When (not) to use uncertainties for ML in particle physics

Bayesian Deep Learning for Cosmology and Time Domain Astrophysics

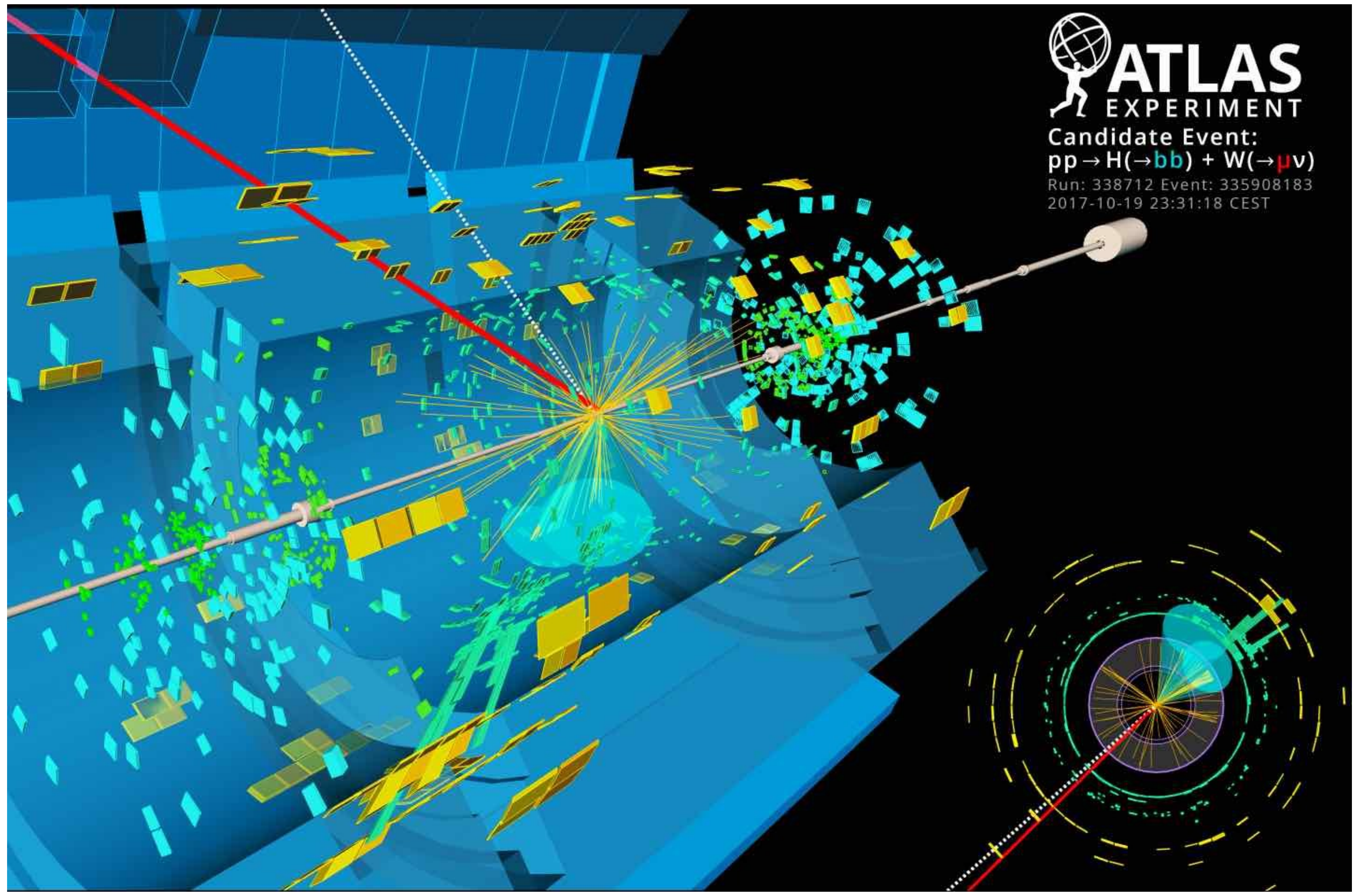
Anja Butter, ITP Heidelberg

Setting the stage for LHC physics

Before the particle accelerator

Setting
- Large Hadron Collider at CERN
- Independent proton collisions at 13 TeV
- Recorded by ATLAS, CMS, LHCb, ALICE
- **Huge** dataset ~1Pb/s before trigger selection

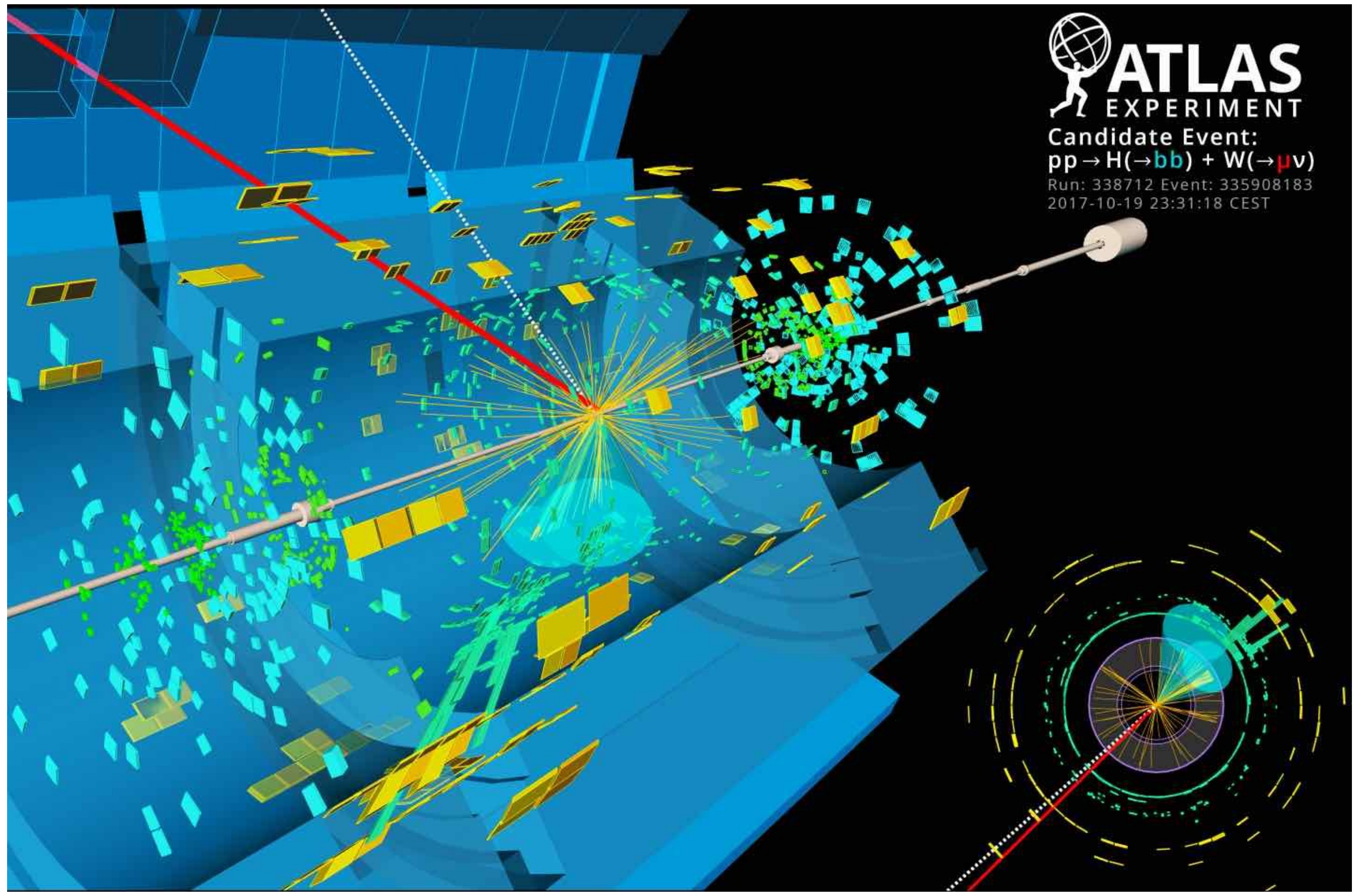
Setting the stage for LHC physics

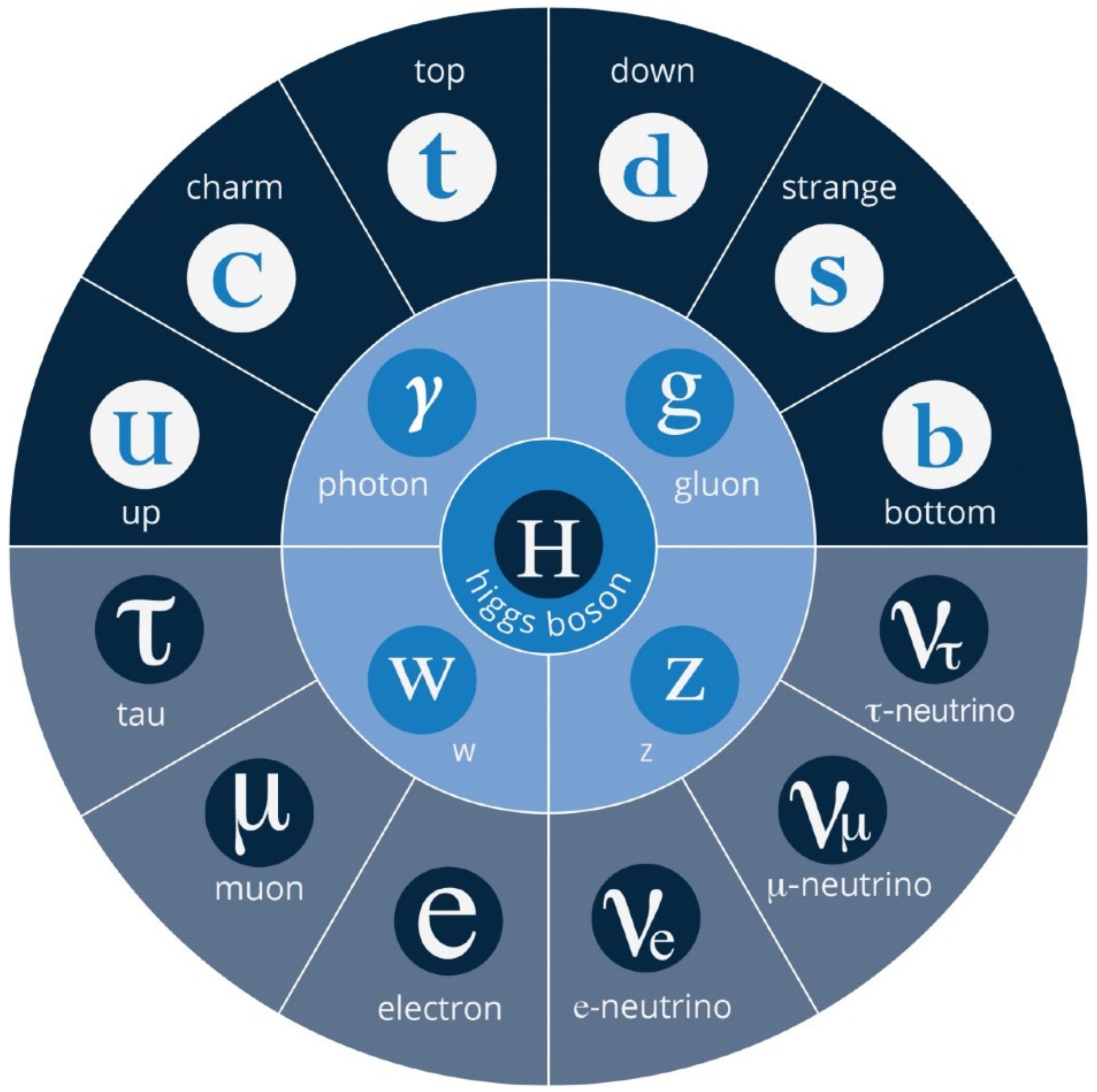


Setting

- Large Hadron Collider at CERN
- Independent proton collisions at 13 TeV
- Recorded by ATLAS, CMS, LHCb, ALICE
- **Huge** dataset ~1Pb/s before trigger selection

Setting the stage for LHC physics





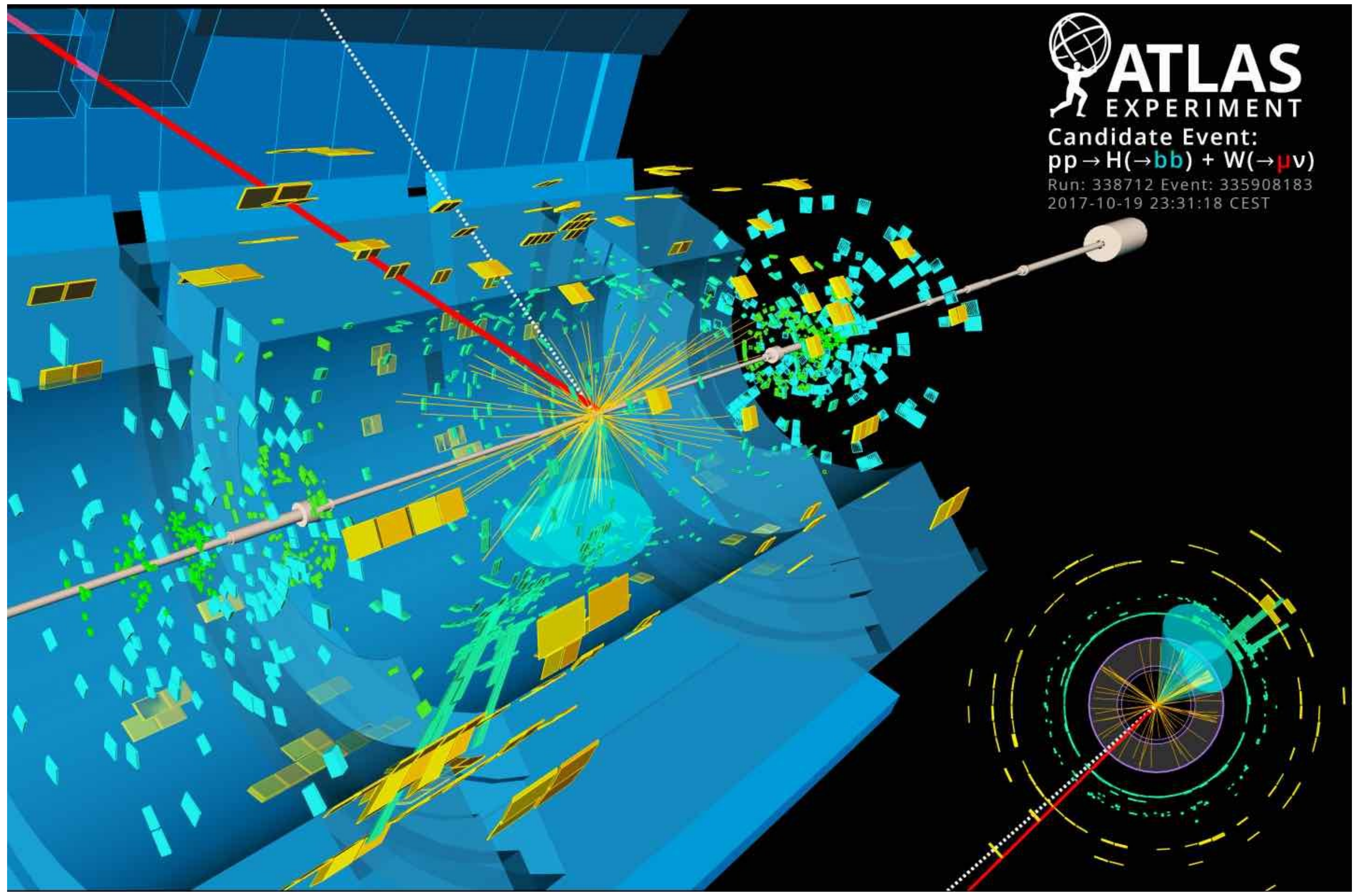
Setting

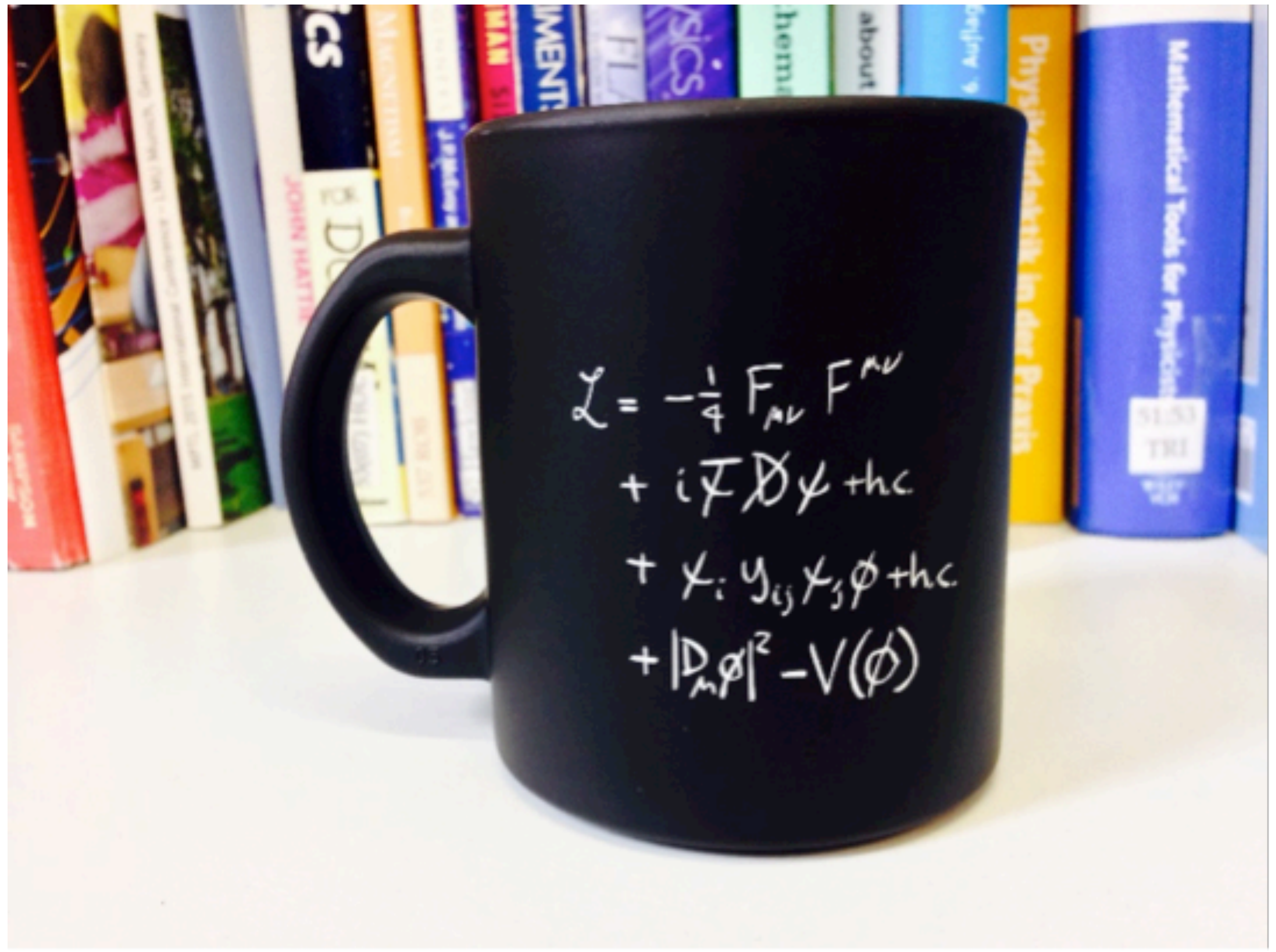
- Large Hadron Collider at CERN
- Independent proton collisions at 13 TeV
- Recorded by ATLAS, CMS, LHCb, ALICE
- **Huge** dataset ~1Pb/s before trigger selection

Goal

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

Setting the stage for LHC physics





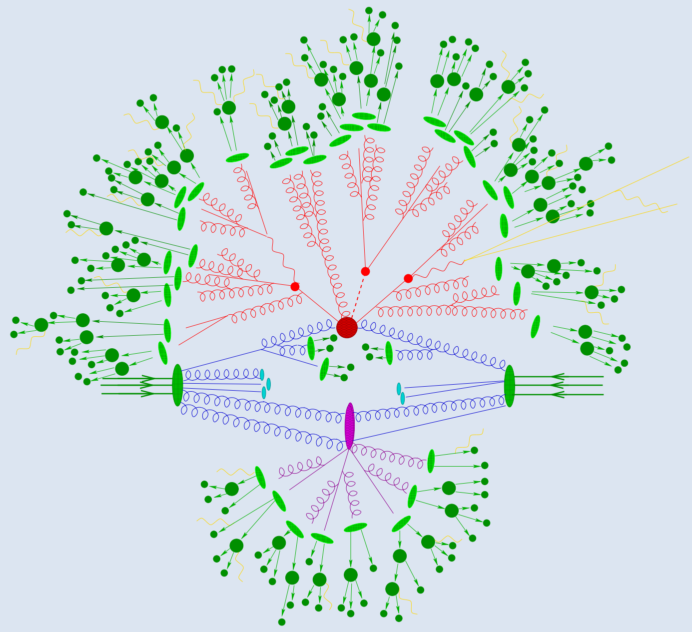
Setting
- Large Hadron Collider at CERN
- Independent proton collisions at 13 TeV
- Recorded by ATLAS, CMS, LHCb, ALICE
- **Huge** dataset ~1Pb/s before trigger selection

