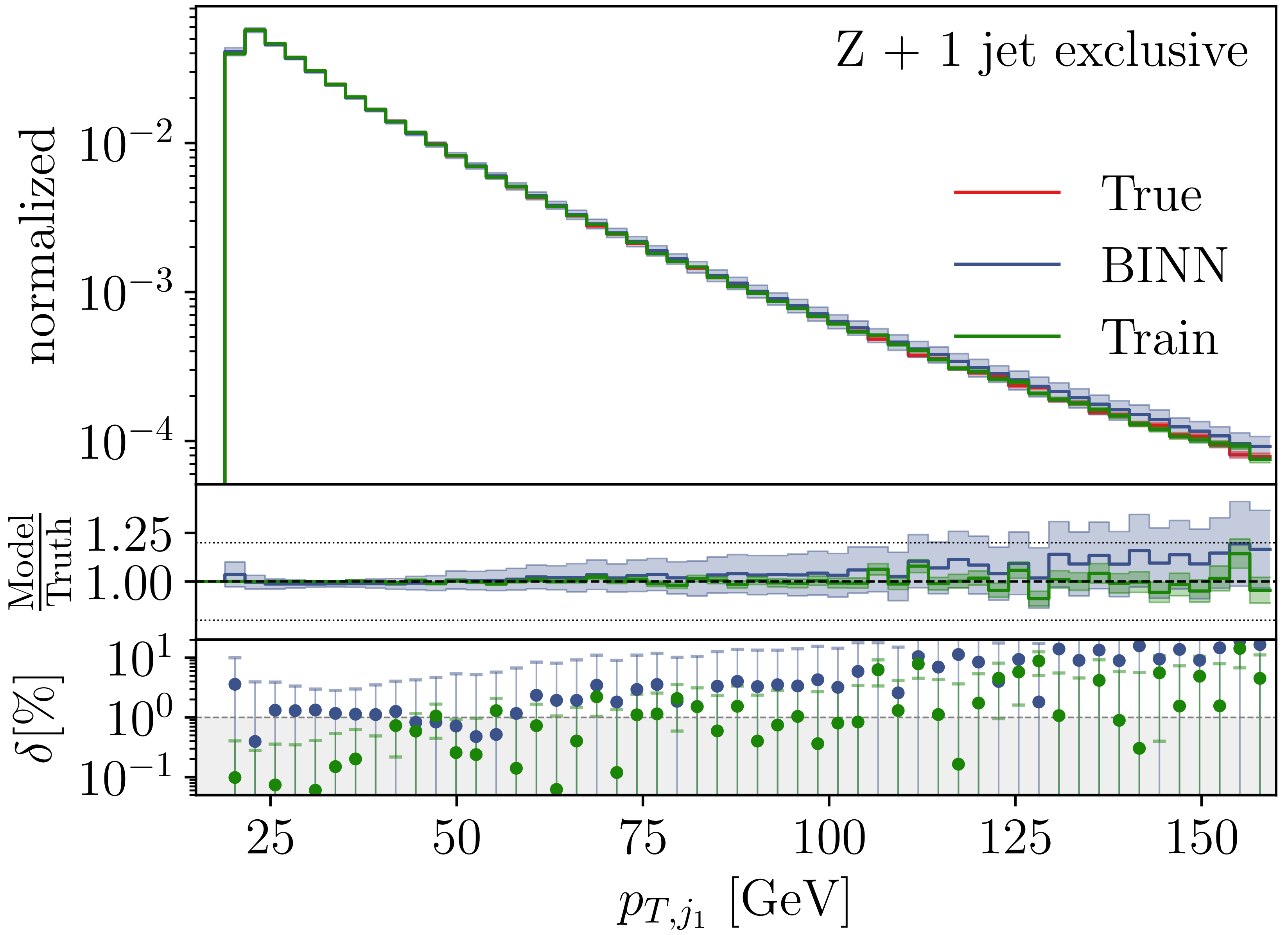
➡ How can we estimate this uncertainty?