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

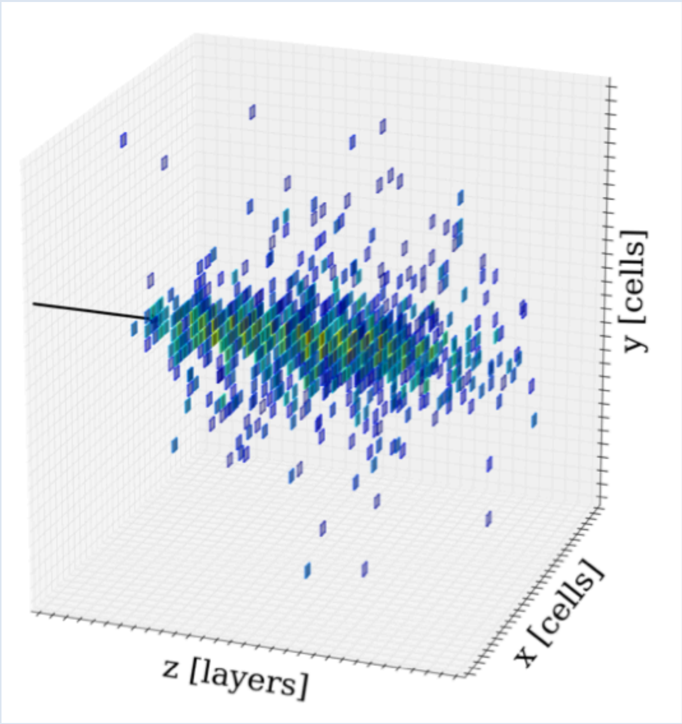
LHC analyses in a nutshell

Theory/Hypothesis

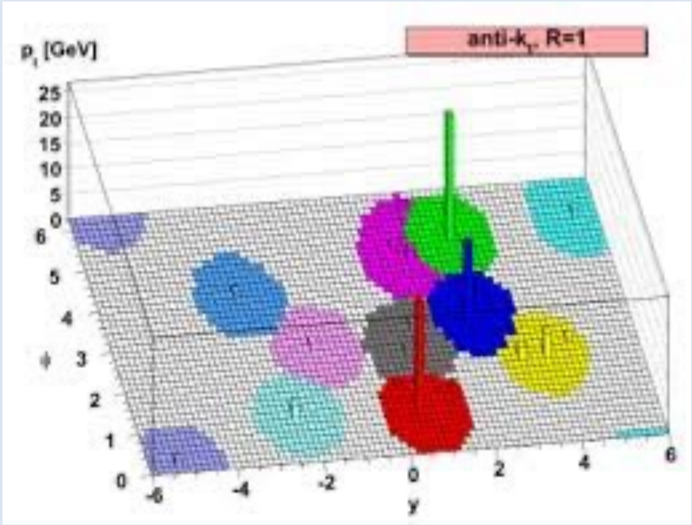
Event generation



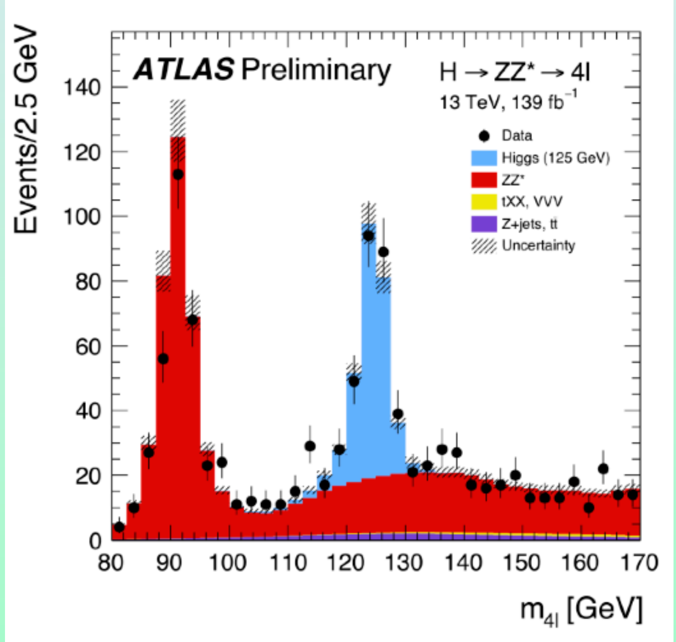
Detector simulation



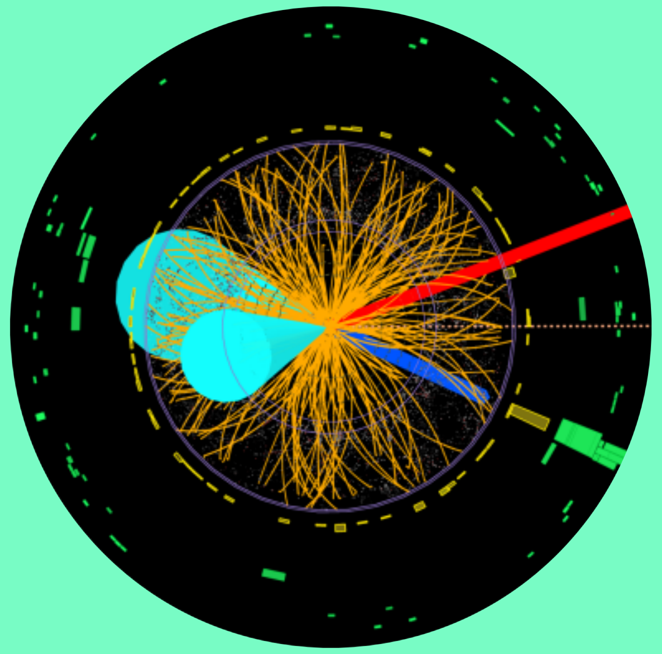
Reconstruction



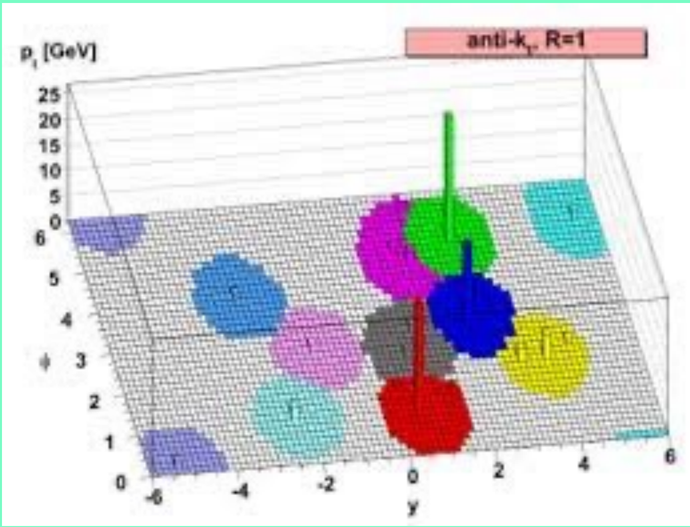
Statistical analysis



Data

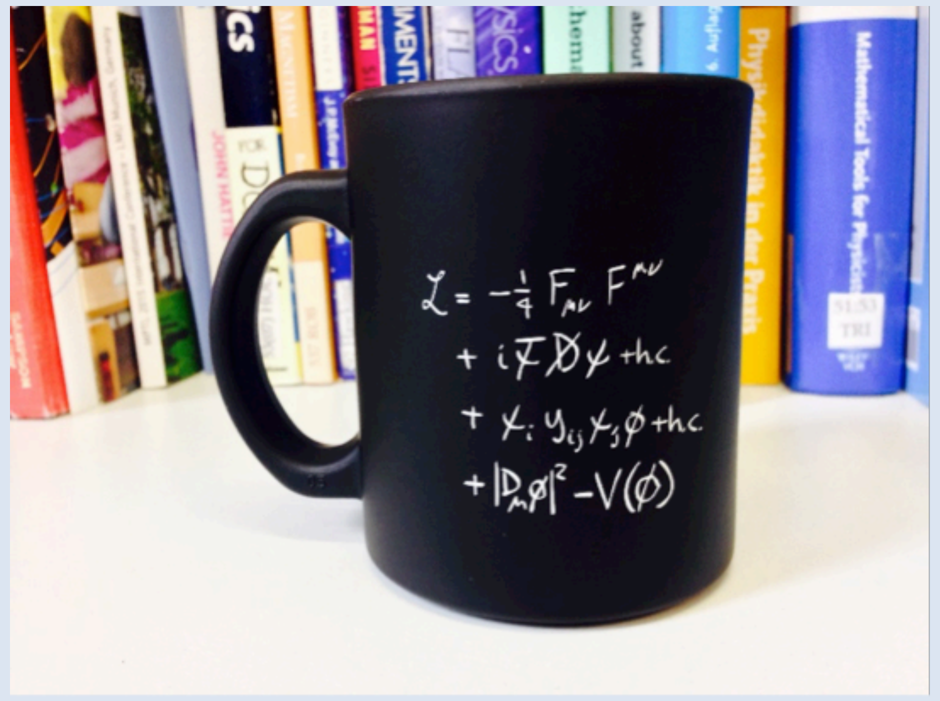


Reconstruction

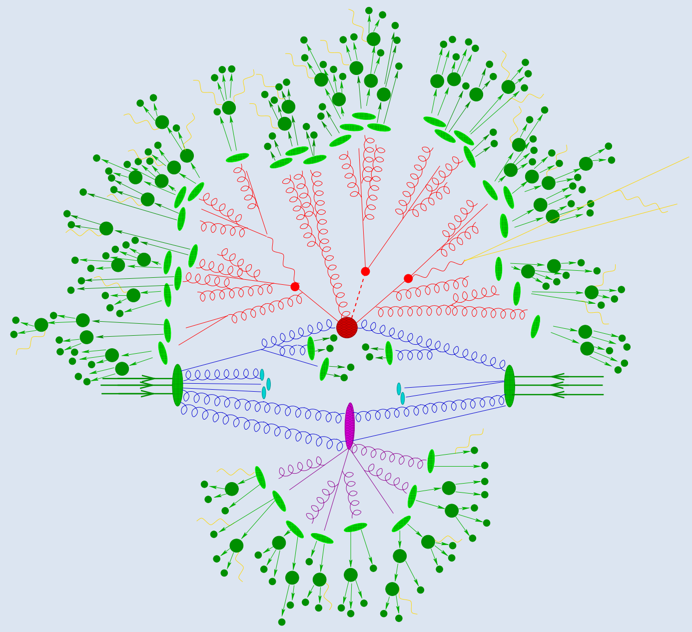


LHC analyses in a nutshell

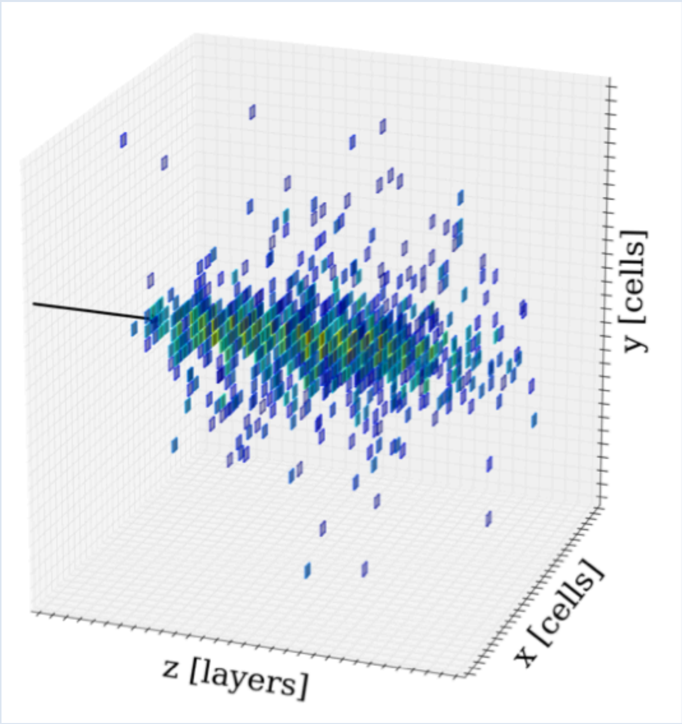
Theory/Hypothesis



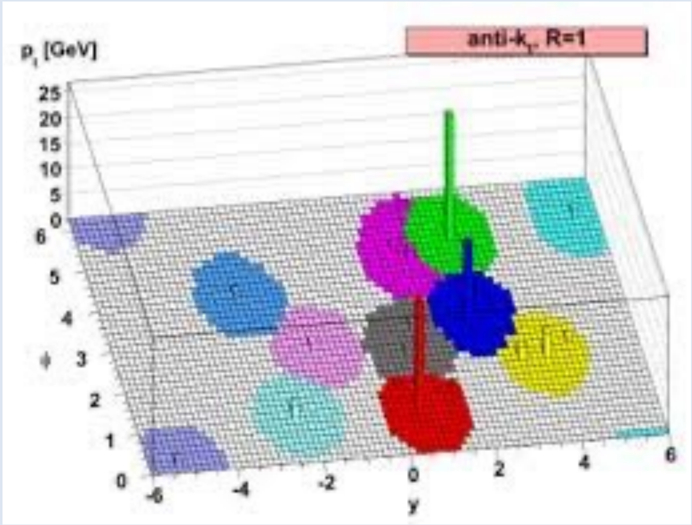
Event generation



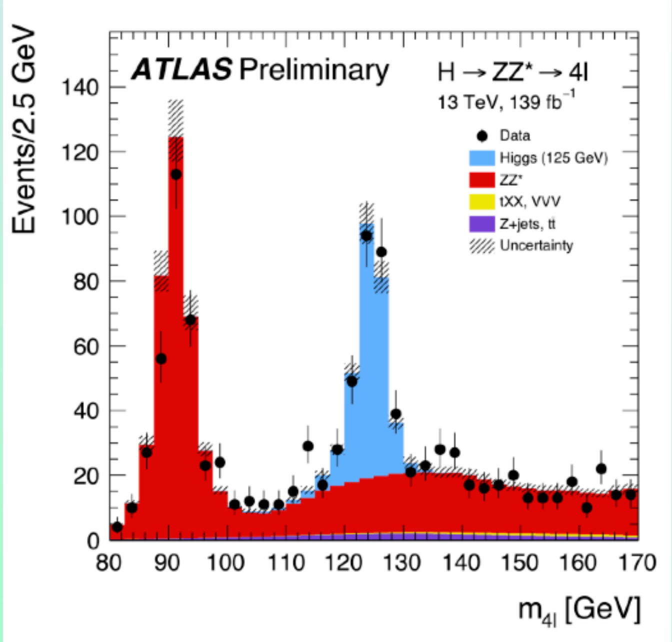
Detector simulation



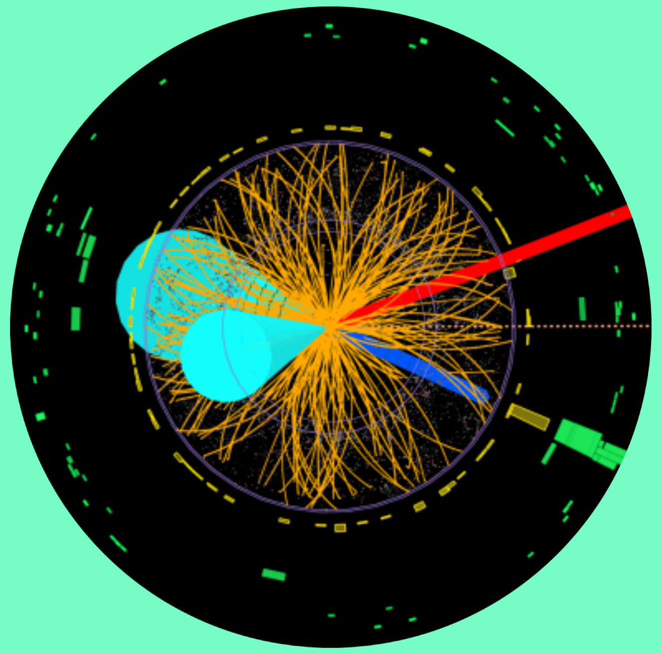
Reconstruction



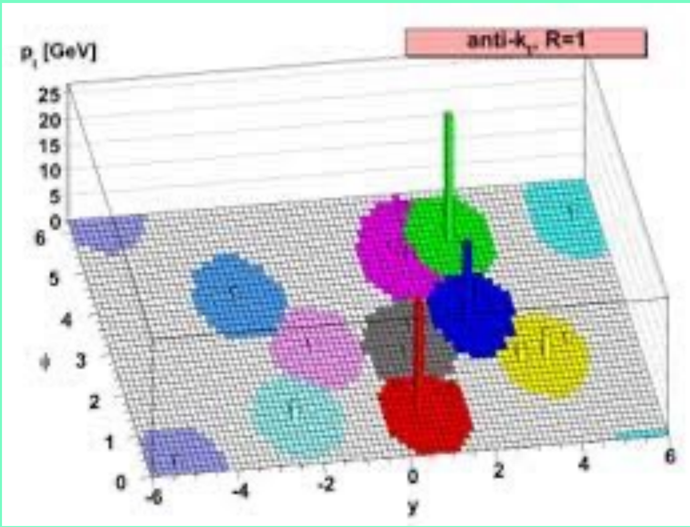
Statistical analysis



Data



Reconstruction



Limit setting for standard analyses

We want to set a limit on new physics parameter θ

$$p(data\,|\,\theta,\nu) = \prod_{i=1}^{n_{bins}} P\left(n_i\,|\,s_i(\theta,\nu)\cdot\epsilon_i(\nu) + b_i(\nu)\right)\ P\left(\nu\,|\,\text{aux. data}\right)$$

Likelihood $\rightarrow L(\theta) = p(data\,|\,\theta)$

Best estimate $\rightarrow \hat{\theta} = \arg\max_{\theta} L(\theta)$

p-value $\rightarrow p_\theta = \int_{n_{obs}}^{\infty} \mathrm{d}n\ p(n\,|\,\theta)$

68% confidence limit $\rightarrow [\theta_{\text{min}}, \theta_{\text{max}}]$ with $p_{\theta_{min}} < 0.16$

LHC discovery = 5 *sigma* $\rightarrow p < 3\times 10^{-7}$

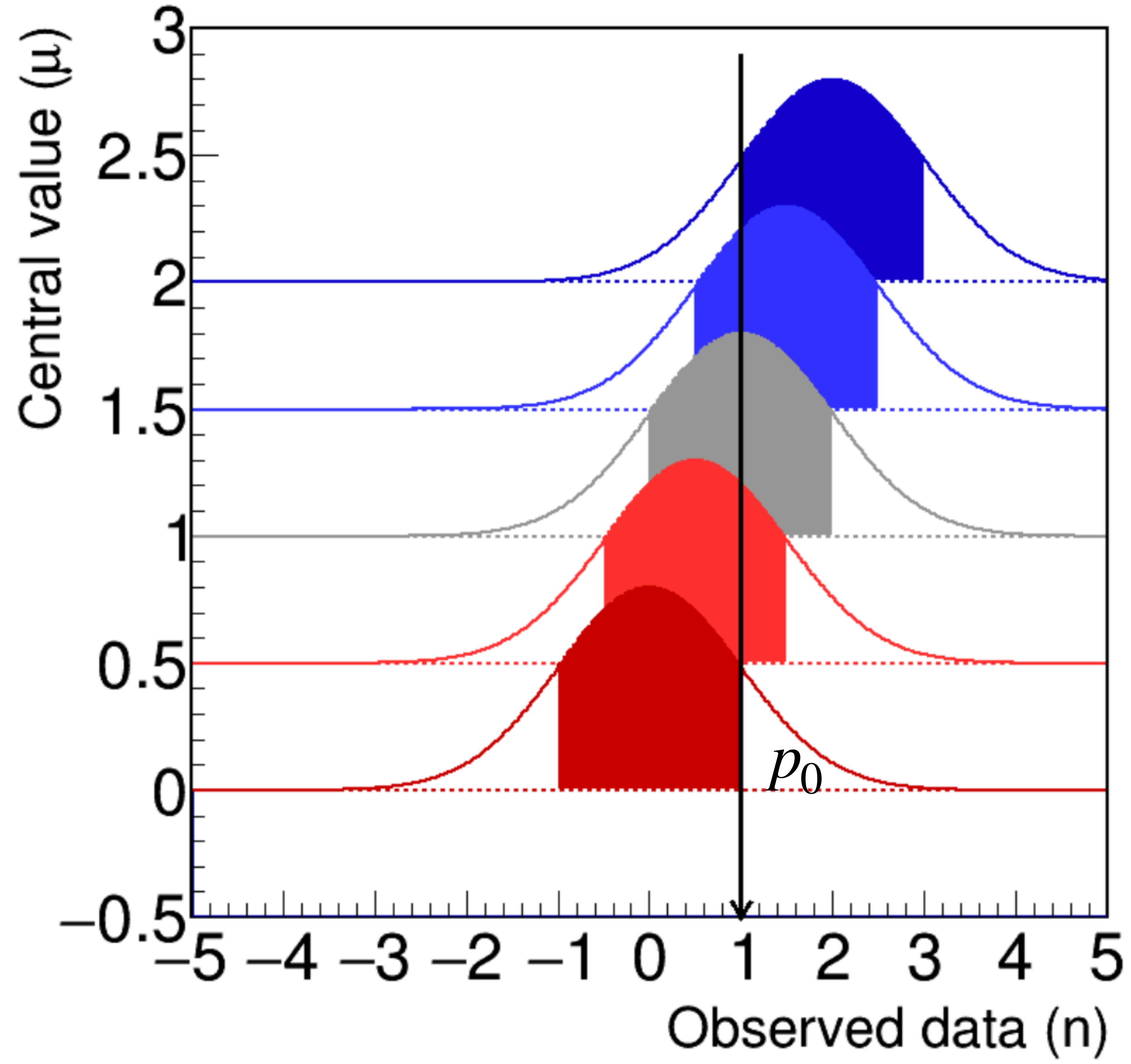


Illustration by Nicolas Berger

Not a statment on the probability of θ !!!

Types of uncertainties to build the likelihood

	Statistical uncertainties	Systematic uncertainties	Theory uncertaintes
What for?	Counting $n_{event} = p_{event} \cdot N_{collisions}$	Anything that is calibrated Efficiencies, Luminosity,…	Scale dependence prediction Two point correlations (Eg. Sherpa vs MadGraph)
Shape	Poisson	Gaussian	Flat/Gaussian
	$\frac{\lambda^x}{x!}\exp(-\lambda)$	$\frac{1}{\sigma\sqrt{2\pi}}\exp-(x-\mu)^2/2\sigma^2$	$\theta(x-\mu_{min})\theta(\mu_{max}-x)$
Correlation	Independent between bins	Correlated Covariance matrix from toys	Correlated Eg. Nuissance parameters

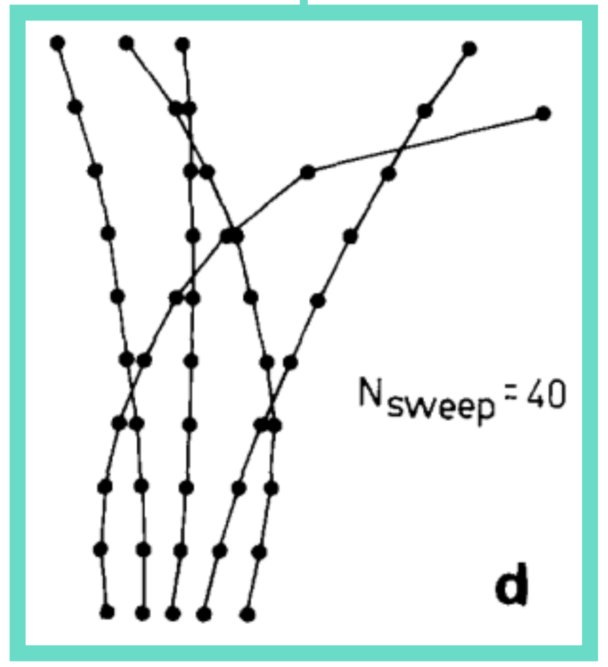
How does ML enter in this?

A short history of ML in HEP

First HEP NN papers

Track finding
Denby (LAL, Orsay) ’87
Peterson (Lund) ’88
→ Jet identification

1987/88

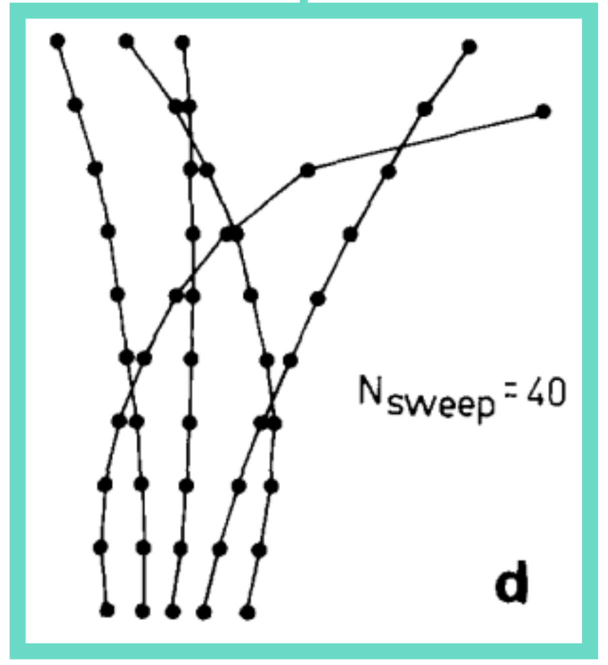


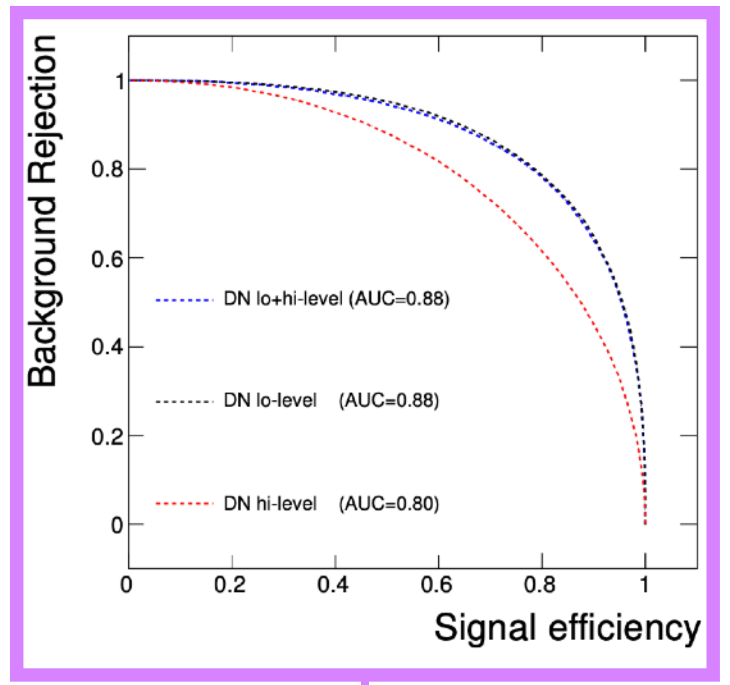
A short history of ML in HEP

First HEP NN papers

Track finding
Denby (LAL, Orsay) ’87
Peterson (Lund) ’88
→ Jet identification

1987/88





2014

Relaunch

Deep Learning in HEP
Signal vs Background

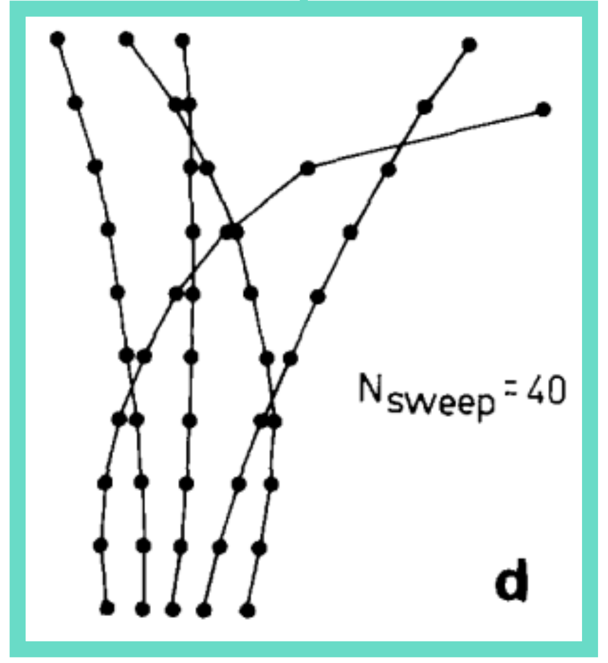
P. Baldi, P. Sadowski,
D. Whiteson

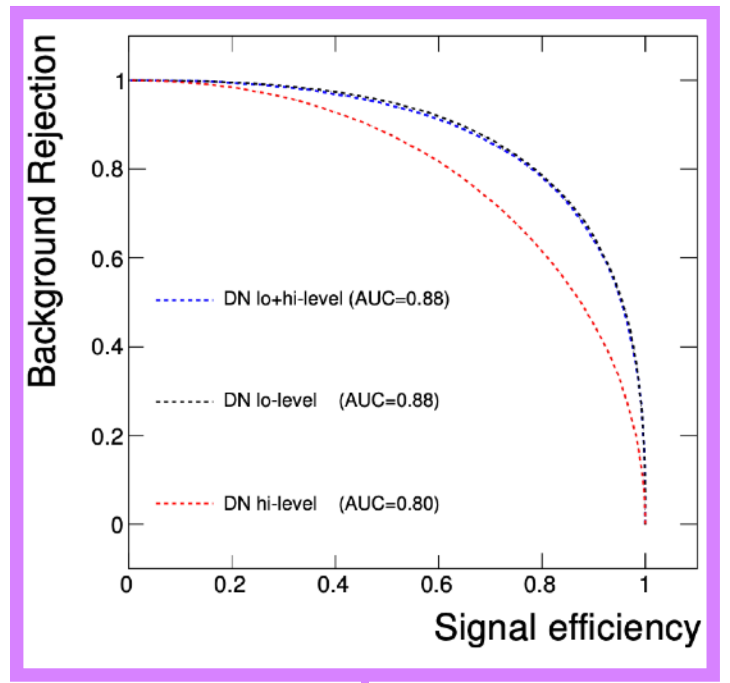
A short history of ML in HEP

First HEP NN papers

Track finding
Denby (LAL, Orsay) ’87
Peterson (Lund) ’88
→ Jet identification

1987/88





2014

Relaunch

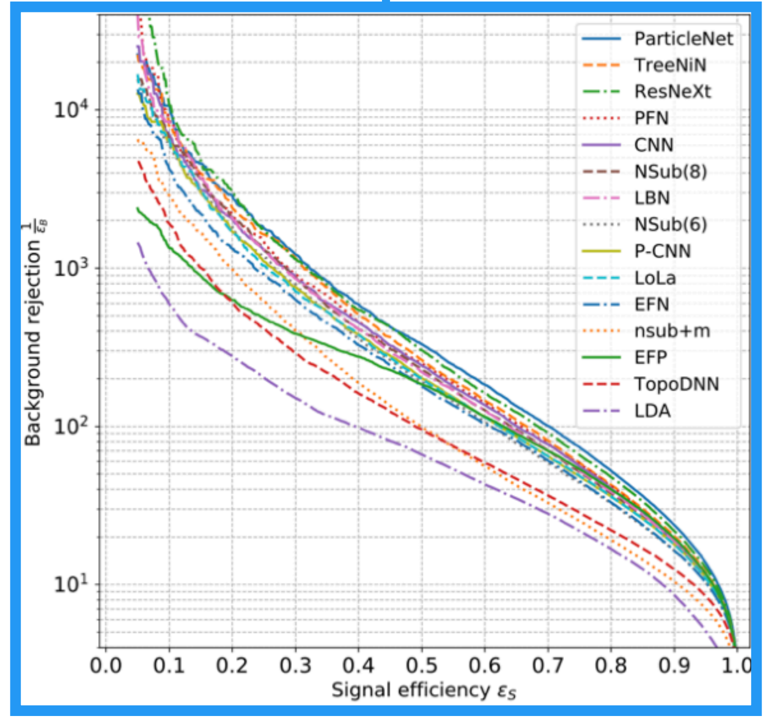
Deep Learning in HEP
Signal vs Background

P. Baldi, P. Sadowski,
D. Whiteson

First community Paper

Machine Learning Landscape of TopTagging
G. Kasieczka, et al.

2019



A short history of ML in HEP

First HEP NN papers

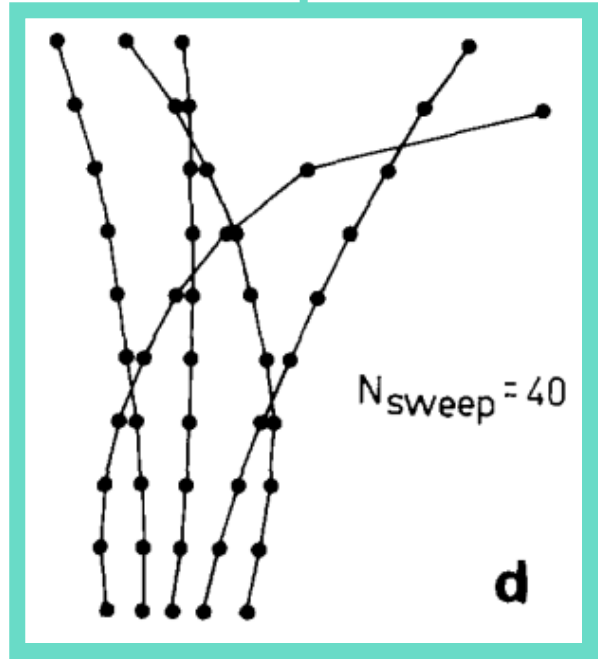
Track finding

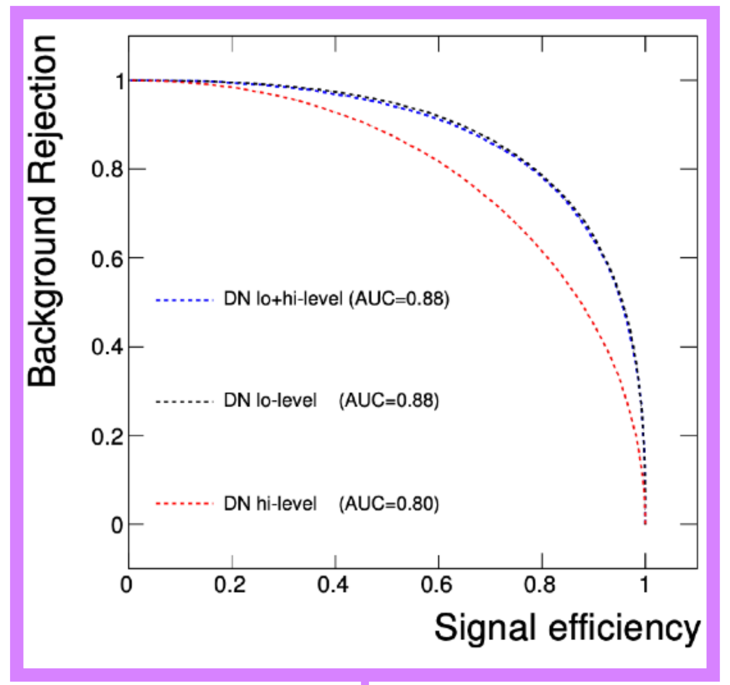
Denby (LAL, Orsay) ’87

Peterson (Lund) ’88

→ Jet identification

1987/88





2014

Relaunch

Deep Learning in HEP

Signal vs Background

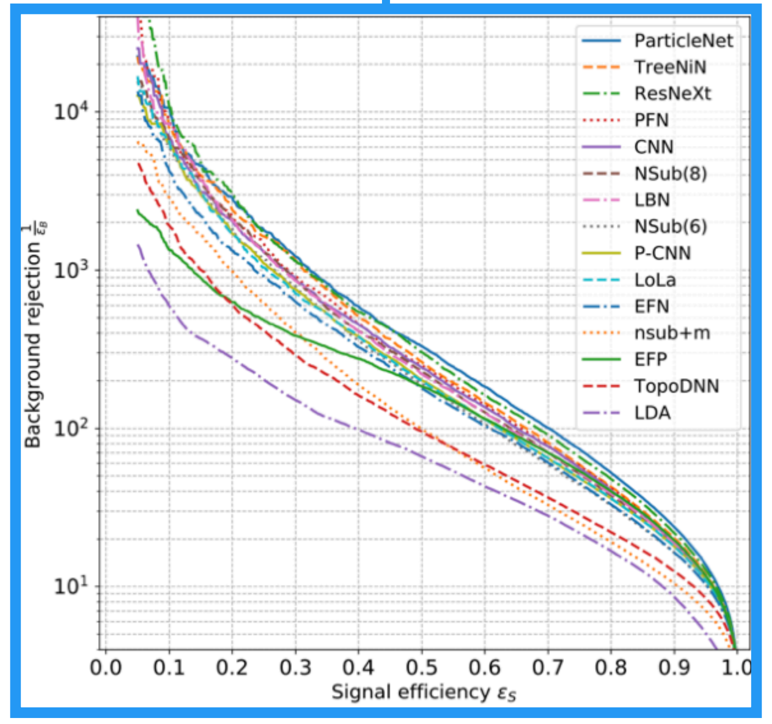
P. Baldi, P. Sadowski, D. Whiteson

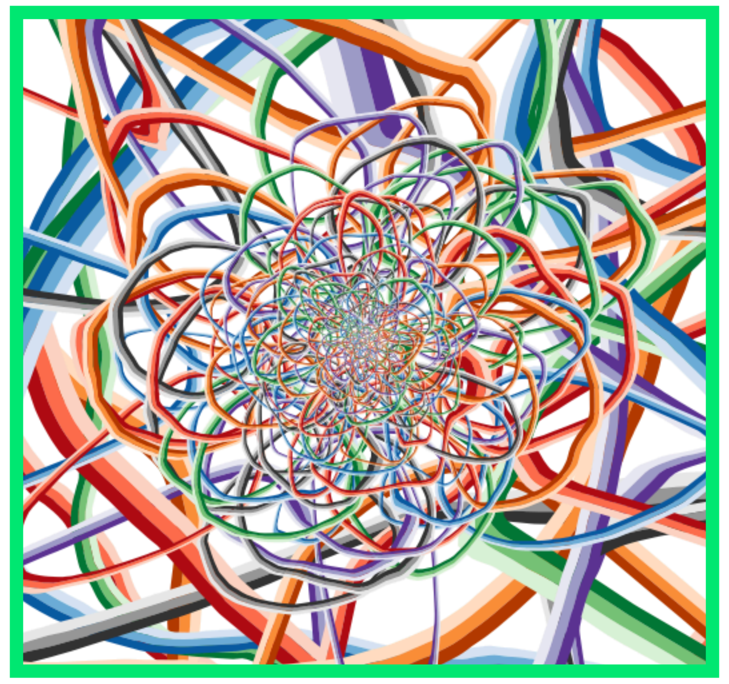
First community Paper

Machine Learning Landscape of TopTagging

G. Kasieczka, et al.

2019





2019

First attempts at “understanding” neural networks

Deep Thinking

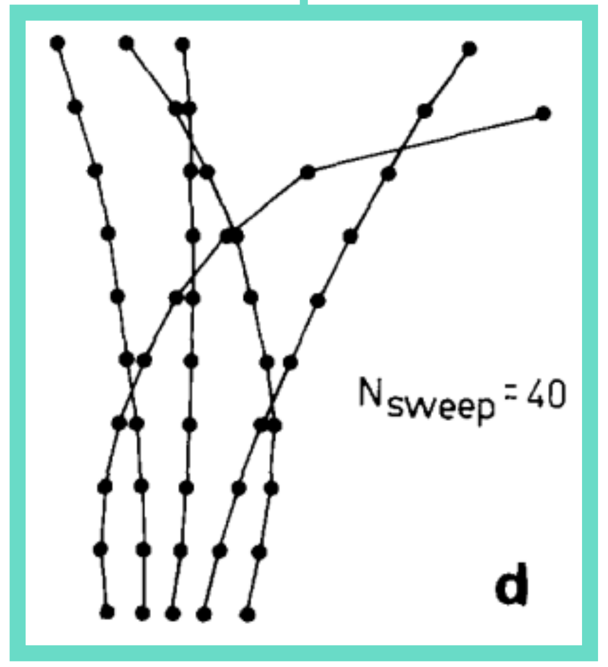
J. Thaler

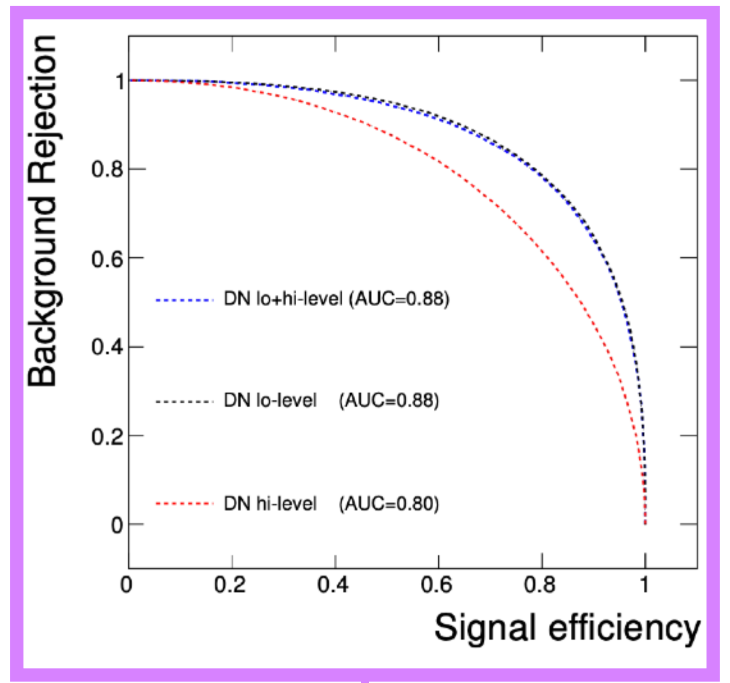
A short history of ML in HEP

First HEP NN papers

Track finding
Denby (LAL, Orsay) ’87
Peterson (Lund) ’88
→ Jet identification

1987/88





2014

Relaunch

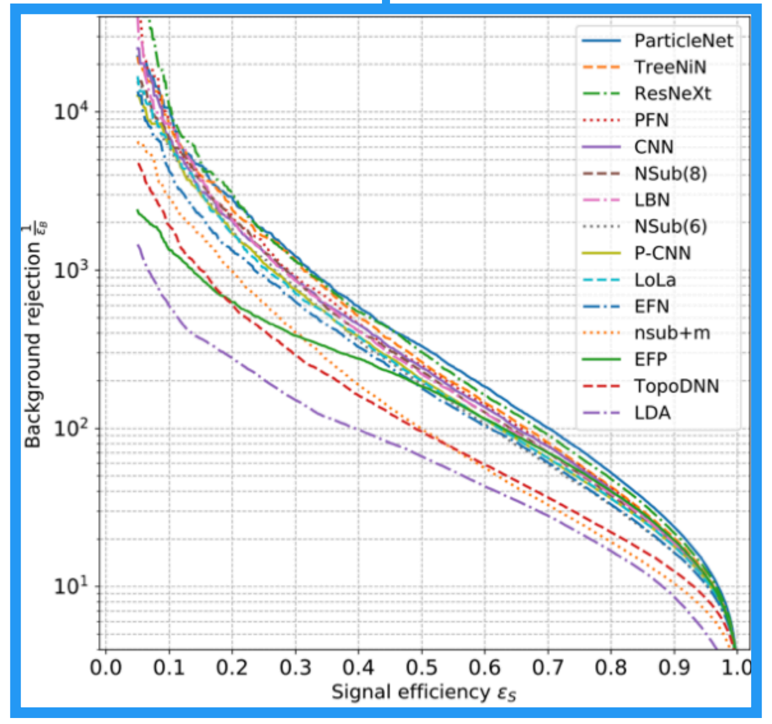
Deep Learning in HEP
Signal vs Background

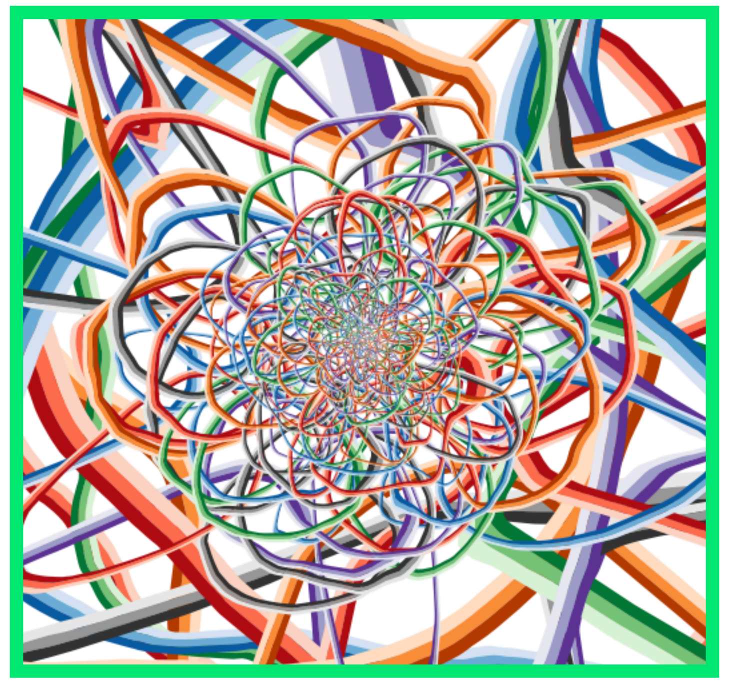
P. Baldi, P. Sadowski,
D. Whiteson

First community Paper

Machine Learning Landscape of TopTagging
G. Kasieczka, et al.

2019





2019

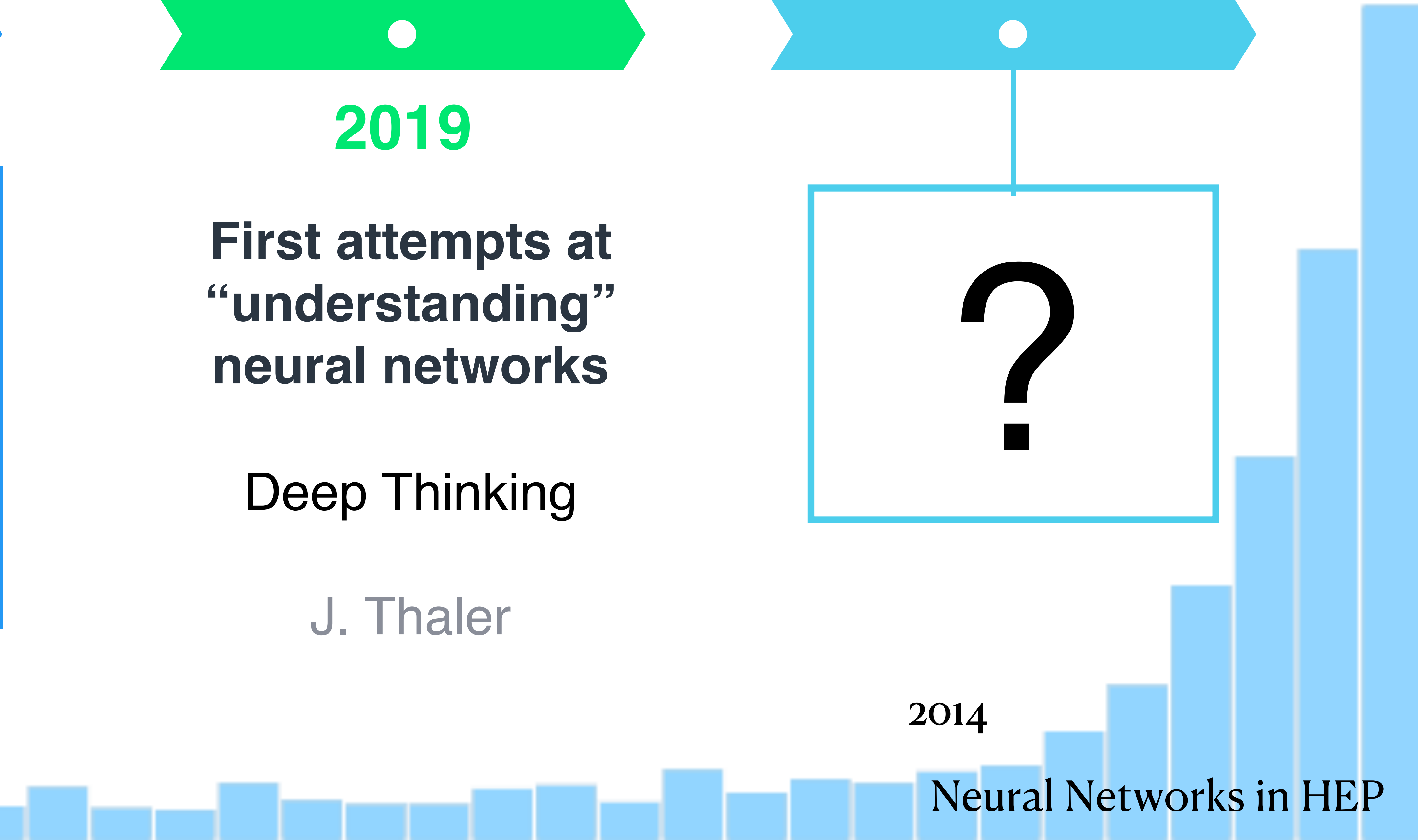
First attempts at “understanding” neural networks

Deep Thinking

J. Thaler

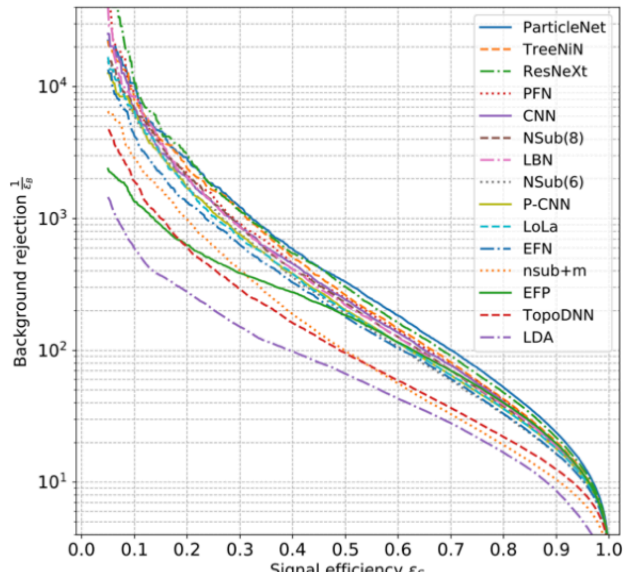
Today

?



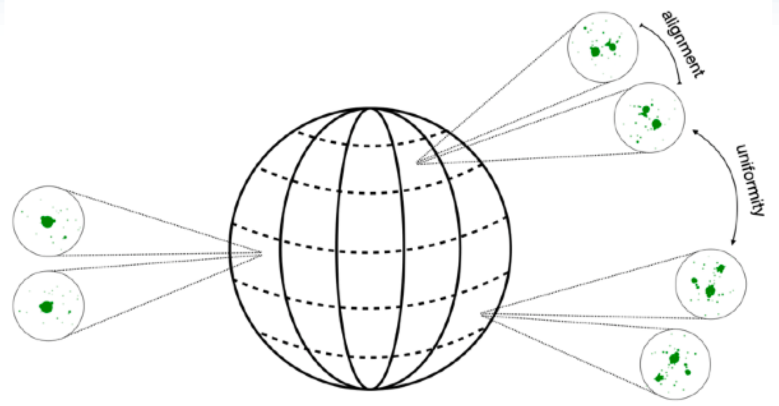
ML for big data in particle physics

Top tagging



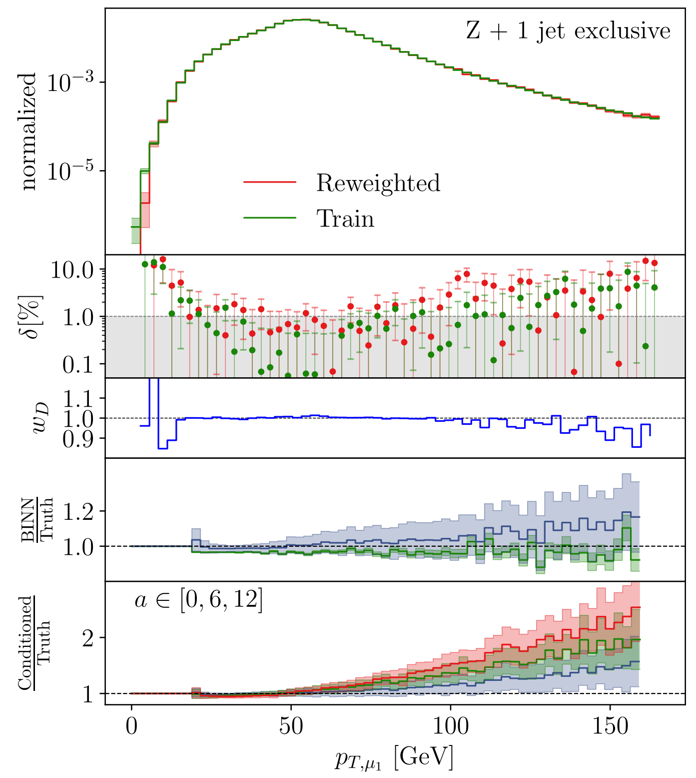
G. Kasieczka et al. [1902.09914]

Anomaly detection



B. Dillon et al. [2108.04253]

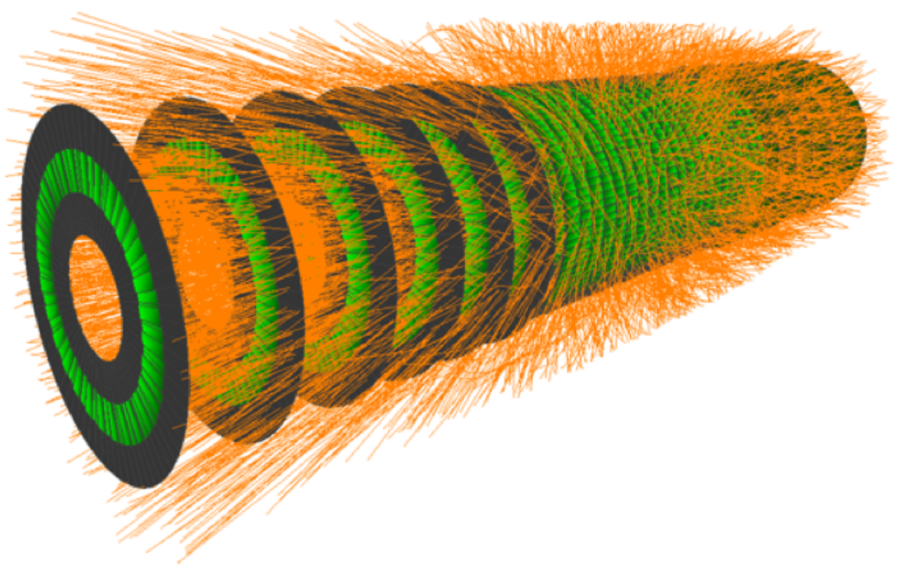
Event generation



A. Butter et al. [2110.13632]

Track reconstruction

Kaggle challenge



Classification

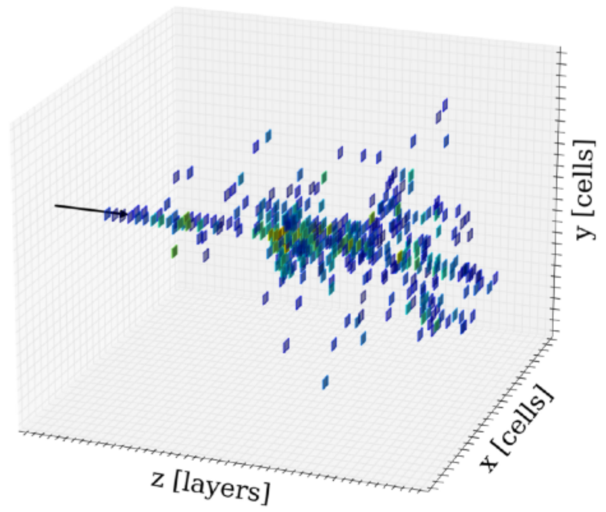
Generative models

Graph networks

Bayesian networks

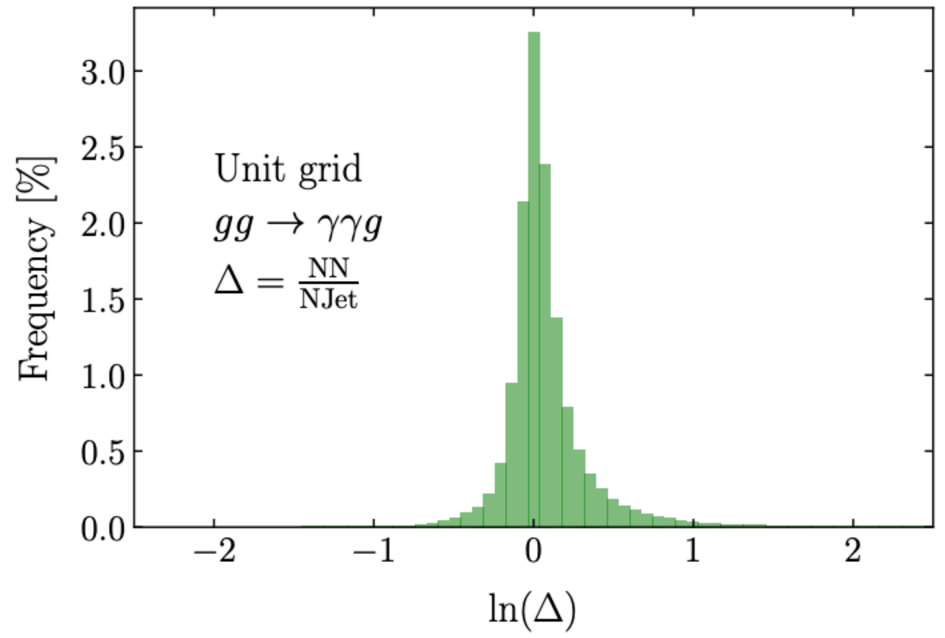
Regression

Detector simulation



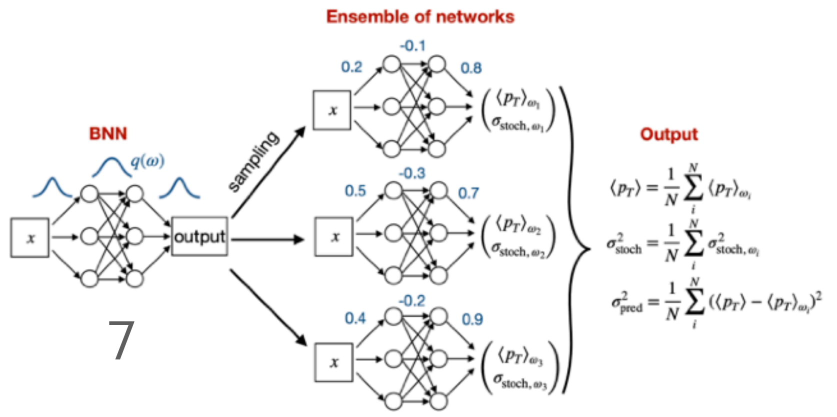
E. Buhmann et al. [2112.09709]

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/

ML examples and their uncertainties

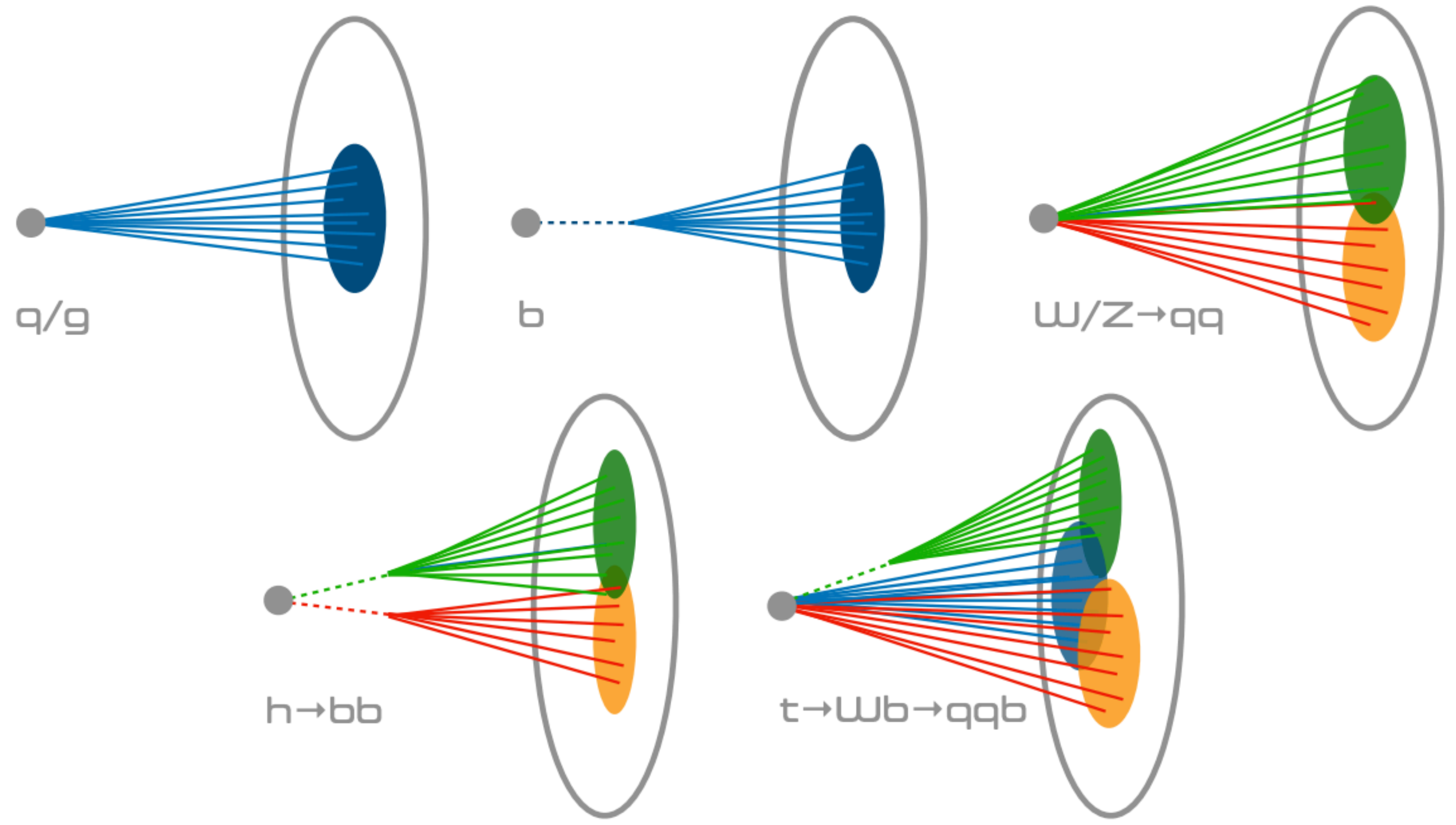
Classification

Simulations

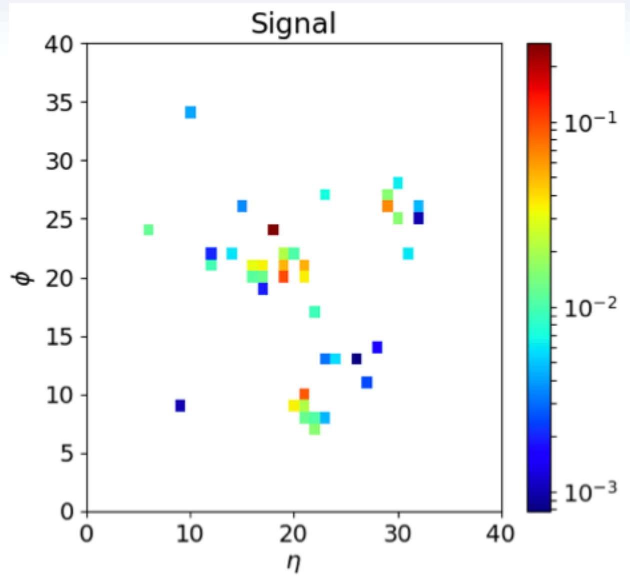
Unfolding

Jet classification

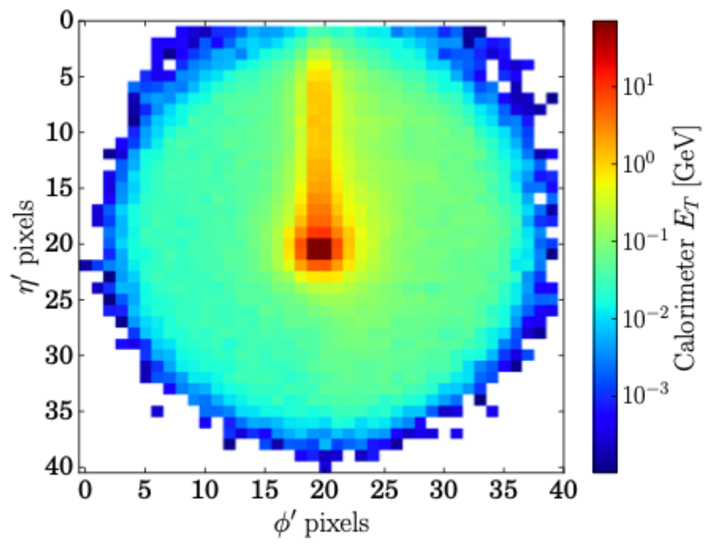
- How to distinguish **top** from **QCD** jets?
- Immensely important for top & Higgs physics studies



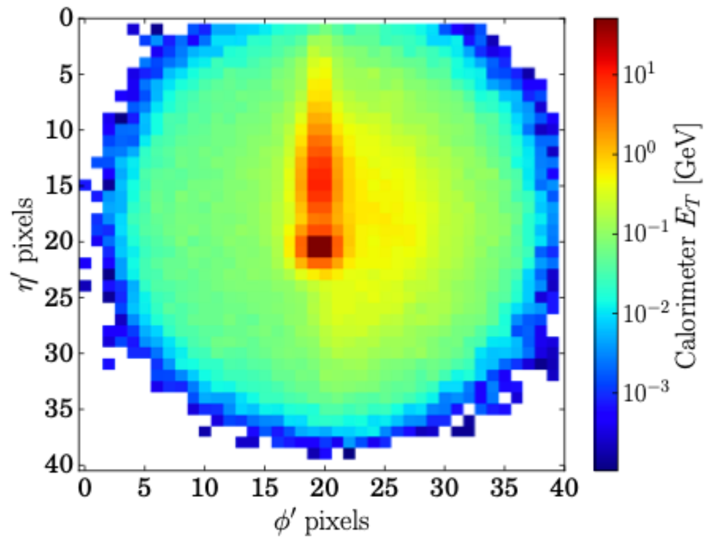
E. Moreno et al. [1909.12285]



Single jet



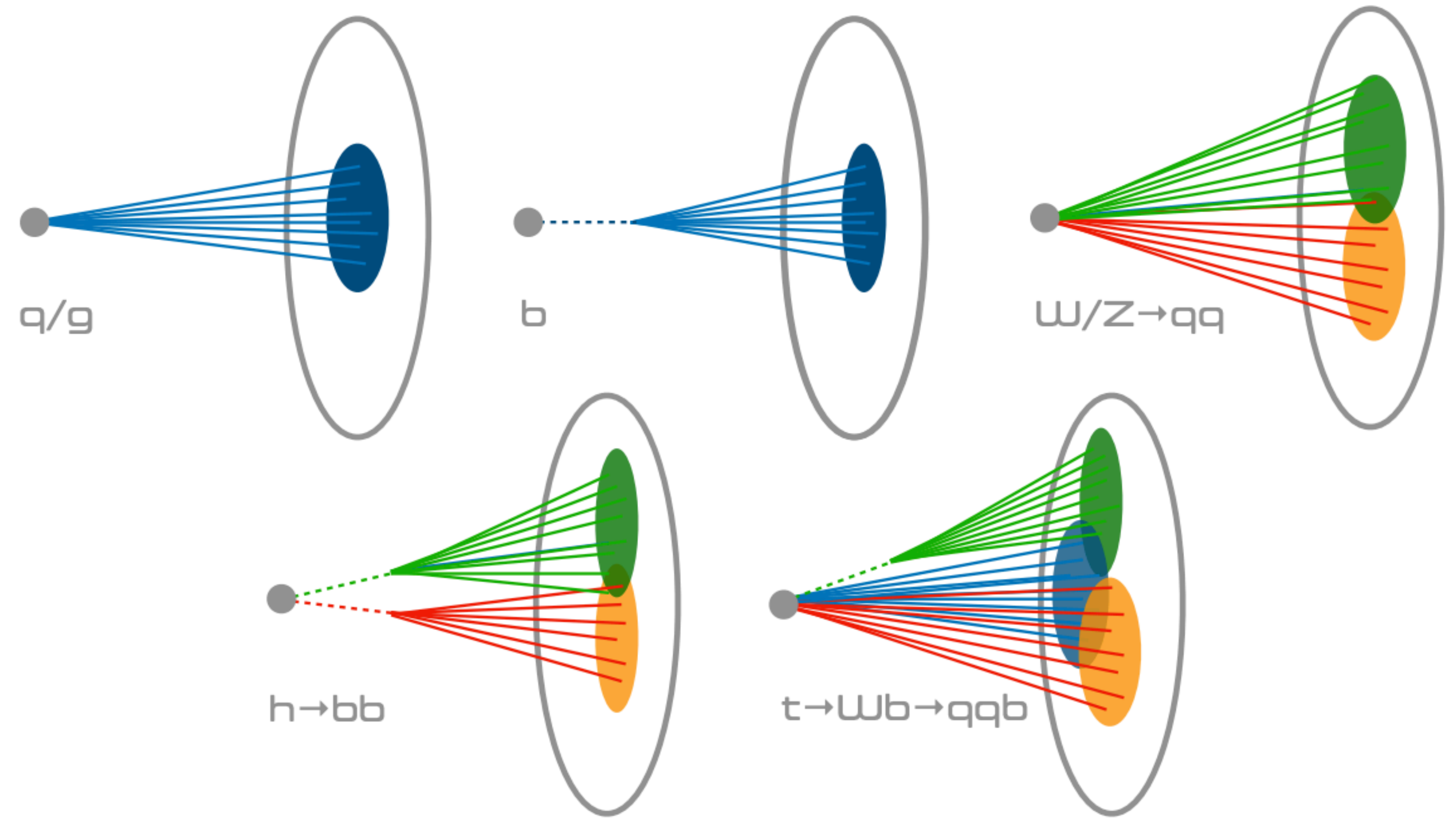
QCD jet averaged



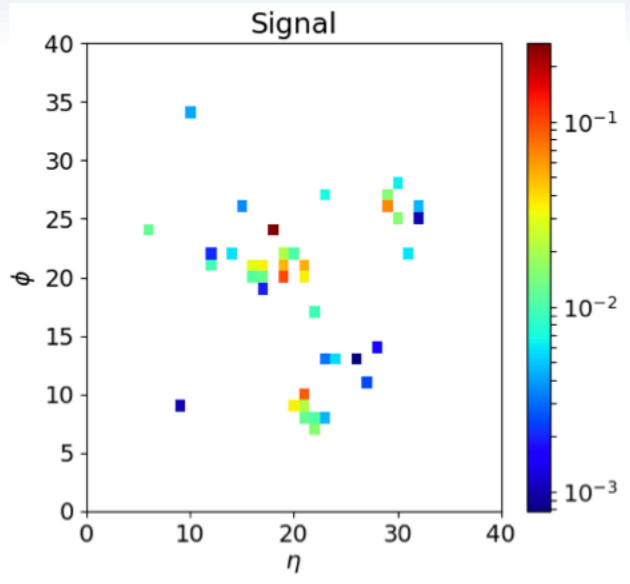
top jet Averaged

Jet classification

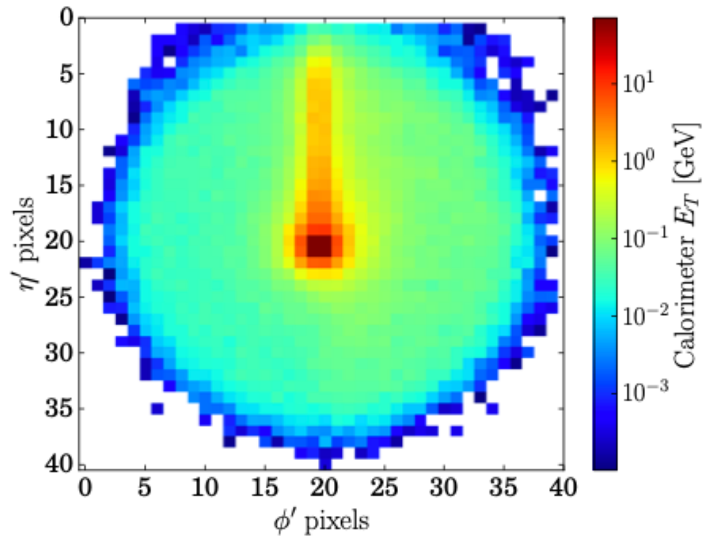
- How to distinguish **top** from **QCD** jets?
- Immensely important for top & Higgs physics studies
- Standard supervised classification task



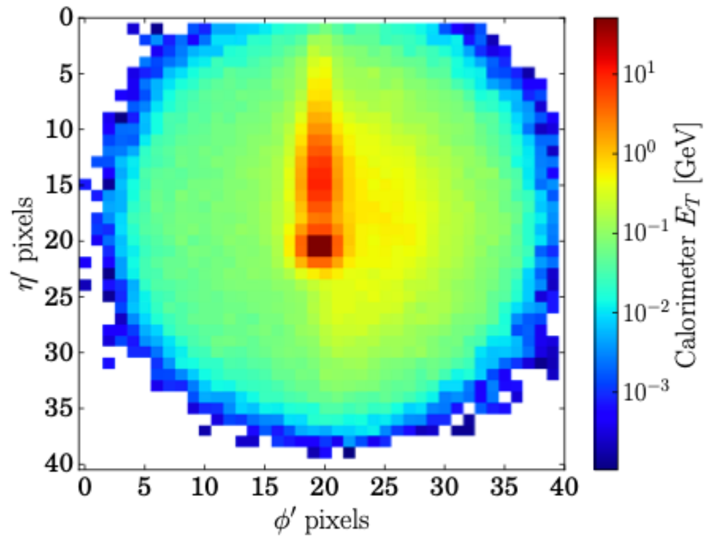
E. Moreno et al. [1909.12285]