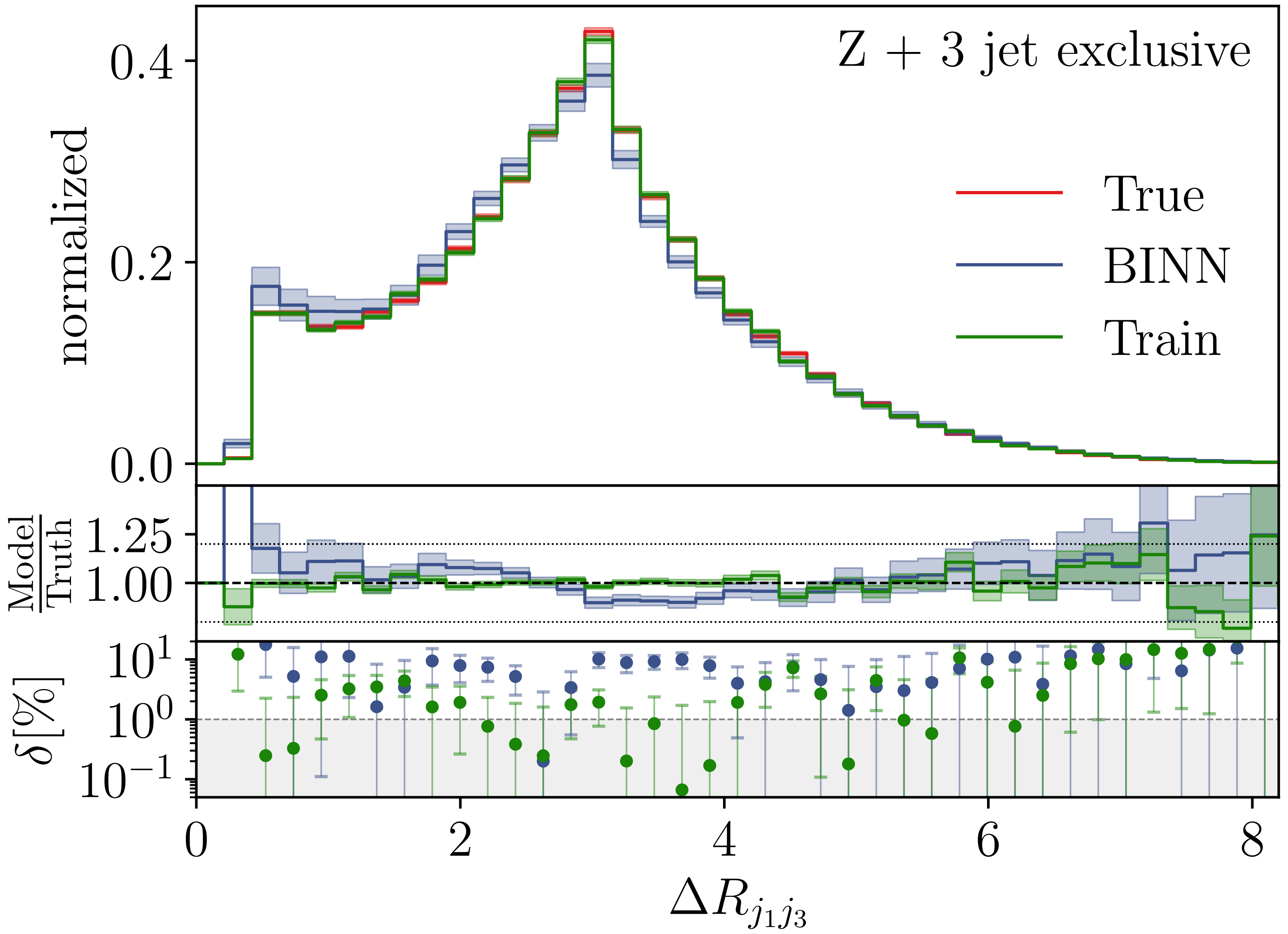
Bayesian Neural Network

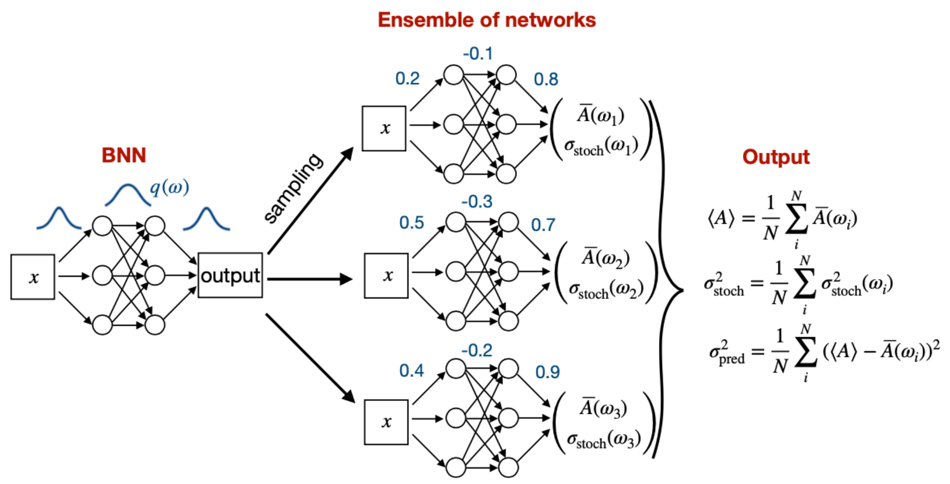


$$
\begin{aligned}
\mathscr{L} &= \mathscr{L}_{INN} + KL_{prior} \\
&= \sum_{n=1}^{N} \langle \log p_X(x_n \,|\, \theta) \rangle_{\theta \sim q_\Phi(\theta)} - KL(q_\Phi(\theta), p(\theta))
\end{aligned}
$$

Bayesian generative networks





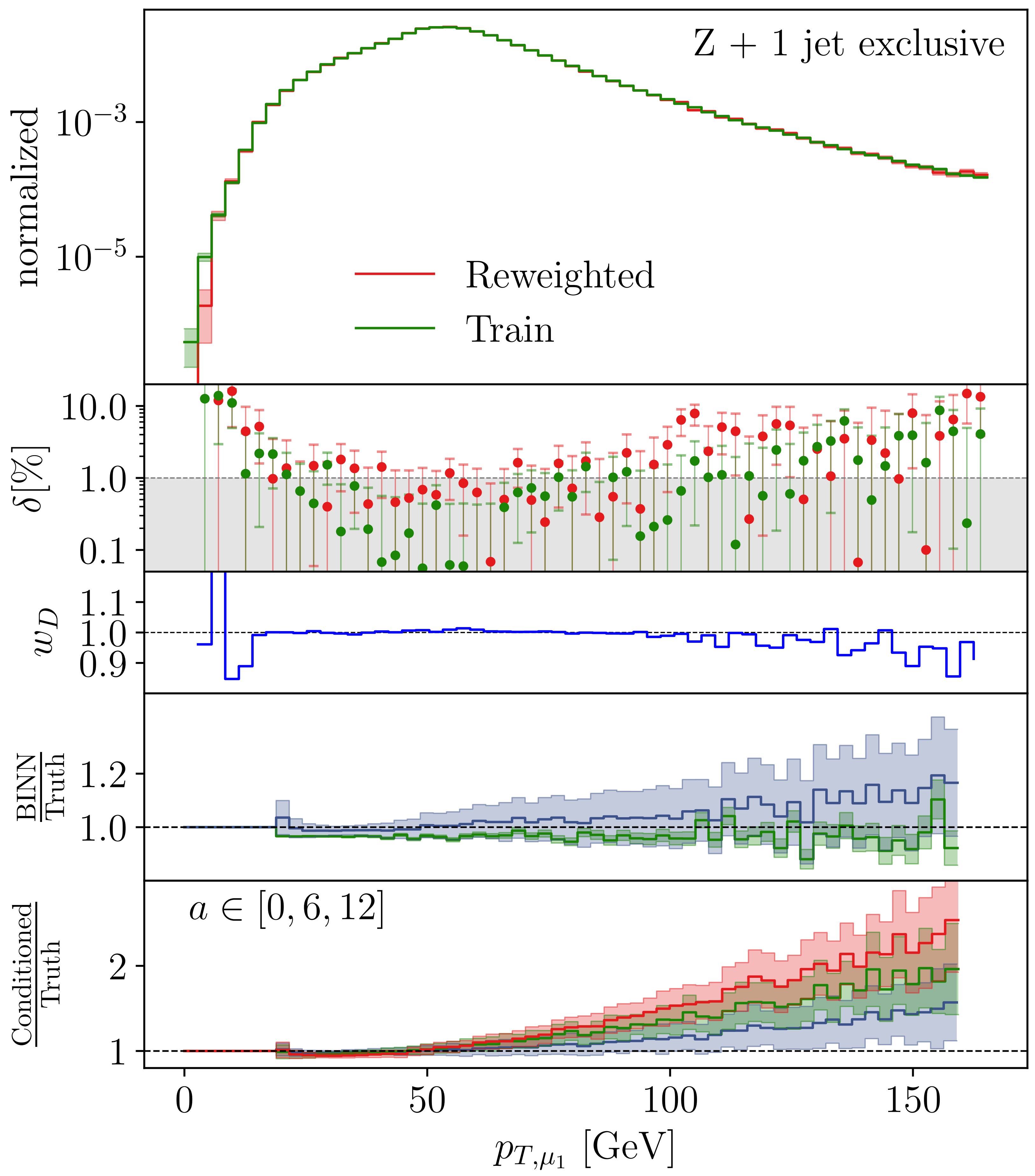


⇒ BINN captures uncertainty related to convergence and statistical uncertainties

⇒ BINN does not capture lack of expressiveness

Putting flows to work

Event generation



- Basis: INN
  - Phase space symmetries in architecture
- Control via classifier D
  - $$\frac{p_{\text{truth}}(x)}{p_{\text{INN}}(x)} = \frac{D(x)}{1 - D(x)}$$
- Precision via reweighting
  - Correct deviations of p_{INN}