Single jet



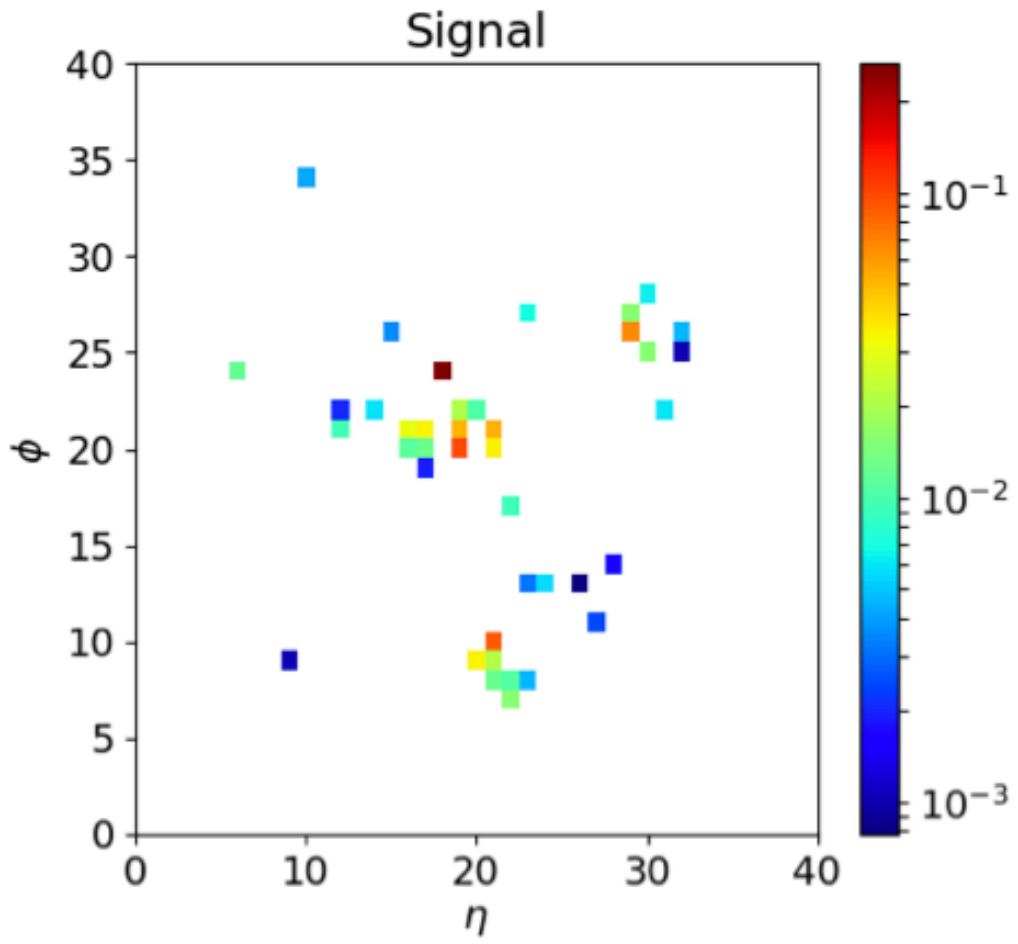
QCD jet averaged

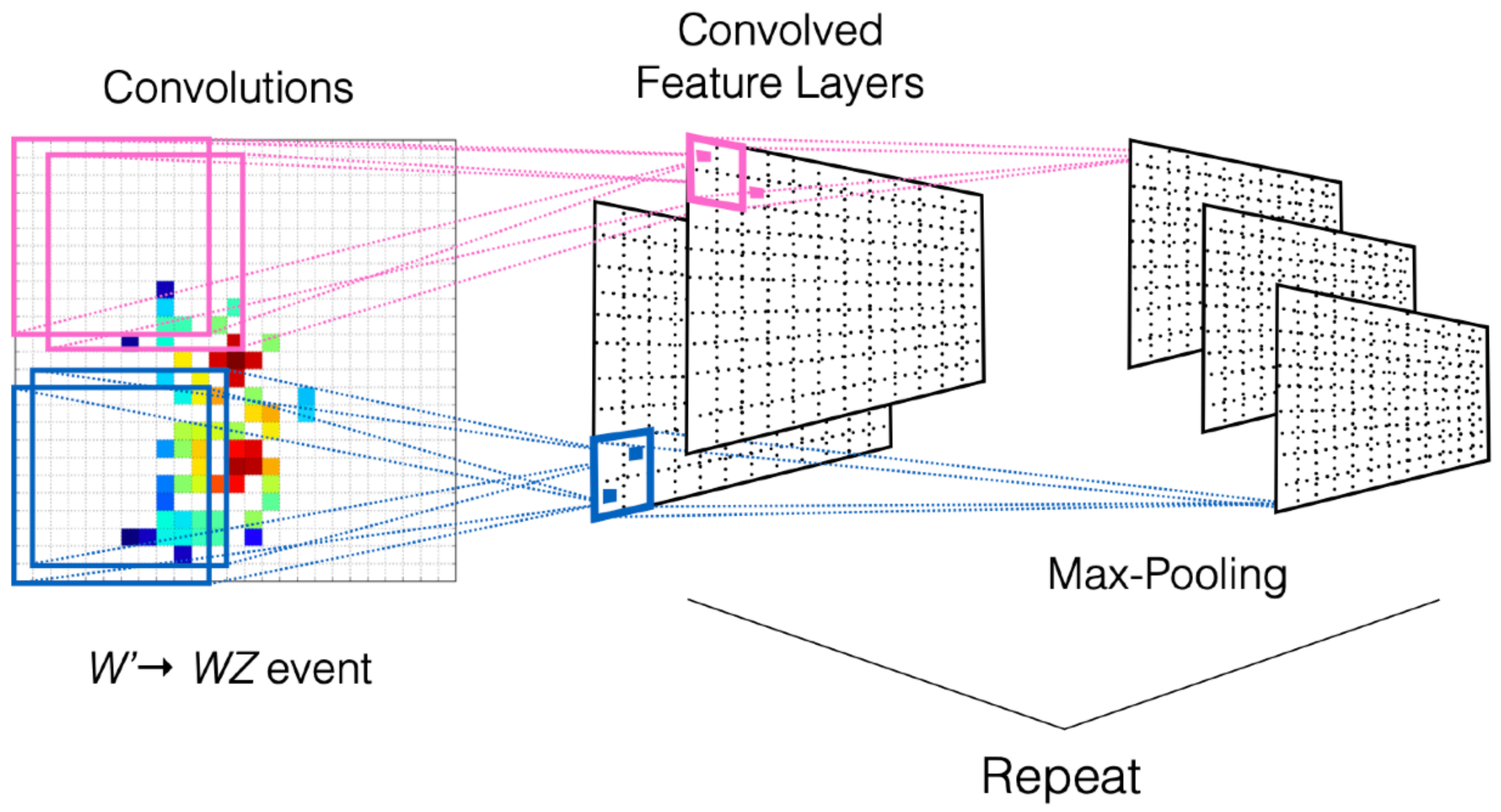


top jet Averaged

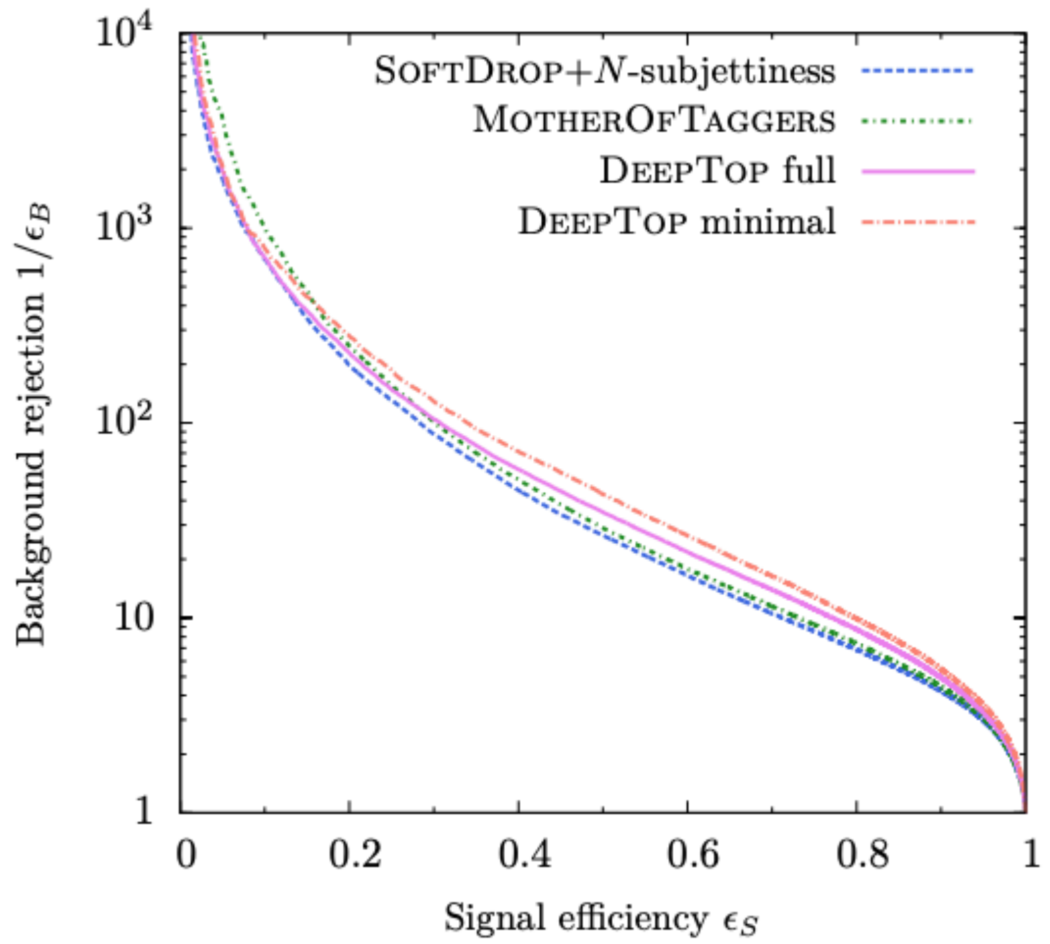
Jet representation I

Single jet





L. Oliveira et al. [1511.05190]



G. Kasieczka et al. [1701.08784]

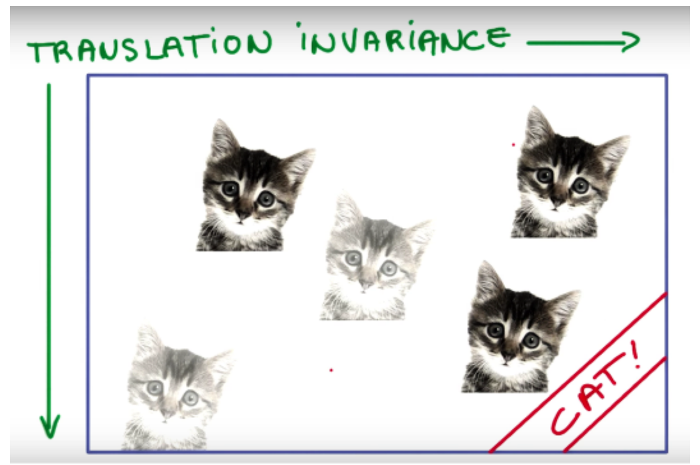
CNNs

Use **equivariance** principle for translation t:

$$\mathrm{CNN}(t(x)) = t(\mathrm{CNN}(x))$$

Pooling layer

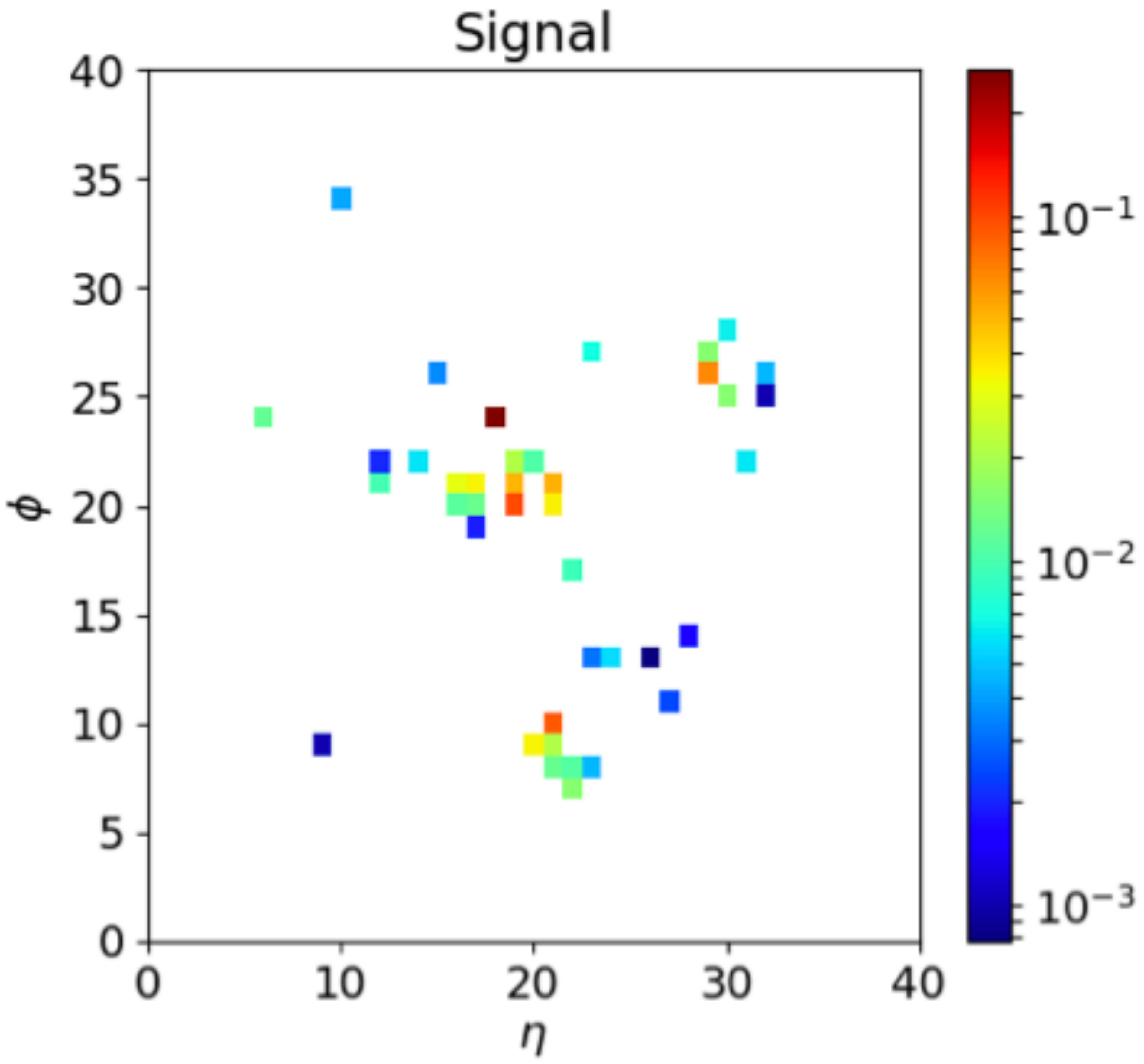
$\rightarrow$ **Invariance**



Our image ain't a very good image...

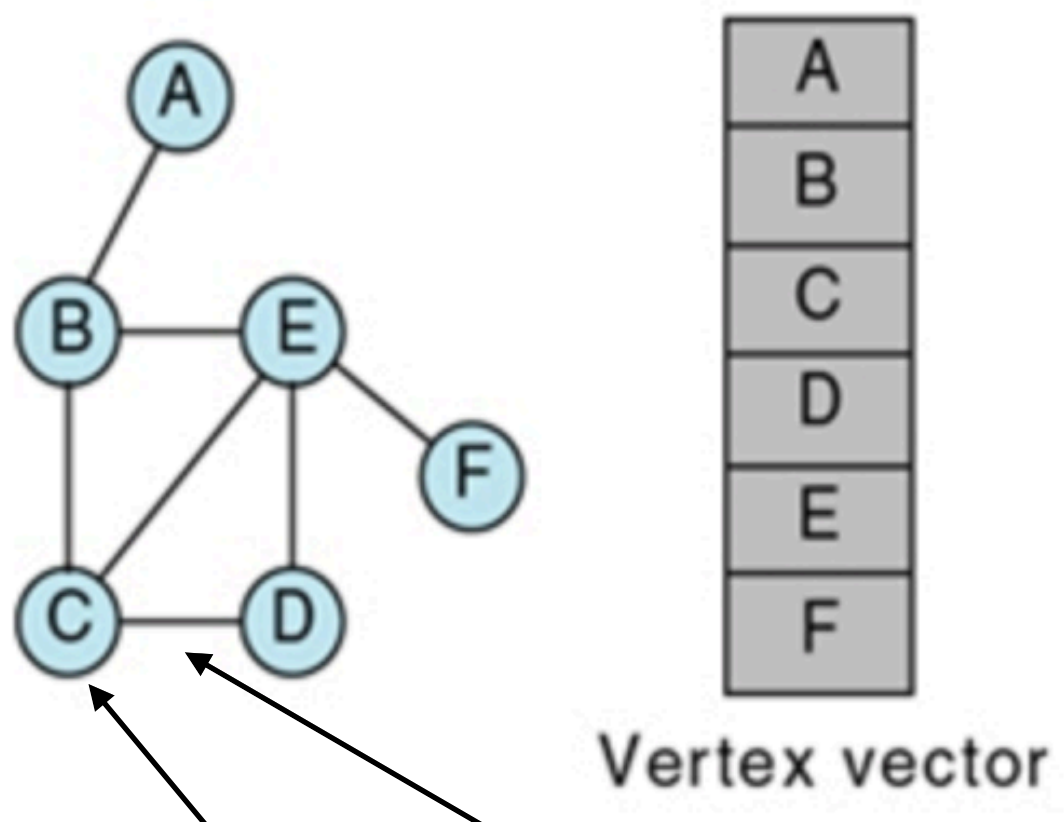
No continuity, no edges, no cats....

VS



Jet representation II

How to represent a graph

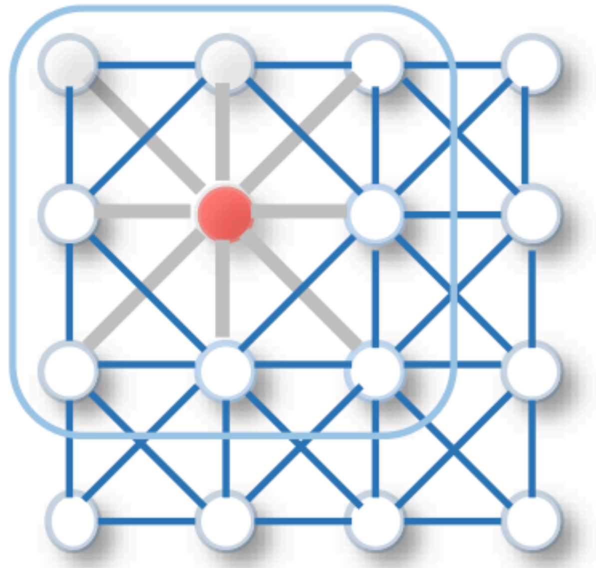


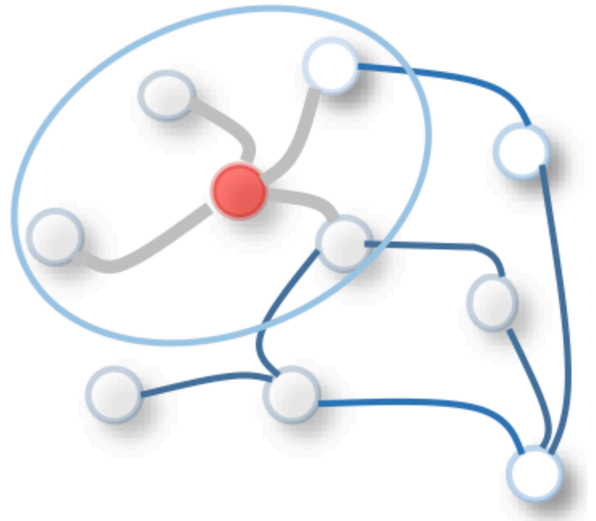
	A	B	C	D	E	F
A	0	1	0	0	0	0
B	1	0	1	0	1	0
C	0	1	0	1	1	0
D	0	0	1	0	1	0
E	0	1	1	1	0	1
F	0	0	0	0	1	0

Adjacency matrix

Node Edge

Image vs Graph





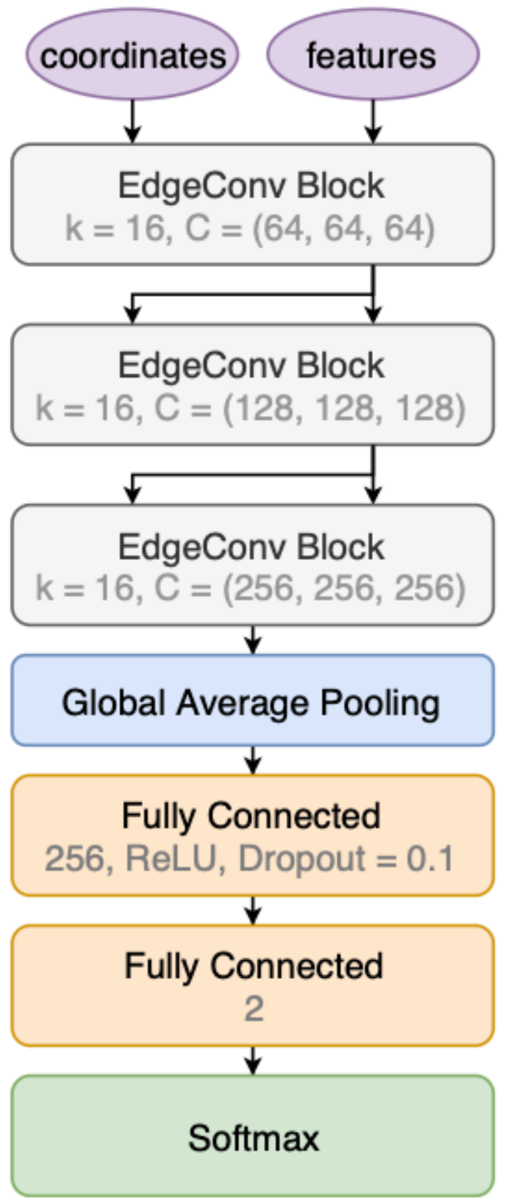
pixels → node

neighbouring pixel → neighbouring node (graph edges)

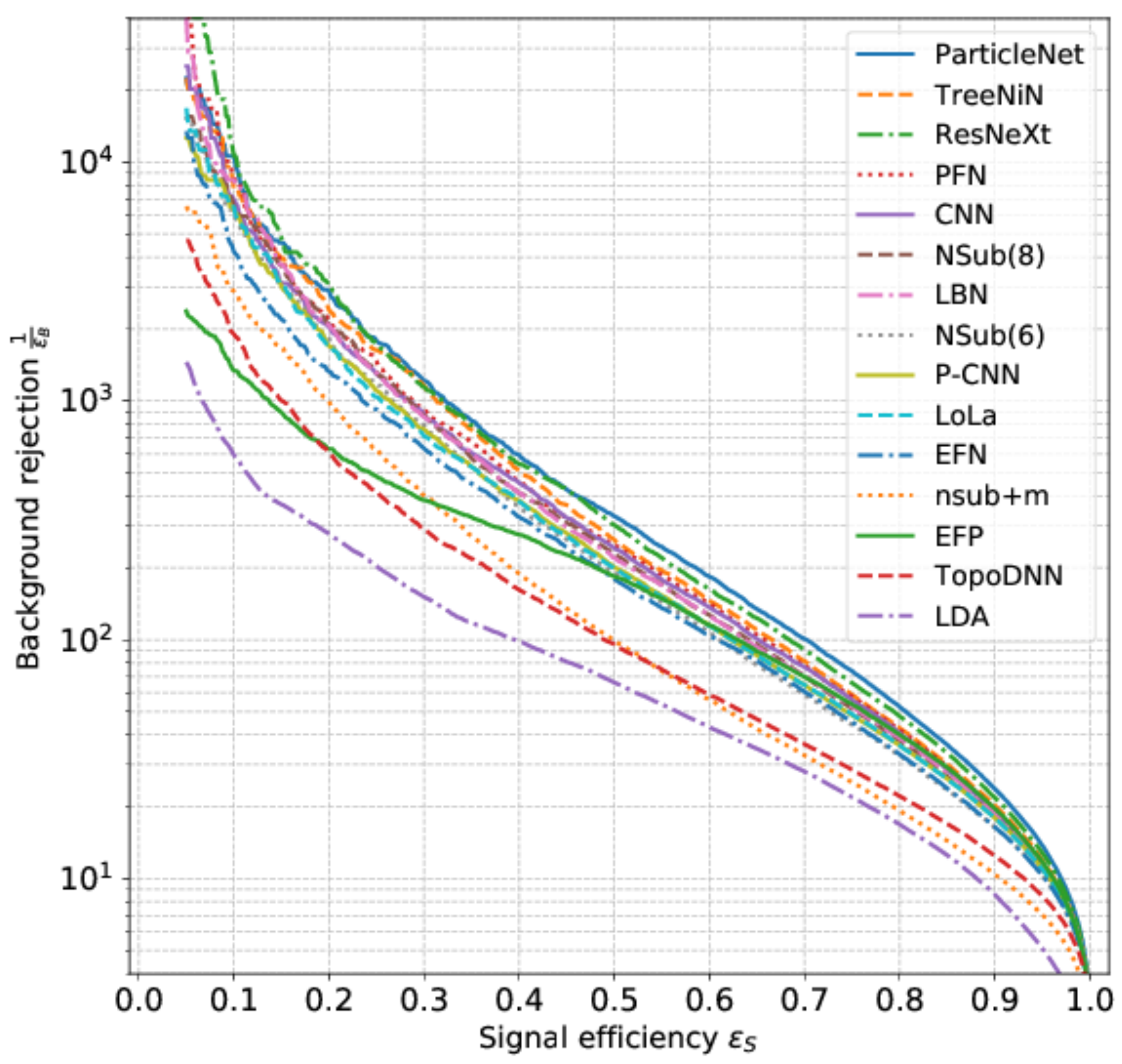
CNN → edge convolution

$$\vec{x}'_i = \frac{1}{k}\sum_{j=1}^{k} h_\Theta(\vec{x}_i, \vec{x}_{i_j} - \vec{x}_i)$$

The ML landscape of top taggers



(a) ParticleNet



H. Qu, L. Gouskos [1902.08570]

Difference in performance for various different approaches!

Where are the uncertainties?

What if the network is not perfect?

A perfect network has learned the likelihood ratio $\dfrac{p(x\,|\,top)}{p(x\,|\,QCD)}$

What if the network is not perfect?

Classification loss function

$$\mathcal{L} = \sum_{x_i} -\log C(x_i)\, y_i - \log(1 - C(x_i))\,(1 - y_i)$$

$$= -\int \mathrm{d}x\, p_{top}(x)\, \log C(x) + p_{QCD}\, \log(1 - C(x))$$

Variance yields $\rightarrow \dfrac{p_{top}(x)}{p_{QCD}(x)} = \dfrac{C(x)}{1 - C(x)}$

A perfect network has learned the likelihood ratio $\dfrac{p(x\,|\,top)}{p(x\,|\,QCD)}$

What if the network is not perfect?

A suboptimal network will label more tops ”wrong”

Classification loss function

$$\mathscr{L} = \sum_{x_i} -\log C(x_i)\, y_i - \log(1 - C(x_i))\,(1 - y_i)$$

$$= -\int \mathrm{d}x\, p_{top}(x)\ \log C(x) + p_{QCD}\ \log(1 - C(x))$$

Variance yields $\rightarrow \dfrac{p_{top}(x)}{p_{QCD}(x)} = \dfrac{C(x)}{1 - C(x)}$

A perfect network has learned the likelihood ratio $\dfrac{p(x\,|\,top)}{p(x\,|\,QCD)}$

What if the network is not perfect?

Classification loss function

$$\mathcal{L} = \sum_{x_i} -\log C(x_i)\, y_i - \log(1 - C(x_i))\,(1 - y_i)$$

$$= -\int \mathrm{d}x\, p_{top}(x)\ \log C(x) + p_{QCD}\ \log(1 - C(x))$$

Variance yields $\rightarrow \dfrac{p_{top}(x)}{p_{QCD}(x)} = \dfrac{C(x)}{1 - C(x)}$

A perfect network has learned the likelihood ratio $\dfrac{p(x\,|\,top)}{p(x\,|\,QCD)}$

A suboptimal network will label more tops "wrong"

Applies to prediction & data!

Equivalent to poor efficiency in

$$p(data\,|\,\theta,\nu) = \prod_{i=1}^{n_{bins}} P\left(n_i\,|\,s_i(\theta,\nu)\cdot\epsilon_i(\nu) + b_i(\nu)\right)\ P\left(\nu\,|\,\text{aux. data}\right)$$

What if the network is not perfect?

What is the network supposed to learn?

Classification loss function

$$\mathscr{L} = \sum_{x_i} - \log C(x_i)\, y_i - \log(1 - C(x_i))\,(1 - y_i)$$

$$= - \int \mathrm{d}x\, p_{top}(x)\, \log C(x) + p_{QCD}\, \log(1 - C(x))$$

Variance yields $\rightarrow \dfrac{p_{top}(x)}{p_{QCD}(x)} = \dfrac{C(x)}{1 - C(x)}$

A perfect network has learned the likelihood ratio $\dfrac{p(x\,|\,top)}{p(x\,|\,QCD)}$

A suboptimal network will label more tops ”wrong”

Applies to prediction & data!

Equivalent to poor efficiency in

$$p(data\,|\,\theta,\nu) = \prod_{i=1}^{n_{bins}} P\left(n_i\,|\,s_i(\theta,\nu)\cdot\epsilon_i(\nu) + b_i(\nu)\right)\; P\left(\nu\,|\,\text{aux. data}\right)$$

What if the network is not perfect?

What is the network supposed to learn?

Classification loss function

$$\mathscr{L} = \sum_{x_i} -\log C(x_i)\, y_i - \log(1 - C(x_i))\,(1 - y_i)$$

$$= -\int \mathrm{d}x\, p_{top}(x)\, \log C(x) + p_{QCD}\, \log(1 - C(x))$$

Variance yields $\rightarrow \dfrac{p_{top}(x)}{p_{QCD}(x)} = \dfrac{C(x)}{1 - C(x)}$

A perfect network has learned the likelihood ratio $\dfrac{p(x\,|\,top)}{p(x\,|\,QCD)}$

A suboptimal network will label more tops "wrong"

Applies to prediction & data!

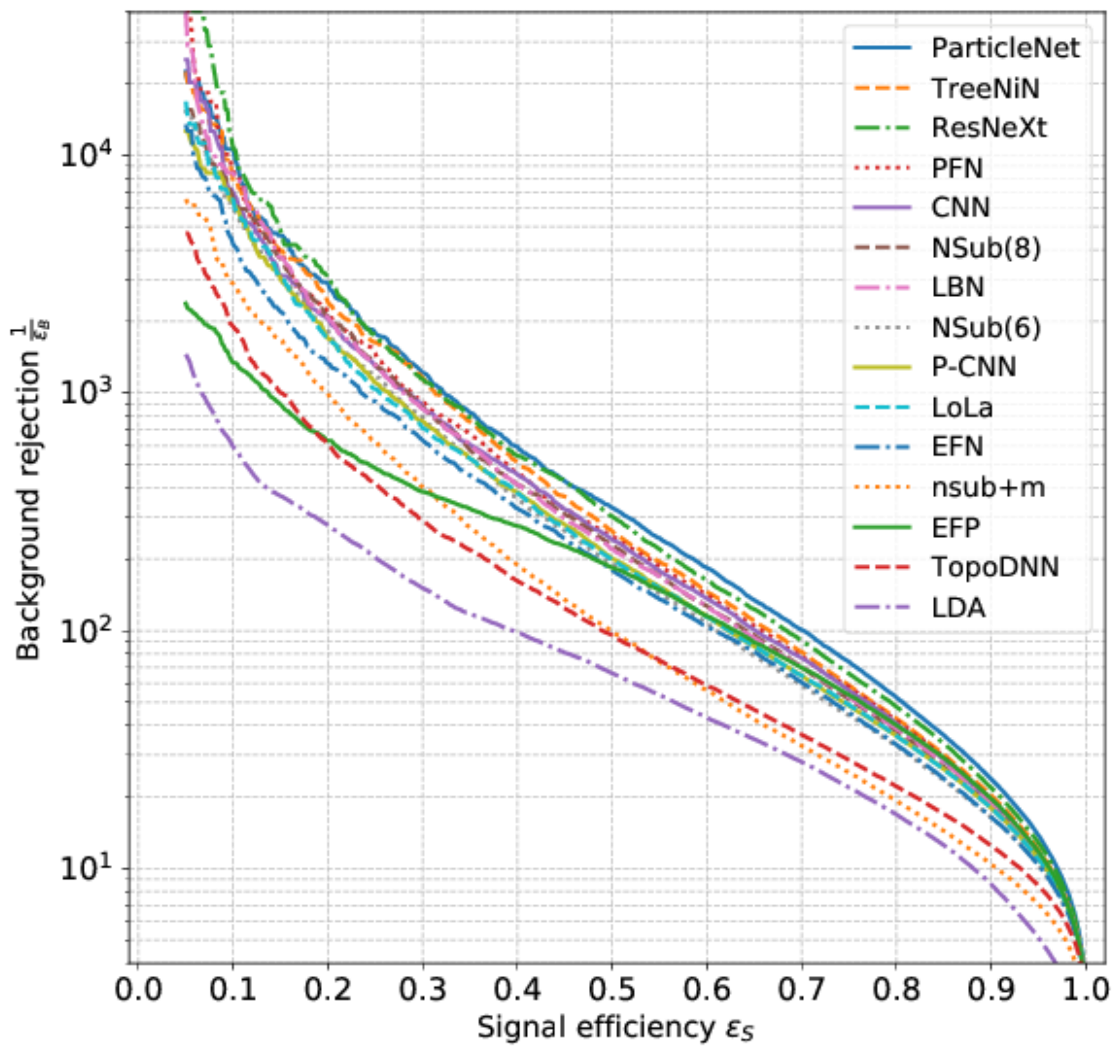
Equivalent to poor efficiency in

$$p(data\,|\,\theta,\nu) = \prod_{i=1}^{n_{bins}} P\left(n_i\,|\,s_i(\theta,\nu)\cdot\epsilon_i(\nu) + b_i(\nu)\right)\; P\left(\nu\,|\,\text{aux. data}\right)$$

The result is not optimal - but still correct!