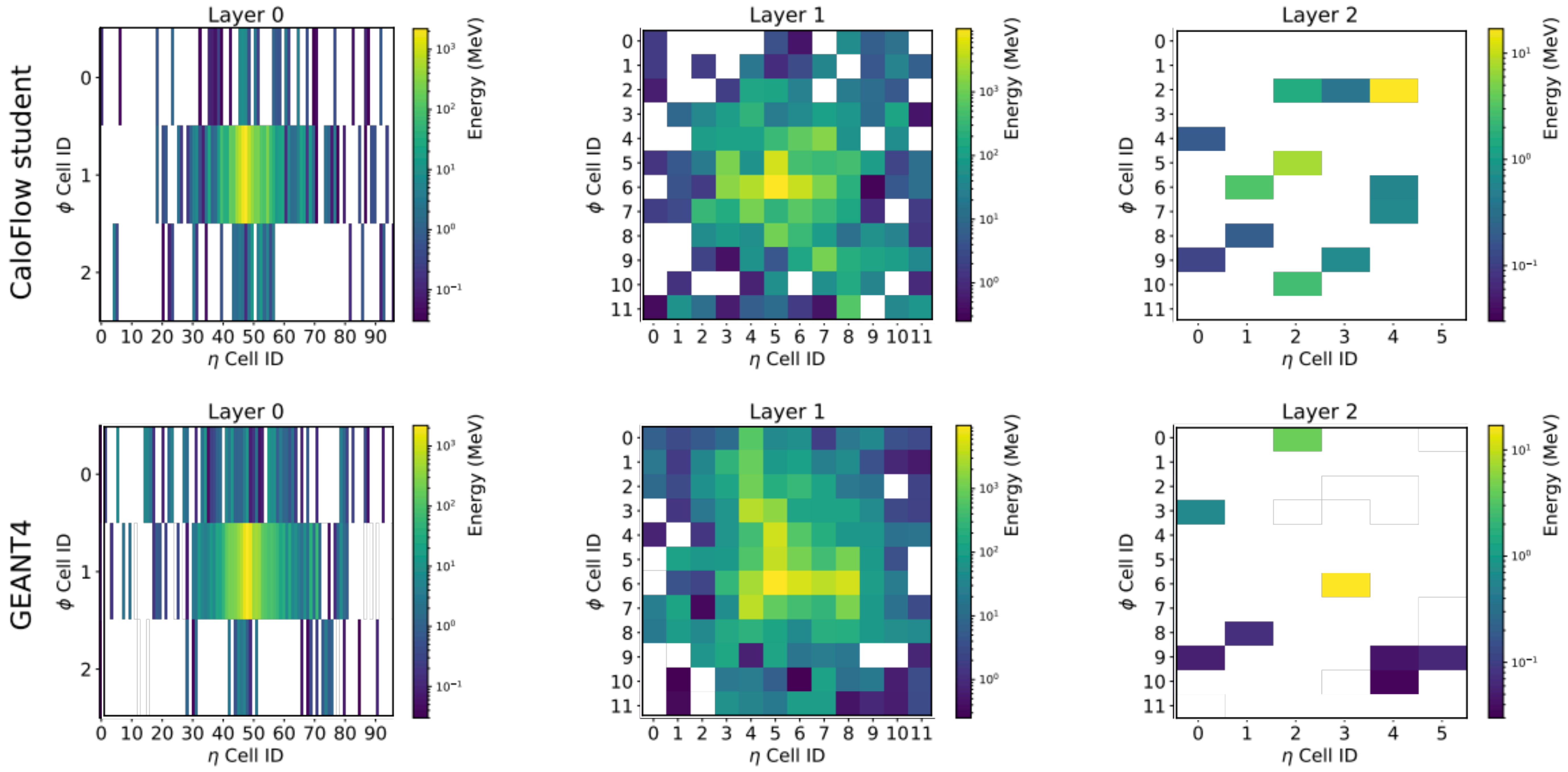
➡ Uncertainty estimation via Bayesian NN

➡ Uncertainty propagation via conditioning

Putting flows to work

Detector simulation

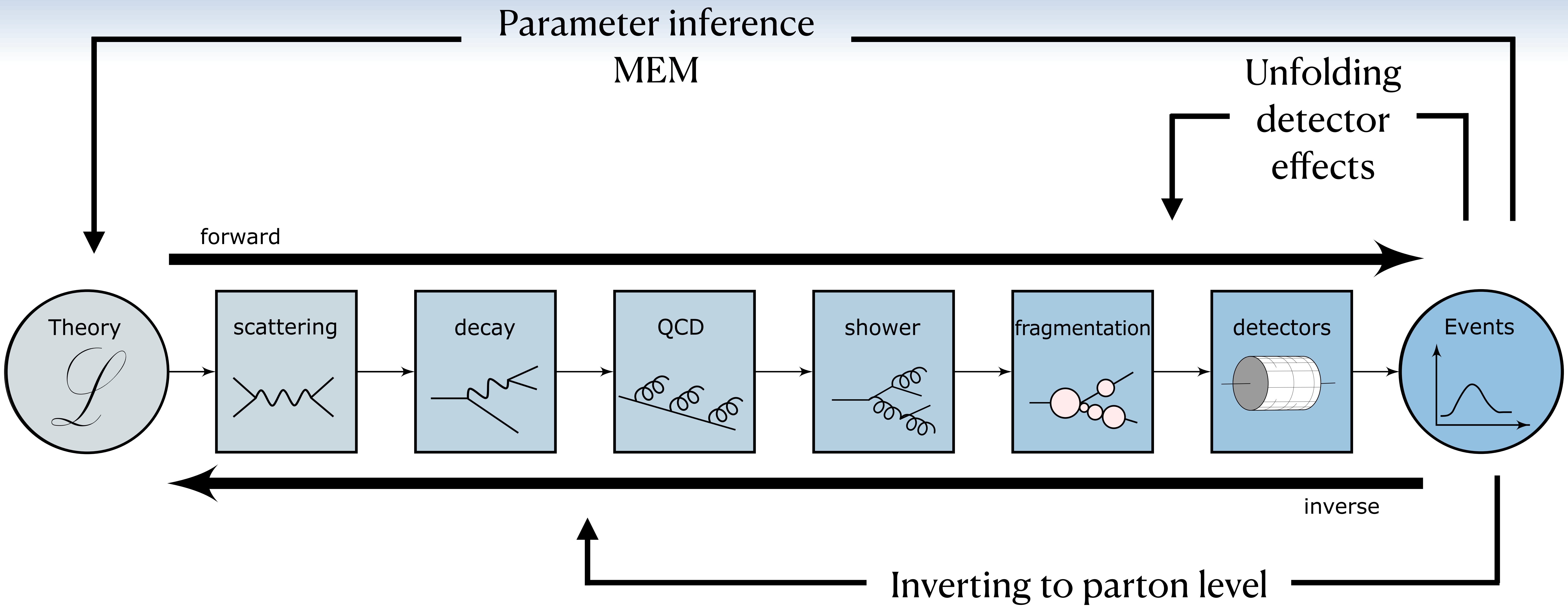
Challenge: large dimensionality $(3 \times 96, 12 \times 12, 12 \times 6)$



C. Krause & D. Shih [2110.11377]

π^+ shower individual & average

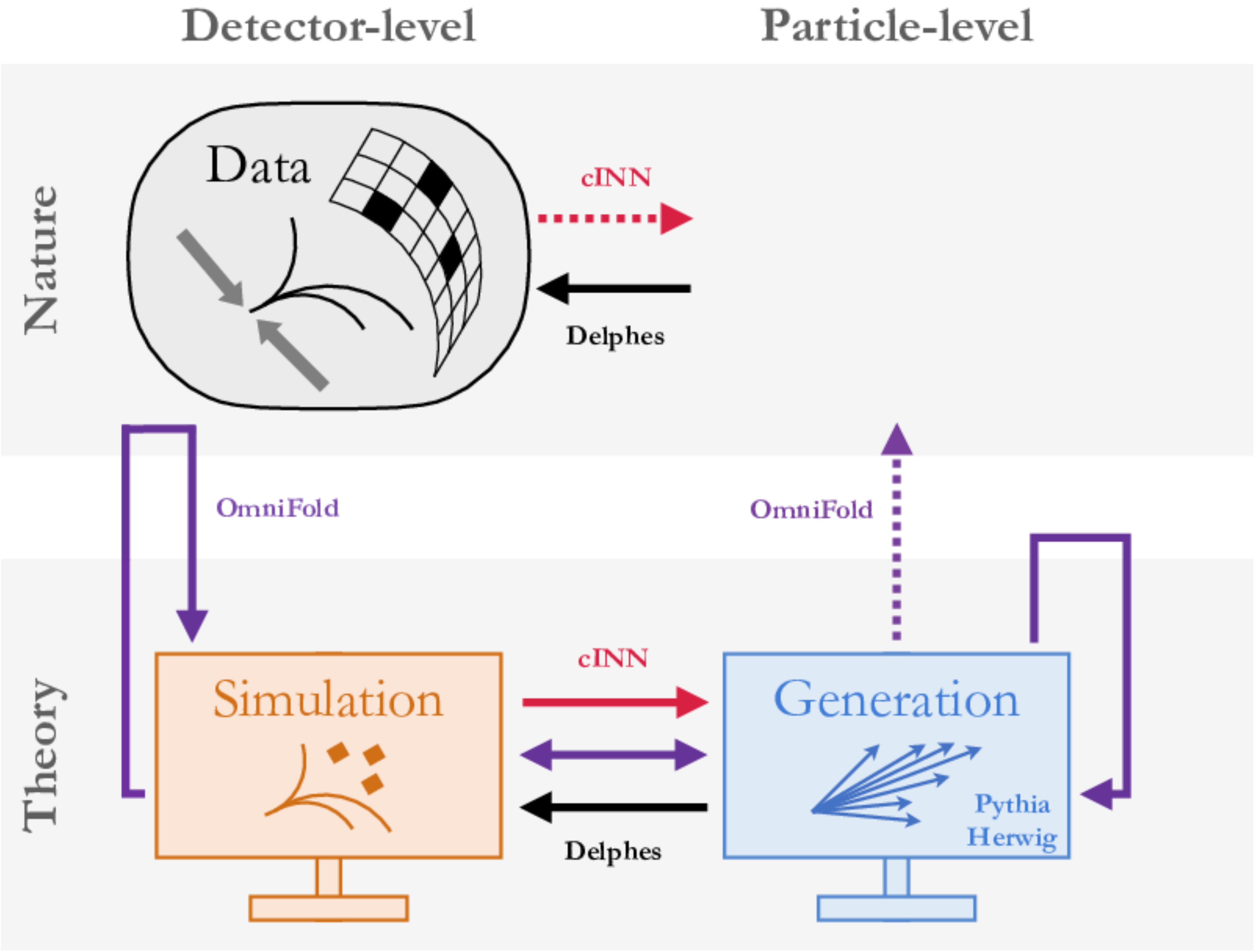
Inverting the simulation chain



Requirements
- ☐ Highdimensional
- ☐ Bin independent
- ☐ Statistically well defined

ML unfolding methods

High-dimensional. Bin independent. Robust.



M. Arratia et al. [2109.13243]

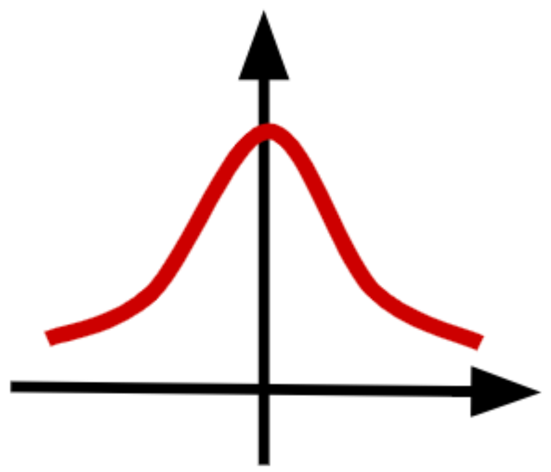
Classifier based aproach
Output: reweighted distribution of MC events

Density based approach
Output: probability density per unfolded event

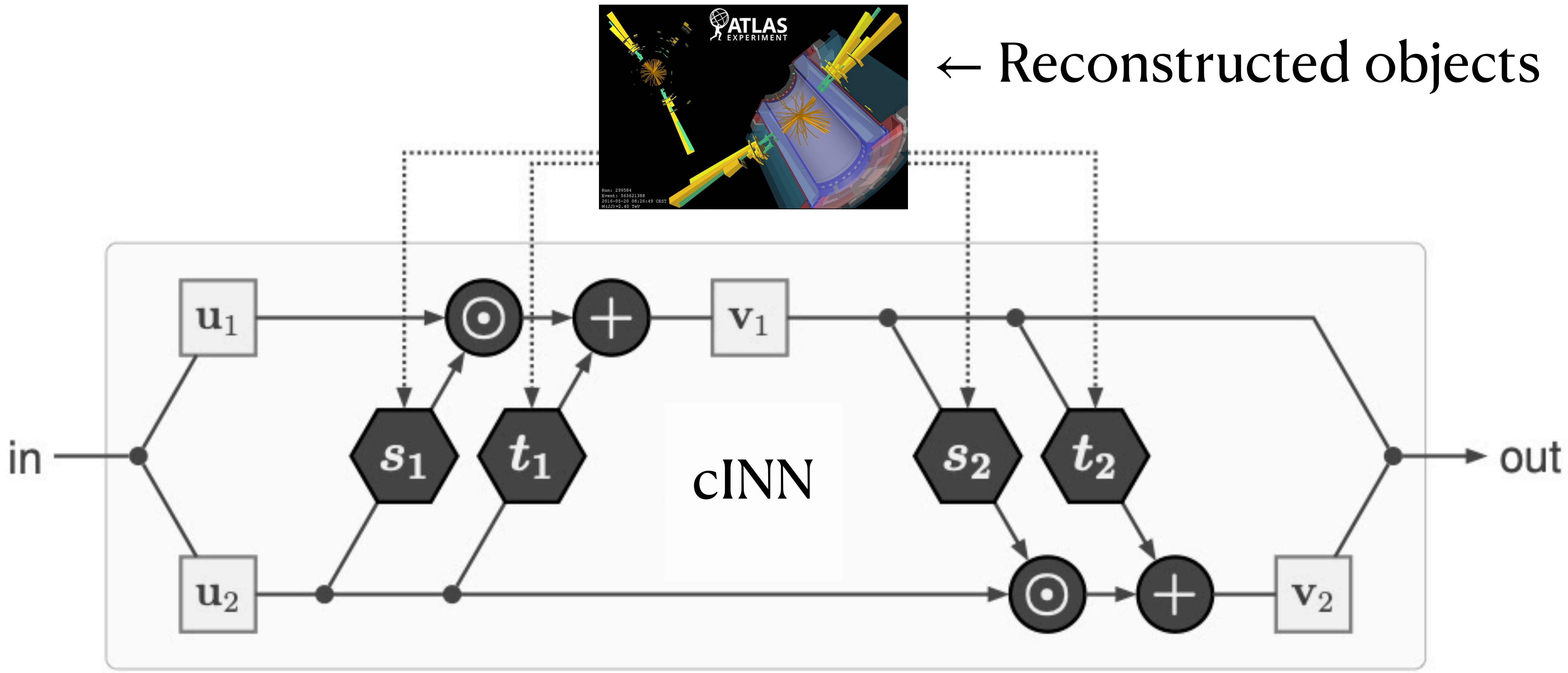
VAE alternative: OTUS by J. N. Howard et al.