ML examples and their uncertainties

Classification

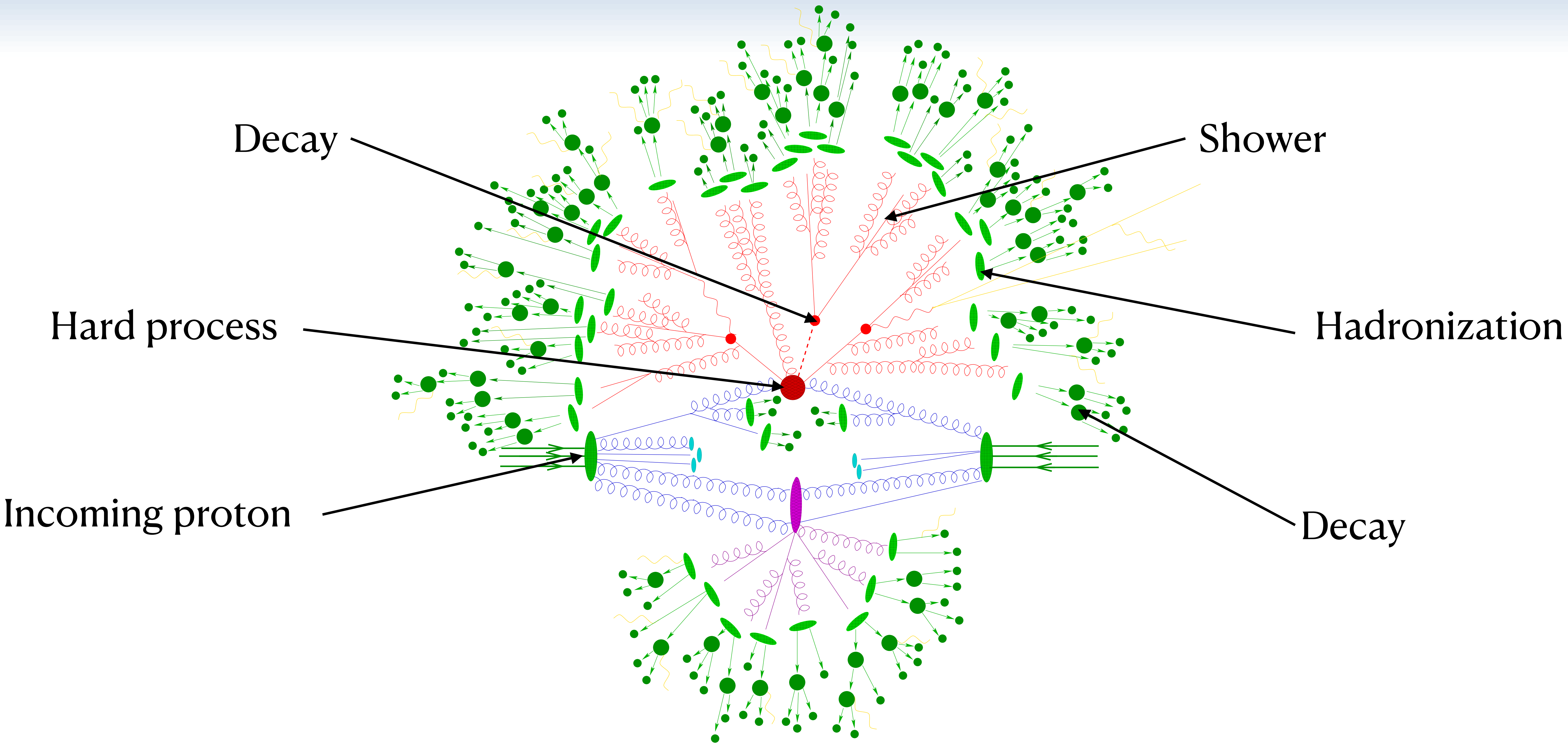


No uncertainty needed

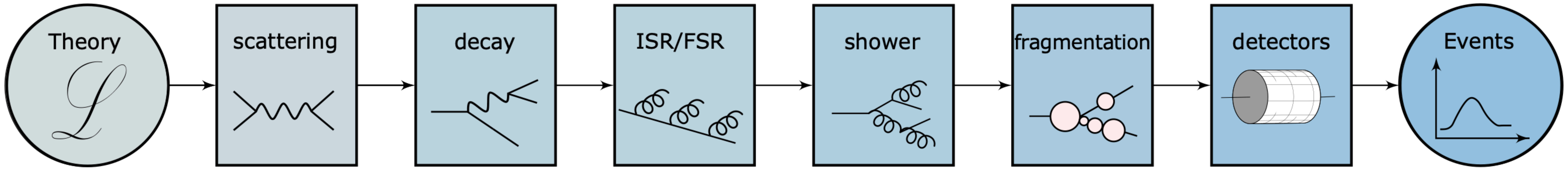
Simulations

Unfolding

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

Monte carlo event generation

1. Generate phase space points

$\rightarrow$ set of four-momenta p_i

2. Calculate event weight

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

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

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

Approximating Amplitudes

- Approximate squared matrix element with NN

- **Regression** problem
- Minimize distance between prediction and truth

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

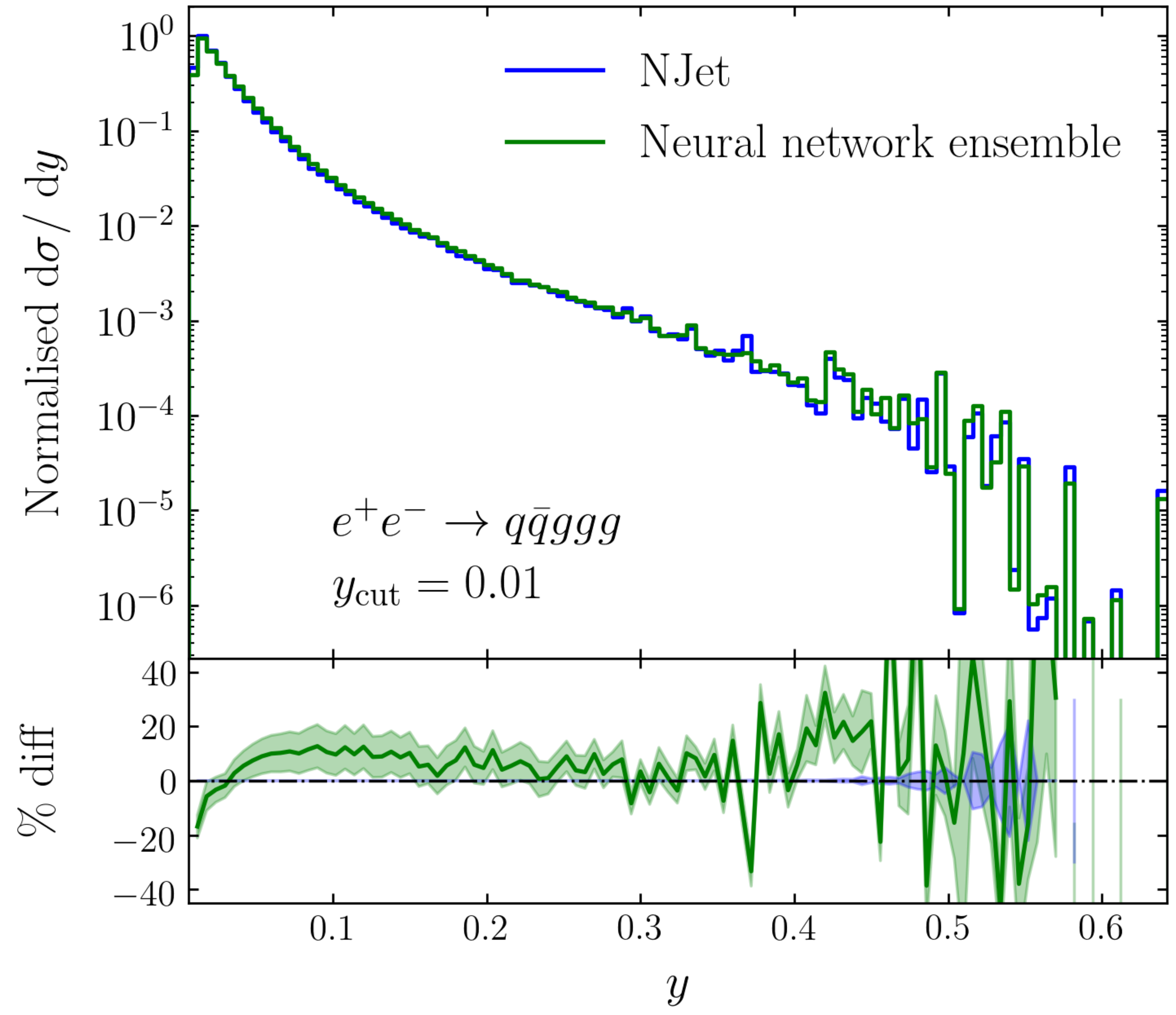
+ Generalization of interpolation
+ Better **scaling** than grids for large dimensions

- Open questions
  - Limited precision ?
  - Overtraining vs interpolation ?

Problem

Wrong estimation leads to wrong prediction!

→ assign uncertainties



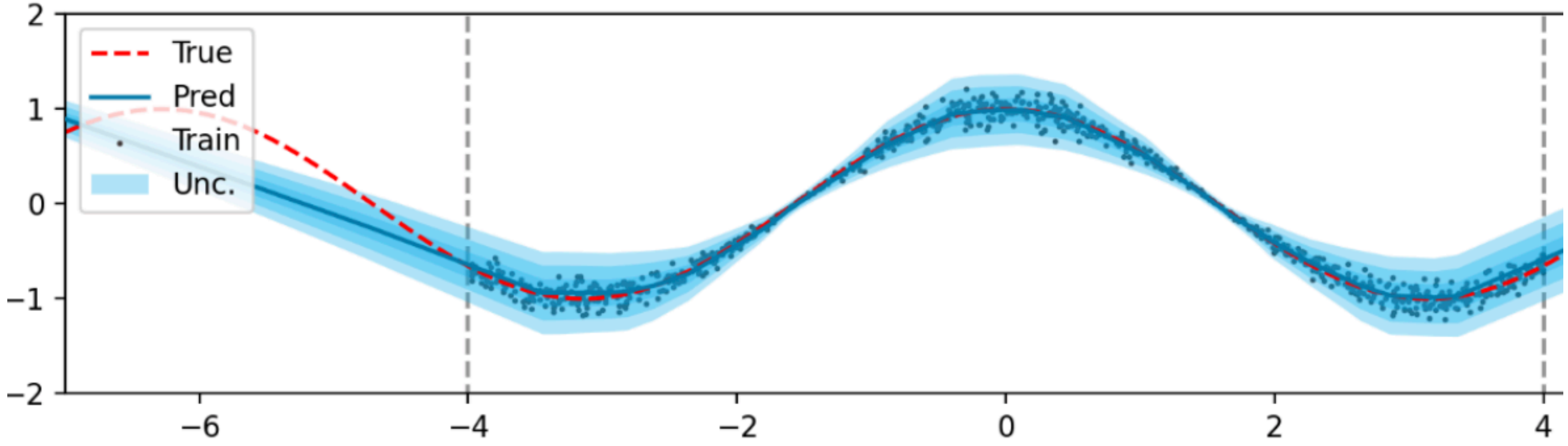
Badger, Bullock [2002.07516]

Estimating uncertainties on amplitude predictions

1. **Extend** standard network **output** to include uncertainty

   $\rightarrow (\mu(x), \sigma(x))$

   - Gaussian approximation
   - $\mathscr{L}_{\text{Gauss}} = -\log(\sqrt{2\pi}\sigma(x)) - \frac{1}{2}\frac{(\mu(x)-y)^2}{\sigma(x)^2}$



   - Captures only $\mathbf{p}(\mathbf{y}\,|\,\mathbf{x}, \mathbf{w})$ for fixed network weights
   - w varies for different trainings!

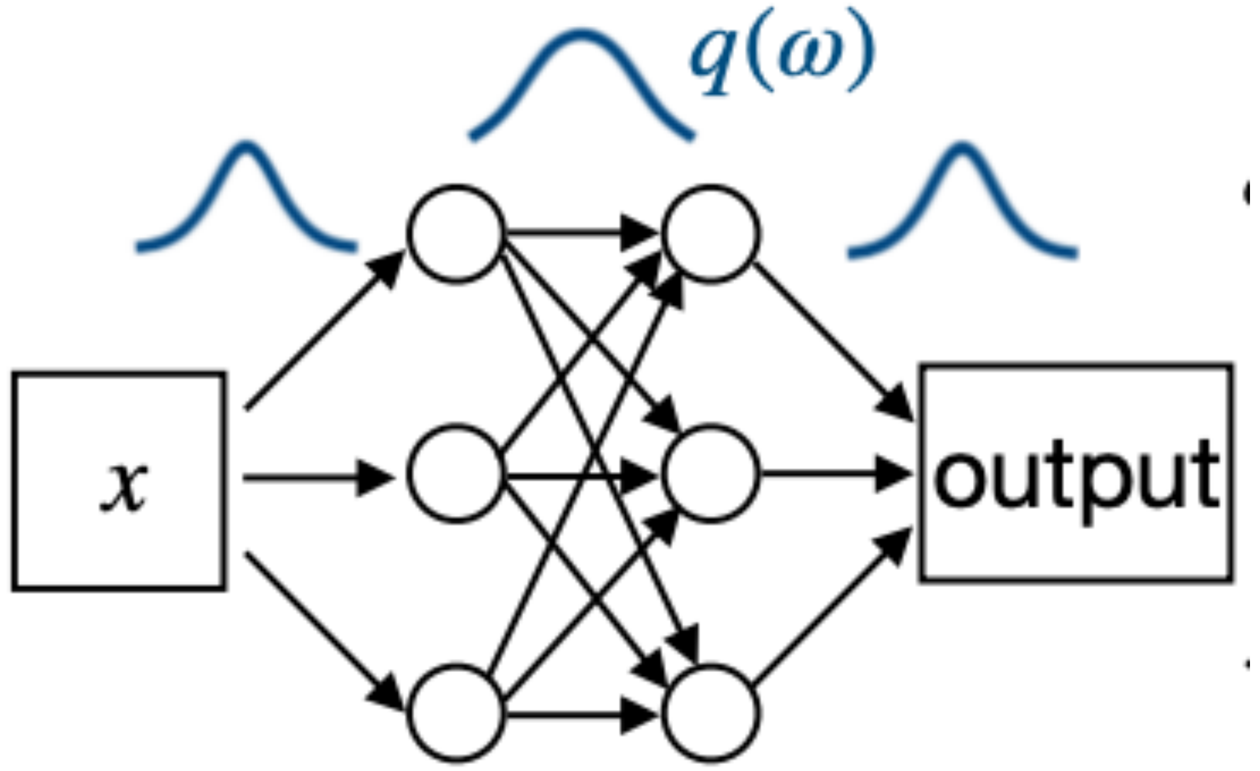
Estimating uncertainties on amplitude predictions

2. Estimating $\mathbf{p}(\mathbf{y}\,|\,\mathbf{x}, \mathbf{D})$ with training dataset D

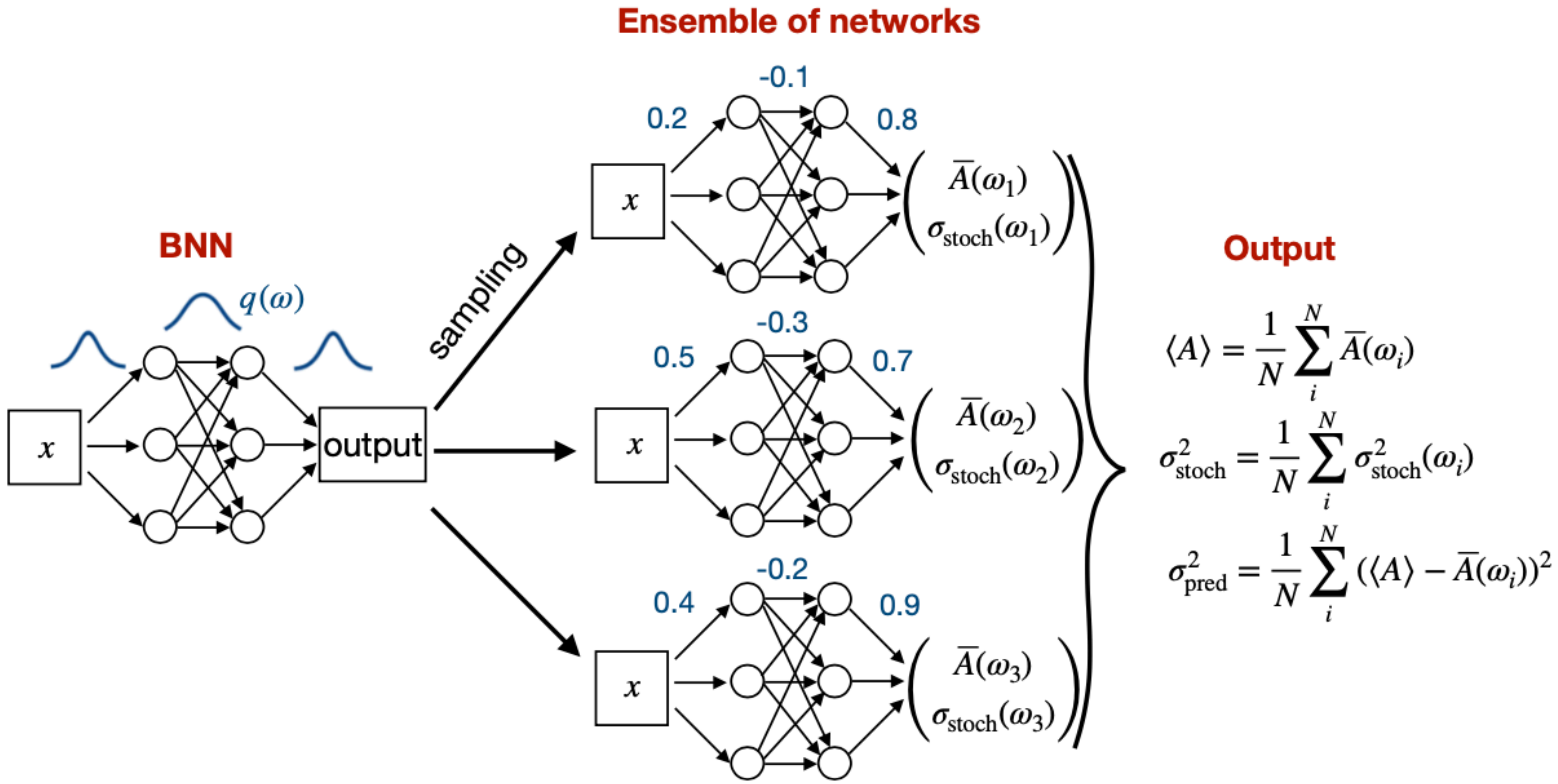
- $p(y\,|\,x, D) = \int \mathrm{d}w \; p(y\,|\,x, w) \; p(w\,|\,D)$

$\mathscr{L}_{\text{Gauss}}$

BNN



Bayesian Neural Network



$$\mathscr{L}_{\text{BNN}} = \int d\omega\; q(\omega) \sum_{\text{points } j} \left[\left| \bar{A}_j(\omega) - A_j^{(\text{truth})} \right|^2 / 2\sigma_{\text{stoch},j}(\omega)^2 + \log \sigma_{\text{stoch},j}(\omega) \right] + \text{KL}[q(\omega), p(\omega)]$$

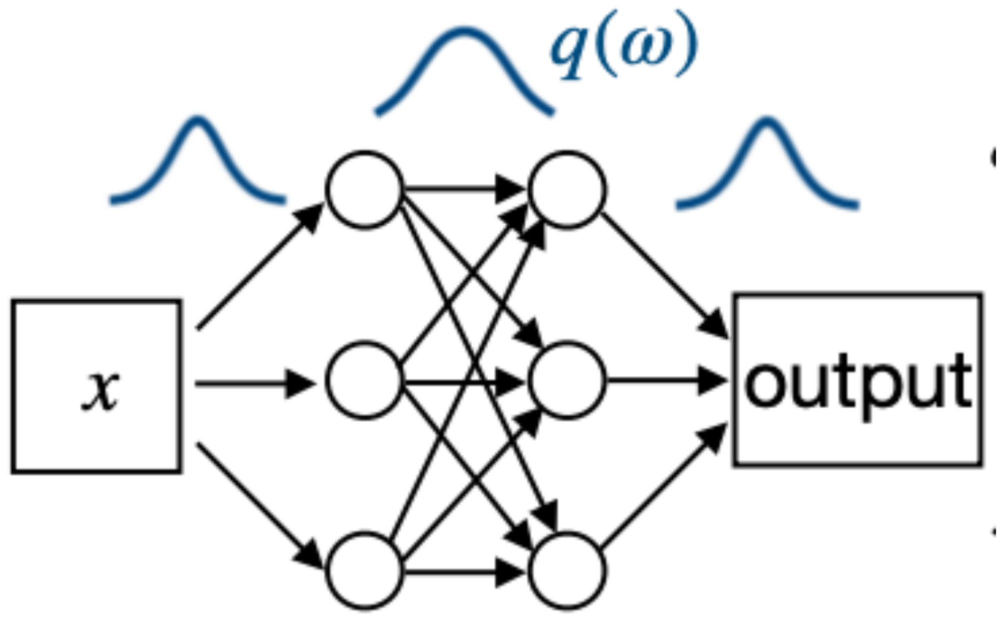
Results BNN

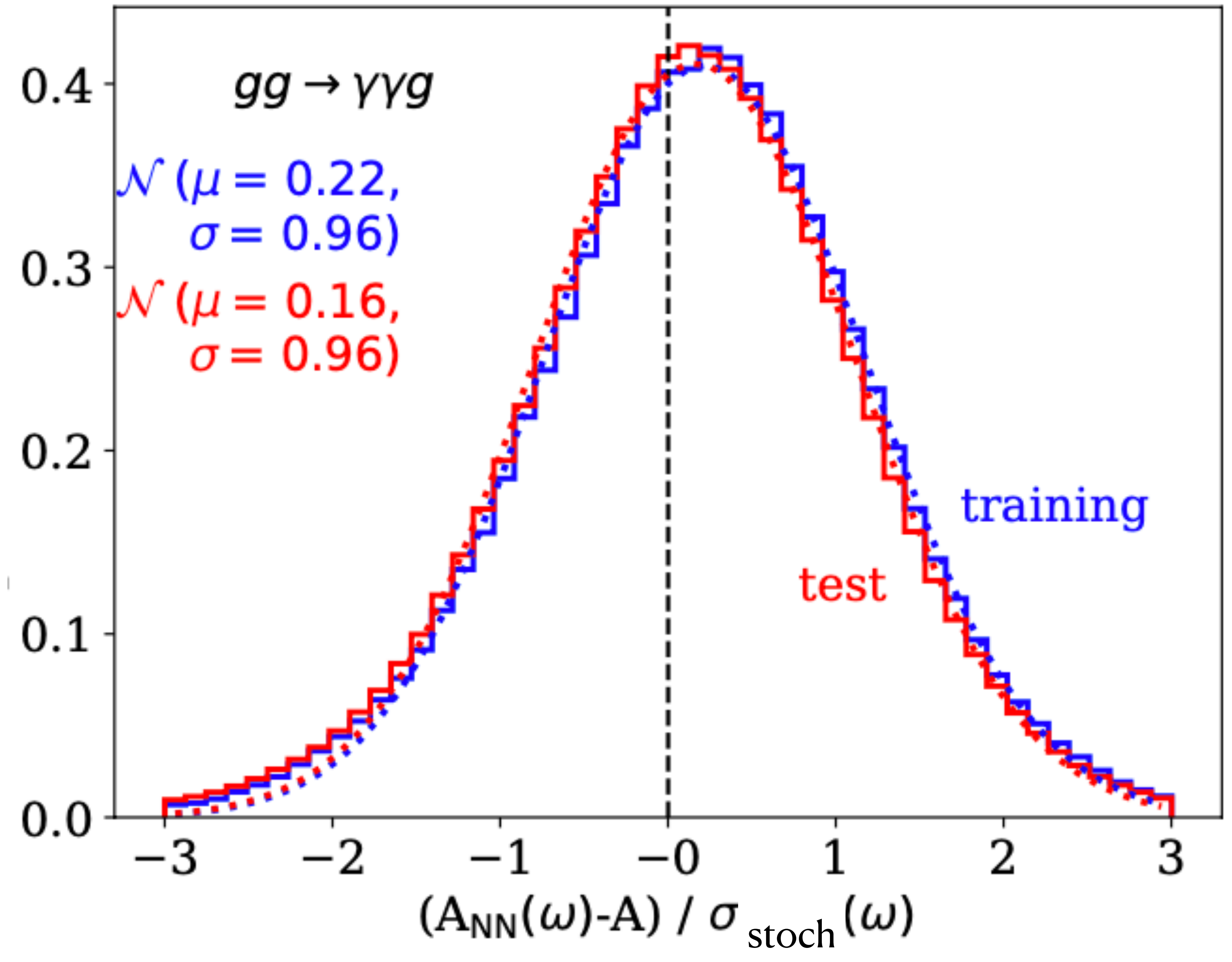
2. Estimating $\mathbf{p}(\mathbf{y}\,|\,\mathbf{x},\mathbf{D})$ with training dataset D

- $p(y\,|\,x, D) = \int \mathrm{d}w \; p(y\,|\,x, w)\; p(w\,|\,D)$

$\mathscr{L}_{\text{Gauss}}$

BNN



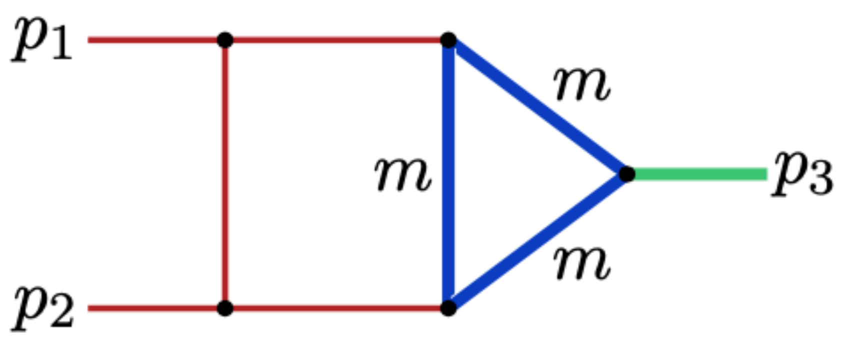


[preliminary]

$$\mathscr{L}_{\text{BNN}} = \int d\omega \; q(\omega) \sum_{\text{points } j} \left[\frac{\left| \bar{A}_j(\omega) - A_j^{(\text{truth})} \right|^2}{2\sigma_{\text{stoch},j}(\omega)^2} + \log \sigma_{\text{stoch},j}(\omega) \right] + \mathrm{KL}[q(\omega), p(\omega)]$$

Multi-loop calculations with NNs

Precision predictions based on loop diagrams



Analytic expression for loop amplitude

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

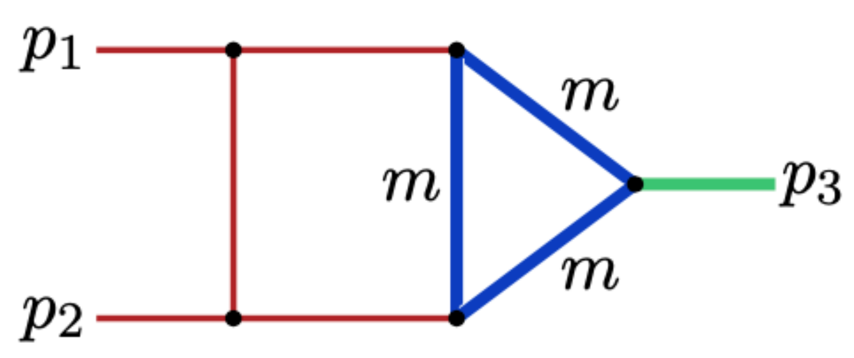
$$= \int_0^1 \prod_{j=1}^{N-1} \mathrm{d}x_j \, x_j^{\nu_j - 1} \frac{U^{\nu-(L+1)D/2}}{F^{\nu - LD/2}} = \int_0^1 \prod_{j=1}^{N-1} \mathrm{d}x_j \; I(\overrightarrow{x})$$

Rewrite with
Feynman parameters

Still contains singularities

Multi-loop calculations with NNs

Precision predictions based on loop diagrams



Analytic expression for loop amplitude

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

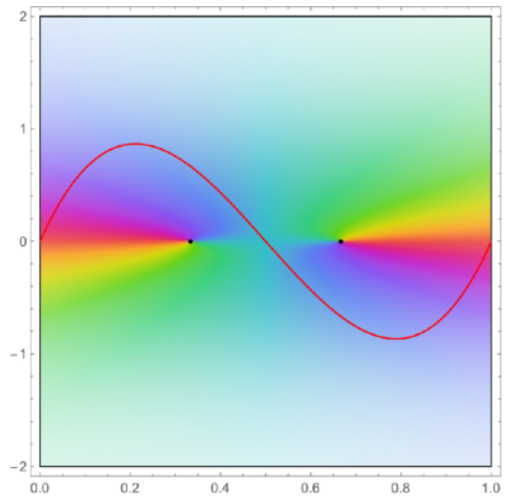
$$= \int_0^1 \prod_{j=1}^{N-1} \mathrm{d}x_j \, x_j^{\nu_j - 1} \frac{U^{\nu-(L+1)D/2}}{F^{\nu-LD/2}} = \int_0^1 \prod_{j=1}^{N-1} \mathrm{d}x_j \, I(\overrightarrow{x})$$

Rewrite with Feynman parameters

Still contains singularities

Solved by contour deformation due to Cauchy's theorem

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



Optimal parametrization = minimal variance

Turn it into an ML Problem

Integration with normalizing flows

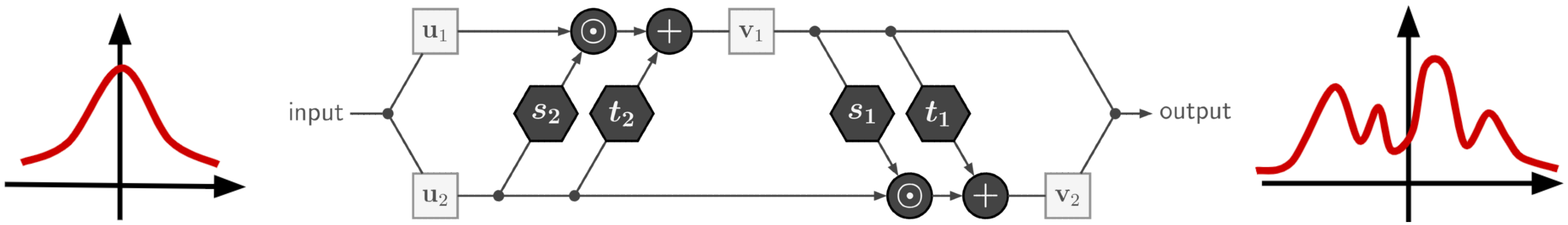
Numeric evaluation of integral $G = \int_0^1 \mathrm{d}x_j \, \det\left(\frac{\partial \vec{z}(\vec{x})}{\partial \vec{x}}\right) I(\vec{z}(\vec{x}))$

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

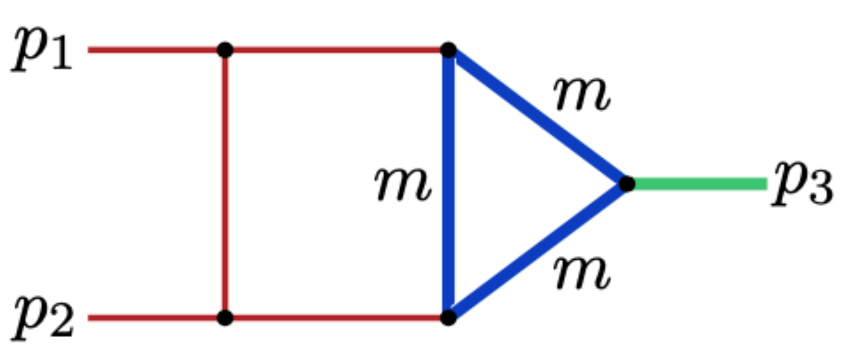
+ Bijective mapping
+ Tractable Jacobian
+ Combine many blocks



Multi-loop calculations with INNs

Profiting from the Jacobian

Precision predictions based on loop diagrams



Analytic expression for loop amplitude

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

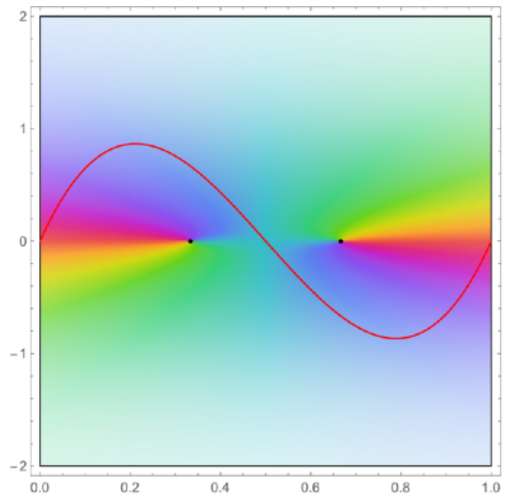
$$= \int_0^1 \prod_{j=1}^{N-1} \mathrm{d}x_j \, x_j^{\nu_j - 1} \frac{U^{\nu-(L+1)D/2}}{F^{\nu-LD/2}} = \int_0^1 \prod_{j=1}^{N-1} \mathrm{d}x_j \; I(\vec{x})$$

Rewrite with
Feynman parameters

Still contains singularities

Solved by contour deformation due to Cauchy's theorem

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



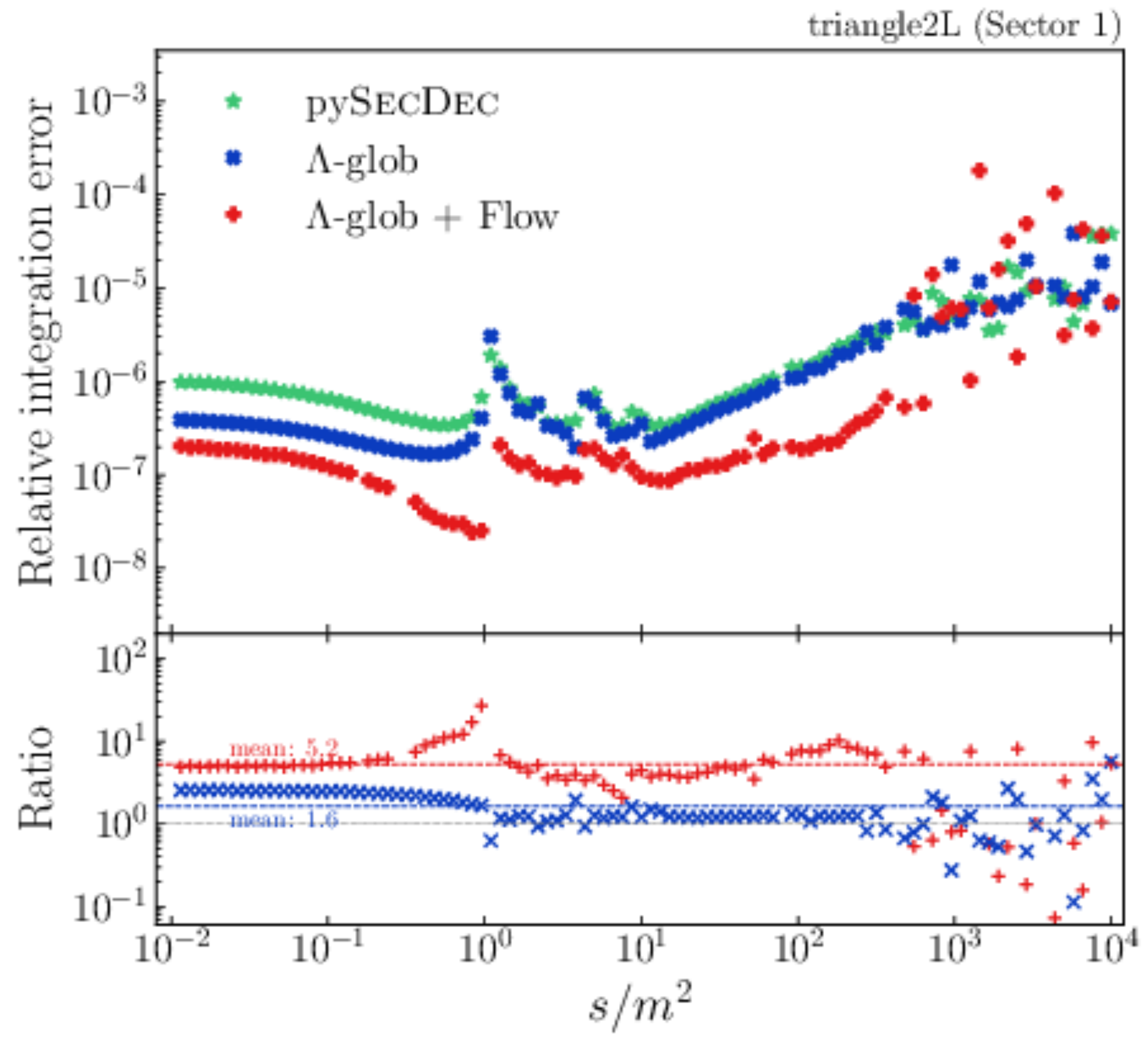
Optimal parametrization = minimal variance

Turn it into an ML Problem

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

Variance $\rightarrow \mathscr{L}$

Better network $\rightarrow$ smaller variance



R. Winterhalder, et al. [2112.09145]

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

† 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 $w_i \ll w_{\text{max}}$
   - Optimize phase space mapping

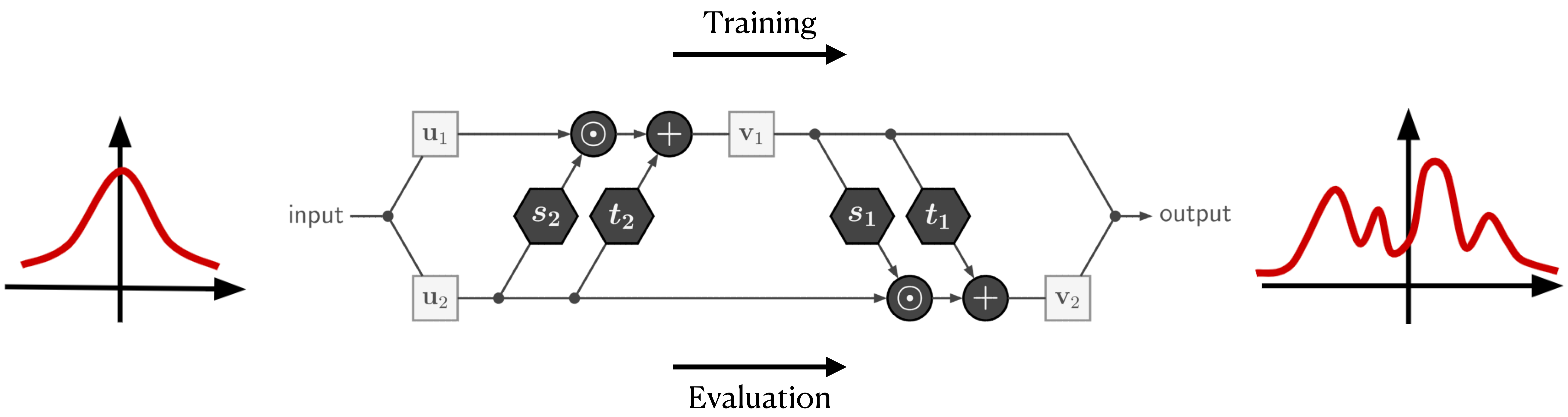
     ➡ $J(p_i(r)) = (f \times \mathcal{M})^{-1}$

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

Normalizing flows

Invertible networks for complex transformations

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



Training on density $t(x)$
$\rightarrow$ Minimize difference

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

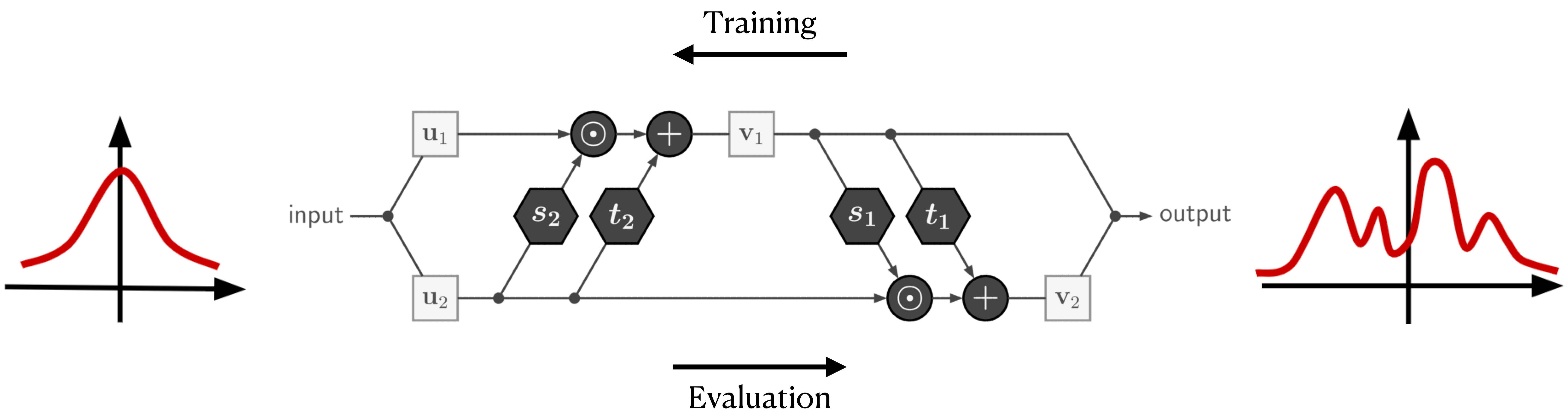
Training on samples x
$\rightarrow$ Maximize the log-likelihood

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

Normalizing flows

Invertible networks for complex transformations

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



Training on density $t(x)$
$\rightarrow$ Minimize difference

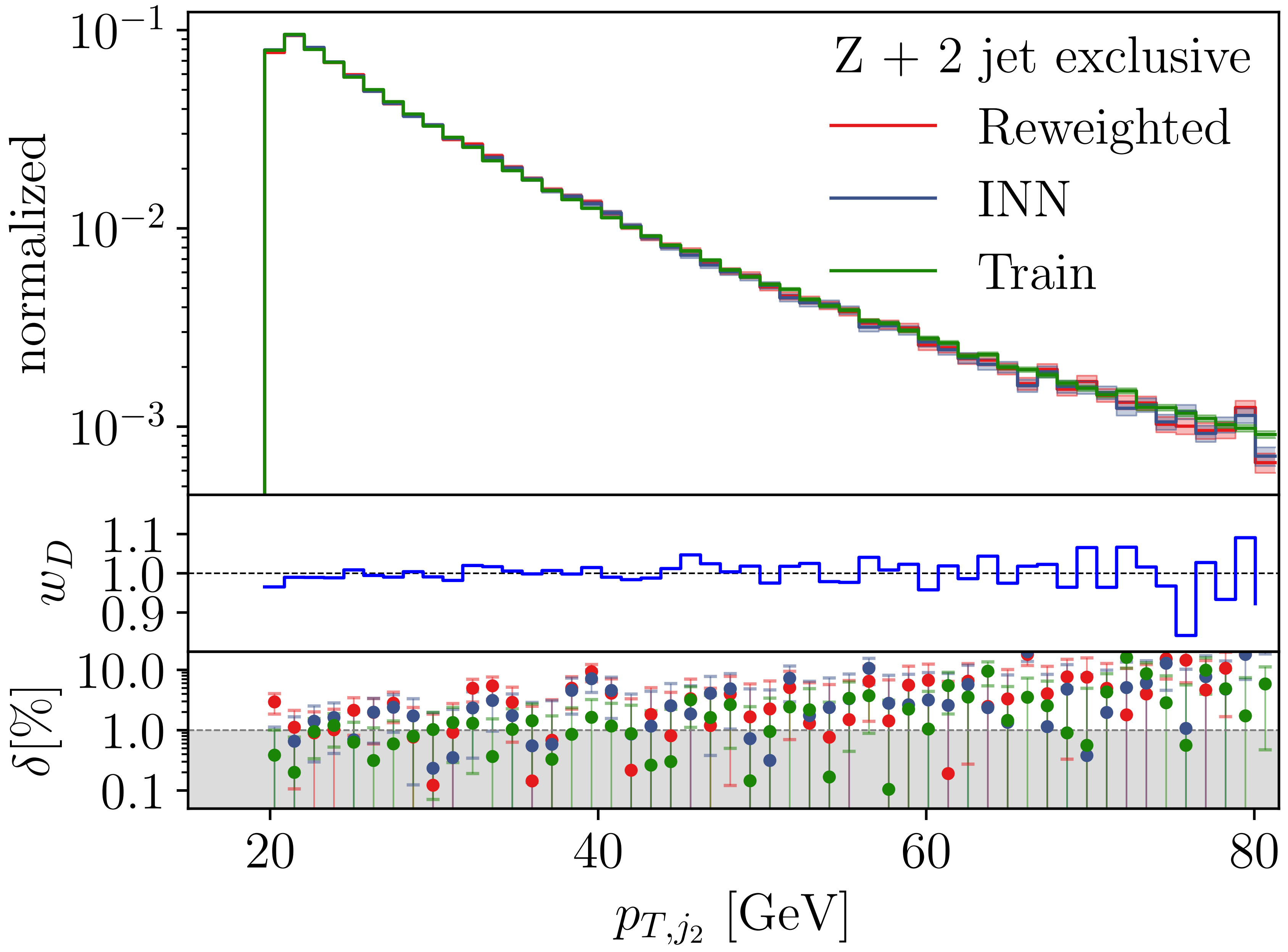
$$\begin{aligned}\mathscr{L} &= \log p_x(x)/t(x) \\ &= \log p_z(z(x))\, J_{NN} \,/\, t(x)\end{aligned}$$

Training on samples x
$\rightarrow$ Maximize the log-likelihood

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

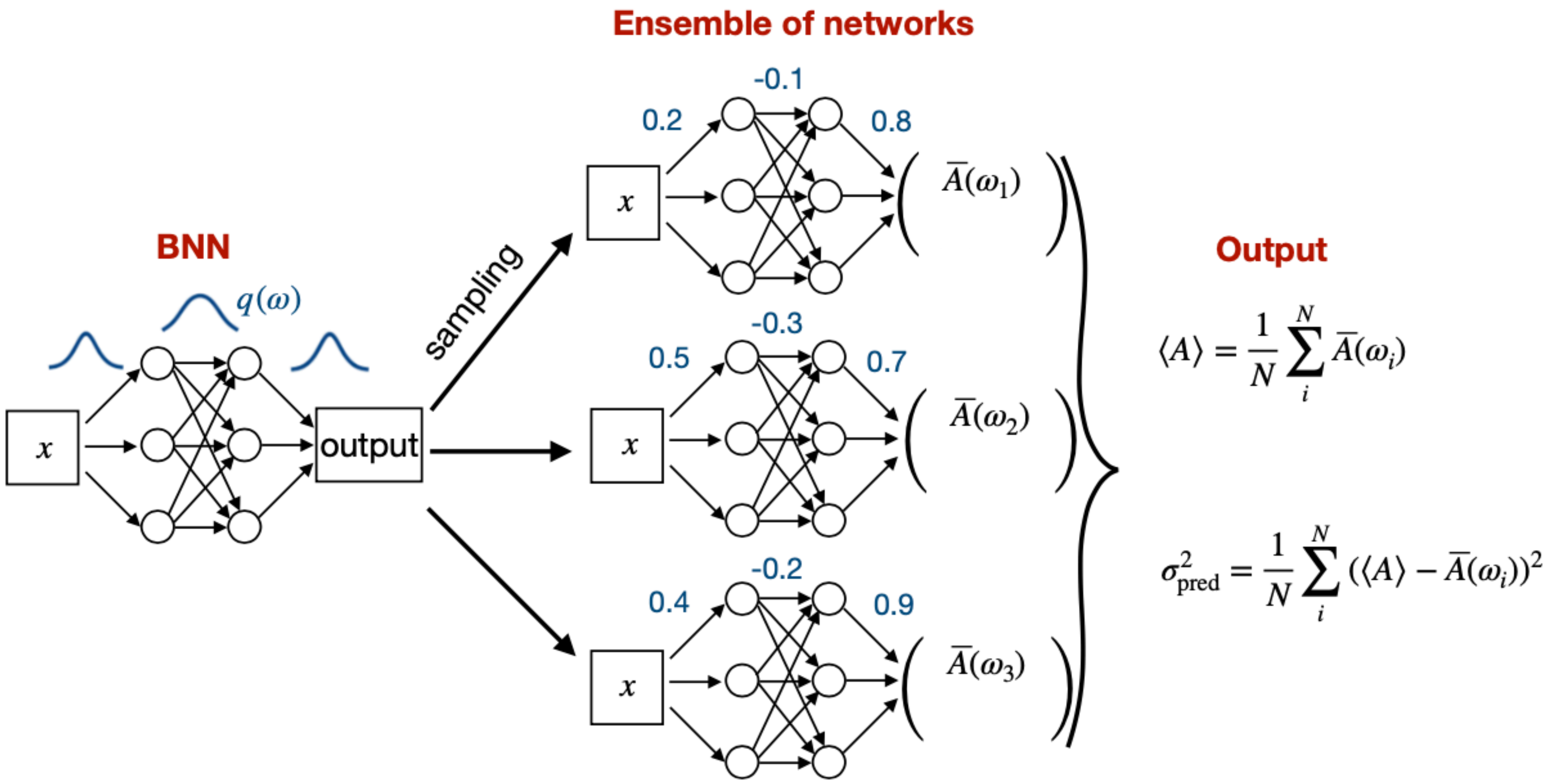
Putting flows to work

Event generation



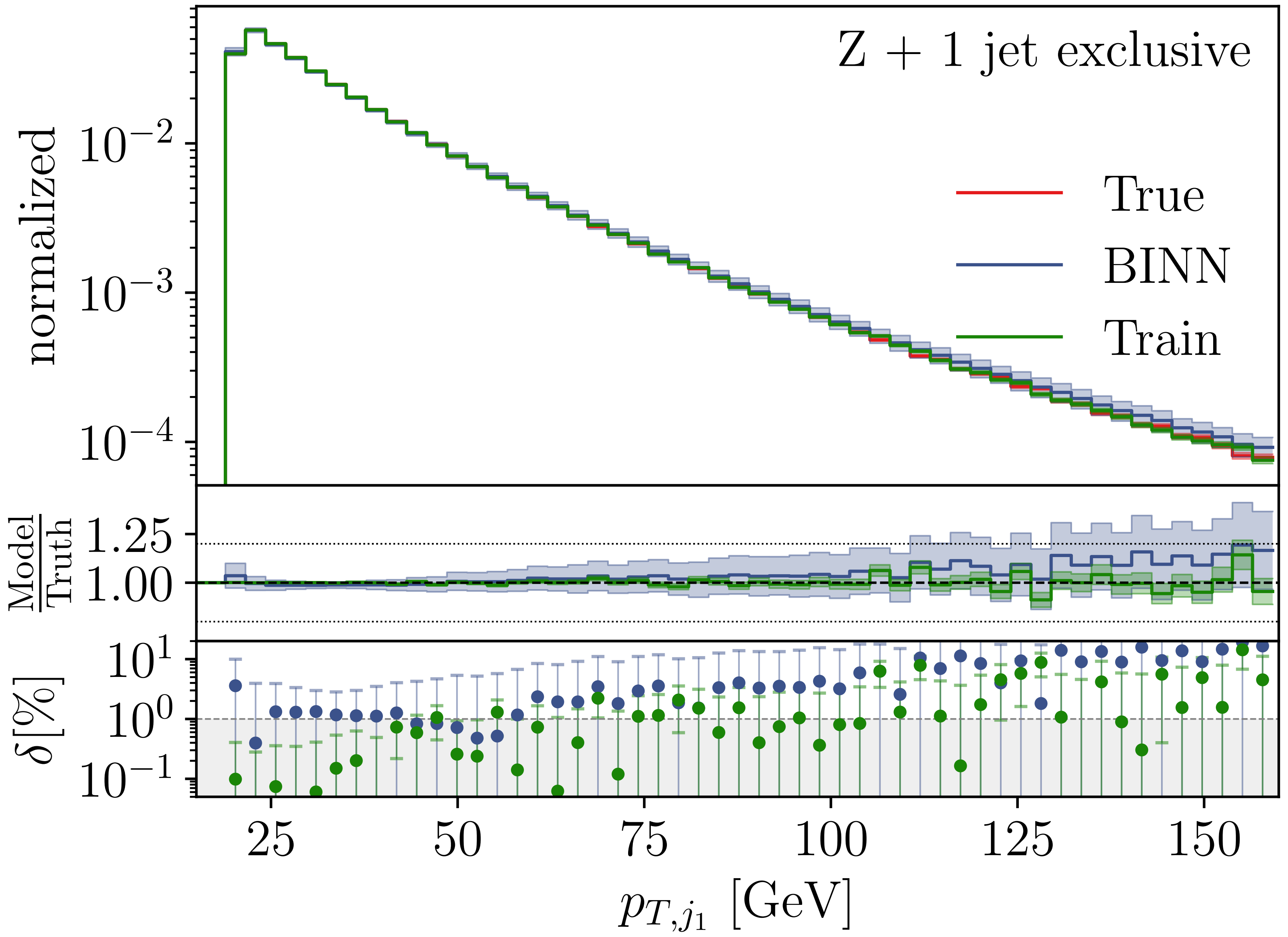
- Train normalizing flow on 4-momenta
- Include symmetries in feature representation
- Excellent performance for direct output

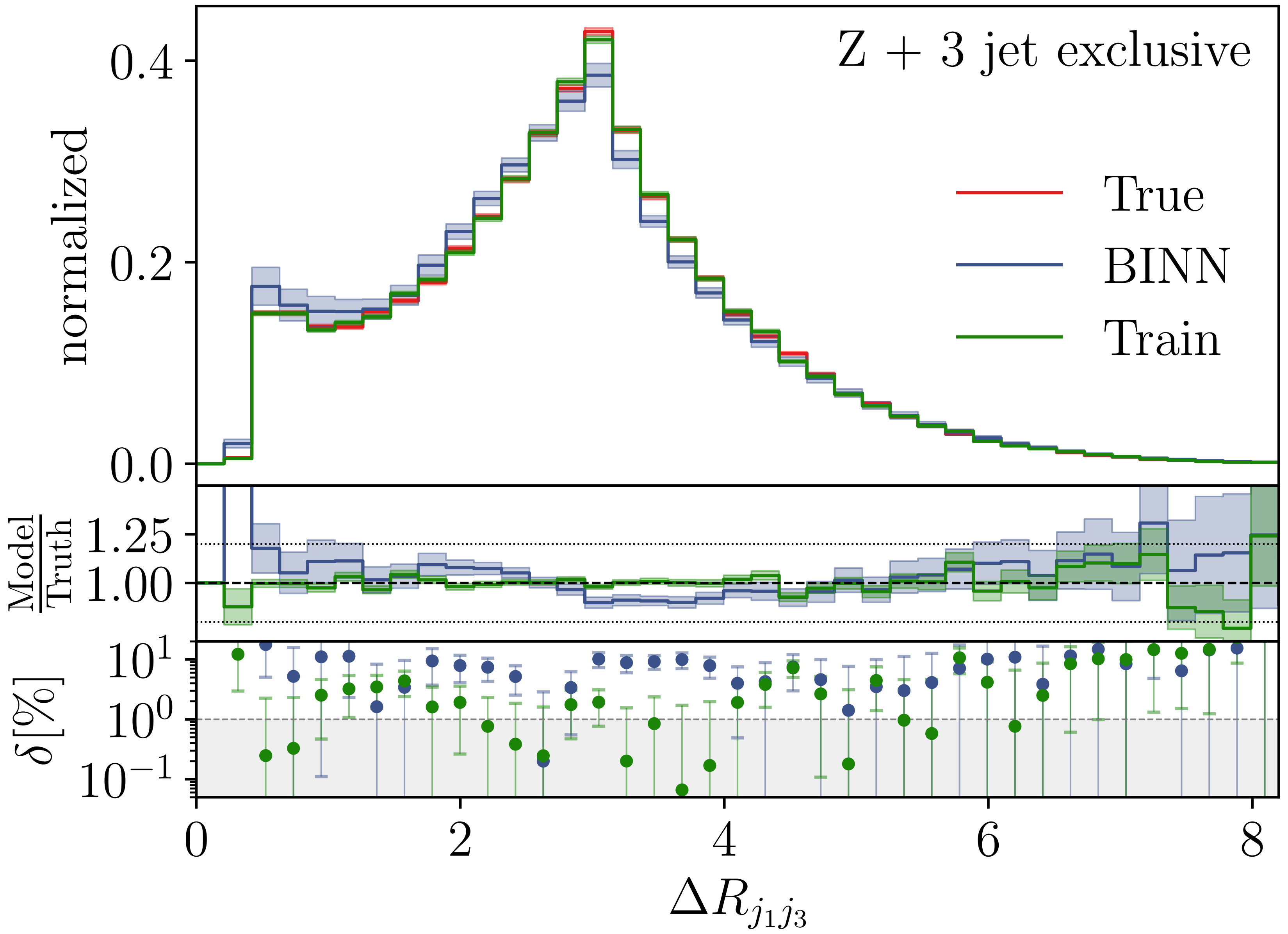
Bayesian Neural Network

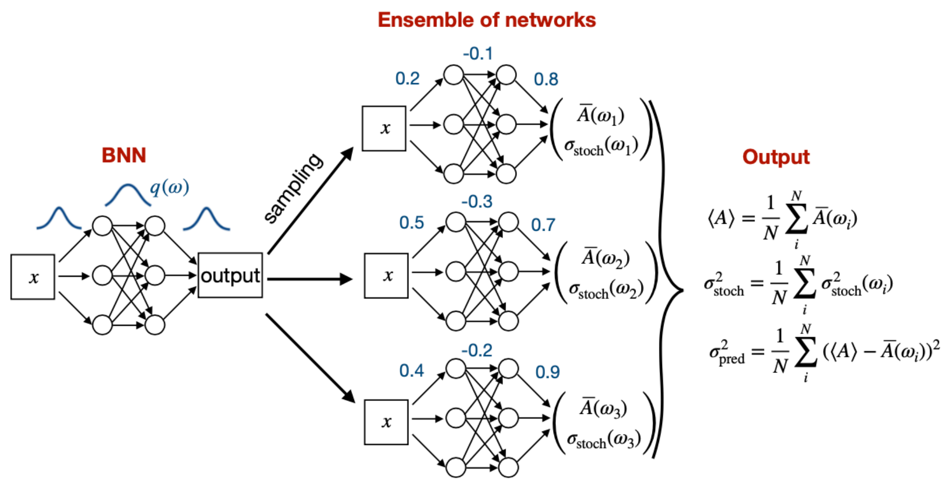


$$
\begin{aligned}
\mathscr{L} &= \mathscr{L}_{INN} + KL_{prior} \\
&= \sum_{n=1}^{N} \langle \log p_X(x_n \,|\, \theta) \rangle_{\theta \sim q_\Phi(\theta)} - KL(q_\Phi(\theta), p(\theta))
\end{aligned}
$$

Bayesian generative networks





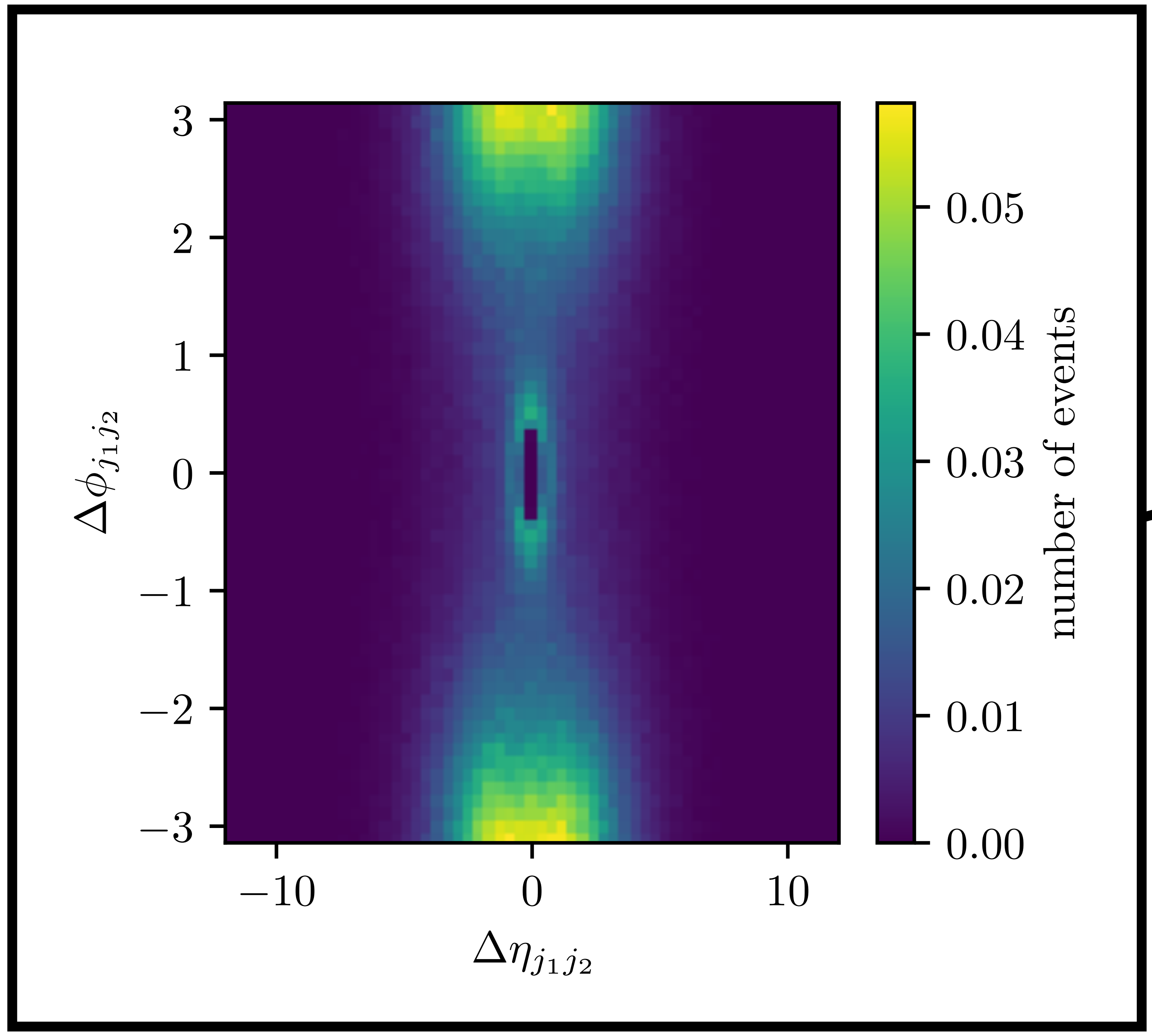


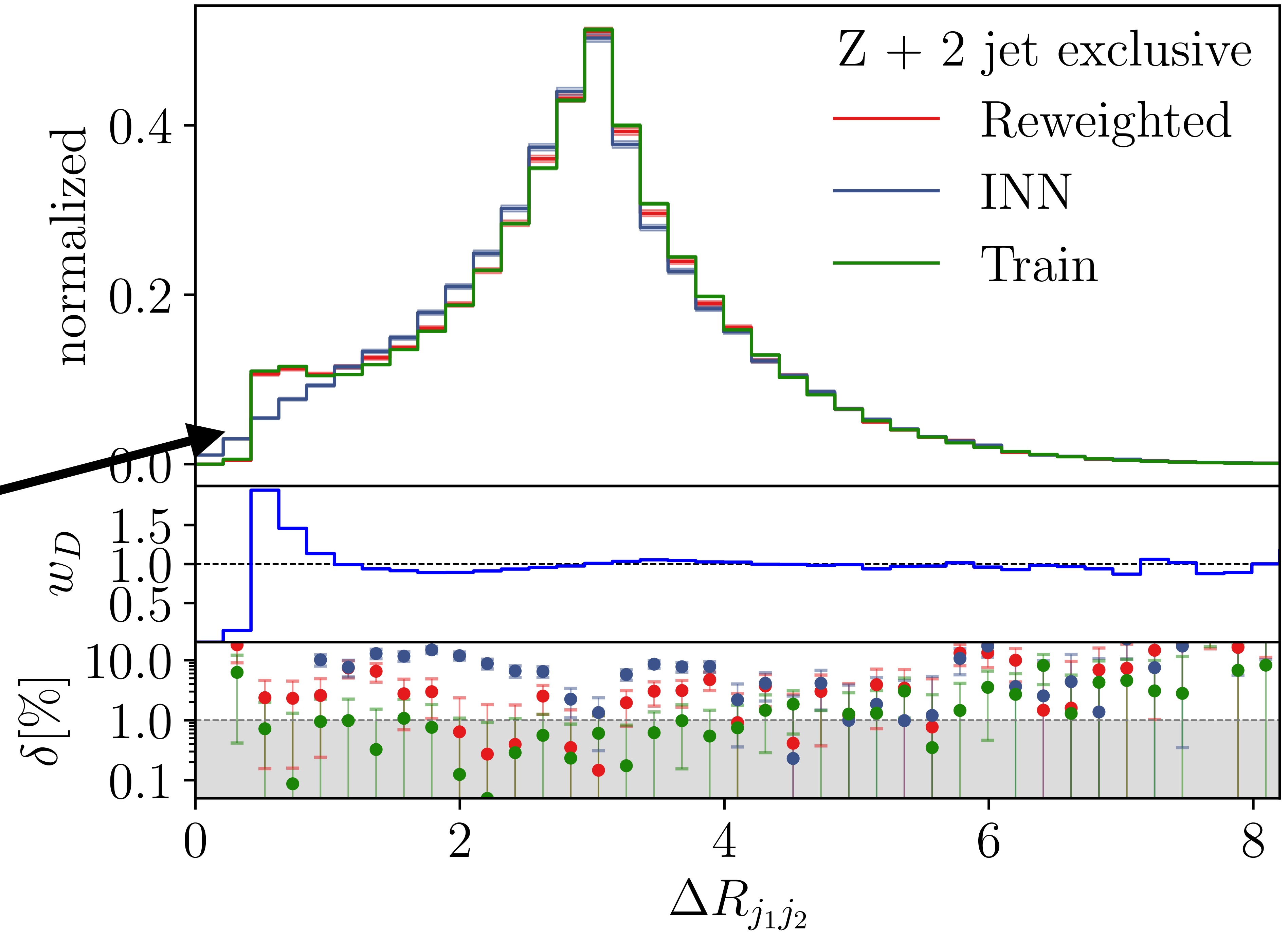
$\Rightarrow$ BINN captures uncertainty related to convergence and statistical uncertainties