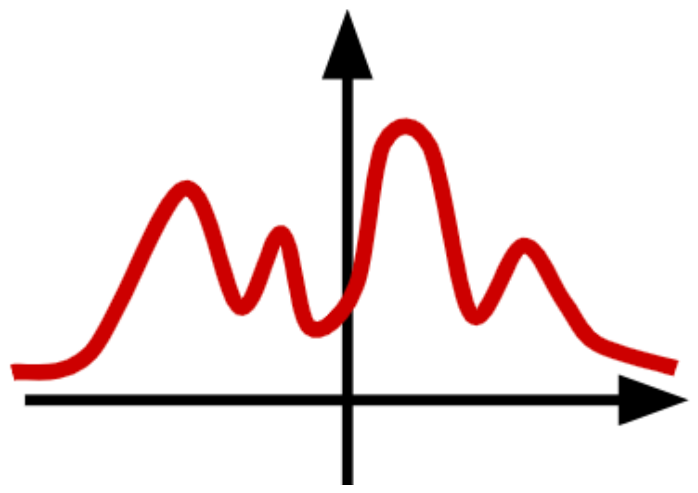
cINN unfolding

Given a reconstructed event:
What is the probability distribution at particle level?



Normal distribution
$z \sim \mathcal{N}$





Probability distribution
$x \sim P_{\text{parton}}$

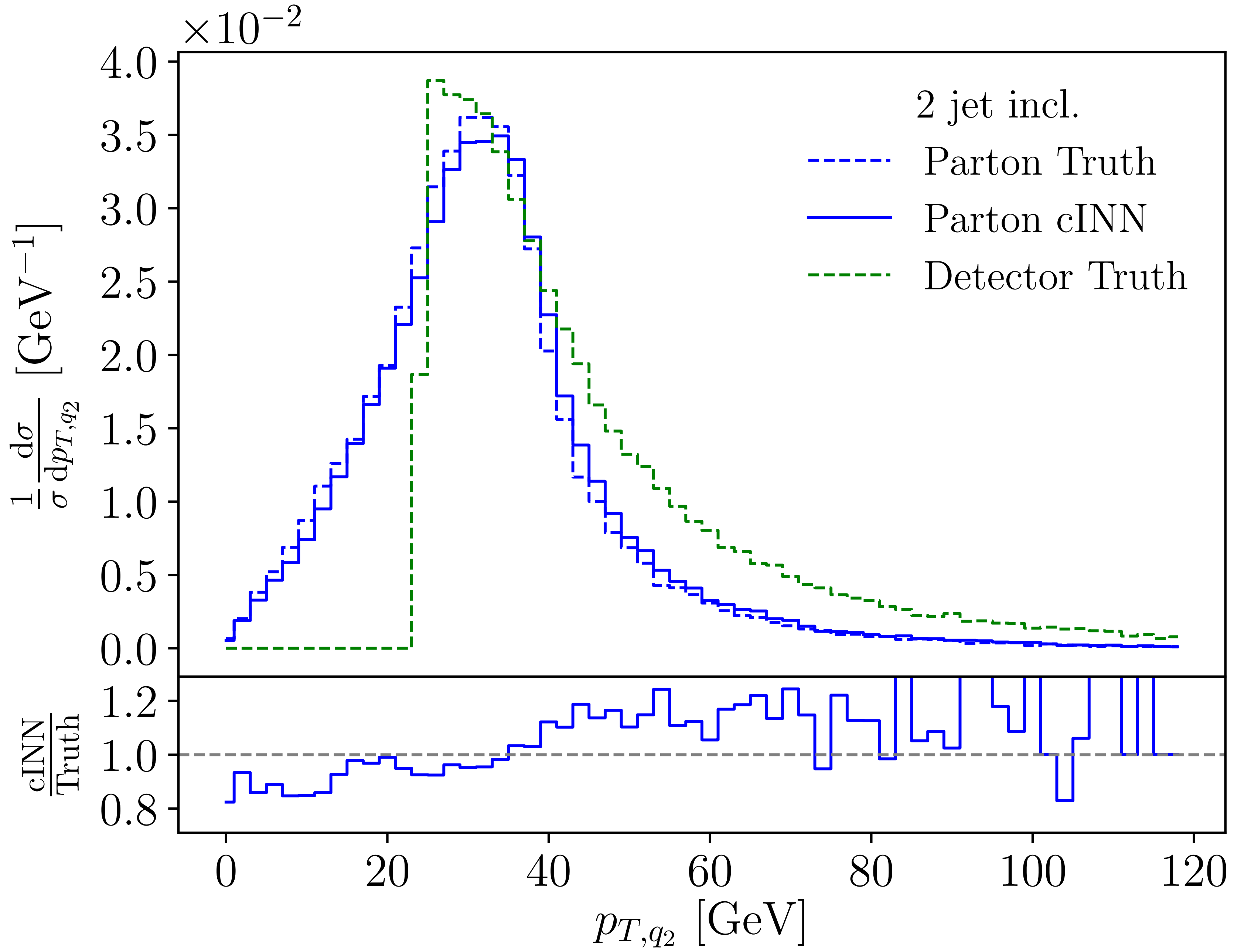