$\Rightarrow$ BINN does not capture lack of expressiveness

Challenges for normalizing flows

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





Reweighting for Precision

- Classifier loss

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

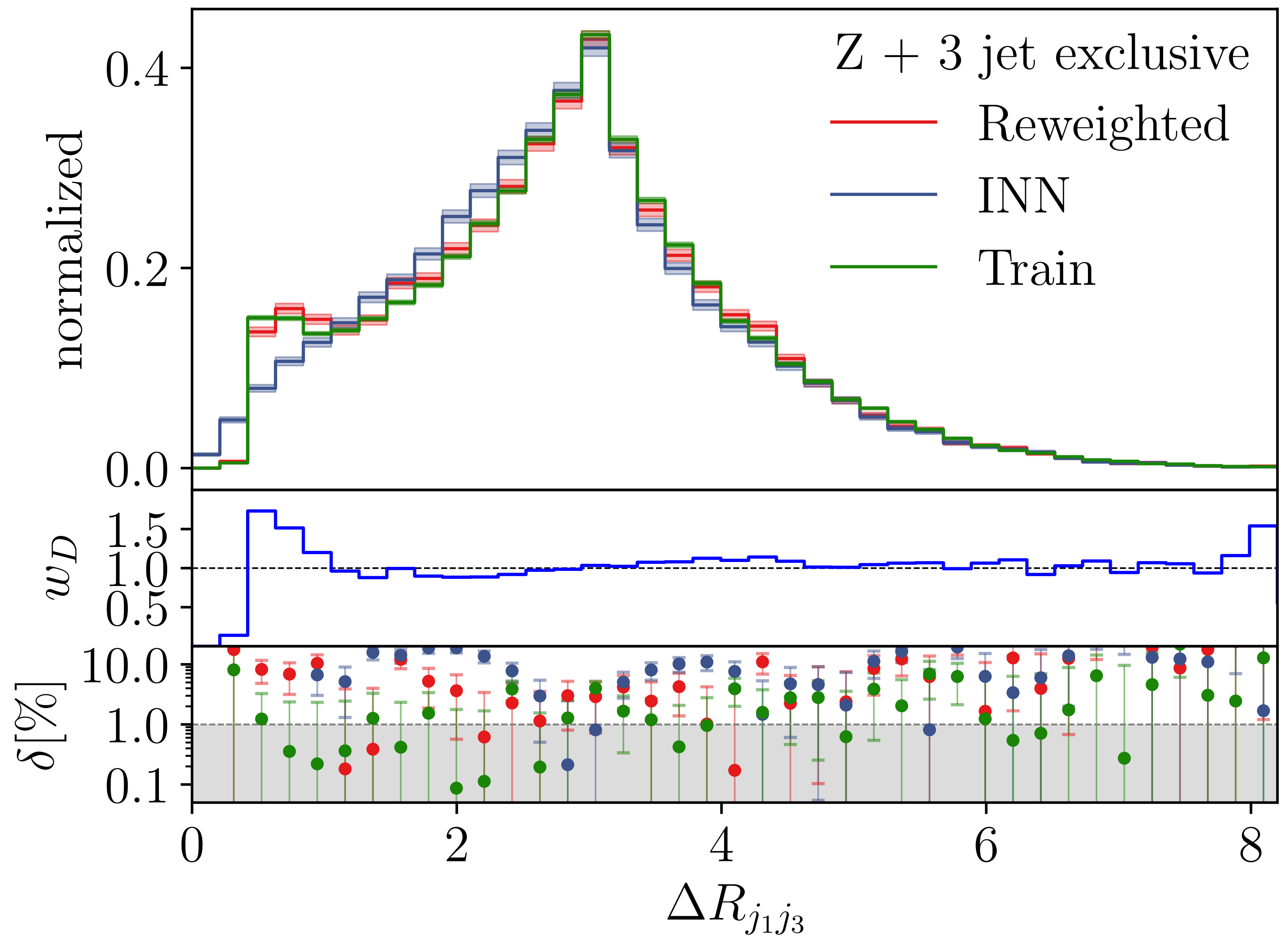
$$= - \int \mathrm{d}x \, p_{data}(x) \, \log(D(x)) + p_{INN}(x) \, \log(1 - D(x))$$

- Upon convergence obtain **reweighting factor**

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

- Improve precision through reweighting

- Quantifies deviation

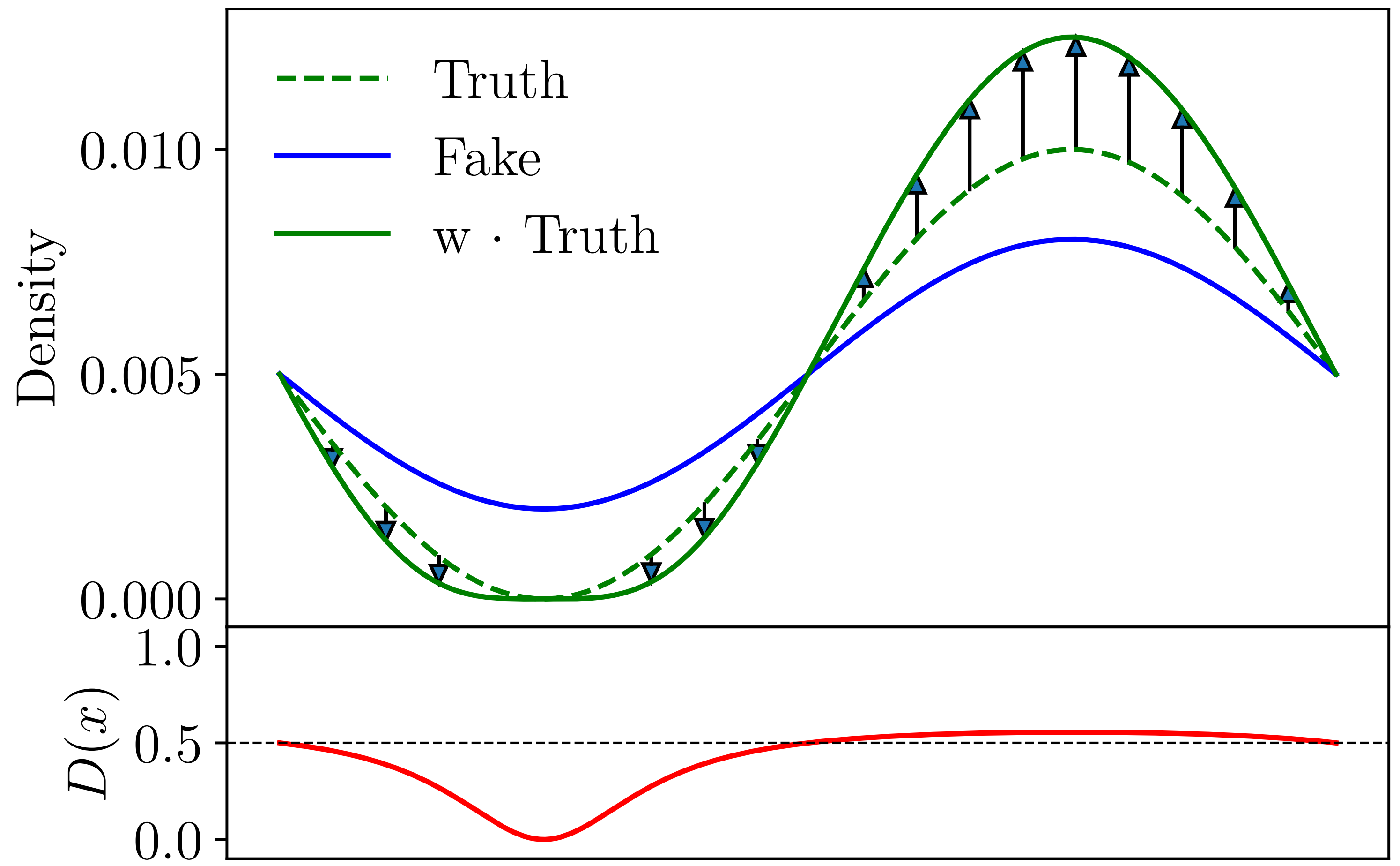


Reweighting for Precision

- 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**

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

- Improve precision through reweighting

- Quantifies deviation

Reweighting for Precision

- Classifier loss

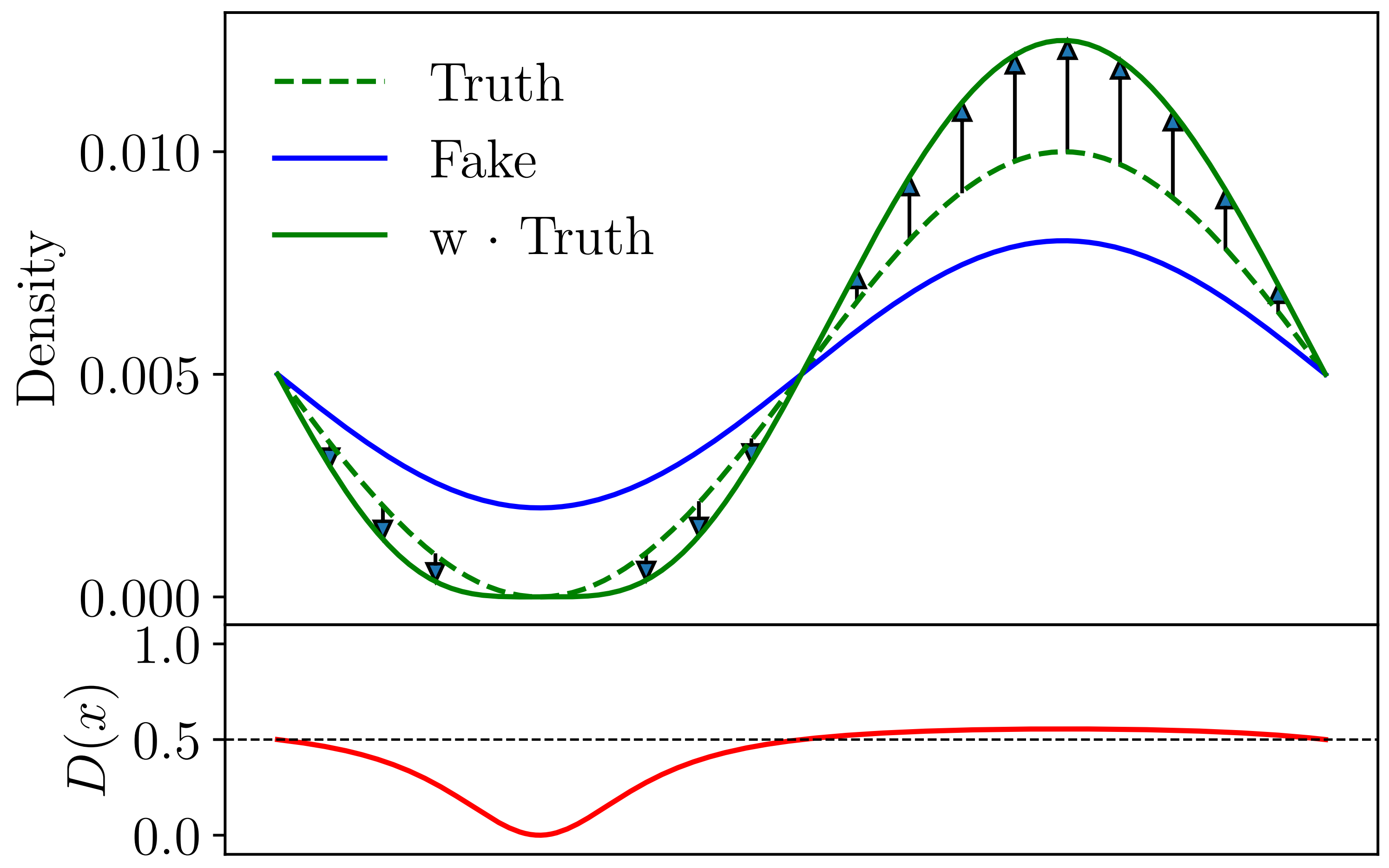
$$\mathscr{L} = -\sum_{x\sim p_{data}} \log(D(x)) - \sum_{x\sim p_{INN}} \log(1 - D(x))$$
$$= -\int \mathrm{d}x\, p_{data}(x)\, \log(D(x)) + p_{INN}(x)\, \log(1 - D(x))$$

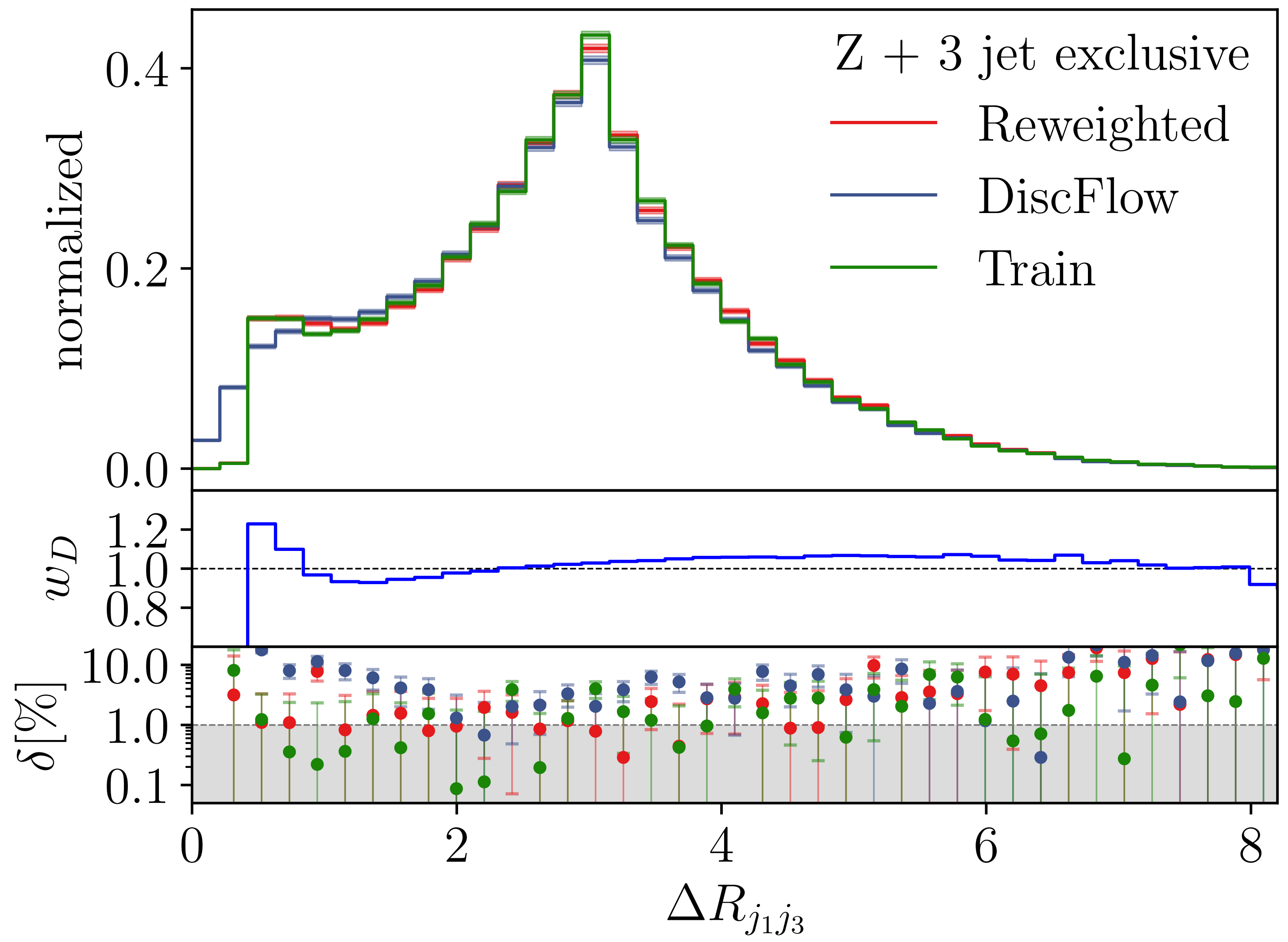
- Upon convergence obtain **reweighting factor**

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

- Improve precision through reweighting

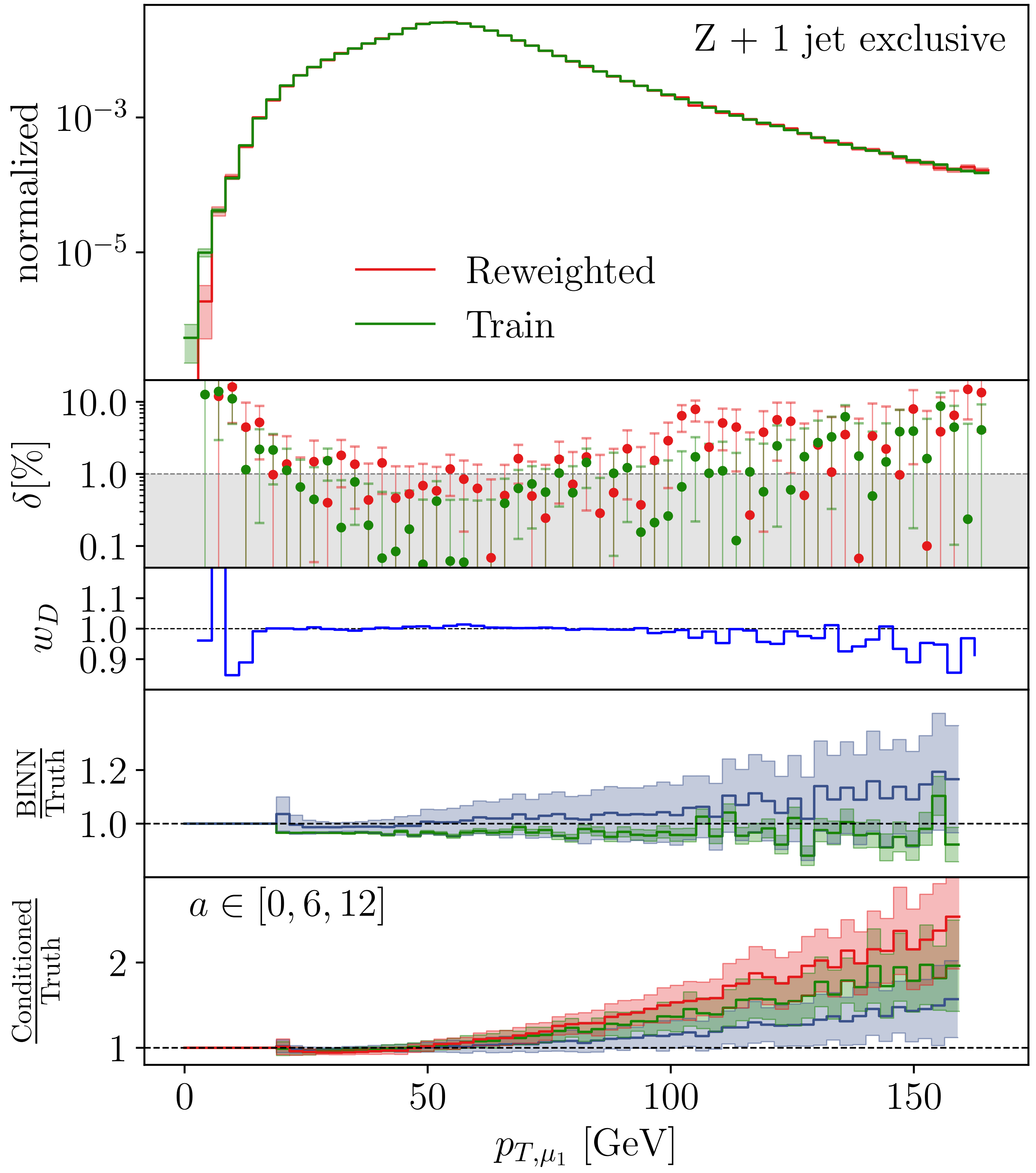
- Quantifies deviation





Putting flows to work

Event generation



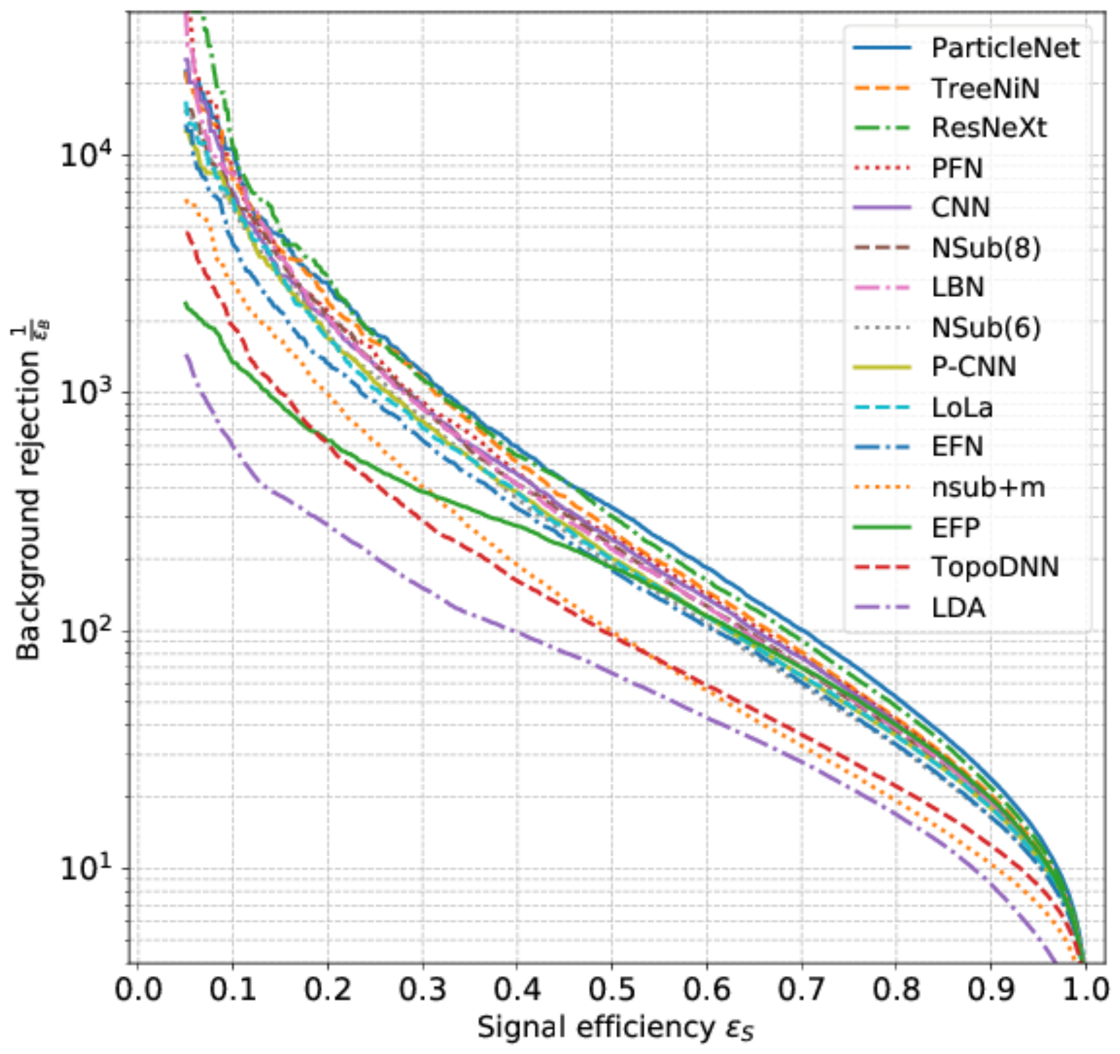
- Basis: INN
  - Phase space symmetries in architecture
- Control via classifier D
  - $\dfrac{p_{\text{truth}}(x)}{p_{\text{INN}}(x)} = \dfrac{D(x)}{1 - D(x)}$
- Precision via reweighting
  - Correct deviations of p_{INN}

➡ Uncertainty estimation via Bayesian NN

➡ Uncertainty propagation via conditioning

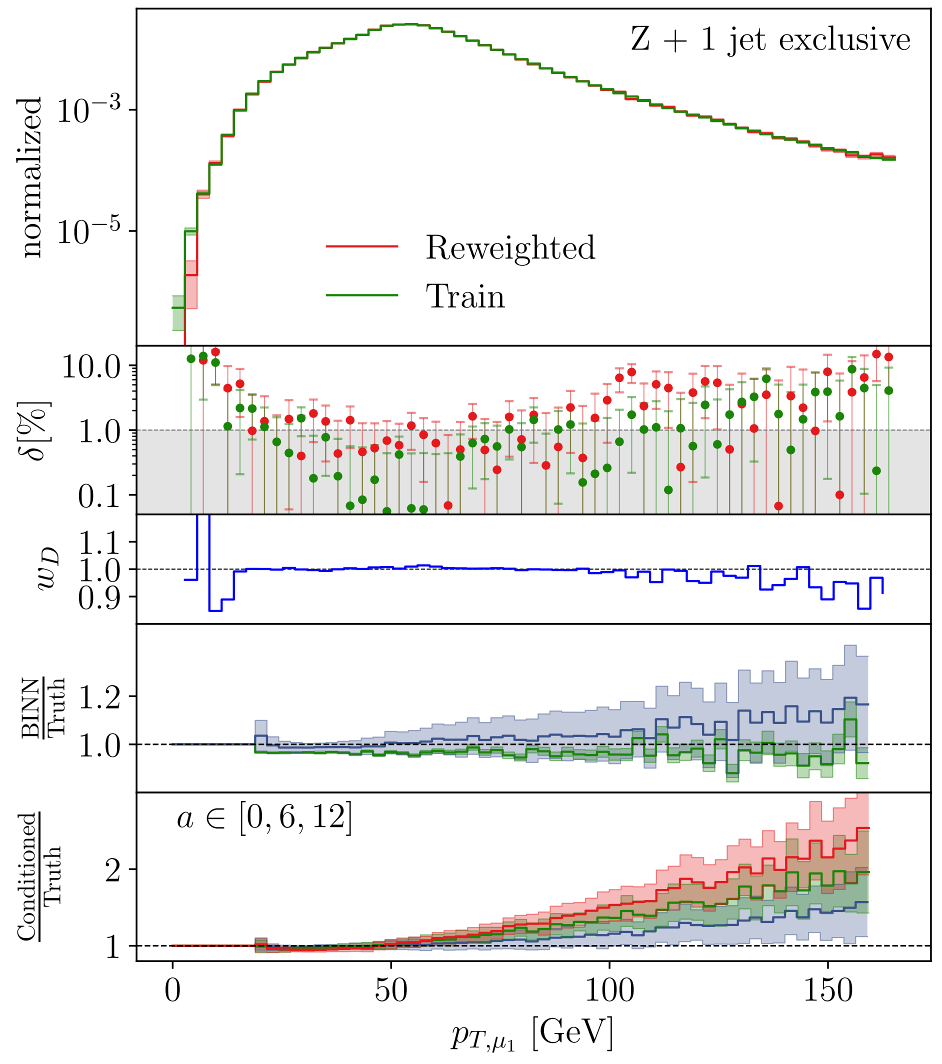
ML examples and their uncertainties

Classification



No uncertainty needed

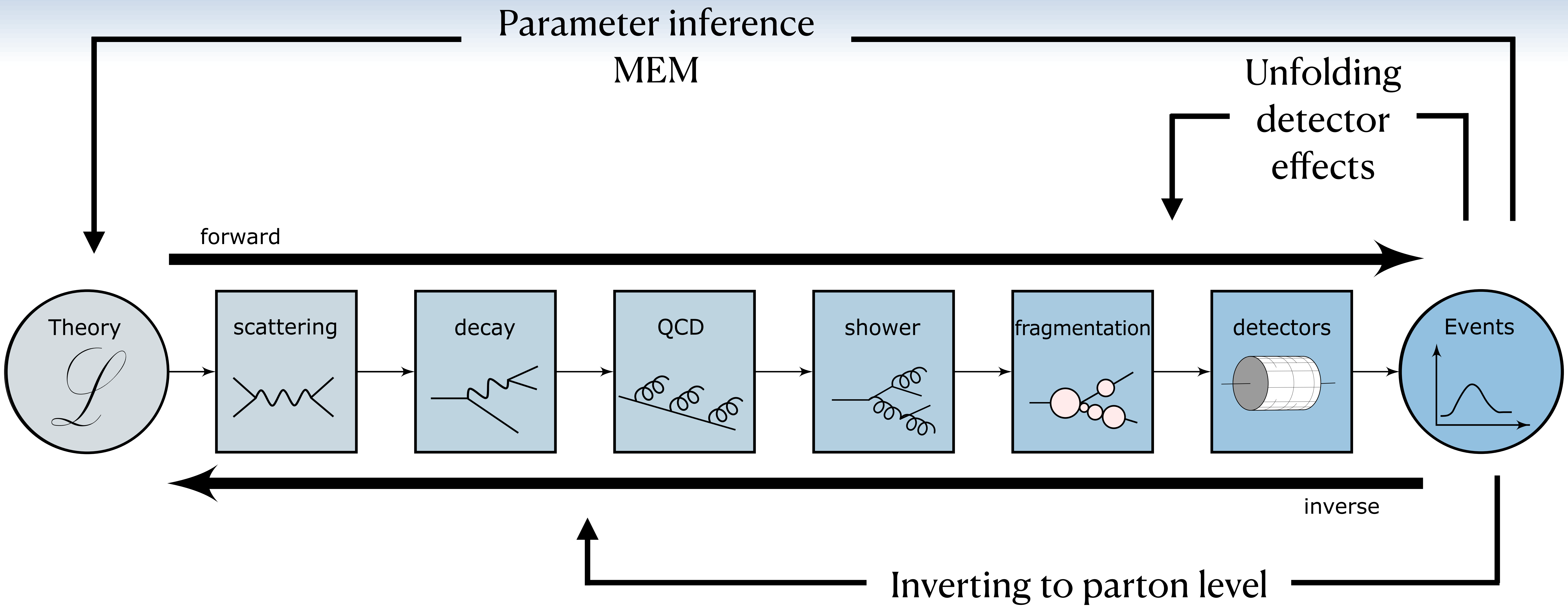
Simulations



Amplitude estimation -> yes (BNN)
Loop integration -> no
Phase space sampling -> no
Data compression -> yes (BNN & classifier)

Unfolding

Inverting the simulation chain

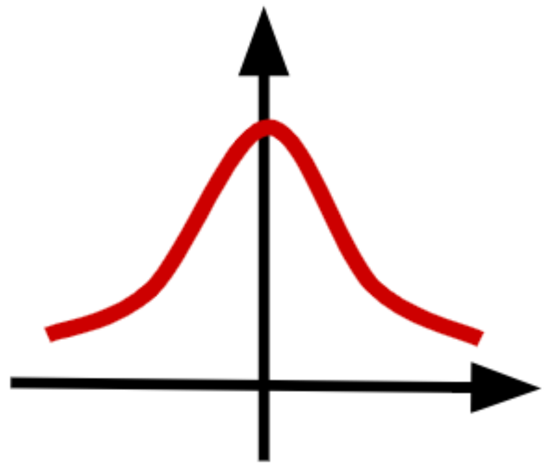


Requirements

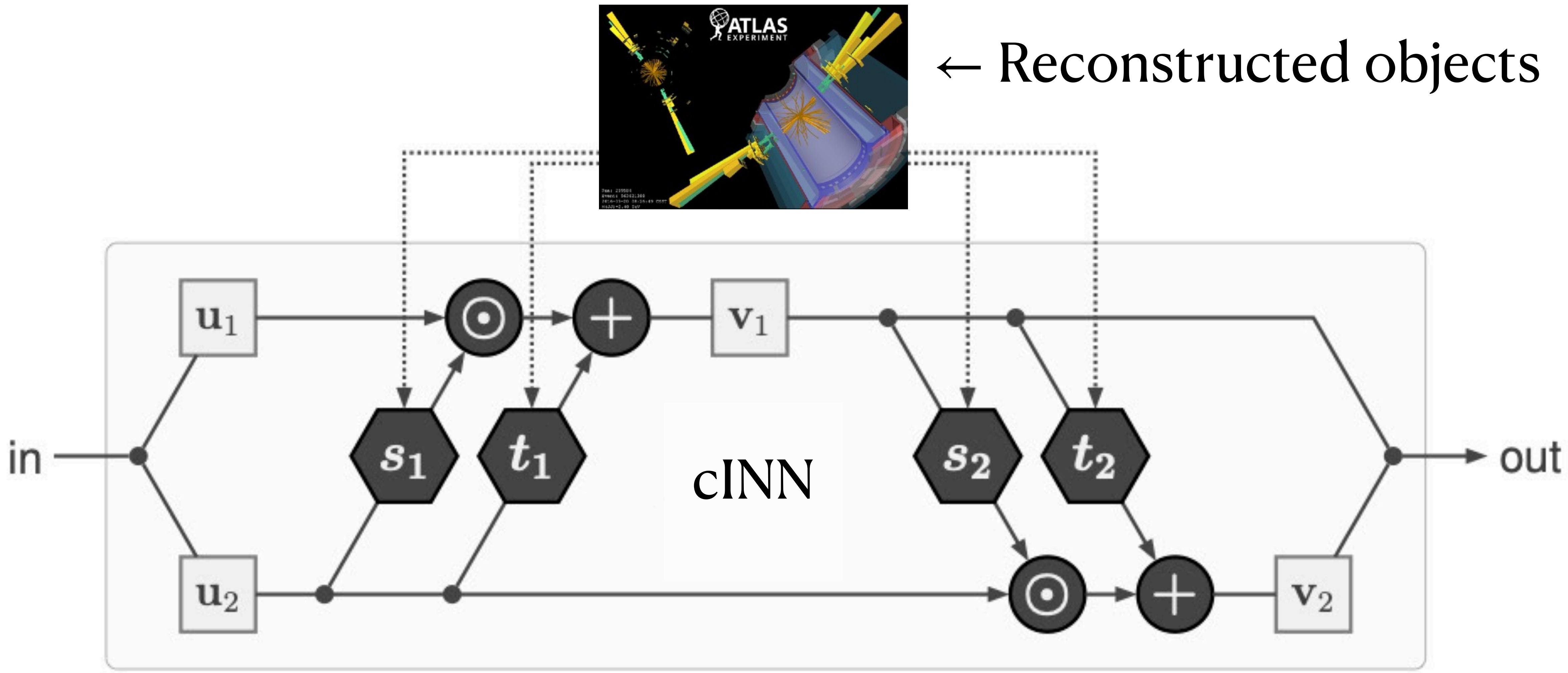
- ☐ Highdimensional
- ☐ Bin independent
- ☐ Statistically well defined

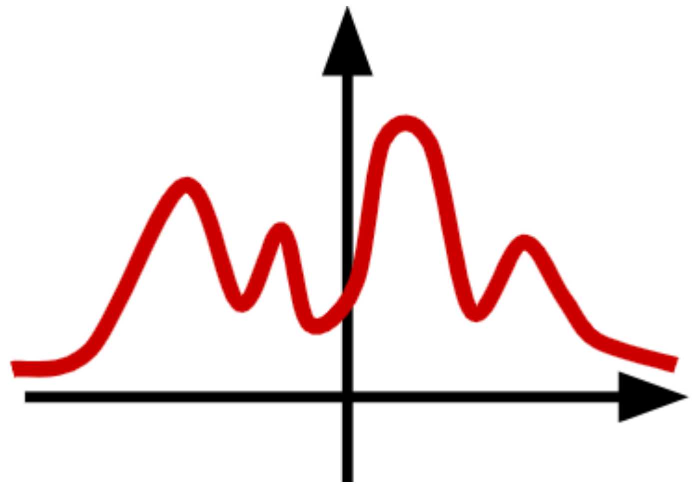
cINN unfolding

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



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





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

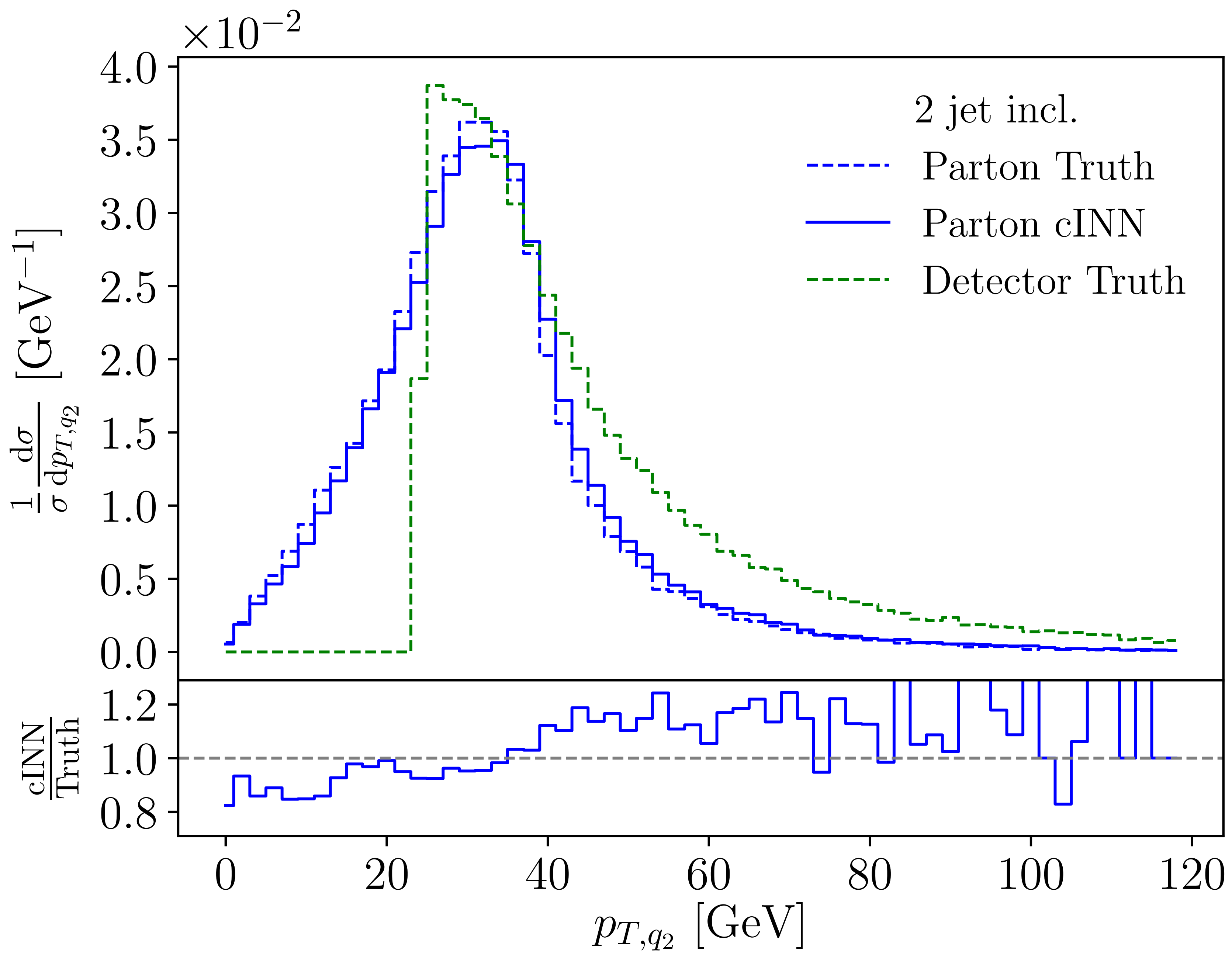
Training

Unfolding

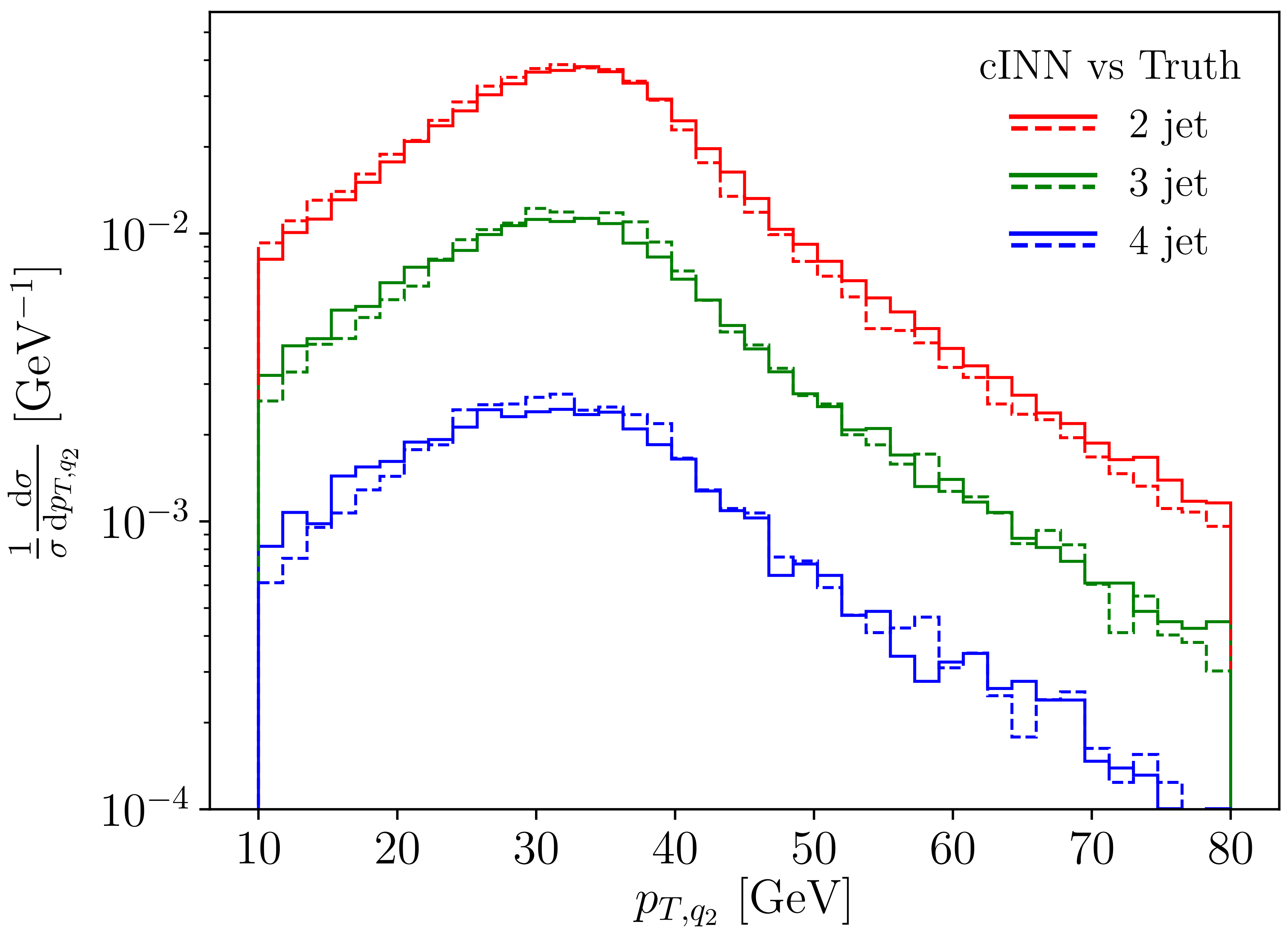
Inverting inclusive distributions

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

Training on inclusive dataset



Evaluate exclusive 2/3/4 jet events

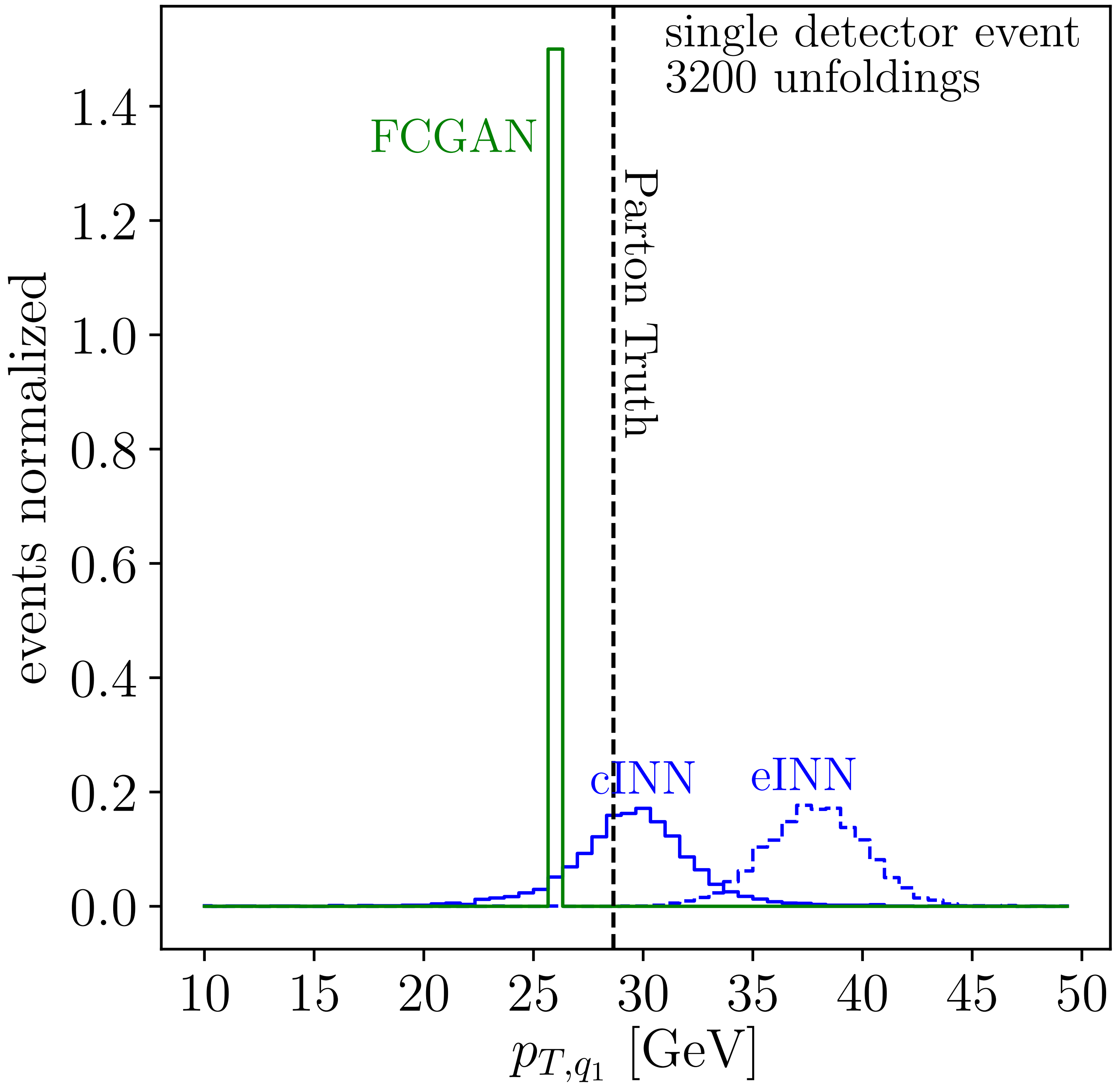


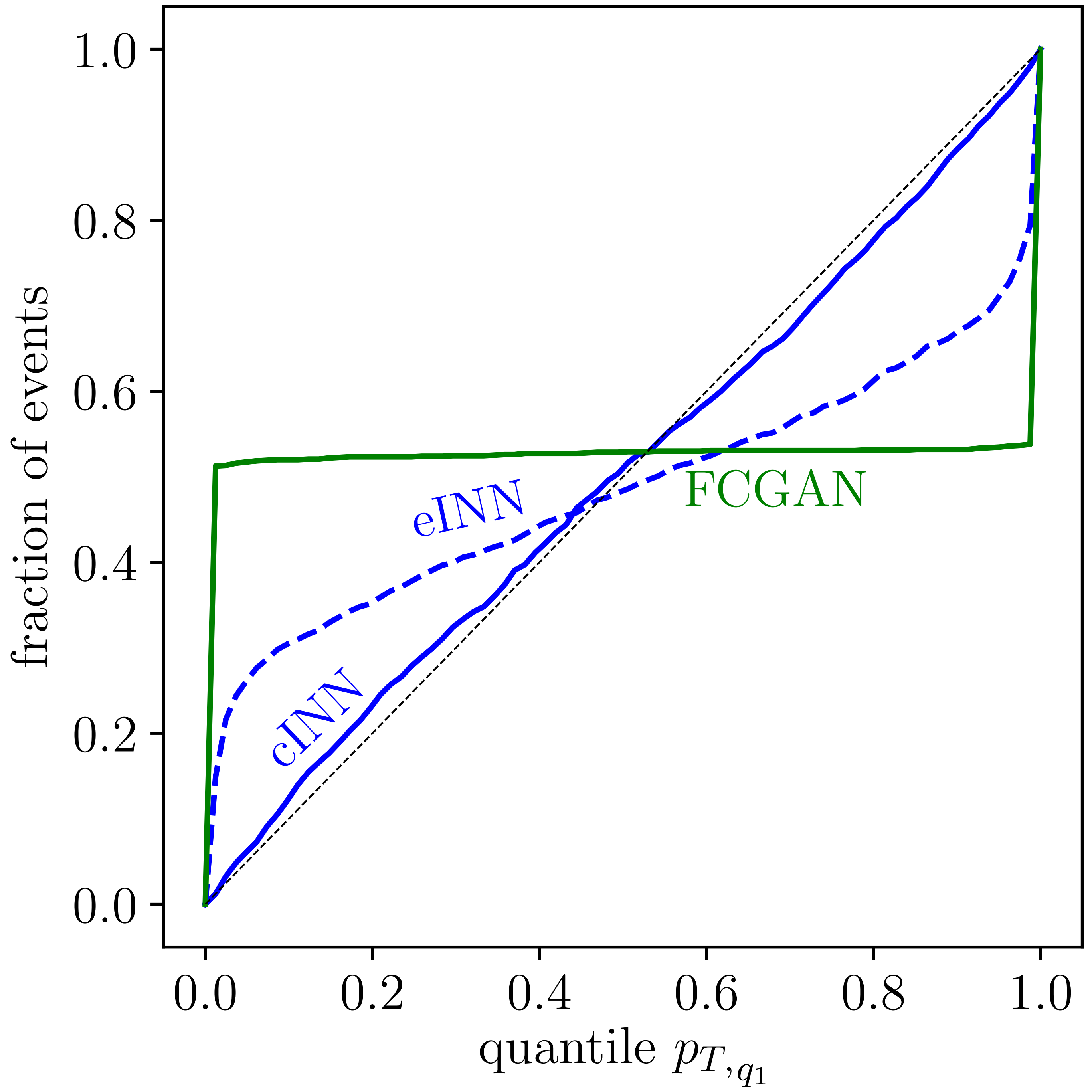
- ☑ High-dimensional
- ☑ Bin-independent
- ☐ Statistically well defined ?

M. Bellagente et al. [2006.06685]

Event-wise unfolding

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



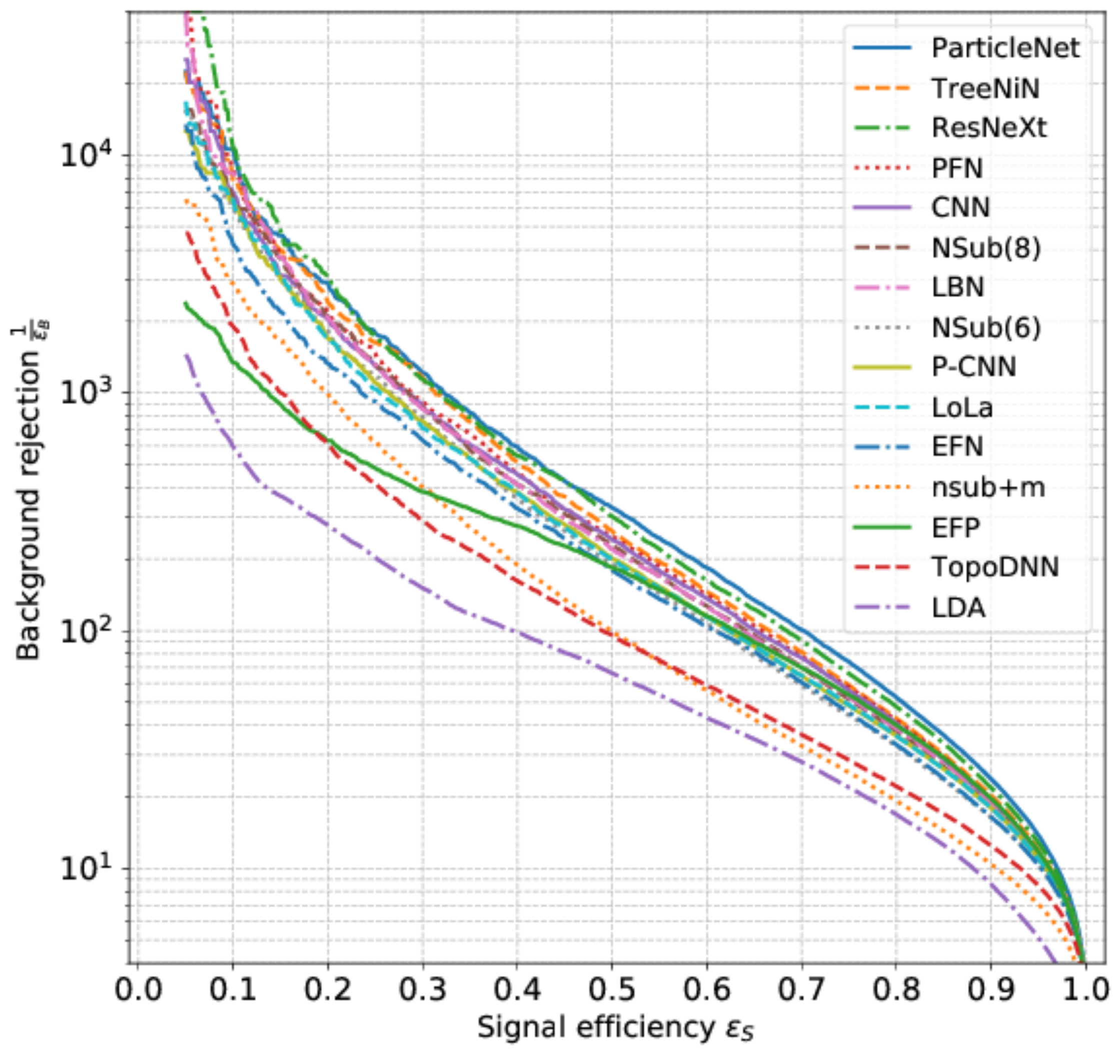


- ☑ High-dimensional
- ☑ Bin-independent
- ☑ Statistically well defined

M. Bellagente et al. [2006.06685]

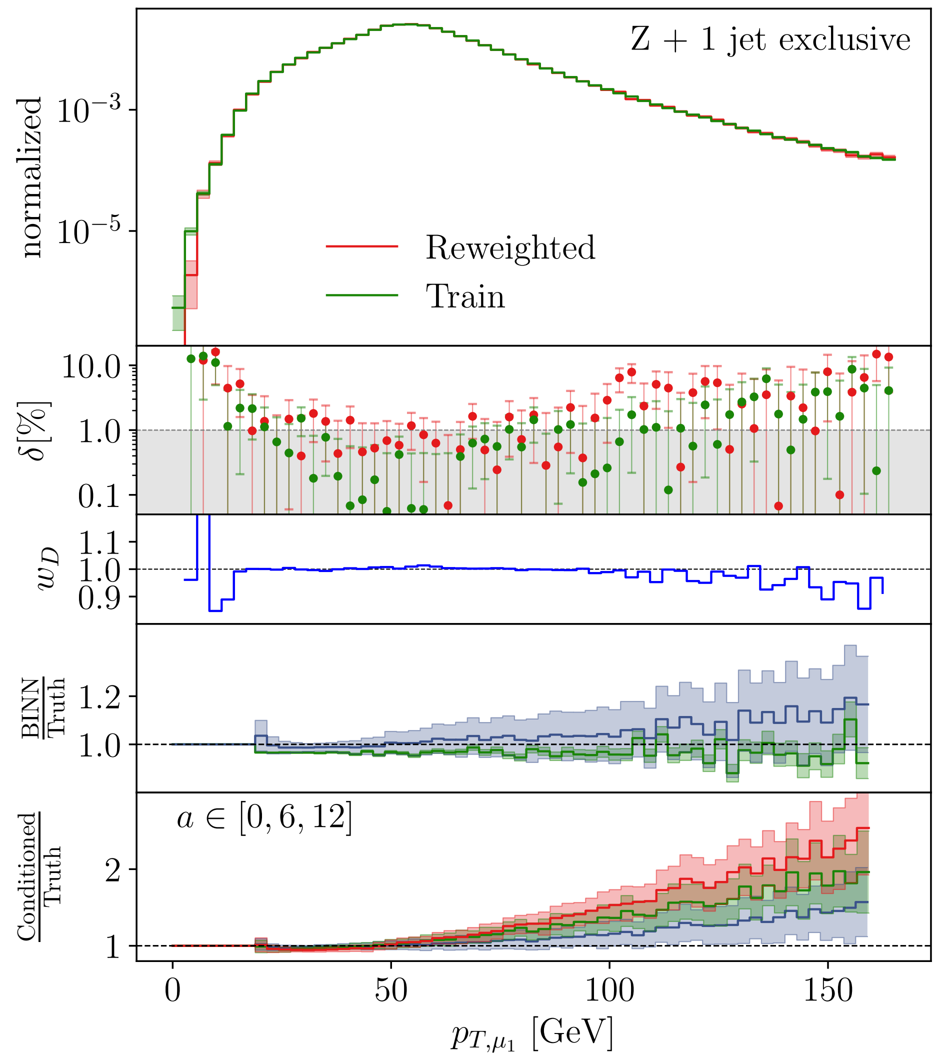
ML examples and their uncertainties

Classification



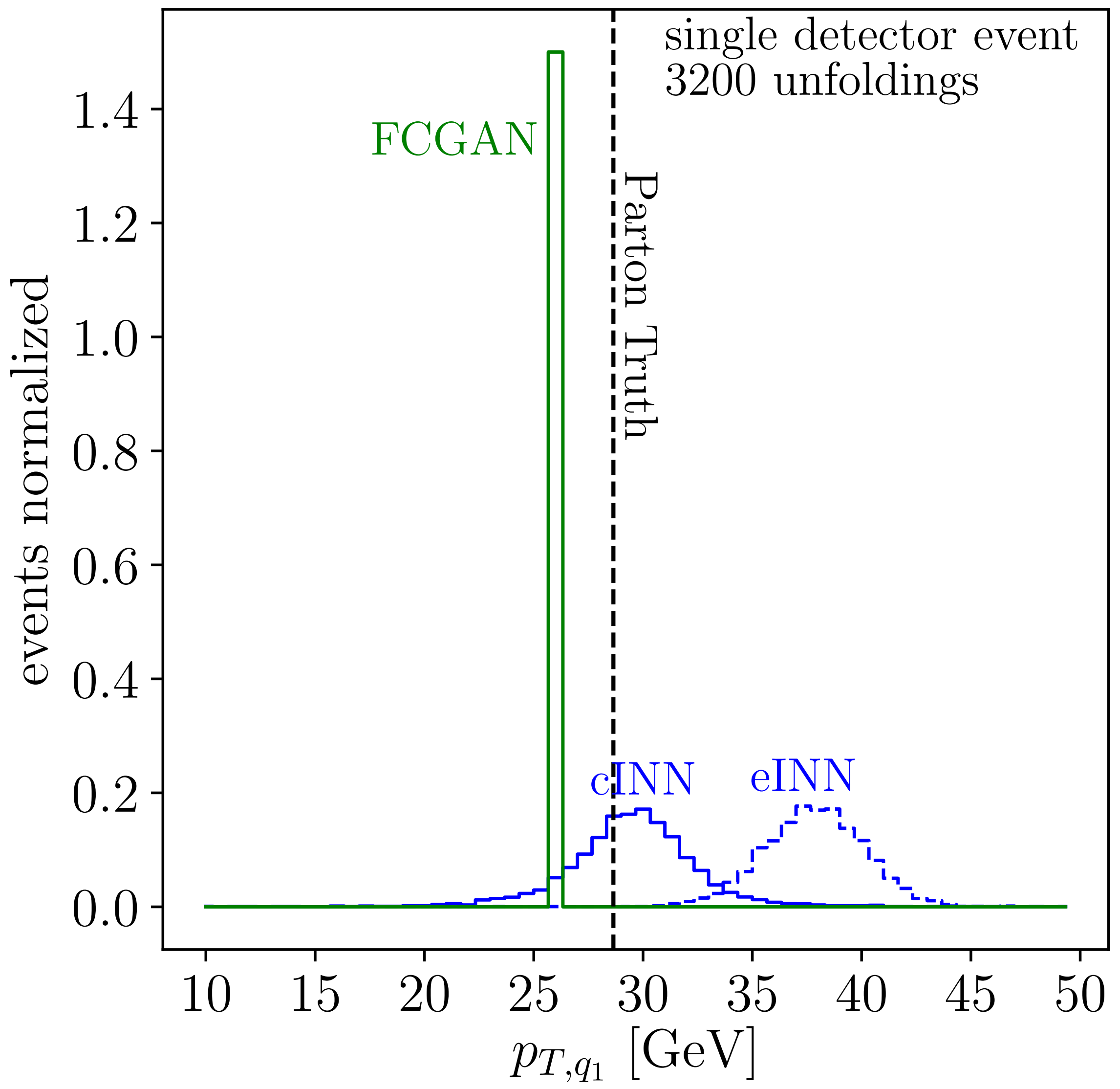
No uncertainty needed

Simulations



Amplitude estimation -> yes (BNN)
Loop integration -> no
Phase space sampling -> no
Data compression -> yes (BNN & classifier)

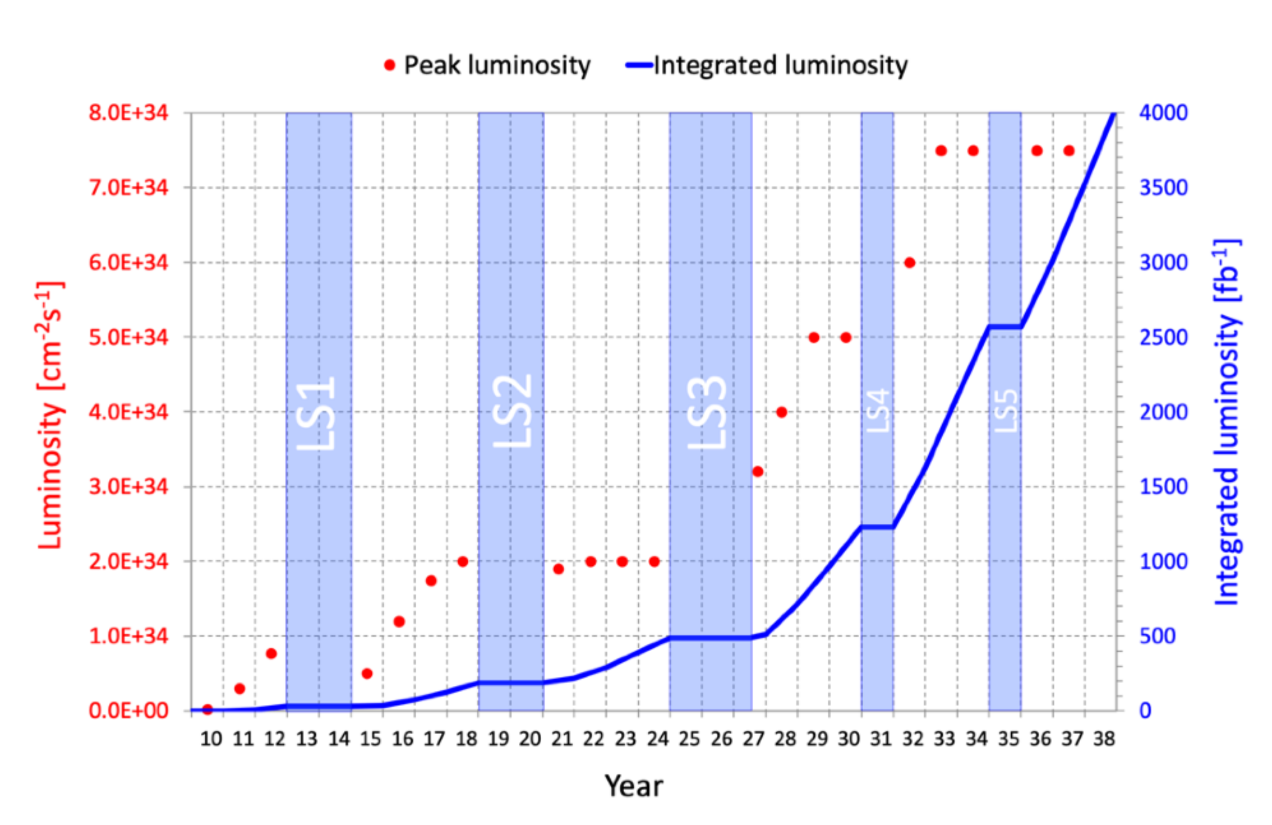
Unfolding



Probability distributions
from generative networks
Uncertainties on pdfs?

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

ML can help tackle all of these problems. Uncertainties included.