Training

Unfolding

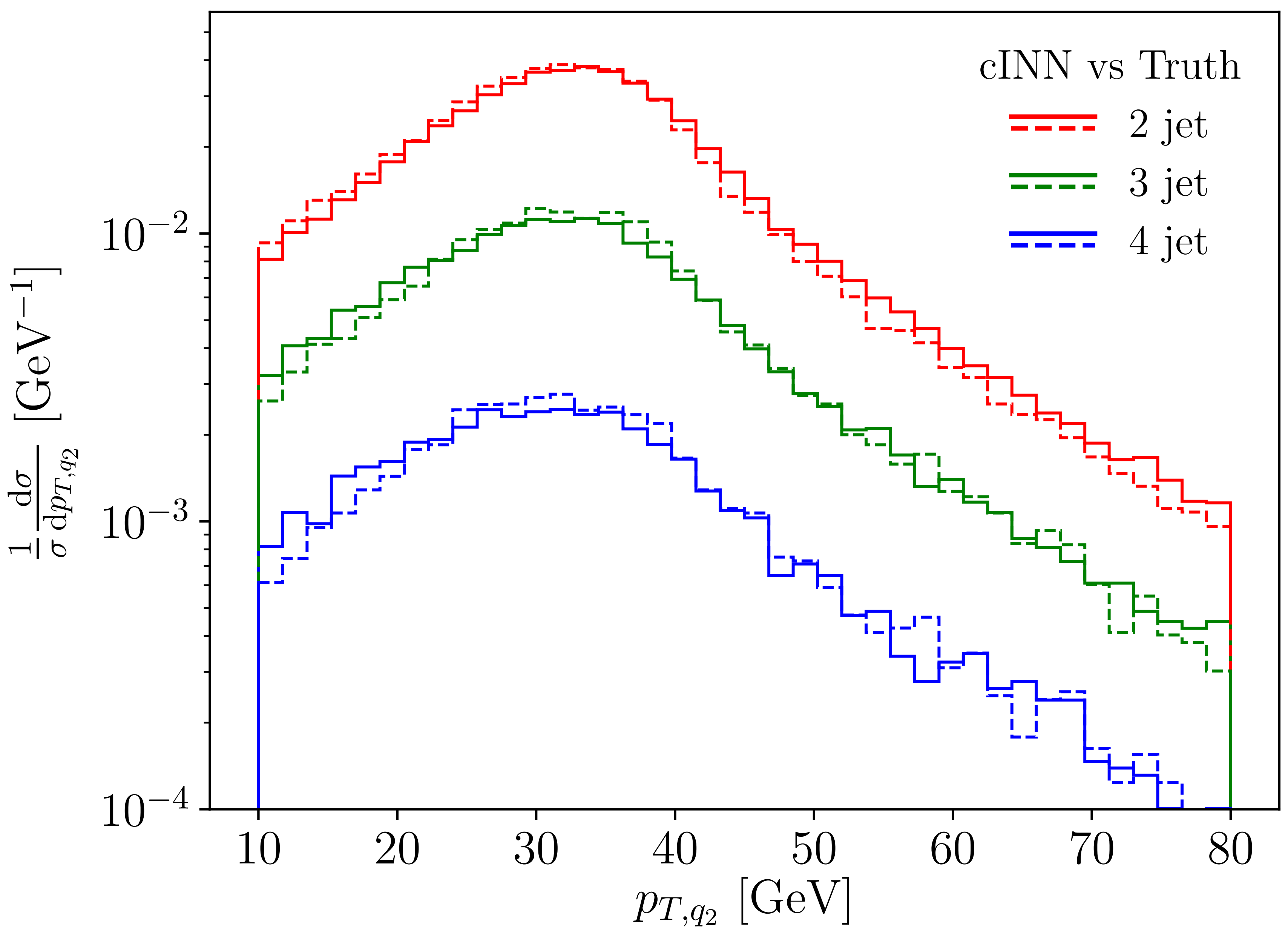
Inverting inclusive distributions

$pp > WZ > q\bar{q}l^+l^- + \text{ISR} \rightarrow$ 2/3/4 jet events

Training on inclusive dataset



Evaluate exclusive 2/3/4 jet events

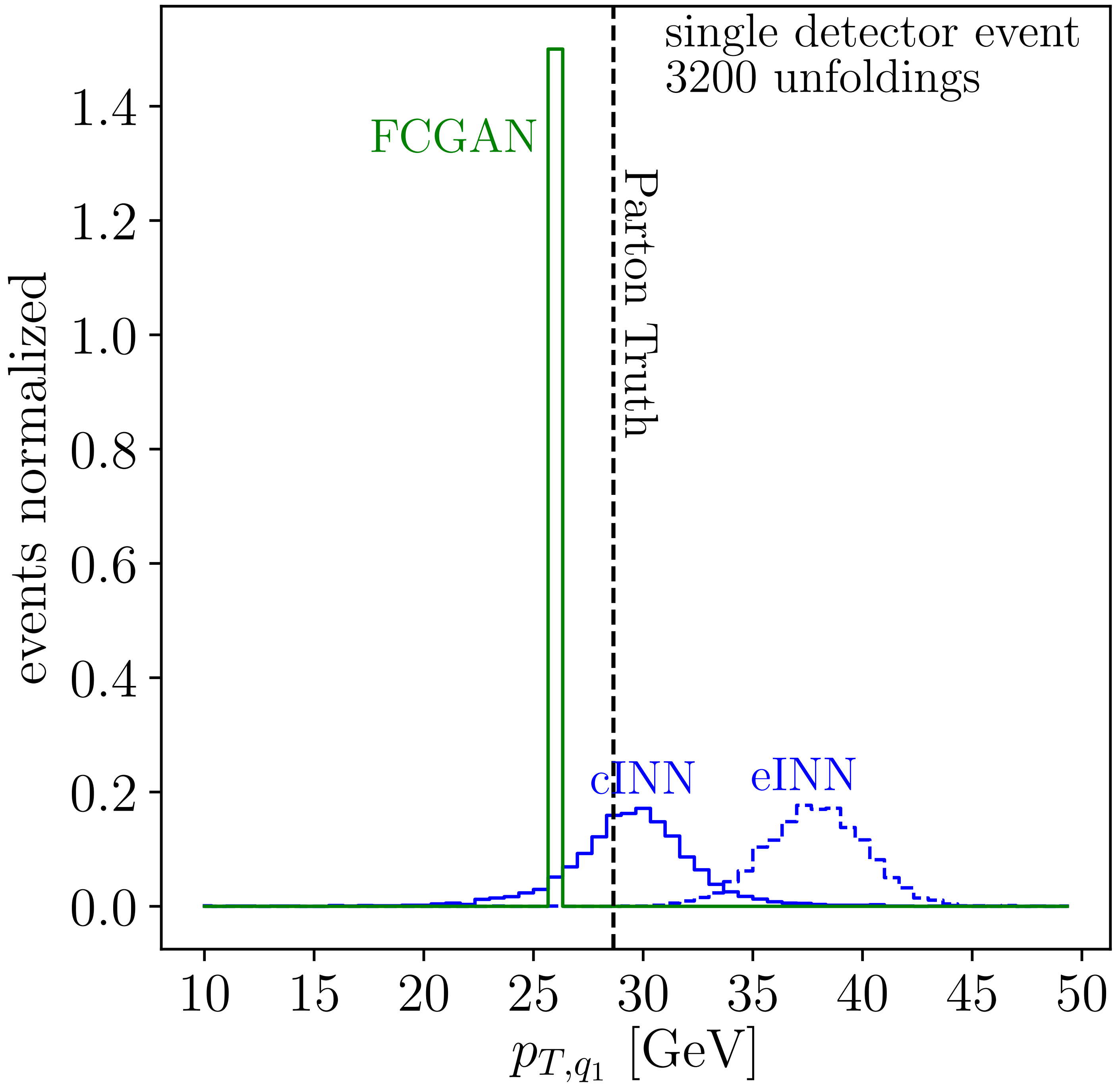


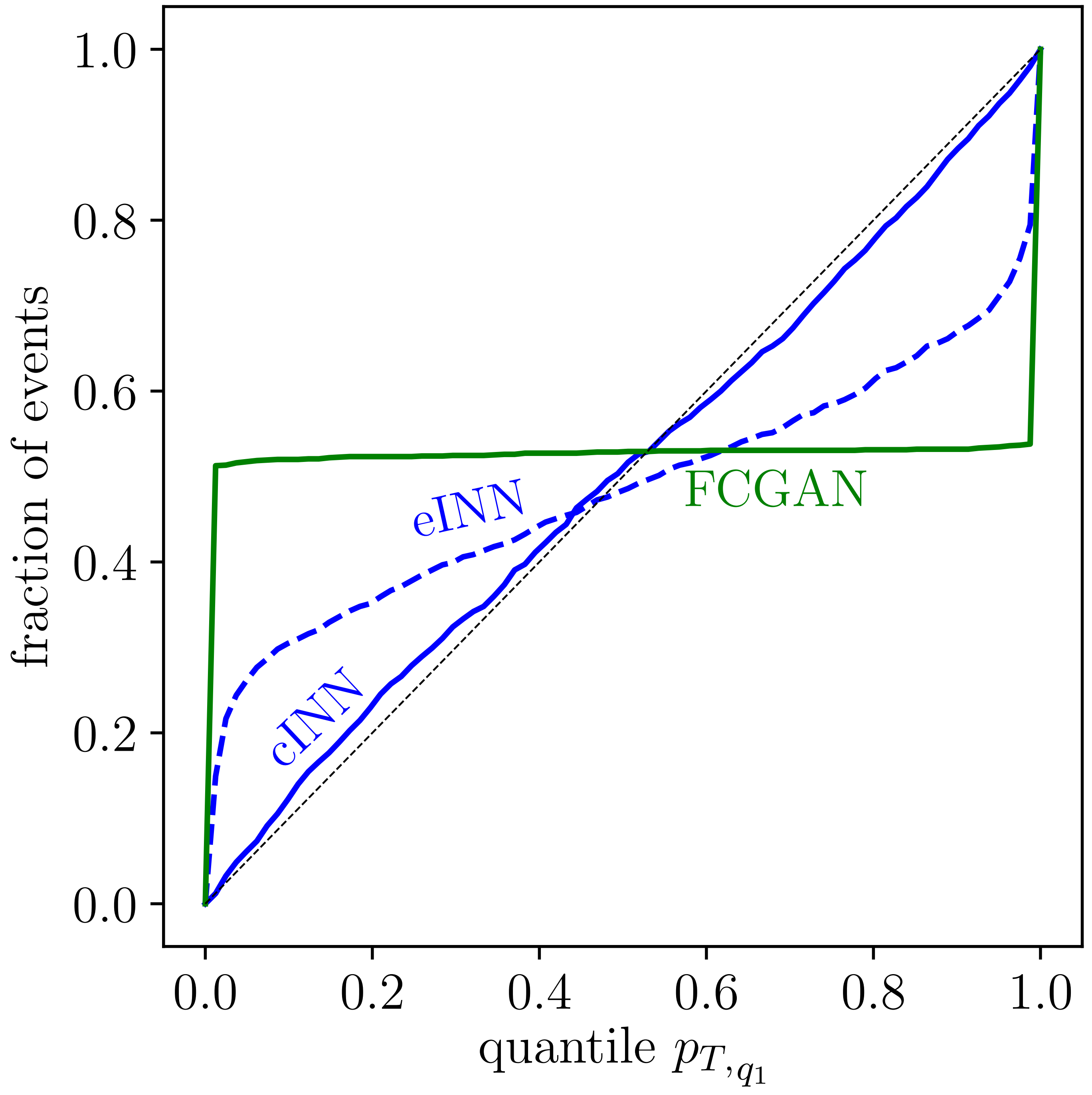
- [x] High-dimensional
- [x] Bin-independent
- [ ] Statistically well defined ?

M. Bellagente et al. [2006.06685]

Event-wise unfolding

No deterministic mapping!
Check calibration of probability density for individual event unfolding

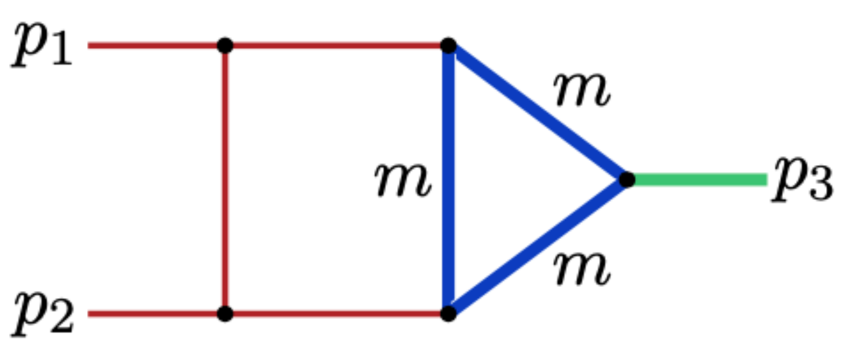


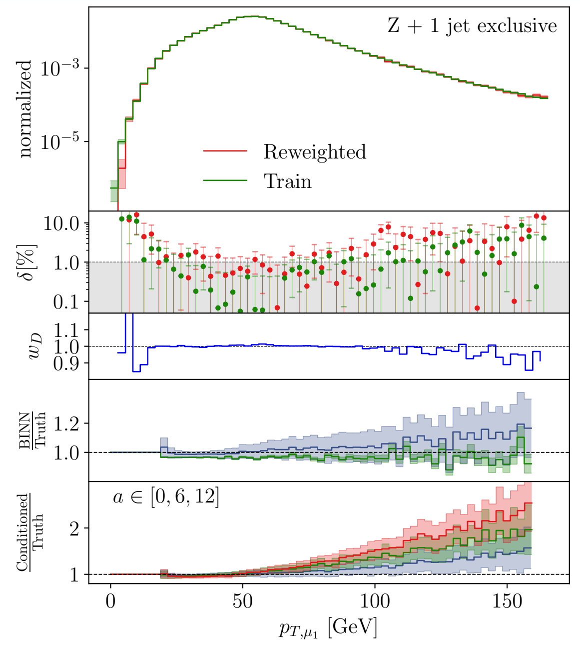


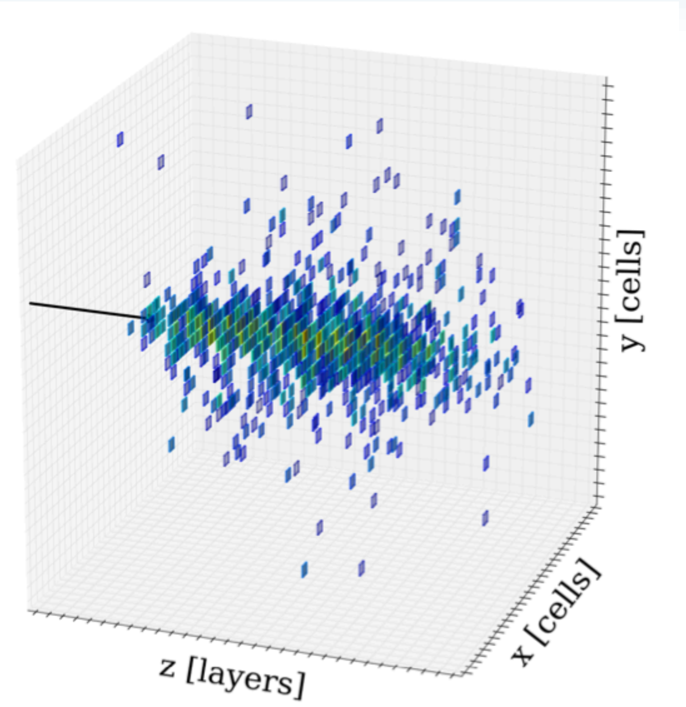
- ☑ High-dimensional
- ☑ Bin-independent
- ☑ Statistically well defined

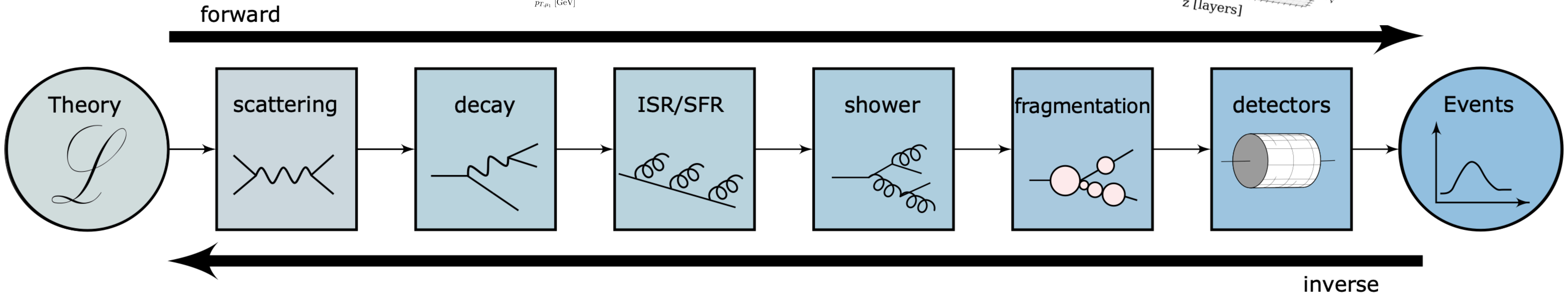
M. Bellagente et al. [2006.06685]

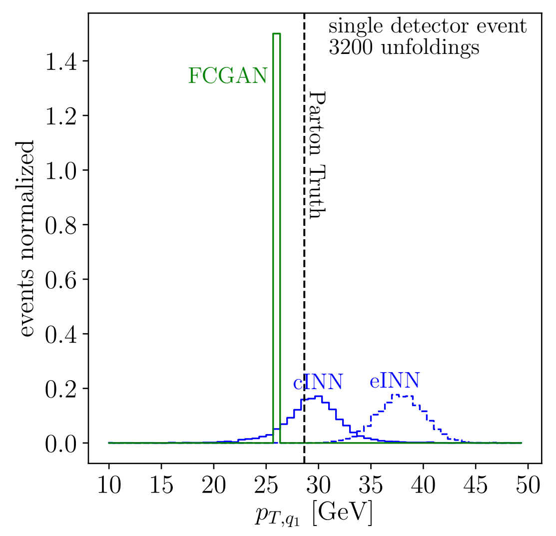
ML 4 LHC Event generation

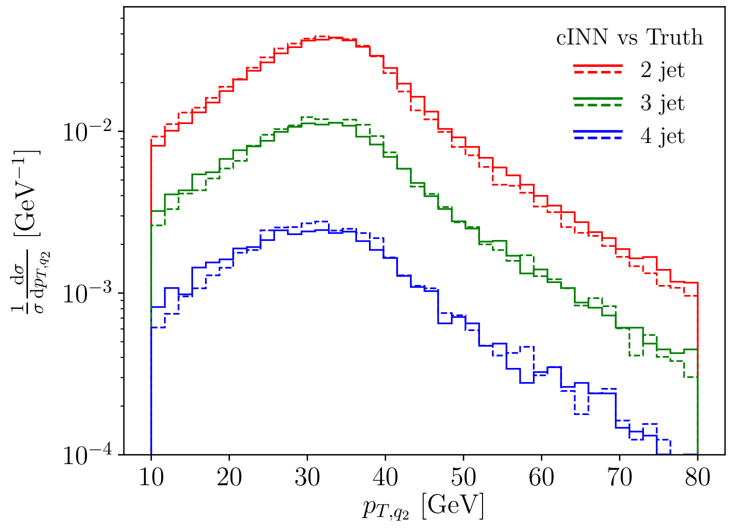


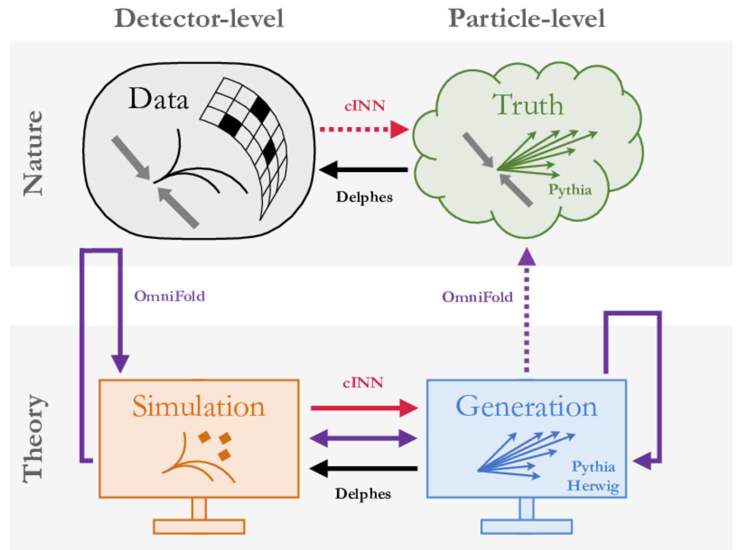












New data are currently on their way...